Advanced Data Analysis
Homework Week 10

Aswin Vijay

July 3, 2023

1 Question

We are given the following squared energy distance:

$$\mathbf{D_E^2} = 2\,\mathbb{E}_{x'\sim p_{test}, x\sim q_\pi}\,||x'-x|| - \mathbb{E}_{x',\tilde{x}'\sim p_{test}}\,||x'-\tilde{x}'|| \\ - \mathbb{E}_{x,\tilde{x}\sim q_\pi}\,||x-\tilde{x}|| \tag{1}$$

$$q_\pi(x) = \pi p_{train}(x|y=+1) + (1-\pi)p_{train}(x|y=-1) \tag{2}$$

We need to derive the following for $D_E(p_{test,q_\pi})$ w.r.t π:

$$J(\pi) = (2A_{+1,-1} - A_{+1,+1} - A_{-1,-1})\pi^2 - 2(A_{+1,-1} - A_{-1,-1} - b_{+1} + b_{-1})\pi \\ + Const \tag{3}$$

where,

$$A_{y,\tilde{y}} = \mathbb{E}_{x\sim p_{train}, \tilde{x}\sim p_{train}(x|\tilde{y})}\,||x-\tilde{x}|| \tag{4}$$

$$b_y = \mathbb{E}_{x'\sim p_{test}, x\sim p_{train}(x|y)}\,||x'-x|| \tag{5}$$

we are also given that,

$$\mathbb{E}_{\tilde{x}\sim q_\pi}[f(\tilde{x})] = \pi\,\mathbb{E}_{\tilde{x}\sim p_{train}(\tilde{x}|+1)}[f(\tilde{x})] + (1-\pi)\,\mathbb{E}_{\tilde{x}\sim p_{train}(\tilde{x}|-1)}[f(\tilde{x})] \tag{6}$$

Derivation $(x', \tilde{x}')$ and $(x, \tilde{x})$:

To derive the expression for $J(\pi)$, substitute (6) into (1) :

$$\begin{aligned}\mathbf{D_E^2} &= 2\,\mathbb{E}_{x'\sim p_{test}, x'\sim q_\pi}\,||x'-x|| - \mathbb{E}_{x',\tilde{x}'\sim p_{test}}\,||x'-\tilde{x}'|| - \mathbb{E}_{x,\tilde{x}\sim q_\pi}\,||x-\tilde{x}|| \\ &= 2\,\mathbb{E}_{x'\sim p_{test}, x'\sim q_\pi}\,||x'-x|| - \mathbb{E}_{x',\tilde{x}'\sim p_{test}}\,||x'-\tilde{x}'|| - \mathbb{E}_{x\sim q_\pi}[\pi\,\mathbb{E}_{\tilde{x}\sim p_{train}(\tilde{x}|y=+1)}\,||x-\tilde{x}|| \\ &\quad + (1-\pi)\,\mathbb{E}_{\tilde{x}\sim p_{train}(\tilde{x}|y=-1)}\,||x-\tilde{x}||]\end{aligned}$$

Rearranging we get:

$$\begin{aligned}\mathbf{D_E^2} &= 2\,\mathbb{E}_{x'\sim p_{test}}\,[\mathbb{E}_{x\sim q_\pi}\,||x'-x||] - \mathbb{E}_{x',\tilde{x}'\sim p_{test}}\,||x'-\tilde{x}'|| \\ &\quad - \pi\,\mathbb{E}_{x\sim q_\pi, \tilde{x}\sim p_{train}(\tilde{x}|y=+1)}\,||x-\tilde{x}|| - (1-\pi)\,\mathbb{E}_{x\sim q_\pi, \tilde{x}\sim p_{train}(\tilde{x}|y=-1)}\,||x-\tilde{x}||\end{aligned}$$

For the first term we get:

$$
\begin{aligned}
2\,\mathbb{E}_{x'\sim p_{test}}\left[\mathbb{E}_{x\sim q_\pi}||x'-x||\right] &= 2\,\mathbb{E}_{x'\sim p_{test}}\Big[\,\pi\,\mathbb{E}_{x\sim p_{train}(x|+1)}[||x'-x||] + (1-\pi)\,\mathbb{E}_{x\sim p_{train}(x|-1)}[||x'-x||]\Big] \\
&= 2\pi\left(\mathbb{E}_{x'\sim p_{test},x\sim p_{train}(x|y=+1)}||x'-x||\right) + 2(1-\pi)\left(\mathbb{E}_{x'\sim p_{test},x\sim p_{train}(x|y=-1)}||x'-x||\right) \\
&= 2\pi b_{+1} + 2(1-\pi)b_{-1}\ using\ (5)
\end{aligned}
$$

For the second term:

$$
\mathbb{E}_{x'\sim p_{test},\tilde{x}'\sim p_{test}}||x'-\tilde{x}'|| = const\ (independent\ of\ \pi)
$$

For the third term:

$$
\begin{aligned}
&\pi\,\mathbb{E}_{x\sim q_\pi,\tilde{x}\sim p_{train}(x|y=+1)}||x-\tilde{x}|| = \pi\,\mathbb{E}_{x\sim q_\pi}||x-\tilde{x}||\cdot\mathbb{E}_{\tilde{x}\sim p_{train}(\tilde{x}|y=+1)}||x-\tilde{x}|| \\
&= \pi(\pi\,\mathbb{E}_{x\sim p_{train}(x|y=+1)}||x-\tilde{x}|| + (1-\pi)\,\mathbb{E}_{x\sim p_{train}(x|y=-1)}||x-\tilde{x}||)\,\mathbb{E}_{\tilde{x}\sim p_{train}(\tilde{x}|y=+1)}||x-\tilde{x}|| \\
&= \pi^2(\mathbb{E}_{x\sim p_{train}(x|y=+1),\tilde{x}\sim p_{train}(\tilde{x}|y=+1)}||x-\tilde{x}||) + (\pi-\pi^2)(\mathbb{E}_{x\sim p_{train}(x|y=-1),\tilde{x}\sim p_{train}(\tilde{x}|y=+1)}||x-\tilde{x}||) \\
&= \pi^2 A_{+1,+1} + (\pi-\pi^2)A_{-1,+1}\ using\ (4)
\end{aligned}
$$

Similarly the fourth term can be written as:

$$
\begin{aligned}
(1-\pi)\,\mathbb{E}_{x\sim q_\pi,\tilde{x}\sim p_{train}(x|y=-1)}||x-\tilde{x}|| &= \pi\,\mathbb{E}_{x\sim q_\pi}||x-\tilde{x}||\cdot\mathbb{E}_{\tilde{x}\sim p_{train}(\tilde{x}|y=-1)}||x-\tilde{x}|| \\
&= (\pi-\pi^2)A_{+1,-1} + (1-\pi)^2 A_{-1,-1}\ using\ (4)
\end{aligned}
$$

Substituting these results back into $\mathbf{D^2_E}$ we get:

$$
\begin{aligned}
\mathbf{D^2_E} &= J(\pi) = 2\pi b_{+1} + 2(1-\pi)b_{-1} - \pi^2 A_{+1,+1} - (\pi-\pi^2)A_{-1,+1} - (\pi-\pi^2)A_{+1,-1} - (1-\pi)^2 A_{-1,-1} \\
&= (2A_{+1,-1} - A_{+1,+1} - A_{-1,-1})\pi^2 - 2(A_{+1,-1} - A_{-1,-1} - b_{+1} + b_{-1})\pi + const \\
&\quad (A_{+1,-1} = A_{-1,+1})
\end{aligned}
$$

Thus, we have derived the expression for $J(\pi)$. The constant terms independent of π has been absorbed into the const term in $J(\pi)$.

class-balance weighted least squares method with linear model

In [ ]:
```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
np.random.seed(1)
```

In [ ]:
```
def generate_data(n_total, n_positive):
    x = np.random.normal(size=(n_total, 2))
    x[:n_positive, 0] -= 2
    x[n_positive:, 0] += 2
    x[:, 1] *= 2.
    y = np.empty(n_total, dtype=np.int64)
    y[:n_positive] = 0
    y[n_positive:] = 1
    return x, y
```

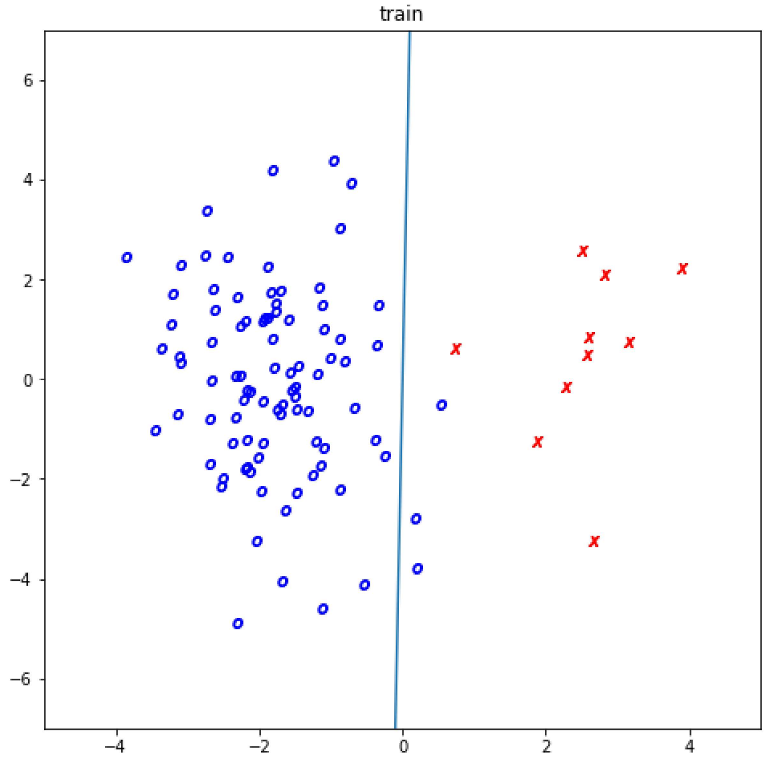
In [ ]:
```
def cwls(train_x, train_y, test_x):

    # implement this function
    N = len(train_x)
    n = len(test_x)
    train_x = np.concatenate( (np.ones((N,1)),train_x) , axis = 1)
    test_x = np.concatenate( (np.ones((n,1)),test_x) , axis = 1)
    phi_train = np.sqrt(np.sum((train_x[None] - train_x[:, None])**2, axis=2))
    phi_test = np.sqrt(np.sum((train_x[None] - test_x[:, None])**2, axis=2))
    labels = np.array([0,1])
    n_i = np.array([len(train_y[train_y == labels[0]])/N, len(train_y[train_y == labels[1]])/N])
    nl = len(labels)
    A = np.zeros((nl,nl))
    b = np.zeros(nl)

    for i in range(nl):
        ind_i = train_y == labels[i]
        b[i] = np.mean(phi_test[:,ind_i])
        for j in range(nl):
            ind_j = train_y == labels[j]
            A[i,j] = np.mean((phi_train[ind_i])[:,ind_j])

    tilde_pi = (A[0,1] - A[1,1] - b[0] + b[1])/(2*A[0,1] - A[0,0] - A[1,1])
    hat_pi = min(1,max(0,tilde_pi))

    # Compute weighting probability
    hat_pi = np.array([hat_pi, 1 - hat_pi])
    weight = hat_pi[train_y - 1] / n_i[train_y - 1]
    target = 2*train_y - 3
    W = np.tile(weight, (train_x.shape[1],1)).T

    # Regression
    theta = np.linalg.solve(train_x.T @ (W * train_x), train_x.T @ (weight * target))
    theta_uw = np.linalg.solve(train_x.T @ train_x, train_x.T @ target)

    return theta, theta_uw
```

In [ ]:
```
def visualize(train_x, train_y, test_x, test_y, theta):
    for x, y, name in [(train_x, train_y, 'train'), (test_x, test_y, 'test')]:
        plt.clf()
        plt.figure(figsize=(8, 8))
        plt.xlim(-5., 5.)
        plt.ylim(-7., 7.)
        lin = np.array([-5., 5.])
        plt.plot(lin, -(theta[2] + lin * theta[0]) / theta[1])
        plt.scatter(x[y == 0][:, 0], x[y == 0][:, 1],marker='$O$', c='blue')
        plt.scatter(x[y == 1][:, 0], x[y == 1][:, 1],marker='$X$', c='red')
        plt.title(name)
        plt.show()
    #plt.savefig('lecture8-h3-{}.png'.format(name))
```

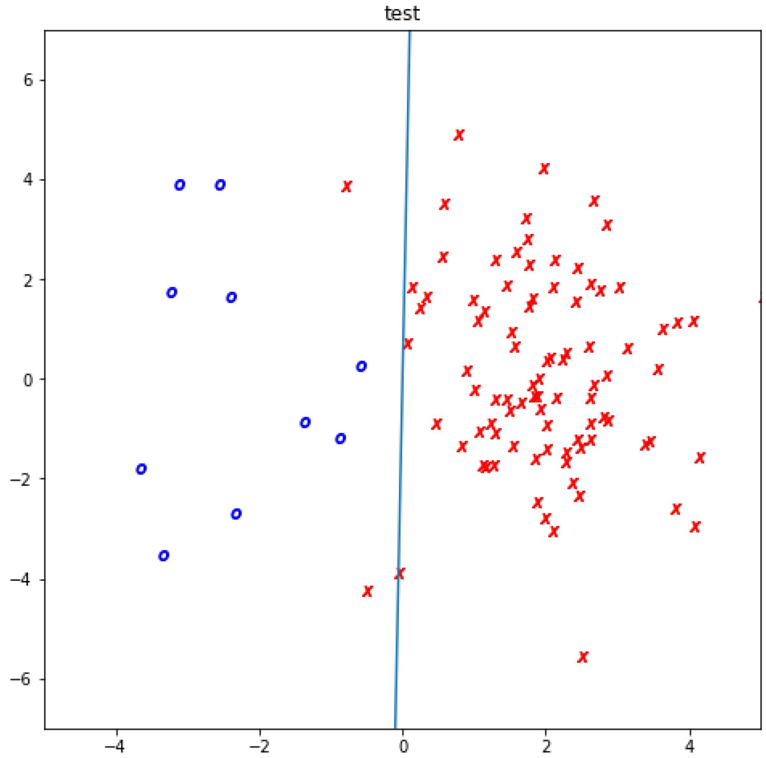
Visualize the class-balance weighted regression

In [ ]:
```
train_x, train_y = generate_data(n_total=100, n_positive=90)
eval_x, eval_y = generate_data(n_total=100, n_positive=10)
theta, theta_uw = cwls(train_x, train_y, eval_x)
visualize(train_x, train_y, eval_x, eval_y, theta)
```

<Figure size 432x288 with 0 Axes>



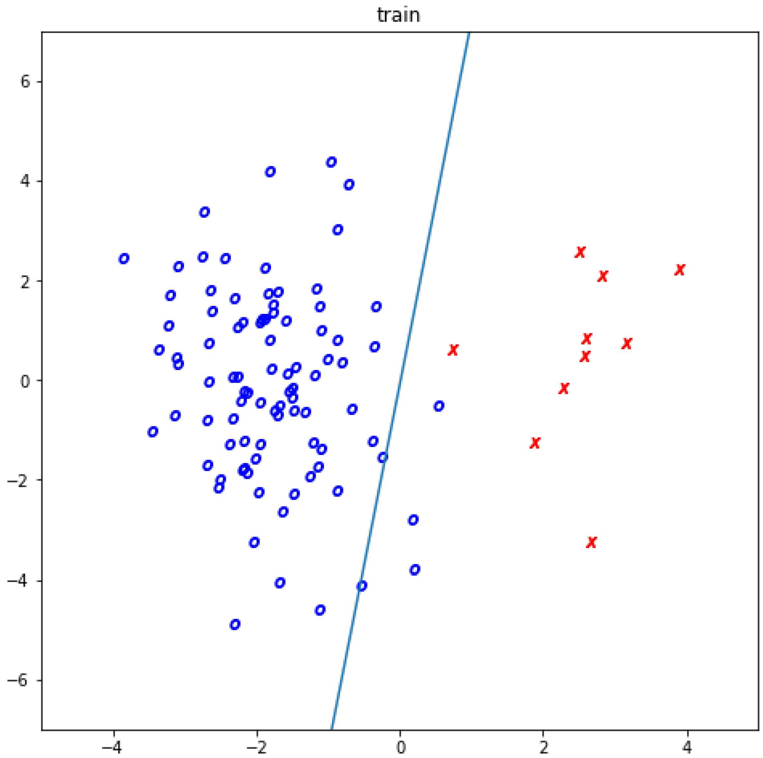
<Figure size 432x288 with 0 Axes>



Visualize Normal Regression

In [ ]:
```
visualize(train_x, train_y, eval_x, eval_y, theta_uw)
```

<Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>