Vertex AI: Predicting Loan Risk with AutoML

Objectives

You learn how to:
- Upload a dataset to Vertex AI.
- Train a machine learning model with AutoML.
- Evaluate the model performance.
- Deploy the model to an endpoint.
- Get predictions.

Prepare the training data

1. In the Google Cloud Console, on the **Navigation menu**, click **Vertex AI**.
2. Click **Create dataset**.
3. On the Datasets page, give the dataset a name.
4. For the data type and objective, click **Tabular**, and then select **Regression/classification**.
5. Click **Create**.

Upload data

Three options to import data in Vertex AI:

- Upload a local file from your computer.
- Select files from Cloud Storage.
- Select data from BigQuery.

For convenience, the dataset is already uploaded to Cloud Storage.

1. For the data source, select **Select CSV files from Cloud Storage**.
2. For **Import file path**, enter spls/cbl455/loan_risk.csv

Train your model

•Click **Train new model**.

1. For **Objective**, select **Classification**. Select classification instead of regression because you are predicting a distinct number (whether a customer will repay a loan: 0 for repay, 1 for default/not repay) instead of a continuous number.
2. Click **Continue**

Model details

Specify the name of the model and the target column.

1. Give the model a name, such as **LoanRisk**.
2. For **Target column**, select **Default** .
3. (Optional) Explore **Advanced options** to determine how to assign the training vs. testing data and specify the encryption.
4. Click **Continue**.

Train your model

Training options

Specify which columns you want to include in the training model. For example, ClientID might be irrelevant to predict loan risk.

1.Click the minus sign on the **ClientID** row to exclude it from the training model.

2.(Optional) Explore **Advanced options** to select different optimization objectives. For more information about optimization objectives for tabular AutoML models, see https://cloud.google.com/vertex-ai/docs/training/tabular-opt-obj.

3.Click **Continue**.

Compute and pricing

1.For **Budget**, which represents the number of node hours for training, enter **1**. Training your AutoML model for 1 compute hour is typically a good start for understanding whether there is a relationship between the features and label you've selected. From there, you can modify your features and train for more time to improve model performance.

2.Leave early stopping enabled.

3.Click **Start training**.

Depending on the data size and the training method, the training can take from a few minutes to a couple of hours. Normally you would receive an email from Google Cloud when the training job is complete. However, in the Qwiklabs environment, you will not receive an email.

To save the waiting for the model training, you download a pre-trained model in **task 5** to get predictions in **task 6**. This pre-trained model is the training result following the same steps from **task 1** to **task 2**.

Evaluate the model performance

Veretex AI provides many metrics to evaluate the model performance. You focus on three:

- **Precision/Recall curve**
- **Confusion Matrix**
- **Feature Importance**

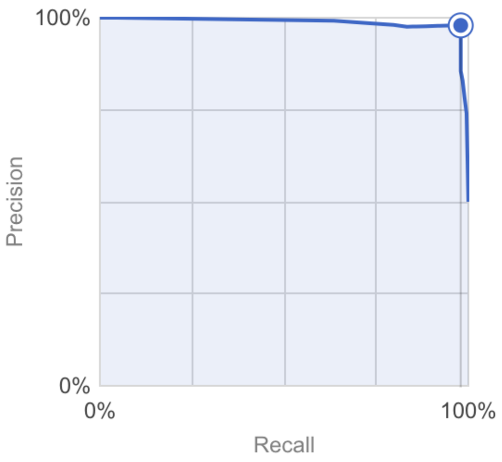
Confidence threshold 0.5

All labels

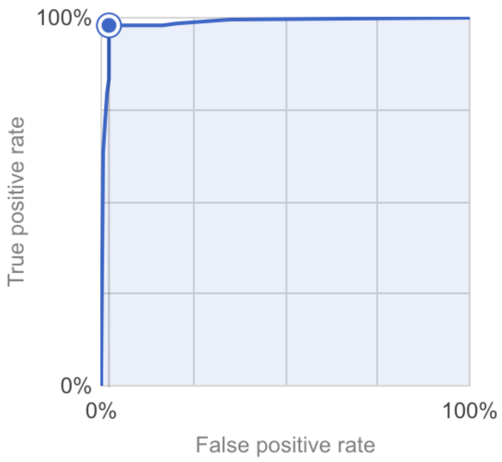
PR AUC	0.984
ROC AUC	0.986
Log loss	0.11
F1 score	0.9791667
Precision	97.9%
Recall	97.9%
Created	Dec 7, 2021, 6:27:33 PM

To evaluate your model, set the **confidence threshold** to see how precision and recall are affected. The best confidence threshold depends on your use case. Read some example scenarios to learn how evaluation metrics can be used.

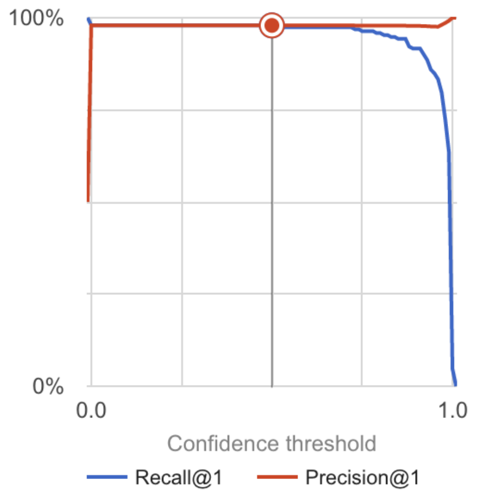
Precision-recall curve



ROC curve



Precision-recall by threshold

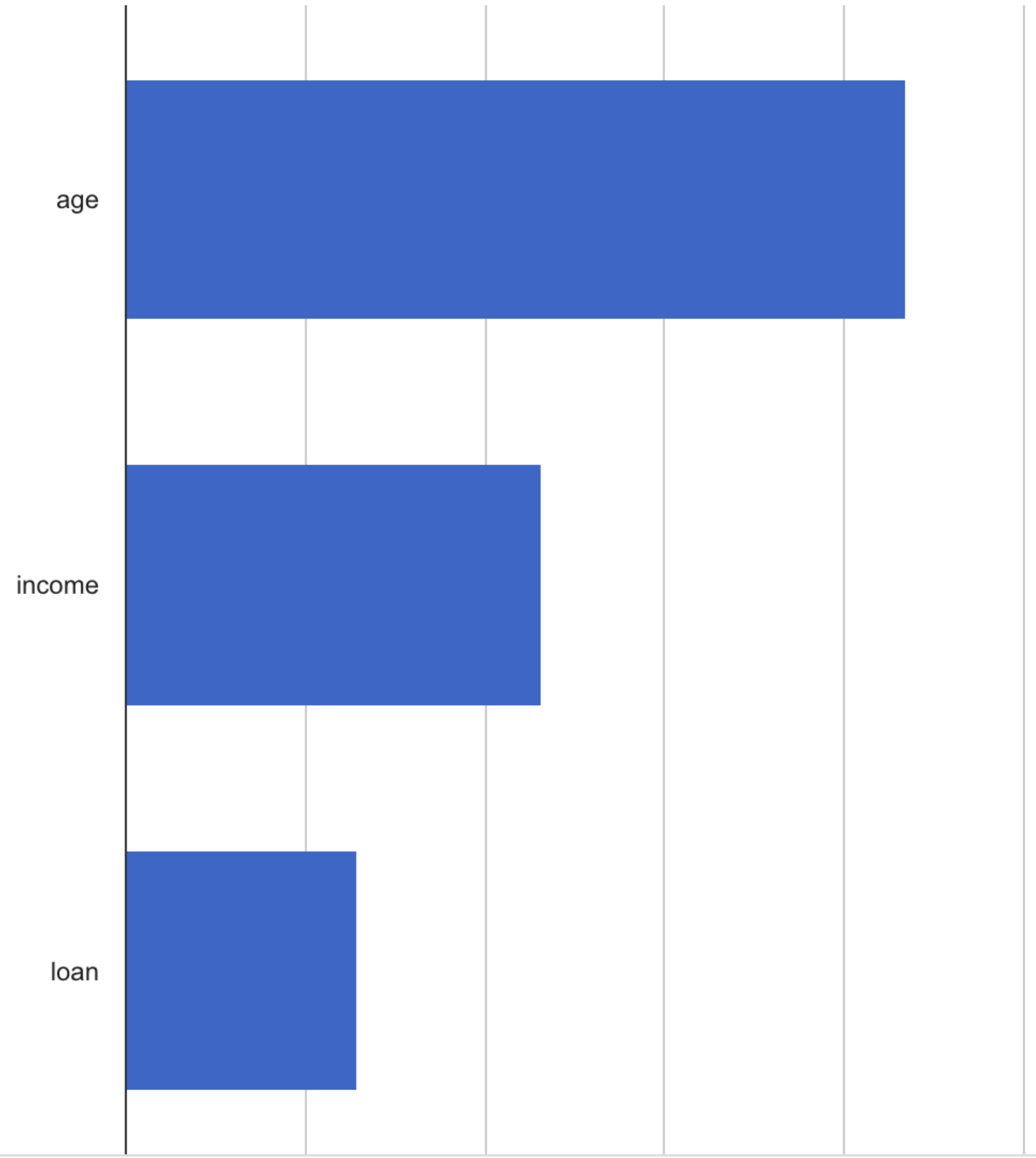


Confusion matrix

This table shows how often the model classified each label correctly (in blue), and which labels were most often confused for that label (in gray).

	Predicted label	
True label	0	1
0	100%	–
1	13%	87%

Feature Importance



Deploy the model

Create and define an endpoint

1. On your model page, on the **Deploy and test** tab, click **Deploy to endpoint**.
2. For **Endpoint name**, enter a name for your endpoint, such as **LoanRisk**.
3. Click **Continue**.

Model settings and monitoring

1. Leave the traffic splitting settings as-is.
2. As the machine type for your model deployment, under **Machine type**, select **n1-standard-8, 8 vCPUs, 30 GiB memory**.
3. Leave the remaining settings as-is.
4. Click **Deploy**.

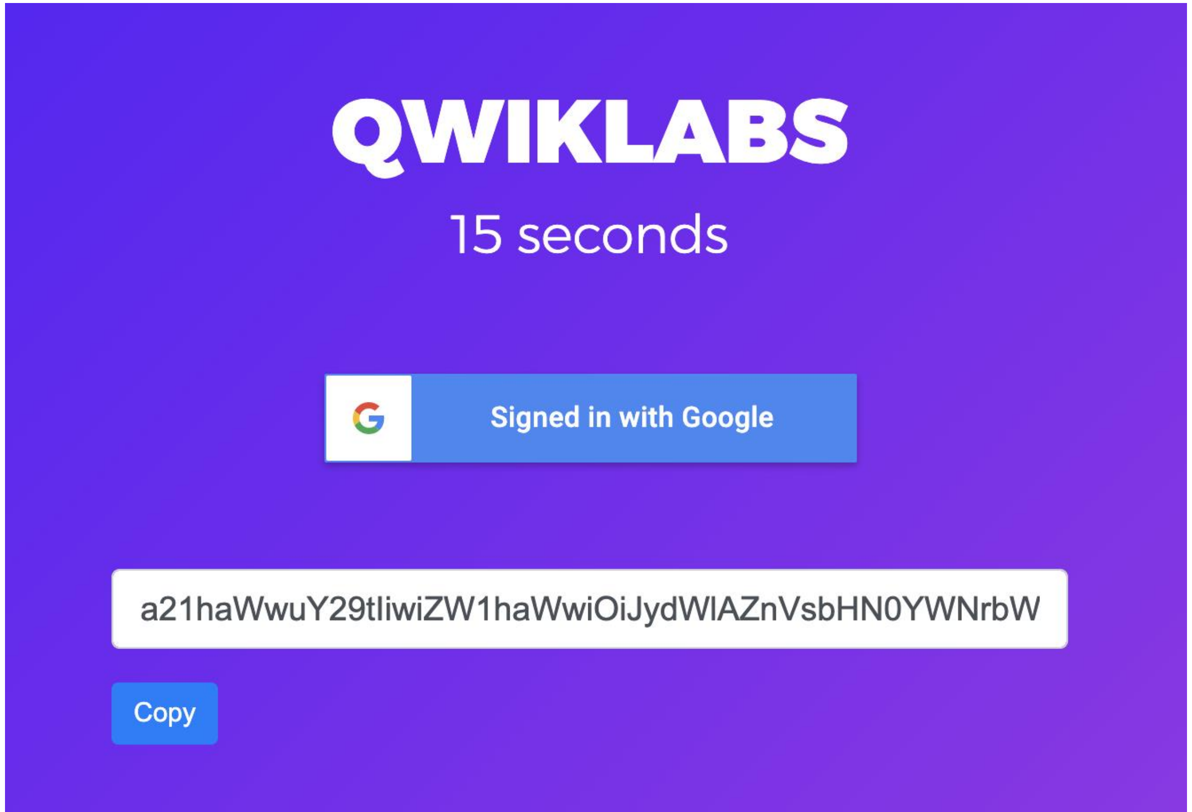
SML Bearer Token

Retrieve your Bearer Token

To allow the pipeline to authenticate, and be authorized to call the endpoint to get the predictions, you will need to provide your Bearer Token.

Follow the instructions below to get your token. If you have issues getting the Bearer Token, this can be due to cookies in the incognito window. If this is happening to you, try this step in a non-incognito window.

1. Log in to https://gsp-auth-kjyo252taq-uc.a.run.app/
2. When logging in, use your student email address and password.
3. Click the **Copy** button. This will copy a very long token to your clipboard.



Get predictions

use the Shared Machine Learning (SML) service to work with an existing trained model

ENVIRONMENT VARIABLE	VALUE
AUTH_TOKEN	Use the value from the previous section
ENDPOINT	https://sml-api-vertex-kjyo252taq-uc.a.run.app/vertex/predict/tabular_classification
INPUT_DATA_FILE	INPUT-JSON

To use the trained model, you will need to create some environment variables.

1. Open a Cloud Shell window.
2. Replace INSERT_SML_BEARER_TOKEN with the bearer token value from the previous section:
   AUTH_TOKEN="INSERT_SML_BEARER_TOKEN„
3. Download the lab assets:
   gsutil cp gs://spls/cbl455/cbl455.tar.gz .

Get predictions

4.Extract the lab assets:
tar -xvf cbl455.tar.gz
5.Create an ENDPOINT environment variable:
ENDPOINT=https://sml-api-vertex-kjyo252taq-uc.a.run.app/vertex/predict/tabular_classification
6.Create a INPUT_DATA_FILE environment variable:
INPUT_DATA_FILE="INPUT-JSON„
The file INPUT-JSON is composed of the follwing values:

age	ClientID	income	loan
40.77	997	44964.01	3944.22

Get predictions

Test the SML Service by passing the parameters specified in the environment variables:

1.Perform a request to the SML service:

```
./smlproxy tabular \
  -a $AUTH_TOKEN \
  -e $ENDPOINT \
  -d $INPUT_DATA_FILE
```

This query should result in a response similar to this:

SML Tabular HTTP Response:

2022/01/10 15:04:45 {"model_class":"0","model_score":0.9999981}

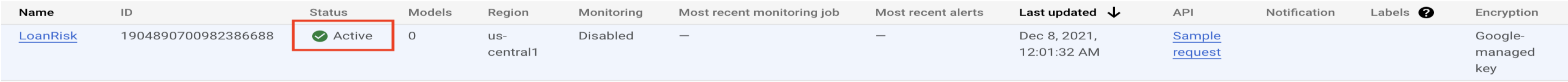
Get predictions

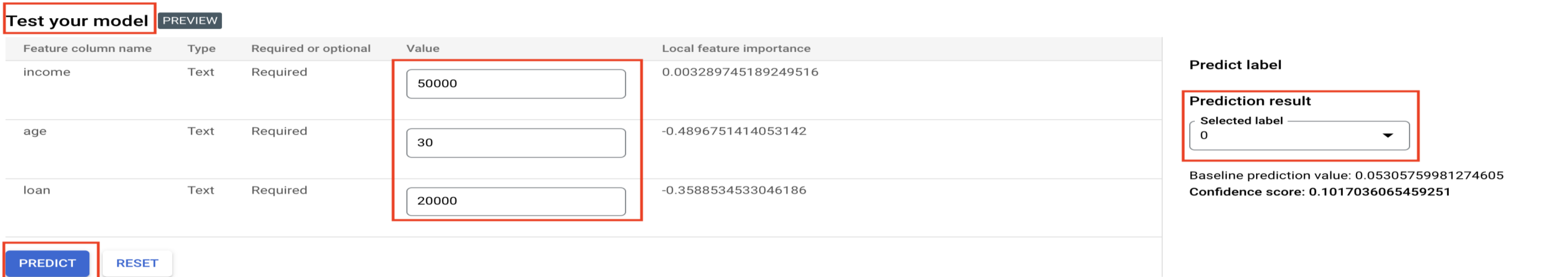
Test the SML Service by passing the parameters specified in the environment variables:
1.Edit the file INPUT-JSON and replace the original values.
2.Perform a request to the SML service:

```
./smlproxy tabular \
  -a $AUTH_TOKEN \
  -e $ENDPOINT \
  -d $INPUT_DATA_FILE
```

In this case, assuming that the person's income is 50,000, age 30, and loan 20,000, the model predicts that this person will repay the loan

SML Tabular HTTP Response:
2022/01/10 15:04:45 {"model_class":"0","model_score":0.9999981}





Adecco Randstadt Euroingenering ARWA