Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI-5438 Inteligencia Artificial II
Trimestre Septiembre - Diciembre 2023

Proyecto 1

Estudiantes:

Figueira, Adelina

Bandez, Jesús

Carnets:

15-10484

17-10046

Profesor

Carlos Infante

3 de noviembre de 2023

Detalles de la implementación

Se realizó la implementación del algoritmo de descenso del gradiente de Russell y Norvig. Como el algoritmo es iterativo y se trabaja con una gran cantidad de ciclos, se puede modificar para ser escrito con matrices. Además, se puede aprovechar la optimización de las operaciones con matrices haciendo uso de la librería numpy de Python.
A continuación se presenta el fragmento de código basado en el algoritmo del descenso del gradiente.

- X es una matriz de $N * M$ que contiene los datos
- W es un vector $M * 1$ en el que se obtendrán los valores finales de los coeficientes de la función lineal
- P es un vector $N * 1$ con las predicciones
- MSE es un vector con los errores de la iteración
- Xt es la matriz traspuesta de X

```
for it in range(iteraciones):
    P = np.matmul(X,W)
    Xt = X.transpose()
    MSE = np.matmul(Xt,(P-Y))*(2/m)
    error_iteracion.append(max(abs(MSE)))
    W = W - (MSE*alpha)
    if max(abs(MSE))<epsilon:
        break
```

$$X = \begin{pmatrix} 1 & X_{12} & X_{13} \\ ... & ... & ... \\ 1 & X_{i2}... & X_{i3} \end{pmatrix} \tag{1}$$

$$W = \begin{pmatrix} W0 \\ W1 \\ W2 \end{pmatrix} \tag{2}$$

$$Y = \begin{pmatrix} Y_0 \\ ... \\ Y_i \end{pmatrix} \tag{3}$$

$$P = X * W = \begin{pmatrix} 1 & X_{12} & X_{13} \\ ... & ... & ... \\ 1 & X_{i2}... & X_{i3} \end{pmatrix} * \begin{pmatrix} W0 \\ W1 \\ W2 \end{pmatrix} = \begin{pmatrix} W0 + X_{12} * W1 + X_{13} * W2 \\ ... \\ W0 + X_{i2} * W1 + X_{i3} * W2 \end{pmatrix} \tag{4}$$

$$P - Y = \begin{pmatrix} W0 + X_{12} * W1 + X_{13} * W2 \\ ... \\ W0 + X_{i2} * W1 + X_{i3} * W2 \end{pmatrix} - \begin{pmatrix} Y_0 \\ ... \\ Y_i \end{pmatrix} = \begin{pmatrix} W0 + X_{12} * W1 + X_{13} * W2 - Y_0 \\ ... \\ W0 + X_{i2} * W1 + X_{i3} * W2 - Y_i \end{pmatrix} \tag{5}$$

Luego con $X * W$ se obtiene la función definida por Russell y Norvig como $hw(X_j)$ y $X * W - Y$ nos permite obtener la sumatoria

$$\sum_j (hw(x_j) - y_j) \tag{6}$$

Al calcular $X^t * (P - Y)$ conseguimos el gradiente.

$$X^t * (P - Y) = \begin{pmatrix} 1 & ... & 1 \\ X_{12} & ... & ... \\ X_{13} & X_{2i}... & X_{i3} \end{pmatrix} * \begin{pmatrix} W0 + X_{12} * W1 + X_{13} * W2 - Y_0 \\ ... \\ W0 + X_{i2} * W1 + X_{i3} * W2 - Y_i \end{pmatrix} \tag{7}$$

$$X^t * (P - Y) = \begin{pmatrix} W0 + X_{12} * W1 + X_{13} * W2 - Y_0 + ... + W0 + X_{i2} * W1 + X_{i3} * W2 - Y_i \\ ... \\ X_{i3} * (W0 + X_{12} * W1 + X_{13} * W2 - Y_0) + ... + X_{i3} * (W0 + X_{i2} * W1 + X_{i3} * W2 - Y_i) \end{pmatrix}$$

$X^t * (P - Y) * \frac{2}{M}$ es equivalente a:

$$\frac{2}{M} * \sum_j (hw(x_j) - y_j) * x_{j,i} \tag{8}$$

Finalmente solo hace falta actualizar el vector de pesos en cada iteración de igual manera que en la versión iterativa:

$$W_i = W_i - \alpha * \frac{2}{M} * \sum_j (hw(x_j) - y_j) * x_{j,i} \tag{9}$$

Por lo que la matriz W se actualiza como:

$$W = W - \alpha * (X^t * (P - Y) * \frac{2}{M}) \tag{10}$$

Además se fijó que el algoritmo se detiene si el valor de tolerancia epsilon es mayor a el máximo de los errores en valor absoluto de la iteración que se realiza.

Validación de la implementación

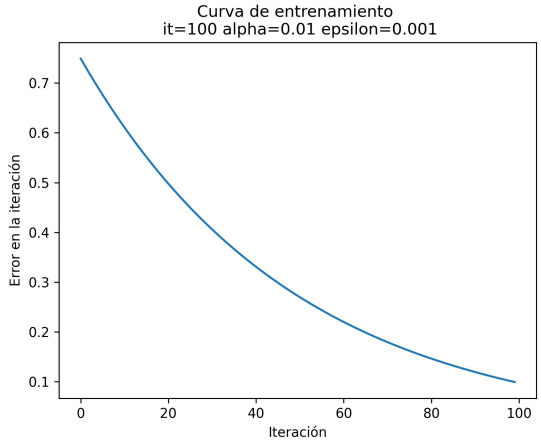
Se usaron pruebas con 1000 puntos aleatorios. En el archivo validation.py se generan 3 conjuntos distintos de puntos aleatorios para diferentes valores de los pesos W. Se genera la matriz X y el vector Y utilizando la función lineal:

$$f(x_1, x_2) = w_1 * x_1 + w_2 * x_2 + w_0 \tag{11}$$

Luego, se aplica el algoritmo de descenso de gradiente sobre estos datos para obtener una aproximación a los coeficientes de la función (11). Para medir la precisión del modelo se usa el coeficiente de determinación R^2.

Con 100 iteraciones, alpha=0.01 y epsilon=0.001

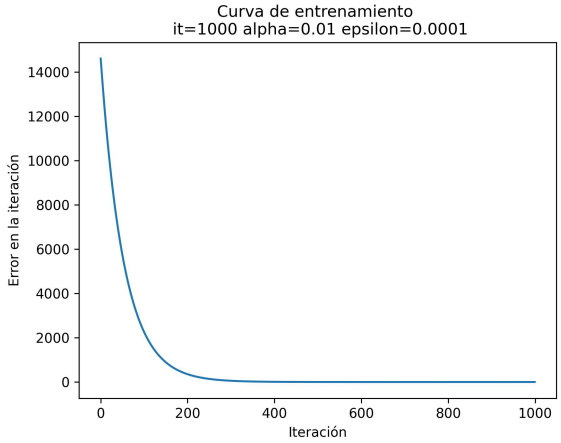
$$\begin{aligned} w_0 &= 0{,}6294111443016626 \\ w_1 &= 0{,}8049650781942079 \\ w_2 &= 0{,}8577551469796346 \end{aligned} \tag{12}$$



Se obtiene un coeficiente de determinación 0.998035895877083

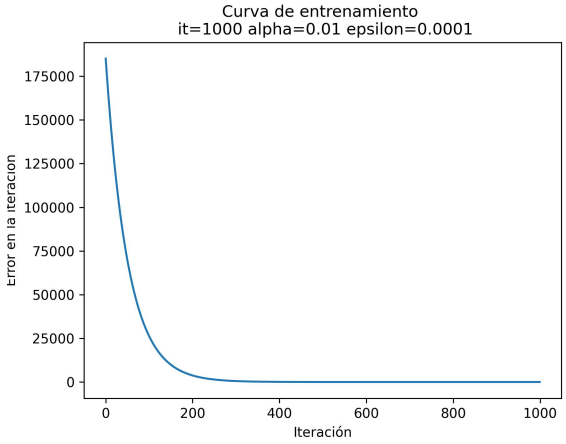
Con 1000 iteraciones, alpha=0.01 y epsilon=0.0001

$$\begin{aligned} w_0 &= 7890 \\ w_1 &= 1456 \\ w_2 &= 6786 \end{aligned} \tag{13}$$



Se obtiene un coeficiente de determinación 0.9999999999999999

$$\begin{aligned} w_0 &= 14578 \\ w_1 &= 58962 \\ w_2 &= 95653 \end{aligned} \tag{14}$$



Se obtiene un coeficiente de determinación 1

Es posible observar que como se obtienen valores cercanos al 1 para el coeficiente de determinación, podemos concluir que el modelo de pruebas obtenido es preciso. Con el gráfico

de curva de entrenamiento podemos notar que el error en la iteración se acerca a 0 a medida que aumenta el número de iteraciones. Por tanto, la implementación del algoritmo de Russel y Norvig es correcta.

Entrenamiento del modelo

Las columnas elegidas para realizar el modelo son las siguientes: “Make”, el fabricante del vehiculo, es bien sabido que existen marcas más cara que otras; “Year”, el año del vehículo, es de esperar que mientras más actual más caro el auto; “Kilometer”, el kilometraje actual; “Fuel Type”, tipo de combustible usado; “Transmission”, transmisión manual o automática; “Owner”, numero de dueños que ha tenido el vehículo; “Engine”, la capacidad de combustible y aire que fluye a través de los cilindros del motor, medido en cc; “Seating Capacity”, cantidad de asientos; y “Fuel Tank Capacity”, capacidad del tanque de combustible del vehiculo. Por supuesto, estas son comparadas con la columna “Price” para realizar el entrenamiento.

Limpieza de datos

Manejo de datos faltantes

Se utilizó el enfoque de eliminar los datos faltantes. Inicialmente el dataset cuenta con 2056 filas. Al eliminar los registros con al menos un dato “null” o “NaN” quedan 1937 filas. El total de filas eliminadas es de 119. Esto constituye el 5,78 % del dataset. Como la cantidad de datos removidos es relativamente baja, es viable eliminarlos.

Formato de columnas

Todas las columnas usadas del dataset cuentan con el formato necesario, a excepción de “Engine”. Bastó con extraer el número de cc de la columna para obtener el formato correcto.

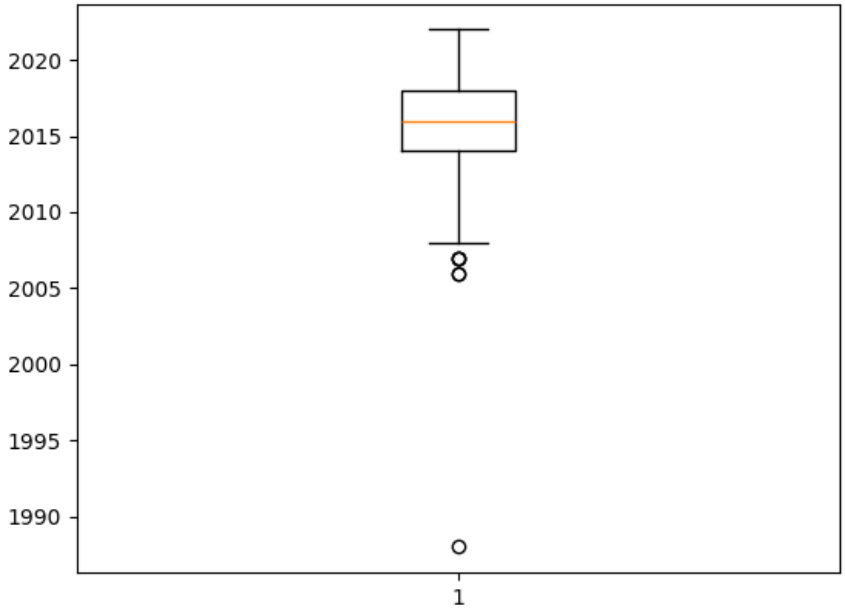
Tratamiento de datos categóricos

El manejo de valores categóricos se realiza sobre las columnas “Make”, “Fuel Type”, “Transmission” y “Owner”, debido a que son valores asignados a una clasificación. Cada una de estas columnas es reemplazada por n columnas binarias mutuamente exclusivas, donde n representa la cantidad de categorías que pertenecen a la columna. Por convención, las nuevas columnas creadas tienen el nombre “columna-categoría”. Donde “columna” corresponde al nombre de la columna reemplazada y “categoría” al nombre de la categoría.

Tratamiento de datos numéricos

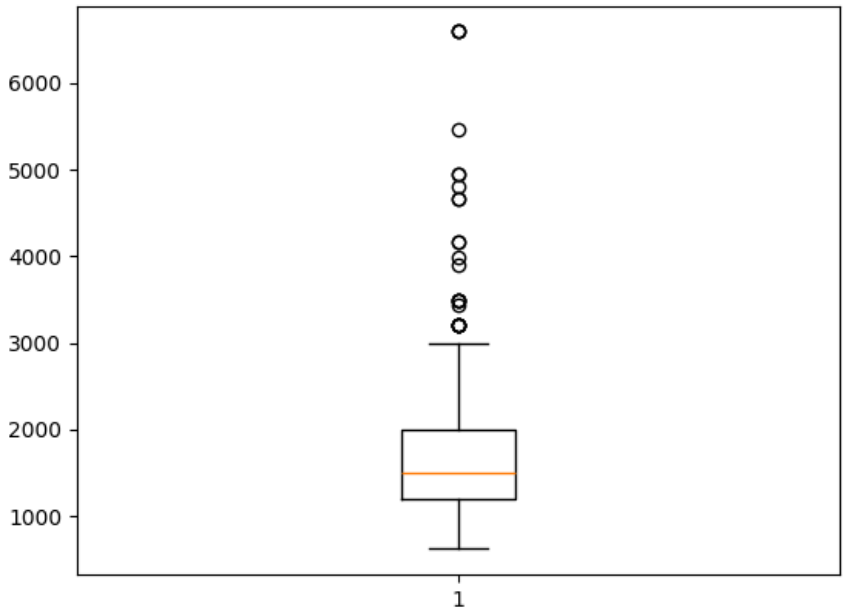
Para aumentar el rendimiento y la estabilidad del entrenamiento, se normalizaron las columnas “Year”, “Kilometer”, “Engine”, “Seating Capacity” y “Fuel Tank Capacity”. También, se realizaron diagramas de caja para comprobar la distribución de los datos. En estos gráficos, se encontraron que la mayoría de las columnas cuentan con datos atípicos. Estos datos podrían afectar la precisión del modelo debido a no ser comunes. A continuación se muestran los gráficos obtenidos:

Para la columna “Year”, se obtuvo el diagrama:



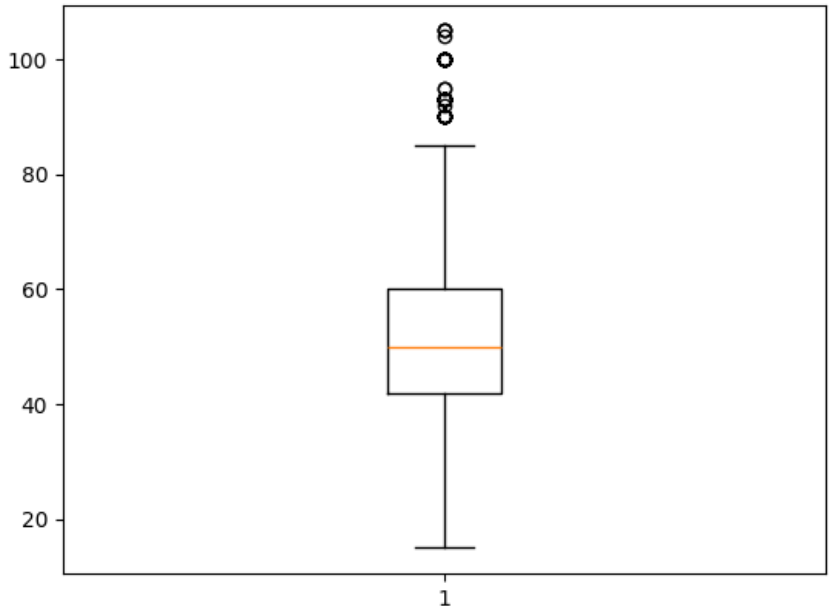
Todos los vehículos con año menor a 2007 se consideran atípicos.

Para “Engine”, se obtuvo:



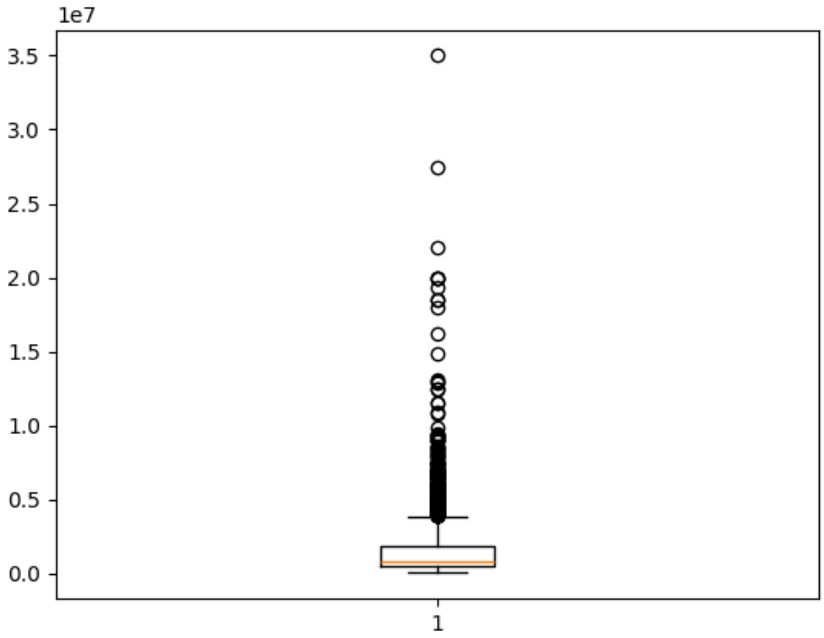
Por ello, todos los registros con engine mayor a 3000 cc son atípicos.

El “Fuel Tank Capacity” dibuja el siguiente diagrama:

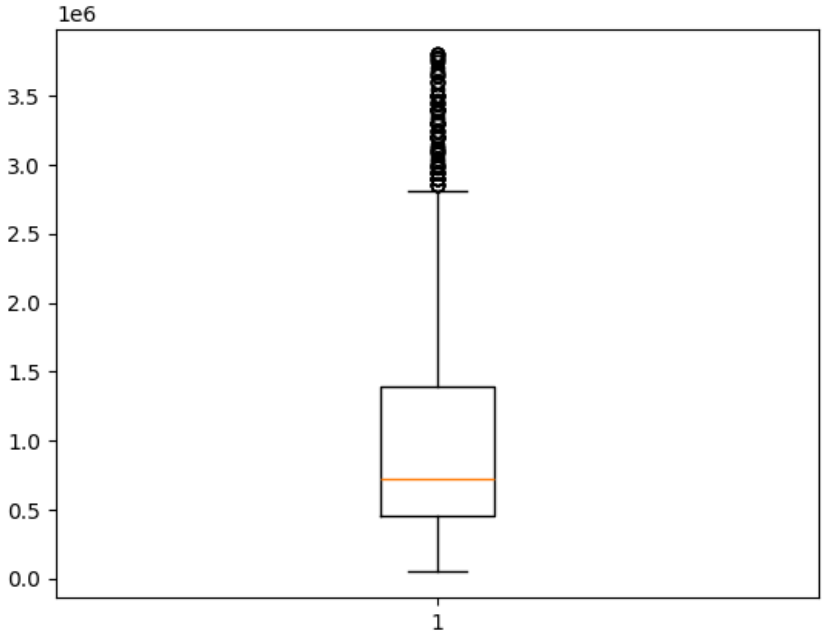


Se puede decir que las filas con fuel tank capacity mayor a 85 son atípicas.

Finalmente, la columna “Price” dibuja la siguiente gráfica:



Puede verse que esta gráfica tiene más datos atípicos. Además, estos se encuentran más alejados de la "caja". Por tanto, se eliminan estos datos y se vuelve a realizar el diagrama. Esto dá como resultado el siguiente gráfico:



Aunque se redujo considerablemente el intervalo de precio, siguen existiendo datos atípicos

en el diagrama. Estos se consideran cono todos aquellos cuyo precio exceda los 2.200.000.

Este análisis de los datos atípicos se realizó con el objetivo de filtrar al modelo de datos muy poco comunes. Si se eliminan estos datos, se podría obtener un modelo que sea capaz de predecir con mayor precisión. Sin embargo, es posible que el modelo que se obtenga tenga menor capacidad de generalizar sobre estos datos poco comunes. Por ello, se realizó el entrenamiento dos veces: en uno se eliminan los datos atípicos y en otro se conservan.

Proceso de entrenamiento

Los datos fueron separados en un 80 % y 20 % utilizando la librería scikit-learn de Python.

```
X_train, X_test, y_train, y_test = train_test_split(X, Y, train_size=0.8)
```

En donde X_{train} es la matriz que contiene un 80 % de los datos preprocesados a entrenar y X_{test} es la matriz que contiene un 20 % de los valores del archivo con los datos preprocesados.

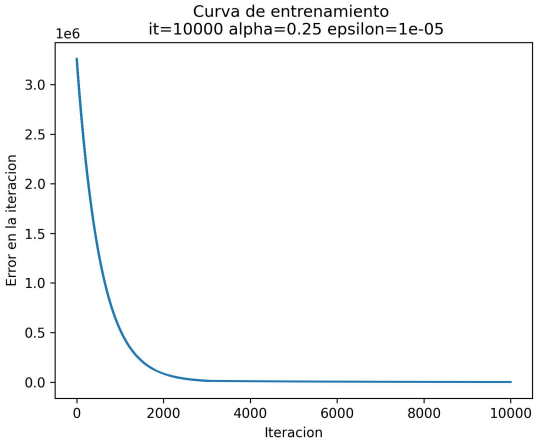
Una vez están separados los datos, se llama al método *gradient_descent* que contiene la implementación del algoritmo del gradiente y se aplica sobre los valores obtenidos de X_{train} y y_{train}. Se utilizan diferentes valores para los parámetros de las "iteraciones", "alpha" y "epsilon". Vea la tabla en la siguiente página:

Resultados obtenidos en el proceso de entrenamiento

Datos atípicos	Iteraciones	alpha	epsilon	Coeficiente de determinación
SI	1000000	0.01	0.001	0.7111669334178543
SI	10000000	0.01	0.1	0.7111669196655451
SI	100000	0.1	0.1	0.7111669334178543
SI	1000000	0.25	0.00001	0.7588030337240192
SI	2000000	0.25	0.00000000001	0.7588030337240192
SI	10000	0.25	0.00001	0.7594419270970644
NO	100000	0.1	0.001	0.813281308324263
NO	10000	0.1	0.001	0.813370021277846
NO	1000000	0.1	0.001	0.813281308324263
NO	1000000	0.1	0.000000001	0.813281308324263
NO	100000	0.01	0.00001	0.8133700605842246
NO	300000	0.001	0.00001	0.8137905212428943
NO	1000000	0.001	0.00001	0.8133700559863668
NO	300000	0.01	0.00001	0.813281888394884
NO	1000000	0.0001	0.00001	0.8103282814240282
NO	500000	0.24	0.00000000001	0.813281308324263
NO	2000000	0.268	0.00000000001	0.813281308324263
NO	100000	0.268	0.00001	0.813281308324263
NO	1000000	0.268	0.00001	0.813281308324263
NO	10000	0.268	0.00001	0.8132823760586806
NO	1000	0.268	0.00001	0.8152709406321238

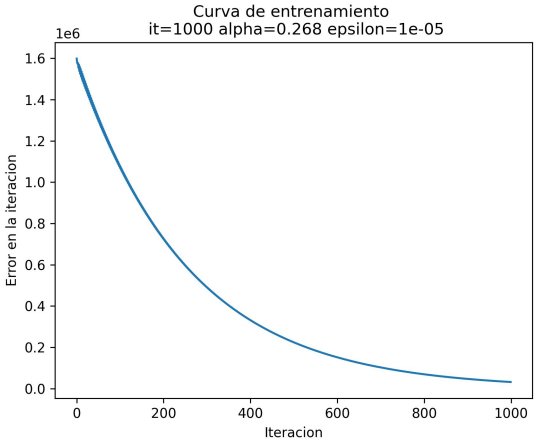
Para medir la precisión del modelo, utilizamos el coeficiente de determinación. Los resultados del entrenamiento indican que el mayor coeficiente de determinación para los datos preprocesados incluyendo los datos atípicos es de 0.7594. Mientras que los resultados obtenidos para los datos preprocesados que obvian los datos atípicos es de 0.8152.

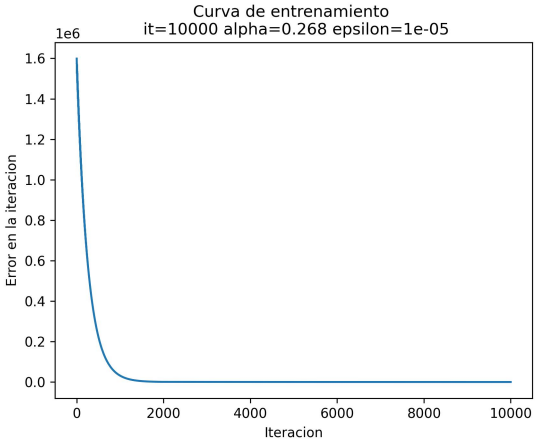
Entrenamiento con datos atípicos

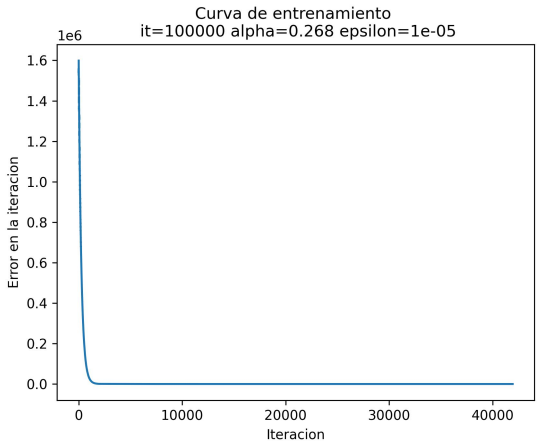


Por el gráfico de la curva de entrenamiento con datos atípicos, se puede observar que el error no se acerca a 0 hasta luego de las 2000 iteraciones.

Entrenamiento sin datos atípicos







Por otra parte, para el entrenamiento sin datos atípicos, el error se acerca a 0 a partir de las 800 iteraciones para finalmente mantenerse en aproximadamente a 0 para el resto de las 1000 iteraciones o más. Es posible ver que al eliminar los datos atípicos se mejora la precisión del modelo. Pero, el modelo no será bueno cuando deba predecir con datos que se asemejen a estos datos eliminados ya que se consigue un coeficiente de determinación de 0.7594.
Además, durante el entrenamiento se eligió un alpha hasta 0.268, ya que los números que se generan durante la ejecución del algoritmo son demasiado grandes y se genera un overflow al hacer el cálculo con las matrices de numpy, a pesar de que los valores se encuentran normalizados. También, se calculó el error medio relativo. Mientras aumenta el número de iteraciones disminuye este error, por lo que los errores en las predicciones son menores con un mayor número de iteraciones. Sin embargo, al realizar más de 100.000 iteraciones, se mantienen los valores para los errores relativos promedio (0.2719356684223598) y el error máximo no disminuye (2.6444), por lo que este valor de iteraciones será escogido para obtener nuestra hipótesis.

Hipótesis seleccionada

Debido a que el coeficiente de determinación obtenido por el modelo entrenado sin datos atípicos es mayor al conseguido por el modelo con datos atípicos, seleccionamos la hipótesis del primero.

La forma genérica de la hipótesis del modelo es:

$$hw(x) = W_0 + \sum_j W_j * x_j \tag{15}$$

Donde W_0 es el término Bias y los W_j son los pesos calculados.

Así, los resultados obtenidos en el entrenamiento eliminando los datos atípicos y seleccionando los siguientes valores:

- iteraciones = 100.000
- alpha = 0,268
- epsilon = 0,0001

Se obtienen los siguientes pesos:

Término Bias	Year	Kilometer	Engine
-159415.245	1210540.94	-483111.657	981706.886
Seating Capacity	Fuel Tank Capacity	Make-Honda	Make-Maruti Suzuki
-52611.8027	375197.483	-180732.523	-207042.76
Make-Hyundai	Make-Toyota	Make-Skoda	Make-Nissan
-152078.088	-20798.7926	-149104.056	-276577.49
Make-Renault	Make-Tata	Make-Volkswagen	Make-Ford
-313692.729	-211484.485	-192252.595	-365967.468
Make-Audi	Make-Mahindra	Make-MG	Make-BMW
391652.413	-256070.729	300794.847	360043.872
Make-Mercedes-Benz	Make-Jeep	Make-Kia	Make-Volvo
177629.495	186904.312	197275.939	331009.776
Make-Isuzu	Make-Fiat	Make-MINI	Make-Land Rover
65727.6272	-228192.975	585169.492	1071826.13
Make-Mitsubishi	Make-Datsun	Make-Chevrolet	Make-Ssangyong
206991.865	-496312.928	-355532.612	-628600.785
Fuel Type-Petrol	Fuel Type-Diesel	Fuel Type-CNG	Fuel Type-LPG
-95500.7971	-41771.6854	-121796.72	280545.796
Fuel Type-CNG + CNG	Fuel Type-Petrol + CNG	Transmission-Manual	Transmission-Automatic
21885.1851	-202777.024	-150694.104	-8721.14125
Owner-First	Owner-Second	Owner-Fourth	Owner-Third
52652.793	27795.142	-368461.159	-13638.0202
Owner-UnRegistered Car	Owner-4 or More		
388.985	141847.01		

Análisis de la hipótesis

Con respecto a las variables numéricas, se puede ver que el peso de la columna "Year" es 1.210.540. En particular, es un número positivo en la escala 10^6. Esto nos indica que mientras más nuevo sea el vehículo, más caro será. Esto se repite para las columnas "Engine" y "Fuel Tank Capacity". En síntesis, mientras más actual sea el vehículo, tenga un motor con más cc y un tanque de combustible con mayor capacidad, más caro será el carro.

En contraste, las variables "Kilometer" y "Seating Capacity" son negativas. Lo que significa que mientras más kilometraje recorrido tenga un vehículo y más capacidad de asientos posea menor será su precio.

Los pesos de las variables categóricas indican qué categorias perjudican o benefician al precio del vehículo.

Entre los distintos tipos de combustible, puede verse que el combustible que más valor agrega al vehículo es el "LPG". Mientras, el que más perjudica al precio es el de "Petrol". Sin embargo,dejando de lado a "LPG", se nota que los valores de los pesos están en el intervalo (-95500, 21885). Y, estas variables son mutuamente exclusivas. Por tanto, se puede decir que el combustible no afecta tanto sobre el precio final del vehículo si no utiliza "LPG".

Por otra parte, ambos pesos de la transmission son negativos. Sin embargo, si se restan ambos pesos, se obtiene que estos difieren en $-150694{,}104 - -8721{,}14125 = 141{,}973$. Por tanto, es claro que los vehículos con transmision manual son más baratos que sus contrapartes con transmision automática.

Finalmente, puede verse que los vehículos que han tenido menos de 5 dueños son más caros. Sin embargo, el peso de la variable de 4 dueño o más es mayor que las otras. Esto puede deberse a una falta de ejemplos en el dataset, a que esta variable se solapa con la variable "Owner-Fourth" o que simplemente debió haberse eliminado del dataset.

Conclusiones

Con el algoritmo de descenso de gradiente de Russel y Norvig es posible conseguir una aproximación a un modelo lineal, este algoritmo retorna una hipótesis. Con los debidos datos, se puede utilizar para conseguir hipótesis que intenten predecir características de los datos que se le dan como entrada. Sin embargo, los datos que se usan para entrenar el modelo juegan un rol importante en la precisión del modelo. En el presente informe, se ha descrito el proceso de entrenamiento de un modelo cuyas predicciones cuentan con métricas relativamente aceptables. Para el tratamiento de los datos, se eliminaron los registros con información faltante. Se normalizaron las variables numéricas y se dividieron en n variables dummys las variables categóricas. Además, se filtraron los registros con datos atípicos para garantizar la precisión del modelo en los datos comunes. El resultado es un modelo capaz de predecir correctamente para los casos semejantes a los datos comunes pero con una pérdida de generalidad en los casos no comunes o los parecidos a los datos atípicos.

Referencias

Khadka, D. (2021, December 11). Linear regression using gradient descent for beginners—intuition, math and code. Medium. https://medium.com/analytics-vidhya/linear-regression-gradient-descent-intuition-and-math-c9a8f5aeeb22

Russell, S. J., y Norvig, P. (2020). Artificial intelligence: A Modern Approach.