Homework 2

Collaborators:

Name: Jessie Peng

Student ID: xxxxxxxxxx

Problem 2-1. A Walk Through Linear Models

(a) Perceptron

Answer:

1. I repeated each experiment for 1000 times. And each time I generated a test set of size 10000 without noise to estimate the testing error.

   When the size of training set is 10, the average training error rate is 0.0, and the expected testing error rate is 0.1115319.

   When the size of training set is 100, the average training error rate is 0.0, and the expected testing error rate is 0.0136943.

2. When the size of training set is 10, the average number of iterations is 10.496.

   When the size of training set is 100, the average number of iterations is 144.045.

3. If I performed the cyclic perceptron learning algorithm as in the linearly separable case, it would never converge because there would always be some misclassified points.

   Instead, I perform Pocket Algorithm once the iteration number exceeds some limit. (In this experiment, I set it to 1000.) When the size of training set is 100, the average training error rate is 0.18603, the expected testing error rate is 0.1302829.

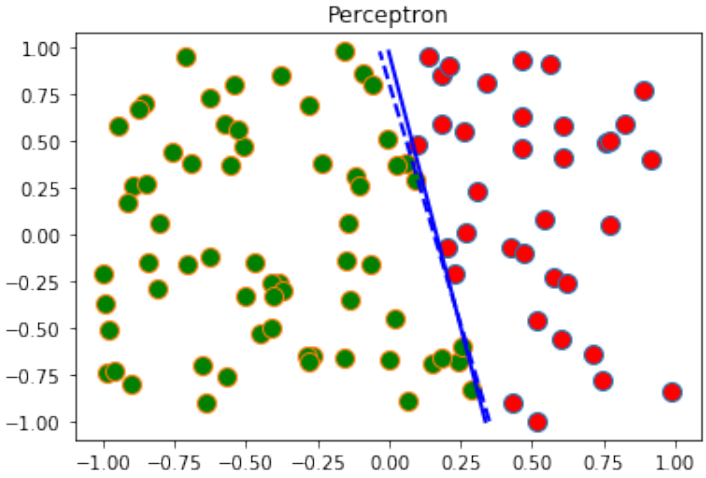

Figure 1: The plotting result for perceptron when nTrain = 100.

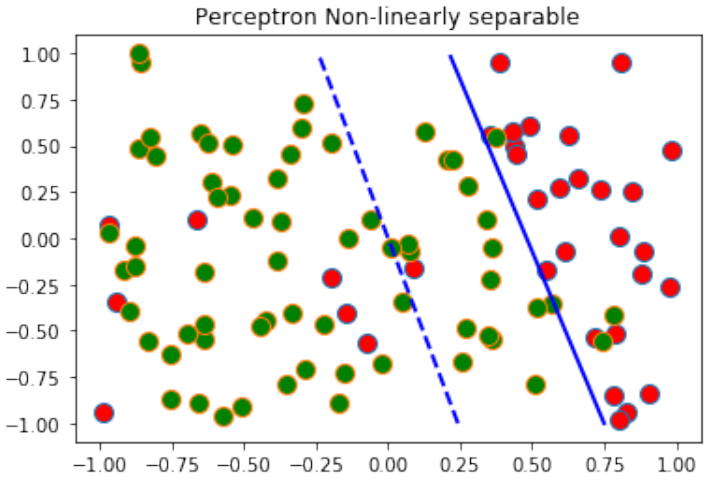

Figure 2: The plotting result for perceptron when training data is not linearly separable.

(b) Linear Regression

Answer:

1. The training error rate is 0.0382, and the testing error rate is 0.0480877.
2. The training error rate is 0.13284, and the testing error rate is 0.0599449.
3. The training error rate is 0.49, and the testing error rate is 0.5496.
4. The training error rate is 0.05, and the testing error rate is 0.066.

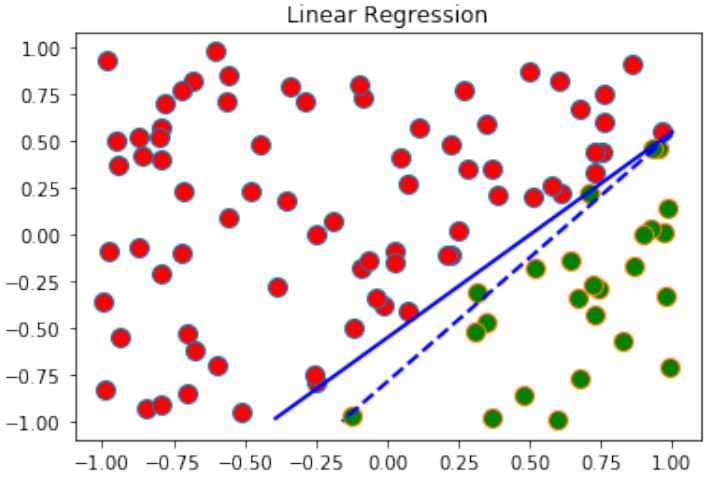

Figure 3: The plotting result for linear regression.

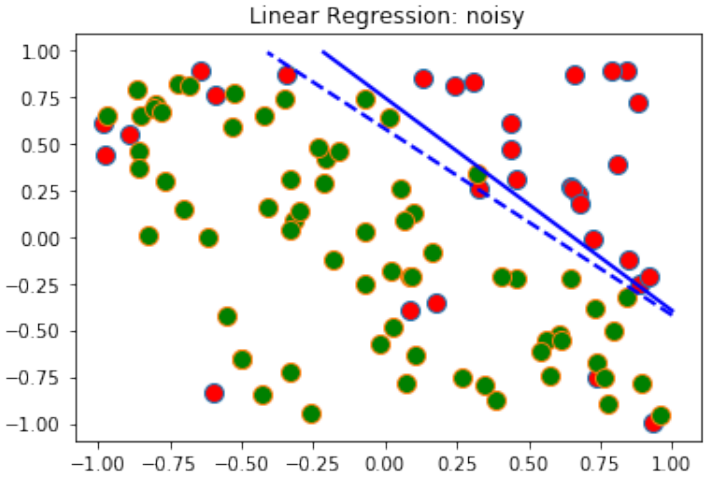

Figure 4: The plotting result for linear regression when training data is not linearly separable.

(c) Logistic Regression

Answer:

1. To perform MLE, we first get the log-likelihood function:

$$\begin{aligned}\log P(y \mid x; \theta) &= \log((h_\theta(x))^y(1-h_\theta(x))^{1-y}) \\ &= y \log h_\theta(x) + (1-y)\log(1-h_\theta(x))\end{aligned}$$

where $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$.

To maximize the log-likelihood function with gradient descent, we calculate the following partial differential:

$$\begin{aligned}
\frac{\partial \log P(y \mid x;\theta)}{\partial \theta} &= \frac{y}{h_\theta(x)}\frac{\partial h_\theta(x)}{\partial \theta} - \frac{1-y}{1-h_\theta(x)}\frac{\partial h_\theta(x)}{\partial \theta} \\
&= \frac{y - h_\theta(x)}{h_\theta(x)(1-h_\theta(x))}\frac{\partial h_\theta(x)}{\partial \theta} \\
&= \frac{y - h_\theta(x)}{h_\theta(x)(1-h_\theta(x))}(-h_\theta^2(x))\frac{\partial e^{-\theta^T x}}{\partial \theta} \\
&= \frac{y - h_\theta(x)}{1-h_\theta(x)}(-h_\theta(x))e^{-\theta^T x}(-x) \\
&= (y - h_\theta(x))x
\end{aligned}$$

Note that it's actually gradient ASCENT here, since we aim to MAXIMIZE the target function. But gradient descent and gradient ascent are essentially the same anyway.

The training error rate is 0.0397, and the expected testing error rate is 0.04588.

2. The training error rate is 0.1533, and the expected testing error rate is 0.08363.

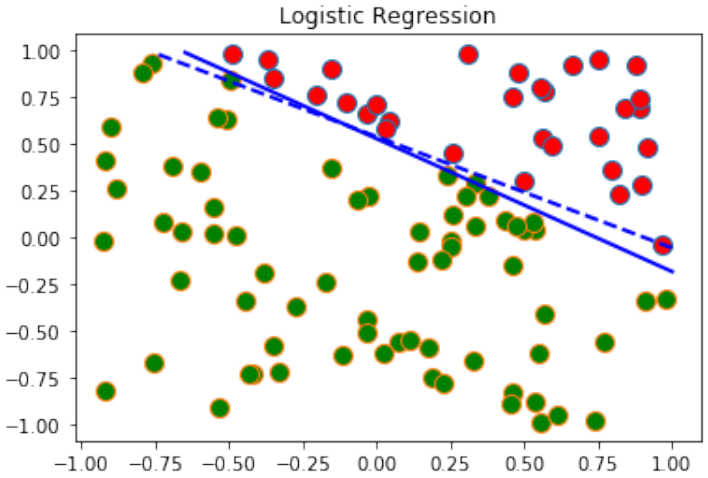

Figure 5: The plotting result for logistic regression.

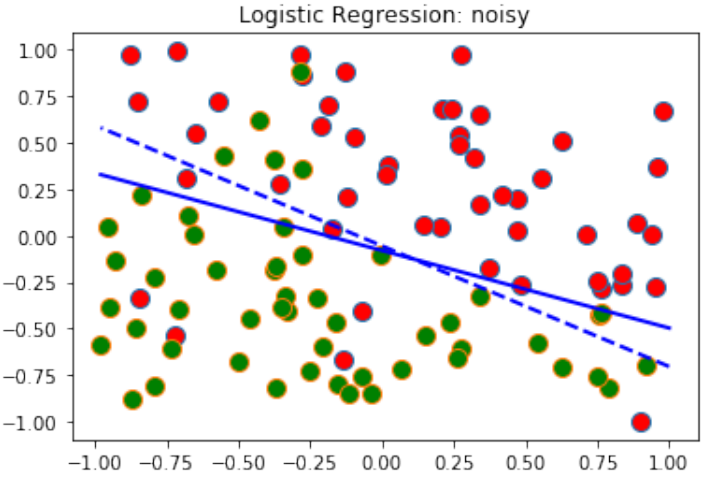

Figure 6: The plotting result for logistic regression when training data is not linearly separable.

(d) Support Vector Machine

Answer:

1. The training error rate is 0.0, and the testing error rate is 0.036606.
2. The training error rate is 0.0, and the testing error rate is 0.009104.
3. The average number of support vectors is 2.99.

Bonus SVM performs poorly with noisy training data. This is because we implement SVM by solving the following quadratic programming:

$$\min_{w,b} \frac{1}{2} w^T H w$$

$$s.t.\ y_i(w^T x_i + b) \geq 1, \forall i = 1, 2, ..., n.$$

If the training data are not linearly separable, those constraints cannot consistently hold. Therefore, the quadratic programming will end up running out of iterations without finding even one feasible solution, and thus will not produce an ideal result.

I also implemented a SVM with slack variables which performed much better. The SVM with slack variables is equivalent to the following quadratic programming:

$$\min_{w,b} \frac{1}{2} w^T H w + C \sum_{i=1}^{n} \xi_i$$

$$s.t.\ \ y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \forall i = 1, 2, ..., n.$$
$$\xi_j \geq 0, \qquad\qquad\qquad\ \ \forall j = 1, 2, ..., n.$$

I simply set the parameter $C = 0.5$, meaning approximately equal importance of the two goals (larger margin and less distance between outliers and the margin). The training error rate is 0.1346, and the testing error rate is 0.062815.

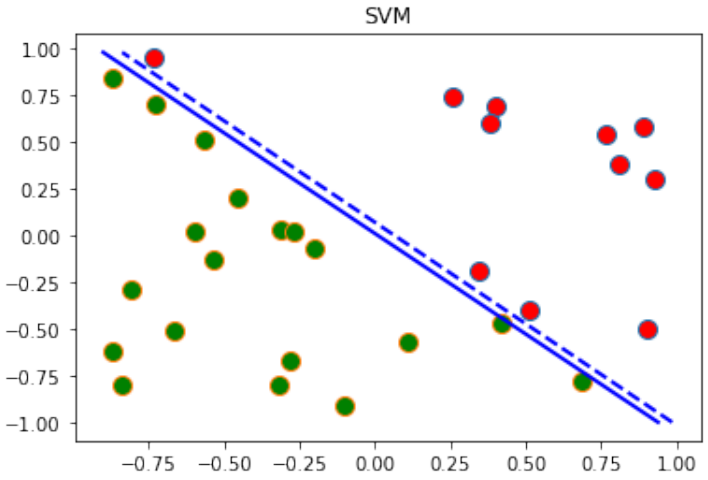

Figure 7: The plotting result for SVM when nTrain is 30.

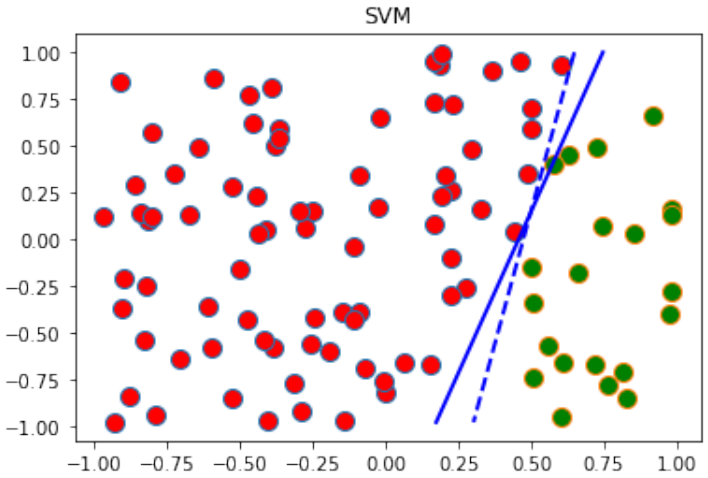

Figure 8: The plotting result for SVM when nTrain is 100.

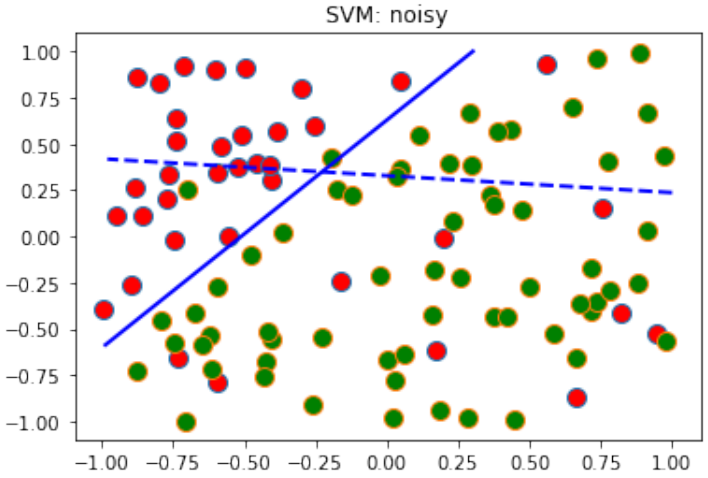

Figure 9: The plotting result for SVM when training data is not linearly separable.

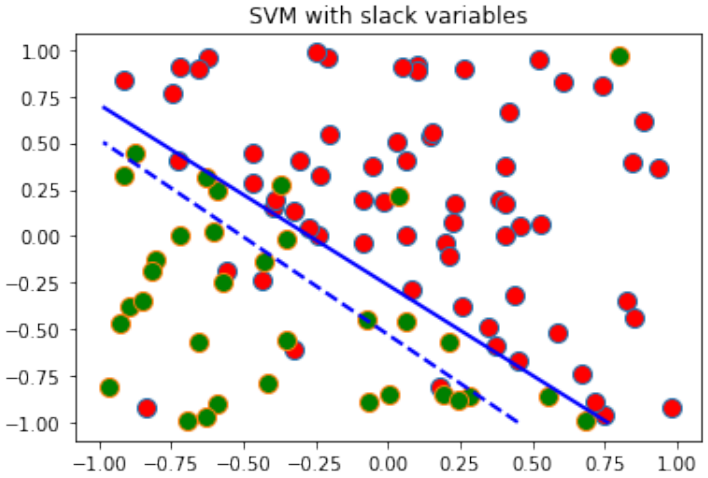

Figure 10: The plotting result for SVM with slack variables.

Problem 2-2. Regularization and Cross-Validation

(a) Implement Ridge Regression, and use LOOCV to tune the regularization parameter λ.

Answer:

1. $\lambda = 1 \times 10^{-2}$ as is chosen by LOOCV.

2. With regularization, $\sum_{i=1}^{m} \omega_i^2 = 877.1308$;
   Without regularization (let $\lambda = 0$), $\sum_{i=1}^{m} \omega_i^2 = 4932.7663$.

3. With regularization, the training error rate is 0, the testing error rate is 0.0658;
   Without regularization, the training error rate is 0, the testing error rate is 0.0939.

(b) Implement Logistic Regression, and use LOOCV to tune the regularization parameter λ.

Answer: $\lambda = 1 \times 10^{-3}$ as is chosen by LOOCV.

With regularization, $\sum_{i=1}^{m} \omega_i^2 = 3482.0388$. The training error rate is 0.005, and the testing error rate is 0.0618.

Without regularization, $\sum_{i=1}^{m} \omega_i^2 = 3497.2904$. The training error rate is 0.005, and the testing error rate is 0.0613.

Problem 2-3. Bias Variance Trade-off

Let's review the bias-variance decomposition first. Now please answer the following questions:

(a) True of False

Answer:

1. False. The bias reflects how well the model can approximate the truth at best, and it only depends on the model itself rather than the training dataset. Adding more training examples can reduce the variance, but cannot reduce the bias. So it cannot improve the test error significantly.

2. False. We usually prefer models with high variance over those with high bias not because they fit the training set better, but because the variance can be reduced with other techniques. If a model has a high bias, we cannot improve the best performance we can get any more; But if a model has a high variance, we can still improve the performance by adding regularization, enlarging the training dataset, etc. Besides, the fact that a model with high variance fits the training set better

is actually the opposite of what we expect of a good model, because it indicates overfitting and higher test error rate.

3. True. More parameters means higher complexity of the model. A model with higher complexity is more specialized in conforming with the training dataset, failing to serve a wider range of testing data, and thus it suffers overfitting. Such an overfitted model typically deviates from what we expect to learn, in other words, has higher variance.

4. False. Since regularization serves to counter overfitting, it will probably reduce the model's accuracy on the training set, rather than improve it. Moreover, overly regularize the model can curt the performance both on training data and testing data, as regularization shadows the initial fitting target to some degree.

5. False. As explained above, a large regularization parameter could overly emphasize the importance of the regularization term, and thus the influence of other terms is weakened. This may lead to worse performance.