Machine Learning from Data CSCI 4100
Assignment 10
Jae Park (RIN: 661994900)

Exercise 6.1

a) High cosine similarity but low Euclidean distance similarity:

$$\text{CosSim}(<1,1>,<100,100>) = \frac{200}{\sqrt{2}\sqrt{20000}} = 1$$
$$d(<1,1>,<100,100>) = \sqrt{(-99)^2 + (-99)^2} = 140.007\ldots$$

Low cosine similarity but high Euclidean distance similarity:

$$\text{CosSim}(<0.1,0.1>,<-0.1,-0.1>) = \frac{-0.02}{\sqrt{0.02}\sqrt{0.02}} = -1$$
$$d(<0.1,0.1>,<-0.1,-0.1>) = \sqrt{(0.02)^2 + (0.02)^2} = 0.0282\ldots$$

b) Euclidean distance similarity is independent of the origin as the relative distance between two vectors doesn't change; only the cosine similarity changes due to a shift in the actual coordinates. Therefore, choosing Euclidean distance similarity will be a better choice of feature.

Exercise 6.2

When $\pi(x) \geq \frac{1}{2}$, $f(x) = +1$. Then,

$$\begin{aligned} e(f(x)) &= \mathbb{P}[f(x) \neq y] \\ &= \mathbb{P}[f(x) = +1, y = -1|x] \\ &= 1 \times (1 - \pi(x)) \\ &= \min\{\pi(x), 1 - \pi(x)\} \text{ since } \pi(x) \geq 1 - \pi(x) \end{aligned}$$

When $\pi(x) < \frac{1}{2}$, $f(x) = -1$. Then,

$$\begin{aligned} e(f(x)) &= \mathbb{P}[f(x) \neq y] \\ &= \mathbb{P}[f(x) = -1, y = +1|x] \\ &= 1 \times (\pi(x)) \\ &= \min\{\pi(x), 1 - \pi(x)\} \text{ since } \pi(x) < 1 - \pi(x) \end{aligned}$$

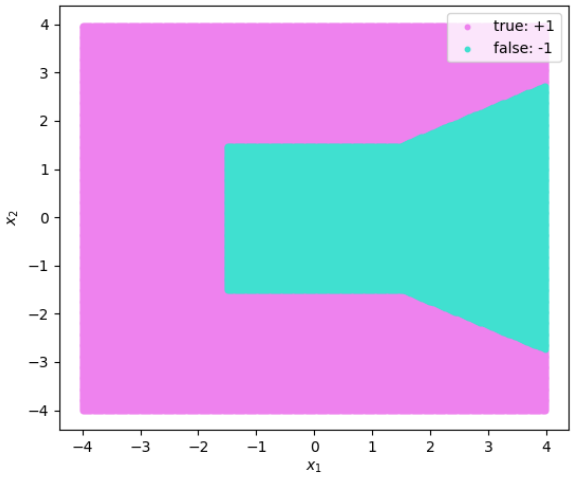
$$\therefore e(f(x)) = \min\{\pi(x), 1 - \pi(x)\}$$

The error $e(f(x))$ is therefore equivalent to the minimum probability of error possible on test point x. This definition infers that any other hypothesis h that differs from f will produce

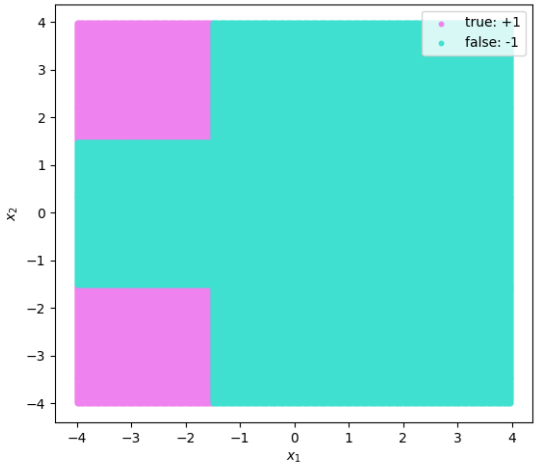
errors such that its probability of error is at least the probability of error given from f, or greater and with more noise.

Problem 6.1

a)

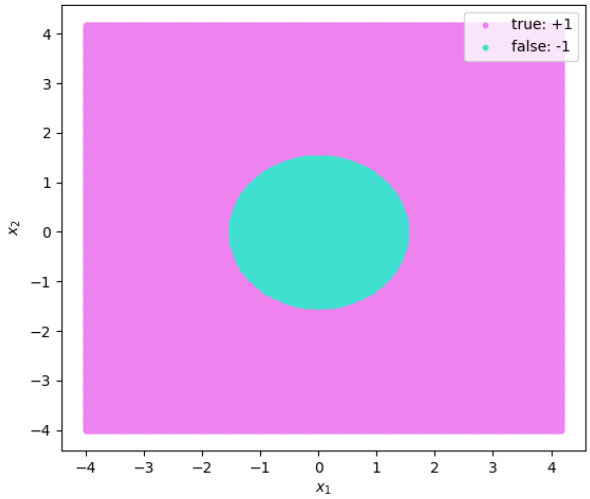


(a) 1-NN Classification without Transformation

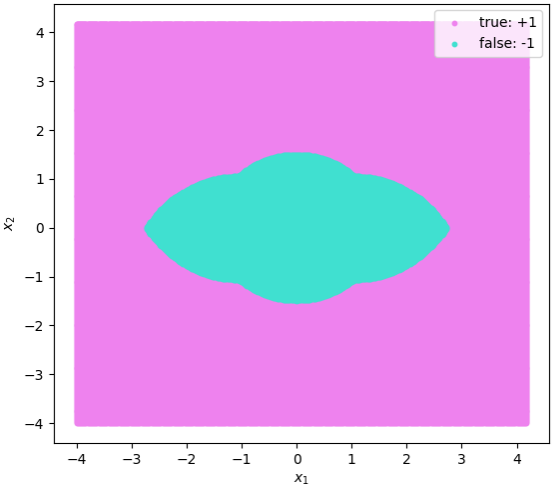


(b) 3-NN Classification without Transformation

b)



(a) 1-NN Classification with Transformation



(b) 3-NN Classification with Transformation

Problem 6.4

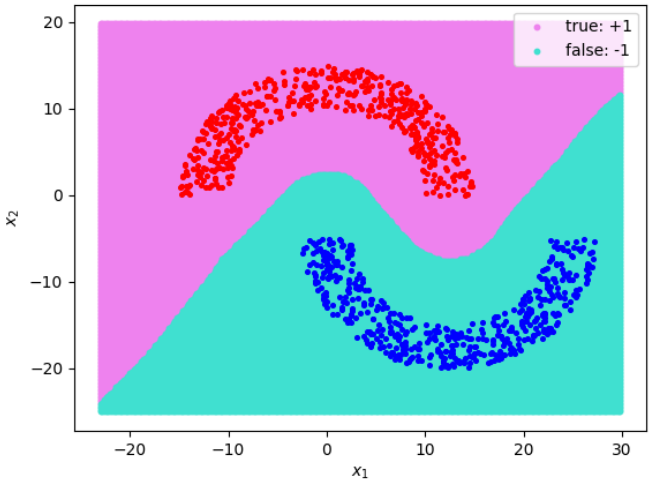


Figure 3: 1-NN Classification

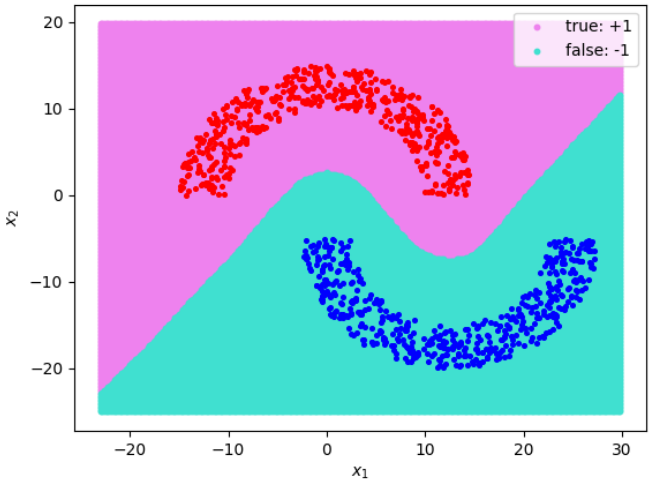


Figure 4: 3-NN Classification

There's essentially no difference in classification between 1-NN and 3-NN with larger amount of data (n = 2000).

Problem 6.16

a) On 10,000 query points,
-First Trial-
Running time for obtaining the nearest neighbor with branch and bound method: 12.098s
Running time for obtaining the nearest neighbor with the brute force approach: 108.911s

-Second Trial-
Running time for obtaining the nearest neighbor with branch and bound method: 11.236s
Running time for obtaining the nearest neighbor with the brute force approach: 110.902s

b) On 10 Gaussian clusters,
-First Trial-
Running time for obtaining the nearest neighbor with branch and bound method: 19.683s
Running time for obtaining the nearest neighbor with the brute force approach: 106.523s

-Second Trial-
Running time for obtaining the nearest neighbor with branch and bound method: 18.284s
Running time for obtaining the nearest neighbor with the brute force approach: 110.793s

c)
Branch and bound method performs much better than the brute force approach. Between 10,000 query points and 10 Gaussian distributions, there was no significant difference in using the brute force approach. However, branch and bound method performs better when the data points are uniformly distributed than clustered into several different groups. This can probably be attributed to the clustering of data points; if the point of interest is located around the center of a Gaussian distribution, there are more points to compare distances than having points randomly and uniformly distributed around it.

d)
Maybe, but probably not. It could be the case that brute force performs better than the branch and bound method with very little data points as there are only a few points to compare overall. Partitioning might have more computational load in such a case. However, in general, branch and bound method should do better than the brute force way.