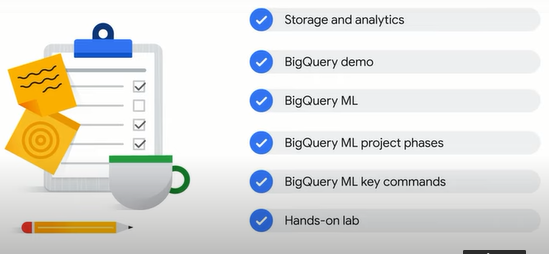
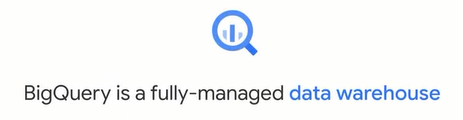
**BigData with Big Query.**

**Introduction.**

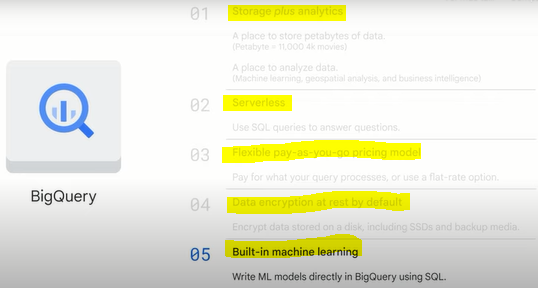
in the previous section of this course you explored dataflow and pub sub google cloud solutions to processing streaming data now let's focus your attention on bigquery you'll begin by exploring bigquery's two main services storage and analytics and then get a demonstration of bigquery in use after that you'll see how bigquery ml provides a data to ai lifecycle all within one place you'll also learn about bigquery ml project phases as well as key commands finally you'll get hands-on practice using bigquery ml to build a custom ml model



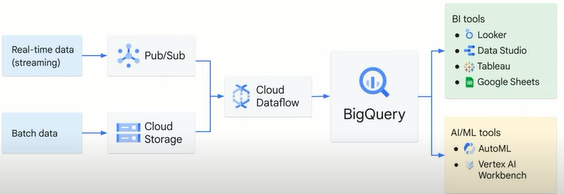
let's get started bigquery is a fully managed data warehouse.



A data warehouse is a large store containing terabytes and petabytes of data gathered from a wide range of sources within an organization that's used to guide management decisions being fully managed means that bigquery takes care of the underlying infrastructure so you can focus on using sql queries to answer business questions without worrying about deployment scalability and security let's look at some of the key features of bigquery bigquery provides two services in one storage plus analytics it's a place to store petabytes of data for reference one petabyte is equivalent to 11 000 movies at 4k quality bigquery is also a place to analyze data with built-in features like machine learning geospatial analysis and business intelligence which we'll explore a bit later on bigquery is a fully managed serverless solution meaning that you can use sql queries to answer your organization's biggest questions in the front end without worrying about infrastructure in the back end if you've never written sql before don't worry this course provides resources and labs to help bigquery has a flexible pay as you go pricing model where you pay for the number of bytes of data your query processes and for any permanent table storage if you prefer to have a fixed bill every month you can also subscribe to flat rate pricing where you have a reserved amount of resources for use data in bigquery is encrypted at rest by default without any action required from a customer by encryption at rest we mean encryption used to protect data that is stored on a disk including solid state drives or backup media bigquery has built-in machine learning features so you can write ml models directly in bigquery using sql also if you decide to use other professional tools such as vertex ai from google cloud to train your ml models you can export data sets from bigquery directly into vertex ai for a seamless integration across the data to ai lifecycle



so what does a typical data warehouse solution architecture look like the input data can be either real time or batch data if you think back to the last section of the course you'll recall that there are four challenges of big data in modern organizations they are that data can be any format variety any size volume any speed velocity and possibly inaccurate veracity if it's streaming data which can be either structured or unstructured high speed and large volume pub sub is needed to digest the data if it's batch data it can be directly uploaded to cloud storage after that both pipelines lead to data flow to process the data data flow is where we etl extract transform and load the data if needed bigquery sits in the middle to link data processing using dataflow and data access through analytics ai and ml tools the job of the analytics engine of bigquery at the end of a data pipeline is to ingest all the processed data after etl store and analyze it and possibly output it for further use such as data visualization and machine learning bigquery outputs usually feed into two buckets business intelligence tools and ainml tools if you're a business analyst or data analyst you can connect to visualization tools like looker data studio tableau and other bi tools if you prefer to work in spreadsheets you can query both small or large bigquery data sets directly from google sheets and even perform common operations like pivot tables alternatively if you're a data scientist or machine learning engineer you can directly call the data from bigquery through automl or workbench these ainml tools are part of vertex ai google's unified ml platform



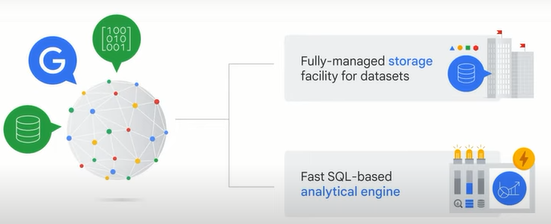
**ETL**

bigquery is like a common staging area for data analytics workloads when your data is there business analysts bi developers data scientists and machine learning engineers can be granted access to your data for their own insights

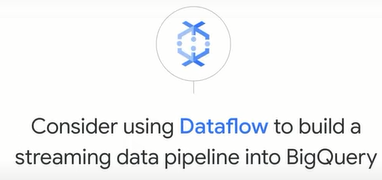
**Storage and analytics.**

bigquery provides two services in one it's both a fully managed storage facility to load and store data sets and also a fast sql-based analytical engine

the two services are connected by google's high-speed internal network it's the super fast network that allows bigquery to scale both storage and compute independently based on demand let's look at how bigquery manages the storage and metadata for data sets



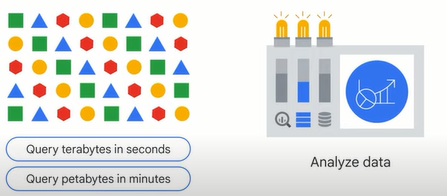
bigquery can ingest data sets from various sources including internal data which is data saved directly in bigquery external data multi-cloud data and public data sets after the data is stored in bigquery it's fully managed and is automatically replicated backed up and set to auto scale bigquery also offers the option to query external data sources like data stored in other google cloud storage services such as cloud storage or in other google cloud database services such as spanner or cloud sql and bypass bigquery manage storage this means a raw csv file in cloud storage or a google sheet can be used to write a query without being ingested by bigquery first one thing to note here inconsistency might result from saving and processing data separately to avoid that risk consider using dataflow to build a streaming data pipeline into bigquery



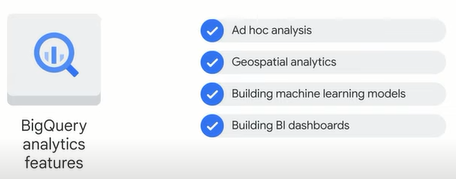
in addition to internal or native and external data sources bigquery can also ingest data from multi-cloud data which is data stored in multiple cloud services such as aws or azure or a public data set if you don't have any data of your own you can analyze any of the data sets available in the public dataset marketplace there are three basic patterns to load data into bigquery the first is a batch load where source data is loaded into a bigquery table in a single batch operation this can be a one-time operation or it can be automated to occur on a schedule a batchload operation can create a new table or append data into an existing table the second is streaming where smaller batches of data are streamed continuously so that the data is available for querying in near real time and the third is generated data where sql statements are used to insert rows into an existing table or to write the results of a query to a table



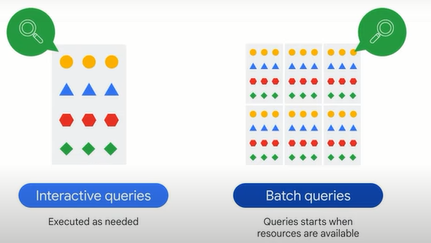
of course the purpose of bigquery is not to just save data it's for analyzing data and helping to make business decisions bigquery is optimized for running analytical queries over large data sets it can perform queries on terabytes of data in seconds in petabytes in minutes this performance lets you analyze large data sets efficiently and get insights in near real time



let's look at the analytics features that are available in bigquery bigquery supports ad-hoc analysis using standard sql the bigquery sql dialect geospatial analytics using geography data types and standard sql geography functions bigquery supports building machine learning models using bigquery ml and building rich interactive business intelligence dashboards using bigquery bi engine



by default bquery runs interactive queries which means that the queries are executed as needed bigquery also offers batch queries where each query is queued on your behalf and the query starts when idle resources are available usually within a few minutes.



up next you'll see a demonstration in bigquery please note that you might notice a slightly different user interface.

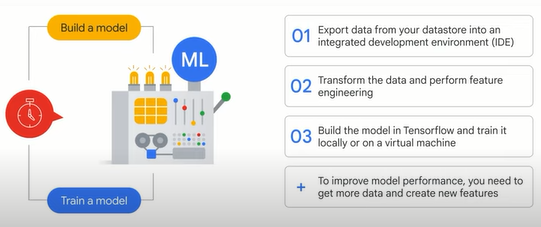
**BigQuery demo - San Francisco bike share.**

as any data analysts will tell you exploring your data set with sql is often one of the first steps that you take to uncover those hidden insights so what is this query actually doing care to take a guess here we're counting the total number of trips taken on a public data set for san francisco as bike share trips now let's explore this public data set together in greater detail with a quick demonstration it's now time to dust off your sql or structure qree language skills as we explore some public data sets inside of google bigquery let's dive right in so in this demo how it got to this public bigquery data set which is what we were looking at before the san francisco bike share bike share trips is from this publicly available data set bigquery public data now if you might not have that inside of your quick labs account or your own personal accounts how you can get those data sets is by clicking add data explore public data sets choosing one of those and it brings you right back into the console so let me show you a couple nifty things that i've picked up inside of bigquery over the years so now we have a ton of bigquery public data sets that you can experiment with once you've found one you like you can do a lot just knowing the name of the table so we're not going to write any sql yet so i just got the table name notice if it has hyphens in it so we named the bigquery project with the bigquery public data with hyphens that's when you need those back ticks now technically you only need them around the project name but that's when you see those backticks which is that character there come into play so if you don't have hyphens anywhere in here you actually don't need those that's the first tip for you second tip one of my absolute favorites if you hold down at least on this mac here it's going to be the command key or the windows key it'll highlight all the data sets in your query why is that useful because as soon as it's highlighted you can click on it so if you have a bunch of different data sets that you just want to explore or you inherited a query from somebody else you can quickly get to this schema the details in the preview of that particular table so we're trying to get as far without writing any any keystrokes in here so take a look at the schema this is san francisco bike share trips you've got things like the trip id everything we could possibly want the duration and the seconds when it started what stations are and this is let your kind of mind wander for some of the cool insights that you could take a look at geographic data we've got start latitude longitude and latitude and longitude maybe some of those cool gis functions and let's take a look at how many rows we have we don't have to do count star or anything like that we have almost 2 million rows and about 375 megabytes into there and in best practice you don't want to do select count star from a table limit 10 when just looking at a preview would suffice so you saw the schema and the data type now you can actually go in here look at all the total rows and c sample data values so you can see the station where they stopped and started everything that'll get you a familiar familiarity with it if you wanted to query the table you can click on query table it'll ask you hey i'm going to overwrite everything in the query editor sure that's fine and it's going to say select some columns from our bike share trips here's a neat tip it's one i just figured out recently and props to the bigquery team if i just wanted to see say uh the total number of trips that started from the station name i'm just clicking on the field names especially if you have field names that are really long or if you have you know you don't want to type things in like i do in a live demo then you can just click on these field names and it'll add them and it'll even add the commas for you which is kind of cool and then what you can do inside of bigquery is you can format the query again even if you're experiencing sql this is still kind of fun to try to do as much as you can without even typing anything into the editor we're gonna run that query it doesn't really do much for us because it's just gonna give the station name it's still at the granularity level of the individual trip so what's one of the things that you could do inside of bigquery if you're looking at that trip id we can say I want to have an aggregate function like a count of all trips now it doesn't it's just counting the rows you could do count stars some people like to put count one or something like that for readability i'm going to keep the count of trip id just so somebody else inheriting my code can very quickly see what their level of granularity of this table and we'll call this the number of trips now naturally immediately you're going to get an error if you work with sql long enough as soon as you do an aggregation in one field all the rest of your other fields better be aggregated as well but if it's just been a late night you might just open up the validator you can see hey uh this references this start station name your other person here is aggregated but this one isn't then you naturally what you want to do you want to make sure that you do the group by and that's how we can say all right well we're rolling up all the trips into a single value let's group those by each of the different stations now if you remember your sql what's one of the things that you can do to get the most first you can do an order by which is going to be a sort and you can order by the alias as we see here now that you actually can't filter in a where clause by an alias field because it actually doesn't exist uh when the when the query engine goes out and performs that where clause so keep that in mind that's where you can use things like temporary tables so orders by the highest number of chips first we'll just let's say the top 10 stations or something like that let's make sure we'll format this if you want to be a really in evan's good favor you can add a comment at the top this is like the top 10 stations by volume and go ahead and run that and we'll see the most popular stations and we've got the san francisco caltrain i can definitely vouch for this one as soon as you get off the train you need to get somewhere in san francisco and it's got 72 000 trips now if you want to experiment a little bit more what you can do is you can add in a filter and just say hey i'm looking for just those trips for 2018 there's another field you can go back hey if you've honestly i've already forgotten the field name i hold down that button and then i'm going to pop in the the start date let's take a look at that start date start date is a time stamp so let's see we can just uh where the start date is say after uh what did we say 2018 so we can do 2017. uh end of 2017. uh 12 31 sure after all this you can convert it if you wanted to hopefully this will this will just take automatically there's a lot of date functions and extractions that you can do but let's see i think it was was it before like 20 000 or something like that uh [Music] 70 000 i think for the caltrain let's see if the caltrain is still number one ah look in the last year caltrain is actually dethroned it's the ferry building super popular tourist spot if you haven't been to san francisco yet and that takes the highest number of chips just for 2018. now if you're doing a lot of this aggregation inside of bigquery and then filtering it's fun for these types of insights but an easier way that you can do it a lot of times is by actually just exporting the data or linking it [Music] directly from a front-end visualization tool like data studio so you can actually say oh i don't want to actually limit the data here i want to throw all of it at a visualization tool and then you can just have a filter for the users that actually has um has the results if i want to store this common query i could do a couple different things i could save the query as my query inside of my project personally just for me or for all people that can see it or i can save it as a view but what i like to do since i commonly commit all my code to version control is i actually do a little bit of data definition language in sql sql ddl and that says i want to create on a creation statement inside of the actual code itself so people if they wonder hey where on earth did you get this top 2018 uh and actually it's not just 2018 it's anything after uh in 2017 as well for people yelling into their screens like hey it's not just 2018. i got you create a replaced table and we can just give it a table name we'll just call we need to we need a data set do we even have a data set i don't think so we need to have a data set first before we're just dumping things in here we call this bike insights or something like that data set is just a collection so now we have an empty data set there's no tables in there so we can start populating one so we have bike insights so create a replace table in the data set and we'll just call this top trips uh 2018 and beyond try to keep your table names a little bit uh more succinct than mine and then once you've created that you should automatically see this show up boom there you have it and now you can actually have that their queries not going to be rewritten every single time now you might be asking uh what happens if the bigquery public data source updates after this that's an excellent excellent point so how would you your table this is this just dumped all of this data here it's going to be static now so one of the things that you can do is if you're looking to link this out to your dashboard instead of creating a table you could simply create a view which a logical view what that means is every single time and we'll just call this view since it says hey that object already exists every single time you can notice the icon changes and you can actually click into here and you can look inside of the table you can actually preview the data inside of the view that preview is gone why is that because the view is just an empty object it's a view is essentially a logical view in this in this instance of sql is just a stored query so if you're trying to query the view we can query the view and then that actually just runs against the query that we we had stored a little bit earlier so it's high level recap you've seen data sets you've seen public data sets you can explore them at your leisure uh some of the neat tips and tricks that you've seen if you want to actually then go in and edit that view later you can scroll down bring that result back into here look back at that schema and keep exploring as you see fit this is just the tip of the iceberg as when it comes to data analysis we just did a simple account of chips by the most popular station names there's also data sets in there for weather so you can do a fun analysis to see we have a bike share data set for new york or san francisco see where weather affects the most the riders the most or how the bike share program is doing as a whole when it comes to seasonality is there a lot of more more or less ridership in some of the winter months um you can check that out all all for yourself but you can do it all with with sql and not have to worry about building any of the infrastructure behind the scenes all that's managed for you inside of bigquery all right that's a wrap.

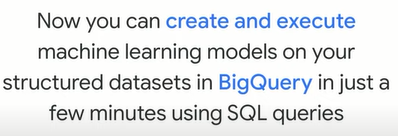
**Introduction to BigQuery ML.**

although bigquery started out solely as a data warehouse over time it has evolved to provide features that support the data to ai lifecycle (*aunque bigquery comenzó únicamente como un almacén de datos, con el tiempo ha evolucionado para proporcionar funciones que respaldan los datos para el ciclo de vida de ia*).

in this section of the course we'll explore bigquery's capabilities for building machine learning models in the ml project phases and walk you through the key ml commands in sql if you've worked with ml models before you know that building and training them can be very time intensive you must first export data from your data store into an ide integrated development environment such as jupyter notebook or google collab and then transform the data and perform your feature engineering steps before you can feed it into a training model then finally you need to build the model in tensorflow or similar library and train it locally on a computer or on a virtual machine to improve the model performance you also need to go back and forth to get more data and create new features this process will need to be repeated but it's so time intensive that you'll probably stop after a few iterations

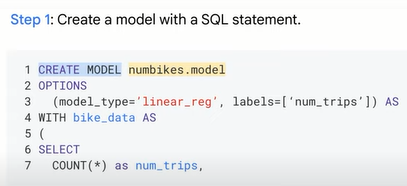


also we just mentioned tensorflow and feature engineering in the past if you weren't familiar with these technologies ml was left to the data scientists on your team it was not available to you now you can create and execute machine learning models on your structured data sets in bigquery in just a few minutes using sql

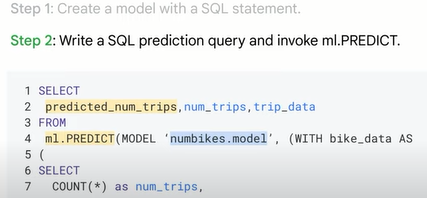


there are two steps needed to start

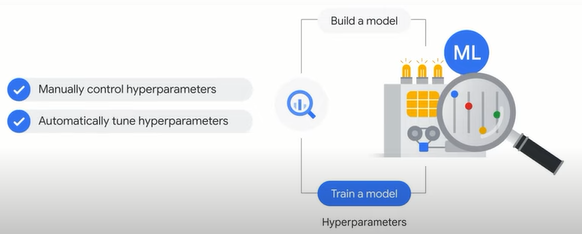
**step 1 create a model** with a sql statement here we can use the bike share data set as an example



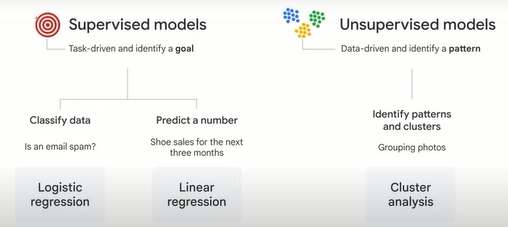
**step 2 write a sql prediction query and invoke ml.predict** and that's it you now have a model and can view the results additional steps might include activities like evaluating the model but if you know basic sql you can now implement ml that's pretty cool



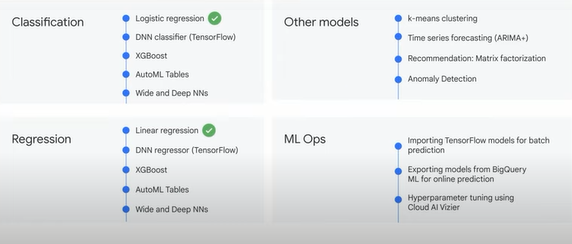
bigquery ml was designed to be simple like building a model in two steps that simplicity extends to defining the machine learning hyper parameters which let you tune the model to achieve the best training result hyperparameters are the settings apply to a model before the training starts like a learning rate with bigquery ml you can either manually control the hyper parameters or hand it to bigquery starting with a default hyperparameter setting and then automatic tuning



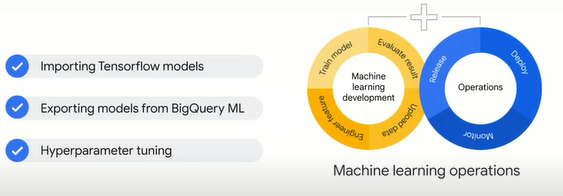
when using a structured data set in bigquery ml you need to choose the appropriate model type choosing which type of ml model depends on your business goal and the data sets bigquery supports supervised and unsupervised models supervised models are task driven and identify a goal alternatively unsupervised models are data driven and identify a pattern within a supervised model if your goal is to classify data like whether an email is spam use logistic regression if your goal is to predict a number like shoe sales for the next three months use linear regression within an unsupervised model if your goal is to identify patterns or clusters and then determine the best way to group them like grouping random photos of flowers into categories you should use cluster analysis



once you have your problem outlined it's time to decide on the best model categories include classification and regression models there are also other model options to choose from along with ml ops logistic regression is an example of a classification model and linear regression is an example of a regression model we recommend that you start with these options and use the results to benchmark to compare against more complex models such as dnn (deep neural networks) which may take more time and computing resources to train and deploy

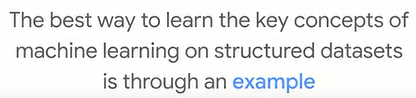


in addition to providing different types of machine learning models bigquery ml supports features to deploy monitor and manage the ml production called mlops which is short for machine learning operations options include importing tensorflow models for batch prediction exporting models from bigquery ml for online prediction and hyperparameter tuning using cloud ai vizier we'll explore mlaps in more detail later in this course.

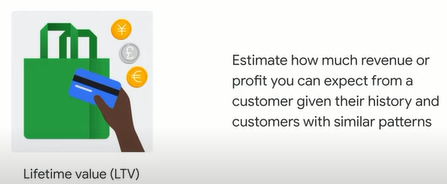


**Using BigQuery ML to predict customer lifetime value.**

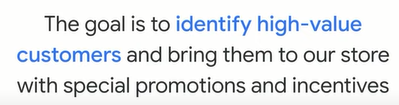
now that you're familiar with the types of ml models available to choose from high quality data must be used to teach the models what they need to learn the best way to learn the key concepts of machine learning on structured data sets is through an example.



This scenario we'll predict customer lifetime value with a model lifetime value or ltv is a common metric in marketing used to estimate how much revenue or profit you can expect from a customer given their history and customers with similar patterns



we'll use a google analytics ecommerce data set from google's own merchandise store that sells branded items like t-shirts and jackets the goal is to identify high-value customers and bring them to our store with special promotions and incentives



having explored the available fields you may find some useful in determining whether a customer is high value based on their behavior on our website these fields include customer lifetime page views total visits average time spent on the site total revenue brought in and e-commerce transactions on the site



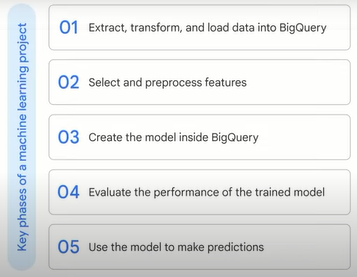
remember that in machine learning you feed in columns of data and let the model figure out the relationship to best predict the label it may turn out that some of the columns weren't useful at all to the model in predicting the outcome you'll see later how to determine this now that we have some data we can prepare to feed it into the model incidentally to keep this example simple we're only using seven records but we'd need tens of thousands of records to train a model effectively before we feed the data into the model we first need to define our data and columns in the language that data scientists and other ml professionals use using the google merchandise store example a record or row in the data set is called an example an observation or an instance **a label is a correct answer and you know it's correct because it comes from historical data this is what you need to train the model on in order to predict future data depending on what you want to predict a label can either be a numeric variable which requires a linear regression model or a categorical variable which requires a logistic regression** model for example if we know that a customer who has made transactions in the past and spends a lot of time on our website often turns out to have high lifetime revenue we could use revenue as the label and predict the same for newer customers with that same spending trajectory this means forecasting a number so we can use a linear regression as a starting point to model labels could also be categorical variables like binary values such as high value customer or not to predict a categorical variable if you recall from the previous section you need to use a logistic regression model.

knowing what you're trying to predict such as a class or a number will greatly influence the type of model you'll use.

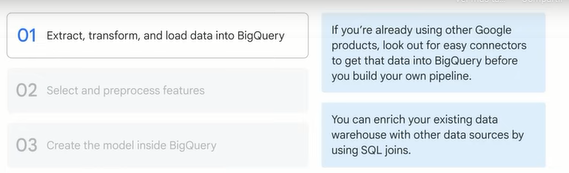
but what do we call all the other data columns in the data table those columns are called features or potential features each column of data is like a cooking ingredient you can use from the kitchen pantry too many ingredients however can ruin a dish the process of sifting through data can be time consuming understanding the quality of the data in each column and working with teams to get the most features or more history is often the hardest part of any ml project you can even combine or transform feature columns in a process called feature engineering if you've ever created calculated fields in sql you've already executed the basics of feature engineering also bigquery ml does much of the hard work for you like automatically one hot encoding categorical values one hot encoding is a method of converting categorical data to numeric data to prepare it for model training from there bigquery ml automatically splits the data set into training data and evaluation data and finally there is predicting on future data let's say new data comes in that you don't have a label for so you don't know whether it's for a high-value customer you do however have a rich history of labeled examples for you to train a model on so if we train a model on the known historical data and are happy with the performance then we can use it to predict our future data sets

**BigQuery ML project phases.**

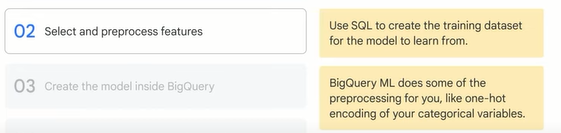
let's explore the key phases of a machine learning project



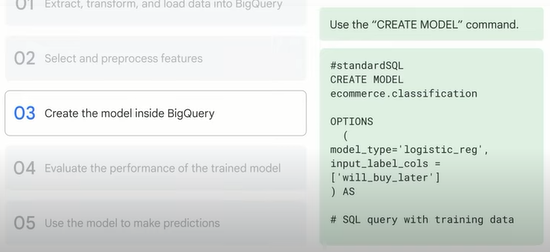
**in phase 1** you extract transform and load data into bigquery if it isn't there already if you're already using other google products like youtube for example look out for easy connectors to get the data into bigquery before you build your own pipeline you can enrich your existing data warehouse with other data sources by using sql joins



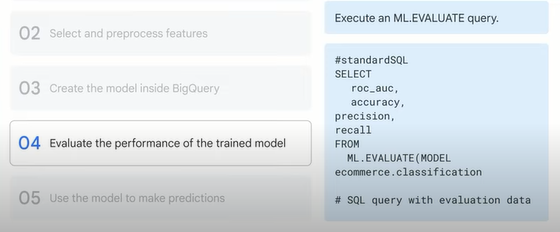
**in phase 2** you select and pre-process features you can use sql to create the training data set for the model to learn from you'll recall that bigquery ml does some of the pre-processing for you like one-hot encoding of your categorical variables one hot encoding converts your categorical data into numeric data that is required by a training model



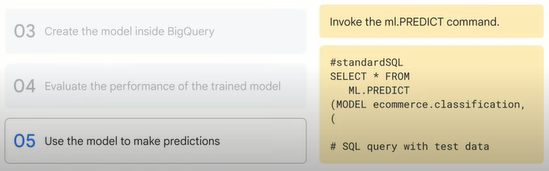
**in phase 3** you create the model inside bigquery this is done by using the create model command give it a name specify the model type and pass it in a sql query with your training data set from there you can run the query



**in phase 4** after your model is trained you can execute an ml.evaluate query to evaluate the performance of the trained model on your evaluation data set it's here that you can analyze loss metrics like a root mean squared error for forecasting models and area under the curve accuracy precision and recall for classification models we'll explore these metrics later in the course.

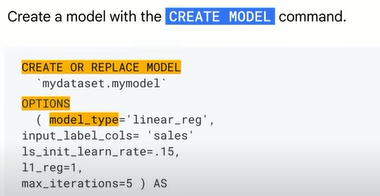


**in phase 5** the final phase when you're happy with your model performance you can then use it to make predictions to do so invoke the ml.predict command on your newly trained model to return with predictions and the model's confidence in those predictions with the results your label field will have predicted added to the field name this is your model's prediction for that label**.**

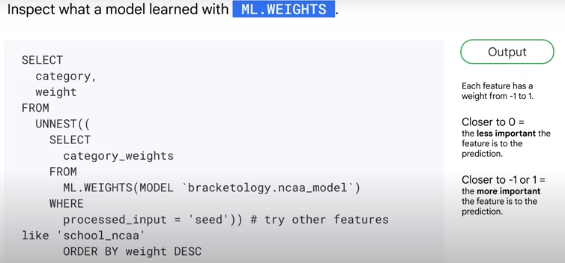


**BigQuery ML key commands.**

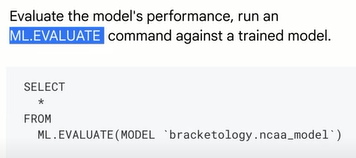
now that you're familiar with the key phases of an ml project let's explore some of the key commands of bigquery ml you'll remember from an earlier video that you can create a model with just the create model command if you want to overwrite an existing model use the **create or replace** model command models have options which you can specify the most important and the only one required is the model type



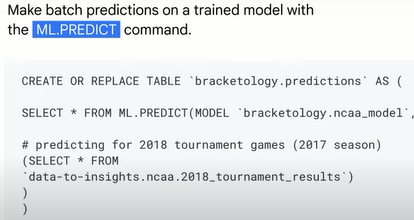
you can inspect what the model learned with the **ml.weights** command and filtering on an input column the output of ml.weights is a numerical value and each feature has a weight from negative 1 to 1. the value indicates how important the feature is for predicting the result or label if the number is closer to zero the feature isn't important for the prediction however if the number is closer to negative one or one then the feature is more important for predicting the result



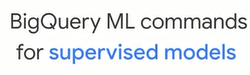
to evaluate the model's performance you can run an **ml.evaluate** command against a trained model you get different performance metrics depending on the model type you choose



and if you want to make batch predictions you can use the **ml.predict** command on a trained model and pass through the data set you want to make the prediction on



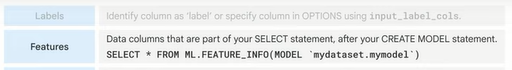
now let's explore a consolidated list of bigquery ml commands for supervised models



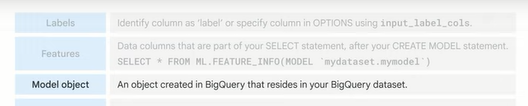
first in bigquery ml you need a field in your training data set titled label or you need to specify which field or fields your labels are using as the input label columns in your model options



second your model features are the data columns that are part of your select statement after your create model statement after a model is trained you can use the ml.feature info command to get statistics and metrics about the column for additional analysis



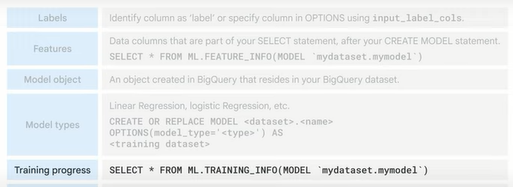
next is the model object itself this is an object created in bigquery that resides in your bigquery data set you train many different models which will all be objects stored under your bigquery data set much like your tables and views model objects can display information for when it was last updated or how many training runs it completed



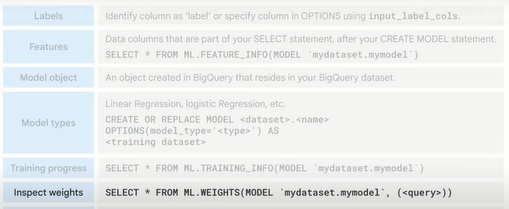
creating a new model is as easy as writing create model choosing a type and passing in a training data set **again if you're predicting on a numeric field such as next year's sales consider linear regression for forecasting if it's a discrete class like high medium low or spam or not spam consider using logistic regression for classification**



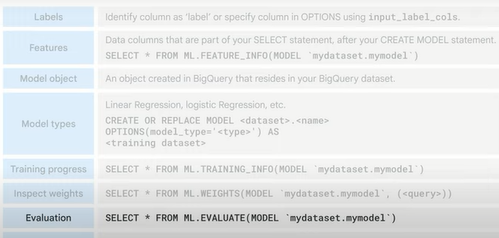
while the model is running and even after it's complete you can view training progress with ml.training info



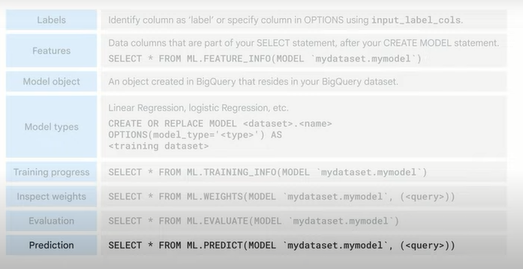
as mentioned earlier you can inspect weights to see what the model learned about the importance of each feature as it relates to the label you're predicting the importance is indicated by the weight of each feature



you can see how well the model performed against its evaluation data set by using ml.evaluate

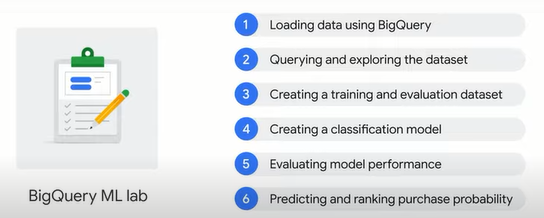


and lastly getting predictions is as simple as writing ml.predict and referencing your model name and prediction data set.



**Lab introduction: Predicting Visitor Purchases Using BigQuery ML.**

now it's time to get some hands-on practice building a machine learning model in bigquery in the lab that follows this video you'll use e-commerce data from the google merchandise store website shop.googlemerchandisestore.com the site's visitor and order data have been loaded into bigquery and you'll build a machine learning model to predict whether a visitor will return for more purchases later you'll get practice loading data into bigquery from a public data set querying and exploring the e-commerce data set creating a training and evaluation data set to be used for batch prediction creating a classification logistic regression model in bigquery ml evaluating the performance of your machine learning model and predicting and ranking the probability that a visitor will make a purchase let's get started.



**Lab: Predicting Visitor Purchases with a Classification Model 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

## 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 (sin procesar) 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? (*Del total de visitantes que visitaron nuestro sitio web, ¿qué % realizó una compra?*)

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

1. Click **Run**.

The result: 2.69%

**Question:** What are the top 5 selling products?

1. 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?

1. 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). (*Este comportamiento es muy común para los artículos de lujo en los que el cliente requiere una investigación y una comparación significativas por adelantado antes de decidir (piense en la compra de automóviles), pero también es cierto, en menor medida, para la mercancía en este sitio (camisetas, accesorios, 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. *(En el mundo del marketing en línea, identificar y comercializar a estos futuros clientes en función de las características de su primera visita aumentará las tasas de conversión y reducirá la salida a los sitios de la competencia.)*

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

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](https://support.google.com/analytics/answer/3437719?hl=en) and then [preview the demo dataset](https://bigquery.cloud.google.com/table/data-to-insights:ecommerce.web_analytics?tab=preview) 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) (*Si un usuario rebota está altamente correlacionado con su tiempo en el sitio*)

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 features are bounces and time\_on\_site. The label is will\_buy\_on\_return\_visit**

The feature is will\_buy\_on\_return\_visit. The labels are bounces and time\_on\_site

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)

**bounces**

**time\_on\_site**

will\_buy\_on\_return\_visit

**visitId**

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

visitId

bounces

**will\_buy\_on\_return\_visit**

time\_on\_site

**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.

(*will\_buy\_on\_return\_visit no se conoce después de la primera visita. Nuevamente, está prediciendo para un subconjunto de usuarios que regresaron a su sitio web y compraron. Dado que no conoce el futuro en el momento de la predicción, no puede decir con certeza si un nuevo visitante regresa y compra. El valor de construir un modelo ML es obtener la probabilidad de una compra futura en función de los datos recopilados sobre su primera sesión*.)

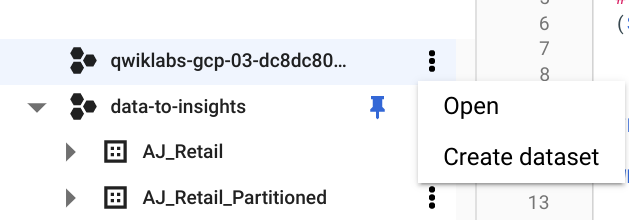
**Question:** Looking at the initial data results, do you think **time\_on\_site** and **bounces** will be a good indicator of whether the user will return and purchase or not? (*Mirando los resultados de los datos iniciales, ¿cree que el tiempo en el sitio y los rebotes serán un buen indicador de si el usuario regresará y comprará o no?)*

**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. (*A menudo es demasiado pronto para saberlo antes de entrenar y evaluar el modelo, pero a primera vista de los 10 principales time\_on\_site, solo 1 cliente volvió a comprar, lo que no es muy prometedor. Vamos a ver qué tan bien lo hace el modelo.*)

## 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**.



1. In the **Create Dataset** dialog:

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

1. 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](https://www.tensorflow.org/tutorials/" \t "_blank). 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)

;

1. 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".

1. Click **Go to model**.

Look inside the ecommerce dataset and confirm **classification\_model** now appears.

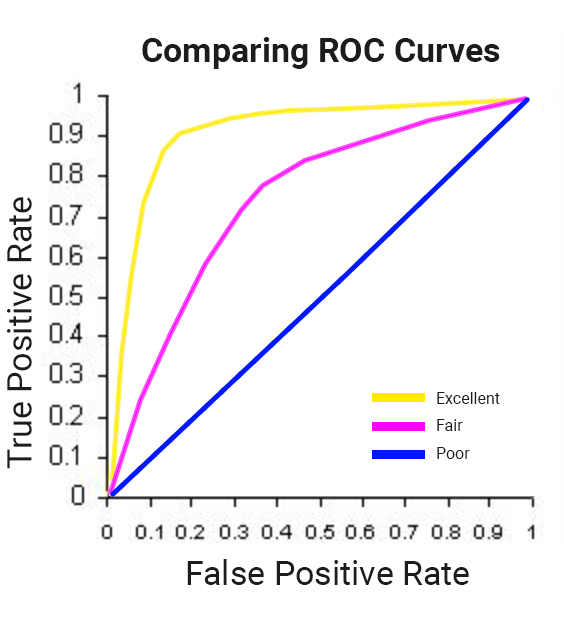
Next, you will evaluate the performance of the model against new unseen evaluation data.

## 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](https://support.google.com/analytics/answer/3437719?hl=en) you will see the field mapping of 6 = Completed Purchase.

As an aside, the web analytics dataset has nested and repeated fields like [ARRAYS](https://cloud.google.com/bigquery/docs/reference/standard-sql/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).

1. 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%)

## 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**

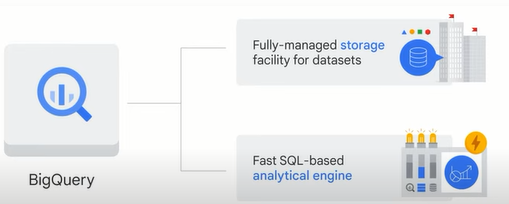
**roc\_auc** is just one of the performance metrics available during model evaluation. Also available are [accuracy, precision, and recall](https://en.wikipedia.org/wiki/Precision_and_recall). Knowing which performance metric to rely on is highly dependent on what your overall objective or goal is.

### **Congratulations!**

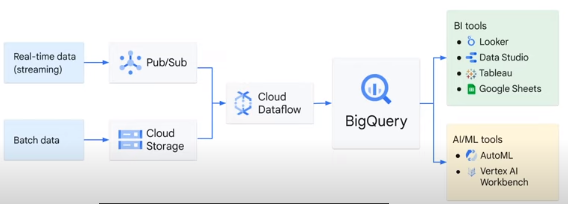
You created a machine learning model using just SQL.

**Summary.**

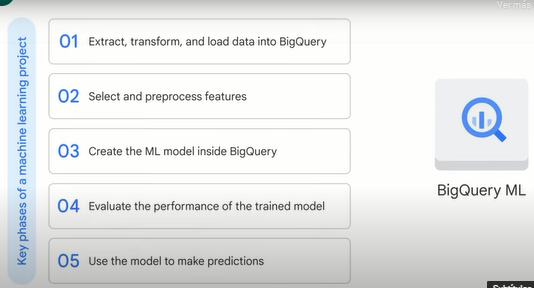
well done on completing another lab hopefully you now feel more comfortable building custom machine learning models with bigquery ml let's review what we explored in this section of the course our focus was on bigquery the data warehouse that provides two services in one it's a fully managed storage facility for data sets and a fast sql-based analytical engine



bigquery sits between data processes and data uses like a common staging area it gets data from ingestion and processing and outputs data to bi tools such as looker and data studio and ml tools such as vertex ai after the data is in bigquery business analysts bi developers data scientists and machine learning engineers can be granted access to the data for their own insights



in addition to traditional data warehouses bigquery offers machine learning features this means you can use bigquery to directly build ml models in five key phases



**in phase one** you extract transform and load data into bigquery if it isn't there already

**in phase two** you select and pre-process features you can use sql to create the training data set for the model to learn from

**in phase 3** you create the ml model inside bigquery

**in phase 4** after your model is trained you can execute an ml.evaluate query to evaluate the performance of the trained model on your evaluation data set

and **in phase 5** the final phase when you're happy with your model performance you can use it to make predictions

**Quiz**

1. BigQuery is a fully managed data warehouse. What does “fully managed” refer to?

BigQuery manages the cost for you.

**BigQuery manages the underlying structure for you.**

BigQuery manages the data quality for you.

BigQuery manages the data source for you.

2.Which pattern describes source data that is moved into a BigQuery table in a single operation?

Spot load

Streaming

**Batch load**

Generated data

3. Which two services does BigQuery provide?

Application services and storage

Application services and analytics

Storage and compute

**Storage and analytics**

4.You want to use machine learning to identify whether an email is spam. Which should you use?

**Supervised learning, logistic regression**

Supervised learning, linear regression

Unsupervised learning, cluster analysis

Unsupervised learning, dimensionality reduction

5.In a supervised machine learning model, what provides historical data that can be used to predict future data?

Data points

Examples

**Labels**

Features

6. Data has been loaded into BigQuery, and the features have been selected and preprocessed. What should happen next when you use BigQuery ML to develop a machine learning model?

Evaluate the performance of the trained ML model.

**Create the ML model inside BigQuery.**

Use the ML model to make predictions.

Classify labels to train on historical data.

7.You want to use machine learning to group random photos into similar groups. Which should you use?

Supervised learning, logistic regression

Supervised learning, linear regression

**Unsupervised learning, cluster analysis**

Unsupervised learning, dimensionality reduction

8.Which BigQuery feature leverages (apalanca, aprovecha) geography data types and standard SQL geography functions to analyze a data set?

Building machine learning models

**Geospatial analysis**

Ad hoc analysis

Building business intelligence dashboards