Use Case Industrialization

Christelle Lusso

@christelle.lusso · Member since March 01, 2023

Feb 29, 2024

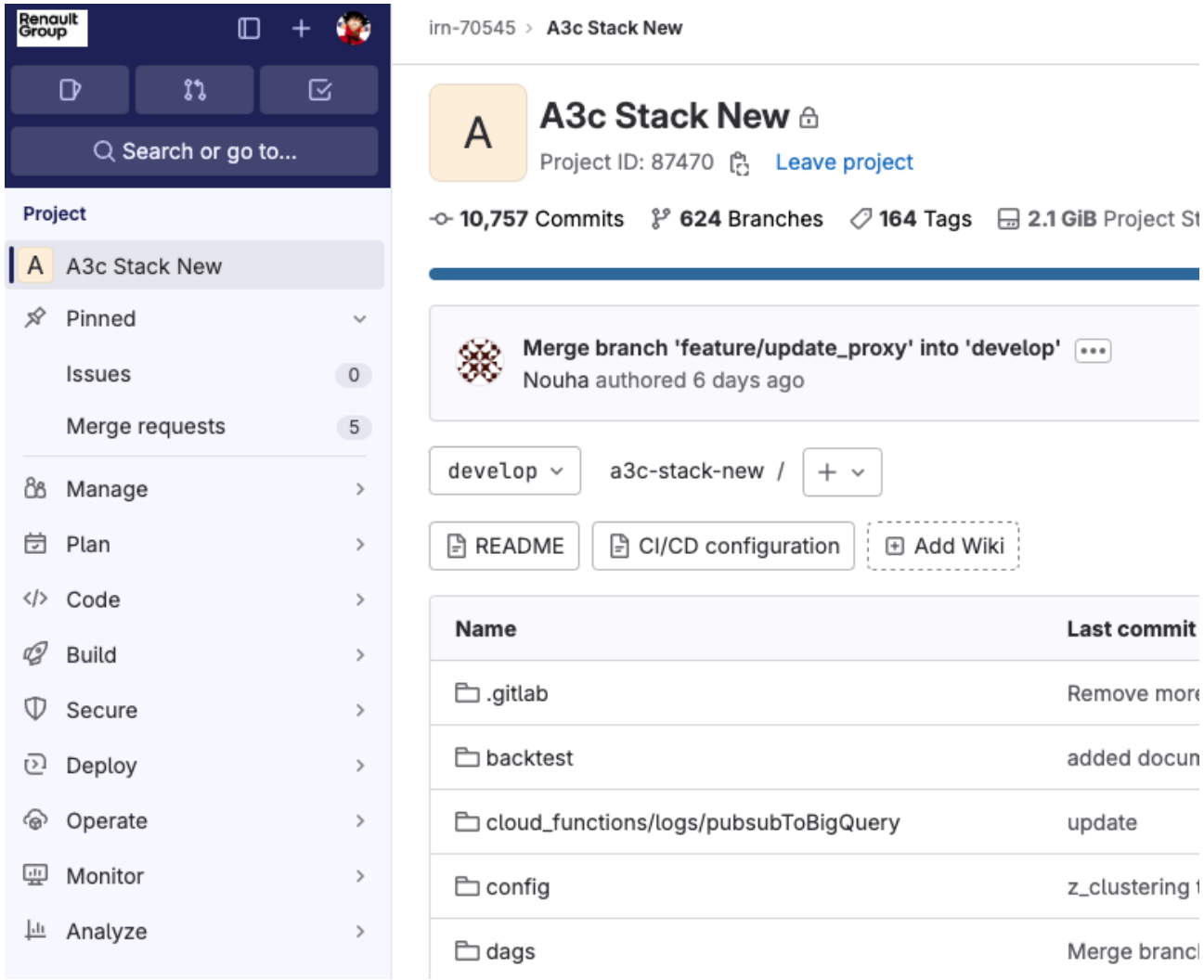
Plan

Introduction

- How to industrialize a machine learning model from A to Z.
- Context:
  - A3CT project.
  - **BigQuery ML** (versus Python/Spark).
  - For newbies.

A3c Stack New Project

1. Add your Use Case to the A3c Stack New Project: https://gitlabee.dt.renault.com/irn-70545/a3c-stack-new.

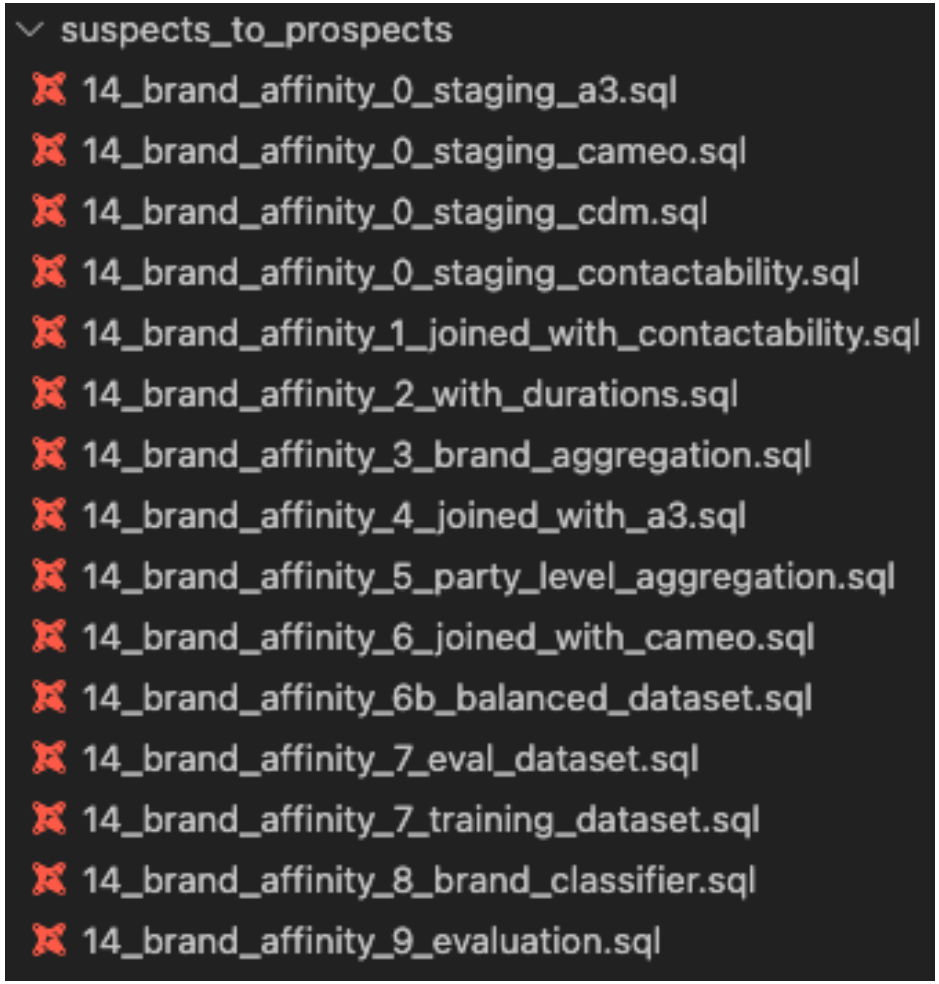


2. In folder **dags/reporting/sql** ⇒ create **my_use_case_folder**.
3. In **my_use_case_folder** ⇒ add my **.sql files**.

2. In folder **dags/reporting/sql** ⇒ create **my_use_case_folder**.
3. In **my_use_case_folder** ⇒ add my **.sql files**.

File names standardization

1. Code of my use case (mdl_cd).
2. Name of my use case (mdl_typ).
3. Step number.
4. Step name.

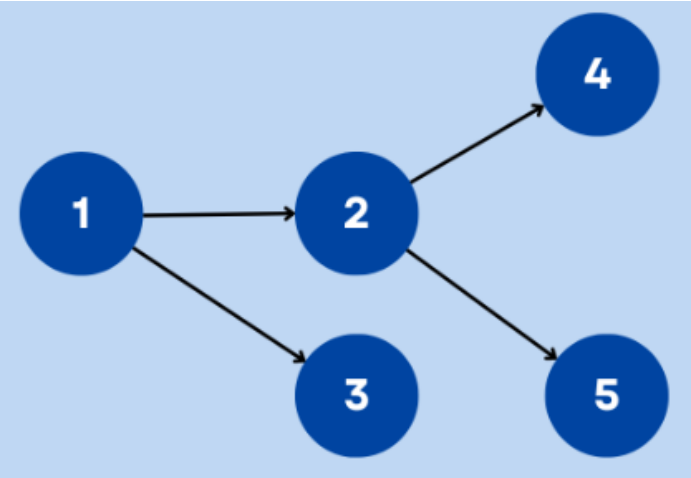


Plan

Add DAGs to pipeline

What is a DAG?

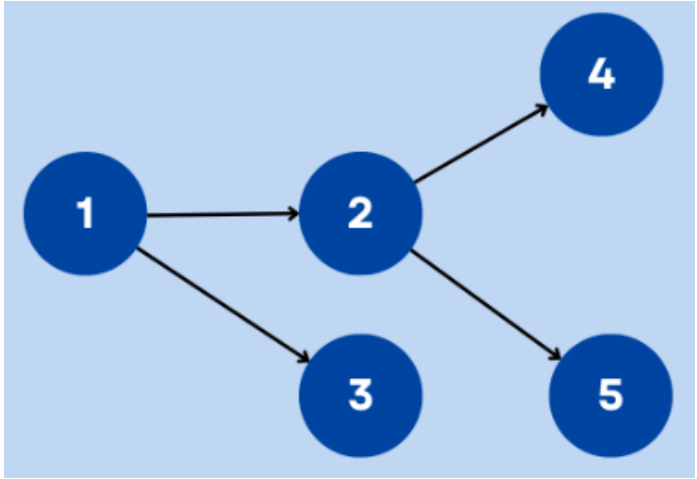
1. **DAG** = Directed Acyclic Graph.
2. Represents dependencies between steps.
3. $\Rightarrow$ sequences of steps or parallel steps.



Add DAGs to pipeline

What is a DAG?

1. **DAG** = Directed Acyclic Graph.
2. Represents dependencies between steps.
3. $\Rightarrow$ sequences of steps or parallel steps.



How to add DAG?

Complete files:

1. **A3ctBigQueryMLVariables.py** (DAG configuration)
2. **prep_dashboard.py** (DAG creation)

located in **dags/reporting** folder.

Plan

A3ctBigQueryMLVariables.py

- Contains DAG configuration in json format.
- ⇒ Define **my_use_case** training/predict DAG dictionary.

```
brand_affinity_training = {
    "area": "bqmlTest",
    "name": "brand_affinity_training",
    "description": "Training pipeline for Brand Affinity",
    "step_list": [
        {
            "name": "staging",
            "type": "table",
            "table_list": [
                {
                    "table": "14_brand_affinity_0_staging_cdm",
                    "dataset_dest": "bqml_uc",
                    "action": "view",
                    "file": {
                        "file_path": "sql/brand_affinity/...,
                        "file_type": 'sql'
                    }
                },
            ...
        },
```

- All my .sql files are listed in **step_list** with specific parameters:
  - "type": "**table**"
  - "table_list": ...
  - ⇒ CREATE **TABLE**
- Tables are created in **dataset_dest** folder.

```
brand_affinity_training = {
    "area": "bqmlTest",
    "name": "brand_affinity_training",
    "description": "Training pipeline for Brand Affinity",
    "step_list": [
        {
            "name": "staging",
            "type": "table",
            "table_list": [
                {
                    "table": "14_brand_affinity_0_staging_cdm",
                    "dataset_dest": "bqml_uc",
                    "action": "view",
                    "file": {
                        "file_path": "sql/brand_affinity/...,
                        "file_type": 'sql'
                    }
                },
            ...
        },
```

BigQuery ML part

- All my .sql files are listed in **step_list** with specific parameters:
  - "type": "**model**"
  - "bqml_list": ...
  - ⇒ CREATE **MODEL**

```
{
    "name": "create_model",
    "type": "model",
    "bqml_list": [
        {
            "table": "14_brand_affinity_8_brand_classifier",
            "action": "train",
            "file": {
                "file_path": "sql/brand_affinity/...,
                "file_type": 'sql',
             }
        },
    ]

},
    ...
```

Train XGBoost model with BigQuery ML

14_brand_affinity_8_brand_classifier.sql

```
create or replace model
`irn-70545-dev-54.bqml_uc.14_brand_affinity_8_brand_classifier`
options (
    model_type='BOOSTED_TREE_CLASSIFIER',
    auto_class_weights=true,
    input_label_cols=['suspects_brand'],
    enable_global_explain=true
) as
select * except (
  ref_country_cd,
  prty_id
)
from
`irn-70545-dev-54.bqml_uc.14_brand_affinity_7_training_dataset`
```

Precision on how to append to a3c_scores

a3c_scores columns:

- ref_country_cd,
- prty_id,
- vhcl_brand_nm,
- mdl_segmentation_ctgry,
- mdl_segmentation_ctgry_cd,
- mdl_sub_typ,
- mdl_sub_cd,
- vhcl_brand_cd,
- mdl_cd,
- mdl_typ,
- vhcl_vin_id,
- mdl_probability_nb,
- mdl_score_ranking_nb,
- mdl_validity_start_dt,
- mdl_validity_end_dt,
- mdl_version,

Precision on how to append to a3c_scores

a3c_scores columns:

- ref_country_cd,
- prty_id,
- vhcl_brand_nm,
- mdl_segmentation_ctgry,
- mdl_segmentation_ctgry_cd,
- mdl_sub_typ,
- mdl_sub_cd,
- vhcl_brand_cd,
- mdl_cd,
- mdl_typ,
- vhcl_vin_id,
- mdl_probability_nb,
- mdl_score_ranking_nb,
- mdl_validity_start_dt,
- mdl_validity_end_dt,
- mdl_version,

⇒ make sure to use the right types:

```
-- OK
ref_country_cd,
-- OK
prty_id,
-- we are trying to predict brand of interest
cast(null as string) as vhcl_brand_nm,
-- not segmentation
cast(null as string) as mdl_segmentation_ctgry,
cast(null as string) as mdl_segmentation_ctgry_cd,
cast(null as string) as mdl_sub_typ,
cast(null as int64) as mdl_sub_cd,
cast(null as string) as vhcl_brand_cd,
14 as mdl_cd,
"BRAND_AFFINITY" as mdl_typ,
cast(null as string) as vhcl_vin_id,
round(renault_score, 2) as mdl_probability_nb,
decile as mdl_score_ranking_nb,
current_date() as mdl_validity_start_dt,
date_add(current_date(), interval 7 day) as mdl_validity_end_dt,
"v0" as mdl_version,
```

Append my results to a3c_scores table:

```
{
    "name": "append_to_a3c_scores",
    "type": "table",
    "table_list": [
        {
            "table": "a3c_scores",
            "dataset_dest": "prediction",
            "action": "query",
            "file": {
                "file_path": "sql/brand_affinity/all_suspects/14_brand_affinity_8_append_to_a3c_scores",
                "file_type": "sql",
                "write_disposition": "WRITE_APPEND",
            },
        },
    ],
},
```

- Parameter dataset_dest = **prediction**.
- Parameter write_disposition = **WRITE_APPEND**.

Plan

prep_dashboard.py

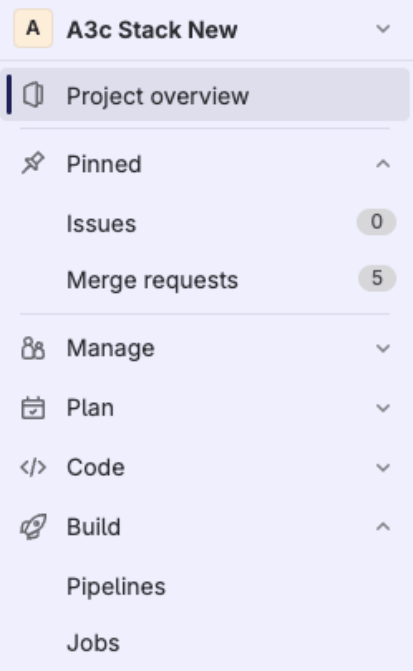
Create DAGs

- Create BigQueryPipeline instance with previous configuration.
- Create DAG with **make_dag_dashboard**.

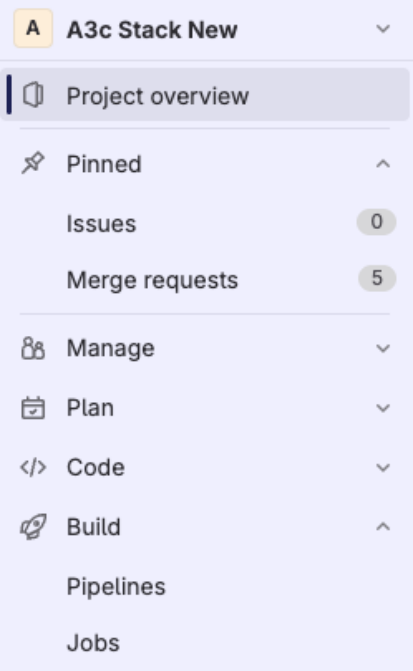
```
dag_bqml_brand_affinity_training = BigQueryPipeline(
    conf_dashboard=a3ct_bqml.brand_affinity_training,
    conf_bqml=a3ct_bqml.brand_affinity_training
)
BRAND_AFFINITY_TRAINING_DAG = dag_bqml_brand_affinity_training.make_dag_dashboard(schedule_interval="0 0 * * 3")

dag_bqml_brand_affinity_predictions = BigQueryPipeline(
    conf_dashboard=a3ct_bqml.brand_affinity_predictions,
    conf_bqml=a3ct_bqml.brand_affinity_predictions
)
BRAND_AFFINITY_PREDICTIONS_DAG = dag_bqml_brand_affinity_predictions.make_dag_dashboard(schedule_interval="0 0 * *
```

Plan

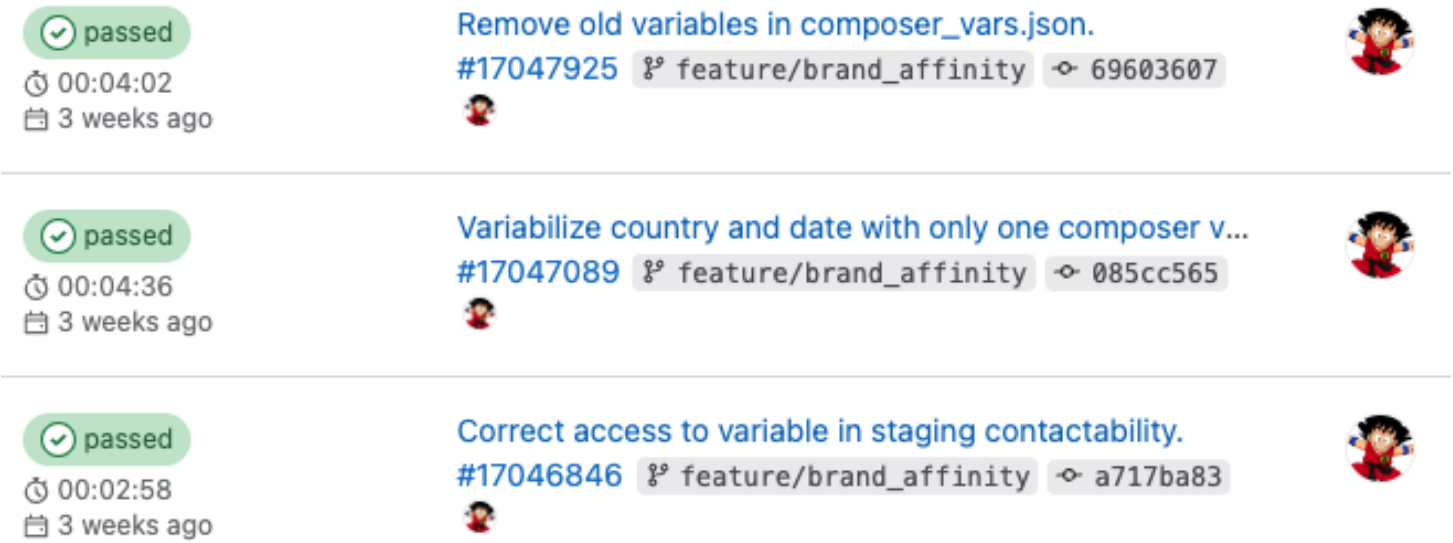
- Push your code branch on GitLab EE.
- Go to the **Pipelines page** of the A3c Stack New project $\longrightarrow$



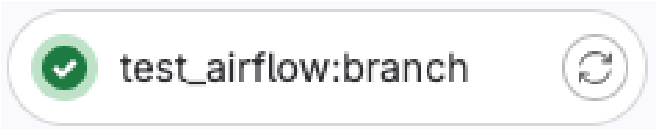
- Push your code branch on GitLab EE.
- Go to the **Pipelines page** of the A3c Stack New project $\longrightarrow$



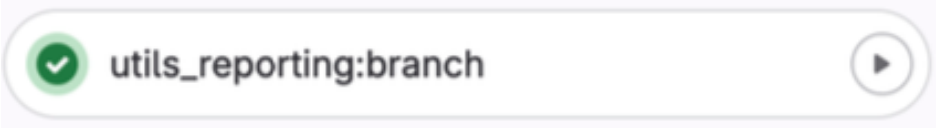
- Click on your most recent pipeline (**#17047925**):



- Airflow tests must pass.

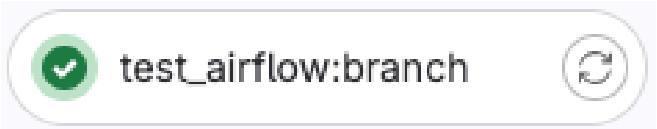


- Run your utils_reporting CI/CD step.

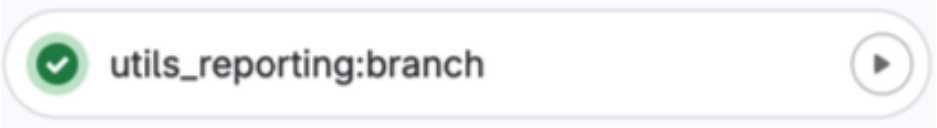


- $\Rightarrow$ All steps must **pass** successfully.

- Airflow tests must pass.



- Run your utils_reporting CI/CD step.



- $\Rightarrow$ All steps must **pass** successfully.

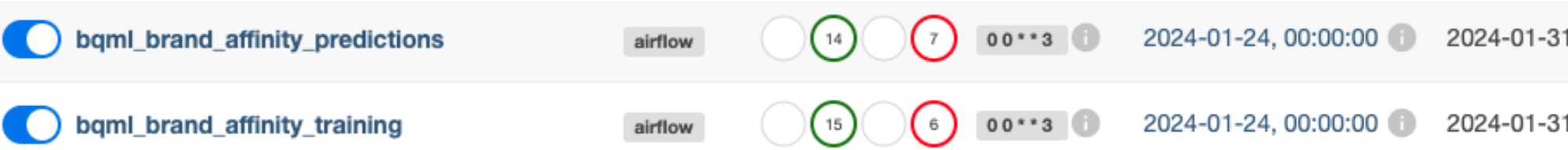
Failed test (✖)

Keep debugging until all tests pass:

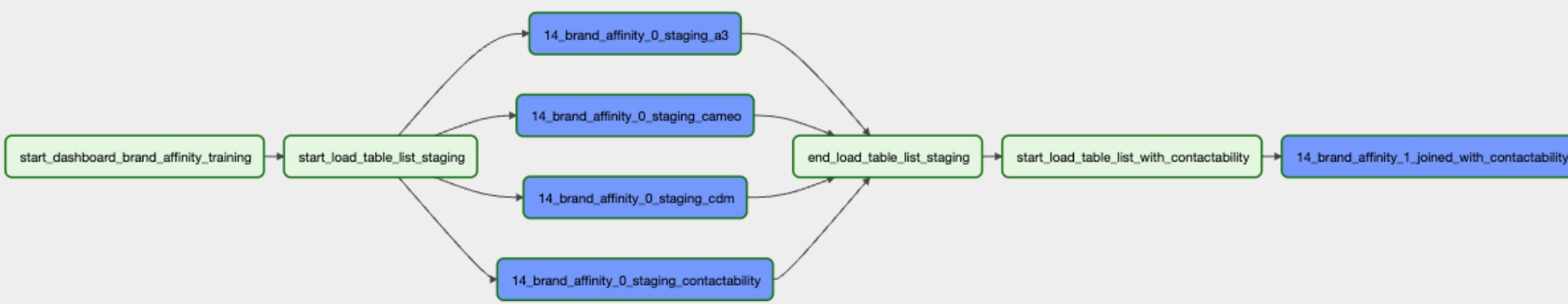
- Correct bugs.
- Push code.
- Run all tests.
- Wait for tests to finish.

Composer

- Go to the dev **Composer** page.
- Check that your DAGs are listed:



- Click on your DAG.
- Go to the Graph tab.



⇒ Wait for all steps to **pass** (shown in green).

Plan

Add Pub/Sub DAG

Pub/Sub (Publish/Subscribe)

- Why? Make results available.
- How? Publish them under a Pub/Sub topic.
- Consumers subscribe to the topic.

Add Pub/Sub DAG

Pub/Sub (Publish/Subscribe)

- Why? Make results available.
- How? Publish them under a Pub/Sub topic.
- Consumers subscribe to the topic.

Create Pub/Sub DAG

- Create a Pub/Sub DAG to publish results.
- ⇒ Publish my **most recent scores** from **a3c_scores**.

Add Pub/Sub DAG

Pub/Sub (Publish/Subscribe)

- Why? Make results available.
- How? Publish them under a Pub/Sub topic.
- Consumers subscribe to the topic.

Create Pub/Sub DAG

- Create a Pub/Sub DAG to publish results.
- ⇒ Publish my **most recent scores** from **a3c_scores**.

Trigger Pub/Sub DAG

- Publication must be done after each prediction run.
- ⇒ Trigger Pub/Sub DAG after predictions DAG.

How?

Complete files:

1. **scoring_export_dag.py** in dags/output.
2. **BigQueryPipeline.py** in dags/utils.
3. **prep_dashboard.py** in dags/reporting.

How?

Complete files:

1. **scoring_export_dag.py** in **dags/output**.
2. **BigQueryPipeline.py** in **dags/utils**.
3. **prep_dashboard.py** in **dags/reporting**.

Create Pub/Sub DAG

- Call **dag_publish_to_pub_sub** in **scoring_export_dag.py**:

```
export_scoring_brand_affinity = dag_publish_to_pub_sub(
    model_code=14,
    model_name="BRAND_AFFINITY",
)
```

- in **dags/output** folder.

Trigger Pub/Sub DAG

- Create* **add_pubsub** method in **BigQueryPipeline.py** file:

```
def add_pubsub(self, trigger_dag_id):
    """
    Add trigger dag run operator as final task
    ----------
    :param trigger_dag_id: string
    :return: airflow.models.DAG
    """
    [end] = self._dag.leaves
    end >> TriggerDagRunOperator(
        task_id="trigger_dagrun_export_pubsub",
        trigger_dag_id=trigger_dag_id,
        trigger_rule="all_success",
        dag=self._dag,
    )
```

* with Paul Beaujean

- in **dags/utils** folder.

prep_dashboard.py

Add Pub/Sub DAG to predictions DAG:

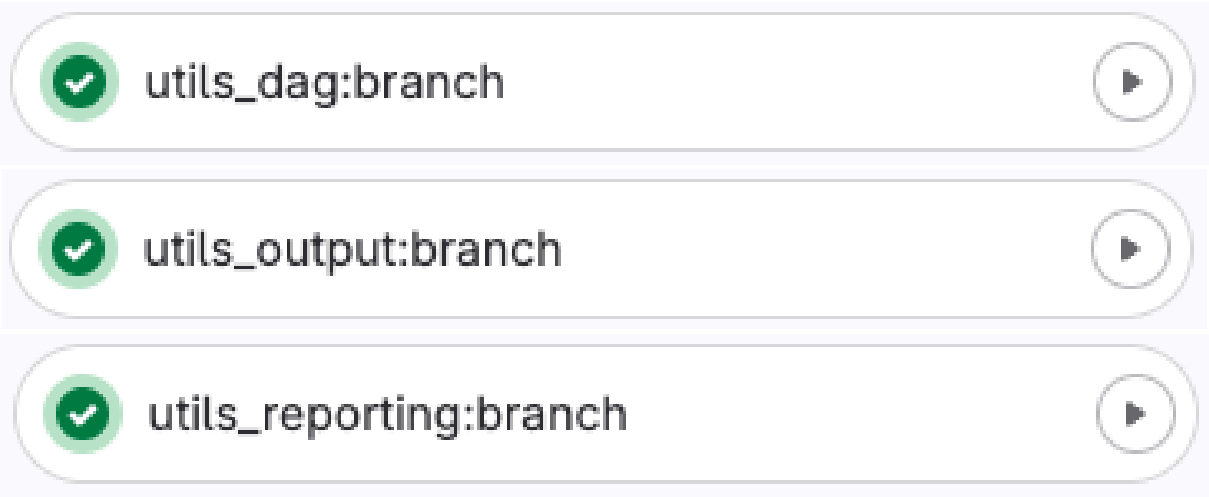
- Call **add_pubsub** method in **prep_dashboard.py**:

```
dag_bqml_brand_affinity_predictions = BigQueryPipeline(
    conf_dashboard=a3ct_bqml.brand_affinity_predictions,
    conf_bqml=a3ct_bqml.brand_affinity_predictions,
)
brand_affinity_predictions_dag = dag_bqml_brand_affinity_predictions.make_dag_dashboard(
    schedule_interval="0 0 * * 3"
)
brand_affinity_predictions_dag = dag_bqml_brand_affinity_predictions.add_pubsub("ml_brand_affinity_pubsub")
```

- in **dags/reporting** folder.

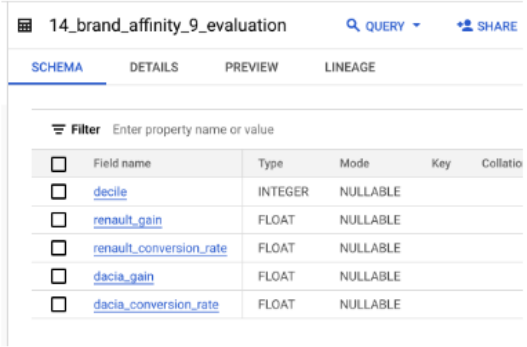
Caveat!

- Modifications in different dags sub-folders.
- ⇒ Run corresponding CI/CD steps.



Plan

- Rebase **develop** and fix merge conflicts locally.
- Create a **Merge Request** and assign a DE to it.
- Merge Request = get your commits accepted into **develop**.
- Fill in planning page at **MEP Planning**.

Sprint	Release	#	Semaine 3	Sujet/US Description	Plan de test validé
83			22 nov. 2023	Brand affinity	MEP en int 11 déc. 2023 MR: https://gitlabee.dt.renault.com/irn-70545/a3c-stack-new/-/merge_requests/483 1) Run the bqml_brand_affinity_training and bqml_brand_affinity_predicts dags and check it executes corre  2) * Check schema of table 14_brand_affinity_9_evaluation 

Field name	Type	Mode	Key	Collatio
decile	INTEGER	NULLABLE		
renault_gain	FLOAT	NULLABLE		
renault_conversion_rate	FLOAT	NULLABLE		
dacia_gain	FLOAT	NULLABLE		
dacia_conversion_rate	FLOAT	NULLABLE		

This is the END!

Go have some coffee...