SVM for Classification of Spam Email Messages

EE5904/ME5404 Part II: Project 1

Report due on April 26, 2019

Manuel Del Rosario Jr
mrdjr@u.nus.edu

Outline

- Project description
- Recap
- Task 1: Train
- Task 2: Test
- Task 3: Evaluate
- Important Notes

Project Description

Goal

Implement SVM to classify spam or no spam for the Spam Email Data Set[1]

- Spam Email Data Set is a collection of 4601 samples of email metadata taken from **UC Irvine Machine Learning Repository**

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, 0.778, 0, 0, 3.756, 61.000, 278.000

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 occurences) / total characters in e-mail

1 continuous real [1,...] attribute of type capital_run_length_average
= average length of uninterrupted sequences of capital letters

1 continuous integer [1,...] attribute of type capital_run_length_longest
= length of longest uninterrupted sequence of capital letters

1 continuous integer [1,...] attribute of type capital_run_length_total
= sum of length of uninterrupted sequences of capital letters
= total number of capital letters in the e-mail

1 nominal {0,1} class attribute of type spam
= denotes whether the e-mail was considered spam (1) or not (0), i.e. unsolicited commercial e-mail

[1] http://archive.ics.uci.edu/ml/datasets/spambase

Project Description

- The data is divided into 3 sample groups
  1. Training set
  2. Test set
  3. Evaluation set (**Not provided**)

eval_data	57x600 double
eval_label	600x1 double
test_data	57x1536 double
test_label	1536x1 double
train_data	57x2000 double
train_label	2000x1 double

- Each dataset: (1) feature, (2) label

  (1) 57 attributes / features:

  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

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

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 occurences) / total characters in e-mail

1 continuous real [1,...] attribute of type capital_run_length_average
= average length of uninterrupted sequences of capital letters

1 continuous integer [1,...] attribute of type capital_run_length_longest
= length of longest uninterrupted sequence of capital letters

1 continuous integer [1,...] attribute of type capital_run_length_total
= sum of length of uninterrupted sequences of capital letters
= total number of capital letters in the e-mail

1 nominal {0,1} class attribute of type spam
= denotes whether the e-mail was considered spam (1) or not (0), i.e. unsolicited commercial e-mail

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 $i \in \{1, 2, \ldots, N\}$,

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

[1]: Compute Discriminant Functions using the specified kernels

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

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

[2]: Classify the data sets (train and test) using the functions from [1] and measure the performance

Evaluate

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

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

[3]: Design your own SVM, train, and test the performance

Recap : Hard Margin

Primal problem

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

Find : $\mathbf{w}$ and b

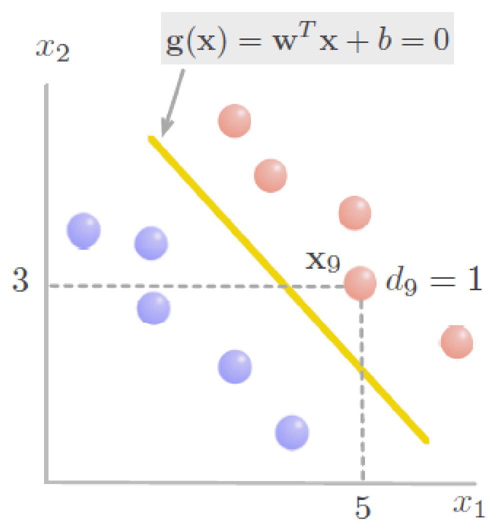
Minimizing : $f(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$

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

In primal problem

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

Solve → Optimal hyperplane



A hyperplane, denoted by $(\mathbf{w}, b)$, can be expressed as

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b = 0$$

Hyperplane classifies a given $\mathbf{x}_i$ with

$$\mathrm{sgn}\left[g(\mathbf{x}_i)\right] = \begin{cases} +1 & \text{if} \quad g(\mathbf{x}_i) > 0 \\ -1 & \text{if} \quad g(\mathbf{x}_i) < 0 \end{cases}$$

Recap : Hard Margin

Primal problem Lecture Slide 75

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

Find : $\mathbf{w}$ and b

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

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

Alternative formulation using method of **Lagrange multipliers**

Finding optimal hyperplane (dual problem) Lecture Slide 81

Given : $S = \{(\mathbf{x}_i, d_i)\}$

Find : Lagrange multipliers $\{\alpha_i\}$

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

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

(2) $\alpha_i \geq 0$

Recap : Hard Margin

Primal problem Lecture Slide 75

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

Find : $\mathbf{w}$ and b

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

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

Alternative formulation using method of **Lagrange multipliers**

Finding optimal hyperplane (dual problem) Lecture Slide 81

Given : $S = \{(\mathbf{x}_i, d_i)\}$

Find : Lagrange multipliers $\{\alpha_i\}$

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

Linear Kernel

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

(2) $\alpha_i \geq 0$

Only unknowns are α_i

For a support vector, $\alpha_i \neq 0$

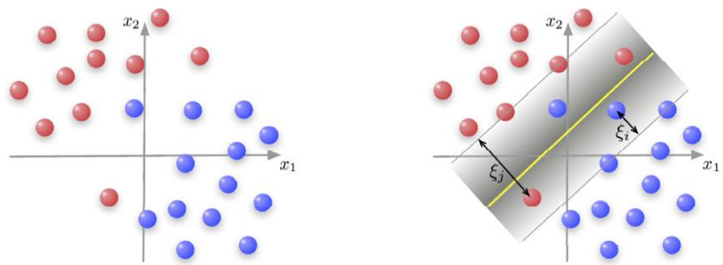
Apply (Karush Kuhn–Tucker) KKT conditions

Recap : Soft Margin

Dealing with non-separable patterns

Lecture Slide 96

1. Find optimal hyperplane to minimize classification error



Dual problem (with soft margin)

Lecture Slide 102

Find : α_i

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

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

Linear Kernel

Soft Margin

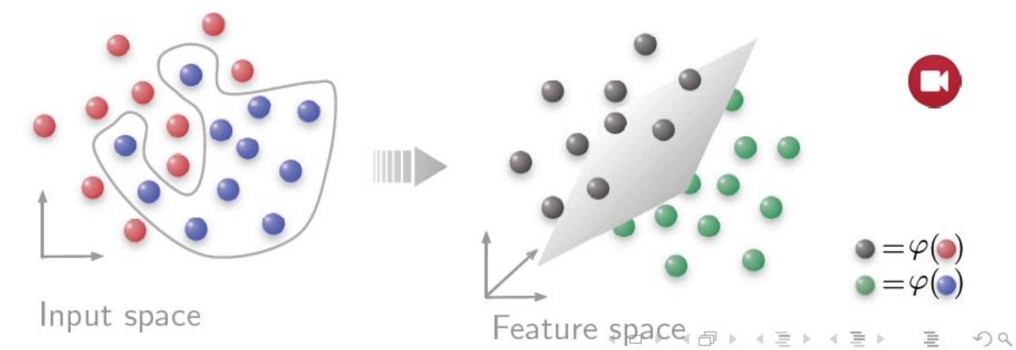
Lecture Slide 98

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

Recap: Soft Margin and Transformation

Dealing with non-separable patterns

Lecture Slide 96



Lecture Slide 117

Dual problem with soft margin and transformation

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

Soft Margin

Nonlinear Kernel

Recap : Summary

Lecture Slide 125

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

2 Choose a value for C

3 Solve for $\alpha_{o,i}$

4 Determine b_o in

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

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

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

Kernel

Soft Margin

Quadratic Programming

KKT conditions For a support vector, $\alpha_i \neq 0$

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

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

Hard Margin	Soft Margin
$\alpha_i \geq 0$	$0 \leq \alpha_i \leq C$

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, 0.778, 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)

- Preprocess the "data" (various methods can be done e.g. **Sample Scaling** or **Standardization [CHOOSE ONE!]**)[a,b]
  - **Scale** the "data" : rescale individual sample x such that $||x|| = 1$.
  - **Standardize** the "data": transform each feature by removing the mean value of each feature then dividing by each feature's standard deviation.

- 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

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

- 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 $\boxed{\alpha_i \geq 0}$
  - C = $+\infty$ (In theory)
  - C = <u>Large value (In practice, e.g. 10^6)</u>

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

Task 1: quadprog

Quadratic programming

- To solve for α_i

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

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

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

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

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.

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

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

Task 1: quadprog

Quadratic programming

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

Soft Margin

Finds a minimum for a problem specified by

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

Not in use

A = [ ];
b = [ ];

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.

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

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

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

```
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

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

For illustration only:

- $\mathrm{H}(i,j) = d_i d_j x_i^T x_j$;
- 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

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

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 = $+\infty$ (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$(i,j) = d_i d_j \, (x_i^T x_j + 1)^p$;
- 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, $\alpha_i \neq$ 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:

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

Lecture Slide 87

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

Task 1:Discriminant function g(x)

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:

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

Lecture Slide 106,107

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, 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 done e.g. Sample Scaling or Standardization [CHOOSE ONE!])[a]
  - Scale the "data" : rescale individual <u>sample x</u> such that $||x|| = 1$.
  - Standardize the "data": using the <u>mean and variance of each feature</u> from your <u>**training set**</u>. Transform each feature in the same manner with the training data.

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

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

<u>For illustration only</u>:

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

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

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

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

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 (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

- Prepare the code so that it could handle the evaluation data set
  - Note: the **eval_data is a (57x600) matrix**

- 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

- Preprocess your data

  Sample Scaling / Mean Normalization / Standardization/ Rescaling / etc. ?

- Use the training set statistics to preprocess the other data sets

- Check Mercer Condition

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 all your codes 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 April 26, 2019

Q & A

mrdjr@u.nus.edu