K Nearest Neighbors (KNN)

Plan

- Introduction
- Principe
- Algorithme
- Calcul de similarité
- Comment choisir la valeur K?
- Bilan: Avantages et Inconvénients

Introduction

Principe de K-NN : ***dis moi qui sont tes voisins, je te dirais qui tu es !***

- Algorithme d'apprentissage supervisé
- utilisé pour la classification et la régression
- pour K-NN, il n'existe pas de phase d'apprentissage. C'est pour cela qu'on le catégorise dans le **Lazy Learning**.
- Pour pouvoir effectuer une prédiction, K-NN **se base sur le jeu de données** pour produire un résultat.

Principe

- Pour effectuer une prédiction, l'algorithme K-NN **se base sur le jeu de données en entier**.
- pour une observation, qui ne fait pas partie du jeu de données, qu'on souhaite prédire, l'algorithme va chercher les **K instances du jeu de données les plus proches de notre observation.**
- Ensuite pour ces voisins, l'algorithme se basera sur leurs variables de sortie *(output variable) pour calculer la valeur de la variable de l'observation qu'on souhaite prédire.*
- Si K-NN est utilisé pour la régression, c'est la moyenne (ou la médiane) des variables des plus proches observations qui servira pour la prédiction
- Si K-NN est utilisé pour la classification, c'est la classe des variables des plus proches observations qui servira pour la prédiction

Algorithme

Début Algorithme

Données en entrée :

- un ensemble de données.
- une fonction de définition distance .
- Un nombre entier

Pour une nouvelle observation z dont on veut prédire sa variable de sortie

Faire :

1. Calculer toutes les distances de cette observation avec les autres observations du jeu de données

2. Retenir les **k** observations du jeu de données les proches de **z** en utilisant la fonction de calcul de distance

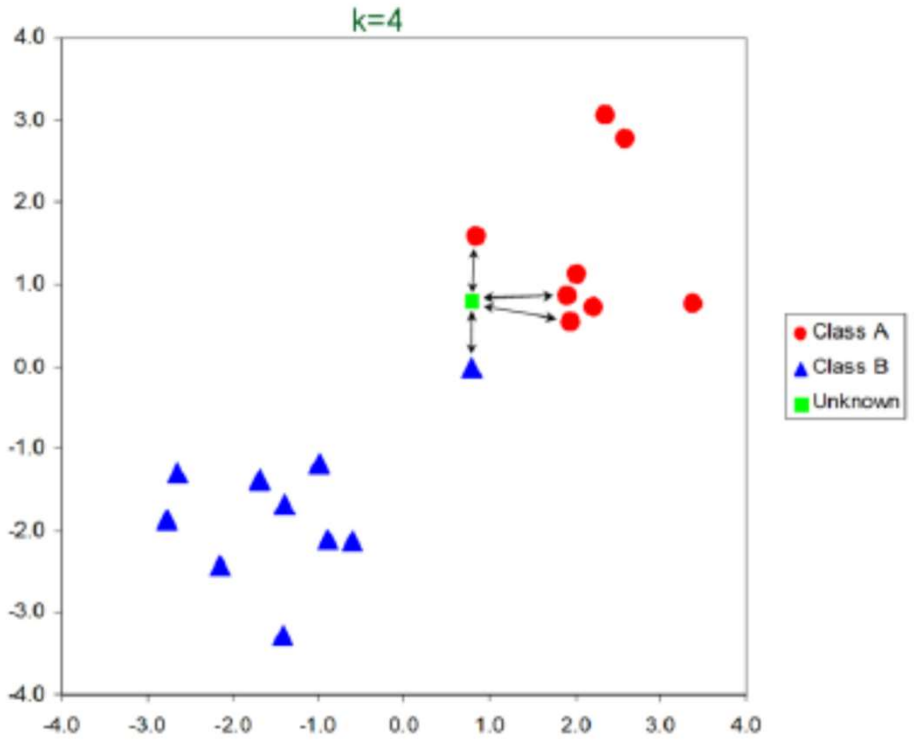
3. Prendre les valeurs des observations retenues :

1. Si on effectue une régression, calculer la moyenne (ou la médiane) de retenues
2. Si on effectue une classification , calculer le mode de k retenues

4.Retourner la valeur calculée dans l'étape 3 comme étant la valeur qui a été prédite par K-NN pour l'observation .

Fin Algorithme

Exemple


k=4
4.0
3.0
2.0
1.0
0.0
-1.0
-2.0
-3.0
-4.0
-4.0
-3.0
-2.0
-1.0
0.0
1.0
2.0
3.0
4.0
Class A
Class B
Unknown

Calcul de similarité

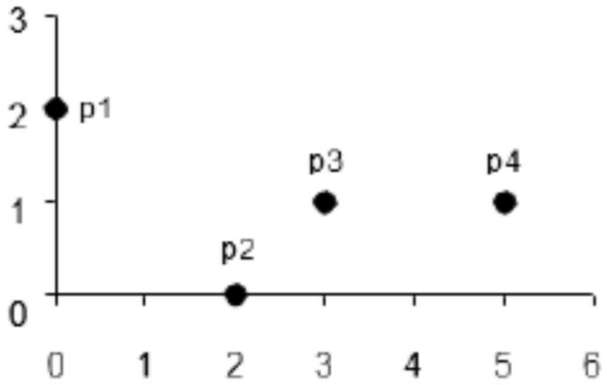
- Comme on vient de le voir dans notre écriture algorithme, K-NN a besoin d'une **fonction de calcul de distance** entre deux observations. Plus deux points sont proches l'un de l'autre, plus ils sont similaires et vice versa.
- Il existe plusieurs fonctions de calcul de distance, notamment, la distance euclidienne, la distance de Manhattan, la distance de Minkowski, celle de Jaccard, la distance de Hamming…etc.
- On choisit la fonction de distance en fonction des types de données qu'on manipule. Ainsi pour les données quantitatives (exemple : poids, salaires, taille, montant de panier éléctronique etc…) et **du même type**, la distance euclidienne est un bon candidat.
- Quant à la distance de Manhattan, elle est une bonne mesure à utiliser quand les données (*input variables)* ne sont pas du même type (exemple :age, sexe, longueur, poids etc…).
- Il est inutile de coder, soi-même ces distances, généralement, les librairies de Machine Learning comme Scikit Learn, effectue ces calculs en interne. Il suffit juste d'indiquer la mesure de distance qu'on souhaite utiliser.

La distance Euclidienne

- distance qui calcule la racine carrée de la somme des différences carrées entre les coordonnées de deux points :

$$D_e(x, y) = \sqrt{\sum_{j=1}^{n} (x_j - y_j)^2}$$

La distance Euclidienne



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Table 1. Euclidean distance matrix D listing all possible pairwise Euclidean distances between 19 samples.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}
x_2	1.5																	
x_3	1.4	1.6																
x_4	1.6	1.4	1.3															
x_5	1.7	1.4	1.5	1.5														
x_6	1.3	1.4	1.4	1.5	1.4													
x_7	1.6	1.3	1.4	1.4	1.5	1.8												
x_8	1.5	1.4	1.6	1.3	1.7	1.6	1.4											
x_9	1.4	1.3	1.4	1.5	1.2	1.4	1.3	1.5										
x_{10}	2.3	2.4	2.5	2.3	2.6	2.7	2.8	2.7	3.1									
x_{11}	2.9	2.8	2.9	3.0	2.9	3.1	2.9	3.1	3.0	1.5								
x_{12}	3.2	3.3	3.2	3.1	3.3	3.4	3.3	3.4	3.5	3.3	1.6							
x_{13}	3.3	3.4	3.2	3.2	3.3	3.4	3.2	3.3	3.5	3.6	1.4	1.7						
x_{14}	3.4	3.2	3.5	3.4	3.7	3.5	3.6	3.3	3.5	3.6	1.5	1.8	0.5					
x_{15}	4.2	4.1	4.1	4.1	4.1	4.1	4.1	4.1	4.1	4.1	1.7	1.6	0.3	0.5				
x_{16}	4.1	4.1	4.1	4.1	4.1	4.1	4.1	4.1	4.1	4.1	1.6	1.5	0.4	0.5	0.4			
x_{17}	5.9	6.2	6.2	5.8	6.1	6.0	6.1	5.9	5.8	6.0	2.3	2.3	2.5	2.3	2.4	2.5		
x_{18}	6.1	6.3	6.2	5.8	6.1	6.0	6.1	5.9	5.8	6.0	3.1	2.7	2.6	2.3	2.5	2.6	3.0	
x_{19}	6.0	6.1	6.2	5.8	6.1	6.0	6.1	5.9	5.8	6.0	3.0	2.9	2.7	2.4	2.5	2.8	3.1	0.4

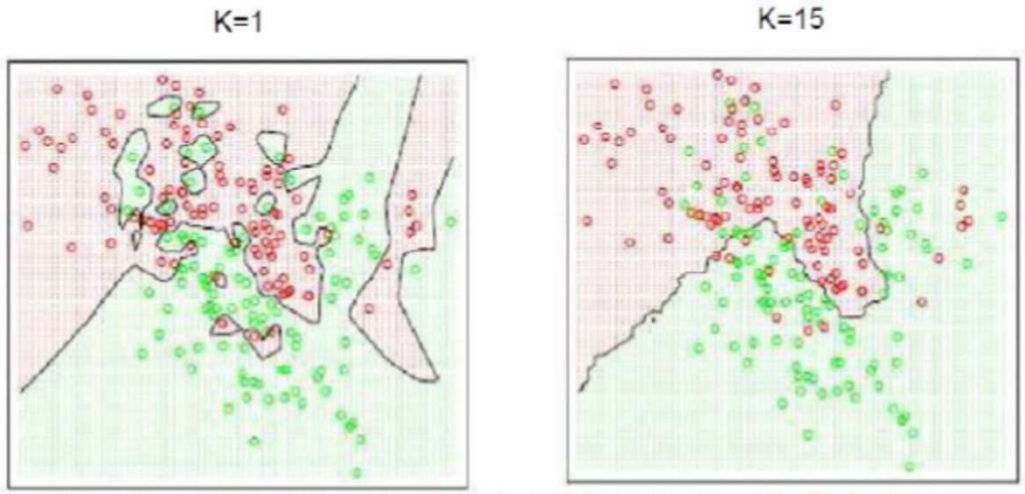
Distance de Manhattan

- la distance de Manhattan: calcule la somme des valeurs absolues des différences entre les coordonnées de deux points

$$D_m(x, y) = \sum_{i=1}^{k} \mid x_i - y_i \mid$$

Comment choisir le paramètre k

- Un k plus grand produit un effet de frontière plus lisse ?? Lorsque K==N, toujours prédire la classe majoritaire



Figures from Hastie, Tibshirani and Friedman (Elements of Statistical Learning)

Bilan

Avantages

- L'apprentissage et la mise en œuvre sont extrêmement simples et Intuitifs
- Limites de décision flexibles

Les inconvénients

- Les caractéristiques non pertinentes ou corrélées ont un impact élevé et doivent être éliminé
- Typiquement difficile à gérer haute dimensionnalité
- Coûts de calcul : temps de calcul et taille de mémoire

Naive Bayes

Plan

- Introduction
- Probabilité conditionnelle
- Algorithme

Définition

Le *naive Bayes classifier* se base sur le **théorème de Bayes**.
Ce dernier est un classique de la théorie des probabilités. Ce théorème est fondé sur les **probabilités conditionnelles**.

Probabilité conditionnelle : Quelle est la probabilité qu'un événement se produise **sachant qu'un autre événement s'est déjà produit.**

$$P(C_k \mid X_1,\ldots,X_n) = \frac{P(X_1,\ldots,X_n \mid C_k)\,P(C_k)}{P(X_1,\ldots,X_n)}$$

Définition

- **Un ensemble d'apprentissage X**, où chaque instance d'apprentissage x est représenté comme un vecteur d'attributs de dimension n : $x\ (x_1, x_2, ..., x_n)$
- **Un ensemble prédéfini de classes: $C = \{C_1, C_2, ..., C_m\}$**
- Étant donné une nouvelle instance z, dans quelle classe z doit-il être classé ?
- Nous voulons trouver la classe la plus probable pour z

$$c_{MAP} = \arg\max_{c_i \in C} P(c_i \mid z) \qquad c_{MAP} = \arg\max_{c_i \in C} P(c_i \mid z_1, z_2, ..., z_n)$$

$$c_{MAP} = \arg\max_{c_i \in C} \frac{P(z_1, z_2, ..., z_n \mid c_i) * P(c_i)}{P(z_1, z_2, ..., z_n)} \qquad \text{(by Bayes theorem)}$$

$$c_{MAP} = \arg\max_{c_i \in C} P(z_1, z_2, ..., z_n \mid c_i) * P(c_i) \qquad (P(z_1,z_2,...,z_n) \text{ is the same for all classes})$$

Définition

- **<u>Hypothèse dans le classificateur Naïve Bayes.</u>** Les attributs sont conditionnellement indépendants,

$$P(z_1, z_2, \ldots, z_n \mid c_i) = \prod_{j=1}^{n} P(z_j \mid c_i)$$

- Le classificateur Naïve Bayes trouve la classe la plus probable pour z

$$c_{NB} = \arg\max_{c_i \in C} P(c_i) * \prod_{j=1}^{n} P(z_j \mid c_i)$$

Algorithme

- The learning (training) phase (given a training set)

  For each class $c_i \in C$

  - Estimate the prior probability: $P(c_i)$
  - For each attribute value z_j, estimate the probability of that attribute value given class c_i: $P(z_j|c_i)$

- The classification phase
  - For each class $c_i \in C$, compute the formula

$$P(c_i) * \prod_{j=1}^{n} P(z_j \mid c_i)$$

  - Select the most probable class c^*

$$c^* = \underset{c_i \in C}{\operatorname{argmax}}\, P(c_i) * \prod_{j=1}^{n} P(z_j \mid c_i)$$

Exemple

Rec. ID	Age	Income	Student	Credit_Rating	Buy_Computer
1	Young	High	No	Fair	No
2	Young	High	No	Excellent	No
3	Medium	High	No	Fair	Yes
4	Old	Medium	No	Fair	Yes
5	Old	Low	Yes	Fair	Yes
6	Old	Low	Yes	Excellent	No
7	Medium	Low	Yes	Excellent	Yes
8	Young	Medium	No	Fair	No
9	Young	Low	Yes	Fair	Yes
10	Old	Medium	Yes	Fair	Yes
11	Young	Medium	Yes	Excellent	Yes
12	Medium	Medium	No	Excellent	Yes
13	Medium	High	Yes	Fair	Yes
14	Old	Medium	No	Excellent	No

- Representation of the problem
  - z = (Age=*Young*,Income=*Medium*,Student=*Yes*,Credit_Rating=*Fair*)
  - Two classes: c_1 (buy a computer) and c_2 (not buy a computer)
- Compute the prior probability for each class
  - $P(c_1) = 9/14$
  - $P(c_2) = 5/14$
- Compute the probability of each attribute value given each class
  - $P(\text{Age=Young}|c_1) = 2/9$; $P(\text{Age=Young}|c_2) = 3/5$
  - $P(\text{Income=Medium}|c_1) = 4/9$; $P(\text{Income=Medium}|c_2) = 2/5$
  - $P(\text{Student=Yes}|c_1) = 6/9$; $P(\text{Student=Yes}|c_2) = 1/5$
  - $P(\text{Credit\_Rating=Fair}|c_1) = 6/9$; $P(\text{Credit\_Rating=Fair}|c_2) = 2/5$

- Compute the likelihood of instance z given each class
  - For class c_1

    $P(z|c_1) = P(Age=Young|c_1)*P(Income=Medium|c_1)*P(Student=Yes|c_1)*$
    $P(Credit\_Rating=Fair|c_1) = (2/9)*(4/9)*(6/9)*(6/9) = 0.044$
  - For class c_2

    $P(z|c_2) = P(Age=Young|c_2)*P(Income=Medium|c_2)*P(Student=Yes|c_2)*$
    $P(Credit\_Rating=Fair|c_2) = (3/5)*(2/5)*(1/5)*(2/5) = 0.019$
- Find the most probable class
  - For class c_1

    $P(c_1)*P(z|c_1) = (9/14)*(0.044) = 0.028$
  - For class c_2

    $P(c_2)*P(z|c_2) = (5/14)*(0.019) = 0.007$

  → Conclusion: ***The person z (*a *young student* with *medium* income and *fair* credit rating) *will buy a computer*!**