CIS4930/5930 Machine Learning

1: SVM

(a)

The optimization problem for the soft margin SVM is formulated as the following

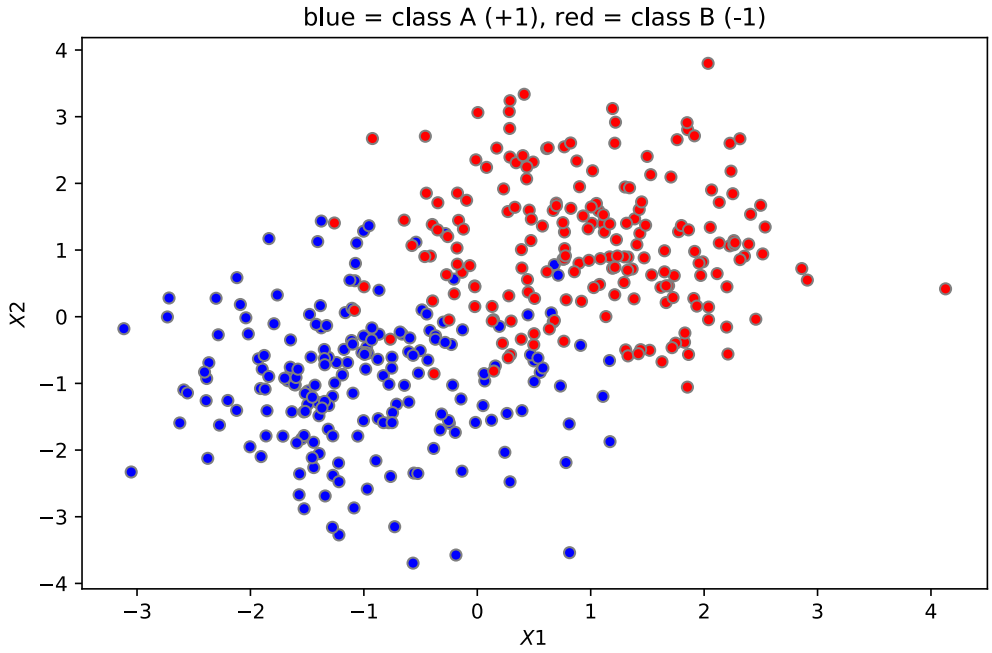
$$\min_{\mathbf{w},b,\xi} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i$$

$$\text{subject to} \quad y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \xi_i \ \wedge \ \xi_i \geq 0, i \in [1, m].$$

According to the question requirements, to solve the convex optimization problem of soft margin SVM classification algorithm, I used an external library called CVXOPT. The details of implementing with CVXOPT are in the codes that I have submitted.

(b)

According to this question requirements and using the required mean and covariance matrix stated in the question, I generated 200 sample points for class A (+1) and 200 sample points for class B (-1). I did seed the random number generator with the last 5 digits of my Lib# which is 65260. Now, I have got 400 two-dimensional sample points that I have used to train my SVM implementation. The scatter plot for the sample points looks like the following.



Then I checked my implementation of soft margin SVM after running on these 400 two-dimensional sample points having divided in two classes. To train the SVM, along with the dataset it also needs a kernel and a trade-off parameter C. When I call the implemented function with these parameters, it returns me the parameters (w, b) and the support vectors (S). According to the class slides, please note that for soft margin SVM, support vectors are the points along the margin or outliers. The soft margin that I have got is by the following formula.

$$\rho = 1/\|\mathbf{w}\|$$

I vary the parameter C as per the suggestion of the question and get different set of results for two different C parameter. They are discussed and plotted below.

For C= 10

The number of support vectors = 68

The margin is: 0.4832573434395237

Leave-one-out cross validation error is : 0.07

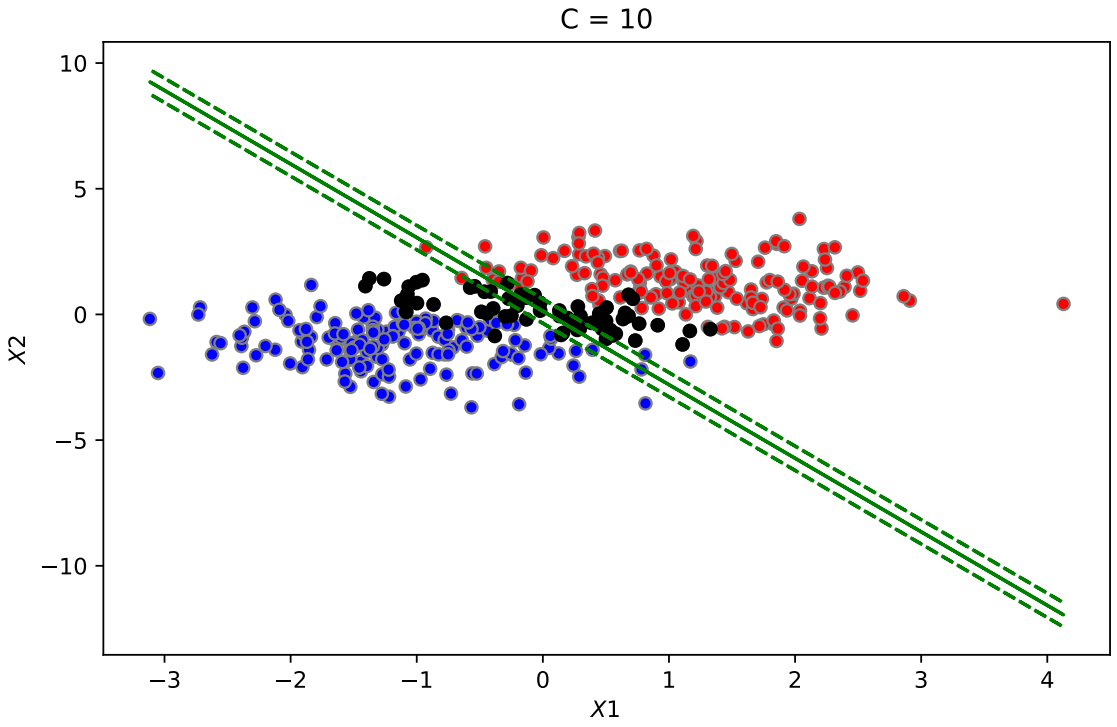
For C= 0.02

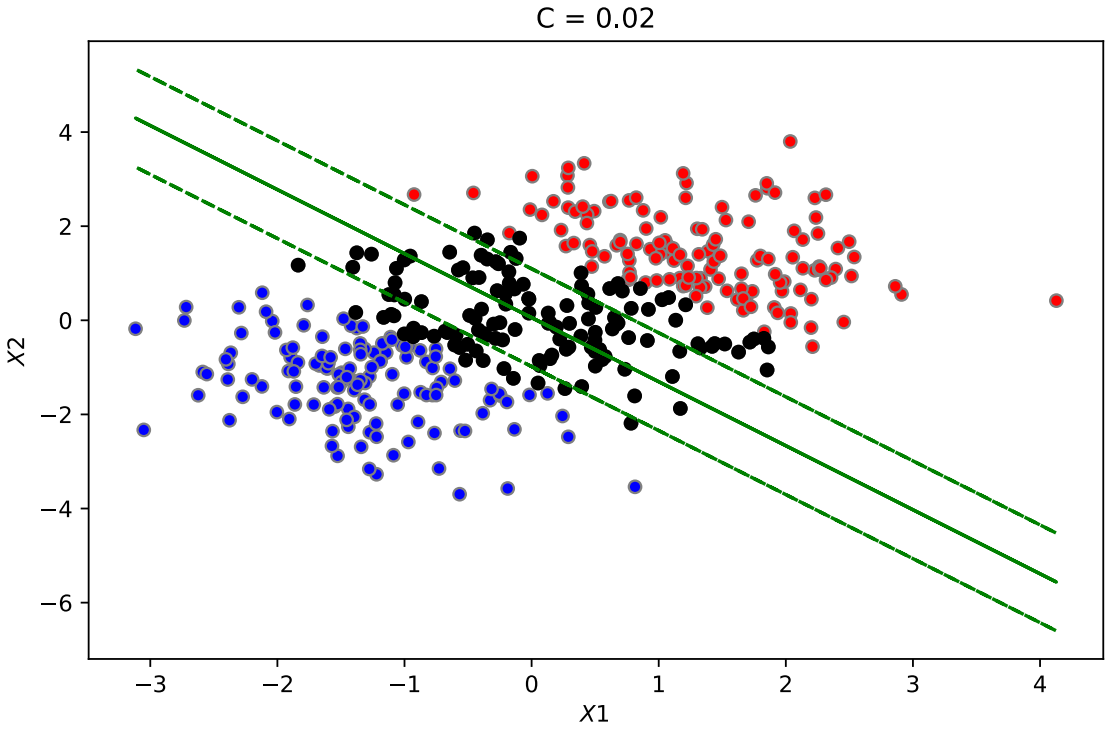
The number of support vectors = 131

The margin is: 1.0385929738907995

Leave-one-out cross validation error is: 0.065

The decision boundary along with the training points are given below for C =10 and C = 0.02. The black ones denote the support vectors.





(c)

To work with the MNIST dataset, I actually use this dataset that is already built-in at the Keras which is an open-source neural-network library written in Python and run on top of tensorflow. After importing Keras I just loaded the dataset using their API. The MNIST dataset consists of 70,000 handwritten digit images of size 28×28 for each image; 60,000 training samples and 10,000 test/validation samples. It has 10 classes (labels are 0 to 9). According to the requirements of this questions, I used only the training and test samples that have labels 0 and 1. In that way, I have got 12665 training samples among which 5923 are in class 0 and 6742 are in class 1. Similarly, I have got 2115 test samples among which 980 are in class 0 and 1135 are in class 1.

After training the soft margin SVM that I have implemented with the train patterns, I have checked how the results work on the test patterns. The results are given below for C= 10.

Total number of MNIST test samples corresponding to 0 and 1 is: 2115
Total number of test samples correctly classified: 2113
Total number of test samples incorrectly classified: 2
The generalization error: 0.0009456264775

2: Linear and Logistic Regression and Gen Error Comparison

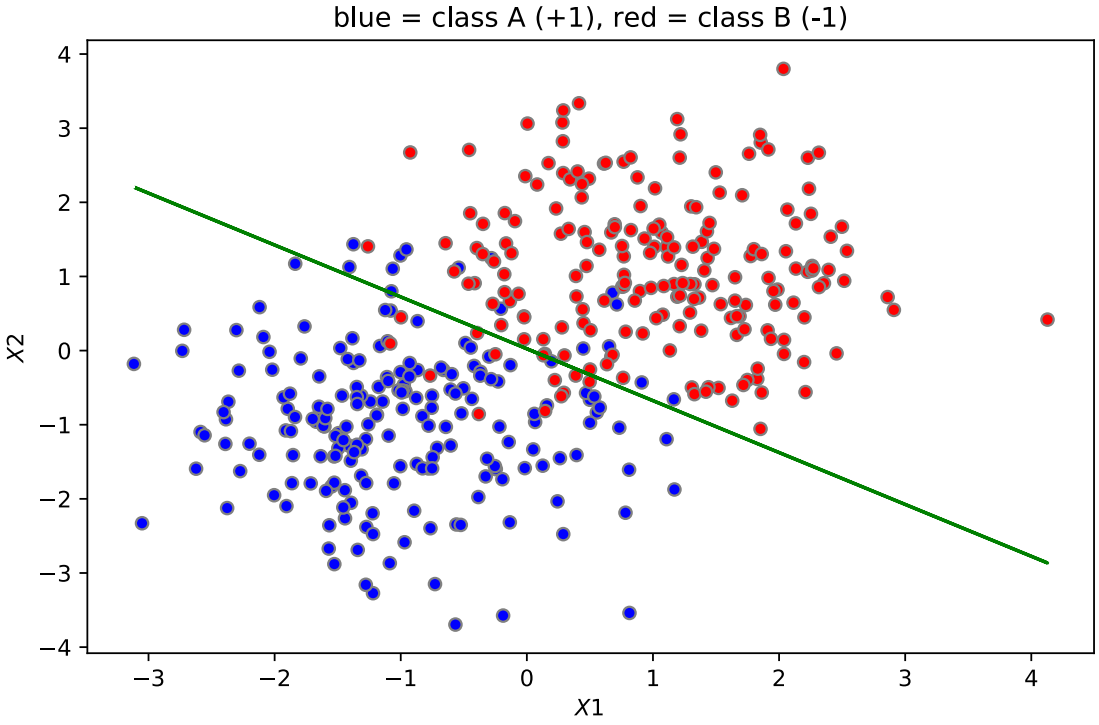
(a)

According to the question requirements, to implement linear regression, I have followed the equations in the slides. Here, the solution is a linear function of the outputs y that is shown below. The implementation details are in the codes.

$$f : \mathcal{R}^d \rightarrow \mathcal{R} \qquad f(\mathbf{x}; \mathbf{w}) = w_0 + w_1 x_1 + \ldots w_d x_d$$

$$\hat{\mathbf{w}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

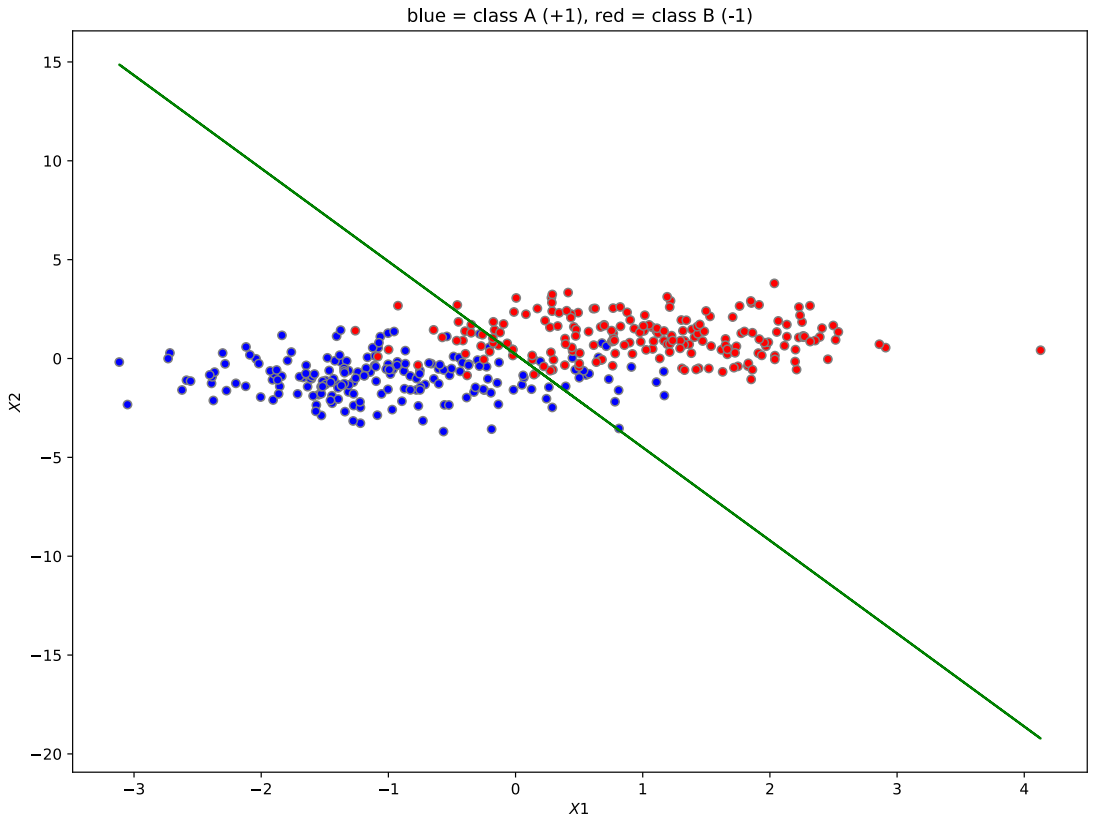
I created a binary classifier by thresholding the output at 0. Then I ran the implementation on the 400 samples that I have generated for 1(b). The decision boundary along with the training points are given below.



Leave-one-out cross validation error is : 0.065

(b)

According to the question requirements, to solve the convex optimization problem of logistic regression, I used an external library called CVXPY. The details of implementing with CVXPY are in the codes. I created a binary classifier by thresholding the output at 0. Then I ran the implementation on the 400 samples that I have generated for 1(b). The decision boundary along with the training points are given below.



Leave-one-out cross validation error is : 0.0675

(c)

Here, Total number of MNIST test samples corresponding to class 0 and 1 is 2115. I have run my implementation to train SVM, linear regression and logistic regression on 12665 MNIST training samples corresponding to class 0 and 1. After that I have found the test/ generalization error by checking the number of samples correctly and incorrectly classified by each of the trained models. The comparisons are given below.

SVM:
Total number of test samples correctly classified: 2113
Total number of test samples incorrectly classified: 2
The generalization error: 0.0009456264775

Linear Regression:
Total number of test samples correctly classified: 2102
Total number of test samples incorrectly classified: 13
The generalization error: 0.006146572104

Logistic Regression:
Total number of test samples correctly classified: 2110
Total number of test samples incorrectly classified: 5
The generalization error: 0.002364066194