## APPC

# TP 7 Proximal Gradient & SVM

### Thomas Robert

## Prepare data

First we prepare the data: we standardize the data and create 2 datasets for learning and testing.

```
1 close all;
2 data = load('housing.data');
4 % make X and y matrices
[n,d] = size(data);
p = d-1;
7 X = data(:, 1:p);
s y = data(:,d);
_{10} % standardize feature values and create target
mu_y = mean(y);
12 y = y - mu_y;
y(y >= 0) = 1;
y(y < 0) = -1;
16 [X, mu, sigma] = standardizeCols(X);
_{17} X = [X \text{ ones}(n,1)];
p = p + 1;
_{\rm 20} % Split learn and test
{\tiny \texttt{21}} \ [Xlearn\,,\ ylearn\,,\ Xtest\,,\ ytest\,] \ = \ split data \, (X,\ y\,,\ 0.5)\,;
```

#### Proximal SVM

```
rho = 0.00005;
lambda = 24;

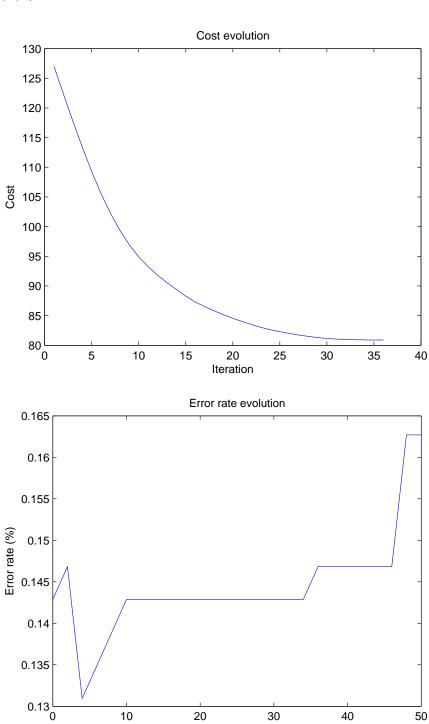
tic
[w, Js] = proximalSVM(Xlearn, ylearn, rho, lambda);
disp(['Proximal time : 'int2str(toc*1000) 'ms']);

Proximal time : 29 ms
```

#### Error rate vs lambda

#### Plots

```
_{1} % Plot cost evolution
 2 figure;
 з plot(Js);
 4 title('Cost evolution');
 5 xlabel('Iteration');
 6 ylabel ('Cost');
 _8 % Plot error rate evolution
9 figure;
10 plot(lambdas, errorRate)
11 title('Error rate evolution')
12 xlabel('\lambda penalization')
13 ylabel('Error rate (%)')
```



20

 $\boldsymbol{\lambda}$  penalization

30

40

50

10