Centro de Enseñanza Técnica Industrial

Plantel Colomos

Ingeniería en Desarrollo de Software

Inteligencia Artificial

REPORTE DE PRÁCTICA

IDENTIFICACIÓN DE LA PRÁCTICA

Práctica	**1**	**Nombre de la práctica**	**Regresión lineal univariable**
Fecha	**23/02/2025**	**Nombre del profesor**	**Alma Nayeli Rodríguez Vázquez**
Nombre del estudiante		**Jesús Alberto Aréchiga Carrillo**	

OBJETIVO

El objetivo de esta práctica consiste en implementar el método de regresión lineal para predicción.

PROCEDIMIENTO

Realiza la implementación siguiendo estas instrucciones.

Implementa el método de regresión lineal en Python y con la paquetería de sklearn. Para ello, considera los siguientes requerimientos:

- Utiliza el set de datos del archivo “dataset_RegresionLineal.csv”.
- No normalizar los datos
- Utiliza los siguientes valores para los parámetros iniciales:
  `a0=0 a1=0 beta=0.024 iteraciones=10000`
- Reporta el error `J` y el valor final de `a0` y `a1`. Además, reporta el valor de `h` para el dato de prueba `x= 9.7687`, cuya salida correcta es `y= 7.5435`.
- Comprueba tus resultados con los siguientes:
  `J=4.4769 a0=-3.8957 a1=1.1930`
  `Dato de prueba x=9.7687. Salida correcta y= 7.5435. Predicción h=7.7586`

IMPLEMENTACIÓN

Agrega el código de tu implementación en Python aquí.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

#Cargar datos
data = pd.read_csv('dataset_RegresionLineal.csv')
data.head()

x = np.array(data['x'])
y = np.array(data['y'])
m = np.size(x)

#Graficar datos
def graficarDatos():
  plt.plot(x, y, 'o', color='w', mec='black')
```

```
  plt.title('Grafica de dispersion')
  plt.xlabel('x: area m^2')
  plt.ylabel('y: precio $')

graficarDatos()
plt.show()

#Parametros Iniciales
a0 = 0
a1 = 0
beta = 0.024
iteracionesMax = 10000
iteracion = 0

#Calcular hipotesis inicial
h = a0 + a1 * x

graficarDatos()

plt.plot(x, h, color='r')
plt.show()

#Entrenamiento
convergencia = np.zeros(iteracionesMax)
while iteracion < iteracionesMax:
  a0 = a0 - beta * (1/m) * np.sum(h - y)
  a1 = a1 - beta * (1/m) * np.sum((h - y) * x)
  h = a0 + a1 * x
  J = (1 / 2 * m) * np.sum(np.square(h - y))
  convergencia[iteracion] = J
  iteracion += 1

# Graficar convergencia
plt.plot(convergencia)
plt.title('Grafica de Convergencia')
plt.xlabel('Iteraciones')
plt.ylabel('Costo')
plt.show()

# Modelo final

#Hipotesis final
h = a0 + a1 * x

#Graficar datos finales o modelo final
graficarDatos()

plt.plot(x, h, color='r')
plt.show()

# Resultados
print('Error final: ', J)
print('a0: ', a0)
```

```
print('a1: ', a1)

# Prediccion
casaNueva = 9.7687
precio = a0 + a1 * casaNueva
print('Precio de la casa nueva: ', precio)

# Graficar precio de la casa nueva
graficarDatos()
plt.plot(casaNueva, precio, 'o', color = 'black')
plt.show()
```

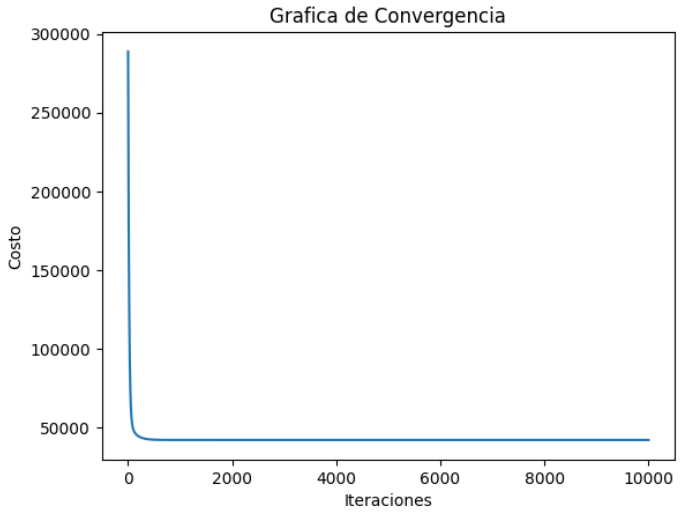
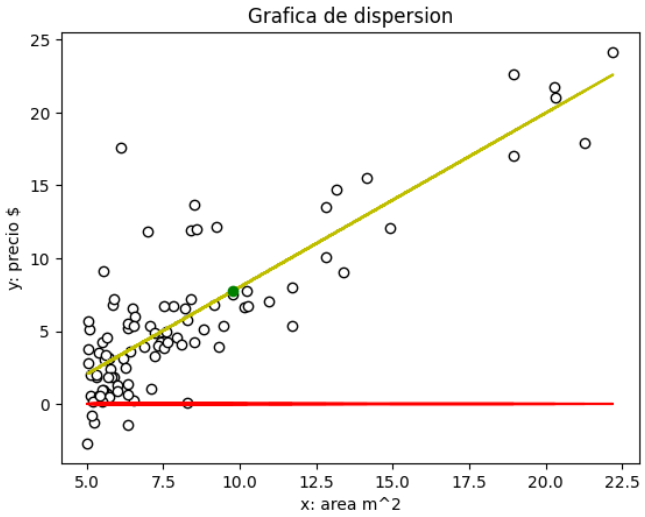
Agrega el código de tu implementación en Python con sklearn aquí.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import linear_model

data = pd.read_csv('dataset_RegresionLineal.csv')

x = np.array(data['x'])
y = np.array(data['y'])

plt.plot(x, y, 'o', mec = 'black')
plt.xlabel('x')
plt.ylabel('y')

model = linear_model.LinearRegression()
x = x.reshape(-1, 1)
model.fit(x, y)
h = model.predict(x)
plt.plot(x, h, 'r')

datoPrueba = 9.7687
h_datoPrueba = model.predict([[datoPrueba]])

plt.plot(datoPrueba, h_datoPrueba, 'o', color = 'black')

print('a0: ', model.intercept_, 'a1: ', model.coef_[0])
print('Dato de prueba: ', datoPrueba, 'Prediccion: ', h_datoPrueba)
```

Inteligencia Artificial

RESULTADOS EN PYTHON

Agrega las imágenes con los resultados obtenidos en los espacios indicados.

Gráfica de convergencia	Gráfica del resultado final donde se aprecian los datos de entrenamiento, la recta del modelo inicial en rojo, las rectas del entrenamiento en amarillo y la final en verde)
	

Impresión de los valores de `J`, `a0`, `a1`, el dato de prueba `x` con la salida correcta `y` y su predicción `h`

```
Error final:  42123.823676550455
a0:  -3.895780878311822
a1:  1.1930336441895901
```

```
Precio de la casa nueva:  7.758606881683029
-3.895780878311822
```

Inteligencia Artificial

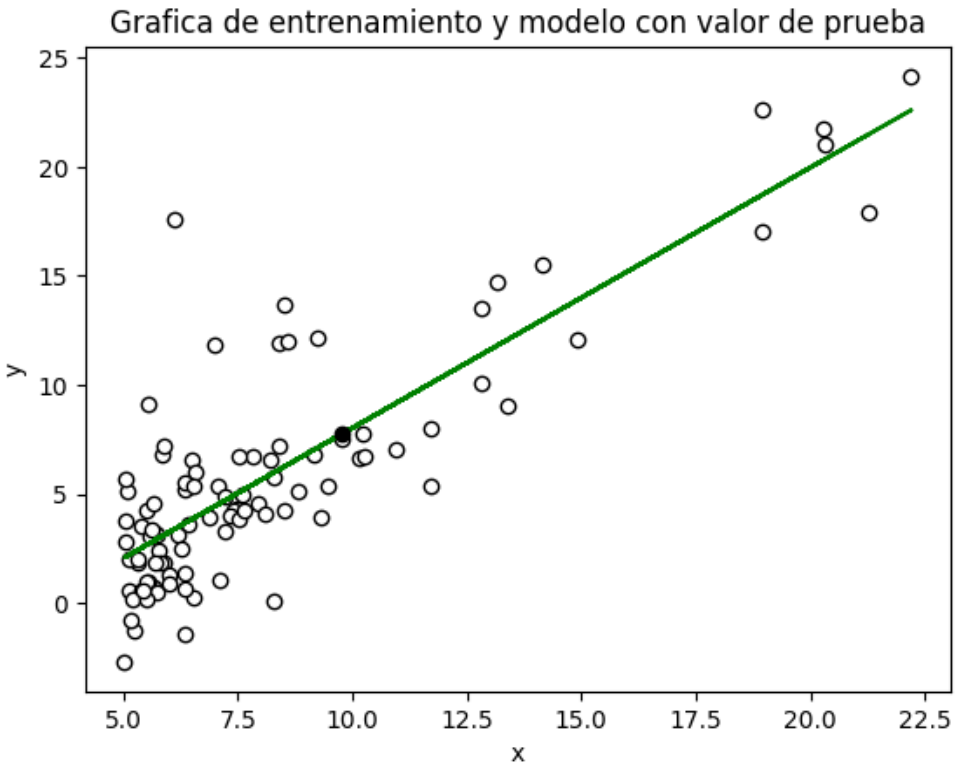
RESULTADOS EN PYTHON CON SKLEARN

Agrega las imágenes con los resultados obtenidos en los espacios indicados.

Impresión de los valores de `a0`, `a1`, el dato de prueba `x` con la salida correcta `y` y su predicción `h`

```
a0:  -3.8957808783118484 a1:  1.1930336441895932
Dato de prueba:  9.7687 Prediccion:  [7.75860688]
```

Gráfica del resultado final donde se aprecian los datos de entrenamiento y la recta del modelo final en verde)



CONCLUSIONES

Escribe tus observaciones y conclusiones.

El entrenamiento son sklearn es mucho más sencillo y rápido que implementarlo desde cero con Python. Gracias a esto se pueden conseguir datos mucho más rápido para poder entrenar y sacar un modelo listo. En el caso de la implementación con Python, se tiene que ver el error que se genera al momento de entrenar porque se tiene que revisar el estado del entrenamiento para tener un buen modelo, con sklearn no es necesario tener eso en consideración, sólo es dar los valores de entrenamiento y obtener los resultados.