6.867: Homework 2

Anonymous authors

***Abstract*— This is Machine Learning homework 2. Topics include Logistic Regression, SVM. All work has been done in Python.**

I. Logistic Regression (LR)

In this section, we consider Logistic Regression (LR) for binary classification of several 2D datasets, and compare the effect of different model parameters and regularization.

A. Part 1

[todo]

B. Part 2

An alternative regularization (L_1) was then compared against L_2 from the previous section. In Fig. 1, the elements of the weight vector are compared. The top row shows L_1 regularization for each of the four datasets, and the bottom row shows L_2. In general, increased regularization parameter, λ, causes smaller magnitude weight vector, since the blue and green points (small λ) are further from zero than the magenta and yellow points (large λ). This is true for both L_1 and L_2 regularization. Especially clear in the 2nd and 4th columns, L_1 regularization creates a more sparse weight vector, as all elements are zero (yellow) for the highly regularized case, whereas with the same regularization constant with a L_2 penalty has some non-zero elements.

In addition to the weight vector, the decision boundary is affected by λ and regularization method. In Fig. 2, the decision boundaries for various λ values are shown for same dataset with L_1 on the left and L_2 on the right. As λ increases, the boundary is placed in a position that will cause training errors ($\lambda > 1$). The boundaries from the two regularization strategies are slightly different, but not by very much. In Fig. 3, the same concept is shown across all four datasets, with the top row using L_1 and bottom row using L_2. Green, blue, and yellow lines are decision boundaries with low λ, and red magenta, cyan are high λ. Again, as λ increases, training error increases. The first dataset (left) is linearly separable, the middle two have some class overlap, and the right dataset is not even close. Accordingly, the decision boundary does not appear in the range of the data on the rightmost plot.

Finally, the test-set classification error is also compared across λ values and regularization methods in Fig. 4. When λ is very high for L_1 regularization (blue), test accuracy suffers, since model is too sparse. L_2 reg. has almost constant accuracy throughout.

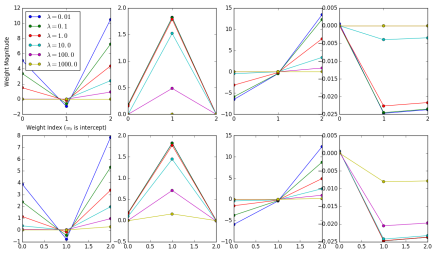

Fig. 1: Weight vectors of L_1 regularization (top row) and L_2 reg. (bottom row) for four datasets. Each column is the model for the same dataset. Within each subplot, several values of λ are shown. For high lambda, weights are smaller in magnitude for all plots. L_1 reg causes sparser weight vector (more zero elements) than L_2.

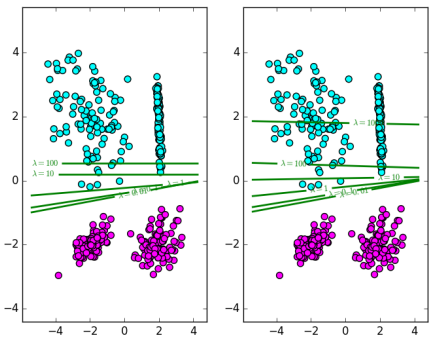

Fig. 2: LR on one dataset with L_1 reg on left, L_2 on right. Green lines show decision boundaries for various λs. As λ increases, training error increases.

C. Part 3

To pick the best regularizer and λ, the LR weights are trained with many values of λ for each regularizer, using the training set. Then, the generalization is evaluated on the validation set, by measuring the classification accuracy. A model that generalizes well will have high accuracy on data not seen during training. In cases where the validation accuracy is identical for multiple values of λ, the higher λ is chosen, because this corresponds to lower model complexity (more regularized).

Dataset	Regularizer	λ	Train Acc. (%)	Val. Acc. (%)
1	L_2	10	100.0	100.0
2	L_2	10	100.0	100.0
3	L_1	10	100.0	100.0
4	L_2	10	100.0	100.0

TABLE I: Accuracy of LR on four datasets.

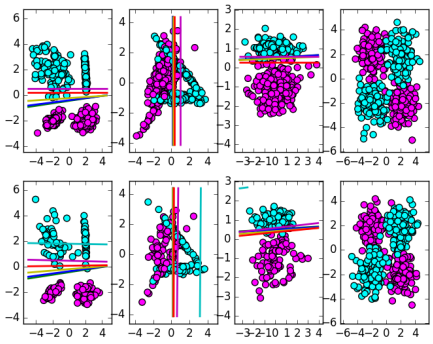

Fig. 3: LR on four datasets, with the top row using L_1 and bottom row using L_2. Green, blue, and yellow lines are decision boundaries with low λ, and red magenta, cyan are high λ. As λ increases, training error increases. The first dataset (left) is linearly separable, the middle two have some class overlap, and the right dataset is not even close. Accordingly, the decision boundary does not appear in the range of the data on the rightmost plot.

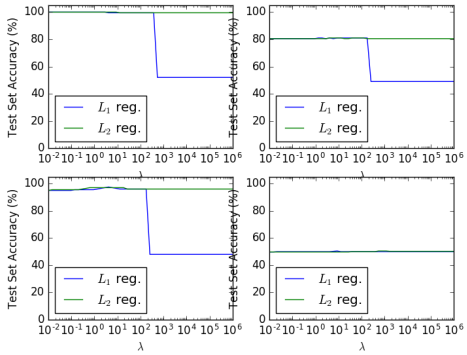

Fig. 4: For each dataset, the test accuracy is plotted across a wide range of λ values. When λ is very high for L_1 regularization (blue), test accuracy suffers, since model is too sparse. L_2 reg has almost constant accuracy throughout.

II. Support Vector Machine (SVM)

In this section, we consider the Support Vector Machine (SVM) for binary classification of several 2D datasets.

A. Part 1

First, we implemented the dual form of a linear SVM with slack variables.

[dual svm eqns]

I first converted the input data into a standard format, then we used the cvxopt Python package to execute the quadratic programming to find the desired α values.

The inputs to the cvxopt solver were:

- $P \in \Re^{nxn}$, where $P[i,j] = Y[i]Y[j]K(X[i], X[j])$
- $q = [-1, -1, ...] \in \Re^n$
- $G = [-I, I]^T \in \Re^{2nxn}$
- $h = [0, 0, ...; C, C, ...]$
- $A = Y^T$
- $b = 0$

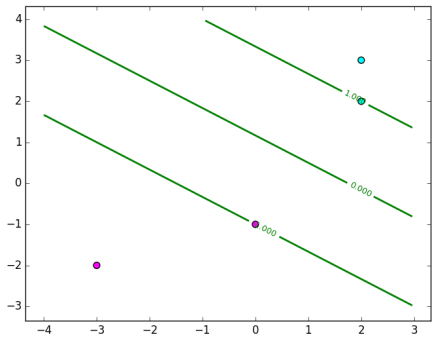

Fig. 5: Simple 4-element dataset (positive elements in cyan, negative in magenta) is separated by SVM. The three lines are the decision boundary ($\hat{y} = 0.0$) and positive and negative margin boundaries ($\hat{y} = \pm 1.0$) and two elements (support vectors) lie directly on boundaries of margin. Training error is zero.

With the simple 4 element dataset $\{((2,2),+1),((2,3),+1),((0,-1),-1),((-3,-2),-1)\}$, the algorithm outputs $(2,2)$ and $(0,-1)$ as the support vectors, meaning they lie on the boundaries of the margin, seen in Fig. 5. The weight vector has elements $w = [0.308, 0.462]$ and bias, $b = -0.538$. The training error is zero for this simple dataset. We use regularization term $C = 1$ for all results in this and the next subsection.

B. Part 2

Next, we used SVM to classify data from four much larger 2D datasets. The weights and bias of the decision boundary are generated using the training data, shown on the top row of Fig. 6, and then tested on the validation set shown in the bottom row of the same figure. Each column represents one dataset, presumably from the same distribution but with slightly different data. The leftmost dataset is easiest to linearly separate, whereas the middle two datasets have significant overlap between classes in this feature space. The rightmost dataset is not well-suited for a linear classifier because the data seems to have four distinct clusters at opposite ends of a rectangle.

The classification accuracy on the validation set matches the intuition from the plots. The first and third datasets are classified most accurately, while the fourth dataset is essentially equivalent to a random coinflip. For three of the four datasets, the training accuracy is at least as high as the validation accuracy. This is expected, because the model typically performs worse on data that was not seen during testing. In these cases, though, the accuracy difference is very minor.

Dataset	Training Accuracy (%)	Training Accuracy (%)
1	100.0	100.0
2	82.75	84.0
3	98.75	96.5
4	51.0	48.5

TABLE II: Accuracy of linear SVM on four datasets in Fig. 6.

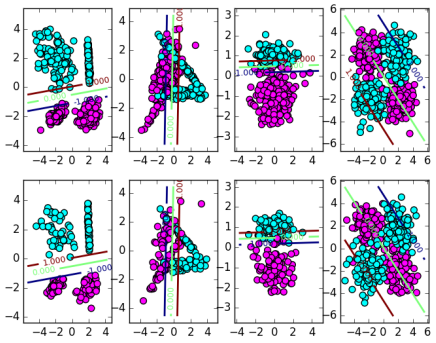

Fig. 6: Four 2D datasets (positive elements in cyan, negative in magenta) are separated by SVM. The top row is the four training datasets, and the bottom row is the four corresponding validation sets. The three lines are the decision boundary ($\hat{y} = 0.0$ in green) and positive and negative margin boundaries ($\hat{y} = \pm 1.0$ in red/blue).

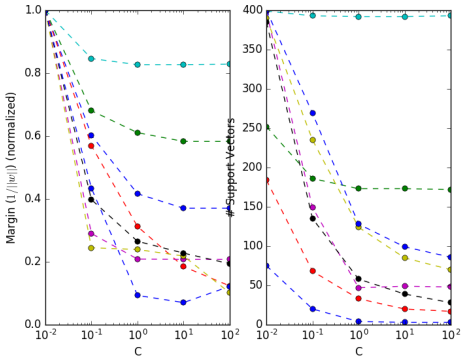

Fig. 7: For each of four datasets, and for each of two kernel functions, on the right, the margin is plotted against regularization paramter C. As C increases, generally margin decreases. The margins are normalized by their value when $C = 10^{-2}$. On the left, the number of support vectors decreases as C increases.

C. Part 3

Next, we add an ability to handle kernel functions to the dual SVM implementation. Two kernels are considered: linear kernel ($K(x, x') = x^T \cdot x$) and gaussian RBF kernel ($K(x, x') = exp(-\gamma||w - w'||^2)$). The regularization parameter, C, was set to 1 in previous sections, but here we are interested in how its value affects the results.

The decision boundaries with the two kernels on the four datasets are shown in Fig. 8. The left two datasets can be separated pretty well by a linear kernel (top row), and while the Gaussian RBF (bottom row) can be tuned to classify reasonably well, but not to an extent that is worth the extra computational cost. The rightmost dataset shows a clear example of a linear kernel failing where a gaussian RBF kernel shines. The data is not linearly separable in the original feature space, so the linear kernel's accuracy is about as bad as guessing. The Gaussian RBF kernel encloses the four clusters and classifies accurately. It could be argued that this decision boundary overfits the data, especially clear around the edges of the boundaries. The C parameter could be tuned to adjust this effect, but the computation time involved is significant. The bandwidth parameter, γ was adjusted for each dataset to get this performance (0.01 for the left two plots, 1.0 for the right two), and this parameter affects the spread of the kernel.

On the left of Fig. 7, the geometric margin ($1/||w||$) decreases as C increases in general. These results are shown for each of the four datasets, for the two kernel functions, so there are 8 curves. In one case, the margin increases after an increase in C. It makes sense that margin would generally decrease with an increase in C, because more penalty is being applied to incorrectly classified samples, and therefore $||w||$ will increase to overfit the dataset slightly more and reduce the number of incorrectly classified samples.

On the right of Fig. 7, a similar concept applies to the number of support vectors as C increases. Any sample with non-zero α_i is a support vector, including mis-classified samples, and with small C, the penalty of having many support vectors is low. As C increases, the margin shrinks and the number of support vectors decreases.

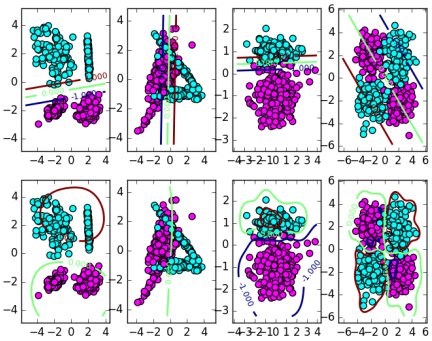

Fig. 8: Each column is one validation set, with the top row showing the SVM decision boundary using a linear kernel, and the bottom row using a Gaussian RBF kernel. The Gaussian RBF can generate curved decision boundaries which is necessary for the rightmost dataset. The left datasets can be accurately/simply classified with a linear kernel, which is also less computationally intensive.

Maximizing the geometric margin on the training set is not an appropriate criterion for selecting C because that method will simply choose the weights corresponding to C=0. This eliminates any allowance of slack, where misclassified points can be ignored to improve the model's ability to generalize. An alternative method to choose C is to train many different SVM classifiers with different C values, and evaluate the classification accuracy on a validation set. Then, choose the value of C that corresponds to high accuracy.