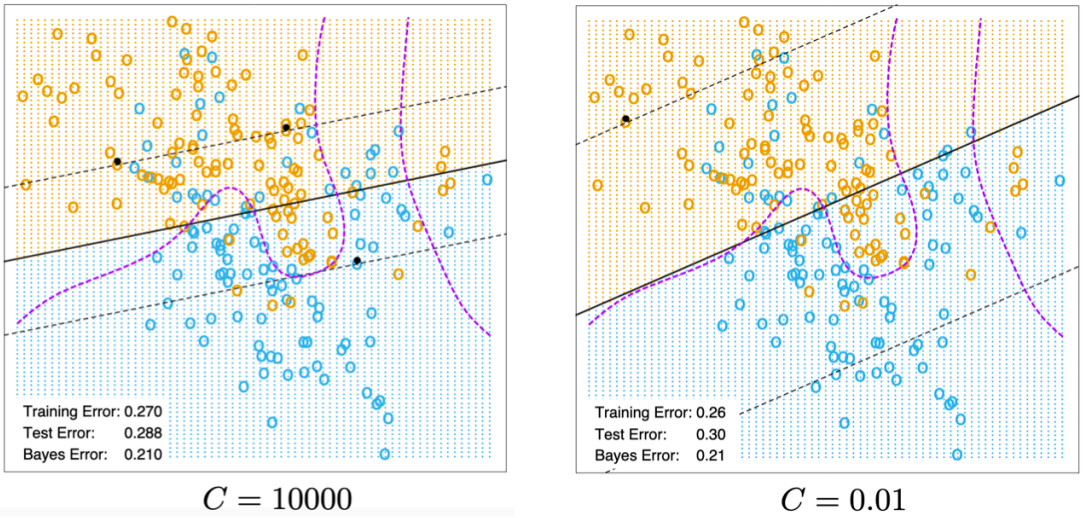
Support Vector Machine Classifier on Mixture Example

This project will reproduce the support vector machine classifier example figures on Hastie et al's book. The reproduced 2 figures(4 plots) are chosen from the book Elements of Statistical Learning by Hastie et al .



$C = 10000$ $C = 0.01$

> Figure source: Hastie, et al. Elements of Statistical Learning, Figure 12.2

The problem is to classify the labeled dots to two different regions based on their given location. The inputs are the given location; the outputs will be the specific label such as either 0 or 1 in this case. The desired function is trying to find one or more than one suitable linear function(s) that can be devided the data to $n + 1$ regions, where n denotes the number of the linear function.

Most of SVM are trying to solve the classification problem in the real world. For this project, we only have two input features; yet, the real-world problem could be high dimension inputs. Such as predicting if the patient has cancer based on the feature like the age, the eating habit, even height, and weight.

The dataset I plan to use is exactly same as the data that produces the $\text{Figures } 12.2, 12.3$ and elsewhere in the Hastie et al's book. The date is called ESL mixture simulation data. It is not a real-world data; it is generated by using Gaussian mixture model simulation, but I will directly use this in order to get the same figure. The data is stored in rda(r date object file) format, which can store the different dimension of data in an object at the same time. However, the really useful input only has 200 observations and 2 features. The two features indicate the x and y location in the figure. The Y label is a 200×1 vector which stores either 0 or 1 integer that refers to the class of each observation.

In short, the algorithm is trying to learn the paramater slope W and intercept β that can divide the most of data into the correct region. The baseline of prediction of this problem could be the prediction based on the most of label in the dataset and uses the label existed above the average as the prediction of all the observation. The crude pseudo-code goes below. The detail and math I wrote behind this algorthim is here.

```
Inputs: data X, target y, penalty C
    Random assign value for slope W and intercept \beta
    repeat
        for all {x_i, y_i}, fo
            Optimize W and \beta
        end for
    until no changes in slope W or other criteria met
```

The number showed in the figure can be computed by cross-validation method. In basic, I will divide all observations into multiple folds. In each iteration, I keep a fold as test set, and others to be the training data. This process will run based on each fold. Then, I can compute the average test error and training error. The Bayes error can be directly computed by the equation given in the description of the dataset.

In this project, I can learn something about reproduction of the Figure. In detail, I could learn the math behind the SVM algorithm since I use the package to get the function in most of the time. Coding from scratch is beneficial to understand SVM. More importantly, I can have the opportunity to use matplotlib and seaborn heavily to get the desired figure as same as the figure shown in chapter 12. It could be helpful to produce a better figure in my future research as well.