MLCC Laboratory 3: Dimensionality reduction, feature selection

In this laboratory we will address the problem of data analysis and dimensionality reduction with a reference to a classification problem.

Think hard before you call the instructors or you look at the solution file!

1 Warm up - data generation

You will generate a training and a test set of `D`-dimensional points (`N` points for each class), with `N = 100` and `D = 30`. Only two of those dimensions will be meaningful, the other one will be a variable we will modify.

- **1.A** For each point, the first two variables will be generated by `MixGauss`, extracted from two gaussian distributions with centroids `(1, 1)` and `(-1,-1)` and standard deviation `0.7` (the first one with labels `1`, the second with label `-1`):

```
[Xtr, Ytr] = MixGauss(...);
Ytr(Ytr==2)= -1;
[Xts, Yts] = MixGauss(...);
Yts(Yts==2) = -1;
```

- **1.B.** You may want to plot the relevant variables of the data:

```
scatter(Xtr(:,1), Xtr(:,2), 50, Ytr, 'filled');
hold on; scatter(Xts(:,1), Xts(:,2), 50, Yts);
```

- **1.C** The remaining variables will be generated as gaussian noise:

```
sigma_noise = 0.01;
Xtr_noise = sigma_noise*randn(2*N, D-2);
Xts_noise = sigma_noise*randn(2*N, D-2);
```

  To compose the final data matrix, run:

```
Xtr = [Xtr, Xtr_noise];
Xts = [Xts, Xts_noise];
```

2 Principal Component Analysis (PCA)

- **2.A** Compute the data principal components (see "`help PCA`").

- **2.B** Plot the first two components of `X_proj` using the following line:

```
scatter(X_proj(:, 1), X_proj(:, 2), 50, Ytr, 'filled');
```

- **2.C** Try now with the first 3 components, by using:

```
scatter3(X_proj(:, 1), X_proj(:, 2), X_proj(:, 3), 50, Ytr, 'filled');
```

Reason on the meaning of the results you are obtaining.

- **2.D** Display the `sqrt` of the first `10` eigenvalues and plot the coefficients (eigenvector) associated with the largest eigenvalue:

```
disp(sqrt(d(1:10)));
scatter(1:D, abs(V(:,1)));
```

- **2.E** Repeat the above steps with dataset generated using different `sigma_noise = 0, 0.01, 0.1, 0.5, 0.7, 1, 1.2, 1.4, 1.6, 2`. To what extent data visualization by PCA is affected by the noise?

3 Variable selection

- **3.A** Use the data generated in part 1. Standardize the data matrix, so that each column has mean `0` and standard deviation `1`:

```
m = mean(Xtr); %(see "help mean")
s = std(Xtr); %(see "help std")
for i = 1:2*N
  Xtr(i,:) = Xtr(i,:) - m;
  Xtr(i,:) = Xtr(i,:) ./ s;
end
```

Do the same for `Xts`, by using `m` and `s` computed on `Xtr`.

- **3.B** Use the orthogonal matching pursuit algorithm (type "`help OMatchingPursuit`").

- **3.C** You may want to check the predicted labels on the training set:

```
Ypred = sign(Xts * w);
err = calcErr(Yts, Ypred);
```

and plot the coefficients `w` with `scatter(1:D, abs(w))`. How the error changes with the number of iterations of the method?

- **3.D** By using the method `holdoutCVOMP` find the best number of iterations with `intIter = 2,...,D` (and, for instance, `perc = 0.75, nrip = 20`).

Moreover, plot the training and validation error with the following lines:

```
figure; plot(intIter, Tm, 'r');
hold on; plot(intIter, Vm, 'b');
xlabel('Number of iterations for OMP'); ylabel('error');
legend('Test', 'Validation');
```

What is the behavior of the training and the validation errors with respect to the number of iterations?

- **3.E** Try to increase the number of relevant variables `D = 3,5,...` (and the corresponding standard deviation of the Gaussians) around the centroids:

```
ones(D, 1); % vector of all 1s
% and
-ones(D, 1); % vector of all -1s
```

  and see how this change is reflected in the cross-validation.

4 If you have time - more experiments

- **4.A** Analyse the results you obtain on sections 2 and 3 once you choose:
  - $N \gg D$
  - $N \approx D$
  - $N \ll D$,

  and evaluate the benefits of the two different analyses.

- **4.B** Dimensionality reduction is often used as a pre-procesing step to a learning (classification) algorithm. The idea is to perform classificatiton in a lower dimensional space and therefore safe computational time. You have the following task:
  - Generate a new training and test datasets as in Laboratory 1.
  - Perform dimensionality reduction on the training set.
  - Using the projection you just found, project the test set.
  - Perform kNN in the lower dimensional (projected) space. Compare the result (both accuracy and running time) with the one in Laboratory 1.