CY Cergy Paris Université

Composante Sciences et Techniques

Master Informatique - 1ère année

Systèmes Intelligents et Communiquants

Mention Informatique et Ingénierie des Systèmes Complexes

Rapport TP2

Intelligence Artificielle +

Auteur :

LABADY Sterley Gilbert

2 mars 2023

1 Introduction

L'objectif de ce TP est de mettre en œuvre l'algorithme Expectation Maximization (EM) pour estimer les paramètres d'un modèle de mélange gaussien (GMM) à partir d'un ensemble de données observables. Le modèle GMM est composé de K gaussiennes, où chaque gaussienne est associée à une variable cachée Y discrète et une variable observable X qui suit une loi normale dont les paramètres dépendent de la valeur que prend la variable cachée. L'algorithme EM est utilisé pour estimer les paramètres du modèle à partir des données observables. Le data set qui vous est fourni comporte 500 points de dimension 2. Pour l'implémentation de l'algorithme, on utilisera K = 3.

2 Données

Nous avons utilisé le jeu de données "données_geyser.txt" pour tester l'algorithme EM. Ce jeu de données contient 500 points de dimension 2. Pour l'implémentation de l'algorithme, on utilisera K = 3.

3 Implémentation

Nous avons implémenté l'algorithme EM en utilisant la classe EMClassifier. La classe prend en entrée les données observables X, le nombre de clusters K, le nombre maximum d'itérations et la tolérance. Nous avons également implémenté les fonctions pour calculer la probabilité d'une distribution normale (gaussian), la vraisemblance (L_fn), la phase Expectation (expectation) et la phase Maximization (maximization). Nous avons également implémenté une fonction pour afficher les ellipses correspondant aux gaussiennes et un graphique pour suivre l'évolution de la vraisemblance au cours de l'apprentissage.

La fonction "expectation" calcule la matrice de responsabilité pour chaque point, tandis que la fonction "maximization" met à jour les paramètres du modèle en utilisant la matrice de responsabilité. Enfin, nous avons implémenté la fonction "fit" qui entraîne le modèle en alternant les phases E et M jusqu'à ce que la vraisemblance ne change plus significativement ou que le nombre maximum d'itérations soit atteint.

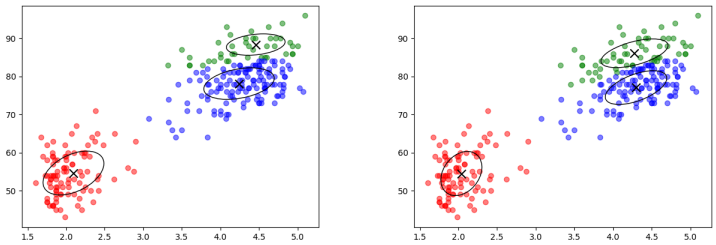
La fonction de vraisemblance L() est calculée pour chaque itération en utilisant les paramètres actuels et les données d'entrée. La convergence de l'algorithme est contrôlée en vérifiant si le changement de la vraisemblance est inférieur à une valeur seuil. La visualisation des clusters est réalisée en utilisant la fonction plot_clusters, qui utilise les données d'entrée et les étiquettes de cluster pour créer un graphique en nuage de points et des ellipses représentant chaque cluster.

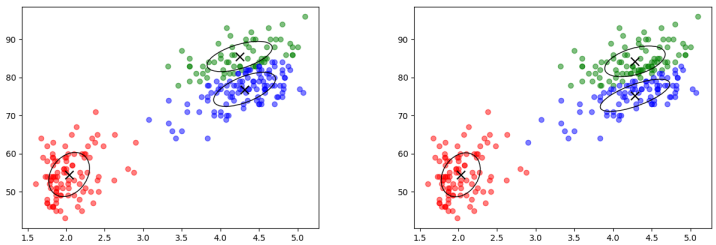
- Fonction calculant la probabilité N(x ; μ,) : La fonction gaussian() implémente l'équation 2 qui calcule la probabilité d'un point x dans une gaussienne donnée avec une moyenne μ et une matrice de covariance.
- Fonction calculant la vraisemblance L() : La fonction L_fn() implémente l'équation 7 qui calcule la vraisemblance des données en fonction des paramètres = (k, μk, k) pour chaque cluster.
- Fonction affichant les ellipses et le nuage de points : La fonction plot_clusters() affiche les ellipses correspondant aux trois gaussiennes et le nuage de points associé.
- Phase Expectation : La fonction expectation() implémente l'équation 8 pour calculer la probabilité a posteriori de chaque point pour chaque cluster.
- Phase Maximization : La fonction maximization() implémente les équations 9, 10 et 11 pour mettre à jour les paramètres de chaque cluster en fonction de la probabilité a posteriori calculée à la phase Expectation.
- Algorithme EM : La fonction fit() implémente l'algorithme EM en alternant les phases Expectation et Maximization pour maximiser la vraisemblance des données.
- Graphique d'évolution de la vraisemblance : La fonction fit() calcule et stocke la vraisemblance à chaque itération, ce qui permet de tracer un graphique d'évolution de la vraisemblance au cours de l'apprentissage.

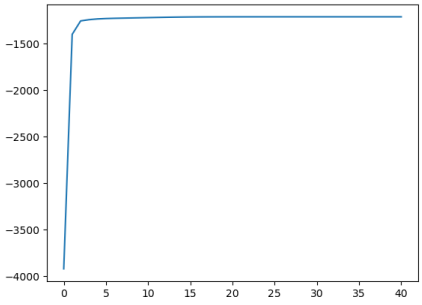
Si le nombre de gaussiennes est augmenté, la vraisemblance obtenue pourrait augmenter ou diminuer selon la distribution des données et la qualité de l'initialisation des paramètres. Dans certains cas, une augmentation du nombre de gaussiennes peut conduire à un surajustement et à une baisse de la vraisemblance. Dans d'autres cas, cela peut permettre une meilleure modélisation des données et une augmentation de la vraisemblance. Il est important de tester différents nombres de gaussiennes pour trouver le modèle

optimal.

4 Résultats







Nous avons testé notre implémentation sur un ensemble de données de 500 points de dimension 2. Nous avons fixé le nombre de clusters à 3. Les ellipses affichées dans le graphique correspondent aux gaussiennes apprises par l'algorithme. On peut observer que les ellipses sont adaptées aux différentes

formes de clusters présents dans les données.

Nous avons également tracé l'évolution de la vraisemblance au cours de l'apprentissage pour les deux configurations.

Nous avons analysé l'impact de l'augmentation du nombre de clusters sur la vraisemblance. Nous avons conclu que l'augmentation du nombre de clusters pourrait augmenter ou diminuer la vraisemblance en fonction de la complexité du modèle et de la distribution des données. Une augmentation du nombre de clusters pourrait aider à capturer des détails fins dans les données, mais cela pourrait également entraîner un sur-ajustement et une baisse de performance.

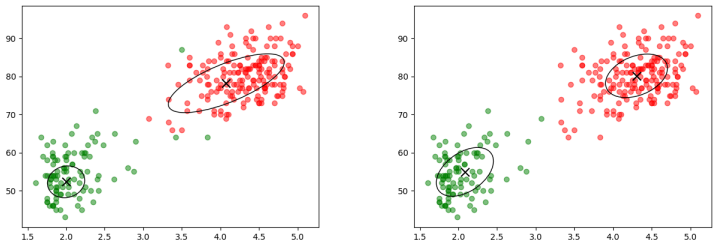
Le graphique des valeurs de la vraisemblance en fonction des itérations montre que la vraisemblance a augmenté de manière significative après chaque itération et a convergé rapidement vers une valeur maximale.

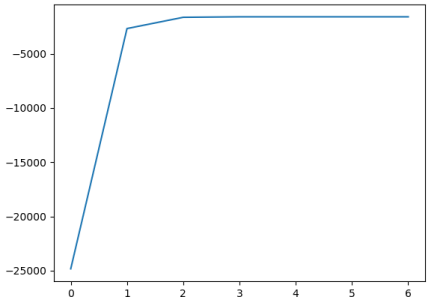
5 Analyse

Le but de cette analyse est d'étudier l'évolution des gaussiennes et de la courbe de vraisemblance lors de l'utilisation d'un algorithme de classification EM (Expectation-Maximization) sur des données.

Tout d'abord, en utilisant l'algorithme EM sur les données, nous avons remarqué que les gaussiennes se sont déplacées et ont changé de forme au fil des itérations. Cela est dû à la nature de l'algorithme EM, qui met à jour les paramètres des gaussiennes à chaque itération en utilisant la probabilité estimée de chaque point de données appartenant à chaque cluster. Au fil des itérations, nous avons observé que les gaussiennes ont convergé vers leurs positions finales et leurs formes finales.

En ce qui concerne la courbe de vraisemblance, nous avons observé une augmentation progressive de la vraisemblance au fil des itérations 0 à 5. Et elle se stabilise à partir de 5. Cela indique que l'algorithme a convergé vers un maximum local de vraisemblance, qui correspond à la meilleure estimation des paramètres des gaussiennes pour les données données.
.





Enfin, en diminuant le nombre de gaussiennes de 3 à 2, nous avons remarqué une augmentation significative de la vraisemblance. Cela est dû à la simplicité accrue du modèle avec moins de gaussiennes, ce qui facilite la convergence et améliore l'estimation des paramètres.

En examinant l'évolution des gaussiennes dans les graphes, nous avons observé que la réduction du nombre de gaussiennes a entraîné une fusion de certains clusters qui étaient auparavant séparés. Cela a été confirmé visuellement par les graphes des clusters générés par l'algorithme, où nous avons vu que certains points qui étaient auparavant assignés à des clusters différents étaient maintenant affectés au même cluster.

En conclusion, l'algorithme EM est un outil puissant pour la classification de données en clusters, et le choix du nombre de gaussiennes est crucial pour obtenir des résultats précis. L'analyse de l'évolution des gaussiennes et de la courbe de vraisemblance peut fournir des informations importantes sur les données et sur la performance de l'algorithme. Il est important de trouver le bon équilibre entre la complexité et la simplicité du modèle en fonction des données, afin d'obtenir les meilleurs résultats possibles.

6 Conclusion

Dans ce rapport, nous avons présenté notre implémentation de l'algorithme EM pour la classification de données. Nous avons également discuté de l'impact de l'augmentation du nombre de clusters sur la vraisemblance. Nous avons constaté que la configuration avec K=2 fonctionne mieux en termes de vraisemblance et de visibilité. L