Modern Theory of Estimation and Detection
Laboratory Exercise: Regression

The objective of this practical session is to implement a number of methods for linear and semi-linear regression to predict the stock prices of an air company using as input variables the stock of their competitors during the same days.

File *DatosP1.mat* contains the data to be used for this practice[1]. After loading this file you can check that you will have six variables in your Matlab workspace:

- `xTrain`: A matrix with observations containing the training data. The size of this matrix is 380 × 9, corresponding to 380 training patterns of length 9; i.e., each vector X consists of 9 different observations.
- `sTrain`: A vector of length 380 with values of the target variable S for each of the training patterns. Data are arranged in the same order than `xTrain`. This means that the *k*-th element in `sTrain` contains the value of S corresponding to the *k*-th row of `xTrain`.
- `xVal`: Matrix 380 × 9 containing the observations of the validation set.
- `sVal`: Vector of length 380 with the values of the target variable S for each of the patterns in `xVal`.
- `xTest`: Matrix 190×9 containing the observations of the test set.
- `sTest`: Vector of length 190 with values of the target variable S for each of the patterns in `xTest`.

Each row of matrices `xTrain`, `xVal` consists of the stock prices, in the same day, of 9 different airline companies. The objective of the estimators to design will be to predict the stock prize corresponding to a tenth airline company. In this practical session, estimators will be designed following a machine approach, for which we also have access to the real values of the target variable S corresponding to each row in the previous matrices. These values are stored in vectors `sTrain` and `sVal`, respectively.

For the design, we will use `xTrain` and `sTrain` to adjust the regression models, whereas `xVal` and `sVal` will be used to validate the models to guarantee that the constructed estimator generalizes well, i.e., it works correctly when applied to data different from the ones used during the design stage.

Once the estimator has been designed, it will be possible to estimate unknown values of S given the corresponding observation vectors. During the session, you can use variables `xTest` and `sTest` to assess the generalization performance of the designs.

The goals of this lab exercise are

- Designing linear and polynomial estimators.

[1] The dataset is an adaptation of the *Stock dataset* in http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html, taken originally from the *StatLib Repository*, in http://lib.stat.cmu.edu/.

- Analyze the feature selection problem.
- To illustrate the concept of *overfitting*. Along the session, students will be able to check that an estimator with an excessive number of free parameters can lead to suboptimal performance with respect to generalization error.
- To study how a validation set helps in the selection of an appropriate model that guarantees good generalization.

1. Data visualization and normalization.

Load the data file, and consider by now just the training data.

a) Visualize the data set:

In order to get a first (though incomplete) view of the problem, represent graphically the value of the variable to be estimated, s, as a function of each feature, x_i (i.e., the pairs $\{x_i^{(k)},\ s^{(k)}\}$ in the training data set).

b) Normalization:

With the aim of guaranteeing the numerical stability of subsequent sections, it is convenient to normalize the data. The fragment of code below implements a linear transformation so that the training data have zero mean and unit variance. Exactly the same transformation needs to be applied to validation and test data:

```
>> mx = mean(xTrain); stdx = std(xTrain);
xTrain = (xTrain - ones(nTrain,1)*mx)./(ones(nTrain, 1)*stdx);
xVal   = (xVal   - ones(nVal,1)*mx)./(ones(nVal,1)*stdx);
xTest  = (xTest  - ones(nTest,1)*mx)./(ones(nTest,1)*stdx);
```

Variables `nTrain`, `nVal` and `nTest` must contain, respectively, the number of training, validation, and test patterns.

2. Linear Regression

a) Calculate the coefficients of minimum Mean Square Error (MSE) linear regression, based on the training data, and compute the MSE measured over both the training and the test data sets. Compare these results to the MSE that would be obtained using "baseline" regression (i.e., without using any variables): $\hat{s} = w_0$.

b) Feature selection. Obtain the MSEs (calculated over the training set) for the regression based in all possible subsets of features and select, for each value $m \leq 9$, the best subset of m variables (i.e. the subset which provides a minimum MSE). Next, visualize the chosen variables for each value m and represent, as a function of the number of variables (m), their corresponding training and validation MSEs.

HINT: you can use Matlab *bitget* function to generate all possible subsets of variables.

3. Polynomial Regression with one variable

In this section we build polynomial regression models using variable x_1 only.

a) Study in detail the case of polynomial regression when using only one variable. Determine, for variable x_1, the regression coefficients corresponding to the models of order 0 (no variables are used), 1 (linear), 2, …, 8, and plot the training MSE as a function of the degree of the polynomial.

Depict the regression function corresponding to the designed polynomial of degree 8.

b) Generalization. In this subsection we will analyze the differences that are observed between the training and the validation MSEs. For this, compute again the corresponding regression coefficients using a subset of K samples randomly selected from the training data set; you could select, for instance, subsets of size K = 20, 40, 60, 80, 120, 140 and 160. Next, plot for each value K the achieved training and validation MSEs as a function of the degree of the polynomial (in the range 0 to 9). Given that the process of selecting K samples is random, it is convenient to average the previous MSEs over (at least) 1000 different selections of samples (for each K).

Using the figure you have just obtained, find, for each K, the degree of the polynomial obtaining the smallest MSE in the validation dataset, and depict such optimal degree as a function of K.

Finally, represent graphically the test MSE, as a function of K, using the optimal degree (selected using the validation dataset) in each case.