TP1 Regression

UV SDATA - 2021 - C. Garnier - CERI SN

Before you start

- Read well the slides of the course.
- The objective of TP1 is to understand regression from simulated data using the formulas of the course and the `NumPy` package. Next in TP2, you will use the regression functions of `scikit-learn` and you will apply regression to real datasets.
- Be curious : Play with the parameters (for example the number of training data, the variance of the noise, ...), change the true underlying function, ... and try to understand.

1. Single linear regression

A hypermarket has 20 checkouts. We focus on the average waiting time for clients expressed in minutes, denoted y, versus the number of available checkouts, denoted x. The dataset of size $N = 7$ is given in the following table:

x	Number of available checkouts	3	4	5	6	8	10	12
y	Average waiting time	16	12	9.5	8	6	4.5	4

The objective is to perform a linear regression to predict the average waiting time for clients given the number of available checkouts.

Reminder: Suppose we have a sample $\{(x_i, y_i), 1 \leq i \leq N\}$ of independent realizations of 2 random variables X and Y.

An estimator of the mean is: $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$ *and* $\bar{y} = \frac{1}{N}\sum_{i=1}^{n} y_i$

A (biased) estimator of the variance is: $s_x^2 = \frac{1}{N}\sum_{i=1}^{n} x_i^2 - \bar{x}^2$ *and* $s_y^2 = \frac{1}{N}\sum_{i=1}^{N} y_i^2 - \bar{y}^2$

A (biased) estimator of the covariance of X and Y is given by: $s_{xy} = \frac{1}{N}\sum_{i=1}^{n} x_i y_i - \bar{x}\bar{y}$

A (biased) estimator of the correlation coefficient is given by: $r_{xy} = \frac{s_{xy}}{s_x s_y}$

1. **Data visualization:** Define the input x and the output y as `numpy` arrays and visualize the scatterplot corresponding to the dataset $\{(x_i, y_i)\}_{i=1}^{N}$.
2. **Data statistics estimation:** Estimate the means, the variances, the covariance and the correlation coefficient of x and y using the `numpy` functions `mean` and `var`. Are the number of available checkouts and the average waiting time correlated ?
3. **Model:** We assume a linear model between the input $x \in \mathbb{R}$ (input dimension $p = 1$) and the output $y \in \mathbb{R}$:
$$y = f(x, \boldsymbol{\beta}) = \beta_0 + \beta_1 x$$
where $\boldsymbol{\beta}$ is the vector of the regression coefficients $\boldsymbol{\beta} = [\beta_0, \beta_1]^\top \in \mathbb{R}^2$
4. **Training / Learning:** We want to estimate the coefficients β_0 and β_1 of the regression line (with equation: $y = \beta_0 + \beta_1 x$) from the training set $\{(x_i, y_i)\}_{i=1}^{N}$. For this, we look for the

coefficients which minimize the following quantity:

$$\mathcal{L}_{\boldsymbol{\beta}} = \frac{1}{2}\sum_{i=1}^{N}(y_i - f(x_i,\beta))^2 = \frac{1}{2}\sum_{i=1}^{N}(y_i - (\beta_0 + \beta_1 x_i))^2$$

What does $\mathcal{L}_{\boldsymbol{\beta}}$ represent ?

Now estimate the regression coefficients from the training data in two different ways:

- By cancelling the partial derivatives of $\mathcal{L}_{\boldsymbol{\beta}}$ for β_0 and β_1: show that the regression coefficients are estimated as: $\hat{\beta}_0 = \bar{y} - \hat{\beta}_1\bar{x}$ and $\hat{\beta}_1 = \frac{s_{xy}}{s_x^2}$. Calculate the coefficients $\hat{\beta}_0$ and $\hat{\beta}_1$.
- By using the generic formula of the course: $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$ where the matrix $\mathbf{X}$ and the column vector $\mathbf{y}$ are defined as:

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

  Check that you get the same results.

Visualize the regression line on the scatterplot.

5. **Test / Prediction:** Give a prediction of the average waiting time if the hypermarket has only 1 available checkout, 7 available checkouts, 20 available checkouts.

   What do you think about the linear model for this problem ? is it appropriate ?

2. Polynomial regression with different model complexities

1. **Training data generation:** Generate $N = 50$ training data $\{(x_i, y_i)\}_{i=1}^{N}$, with uniformly distributed inputs $x_i \sim \mathcal{U}([0,1])$ and noisy outputs $y_i = f(x_i) + \varepsilon_i$.

   - The true function is defined as: $f(x) = \sin(1 + x^2)$
   - The noise samples are i.i.d. and Gaussian: $\varepsilon_i \sim \mathcal{N}(0, \sigma_\varepsilon^2)$, with $\sigma_\varepsilon^2 = 0.002$.

   Display the scatterplot corresponding to the training data and the true function in the interval $[0,1]$ (for N regularly spaced values over the interval $[0,1]$).

2. **Model:** A polynomial dependence is assumed between the input and the output, ie y is assumed to depend not only on x, but also on x^2, x^3,...,x^m with m the polynomial degree also called the model complexity:

$$y = \boldsymbol{\beta}^\top\mathbf{x} = \beta_0 + \sum_{j=1}^{m}\beta_j x^j$$

   where $\boldsymbol{\beta} = [\beta_0, \beta_1, ..., \beta_m]^\top \in \mathbb{R}^{m+1}$ and $\mathbf{x} = [1, x, x^2, ..., x^m]^\top \in \mathbb{R}^{m+1}$.

   - Here we consider different polynomial degrees (or model complexities): $m = 1, 2, ..., 8$, ie all the polynomials up to order 8.

3. **Training / Learning:** Estimate the vector of the regression coefficients $\widehat{\boldsymbol{\beta}}^{(m)} \in \mathbb{R}^{m+1}$ for $m = 1, 2, ..., 8$ using the formula of the course: $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$ where the matrix $\mathbf{X}$ and the column vector $\mathbf{y}$ are defined as:

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^m \\ 1 & x_2 & x_2^2 & \dots & x_2^m \\ 1 & x_3 & x_3^2 & \dots & x_3^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & x_N^2 & \dots & x_N^m \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_N \end{bmatrix}$$

- Note that for each model complexity m, you have to build a different matrix $\mathbf{X}$.
- Obviously, you obtain a different vector $\widehat{\boldsymbol{\beta}}^{(m)}$ depending on the polynomial order m.

4. **Test / Prediction:** For each model complexity m, predict the outputs y^* for N new inputs x^* regularly spaced over the interval $[0, 1]$.

   Reminder: for a new input $x^* \in \mathbb{R}$, the output $y^* \in \mathbb{R}$ is predicted as:

$$\widehat{y}^* = \widehat{\boldsymbol{\beta}}^\top \mathbf{x}^* = \widehat{\beta}_0 + \sum_{j=1}^{m} \widehat{\beta}_j x^{*j}$$

   where $\mathbf{x}^* = [1, x^*, x^{*2}, ..., x^{*m}]^\top \in \mathbb{R}^{m+1}$.

   - Using all these predictions, plot the predicted function in the interval $[0, 1]$ for each model complexity m on the scatterplot. Compare to the true function.

5. **Prediction errors:** At this point, start a new file: for each model, add a *for* loop that will run the program several times. Consider a number of $RUNS = 200$ times.

   - The goal is to characterize how well the learning task works. At each run, you have to re-generate the training data and learn $\widehat{\boldsymbol{\beta}}^{(m)} \in \mathbb{R}^{m+1}$ from these training data.
   - For each run, compute the mean squared error (MSE) between the N predicted outputs and the N true outputs.
     Reminder: The MSE is the mean of the squares of errors: $MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i^* - f(x_i^*))^2$.
   - At the end, for each model, average the MSE over all runs.
   - Finally, plot the MSE versus the model complexity m.
   - According to the results, which model complexity m would you choose ?