_error

DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

TP 3

Cuadrados mínimos con regularización con SVD e interpolación de Legendre

Junio-Julio 2024 Métodos Numéricos

Integrante	LU	Correo electrónico
Begalli, Juan Martin	139/22	juanbegalli@gmail.com
Carrillo, Mariano	358/22	carr.mariano@gmail.com
Serapio, Noelia	871/03	noeliaserapio@gmail.com

Resumen

En el presente trabajo implementaremos y experimentaremos con cuadrados mínimos utilizando SVD y estudiaremos el ajuste de curvas mediante polinomios de Legendre.
Desarrollamos dos algoritmos que calculan cuadrados mínimos con y sin regulación.
Además, realizamos otros 2 algoritmos para encontrar el grado del polinomio utilizando cuadrados mínimos sin regulación en un caso y en el otro, los hiperparámetros grado del polinomio y nivel de regulación adecuados con cuadrados mínimos con regulación, utilizando el método de validación cruzada en ambos algoritmos. Luego, analizaremos el comportamiento de dichos algoritmos.

Palabras Clave: *cuadrados mínimos con SVD, regulación, polinomios de Legendre*

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2610 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (+54 +11) 4576-3300
http://www.exactas.uba.ar

Índice

1. Introducción **2**

2. Desarrollo **2**

2.1. Cuadrados mínimos 2

2.2. Ajuste con polinomios de Legendre 4

3. Resultados **5**

3.1. Errores de ajuste y validación 5

3.2. Exploración de hiperparámetros: Grado y valor de regularización 6

4. Conclusiones **8**

Referencias 9

1. Introducción

El método de cuadrados mínimos lineales busca resolver el problema de encontrar la función que "mejor aproxime" a un conjunto de datos. Este conjunto de datos usualmente son pares ordenados (x_i, y_i) y la función a buscar pertenece a una *familia de funciones* $\mathcal{F}$. En particular, la aproximación buscada está dada por:

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{m} (f(x_i) - y_i)^2 \tag{1}$$

Por otro lado, si definimos $\mathcal{F} = \{f(x) = \sum_{j=1}^{n} c_j \phi_j\}$ el problema de cuadrados mínimos lineales (CML) se expresa como:

$$\min_{c_1,\ldots,c_n} \sum_{i=1}^{m} (\sum_{j=1}^{n} c_j \phi_j(x_i) - y_i)^2 \tag{2}$$

A su vez, podemos definir las matrices:

$$\mathrm{A} = \begin{bmatrix} \phi_1(x_1) & \phi_2(x_1) & \ldots & \phi_n(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \ldots & \phi_n(x_2) \\ \vdots & \vdots & \ldots & \vdots \\ \phi_1(x_m) & \phi_2(x_m) & \ldots & \phi_n(x_m) \end{bmatrix} \mathrm{b} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \mathrm{x} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}$$

Y el problema de cuadrados mínimos lineales consiste en buscar

$$\min_{x \in R^n} \|Ax - b\|_2^2 \tag{3}$$

La solución a este problema, en el caso en el que $m > n$, está dada a partir de las *ecuaciones normales* $A^t Ax = A^t b$ [3]. Donde $x = (A^t A)^{-1} A^t b$. Por otro lado, si se conoce la descomposición SVD de $A = USV^t$ y suponemos que el rango de A es completo, podemos reducir a V, S y U de manera que $\mathrm{S} \in R^{nxn}$ eliminando las filas nulas por lo tanto V también será cuadrada y $\mathrm{U} \in R^{mxn}$. Obteniendo así $x = VS^{-1}U^t b$. De lo que deriva que el error en términos de cuadrados mínimos sea $\|Ax - b\|_2^2$

Si quisiéramos que nuestra solución tuviera en cuenta otros aspectos, como por ejemplo penalizar coeficientes c muy grandes, podríamos modificar la ecuación 2. Es decir, el ajuste tendrá en cuenta no solo qué tan buena es la aproximación hallada sino también, qué tan grande es el coeficiente c. Así, podría ocurrir que una solución que sin la regularización era la mínima, deje de serlo al introducir la regularización. Usaremos regularización L2 (regresión *ridge*). De esa forma, el problema queda expresado como

$$\min_{c_1,\ldots,c_n} \sum_{i=1}^{m} (\sum_{j=1}^{n} c_j \phi_j(x_i) - y_i)^2 + \lambda \sum_{j=1}^{n} c_j^2 \tag{4}$$

λ es un parámetro que cuanto más grande sea mayor será la penalización. Despejando x a partir de la expresión de las ecuaciones normales anteriores y utilizando la descomposición SVD de A, llegamos a:

$$x = VS(S^2 + \lambda I)^{-1} U^t b \tag{5}$$

(más detalles en 2.1)

En este trabajo, utilizaremos las funciones polinómicas de Legendre para resolver el problema de cuadrados mínimos. Estos polinomios $P_n(x)$ son adecuados porque forman una base ortonormal y, además, el número de condición de la matriz resultante es "bueno´´. El objetivo será buscar el grado de polinomio y el nivel de regularización adecuados (donde ambos son hiperparámetros). Para ello, se puede realizar validación cruzada con un conjunto de datos que se usará para ajustar, y otro para medir el error de dicho ajuste. Es decir, dado un conjunto de datos X, valores de b y la base de funciones:

$$x = (X_{ajuste}^t X_{ajuste} + \lambda I)^{-1} X_{ajuste}^t b_{ajuste} \tag{6}$$

$$ECM_{ajuste} = \|X_{ajuste} x - b_{ajuste}\|_2^2 \tag{7}$$

$$ECM_{val} = \|X_{val} x - b_{val}\|_2^2 \tag{8}$$

2. Desarrollo

2.1. Cuadrados mínimos

En este caso si conocemos la descomposición SVD de X de manera que $X = USV^t$ y si X tiene rango completo (rg(X) = minm,n = n), es decir que U y S pueden reducirse para que S quede cuadrada eliminando las filas nulas entonces podemos expresar la solución de cuadrados mínimos de la siguiente manera:

$\beta = VS^{-1}U^t y$

Luego para realizar el algoritmo creamos una función llamada *cuadrados_minimos* que recibe por parámetro la matriz X y el vector *y*. Usamos la función *np.linalg.svd()* para obtener las matrices U y V^t y un vector con los valores singulares. Después calculamos la matriz S^{-1} usando la propiedad de que cuando una matriz es diagonal la inversa son los valores de la diagonal inversos y finalmente calculamos el β.

in: $X \in R^{m \times n}$, $y \in R^m$, **out:** β

cuadrados_minimos

1: $U, s, V^t \leftarrow SVD(X)$
2: $S^{-1} \leftarrow diagonal(1/s)$
3: $\beta \leftarrow VS^{-1}U^t y$
4: **return** β

Para resolver el problema de cuadrados mínimos con regularización podemos usar la descomposición SVD para reemplazar en la expresión $\beta = (X^t X + \lambda I)^{-1} X^t y$. Para este despeje hay que tener en cuenta que $U^t U = I$, $V^t V = I$ y $S^t = S$, y que en S se eliminaron las filas nulas.

Analicemos más en detalle esto. X es una matriz de $R^{m \times n}$ $(m > n)$, por lo tanto, su descomposición SVD "completa" $(X = USV^t)$ tiene las siguientes dimensiones: $S \in R^{m \times n}, U \in R^{m \times m}, V \in R^{n \times n}$. Ahora bien, si analizamos la teoría detras de la descomposición [2] y tomando rg(X)=r, v_i, u_i las columnas de V y U respectivamente, y σ_i $(1 \leq i \leq r)$ los valores singulares de X. Notamos que se cumplen las siguientes 4 ecuaciones:

$$\begin{aligned} Xv_i &= \sigma_i u_i \quad && i = 1, ..., r \\ Xv_i &= 0 && i = r+1, ..., n \\ X^t u_i &= \sigma_i v_i && i = 1, ..., r \\ X^t u_i &= 0 && i = r+1, ..., m \end{aligned}$$

Por lo tanto, si suponemos que el rango de X es máximo (rg(X)=n pues $m > n$) y utilizando la primer ecuacion, llegamos a que:

$$Xv_i = \sigma_i u_i \quad 1 \leq i \leq n \tag{9}$$

Usando esto, se puede plantear una descomposicion SVD reducida (*Reduced SVD*) [1]. Tal que $XV = \hat{U}\hat{S}$, donde en $\hat{U}$ solo están las primeras n columnas pues las columnas $n+1, ..., m$, por lo visto en las ecuaciones de SVD, al ser multiplicadas por X^t daban 0. Al mismo tiempo, $\hat{S}$ es una matriz diagonal con los n valores singulares en su diagonal, es decir, se eliminaron las filas nulas. Por último, V quedaría igual que la descomposición original, pues tenía exactamente n columnas y filas. Las dimensiones de las matrices en la nueva SVD (la reducida) son: $S \in R^{n \times n}, U \in R^{m \times n}, V \in R^{n \times n}$. Puedo reducir el tamaño de U porque establecimos que el $rg(X) = n$ y $m > n$ y U era de $m \times m$. Notar que al realizar la multiplicacion (USV^t) usando estas nuevas matrices el resultado es una matriz de $m \times n$, pues X tenía esas dimensiones.

Observamos que al plantear la descomposición SVD reducida, no "pierdo información" con respecto a la descomposición SVD completa, es decir, es equivalente a la completa. Porque, teniendo en cuenta las consideraciones anteriores y observando las ecuaciones, se concluye que al eliminar las columnas $n+1, ..., m$ de U no se altera el resultado de US pues las columnas $(u_{n+1} \times 0, ..., u_m \times 0)$ de dicho resultado daban 0. Entonces es válido solo tomar las primeras n columnas de U, y las primeras n filas de S (eliminando las filas nulas).

Otra forma de demostrar esto es tomando a las matrices de la descomposicion SVD original en bloques. Basandonos fuertemente en los resultados de las ecuaciones de la descomposicion SVD: Siendo r el rango de X, divido a U en dos bloques U_r, U_{m-r} el primero de $m \times r$ y el segundo de $m \times m - r$. Divido a V en dos bloques V_r, V_{n-r} el primero de $n \times r$ y el segundo de $n \times n - r$. Y a S en cuatro bloques, donde el primero (S_r) es un bloque de $r \times r$ (el bloque con los valores singulares) y el resto son bloques de 0s con las dimensiones apropiadas. Quiero ver que partiendo de la descomposicion SVD completa puedo llegar a una descomposición SVD reducida equivalente, de la forma $X = U_r S_r V_r^t$. Es decir, la igualdad $X = USV^t$ quedaría:

$$\begin{aligned} & [U_r \quad U_{m-r}] \begin{bmatrix} S_r & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_r^t \\ V_{n-r}^t \end{bmatrix} \\ &= [U_r \quad U_{m-r}] \begin{bmatrix} S_r V_r^t \\ 0 \end{bmatrix} \\ &= U_r S_r V_r^t \end{aligned}$$

Llegando a lo que se buscaba probar[1].

[1] En "Numerical Linear Algebra, Lloyd N. Trefethen, SIAM, 1997", página 25-28, se explica el porqué usar una descomposición completa contra una reducida y sus diferencias a nivel matemático. Pero escapa al scope de este trabajo.

Finalmente, el despeje de β quedaría así:

$\beta = ((USV^t)^t USV^t + \lambda I)^{-1}(USV^t)^t y$

$\beta = (VS^t U^t USV^t + \lambda I)^{-1} VS^t U^t y$

$\beta = (VS^t SV^t + \lambda I)^{-1} VS^t U^t y$

$\beta = (VS^t SV^t + \lambda I)^{-1} VS^t U^t y$

$\beta = (VS^2 V^t + \lambda I)^{-1} VSU^t y$

$\beta = (VS^2 V^t + \lambda VV^t)^{-1} VSU^t y$

$\beta = (V(S^2 + \lambda I)V^t)^{-1} VSU^t y$

$\beta = V(S^2 + \lambda I)^{-1} V^t VSU^t y$

$\beta = V(S^2 + \lambda I)^{-1} SU^t y$

$\beta = VS(S^2 + \lambda I)^{-1} U^t y$

Notar que $(S^2+I)^{-1}$ es diagonal(pues S e I son diagonales y la inversa de su suma también). Además, S es diagonal. Entonces hacer $(S^2+I)^{-1}S$ es lo mismo que hacer $S(S^2+I)^{-1}$ por propiedad de las matrices diagonales. De ahi la justificación del ultimo paso del despeje.

El algoritmo de cuadrados mínimos con regulación (*cuadrados_minimos_reg*) recibe por parámetros a la matriz X y al vector y como en el anterior pero además recibe un λ quien determina que tan fuerte es la penalización cuando el valor de β es arbitrariamente grande. Además hay que tener en cuenta que $S^2 + \lambda I$ es una matriz diagonal por lo tanto calcular su inversa es simplemente calcular la inversa de los elementos de su diagonal.

Finalmente usamos esto tres parámetros para calcular la solución β de la expresión anterior.

in: $X \in R^{m \times n}$, $y \in R^m$, $\lambda \in float$, **out:** β

cuadrados_minimos_reg

1: $U, s, V^t \leftarrow SVD(X)$
2: $S \leftarrow diagonal(s)$
3: $R \leftarrow S^2 + \lambda I$
4: $R^{-1} \leftarrow diagonal(1/diagonal(R))$
5: $\beta \leftarrow VSR^{-1}U^t y$
6: **return** β

2.2. Ajuste con polinomios de Legendre

Cuando tenemos un conjuntos de datos de entrenamiento que llamaremos *datos_ajuste* y la base de funciones (en nuestro caso, polinomios de Legendre), es posible encontrar la mejor solución en términos de cuadrados mínimos. Dado un nuevo conjunto de datos *datos_val* los cuales provienen de la misma distribución de los datos de entrenamiento. Si se utiliza un grado de polinomio muy alto, o no se utiliza regularización, el error del ajuste aumenta al probar los nuevos datos. Por eso, a la hora de encontrar el grado del polinomio y el nivel de regularización adecuados, una buena práctica suele ser ajustar con ciertos datos y evaluar el error del ajuste en datos no utilizados anteriormente. Este proceso se llama optimización de hiperparámetros con validación cruzada. El procedimiento quedaría:

$$\beta = (X^t_{ajuste} X_{ajuste} + \lambda I)^{-1} X^t_{ajuste} y_{ajuste}$$
$$ECM_{ajuste} = \|X_{ajuste}\beta - y_{ajuste}\|_2^2$$
$$ECM_{val} = \|X_{val}\beta - y_{val}\|_2^2$$

Para ajustar una combinación lineal de polinomios de Legendre (sin regularización) en los datos de ajuste y predecir los valores de validación, calculando los respectivos errores de ajuste y validación realizamos el siguiente algoritmo:

in: $grados \in N^n$ **out:** $resultados\_ajuste$, $resultados\_val$

ajuste_error

$resultados\_val \leftarrow \emptyset$, $resultados\_ajuste \leftarrow \emptyset$
for $grado \in grados$ **do**
  $X\_val \leftarrow legender(datos\_val.x, grado)$
  $X\_ajuste \leftarrow legender(datos\_ajuste.x, grado)$
  $\beta \leftarrow cuadrados\_minimos(X\_ajuste, datos\_ajuste.y)$
  $y\_ajuste\_pred \leftarrow X\_ajuste\beta$
  $y\_val\_pred \leftarrow X\_val\beta$
  $error\_val \leftarrow ECM(datos\_val.y, y\_val\_pred)$
  $error\_ajuste \leftarrow ECM(datos\_ajuste.y, y\_ajuste\_pred)$
  $resultados\_val.agregar((grado, error\_val))$
  $resultados\_ajuste.agregar((grado, error\_ajuste))$
  return $resultados\_ajuste, resultados\_val$
end for

Donde para calcular *legender* utilizamos la función de numpy *np.polynomial.legendre.legvander*.

Con respecto al error de validación (principalmente), lo más intuitivo sería pensar que a medida que el grado del polinomio aumenta el error sea menor. La disminución del error no necesariamente debería ser lineal en función del grado del polinomio. Por eso mismo, se decidió explorar polinomios de grado 1 hasta grado 2n, siendo n la dimensión de los datos.

En la sección 3.1 se analizan los resultados obtenidos.

Luego, realizamos una **exploración de los hiperparámetros** grado del polinomio y valor de regularización λ, utilizando polinomios de Legendre, el algoritmo que realiza dicha exploración es:

in: $grados \in \mathbb{N}^n$, $lambdas \in \mathbb{R}^n$ **out:** $resultados\_ajuste$, $resultados\_val$

exploracion_hiperparametros

$resultados\_val \leftarrow \emptyset$, $resultados\_ajuste \leftarrow \emptyset$
for $grado \in grados$ **do**
  for $l \in lambdas$ **do**
    $X\_val \leftarrow legender(datos\_val.x, grado)$
    $X\_ajuste \leftarrow legender(datos\_ajuste.x, grado)$
    $\beta \leftarrow cuadrados\_minimos\_reg(X\_ajuste, datos\_ajuste.y, l)$
    $y\_ajuste\_pred \leftarrow X\_ajuste\beta$
    $y\_val\_pred \leftarrow X\_val\beta$
    $error\_val \leftarrow ECM(datos\_val.y, y\_val\_pred)$
    $error\_ajuste \leftarrow ECM(datos\_ajuste.y, y\_ajuste\_pred)$
    $resultados\_val.agregar((grado, error\_val))$
    $resultados\_ajuste.agregar((grado, error\_ajuste))$
    return $resultados\_ajuste, resultados\_val$
  end for
end for

Tanto para extraer los datos de la fuente dada como para organizar y operar con los mismos se utilizo el paquete *Pandas* por una cuestión de practicidad.

3. Resultados

3.1. Errores de ajuste y validación

Se exploraron los errores de ajuste y validación utilizando como medida al error cuadrático medio (ECM). Obtuvimos, para cada grado del polinomio (grados desde 1 hasta 2 veces a dimensión de los datos), el error entre el valor de validación predicho y el original. Como se observa en la figura 3.1 para los datos de validación se obtuvo un error muy grande cuando el grado del polinomio estaba entre 40 y 60 aproximadamente. Mientras que para grados mayores o grados muy chicos, el error en esos datos era menor (y más estable). Este fenómeno estadístico se denomina *double descent* y se caracteriza porque el modelo presenta un error chico cuando la cantidad de parámetros (en este caso el grado del polinomio) es o muy grande o muy chica. Y cuando la cantidad de parámetros es similar a la cantidad de datos (50 en este caso) el error de validación aumenta considerablemente.

Por otro lado, el error de ajuste disminuye en gran medida cuando el grado del polinomio es mayor a 50 aproximadamente. Esto tiene sentido, pues si estamos ajustando con una gran cantidad de parámetros, nuestro modelo se irá acercando cada vez más al valor de ajuste conocido. Por eso mismo, la validación cruzada permite ir ajustando los hiperparámetros para que el modelo no quede sesgado solamente por los datos de ajuste. Y como consecuencia, sea lo más generalizable posible a datos desconocidos (no usados para ajustar).

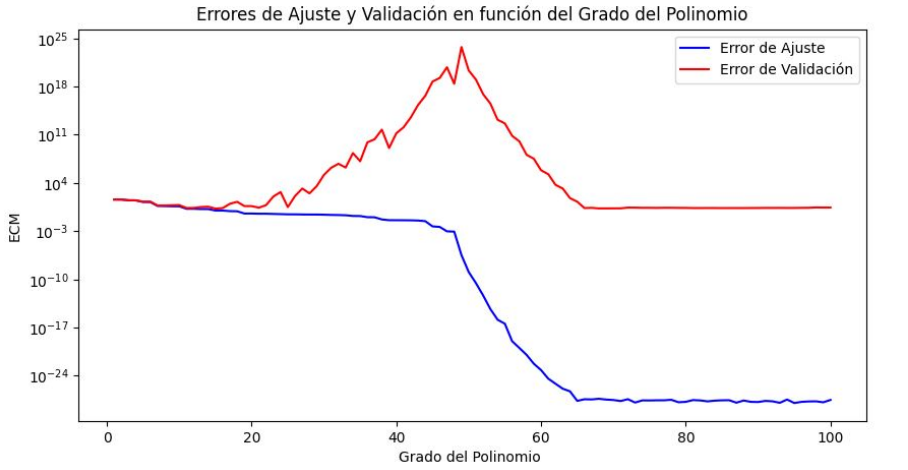

Figura 1: Errores de ajuste y validacion en funcion del grado del polinomio

El gráfico de los errores, al presentar el fenómeno de *double descent* se desvió un poco de lo que habíamos pensado antes de realizar el experimento. El hecho de que cuando el grado del polinomio sea parecido a la cantidad de datos el error sea muy grande resulta algo antiintuitivo. Pues uno esperaría que cuanto menor sea el grado, mayor sea el error. Y que vaya disminuyendo (no necesariamente linealmente) a medida que aumenta el grado.

3.2. Exploración de hiperparámetros: Grado y valor de regularización

Realizamos una primera exploracion de los valores de λ tomando 130 valores logaritmicamente espaciados de manera homogenea, entre $\{10^{-7}, 10^{2}\}$. Obtuvimos los resultados que se ven en la figura 2. Podemos observar como los máximos ECM se encuentran en la región de alto grado polinomial (mayor a 30 aprox.) y bajo coeficiente de regularización, es decir un λ bajo. Al tener un coeficiente bajo es entendible que haya un sobreajuste en los datos de entrenamiento y por eso tiene un ECM mayor. Nos damos cuenta de esto, también, ya que el mejor y peor error (discutidos al final de esta sección) poseen el mismo grado pero lo que varia es el coeficiente de regularización.

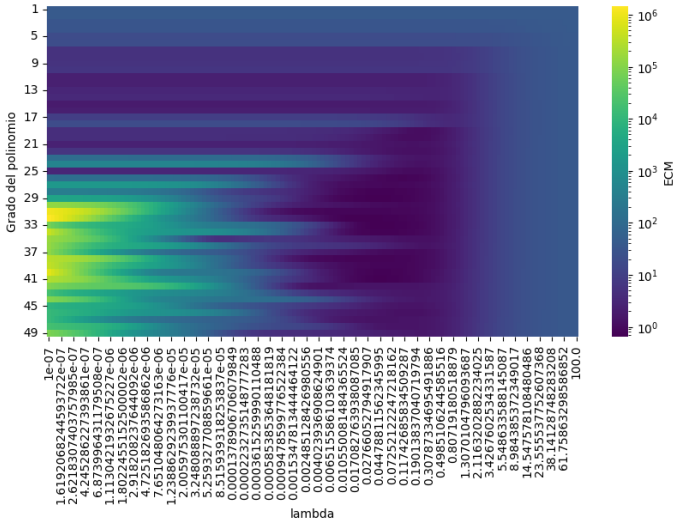

Figura 2: Error de validación para la primer grilla de hiperparámetros explorada.

Otra cosa interesante del heatmap es ver como a partir de un grado alto como el mencionado anteriormente, se comporta según lo esperado (bajo ECM) pero luego de llegar a una región óptima empieza a aumentar el ECM a pesar de que el coeficiente de regularización siga creciendo. Por lo tanto vemos que tampoco conviene que el λ sea muy grande.

Además vemos que en grados bajos, entre 1 y 6 más o menos, el error cuadrático es constante a lo largo de todos los valores

de los coeficientes lo cual tiene sentido pues con un grado bajo comparado con el tamaño de los datos el ajuste no va a ser de los mejores.

Por último observamos como para valores arbitrariamente grandes(más grande que el valor óptimo encontrado) de λ el ECM se mantiene constante para cualquier grado.

Para complementar lo anteriormente mencionado, realizamos y graficamos dos exploraciones más del hiperparametro λ. Observando la figura 2 notamos que la region con menor error se encontraba, aproximadamente, entre $\lambda = 0{,}001$ y $\lambda = 0{,}1$. La segunda exploración se realizó entre dichos valores obteniendo el gráfico de la figura 3.

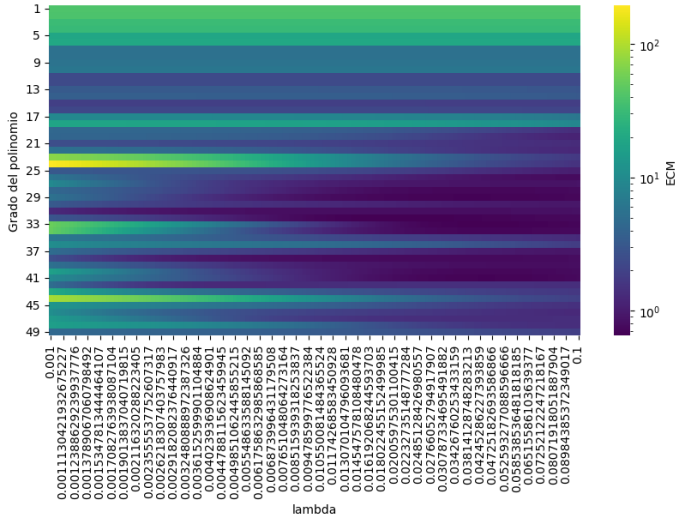

Figura 3: Error de validacion para la segunda grilla de hiperparametros explorada

Luego de esta exploración, ajustamos aún más el rango de busqueda para el valor de λ. Se observa que los valores con menor error se encuentran aproximadamente entre $\lambda = 0{,}01$ y $\lambda = 0{,}05$. Así que, de nuevo, tomamos 130 puntos entre esos valores con los mismos criterios que antes. El resultado de la tercer exploración se muestra en la figura 4

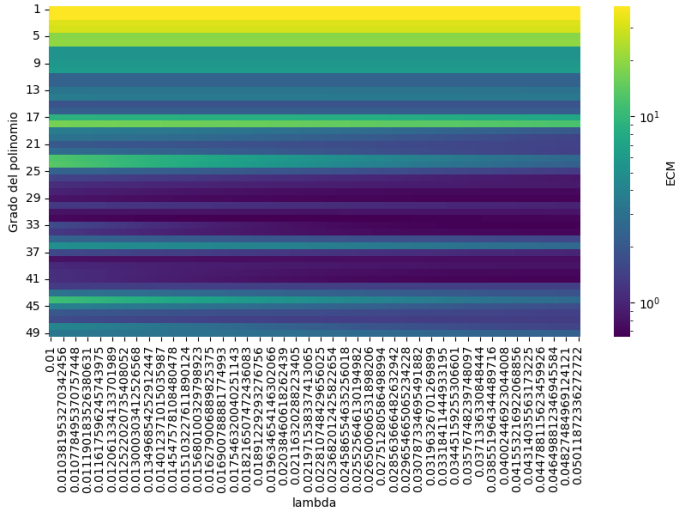

Figura 4: Error de validacion para la tercer grilla de hiperparametros explorada

En esta última exploración se observa como el error general con todos los valores de λ es menor al obtenido con algunos de los anteriores valores explorados. Por lo tanto, se puede concluir, a partir de este gráfico, que los mejores valores de λ para las condiciones dadas estarán en este rango. Por otro lado, el mejor grado del polinomio se observa, tanto en este grafico como en los anteriores, entre 29 y 33 aproximadamente. Consideramos que este rango era lo suficientemente acotado como para no explorar otra vez. Y de esta forma, se ve como iterativamente aproximamos el mejor valor de λ

Por último, para corroborar nuestras hipotesis, utilizamos los datos obtenidos en la primera exploracion pues consideramos era la mas representativa tanto para los mejores como los peores valores. Asi, determinamos los mejores valores de λ y el grado del polinomio. El mejor error obtenido fue con $\lambda \approx 0,02$ y grado $= 32$ y el peor fue con $\lambda = 10^{-7}$ y grado $= 32$. Notar que si quisieramos solamente el mejor λ, podríamos encontrar el valor más preciso analizando los λ de la última exploración. Esto último porque es un rango de valores muy acotado, comparado con las primeras dos exploraciones.

4. Conclusiones

Tras implementar la solución de cuadrados mínimos sin y con regularización usando la descomposición SVD, y la regresión polinomial utilizando los polinomios de Legendre, podemos concluir que es una manera efectiva de optimizar la capacidad predictiva de un modelo de ajuste.

Por otro lado, haber explorado los errores en función del grado utilizado fue importante para observar cómo se comportaba el error en cada caso. Porque de no haberlo hecho, no se habría observado que utilizar un grado de polinomio parecido a la cantidad de datos era la peor opción.

Finalmente gracias a la exploración de los hiperparámetros nos dimos cuenta como un coeficiente de regularización bajo no significa que el resultado sea mejor, sino que es todo lo contrario. De esta misma manera cuando tenemos un coeficiente muy grande el error cuadrático medio aumenta por lo tanto lo mejor es usar un coeficiente óptimo que este entre medio de estos valore. Por último pudimos confirmar lo que mencionamos en el párrafo anterior, los mejores valores no se encuentran cuando el grado del polinomio es parecido a la cantidad de datos. Pero tampoco conviene que el grado sea muy bajo pues el error se mantiene constante y bastante elevado en estos casos sin importar el coeficiente de regularización. Por lo tanto, al igual que con el coeficiente de regularización, lo mejor es usar un valor intermedio.

Referencias

[1] Lloyd N. Trefethen. *Numerical Linear Algebra*. SIAM, 1997.

[2] J. Douglas Faires Richard L. Burden. *Análisis numérico*. International Thomson Editores, 2002.

[3] Timohty Sauer. *Numerical Analysis*. Pearson, 3rd Edition, 2017.