# Performance Analysis of Machine Learning Methods for Image Classification

Xuhui Wang

February 2021

## Introduction

This report has 4 parts where each part involves applying a distinct method with different optimization and hyper-parameter tuning to quantitatively analyze the results on a binary classification problem. The theme of these parts is to use cross-validation for hyperparameter tuning and model selection.

We analyze the performance of these methods on a somewhat-recently devised classification problem, known as Fashion-MNIST. Basically, the idea is to classify $28 \times 28$ grayscale images of clothing/apparel items, where each image belongs to one of ten classes.[1]

Fashion-MNIST corresponds to a multi-class problem with ten classes. Given that this assignment is about binary classification, we use only the data points (from both the training and test sets) from classes 5 and 7 (corresponding to "Sandal" and "Sneaker" respectively). We let class 5 from Fashion-MNIST be class 0 of the binary classification problem (and class 7 from Fashion-MNIST is class 1 of the binary classification problem).[1]

## Regularization

Regularizations are techniques used to reduce the error by fitting a function appropriately on the given training set and avoid overfitting. For the methods used in the following, I primarily go with $l_2$ regularization, which means that I penalize the training objective according to $||w||^2$, i.e., the sum of the squares of the weights (not including the bias term). Specifically, for binary labels $y_i$  0, 1, the suggested training objective is:

$$\min_{\mathrm{w}\in\mathbb{R}^{\mathrm{d}},\mathrm{b}\in\mathbb{R}} \quad \frac{1}{2}||w||^2 + C\sum_{i=1}^{n} l(y_i, \sigma(\langle w, x_i\rangle + b))$$

where

- $l(y, \hat{y}) = -y \log \hat{y} - (1-y)\log(1-\hat{y})$ denotes the cross-entropy loss;
- $\sigma(z) = \frac{1}{1+e^{-z}}$ denotes the sigmoid function;
- $||w||^2 = \sum_{j=1}^{d} w_j^2$ denotes the squared Euclidean norm of the vector $w$.
- $C \leq 0$ denotes the regularization parameter. Higher $C$ means less regularization, so more emphasis is given to the training error.[1]

# Logistic Regression

## Method

Based on logistic regression, we plot how the training and test error change as we vary the regularization parameter $C$. For training, we use the whole training set, and for testing, we use the whole test set. We try more than ten values of the regularization parameter. And we use a logarithmically spaced grid for initial value $C_0 > 0$ and base $\alpha > 1$, which means that we try $C_0, \alpha C_0, \alpha^2 C_0$, and so on. We expect that this method would capture the regime of both under-fitting (too much regularization) and over-fitting (too little regularization).

## Experiment

Specifically, we make $C$ value vary logarithmically between the range $[10^{-7}, 10^{7}]$. Figure 1 (in the next page) shows how training error and test error change as $C$ value varies between the given range. The training error keeps decreasing, while the test error experiences a decreasing trend until $C$ value reaches $10^{-2}$, then followed by a stable increase and keeps constant.

It is reasonable to consider the decreasing process as the regime of underfitting where the regularization is more, and consider the following process as the regime of overfitting where the regularization is considerably less. The statistics, when executing the program, gives $C$ value around $10^{-2}$ as the best circumstance, corresponding to the minimum test error equal 0.038.

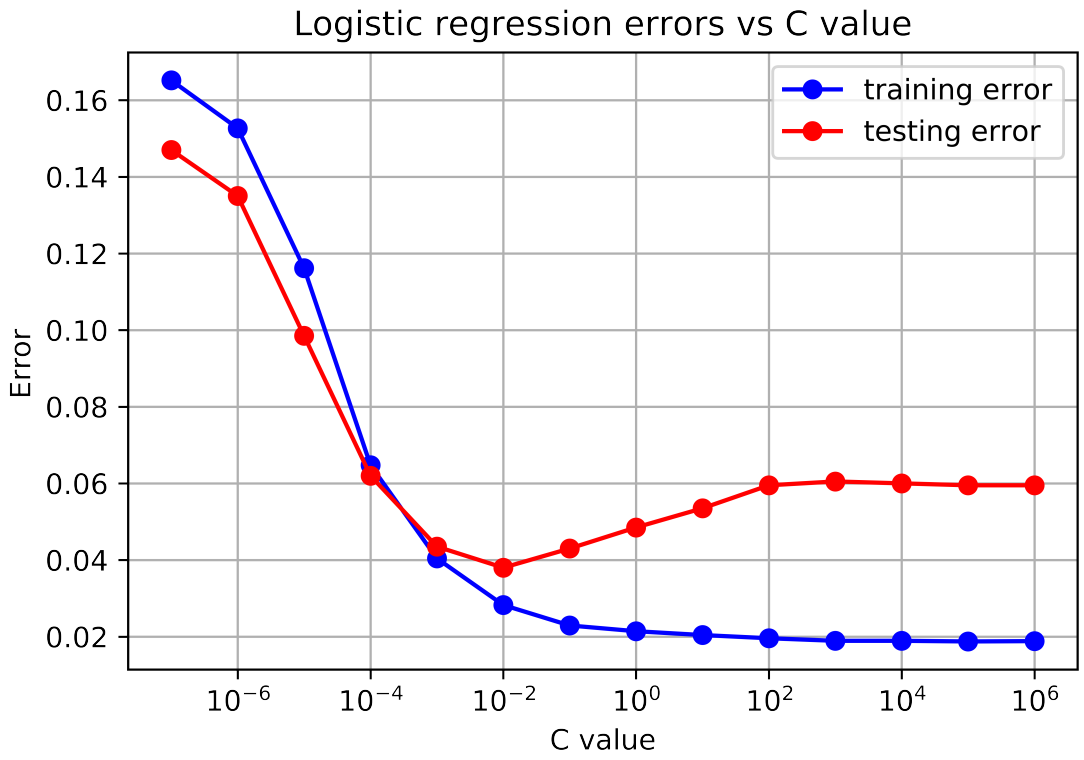


Figure 1: Logistic Regression errors vs $C$ value

### Conclusion

We have already explored how underfitting and overfitting happen, given the previous experiment. Though higher $C$ value means less regularization, it does not mean that we have to make $C$ value as large as possible to achieve better result. Only appropriate regularization better minimizes the gap between test error and training error and helps achieve the better result.

## Support Vector Machine

### Method

Based on linear SVM (SVM with the linear kernel), we plot how the training and test error change as we vary the regularization parameter $C$. We proceed the experiment as we do for the previous logistic regression, but the range of values we try for the regularization parameter would be different.

### Experiment

Specifically, we make $C$ value vary logarithmically between the range $[10^{-5}, 10^{10}]$ different from that of Part 1. Figure 2 shows how training error and test error change as $C$ value varies between the given range. We could observe that the training error and the test error keep a similar moving

pattern, despite a little gap between their values, as the $C$ value grows up. The training error reaches the lowest point at $C$ value equal to 1, followed by an increasing process and fluctuation. The test error experiences the same moving, while achieving the best performance at $C$ value equal to $10^{-2}$.

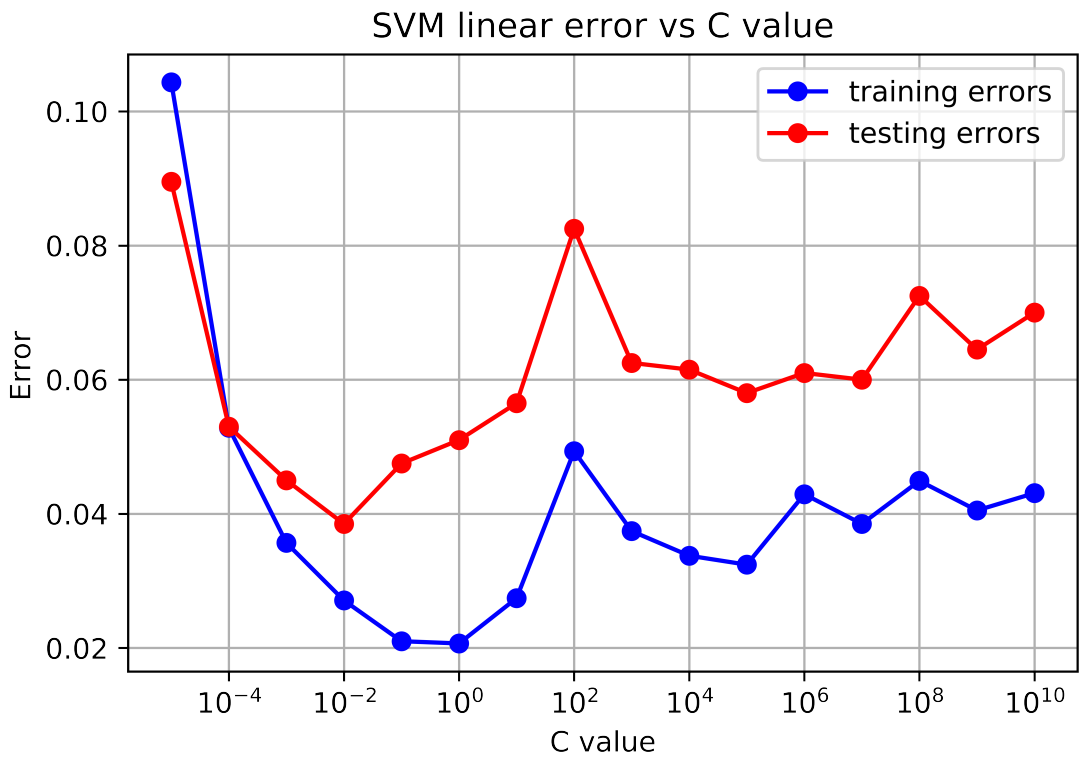


Figure 2: SVM linear error vs $C$ value

It is reasonable to consider the decreasing process as the regime of underfitting where the regularization is more, and consider the following process as the regime of overfitting where the regularization is considerably less. The statistics, when executing the program, gives $C$ value around $10^{-2}$ as the best circumstance, corresponding to the minimum test error equal 0.037.

## Conclusion

We have already explored how underfitting and overfitting happen, given the previous experiment, similar to the results of the logistic regression part. Though higher $C$ value means less regularization, it does not mean that we have to make $C$ value as large as possible to achieve better result. Only appropriate regularization better minimizes the gap between test error and training error and helps achieve the better result.

# Hyperparameter Tuning & Cross Validation

## Method

We have already done some analysis for each algorithm. In this part, we would investigate the comparison of an optimally regularized logistic regression classifier and an optimally regularized linear SVM. We would use only the training data and k-fold cross-validation (for k being 7) to select the optimal value of the logistic regression regularization parameter. Likewise, we would use k-fold cross-validation to select the optimal value of the SVM regularization parameter. Then we would give result for each optimally-tuned method on the entire training set and report the method's test error.

## Experiment

We make $C$ value vary logarithmically between the range $[10^{-6}, 10^{5}]$ to perform the analysis. For hyperparameter tuning, 7-fold cross-validation is performed on both logistic and SVM methods. Figure 3 and Figure 4 show how training error and test error change as $C$ value varies between the given range for two different methods respectively. As shown in the graphs, the test errors and training errors vary with generally similar patterns as shown in the first and second parts, with the values slightly changed.

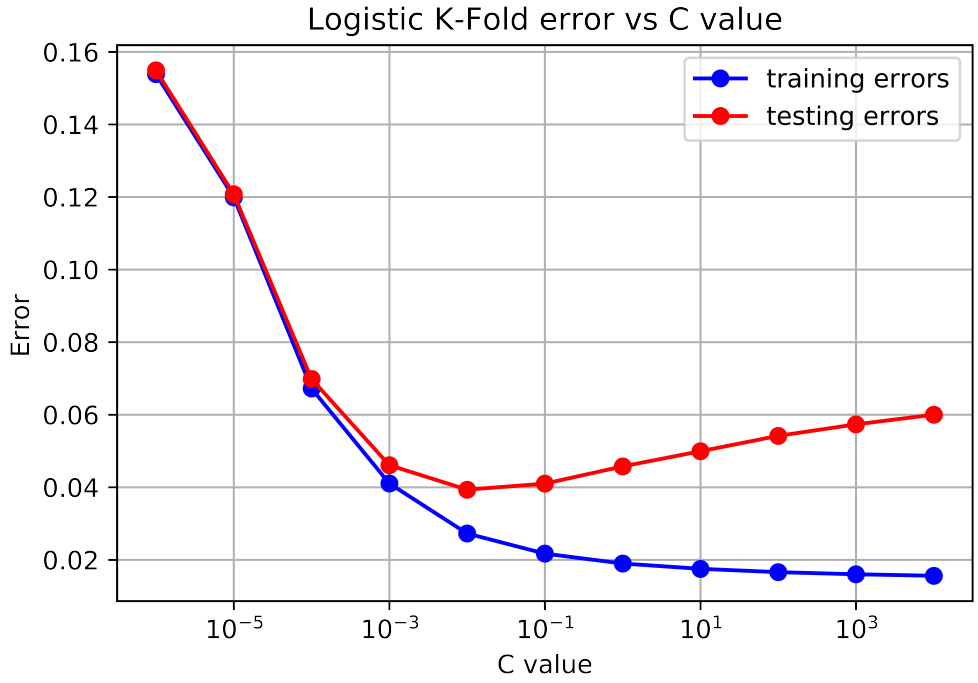


Figure 3: Logistic Regression K-fold errors vs $C$ value

Figure 3 shows that the logistic regression method test error reaches the lowest point at $C$ value $= 10^{-2}$, same as the result in the first part, and the test error is around 0.04. At this point, we could observe that there is no obvious underfitting and overfitting.

It is also reasonable to consider the decreasing process as the regime of underfitting where the regularization is more, and consider the following process as the regime of overfitting where the regularization is considerably less. The statistics, when executing the program, gives $C$ value around $10^{-2}$ as the best circumstance, corresponding to the minimum test error equal 0.04.

Figure 4 shows that the SVM method test error reaches the lowest point at $C$ value = $10^{-2}$, also similar to the result in the second part, and the test error is around 0.04. At this point, we could observe that there is no obvious underfitting and overfitting.

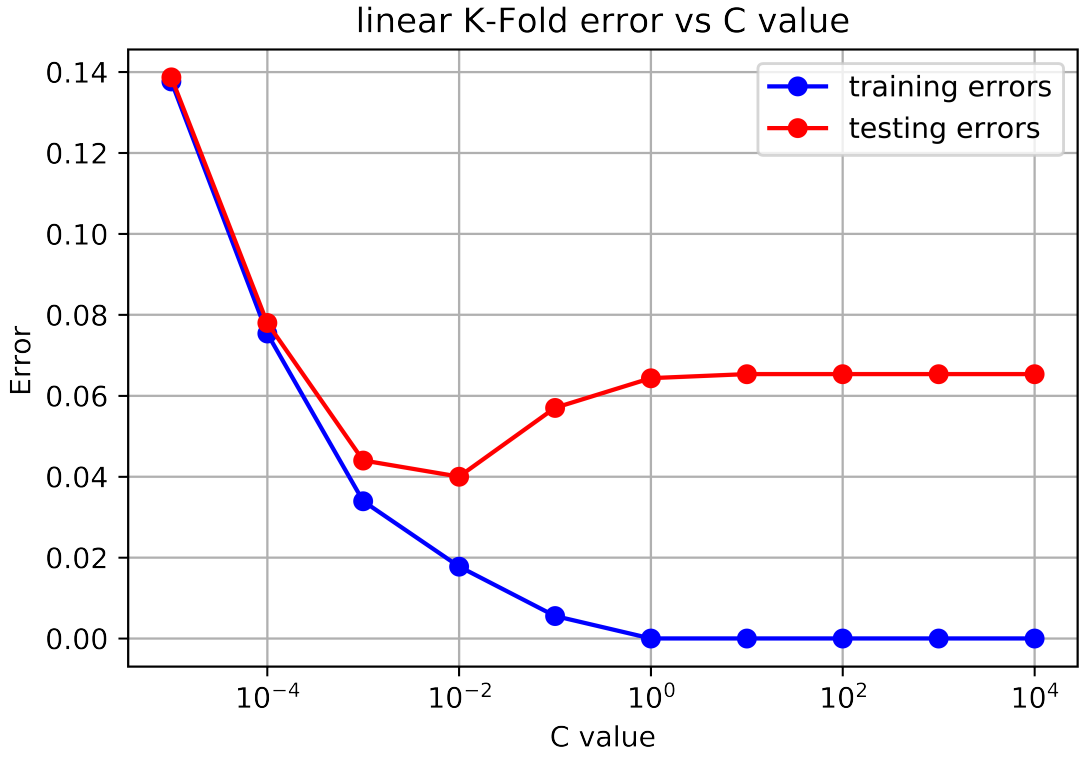


Figure 4: Linear SVM K-fold errors vs $C$ value

It is also reasonable to consider the decreasing process as the regime of underfitting where the regularization is more, and consider the following process as the regime of overfitting where the regularization is considerably less. The statistics, when executing the program, gives $C$ value around $10^{-2}$ as the best circumstance, corresponding to the minimum test error equal 0.04.

## Conclusion

So far we already have results for 4 different conditions: logistic regression by K-fold cross-validation on training set (Part 3), SVM by K-fold cross-validation on training set (Part 3),

logistic regression on entire training set (Part 1) and SVM on entire training set (Part 2). Surprisingly, the optimal regularization parameter for each method that we obtained from the previous analysis had already been tested on the entire training set in Part 1 and Part 2. Below table shows the comparison between four different conditions.

| Methods | $C$ value | test error (K-fold cross-validation) | test error (entire training set) |
|---|---|---|---|
| Logistic Regression | $10^{-2}$ | 0.04 | 0.038 |
| SVM | $10^{-2}$ | 0.04 | 0.037 |

Using the optimal $C$ values determined by the K-fold cross-validation on training set, we could achieve better performances for two different methods on the entire training set. Meanwhile, underfitting and overfitting also could be observed on the analysis by K-fold cross-validation.

# Support Vector Machine with Gaussian kernel

## Method

We would go beyond linear classifiers in this part. We would use kernelized SVMs with the Gaussian kernel. This kernel has a scale parameter. For each value $\gamma$ in a good range of values of the scale parameter, we use k-fold cross-validation on the training set to select the optimal regularization parameter $C_\gamma$. For each $(\gamma, C_\gamma)$ pair, we would have a tuned SVM that can be trained. For each tuned SVM, we train on the full training set and compute both the training error and the test error. Finally, we plot the results. The x-axis will be $\gamma$, and the y-axis will be the training error (or test error) based on an SVM with parameters tuned to $(\gamma, C_\gamma)$.

## Experiment

We make $\gamma$ value vary logarithmically between the range $[10^{-11}, 10]$, and $C$ value, the potential optimal regualarization parameter for each $\gamma$, vary logarithmically between the range$[10^{-5}, 10^{12}]$ to perform the analysis. K-fold cross-validation is performed on the training set to select the optimal regularization parameter $C_\gamma$ for each $\gamma$.

Figure 5 shows how training error and test error change for the different combinations of $(\gamma, C_\gamma)$. Errors are computed by the SVM with the parameters tuned to $(\gamma, C_\gamma)$ trained on the full training set. Given that figure 5 shows only the $\gamma$ value and the errors, we create a table to show all the $(\gamma, C_\gamma)$ combinations for the ease of observation.

| $\gamma$ | $10^{-11}$ | $10^{-10}$ | $10^{-9}$ | $10^{-8}$ | $10^{-7}$ | $10^{-6}$ | $10^{-5}$ | $10^{-4}$ | $10^{-3}$ | $10^{-2}$ | $10^{-1}$ | 1 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $C_\gamma$ | $10^5$ | $10^5$ | $10^6$ | $10^5$ | $10^4$ | $10^3$ | $10^2$ | 10 | 10 | 10 | 10 | 1 | 1 |

From the figure 5, we could observe that the training error decreases as the $\gamma$ grows up, while the test error decreases until $\gamma$ reaches $10^{-3}$ where the $C_\gamma$ is 10 and the test error is around 0.22, followed by a rapid rise.

It is also reasonable to consider the decreasing process as the regime of underfitting where the regularization is more, and consider the following process as the regime of overfitting where the regularization is considerably less. The statistics, when executing the program, gives $\gamma$ value = $10^{-3}$, $C_\gamma = 10$ as the best circumstance, corresponding to the minimum test error equal 0.022.

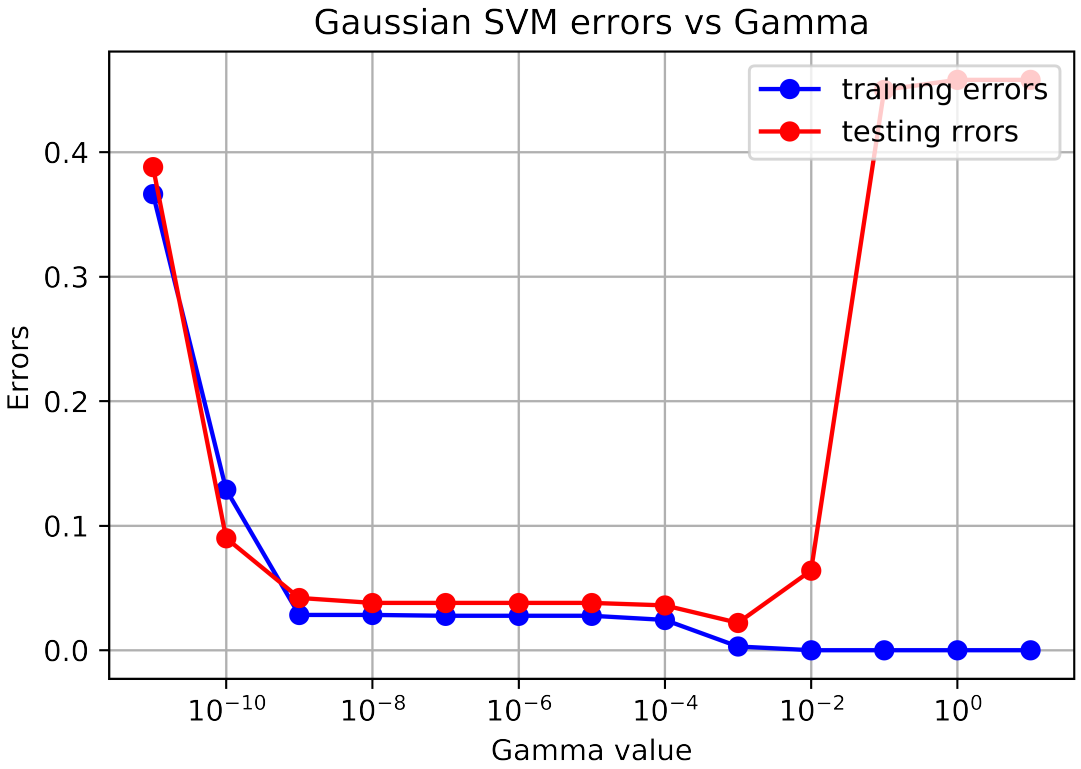


Figure 5: Gaussian SVM errors vs $\gamma$

### Conclusion

The table in the third part shows that the test error of linear SVM (with regularization parameter tuned via cross-validation on the training set) is 0.037, slightly lower than the result, test error is 0.022, of SVM with Gaussian kernel.

Using SVM with Gaussian kernel and the optimal $C_\gamma$ values determined by the K-fold cross-validation on training set, we could achieve better performance than that of linear SVM. Meanwhile, underfitting and overfitting also could be observed on the analysis.

## Results

For this problem, Logistic regression is a relatively faster method. However, SVM may have a better performance when the best regularization parameter is chosen. For SVM, Gaussian kernel model may have better performance than linear one. In addition, the use of K-fold cross-validation to determine the best regularization parameter also helps improve the performance.

## References

[1] N. Mehta, *Seng474: Data mining course materials.*