TALLER DE PROCESAMIENTO DE SEÑALES

1er Cuatrimestre 2026 - Trabajo Práctico Nº 1 - Regresión Lineal

---

IMPORTANTE: Solamente se podrá importar numpy, matplotlib.pyplot y pandas.

Se desea predecir la cantidad de pasajeros que viajarán con una determinada aerolínea. Para ellos se cuenta con su serie de tiempo histórica.

(**a**) *Creación de la base de datos:*

1. Descargar la base de datos de `https://raw.githubusercontent.com/jbrownlee/Datasets/master/airline-passengers.csv`. Dicha descarga debe estar autocontenida en el código.
2. Utilizando `plot` (matplotlib) graficar la serie de tiempo. Incluir nombres en los ejes. Explicar cualitativamente los datos observados.
3. Se desea construir un conjunto de datos para efectuar la regresión, pero por desgracia se cuenta con una sola serie de tiempo. Sin embargo, dicha serie posee una componente estacional de 12 meses. Es razonable entonces pensar que la información necesaria para predecir un mes se encuentra presente en los 12 meses inmediatamente anteriores. Construir la base de datos teniendo en cuenta todos los solapamientos posibles.

(**b**) *Regresión Lineal:*

1. Implementar una regresión lineal (matricial) a partir de los datos generados previamente. El código debe estar estructurado de la siguiente manera:

```
class LinearRegression:
    # Opcional, para inicializar atributos o declarar hiperparámetros
    def __init__(self,...

    # Etapa de entrenamiento
    def fit(self,X,y):

    # Etapa de testeo
    def predict(self,X):

    # Cómputo del error
    def mean_squared_error(self,X,y):
```

   A su vez, se debe poder extraer el atributo `LinearRegression.params_` (parámetros encontrados durante el entrenamiento).
2. Entrenar un algoritmo de regresión lineal con el dataset generado anteriormente e indicar el error cuadrático medio de entrenamiento. Comparar gráficamente la serie original con la predicción.
3. Predecir la cantidad de pasajeros de los próximos 30 meses de forma *autorregresiva*. Es decir, utilizar los últimos 12 meses para predecir el último, incluirlo en la serie, utilizar los nuevos últimos 12 meses para predecir el próximo, etc. Graficar e interpretar.
4. El conjunto de datos creado anteriormente no posee muestras independientes. ¿Qué supuestos se ven afectados? Analizar por qué el algoritmo funciona. : El cuaderno de *procesos estocásticos* puede ser útil.

(**c**) *Gradiente Descendente:*

1. Agregar un método `fit_gradient(self,X,y,learning_rate)` a la implementación desarrollada anteriormente que reemplace el entrenamiento matricial por gradiente descendente. Justificar la expresión del gradiente.
2. Repetir los puntos 2 y 3 del inciso (**b**) utilizando gradiente descendente. Elegir el *learning rate* de manera que el error cuadrático de actualización de los parámetros converjan (a nivel *tiny* de numpy). Indicar el número de iteraciones hasta converger.