**Homework 1**

**Student ID**: 2020321249

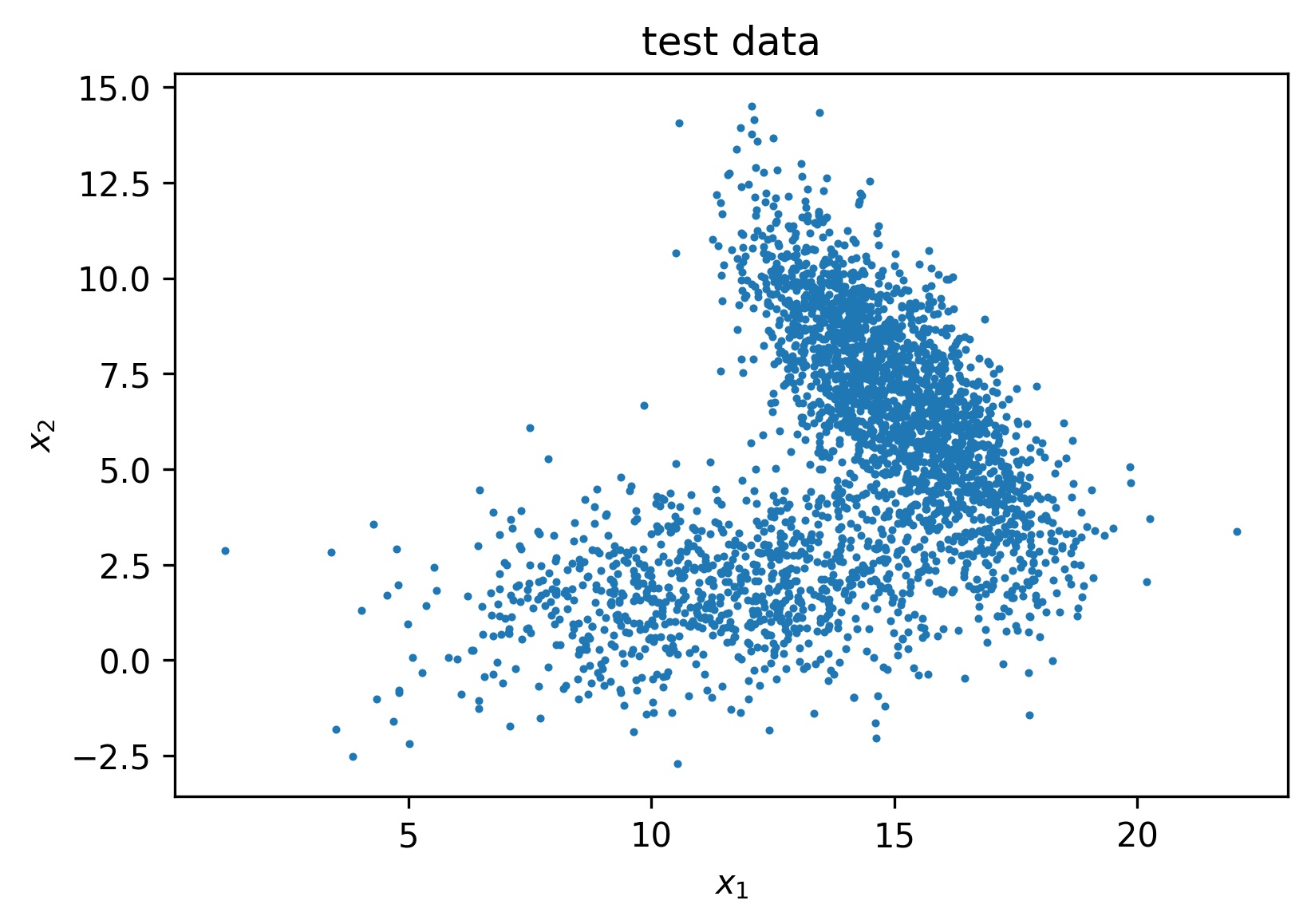
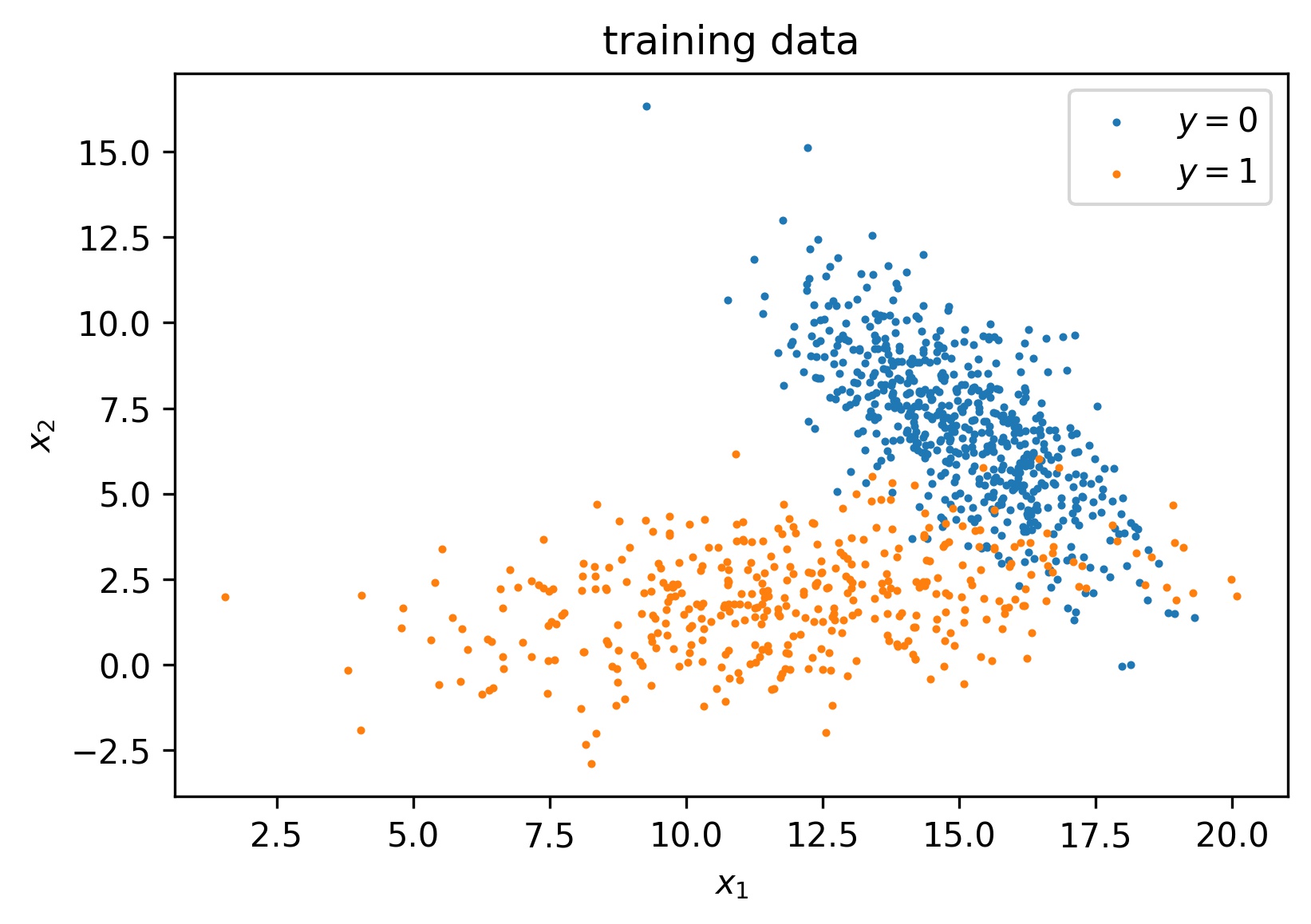
**Name**: 남혜린 Hyelin Nam

**(i)**

a) Training data, Test data

I can easily see data from each class are distributed discriminately.

Even though the class of test data are given, I assumed that I have no information of them, in order to purely inference the class of given test data with built model. However, I can easily predict the bottom samples and data distributed diagonally at right side belong to different classes.



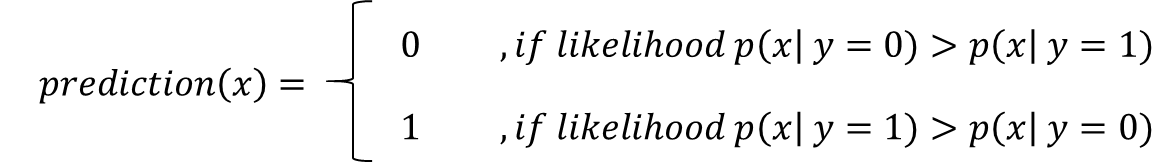
**(ii)**

First, I predict classes of each data sample, with different methods such as MLE or model output. Each will be described in detail at problems below.

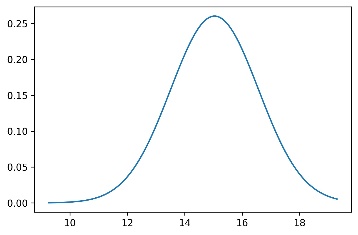
Then count the difference of predicted class and ground truth. This devided by number of total inferenced data means error rate. (1-error rate)\*100 is accuracy in percentage.

**(iii)**

a) Maximum Likelihood Estimation

In Naïve Bayes decision output, likelihood of a sample is in class 0 and that in class 1 is compared. The class with bigger likelihood value is the prediction result.

When calculating likelihood, each features of sample are independently distributed. Therefore, likelihood is multiplication of likelihood of each feature.

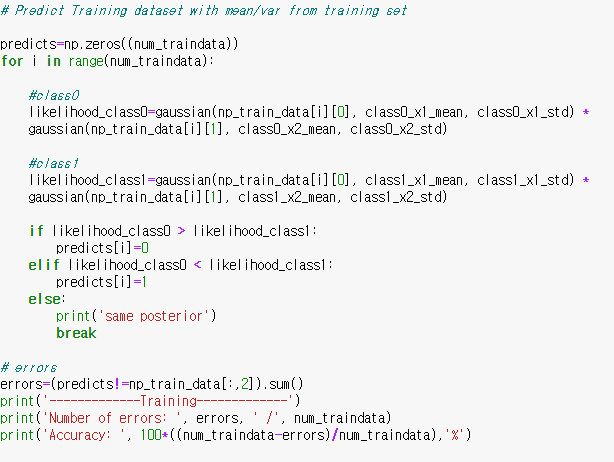


Here, each feature follows Gaussian distribution. This is plot of likelihood

, which follows

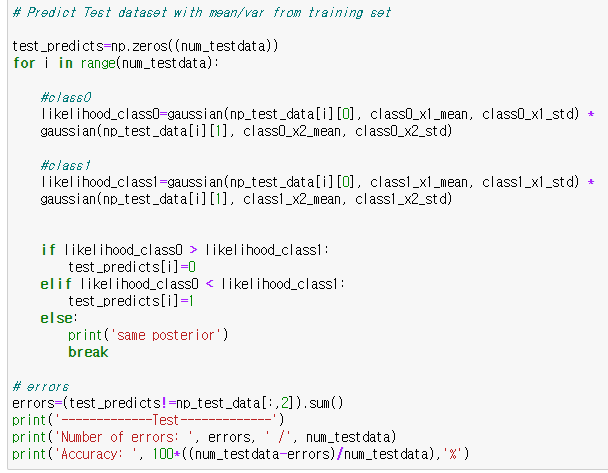
b) Predict Train dataset with mean/var from training set

Like above, I measured likelihood of each data by calculating gaussian pdf with mean and std of training set belong to each class. It resulted 93% of accuracy, with number of errors 70 out of 1,000 training set.



c) Predict Test dataset with mean/var from training set

Likewise, with gaussian distribution generated with training dataset, I measured likelihood of each test data. If the likelihood of the datum with condition of class 0 is bigger than that of class 1, I made a decision with the sample as class 0. It resulted 93.53% of accuracy, with number of errors 194 out of 1,000 training set.

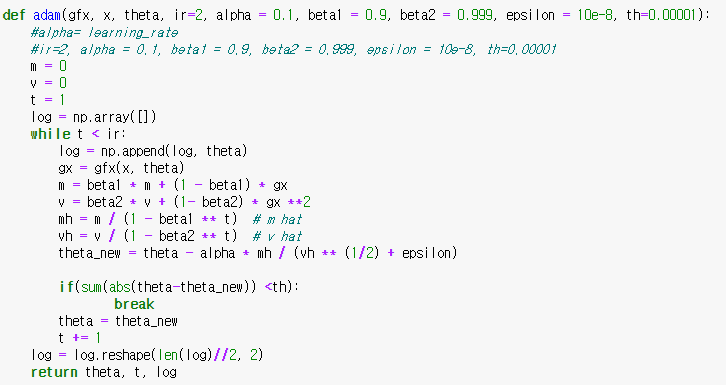


d) Training

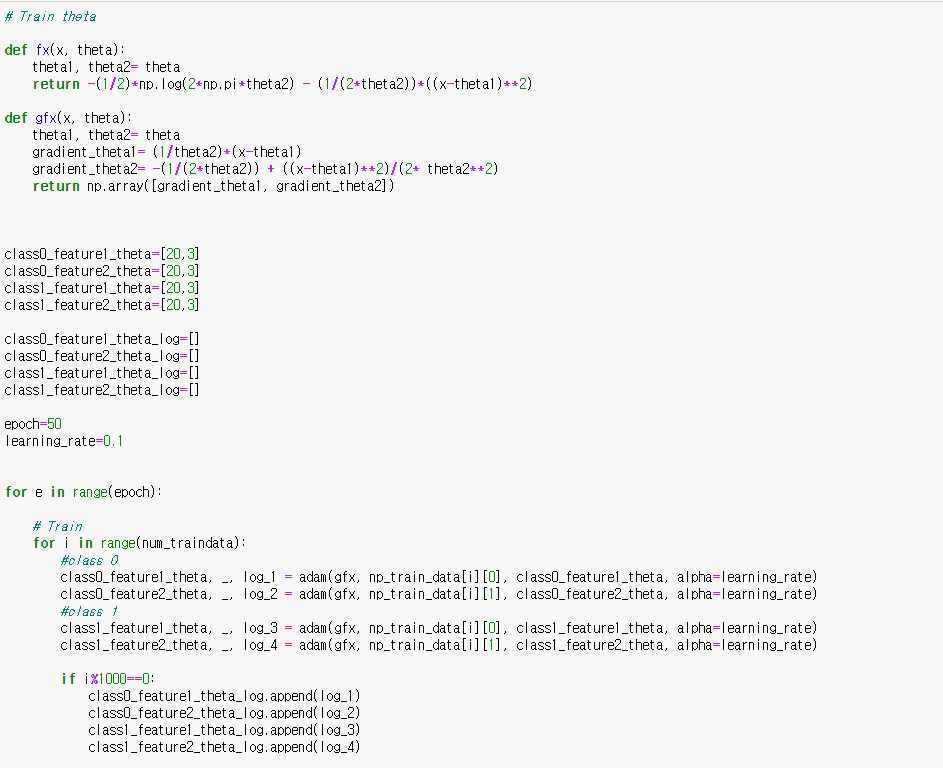
Also, we can define that comprises each mean and std of Gaussian distribution that can describe the likelihood probability of data.

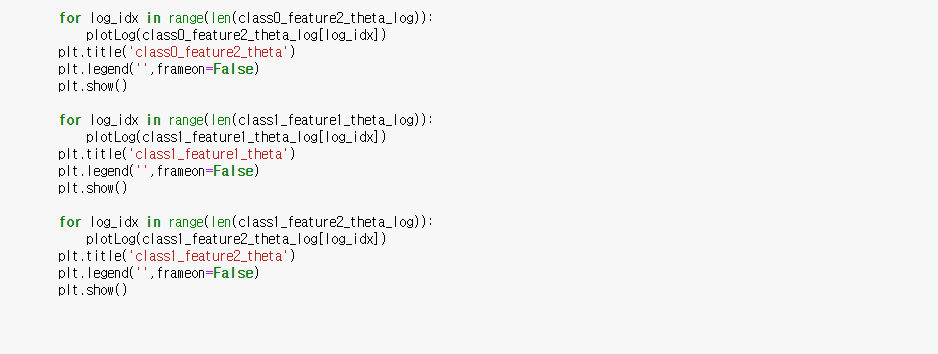
I calculated gradient of log-likelihood as below, and updated the with adam optimizer.

Adam optimizer with defined gradient (gfx). x means input, theta is the updating parameters, and alpha is learning rate



Training





The Training accuracy was 89.7%, with 103 errors.

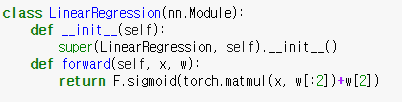
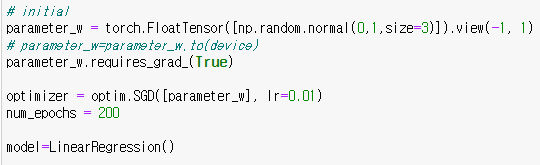
The Test accuracy was 80.93% ,with 572 errors.

**(iv, vii)**

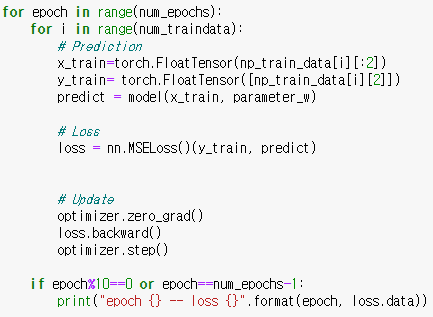
In linear regression, I built an network with 2-dimension weight and bias. For loss function, I use Mean Squre Error loss. With optimizing the weights with SGD optimizer, a linear decision boundary came out. It seems weakly performing, since the line do not discriminate data from different classes. For reasons, I can predict wrong initial value of weights, since I generated with random values, not proper activation function, sigmoid. Also, other loss functions such as cross entropy may work better.

When the model output is closer to 1 than 0, in other words, the output is larger than 0.5, the datum is predicted to class 0. The training accuracy is 61.19% and the test accuracy s 51.93%

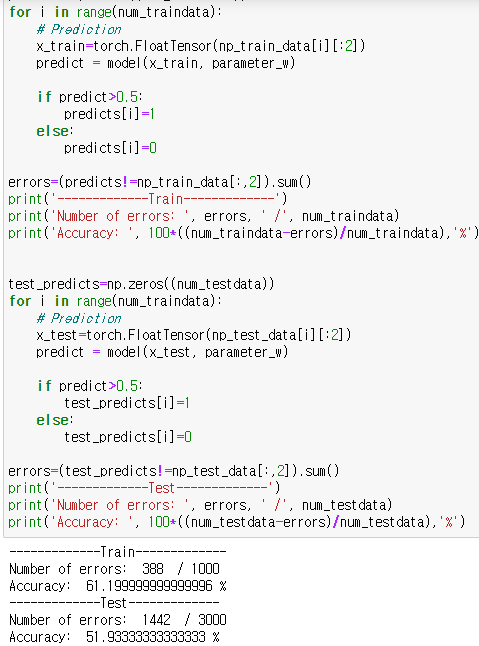
Model Parameters

Training



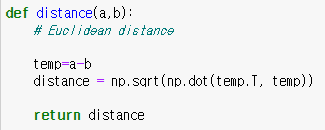
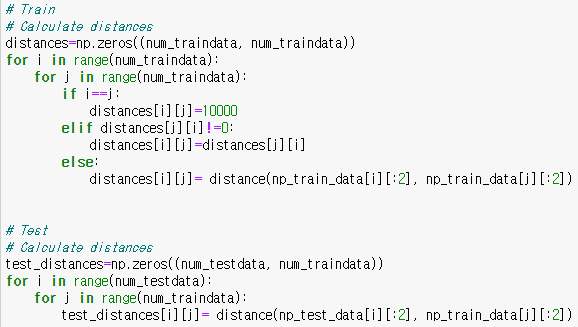
Results

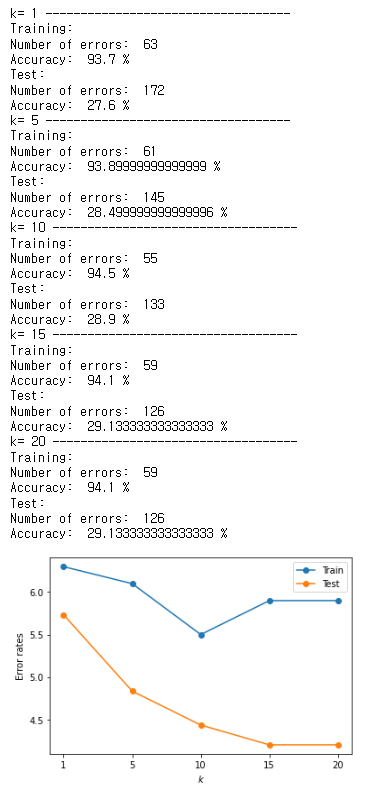
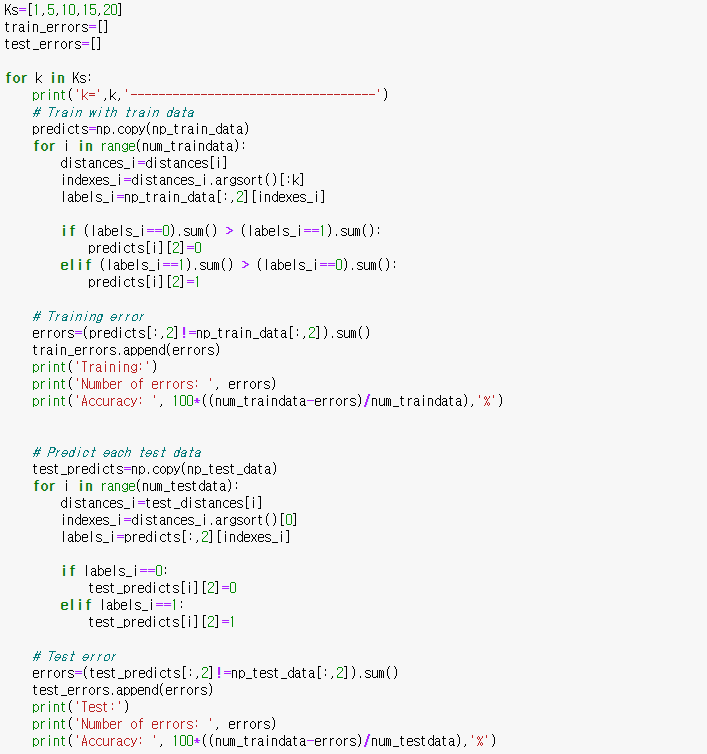
**(v, vi)**

In k-NN, when predicting for one sample, I draw a bound with k nearest samples in Euclidean distance samples. If the number of data belong to class 0 is more than that of class1 in the bound, the corresponding sample is predicted to class 0. After predicting with train data, I draw a Voronoi cell with regard to the prediction, then in inference with test dataset, each test sample follows the class of nearest dot, which is a data in the cell where the test sample is located.

Euclidean distance Calculating distance

Train and test with different ks.



**(viii, ix) is included in the above answers.**