Assignment -3

NANDINI NAIR

2023-12-06

PART- A

Q1.

In Support Vector Machines (SVMs), margins represent the space between the decision boundary and the closest data points from each class. A hard margin in an SVM defines a boundary that perfectly separates all data points without any classification errors. This scenario assumes linear separability, where the boundary cleanly divides the classes.However, in real-world scenarios, data is often not linearly separable, making hard margins impractical. Soft margins address this limitation to some extent. They introduce flexibility by allowing a certain degree of classification error. This is facilitated by the use of slack variables, relaxing the strictness of the constraints in SVMs.Soft margins, typically controlled by a regularization parameter (C), strike a balance between maximizing the margin and tolerating some classification errors. This parameter helps manage the trade-off between a wider margin and permitting a controlled number of errors or margin violations in the training data. Furthermore, when dealing with nonlinearly separable data, the kernel trick becomes relevant in SVMs. It enables SVMs to handle complex patterns by implicitly transforming the data into a higher-dimensional space, where a linear separation might be feasible.

Q2.

The parameter ‘C’ in Support Vector Machines (SVMs) signifies the tolerance level for misclassification. A larger or infinite ‘C’ value implies a stricter classifier that allows no margin violations or errors in classification. Conversely, a smaller ‘C’ value denotes a more flexible classifier capable of accommodating some margin violations or training errors.SVMs are often viewed as an optimization challenge where a balance is sought between maximizing the margin separating classes while minimizing classification errors. This delicate trade-off defines the essence of ‘C’ in SVMs, aiming to achieve an optimal balance between precision and generalization.

Q3.

The given inputs and their corresponding mappings are as follows:

- 0.1 maps to 0.8
- 2.8 acts as the threshold
- 11.1 maps to -0.2

Upon computing the weighted sum (0.1 * 0.8 + 11.1 * -0.2 = -2.14), the resulting value, -2.14, falls below the activation threshold of 2.8.

Hence -2.14 is less than the threshold, the perceptron remains inactive based on this calculation.

Q4.

The delta rule, essential for training perceptron learners, comprises three key components. The third element focuses on adjusting the weights to align the current output closer to the desired one. This adjustment process relies on a parameter known as alpha or the learning rate.Alpha determines the extent of weight updates during training. If set too high, there's a risk of overshooting the optimal values. Conversely, a very low alpha prolongs convergence. Essentially, alpha governs the step size in gradient descent, controlling the magnitude of weight adjustments.Commonly, alpha starts with a higher value during initialization, gradually decreasing over time. This strategy ensures a balance: learning rates aren't excessively slow, yet they avoid overshooting optimal values. The dynamic adjustment of alpha aims for effective convergence without compromising accuracy.