Nombre. – Lopez Chavez Pablo

Materia. – Inteligencia Artificial 1

Regresión: Regresión logística (Binaria)

Links

https://colab.research.google.com/drive/1i76pcQNowi3_RyZsgLg3-KbSXP7elkOL?usp=sharing

Regresión Logistica

Tomamos un dataset Rice - Pest and Diseases, donde evaluaremos si hay o no plaga tomando las columnas 'MaxT', 'MinT', 'RH1(%)', 'RF(mm)' como valores de X y Pest Value como variable Y

Tratamiento de la informacion mediante pandas definimos las columnas que nos sirve y la variable y, convirtiendo los valores de la variable "y" a binario donde si presenta plaga es cuando "y>0 " pero si no presenta plaga "y<=0"

Visualizar el, Se llama a la función implementada para mostrar los datos cargados:

función sigmoide (también conocida como función logística) calcula la función sigmoide para la entrada z.La función sigmoide se define como: sigmoid(z) = 1 / (1 + exp(-z))Parámetros:z : escalar, array unidimensional o multidimensional de numpy El valor o array sobre el cual se calculará la función sigmoide.Retorna:Un escalar o array numpy de la misma forma que z, con la función sigmoide aplicada.

Configurar la matriz adecuadamente, y agregar una columna de unos que corresponde al termino de intercepción. Agraga el termino de intercepción a X_entranamiento

La funcion costo o función de pérdida en un modelo de regresión logística La hipótesis h se calcula aplicando la función sigmoide a la multiplicación de la matriz de características X_entrenamiento y el vector de parámetros theta. La función sigmoide se utiliza para obtener probabilidades en la regresión logística.El costo J se calcula utilizando la fórmula de la función de pérdida de la regresión logística. Esta fórmula se basa en la entropía cruzada entre las predicciones h y las etiquetas reales Y_entrenamiento

implementa el algoritmo de descenso por el gradiente para optimizar los parámetros theta, donde m es el numero de ejemplos, luego en el for se repite las heces de interaciones, donde aplicando la función sigmoide a la multiplicación de X_entrenamiento por theta.

Aquí theta.T es la transposición de theta, asegurando que las dimensiones Se calcula y almacena el costo actual en J_history usando la función calcularCosto

Elegimos numero de iteraciones y porcentaje de aprendisaje, damos un tamaño de tetha con la canyidad de columnas de X_entramiento, llamamos a la funcion decensoGradiente, lo mostramos, damos el sesgo a X_Prueba luego se "Y_preb_prueba" realiza la predicción de las etiquetas para un conjunto de prueba en un modelo de regresión logística, luego se realiza el cálculo del porcentaje de aciertos del modelo en el conjunto de prueba y muestra el resultado en la consola

Luego de graficar

está diseñada para calcular tanto el costo como el gradiente en un modelo de regresión logística, m es numero de entranamiento, Cálculo de la Predicción (h), luego Evitar Valores Extremadamente Cercanos a 0 o 1, Cálculo del Costo (J), Cálculo del Gradiente (grad)

Iniciciamos tetha con la cantidad de columnas de X_prueba y llamamos a la funcion costFunction con los nuevos valores

define un diccionario de opciones para el algoritmo de optimización. En este caso, 'maxiter': 1000 establece el número máximo de iteraciones, optimize.minimize busca encontrar el valor óptimo de theta que minimiza la función de costo:

1. costFunction: La función de costo que se quiere minimizar, que debe devolver tanto el costo como el gradiente.
2. initial_theta: El punto de partida para el optimizador. Aquí, initial_theta es el valor inicial de los parámetros theta.
3. (X_prueba, Y_prueba): Los datos de entrenamiento que se pasan a la función de costo. La tupla contiene la matriz de características y el vector de etiquetas.
4. jac=True: Indica que la función de costo devuelve también el gradiente (derivadas) junto con el costo. Esto permite que el optimizador utilice el gradiente para mejorar la eficiencia.
5. method='TNC': El algoritmo de optimización a utilizar. 'TNC' es el algoritmo de Newton Truncated, que es un método de optimización que usa la información de segunda orden (aproximación de la matriz Hessiana) para encontrar el mínimo.
6. options=options: Las opciones para el optimizador, como el número máximo de iteraciones.

Obtener el Costo Optimizado
Obtener los Parámetros Optimizados
Imprimir los Resultados

damos numero de ejemplos de entranamiento, creamos una varibale con la cantidad de columnas X.

1. Calcula la predicción lineal (logit) para cada ejemplo en X_prueba.
2. Aplica la función sigmoide a estos logits para obtener las probabilidades de pertenecer a la clase positiva.
3. Redondea las probabilidades a 0 o 1 para obtener las predicciones finales de clase.

Una vez entrenado el modelo se procede a realizar la prediccion y evaluación de los resultados de predecir cual es el valor que vota el modelo para todos los datos utilizados en el entrenamiento.