TAREA #3:

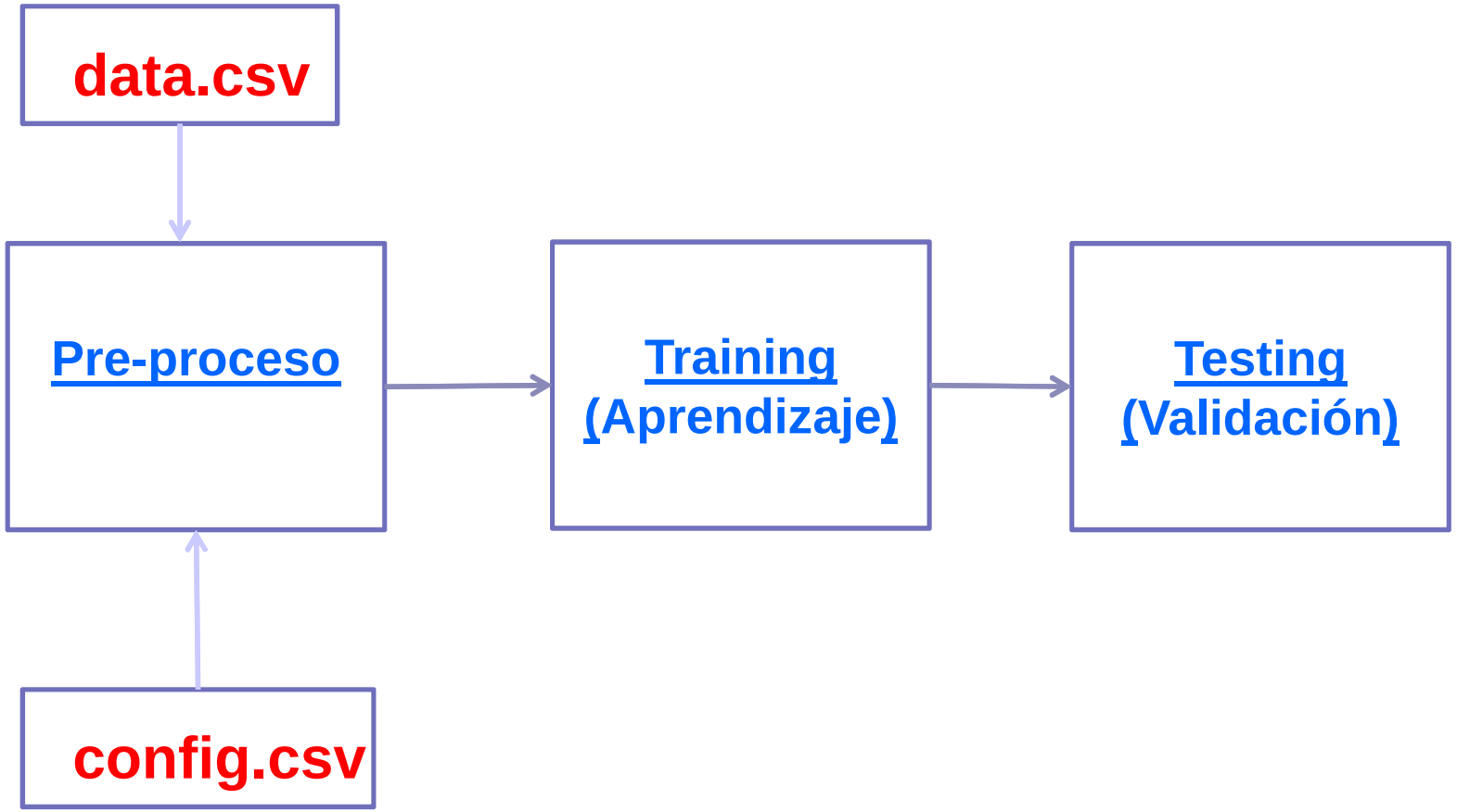
Diseños de Atributos
+
Red Neuronal Superficial (SNN)

Prof. NIBALDO RODRÍGUEZ A.

OBJETIVO

- Implementar y evaluar el rendimiento de un modelo neuronal basado en valores singulares de la matriz de Hankel.

Etapas del Modelo:



Pre-proceso: prep.py

- Formato: Data**.csv** :
  - M-filas
  - N-columnas
- Donde:
  - **(N-1)-columnas**: la data de entrada
  - **N-columna** : la etiqueta numérica
  - **M-filas** : números de muestras.
- Cargar parámetros de configuración.

Pre-proceso: prep.py

- Crear Atributos para cada muestra.

- 1.- Dividir cada muestra en segmentos de tamaño ***L-puntos*** (valores):
- 2.- Cada segmento de ***L***-puntos es descompuesto en ***J***-componentes usando el método siguiente:
  - 2.1 **Grupo #A:**
    - Descomposición de Hankel
  - 2.2 **Grupo #B:**
    - Descomposición Multinivel Diádica de Hankel.

Pre-proceso: prep.py

- Calcular la Entropía Espectral:
  - 1.- Para cada componentes estimado.
  - 2.- Para el segmento de tamaño **L** (data cruda).
- Calcular Valores Singulares de la Matriz de Componentes estimados .
- Número total de atributos por segmento:
  - **F= J+J+1.**
- Crear Etiquetas Binaria para los atributos.

Pre-proceso: prep.py

- Normalizar la base de datos de atributos usando la ecuación dada en clases.
- Crea archivos de Atributos y Etiquetas en formato '**csv**' usando los siguientes nombres:
  - **xData.csv**: representa la base de datos con los nuevos atributos creados por el método de Hankel.
  - **yData.csv**: representa la base de datos correspondiente a las etiquetas o clases de cada muestra.

train.py

- Cargar datos de configuración.
- Cargar la base de datos de atributos y etiquetas .
- Re-ordenar aleatoriamente las posiciones de cada muestra en las bases de datos previas.
- Dividir las bases de datos :
  - Data training: **dtrain.csv**
    - Matriz : xe(F,N), ye(nC,N).
      - F: número de atributos, N: número de muestras.
      - nC: número de clases.
  - Data de testing: **dtest.csv**
    - Matriz: xv(F,M), yv(nC,M), M: número de muestras

train.py

- Estructura del modelo SNN:
  - **Grupo #A:**
    - Capas Ocultas: función activación dada al final.
    - Capa de Salida: función activación Sigmoidal
    - Algoritmo de Aprendizaje: RMSprop.
  - **Grupo #B:**
    - Capas Ocultas: función activación dada al final.
    - Capa de Salida: función de activación Tan-Hiperbólica.
    - Algoritmo de aprendizaje: Adam.

train.py

- Crear archivo de costo :
  - **costo_snn.csv**
    - T-filas por 1-columna.
- Crear archivo de pesos:
  - **w_snn.npz**

test.py

- Cargar data de test y peso entrenados.
- Realizar proceso de forward de SNN.
- Crear archivo de métricas:
  - Matriz de Confusión:
    - **cmatriz.csv**.
  - Crear archivo de F-scores:
    - **fscores.csv**
      - (nC+1)-filas por 1-columa
        - Fila (nC+1) representa el F-scores promedio de las nC-clases.

Configuración: Pre-proceso:

- Parámetros : **param_prep.csv**

- Línea 1: Número de segmentos
- Linea 2: Longitud del segmento
- Línea 3: Número de componentes
- …
- …

Configuración : SNN:

- Parámetros : **param_snn.csv**

- Línea 1: Porcentaje de training
- Línea 2: Número Máx. Iteraciones
- Línea 3: Tasa de aprendizaje
- Línea 4: Nodos Ocultos de $Capa_1$.
- Línea 5: Nodos Ocultos de $Capa_2$.
- ……
- ……..

Funcion de Activación:

Grupos: Función de Activación

- 1. ReLu:

$$f(x) = \begin{cases} x, & x > 0 \\ 0, & x \le 0 \end{cases}, x \in \Re^d$$

- 2. L-ReLu:

$$f(x) = \begin{cases} 0.01x, & x < 0 \\ x, & x \ge 0 \end{cases}, x \in \Re^d$$

- 3. ELU:

$$f(x) = \begin{cases} a\left(e^x - 1\right), & x \le 0 \\ x, & x > 0 \end{cases}, x \in \Re^d$$

- 4. SELU:

$$f(x) = \lambda \times \begin{cases} a\left(e^x - 1\right), & x \le 0 \\ x, & x > 0 \end{cases}$$

$$x \in \Re^d, \ \lambda = 1.0507, \ a = 1.6732$$

Función de Activación

- 5. C-ReLU: $f(x) = \max(0, x) + \cos(x), \quad x \in \Re^d$

- 6. S-ReLU: $f(x) = \max(0, x) + \sin(x), \quad x \in \Re^d$

- 7. Tangente Hiperbólica (tanh): $f(x) = \dfrac{e^x - e^{-x}}{e^x + e^x}, \quad x \in \Re^d$

Función de Activación

- 8. mTanh:

$$f(x) = a \times Tanh\ (bx),\ \ x \in \Re^d$$
$$a = 1.7159\ ,\quad b = \frac{2}{3}$$

- 9. Bipolar sigmoid:

$$f(x) = \frac{2}{1+e^{-x}} - 1,\ \ x \in \Re^d$$

- 10. Softplus:

$$f(x) = \log\left(1+e^{x}\right),\ \ x \in \Re^d$$

ENTREGA

- Lunes 20/Junio/2022
  - Hora : 09:00 am
  - Lugar : Aula Virtual del curso

- **Lenguaje Programación:**
  - Python version: 3.7.6 window (anaconda)
    - numpy
    - panda

OBSERVACIÓN:

- Si un Grupo no Cumple con los requerimientos funcionales y no-funcionales, entonces la nota máxima será igual a 3,0 (tres coma cero).