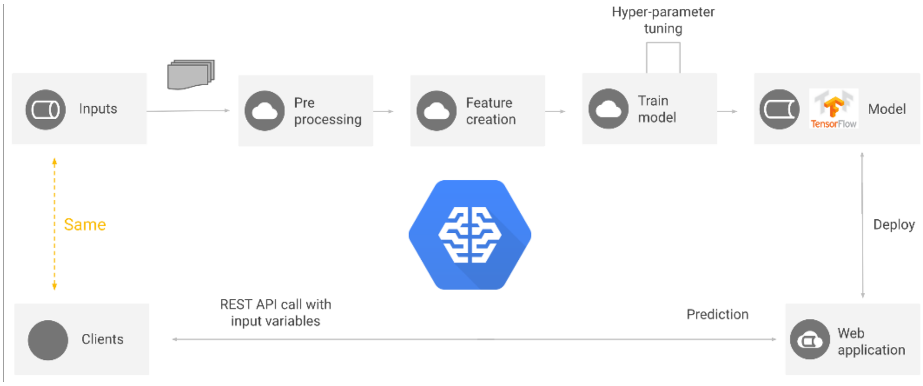
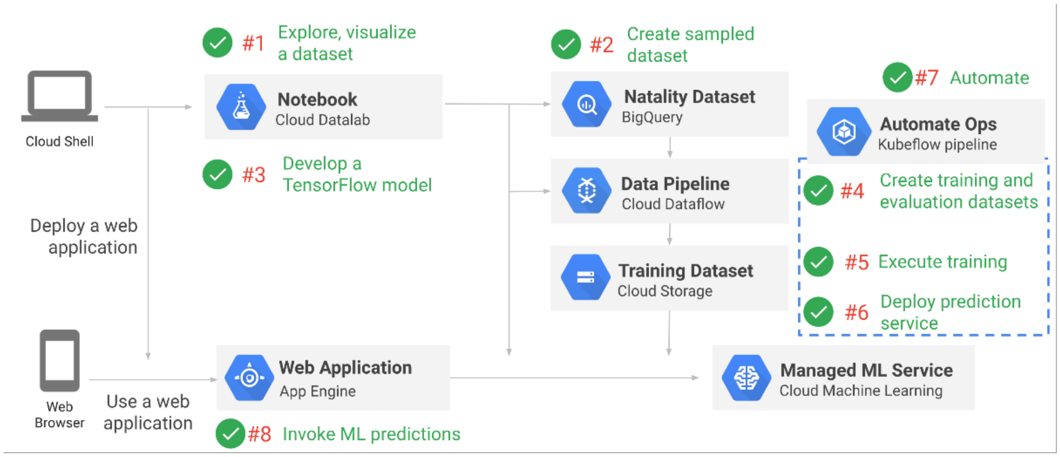
Machine learning class – Vinicius F. Caridá

End-to-End machine learning solution





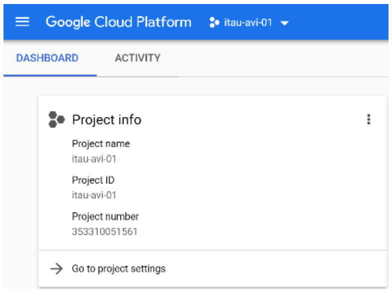
Before start the class, I recommend read and do the TensorFlow Get Started examples

https://www.tensorflow.org/tutorials?hl=pt_BR

0 – INICIAL CONFIGURATION (15 min)

Go to gcp console https://console.cloud.google.com

From the GCP First access: Hamburger menu -> home and get the informations “Project name” and “Project ID”



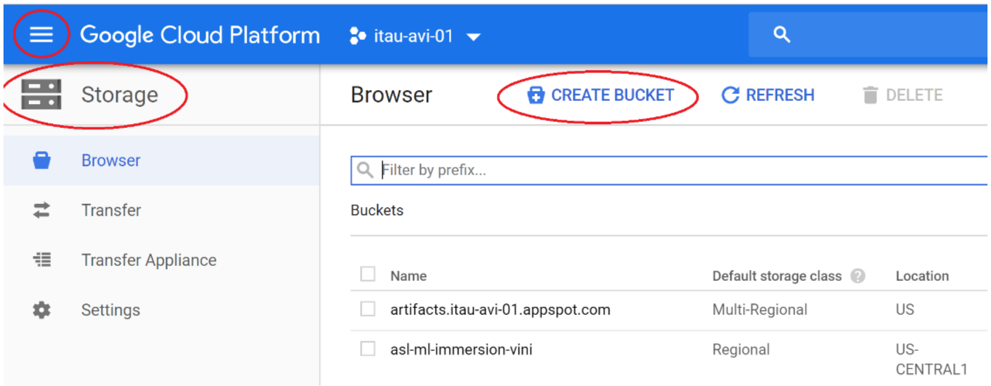
Project info

Project name
itau-avi-01

Project ID
itau-avi-01

Project number
353310051561

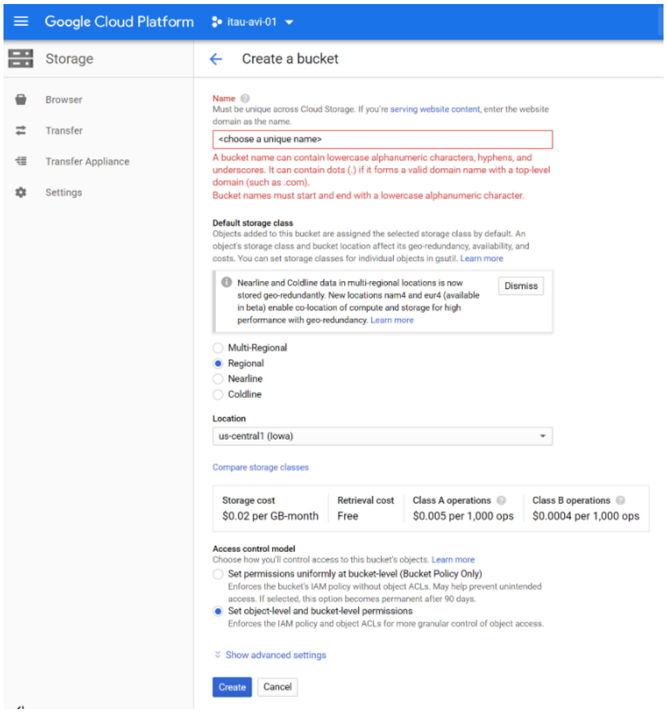
Then access: hamburger menu -> Storage and click on “create bucket”



Buckets

□	Name	Default storage class	Location
□	artifacts.itau-avi-01.appspot.com	Multi-Regional	US
□	asl-ml-immersion-vini	Regional	US-CENTRAL1

Choose a unique name to your bucket, select the “regional” option and choose one location. I select the “US-CENTRAL1”



Create a bucket

Name
Must be unique across Cloud Storage. If you're serving website content, enter the website domain as the name.

<choose a unique name>

A bucket name can contain lowercase alphanumeric characters, hyphens, and underscores. It can contain dots (.) if it forms a valid domain name with a top-level domain (such as .com).
Bucket names must start and end with a lowercase alphanumeric character.

Default storage class
Objects added to this bucket are assigned the selected storage class by default. An object's storage class and bucket location affect its geo-redundancy, availability, and costs. You can set storage classes for individual objects in gsutil. Learn more

Nearline and Coldline data in multi-regional locations is now stored geo-redundantly. New locations nam4 and eur4 (available in beta) enable co-location of compute and storage for high performance with geo-redundancy. Learn more Dismiss

○ Multi-Regional
● Regional
○ Nearline
○ Coldline

Location
us-central1 (Iowa)

Compare storage classes

Storage cost	Retrieval cost	Class A operations	Class B operations
$0.02 per GB-month	Free	$0.005 per 1,000 ops	$0.0004 per 1,000 ops

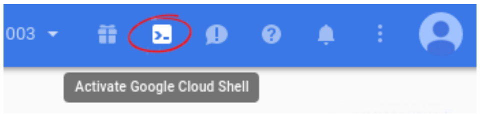
Access control model
Choose how you'll control access to this bucket's objects. Learn more

○ Set permissions uniformly at bucket-level (Bucket Policy Only)
Enforces the bucket's IAM policy without object ACLs. May help prevent unintended access. If selected, this option becomes permanent after 90 days.

● Set object-level and bucket-level permissions
Enforces the IAM policy and object ACLs for more granular control of object access.

Show advanced settings

From the GCP Console click the Cloud Shell icon on the top right toolbar:



Then click "Start Cloud Shell":

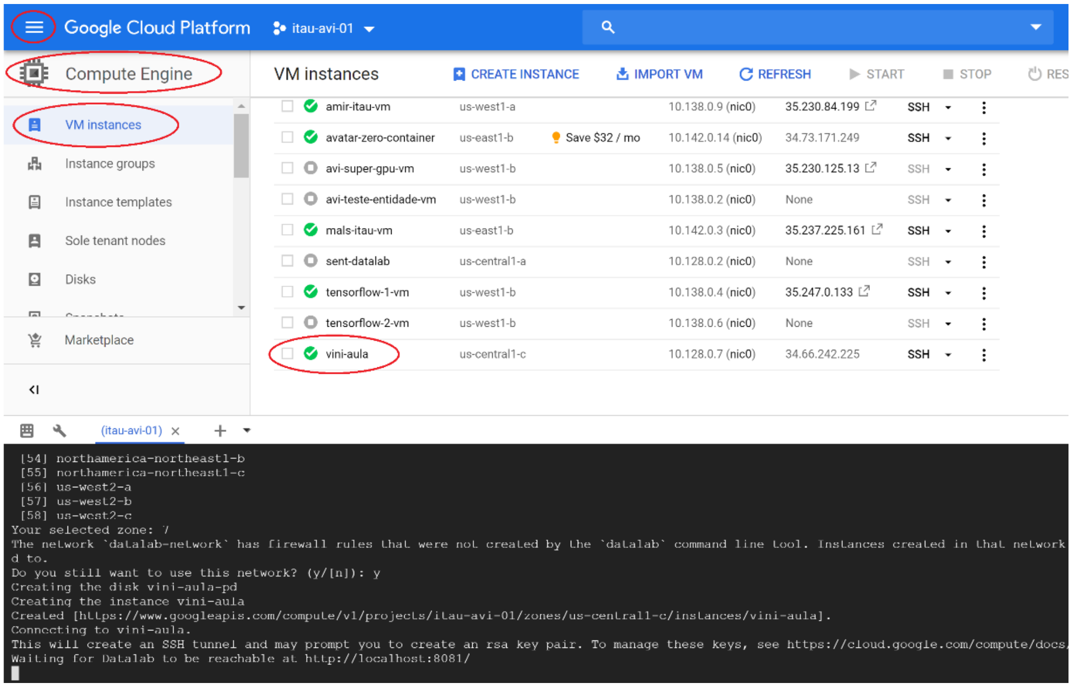
In Cloud Shell, type:

```
datalab create datalab-instance-name
```

Select zone <I selected the zone 7 because is the same zone that my bucket>

When the process are complete, access:

Hamburger menu -> Compute Engine -> VM instances and chek if your machine are ready

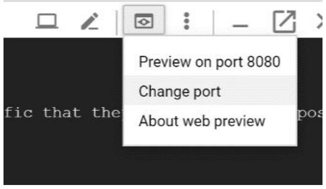


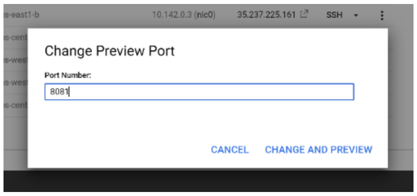
□	✓	amir-itau-vm	us-west1-a		10.138.0.9 (nic0)	35.230.84.199	SSH
□	✓	avatar-zero-container	us-east1-b	Save $32 / mo	10.142.0.14 (nic0)	34.73.171.249	SSH
□	○	avi-super-gpu-vm	us-west1-b		10.138.0.5 (nic0)	35.230.125.13	SSH
□	○	avi-teste-entidade-vm	us-west1-b		10.138.0.2 (nic0)	None	SSH
□	✓	mals-itau-vm	us-east1-b		10.142.0.3 (nic0)	35.237.225.161	SSH
□	○	sent-datalab	us-central1-a		10.128.0.2 (nic0)	None	SSH
□	✓	tensorflow-1-vm	us-west1-b		10.138.0.4 (nic0)	35.247.0.133	SSH
□	○	tensorflow-2-vm	us-west1-b		10.138.0.6 (nic0)	None	SSH
□	✓	vini-aula	us-central1-c		10.128.0.7 (nic0)	34.66.242.225	SSH

```
 [54] northamerica-northeast1-b
 [55] northamerica-northeast1-c
 [56] us-west2-a
 [57] us-west2-b
 [58] us-west2-c
Your selected zone: 7
The network `datalab-network` has firewall rules that were not created by the `datalab` command line tool. Instances created in that network
d to.
Do you still want to use this network? (y/[n]): y
Creating the disk vini-aula-pd
Creating the instance vini-aula
Created [https://www.googleapis.com/compute/v1/projects/itau-avi-01/zones/us-central1-c/instances/vini-aula].
Connecting to vini-aula.
This will create an SSH tunnel and may prompt you to create an rsa key pair. To manage these keys, see https://cloud.google.com/compute/docs
Waiting for Datalab to be reachable at http://localhost:8081/
```

PS: If in any moment you lost the connection with the server (you closed the cloud shell, close the browser, etc.), open cloud shell again and use the command: `datalab connect` ***datalab-instance-name***

Open the: web preview -> change port -> 8081 -> change and preview





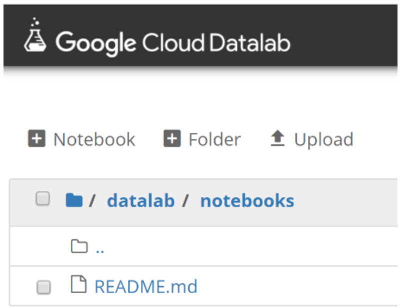
Change Preview Port

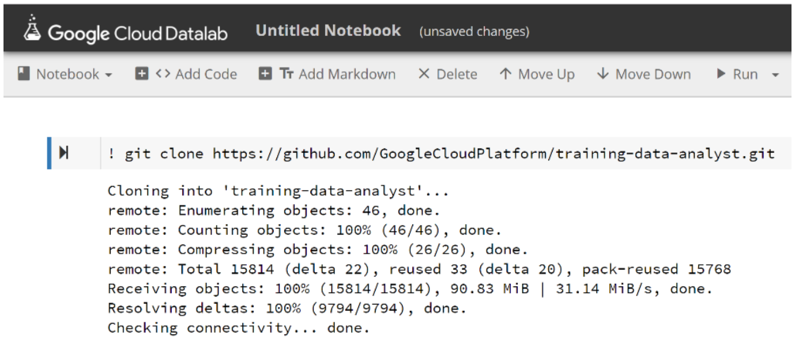
Port Number: 8081

Navigate to: datalab -> notebooks -> Click on: +Notebook

Insert and run the line below to clone a repository

! git clone https://github.com/GoogleCloudPlatform/training-data-analyst.git





```
! git clone https://github.com/GoogleCloudPlatform/training-data-analyst.git
```

```
Cloning into 'training-data-analyst'...
remote: Enumerating objects: 46, done.
remote: Counting objects: 100% (46/46), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 15814 (delta 22), reused 33 (delta 20), pack-reused 15768
Receiving objects: 100% (15814/15814), 90.83 MiB | 31.14 MiB/s, done.
Resolving deltas: 100% (9794/9794), done.
Checking connectivity... done.
```

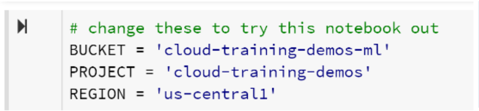
Navigate to: datalab -> notebooks -> Click on: +Folder

In this new folder, do the upload of all files of zip file “tensor_keras_pytorch.zip”

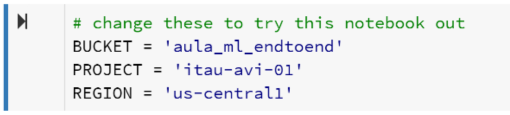
1- RUN SOME EXPERIMENTS TO START THE TRAINING MODEL (15 min)

In Cloud Shell (**NOT Datalab**), run the code: gsutil -m cp -r gs://cloud-training-demos/babyweight/preproc gs://**your-bucket**/

Collect the data of your project. Change this information in all python notebooks



```
# change these to try this notebook out
BUCKET = 'cloud-training-demos-ml'
PROJECT = 'cloud-training-demos'
REGION = 'us-central1'
```

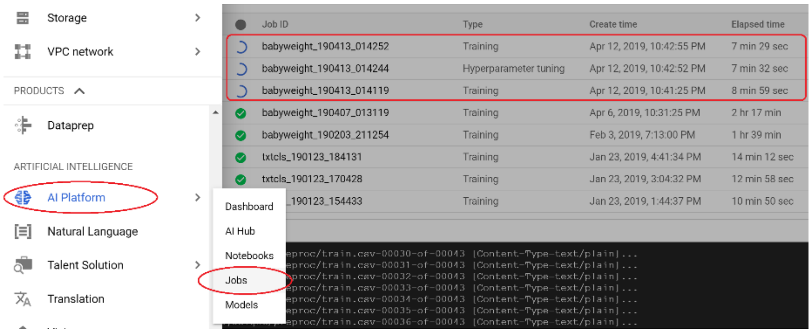


```
# change these to try this notebook out
BUCKET = 'aula_ml_endtoend'
PROJECT = 'itau-avi-01'
REGION = 'us-central1'
```

navigate to **notebooks/training-data-analyst/courses/machine_learning/deepdive/06_structured**

The objective here is not understand the code. Please, read very quickly each cell and rull all. open 5_train.ipynb -> Click on **Clear -> All Cells.** Execute each cell in turn.

access: Hamburger menu -> AI Platform -> Jobs and confirm if the jobs was start



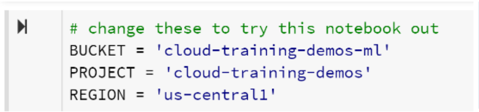
Job ID	Type	Create time	Elapsed time
babyweight_190413_014252	Training	Apr 12, 2019, 10:42:55 PM	7 min 29 sec
babyweight_190413_014244	Hyperparameter tuning	Apr 12, 2019, 10:42:52 PM	7 min 32 sec
babyweight_190413_014119	Training	Apr 12, 2019, 10:41:25 PM	8 min 59 sec
babyweight_190407_013119	Training	Apr 6, 2019, 10:31:25 PM	2 hr 17 min
babyweight_190203_211254	Training	Feb 3, 2019, 7:13:00 PM	1 hr 39 min
txtcls_190123_184131	Training	Jan 23, 2019, 4:41:34 PM	14 min 12 sec
txtcls_190123_170428	Training	Jan 23, 2019, 3:04:32 PM	12 min 58 sec
_190123_154433	Training	Jan 23, 2019, 1:44:37 PM	10 min 50 sec

```
eproc/train.csv-00030-of-00043 [Content-Type-text/plain]...
eproc/train.csv-00031-of-00043 [Content-Type-text/plain]...
eproc/train.csv-00032-of-00043 [Content-Type-text/plain]...
eproc/train.csv-00033-of-00043 [Content-Type-text/plain]...
eproc/train.csv-00034-of-00043 [Content-Type-text/plain]...
eproc/train.csv-00035-of-00043 [Content-Type-text/plain]...
eproc/train.csv-00036-of-00043 [Content-Type-text/plain]...
```

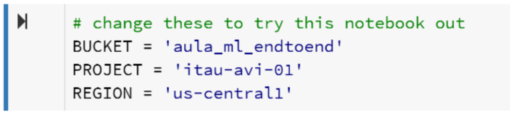
2 – EXPLORER (20 min)

In Cloud Datalab, navigate to **notebooks/training-data-analyst/courses/machine_learning/deepdive/06_structured/labs** and open **1_explore.ipynb**.

In Datalab, change the variables in the first code cell for yours informations



```
# change these to try this notebook out
BUCKET = 'cloud-training-demos-ml'
PROJECT = 'cloud-training-demos'
REGION = 'us-central1'
```



```
# change these to try this notebook out
BUCKET = 'aula_ml_endtoend'
PROJECT = 'itau-avi-01'
REGION = 'us-central1'
```

Click on **Clear -> All Cells**. Now read the narrative and execute each cell in turn.

More details to natality dataset, looking: https://bigquery.cloud.google.com/table/publicdata:samples.natality

To do: Try to do lab task #1, lab task #2 and other analysis/plots to explore and understand the dataset

Write a short conclusion of your data analysis

3 – SAMPLE (20 min.)

In Cloud Datalab, click on the Home icon, and then navigate to **notebooks/training-data-analyst/courses/machine_learning/deepdive/06_structured/labs** and open **2_sample.ipynb**.

Again, change the variables in the first code cell for yours informations. Click on **Clear -> All Cells**. Now read the narrative and execute each cell in turn.

To do: Try to do lab task #1, lab task #2, lab task #3 and other preprocessing in the dataset

Write a short conclusion of your view of the preprocessing

4 – TENSORFLOW (30 min.)

In Cloud Datalab, click on the Home icon, and then navigate to **notebooks/training-data-analyst/courses/machine_learning/deepdive/06_structured/labs** and open **3_tensorflow.ipynb**.

Again, change the variables in the first code cell for yours informations. Click on **Clear -> All Cells**. Now read the narrative and execute each cell in turn.

To do: Try to do lab task #1, lab task #2, lab task #3, lab task #4, lab task #5

More information of tensorflow:

https://www.youtube.com/watch?v=er8RQZoX3yk&list=PL9Hr9sNUjfsmEu1ZniY0XpHSzl5uihcXZ

https://github.com/Hvass-Labs/TensorFlow-Tutorials

paper: https://arxiv.org/abs/1708.02637

5 – PRE-PROCESSING (15 min.)

In Cloud Datalab, click on the Home icon, and then navigate to **notebooks/training-data-analyst/courses/machine_learning/deepdive/06_structured/labs** and open **4_preproc.ipynb**.

Again, change the variables in the code cell for yours informations. Click on **Clear -> All Cells**. Now read the narrative and execute each cell in turn.

To do: Try to do the ToDos

6 – TRAINING (30 min.)

In Cloud Datalab, click on the Home icon, and then navigate to **notebooks/training-data-analyst/courses/machine_learning/deepdive/06_structured/labs** and open **5_train.ipynb**.

Again, change the variables in the code cell for yours informations. Click on **Clear -> All Cells**. Now read the narrative and execute each cell in turn.

To do: Try to do lab task #1, lab task #2, lab task #3, lab task #4, lab task #5

7 – DEPLOY (20 min.)

In Cloud Datalab, click on the Home icon, and then navigate to **notebooks/training-data-analyst/courses/machine_learning/deepdive/06_structured/labs** and open **6_deploy.ipynb**.

Again, change the variables in the code cell for yours informations. Click on **Clear -> All Cells**. Now read the narrative and execute each cell in turn.

To do: Try to do lab task #1, lab task #2, lab task #3, lab task #4

Exercise – (1 hour)

What is the deference between TensorFlow x Keras x Pytorch

LUNCH – (1 hour)

8 – RUN ALL SOLUTION (2 hours)

In Cloud Datalab, navigate to **notebooks/training-data-analyst/courses/machine_learning/deepdive/06_structured**

In this folder we have all previous exercises with the answers. Now, go to each one python notebook, read the narrative and execute each cell in turn.

+2 hours

Build an AppEngine app to serve ML predictions

Add new cloud shell session

In Cloud Shell (**NOT Datalab**), navigate to the folder containing the starter code for this lab

```
cd ~/training-data-analyst/courses/machine_learning/deepdive/06_structured/lab/serving
```

1. Fix the code

Step 1

Run the `what_to_fix.sh` script to see a list of items you need to add/modify to existing code to run your app:

```
./what_to_fix.sh
```

As a result of this, you will a list of filenames and lines within those files marked with "TODO". These are the lines where you have to add/modify code. For this lab, you will focus on #*TODO* items for `main.py` and `form.html` only.

Step 2

You may use the Cloud Shell code editor to view and edit the contents of these files.

Click on the ✎ icon on the top right of your Cloud Shell window to launch Code Editor

Once launched, navigate to the `~/training-data-analyst/courses/machine_learning/deepdive/06_structured/labs/serving` directory.

Step 3

Open the **`application/main.py`** and **`application/templates/form.html`** files and replace #*TODO*s with code. For hints, see the following section.

2. Hints to modify main.py

Step 1

Set the credentials to use Google Application Default Credentials and specify the ML Engine API with version.

Step 2

Specify the name of your trained model deployed on Cloud MLE using the **parent** variable.

Step 3

Build the call request with the **prediction** variable.

Step 4

Cast the gestation_weeks feature into a float within the **features** array.

3. Hints to modify form.html

Step 1

Add more values for the drop down option for **Plurality**.

4. Code to modify main.py

Step 1

Open the main.py file by clicking on it. Notice the lines with # *TODO* for setting credentials and the api to use.

Set the credentials to use Google Application Default Credentials (recommended way to authorize calls to our APIs when building apps deployed on AppEngine):

```
credentials = GoogleCredentials.get_application_default()
```

Specify the api name (ML Engine API) and version to use:

```
api = discovery.build('ml', 'v1', credentials=credentials)
```

Step 2

Scroll further down in main.py and look for the next #*TODO* in the method get_prediction(). In there, specify, using the **parent** variable, the name of your trained model deployed on Cloud MLE:

```
parent = 'projects/%s/models/%s' % (project, model_name)
```

Step 3

Now that you have all the pieces for making the call to your model, build the call request by specifying it in the **prediction** variable:

```
prediction = api.projects().predict(body=input_data, name=parent).execute()
```

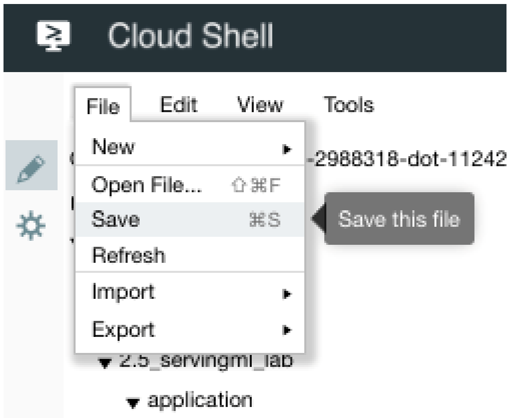
Step 4

The final *#TODO* (scroll towards bottom) is to get gestation_weeks from the form data and cast into a float within the **features** array:

```
features['gestation_weeks'] = float(data['gestation_weeks'])
```

Step 5

Save the changes you made using the **File** > **Save** button on the top left of your code editor window.



5. Code to modify form.html

form.html is the front-end of your app. The user fills in data (features) about the mother based on which we will make the predictions using our trained model.

Step 1

In code editor, navigate to the `application/templates` directory and click to open the `form.html` file

Step 2

There is one #TODO item here. Look for the div segment for **Plurality** and add options for other plurality values (2, 3, etc)

```
<md-option value="2">Twins</md-option>
<md-option value="3">Triplets</md-option>
```

Step 3

Save the changes you made using the **File** > **Save** button on the top left of your code editor window

6. Deploy and test your app

navigate to the /training-data-analyst/courses/machine_learning/deepdive/06_structured/serving/application/

Delete the file main.py and upload the main.py that the professor shared

navigate to the /training-data-analyst/courses/machine_learning/deepdive/06_structured/serving/application/templates/

Delete the file form.html and upload the form.html that the professor shared

Step 1

In Cloud Shell, go to /training-data-analyst/courses/machine_learning/deepdive/06_structured/serving/

run the deploy.sh script to install required dependencies and deploy your app engine app to the cloud.

```
./deploy.sh
```

Step 2

Go to the url `https://<PROJECT-ID>.appspot.com` and start making predictions.

Note: Replace <PROJECT-ID> with your Project ID.

Serving ML Predictions in batch and real-time

Step 1

In Cloud Shell, navigate to the folder containing the starter code for this lab

```
cd ~/training-data-analyst/courses/machine_learning/deepdive/06_structured/labs/serving
```

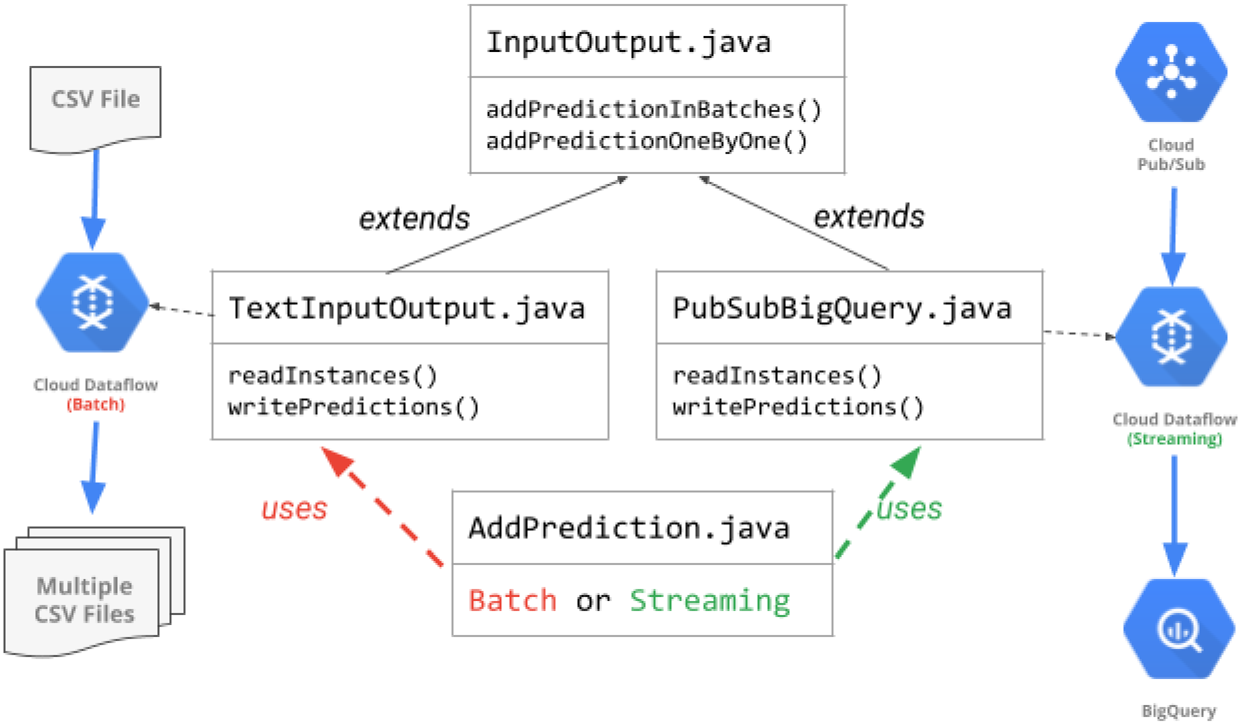
Step 2

Run the `what_to_fix.sh` script to see a list of items you need to add/modify to existing code to run your app:

```
./what_to_fix.sh
```

As a result of this, you will see a list of filenames and lines within those files marked with "TODO". These are the lines where you have to add/modify code. For this lab, you will focus on #TODO items for **.java files only**, namely `BabyweightMLService.java` : which is your prediction service

7. How the code is organized



8. Prediction service

In this section, you fix the code in **BabyweightMLService.java** and test it with the **run_once.sh** script that is provided. If you need help with the code, look at the next section that provides hints on how to fix code in BabyweightMLService.java

Step 1

You may use the Cloud Shell code editor to view and edit the contents of these files.

Click on the ✎ icon on the top right of your Cloud Shell window to launch Code Editor

Step 2

After it is launched, navigate to the following directory: training-data-analyst/courses/machine_learning/deepdive/06_structured/labs/serving/pipeline/src/main/java/com/google/cloud/training/mlongcp

Step 3

Open the **BabyweightMLService.java** files and replace #*TODO*s in the code.

Step 4

Once completed, go into your Cloud Shell and run the `run_once.sh` script to test your ML service

```
cd ~/training-data-analyst/courses/machine_learning/deepdive/06_structured/serving
./run_once.sh
```

9. Serve predictions for batch requests

This section of the lab calls AddPrediction.java that takes a batch input (one big CSV), calls the prediction service to generate baby weight predictions and writes them into local files (multiple CSVs).

Step 1

In your Cloud Shell code editor, open the **AddPrediction.java** file available in the following directory: `training-data-analyst/courses/machine_learning/deepdive/06_structured/labs/serving/pipeline/src/main/java/com/google/cloud/training/mlongcp`

Step 2

Look through the code and notice how, based on input argument, it decides to set up a batch or streaming pipeline, and creates the appropriate TextInputOutput or PubSubBigQuery io object respectively to handle the reading and writing.

Note: Look back at the diagram in "how code is organized" section to make sense of it all.

Step 3

Test batch mode by running the `run_ontext.sh` script provided in the lab directory:

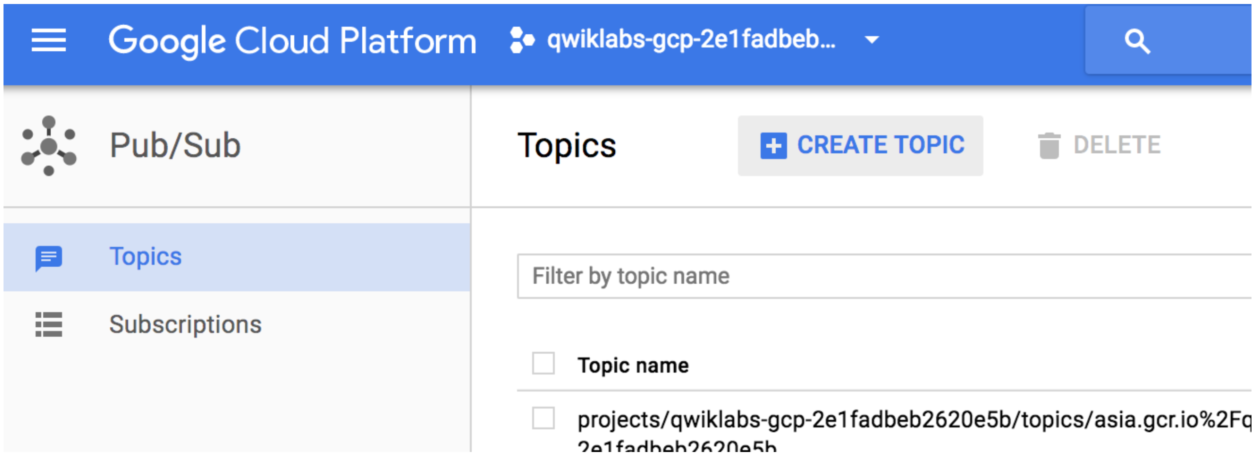
```
cd ~/training-data-analyst/courses/machine_learning/deepdive/06_structured/serving
./run_ontext.sh
```

10. Serve predictions real-time with a streaming pipeline

In this section of the lab, you will launch a streaming pipeline with Dataflow, which will accept incoming information from Cloud Pub/Sub, use the info to call the prediction service to get baby weight predictions, and finally write that info into a BigQuery table.

Step 1

On your GCP Console's left-side menu, go into Pub/Sub and click the "Create Topic" button on top. Create a topic called **babies**.



□	Topic name
□	projects/qwiklabs-gcp-2e1fadbeb2620e5b/topics/asia.gcr.io%2Fq 2e1fadbeb2620e5b

Step 2

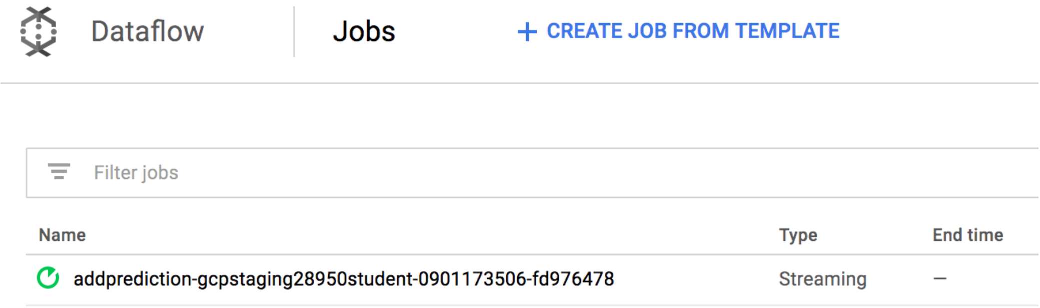
Back in your Cloud Shell, modify the script **run_dataflow.sh** to get Project Id (using *--project*) from command line arguments, and then run as follows:

```
cd ~/training-data-analyst/courses/machine_learning/deepdive/06_structured/serving
./run_dataflow.sh
```

This will create a streaming Dataflow pipeline.

Step 3

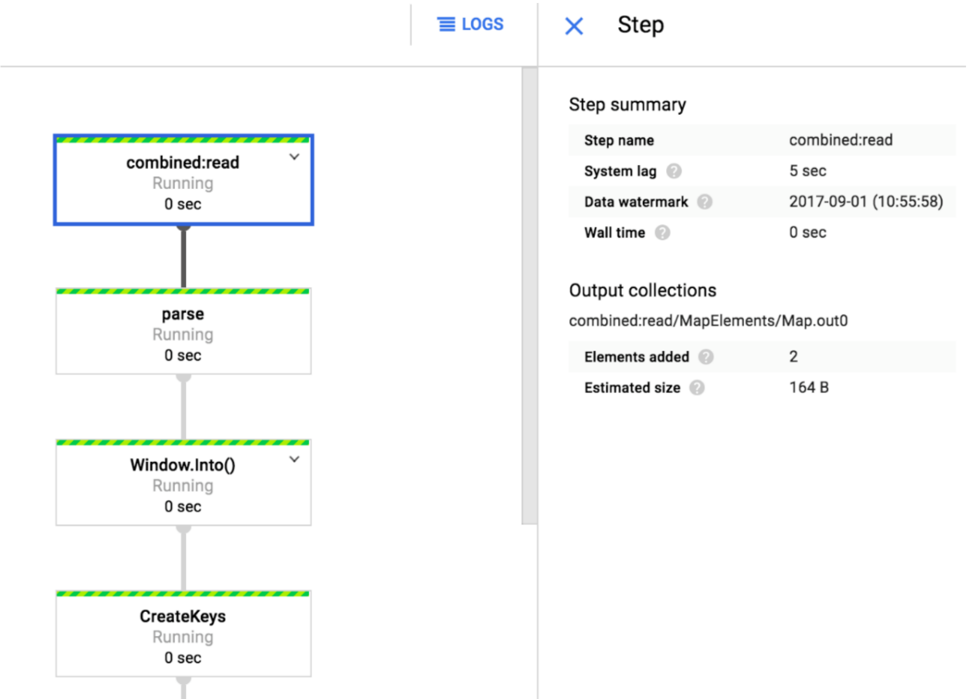
Back in your GCP Console, use the left-side menu to go into Dataflow and verify that the streaming job is created.



Name	Type	End time
addprediction-gcpstaging28950student-0901173506-fd976478	Streaming	–

Step 4

Next, click on the job name to view the pipeline graph. Click on the pipeline steps (boxes) and look at the run details (like system lag, elements added, etc.) of that step on the right side.



Step

Step summary

Step name	combined:read
System lag	5 sec
Data watermark	2017-09-01 (10:55:58)
Wall time	0 sec

Output collections

combined:read/MapElements/Map.out0

Elements added	2
Estimated size	164 B

This means that your pipeline is running and waiting for input. Let's provide input through the Pub/Sub topic.

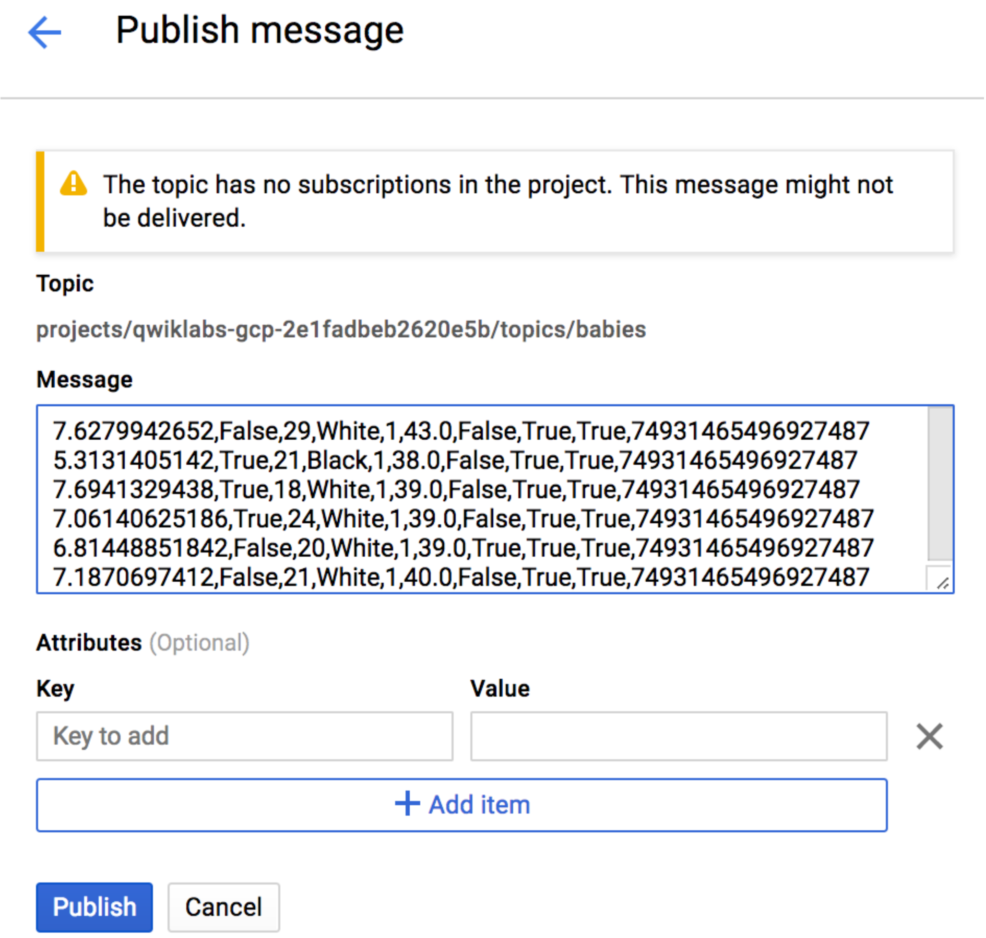
Step 5

Copy some lines from your example.csv.gz

```
cd ~/training-data-analyst/courses/machine_learning/deepdive/06_structured/serving
zcat exampledata.csv.gz
```

Step 6

On your GCP Console, go back into Pub/Sub, click on the **babies** topic, and then click on "Publish message" button on top. In the message box, paste the lines you just copied from exampledata.csv.gz and click on **Publish** button.



Publish message

The topic has no subscriptions in the project. This message might not be delivered.

Topic

projects/qwiklabs-gcp-2e1fadbeb2620e5b/topics/babies

Message

```
7.6279942652,False,29,White,1,43.0,False,True,True,74931465496927487
5.3131405142,True,21,Black,1,38.0,False,True,True,74931465496927487
7.6941329438,True,18,White,1,39.0,False,True,True,74931465496927487
7.06140625186,True,24,White,1,39.0,False,True,True,74931465496927487
6.81448851842,False,20,White,1,39.0,True,True,True,74931465496927487
7.1870697412,False,21,White,1,40.0,False,True,True,74931465496927487
```

Attributes (Optional)

Key	Value
Key to add	

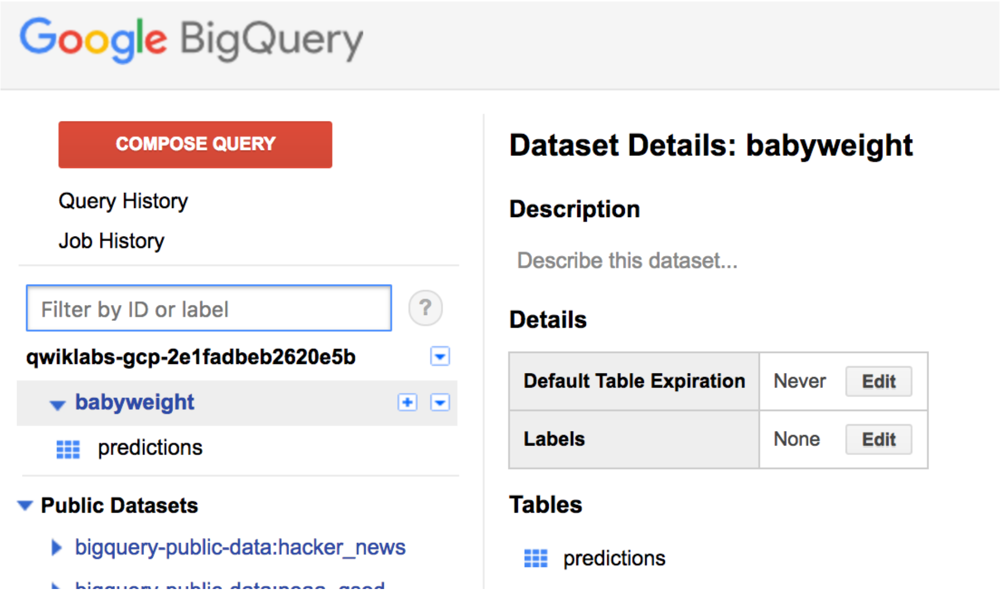
Step 7

You may go back into Dataflow jobs on your GCP Console, click on your job and see how the run details have changed for the steps, for example click on write_toBQ and look at Elements added.

Step 8

Lets verify that the predicted weights have been recorded into the BigQuery table. On your GCP console, click on BigQuery. This typically opens a new tab and may ask for you qwiklabs account's password. Once entered, you will be redirected to BigQuery console. Look at the left-side menu and you should see the **babyweight** dataset. Click on the blue down arrow to its left, and you should see your **prediction** table.

Note: If you do not see the prediction table, give it a few minutes as the pipeline has allowed-latency and that can add some delay.



Dataset Details: babyweight

Description

Describe this dataset...

Details

Default Table Expiration	Never	Edit
Labels	None	Edit

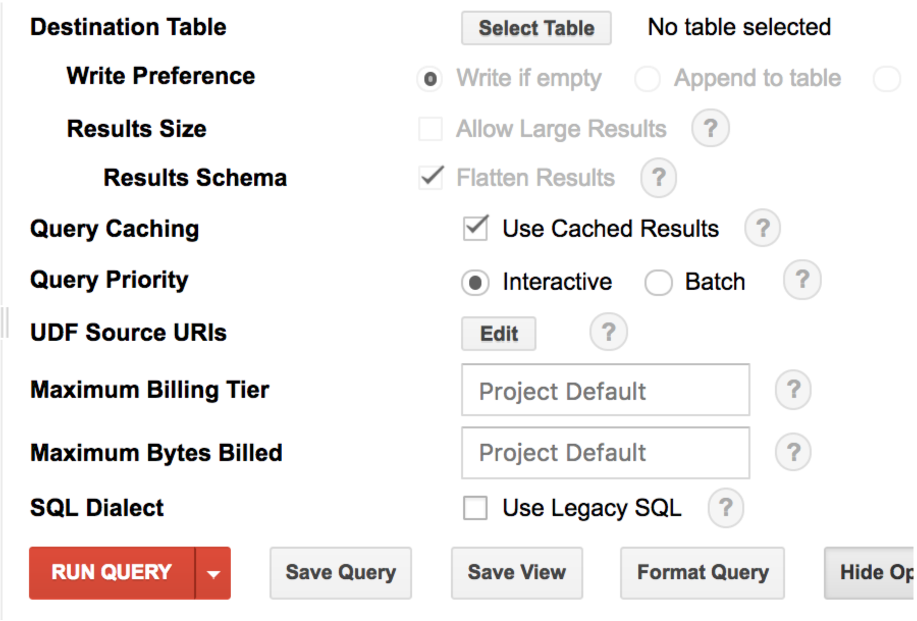
Tables

predictions

Step 9

Click on Compose Query button on the top left. Type the query below in the query box to retrieve rows from your predictions table. Click on **Show Options** button under the query box and uncheck "**Use Legacy SQL**".

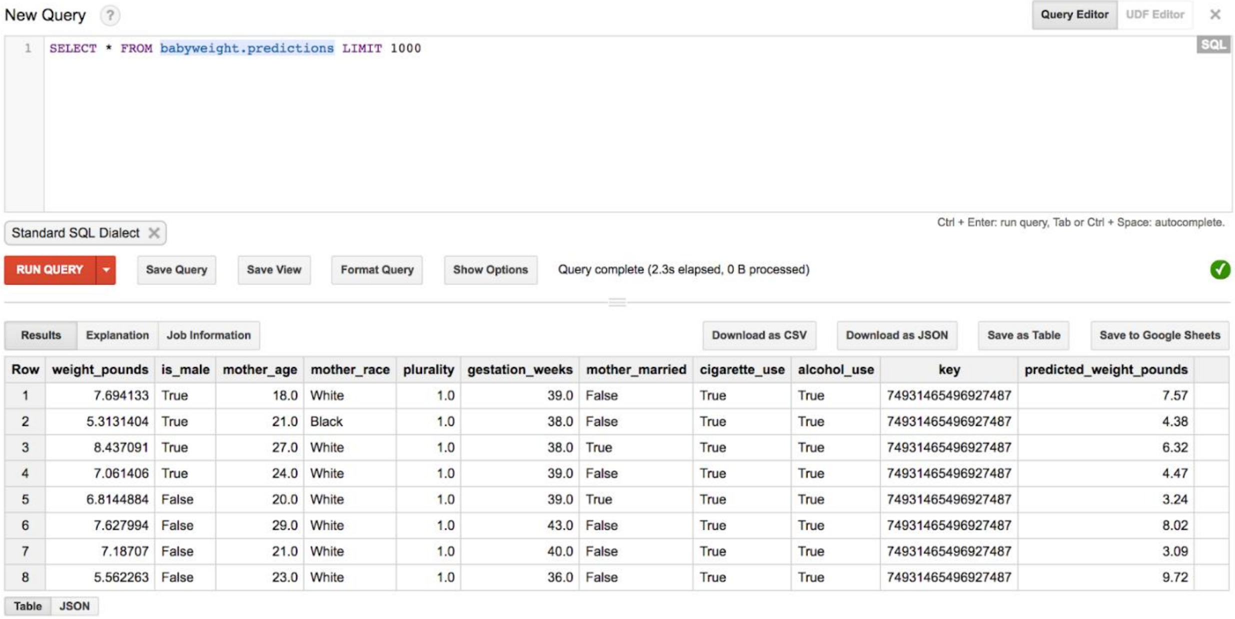
```
SELECT * FROM babyweight.predictions LIMIT 1000
```



Destination Table	Select Table No table selected
Write Preference	● Write if empty ○ Append to table ○
Results Size	□ Allow Large Results ?
Results Schema	✓ Flatten Results ?
Query Caching	☑ Use Cached Results ?
Query Priority	● Interactive ○ Batch ?
UDF Source URIs	Edit ?
Maximum Billing Tier	Project Default ?
Maximum Bytes Billed	Project Default ?
SQL Dialect	□ Use Legacy SQL ?

Step 10

Click the Run Query button. Notice the **predicted_weights_pounds** column in the result.



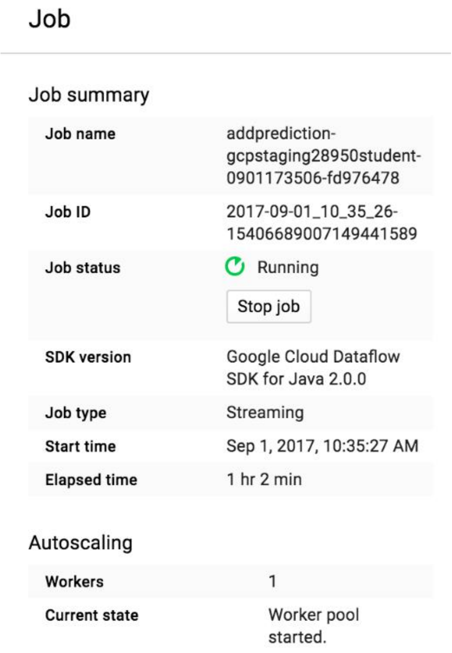
```
SELECT * FROM babyweight.predictions LIMIT 1000
```

Query complete (2.3s elapsed, 0 B processed)

Row	weight_pounds	is_male	mother_age	mother_race	plurality	gestation_weeks	mother_married	cigarette_use	alcohol_use	key	predicted_weight_pounds
1	7.694133	True	18.0	White	1.0	39.0	False	True	True	74931465496927487	7.57
2	5.3131404	True	21.0	Black	1.0	38.0	False	True	True	74931465496927487	4.38
3	8.437091	True	27.0	White	1.0	38.0	True	True	True	74931465496927487	6.32
4	7.061406	True	24.0	White	1.0	39.0	False	True	True	74931465496927487	4.47
5	6.8144884	False	20.0	White	1.0	39.0	True	True	True	74931465496927487	3.24
6	7.627994	False	29.0	White	1.0	43.0	False	True	True	74931465496927487	8.02
7	7.18707	False	21.0	White	1.0	40.0	False	True	True	74931465496927487	3.09
8	5.562263	False	23.0	White	1.0	36.0	False	True	True	74931465496927487	9.72

Step 11

Remember that your pipeline is still running. You can publish additional messages from your example.csv.gz and verify new rows added to your predictions table. Once you are satisfied, you may stop the Dataflow pipeline by going into your Dataflow Jobs page, and click the **Stop job** button on the right side Job summary window.



Job summary

Job name	addprediction-gcpstaging28950student-0901173506-fd976478
Job ID	2017-09-01_10_35_26-15406689007149441589
Job status	Running
SDK version	Google Cloud Dataflow SDK for Java 2.0.0
Job type	Streaming
Start time	Sep 1, 2017, 10:35:27 AM
Elapsed time	1 hr 2 min

Autoscaling

Workers	1
Current state	Worker pool started.