

# DESPLIEGUE DE UN MODELO DE IA DEL DATASET MNIST FASHION USANDO UN SERVICIO PaaS PARA SER EXPUESTO COMO API USANDO DOCKER + PYTHON + FLASK

Felipe García (COL)

M.Sc Ingeniería de Sistemas y Computación - UNAL

Ingeniero Electrónico - UTP

# Introducción

- Ingeniero Electrónico
- MSc Ingeniería de Sistemas y Computación – UNAL Medellín, COL
- Sr Software Engineer at Devbase Inc
- Peoples Company – Real State
- Cash Rent – Real State
- Impossible Foods – e-commerce





```
mirror_mod = modifier_ob.  
# Add mirror object to mirror  
mirror_mod.mirror_object  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
  
#selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.  
mirror_ob.select = 0  
= bpy.context.selected_object  
data.objects[one.name].select  
  
print("please select exactly  
  
-- OPERATOR CLASSES ----  
  
types.Operator):  
on X mirror to the selected  
object.mirror_mirror_x"  
mirror X"  
  
context):  
context.active_object is not
```

# Orden del día

- Motivación
- Conceptos Clave
- Modelo de IA
- API con Python + Flask
- Empaquetamiento con Docker
- Despliegue de la API con fl0.io
- Uso de la API

# Motivación o Necesidad

- Puesta en producción
- Escalabilidad
- Disponibilidad



# Conceptos Clave

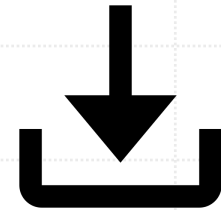
- Python
- Tensor Flow + Keras
- Flask
- Docker
- REST API
- Platform as a Service
- MLOps

## Recursos del Taller

- [Repositorio API](#)
- [Repositorio Web App](#)
- [fl0.io](#)

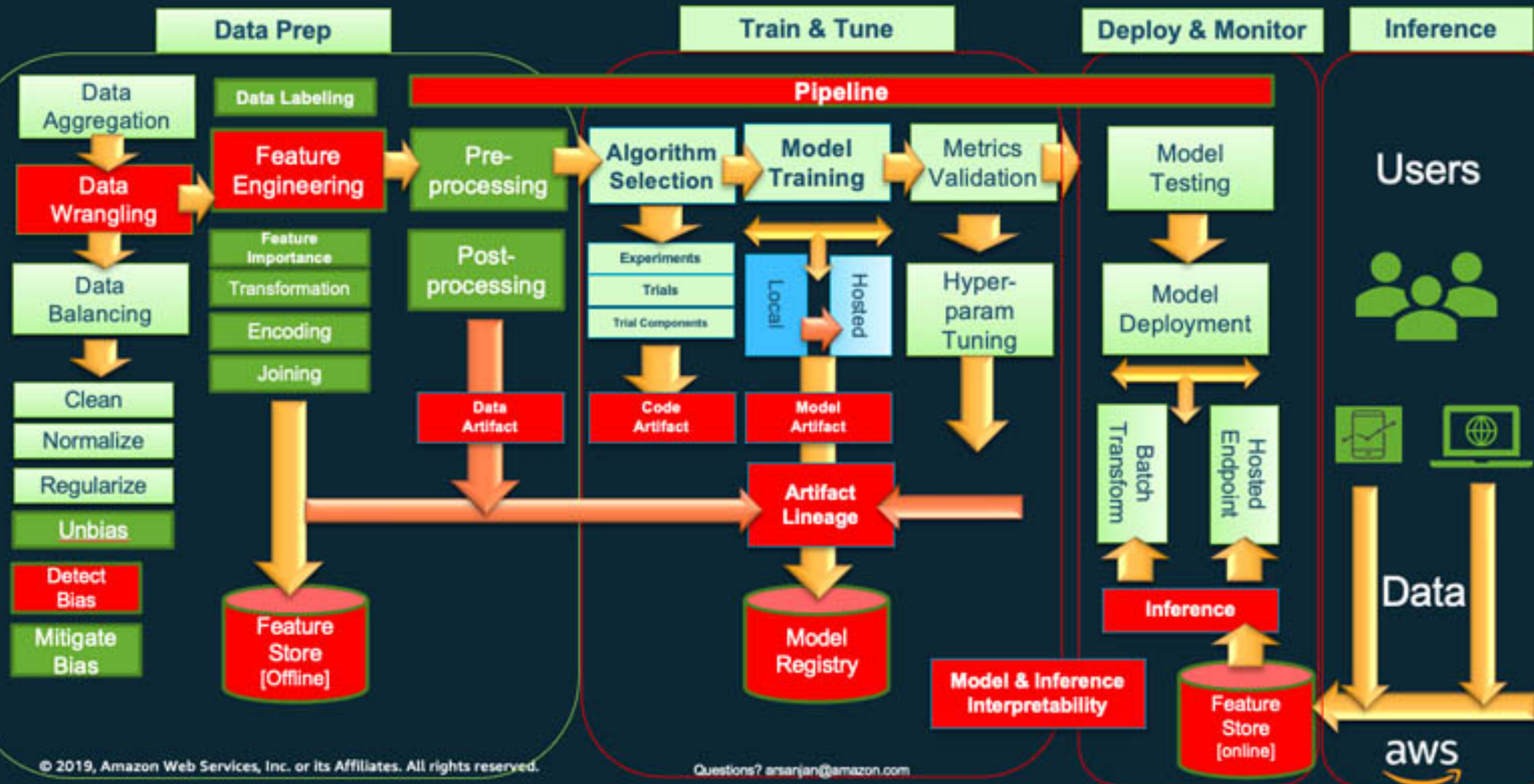
## App Web

- <https://ml-model-web.onrender.com>



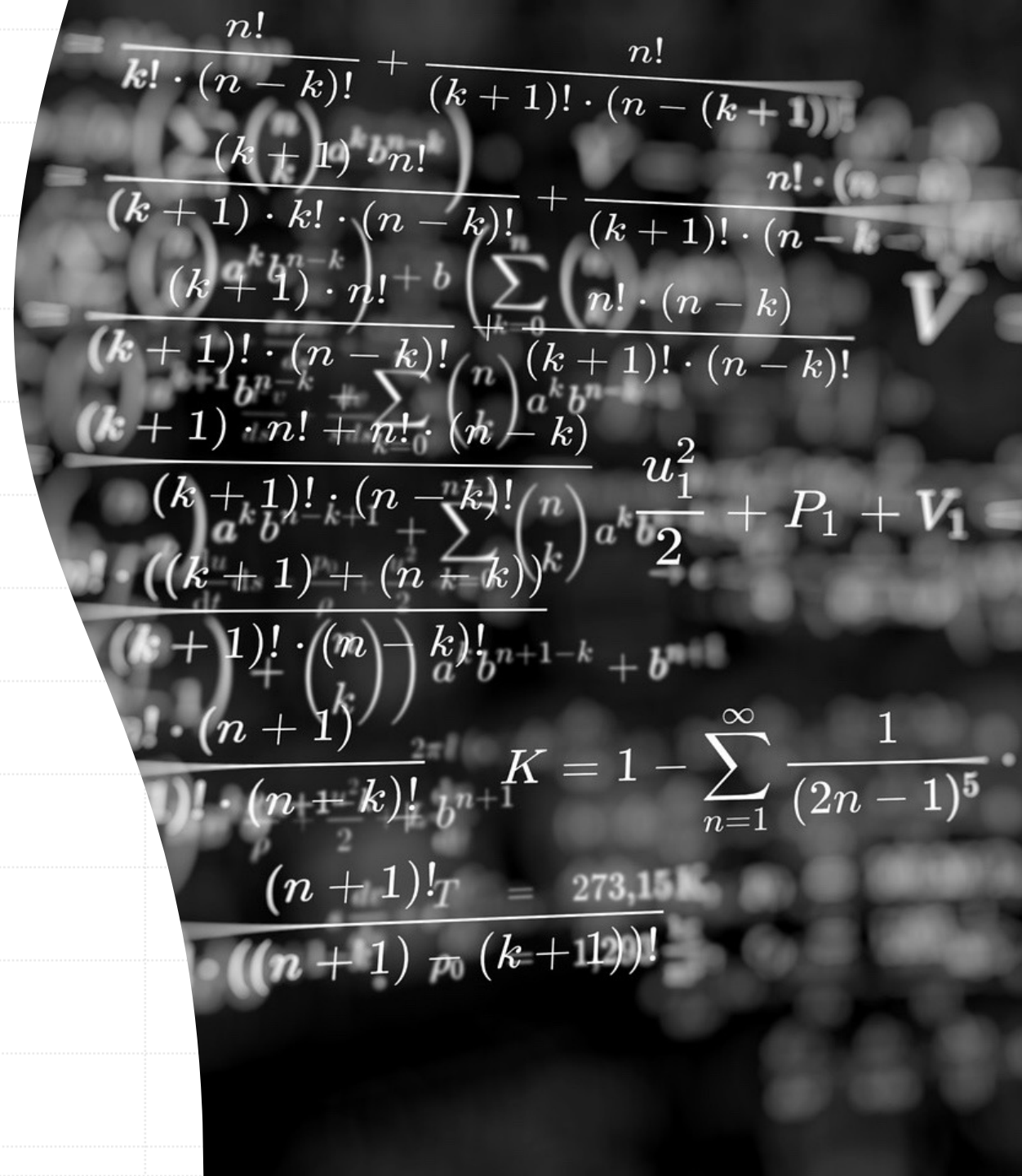


# The ML-Lifecycle: Detailed View



# Modelo de IA: MNIST Fashion + TF

- Descripción general
- Almacenamiento del modelo entrenado.



# API con Python + Flask

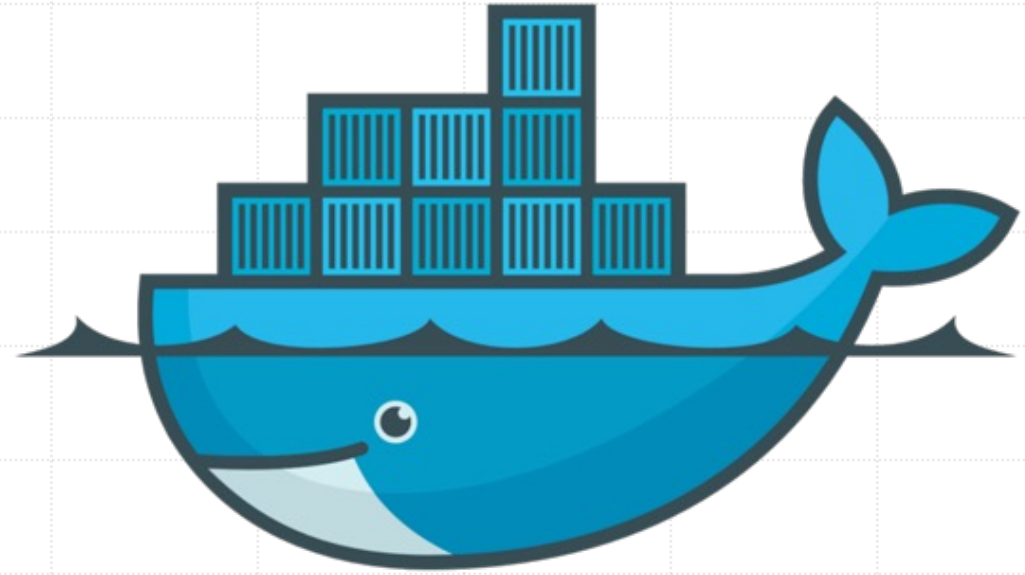
- Creación del servidor
- Creación de una ruta en el servidor
- Creación de la ruta para el modelo





# Empaquetamiento con Docker

- Conceptos Clave
- Archivo Dockerfile
- Archivo requirements.txt



docker

# Despliegue de la API con fl0.io

- Creación de la cuenta en fl0.io
- Despliegue del proyecto en fl0.io



# Uso de la API desplegada

- Obtención del endpoint
- Prueba del endpoint con Thunder Client

```
$(window).scrollTop() > header1  
if (parseInt(header1.css('padding-top')) > header1.css('padding-top', '0'))  
}  
else {  
  header1.css('padding-top', '' + header1.css('padding-top') + 10);  
}  
  
$(window).scrollTop() > header2_init  
(parseInt(header2.css('padding-top')) > header2_init) ?  
header2.css('padding-top', '' + header2.css('padding-top') + 10) :  
header2.css('padding-top', '' + header2_init + 10);
```





# Conclusiones

- Un modelo de IA creado con TF puede ser empaquetado en Docker y desplegado en la nube de manera ágil y sencilla para ser consumido mediante una API
- Python + Flask + Docker son un conjunto de herramientas que permiten colocar modelos de IA en producción de manera ágil pero escalable basado en las necesidades del proyecto.
- Las herramientas de la nube tipo PaaS/SaaS aceleran el proceso de despliegue de aplicaciones sin preocuparse por temas de "bajo nivel"

► Preguntas?





**Gracias!**