SVM for Classification of Spam Email Messages

EE5904/ME5404 Part II: Project 1

Report due on 25 April 2025, 23:59 Singapore time

Outline

- Project description
- 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:**
    - 4061 samples of email metadata taken from UC Irvine Machine Learning Repository
    - 57 features per sample
    - Label: +1 (spam), -1 (non-spam) | 1 and 0 in original dataset
    - 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
```

```
Class Distribution:
   Spam     1813   (39.4%)
   Non-Spam  2788   (60.6%)
```

Project Description

- The dataset is divided into 3 subset according to Project Requirement:
  - Training set: 2000
  - Test set: 1536
  - Evaluation set (not given): 600
- Each dataset: (1) feature, (2) label

Name	Value
train_data	57x2000 double
train_label	2000x1 double

Name	Value
eval_data	57x600 double
eval_label	600x1 double

Name	Value
test_data	57x1536 double
test_label	1536x1 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 $i \in \{1, 2, \ldots, N\}$,

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

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

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

Evaluate

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

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

Task 1: Data

- Training set (with 2000 samples)

Given: "train.mat"
  - feature (57 x 2000)
  - label (2000 x 1)

57 attributes

Feature of a sample:

0, 0.640, 0.640, 0, 0.320, 0, 0, 0, 0, 0, 0, 0.640, 0, 0, 0, 0.320, 0, 1.290, 1.930, 0, 0.960, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.778, 0, 0, 3.756, 61.000, 278.000

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

Task 1: Training set

- Import the training set (i.e. train.mat)
  - train_data (57x2000)
  - train_label (2000x1)
- Preprocess the "data" (various methods can be used including normalization and standardization [CHOOSE ONE METHOD])
  - **Normalize** the data: rescale the individual sample x such that $||x|| = 1$
  - **Standardize** the data: transform each feature by removing the mean value of each feature and then dividing by each feature's standard deviation

  - Please ensure the "label" is mapped into the set of {-1, 1}.

Task 1: Kernels

- Hard-margin SVM with the linear kernel
$$\boldsymbol{K}(\boldsymbol{x_1}, \boldsymbol{x_2}) = \boldsymbol{x_1^T x_2}$$
- Hard-margin SVM with a polynomial kernel
$$\boldsymbol{K}(\boldsymbol{x_1}, \boldsymbol{x_2}) = (\boldsymbol{x_1^T x_2} + 1)^p$$
- Soft-margin SVM with a polynomial kernel
$$\boldsymbol{K}(\boldsymbol{x_1}, \boldsymbol{x_2}) = (\boldsymbol{x_1^T x_2} + 1)^p$$

Task 1: Hard and Soft Margins

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

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

Task 1: Calculate α

- How To solve for α_i

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

- Use quadprog function

quadprog

Quadratic programming

Finds a minimum for a problem specified by

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

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

f, *lb*, and *ub* can be passed 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: quadprog

Quadratic programming

Not in use

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

Finds a minimum for a problem specified by

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

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

f, *lb*, and *ub* can be passed as vectors or matrices; see Matrix Arguments.

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

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

Soft Margin

Convert the problem from "Max" to "Min"

- Max $Q(\alpha) \rightarrow$ Min $-Q(\alpha)$

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

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

$$\min_x \frac{1}{2}x^T Hx + f^T x \quad \begin{cases} H_{ij} = d_i d_j K(x_i, x_j) \\ f = (-1, -1, \ldots, -1)^T \end{cases}$$

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

Task 1: quadprog

Quadratic programming

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

- Hard-margin SVM with the Linear kernel

$$\boldsymbol{K(x_1, x_2) = x_1^T x_2}$$

For illustration only:

- H$(i, j) = d_i d_j x_i^T x_j$;
- f = -ones(2000,1);
- Aeq = train_label';
- Beq = 0;
- lb = zeros(2000,1);
- ub = ones(2000,1)*C;
- x0 = [ ];
- options = optimset('LargeScale','off', 'MaxIter',1000);

- Hard Margin $\alpha_i \geq 0$
  - C = +∞ (In theory)
  - C = Large value (In practice, e.g. 10^6)

Task 1: quadprog

Quadratic programming

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

- Hard-margin SVM with the Polynomial kernel

$$K(\boldsymbol{x_1}, \boldsymbol{x_2}) = (\boldsymbol{x_1^T x_2} + 1)^p$$

For illustration only:

- H$(i,j) = d_i d_j\,(x_i^T x_j + 1)^p$;
- f = -ones(2000,1);
- Aeq = train_label';
- Beq = 0;
- lb = zeros(2000,1);
- ub = ones(2000,1)*C;
- x0 = [ ];
- options = optimset('LargeScale','off' , 'MaxIter',1000);

- Hard Margin $\alpha_i \geq 0$
  - C = +∞ (In theory)
  - C = Large value (In practice, e.g. 10^6)

Task 1: quadprog

Quadratic programming

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

- Soft-margin SVM with the Polynomial kernel

$$K(\boldsymbol{x_1}, \boldsymbol{x_2}) = (\boldsymbol{x_1^T x_2} + 1)^p$$

For illustration only:

- $H(i,j) = d_i d_j\, (\underline{x_i^T x_j + 1})^p$;
- f = -ones(2000,1);
- Aeq = train_label';
- Beq = 0;
- lb = zeros(2000,1);
- ub = ones(2000,1)*C;
- x0 = [ ];
- options = optimset('LargeScale','off', 'MaxIter',1000);

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

Task 1: Selection of Support Vectors

Based on KKT conditions

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

However in practice, α_i = small value

How to decide ?

- Choose an appropriate threshold (e.g.1e-4) to determine the support vectors

Task 1:Discriminant function g(x)

Hard Margin SVM with linear kernel

Maximizing $Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, \alpha_j \, d_i \, d_j \, \underline{\mathbf{x}_i^T \mathbf{x}_j}$ Subject to $\sum_{i=1}^{N} \alpha_i d_i = 0$, $\alpha_i \geq 0$

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

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

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

Soft Margin SVM with linear kernel

Maximizing $Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, \alpha_j \, d_i \, d_j \, \underline{\mathbf{x}_i^T \mathbf{x}_j}$ Subject to $\sum_{i=1}^{N} \alpha_i d_i = 0$, $0 \leq \alpha_i \leq C$

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

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

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

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

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

2. 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:Discriminant function g(x)

Soft Margin SVM with nonlinear kernel

Determine $b_\circ$ in

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

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 2: Data

- Test set (with 1536 samples)

Given: “test.mat”

  - feature (57 x 1536)
  - label (1536 x 1)

57 attributes

Feature of a sample:

0, 0.640, 0.640, 0, 0.320, 0, 0, 0, 0, 0, 0, 0.640, 0, 0, 0, 0.320, 0, 1.290, 1.930, 0, 0.960, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.778, 0, 0, 3.756, 61.000, 278.000

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

Task 2: Testing set

- Import the testing set (i.e. test.mat)
- Preprocess the "data" (various methods can be used including normalization and standardization [CHOOSE ONE METHOD])
  - **Normalize** the data: rescale the individual sample x such that $||x|| = 1$
  - **Standardize** the data: transform each feature by removing the mean value of each feature and then dividing by each feature's standard deviation

  - Please ensure the "label" is mapped into the range of {-1, 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 $\mathbf{x}_{\text{new}}$

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

<u>For illustration only</u>:

$$\mathbf{g}(\mathbf{x}_{\text{test}}) = \sum_{i=1}^{N} \boldsymbol{\alpha_{o,i}}\, \boldsymbol{d_i} \boldsymbol{K}(\boldsymbol{x_i}, \boldsymbol{x_{test}}) + \boldsymbol{b_o}$$

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 $\mathbf{x}_{\text{new}}$

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

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	$C=0.1$	$C=0.6$	$C=1.1$	$C=2.1$	$C=0.1$	$C=0.6$	$C=1.1$	$C=2.1$
$p=2$	?	?	?	?	?	?	?	?
$p=3$	?	?	?	?	?	?	?	?
$p=4$	?	?	?	?	?	?	?	?
$p=5$	?	?	?	?	?	?	?	?

TABLE I

RESULTS OF SVM CLASSIFICATION.

Fill in

Task 3 : Data

- Evaluation set (with 600 samples)

Not given: “eval.mat”

  - **feature: “eval_data” (57 x 600)**
  - **label: “eval_label” (600 x 1)**

Task 3 : Evaluation

- Design a SVM of your own
  - Hard or Soft Margin ?
  - Linear or Polynomial Kernel ?
  - What are the p and C values ?
- To classify the 600 samples in the evaluation set

Produce the best performance

Not given: eval.mat
eval_data (57x 600)
eval_label (600 x 1)

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

Task 3 : Evaluation

- You can assume the “train.mat” and “eval.mat” are loaded into the workspace

- Your code should be able to generate **“eval_predicted” (600 x 1)**

Task 3 : Evaluation

- Please name your M file for Task 3 as **“svm_main”**
- Do not clear any variables in the “svm_main” script
- Before submitting your code, please **ensure that it works** by testing it with the training and test set

Important Notes : All tasks

Procedure to build SVM

- Choose a suitable Kernel

  Linear/Nonlinear ?

- Choose C

  Hard/Soft Margin ?

- Solve α_i

  Quadratic Programming

- Determine discriminant function

  g(x)

Important Notes : All tasks

- Hard Margin
  - $C = +\infty$ (In theory)
  - C = Large value (In practice, e.g. 10^6)

Important Notes : All tasks

- Selection of support vectors
  - Select an appropriate threshold (e.g. 1e-4) for choosing the support vectors

Important Notes : Submission

- Submit <u>all your codes </u>that you have implemented for the entire project

- Make sure your codes work without errors.

- All codes should be executable with the given data sets in the workspace without any additional inputs

Report due on 25 April 2025, 23:59 Singapore time

Q & A