SGN-41007 Pattern Recognition and Machine Learning

Exercise Set 2: January 15 – 19, 2017

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**

1. **pen&paper** *Least squares fit*. Two measurements $x(n)$ and $y(n)$ depend on each other in a linear manner, and there are the following measurements available:

n	0	1	2
$x(n)$	7	9	2
$y(n)$	11.6	14.8	3.5

   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.

2. **python** *Least squares fit (like question 1 but with numpy)*. Download the following dataset onto your machine

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

   Extract the contents (two numpy arrays) 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.

3. **python** *Same as last week (Moodle) Question 1, but without numpy.*
   Download the following file and extract the contents:

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

   a) Read the file 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) Download the following file to your local folder:

   `http://www.cs.tut.fi/courses/SGN-41007/exercises/twoClassData.mat`

   b) Load the file contents into Python. This can be done as follows.

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

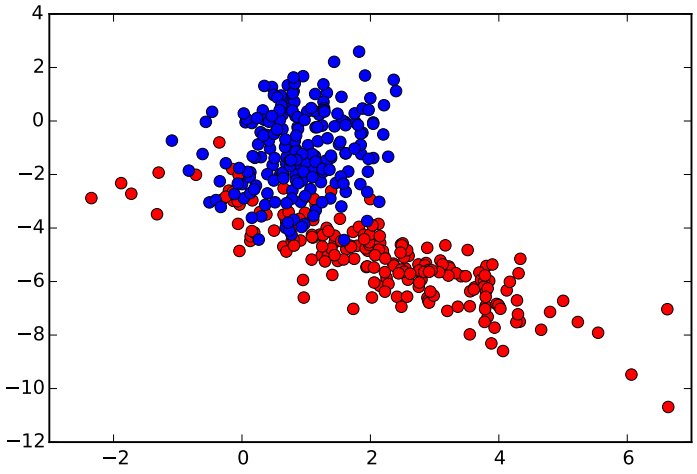
   This generates a **dict** structure, whose elements can be accessed through their names.

```
>>> 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.

   c) 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** *Remove the uneven illumination from a microscope image using least squares fitting like we did at the lecture.*

   Download the following file to your local folder:

   `http://www.cs.tut.fi/courses/SGN-41007/exercises/uneven_illumination.jpg`

Load the image into numpy array `Z` using the `imread` function of `matplotlib.image`[1]. Finally, show the image on screen (using `matplotlib.pyplot.imshow`) and check that the image shape is 1300×1030. Let's next fit a 2nd order surface to the grayscales.

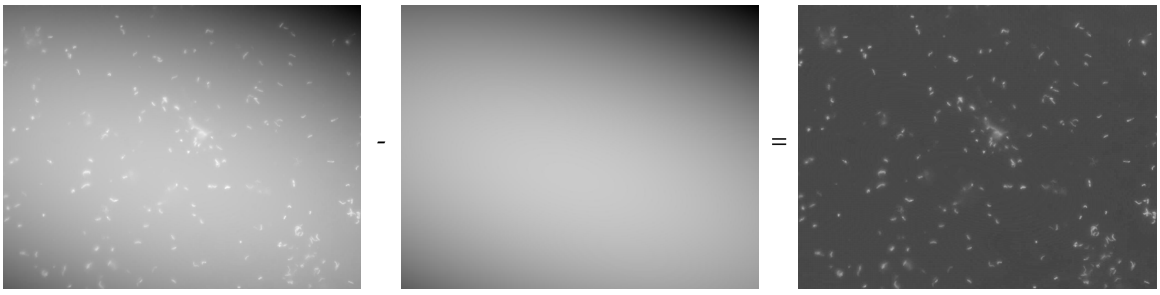
a) Create the explanatory variables (all x,y-coordinates in a matrix): `X, Y = np.meshgrid(range(1300), range(1030))`

b) Vectorize the matrices $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ using `ravel`, *e.g.*, `z = Z.ravel()`.

c) Prepare the design matrix $\mathbf{H}$ like in the lectures and solve the LS coefficients $\mathbf{c}$.

d) Compute the model prediction as `z_pred = np.dot(H, c)` and resize the vector to the original size.

e) Subtract the model prediction from the original image and show the result on screen.



[1]Note: There are several other ways of doing this, such as: `matplotlib.image.imread`, `scipy.ndimage.imread`, `PIL.Image.open` or `cv2.imread`