



# docker

DigitalHouse >  
Coding School

## DATA SCIENCE

MÓDULO 3

**Docker**

1

**Introducir nociones de Docker a nivel usuario.**

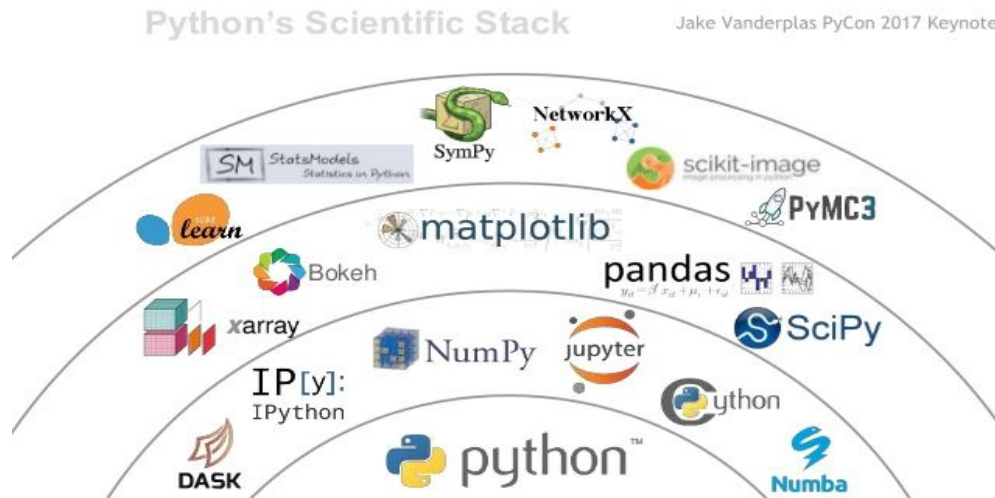
2

**Introducir nociones básicas de Anaconda a nivel usuario.**

# DOCKER



Diferentes programas o aplicaciones requieren múltiples recursos virtuales para funcionar correctamente..



El conjunto de estos recursos podrían no ser compatible entre sí.

Lidiar con la incompatibilidad dentro de los sistemas es bastante común en Data Science dada la cantidad de instancias:

- Estructura de datos
- Recursos Algorítmicos
- Gráficos
- Bases de Datos
- Formas de adquisición de la información
- Entornos de Desarrollo
- Recursos Matemáticos
- Mapas

Muchas formas de realizar una tarea en particular, pero con enfoques y técnicas que podrían diferir.

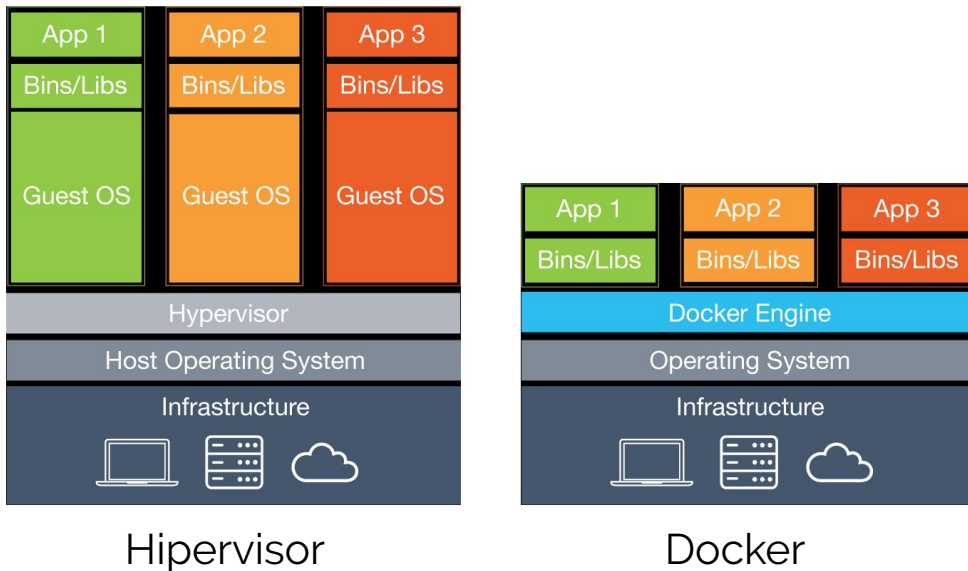
En la virtualización se crean instancias de un entorno que incluyen:

- I/O
- Conjunto de instrucciones de CPU
- Acceso a memoria
- Interrupciones de sistema
- Entre otras

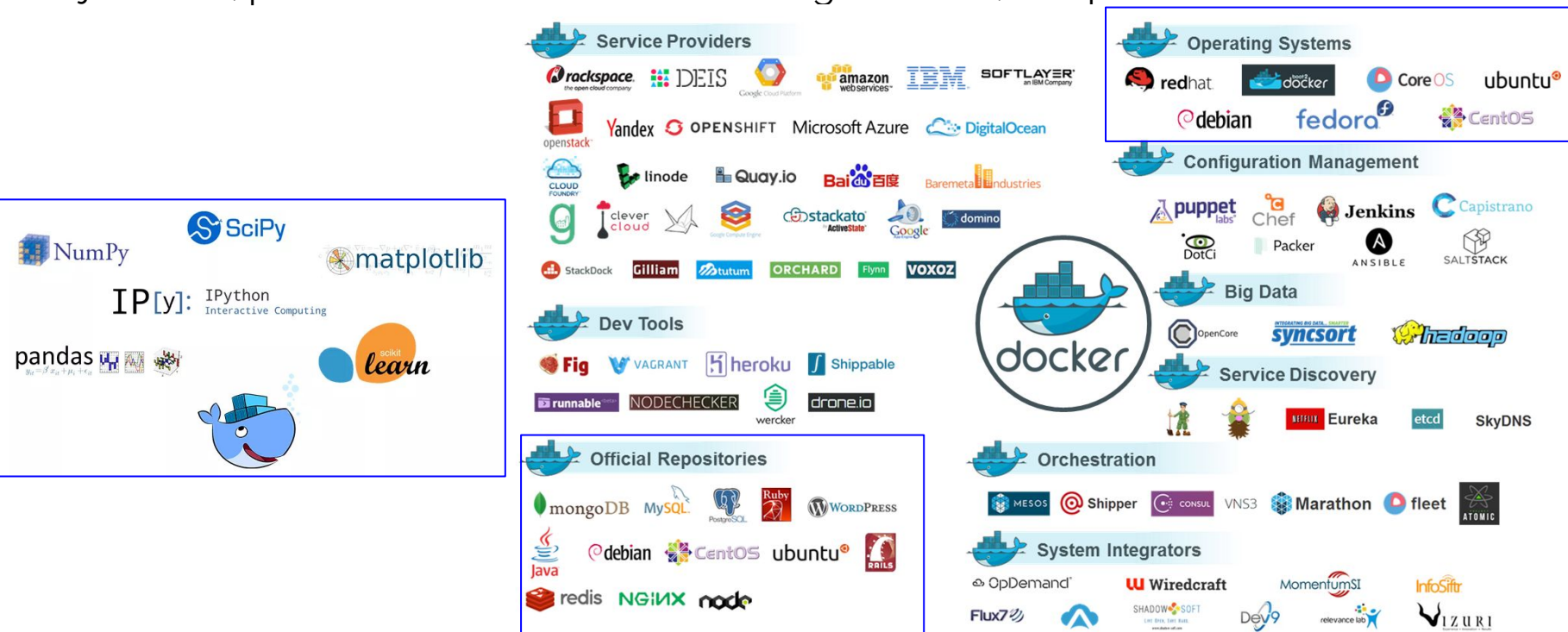
Es diferente a la simulación (que modela el entorno) y a la emulación (que ejecuta algo similar pero diferente).

Docker provee una forma de ejecutar software 'de forma segura', en un entorno aislado, empaquetado y con todas sus dependencias y librerías, esta instancia se conoce como contenedor. Veamos qué son y cómo se crean los contenedores...

Docker provee una forma de ejecutar software de forma segura, en un entorno aislado, empaquetado y con todas sus dependencias y librerías, esta instancia se conoce como contenedor.

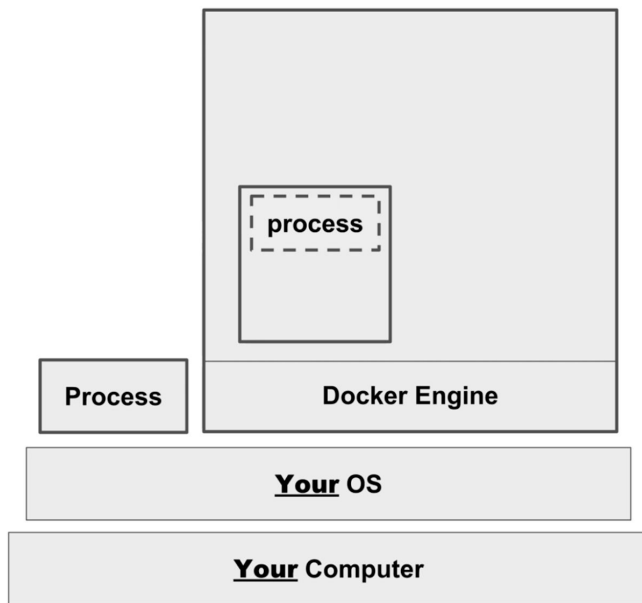


Un contenedor es un paquete liviano y autocontenido, con todo lo necesario para ser ejecutado, por la naturaleza virtual corre de igual forma, independiente del entorno.

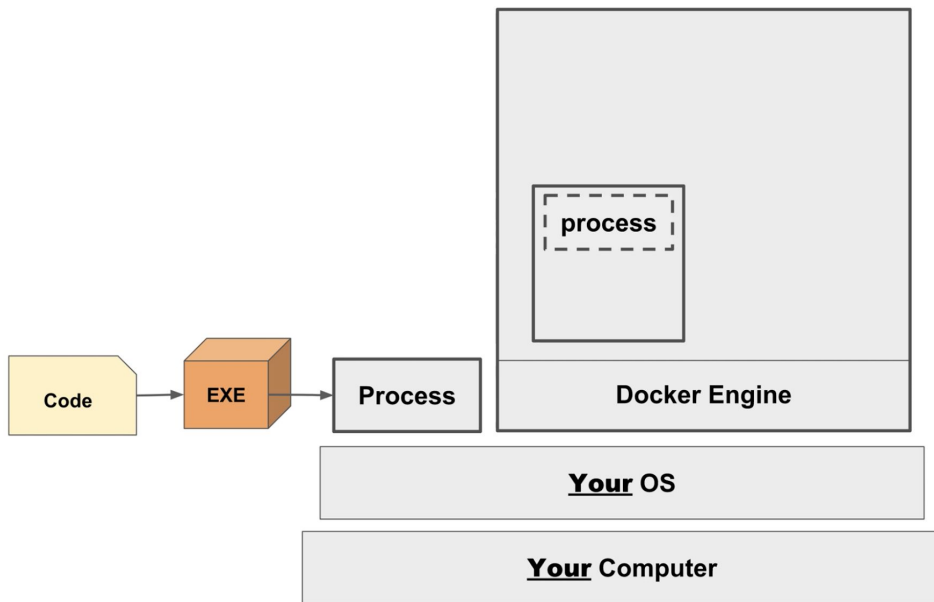




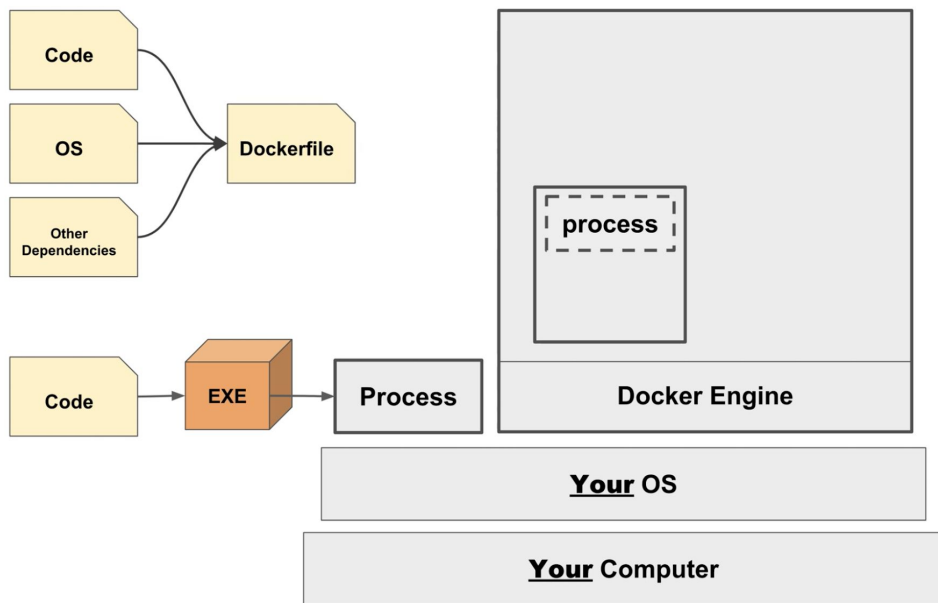
Comparemos como se crean un proceso regular y un contenedor de Docker...



Para el proceso regular empezamos con un código que se compila como un ejecutable que corremos como un proceso en el OS.

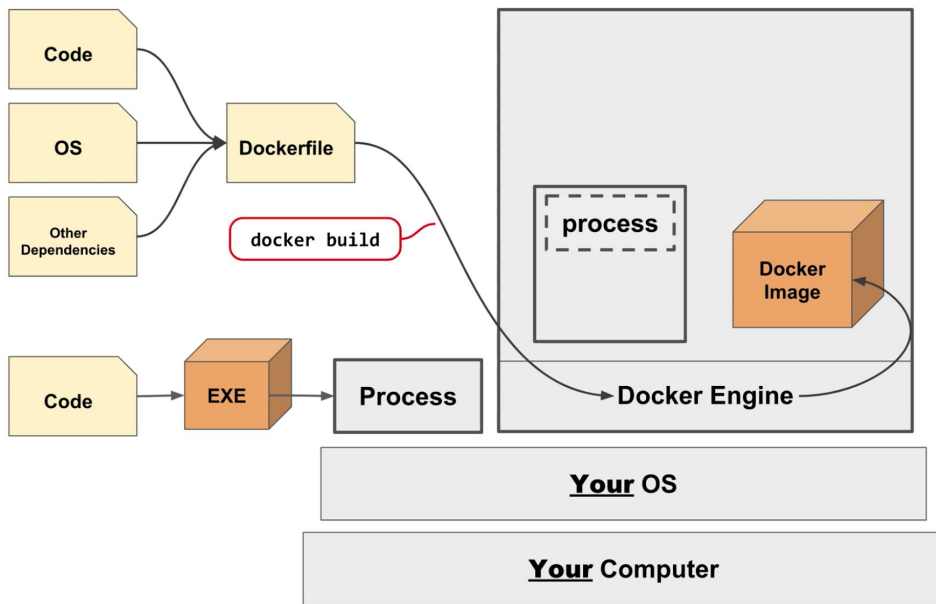


Para el contenedor, empezamos con el código, pero también tenemos agrupar el OS y las demás dependencias y crear un Dockerfile.



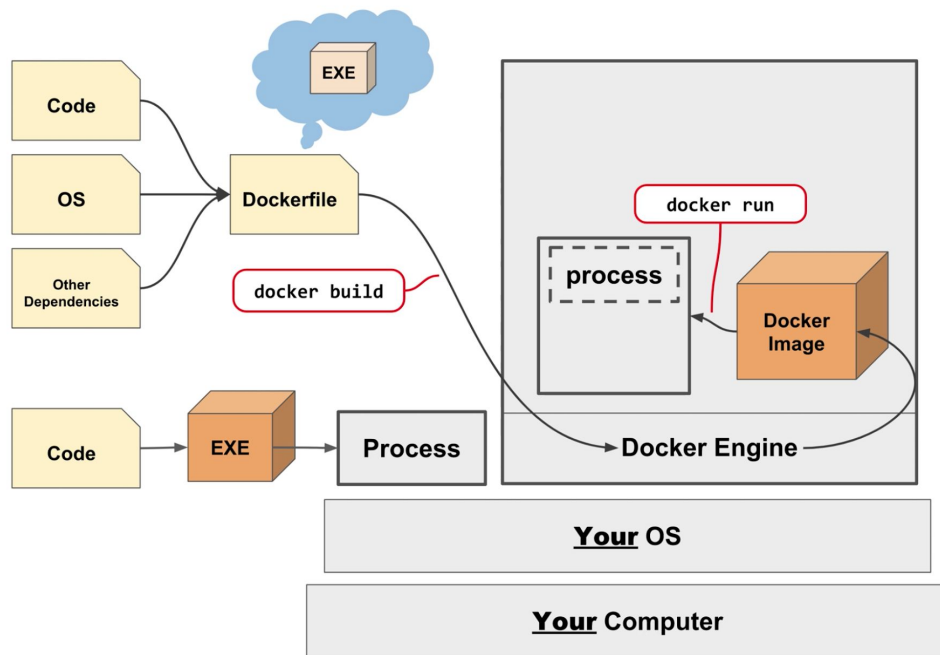
El Dockerfile es una lista de instrucciones que le pasamos al Docker Engine para que construya el contenedor.

Con el comando **docker build** le pasamos el Dockerfile al Docker Engine, que sigue las instrucciones para crear una Imagen de Docker que luego se usa para iniciar un contenedor.



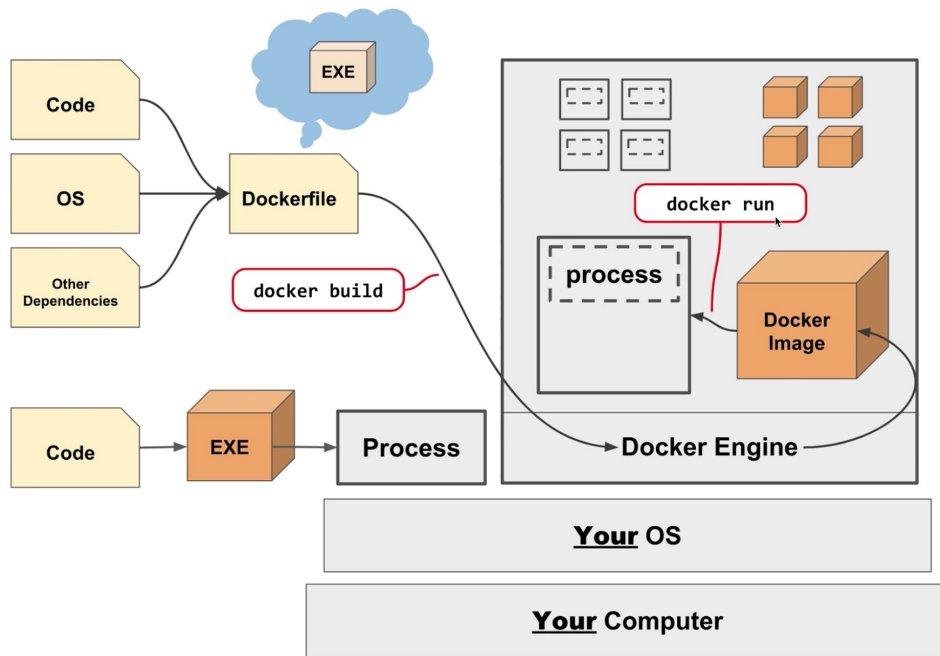
Un contenedor se construye a partir de un molde, este molde se conoce como imagen.

Con la Imagen de Docker iniciamos un contenedor con **docker run**.



Podemos compilar el ejecutable manualmente y pasarlo al Dockerfile o podemos describir como compilar el ejecutable en el Dockerfile como una instrucción.

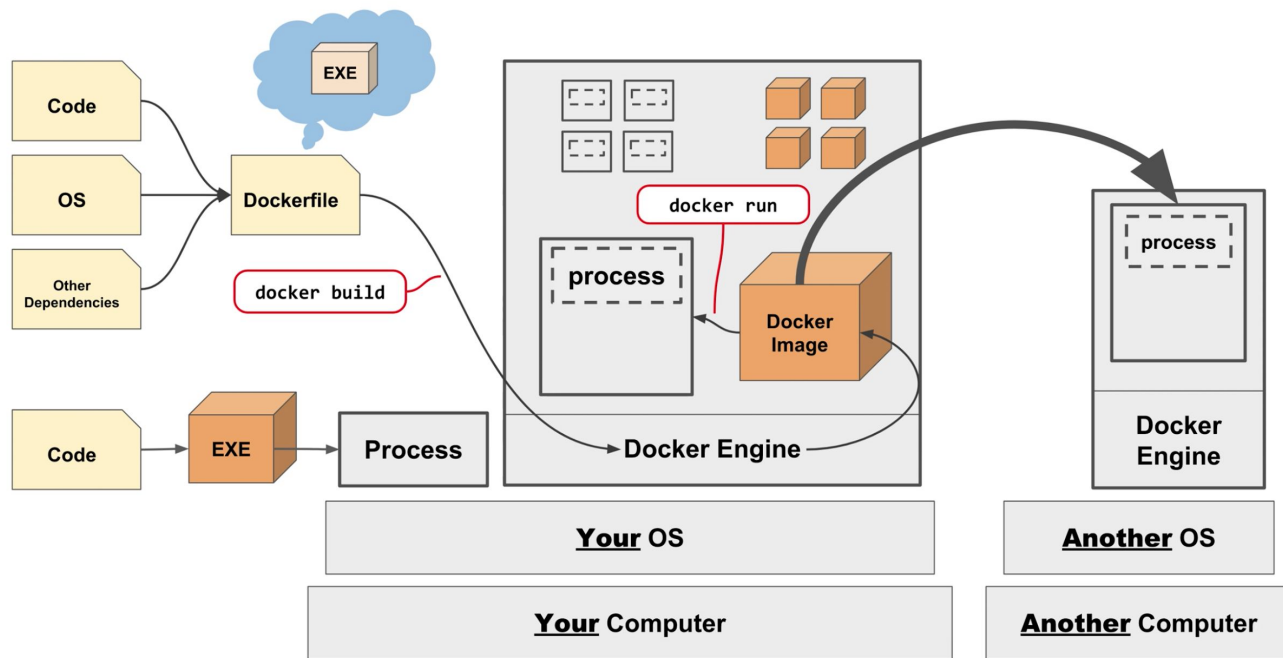
El nuevo contenedor va a correr junto a los demás contenedores que tenga dentro de mi Docker Engine.



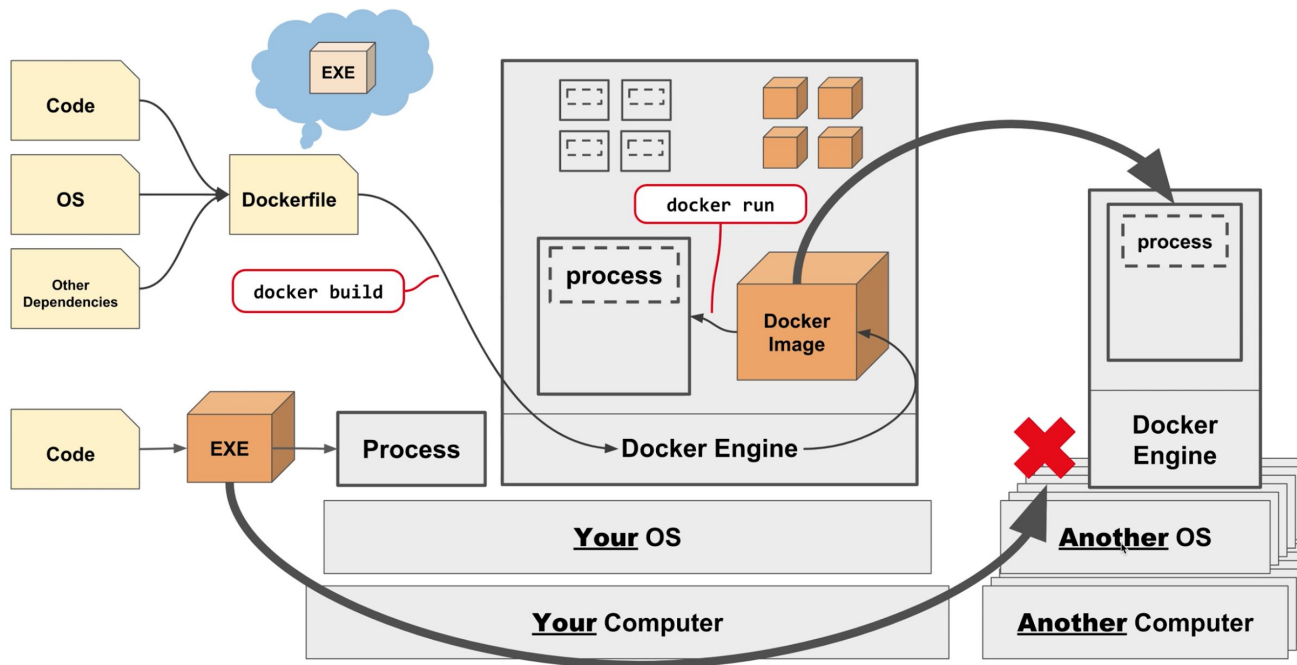
Los contenedores están aislados los unos de los otros: Docker Engine regula cuántos recursos asigna a cada uno.

Cada vez que corremos **docker build** creamos una imagen nueva

Podemos correr una imagen de Docker como un contenedor en otro sistema. Docker nos asegura que este nuevo contenedor va a correr como el original.



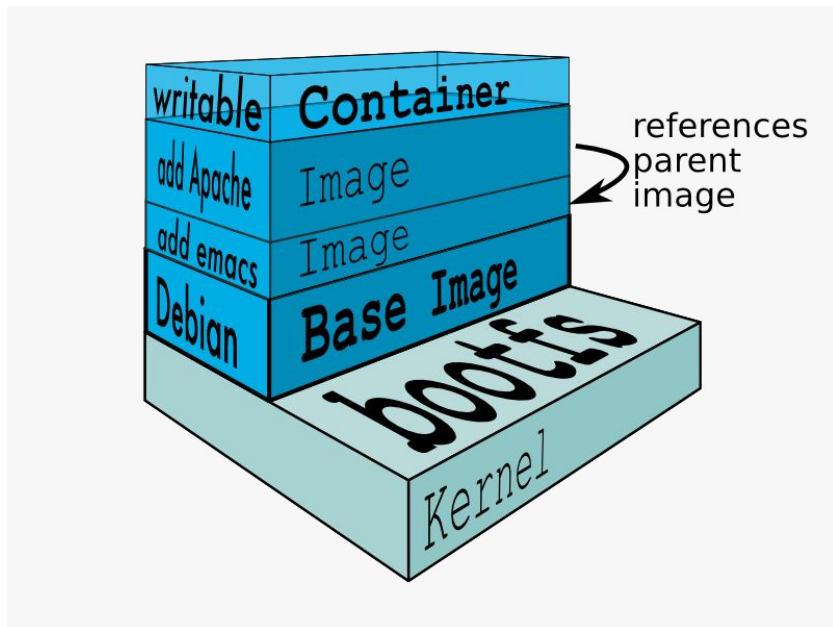
Esta portabilidad permite resolver el problema de compatibilidad entre sistemas que tendríamos si quisiéramos correr el proceso directamente en otro sistema.





Un contenedor se construye a partir de un molde, este molde se conoce como imagen.

Una imagen es una colección de cambios sobre el sistema de archivos.



Una imagen contiene una unión de capas apiladas.

Cada capa hereda de la capa anterior.

Las imágenes son inmutables.

Los resultados intermedios pueden ser cacheados para un uso posterior.

- Docker cuenta con un conjunto de instrucciones para la construcción de imágenes.
- El set instrucciones se suele agrupar en un archivo llamado **Dockerfile**.
- Estas instrucciones son ejecutadas al invocar el comando **build**
- Para la construcción de imágenes es necesario el **Dockerfile** y un contexto de construcción.
- El **contexto** de construcción es una carpeta local, o un repositorio tipo git.
- Este contexto puede contener archivos disponibles para la construcción, si es que es necesario.

Algunas de las instrucciones para la construcción de una imagen en el dockerfile

- FROM: Para llamar a una imagen de base (obligatorio para un dockerfile)
- USER : Seleccionar un usuario con sus respectivos permisos
- RUN : Corre un comando durante la construcción
- COPY : Copia un archivo del contexto
- WORKDIR : Selecciona un directorio dentro del contenedor de construcción.
- CMD : Ejecuta un comando en tiempo de ejecución, en el contenedor final
- ENV : Asigna una variable de entorno
- LABEL : Variable dentro de la imagen durante la construcción

[https://github.com/ds-dh/digital\\_data/blob/master/Dockerfile](https://github.com/ds-dh/digital_data/blob/master/Dockerfile)

# USO DE DOCKER



- **Quickstart Terminal** es una terminal de Docker, contiene el entorno (comandos y herramientas) para realizar las tareas de gestión de los elementos de docker.

- Contenedores
- Imágenes
- Gestión de Virtualización

```
Machine default already exists in VirtualBox.
Starting machine default...
Starting VM...
Started machines may have new IP addresses. You may need to re-run the 'docker-m
achine env' command.
Setting environment variables for machine default...
open C:\Users\30140943\.docker\machine\machines\default\ca.pem: The system cannot find the file specified.
```



```
docker is configured to use the default machine with IP 192.168.99.100
For help getting started, check out the docs at https://docs.docker.com
```

```
3014 W00044786 MINGW64 ~
$ |
```

Comandos útiles para el manejo de contenedores:

**run** : Este comando se utiliza para la creación de un contenedor a partir de una imagen. Una forma de usarlo es:

```
$ docker run [OPCIONES] imagen /comando
```

Las **Opciones** configuran el comportamiento del contenedor, la forma en la que se ejecuta y construye la imagen, e incluso qué pasa con la imagen después de su ejecución.

Dentro de las opciones más comunes dentro de **run** podemos encontrar:

- p : Asigna un puerto de salida al contenedor
- rm : Borra la imagen después de la ejecución de la tarea
- it : Abre una terminal interactiva que se comunica con el interior del contenedor
- user : Nombre de usuario que elegimos dentro del contenedor (tiene que existir dentro del contenedor)
- name : Nombre que elegimos para el contenedor.

Ejemplos:

```
$ docker run hello-world
```

```
$ docker run --rm ubuntu /bin/echo "hola"
```

```
$ docker run -it --name mi_ubuntu ubuntu /bin/bash
```

```
$ docker run -p 8888:8888 jupyter/base-notebook
```



Comandos útiles para el manejo de contenedores:

`ps -a` : Lista los contenedores que hay en la máquina.  
`$ docker ps -a`

`stop` : Detiene un contenedor en ejecución  
`$ docker stop contenedor`

`start` : Ejecuta un contenedor.  
`$ docker start contenedor`

`rm` : Elimina un contenedor (tiene que estar detenido)  
`$ docker rm contenedor`

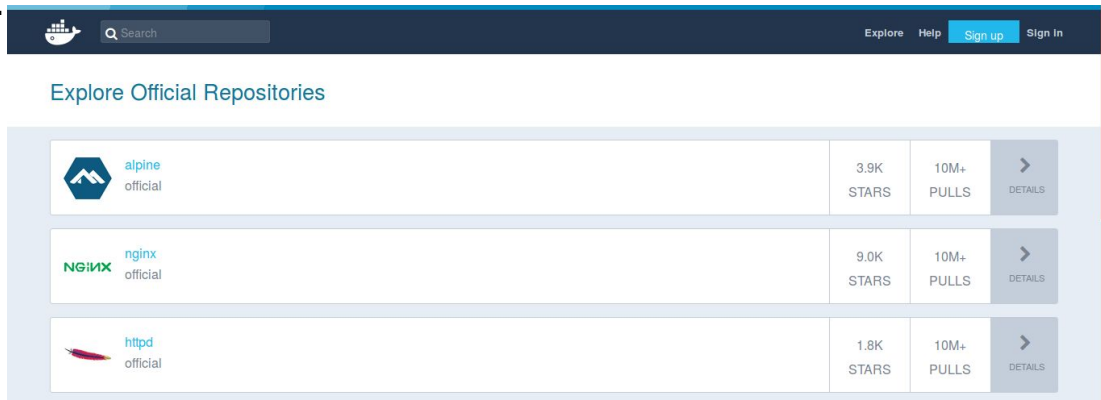
Comandos útiles para el manejo de imágenes:

**images** : Lista las imágenes que hay en la máquina.  
`$ docker images`

**rmi** : Borra una imagen  
`$ docker rmi imagen`

**build** : Construye una imagen en su contexto (en este caso local, notar el punto)  
`$ docker build .`

Docker cuenta con un repositorio en donde se alojan las imágenes desarrolladas.



Puede ser público o privado y es posible subir y descargar una gran variedad de recursos, el repositorio cuenta con imágenes oficiales de los desarrolladores de las soluciones.



# KITEMATIC



Para evitarnos el manejo a nivel de usuario se creó una aplicación gráfica para la gestión y ejecución de Docker: **Kitematic**

Permite buscar las imágenes (está conectado al repositorio docker-hub), levantar, inicializar, detener, y borrar los contenedores, definir variables de entorno del contenedor, establecer sincronizaciones entre el contenedor y el host, configurar redes entre contenedores, etc.

Aún se encuentra en fase de desarrollo (actualmente está en versión alpha) y presenta algunos problemas.



**+NEW** : Construir un nuevo contenedor a partir de una imagen

**Stop/Start** : Para detener o iniciar un contenedor

**Restart** : Resetea un contenedor

**Exec** : Abre una terminal al interior del contenedor

Así se ve docker-hub en Kitematic

The screenshot shows the Kitematic Docker Hub interface. On the left, a sidebar lists containers: 'examples\_db\_1' (mysql:5.7) and 'examples\_wordpress\_1' (wordpress:latest). The 'examples\_wordpress\_1' container is highlighted with a purple box, and a red arrow points from it to the text 'Contenedor' and a blue arrow points to 'Nombre de la imagen'. The main area displays a grid of recommended Docker images. A search bar at the top is labeled 'Buscador de imagenes' with a red arrow. The 'official rethinkdb' image is highlighted with a purple box, and a red arrow points from its 'CREATE' button to the text 'Bajar imagen'. A green arrow points from the 'official rethinkdb' image to the text 'Repositorio', and a red arrow points from the 'official rethinkdb' image to the text 'Detalles del repo'.

Containers

examples\_db\_1  
mysql:5.7

examples\_wordpress\_1  
wordpress:latest

Contenedor

Nombre de la imagen

Search for Docker Images from Docker Hub

FILTER BY All Recommended My Repos My Images

Recommended

kitematic  
hello-world-nginx  
A light-weight nginx container that demonstrates the features of Kitematic  
81 3M CREATE

ghost  
official  
ghost  
Ghost is a free and open source blogging platform written in JavaScript  
591 6M CREATE

official  
jenkins  
Official Jenkins Docker image  
2.9K 27M CREATE

official  
redis  
Redis is an open source key-value store that functions as a data structure server.  
3.9K 288M CREATE

official  
rethinkdb  
RethinkDB is an open-source, document database that makes it easy to build and scale realtime...  
411 7M CREATE

official  
solr  
Solr is the popular, blazing-fast, open source enterprise search platform built on Apache...  
400 3M CREATE

official  
elasticsearch  
Elasticsearch is a powerful open source search and analytics engine that makes data easy to...  
2.3K 77M CREATE

official  
postgres  
The PostgreSQL object-relational database system provides reliability and data integrity.  
3.7K 65M CREATE

Repositorio

Detalles del repo

Bajar imagen

<https://docs.docker.com/install/>

<https://docs.docker.com/engine/docker-overview/>

<https://opensource.com/business/14/7/guide-docker>

[https://www.youtube.com/watch?time\\_continue=35&v=Q5POuMHxW-o](https://www.youtube.com/watch?time_continue=35&v=Q5POuMHxW-o)



# ANACONDA



- **Anaconda** es una plataforma Open Data Science motorizada por Python
- Dispone de un administrador de entorno y de paquetes (librerías)
- Conda**
- Cuenta con un entorno de desarrollo para Python (IDE): **Spyder**
- Es posible ejecutar notebooks interactivas con:
  - o **Jupyter Notebooks**
  - o **Jupyter Lab**
- Una interfaz interactiva para graficar el flujo de trabajo: **Orange**



Leading Open Data Science Platform Powered by Python



Leading Package and Environment Manager

### OPEN DATA SCIENCE



### DATA



### COMPUTATION



## RESULTS

- ✓ Reproducibility for packages and environments
- ✓ Building interactive visualizations

Editor - C:\Documents and Settings\carlos\Mis documentos\Python\Interpolation.py

Interpolation.py x montecarlo\_pi.py x

```
1 """
2 Interpolation of an N-D curve
3 From the SciPy Cookbook
4 """
5
6 from numpy import arange, cos, linspace, pi, sin, random
7 from scipy.interpolate import splprep, splev
8
9 # make ascending spiral in 3-space
10 t=linspace(0,1.75*2*pi,100)
11
12 x = sin(t)
13 y = cos(t)
14 z = t
15
```

Variable explorer

Name	Type	Size	Value
e	float	1	2.7182818284590451
pi	float	1	3.1415926535897931

Object inspector

Source: Console Object: array Options

**array(...)**  
Function of numpy.core.multiarray module

array(object, dtype=None, copy=True, order=None, subok=False, ndmin=0)

Create an array.

**Parameters**

**object**: array\_like  
An array, any object exposing the array interface, an object whose `__array__` method returns an array, or any (nested) sequence.

**dtype**: data-type, optional  
The desired data-type for the array. If not given, then the type will be determined as the minimum type required to hold

Console

IPython 1 x 00:02:25

IPython 0.10.1 -- An enhanced Interactive Python.

? -> Introduction and overview of IPython's features.

%quickref -> Quick reference.

help -> Python's own help system.

object? -> Details about 'object'. ?object also works, ?? prints more.

Welcome to pylab, a matplotlib-based Python environment.  
For more information, type 'help(pylab)'.

In [1]:

Permissions: RW End-of-lines: LF Encoding: UTF-8-GUESSED Line: 7 Column: 1

jupyter convnets Last Checkpoint: 08/29/2017 (unsaved changes) Logout

File Edit View Insert Cell Kernel Help Python 3

Code CellToolbar

## Looking for Collisions in Training, Test and Validation Sets

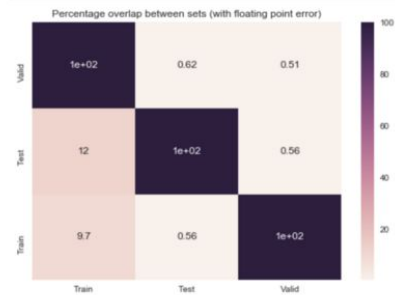
```
In [47]: train_hashes = set(hash(i.tostring()) for i in train_dataset)
test_hashes = set(hash(i.tostring()) for i in test_dataset)
valid_hashes = set(hash(i.tostring()) for i in valid_dataset)
```

```
In [66]: with Timer('28x 28 Pixel Photo Set Sizes {} {}'.format(
len(train_dataset), len(test_dataset), len(valid_dataset))):
train_hashes = set(hash(i.tostring()) for i in train_dataset)
test_hashes = set(hash(i.tostring()) for i in test_dataset)
valid_hashes = set(hash(i.tostring()) for i in valid_dataset)

result = np.ndarray(shape=(3, 3))
hashes = [train_hashes, test_hashes, valid_hashes]

for i, first_hashes in enumerate(hashes):
    for j, next_hashes in enumerate(hashes):
        result[i, j] = 100.0 * len(first_hashes.intersection(next_hashes)) / len(first_hashes)

ax = sns.heatmap(result, annot=True)
plt.title('Percentage overlap between sets (with floating point error)')
ax.set_yticklabels(['Train', 'Test', 'Valid'])
ax.set_xticklabels(['Train', 'Test', 'Valid'])
plt.show();
```



28x 28 Pixel Photo Set Sizes 200000 10000 10000 taken 0.89 secs

```
In [25]: import time

def softmax(x):
    return np.exp(x) / np.sum(np.exp(x), axis=0)
```

**DigitalHouse** >  
Coding School

File Notebook Editor Terminal Console Help

## Commands

[🏠](#) > [drivendata](#) > [deep-water](#)

Name	Last Modified
data_clean	5 months ago
data_raw	5 months ago
img	2 months ago
plots	5 months ago
rconnect	2 months ago
report.ipynb	2 months ago
cleaning.R	5 months ago
data_load.R	5 months ago
data_water.Rproj	5 months ago
eda.R	5 months ago
helpers.R	5 months ago
machine_learning.R	5 months ago
maps.R	5 months ago
notes.Rmd	5 months ago
report.html	2 months ago
report.Rmd	2 months ago

Launcher

report.ipynb

Python 2 ☒

```
In [2]: import pandas as pd
import missingno

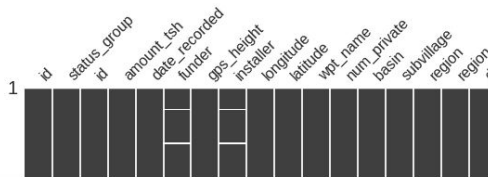
%matplotlib inline
```

```
In [4]: # collect data urls
train_features_url = "http://s3.amazonaws.com/drivendata/data/7/pub
train_labels_url = "http://s3.amazonaws.com/drivendata/data/7/publi
test_features_url = "http://s3.amazonaws.com/drivendata/data/7/publ

# read in data
train_features = pd.read_csv(train_features_url)
train_labels = pd.read_csv(train_labels_url)
test_features = pd.read_csv(test_features_url)
```

```
In [5]: # merge dataframes
train = pd.concat([train_labels, train_features], axis = 1)
```

```
In [6]: # missing data visualise
missingno.matrix(train)
```




..ta/deep-water ✕

```

-rwxr-xr-x 1 boyanangelov staff      256 Feb 16 20:16 maps.R
-rwxr-xr-x 1 boyanangelov staff      706 Feb 16 20:16 notes.Rmd
drwxr-xr-x 3 boyanangelov staff      102 Feb 16 20:16 plots
-rwxr-xr-x 1 boyanangelov staff      7316 Jun 5 21:07 report.Rmd
-rwxr-xr-x 1 boyanangelov staff    3136447 Jun 5 21:09 report.html
-rwxr-xr-x 1 boyanangelov staff    3951919 Jun 5 21:09 report_ipynb
drwxr-xr-x 3 boyanangelov staff      102 Jun 5 21:00 rconnect

# boyanangelov @ mac-home in ~/ds/drivendata/deep-water on git:master * [11:54:12]

```

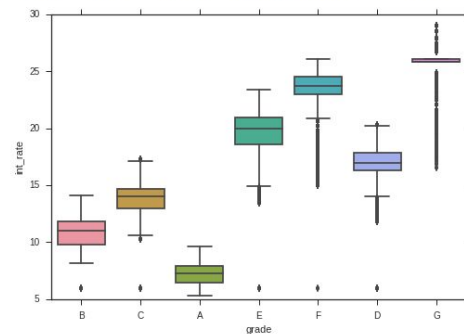


Markdown

Python 2 ○

```
In [6]: sns.boxplot(x = data.raw.grade, y = data.raw.int_rate)
```

```
Out[6]: <matplotlib.axes. subplots.AxesSubplot at 0x7fd42ebc6290>
```



```
In [7]: data_raw.shape
```

```
Out[7]: (887379, 74)
```

```
In [15]: sns.distplot(data raw['loan amnt']);
```

