ECE 662: Homework 1

by

Zhankun Luo

Elmore Family School of Electrical and Computer Engineering

West Lafayette, Indiana

Spring 2022

Question

N samples $(x_i, \omega_i) \in S \times \Omega, i \in \{1, \cdots, N\}$ for r.v. (X, W), and $S = \mathbb{R}^2, \Omega = \{1, 2\}$

we can reorder (x_i, ω_i) to ensure $\omega_1 = \cdots = \omega_{n_1} = 1, \omega_{n_1+1} = \cdots = \omega_N = 2$

denote the class 1, 2 means by $\mu_1 \equiv \frac{\sum_{i=1}^{n_1} x_i}{n_1}, \mu_2 \equiv \frac{\sum_{i=n_1+1}^{N} x_i}{N-n_1}$

define the inner product as $\cdot : S \times S \to \mathbb{R}$

and S is equipped with a metric $d : S \times S \to \mathbb{R}_{\geq 0}$ which satisfies

- $d(x, x') \geq 0, d(x, x') = 0 \Leftrightarrow x = x'$
- $d(x, x') = d(x', x)$
- $d(x, x') + d(x', x'') \geq d(x, x'')$

Consider the classification rule $\hat{\omega} \equiv f(x)$, and suppose $\mu_1 \neq \mu_2$, thus $d(\mu_1, \mu_2) > 0$

$$f(x) = \omega_{i^*}, \quad \text{where } i^* \equiv \underset{i \in \Omega = \{1,2\}}{\operatorname{argmin}} d(x, \mu_i)$$

---

assumption: suppose the metric for S to be $d(x, x') = \sqrt{(x - x') \cdot (x - x')}$

notice $d(x, \mu_1) + d(x, \mu_2) \geq d(\mu_1, \mu_2) > 0$

the separation is $g(x) = 0 \Leftrightarrow d(x, \mu_2) - d(x, \mu_1) = 0$, where the discriminator $g(x)$ is

$$\begin{aligned} g(x) &= [d(x, \mu_2) - d(x, \mu_1)] \cdot \frac{d(x, \mu_1) + d(x, \mu_2)}{2} \\ &= \frac{1}{2}[d^2(x, \mu_2) - d^2(x, \mu_1)] \\ &= (\mu_1 - \mu_2) \cdot \left(x - \left[\frac{\mu_1 + \mu_2}{2} \right] \right) \end{aligned}$$

$$g(x) \begin{cases} > 0 & \Rightarrow f(x) = 1, \text{decide class 1} \\ < 0 & \Rightarrow f(x) = 2, \text{decide class 2} \end{cases}$$

Euclidean inner product

$$x \cdot x' = x^\top x'$$

The separation $g(x) = 0$ becomes a straight line

$$g(x) = (\mu_1 - \mu_2)^\top \left(x - \left[\frac{\mu_1 + \mu_2}{2}\right]\right) = 0$$

Riemannian inner product

$$x \cdot x' = x^\top M x'$$

where $M = M^\top$ and the matrix M is positive definite

The separation $g(x) = 0$ also becomes a straight line

$$g(x) = [M(\mu_1 - \mu_2)]^\top \left(x - \left[\frac{\mu_1 + \mu_2}{2}\right]\right) = 0$$

Can we define a metric $d(x, x')$ on $S = \mathbb{R}^2$ such that the above classifier $f(x)$ will yield a separation $g(x) = 0 \Leftrightarrow d(x, \mu_2) - d(x, \mu_1) = 0$ that is **NOT** a straight line?

Solution

Let $S = \mathbb{R}^2$ to be equipped with the **Minkowski distance** of order $p \in [1, +\infty)$ instead

$$d(x, x') = \left\|x - x'\right\|_p \equiv \left(\sum_{k=1}^{d} |x_{(k)} - x'_{(k)}|^p\right)^{\frac{1}{p}}$$

where $d = 2$ is the dimension of $S = \mathbb{R}^2$ and column vectors
$x = (x_{(1)}, x_{(2)}, \cdots, x_{(d)})^\top, x' = (x'_{(1)}, x'_{(2)}, \cdots, x'_{(d)})^\top$

Since the **Minkowski inequality** holds for $p \in [1, +\infty)$, it satisfies

- $\left\|x - x'\right\|_p \geq 0, \left\|x - x'\right\|_p = 0 \Leftrightarrow x = x'$
- $\left\|x - x'\right\|_p = \left\|x' - x\right\|_p$
- $\left\|x - x'\right\|_p + \left\|x' - x''\right\|_p \geq \left\|x - x''\right\|_p$

Thus, the separation $g(x) = 0 \Leftrightarrow d(x, \mu_2) - d(x, \mu_1) = 0$ becomes

$$\left\|x - \mu_2\right\|_p - \left\|x - \mu_1\right\|_p = 0$$

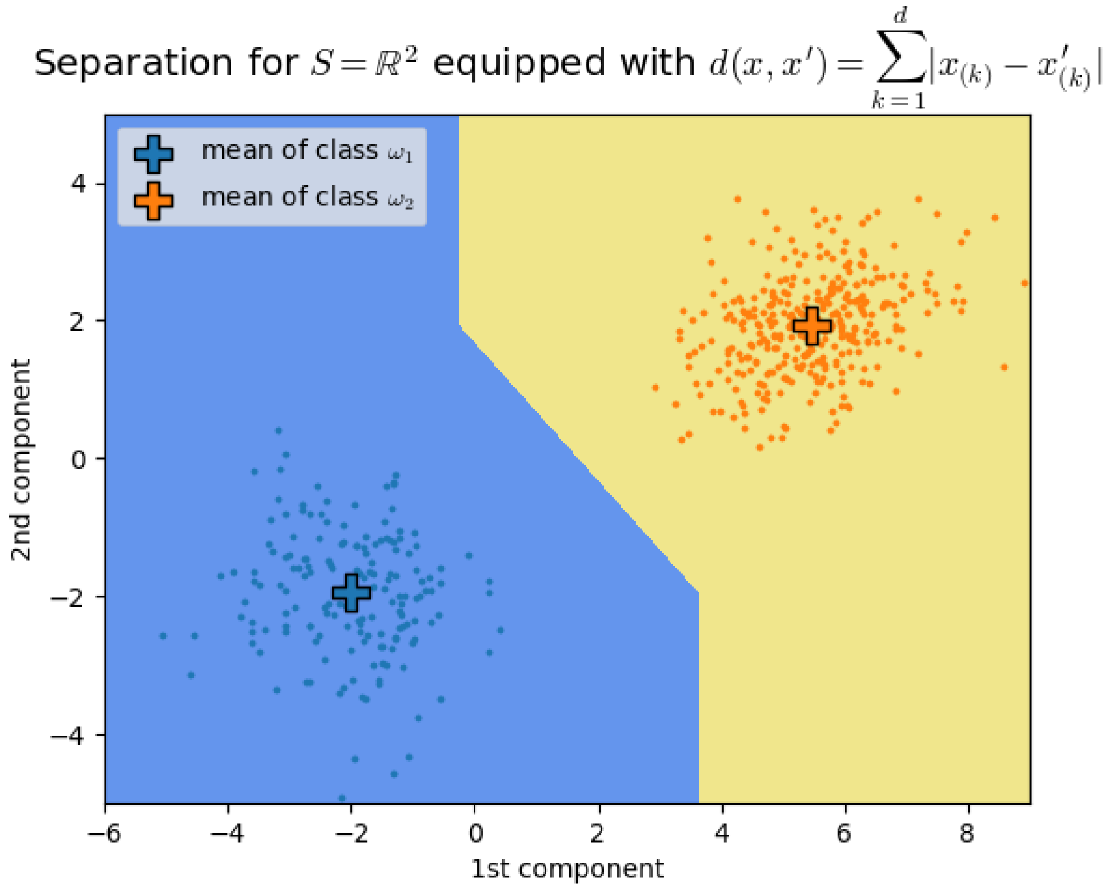
$p = 1$: Manhattan distance

For $p = 1$, the **Minkowski distance** becomes the **Manhattan distance**

$$d(x, x') = \|x - x'\|_1 \equiv \sum_{k=1}^{d} |x_{(k)} - x'_{(k)}|$$

The separation becomes below, where the dimension $d = 2$ for $S = \mathbb{R}^2$

$$\sum_{k=1}^{d} |x_{(k)} - \mu_{2(k)}| - \sum_{k=1}^{d} |x_{(k)} - \mu_{1(k)}| = \sum_{k=1}^{d} |x_{(k)} - \mu_{2(k)}| - |x_{(k)} - \mu_{1(k)}| = 0$$



Generated Data and Saparation Based on the Manhattan Distance

For the above example when $S = \mathbb{R}^2$ equipped with $d(x, x') = \sum_{k=1}^{d} |x_{(k)} - x'_{(k)}|$, the classifier $f(x)$ yields a separation $g(x) = 0 \Leftrightarrow d(x, \mu_2) - d(x, \mu_1) = 0$ that is **NOT** a straight line

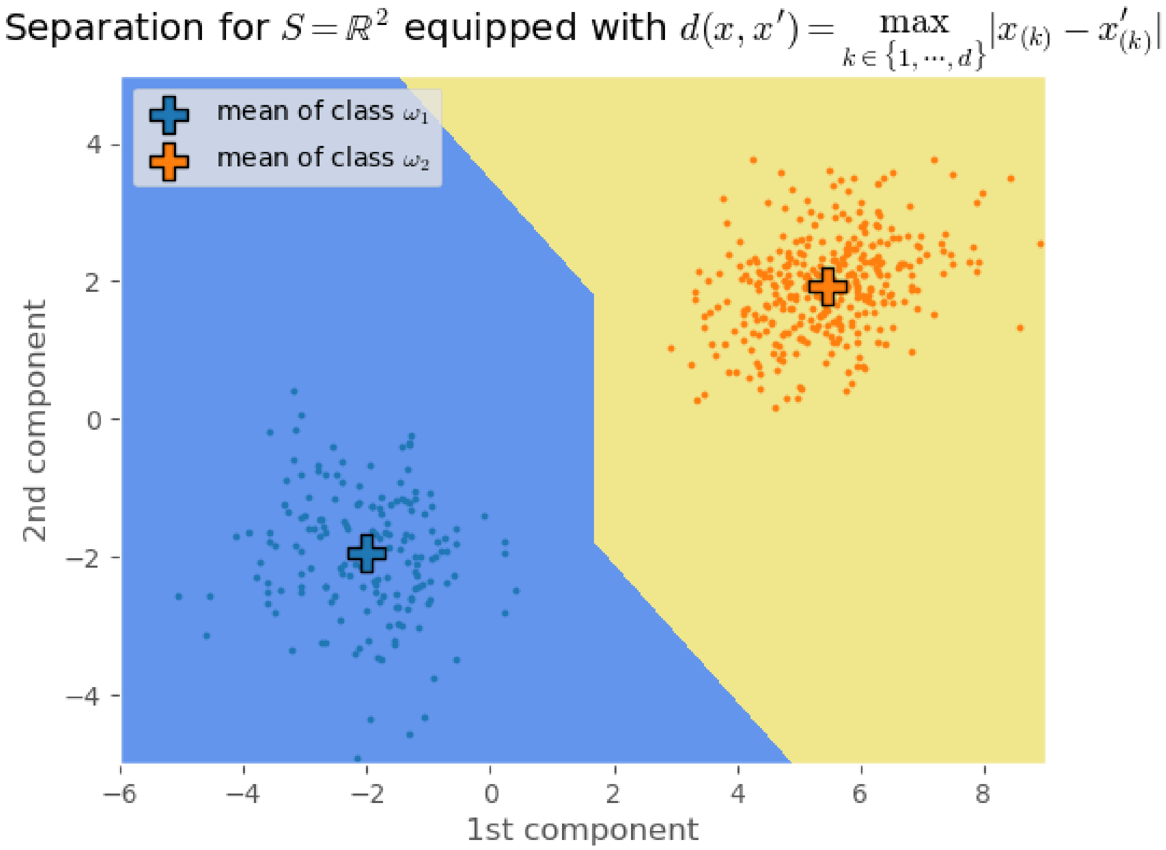
$p = +\infty$: Chebyshev distance

For $p = +\infty$, the **Minkowski distance** becomes the **Chebyshev distance**

$$d(x, x') = \lim_{p \to +\infty} \left(\sum_{k=1}^{d} |x_{(k)} - x'_{(k)}|^p \right)^{\frac{1}{p}} \equiv \max_{k \in \{1, \cdots, d\}} |x_{(k)} - x'_{(k)}|$$

The separation becomes below, where the dimension $d = 2$ for $S = \mathbb{R}^2$

$$\max_{k \in \{1, \cdots, d\}} |x_{(k)} - \mu_{2(k)}| - \max_{k \in \{1, \cdots, d\}} |x_{(k)} - \mu_{1(k)}| = 0$$



Generated Data and Saparation Based on the Chebyshev Distance

For the above example when $S = \mathbb{R}^2$ equipped with $d(x, x') = \max_{k \in \{1, \cdots, d\}} |x_{(k)} - x'_{(k)}|$, the classifier $f(x)$ yields a separation $g(x) = 0 \Leftrightarrow d(x, \mu_2) - d(x, \mu_1) = 0$ that is **NOT** a straight line

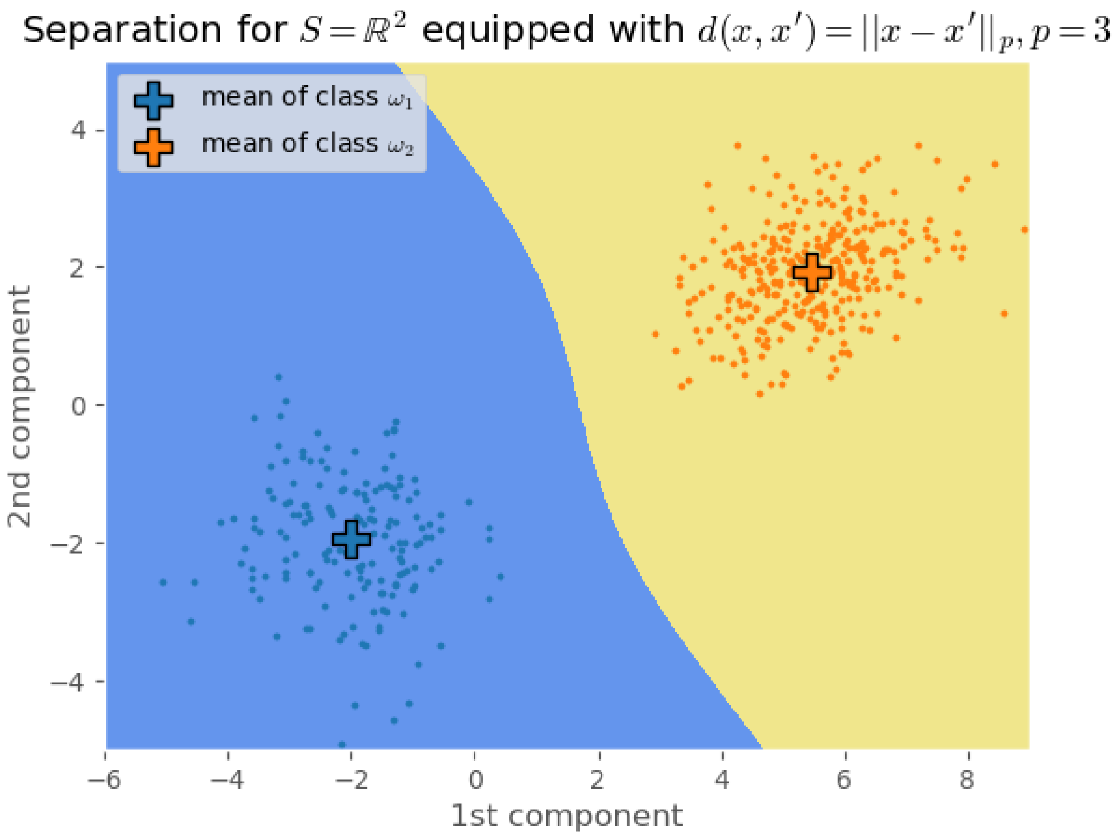
$p = 3$: Minkowski distance

For $p = 3$, the **Minkowski distance** is below, notice $t^3 : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is an One-to-One and Onto function

$$d(x, x') = \left(\sum_{k=1}^{d} |x_{(k)} - x'_{(k)}|^3 \right)^{\frac{1}{3}}$$

The separation $d^3(x, \mu_2) - d^3(x, \mu_1) = 0$ is below, where the dimension $d = 2$ for $S = \mathbb{R}^2$

$$\sum_{k=1}^{d} |x_{(k)} - \mu_{2(k)}|^3 - |x_{(k)} - \mu_{1(k)}|^3 = 0$$



Generated Data and Saparation Based on the Minkowski Distance when p=3

For the above example when $S = \mathbb{R}^2$ equipped with $d(x, x') = \|x - x'\|_p, p = 3$, the classifier $f(x)$ yields a separation $g(x) = 0 \Leftrightarrow d(x, \mu_2) - d(x, \mu_1) = 0$ that is **NOT** a straight line

Chordal distance on the Riemann Sphere

Consider $\psi : \mathbb{R}^2 \to S^3$, the unit sphere $S^3 \equiv \{Z = (\xi, \eta, \zeta)^\top \mid Z \in \mathbb{R}^3, \|Z\|_2 = 1\}$

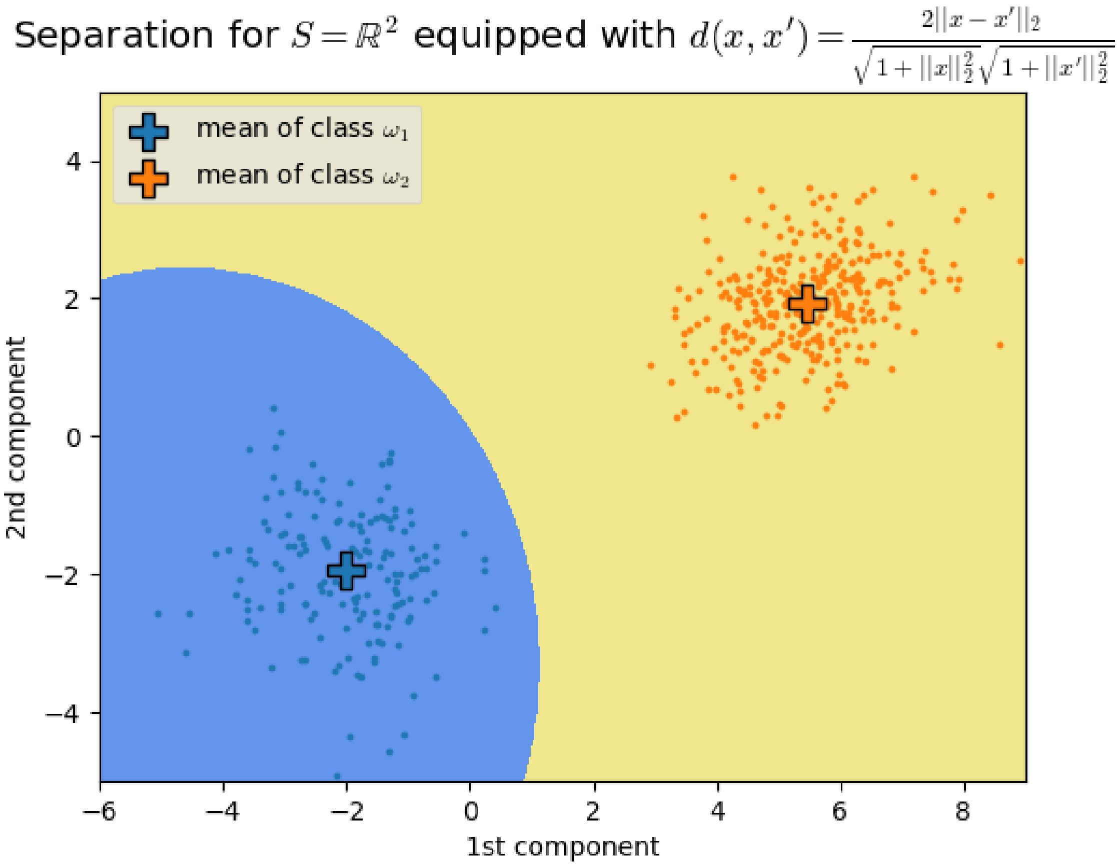
$$\psi : x = (x_{(1)}, x_{(2)})^\top \to (\xi, \eta, \zeta)^\top = \left(\frac{2x_{(1)}}{1 + x_{(1)}^2 + x_{(2)}^2}, \frac{2x_{(2)}}{1 + x_{(1)}^2 + x_{(2)}^2}, \frac{-1 + x_{(1)}^2 + x_{(2)}^2}{1 + x_{(1)}^2 + x_{(2)}^2} \right)$$

Let $Z = \psi(x)$, $Z' = \psi(x')$, then Choldal distance on the Riemann Sphere S^3 is

$$d(x, x') = \|Z - Z'\|_2 = \frac{2\|x - x'\|_2}{\sqrt{1 + \|x\|_2^2}\sqrt{1 + \|x'\|_2^2}}$$

The traingle inequality holds since $||, ||_2$ satisfies it on $S^3 \subset \mathbb{R}^3$. The separation $\frac{(1+\|x\|_2^2)}{4}[d^2(x, \mu_2) - d^2(x, \mu_1)] = 0$ is below, the boundary is **NOT** a straight line

$$\frac{\|x - \mu_2\|_2^2}{\left(1 + \|\mu_2\|_2^2\right)} - \frac{\|x - \mu_1\|_2^2}{\left(1 + \|\mu_1\|_2^2\right)} = 0$$



Generated Data and Saparation Based on the Chordal Distance

Canberra distance

The Canberra distance is defined as below, where the dimension $d = 2$ for $S = \mathbb{R}^2$

$$d(x, x') = \sum_{k=1}^{d} \frac{|x_{(k)} - x'_{(k)}|}{|x_{(k)}| + |x'_{(k)}|}$$

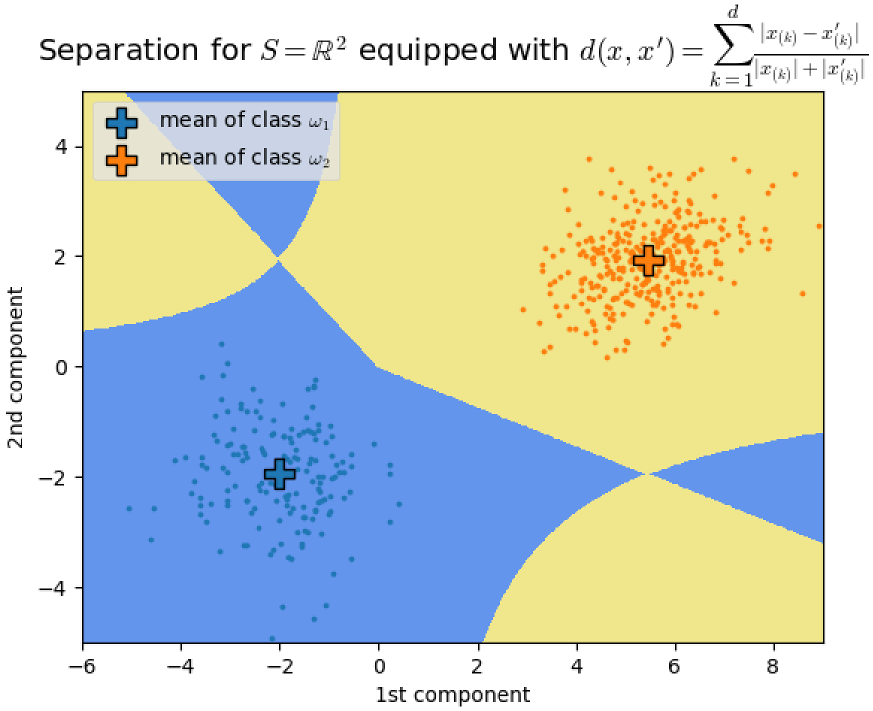
The triangle inequlity $d(x, x') + d(x', x'') \geq d(x, x'')$ can by proved by summing up

$$\frac{|x_{(k)} - x'_{(k)}|}{|x_{(k)}| + |x'_{(k)}|} + \frac{|x''_{(k)} - x'_{(k)}|}{|x''_{(k)}| + |x'_{(k)}|} \geq \frac{|x_{(k)} - x''_{(k)}|}{|x_{(k)}| + |x''_{(k)}|}$$

Above is verified if any $x_{(k)}, x'_{(k)}, x''_{(k)}$ are 0 or have different signs, since left-hand≥ 1, right-hand≤ 1. Only need to prove the case $x_{(k)}, x'_{(k)}, x''_{(k)} > 0$, assume $0 < x_{(k)} \leq x''_{(k)}$

- (a) $x'_{(k)} \in [x_{(k)}, x''_{(k)}]$, it is equivalent to $0 \geq (x'_{(k)} - x_{(k)})(x'_{(k)} - x''_{(k)})$
- (b) $x'_{(k)} \in (0, x_{(k)})$, $\frac{x''_{(k)} - x'_{(k)}}{x''_{(k)} + x'_{(k)}} > \frac{x''_{(k)} - x_{(k)}}{x''_{(k)} + x'_{(k)}} > \frac{x''_{(k)} - x_{(k)}}{x_{(k)} + x''_{(k)}}$ are sufficient to prove it
- (c) $x'_{(k)} \in (x''_{(k)}, \infty)$, $g(t) \equiv -2(\frac{x_{(k)}}{t + x_{(k)}} + \frac{x''_{(k)}}{t + x''_{(k)}} - 1)$ and it is $g(x'_{(k)}) \geq g(x''_{(k)})$

Boundary $d(x, \mu_2) - d(x, \mu_1) = \sum_{k=1}^{d} \left(\frac{|x_{(k)} - \mu_{2(k)}|}{|x_{(k)}| + |\mu_{2(k)}|} - \frac{|x_{(k)} - \mu_{1(k)}|}{|x_{(k)}| + |\mu_{1(k)}|} \right) = 0,$ is **NOT** straight



Generated Data and Saparation Based on the Canberra Distance

Appendix

Python code to generate data and labels (x_i, ω_i): `mk_data.py`

```
import numpy as np
from numpy import sqrt, diag, transpose
from numpy.linalg import eig
from numpy.random import seed, randn, random_sample
from os.path import join, abspath, dirname
import matplotlib.pyplot as plt

def generate_data(R, mu, N, se=0):
    """Generate N data samples for a Guassian distribution cluster"""
    # generate N points for each cluster
    # V[:,i] is the eigenvector corresponding to the eigenvalue D[i]
    # if X~N(0, D), and Y=V X <=> V^T Y=X then Y~N(0,V D V^T)
    # W~N(0, I), sqrt(D) W~N(0,D), V sqrt(D) W~N(0, V D V^T)~N(0, R)
    # with decomposition: R = V D V^T
    seed(se)
    D, V = eig(R)
    return transpose(V @ diag(sqrt(D)) @ randn(2, N) + mu)

def choose_data(X, pi, se=0):
    """Choose samples from different Gaussian dist. with priors"""
    N = len(X[0])
    seed(se); switch = random_sample(N)
    def get_category(x):
        sum = 0
        for c, p in enumerate(pi):
            if sum <= x and x < sum + p:
                return c+1 # lowest index 0 mapping to class 1
            sum += p
    label = list(map(get_category, switch))
    return np.asarray([X[c-1][i]
                       for i, c in enumerate(label)]), label

def plot_data(data):
    """Plot the data generated by the Gaussian Mixture Model"""
    x1, x2 = data[:, 0], data[:, 1]
    plt.style.use('ggplot')
```

```
    plt.scatter(x1, x2, s=5, marker='o', c='#1f77b4')
    plt.title('Scatter Plot of Multimodal Data')
    plt.xlim([-6, 9]); plt.ylim([-5, 5])
    plt.xlabel('first component')
    plt.ylabel('second component')
    path_save = join(dirname(abspath(__file__)), 'data.png')
    plt.savefig(path_save,
                bbox_inches='tight',
                pad_inches=0)
    plt.show()

def save_data(data, label):
    """Save both data and labels of the Gaussian Mixture Model"""
    path_data  = join(dirname(abspath(__file__)), 'data.txt')
    path_label = join(dirname(abspath(__file__)), 'label.txt')
    np.savetxt(path_data, data,
               fmt='%16.7e', delimiter='', newline='\n')
    np.savetxt(path_label, label,
               fmt='%d', delimiter=' ', newline='\n')

if __name__ == "__main__":
    N = 500 # total number of generated points
    R1 = np.asarray([[1,-0.1],[-0.1,1]])
    mu1 = np.asarray([[-2], [-2]])
    R2 = np.asarray([[1,0.2],[0.2,0.5]])
    mu2 = np.asarray([[5.5], [2]])
    X_all = [generate_data(R, mu, N, se) \
            for R, mu, se in \
            list(zip([R1, R2], [mu1, mu2], [1, 2]))]
    pi = [0.3, 0.7]
    data, label = choose_data(X_all, pi, se=19)
    plot_data(data)
    save_data(data, label)
```

Python code to draw the separation for classifiers $\hat{\omega} \equiv f(x)$:

`solution.py`

```
from matplotlib.transforms import Bbox
import numpy as np
from numpy import arange, meshgrid, ndarray, reshape, unique
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
matplotlib.rcParams['mathtext.fontset'] = 'cm'
from typing import Callable
from math import sqrt, pow

def plot_data_label(data: ndarray, label: ndarray) -> None:
    """Plot both data and labels"""
    list_color = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', \
                  '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', \
                  '#bcbd22', '#17becf']
    category = unique(label)
    plt.style.use('ggplot')
    for c in category:
        color = list_color[(c-1)%len(list_color)]
        x = data[label == c]
        plt.scatter(x[:, 0], x[:, 1], s=5, marker='o',
                    c=color) #, edgecolors='black')
    plt.xlim([-6, 9]); plt.ylim([-5, 5])
    plt.xlabel('1st component')
    plt.ylabel('2nd component')

def plot_cluster_center(k: int, mean: ndarray) -> None:
    """Plot the center of a cluster"""
    mean = reshape(mean, (2,))
    list_color = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', \
                  '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', \
                  '#bcbd22', '#17becf']
    color = list_color[(k-1)%len(list_color)]
    plt.scatter(mean[0], mean[1], s=200, marker='P', alpha=1,
        label=r'mean of class $\omega_{}$'.format(k), color=color,
        linewidths = 1,
        edgecolor ="black",)
    plt.legend()
```

```python
def plot_region(f: Callable[[ndarray], ndarray]):
    delta = 0.02
    range_x = arange(-10, 10, delta); range_y = range_x
    X, Y = meshgrid(range_x, range_y)
    x = np.c_[X.ravel(), Y.ravel()] # stack 1D as columns into 2D
    Z = f(x)
    Z = Z.reshape(X.shape)
    plt.contourf(X, Y, Z,
                 cmap=ListedColormap(["cornflowerblue", "khaki"]))

def classifier(x: ndarray, mu: ndarray,
               metric: Callable[[list, list], float]) -> ndarray:
    category_predicted = []
    for x_sample in x:
        k = min(enumerate(mu),
                key=lambda t: metric(x_sample, t[1]))[0]
        category_predicted.append(k+1)
    return np.array(category_predicted)

def metric_Manhattan(x1: list, x2: list) -> float:
    return sum([abs(e1-e2) for e1, e2 in list(zip(x1, x2))])

def metric_Chebyshev(x1: list, x2: list) -> float:
    return max([abs(e1-e2) for e1, e2 in list(zip(x1, x2))])

def metric_Euclidean(x1: list, x2: list) -> float:
    return sqrt(sum([(e1-e2)*(e1-e2)
                     for e1, e2 in list(zip(x1, x2))]))

def metric_Minkowski(x1: list, x2: list, p: float) -> float:
    return pow(sum([pow(abs(e1-e2), p)
                    for e1, e2 in list(zip(x1, x2))]) , 1./p)

def metric_Chordal(x1: list, x2: list) -> float:
    num = sum([(e1-e2)*(e1-e2) for e1, e2 in list(zip(x1, x2))])
    denom1 = sum([e*e for e in x1]) + 1
    denom2 = sum([e*e for e in x2]) + 1
    return 2 * sqrt(num / (denom1 * denom2))

def metric_Canberra(x1: list, x2: list) -> float:
    return sum([abs(e1-e2) / (abs(e1) + abs(e2))
                for e1, e2 in list(zip(x1, x2))])
```

```python
def estimate_mean(data: ndarray, label: list) -> ndarray:
    category = unique(label)
    means = []
    for c in category:
        mean = [x for x, w in list(zip(data, label)) if w == c]
        means.append(sum(mean)/len(mean))
    return np.array(means)

if __name__ == "__main__":
    path_data  = 'data.txt'
    path_label = 'label.txt'
    data = np.loadtxt(path_data, dtype='float', delimiter=None)
    label = np.loadtxt(path_label, dtype=np.int32, delimiter=None)
    mu = estimate_mean(data, label)
    f1 = lambda x: classifier(x, mu, metric_Manhattan)
    f2 = lambda x: classifier(x, mu, metric_Chebyshev)
    f3 = lambda x: classifier(x, mu,
                              lambda x1, x2:
                              metric_Minkowski(x1, x2, p=3))
    f4 = lambda x: classifier(x, mu, metric_Chordal)
    f5 = lambda x: classifier(x, mu, metric_Canberra)
    plot_region(f1)
    plot_data_label(data, label)
    list([plot_cluster_center(k, mean)
          for k, mean in list(zip([1, 2], mu))])
    plt.title(r"Separation for $S=\mathbb{R}^2$ equipped with "\
    + r"$d(x, x') = \sum_{k=1}^d |x_{(k)} - x^\prime_{(k)}|$")
    plt.savefig("fig_Manhattan.png", Bbox='tight')
    plt.show()
    plot_region(f2)
    plot_data_label(data, label)
    list([plot_cluster_center(k, mean)
          for k, mean in list(zip([1, 2], mu))])
    plt.title(r"Separation for $S=\mathbb{R}^2$ equipped with "\
    +r"$d(x, x')=\max_{k\in\{1,\cdots,d\}}|x_{(k)}-x^\prime_{(k)}|$")
    plt.savefig("fig_Chebyshev.png", Bbox='tight')
    plt.show()
    plot_region(f3)
    plot_data_label(data, label)
    list([plot_cluster_center(k, mean)
          for k, mean in list(zip([1, 2], mu))])
    plt.title(r"Separation for $S=\mathbb{R}^2$ equipped with "\
```

```
    + r"$d(x, x') = ||x-x'||_p, p=3$")
    plt.savefig("fig_Minkowski_p=3.png", Bbox='tight')
    plt.show()
    plot_region(f4)
    plot_data_label(data, label)
    list([plot_cluster_center(k, mean) \
          for k, mean in list(zip([1, 2], mu))])
    plt.title(r"Separation for $S=\mathbb{R}^2$ equipped with "\
    + r"$d(x, x') = \frac{2||x-x'||_2}{\sqrt{1+||x||_2^{2}}
\sqrt{1+||x'||_2^{2}}}$")
    plt.savefig("fig_Chordal.png", Bbox='tight')
    plt.show()
    plot_region(f5)
    plot_data_label(data, label)
    list([plot_cluster_center(k, mean)
          for k, mean in list(zip([1, 2], mu))])
    plt.title(r"Separation for $S=\mathbb{R}^2$ equipped with "\
    + r"$d(x, x') = \sum_{k=1}^d \frac{|x_{(k)} - x^\prime_{(k)}|}
{|x_{(k)}| + |x^\prime_{(k)}|}$")
    plt.savefig("fig_Canberra.png", Bbox='tight')
    plt.show()
```