EECE 5644 Homework 3

Emily Costa (costa.em@northeastern.edu)

April 6, 2021

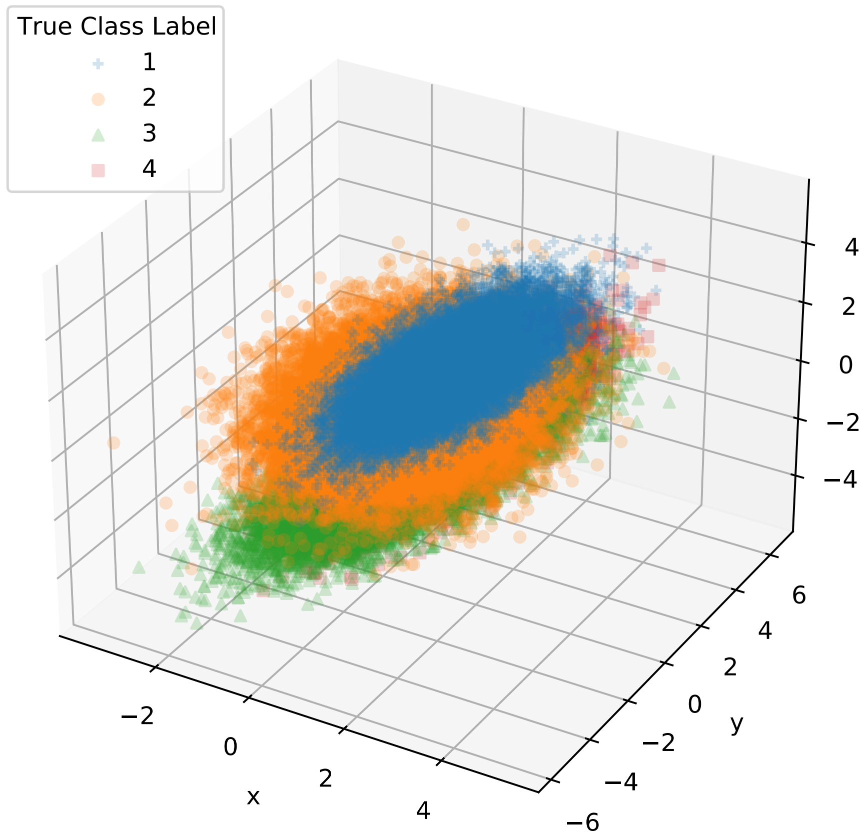

Figure 1: *The non-classified four-class Gaussian distribution with uniform priors. This data set was used for testing the accuracy of the developed models.*

1 Multilayer Perceptrons (MLP)

In this section, I train an MLP model to approximate class label posteriors of a four-class Gaussian distribution using maximum likelihood parameter estimation. I use the trained models to approximate a MAP classification rule in an attempt to achieve minimum probability of error (i.e. to minimize expected loss with 0-1 loss assignments to correct-incorrect decisions).

First, I generate data sets for training and testing using a four-class Gaussian class-conditional probability distribution function (PDF). Figure 1 provides an example of one of the generated data set, the validation data set. The mean vectors and covariance matrices of the 4 class are as follows:

$$\sigma_1^2 = \begin{bmatrix} 1 & 0.5 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \sigma_2^2 = \begin{bmatrix} 0.5 & 0 & 1 \\ 1.2 & 1 & 0.8 \\ 1 & 0.6 & 0 \end{bmatrix}, \sigma_3^2 = \begin{bmatrix} 1 & 0.5 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \sigma_4^2 = \begin{bmatrix} 0.5 & 0.5 & 0.5 \\ 1.4 & 1.2 & 0.7 \\ 0.6 & 1.1 & 0.5 \end{bmatrix}$$

$$\mu_1 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}, \mu_2 = \begin{bmatrix} 1 & -1 & 1 \end{bmatrix}, \mu_3 = \begin{bmatrix} 1 & -1 & -1 \end{bmatrix}, \mu_4 = \begin{bmatrix} 1 & 1 & -1 \end{bmatrix}$$

Next, I used 10-fold cross-validation to determine the optimal number of perceptrons, a hyperparameter, to

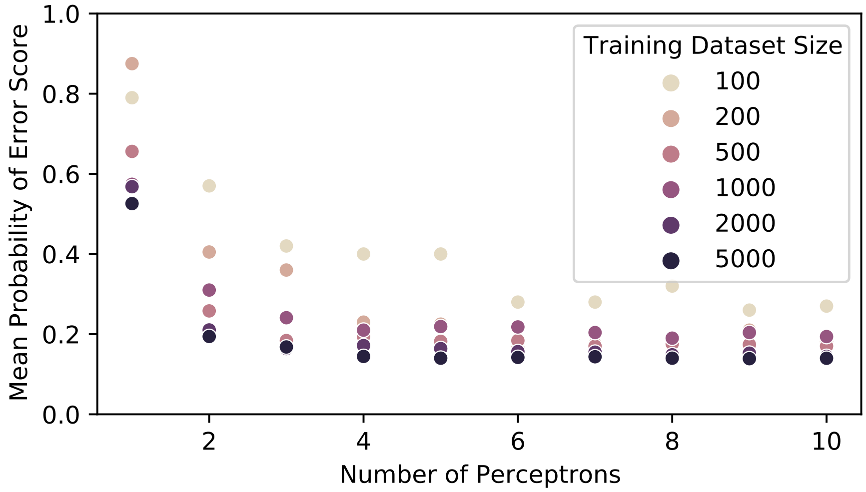

Figure 2: *Trends in cross-validation predicted accuracy of the model based on varying model hyperparameter for various sized data sets.*

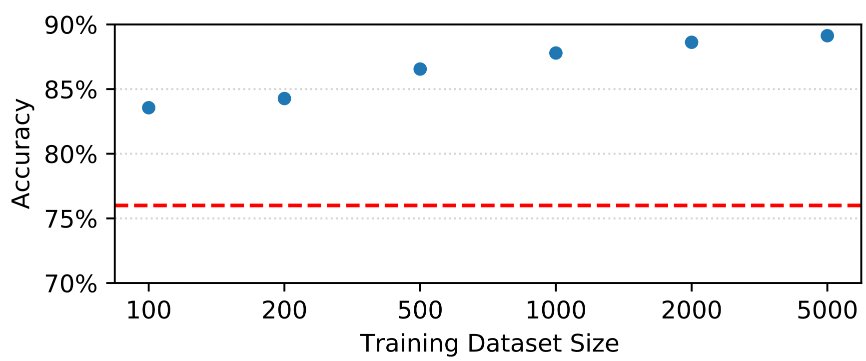

Figure 3: *Model accuracy based on optimal hyperparameter identified by cross-validation. The red line indicates to optimal theoretical classification according to the minimum-probability-of-error classification rule.*

use when training my MLP model for each of the training data sets. Figure 2 shows the predicted accuracy a model will have depending on the size of the training data set and the number of perceptrons used as a hyperparameter. This assignment requested that the perceptrons with the minimum classification error probability for each training set be used in training the model, which is what I used. However, Fig. 2 does show that the accuracy does not significantly increase after 3-4 perceptrons are already used so 3 or 4 my be a more practical choice.

Finally, I complete the model training according to the selected number of perceptrons for each training set. Additionally, I calculated the probability of error for the theoretical optimal classifier using the minimum-probability-of-error classification rule with knowledge of the true PDF. The optimal classification is estimated to be 76% accurate according to this methodology. Though this provides the aspirational performance for the MLP classifier, it actually performed far worse than any of the models. Figure 3 provides the real accuracy of the models, trained by different sized training data sets, on predicting the true class labels of the validation data set. As expected, the larger the data set used to train a model, the better the model performed when classifying the validation data. For visualization on the distribution and classification of the validation data based on the model, see the figures provides in my Github.

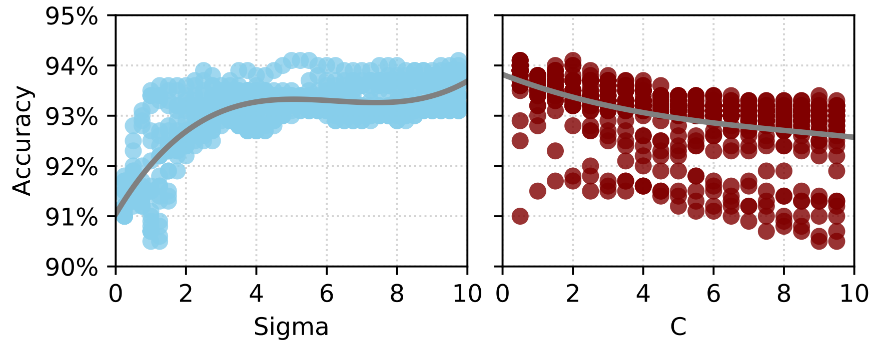

Figure 4: *Trends in cross-validation predicted accuracy of the model based on varying model hyperparameters.*

2 Support Vector Machine (SVM) Classifier with a Spherical Symmetric Gaussian Kernel

In this section, I develop an SVM classifier to classify samples in a two-class multi-ring data set. First, I generate my training and testing data for the model using the Python function provided in the course.

Next, I used minimum-average-cross-validation-probability-of-error to determine the best hyperparameters to use when for the SVM model. I tested a range of C and gamma values by finding the probability of error for each configuration, or combination of C and gamma parameters. Figure 1 shows the average score of the 10 folds for each configuration. We observe that as sigma increase, the probability of error decreases and hence the accuracy score increases. For the C parameter, we observe the opposite trend. Accuracy score decreases as the C parameter is increased. From the average scores of the configurations, I selected the minimum probability of error and used that configuration to train my final SVM model. The following were the hyperparameters chosen based on this methodology, which gave the minimum average probability of error of 0.059:

$$C = 2.0$$

$$\sigma = 9.75$$

Finally, I trained my final SVM model using the chosen hyperparameters. I then tested the model on 10000 samples in the test data set, which showed an accuracy of $\approx 93.1\%$. Figure 2 shows the correct and incorrectly classified labels in the test data set based on the model developed in this section.

3 Appendix

See my GitHub for all the source code: Click Me!
or copy and paste:

`https://github.com/emilyjcosta5/machine_learning/tree/main/homework_3`

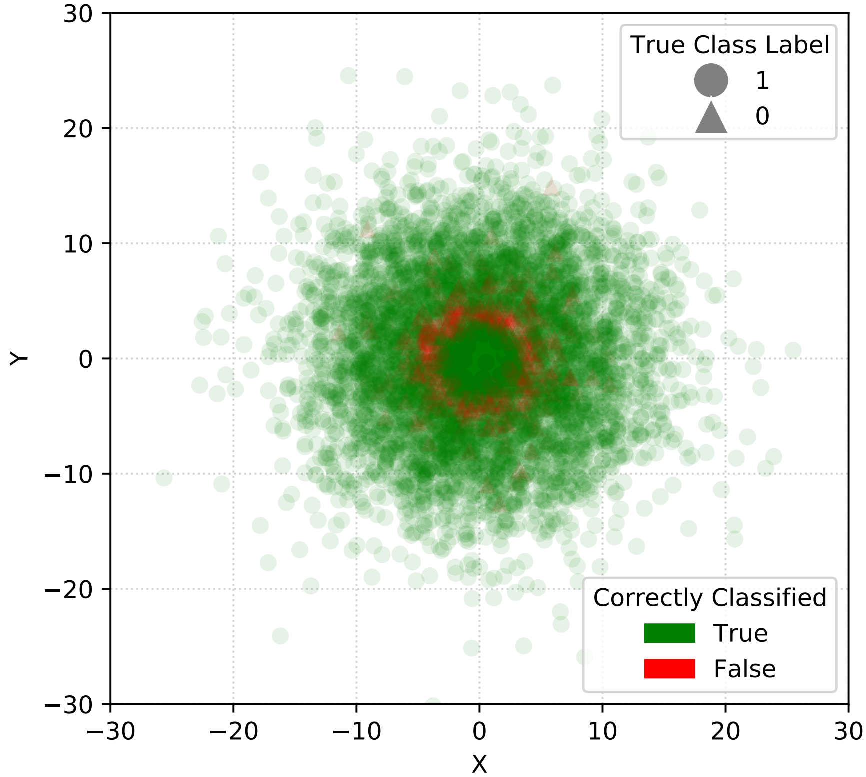

Figure 5: *Accuracy of the developed SVM in the two-class multi-ring data set.*