FLOP Un Laboratorio Libre de Programación

Saltar menú

- Inicio
- Área de prácticas
- Gestión de colecciones
- Área del principiante
- Lenguajes de programación soportados
- Join & us
- Recursos

Geometría Computacional 2016
Práctica 2

- Fecha límite de entrega: viernes 8 de abril de 2016, 23:55.
- Forma de entrega: a través del Campus Virtual
- La práctica se realizará en un módulo python cuyo nombre será `reduce_dim.py`
- La estructura básica del módulo será la que se indica a continuación. No se importará ningún otro módulo o función.

```
from __future__ import division
import numpy as np
from scipy.linalg import eigh

class LDA:

    def fit(self, X_train, y_train, reduced_dim):
        pass #type your code here

    def transform(self, X):
        pass #type your code here

class PCA:

    def fit(self, X_train, reduced_dim):
        pass #type your code here

    def transform(self, X):
        pass #type your code here
```

- La clase `LDA` implementará la reducción de dimensión de puntos multiclase del análisis discriminante lineal, mientras que la clase `PCA` llevará a cabo la reducción de dimensión usando análisis de componentes principales.
- Antes de proyectar, se centrarán los datos usando la media de los **datos de entrenamiento**. Es decir, los vectores que se proyectan serán `X - np.mean(X_train, axis=0)`.
- ~~S_b u = \lambda S_w u forman un array `A` de dimensión `(D, D')` y no son necesariamente ortogonales. Por tanto, es necesario ortonormalizarlos antes de usarlos para proyectar. Esto se puede hacer mediante una descomposición QR de la matriz~~
  ~~`q, r = np.linalg.qr(A).`~~
  ~~La matriz `q` define la proyección ortogonal sobre el subespacio deseado. La matriz `r` es una matriz triangular superior que se puede interpretar como una afinidad y que no utilizaremos.~~
  Corrección: no es conveniente hacer la descomposición QR que se mencionaba. Ejercicio: ¿por qué?
- Los datos de entrenamiento `X_train` y, en su caso, `y_train` serán dos np.arrays de dimensiones `(N, D)` y `(N,)` arbitrarias, respectivamente. El array `y_train` consistirá en una lista de números enteros entre `0` y `K - 1` que especifica a qué clase pertenece cada uno de los puntos de `X_train`.
- Los métodos `transform` de cada una de las clases `LDA` y `PCA` reciben como input `X` un `np.array` de dimensión `(M, D)` y devuelven la proyección como un `np.array` de dimensión `(M, reduced_dim)`.
- El lenguaje de la práctica será el inglés.

- Se valorará la corrección y claridad del código y la velocidad de ejecución.

Archivo Examinar... reduce_dim.py Enviar