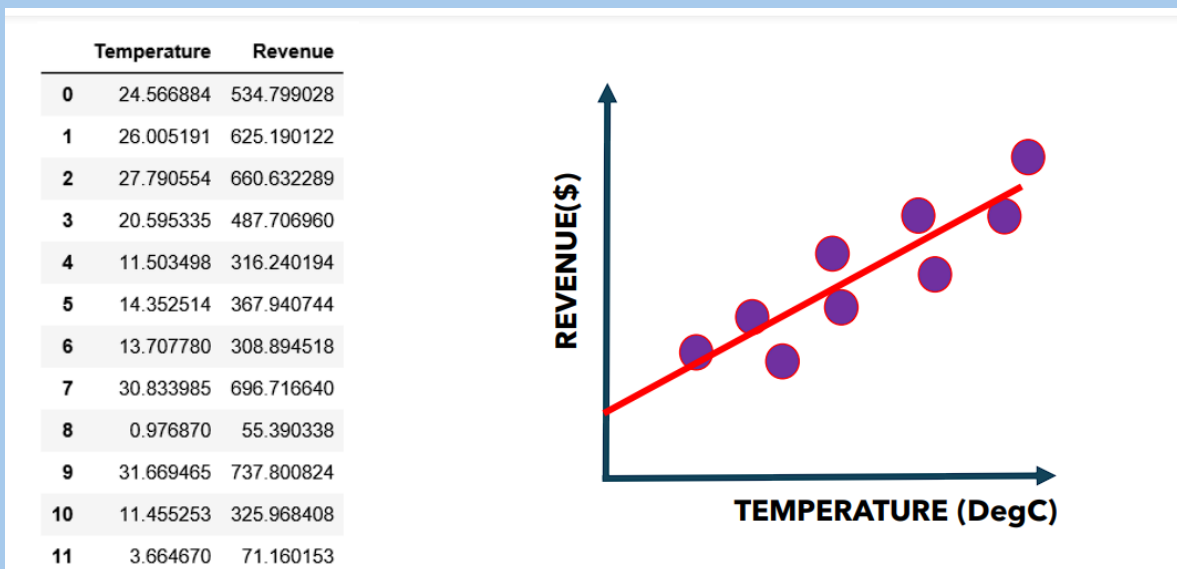
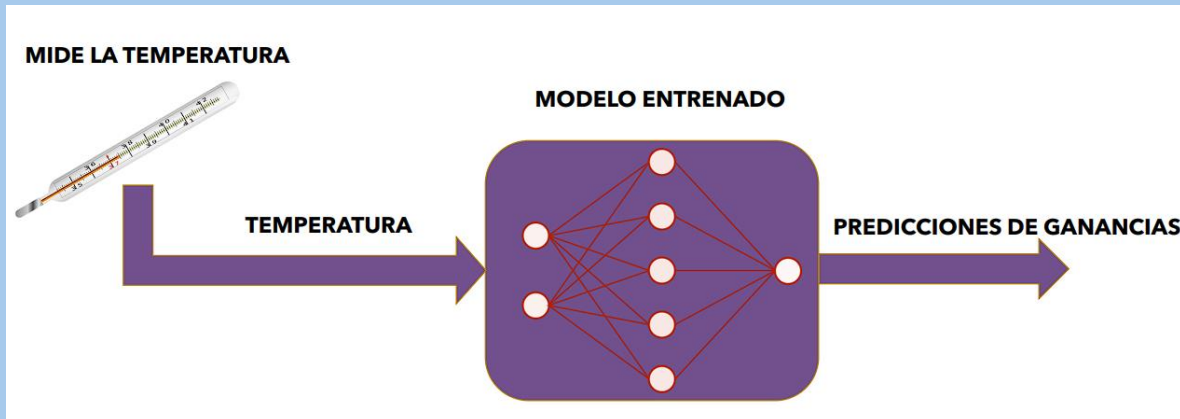


PREDICCIONES DE INGRESOS

Proyecto 3:

Eres dueño de un negocio de helados y quieres crear un modelo para predecir los ingresos diarios en dólares basados en la temperatura (degC). • Decidiste construir una simple Red Neural Artificial para resolver este problema. • Conjunto de datos: • Entrada (X): Temperatura del aire exterior • Salida (Y): Ingresos diarios totales generados en dólares

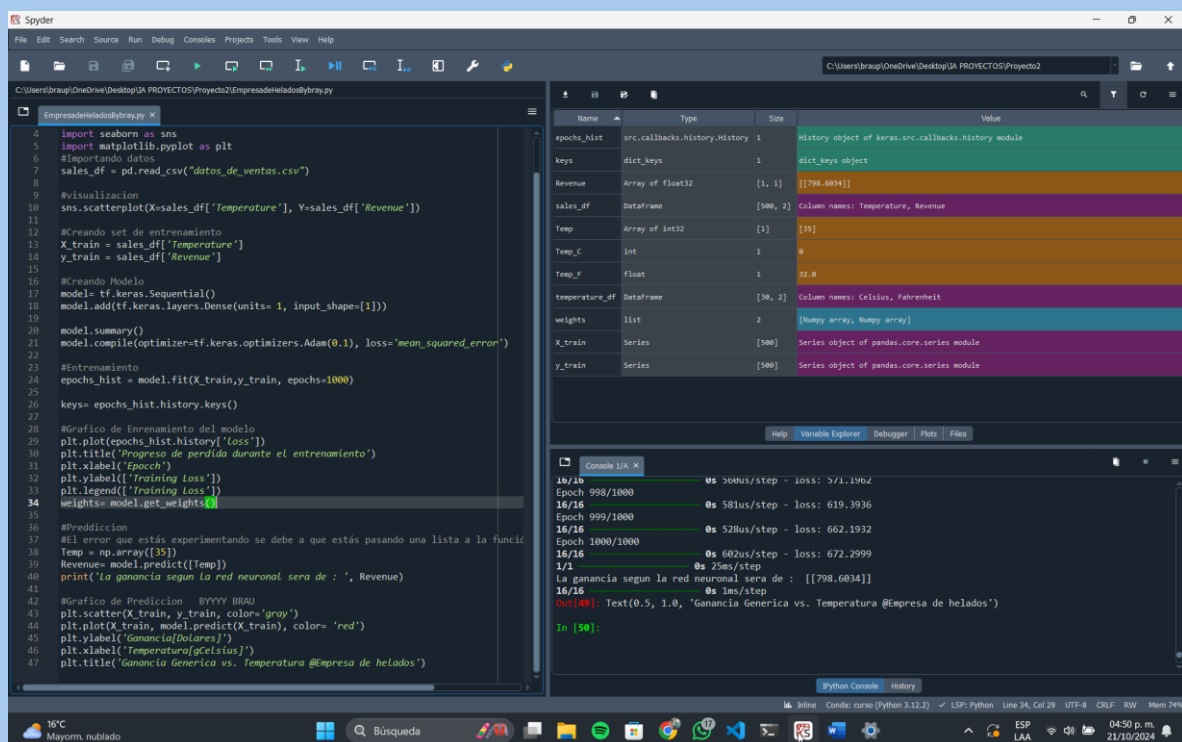


PREDICCIONES DE INGRESOS

Este código entrena un modelo de red neuronal simple para predecir las ganancias de una empresa de helados basadas en la temperatura. Comienza cargando un conjunto de datos que contiene la temperatura y las ganancias correspondientes. Luego, visualiza estos datos con un gráfico para entender su relación.

A continuación, crea un modelo de regresión lineal simple utilizando la biblioteca TensorFlow, con una sola neurona que toma la temperatura como entrada y produce las ganancias como salida. El modelo se entrena ajustando los pesos para minimizar el error, usando un optimizador (Adam) y una función de pérdida (error cuadrático medio). El proceso de entrenamiento se ejecuta durante 1000 épocas, y el historial de pérdida se visualiza en un gráfico, mostrando cómo mejora el modelo con el tiempo.

Después del entrenamiento, se utiliza el modelo para predecir las ganancias para una temperatura específica, y se visualizan tanto los datos reales como la predicción del modelo en un gráfico final. Este enfoque ayuda a entender cómo la temperatura influye en las ganancias y cómo el modelo ajusta esta relación para hacer predicciones.



```
4 import seaborn as sns
5 import matplotlib.pyplot as plt
6 #Importando datos
7 sales_df = pd.read_csv("datos_de_ventas.csv")
8
9 #visualizacion
10 sns.scatterplot(X=sales_df['Temperature'], Y=sales_df['Revenue'])
11
12 #Creando set de entrenamiento
13 X_train = sales_df['Temperature']
14 y_train = sales_df['Revenue']
15
16 #Creando Modelo
17 model = tf.keras.Sequential()
18 model.add(tf.keras.layers.Dense(units= 1, input_shape=[1]))
19
20 model.summary()
21 model.compile(optimizer=tf.keras.optimizers.Adam(0.1), loss='mean_squared_error')
22
23 #Entrenamiento
24 epochs_hist = model.fit(X_train,y_train, epochs=1000)
25
26 keys= epochs_hist.history.keys()
27
28 #Grafico de Entrenamiento del modelo
29 plt.plot(epochs_hist.history['loss'])
30 plt.title('Progreso de perdida durante el entrenamiento')
31 plt.xlabel('epoch')
32 plt.ylabel(['Training Loss'])
33 plt.legend(['Training Loss'])
34 weights= model.get_weights()
35
36 #Prediccion
37 #El error que estás experimentando se debe a que estás pasando una lista a la función
38 Temp = np.array([39])
39 Revenue= model.predict([Temp])
40 print('La ganancia segun la red neuronal sera de : ', Revenue)
41
42 #Grafico de Prediccion BYPPY BRAU
43 plt.scatter(X_train, y_train, color='gray')
44 plt.plot(X_train, model.predict(X_train), color= 'red')
45 plt.ylabel('Ganancia[Dolares]')
46 plt.xlabel('Temperatura[celsius]')
47 plt.title('Ganancia Generica vs. Temperatura @Empresa de helados')
```

Name	Type	Size	Value
epochs_hist	src.callbacks.history.History	1	History object of keras.src.callbacks.history module
keys	dict_keys	1	dict_keys object
Revenue	Array of float32	[1, 1]	[[798.6034]]
sales_df	Dataframe	[500, 2]	Column names: Temperature, Revenue
Temp	Array of int32	[1]	[39]
Temp_C	int	1	0
Temp_F	float	1	32.0
temperature_df	Dataframe	[30, 2]	Column names: Celsius, Fahrenheit
weights	list	2	[Numpy array, Numpy array]
X_train	Series	[500]	Series object of pandas.core.series module
y_train	Series	[500]	Series object of pandas.core.series module

```
16/16 0s 300us/step - loss: 571.1962
Epoch 998/1000
16/16 0s 581us/step - loss: 619.3936
Epoch 999/1000
16/16 0s 528us/step - loss: 662.1932
Epoch 1000/1000
16/16 0s 682us/step - loss: 672.2999
1/1 0s 25us/step
La ganancia segun la red neuronal sera de : [[798.6034]]
16/16 0s 1ms/step
Out[49]: Text(0.5, 1.0, 'Ganancia Generica vs. Temperatura @Empresa de helados')
In [50]:
```

Referencia : <https://www.youtube.com/watch?v=TWk0YtF7pDM&t=2252s>

PREDICCIONES DE INGRESOS

```
import tensorflow as tf
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
#Importando datos
sales_df = pd.read_csv("datos_de_ventas.csv")

#visualizacion
sns.scatterplot(X=sales_df['Temperature'], Y=sales_df['Revenue'])

#Creando set de entrenamiento
X_train = sales_df['Temperature']
y_train = sales_df['Revenue']

#Creando Modelo
model= tf.keras.Sequential()
model.add(tf.keras.layers.Dense(units= 1, input_shape=[1]))

model.summary()
model.compile(optimizer=tf.keras.optimizers.Adam(0.1), loss='mean_squared_error')

#Entrenamiento
epochs_hist = model.fit(X_train,y_train, epochs=1000)

keys= epochs_hist.history.keys()

#Grafico de Enrenamiento del modelo
plt.plot(epochs_hist.history['loss'])
plt.title('Progreso de pérdida durante el entrenamiento')
plt.xlabel('Epoch')
plt.ylabel(['Training Loss'])
plt.legend(['Training Loss'])
weights= model.get_weights()

#Preddiccion
#El error que estás experimentando se debe a que estás pasando una lista a la función model.predict()
Temp = np.array([35])
Revenue= model.predict([Temp])
print('La ganancia segun la red neuronal sera de : ', Revenue)

#Grafico de Prediccion   BYYYY BRAU
plt.scatter(X_train, y_train, color='gray')
plt.plot(X_train, model.predict(X_train), color= 'red')
plt.ylabel('Ganancia[Dolares]')
plt.xlabel('Temperatura[gCelsius]')
plt.title('Ganancia Generica vs. Temperatura @Empresa de helados')
```