Universidad Autónoma de Querétaro

Facultad de Ingeniería
Maestría en Inteligencia Artificial
Machine Learning
Cecilia Gabriela Rodríguez Flores
Sheila Leyva López

Gradiente Descendente

12 de diciembre del 2022

Objetivo

Implementar el algoritmo del gradiente descendente para encontrar los hiperparámetros θ, tal que, la función de coste J(θ), que en este caso corresponde al MSE, minimice su valor.

Introducción

En aprendizaje profundo usualmente se trabaja con redes neuronales que incluyen muchas variables, entonces es necesario el uso del gradiente descendente para encontrar aquellos parámetros que permitan minimizar la función de coste, lo que a su vez permite optimizar el algoritmo diseñado. En este trabajo se utiliza el gradiente descendiente para reducir el valor el valor del MSE, sin embargo, también se puede utilizar cuando se requiere hacer clasificación, y entonces la función de coste es la entropía cruzada; incluso también se puede aplicar a funciones de coste que defina y requiera el autor, siempre y cuando la función de coste cumpla con los requerimientos para ser una función de coste.

En este trabajo se aplica dicha técnica de agrupamiento a la base de datos *Indicadores personales clave de enfermedad cardíaca*. La base de datos.

Específicamente se presenta la implementación del modelo matemático: Gradiente Descendente, con el objetivo de encontrar los hiperparámetros θ, tal que, permitan optimizar la función de coste del modelo de regresión lineal entre los datos.

Marco Teórico

Para la comprensión de la metodología y resultados de este trabajo es necesario definir otros modelos matemáticos, tal como, regresión lineal, covarianza, correlación de Pearson y función de costo.

Regresión lineal

La regresión lineal es un análisis, cuyo principal objetivo es la predicción de una variable dependiente, a partir del valor de otra variable, llamada independiente [1], [2], dicho enunciado se puede representar a partir de la siguiente ecuación matemática (adaptada de [2]):

$$h_\theta(x) = \theta_1 x + \theta_0 \quad \ldots ecuación\ (1)$$

Donde:

- $h_\theta(x)$: $hipótesis$.
- θ_1: $la\ pendiente\ de\ la\ recta$.
- θ_0: $constante\ u\ ordenada\ al\ origen$.
- x: datos.

En otras palabras, $h_\theta(x)$ es una hipótesis de cómo es el comportamiento de los datos en x.

Regresión logística

Ahora bien, la regresión logística es una técnica matemática utilizada en aprendizaje automático para la clasificación de datos. Básicamente, es una neurona. Recibe este nombre dado que, primero se realiza una combinación lineal y después se aplica una función logística, aunque no es una regresión sino un modelo de clasificación [3]. El proceso se describe a través del diagrama

de la Ilustración 1. Básicamente, lo que en regresión lineal se denominaba hipótesis $\boldsymbol{h_\theta}$ ahora es **Z** y $\sigma(Z)$ es la función de transferencia definida por la función sigmoidea.

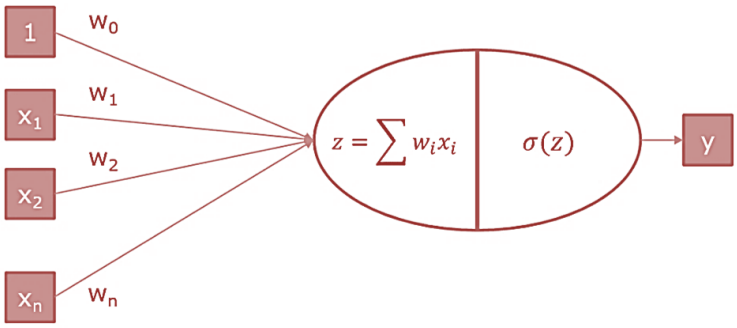


Ilustración 1. Esquema regresión logística.

Donde:

- X: atributos
- W: coeficientes o pesos.
- $\sigma(Z)$: función logística sigmoide

$$\sigma(Z) = \frac{1}{1 + e^{-Z}} \quad \ldots ecuación\ (18)$$

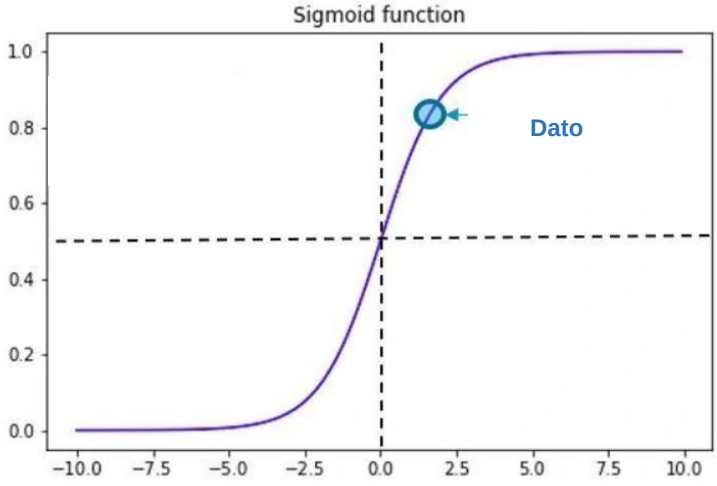


Ilustración 2. Función sigmoide.

Para medir el rendimiento de un modelo, en este caso el modelo de regresión lineal, existe una función, llamada función de costo o coste $J(\theta)$, la cual calcula el error entre los valores esperados o predichos por h_θ y los valores reales .

Dicha función de coste, en el mejor de los casos, será igual al error que se desea minimizar u optimizar, existen dos tipos de errores:

- Error Cuadrático Medio (MSE, por sus siglas en inglés)(ecuación adaptada de [8]):

$$MSE = \frac{1}{2m}\sum_{i=1}^{m}(valores\ estimados_i - valores\ reales_i)^2 \quad \dots ecuación\ (4)$$

- Error Absoluto Medio (MAE, por sus siglas en inglés):

$$MAE = \frac{1}{2m}\sum_{i=1}^{m}|valores\ estimados_i - valores\ reales_i| \quad \dots ecuación\ (5)$$

Donde:

- m: $es\ el\ número\ total\ de\ datos$.

Mencionado todo lo anterior, existe un método matemático que permite encontrar el valor mínimo de la función de coste, el método del gradiente descendente.

El gradiente, se considera como la versión generalizada de la derivada en cálculo diferencial de una variable. Es decir, el gradiente es el conjunto de las derivadas parciales de una función multivariable, además, como se busca el valor mínimo de la función, el gradiente es negativo o descendente.

El gradiente descendente para el caso particular en el que la función de coste es el MSE, se define como[2,3]:

Para el parámetro θ_0:

$$\theta_0 \coloneqq \theta_0 - \alpha\frac{\partial MSE}{\partial \theta_0} \dots ecuación\ (6)$$

$$\frac{\partial MSE}{\partial \theta_0} = \frac{\partial}{\partial \theta_0}\left[\frac{1}{2m}\sum_{i=1}^{m}(h_{\theta_0}(x_i^1) - y_i)^2\right] = \frac{1}{m}\sum_{i=1}^{m}(h_{\theta_0}(x_i^1) - y_i) \ \ldots ecuación\ (7)$$

De la ecuación (6) y (7) resulta la ecuación (8):

$$\boldsymbol{\theta_0 \coloneqq \theta_0 - \frac{\alpha}{m}\sum_{i=1}^{m}\left(h_{\theta_0}(x_i^1) - y_i\right)} \ldots ecuación\ (8)$$

Para el parámetro θ_1:

$$\theta_1 \coloneqq \theta_1 - \alpha\frac{\partial MSE}{\partial \theta_1} \ \ldots ecuación\ (9)$$

$$\frac{\partial MSE}{\partial \theta_1} = \frac{\partial}{\partial \theta_1}\left[\frac{1}{2m}\sum_{i=1}^{m}(h_{\theta_1}(x_i^1) - y_i)^2\right] = \frac{1}{m}\sum_{i=1}^{m}(h_{\theta_1}(x_i^1) - y_i)\, x_i^1 \ \ldots ecuación\ (10)$$

De la ecuación (9) y (10) resulta la ecuación (11):

$$\boldsymbol{\theta_1 \coloneqq \theta_1 - \frac{\alpha}{m}\sum_{i=1}^{m}\left(h_{\theta_1}(x_i^1) - y_i\right) x_i^1} \ldots ecuación\ (11)$$

Donde:

- α: $learning\ rate\ o\ tasa\ de\ aprendizaje.$
- m: $número\ total\ de\ datos.$
- h_{θ_0}: $hipótesis\ de\ la\ variable\ \theta_0.$
- h_{θ_1}: $hipótesis\ de\ la\ variable\ \theta_1.$

Por otra parte, la entropía cruzada (cross entropy en inglés) es una función de error que recibe x=[p_1,...,p_C], un vector de probabilidades de cada clase (cuya suma es 1), e c la etiqueta de la clase verdadera, es decir, la clase cuya probabilidad debería ser 1.Entonces, la función de entropía

cruzada (CrossEntropy) se define como el menos logaritmo de la probabilidad que generó el modelo para la clase verdadera:

$$= (y * \log(p) + (1 - y) * \log(1 - p) \dots ecuación\ (12)$$

Donde:

- y: $Producto\ real.$
- p: $probabilidad\ predicha\ por\ la\ regresión\ logistica.$

Learning rate

En el aprendizaje automático y las estadísticas, la tasa de aprendizaje o learning rate (lr) es un parámetro de ajuste en un algoritmo de optimización que determina el tamaño del paso en cada iteración mientras avanza hacia un mínimo de una función de pérdida. Dado que influye en qué medida la información recién adquirida anula la información anterior, representa metafóricamente la velocidad a la que "aprende" un modelo de aprendizaje automático.

Regla 60-20-20

Este método consiste en la creación de subconjuntos, en los cuales el 60% de los datos de considera para el entrenamiento, mientras que un 20% es para la validación y otro 20% corresponde a pruebas del modelo de clasificación.

Métricas

Exactitud

La exactitud es la relación entre los simples correctamente clasificados y el número total de simples en el conjunto de datos de evaluación. Esta métrica es conocida por ser engañosa en el caso de diferentes proporciones de clases, ya que asignar simplemente todos los simples a la clase predominante es una forma fácil de lograr una alta precisión [2].

$$ACC = \frac{\#\ correctly\ classified\ samples}{\#\ all\ samples} = \frac{TP+TN}{TP+FP+TN+FN} \qquad ecuación(13)$$

Materiales y métodos

Herramientas utilizadas

- Google Collaboratory
- Conjunto de datos: *Indicadores personales clave de enfermedad cardíaca*
- AMD Ryzen 9 5900HS with Radeon Graphics 3.30 GHz
- RAM 16.0 GB
- 64-bit operating system, x64-based processor

Conjunto de datos

En este trabajo, se utilizará el conjunto de datos: *Indicadores personales clave de enfermedad cardíaca*, datos provenientes de 400,000 adultos, obtenidos durante la encuesta anual 2020 de los Centros para el Control y prevención de Enfermedades (CDC, por sus siglas en inglés) pertenecientes al departamento de salud y servicios humanos en los Estados Unidos. Originalmente el conjunto de datos contenía alrededor de 300 atributos, sin embargo, se redujo a solo 18 variables, los cuales son los que se encuentran disponibles públicamente en la plataforma Kaggle [7].

Casi la mitad de los estadounidenses (47%), incluyendo afroamericanos, indios americanos, nativos de Alaska y blancos; tienen al menos de 1 a 3 factores de riesgo de padecer alguna enfermedad cardíaca. A continuación, se agrega una breve descripción de los atributos incluidos en este conjunto de datos:

- HeartDisease: (atributo de decisión): personas encuestadas que informaron alguna vez haber padecido alguna enfermedad coronaria (CHD, por sus siglas en inglés) o infarto al miocardio(IM, por sus siglas en inglés).
- BMI: Índice de Masa Corporal.
- Smoking: personas encuestadas que han fumado al menos 100 cigarros en su vida entera.

- AlcoholDrinking: corresponde a hombres adultos que beben más de 14 tragos por semana y mujeres adultas que beben más de 7 tragos por semana.
- Stroke: responde a la pregunta: ¿alguna vez le dijeron o usted tuvo un derrame cerebral?
- PhysicalHealth: incluyendo enfermedades y lesiones físicas, responde a la pregunta: ¿durante cuántos días en los últimos 30 días su salud física no fue buena? (de 0 a 30 días)
- MentalHealth: ¿durante cuántos días en los últimos 30 días su salud mental no fue buena? (de 0 a 30 días).
- DiffWalking: responde a ¿tiene serias dificultades para caminar o subir escaleras?
- Sex: hombre o mujer.
- AgeCategory: 14 rangos de edad.
- Race: valor de raza / etnicidad imputada.
- Diabetic: responde a ¿alguna vez ha sido diagnosticada con diabetes?
- PhysiclActivity: adultos que informaron haber realizado actividad física o ejercicio en los últimos 30 días, no incluyendo su trabajo habitual.
- GenHealth: responde a ¿cómo calificarías tu salud en general?
- SleepTime: responde a un promedio de horas que duerme, en un periodo de 24 horas, la persona encuestada.
- Asthma: responde a ¿alguna vez ha sido diagnosticado con asma?

Métodos

Para la realización de esta tarea, fue necesario importar las librerías de Pandas, numpy, etc., así como el montaje de Google Drive en el entorno de ejecución a fin de poder hacer uso de la base de datos propuesta.

Posteriormente, se optó por crear diversas funciones a fin de distribuir las tareas de una manera más eficiente como se muestra a continuación:

- Función method_602020: realiza la separación del conjunto de datos de acuerdo con el porcentaje de 60% para el conjunto de entrenamiento, 20% para el conjunto de validación

y 20% para el conjunto de prueba. Esta función requiere como entradas los datos y las etiquetas.

```
def method_602020(x,y):
  train_x = x[0 : int(len(x)*0.6)]
  train_y = y[0 : int(len(y)*0.6)]
  val_x = x[int(len(x)*0.6) : int(len(x)*0.8)]
  val_y = y[int(len(y)*0.6) : int(len(y)*0.8)]
  test_x = x[int(len(x)*0.8) : ]
  test_y = y[int(len(y)*0.8) : ]
  return train_x, train_y, val_x, val_y, test_x, test_y
```

Ilustración 3. Función propuesta para la división del conjunto de datos.

- Función regre_logistic: realiza la actualización del cálculo de los pesos de acuerdo con el número de épocas. Esta función requiere como entradas los valores de x, y, theta, alpha y épocas.

```
def regre_logist(x, y, theta, alpha, epocas,tolerance):
    iterations = 1
    for i in tqdm(range(epocas)):
        Z = np.dot(x,theta)
        ft = 1 / (1 + np.exp(-Z))
        dJ = (1/len(x)) * np.dot(np.transpose(x),(ft - y))
        mse.append(MSE(x, y, theta))
        vc.append(cross_entropy(x, y, theta))
        new_theta = theta - alpha * dJ
        #n_theta = [item for lista in new_theta for item in lista]
        it.append(iterations)
        # Stopping Condition
        if np.sum(abs(new_theta - theta)) < tolerance:
            print('Gradient Descent has converged')
            break
        iterations += 1
        theta = new_theta
    return theta, it,mse,vc
```

Ilustración 4. Función propuesta para el modelo de regresión logística.

- Función clasificacion: se enfoca en la obtención de la predicción de los valores de y. Esta función requiere de como entradas los valores de x y los pesos obtenidos de la etapa de entrenamiento.

```
def clasificacion(x,theta):
    Z = np.dot(x,theta)
    ft = 1 / (1 + np.exp(-Z))
    clas = np.round(ft)
    return clas
```

Ilustración 5. Función propuesta para la obtención de la clasificación.

- Función MSE: se enfoca calcular la función de costo. Esta función requiere de como entradas los valores de y, x y thetas.

```
def MSE(x, y, theta):
  h = np.dot(x,theta)
  J = np.sum((h-y)**2)/(2 * (len(h)))
  return J
```

Ilustración 6. Función propuesta para la función de costo.

- Función cross_entropy: se enfoca calcular la función de costo. Esta función requiere de como entradas los valores de y, x y thetas.

```
def cross_entropy(x, y, theta):
    z = np.dot(x,theta)
    p = 1 / (1 + np.exp(-z))
    J = - np.mean(y*np.log(p) + (1-y)*np.log(1-p)) #Cross-entropy cost function
    return J
```

Ilustración 7. Función propuesta para la función de costo.

- Función accuracy: se enfoca en obtener la métrica de exactitud a fin de poder analizar el comportamiento del modelo. Esta función requiere como entradas los datos originales de y y los valores predichos.

```
def accuracy(ycalculada, yreal):
  coincidencias = np.equal(ycalculada, yreal)
  totalcoin = np.sum(coincidencias)
  porcentaje = (totalcoin/len(ycalculada))*100
  return porcentaje
```

Ilustración 3. Función propuesta para la obtención de la métrica de exactitud.

Diagrama de metodología

Para el análisis de cualquier base de datos se deben seguir los siguientes pasos:

1. Recolección: Definir y cargar la base de datos a utilizar.
2. Preparar: Comprobar la no existencia de datos faltantes, identificación de outliers, así como la normalización de los datos.
3. Analizar: Conocer la distribución de la información por cada atributo, reducción de dimensionalidad, división de los conjuntos de entrenamiento y prueba del modelo, asignando el 60% de ellos para el entrenamiento, 20% para la validación y el 20% restante a la etapa de prueba.
4. Etapa de entrenamiento y prueba: Implementación del modelo de regresión logística. Así bien, se calcula la métrica de exactitud para conocer el desempeño obtenido.

A continuación, se muestra el diagrama de flujo que representa el proceso de desarrollo, el cual fue implementado en un programa basado en lenguaje Python.

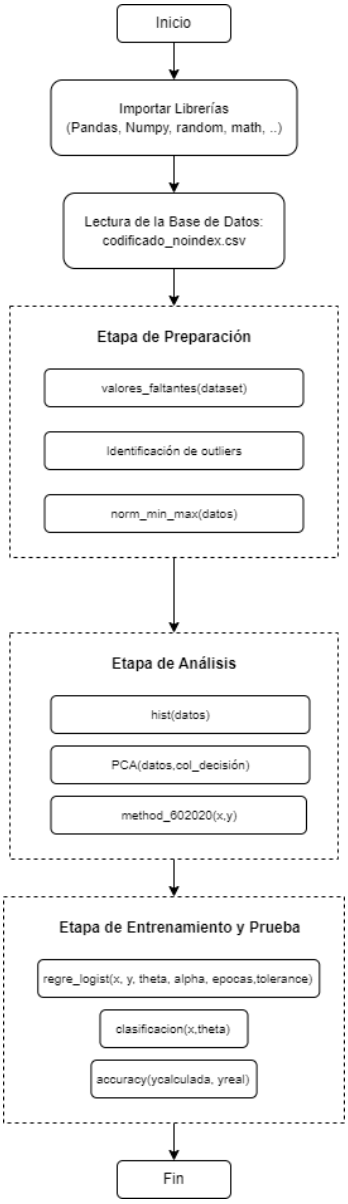
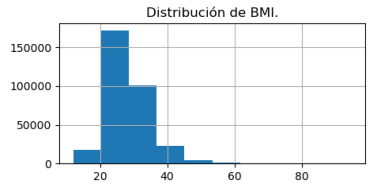
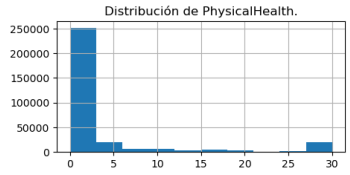


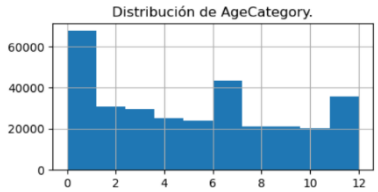
Ilustración 9. Diagrama de flujo para el análisis del conjunto de datos.

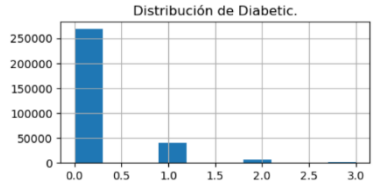
Resultados y discusión

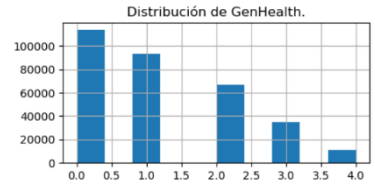
A continuación, se presentan las distribuciones de los atributos de la base de datos en cuestión, al cual se le efectuó una preparación, la cual comprende los siguientes puntos: la comprobación de la no existencia de datos faltantes, la identificación de outliers y la normalización de los datos.

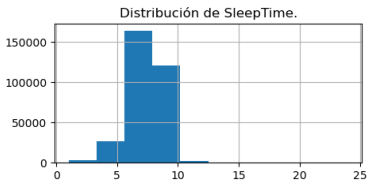


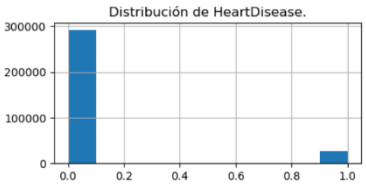


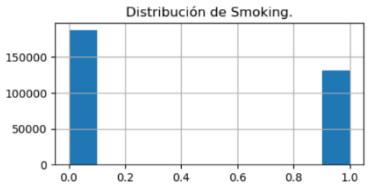


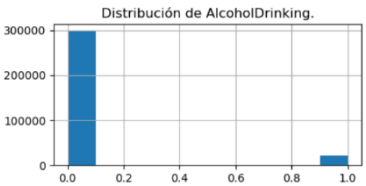


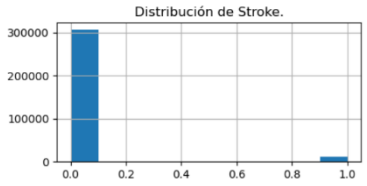


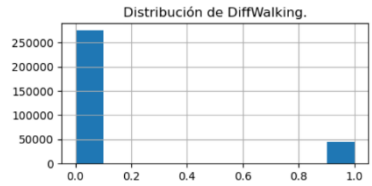


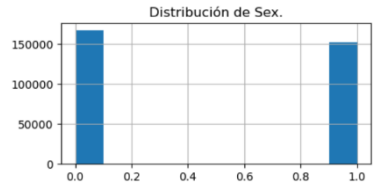


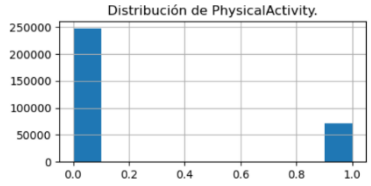












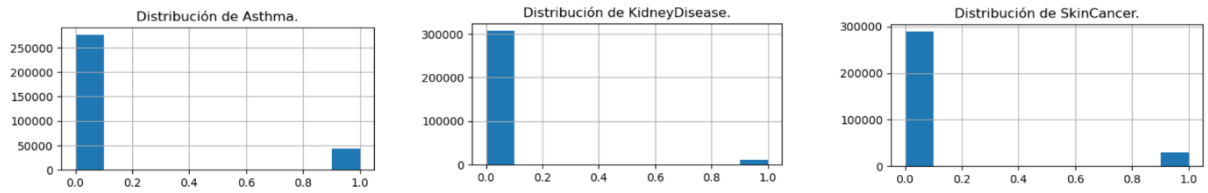


Ilustración 10. Distribución de los atributos de la base de datos.

A partir de haber implementado la técnica de análisis de componentes principales (PCA) en la base de datos en cuestión, se logró reducir la cantidad de atributos de 18 a 12, entre los atributos que contenían la mayor información se encuentran: BMI, PhysicalHealth, MentalHealth, AgeCategory, Race, Diabetic, Smoking, AlcoholDrinking, Stroke, DiffWalking y PhysicalActivity. Así bien, para los siguientes pasos se incluye a estos atributos seleccionados el atributo de decisión HeartDisease.

Antes de proseguir con la implementación de algoritmos de agrupamiento será necesario dividir el conjunto de datos en un conjunto de entrenamiento y un conjunto de prueba. En la Tabla 1 se presentan el total de los datos para cada uno de los conjuntos, en donde se utilizó la división 60/20/20.

Tabla 1. Conjunto de Entrenamiento, Validación y Prueba.

	Conjunto de Entrenamiento	Conjunto de Validación	Conjunto de Prueba
Total de datos	191877	63959	63959

Regresión Logística

A continuación, se pueden observar los parámetros propuestos para la implementación del modelo de Regresión Logística. Cabe mencionar que los valores iniciales de theta se obtuvieron de manera aleatoria. Así bien, se utilizaron únicamente los atributos con mayor relevancia en la base de datos: BMI y PhysicalHealth.

Épocas = 100, alpha= 0.6

En la Ilustración 11, se puede observar el tiempo que tardo el modelo en la etapa de entrenamiento. Así bien, en esta ocasión el modelo no convergió antes de la época 100.

```
100%|██████████| 100/100 [00:02<00:00, 43.90it/s]
```

Ilustración 11. Tiempo de entrenamiento del modelo regresión logística.

En la Ilustración 12, se puede visualizar como se va modificando el valor de las funciones de costo con respecto al número de iteración en cuestión.

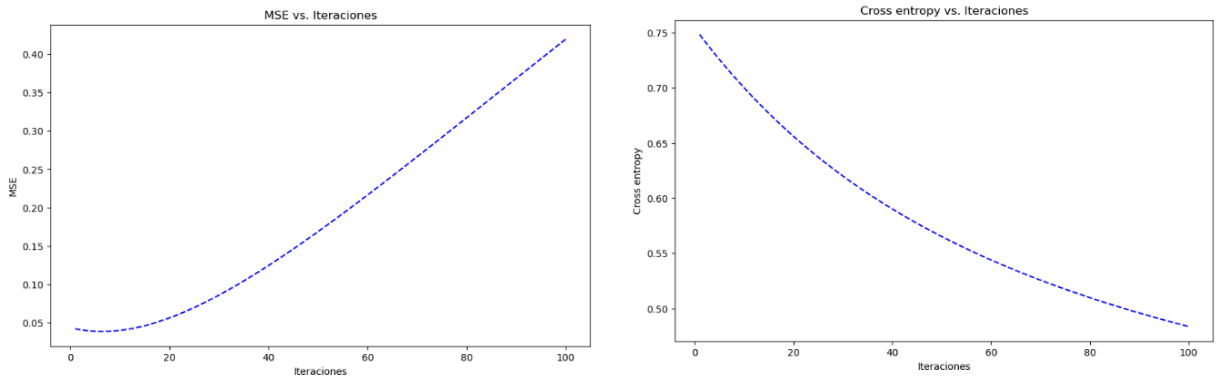


Ilustración 12.Funciones de costo vs Iteraciones.

Por otra parte, cabe mencionar que los valores obtenidos en la métrica de exactitud para los conjuntos de validación y prueba supera el 90 por ciento. Mientras que los valores obtenidos en las funciones de costo se encuentran muy cercanos al cero.

Tabla 2. Resultados obtenidos en la métrica de exactitud y las funciones de costo.

	Exactitud	MSE	Cross entropy
Conjunto de Validación	91.29285	0.30940	0.40846
Conjunto de Prueba	91.57585	0.31359	0.40699

Épocas = 100, alpha= 0.3

En la Ilustración 13, se puede observar el tiempo que tardo el modelo en la etapa de entrenamiento. Así bien, en esta ocasión el modelo no convergió antes de la época 100.

```
100%|██████████| 100/100 [00:02<00:00, 40.31it/s]
```

Ilustración 13. Tiempo de entrenamiento del modelo regresión logística.

En la Ilustración 14, se puede visualizar como se va modificando el valor de las funciones de costo con respecto al número de iteración en cuestión.

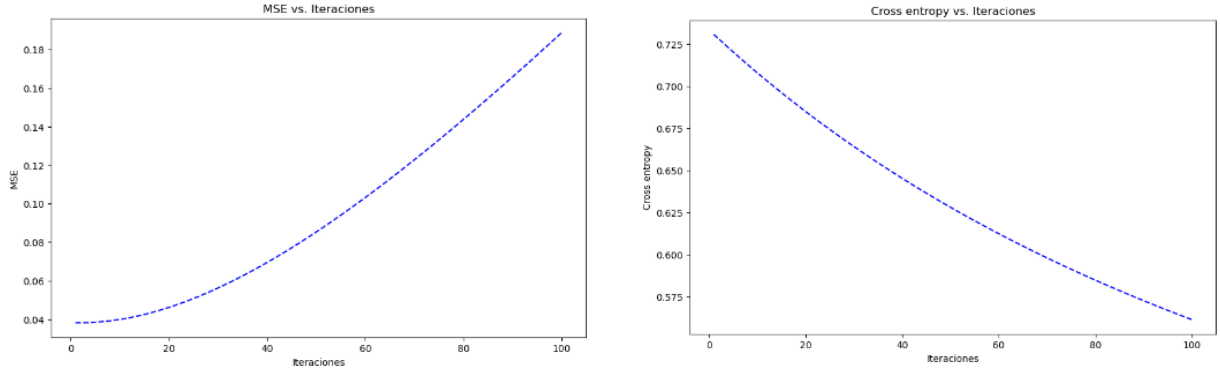


Ilustración 14. Funciones de costo vs Iteraciones.

Así mismo, los valores obtenidos en la métrica de exactitud para los conjuntos de validación y prueba supera el 90 por ciento. Mientras que los valores obtenidos en la función de costo se encuentran muy cercanos al cero.

Tabla 3. Resultados obtenidos en la métrica de exactitud y las funciones de costo.

	Exactitud	MSE	Cross entropy
Conjunto de Validación	91.29285	0.10506	0.51622
Conjunto de Prueba	91.57585	0.10693	0.51489

Épocas = 100, alpha= 0.01

En la Ilustración 15, se puede observar el tiempo que tardo el modelo en la etapa de entrenamiento. Así bien, en esta ocasión el modelo no convergió antes de la época 100.

```
100%|██████████| 100/100 [00:02<00:00, 43.16it/s]
```

Ilustración 15. Tiempo de entrenamiento del modelo regresión logística.

En la Ilustración 16, se puede visualizar como va modificando el valor de la función de costo con respecto al número de iteración en cuestión, en donde se puede observar que al incrementar el número de iteraciones el valor de la función de costo va decreciendo.

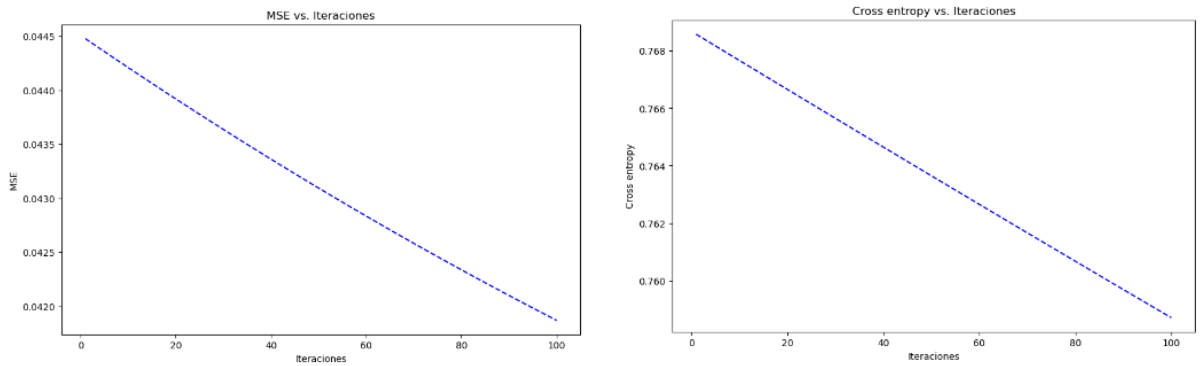


Ilustración 16. Funciones de costo vs Iteraciones.

Así mismo, los valores obtenidos en la métrica de exactitud para los conjuntos de validación y prueba no supera el 10 por ciento. Mientras que los valores obtenidos en las funciones de costo para los conjuntos de validación y prueba de datos se encuentran muy cercanos al cero.

Tabla 4. Resultados obtenidos en la métrica de exactitud y las funciones de costo.

	Exactitud	MSE	Cross entropy
Conjunto de Validación	8.70714	0.35839	0.61811
Conjunto de Prueba	8.42414	0.35714	0.61739

Épocas = 1000, alpha= 0.6

En la Ilustración 17, se puede observar el tiempo que tardo el modelo en la etapa de entrenamiento. Así bien, en esta ocasión el modelo no convergió antes de la época 1000.

```
100%|██████████| 1000/1000 [00:27<00:00, 36.61it/s]
```

Ilustración 17. Tiempo de entrenamiento del modelo regresión logística.

En la Ilustración 18, se puede visualizar como se va modificando los valores de las funciones de costo con respecto al número de iteración en cuestión, en donde se puede observar que al incrementar el número de iteraciones el valor de la función de costo MSE aumenta de igual manera, mientras que el valor de la función de costo Cross entropy disminuye.

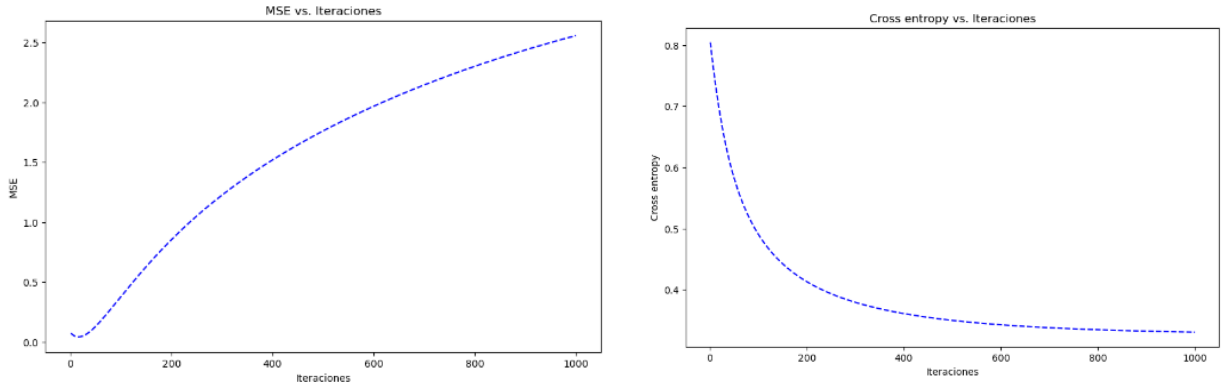


Ilustración 18. Funciones de costo vs Iteraciones.

Así mismo, los valores obtenidos en la métrica de exactitud para los conjuntos de validación y prueba supera el 90 por ciento. Mientras que los valores obtenidos en las funciones de costo para los conjuntos de validación y prueba se encuentran muy cercanos al cero.

Tabla 5. Resultados obtenidos en la métrica de exactitud y las funciones de costo.

	Exactitud	MSE	Cross entropy
Conjunto de Validación	91.20686	2.34594	0.15670
Conjunto de Prueba	91.49611	2.35146	0.15663

Épocas = 1000, alpha= 0.3

En la Ilustración 19, se puede observar el tiempo que tardo el modelo en la etapa de entrenamiento. Así bien, en esta ocasión el modelo no convergió antes de la época 1000.

```
100%|██████████| 1000/1000 [00:27<00:00, 36.06it/s]
```

Ilustración 19. Tiempo de entrenamiento del modelo regresión logística.

En la Ilustración 20, se puede visualizar como se va modificando los valores de las funciones de costo con respecto al número de iteración en cuestión, en donde se puede observar que al incrementar el número de iteraciones el valor de la función de costo MSE aumenta de igual manera, mientras que el valor de la función de costo Cross entropy disminuye.

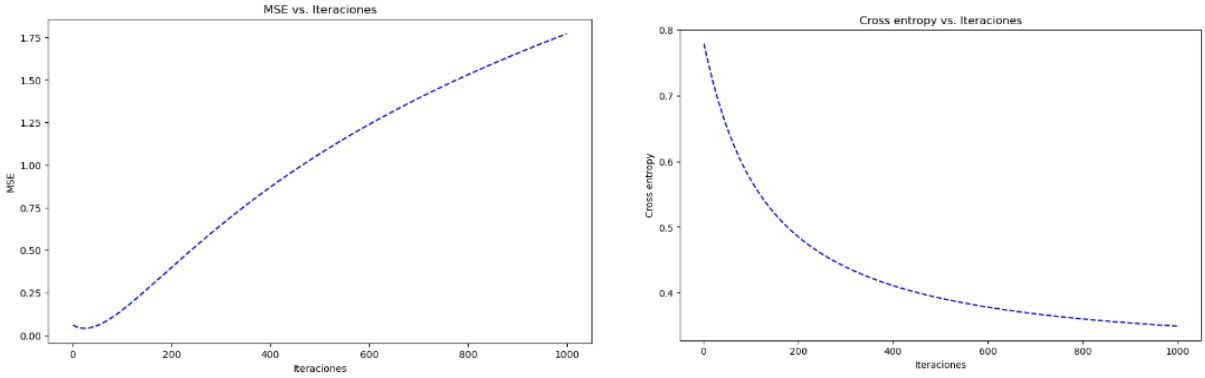


Ilustración 20. Funciones de costo vs Iteraciones.

Así mismo, los valores obtenidos en la métrica de exactitud para los conjuntos de validación y prueba supera el 90 por ciento. Mientras que los valores obtenidos en la función de costo de cross entropy se encuentran muy cercanos al cero. Así bien, los valores obtenidos en la función de costo MSE varían por milésimas.

Tabla 6. Resultados obtenidos en la métrica de exactitud y la función de costo.

	Exactitud	MSE	Cross entropy
Conjunto de Validación	91.29129	1.58132	0.19883
Conjunto de Prueba	91.57585	1.58905	0.19819

Épocas = 1000, alpha= 0.01

En la Ilustración 21, se puede observar el tiempo que tardo el modelo en la etapa de entrenamiento. Así bien, en esta ocasión el modelo no convergió antes de la época 1000.

```
100%|██████████| 1000/1000 [00:28<00:00, 35.29it/s]
```

Ilustración 21. Tiempo de entrenamiento del modelo regresión logística.

En la Ilustración 22, se puede visualizar como se va modificando los valores de las funciones de costo con respecto al número de iteración en cuestión, en donde se puede observar que al incrementar el número de iteraciones el valor de la función de costo MSE aumenta de igual manera posterior a la época 400. Por otra parte, los valores de la función de costo Cross entropy van disminuyendo con respecto al incremento de las iteraciones.

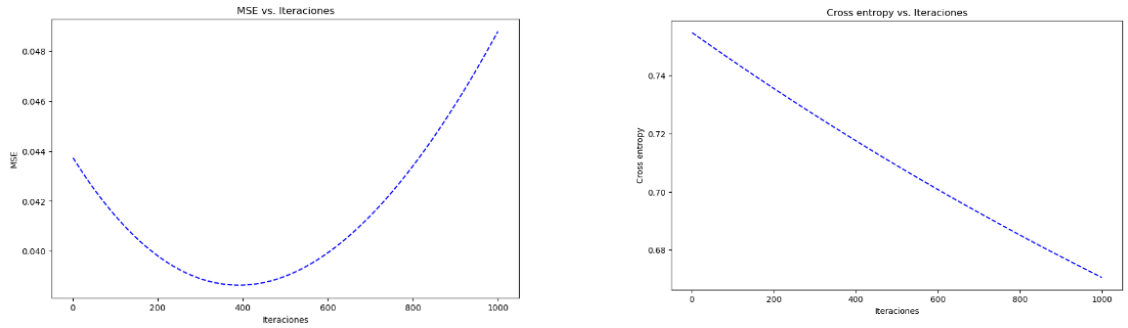


Ilustración 22. Funciones de costo vs Iteraciones.

Así mismo, los valores obtenidos en la métrica de exactitud para los conjuntos de validación y prueba no superan el 90 por ciento. Mientras que los valores obtenidos en la función de costo Cross entropy se encuentran muy cercanos al cero.

Tabla 7. Resultados obtenidos en la métrica de exactitud y las funciones de costo.

	Exactitud	MSE	Cross entropy
Conjunto de Validación	88.52389	0.02471	0.66536
Conjunto de Prueba	88.79438	0.02531	0.66541

Épocas = 10000, alpha= 0.6

En la Ilustración 23, se puede observar el tiempo que tardo el modelo en la etapa de entrenamiento. Así bien, en esta ocasión el modelo si convergió en la época 2751.

```
19%|██         | 1868/10000 [00:55<04:00, 33.80it/s]
Gradient Descent has converged
```

Ilustración 23. Tiempo de entrenamiento del modelo regresión logística.

En la Ilustración 24, se puede visualizar como se va modificando los valores de las funciones de costo con respecto al número de iteración en cuestión, en donde se puede observar que al incrementar el número de iteraciones el valor de la función de costo MSE aumenta de igual manera. Por otra parte, los valores de la función de costo Cross entropy van disminuyendo con respecto al incremento de las iteraciones.

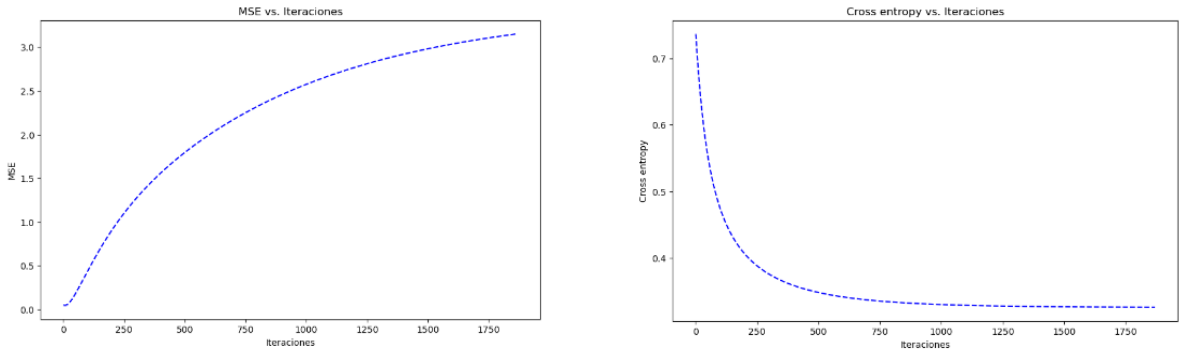


Ilustración 24. Funciones de costo vs Iteraciones.

Así mismo, los valores obtenidos en la métrica de exactitud para los conjuntos de validación y prueba superan el 90 por ciento. Mientras que los valores obtenidos en las funciones de costo Cross entropy se encuentran muy cercanos al cero.

Tabla 8. Resultados obtenidos en la métrica de exactitud y las funciones de costo.

	Exactitud	MSE	Cross entropy
Conjunto de Validación	91.04113	2.92641	0.13605
Conjunto de Prueba	91.35227	2.93068	0.13623

Épocas = 10000, alpha= 0.3

En la Ilustración 25, se puede observar el tiempo que tardo el modelo en la etapa de entrenamiento. Así bien, en esta ocasión el modelo si convergió en la época 2751.

```
28%|███        | 2751/10000 [01:15<03:19, 36.30it/s]
Gradient Descent has converged
```

Ilustración 25. Tiempo de entrenamiento del modelo regresión logística.

En la Ilustración 26, se puede visualizar como se va modificando los valores de las funciones de costo con respecto al número de iteración en cuestión, en donde se puede observar que al incrementar el número de iteraciones el valor de la función de costo MSE aumenta de igual manera. Por otra parte, los valores de la función de costo Cross entropy van disminuyendo con respecto al incremento de las iteraciones.

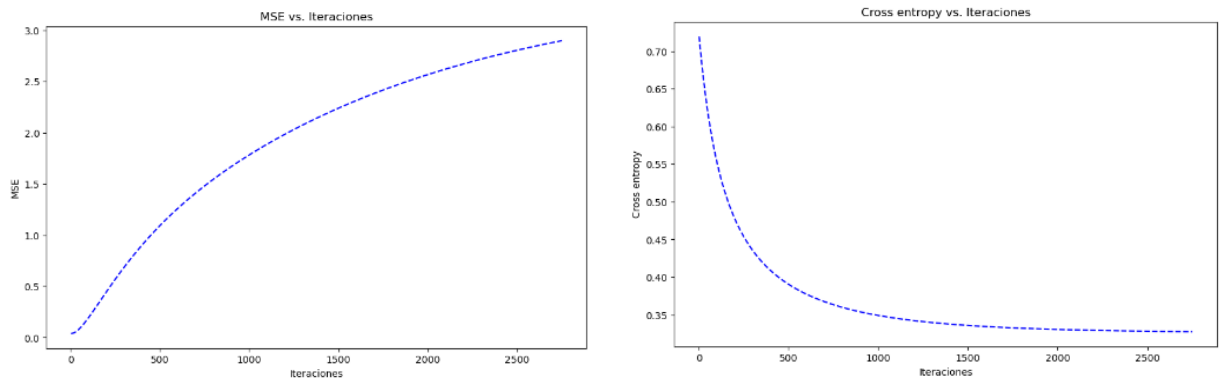


Ilustración 26. Funciones de costo vs Iteraciones.

Así mismo, los valores obtenidos en la métrica de exactitud para los conjuntos de validación y prueba superan el 90 por ciento. Mientras que los valores obtenidos en las funciones de costo Cross entropy se encuentran muy cercanos al cero.

Tabla 9. Resultados obtenidos en la métrica de exactitud y las funciones de costo.

	Exactitud	MSE	Cross entropy
Conjunto de Validación	91.11462	2.67755	0.14418
Conjunto de Prueba	91.42263	2.68221	0.14426

Épocas = 10000, alpha= 0.01

En la Ilustración 27, se puede observar el tiempo que tardo el modelo en la etapa de entrenamiento. Así bien, en esta ocasión el modelo si convergió en la época 1332.

```
13%|          | 1332/10000 [00:38<04:08, 34.83it/s]
Gradient Descent has converged
```

Ilustración 27. Tiempo de entrenamiento del modelo regresión logística.

En la Ilustración 28, se puede visualizar como se va modificando los valores de las funciones de costo con respecto al número de iteración en cuestión, en donde se puede observar que al incrementar el número de iteraciones el valor de la función de costo MSE aumenta de igual manera. Por otra parte, los valores de la función de costo Cross entropy van disminuyendo con respecto al incremento de las iteraciones.

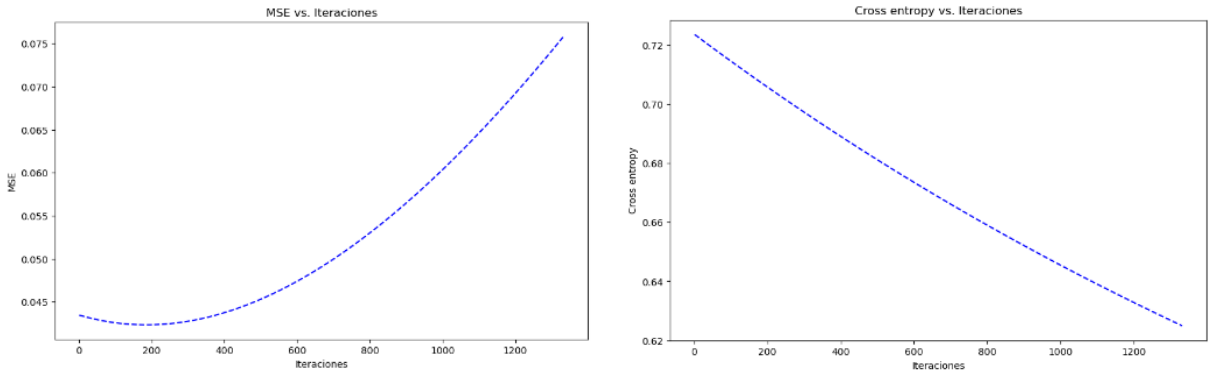


Ilustración 28. Funciones de costo vs Iteraciones.

Así mismo, los valores obtenidos en la métrica de exactitud para los conjuntos de validación y prueba superan el 90 por ciento. Mientras que los valores obtenidos en las funciones de costo se encuentran muy cercanos al cero.

Tabla 10. Resultados obtenidos en la métrica de exactitud y las funciones de costo.

	Exactitud	MSE	Cross entropy
Conjunto de Validación	91.11149	0.01957	0.60978
Conjunto de Prueba	91.42419	0.01954	0.60974

En las siguientes tablas, se puede realizar una comparación entre las diferentes pruebas realizadas de acuerdo con la variación en el número de épocas con respecto a cada learning rate propuesto, en donde se puede observar que el mejor porcentaje de exactitud se obtuvo en la época 100 tanto en el learning rate de 0.6 y 0.3 para los conjuntos de validación y prueba.

Tabla 11. Comparación de los resultados obtenidos con un learning rate = 0.6.

Épocas	Conjunto de Validación			Conjunto de Prueba		
	Exactitud	MSE	Cross entropy	Exactitud	MSE	Cross entropy
100	91.29285	0.30940	0.40846	91.57585	0.31359	0.40699
1000	91.20686	2.34594	0.15670	91.49611	2.35146	0.15663
10000	91.04113	2.92641	0.13605	91.35227	2.93068	0.13623

Tabla 12. Comparación de los resultados obtenidos con un learning rate = 0.3.

Épocas	Conjunto de Validación			Conjunto de Prueba		
	Exactitud	MSE	Cross entropy	Exactitud	MSE	Cross entropy
100	91.29285	0.10506	0.51622	91.57585	0.10693	0.51489
1000	91.29129	1.58132	0.19883	91.57585	1.58905	0.19819
10000	91.11462	2.67755	0.14418	91.42263	2.68221	0.14426

Mientras que el peor porcentaje de exactitud se obtuvo en la época 100 con un learning rate de 0.01 para los conjuntos de validación y prueba.

Tabla 13. Comparación de los resultados obtenidos con un learning rate = 0.01.

Épocas	Conjunto de Validación			Conjunto de Prueba		
	Exactitud	MSE	Cross entropy	Exactitud	MSE	Cross entropy
100	8.70714	0.35839	0.61811	8.42414	0.35714	0.61739
1000	88.52389	0.02471	0.66536	88.794383	0.02531	0.66541
10000	91.11149	0.01957	0.60978	91.42419	0.01954	0.60974

Conclusiones

En este programa generalizado inicialmente se generaron funciones que permitieran conocer el comportamiento de los datos mediante la regresión logística, la cual permite describir y estimar la relación entre una variable binaria dependiente y las variables independientes. Así bien, en comparación con otros algoritmos de machine learning, este no requiere de demasiados recursos computaciones, como se pudo comprobar anteriormente ya que no fue necesario la utilización de un subsampling del conjunto de datos, además el tiempo de ejecución fue considerablemente rápido si se compara con otros algoritmos.

Asimismo, la implementación del gradiente descendente permitió encontrar de forma automática el mínimo de la función de error MSE, mediante la determinación de parámetros theta para la función de regresión logística. A partir de las pruebas de variación de los parámetros: learning rate y el número de épocas de entrenamiento, se puede decir que un número de learning

rate cuando es demasiado grande provoca que los cambios en los hiperparámetros theta sean también muy grandes y resulte más difícil encontrar los coeficientes que minimicen la función de costo, esto mismo sucede cuando el valor del learning rate es demasiado pequeño. Para este conjunto de datos en específico, los mejores resultados se obtuvieron cuando el valor de learning rate era (0.6 y 0.3) y el número de épocas pequeño (100).

Así bien, como se pudo observar al visualizar las funciones de costo, es que la función de costo MSE no funciona con la regresión logística ya que cuando se grafica dicha función con respecto a los pesos del modelo, la curva que se obtiene no es convexa lo cual dificulta la obtención del mínimo global, debido a que se ha introducido la no linealidad en el modelo en forma de una función sigmoidea que hace que la relación entre los parámetros de peso y el error sea muy compleja. Por tal motivo, una opción viable para la función de costo es Cross entropy, la cual se utiliza para evitar el sobreajuste de los datos al determinar que variables deben conservarse.

Referencias

[1] "Acerca de la regresión lineal - México | IBM." [Online]. Available: https://www.ibm.com/mx-es/analytics/learn/linear-regression].

[2] "Regresión Lineal: teoría y ejemplos en Python - IArtificial.net." [Online]. Available: https://www.iartificial.net/regresion-lineal-con-ejemplos-en-python/.

[3] "Gradiente Descendiente para aprendizaje automático - IArtificial.net." [Online]. Available: https://www.iartificial.net/gradiente-descendiente-para-aprendizaje-automatico/.

[4] "Error Cuadrático Medio para Regresión - IArtificial.net." [Online]. Available: https://www.iartificial.net/error-cuadratico-medio-para-regresion/.

[5] Martinez, J. (2020, September 21). Regresión Logística para la Clasificación. IArtificial.net. 2022, from https://www.iartificial.net/regresion-logistica-para-clasificacion/

[6] M. Antonio and A. Fernández, *Inteligencia artificial para programadores con prisa by Marco Antonio Aceves Fernández - Books on Google Play*. Universo de Letras. [Online].

https://play.google.com/store/books/details/Inteligencia_artificial_para_programadores_con_pri?id=ieFYEAAAQBAJ&hl=en_US&gl=US

[7] “Personal Key Indicators of Heart Disease | Kaggle.” https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease/code
https://learn.microsoft.com/es-es/azure/machine-learning/component-reference/smote