

Herramientas para la Computación en la Nube Dirigida a  
Inteligencia Artificial

---

## Tema 2. Creación de modelos de IA en Google Cloud con Vertex AI

# Índice

## Ideas clave

- 2.1. Introducción y objetivos
- 2.2. Almacenamiento y transformación de conjuntos de datos
- 2.3. Características del entorno de creación de modelos AI Workbench
- 2.4. Creación de modelos con AutoML
- 2.5. Creación de modelos con BigQuery Machine Learning
- 2.6. Interpretación de modelos
- 2.7. Optimización de modelos mediante ajuste de hiperparámetros
- 2.8. Cuaderno de ejercicios
- 2.9. Referencias bibliográficas

## A fondo

The definitive guide to Google Vertex AI: accelerate your machine learning journey with Google Cloud Vertex AI and MLOps best practices

Machine learning with BigQuery ML: create, execute, and improve machine learning models in BigQuery using standard SQL queries

## Test

## 2.1. Introducción y objetivos

En este tema, se describen las funcionalidades que proporciona la plataforma de inteligencia artificial de Google: **Vertex AI**. Esta comprende todas las fases de creación de un modelo, desde la ingestión, almacenamiento y transformación de los datos hasta el ajuste de hiperparámetros de los modelos para encontrar aquel que presenta mejor rendimiento.

Vertex AI ofrece la posibilidad de trabajar en **modelos personalizados** con herramientas a bajo nivel, como Jupyter, y también capacidades de AutoML, que generan modelos de forma automática, sin necesidad de conocer un lenguaje de programación o disponer de conocimientos avanzados en materia de ciencia de datos.

También proporciona un entorno, **BigQuery Machine Learning**, que permite construir modelos a partir de consultas SQL. De esta forma, los usuarios pueden explotar sus conocimientos en bases de datos para crear modelos de forma ágil.

Por último, Vertex AI proporciona mecanismos de explicabilidad de los modelos y de las predicciones individuales generadas por estos para garantizar la identificación de sesgos y facilitar el uso de una IA responsable.

Los objetivos de este tema son los que se indican a continuación:

- ▶ Identificar las herramientas disponibles en Vertex AI y cuáles deben utilizarse en función de la problemática que se desea resolver.
- ▶ Que el alumno adquiera la habilidad para ejecutar todas las fases de un proceso de creación de modelos en la plataforma Vertex AI.

- ▶ Utilizar las funcionalidades diferenciales de Vertex AI respecto a la infraestructura *on premise* para ser capaces de materializar las ventajas de computación en la nube en el ámbito de la IA.

## 2.2. Almacenamiento y transformación de conjuntos de datos

### Transferencias de datos a Google Cloud

Para empezar a trabajar en Vertex AI, el primer paso es transferir los datos al entorno de Google Cloud. Este proporciona **cuatro herramientas** para este fin.

### Google Cloud Storage Transfer Tools

Es la opción recomendada para **volúmenes pequeños** (<10 TB). Los datos transferidos se almacenan en un *bucket* de Google Cloud Storage. Un *bucket* es un contenedor de ficheros independiente, similar al sistema de ficheros de un ordenador.

Se pueden utilizar los siguientes métodos para enviar ficheros o carpetas a un *bucket* de GCS:

- ▶ **Desde Google Cloud Console.** En el navegador web, accedemos a la Google Cloud Console y seleccionamos «Cloud Storage». A continuación, seleccionamos «Buckets». Desde esta página es posible ver los *buckets* existentes o crear uno nuevo. Si hacemos clic en un *bucket*, podemos subir ficheros o carpetas con los botones «Upload Files» o «Upload Folder», respectivamente.
- ▶ **Línea de comandos.** Se emplea la utilidad *open-source* GSUTIL. Por ejemplo, para transferir un fichero a GCS:

```
$ gsutil cp Carpeta/fichero.txt gs://mibucket
```

- ▶ **A través de REST API.** Para consumir la API, se lanzan peticiones HTTP con la sintaxis requerida desde el código fuente de un programa. Por ejemplo, para visualizar los ficheros y carpetas de un *bucket*:

```
GET https://storage.googleapis.com/storage/v1/b/mibucket/o
```

## Big Query Data Transfer Service

Se utiliza para transferir datos de forma **automática y programada** a BigQuery, el entorno de *data warehouse* de Google Cloud. Una vez configurado, la transferencia se realiza de acuerdo con los parámetros y periodicidad especificados. A diferencia del anterior, por tanto, la transferencia no se ejecuta de forma manual por el usuario.

Este servicio puede configurarse, como GCS Transfer Tools , a través de la consola de Google Cloud, la línea de comandos o vía API. Dispone de integraciones para transferir datos desde **múltiples entornos**, como otros servicios de Google (Play, Ads, Merchant, YouTube), sistemas de almacenamiento de otros proveedores de nube (Amazon, Azure, Salesforce), ServiceNow o Teradata.

## Storage Transfer Service

Es un servicio especializado en transferencias entre entornos de nube de diferentes proveedores (Azure, Amazon) o entre la nube y un centro de datos. Proporciona **trazas** (*logs*) de las transferencias realizadas y dispone de mecanismos para hacer varios intentos de envío en caso de error.

## Transfer Appliance

Se trata de un *appliance* de **almacenamiento físico de datos** que puede solicitarse a Google. Una vez recibido, se copian los datos en el dispositivo y se devuelve a Google para que vuelque los datos en Google Cloud Storage. Solo se utiliza para grandes volúmenes de datos o cuando no se dispone del ancho de banda suficiente.

## Almacenamiento de datos en Google Cloud

Se utilizan dos entornos: **Google Cloud Storage (GCS)** y **BigQuery**. El primero es más recomendable para datos de imagen, vídeo, audio o cualquier otro tipo de dato no estructurado. El segundo se emplea para datos estructurados o semiestructurados, característicos de los algoritmos de *machine learning*.

### Google Cloud Storage

Es un servicio de almacenamiento basado en **objetos**, una estructura de datos especializada en el almacenamiento eficiente de datos no estructurados a gran escala. Se utiliza en escenarios en los que los datos se escriben una sola vez, pero se leen frecuentemente. Es eficaz con **datos estáticos**: si los datos cambian con frecuencia, el rendimiento de este tipo de almacenamiento se degrada considerablemente.

Las principales **ventajas** incluyen:

- ▶ Puede escalar tanto como sea necesario para **incrementar** la capacidad de **almacenamiento**.
- ▶ Gracias a los metadatos y los *tags*, es fácil **localizar ficheros** a través de búsquedas y filtros.
- ▶ Los datos se almacenan de forma redundante en **múltiples dispositivos o regiones**, incrementando la tolerancia a fallos y desastres.
- ▶ El coste del servicio es **bajo** y ajustado a la capacidad de almacenamiento que realmente se utiliza.

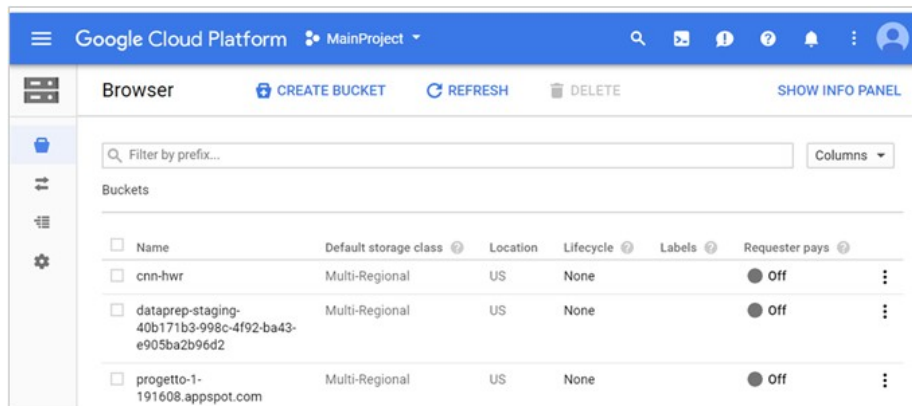


Figura 1. Interfaz de usuario de Google Cloud Storage en Google Cloud Console. Fuente: Ciaburro, Ayyadevara y Perrier, s. f.

## BigQuery

Es el **entorno de data warehouse** de Google Cloud. Soporta capacidades de almacenamiento del orden de petabytes. Utiliza un formato de almacenamiento basado en columnas, que es especialmente eficaz para procesamiento analítico de datos.

Los datos se muestran de la misma forma que en una **base de datos**, con tablas, filas y columnas. También implementa las propiedades ACID (atomicidad, consistencia, aislamiento y durabilidad) que encontramos en sistemas de bases de datos transaccionales. Además, proporciona un elevado grado de redundancia mediante replicación en múltiples ubicaciones y regiones.

Permite el análisis de grandes *datasets* y la creación de modelos de *machine learning* a través de consultas SQL. Los modelos construidos pueden exportarse al servicio de predicciones de Vertex AI para completar su operacionalización.

Por último, BigQuery también soporta la ingestión de **datos en streaming** y su análisis en tiempo real, así como el análisis de datos que se almacenan en entorno externos, sin necesidad de traerlos a Google Cloud. Esta característica es fundamental para la reducción de costes y el cumplimiento regulatorio.



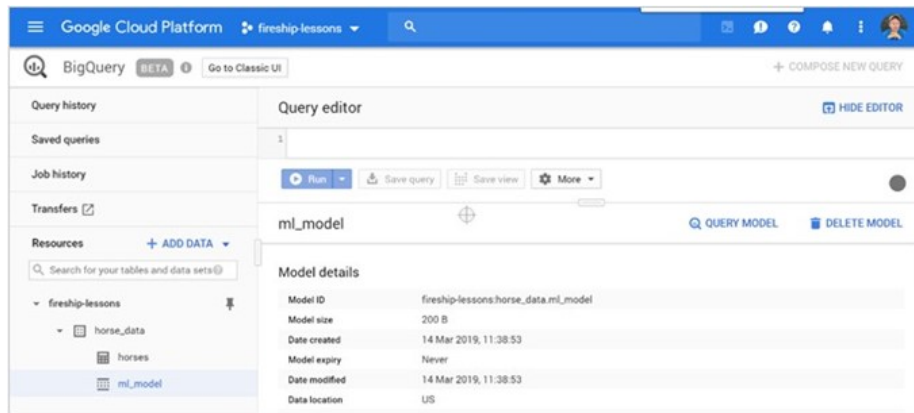


Figura 2. Interfaz de usuario de BigQuery ML con los detalles de un modelo. Fuente: Delaney, 2019.

## Transformación de datos en Google Cloud

Para utilizar algoritmos de *machine learning* en Vertex AI, es necesario aplicar un **proceso de transformación** de los datos, cuyas características dependen del tipo de algoritmo utilizado y del propósito del modelo que se construye a partir de él. Google Cloud proporciona varios mecanismos para llevar a cabo este proceso.

## Jupyter Notebooks

Los datos almacenados en Google Cloud Storage o BigQuery pueden leerse desde un cuaderno de Jupyter para aplicarles el tratamiento necesario. El resultado de este puede a su vez almacenarse en el entorno correspondiente (GCS o BQ) para ser utilizado posteriormente en el proceso de creación de modelos.

También es posible aplicar diferentes técnicas de transformación al mismo conjunto de datos y guardarlos con distintos **números de versión** para poder probar después cuáles de estas transformaciones arrojan mejores resultados de rendimiento con los algoritmos utilizados.

Las transformaciones habituales en datos numéricos son la **normalización** (escalado) o el **binning** (asignación a intervalos), mientras que en variables

categorías se utilizan técnicas como codificación de etiquetas, *one-hot encoding* o *embeddings*. Todas ellas pueden realizarse desde el cuaderno Jupyter con las librerías adecuadas del lenguaje de programación utilizado (habitualmente Python).

### Cloud Data Fusion

Es un servicio que permite llevar a cabo las transformaciones de los datos sin necesidad de escribir código fuente. Se utiliza una interfaz de usuario web en la que es posible definir un *pipeline* personalizado para limpiar, preparar, combinar o transformar datos. Para ello, dispone de un catálogo extenso de **conectores** con transformaciones ya implementadas, que pueden aplicarse directamente sobre los datos.

La creación o gestión de estos *pipelines* puede realizarse también mediante línea de comandos. Soporta la creación de ***pipelines* ETL** (*extract, transform, load*) o **ELT** (*extract, load, transform*). Además, puede utilizarse con datos en *batch* o en tiempo real.

### Dataflow Pipelines

Es un servicio especializado en el procesamiento y transformación de datos mediante ***pipelines* de Apache Beam**, una solución *open-source* especializada en estas actividades. Los *pipelines* también pueden crearse mediante consultas SQL en BigQuery.

Permite el **procesamiento *batch* y en tiempo real**, y es especialmente recomendable para casos de uso como detección de anomalías, *forecasting* de series temporales y sincronización de datos entre múltiples fuentes de datos. Por último, puede escalar de forma automática en función de las necesidades del *pipeline* en el momento de la ejecución.

## 2.3. Características del entorno de creación de modelos AI Workbench

### Jupyter Notebooks

Jupyter es una aplicación web *open-source* para escribir y probar **código fuente** de forma dinámica, generar visualizaciones y documentación y compartir los resultados con otros especialistas. Se utiliza en las fases de exploración y preparación de datos y para la experimentación con diferentes algoritmos.

La posibilidad de ejecutar fragmentos del código de forma independiente y de generar visualizaciones al vuelo lo convierten en una herramienta muy útil para el proceso de **comprensión de los datos** y **generación de modelos predictivos**. Por otro lado, la combinación de código y documentación en un único documento facilita la compartición de conocimientos en la comunidad, especialmente en GitHub y Kaggle.

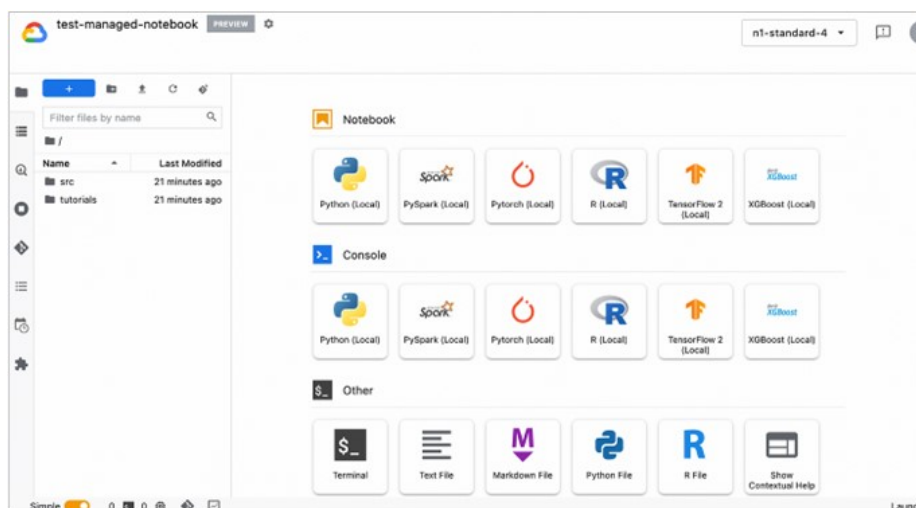


Figura 3. Entorno Jupyter en AI Workbench. Fuente: Keyan, 2022.

## AI Workbench

### Características de AI Workbench

El uso de cuadernos Jupyter en un PC solo permite realizar experimentos a pequeña escala, con volúmenes de datos reducidos y algoritmos que no demandan excesiva capacidad de computación. Cuando se necesita procesar **grandes volúmenes de datos**, con ejecuciones que pueden necesitar mucho tiempo para completarse, la alternativa es utilizar el entorno gestionado de Jupyter que proporciona Google a través de Vertex AI, denominado AI Workbench.

Este entorno, además, proporciona **integración** con los servicios de almacenamiento y transformación de datos vistos en la sección previa, y también proporciona un sistema de permisos granulares que permiten acotar qué puede hacer cada usuario, incrementando con ello la seguridad.

Por último, AI Workbench posibilita la puesta en **producción de los modelos**, de manera que el proceso de generación de modelos se implementa de extremo a extremo a través de un entorno unificado. Por defecto, es posible utilizar uno de los siguientes *kernels* o motores de ejecución: TensorFlow, PyTorch, Spark y R. En cada uno de ellos, se pueden instalar librerías adicionales para cubrir necesidades específicas.

### Tipos de cuadernos

AI Workbench proporciona dos tipos de cuadernos:

- **Gestionados por el usuario.** Recomendado cuando el usuario demanda un elevado grado de personalización. En este caso, el usuario puede elegir las características de la máquina virtual (VM) que ejecuta el entorno y actualizar su tamaño o recursos más adelante, si bien estos cambios requieren reiniciar el entorno. También es posible configurar el uso de GPUs si consideramos que el proyecto obtendrá mejor rendimiento con este tipo de *hardware*. Por último, el usuario puede cambiar

características de *networking* o seguridad y elegir entre múltiples configuraciones de *deep learning* para emplear aquella que mejor se ajuste a su caso de uso.

- **Gestionados por Google.** Recomendado cuando el usuario desea disponer del entorno rápidamente, sin apenas requisitos de personalización. En este caso, el escalado de recursos no requiere reinicio, y es posible visualizar datos de Google Cloud Storage o BigQuery sin abandonar el entorno de Jupyter. También posibilita la ejecución automática de cuadernos en forma de tareas programadas.

En la Figura 4, se muestra la sección de Google Cloud Console, en la que puede seleccionarse el tipo de cuaderno:

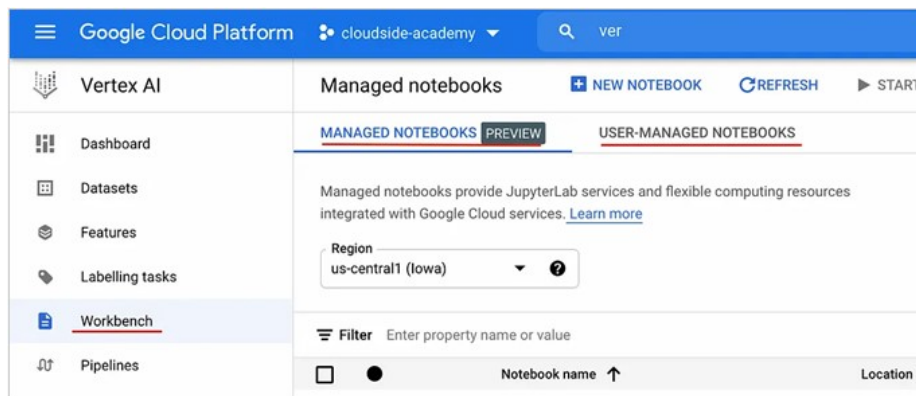


Figura 4. Entorno Jupyter en AI Workbench. Fuente: Keyan, 2022.

## Acceso al entorno

Se puede acceder a AI Workbench desde la Google Cloud Console, seleccionando «Vertex AI» en el menú de productos, y a continuación «Workbench», también en el menú de la izquierda.

En la página web de Workbench, se muestran las **instancias** creadas y se ofrece la posibilidad de crear nuevas. Cada instancia es un entorno independiente que puede albergar múltiples cuadernos.


## Contenedores personalizados

Google Vertex AI ofrece esta opción cuando las posibilidades de personalización de un cuaderno gestionado por el usuario no son suficientes para el caso de uso considerado. Por ejemplo, cuando se requiere una versión de un *software* o *framework* de *deep learning* que no se incluye en el catálogo de AI Workbench. En ese caso, es posible definir un **contenedor personalizado** con todos los paquetes de *software* necesarios, subirlo a Google Cloud y crear una instancia de Workbench basada en ese contenedor.

Para ello, deben ejecutarse los siguientes **pasos**:

- ▶ Crear un contenedor a través de un Dockerfile y una imagen de base.
- ▶ Construir la imagen asociada al Dockerfile y publicarla en el Google Container Registry, un Docker *registry* propio de Google.
- ▶ Las imágenes almacenadas en este *registry* pueden instanciarse a través de Google Compute Engine (GCE).
- ▶ A continuación, se crea un cuaderno gestionado por el usuario y se hace clic en «Customize». En el campo «Environment», se selecciona «Custom container» y se especifica la ruta a la imagen Docker publicada en el *registry*.
- ▶ En el momento de crear el primer cuaderno, se selecciona el contenedor personalizado como *kernel*.

En la Figura 5, se muestra cómo llevar a cabo este procedimiento desde la CLI de Google Cloud como alternativa a la interfaz web:



```
gcloud workbench instances create INSTANCE_NAME \
  --project=PROJECT_ID \
  --location=LOCATION \
  --container-repository=CUSTOM_CONTAINER_URL \
  --container-tag=latest \
  --metadata=METADATA
```

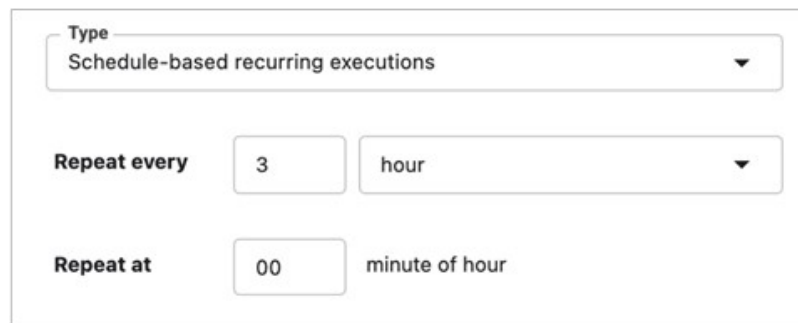
Figura 5. Creación de contenedor personalizado desde CLI. Fuente: Google Cloud, s. f.

### Ejecución programada de cuadernos

Una vez que se completa la fase de experimentación en el cuaderno, es posible lanzar su ejecución para que el contenido se ejecute **celda a celda** tanto de forma **puntual** como **programada**. Esta funcionalidad evita tener que trasladar el código a otro contenedor para su productización, y por ello contribuye a agilizar el proceso general de creación de modelos extremo a extremo.

El cuaderno Jupyter se puede ejecutar pulsando el botón «Execute». Se deben especificar las características de la VM que asumirá la ejecución, la frecuencia de ejecución (salvo que se ejecute una única vez) y el valor de los parámetros que hayamos definido en el código. Los resultados se almacenan en Google Cloud Storage, y pueden visualizarse a través de la pestaña «Executions», de Workbench, haciendo clic en «View Result».

En la Figura 6, se muestra el cuadro de diálogo que permite configurar los parámetros de la ejecución programada:



The image shows a configuration panel for scheduled executions. It contains three main sections: a 'Type' dropdown menu set to 'Schedule-based recurring executions', a 'Repeat every' section with a value of '3' and a unit of 'hour', and a 'Repeat at' section with a value of '00' and a unit of 'minute of hour'.

Type	Schedule-based recurring executions	
Repeat every	3	hour
Repeat at	00	minute of hour

Figura 6. Parámetros de ejecución programada en AI Workbench. Fuente: Namjoshi, 2021.



## 2.4. Creación de modelos con AutoML

### Definición de AutoML

AutoML consiste en la **automatización de todas las etapas de creación de modelos**, incluyendo la preparación de los datos, la ingeniería de variables, la selección de modelos, el ajuste de parámetros y el despliegue en producción.

El AutoML permite poner las capacidades de IA y ML al alcance de profesionales no expertos en estas materias, y también acelerar los procesos de los usuarios más especializados, evitando gran parte del trabajo manual y repetitivo. Puede funcionar con **datos estructurados o no estructurados** y ajustar modelos basados en los algoritmos más habituales.

Con AutoML es posible definir **especificaciones o criterios**, como grado de precisión necesario en los modelos, capacidad de interpretación de las predicciones o tiempo disponible de entrenamiento, para conducir el proceso automático en la dirección que requiere el caso de uso.

### Características de Vertex AI AutoML

Vertex AI proporciona **funcionalidades de AutoML** para creación automatizada de modelos. En particular, provee **soporte** para las siguientes aplicaciones: clasificación y detección de objetos en imágenes, clasificación de textos y extracción de entidades, análisis de sentimiento, clasificación de vídeos, seguimiento de objetos e identificación de acciones en vídeos y modelos de regresión, clasificación y series temporales para datos tabulares.

## Importar datos a Vertex AI AutoML

Para **importar los datos**, se deben ejecutar los siguientes pasos:

- ▶ En Vertex AI, se hace clic en «Datasets», y después se pulsa el botón «Create». Se especifica el nombre del *dataset* y el tipo, por ejemplo, «Tabular para datos tabulares». En función del tipo de datos, se muestran diferentes opciones de modelos. Por ejemplo, para tabular se puede seleccionar «Regression/classification» o «Forecasting» (para series temporales).
- ▶ A continuación, se sube el *dataset* en formato CSV. El contenido se almacena en Google Cloud Storage, en el *bucket* y carpeta especificados.
- ▶ En la sección «Analyze», se hace clic en «Generate statistics» para analizar el contenido de cada variable y extraer detalles relevantes, como el número de valores diferentes y su distribución o el porcentaje de datos ausentes.

Una vez que los datos están disponibles, el siguiente paso es arrancar el proceso de **entrenamiento del modelo**.

## Entrenamiento de modelos

Para llevar a cabo el proceso de entrenamiento de modelos, se deben ejecutar los siguientes pasos:

- ▶ En la sección «Analyze», se hace clic en «Train new model» y se selecciona «Other».
- ▶ Se selecciona el tipo de modelo en el campo «Objective», por ejemplo, «Classification», y la opción de AutoML en el campo «Model training method».

- ▶ En la sección «Model details», se especifica: el nombre del modelo, la columna sobre la que se realizarán las predicciones, la opción de exportar los resultados a BigQuery y la forma en que se distribuirán los datos entre los conjuntos de entrenamiento, validación y pruebas, con la posibilidad de que este paso se realice automáticamente.
- ▶ En la sección «Training options» se seleccionan las variables que se utilizarán para entrenar el modelo. Es importante excluir aquellas que resulten irrelevantes para la tarea de predicción. Por otro lado, Vertex AI identifica automáticamente a qué categoría pertenecen y realiza las transformaciones oportunas. Si es necesario, es posible incluir una columna para asignar pesos a cada observación del **dataset** para rebalancear las clases en tareas de clasificación.
- ▶ Se selecciona la métrica de optimización en «Optimization objective». Las opciones disponibles para clasificación son: AUC ROC, *log loss*, PRC, *precision at recall* y *recall at precision*. En el caso de regresión, son las siguientes: RMSE, MAE o RMSLE.
- ▶ Se selecciona el tiempo máximo de ejecución del proceso de entrenamiento y se hace clic en «Start training». En ese momento, comienza la generación del modelo.
- ▶ Como se ha indicado previamente, Vertex AI aplica automáticamente determinadas transformaciones a las variables en función de su tipología.

Es importante conocer cuáles son para entender cómo se realiza el proceso de entrenamiento. Se describen a continuación:

- ▶ **Variables categóricas:** se genera un índice por cada valor y un *embedding* asociado a cada índice. Los valores que aparecen menos de cinco veces en el *dataset* se asignan a la categoría *unknown*, que también tiene un índice y un *embedding*.

- ▶ **Variables de tipo texto:** se extraen las palabras y se generan los *n-grams* (combinaciones de palabras consecutivas) correspondientes. Por cada *n-gram* se genera un índice y su *embedding*. Se combinan los *embeddings* a través de la media para generar el *embedding* del texto completo.
- ▶ **Variables de tipo numérico:** se descartan los valores no numéricos, se convierte el resto a tipo decimal y se normaliza mediante *z-score* tanto el valor original como el logaritmo de dicho valor.
- ▶ **Variables temporales:** se extrae año, mes, día y día de la semana como variables categóricas y se aplican las transformaciones de variable numérica a la variable original.

Los resultados del modelo, incluyendo los indicadores de rendimiento sobre el conjunto de datos de pruebas, se muestran en la sección «Evaluation details», de «Model Registry». El valor de *confidence threshold*, que se muestra en la parte superior, indica cómo de fiables son las predicciones. También se proporcionan detalles como la matriz de confusión y la importancia de cada variable en el modelo.

Column name	Data type	Nullability	Missing% (Count)	Invalid values	Distinct values
Age	Numeric	Nullable	0% (0)	0% (0)	77
Balance	Numeric	Nullable	0% (0)	0% (0)	7,168
Campaign	Numeric	Nullable	0% (0)	0% (0)	48
Contact	Categorical	Nullable	0% (0)	0% (0)	3

Figura 7. Interfaz de usuario de Google AutoML, sección de entrenamiento de modelos. Fuente: Google Cloud, s. f.

## Despliegue de modelos en Vertex AI

Una vez que disponemos del modelo, puede utilizarse para hacer **predicciones batch** u **online**. En el primer caso, el modelo puede consumirse directamente desde la sección «Model Registry» que hemos visto anteriormente. Para el segundo caso, es necesario desplegar el modelo en un *endpoint* que se utiliza para recibir peticiones y responder con las predicciones generadas por el modelo.

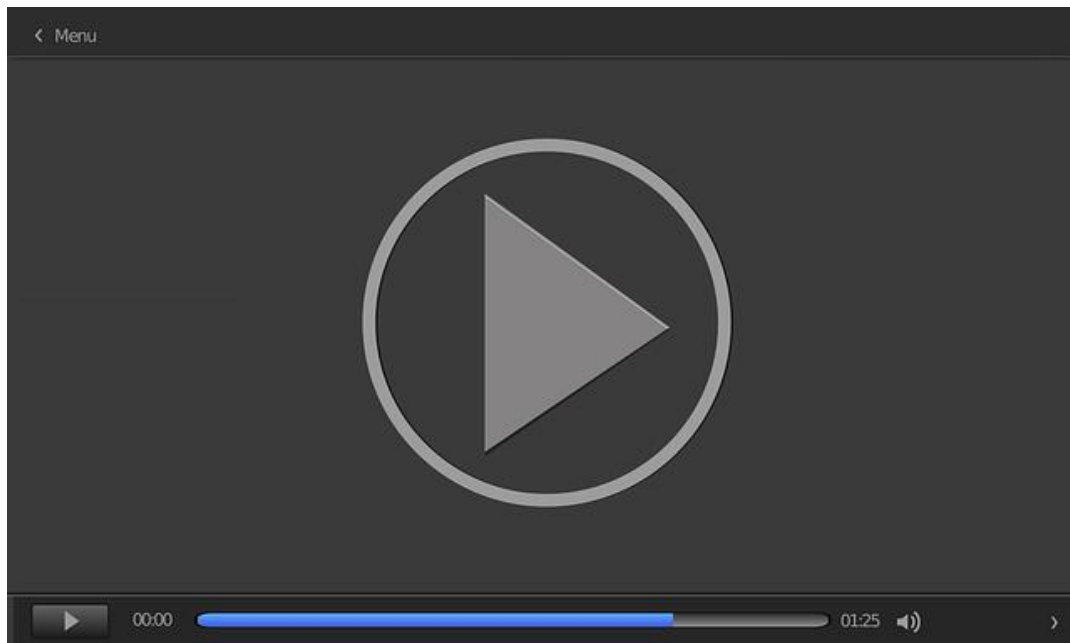
Para desplegar un modelo, se deben ejecutar los siguientes pasos:

- ▶ En la sección «Model Registry», se selecciona el modelo y la versión que se desea desplegar, se hace clic en «Deploy & test», y después en «Deploy to endpoint».
- ▶ Se selecciona un nombre para el *endpoint*.
- ▶ Si se despliegan varias versiones de un modelo, es posible indicar el porcentaje de reparto de cada una en el *endpoint* a través de la opción «Traffic split».
- ▶ Se especifica el número mínimo de nodos de computación disponibles para atender peticiones. Estos nodos generan consumo, aunque no se utilicen.
- ▶ También se especifica el número máximo de nodos que se pueden provisionar en el proceso de autoescalado que realiza Vertex AI para satisfacer una demanda creciente o excesiva de peticiones.
- ▶ Por último, es posible indicar en qué tipo de VM correrá el *endpoint* y si se desea incluir datos de importancia de cada variable en las predicciones generadas.
- ▶ Con todas las opciones configuradas, se pulsa en el botón «Deploy».

Una vez desplegado el modelo, se muestra el *endpoint* en la sección «Deploy & test». Es posible realizar pruebas sobre el servicio de predicciones, accediendo a «Test your model». Para ello, se especifican los datos de entrada y se pulsa en «Predict».

Los siguientes vídeos muestran diferentes escenarios de uso de AutoML en Vertex AI:

*Creación de un modelo de clasificación en Google Cloud con Vertex AI*



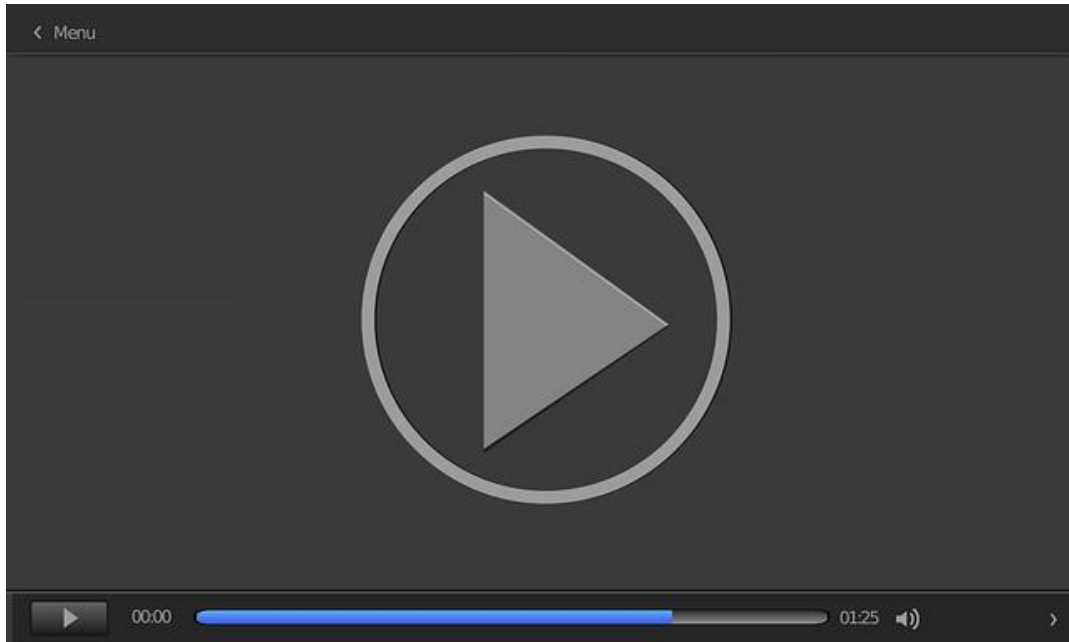
---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=5b488548-ed2c-4f7f-9455-b14800ff678e>

---

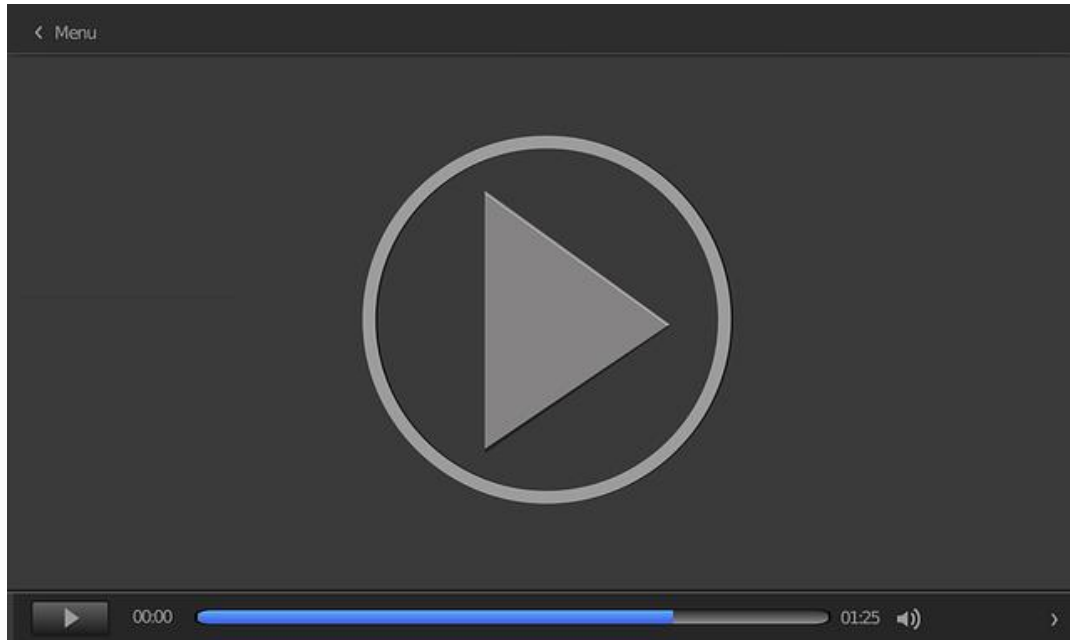
### *Creación de un modelo de regresión en Google Cloud con Vertex AI*



Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=94f09676-977c-4b9e-b520-b14600c11779>

*Creación de un modelo de forecasting de series temporales en Google Cloud con Vertex AI*



---

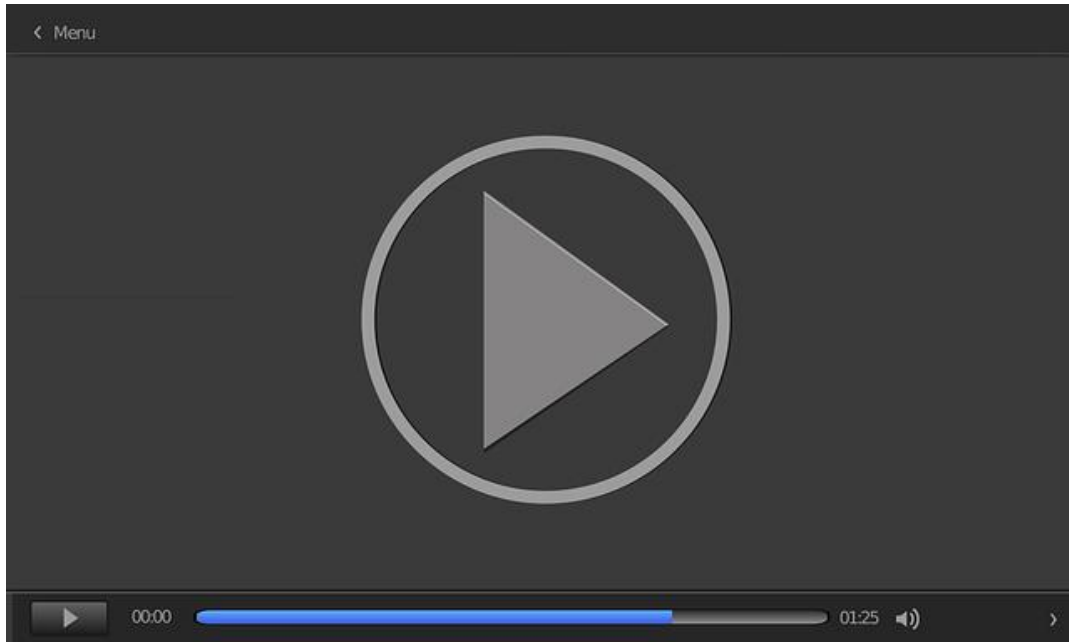
Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=74779eb7-c786-42ad-8118-b1480103f69a>

---



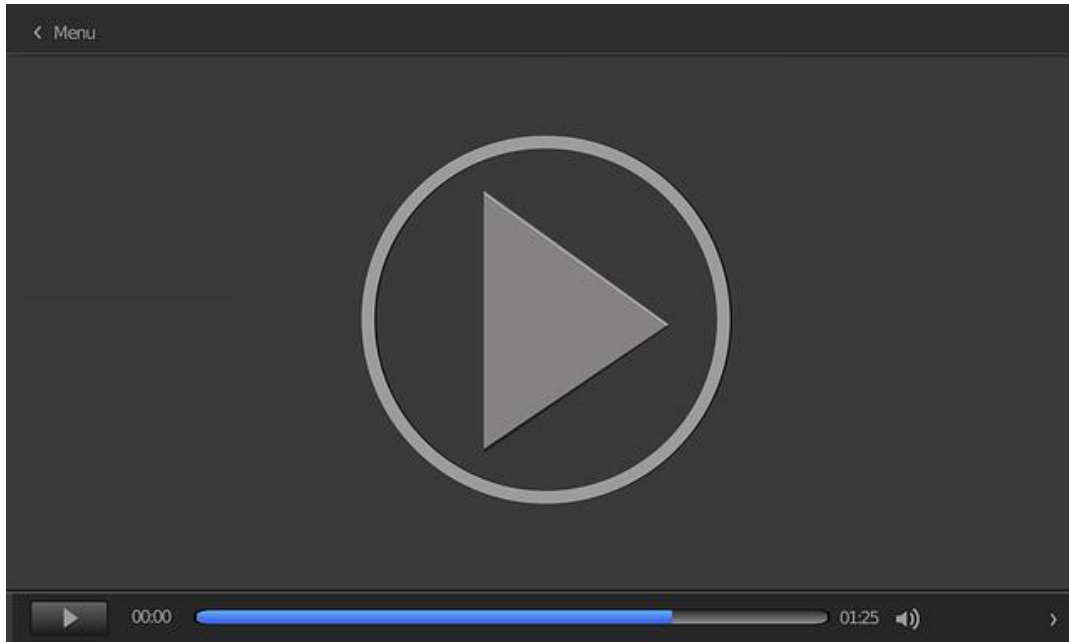
### *Creación de un modelo de clasificación de textos en Google Cloud con Vertex AI*



Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=2e2997c5-ae8d-4ce0-a0a0-b147009bf5ed>

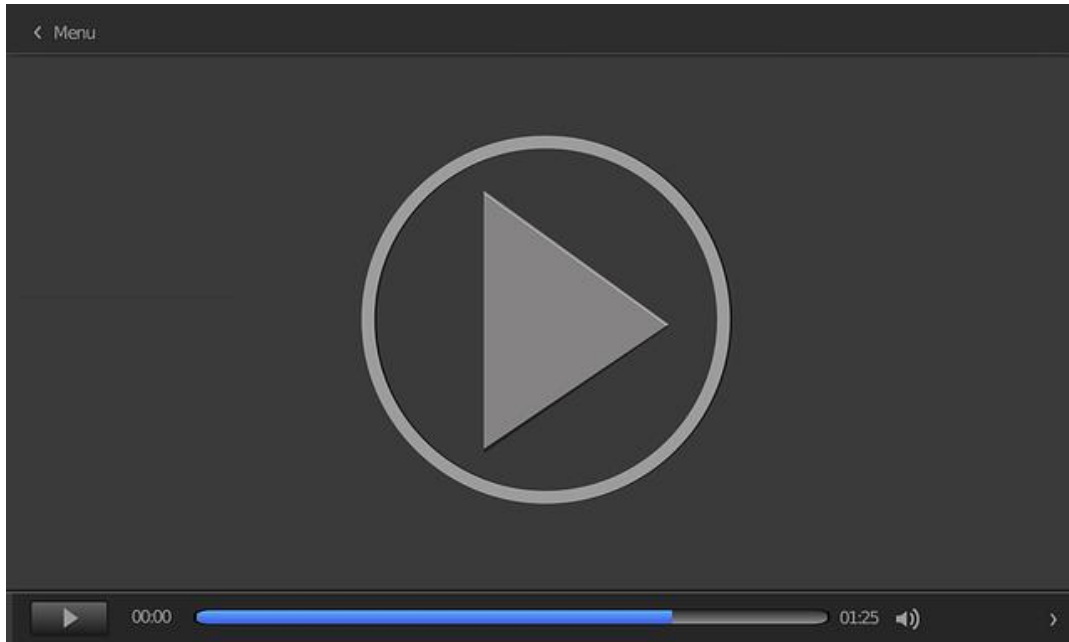
### *Creación de un modelo de clasificación de imágenes en Google Cloud con Vertex AI*



Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=d671bd8e-9837-4acc-8c38-b1480107b3b4>

### *Creación de un modelo de clasificación de vídeo en Google Cloud con Vertex AI*



Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=237ef8e1-25c3-4a76-8767-b14700b9e81f>

## 2.5. Creación de modelos con BigQuery Machine Learning

### Características de BigQuery ML

BigQuery ML es un servicio que permite **construir modelos mediante consultas SQL**. Entre sus principales ventajas se encuentra la posibilidad de trabajar con **volúmenes de datos elevados** sin necesidad de cargarlos en memoria. También se encarga de elegir la técnica de optimización o ajustar el *learning rate* para obtener los mejores resultados. Por último, permite versionar modelos para poder mantener constancia de los cambios realizados y poder comparar los indicadores de rendimiento.

Ofrece **soporte** para los siguientes tipos de modelos:

- ▶ Regresión lineal, regresión logística, ARIMA.
- ▶ *Boosted trees, random forest, xgboost.*
- ▶ *K-means*, factorización de matrices, PCA, *autoencoders*.
- ▶ Redes neuronales profundas, TensorFlow y ONNX.

### Acceso a BigQuery

El primer paso para utilizar BigQuery ML es crear un **entorno** de BigQuery. Para ello, deben ejecutarse los siguientes pasos:

- ▶ Crear un proyecto en Google Cloud.
- ▶ Activar la API de BigQuery.
- ▶ Acceder a la consola de BigQuery.
- ▶ Crear un *dataset* de BigQuery en el que se almacenarán los datos.

- ▶ Cargar los datos desde un CSV o JSON, desde la API o desde Google Cloud Storage.

Una vez que los datos están disponibles, se pueden lanzar **consultas SQL** para analizar los datos y empezar a extraer conclusiones útiles para el paso posterior de creación de modelos.

### Transformación de variables con BigQuery

El proceso de transformación de variables puede hacerse de manera **automática o manual**. En el primer caso, es importante ser conscientes de las limitaciones del servicio: existe un número limitado de transformaciones disponibles y la ausencia de mecanismos de selección de variables o creación de variables nuevas. Cualquier escenario de **transformaciones complejas** demanda llevar a cabo esta labor de **forma manual**.

La **transformación manual** de variables se especifica en la cláusula `TRANSFORM`, dentro del *statement* `SQL CREATE MODEL` (similar a `CREATE TABLE`, en el caso de creación de tablas en una base de datos). Estas transformaciones se aplican tanto para la fase de entrenamiento como para la generación de las predicciones, lo que significa que se aplican de forma automática a los datos de entrada una vez que el modelo ha sido creado.

Entre las **funciones de transformación** disponibles, se incluyen:

- ▶ **Binning o generación de buckets**, especificando vía SQL los *split points* de cada categoría. También ofrece la opción de *splits* basados en cuantiles o la generación de categorías de valores para variables categóricas.
- ▶ **Generación de todas las combinaciones posibles** a partir de los valores de un conjunto de variables categóricas.
- ▶ **N-grams** para fragmentos de texto.

- ▶ **Normalización y estandarización** de variables numéricas.
- ▶ **Imputación de valores ausentes** (*missing data*).
- ▶ **Codificación** basada en etiquetas o en *one-hot encoding*.

Una vez aplicadas las transformaciones pertinentes, el siguiente paso es la **ejecución del proceso de entrenamiento** para la creación del modelo.

### Creación de modelos con BigQuery ML

Además de los tipos de modelos disponibles en BigQuery ML, indicados al comienzo de la sección, también es posible importar modelos creados con TensorFlow, ONNX o XGBoost, o bien utilizar un *endpoint* de Vertex AI para generar predicciones a partir de él con los datos almacenados en BigQuery. El *statement* SQL utilizado para crear modelos es `CREATE MODEL`.

Como ejemplo, a continuación, se indican las principales **opciones de configuración** para **modelos de regresión**:

- ▶ Tipo de modelo.
- ▶ Columna con la variable que se desea predecir.
- ▶ Técnica de optimización utilizada durante el proceso de entrenamiento.
- ▶ Parámetros de regularización L1 y L2.
- ▶ Número máximo de iteraciones.
- ▶ Parámetros relacionados con el *learning rate* y criterios de *early stopping*.
- ▶ Técnicas de división de los datos en conjuntos de entrenamiento y validación.
- ▶ Asignación de pesos a clases para rebalanceo del *dataset*.
- ▶ Método de codificación de variables categóricas.

Existen **opciones equivalentes** para la creación de modelos basados en aprendizaje profundo (*deep learning*), *random forest*, *boosted trees* o *k-means*.

Para importar **modelos de otros frameworks**, se utiliza también `CREATE MODEL` con las cláusulas `MODEL_TYPE` y `MODEL_PATH` para indicar el *framework* y la ruta al modelo, respectivamente.

### Ajuste de hiperparámetros con BigQuery ML

La configuración del proceso de ajuste de hiperparámetros en BigQuery ML se realiza también a través del *statement* `SQL CREATE MODEL`.

Las **opciones de configuración** más relevantes son las siguientes:

- ▶ Número máximo de modelos con hiperparámetros diferentes que se pueden entrenar.
- ▶ Número máximo de ejecuciones paralelas de procesos de entrenamiento.
- ▶ Algoritmo de ajuste de hiperparámetros: ofrece las opciones *random search*, *grid search* o Vizier (algoritmo propio de Vertex AI).
- ▶ Especificación de hiperparámetros y los rangos o valores sobre los cuales se quiere realizar el ajuste.
- ▶ Métricas objetivo para determinar con qué hiperparámetros se obtienen mejores resultados. Se proporcionan métricas apropiadas en función del tipo de modelo que se va a entrenar. Por ejemplo, para regresión logística se ofrecen los siguientes: `precision`, `recall`, `accuracy`, `f1_score`, `log_loss`, `roc_auc`.

## Evaluación de modelos e inferencia

Para evaluar el **rendimiento de un modelo entrenado**, se utiliza la función `ML.EVALUATE()`. Esta función recibe como parámetros el modelo que se desea evaluar y la tabla que almacena los datos de validación, o la *query* que permite generarlos. Devuelve como resultado el valor de las métricas correspondientes al tipo de modelo. Por ejemplo, para un **modelo de tipo PCA**, genera el ratio de varianza total explicada (`total_explained_variance_ratio`).

Para realizar inferencias con un modelo, se utiliza la función `ML.PREDICT()`. Esta función recibe como parámetros el modelo a partir del cual se desean generar las inferencias y la tabla que almacena las entradas al modelo, o la *query* que permite generarlas. Devuelve como resultado una tabla con las entradas y columnas adicionales que albergan los valores correspondientes de las predicciones. Estos valores dependen del **tipo de modelo**. Por ejemplo, para *k-means*, retorna el identificador del *centroid* (*cluster* al que pertenece la observación) y la distancia a los *centroids* más cercanos.



## 2.6. Interpretación de modelos

### Introducción a la explicabilidad

La explicabilidad de los modelos, es decir, la **capacidad de explicar el porqué de una predicción**, es una característica deseable para generar confianza entre los usuarios de este. Es importante diferenciar entre la **explicabilidad global** y **local** de un modelo. La primera se refiere a la importancia o influencia de una variable del modelo en los valores de todas las predicciones, mientras que la segunda se refiere a dicha influencia en el contexto de una observación específica (cuánto ha influido la variable en el resultado obtenido para esos datos de entrada).

### Técnicas de explicabilidad

Se utilizan **diferentes técnicas** de explicabilidad en función del **tipo de datos**:

Categoría	Nombre	Descripción
Para imágenes	<b><i>Integrated gradients</i></b>	Calcula el gradiente del valor de salida respecto a los píxeles de la imagen de entrada para determinar qué píxeles realizan una mayor contribución al resultado de la clasificación.
	<b><i>eXtended relevance-weighted attribution of importance (XRAI)</i></b>	Variante del anterior que emplea técnicas de segmentación para mostrar las regiones, en lugar de los píxeles, que más influyen en el resultado de la clasificación.
	<b><i>Local interpretable model-agnostic explanations (LIME)</i></b>	Genera perturbaciones sobre la imagen y obtiene las predicciones correspondientes. A continuación, ajusta un modelo sencillo (por ejemplo, regresión lineal) y utiliza los resultados para explicar la clasificación de la imagen original.
	<b><i>Gradient-weighted class activation mapping (Grad-CAM)</i></b>	Calcula el gradiente del valor de salida respecto a la salida de la última capa convolucional para generar un <i>heatmap</i> con las regiones clave en el proceso de clasificación.

Categoría	Nombre	Descripción
Para datos tabulares	<i>Local interpretable model-agnostic explanations (LIME)</i>	Como en el caso de las imágenes, LIME genera observaciones sintéticas alrededor de la instancia que se quiere explicar y ajusta un modelo sencillo e interpretable a partir de dichas observaciones adicionales. A continuación, utiliza los criterios de interpretabilidad del modelo para explicar la predicción de la observación original.
	<i>Shapley additive explanations (SHAP)</i>	Calcula el promedio de la contribución marginal de cada variable para todas las combinaciones posibles de variables de entrada. De esta forma, obtiene la influencia de cada una de ellas en las predicciones.
	<i>Permutation feature importance</i>	Mide el deterioro en el rendimiento de un modelo cuando los valores de una variable se reemplazan aleatoriamente. Este proceso se repite para cada variable. Si la variable tiene gran importancia, el deterioro del rendimiento es mayor.
	<i>Partial dependence plots (PDP)</i>	Evalúa la predicción de cada variable cuando se mantienen el resto con valores constantes. De esa forma, puede determinar en qué medida influye esa variable en las predicciones del modelo.
	<i>GINI importance</i>	Se utiliza en árboles de decisión y determina la reducción de impureza (incertidumbre) que proporciona cada variable en un <i>split</i> . Cuanto mayor es la reducción, mayor es la influencia de esa variable en la predicción.
	<i>Coefficients in linear models</i>	Los mayores coeficientes en un modelo de regresión lineal están asociados a mayor importancia de la variable correspondiente.

Categoría	Nombre	Descripción
	<i>Coefficients in linear models</i>	Los mayores coeficientes en un modelo de regresión lineal están asociados a mayor importancia de la variable correspondiente.
Para texto	<i>Text-specific LIME, text-specific SHAP</i>	Variantes de LIME y SHAP que permite identificar las palabras o frases más importantes de un texto.
	Mecanismos de atención	Se utilizan en modelos de tipo <i>transformers en deep learning</i> y generan unos pesos de atención para determinar qué palabras contribuyen más a las predicciones, es decir, a los textos que genera el modelo.

Tabla 1. Resumen de técnicas de explicabilidad. Fuente: elaboración propia.

## Funcionalidades de explicabilidad en Vertex AI

Google Vertex AI implementa las siguientes técnicas de explicabilidad. Todas ellas se encuentran entre las descritas en la sección previa.

- ▶ **SHAP:** se utiliza con modelos tabulares de AutoML y cualquier modelo que esté corriendo en un contenedor de predicciones, incluyendo aquellos generados de forma manual en Jupyter. Se aplica para tareas de clasificación y regresión.
- ▶ **Integrated gradients:** se utiliza con modelos de imagen de AutoML y modelos de TensorFlow personalizados, ejecutados en alguno de los contenedores de predicción disponibles por defecto. Se aplica a tareas de clasificación y regresión o a clasificación de imágenes.
- ▶ **XRAI:** se utiliza con el mismo tipo de modelos que *integrated gradients*, pero solo se aplica la clasificación de imágenes.

Google Vertex AI también proporciona mecanismos de explicabilidad **basada en ejemplos**. Para ello, genera ejemplos similares al *input* sobre el que se solicita una predicción a partir de los datos de entrenamiento.

Si las predicciones asociadas a estos ejemplos coinciden con la del *input*, las predicciones son **sólidas** y el modelo funciona correctamente. Si, en cambio, las predicciones difieren, significa que el modelo **no es estable** o bien que hay datos incorrectamente etiquetados en el conjunto de datos de entrenamiento.

**Deploy to endpoint**

- ✓ Define your endpoint
- 2 Model settings
- 3 Model monitoring

**DEPLOY** **CANCEL**

Accelerator type  
Select an accelerator type

Service account

A service account determines what Google Cloud resources your service code can access. By default, a Google-managed service account is used with permissions appropriate for most models. You can also use a user-managed service account to customize permissions. [Learn more.](#)

**Logging**

Logging settings are permanent for this endpoint, and 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

**Explainability options**

☒ Enable feature attributions for this model

Sampled Shapley	5 samples
Input features	2(f1, f2)
Output feature	predictions

**EDIT**

It may take several minutes for endpoint settings to take effect.

**DONE**

ADD AN ITEM

**CONTINUE**

Figura 8. Interfaz de despliegue de modelos en Google BigQuery ML con opciones de explicabilidad.

Fuente: Google Cloud, s. f.

## 2.7. Optimización de modelos mediante ajuste de hiperparámetros

### Introducción

Los **hiperparámetros** de un modelo son aquellos fijados de forma previa al proceso de entrenamiento; mientras que los **parámetros** son aquellos que se ajustan como resultado de dicho proceso.

Los hiperparámetros representan **configuraciones posibles** del modelo. Cada configuración o conjunto de valores de hiperparámetros conduce a un nivel de rendimiento diferente del modelo resultante. Por ello, se lleva a cabo un **proceso de ajuste** de estos hiperparámetros para determinar qué configuración del modelo proporciona el máximo rendimiento.

Hay **tres técnicas principales** para llevar a cabo este proceso:

- ▶ **Grid search:** consiste en probar todas las combinaciones posibles de valores de hiperparámetros para identificar la mejor. En función del número de combinaciones, esta técnica puede ser computacionalmente inviable.
- ▶ **Random search:** consiste en seleccionar aleatoriamente un subconjunto de combinaciones posibles. El tamaño de este subconjunto depende de la capacidad de computación disponible. Por tanto, evita el problema asociado a *grid search*, pero presenta el inconveniente de que quizá no encuentre la combinación óptima, puesto que no está probando todas las combinaciones posibles.
- ▶ **Bayesian optimization:** después de probar una combinación, el sistema aprende la dirección que debe seguir —qué combinaciones debe probar a continuación— para ir mejorando los resultados progresivamente. Es, por tanto, una variante inteligente del *random search*.

## Ajuste de hiperparámetros en Vertex AI

Google Vertex AI dispone de implementaciones para realizar el proceso de ajuste de hiperparámetros en torno a un modelo. Estas implementaciones se encuentran en la librería de Python `google.cloud.aiplatform import hyperparameter_tuning`.

Para ejecutar el proceso, se debe crear un **contenedor**, que alberga el código asociado al proceso de entrenamiento, y un **job**, que lanza múltiples ejecuciones de este contenedor. Para poder comparar los resultados de cada ejecución, se debe especificar la **métrica de rendimiento** que medirá dichos resultados. Esta métrica depende del tipo de modelo que se está ajustando.

Vertex AI soporta las técnicas de *grid search* y *random search* descritas en la sección previa, pero además incluye otras dos (*gaussian process bandits*, *linear combination search*) que utilizan diferentes mecanismos para reducir el espacio de combinaciones de una manera inteligente, como en el caso de *bayesian optimization*.

## Neural architecture search

*Neural architecture search* (NAS) es una técnica de **diseño automático de redes neuronales**, que permite encontrar la arquitectura que obtiene un **mejor rendimiento** para la tarea dada. Mientras el ajuste de hiperparámetros considera fija la arquitectura del modelo (en este caso, la red neuronal), NAS parte de cero y se focaliza en un proceso de ajuste orientado al **propio diseño** de la arquitectura. Por tanto, se genera un espacio de arquitecturas posibles y se utiliza un proceso de optimización similar al del ajuste de hiperparámetros para probar todas las combinaciones y determinar cuál proporciona **mejores resultados**.

Google Vertex AI proporciona un **lenguaje específico** para NAS, que permite implementar el proceso de búsqueda de la arquitectura óptima. También proporciona recursos por defecto para acotar el espacio de soluciones, *hardware* especializado para acelerar el proceso (GPUs) y la posibilidad de definir métricas personalizadas para evaluar el rendimiento de cada arquitectura.



## 2.8. Cuaderno de ejercicios

1. Especificar la línea de comandos que permite subir una lista de ficheros contenida en un fichero a un *bucket* de Google Cloud Storage con la herramienta `gsutil`. Especificar también la línea de comandos que permite descargar los ficheros subidos a Google Cloud Storage a una carpeta local.

```
cat lista_de_ficheros | gsutil -m cp -I gs://bucket-destino
```

```
cat lista_de_urls | gsutil -m cp -I ./carpeta-destino
```

2. Indicar la sintaxis de BigQuery ML para entrenar un modelo de regresión lineal con el *dataset* público `bigquery-public-data.samples.natality`, que predice el peso de nacimiento de un bebé (`weight_pounds`) a partir de las siguientes variables: `mother_age`, `mother_married`, `gestation_weeks`, `weight_gain_pounds`, `is_male`. Se deben descartar los valores nulos de `weight_pounds`.

```
CREATE OR REPLACE MODEL `mi_modelo`
```

```
OPTIONS
```

```
(model_type='linear_reg',  
  
input_label_cols=['weight_pounds']) AS
```

```
SELECT
```

```
mother_age,  
  
mother_married,  
  
gestation_weeks,  
  
weight_gain_pounds,
```

```
is_male,  
  
weight_pounds  
  
FROM  
  
`bigquery-public-data.samples.natality`  
  
WHERE  
  
weight_pounds IS NOT NULL
```

**3.** ¿En qué consisten las métricas de `precision` y `recall` que proporciona Google AutoML en una tarea de clasificación? Utilizar como ejemplo un modelo que predice si un cliente va a comprar o no un producto.

La `precision` es el porcentaje de predicciones de compra que fueron correctas. El porcentaje restante serían errores porque el cliente no compró el producto. El `recall` es el porcentaje del total de compras que fue capaz de detectar el modelo. El porcentaje restante serían aquellas compras que el modelo no fue capaz de detectar.

**4.** Especificar la sintaxis del parámetro `parameter_spec` para indicar los hiperparámetros de una red neuronal que deben ajustarse en un `job` de Vertex AI con las siguientes características:

- ▶ `lr` (*learning rate*): entre 0.001 y 0.1, con escala logarítmica.
- ▶ `units` : entre 4 y 128, con escala lineal.
- ▶ `activation` : `relu` o `selu`.
- ▶ `batch_size` : entre 128 y 256, con escala lineal.

```
parameter_spec={  
  
    'lr': hpt.DoubleParameterSpec(min=0.001, max=0.1, scale='log'),
```

```
        'units': hpt.IntegerParameterSpec(min=4, max=128,
scale='linear'),

        'activation': hpt.CategoricalParameterSpec(values=['relu',
'selu']),

        'batch_size': hpt.DiscreteParameterSpec(values=[128, 256],
scale='linear')

    },
```

**5.** Escribir la sentencia SQL con la que podemos generar la matriz de confusión de un modelo de clasificación `model_name`, con los datos almacenados en la tabla `table_name` de `dataset_name`, con un umbral de probabilidad de 0.6.

```
SELECT * FROM ML.CONFUSION_MATRIX(MODEL `dataset_name.model_name`, (SELECT
* FROM `dataset_name.table_name`), STRUCT(0.6 AS threshold))
```

## 2.9. Referencias bibliográficas

Ciaburro, G., Ayyadevara, V. K. y Perrier, A. (s. f.). *Hands-On Machine Learning on Google Cloud Platform*. O'Reilly Online Learning.

<https://www.oreilly.com/library/view/hands-on-machine-learning/9781788393485/dc02f060-3ed9-4896-91e0-79b8a90a2999.xhtml>

Delaney, J. (2019, marzo 13). Bigquery ml quickstart. Fireship.

<https://fireship.io/lessons/bigquery-ml-quickstart/>

Google Cloud (s. f.). *Create a Vertex AI Workbench instance using a custom container*. Google.

<https://cloud.google.com/vertex-ai/docs/workbench/instances/create-custom-container>

Google Cloud (s. f.). *Explainable AI for BigQuery ML models*. Google.

<https://cloud.google.com/bigquery/docs/vertex-xai>

Google Cloud (s. f.). Guía para principiantes de AutoML. Google.

<https://cloud.google.com/vertex-ai/docs/beginner/beginners-guide?hl=es-419>

Keyan, K. (2022, enero 8). Vertex AI Workbench - the Cloudside View. *Medium*.

<https://blog.thecloudside.com/exploring-vertex-ai-workbench-google-cloud-338de45270c1>

Namjoshi, N. (2021, noviembre 11). Schedule and execute notebooks with Vertex AI

Workbench. *Google Cloud Blog*. <https://cloud.google.com/blog/products/ai-machine-learning/schedule-and-execute-notebooks-with-vertex-ai-workbench>

### The definitive guide to Google Vertex AI: accelerate your machine learning journey with Google Cloud Vertex AI and MLOps best practices

Bhatia, J. y Chaudhary, K. (2023). *The definitive guide to Google Vertex AI: accelerate your machine learning journey with Google Cloud Vertex AI and MLOps best practices*.

Este libro realiza un recorrido exhaustivo por las funcionalidades que ofrece la plataforma Vertex AI de Google, sintetizando los conceptos clave y proporcionando ejemplos prácticos que facilitan el proceso de aprendizaje y la adquisición de experiencia práctica.

### Machine learning with BigQuery ML: create, execute, and improve machine learning models in BigQuery using standard SQL queries

Marrandino, A. (2021). *Machine learning with BigQuery ML: create, execute, and improve machine learning models in BigQuery using standard SQL queries*.

BigQuery ML es una de las principales innovaciones de Google en materia de *machine learning*. Este libro presenta la plataforma y describe en detalle las capacidades que proporciona a usuarios que conocen el lenguaje SQL y quieren utilizarlo para desarrollar de forma muy ágil modelos adecuados para resolver diferentes problemáticas de negocio.

1. ¿Qué métodos se pueden utilizar para enviar ficheros o carpetas a un *bucket* de Google Cloud Storage?
  - A. Se debe utilizar el cliente pesado facilitado por Google, que puede descargarse en el sitio web de Google Cloud.
  - B. Google Cloud Console, línea de comandos, REST API.
  - C. Solamente Google Cloud Console.
  - D. Únicamente REST API, como medida de seguridad para evitar fugas de datos sensibles.
  
2. Google Cloud Storage es un servicio de almacenamiento basado en:
  - A. Ficheros.
  - B. Objetos.
  - C. Bloques.
  - D. Columnas.
  
3. ¿Cuál de las siguientes es una ventaja de los cuadernos gestionados por el usuario?
  - A. No requieren reinicio tras un cambio de la VM en la que se ejecutan.
  - B. Admiten un mayor grado de personalización que los cuadernos gestionados por Google.
  - C. Reducen las opciones de personalización para proporcionar mayor seguridad.
  - D. En general, se ejecutan con menor latencia que los cuadernos gestionados por Google.

4. ¿Qué lenguajes de programación están soportados en Google AutoML?
- A. Java, Python y Rust.
  - B. Ninguno, porque AutoML no requiere escribir código fuente para entrenar modelos.
  - C. Java y Python.
  - D. Solo Python, que es lenguaje más habitual en los Jupyter Notebooks.
5. ¿Cuál de las siguientes no es una métrica de optimización (*optimization objective*) para un modelo de regresión creado con Google AutoML?
- A. RMSE.
  - B. AUC ROC.
  - C. MAE.
  - D. RMSLE.
6. ¿Qué tratamiento se aplica a las variables categóricas en el proceso de transformación de datos de Google AutoML?
- A. Se genera un *embedding* asociado a cada valor de la variable categórica.
  - B. Se genera un índice por cada valor y un *embedding* asociado al índice.
  - C. No se realiza ningún tratamiento, se utilizan los valores tal cual aparecen en el *dataset*.
  - D. Se asigna un valor entero correlativo a cada valor encontrado.
7. ¿Cuál de las siguientes no es una opción de configuración de CREATE MODEL para el proceso de entrenamiento en BigQuery ML?
- A. Número máximo de iteraciones.
  - B. Ratio de varianza total explicada.
  - C. Tipo de modelo.
  - D. Parámetros de regularización L1 y L2.



8. ¿Cuál es la característica diferencial de la técnica *bayesian optimization*?
- A. Que prueba exhaustivamente todas las combinaciones posibles.
  - B. Que es capaz de seleccionar las combinaciones más prometedoras para maximizar el rendimiento.
  - C. Que elige aleatoriamente qué combinaciones probar para hacer que el proceso de optimización sea computacionalmente viable.
  - D. Que elige qué combinaciones probar a partir de reglas que especifica el usuario.
9. ¿Cuál es el espacio de soluciones de *neural architecture search*?
- A. Los posibles valores de hiperparámetros de la red neuronal.
  - B. Los posibles diseños de arquitectura de la red neuronal.
  - C. Los valores de parámetros que se ajustarán en el proceso de entrenamiento de la red neuronal.
  - D. Ninguno de los anteriores.
10. ¿En qué se diferencian los métodos de *integrated gradients* y XRAI para determinar las regiones de una imagen que más contribuyen al resultado de la clasificación?
- A. *Integrated gradients* identifica áreas de la imagen; XRAI identifica píxeles.
  - B. *Integrated gradients* identifica píxeles; XRAI identifica áreas de la imagen.
  - C. XRAI es una variante de *integrated gradients* que se ejecuta más rápido.
  - D. *Integrated gradients* es una versión mejorada de XRAI.