Mondragon Unibertsitatea

Escuela Politécnica Superior

Regression

Outline

- Simple regression
  - Parametric. Simple Linear Regression
  - Non-parametric. K-Nearest Neighbour
- Multiple regression
  - Non-aditive
  - Non-linear
    - Polinomial regression
    - Splines regression
- Fitting procedures
  - Dimensionality Reduction
    - Principal Components Regression (PCR)
    - Partial Least Squares (PLS)
  - Shrinkage
    - Ridge regression
    - Lasso
    - ElasticNet
- Validation and model selection
  - Explained variance
  - Mean absolute, quadratic and quadratic logarithmic errors
  - Coefficient of determination

Simple linear regression

Mondragon Unibertsitatea
Escuela Politécnica Superior

$$Y = f(X) + \epsilon = a + bX + \epsilon$$

- A **linear relation** between the predictor X and the response Y is assumed.
- The **coefficients** to be optimized are a and b. The **error** would be ϵ.
- It is not efficient by itself, but it is the basis for many sophisticated models.

- Given pairs of values $(x_i, y_i), i = 1, \dots, N$ the coefficients a and b must be estimated in such a way that the predictions of the y_i from the x_i are as good as possible.
- For instance:

$$\hat{a}, \hat{b} = \arg min_{a,b} \sum_{i=1}^{N} (y_i - a - bx_i)^2$$

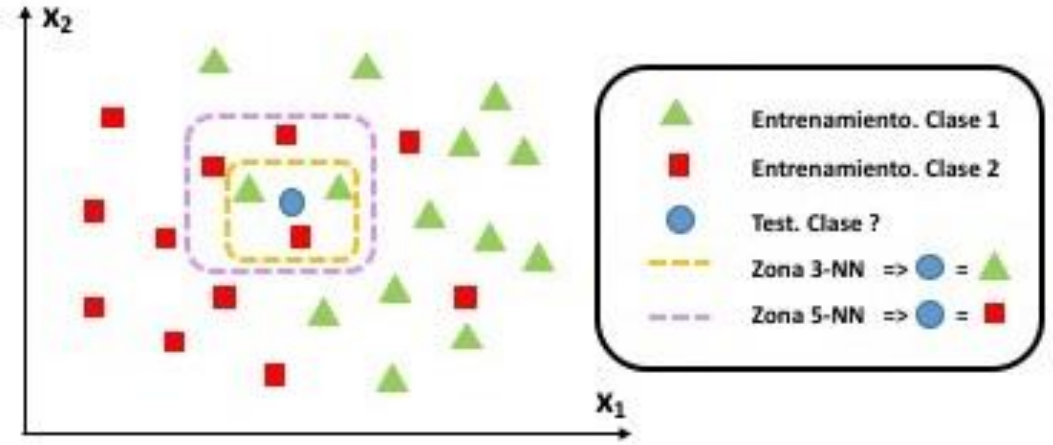
Therefore, the coefficients that minimize the sum of the quadratic error.

- Note: the minimum above is located at the same point $(\hat{a}, \hat{b})$ as the minimum mean square error, then it is equivalent to using that *score*.

K-Nearest Neighbors regression

Mondragon Unibertsitatea
Escuela Politécnica Superior

- Given a training set and a sample, for which we want to obtain a prediction:
  - Calculate all pairwise “distances” between the test samples and the train samples
  - Select the K smaller distances
  - Predict the class for the test sample by means of averaging among its K nearest neighbors



- Variants:
  - Use a weighted mean, which weights are inversely proportional to the distance
  - Without prefixing K, choose a threshold for defining the neighborhood (radius) and consider as neighbors all training samples that are not further than that radius
- **Prefixing K** is a tricky task to solve
- We must design an strategy for **breaking ties**

Multiple linear regression

Mondragon Unibertsitatea
Escuela Politécnica Superior

$$Y = f(\boldsymbol{X}) + \epsilon = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

- A **linear relation** between the predictors $X_1, \ldots, X_p$, and the response Y is assumed.
- The **coefficients** to be optimized are p+1 coefficients β_j. The **error** would be ϵ.
- It suffers in situations de co-linearity of the predictions.

- Given pairs of values $(x_i, y_i), i = 1, \ldots, N$ the coefficients β_j estimated in such a way that the predictions of the y_i from the x_i are as good as possible.

- For instance:

$$\hat{\boldsymbol{\beta}} = \arg min_{\boldsymbol{\beta}} \sum_{i=1}^{N} \left(y_i - \beta_0 + \beta_1 X_{1,i} + \cdots + \beta_p X_{p,i}\right)^2$$

  Therefore, the coefficients that minimize the sum of the quadratic error.

- Note: the minimum above is located at the same point $\hat{\boldsymbol{\beta}}$ as the minimum mean square error, then it is equivalent to using that *score*.

Linearity and additivity

Mondragon Unibertsitatea
Escuela Politécnica Superior

$$Y = f(\boldsymbol{X}) + \epsilon = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

- Linearity. The predictors contribute linearly in the prediction.
- Additivity. The effects over the prediction of the perturbations in a predictor are independent of the value of the rest of predictors.
- Avoiding linearity implies, for instance, include quadratic effects.
- Avoiding additivity implies introducing interaction terms between predictors. Moreover, it diminishes **interpretability** of the model. For example:

$$Y = f(\boldsymbol{X}) + \epsilon = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2 + \epsilon$$

- The effect of X_1 is no longer constant, as changes in X_1 modify the impact of X_1 over Y

$$Y = f(\boldsymbol{X}) + \epsilon = \beta_0 + (\beta_1 + \beta_3 X_2) X_1 + \beta_2 X_2 + \epsilon$$

Non-linear regression

Mondragon Unibertsitatea
Escuela Politécnica Superior

- We skip the need for linear relations
- Certain functions $b_k(\cdot)$ acting over $\boldsymbol{X}$ in a non-linear way are chosen

$$Y \sim \beta_0 + \beta_1 b_1(\boldsymbol{X}) + \cdots + \beta_k b_k(\boldsymbol{X})$$

- Notice that $\boldsymbol{X}$ could be a single predictor (simple) or several ones (multiple)
- In case of multiple, the functions would affect matrixally by columns, being even possible the consideration of the $b_k(\cdot)$ having p components, one per predictor

- For instance, simple quadratic regression, which is a particular case of polynomic regression, is given by

$$Y \sim \beta_0 + \beta_1 X + \cdots + \beta_2 X^2$$

- The base functions would be in this case: $b_1(x) = x$ y $b_2(x) = x^2$.

Polynomic regression

Mondragon Unibertsitatea
Escuela Politécnica Superior

- The regression function has the shape

$$Y \sim \beta_0 + \beta_1 X + \beta_2 X^2 + \cdots + \beta_d X^d$$

- Base functions would be in this case: $b_k(x) = x^k, k = 0, \ldots, d$.

- Polynomic curves (simple) or surfaces (multiple) of degree d are generated.

Splines

Mondragon Unibertsitatea
Escuela Politécnica Superior

- Instead of a polynomial of high degree for the whole prediction space, it is possible that we could be interested on using locally different polynomials of lower degrees in certain concrete regions

- Those regions are defined by mean of cut-points called *knots* that indicate the points where the model changes

- For instance, a cubic polynomial in two pieces with a knot in a point c would be:

$$Y = \begin{cases} \beta_{01} + \beta_{11}X + \beta_{21}X^2 + \beta_{31}X^3, si\ X < c \\ \beta_{02} + \beta_{12}X + \beta_{22}X^2 + \beta_{32}X^3, si\ X \geq c \end{cases}$$

Cubic spline regression

Mondragon Unibertsitatea
Escuela Politécnica Superior

$$Y \sim \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_{3+1} h(X, c_1) + \cdots + \beta_{3+k} h(X, c_k)$$

- where

$$h(X, c\,) = \begin{cases} (X - c)^3, si\ X > c \\ 0 \quad\quad , si\ X \geq c \end{cases}$$

- Defines piecewise polynomials of degree 3 that is continuous and with smooth curvature
- We could enforce certain conditions at the knots: continuity, or first order differentiability and/or second order continuous derivatives, etc.
- The total number of coefficients is: 1 (β_0) + 3 (degree) + k (knots)

Cubic spline regression

Mondragon Unibertsitatea
Escuela Politécnica Superior

- Cubic splines are the ones with lower order so that the discontinuity at the knots is not perceptible by human eye

- Splines vs polynomic. In polynomic regression we need really high degrees in many problems. This causes issues in low represented zones of the space as well as in extreme extrapolation areas

- Splines vs polynomic. The flexibility in splines is achieved by incrementing the number of knots, keeping the degree fixed

Fitting procedures

Mondragon Unibertsitatea
Escuela Politécnica Superior

- Fitting a multiple linear regression model

$$Y = f(\boldsymbol{X}) + \epsilon = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

is an optimization problem

$$(\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p) = \arg min_{\boldsymbol{\beta}} \sum_{i=1}^{N} e_i^2 \quad \rightarrow \quad \hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \cdots + \hat{\beta}_p X_p$$

- Instead of optimizing only the error, we could modify the objetive funtion in order to look for fitting alternatives.
- Why?
  - Getting better fitting, i.e. lower error
  - Improving interpretability, e.g. reducing the use of certain variables internally (embedded feature selection)

Fitting procedures

Mondragon Unibertsitatea
Escuela Politécnica Superior

- Available alternatives:
  - **Feature subset selection**. Identify those variables we consider more related with the response variable
  - **Dimensionality reduction**. Projecting the p predictors over a M-dimensional space, with $M < p$, using the projections as the new predictors (feature extraction). An example would be the application of PCA, previously to the multiple linear regression. This algorithm is called *principal components regression* (PCR).
  - **Regularization (or *shrinkage*)**. Fitting the model with all predictors, forcing, during the optimization, that the absolute values of the coefficients become small, even null. For instance, Lasso.

Principal components regression

Mondragon Unibertsitatea
Escuela Politécnica Superior

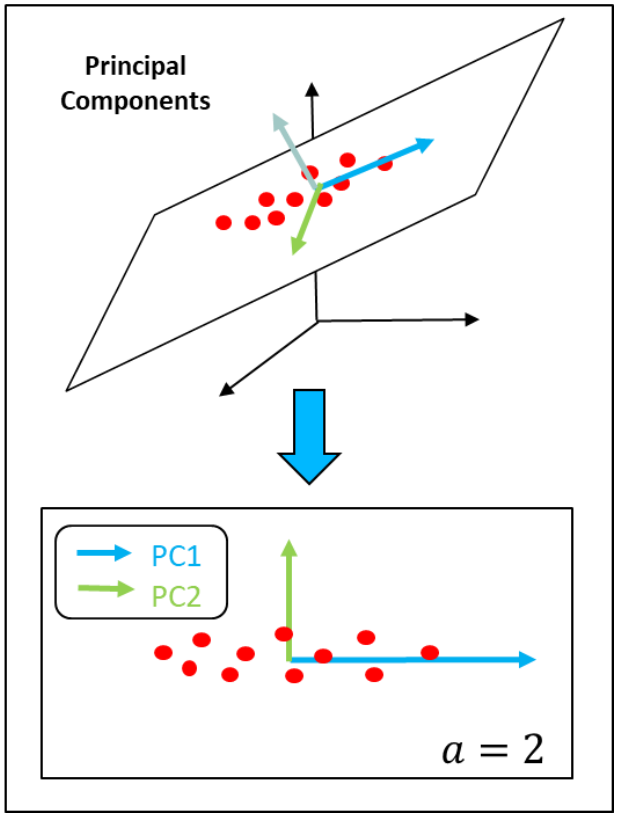
- We apply PCA, fixing the right number, a, of PCs
- We project on the a-dimensional space, using the scores as new predictors
- We apply MLR using them

Principal component analysis

- Based on the decomposition given by:

$$X = T_a P_a^T + E_a \qquad a \equiv \#PCs$$

- The PCs (LVs) are selected by maximizing scores variance
- $P \equiv$ Loadings => Coefficients of the linear combination that defines the PCs
- $T \equiv$ Scores => Coordinates of the data in the new axes
- $E \equiv$ Residuals (null if $a = N$)



Partial least squares

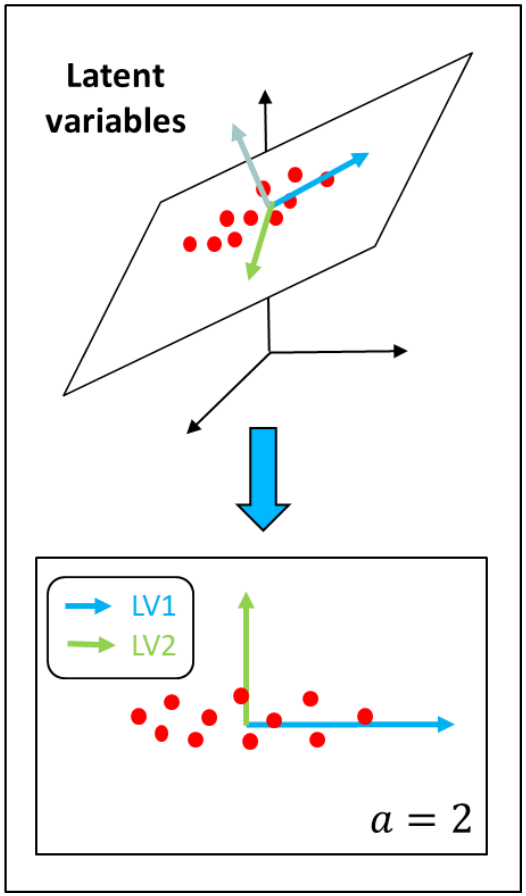
- We apply PLS, chosing the right number, a, of latent variables (LVs)
- The prediction now is obtained directly, without the need of MLR, using

$$Y = T_a Q_a$$

Partial least squares

$$X = T_a P_a^T + E_a$$
$$Y = T_a Q_a + F_a$$
$$a \equiv \#LVs$$

- The LVs are selected by maximizing scores variance and correlation with the target
- $P, Q \equiv$ Loadings
- $T \equiv$ Scores
- $E, F \equiv$ Residuals
- Supervised and linear



Lasso & Elastic Net

Mondragon Unibertsitatea
Escuela Politécnica Superior

- We modify the objetive function of the multiple linear regression model

$$Y = f(\boldsymbol{X}) + \epsilon = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon\,,$$

  appearing in the optimization problem

$$(\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p) = \arg min_{\boldsymbol{\beta}} \sum_{i=1}^{N} e_i^2 \;\rightarrow\; \hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \cdots + \hat{\beta}_p X_p.$$

- The new objective function is not only considering the quadratic errors e_i^2, but it also includes a penalization term on the absolute values of the regression coefficients

$$\hat{\boldsymbol{\beta}} = \arg min_{\boldsymbol{\beta}} \sum_{i=1}^{N} e_i^2 + \alpha \sum_{j=1}^{p} |\beta_j| \rightarrow \hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \cdots + \hat{\beta}_p X_p.$$

  - The penalization term uses the l_1 norm of the coefficients vector. A similar penalization, but employing the l_2 norm would be the case of ***ridge regression***.
  - The value $\boldsymbol{\alpha}$ controls the importance of the penalization. High values would force some coefficients to become null.
  - A penalization term combining l_1 and l_2 norms by means of a convex combination is called **Elastic Net**

Mondragon Unibertsitatea

Escuela Politécnica Superior

Validation and model selection

Explained variance

Mondragon Unibertsitatea
Escuela Politécnica Superior

- Let y and $\hat{y}$ be real and predicted values respectively
- The **explained variance i**s:

$$ExplainedVariance(y, \hat{y}) = 1 - \frac{Var(y - \hat{y})}{Var(y)}$$

- The best possible value, without error, would be 1 (maximum)

- In *Python*, we would use **explained_variance_score** function, available in the part of scikit-learn concerning *scores* (metrics).
- Concretely, it is imported with: **from sklearn.metrics import explained_variance_score**
- It is calculated using the prediction by: **explained_variance_score(y_true, y_pred)**

Mean absolute error

Mondragon Unibertsitatea
Escuela Politécnica Superior

- Let y and $\hat{y}$ be real and predicted values respectively
- The **mean absolute error i**s:

$$MAE(y, \hat{y}) = \frac{1}{N} \sum_{i=0}^{N-1} |y_i - \hat{y}_i|$$

- The best possible value, without error, would be 0 (minimum)

- In *Python*, we would use **mean_absolute_error** function, available in the part of scikit-learn concerning *scores* (metrics).
- Concretely, it is imported with: **from sklearn.metrics import mean_absolute_error**
- It is calculated using the prediction by: **mean_absolute_error (y_true, y_pred)**

Mean squared error

Mondragon Unibertsitatea
Escuela Politécnica Superior

- Let y and $\hat{y}$ be real and predicted values respectively
- The **mean squared error i**s:

$$MSE(y, \hat{y}) = \frac{1}{N} \sum_{i=0}^{N-1} (y_i - \hat{y}_i)^2$$

- The best possible value, without error, would be 0 (minimum)
- It is a quadratic measure. If we want an error value in a similar range as MAE, we must take its square root (denoted by RMSE, from *root mean squared error*).
- Regarding its minimization, it is irrelevant as the maxima and minima of a function $f(x)$ are located at the same points as the ones of $g(x) = \sqrt{f(x)}$

- In *Python*, we would use **mean_squared_error** function, available in the part of scikit-learn concerning *scores* (metrics).
- Concretely, it is imported with: **from sklearn.metrics import mean_squared_error**
- It is calculated using the prediction by: **mean_squared_error(y_true, y_pred)**

Mean squared logarithmic error

Mondragon Unibertsitatea
Escuela Politécnica Superior

- Let y and $\hat{y}$ be real and predicted values respectively
- The **mean squared logarithmic error** is:

$$MLSE(y, \hat{y}) = \frac{1}{N}\sum_{i=0}^{N-1}(ln(1+y_i) - ln(1+\hat{y}_i))^2$$

- The best possible value, without error, would be 0 (minimum)
- Due to the shape of the logarithmic function, errors by default are penalized harder that errors by excess

- In *Python*, we would use **mean_squared_log_error** function, available in the part of scikit-learn concerning *scores* (metrics).
- Concretely, it is imported with: **from sklearn.metrics import mean_squared_log_error**
- It is calculated using the prediction by: **mean_squared_log_error(y_true, y_pred)**

Coefficient of determination R^2

Mondragon Unibertsitatea
Escuela Politécnica Superior

- Let y and $\hat{y}$ be real and predicted values respectively, and let $\bar{y}$ be the average of real values
- The **coefficient of determination** is:

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{N-1}(y_i - \bar{y}_i)^2}$$

- The best possible value, without error, would be 1 (maximum)
- It could take values arbitrarily negative
- In case of predicting always the mean, we would obtain a value 0

- In *Python*, we would use **r2_score** function, available in the part of scikit-learn concerning *scores* (metrics).
- Concretely, it is imported with: **from sklearn.metrics import r2_score**
- It is calculated using the prediction by: **r2_score(y_true, y_pred)**

Mondragon Unibertsitatea
Escuela Politécnica Superior

Eskerrik asko
Muchas gracias
Thank you

Carlos Cernuda
ccernuda@mondragon.edu

MGEP
Goiru, 2
20500 Arrasate – Mondragon
Tlf. 662420414