Máster en Big Data Analytics

Curso 2017-2018

Introducción al Aprendizaje Automático

Práctica 4

Gaussian Mixture Models

Jon Ander Gómez Adrián

jon@dsic.upv.es

Departament de Sistemes Informàtics i Computació

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Índice

1. Motivación

Las mixturas de distribuciones normales o de Gauss son un de las técnicas más utilizadas para estimación de densidades en el estudio de datos cuando

en su mayoría no están etiquetados y por tanto no se puede realizar un aprendizaje supervisado.

Esta técnica permite identificar/descubrir agrupaciones naturales en conjuntos de datos, cada grupo suele responder a un patrón de comportamiento del proceso objeto de estudio. En otras palabras, sirve para identificar ciertas estructuras en los datos identificado los distintos patrones regulares que puedan existir.

También existe otro tipo de aplicación de esta técnica, se aplica con frecuencia a problemas de clasificación en los que las muestras de una misma clase no se distribuyen exactamente según una Gaussiana, o bien dentro de cada clase se observan varios clústeres naturales. En este caso lo que se hace es estimar un GMM por cada clase para después obtener la densidad de probabilidad de cada clase como la suma de las densidades de probabilidad de cada componente del GMM.

2. Objetivos

2.1. Conceptuales

- Estudiar y utilizar una de las técnicas de *clustering* paramétrico más robustas y que mejor resultados aportan a muchos problemas. El desarrollo de cómo se realiza el aprendizaje mediante el algoritmo EM se verá fuera de esta práctica.

- Aplicar GMM a problemas de clasificación estimando un GMM por cada clase para después recomponer.

2.2. En el manejo de utilidades

- Aprender a utilizar la clase GMM del paquete *sklearn.mixture*. Existen otras más sofisticadas, pero de momento trabajaremos con esta.

3. Tareas

3.1. Tarea 1

A partir del código desarrollado en la práctica anterior, quitar el bucle de validación cruzada y añadir el código para dividir el *training set* en tantos subconjuntos como clases se sepa existen en el *training set*.

3.2. Tarea 2

Crear una lista de objetos de la clase `sklearn.mixture.GMM` para obtener un GMM por cada clase. En principio utilizad 10 componentes por cada GMM.

Añadid el código correspondiente para calcular la densidad de probabilidad de una clase a partir de las densidades de probabilidad de cada muestra con respecto a cada componente del GMM de cada clase.

$$p(x_n, \mathcal{C}_k) = p(\mathcal{C}_k) \sum_{j=1}^{J} p(j)\, p(x_n \mid j)$$

donde J es el número de componentes en el GMM de cada clase, y j es el índice que identifica cada componente.

¿Qué función de la clase `sklearn.mixture.GMM` debe utilizarse para obtener las densidades? Mirad el manual de la clase GMM en la documentación del *Scikit-Learn*. ¿Puede que sea el método `score()`?

Ojo que muchas funciones devuelven el logaritmo de la densidad de probabilidad.

¿Hace falta añadir las probabilidades a priori de cada componente dentro de cada GMM? ¿Dónde está dicha información en el objeto de la clase GMM? ¿O viene ya calculada por la función que utilizamos?

3.3. Tarea 3

Una vez resuelta la tarea 2, realizad un barrido para ver el efecto de variar los siguientes hiper-parámetros:

- el número de componentes de cada GMM,
- el número de componentes para representar cada muestra aplicando PCA, y
- el tipo de matriz de varianzas-covarianzas.

Se debe ir complentando la tabla que se comentó en prácticas anteriores para ver la precisión que puede alcanzarse modificando los hiper-parámetros que afectan al comportamiento de cada técnica.

4. Conclusiones

En esta práctica se ha trabajado con la clase `sklearn.mixture.GMM` en un problema de aprendizaje supervisado, se utiliza un GMM dentro de cada clase para mejorar, en teoría, la estimación de la densidad de probabilidad de cada muestra con respecto de cada clase conocida.

5. Bibliografía

[1] S. authors. Deep Learning. `http://www.deeplearning.net`, 2015. [Online: accessed June 2015].

[2] Y. Bengio, I. J. Goodfellow, and A. Courville. Deep learning. Book in preparation for MIT Press, 2015.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

[5] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Addison-Wesley, second edition, 2000.

[6] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):2–42, 2006.

[7] G. Hinton. A practical guide to training restricted boltzmann machines, 2010.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, JMLR*, 12:2825–2830, 2011.

[9] Wikipedia. Machine Learning. `http://en.wikipedia.org/wiki/Machine_learning`, 2015. [Online: accessed April 2015].