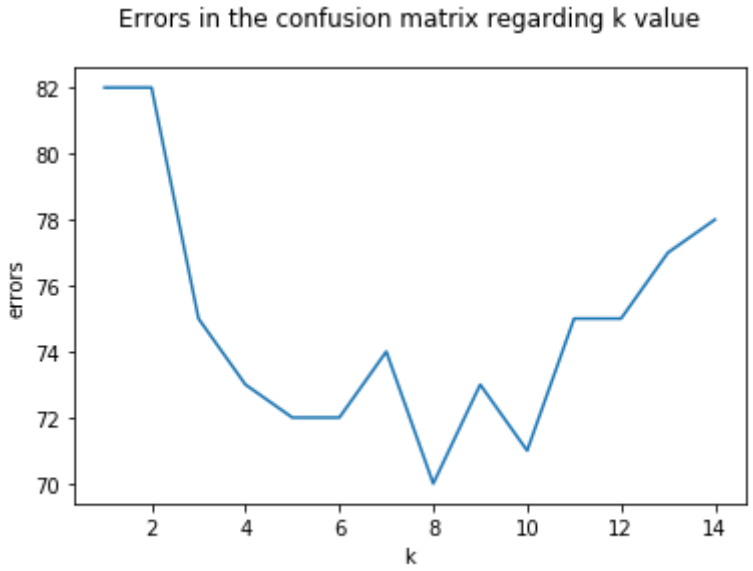
Rapport de code - Classification

Revue de code

Le code est composé de deux scripts, un script permettant de définir les labels d'un set de données non labelisées en s'appuyant sur une valeur précise de 'k', et un deuxième script prenant deux datasets labelisés et calcul le pourcentage de précision obtenu selon chaque 'k' en faisant changer les valeurs de celui-ci dans un range.

Toutefois, dû au grand nombre de data dans le dataset l'opération prend énormément de temps et on ne peut pas tester tous les k allant de 1 au nombre de donnée totale – 1. Mais sachant qu'il n'est de toute façon pas pertinent de prendre un 'k' important, on commence par évaluer les 14 premières valeurs.




```
Best K :  8
Accuracy : 88.33 %
```

Il semblerait qu'on atteint un minimum d'erreur pour k = 8 obtenant ainsi un taux de précision de 88.33 %. Le taux d'erreur repart ensuite à la hausse et on suppose alors qu'il ne sera pas meilleur que 8. Ce sera donc la valeur de 'k' que l'on retiendra.

Par mauvaise gestion du temps je n'ai pu intégrer la matrice de confusion réalisé dans le premier TD à ce dataset.

Afin de trouver le label d'une donnée nous allons d'abord chercher à trouver ses k plus proche voisins avec la méthode k_selection( ), puis déduire le label selon celui qui est le plus récurent dans la liste des k plus proche avec getlabel( ).

La sélection des k plus proches données se fait avec un simple calcul euclidien, cette distance est alors associée à la donnée et nous retournons les plus proches par un simple tri en prenant les k premiers.

Axes d'améliorations

Il y'a 4 type de calcul de distance utilisées dans les espaces vectoriels :

- Distance Euclidienne

C'est celle qui est utilisée ici et permet d'avoir bon compromis entre la gestion des valeurs faibles et celle des valeurs élevées.

→ On aurait pu associer à notre distance euclidienne une normalisation qui aurait permis d'accorder moins d'importance aux valeurs extrêmes s'il y en avait.

- Distance de Manhattan

Méthode de calcul plutôt intéressante pour un espace avec beaucoup de dimension, qui lui permet alors de converger plus rapidement, ce qui n'est pas le cas ici. Par ailleurs, cette méthode accorde moins d'importance aux valeurs extrêmes.

- Distance de Minkowski avec p>0

Généralisation des distances Euclidienne et de Manhattan qui donnent de l'emphase aux valeurs élevées par rapport aux autres valeurs.

- Distance fractionnaire

A l'inverse de la distance de Minkowski, celle-ci atténue les valeurs extrêmes. De même que la distance de Manhattan elle a une meilleure convergence dans des espaces à hautes dimensions.

Par ailleurs, pour éviter le surapprentissage il aurait été intéressant de tester nos valeurs de 'k' sur différents datasets de test et observer s'il est toujours pertinent ou non de garder 8. Ou bien il est possible aussi (mais a peut-être moins d'impact) de diviser notre dataset de test en plusieurs datasets, de même que prendre une portion (20%) du data training et faire varier les valeurs de k dessus.