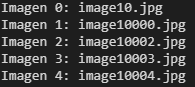
descripción de los datos empleado

El punto de partida consta de un banco de 5983 imágenes y un archivo Data.csv.

En lo referente a las imágenes se cumple que:

* Nombre: image**#N**.jpg-🡪 **#N**: número de imagen.
* Imagen a color RGB.
* Dimensiones distintas.

Ejemplo:



En lo referente al archivo Data.csv, se clasifican las imágenes en 4 grupos:

'Food', 'Attire', 'Decorationandsignage', 'misc'.

Ejemplo:

|  |  |
| --- | --- |
| **Image** | **Class** |
| image7042.jpg | Food |
| image3327.jpg | misc |
| image10335.jpg | Attire |
| image8019.jpg | Food |
| image2128.jpg | Attire |
| image1106.jpg | misc |
| image6750.jpg | Food |

Partiendo de estos datos y con el uso del lenguaje de programación Python, se realizará el entrenamiento y posterior comparación de modelos de aprendizaje automático aplicados a la clasificación de imágenes.

Partida: 5983 Imágenes Clasificadas.

El primer paso, antes de entrenar cualquier modelo, es preparar las variables que vamos a introducir como entrada.

**Tratamiento: Entradas**

Objetivo:

x: entrada del modelo, matriz de bits de una imagen.

t: salida del modelo, clasificación entrada.

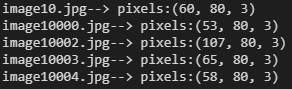
x 🡪 🡪t

MODELO

PASO 1:

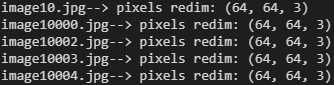
Las imágenes no tienen la misma dimensión, por lo que se redimensionan para que todas tengan la misma 64x64 Pixeles.

Imágenes originales:





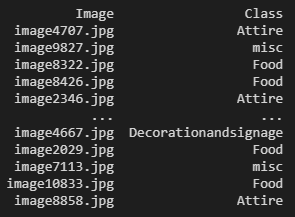
Imágenes redimensionadas:



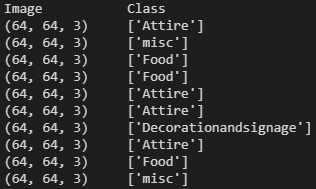
PASO 2:

La clasificación de la imagen que se representa en el archivo Data.csv se asocia el nombre de una imagen con una clase. En nuestro caso nos interesa asociar la matriz de pixeles de cada imagen con su clase.

imagen.jpg 🡪clase

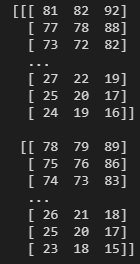


Matriz píxeles 🡪clase



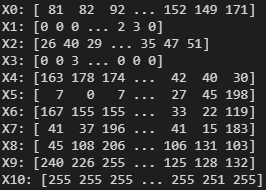
PASO 3:

Las matrices de pixeles no se pueden introducir como entradas a los modelos directamente, hay que realizar un proceso de aplanamiento de la matriz de forma que se representen todos los pixeles de la imagen en un único vector.

(64,64,3) (1, 12288)

🡪 

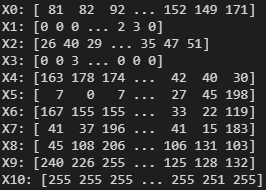
De esta forma para las 5983 imágenes tendremos todos los vectores de entrada de nuestro modelo, en el siguiente ejemplo se muestran las primeras 11 entradas.



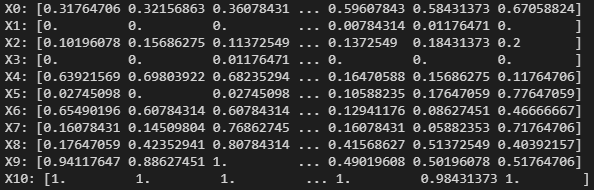
PASO 4:

Para miminizar el tiempo de entrenamiento y debido a que los datos tienen una variación de 0 a 255 debido a que son de 8 bits, se opta por normalizar las entradas de forma que varien entre 0 y 1 y de esta forma lograr que el resultado del entrenamiento sea mas preciso.

Sin normalizar



Normalizado

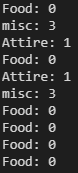


PASO 5:

Para facilitar el entrenamiento de los modelos, las clases que tienen un formato del tipo “string” se asocian con un número.

'Food': 0, 'Attire': 1, 'Decorationandsignage':2, 'misc':3

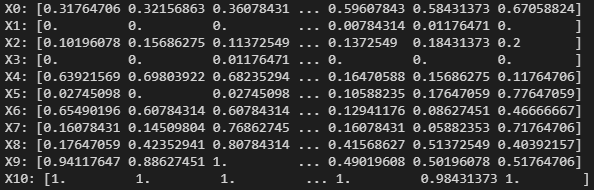
Ejemplo:



De esta forma obtenemos que:



Finalmente ya podemos realizar los entrenamientos de los modelos con el uso de nuestras entradas salidas previamente tratadas:

MODELO

experimentos realizados

análisis de los resultados

conclusiones obtenidas