REGRESSION LINEAIRE UNIVARIEE

INTRODUCTION

Régression linéaire univariée... Sous ce nom un peu barbare se cache en réalité un algorithme d'apprentissage supervisé très basique qui vise à trouver la meilleure droite possible à l'aide d'une seule variable explicative (on parle aussi d'un seul degré de liberté). Du fait de cette unique variable explicative, on parle de modèle univarié, par opposition aux modèles multivariés, qui font appel à plusieurs variables.

L'utilité de ce chapitre est plus pédagogique que scientifique, car dans les faits, vous utiliserez rarement des modèles univariés. Toutefois, leur simplicité permet d'introduire des notions qui peuvent devenir complexes lorsque l'on se place dans le cadre général.

Pour passer des données brutes au meilleur modèle de régression linéaire univariée, trois étapes sont nécessaires :

- La définition d'une fonction hypothèse
- La construction d'une fonction de coût
- La minimisation de cette fonction de coût

I. LA DEFNITION D'UNE FONCTION HYPOTHESE

Dans le cas de la régression linéaire univariée, on prend une hypothèse simplificatrice très forte : le modèle dépend d'une unique variable explicative, nommée X. Cette variable est un entrant de notre problème, une donnée que nous connaissons. Par exemple, le nombre de pièces d'un appartement, une pression atmosphérique, le revenu de vos clients, ou n'importe quelle grandeur qui vous paraît influencer une cible, qu'on nommera Y et qui, pour continuer nos exemples, serait respectivement, le prix d'un appartement, le nombre de millilitres de pluie tombés, ou encore le montant du panier moyen d'achat de vos clients.

On cherche alors à trouver la meilleure fonction hypothèse, qu'on nommera h et qui aura pour rôle d'approximer les valeurs de sortie Y :

$$\textit{valeur d'entrée } x \xrightarrow{\textit{hypothèse } h} \textit{valeur de sortie } y$$

Dans le cas de la régression linéaire à une variable, la **fonction hypothèse *h*** sera de la forme :

$$h(X) = \theta_0 + \theta_1 X$$

Notre problème revient donc à trouver le meilleur couple (θ_1,θ_2) tel que $h(x)$ soit « proche » de Y pour les couples (x, y) de notre base de données, que l'on peut commencer à nommer base d'apprentissage puisque les données dont vous disposez serviront à entraîner la fonction hypothèse *h*.

La notion de proximité qui vient d'être introduite incite assez naturellement à calculer une fonction d'erreur.

En effet, pour chaque point xi, la fonction hypothèse associe une valeur définie par $h(xi)$ qui est plus ou moins proche de yi.

On définit ainsi « **l'erreur unitaire** » pour xi par :

$$\sum_{i=1}^{m} \left(h(x_i) - y_i\right)^2$$

N.B : on élève au carré pour s'assurer que la contribution de l'erreur viendra bien pénaliser une fonction de coût.

La **fonction de coût** est alors définie en normant cette somme par le nombre m de points dans la base d'apprentissage.

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h(x_i) - y_i\right)^2$$

II. MINIMISER LA FONCTION DE COUT

Ainsi, trouver les meilleurs paramètres (θ_1,θ_2) de h – et donc la meilleure droite pour modéliser notre problème – équivaut exactement à trouver le minimum de la fonction J.

Une façon de trouver le meilleur couple (θ_1,θ_2) est la descente de gradient , méthode itérative dont le principe est assez intuitif : que ferait une balle lâchée assez haut dans notre bol?

Elle prendrait, à chaque instant, la meilleure pente jusqu'au point bas du bol.

Comprendre la descente de gradient en 3 étapes

L'algorithme de Gradient Descent

N.B : Un data scientist n'aura jamais à coder sa propre descente de gradient car tous les outils de ML qui utilisent cette méthode d'optimisation rendent transparente cette étape. En revanche, il lui appartiendra souvent de choisir les méta-paramètres comme α.

Notions fondamentales de la régression linéaire

III. IMPLEMENTATIONS DE LA REGRESSION LINEAIRE UNIVARIEE

1. AVEC NUMPY

les équations de Régression Linéaire écrites sous forme de matrices

Comment développer un programme de régression linéaire avec Numpy ?

2. SCIKIT-LEARN

LinearRegression - Documentation officielle

Exemple de la documentation officielle

3. STATSMODELS

Regression - Documentation officielle

Exemple pas à pas - page 1 à 7

4. SCIPY

linregress - Documentation officielle

Exemple pas à pas

IV. EVALUATION DE LA QUALITE DE LA REGRESSION

Un des plus gros inconvénients de la méthode de régression linéaire est qu'elle marche toujours ! En effet le calcul effectué par l'utilisateur des coefficients $\hat{a}$ et $\hat{b}$ de la droite de régression ne permet pas de juger si le modèle est acceptable ou non.

5. COEFFICIENT DE CORRELATION LINEAIRE

Pour mesurer la qualité de l'approximation d'un nuage de point par rapport à sa droite des moindres carrés il faut calculer le **coefficient de corrélation linéaire** :

$$\rho(X,Y) = Cov(X,Y)/\sqrt{var(X)var(Y)}$$

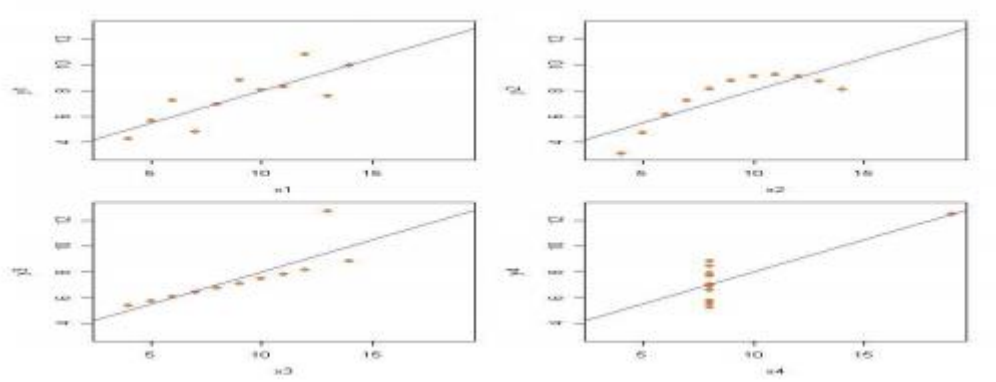
Il est à noter que ce coefficient est compris entre -1 et 1 est une mesure de dispersion du nuage :

- $\rho = 1$ si les points du nuage sont exactement alignés sur une droite de pente positive
- $\rho = -1$ si les points de nuage sont exactement alignés sur une droite de pente négative

Généralement, on considère l'approximation d'un nuage de points par sa droite de moindres carrés est de bonne qualité lorsqu'il est proche de 1 ou de -1 et de médiocre qualité lorsqu'il est proche de 0.

Attention : Il ne faut jamais généraliser et il faut toujours avoir un avis critique.

Voici quatre jeux de données choisis de manière à définir la même droite de régression, et avec le même coefficient de corrélation linéaire ρ. De gauche à droite et de haut en bas, le premier jeu de donnée est, au mieux, très bruité mais on peut douter que les données soient liées par une relation affine. Le second jeu correspond assez clairement à une relation quadratique : c'est plutôt une courbe $y = ax^2 + bx + c$ qu'il conviendrait d'ajuster. Dans le troisième jeu tous les points sauf un semblent alignés. Il y a visiblement un "point aberrant" dont il faudrait vérifier la provenance (ou la saisie dans le logiciel !) ; la situation est semblable pour le dernier échantillon.



CONCLUSION : la régression linéaire donne (presque) toujours une droite, mais il convient de regarder le résultat pour débusquer les situations par trop absurdes.

Pratiquement, les logiciels calculent, non pas le coefficient de corrélation linéaire ρ, mais son carré, appelé **R-deux** et noté **$R^2 = \rho^2$**

6. CALCUL DE R^2

Le coefficient R^2 peut être calculer avec plusieurs librairies python :

SKLEARN.METRICS

Fonction r2_score

STATSMODELS

statsmodels.regression.linear_model.OLS Results

SCIPY

scipy.stats.linregress

V. MINI BRIEF PROJET

Pour ce mini brief projet, nous allons utiliser le dataset Boston Housing data de sklearn.datasets qui comprend le prix des maisons dans les différentes zones de Boston.

L'objectif de ce mini projet est de prédire le prix des maisons (target) en utilisant les autres données (features) en créant un modèle de régression linéaire.

Pour cela, vous pouvez suivre la méthodologie méthodologie CRISP-DM proposée par IBM.

1. IMPORTATION DES DONNEES

- Importer le dataset à partir de sklearn.datasets
- Analyser les variables du dataset en affichant sa description

2. PREPARATION DES DONNEES

- Transformer le jeu de données en data frame
- Supprimer les valeurs nulles, s'il en existe.
- Diviser votre dataset en deux jeux de données : un jeu de données pour l'apprentissage et un jeu de données pour le test.

3. EXPLORATION DES DONNEES

- Choisir les variables explicatives (features) en menant une analyse de corrélation.

4. CREATION DU MODELE DE REGRESSION LINEAIRE

- Créer le modèle de régression linéaire
- Entrainer votre modèle

5. TEST DU MODULE

- Tester votre modèle

6. EVALUATION DU MODELE DE REGRESSION LINEAIRE

- Calculer l'erreur quadratique moyenne (erreur d'estimation)
- Calculer R^2
- Conclure

POUR ALLER PLUS LOIN ...

Test de significativité du lien linéaire entre deux variables avec R