Projet Model-Based Learning
Classification par modèle de mélange

Lia Furtado et Hugo Vinson
Professeur: Julien Jacques

Université Lumière Lyon 2, France
lia.furtado@univ-lyon2.fr
hugo.vinson@univ-lyon2.fr

1 Introduction

Dans le cadre de l'enseignement de Mode-Based Learning du Master 2 Informatique mention Data Mining, nous proposons un package R permettant de réaliser une classification. Durant l'enseignement nous avons commencé à implémenter l'algorithme EM qui a servit de base pour cet algorithme. Nous avons donc poursuivit l'implémentation de ce dernier pour en faire un module de la méthode de classification. Une fois implémentez nous devions tester les performances sur un jeux de donnée classique puis sur un échantillon des données Mnist. Dans ce rapport, nous présenterons le principe de l'algorithme de classification puis présenterons les résultats de ce dernier sur les données IRIS et MNIST.

2 Classification par modèle de mélange

Le principe de base de l'algorithme repose sur le modèle de mélange. L'objectif va être de générer un modèle qui correspond aux différentes classe afin de pouvoir calculer la probabilité d'appartenance à une classe sachant les données: Contrairement à un modèle de mélange type Gaussien, ici nous allons utiliser un second modèle de mélange pour chaque classe afin de construire le modèle. Dans le premier cas il s'agit d'un problème de classification et dans le second de clustering. L'objectif est alors de construire un modèle de mélange pour chaque classe à l'aide de l'algorithme EM afin d'estimer les paramètres de ces derniers. Pour trouver le meilleur modèle nous allons construire différents modèles avec différents nombres de cluster, enfin nous choisissons le meilleure modèle en nous basant sur un critère d'évaluation tiré de la littérature, le critère BIC. Enfin, pour prédire la classe d'une nouvelle observation nous allons comparer les probabilités d'appartenance aux différents modèles et choisir la classe qui correspond au modèle qui maximise cette probabilité.

3 MixtureClass

MixtureClass reprend notre implémentation de l'algorithme décrit ci dessus. Il s'agit d'un package développé en langage R. Il se compose d'une dizaine de fonctions mais seulement deux sont utilisables par l'utilisateur. La première fonction

principale est : *MixtureMixture.train*.
Cette fonction permet de générer les différents modèles qui serviront à prédire la classe. Pour cela, l'utilisateur doit mettre en paramètre de la fonction des données, les classes associées, la liste du nombre de cluster par modèle, le nombre de fois que le programme doit exécuter l'algorithme EM (par nombre de cluster), et un paramètre epsilon qui est le critère d'arrêt de l'algorithme EM.
Voyons maintenant la manière dont fonctionne la boucle d'entraînement. Premièrement les données sont séparées selon leur classe d'appartenance. Pour chaque classe, nous allons générer plusieurs modèle grâce à l'algorithme EM, le nombre est déterminé par le produit entre le nombre maximum de cluster et le nombre maximum de modèle préalablement défini. L'algorithme EM utilise une initialisation qui implique de l'aléatoire afin de générer différents résultats à chaque nouveau modèle. Pour chaque nombre de cluster nous gardons le modèle qui a obtenu la log-vraisemblance maximale. Enfin nous calculons le critère BIC pour sélectionner le modèle finale. Une fois que tous les modèles sont générés, le modèle renvoi les paramètres du modèle et le nombre de cluster du modèle pour chaque classes.
La seconde fonction principal est : *MixtureMixture.predict*.
Cette fonction permet d'émettre les prédictions sur de nouvelles données similaires à celles utilisées lors de l'entraînement. Pour classer les nouvelles observations, la fonction va calculer la densité de l'observation pour chaque modèle, la classe sélectionnée est celle dont le modèle maximise la densité.

4 Résultat expérimental

4.1 Donnée Iris

Pour tester notre algorithme nous avons utiliser un jeu de données connu pour les premiers tests: le jeu de données Iris. Nous avons donc sélectionné, un échantillon de 100 données de manière aléatoire pour entraîner le modèle et générer les différents modèle de mélange pour chaque classes. Nous avons ensuite prédit pour les 50 autres données du jeu, la classe à l'aide des modèles générées. Voici les résultats obtenus :

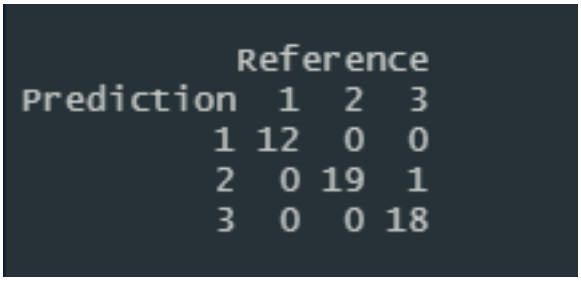

Fig. 1: Résultat de la prédiction en test sur le jeu de données Iris

Model	Accuracy	Precision	Recall
Classification	**0.98**	0.98	0.99

Table 1: Métrique du modèle de classification sur Iris

Nous pouvons voir que la classification est bonne, ce jeu de donnée est réputé pour être difficile de distinguer les classes virginica (2) et versicolor (3). Nous pouvons voir qu'il s'agit des classes les plus dures à prédire pour le modèle. En terme de temps l'apprentissage est assez rapide l'apprentissage s'est réalisé en 10 secondes environs, la prédiction est presque instantanée.

4.2 Donnée MNIST

La base de donnée MNIST est une base de donnée de référence, elle contient environ 70000 images représentées par des vecteurs, chaque vecteurs peut être représentés par une matrice de pixel de 28x28 et représente un chiffre entre 0 et 9. Nous avons donc un jeu de données composé de 720 features. Nous avons donc appliquer notre algorithme sur un échantillon des données, nous avons testé de 10 à 30 clusters. Premièrement, le temps d'exécution est très long, ensuite les résultats ne convergent pas. Nous pensons que cela est en partie du à l'initialisation, en effet l'initialisation offre de meilleure résultats lorsque les données formes des ensembles concaves, car les moyennes se situe au sein de différents potentielle cluster. De plus l'algorithme EM ne produit que des modèles de mélange non parcimonieux ce qui est plus coûteux en temps de calcul.

5 Conclusion

Ce travail consistait en l'implémentation d'un modèle de classification basé sur les modèles de mélange. Nous avons développé un package R afin de partager notre modèle. Nous avons testé l'algorithme sur des jeux de données classiques. Nous avons pu voir que le modèle avait de bons résultats en prédiction, la complexité en temps du modèle est tout de même importante. Nous ne sommes pas parvenu à obtenir de résultats sur les données MNIST. Nous pensons qu'un changement au niveau de l'initialisation ou de la parcimonie du modèle aurait pu aider à l'apprentissage.
Implémenter l'algorithme EM a été un vrai challenge pour nous, suivie de cela nous avons imaginer la boucle d'entraînement basé sur la générations successive de modèle de mélange. cela nous a permis de voir comment combiner différents algorithme pour arriver à un nouveau modèle de classification. Malgré les difficultés un projet tel que celui ci nous a permis de bien comprendre le fonctionnement des modèles de mélange.