2022

MSIN0166 Individual Coursework

GITHUB LINK:

HTTPS://GITHUB.COM/UCEIS42/INDIVIDUAL.GIT

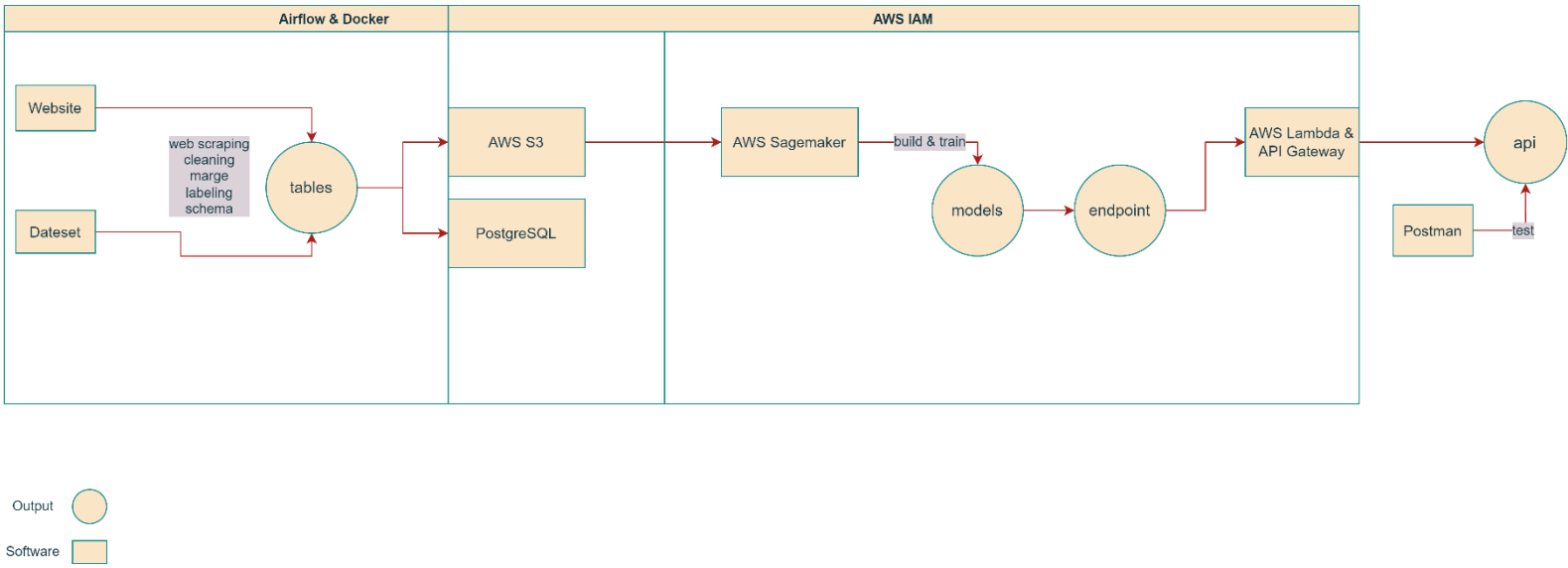
Word count 2507

Table of Contents

1. Introduction

This project aims to gather new datasets that contain company, geographic information as well as Co2 emission information. This project standalone contains a simple model, which provides simple prediction, and the trained endpoint can be accessed by API. My dissertation will build on this project with the addition of larger size company data and third-party satellite data.

The overall structure of this project is shown in the below pipeline:



The entire process can be divided into two parts. The first part is the data collecting part, which is automated with the help of airflow and docker. The data file is stored in the AWS S3 database, then it will be used as model input to train a machine learning model. The final output is an API that contains the endpoint of a trained model. It will then be used to make a prediction. The second part is fully AWS cloud-based, and it is managed by AWS IAM, which is AWS Identity and Access Management (IAM), a web service for securely controlling access to AWS resources. (Simplilearn, 2022)

2. Data Gathering

There are mainly two data sources. Company information data was captured from UK government website:
https://find-and-update.company-information.service.gov.uk/alphabetical-search
by python web scraping technique.
Unlike company data, emission data was from an existing excel file:
https://data.gov.uk/dataset/723c243d-2f1a-4d27-8b61-cdb93e5b10ff/uk-local-authority-and-regional-carbon-dioxide-emissions-national-statistics-2005-to-2019

Here are the data headers from the two sources:

Data retired from company information gov uk	Data retired from emissions dataset gov uk
1. Company name	1. Local authority
2. Company number	2. Region country
3. Status	3. Year
4. Link	4. Industry electricity
5. Address	5. Industry gas
6. Type	6. Industry other fuels
7. Incorporated time	7. Large industrial installations
8. Industry code	8. Agriculture
9. Post code	9. Industry total
10. Latitude	10. Operator
11. Longitude	11. Site
	12. Post code
	13. Reference
	14. Substance name

2.1 Company Data

The company data has consisted of two tables: the company list table and the company details table. The data retrieval process from Gov UK did not have too many restrictions. However, the size of the data was extremely large. For time and resource concerns, only the data for the first 100 pages were retrieved. Each page included 40 records of companies. In this case, there were 4000 records in the company list table. Based on the URLs stored in the company list, company details were captured and stored in the company detail table. Likewise, for time and resource concerns, there were only the first 700 company details data is retrieved.

2.2 Emission Data

Emission data were directly downloaded from the Gov UK dataset website. Within the dataset, two excel sheets were used in this project. One showed the data on Co2 emissions for companies in different years. Another one showed the co2 emission on an area scale.

3. Data Storage and Processing

After gathering the data from different sources, these data were stored in CSV and parquet formats since some S3 operation requires CSV format and Parquet has the advantage of storing large-size data. Parquet is efficient and performant in both storage and processing. If your dataset has many columns, and your use case typically involves working with a subset of those columns rather than entire records, Parquet is optimized for that kind of work. (Team et al., 2018)

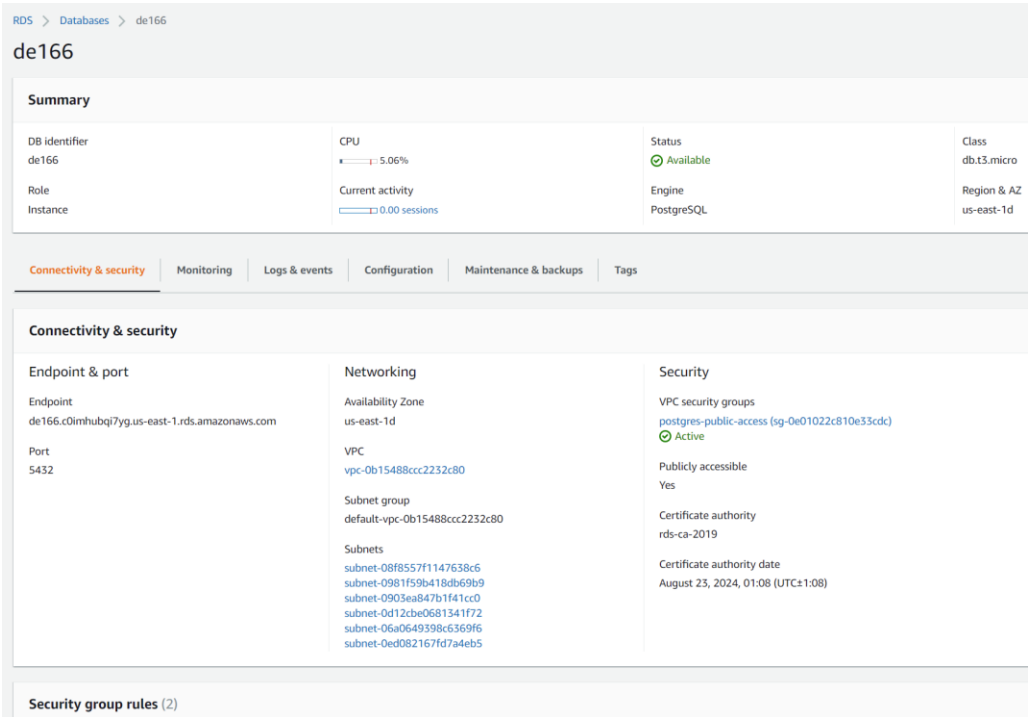
In the processing stage, data from different sources were cleaned, merge, and labeling by simple rules. These include basic type change, string replacements, format changing, filling or dropping necessary data, getting dummy variables for certain data, and so on. Most of the operations were performed by python and third-party packages like pandas.

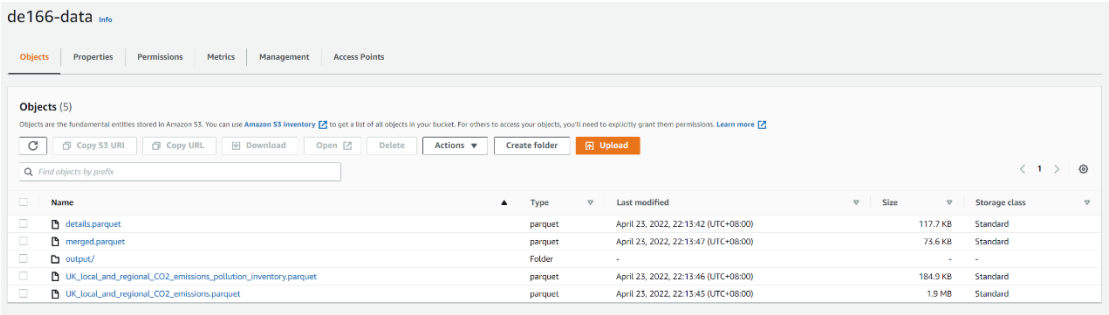
4. Database Selection

Two databases (PostgreSQL and AWS S3) were used in this project. As a relational database, PostgreSQL is highly reliable, stable, and secure. It is also easy to manage since tables have standard relations and schemas with each other.

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. (What is Amazon S3?, 2022) The reason for having AWS S3 is that many other AWS services require or perform better on S3's data. Meanwhile, AWS S3 has the advantage of scalable and secure. This project may require more data in the future so a scalable system is preferred.

The two figures below are the screenshots of PostgreSQL and AWS S3.





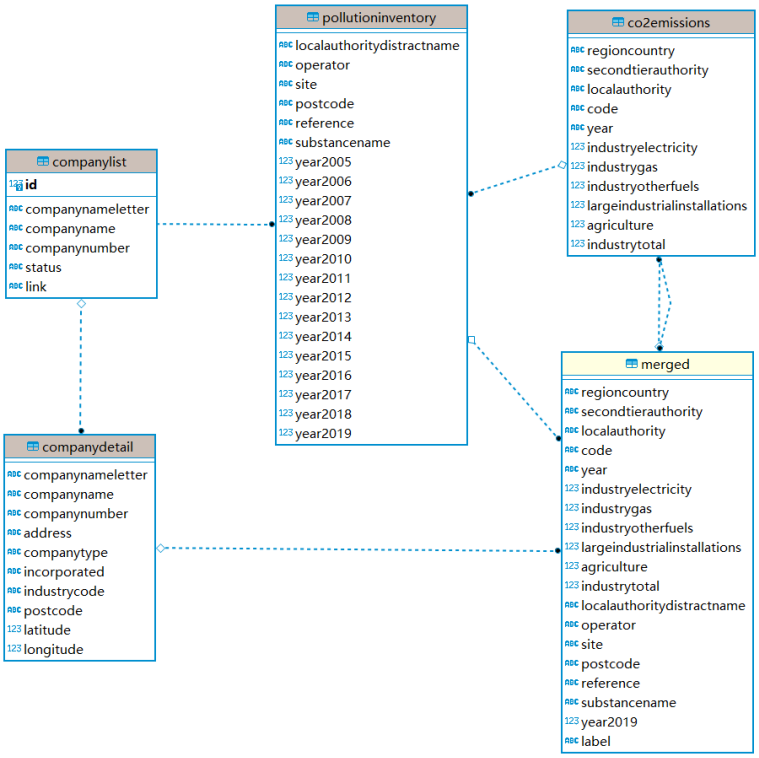
5. Data Size and Complexity

In this project, Pyspark and psycopg2 are used to achieve better processing speed for large datasets. PySpark is a Python API for Apache Spark, and psycopg2 is the database adapter for PostgreSQL in Python.

As a baseline, the maximin records number for one table is 4000, but in the future, it may largely increase.

The merged table was the input for the model, and it needed data from both company data and emission data. However, due to the limited record number in company detail, the merged table mainly consisted of data from emission data. This issue will be solved by increasing the record number of company details.

The picture below shows the schema for the database.



This figure below presents an overview of the merged table. However, to apply an xgboost model, this dataset needs further cleaning. The label column was moved to the first column, and necessary categorical data was turned into dummy variables.

Local Authority	Code	Year	Industry Electricity	Industry Gas	Industry 'Other Fuels'	Large Industrial Installations	Agriculture	Industry Total	Local Authority Distract Name	Operator	Site	Postcod
Darlington	E06000005	2019	17.786253	64.760412	23.930058	0.050426	5.952103	112.479252	Darlington	Ineos ChlorVinyls Ltd	Newton Aycliffe	DL5 6E
County Durham	E06000047	2019	110.906985	86.680235	130.746196	10.899522	45.994055	385.226992	County Durham	Dalkia Bio Energy Ltd	Chilton Biomass Plant	DL1 0S
County Durham	E06000047	2019	110.906985	86.680235	130.746196	10.899522	45.994055	385.226992	County Durham	Lucite International UK Ltd	Newton Aycliffe	DL5 6Y
Gateshead	E08000037	2019	45.752494	76.066494	35.248639	4.795098	1.542726	163.405451	Gateshead	Octagon Green Solutions Ltd	Blaydon Quarry	NE2 4S
Gateshead	E08000037	2019	45.752494	76.066494	35.248639	4.795098	1.542726	163.405451	Gateshead	Sita Environment Ltd	Path Head Quarry Landfill	NE2 4S
...	...	...	...	...	...	...	...	...	...	...	...	.
Mid and East Antrim	N09000008	2019	38.559634	56.107100	217.406073	11.847051	45.652023	369.571881	Mid and East Antrim	Kilroot Power Ltd	Kilroot	BT38 7L
Mid and East Antrim	N09000008	2019	38.559634	56.107100	217.406073	11.847051	45.652023	369.571881	Mid and East Antrim	Premier Power Ltd	Ballylumford	BT4 3R
Mid Ulster	N09000009	2019	75.808282	24.178805	349.697965	357.242559	92.253520	899.181131	Mid Ulster	Dale Farm Ltd	Dunmanbridge	BT8 9U
Mid Ulster	N09000009	2019	75.808282	24.178805	349.697965	357.242559	92.253520	899.181131	Mid Ulster	Lafarge Cement (Ireland) Ltd	Cookstown	BT8 9A
Mid Ulster	N09000009	2019	75.808282	24.178805	349.697965	357.242559	92.253520	899.181131	Mid Ulster	Moy Park Ltd	Dungannon	BT7 6L

6. Modeling

The modeling process includes model building and training, and the usage of the endpoint and API. In this project, XGBoost model was preferred due to the fact that it is more scalable then a regular decision tree.

The outcome of our prediction is shown below.

```
...
    Overall Classification Rate: 93.1%

    Predicted      Less than 50%  More than 50%
    Observed
    Less than 50%  93% (52)      7% (9)
    More than 50%   7% (4)      93% (123)
```

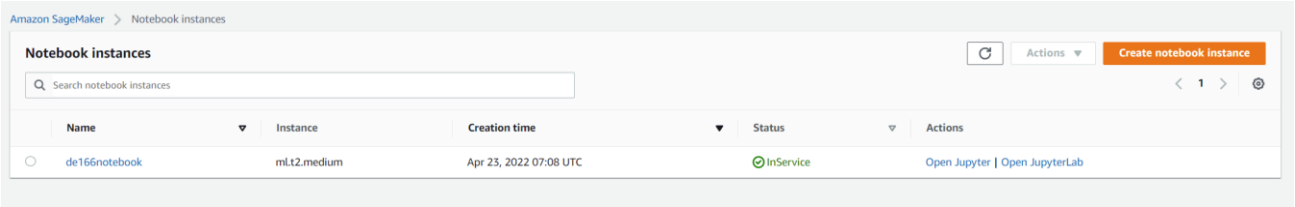
6.1 Model Building and Training

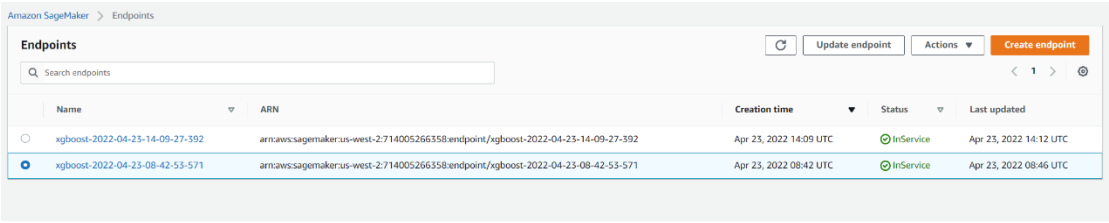
The entire model building and training process was built on AWS Sagemaker. It provides the tools to build, train and deploy machine learning (ML) models for predictive analytics applications. (Kranz and Carty, 2021) First, the notebook instance

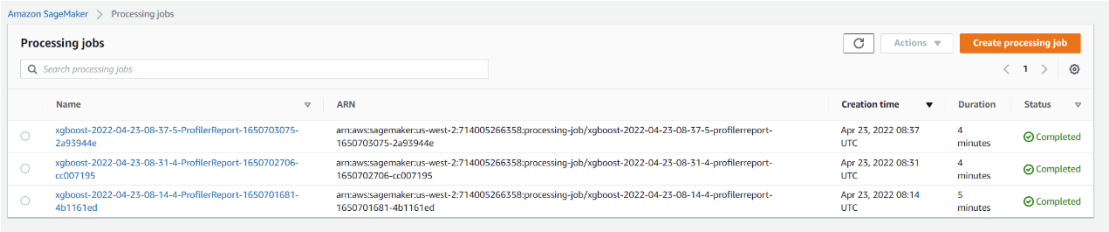
was trigged, then based on the script in the notebook, it performed a processing job and training job. Meanwhile, the XGBoost model is created. After the training job was done, the endpoint was created, and it was ready to predict.

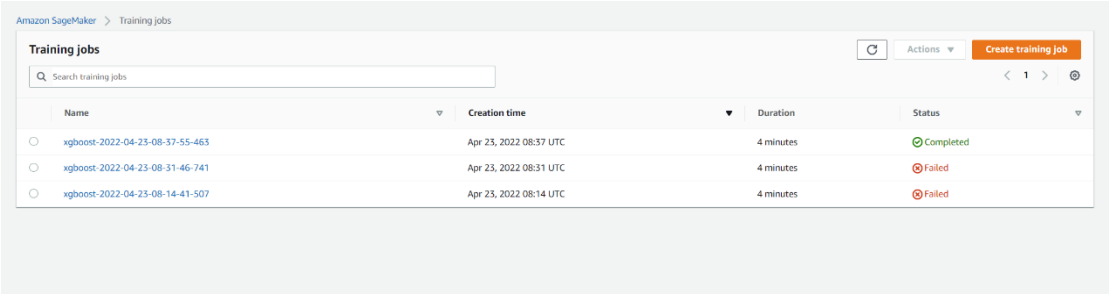
In this project, the model was mainly used to predict how likely a co2 emission is largely caused by industrial gas usage.

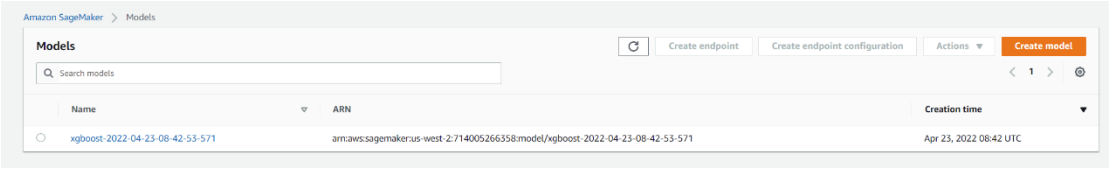
The following five Sagemaker screenshots present the webpages for notebook instances, endpoint, processing job list, training job lists, and models.











6.2 Endpoint and API testing

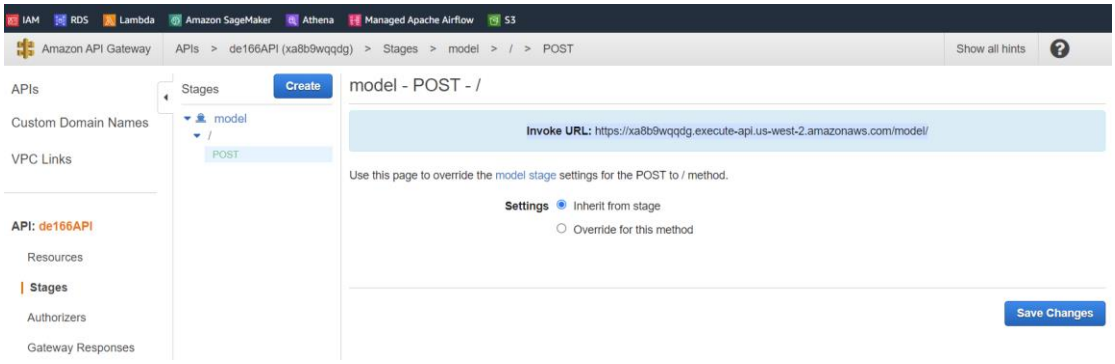
To call the endpoint created by Sagemaker, the lambda function is essential. In this case, a lambda function is a function that takes input data and invokes the trained model endpoint to predict. After that, it returns the prediction output. However, it is hard to trigger by the lambda function alone. For this reason, the REST API was built.
The REST API was built from AWS API Gateway, and it provides a link to call the prepared lambda function.

The testing stage was contributed by Postman. Postman is an API platform for building and using APIs. Postman simplifies each step of the API lifecycle and streamlines collaboration. (Postman) By sending a JSON script in a post request to the API, the lambda function was triggered. It then called the endpoint and inputted the data we wrote in JSON script. Finally, it returned the expected prediction outcome.

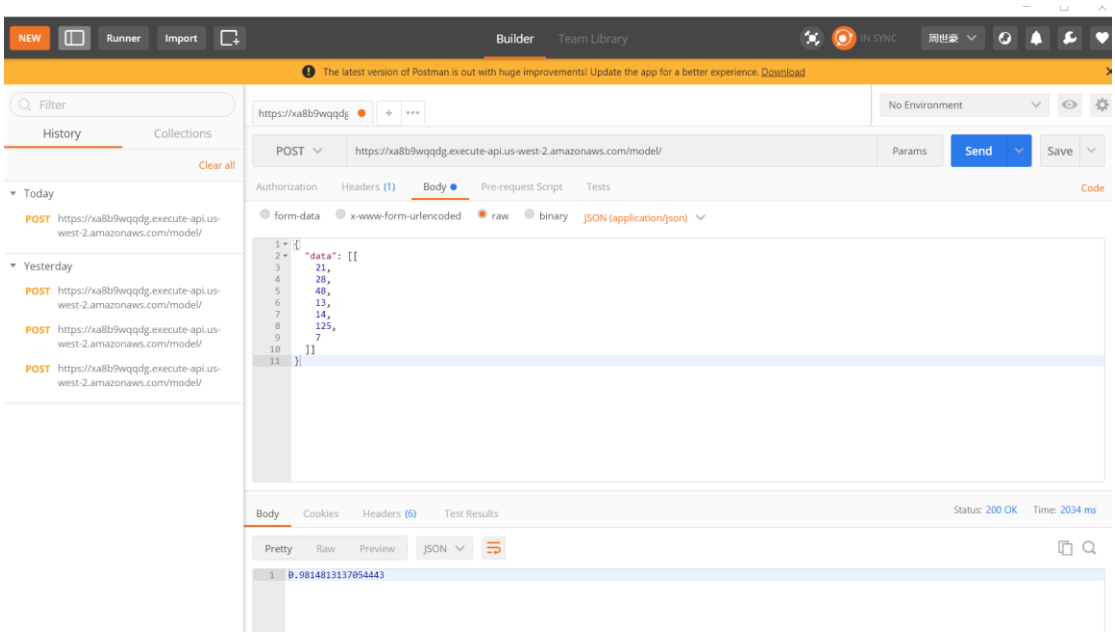
This is a screenshot of the script for the lambda function. The comment of code was added after the screenshot was taken.

```
import os
import io
import boto3
import json
import numpy as np

# grab environment variables
ENDPOINT_NAME = "xgboost-2022-04-23-14-09-27-392"
runtime= boto3.client('runtime.sagemaker')


def np2csv(arr):

    csv = io.BytesIO()

    np.savetxt(csv, arr, delimiter=",", fmt="%g")

    return csv.getvalue().decode().rstrip()

def lambda_handler(event, context):
    print("Received event: " + json.dumps(event, indent=2))

    data = json.loads(json.dumps(event))
    payload = data["data"]
    #payload = np2csv(payload)
    print(payload)

    response = runtime.invoke_endpoint(EndpointName=ENDPOINT_NAME,
                                       Body=np2csv(payload),ContentType="text/csv")
    print(response)
    result = json.loads(response['Body'].read().decode())
    print(result)

    return result
```

Below is a screenshot of the AWS API Gateway



This is the screenshot for using Postman to call the endpoint and get the prediction outcome.



7. Pipeline and Environment Setup

In real life, all the steps above will not only be used once. Instead, oftentimes, these processes need to be performed on a weekly basis or even a daily basis. In this case, integrating all the above steps and setting up the right environment once and for all is vital.

7.1 Airflow and Docker

As mentioned earlier in the introduction. The entire process can be viewed as two parts. The first part is from gathering data to storing it in the database. In practice, this part of the process is complicated and highly dependent on a stable environment. It is easy to make mistakes since it has so many detailed steps. In order to automatedly generate datasets by simply passing an URL, a combination of Airflow and Docker was utilized

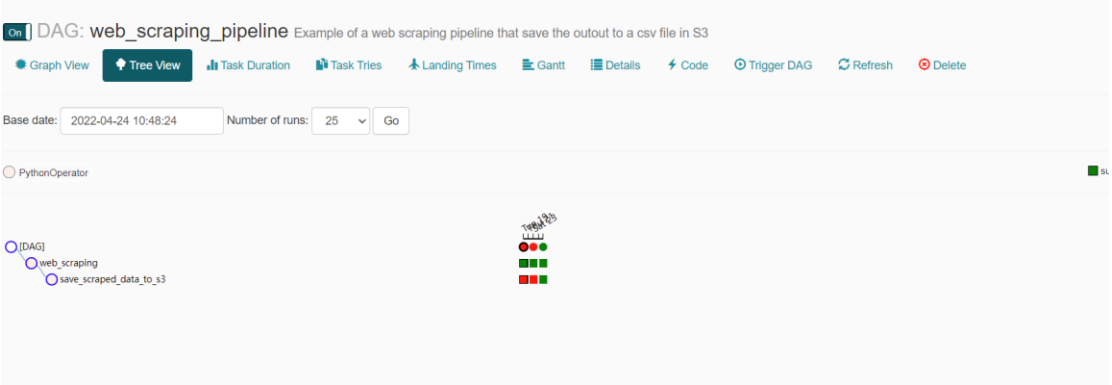
for this project.

Docker is a platform that designed to build programs or applications in a simple and fast way. By utilizing Docker, the entire workflow can be performed in a stable setup with all the necessary packages or software installed. To be more specific, Docker packs all the supporting packages and software into a container and runs the program in the container. It is similar to a virtual machine but it is more light weight than a virtual machine since it doesn't contain the entire operating system and it does not require any additional physical hardware.

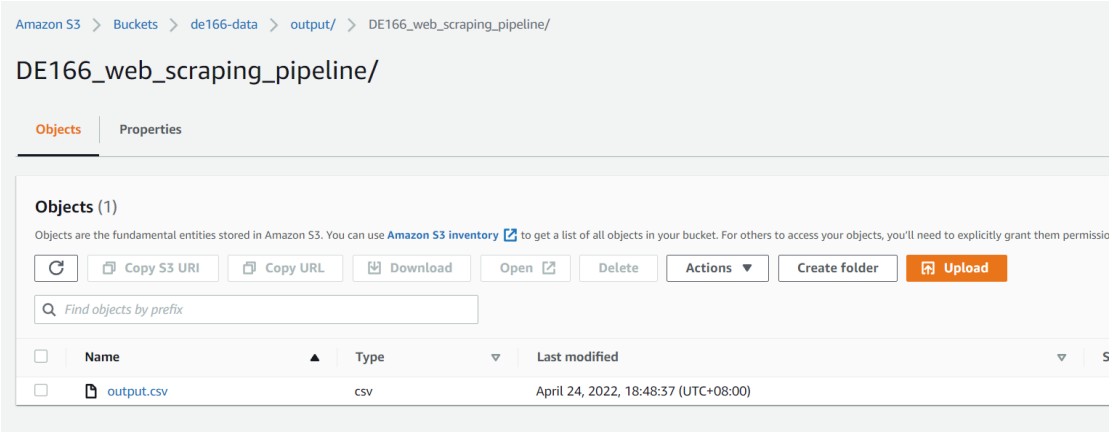
Apache Airflow is a workflow engine that will easily schedule and run complex data pipelines. It will make sure that each task of data pipeline will get executed in the correct order and each task gets the required resources. (lakshayarora, 2020)

The Airflow for this project is also built upon a docker environment. After writing the DAG file and setting up all the needed variables, scheduled jobs can be performed daily weekly, or in any other arrangement. In other words, the web scraping program will run automatedly, then it will pass the generated datasets to AWS S3 database.

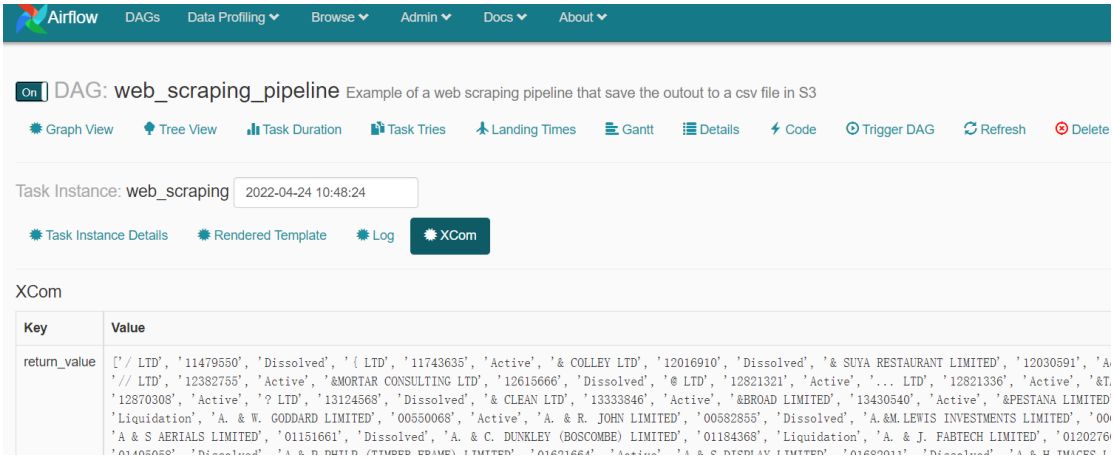
This screenshot shows the tree view of the pipeline.



Here is the screenshot for S3. The output is saved as CSV in the expected path.



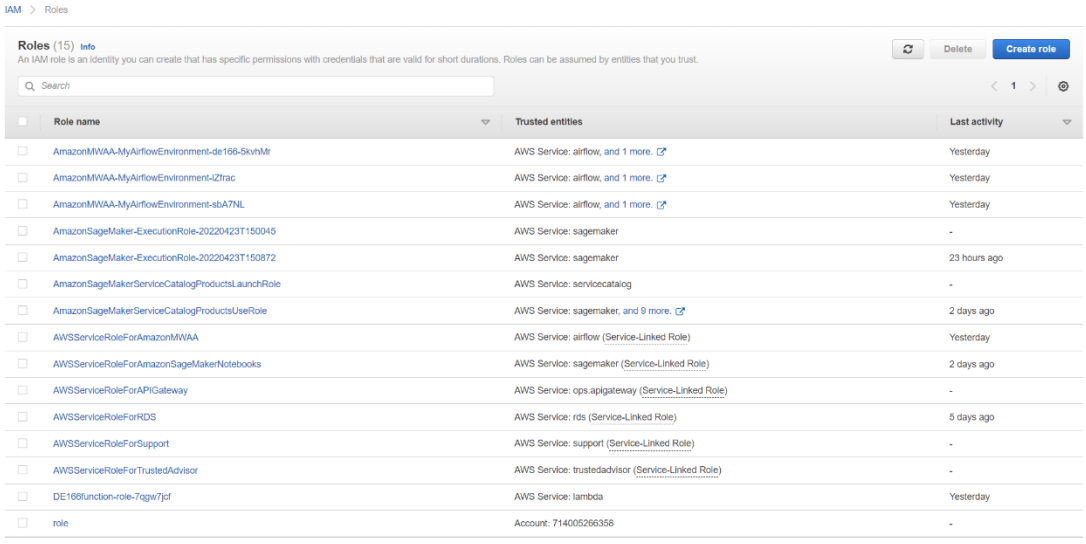
This is the XCom screenshot for the web scraping pipeline. By checking the XCom, it shows the content for the generated dataset.



7.2 AWS IAM

The second part of the project, which is from getting data from S3 to generating the API from the endpoint, was completely run on the AWS cloud server. This part was managed by the AWS Identity and Access Management (IAM) system, which is a security system that grants different levels of access to users or roles.

Below is the picture showing different roles in IAM.



8. Reproducibility and Version Control

As for reproducibility, since Airflow and Docker control the first part of this project, and AWS Sagemaker pipeline and lambda function cover the second part, the project is highly automated and reproducible. Each software mentioned is explained in their

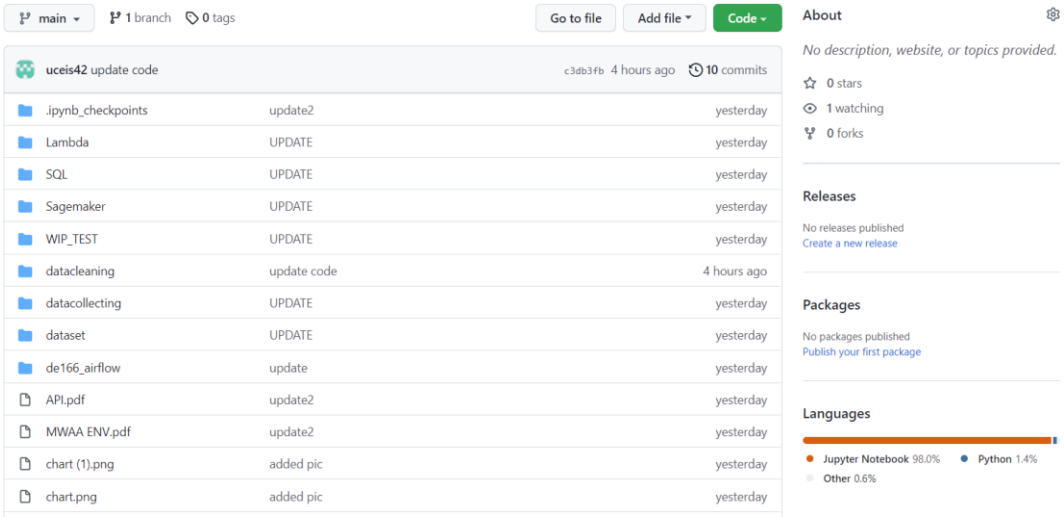
sections.

8.1 GitHub

Git was chosen for this project for version control. It would be even easier to use GitHub, which is a cloud-based Git repository hosting service that also provides a graphical UI for users. It is chosen mainly for three reasons. First, the file comparison function is easy to use. During the project, there are times that I might need to use some part of the code from the last version in the new version file. The code comparison helped save a lot of time.

Second, Git is a free and widely used software so using it for version control is quick and simple. Lastly, it supports various types of files. There are many different file types in this project and they can be managed well by Git.

This screenshot shows the overview of the repository of this project. It will be updated after the time that the screenshot was taken.



9. Project Management

Even though this is an individual project for myself only, it is still necessary to use a project management tool. Trello was selected for this project, but I only need some basic features from it. It is mainly used for me to set up deadlines and divide tasks. It also helps me to make outlines as I divide the whole project into many small tasks.

10. Data Lineage

As for the data lineage part, there is not any specific software for this project. However, the idea of tracking all the data is in the project. For example, PostgreSQL has a copy of the original datasets, and it can be viewed as metadata. Each step in our airflow pipeline or Sagemaker pipeline are well documented and there are logs for these steps as well.

11. Real World Application

This project aims to be a part of a larger project for the dissertation. The real-world application for it at this stage is to predict how likely a co2 emission is largely caused by industrial gas usage. It is expected that in the future, with the help of some satellite data, it could use different models to predict co2 emission at individual company level since the location information and company information is provided, and most of the pipeline and automation workflow is established.

12. Summary and Limitations

The limitation of this project is that due to the time and resources limitation, the data for company and company details is massively missing. Additionally, the address for companies in legislation may differ from the actual address. This could be solved by adding satellite data.

Moreover, to better support data lineage, Marquez can also be integrated into this project. Further actions for this project can be taken. These may include integrating Marquez for better data linkage support, adding satellite data and matching with the geographical data, and massively adding company detail information.

To sum up, there are mainly two parts to the project. The first part is integrated into Airflow, which collects data and stores them in a database. The second part is using data to train and build a model, and then using the model to predict based on additional data. The second part is fully built on the AWS cloud platform. The end products for this project includes a merged dataset, a trained model endpoint, and an API that call the endpoint then return the prediction output.

13. Reference

Department for Business, Energy & Industrial Strategy. “UK Local Authority and Regional Carbon Dioxide Emissions National Statistics: 2005 to 2019.” GOV.UK, GOV.UK, 5 Aug. 2021, https://www.gov.uk/government/statistics/uk-local-authority-and-regional-carbon-dioxide-emissions-national-statistics-2005-to-2019.

Kranz, Garry, and David Carty. “What Is Amazon Sagemaker?” SearchAWS, TechTarget, 31 Aug. 2021, https://www.techtarget.com/searchaws/definition/Amazon-SageMaker.

lakshayarora. “What Is Apache Airflow: Introduction to Apache Airflow.” Analytics Vidhya, 23 Nov. 2020, https://www.analyticsvidhya.com/blog/2020/11/getting-started-with-apache-airflow/.

Postman, https://www.postman.com/.

“Search the Register.” GOV.UK, https://find-and-update.company-information.service.gov.uk/.

Simplilearn. “AWS IAM Tutorial: Working, Components, and Features Explained [2022 Edition].” Simplilearn.com, Simplilearn, 4 Apr. 2022, https://www.simplilearn.com/tutorials/aws-tutorial/aws-iam.

Team, DataFlair, et al. “List the Advantage of Parquet File in Apache Spark.” DataFlair, 20 Sept. 2018, https://data-flair.training/forums/topic/list-the-advantage-of-parquet-file-in-apache-spark/.

“Welcome to GeoPy's Documentation!¶.” Welcome to GeoPy's Documentation! - GeoPy 2.2.0 Documentation, https://geopy.readthedocs.io/en/stable/#installation.

What Is Amazon S3? - Amazon Simple Storage Service. https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html.