Python
El lenguaje del futuro

Antonio Espín Herranz

Contenidos

- ¿QUÉ ES PYTHON?
  - En pocas palabras
  - ¿Por qué se creó? ¿Quién y dónde formaron ideas?
  - Características diferenciales

- CASOS DE ÉXITO EN PYTHON

- CAMPOS EN LOS QUE SE APLICA PYTHON
  - Módulos implicados en cada campo
  - Resumen de uso en cada campo

¿Qué es Python?

- Definición:
  - Es un lenguaje interpretado, interactivo y orientado a objetos que ofrece una gran cantidad de estructuras de datos de alto nivel por medio de un tipado dinámico y fuerte, además de estas características es multiparadigma y multiplataforma.
  - Para aumentar su velocidad se puede integrar con otros lenguajes de nivel medio como C.

Historia de Python

- Python fue creado por Guido van Rossum
- Entre finales de los años 80 y principios de los 90
- El primer artículo data de 1991
- El nombre de Python viene del grupo Monty Python.
- Es la evolución de un lenguaje llamado ABC

Principales Ideas

- Las principales ideas de ABC que influyeron o se incluyeron en Python fueron según el propio Guido:
  - La sangría para agrupar el código.
    - Desde cero te obliga a ser ordenado en la forma de escribir el código
  - El carácter **:** para indicar que comienza un bloque indentado (después de pruebas con usuarios)
  - El diseño simple de las instrucciones: if, while, for,...
  - Tuplas, listas, diccionarios (fuertemente modificados en Python)

Principales Ideas II

- Tipos de datos mutables / inmutables.
  - Python distingue entre tipos mutables (**list, set, dict**) e inmutables (**int, float, … str, tuple**)

  - No imponer límites, como tamaño de un array, etc...
    - Las colecciones crecen automáticamente bajo demanda
    - No es necesario liberar memoria con en C / C++

Principales Ideas III

- El "**prompt" >>>**
  - Para interactuar directamente con el interprete en modo consola.

```
C:\Users\Anton>python
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from random import randint
>>> L = [randint(1,100) for _ in range(20)]
>>> L
[71, 21, 68, 49, 29, 31, 3, 52, 66, 74, 90, 32, 44, 67, 60, 87, 6, 28, 19, 52]
>>>
```

La consola se arranca con el comando: **python**
Python3 desde Linux
OJO con la variable de entorno PATH.

Versiones en Python

- Actualmente 3.10.8
- En desuso Python 2.7 ya no se actualiza.

- Versionado en Python / Librerías:
- **X.Y.Z**
  - Cambios en Z → Correcciones de Bugs
  - Cambios en Y → Nuevas funcionalidades
  - Cambios en X → Cambios importantes en el core.
    - Por ejemplo de Python 2 al 3.
      - Sin cambios importantes en la sintaxis
      - Muchas mejoras en la gestión de la memoria.

Python - PyPI

- **Referencia oficial de Python**
- www.python.org
  - Para descargar el interprete
  - Consultar documentación.

  - El **PyPI**.
    - https://pypi.org/
    - El repositorio donde se encuentran todas las librerías publicadas.
    - Comando importante en Python: **pip**
      - Para la instalación de librerías: ***pip install <nombre_librería>***

- **Historia de Python**
  - https://es.wikipedia.org/wiki/Historia_de_Python

Comparativa con otros lenguajes

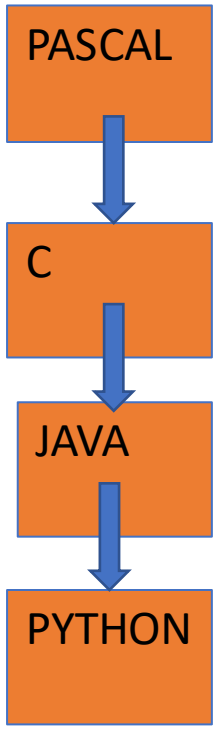
- Más fácil de aprender que otros lenguajes: C, Java
  - Actualmente se utiliza en universidades para aprender a programar.
- Sintaxis minimalista y a la vez elegante y estructurado.
- Gratuito
- Mucho más que un lenguaje tipo Script
- Código muy denso, con pocas instrucciones se han muchas cosas.
  - Descargar una página Web con 3 o 4 instrucciones.

Lenguajes de programación más utilizados

- Esta es la lista de los lenguajes más utilizados.
- En el año 2021 ocupaba el tercer lugar
- Este año se encontraba en el primer puesto

Jul 2022	Jul 2021	Change	Programming Language	Ratings	Change
1	3	↑	Python	13.44%	+2.48%
2	1	↓	C	13.13%	+1.50%
3	2	↓	Java	11.59%	+0.40%
4	4		C++	10.00%	+1.98%
5	5		C#	5.65%	+0.82%
6	6		Visual Basic	4.97%	+0.47%
7	7		JavaScript	1.78%	-0.93%
8	9	↑	Assembly language	1.65%	-0.76%
9	10	↑	SQL	1.64%	+0.11%
10	16	⇈	Swift	1.27%	+0.20%

Universidades



- En las universidades hace años se enseñaba a programar con Pascal y C.
- En la década de los 90 con Java
- Y ahora aunque se sigue utilizando Java y C, mayoritariamente se empieza con el lenguaje Python

Empresas

- Mucha formación en Java se sustituye por Python.
- Mucha necesidad de análisis para la toma de decisiones.
- Aparecen nuevos perfiles de:
  - Científico de datos
  - Se empiezan a solicitar cursos de Machine Learning
  - El Big Data se incorpora a la empresa y es preciso desarrollar herramientas para explotar el gran volumen de datos.

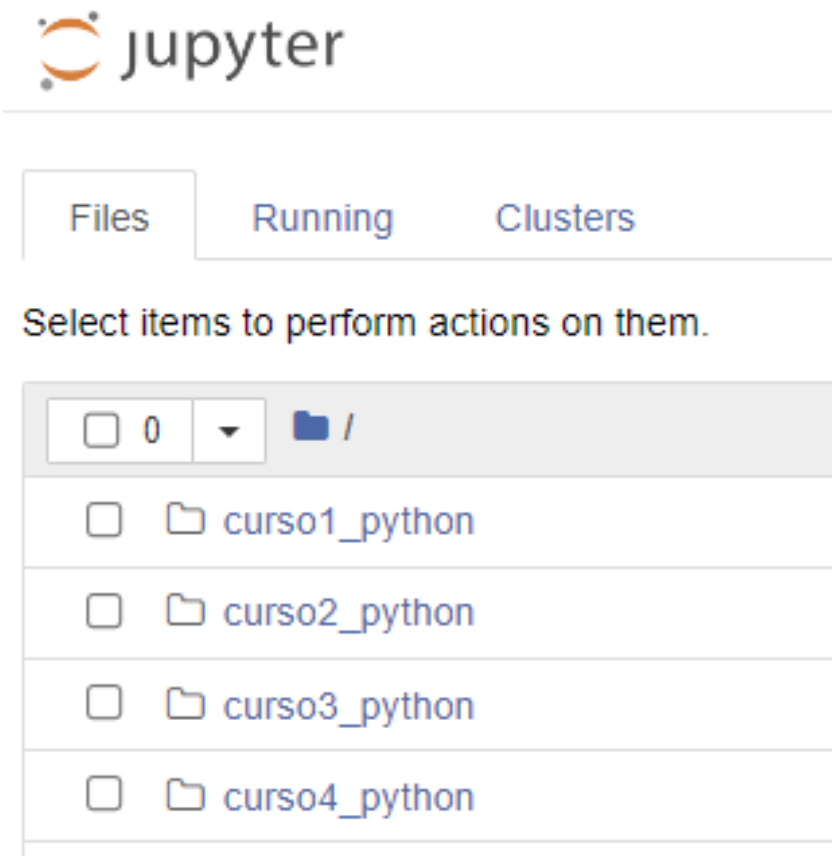
Herramientas para trabajar con Python

- Hay multitud de entornos para trabajar con Python, los más habituales:
  - **Visual Studio Code**: https://code.visualstudio.com/download
    - Es multi lenguaje. Se complementa con Plugins: para Python → **Pylance**
    - Se puede utilizar para Java, PHP, HTML5+CSS3, JavaScript …
    - No integra el compilador, en el caso de Python hay que tener instalado el interprete.
  - **Spyder**: https://www.spyder-ide.org/
    - Dentro de la honda del Code.
  - **Jupyter**: Es una librería de Python. Se instala desde la consola con:
    - **pip install jupyter**
    - Arranca un servidor local y permite ejecutar código Python desde una página HTML, donde es posible incrustar celdas de código, fotos y gráficos.

Jupyter

- Ideal para análisis de datos, permite documentar el proceso, incrustar las gráficas generadas.
- Visualizar el contenido de ficheros CSV, o Excel (con pandas) con estilos de CSS.

- Se arranca desde la consola una vez instalado con:
- **jupyter notebook**

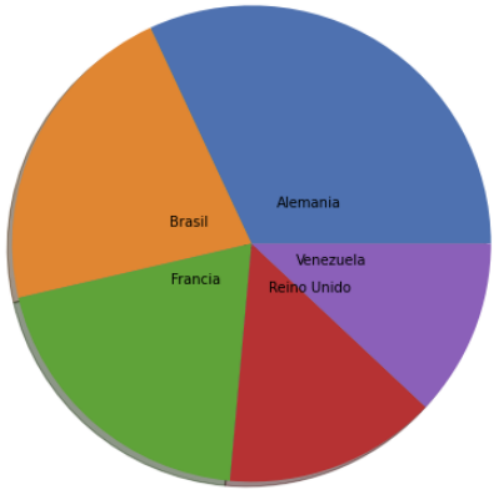


Jupyter

- Combinar celdas de código, visualización de datos, gráficas.
- Permite ejecutar celdas de forma independiente sin tener que ejecutar todo el notebook.
- Se pueden incrustar títulos, celdas de texto, etc. El proceso de análisis puede quedar perfectamente documentado.

	pais	cuenta
0	Alemania	122
1	Brasil	83
2	Francia	76
3	Reino Unido	55
4	Venezuela	46

```
In [24]: plt.figure(figsize=(15,8))
         plt.pie(df.cuenta, labels=df.pais, labeldistance=0.2, shadow=True)
         plt.show();
```



Herramientas para trabajar con Python

- **IDEs** más pesados para grandes proyectos:
  - **Eclipse + Plugin PyDev**
    - Eclipse requiere que esté instalado Java:
      - https://www.oracle.com/java/technologies/downloads/
    - Descargar el instalador de Eclipse:
      - https://www.eclipse.org/downloads/
    - Y configurar el plugin:
      - http://blog.enriqueoriol.com/2015/02/pydev-eclipse-como-ide-de-python-o.html
  - **PyCharm**:
    - Con dos versiones: Profesional y el de la Comunidad
    - https://www.jetbrains.com/pycharm/download/#section=windows

FrameWorks

- A parte de los IDEs podemos instalar frameWorks.
- Un **frameWork** es un conjunto de librerías y utilidades ya configuradas que nos permiten desarrollar aplicaciones en Python (o en otro lenguaje).
- El frameWork por excelencia en el mundo Python es **Anaconda**.
  - https://www.anaconda.com/products/distribution
  - Existe una versión más reducida: **miniConda**
    - https://docs.conda.io/en/latest/miniconda.html
- Anaconda NO es un IDE para desarrollo, nos proporciona un amplio conjunto de librerías, después elegiremos el IDE.
- Viene con su propio intérprete ya instalado y unas **1500** librerías incluidas para **Data Science**.

Anaconda

- Anaconda es 100% compatible con la distribución de Python instalada.
- Podemos tener varios interpretes instalados en la misma máquina siendo uno de ellos Anaconda.

- Python tiene en cuenta la posibilidad de estar desarrollando varios proyectos que necesiten distintas versiones de librerías o incluso del interprete.
  - Reciben el nombre de **ENTORNO VIRTUALES**
    - https://docs.python.org/es/3/tutorial/venv.html

  - Es el concepto de desarrollo en **SandBox**, cada proyecto se le asignan unas versiones de librerías y son independientes de la instalación global de la máquina.

Casos de éxito en Python

- **Pinterest**
  - Red social enfocada a imágenes
  - https://www.pinterest.es/
- **Panda 3D**
  - Motor videojuegos de Walt Disney
  - https://www.panda3d.org/
- **Dropbox**
  - Almacenamiento en la Nube
  - https://www.dropbox.com/
  - Espacio asignado de forma gratuita con una cuenta en Gmail (15 Gb), con posibilidad de contratar más espacio.
- **Spotify**
  - Proveedor de música bajo Streaming.
  - https://www.spotify.com/
- **Netflix**
  - Proveedor de contenidos para TV.
  - https://www.netflix.com/es/

Casos de éxito en Python II

- **Uber**
  - Empresa de movilidad a nivel mundial.
  - https://www.uber.com/es/es-es/
- **Instagram**
  - Red social orientada a imagen / video
  - https://www.instagram.com/?hl=es
- **Reddit**
  - Reddit es un sitio de entretenimiento, redes sociales y noticias que presenta una de las más grandes comunidades en la web.
  - https://www.reddit.com/
- **Google**
  - Buscador de internet
  - Utiliza Python y C++ en los algoritmos de búsqueda.
  - www.google.es
- **YouTube**
  - Plataforma de Videos
  - https://www.youtube.com/

Casos de éxito en Python

- ***Mas detallado en:***
  - https://educacionprofesional.ing.uc.cl/10-programas-en-python-que-serian-estas-grandes-marcas-sin-este-lenguaje-de-programacion/

R competencia de Python

- Diferencias entre **R** y **Python**
- Las principales diferencias son: **R** es un lenguaje orientado al análisis estadístico que se utiliza ampliamente en el campo de la ciencia de datos, añade también paquetes para Machine Learning.

- **Python** es un lenguaje de alto nivel multipropósito utilizado además en otros campos (desarrollo web, scripting

Campos de Aplicación

- Python se puede aplicar a multitud de campos que van desde:
  - **Análisis de datos**
  - **Aplicaciones distribuidas / Web**
  - **Big Data**
  - **Machine Learning**
  - **Deep Learning**
  - Todo tipo de aplicaciones de alto nivel.
  - Soporte para microcontroladores → **MicroPython** → **IoT (*Internet de las Cosas*)**

- Ojo, soporte para móviles → NO
  - Aunque si podemos instalar apps con el interprete de Python.
  - https://play.google.com/store/apps/details?id=innovationlabs.python.com&hl=es_CL

- Aplicaciones a bajo nivel: desarrollo de drivers, S.O., compiladores → Lenguaje C

Análisis de datos

Librerías

Análisis de datos

- Para el análisis de datos disponemos de las siguientes librerías:
  - numpy
  - pandas
  - matplotlib

  - Estas 3 herramientas se integran perfectamente. Los tipos de datos de pandas se pueden representar gráficamente o aplicar operaciones de numpy sin necesidad de bucles.

  - Otras:
    - statsmodels Funciones y modelos estadísticos. https://www.statsmodels.org/stable/index.html
    - scipy Cálculo científico: https://scipy.org/
    - seaborn Gráficas. https://seaborn.pydata.org/
    - numba Optimizador de código Python. https://numba.pydata.org/

Numpy

- Instalación: ***pip install numpy***
- https://numpy.org/

- Librería enfocada al cálculo matemático altamente optimizada (muy rápida)
  - Rebaja el tiempo de ejecución de un algoritmo en Python a utilizar numpy del orden de 10 veces!

- Proporciona un conjunto de operaciones a nivel de array multidimensional.
- Operadores para trabajar directamente con vectores.
- Calcular funciones estadísticas
- Correlación entre variables
- Operaciones a nivel de Arrays (dos columnas entre si).

- Previamente cargando un fichero de Excel o CSV con pandas se pueden aplicar estas operaciones a una columna o fila del libro.

Pandas

- Instalación: ***pip install pandas***
- https://pandas.pydata.org/

- Es **la herramienta estrella** para trabajar con datos en forma tabular:
  - CSV
  - Excel
  - HTML
  - Etc.

- Carga ficheros en memoria para su edición de numerosos formatos
- Después del tratamiento se pueden exportar a esos formatos.

Pandas características

- Principales operaciones:
  - Renombrar columnas, eliminar o seleccionar las columnas que nos interesan.
  - Filtrar datos.
  - Añadir nuevas columnas calculadas.(sin necesidad de bucles).
  - Fusionar ficheros por una columna en común.
  - Anexar filas de un fichero en otro.
  - Exportar a otros formatos.
  - Resumen estadístico de los datos
  - Agrupar datos por una o varias columnas y realizar cálculos.

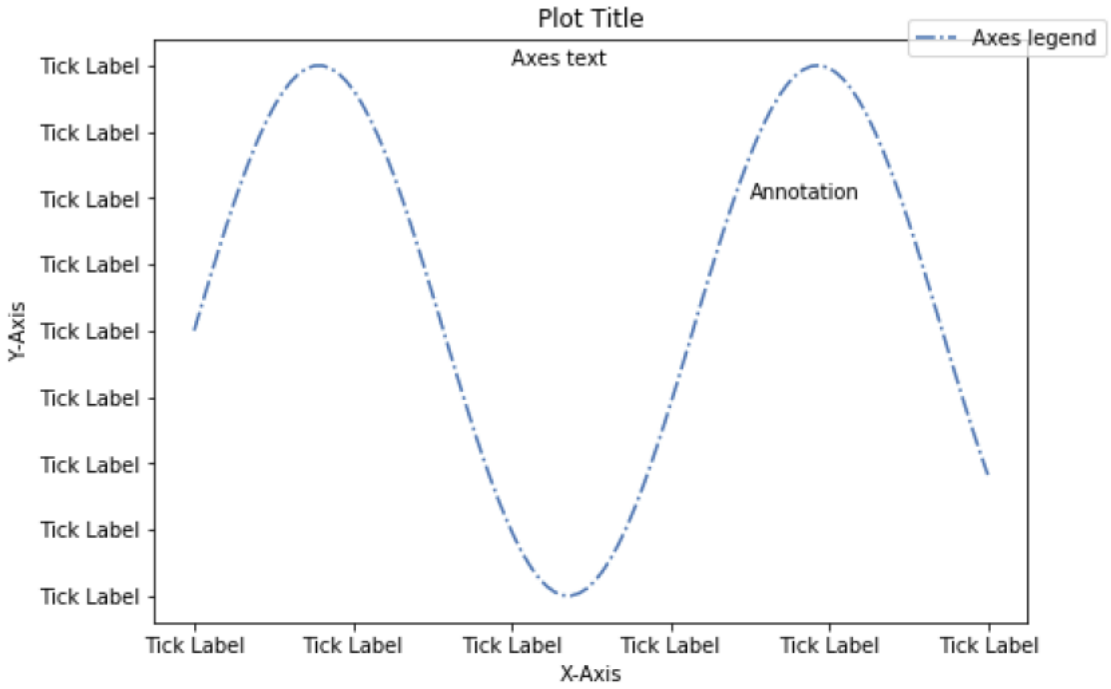
Matplotlib

- Instalación: ***pip install matplotlib***
- https://matplotlib.org/

- Permite representan gráficamente los datos.
  - Soporte distintos tipos de datos
- Proporciona unos 12 o 13 tipos de gráficos distintos.
- La librería proporciona una función distinta para cada tipo de gráfica:
  - De líneas → plot()
  - De puntos → scatter()
  - Tipo tarta → pie()
  - Histogramas → histo()
  - Etc.

Estructura del gráfico

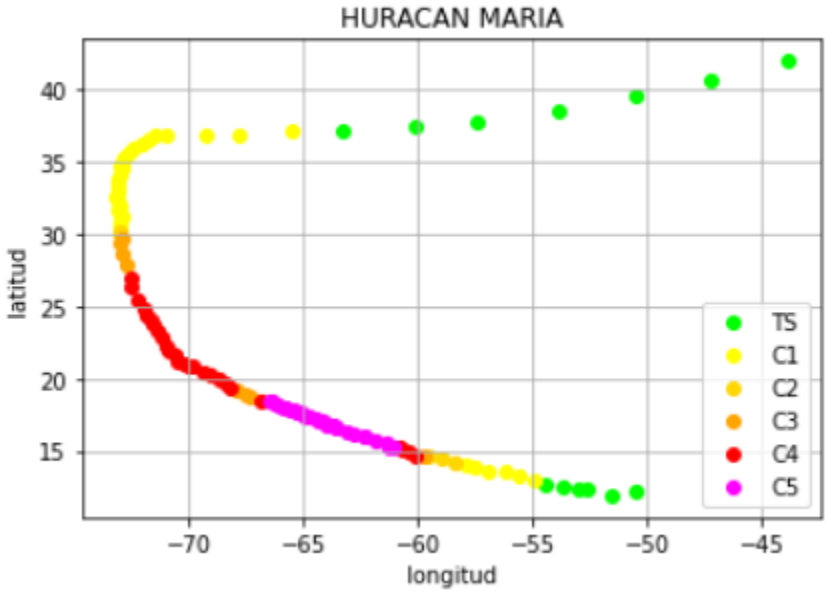
- Cada parte del gráfico es configurable:
  - Título
  - Etiquetas
  - Leyenda
  - Anotaciones
  - …



Ejemplo: trayectoria de un huracán indicando la categoría según escala: Saffir-Simpson

	FECHA	HORA	LATITUD	LONGITUD	NUDOS	PRESION
0	sep-16	15:00 GMT	12.2	-50.5	35	1008
1	sep-16	18:00 GMT	11.9	-51.6	35	1006
2	sep-16	21:00 GMT	12.3	-52.6	50	1002
3	sep-17	00:00 GMT	12.4	-53.0	50	1002
4	sep-17	03:00 GMT	12.5	-53.7	50	1002
...	...	...	...	...	...	...
115	sep-29	21:00 GMT	37.8	-57.4	60	988
116	sep-30	03:00 GMT	38.6	-53.9	60	988
117	sep-30	09:00 GMT	39.6	-50.5	60	988
118	sep-30	15:00 GMT	40.7	-47.2	60	989
119	sep-30	21:00 GMT	42.0	-43.9	50	991

120 rows × 6 columns



Carga datos de Excel → Selecciona, trasforma datos → Genera gráfica → Exporta a PDF

Aplicaciones Distribuidas

Aplicaciones Web

Flask

Django

Aplicaciones distribuidas. Servicios REST

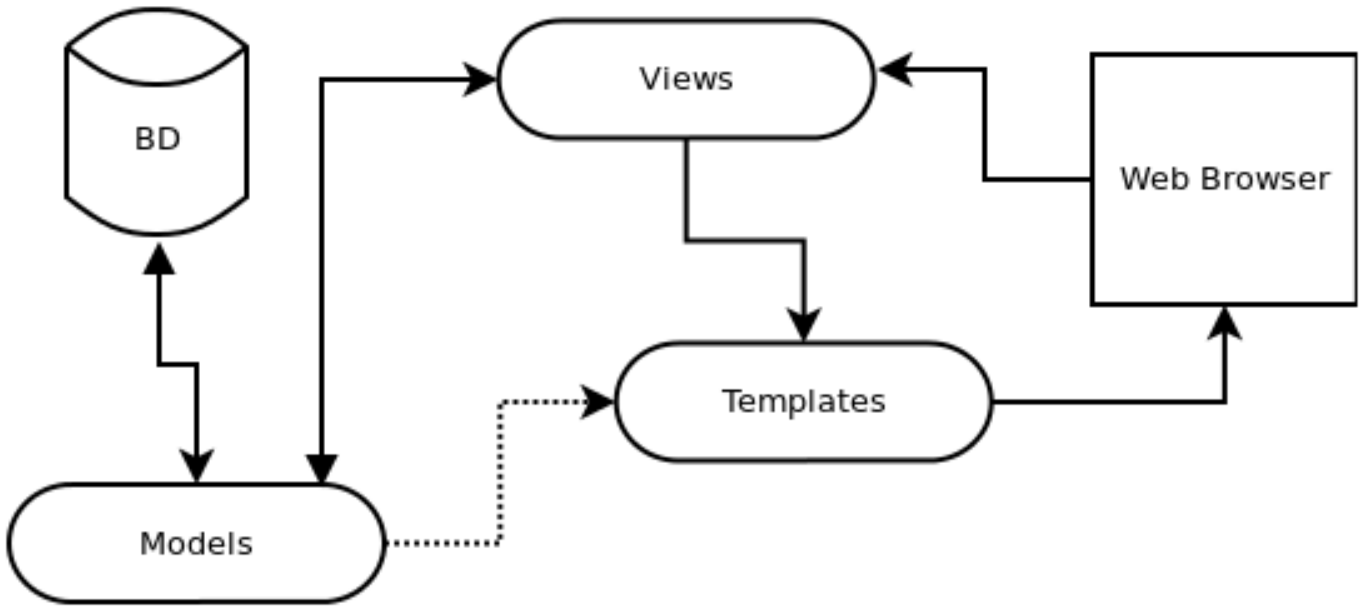
- Servicios Web con **Flask**: https://flask.palletsprojects.com/en/2.2.x/
- Instalación: ***pip install Flask***

- Soporte para el desarrollo de servicios REST.
- Implementación de APIs de REST con operaciones: HTTP: get, put, post, delete.

- Esta librería se integra con **SQL Alchemy**: https://www.sqlalchemy.org/
  - Instalación: ***pip install SQLAlchemy***
  - Es un **ORM** (Object Relational Mapping)
    - Los ORM mapean el modelo relacional en un modelo de objetos y viceversa.
  - Permite operaciones **CRUD**: create, read, update y delete sobre una base de datos sin hacer referencia al Lenguaje SQL

Django

- Framework Django: ***pip install Django***
- El proyecto Django:
  - Desarrollo rápido de aplicaciones Web de 3 capas.
  - Sigue el patrón: Modelo - Vista - Template



- https://www.djangoproject.com/

Django

- Inspirado en la publicación de noticias.

- Se necesitaba velocidad de publicación y era un mundo muy cambiante.
  - Parte de **back** - Acceso a datos, cálculos, seguridad
  - Parte del **front** - Publicación, vista del cliente.

- En toda aplicación Web tendremos una base de datos.
  - Django se integran con la gran mayoría de base de datos relacionales:
    - Oracle, MySQL, SQLite3, PostGreSQL.

- Se despliega en un Servidor Apache

Django

- Proporciona su propio ORM, permite crear una BD a partir de un modelo de objetos o al revés, partimos de una BD creada y generamos el modelo de objetos.

- Regenera automáticamente la parte del Back ante un cambio en el modelo de objetos: añadir nuevos campos, quitar, etc.
  - Hace mediante comandos la migración del modelo de objetos a la BD.

- El front se puede integrar con **Bootstrap**

- Permite la incrustación de gráficas en un navegador con la librería matplotlib: ya comentada.

- Las plantillas generar la vista para el cliente ya sea en HTML o en otros formatos: PDF, Excel, CSV, etc.

Django

- En la parte privada (Back) del sitio Web proporciona seguridad de acceso y autenticación.

- Se crean automáticamente tablas para usuarios y roles para gestionar los permisos de acceso a la aplicación.

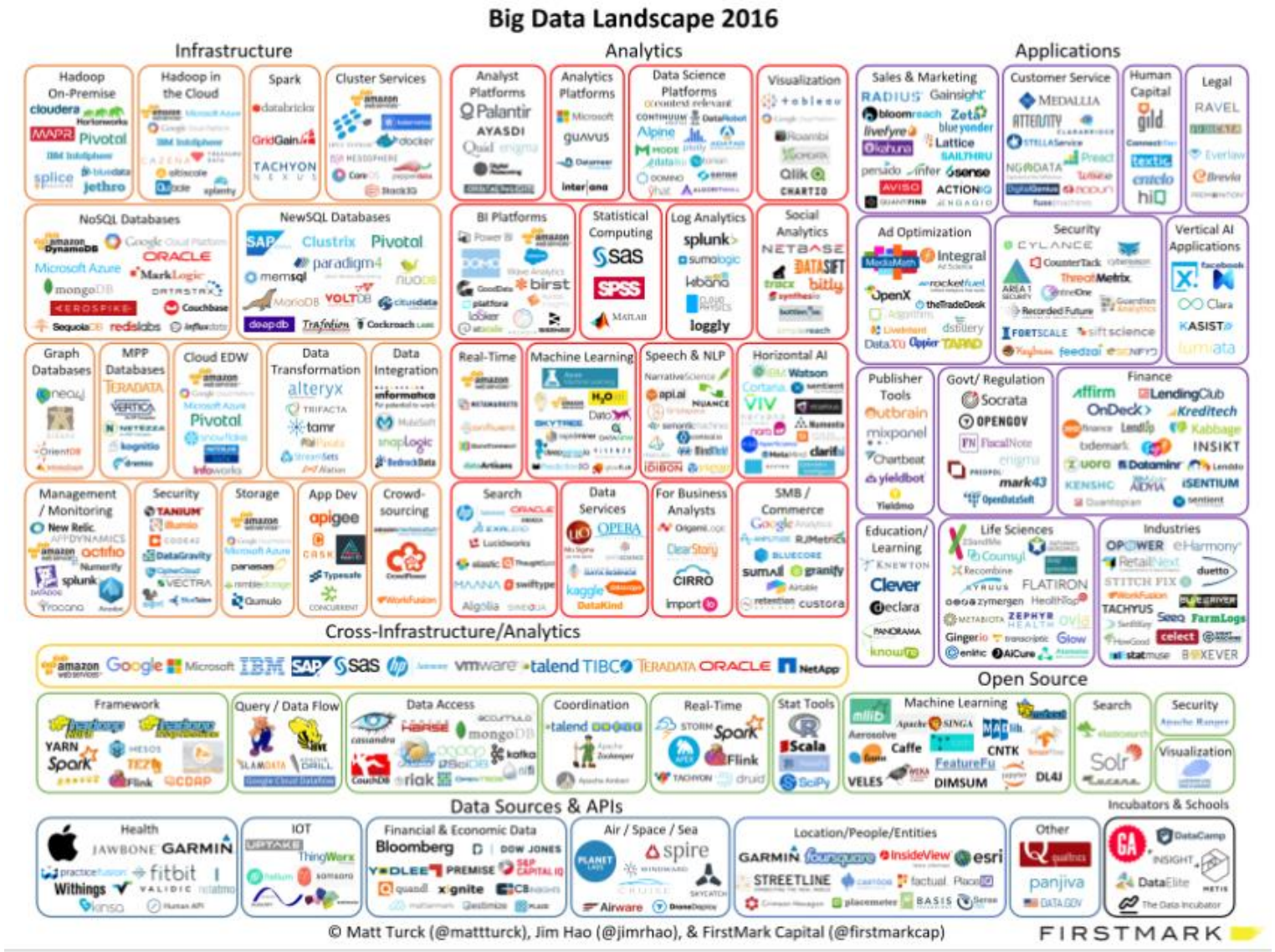
- Se crean permisos automáticamente para crear, borrar, modificar, consultar y se asignan a roles. Posteriormente a cada usuario se le asigna un role.
  - Se pueden crear permisos personalizados: por ejemplo, para mostrar ciertas columnas de una tabla de la BD.

Big Data

Apache Spark

Bases de datos NoSQL

Redis, mongo, etc.



Herramientas Big Data

- Dentro de la multitud de herramientas que tenemos dentro del Big Data se proporcionan APIs de programación en distintos lenguajes:
  - Java, Scala, Python, R

- Herramientas de almacenamiento y explotación de macrodatos como son Apache Hadoop, Apache Spark y bases de datos NoSQL como:
  - Redis: Base de datos clave-valor
  - Mongo: Orientada a documentos en json.
  - Se pueden programar utilidades con Python.

- Campo muy extenso tanto en herramientas como librerías!

Librería para Apache Spark

- Para interactuar con Apache Spark, Python proporciona la librería: **pyspark**
- ***pip install pyspark***

- Esta librería permite trabajar entre otros módulos con:

  - Spark **SQL**:
    - Ejecutar consultas enlazando múltiples ficheros con un lenguaje similar al SQL
  - Datos bajo **Streaming**
    - Procesamiento de datos en tiempo real.
  - **Mlib**
    - Soporte para algoritmos de Machine Learning

  - https://spark.apache.org/docs/latest/api/python/

Librería para redis

- Trabajar con bases de datos NoSQL del **tipo Clave-Valor:**
- https://redis.io/docs/getting-started/

- Utilizada por **Twitter**.
- Python proporciona la librería **redis**

- ***pip install redis***
- Por supuesto es necesario tener instalada la BD redis.
- https://pypi.org/project/redis/

Librería para mongodb

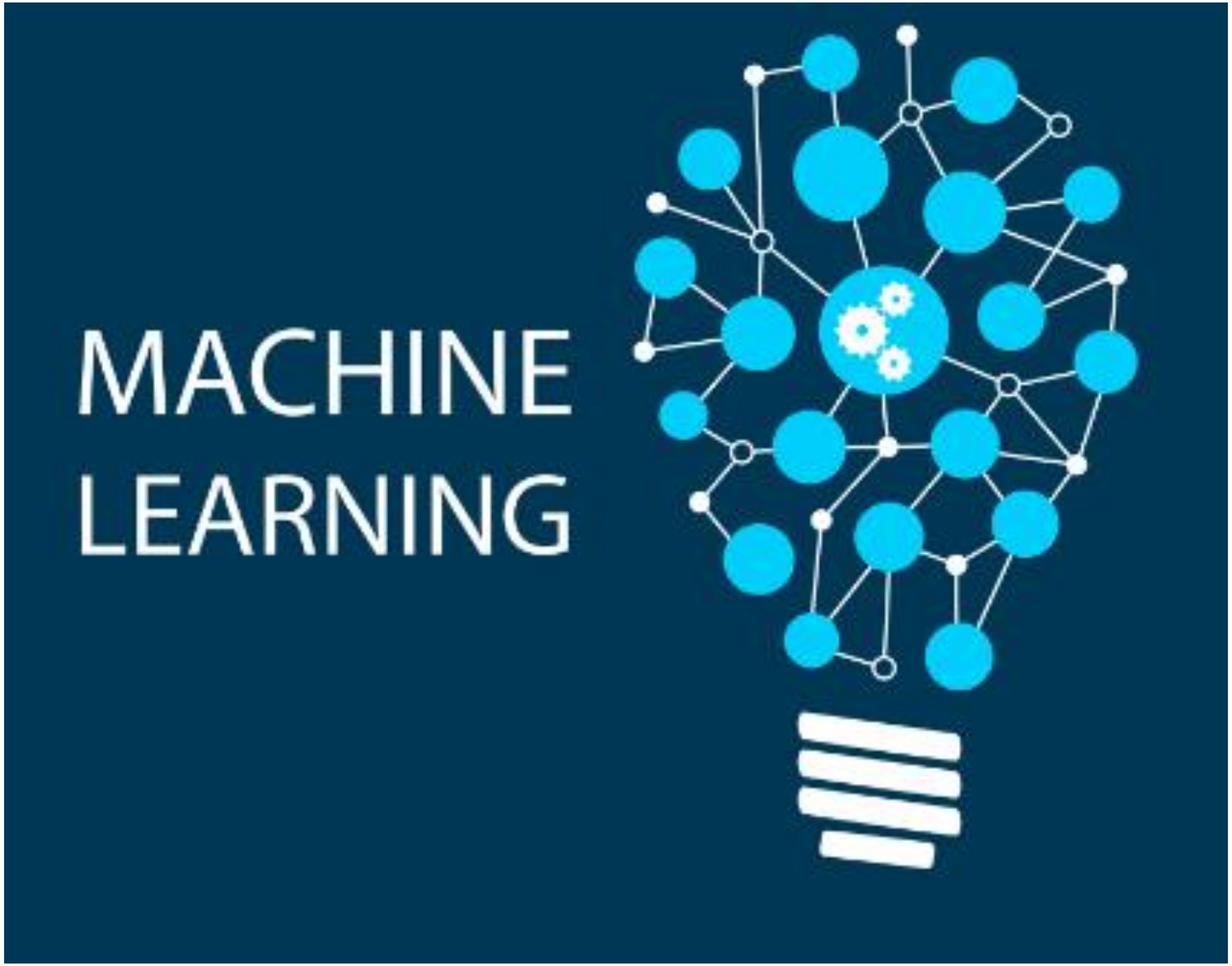
- Base de datos orientada a documentos.
- Datos no estructurados como ocurre con las base de datos relacionales (como MySQL, Oracle, Access …)
- https://www.mongodb.com/languages/python/pymongo-tutorial

- Instalación previa de mongo
  - https://www.mongodb.com/

- Librería: **pymongo**
- ***pip install pymongo***
- https://pypi.org/project/pymongo/
- Para crear base de datos documentales, cargar ficheros en json y explotar los datos mediante consultas Orientadas a Objetos.

Machine Learning

Librería scikit-learn



Machine Learning

- La librería por excelencia que recopila todos los algoritmos de ML es Scikit-learn.
  - Proporciona algoritmos para:
    - Clasificación
    - Regresión
    - Clustering (agrupamiento)

  - A parte de los algoritmos:
    - Proporciona funciones para procesar los datos previamente. Por ejemplo, escalado de datos, estos algoritmos funcionan mejor cuando todos los datos tienen la misma escala, mejor si está entre 0 y 1

- Enlace a la librería: https://scikit-learn.org/stable/
- Instalación: ***pip install scikit-learn***

Tipos de algoritmos

- Dentro de los algoritmos distinguimos entre:

  - **Aprendizaje Supervisado**:
    - Los datos vienen etiquetados.
    - Detección del correo Spam, se proporciona el contenido del correo y una marca que indica
    - **Clasificación** y **Regresión**

  - **Aprendizaje No Supervisado / SemiSupervisado**
    - Datos no etiquetados o sólo una parte vienen etiquetados
    - Buscar semejanzas entre las instancias por ejemplo formado grupos con unas determinadas similitudes que tiene que encontrar el algoritmo.
    - **Agrupamiento, K-means**

  - **Aprendizaje por Refuerzo**
    - Algoritmo de tipo prueba y error que se obtienen recompensas cada vez que nos acercamos a la solución.

Clasificación / Regresión

- **Clasificación**:
  - Identificación de dígitos
  - Diagnósticos
  - Detención de Fraude

- **Regresión**:
  - Calcular el precio de una casa, en base a la zona, etc.
  - Predicciones meteorológicas
  - Expectativa de vida

Algoritmos de Aprendizaje Supervisado

- Árboles de decisión.
- Clasificación de Naïve Bayes.
- Regresión por mínimos cuadrados.
- Regresión Logística.
- Support Vector Machines (SVM).
- Métodos “Ensemble” (Conjuntos de clasificadores).

- ***Definidos dentro de Scikit-Learn***

Algoritmos de Aprendizaje No Supervisado

- Problemas de clustering
- Agrupamientos de co-ocurrencias
- Perfilado o profiling.
- Tienen un carácter exploratorio de los datos, e intentan encontrar patrones que no se encuentran a simple vista.

- Los tipos de algoritmo más habituales en aprendizaje no supervisado son:
  1. Algoritmos de clustering
  2. Análisis de componentes principales
  3. Descomposición en valores singulares (singular value decomposition)
  4. Análisis de componentes principales (Independent Component Analysis)

Etapas de un problema de ML

- Definir el problema: ¿Qué se pretende predecir? ¿De qué datos se dispone? o ¿Qué datos es necesario conseguir?
- Explorar y entender los datos que se van a emplear para crear el modelo.
- Métrica de éxito: definir una forma apropiada de cuantificar cómo de buenos son los resultados obtenidos.
- Preparar la estrategia para evaluar el modelo: separar las observaciones en un conjunto de entrenamiento, un conjunto de validación (o validación cruzada) y un conjunto de test. Es muy importante asegurar que ninguna información del conjunto de test participa en el proceso de entrenamiento del modelo.
- Preprocesar los datos: aplicar las transformaciones necesarias para que los datos puedan ser interpretados por el algoritmo de machine learning seleccionado.
- Ajustar un primer modelo capaz de superar unos resultados mínimos. Por ejemplo, en problemas de clasificación, el mínimo a superar es el porcentaje de la clase mayoritaria (la moda). En un modelo de regresión, la media de la variable respuesta.
- Gradualmente, mejorar el modelo incorporando-creando nuevas variables u optimizando los hiperparámetros.
- Evaluar la capacidad del modelo final con el conjunto de test para tener una estimación de la capacidad que tiene el modelo cuando predice nuevas observaciones.
- Entrenar el modelo final con todos los datos disponibles.

Machine Learning

- **Enlace** (***más detallado***):
  - https://www.cienciadedatos.net/documentos/py06_machine_learning_python_scikitlearn.html

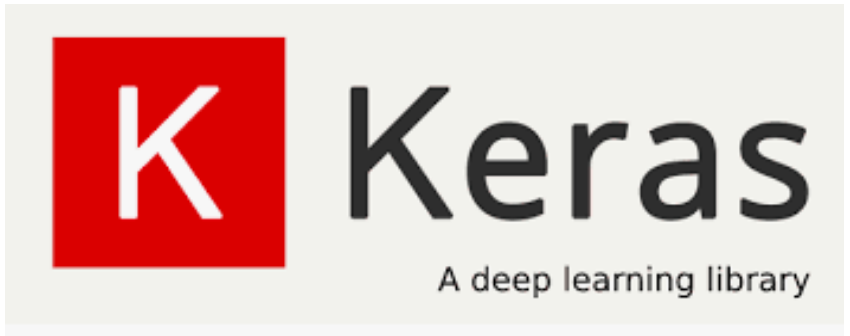
Deep Learning

Librerías: tensorflow, keras

NLTK

Deep Learning

- Dentro del campo del Deep learning Python también proporciona librerías para crear y entrenar redes neuronales.
- Tensorflow
- Keras
- Otras:
  - NLTK



Deep Learning

- **El entrenamiento de redes neuronales tiene dos hándicap:**
- 1) La recopilación de datos para el entrenamiento de la red.
- 2) Potencia de cálculo:
  - Las CPUs se quedan cortas
  - Mejor utilizar **GPUs** → Unidades de procesamiento gráfico
  - Se encuentran disponibles dentro de las tarjetas gráficas permite reducir los entrenamientos de la red del orden de horas e incluso días.

  - PYTHON proporciona la posibilidad de activar la tarjeta gráfica y poder enviar trabajos de computación → disponible en la librería **tensorflow**

Campos de Acción: Python y Deep Learning

- Procesamiento del lenguaje natural
  - Hablado o escrito
- Procesamiento de imagen. Visión artificial.
  - Reconocimiento facial, detención de objetos
- Generación fake news, generación de video.
- Generación de imagen a través del entrenamiento y aprendizaje de redes.
  - Ejemplo de generación de caras que parecen reales con redes GAN (Generative adversarial network) → *El falsificador vs el policía*
  - https://thispersondoesnotexist.com/

Tensorflow

- ***pip install tensorflow***

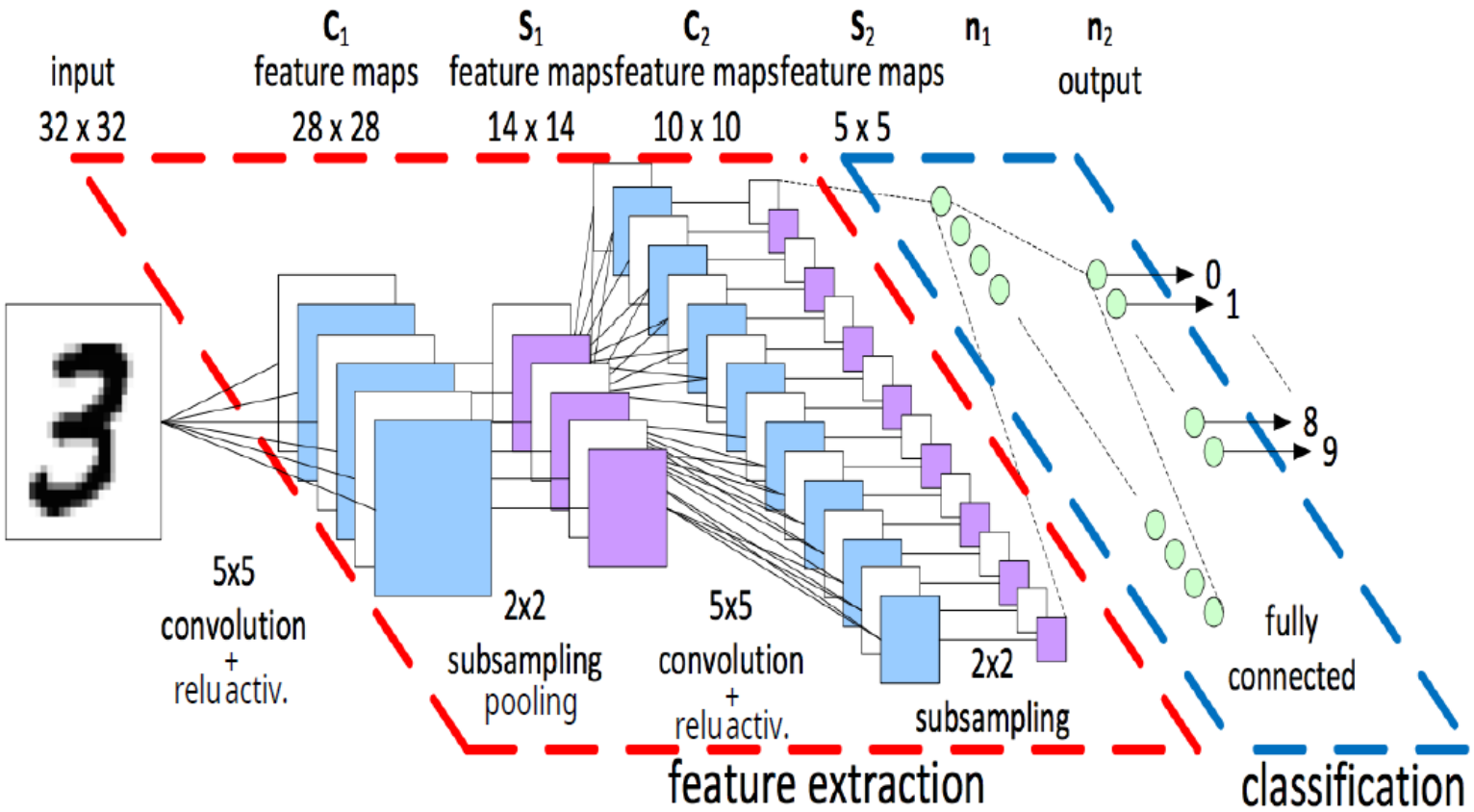
- Tensorflow contiene a la librería Keras.
  - Librería de alto nivel para la implementación / entrenamiento de redes neuronales.
    - Configurar una red capa a capa.

  - Permite extraer características de la imágenes. Todo mediante redes formadas por capas.
  - Reconocimiento de secuencias
  - Procesamiento del lenguaje natural

- Proporcionan principalmente soporte para redes de tipo:
  - CNN: Redes convolucionales → procesamiento de imagen
  - RNN: Redes recurrentes → procesamiento del lenguaje natural
  - GAN: Redes generativas de adversarios → Generar falsificaciones, generar imágenes que no seamos capaces de distinguir si una imagen es real o no.
    - Dos redes enfrentadas compiten entre si.

Como funcionan

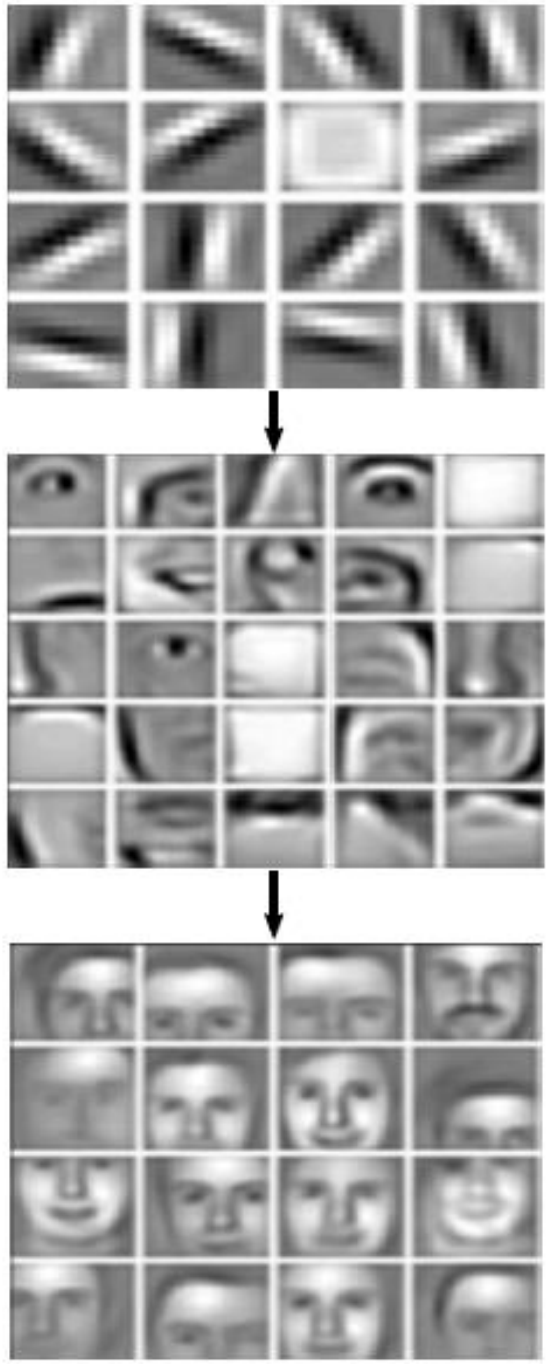
Reconocimiento de números escritos a mano



Como funcionan

Reconocimiento y extracción de características de caras

- Empiezan detectando bordes.
- Partes como una oreja, un ojo
- Hasta reconocer una cara completa.

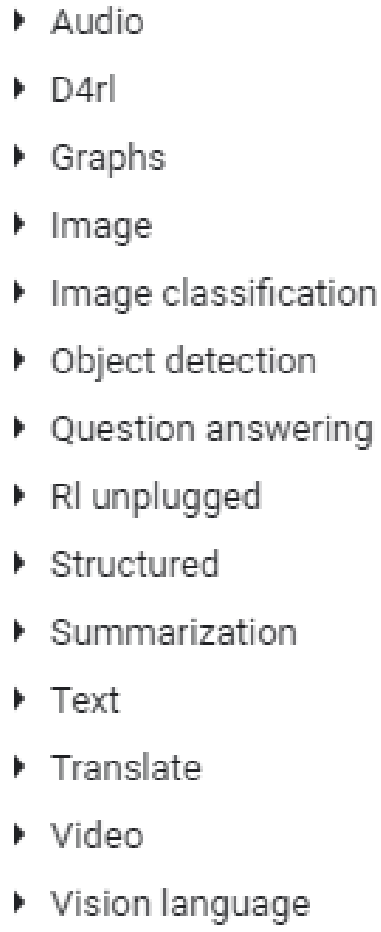


Ejemplos online

- https://www.dlology.com/blog/top-10-deep-learning-experiences-run-on-your-browser/

Tensorflow datasets

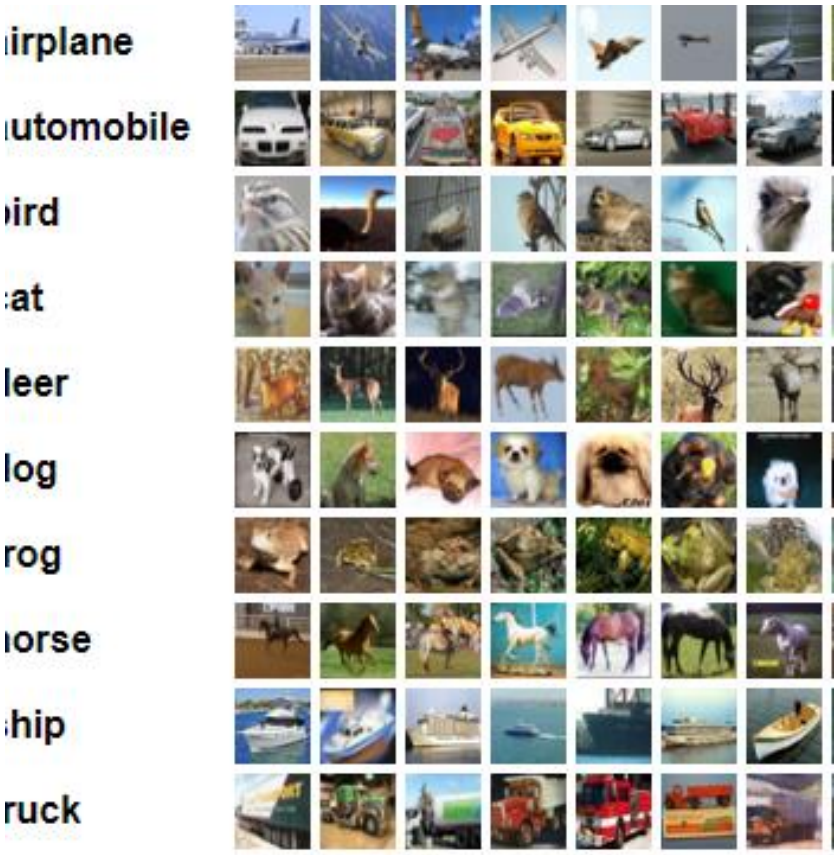
- Otra de las cosas que nos proporciona tensorflow es acceso a **datasets** de imágenes para utilizar en el entrenamiento de nuestras redes.
- Suelen ser catálogos de imágenes previamente clasificadas, o texto, audio, etc.
- https://www.tensorflow.org/datasets/catalog/overview



CIFAR-10 dataset

- Contiene 60.000 imágenes de 32x32 pixels de 10 clases: gatos, perros, coches, pájaros, etc.
- https://www.cs.toronto.edu/~kriz/cifar.html

- Con estos catálogos lo que le venimos a decir a la red esta foto es un perro, esta es un gato, así con un conjunto de entrenamiento, Detecta cuales son las características generales de un perro, de un gato, y luego ante una nueva imagen que la red no ha procesado nos da un % de pertenecer a cada una de estas clases.
- Permite luego evaluar mediante un conjunto de prueba cual es la eficiencia del modelo. De 80% de un 90% …



Keras

- El **API** de **Keras** proporciona un modelo de objetos para la creación de Capas de la red neuronal.
- Funciones de activación.
- Distintos tipos de modelos.
- Optimizadores
- Métricas
- Funciones de pérdida.
- https://keras.io/api/

No son difíciles de programar
La dificultad radica en la cantidad de
Parámetros configurables que tienen
Requiere de mucha experiencia o incluso
hablaríamos de arte para conseguir unos buenos
resultados.
Identificar bien el problema a resolver implicará
Elegir un tipo de red u otro.

NLTK

- https://www.nltk.org/
- ***pip install nltk***
  - A parte luego se descargan los corpus.
- Es un kit de herramientas para el procesamiento del lenguaje natural.
- Procesamiento de texto escrito, la librería proporciona CORPUS previamente anotados en varios idiomas, para luego identificar nombres propios, verbos, etc.
- Se utilizan para traducir textos a diferentes idiomas.
- Clasificación de documentos, por el % de palabras de un determinado tema.

Deep Learning - Google Colab

- Cuando tenemos que entrenar redes y no disponemos de una buena tarjeta gráfica tenemos la opción de trabajar con Google Colab.

- Es un notebook de Jupyter que se ejecuta en remoto y se le puede asociar computación con GPU, nos puede reducir el trabajo en horas.

- Los notebook se almacenan en nuestro **Google Drive**, solo es necesario tener una cuenta en **GMail**

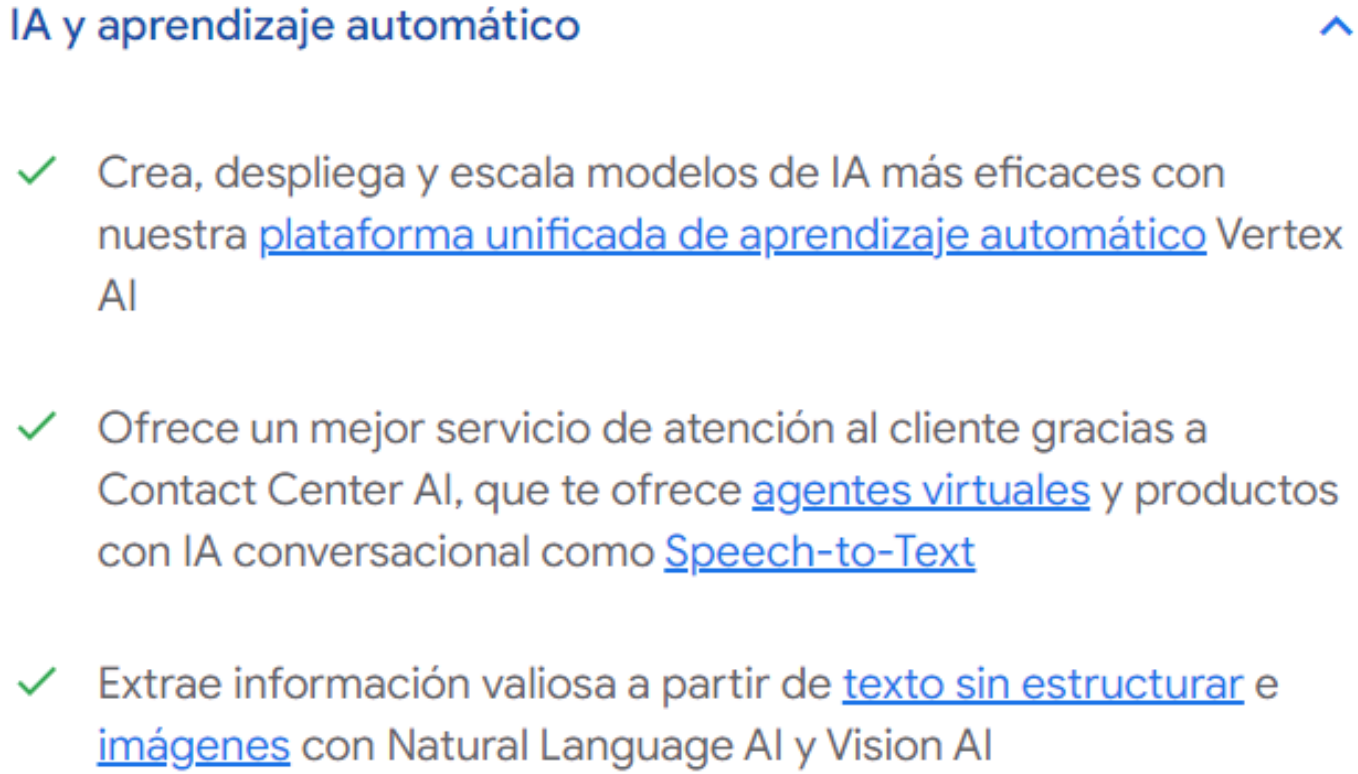
Otras opciones en remoto

- Google Cloud Platform
- AWS: Amazon Web Service
- Microsoft Azure

- Proporcionan alquiler de máquinas con CPU / GPU para computación de alto rendimiento.
- Modelos de Deep Learning ya entrenados para reconocimiento de voz o imagen.

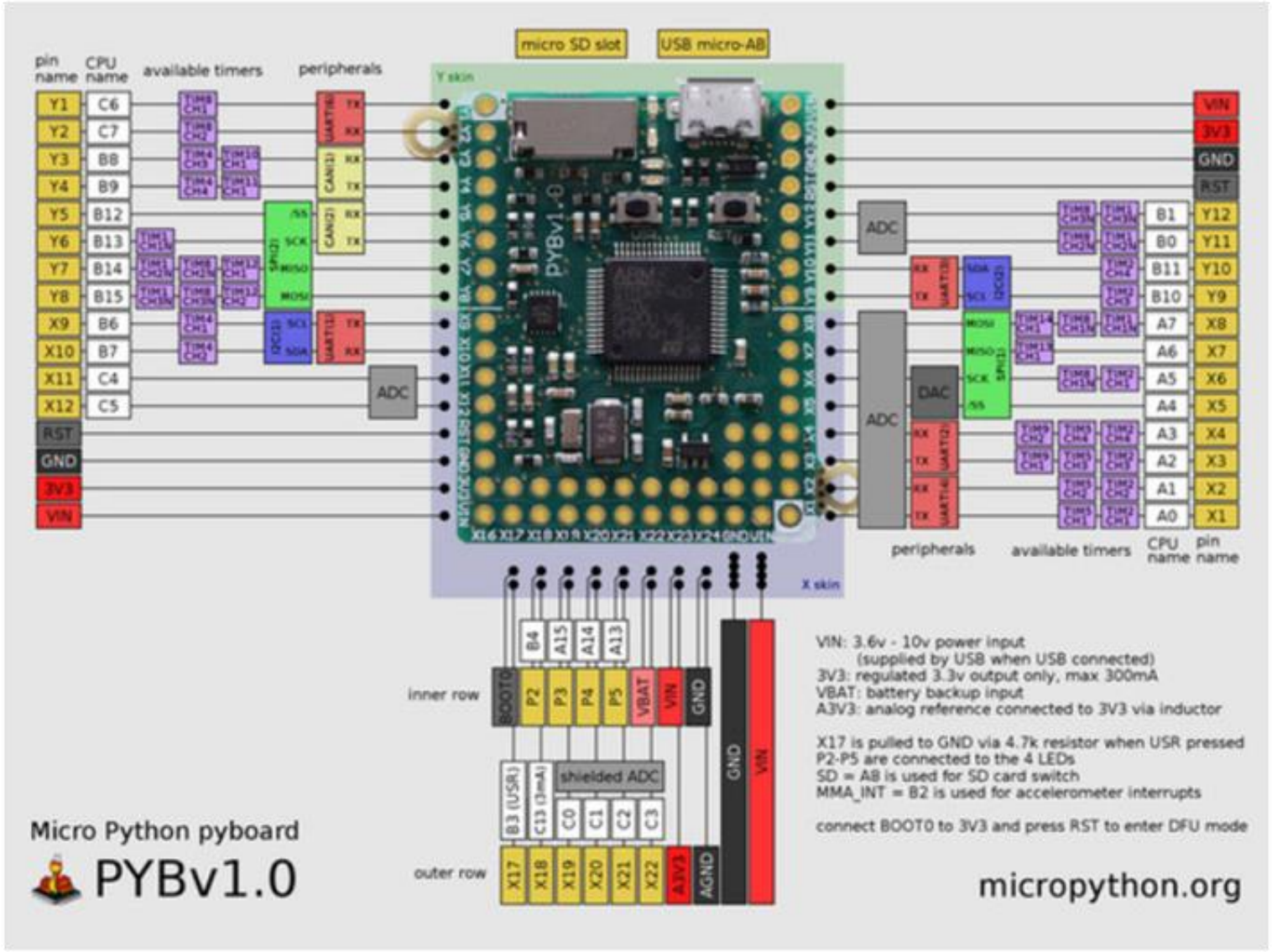
Ejemplo: Google Cloud

- El desarrollo se realiza con Python
- Y proporcionan máquinas donde
- Podemos instalar librerías como tensorflow



MicroPython

https://micropython.org/



MicroPython

- Soporta la programación de microcontroladores como RaspBerry Pi o Arduino → Internet Of Things
- MicroPython:
  - Pequeño pero eficiente intérprete para el lenguaje de programación Python.
  - Librerías enfocadas a interactuar con sensores y comunicaciones de todo tipo:
    - De gas, de humedad, temperatura, de movimiento.
    - Envío de datos: a través de Wifi, GPRS, BlueTooth.
    - Control de movimiento: GPS, Acelerómetros,
    - https://www.raspberrypi.com/documentation/microcontrollers/micropython.html

Raspberry Pi

- Miniordenador con un interprete de Python instalado que permite el desarrollo de multitud de proyectos.
  - Sistemas de seguridad en el hogar, con envío de datos por Telegram…

Raspberry Pi

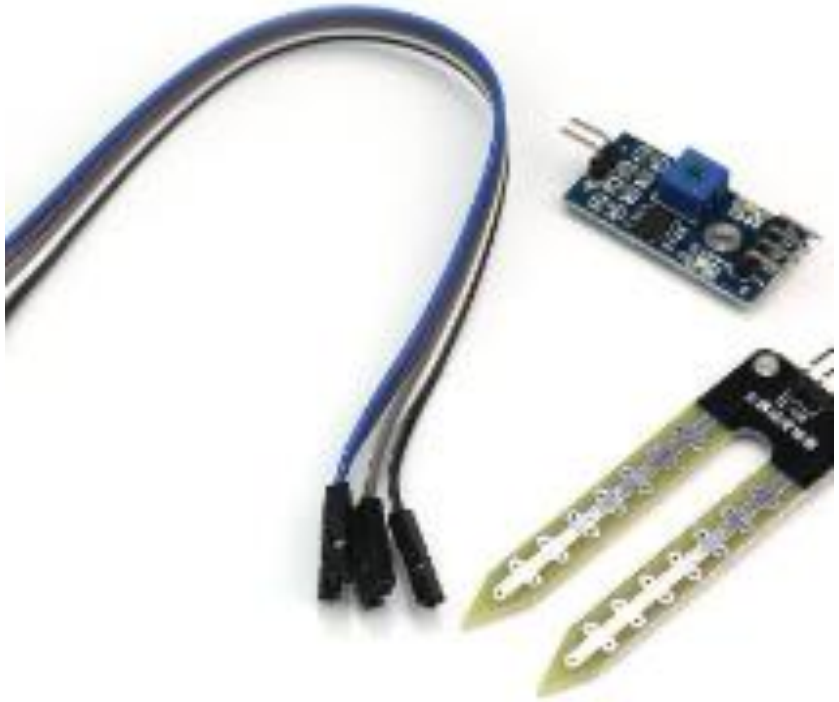
- Servidores multimedia en el hogar o empresa con la integración de múltiples dispositivos: Smart TV, Teléfonos, Tablets …

- **NAS** (Network Attach Solutions) : Copias de seguridad centralizadas.

Raspberry Pi

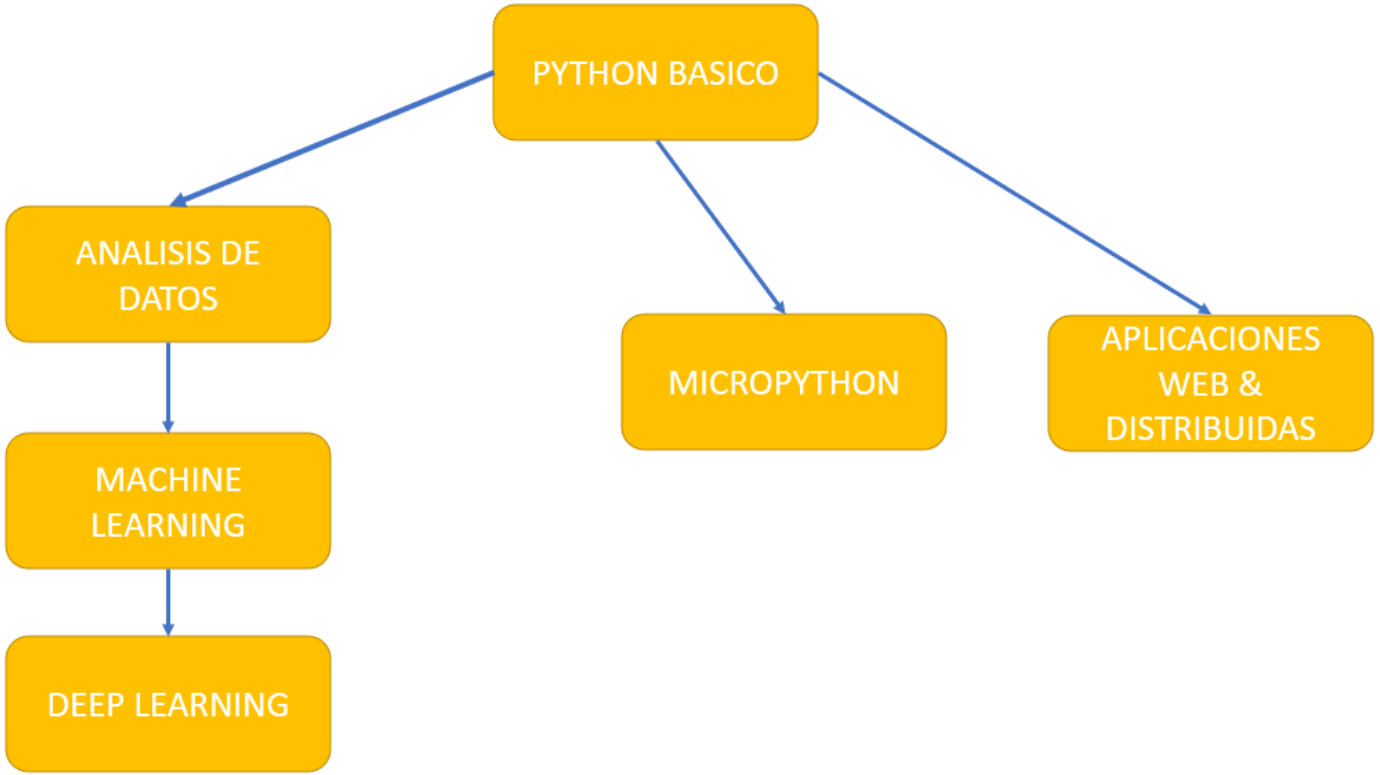
- Sistemas de riego automático / huertos ecológicos

*MicroPython proporciona un **API** (Interface de Programación De Aplicaciones) para interactuar de una forma sencilla con Redes de sensores y multitud de dispositivos conectados a Microcontroladores como Arduino o RaspBerry. Con la potencia y simplicidad del lenguaje **PYTHON***



Disponemos de librerías como **pyboard**

Evolución en el aprendizaje de Python



Fin de la presentación

¿Alguna pregunta?