Universidad Córdoba

Ingeniería Informática
Mención en Computación
3º Curso

Introducción al Aprendizaje Automático

Regresión Lineal

Valentín Gabriel Avram Aenachioei
Víctor Rojas Muñoz
Damián González Carrasco
Ángel Simón Mesa
Pablo Fernández Arenas

Año académico 2022-2023
Córdoba, 25 de Marzo, 2023

Índice

Índice de tablas

Índice de figuras

En este documento, vamos a realizar las explicaciones y análisis de resultados pertinentes a la segunda práctica de la asignatura, regresión lineal múltiple.
No se va a especificar nada sobre el desarrollo del código. Este se ha realizado usando Google Colab. Usando **este enlance** se puede visualizar y ejecutar el código usado para la implementación del algoritmo y pruebas realizadas para esta práctica.

1. Regresión Lineal Múltiple

1.1. Codifica una función que aplique el algoritmo de regresión lineal múltiple utilizando los dos últimos "trucos" o criterios que se explicaron en clase, y que coinciden con la aplicación del método del gradiente descendente sobre la función de error absoluto medio y error cuadrático medio.

Respecto a este apartado, la codificación se desarrolla en el Google Colab adjunto.

1.2. Desarrolla un programa principal que haga uso de estas dos funciones, y que las aplique sobre el siguiente conjunto de datos. Compara los resultados obtenidos con ambas versiones del algoritmo.

Una vez creadas las funciones para el método del gradiente usando como medida del error el error cuadrático medio y el error absoluto medio, hemos creado un modelo de regresión lineal múltiple para cada método sobre cada conjunto de datos.
Aunque cada método este desarrollando el método del gradiente descendiente sobre funciones de error distintas, hemos calculado ambas medidas de error en ambos métodos, para tener un método de comparación sencillo sin necesidad de entrar en tests estadísticos complejos.

Con respecto al Error Cuadrático medio *(MSE)*, en la tabla 1 podemos ver los valores obtenidos, aplicando cada uno de los métodos sobre cada conjunto de datos, donde el conjunto de datos *Datos* representa el conjunto de datos dado en el segundo ejercicio del guión de prácticas.

Aunque se podrían haber realizado comparativas mas complejas y significativas, se ha optado por realizar una comparativa sencilla, por la sencillez del propio algoritmo desarrollado.

Conjunto de datos	**Método**	**Error Cuadrático Medio**
Datos	Método 2	0,0084
Datos	Método 3	0,0432
Diabetes	Método 2	3849
Diabetes	Método 3	66,1258
Hyderabad	Método 2	170177318360412
Hyderabad	Método 3	175262607737574,06

Tabla 1: MSE por método y conjunto de datos

De igual forma, podemos ver los valor de Error Absoluto medio obtenido en ambos métodos en la tabla 2.

Conjunto de datos	**Método**	**Error Absoluto Medio**
Datos	Método 2	0,08295202459
Datos	Método 3	0,1657
Diabetes	Método 2	6830,8524
Diabetes	Método 3	50
Hyderabad	Método 2	9573613,9355
Hyderabad	Método 3	9844801,7905

Tabla 2: MAE por método y conjunto de datos

Una primera conclusión notable es la obvia diferencia entre conjuntos de datos, tanto en dimensionalidad, variando de 2 a 39 variables, y en número de patrones, variando de 7 a más de 4 mil patrones. Para el conjunto de datos mas pequeño, donde se tiende al sobreaprendizaje de los patrones, el método del gradiente descendiente sobre el Error Cuadrático, obtenemos valores de error muy pequeños, menores que en el caso del Error Absoluto. Esto se puede explicar por el propio calculo del error cuadrático, y la pequeña diferencia entre las predicciones realizadas y los valores reales.

Al aumentar el número de patrones en el conjunto de datos, aumentado así la complejidad, el modelo es mucho menos preciso, aumentando la diferencia entre predicciones y valores reales, y por lo tanto, aumentando el valor de los errores. Aún así, el error absoluto resulta ser mejor en ambos métodos, hecho que se explica con el cálculo del error cuadrático medio, pues las diferencias entre predicciones dan un gran peso a las predicciones incorrectas en la medida del error.

Se debe destacar el tratamiento de datos aplicado sobre el conjunto de datos *Hyderabad*, pues este contenía valores perdidos cuyos patrones se han decidido eliminar, variables nominales que se han tenido que convertir en numéricas, y variables muy desiguales en cuanto a orden de proporción, que se han optado por eliminar. Aún así, se ha seguido usando la variable *Area*, que por su orden de magnitud, sigue elevando el valor de su peso hasta infinito a medida que avanzan las iteraciones.
Para evitar eliminar demasiada información y crear así un modelo impreciso, no se han eliminado más patrones ni variables, al coste de tener un modelo muy impreciso y mal optimizado.

Estas medidas del error para ambos métodos se pueden visualizar gráficamente en las figuras 1 y 2.

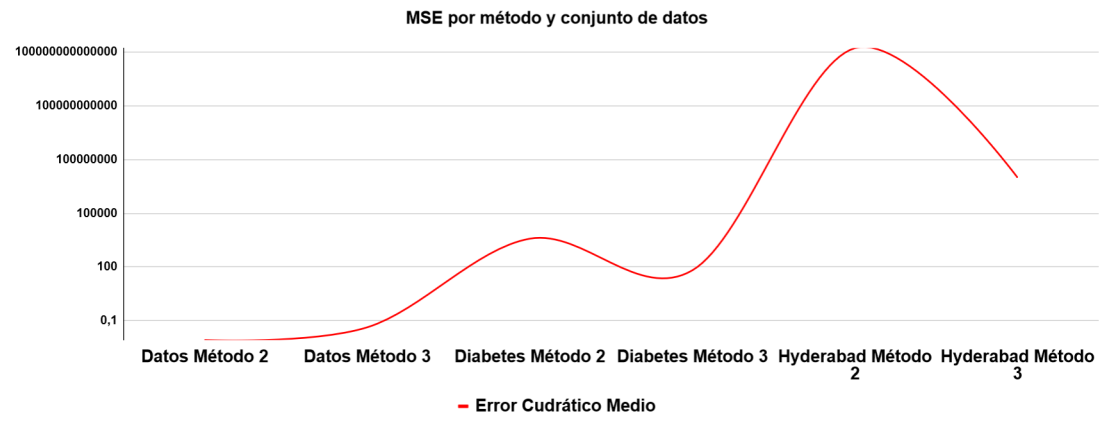

Figura 1: MSE por método y conjunto de datos

Entre ambas gráficas, podemos ver que el error absoluto tiene una evolución constante entre métodos y conjunto de datos, siendo siempre mayor en el método relativo al error absoluto que al del error cuadrático, mientras que el error cuadrático va variando, manteniéndose generalmente menor en el método relativo al error absoluto.

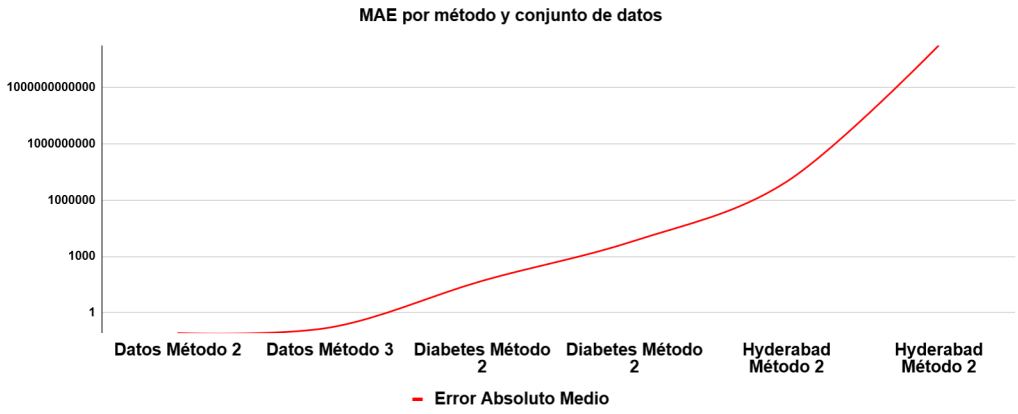

Figura 2: MAE por método y conjunto de datos

1.3. Utiliza ahora las funciones propias de Scikit-learn para resolver el mismo problema y compara los resultados. Compara también la eficiencia de este método con el que has implementado, midiendo varias veces la ejecución de este y comparando con tu implementación

En cuanto a Scikit-learn, se ha usado la implementación propia de la librería, tanto usando como función de error el error cuadrático como el absoluto medio. Los resultados se pueden visualizar en la tabla 3.

Conjunto de datos	Medida del error	Valor del error
Datos	MSE	0,0
Datos	MAE	0,1
Diabetes	MSE	2859,7
Diabetes	MAE	43,3
Hyderabad	MSE	41722278489061,039062
Hyderabad	MAE	3602196,445749

Tabla 3: Error por conjunto de datos

Estas medidas del error para cada conjunto de datos se pueden visualizar gráficamente en la figura 3.

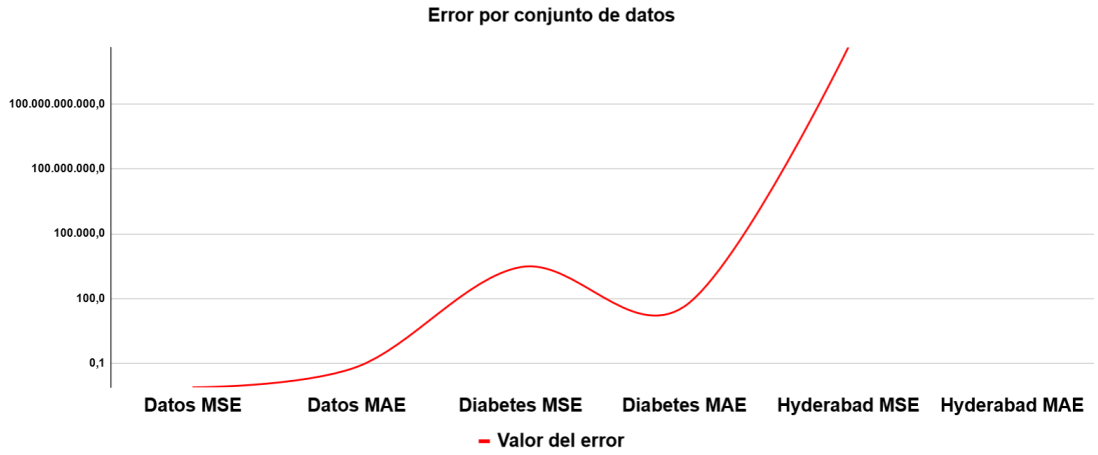

Figura 3: Error por conjunto de datos

Como conclusiones, podemos ver la gran diferencia de rendimiento con respecto a nuestra propia implementación de los métodos.
En el conjunto de datos mas pequeños, el modelo implementado por scikit-learn realiza unas predicciones casi sin error, mientras que en los conjuntos de datos de mayor tamaño, aunque con errores significativos por la escala de los datos y las diferencias entre predicciones y datos reales, consigue mejorar ampliamente nuestro rendimiento, obteniendo resultados mejores en varios órdenes de unidad.
Además, cabe destacar que estos resultados se obtienen aplicando menos iteraciones, y por lo tanto, gastando menos recursos y usando un menor tiempo de cómputo.

1.4. Ahora vas a modificar el código desarrollado en los ejemplos anteriores para que ambas funciones de error incluyan, además del error cuadrático, un término de regularización L1 y L2. Es decir, vas a implementar los métodos de regresión lasso y ridge, y aplicarás la función sobre los datos del conjunto diabetes.

Una vez aplicada nuestra implementación, solo ha sido necesario calcular los términos L1 y L2 respectivamente, y agregárselo a la respectiva medida del error teniendo en cuenta un valor estándar para el factor de regularización de *0,1*.

Los resultados obtenidos pueden verse en la tabla 4.

Conjunto de datos	Método 2 L1 MSE	Método 2 L2 MSE	Método 3 L1 MAE	Método 3 L2 MAE
Data	0,9085	4,2853	2,0040	5,2997
Diabetes	3850,3868	3854,7521	68,6844	84,4273
Hyderabad	69916729275,1596	69917365254,4498	4048,5012	4048,5015

Tabla 4: Regresión por método y conjunto de datos

Estas diferencias también se pueden apreciar gráficamente en la figura 4.

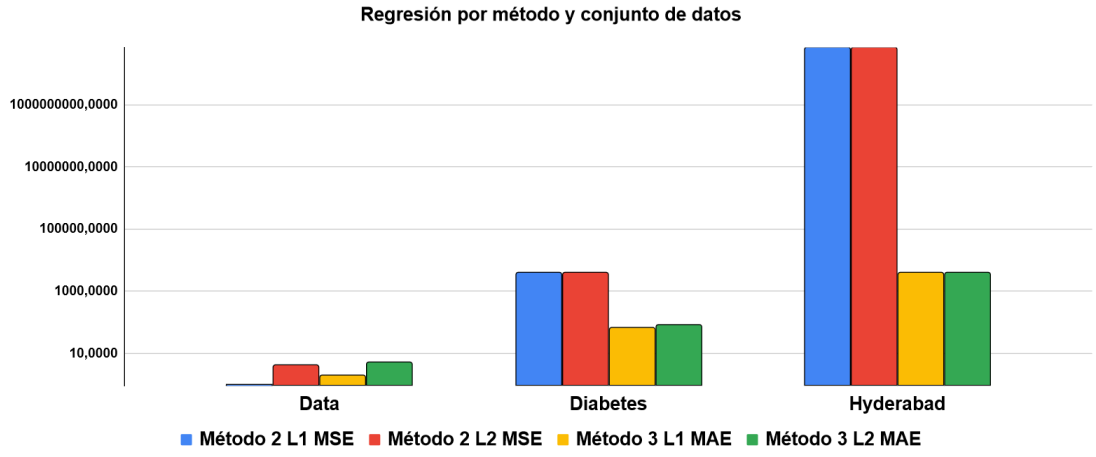

Figura 4: Regresión por método y conjunto de datos

En el conjunto de datos más pequeño, vemos que no mejoramos la medida del error, es más, la aumentamos, cosa lógica entendiendo el cálculo realizado en la regularización, tanto L1 como L2. También, por el cálculo de la regularización Ridge, el valor del error es mucho mas alto al aplicar esta.

En los conjuntos de datos de mayor tamaño, podemos ver que tanto la regularización L1 como la L2 llegan a valores de error practicamente iguales, al tratar con un mayor número de variables, y pesos asociados con valores más dispares. Aunque en el conjunto de datos *Diabetes* no conseguimos mejorar el MSE al aplicar el método 3, y consiguiendo un valor del MSE similar al aplicar el método 2, en el conjunto de datos *Hyderabad* conseguimos mejorar tanto el MSE como el MAE en varias unidades de magnitud, aplicando los métodos respectivos.

1.5. Utiliza ahora las funciones propias de Turi Create y Scikit-learn para realizar las regresiones lasso, ridge y elastic net.

Al igual que en la pregunta anterior, tanto la regularización Lasso, Ridge como la ElasticNet ya están implementadas en la librería, solo hace falta ejecutarlas. Los resultados de las pruebas se pueden ver en la tabla 5.

Conjunto de datos	Medida del error	Regularización	Valor del error
Data	MSE	Lasso	0,128739
Data	MSE	Ridge	0,107433
Data	MSE	ElasticNet	0,288458
Data	MAE	Lasso	0,021259
Data	MAE	Ridge	0,01484
Data	MAE	ElasticNet	0,120116
Diabetes	MSE	Lasso	43,294603
Diabetes	MSE	Ridge	43,277366
Diabetes	MSE	ElasticNet	43,935585
Diabetes	MAE	Lasso	2860,290116
Diabetes	MAE	Ridge	2859,705289
Diabetes	MAE	ElasticNet	2859,705289
Hyderabad	MSE	Lasso	3602197,906
Hyderabad	MSE	Ridge	3601987,4
Hyderabad	MSE	ElasticNet	3554910,175
Hyderabad	MAE	Lasso	4,17E+13
Hyderabad	MAE	Ridge	4,17E+13
Hyderabad	MAE	ElasticNet	4,26E+13

Tabla 5: Regularización usando Scikit-learn

Como podemos ver, en el conjunto de datos mas pequeño, conseguimos reducir los valores de error, de nuevo, en varios ordenes de unidad, al igual que pasa que pasa en los conjuntos de datos mas grandes. Esta diferencia es visible en ambas medidas de error, pero en esta implementación y en los 3 conjuntos de datos, la medida que mas se consigue reducir en todos los casos es el error absoluto.
Esta mejora en los valores de los errores medidos se puede explicar a dos factores, las cuidadas optimizaciones, tanto del método como de la regularización, dentro de la librería scikit-learn.

Además, estos resultados se pueden ver gráficamente en la figura 5.

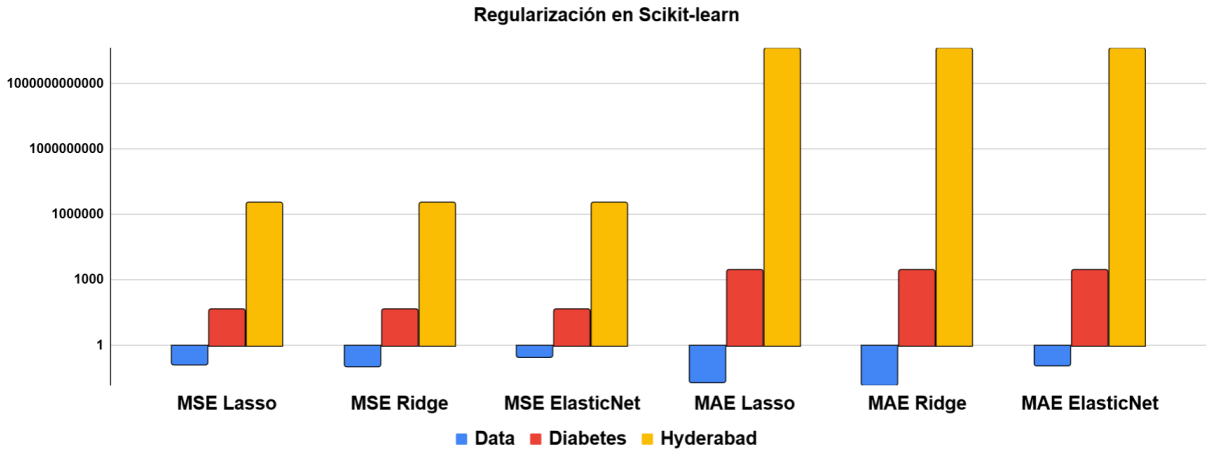

Figura 5: Regularización en Scikit-learn