

# Seminario de deep learning y redes neuronales

## Procesamiento de datos

Carlos Andrés Delgado S. Msc

Universidad San Buenaventura, Cali

Febrero de 2021

# Contenido

- 1 Introducción
- 2 Consideraciones de procesamiento de datos
- 3 Procesamiento de las entradas
- 4 Procesamiento de las salidas
- 5 Análisis de rendimiento de las redes neuronales

# Contenido

- 1 Introducción
- 2 Consideraciones de procesamiento de datos
- 3 Procesamiento de las entradas
- 4 Procesamiento de las salidas
- 5 Análisis de rendimiento de las redes neuronales

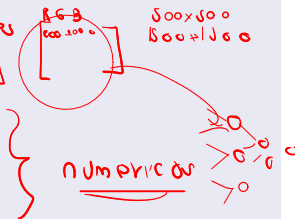
# Introducción

## Aplicaciones de las redes Neuronales

Las redes Neuronales pueden aplicarse a problemas de:

- Reconocimiento de imágenes ← matrices
- Reconocimiento de voz ←  $W = [000]$
- Análisis y filtrado de señales ←
- Clasificación
- Análisis de textos ←  $[101, 001]$

$[101, 001]$   $[20 15]$



# Introducción

## Modelo de Red Neuronal

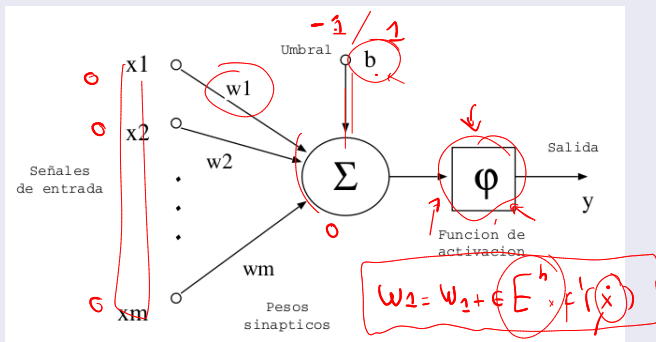


Figura 1: Modelo de red neuronal. Tomado de: [?]

# Introducción

## Modelo de Red Neuronal

La salida de la red neuronal  $z$  obedece a:

$$z = \varphi\left(\sum_{i=1} \underline{mw_i x_i} + b\right)$$

En forma vectorial:

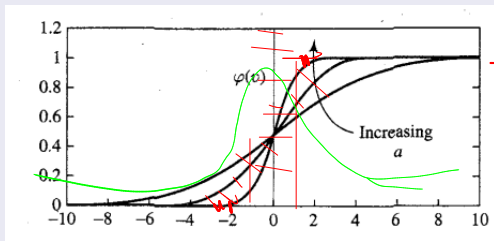
$$z = \varphi(wx^T + b)$$

Donde  $\varphi$  es una función conocida como función de activación:

¿Como podemos procesar texto o imágenes aquí? ¿Que debemos considerar?

# Introducción

## Funciones de activación

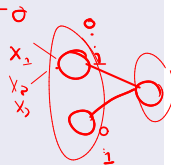


numeros

Figura 2: Función sigmoide. Tomado de: [?]

$$\frac{1}{1 + e^{-ax}}$$

$$a=2$$



¿Como debemos ingresar los datos a la red Neuronal, para usar apropiadamente las funciones de activación?

# Contenido

- 1 Introducción
- 2 Consideraciones de procesamiento de datos**
- 3 Procesamiento de las entradas
- 4 Procesamiento de las salidas
- 5 Análisis de rendimiento de las redes neuronales



# Consideraciones

## Introducción al procesamiento de datos

- 1 **Reducir el tamaño del espacio de entrada:** Reducir el número de variables de entrada es un objetivo común en el preprocesamiento
- 2 **Normalización:** Para muchos problemas es necesario ajustar los valores de variables a:
  - Valores de operación de funciones de activación ✓ -1 3
  - Mejorar la distribución de los datos
- 3 **Importante:** Estas estrategias se deben aplicar a todos los datos (entrenamiento, prueba o futuros) que ingresen a la red Neuronal.

→ Scalr

# Consideraciones

## Introducción al procesamiento de datos

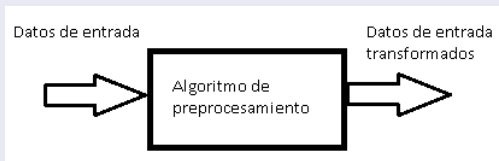
- 3 **Mantener la relación:** La transformación de datos es problema de mapeo de datos. Se trata de conservar el significado de las entradas.
- 4 **Reducción del ruido:** Eliminar datos ruidosos o sin contextos en el problema, ayuda a mejorar la precisión de los algoritmos

-54  
170

## Consideraciones

# Introducción al procesamiento de datos

Los algoritmos de procesamiento de datos pueden verse de la siguiente forma:



# RNN

# Contenido

- 1 Introducción
- 2 Consideraciones de procesamiento de datos
- 3 **Procesamiento de las entradas**
- 4 Procesamiento de las salidas
- 5 Análisis de rendimiento de las redes neuronales



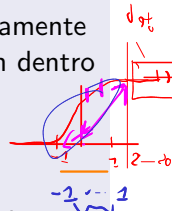
# Estrategias para las entradas



0 9999  
0 9998

## Estandarizar entradas

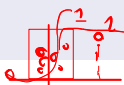
- Para que las redes neuronales puedan procesar correctamente los datos, debemos asegurarnos que sus entradas estén dentro de los rangos de las funciones de activación.
- Así mismo, se debe asegurar que estos se encuentran apropiadamente distribuidos.
- Se busca garantizar que los diferentes valores de entrada que se puedan presentar, sean reconocidos por la red Neuronal.
- Excluir los valores únicos, ya que estos no presentan patrones, por ejemplo ID.
- Para el manejo de datos, vamos a utilizar las librerías Sklearn y Pandas.



80%

20%

privado



ruído

# Estrategias para las entradas

## Preprocesamiento

Inicialmente, se debe realizar una limpieza de datos no deseados:

- Se deben eliminar los duplicados, para esto contamos con **drop\_duplicates** que nos ofrece Pandas.
- Eliminar valores nulos, con esto contamos con **dropna**
- Binarizar, categorizar y discretizar valores continuos. Este proceso depende del problema

# Estrategias para las entradas

## Categorizar

```
>>edades = [20, 22, 25, 27, 21, 23, 37, 31, 61, 45, 41, 32]
>>rangos = [18, 25, 35, 60, 100]
>>catEdades = pd.cut(edades, rangos)
>>##Tambien se puede pd.cut(edades, num) num = numero
    Natural
>>catEdades
Categorical:
array([(18, 25], (18, 25], (18, 25], (25, 35], (18, 25],
      (18, 25],
      (35, 60], (25, 35], (60, 100], (35, 60], (35, 60], (25,
      35]], dtype=object)
```

# Estrategias para las entradas

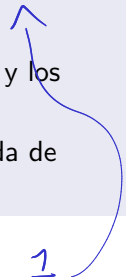
## Binarizar

Para esto contamos con **sklearn.preprocessing.Binarizer**

- Se selecciona un valor conocido como limite
- Los valores mayores o iguales se transforman en 1 y los menores en 0
- Debe usarse con cuidado, ya que representa pérdida de información

Person

$> 5$	$1$	$120$	$> 999$	$1$
$\leq 5$	$0$	$120$	$< 999$	$0$





# Estrategias para las entradas

## Binarizar

```
>>edades = [20, 22, 25, 27, 21, 23, 37, 31, 61, 45, 41, 32]
>>transformer = Binarizer(threshold=25).fit(edades) #
    Construye la transformación
>>catEdades = transformer.transform(edades) #Aplicar la
    transformación
>>print(catEdades)
[[0 0 0 1 0 0 1 1 1 1 1 1]]
```

## Factorización

Transformar datos categoricos/cadenas de texto en número

Sexo

Male= 0

Female = 1

Edad

[0, 5) = 0

[5 10) = 1

[10,15) = 2

....

[60,70) = 8

# Estrategias para las entradas

## Normalización Min-Max

- La normalización es un algoritmo de escalamiento lineal
- Se busca que los datos se escalen y queden en un rango 0-1 o -1 - 1.
- En Sklearn contamos con la clase **`sklearn.preprocessing.MinMaxScaler()`** para realizar este proceso.

# Estrategias para las entradas

## Normalización Min-Max

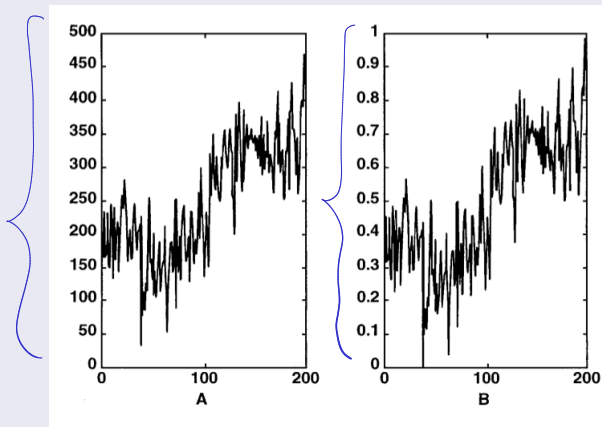
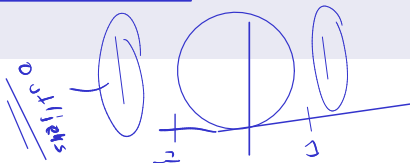


Figura 4: Normalización Min-Max [Li, 2000]

# Estrategias para las entradas

## Normalización Zscore

- Se busca que los datos correspondan a una distribución normal con media cero y varianza 1.
- Este algoritmo está basado en la desviación estándar de los datos.
- En Sklearn contamos con la clase `sklearn.preprocessing.StandardScaler()` para realizar este proceso.



# Estrategias para las entradas

## Normalización Zscore

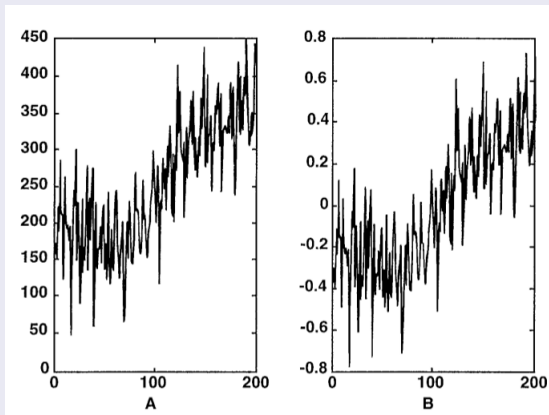


Figura 5: Normalización ZScore [Li, 2000]

# Estrategias para las entradas

## Reducir el tamaño del espacio de entrada:

- Si el número de datos no es tan grande comparado con el número de variables de entrada, podemos reducir su espacio
- Existen muchas estrategias para el procesamiento de datos, una de ellas es el análisis de componentes principales (PCA)[Li, 2000].

# Estrategias para las entradas

## Análisis de componentes principales

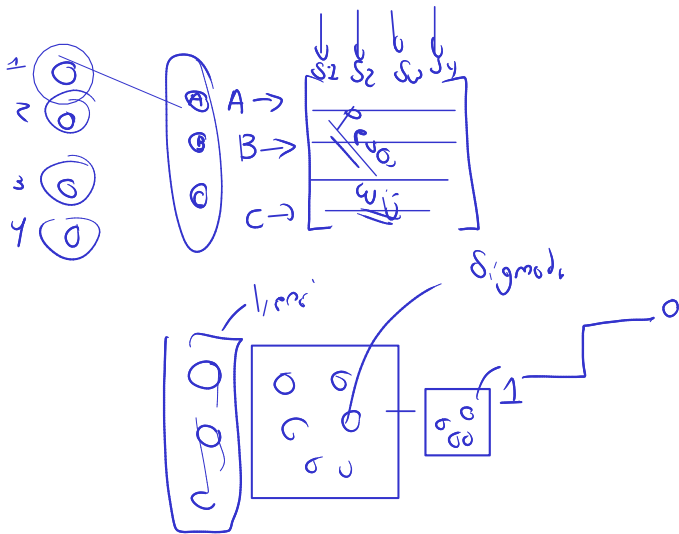
- Es un utilizado para determinar si un espacio  $m$ -dimensiones es significativo a partir de un espacio  $n$ -dimensional.
- En términos prácticos consiste en reducir el número de entradas de  $m$  a  $n$ .
- Se utiliza una transformación que toma  $m$  características de entrada y la transforma en  $n$ .



# Estrategias para las entradas

## Análisis de componentes principales

- Dado un conjunto de entrenamiento con  $N$  características  $x = \{x_1, x_2, \dots, x_N\}$  y  $n$  patrones
- Se puede plantear una transformación  $y = \widehat{W}x$  el cual transforma las  $N$  características en  $m$  minimizando el error cuadrático de reconstrucción. En otras palabras, se minimiza la pérdida de información.
- En Sklearn contamos con la clase **sklearn.decomposition.PCA** para realizar este proceso.



# Contenido

- 1 Introducción
- 2 Consideraciones de procesamiento de datos
- 3 Procesamiento de las entradas
- 4 Procesamiento de las salidas**
- 5 Análisis de rendimiento de las redes neuronales

# Estrategias para las salidas

## Introducción

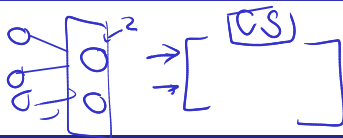
- Hasta el momento hemos manejado clasificadores binarios, por ejemplo, hombre o mujer, si o no.
- Para los clasificadores binarios, basta con tener una neurona de salida, cuya función de salida es un escalón.
- Sin embargo, para muchos problemas tenemos más de dos clases o salidas enteras.

# Estrategias para las salidas

## Estrategias de procesamiento

- Hasta el momento hemos manejado clasificadores binarios, por ejemplo, hombre o mujer, si o no.
- Para los clasificadores binarios, basta con tener una neurona de salida, cuya función de salida es un escalón.
- Sin embargo, para muchos problemas tenemos más de dos clases o salidas reales.

# Estrategias para las salidas



## Binarización salidas

- Podemos aplicarla si existe un número finito de clases.
- Consiste en convertir cada clase en una salida binaria, 0 si no se cumple y 1 si se cumple.
- Esto implica un aumento en el número de salidas.
- Ver archivo **4-BinarizarSalidas.py**.

C1 C2 C3  
Revo ~~Reu~~ Verd

0 1 2

S1 S2 S3  
| | |

# Estrategias para las salidas

## Datos continuos

- Se deben categorizar por rangos. Por lo que, debemos conocer los valores mínimo y máximo.
- Una vez esté realizado este paso, se deben binarizar.

## Categorías de Texto

- Se deben categorizar al igual que las entradas. Tener presente que debe existir un número finito de clases.
- Posteriormente binarizarlos.

# Contenido

- 1 Introducción
- 2 Consideraciones de procesamiento de datos
- 3 Procesamiento de las entradas
- 4 Procesamiento de las salidas
- 5 **Análisis de rendimiento de las redes neuronales**



# Análisis de rendimiento

## Introducción

- La precisión no es siempre la mejor medida para calificar que tan buena es una red Neuronal
- Puede suceder que la red sea muy buena prediciendo una clase y otras no. Si hay un gran número de datos de la clase que se predice correctamente, puede ocultarnos la realidad: **La red no está dando buenos resultados para un problema dado**
- Una herramienta que nos permite ver mejor el desempeño es la matriz de confusión

# Análisis de rendimiento

## Matriz de confusión

- Nos permite estudiar el desempeño de cada una de las clases individualmente
- Permite analizar las situaciones:
  - 1 **Falsos positivos:** La red indica que es de la clase y realmente no lo es
  - 2 **Verdaderos positivos:** Es de la clase y realmente lo es
  - 3 **Verdaderos negativos:** No es de la clase y esto es cierto
  - 4 **Falsos negativos:** No es de clase, pero realmente si es de la clase

# Análisis de rendimiento

## Matriz de confusión

Un ejemplo es la clasificación de animales

Valor Real	Valor predecido				
		Lobo	Zorro	Caballo	Elefante
	Lobo	2	1	0	0
	Zorro	3	3	4	2
	Caballo	1	1	1	6
	Elefante	2	0	7	4

Figura 6: Ejemplo matriz de confusión

# Análisis de rendimiento

## Matriz de confusión

- En Sklearn contamos con la función **`sklearn.metrics.confusion_matrix`** para generar la matriz de confusión.
- Se puede elegir estudiar con todos los datos disponibles, los de entrenamiento o los de prueba.
- Revisar: **6-MatrizConfusion.py**

## Curva ROC

- La Curva ROC nos permite comparar el desempeño de dos o mas algoritmos de predicción
- Utiliza los elementos calculados en la matriz de confusión para generar una gráfica
- El desempeño de un algoritmo es el área por debajo de esta curva.

# Análisis de rendimiento

## Curva ROC

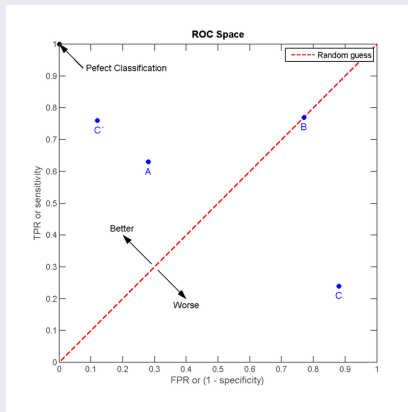


Figura 7: Ejemplo curva ROC: Tomado de Wikipedia

# Análisis de rendimiento

## Curva ROC

Tomemos un ejemplo sencillo con un clasificador binario para una población  $P = 99$ :

VP= 40	FP= 15
FN= 18	VN= 26

- $TPR = \frac{VP}{VP+FN} = 0,588$  Tasa positiva
- $FPR = \frac{FP}{FP+VN} = 0,366$  Tasa negativa
- $ACC = \frac{TP+VN}{P} = 0,666$  Precisión

El punto en la curva ROC corresponde a  $(TPR, 1 - FPR)$  o sea  $(0,588, 0,634)$ . La curva se genera a partir de la distribución logística.

# Análisis de rendimiento

## Curva ROC

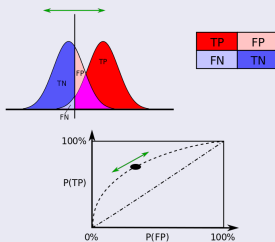


Figura 8: Generación de curva ROC: Tomado de Wikipedia

Revisar **7-CurvaROC.py**



# Referencias I



Curso de modelos computacionales.

<http://www.lcc.uma.es/~munozp/>.

Accessed: Octubre-2017.



Du, K.-L. and Swamy, M. N. S. (2010).

*Neural Networks in a Softcomputing Framework.*

Springer Publishing Company, Incorporated, 1st edition.



Heaton, J. (2008).

*Introduction to Neural Networks with Java.*

Heaton Research.



Li, H. (2000).

*Fuzzy Neural Intelligent Systems: Mathematical Foundation and the Applications in Engineering.*

CRC Press.