Institut National des Télécommunications et des Technologies de l'Information et de la Communication INT-TIC d'Oran Algérie

Analyse de Données

3ème année ingénieur

Fiche de TP N° 5

Méthodes Inductives

Régression Linéaire Multiple

Nous avons vu que l'analyse linéaire discriminante permet, sur la base d'un ensemble de variables quantitatives, de discriminer au mieux entre les individus appartenant à différentes classes (modalités différentes de la variable qualitative). Ainsi, il est possible de déterminer (de prédire) la classe d'un nouvel individu à partir des valeurs des variables quantitatives. Ceci peut être vu comme sorte de une mise en relation entre l'ensemble des variables quantitatives d'une part, et la variable qualitative d'autre part.

La **régression linéaire multiple** permet, quant à elle, de mettre en relation un ensemble (d'où le mot « multiple ») de variables quantitatives d'une part, et une variable ***quantitative*** d'autre part. Cette dernière est appelée ***variable expliquée*** ; les autres variables quantitatives étant appelées ***variables explicatives***.

La relation entre les variables explicatives et la variable expliquée est donnée par la relation linéaire suivante :

$$Y = X\beta + \epsilon$$

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad X = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix} \quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix} \quad \epsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

Où :

- y_i : la valeur de la variable expliquée pour l'individu i.
- x_{ij} : la valeur de la variable explicative j pour l'individu i.
- β : vecteur des paramètres du modèle (à estimer).
- ϵ : erreur générée par le modèle (par la régression).

n et p sont respectivement le nombre d'individus et le nombre de variables explicatives.

Il s'agit donc d'estimer les paramètres β_j qui minimisent l'erreur de la régression (erreur des moindres carrés). Sous certaines conditions ($n > p + 1$ et $rang(X) = p + 1$), l'estimation de β est donnée par la formule suivante :

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

Nous allons appliquer ce modèle sur un jeu de données contenant des informations sur les pourcentages de personnes atteintes de maladies cardiaques dans différentes régions. Le fichier de données `heart.data.csv` contient les 3 variables quantitatives suivantes, pour un échantillon de 498 individus (498 régions) :

`biking`	Pourcentage de personnes qui utilisent le vélo pour aller au travail
`smoking`	Pourcentage de personnes qui fument
`heart.disease`	Pourcentage de personnes atteintes de maladies cardiaques

Dans ce cas de figure, nous recherchons à mettre en relation les variables `biking` et `smoking` (les variables explicatives) d'une part, et la variable `heart.disease` (la variable expliquée) d'autre part.

Les modules utilisés sont les mêmes utilisés dans les TPs précédents : `numpy`, `pandas` et `matplotlib`.

```
>>> import numpy as np
>>> import pandas as pd
>>> import matplotlib.pyplot as plt
>>> import os
```

Importation et préparation des données

```
>>> os.chdir('C:\\...\\...\\DataExamples')
>>> data = pd.read_csv('heart.data.csv', sep=',')

>>> data.head()
   Unnamed: 0     biking    smoking  heart.disease
0           1  30.801246  10.896608      11.769423
1           2  65.129215   2.219563       2.854081
2           3   1.959665  17.588331      17.177803
3           4  44.800196   2.802559       6.816647
4           5  69.428454  15.974505       4.062224

>>> Y = np.asarray(data['heart.disease'])     # Extraction de la variable expliquée

                                              # Extraction des variables explicatives
>>> X = np.asarray([data['biking'],data['smoking']]).T
>>> X[:5,:]
array([[30.80124571, 10.89660802],
       [65.12921517,  2.21956318],
       [ 1.95966453, 17.58833051],
       [44.80019562,  2.80255888],
       [69.42845368, 15.9745046 ]]

>>> n = data.shape[0]                         # Nombre d'individus
>>> n
498
>>> p = data.shape[1]                         # Nombre de variables (explicatives)
>>> p
2

>>> X = np.insert(X, 0, np.ones(n), axis=1)   # Ajout du vecteur colonnes (contenant
>>> X[:5,:]                                   # des 1 partout) à la 1ère position
array([[ 1.        , 30.80124571, 10.89660802],
       [ 1.        , 65.12921517,  2.21956318],
       [ 1.        ,  1.95966453, 17.58833051],
       [ 1.        , 44.80019562,  2.80255888],
       [ 1.        , 69.42845368, 15.9745046 ]])
```

Calcul des paramètres du modèle (vecteur β) et utilisation

Après avoir préparé le vecteur Y et la matrice X, il est à présent possible de calculer l'estimation du vecteur β. Mais il faut s'assurer avant que la matrice X^TX est inversible (déterminant différent de 0), et que les deux variables explicatives ne représentent pas la même information (en d'autres termes, que le rang de la matrice est égal à p=2) :

```
>>> np.linalg.matrix_rank(X[:,1:])      # Le rang de la matrice des variables
2                                       # explicatives est bien égal à p=2

>>> np.linalg.det(X.T.dot(X))           # Le déterminant est bien différent de 0
3900114699799.9175
```

Nous calculons l'estimation du vecteur β :

```
>>> beta = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(Y)
>>> beta
array([14.98465799, -0.20013305,  0.17833391])
```

Autrement dit :

`heart.disease` = 14.985 – 0.2 * `biking` + 0.178 * `smoking`

$\hat{Y}$ le vecteur estimé (ou l'approximation par la régression) de Y se calcule ainsi :

$$\hat{Y} = X\hat{\beta}$$

```
>>> Ye = X.dot(beta)                    # Ye est le vecteur estimé de Y
```

Supposons que nous voulions estimer le pourcentage de personnes atteintes de maladies cardiaques d'une nouvelle région dont on connaît seulement le pourcentage de personnes qui fument (25% par exemple) et le pourcentage de personnes utilisant le vélo pour aller au travail (2% par exemple). L'estimation se fait comme suit :

```
>>> np.dot([1, 2, 25], beta)
19.042739729263246
```

Le pourcentage de personnes atteintes de maladies cardiaques de cette région serait donc de 19.04%.

Coefficient de détermination

Le coefficient de détermination permet de mesurer la qualité d'ajustement du modèle (la régression) aux données initiales. Il se repose sur le calcul des statistiques suivantes :

$SCT = \sum_{i=1}^{n}(y_i - \bar{y})^2$ — Somme des Carrés Totaux
Traduit la variabilité totale.

$SCE = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$ — Somme des Carrés Expliqués
Traduit la variabilité expliquée par le modèle.

$SCR = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ — Sommes des Carrés Résiduels
Traduit la variabilité non expliqué par le modèle

Le coefficient de détermination R^2 est donné par :

$$R^2 = \frac{SCE}{SCT}$$

```
>>> SCE = pow(Ye-Y.mean(),2).sum(
>>> SCE
10176.571092319944
>>>
>>> SCR = pow(Ye-Y,2).sum()
>>> SCR
211.74025108752582
>>>
>>> SCT = pow(Y-Y.mean(),2).sum()
>>> SCT
10388.311343407451
>>>
>>> R2 = SCE/SCT
>>> R2
0.9796174523377296
```

Nous avons un coefficient proche de 1 (donc SCE et proche de SCT), ce qui veut dire la variabilité expliquée par la régression (le modèle) est proche de la variabilité totale. On en déduit que la régression s'ajuste bien aux données initiales.

Test de significativité du modèle

L'étude de la régression linéaire est souvent complétée par le test d'hypothèse suivant :

- $H_0 : \beta_1 = \cdots = \beta_p = 0$ (indépendance linéaire entre Y et X)
- $H_1 : \exists j \,/\, \beta \neq 0$

La statistique F suivante suit une loi de Fisher à $(p, n-p-1)$ degrés de liberté:

$$F = \frac{SCE/p}{SCR/(n-p-1)}$$

```
>>> import scipy.stats as stat
>>>
>>> F = (SCE/p)/(SCR/(n-p-1))
>>> F
11895.241138200245
>>>
>>> p_value = 1 - stat.f.cdf(F, p, n-p-1)
>>> p_value
1.1102230246251565e-16
>>>
>>> if p_value <= 0.05:
...  print ('Rejet de H0 : Dépendance linéaire')
... else:
...  print ('Pas de rejet de H0 : Indépendance linéaire')
...
Rejet de H0 : Dépendance linéaire
```

R. HACHEMI