
Universidad de
SanAndrés

# I302 - Aprendizaje Automático y Aprendizaje Profundo

**1er Semestre 2025**

Trabajo Práctico 1

---

**Fecha de entrega:** Lunes 24 de marzo de 2025, 23:59 hs

**Formato de entrega:** Los archivos desarrollados deberán entregarse en un archivo comprimido (.zip) a través del Campus Virtual, utilizando el siguiente formato de nombre: *Apellido_ Nombre_ TP1.zip*. Se aceptará únicamente un archivo por estudiante. En caso de que el nombre del archivo no cumpla con la nomenclatura especificada, el trabajo no será corregido.

Dentro del archivo .zip se deberá incluir un Jupyter Notebook llamado *Entrega_ TP1.ipynb* con las respuestas a los ejercicios y los gráficos resultantes. Se recomienda, en la medida de lo posible, no realizar todo el desarrollo dentro del Jupyter Notebook; en su lugar, se sugiere utilizar archivos `.py` para desarrollar el código, siguiendo las buenas prácticas de programación y modularización vistas en clase.
Se recomienda seguir la estructura sugerida al final del trabajo práctico.

---

# Trabajo Práctico 1: Regresión

El objetivo de este trabajo es desarrollar y evaluar diversos modelos de regresión para estimar el precio de venta de una vivienda. El dataset contiene información sobre los valores en dólares de distintas propiedades residenciales, recopilados de diversas fuentes del mercado inmobiliario. Además, incluye características adicionales relacionadas con cada propiedad, tales como área, número de habitaciones, año de construcción, entre otros. Previamente, el conjunto de datos fue dividido en subconjuntos de desarrollo (*casas_dev.csv*) y de prueba (*casas_test.csv*).

**NOTA**: Utilice exclusivamente NumPy para la implementación de funciones y/o clases; Pandas y Matplotlib/Seaborn para el manejo y gráfico de datos. No se permite el uso de librerías de Machine Learning como scikit-learn.

## 1. Exploración de datos

El primer paso consiste en explorar la naturaleza de los conjunto de datos.

1.1) Se puede decir que este dataset está "sucio" ya que pueden haber datos erróneos o faltantes. Imprima en pantalla un fragmento aleatorio del dataset e indique qué columnas sería útil modificar y de qué manera. Es útil explorar un resumen con los estadísticos básicos y les recordamos que pueden existir valores faltantes (expresados como NaNs).

1.2) Realice una serie de histogramas y diagramas de dispersión (scatterplots) para mostrar la relación entre las variables. Para esto, es útil el comando `pairplot`. ¿Qué conclusiones puede extraer a simple vista acerca de las relaciones entre las variables? Se permite la creación de gráficos adicionales que considere relevantes.

1.3) Divida el conjunto de desarrollo *casas_dev.csv* en un 80 % para entrenamiento (train) y un 20 % para validación (validation). Estos conjuntos se utilizarán para entrenar y validar los modelos que desarrollarán en las siguientes secciones. Se sugiere normalizar los datos para lograr un mejora el condicionamiento numérico de los algortmos de aprendizaje. Recuerde que si normaliza los datos, cuando haga una predicción deberá invertir la normalización para volver a las escalas originales.

## 2. Implementación de Regresión Lineal

2.1) Implemente una clase de regresión lineal. El constructor deberá recibir $X$ e $y$, y se deberá incluir como métodos de entrenamiento de la clase: entrenamiento mediante la pseudo-inversa y entrenamiento mediante descenso por gradiente. En ambos casos se debe almacenar como atributo `self.coef` los coeficientes (pesos) resultantes de la regresión.

**NOTA:** Asegúrese de que haya un método que permita imprimir los coeficientes con los nombres de sus respectivas variables de forma prolija.

2.2) Implemente la función de pérdida utilizando el Error Cuadrático Medio (ECM).

2.3) Verifique que su implementación de la regresión lineal funcione correctamente tanto para cuando se tiene una característica como para múltiples características.

# 3. Aplicación de Modelos de Regresión

3.1) Realice una regresión lineal simple (con una sola característica) para predecir el precio de una propiedad utilizando como variable explicativa solo el *area*.

3.2) Desarrolle un modelo de regresión lineal que prediga el precio en función de las caraceterísticas que considere relevantes. Utilizando el modelo entrenado, prediga el precio de *vivienda_Amanda.csv*.

3.3) En el dataset, ¿cuál es el valor promedio por metro cuadrado de una **casa**?

3.4) Benito está evaluando la posibilidad de construir una **pileta**. ¿Cómo afectaría la construcción de una pileta al precio de su propiedad? Explique cómo llegó a este resultado.

# 4. Feature Engineering

El feature engineering es una técnica que se utiliza para aportar mayor flexibilidad a los modelos.

4.1) Construya un conjunto de características derivadas a partir de las originales, que le parezca relevante para predecir el precio de una propiedad, utilizando técnicas de Feature Engineering.

4.2) Realice una regresión lineal como en 3.2), incorporando las características que desarrolló en el punto 4.1).

4.3) Repita el proceso del ítem anterior generando 300 nuevos features como potencias de los features anteriores (por ejemplo: $[x_1^2, x_2^5, \ldots, x_n^{12}]$).

4.4) ¿Qué resultados esperaba obtener con estos experimentos?

# 5. Regularización

5.1) Modifique la clase de Regresión Lineal para que pueda recibir como parámetros dos coeficientes de regularización L2 y L1, que por defecto serán 0.

5.2) Entrene un modelo de regresión lineal con regularización L2 (a veces llamado "Ridge regression") utilizando como características las mismas que en el punto 4.2) y grafique el valor de los parametros $w^*$ en función del coeficiente de penalización $\lambda$.

5.3) Realice el mismo gráfico del ítem anterior pero ahora utilizando regularización L1 (a veces llamado "LASSO regression"). Recuerde que no hay una solución analítica para $w^*$ con regularización L1, por lo cual solo será posible entrenar el modelo mediante gradiente descendiente. Analice el gráfico obtenido y compárelo con el anterior. ¿Qué conclusiones se pueden extraer en cuanto al efecto de utilizar regularización L1 vs. L2?

5.4) Ajuste el modelo de regresión del item 4.2) utilizando regularización L2 con coeficiente de regularización $\lambda$ elegido de manera fundamentada, tomando como referencia lo graficado en el punto 5.2). ¿Cuál es el ECM obtenido en el conjunto de validación (y_val)? Explique en qué beneficia al modelo la regularización L2.

5.4) En el ítem anterior se eligió el coeficiente de regularización $\lambda$ aleatoriamente. Ahora determine su valor mediante un barrido del hiperparámetro y observando la variación del ECM evaluado sobre el conjunto de validación. ¿Se logró mejorar el ECM de validación?

5.6) Repita el procedimiento del ítem anterior para elegir el valor del coeficiente de regularización $\lambda$, pero ahora evalue el ECM mediante validación cruzada. Grafique la variación del ECM vs. el valor del coeficiente de regularizacion.

# 6. Selección de Modelo y Evaluación de Capacidad Predictiva

6.1) De todos los modelos realizados hasta ahora, ¿cuál seleccionaría para implementar en producción? Justifique su respuesta.

6.2) Para evaluar la capacidad predictiva del modelo seleccionado, compute y reporte el MAE (*Mean Average Error*) y RMSE (*Root Mean Squared Error*) sobre el conjunto de prueba.

**Estructura Sugerida para la Entrega del Trabajo Práctico**

Para organizar el desarrollo de este trabajo práctico de manera eficiente, se recomienda modularizar las diferentes funcionalidades en archivos `.py` y carpetas separadas, lo que facilitará la reutilización del código y la depuración. A continuación se propone una posible estructura de entrega:

```
Apellido_Nombre_TP2.zip
|- data/                  # Carpeta para los datos del proyecto
    |- raw/                   # Datos originales sin modificar
        |- casas_dev.csv
        |- casas_test.csv
    |- processed/             # Datos procesados y curados

|- src/                   # Carpeta para el código fuente del proyecto
    |- utils.py               # Funciones auxiliares
        |- save_results()           # Para guardar resultados
        |- load_model()             # Para cargar un modelo guardado

    |- metrics.py             # Funciones para calcular métricas
        |- mse()
        |- mae()
        |- rmse()

    |- preprocessing.py       # Funciones para el preprocesamiento
        |- one_hot_encoder()
        |- normalize()
        |- handle_missing_values()

    |- models.py              # Clases para los modelos de ML
        |- class LinearRegression()

    |- data_splitting.py      # Funciones para dividir los datos
        |- train_val_split()
        |- cross_val()

|- notebooks/             # Carpeta para Jupyter Notebooks
    |- Entrega_TP1.ipynb    # Respuestas de todos los ejs del TP

|- requirements.txt         # Especificar dependencias del proyecto
|- README.md                # Descripción del TP e instrucciones de uso
```

Esta estructura es flexible. Se pueden agregar o eliminar archivos según sea necesario, pero es obligatorio incluir un Jupyter Notebook con todas las respuestas del TP.