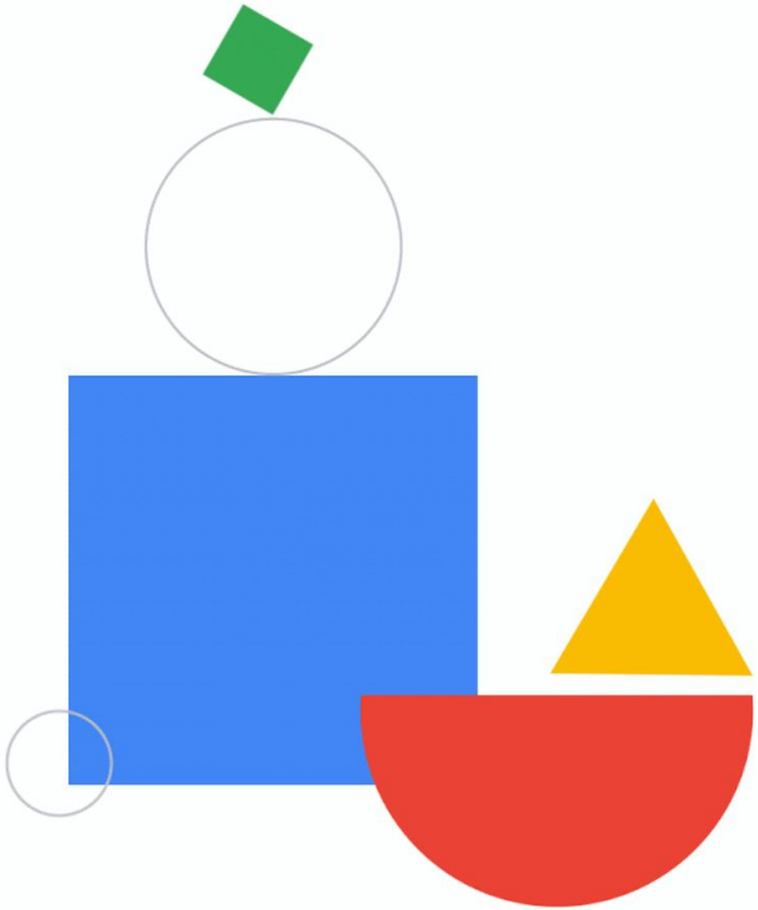
Google Cloud Academy

The Path to Partner Technical Readiness
Professional Machine Learning Engineer Certification

Session 3: Vertex AI & ML Model Metrics on Vertex AI



Instructor: Ben Ahmed

Google Cloud
Authorized Trainer

Working in IT industry for 12 years. A mix of roles, hands on engineering roles in ML (NLP, CNN's, DNN's, VertexAI, Python, TensorFlow, PyTorch, CNN's,) and MLOps solution designing

Years of experience instructing

a Authorized Google Cloud Trainer, I have now been running workshops with Google Customers for over 4 Years for Google. Conducted +25 Workshops, +600 Attendees

Other pertinent tech roles

Machine learning Engineer & Data Engineer for Startups, Oil & Gas (BP) & Banking (HSBC)

Google Cloud Certified PROFESSIONAL CLOUD ARCHITECT

Google Cloud Certified PROFESSIONAL DATA ENGINEER

Google Cloud Certified PROFESSIONAL MACHINE LEARNING ENGINEER

The following course materials are copyright protected materials.

They may not be reproduced or distributed and may only be used by students attending this Google Cloud Partner Learning Services program.

Course etiquette

Please silence your phone and take calls outside.

Recording this class is prohibited.

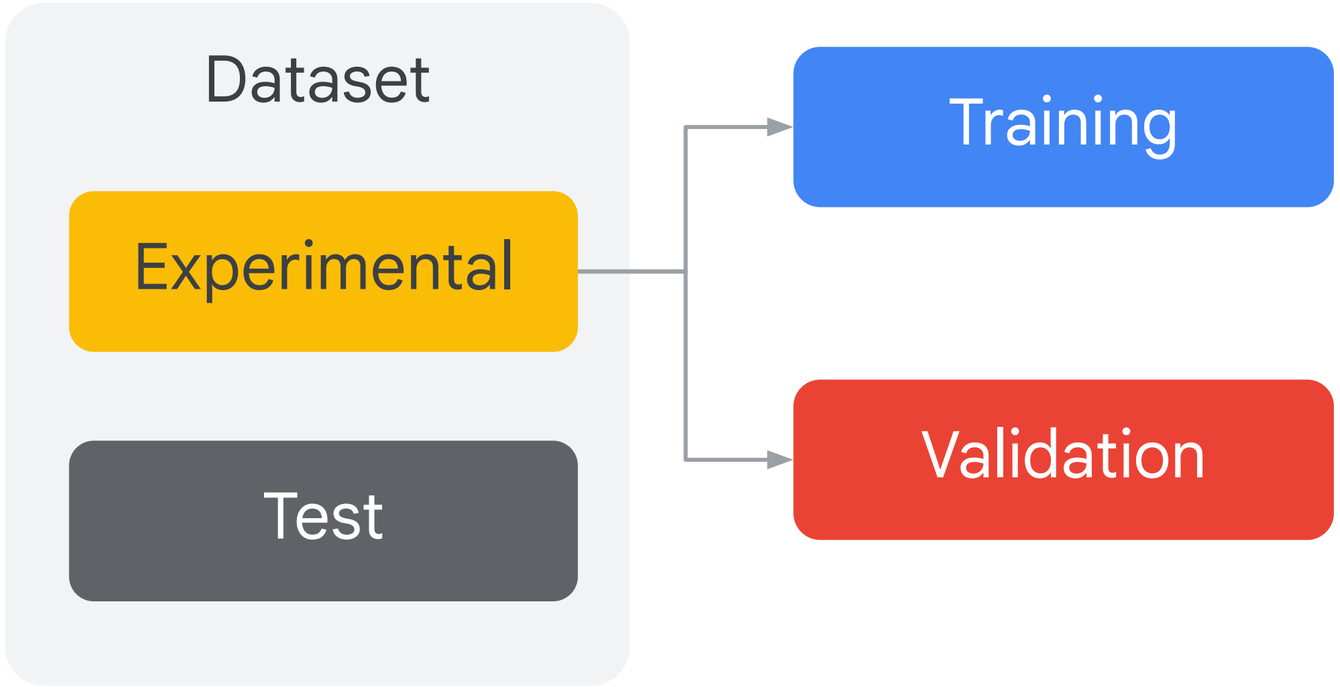
Ask questions via Chat, If time permits

Week's agenda

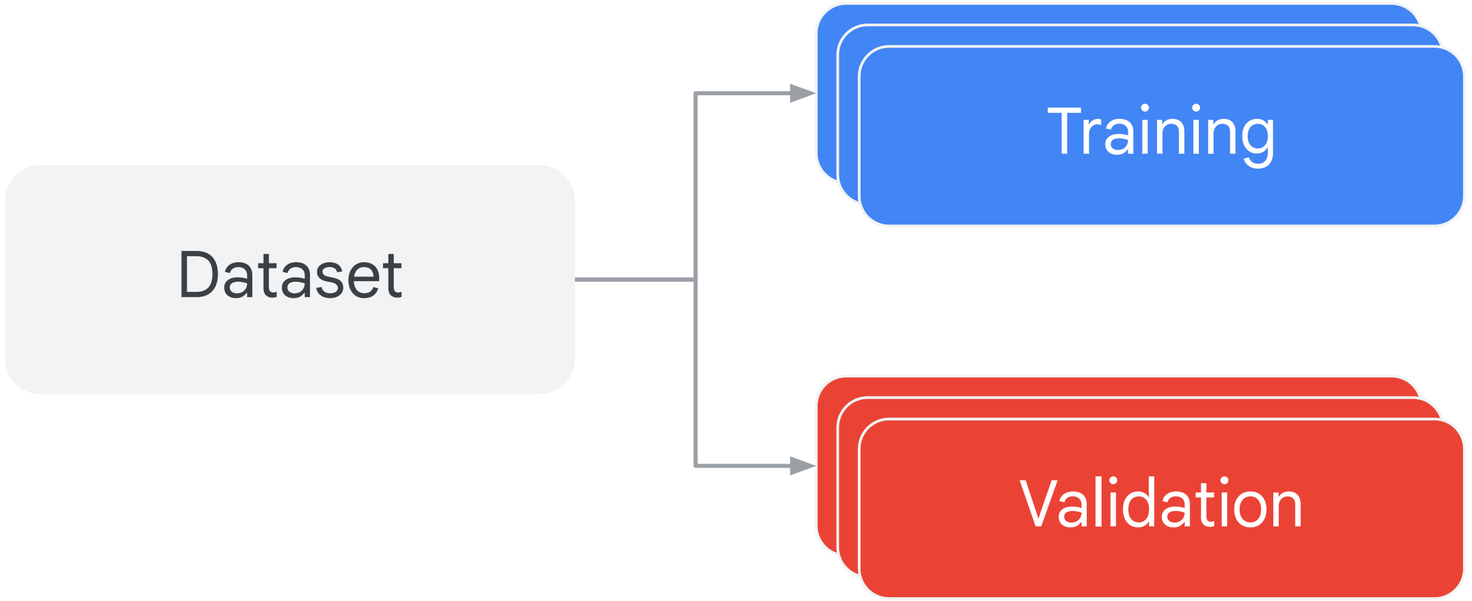
Some Sample Questions

Data Extraction

Evaluate the final model with independent test data

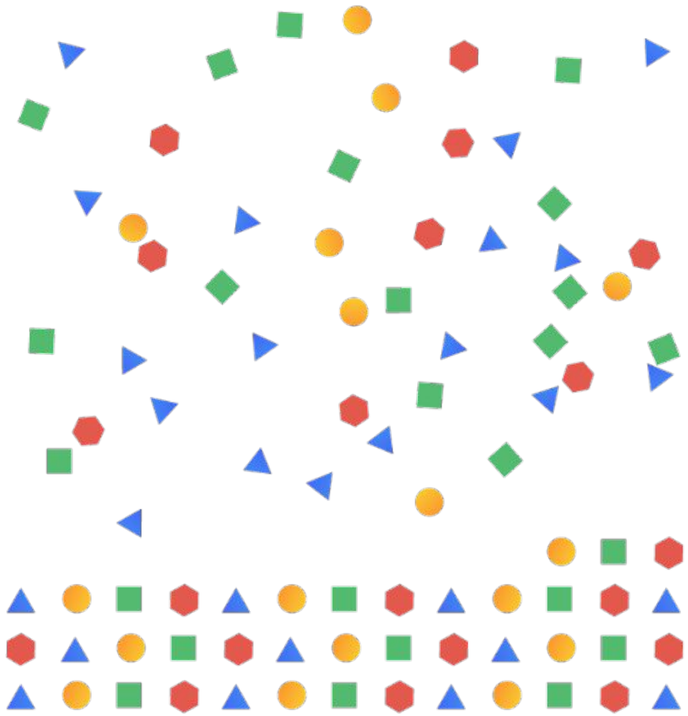


Evaluate the final model with cross-validation



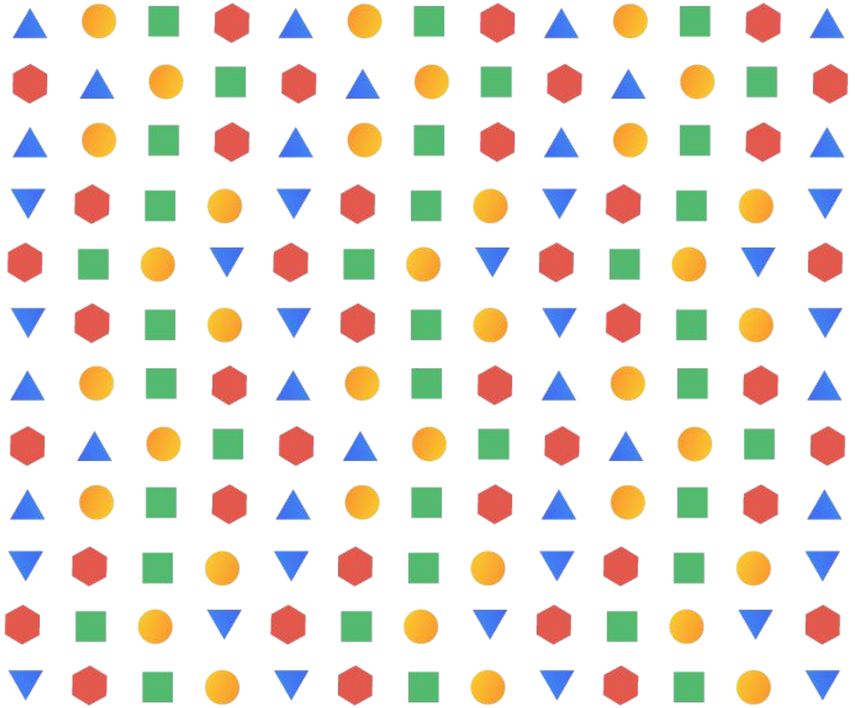
Training set

- The vast majority of your data should be here.
- This is the data your model “sees.”
- The model uses it to learn its parameters.



Validation set

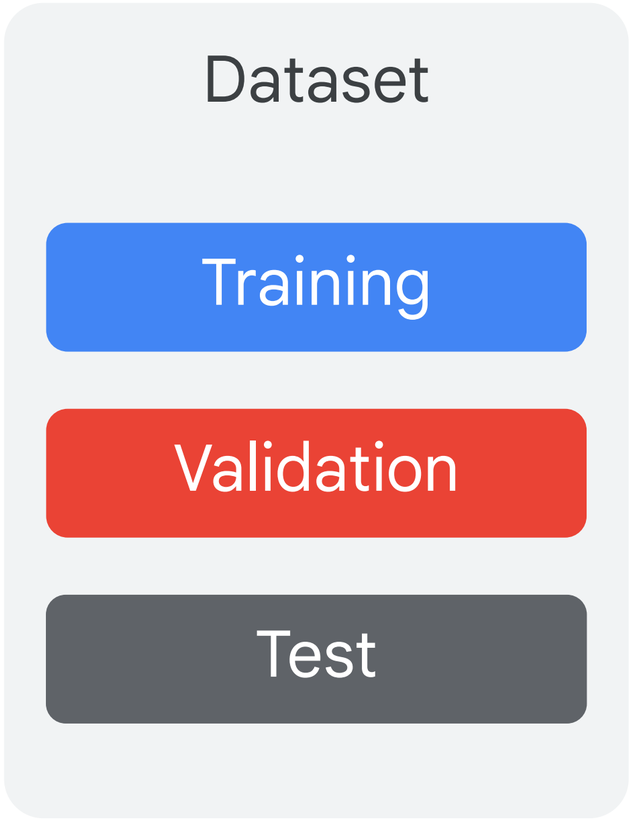
- The validation set is sometimes referred to as the “dev” set, and is also used in the training process.
- It is used to tune the hyperparameters (variables that specify the model’s structure) after the model uses the training data in each iteration of the training process.
- Using the validation set to tune the hyperparameters means the model will generalize better.



Test set

- The test set is used after the training process is complete.
- The test set is a new challenge for the model to get a strong indication of how the model will perform on real-world data.

We often have large datasets in BigQuery that we want to use for machine learning



Row	date	airline	departure_airport	departure_schedule	arrival_airport	arrival_delay
1	2019-08-07	TZ	SRQ	1255	IND	-14.0
2	2019-03-05	TZ	SRQ	2117	IND	-9.0
3	2020-04-12	TZ	SRQ	2000	IND	-17.0
4	2021-04-16	TZ	SRQ	1215	IND	-5.0
5	2021-03-20	TZ	SRQ	645	IND	14.0
6	2020-04-06	TZ	SRQ	1235	IND	-8.0

It’s easy to get a random 80% of your dataset for training

```
#standardSQL
SELECT
  date,
  airline,
  departure_airport,
  departure_schedule,
  arrival_airport,
  arrival_delay
FROM
  `bigquery-samples.airline_ontime_data.flights`
WHERE
  RAND() < 0.8
```

RAND will return a number between 0 and 1.

However, experimentation requires repeatability

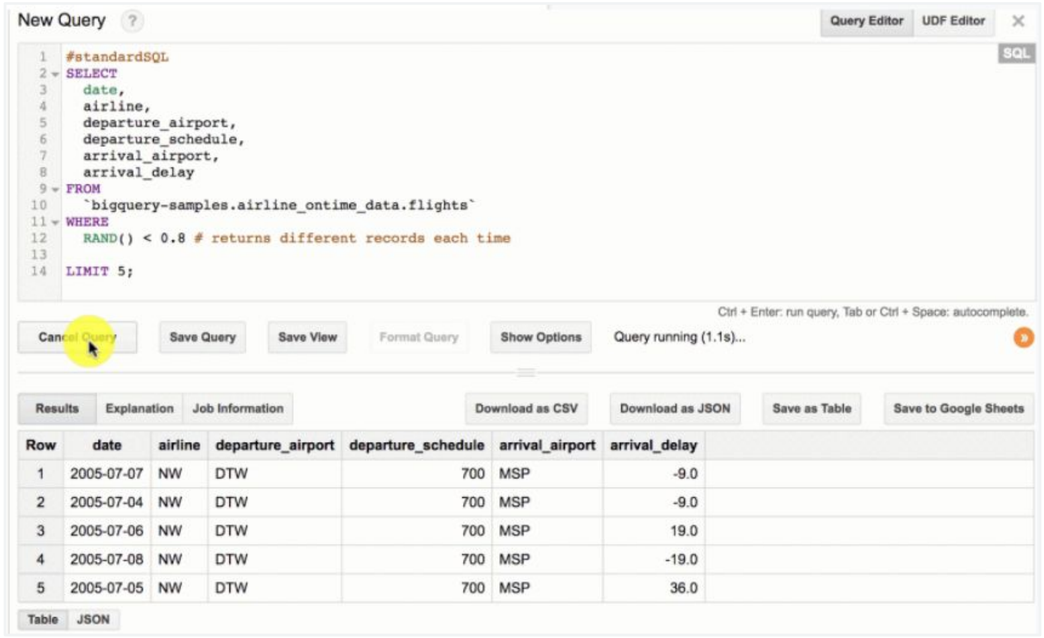
You need to know which specific data was involved in training, validation, and testing.

Naive random splitting is not repeatable

Order of rows in BigQuery is not certain without ORDER BY.

Hard to identify and split the remaining 20% of data for validation and testing.

RAND() will return different results each time



Solution: Split a dataset into training/validation/test using the hashing and modulo operators

```
#standardSQL
SELECT
  date,
  airline,
  departure_airport,
  departure_schedule,
  arrival_airport,
  arrival_delay
FROM
 `bigquery-samples.airline_ontime_data.flights`

WHERE
  MOD(ABS(FARM_FINGERPRINT(date)),10) < 8
```

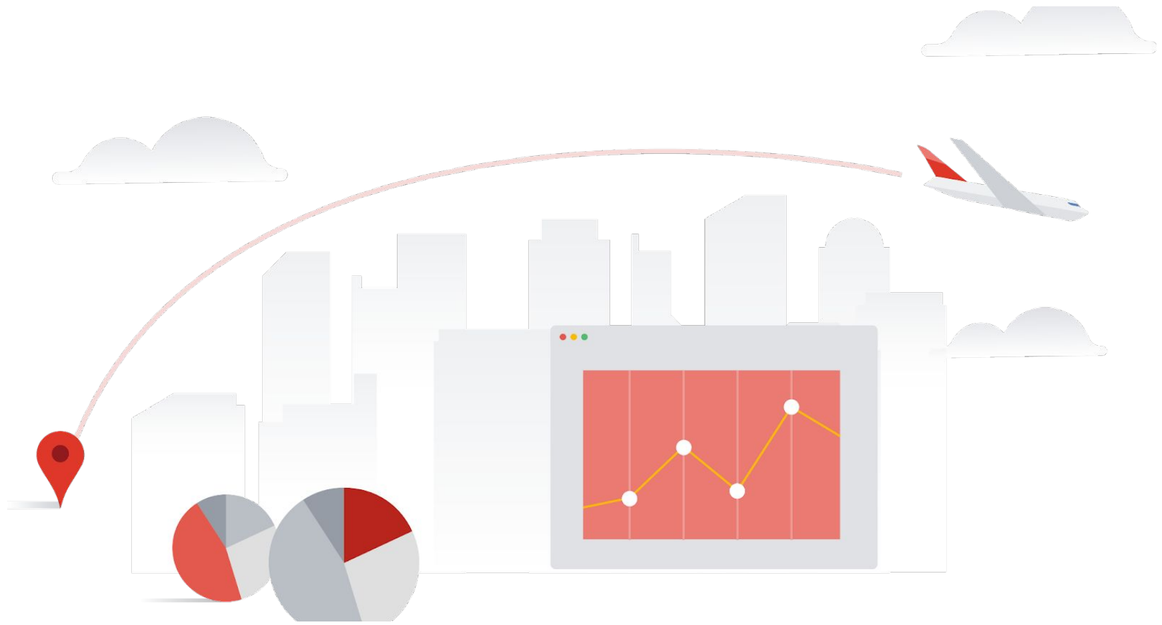
Note:

Even though we select date, our model wouldn't actually use it during training.

Hash value on the date will always return the same value.

Then we can use a modulo operator to only pull 80% of that data based on the last few hash digits.

Carefully choose which field will split your data



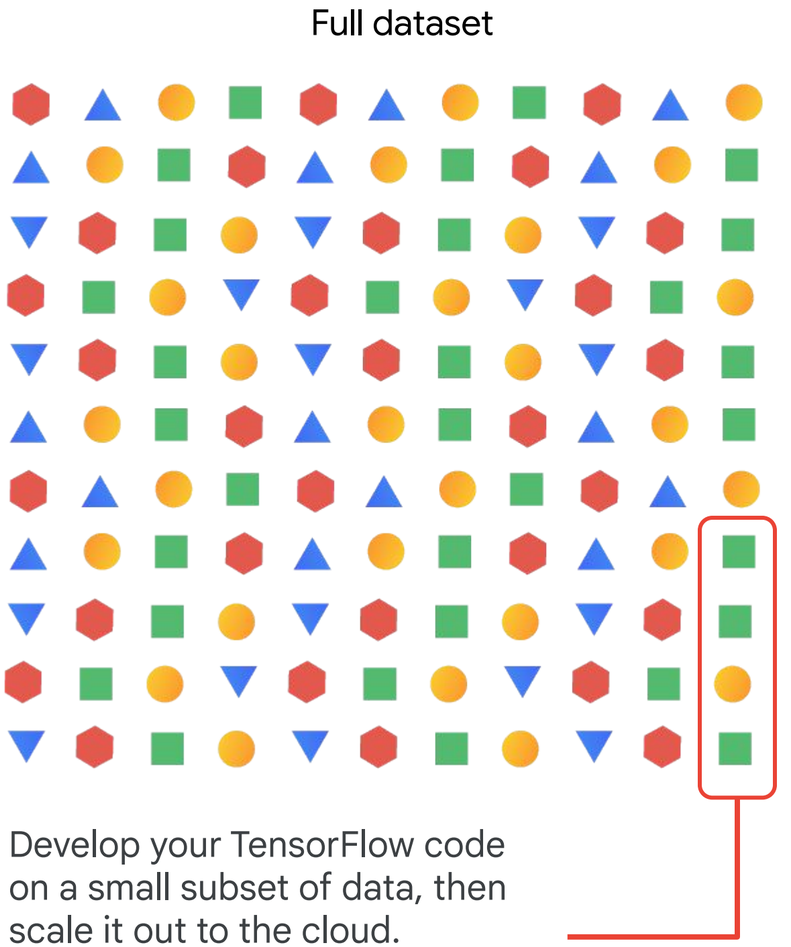
We hypothesize that flight delay depends on the carrier, time of day, weather, and airport characteristics (# of runways, etc.) We want to predict flight delays.

What field should we split our data on?

- Hash on date?
- Hash on airport?
- Hash on carrier name?

Split your data on a field you can afford to lose.

Developing the ML model software on the entire dataset can be expensive; you want to to develop on a smaller sample



Pitfall: Chaining hashes to create subsets won't work

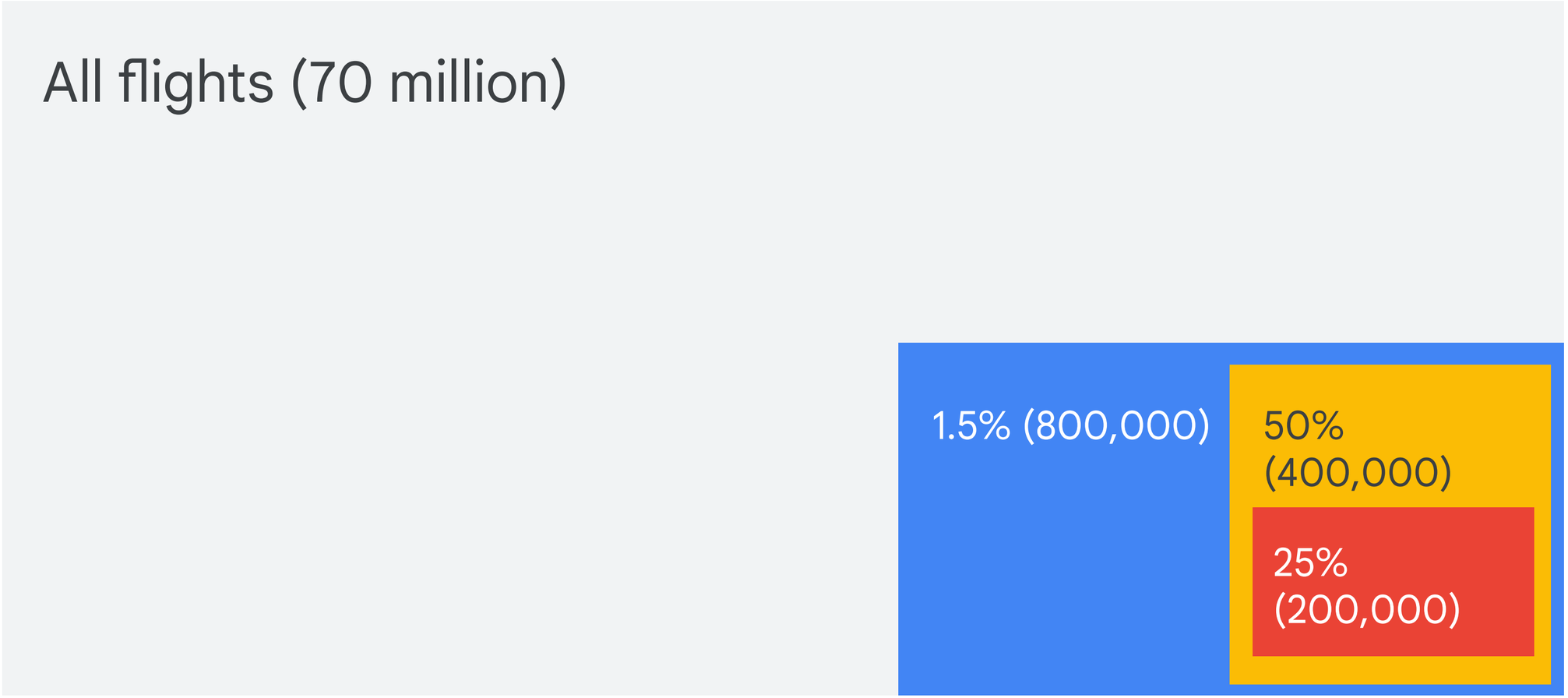
```
#standardSQL
SELECT
  date,
  airline,
  departure_airport,
  departure_schedule,
  arrival_airport,
  arrival_delay
FROM
 `bigquery-samples.airline_ontime_data.flights`
WHERE
  MOD(ABS(FARM_FINGERPRINT(date)),70) = 0
AND
  MOD(ABS(FARM_FINGERPRINT(date)),10) < 8
```

Then take 1 in 70 flights.

Take 80% of the dataset? Incorrect!

All records here will also be divisible by 10 (there is no new filtering happening!)

How we want to split our data



We can extend this to creating 3 splits

```
#standardSQL
SELECT
  date,
  airline,
  departure_airport,
  departure_schedule,
  arrival_airport,
  arrival_delay
FROM
 `bigquery-samples.airline_ontime_data.flights`
WHERE
  MOD(ABS(FARM_FINGERPRINT(date)),70) = 0
      AND
  MOD(ABS(FARM_FINGERPRINT(date)),700) >= 350
      AND
  MOD(ABS(FARM_FINGERPRINT(date)),700) < 525
```

Then take 1 in 70 flights.

Ignore the 50% of the dataset (training).

Choose data between 350 and 524 which is a new 25% sample for validation.

VertexAI: Overview

Course:
Launching into Machine Learning:
https://partner.cloudskillsboost.google/course_templates/8

LAB:
Training AutoML Classification Model - Structured Data:
https://partner.cloudskillsboost.google/course_sessions/2664667/labs/357113

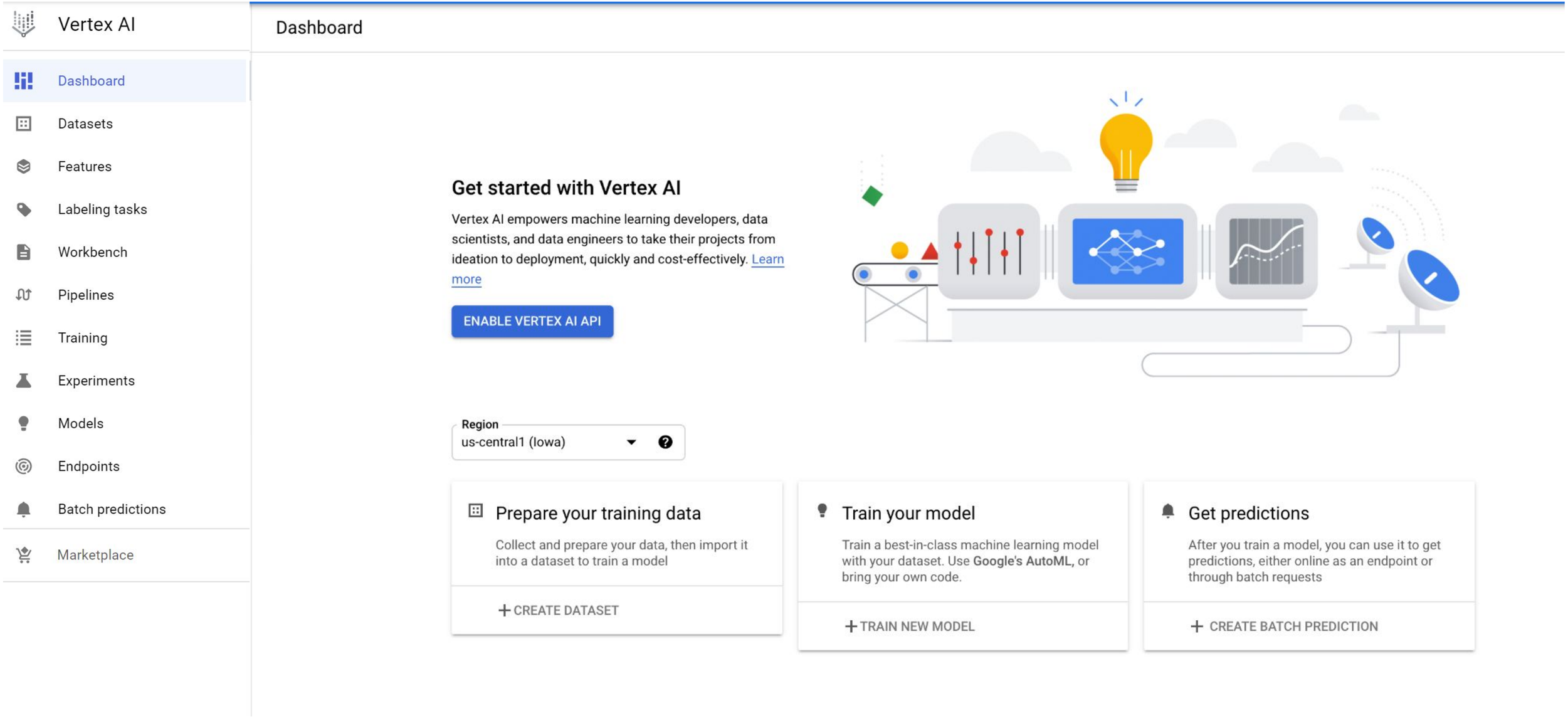
Build and Deploy Machine Learning Solutions on Vertex AI:
https://partner.cloudskillsboost.google/quests/183

Pause to setup your lab:
(link in the chat)

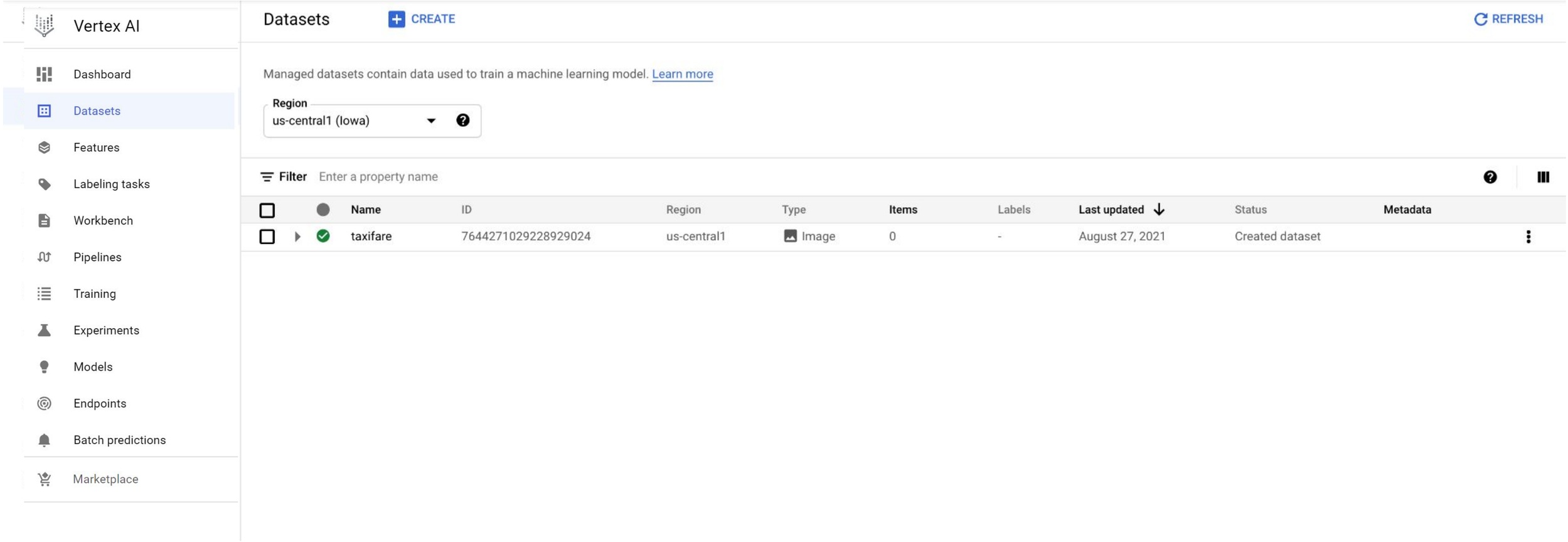
We will resume in:

<<8:00->>

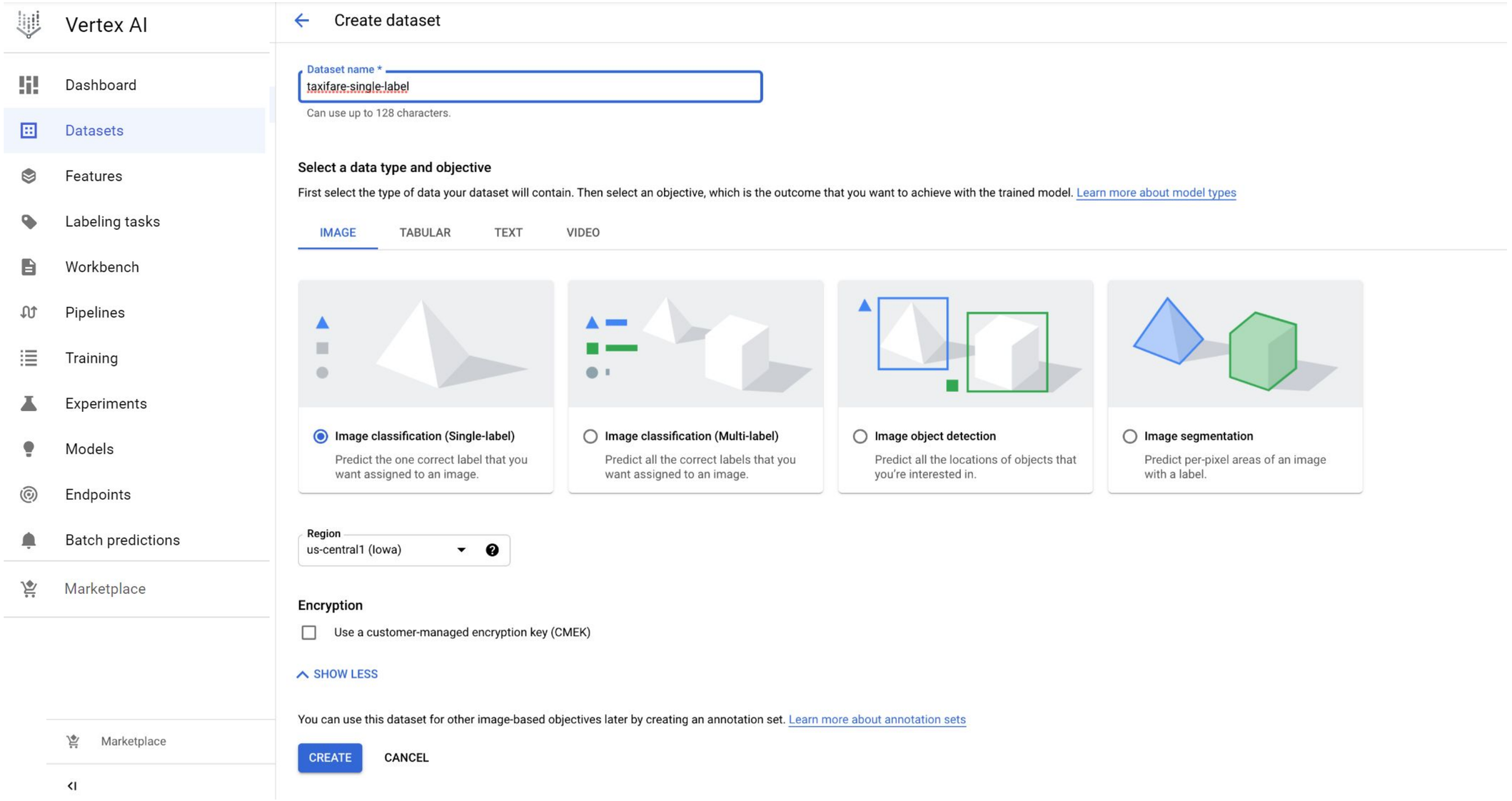
Vertex AI > Dashboard



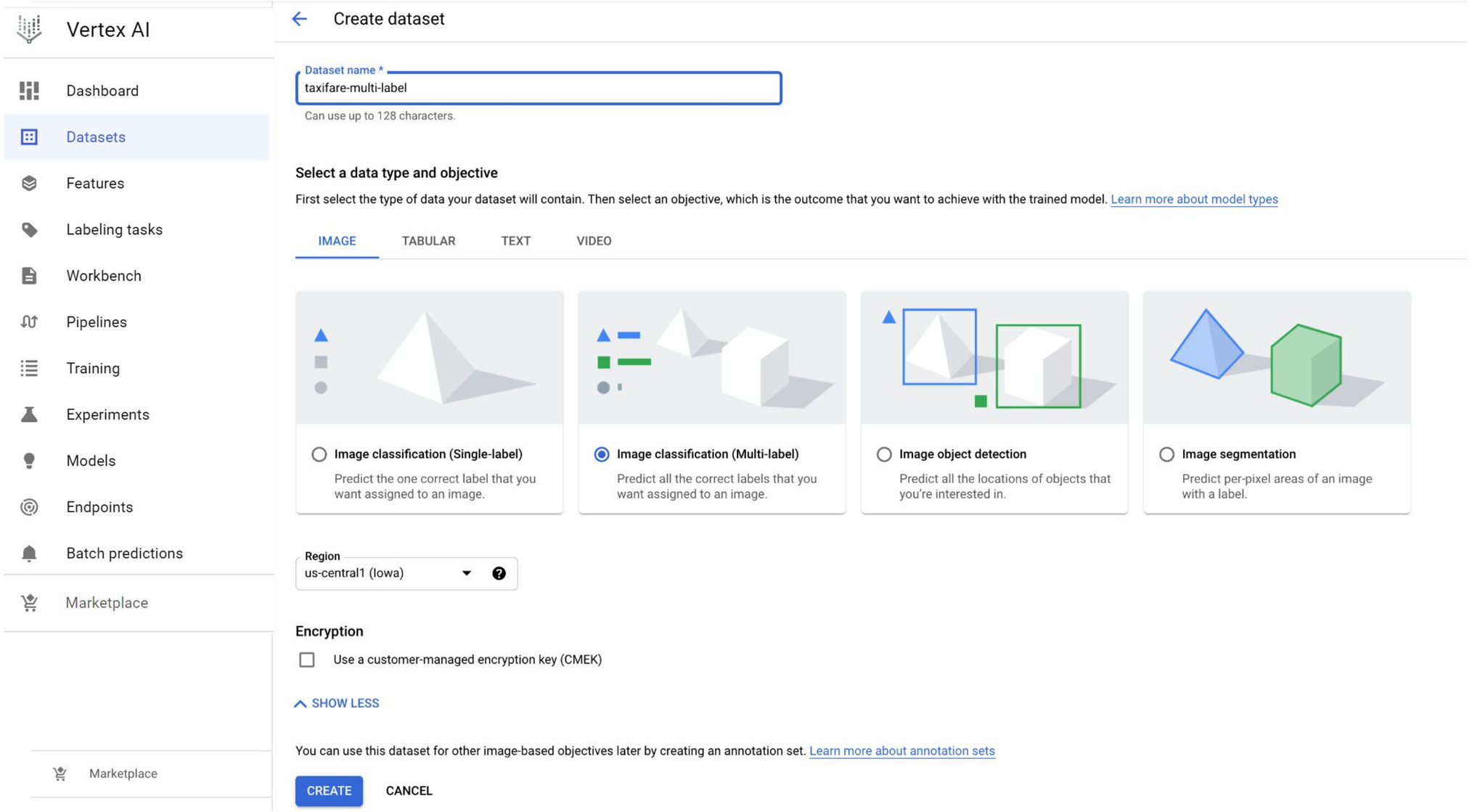
Vertex AI > Datasets



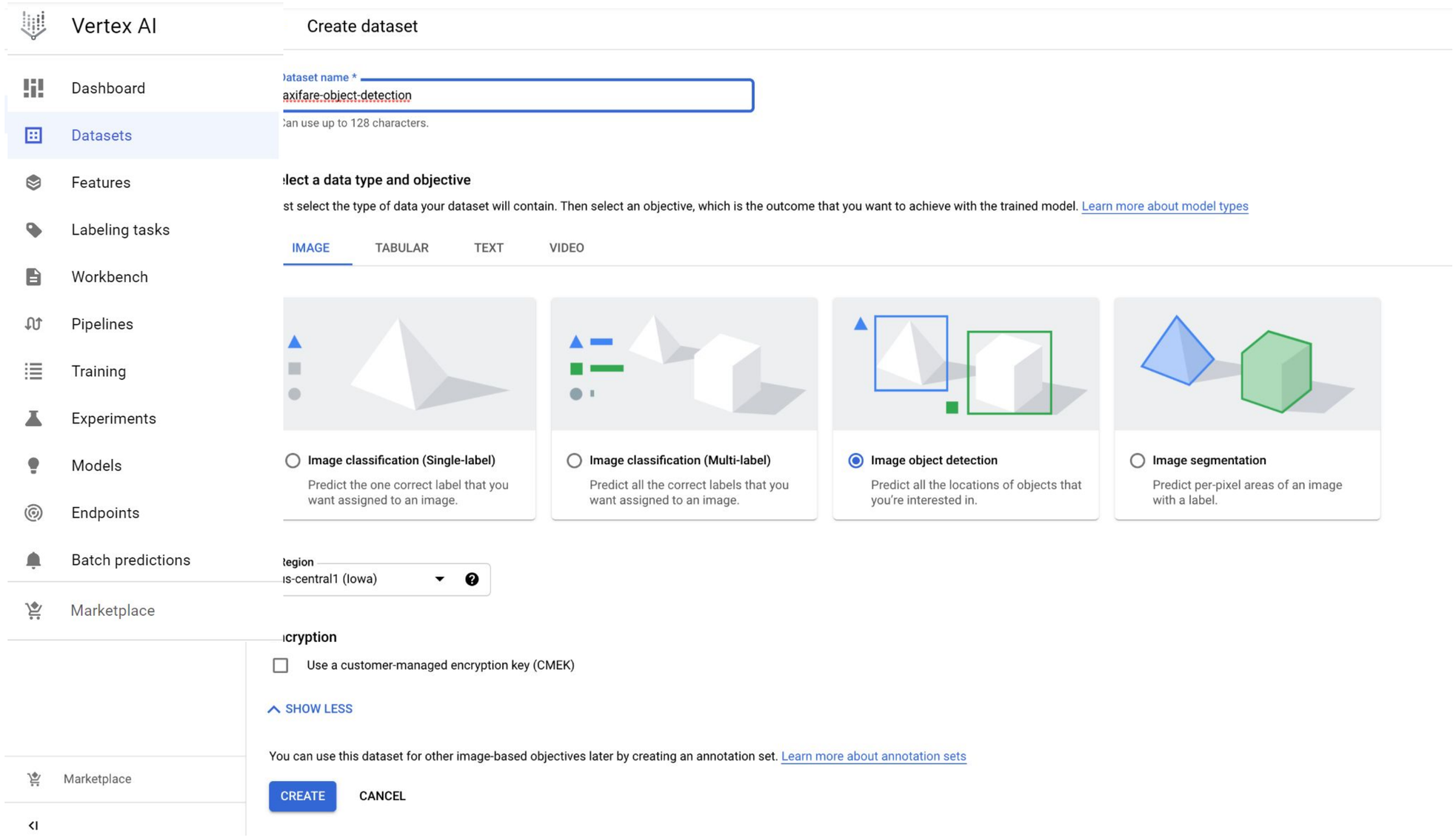
Vertex AI > Datasets > Create > Image classification (Single-label)



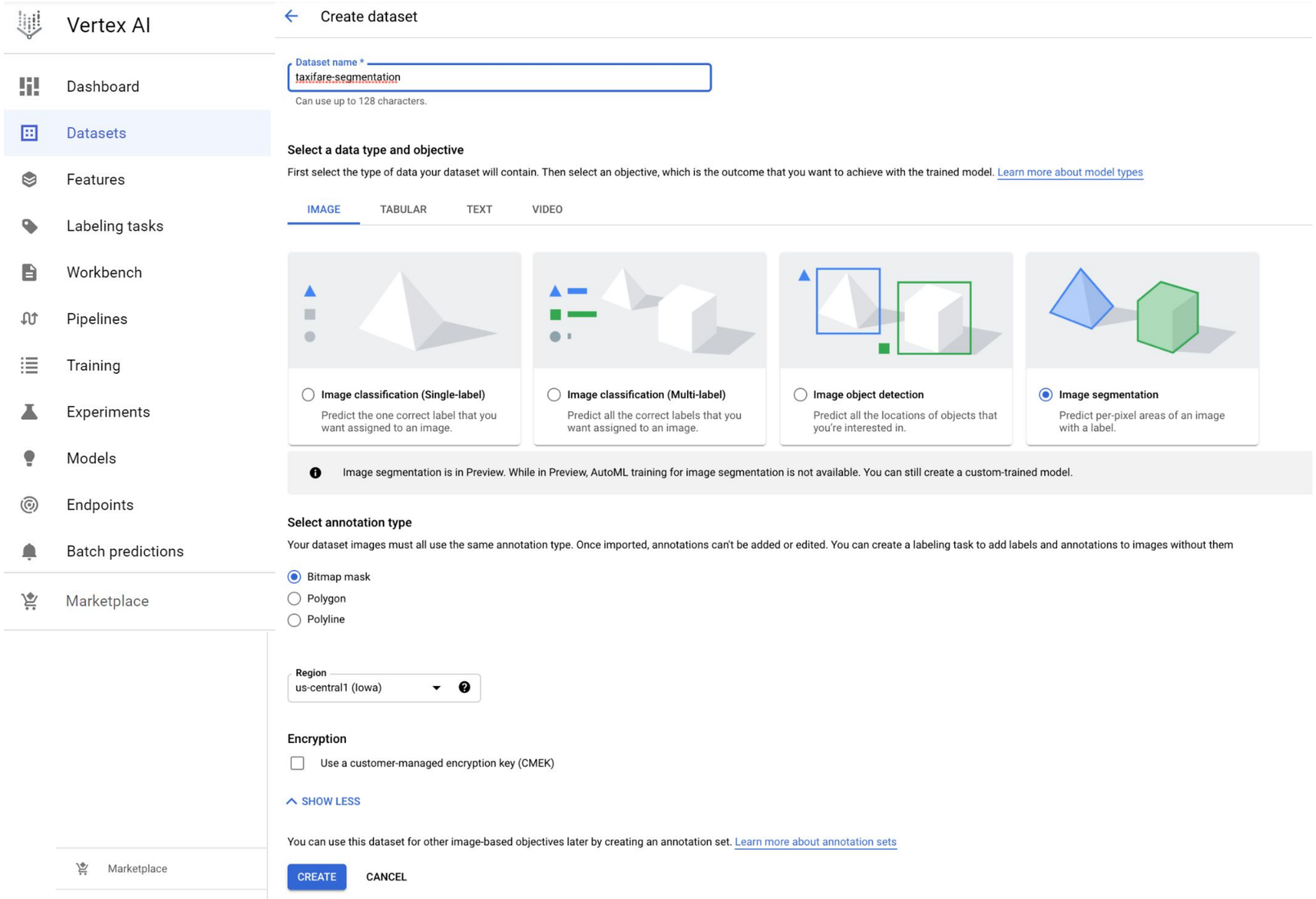
Vertex AI > Datasets > Create > Image classification (Multi-label)



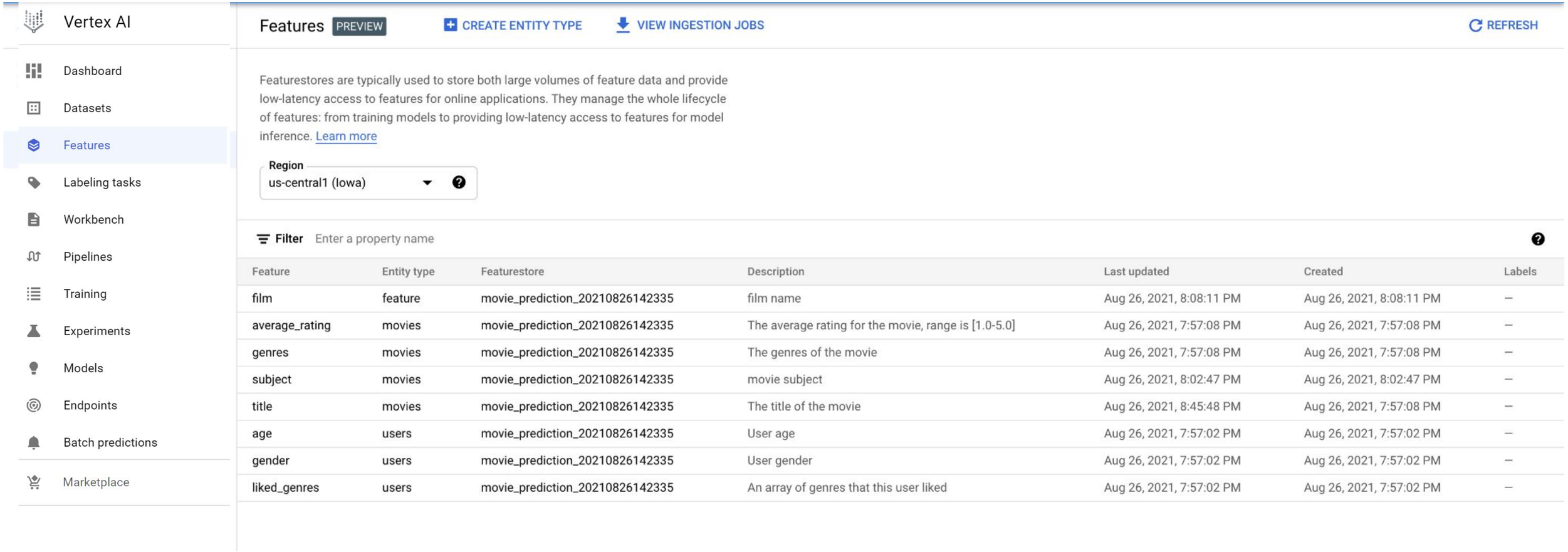
Vertex AI > Datasets > Create > Image object detection



Vertex AI > Datasets > Create > Image segmentation

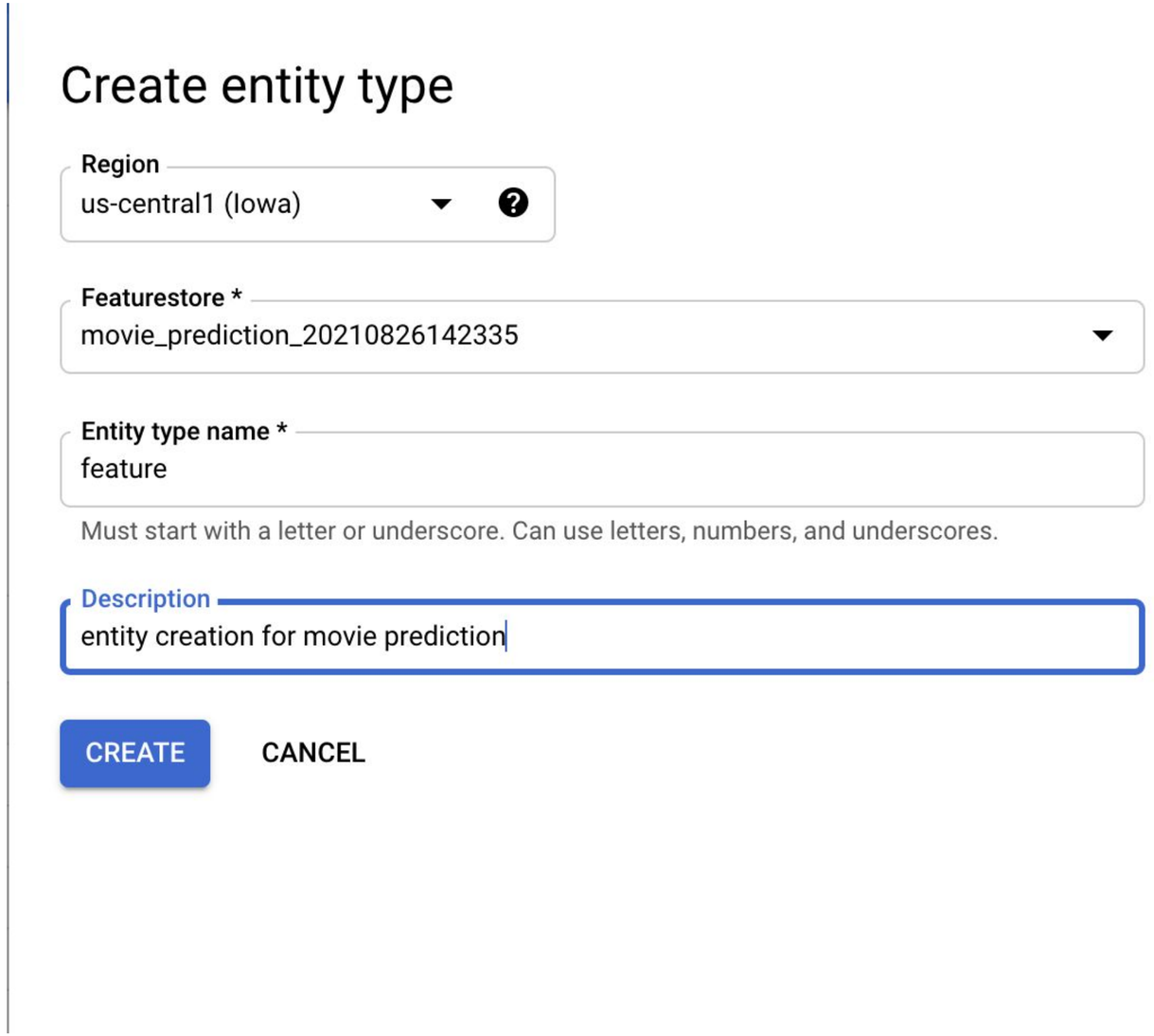


Vertex AI > Features



Feature	Entity type	Featurestore	Description	Last updated	Created	Labels
film	feature	movie_prediction_20210826142335	film name	Aug 26, 2021, 8:08:11 PM	Aug 26, 2021, 8:08:11 PM	–
average_rating	movies	movie_prediction_20210826142335	The average rating for the movie, range is [1.0-5.0]	Aug 26, 2021, 7:57:08 PM	Aug 26, 2021, 7:57:08 PM	–
genres	movies	movie_prediction_20210826142335	The genres of the movie	Aug 26, 2021, 7:57:08 PM	Aug 26, 2021, 7:57:08 PM	–
subject	movies	movie_prediction_20210826142335	movie subject	Aug 26, 2021, 8:02:47 PM	Aug 26, 2021, 8:02:47 PM	–
title	movies	movie_prediction_20210826142335	The title of the movie	Aug 26, 2021, 8:45:48 PM	Aug 26, 2021, 7:57:08 PM	–
age	users	movie_prediction_20210826142335	User age	Aug 26, 2021, 7:57:02 PM	Aug 26, 2021, 7:57:02 PM	–
gender	users	movie_prediction_20210826142335	User gender	Aug 26, 2021, 7:57:02 PM	Aug 26, 2021, 7:57:02 PM	–
liked_genres	users	movie_prediction_20210826142335	An array of genres that this user liked	Aug 26, 2021, 7:57:02 PM	Aug 26, 2021, 7:57:02 PM	–

Vertex AI > Features > Create Entity Type



Vertex AI > Features > Create Features

Create features

Feature names must start with a lowercase letter or underscore. Can use lowercase letters, numbers and underscores.

Feature name *	Value type *	Description
Feature name 1 * subject	Value type 1 * STRING	Description 1 movie subject

+ ADD ANOTHER FEATURE

CREATE CANCEL

Vertex AI > Features > View Ingestion Jobs

Ingestion jobs

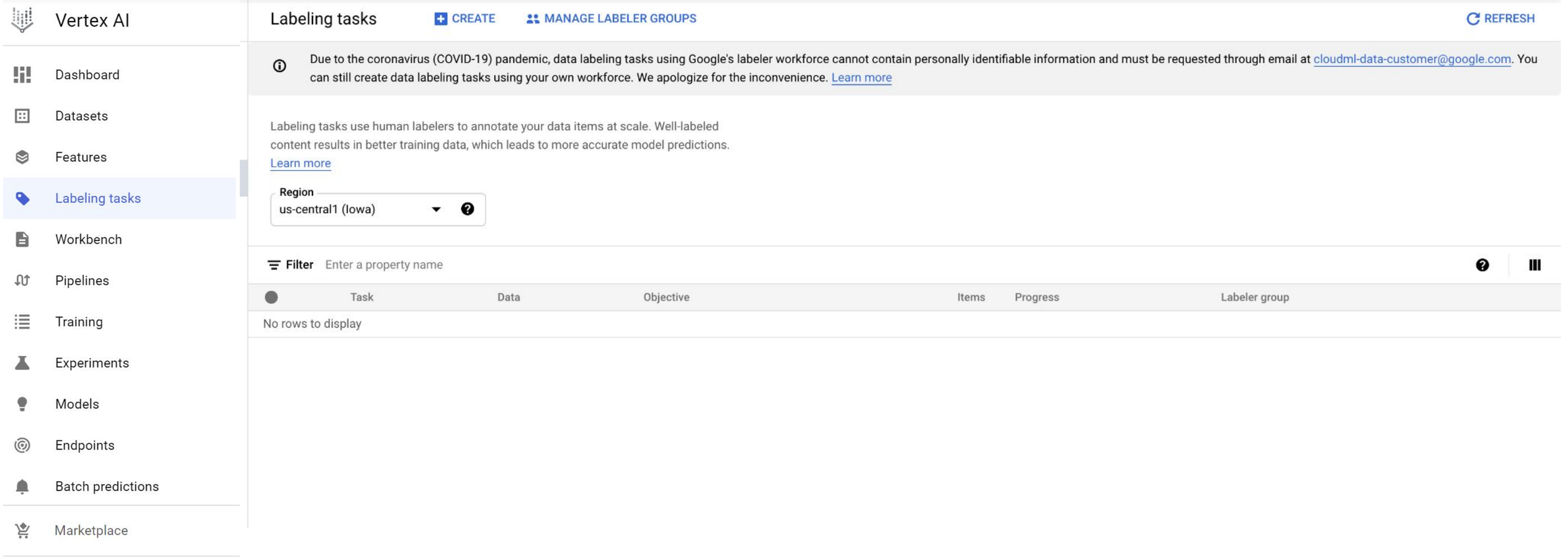
REFRESH

List of operations importing features from a storage source.

Region
us-central1 (Iowa)

Ingestion job	Entity type	Featurestore	Elapsed time	Entities	Feature values	Last updated	Started	Status
5600770669768867840	movies	movie_prediction_20210826142335	5 min 33 sec	4	12	Aug 26, 2021, 8:09:36 PM	Aug 26, 2021, 8:04:03 PM	Done
2250092547005218816	users	movie_prediction_20210826142335	5 min 35 sec	7	12	Aug 26, 2021, 8:03:06 PM	Aug 26, 2021, 7:57:31 PM	Done

Vertex AI > Labeling tasks



Vertex AI > Labeling tasks > Create

New labeling task

1. Define labeling task
2. Set up labels
3. Set up instructions
4. Customize labelers

START TASK CANCEL

Name *
labeling-task-1

Dataset *
movie

Annotation set *
movie_icn

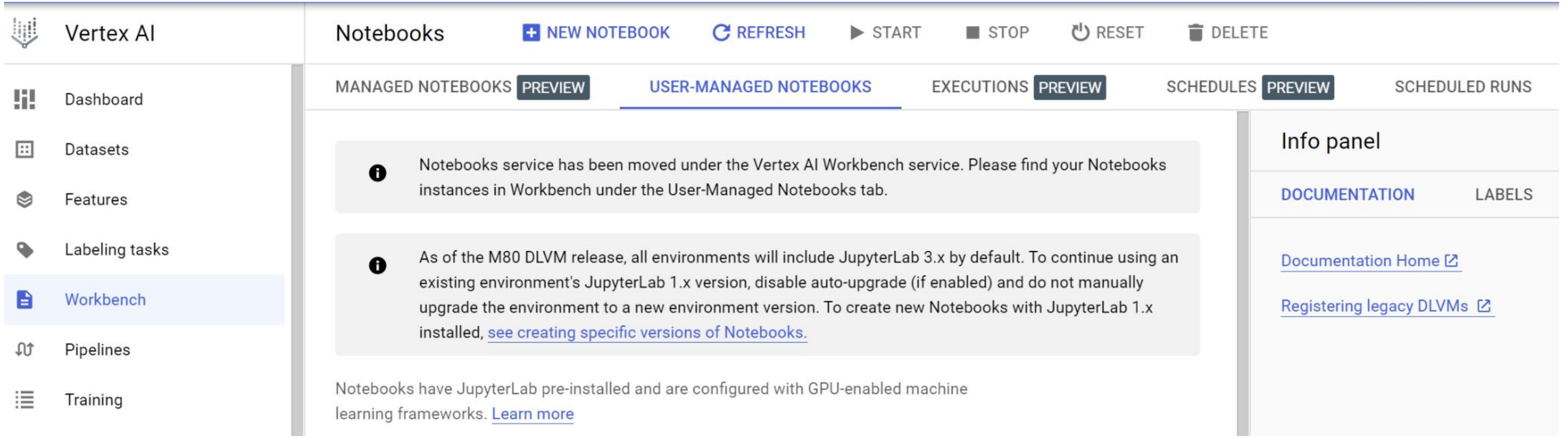
Objective
Image classification (Single-label)

Use active learning
Active learning expedites the labeling process by partially labeling your data with human labeler, then applying machine learning techniques to automatically label the rest.

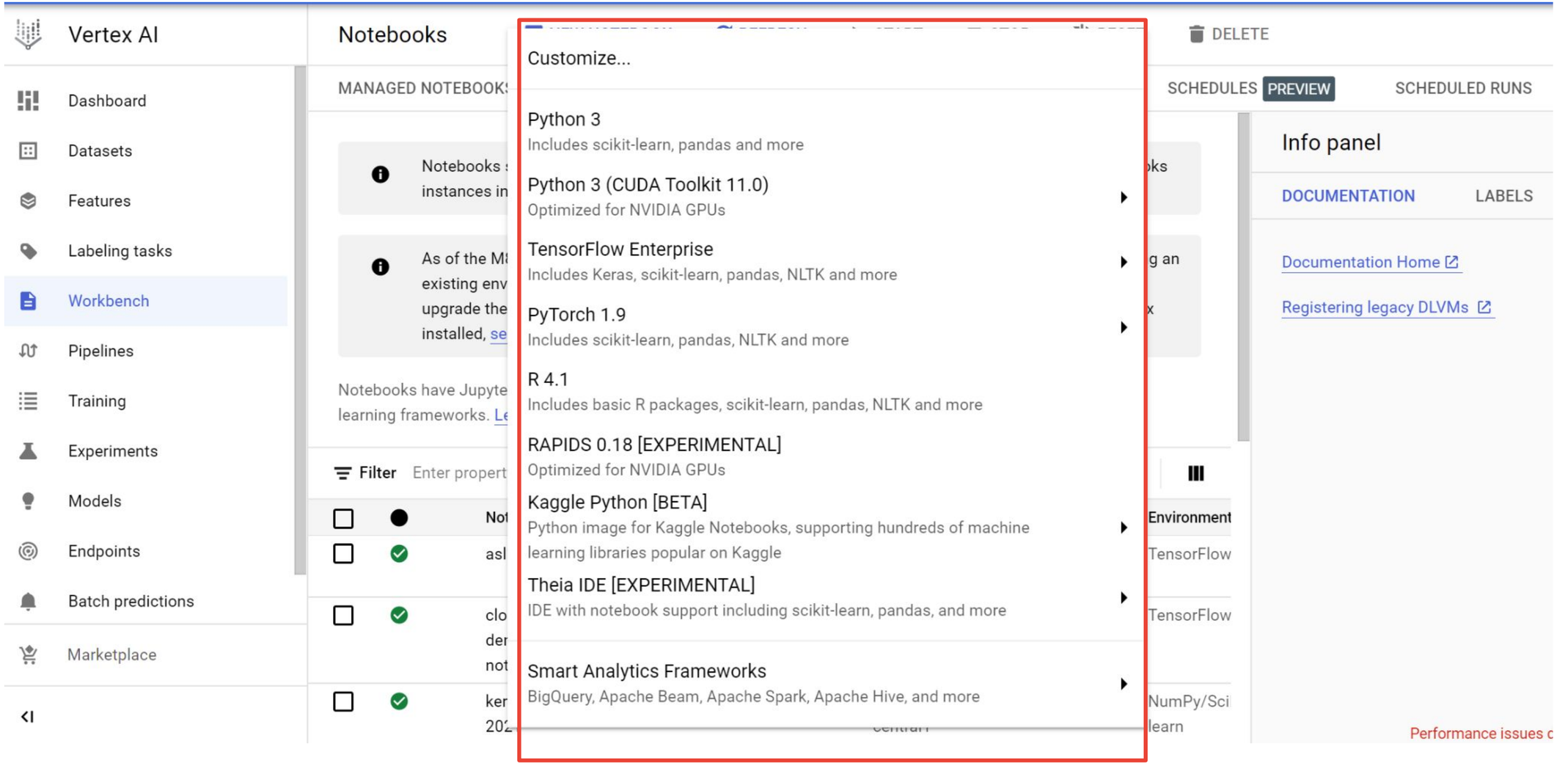
ADVANCED OPTIONS

CONTINUE

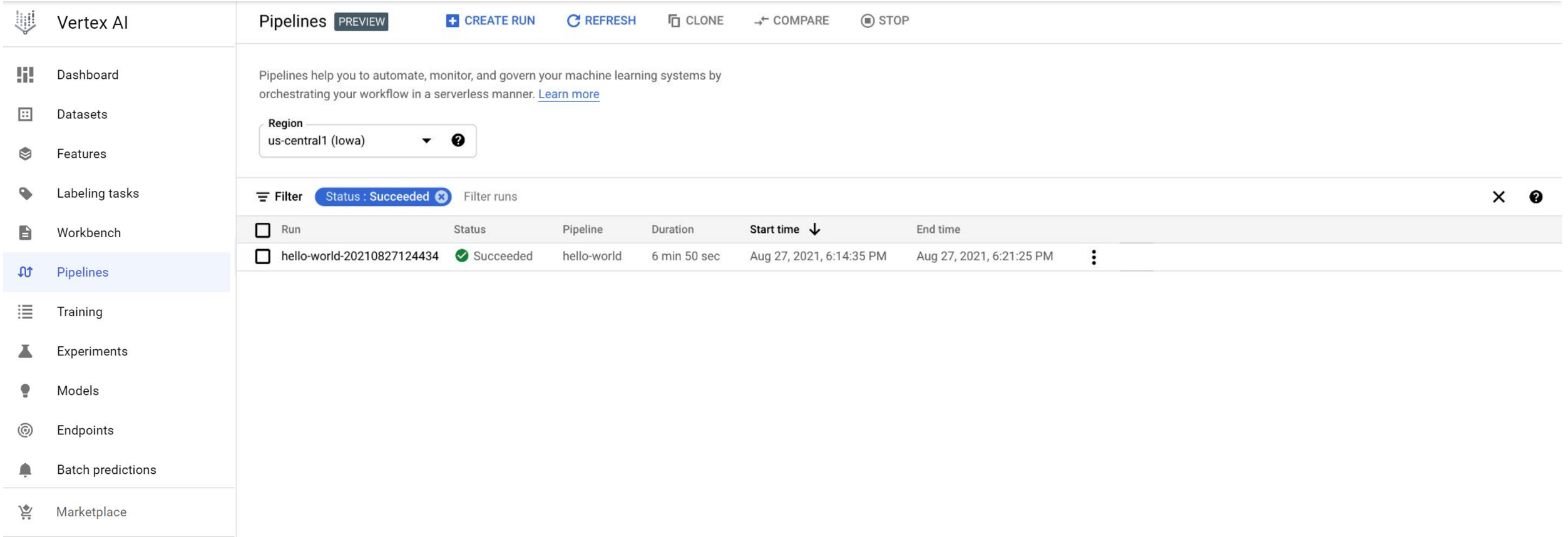
Vertex AI > Workbench



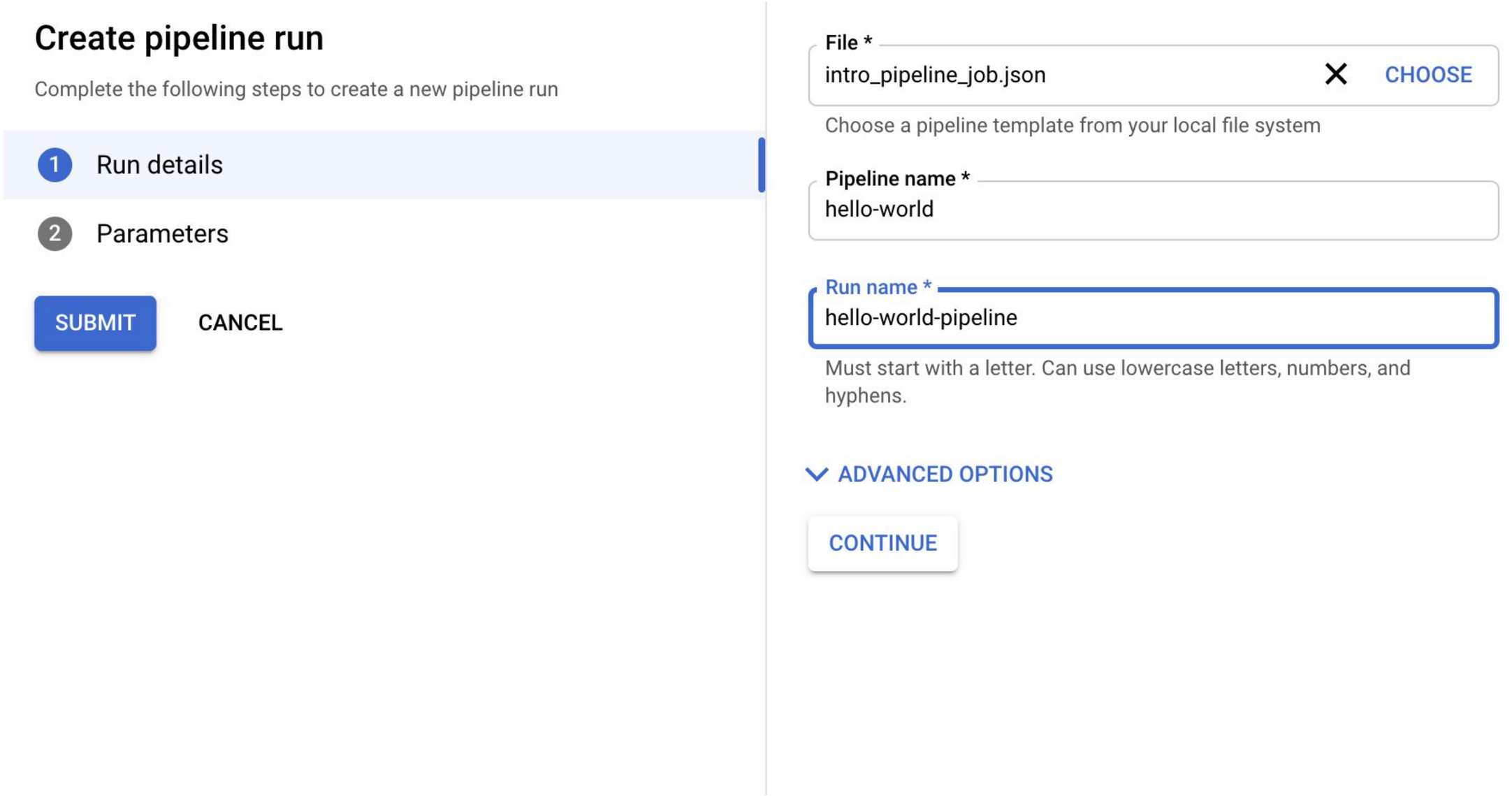
Vertex AI > Workbench > New Instance



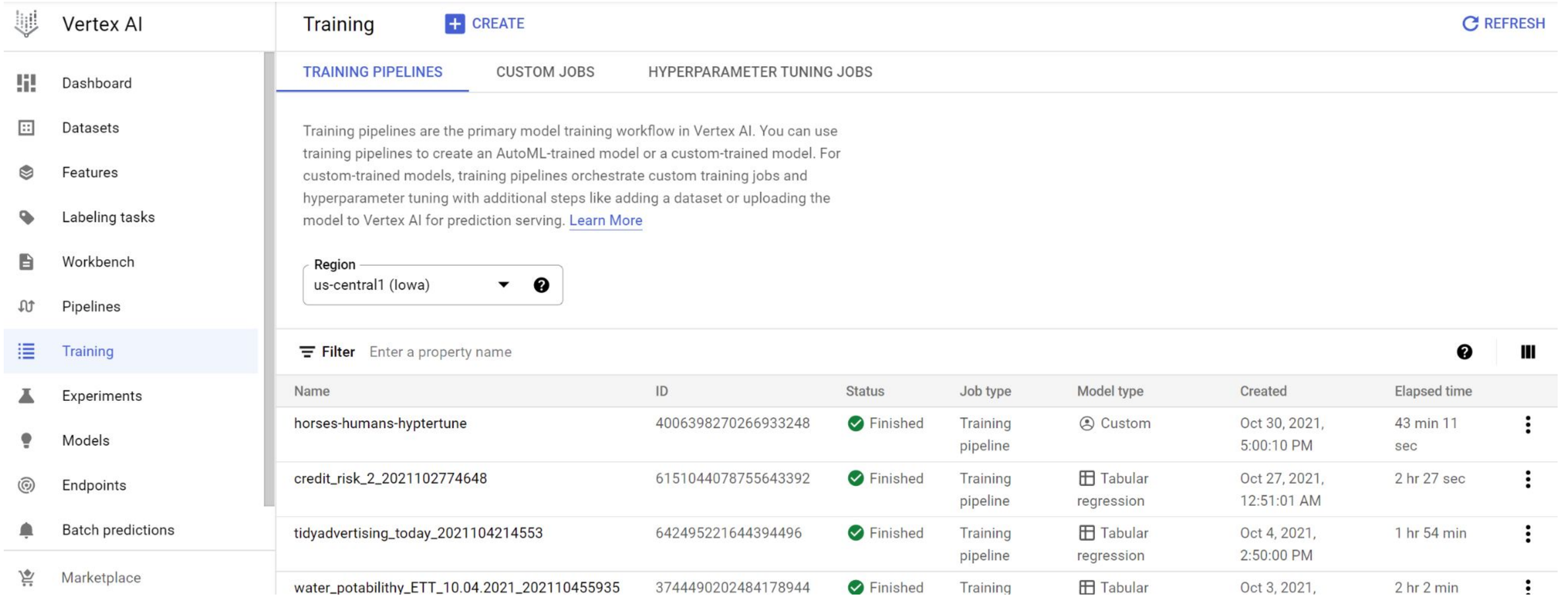
Vertex AI > Pipelines



Vertex AI > Pipelines > Create Run



Vertex AI > Training



Vertex AI > Training > Create

Train new model

- ✓ Training method
- ✓ Model details
- 3 Compute and pricing

START TRAINING CANCEL

Dataset *
movie (8 images)

Annotation set *
movie_icn

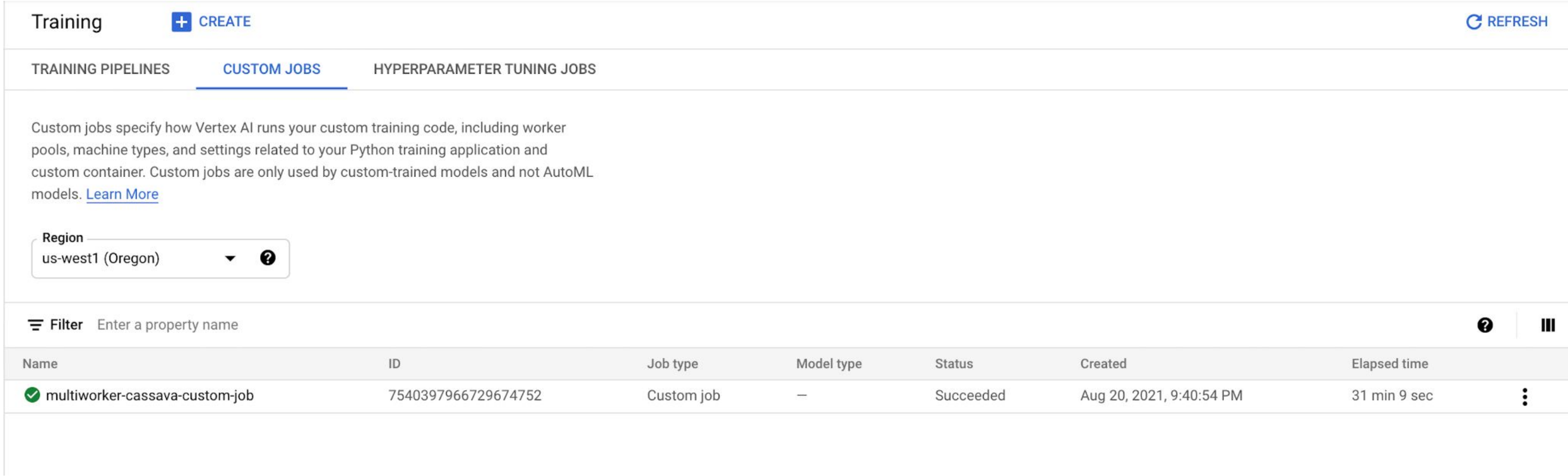
Objective
Image classification (Single-label)

Please refer to the pricing guide for more details (and available deployment options) for each method.

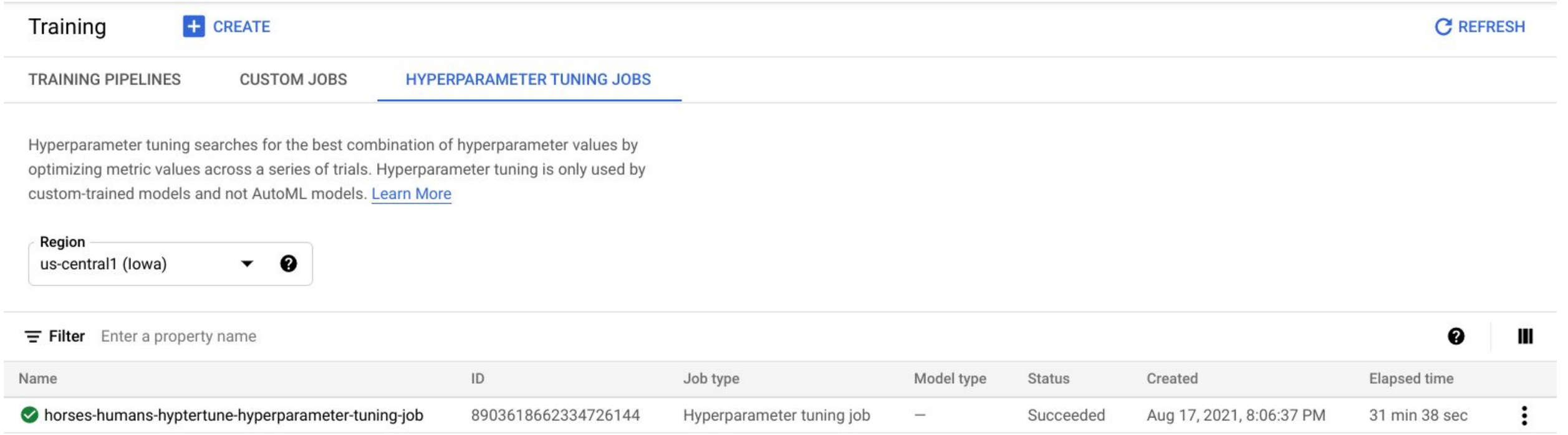
- (•) AutoML
  Train high-quality models with minimal effort and machine learning expertise. Just specify how long you want to train. Learn more
- ( ) AutoML Edge
  Train a model that can be exported for on-prem/on-device use. Typically has lower accuracy. Learn more
- ( ) Custom training (advanced)
  Run your TensorFlow, scikit-learn, and XGBoost training applications in the cloud. Train with one of Google Cloud's pre-built containers or use your own. Learn more

CONTINUE

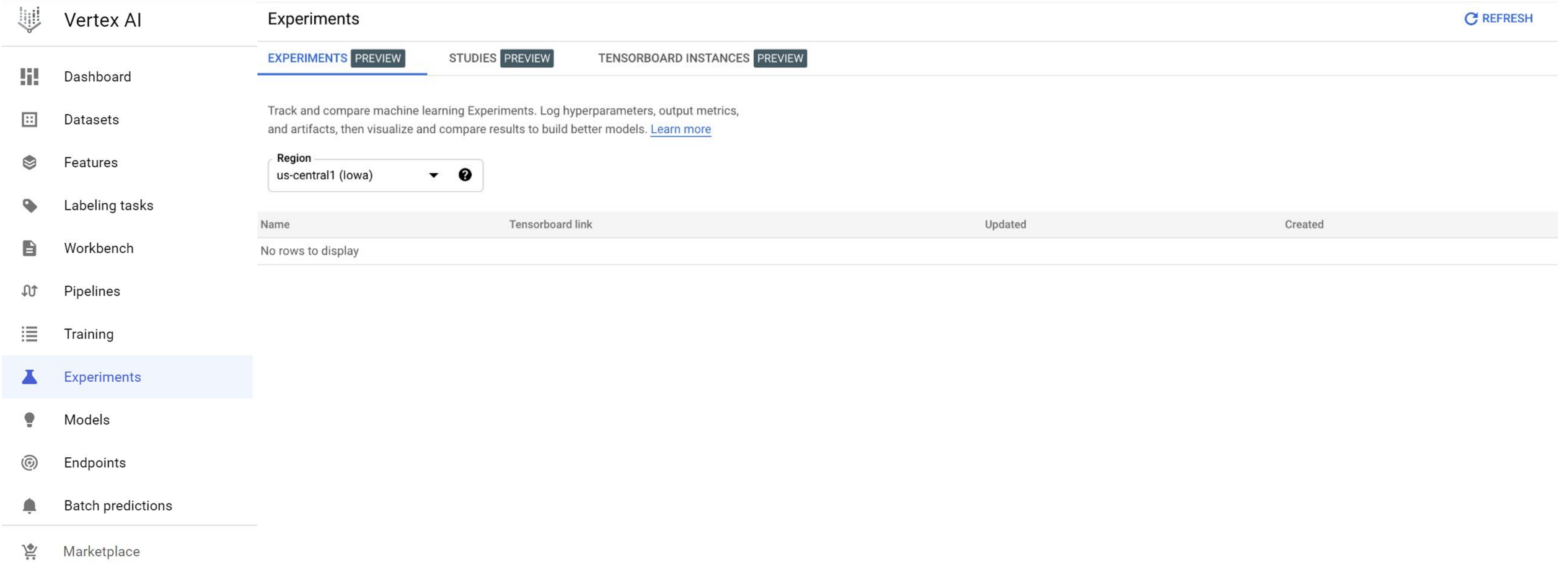
Vertex AI > Training > Custom Jobs



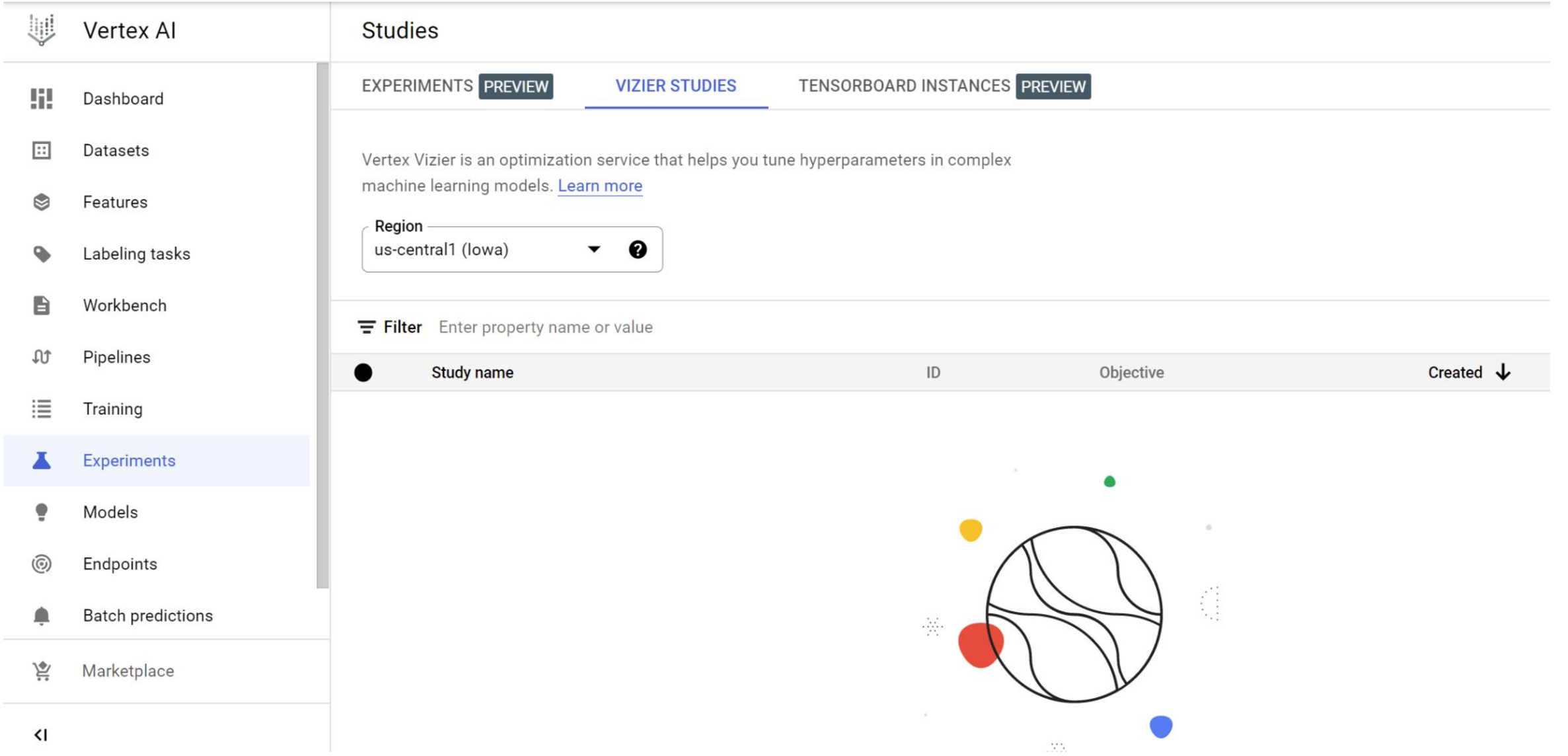
Vertex AI > Training > Hyperparameter Tuning Jobs



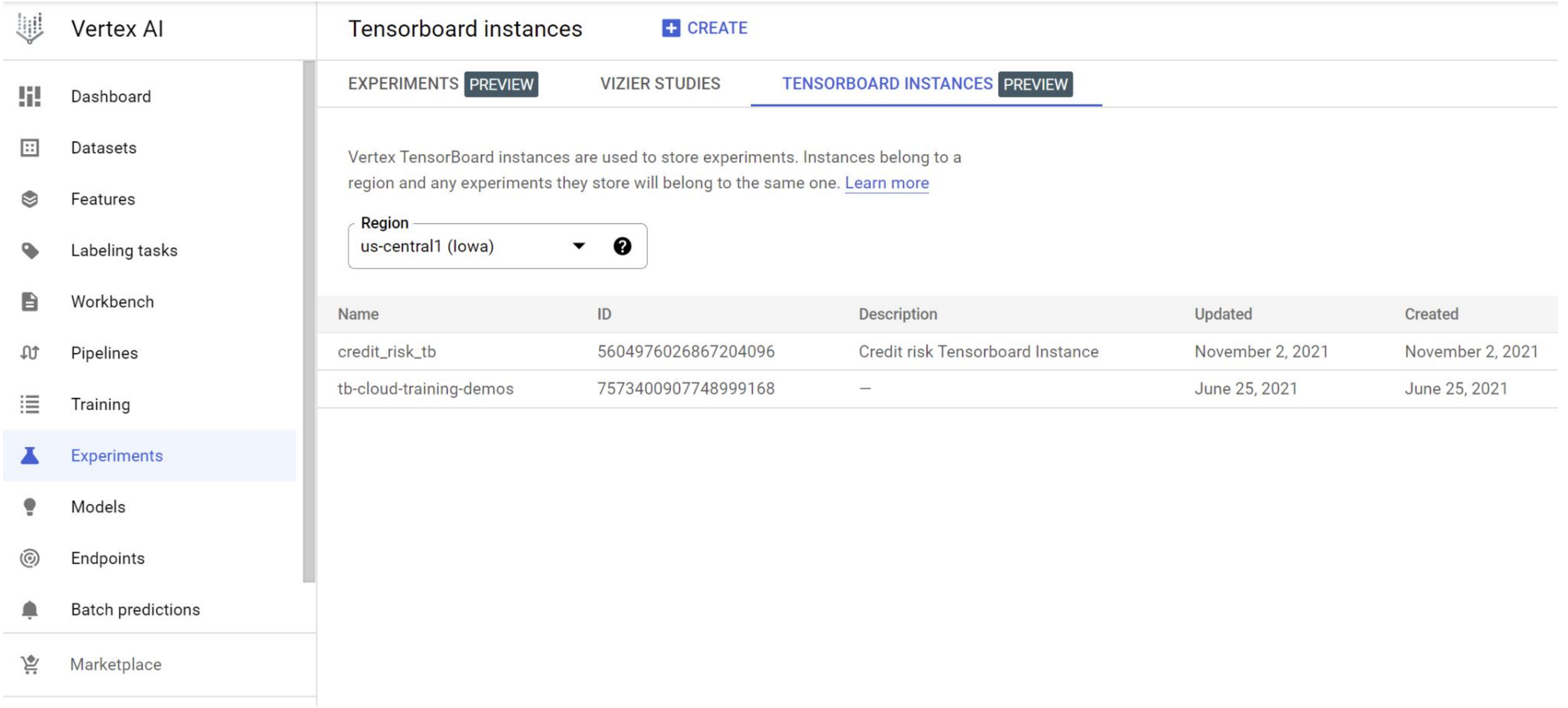
Vertex AI > Experiments > Experiments



Vertex AI > Experiments > Vizier Studies



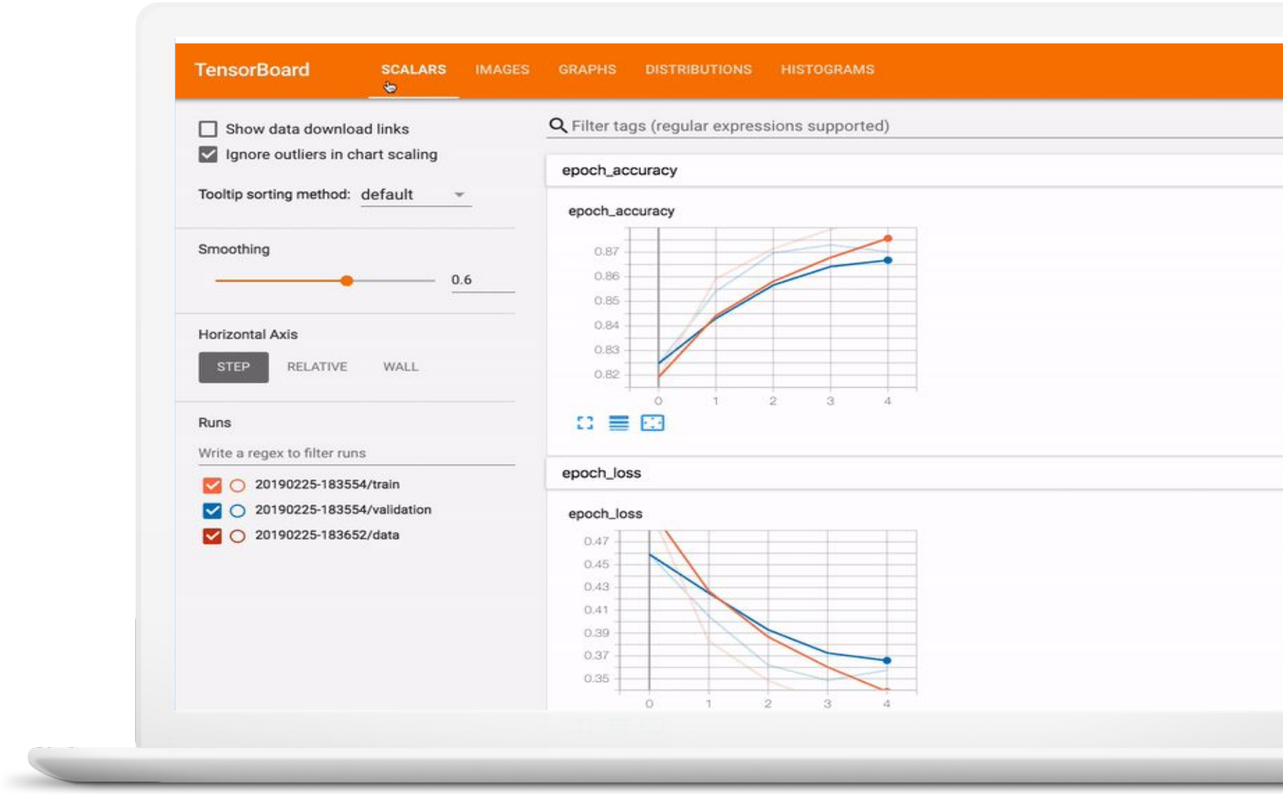
Vertex AI > Experiments > Tensorboard Instances



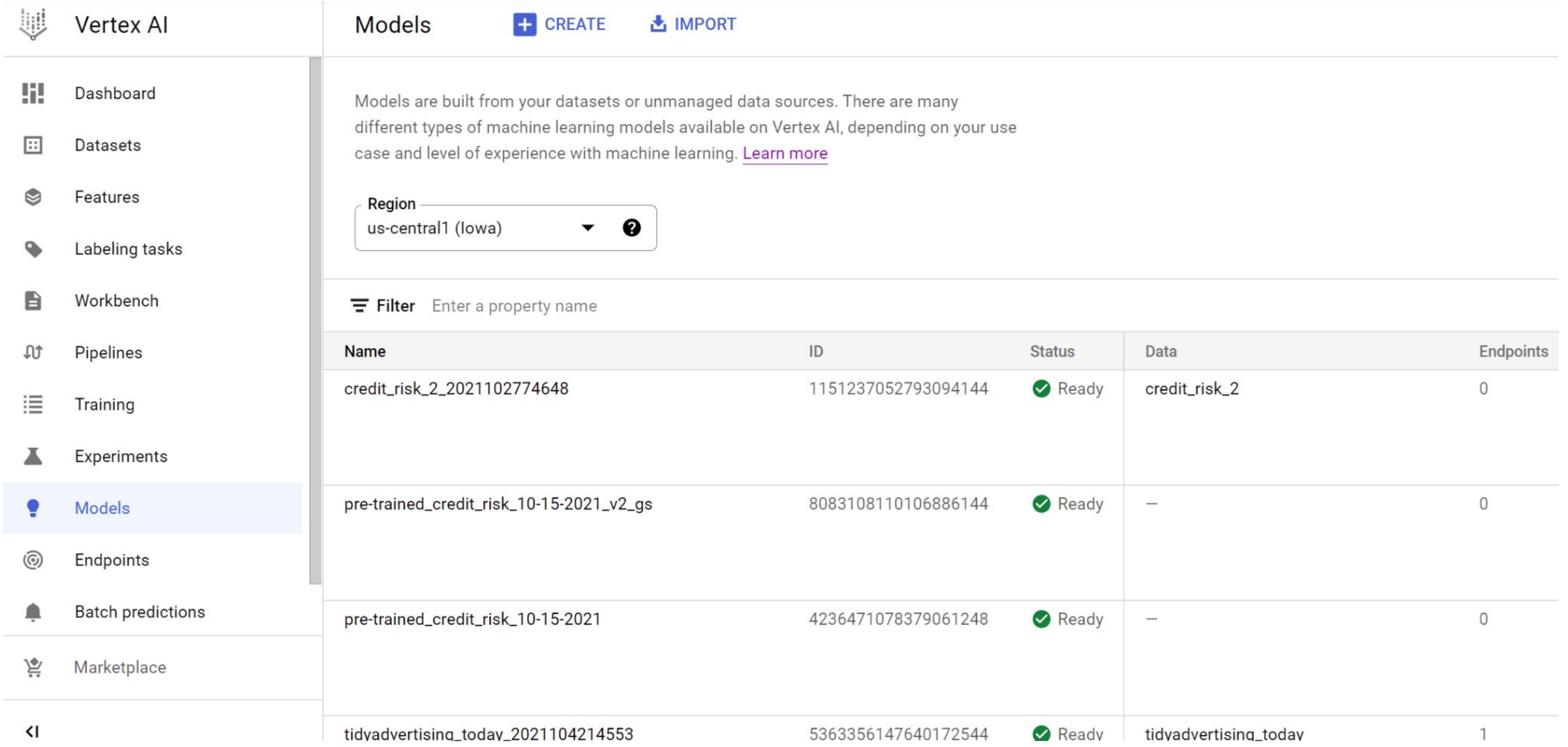
Vertex Tensorboard

Allow Data Scientists and ML Researchers to easily collaborate by making it seamless to track, compare, and share their experiments.

- Familiar experience: Experience matches the OSS TensorBoard tool users already know and love
- No Setup: Readily available TensorBoard that can be used with no installation or additional deployment
- Easy Collaboration: Share results and insights with peers to accelerate ML research and development
- Search & Compare: Quickly find specific experiments (e.g. based on hyperparameters) and compare them
- Enterprise Ready: Secure, cost-effective, scalable and integrated with the rest of GCP



Vertex AI > Models



Vertex AI > Models > Create

Train new model

- Training method
- Model details
- 3 Compute and pricing

START TRAINING CANCEL

Dataset *
movie (8 images)

Annotation set *
movie_icn

Objective
Image classification (Single-label)

Please refer to the pricing guide for more details (and available deployment options) for each method.

- AutoML
  Train high-quality models with minimal effort and machine learning expertise. Just specify how long you want to train. Learn more
- AutoML Edge
  Train a model that can be exported for on-prem/on-device use. Typically has lower accuracy. Learn more
- Custom training (advanced)
  Run your TensorFlow, scikit-learn, and XGBoost training applications in the cloud. Train with one of Google Cloud's pre-built containers or use your own. Learn more

CONTINUE

Vertex AI > Models > Import

Import model

1. Name and region
2. Model settings

IMPORT CANCEL

Import model artifacts into a new pre-built container
View the list of supported runtimes including TensorFlow, scikit-learn and XGBoost versions

Import an existing custom container
Build a custom Docker container. Must be stored in Container Registry or Artifact Registry

Pre-built container settings

In order to run in a pre-built container, your code needs to be in Python 3.7

Model framework *
TensorFlow

Model framework version *
2.5

Accelerator type *
None

Model artifact location (Cloud storage path) *
gs:// cloud-ai-platform-f267098a-366f-478f-831d-bfa8d2ab9a5a BROWSE

Path to the Cloud Storage directory where the exported model file is stored (not the path to the model file itself). The model name must be one of: saved_model.pb, model.pkl, model.joblib, or model.bst, depending on which library you used.

Predict schemata

Optional. Learn more about the predict schemata

gs:// Instances BROWSE

Cloud Storage location to a YAML file that defines the format of a single instance used in prediction and explanation requests.

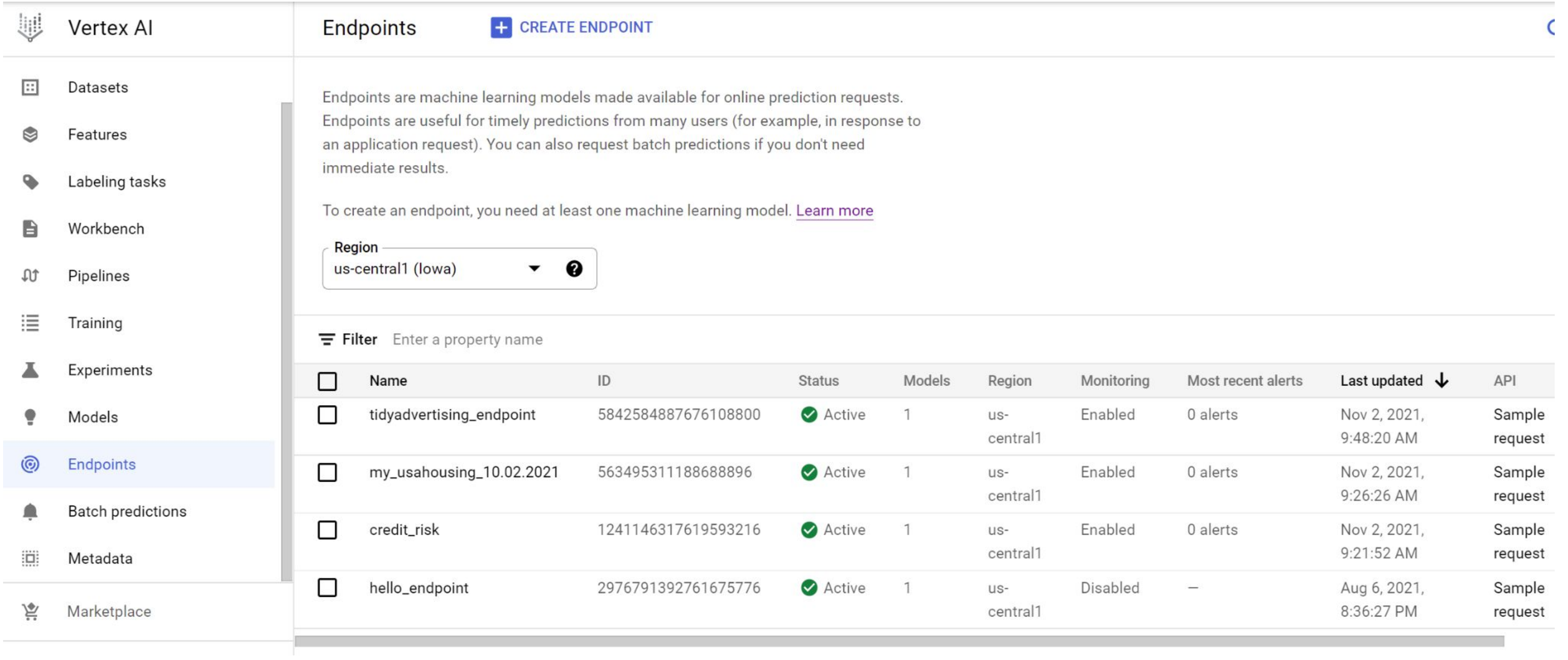
gs:// Parameters BROWSE

Cloud Storage location to a YAML file that defines the prediction and explanation parameters.

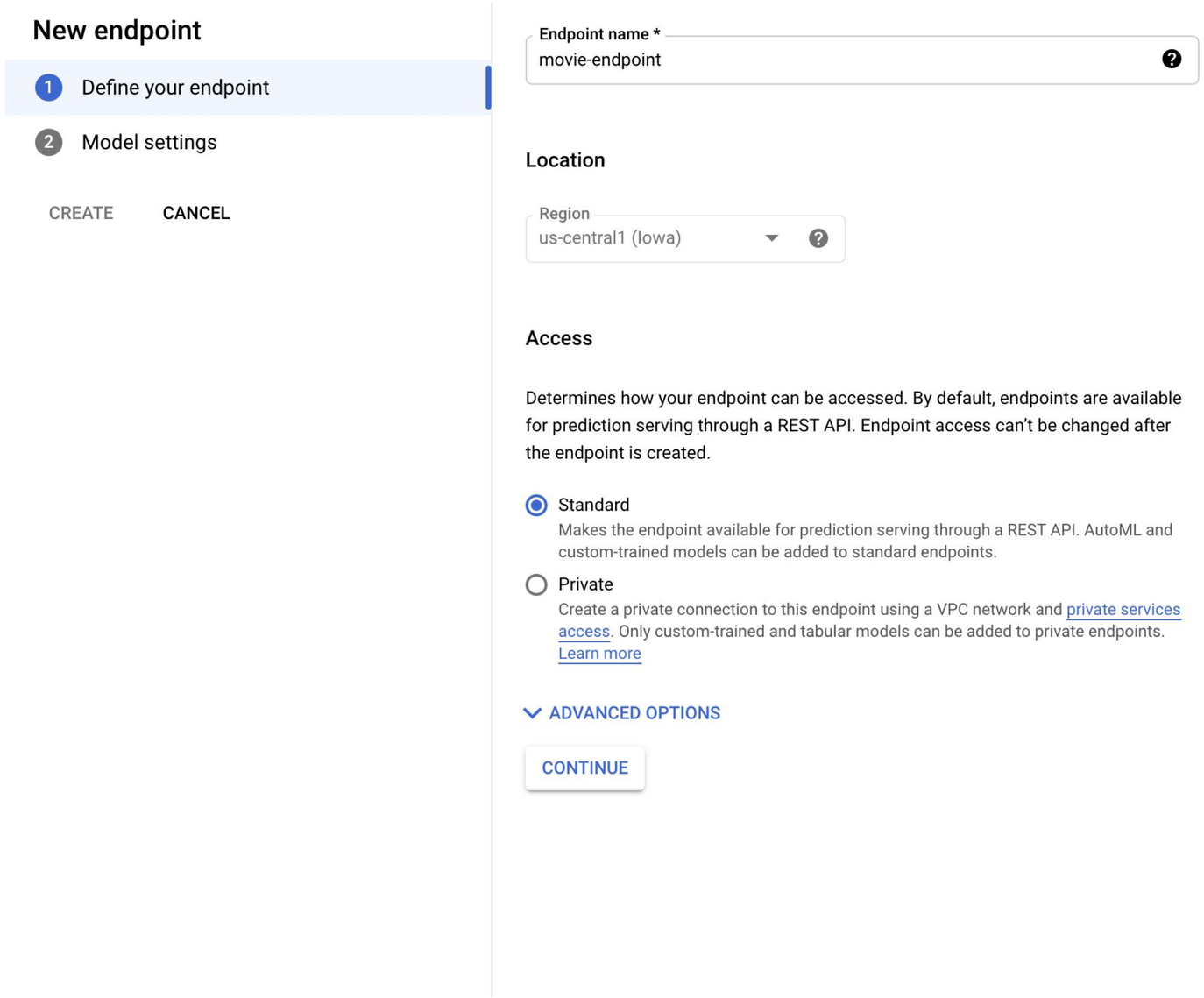
gs:// Predictions BROWSE

Cloud Storage location to a YAML file that defines the format of a single prediction or explanation.

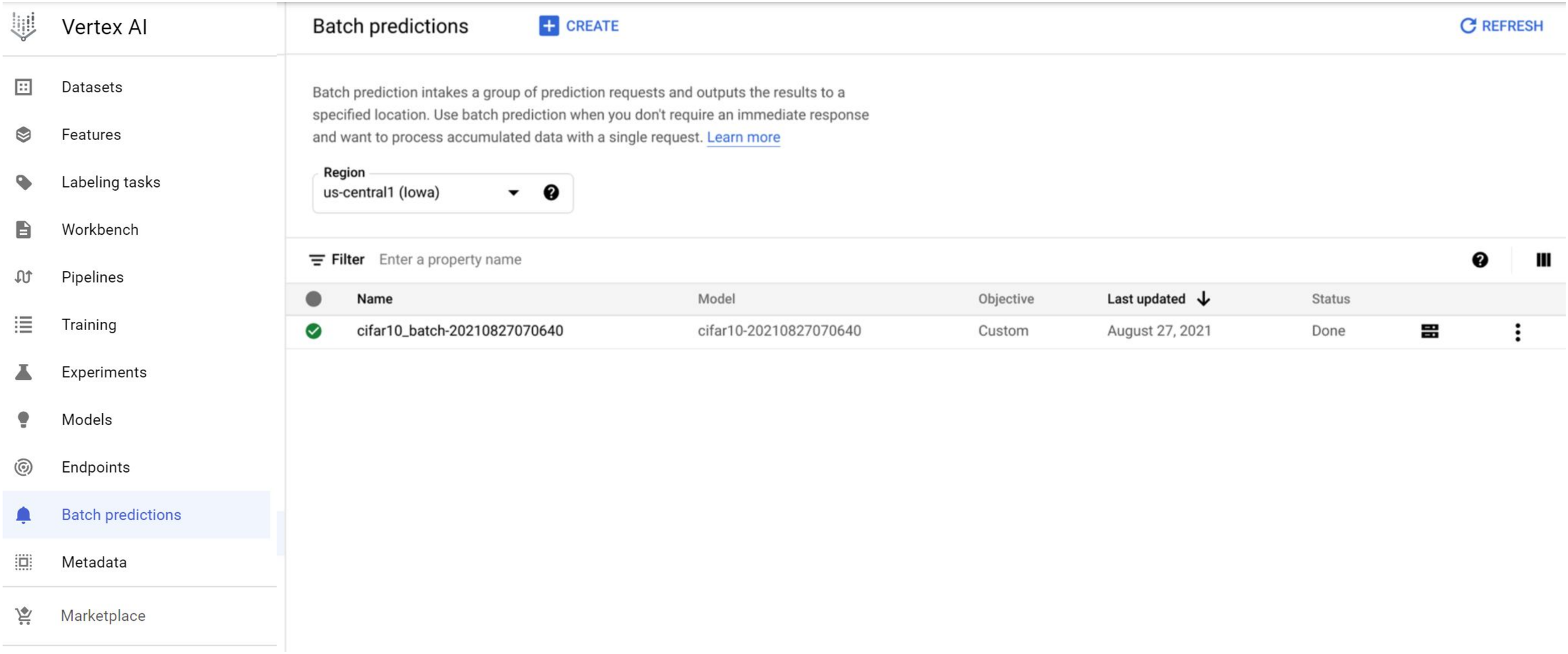
Vertex AI > Endpoints



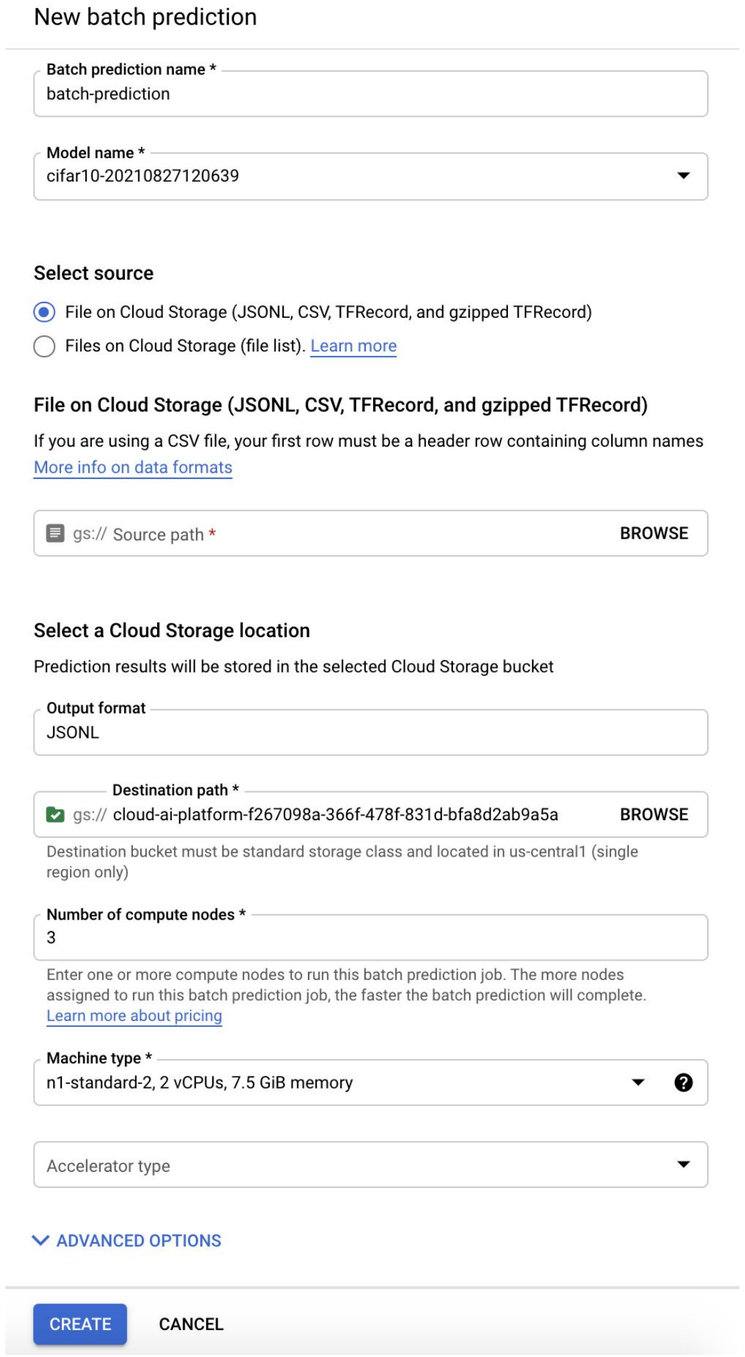
Vertex AI > Endpoints > Create Endpoint



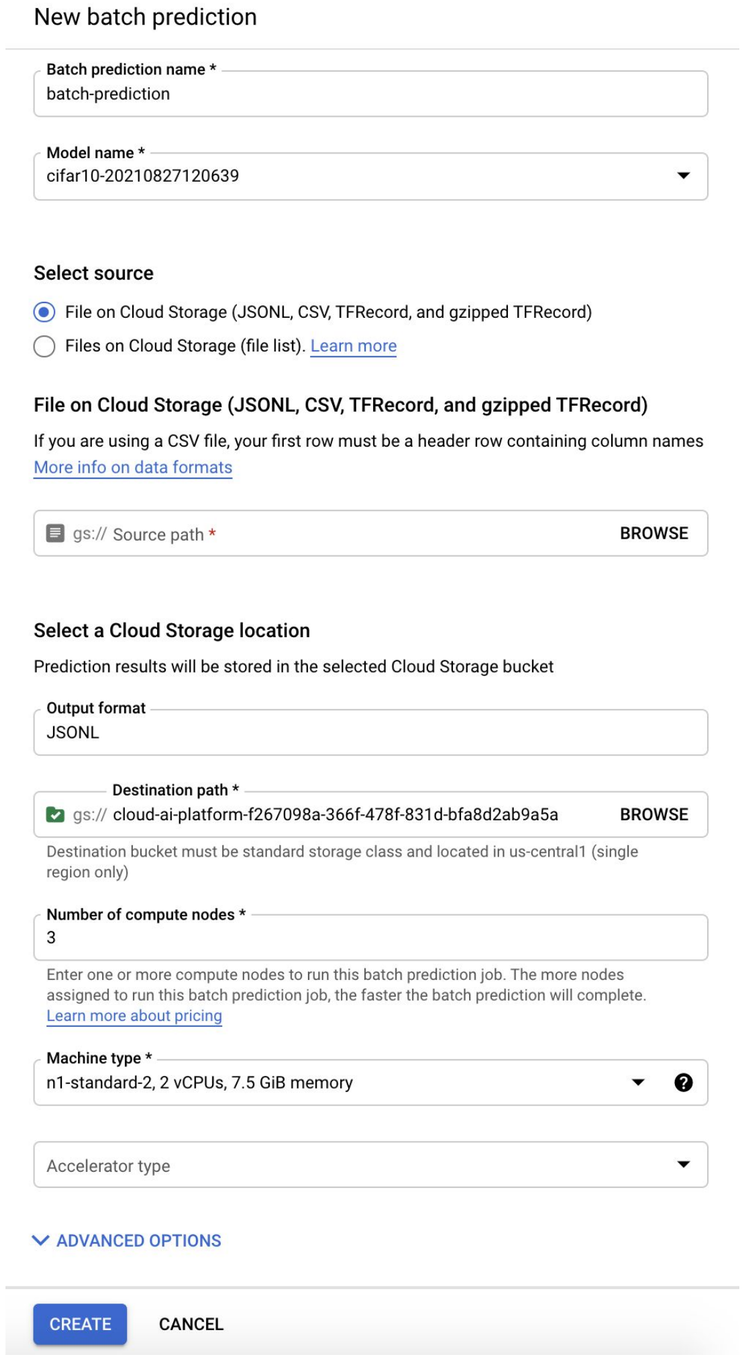
Vertex AI > Batch Predictions



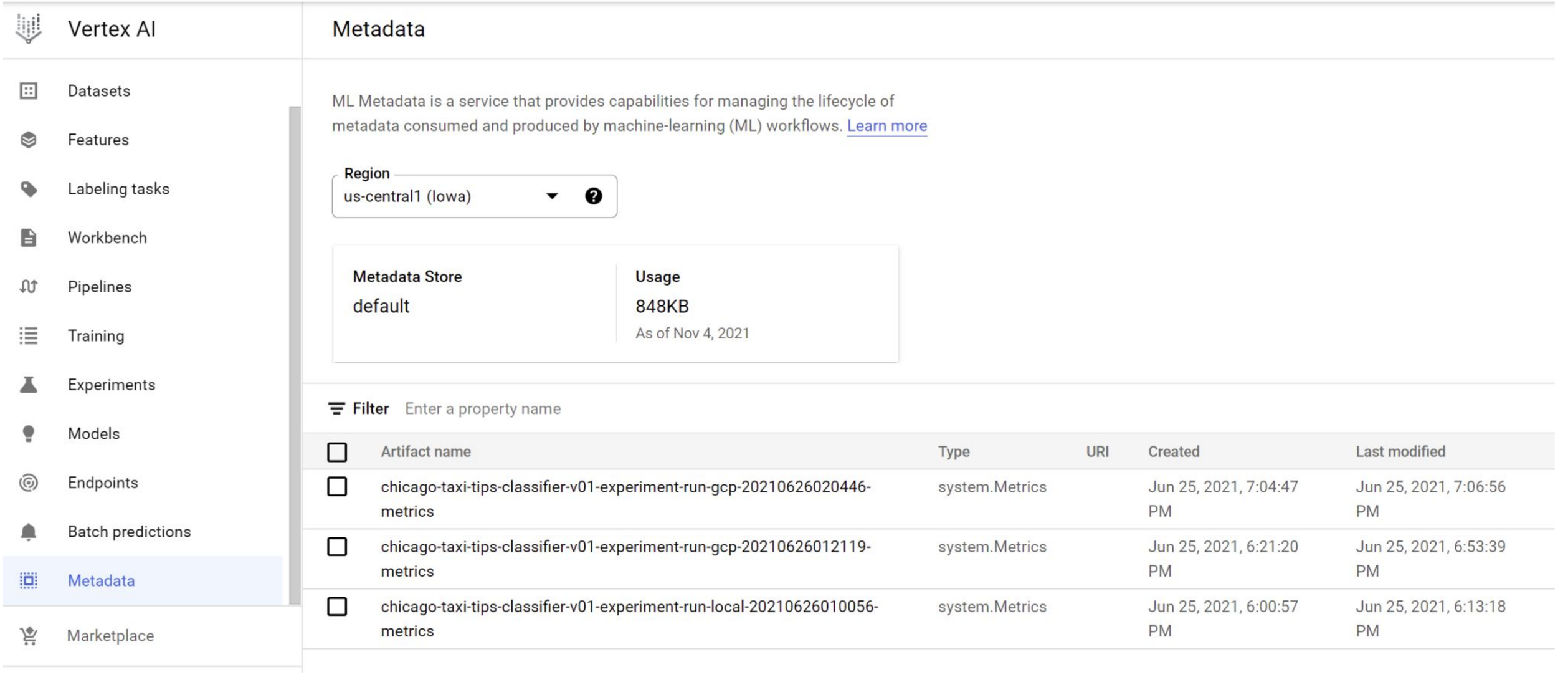
Vertex AI > Batch Predictions > Create



Vertex AI > Batch Predictions > Create



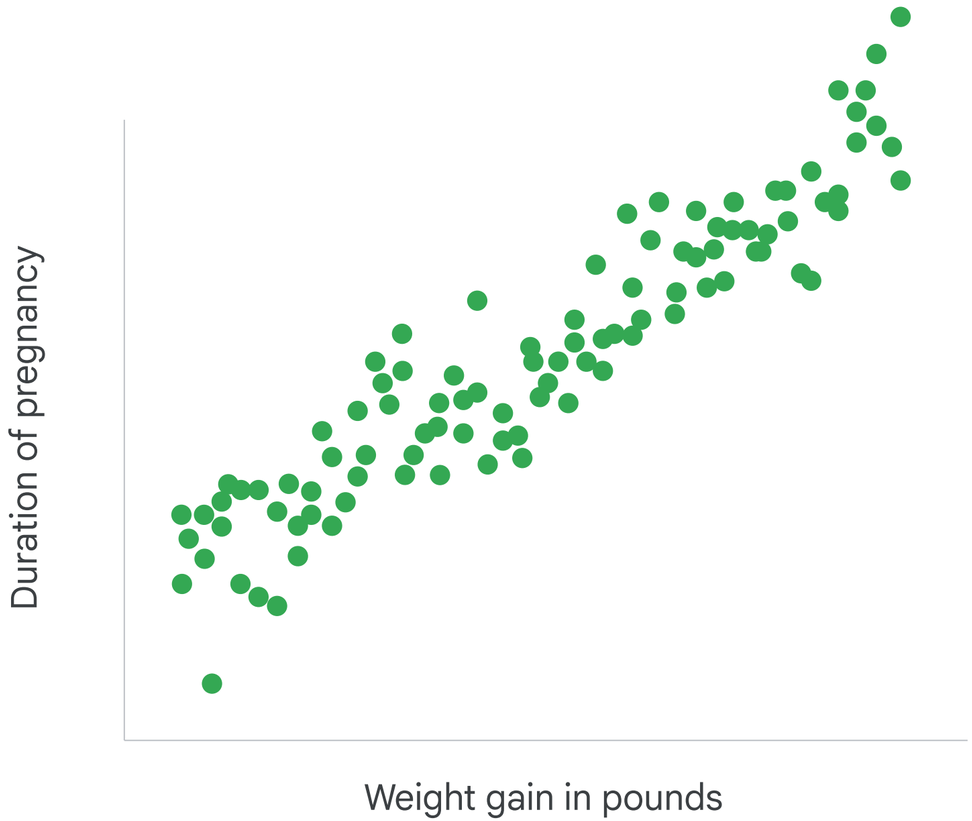
Vertex AI > Metadata



Model Generalization

Suppose we want to predict duration of pregnancy based on mother's weight gain in pounds

What is the error measure to optimize?



Model 1 is a linear model using linear regression

RMSE = 2.224

Model 2 has more free parameters

RMSE = 0

Which model is better?
How can you tell?

Does the model generalize to new data?

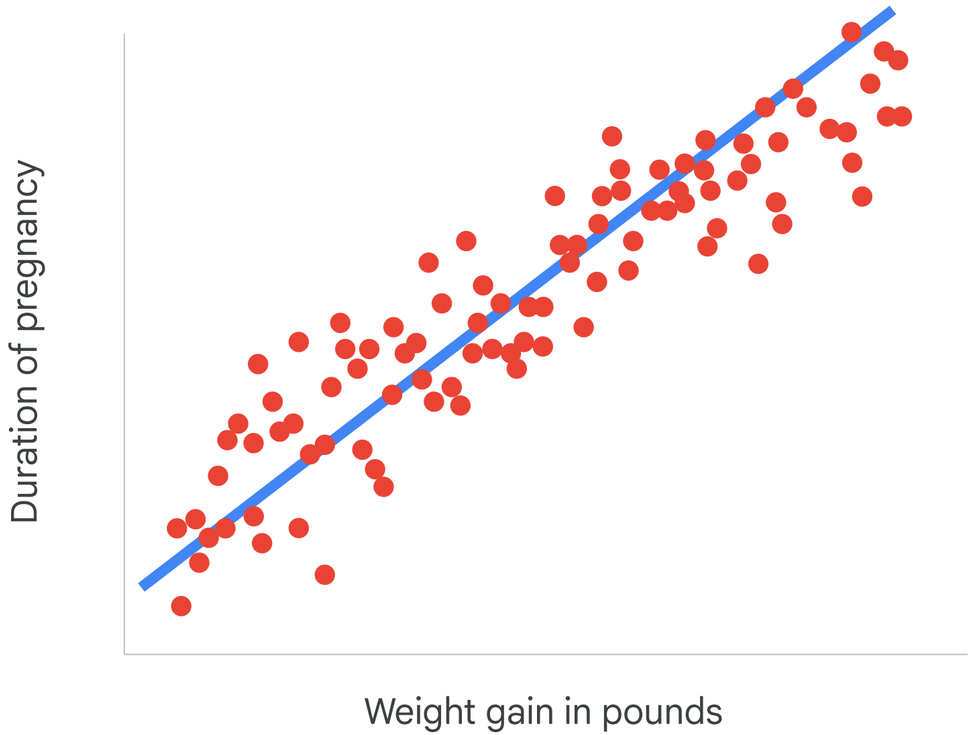
Need data that were not used in training.

Model 1 generalizes well

Old RMSE = 2.224
New RMSE = 2.198

Pretty similar = good

New data the model hasn't seen before.

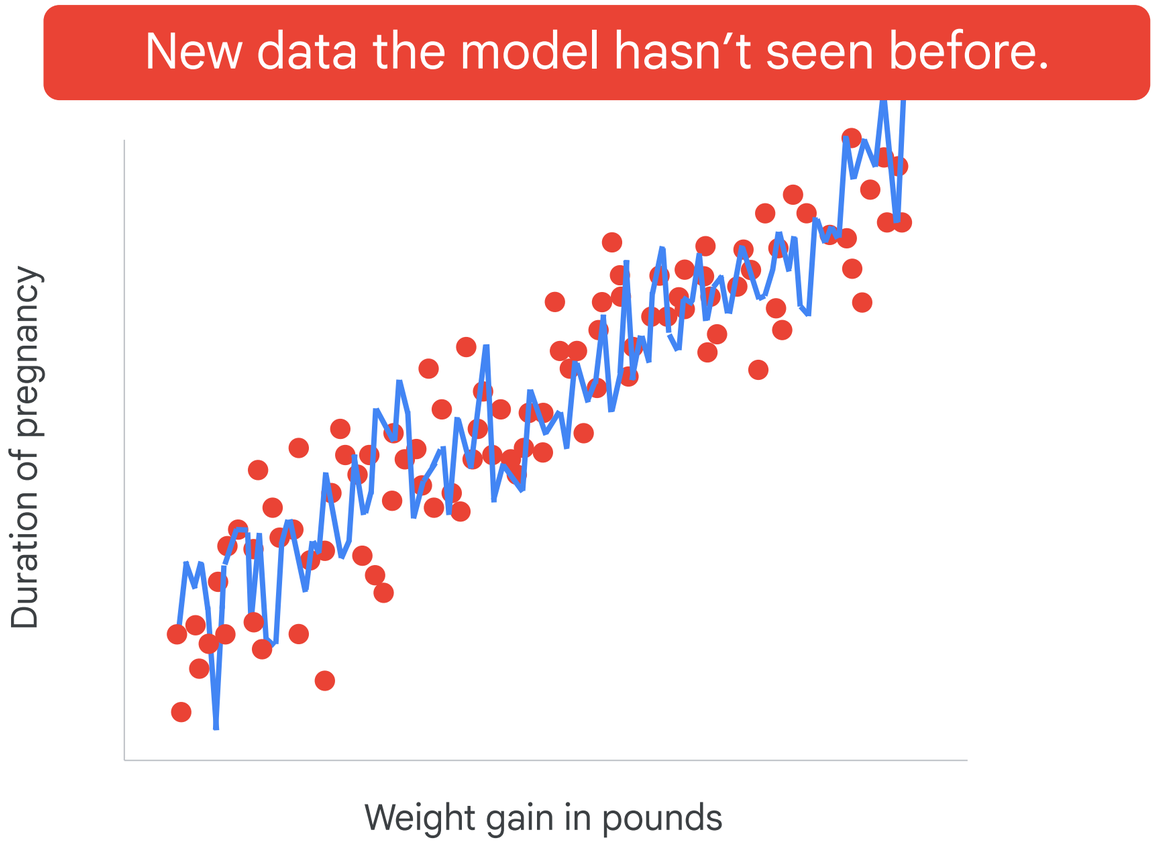


Model 2 does not generalize well

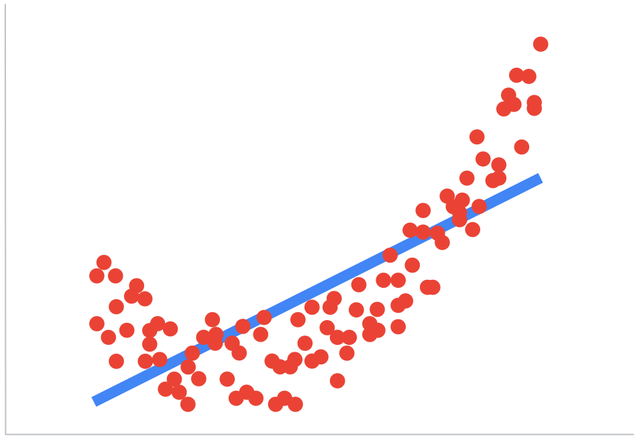
Old RMSE = 0

New RMSE = 3.2

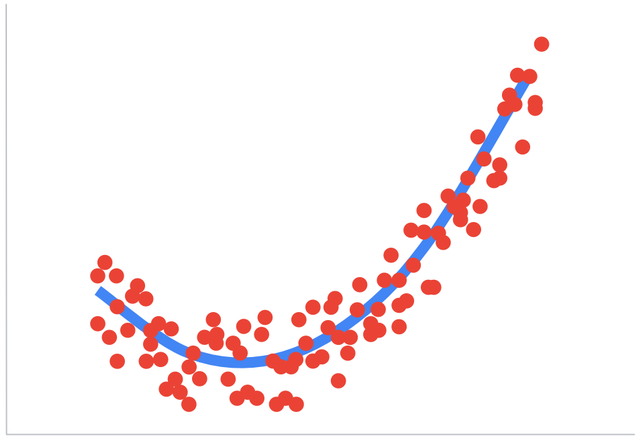
This is a red flag.



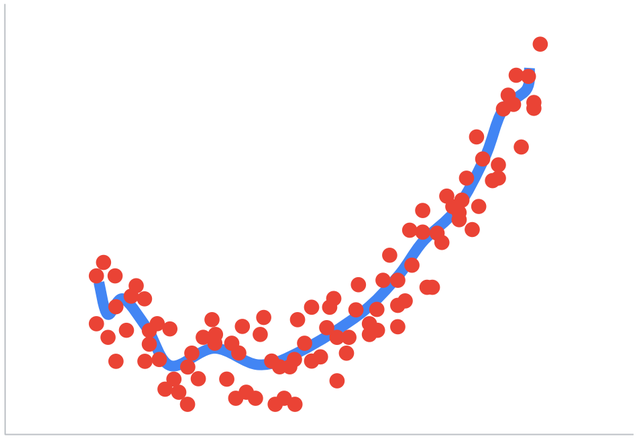
Beware of overfitting as you increase model complexity



Underfit

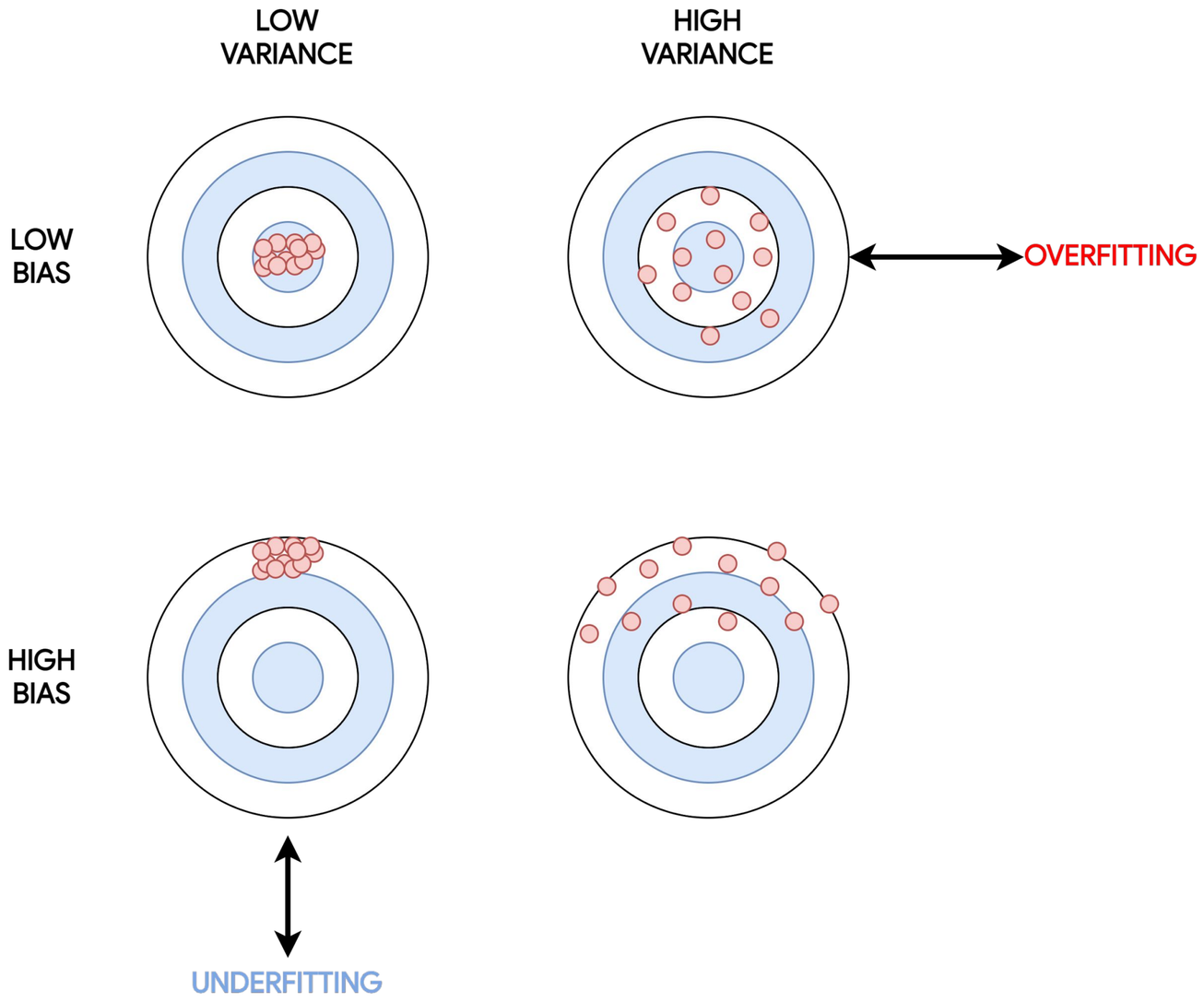


Fit

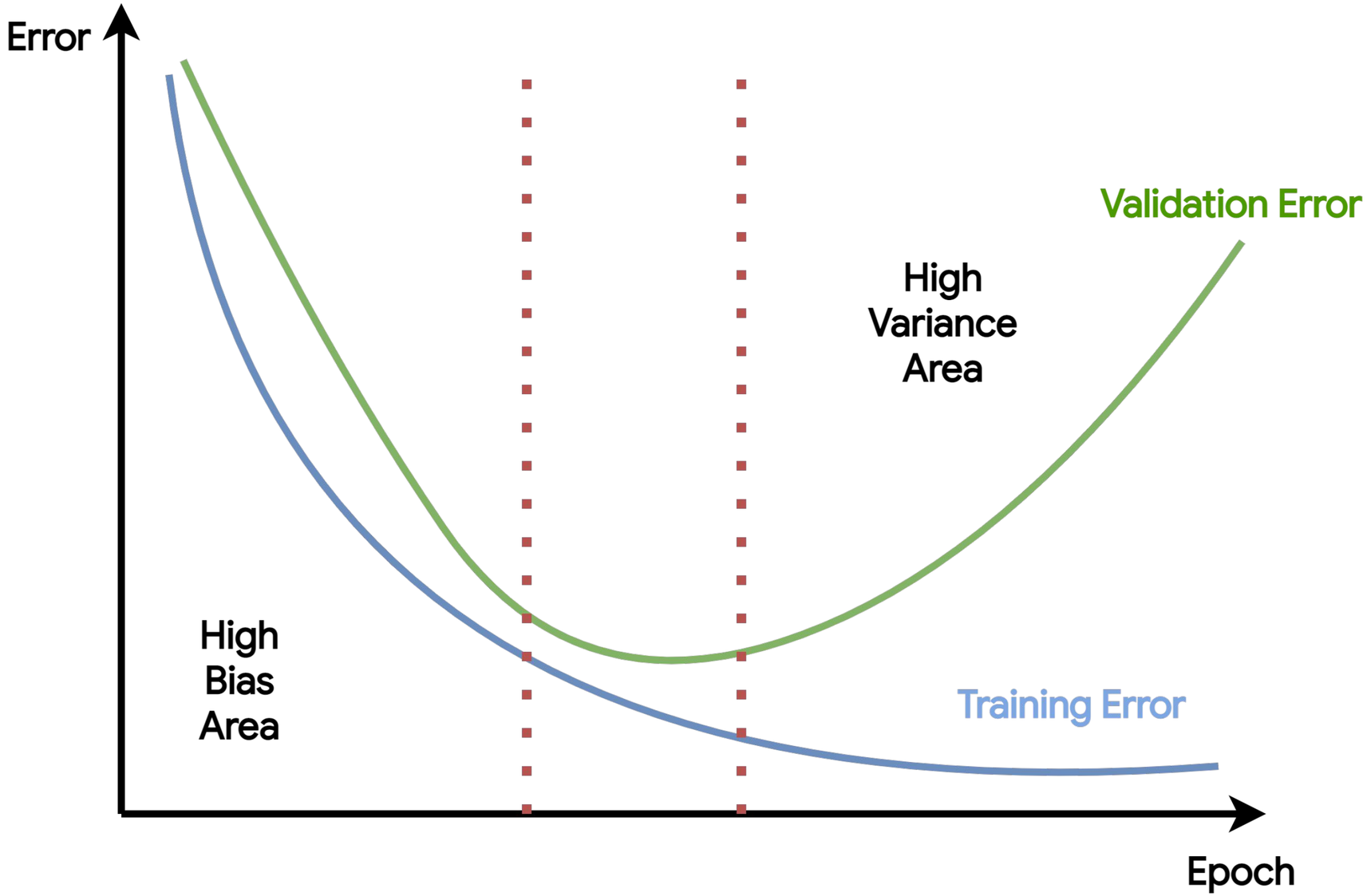


Overfit

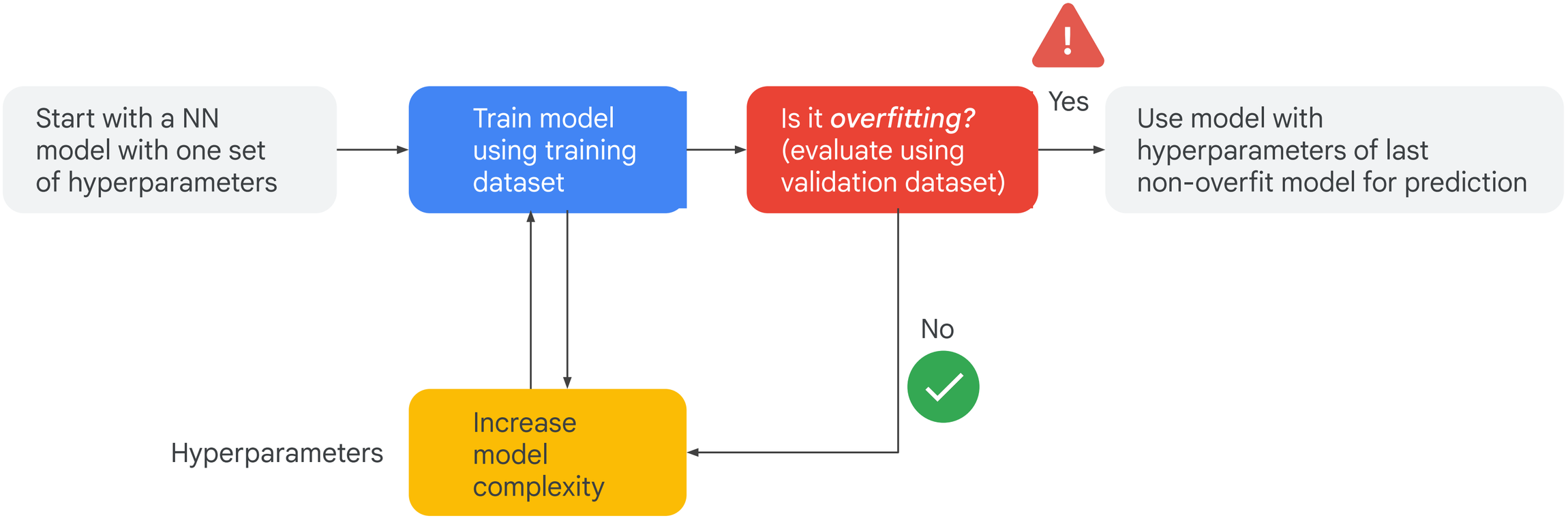
Bias VS Variance



Bias VS Variance



You can use the validation dataset to experiment with model complexity



How can we measure model complexity?

Regularization is a major field of ML research

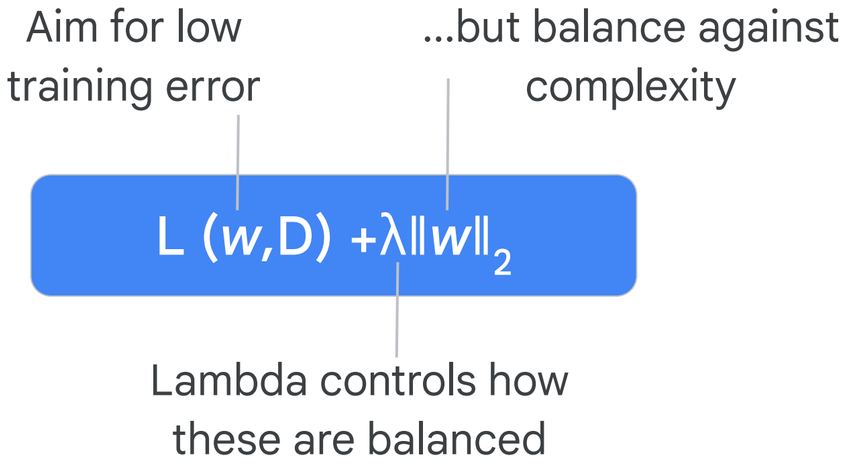
```
Early Stopping
Parameter Norm Penalties
        L1 regularization
        L2 regularization
        Max-norm regularization
Dataset Augmentation
Noise Robustness
Sparse Representations
. . .
```

We will look into these methods.

L1 regularization:
Creating a sparse model with some feature weights being exactly zero, effectively performing feature selection

L2 regularization:
prevent overfitting and distribute the weights more evenly among features

In L2 regularization, complexity of model is defined by the L2 norm of the weight vector



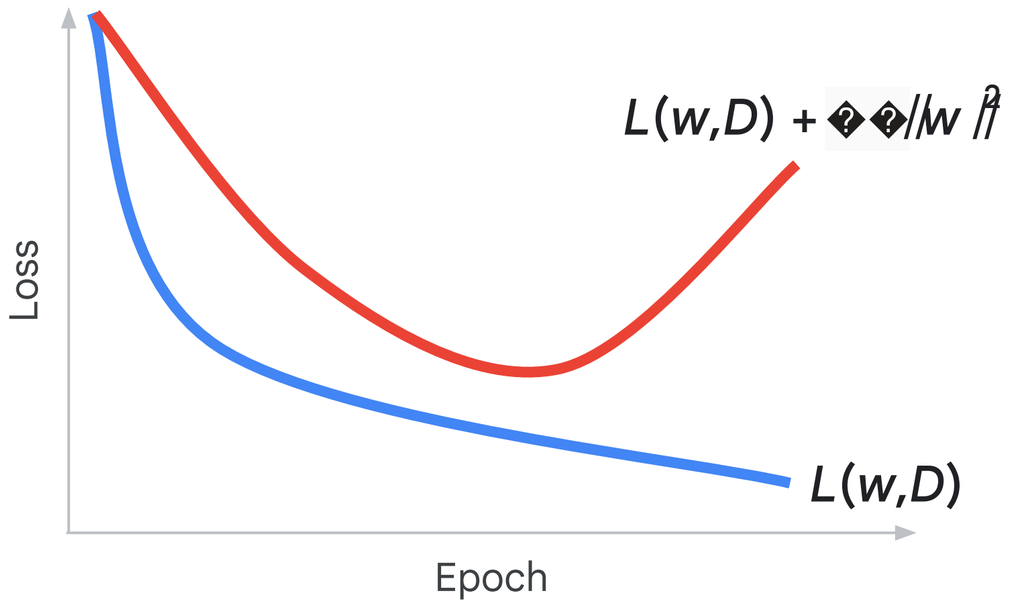
In L1 regularization, complexity of model is defined by the L1 norm of the weight vector

$$L(w,D) + \lambda \|w\|_1$$

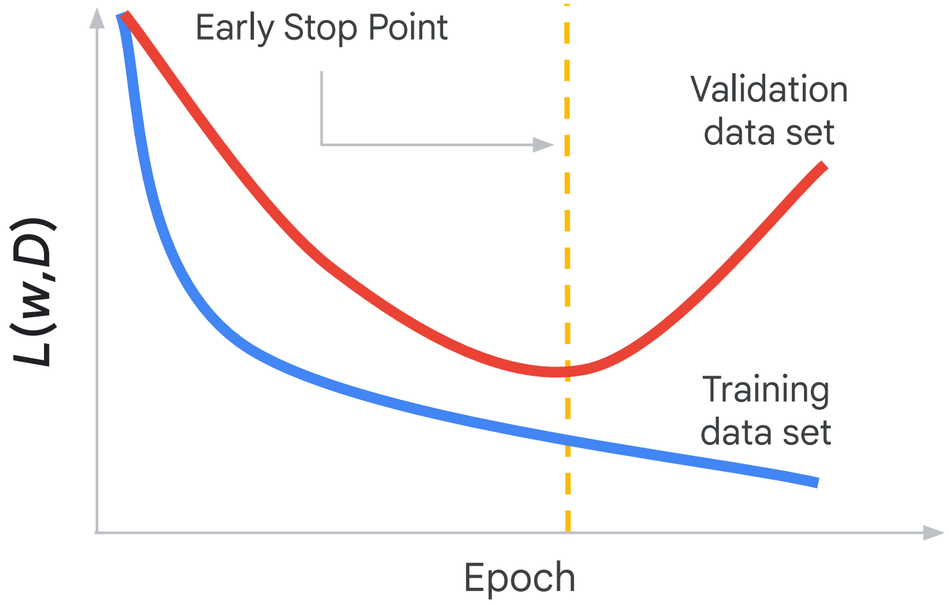
L1 regularization can be used as a feature selection mechanism.

Often we do both regularization and early stopping to counteract overfitting

Regularization

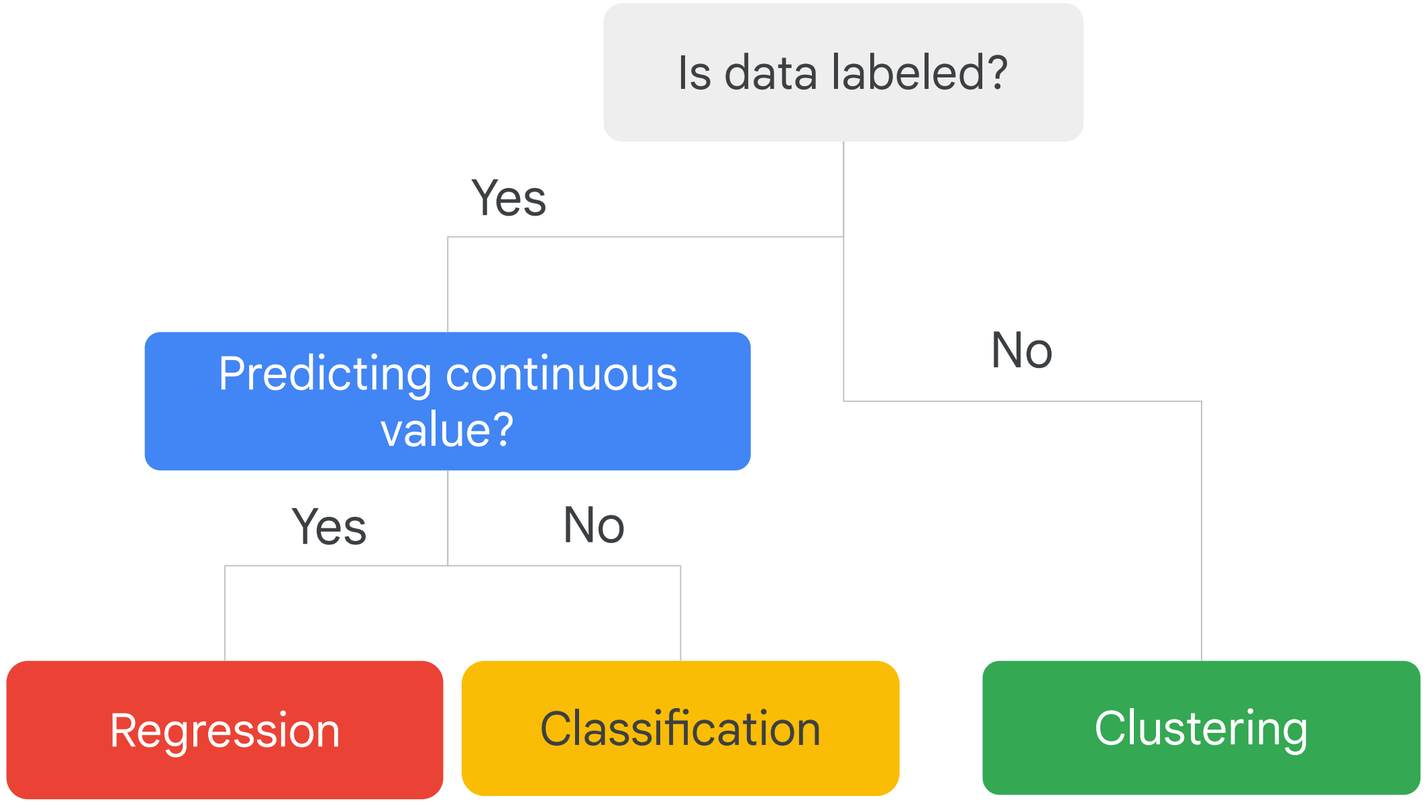


Early stopping



ML Models Overview

The type of ML problem depends on whether or not you have labeled data and what you are interested in predicting

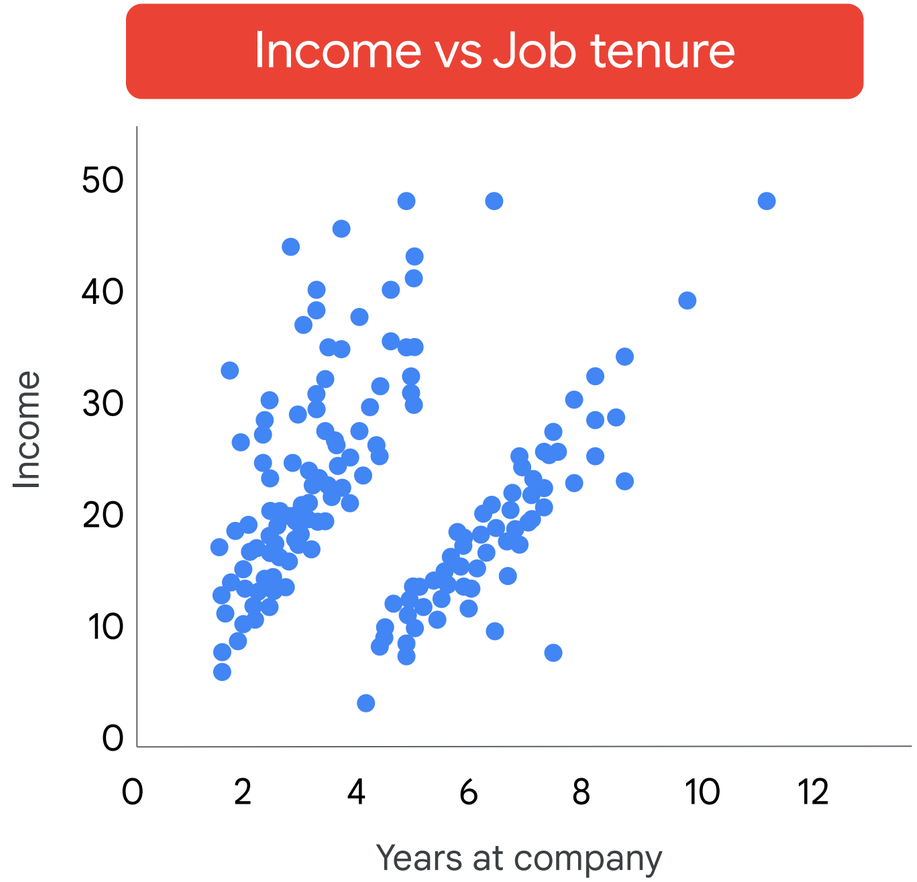


Unsupervised and supervised learning are the two types of ML algorithms

In unsupervised learning, data is not labeled.

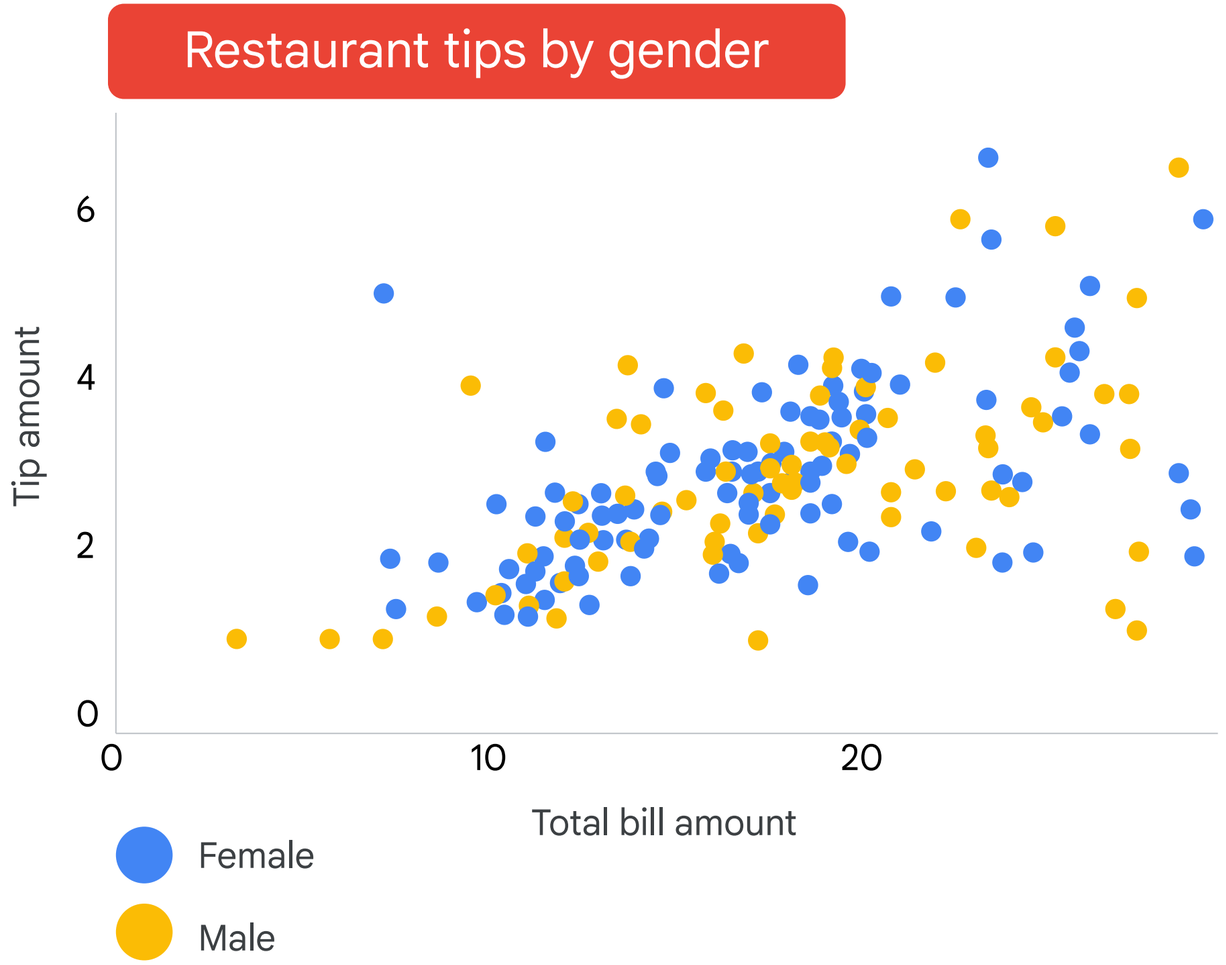
Example Model: Clustering

Is this employee on the “fast-track” or not?



Supervised learning implies the data is already labeled

In supervised learning we are learning from past examples to predict future values.



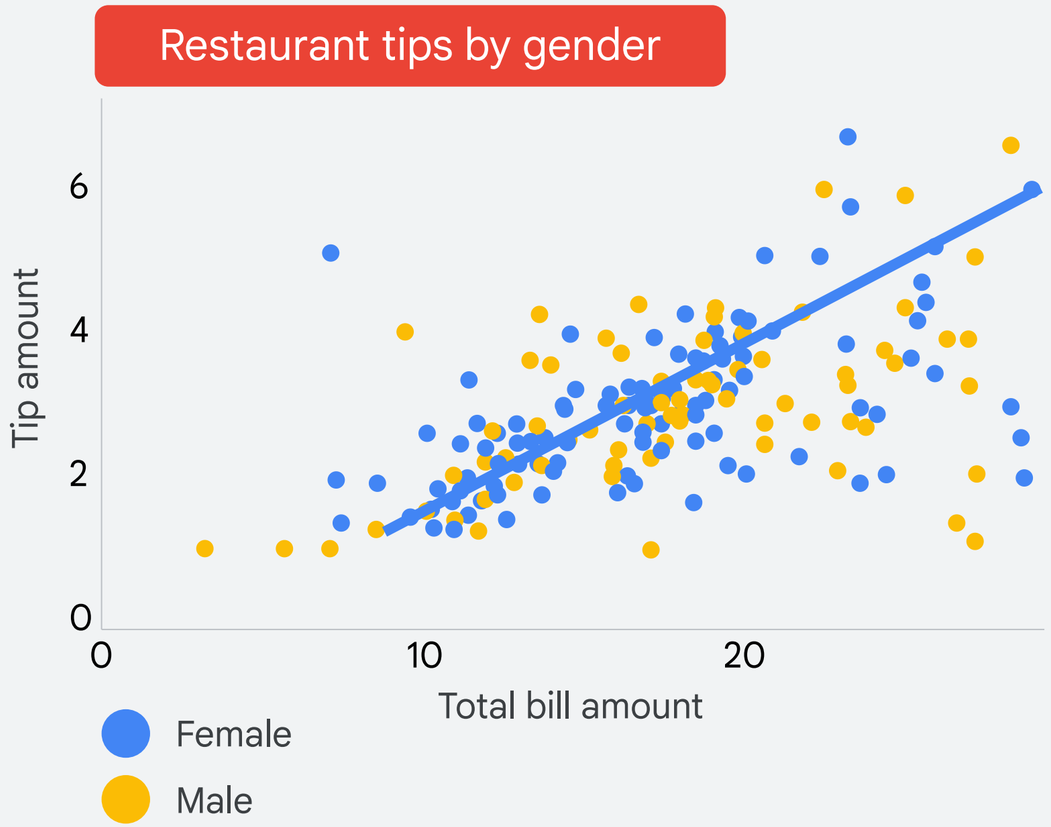
Supervised ML model types:

Regression and classification

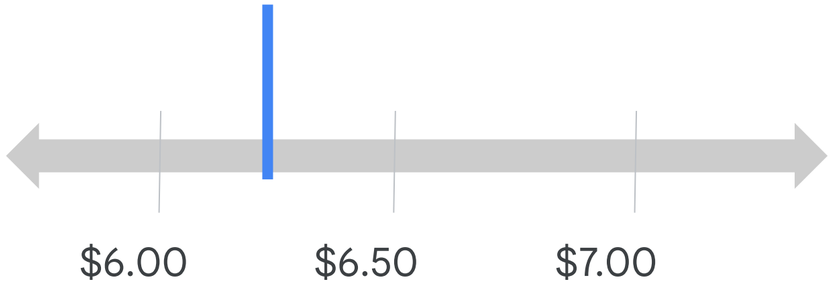
	total_bill	tip	sex	smoker	day	time
2	16.99	1.01	Female	No	Sun	Dinner
3	10.34	1.66	Male	No	Sun	Dinner
4	21.01	3.5	Male	No	Sun	Dinner
5	23.68	3.31	Male	No	Sun	Dinner
6	24.59	3.61	Female	No	Sun	Dinner
7	25.29	4.71	Male	No	Sun	Dinner
8	8.77	2	Male	No	Sun	Dinner
9	26.88	3.12	Male	No	Sun	Dinner

Option 1
Regression model
Predict the tip amount

Option 2
Classification model
Predict the sex of the customer



Use regression for predicting continuous label values

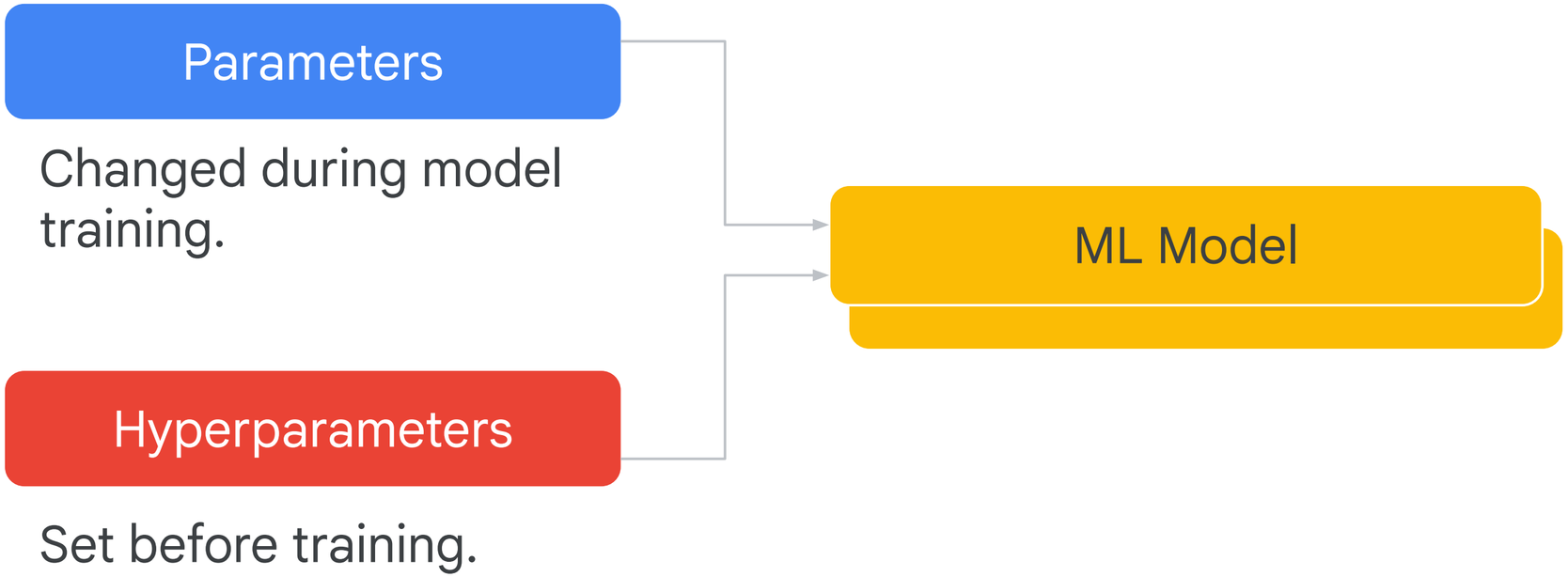


Regression model (linear) predicts tip amount of $6.25.

Use classification for predicting categorical label values

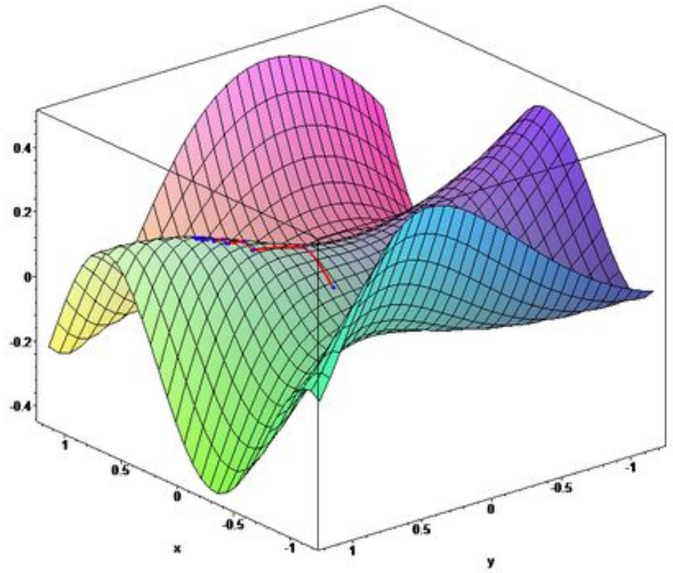
Classification model (binary) predicts category of female.

ML models are mathematical functions with parameters and hyperparameters



Model Optimization

Stochastic Gradient Descent



Optimizers in Tensor Flow… Stochastic Gradient Descent (SGD)

tf.keras.optimizers.SGD

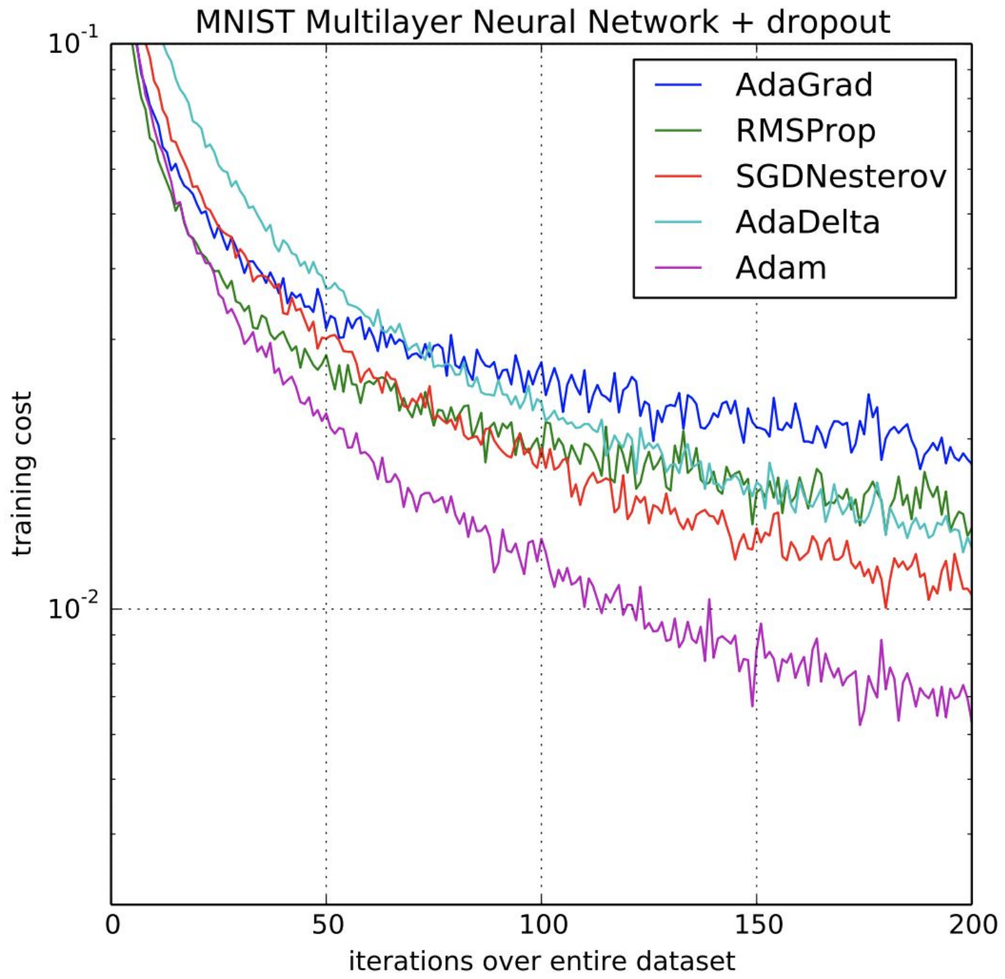
- Implements Gradient Descent with Momentum
- Parameters can be
  Learning Rate => w(i+1) = w(i) – LR*G
  Momentum => velocity (i+1) = velocity(i)*momentum – LR*G
  w(i+1) = w(i) + velocity(i+1)
- Only one parameter is updated at each time.

What if we want to update multiple parameter w1, w2, …
at each time

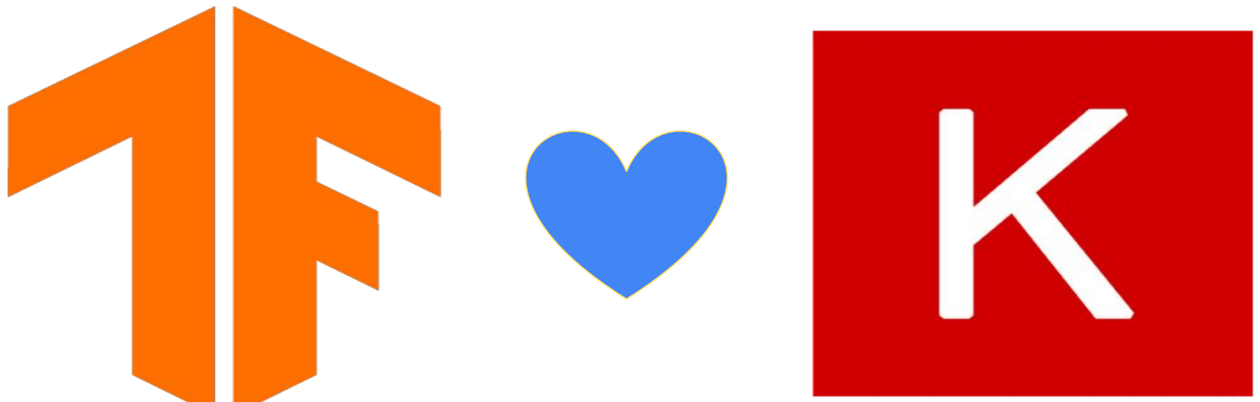
Optimizers in Tensor Flow… Adaptives Gradient Optimizers

tf.keras.optimizers.Adam

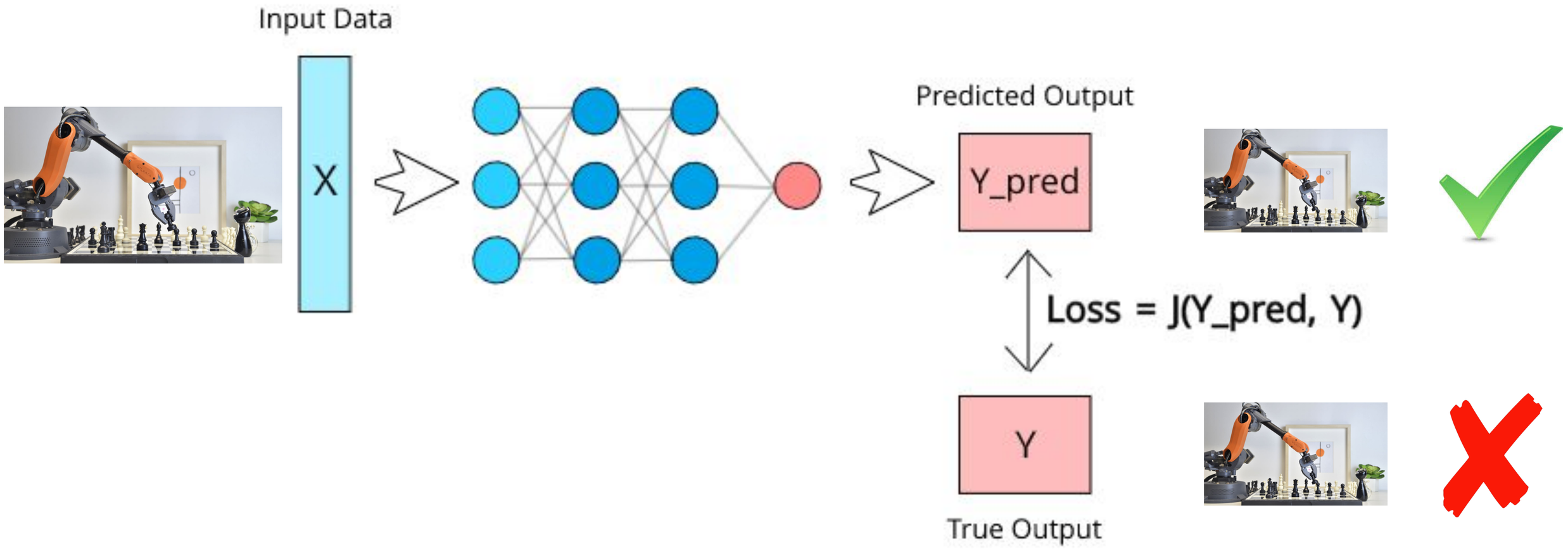
- Adam adds momentum to Adadelta thus improving convergence speed for every parameters
- Adam is commonly used optimizer for training process of large dataset because its reduced training cost



Loss functions in TensorFlow



Loss functions in TensorFlow

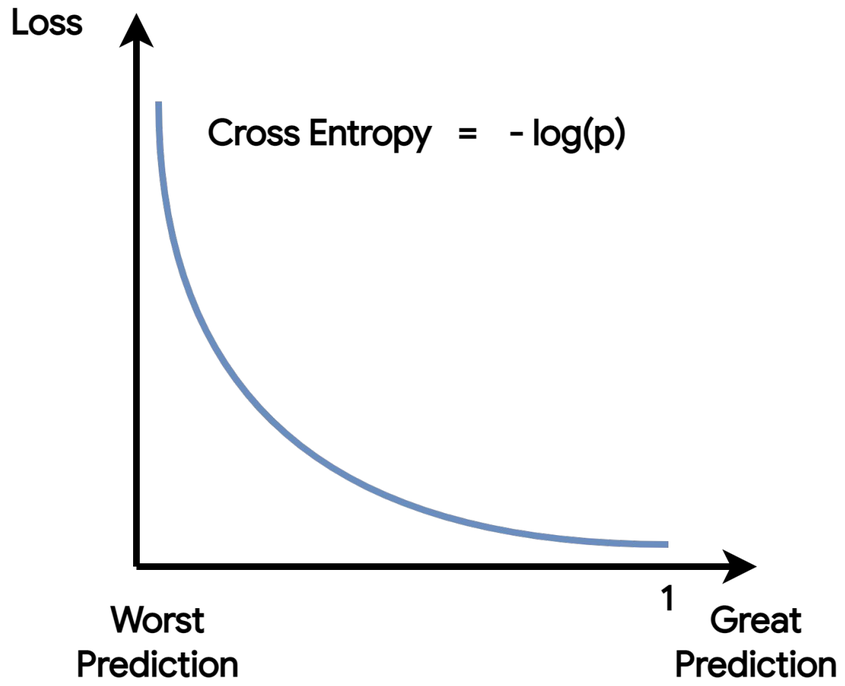


Loss functions in Tensor Flow… Categorical (1)

tf.keras.losses.BinaryCrossentropy

- Compute cross-entropy loss for binary (0,1) classification problems

```
# Example 1: (batch_size = 1, number of samples = 4)
y_true = [0, 1, 0, 0]
y_pred = [-18.6, 0.51, 2.94, -12.8]
bce = tf.keras.losses.BinaryCrossentropy(from_logits=True)
bce(y_true, y_pred).numpy()
0.865
```



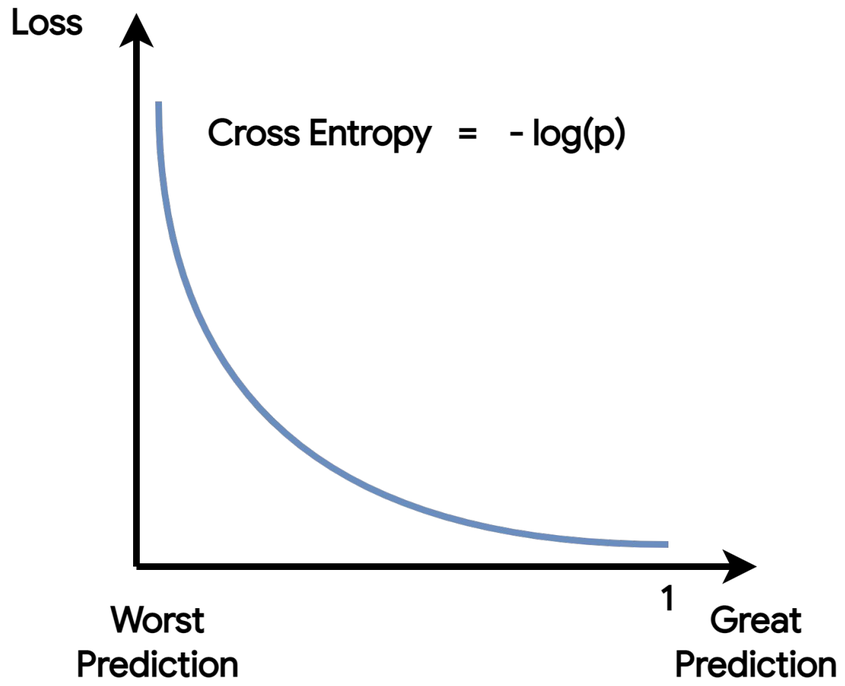
Loss functions in Tensor Flow… Categorical (2)

tf.keras.losses.CategoricalCrossentropy

- Compute cross-entropy loss when there are with 2 or more label classes.

- Labels need to be one-hot encoded otherwise use SparseCategoricalCrossentropy

```
y_true = [[0, 1, 0], [0, 0, 1]]
y_pred = [[0.05, 0.95, 0], [0.1, 0.8, 0.1]]
cce = tf.keras.losses.CategoricalCrossentropy()
cce(y_true, y_pred).numpy()
1.117
```

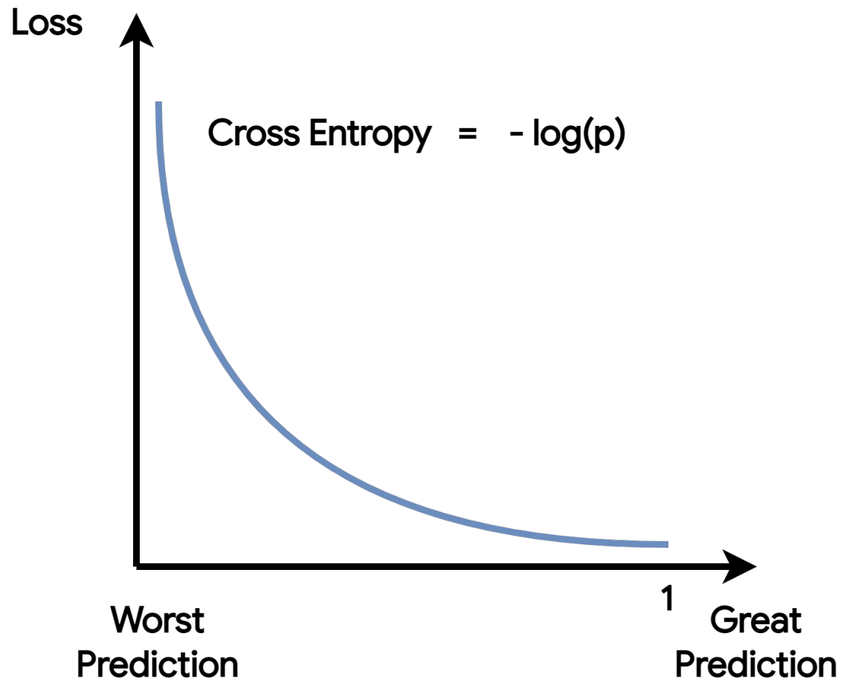


Loss functions in Tensor Flow… Categorical (3)

tf.keras.losses.SparseCategoricalCrossentropy

- Similar to the CategoricalCrossentropy but labels are expected to be interger

```
y_true = [1, 2]
y_pred = [[0.05, 0.95, 0], [0.1, 0.8, 0.1]]
scce = tf.keras.losses.SparseCategoricalCrossentropy()
scce(y_true, y_pred).numpy()
1.117
```



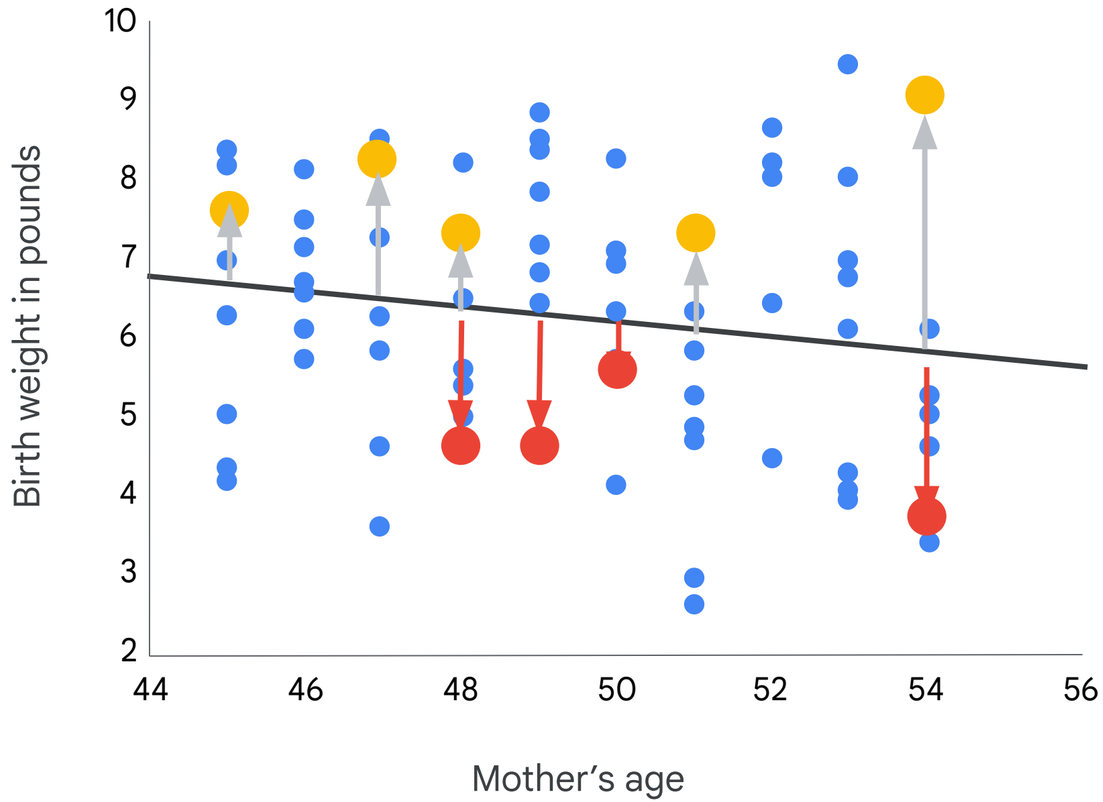
Compose a loss function by calculating errors

Each error makes sense. How about all the errors added together?

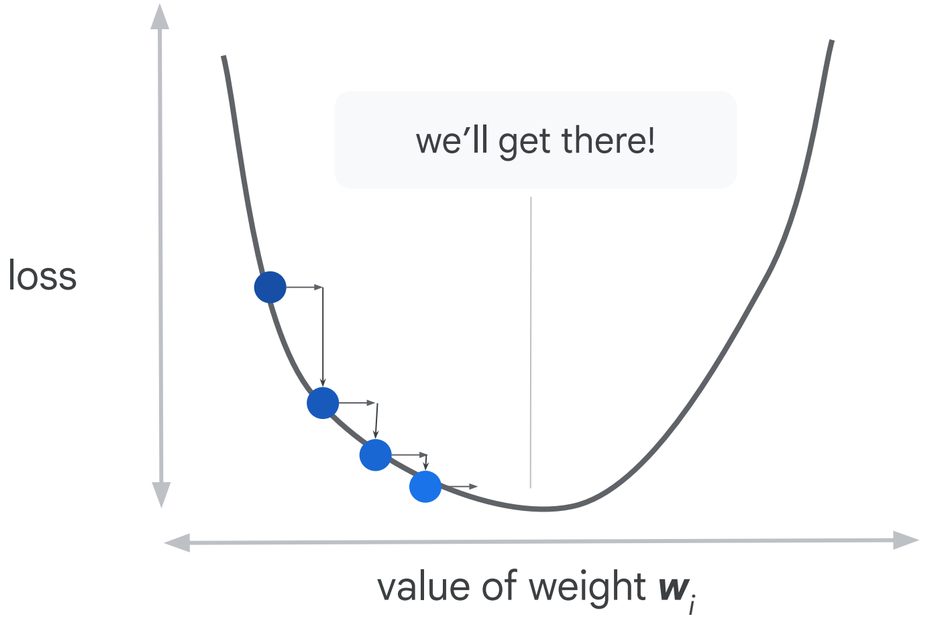
Error = actual (true) - predicted value

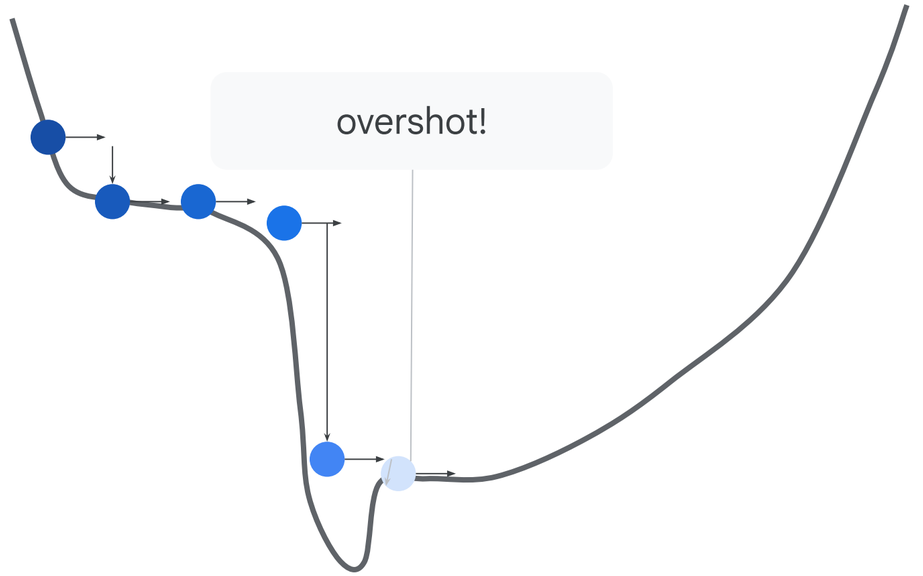
Compute the errors:

+0.70
+1.10
+0.65
-1.20
-1.15
+1.10
+3.09
-2.10



A correct and constant step size (Learning Rate) can be difficult to find



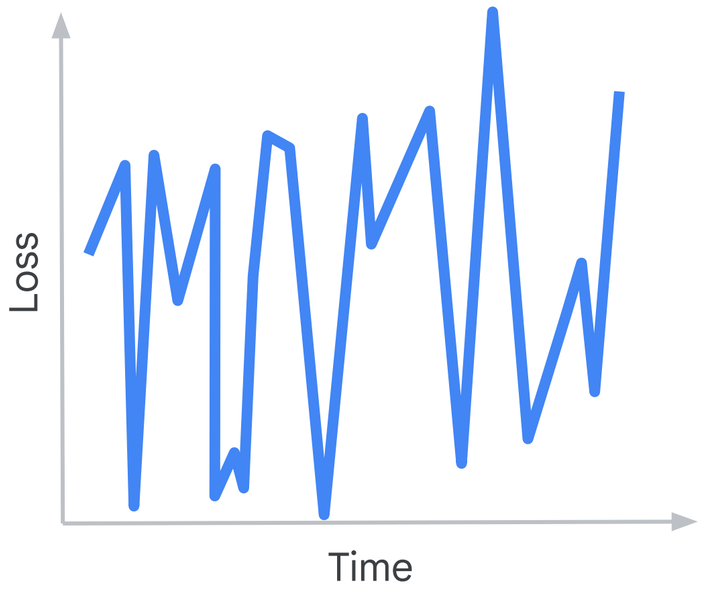


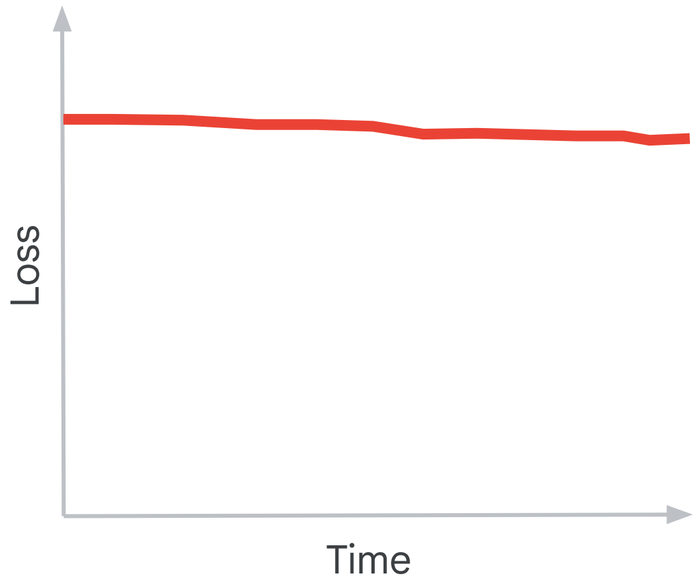
Step size or "learning rate" is a hyperparameter which is set before training.

One size does not fit all models.

A typical loss curve

Troubleshooting a loss curve





Adding a scaling hyperparameter

```
while loss is > Epsilon:
     derivative = computeDerivative()
   for i in range(self.params):
     self.params[i] = //
                self.params[i] //
                     - learning_rate //
                     * derivative[i]
   loss = computeLoss()
```

Checking loss with reduced frequency

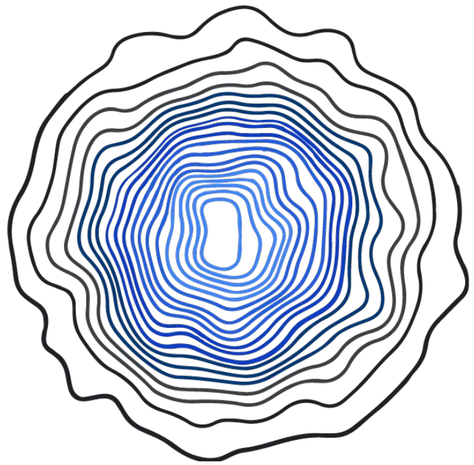
```
while loss is > Epsilon:
     derivative = computeDerivative()
   for i in range(self.params):
     self.params[i] = //
                self.params[i] //
                     - learning_rate //
                     * derivative[i]
   loss = computeLoss()
```

Popular implementations for readyToUpdateLoss():

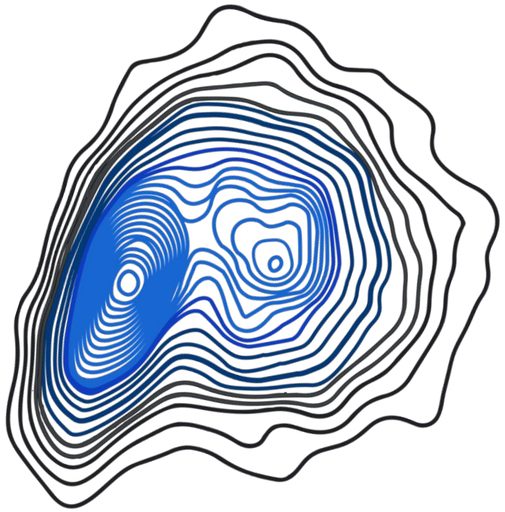
- Time-based (e.g., every hour)
- Step-based (e.g., every 1000 steps)

Problem: My model changes every time I retrain it

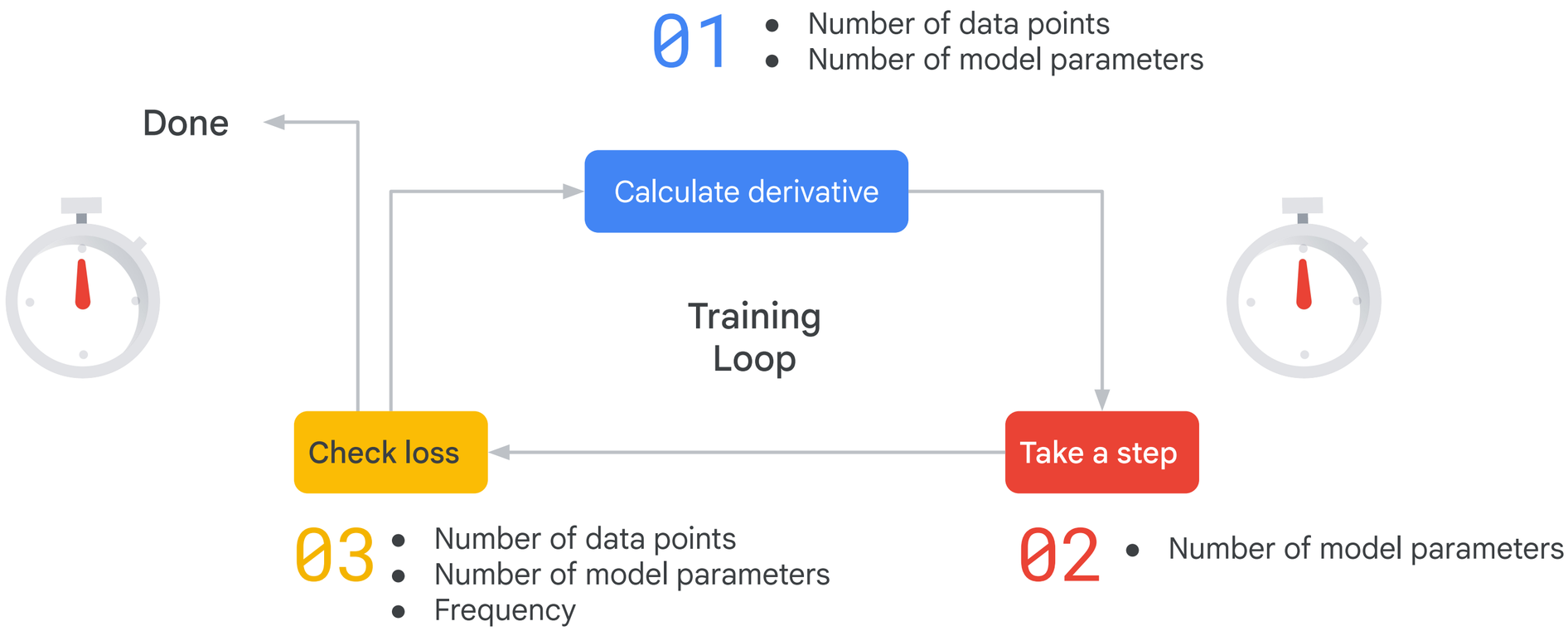
Loss surface with a global minimum



Loss surface with more than one minima



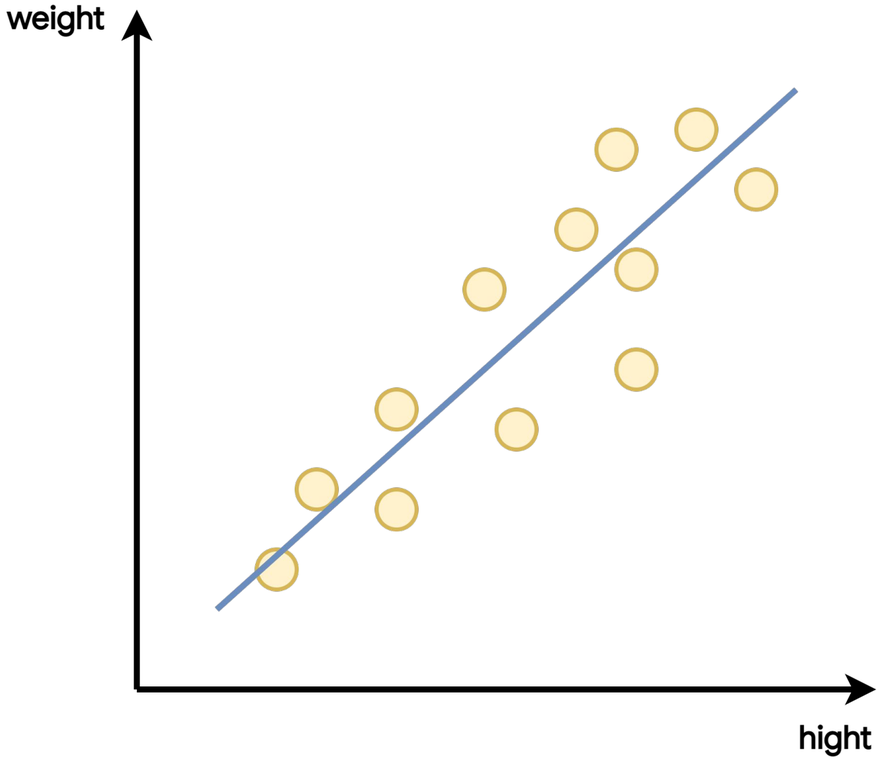
Problem: Model training is still too slow



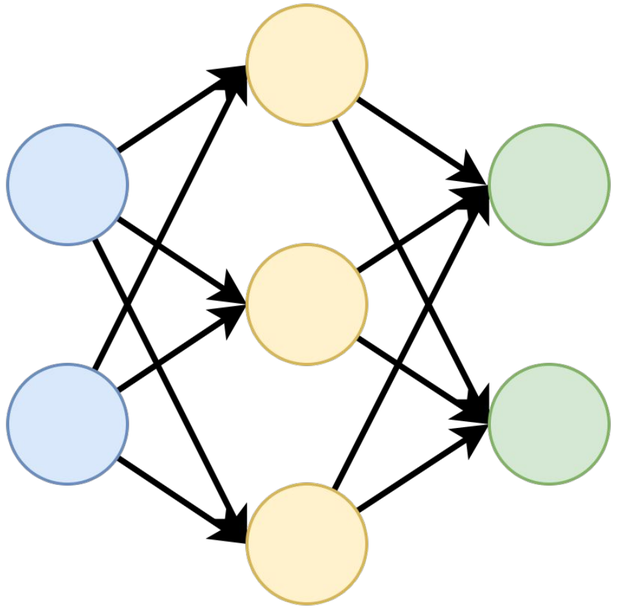
Model Metrics

RMSE: Regression problem is easy

CNN:



< = >



The lower RMSE wins (pss... careful not overfitting)

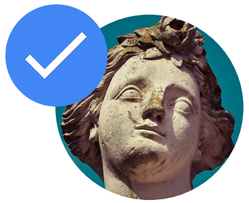
A confusion matrix leads to evaluation metric insights (for a classification model)

Model predictions

Labels	Positive	Negative
Positive	**True positives (TP)** Label says something exists. Model predicts it. 	
Negative		

The confusion matrix leads to evaluation metric insights

Model predictions

Labels	Positive	Negative
Positive	**True positives (TP)** Label = something exists. Model predicts it. 	**False negatives (FN)** Type II error Label = something exists. Model doesn't predict it. 
Negative	**False positives (FP)** Type I error Something doesn't exist. Model predicts it. 	**True negatives (TN)** Something doesn't exist. Model doesn't predict it. 

False positives and false negatives errors occur when predictions and labels disagree

Model predictions

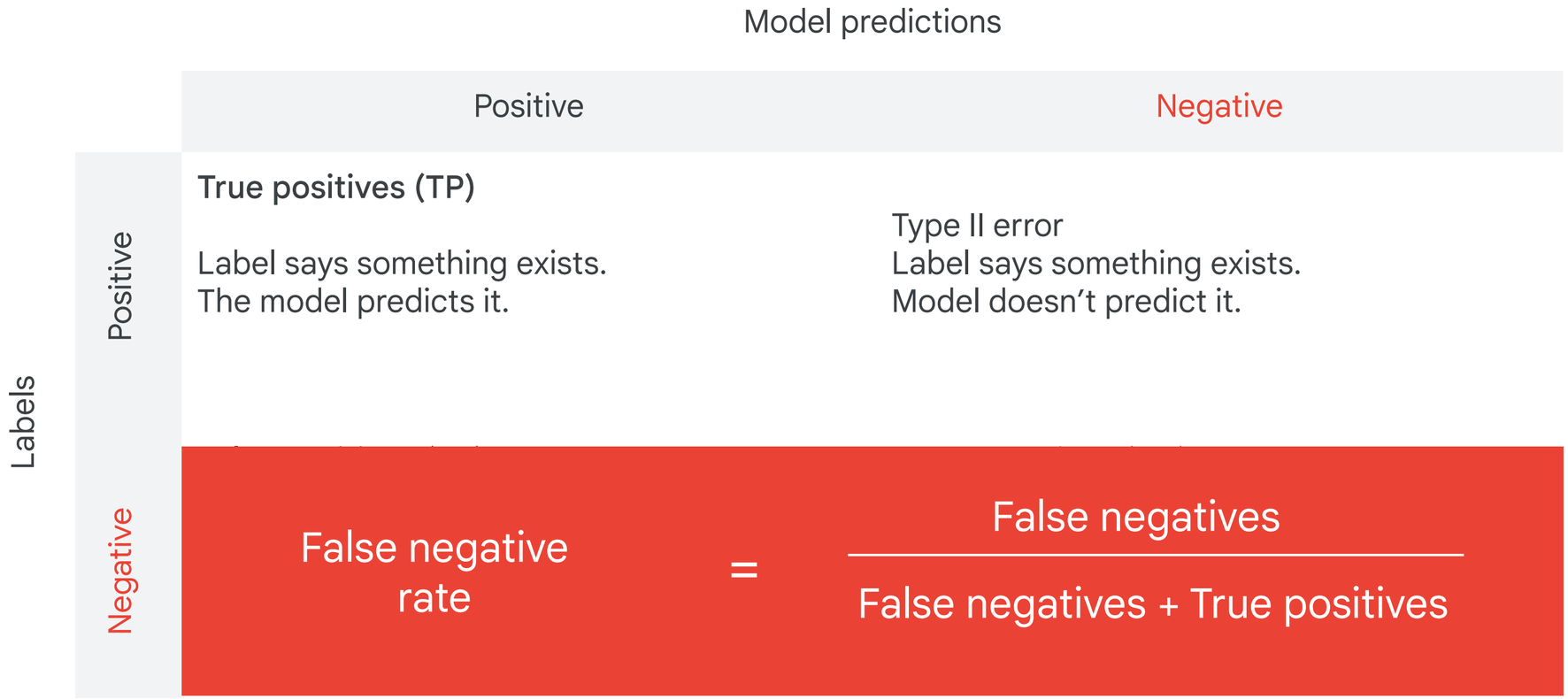
Labels	Positive	Negative
Positive	True positives (TP) 	False negatives (FN) Type II error **Model says: no** 
Negative	False positives (FP) Type I error **Model says: yes** 	True negatives (TN) 

Evaluation metrics can help highlight areas where machine learning could be more inclusive

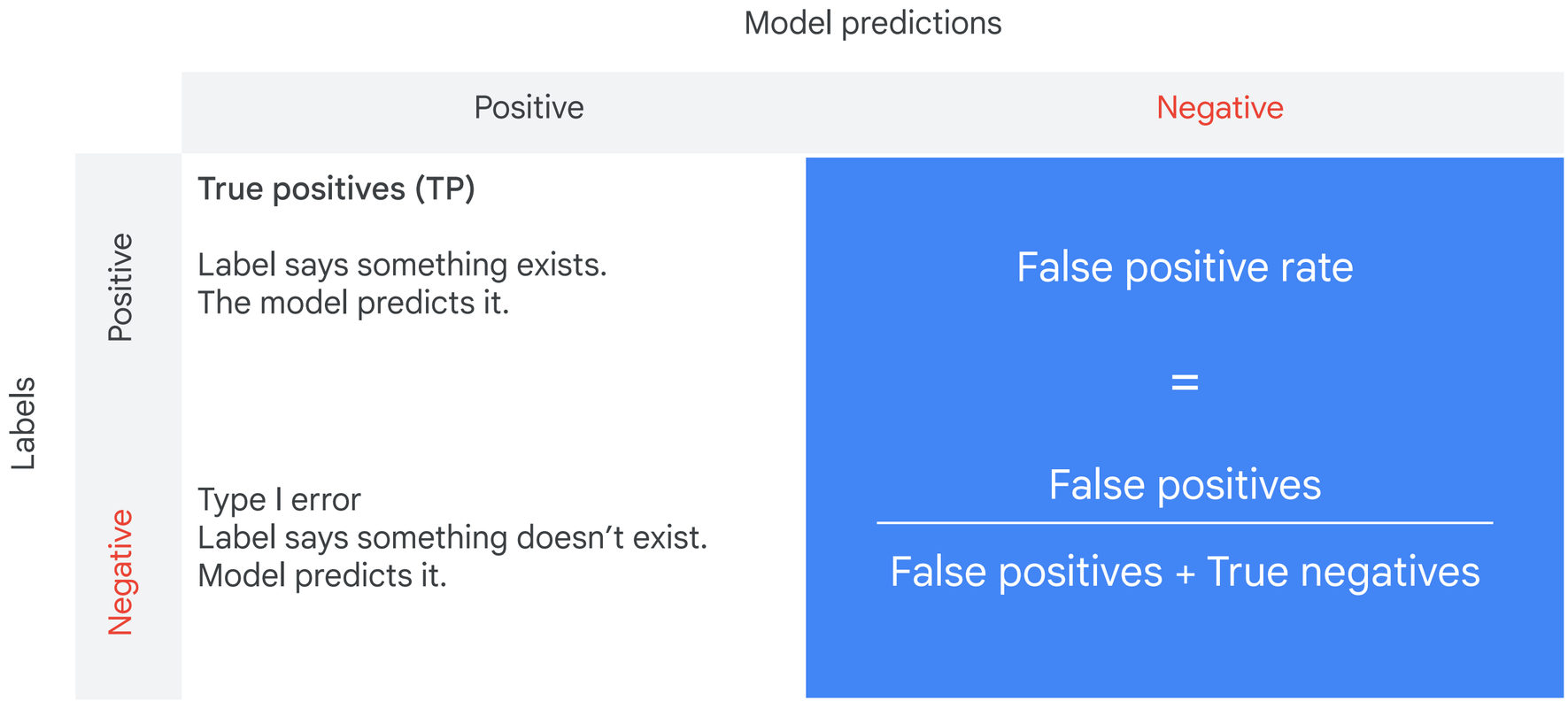
Model predictions

Labels	Positive	Negative
Positive	**True positives (TP)** Label says something exists. The model predicts it.	**False negatives (FN)** Type II error Label says something exists. Model doesn't predict it. **Model says: no**
Negative	**False positives (FP)** Type I error Label says something doesn't exist. Model predicts it. **Model says: yes**	**True negatives (TN)** Label says something doesn't exist. Model doesn't predict it.

False negative rate is the fraction of true faces that are not detected by the ML system



False positive rate is the fraction of the faces that the ML model detects that are not really faces



Classification evaluation metrics: Precision

Precision attempts to answer the following question:

What proportion of positive identifications was actually correct?

Precision is defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Confidence threshold 0.5

All labels	
PR AUC	0.696
ROC AUC	0.81
Log loss	0.815
F1 score	0.5686126
Precision	60.4%
Recall	53.7%
Created	Sep 29, 2021 12:52:20 AM

Classification evaluation metrics: Recall

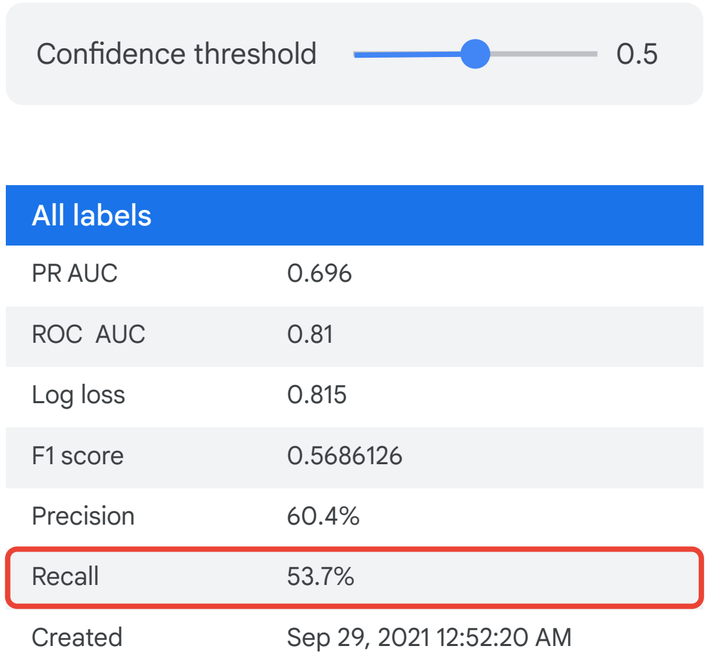
Recall attempts to answer the following question:

What proportion of actual positives was identified correctly?

Mathematically, recall is defined as follows:

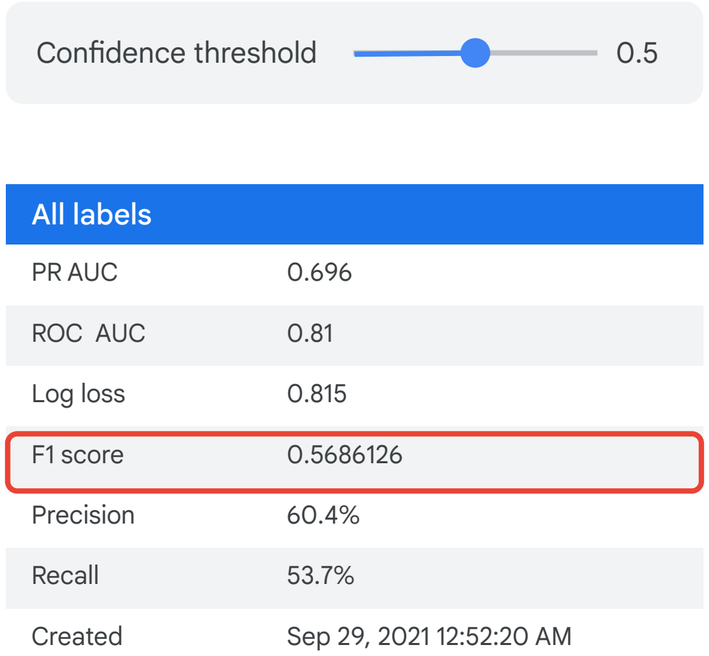
$$\text{Recall} = \frac{TP}{TP + FN}$$

Note: A model that produces no false negative has a recall of 1.0



F1 score

- Precision answers the question
  - Out of the equipment classified “will fail,” what fraction was correct?
- Recall answers the question
  - Out of the equipment that actually failed, what fraction did the classifier pick up?
- F1 score is the harmonic mean of precision and recall.



Sometimes, false positives are better than false negatives

Privacy in images

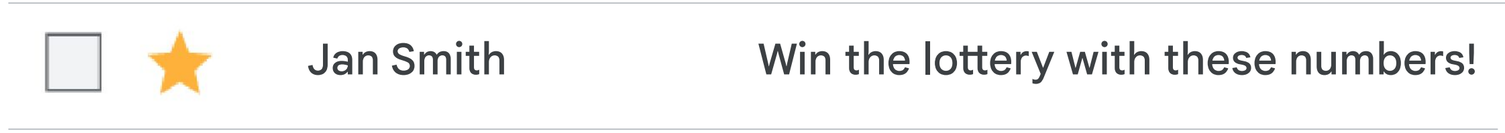
False positive

False negative

Sometimes, false negatives are better than false positives

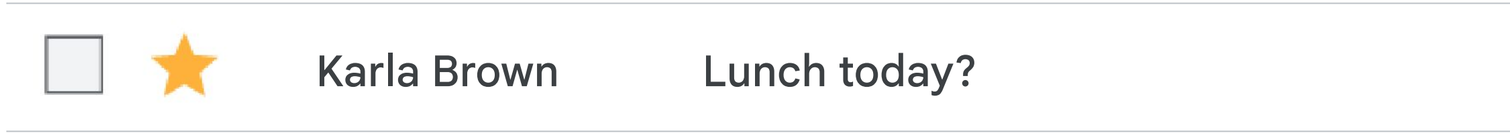
False negative:

E-mail that is SPAM is not caught, so you see it in your inbox.

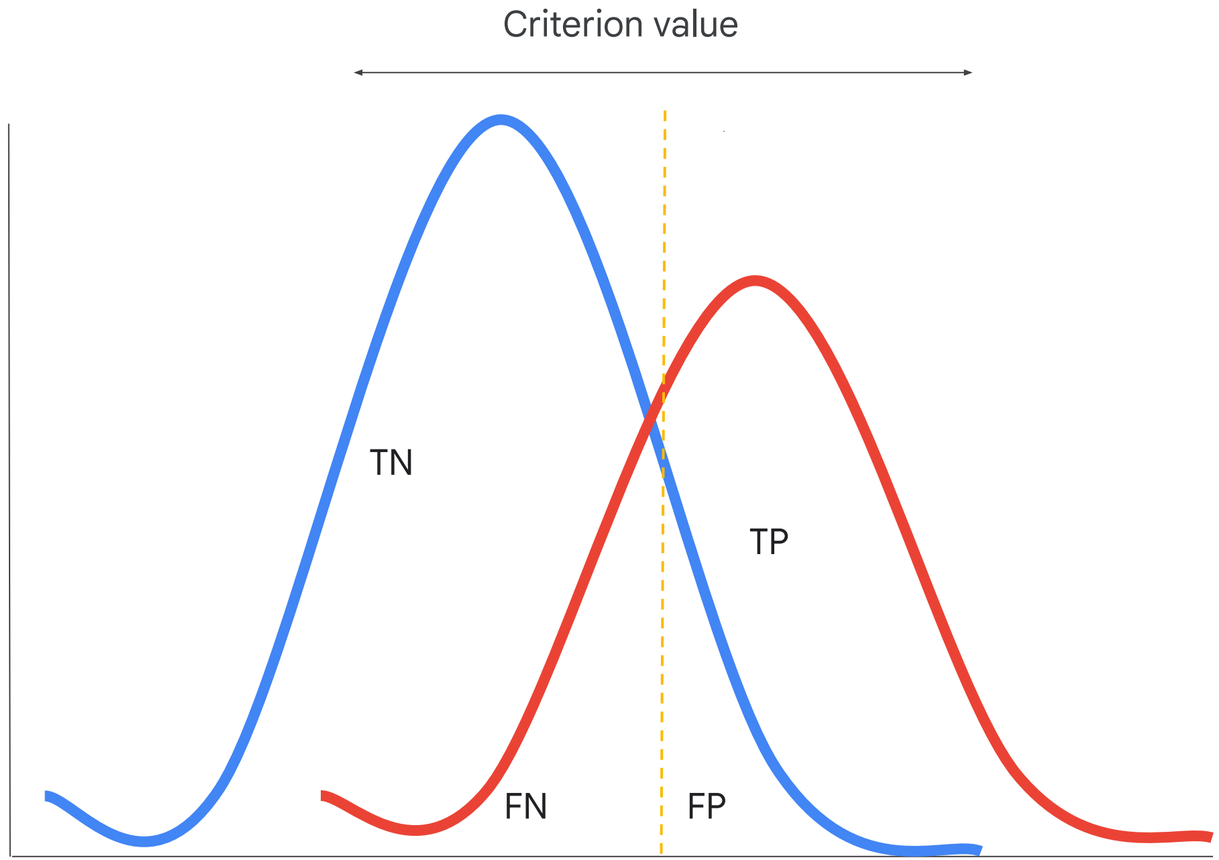


False positive:

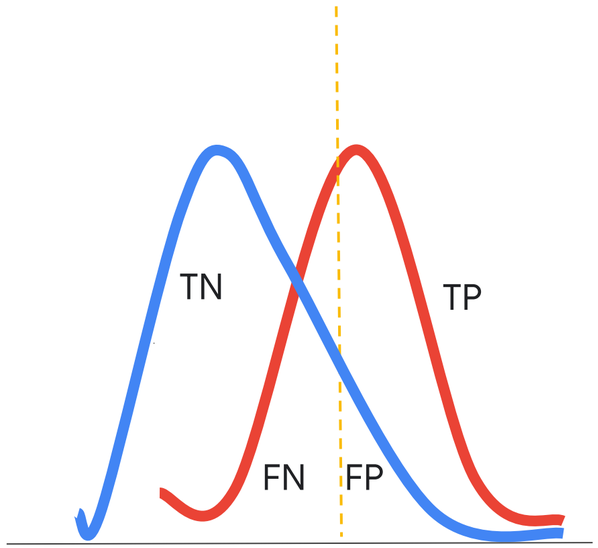
E-mail flagged as SPAM is removed from your inbox.



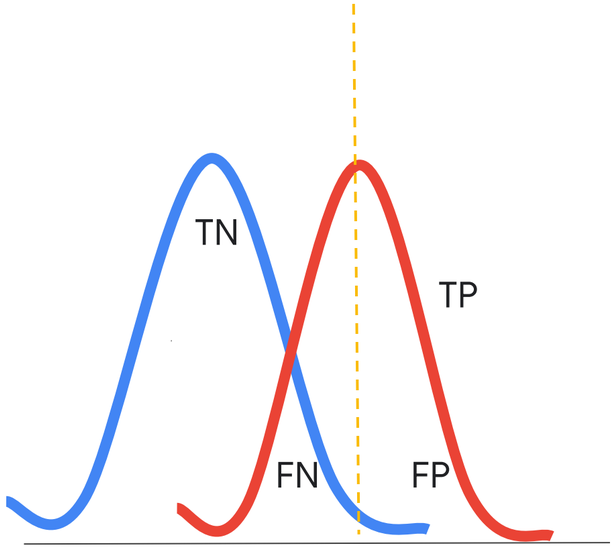
Find the threshold that brings the precision or recall to acceptable values



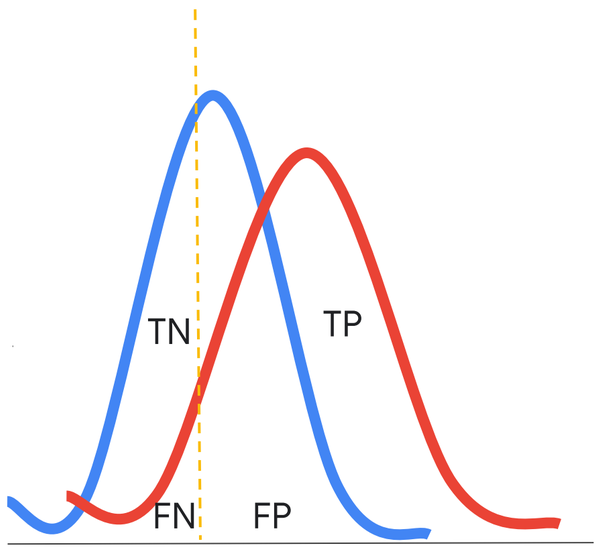
Check the precision/recall you obtain with that threshold in each of your subgroups



Sub-group 1

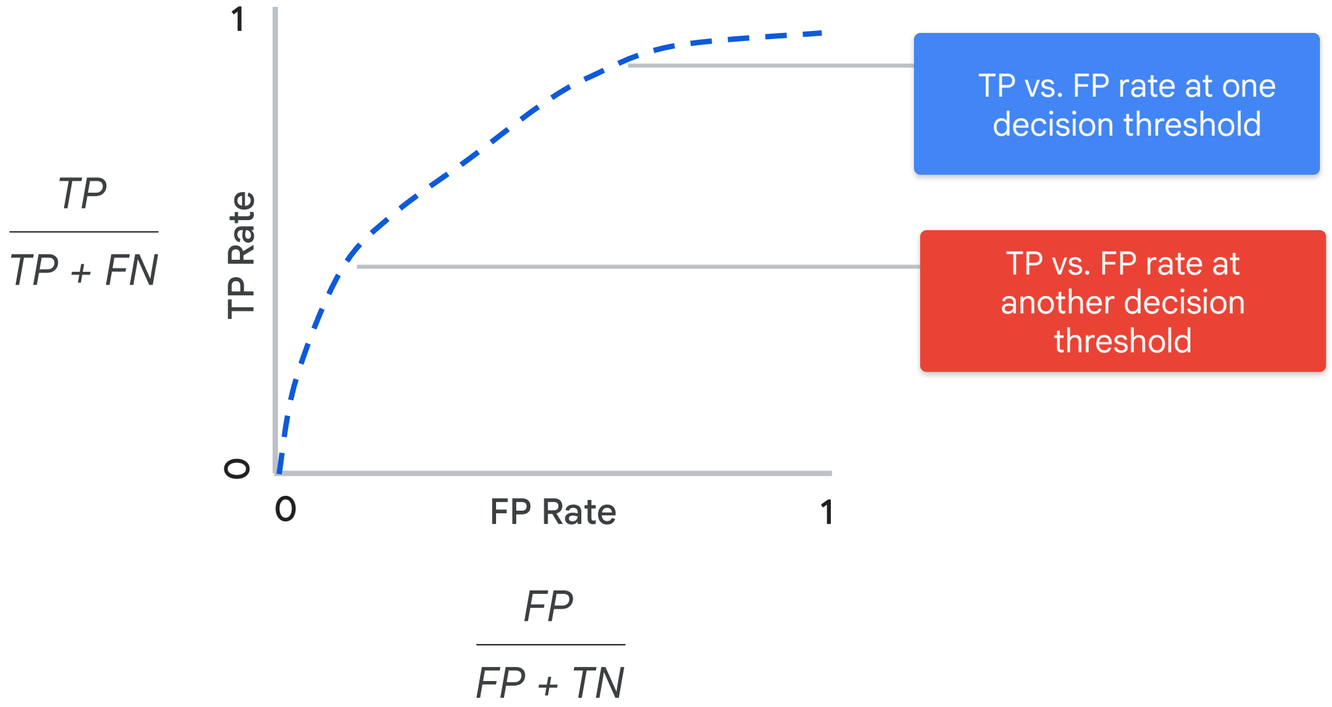


Sub-group 2

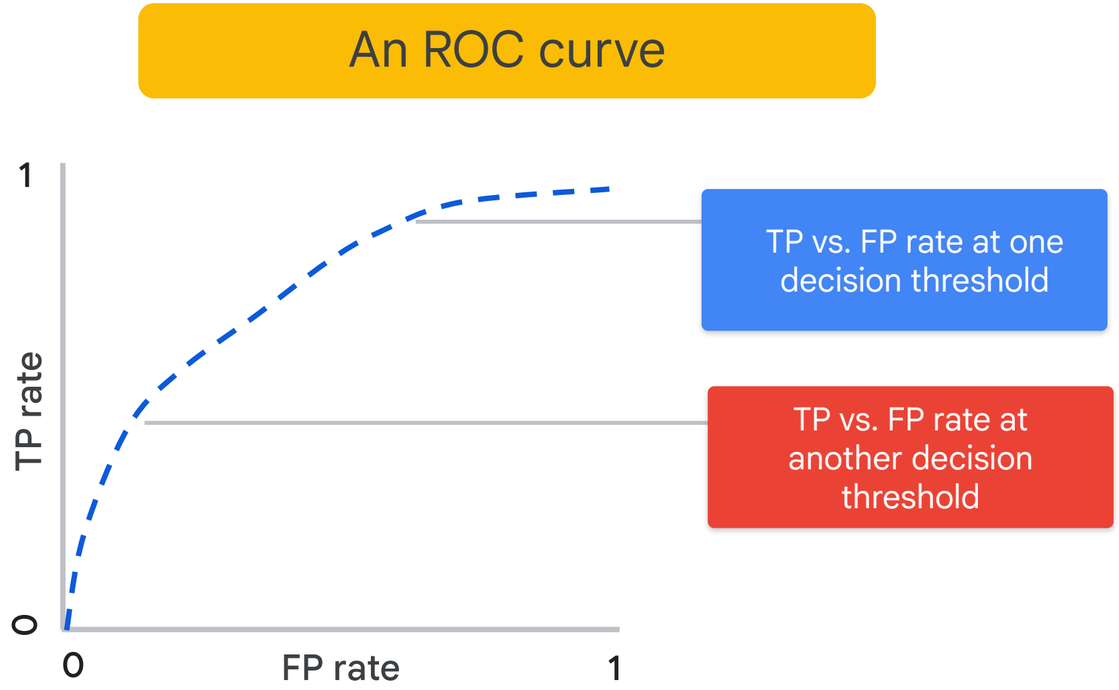


Sub-group 3

Use the ROC curve to choose the decision threshold based on decision criteria



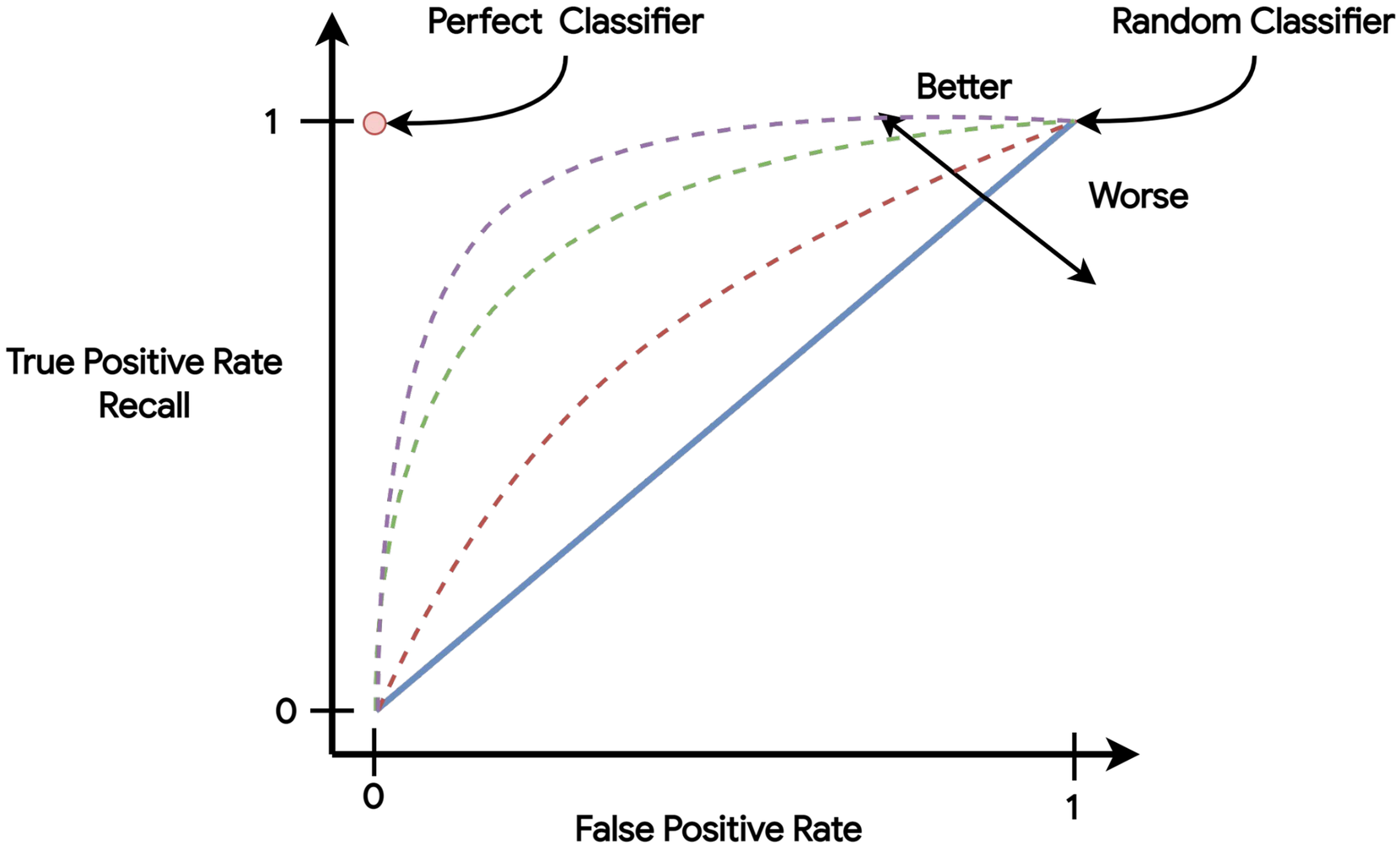
ROC curve



Confidence threshold 0.5

All labels	
PR AUC	0.696
ROC AUC	0.81
Log loss	0.815
F1 score	0.5686126
Precision	60.4%
Recall	53.7%
Created	Sep 29, 2021 12:52:20 AM

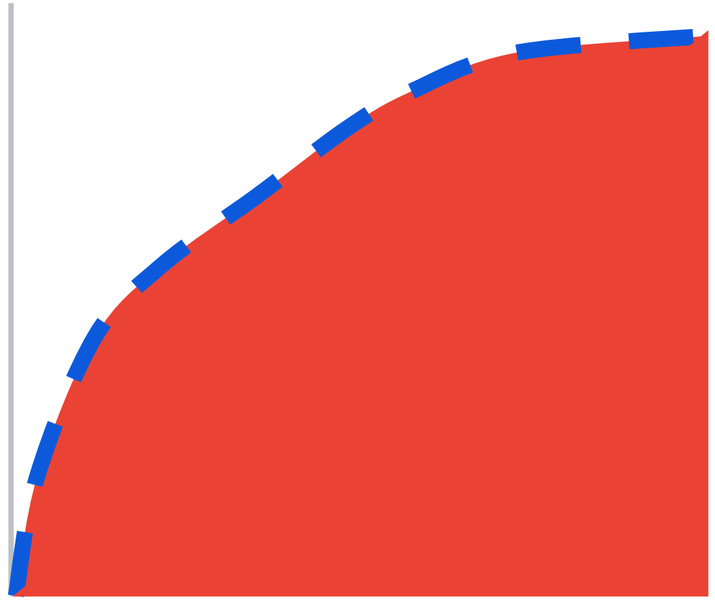
AUC between Classifiers



The Area-Under-Curve (AUC) provides an aggregate measure of performance across all possible classification thresholds

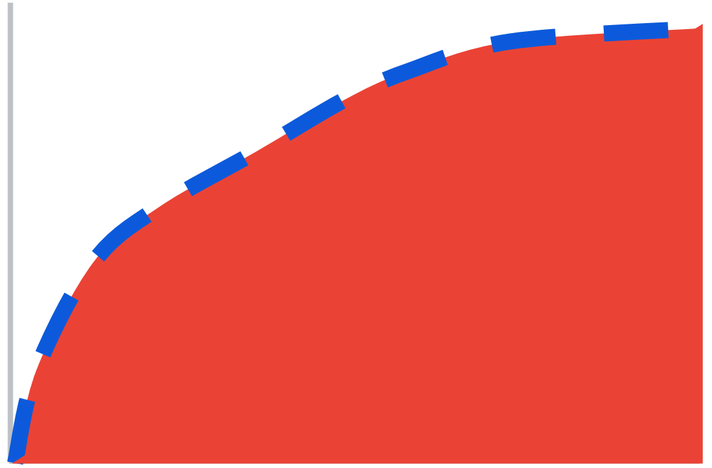
AUC helps you choose between models when you don’t know what decision threshold is going to be ultimately used.

“If we pick a random positive and a random negative, what’s the probability my model scores them in the correct relative order?”



AUC

- AUC = area under the ROC curve.
- Interpretation: If we pick a random positive and a random negative, what's the probability that my model scores them in the correct relative order?
- Intuition: Gives an aggregate measure of performance aggregated across all possible classification thresholds.



Confidence threshold 0.5

All labels	
PR AUC	0.696
ROC AUC	0.81
Log loss	0.815
F1 score	0.5686126
Precision	60.4%
Recall	53.7%
Created	Sep 29, 2021 12:52:20 AM

Recommended Additional Activities

Recommendation Systems with TensorFlow on Google Cloud

Sometimes, false positives are better than false negatives

Privacy in images

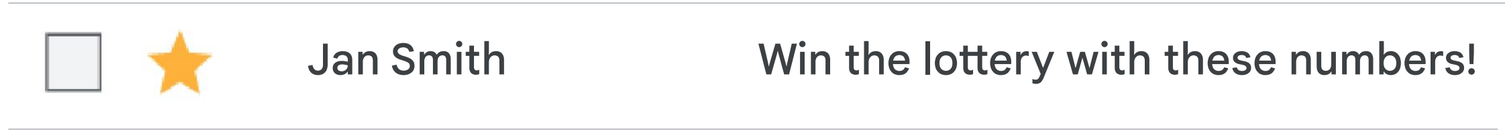
False positive

False negative

Sometimes, false negatives are better than false positives

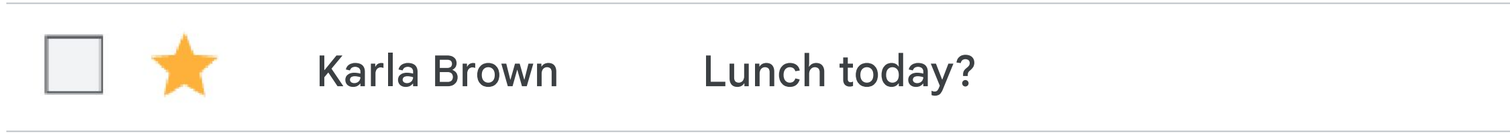
False negative:

E-mail that is SPAM is not caught, so you see it in your inbox.

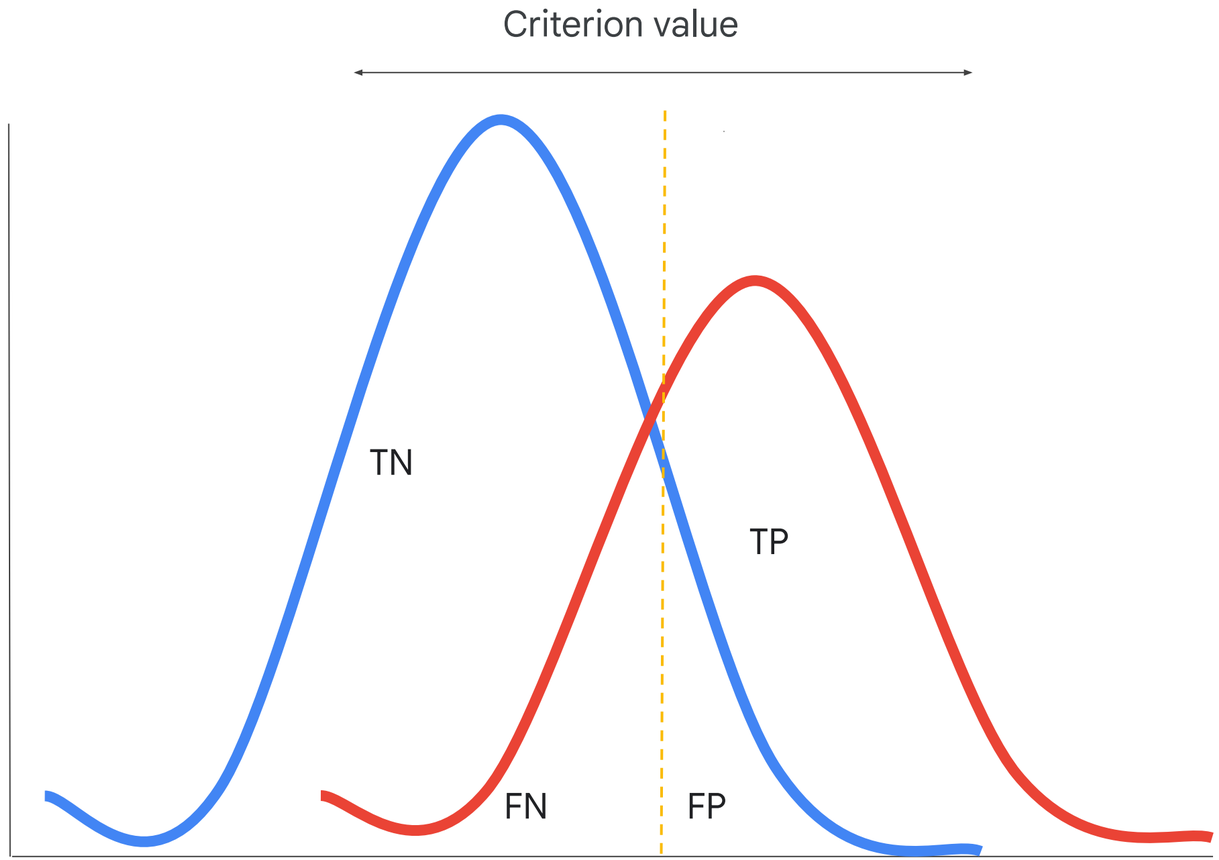


False positive:

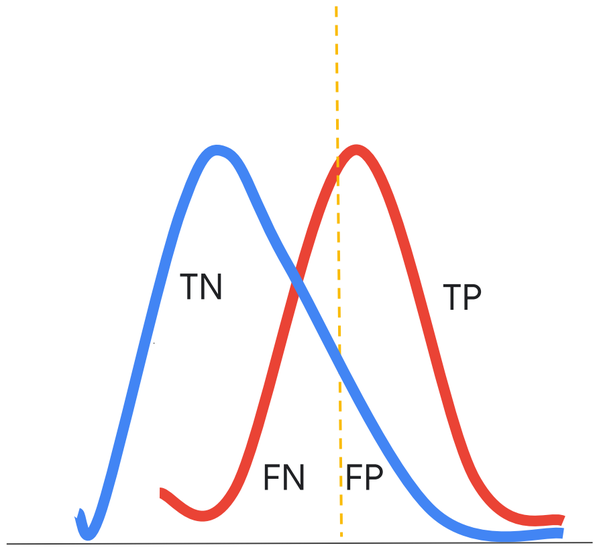
E-mail flagged as SPAM is removed from your inbox.



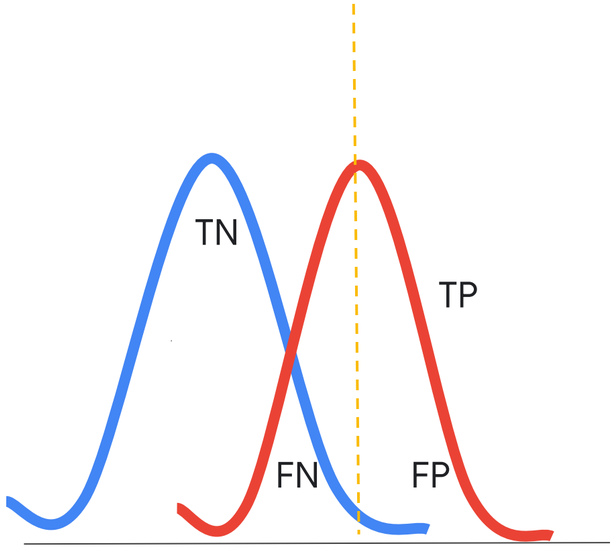
Find the threshold that brings the precision or recall to acceptable values



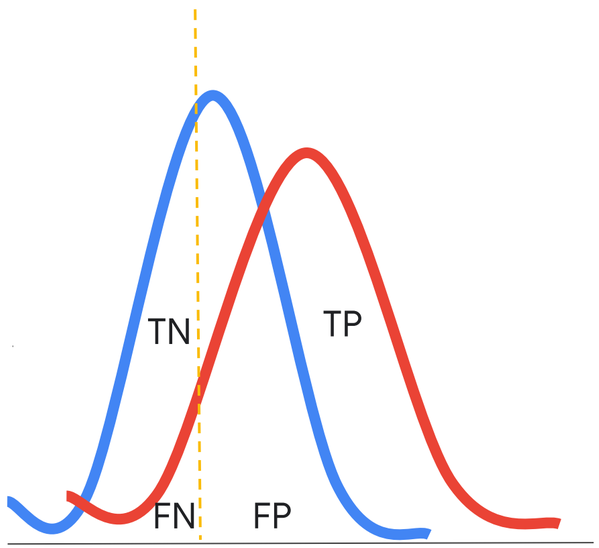
Check the precision/recall you obtain with that threshold in each of your subgroups



Sub-group 1

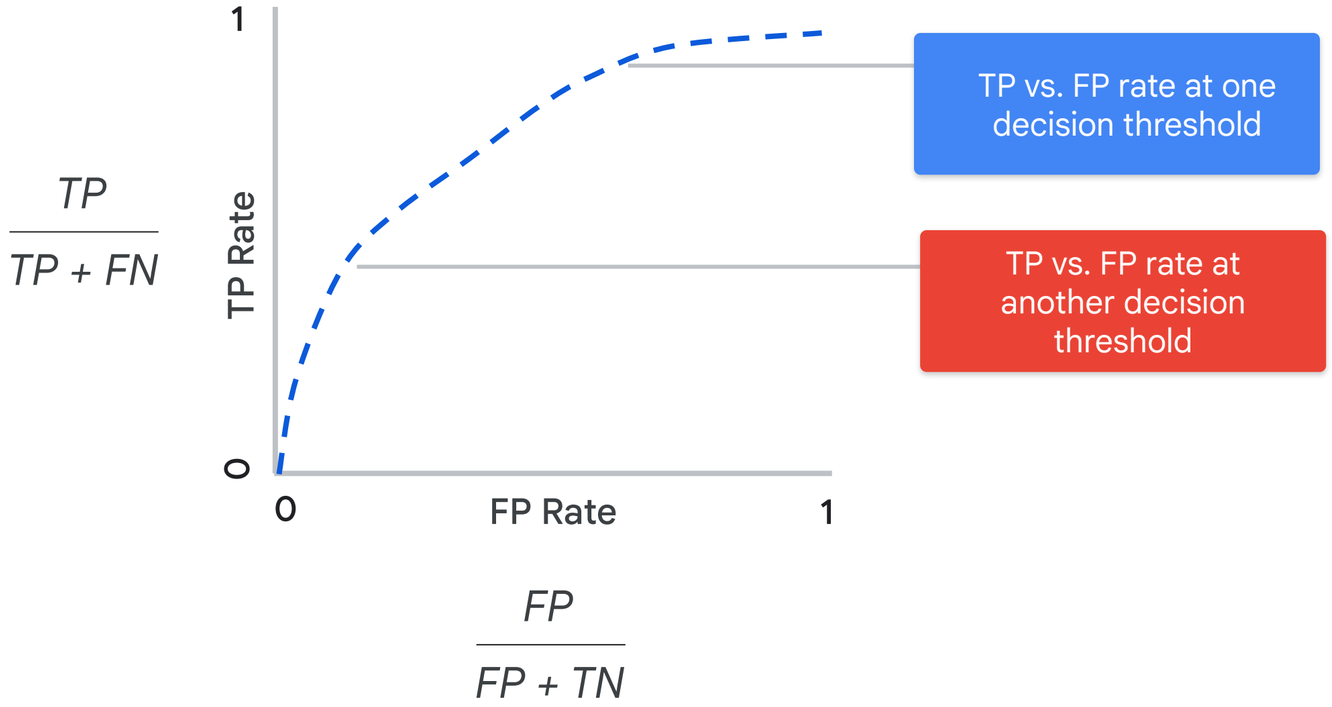


Sub-group 2

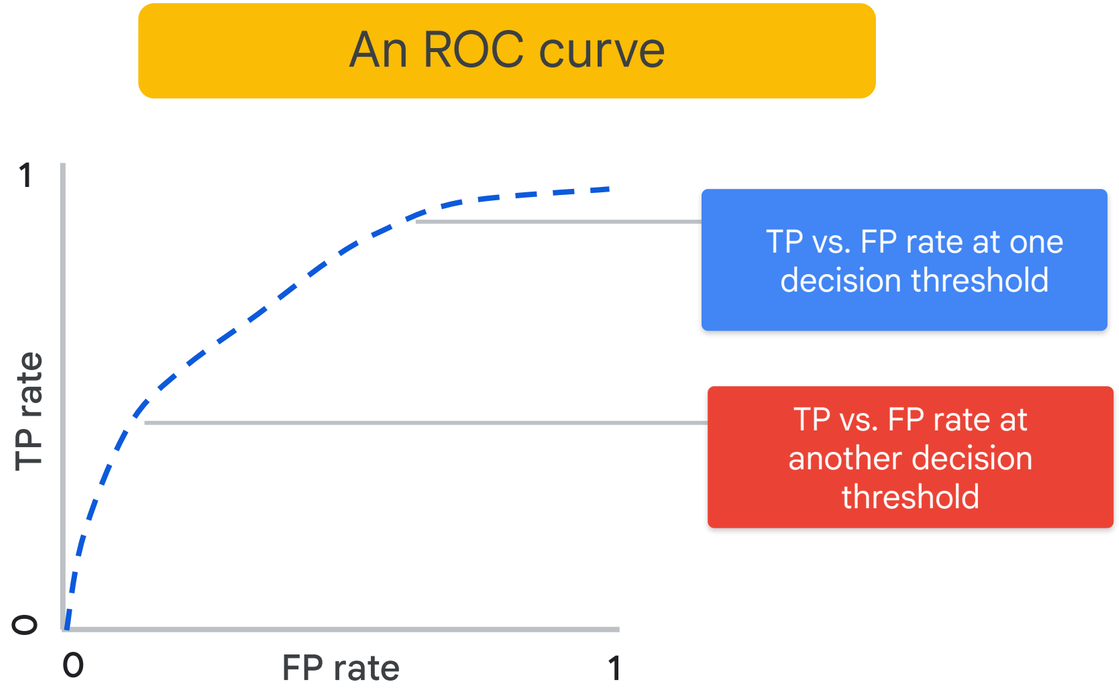


Sub-group 3

Use the ROC curve to choose the decision threshold based on decision criteria



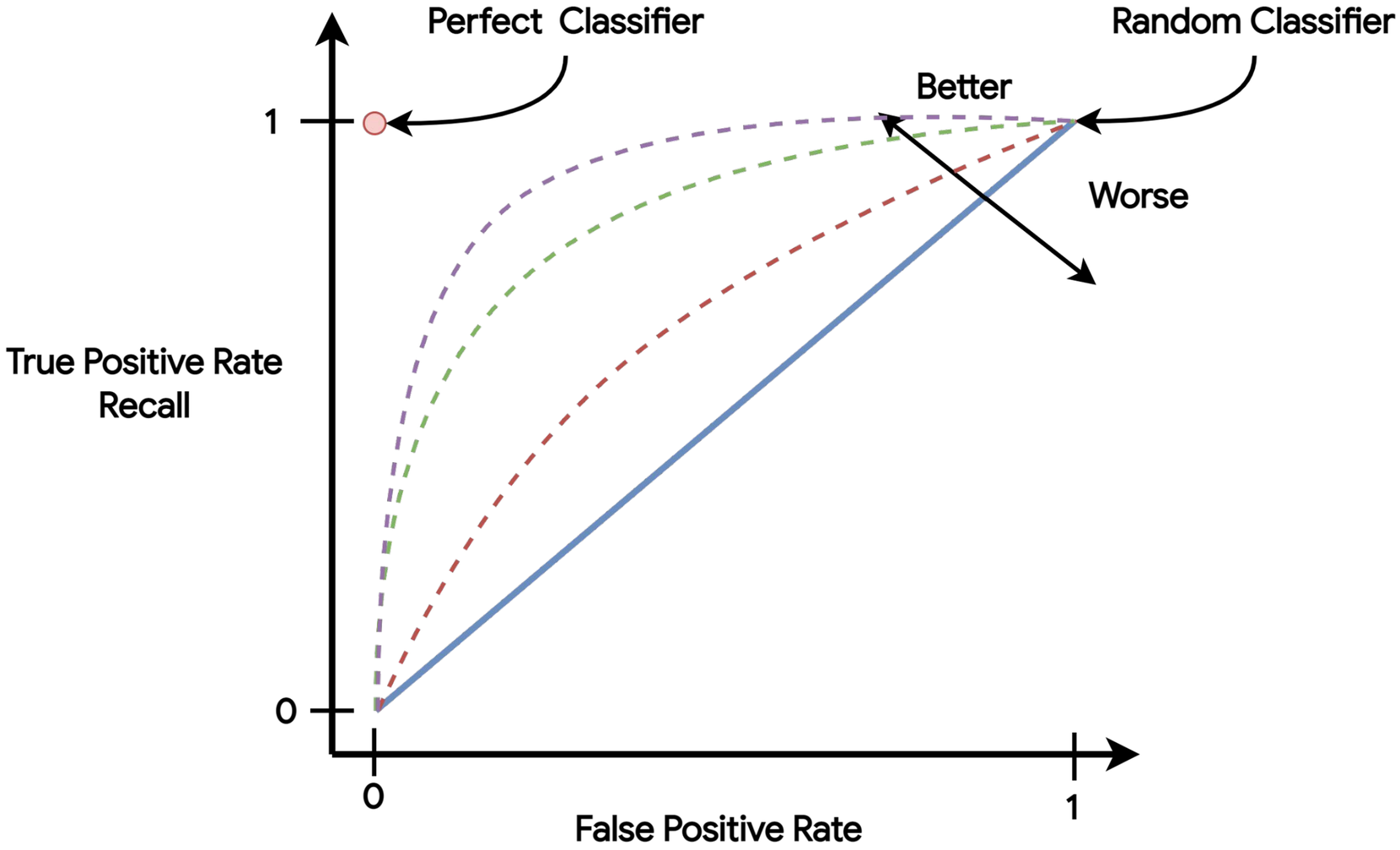
ROC curve



Confidence threshold 0.5

All labels	
PR AUC	0.696
ROC AUC	0.81
Log loss	0.815
F1 score	0.5686126
Precision	60.4%
Recall	53.7%
Created	Sep 29, 2021 12:52:20 AM

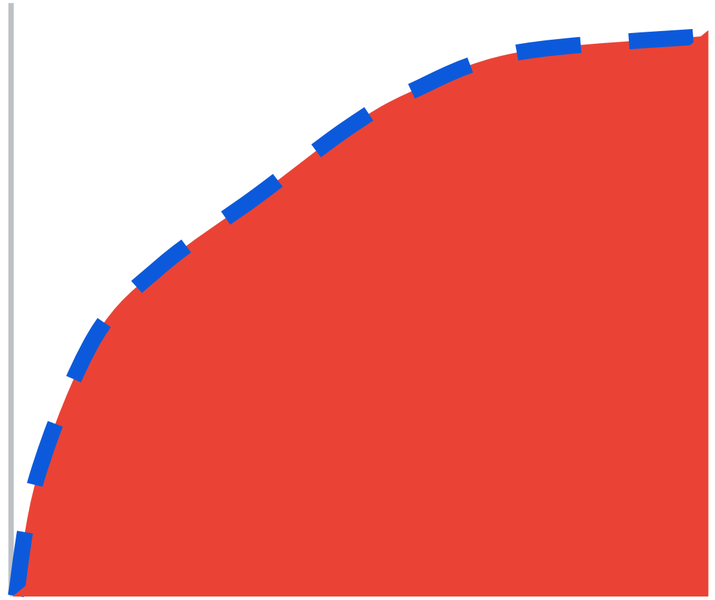
AUC between Classifiers



The Area-Under-Curve (AUC) provides an aggregate measure of performance across all possible classification thresholds

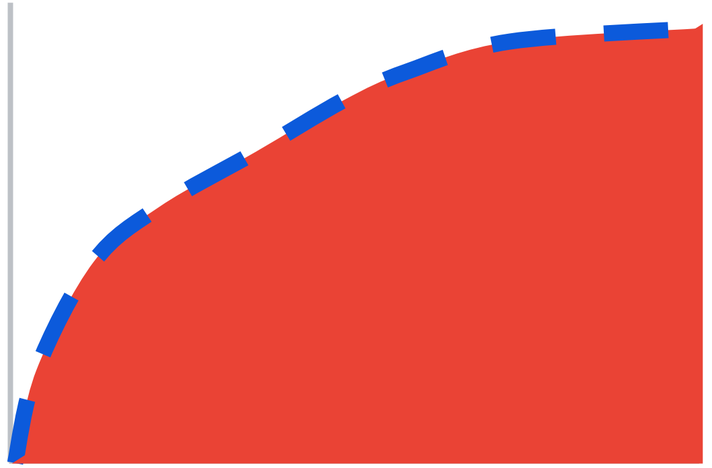
AUC helps you choose between models when you don’t know what decision threshold is going to be ultimately used.

“If we pick a random positive and a random negative, what’s the probability my model scores them in the correct relative order?”



AUC

- AUC = area under the ROC curve.
- Interpretation: If we pick a random positive and a random negative, what's the probability that my model scores them in the correct relative order?
- Intuition: Gives an aggregate measure of performance aggregated across all possible classification thresholds.



Confidence threshold 0.5

All labels	
PR AUC	0.696
ROC AUC	0.81
Log loss	0.815
F1 score	0.5686126
Precision	60.4%
Recall	53.7%
Created	Sep 29, 2021 12:52:20 AM

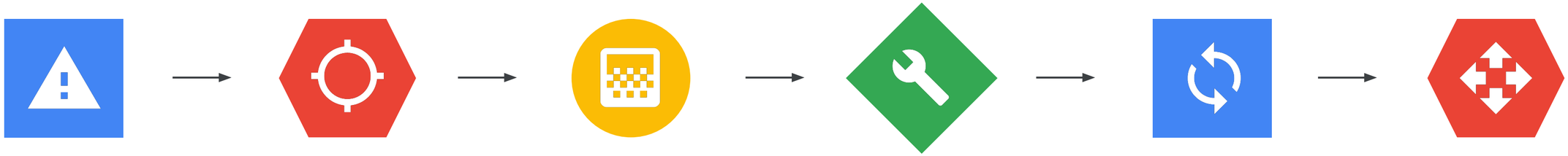
Recommended Additional Activities

Recommendation Systems with TensorFlow on Google Cloud

Machine Learning VS Deep Learning

All machine learning starts with a business requirement or problem you are trying to solve.

To build a machine learning model



1 Identify business problem

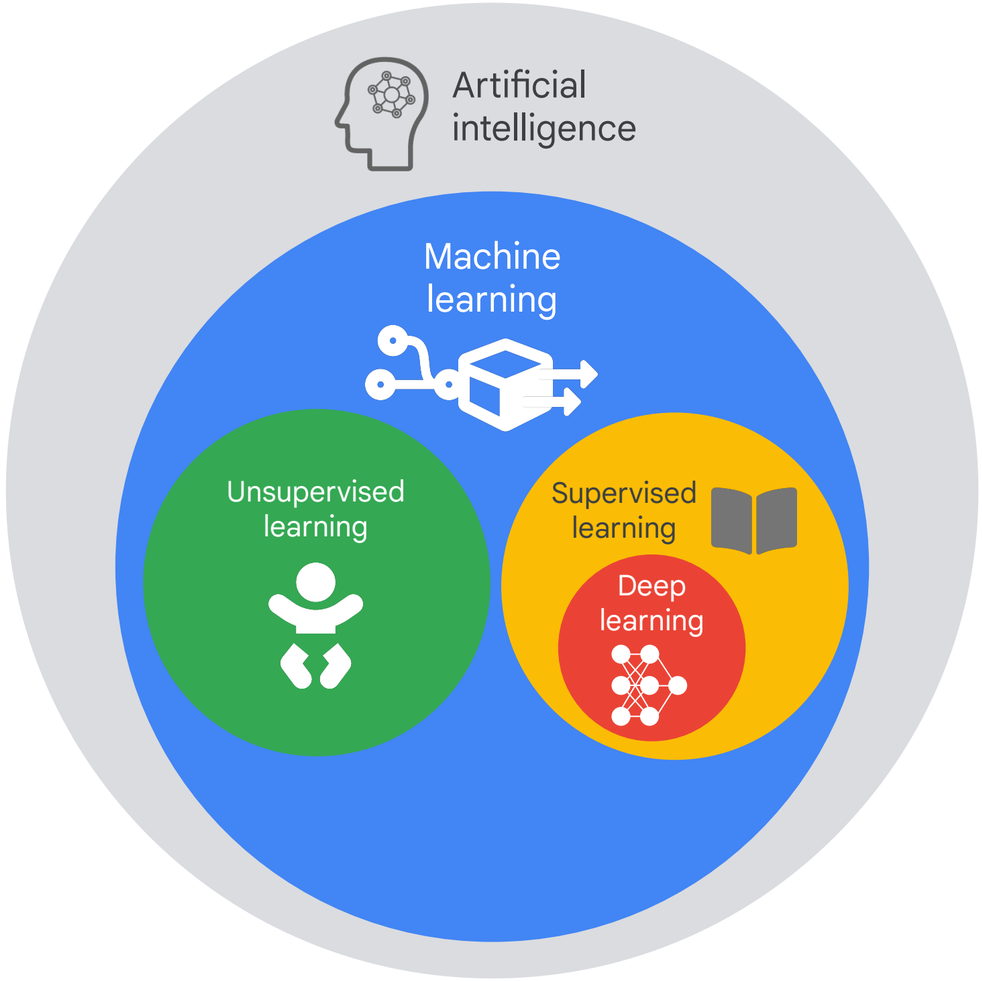
2 Develop hypothesis

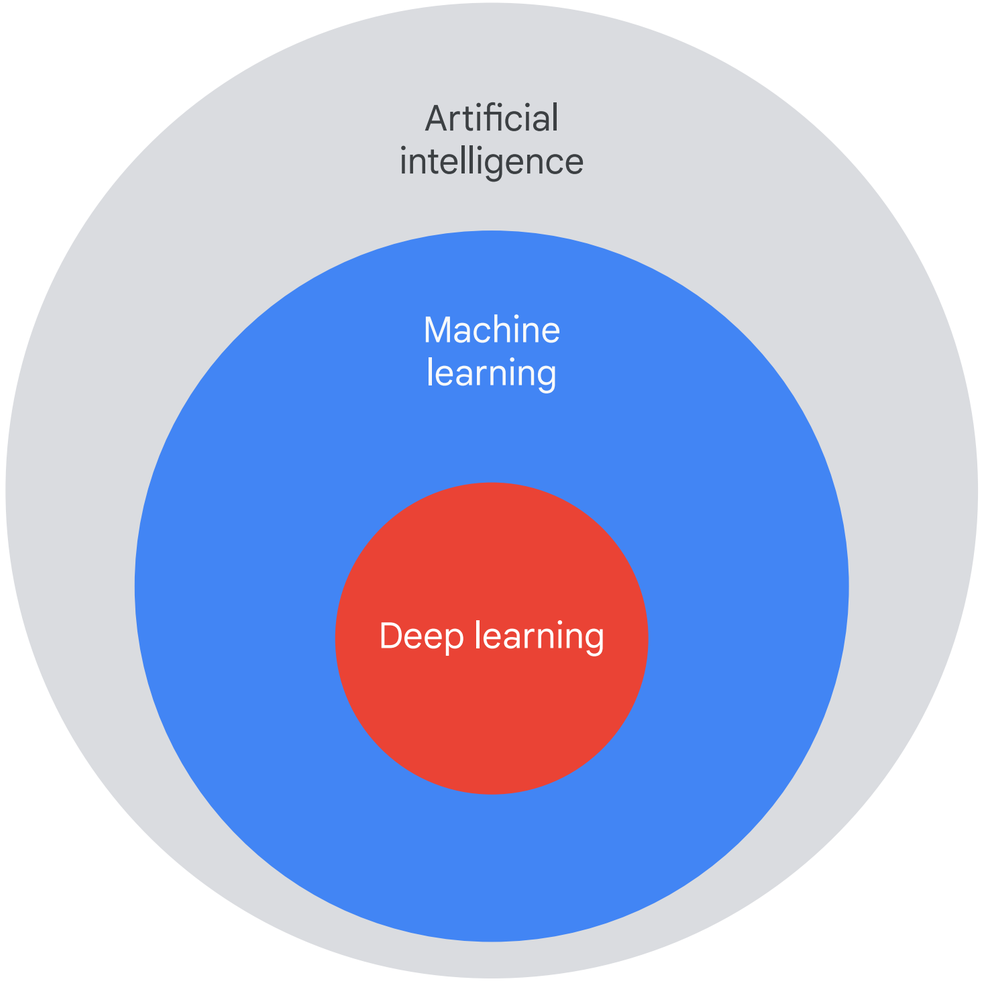
3 Acquire + explore data

4 Build a model

5 Train the model

6 Apply and scale





Factors

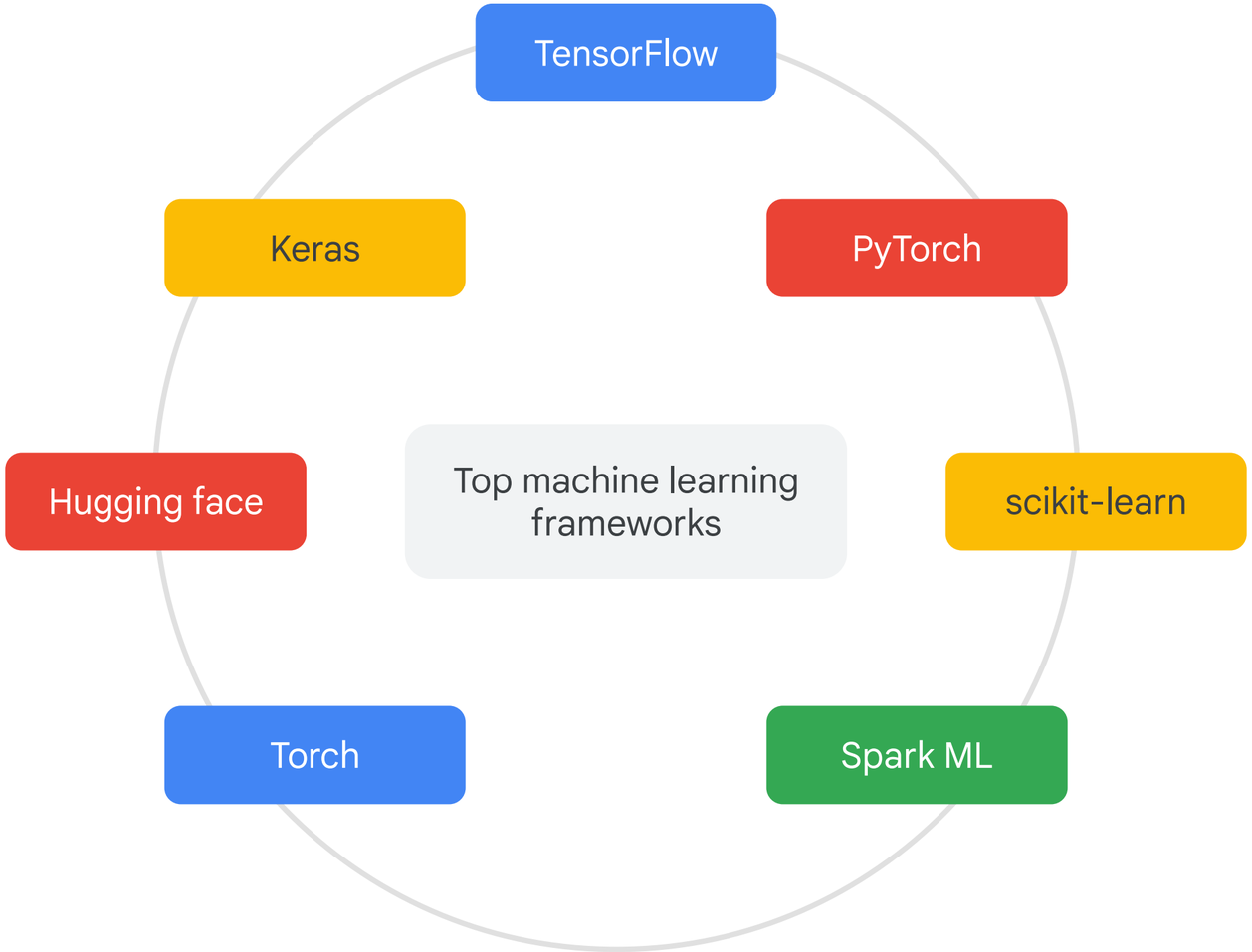
- Data requirement
- Accuracy
- Training time
- Hardware dependency
- Hyperparameter tuning

Deep learning

- Requires large data
- Provides high accuracy
- Takes longer to train
- Requires GPU to train properly
- Can be tuned in various different ways

Machine learning

- Can train on lesser data
- Gives lesser accuracy
- Takes less time to train
- Trains on CPU
- Limited tuning capabilities



Thank you
End of Session 3

Google Cloud


Google Cloud