Pattern Recognition - Assignment 2 - Feature selection

Name: Morten D. Laursen

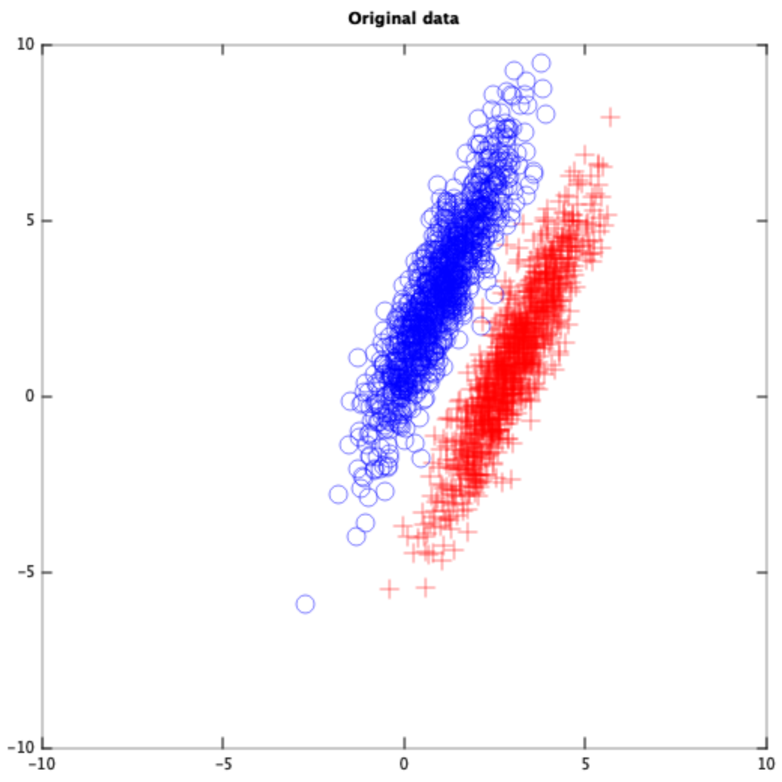
Student id: a0197363

1) Show scatter plot of bi-variate random numbers

To generate random numbers from a gaussian distribution with mean and sigma, we use `mvnrnd()`

```
% constants
m_1 = [3;1];
m_2 = [1;3];
sig_1 = [1 2; 2 5];
sig_2 = sig_1;
n = 1000; % number of random datapoint to generate x 2
c = 2;
% Exercise 1 - Initialize data
R1 = mvnrnd(m_1,sig_1,n);
R2 = mvnrnd(m_2,sig_2,n);
```

And we draw the following scatterplot



2) Apply PCA

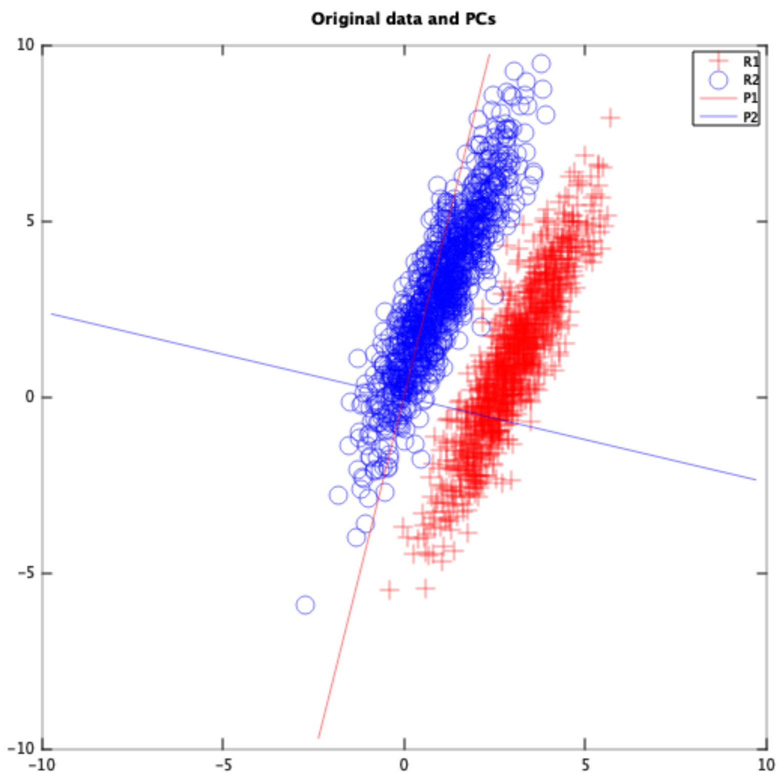
To apply PCA, we follow the following steps:

1. As we have assumed it's a single data set, we first merge the data.
2. Next we get the covariance matrix
3. From this we get the eigenvectors and the eigenvalue
4. We sort the eigenvector, so we know which row-vector is PC1

The above steps in code:

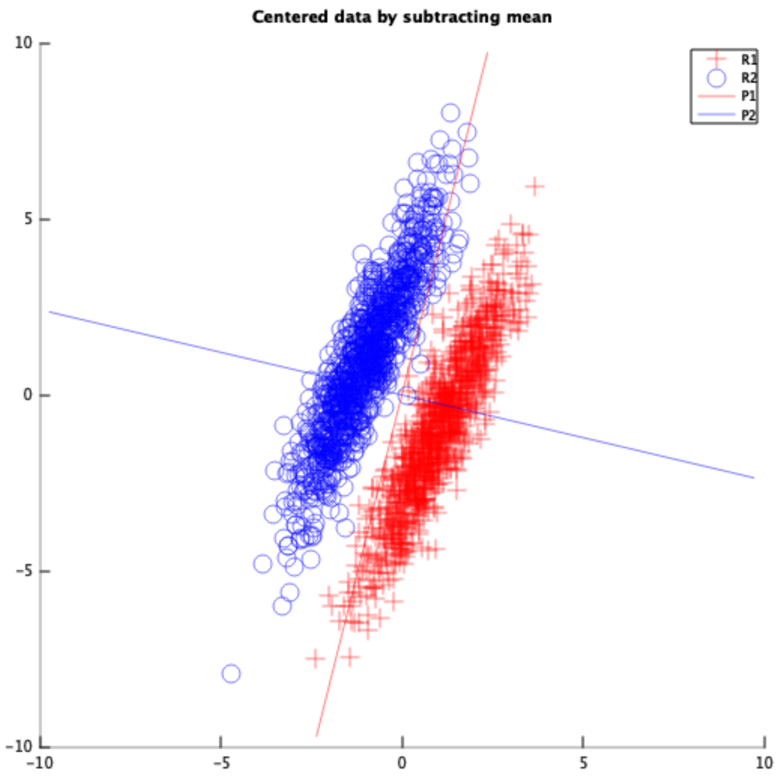
```
% Step 1: Merge data;
R = [R1;R2];
% step 2: Calculate covariance matrix
cov_R = cov(R);
% step 3: Calculate eigenvector and eigenvalues
[eig_vec, eig_val] = eig(cov_R);
% step 4: Eigenvalues are not ordered. Sort them
[eig_val, i] = sort(diag(eig_val), 'descend');
eig_vec = eig_vec(:,i);
```

Plot of original data and PCs



5. We centre data around 0 by subtracting the mean and thus being able to use the eigenvectors as the new axises.

Plot of centred data around PCs.

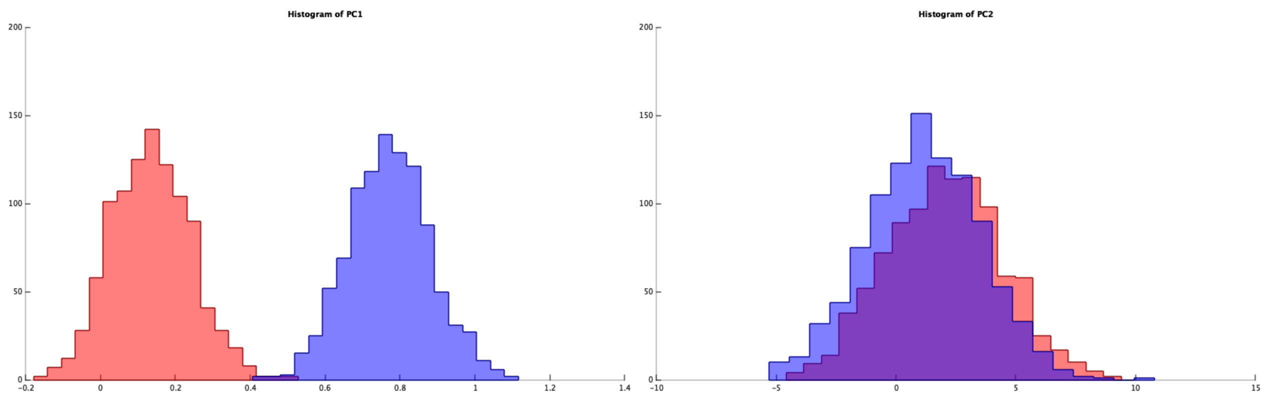


6. Project data on PC
7. We reconstruct vector x from PC

The above steps in code:

```
% step 5: Centre data by subtracting mean.
R_new = R-mean(R);
% step 6: project data on PC
y = R_new*eig_vec;
% Step 7: Reconstruct vector x from PC
x_hat = y*eig_vec + mean(R);
```

After this step all of our data will be on a straight line of the eigenvectors, and thus we will be able to display it as a single dimension histogram:



3) Apply LDA

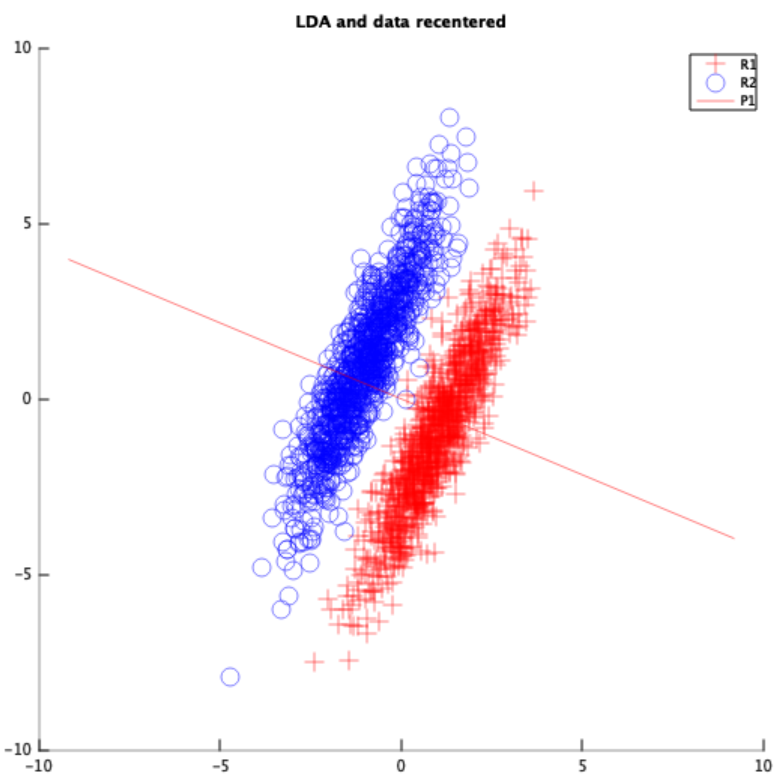
To apply LDA we follow the following steps:

1. As we are that the two distributions of data are of two different classes, we first calculate the mean of the two classes.
2. Next we calculate the between-class scatter, Sb
3. Then we calculate the mean within-class scatter, Sw
4. We get the eigenvector and eigenvalue from this ratio S
5. We sort the eigenvectors
6. Centre data around 0 by subtracting mean

The above steps in code:

```
% Step 1: Get mean of the two classes.
mu = [mean(R1); mean(R2)];
% Step 2: Calculate between-class scatter
Sb = (ones(c,1) * mean(R) - mu)' * (ones(c,1) * mean(R) - mu);
% Step 3: Calculate mean within-class scatter
Sw = (R - mu(c,:))'*(R - mu(c,:));
% Step 4: get eigenvector and eigenvalue of maxmimum between-class scatter to
within-class scatter ratio.
[eig_vec, eig_val] = eig(Sb/Sw);
% Step 5: Sort eigenvectors
[eig_val, i] = sort(diag(eig_val), 'descend');
eig_vec = eig_vec(:,i);
% Step 6: Center data by substracting mean.
R_new = R-mean(R);
```

LDA and the data centred around 0:

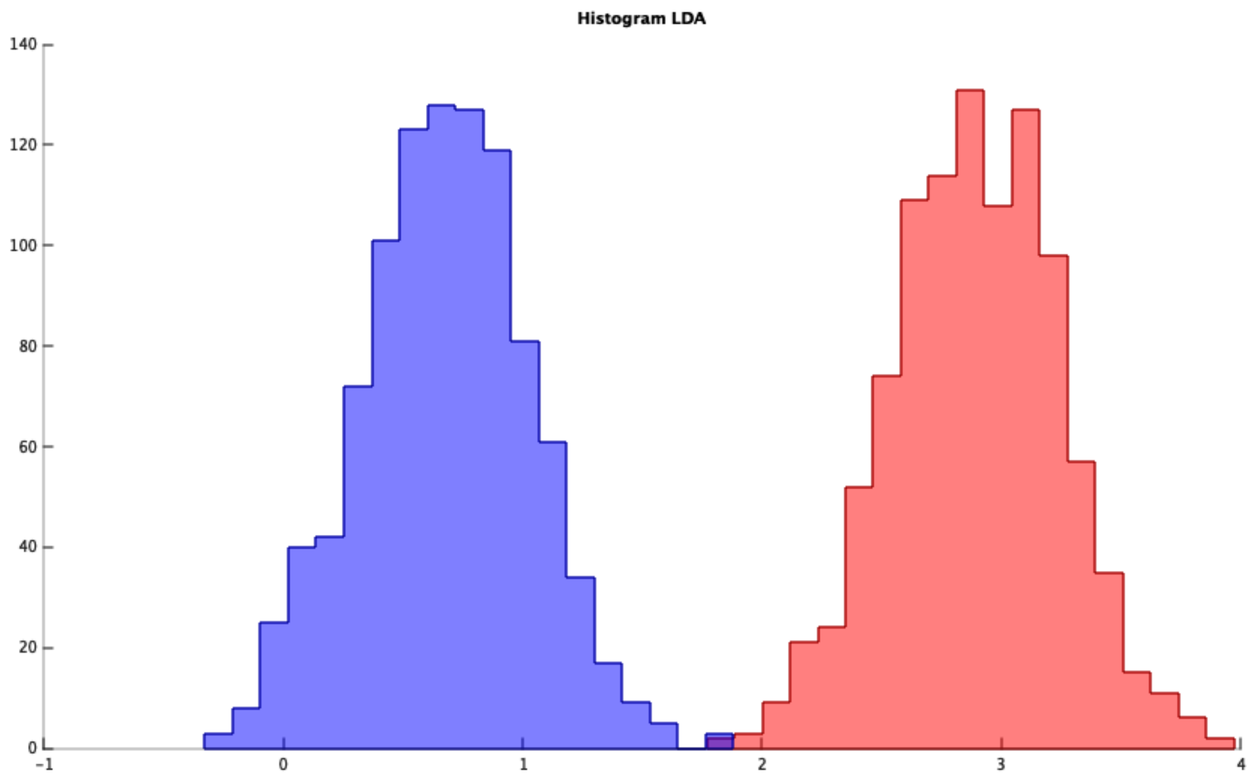


7. Project data on LDA axis
8. Reconstruct data

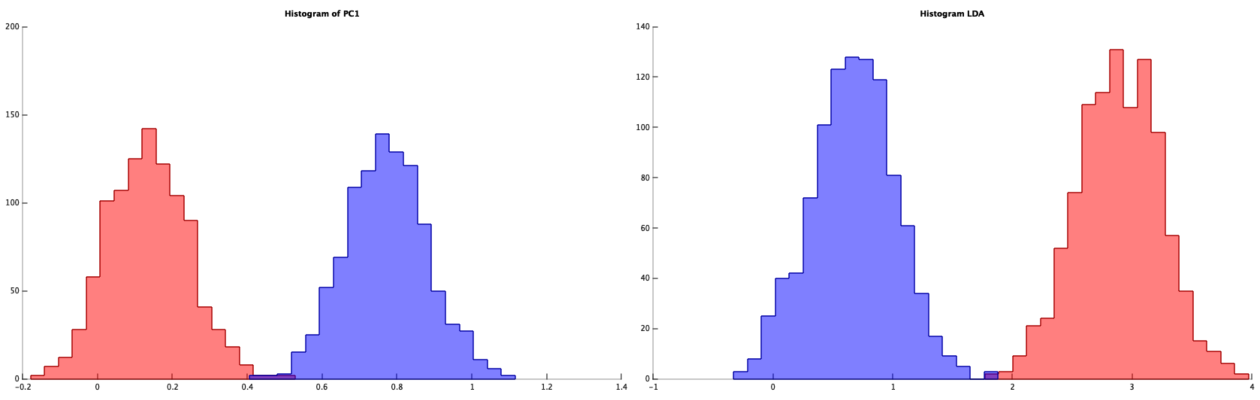
In code:

```
% Step 7: Project data on LDA
y = R_new*eig_vec;
% Step 8: Reconstruct data
x_hat = y(:,1)*eig_vec(:,1)' + mean(R);
```

And now we can plot the histogram again as well.



4) Show the histograms



5) Discussion

In this assignment we have seen how we are able to feature select and thus making very complicated data seem simpler, but still being able to keep sufficient accuracies.

When comparing the two histograms in section 4, we see that these are very much alike as expected. The methods vary a bit and therefore they are not totally similar.