

# Vertex AI - Course Guide

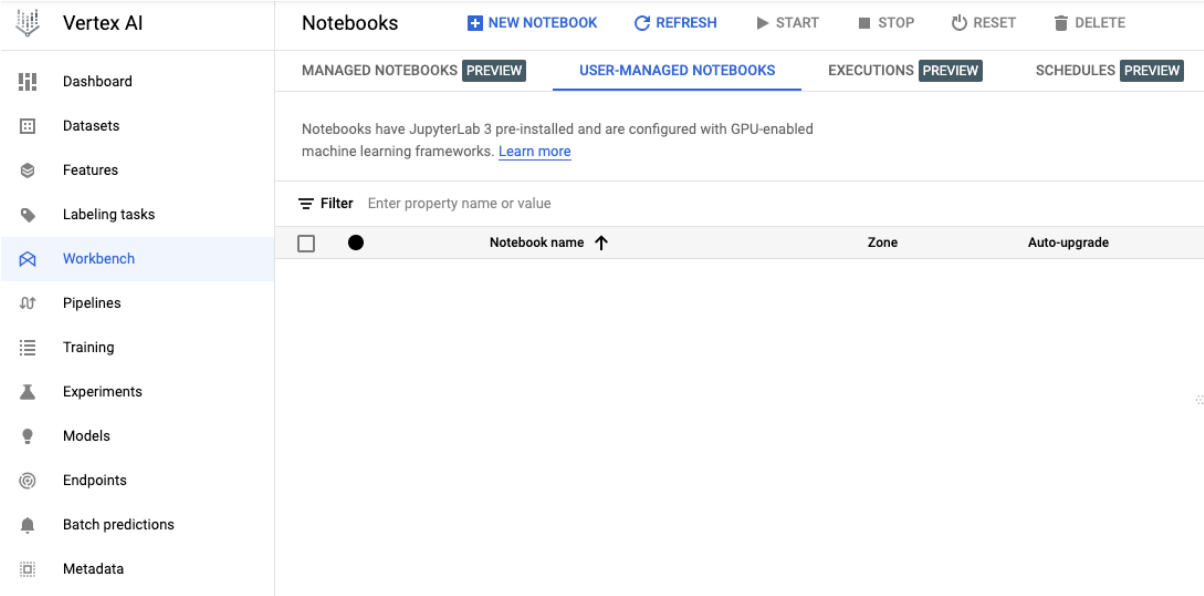
## AutoML

Cuando utilizar AutoML

- Tenemos un problema que involucra una de estas opciones.
  - Imágenes
    - Clasificación (A nivel de imagen)
    - Detección de objetos (A nivel de sección)
    - Segmentation (A nivel de pixel)
  - Texto
    - Clasificación de texto
    - Extracción de entidades
    - Analisis de sentimiento
  - Datos tabulares (CSV)
    - Regresión
    - Clasificación
    - Predicción (Forecasting)
  - Video
    - Clasificación
    - Reconocimiento de acciones
    - Seguimiento de objetos (Object tracking)
- El problema es básico.
- El equipo es pequeño o no tiene experiencia.
- Se busca un prototipo o prueba de concepto rápida.
- El modelo no necesitaria ser mejorado

## Notebooks

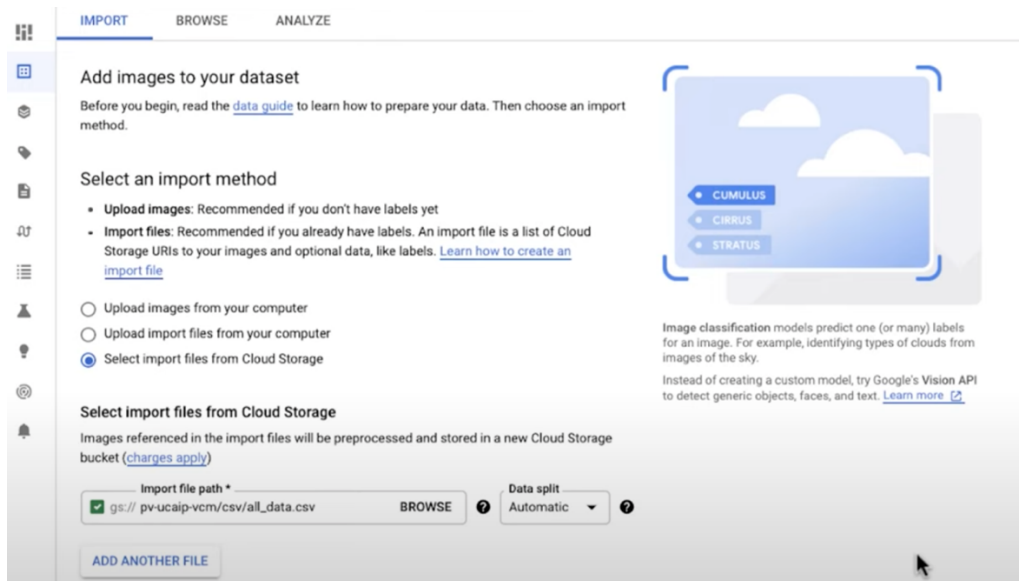
Los notebooks no son más que Jupyter Lab notebooks integrados en Vertex AI. Estos notebooks corren en máquinas que nosotros mismos configuramos. Podemos poner en marcha y parar estas máquinas cuando queramos (En teoría).



## Datasets

En esta sección es donde gestionaremos el proceso de convertir los datos que tenemos en un dataset valido para su uso en Vertex AI. Este proceso puede ser realizado desde la CLI o desde la interfaz, a continuación, se muestran unas capturas de los pasos a realizar en la interfaz.

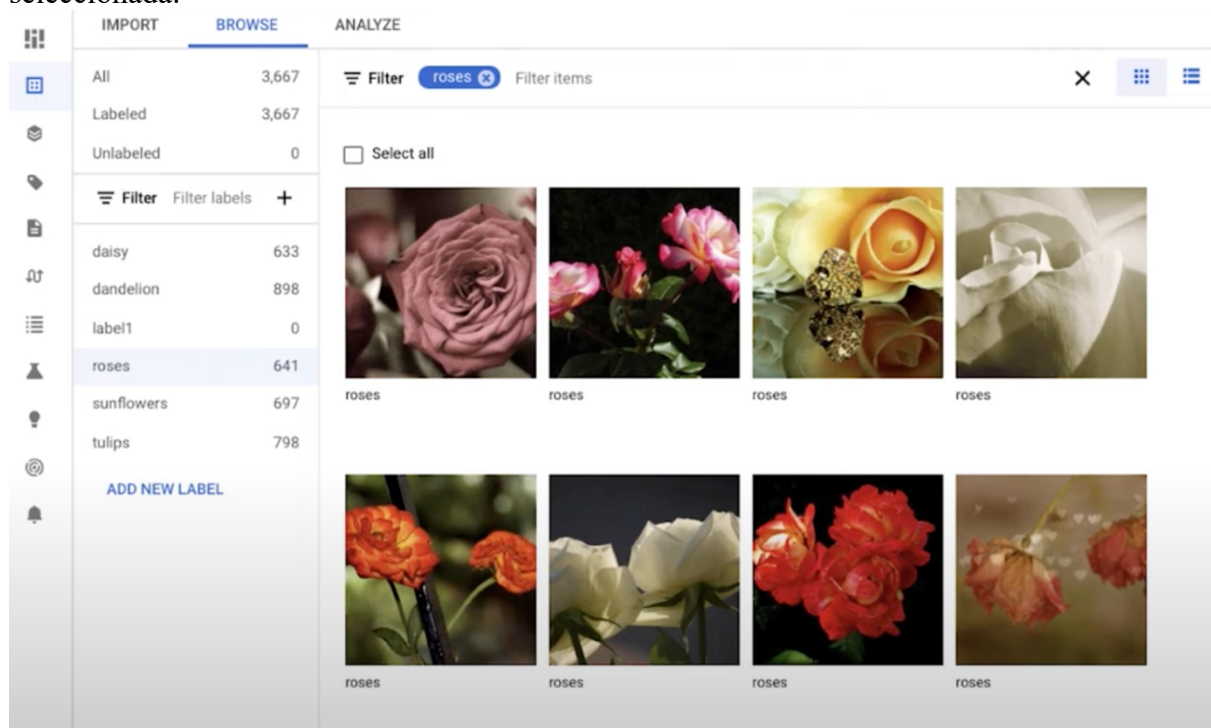
La sección de datasets tiene el siguiente aspecto.



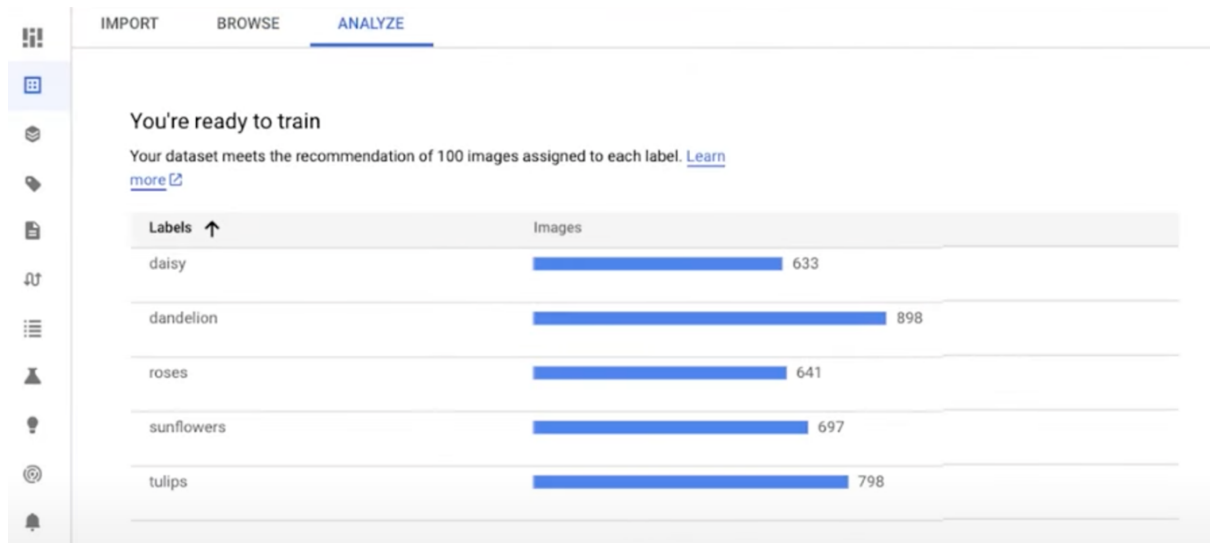
Podemos observar que tenemos diferentes opciones para importar datos. En este caso se trata de un dataset de imágenes.

- Podemos subir las imágenes directamente desde nuestro ordenador.
- Subir el *import file* que más adelante explicaremos.
- Seleccionar los datos desde en el Cloud Storage (Esto implica que previamente han sido subidos)

Una vez realizamos la importación tendremos una sección de Browse/Exploración donde podemos ver las clases, cuantas imágenes por clase e imágenes de la clase que tenemos seleccionada.



Por último, tenemos una sección de Analyze/Análisis donde podemos ver un breve resumen del dataset. Esta sección es particularmente interesante si previamente no hemos realizado un análisis por nuestra cuenta.



Aquí mostramos otro ejemplo del aspecto que tendría un dataset de tipo texto. Vemos que tenemos distintas clases, y una selección de ejemplos de dichas clases, muy similar al caso de imágenes.

Filter	Filter labels	Filter items	Labels
<input type="checkbox"/>		Text	
<input type="checkbox"/>	▼	My eldest son who is 27 just got word he has a new job after finishing his bache...	achievement
<input type="checkbox"/>	▼	I visited my best friend at her school on St. Patrick's day.	bonding
<input type="checkbox"/>	▼	My mom cooked some delicious rice for me with curd.	affection
<input type="checkbox"/>	▼	Today I make Eye contact with my crush. She Also look into my Eyes For a Seco...	affection
<input type="checkbox"/>	▼	I was dropping off my son for a sleepover. He was really excited to go. I dropped...	affection
<input type="checkbox"/>	▼	Dinner tonight was really good.	leisure
<input type="checkbox"/>	▼	I WENT TO MEENAKSHI AMMAN TEMPLE WITH MY FAMILY MEMBERS.	affection
<input type="checkbox"/>	▼	I got the test results back from my father's echo and neck arteries taken at the d...	affection
<input type="checkbox"/>	▼	I was selected as the winner for a random lottery drawing from an mturk hit. It w...	achievement
<input type="checkbox"/>	▼	My brother told me he got into med school!	affection

Items per page: 10 1 - 10 of many

Una vez tenemos un dataset configurado en Verte AI, a la izquierda de la pantalla tendremos un botón que nos permitirá comenzar un nuevo entrenamiento.

Column name	Missing % (count)	Distinct values
Aspect	-	358
Color_Type	-	2
Elevation	-	826
Hillshade_3m	-	233
Hillshade_9m	-	171
Hillshade_18m	-	133
Horizontal_Distance_To_Fire_Points	-	850
Horizontal_Distance_To_Hydrology	-	156
Horizontal_Distance_To_Roadways	-	865
Slope	-	49
Soil_Type_0	-	2
Soil_Type_1	-	2
Soil_Type_10	-	2
Soil_Type_11	-	1

# AutoML

## Ejemplo con datos tabulares

- La primera línea del CSV ha de ser la cabecera, estos son los nombres de las columnas
- Las columnas pueden contener caracteres alfanuméricos y la barra baja "\_" (no puede comenzar por)
- Cada CSV no puede pasar de 10GB, si pesa más de 10GB lo puedes repartir en varios CSV hasta un máximo de 100GB
- El delimitador ha de ser la coma ",".
- Al menos 1000 filas para Tabular Data, 100 imágenes por clase para Vision AI
- El CSV se sube con la variable ha predecir incluida

No hace falta delimitar el schema del CSV (si las columnas son enteros, flotantes, strings..., etc), Vertex AI lo hace por ti. Se puede repartir los datos entre entrenamiento, validación y test de forma automática o manual

Vamos a entrenar un modelo con datos tabulados. Tenemos un ejemplo en el notebook llamado 2-AutoML que está entrenando, usando la famosa librería Scikit-Learn.

El ejercicio consiste en preparar los datos que usamos en este mismo ejercicio, crear un dataset en Vertex AI y entrenar un modelo.

## Ejemplo con imágenes

Para este ejercicio, vamos directamente a la documentación para aprender por nuestra cuenta como realizar este caso. La documentación la podemos encontrar en el [LINK](#) los pasos son los siguientes

- Creamos un dataset subiendo las imágenes ([Descargar](#))
- Etiquetamos las imágenes.

Buckets > example\_od\_1 > Obama001.jpg

LIVE OBJECT

VERSION HISTORY

DOWNLOAD

EDIT METADATA

EDIT ACCESS

DELETE

Overview

Type	image/jpeg
Size	1.2 MB
Created	Jan 29, 2022, 7:00:24 PM
Last modified	Jan 29, 2022, 7:00:24 PM
Storage class	Standard
Custom time	—
Public URL ?	Not applicable
Authenticated URL ?	<a href="https://storage.cloud.google.com/example_od_1/Obama001.jpg">https://storage.cloud.google.com/example_od_1/Obama001.jpg</a>
gsutil URI ?	gs://example_od_1/Obama001.jpg

Permissions

Public access	Not public
---------------	------------

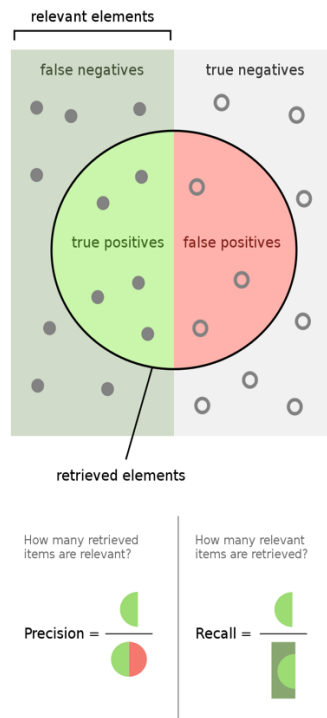
Protection

Hold status	None
Version history ?	—
Retention policy	None
Encryption type	Google-managed key



# Métricas

Como interpretar los resultados, dependiendo del *dataset* obtendremos unos tipos de resultados u otros. En el caso de datos tabulados por ejemplo tendremos las siguientes métricas.



$$\text{Precision} = \text{TP} / (\text{FN} + \text{FP})$$

$$\text{Recall} = \text{TP} / (\text{FN} + \text{TP})$$

$$\text{F1 Score} = 2 \frac{P R}{P + R}$$

## Precision Recall Curve

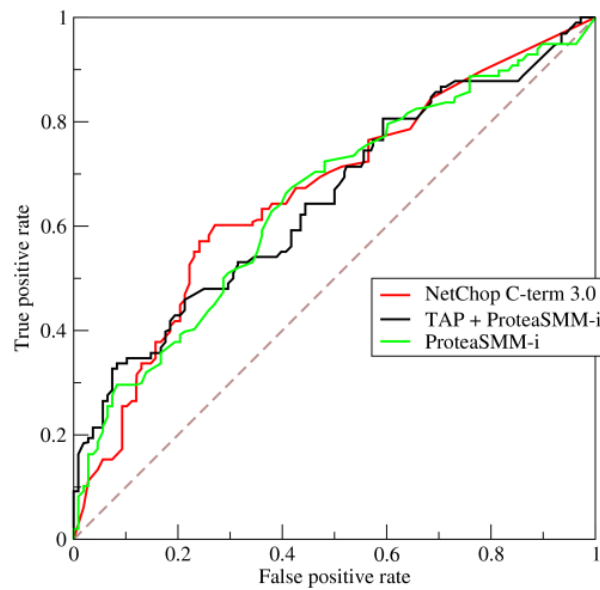
Para los problemas binarios de clasificación con clases desbalanceadas esta curva es muy útil para rápidamente de un vistazo observar si las dos clases son correctamente clasificadas o de lo contrario hay una tendencia a clasificar la clase mayoritaria como comúnmente ocurre en estas situaciones.

## ROC AUC

Para problemas de clasificación binarios tenemos una gran métrica entre TPR (*True Positive Rate*) y FPR (*False Positive Rate*)

$$\text{TPR} = \text{TP} / (\text{TP} + \text{FN})$$

$$\text{FPR} = \text{FP} / (\text{FP} + \text{TB})$$



## Matriz de confusion

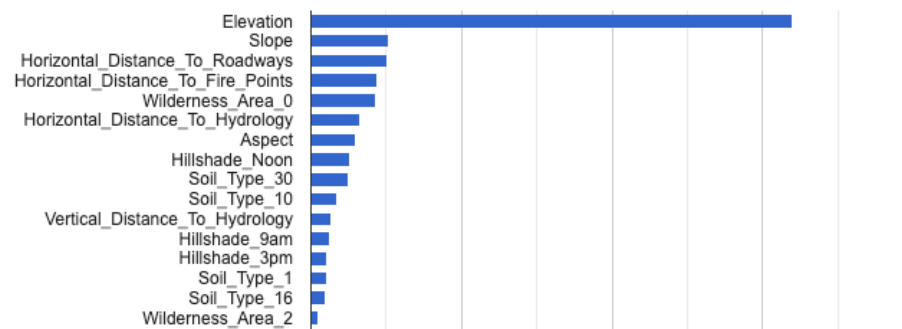
True label	Predicted label	
	0	1
0	100%	0%
1	15%	85%

Tenemos los porcentajes de TP FP TN FN respectivamente en una matriz. Este caso es binario, para casos de más clases tendremos una matriz de NxN donde N es el número de clases.

## Feature Importance

Ranking de importancia de las variables para lograr la clasificación del modelo.

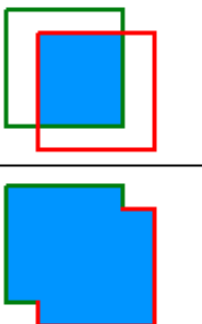
### Feature Importance

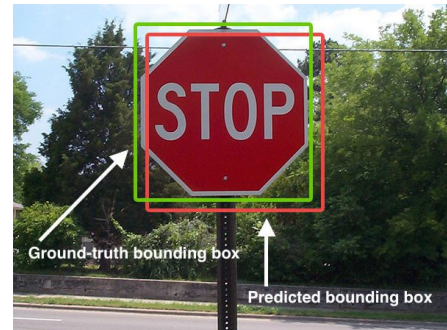




## IoU Intersection Over Union

Esta es una métrica exclusivamente de computer visión. Se trata de una métrica de similitud, por lo tanto, cuanto mayor es la cifra mayor similitud. Su valor esta acotado entre 0 y 1.

$$IOU = \frac{\text{area of overlap}}{\text{area of union}} =$$




# Deploy

Una vez entrenado un modelo, en la sección de models seleccionamos el modelo el cual deseamos crear un endpoint.

### Deploy to endpoint

✓ Define your endpoint

**2 Model settings**

3 Model monitoring

DEPLOY

CANCEL

### Model settings ?

2Ejercicio\_202212417569

Traffic split \*  
100 % ?

#### Compute resources

Choose how compute resources will serve prediction traffic to your model

- **Autoscaling:** If you set a minimum and maximum, compute nodes will scale to meet traffic demand within those boundaries
- **No scaling:** If you only set a minimum, then that number of compute nodes will always run regardless of traffic demand (the maximum will be set to minimum)

Once scaling settings are set, they can't be changed unless you redeploy the model. [Pricing guide](#)

Minimum number of compute nodes \*  
1

Default is 1. If set to 1 or more, then compute resources will continuously run even without traffic demand. This can increase cost but avoid dropped requests due to node initialization.

Maximum number of compute nodes (optional)

Enter a number equal to or greater than the minimum nodes. Can reduce costs but may cause reliability issues for high traffic.

✓ ADVANCED SCALING OPTIONS

Machine type \*  
n1-standard-8, 8 vCPUs, 30 GiB memory

#### Logging

Logging settings are permanent for this endpoint, and Cloud Logging charges will apply. To change your logging preference in the future, create a new endpoint. [Learn more](#)

☒ Enable access logging for this endpoint

☐ Disable container logging for this endpoint

**Traffic Split**, esta variable puede ser entendida en el siguiente escenario.

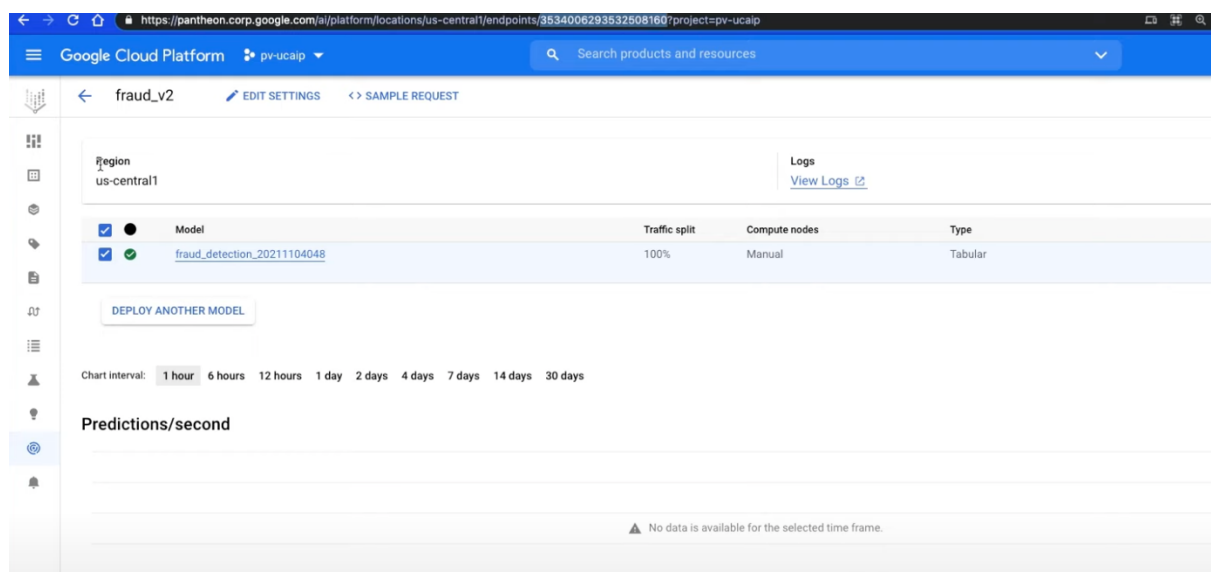
Cuando tenemos un modelo en un endpoint y queremos realizar una actualización de dicho modelo, sí cambiamos el modelo en el endpoint lógicamente va a devolver resultados que

pueden variar con el modelo anterior. Para evitar cambios abruptos en los resultados podemos tener dos modelos en un mismo endpoint. Traffic Split nos permite desviar un porcentaje de las peticiones a un modelo e ir incrementado con el paso del tiempo, permitiendo una transición entre modelos mucho más suave, esto puede ser de especial importancia cuando es un cliente el que usa el endpoint.

**Minimum de compute nodes** indica como mínimo cuantos modos vamos a tener en funcionamiento en todo momento, con el consecuente gasto económico.

**Maximum compute nodes** indica cuantos nodos como máximo podemos tener, esto es especialmente útil para situaciones donde queremos autoscaling.

Cuando tenemos el endpoint creado, si pulsamos en el veremos la información básica que lo define, si nos fijamos en la URL tendremos un número, como en la siguiente foto.



Una vez desplegado el endpoint, podemos realizar queries desde la interfaz o desde código Python.

Si necesitamos procesar datos en forma de batch tenemos la opción Batch Processing desde la interfaz.

# Custom model (Avanzado)

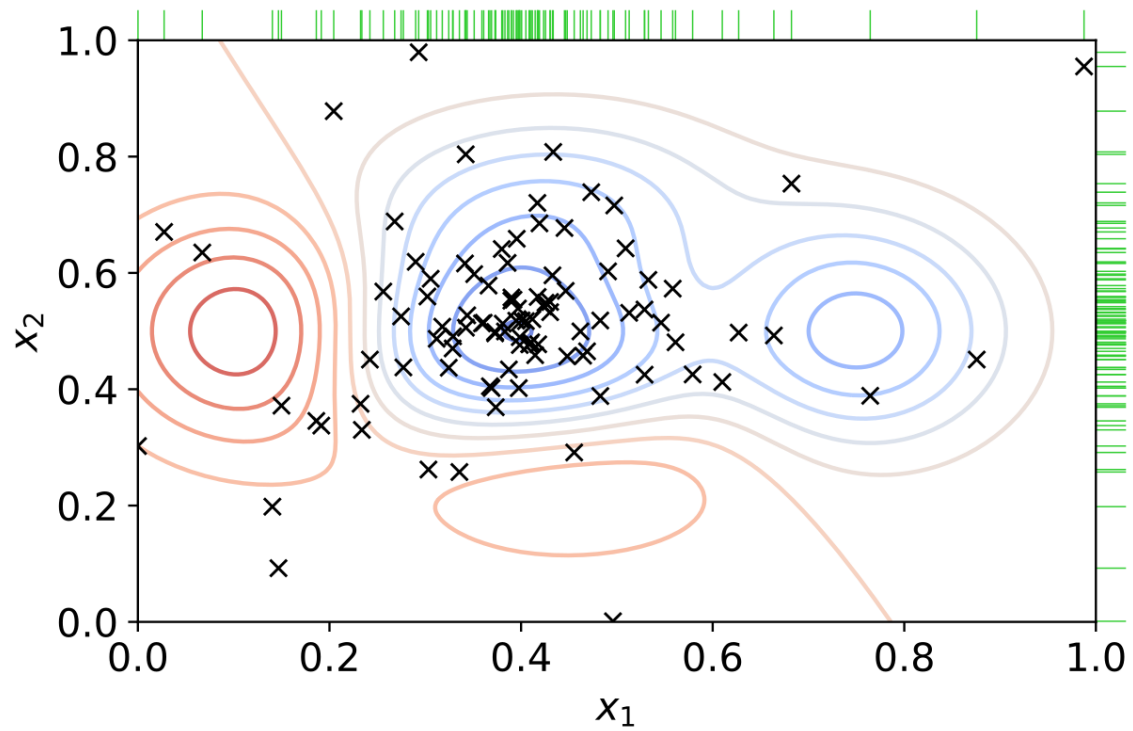
Cuando utilizar emplear un custom model

- Tenemos un problema que involucra más de un problema elemental (ie. Imágenes con datos tabulados)
- El problema no es básico.
- Necesitamos un control particular sobre el modelo, como el framework (TensorFlow, PyTorch, Scikit-Learn)
- El equipo que se encarga del modelo no es pequeño.
- El equipo que se encarga del modelo tiene conocimientos.
- Hay tiempo para realizar ajustes y mantenimientos.
- El modelo ha de poder ser mejorado.

¿Qué necesitamos?

- Training Package
  - Pre-built (Tensorflow, Scikit-learn, PyTorch)
  - Container o Artifact Registry (Custom Container)
- Model Artifacts
  - Se deben de guardar en un bucket del Storage
- Hyperparameter Tuning
  - Cambiar parametros de entrenamiento, el sistema se encarga de buscar los mejores parametros que den el mejor resultado
- Compute resources
  - Para entrenar el modelo debemos asignar un máquina.
- Endpoint
  - Deploy pre-built (Tensorflow, Scikit-Learn, PyTorch)
  - Deploy Custom

# HP Tuning (Avanzado)



[GCP HP Tuning](#)

[Third Party HP Tuning](#)

# Edge Computing

- [Coral.ai](#)
- [Jetson Nano](#)