Práctica no. 1: Regresión lineal

Jair Antonio Bautista Loranca
a01365850@itesm.mx
Tecnológico de Monterrey
Monterrey, N.L., México

Maximiliano Zambada Camacho
a01570146@itesm.mx
Tecnológico de Monterrey
Monterrey, N.L., México

RESUMEN

La regresión lineal es uno de los modelos fundamentales para el entrenamiento supervisado de IAs. Si se puede considerar como el más sencillo, este nos permite conocer la tendencia de un conjunto de datos y además proporciona información suficiente para predecir el comportamiento de nuevos datos que estén relacionados con el proceso modelado. A través de esta práctica se discutirá los detalles relacionados con la teoría e implementación de una regresión lineal.

ACM Reference Format:
Jair Antonio Bautista Loranca and Maximiliano Zambada Camacho. 2021. Práctica no. 1: Regresión lineal. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1. INTRODUCCIÓN

Dados los avances que las computadoras han traído a nuestro día a día se ha hecho bastante sencillo el tratamiento una gran cantidad de datos. Esto ha permitido que el procesamiento de datos sea más sencillo que nunca. Por lo tanto las técnicas y metodologías para análisis de datos han tomado mucho mayor relevancia.

. Por ello es que en esta práctica se enfocará en la regresión lineal. Que es uno de los primeros modelos que nos permite conocer el comportamiento de un conjunto de datos y obtener información que nos permita predecir nuevos datos de entrada que sigan el modelo establecido.

2. CONCEPTOS PREVIOS

. El modelo de regresión lineal consiste en una generalización de la ecuación de de una recta.

$$h\left(\vec{\beta},\ \vec{x}\right) = \sum_{i=0}^{m} \beta_i x_i = \beta_0 x_0 + \beta_1 x_1 + \ldots + \beta_m x_m$$

Esto se puede reescribir como vectores de la siguiente manera.

$$h\left(\vec{\beta},\ \vec{x}\right) = \vec{\beta} \cdot \vec{x}$$

. Este modelo nos permite predecir una salida para un vector $\vec{x}$ dado, sin embargo desconocemos los valores del vector $\vec{\beta}$. Por lo tanto requerimos entrenar el modelo para calibrar los coeficientes β al conjunto de datos que tenemos.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

. Para el entrenamiento requerimos un conjunto de entrenamiento. Debido a que cada entrada a nuestro modelo consiste en un vector $\vec{x}$ entonces podemos formar el conjunto de entrenamiento como la matriz X de dimensiones $n \times m$. Donde cada renglón corresponde a un vector de entrada para el modelo. Además requerimos del vector $\vec{y}$ que corresponde a los resultados históricos que tenemos del modelo.

$$X = \begin{pmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vdots \\ \vec{x}_n \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1m} \\ x_{21} & x_{22} & \ldots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nm} \end{pmatrix}, \ \vec{y} = \begin{pmatrix} \vec{y}_1 \\ \vec{y}_2 \\ \vdots \\ \vec{y}_n \end{pmatrix}$$

. Para entrenar nuestro modelo con los datos que tenemos necesitamos una función de costo que determina que tan aproximado es nuestro a los datos que tenemos. Usualmente se utiliza la función de error cuadrático medio que para nuestro modelo quedaría de la siguiente manera.

$$\text{MSE}(\vec{\beta}) = \frac{1}{n} \sum_{i=1}^{n} \left(h(\vec{\beta},\ \vec{x}_i) - \vec{y} \right)^2$$

. Existe una forma cerrada para minimizar el error cuadrático medio para nuestro modelo, sin embargo no es computacionalmente viable. Por ello recurrimos al método de gradiente para encontrar el $\vec{\beta}$ que minimice el error para nuestros datos.

. Este método consiste en lo siguiente, dado que el costo está en función del vector $\vec{\beta}$. Entonces podemos utiliza el gradiente negativo para encontrar la dirección a donde se minimiza la función. Por lo tanto podemos minimizar el costo por medio del siguiente algoritmo.

Algorithm 1 Descenso de gradiente

1: Inicializar $\vec{\beta}_0$
2: $\alpha \leftarrow (0, 1]$
3: $j \leftarrow 0$
4: **while** $\left\| \nabla \text{MSE}(\vec{\beta}) \right\| \not\approx 0$ **do**
5: $\quad \vec{\beta}_{j+1} \leftarrow \vec{\beta} - \alpha \nabla \text{MSE}(X, h)$
6: $\quad j \leftarrow j + 1$
7: **end while**

3. METODOLOGÍA

La metodología a seguir durante el proceso de desarrollo de la práctica fue una metodología colaborativa. Ambos integrantes trabajamos de manera paralela en los pasos de la práctica para poder crear los códigos necesarios. Tanto para el desarrollo del análisis de las bases de datos en lenguaje R como el análisis de regresión

lineal y gradiente descendente en Python fueron programados de manera colaborativa.

. Para el análisis de estadística descriptiva realizado en R para ambos archivos de datos, realizamos el procedimiento visto anteriormente, donde simplemente se crean data sets a partir de los archivos, para poder sacar el análisis estadístico y las gráficas correspondientes con sus respectivas funciones.

. Continuando con el análisis y evaluación de regresión lineal, se realizó un proceso más elaborado. Primeramente, se investigaron las librerías correspondientes para realizar desde los data sets hasta las regresiones lineales en sus gráficas. Esto se consiguió importando las librerías de Pandas, NumPy, Seaborn, MatPlotLib, y la más importante, Scikit Learn, de donde se utiliza la regresión linear. Para comenzar a realizar el código, se fue desarrollando para un solo archivo, con el objetivo de realizar pruebas individuales e ir analizando fallas. Después de importar el data set con Pandas y dividir las variables en X y Y, se dividieron los datos en 80 % ara el Training Set y 20 % para el Test Set, de manera que no se utilicen los 10,000 datos del archivo. Utilizando el Training Set, se puede realizar la función de LinearRegression() y .fit(), con lo que se consigue este análisis de regresión linear. Ya que se consigue esta regresión, se pudo proceder a sacar una variable de predicciones, utilizando esta regresión, y el Test Set, con la función de .predict(XTest).

. Finalmente, ya que se tiene esto realizado, lo único que queda pendiente es realizar la gráfica con el Test Set, y trazar la línea de regresión linear utilizando la variable de predicción que se consiguió anteriormente. Asimismo, con las funciones correspondientes se pudo evaluar el algoritmo, con el MAE, MSE y el Root MSE.

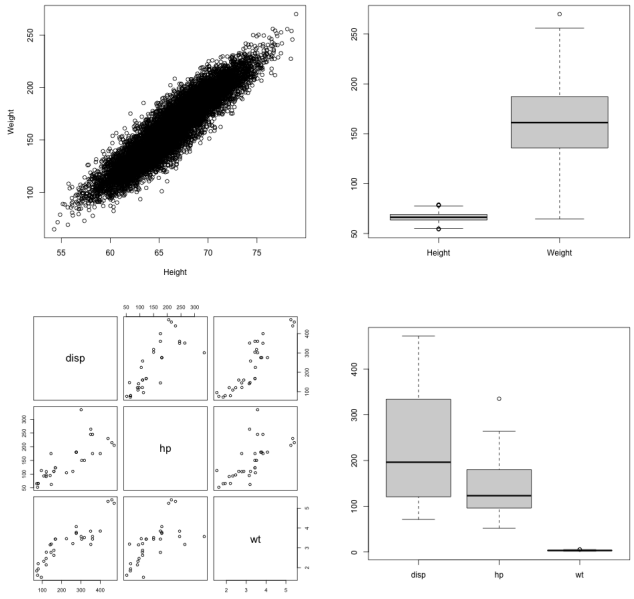
. Ya que se finalizaron las pruebas y se consiguió el resultado esperado, el siguiente paso fue crearlo en una función para poder ejecutarla por separado. Esto mismo se realizó para el archivo de texto de mtcars, sin embargo, fue un poco diferente, ya que se utilizaron 3 variables en este caso. La diferencia que hubo fue, primero que nada, separar en X, Y y Z los datos, y juntar X y Y, ya que Z sería la salida. Después de este paso, el proceso fue el mismo para sacar la regresión linear y las predicciones. Al final, la gráfica se consiguió en 3D, utilizando plt.axes(projection="3d"), y la evaluación se consiguió de la misma manera.

. Finalmente, para el código de la Gradiente Descendente, se siguieron los siguientes pasos. Primeramente, fue un proceso más lento y con mayores iteraciones en las correcciones y pruebas, ya que era necesario implementar un algoritmo nuevo para nosotros. Esto fue realizado utilizando solamente la librería de NumPy, ya que dentro de la misma se encontraban todas las funciones que se necesitaban para el manejo de nuestras variables. De la misma manera que con la regresión linear, se importó a un data set el archivo, pero en este caso se dividieron en dos arreglos, utilizando np.array. Para poder calcular esta gradiente, utilizamos los datos divididos en X y Y, así como una variable Alpha, la cual la inicializamos como 0.00001. Con estas variables, se procedió a realizar las iteraciones necesarias con la implementación de la fórmula de Gradiente Descendente, para finalmente poder conseguir el valor del intercepto, y el coeficiente del gradiente, y así poder formar la gráfica correspondiente.

4. RESULTADOS

4.1. Análisis por R

Por medio de los resultados obtenidos en R, pudimos utilizarlos como referencia para el resto del trabajo debido a que nos presenta un resumen bastante útil sobre el comportamiento de los datos y podemos estar más seguros de que nuestras predicciones realizadas sean precisas.



genero.txt	Height	Weight	mtcars.txt	disp	wt	hp
min	54.26	64.7	min	71.1	1.513	52
1st qu.	63.51	135.8	1st qu.	120.8	2.581	96.5
median	66.32	161.2	median	196.3	3.325	123
mean	66.37	161.4	mean	230.7	3.217	146.7
3rd qu.	69.17	187.2	3rd qu.	326	3.61	180
max	79	270	max	472	5.424	335

. Como podemos ver nuestros datos tienen una tendencia lineal, lo que nos indica que es adecuado utilizar modelos lineales.

4.2. Regresión lineal vs Descenso del gradiente

```
=== genero.txt ===
Intercept value:  [-350.19040286]
Coefficients:  [[7.70936331]]
          Actual   Predicted
0     138.085796  148.789477
1     187.363366  168.435201
2     216.533191  224.318845
3     131.761443  157.702560
4     157.718438  149.733294
...          ...         ...
1995  189.856786  182.410166
1996  202.906859  203.889260
1997  152.420031  159.176764
1998  170.100647  155.864045
1999  158.206206  144.834493

[2000 rows x 2 columns]
Mean Absolute Error: 9.662897842061215
Mean Squared Error: 146.53677213957743
Root Mean Squared Error: 12.105237384684958
=== mtcars.txt ===
Intercept value:  [-8.8817842e-16]
Coefficients:  [[-1.38254841e-18  1.00000000e+00]]
   Actual  Predicted
0   4.070      4.070
1   3.435      3.435
2   3.440      3.440
3   2.320      2.320
4   5.345      5.345
5   5.250      5.250
6   3.170      3.170
Mean Absolute Error: 3.8064789415719656e-16
Mean Squared Error: 3.380832450947193e-31
Root Mean Squared Error: 5.814492627002973e-16
λ
```

. Comparando los resultados obtenidos por la regresión lineal y por nuestra implementación del algoritmo de descenso de gradiente. Podemos ver que nuestros resultados no son tan precisos como los obtenidos con SKLearn. Consideramos que esto se debe a la inicialización de $\vec{\beta}_0$ y el α elegido. Específicamente lo que ocurre es que si utilizamos αs muy pequeños el algoritmo converge de manera más sutil y precisa pero tiene el costo de que esto puede ser muy tardado si $\vec{\beta}_0$ está muy alejado de $\vec{\beta}_j$ óptimo. En el otro caso, cuando elegimos valores α muy grandes, es que el algoritmo tiende a "saltar" demasiado en el espacio de valores $\vec{\beta}$ por lo que no logra converger en una solución esperada y de hecho diverge. Todo esto implica que el desempeño y precisión del algoritmo de descenso de gradiente está en función de α y $\vec{\beta}$.

. Como observación podemos mencionar que podríamos mejorar el desempeño del algoritmo de diferentes maneras. Dado que el algoritmo le favorece que $\vec{\beta}$ esté cercano a la solución, entonces podemos correr el algoritmo varias veces. De manera que la primera ejecución obtendremos un $\vec{\beta}$ cerca a nuestra solución, esto permite utilizarlo como una nueva β inicial y podemos aumentar ligeramente α para que pueda converger con mayor rapidez a la solución.

5. CONCLUSIONES Y REFLEXIONES

Por lo discutido en el documento podemos ver que la regresión lineal y el descenso de gradiente son herramientas importantes para la creación de modelos de predicción. Siendo la regresión lineal muy utilizada para modelos sencillos donde podemos ver comportamientos proporcionales en los datos. Y por otro lado el descenso de gradiente

Jair Antonio. La realización de esta práctica me ha permitido darme cuenta de la importancia que tienen diferentes áreas de las matemáticas en los nuevos avances de inteligencia artificial. Utilizando conceptos de álgebra lineal, calculo multivariable, y estadística, he podido comprender el funcionamiento de la regresión lineal y el descenso de gradiente.

Maximiliano Zambada. Con el trabajo que realizamos en esta práctica, pude familiarizarme más con el lenguaje de Python. Como soy alguien nuevo al lenguaje, el cambio de sintaxis es un poco complicado al principio, pero al conocer bien el proceso a seguir y las librerías utilizadas, es más sencillo implementar soluciones. Al combinar estos nuevos conocimientos con el uso de fórmulas de álgebra y la creación de gráficas, puedo comprender más la aplicación de varias técnicas de análisis de datos.