**Actividades del 27 de septiembre 2019 al 29 de febrero 2020**

Tomé un curso en línea en la página web [*www.udemy.com*](http://www.udemy.com) en la cuenta *Jacmuami@gmail.com* en la cual se provee de cursos a los becarios, el curso en cuestión tiene por título “***Curso completo de Machine Learning: Data Science en Python****”.* En tal cuenta se puede consultar el curso y el avance.

También estudie un minicurso específico de Redes Neuronales en el canal de Youtube, *“****DotCSV****”* para mayor comprensión del tema.

Las tareas realizadas en esos cursos fueron ejercicios en Python y se encuentran en el repositorio creé exclusivamente para que consulten, ya que, sería inadecuado incrustarlos en este documento.

Repositorio de ejercicios: [aquí va dirección de github]

Lo anterior, lo realicé con el objetivo de poder llevar a cabo las actividades que prosiguen.

**Actividades del 01 de marzo 2020 al 05 de mayo 2020**

Se cuenta con una base de datos de imágenes, cada imagen corresponde a una foto por microscopio de algún hongo.

La base de datos está compuesta por cinco carpetas, cada carpeta está nombrada con el nombre de la especie a la que pertenecen las fotos que contiene, de esta forma se tienen fotos de cinco especies de hongos

Se desea entrenar un modelo computacional con estas imágenes para que pueda reconocer una imagen ajena a la base, perteneciente a alguna de las cinco especies.

A continuación, reporto los resultados que arrojó el programa en lenguaje Python que codifiqué para llevar a cabo dicho objetivo.

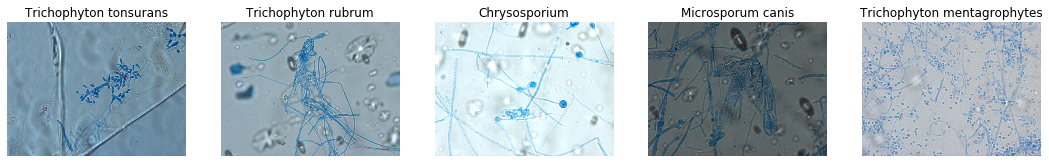
[salto de página]

**Reconocimiento de imágenes de hongos**

Se tienen 412 imágenes que conforman la base.

Todas las imágenes poseen las mismas tres dimensiones:

(1040, 1388, 3) = (número de píxeles en eje vertical, número de píxeles en eje horizontal, número de colores en cada pixel)



1040 píxeles

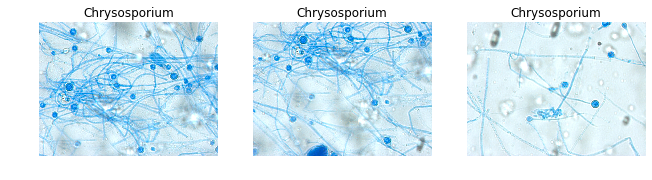
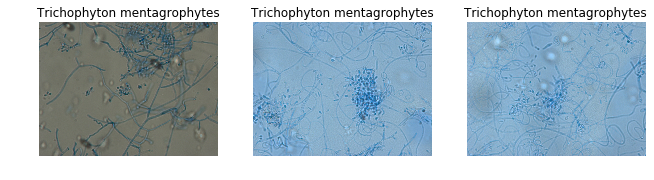
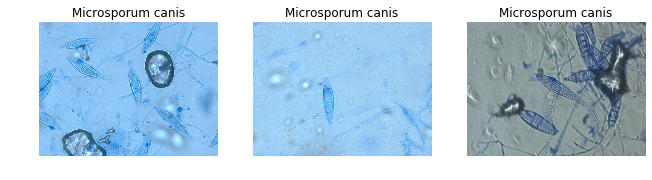
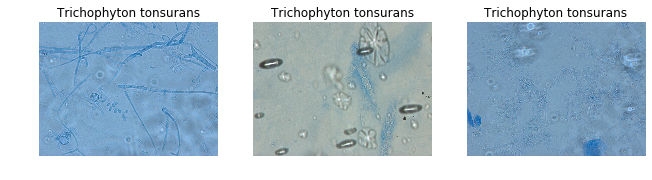
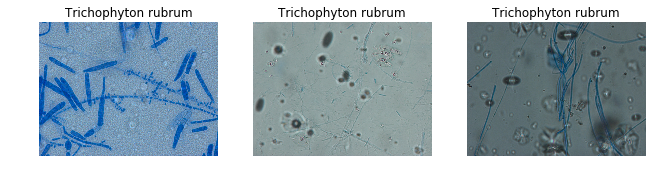
1388 píxeles

A cada imagen se le asigna una etiqueta de texto, en la cual se especifica la especie a la que pertenecen los hongos en la imagen.

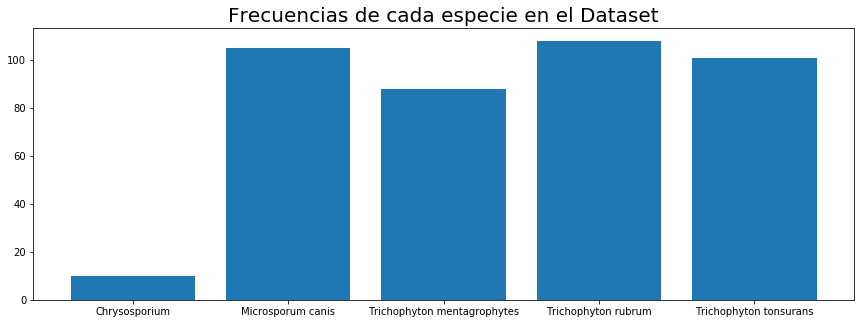
Las especies de hongos que conforman la base son:

* *Chrysosporium*
* *Microsporum canis*
* *Trichophyton mentagrophytes*
* *Trichophyton rubrum*
* *Trichophyton tonsurans*

A continuación, un ejemplo de fotos con su respectiva etiqueta:



Se despliega un histograma de frecuencias para visualizar el número de fotos con las que cuenta la base para cada especie:



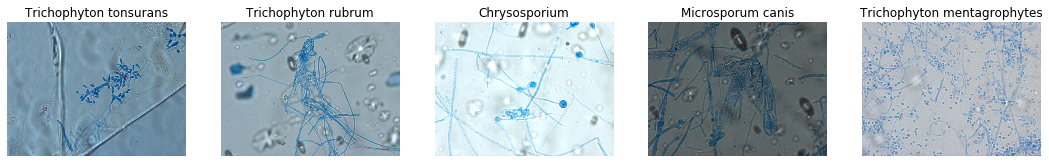
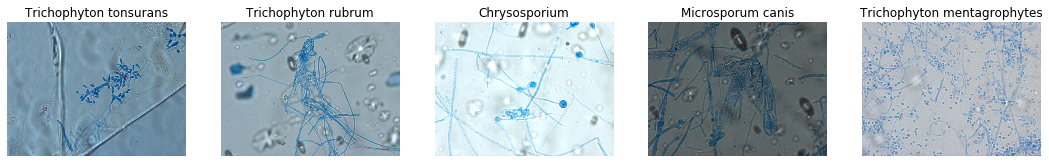
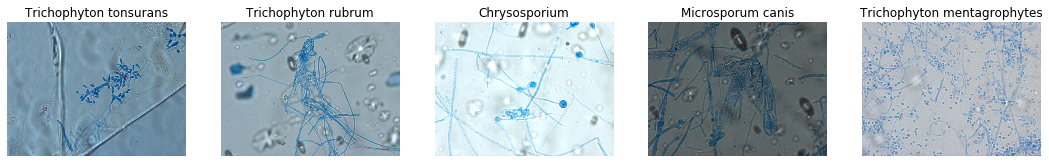
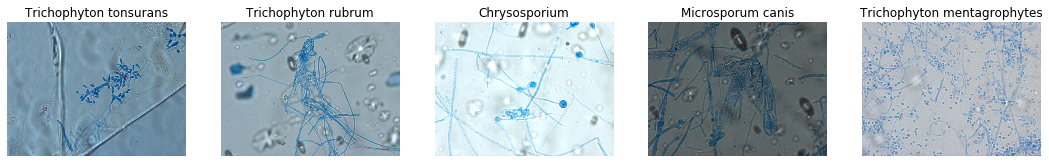
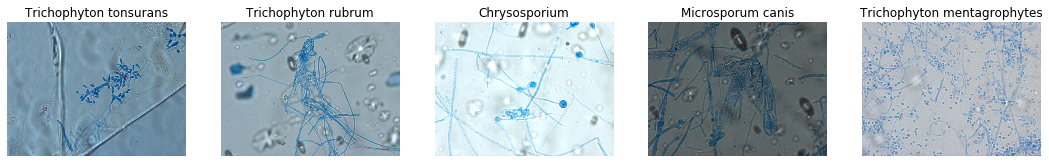
10 105 88 108 101

Se puede apreciar que la especie *Chrysosporium* tiene tan sólo 10 elementos, por lo tanto, decido que no se dividirá el dataset en imágenes de entrenamiento e imágenes de prueba, ya que el modelo podría perder demasiada información de esa especie, quedándose con 7 u 8 imágenes. En cambio, la validación al final se hará utilizando una muestra aleatoria del dataset (20% de fotos elegidas aleatoriamente).

Se consideran todas las imágenes como conjunto de entrenamiento.

Se procede al tratamiento de las imágenes.

El tamaño de cada imagen es 1040 x 1388 x 3 = 4 330 560 bytes (producto de las dimensiones), esto es demasiado grande para trabajar. Después de probar varias escalas para redimensionar, se obtiene que una proporción donde se pueden aún identificar los hongos pero que ya no le cuesta trabajo procesar al computador es 1/10. Entonces, las nuevas dimensiones de las imágenes son: (104, 139, 3)



**Imágenes Originales**

**(1040, 1388, 3)**



**Imágenes Redimensionadas**

**(104, 139, 3)**

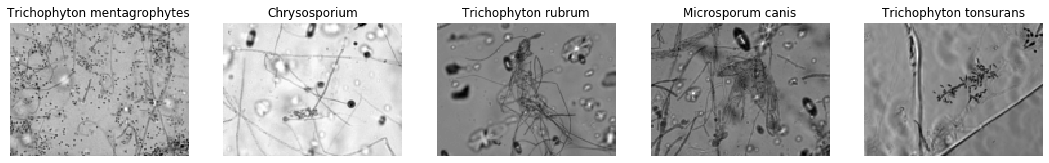
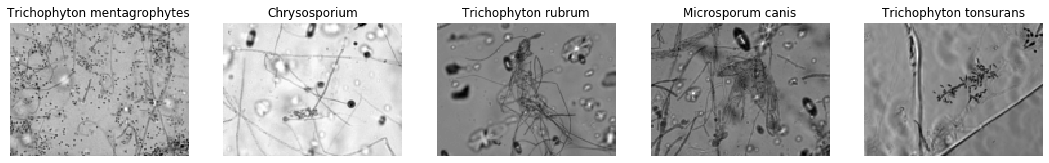
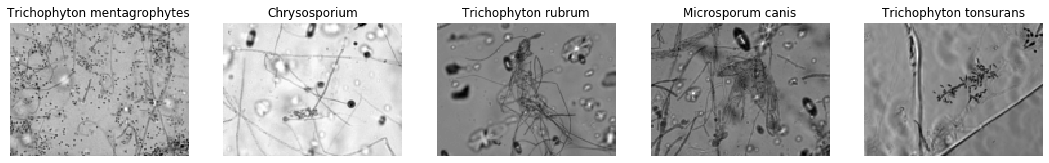
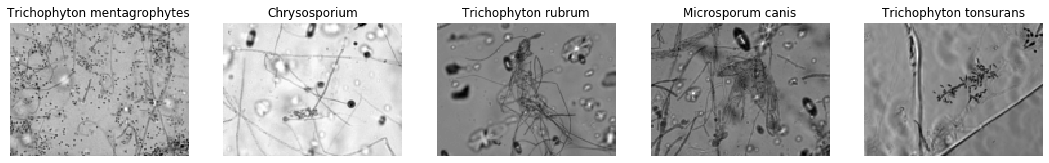
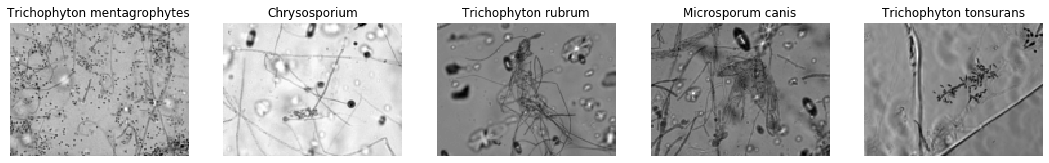
Las imágenes están dadas en escala de color RGB. Significa que en cada pixel hay tres valores, que especifican la carga de cada color (*Red, Green, Blue*) para componer el color final del pixel. Sin embargo, esta manera de leer las fotos se presta a malas interpretaciones por parte del modelo, ya que, si dos fotos que muestran formas muy parecidas se tomaron bajo distinta iluminación, el modelo puede reconocerlas como de diferentes categorías. Para evitar tales malos entendidos, y solamente estudiar el patrón de las figuras, se deben transformar las imágenes a escala de grises.

Ejemplo de representación computacional de un pixel en escala RGB:

[0.87058824, 0.92352941, 0.96470588]

Ejemplo de representación computacional de un pixel en escala de grises:

[0.91524824]



**Imágenes en**

**Escala de Grises**

**(104, 139, 1)**

Una vez que las imágenes están redimensionadas y en escala de grises, cada imagen está representada por una matriz de 104 x 139, lo cual es inconveniente porque los modelos de perceptrón reciben a cada individuo como un arreglo de una sola dimensión. Para cumplir este requerimiento, se procede a trasformar las matrices (arreglos de dos dimensiones) en vectores (arreglos de una dimensión).

En este punto, ya no se pueden visualizar las imágenes, ya que, cada una es un vector de 14456 entradas.

Se procede a construir el modelo de identificación de imágenes.

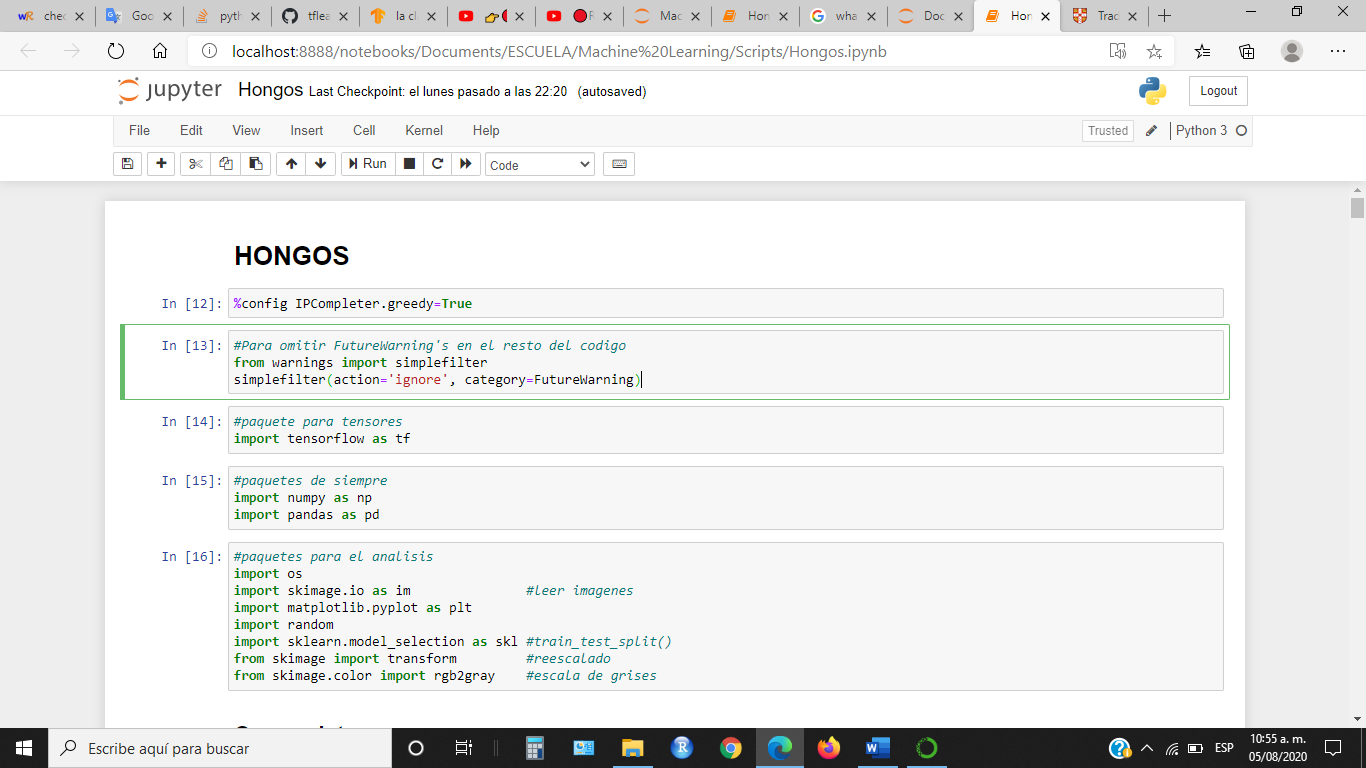
El modelo planeado es una red neuronal artificial básica, un perceptrón multicapa contruido con Tensorflow.

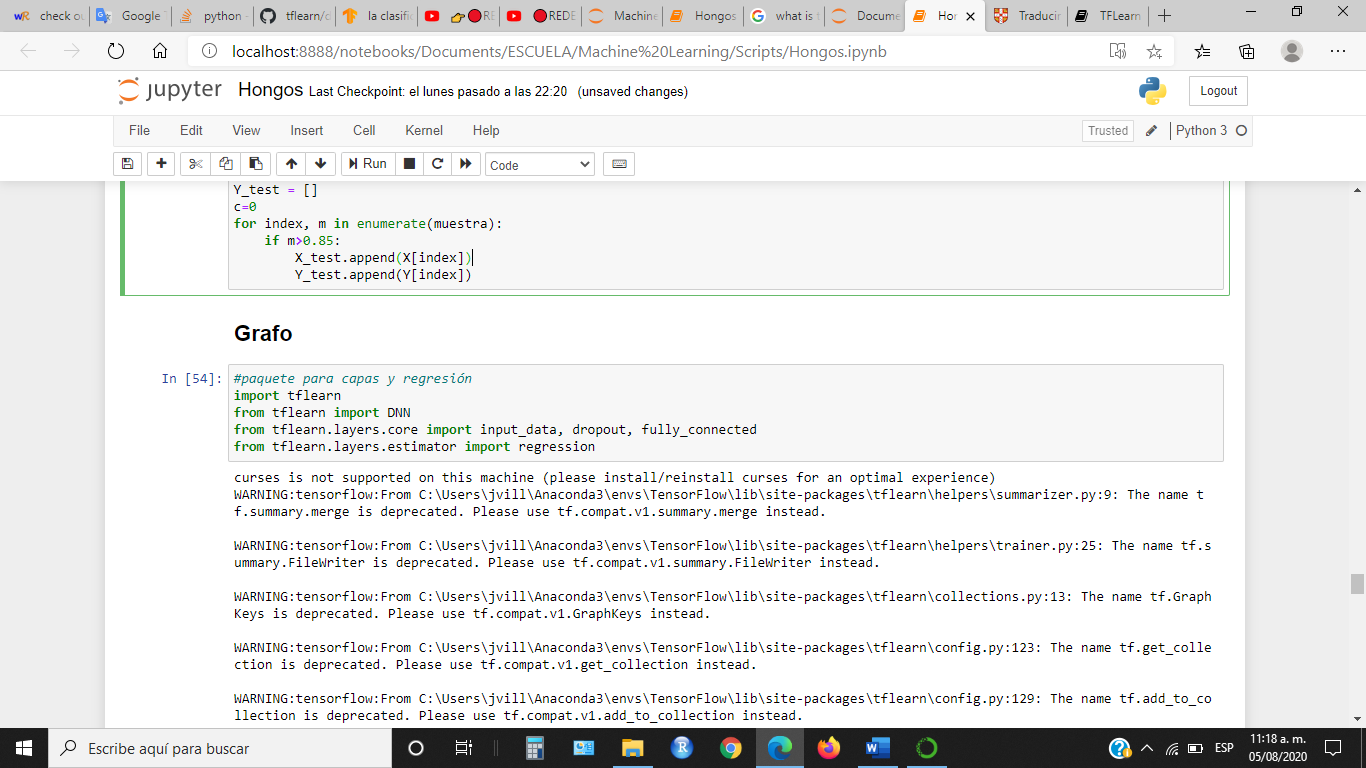
*“****TensorFlow*** *es una biblioteca de software libre y de código abierto para el flujo de datos y la programación diferenciable en una amplia gama de tareas. Es una biblioteca matemática simbólica, y también se utiliza para aplicaciones de aprendizaje automático como redes neuronales. Se utiliza tanto para la investigación como para la producción en Google...TensorFlow fue desarrollado por el equipo de* ***Google Brain*** *para uso interno de Google.”* Citado de <https://en.wikipedia.org/wiki/TensorFlow>

TensorFlow aislado es más aprovechable cuando se construye la red al nivel más bajo, es decir, neurona por neurona y conexión por conexión. Pero para abreviar el código se utiliza construcción de un nivel un poco más alto, es decir, capa por capa. Para ello se utilizará TFlearn.

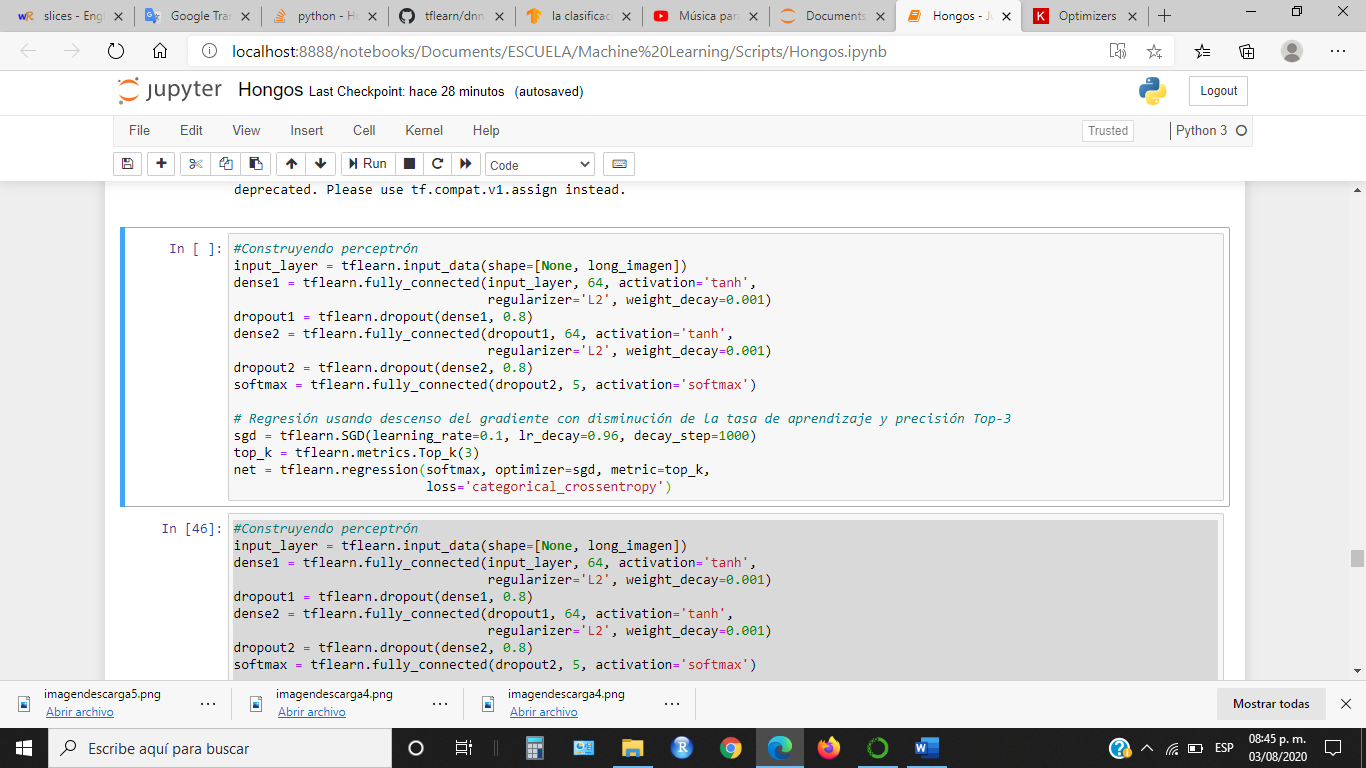
*“****TFlearn*** *es una biblioteca de aprendizaje profundo modular y transparente construida sobre* ***Tensorflow****. Fue diseñado para proporcionar una API de nivel superior a TensorFlow con el fin de facilitar y acelerar los experimentos”* Citado de <http://tflearn.org/>

Basta con importar las librerías adecuadas:



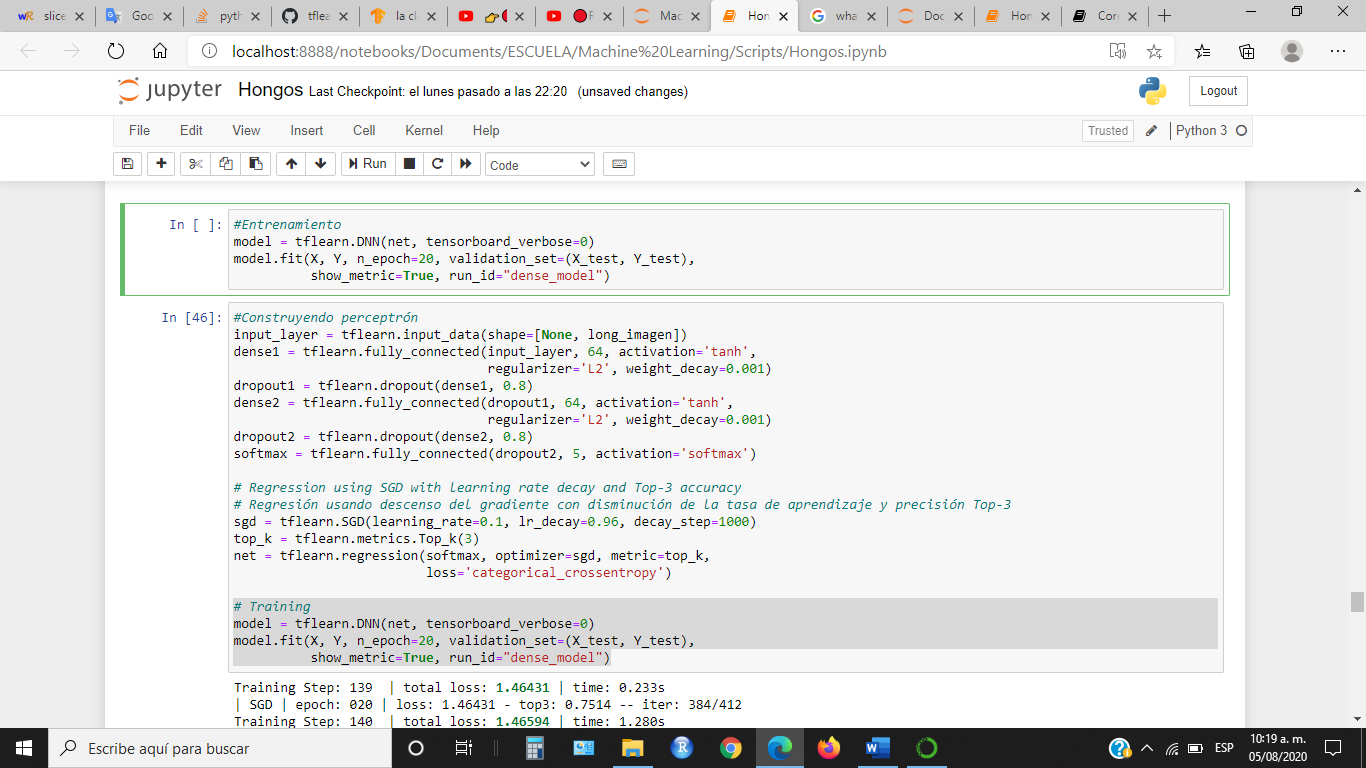


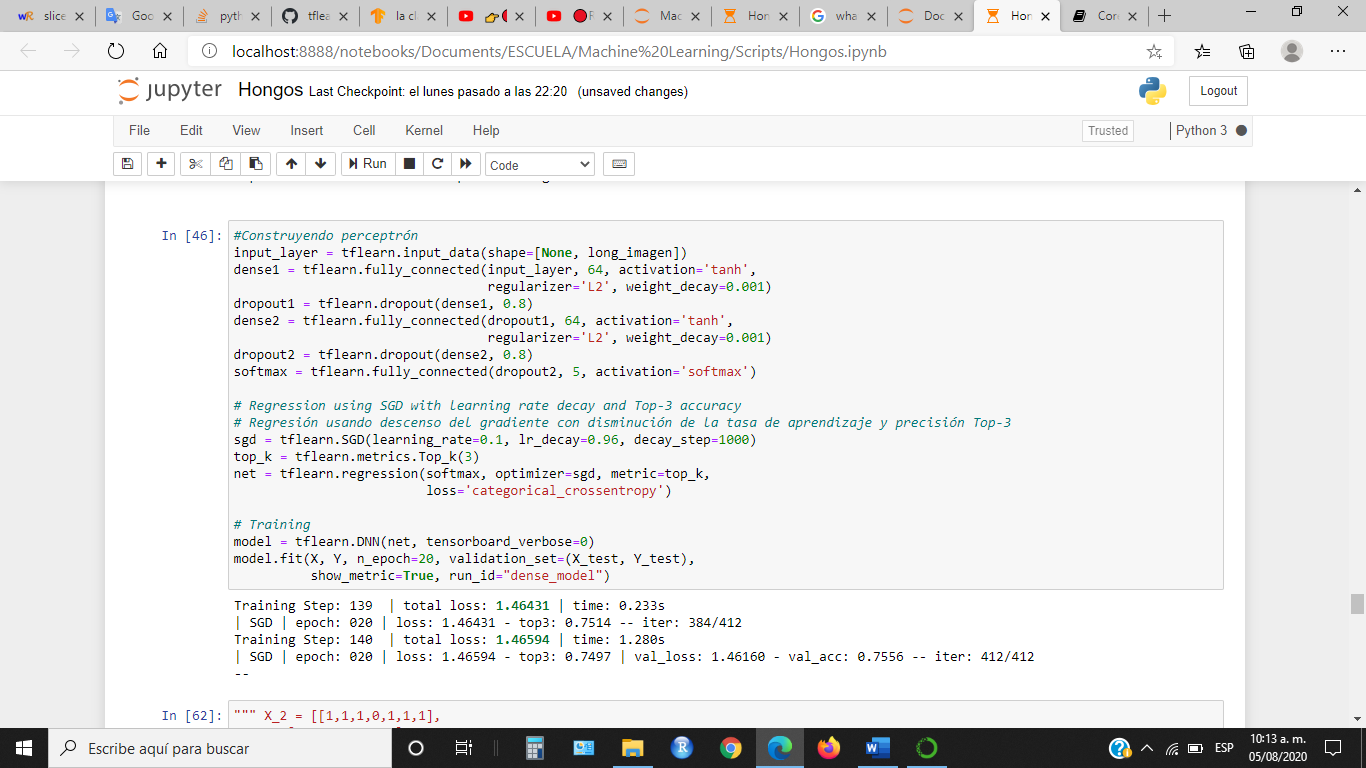
A continuación, se muestra un fragmento del código donde se construye el modelo de perceptrón multicapa para que identifique las imágenes.



El código anterior es un grafo está compuesto por una capa de entrada, dos capas ocultas completamente conectadas (todas las neuronas de la capa anterior se conectan con todas las neuronas de la nueva capa) de 64 neuronas cada una y una capa de salida.

En seguida se entrena la red, proporcionándole de entradas las imágenes ya tratadas. Y se muestra el resultado de aplicar el modelo entrenado a un conjunto de validación, el cual, en este caso es una muestra aleatoria de 82 fotos de la misma base.





La métrica de pérdida no mostró mejora aproximadamente desde el paso 100 de ajuste, y antes de 100 disminuyó muy poco, nada significativo. Y la métrica de precisión disminuyó en los últimos *epochs* (conjuntos de pasos de entrenamiento).

Al final, el modelo acertó con el 75% de las imágenes.

El programa se puede consultar en: [link del repositorio y nombre de script]