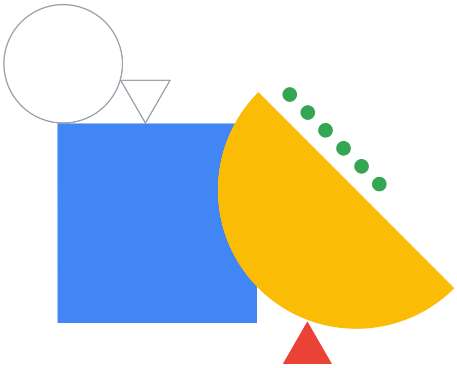
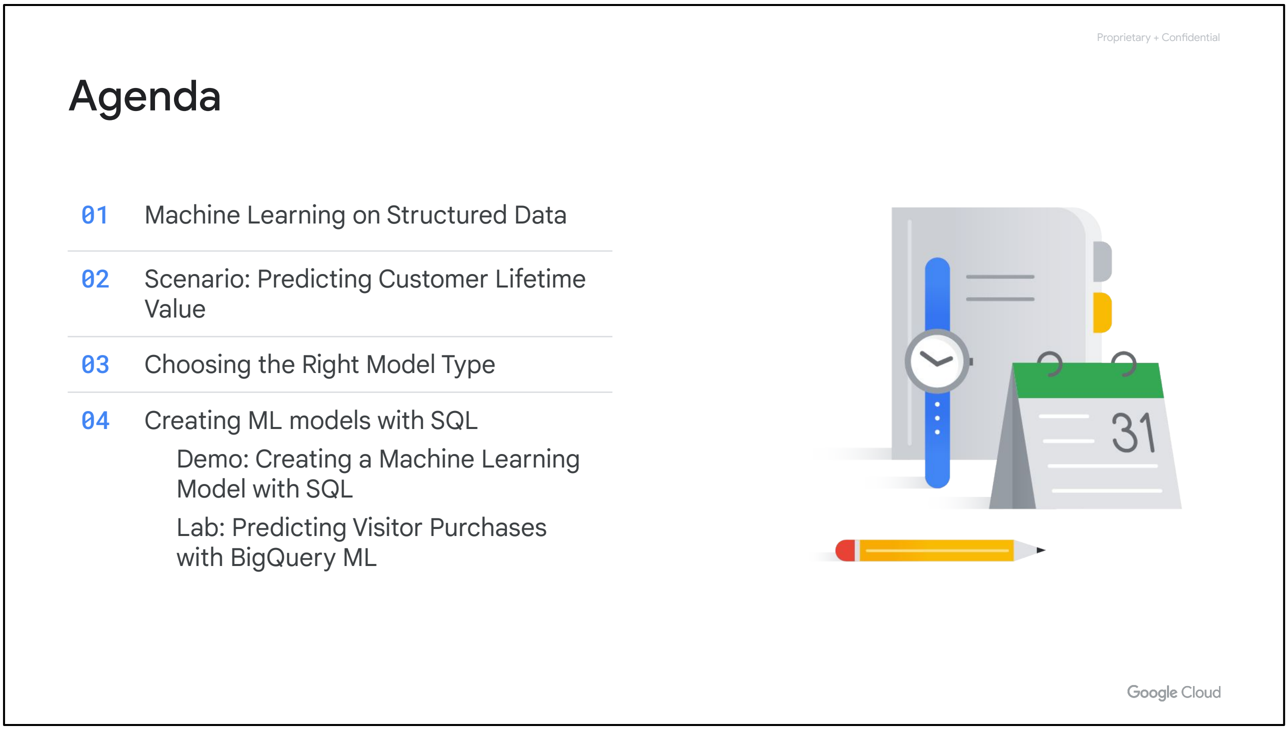
Google Cloud

Predicting Visitor Return Purchases with BigQuery ML





Agenda

01 Machine Learning on Structured Data

02 Scenario: Predicting Customer Lifetime Value

03 Choosing the Right Model Type

04 Creating ML models with SQL

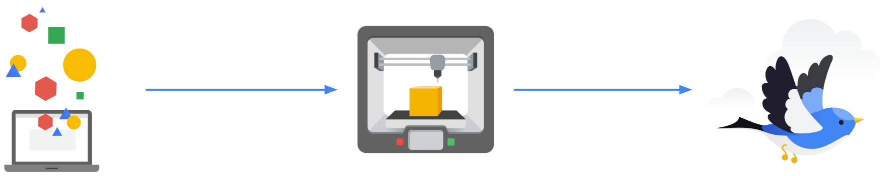
- Demo: Creating a Machine Learning Model with SQL
- Lab: Predicting Visitor Purchases with BigQuery ML

It’s time to revisit the exciting topic of machine learning. Before we jump into the code, we need to expand our machine learning foundation and cover the models and key terminology you need to know before you set off to build your models.

01

Machine Learning on Structured Data

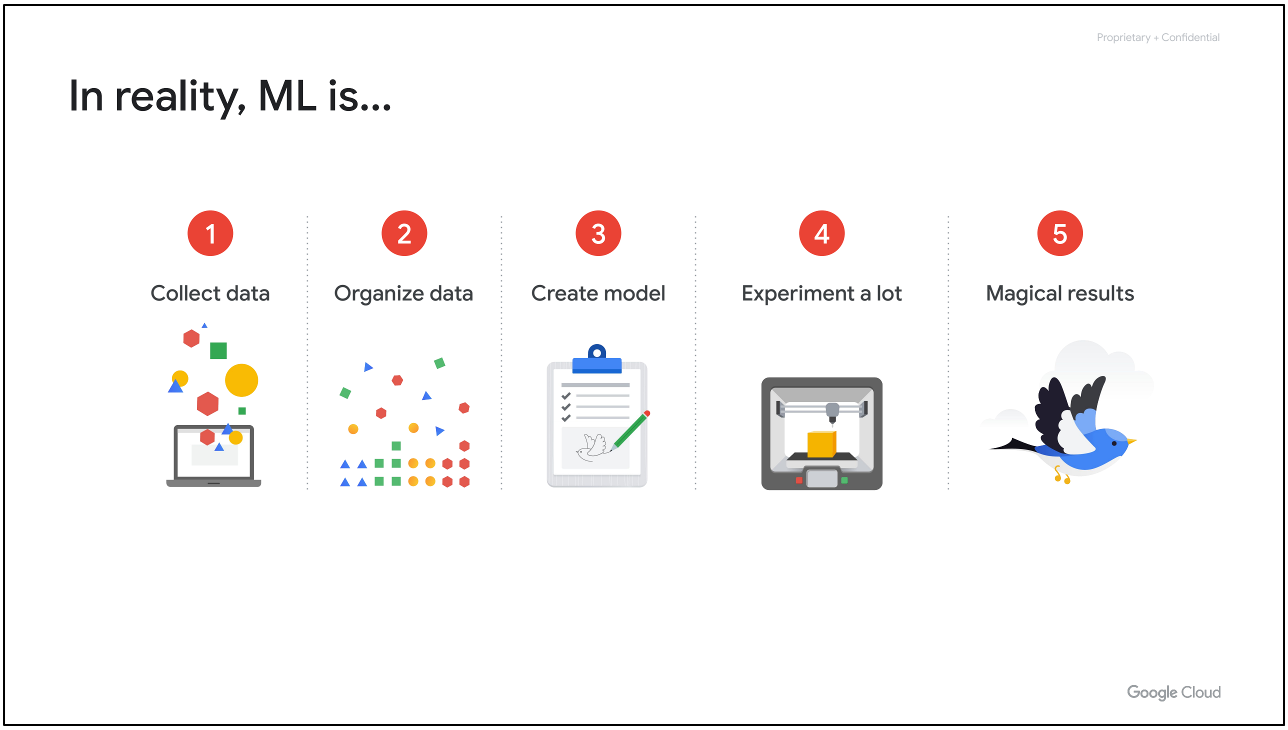
The popular imagination of what ML is



1 Loads of data

2 Complex mathematics in multidimensional spaces

3 Magical results



In reality, ML is...

1. Collect data
2. Organize data
3. Create model
4. Experiment a lot
5. Magical results

On Google Cloud, we can use:
Logging APIs, Pub/Sub, etc. and other real-time streaming to collect the data.
BigQuery, Dataflow and ML preprocessing SDK to organize the data [different types of organization].
TensorFlow to create the model.
Cloud ML to train, deploy the model.
The magic is still there …

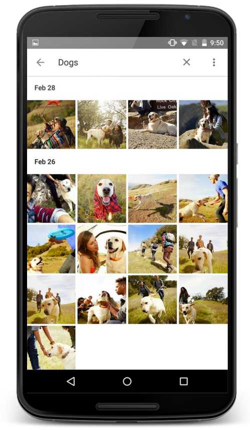
When you hear “AI or ML,” you probably think of:

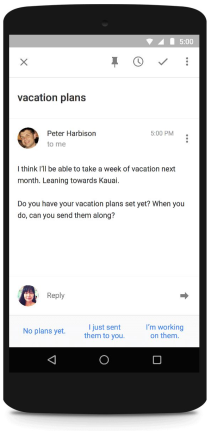
Image models

Sequence models

CNNs

RNNs





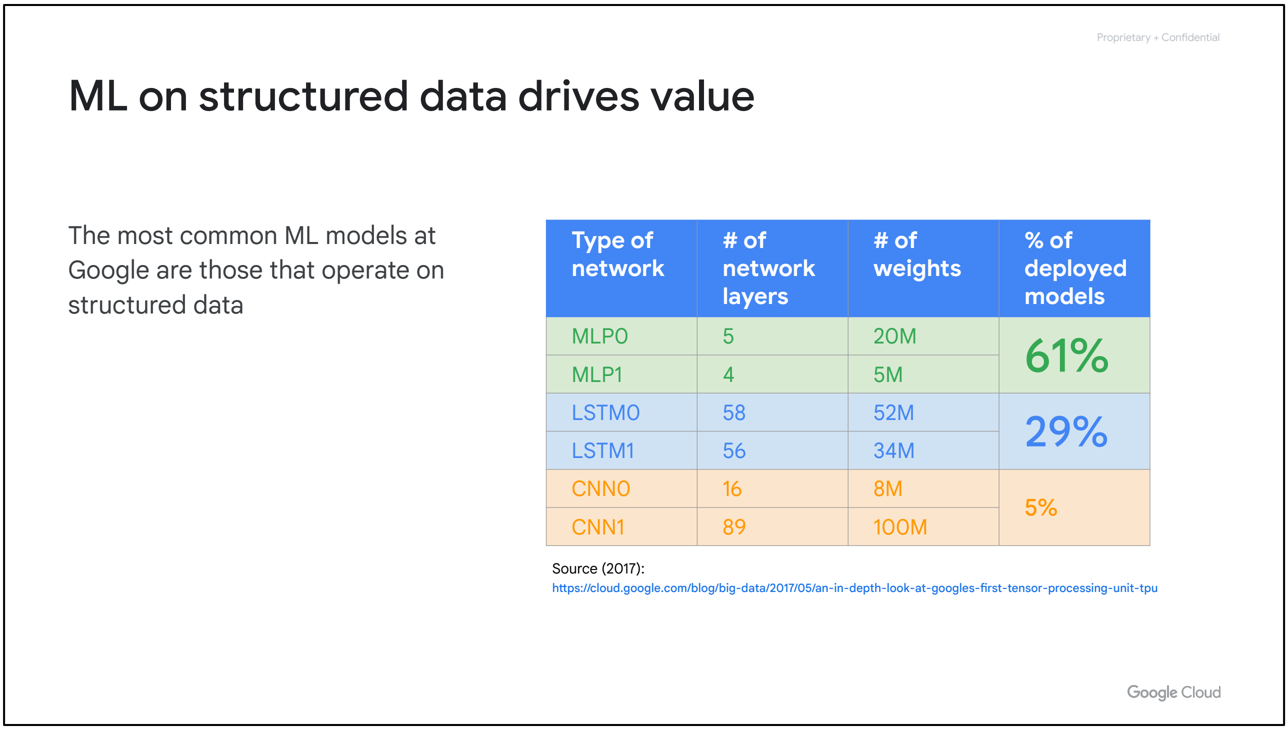
vacation plans

Peter Harbison
to me
5:00 PM

I think I'll be able to take a week of vacation next month. Leaning towards Kauai.

Do you have your vacation plans set yet? When you do, can you send them along?

When people think of AI or Machine Learning they generally think of the advanced models like those you saw earlier for Google Photos video stabilization and Smart Reply in Gmail. And yes, later in this course you built image models on unstructured datasets, but did you know that at Google …



ML on structured data drives value

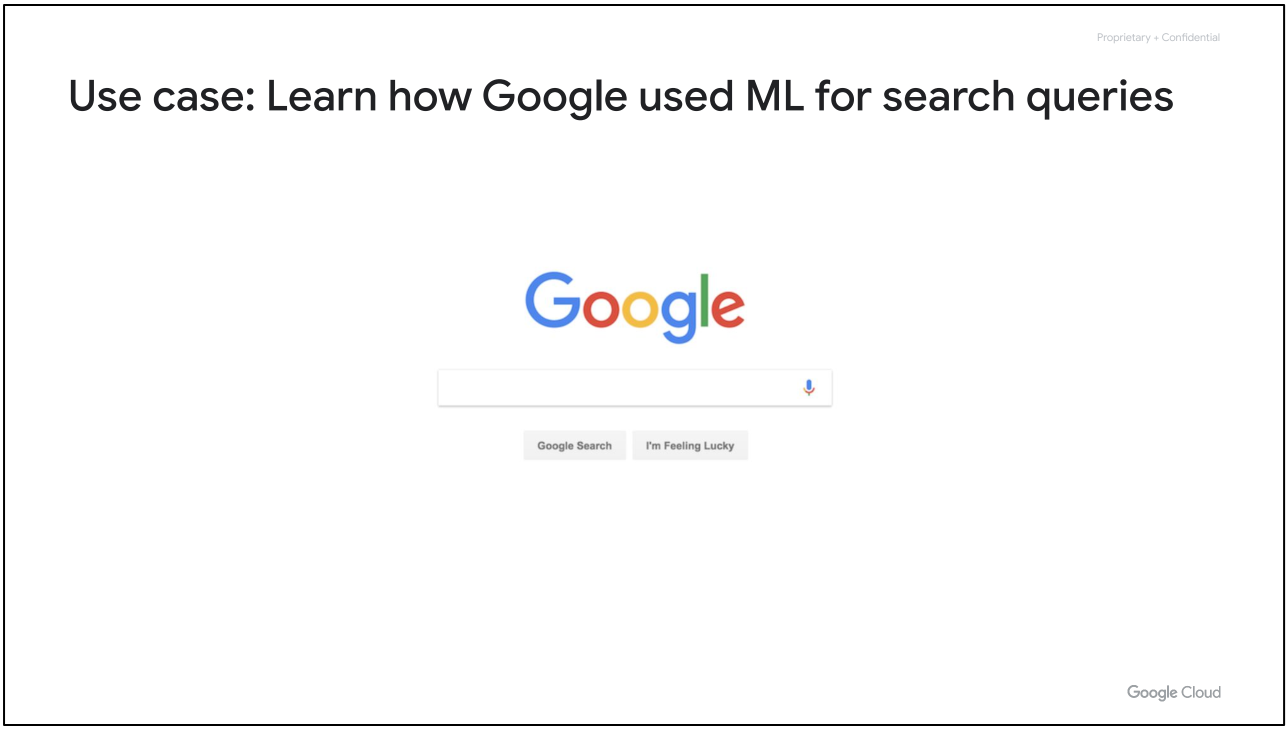
The most common ML models at Google are those that operate on structured data

Type of network	# of network layers	# of weights	% of deployed models
MLP0	5	20M	61%
MLP1	4	5M	
LSTM0	58	52M	29%
LSTM1	56	34M	
CNN0	16	8M	5%
CNN1	89	100M	

Source (2017):
https://cloud.google.com/blog/big-data/2017/05/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu

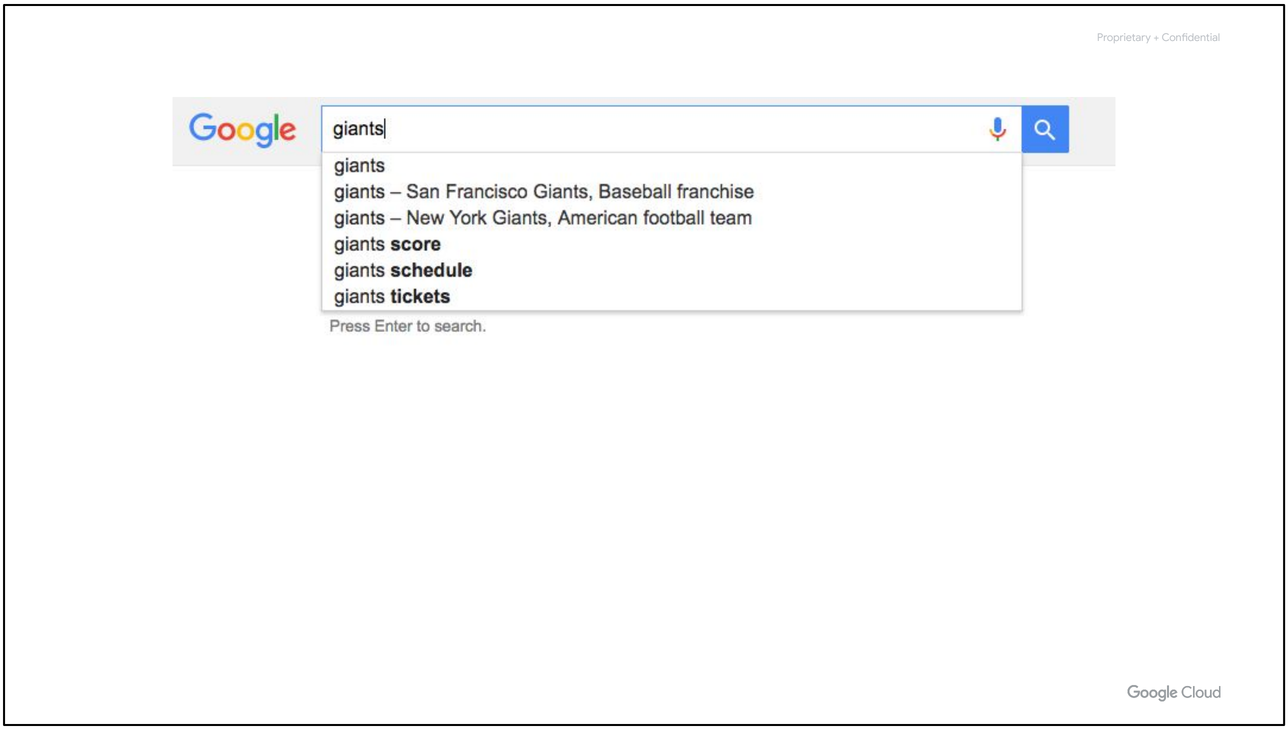
... the majority of models deployed are models that operate on structured data? These aren't your 50+ layer deep neural networks that play Starcraft or Chess. They're models built on rows and columns of data just like you've seen in BigQuery.

Source:
https://cloud.google.com/blog/big-data/2017/05/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu



Use case: Learn how Google used ML for search queries

Let’s extrapolate this to a real-world application. Let’s take google search for example



giants

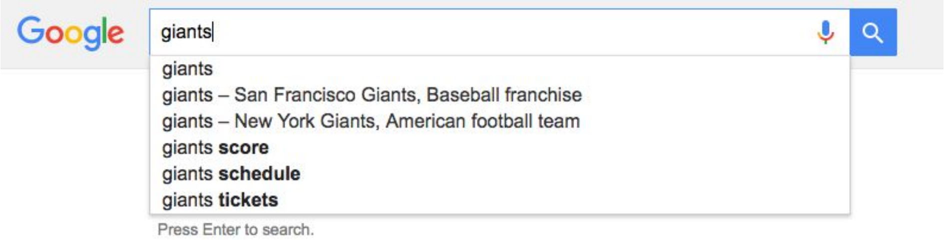
giants
giants – San Francisco Giants, Baseball franchise
giants – New York Giants, American football team
giants **score**
giants **schedule**
giants **tickets**

Press Enter to search.

Say you go to Google and search for “giants”

What should we show you as your results to make it most relevant for you?

Proprietary + Confidential



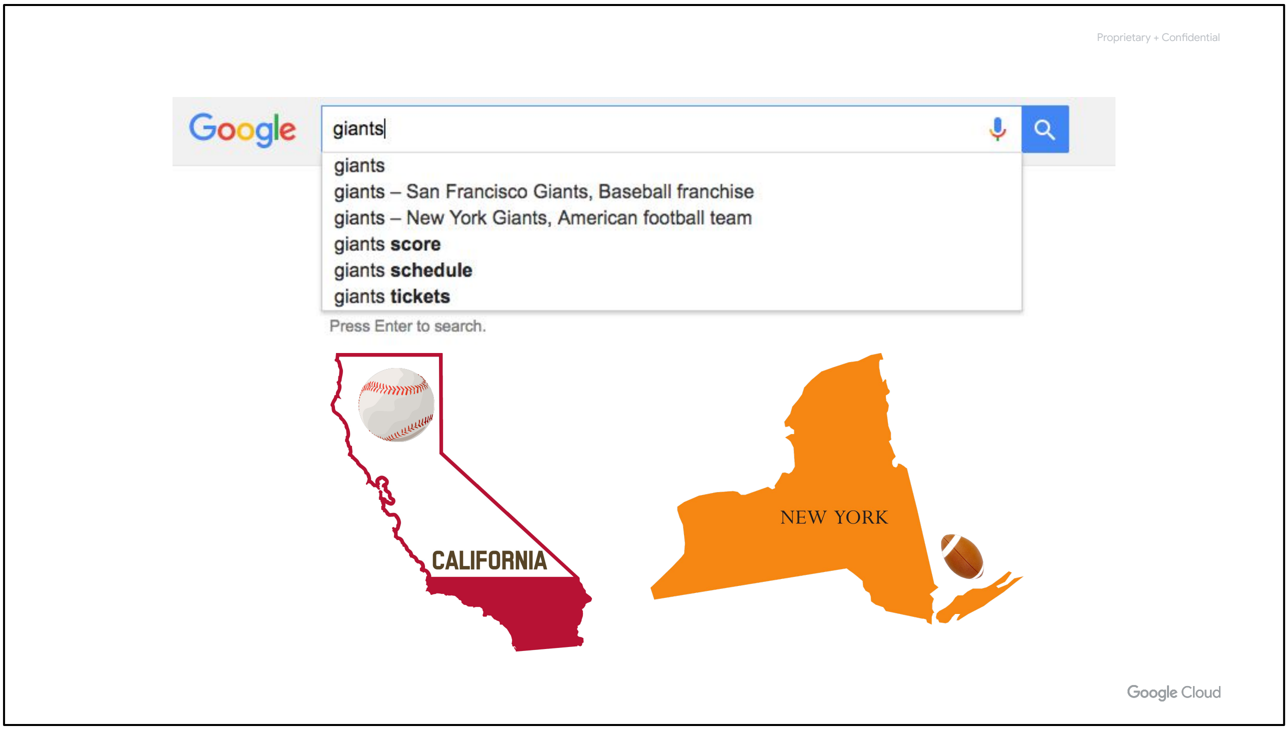
giants

giants
giants – San Francisco Giants, Baseball franchise
giants – New York Giants, American football team
giants **score**
giants **schedule**
giants **tickets**

Press Enter to search.

Google Cloud

If you’re in California, should we show results for the San Francisco Giants baseball team and local games nearby?

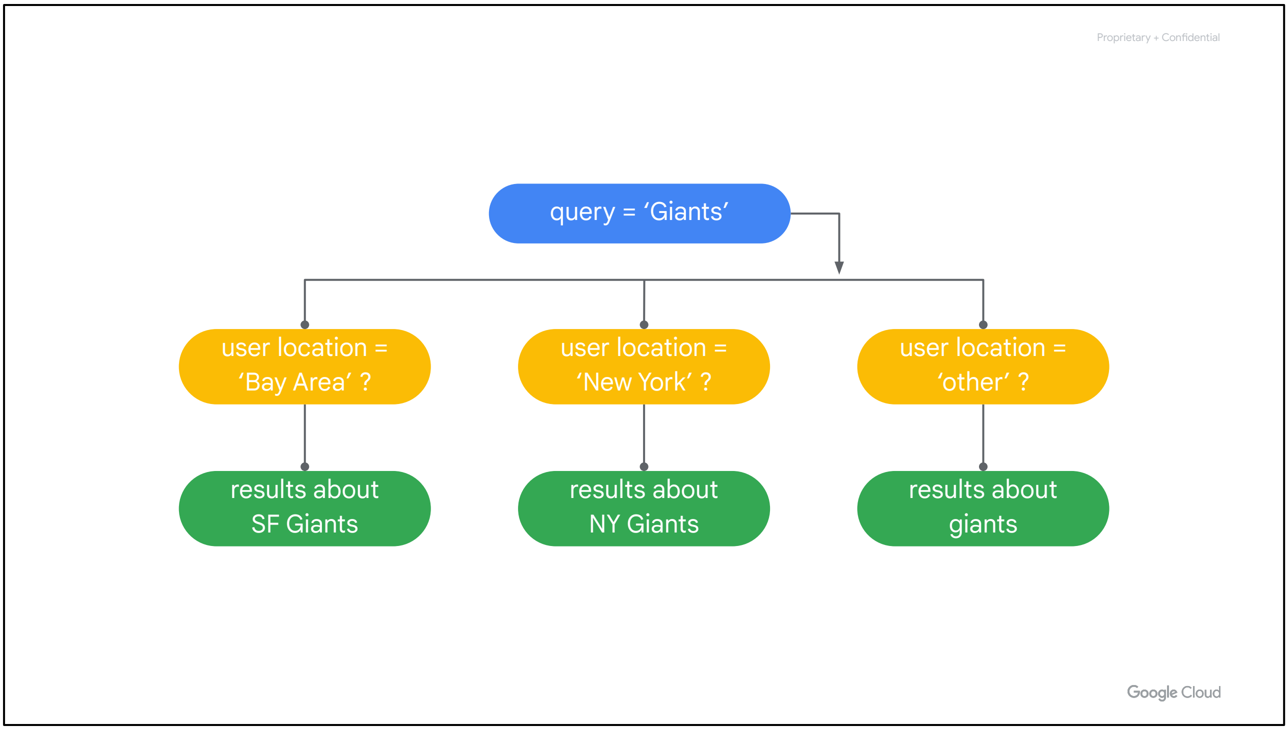


giants

giants
giants – San Francisco Giants, Baseball franchise
giants – New York Giants, American football team
giants **score**
giants **schedule**
giants **tickets**

Press Enter to search.

What about if you are based in New York, should we tailor the results to show the New York Giants football team instead as a rule?



Proprietary + Confidential

query = 'Giants'

user location = 'Bay Area' ?

user location = 'New York' ?

user location = 'other' ?

results about SF Giants

results about NY Giants

results about giants

Google Cloud

Up until a few years ago, this is how Google search worked.

There were a ton of rules that were part of the search engine code base to decide which sports team to show a user.

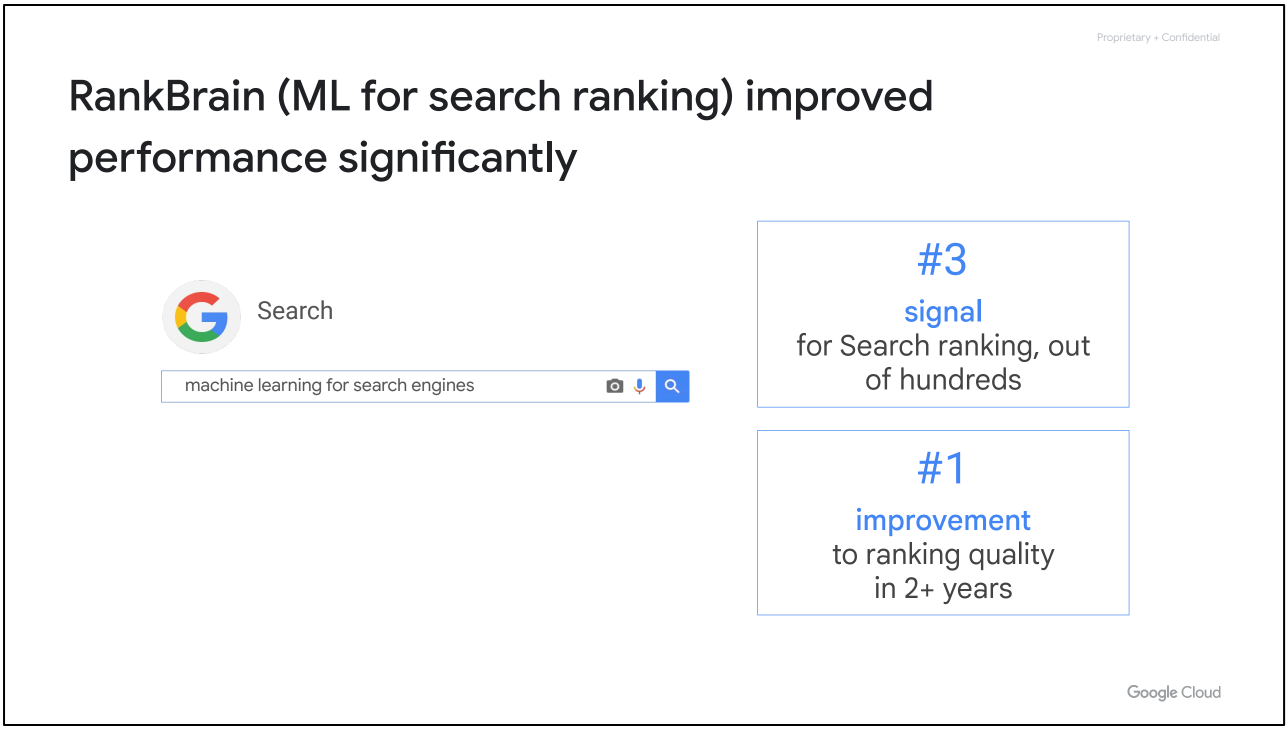
If the query is 'giants' AND the user is in the bay area, show them results about san francisco giants.
If the user is in the new york area, show them results about ny giants
If they are anywhere else, show them results about tall people.

Those of you who have worked with SQL before, just imagine how many CASE statements this would be and how hard it would be to maintain. And this is for just one query!

Multiply this by the large variety of queries that people make, where they make them from, what device they're on, and you can imagine how complex the whole codebase had become.
The codebase was getting unwieldy. Hand-coded rules are hard to maintain.

This is where ML comes into play. It scales much better because it requires no hardcoded rules and it's all automated.

Our dataset in this case is we know historically which people clicked on what links, why couldn't we train a ML model to provide input in the search ranking?



RankBrain (ML for search ranking) improved performance significantly

Search

machine learning for search engines

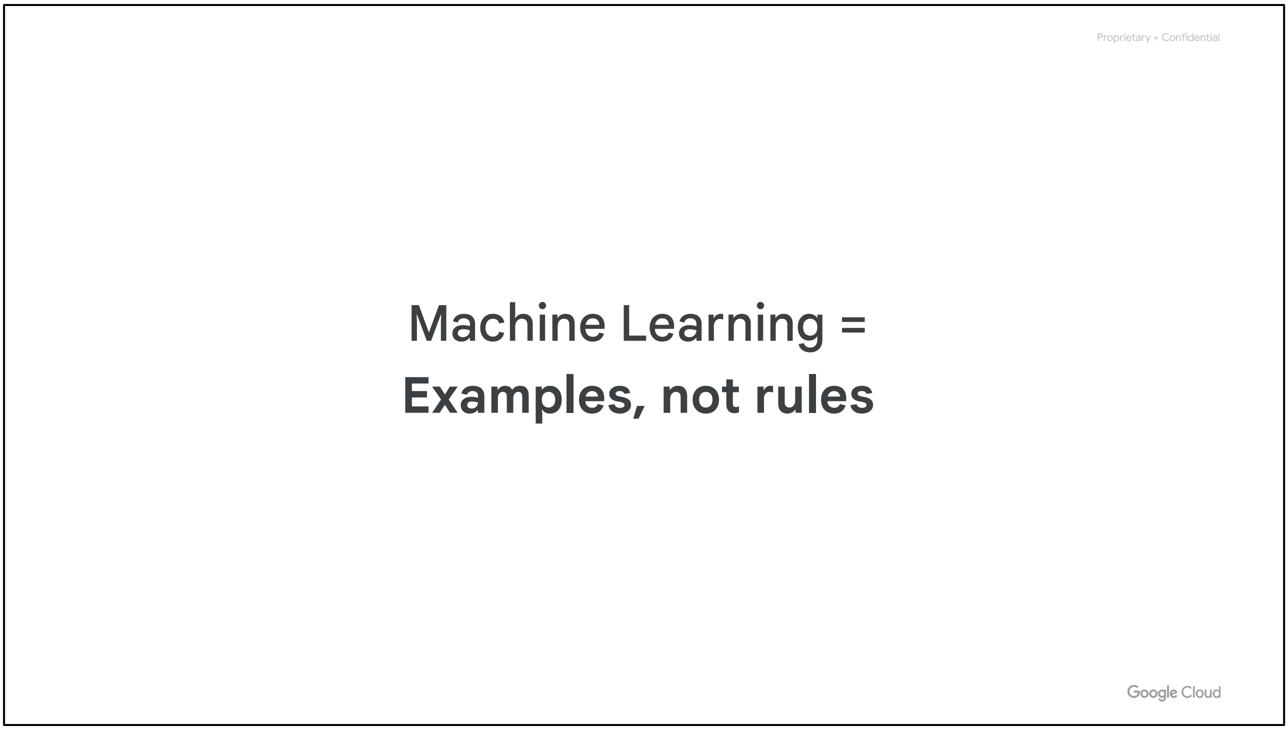
#3
signal
for Search ranking, out of hundreds

#1
improvement
to ranking quality in 2+ years

Google Cloud

And that’s exactly what Google itself has done internally with a deep learning ML model called RankBrain. After rolling it out, the quality of search ranking results improved dramatically with the signal coming from RankBrain becoming one of the top 3 for influencing how results are ranked

If you’re interested, I’ll provide a link where you can read more about it.



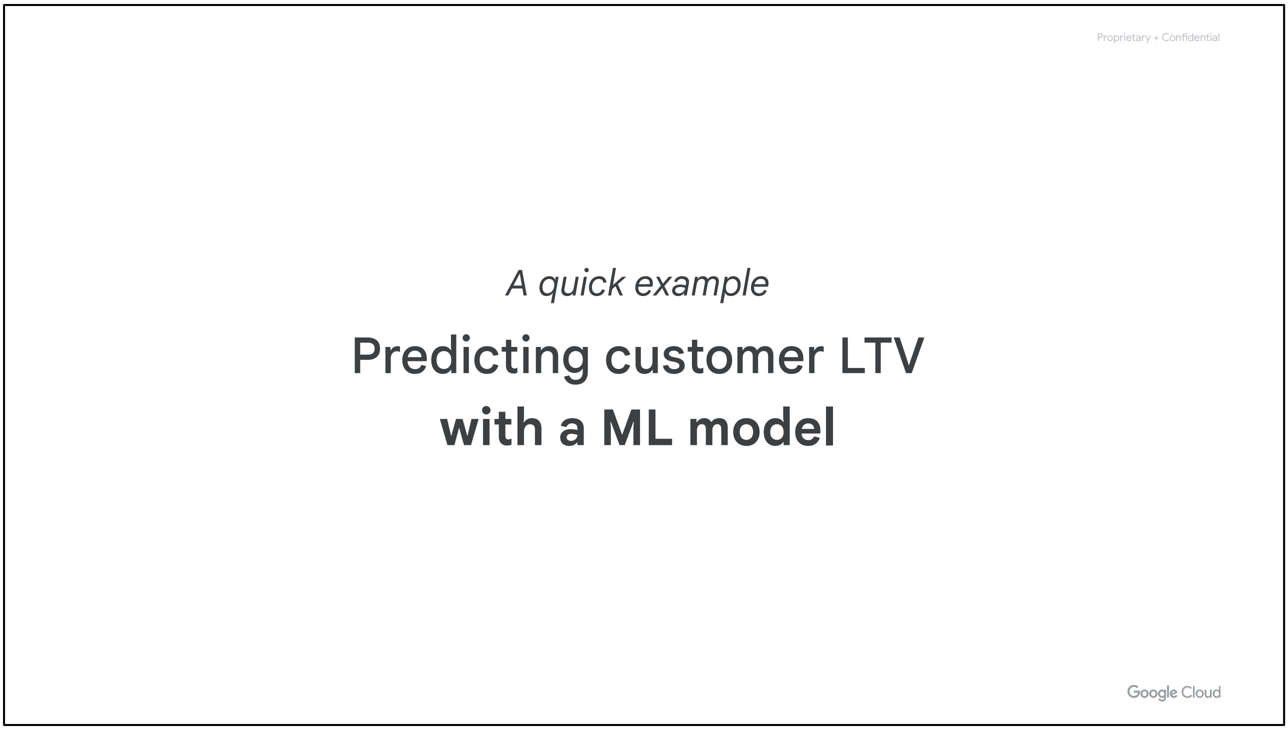
Machine Learning =
Examples, not rules

We will revisit machine learning in greater depth in each of the modules in this course. For now just remember that we want to teach the computer using examples, not with rules.

Any business application where you have those long SWITCH or CASE statements or IF THEN logic manually coded AND you have a history of good labeled data is a possible application for Machine Learning.

02

Scenario: Predicting Customer Lifetime Value

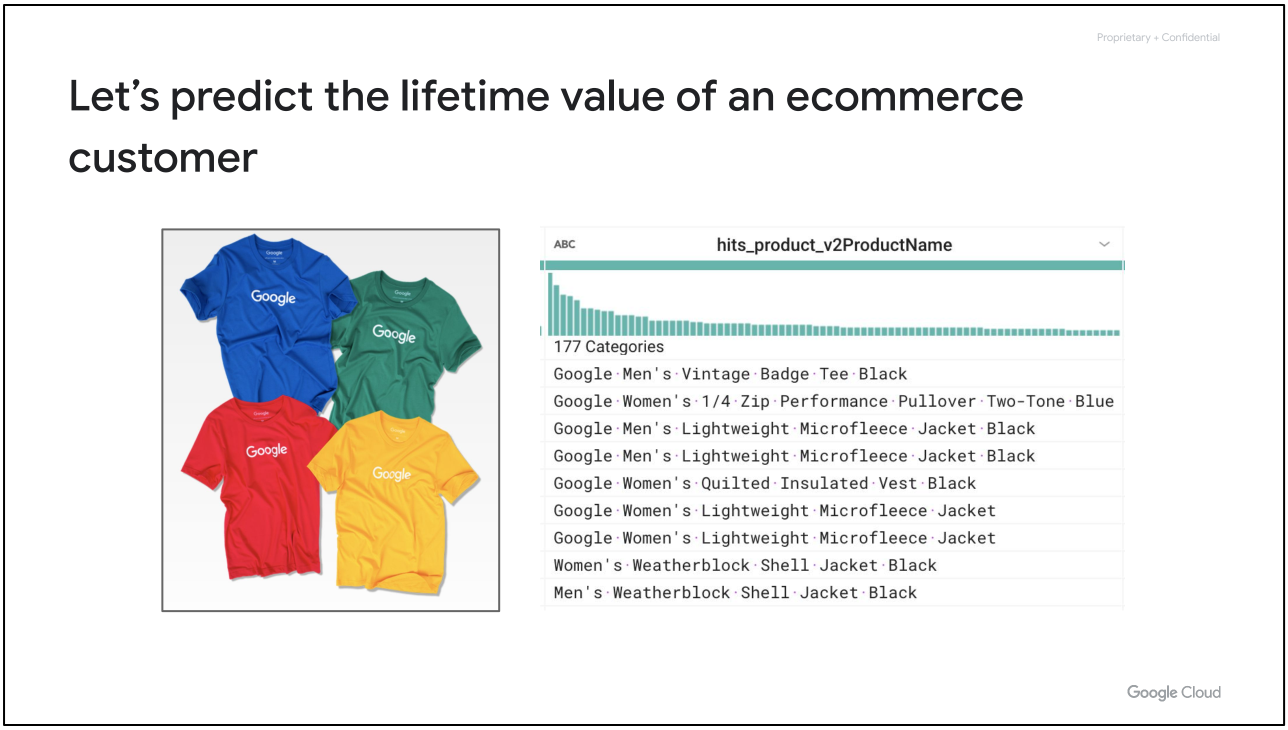


Proprietary + Confidential

A quick example

Predicting customer LTV **with a ML model**

Here our quick scenario will be predicting customer lifetime value with a model.



Let’s predict the lifetime value of an ecommerce customer

ABC hits_product_v2ProductName

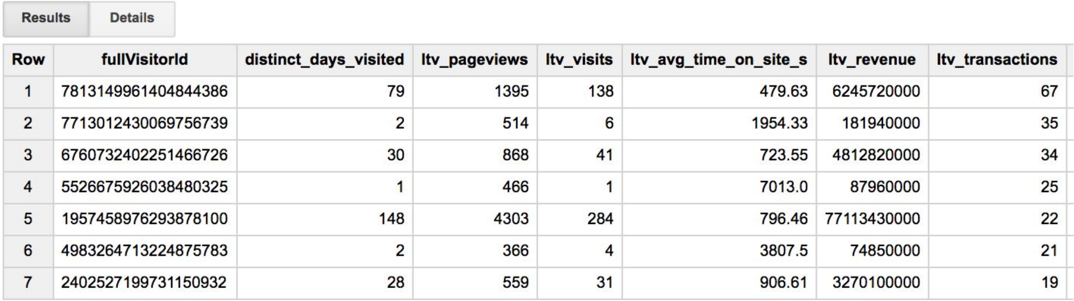
177 Categories

```
Google Men's Vintage Badge Tee Black
Google Women's 1/4 Zip Performance Pullover Two-Tone Blue
Google Men's Lightweight Microfleece Jacket Black
Google Men's Lightweight Microfleece Jacket Black
Google Women's Quilted Insulated Vest Black
Google Women's Lightweight Microfleece Jacket
Google Women's Lightweight Microfleece Jacket
Women's Weatherblock Shell Jacket Black
Men's Weatherblock Shell Jacket Black
```

Our goal is to better target high value customers to our ecommerce throughout their customer lifecycle with special promotions and incentives.

Proprietary + Confidential

Google Analytics provides us with aggregated site visit metrics



Row	fullVisitorId	distinct_days_visited	ltv_pageviews	ltv_visits	ltv_avg_time_on_site_s	ltv_revenue	ltv_transactions
1	7813149961404844386	79	1395	138	479.63	6245720000	67
2	7713012430069756739	2	514	6	1954.33	181940000	35
3	6760732402251466726	30	868	41	723.55	4812820000	34
4	5526675926038480325	1	466	1	7013.0	87960000	25
5	1957458976293878100	148	4303	284	796.46	77113430000	22
6	4983264713224875783	2	366	4	3807.5	74850000	21
7	2402527199731150932	28	559	31	906.61	3270100000	19

Google Cloud

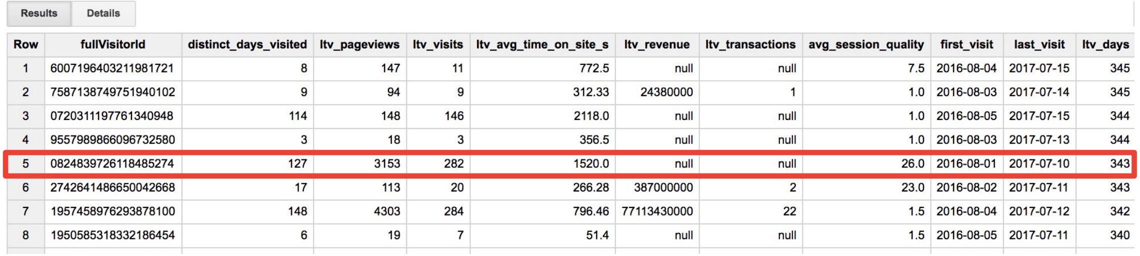
After exploring the data, we can provide a number of useful fields to the model like the number of different days the visitor has been to our website, how many lifetime pageviews, how many total visits, what their average time on site was, the total revenue brought in, and the count of ecommerce transactions on our site.

Now all that is shown here is basic analytics to get a sense of the data but you could feed this historical lifetime-value data and use that to power and predict which customers are high value customers to help you focus and target them for promotions and incentives.

But, before we get too deep building models in BigQuery, we first need to define our data terms in the language that data scientists and other ML professionals use.

Query:
https://console.cloud.google.com/bigquery?project=data-to-insights&ws=!1m3!1m2!2m1!1sdata-to-insights

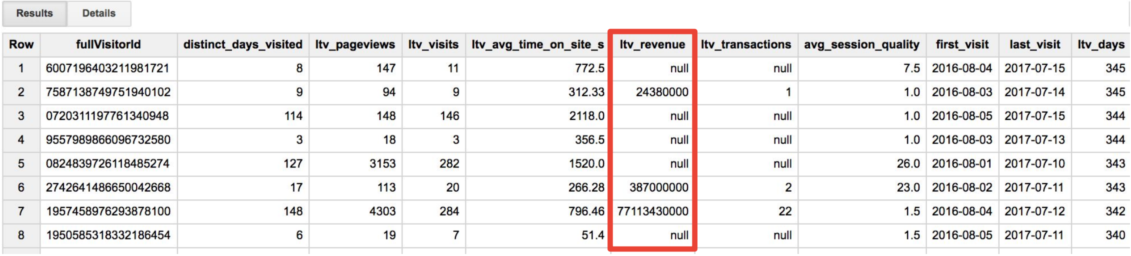
In ML terms, an instance (or observation) is a row of data



Row	fullVisitorId	distinct_days_visited	ltv_pageviews	ltv_visits	ltv_avg_time_on_site_s	ltv_revenue	ltv_transactions	avg_session_quality	first_visit	last_visit	ltv_days
1	6007196403211981721	8	147	11	772.5	null	null	7.5	2016-08-04	2017-07-15	345
2	7587138749751940102	9	94	9	312.33	24380000	1	1.0	2016-08-03	2017-07-14	345
3	0720311197761340948	114	148	146	2118.0	null	null	1.0	2016-08-05	2017-07-15	344
4	9557989866096732580	3	18	3	356.5	null	null	1.0	2016-08-03	2017-07-13	344
5	0824839726118485274	127	3153	282	1520.0	null	null	26.0	2016-08-01	2017-07-10	343
6	2742641486650042668	17	113	20	266.28	387000000	2	23.0	2016-08-02	2017-07-11	343
7	1957458976293878100	148	4303	284	796.46	77113430000	22	1.5	2016-08-04	2017-07-12	342
8	1950585318332186454	6	19	7	51.4	null	null	1.5	2016-08-05	2017-07-11	340

Taking the ecommerce example we had in the previous lesson, a record or row is called an instance or observation. In the screenshot you see here we have 8 instances.

What we are trying to predict for is the label



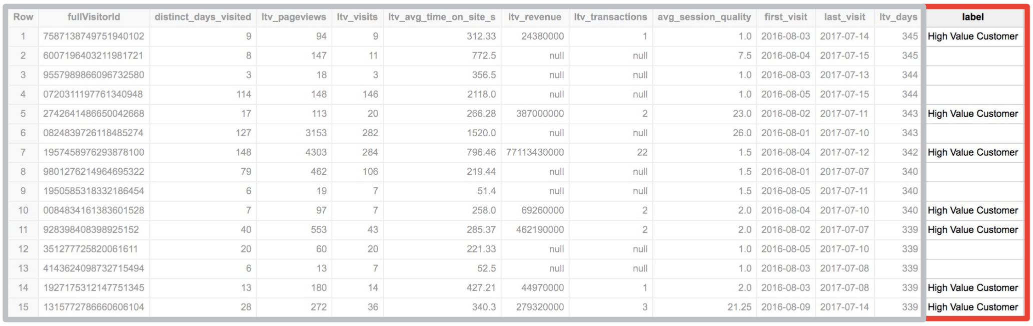
Row	fullVisitorId	distinct_days_visited	ltv_pageviews	ltv_visits	ltv_avg_time_on_site_s	ltv_revenue	ltv_transactions	avg_session_quality	first_visit	last_visit	ltv_days
1	6007196403211981721	8	147	11	772.5	null	null	7.5	2016-08-04	2017-07-15	345
2	7587138749751940102	9	94	9	312.33	24380000	1	1.0	2016-08-03	2017-07-14	345
3	0720311197761340948	114	148	146	2118.0	null	null	1.0	2016-08-05	2017-07-15	344
4	9557989866096732580	3	18	3	356.5	null	null	1.0	2016-08-03	2017-07-13	344
5	0824839726118485274	127	3153	282	1520.0	null	null	26.0	2016-08-01	2017-07-10	343
6	2742641486650042668	17	113	20	266.28	387000000	2	23.0	2016-08-02	2017-07-11	343
7	1957458976293878100	148	4303	284	796.46	77113430000	22	1.5	2016-08-04	2017-07-12	342
8	1950585318332186454	6	19	7	51.4	null	null	1.5	2016-08-05	2017-07-11	340

Here we are predicting the lifetime revenue (number)

A label is the correct answer and will be what you are looking to train the model on with your existing data and predict with your model on future data.

Here the label is lifetime-revenue which is a number we will be trying to predict.

Labels could also be a discrete class of customer like "high value"

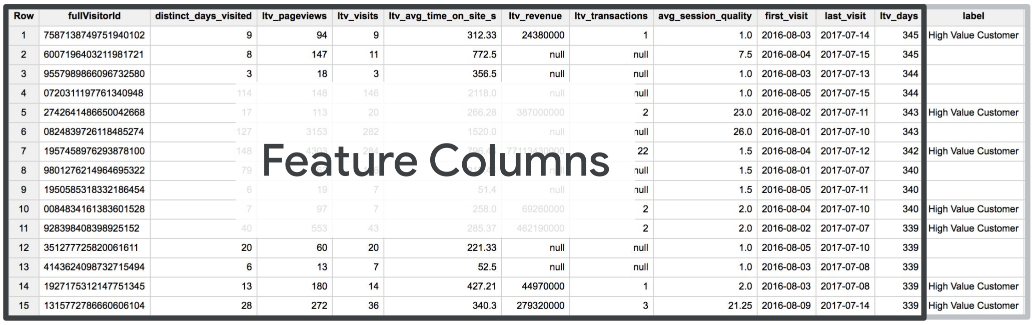


Row	fullVisitorId	distinct_days_visited	ltv_pageviews	ltv_visits	ltv_avg_time_on_site_s	ltv_revenue	ltv_transactions	avg_session_quality	first_visit	last_visit	ltv_days	label
1	7587138749751940102	9	94	9	312.33	24380000	1	1.0	2016-08-03	2017-07-14	345	High Value Customer
2	6007196403211981721	8	147	11	772.5	null	null	7.5	2016-08-04	2017-07-15	345	
3	9557989866096732580	3	18	3	356.5	null	null	1.0	2016-08-03	2017-07-13	344	
4	0720311197761340948	114	148	146	2118.0	null	null	1.0	2016-08-05	2017-07-15	344	
5	2742641486650042668	17	113	20	266.28	387000000	2	23.0	2016-08-02	2017-07-11	343	High Value Customer
6	0824839726118485274	127	3153	282	1520.0	null	null	26.0	2016-08-01	2017-07-10	343	
7	1957458976293878100	148	4303	284	796.46	77113430000	22	1.5	2016-08-04	2017-07-12	342	High Value Customer
8	9801276214964695322	79	462	106	219.44	null	null	1.5	2016-08-01	2017-07-07	340	
9	1950585318332186454	6	19	7	51.4	null	null	1.5	2016-08-05	2017-07-11	340	
10	0084834161383601528	7	97	7	258.0	69260000	2	2.0	2016-08-04	2017-07-10	340	High Value Customer
11	928398408398925152	40	553	43	285.37	462190000	2	2.0	2016-08-02	2017-07-07	339	High Value Customer
12	351277725820061611	20	60	20	221.33	null	null	1.0	2016-08-05	2017-07-10	339	
13	4143624098732715494	6	13	7	52.5	null	null	1.0	2016-08-03	2017-07-08	339	
14	1927175312147751345	13	180	14	427.21	44970000	1	2.0	2016-08-03	2017-07-08	339	High Value Customer
15	1315772786660606104	28	272	36	340.3	279320000	3	21.25	2016-08-09	2017-07-14	339	High Value Customer

What you are trying to predict for (number or discrete class) influences the model you will choose

Labels could also be things like binary values like "High Value Customer" or not as shown here. Knowing what you are trying to predict (a class, a number etc) will greatly influence the type of model you will use later.

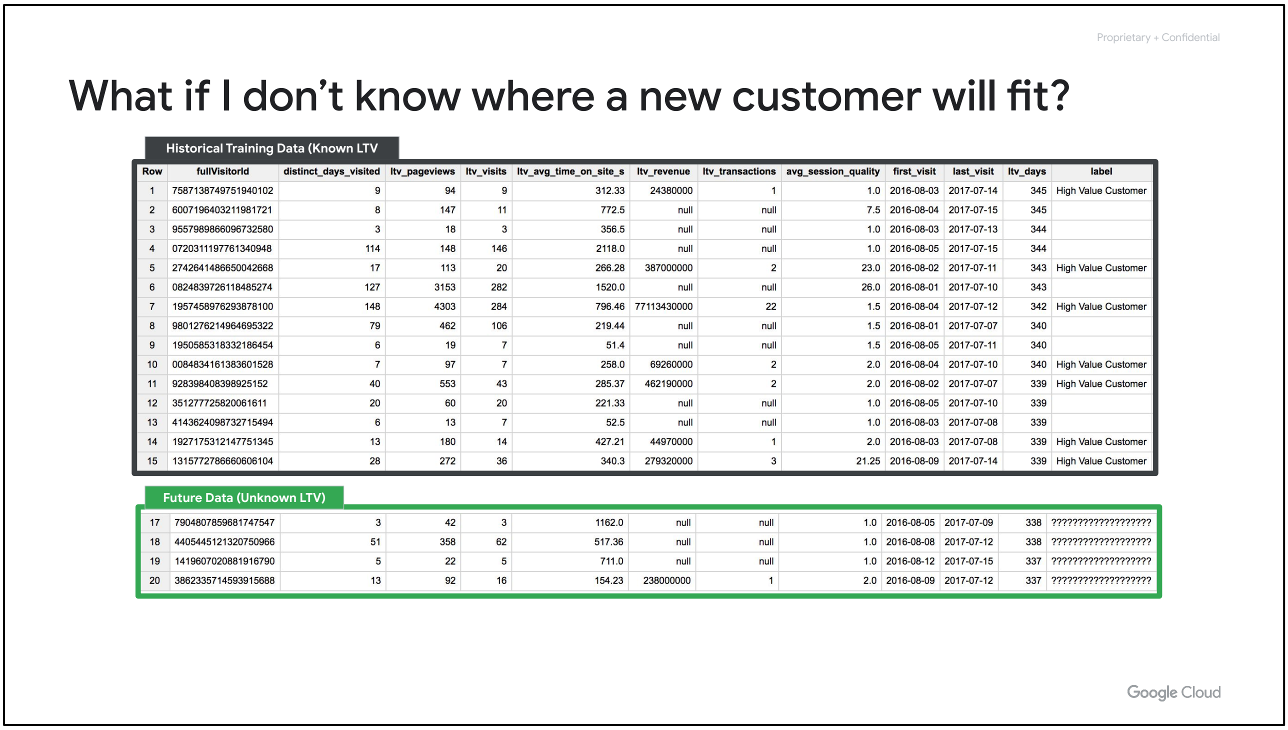
The other columns of data are your potential feature columns


Feature Columns

Row	fullVisitorId	distinct_days_visited	ltv_pageviews	ltv_visits	ltv_avg_time_on_site_s	ltv_revenue	ltv_transactions	avg_session_quality	first_visit	last_visit	ltv_days	label
1	7587138749751940102	9	94	9	312.33	24380000	1	1.0	2016-08-03	2017-07-14	345	High Value Customer
2	6007196403211981721	8	147	11	772.5	null	null	7.5	2016-08-04	2017-07-15	345	
3	9557989866096732580	3	18	3	356.5	null	null	1.0	2016-08-03	2017-07-13	344	
4	0720311197761340948	114	148	146	2118.0	null	null	1.0	2016-08-05	2017-07-15	344	
5	2742641486650042668	17	113	20	266.28	387000000	2	23.0	2016-08-02	2017-07-11	343	High Value Customer
6	0824839726118485274	127	3153	282	1520.0	null	null	26.0	2016-08-01	2017-07-10	343	
7	1957458976293878100	148	4303	284	796.45	77113430000	22	1.5	2016-08-04	2017-07-12	342	High Value Customer
8	9801276214964695322	79	[illegible]	[illegible]	[illegible]	null	null	1.5	2016-08-01	2017-07-07	340	
9	1950585318332186454	6	19	7	51.4	null	null	1.5	2016-08-05	2017-07-11	340	
10	0084834161383601528	7	97	7	258.0	69260000	2	2.0	2016-08-04	2017-07-10	340	High Value Customer
11	928398408398925152	40	553	43	285.37	462190000	2	2.0	2016-08-02	2017-07-07	339	High Value Customer
12	351277725820061611	20	60	20	221.33	null	null	1.0	2016-08-05	2017-07-10	339	
13	4143624098732715494	6	13	7	52.5	null	null	1.0	2016-08-03	2017-07-08	339	
14	1927175312147751345	13	180	14	427.21	44970000	1	2.0	2016-08-03	2017-07-08	339	High Value Customer
15	1315772786660606104	28	272	36	340.3	279320000	3	21.25	2016-08-09	2017-07-14	339	High Value Customer

You will try to *model* the relationship between the features and your label

Those columns are called features. We have a whole module dedicated to creating ML datasets in BigQuery which touches on the critical topic of “feature engineering” which is exploring, cleaning, and preprocessing your data before you input it into your ML model. This is often the hardest part of any ML project (and why it’s great you already enjoy working with data as an analyst!).



What if I don’t know where a new customer will fit?

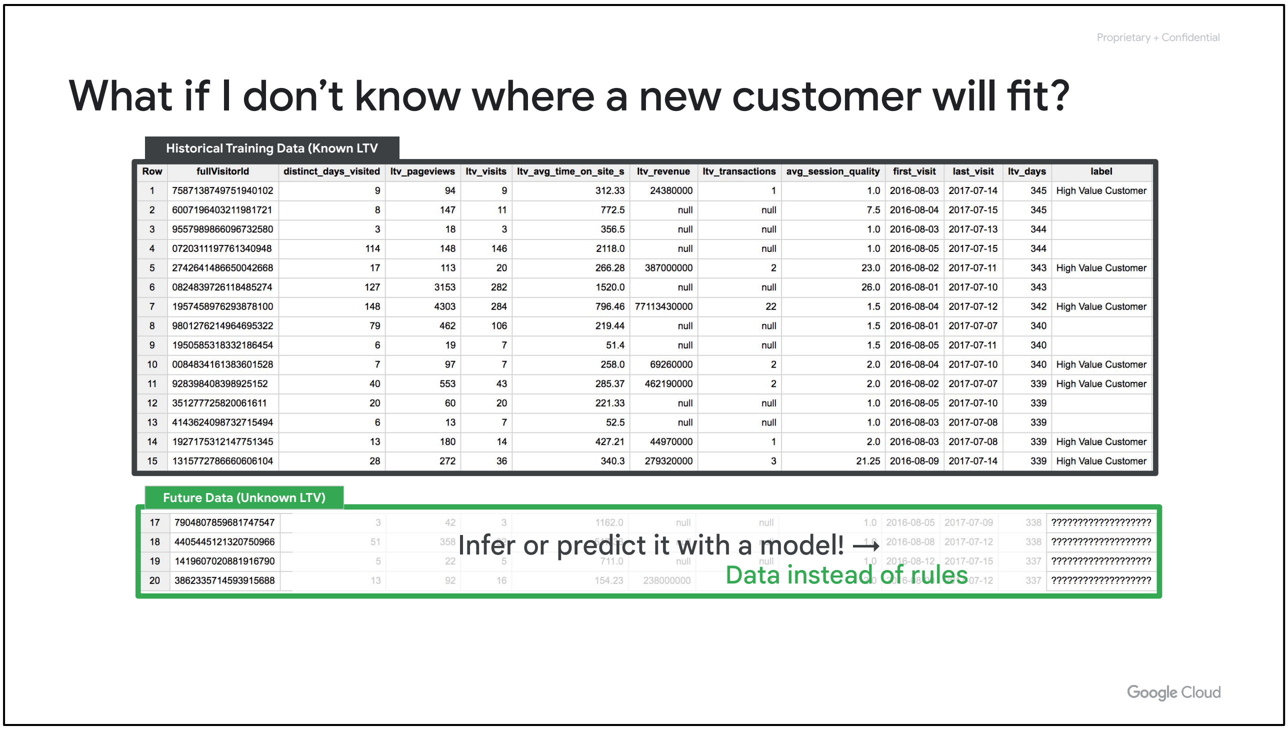
Historical Training Data (Known LTV

Row	fullVisitorId	distinct_days_visited	ltv_pageviews	ltv_visits	ltv_avg_time_on_site_s	ltv_revenue	ltv_transactions	avg_session_quality	first_visit	last_visit	ltv_days	label
1	7587138749751940102	9	94	9	312.33	24380000	1	1.0	2016-08-03	2017-07-14	345	High Value Customer
2	6007196403211981721	8	147	11	772.5	null	null	7.5	2016-08-04	2017-07-15	345	
3	9557989866096732580	3	18	3	356.5	null	null	1.0	2016-08-03	2017-07-13	344	
4	0720311197761340948	114	148	146	2118.0	null	null	1.0	2016-08-05	2017-07-15	344	
5	2742641486650042668	17	113	20	266.28	387000000	2	23.0	2016-08-02	2017-07-11	343	High Value Customer
6	0824839726118485274	127	3153	282	1520.0	null	null	26.0	2016-08-01	2017-07-10	343	
7	1957458976293878100	148	4303	284	796.46	77113430000	22	1.5	2016-08-04	2017-07-12	342	High Value Customer
8	9801276214964695322	79	462	106	219.44	null	null	1.5	2016-08-01	2017-07-07	340	
9	1950585318332186454	6	19	7	51.4	null	null	1.5	2016-08-05	2017-07-11	340	
10	0084834161383601528	7	97	7	258.0	69260000	2	2.0	2016-08-04	2017-07-10	340	High Value Customer
11	928398408398925152	40	553	43	285.37	462190000	2	2.0	2016-08-02	2017-07-07	339	High Value Customer
12	351277725820061611	20	60	20	221.33	null	null	1.0	2016-08-05	2017-07-10	339	
13	4143624098732715494	6	13	7	52.5	null	null	1.0	2016-08-03	2017-07-08	339	
14	1927175312147751345	13	180	14	427.21	44970000	1	2.0	2016-08-03	2017-07-08	339	High Value Customer
15	1315772786660606104	28	272	36	340.3	279320000	3	21.25	2016-08-09	2017-07-14	339	High Value Customer

Future Data (Unknown LTV)

17	7904807859681747547	3	42	3	1162.0	null	null	1.0	2016-08-05	2017-07-09	338	??????????????????
18	4405445121320750966	51	358	62	517.36	null	null	1.0	2016-08-08	2017-07-12	338	??????????????????
19	1419607020881916790	5	22	5	711.0	null	null	1.0	2016-08-12	2017-07-15	337	??????????????????
20	3862335714593915688	13	92	16	154.23	238000000	1	2.0	2016-08-09	2017-07-12	337	??????????????????

Now, say some new data comes in that you don’t have a label for. We now have a dataset of labeled examples and a dataset with unknowns.



Proprietary + Confidential

What if I don't know where a new customer will fit?

Historical Training Data (Known LTV

Row	fullVisitorId	distinct_days_visited	ltv_pageviews	ltv_visits	ltv_avg_time_on_site_s	ltv_revenue	ltv_transactions	avg_session_quality	first_visit	last_visit	ltv_days	label
1	7587138749751940102	9	94	9	312.33	24380000	1	1.0	2016-08-03	2017-07-14	345	High Value Customer
2	6007196403211981721	8	147	11	772.5	null	null	7.5	2016-08-04	2017-07-15	345	
3	9557989866096732580	3	18	3	356.5	null	null	1.0	2016-08-03	2017-07-13	344	
4	0720311197761340948	114	148	146	2118.0	null	null	1.0	2016-08-05	2017-07-15	344	
5	2742641486650042668	17	113	20	266.28	387000000	2	23.0	2016-08-02	2017-07-11	343	High Value Customer
6	0824839726118485274	127	3153	282	1520.0	null	null	26.0	2016-08-01	2017-07-10	343	
7	1957458976293878100	148	4303	284	796.46	77113430000	22	1.5	2016-08-04	2017-07-12	342	High Value Customer
8	9801276214964695322	79	462	106	219.44	null	null	1.5	2016-08-01	2017-07-07	340	
9	1950585318332186454	6	19	7	51.4	null	null	1.5	2016-08-05	2017-07-11	340	
10	0084834161383601528	7	97	7	258.0	69260000	2	2.0	2016-08-04	2017-07-10	340	High Value Customer
11	928398408398925152	40	553	43	285.37	462190000	2	2.0	2016-08-02	2017-07-07	339	High Value Customer
12	351277725820061611	20	60	20	221.33	null	null	1.0	2016-08-05	2017-07-10	339	
13	4143624098732715494	6	13	7	52.5	null	null	1.0	2016-08-03	2017-07-08	339	
14	1927175312147751345	13	180	14	427.21	44970000	1	2.0	2016-08-03	2017-07-08	339	High Value Customer
15	1315772786660606104	28	272	36	340.3	279320000	3	21.25	2016-08-09	2017-07-14	339	High Value Customer

Future Data (Unknown LTV)

17	7904807859681747547	3	42	3	1162.0	null	null	1.0	2016-08-05	2017-07-09	338	??????????????????
18	4405445121320750966	51	358	62	517.36	null	null	1.0	2016-08-08	2017-07-12	338	??????????????????
19	1419607020881916790	5	22	5	711.0	null	null	1.0	2016-08-12	2017-07-15	337	??????????????????
20	3862335714593915688	13	92	16	154.23	238000000	[illegible]	[illegible]	[illegible]	2017-07-12	337	??????????????????

Infer or predict it with a model! ⟶

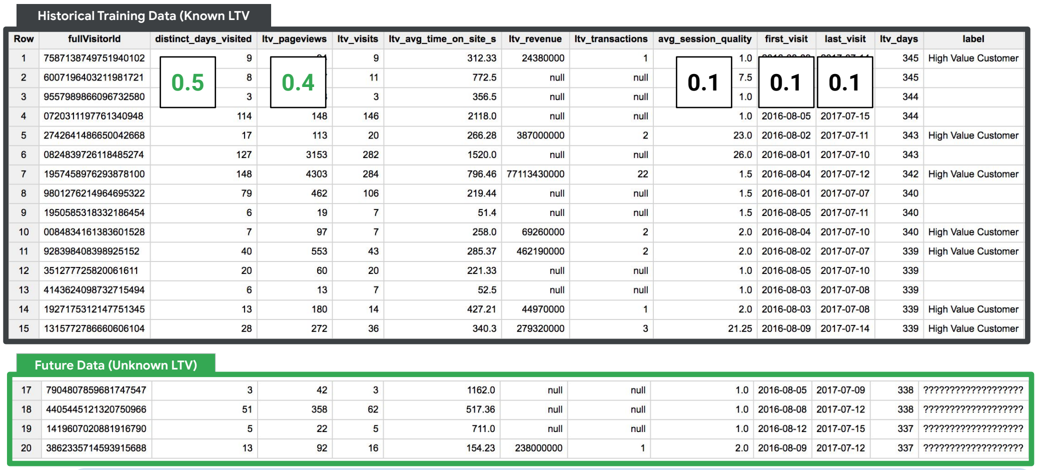
Data instead of rules

Google Cloud

Well this is the fun part! We can draw inference or predict those values with a model! Again an ML model will build a recipe for determining those output values (in this case classifying whether that customer is High Value or not) based on your labeled training data (which is the blue box shown here).

Next up, you'll learn how to code these models yourself with just SQL.

Your model will learn the weight to give each feature (-1, 1)



0.5
0.4
0.1
0.1
0.1

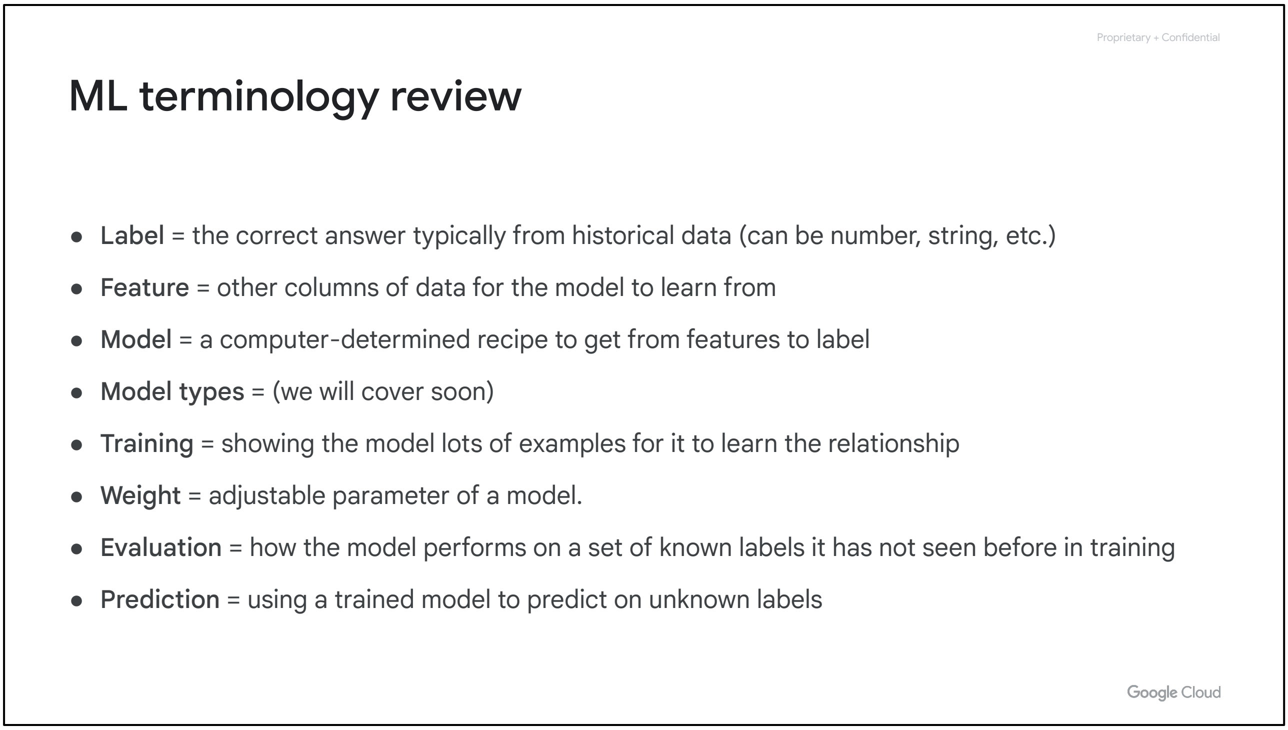
Historical Training Data (Known LTV

Row	fullVisitorId	distinct_days_visited	ltv_pageviews	ltv_visits	ltv_avg_time_on_site_s	ltv_revenue	ltv_transactions	avg_session_quality	first_visit	last_visit	ltv_days	label
1	7587138749751940102	9	[illegible]	9	312.33	24380000	1	1.0	[illegible]	[illegible]	345	High Value Customer
2	6007196403211981721	8	[illegible]	11	772.5	null	null	7.5	[illegible]	[illegible]	345	
3	9557989866096732580	3	[illegible]	3	356.5	null	null	1.0	[illegible]	[illegible]	344	
4	0720311197761340948	114	148	146	2118.0	null	null	1.0	2016-08-05	2017-07-15	344	
5	2742641486650042668	17	113	20	266.28	387000000	2	23.0	2016-08-02	2017-07-11	343	High Value Customer
6	0824839726118485274	127	3153	282	1520.0	null	null	26.0	2016-08-01	2017-07-10	343	
7	1957458976293878100	148	4303	284	796.46	77113430000	22	1.5	2016-08-04	2017-07-12	342	High Value Customer
8	9801276214964695322	79	462	106	219.44	null	null	1.5	2016-08-01	2017-07-07	340	
9	1950585318332186454	6	19	7	51.4	null	null	1.5	2016-08-05	2017-07-11	340	
10	0084834161383601528	7	97	7	258.0	69260000	2	2.0	2016-08-04	2017-07-10	340	High Value Customer
11	928398408398925152	40	553	43	285.37	462190000	2	2.0	2016-08-02	2017-07-07	339	High Value Customer
12	351277725820061611	20	60	20	221.33	null	null	1.0	2016-08-05	2017-07-10	339	
13	4143624098732715494	6	13	7	52.5	null	null	1.0	2016-08-03	2017-07-08	339	
14	1927175312147751345	13	180	14	427.21	44970000	1	2.0	2016-08-03	2017-07-08	339	High Value Customer
15	1315772786660606104	28	272	36	340.3	279320000	3	21.25	2016-08-09	2017-07-14	339	High Value Customer

Future Data (Unknown LTV)

17	7904807859681747547	3	42	3	1162.0	null	null	1.0	2016-08-05	2017-07-09	338	??????????????????
18	4405445121320750966	51	358	62	517.36	null	null	1.0	2016-08-08	2017-07-12	338	??????????????????
19	1419607020881916790	5	22	5	711.0	null	null	1.0	2016-08-12	2017-07-15	337	??????????????????
20	3862335714593915688	13	92	16	154.23	238000000	1	2.0	2016-08-09	2017-07-12	337	??????????????????

After the model is trained, you can see the relative importance of each field

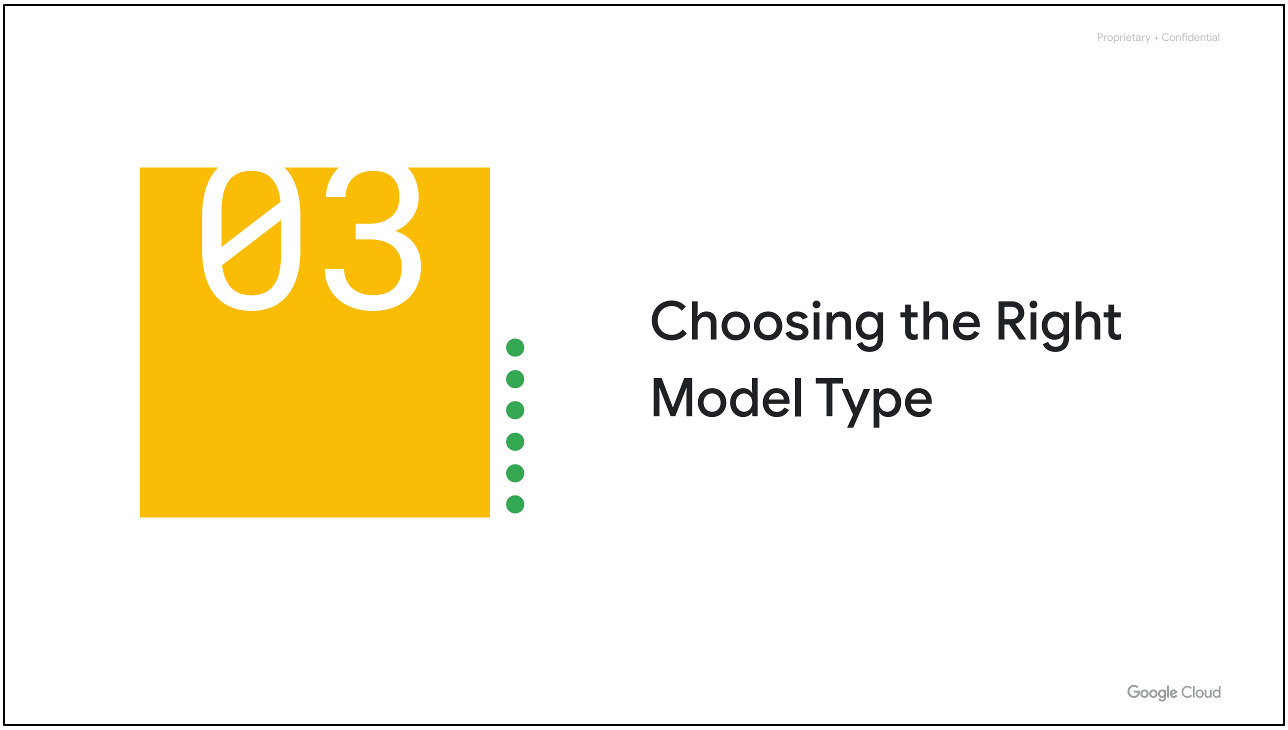


ML terminology review

- **Label** = the correct answer typically from historical data (can be number, string, etc.)
- **Feature** = other columns of data for the model to learn from
- **Model** = a computer-determined recipe to get from features to label
- **Model types** = (we will cover soon)
- **Training** = showing the model lots of examples for it to learn the relationship
- **Weight** = adjustable parameter of a model.
- **Evaluation** = how the model performs on a set of known labels it has not seen before in training
- **Prediction** = using a trained model to predict on unknown labels

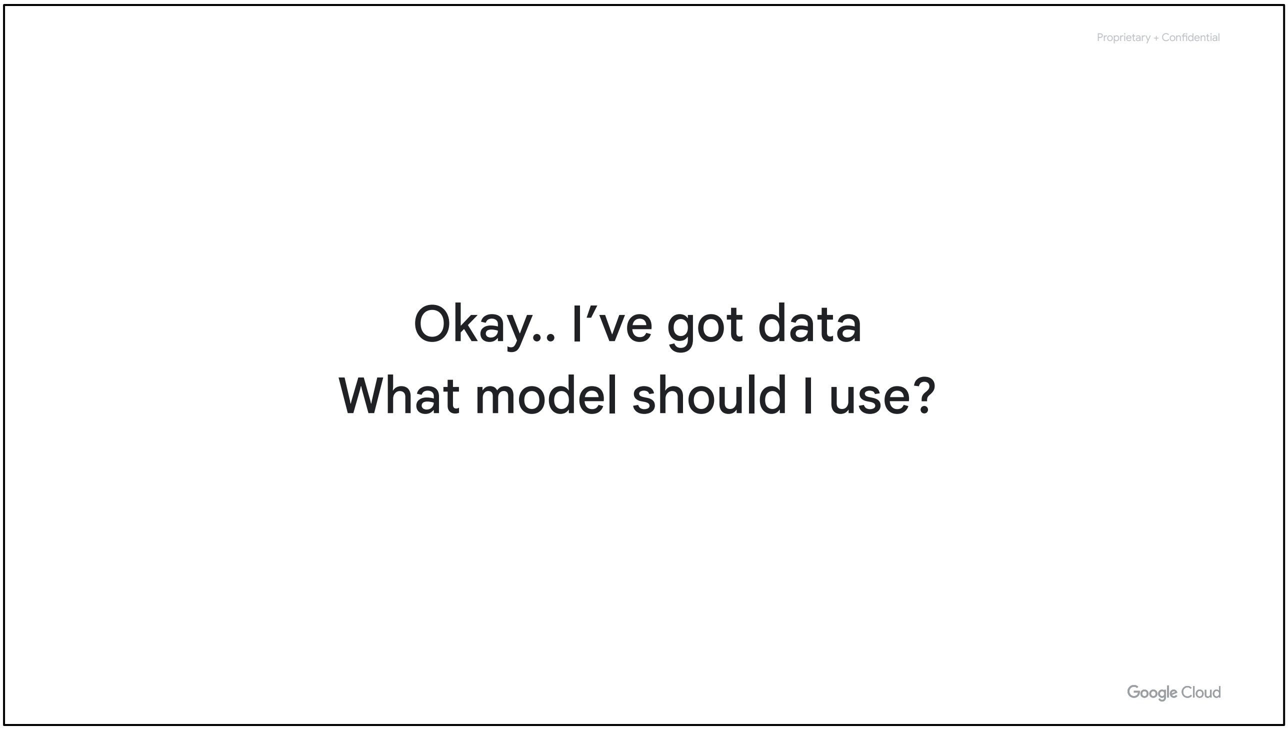
Google Cloud

Now, say some new data comes in that you don’t have a label for. We now have a dataset of labeled examples and a dataset with unknowns.



03

Choosing the Right Model Type

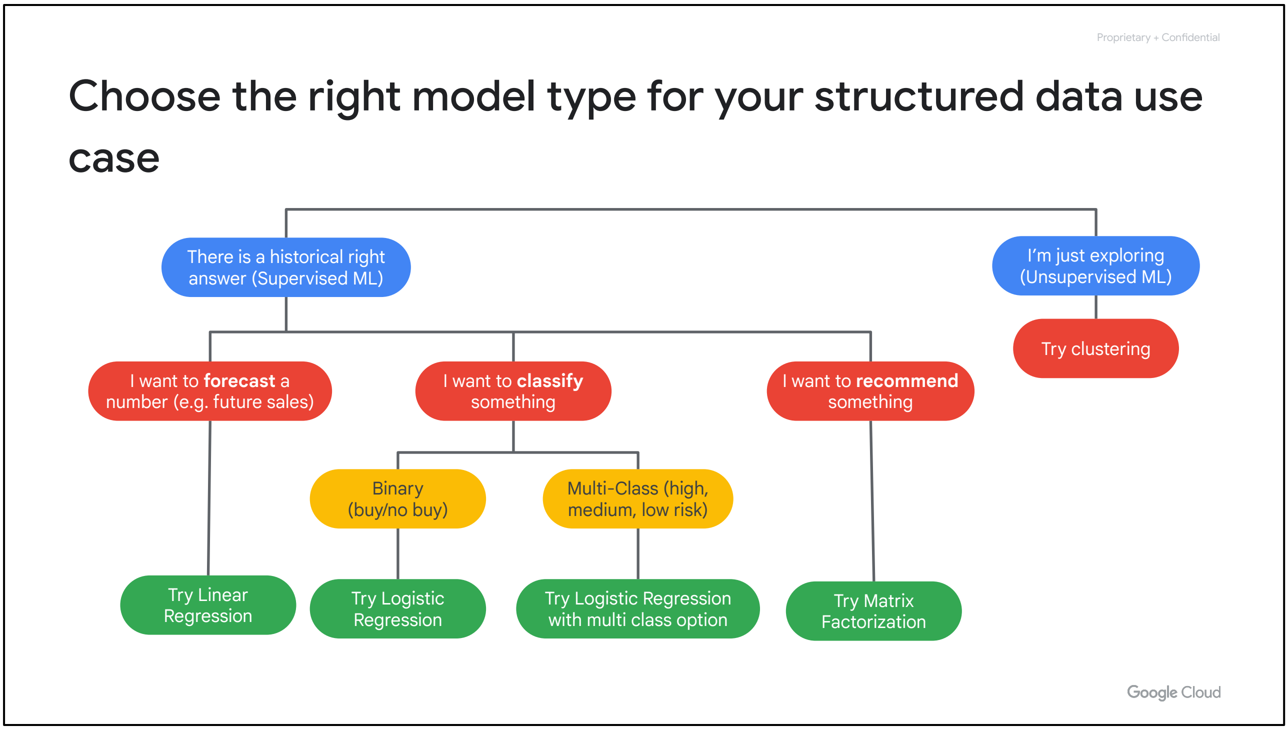


Okay.. I’ve got data
What model should I use?

Google Cloud

We will revisit machine learning in greater depth in each of the modules in this course. For now just remember that we want to teach the computer using examples, not with rules.

Any business application where you have those long SWITCH or CASE statements or IF THEN logic manually coded AND you have a history of good labeled data is a possible application for Machine Learning.



Proprietary + Confidential

Choose the right model type for your structured data use case

There is a historical right answer (Supervised ML)

I'm just exploring (Unsupervised ML)

Try clustering

I want to **forecast** a number (e.g. future sales)

I want to **classify** something

I want to **recommend** something

Binary (buy/no buy)

Multi-Class (high, medium, low risk)

Try Linear Regression

Try Logistic Regression

Try Logistic Regression with multi class option

Try Matrix Factorization

Google Cloud

So, if you have a structured dataset that you think is a good use case for machine learning the next step is to find a model type that is appropriate for your use case.

Out of all the models out there, what's a good place for you to start for simple prototyping?

Here's a decision tree to help guide us. We'll walk through each of the branches.

The first question is what kind of activity you're engaging in. Is there a right answer or ground truth that exists in your historical data that you want to model?

You'll see later in BigQuery ML that you can just specify model type equal 'linear regression' and BigQuery handles the rest for you.

What didn't you see here that you may have heard of?

There are many different types of models out there that you may not see on this chart. More complex models like deep neural networks, decision trees, random forests are also available for modelling. You'll even build a custom model using Neural Architecture Search to build a Deep Neural Network later in this course without using any code with AutoML. It's my recommendation that even if you know how to build advanced models that you start with simpler ones first because they often train faster and give you an indication of whether or not ML is a viable solution for your problem.

Quiz: What model should you use if...

Question

I want to predict ecommerce sales figures for the next quarter

A. Forecasting (linear regression, etc.)

B. Classification (logistic regression, etc.)

C. Recommendation (matrix factorialization, etc.)

D. Unsupervised Learning (clustering, etc.)

E. All of the above

Quiz: What model should you use if...

Answer

I want to predict ecommerce sales figures for the next quarter

A. **Forecasting (linear regression, etc.)** ✓

B. Classification (logistic regression, etc.)

C. Recommendation (matrix factorialization, etc.)

D. Unsupervised Learning (clustering, etc.)

E. All of the above

Quiz: What model should you use if...

Question

I want to predict whether a user will buy or not buy on a visit

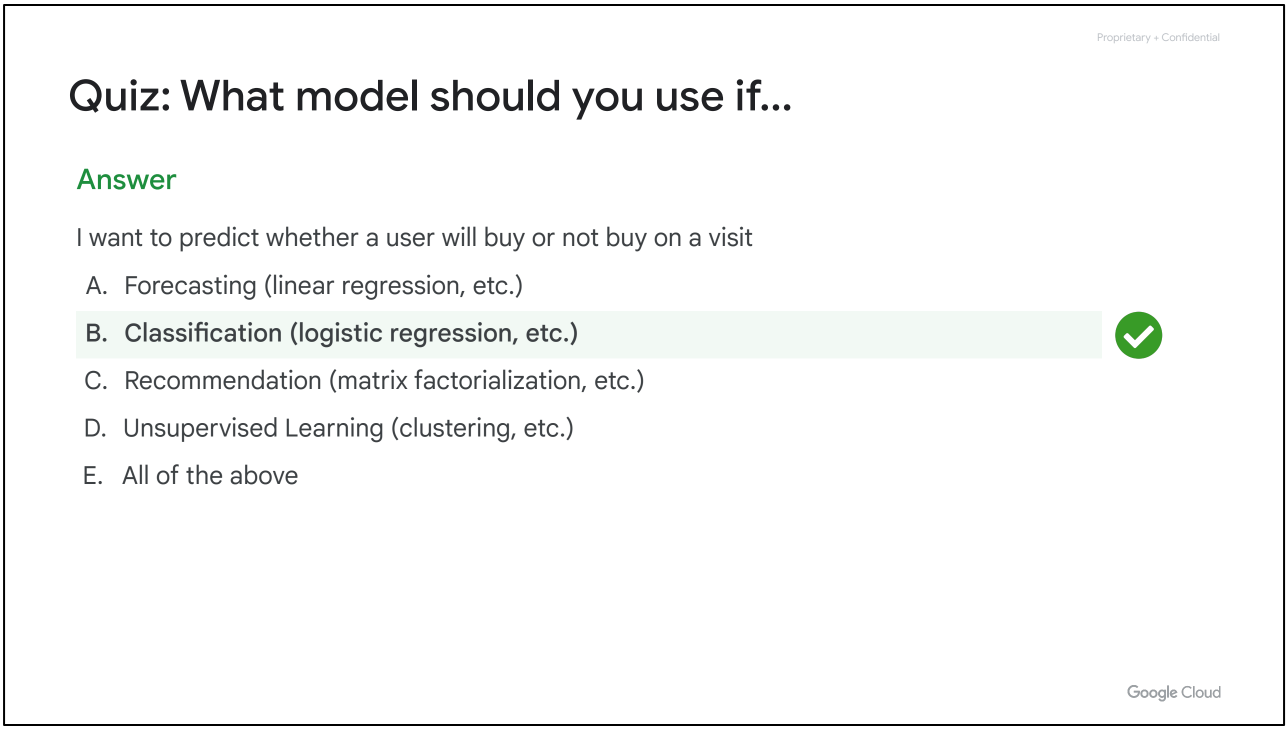
A. Forecasting (linear regression, etc.)

B. Classification (logistic regression, etc.)

C. Recommendation (matrix factorialization, etc.)

D. Unsupervised Learning (clustering, etc.)

E. All of the above



Quiz: What model should you use if...

Answer

I want to predict whether a user will buy or not buy on a visit

A. Forecasting (linear regression, etc.)

B. Classification (logistic regression, etc.) ✓

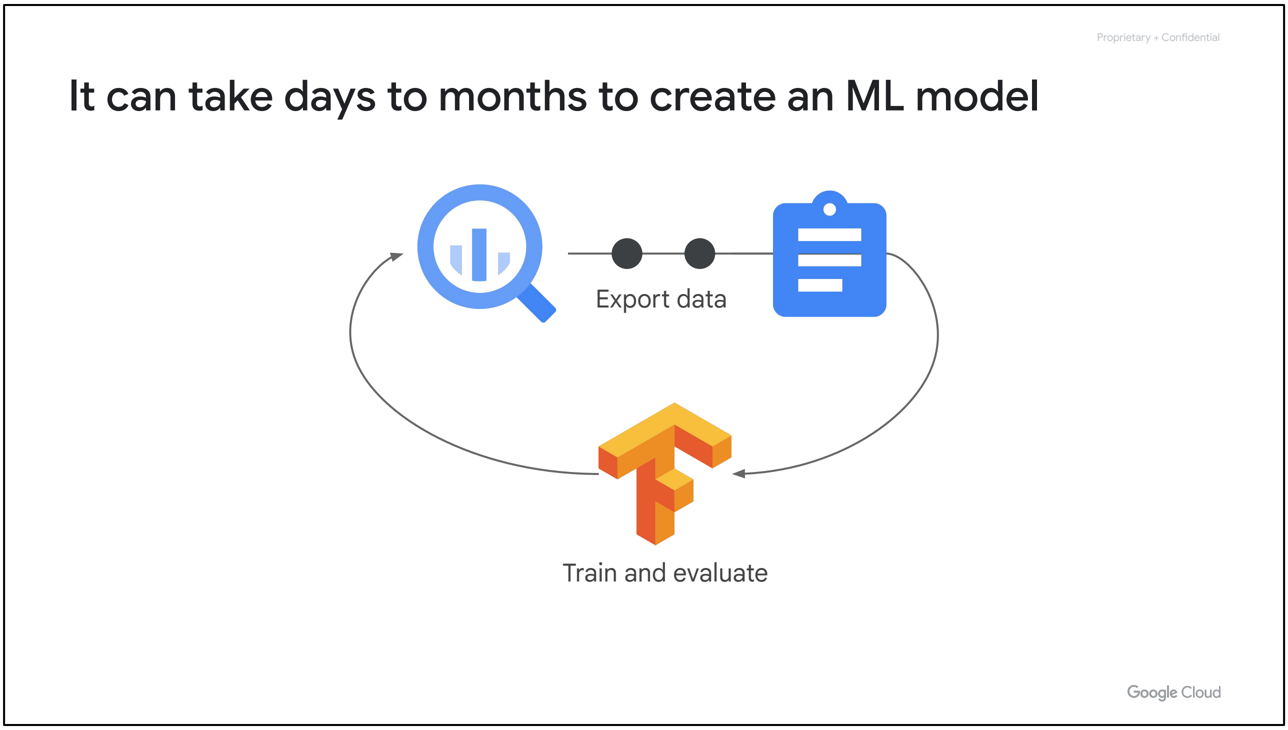
C. Recommendation (matrix factorialization, etc.)

D. Unsupervised Learning (clustering, etc.)

E. All of the above

04

Creating ML Models with SQL



Proprietary + Confidential

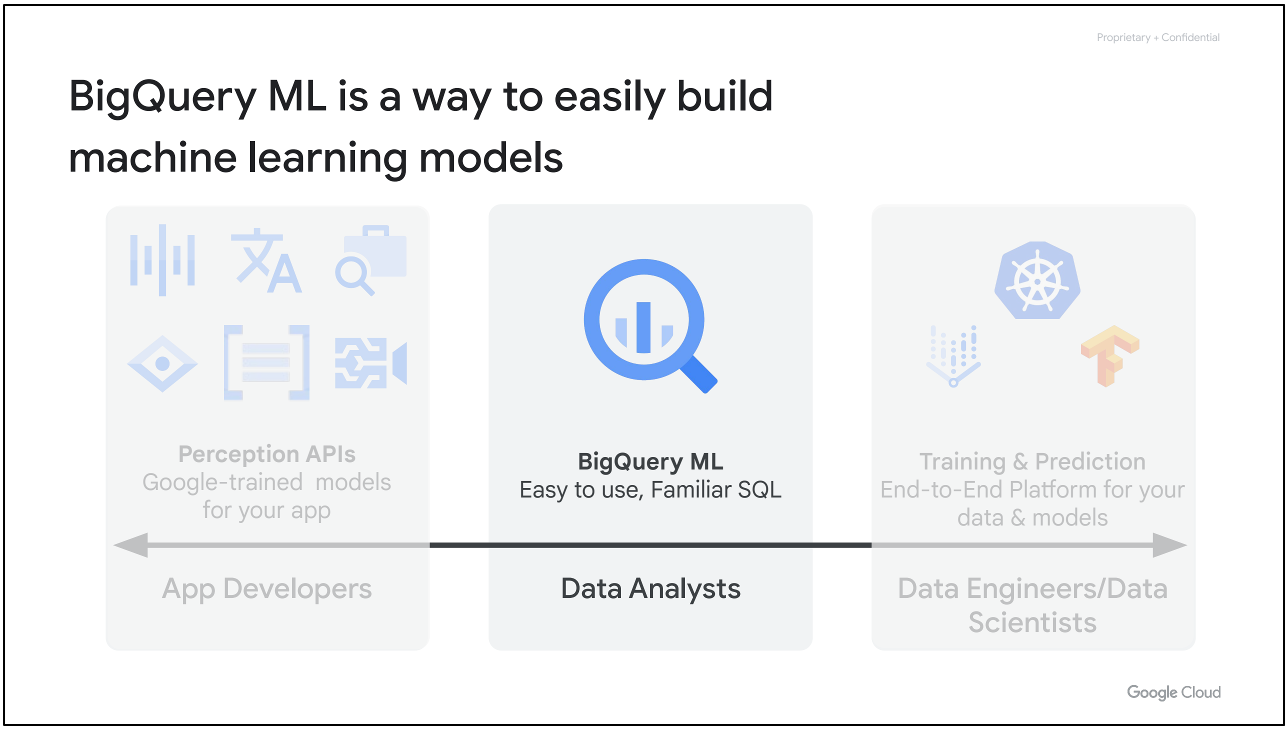
It can take days to months to create an ML model

Export data

Train and evaluate

Google Cloud

You know that building ML models can be very time intensive. Your first must export small amounts of data from BQ in pandas and DataLab. You then transform the data to be used in Tensorflow. You build the model in Tensorflow and train it locally or on a VM. Doing that with a small model then requires that you go back and get more data to create new features, and improve performance. Repeat. It’s hard, so you stop after a few iterations.



BigQuery ML is a way to easily build machine learning models

Perception APIs
Google-trained models
for your app

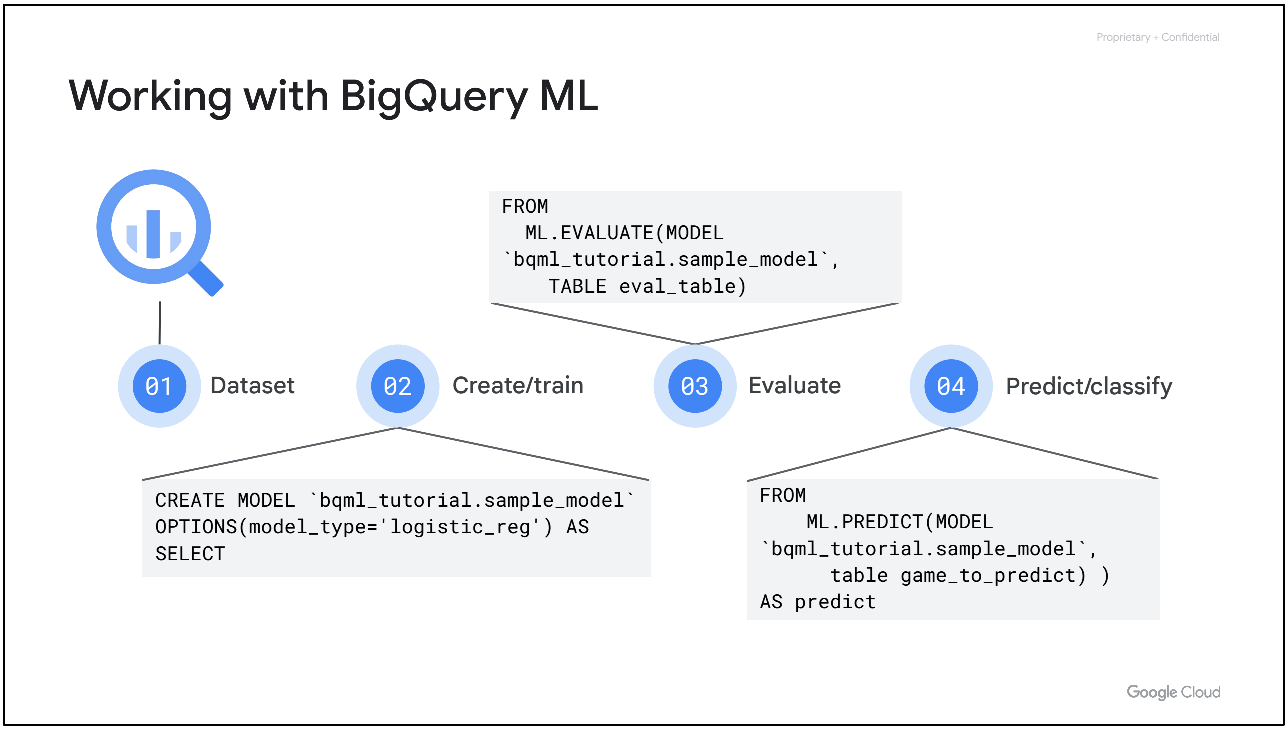
App Developers

BigQuery ML
Easy to use, Familiar SQL

Data Analysts

Training & Prediction
End-to-End Platform for your
data & models

Data Engineers/Data
Scientists



Working with BigQuery ML

```
FROM
  ML.EVALUATE(MODEL
`bqml_tutorial.sample_model`,
    TABLE eval_table)
```

01 Dataset

02 Create/train

03 Evaluate

04 Predict/classify

```
CREATE MODEL `bqml_tutorial.sample_model`
OPTIONS(model_type='logistic_reg') AS
SELECT
```

```
FROM
    ML.PREDICT(MODEL
`bqml_tutorial.sample_model`,
      table game_to_predict) )
AS predict
```

Google Cloud

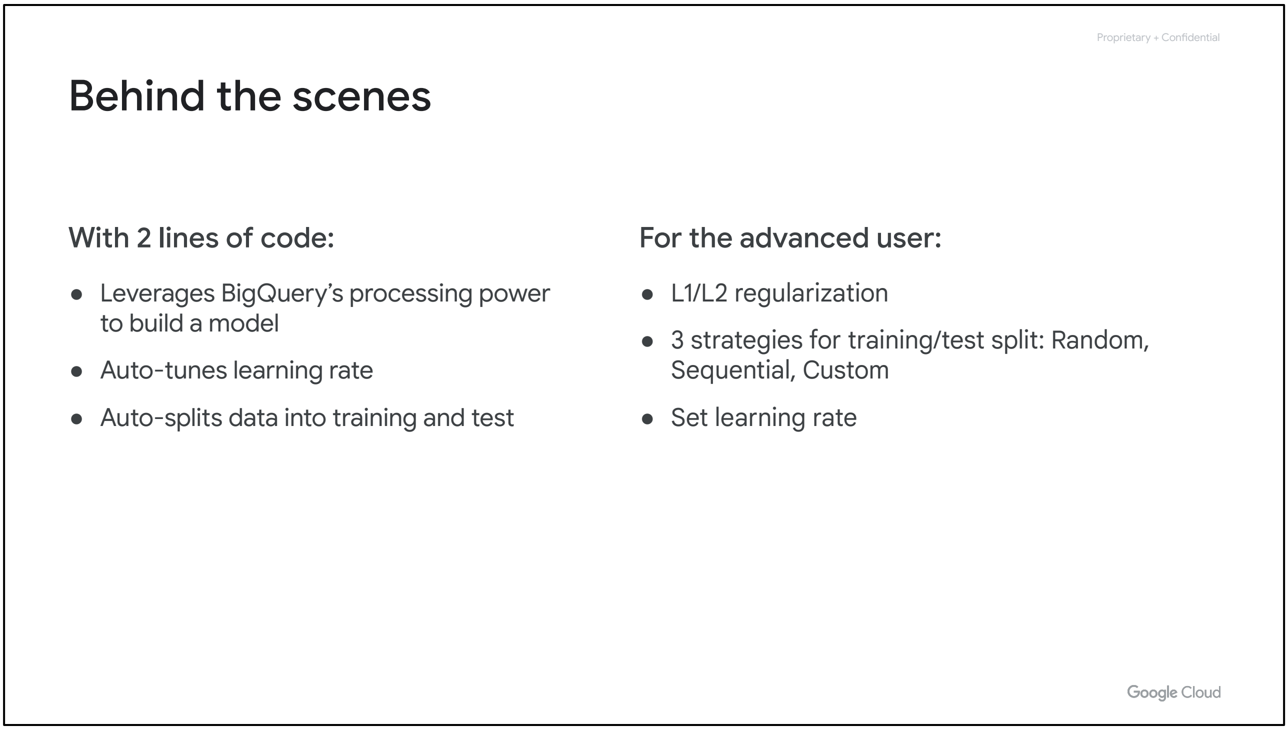
In four major steps it looks like this:

Write SQL query to extract training data from BigQuery

Create a model, specifying model type

Evaluate model and verify that it meets requirements

Predict using model on data extracted from BigQuery



Behind the scenes

With 2 lines of code:

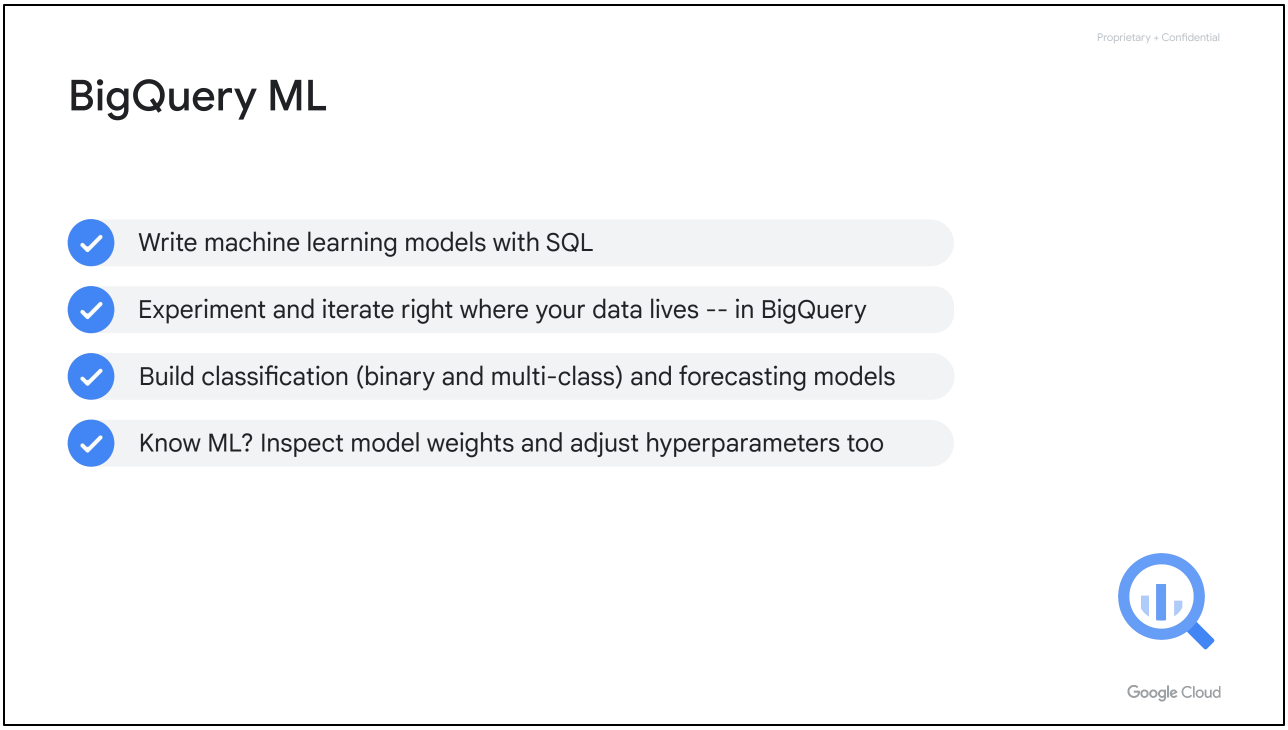
- Leverages BigQuery's processing power to build a model
- Auto-tunes learning rate
- Auto-splits data into training and test

For the advanced user:

- L1/L2 regularization
- 3 strategies for training/test split: Random, Sequential, Custom
- Set learning rate

BigQuery ML was designed with simplicity in mind. To that end, you don't have to define the ML hyperparameters (a fancy way of saying knobs that are set on the model before the training starts) like the learning rate or even the training and test set split. BigQuery ML does that for you.

In addition, with OPTIONS if you wanted to you can also set regularization or different strategies for creating your training and test sets, and manually setting the learning rate.



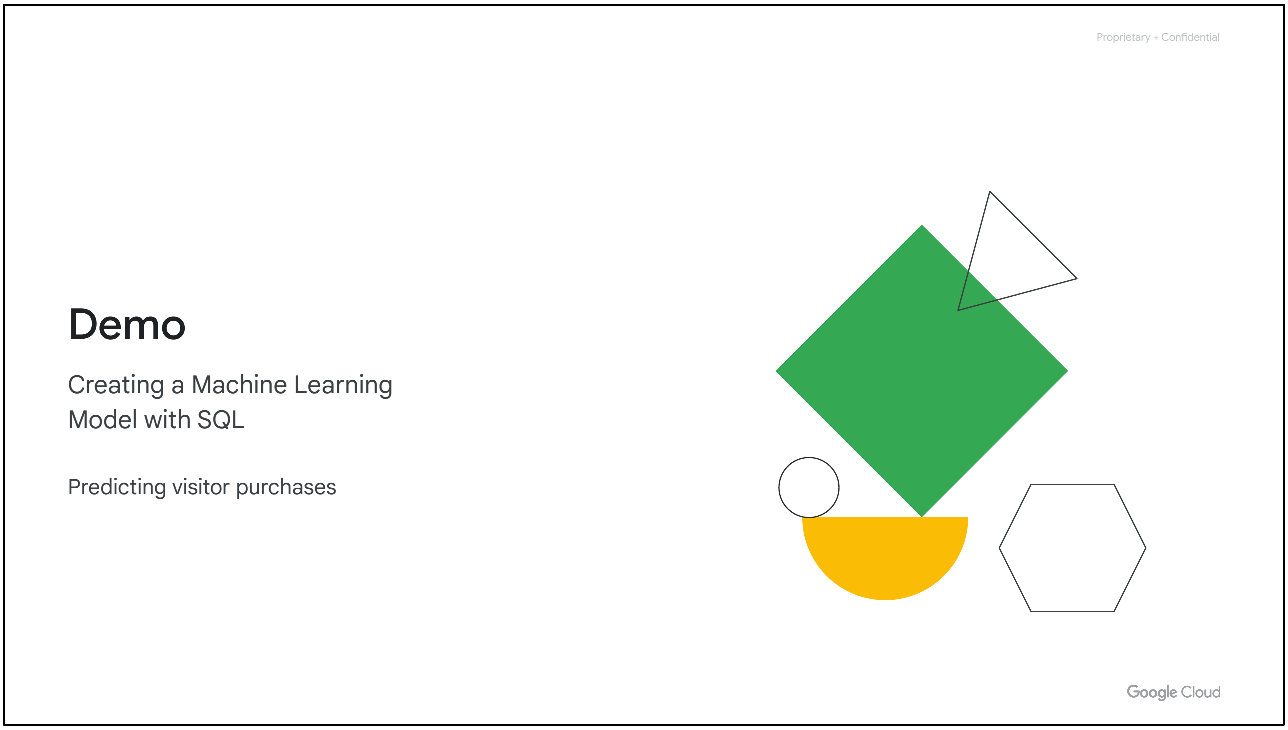
BigQuery ML

- ✓ Write machine learning models with SQL
- ✓ Experiment and iterate right where your data lives -- in BigQuery
- ✓ Build classification (binary and multi-class) and forecasting models
- ✓ Know ML? Inspect model weights and adjust hyperparameters too

Google Cloud

And if you're explaining BigQuery ML to others, I often just list these main points. To recap

BigQuery ML allows you to:

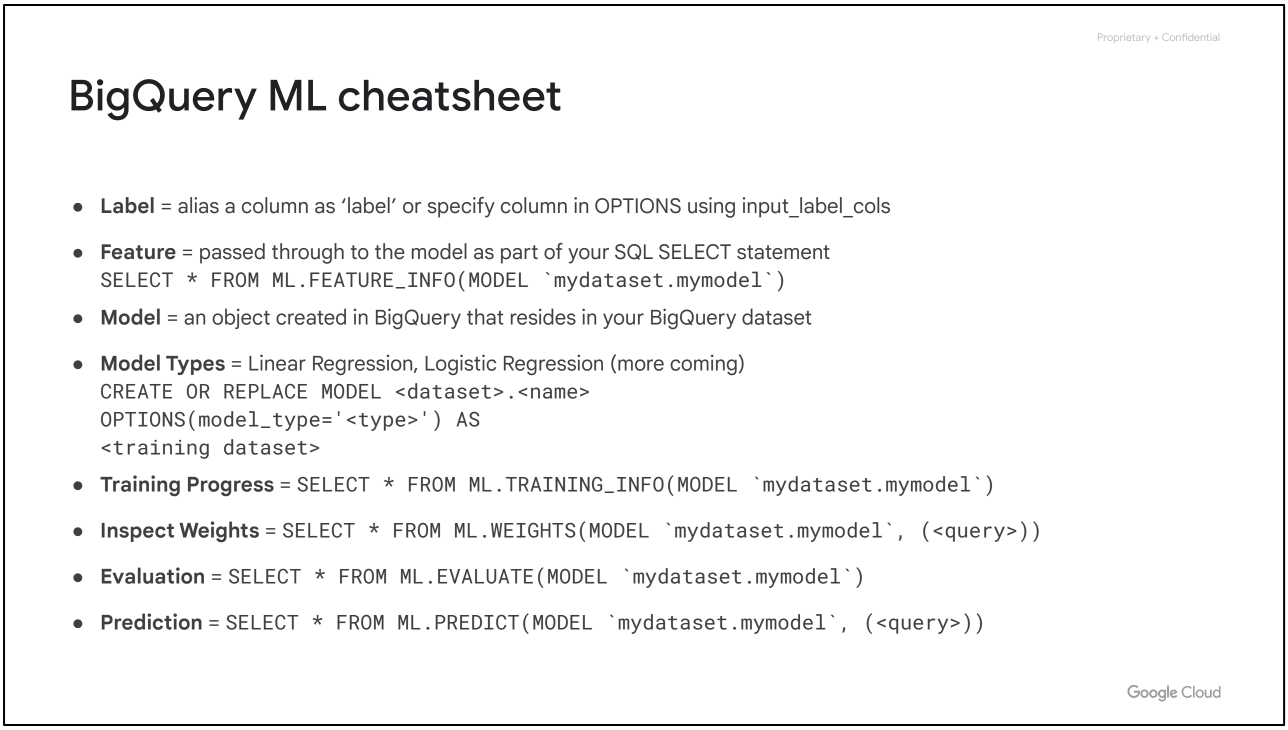


Demo

Creating a Machine Learning Model with SQL

Predicting visitor purchases

Refer to https://github.com/GoogleCloudPlatform/training-data-analyst/blob/master/courses/data-to-insights/demos/bqml-classification-model.sql

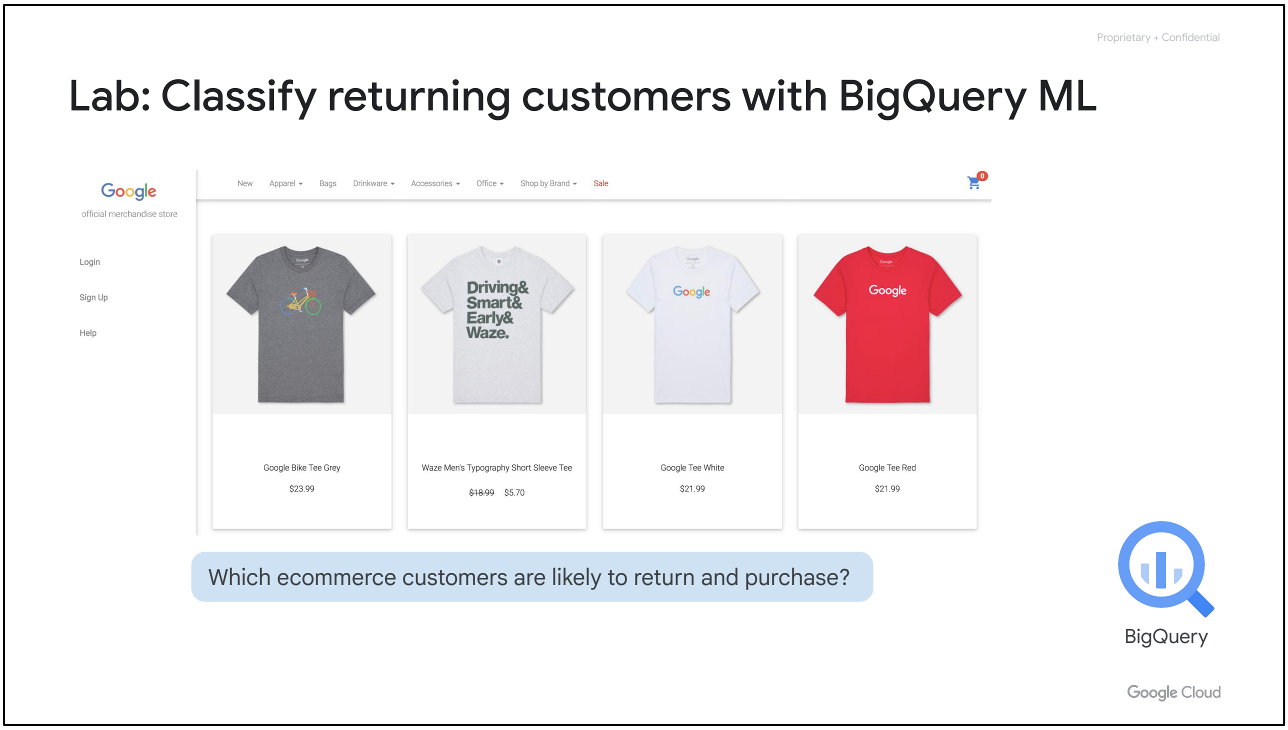


BigQuery ML cheatsheet

- **Label** = alias a column as ‘label’ or specify column in OPTIONS using input_label_cols
- **Feature** = passed through to the model as part of your SQL SELECT statement
  `SELECT * FROM ML.FEATURE_INFO(MODEL `mydataset.mymodel`)`
- **Model** = an object created in BigQuery that resides in your BigQuery dataset
- **Model Types** = Linear Regression, Logistic Regression (more coming)
  ```
  CREATE OR REPLACE MODEL <dataset>.<name>
  OPTIONS(model_type='<type>') AS
  <training dataset>
  ```
- **Training Progress** = `SELECT * FROM ML.TRAINING_INFO(MODEL `mydataset.mymodel`)`
- **Inspect Weights** = `SELECT * FROM ML.WEIGHTS(MODEL `mydataset.mymodel`, (<query>))`
- **Evaluation** = `SELECT * FROM ML.EVALUATE(MODEL `mydataset.mymodel`)`
- **Prediction** = `SELECT * FROM ML.PREDICT(MODEL `mydataset.mymodel`, (<query>))`

Lastly, it’s as simple as writing ML.PREDICT and referencing your model and prediction dataset to get predictions.

An important note here is that when using ML.PREDICT and passing in a new dataset with an unknown label you can add in other columns that you didn’t train on initially. The model is not being re-trained during prediction. Note that if you happen to REMOVE or RENAME columns from your prediction dataset that the model was expecting then you will be given an error.



Proprietary + Confidential

Lab: Classify returning customers with BigQuery ML

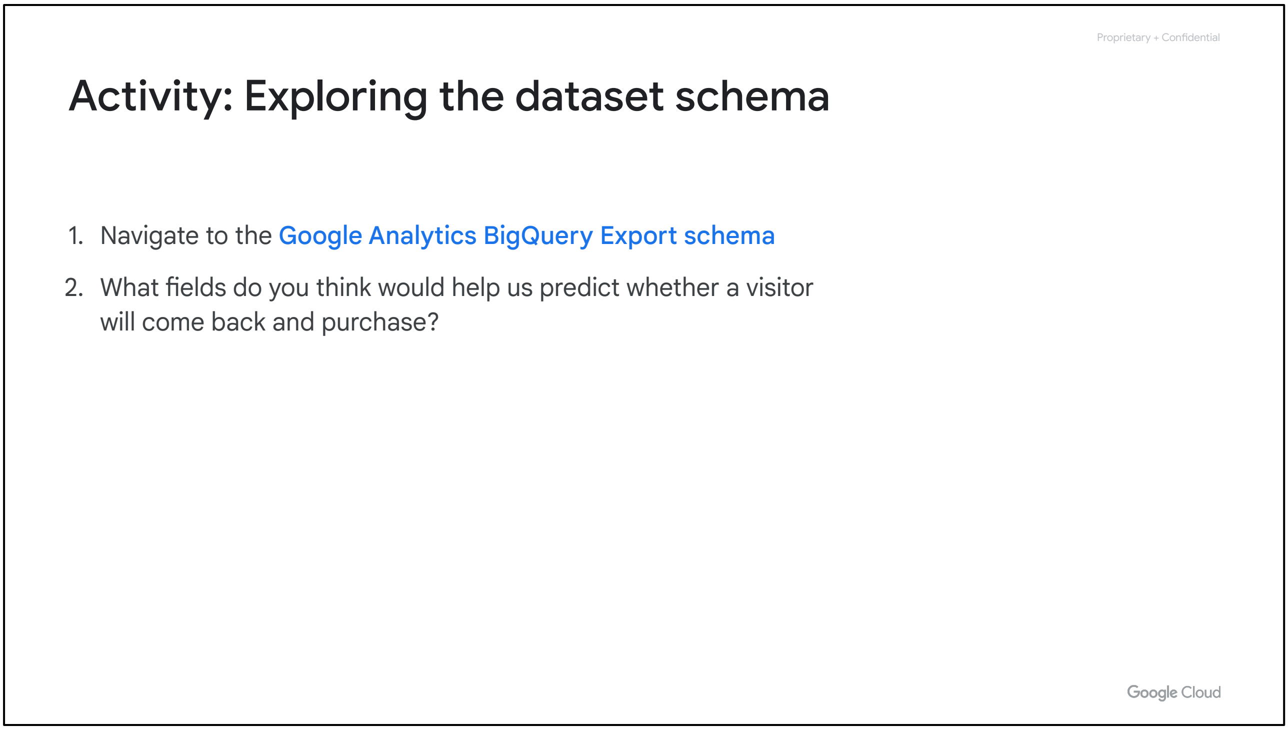
Which ecommerce customers are likely to return and purchase?

BigQuery

Google Cloud

Next, you'll explore the public BigQuery dataset for the San Francisco bike share program. You'll look at the number of stations and the trips the bikes have taken, and analyze the capacity and efficiency of each station to build a demand forecasting model.

Two aspects make this problem unique. First, you won't have to set up any servers to hold your datasets or to run your queries. And second, you'll be building your ML model using SQL right within BigQuery. You'll see a demo of how this is done a little bit later.



Proprietary + Confidential

Activity: Exploring the dataset schema

1. Navigate to the Google Analytics BigQuery Export schema
2. What fields do you think would help us predict whether a visitor will come back and purchase?

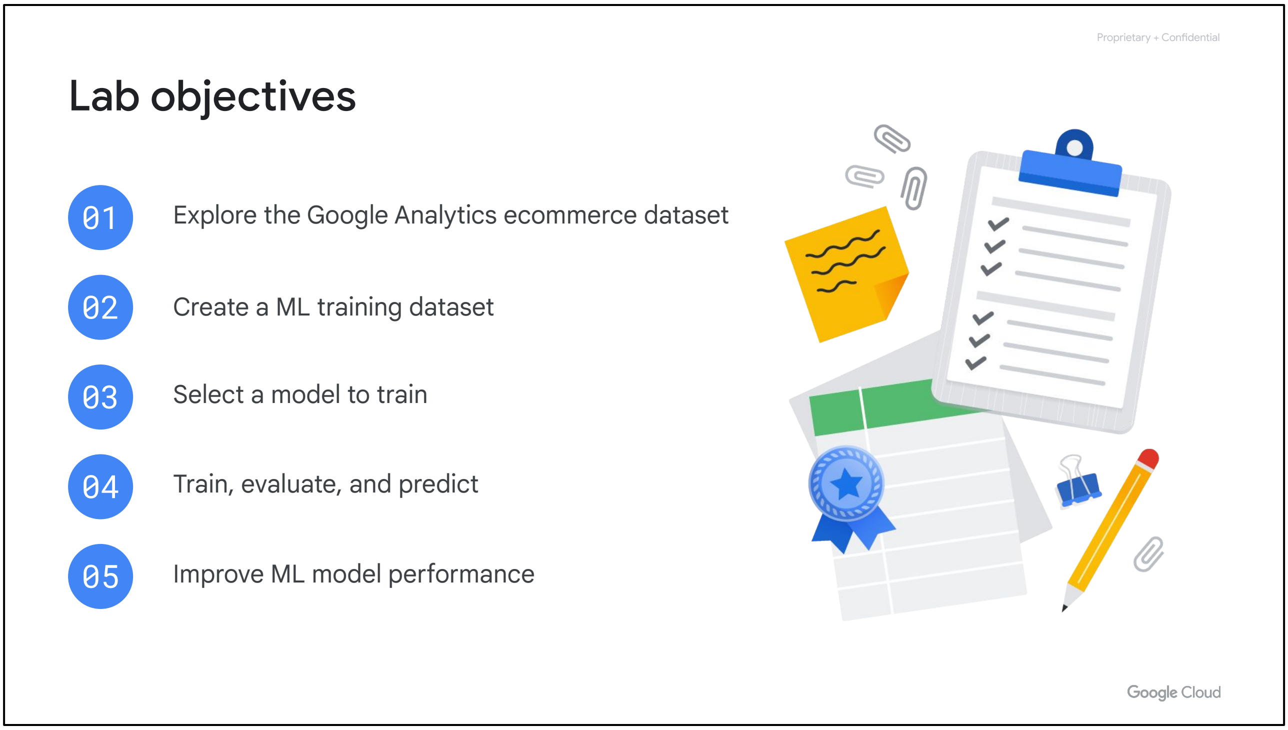
Google Cloud

Next, you’ll explore the public BigQuery dataset for the San Francisco bike share program. You’ll look at the number of stations and the trips the bikes have taken, and analyze the capacity and efficiency of each station to build a demand forecasting model.

Two aspects make this problem unique. First, you won’t have to set up any servers to hold your datasets or to run your queries. And second, you’ll be building your ML model using SQL right within BigQuery. You’ll see a demo of how this is done a little bit later.

Lab Intro

Predicting Visitor Purchases with
BigQuery ML



Lab objectives

01 Explore the Google Analytics ecommerce dataset

02 Create a ML training dataset

03 Select a model to train

04 Train, evaluate, and predict

05 Improve ML model performance