EE5904/ME5404 Neural Networks Part II Project Report

AY2021/2022, Semester 2

Department of Electrical and Computer Engineering

SVM for Classification of Spam Email Messages

Jin Lexuan A0232696W

e0724495@u.nus.edu

A report submitted in fulfilment of the requirements of
the National University of Singapore for
the EE5904/ME5404 Neural Networks module

March 31, 2022

1 Data pre-processing

The data pre-processing method is standardization of the data: Transform each feature by removing the mean value of each feature and then dividing by each feature's standard deviation. This is a z-score standardization, the mean and the standard deviation of the processed data is 0 and 1 respectively, here is the formula of processing:

$$standardization : x^* = \frac{x - \bar{x}}{\sigma}$$

$$mean : \bar{x} = \frac{1}{n}\sum_{i}^{n} x_i$$

$$standard\ deviation : \sigma = \sqrt{\frac{1}{n}\sum_{i}^{n}(\bar{x} - x_i)^2}$$

2 Admissibility of the kernels

To judge the admissibility of a kernel is to choose an expression for $K(\cdot, \cdot)$. If this expression satisfies the Mercer's Condition, then it can be used as a kernel.

Mercer's Condition: For training set, $S = (\mathbf{x}_i, d_i), i = 1, 2, \cdots, N$, the Gram matrix is shown in Figure 1 is positive semi-definite (i.e., its eigenvalues are non-negative).

$$\mathbf{K} = \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & \dots & K(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ K(\mathbf{x}_N, \mathbf{x}_1) & \dots & K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \in R^{N \times N}$$

Figure 1: Mercer's Condition

In this project, according to the calculation of MATLAB, when p equals 5, it is not admissible as a kernel. When p equals to $1, 2, 3, 4$, kernels are appropriate. However, the accuracy of inadmissible kernel situation is also calculated as comparison.

3 Dual Problem

The target of task 1 is to solve dual problem with *quadprog* function in MATLAB. The dual problem are listed in the below subsection respectively. The calculation of $\boldsymbol{\alpha_i}$ is based on them.

3.1 Hard margin 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:

$$\sum_{i=1}^{N} \alpha_i d_i = 0,\ \alpha_i \geq 0$$

3.2 Hard margin with polynomial kernel

Maximize:

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

Subject to:

$$\sum_{i=1}^{N} \alpha_i d_i = 0,\ \alpha_i \geq 0$$

3.3 Soft margin with polynomial kernel

Maximize:

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

Subject to:

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

In this project, p equals to $\{1, 2, 3, 4, 5\}$, C equals to $\{0.1, 0.6, 1.1, 2.1\}$

4 Existence of optimal hyperplanes

The statement of existence of optimal hyperplanes is in Table 1. It depends on the result of *quadprog* function.

Type of SVM	Training accuracy			
Hard margin with linear kernel	No			
Hard margin with polynomial kernel	p = 2	p = 3	p = 4	p = 5
	Yes	Yes	Yes	No
Soft margin with polynomial kernel	C = 0.1	C = 0.6	C = 1.1	C = 2.1
p = 1	Yes	Yes	Yes	Yes
p = 2	Yes	Yes	Yes	Yes
p = 3	Yes	Yes	Yes	Yes
p = 4	Yes	Yes	Yes	Yes
p = 5	Yes	Yes	Yes	Yes

Table 1: Existence of optimal hyperplanes

After solving of these dual problems, according to the solver of MATLAB, the hard margin with linear kernel situation and the hard margin with polynomial kernel when p = 5 does not have optimal hyperplanes. The solver cannot find the solution until 1000 iteration, it stops prematurely. For other situation, optimization completed because the objective function is non-decreasing in feasible directions, to within the value of the optimality tolerance, and constraints are satisfied to within the value of the constraint tolerance. The optimal hyperplanes exist.

5 Comments on results

After the implementation of codes, the corresponding $\boldsymbol{\alpha}$ value and bias item can be obtained. The the predicted label of training set and testing set is acquired accordingly. The results under each condition are collected in the Table 2 below.

Type of SVM	Training accuracy				Testing accuracy			
Hard margin with linear kernel	93.50%				93.23%			
Hard margin with	p = 2	p = 3	p = 4	p = 5	p= 2	p = 3	p = 4	p = 5
polynomial kernel	100%	100%	100%	41.15%	85.87%	85.94%	85.94%	40.04%
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	92.65%	92.50%	92.65%	92.90%	92.97%	92.71%	92.97%	93.23%
p = 2	98.80%	99.50%	99.50%	99.55%	89.39%	89.45%	88.48%	88.09%
p = 3	99.65%	99.80%	99.85%	99.90%	89.13%	88.48%	88.35%	88.35%
p = 4	99.90%	99.95%	100%	100%	87.89%	86.78%	86.00%	86.00%
p = 5	99.05%	98.85%	98.85%	98.90%	87.04%	86.00%	86.00%	86.07%

Table 2: Results of SVM classification

With the combination of theories and practical situation, it can be concluded that:

- For hard margin with linear kernel situation, it has a good performance that the accuracy of both training set and testing set are over 93%.
- For hard margin with polynomial kernel, according to mercer condition, p = 5 is a not admissible situation and the accuracy verifies it with both 40% on training set and testing set. For admissible situation, it outperforms on training set with 100% accuracy compared with 85% accuracy on testing set. The big difference of two accuracy indicates the over-fitting.
- For soft margin with polynomial kernel, when p equals 1, the accuracy of both training set and testing set is about 93%. The performance is fine. With the increasing of C value, the accuracy also grows slightly.

- For other p value for soft margin situation, the accuracy of training set is very high, almost 100%, while the accuracy of testing set is no more than 90%, around 88%. This phenomenon also suggests the over-fitting. Unlike p equals to 1 circumstance, the accuracy of testing set decreases with the growth of C. The reason is that, C is the tolerance of misclassification, that higher C contributes to over-fitting.
- Differently, the p equaling 5 situation does not result in the massive decreasing the accuracy in hard margin with polynomial kernel situation. The penalty coefficient C protects the kernel performance.
- Since the impact of different selection of kernel function and hyperparamters can be deducted, for further study, suitable choice according to the specific data of them is necessary to guarantee the performance.

6 Discussion on design decisions

In this self-design part, for the kernel, radial basis function is selected. For this kernel, the dual problem is shown below:

Maximize:

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j exp(-\gamma * |\mathbf{x}_i - \mathbf{x}_j|^2)$$

Subject to:

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

The hyper-parameters of radial basis function kernel are γ and C. γ is decided by σ according to the formula:

$$\gamma = \frac{1}{2\sigma^2}$$

If γ is set too large, it will easily lead to over-fitting. Because σ will be very small, and the Gaussian distribution will grow tall and thin, it will only act closely to the support vector samples, and the classification effect for unknown samples is very poor, but the training accuracy can be very high. If let Infinitely small, then theoretically, SVM with Gaussian kernel can fit any nonlinear data, but it is easy to over-fit.Conversely, the smaller the gamma, the less fitting it is.

C is the penalty coefficient, that is, the tolerance for misclassification of samples. The higher the C, the more intolerant of errors and easy over-fitting. The smaller C is, the easier it is to under-fit. If C is too large or too small, the generalization ability becomes poor.

According to the above design rules, the C is chosen as 400 and γ is chosen as 30. The accuracy of 3 data set are shown in Table 3. The fake evaluation set is 600 random selected data from test set.

	Train	Test	Fake eval
Accuracy	94.05%	93.42%	93.83%

Table 3: Result of the design