Nombre del Laboratorio

Juárez Torres, Carlos Alberto[2]. kr_ipton @ciencias.unam.mx
Vega Gonzales, Ruben [2]. ruben_vega34@ciencias.unam.mx
1. Departamento de Matemáticas. Facultad de Ciencias, UNAM.
6 de diciembre de 2021

Resumen: El laboratorio está constituido por el ajuste de un plano a una nube de datos en 3d mediante un algoritmo a elección el cual fue descenso de gradiente, con el uso de una función costo y su evaluación de la complejidad computacional para solucionar el problema y sus aplicaciones.

Palabras clave: Plano, datos, Gradiente, Algoritmo, Costo, Complejidad.

1. Introducción

El laboratorio presenta un problema sobre el uso de aprendizaje automático para solución de problemas en el cual a opción se podía usar uno de los dos algoritmos, grid search o descenso del gradiente en el cual optamos por usar este ultimo para solucionar el problema en cuestión sobre el ajuste de un plano a una nube de datos.

2. Metodología

- Como ya se mencionó optamos por usar el descenso del gradiente para resolver ajustar un plano a la nube de datos por lo que comenzamos a definir la función de coste como:

$$\frac{1}{n}\sum_{1}^{n}[z-(ax+by+c)]^2 \tag{1}$$

- Para manejar la base de datos primero interpretamos a las variables donde concluimos mediante el uso de el mapa a que corresponden concretamente cada una, de modo que pasamos los datos a un formato .CSV, por lo tanto su acceso a dicha base se convirtió más fácil.

- Para la elaboración de programa se usó en este caso un IDE de archivos .PY en este caso usamos a colaboratory para archivos .IPYNB dado que se tuvo que elaborar más modificaciones para elaborar el código por lo que optamos por fragmentar la elaboración para ser más dinámicos.

- Concretamente utilizamos librerías ya usadas como numpy, matplotlib y pandas cada uno para el correcto manejo de los datos obtenidos.

- El uso de gráficas 3d fue realmente de gran ayuda para visualizar los errores que podían existir en el código, aunque los fragmentos de código referentes a la visualización de datos no quedaron propiamente en la versión final.

3. Resultados y discusión

Tras lo dicho anteriormente obtuvimos un plano que intersecta a la nube de datos, donde el programa es capaz de dar la ecuación de dicho plano tras los ajustes creados a partir de que el plano inicial fue:

$$z = x + y + c \tag{2}$$

y el dado por el algoritmo terminó siendo:

$$z = 0{,}42838319342789455x + 1{,}0772912135244803y + 1{,}0060862 \tag{3}$$

el cual al graficarse quedó como la fig 1

En principio no se tuvieron problemas con los datos ni con la generación del plano salvo algunos problemas de sintaxis pero lo más interesante es el manejo de los datos en colaboratory por lo

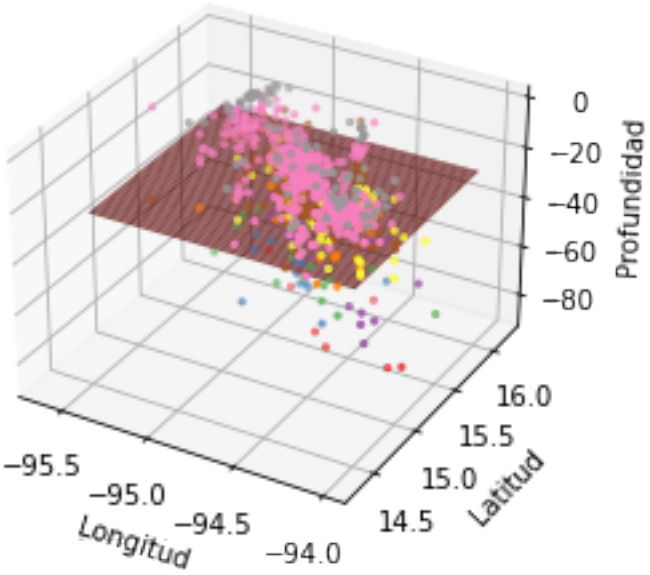

Figura 1: Plano ajustado a datos de temblores

que para la generación del gráfico 1 se tuvo que elaborar en un medio local para obtener dicho resultado.

Por otro lado se generó otra función para el calculo del coste la cual va a ser añadida a la función de ajuste genera el historial de las diferencias del la estimación y los datos.

De dicho modo se obtuvo a la gráfica 2 la cual fue hecha para 1000 iteraciones y los datos resultantes quedaron en una base de datos exportada a CSV, la cual queda de un tamaño de 950001 elementos los cuales se usaron para generar la 2. la cual realmente cambia solo un poco si se usan solo 100 iteraciones.

Por otro lado queda la complejidad del algoritmo, concretamente lo tenemos que en google colaboratory el programa no pudo ser corrido dado problemas con el manejo de los datos por dos usuarios así que se corrió en dos computadoras con las siguientes especificaciones:
Computador A

1. Procesador AMD Ryzen 3 3200g
2. Tarjeta gráfica Radeon Vega 8

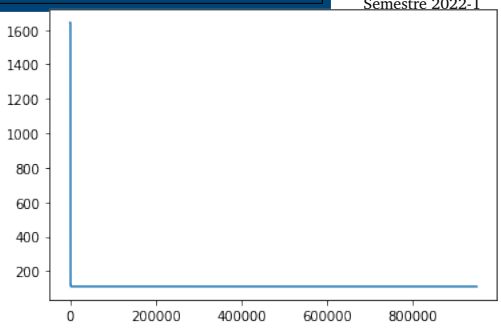

Figura 2: Eficiencia

3. Memoria 16 Gb

Computador B

1. Procesador Intel Core i3 1005g1
2. Tarjeta gráfica Intel Iris Plus G1
3. Memoria 4 Gb

En donde encontramos lo siguientes resultados en tiempos 1 para cada una de las computadoras donde como se enuncia por los tiempos que tomó correr el programa en la computadora A para el calculo de 1000 iteraciones no se realizó dicha prueba.

4. Conclusión

- Para finalizar tenemos que este laboratorio presentó un reto para obtener su solución pero tras obtener una mejor perspectiva gráfica del avance por medio de las gráficas elaboradas en el proceso fue más sencillo realizar el ajuste del plano, concretamente gran parte de los problemas surgen a partir de que no era sencillo encontrar una buena documentación para resolver el problema pero en cuanto pudimos comprender la aplicación del algoritmo del descenso del gradiente el programa fue terminado aún más rápido.

A	B	iteraciones	Calculos
18m 30.3 s	40m, 12.5s	100	95000
184m27s	No se hizo	1000	950000

Tabla 1: Tiempos de ejecución

- Este laboratorio nos permitió observar como podemos resolver nuestro trabajo de investigación en el cual buscamos obtener una función de ajuste a datos que serian previamente cambiantes.

- Particularmente el descenso del gradiente por lo que investigamos es un algoritmo bastante presente en algoritmos de IA y verlo de esta forma es claramente lo más superficial del potencial de un algoritmo como este.

5. Referencias

Referencias

[1] I. Ruiz. "Descenso por gradiente (Gradient descent)" Consulta: 30 de noviembre del 2021.

[2] Academy, K. (2021). Descenso de gradiente (artículo) — Khan Academy. Consulta: 30 de noviembre del 2021., de https://es.khanacademy.org/math/multivariable-calculus/applications-of-multivariable-derivatives/optimizing-multivariable-functions/a/what-is-gradient-descent

[3] CSV, D. (2021). ¿Qué es el Descenso del Gradiente? Algoritmo de Inteligencia Artificial — DotCSV. Retrieved Consulta: 30 de noviembre del 2021., de https://www.youtube.com/watch?v=A6FiCDoz8_4+

[4] Martínez Heras, J. (2021). Gradiente Descendiente para aprendizaje automático - IArtificial.net. Consulta: 30 de noviembre del 2021. de https://www.iartificial.net/gradiente-descendiente-para-aprendizaje-automatico/