

UNIVERSIDAD AUTÓNOMA DE YUCATÁN (UADY) CAMPUS DE CIENCIAS EXACTAS E INGENIERÍA FACULTAD DE INGENIERÍA (FIUADY)

**-Aprendizaje automático-**

**Proyecto #1: Algoritmos de regresión**

Izquierdo Berdugo Luis-Ingeniería Mecatrónica

Orozco Poot Jorge-Ingeniería Mecatrónica

Sánchez Franco Nicolás Ingeniería Mecatrónica

Zurita Martínez Leonardo-Ingeniería Mecatrónica

Profesor: Edson G. Estrada López

PERIODO AGOSTO – DICIEMBRE DE 2021

Fecha de elaboración del reporte: lunes 4 de octubre del 2021

Fecha límite de entrega: lunes 4 de octubre del 2021

Índice

[Objetivo 1](#_Toc84280047)

[Objetivo general 1](#_Toc84280048)

[Objetivos particulares 1](#_Toc84280049)

[Contenido 1](#_Toc84280050)

[Problemas para solucionar 1](#_Toc84280051)

[Bases de datos 1](#_Toc84280052)

[Descripción general del proyecto 1](#_Toc84280053)

[Tabla de requerimientos 1](#_Toc84280054)

[Marco teórico 2](#_Toc84280055)

[Batch Gradient Descent 2](#_Toc84280056)

[Regresión polinomial 2](#_Toc84280057)

[Regresión polinomial multivariable. 4](#_Toc84280058)

[Descripción de la implementación 4](#_Toc84280059)

[Investigación de las variantes del algoritmo 5](#_Toc84280060)

[Stochastic Gradient Descent 5](#_Toc84280061)

[Mini-batch Gradient Descent 5](#_Toc84280062)

[Stochastic Gradient Descent 5](#_Toc84280063)

[Mini-batch Gradient Descent 5](#_Toc84280064)

[Listado de bugs conocidos 6](#_Toc84280065)

[Conclusiones 6](#_Toc84280066)

[Conclusión general 6](#_Toc84280067)

[Conclusiones individuales 7](#_Toc84280068)

[Luis Izquierdo 7](#_Toc84280069)

[Isaac Orozco 7](#_Toc84280070)

[Nicolas Sánchez 7](#_Toc84280071)

[Leonardo Zurita 7](#_Toc84280072)

[Referencias 8](#_Toc84280073)

# Objetivo

## Objetivo general

* Aplicar diversas variantes del algoritmo de regresión y modelos de optimización en la solución de problemas de predicción.

## Objetivos particulares

* Implementar el algoritmo de regresión lineal y hallar mediante diferentes métodos de optimización.
* Implementar el algoritmo de regresión polinomial

# Contenido

## Problemas para solucionar

* Estimar el costo de un seguro.
* Estimar la puntuación de la calidad de un vino.

## Bases de datos

* https://archive.ics.uci.edu/ml/datasets/wine+quality
* https://www.kaggle.com/mirichoi0218/insurance
* https://github.com/stedy/Machine-Learning-with-R-datasets/blob/master/insurance.csv

## Descripción general del proyecto

1. Se implementó el algoritmo de regresión lineal múltiple para estimar el costo de un seguro. Calculando los valores de con los algoritmos de la Ecuación normal, Batch gradient descent, stochastic gradient descent y mini batch gradient descent, se realizó una comparativa entre los resultados obtenidos de cada algoritmo.
2. Usando el modelo de regresión lineal con la ayuda de la librería de sklearn se estimó la puntuación de calidad de un vino.
3. Generamos una nube de puntos artificialmente con una función polinomial y aplicamos el modelo de regresión polinomial para estimar los valores de .
4. Implementamos mediante modelos de regresión polinomial y las librerías de sklearn una estimación para la puntuación de la calidad de un vino.

## Tabla de requerimientos

|  |  |  |
| --- | --- | --- |
| Requisito | | Estado |
| Investigación de las variantes del algoritmo “Batch Gradient Descent” | | Completado |
| Regresión lineal multivariable para costo de seguro | | Completado |
| Hallar aplicando: | Ecuación Normal | Completado |
| Batch Gradient Descent | Completado |
| Stochastic Gradient Descent | Completado |
| Mini-Batch Gradient Descent | Completado |
| Regresión lineal multivariable para calidad de vino | | Completado |
| Investigación del algoritmo “Polynomial Regression” | | Completado |
| Regresión polinomial para nube artificial de datos | | Completado |
| Regresión polinomial multivariable para calidad de vino | | Completado |

## Marco teórico

### Batch Gradient Descent

Es un algoritmo de optimización iterativo en el que todos los datos de entrenamiento se tienen en cuenta para obtener cada resultado. Se toma el promedio de los gradientes de todos los ejemplos de entrenamiento y luego se usa ese gradiente medio para actualizar los parámetros. Así que eso es solo un paso de descenso de gradiente en una época. Este algoritmo es ideal para múltiples errores convexos o relativamente suaves.

### Regresión polinomial

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Ilustración Regresión lineal sobre un conjunto de datos claramente no lineales

En algunas ocasiones, la línea recta que se ajusta a un modelo de predicción no es la mejor elección para un conjunto de datos donde no existe una relación lineal entre las variables independientes y la variable dependiente.

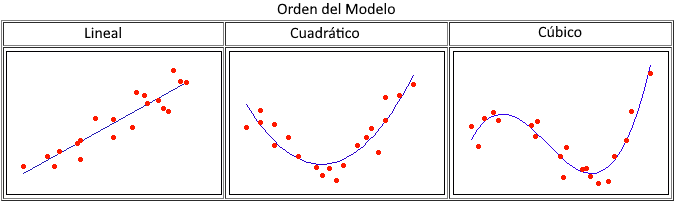


Ilustración Modelos de regresión de diferente orden

Agregar predictores adicionales a una regresión lineal la convierte en una regresión polinomial, siempre que estos predictores se obtengan de elevar cada predictor original a una cierta potencia. El modelo de la regresión lineal de una variable tiene esta forma:

Los predictores que se agregan se obtienen de un predictor original elevado a una cierta potencia, es decir generamos más datos para la correlación con la intención de realizar un modelo con menor error. De esta manera podemos realizar un modelo de ajuste con una función de algún cierto grado , que puede ser una mejor función de predicción:

De forma matricial la función guarda una relación muy cercana al modelo de regresión lineal, donde lo que sufre modificaciones es la matriz de predictores formada por las potencias de los predictores hasta grados.

Incluso la estimación de los parámetros se realiza con la misma formula de estimación que en el caso lineal:

Esto se debe a lo antes mencionado sobre que una regresión polinomial se puede ver como si fuera una regresión lineal, pero con múltiples variables, es decir cada predictor de la regresión polinomial sería una variable independiente distinta:

### Regresión polinomial multivariable.

Al tener múltiples variables independientes para realizar una regresión polinomial, las cosas cambian un poco. De forma general para 2 variables independientes y un polinomio de grado 2, tenemos el siguiente modelo:

Centramos nuestra atención en el último término del modelo, este se obtiene de forma general, realizando todas las combinaciones en 2 posibles de los regresores. Un modelo de grado 2 con tres variables independientes se vería de la siguiente forma.

Como se observa las combinaciones de 3 en parejas son 3, las enumeradas para los coeficientes , y . El resto de los términos se pueden obtener de forma intuitiva es como si sumásemos las funciones polinomiales de grado de cada variable independiente respectivamente. Otro punto para tomar en cuenta sobre la cantidad de coeficientes a estimar que será igual al número de columnas en la matriz .

Una vez que se genera la nueva matriz de regresores para hallar la estimación de basta con aplicar las mismas fórmulas que para el caso de la regresión lineal y la regresión polinomial de una variable independiente.

El problema de las regresiones polinomiales con varias variables independientes es la multicolinealidad, es decir, que exista entre los regresores del modelo relaciones aproximadamente lineales. Podemos predecir un regresor a partir de otro.

Otro inconveniente que surge es que un mayor grado para la función polinomial no contribuye necesariamente a mejorar la capacidad de predicción, porque podría generar un sobre ajuste del modelo a los datos de entrenamiento. Además, que las funciones polinomiales de mayor grado son más sensibles.

Imagen que contiene reloj, medidor, dibujo

Descripción generada automáticamente

Ilustración Sobre ajuste que se percibe con una función polinomial de grado mayor

## Descripción de la implementación

Para la descripción detallada del procedimiento y resultados revisar los códigos anexos en la carpeta ‘Códigos’:

1. Revisar: *“*[*Proyecto1\_Inciso\_2.ipynb*](https://github.com/LeonarZuma/Proyecto1-Aprendizaje-autom-tico/blob/main/Proyecto1/Codigos/Proyecto1_Inciso_2.ipynb)*”*

Se estimó el precio de un seguro por medio de la ecuación normal. Se uso los algoritmos de optimización Batch Gradient Descent, Stochastic Gradient Descent y Mini-batch Gradient Descent para estimar el valor .

1. Revisar: *“*[*Proyecto1\_Inciso\_3.ipynb*](https://github.com/LeonarZuma/Proyecto1-Aprendizaje-autom-tico/blob/main/Proyecto1/Codigos/Proyecto1_Inciso_3.ipynb)*”*

Se generó un modelo de regresión lineal multivariable para estimar la calidad del vino con las librerías de sklearn.

1. Revisar*: “*[*Proyecto1\_Inciso\_5\_1.ipynb*](https://github.com/LeonarZuma/Proyecto1-Aprendizaje-autom-tico/blob/main/Proyecto1/Codigos/Proyecto1_Inciso_5_1.ipynb)*”*

Se desarrollo un modelo de regresión polinomial a un conjunto de valores aleatorios.

1. Revisar: *“*[*Proyecto1\_Inciso\_5\_2.ipynb*](https://github.com/LeonarZuma/Proyecto1-Aprendizaje-autom-tico/blob/main/Proyecto1/Codigos/Proyecto1_Inciso_5_2.ipynb)*”*

Se generó un modelo de regresión polinomial multivariable para estimar la calidad del vino.

### Investigación de las variantes del algoritmo

### Stochastic Gradient Descent

Es una variante del algoritmo “Batch Gradient Descent” en la que se tiene un conjunto de datos muy grande por lo que calcular el gradiente en cada época se vuelve ineficiente por lo que se toma un ejemplo, se alimenta a la red neuronal y se le saca su gradiente, con este gradiente es que se actualizan los pesos. Este proceso se repite para todos los ejemplos del conjunto de datos.

Dado que estamos considerando solo un ejemplo a la vez, el valor fluctuará con los ejemplos de capacitación y **no** necesariamente disminuirá. Pero a la larga, verá que el valor disminuye con las fluctuaciones.

### Mini-batch Gradient Descent

Ya que en Stochastic Gradiente Descent se usa solo un ejemplo a la vez, no se puede implementar la implementación vectorizada en él. Esto puede ralentizar los cálculos. Para abordar este problema, se utiliza una mezcla de Batch Gradient Descent y Stochastic Gradient Descent.

No se usa todo el conjunto de datos a la vez ni se usa un solo ejemplo a la vez, se usa un lote de un número fijo de ejemplos de entrenamiento que es menor al conjunto de datos real y se le llama mini-lote. Hacer esto ayuda a lograr las ventajas de Batch Gradient Descent y de Stochastic Gradiente Descent. Después de crear los mini-lotes de tamaño fijo se elige uno de los mini lotes y se alimenta a la red con él, se calcula el gradiente y con este último se actualizan los pesos, estos pasos se repiten para todos los mini lotes creados

|  |  |  |
| --- | --- | --- |
| Variantes del algoritmo | Stochastic Gradient Descent | Mini-batch Gradient Descent |
| Ventajas | * Permite analizar muchos datos * Converge más rápido * Provoca actualizaciones de los parámetros con mayor frecuencia * Reduce el tiempo de computación al usar menos datos para el cálculo del gradiente * Fácil implementación * El proceso de actualización ruidoso puede permitir que el modelo evite mínimos locales | * Toma un lote mayor a un solo valor, pero menor a todo el conjunto de datos para calcular el gradiente volviéndose más exacto que el método estocástico * Se pueden vectorizar todos los datos * Evita mínimos locales * Tiene un proceso computacional más eficiente |
| Desventajas | * Toma un solo dato para calcular el gradiente, lo que lo vuelve ligeramente inexacto * Los valores en el descenso de la convergencia fluctúan * Los valores fluctúan alrededor del mínimo sin nunca obtenerlo realmente * Ya que solo se usa un dato a la vez no se pueden vectorizar todos los datos * Al actualizar el modelo con tanta frecuencia se vuelve costoso computacionalmente | * Los valores en el descenso de la convergencia fluctúan * Requiere la configuración de un hiperparámetro adicional “tamaño de mini lote” * La información del error se debe acumular en mini lotes de ejemplos de entrenamiento como el descenso de gradientes por lotes |
| Condiciones de uso | * Cuando se tienen muchos datos * Cuando no se necesita ser muy exacto * Cuando no se tiene mucha experiencia programando | * Cuando se tienen muchos datos * Cuando se quiere obtener resultado más exacto * Cuando se necesita una convergencia rápida |

## Listado de bugs conocidos

# No se tiene conocimiento de algún bug que pueda surgir del código realizado.

# Conclusiones

## Conclusión general

Existen diferentes formas de optimizar la forma en que obtenemos los valores de , los algoritmos estudiados tienen distintos comportamientos y lo que es aún más importante, trabajan con una cierta cantidad de datos. Esto a pequeña escala no es muy relevante, pero en cuanto crece la cantidad de datos, aplicar ecuaciones normales para estimar no es viable. Estudiar las formas de optimizar estos procesos nos acerca un poco a un concepto importante en el machine learning, optimizar los modelos es la clave del éxito. No nos sirve de mucho tener un modelo que sea perfecto si para realizar los cálculos nosotros tomamos una cantidad de tiempo exorbitante, es por lo que ponemos en balance los resultados y el tiempo. Los modelos tratan de aproximarse a la realidad, no se tratade imitarla por completo, porque el fenómeno en su totalidad es difícil de imitar, es una aproximación.

De igual modo con los modelos de regresión polinomiales, entender cuando usarlos es la clave de su optimización, pero nuevamente, llegar a esa comprensión requiere de adentrarnos a profundidad en los conceptos estadísticos y particularidades de cada conjunto de datos. En ocasiones no buscamos perfección, más bien similitud.

## Conclusiones individuales

### Luis Izquierdo

Definitivamente fue interesante la implementación en código de las diferentes técnicas de regresión lineal. Fue de gran utilidad comprender la diferencia entre los métodos de Stochastic y Mini-batch, ya que ayudó a programar de manera correcta todos estos. Una de las dificultades que se enfrentaron fue al momento de programar los mencionados anteriormente por separado, pero comprendiendo que tenían módulos muy parecidos y haciéndolo todo en la misma notebook, se lograron los resultados esperados.

### Isaac Orozco

Después de analizar los datos obtenido por cada método se puede apreciar que con todos se llega a resultados similares, por lo que realmente depende de que tan exacto se quiere ser, de que tantos recursos tenga la computadora con la que se esté trabajando, la cantidad de datos con la que se esté trabajando entre otras variables. En mi opinión uno de los logros importantes del proyecto fue poder entender e implementar las variantes del algoritmo “Batch Gradient desent” ya que a pesar de haber sido visto e implementado en clase el trabajar con sus variantes no solo nos ayudo a afianzar la teoría sino que también representó un reto al buscar su implementación.

### Nicolas Sánchez

Con este proyecto se pudo aplicar los conocimientos sobre regresión lineal multivariable para hacer predicciones con los datasets dados. Se pudo comparar las predicciones del modelo obtenido con sklearn contra los algoritmos elaborados a través de las ecuaciones. Por mi parte se programó el método de ecuación normal y BGD que, aunque se vea algo sencillo realmente se complicó bastante por momentos sobre todo en la parte de los tamaños de las matrices ya que muchas veces no eran correctas. El proyecto logró que entendiera mejor la teoría de la regresión ya que al hacerlo tú mismo tienes que andar investigando y entendiendo que métodos tienes que utilizar para resolver el ejercicio de manera correcta.

### Leonardo Zurita

La regresión lineal es una herramienta potente para realizar algoritmos de predicción del comportamiento de variables. Aunque en primera instancia pudiese parecer un modelo sencillo, desprende una gran complejidad que demuestra su amplio espectro de aplicación en problemas de predicción, como precios de casas o seguros. Si bien el mundo no es tan sencillo como el comportamiento de una función, la realidad es que vivimos de realizar aproximaciones que puedan modelar el comportamiento de ciertos sucesos (físicos, demográficos, económicos, etc.). Estos tipos de herramientas tienen como objetivo ayudar en la toma de decisiones, basados en datos estadísticos, lo importante es saber una buena interpretación de los resultados, eso dependerá en gran medida de nuestro conocimiento estadístico.

# Referencias

* Brownlee J. (2017) Machine Learning Mastery, A Gentle Introduction to Mini-Batch Gradient Descent and How to Configure Batch Size, Recuperado de: <https://machinelearningmastery.com/gentle-introduction-mini-batch-gradient-descent-configure-batch-size/>
* Raschka, S. (2015). *Python machine learning*. Packt publishing Ltd.(pp-298)
* Shushant P. (2019) toward data science, Batch, Mini Batch & Stochastic Gradient Descent, Recuperado de: <https://towardsdatascience.com/batch-mini-batch-stochastic-gradient-descent-7a62ecba642a>
* Sinha, P. (2013). Multivariate polynomial regression in data mining: methodology, problems and solutions. *International Journal of Scientific and Engineering Research*, *4*(12), 962-965.
* Stojiljkovic M, (2021) Reak Python, Stochastic Gradient Descent Algorithm with Python and NumPy, Recuperado de: <https://realpython.com/gradient-descent-algorithm-python/>
* Yang L. (2018) towards data science, Vectorization Implementation in Machine Learning, Recuperado de: <https://towardsdatascience.com/vectorization-implementation-in-machine-learning-ca652920c55d>