Reconocimiento de Patrones
Tarea 6 - Fecha de entrega: 8 de Noviembre

1 Introducción

En este proyecto se aplicará lo visto en el curso para desarrollar un detector de epilepsia a partir de señales de EEG.

Utilizaremos como datos de prueba la base de datos de epilepsia de la Universidad de Bonn [1]. Estos datos pueden descargarse de la siguiente página:

`http://epileptologie-bonn.de/cms/front_content.php?idcat=193`

Estos datos están agrupados en cinco clases:

- Clase Z: EEG superficial de pacientes sanos con ojos abiertos.
- Clase O: EEG superficial de pacientes sanos con ojos cerrados.
- Clase F: EEG intracranial de base de pacientes con epilepsia medido en la zona epileptogénica.
- Clase N: EEG intracranial de base de pacientes con epilepsia medido en el hemisferio opuesto a la zona epileptogénica.
- Clase S: EEG intracranial tomado durante la presencia de actividad epiléptica.

Para cada clase, se tienen 100 registros (señales) de EEG de un solo canal, con una duración de 23.6 segundos (4097 muestras) y una frecuencia de muestreo de 173.61 Hz. El ancho de banda de las señales es de 0.5 Hz a 85 Hz.

2 Extracción de rasgos

Cada registro se encuentra almacenado en un archivo de texto; por lo tanto, para cada clase se tienen 100 archivos (e.g., `Z001.txt` hasta `Z100.txt`), cada uno de los cuales contiene una serie de tiempo de 4097 valores. El primer paso consiste en cargar los datos en Matlab/Octave, lo cual se hace simplemente con la función `load()`. Por ejemplo, para cargar uno de los registros podemos escribir

```
x = load('Z/Z001.txt');
```

de manera que la señal quede almacenada en la variable `x` como un vector.

Aunque en la literatura se han utilizando muchos rasgos para la detección de epilepsia, en este trabajo nos enfocaremos en las características espectrales de las señales. En particular, utilizaremos como rasgos la energía del EEG a distintas bandas de frecuencia. Históricamente, los neurocientíficos han estudiado algunas bandas específicas debido a la relación que han mostrado con distintos estados de atención; estas bandas son:

- Delta (0.1 - 4 Hz)
- Theta (4 - 7 Hz)
- Alfa (7 - 12 Hz)
- Beta (12 - 30 Hz)
- Gamma (30 - 100 Hz)

Por lo tanto, como primer intento podemos construír un clasificador basado en la energía de cada una de estas bandas; es decir, con cinco rasgos. Aunque existen múltiples maneras de estimar la energía de una señal, una de las más simples consiste en definir el espectro de energía como la magnitud al cuadrado de la transformada de Fourier, y tomar como rasgo la integral del espectro de energía en la banda de frecuencia correspondiente. Específicamente, dada una señal discreta $x[n]$, podemos calcular el j-ésimo rasgo y_j como la energía en la rango de frecuencias de ω_j a $\omega j + 1$ dada por

$$y_j = \int_{\omega_j}^{\omega_{j+1}} \|X(\omega)\|^2 \, d\omega, \tag{1}$$

donde $X(\omega)$ es la transformada de Fourier de $x[n]$.

Ejercicios:

1. Escriba una función en Matlab/Octave que tome como entrada un vector x, y que calcule y devuelva un vector columna con los cinco rasgos espectrales que le corresponden a x.
2. Escriba una función de Matlab/Octave que cargue todas las señales correspondientes a una condición o clase dada (e.g., Z o S), que calcule sus vectores de rasgos y los devuelva como una matriz de $M \times N$, donde M es el número de rasgos y N el número de datos (señales). La función debe tomar como entrada una cadena o caracter que indique el juego de datos que se debe cargar (e.g., 'Z' o 'S').

3 Estimación de parámetros

Para simplificar esta etapa, supondremos que los datos de cada clase siguen un modelo de distribución Gaussiano.

Ejercicios:

1. Escriba una función que, dada una matriz de $M \times N$ que representa N datos con M rasgos, calcule y devuelva los estimadores de máxima verosimilitud de la media y matriz de covarianzas.

2. Escriba una función que tome como entrada dos matrices de datos correspondientes a dos clases distintas, y dos índices i y j, y que grafique en un plano las nubes de puntos de ambas clases tomando solamente el i-ésimo y j-ésimo rasgos. Utilice colores distintos para cada clase y grafique también la curvas de nivel de la distribución condicional de cada clase.

3. Cargue las matrices de datos para las clases 'Z' y 'S', y grafique todas las parejas de rasgos utilizando la función anterior. Discuta sobre la validez de asumir modelos Gaussianos para las distribuciones condicionales y sobre el poder discriminativo de los rasgos para el caso en que se desee distinguir solamente estas dos clases.

4 Construcción del clasificador

Como un primer paso, construiremos un dicotomizador (clasificador para dos clases) para las clases 'Z' y 'S'. Supondremos que las probabilidades a priori para cada clase son idénticas; es decir, $p(\mathrm{Z}) = p(\mathrm{S}) = 1/2$. Entonces, podemos simplemente tomar como funciónes discriminantes el negativo de la distancia de Mahalanobis correspondientes a cada clase (ver Ejercicio 1):

$$g_c(x) = -(x - \mu_c)^t \Sigma_c^{-1} (x - \mu_c), \tag{2}$$

donde μ_c y Σ_c son los parámetros de la distribución condicional de la clase c (para $c \in \{\mathrm{Z}, \mathrm{O}, \mathrm{F}, \mathrm{N}, \mathrm{S}\}$).

De esta manera, el clasificador está dado por

$$\alpha(x) = \arg\max_{c \in C} \{g_c(x)\}, \tag{3}$$

donde C es el conjunto de etiquetas de las clases; por ejemplo, para el caso del dicotomizador se tiene $C = \{\mathrm{Z}, \mathrm{S}\}$.

1. Muestre que las ecuaciones 2 y 3 son equivalentes a la regla de decisión Bayesiana que se obtiene al minimizar el riesgo total

$$R = \int R(\alpha(x) \mid x) p(x) dx,$$

dadas las características del problema (distribuciones condicionales Gaussianas y distribución a priori uniforme) y una función de costo de mínima tasa de error $\lambda(\alpha_i \mid \omega_j) = 1 - \delta(i - j)$.

2. Escriba una función que tome como entrada una matriz de datos de tamaño $M \times N$, así como la media y matriz de covarianzas de una distribución Gaussiana, y que calcule, para cada dato, la distancia de Mahalanobis a la media. La función debe devolver un vector renglón de distancias de tamaño N.

3. Escriba una función que tome como parámetros una matriz de datos de tamaño $M \times N$, una matriz μ de $M \times K$, y una matriz tridimensional Σ de tamaño $M \times M \times K$, donde M es el número de rasgos, N es el número de datos, y K es el número de clases. La k-ésima columna de μ representa la media de la k-ésima clase, mientras que la k-ésima rebanada de Σ representa su matriz de covarianzas. La función debe devolver un vector renglón de tamaño N con la salida del clasificador (Eq. 3) correspondiente a cada uno de los datos.

5 Evaluación del clasificador

Una vez construído el clasificador, es necesario evaluar su desempeño. Esto puede hacerse definiendo una función de error que mida qué tanto se equivoca el clasificador dado un conjunto de datos $X = \{x_1, \ldots, x_n\}$ para cada uno de los cuales se conoce la verdadera clase (objetivo) $o(x_i)$. En nuestro caso, podemos simplemente tomar el costo promedio sobre los datos X:

$$E(X) = \frac{1}{|X|} \sum_{x \in X} \lambda(\alpha(x) \mid o(x)) = \frac{1}{|X|} \sum_{x \in X} I(\alpha(x) \neq o(x)), \tag{4}$$

donde $I(\cdot)$ es la función indicadora cuyo valor es 1 si su argumento es verdadero y 0 si es falso; en otras palabras, el error E en este caso es igual a la proporción de datos mal clasificados. Cuando este error se calcula usando el mismo conjunto de datos que se utilizó para la estimación de parámetros, entonces se le conoce como *error de ajuste* o *error de entrenamiento*.

Es posible que un clasificador tenga un muy bajo error de ajuste, pero que muestre un mal desempeño con datos "nuevos" (que no se usaron para la estimación de parámetros). Este fenómeno se conoce como *overfitting* o *sobreajuste*, y por lo general sucede cuando el modelo que describe a los datos es tan complejo que comienza a describir las particularidades del conjunto de dato, en lugar de su comportamiento general. Una manera de detectar si un clasificador está sobreajustado consiste en dividir el conjunto de datos con el que se cuenta en dos grupos: un grupo *de entrenamiento* X_E que se utilizará para estimar los parámetros del modelo, y un grupo *de prueba* X_P que se utilizará para medir el desempeño del clasificador obtenido. Si $E(X_P)$ no es significativamente menor que $E(X_E)$, podemos decir entonces que no existe sobreajuste y la complejidad del modelo es adecuada. Dado un conjunto suficientemente grande de datos, lo más común es tomar entre un 50% y un 80% de ellos como datos de entrenamiento, y el resto como conjunto de prueba.

1. Escriba una función que tome como parámetros dos matrices de datos X y Y de tamaño $M \times N_X$ y $M \times N_Y$, respectivamente, y un valor real p entre 0 y 1. La función debe realizar lo siguiente:

   (a) Dividir cada una de las matrices de datos en una matriz de entrenamiento (con proporción p) y una de prueba (con proporción $1-p$).

Por ejemplo, la matriz X_E estaría formada por las primeras $\lfloor p|X| \rfloor$ de X, mientras que la matriz X_P contendría al resto de las columnas.

(b) Estimar los parámetros de las distribuciones condicionales de X_E y Y_E.

(c) Clasificar los datos en X_P y Y_P con respecto a las distribuciones estimadas en el paso anterior y estimar el error de ajuste E_E.

(d) Clasificar los datos en X_P y Y_P con respecto a las distribuciones estimadas en el paso anterior y estimar el error de prueba E_P.

(e) Devolver el vector $[E_E, E_P]$ como resultado.

2. Utilizando la función anterior, estime los errores de ajuste y prueba cuando la proporción de datos de entrenamiento es del 50%, 60%, 70%, 80%, y 90%. Cuál es la proporción óptima (es decir, la que minimiza el error de prueba)?

3. Escriba una función que tome como entrada una matriz X y devuelva como salida una versión de X con las columnas permutadas de manera aleatoria.

4. Escriba una función que tome como parámetros dos matrices de datos X y Y, la proporción p de datos de entrenamiento, y un número entero n. La función debe llamar n veces a la función implementada en el Ejercicio 1 pasándole permutaciones aleatorias de X y Y en cada iteración, y calcular el promedio de los errores de ajuste y prueba sobre las n iteraciones. La función debe devolver un vector de dos elementos con los errores promedio.

5. Repita el Ejercicio 2, pero ahora estimando los errores promedio con la función implementada en el Ejercicio anterior (utilice $n = 100$).

6 Refinamiento

Aún cuando se obtenga un error de prueba similar al de entrenamiento, es posible que este error sea demasiado alto para lo que la aplicación requiere. En este caso, será necesario revisar nuestra elección de rasgos y/o del modelo de distribución condicional elegido para cada clase. Para el caso de los datos de EEG, las gráficas elaboradas en la Sección 3 pueden ayudar a determinar si el modelo Gaussiano se ajusta más o menos bien a los datos; sin embargo, es posible que los rasgos elegidos no sean los más adecuados para la clasificación.

Algo que sabemos es que las señales de EEG con las que contamos provienen de distintos sujetos, por lo cual pueden existir variaciones significativas en la potencia de la señal que se deban al sujeto, y no tanto a la presencia o ausencia de epilepsia. Por ejemplo, es posible que los electrodos hicieran mejor contacto en un sujeto calvo, que en uno que no lo es. Por otra parte, las señales correspondientes al conjunto S (episodio epiléptico) fueron registradas mediante electrodos intracraniales (colocados directamente en la corteza cerebral), mientras que los

del grupo Z se colocaron en la superficie de la cabeza, por lo cual es de esperarse que existan variaciones en la potencia promedio entre ambos grupos. Entonces, no está claro si nuestro clasificador está distinguiendo entre presencia y ausencia de epilepsia, o entre registros intracraniales o superficiales (esto puede probarse, hasta cierto punto, construyendo un dicotomizador para las clases Z y F).

Por lo tanto, es posible que un nuevo conjunto de rasgos, donde se reduzca la influencia de factores externos que afecten la potencia absoluta de las señales, mejore el desempeño del clasificador. Una manera de lograr esto, es tomar como rasgos la proporción de energía en cada banda de frecuencias con respecto a la energía total de la señal; es decir,

$$y_j = \frac{\int_{\omega_j}^{\omega_{j+1}} \|X(\omega)\|^2 \, d\omega}{\int_0^{\pi} \|X(\omega)\|^2 \, d\omega}. \tag{5}$$

Ejercicios:

1. Modifique la función que calcula los rasgos de cada señal para que calcule los rasgos basados en la energía relativa (Ecuación 5).

2. Recalcule los errores de entrenamiento y de prueba (Sección 5, Ejercicio 5) con los nuevos rasgos y discuta los resultados.

3. Pruebe eliminando algunos de los rasgos (renglones de las matrices de datos) para determinar si existen rasgos que sean redundantes, o que incluso degraden el desempeño del clasificador.

4. Utilice las funciones ya implementadas para diseñar un clasificador que distinga entre las clases Z, F y S, y evalúe el error promedio del clasificador. Elija los rasgos que le parezcan más adecuados y justifique su elección.

References

[1] Andrzejak, R.G., Lehnertz, K., Mormann, F., Rieke, C., David, P., Elger, C.E., 2001. *Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state.* Physical Review E, 64(061907).