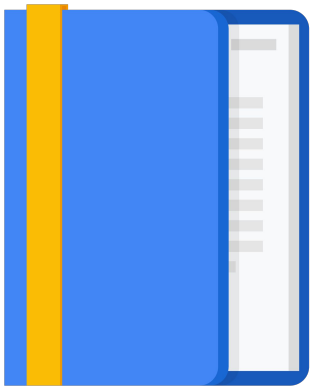
Google Cloud

Custom Model Building with SQL in BigQuery ML

Agenda

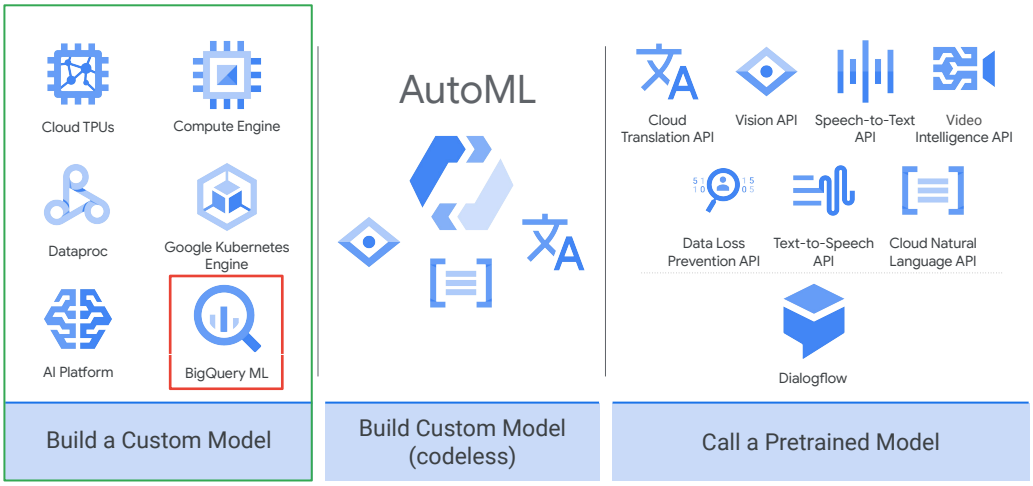
BigQuery ML for Quick Model Building

Supported Models



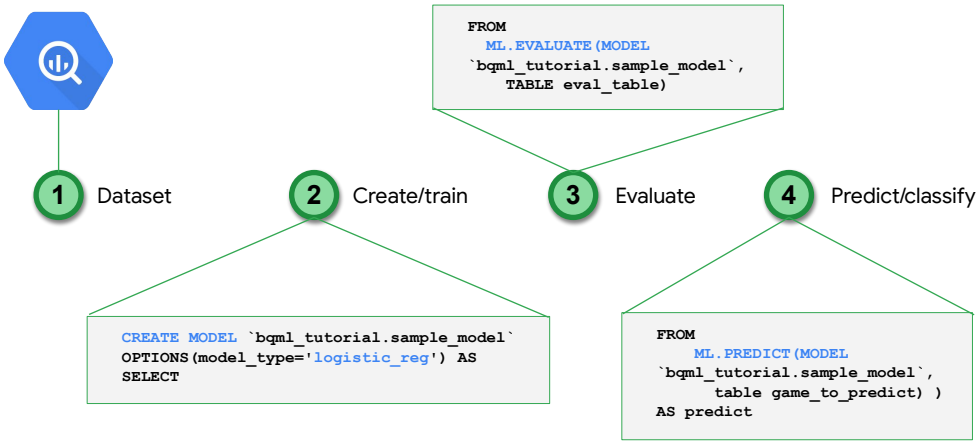
Google Cloud

BigQuery ML is a way to build custom models



Google Cloud

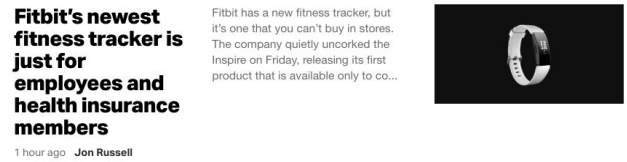
Working with BigQuery ML



Google Cloud

Where was this article published?

1. Techcrunch
2. GitHub
3. NY Times



Google Cloud

SQL query to extract data

```
SELECT
  url, title
FROM
  `bigquery-public-data.hacker_news.stories`
WHERE
  LENGTH(title) > 10
  AND LENGTH(url) > 0
LIMIT 10
```

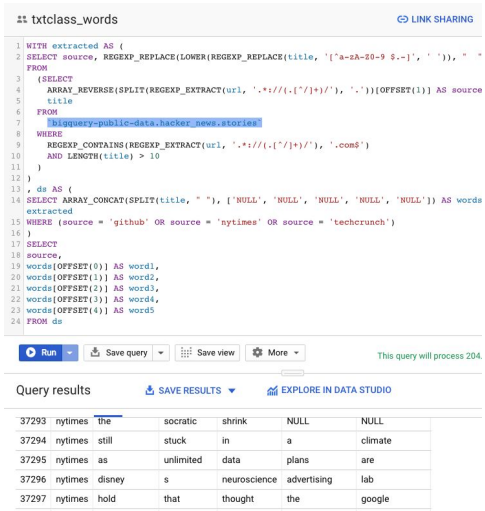
no clusters, no indexes, ad hoc query!

url	title
http://www.bbc.co.uk/news/business-27732743	Vodafone reveals direct government wiretaps
https://www.kickstarter.com/projects/appdocu/a...	Doc – App: The Human Story
http://www.starwebworld.com/android-jelly-bean...	Android Jelly Bean: Streaming Audio Through th...
http://www.myplanetdigital.com/digital_strateg...	Why Canadian Tech Entrepreneurs Need to Man/Wo...
http://startupislandconference.com/index.html	StartupConference June 13. - 16. 2013, HVAR Cr...
http://kopimism.org/	Kopimism Hactivism Meetup Tomorrow (Sunday) in...
http://unearthedgadget.com/xbox-live-gold-2/14...	Xbox Live Gold Membership Is It Really Worth -...
https://evertale.com	Evertale changes the way people remember
http://www.racketboy.com/retro/commodore-amiga...	Commodore Amiga: A Beginner's Guide
http://www.extremetech.com/extreme/156393-cold...	Cold fusion reactor "independently verified"

Google Cloud


```
SELECT
  url, title
FROM
  `bigquery-public-data.hacker_news.stories`
WHERE
  LENGTH(title) > 10
  AND LENGTH(url) > 0
LIMIT 10
```

Use regex to get source + train on words of title



Google Cloud

```
WITH extracted AS (
SELECT source, REGEXP_REPLACE(LOWER(REGEXP_REPLACE(title,
'[^a-zA-Z0-9 $.-]', ' ')), "  ", " ") AS title FROM
  (SELECT
    ARRAY_REVERSE(SPLIT(REGEXP_EXTRACT(url, '.*://(.[^/]+)/'), '.'))[OFFSET(1)]
AS source,
    title
  FROM
    `bigquery-public-data.hacker_news.stories`
  WHERE
    REGEXP_CONTAINS(REGEXP_EXTRACT(url, '.*://(.[^/]+)/'), '.com$')
    AND LENGTH(title) > 10
  )
)
, ds AS (
SELECT ARRAY_CONCAT(SPLIT(title, " "), ['NULL', 'NULL', 'NULL', 'NULL', 'NULL'])
AS words, source FROM extracted
WHERE (source = 'github' OR source = 'nytimes' OR source = 'techcrunch')
)
SELECT
source,
words[OFFSET(0)] AS word1,
words[OFFSET(1)] AS word2,
words[OFFSET(2)] AS word3,
```

```
words[OFFSET(3)] AS word4,
words[OFFSET(4)] AS word5
FROM ds
```

Create model

Query to extract training data

```
CREATE OR REPLACE MODEL advdata.txtclass
OPTIONS(model_type='logistic_reg',
input_label_cols=['source'])
AS


WITH extracted AS (
...
)
, ds AS (
SELECT ARRAY_CONCAT(SPLIT(title, " "), ['NULL', 'NULL',
'NULL', 'NULL', 'NULL']) AS words, source FROM extracted
WHERE (source = 'github' OR source = 'nytimes' OR source
= 'techcrunch')
)


SELECT
source,
words[OFFSET(0)] AS word1,
words[OFFSET(1)] AS word2,
words[OFFSET(2)] AS word3,
words[OFFSET(3)] AS word4,
words[OFFSET(4)] AS word5
FROM ds
```

Google Cloud

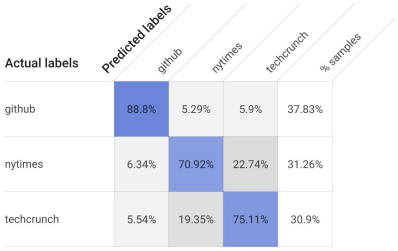
A model feels like just another table that is being created.
https://towardsdatascience.com/choosing-between-tensorflow-keras-bigquery-ml-and-automl-natural-language-for-text-classification-6b1c9fc21013

Evaluate model

```
SELECT * FROM ML.EVALUATE(MODEL advdata.txtclass)
```

precision	recall	accuracy	f1_score	log_loss	roc_auc
0.783	0.783	0.79	0.783	0.858	0.918

(BQML splits the training data and reports evaluation statistics on the held-out set)



Google Cloud

```
SELECT * FROM ML.EVALUATE(MODEL advdata.txtclass)
```

Predict using trained model

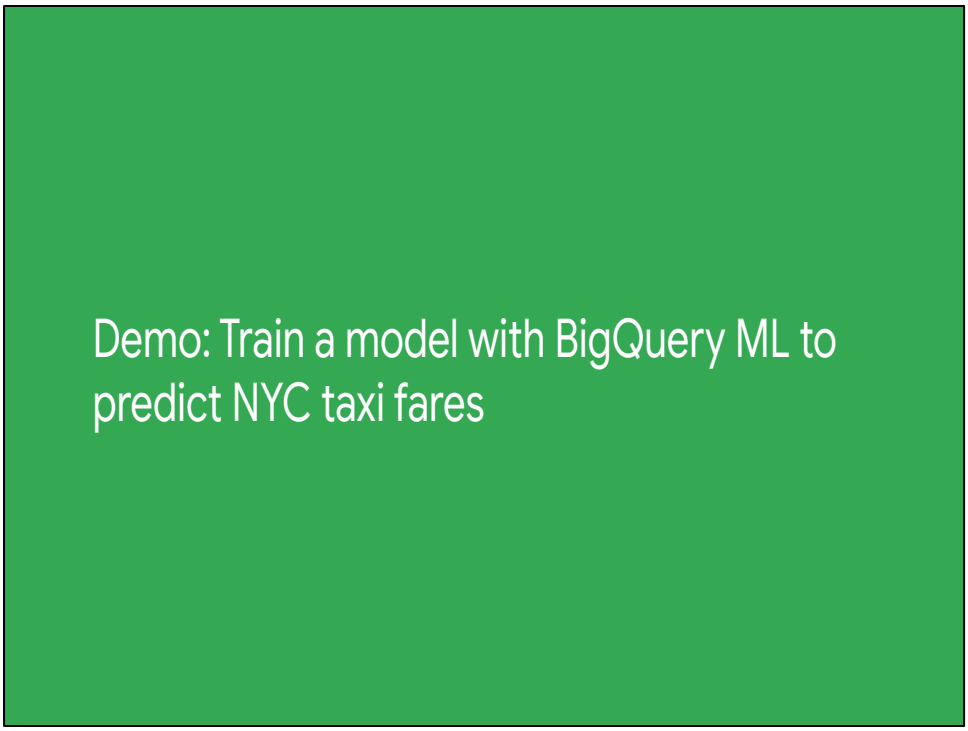
```
SELECT * FROM ML.PREDICT(MODEL advdata.txtclass,(
     SELECT 'government' AS word1, 'shutdown' AS word2, 'leaves'
AS word3, 'workers' AS word4, 'reeling' AS word5
     UNION ALL SELECT 'unlikely', 'partnership', 'in', 'house',
'gives'
     UNION ALL SELECT 'fitbit', 's', 'fitness', 'tracker', 'is'
     UNION ALL SELECT 'downloading', 'the', 'android', 'studio',
'project'
))
```

"Batch prediction"

Row	predicted_source	word1	word2	word3	word4	word5
1	nytimes	government	shutdown	leaves	workers	reeling
2	nytimes	unlikely	partnership	in	house	gives
3	techcrunch	fitbit	s	fitness	tracker	is
4	techcrunch	downloading	the	android	studio	project

Google Cloud


```
SELECT * FROM ML.PREDICT(MODEL advdata.txtclass,(
     SELECT 'government' AS word1, 'shutdown' AS word2, 'leaves' AS word3,
'workers' AS word4, 'reeling' AS word5
     UNION ALL SELECT 'unlikely', 'partnership', 'in', 'house', 'gives'
     UNION ALL SELECT 'fitbit', 's', 'fitness', 'tracker', 'is'
     UNION ALL SELECT 'downloading', 'the', 'android', 'studio', 'project'
))
```

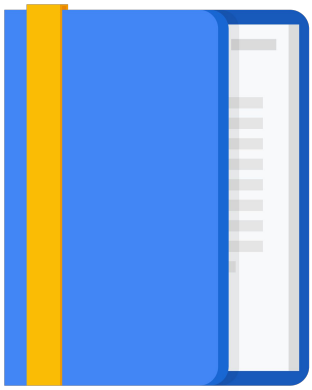


Demo Instructions:
https://github.com/GoogleCloudPlatform/training-data-analyst/blob/master/courses/data-engineering/demos/predict_taxi_bigqueryml.md

Agenda

BigQuery ML for Quick Model Building

Supported Models



Google Cloud

Linear Classifier (Logistic regression)

```
#standardsql
CREATE OR REPLACE MODEL flights.ontime
OPTIONS
  (model_type='logistic_reg', input_label_cols=['on_time']) AS
SELECT
  IF(arr_delay < 15, 1, 0) AS on_time,
  carrier,
  origin,
  dest,
  dep_delay,
  taxi_out,
  distance
FROM
  `cloud-training-demos.flights.tzcorr`
WHERE
  arr_delay IS NOT NULL
```

Google Cloud

Show them running this model (If so, be sure and create a flights dataset first in US)

QUERY:

```
#standardsql
CREATE OR REPLACE MODEL flights.ontime
OPTIONS
  (model_type='logistic_reg', input_label_cols=['on_time']) AS
SELECT
  IF(arr_delay < 15, 1, 0) AS on_time,
  carrier,
  origin,
  dest,
  dep_delay,
  taxi_out,
  distance
FROM
  `cloud-training-demos.flights.tzcorr`
WHERE
  arr_delay IS NOT NULL
```

DNN Classifier (alpha)

```
#standardsql
CREATE OR REPLACE MODEL flights.ontime
OPTIONS
  (model_type='dnn_classifier', hidden_units = [47,29,18],
  input_label_cols=['on_time']) AS
SELECT
  IF(arr_delay < 15, 1, 0) AS on_time,
  carrier,
  origin,
  dest,
  dep_delay,
  taxi_out,
  distance
FROM
  `cloud-training-demos.flights.tzcorr`
WHERE
  arr_delay IS NOT NULL
```

Show them running this model

QUERY:

```
#standardsql
CREATE OR REPLACE MODEL flights.ontime
OPTIONS
  (model_type='dnn_classifier', hidden_units = [47,29,18],
  input_label_cols=['on_time']) AS
SELECT
  IF(arr_delay < 15, 1, 0) AS on_time,
  carrier,
  origin,
  dest,
  dep_delay,
  taxi_out,
  distance
FROM
  `cloud-training-demos.flights.tzcorr`
WHERE
  arr_delay IS NOT NULL
```

xgboost Classifier (alpha)

```
#standardsql
CREATE OR REPLACE MODEL flights.ontime
OPTIONS
  (model_type='boosted_tree_classifier', input_label_cols=['on_time']) AS
SELECT
  IF(arr_delay < 15, 1, 0) AS on_time,
  carrier,
  origin,
  dest,
  dep_delay,
  taxi_out,
  distance
FROM
  `cloud-training-demos.flights.tzcorr`
WHERE
  arr_delay IS NOT NULL
```

Google Cloud

Show them running this model

QUERY:

```
#standardsql
CREATE OR REPLACE MODEL flights.ontime
OPTIONS
  (model_type='boosted_tree_classifier'', input_label_cols=['on_time']) AS
SELECT
  IF(arr_delay < 15, 1, 0) AS on_time,
  carrier,
  origin,
  dest,
  dep_delay,
  taxi_out,
  distance
FROM
  `cloud-training-demos.flights.tzcorr`
WHERE
  arr_delay IS NOT NULL
```

Linear Regression

```
CREATE OR REPLACE MODEL
  taxi.taxifare_dnn OPTIONS (model_type='linear_reg',
    labels=['fare_amount']) AS
SELECT
    fare_amount,
    hourofday, dayofweek,
    pickuplon, pickuplat, dropofflon, dropofflat,
    passenger_count
  FROM
    `taxi.taxi3m`
```

Google Cloud

```
CREATE OR REPLACE MODEL
  taxi.taxifare_dnn OPTIONS (model_type='linear_regressor',
    labels=['fare_amount']) AS
SELECT
    fare_amount,
    hourofday, dayofweek,
    pickuplon, pickuplat, dropofflon, dropofflat,
    passenger_count
  FROM
    `taxi.taxi3m`
```

DNN Regression (alpha)

```
CREATE OR REPLACE MODEL
  taxi.taxifare_dnn OPTIONS (model_type='dnn_regressor',
    hidden_units=[144,89,55],
    labels=['fare_amount']) AS
SELECT
    fare_amount,
    hourofday, dayofweek,
    pickuplon, pickuplat, dropofflon, dropofflat,
    passenger_count
  FROM
    `taxi.taxi3m`
```

Google Cloud


```
CREATE OR REPLACE MODEL
  taxi.taxifare_dnn OPTIONS (model_type='dnn_regressor',
    hidden_units=[144,89,55],
    labels=['fare_amount']) AS
SELECT
    fare_amount,
    hourofday, dayofweek,
    pickuplon, pickuplat, dropofflon, dropofflat,
    passenger_count
  FROM
    `taxi.taxi3m`
```

xgboost Regression (alpha)

```
CREATE OR REPLACE MODEL
  taxi.taxifare_xgboost
  OPTIONS (model_type='boosted_tree_regressor',
    labels=['fare_amount']) AS
SELECT
    fare_amount,
    hourofday, dayofweek,
    pickuplon, pickuplat, dropofflon, dropofflat,
    passenger_count
  FROM
    `taxi.taxi3m`
```

Google Cloud


```
CREATE OR REPLACE MODEL
  taxi.taxifare_xgboost
  OPTIONS (model_type='boosted_tree_regressor',
    labels=['fare_amount']) AS
SELECT
    fare_amount,
    hourofday, dayofweek,
    pickuplon, pickuplat, dropofflon, dropofflat,
    passenger_count
  FROM
    `taxi.taxi3m`
```

Train on TF, predict with BigQuery

```
CREATE OR REPLACE MODEL advdata.txtclass_tf2
OPTIONS (model_type='tensorflow',

model_path='gs://cloud-training-demos-ml/txtcls/trained_finetune_native
/export/exporter/1549825580/*')
```

```
SELECT
  input,
  (SELECT AS STRUCT(p, ['github', 'nytimes', 'techcrunch'][ORDINAL(s)])
prediction FROM
    (SELECT p, ROW_NUMBER() OVER() AS s FROM
      (SELECT * FROM UNNEST(dense_1) AS p))
  ORDER BY p DESC LIMIT 1).*

FROM ML.PREDICT(MODEL advdata.txtclass_tf2,
(
SELECT 'Unlikely Partnership in House Gives Lawmakers Hope for Border
Deal' AS input
UNION ALL SELECT "Fitbit\'s newest fitness tracker is just for
employees and health insurance members"
UNION ALL SELECT "Show HN: Hello, a CLI tool for managing social media"
))
```

Recommendation engine (matrix factorization alpha)

```
create or replace model models.suggested_products_1or2_example
options(model_type='matrix_factorization',
        user_col='user_id', item_col='product_id', rating_col='rating',
        l2_reg=10)
AS

with purchases AS (
 select product_id, user_id from
 operations.orders_with_lines, unnest(order_lines)
),

total_purchases as (
select product_id, user_id, count(*) as numtimes
from purchases
group by product_id, user_id
)

select
product_id, user_id,
IF(numtimes < 2, 1, 2) AS rating
FROM total_purchases
```

Google Cloud

Show them running this model

So what do we recommend for a given set of users?

```
with users AS (
SELECT
user_id, count(*) as num_orders
from operations.orders_with_lines
group by user_id
order by num_orders desc
limit 10
),

products as (
select product_id, count(*) as num_orders
from operations.orders_with_lines, unnest(order_lines)
group by product_id
order by num_orders desc
limit 10
)

SELECT * FROM ML.PREDICT(MODEL models.suggested_products_1or2,
(SELECT user_id, product_id
FROM users, products)
)
```

Google Cloud

I'll assume that we have s

So what do we recommend for a given set of users?

Row	predicted_rating	user_id	product_id
1	1.5746015507788755	101797	26209
2	1.8070705987455633	101797	13176
3	1.7171094544245578	101797	27845
4	1.9763373899260837	101797	47209
5	1.8659380090171271	101797	21137
6	1.721610848530093	101797	47766
7	1.9516130703939483	101797	21903

Google Cloud

I'll assume that we have s

Clustering

```
CREATE OR REPLACE MODEL
demos_eu.london_station_clusters
OPTIONS(model_type='kmeans', num_clusters=4,
standardize_features = true) AS

WITH hs AS ...,
stationstats AS ...

SELECT * except(station_name, isweekday)
from stationstats
```

1. 4 clusters (hardcoded)
2. Standardize features since different dynamic ranges
3. Remove the cluster "id" fields (keep just the attributes)

Google Cloud

Which cluster?

```
WITH hs AS ...,
stationstats AS ...,

SELECT * except(nearest_centroids_distance)
FROM ML.PREDICT(MODEL
demos_eu.london_station_clusters,
(SELECT * FROM stationstats WHERE
REGEXP_CONTAINS(station_name, 'Kennington')))
```

Row	CENTROID_ID	station_name	isweekday	duration	num_trips	bikes_count	distance_from_city_center
1	3	Kennington Lane Tesco, Vauxhall	weekday	911.5810637908974	5471	9	1.8345619962343163
2	3	Kennington Lane Rail Bridge, Vauxhall	weekday	979.3919952622995	20263	19	2.175032834765301
3	4	Doddington Grove, Kennington	weekday	1397.7189755200225	7067	28	1.468140527379382
4	4	Kennington Cross, Kennington	weekday	911.5238777770538	15349	35	1.4625875338501981

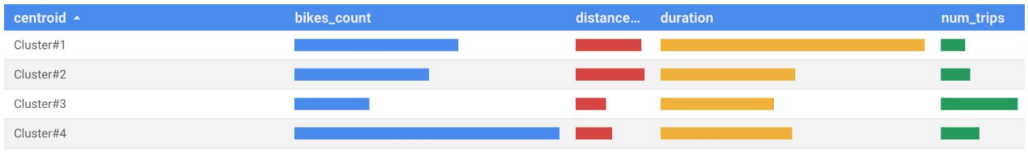
Google Cloud

Find cluster attributes

```
WITH T AS (
SELECT
centroid_id,
ARRAY_AGG(STRUCT(numerical_feature AS name, ROUND(feature_value,1)
AS value) ORDER BY centroid_id) AS cluster
FROM ML.CENTROIDS(MODEL demos_eu.london_station_clusters)
GROUP BY centroid_id
)
SELECT
CONCAT('Cluster#', CAST(centroid_id AS STRING)) AS centroid,
(SELECT value from unnest(cluster) WHERE name = 'duration') AS
duration,
(SELECT value from unnest(cluster) WHERE name = 'num_trips') AS
num_trips,
(SELECT value from unnest(cluster) WHERE name = 'bikes_count') AS
bikes_count,
(SELECT value from unnest(cluster) WHERE name =
'distance_from_city_center') AS distance_from_city_center
FROM T
ORDER BY centroid_id ASC
```

Visualize attributes in Data Studio ...

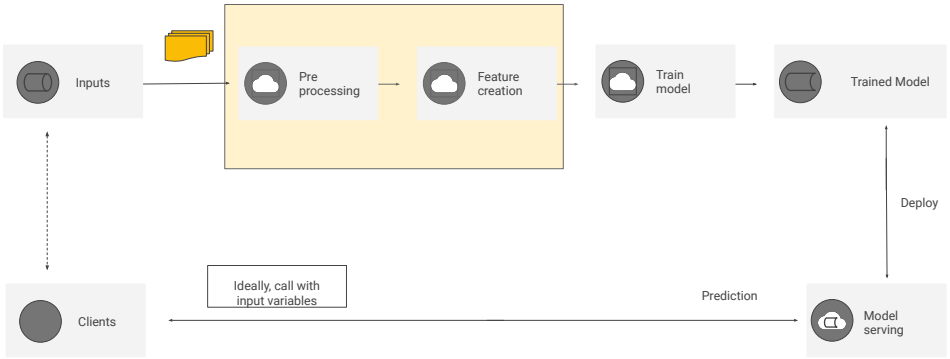
Row	centroid	duration	num_trips	bikes_count	distance_from_city_center
1	Cluster#1	3079.5	3026.1	14.0	6.2
2	Cluster#2	1564.0	3635.1	11.5	6.5
3	Cluster#3	1319.6	9654.8	6.4	2.9
4	Cluster#4	1527.7	4846.8	22.6	3.5



Cluster attributes visualized in Data Studio

Google Cloud

Use the transform clause



Google Cloud

Notes:

Not as obvious … who will do the input transformations on behalf of the client code? You can't pass in the raw input variables to the trained model -- it expects scaled, transformed inputs!

You also have to worry about model changes -- when you do a bag-of-words, for example, with IBM=32, the embedding might change in the next model run because your input data is larger. Similarly, in scaling, min/max/stdev can all change. Doing the bookkeeping associated with preprocessing and feature crosses is painful and a major source of error. It is also near-impossible to find, so there are probably many ML models out there that have a "**training/serving skew**" (yes, this is a real thing, with a real jargon word for it, but it is rarely discussed because the majority of ML research papers are from college settings where routine model updates are not a concern.)

TRANSFORM ensures transformations are automatically applied during ML.PREDICT

```
CREATE OR REPLACE MODEL ch09edu.bicycle_model
OPTIONS(input_label_cols=['duration'],
        model_type='linear_reg')
AS

SELECT
  duration
  , start_station_name
  , CAST(EXTRACT(dayofweek from start_date) AS STRING)
        as dayofweek
  , CAST(EXTRACT(hour from start_date) AS STRING)
        as hourofday
FROM
  `bigquery-public-data.london_bicycles.cycle_hire`
```

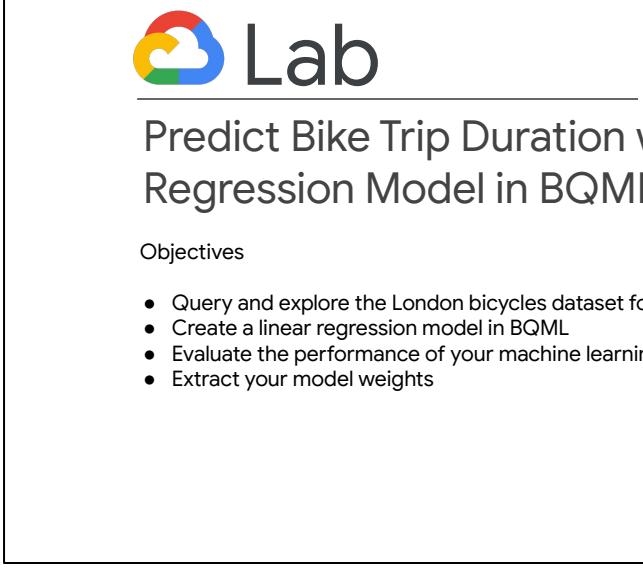
```
SELECT * FROM ML.PREDICT(MODEL ch09edu.bicycle_model,(
  350 AS duration
  , 'Kings Cross' AS start_station_name
  , '3' as dayofweek
  , '18' as hourofday
))
```

```
CREATE OR REPLACE MODEL ch09edu.bicycle_model
OPTIONS(input_label_cols=['duration'],
        model_type='linear_reg')
TRANSFORM(
  SELECT * EXCEPT(start_date)
  , CAST(EXTRACT(dayofweek from start_date) AS STRING)
        as dayofweek
  , CAST(EXTRACT(hour from start_date) AS STRING)
        as hourofday
)
AS
SELECT
  duration, start_station_name, start_date
FROM
  `bigquery-public-data.london_bicycles.cycle_hire`
```

```
SELECT * FROM ML.PREDICT(MODEL ch09edu.bicycle_model,(
  350 AS duration
  , 'Kings Cross' AS start_station_name
  , CURRENT_TIMESTAMP() as start_date
))
```

Reminder: 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
  ```
  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>))`



Predict Bike Trip Duration with a Regression Model in BQML
https://gcpstaging.qwiklabs.com/labs/25421/edit

Movie Recommendations in BigQuery ML

Objectives

- Create a BigQuery dataset to store and load MovieLens data
- Explore the MovieLens dataset
- Use a trained model to make recommendations in BigQuery
- Make product predictions for both single users and batch users

Movie Recommendations in BigQuery ML
https://gcpstaging.qwiklabs.com/labs/25983/

Module Summary

- You can train and evaluate machine learning models directly in BigQuery