Predicting Visitor Purchases with BigQuery ML

Overview

BigQuery ML (BigQuery machine learning) is a feature in BigQuery where data analysts can create, train, evaluate, and predict with machine learning models with minimal coding.
The Google Analytics Sample Ecommerce dataset that has millions of Google Analytics records for the Google Merchandise Store loaded into BigQuery. In this lab, you will use this data to run some typical queries that businesses would want to know about their customers' purchasing habits.

Objectives

In this lab, you learn to perform the following tasks:

- Use BigQuery to find public datasets
- Query and explore the ecommerce dataset
- Create a training and evaluation dataset to be used for batch prediction
- Create a classification (logistic regression) model in BigQuery ML
- Evaluate the performance of your machine learning model
- Predict and rank the probability that a visitor will make a purchase

Set up your environments

Lab setup

For each lab, you get a new Google Cloud project and set of resources for a fixed time at no cost.

1. Sign in to Qwiklabs using an **incognito window**.

2. Note the lab's access time (for example, `1:15:00`), and make sure you can finish within that time.
   There is no pause feature. You can restart if needed, but you have to start at the beginning.

3. When ready, click **Start lab**.

4. Note your lab credentials (**Username** and **Password**). You will use them to sign in to the Google Cloud Console.

5. Click **Open Google Console**.

6. Click **Use another account** and copy/paste credentials for **this** lab into the prompts.
   If you use other credentials, you'll receive errors or **incur charges**.

7. Accept the terms and skip the recovery resource page.

Note: Do not click **End Lab** unless you have finished the lab or want to restart it. This clears your work and removes the project.

Open BigQuery Console

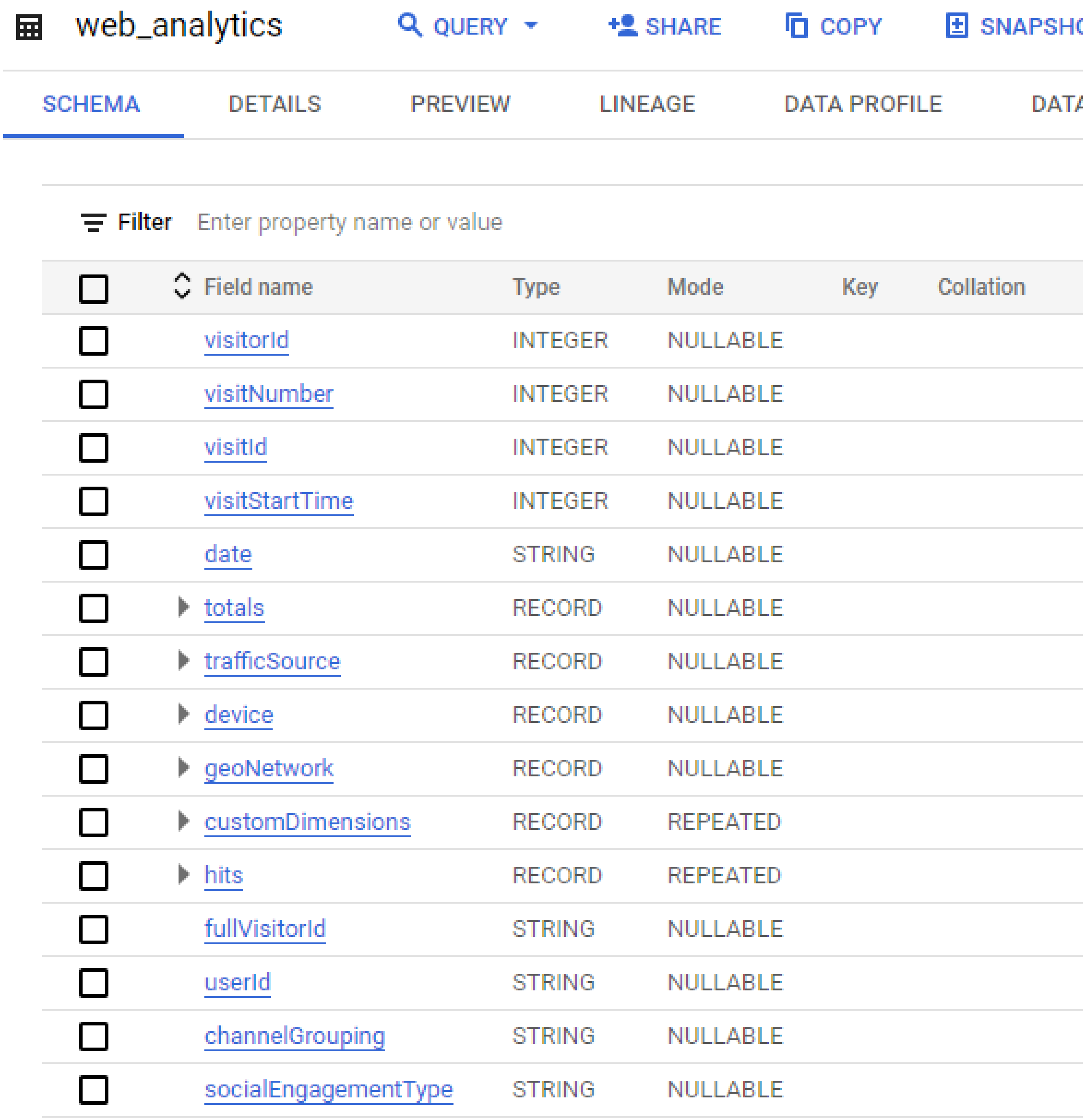
1. In the Google Cloud Console, select **Navigation menu** > **BigQuery**.

The **Welcome to BigQuery in the Cloud Console** message box opens. This message box provides a link to the quickstart guide and lists UI updates.

2. Click **Done**.

Access the course dataset

Once BigQuery is open, open the data-to-insights project in a new browser tab to bring this project into your BigQuery projects panel.
The field definitions for the **data-to-insights** ecommerce dataset are on the[UA] BigQuery Export schema page. Keep the link open in a new tab for reference.



□	Field name	Type	Mode	Key	Collation
□	visitorId	INTEGER	NULLABLE		
□	visitNumber	INTEGER	NULLABLE		
□	visitId	INTEGER	NULLABLE		
□	visitStartTime	INTEGER	NULLABLE		
□	date	STRING	NULLABLE		
□	▸ totals	RECORD	NULLABLE		
□	▸ trafficSource	RECORD	NULLABLE		
□	▸ device	RECORD	NULLABLE		
□	▸ geoNetwork	RECORD	NULLABLE		
□	▸ customDimensions	RECORD	REPEATED		
□	▸ hits	RECORD	REPEATED		
□	fullVisitorId	STRING	NULLABLE		
□	userId	STRING	NULLABLE		
□	channelGrouping	STRING	NULLABLE		
□	socialEngagementType	STRING	NULLABLE		

Task 1. Explore ecommerce data

Scenario: Your data analyst team exported the Google Analytics logs for an ecommerce website into BigQuery and created a new table of all the raw ecommerce visitor session data for you to explore. Using this data, you'll try to answer a few questions.

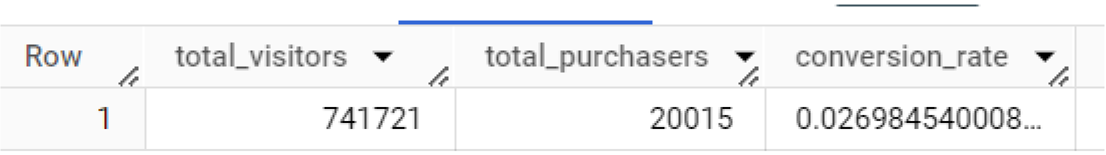
Question: Out of the total visitors who visited our website, what % made a purchase?

1. Click the query **EDITOR**.
2. Add the following to the New Query field:

```
#standardSQL
WITH visitors AS(
SELECT
COUNT(DISTINCT fullVisitorId) AS total_visitors
FROM `data-to-insights.ecommerce.web_analytics`
),
purchasers AS(
SELECT
COUNT(DISTINCT fullVisitorId) AS total_purchasers
FROM `data-to-insights.ecommerce.web_analytics`
WHERE totals.transactions IS NOT NULL
)
SELECT
  total_visitors,
  total_purchasers,
  total_purchasers / total_visitors AS conversion_rate
FROM visitors, purchasers
```

3. Click **Run**.

The result: 2.69%



Row	total_visitors	total_purchasers	conversion_rate
1	741721	20015	0.026984540008...

Question: What are the top 5 selling products?

4. Add the following query in the query **EDITOR**, and then click **Run**:

```
SELECT
  p.v2ProductName,
  p.v2ProductCategory,
  SUM(p.productQuantity) AS units_sold,
  ROUND(SUM(p.localProductRevenue/1000000),2) AS revenue
FROM `data-to-insights.ecommerce.web_analytics`,
UNNEST(hits) AS h,
UNNEST(h.product) AS p
GROUP BY 1, 2
ORDER BY revenue DESC
LIMIT 5;
```

The result:

Row	v2ProductName	v2ProductCategory	units_sold	revenue
1	Nest® Learning Thermostat 3rd Gen-USA - Stainless Steel	Nest-USA	17651	870976.95
2	Nest® Cam Outdoor Security Camera - USA	Nest-USA	16930	684034.55
3	Nest® Cam Indoor Security Camera - USA	Nest-USA	14155	548104.47
4	Nest® Protect Smoke + CO White Wired Alarm-USA	Nest-USA	6394	178937.6
5	Nest® Protect Smoke + CO White Battery Alarm-USA	Nest-USA	6340	178572.4

Question: How many visitors bought on subsequent visits to the website?

5. Run the following query to find out:

```
# visitors who bought on a return visit (could have bought on first as
well
WITH all_visitor_stats AS (
SELECT
  fullvisitorid, # 741,721 unique visitors
  IF(COUNTIF(totals.transactions > 0 AND totals.newVisits IS NULL) > 0, 1,
0) AS will_buy_on_return_visit
  FROM `data-to-insights.ecommerce.web_analytics`
  GROUP BY fullvisitorid
)
SELECT
  COUNT(DISTINCT fullvisitorid) AS total_visitors,
  will_buy_on_return_visit
FROM all_visitor_stats
GROUP BY will_buy_on_return_visit
```

The results:

Row	total_visitors	will_buy_on_return_visit
1	729848	0
2	11873	1

Analyzing the results, you can see that (11873 / 729848) = 1.6% of total visitors will return and purchase from the website. This includes the subset of visitors who bought on their very first session and then came back and bought again.

What are some of the reasons a typical ecommerce customer will browse but not buy until a later visit? Choose all that could apply.

The customer wants to comparison shop on other sites before making a purchase decision.

The customer is waiting for products to go on sale or other promotion

The customer is doing additional research

This behavior is very common for luxury goods where significant up-front research and comparison is required by the customer before deciding (think car purchases) but also true to a lesser extent for the merchandise on this site (t-shirts, accessories, etc).

In the world of online marketing, identifying and marketing to these future customers based on the characteristics of their first visit will increase conversion rates and reduce the outflow to competitor sites.

Task 2. Select features and create your training dataset

Now you will create a Machine Learning model in BigQuery to predict whether or not a new user is likely to purchase in the future. Identifying these high-value users can help your marketing team target them with special promotions and ad campaigns.

Google Analytics captures a wide variety of dimensions and measures about a user's visit on this ecommerce website. Browse the complete list of fields in the [UA] BigQuery Export schema Guide and then preview the demo dataset to find useful features that will help a machine learning model understand the relationship between data about a visitor's first time on your website and whether they will return and make a purchase.
Your team decides to test whether these two fields are good inputs for your classification model:

- `totals.bounces` (whether the visitor left the website immediately)
- `totals.timeOnSite` (how long the visitor was on our website)

What are the risks of only using the above two fields?

Whether a user bounces is highly correlated with their time on site (e.g. 0 seconds)

Only using time spent on the site ignores other potential useful columns (features)

Both of the above

Machine learning is only as good as the training data that is fed into it. If there isn't enough information for the model to determine and learn the relationship between your input features and your label (in this case, whether the visitor bought in the future) then you will not have an accurate model. While training a model on just these two fields is a start, you will see if they're good enough to produce an accurate model.

- In the query **EDITOR**, add the following query and then click **Run**:

```
SELECT
  * EXCEPT(fullVisitorId)
FROM
  # features
  (SELECT
    fullVisitorId,
    IFNULL(totals.bounces, 0) AS bounces,
    IFNULL(totals.timeOnSite, 0) AS time_on_site
  FROM
    `data-to-insights.ecommerce.web_analytics`
  WHERE
    totals.newVisits = 1)
  JOIN
  (SELECT
    fullvisitorid,
    IF(COUNTIF(totals.transactions > 0 AND totals.newVisits IS NULL) > 0,
1, 0) AS will_buy_on_return_visit
  FROM
      `data-to-insights.ecommerce.web_analytics`
  GROUP BY fullvisitorid)
  USING (fullVisitorId)
ORDER BY time_on_site DESC
LIMIT 10;
```

Results:

Row	**bounces**	**time_on_site**	**will_buy_on_return_visit**
1	0	15047	0
2	0	12136	0
3	0	11201	0
4	0	10046	0
5	0	9974	0
6	0	9564	0

7	0	9520	0
8	0	9275	1
9	0	9138	0
10	0	8872	0

Which fields are the model features? What is the label (correct answer)?

The feature is will_buy_on_return_visit. The labels are bounces and time_on_site

[✓ The features are bounces and time_on_site. The label is will_buy_on_return_visit]

The features are bounces and will_buy_on_return_visit. The label is time_on_site

Which fields are known after a visitor's first session? (Check all that apply)

[✓ time_on_site]

will_buy_on_return_visit

[✓ visitId]

[✓ bounces]

Which field isn't known until later in the future after their first session?

time_on_site

visitId

[✓ will_buy_on_return_visit]

bounces

Discussion: will_buy_on_return_visit is not known after the first visit. Again, you're predicting for a subset of users who returned to your website and purchased. Since you don't know the future at prediction time, you cannot say with certainty whether a new visitor comes back and purchases. The value of building a ML model is to get the probability of future purchase based on the data gleaned about their first session.

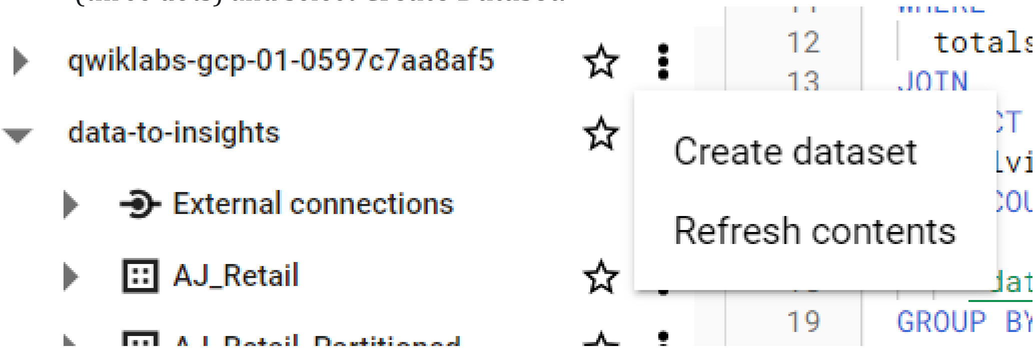
Question: Looking at the initial data results, do you think **time_on_site** and **bounces** will be a good indicator of whether the user will return and purchase or not?

Answer: It's often too early to tell before training and evaluating the model, but at first glance out of the top 10 `time_on_site`, only 1 customer returned to buy, which isn't very promising. Let's see how well the model does.

Task 3. Create a BigQuery dataset to store models

Next, create a new BigQuery dataset which will also store your ML models.

1. In the left pane, click on your project name, and then click on the `View action` icon (three dots) and select **Create Dataset**.



2. In the **Create Dataset** dialog:

- For **Dataset ID**, type **ecommerce**.
- Leave the other values at their defaults.

3. Click **Create dataset**.

Task 4. Select a BigQuery ML model type and specify options

Now that you have your initial features selected, you are now ready to create your first ML model in BigQuery.

There are the two model types to choose from:

Model	Model Type	Label Data type	Example
Forecasting	linear_reg	Numeric value (typically an integer or floating point)	Forecast sales figures for next year given historical sales data.
Classification	logistic_reg	0 or 1 for binary classification	Classify an email as spam or not spam given the context.

Note: There are many additional model types used in Machine Learning (like Neural Networks and decision trees) and available using libraries like TensorFlow. At the time of writing, BigQuery ML supports the two listed above.

Which model type should you choose that will buy or won't buy?

Recommendation model (like matrix_factorization etc.)

Classification model (like logistic_reg etc.)

Forecasting model (like linear_reg etc.)

1. Enter the following query to create a model and specify model options:

```
CREATE OR REPLACE MODEL `ecommerce.classification_model`
OPTIONS
(
model_type='logistic_reg',
labels = ['will_buy_on_return_visit']
)
AS
#standardSQL
SELECT
  * EXCEPT(fullVisitorId)
FROM
  # features
  (SELECT
    fullVisitorId,
    IFNULL(totals.bounces, 0) AS bounces,
    IFNULL(totals.timeOnSite, 0) AS time_on_site
  FROM
    `data-to-insights.ecommerce.web_analytics`
  WHERE
    totals.newVisits = 1
    AND date BETWEEN '20160801' AND '20170430') # train on first 9 months
  JOIN
  (SELECT
    fullvisitorid,
    IF(COUNTIF(totals.transactions > 0 AND totals.newVisits IS NULL) > 0,
1, 0) AS will_buy_on_return_visit
  FROM
      `data-to-insights.ecommerce.web_analytics`
  GROUP BY fullvisitorid)
  USING (fullVisitorId)
;
```

2. Next, click **Run** to train your model.

Wait for the model to train (5 - 10 minutes).

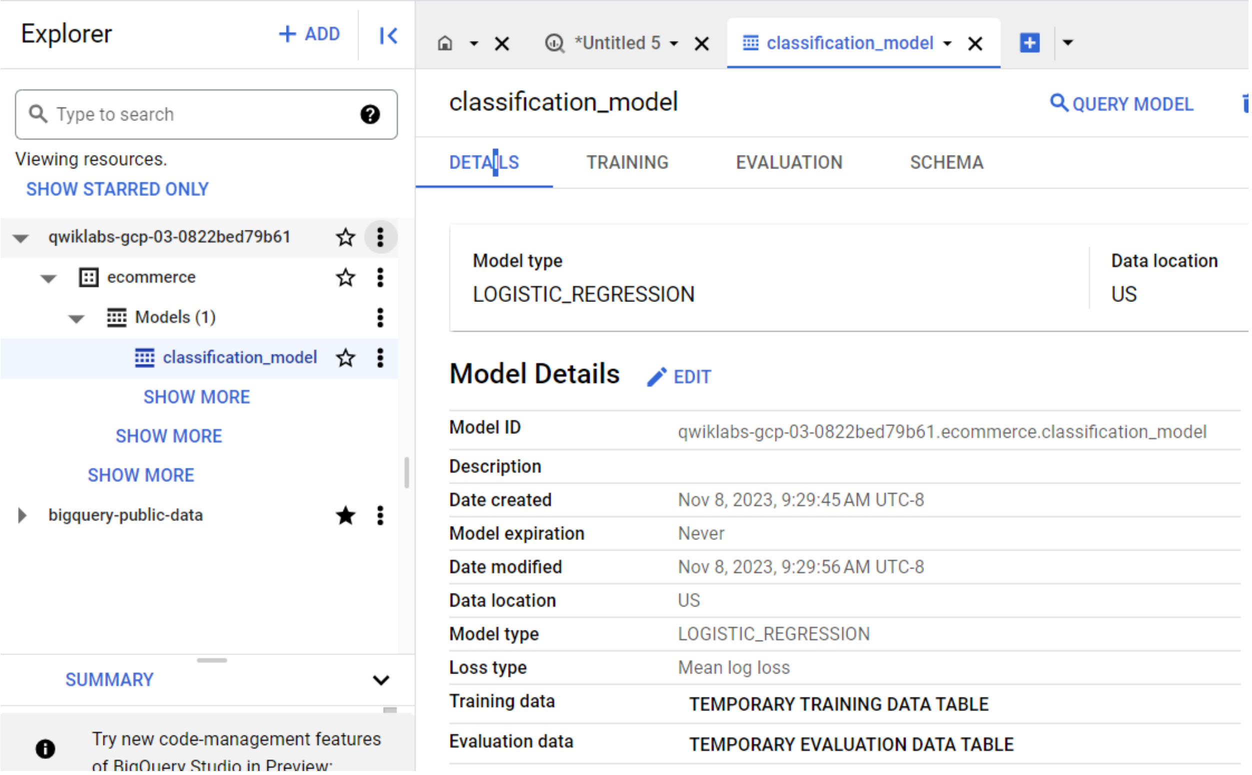
Note: You cannot feed all of your available data to the model during training since you need to save some unseen data points for model evaluation and testing. To accomplish this, add a WHERE clause condition is being used to filter and train on only the first 9 months of session data in your 12 month dataset.

After your model is trained, you will see the message "This statement created a new model named qwiklabs-gcp-xxxxxxxxx:ecommerce.classification_model".

3. Click **Go to model**.

Look inside the ecommerce dataset and confirm **classification_model** now appears.

Next, you will evaluate the performance of the model against new unseen evaluation data.

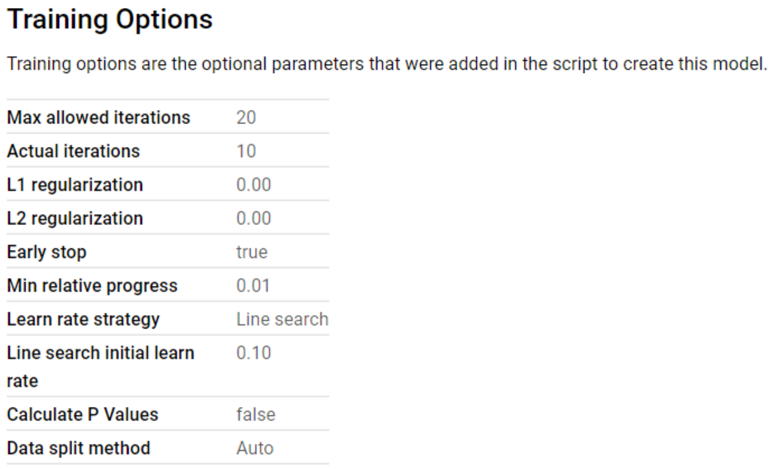


classification_model

Model type	Data location
LOGISTIC_REGRESSION	US

Model Details

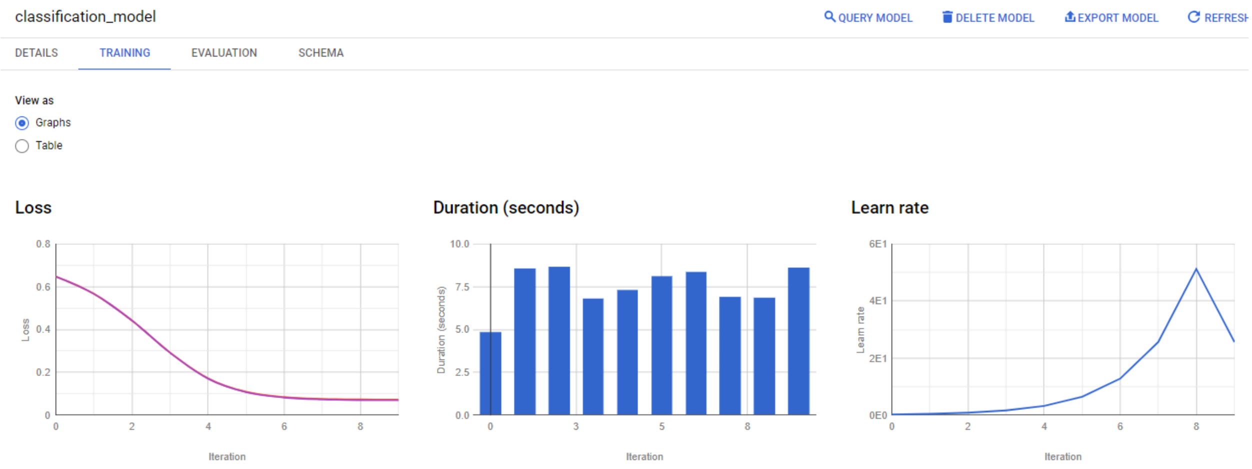
Model ID	qwiklabs-gcp-03-0822bed79b61.ecommerce.classification_model
Description	
Date created	Nov 8, 2023, 9:29:45 AM UTC-8
Model expiration	Never
Date modified	Nov 8, 2023, 9:29:56 AM UTC-8
Data location	US
Model type	LOGISTIC_REGRESSION
Loss type	Mean log loss
Training data	TEMPORARY TRAINING DATA TABLE
Evaluation data	TEMPORARY EVALUATION DATA TABLE



Training Options

Training options are the optional parameters that were added in the script to create this model.

Max allowed iterations	20
Actual iterations	10
L1 regularization	0.00
L2 regularization	0.00
Early stop	true
Min relative progress	0.01
Learn rate strategy	Line search
Line search initial learn rate	0.10
Calculate P Values	false
Data split method	Auto



View as

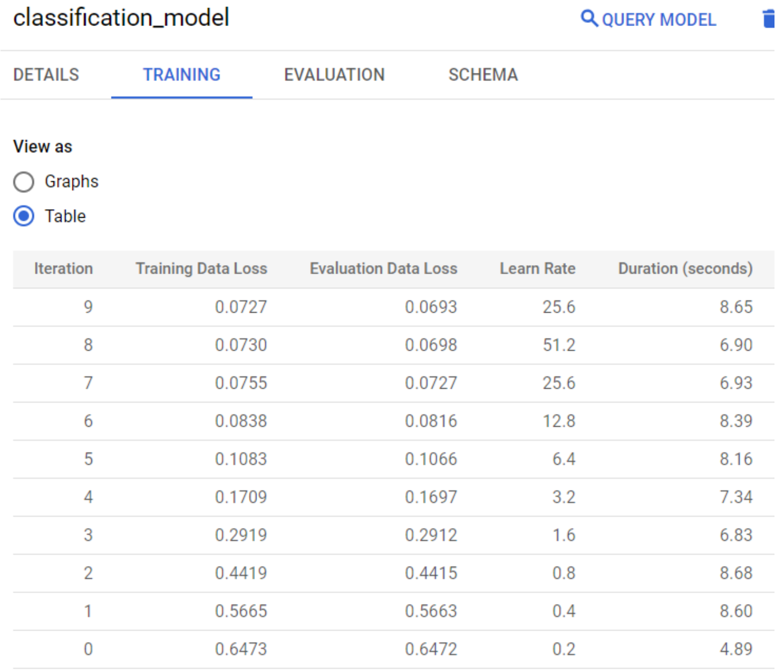
● Graphs

○ Table

Loss

Duration (seconds)

Learn rate

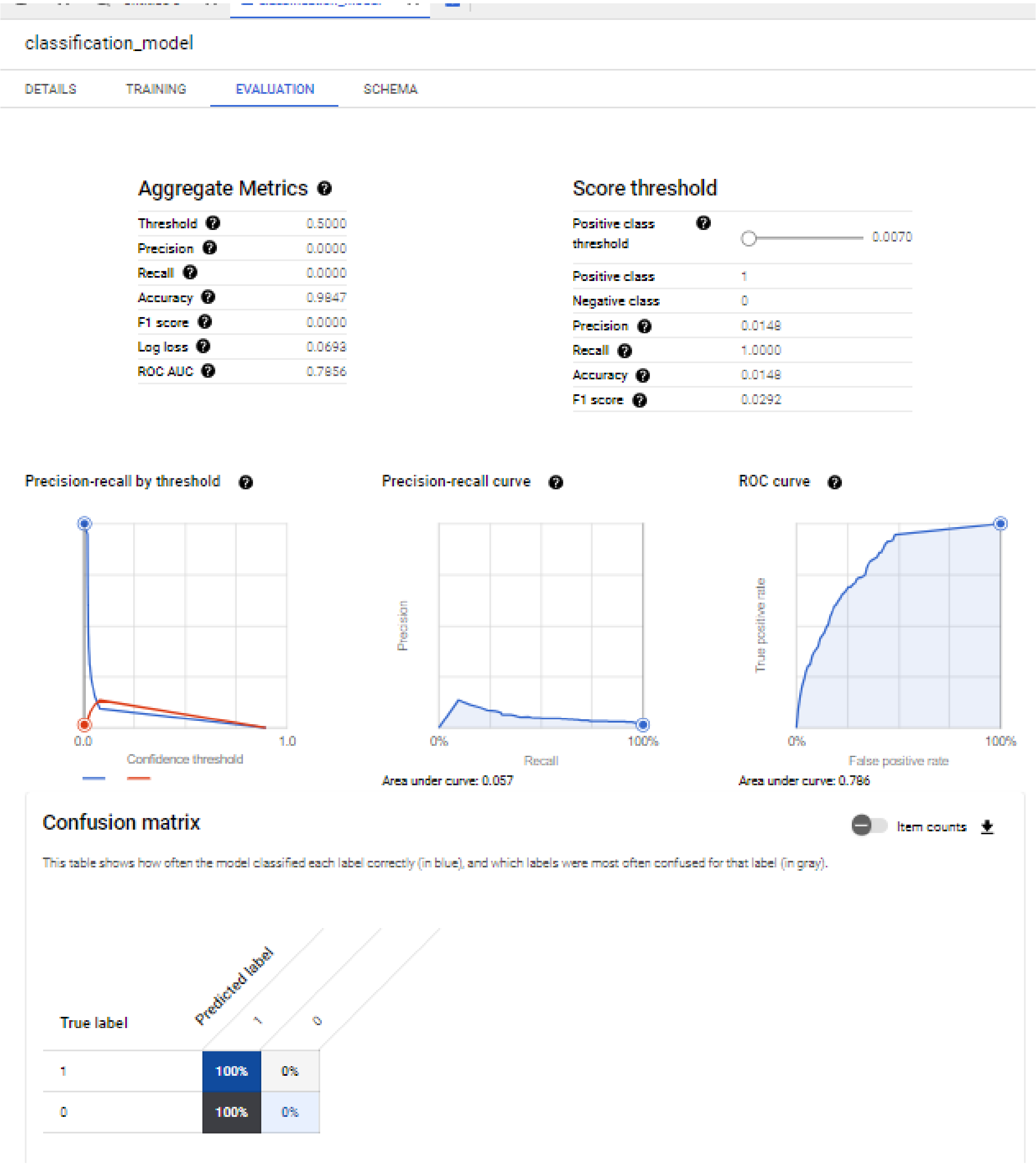


View as

○ Graphs

● Table

Iteration	Training Data Loss	Evaluation Data Loss	Learn Rate	Duration (seconds)
9	0.0727	0.0693	25.6	8.65
8	0.0730	0.0698	51.2	6.90
7	0.0755	0.0727	25.6	6.93
6	0.0838	0.0816	12.8	8.39
5	0.1083	0.1066	6.4	8.16
4	0.1709	0.1697	3.2	7.34
3	0.2919	0.2912	1.6	6.83
2	0.4419	0.4415	0.8	8.68
1	0.5665	0.5663	0.4	8.60
0	0.6473	0.6472	0.2	4.89



classification_model

Aggregate Metrics

Threshold	0.5000
Precision	0.0000
Recall	0.0000
Accuracy	0.9847
F1 score	0.0000
Log loss	0.0693
ROC AUC	0.7856

Score threshold

Positive class threshold	0.0070
Positive class	1
Negative class	0
Precision	0.0148
Recall	1.0000
Accuracy	0.0148
F1 score	0.0292

Precision-recall by threshold

Precision-recall curve

Area under curve: 0.057

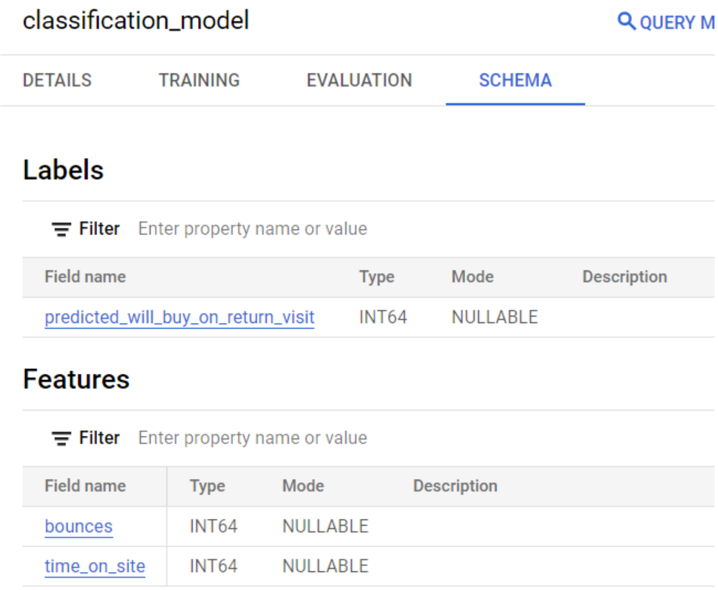
ROC curve

Area under curve: 0.786

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).

True label \ Predicted label	1	0
1	100%	0%
0	100%	0%



classification_model

Labels

Field name	Type	Mode	Description
predicted_will_buy_on_return_visit	INT64	NULLABLE	

Features

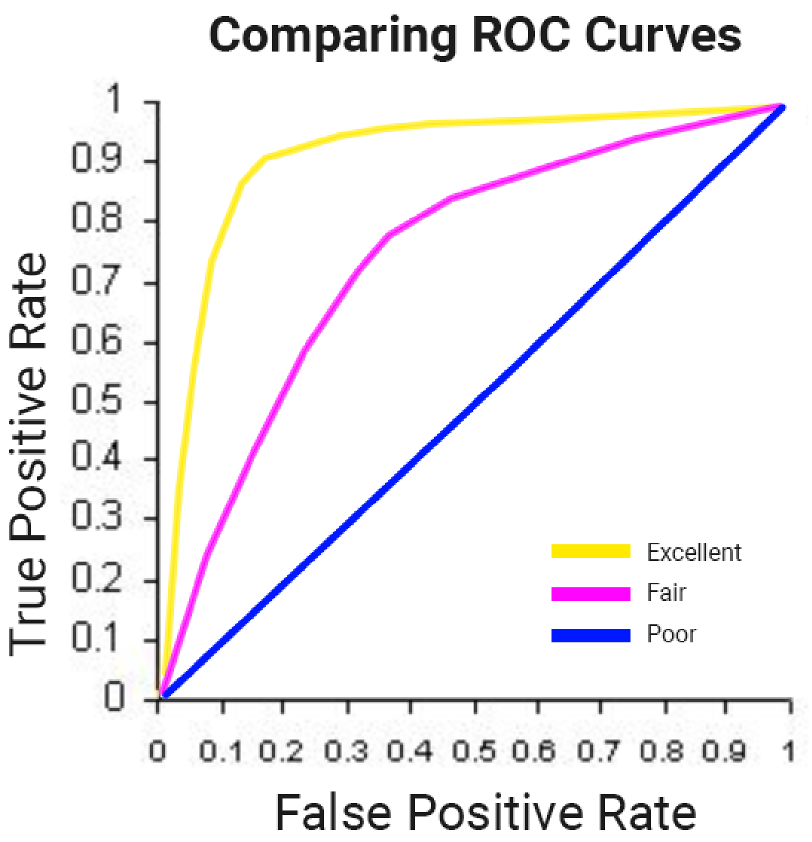
Field name	Type	Mode	Description
bounces	INT64	NULLABLE	
time_on_site	INT64	NULLABLE	

Task 5. Evaluate classification model performance

Select your performance criteria

For classification problems in ML, you want to minimize the False Positive Rate (predict that the user will return and purchase and they don't) and maximize the True Positive Rate (predict that the user will return and purchase and they do).

This relationship is visualized with a ROC (Receiver Operating Characteristic) curve like the one shown here, where you try to maximize the area under the curve or AUC:



In BigQuery ML, **roc_auc** is simply a queryable field when evaluating your trained ML model.

- Now that training is complete, you can evaluate how well the model performs by running this query using `ML.EVALUATE`:

```
SELECT
  roc_auc,
  CASE
    WHEN roc_auc > .9 THEN 'good'
    WHEN roc_auc > .8 THEN 'fair'
    WHEN roc_auc > .7 THEN 'not great'
  ELSE 'poor' END AS model_quality
FROM
  ML.EVALUATE(MODEL ecommerce.classification_model,  (
SELECT
  * EXCEPT(fullVisitorId)
FROM
  # features
  (SELECT
    fullVisitorId,
    IFNULL(totals.bounces, 0) AS bounces,
    IFNULL(totals.timeOnSite, 0) AS time_on_site
  FROM
    `data-to-insights.ecommerce.web_analytics`
  WHERE
    totals.newVisits = 1
    AND date BETWEEN '20170501' AND '20170630') # eval on 2 months
  JOIN
  (SELECT
    fullvisitorid,
    IF(COUNTIF(totals.transactions > 0 AND totals.newVisits IS NULL) > 0,
1, 0) AS will_buy_on_return_visit
  FROM
      `data-to-insights.ecommerce.web_analytics`
  GROUP BY fullvisitorid)
  USING (fullVisitorId)
));
```

You should see the following result:

Row	roc_auc	model_quality
1	0.724588	not great

After evaluating your model you get a **roc_auc** of 0.72, which shows that the model has not great predictive power. Since the goal is to get the area under the curve as close to 1.0 as possible, there is room for improvement.

Task 6. Improve model performance with feature engineering

As was hinted at earlier, there are many more features in the dataset that may help the model better understand the relationship between a visitor's first session and the likelihood that they will purchase on a subsequent visit.

Add some new features and create a second machine learning model called `classification_model_2`:

- How far the visitor got in the checkout process on their first visit
- Where the visitor came from (traffic source: organic search, referring site etc.)
- Device category (mobile, tablet, desktop)
- Geographic information (country)

1. Create this second model by running the below query:

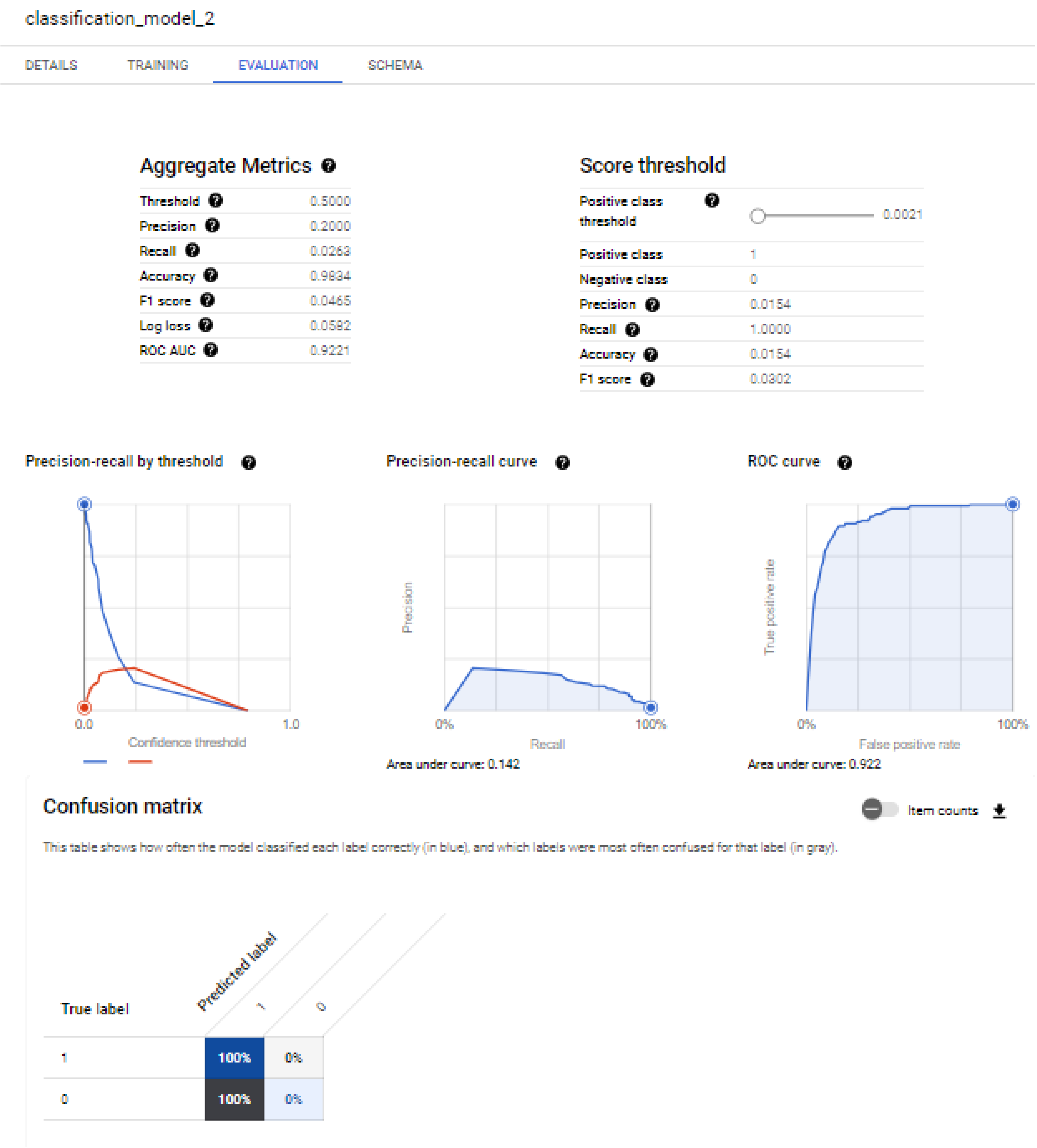
```
CREATE OR REPLACE MODEL `ecommerce.classification_model_2`
OPTIONS
  (model_type='logistic_reg', labels = ['will_buy_on_return_visit']) AS
WITH all_visitor_stats AS (
SELECT
  fullvisitorid,
  IF(COUNTIF(totals.transactions > 0 AND totals.newVisits IS NULL) > 0, 1,
0) AS will_buy_on_return_visit
  FROM `data-to-insights.ecommerce.web_analytics`
  GROUP BY fullvisitorid
)
# add in new features
SELECT * EXCEPT(unique_session_id) FROM (
  SELECT
      CONCAT(fullvisitorid, CAST(visitId AS STRING)) AS unique_session_id,
      # labels
      will_buy_on_return_visit,
      MAX(CAST(h.eCommerceAction.action_type AS INT64)) AS
latest_ecommerce_progress,
      # behavior on the site
      IFNULL(totals.bounces, 0) AS bounces,
      IFNULL(totals.timeOnSite, 0) AS time_on_site,
      totals.pageviews,
      # where the visitor came from
      trafficSource.source,
      trafficSource.medium,
      channelGrouping,
      # mobile or desktop
      device.deviceCategory,
      # geographic
      IFNULL(geoNetwork.country, "") AS country
  FROM `data-to-insights.ecommerce.web_analytics`,
     UNNEST(hits) AS h
    JOIN all_visitor_stats USING(fullvisitorid)
  WHERE 1=1
    # only predict for new visits
    AND totals.newVisits = 1
    AND date BETWEEN '20160801' AND '20170430' # train 9 months
  GROUP BY
  unique_session_id,
  will_buy_on_return_visit,
  bounces,
  time_on_site,
  totals.pageviews,
  trafficSource.source,
  trafficSource.medium,
  channelGrouping,
  device.deviceCategory,
  country
);
```

Note: You are still training on the same first 9 months of data, even with this new model. It's important to have the same training dataset so you can be certain a better model output is attributable to better input features and not new or different training data.

A key new feature that was added to the training dataset query is the maximum checkout progress each visitor reached in their session, which is recorded in the field `hits.eCommerceAction.action_type`. If you search for that field in the field definitions you will see the field mapping of 6 = Completed Purchase.

As an aside, the web analytics dataset has nested and repeated fields like ARRAYS which need to be broken apart into separate rows in your dataset. This is accomplished by using the UNNEST() function, which you can see in the above query.

Wait for the new model to finish training (5-10 minutes).



classification_model_2

Aggregate Metrics

Threshold	0.5000
Precision	0.2000
Recall	0.0263
Accuracy	0.9834
F1 score	0.0465
Log loss	0.0582
ROC AUC	0.9221

Score threshold

Positive class threshold	0.0021
Positive class	1
Negative class	0
Precision	0.0154
Recall	1.0000
Accuracy	0.0154
F1 score	0.0302

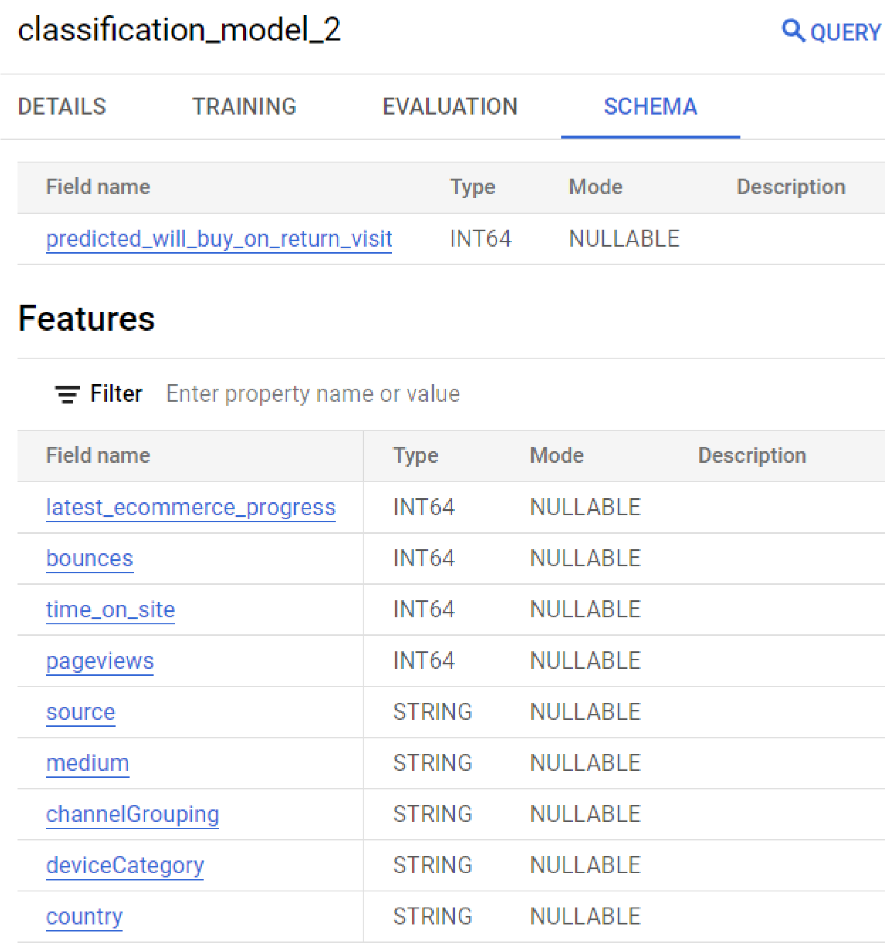
Area under curve: 0.142

Area under curve: 0.922

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).

True label	Predicted label 1	0
1	100%	0%
0	100%	0%



classification_model_2

Field name	Type	Mode	Description
predicted_will_buy_on_return_visit	INT64	NULLABLE	

Features

Field name	Type	Mode	Description
latest_ecommerce_progress	INT64	NULLABLE	
bounces	INT64	NULLABLE	
time_on_site	INT64	NULLABLE	
pageviews	INT64	NULLABLE	
source	STRING	NULLABLE	
medium	STRING	NULLABLE	
channelGrouping	STRING	NULLABLE	
deviceCategory	STRING	NULLABLE	
country	STRING	NULLABLE	

2. Evaluate this new model to see if there is better predictive power by running the below query:

```
#standardSQL
SELECT
  roc_auc,
  CASE
    WHEN roc_auc > .9 THEN 'good'
    WHEN roc_auc > .8 THEN 'fair'
    WHEN roc_auc > .7 THEN 'not great'
  ELSE 'poor' END AS model_quality
FROM
  ML.EVALUATE(MODEL ecommerce.classification_model_2,  (
WITH all_visitor_stats AS (
SELECT
  fullvisitorid,
  IF(COUNTIF(totals.transactions > 0 AND totals.newVisits IS NULL) > 0, 1,
0) AS will_buy_on_return_visit
  FROM `data-to-insights.ecommerce.web_analytics`
  GROUP BY fullvisitorid
)
# add in new features
SELECT * EXCEPT(unique_session_id) FROM (
  SELECT
      CONCAT(fullvisitorid, CAST(visitId AS STRING)) AS unique_session_id,
      # labels
      will_buy_on_return_visit,
      MAX(CAST(h.eCommerceAction.action_type AS INT64)) AS
latest_ecommerce_progress,
      # behavior on the site
      IFNULL(totals.bounces, 0) AS bounces,
      IFNULL(totals.timeOnSite, 0) AS time_on_site,
      totals.pageviews,
      # where the visitor came from
      trafficSource.source,
      trafficSource.medium,
      channelGrouping,
      # mobile or desktop
      device.deviceCategory,
      # geographic
      IFNULL(geoNetwork.country, "") AS country
  FROM `data-to-insights.ecommerce.web_analytics`,
     UNNEST(hits) AS h
    JOIN all_visitor_stats USING(fullvisitorid)
  WHERE 1=1
    # only predict for new visits
    AND totals.newVisits = 1
    AND date BETWEEN '20170501' AND '20170630' # eval 2 months
  GROUP BY
  unique_session_id,
  will_buy_on_return_visit,
  bounces,
  time_on_site,
  totals.pageviews,
  trafficSource.source,
  trafficSource.medium,
  channelGrouping,
  device.deviceCategory,
  country
)
));
```

(Output)

Row	roc_auc	model_quality
1	0.910382	good

With this new model you now get a **roc_auc** of 0.91 which is significantly better than the first model.

Now that you have a trained model, time to make some predictions.

Task 7. Predict which new visitors will come back and purchase

Next you will write a query to predict which new visitors will come back and make a purchase.

- Run the prediction query below which uses the improved classification model to predict the probability that a first-time visitor to the Google Merchandise Store will make a purchase in a later visit:

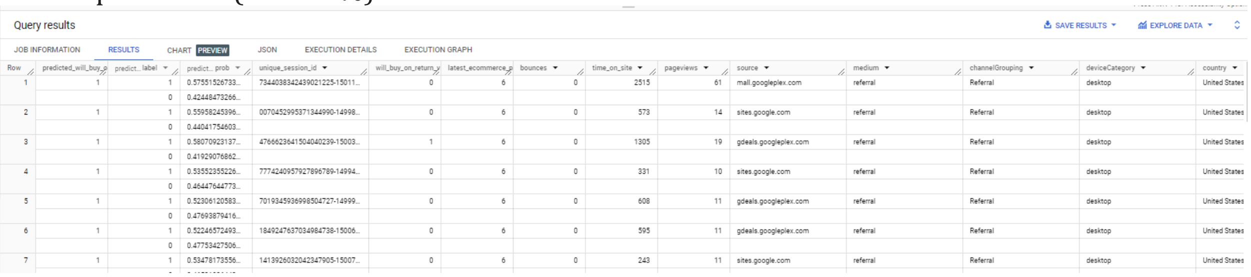
```
SELECT
*
FROM
  ml.PREDICT(MODEL `ecommerce.classification_model_2`,
   (
WITH all_visitor_stats AS (
SELECT
  fullvisitorid,
  IF(COUNTIF(totals.transactions > 0 AND totals.newVisits IS NULL) > 0, 1,
0) AS will_buy_on_return_visit
  FROM `data-to-insights.ecommerce.web_analytics`
  GROUP BY fullvisitorid
)
  SELECT
      CONCAT(fullvisitorid, '-',CAST(visitId AS STRING)) AS
unique_session_id,
      # labels
      will_buy_on_return_visit,
      MAX(CAST(h.eCommerceAction.action_type AS INT64)) AS
latest_ecommerce_progress,
      # behavior on the site
      IFNULL(totals.bounces, 0) AS bounces,
      IFNULL(totals.timeOnSite, 0) AS time_on_site,
      totals.pageviews,
      # where the visitor came from
      trafficSource.source,
      trafficSource.medium,
      channelGrouping,
      # mobile or desktop
      device.deviceCategory,
      # geographic
      IFNULL(geoNetwork.country, "") AS country
  FROM `data-to-insights.ecommerce.web_analytics`,
     UNNEST(hits) AS h
    JOIN all_visitor_stats USING(fullvisitorid)
  WHERE
    # only predict for new visits
    totals.newVisits = 1
    AND date BETWEEN '20170701' AND '20170801' # test 1 month
  GROUP BY
  unique_session_id,
  will_buy_on_return_visit,
  bounces,
  time_on_site,
  totals.pageviews,
  trafficSource.source,
  trafficSource.medium,
  channelGrouping,
  device.deviceCategory,
  country
)
)
ORDER BY
  predicted_will_buy_on_return_visit DESC;
```

The predictions are made in the last 1 month (out of 12 months) of the dataset.

Your model will now output the predictions it has for those July 2017 ecommerce sessions. You can see three newly added fields:

- predicted_will_buy_on_return_visit: whether the model thinks the visitor will buy later (1 = yes)
- predicted_will_buy_on_return_visit_probs.label: the binary classifier for yes / no

- predicted_will_buy_on_return_visit_probs.prob: the confidence the model has in it's prediction (1 = 100%)



Row	predicted_will_buy_o	predict... label	predict... prob	unique_session_id	will_buy_on_return_v	latest_ecommerce_p	bounces	time_on_site	pageviews	source	medium	channelGrouping	deviceCategory	country
1	1	1	0.57551526733...	7344038342439021225-15011...	0	6	0	2515	61	mall.googleplex.com	referral	Referral	desktop	United States
		0	0.42448473266...											
2	1	1	0.55958245396...	0070452995371344990-14998...	0	6	0	573	14	sites.google.com	referral	Referral	desktop	United States
		0	0.44041754603...											
3	1	1	0.58070923137...	4766623641504040239-15003...	1	6	0	1305	19	gdeals.googleplex.com	referral	Referral	desktop	United States
		0	0.41929076862...											
4	1	1	0.53552355226...	7774240957927896789-14994...	0	6	0	331	10	sites.google.com	referral	Referral	desktop	United States
		0	0.46447644773...											
5	1	1	0.52306120583...	7019345936998504727-14999...	0	6	0	608	11	gdeals.googleplex.com	referral	Referral	desktop	United States
		0	0.47693879416...											
6	1	1	0.52246572493...	1849247637034984738-15006...	0	6	0	595	11	gdeals.googleplex.com	referral	Referral	desktop	United States
		0	0.47753427506...											
7	1	1	0.53478173556...	1413926032042347905-15007...	0	6	0	243	11	sites.google.com	referral	Referral	desktop	United States

Results

- Of the top 6% of first-time visitors (sorted in decreasing order of predicted probability), more than 6% make a purchase in a later visit.
- These users represent nearly 50% of all first-time visitors who make a purchase in a later visit.
- Overall, only 0.7% of first-time visitors make a purchase in a later visit.
- Targeting the top 6% of first-time increases marketing ROI by 9x vs targeting them all!

Additional information

roc_auc is just one of the performance metrics available during model evaluation. Also available are accuracy, precision, and recall. Knowing which performance metric to rely on is highly dependent on what your overall objective or goal is.

Congratulations!

You created a machine learning model using just SQL.

Challenge

Summary

In the previous two tasks you saw the power of feature engineering at work in improving our models performance. However, we still may be able to improve our performance by exploring other model types. For classification problems, BigQuery ML also supports the following model types:

- Deep Neural Networks.
- Boosted Decision Trees (XGBoost).
- AutoML Tables Models.
- Importing Custom TensorFlow Models.

Task

Though our linear classification (logistic regression) model performed well after feature engineering, it may be too simple of a model to fully capture the relationship between the features and the label. Using the same dataset and labels as you did in Task 6 to create the model ecommerce.classification_model_2, your challenge is to create a XGBoost Classifier.

Note: Hint : Use following options for Boosted_Tree_Classifier:

1. L2_reg = 0.1

2. num_parallel_tree = 8

3. max_tree_depth = 10

You may need to look at the documentation linked above to see the exact syntax. The model will take around 7 minutes to train. The solution can be found in the solution section below if you need help writing the query. pa

Solution:

This is the solution that you require in order to create a XGBoost Classifier:

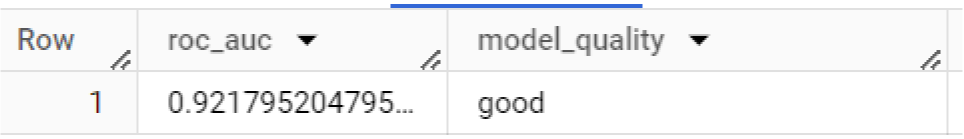
```
CREATE OR REPLACE MODEL `ecommerce.classification_model_3`
        OPTIONS
          (model_type='BOOSTED_TREE_CLASSIFIER' , l2_reg = 0.1,
num_parallel_tree = 8, max_tree_depth = 10,
              labels = ['will_buy_on_return_visit']) AS
        WITH all_visitor_stats AS (
        SELECT
          fullvisitorid,
          IF(COUNTIF(totals.transactions > 0 AND totals.newVisits IS
NULL) > 0, 1, 0) AS will_buy_on_return_visit
          FROM `data-to-insights.ecommerce.web_analytics`
          GROUP BY fullvisitorid
        )
        # add in new features
        SELECT * EXCEPT(unique_session_id) FROM (
          SELECT
              CONCAT(fullvisitorid, CAST(visitId AS STRING)) AS
unique_session_id,
              # labels
              will_buy_on_return_visit,
              MAX(CAST(h.eCommerceAction.action_type AS INT64)) AS
latest_ecommerce_progress,
              # behavior on the site
              IFNULL(totals.bounces, 0) AS bounces,
              IFNULL(totals.timeOnSite, 0) AS time_on_site,
              totals.pageviews,
              # where the visitor came from
              trafficSource.source,
              trafficSource.medium,
              channelGrouping,
              # mobile or desktop
              device.deviceCategory,
              # geographic
              IFNULL(geoNetwork.country, "") AS country
          FROM `data-to-insights.ecommerce.web_analytics`,
             UNNEST(hits) AS h
            JOIN all_visitor_stats USING(fullvisitorid)
          WHERE 1=1
            # only predict for new visits
            AND totals.newVisits = 1
            AND date BETWEEN '20160801' AND '20170430' # train 9 months
          GROUP BY
          unique_session_id,
          will_buy_on_return_visit,
          bounces,
          time_on_site,
          totals.pageviews,
          trafficSource.source,
          trafficSource.medium,
          channelGrouping,
          device.deviceCategory,
          country
        );
```

Let us now evaluate our model and see how we did:

```
#standardSQL

SELECT
  roc_auc,
  CASE
    WHEN roc_auc > .9 THEN 'good'
    WHEN roc_auc > .8 THEN 'fair'
    WHEN roc_auc > .7 THEN 'not great'
  ELSE 'poor' END AS model_quality
FROM
  ML.EVALUATE(MODEL ecommerce.classification_model_3,  (
WITH all_visitor_stats AS (
SELECT
  fullvisitorid,
  IF(COUNTIF(totals.transactions > 0 AND totals.newVisits IS NULL) > 0, 1,
0) AS will_buy_on_return_visit
  FROM `data-to-insights.ecommerce.web_analytics`
  GROUP BY fullvisitorid
)
# add in new features
SELECT * EXCEPT(unique_session_id) FROM (
  SELECT
      CONCAT(fullvisitorid, CAST(visitId AS STRING)) AS unique_session_id,
      # labels
      will_buy_on_return_visit,
      MAX(CAST(h.eCommerceAction.action_type AS INT64)) AS
latest_ecommerce_progress,
      # behavior on the site
      IFNULL(totals.bounces, 0) AS bounces,
      IFNULL(totals.timeOnSite, 0) AS time_on_site,
      totals.pageviews,
      # where the visitor came from
      trafficSource.source,
      trafficSource.medium,
      channelGrouping,
      # mobile or desktop
      device.deviceCategory,
      # geographic
      IFNULL(geoNetwork.country, "") AS country
  FROM `data-to-insights.ecommerce.web_analytics`,
     UNNEST(hits) AS h
    JOIN all_visitor_stats USING(fullvisitorid)
  WHERE 1=1
    # only predict for new visits
    AND totals.newVisits = 1
    AND date BETWEEN '20170501' AND '20170630' # eval 2 months
  GROUP BY
  unique_session_id,
  will_buy_on_return_visit,
  bounces,
  time_on_site,
  totals.pageviews,
  trafficSource.source,
  trafficSource.medium,
  channelGrouping,
  device.deviceCategory,
  country
)
));
```

Our roc_auc has increased by about .02 to around .94!



Row	roc_auc ▼	model_quality ▼
1	0.921795204795...	good

Note : Your exact values will differ due to the randomness involved in the training process.

It’s a small change in the roc_auc, but note that since 1 is a perfect roc_auc, it gets more difficult to improve the metric the closer to 1 it gets.

This is a great example of how easy it is in BigQuery ML to try out different model types with different options to see how they perform. We were able to use a much more complex model type by only changing one line of SQL.

One may reasonably ask “Where did the choices for these options come from?”, and the answer is experimentation! When you are trying to find the best model type for your problems, then one has to experiment with different sets of options in a process known as hyperparameter tuning.

Let's finish up by generating predictions with our improved model and see how they compare to those we generated before. By using a **Boosted tree classifier model**, you can observe a slight improvement of 0.2 in our ROC AUC compared to the previous model. The query below will predict which new visitors will come back and make a purchase:

```
SELECT
*
FROM
  ml.PREDICT(MODEL `ecommerce.classification_model_3`,
   (
WITH all_visitor_stats AS (
SELECT
  fullvisitorid,
  IF(COUNTIF(totals.transactions > 0 AND totals.newVisits IS NULL) > 0, 1,
0) AS will_buy_on_return_visit
  FROM `data-to-insights.ecommerce.web_analytics`
  GROUP BY fullvisitorid
)
  SELECT
      CONCAT(fullvisitorid, '-',CAST(visitId AS STRING)) AS
unique_session_id,
      # labels
      will_buy_on_return_visit,
      MAX(CAST(h.eCommerceAction.action_type AS INT64)) AS
latest_ecommerce_progress,
      # behavior on the site
      IFNULL(totals.bounces, 0) AS bounces,
      IFNULL(totals.timeOnSite, 0) AS time_on_site,
      totals.pageviews,
      # where the visitor came from
      trafficSource.source,
      trafficSource.medium,
      channelGrouping,
      # mobile or desktop
      device.deviceCategory,
      # geographic
      IFNULL(geoNetwork.country, "") AS country
  FROM `data-to-insights.ecommerce.web_analytics`,
     UNNEST(hits) AS h
    JOIN all_visitor_stats USING(fullvisitorid)
  WHERE
    # only predict for new visits
    totals.newVisits = 1
    AND date BETWEEN '20170701' AND '20170801' # test 1 month
  GROUP BY
  unique_session_id,
  will_buy_on_return_visit,
  bounces,
  time_on_site,
  totals.pageviews,
  trafficSource.source,
  trafficSource.medium,
  channelGrouping,
  device.deviceCategory,
  country
)
)
ORDER BY
  predicted_will_buy_on_return_visit DESC;
```

Copied!

content_copy

The output now shows a classification model that can better predict the probability that a first-time visitor to the Google Merchandise Store will make a purchase in a later visit. By comparing the result above with the previous model shown in Task 7, you can see the confidence the model has in its predictions is more accurate when compared to the logistic_regression model type.

Save results to "Boosted tree classifier model results.csv"

End your lab

When you have completed your lab, click **End Lab**. Google Cloud Skills Boost removes the resources you've used and cleans the account for you.

You will be given an opportunity to rate the lab experience. Select the applicable number of stars, type a comment, and then click **Submit**.