CS244 Exercise 7

Task 1: SVMs

In some cases, feature scaling is important and in other cases it is not. True or False: feature scaling is important when using SVMs with the RBF kernel.

- True
- False

The parameter σ in the RBF kernel is similar to the parameter *k* in *k*-nearest-neighbor algorithms. One possible way to combat overfitting is to:

- Increase σ
- Decrease σ

There are different approaches to lessen the problem of overfitting. Circle all of the approaches below that are likely to combat overfitting.

- Gather more training examples
- Reduce the number of features
- Add additional features, e.g., generate polynomial features
- Decrease the regularization parameter λ

Task 2: Data Leakage

Data leakage occurs when testing data are used to inform decisions about a machine learning model. This is problematic because a model's performance on the testing data will no longer be a good indicator of how the model will perform on new data, and an important measure of a model is how it will perform on new data (which we can no longer evaluate because the testing data have been compromised). Which of the following is an example of data leakage?

- Feature scaling is performed on an entire data matrix X (before X is split into training and testing data).

- We split our data into training (80%) and testing (20%). We train a model on the training data and evaluate its performance on the testing data. The performance is good, but we wonder if our 80/20 train/test split was representative. We try splitting the data again using a different 80% of the data for training and a different 20% for testing. The performance is better so we use this new 80/20 split that leads to improved performance.

- Several different machine learning algorithms are trained and the algorithm that performs best on the testing data is selected as the one that will be used going forward.

- We split our data matrix X into training and testing data. We train a model using the training data and we evaluate the model using the testing data. The performance is disappointing so, as an alternative, we try training a model using the entire data matrix X and we then evaluate the model using the entire matrix X. The performance has improved so we use this new model that is trained on a larger set of data.

- For a neural network, we're not sure what architecture to use, so we try different numbers of hidden layers and different numbers of nodes per layer. The architecture that performs best on the testing data is the one we ultimately keep.

- All of the above

Download the Jupyter Notebook for Exercise 7 from the course website. Open the Notebook in your web browser and work through it. As you work through the Notebook, answer the following questions.

Task 3: Nonlinear Data

What is the accuracy of your SVM on the *training* data? What is the accuracy of your SVM on the *testing* data?

What is the accuracy of the *linear* kernel SVM on the *training* data? What is the accuracy of the *linear* kernel SVM on the *testing* data?

What values for parameter `C` and parameter `gamma` lead to the highest accuracy on the testing data? Using these two optimal parameter values, what is the accuracy of your SVM on the *training* data and on the *testing* data?

Task 4: Face Recognition

How many features does each image have?

Following our PCA, now how many features does each example have?

What is the accuracy of your SVM on the *training* data? What is the accuracy of your SVM on the *testing* data?

What values for parameter `C` and parameter `gamma` lead to the highest accuracy on the testing data? Using these two optimal parameter values, what is the accuracy of your SVM on the *training* data and on the *testing* data?

For whose images did the SVM achieve the highest performance? For whose images did the SVM achieve the lowest performance?

Task 5: Recycling, Again

What is the accuracy of your SVM on the *training* data? What is the accuracy of your SVM on the *testing* data?

Task 6: Free Apps from Google Play

What was the highest accuracy you obtained on your *testing* data? What values for parameters `C` and `gamma` resulted in this accuracy?