RECONAISSANCE DES FORMES

Mise en œuvre de l'algorithme des K plus proches voisins

CARON Romain, NANTEL Maëlig
IMR3 — 2016/2017

Sommaire

1. Introduction

L'objectif de ce premier TP du module « reconnaissance des formes » est de mettre en œuvre l'algorithme du « k plus proche voisin » afin d'effectuer de la prédiction de données. C'est un algorithme d'approche statistique, c'est-à-dire qu'il utilise un ensemble d'apprentissages (training set) afin d'acquérir de la connaissance, qui sera alors testée sur un ensemble de tests (test set) afin de prédire un résultat manquant.

2. Jeu de données

2.1. Présentation

Pour mettre en œuvre cet algorithme, nous nous sommes basés sur le célèbre ensemble « Iris Data Set » de R.A Fisher. Ce jeu de données concerne les fleurs et l'objectif est de permettre une classification de ces dernières. Les différentes classes possibles sont :

- Iris Setosa
- Iris Versicolour
- Iris Virginica

Après une première analyse du jeu de données, sait que :

- Il n'y a pas de valeur manquante
- Il y a 4 attributs (feature) par fleur
- Il y a 150 instances équitablement réparties dans les 3 classes (50 par classe, soit 33,3 % de l'ensemble).

Les attributs considérés pour les fleurs sont :

- La taille de la tige en centimètres (sepal length)
- La largeur de la tige en centimètres (sepal width)
- La taille des pétales en centimètres (petal length)
- La largeur des pétales en centimètres (petal width)

Tous ces attributs sont des nombres réels positifs et continus.

Les statistiques du jeu de données sont :

	Min	Max	Moyenne	Écart type	Class corrélation
Longueur tige	4,3	7,9	5,84	0,83	0,782 6
Largeur tige	2,0	4,4	3,05	0,43	-0,419 4
Longueur pétale	1,0	6,9	3,76	1,76	0,949 0
Largeur pétale	0,1	2,5	1,20	0,76	0,956 5

À l'aide de ces statistiques, on observe qu'il y a une forte corrélation entre le résultat à prédire et les attributs liés aux pétales. Au contraire, la largeur de la tige semble peu importante pour prédire la classe de l'iris.

2.2. Décomposition en « training set » et « test set »

Comme nous l'avons indiqué dans l'introduction, les approches statistiques de prédiction nécessitent deux jeux donnés distincts :

- Un ensemble d'apprentissages (training set)
- Un ensemble de test (test set)

Il est donc nécessaire de décomposer ces 150 instances en deux groupes. Cette décomposition va bien évidemment impacter les résultats de prédiction, quel que soit l'algorithme utilisé.

3. Algorithme des K plus proches voisins

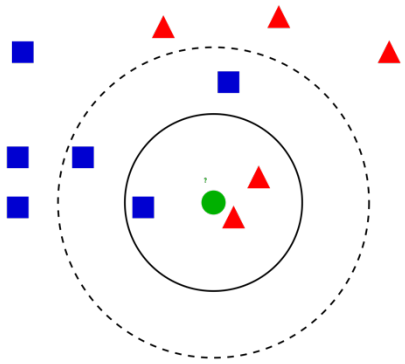
3.1. Présentation et fonctionnement

L'algorithme des K plus proches voisins (K nearest neighbor ou KNN) est une méthode d'apprentissage supervisé dont le principe sous-jacent est : « dis-moi qui sont tes amis et je te dirais qui tu es ». Cet algorithme va se baser sur le voisinage d'un point que l'on cherche à classifier. La classe la plus présente dans le voisinage considéré sera alors prédite comme étant la classe de l'instance testée.

Par définition, la mise en œuvre de cette méthodologie soulève deux questions importantes :

- Combien de voisins retenir ? (Valeur de K)
- Quelle distance considérée pour déterminer le voisinage ?

Les résultats obtenus par la méthode dépendent de ces deux critères. On peut arriver à des résultats totalement différents selon les paramètres utilisés.



Source : en.wikipedia.org

Sur l'exemple ci-dessus, on cherche à déterminer la classe de l'instance verte. Est un carré bleu ou un triangle rouge ? L'algorithme va donc calculer la distance du rond vert par rapport à tous les autres points. Cette étape permet de disposer les différentes instances comme sur la figure ci-dessus. Une fois cette étape réalisée, on cherche les K plus proches voisins (à partir de la distance que l'on vient de calculer) et on regarde la classe la plus présente dans ce voisinage. Ici, pour K = 3 (premier cercle), on prédira que le rond vert est en réalisé un triangle rouge. En revanche, si on prend K = 5 (cercle pointillé), ce sera un carré bleu !

Comparé à d'autres algorithmes de Machine Learning, la particularité du KNN est de traiter le training set à chaque fois. Il ne nécessite donc pas une « phase d'apprentissage ». Le principal inconvénient de cette approche sera visible pour des jeux de données très

volumineux. Pour chaque nouvelle prédiction, le training set entier est étudié, ce qui peut prendre du temps.

3.2. Mises-en œuvre et choix d'implémentation

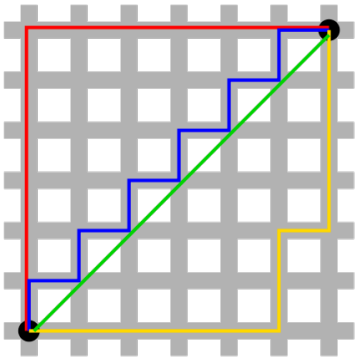
Pour mettre en œuvre cet algorithme, nous avons fait le choix du langage Java. Nous avons utilisé les librairies externes suivantes :

- Apache Commons CSV, pour la lecture des données CSV
- Habernal Confusion Matrix, pour l'affichage de la matrice de confusion

Ce sont des outils sur lesquels nous nous sommes appuyés. Le fonctionnement de l'algorithme a été codé par nos soins.

Nous avons décidé de prendre en compte deux distances différentes :

- Distance Euclidienne (distance à vol d'oiseau entre 2 points)
- Distance de Manhattan (distance en angles droits entre 2 points)



Source : fr.wikipedia.org

En verte, on retrouve la distance Euclidienne. En rouge, bleu et jaune (équivalents), la distance de Manhattan.

4. Résultats

Pour étudier les résultats de l'algorithme en fonction de ses paramètres, nous avons décidé de nous baser sur les mesures suivantes :

- Nombre de bonnes et mauvaises prédictions
- Précision moyenne sur l'ensemble du test set
- Matrice de confusion des prédictions. Elle nous permettra en plus de voir où se trouvent les mauvaises prédictions.

En fonction de K et du type de distance prise en compte, nous obtenons les résultats suivants :

```
For K=1 and DistanceType EUCLIDIENNE results are:
     - Total predictions = 75 | good = 73 | bad = 2
     - Accuracy = 97.33333333333334%

 ↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
      SETOSA          25           0           0
 VERSICOLOUR           0          23           2
   VIRGINICA           0           0          25
```

```
For K=1 and DistanceType MANHATTAN results are:
     - Total predictions = 75 | good = 67 | bad = 8
     - Accuracy = 89.33333333333333%

 ↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
      SETOSA          25           0           0
 VERSICOLOUR           0          24           1
   VIRGINICA           0           7          18
```

```
For K=2 and DistanceType EUCLIDIENNE results are:
     - Total predictions = 75 | good = 69 | bad = 6
     - Accuracy = 92.0%

 ↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
      SETOSA          25           0           0
 VERSICOLOUR           0          23           2
   VIRGINICA           0           4          21
```

```
For K=2 and DistanceType MANHATTAN results are:
     - Total predictions = 75 | good = 69 | bad = 6
     - Accuracy = 92.0%

 ↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
      SETOSA          25           0           0
 VERSICOLOUR           0          23           2
   VIRGINICA           0           4          21
```

```
For K=5 and DistanceType EUCLIDIENNE results are:
     - Total predictions = 75 | good = 71 | bad = 4
     - Accuracy = 94.66666666666667%

 ↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
      SETOSA          25           0           0
 VERSICOLOUR           0          23           2
   VIRGINICA           0           2          23
```

```
For K=5 and DistanceType MANHATTAN results are:
     - Total predictions = 75 | good = 68 | bad = 7
     - Accuracy = 90.66666666666666%

 ↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
      SETOSA          25           0           0
 VERSICOLOUR           0          23           2
   VIRGINICA           0           5          20
```

```
For K=15 and DistanceType EUCLIDIENNE results are:
     - Total predictions = 75 | good = 71 | bad = 4
     - Accuracy = 94.66666666666667%

 ↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
      SETOSA          25           0           0
 VERSICOLOUR           0          24           1
   VIRGINICA           0           3          22
```

```
For K=15 and DistanceType MANHATTAN results are:
     - Total predictions = 75 | good = 71 | bad = 4
     - Accuracy = 94.66666666666667%

 ↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
      SETOSA          25           0           0
 VERSICOLOUR           0          25           0
   VIRGINICA           0           4          21
```

```
For K=75 and DistanceType EUCLIDIENNE results are:
     - Total predictions = 75 | good = 25 | bad = 50
     - Accuracy = 33.33333333333333%

↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
     SETOSA           0           0          25
VERSICOLOUR           0           0          25
  VIRGINICA           0           0          25
```

```
For K=75 and DistanceType MANHATTAN results are:
     - Total predictions = 75 | good = 25 | bad = 50
     - Accuracy = 33.33333333333333%

↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
     SETOSA           0          25           0
VERSICOLOUR           0          25           0
  VIRGINICA           0          25           0
```

On peut donc dire que globalement, les résultats sont satisfaisants. Cependant, lorsque K devient très grand (ici, 75), on observe bien une baisse importante du taux de précision. Il y a autant de voisins de chaque classe à proximité de l'instance à prédire, ce qui donne un taux de précision de 33,3 % (on prédit toujours la même classe). En fonction de la distance utilisée, la classe prédite peut varier.

Dans l'étude du jeu de données, nous avons vu qu'il y avait une forte corrélation entre les features concernant les pétales et la classe à prédire. À l'inverse, les features concernant les tiges semblaient moins importantes. Nous avons donc décidé de tester l'algorithme en ne prenant en compte que la distance concernant les pétales.

```
For K=2 and DistanceType EUCLIDIENNE results are:
     - Total predictions = 75 | good = 71 | bad = 4
     - Accuracy = 94.66666666666667%

↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
     SETOSA          25           0           0
VERSICOLOUR           0          24           1
  VIRGINICA           0           3          22
```

```
For K=2 and DistanceType MANHATTAN results are:
     - Total predictions = 75 | good = 71 | bad = 4
     - Accuracy = 94.66666666666667%

↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
     SETOSA          25           0           0
VERSICOLOUR           0          24           1
  VIRGINICA           0           3          22
```

```
For K=5 and DistanceType EUCLIDIENNE results are:
     - Total predictions = 75 | good = 73 | bad = 2
     - Accuracy = 97.33333333333334%

↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
     SETOSA          25           0           0
VERSICOLOUR           0          23           2
  VIRGINICA           0           0          25
```

```
For K=5 and DistanceType MANHATTAN results are:
     - Total predictions = 75 | good = 73 | bad = 2
     - Accuracy = 97.33333333333334%

↓gold\pred→      SETOSA VERSICOLOUR   VIRGINICA
     SETOSA          25           0           0
VERSICOLOUR           0          23           2
  VIRGINICA           0           0          25
```

On remarque alors que les deux distances produisent ici exactement les mêmes résultats. De plus, on note une légère diminution du nombre de mauvaises prédictions (jusqu'à — 5/75 !), ce qui tend à confirmer que les features sur les pétales sont plus importantes que les features sur les tiges.

Les résultats étant liés à la répartition du training set et du data set, nous avons effectué les mêmes tests en inversant les deux (le training set devant le test set et inversement). Aucune différence notable n'a été observée, ce qui semble confirmer que les instances ne sont pas réparties dans un autre ordre que les classes d'iris.

Tous ces résultats sont entièrement déterministes. Vous pouvez les obtenir à condition d'utiliser les mêmes paramètres et la même décomposition des training set et test set.

4.1. Pour aller plus loin…

Il serait intéressant d'étudier d'autres distances afin de voir leur impact. Ici, nous nous sommes seulement concentrés sur les distances de Manhattan et Euclidienne. De même, comme nous l'avons vu dans l'analyse du jeu de donnée, toutes les features n'ont pas les mêmes ordres de valeurs. Avec l'implémentation que nous avons faite, les features aux valeurs élevées (comme la longueur de la tige) vont avoir un poids plus important dans le calcul de la distance. Afin de ne pas privilégier une feature par rapport à une autre, il serait intéressant de centrer et réduire ces variables.

5. Conclusion

Ce premier TP consistant à mettre en œuvre l'algorithme des K plus proches voisins nous a permis de mettre en application les notions vues en cours magistraux. Ce TP reprend les notions que nous avons abordé l'an passé en cours de « Machine Learning » à l'université de Dublin. L'an dernier, nous nous sommes « contentés » d'utiliser ces algorithmes afin d'observer leurs résultats. Ainsi, cette année il était très intéressant pour nous de l'implémenter afin de mieux en comprendre son les tenants et aboutissants.

Nous avons pu analyser que chaque jeu de donnée produira des résultats différents et qu'il ne pouvait pas y avoir de paramètres universels convenant au mieux à chaque data set. Il est nécessaire de bien comprendre et étudier les données dont nous disposons afin de pouvoir utiliser des algorithmes de reconnaissance.