Support Vector Machines and Ensemble Methods

Algorithmic Data Analysis – Coding Assignment 1

Giulia Ortolani

January 25, 2024

Contents

1 Task 1

In this task I'm implementing the linear SVM algorithm with hard-margin and soft-margin variants. Then, I'm applying the SVM with hard-margin to the irisSV dataset and the SVM with soft-margin to the irisVV dataset.

1.1 SVM with hard-margin – irisSV dataset

After dividing the irisSV dataset into training (4/5) and test (1/5) subsets, I've trained the hard-margin SVM on the training subset and applied the resulting model to the test subset. Figure 1 shows the plot of the hyperplane with the support vectors highlighted.

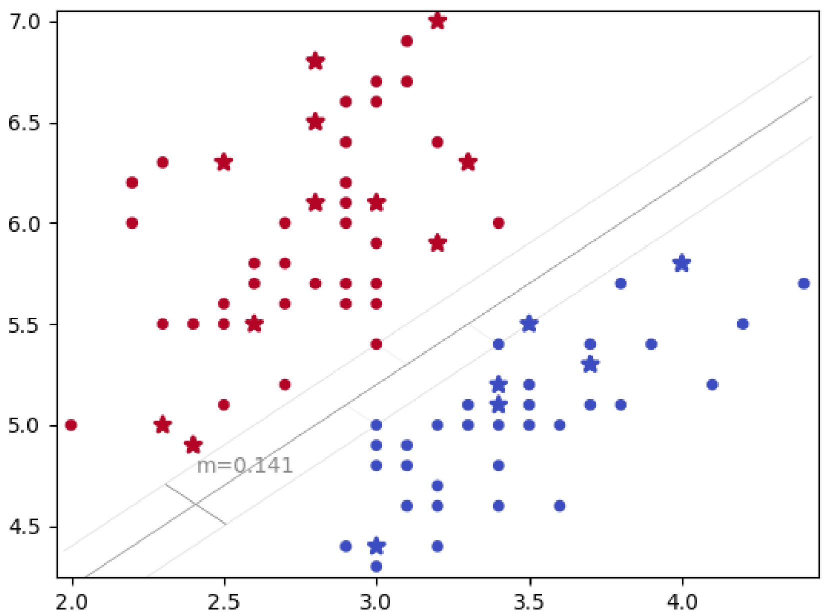

Figure 1: SVM with hard-margin – irisSV dataset

1.1.1 Confusion matrix

Actual / Predicted	Iris setosa	Iris versicolor
Iris setosa	6	0
Iris versicolor	0	11

1.1.2 Accuracy, recall and precision

$$\begin{aligned} \text{Recall:} &\quad 1.0 \\ \text{Precision:} &\quad 1.0 \\ \text{Accuracy:} &\quad 1.0 \end{aligned}$$

1.1.3 Equation of the separating hyperplane

$$\begin{bmatrix} 5 & -5 \end{bmatrix} \cdot x^T - 10.99 = 0$$

1.2 SVM with soft-margin – irisVV dataset

After dividing the irisVV dataset into training (4/5) and test (1/5) subsets, I've trained the soft-margin SVM on the training subset setting $c = 2$ and applied the resulting model to the test subset.

Figure 2 shows the plot of the hyperplane with the support vectors highlighted.

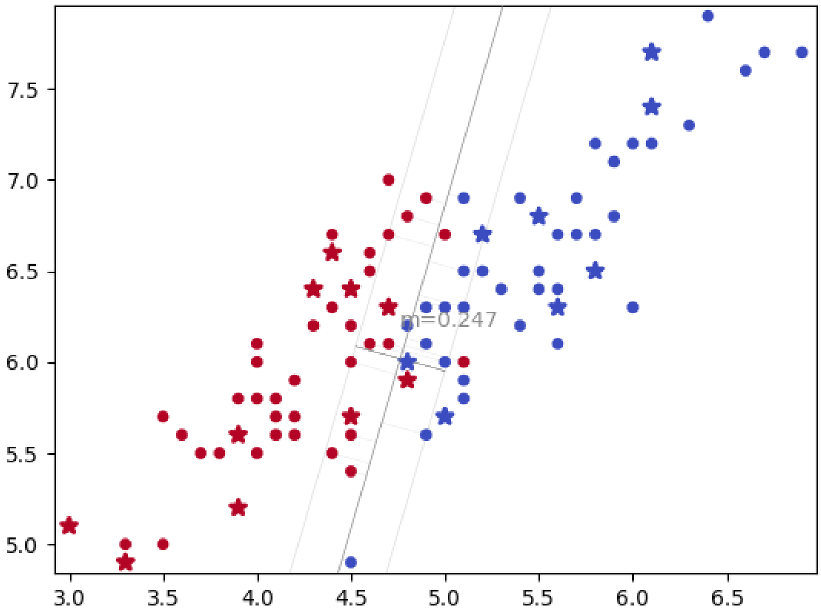

Figure 2: SVM with soft-margin – irisVV dataset

1.2.1 Confusion matrix

Actual / Predicted	Iris setosa	Iris versicolor
Iris setosa	8	1
Iris versicolor	0	9

1.2.2 Accuracy, recall and precision

$$\begin{aligned} \text{Recall:} &\quad 0.9 \\ \text{Precision:} &\quad 1.0 \\ \text{Accuracy:} &\quad 0.94 \end{aligned}$$

1.2.3 Equation of the separating hyperplane

$$\begin{bmatrix} 1.1 & -3.9 \end{bmatrix} \cdot x^T - 11.96 = 0$$

2 Task 2

Here I run an evaluation of the soft-margin SVM on the irisVV dataset with 5 fold cross-validation (10 rounds).

	Round									
Fold / Accuracy	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1/5	0.88	1.0	0.94	0.88	0.88	0.94	0.94	1.0	0.82	1.0
2/5	1.0	0.94	1.0	0.94	0.88	0.94	0.94	1.0	1.0	1.0
3/5	0.88	0.82	0.94	0.88	0.94	0.94	0.94	0.94	0.94	0.94
4/5	0.94	0.82	0.94	1.0	1.0	0.94	0.94	0.88	0.94	0.94
5/5	1.0	1.0	0.88	0.94	1.0	1.0	0.94	1.0	0.94	0.76
Mean Accuracy	0.94	0.92	0.94	0.93	0.94	0.95	0.94	0.96	0.93	0.93
Variance of Accuracy	0.0028	0.0064	0.0014	0.0019	0.0028	0.0006	0.0	0.0022	0.0033	0.0075

Table 1: Soft-Margin SVM Evaluation Results

Moreover, averaging the accuracy and the variance obtained for each round we can obtain:

$$\begin{aligned} \text{Overall Mean Accuracy:} &\quad 0.939 \\ \text{Overall Variance of Accuracy:} &\quad 0.0029 \end{aligned}$$

3 Task 3

These are the results of applying AdaBoost with different value for β:

Error rate					
Beta	**1**	**0.8**	**0.5**	**0.2**	**0**
Classifier n.1	0.2225	0.22625	0.23	0.2275	0.235
Classifier n.2	0.21625	0.20375	0.225	0.21375	0.24
Classifier n.3	0.21375	0.21875	0.22125	0.22625	0.23625
Classifier n.4	0.21375	0.215	0.22	0.22125	0.23
Classifier n.5	0.2225	0.2175	0.2275	0.22375	0.23375
Classifier n.6	0.22125	0.2175	0.2225	0.21375	0.23
Classifier n.7	0.215	0.21625	0.2225	0.21875	0.23
Classifier n.8	0.21625	0.215	0.22	0.22125	0.225
Classifier n.9	0.2225	0.22125	0.2225	0.21625	0.22
Classifier n.10	**0.2225**	**0.22375**	**0.225**	**0.21375**	**0.23**
Metrics					
Recall	0.4407	0.4068	0.4237	0.4237	0.3729
Precision	0.6341	0.6154	0.6410	0.6098	0.6286
Accuracy	0.76	0.75	0.76	0.75	0.75
Bias	3.0819	11.0102	7.5730	10.9646	2.3787

Table 2: AdaBoost Results for Different β Values

If $\beta = 0$ we're not weighting the samples, that is we're not improving our model and the AdaBoost implementation is pointless: we're just taking our classifier and train it on a dataset in which each data point has the same weight.
We can see that with $\beta \neq 0$ the error rate is slightly lower.

4 Task 4

These are the performances of bagging applied to a SVM with a RBF kernel on the credit dataset:

Metric	Value
Recall	0.7119
Precision	1.0
Accuracy	0.915

This is the comparison of the application of boosting and bagging when combined to a linear SVM vs. to a SVM with a RBF kernel on the credit dataset:

Metric	Linear Kernel	RBF Kernel
Beta	0.5	0.5
Error Rates		
Classifier n.1	0.239	0.113
Classifier n.2	0.221	0.153
Classifier n.3	0.227	0.193
Classifier n.4	0.219	0.222
Classifier n.5	0.225	0.245
Classifier n.6	0.226	0.257
Classifier n.7	0.228	0.271
Classifier n.8	0.229	0.277
Classifier n.9	0.231	0.279
Classifier n.10	0.229	0.28
Accuracy	0.77	0.705

Table 3: Comparison between Linear and RBF Kernels

We can see that the error rates with the RBF kernel are increasing with respect to the iteration, and the accuracy is lower.