NUS National University of Singapore | Department of Electrical & Computer Engineering
Faculty of Engineering

SVM for Classification of Spam Email Messages
—— EE5904/ME5404 Neural Networks
Part II Project1 Report

Student Name: Gu Jiapan

Student Number: A0186538N

Email Address: e0320776@u.nus.edu

Task 1&2:

➢ Implementation:

Several SVM models with different kernels and parameters are implemented in 3 M-files. Each file is focused on as described in their file name. There are three types of SVM in this project: hard margin with linear kernel, hard margin with polynomial kernel and soft margin with polynomial kernel.

The key parameter for polynomial kernel is the order of its polynomial (*p*). Another key parameter for soft margin is the coefficient of error penalty (*C*), which reflects cost of violating constraints. A large *C* generally leads to smaller margin but also fewer misclassification of training data, while a small *C* generally leads to larger margin but more misclassification of training data.

As we know, an SVM problem can be considered as a dual problem as following.

Given: $S = \{(x_i, d_i)\}$

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

Maximize: $$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j K(x_i, x_j)$$

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

If it is a hard margin SVM, $C \to \infty$. $K(x_1, x_2)$ is the kernel function. For linear kernel, $K(x_i, x_j) = x_i^T x_j$. For polynomial kernel, $K(x_i, x_j) = (x_i^T x_j + 1)^p$

In order to solve this SVM problem in MATLAB, first standardize the training data by removing the mean value of each feature then dividing by each feature's standard deviation. Similarly, using the mean and variance of each feature from the training set to transform each feature of test data in the same manner with training data.

Before the next step, we need to check the Mercer condition of each kernel. Because a kernel satisfying the Mercer condition ensures the existence of a global optimum for the resulting optimization problem. For training set, if the Gram matrix is positive semi-definite, i.e. all its eigenvalues are nonnegative, the kernel satisfies the Mercer condition.

Then, we can do the quadratic programming using the kernel and other parameters to solve the Lagrange multipliers. In MATLAB, the function "x = quadprog (H, f, A, b, Aeq, beq, lb, ub, x0, options)" can solve the following problem using the optimization options specified in options.

$$\min_{x}\left(\frac{1}{2}x^{T}Hx+f^{T}x\right) \text{ such that } \begin{cases} A\bullet x\le b \\ Aeq\bullet x=beq \\ lb\le x\le ub \end{cases}$$

By setting its parameters correspondingly, we can convert it to solve our goal problem. After using it to get Lagrange multipliers α_i, based on KKT conditions, for a support vector, they can not equal a small value. The threshold is chosen as 1e-4.

Then, we can calculate the weights and bias as follow.

$$w_o=\sum_{i=1}^{N}\alpha_{o,i}d_ix_i, \quad b_{o,i}=\frac{1}{d_i}-w_o^Tx_i.$$

Based on the discriminant function as follow, we can use sign function to do the classification for each piece of test data.

$$g(x)=\sum_{i=1}^{N}\alpha_{o,i}d_iK(x,x_i)+b_o$$

Finally, statistic and output the accuracy for both training data and test data separately.

- **Results:**

Type of SVM	Training accuracy	Test accuracy
Hard margin with linear kernel	93.30%	92.84%

Hard margin with polynomial kernel	P = 2	P = 3	P = 4	P = 5	P = 2	P = 3	P = 4	P = 5
	90.70%	86.05%	--	--	86.13%	80.60%	--	--

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	93.25%	93.30%	93.35%	93.35%	92.58%	92.71%	92.58%	92.84%
P = 2	98.25%	99.20%	99.35%	99.40%	91.08%	90.56%	90.49%	90.17%
P = 3	99.50%	99.70%	99.75%	99.75%	90.36%	89.45%	88.67%	88.67%
P = 4	--	--	--	--	--	--	--	--
P = 5	--	--	--	--	--	--	--	--

As shown in the table above, most of the cases have a good result. But in some cases, there is no optimal solution since the kernel doesn't satisfied the Mercer condition.

Specifically, for linear kernel and polynomial kernels with $p = 2$ and 3, the Mercer condition can be satisfied, while for polynomial kernels with $p = 4$ and 5, the Mercer condition is no longer satisfied. Therefore, when $p = 4$ and 5, there is no optimal hyperplane existing. Meanwhile, the problem is non-convex for quadratic programming.

From these results, we can find that, with the increase of both parameter p and C, the training accuracy increases, but the test accuracy decreases. It indicates that if the training accuracy is too high, the model is overfitting with the training set. Thus, the model will show a relatively worse performance on the test set. In summary, different models have different characteristics and performance on training set and test set. The influence of the parameters is obvious as shown above. Fine-tuning of parameters is important both in experiments and practice.

Task 3:

Based on the results in task 1 & 2 and the analysis above, in order to designing a model with better performance on the unknown evaluation data, we need do some trade off between the training accuracy and test accuracy.

If the training accuracy is too high, the model may overfitting with the training data so that it can not perform well on the test data and evaluation data. Otherwise, the model may be trained not mature enough.

From the perspective of balance, the SVM model of soft margin with polynomial kernel is applied in "svm_main.m". The parameters are selected as $p = 2$ and $C = 0.1$, i.e. the kernel function is $K\left(x_i, x_j\right) = (x_i^T x_j + 1)^2$.

For convenience, I have already stored the required variables ("alpha" and "Bo") in a separate file "parameters.mat". Thus, both "parameters.mat" and "train.mat" are loaded in "svm_main.m". The "eval_data" will also be standardized using the mean and variance of each feature from the training set. Finally, it will generate a vector with the name "eval_predicted" for evaluation.