TP1

November 28, 2019

Apprentissage par la méthode des k-nn

Prendre un échantillon de données appelé data avec une taille de 5000 exemples à l'aide de la fonction np.random.randint(70000, size=5000). Diviser la base de données à 80% pour l'apprentissage (training) et à 20% pour les tests. Entrainez un classifieur k-nn avec k = 10 sur le jeu de données chargé.

[2]:
```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=10, p=2,
                     weights='uniform')
```

Afficher la classe de l'image 9 et sa classe prédite

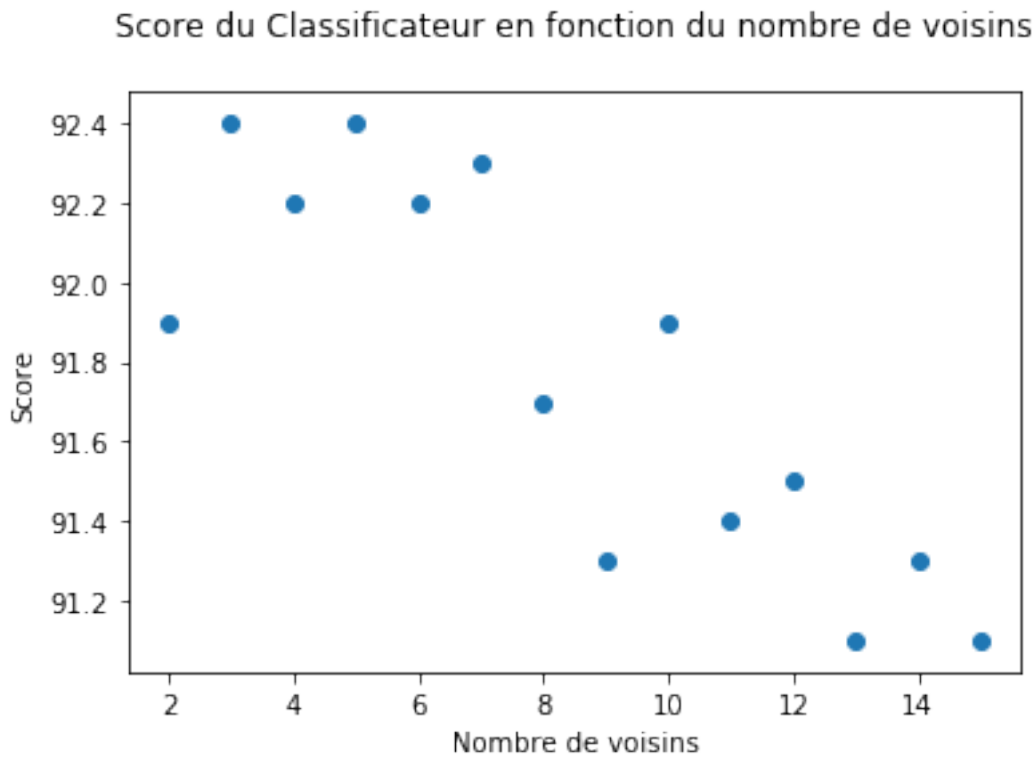
```
Le classificateur prédit: ['9'], et le réel nombre est: 9
```

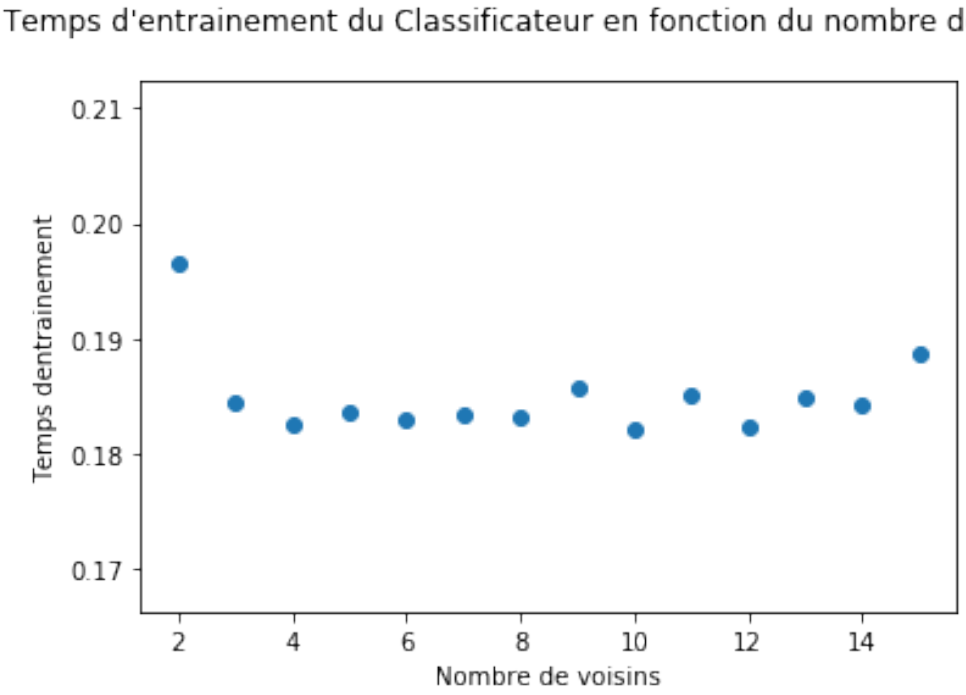
Affiche le score sur l'échantillon de test. Quel est le taux d'erreur sur vos données d'apprentissage ? Est-ce normal ?

```
Score: 91.9
Taux d'erreur: 8.099999999999994

A première vue, le taux d'erreur peut paraitre anormalement grand (environ 8%)
malgré le fait qu'on ait un grand  nombre de voisins (k=10). Mais en réalité, ce
score est tout à fait normal car quand il y a trop de voisins, cela diminue la
précision. On verra cela plus tard, pour détérminer à quel niveau environ se
situe ce seuil.
```

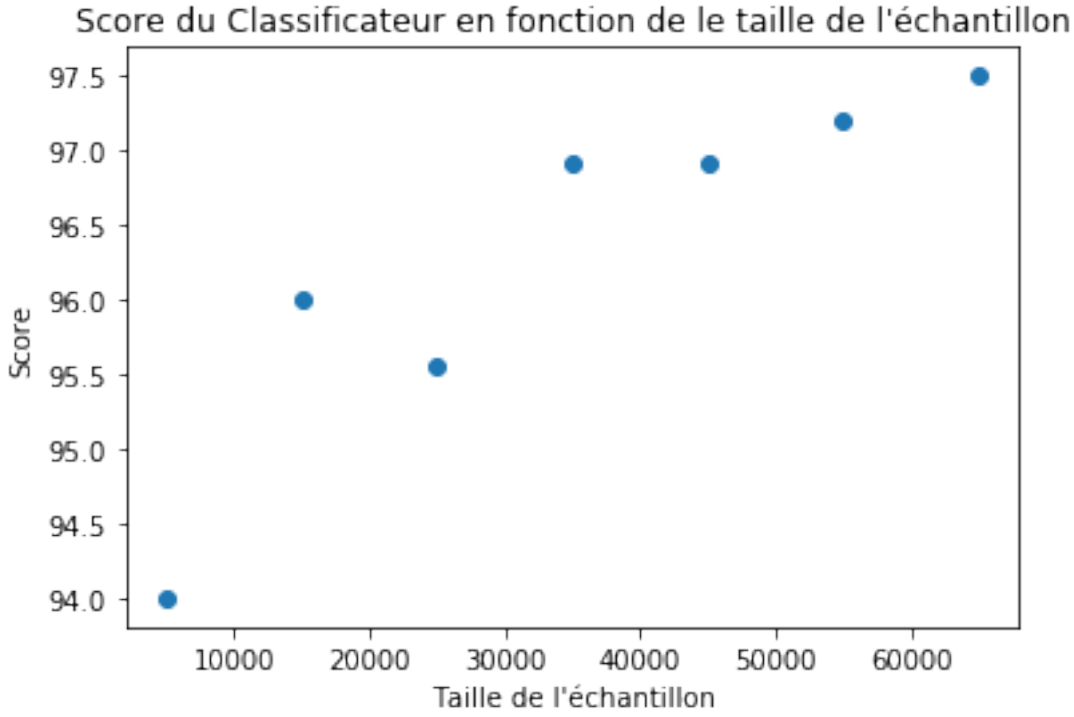
Faites varier le nombre de voisins (k) de 2 jusqu'à 15 et afficher le score. Quel est le k optimal ?

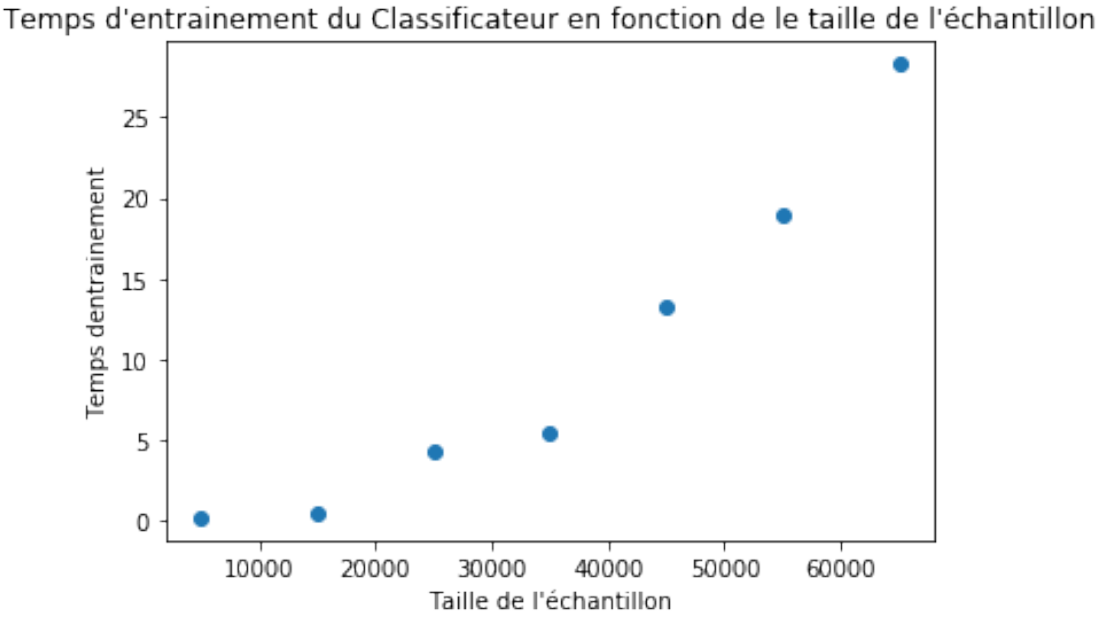

Score du Classificateur en fonction du nombre de voisins
Score
92.4
92.2
92.0
91.8
91.6
91.4
91.2
2
4
6
8
10
12
14
Nombre de voisins


Temps d'entrainement du Classificateur en fonction du nombre de voisins
Temps dentrainement
0.21
0.20
0.19
0.18
0.17
2
4
6
8
10
12
14
Nombre de voisins

`k est optimal pour les valeurs 4 et 5.`

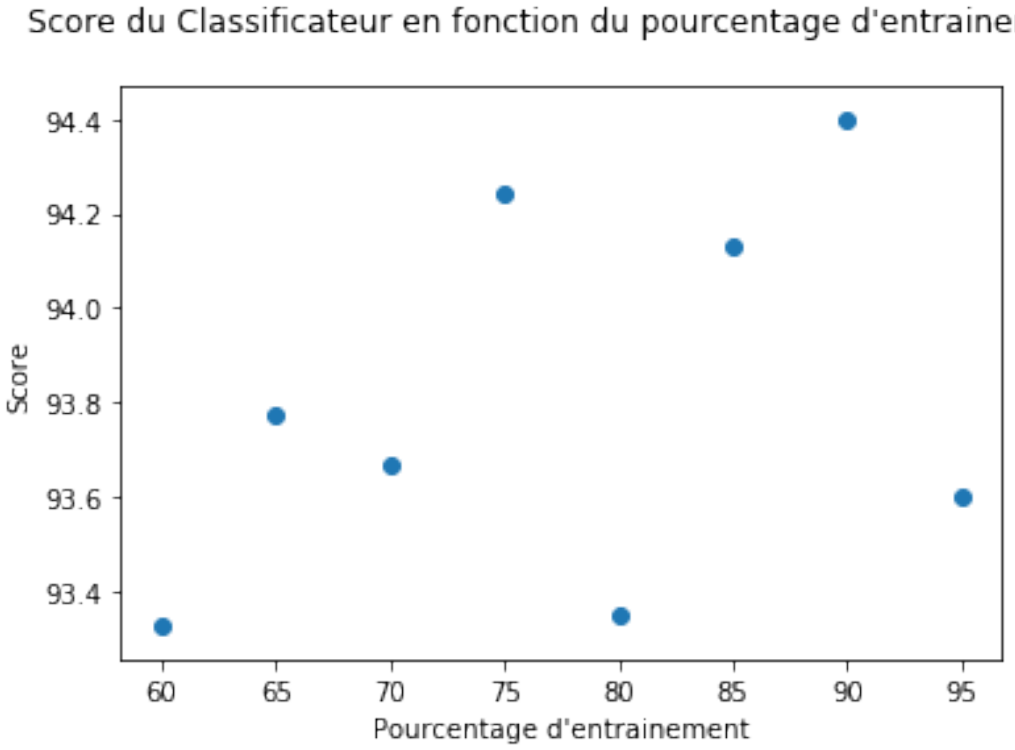
Faites varier le pourcentage des échantillons (training et test) et afficher le score. Quel est le pourcentage remarquable ?

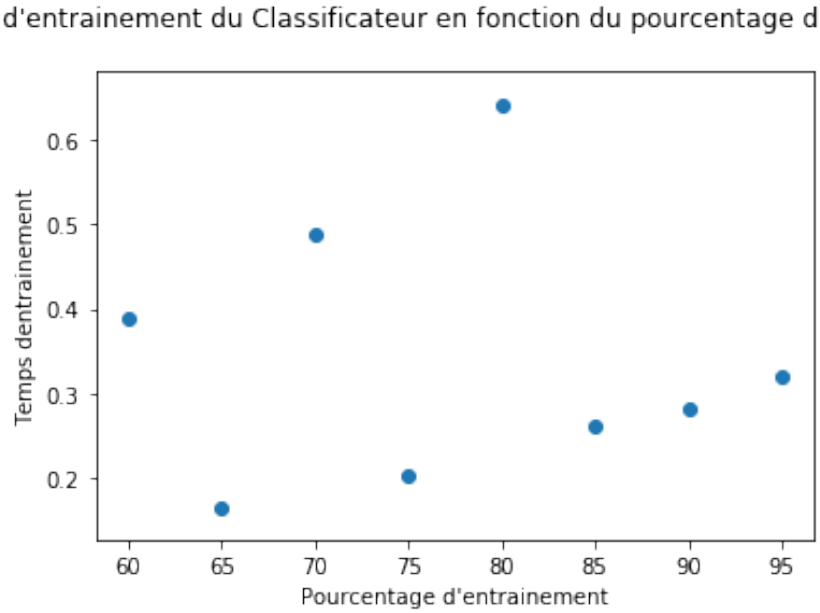




On remarque que plus l'échantillon est grand, plus le classificateur met de temps, surtout à partir des échantillons supérieurs à 2500. On remarque aussi que plus la taille de l'échantillon est grande, plus le score est meilleure, et on se rapproche des 100%

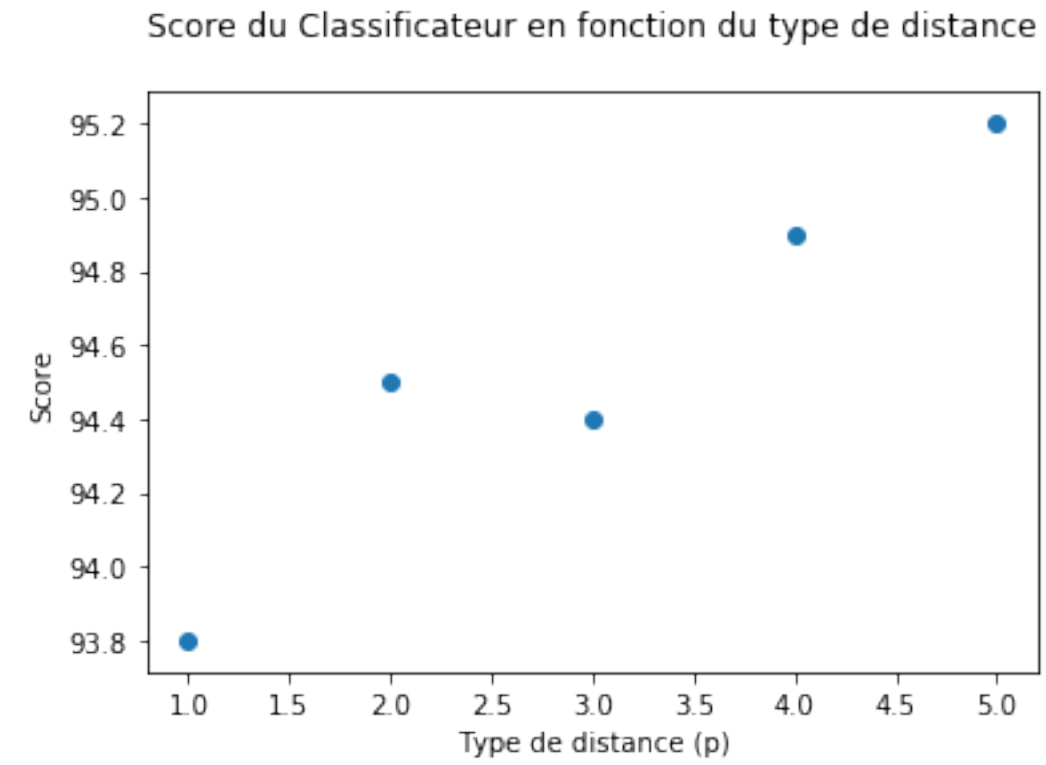
Faites varier la taille de l'échantillon training et afficher la précision. Qu'est-ce que vous remarquez ?

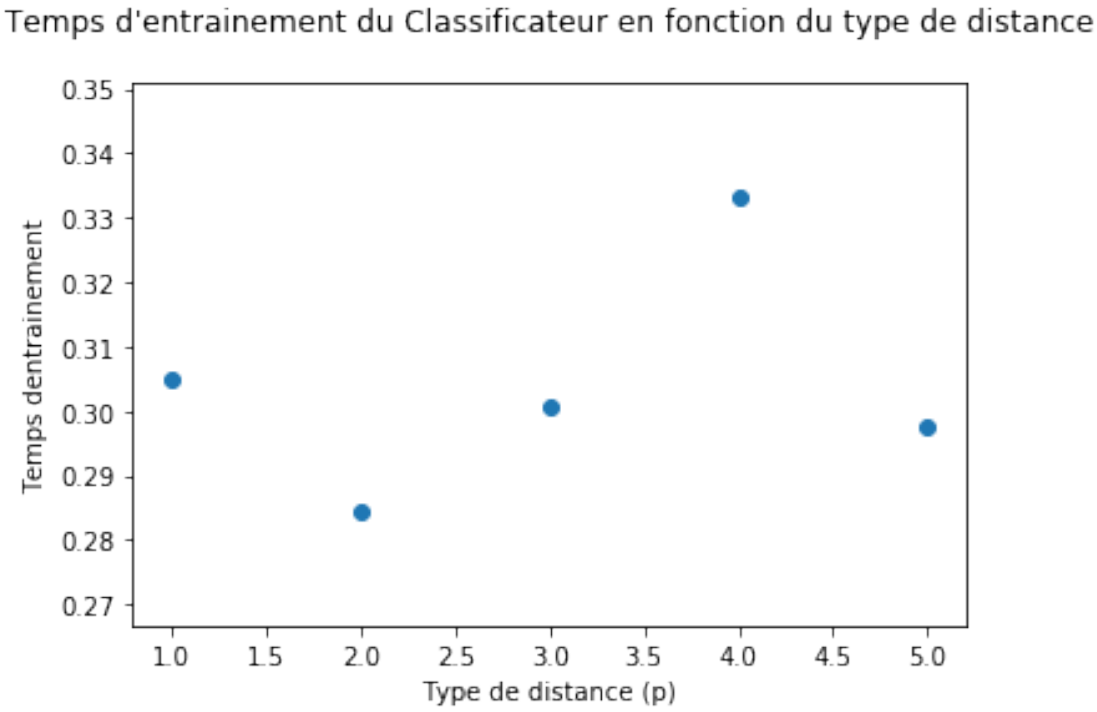




On remarque le max est atteint avec 90% d'entraînement et 10% de test et que au delà le score diminue, certainement car il n'y a quasiment plus de part de test

Faites varier les types de distances (p). Quelle est la meilleure distance ?


Score du Classificateur en fonction du type de distance
Score
95.2
95.0
94.8
94.6
94.4
94.2
94.0
93.8
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Type de distance (p)


Temps d'entrainement du Classificateur en fonction du type de distance
Temps dentrainement
0.35
0.34
0.33
0.32
0.31
0.30
0.29
0.28
0.27
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Type de distance (p)

La mesure Euclidienne apporte de meilleures résultat que la distance de Manhattan. Les métriques suppérieurs à 3 offrent de meilleurs résultats que les 2 précédentes mais on ne connaît pas leurs natures.

Fixer n_job à 1 puis à -1 et calculer le temps dans chacun

n_jobs= -1
score: 92.60000000000001
error: 7.3999999999999915

n_jobs= 1
score: 92.60000000000001
error: 7.3999999999999915

A votre avis, quels sont les avantages et les inconvénients des k-nn : optimalité ? temps de calcul ? passage à l'échelle ?

Avantages:

- Possibilité de réaliser des prédictions
- Algorithme simple
- Adapté aux modèles avec des frontières sont irrégulières, comme des chiffres dans notre cas

Inconvénients:

- Non adapté aux données massives
- Obligation de stocker tous l'apprentissage en mémoire
- Prédiction plutôt lente, car nécessité de calculer les distances pour chaque points