CMPE 462 - Project 2
Implementing an SVM Classifier
Due: May 18, 2020, 23:59

Task 1 - 30 pts

In order to achieve a hard-margin linear model, we trained the SVM model using the linear kernel and choosing $C = 10^8$. We chose C as such, because for values less than 10^8, there were still some bounded support vectors (where $\alpha_s = C$) that lied within the margins. When we set $C = 10^8$, all support vectors were free (where $0 < \alpha_s < C$), hence our model was hard-margin. However, test accuracy turned out to be higher than training accuracy, which was unexpected. We believe this may be due to insufficient training (because of scarce data and inadequate split ratio of training/test) therefore, we shuffled the data and reassigned %80 of it as training and rest as test. Even after applying these, we observed the same situation, where test accuracy was higher than training. Therefore, after consulting Inci Hoca, we fixed the seed number at 100, where training accuracy was higher, giving a more 'expected' result.

For 150-sample training data:

- Training accuracy = 74.6667% (112/150)
- Test accuracy = 77.5% (93/120)

For 216-sample randomized training data:

- Training accuracy = 79.1667% (171/216)
- Test accuracy = 72.2222% (39/54) (40/54)

Task 2 - 40 pts

When training the soft-margin SVM, we checked values of $C = [10^{-2}, 2^{-1}, \ldots, 10^{2}]$ using linear, polynomial, RBF and sigmoid kernels. Resulting accuracy values are shown below.

For 150-sample training data:

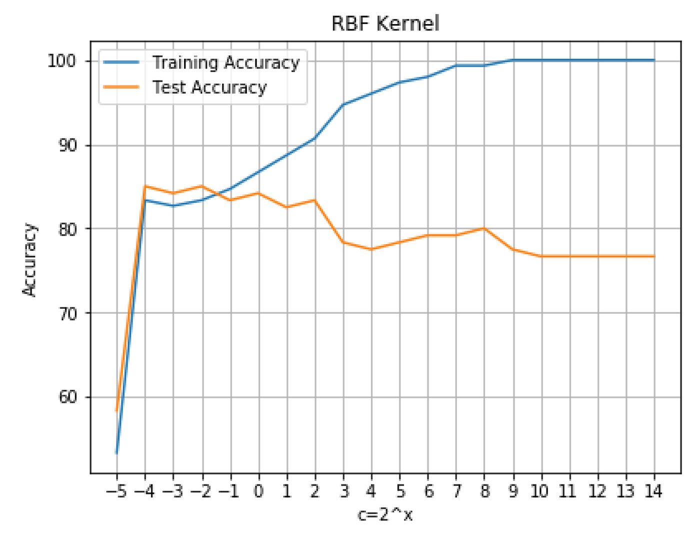
	Linear	Polynomial	RBF	Sigmoid
C=0.01 - Train	82.67	53.33	53.33	53.33
C=0.01 - Test	84.17	58.33	58.33	58.33
C=0.1 - Train	86.0	53.33	83.33	82.0
C=0.1 - Test	83.33	58.33	84.17	84.17
C=1 - Train	86.67	86.0	86.67	82.67
C=1 - Test	85.0	82.5	84.17	84.17
C=10 - Train	88.67	94.0	95.33	78.0
C=10 - Test	81.67	80.83	77.5	80.0
C=100 - Train	88.67	98.67	99.33	76.67
C=100 - Test	81.67	75.0	78.33	72.5

For 216-sample randomized training data:

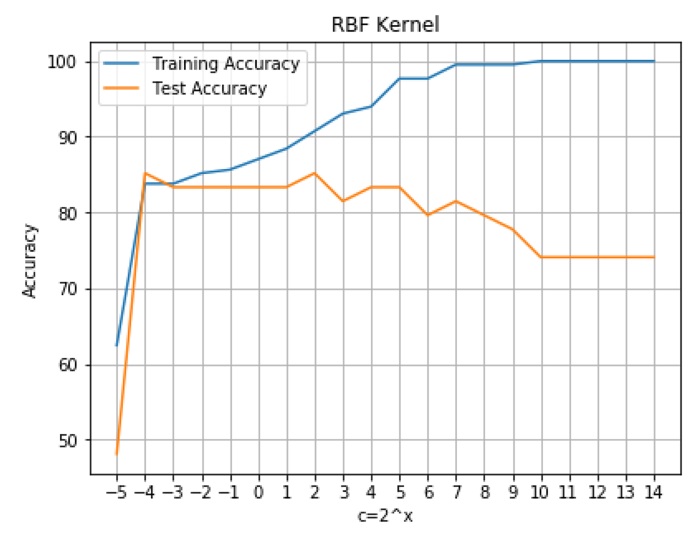
	Linear	Polynomial	RBF	Sigmoid
C=0.01 - Train	84.26	57.41	57.41	57.41
C=0.01 - Test	83.33	48.15	48.15	48.15
C=0.1 - Train	85.19	57.41	84.72	83.33
C=0.1 - Test	85.19	48.15	83.33	83.33
C=1 - Train	86.11	85.65	87.04	83.8
C=1 - Test	85.19	83.33	83.33	81.48
C=10 - Train	84.72	93.06	93.06	81.02
C=10 - Test	83.33	83.33	79.63	79.63
C=100 - Train	84.72	99.07	99.07	78.24
C=100 - Test	83.33	74.07	81.48	68.52

We observed that as we increased C the model began to overfit, as training accuracies increased and test accuracies decreased (shown in the graph below using RBF kernel).

For 150-sample training data:



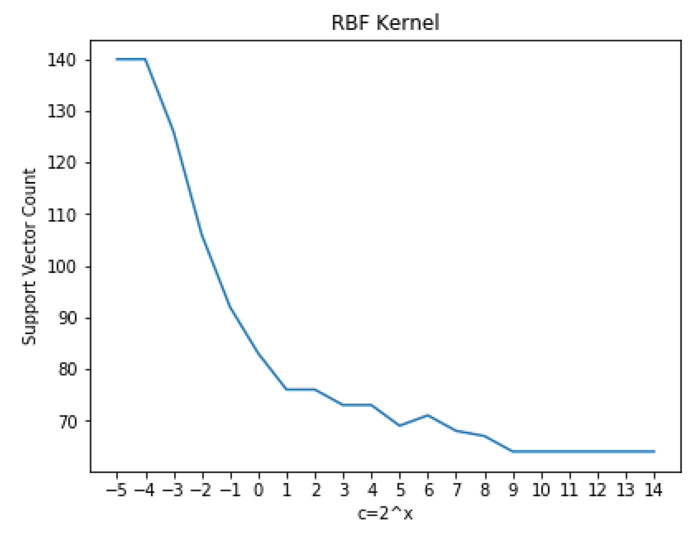
For 216-sample randomized training data:



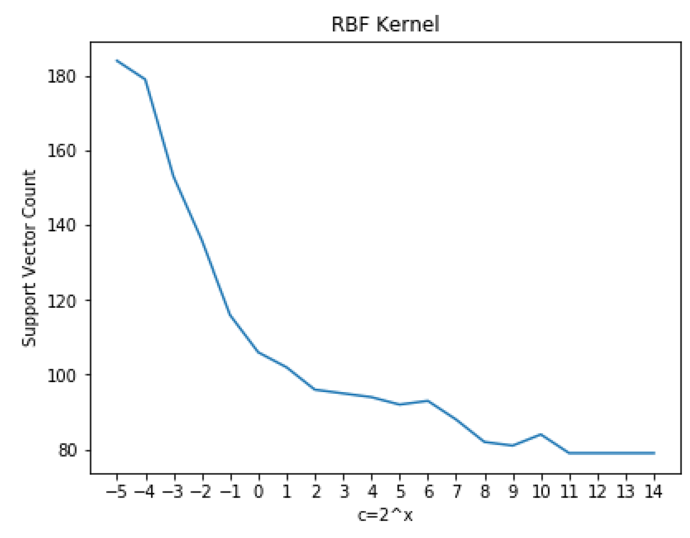
Task 3 - 15 pts

We observed that the number of support vectors decreased as we increased C. This is in accordance with the theory, since as C increases, the model becomes less tolerant to margin violations and therefore in order to decrease such violations, the margin is narrower and the number of support vectors decrease.

For 150-sample training data:



For 216-sample randomized training data:



Task 4 - 15 pts

When we removed a data point that was also a support vector, we observed that mean-squared difference between the original and the new w $(= [b, w]^T)$ was 10.39. When we removed a data point that was not a support vector, we observed that mean-squared difference between the original and the new w was 0.00198, much less than the first one. This was expected, since support vectors have much larger effect on the w than ordinary data points.

Bonus Task - 10 pts

Using the QP solver in the CVXOPT module, we implemented the hard margin SVM on the toy data in the SVM slide at page 14, and reached the same result: $u* = [-1, 1, -1]$