ML-PA2

Manthan Thakar

February 2018

Note

The code for Logistic regression can be found in PA2_code/logreg_test.py and for SVM in PA2_code/svm_test.py.

1 Logistic Regression

1.1 Implementation

We are given the negative log likelihood of logistic regression loss function with target values $y \in \{-1, 1\}$ as :

$$NLL = \sum_{\mathbf{n=1}}^{\mathbf{N}} \mathbf{ln}(\mathbf{1} + \mathbf{exp}(-\mathbf{yt}) + \lambda||\mathbf{w}||^{\mathbf{2}}$$

.

To get the optimum values for w, we take the derivative of the NLL w.r.t w and we get following,

$$\sum_{n=1}^{N}(\frac{e^{(-y_i w^T x)}}{1 + e^{(-y_i w^T x)}} + 2\lambda w)$$

We use the gradient descent to optimize the parameters using the gradient of NLL obtained above.

Mistakes

After optimizing using gradient descent, we get 0 mistakes on training data set and 4 mistakes on the validation set for linearly separable data.

On non-linearly separable data, we observe 30 mistakes on training data and 66 mistakes on validation set.

On nonlin data set, we observe 69 mistakes on training data and 130 mistakes on validation set.

All the results reported above are for $\lambda = 0$.

1.2 Testing

For $\lambda = 0$, following results are observed for all data sets for training data set.

The results are displayed in Figure 1, Figure 2 and Figure 3.

Figure 1: Logistic Regression on linearly separable data $\lambda = 0$

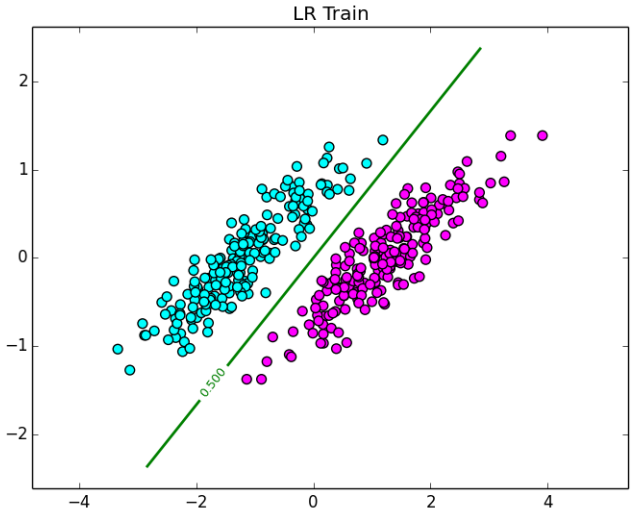


Figure 2: Logistic Regression on not linearly separable data $\lambda = 0$

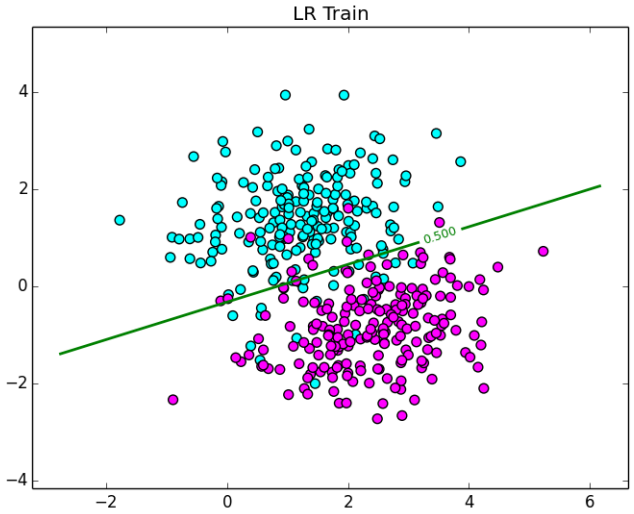
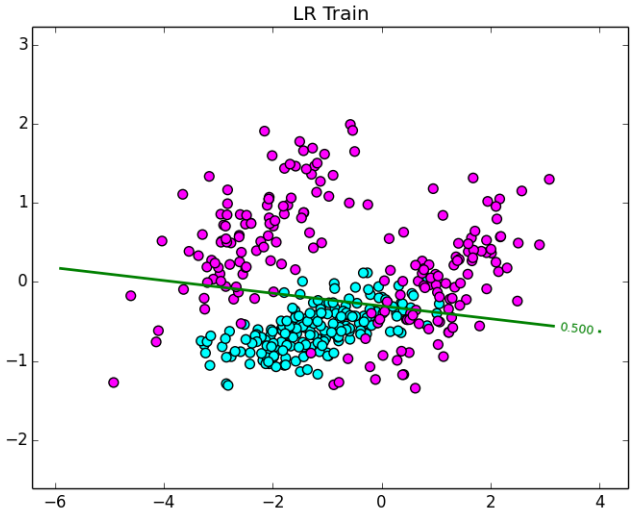


Figure 3: Logistic Regression on non-linear data $\lambda = 0$



Here, it can be noted that, the classifier tries to classify data exactly correctly for the train data set, therefore we see the line for linearly separable data. This might lead to overfitting if the validation set doesn't have similar patterns.

We can see similar things for not linearly separable data set as well. For non linear data, we see that the linear classifier makes a lot of mistakes due to the nature of the data.

Increasing the value of λ

Next, we increase the value of λ to see the trend. For the linearly separable case, we see that the line moves towards pink dots to generalize the classifier for unseen data.

Interesting results are obtained when λ is increased for nls data set. The fit for training set can be seen in Figure 4 and Figure 5.

If we compare the results from Figure 4 (where $\lambda = 0.2$) and Figure 5 (where $\lambda = 0.8$) with Figure 1 (where $\lambda = 0$) we observe that decision boundary keeps getting horizontally aligned as we increase the value of λ.

A note on values of λ It is to be noted that for all the data sets, when we increase the value of λ to a certain level (¿ 1), it results in exp overflow error. This could be because of the small values obtained in the dot product of weights and features. To accommodate such values, it was helpful to decrase the step size.

Figure 4: Logistic Regression on nls data $\lambda = 0.2$

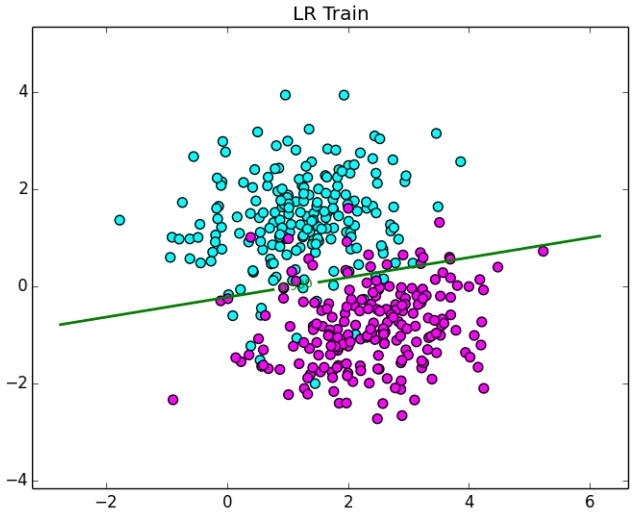


Figure 5: Logistic Regression on nls data $\lambda = 0.8$

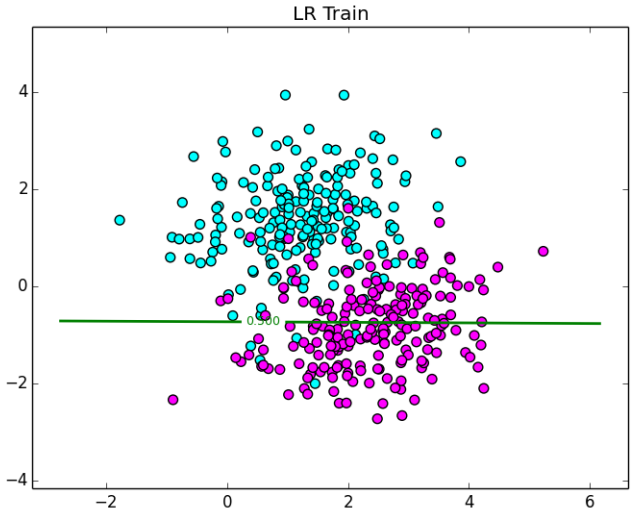
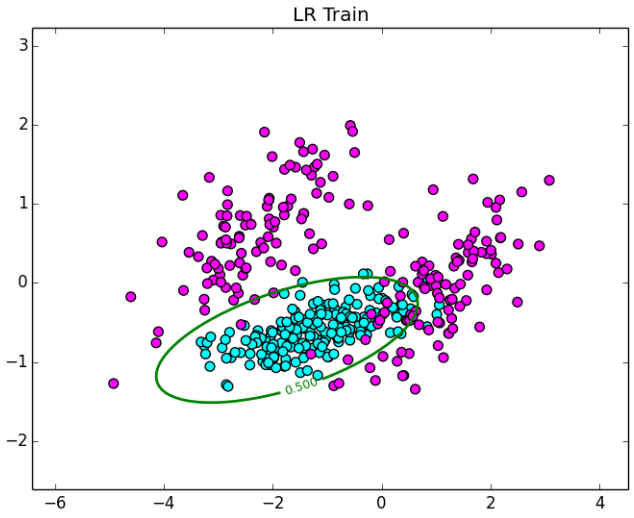


Figure 6: Polynomial Logistic Regression on nonlin data (training) $\lambda = 0$



1.3 Polynomial Logistic Regression

Applying polynomial basis expansion function on input features yields a non-linear decision boundary for all cases. The most interesting results are yielded for the non-linear data set which are attached below in Figure 6 and Figure 7 for $\lambda = 0$.

We see that the decision boundary is elliptical for nonlin data set and fits the data almost perfectly.

But it can also be seen that the ellipse is closed an does not account for data points that might occur outside of it. Increasing λ gives us that generalization. In figure 8, we can see that for $\lambda = 0.1$ the ellipse becomes a bit wider to accommodate more points for blue class.

2 SVM Implementation

Using the dual form of SVM and cvxopt we obtain the support vector weights for linear SVM.

We try different values of C for all data sets. As shown in Figure 9 and Figure 11, when the value of C increases, the margins get tighter as compared to smaller values of C.

For linear SVM ($C = 0.05$), on linearly separable data we get 0 mistakes on train set and 3 mistakes on validation set. For nls data, we get 31 mistakes on train set and 67 on validation set. For nonlin data, we get 67 mistakes on train set and 126 mistakes for validation set.

Figure 7: Logistic Regression on nonlin data (validation) $\lambda = 0$

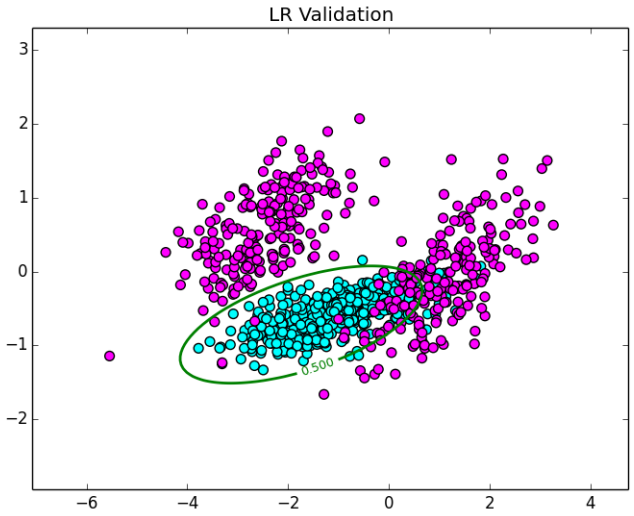


Figure 8: Logistic Regression on nonlin data (train) $\lambda = 0.1$

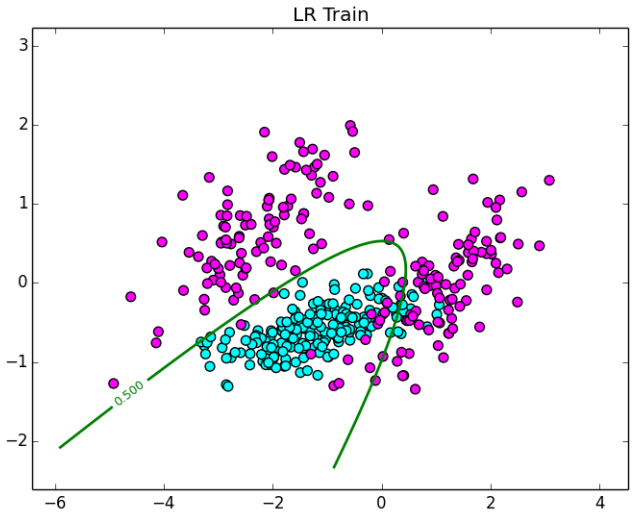
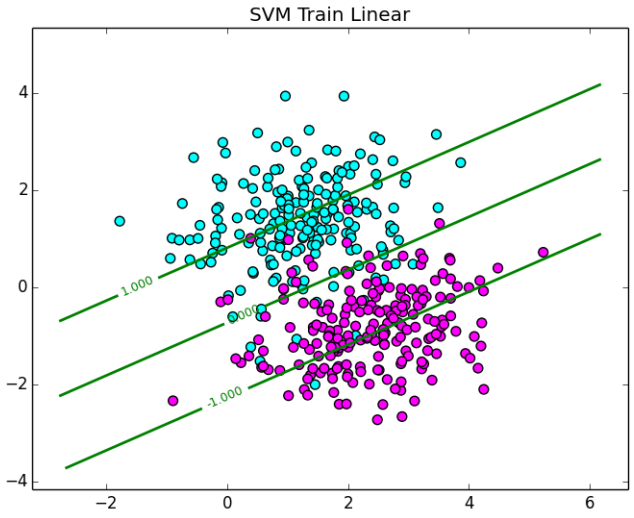


Figure 9: Linear SVM on nls data (training) $C = 0.05$



For linear SVM ($C = 10$), on linearly separable data we get 0 mistakes on train set and 4 mistakes on validation set. For nls data, we get 31 mistakes on train set and 69 on validation set. But For nonlin data, we get 61 mistakes on train set and 125 mistakes for validation set. So we see fewer mistakes on nonlin dataset for higher value of C.

We also observed that the linear SVM just like linear logistic regression doesn't give a decision boundary for non-linear data and nls data.

3 Kernel SVM

Now we apply Polynomial and Gaussian kernels to SVM to get better decision boundaries for non lin and nls data sets. The general observation is that, for higher values of C we get tight margins.

Gaussian Kernel

For gaussian kernel, we see that when the value of σ is decreased, the decision boundary gets very complicated and starts to form clusters, which indicates overfitting of data. For example, figure 13 and figure 14 shows different values for σ and C is kept constant at 0.5. We see that, for $\sigma = 0.5$ the decision boundaries are more complicated and tighter than $\sigma = 2.5$. The results get even more complicated when $\sigma = 0.05$ in figure 15. As expected, the fit in figure 15, gives bad results for validation set.

Polynomial Kernel

For polynomial kernel, we experiment with different values of C. In figure 16 and Figure 17 we see the difference in decision boundaries for small and big

Figure 10: Linear SVM on nls data (validation) $C = 0.05$

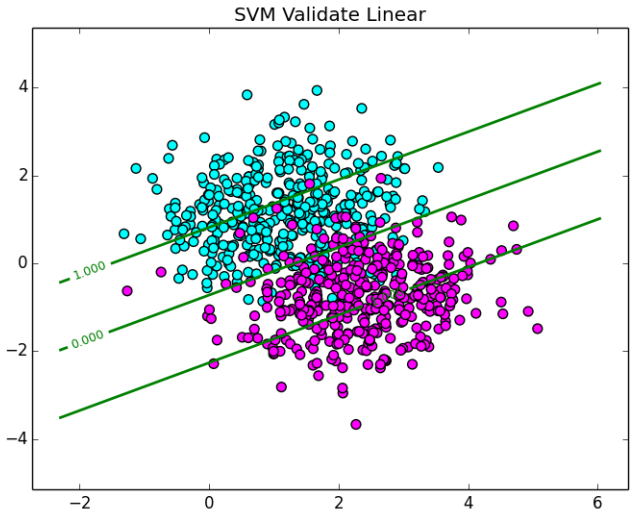


Figure 11: Linear SVM on nls data (training) $C = 20$

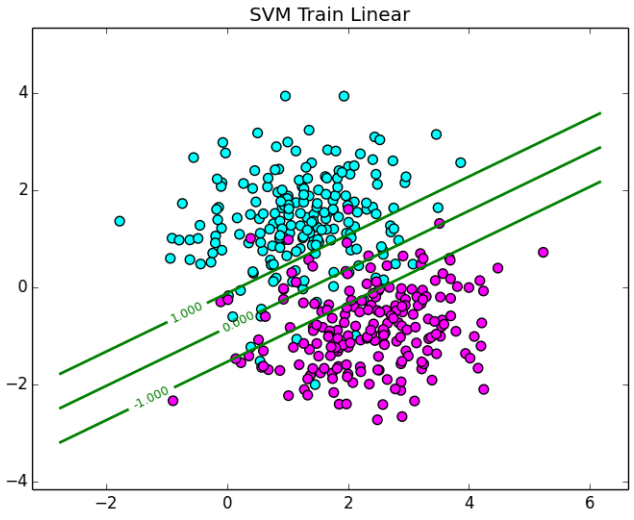


Figure 12: Linear SVM on nls data (validation) $C = 20$

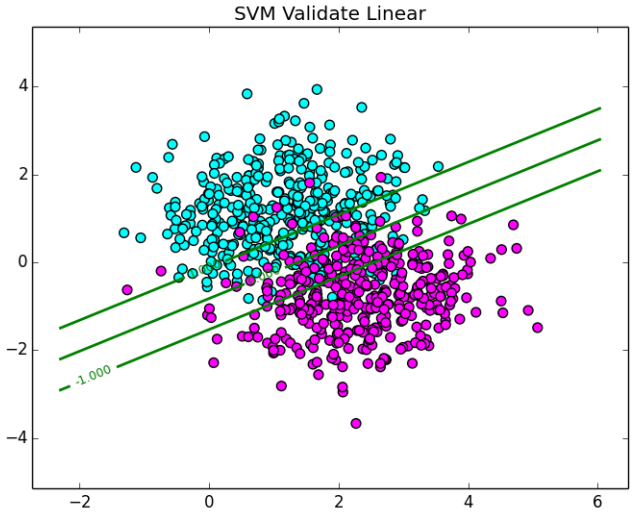


Figure 13: Gaussian SVM on nonlin data $C = 0.5\sigma = 0.5$

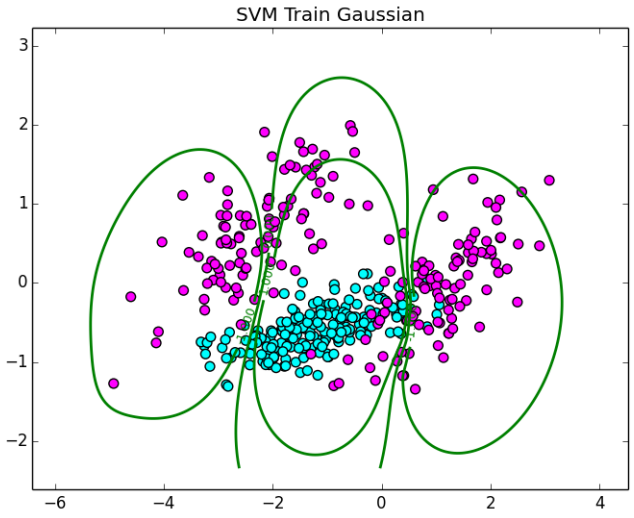


Figure 14: Gaussian SVM on nonlin data $C = 0.5\sigma = 2.5$

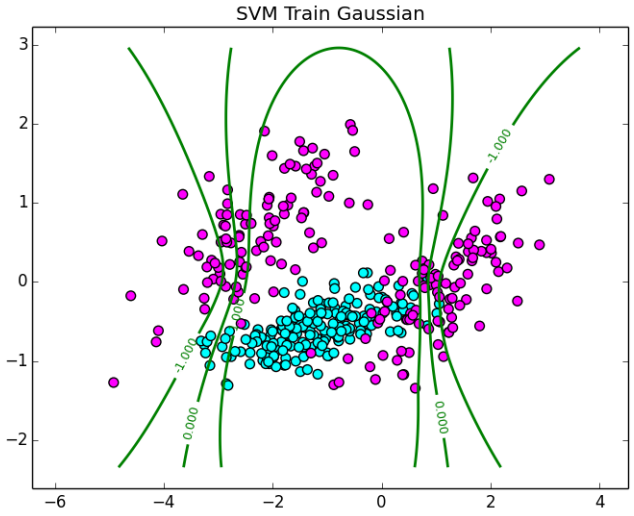


Figure 15: Gaussian SVM on nonlin data $C = 0.5\sigma = 0.05$

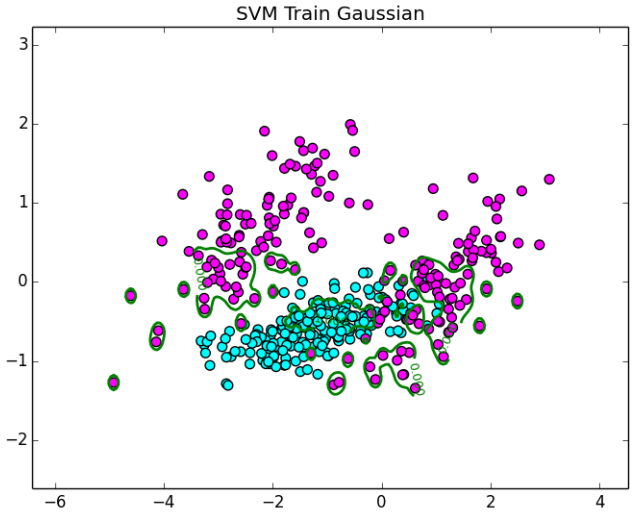
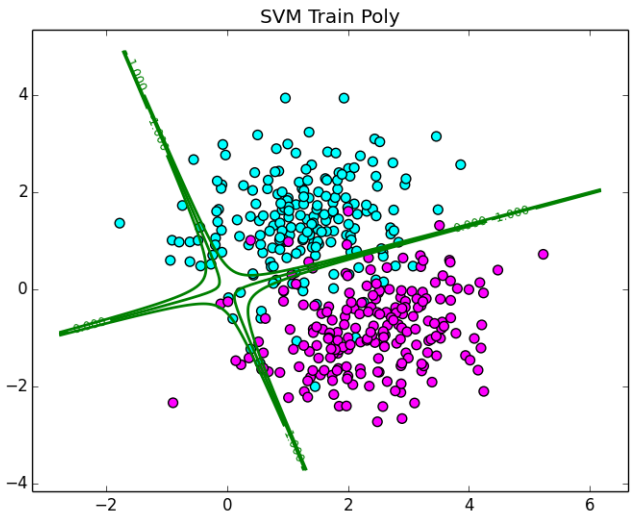


Figure 16: Polynomial SVM on nls data $C = 0.5$



value of C. Here we see that, for C = 0.5, the margins are bigger than C = 10 where margins are small.

Figure 17: Polynomial SVM on nls data $C = 10$