Regresión Lineal Múltiple. Introducción a Keras.

**Objetivos**

Estos ejercicios tienen como objetivo que te familiarices con la librería Keras implementando modelos simples de Regresión Lineal y analices el resultado.

En el primer ejercicio ofrecemos unos cuadernos para comprender mejor el modelo de regresión lineal con 2 variables de entrada.

En el ejercicio 2 ofrecemos una implementación muy simple y didáctica de regresión lineal con varias variables de entrada.

En el ejercicio 3, haremos exactamente lo mismo que en el 2 pero utilizando Keras. El objetivo de esta progresión es comprender mejor cómo funciona Keras internamente

En los ejercicios 4 y 5, buscamos aplicar este modelo a problemas reales.

**Preparación**

Desde una terminal con python en el path, ubicarse en el directorio de la práctica donde se encuentran los archivos con extensión “.ipynb” (notebooks) y escribir: “jupyter notebook”. Desde el entorno web, acceder a la dirección del servidor (generalmente<http://localhost:8888/>) para abrir los archivos. Opcionalmente, si lanzás jupyter notebook desde Anaconda Navigator, copiá los archivos de la práctica a la carpeta raíz del notebook.

**Ejercicio 1** En este ejercicio, explorarás el modelo de Regresión Lineal con 2 variables de entrada y 1 de salida a través de 2 notebooks que muestran cómo cambia el plano de predicción en base a los valores de los parámetros, y cómo se ajusta este plano durante el entrenamiento.

*Archivo*: Regresion Lineal 2D - Modelo.ipynb

*Archivo*: Regresion Lineal 2D - Aprendizaje.ipynb

**Ejercicio 2** En este ejercicio deberás implementar la función de *predicción* o función *forward* de un modelo de regresión lineal con múltiples variables de entrada y 1 de salida. Verifica que la implementación sea correcta con los casos de prueba, y luego que el modelo entrene correctamente. Esta implementación utiliza una clase reutilizable similar al modelo de Keras.

*Archivo*: Regresion Lineal - Clase.ipynb

**Ejercicio 3** El código de este ejercicio implementa un el modelo de regresión pero con la librería Keras. Si bien no hay nada que implementar, estudia y ejecuta el código y prueba también distintas combinaciones de la tasa de aprendizaje (lr), la cantidad de iteraciones (epochs) y el tamaño del batch (batch\_size) para ver su efecto en el aprendizaje.

*Archivo*: Regresión Lineal con Keras.ipynb

**Ejercicio 4** Utilizando datos de países obtenidos de la [OMS](https://www.who.int/es), vamos a predecir la esperanza de vida promedio de sus habitantes. Dichos datos se encuentran en el archivo **who\_life\_expectancy.csv**. Podés encontrar más información sobre el conjunto de datos en su [sitio de origen](https://www.kaggle.com/kumarajarshi/life-expectancy-who).

1. Preprocesamiento. (podés utilizar el dataset **who\_no\_missing\_numeric.csv** y saltear los pasos a-c)
   1. Quitar las filas con datos faltantes, ya que el modelo no puede entrenarse o evaluarse con datos faltantes. Podés hacer esto con la función [dropna](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.dropna.html) de Pandas
   2. Preprocesamiento: Quitar la columna “Country” (país) para simplificar el análisis, ya que no aporta información real, y además al ser nominal o string, no se puede utilizar directamente por un modelo de regresión o una red neuronal.
   3. Identificar y convertir toda otra columna del conjunto de datos que no sea numérica, ya que nuevamente, estos modelos no pueden utilizar datos no númericos como entrada. Podés utilizar pandas y un script de python, o simplemente un programa de planilla de cálculo.
2. Normalizar los valores de las variables numéricas. *Opcional:* probá entrenar también sin normalizar, y analizá qué sucede en base al valor de α, la tasa de aprendizaje.
3. Escribí un modelo de Regresión Lineal en Keras para predecir la expectativa de vida. Para ayudarte, podés hacer una copia del notebook del ejercicio anterior, y adaptarlo para cargar el archivo **who\_life\_expectancy.csv**.
4. Una vez entrenado el modelo, indicá cuál es su error MSE (Mean Squared Error o Error Cuadrático Medio). Interpretarlo en términos de la expectativa de vida ¿qué tan bien predice el modelo?
5. Podemos analizar qué variables utiliza el modelo para realizar la predicción en base a los valores de sus pesos. Analizá los pesos del modelo para determinar cuáles son las variables correspondientes que más impacto tienen en la predicción. Poniéndote en lugar de un estadista, ¿cuáles de estas variables es más importante influenciar si se busca aumentar la esperanza de vida?
6. Podría argumentarse que algunas variables deberían quitarse, debido a que tienen una relación muy directa con la variable a predecir. En particular, la mortalidad adulta e infantil están demasiado relacionadas con la expectativa de vida, y entonces el modelo no aprende algo útil, ya que no son variables que un estadista pueda cambiar directamente. Quitá esas variables del conjunto de datos y repetí el análisis. ¿Cuáles son las variables relevantes ahora?

Código para imprimir los pesos, asumiendo que la variable **model** tiene el modelo:

for layer in model.layers:

print(layer.get\_config(), layer.get\_weights())

Nota: El notebook **Expectativa de Vida - WHO - resuelto.ipynb**  tiene la solución de código de este ejercicio.

**Ejercicio 5** (opcional)Si bien el descenso de gradiente permite entrenar modelos de regresión lineal, también se puede hacer con métodos analíticos tradicionales. Vamos a ver un caso en donde estos métodos no funcionan correctamente. En particular, esun problema con imágenes, de modo que el tamaño de los vectores de entrada es bastante grande. Además, vamos a realizar predicciones con *varias* variables de salida. Explorá el cuaderno y el uso de Keras en este caso.

*Archivo*: Regresion Lineal Multiple - Keras - Imágenes.ipynb

**Ejercicio 6 (Avanzado, Opcional).** Realiza las guías 1 a 9 de [*simplenn*](https://github.com/facundoq/simplenn)