PRACTICA 2 – UNIDAD III Y IV

Inteligencia Artificial

Estudiante: Maribel Amaya Galaviz
Docente: Stephanie Cordero Martínez

Enlace en git

https://github.com/AmayaGM/IA/tree/main/Unidad%20III%20y%20IV/Practica2

Práctica 2 – Ropa


```
[ ] metadatos
```

```
tfds.core.DatasetInfo(
    name='fashion_mnist',
    full_name='fashion_mnist/3.0.1',
    description="""
    Fashion-MNIST is a dataset of Zalando's article images consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a
28x28 grayscale image, associated with a label from 10 classes.
    """,
    homepage='https://github.com/zalandoresearch/fashion-mnist',
    data_dir=PosixGPath('/tmp/tmpies5wqo1tfds'),
    file_format=tfrecord,
    download_size=29.45 MiB,
    dataset_size=36.42 MiB,
    features=FeaturesDict({
        'image': Image(shape=(28, 28, 1), dtype=uint8),
        'label': ClassLabel(shape=(), dtype=int64, num_classes=10),
    }),
    supervised_keys=('image', 'label'),
    disable_shuffling=False,
    splits={
        'test': <SplitInfo num_examples=10000, num_shards=1>,
        'train': <SplitInfo num_examples=60000, num_shards=1>,
    },
    citation="""@article{DBLP:journals/corr/abs-1708-07747,
      author    = {Han Xiao and
                   Kashif Rasul and
                   Roland Vollgraf},
      title     = {Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning
                   Algorithms},
      journal   = {CoRR},
      volume    = {abs/1708.07747},
      year      = {2017},
      url       = {http://arxiv.org/abs/1708.07747},
      archivePrefix = {arXiv},
      eprint    = {1708.07747},
      timestamp = {Mon, 13 Aug 2018 16:47:27 +0200},
      biburl    = {https://dblp.org/rec/bib/journals/corr/abs-1708-07747},
      bibsource = {dblp computer science bibliography, https://dblp.org}
    }""",
)
```

```
[ ] datos_entrenamiento, datos_pruebas = datos[('train')], datos['test']
```

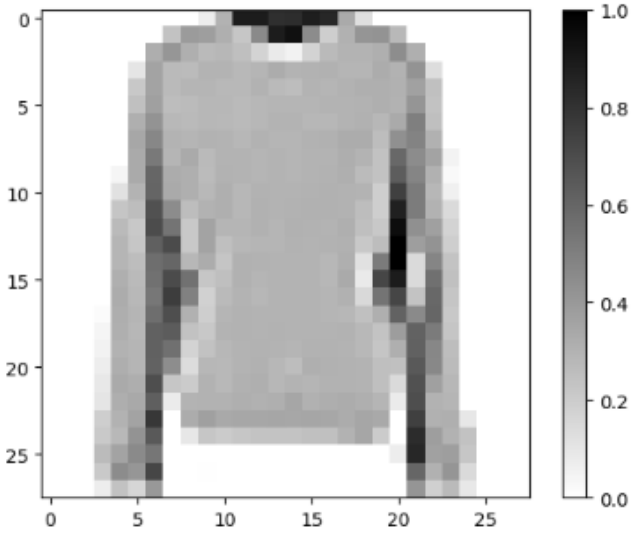
```
[ ] nombres_clases = metadatos.features['label'].names
```

```
[ ]  nombres_clases
```

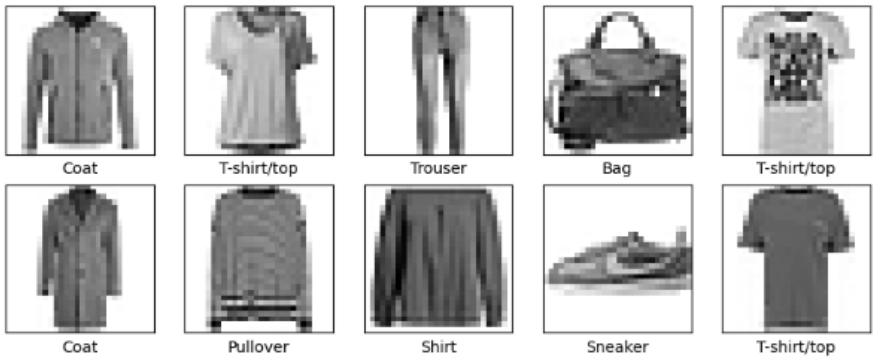
```
['T-shirt/top',
 'Trouser',
 'Pullover',
 'Dress',
 'Coat',
 'Sandal',
 'Shirt',
 'Sneaker',
 'Bag',
 'Ankle boot']
```

```
[ ]  def normalizar(imagenes,etiquetas):
       imagenes = tf.cast(imagenes, tf.float32)
       imagenes /= 255
       return imagenes,etiquetas

     datos_entrenamiento = datos_entrenamiento.map(normalizar)
     datos_pruebas = datos_pruebas.map(normalizar)

     datos_entrenamiento = datos_entrenamiento.cache()
     datos_pruebas = datos_pruebas.cache()
```

```
for imagen, etiqueta in datos_entrenamiento.take(1):
  break
imagen = imagen.numpy().reshape((28,28))

import matplotlib.pyplot as plt

plt.figure()
plt.imshow(imagen, cmap=plt.cm.binary)
plt.colorbar()
plt.grid(False)
plt.show()
```



```
[ ]  plt.figure(figsize=(10,10))
     for i, (imagen, etiqueta) in enumerate(datos_entrenamiento.take(25)):
       imagen = imagen.numpy().reshape((28,28))
       plt.subplot(5,5,i+1)
       plt.xticks([])
       plt.yticks([])
       plt.grid(False)
       plt.imshow(imagen, cmap=plt.cm.binary)
       plt.xlabel(nombres_clases[etiqueta])
     plt.show()
```

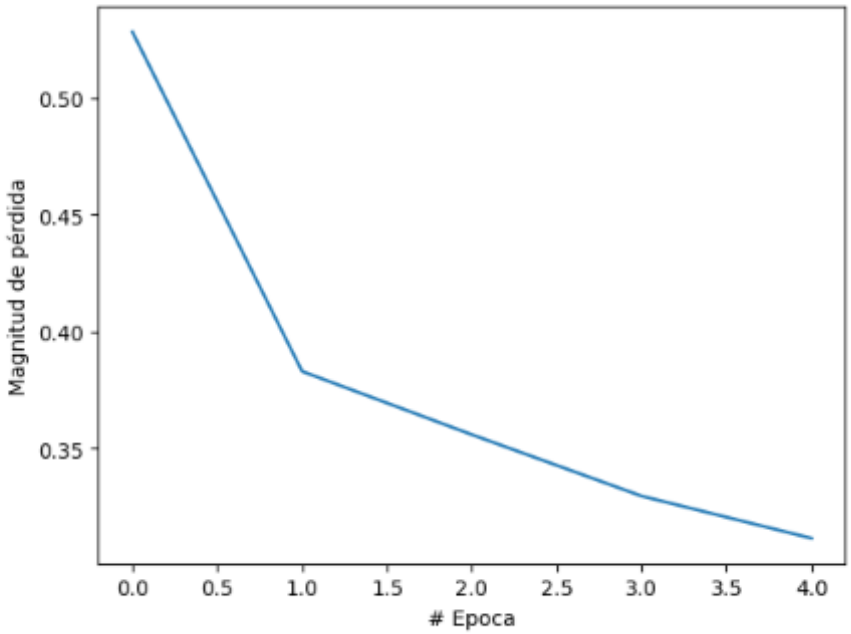



```
[ ]  modelo = tf.keras.Sequential([
       tf.keras.layers.Flatten(input_shape=(28,28,1)), #1 - blanco y negro
       tf.keras.layers.Dense(50, activation=tf.nn.relu),
       tf.keras.layers.Dense(50, activation=tf.nn.relu),
       tf.keras.layers.Dense(10, activation=tf.nn.softmax) #Para redes de clasificacion
     ])
```

```
[ ]  modelo.compile(
         optimizer='adam',
         loss=tf.keras.losses.SparseCategoricalCrossentropy(),
         metrics=['accuracy']
     )
```

```
[ ]  plt.xlabel("# Epoca")
     plt.ylabel("Magnitud de pérdida")
     plt.plot(historial.history["loss"])
```

```
[<matplotlib.lines.Line2D at 0x7d4564931810>]
```




```
[ ]  grafica[etiqueta_prediccion].set_color('red')
     grafica[etiqueta_real].set_color('blue')

     filas = 5
     columnas = 5
     num_imagenes = filas*columnas
     plt.figure(figsize=(2*2*columnas, 2*filas))
     for i in range(num_imagenes):
       plt.subplot(filas, 2*columnas, 2*i+1)
       graficar_imagen(i, predicciones, etiquetas_prueba, imagenes_prueba)
       plt.subplot(filas, 2*columnas, 2*i+2)
       graficar_valor_arreglo(i, predicciones, etiquetas_prueba)
```

```
1/1 [==============================] - 0s 90ms/step
```

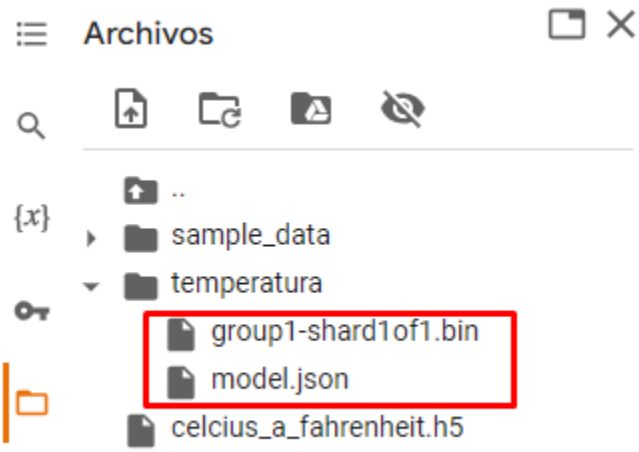
Conclusión

El código utiliza TensorFlow para construir, entrenar y evaluar un modelo de red neuronal para la clasificación de imágenes en el conjunto de datos Fashion MNIST. Este conjunto de datos contiene imágenes en escala de grises de prendas de ropa, y el objetivo es entrenar un modelo que pueda predecir la categoría de la prenda en la imagen.

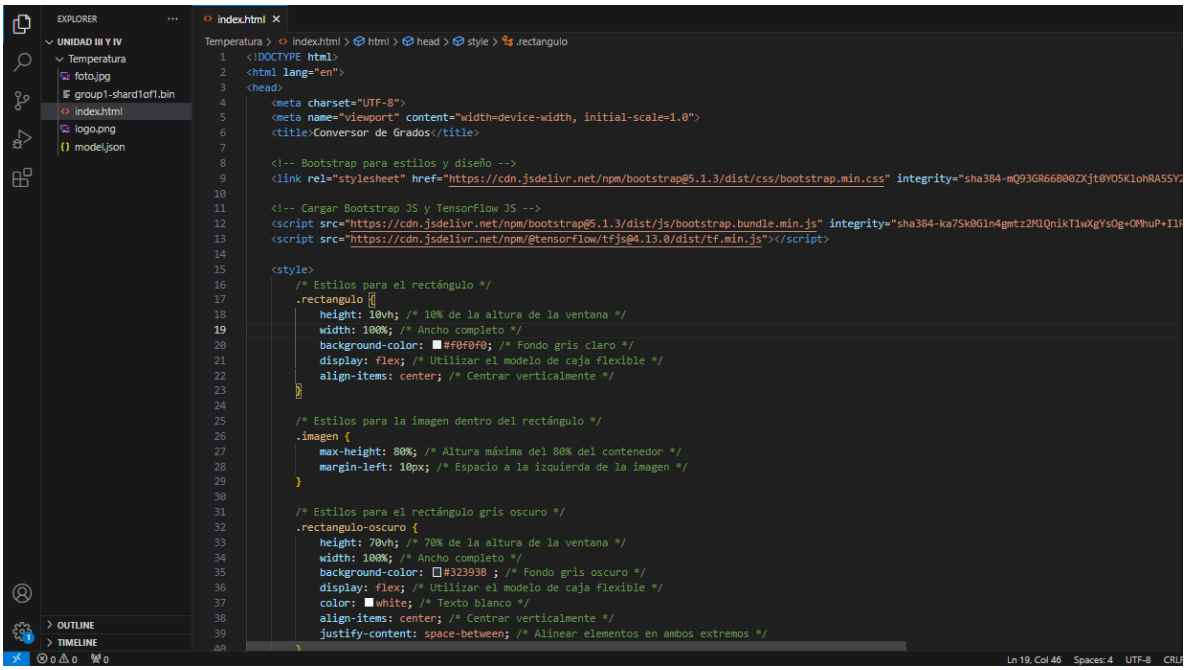
El flujo del código incluye la carga del conjunto de datos, la normalización de las imágenes, la definición de un modelo de red neuronal con capas densas, la compilación del modelo con una función de pérdida y un optimizador, y finalmente, el entrenamiento del modelo utilizando el conjunto de datos de entrenamiento. Después del entrenamiento, se visualizan algunas imágenes y se realizan predicciones en un conjunto de datos de prueba para evaluar el rendimiento del modelo. En última instancia, el código tiene como objetivo lograr la clasificación precisa de diferentes tipos de prendas de ropa en el conjunto de datos Fashion MNIST.

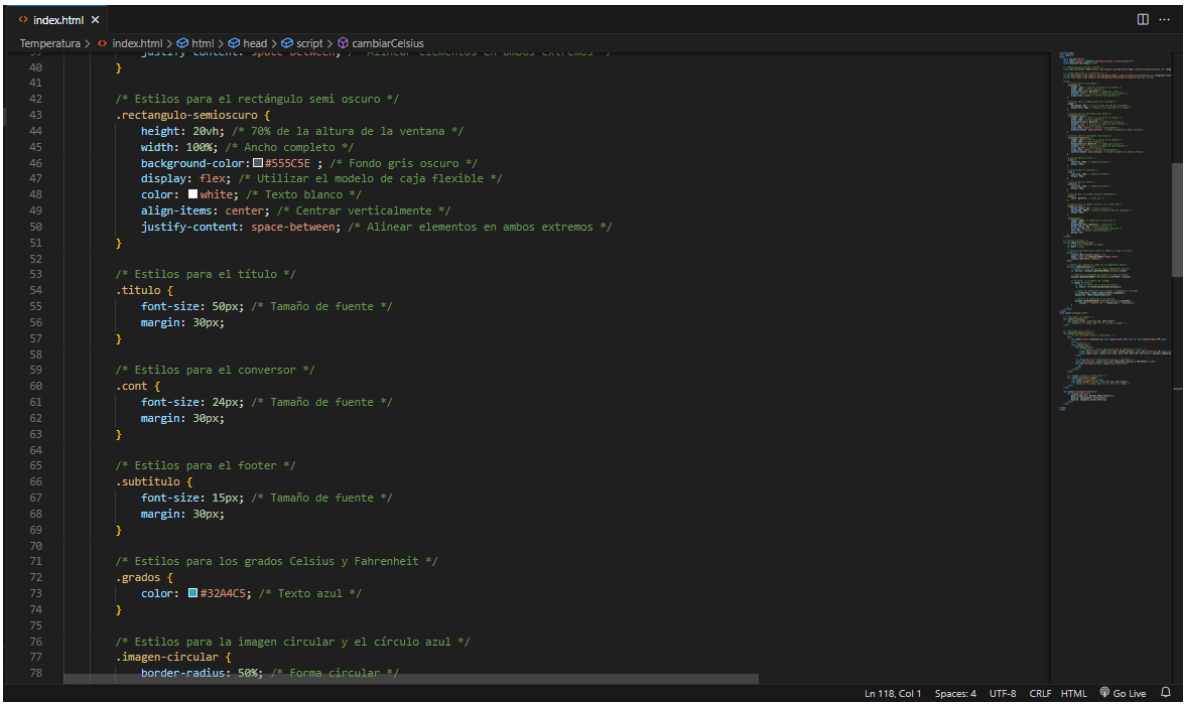
Practica 2 – Temperatura

En el cuaderno de la practica uno de temperatura en colab, se descargan los archivos .json y .bin.



Posteriormente se crea una carpeta llamada temperatura y dentro de la misma se hace un código html para crear una pagina web con los archivos de la IA como backend.


EXPLORER
UNIDAD III Y IV
Temperatura
foto.jpg
group1-shard1of1.bin
index.html
logo.png
model.json
OUTLINE
TIMELINE
index.html
Temperatura > index.html > html > head > style > .rectangulo
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Conversor de Grados</title>
<!-- Bootstrap para estilos y diseño -->
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" integrity="sha384-mQ93GR66B00ZXjt0YO5KlohRA5SY2
<!-- Cargar Bootstrap JS y TensorFlow JS -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlR
<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@4.13.0/dist/tf.min.js"></script>
<style>
/* Estilos para el rectángulo */
.rectangulo {
height: 10vh; /* 10% de la altura de la ventana */
width: 100%; /* Ancho completo */
background-color: #f0f0f0; /* Fondo gris claro */
display: flex; /* Utilizar el modelo de caja flexible */
align-items: center; /* Centrar verticalmente */
}
/* Estilos para la imagen dentro del rectángulo */
.imagen {
max-height: 80%; /* Altura máxima del 80% del contenedor */
margin-left: 10px; /* Espacio a la izquierda de la imagen */
}
/* Estilos para el rectángulo gris oscuro */
.rectangulo-oscuro {
height: 70vh; /* 70% de la altura de la ventana */
width: 100%; /* Ancho completo */
background-color: #32393B ; /* Fondo gris oscuro */
display: flex; /* Utilizar el modelo de caja flexible */
color: white; /* Texto blanco */
align-items: center; /* Centrar verticalmente */
justify-content: space-between; /* Alinear elementos en ambos extremos */
Ln 19, Col 46
Spaces: 4
UTF-8
CRLF


index.html
Temperatura > index.html > html > head > script > cambiarCelsius
}
/* Estilos para el rectángulo semi oscuro */
.rectangulo-semioscuro {
height: 20vh; /* 70% de la altura de la ventana */
width: 100%; /* Ancho completo */
background-color:#555C5E ; /* Fondo gris oscuro */
display: flex; /* Utilizar el modelo de caja flexible */
color: white; /* Texto blanco */
align-items: center; /* Centrar verticalmente */
justify-content: space-between; /* Alinear elementos en ambos extremos */
}
/* Estilos para el título */
.titulo {
font-size: 50px; /* Tamaño de fuente */
margin: 30px;
}
/* Estilos para el conversor */
.cont {
font-size: 24px; /* Tamaño de fuente */
margin: 30px;
}
/* Estilos para el footer */
.subtitulo {
font-size: 15px; /* Tamaño de fuente */
margin: 30px;
}
/* Estilos para los grados Celsius y Fahrenheit */
.grados {
color: #32A4C5; /* Texto azul */
}
/* Estilos para la imagen circular y el círculo azul */
.imagen-circular {
border-radius: 50%; /* Forma circular */
Ln 118, Col 1
Spaces: 4
UTF-8
CRLF
HTML
Go Live

index.html

Temperatura > index.html > html > head > style > .rectangulo

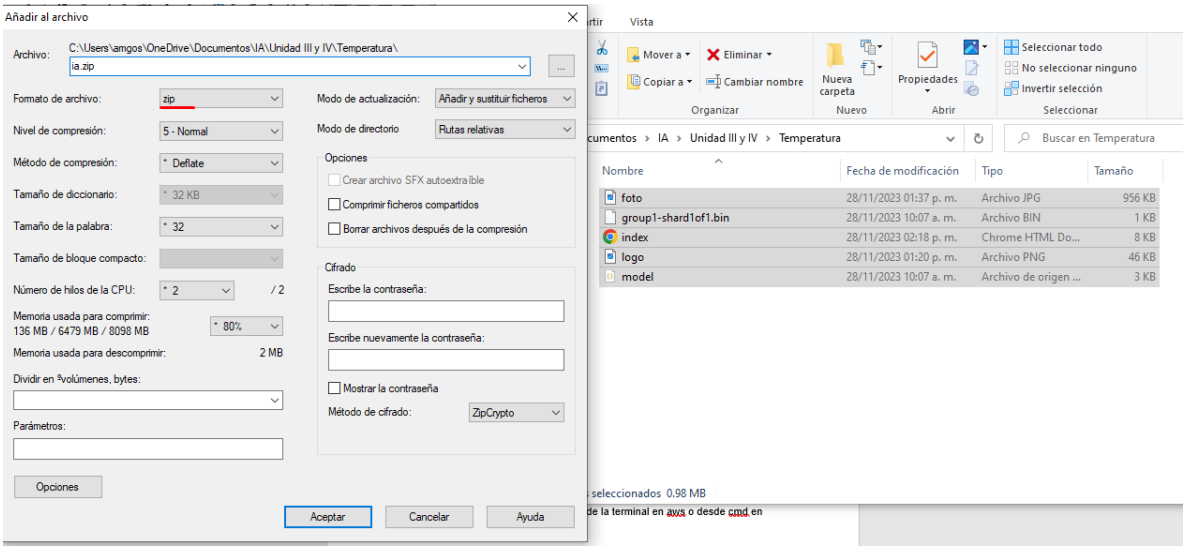
```
    </style>

    <!-- Script principal -->
    <script type="text/javascript">
        // Variable para almacenar el modelo
        var model = null;

        // Función asincrónica para cargar el modelo al cargar la página
        (async () => {
            console.log("Cargando modelo...");
            model = await tf.loadLayersModel("model.json");
            console.log("Modelo cargado");
        })();

        // Función para manejar el cambio en la temperatura Celsius
        function cambiarCelsius() {
            // Obtener el valor actual del rango (temperatura Celsius)
            var celsius = document.getElementById("celsius").value;

            // Actualizar la etiqueta que muestra el valor de Celsius
            document.getElementById("lbl-celsius").innerHTML = celsius;

            // Verificar si el modelo está cargado
            if (model != null) {
                // Crear un tensor con el valor de Celsius
                var tensor = tf.tensor1d([parseInt(celsius)]);

                // Hacer una predicción con el modelo y redondear el resultado
                var prediccion = model.predict(tensor).dataSync();
                prediccion = Math.round(prediccion);

                // Mostrar la conversión en la interfaz
                document.getElementById("resultado-celsius").innerHTML =
                    celsius + " Celsius son " + prediccion + " Fahrenheit";
            }
        }
    </script>
</head>
<body class="container mt-5">
```

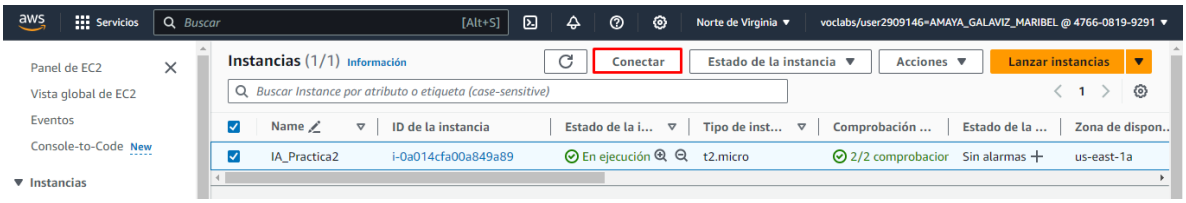
index.html

Temperatura > index.html > html > body.container.mt-5 > div.rectangulo-oscuro

```
130  <body class="container mt-5">
131      <!-- Rectángulo con Imagen -->
132      <div class="rectangulo">
133          <img class="imagen" src="logo.png" alt="Imagen">
134          <!-- Reemplaza "tu-imagen.jpg" con la ruta de tu imagen -->
135      </div>
136      <!-- Rectángulo gris oscuro -->
137      <div class="rectangulo-oscuro">
138          <!-- Título con grados Celsius y Fahrenheit -->
139          <div>
140              <h1 class="titulo">Conversor de <span class="grados">ºC</span> a <span class="grados">ºF</span>
141              </h1>
142              <!-- Formulario -->
143              <form class="cont">
144                  <div class="mb-3">
145                      <!-- Etiqueta y control deslizante para la temperatura Celsius -->
146                      <label for="celsius" class="form-label">Grados Celsius: <span id="lbl-celsius">0</span></label>
147                      <input type="range" class="form-range" min="-200" max="200" id="celsius" oninput="cambiarCelsius();">
148                  </div>
149
150                  <!-- Resultado de la conversión de Celsius a Fahrenheit -->
151                  <label for="celsius" class="form-label">Resultado (Celsius a Fahrenheit)</label>
152                  <div id="resultado-celsius" class="alert alert-info">
153                      0...
154                  </div>
155              </form>
156          </div>
157          <!-- Imagen circular y círculo azul -->
158          <div class="contenedor-imagen">
159              <div class="circulo-azul"></div>
160              <img class="imagen-circular" src="foto.jpg" alt="Imagen">
161              <!-- Reemplaza "tu-imagen.jpg" con la ruta de tu imagen -->
162          </div>
163      </div>
164      <div class="rectangulo-semioscuro">
165          <div class="subtitulo">
166              Desarrollado por: Maribel Amaya Galaviz<br>
167              Materia: Inteligencia artificial<br>
168              Maestra: Stephanie Cordero Martínez
```

Posteriormente se crear un archivo zip con todos los archivos necesarios para cargar nuestra página y creamos una mv en la nube de aws





Nos conectamos a ella, puede ser desde la terminal en aws o desde cmd en nuestra computadora.

En este caso se conecta a cmd y se cargan los archivos a la máquina virtual.

```
C:\Users\amgos\Downloads>scp -i IA.pem C:\Users\amgos\Downloads\ia.zip ubuntu@ec2-54-160-143-208.compute-1.amazonaws.com
:\home\ubuntu
ia.zip                                                    100% 1004KB 698.7KB/s   00:01
```

Después, entramos a nuestra máquina virtual

```
ubuntu@ip-172-31-27-103: ~

C:\Users\amgos\Downloads>ssh -i "IA.pem" ubuntu@ec2-54-160-143-208.compute-1.amazonaws.com
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-1012-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Nov 28 20:20:19 UTC 2023

  System load:  0.01220703125     Processes:             100
  Usage of /:   6.7% of 28.89GB   Users logged in:       0
  Memory usage: 25%               IPv4 address for eth0: 172.31.27.103
  Swap usage:   0%

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

9 updates can be applied immediately.
7 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
```

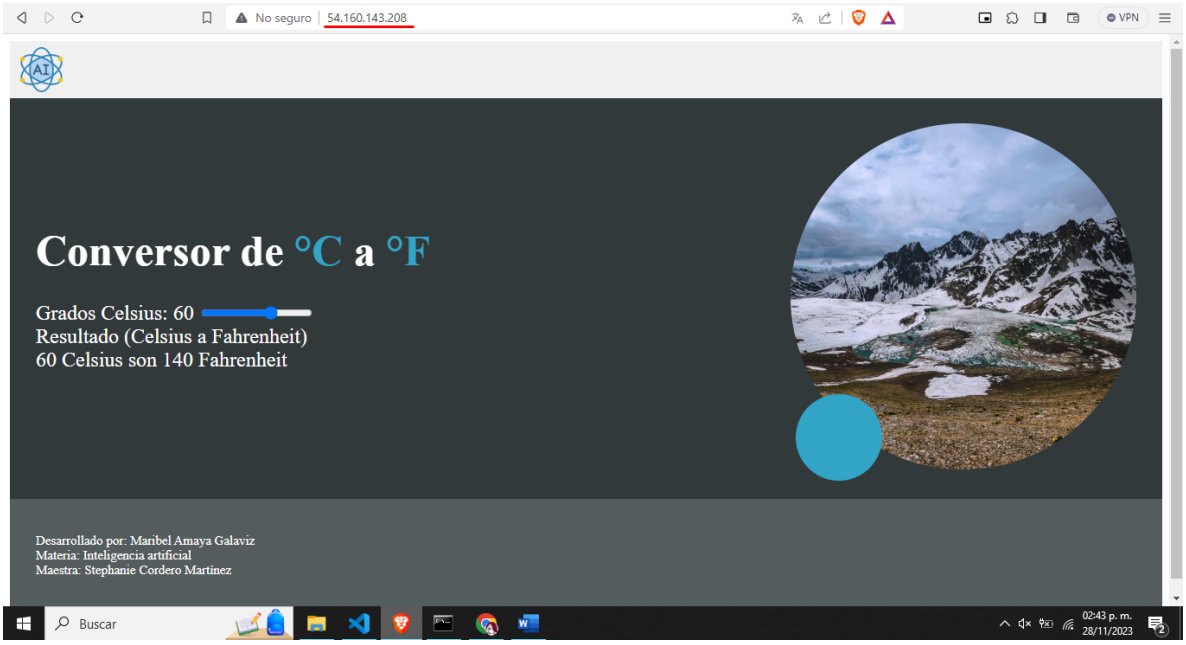
Una vez dentro de la maquina virtual se comprueba que el zip se haya cargado y se descomprime.

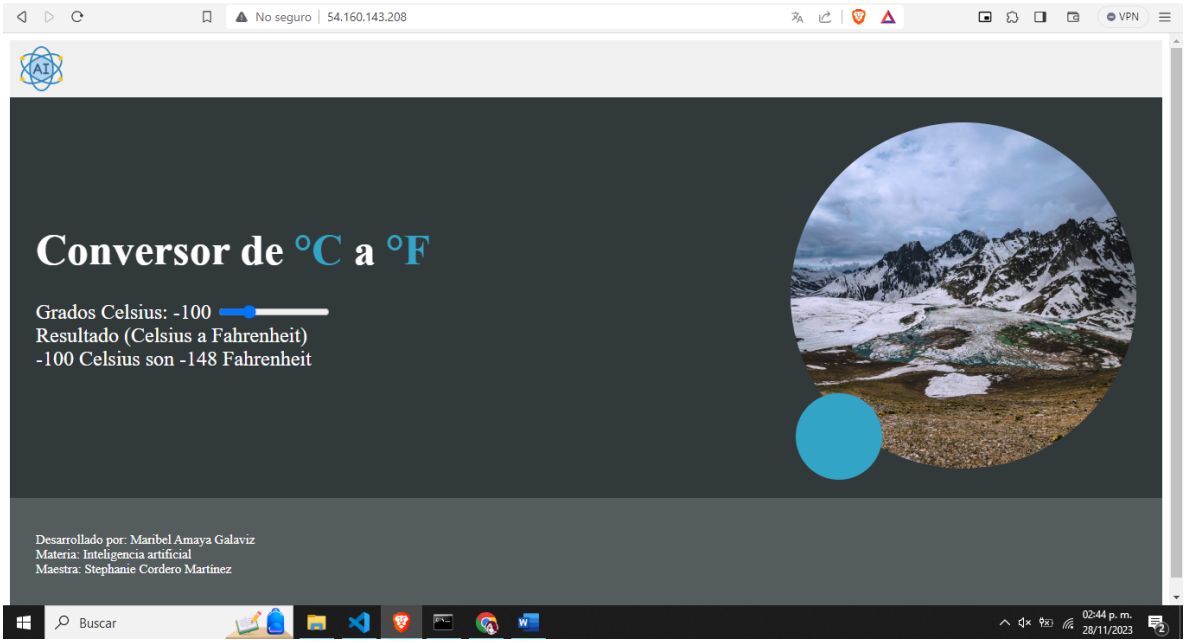
```
*** System restart required ***
Last login: Tue Nov 28 18:45:59 2023 from 201.152.103.193
ubuntu@ip-172-31-27-103:~$ ls
ia.zip
ubuntu@ip-172-31-27-103:~$ unzip ia.zip
Archive:  ia.zip
   creating: ia/
 extracting: ia/foto.jpg
 extracting: ia/group1-shard1of1.bin
  inflating: ia/index.html
  inflating: ia/logo.png
  inflating: ia/model.json
```

Por último, cambiamos la carpeta de la pagina web al servidor y reiniciamos el servidor

```
ubuntu@ip-172-31-27-103:~$ sudo cp -r ia/* /var/www/html/
ubuntu@ip-172-31-27-103:~$ sudo systemctl restart apache2
```

Nuestra página estará lista, para comprobarlo tecleamos la ip de la pagina y podremos ver como esta se carga correctamente.





Conclusión

Se desarrollo una página web que incluye el conversor de grados utilizado en la practica 1. Se utilizó un modelo JSON generado por una red neuronal para realizar la conversión de °C a °F, en este punto fue de gran utilidad el código desarrollado en colab ya que gracias a este se pudo obtener el archivo JSON. Además, se hizo uso de un servidor AWS para alojar la página web, asegurando así su disponibilidad en línea. Los conocimientos adquiridos durante las clases de IA fueron de gran utilidad para el desarrollo de esta práctica.