

Cahier thématique

Classification avec k-neighbors (KNN) et évaluation de performance

**Manipulation 1**: Utilisation d’un algorithme de classification du type k-Neighbors  
**Manipulation 2**: Utilisation d’un algorithme de classification du type logistic regression pour comparaison avec kNN

**Manipulation 3**: Évaluation de la performance de l'algorithme kNN avec différentes valeurs de k

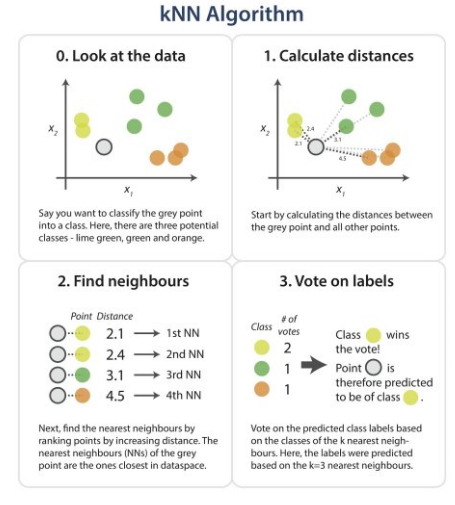
**Préparé par :** Hafed Benteftifa  
  
© Hafed Benteftifa et Nesrine Zemirli 2015-2016

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** |
| 10 juin 2016 | 1.0 | Version initiale |
|  |  |  |
|  |  |  |

**Contexte**

****

**Réf: http://www.kdnuggets.com/2016/01/implementing-your-own-knn-using-python.html**

**Manipulation 1**: Utilisation d’un algorithme de classification du type k-Neighbors

**Objectif**

Procéder à une classification d'instances en utilisant l'algorithme kNN.

**Préliminaire**

* Anaconda ou pycharm est disponible.

**Approche**

1. On utilisera le dataset connu comme le iris flower dataset. Celui-ci contient 150 instances. Les features sont au nombre de 4 et sont des valeurs numériques et le label de classe ou cible sera l'espèce d'iris.

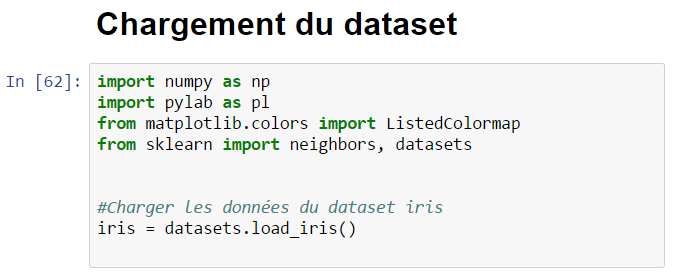
Les étapes que l'on va suivre sont les mêmes que pour l'algorithme de régression linéaire et sont les suivantes

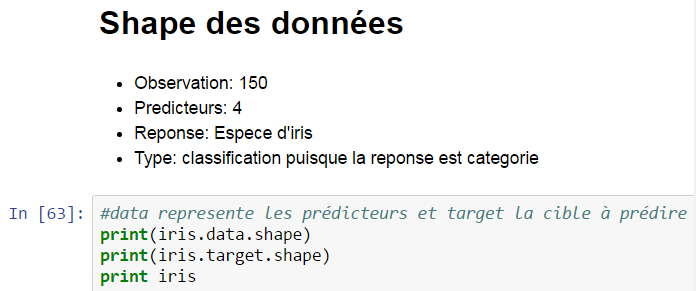
* Créer le modèle
  + Étape 1: obtenir le data
  + Étape 2: faire un pré-traitement des données
  + Étape 3: Définir les features
* Faire l'apprentissage du modele
  + Étape 4: choisir un algorithme ML et procéder à l'apprentissage
* Évaluer la performance de l'algorithme et tester le modele
  + Étape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

**Démarche**

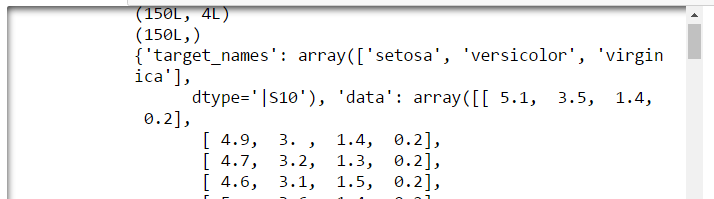
Étape 1: obtenir le data

1. On importe le dataset iris.

Noter les différents import de modules qui seront nécessaires pour les étapes de traitement

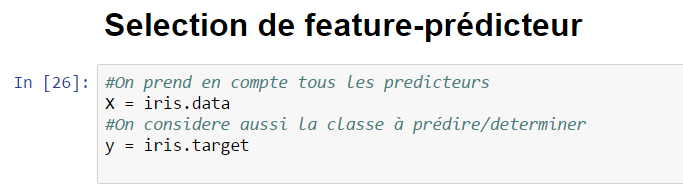
2. Dans la deuxième étape, on procède à l'exploration des données. Les dimensions du dataset iris sont affichées comme suit:

En sortie, on devrait avoir un résultat similaire au suivant:

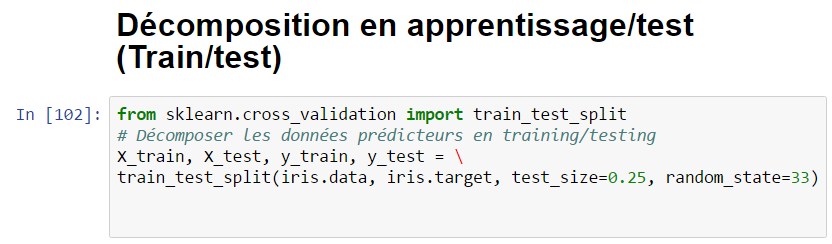


1. On sélectionne tous les prédicteurs du dataset existant iris comme montré ci-dessous

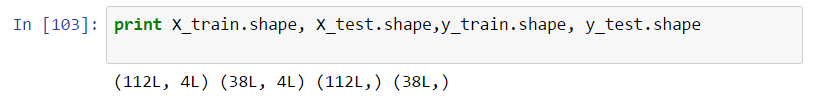




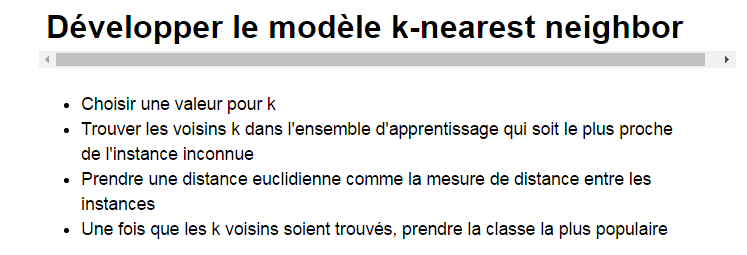
3. On décompose maintenant le dataset en ensemble d'apprentissage et test avec une séparation de 75%, soit



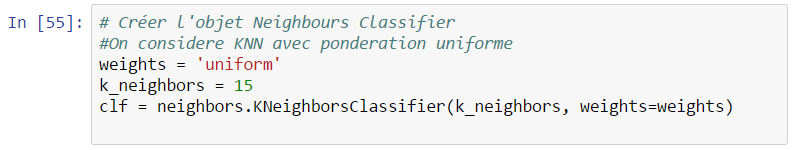
ce qui donne pour les deux ensembles les résultats suivants:



1. On procède maintenant à l'instantiation du modèle KNN comme suit:



Le code nécessaire pour le modèle est alors:

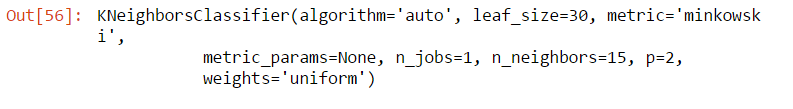


Dans ce cas, on voit que les poids des coefficients sont choisis de manière uniforme et k=15. Vous aurez à tester l'algorithme avec k=1 pour vérifier la performance de votre algorithme dans une autre manipulation.

4. On procède maintenant à l'apprentissage du modèle comme suit:

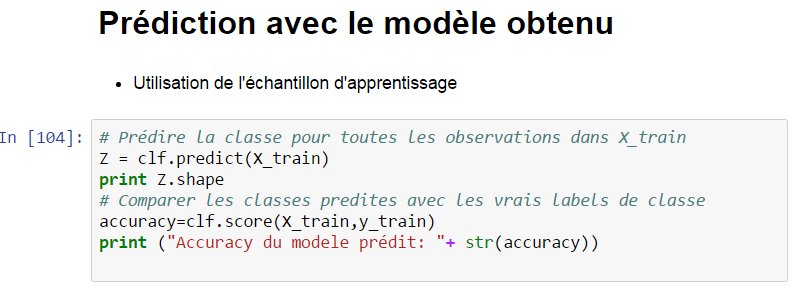


ce qui donne le détail suivant:



On constate que la distance critère par défaut est "minkowski"

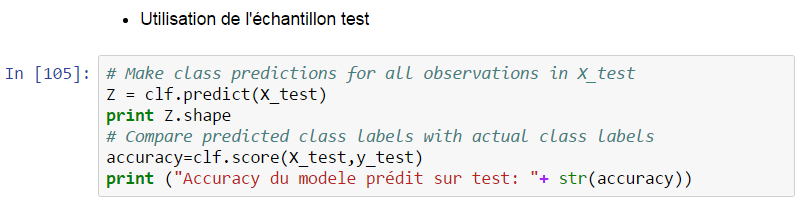
6. On évalue maintenant la performance de l’algorithme utilisé comme suit en utilisant l'echantillon  d'apprentissage, soit X\_train



ce qui donne en sortie:



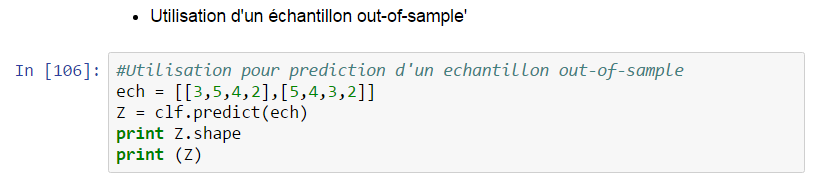
6. On évalue maintenant la performance de l’algorithme utilisé comme suit en utilisant l'echantillon  test, soit X\_test



ce qui donne:



6. On évalue maintenant la performance de l’algorithme utilisé comme suit en utilisant un echantillon out-of-sample



ce qui donne:



De ce fait, la classe cible pour chacun des deux échantillons est 1. Il suffit d'évaluer le dataset iris pour relier la classe 1 à la vraie espèce d'iris.

**Manipulation 2**: Utilisation d’un algorithme de classification du type logistic regression pour comparaison avec kNN

**Objectif**

Procéder à une classification d'instances en utilisant l'algorithme logistic regression

**Préliminaire**

* Anaconda ou pycharm est disponible.

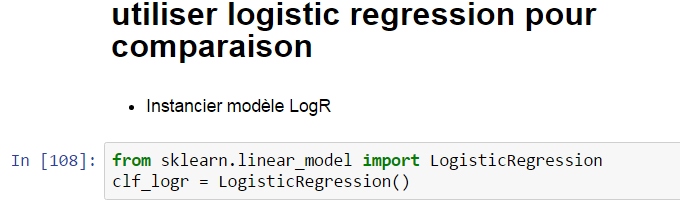
**Approche**

1. Une fois les résultats obtenus pour la classification KNN, on est tenté de déclarer qu'avec plus de 90% d'accuracy que KNN est un très bon classificateur (avec k=15).

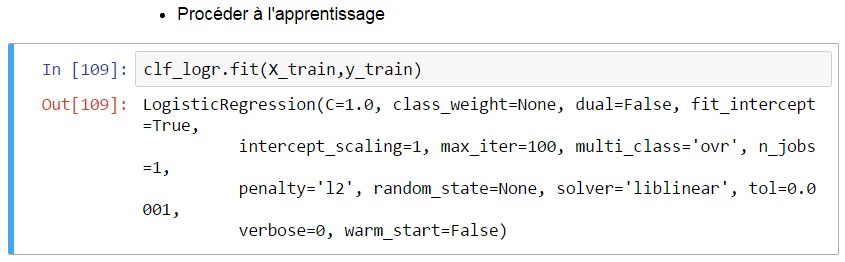
En général, la conclusion ne peut pas être aussi évidente et l'on doit évaluer d'autres alternatives en termes d'algorithmes.

Dans le cas présent, on va dérouler l'algorithme de logistic regression afin de pouvoir comparer le résultat en termes d'accuracy avec celui de KNN.

2. Créer le modèle comme suit.

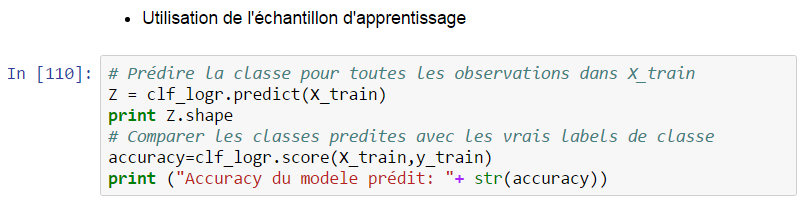


3. Procéder à l'apprentissage avec la fonction fit()



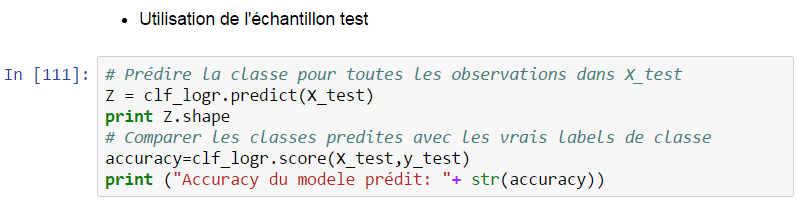
On notera les différentes valeurs par défaut du modèle que l'on a instantié.

4. En termes de prédiction de la classe cible et en utilisant l'échantillon d'apprentissage, on aura:

ce qui donne en sortie le résultat suivant:



5. On répete la même opération pour l'échantillon test et on aura:

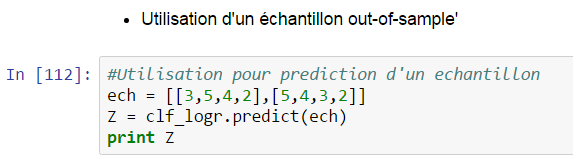


En sortie, on aura:



On constate que l'accuracy est à 1. Il est fort probable que c'est une valeur qui sera différentes pour un autre échantillon test.

6. Finalement, on teste pour un échantillon out-of sample, et on aura:



En sortie, les deux classes obtenues sont:



Le résultat est totalement différent de celui obtenu avec KNN.

**Manipulation 3**: Évaluation de la performance de l'algorithme kNN avec différentes valeurs de k

**Objectif**

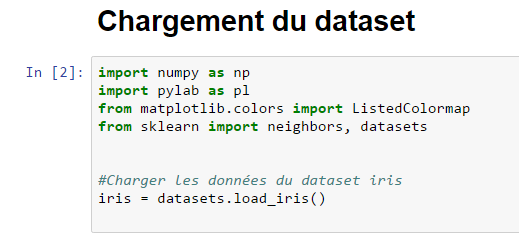
Procéder à une évaluation de la performance de l'algorithme kNN en variant la valeur de k.

**Préliminaire**

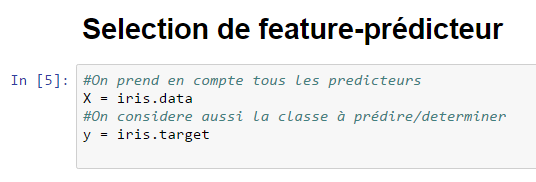
* Anaconda ou pycharm est disponible.

**Approche**

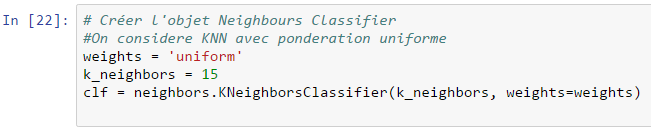
1. Créer un notebook et reprendre toute la partie de mise en place du classificateur KNN des manipulations précédentes. En premier, on procède au chargement du dataset iris, soit:

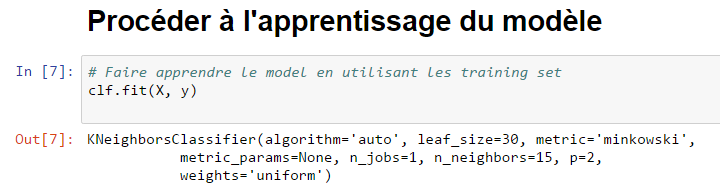


2. On sélectionne les prédicteurs et la classe cible en prenant la totalité du dataset, soit:

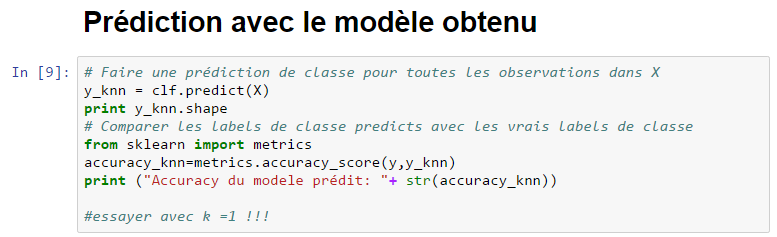


3. On instantie alors le modèle KNN avec K=15

2. On procède à l'apprentissage du modèle comme d'habitude

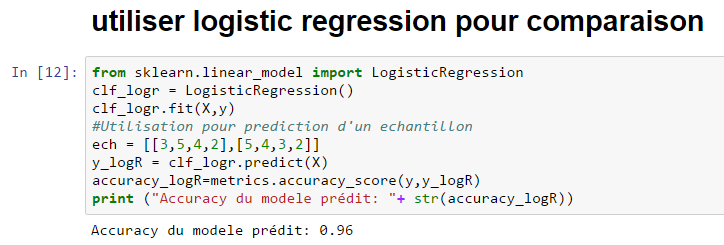


5. On évalue ensuite le score accuracy

 ce qui donne comme valeur:

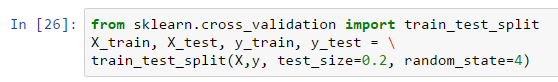


6. On détermine l'accuracy pour l'approche logistic regression, ce qui donne:

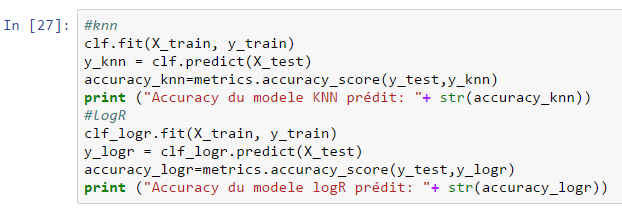


On constate que pour les paramètres actuels, l'algorithme KNN performe mieux que l'algorithme logistic regression.

7. On évalue maintenant la performance avec un split du dataset en échantillon d'apprentissage et test avec coupure à 80%

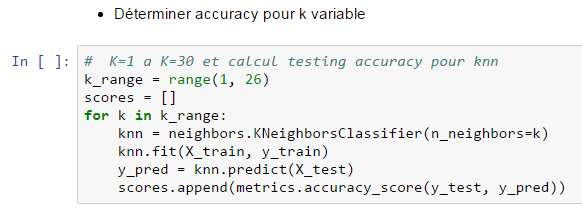


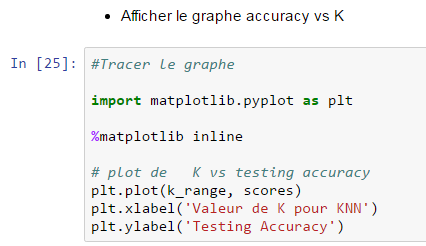
8. On refait le même calcul d'accuracy et on aura pour les deux algorithmes:

 On constate que KNN est encore plus performant que LogR

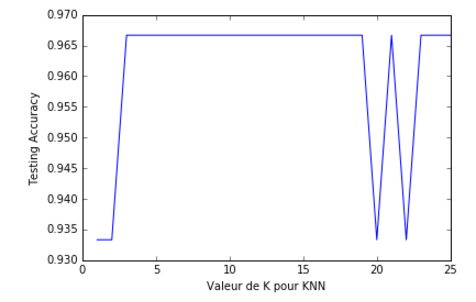


9. On détermine maintenant la variation de l'accuracy selon la valeur de k pour l'algorithme KNN.

 10. Une fois les scores obtenus, on trace le graphe correspondant, soit



ce qui donne en sortie le graphe suivant:



On voit que les valeurs adéquates pour K se situent entre k=3 et k=19

**Annexe**

https://blog.cambridgecoding.com/2016/01/16/machine-learning-under-the-hood-writing-your-own-k-nearest-neighbour-algorithm/

http://online.cambridgecoding.com/notebooks/cca\_admin/getting-started-with-regression-and-decision-trees