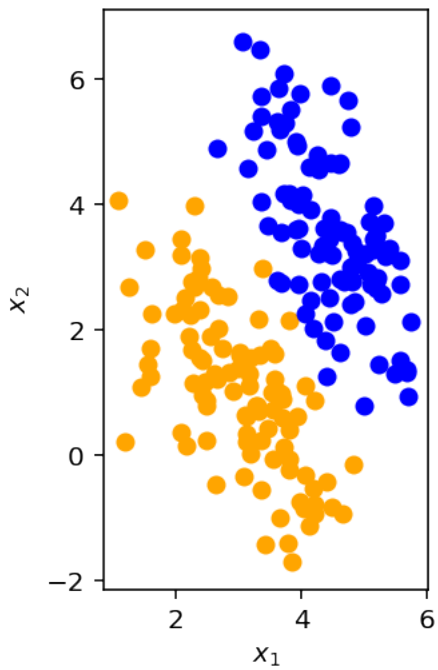
hw4-q3-lda

February 14, 2018

```
In [2]: import numpy as np
        import matplotlib.pyplot as plt
        %config InlineBackend.figure_format = 'retina'

In [3]: # Load the data and visualize.
        Xs = np.load('hw4-q3-lda.npy')

        X_0 = np.matrix(Xs[:, 0:2]).T  # Shape: (2, 100).
        X_1 = np.matrix(Xs[:, 2:4]).T  # Shape: (2, 100).

        print(X_0.shape, X_1.shape)
        plt.scatter(X_0[0].tolist(), X_0[1].tolist(), color='orange')
        plt.scatter(X_1[0].tolist(), X_1[1].tolist(), color='blue')
        plt.axis('scaled')
        plt.xlabel('$x_1$')
        plt.ylabel('$x_2$')
        plt.show()
```

```
(2, 100) (2, 100)
```




```
In [12]: # (a) Compute mean of each class.
         mu_0 = np.matrix([X_0[0].mean(), X_0[1].mean()]).T # Shape: (2, 1).
         mu_1 = np.matrix([X_1[0].mean(), X_1[1].mean()]).T # Shape: (2, 1).

         print(mu_0.shape, mu_1.shape)
         print('mu_0=\n{},\nmu_1=\n{}'.format(mu_0, mu_1))
```

```
(2, 1) (2, 1)
mu_0=
[[ 2.98351552]
 [ 1.06453902]],
mu_1=
[[ 4.46952033]
 [ 3.52885988]]
```

```
In [14]: # (b) Compute the covariance matrix for each class, Sigma_0 and Sigma_1.
         Sigma_0 = np.cov(X_0) # Shape: (2, 2).
         Sigma_1 = np.cov(X_1) # Shape: (2, 2).

         print(Sigma_0.shape, Sigma_1.shape)
         print('Sigma_0=\n{},\nSigma_1=\n{}'.format(Sigma_0, Sigma_1))
```

```
(2, 2) (2, 2)
Sigma_0=
```

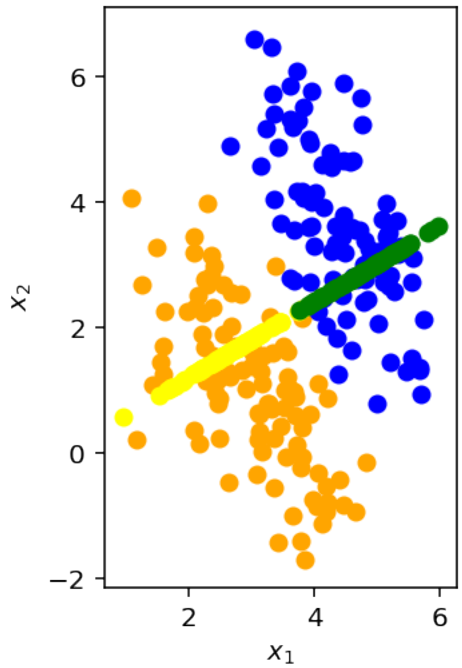
```
[[ 0.70628859 -0.6905174 ]
 [-0.6905174   1.61474336]],
Sigma_1=
[[ 0.48981843 -0.57477756]
 [-0.57477756  1.67666167]]
```

```
In [16]: from numpy import linalg as LA

         # (c) Find the optimal w_star and w_tilde_star with unit length.
         w_star       = LA.inv(Sigma_0 + Sigma_1).dot(mu_0 - mu_1) # Shape: (2, 1).
         w_tilde_star = w_star/LA.norm(w_star, 2) # Shape: (2, 1).

         print(w_star.shape, w_tilde_star.shape)
         print('w_star=\n{},\nw_tilde_star=\n{}'.format(w_star, w_tilde_star))
```

```
(2, 1) (2, 1)
w_star=
[[-3.42871346]
 [-2.06679356]],
w_tilde_star=
[[-0.85643702]
 [-0.51625152]]
```

```
In [19]: # (d) Compute the projection and plot the figure.
         Xproj_0 = w_tilde_star.T.dot(X_0).T.dot(w_tilde_star.T).T # Shape: (2, 100).
         Xproj_1 = w_tilde_star.T.dot(X_1).T.dot(w_tilde_star.T).T # Shape: (2, 100).

         print(Xproj_0.shape, Xproj_1.shape)
         plt.scatter(X_0[0].tolist(), X_0[1].tolist(), color='orange')
         plt.scatter(X_1[0].tolist(), X_1[1].tolist(), color='blue')
         plt.scatter(Xproj_0[0].tolist(), Xproj_0[1].tolist(), color='yellow')
         plt.scatter(Xproj_1[0].tolist(), Xproj_1[1].tolist(), color='green')
         plt.axis('scaled')
         plt.xlabel('$x_1$')
         plt.ylabel('$x_2$')
         plt.show()
```

```
(2, 100) (2, 100)
```


6
4
2
0
−2
2
4
6
x_2
x_1