



Predict Visitor Purchases with a Classification Model with BigQuery ML

1 hour 20 minutes

1 Credit



Rate Lab

Overview

[BigQuery Machine Learning](#) (BQML, product in beta) is a new feature in BigQuery where data analysts can create, train, evaluate, and predict with machine learning models with minimal coding.

There is a newly available [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 BQML
- Evaluate the performance of your machine learning model
- Predict and rank the probability that a visitor will make a purchase

Setup and requirements

Qwiklabs setup

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click Start Lab, shows how long Cloud resources will be made available to you.

This Qwiklabs hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access the Google Cloud Platform for the duration of the lab.

What you need

To complete this lab, you need:

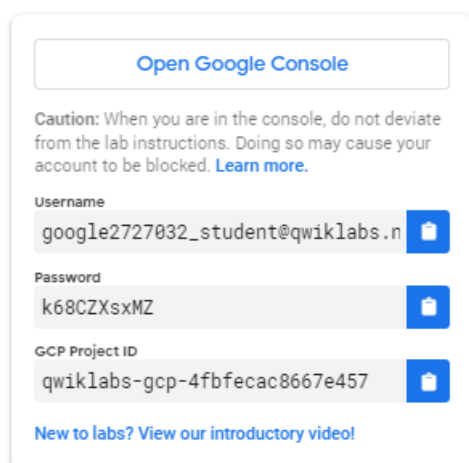
- Access to a standard internet browser (Chrome browser recommended).
- Time to complete the lab.

Note: If you already have your own personal GCP account or project, do not use it for this lab.

Google Cloud Platform Console

How to start your lab and sign in to the Console

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.



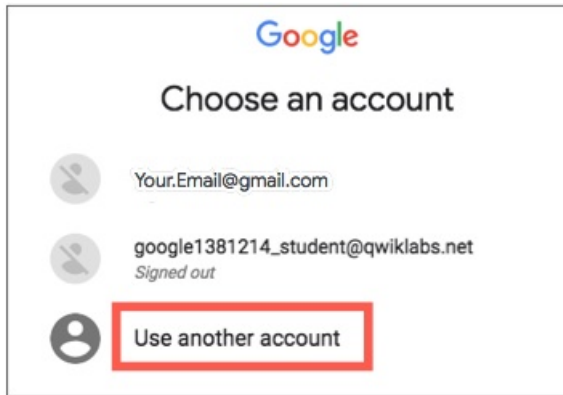
The screenshot shows a sign-in panel for the Google Cloud Platform Console. At the top is a button labeled "Open Google Console". Below it is a caution message: "Caution: When you are in the console, do not deviate from the lab instructions. Doing so may cause your account to be blocked. [Learn more.](#)". The panel contains three input fields, each with a copy icon to its right: "Username" with the value "google2727032_student@qwiklabs.n", "Password" with the value "k68CZsxMZ", and "GCP Project ID" with the value "qwiklabs-gcp-4fbfecac8667e457". At the bottom is a link: "New to labs? [View our introductory video!](#)".

2. Copy the username, and then click **Open Google Console**. The lab spins up resources,

and then opens another tab that shows the **Choose an account** page.

Tip: Open the tabs in separate windows, side-by-side.

3. On the Choose an account page, click **Use Another Account**.



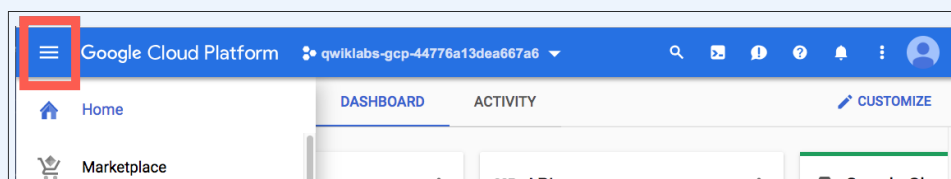
4. The Sign in page opens. Paste the username that you copied from the Connection Details panel. Then copy and paste the password.

Important: You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own GCP account, do not use it for this lab (avoids incurring charges).

5. Click through the subsequent pages:
 - Accept the terms and conditions.
 - Do not add recovery options or two-factor authentication (because this is a temporary account).
 - Do not sign up for free trials.

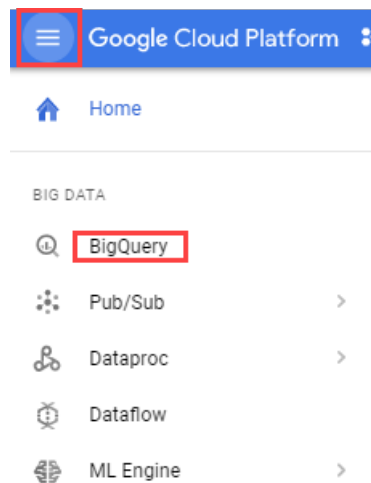
After a few moments, the GCP console opens in this tab.

Note: You can view the menu with a list of GCP Products and Services by clicking the **Navigation menu** at the top-left, next to "Google Cloud Platform".



Open BigQuery Console

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.

Click **Done**.

Access the course dataset

Once BigQuery is open, open on the below direct link in a new browser tab to bring the public **data-to-insights** project into your BigQuery projects panel:

- https://console.cloud.google.com/bigquery?p=data-to-insights&d=ecommerce&t=web_analytics&page=table

The field definitions for the **data-to-insights** ecommerce dataset are [here](#). Keep the link open in a new tab for reference.

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?

Click the **Query Editor** 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
```

Then click **Run**.

The result: 2.69%

Question: What are the top 5 selling products?

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?

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.

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.

Identify an objective

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 to ensure a conversion while they comparison shop between visits to your ecommerce site.

Select features and create your training dataset

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 [here](#) 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)

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:

```
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;
```

Then click **Run**.

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

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 come back and purchase. The value of building a ML model is to get the probability of future purchase based on the data gleaned about their first session.

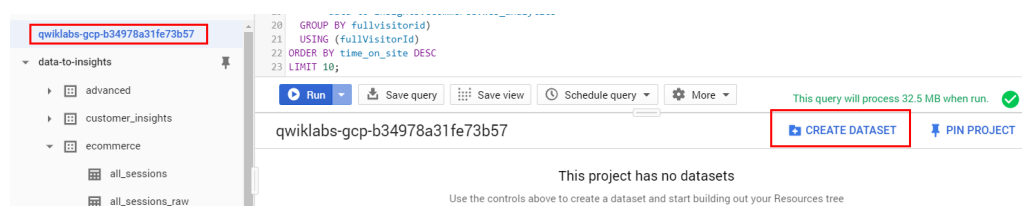
Question: Looking at the initial data results, do you think `time_on_site` and `bounces` will be a good indicator of whether the user will return and purchase or not?

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.

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 **Create Dataset**.



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

- For **Dataset ID**, type **ecommerce**.

- Leave the other values at their defaults.

3. Click **Create dataset**.

Select a BQML 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 time of writing, BQML supports the two listed above.

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)
;

```

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-xxxxxxx:ecommerce.classification_model".

Click **Go to model**.

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

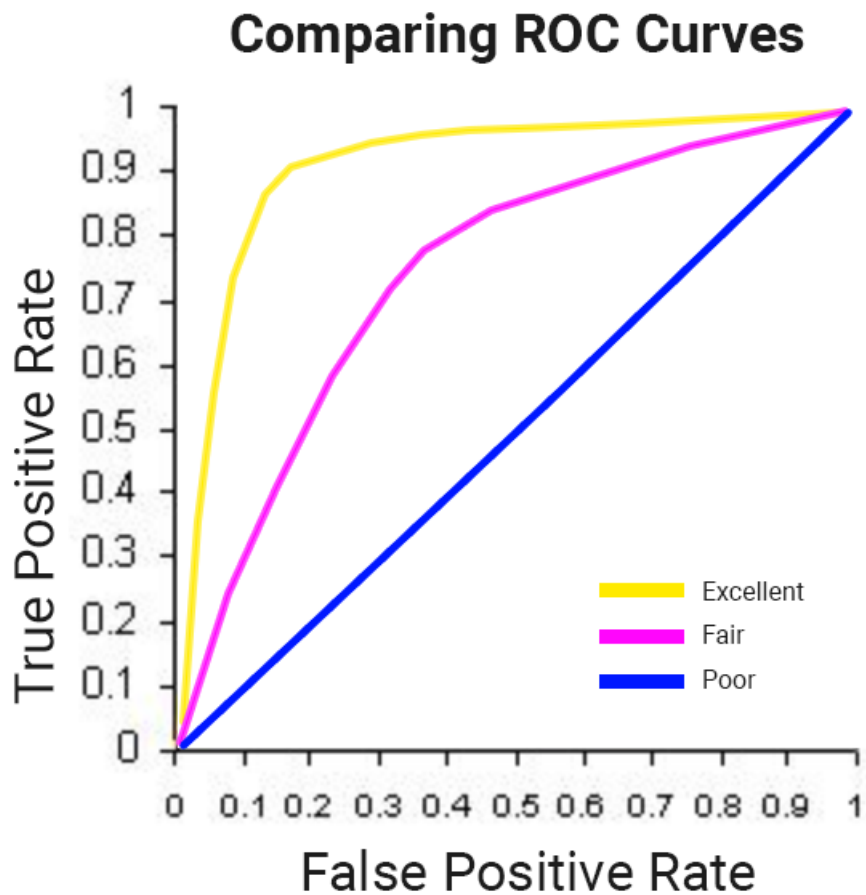
Next, you will evaluate the performance of the model against new unseen evaluation data.

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 BQML, **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 with 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 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.

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)

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).

Evaluate this new model to see if there is better predictive power:

```

#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

```

```

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.

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.

The prediction query below 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 on 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_on_return_visit	predicted_will_buy_on_return_visit_probs.label	predicted_will_buy_on_return_visit_probs.prob	unique_session_id	will_buy_on_return_visit
1	1	1	0.5063877442980596	1138389983344638566-1501537260	0
		0	0.49361225570194045		
2	1	1	0.6177436820092239	273427315284151453-1499785490	0
		0	0.3822563179907761		
3	1	1	0.5608212570496836	9756202106186308060-1499477518	1
		0	0.43917874295031645		
4	1	1	0.5496589421617243	3584433599055417628-1500581559	0
		0	0.4503410578382757		
5	1	1	0.6745622736082219	8633380214002553788-1499313933	0
		0	0.32543772639177815		
6	1	1	0.5439317028160215	450153187928705091-1501016343	0
		0	0.45606829718397845		

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.

End your lab

When you have completed your lab, click **End Lab**. Qwiklabs 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**.

The number of stars indicates the following:

- 1 star = Very dissatisfied
- 2 stars = Dissatisfied
- 3 stars = Neutral
- 4 stars = Satisfied
- 5 stars = Very satisfied

You can close the dialog box if you don't want to provide feedback.

For feedback, suggestions, or corrections, please use the **Support** tab.

Copyright 2019 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.