









# Sagemaker Pipelines con R.

# Algún warning.

-  Sagemaker 
-  Python 
-  Arquitecturas 
-  Ingeniería de Machine Learning 

**Qué  
vamos  
a ver.**

**Cómo gestionar tu modelo de R  
en Sagemaker**

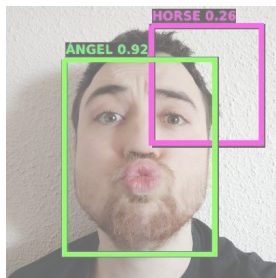
# El equipo.



**Bea Hernández**

**ML Engineer**

[twitter.com/chucheria](https://twitter.com/chucheria)  
[github.com/chucheria](https://github.com/chucheria)



**Ángel Delgado**

**ML Engineer**

[twitter.com/thinbaker](https://twitter.com/thinbaker)



**María López**

**Data Scientist**

[linkedin.com/in/maria-lopez-a67796199/](https://linkedin.com/in/maria-lopez-a67796199/)



**Alejandro Hernández**

**Data Scientist**

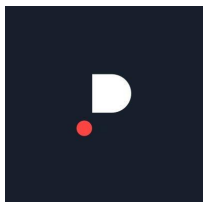
[linkedin.com/in/alejandrohdez/](https://linkedin.com/in/alejandrohdez/)

# Gracias.

Antes de nada...



A R-Ladies Madrid y Kairos por invitarme



A Paradigma Digital por darme el tiempo

# 01.



R en Sagemaker

# Las bases

# 01.01



R en Sagemaker

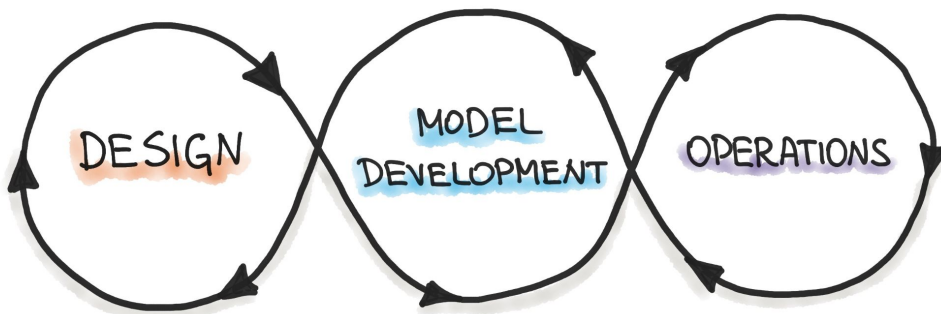


Las bases

# ML Engineering.

# ML Engineering.

Tres fases principales del proceso



Comprensión del  
problema de negocio y su  
solución

Desarrollo de una  
solución con un modelo  
de ML

Puesta en producción y  
mantenimiento  
automático

[Referencia](#)



# 01.02



R en Sagemaker



Las bases

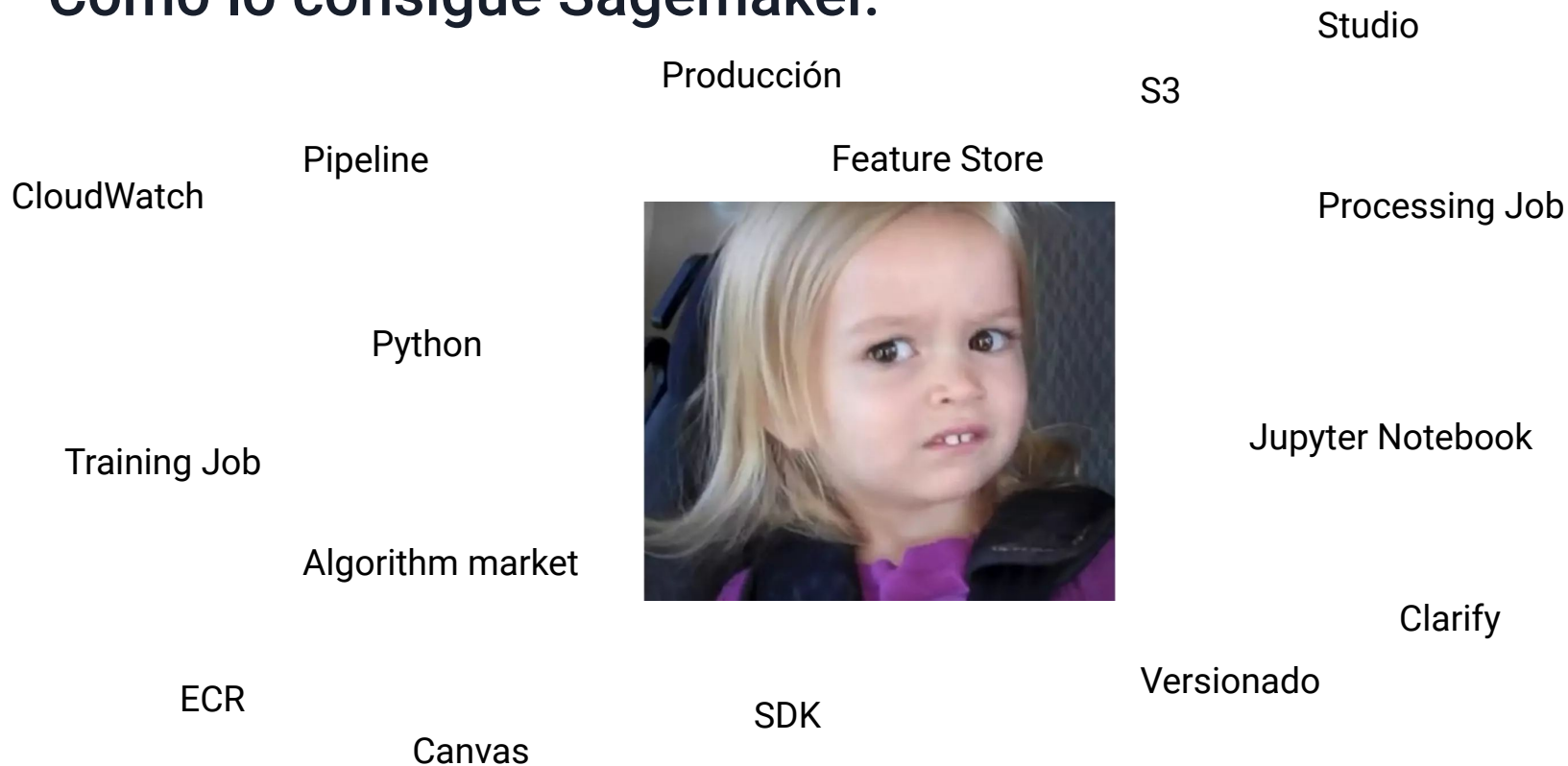
# Sagemaker.

# De la idea al producto.

- Exploración de datos
- Desarrollo del modelo
- Entrenamiento
- Producción
- Monitorización



# Cómo lo consigue Sagemaker.



# 02.



R en Sagemaker

## El planteamiento.

# 02.01

...

R en Sagemaker

—

El planteamiento

## El problema.

# Premisas.



## Negocio.

Tenemos un problema a resolver con un modelo.

El problema puede venir de una necesidad de negocio, de nuestro proyecto, para resolver un concurso, ...



## Entorno.

El entorno de trabajo es local, está fuera de Sagemaker. Es el entorno que tenemos predefinido y no lo vamos a cambiar.



## Modelo.

El problema resuelto es el famoso Abalone. Lo que se busca es la predicción del número de anillos del molusco.

# 02.02

• • •

R en Sagemaker

— El planteamiento

## La solución.

# El truqui.



## Sagemaker

Sagemaker me ofrece unos servicios y una manera de usarlos.

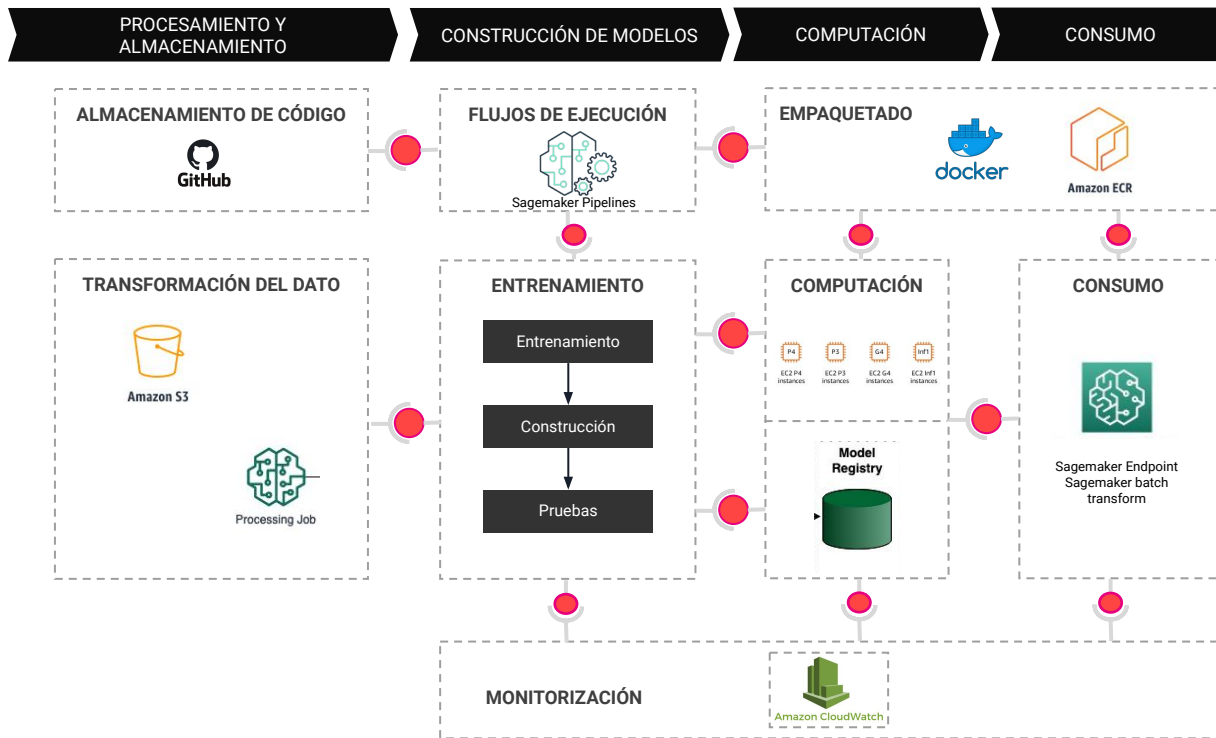


## Mi Sagemaker

El servicio es lo suficientemente flexible para que cubra mi solución.



# Arquitectura de la solución.



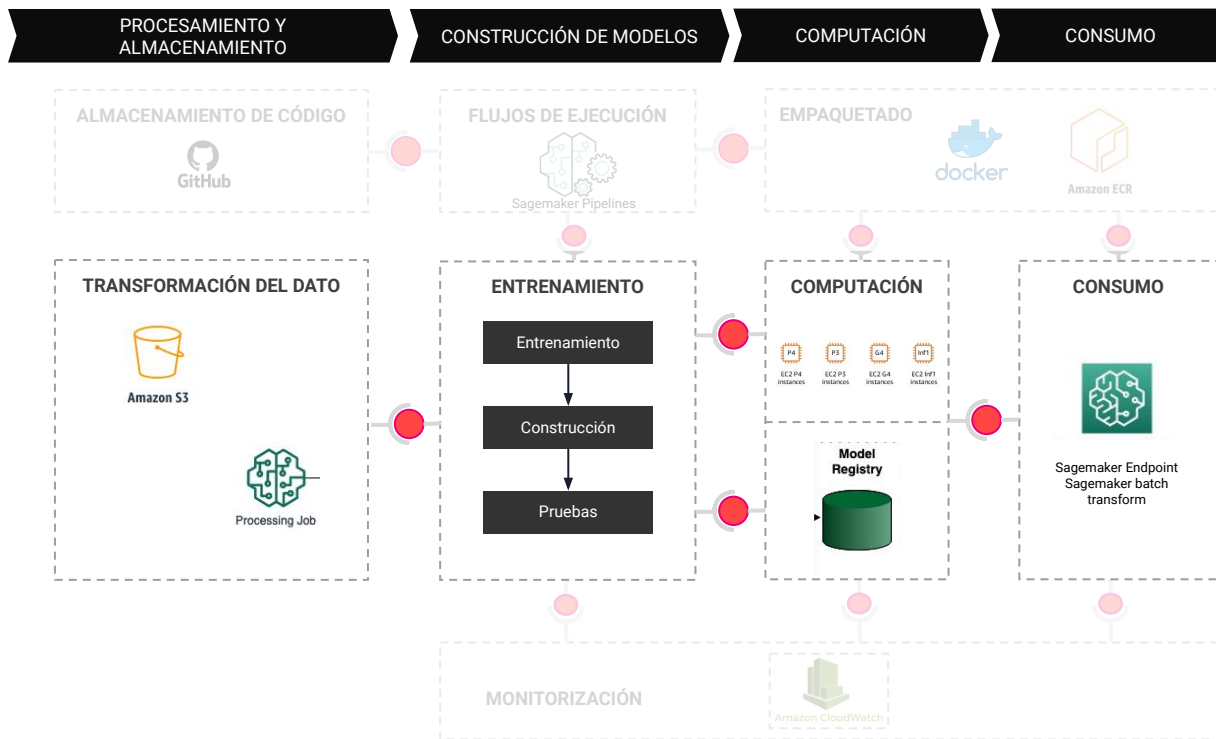
# 03.



R en Sagemaker

## El modelo.

# Arquitectura de la solución.



# Procesamiento de datos.

```
# Read data
data <- read_csv(FILE_DATA, col_names = FALSE)
names(data) <- c("sex", "length", "diameter", "height", "whole_weight",
                "shucked_weight", "viscera_weight", "shell_weight", "rings")

# Wrangle
data$sex <- as.factor(data$sex)
abalone <- data %>%
  mutate(female = as.integer(iffelse(sex == "F", 1, 0)),
         male = as.integer(iffelse(sex == "M", 1, 0)),
         infant = as.integer(iffelse(sex == "I", 1, 0))) %>%
  select(-sex)

abalone <- abalone %>%
  filter(height != 0)

train <- abalone %>%
  sample_frac(size = 0.7)

abalone <- anti_join(abalone, train)
test <- abalone %>%
  sample_frac(size = 0.5)
validation <- anti_join(abalone, test) %>%
  select(-rings)

# Write data
write_csv(train, FILE_TRAIN)
write_csv(test, FILE_TEST)
write_csv(validation, FILE_VALIDATION)
```

# Entrenamiento del modelo.

```
# Train model
train_data <- read_csv(FILE_TRAIN)
xg <- boost_tree(trees = 200) %>%
  set_engine("xgboost") %>%
  set_mode("regression") %>%
  fit(rings ~ ., data = train_data)
```

```
saveRDS(xg, MODEL_PATH)
```

```
# Test model
test_data <- read_csv(FILE_TEST)
x <- test_data %>%
  select(-rings)
y <- test_data %>%
  select(rings)
```

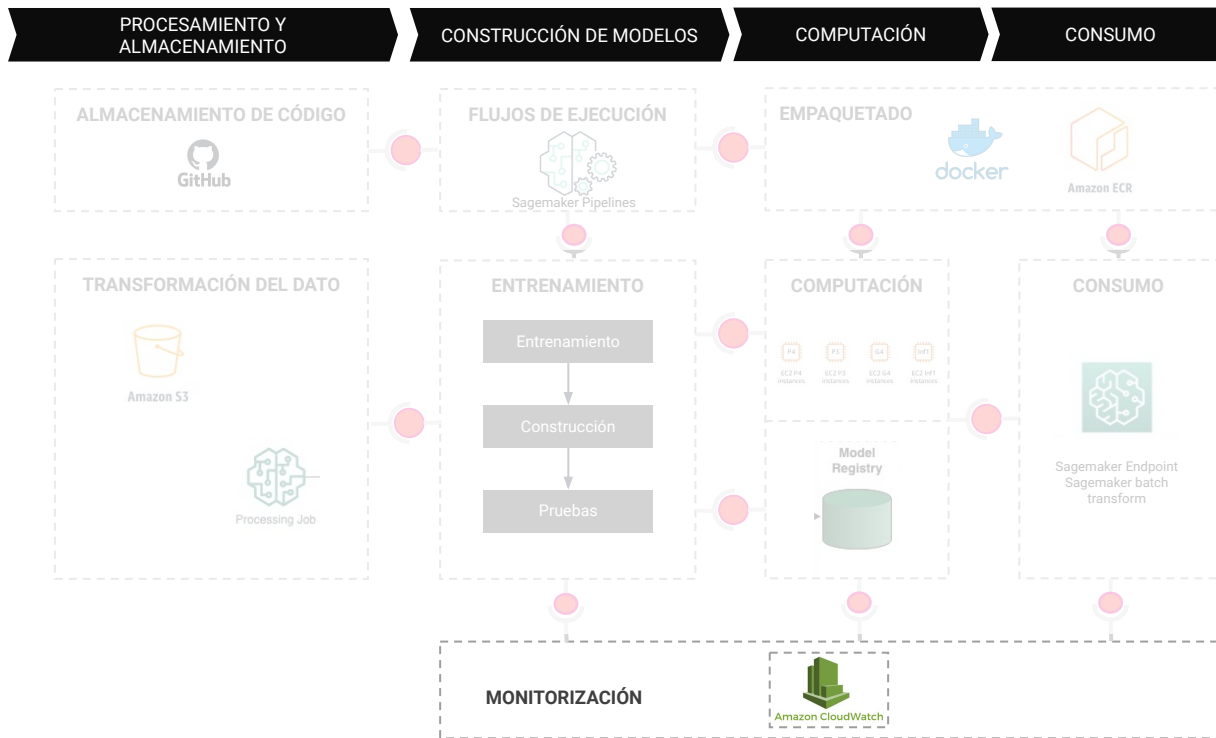
# Inferencia.

```
# Load model
model <- readRDS(INFERENCE_MODEL_PATH)
features <- c("sex", "length", "diameter", "height", "whole_weight",
              "shucked_weight", "viscera_weight", "shell_weight")

# Run service
serve <- function() {
  app <- plumb('plumber.R')
  app$run(host = INFERENCE_SERVER_IP, port = INFERENCE_SERVER_PORT)
}

serve()
```

# Arquitectura de la solución.



# Métricas de entrenamiento.

```
# Test model
test_data <- read_csv(FILE_TEST)
x <- test_data %>%
  select(-rings)
y <- test_data %>%
  select(rings)

pred <- predict(xg, x, type = "raw") %>%
  bind_cols(test_data)

mae <- pred %>%
  mae(truth = rings, estimate = ...1) %>%
  select(estimate)
cat(paste0('MAE=', mae, '\n'))

mape <- pred %>%
  mape(truth = rings, estimate = ...1) %>%
  select(estimate)
cat(paste0('MAPE=', mape, '\n'))

rmse <- pred %>%
  rmse(truth = rings, estimate = ...1) %>%
  select(estimate)
cat(paste0('RMSE=', rmse, '\n'))
```



# 04.

• • •

R en Sagemaker

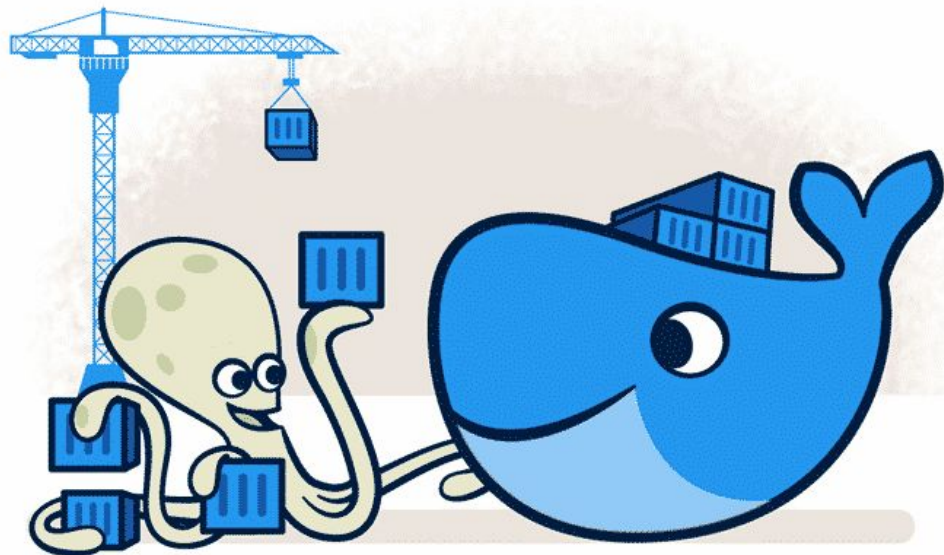
## El contenedor.

# Docker.

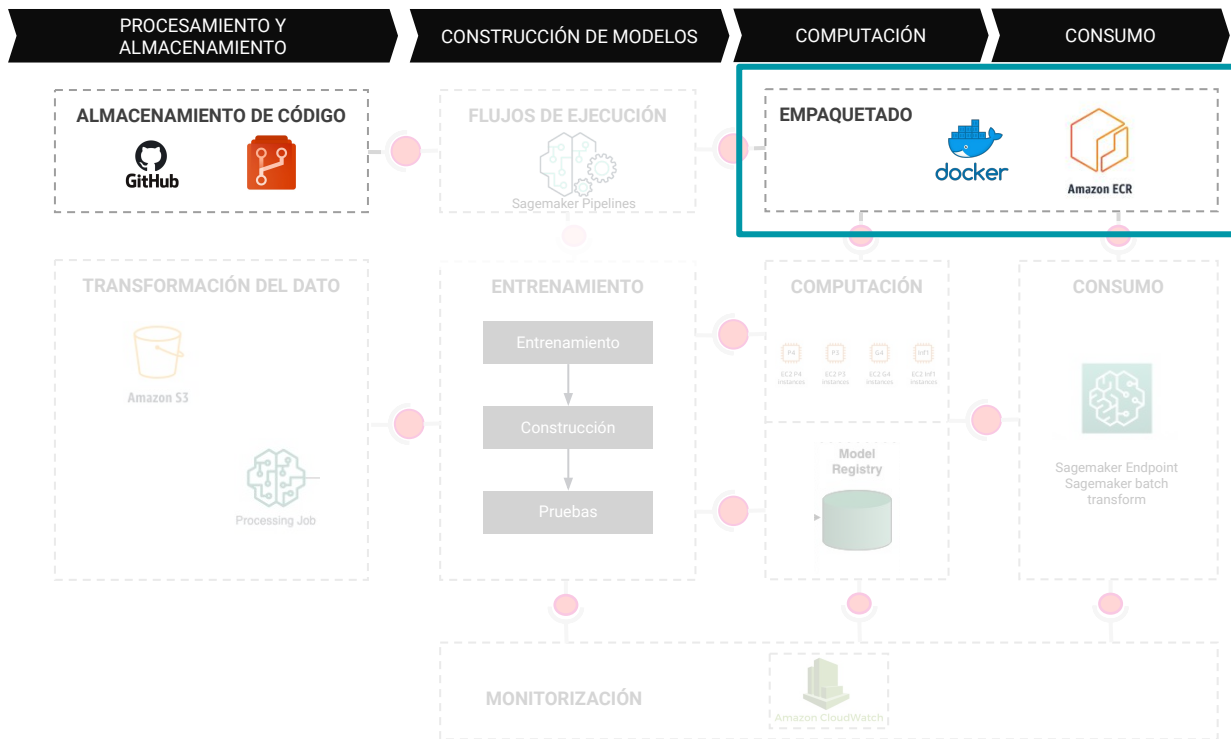
Lo usamos para **contenerizar software**.

¿Qué significa?

Creamos un paquete estándar que incluye los requisitos mínimos para ejecutar el software.



# Arquitectura de la solución.



# Dockerfile.

```
FROM r-base:latest

# Install dependencies
RUN apt-get -y update && \
    apt-get install -y --no-install-recommends \
    wget \
    apt-transport-https \
    ca-certificates \
    libcurl4-openssl-dev \
    libsodium-dev

# Copy project and install packages
COPY ./src /home
WORKDIR /home
RUN Rscript install.R

# Create alias
RUN echo "#!/bin/bash\n/usr/bin/Rscript /home/preprocess.R" > /usr/bin/processing
RUN echo "#!/bin/bash\n/usr/bin/Rscript /home/train.R" > /usr/bin/train
RUN echo "#!/bin/bash\n/usr/bin/Rscript /home/serve.R" > /usr/bin/serve
# RUN echo "#!/bin/bash\n/usr/bin/Rscript /home/explain/main.R" > /usr/bin/explain

RUN chmod +x /usr/bin/processing
RUN chmod +x /usr/bin/train
RUN chmod +x /usr/bin/serve
# RUN chmod +x /usr/bin/explain
```

# Instalar paquetes.

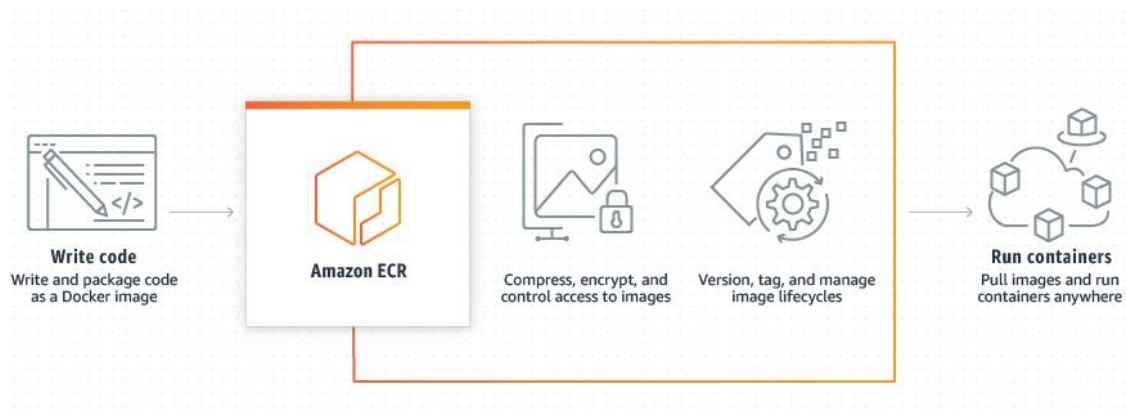
```
install.packages("remotes")  
library("remotes")
```

```
install_version("xgboost", "1.5.2.1")  
install_version("tidymodels", "0.1.4")  
install_version("readr", "2.1.2")  
install_version("dplyr", "1.0.8")  
install_version("plumber", "1.1.0")
```

# Amazon ECR.

Amazon Elastic Container Registry.

- Otro servicio
- Todas las imágenes que empiecen por *sagemaker-* se pueden gestionar desde ahí



# Amazon ECR.

```
image="sagemaker-rladies"

fullname="${account}.dkr.ecr.${region}.amazonaws.com/${image}:latest"

#aws ecr create-repository --repository-name "${image}" > /dev/null

aws ecr get-login-password --region "${region}" | docker login --username AWS --password-stdin "${account}.dkr.ecr.${region}.amazonaws.com

docker tag ${image} ${fullname}

docker push ${fullname}
```

# 05.

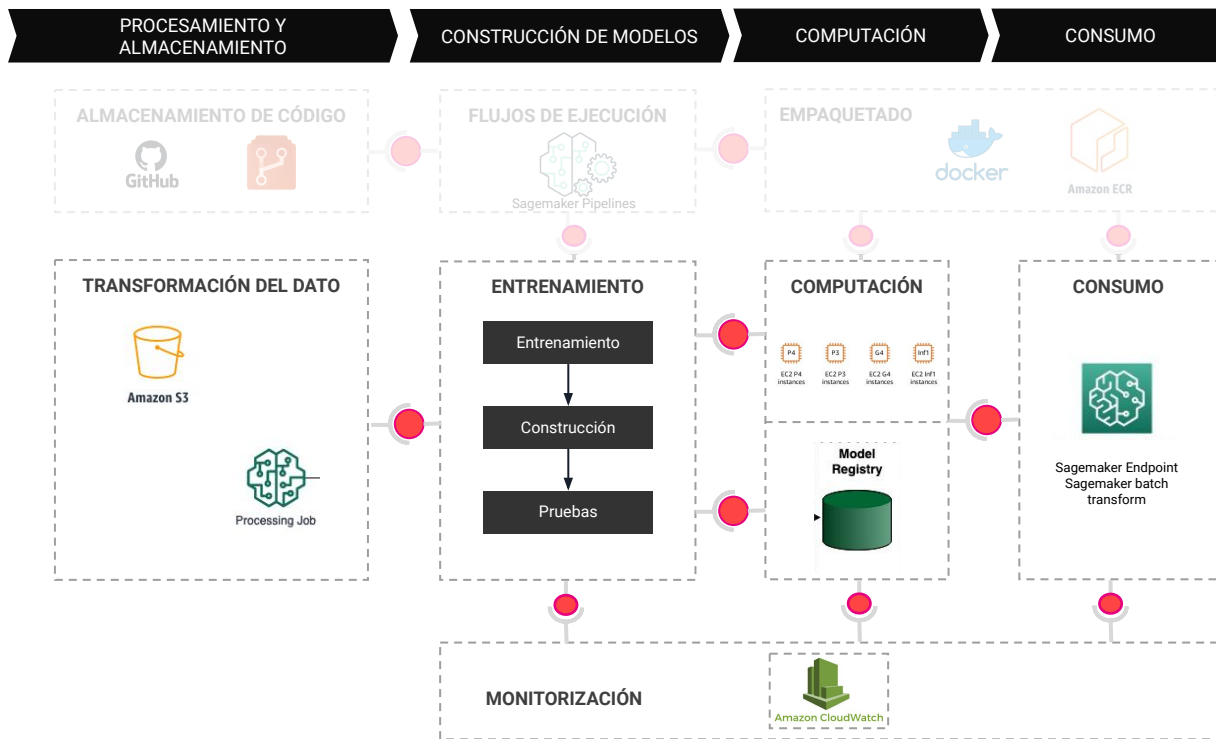


R en Sagemaker

## El pipeline.



# Arquitectura de la solución.



# 05.01



R en Sagemaker



El pipeline

# Sagemaker SDK.

# Sagemaker SDK.



## Software development kit.

Librería o recursos que te permiten usar una herramienta dentro de un desarrollo software.



## Lenguaje.

El SDK de Sagemaker está escrito en Python. Se puede acceder a él mediante reticulate.



## Ventajas.

Nos permite acceder a los recursos de Sagemaker fuera de la herramienta e incluirla a nuestro desarrollo.



## Saber más.

[Documentación](#)

# Procesamiento.

```

s_processing = ProcessingStep(
    name='processing-{}'.format(name),
    processor=Processor(
        image_uri=image,
        role=role,
        entrypoint=['/usr/bin/bash'],
        instance_count=1,
        instance_type=instance,
        env=env
    ),
    inputs=[
        ProcessingInput(
            source='s3://sagemaker-pipelines-r/data/abalone.csv',
            destination='/opt/ml/processing/input/',
            input_name='input_data',
        )
    ],
    outputs=[
        ProcessingOutput(
            source='/opt/ml/processing/output/',
            destination='s3://sagemaker-pipelines-r/data/processing/output',
            output_name='output_data',
        )
    ],
    code=None #This is treaky
)

```

```

RUN echo "#!/bin/bash\n/usr/bin/Rscript /home/preprocess.R" > /usr/bin/processing
...

# Read data
data <- read_csv(FILE_DATA, col_names = FALSE)

# Write data
write_csv(train, FILE_TRAIN)
write_csv(test, FILE_TEST)
write_csv(validation, FILE_VALIDATION)

```

# Entrenamiento.

```
s_training = TrainingStep(
    name='train-{}'.format(name),
    estimator=Estimator(
        image_uri=image,
        instance_type=instance,
        instance_count=1,
        output_path='s3://sagemaker-pipelines-r/train/',
        role=role,
        environment=env,
        disable_profiler=True,
        metric_definitions=[
            {
                "Name": "train:mae",
                "Regex": "MAE=(.*)"
            },
            {
                "Name": "train:mape",
                "Regex": "MAPE=(.*)"
            },
            {
                "Name": "train:rmse",
                "Regex": "RMSE=(.*)"
            },
        ],
        enable_sagemaker_metrics=True,
    ),
    inputs=TrainingInput(
        s3_data=s_processing.properties.ProcessingOutputConfig.Outputs[
            "output_data"].S3Output.S3Uri
    )
)
```

```
mae <- pred %>%
  mae(truth = rings, estimate = ...1) %>%
  select(.estimate)
cat(paste0('MAE=', mae, '\n'))

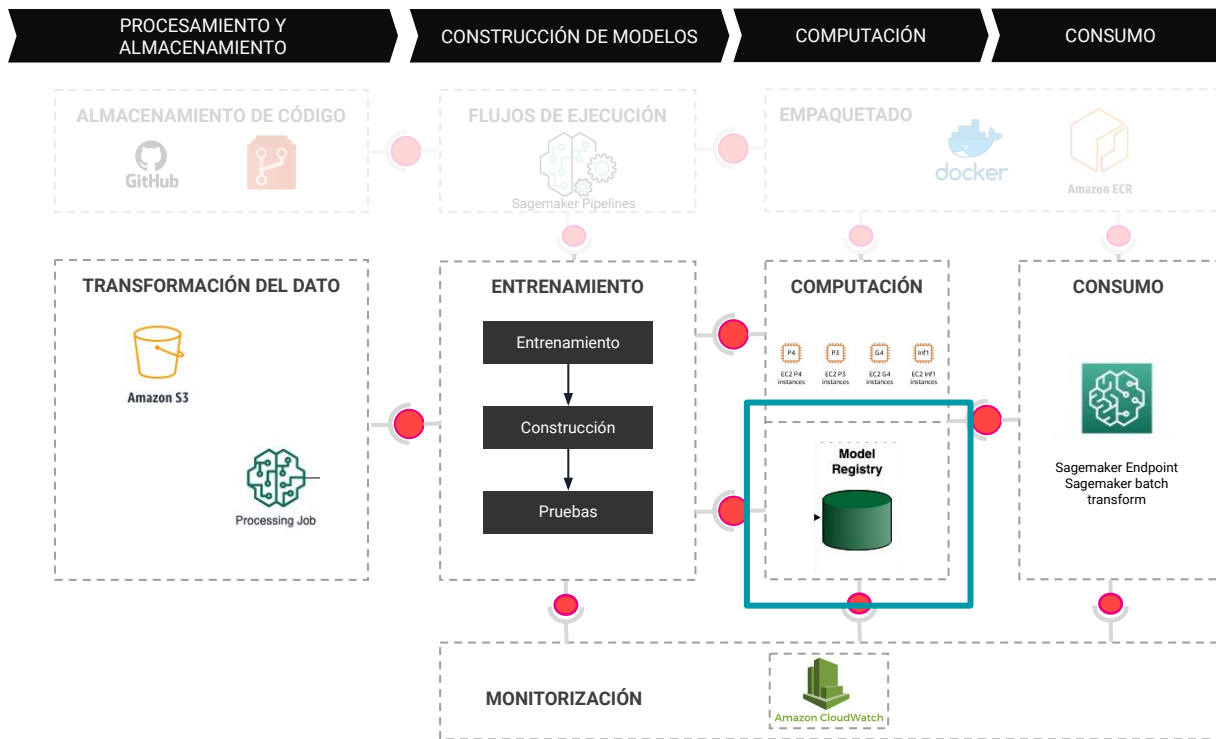
mape <- pred %>%
  mape(truth = rings, estimate = ...1) %>%
  select(.estimate)
cat(paste0('MAPE=', mape, '\n'))

rmse <- pred %>%
  rmse(truth = rings, estimate = ...1) %>%
  select(.estimate)
cat(paste0('RMSE=', rmse, '\n'))
```

# Inferencia.

```
s_batch_predict = TransformStep(
    name='predict-{}'.format(name),
    transformer=Transformer(
        model_name=s_create_model.properties.ModelName,
        instance_type=instance,
        instance_count=1,
        output_path="s3://sagemaker-pipelines-r/data/inference",
        env=env
    ),
    inputs=TransformInput(
        data="s3://sagemaker-pipelines-r/data/processing/output/validation.csv",
        content_type='text/csv'
    )
)
```

# Arquitectura de la solución.



# Registrar modelo.

```
s_create_model = CreateModelStep(
    name='create-{}'.format(name),
    model=Model(
        image_uri=image,
        model_data=s_training.properties.ModelArtifacts.S3ModelArtifacts,
        role=role,
        sagemaker_session=session
    ),
    inputs= CreateModelInput(
        instance_type=instance
    ),
)
steps.append(s_create_model)

s_register = RegisterModel(
    name='register-{}'.format(name),
    estimator=s_training.estimator,
    model_data=s_training.properties.ModelArtifacts.S3ModelArtifacts,
    content_types=["text/csv"],
    response_types=["text/csv"],
    inference_instances=[instance],
    transform_instances=[instance],
    approval_status="Approved",
    model_package_group_name= "test-rladies",
    description="Model group for rladies Puebla - XGB for abalon dataset",
    depends_on=[s_training],
)
```

Versions

Settings

Search column name to start

Version	Stage	Status	Short description	Modified by
11	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	
10	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	
9	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	
8	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	
7	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	
6	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	
5	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	
4	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	
3	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	
2	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	
1	None	Approved	Model group for rladies Puebla - XGB for abalon dataset	



# 06.

• • •

R en Sagemaker

# Conclusiones.

# 08.01



R en Sagemaker

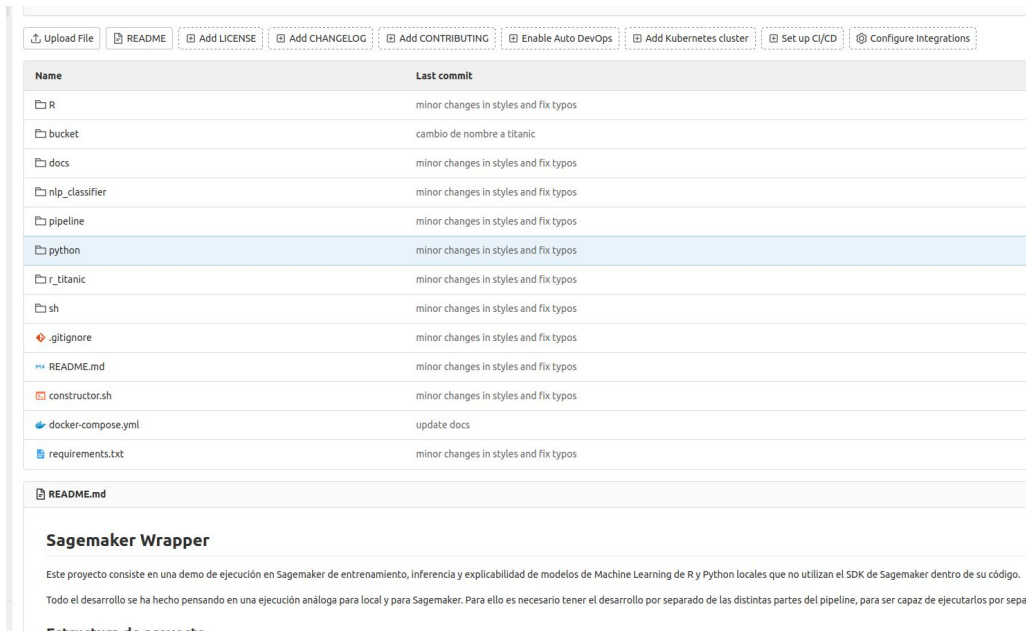


Conclusiones

## Nuestra solución.

# ML Engineering con Sagemaker.

- Sagemaker pipelines es lo suficientemente flexible como para admitir casi cualquier solución
- Tiene un escalón de aprendizaje en la parte de MLOps
- Solución completa para gestión de modelos
- Nuestra solución permite incorporar un modelo ya desarrollado haciendo pocos cambios



The screenshot shows a GitHub repository interface. At the top, there are buttons for 'Upload File', 'README', 'Add LICENSE', 'Add CHANGELOG', 'Add CONTRIBUTING', 'Enable Auto DevOps', 'Add Kubernetes cluster', 'Set up CI/CD', and 'Configure Integrations'. Below this is a table listing files and their last commit messages.

Name	Last commit
R	minor changes in styles and fix typos
bucket	cambio de nombre a titanic
docs	minor changes in styles and fix typos
nlp_classifier	minor changes in styles and fix typos
pipeline	minor changes in styles and fix typos
python	minor changes in styles and fix typos
r_titanic	minor changes in styles and fix typos
sh	minor changes in styles and fix typos
.gitignore	minor changes in styles and fix typos
README.md	minor changes in styles and fix typos
constructor.sh	minor changes in styles and fix typos
docker-compose.yml	update docs
requirements.txt	minor changes in styles and fix typos

Below the table, the 'README.md' file is selected, showing the following content:

### Sagemaker Wrapper

Este proyecto consiste en una demo de ejecución en Sagemaker de entrenamiento, inferencia y explicabilidad de modelos de Machine Learning de R y Python locales que no utilizan el SDK de Sagemaker dentro de su código.

Todo el desarrollo se ha hecho pensando en una ejecución análoga para local y para Sagemaker. Para ello es necesario tener el desarrollo por separado de las distintas partes del pipeline, para ser capaz de ejecutarlos por separado.

### Estructura de archivos

# 08.02



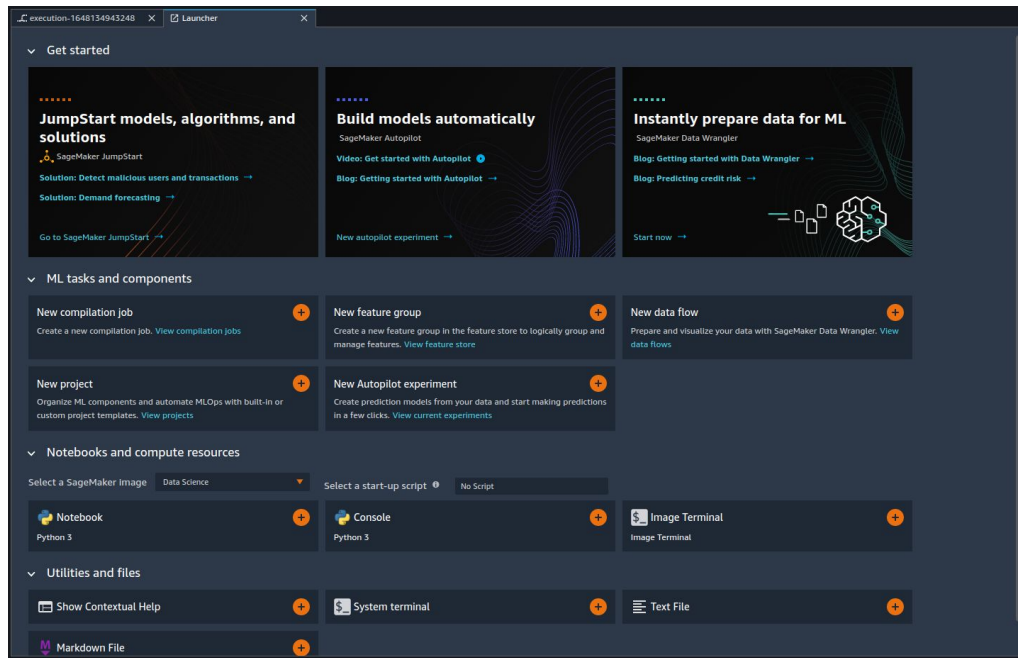
R en Sagemaker

Conclusiones

## La solución de Sagemaker.

# Sagemaker Studio.

- Sagemaker Studio te facilita la integración
- Es una solución menos flexible
- Solución completa para gestión de modelos
- Sigues necesitando conocimientos para gestionar todo el flujo y conocer sus productos



# Gracias.

Estamos contratando remoto o híbrido

<https://www.paradigmadigital.com/empleo>

