# Machine Learning for Medicine TP 6

## Dimensionality Reduction
## Principal Component Analysis

The goal of the TME is to understand how to perform dimensionality reduction and, in particular, get skills to use Principal Component Analysis.

**Data** (both data sets are provided)

- Molecular classification of leukemia data set of *Golub et al. 1999* contains gene expressions of 72 patients and 3562 genes.
- Breast cancer data set

You will need to load at least the following packages:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import svm
from sklearn.decomposition import PCA, IncrementalPCA, KernelPCA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
```

**Analysis**

Repeat the same analyses for the two data sets.

To read the data:

- For the *Golub et al. 1999* data

```
X = pd.read_csv('data/Golub_X',sep=' ') # Observations
y = pd.read_csv('data/Golub_y',sep=' ') # Classes
```

- For the Breast cancer data

```
X = pd.read_csv('data/Breast.txt',sep=' ')
y = X.as_matrix()[:,30] # Classes
X = X.as_matrix()[:,0:29] # Observations
```

1. Perform PCA to cluster the observations

   Note: fix the number of components = 2 in order to visualize the clustering in 2-D (or fix it to 3, and plot it in 3-D)

   Here is a simple example how to do it

```
pca = decomposition.PCA(n_components=2)
pca.fit(X)
X_pca = pca.transform(X)

plt.scatter(X_pca[:, 0], X_pca[:, 1],marker='o', c=y,s=25, edgecolor='k')
```

   A link to a more complex example (vizualisation in 3-D):

   `http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html`

2. Perform and visualize the results of the kernel PCA

   http://scikit-learn.org/stable/auto_examples/decomposition/plot_kernel_pca.html#sphx-glr-auto-examples-decomposition-plot-kernel-pca-py

3. Test an incremental PCA which is used if a data set is too big to be fitted into the memory

   http://scikit-learn.org/stable/auto_examples/decomposition/plot_incremental_pca.html#sphx-glr-auto-examples-decomposition-plot-incremental-pca-py

4. Quite often, in real high-dimensional applications, one first performs PCA to reduce the dimensionality, and then runs prediction on the reduced data. Run a PCA to reduce the original dimension (test a different number of components, e.g., 2, 5, 10, 20), and then try to predict with

   - Logisitc regression
   - Support vector machines (use the rbf kernel)

5. Compare the LDA (Linear Discriminant Analysis) and the PCA. You can get inspiration from the following example:

   http://scikit-learn.org/stable/auto_examples/decomposition/plot_pca_vs_lda.html#sphx-glr-auto-examples-decomposition-plot-pca-vs-lda-py

6. Which data set is "easier" for prediction, i.e., in which data sets the classes are better separated?

7. Was it useful to reduce the dimension of the original problems before running classification? Are the error rates of the models learned from the reduced data smaller than ones from the original (high-dimensional) data?

References

*A Tutorial on Principal Component Analysis*
https://arxiv.org/pdf/1404.1100.pdf