LELEC2870: Exercise Session 2
Linear Methods for Regression

1 Objectives

Linear methods often allow to obtain good results in regression, when the relationship between the features (or inputs) and the target (or output) is not too non-linear. In this second exercise session, you will implement linear regression for a set of features that you will select using simple linear methods.

2 Dataset and Notations

The datasets for this session are available on the Moodle website. After loading the `diabetes.mat` file, you will be able to work on a total of 442 instances coming from a diabetes study. For each one, the values of 10 features are given, as well as the target value. The features include the age (feature 1), the sex (feature 2), the body mass index (feature 3) and the blood pressure (feature 4) of the patient, as well as the result of several serum measurements (features 5 to 10). During this exercise session, it may be interesting to link your results with the feature's interpretation.
In this session, the learning set is $\{(\mathbf{x}_p, t_p)|p = 1...P\}$ where P is the number of samples, $\mathbf{x}_p$ is an input row vector of dimension D

$$\mathbf{x}_p = \begin{pmatrix} x_p^1 & x_p^2 & \cdots & x_p^D \end{pmatrix} \tag{1}$$

and t_p is the scalar output. For computations, inputs are placed in the matrix

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_P \end{pmatrix} = \begin{pmatrix} x_1^1 & x_1^2 & \cdots & x_1^D \\ x_2^1 & x_2^2 & \cdots & x_2^D \\ \vdots & \vdots & \ddots & \vdots \\ x_P^1 & x_P^2 & \cdots & x_P^D \end{pmatrix} \tag{2}$$

and targets are placed in the column vector

$$\mathbf{t} = \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_P \end{pmatrix}. \tag{3}$$

3 Feature Selection with Correlation

Since 10 features are available, we'll ask you to select one (or more) of them to perform regression on in the following exercises. An example of a good selection criterion is the *correlation* between the feature itself and the target.

For each feature, **visualize** the relationship between the feature and the target (function `scatterplot` in the module `matplotlib`) and **compute** the correlation (function `corrcoef` in the module `numpy`). Can you use correlation to select the variables used by a linear model?

4 Univariate Linear Regression

For this first regression you'll have to **choose** one feature. Once you have selected it (based on your results from sect. 3), you can perform linear regression, which can be equivalently described as a simple neural network with only one layer and a linear activation. The analytical expression of its output is

$$y = \sigma(\mathbf{x}\mathbf{w} + w_0). \tag{4}$$

where $\mathbf{w}$ is the weights column vector, w_0 is the bias, σ is the activation function and y is the output of the network. In order to simplify notations, the bias is generally included in the inputs and weights vectors, i.e. $\mathbf{x}$ becomes

$$\begin{pmatrix} 1 & \mathbf{x} \end{pmatrix} \tag{5}$$

and $\mathbf{w}$ becomes

$$\begin{pmatrix} w_0 \\ \mathbf{w} \end{pmatrix}. \tag{6}$$

This convention is used in the rest of this exercise session. Since the activation is linear, the output of one-layer neural networks becomes simply

$$y = \mathbf{x}\mathbf{w}. \tag{7}$$

Notice that now, $\mathbf{x}$ and $\mathbf{w}$ are both columns vectors of length $(D + 1)$.

Implement linear regression using the pseudo-inverse method (function `inv` in the module `numpy.linalg`). **Visualize** the result of your regression. In particular, compare your prediction with the actual target values. Are you satisfied with the result ? How could you improve it?

5 Learning as Training Error Minimisation

For regression problems, the mean square error (MSE) is usually used to assess the quality of a model:

$$\begin{aligned} E(w|\mathbf{X}, \mathbf{T}) &= \frac{1}{P}\sum_{p=1}^{P}(y_p - t_p)^2 = \frac{1}{P}\|\mathbf{y} - \mathbf{t}\|^2 \\ &= \frac{1}{P}\left[\sigma(\mathbf{X}\mathbf{w}) - \mathbf{t}\right]^T\left[\sigma(\mathbf{X}\mathbf{w}) - \mathbf{t}\right] \end{aligned} \tag{8}$$

It estimates the expectation of the squared error made by the model. When the error is computed on the training set, it becomes a *training error*. This error can thus be used to adjust the model parameters and doing so reducing the overall error.

By design, the solution given by the pseudo-inverse method actually minimises the training MSE. **Verify** this property by:

- **visualizing** the value of the training MSE as a function of the two model parameters (function `meshgrid` in the module `numpy`) (function `contourf` in the module `matplotlib`)
- **locating** your solution on this surface.

What are the properties of *this* error surface that make this minimization possible?

6 Bivariate Linear Regression

Choose the 2 best features (still using correlation as criterion) and **train** another linear regression. **Visualize** the results and compare the predicted target values with the actual target values. **Find** a formal criterion to show that linear regression is able to approximate the target values more accurately using two features than only one feature (with respect to training data) (tip: look above).

7 Principal Component Analysis

Instead of selecting two features, principal component analysis allows to obtain two linear combinations of the 10 features which concentrate most of the features' variance.
Before applying pca, you should not forget to **normalize** the data. Why is this is good practice?
Implement principal component analysis (do *not* use `PCA` in `sklearn`) and select the two principal components. Once again **perform** linear regression, using these two newly selected features. **Visualize** the results and compare the predicted target values with the actual target values.
Moreover, **compare** your new linear model with the linear models obtained previously. How can you explain your conclusions ?
Implement whitening and check if this helps your results. Any idea why this would be?

8 Multivariate Linear Regression

Perform a linear regression using a certain number of features. **Plot** the RMSE in function of the number of features. **Apply** this same exercise to PCA (=augment the number of selected components). What do you conclude?

9 Train-test Generalization

(if you have time) Load the second dataset (X,t). Split it in two equal parts at random ({X1, t1} and {X2,t2}). Build a multivariate regression model on the first dataset (X1, t1) and test it on (X1,t1). Draw a target vs prediction scatterplots and compute the rmse. Is it a good model? Let's see how it generalizes. Test your model on (X2, t2). Is it as good as the first one? How do you explain the difference between the two? What are the key concepts behind that?