Cómo crear una ejecución

# Paso 1: Tener código del modelo a utilizar en Python o R

En este caso, el código está echo en Python, es una regresión lineal simple, la cual predecirá el salario de un empleado en base a su edad, el código será el siguiente:

|  |
| --- |
| # Regresión Lineal Simple  # Cómo importar las librerías  import numpy as np  #import matplotlib.pyplot as plt  import pandas as pd  # Importar el data set  dataset = pd.read\_csv('Salary\_Data.csv')  X = dataset.iloc[:, :-1].values  y = dataset.iloc[:, 1].values  # Dividir el data set en conjunto de entrenamiento y conjunto de testing  from sklearn.model\_selection import train\_test\_split  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 1/3, random\_state = 0)  # Crear modelo de Regresión Lienal Simple con el conjunto de entrenamiento  from sklearn.linear\_model import LinearRegression  regression = LinearRegression()  regression.fit(X\_train, y\_train)  # Predecir el conjunto de test  y\_pred = regression.predict(X\_test)  print(y\_pred) |

# Nota:

Tomar en cuenta cuales son las librerías que se deberían utilizar, ya que este programa se ejecutará en una imagen de Docker, la cual, para funcionar necesitará que se le indiquen las librerías que se están utilizando. El documento que indicará qué librería se debe de instalar en la imagen de Docker, será un archivo tipo Yaml.

En este caso las librerías usadas son:

1. Pandas
2. Numpy
3. Sklearn

# Paso 2: Tener archivo Yaml con codificación correcta

Este archivo Yaml, se puede generar con Notepad ++, en la sección de lenguajes. Cuando se tenga configurado, proceder a escribir el siguiente texto:

|  |
| --- |
| ---  - step:  name: Train MNIST model  image: tensorflow/tensorflow:2.0.1-gpu-py3  command:  - pip install matplotlib  - pip install pandas  - pip install scikit-learn  - python simple\_linear\_regression.py |

Name: en este espacio se dejará el nombre que se le quiere dar al programa a ejecutar en el pipeline de Valohai.

Image: La imagen de Docker que se usará para hacer funcionar el programa de Python o R, el nombre de la imagen puede variar en base al problema a resolver.

Command: Debe escribir con comandos pip que librerías quiere que se instalen en la imagen de Docker que hará funcionar su programa.

# Paso 3: Tener un repositorio con los archivos que solicita Valohai.

Los archivos que Valohai necesita son los siguientes:

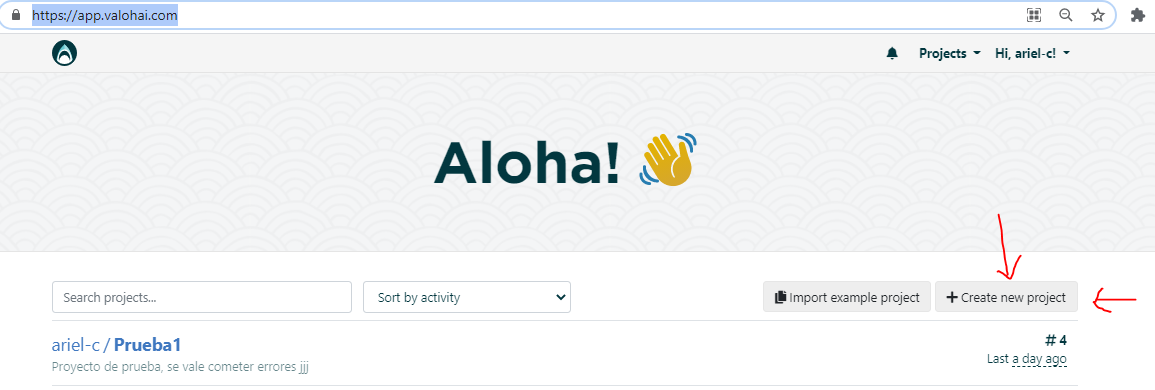
1. Archivo con código en Python o R que ya se halla comprobado su funcionamiento.
2. Archivo tipo Yaml con la configuración correcta.
3. Dataset que servirá para entrenar el modelo

El repositorio utilizado para este ejemplo es el siguiente:

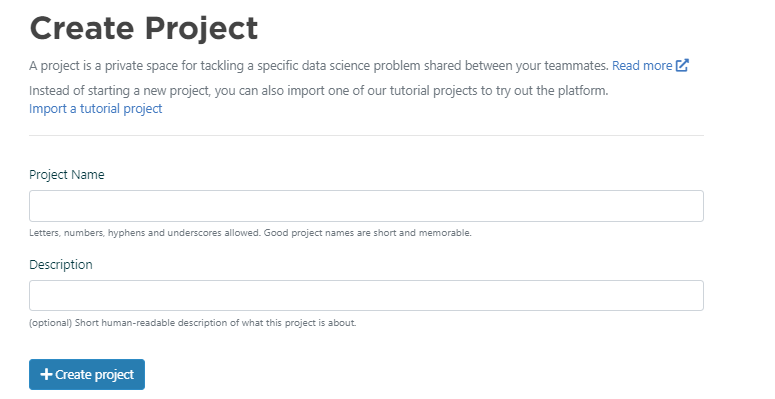
<https://github.com/ariel91/ValohaiPrueba>

# Paso 4: Crear un nuevo proyecto en la plataforma Valohai

* Acceder a <https://app.valohai.com/> crear tu usuario y contraseña e ir a la sección de crear nuevo proyecto como se presenta a continuación



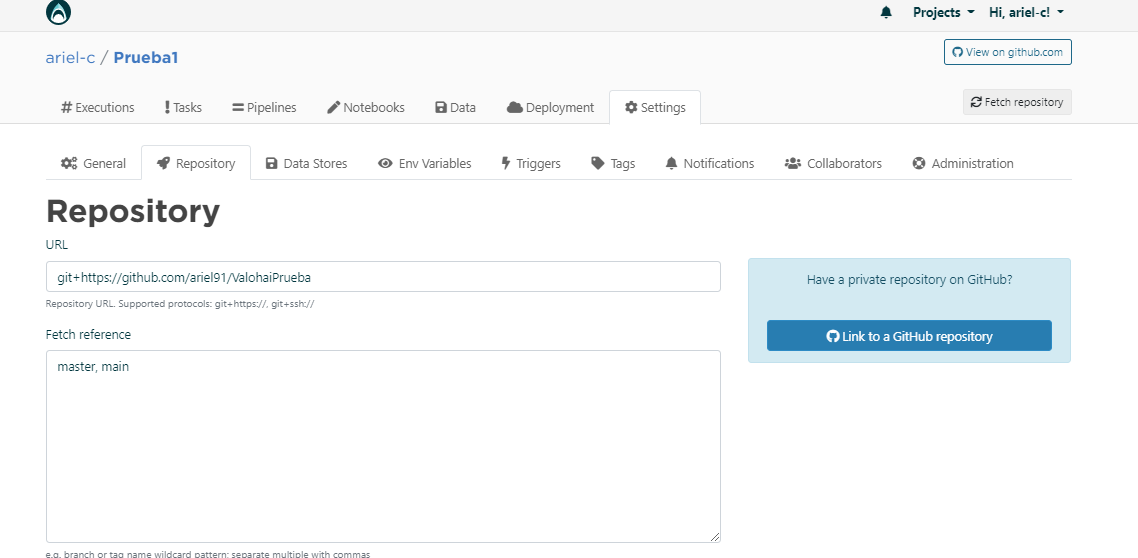
* Colocarle nombre al proyecto y su respectiva descripción



# Unir la sección de Valohai con el repositorio de github

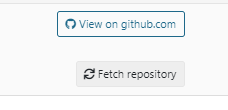
Ir a la sección de repositorio: https://app.valohai.com/p/[NombreUsuario]/[NombreProyecto]/settings/repository/

Y colocar el link del repositorio en el espacio indicado



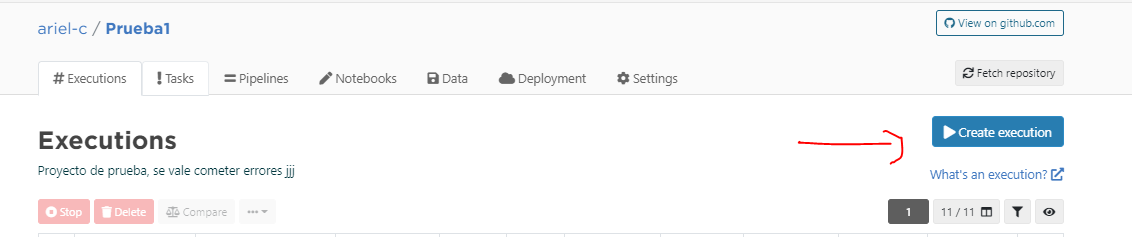
Y guardar.

Cuando haga una actualización en el repositorio, no olvide dar clic en Fetch Repository, en la esquina superior derecha de la plataforma.

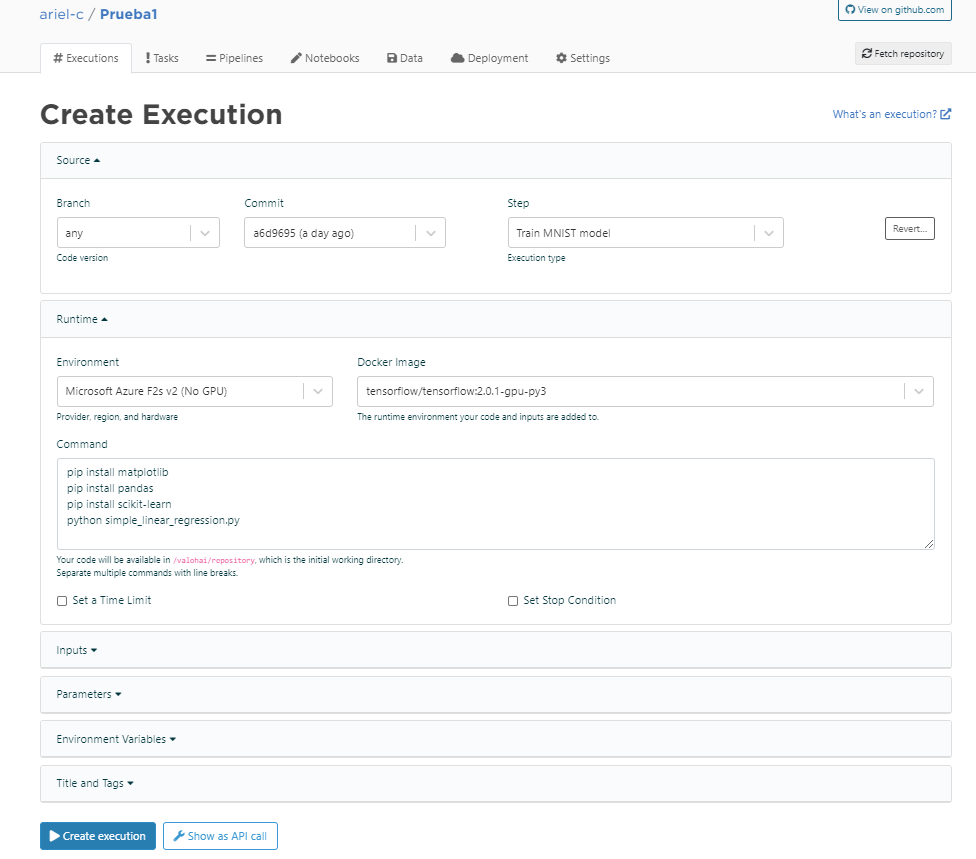


# Paso 4: Crear primera ejecución en Valohai

Una vez tenga vinculado su repositorio a la plataforma de Valohai, deberá crear una ejecución. Esto lo logrará en el apartado: [https://app.valohai.com/p/[usuario]/[NombreProyecto]/](https://app.valohai.com/p/%5busuario%5d/%5bNombreProyecto%5d/)



Donde se le dará una pequeña descripción del espacio virtual se le está dando para trabajar con todas las características que ya ha configurado en los espacios anteriores.



Antes de crear la ejecución, tome un momento para notar si todo lo indicó en su Yaml es suficiente para que su programa se ejecute. Si la ejecución se creará correctamente, saldrá un mensaje en verde, de lo contrario saldrá rojo. Por lo que habrá finalizado su primera ejecución.

