Máster en Big Data Analytics

Curso 2017-2018

Introducción al Aprendizaje Automático

Práctica 7

Funciones Discriminantes Lineales

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

Trabajar con clasificadores basados en Funciones Discriminantes Lineales es un paso previo que ayudará a la comprensión de las *Support Vector Machines* (SVM) y las Redes Neuronales Artificiales (ANN).

Además, es importante destacar que para muchos problemas reales este tipo de clasificadores suelen ser más que suficientes. En particular cuando se realiza cierta transformación sobre el espacio de entrada, de tal manera que el discriminante lineal se aplica sobre muestras cuyas componentes son fruto de transformaciones lineales y/o no lineales sobre las originales, esto se representa como sigue:

$$\phi(\mathbf{x}) = \{\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_m(\mathbf{x}), \ldots, \phi_M(\mathbf{x})\} \tag{1}$$

donde cada $\phi_m()$ realiza una transformación sobre las muestras $\mathbf{x} \in \mathbb{R}^d$, de manera que $\phi(\mathbf{x}) \in \mathbb{R}^M$. $\mathbb{R}^d$ es el espacio de entrada o *feature space* original y $\mathbb{R}^M$ es el *feature space* transformado. M puede ser mucho mayor que d, como es el caso de cuando se expande de manera polinomial, o mucho menor cuando se aplican técnicas de reducción de la dimensionalidad.

2. Objetivos

2.1. Conceptuales

- Entender el funcionamiento de los discriminantes lineales y el efecto positivo de realizar ciertas transformaciones sobre el *feature space* original.
- Comprender el concepto de hiperplano de separación.
- Descubrir la utilidad de los discriminantes lineales para muchas tareas, pues cuando las muestras son separables lineal o polinómicamente, los resultados obtenidos con estos clasificadores no son fácilmente mejorables, y una vez estimados éstos clasifican de manera muy rápida.
- Descubrir como ciertas transformaciones tienen efectos beneficiosos con unas técnicas y desastrosos sobre otras.

2.2. En el manejo de utilidades

- Aprender a utilizar las funcionalidades de *numpy* para operaciones algebraicas con matrices. Ver código de `show-linear-discriminant.py` y `LinearDiscriminant.py`.

- Seguir familiarizándose con la comparación de dos o más técnicas, de clasificación en este caso, sobre un mismo *dataset*.
- Seguir acostumbrándose a comprobar una misma técnica con muestras generadas sintéticamente o frente a distintos *datasets*.

3. Tareas

3.1. Tarea 1

Ejecutad repetidamente

```
python show-linear-discriminant.py
```

Estúdiese el efecto que tiene utilizar la formulación de Fisher para obtener el vector **w** perpendicular al hiperplano de separación.

La primera función obtiene el hiperplano como la recta perpendicular al segmento que une los centroides de las clases. Mediante Fisher se tiene en cuenta la forma en cómo se distribuyen las muestras en cada clase. En el código que calcula el hiperplano según Fisher, S_w es la suma de las matrices de varianzas-covarianzas de cada clase $S_w = \Sigma_1 + \Sigma_2$. m_1 y m_2 son los vectores media (centroides) de cada clase.

Una vez obtenido el vector **w**, este se devuelve con la ecuación general de la recta del tipo $Ax + By + C = 0$.

Para más información sobre el discriminante lineal de Fisher podéis consultar:

`http://en.wikipedia.org/wiki/Linear_discriminant_analysis`

Sólo mediante la observación, ¿cuál consideráis que es mejor manera de determinar el hiperplano de separación entre clases?

3.2. Tarea 2

Ejecutad repetidamente

```
python test-linear-discriminant.py
```

para ver como se comporta un clasificador basado en funciones discriminantes lineales cuando se transforma el *feature space* original. Observad también que a partir de cierto grado del polinomio ya no se obtienen mejoras. Incluso en algunos casos empeora.

La parte del código donde se transforman las muestras de manera polinómica es la siguiente:

```
poly = PolynomialFeatures( degree = degree )
temp_X_train = poly.fit_transform(X_train)
temp_X_test = poly.fit_transform(X_test)
```

Con estas instrucciones se aplican las transformaciones

$$\texttt{temp\_X\_train} = \phi(\texttt{X\_train})$$

y

$$\texttt{temp\_X\_test} = \phi(\texttt{X\_test})$$

Comparad la eficacia de un clasificador de este tipo con uno del tipo KDE, para ello sólo es necesario cambiar a `True` el valor de la variable `compare_with_kde`.

¿Qué efecto tiene la expansión polinómica sobre cada una de las técnicas? ¿Vale la pena modificar el *feature space* original cuando se utilizan clasificadores tipo KDE?

3.3. Tarea 3

Variad los siguientes parámetros para la generación de muestras sintéticas.

Número de clases: (5)

Dimensionalidad: (2)

Número de muestras por clase para *training*: (150)

Número de muestras por clase para *test*: (50)

Dispersión de las clases entre sí: (5,0)

Dispersión de las muestras dentro de cada clase: (2,0) En combinación con el anterior se puede controlar el grado de superposición entre clases.

```
X_train,Y_train,X_test,Y_test = \
generate_datasets.generate_multivariate_normals( 5,
                                                 2,
                                               150,
                                                50,
                                                 5.0,
                                                 2.0 )
```

Tras probar varias combinaciones:

- ¿Qué efecto tiene sobre los KDE el hecho de que las clases estén más o menos superpuestas?
- ¿Y sobre los discriminantes lineales?
- Cuando se aumenta el número de muestras de aprendizaje, ¿cuál de las dos técnicas es más rápida en la fase de predicción?
- ¿Afecta por igual a ambas técnicas que se aumente la dimensionalidad de las muestras en el *feature space* original?
- ¿Qué técnica ve reducida su efectividad cuando aumenta el número de clases y existe superposición entre las clases?

Cuando las clases son linealmente separables ambas técnicas deben dar resultados cercanos al 100 %.

3.4. Tarea 4

Tarea para realizar fuera del horario lectivo, tiempo de ejecución demasiado largo.

A partir de `test-linear-discriminant.py` aplicad ambas técnicas (discriminantes lineales y KDE) al *dataset* IRIS y después al MNIST.

La ejecución en el caso del MNIST será especialmente larga. Para este *dataset* no debe realizarse la transformación polinómica, en todo caso reducción de la dimensionalidad mediante PCA.

4. Conclusiones

Los objetivos de la presente práctica es que el alumno esté familiarizado con los discriminantes lineales como clasificadores y haya experimentado lo suficiente para entender en qué casos son eficientes y en qué se ven fácilmente superados por otras técnicas. Y por supuesto, debe entenderse cuando no es apropiado utilizar un clasificador basado en discriminantes lineales.

También es importante que se comprenda lo fácil que resulta aprovechar los discriminantes lineales tras realizar una transformación no lineal en el *feature space* original.

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].