Memoria de Prácticas de Aprenentatge Computacional:

Practica 1 - Regresión

Raül Salinas Natal	1497706
Juan Manuel Camara Diaz	1566532
Juan Aguilera Toro	1566837

Índice

Esta primera práctica de la asignatura de Aprenentatge Computacional nos ha pedido trabajar un código desde un punto de vista de aplicación de modelos regresivos y del análisis de los datos obtenidos mediante este estudio. Estudiar la necesidad de normalizar datos para obtener resultados concluyentes, hacer un estudio de si realmente era adecuado el nivel de error con el que nos encontrábamos y que, si no fuese ese el caso, deberíamos plantear el problema que nos surgía de otra manera para poder llegar a un nivel deseado. Visualizar una gran cantidad de gráficas y sus datos, debido a que una presentación gráfica de los objetos con los que trabajamos comporta una mayor facilidad de observación de la información. Estas gráficas también permiten observar errores, debido a que un error se puede detectar muy fácilmente si se representa en una gráfica adecuada.

El tema principal de la práctica es la regresión. La tratamos en tres apartados diferentes, los cuales nos harán practicar este tema de diferentes maneras con la regresión y el descenso del gradiente como objetivo final.

Apartado C

Introducción

El primer apartado es el C, el cual nos introduce al análisis de datos. Aquí tratamos directamente con la base de datos que se nos ha otorgado como grupo. Siendo nosotros el grupo GPA606 la siguiente base de datos es la que se nos concedió y con la que hemos estado trabajando → [https://www.kaggle.com/noordeen/insurance-premium-prediction]. Esta base de datos se llama Insurance Premium Predicction, es decir, Predicción de Premium a Seguros… ¿pero que es este “Premium”? Nosotros tenemos un conjunto de datos, que culmina en determinar si cobrar un extra que es lo que llamamos Premium. Aunque la premisa de la base de datos es la explicada anteriormente, nosotros hemos enfocado el siguiente dataset como un problema de regresión.

Cada tupla de la base de datos representa a una persona, y de esta se nos dan muchos datos útiles que nos sirven para entender si esta persona ha de pagar o no el Premium en su seguro médico. En cada tupla contamos con 4 valores numéricos y 3 nominales. De cada persona guardamos su género, si es fumador o no y en qué región vive esa persona, estos datos son los que se representan mediante Dtype object. Los siguientes datos son los que se representan con variables numéricas, dónde vemos el Índice de Masa Corporal de cada persona, así como la cantidad de hijos que tienen. A estos datos también se le suman la edad de cada persona y el gasto acumulado en ámbitos médicos.

Una de las cosas de mayor interés que se nos pide el apartado C es que decidamos sobre cual creemos que pueda ser el Atributo Objetivo, queriendo decir, el atributo más representativo del problema y que causará mayor interés a la hora de ser tratado para obtener resultados concluyentes. Releyendo el contenido que hallamos en la página Kaggle, vemos que en el apartado de la inspiración y objetivo de este ejercicio se menciona como desean comparar todos los datos personales de cada tupla de la base de datos contra el valor de “expenses” que tiene acumulado cada persona en gastos médicos. Ese fragmento de texto nos parece muy interesante para enfocar el problema de una manera determinada, y ha hecho que nos planteemos la práctica con “Expenses” como nuestro Atributo Objetivo.

Estudio Base de Datos

Lo primero que hicimos nosotros en este momento fue cargar la base de datos y ver como se encontraban ordenadas las variables y como estaba guardada la información. Mediante la librería pandas y la función read_csv() obtuvimos los datos del fichero csv en que estaban guardadas y pudimos observar como estaban guardadas, datos varios sobre los valores que contenían y que tipo de datos eran.

	age	sex	bmi	children	smoker	region	expenses
0	19	female	27.9	0	yes	southwest	16884.92
1	18	male	33.8	1	no	southeast	1725.55
2	28	male	33.0	3	no	southeast	4449.46
3	33	male	22.7	0	no	northwest	21984.47
4	32	male	28.9	0	no	northwest	3866.86

Img 1: Taula mostrant les 5 primeres files de la base de dades

	age	bmi	children	expenses
count	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	30.665471	1.094918	13270.422414
std	14.049960	6.098382	1.205493	12110.011240
min	18.000000	16.000000	0.000000	1121.870000
25%	27.000000	26.300000	0.000000	4740.287500
50%	39.000000	30.400000	1.000000	9382.030000
75%	51.000000	34.700000	2.000000	16639.915000
max	64.000000	53.100000	5.000000	63770.430000

Img 2: Taula resultant de la funció describe() de la llibreria pandas

#	Column	Non-Null Count	Dtype
0	age	1338 non-null	int64
1	sex	1338 non-null	object
2	bmi	1338 non-null	float64
3	children	1338 non-null	int64
4	smoker	1338 non-null	object
5	region	1338 non-null	object
6	expenses	1338 non-null	float64

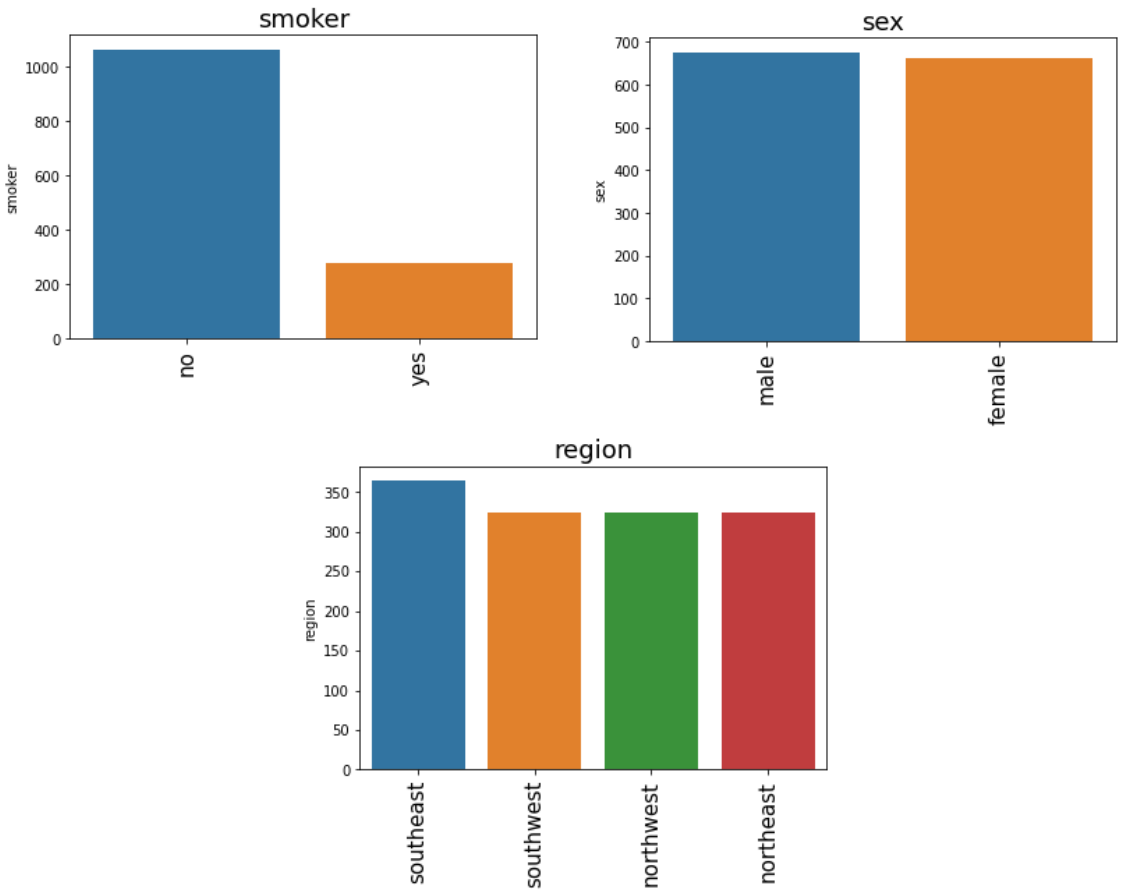
dtypes: float64(2), int64(2), object(3)

Img 3: Taula resultant de la funció info() de pandas, on es mostra els tipus de cada variable

En estas tablas podemos ver que nuestra base de datos se compone principalmente de atributos categóricos y binarios.

Tipos de Variables

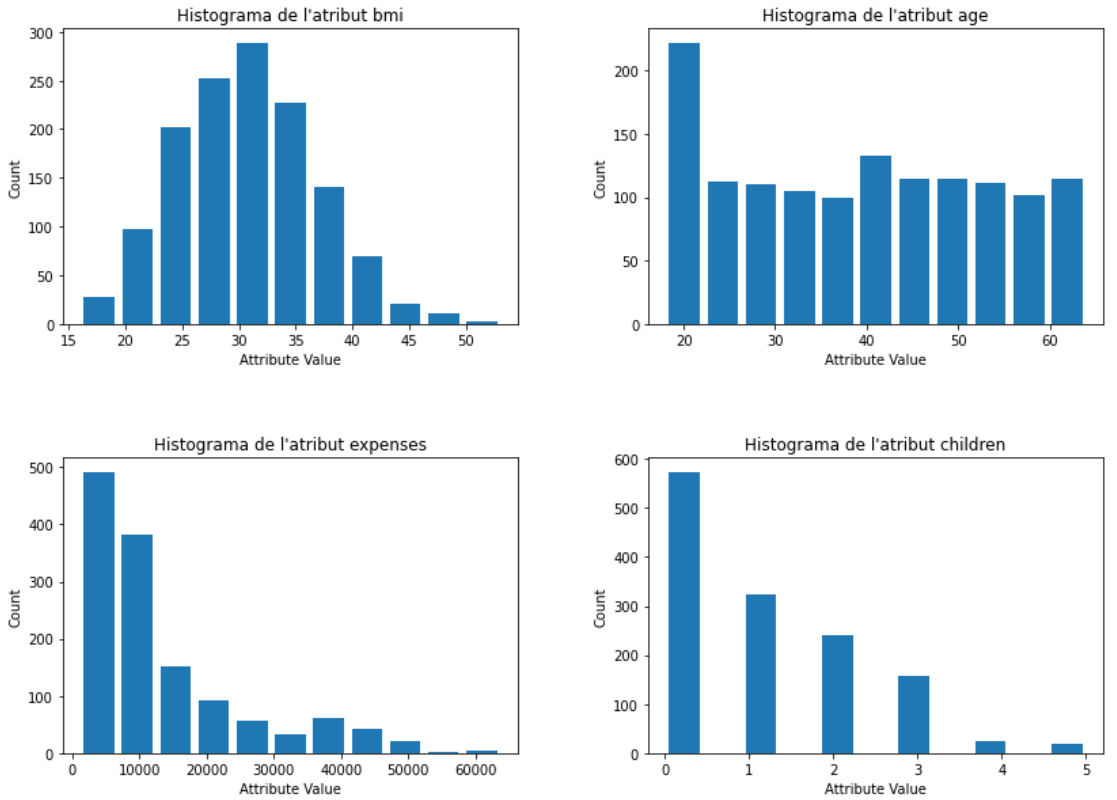
Una vez conocidos los valores que tenemos, lo primero que hicimos fue representar la información en gráficas. Los valores categóricos los representamos como gráficas de barras, mediante barplot_gen(), lo cual resultó en las siguientes gráficas.



Img 4: Gráficas de barras en que observamos la cantidad de veces que se presenta una variable

Vamos que en nuestro dataset tenemos una cantidad superior de gente que no fuma de la que si fuma. El resto de los valores categóricos están compensados.

De la misma manera que las variables categóricas, deberemos representar las no categóricas, pero éstas las representaremos mediante histogramas con la variable hist_gen(). Esto se debe a que la información que guardan estas variables se ven más adecuadamente representadas bajo un histograma, que muestra la cantidad de veces que aparece cada valor y no un simple valor binario como en por ejemplo la variable Fuma.

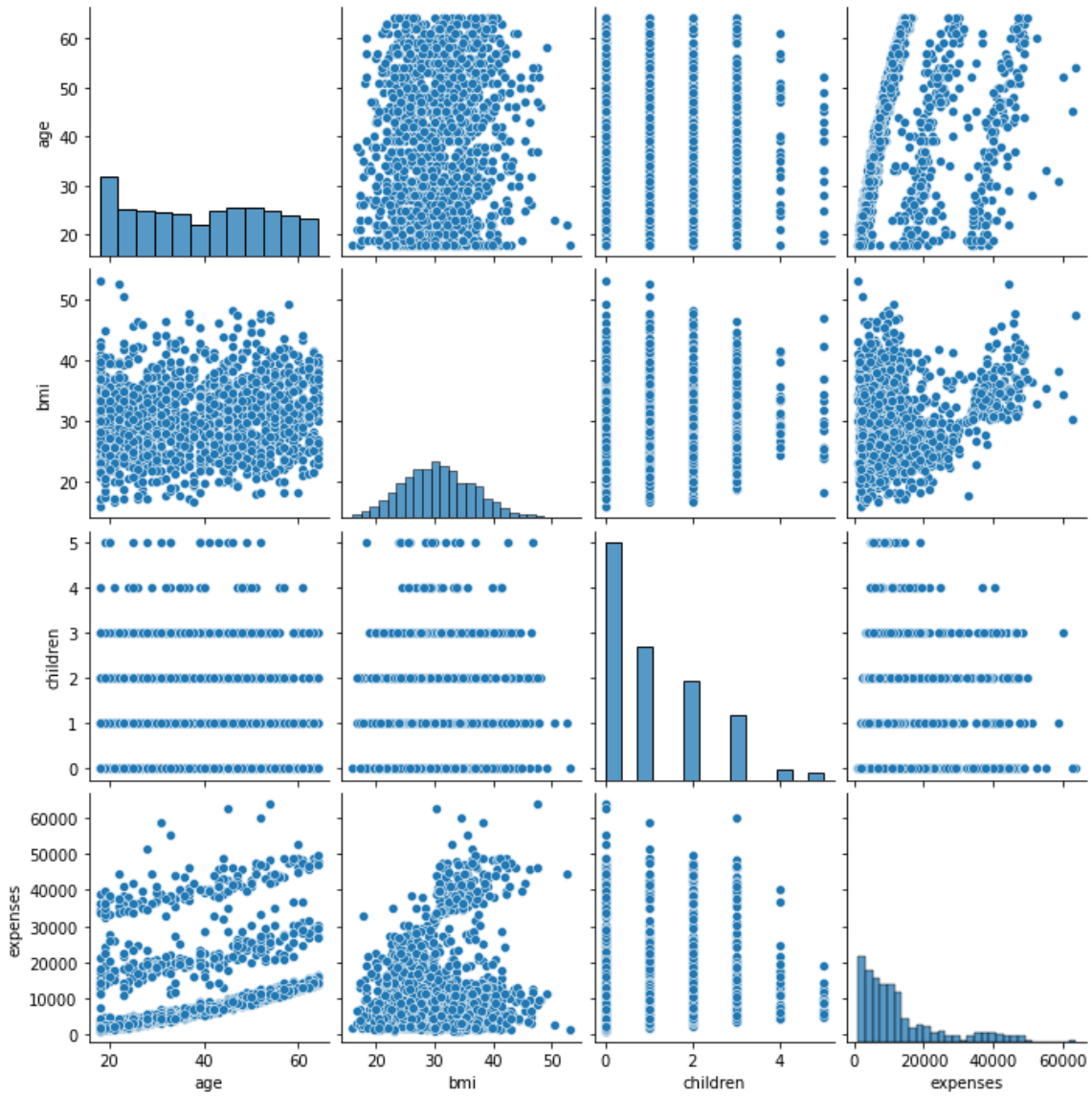


Img 5: Histogramas de las variables bmi, age, expenses y children

Cabe mencionar que la distribución del atributo bmi es gaussiana.

Comparación de Datos

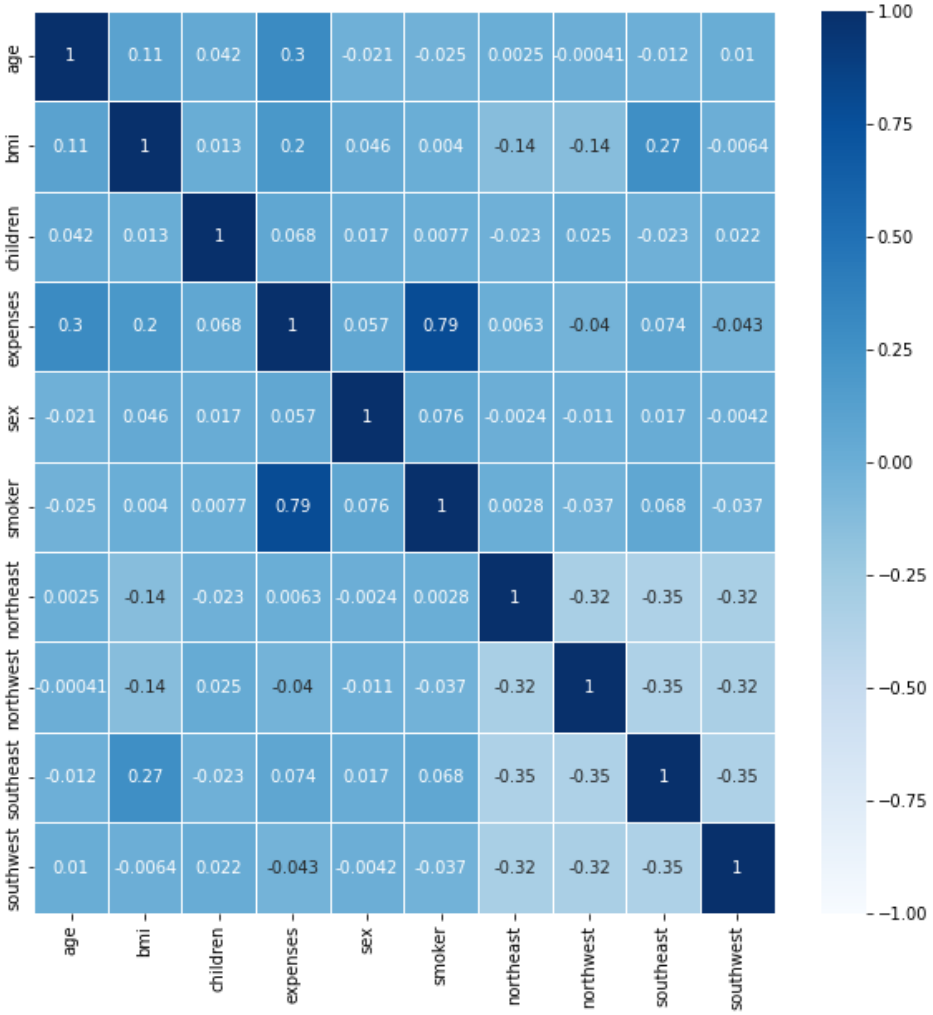
Después de haber representado todos los datos que tenemos de la base de datos decidimos representar las variables no categóricas mediante la función pairplot() de la librería seaborn, para que pudiésemos observar cómo se relacionaba cada una de las variables con las otras.



Img 6: Gráficas mostradas mediante la función pairplot() donde observamos cómo se relacionan las variables no categóricas

Tras representar todos los datos nos dispusimos a pasar las variables categóricas a valores que pudiesen ser representados como variables no categóricas. De esta manera creamos dos funciones. La primera de las dos hace que una variable categórica con una cantidad binaria de posibles aspectos se pase a una variable no categórica de valor1 → 1 y valor0 → 0. La segunda función la creamos con el objetivo de tratar el atributo región, ya que tiene diversos valores, no solo dos. Lo que hicimos entonces fue crear una nueva variable por cada valor que tenga, y añadimos las nuevas columnas al Data Frame. Así se crearon tantas nuevas columnas como valores diferentes tiene la variable categórica y se guardaron los datos comprobando si la región coincidía con la nueva variable creada, si coincidía se guardaba la tupla.

Tras tener todos los valores de una misma manera que pudiéramos tratar, quisimos hacer un mapa de calor para ver la relación entre todas las variables, y donde pudimos observar qué variables serían las más interesantes de trabajar.



Img 7: Mapa de calor donde se expresa la correlación entre todas las variables

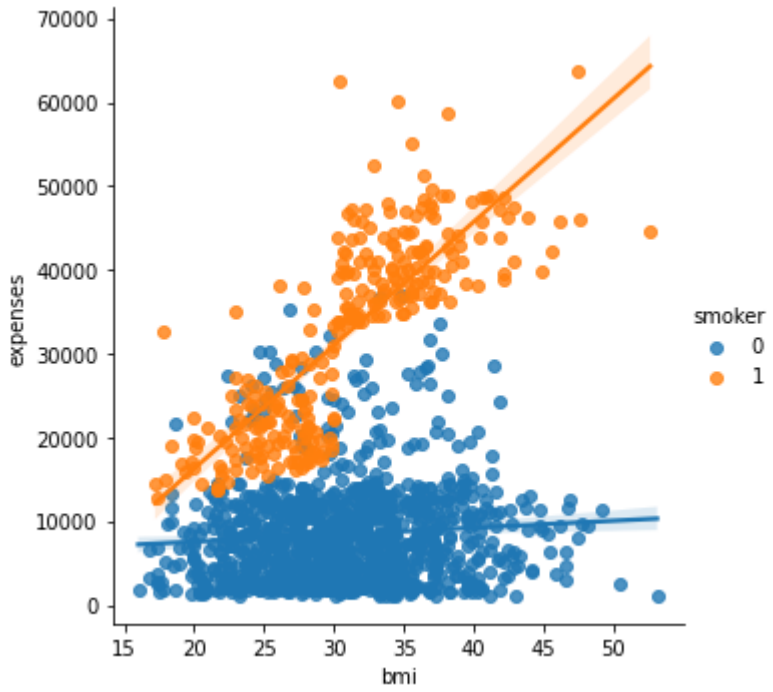
Gracias a la librería de seaborn conseguimos este mapa de calor (Img 7), con la función heatmap, que nos permite observar lo altamente relacionada que se halla "expenses" con "smoker" y que también comparte cierta correlación con "age" y "bmi".

Debido a que observamos la alta relación entre "expenses" y "smoker" decidimos indagar en los datos que representaban esa relación.

smoker	mean	median	count
0	8434.268449	7345.405	1064
1	32050.231971	34456.350	274

Img 8: Tabla de relación de datos entre "smoker" y "expenses" donde observamos que si una persona fuma tiene un gasto superior, de media

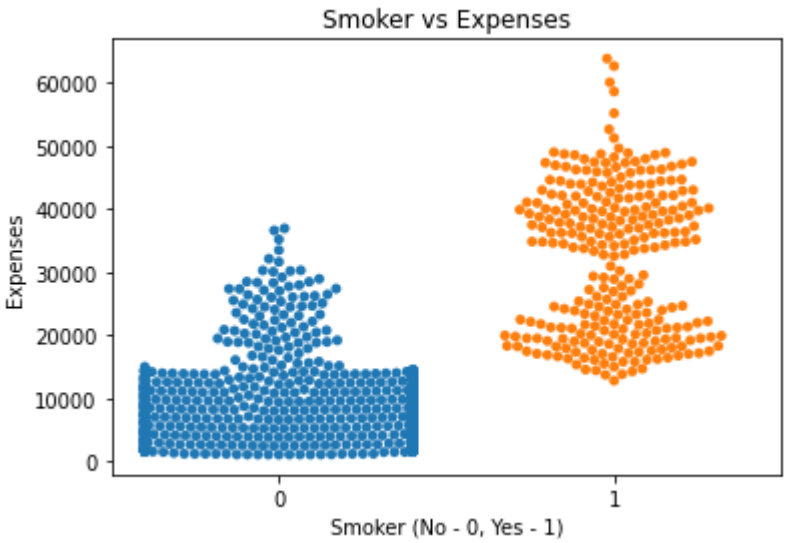
También quisimos expresar esta información en gráfica, pero si intentábamos hacer una gráfica con "expenses" y "smoker" nos quedaba una gráfica separada a dos bandas, según si fumaban qué gastos tiene cada persona. De esta manera decidimos observar la relación entre los "expenses" y "bmi", decidimos esto debido a que ambas eran variables no categóricas y a que compartían una cierta correlación. Teniendo esa gráfica en mente lo siguiente que hicimos fue un marcador para visualizar la gente que fumaba y la que no en esa relación, y lo que obtuvimos fue lo siguiente.



Img 9: Función lmplot() de la librería seaborn donde x = bmi, y = expenses y tintamos los puntos según si son fumadores o no.

Podemos ver cómo la gente que fuma, cuando mayor es el bmi, mayor es el gasto. En cambio cuando no es fumador, por mucho que aumente su bmi, no aumenta su gasto. También podemos ver las líneas de regresión que representan como aumentará el gasto dependiendo de si fuma o no.

Después de un tiempo, descubrimos la función swarmplot(), la cual nos permitió observar la relación entre "smoker" y "expenses" aún más.

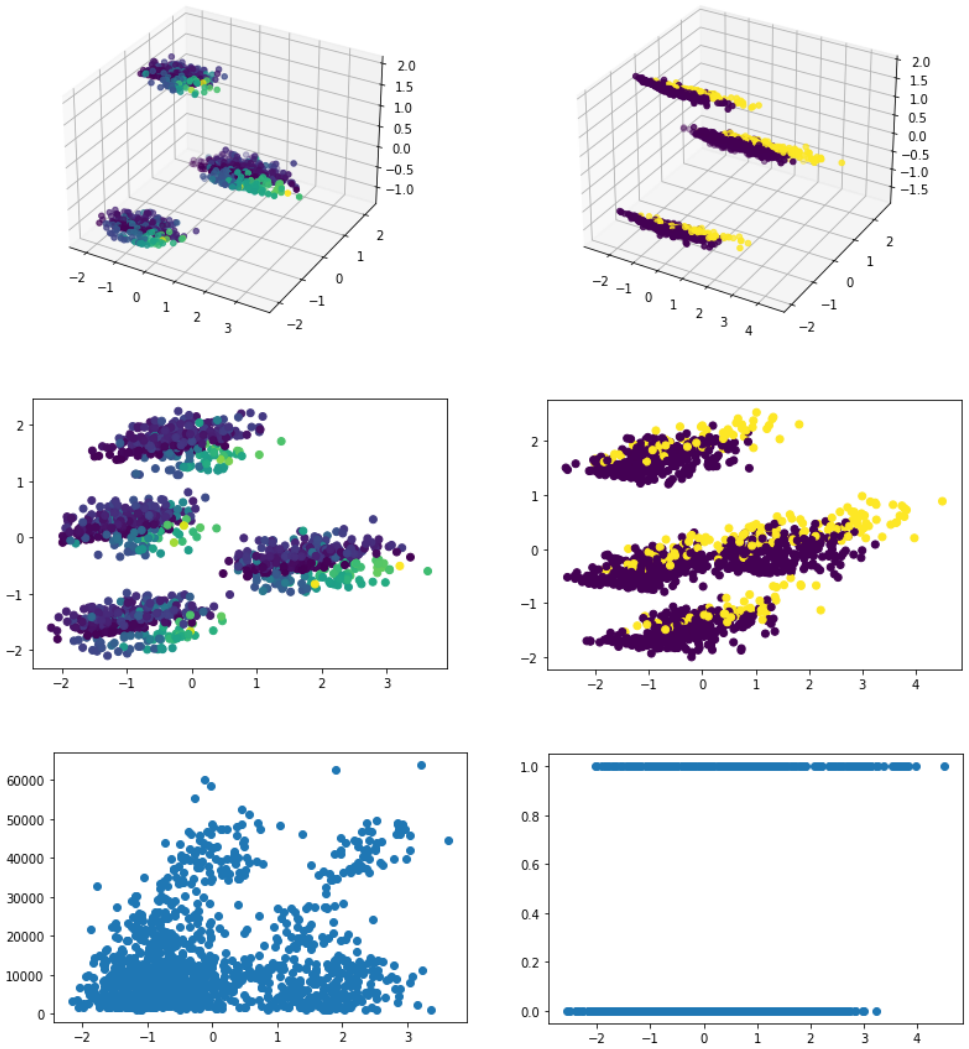


Img 10: Función swarmplot() de la librería seaborn donde x = smoker, y = expenses

En esta gráfica podemos ver la cantidad de gente fumadora, no fumadora y sus gastos. Se concentra en la parte inferior aquellos que no fuman y los que fuman, se centran más en la parte superior de la gráfica. Esto refuerza la hipótesis de que la mayoría de fumadores gastan más que los que no fuman.

Funciones para Observar Datos

Lo último que hacemos en este punto es crear tres funciones para representar los datos con la función scatter. La razón por la cual hemos hecho tres funciones es para poder representar los datos desde una hasta tres dimensiones.



Img 11: Gráficas con scatter() en 3D, 2D y 1D de "expenses" a la izquierda y "smoker" a la derecha

Tras visualizar todas las gráficas, decidimos deshacernos de la función que mostraba la función con una PCA de n_componentes = 1, debido a que creíamos que no nos aportaba una información suficiente para trabajar los datos. Aún así mantuvimos y usamos las de dos y tres dimensiones.

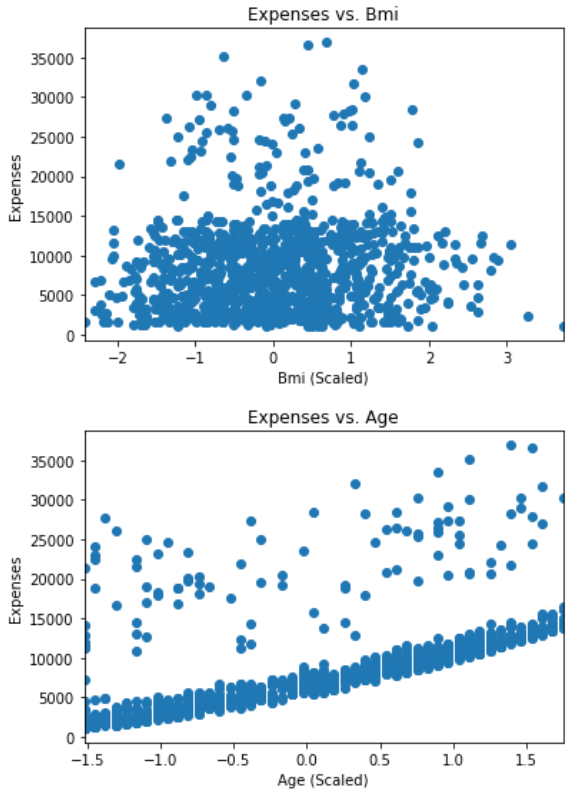
Al visualizar los datos con la dimensionalidad reducida nos damos cuenta que este dataset para un problema de clasificación iría muy bien, ya que a simple vista se pueden diferenciar los datos ploteados en la gráfica.

Apartado B

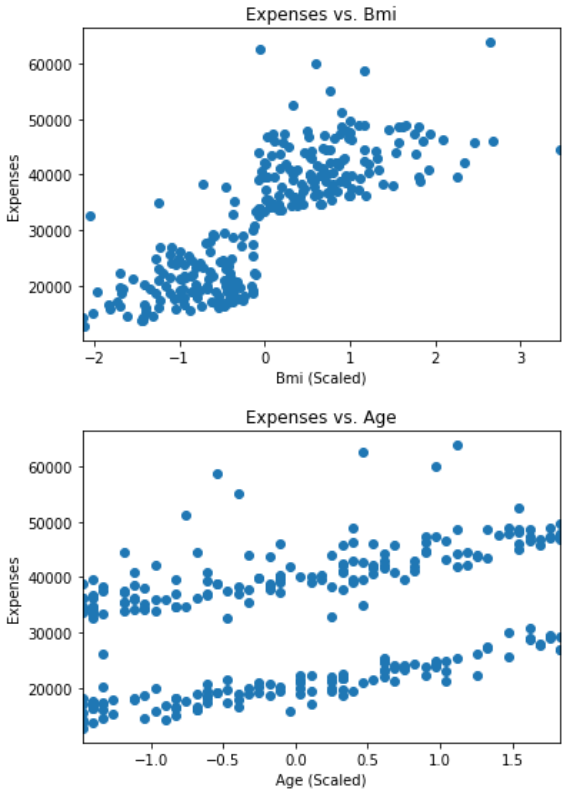
Introducción

En el apartado B trabajaremos las primeras regresiones, para ello, debemos decidir qué variable es la más importante para hacer una buena predicción. Calculamos el MSE y el error de nuestras variables y escogemos la variable cuyo R2 score sea el más cercano a 1. Una vez calculados, vemos que el atributo con el valor R2 más cercano a 1 es también el que tiene el MSE menor, ese atributo es el que dice si una persona fuma o no.

Dividiendo los datos en función de si la persona fuma o no podemos generar 2 regresiones lineales. Para ello creamos una función llamada split_smoker() que recibe los datos, los procesa y luego los divide en según si la variable smoker es 1 o 0. En las siguientes gráficas podemos observar las diferencias que existen si comparamos los gastos de una persona fumadora en función de su edad o su bmi y una persona no fumadora.



Img 12: Gráficas representativas de los no fumadores



Img 13: Gráficas representativas de los fumadores

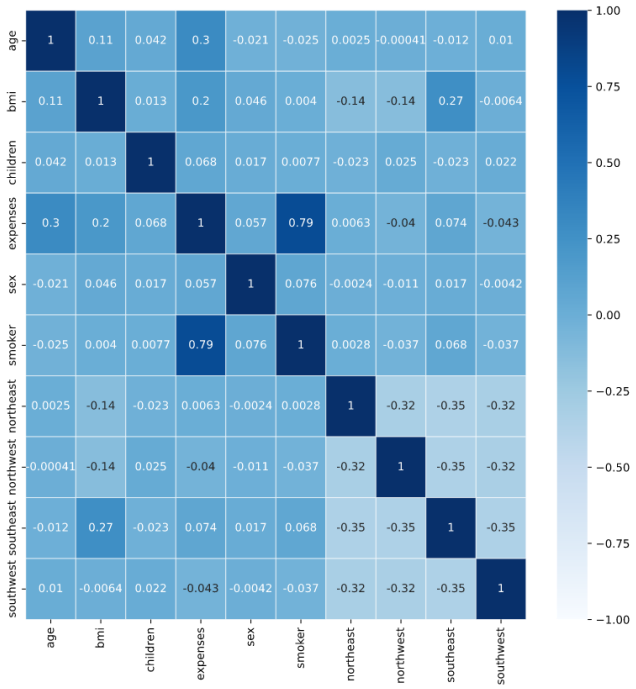
Como se puede apreciar, las dos gráficas son muy diferentes, por lo que podemos afirmar que crear una distinción entre las personas que fuman y las que no hace que nuestras predicciones sean más precisas.Con esta distinción, calculamos el MSE y el error R2, dando como resultado los valores siguientes:

Fuman	No fuman
Error en atribut 0 age: 127766199.340951 R2 score en atribut 0 age: 0.115129 Error en atribut 1 bmi: 38738628.362017 R2 score en atribut 1 bmi: 0.731708 Error en atribut 2 children: 147335927.330207 R2 score en atribut 2 children: -0.020405 Error en atribut 3 expenses: 140806063.150975 R2 score en atribut 3 expenses: 0.024819 Error en atribut 4 sex: 146136262.628218 R2 score en atribut 4 sex: -0.012096 Error en atribut 5 northeast: 142254028.970696 R2 score en atribut 5 northeast: 0.014791 Error en atribut 6 northwest: 136894729.701176 R2 score en atribut 6 northwest: 0.051908 Error en atribut 7 southeast: 144505447.141567 R2 score en atribut 7 southeast: -0.000802	Error en atribut 0 age: 19079260.800555 R2 score en atribut 0 age: 0.386789 Error en atribut 1 bmi: 31169319.068749 R2 score en atribut 1 bmi: -0.001789 Error en atribut 2 children: 31097856.290510 R2 score en atribut 2 children: 0.000508 Error en atribut 3 expenses: 31024355.096600 R2 score en atribut 3 expenses: 0.002871 Error en atribut 4 sex: 31158103.544536 R2 score en atribut 4 sex: -0.001428 Error en atribut 5 northeast: 31234956.955154 R2 score en atribut 5 northeast: -0.003898 Error en atribut 6 northwest: 31237457.328256 R2 score en atribut 6 northwest: -0.003979 Error en atribut 7 southeast: 31233675.455773 R2 score en atribut 7 southeast: -0.003857

Como podemos observar en la tabla anterior existen diferencias notables en algunos datos en función de la variable, por ejemplo, la variación que existe en el primer atributo no es demasiado significativa si la comparamos con la diferencia que existe con el segundo atributo, que pasa de tener un valor negativo a estar muy cerca del 1, lo que nos indica que la relación que existe entre el bmi de las personas y si son fumadoras o no. El resto de variables sufren cambios poco relevantes comparados con los atributos anteriores, lo que nos indica que existe una mayor relación entre estos atributos. Para saber exactamente hasta qué punto llega ésta relación, decidimos representarlo en un mapa de calor.

Correlación

Calculamos ahora la correlación que existe entre los atributos de nuestra base de datos una vez más, siguiendo de guía las preguntas que se nos formulen y volvemos a mostrar la misma gráfica que hemos calculado anteriormente:



Img 7: Mapa de calor donde se expresa la correlación entre todas las variables

Normalización

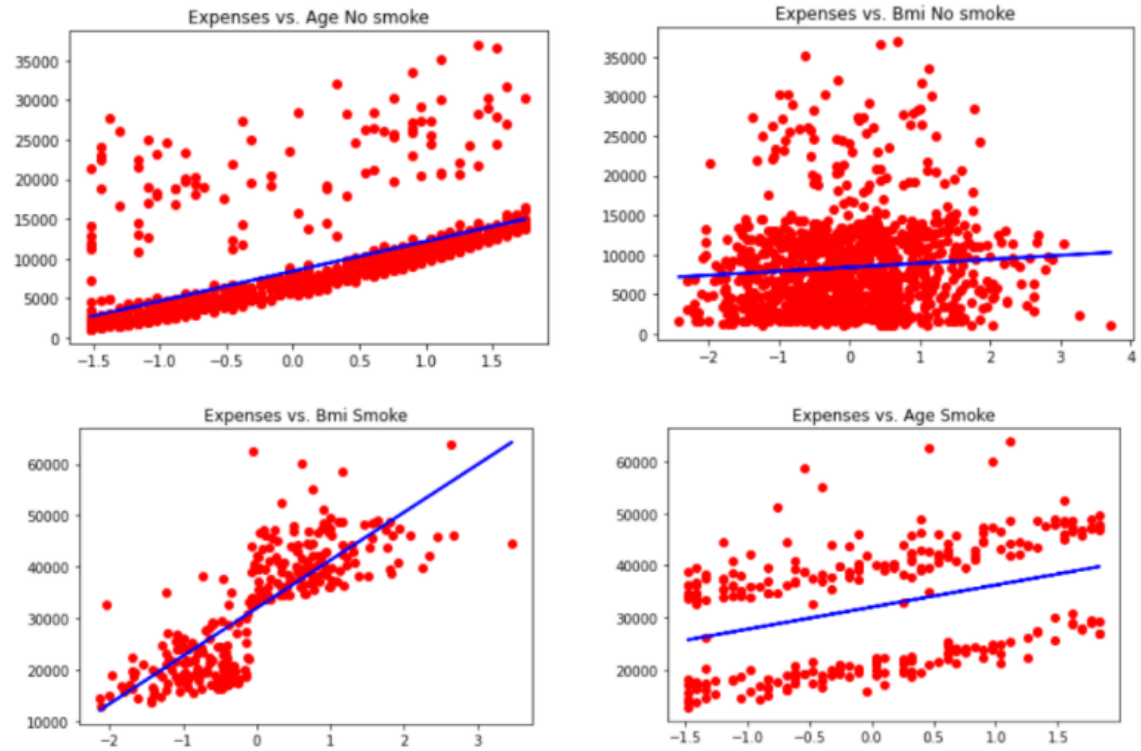
A continuación, comparamos el efecto que tiene normalizar los datos a la hora de calcular el MSE y el error. En nuestro caso, vemos que normalizar no tiene un efecto muy significativo en los valores finales.

	Normalizados	No normalizados
Fuman	Error en atribut 0 age: 139423610.749232 R2 score en atribut 0 age: 0.130965 Error en atribut 1 bmi: 39464116.443508 R2 score en atribut 1 bmi: 0.754018 Error en atribut 2 children: 168019816.634773 R2 score en atribut 2 children: -0.047277 Error en atribut 3 expenses: 164569965.522672 R2 score en atribut 3 expenses: -0.025774 Error en atribut 4 sex: 161081401.036499 R2 score en atribut 4 sex: -0.004029 Error en atribut 5 northeast: 165449570.442259 R2 score en atribut 5 northeast: -0.031256 Error en atribut 6 northwest: 156043216.305242 R2 score en atribut 6 northwest: 0.027374 Error en atribut 7 southeast: 166712410.491322 R2 score en atribut 7 southeast: -0.039128	Error en atribut 0 age: 132533687.792155 R2 score en atribut 0 age: 0.008680 Error en atribut 1 bmi: 55076980.440049 R2 score en atribut 1 bmi: 0.588038 Error en atribut 2 children: 137947547.975719 R2 score en atribut 2 children: -0.031814 Error en atribut 3 expenses: 141721364.266430 R2 score en atribut 3 expenses: -0.060041 Error en atribut 4 sex: 141732781.547119 R2 score en atribut 4 sex: -0.060127 Error en atribut 5 northeast: 146146306.677267 R2 score en atribut 5 northeast: -0.093139 Error en atribut 6 northwest: 138314088.391186 R2 score en atribut 6 northwest: -0.034556 Error en atribut 7 southeast: 142827445.320557 R2 score en atribut 7 southeast: -0.068314
No fuman	Error en atribut 0 age: 17297532.368592 R2 score en atribut 0 age: 0.362956 Error en atribut 1 bmi: 27114197.514390 R2 score en atribut 1 bmi: 0.001423 Error en atribut 2 children: 27606240.643686 R2 score en atribut 2 children: -0.016699 Error en atribut 3 expenses: 27735564.150629 R2 score en atribut 3 expenses: -0.021461 Error en atribut 4 sex: 27628602.015564 R2 score en atribut 4 sex: -0.017522 Error en atribut 5 northeast: 27446670.480679 R2 score en atribut 5 northeast: -0.010822 Error en atribut 6 northwest: 27582042.787077 R2 score en atribut 6 northwest: -0.015807 Error en atribut 7 southeast: 27476011.866666 R2 score en atribut 7 southeast: -0.011903	Error en atribut 0 age: 19761133.268150 R2 score en atribut 0 age: 0.309007 Error en atribut 1 bmi: 28806225.262765 R2 score en atribut 1 bmi: -0.007276 Error en atribut 2 children: 28069688.356532 R2 score en atribut 2 children: 0.018479 Error en atribut 3 expenses: 28491406.710602 R2 score en atribut 3 expenses: 0.003733 Error en atribut 4 sex: 28766962.281738 R2 score en atribut 4 sex: -0.005903 Error en atribut 5 northeast: 28867266.039021 R2 score en atribut 5 northeast: -0.009410 Error en atribut 6 northwest: 29568413.642810 R2 score en atribut 6 northwest: -0.033927 Error en atribut 7 southeast: 28622677.691249 R2 score en atribut 7 southeast: -0.000858

En este caso, se puede observar que existen pequeñas diferencias entre los datos normalizados y los no normalizados pero ningún cambio muy significativo. Lo que sí que se mantiene en ambos casos es la diferencia entre los datos más significativos ya mencionados en apartados anteriores, “bmi” y “age”.

Regresiones

Sabiendo esto realizamos las regresiones lineales de los atributos escogidos y normalizando los datos.
Podemos observar que gracias a separar el dataset en fumadores y no fumadores podemos generar diferentes líneas de regresión para un menor mse.



Img 14: Gráfica comparativa con los valores normalizados y los no normalizados

Filtros

Más adelante, nos preguntamos cómo mejoraría la regresión si eliminamos los datos que no contienen información. Primero determinamos un threshold, en este caso si fuma, luego volvemos a calcular el error

	Filtro	No filtro
Fuma	(851, 2) Error: 25739825.95956485 R2: 0.3778368234312818	(851, 8) Error: 17991709.36618042 R2: 0.43013825881265455
No fuma	(219, 2) Error: 32793953.119723886 R2: 0.8056667827608712	(219, 8) Error: 1256458.012452453 R2: 0.7348128222568906

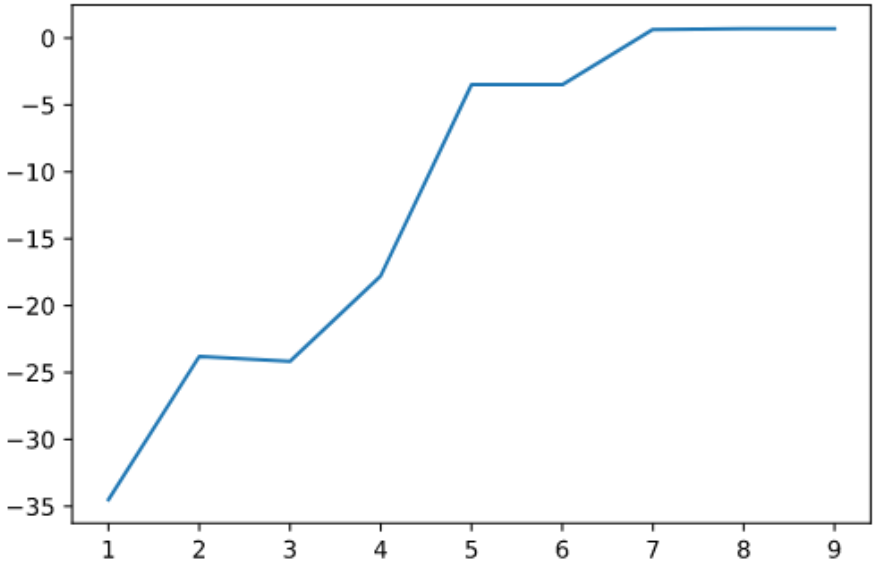
En este caso podemos ver como en el caso de los fumadores utilizar el filtro aumenta los errores y empeora un poco el R2 pero pasamos de 8 dimensiones a 2, cosa que reduce considerablemente el trabajo. En el caso de los no fumadores utilizar el filtro aumenta los errores en una gran cantidad, aunque mejora el R2 y, de nuevo, reduce las dimensiones a 2.

PCA

Para acabar este apartado, aplicamos un PCA para saber el número de componentes principales en nuestros datos. Para calcular la mejor dimensión, creamos una función llamada best_dimension_pca(x, y). Usando la variable “mle” en el número de componentes hacemos que la función nos devuelva los atributos más significativos. El porqué la variable “mle” escoge los atributos más significativos lo hemos encontrado en un “paper” [https://tminka.github.io/papers/pca/minka-pca.pdf]. Para ver la dimensionalidad que necesitamos para nuestra regresión vamos a buscar el error medio más pequeño que obtenemos por cada dimensión.

```
Original Dimension: 9
Dimension:  1  R2:  -34.49654268884269
Dimension:  2  R2:  -23.79883617355602
Dimension:  3  R2:  -24.16655776820443
Dimension:  4  R2:  -17.77236915097415
Dimension:  5  R2:  -3.480934585766419
Dimension:  6  R2:  -3.489292669486506
Dimension:  7  R2:  0.639300048446942
Dimension:  8  R2:  0.700907883112964
Dimension:  9  R2:  0.700907883112964
La mejor dimensión es (8, 0.700907883112964)
```

En esta tabla podemos ver cómo a partir de 8 dimensiones es cuando conseguimos un el mayor r2_score.



Img 15: Gràfica que representa la evolución de R2 al variar las dimensiones.

Error y valor de r2_score y el shape utilizando el algoritmo mle.

```
error:  41334303.42089423
R2:  0.7095668074840411
shape:  (1338, 8)
```

Podemos observar que el algoritmo de mle calcula el valor de R2 para cada posible dimensión y considera como mejor opción aquella que de un resultado más cercano a 1. También escoge el número de dimensionalidad que minimice el error medio, en este caso el algoritmo ha decidido que la mejor opción es escoger 8 dimensiones. Como podemos observar en el gráfico anterior, la diferencia entre 7, 8 y 9 dimensiones es muy pequeña y pasamos de tener valores negativos a valores muy cercanos a 1.

Apartado A

Introducción

En este apartado hemos realizado la implementación del descenso del gradiente. Con las fórmulas proporcionadas en el informe orientativo.

En nuestro algoritmo del descenso del gradiente hemos puesto los siguientes parámetros:

Mientras max_iters y error_diff(mse) < epsilon:

$$w_0 = w_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (f(x^i; w) - y^i) \cdot 1$$

$$w_j = w_j - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} (f(x^i; w) - y^i) \cdot x_j^i - \frac{\lambda}{m} w_j \right]$$

End mientras

- **Learning Rate (α):** Es el porcentaje de cambio con el que se actualizan los pesos en cada iteración, cada vez que es más grande el valor de alpha más rápido va a converger hacia una solución. Pero si utilizamos valores muy grandes, es posible que no saltemos mínimos locales muy buenos.
- **Regularizador (λ):** Valor por el cual evitamos tener un overfitting.
- **Max_iters:** Número máximo de iteraciones (actualizaciones de los pesos) que se realizará en la fase de entrenamiento.
- **Épsilon:** Diferencia numérica entre el mse actual y el actual -1 que indicará si el modelo ha convergido.

También hemos utilizado la clase sklearn.preprocessing.PolynomialFeatures para transformar los datos de entrada a polinomial, para tener modelos más complejos. El valor del grado del polinomio lo hemos llamado:

- **Degree:** Grado de la función polinomial. Que se utilizan para modificar los datos de entrenamiento y tener modelos polinomiales.

La implementación tiene los siguientes métodos públicos:

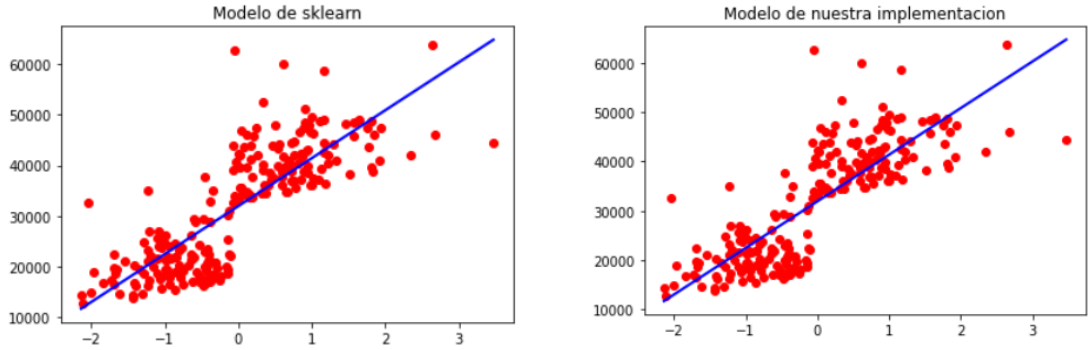
```
def __init__(self, X: np.array, y: np.array, lr=0.01, regulador=0.001) -> None:
        """
        Inicializa el modelo de regresión lineal,
        con los datos de entrenamiento,
        y los parámetros de aprendizaje.

        Args:
            X (np.array): Matriz de datos de entrenamiento.
            y (np.array): Columna objetivo de entrenamiento.
            lr (float, optional): Tasa de aprendizaje. Defaults to 0.01.
            regulador (float, optional): Regularizador. Defaults to 0.001.
        """

def predict(self, X: np.array) -> np.array:
        """
        Predice el valor de la columna objetivo

        Args:
            X (np.array): Matriz de datos de prueba.

        Returns:
            np.array: Valor de la columna objetivo.
        """

def mse_lambda(self) -> float:
        """
        Calcula el error cuadrático medio. Con un regularizador.

        Returns:
            float: Error cuadrático medio.
        """

 def train(self, max_iter=10000, epsilon=0.1) -> None:
        """
        Entrena el modelo de regresión lineal, utilizando el algoritmo de
gradiente descendente.
        Utiliza un criterio de parada para el aprendizaje.
        Tiene implementeado un regularizador.

        Args:
            max_iter (int, optional): Numero maximo de iteraciones. Defaults to
10000.
            epsilon (float, optional): Umbral de parada. Defaults to 0.1.
         """
```

Comparación librería sklearn

Antes de realizar pruebas sobre los parámetros de nuestro algoritmo, vamos a compararlo con el regresor lineal de la librería de sklearn, para ellos vamos a realizar 3 pruebas diferentes, con un modelo lineal, con un modelo polinomial, y la visión con el Coeficiente prismático.

Modelo lineal

- El valor de los pesos de nuestra implementación son: [0, 9457.23 , -664.84]
- El valor de los pesos de la implementación de sklearn son: [0, 9460.09, -666.97]
- El valor del bias de nuestra implementación es de: 32621.17
- El valor del bias de la implementación de sklearn es: 32625.23
- El r2 score de nuestra implementación es: 0.66152
- El r2 score de la implementación de sklearn es: 0.66165

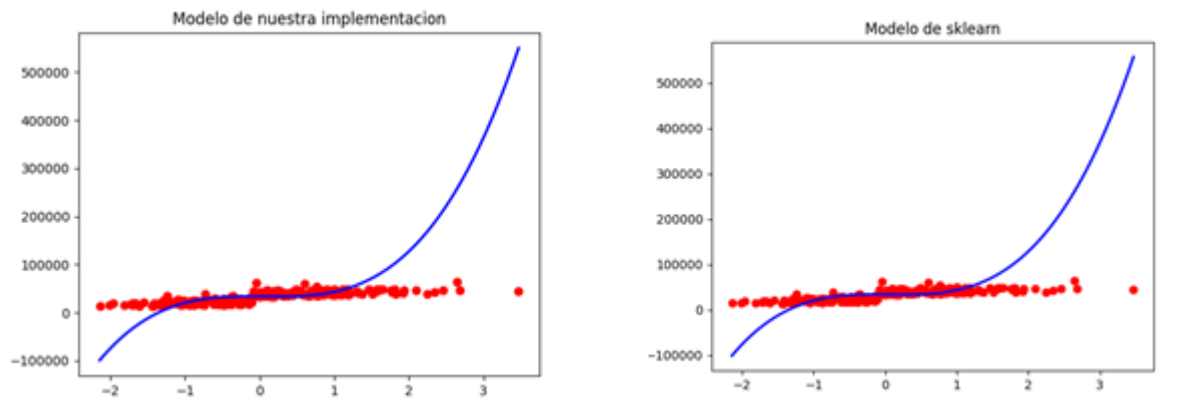


Img 16: Comparativa con sklearn de un modelo lineal.

Vemos que nuestra implementación responde bien ante modelos lineales. Ya que los coeficientes y el bias, son prácticamente iguales que los de sklearn.

Modelo polinomial

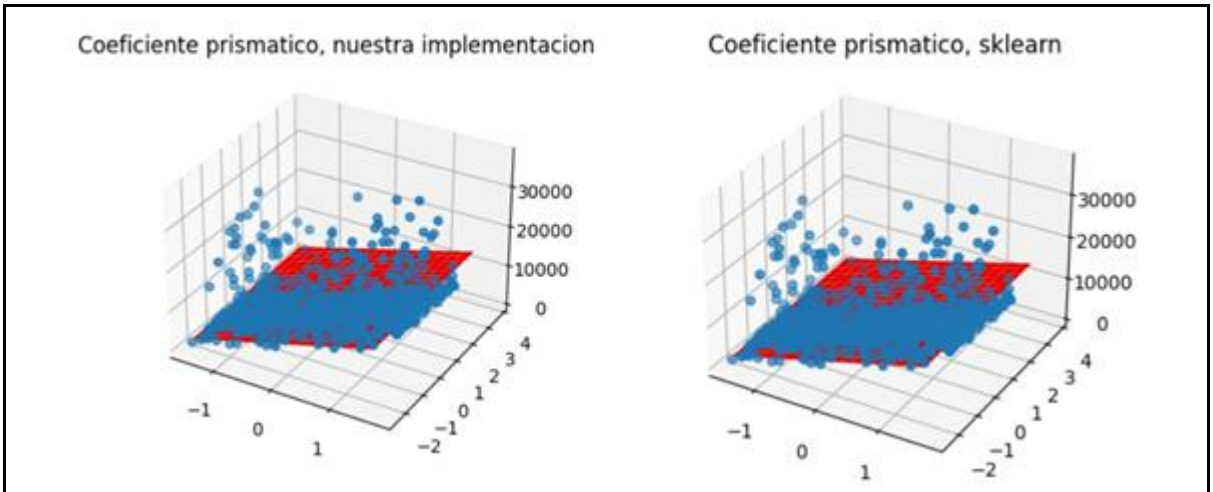
- El valor de los pesos de nuestra implementación es: [0, 13009.56, -1749.31 ,-1551.2797673, 382.90]
- El valor de los pesos de la implementación de sklearn es: [0, 13224.78 , -1848.23 -1629.12, 409.68]
- El valor del bias de nuestra implementación es de: 33149.82
- El valor del bias de la implementación de sklearn es: 33199.76
- El r2 score de nuestra implementación es: 0.7333
- El r2 score de la implementación de sklearn es: 0.7363



Img 17: Comparativa con sklearn de un modelo polinomial.

Nuestra implementación también responde bien ante modelos polinomiales.

Coeficiente prismático



Img 18: Visión del Coeficiente prismático.

Al visualizar estas gráficas y los valores de los pesos de ambas implementaciones, podemos asegurar que nuestra implementación del descenso del gradiente es similar a la de sklearn.

Obtención de los mejores parámetros para nuestro modelo

Una vez sabido los parámetros vamos a probar diferentes valores en los parámetros para encontrar cual es la mejor combinación de ellos, los rangos escogidos para esta primera prueba han sido:

- **Learning Rate (α):** [0.0001, 0.001, 0.01, 0.1, 1]
- **Regularizador (λ):** [0.0001, 0.001, 0.01, 0.1, 1]
- **Epsilon:** [0.001]
- **Degree**: [1, 2, 3, 4]

Después de probar todos estos parámetros (con los datos de prueba) hemos obtenido que la mejor combinación de parámetros del modelo con el dataset de fumadores es:

- **Learning Rate (α):** 0.01
- **Regularizador (λ):** 0.0001
- **Epsilon:** 0.001
- **Degree**: 4

Y con el dataset de no fumadores es:

- **Learning Rate (α):** 0.01
- **Regularizador (λ):** 0.0001
- **Epsilon:** 0.001
- **Degree**: 2

Donde el R2 score ha sido de 0.76 para el dataset de los fumadores. Y un R2 score del 0.41 para el dataset de los no fumadores.

Viendo estos resultados hemos visto que, para el dataset de los fumadores, puede ser que al subir el grado tenga un mejor resultado del R2. Probaremos a buscar otra vez los valores de los parámetros para este dataset.

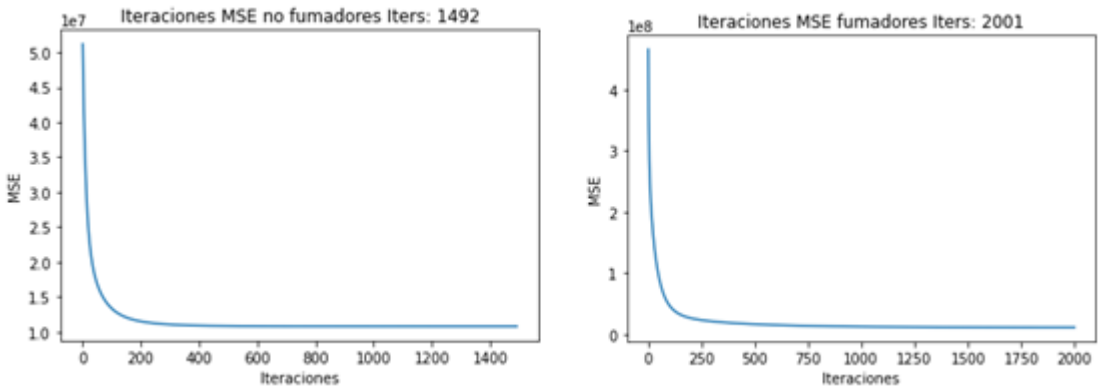
Los parámetros nuevos para probar son:

- **Learning Rate (α):** [0.01]
- **Regularizador (λ):** [0.0001, 0.001, 0.01, 0.1, 1]
- **Epsilon:** [0.001]
- **Degree**: [1, 2, 3, 4]

Vemos que los valores que nos salen son exactamente los mismos. Por lo que los parámetros finales para el dataset de fumadores son los propuestos anteriormente.

Observación: Realizando las pruebas hemos visto que el valor del **Regularizador (λ),** afecta muy poco en el entrenamiento del modelo.

Visualización del entrenamiento, el número de iteraciones que realiza el entrenamiento con los parámetros escogidos y el mse, de los dos datasets.



Img 19:

Al utilizar un epsilon tan pequeño (teniendo en cuenta que cogemos el mse como valor para el cálculo de la convergencia) en el entrenamiento de los fumadores no llega a pararse prematuramente.

Comparativa cualitativa

En esta comparativa vemos como mejora el modelo al utilizar uno lineal con uno polinomial:

Los parámetros que hemos escogido para cada prueba han sido:

- **Learning Rate (α):** 0.01
- **Regularizador (λ):** 0.0001
- **Epsilon:** 0.001
- **Degree**: [Dependiendo del modelo]

Media del error medio de todas las pruebas de un modelo lineal: 16291982.80
Media del error medio de todas las pruebas de un modelo polinomial grado 2: 13685779.53
Media del error medio de todas las pruebas de un modelo polinomial grado 3: 12667446.98

Podemos observar que mientras mayor es el grado del polinomio más se reduce el error medio. Esto se debe a que el modelo es capaz de adaptarse a datos que no siguen una distribución lineal (cosa que es muy común cuando estamos hablando de datasets con más de 2 columnas).

Conclusión

En base a la práctica.

El dataset que nos ha tocado ha tenido muchos valores categóricos y binarios, esto nos ha dificultado a la hora de realizar un regresion lineal. Por ese motivo hemos acabado dividiendo el dataset en dos, la columna por la que hemos separado ha sido “smoker”, siendo esta la que tenía una correlación más alta con la columna objetivo “expenses”. El motivo de escoger la columna “expenses” como columna objetivo se debe a que era la columna no categórica con un rango de valores más dispar.

Esta estrategia de separar el dataset nos ha funcionado bastante bien, ya teniendo un dataset que está más pensado para realizar una clasificación, hemos podido realizar diferentes regresiones lineales para atributos distintos.

Otra cosa a mencionar es que en nuestro caso el escalado de los valores no han tenido mucho impacto, ya que la mayoría de estos valores proceden de una columna categórica.

También hemos conseguido implementar correctamente el descenso del gradiente y utilizar modelos polinomiales para realizar regresiones. En este apartado hemos visto como afecta los parámetros del descenso del gradiente y hemos observado que el regularizador lambda en valores de 1-e4 y 1e4 no tiene suficiente impacto al menos valores polinomiales bajos. Pero donde sí hemos visto una diferencia notable ha sido con el learning rate (alpha). Siendo 1-e4 y 1-e1 el rango de valores que mejor nos ha funcionado.

A nivel personal.

Tras haber seguido el Jupyter Notebook que se nos daba y habiendo ido respondiendo las preguntas que se nos planteaban, hemos conseguido poco a poco reforzar la materia que se nos había presentado en clase como vernos enfrentados a problemas que no esperábamos que pudieran suceder. Como por ejemplo entender el motivo de normalizar los datos, visualizar la matriz de correlación.

También mencionar el hecho de que gracias a visualizar los datos en el mapa de calor (correlaciones) hemos podido comprobar que existen relaciones entre datos muy curiosas y divertidas que no se podrían haber visto a simple vista, por ejemplo, que el atributo “bmi” tiene correlación solo con las personas que viven en el sureste.

El hecho de implementar de cero el descenso del gradiente, nos ha hecho entender mejor cómo funciona.

Respecto a la complejidad de la práctica la hemos visto aceptable, aunque hemos sufrido un poco por el tipo de dataset que nos ha tocado.

Webgrafía

- https://www.kaggle.com/noordeen/insurance-premium-prediction
- https://en.wikipedia.org/wiki/Gradient_descent
- https://tminka.github.io/papers/pca/minka-pca.pdf
- https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/