

Cahier thématique

Regression lineaire processus complet

**Manipulation 1**: Utilisation d’un algorithme de régression linéaire dans un contexte de données incomplètes

**Préparé par :** Hafed Benteftifa  
  
© Hafed Benteftifa et Nesrine Zemirli 2015-2016

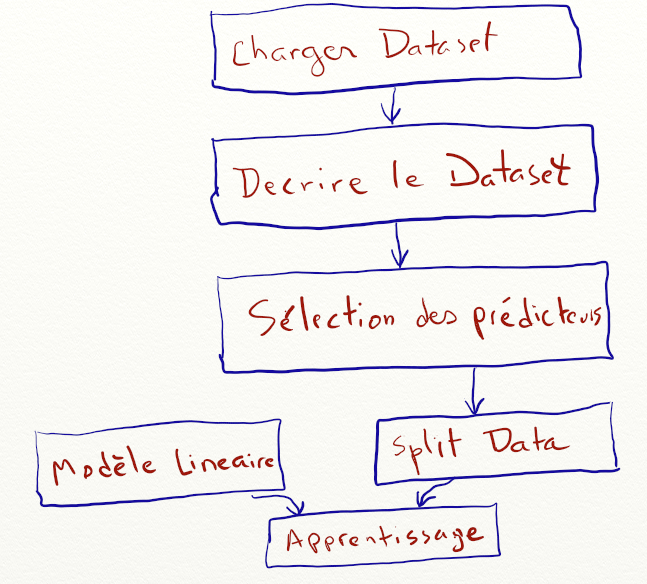
Ce document ne peut être utilisé dans le cadre d’une formation, publication papier, site internet ou tout support sans mon accord express.

Aucune reproduction, même partielle, ne peut être faite de ce document et de l'ensemble de son contenu : textes, images, etc. sans mon autorisation express. Pour toutes informations, communiquer avec moi sur [info@degenio.com](mailto:info@degenio.com).

|  |  |  |
| --- | --- | --- |
| **Date** | **Version** | **Changement** |
| 20 Mai 2016 | 1.0 | Version initiale |
|  |  |  |
|  |  |  |

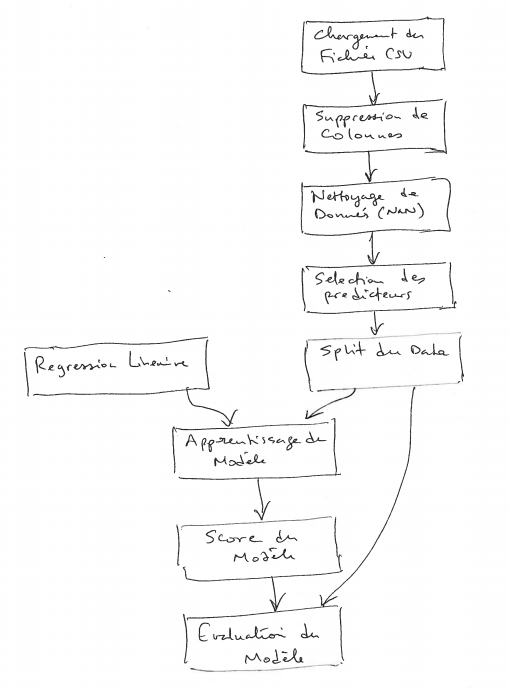
**Démarche**

Le processus d'apprentissage que l'on va développer avec le modèle de régression linéaire est décrit dans le diagramme suivant. Il faut noter que ce processus est assez générique pour être repris pour les autres modèles de régression et de classification.

****

Les modules python utilisés dans cette pratique sont pandas, scikit-learn, numpy et matplotlib.

Le processus détaillé est décrit comme suit:



**Manipulation 1**: Utilisation d’un algorithme de régression linéaire dans un contexte de données incomplètes

**Objectif**

Prédire le prix de vente d'une voiture comme une fonction des attributs de la voiture par la mise en place et l’exploitation de l’algorithme de régression linéaire. On utilisera aussi des étapes de pré-traitement et sélection appropriée des features et du data

**Préliminaire**

* Anaconda ou pycharm est disponible.

**Approche**

1. On utilisera le dataset connu comme le Automotive-price dataset. Celui-ci contient 506 instances. Les features-prédicteurs potentiels sont au nombre de 14 dont la valeur mediane ou prix qui sera la classe cible. Les attributs sont des réels.

Les étapes que l'on va suivre sont les suivantes

* Créer le modele
  + Étape 1: obtenir le data
  + Étape 2: faire un pré-traitement des données
  + Étape 3: Définir les features
* Faire l'apprentissage du modele
  + Étape 4: choisir un algorithme ML et procéder à l'apprentissage
* Évaluer le Score et tester le modele
  + Étape 5: Passer au test et faire la prédiction

**Démarche**

Étape 1: obtenir le data

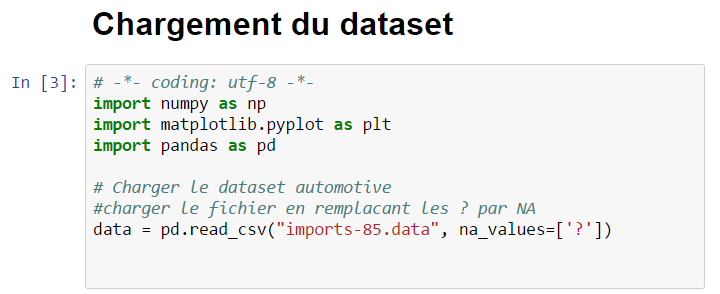
1. Importer le dataset concernant les automobiles

Les attributs disponibles sont les suivants: (https://archive.ics.uci.edu/ml/datasets/Automobile)

Attribute: Attribute Range   
  
1. symboling: -3, -2, -1, 0, 1, 2, 3.   
2. normalized-losses: continuous from 65 to 256.   
3. make:   
alfa-romero, audi, bmw, chevrolet, dodge, honda,   
isuzu, jaguar, mazda, mercedes-benz, mercury,   
mitsubishi, nissan, peugot, plymouth, porsche,   
renault, saab, subaru, toyota, volkswagen, volvo   
  
4. fuel-type: diesel, gas.   
5. aspiration: std, turbo.   
6. num-of-doors: four, two.   
7. body-style: hardtop, wagon, sedan, hatchback, convertible.   
8. drive-wheels: 4wd, fwd, rwd.   
9. engine-location: front, rear.   
10. wheel-base: continuous from 86.6 120.9.   
11. length: continuous from 141.1 to 208.1.   
12. width: continuous from 60.3 to 72.3.   
13. height: continuous from 47.8 to 59.8.   
14. curb-weight: continuous from 1488 to 4066.   
15. engine-type: dohc, dohcv, l, ohc, ohcf, ohcv, rotor.   
16. num-of-cylinders: eight, five, four, six, three, twelve, two.   
17. engine-size: continuous from 61 to 326.   
18. fuel-system: 1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi.   
19. bore: continuous from 2.54 to 3.94.   
20. stroke: continuous from 2.07 to 4.17.   
21. compression-ratio: continuous from 7 to 23.   
22. horsepower: continuous from 48 to 288.   
23. peak-rpm: continuous from 4150 to 6600.   
24. city-mpg: continuous from 13 to 49.   
25. highway-mpg: continuous from 16 to 54.   
26. price: continuous from 5118 to 45400.

Certains des prédicteurs seront utilisés alors que d'autres (categories et avec valeurs manquantes) ne le seront pas.

1. On procède à la lecteur du fichier csv en utilisant le module pandas.



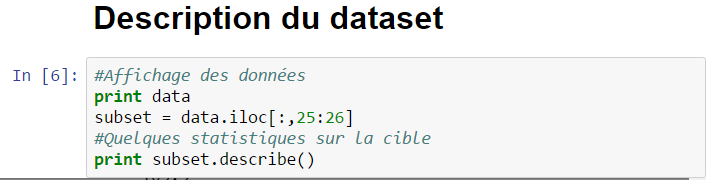
L'objet data est un dataframe qui contient le dataset. Il faut noter qu'à ce niveau on a un header au niveau du fichier csv. De ce fait, on n'a pas eu besoin d'utiliser le paramètre sur les colonnes.

1. On procède maintenant à une exploration sommaire du dataframe. En premier, on determine le shape, ce qui nous donne les valeurs suivantes.

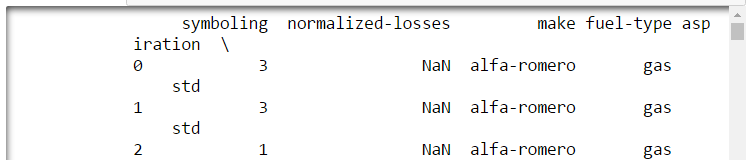


On constate que l'on a bien 26 colonnes dont la dernière colonne (price) qui va être la colonne cible.

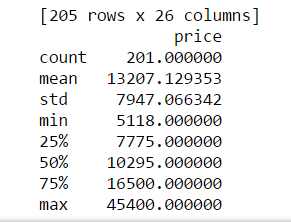
1. On obtient une description des données dans le dataframe ainsi que quelques statistiques sur la colonne price, soit:



ce qui donne en partie:

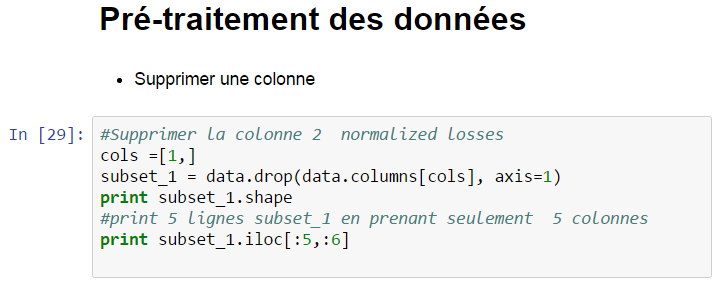


et les statistiques sur la colonne cible sont les suivantes

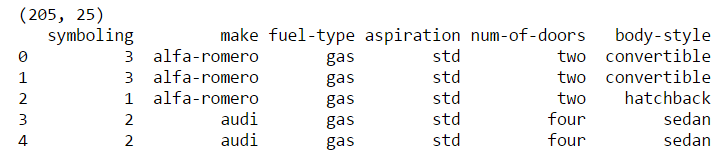


**Étape 2**: faire un pré-traitement des données

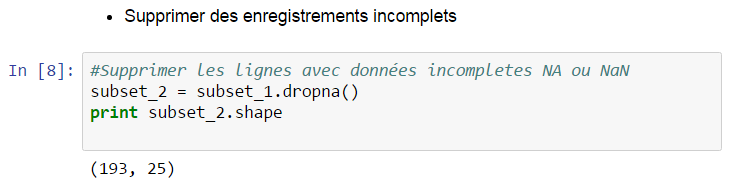
1. Dans le cas présent, on va supprimer une colonne non nécessaire et des enregistrements qui contiennent des données manquante.



ce qui donne en sortie en partie:



1. On utilise la fonction dropna() pour supprimer les enregistrements qui contiennent NaN,



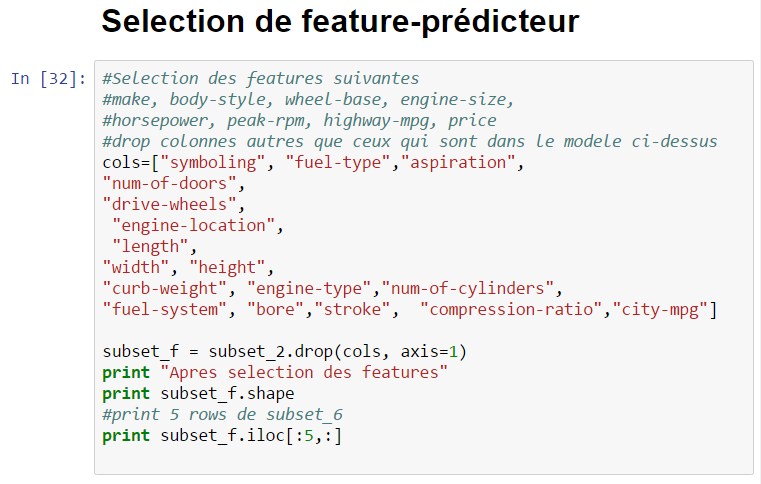
**Étape 3**: Définir les features

1. Sélection des features ou predicteurs

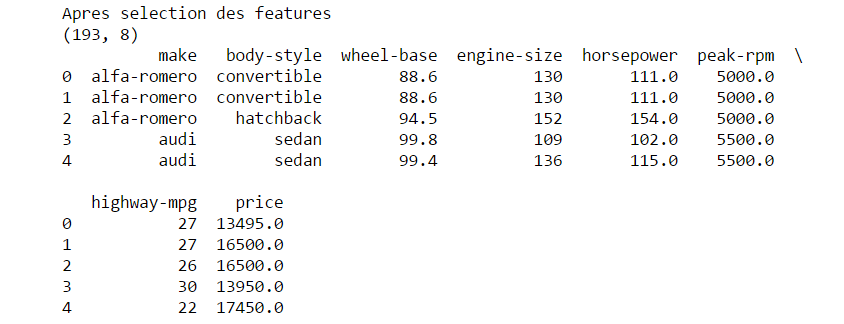
Dans le cas présent, on choisit les prédicteurs suivant :

make, body-style, wheel-base, engine-size, horsepower, peak-rpm, highway-mpg, price

Sur cette base, on modifie subset\_2 pour ne garder que les colonnes indiquées



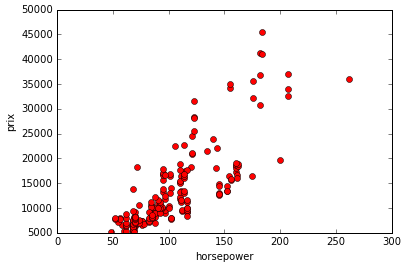
ce qui donne après sélection :



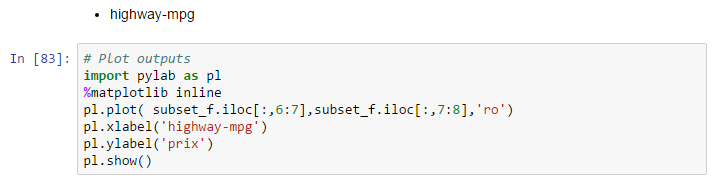
1. On procède à une exploration sommaire des données.

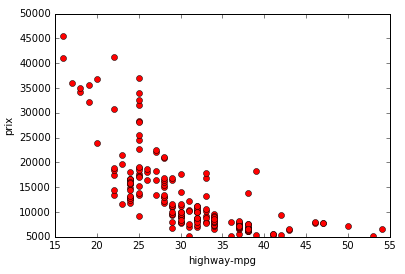


Ce qui donne le graphe suivant:



On répète la même opération pour le prédicteur highway-mpg, ce qui donne:



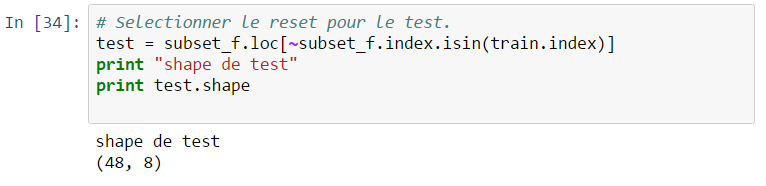


* Faire l'apprentissage du modèle
  + Étape 4: choisir un algorithme ML et procéder à l'apprentissage

On sépare en training et test set avec pourcentage de 75%. En premier, on commence par l'ensemble de training



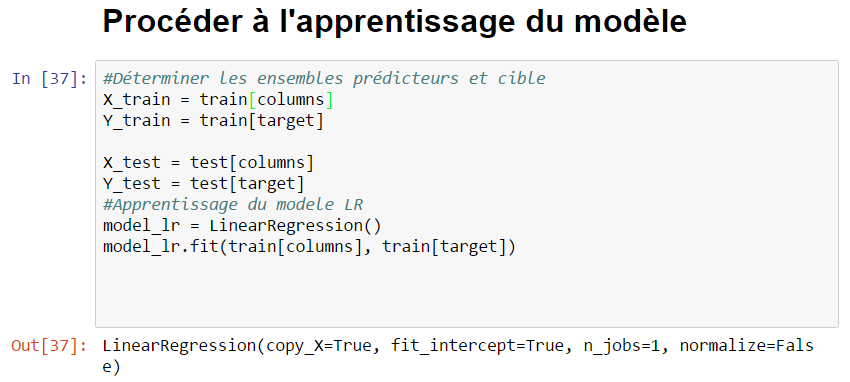
Finalement, on obtient l'ensemble de test comme suit:



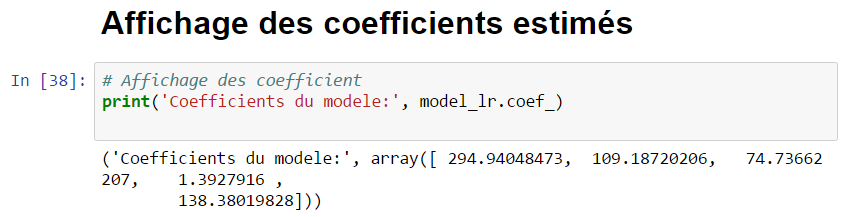
1. On choisit le modèle Linear regression pour la régression.



1. On procède à l'apprentissage du modèle avec la fonction fit()

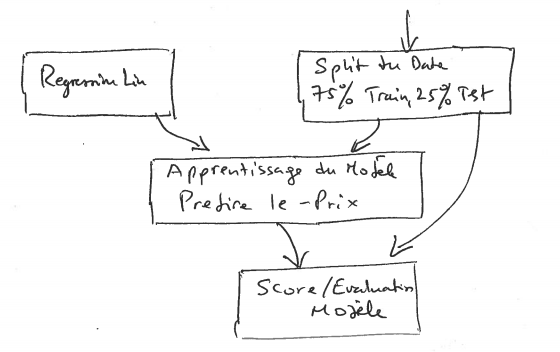


1. Après l'apprentissage, on choisit d'afficher les coefficients estimés du modèle, soit:

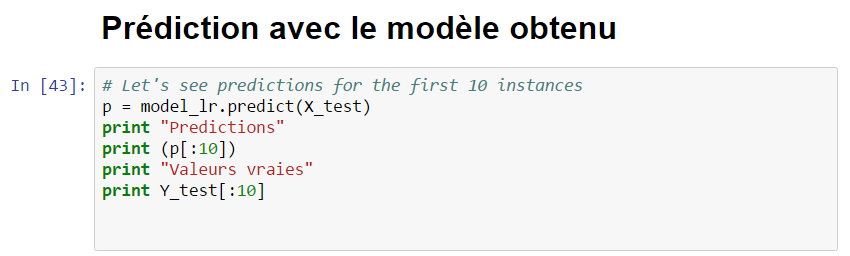


Évaluer le Score et tester le modele

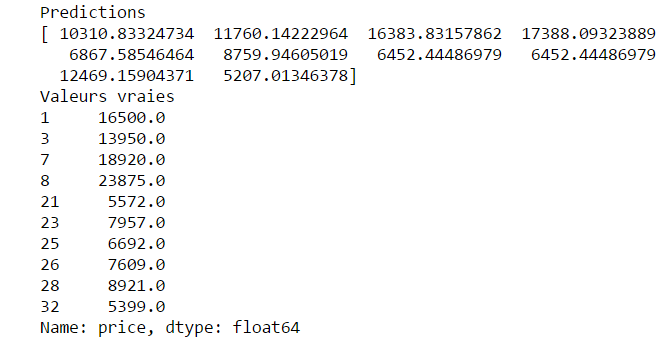
* + Étape 5: Passer au test et faire la prédiction selon le processus suivant.



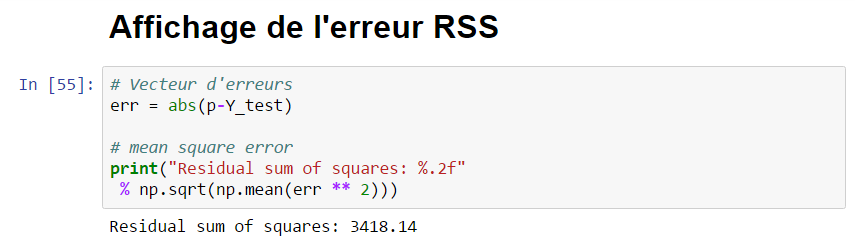
1. On exécute une prédiction sur la base de l'ensemble de test.



ce qui donne:



1. Comme mesure, on calcule l'erreur RSS, ce qui nous donne:



1. On calcule aussi le coefficient de détermination ou R2 squared:



La valeur obtenue est donc de 78%.

**Références**

<https://www.dataquest.io/blog/machine-learning-python/>

Code

# -\*- coding: utf-8 -\*-

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

# Charger le dataset automotive

#charger le fichier en remplacant les ? par NA

data = pd.read\_csv("imports-85.data", na\_values=['?'])

#data represente les données du dataset incluant la cible

print data.shape

#Affichage des données

print data

subset = data.iloc[:,25:26]

#Quelques statistiques sur la cible

print subset.describe()

#Supprimer la colonne 2 normalized losses

cols =[1,]

subset\_1 = data.drop(data.columns[cols], axis=1)

print subset\_1.shape

#print 5 lignes subset\_1 en prenant seulement 5 colonnes

print subset\_1.iloc[:5,:6]

#Supprimer les lignes avec données incompletes NA ou NaN

subset\_2 = subset\_1.dropna()

print subset\_2.shape

#Selection des features suivantes

#make, body-style, wheel-base, engine-size,

#horsepower, peak-rpm, highway-mpg, price

#drop colonnes autres que ceux qui sont dans le modele ci-dessus

cols=["symboling", "fuel-type","aspiration",

"num-of-doors",

"drive-wheels",

"engine-location",

"length",

"width", "height",

"curb-weight", "engine-type","num-of-cylinders",

"fuel-system", "bore","stroke", "compression-ratio","city-mpg"]

subset\_f = subset\_2.drop(cols, axis=1)

print "Apres selection des features"

print subset\_f.shape

#print 5 rows de subset\_6

print subset\_f.iloc[:5,:]

# Plot outputs

import pylab as pl

%matplotlib inline

pl.plot( subset\_f.iloc[:,4:5],subset\_f.iloc[:,7:8],'ro')

pl.xlabel('horsepower')

pl.ylabel('prix')

pl.show()

# Plot outputs

import pylab as pl

%matplotlib inline

pl.plot( subset\_f.iloc[:,6:7],subset\_f.iloc[:,7:8],'ro')

pl.xlabel('highway-mpg')

pl.ylabel('prix')

pl.show()

#test et training data

# Import de la fonction pour le split

from sklearn.cross\_validation import train\_test\_split

# Generer le training set.

train = subset\_f.sample(frac=0.75, random\_state=1)

# Print shapes de l'ensemble de training

print "shape de training"

print train.shape

# Selectionner le reset pour le test.

test = subset\_f.loc[~subset\_f.index.isin(train.index)]

print "shape de test"

print test.shape

#Training du model

# Importer le modele linear regression.

from sklearn.linear\_model import LinearRegression

columns = [ "wheel-base", "engine-size", "horsepower",

"peak-rpm", "highway-mpg"]

# Afficher les colonnes predicteurs.

print train.columns.tolist()

target="price"

#Déterminer les ensembles prédicteurs et cible

X\_train = train[columns]

Y\_train = train[target]

X\_test = test[columns]

Y\_test = test[target]

#Apprentissage du modele LR

model\_lr = LinearRegression()

model\_lr.fit(train[columns], train[target])

# Affichage des coefficient

print('Coefficients du modele:', model\_lr.coef\_)

# Let's see predictions for the first 10 instances

p = model\_lr.predict(X\_test)

print "Predictions"

print (p[:10])

print "Valeurs vraies"

print Y\_test[:10]

# Vecteur d'erreurs

err = abs(p-Y\_test)

# mean square error

print("Residual sum of squares: %.2f"

% np.sqrt(np.mean(err \*\* 2)))

from sklearn.metrics import r2\_score

print("R2 ou coefficient de determination:%.3f ") % (r2\_score(Y\_test, p))