SGN-41007 Pattern Recognition and Machine Learning

Exercise Set 1: January 7 – 11, 2019

Exercises consist of both pen&paper and computer assignments. Pen&paper questions are solved at home before exercises, while computer assignments are solved during exercise hours. The computer assignments are marked by **python** and Pen&paper questions by **pen&paper**

On the first week all exercises are **python** tasks.

Before you start, load all the data for all questions from the following links. We will need them in later weeks as well.

`http://www.cs.tut.fi/courses/SGN-41007/Ex1_data.zip`
`http://www.cs.tut.fi/courses/SGN-41007/least_squares_data.zip`
`http://www.cs.tut.fi/courses/SGN-41007/locationData.zip`

---

1. **python** *Load CSV file into Python workspace.*

   Extract the contents of `locationData.zip`. After that:

   a) Read the file into a *numpy* array using `numpy.loadtxt` function. Search for instructions using Google.

   b) When loaded, print the array shape using `numpy.shape`. It should be 600×3.

2. **python** *Plot the contents of the loaded matrix.*

   Using the same data as in the previous exercise:

   a) Create a 2D plot of the first two columns of the matrix. You need to import `matplotlib.pyplot` and execute something like

   ```
   plt.plot(<column 1>, <column 2>)
   ```

   b) Create a 3D plot of all 3 columns. You will need to create a special subplot for this purpose with

   ```
   ax = plt.subplot(1, 1, 1, projection = "3d")
   ```

   After this, the plotting is done as

   ```
   plt.plot(<column 1>, <column 2>, <column 3>)
   ```

3. **python**

   a) Read the file `locationData.csv` into memory one line at a time (in a for loop). See similar example at the end of lecture slide set 1.

   b) Load the same data into another variable using `numpy.loadtxt`. Check that the contents of the two arrays are equal using `numpy.all` or `numpy.any`.

4. **python** *Load Matlab data into Python.*

   a) Load the file `twoClassData.mat` into Python. This can be done as follows.

```
>>> from scipy.io import loadmat
>>> mat = loadmat("twoClassData.mat")
```

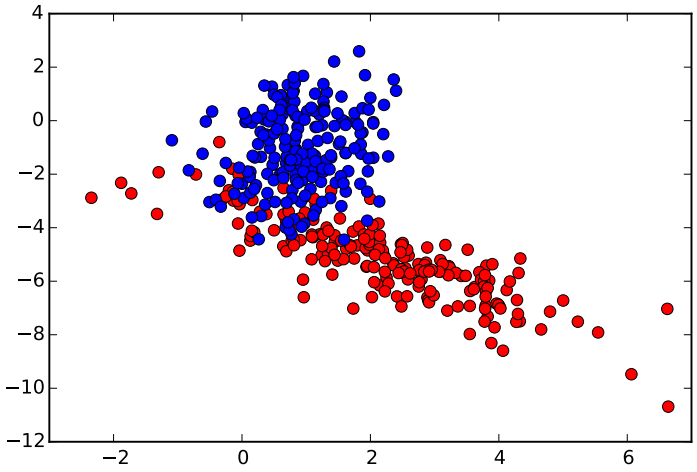
   This generates a **dict** structure, whose elements can be accessed through their names.

```
>>> print(mat.keys()) # Which variables mat contains?
['y', 'X', '__version__', '__header__', '__globals__']
>>> X = mat["X"] # Collect the two variables.
>>> y = mat["y"].ravel()
```

   The function `ravel()` transforms y from 400×1 matrix into a 400-length array. In Python these are different things unlike Matlab.

   b) The matrix X contains two-dimensional samples from two classes, as defined by y. Plot the data as a scatter plot like the picture below. Hints:

   - You can access all class 0 samples from X as: `X[y == 0, :]`.
   - The samples can be plotted like: `plt.plot(X[:, 0], X[:, 1], 'ro')`



5. **python** *Least squares fit.*

   Extract the contents (two numpy arrays) of `least_squares_data.zip` and open in numpy (see `numpy.load`). We want to model the relationship between the two variables using the model:

$$y(n) = ax(n) + b.$$

   Find the least squares estimates $\hat{a}$ and $\hat{b}$ that minimize the squared error.

   Hint: Search for function `numpy.linalg.lstsq` and study how it is used.