|  |  |
| --- | --- |
|  | Sullivan Honnet  Jules Vittone |

|  |
| --- |
| Machine Learning  2ème Année Polytech Paris-Sud – Cycle Ingénieur  18 mars 2019 |

|  |
| --- |
| Projet de Machine Learning |

|  |
| --- |
| Polytech Paris-Sud  Maison de l’ingénieur 620 Centre scientifique d’Orsay 91405 Orsay France |

**Question 1 :**

Pour la méthode du plus proche, on commence par calculer la position des barycentres des différentes classes. On calcule ensuite la distance entre les différents barycentres et le point dont on cherche à connaître. On lui attribue ensuite pour classe celle du barycentre dont il est le plus proche.

Le taux d'erreur du plus proche est 32.42%.

Le temps d'exécution est de 0.312 secondes.

**Question 2 :**

Pour la question 2, on utilise les fonctions et les méthodes que l’on a créé pour la question précédente mais on applique aux données en entrée une PCA (c’est-à-dire que l’on réduit le niveau de détails des données entrantes) qui permet d’accélérer le temps d’exécution mais qui réduit la précision des résultats. Voici les taux d’erreur et le temps d’exécution pour différents niveaux de précision.

Pour 95.0 % :

Taux d’erreur : 32.44

Temps d'exécution : 1.85222220421

Pour 75.0 % :

Taux d’erreur : 33.76

Temps d'exécution : 1.72505497932

Pour 25.0 % :

Taux d’erreur : 72.56

Temps d'exécution : 1.69915890694

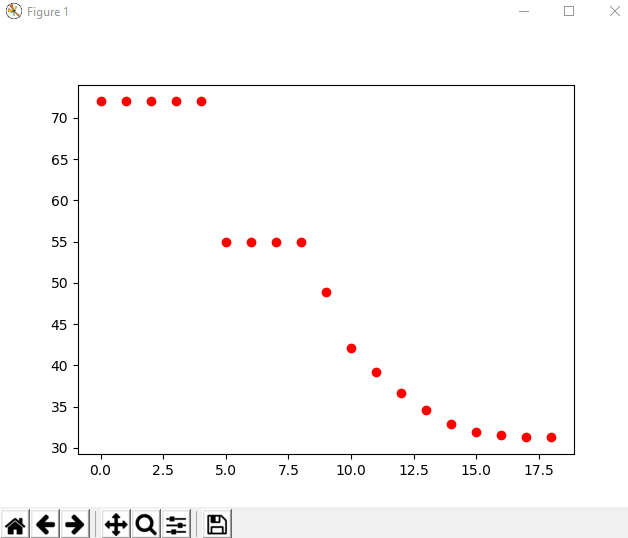


Figure 1 : Evolution du taux d'erreur en fonction du taux de PCA

**Question 3 :**

Pour cette partie, on s’intéresse aux performances de différents classifieurs de la librairie Scikit-Learn, en particulier les classifieurs pour le SVM et les plus proches voisins. Pour la SVM nous avons utilisé SVC comme classifieur avec comme performance :

Taux d'erreur du SVM avec SVC :

Taux d’erreur : 89.76

Temps d'exécution : 244.987223148

Le temps d’exécution est justifié car la méthode fit de SVC à une complexité quadratique et à des difficultés avec des jeux de données supérieur à 10000 données.

Pour les plus proches voisins, nous avons utilisé le classifieur Nearest Neighbors, avec lequel on peut modifier le nombre de points voisins que l’on considère pour estimer la classe de notre donnée. Voici les performances des différents tests réalisés :

Taux d'erreur du plus proche voisin :

Taux d’erreur : 18.34

Temps d'exécution : 64.2929568291

Taux d'erreur des 5 plus proches voisins :

Taux d’erreur : 17.02

Temps d'exécution : 72.1514778137

Taux d'erreur des 10 plus proches voisins :

Taux d’erreur : 16.98

Temps d'exécution : 95.6245448589

Taux d'erreur des 20 plus proches voisins :

Taux d’erreur : 18.48

Temps d'exécution : 90.3468580246

Taux d'erreur des 100 plus proches voisins :

Taux d’erreur : 22.04

Temps d'exécution : 81.2785902023

Taux d'erreur des 1000 plus proches voisins :

Taux d’erreur : 31.22

Temps d'exécution : 83.855672121

**Comparaison des modèles :**

Si on compare la méthode du plus proche centre avec et sans la PCA, on constate que la précision et le temps d’exécution son meilleur sans la PCA. Pour la précision c’était attendu car la PCA réduit le champ des données pour permettre au programme d’être plus rapide. Pour le temps de calcul, cela s’explique par la faible complexité du traitement et la petite quantité de données traitées. En effet, la PCA est une opération relativement coûteuse qui n’est intéressante que si le calcul qui suit est encore plus coûteux ce qui n’est pas le cas ici.

Si on compare ensuite les différents modèles entre eux, on peut observer que la méthode des plus proches voisins est la meilleure. En effet, la SVM est particulièrement inefficace en revanche, il est intéressant de noter que les classes sont plutôt mélangées et ne forme pas de gros amas très détachés comme le montre la comparaison entre le plus proche voisin et le plus proche du centre. En effet, s’ils étaient concentrés tous dans une zone en fonction de leur classe alors les résultats des deux méthodes devraient être presque identiques et nous trouvons presque 15% de différence entre les taux d’erreur.