# Mentoría Data Science aplicado a BCI

## Consignas TP3: Introducción al Aprendizaje Automático

**Extracción de Características y Armado de Datasets**

A) Introducción:

a) BCIDataset es una clase que nos genera los objetos que son los dataset donde están las señales de la BCI. La clase BCIDataset contiene en primer lugar el método ‘’ínit’’ el cual es un método que construye nuestro dataset y el cual toma los siguientes argumentos:

- csvs_path: Dirección en la cual se encuentran los archivos.
- subject: sujeto que se va a estudiar, puede tomar el valor de todos los sujetos (por defecto el valor que toma son todos los sujetos).
- session: sesión que se va a estudiar, puede tomar el valor de todas las sesiones (por defecto el valor que toma son todas las sesiones).
- channel: canal que se va a estudiar, puede tomar el valor de todos los canales (por defecto el valor que toma son todos los canales).
- overlapping_fraction: desplazamiento que va a realizar la ventana en fracción (por defecto el valor de desplazamiento que toma es de 1/3).
- window_size: tamaño de la ventana en muestras (por defecto el valor que toma es de 900 muestras).
- feature_extractor: función que contiene el extractor de features de los ejemplos (la cual por defecto es la transformada de Fourier).

Cada objeto que se genere con la clase BCIDataset tendrán los siguientes atributos (es decir características que tendrán los objetos de la clase):

- csvs_path = archivos entregados al construir el objeto.
- channel = canal/es.
- parts = 1 / overlapping_fraction.
- fraction = 1 / parts
- ws = tamaño de la ventana en muestras.
- subject = sujeto.
- session = sesión.
- channels = ['ch0','ch1','ch2','ch3'] (lista con los canales que se van a encontrar en el dataset)
- feature_extractor = función que contiene el extractor de features de los ejemplos.
- complete_dataset = función que le otorga formato de dataframe a los datos, el cual contiene todas las muestras en las filas y todos los sujetos, todas las sesiones, y todos los canales en las columnas.

Dentro de la clase BCIDataset además se encuentra definido el método ‘’generate_examples’’ el cual genera los siguientes objetos:

- complete_examples_signal: arreglos que toman como features las muestras.

- complete_examples_features: arreglos que toma como features las muestras de acuerdo a la función de extracción que se les haya aplicado.
- complete_labels: las etiquetas que le corresponde a cada ejemplo.
- complete_metadata: la metadata que le corresponde a cada ejemplo (sujeto, sesión, canal y si un ejemplo es puro o impuro).

Este método lo que realiza en primer lugar es seleccionar del ''complete_dataset" los sujetos y las sesiones escogidas, luego lo que realiza es seleccionar el numero de muestras del dataset de modo tal que se formen cierto número entero de ventanas completas. Posteriormente se generan los ejemplos de las señales totales (multiplicando el número de ventanas que entran en la señal con las partes en la cual se dividen las ventanas), y se ubican las ventanas en la posición que les corresponderían según el orden cronológico (en el caso de que exista solapamiento de ventanas). Finalmente, se le asigna la etiqueta correspondiente a cada ventana, en donde se toma la moda del ejemplo y se le asigna el valor de etiqueta correspondiente (es decir que se le asigna la etiqueta mas presente en cada uno de los ejemplos), se determina si cada ventana es pura o impura (es decir si todos los valores corresponden a una misma etiqueta o existen valores que toman etiquetas diferentes dentro de la misma ventana) y se le aplica la función de extracción en el caso de ''complete_examples_features'', obteniéndose finalmente los dataset con la señal original y con la señal transformada, con sus respectivas etiquetas y metadatos.

Además, posteriormente se encuentran otros métodos definidos dentro de la clase BCIDataset, entre ellos se encuentran:

- get_X_signal: devuelve los valores de los features de los ejemplos (muestras en el dominio del tiempo).
- get_X_features: devuelve los valores de los features de los ejemplos extraídos con la función de extracción.
- get_Y: devuelve las etiquetas de los ejemplos.
- get_metadata: devuelve la metadata de los ejemplos.

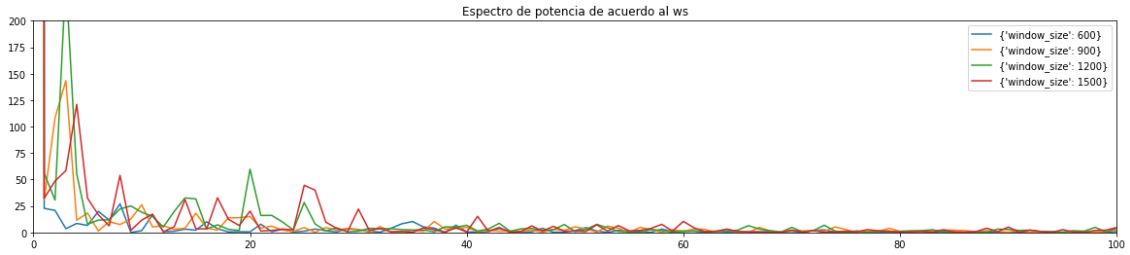
b) Variación del número de ejemplos en el dataset y la dimensión de cada dato según la variación de la ventana de tiempo seleccionada y el criterio de solapamiento.

| Fracción de desplazamiento | Tamaño de la ventana en muestras | Numero de ejemplos del dataset con la señal original | Dimensiones del dataset con la señal origina | Numero de ejemplos del dataset con la señal transformada | Dimensiones del dataset con la señal transformada |
|---|---|---|---|---|---|
| 1 | 600 | 489 | 600 | 489 | 301 |
| 1 | 900 | 326 | 900 | 326 | 451 |
| 1 | 1200 | 244 | 1200 | 244 | 601 |
| 1/2 | 600 | 978 | 600 | 978 | 301 |
| 1/2 | 900 | 652 | 900 | 652 | 451 |
| 1/2 | 1200 | 488 | 1200 | 488 | 601 |
| 1/3 | 600 | 1467 | 600 | 1467 | 301 |
| 1/3 | 900 | 978 | 900 | 978 | 451 |
| 1/3 | 1200 | 732 | 1200 | 732 | 601 |

Se puede observar que al aumentar el tamaño de la ventana en número de muestras disminuye el número de ejemplos en el dataset final, ya que al ser ventanas de mayor tamaño entran un menor número de veces en el dataset original y por lo tanto se obtiene un menor número de ejemplos; y por otra parte, al disminuir el valor de la fracción de solapamiento se puede observar que aumenta el número de muestras debido a que en una misma ventana se van a solapar un mayor numero de veces ventanas subsiguientes, obteniéndose así el doble de ventanas que las originales si el solapamiento es a la mitad, el triple de ventanas que las originales si el solapamiento es a un tercio, etcétera. En relación a la dimensionalidad, esta, esta dada por el número de muestras que poseen las ventanas de tiempo, por lo que la dimensionalidad se modifica de la misma forma que se modifica el tamaño de la ventana en número de muestras.

B) Características Temporales:

a) Influencia que tiene el tamaño de la ventana en el dominio de tiempo en la resolución en frecuencia del espectrograma de potencia:




Al graficar el espectro de potencia de la señal variando los tamaños de las ventanas en número de muestra se puede observar que al aumentar el tamaño de la ventana la resolución de la frecuencia aumenta ya que se tienen más ejemplos por ventana. Pero la resolución en el tiempo disminuye, ya que al aumentar la ventana se generan menos cantidad de ejemplos temporales. La frecuencia de resolución nos indica a partir de qué frecuencia el analizador nos va a mostrar datos, es decir que, el analizador no mostrará nada por debajo de la frecuencia de resolución. La frecuencia de resolución se puede calcular dividiendo la frecuencia de muestreo entre el tamaño de la FFT, donde el tamaño de la FFT es el número de datos que el analizador toma en cada medición (FR=FM/FTT).

| Tamaño de la ventana en muestras | Resolución de la frecuencia |
|---|---|
| 200 | 1.0 |
| 400 | 0.5 |
| 600 | 0.33 |
| 900 | 0.22 |
| 1200 | 0.17 |
| 1500 | 0.13 |

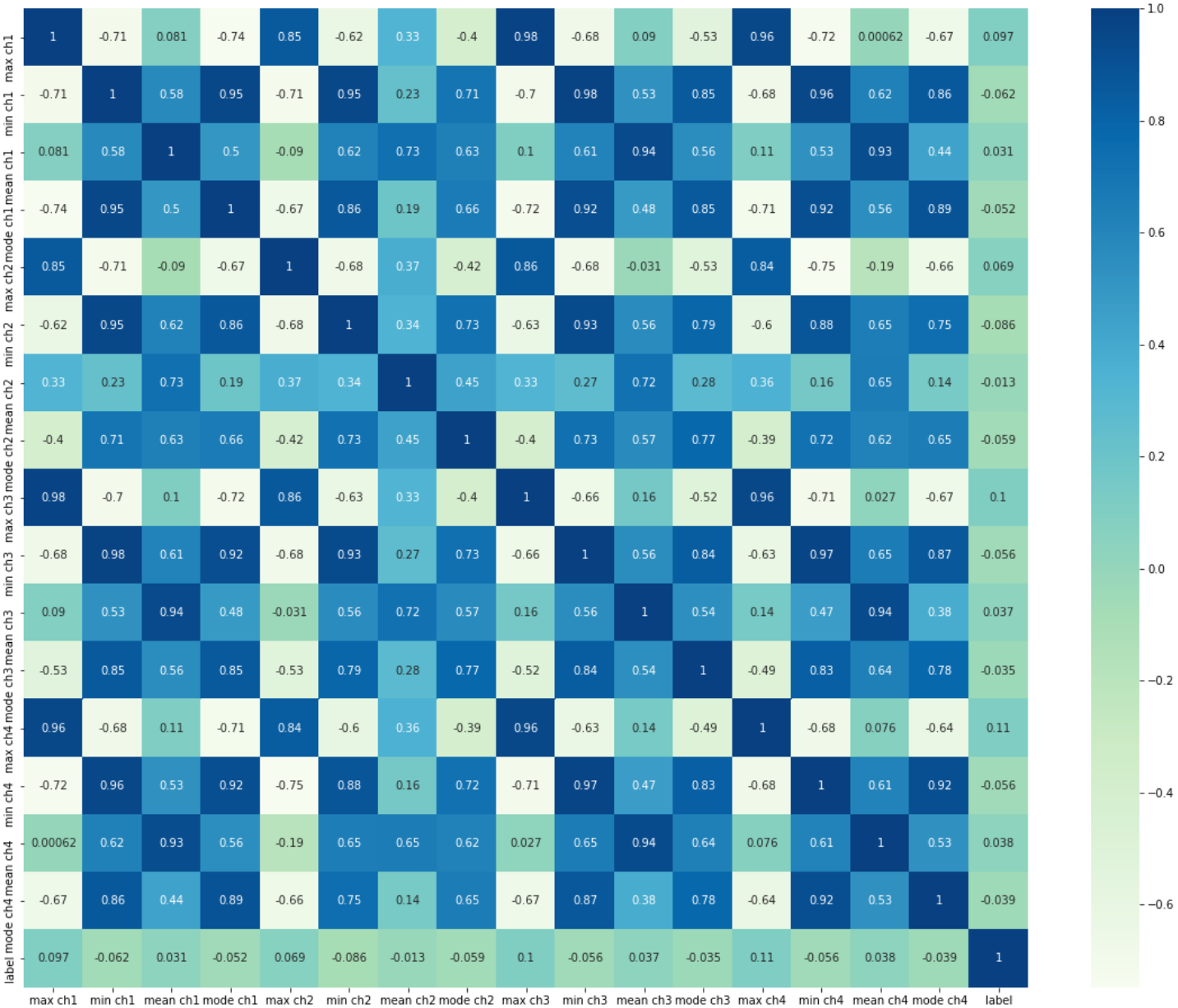
b) Como nuestras frecuencias de muestreo (12.5 Hz y 16.5 Hz) requieren una resolución a partir de 0.5 se considera que el tamaño mínimo que debería tener la ventana en muestras, sería de 400 muestras en adelante.

c) Se generó un dataset al cual se le aplicaron las siguientes estrategias de extracción a las ventanas de la señal cruda: Mínimo, máximo, media, y moda. Todas estas medidas fueron

aplicadas a cada uno de los canales y concatenadas en un mismo dataset. Se decidió utilizar los canales concatenados, para así poder tener más features disponibles para aplicar a un posible modelo.

d) Este dataset fue almacenado en forma de dataframe y en formato csv, denominado "df_time".

e) Heatmap de features de extracción de la señal cruda:




A partir de la gráfica de correlación se puede observar que los features generados no tienen buena correlación con los labels, pero existe una cierta correlación entre ellos.

C) Características espectrales (en frecuencia):

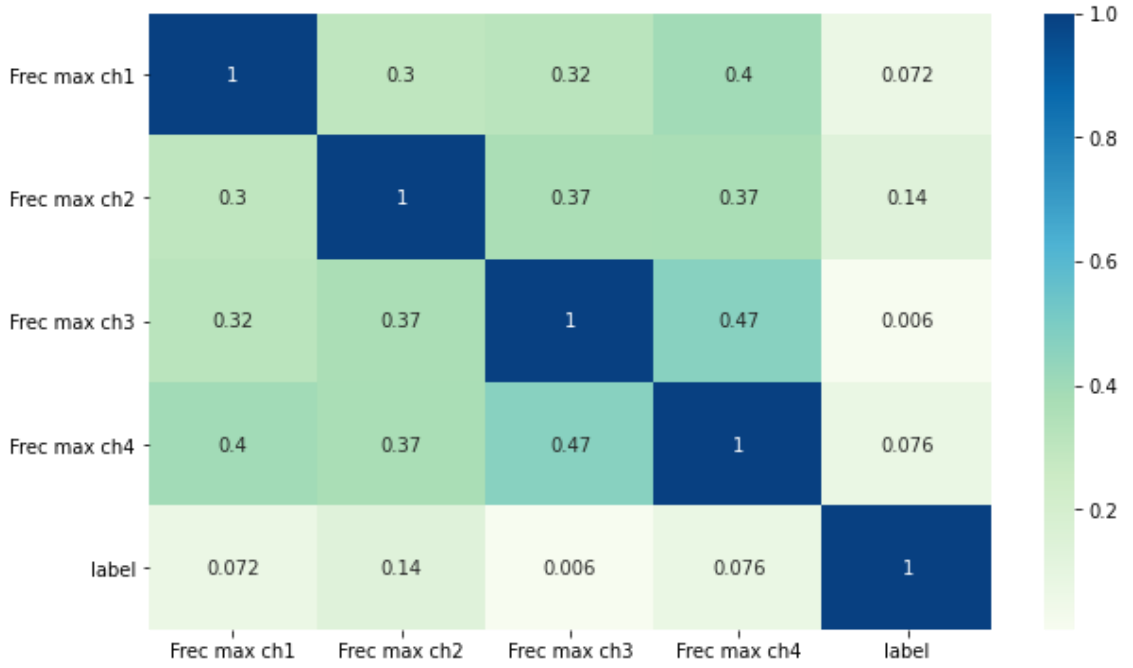
a) Se generó el dataset a partir de la clase BCIDataset utilizando el feature de extracción la función ¨filtered_fft_features¨, la cual le aplica la transformada de Fourier a las ventanas de

tiempo, y a su vez filtra la señal. Se decidió trabajar con los canales concatenados en todos los casos siguientes, para de esta forma disponer de más features para la posible aplicación de un modelo de machine learning.

b) En el dominio de la frecuencia se utilizó como extractor de features la frecuencia máxima obtenida en cada canal por cada ventana de tiempo, luego de aplicar el espectro de potencia a la ventana, y posteriormente se utilizó la media del resultado de la transformada de Fourier sobre la ventana de tiempo.

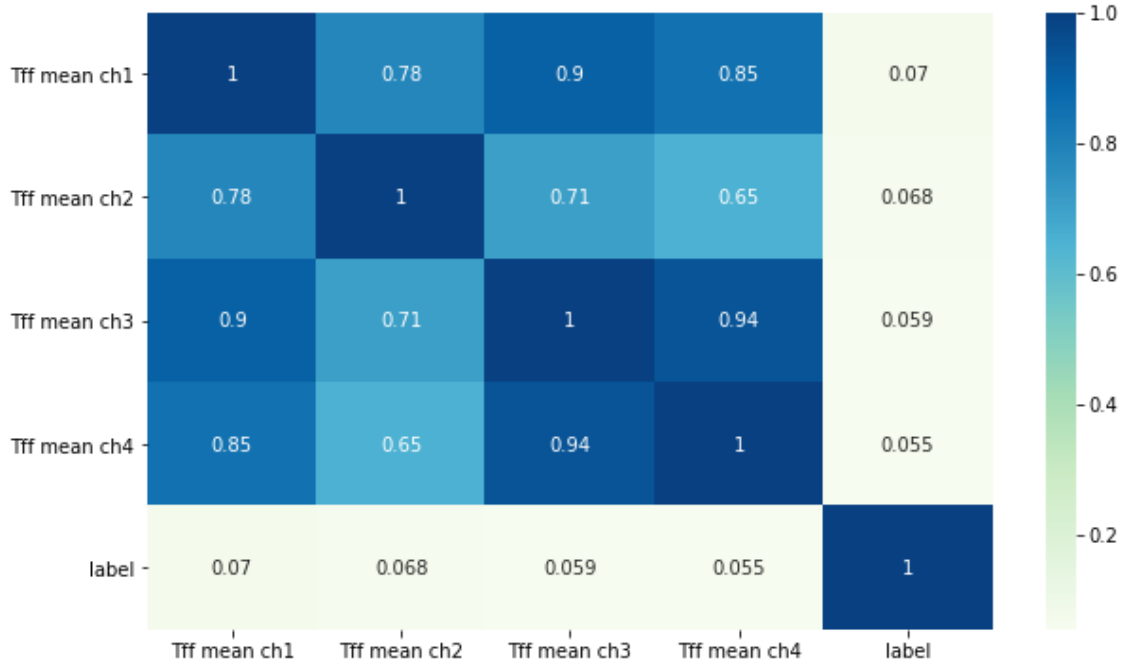
c) Ambos dataset generados fueron almacenados en Dataframes, y guardados en formatos csv, como df_frec_max y df_sxx_mean, respectivamente.

d) Heatmap de frecuencias máximas:




Para este caso, en el que se tomó como feature de extracción la frecuencia máxima del espectro de potencia aplicado sobre la ventana de tiempo, se observa que no se obtienen los resultados que se desearían, ya que no existe correlación entre los features y las etiquetas como se esperaría observar, quizas se deba a que existen pocos ejemplos, y la mayoría de ellos pertenecen a una etiqueta en la cual el paciente no observa ningún estímulo (label 99), por lo que corresponderían a ruido en el análisis de interés.

Heatmap de media de la transformada de Fourier:




Para este otro caso, en el que se tomó como feature de extracción la media del resultado de aplicar la transformada de Fourier sobre la ventana de tiempo, se observa que tampoco se obtienen los resultados que se desearían, ya que no existe correlación entre los features y las etiquetas como se esperaría observar, al igual que sucede en el caso anterior, quizas se deba a que como en el otro caso existen pocos ejemplos, y la mayoría de ellos pertenecen a una etiqueta en la cual el paciente no observa ningún estímulo (label 99), por lo que corresponderían a ruido en el análisis de interés, o quizás pueda ser porque no es un feature de extracción adecuado para aplicarle a la señal.

D) Características Temporales:

Cantidad de registros por clase:

- `99.0  1337`
- `2.0  434`
- `1.0  402`

Cantidad de ejemplos puros:

- `True  1499`
- `False  674`

Como se puede observar las clases están muy desbalanceadas, y hay un gran número de ejemplos impuros, pero se considera que es conveniente conservar esas características ya que es lo que ocurriría en la realidad, es decir, este sería el tipo de información al cual el modelo se enfrentaría al ser aplicado en la vida real.

En el particionado del dataset se considera por un lado como variable dependiente (y) las etiquetas, y como variable independiente (x) las features; debido a que se decidió analizar todos las sesiones y todos los pacientes juntos, se dividirá una parte del dataset (20%) para conservar como muestras de evaluación (test), y el resto del datase (80%) será utilizado como

muestras de entrenamiento (train), siendo estos escogidos al azar, de esta forma se evita que se introduzca un sesgo.