💼

WORKSHOP 003: Happiness Score

🤵🏾 By: David Venté Polo

- Table of contents

Description

In this repository we will develop a training procedure for a model that allows us to predict happiness within a given space described by several variables. It is based on 5 files in csv format that besides describing the year they are from, contain information of different characteristics that will help us to achieve the objective.

This process was carried out by means of Airflow with two dags, the first one starts with the creation of the database where the data will be loaded; from this comes the listening of a Kafka channel (topic) for the reception of data that via streaming will be received to predict the value we are looking for and finally upload it to a database. While the Kafka channel is listening, the other dag extracts the data from the 5 files, joins and transforms them and then sends them through the same Kafka channel to the receiver.

- Tools used are:
    - Python 🐍.
    - Git 📦.
    - Postgres 📀.
    - PgAdmin 4 📠.
    - GitHub 💼.
    - Visual Studio Code 📝.
    - Apache Airflow.
    - Jupyter Notebook.
    - Docker desktop 💠

Purpose - Raison d'être

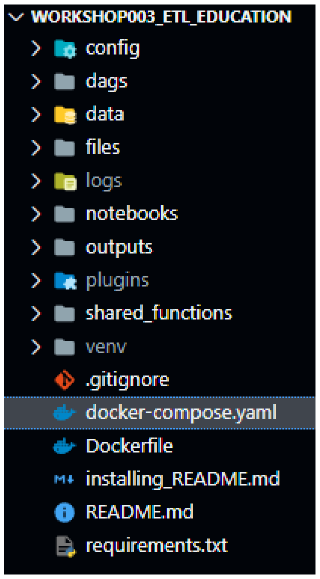
The project is aimed at demonstrating the skills necessary to train a model in a basic way and provide good results. In addition, to demonstrate the execution of Kafka and its data transmission.

This project is the first workshop of the ETL subject of Artificial Intelligence & Data Engineering at the Universidad Autónoma de Occidente under the teaching of Javier Alejandro Vergara Zorilla.

Process

The process starts with:

1. The creation of the container orchestration by means of the docker-compose.yaml, this is in the root of the document and allows us to create all the environment that we will use for the execution.



```
WORKSHOP003_ETL_EDUCATION
  config
  dags
  data
  files
  logs
  notebooks
  outputs
  plugins
  shared_functions
  venv
  .gitignore
  docker-compose.yaml
  Dockerfile
  installing_README.md
  README.md
  requirements.txt
```

2. We must run it with:

   ```
   docker-compose up
   ```

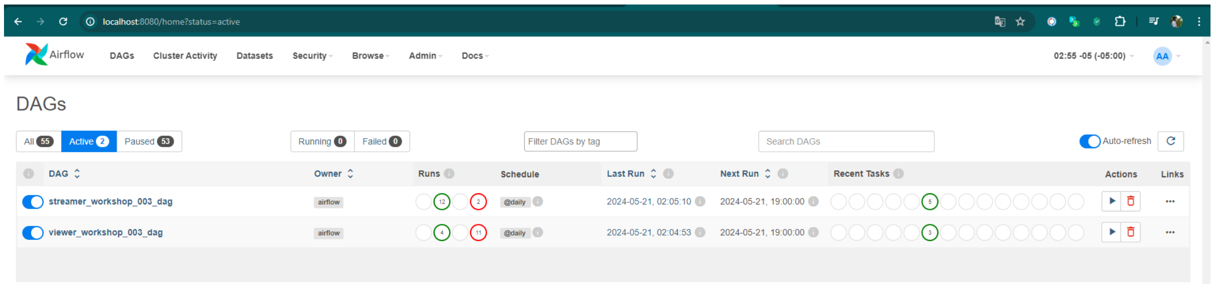
   And later, we execute:

   ```
   docker-compose up airflow-init
   ```

   This is to ensure that the environment is fully operational.

2. Let's access the following url: http://localhost:8080/home.

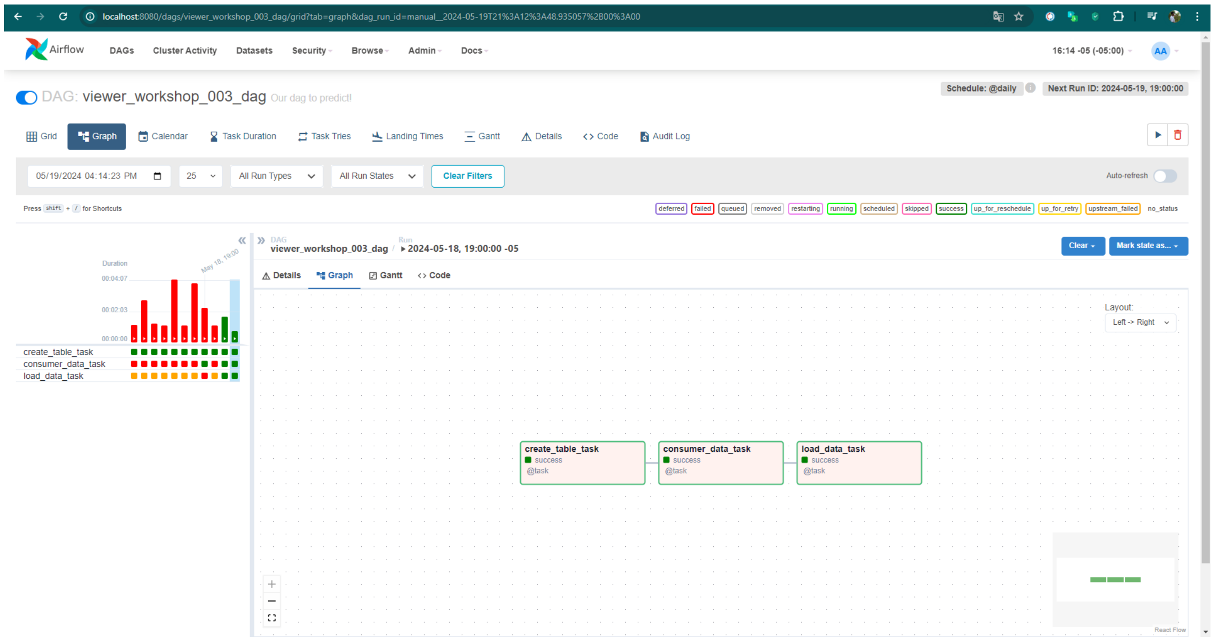
   In this one, after putting our initial credentials which are: **airflow** in username and **airflow** in password, we will see this:



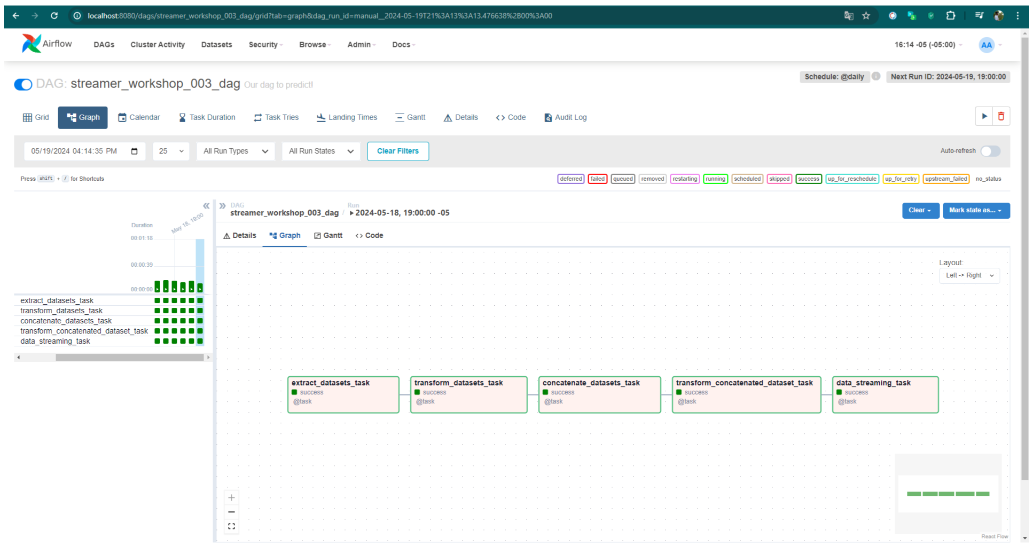
DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions	Links
streamer_workshop_003_dag	airflow	12 2	@daily	2024-05-21, 02:05:10	2024-05-21, 19:00:00	5		
viewer_workshop_003_dag	airflow	4 11	@daily	2024-05-21, 02:04:53	2024-05-21, 19:00:00	3		

We will usually see them turned off. So we start with **viewer_workshop_003_dag** and then **streamer_workshop_003_dag** .

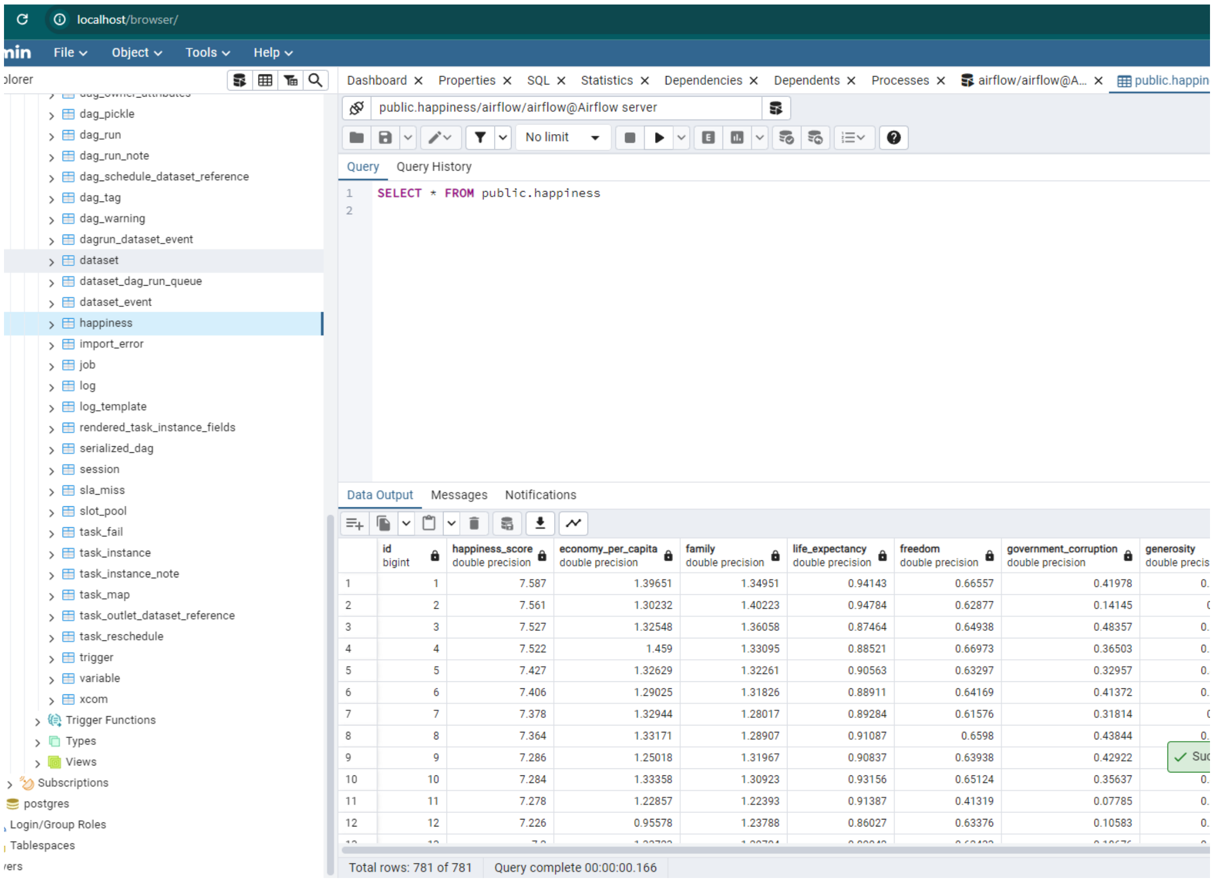
2. On the **viewer_workshop_003_dag** side we have the interaction with the database and the Kafka consumer. This must be the first to be executed because it must be aware of the messages that are sent.



3. On the other hand, there is **streamer_workshop_003_dag** that will interact with the 5 datasets, transform and merge them into one and then send their data through the Kafka producer.


localhost:8080/dags/streamer_workshop_003_dag/grid?tab=graph&dag_run_id=manual__2024-05-19T21%3A13%3A13.476638%2B00%3A00
DAG: streamer_workshop_003_dag
Schedule: @daily
Next Run ID: 2024-05-19, 19:00:00
streamer_workshop_003_dag ▸ 2024-05-18, 19:00:00 -05
extract_datasets_task
transform_datasets_task
concatenate_datasets_task
transform_concatenated_dataset_task
data_streaming_task
success
@task

4. Finally we validate its creation in the database using PgAdmin.



```
SELECT * FROM public.happiness
```

	id bigint	happiness_score double precision	economy_per_capita double precision	family double precision	life_expectancy double precision	freedom double precision	government_corruption double precision	generosity double precis
1	1	7.587	1.39651	1.34951	0.94143	0.66557	0.41978	0.
2	2	7.561	1.30232	1.40223	0.94784	0.62877	0.14145	
3	3	7.527	1.32548	1.36058	0.87464	0.64938	0.48357	0.
4	4	7.522	1.459	1.33095	0.88521	0.66973	0.36503	0.
5	5	7.427	1.32629	1.32261	0.90563	0.63297	0.32957	0.
6	6	7.406	1.29025	1.31826	0.88911	0.64169	0.41372	0.
7	7	7.378	1.32944	1.28017	0.89284	0.61576	0.31814	
8	8	7.364	1.33171	1.28907	0.91087	0.6598	0.43844	0.
9	9	7.286	1.25018	1.31967	0.90837	0.63938	0.42922	✓ Suc
10	10	7.284	1.33358	1.30923	0.93156	0.65124	0.35637	0.
11	11	7.278	1.22857	1.22393	0.91387	0.41319	0.07785	0.
12	12	7.226	0.95578	1.23788	0.86027	0.63376	0.10583	0.

Total rows: 781 of 781 Query complete 00:00:00.166

Results

The first data obtained (2015 dataset) are analyzed:

- **Happiness Rank:** Happiness ranges from 1 to 158, with a mean and median around 79,5. The high standard deviation (45,75) suggests a wide and relatively uniform distribution of happiness ranks.

- **Happiness Score (To predict):** Happiness scores range from 2.839 to 7.587, with a mean of 5,3757 and a median of 5,2325, indicating that most scores are centered around these values. The standard deviation (1.145) shows that there is moderate variability in happiness across countries.
- **The standard error** is low on average (0,047885), indicating that the happiness scores have reasonable precision. The small standard deviation (0,017146) suggests that this accuracy is relatively consistent across countries.
- **Economy (GDP per Capita):** GDP per capita varies considerably, with values ranging from 0 to 1,69042. The mean (0,846137) and median (0,910245) are near the center of the range, but the high standard deviation (0,403121) indicates considerable economic disparity between countries.

  Economy and financial well-being are often strongly correlated with happiness. The ability to meet basic needs and enjoy a good quality of life is highly dependent on GDP per capita.
- **Family:** Family support shows values from 0 to 1,40223, with a mean close to 1 (0,991046) and a median of 1,02951, suggesting that most countries have good family support. The standard deviation (0,272369) indicates moderate variability in this factor.
  This and interpersonal relationships are key factors in perceived happiness. Human beings are social by nature, and supportive relation ships significantly influence their happiness.
- **Health (Life Expectancy):** Life expectancy ranges from 0 to 1,02525, with a mean of 0,630259 and a median of 0,696705. The standard deviation (0,24707078) suggests significant differences in health between countries.

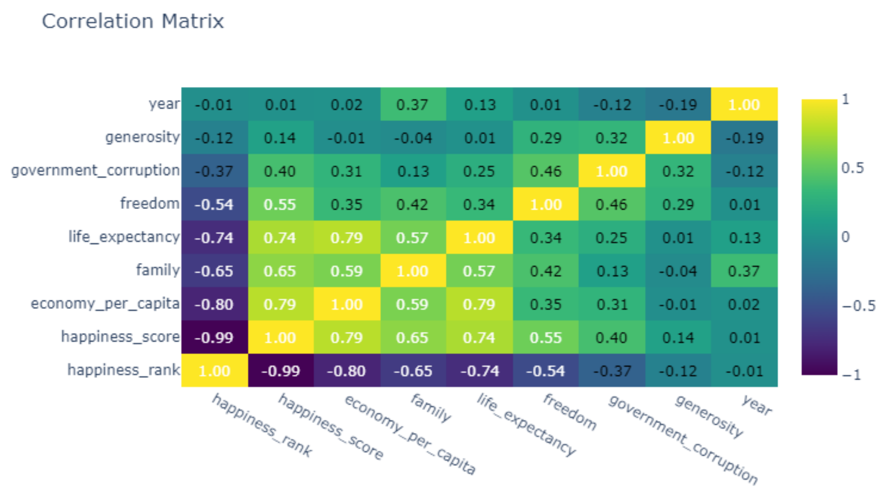
  Those are direct indicators of physical well-being, which has a considerable impact on happiness. People in good health tend to be happier.
- **Freedom:** Perception of freedom ranges from 0 to 0,66973, with a mean of 0,428615 and a median of 0,435515, indicating that most countries have a moderate perception of freedom.

  This to make decisions about one's life is an important component of happiness. Societies with greater personal freedoms tend to have happier citizens.
- **Trust (Government Corruption):** Trust in government (absence of corruption) ranges from 0 to 0,55191, with a low mean (0,143422) and a median of 0,10722. The standard deviation (0,120034) suggests that the perception of corruption varies significantly across countries.

  This in institutions and perceptions of government corruption affect emotional stability and confidence in the future. Less corruption generally correlates with greater happiness.
- **Generosity:** Generosity ranges from 0 to 0,79588, with a mean of 0,237296 and a median of 0,21613. The standard deviation (0,126685) indicates considerable variability in generosity across countries.

  Acts of generosity and altruism are related to higher levels of personal happiness. The ability and willingness to help others can increase perceptions of well-being.
- **Residual Dystopia:** The residual dystopia component ranges from 0,32858 to 3,60214, with a mean of 2,098977 and a median of 2,095415. The standard deviation (0,55355) suggests considerable variability in this component, reflecting a wide disparity in baseline happiness conditions not explained by other factors.
  This factor captures what remains of happiness that is not explained by the other factors. It is a crucial component in understanding relative happiness across countries.

Now, joining and in order to explore the variables to be used, I plot:



	happiness_rank	happiness_score	economy_per_capita	family	life_expectancy	freedom	government_corruption	generosity	year
year	-0.01	0.01	0.02	0.37	0.13	0.01	-0.12	-0.19	1.00
generosity	-0.12	0.14	-0.01	-0.04	0.01	0.29	0.32	1.00	-0.19
government_corruption	-0.37	0.40	0.31	0.13	0.25	0.46	1.00	0.32	-0.12
freedom	-0.54	0.55	0.35	0.42	0.34	1.00	0.46	0.29	0.01
life_expectancy	-0.74	0.74	0.79	0.57	1.00	0.34	0.25	0.01	0.13
family	-0.65	0.65	0.59	1.00	0.57	0.42	0.13	-0.04	0.37
economy_per_capita	-0.80	0.79	1.00	0.59	0.79	0.35	0.31	-0.01	0.02
happiness_score	-0.99	1.00	0.79	0.65	0.74	0.55	0.40	0.14	0.01
happiness_rank	1.00	-0.99	-0.80	-0.65	-0.74	-0.54	-0.37	-0.12	-0.01

As we can see, it would be a good match to take **economy_per_capita**, **family**, **life_expectancy** and **freedom** to predict the model, but that does not mean that we will deal with all columns.

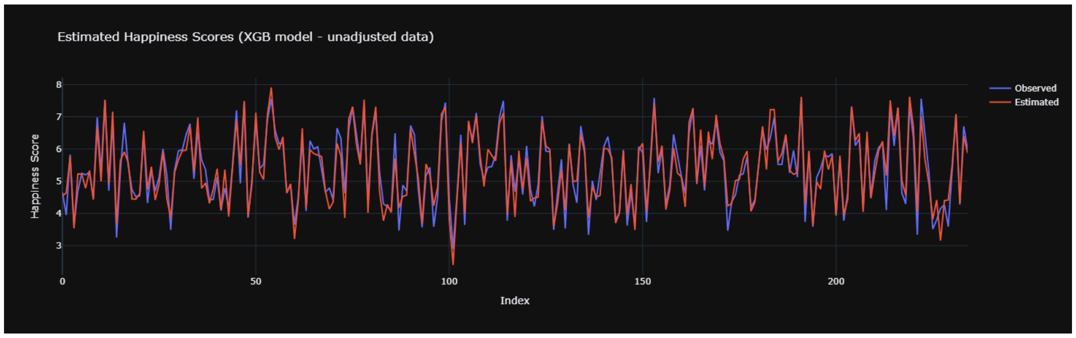
Additionally, we took the Country and Region data and joined them by means of a ' **-** ' separator to the Country or Region column which was subsequently renamed country_region and these data were treated as categorical using OneHotEncoder.

By performing several tests with selected models and columns, the best model is obtained with: **Random Forest Regressor** taking:

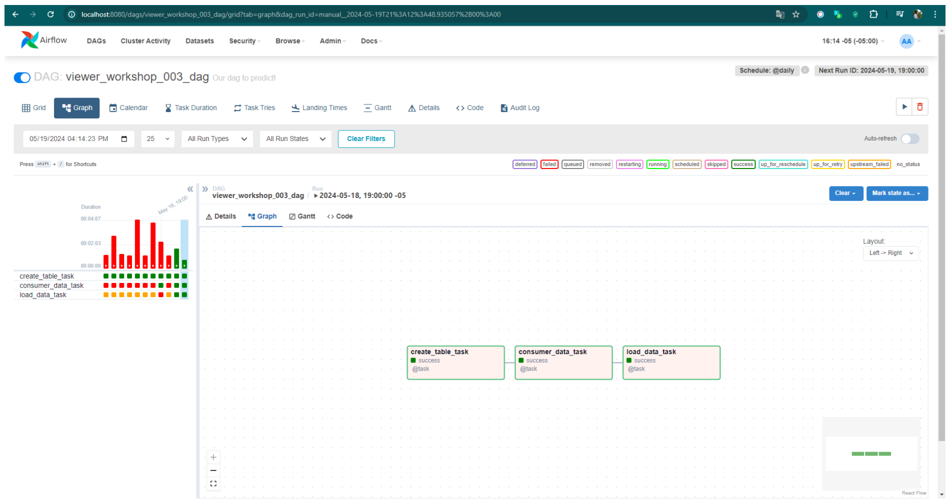
- **economy_per_capita**
- **family**
- **life_expectancy**
- **freedom**
- **government_corruption**
- **generosity**
- **year**
- **country_region**

Getting a **R2: 0.89995**.

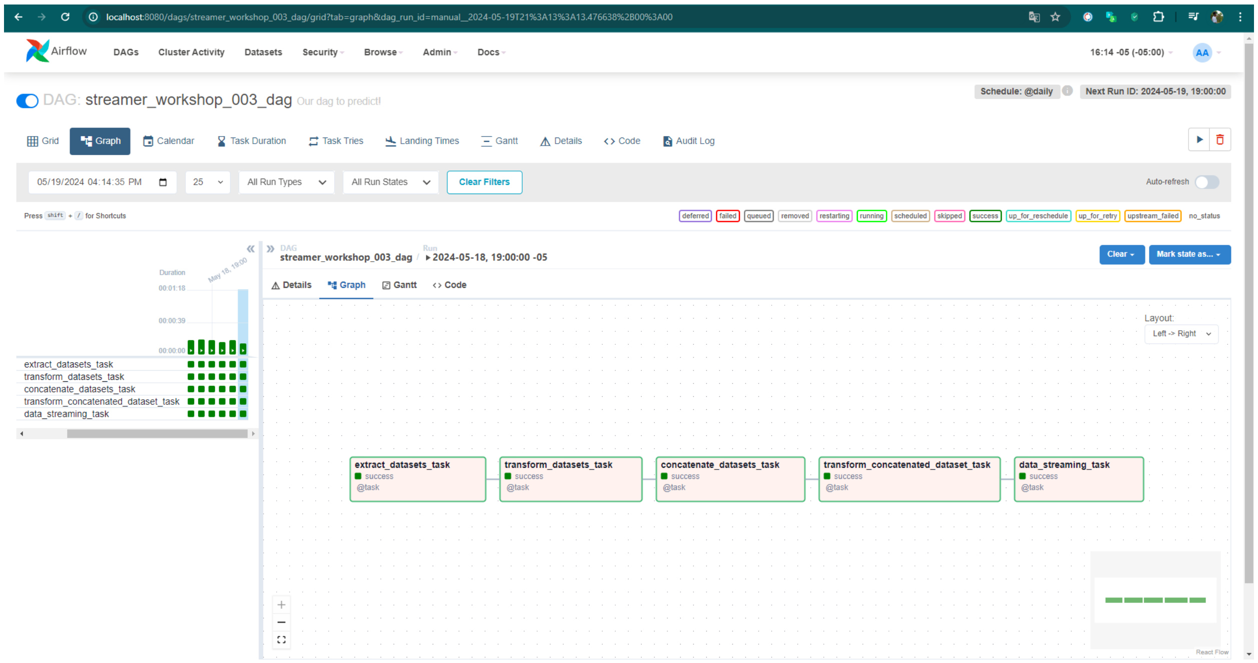
Achieving:



The dags are executed to validate the process:

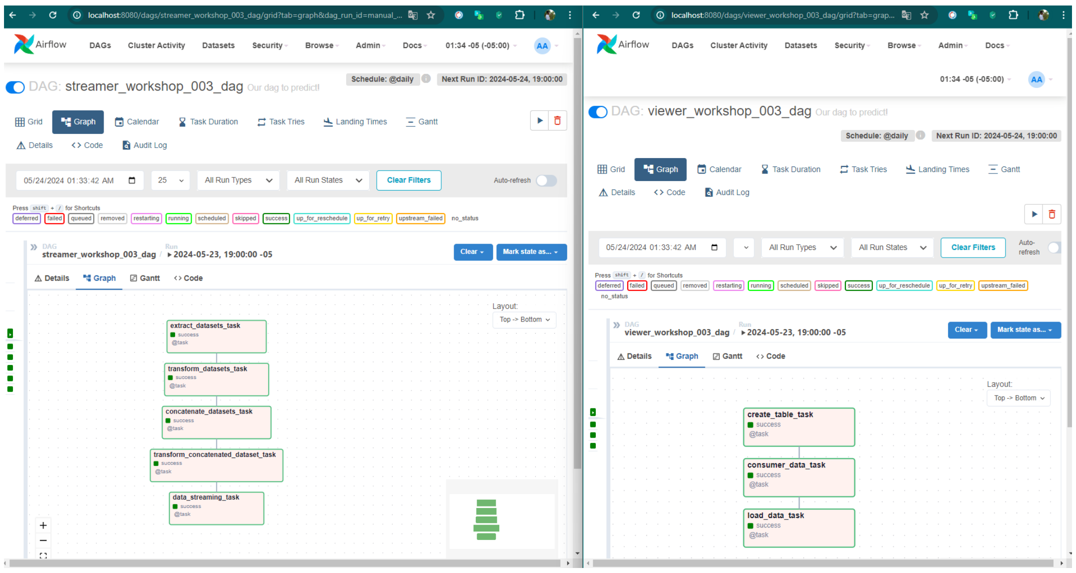


Complete and correct execution of dag Viewer_workshop_003_dag.



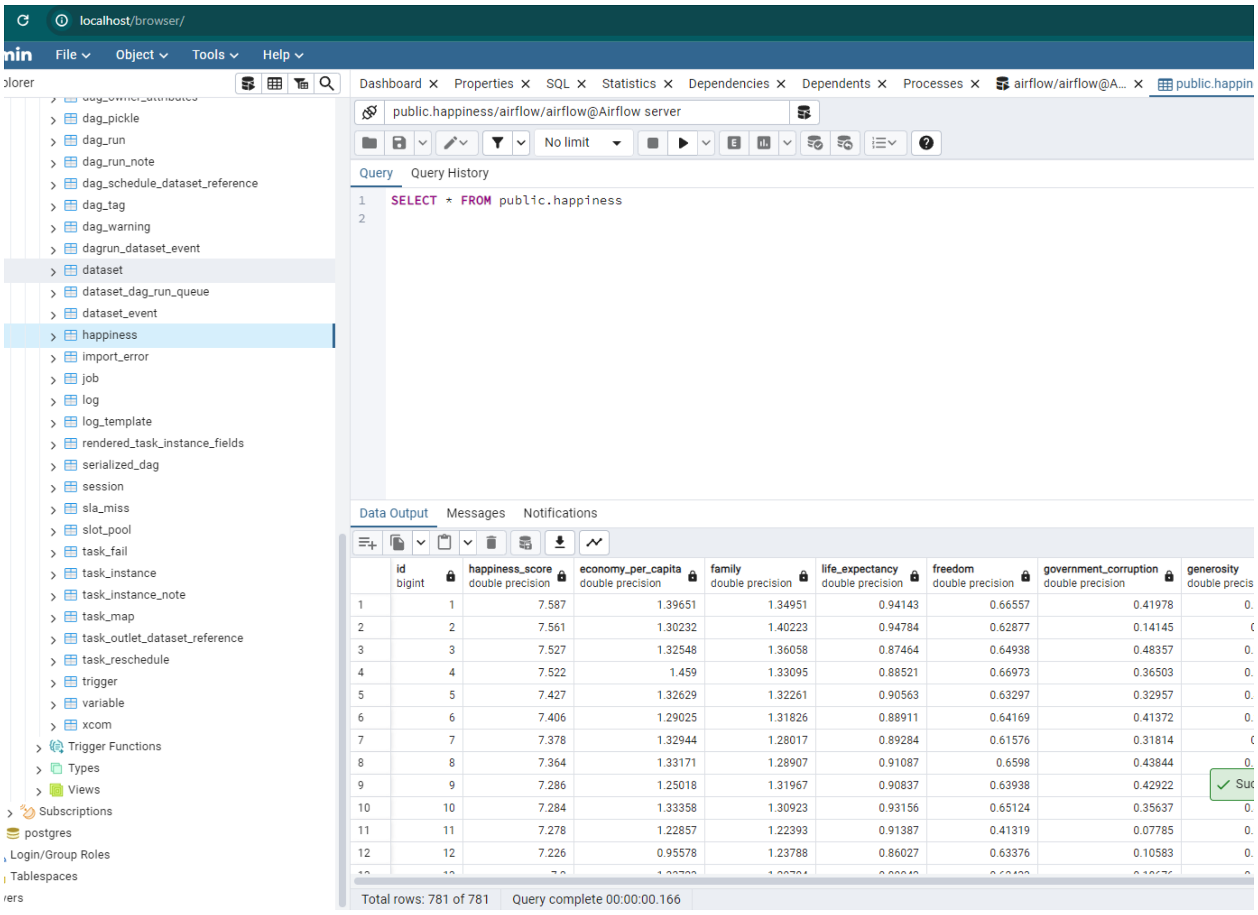
Complete and correct execution of dag Streamer_workshop_003_dag.

Both:



Check the execution both dags.

Checking data loaded in database:



```
SELECT * FROM public.happiness
```

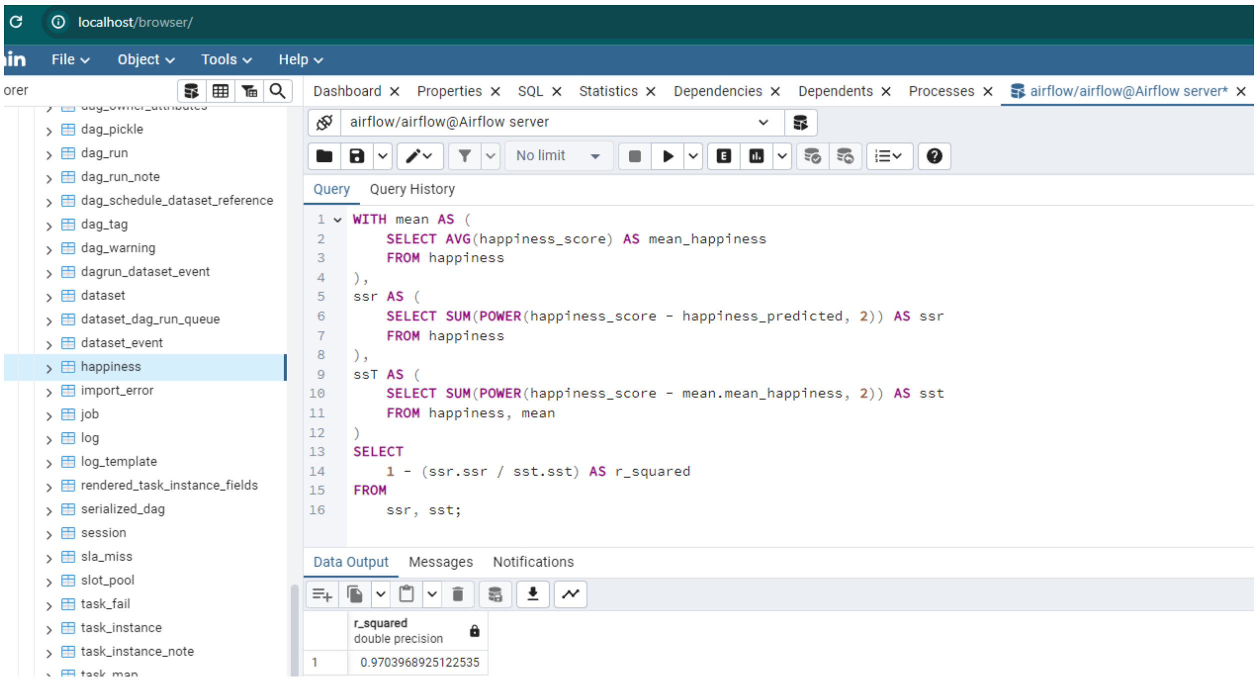
	id bigint	happiness_score double precision	economy_per_capita double precision	family double precision	life_expectancy double precision	freedom double precision	government_corruption double precision	generosity double precis
1	1	7.587	1.39651	1.34951	0.94143	0.66557	0.41978	0.
2	2	7.561	1.30232	1.40223	0.94784	0.62877	0.14145	(
3	3	7.527	1.32548	1.36058	0.87464	0.64938	0.48357	0.
4	4	7.522	1.459	1.33095	0.88521	0.66973	0.36503	0.
5	5	7.427	1.32629	1.32261	0.90563	0.63297	0.32957	0.
6	6	7.406	1.29025	1.31826	0.88911	0.64169	0.41372	0.
7	7	7.378	1.32944	1.28017	0.89284	0.61576	0.31814	(
8	8	7.364	1.33171	1.28907	0.91087	0.6598	0.43844	0.
9	9	7.286	1.25018	1.31967	0.90837	0.63938	0.42922	
10	10	7.284	1.33358	1.30923	0.93156	0.65124	0.35637	0.
11	11	7.278	1.22857	1.22393	0.91387	0.41319	0.07785	0.
12	12	7.226	0.95578	1.23788	0.86027	0.63376	0.10583	0.

Total rows: 781 of 781   Query complete 00:00:00.166

Evidence of load.

Reviewing the prediction made in our table loaded in the docker database, let's run the calculation to obtain R2:

```
WITH mean AS (
    SELECT AVG(happiness_score) AS mean_happiness
    FROM happiness
),
ssr AS (
    SELECT SUM(POWER(happiness_score - happiness_predicted, 2)) AS ssr
    FROM happiness
),
ssT AS (
    SELECT SUM(POWER(happiness_score - mean.mean_happiness, 2)) AS sst
    FROM happiness, mean
)
SELECT
    1 - (ssr.ssr / sst.sst) AS r_squared
FROM
    ssr, sst;
```



```
WITH mean AS (
    SELECT AVG(happiness_score) AS mean_happiness
    FROM happiness
),
ssr AS (
    SELECT SUM(POWER(happiness_score - happiness_predicted, 2)) AS ssr
    FROM happiness
),
ssT AS (
    SELECT SUM(POWER(happiness_score - mean.mean_happiness, 2)) AS sst
    FROM happiness, mean
)
SELECT
    1 - (ssr.ssr / sst.sst) AS r_squared
FROM
    ssr, sst;
```

	r_squared double precision
1	0.9703968925122535

Conclusions

After exploring the data of the 5 datasets, we obtain 17 columns, which are:

- **Country.**
- **Region.**
- **Country or Region.**

- **Happiness Ranking.**
- **Happiness Score.**
- **Standard Error.**
- **Lower Confidence Interval.**
- **Upper Confidence Interval.**
- **Whisker High.**
- **Whisker Low.**
- **Economy (GDP per Capita).**
- **Family**
- **Health (Life Expectancy).**
- **Freedom.**
- **Trust (government corruption).**
- **Generosity.**
- **Residual Dystopia.**

From which good information is obtained to find the Happiness Score. 5 of these columns are discarded because we only have data from a little more than 1 quarter of the complete dataset, which are: **Standard Error**, **Upper and Lower Confidence Interval**, **Whisker High and Low**.

The **Residual Dystopia** would bring us much the same as Happiness Rank, but this is part of the Happiness Score, so it will not be taken into account to predict the latter value.

There are a total of 781 records, but we will validate all of these assumptions at a later date.

After testing 5 models which were:

- Linear regression.
- Random forest regressor.
- Gradient boosting regressor.
- XGBoost regressor.
- Elastic Net.

Along with various splits of the data for training and testing, and the random state of the model, we found that using XGBoost Regressor along with a data split with random_state of 5432 (seeing a pattern in the .log file) and 5452 as the random_state of the model we get the result shown: **R2: 0.89995** with columns: ***'economy_per_capita'***, ***'family'***, ***'life_expectancy'***, ***'freedom'***, ***'government_corruption'***, ***'generosity'***, ***'year'*** and ***'country_region'***.

References

- Codigofacilito (2020, october 5). Mini Curso SQLAlchemy con Python 1 Crear Modelo. https://www.youtube.com/watch?v=XSAjQDM8ZS4
- Codigofacilito (2020, october 15). Mini Curso SQLAlchemy con Python 2 Persistir objetos. https://www.youtube.com/watch?v=110TZCWIOjo.

- Codigofacilito (2020, october 16). Mini Curso SQLAlchemy con Python 3 Consultas. https://youtu.be/T4MgHCiHwEY?si=twZbAkgUODQaWq9s.
- Scikit Learn. Scikit-learn library. https://scikit-learn.org/stable/index.html.