UNIVERSIDAD SANTIAGO DE CALI
Facultad de Ingeniería
Programa de Ingeniería Electrónica

GUÍA PRÁCTICA DE LABORATORIO

Tema: Pre-procesamiento de datos e clasificadores KNN y SVM
Curso: Aprendizaje de maquinas
Profesor: Milton Orlando Sarria P.
Correo Electrónico: milton.sarria00@usc.edu.co

Objetivos:

1. Conocer las técnicas comunes para preprocesar datos antes de usar un sistema de clasificación
2. Identificar los conceptos mas relevantes asociados a KNN y SVM

Información Preliminar:

Varianza y covarianza

La varianza de una variable describe que tan dispersos, o alejados de la media, estan los valores de una variable. La covarianza mide el nivel de dependencia entre dos variables. El valor de la covarianza depende de la escala de los valores por lo que es dificil de analizar, en muchos casos se prefiere usar el coeficiente de correlación, que corresponde a la covarianza normalizada.

La matriz de covarianza es una matriz **simetrica semidefinida positiva** (consultar que significa que una matriz simetrica sea semi definida positiva). En general es una matriz que resume una cantidad de información importante sobre un conjunto de variables organizadas en forma de vector.

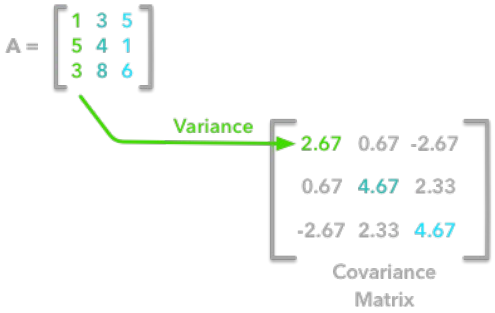

Figura 1.

Considere los datos de la matriz en la Figura 1, y considere la formula para calcular la varianza de una variable que corresponde a un vector columna.

$$V(\boldsymbol{X}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Donde *n* es el numero de datos o la longitud del vector, y $\bar{x}$ es la media del vector. Se puede calcular la varianza de la primer columna como:

$$V(\boldsymbol{A}_{:,1}) = \frac{(1-3)^2 + (5-3)^2 + (3-3)^2}{3} = 2.67$$

Verificar el resultado para las otras dos columnas.

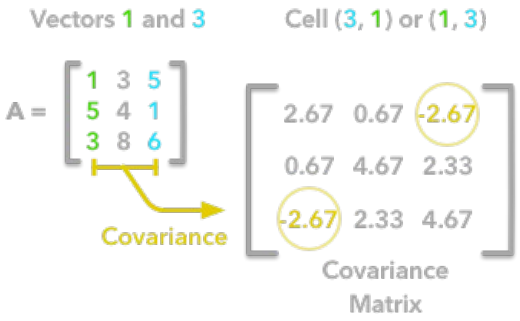

Figura 2

Para calcular la covarianza se toman dos variables. Considere por ejemplo los datos en la Figura 2, para este caso la variable de la primer columna se denota con X y la variable de la tercer columna se denota con Y. Considere la formula de la covarianza:

$$cov(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

Las medias de cada columna son $\bar{x}$ =3, $\bar{y}$ =4

Con estos valores dados, es posible calcular la covarianza entre **X** y **Y**:

$$cov(X, Y) = \frac{(1-3)(5-4) + (5-3)(1-4) + (3-3)(6-4)}{3} = \frac{-8}{3} = -2.67$$

Verificar los valores para las demás posiciones de la matriz.
Como calcular la matriz de covarianzas usando Python?
Primero se debe crear una variable que contenga los datos:

A = np.array([[1, 3, 5], [5, 4, 1], [3, 8, 6]])

Luego se calcula la matriz S usando np.cov():

S= np.cov(A, rowvar=False, bias=True)

Tener en cuenta que en clase se habia hecho uso de la función np.cov(), sin enviando los datos de forma transpuesta, esto se puede evitar enviando un parámetro adicional que es **rowvar=False**, es necesario también notar que se envía el parámetro **bias**, **Consultar porque algunas veces se normaliza sobre n y en otras ocaciones sobre n-1 ?**

Normalización de datos

Inicialmente crear datos de forma aleatoria para ilustrar el proceso

```
np.random.seed(1234)
x1 = np.random.normal(2, 1, 300)
x2 = np.random.normal(1, 3, 300)
X = np.array([a1, a2]).T
X.shape
```

Notar que se crean dos variables aleatorias con distribución normal usando np.random.normal()**,** en este caso se tienen tres parámetros el primero indica el valor medio de cada variable, el segundo la escala o desviación estándar y el tercero el numero de puntos.

También se puede usar la función np.random.seed(1234) que nos permite repetir los números aleatorios. (1234) es un valor que permite crear una semilla, puede ser un valor arbitrario.

Substracción de media:

$$\boldsymbol{X}' = \boldsymbol{X} - \bar{x}$$

Definir una función que permita sustraer la media de los datos, es decir, centralizarlos

```
def center(X):
    Mu =np.mean(X, axis = 0)
    newX = X - Mu
    return newX, Mu
```

#hacemos uso de la función:

```
Xcentered, Mu = center(X)
```

Estandarización:

Corresponde a poner todas las características en una misma escala. En este caso recordar que cada característica es una columna de X. Lo que se hace es dividir cada variable centralizada (característica) por su desviación estándar.

$$\boldsymbol{X}' = \frac{\boldsymbol{X} - \bar{x}}{\sigma_{\boldsymbol{X}}}$$

Se define la siguiente función

```
def standardize(X):
    sigma = np.std(X, axis = 0)
    newX = center(X)/sigma
    return newX, sigma
```

Y se hace uso de la **función**:

```
Xstandar, Mu = standardize(X)
```

Whitening

Este procedimiento busca que se quiere transformar los datos de tal forma que tenga una matriz de covarianzas igual a la matriz identidad (1 sobre la diagonal, 0 en los demás valores). Se llama Whitening en referencia a que cada variable se convierte a ruido blanco.

Lo que se hace es rotar los datos hasta que no haya correlación entre ellos, es decir, que la covarianza entre dos variables sea 0. Luego reescalar para garantizar que la varianza de una variable sea 1.

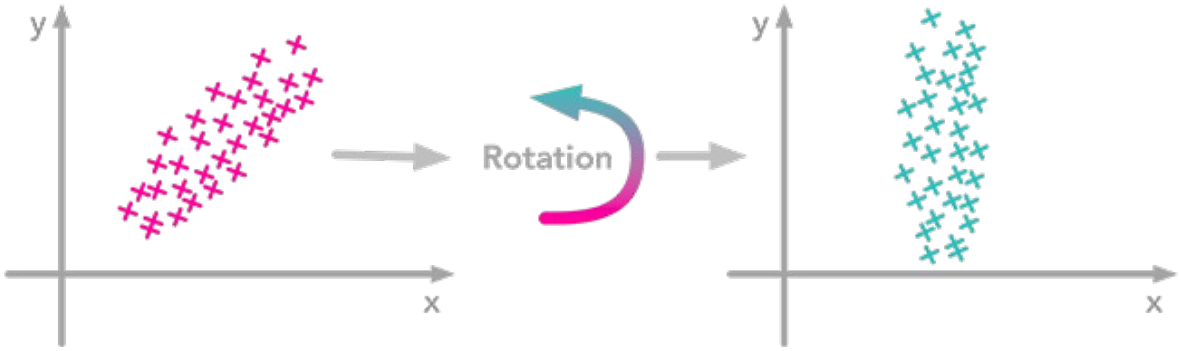

Figura 3.

Para lograr esto es necesario calcular los valores y vectores propios de la matriz de covarianza de los datos ya centralizados.

El algoritmo se puede resumir en tres pasos:

1- Calcular la matriz de covarianzas sobre datos centralizados
2- Calcular los vectores propios de la matriz de covarianzas
3- Aplicar la matriz de vectores propios a la matriz de datos (esto genera la rotacion deseada)

Se define la función que hace el proceso :

```
def whiten(X):
    newX, Mu = center(X)
    cov= np.cov(newX, rowvar=False, bias=True)
    #  Calcular los valores y vectores propios
    eigVals, eigVecs = np.linalg.eig(cov)
```

```
    eigVecs = np.real(eigVecs)
    eigVals = np.real(eigVals)
    # Aplicar los vectores propios a los datos (rotar)
    newX = newX.dot(eigVecs)
    # Re-escalar los datos
    newX = newX / np.sqrt(eigVals + 1e-5)
    return  newX,  Mu,  eigVecs, eigVals
```

Usar la función para procesar datos

```
Xw = whiten(X):
```

Ver el archivo template1.py

Actividad.

1) Familiarizarse con los métodos de pre-procesamiento descritos en la primera parte, para esto, deben ilustrar el resultado de aplicar Estandarización y whitening sobre los datos artificiales que genera el modulo gen_data.py, tanto para 2 como para 3 dimensiones. Gráficar el resultado teniendo en cuenta las clases.

2) Usando la base de datos iris.data y diabetes.data, aplicar Estandarización y whitening. Unicamente para el caso de la base de datos IRIS, ilustrar los resultados mostrando varias combinaciones de dos o tres variables y realizar las graficas. Ver ejemplo template1_data.py.

3) Sin aplicar ningún pre-proceso, separar los datos tanto de IRIS como DIABETES en dos conjuntos: TRAIN y TEST. Implementar un clasificador KNN y variar el numero de vecinos cercanos. Evaluar el clasificador usando unicamente el conjunto TEST y presentar los resultados. (Usar solo vecinos impares)

4) Aplicar whitening a las dos bases de datos y repetir el proceso del punto 3.

5) Repetir el proceso de los puntos 3 y 4 usando un clasificador SVM. Usar kernel lineal, de base radial y polinómico.

6) Usar las imagenes recolectadas durante el ejercicio dejado en clase sobre mano abierta y mano cerrada. Realizar una comparación entre KNN y SVM para clasificar entre las dos clases definidas. Comparar los resultados cuando se aplica ZCA o Whitening. (Ver template2_images.py)

6) En todos los casos medir el tiempo necesario para entrenar y evaluar el clasificador.

Presentar un informe con los resultados y conclusiones de las pruebas realizadas.