Práctica de Laboratorio Regresión Lineal CComp9-1

Ricardo Lazo Vásquez
ricardo.lazo@ucsp.edu.pe

Universidad Católica San Pablo

1 Introducción

La presente practica desarrolla la implementación de la regresión lineal tanto univariada como multivariada en tres datasets base (Precio-Viviendas, Temperatura-Océano, Consumo-Petróleo). El principal objetivo es demostrar la efectividad en función de la regresión lineal con respecto a la función costo y compararla con la ecuación normal. En la sección 2 se explicará la implementación, Los experimentos propuestos por la practica asi como sus resultados serán explicados en la sección 3, finalmente se darán las conclusiones en la sección 4.

2 Implementación

La regresión lineal a presentar debía tener funciones obligatorias, además de tener los experimentos de evaluación listos para correr al momento de hacer la evaluación de implementación. La organización de las funciones se modularizo en distintas librerías para su facilidad de uso y su reutilización.

Documento .py	**Contenido**
Complementos.py	Funciones complementarias para la librería my_lib.py
Intermediarios.py	Experimentos implementados para demostración
my_lib.py	Funciones por calificar validez
tests_file.py	Script de prueba inicial de las funciones de my_lib.py
jsonsolver.py	Archivo de soporte al experimento numero 2
main.py	Archivo donde se lanza el menú de selección de experimentos
Transposerhelper	Script de ayuda para obtener los resultados del experimento 2 con mayor facilidad.

Table 1: Scripts de implementación

El código se encuentra disponible en https://github.com/TheReverseWasp/TIA-Lab-1st-Half/tree/master/Regresion%20Lineal y los trabajos futuros de la segunda mitad estarán disponibles en la carpeta precedente a la del GitHub.

En la carpeta Implementación/Código/, mientras que los resultados del código en la carpeta /Implementación/Resultados/. Y la descripción de los scripts esta descrita en la Tabla 1

2.1 Detalles de la implementación

Dividí gran parte de la implementación en dos archivos "my_lib.py" y "Intermediarios.py", la distribución general de las funciones esta descrita en la Tabla 1.

Función	Descripción
Leer_Datos	Recibe el nombre del archivo y devuelve un numpy_array
Normalizar_Datos	Normaliza los datos en un intervalo más pequeño.
Crear_Entrenamiento_Prueba	Función que devuelve un set del 70% de datos donde se realizara el entrenamiento y 30 % de datos para hacer pruebas.
Calcular_Costo	Calculo de la función costo de Mean Square Error (J) dado una matriz X, un vector θ y un vector Y.
Gradiente_Descendiente	Función de gradiente descendiente, recibe una matriz X, un vector Y, un learning_rate y numero de iteraciones.
Ecuación_Normal	Función que recibe una matriz X y un vector Y y devuelve un θ

Table 2: Descripción de las funciones a implementar

Las funciones obligatorias para la implementación son las de la Tabla 2 y están implementadas en el script "my_libs.py". Y Finalmente la implementación de los experimentos están implementados en el script "Intermediarios.py".

3 Experimentos y Resultados

3.1 Experimentos

Uno de los objetivos de esta implementación es observar los resultados de cuatro experimentos explicados en la Tabla 3.

3.2 Resultados

Experimento 1 Se desea saber el MSE calculado con la ecuación normal de los tres datasets predefinidos. Los resultados del experimento uno están en la Tabla 4.

Experimento	Descripción
Experimento 1	Mostrar el MSE (J) obtenido de la ecuación normal de los 3 datasets base.
Experimento 2	Buscar los mejores parámetros de entrenamiento para cada dataset dado el vector de índices de aprendizajes: [0.01; 0.05; 0.1; 0.2; 0.3; 0.4] y el vector de iteraciones [500, 3500] con saltos de 500.
Experimento 3	Para el dataset de Precio-Viviendas, graficar la gradiente descendiente con la ecuación normal.
Experimento 4	Para los tres datasets base, mostrar la función de costo en el conjunto de entrenamiento y el de prueba.

Table 3: Descripción de los experimentos a desarrollar

Dataset	**MSE**
Precio-Viviendas	3.1282361369550183e-28
Temperatura-Océano	1.00738000215988e-27
Consumo-Petróleo	2.0891914879961286e-27

Table 4: Resultados del primer experimento

Experimento 2 Para el experimento son se desea saber cuáles son los mejores parámetros de entrenamiento (iteraciones, índice de aprendizaje). Para esto se evaluaron con las listas de la Tabla 3. Tanto los resultados del entrenamiento como los de prueba están en el GitHub, en este trabajo solo listaremos los de entrenamiento.

	0.01	**0.05**	**0.1**	**0.2**	**0.4**
500	0.0013	0.0005	0.0001	1.8251e-05	1.8962e-07
1000	0.0010	0.0001	1.8429e-05	1.9536e-07	2.1281e-11
1500	0.0008	5.7462e-05	1.9115e-06	2.0911e-09	2.3883e-15
2000	0.0006	1.8518e-05	1.9828e-07	2.2382e-11	2.6803e-19
2500	0.0005	5.9680e-06	2.0566e-08	2.3958e-13	3.0077-23
3000	0.0004	1.9233e-06	2.1333e-09	2.5644e-15	3.3452e-27
3500	0.0003	6.1983e-07	2.2127e-10	2.7449e-17	8.3595e-32

Table 5: Experimento 2 en el dataset Precio-Viviendas en los datos de entrenamiento

En la Tabla 5 se puede ver como el mejor índice de aprendizaje es **0.4**, notemos que el MSE en la iteración 3000 llega muy cerca del MSE de la ecuación normal. En la Tabla 6 de manera similar el índice de aprendizaje que más rápido

	0.01	**0.05**	**0.1**	**0.2**	**0.4**
500	0.4767	0.0787	0.0082	8.9956e-05	9.9896e-09
1000	0.3041	0.0083	9.2631e-05	1.1040e-08	1.3864e-16
1500	0.1942	0.0008	1.0366e-06	1.3549e-12	1.9277e-24
2000	0.1240	9.3993e-05	1.1601e-08	1.6629e-16	1.3039e-29
2500	0.0792	9.9685e-06	1.2983e-10	2.0409e-20	1.3039e-29
3000	0.0506	1.0572e-06	1.4530e-12	2.5117e-24	1.3039e-29
3500	0.0323	1.1212e-07	1.6261e-14	4.5391e-28	1.3039e-29

Table 6: Experimento 2 en el dataset Temperatura-Océano en los datos de entrenamiento

	0.01	**0.05**	**0.1**	**0.2**	**0.4**
500	0.0002	0.0001	0.0001	7.4935e-05	9.1072e+16
1000	0.0002	0.0001	7.5020e-05	3.8411e-05	3.9887e+32
1500	0.0002	9.2478e-05	5.2993e-05	2.0422e-05	1.7469e+48
2000	0.0001	7.5062e-05	3.8444e-05	1.0868e-05	7.6511e+63
2500	0.0001	6.2674e-05	2.8021e-05	5.7847e-06	3.3510e+79
3000	0.0001	5.3016e-05	2.0441e-05	3.0788e-06	1.4676e+95
3500	0.0001	4.5101e-05	1.4912e-05	1.6386e-06	6.4279e+110

Table 7: Experimento 2 en el dataset Consumo-Petróleo en los datos de entrenamiento

converge es **0.4** necesitando entre 500 y 1000 iteraciones. Pero en el caso del petróleo se puede ver como en la Tabla 7 que el mejor índice de aprendizaje es **0.2** necesitando menos de 500 iteraciones antes de empezar a variar en el costo por intervalos grandes.

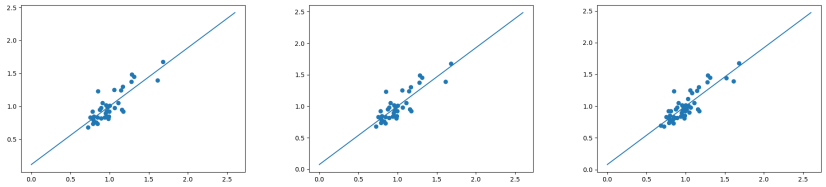
Experimento 3 El experimento consiste en evaluar las θs de la ecuación normal y de gradiente descendiente en el dataset del precio de viviendas.

Se puede ver en la Figura 1 el comportamiento general de los θs siendo estos muy similares en la normal como en la gradiente descendente.

3.3 Experimento 4

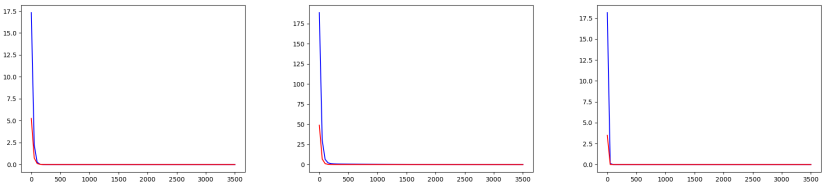
Los resultados seleccionados para el experimento 4 serían los de índices de aprendizaje **0.01**, **0.4** y **0.02**. En ellos se debe comparar los costos de los datos de entrenamiento y de prueba. Los resultados se pueden ver en las Figura 2, 3 y 4. En ellos se puede visualizar la función de consto de entrenamiento en azul y la función de costo de prueba en rojo.

Los resultados de las funciones de costo de los tres datasets con índice de aprendizaje **0.01** se pueden apreciar en la figura 2, en ellas se puede ver una pronta convergencia en no menos que 500 iteraciones y mejor rendimiento en la función de costo-prueba.



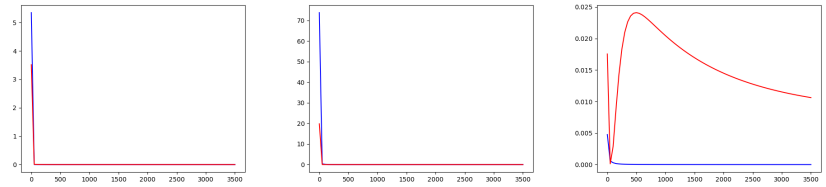
(a) Gradiente Descendente (b) Ec. Normal con datos de entrenamiento (c) Ec. Normal con todos los datos

Fig. 1: Resultados del experimento 3



(a) Costo en Precio-Viviendas (b) Costo en Temperatura-Océano (c) Costo en Consumo-Petróleo

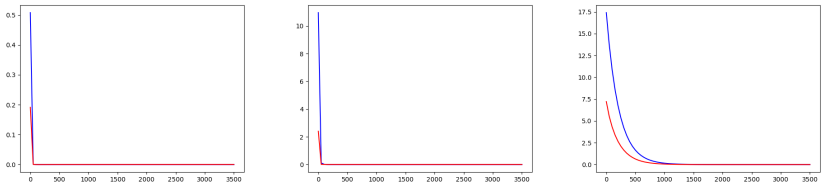
Fig. 2: Resultados de la Función Costo con gradiente descendiente con índice de aprendizaje de 0.01



(a) Costo en Precio-Viviendas (b) Costo en Temperatura-Océano (c) Costo en Consumo-Petróleo

Fig. 3: Resultados de la Función Costo con gradiente descendiente con índice de aprendizaje de 0.2

En el caso de la Figura 3, la función de índice de aprendizaje de **0.2** se tiene un comportamiento similar tanto en (a) como en (b) pero en (c) se puede observar una alteración en la función de costo de prueba, una razón es la selección del conjunto de entrenamiento/prueba.



(a) Costo en Precio-Viviendas (b) Costo en Temperatura-Océano (c) Costo en Consumo-Petróleo

Fig. 4: Resultados de la Función Costo con gradiente descendiente con indice de aprendizaje de 0.4

Finalmente en la Figura 4 se puede apreciar el caso del índice de aprendizaje **0.4** se ve una mejora en los dos primeros datasets, pero en el último se ve una convergencia mucho más lenta en los conjuntos entrenamiento/prueba.

Recordemos que los resultados del experimento 2 mostraron el comportamiento de las gráficas del experimento 4.

4 Conclusiones

De la implementación realizada se pueden considerar las siguientes mejoras: primero mejorar la función de Crear_Entrenamiento_Prueba para mejorar la performance de los experimentos, segundo pasar el código a un notebook para una experiencia más interactiva. Tener en cuenta que ambas alternativas no estaban dentro de los parámetros de este informe. Se puede notar además que la elección de un adecuado índice de aprendizaje asi como un numero de iteraciones pueden fortalecer el desempeño de nuestro algoritmo, por lo que otra posible mejora podría ser el uso de una matriz de calor como en la Figura 5.

References

1. Andrew Ng. Specialization in Deep Learning.
   `https://www.coursera.org/specializations/deep-learning`
2. genixpro
   `https://github.com/electricbrainio/hypermax`

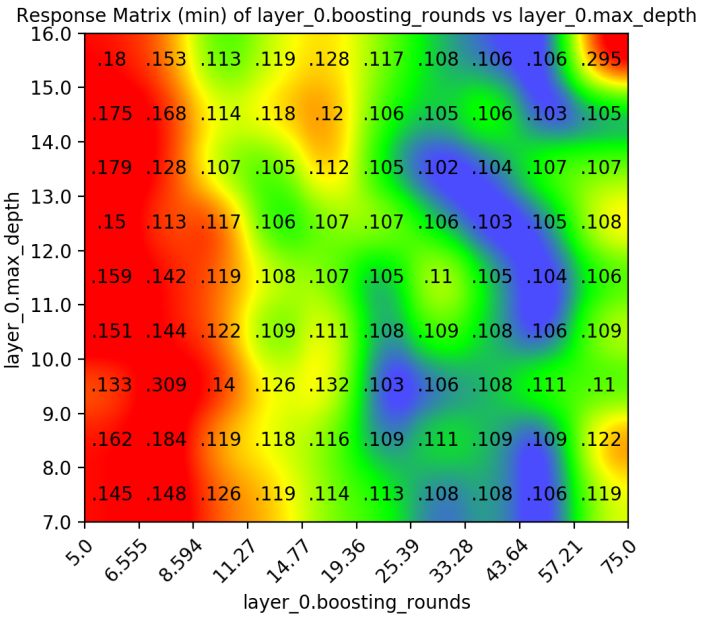


Fig. 5: Matriz de calor de ejemplo del GitHub en [2]