

# **MSIN0166 Data Engineering Individual Assignment**

## **Project Title:**

Building a Data Pipeline with Machine Learning Models for  
Mobile Food Facilities in San Francisco, USA

**Word Count: 3572**

# **1. Introduction**

## **1.1 Project Description**

The purpose of this project is to build a data pipeline with Apache Airflow regarding all Mobile Food Establishments (E.g., Food Trucks, Ice cream vans etc.) in San Francisco, USA. Data regarding the Permits, Schedules and Yelp Information of all Mobile Food Facilities in San Francisco are extracted and stored for further analysis. An Extract, Transform and Load pipeline (ETL) is created with PySpark, PostgreSQL, Amazon Simple Storage Service (S3) and Amazon Relational Database Service (RDS), streamlined with Apache Airflow. Amazon SageMaker Autopilot is used for the final Machine Learning Tasks. Version Control is accomplished by using Git and Github.

## **1.2 Objective**

The extracted and stored data can be used for multiple machine-learning tasks, including running regression tasks to see the factors that affect establishments' Yelp rating, factors that increase customer engagement (E.g., more comments/reviews) and even create a dashboard to check whether any establishment is operating illegally with an expired or rejected permit. The permit and schedule data are extracted from DataSF, the official Open Data portal created by the City and County of San Francisco, containing different datasets including crimes, housing, Covid-19 data etc, and are updated periodically (DataSF, 2022).

The chosen datasets – Mobile Food Facilities Permit and Schedule datasets, are updated weekly by the City and County of San Francisco. Considering the constant updates, Apache Airflow is set to run weekly to extract the most updated data from DataSF. Afterwards, based on the facilities' name and location from the updated Permit and Schedule dataset, another extraction is done on Yelp Fusion API, an online directory that publishes crowd sourced reviews about local businesses, in order to obtain relevant data from the website (Yelp, 2022). Currently a limit has been set to only return the top 50 results, with no offset set on the parameter, meaning no subset of record is retrieved (Algolia, 2022). This was done deliberately as there are daily limit constraints on Yelp Fusion API calls. Nevertheless, a smaller dataset enables a better focus on building the foundation of ETL, which is purpose of this project. If developer access were granted by Yelp Fusion API, a much higher rate limit on API extraction would be possible. The database on Yelp information would be able to scale substantially, to provide a more all rounded dataset for ML model training.

It is noteworthy to mention that this project focuses on creating a data architecture by using multiple different data engineering infrastructures and tools available on the market. Machine Learning models have been deployed; however, the result might not be of highest accuracy as model accuracy is out of the project scope.

# **2. Architecture Overview**

## **2.1 Apache Airflow**

Apache Airflow has been chosen to be the backbone of this Data Architecture. It allows Data Engineers to programmatically write, schedule and monitor ETL workflows. It was designed for cloud and on-premises data integration – the ETL workflow is written in a Python script, taking data from multiple sources, transforming them into meaningful information, and finally loading them into data warehouses in a structured manner (Zabor, 2022). The following tools have been used in this particular Airflow infrastructure.

### 2.1.1 Amazon Simple Storage Service (S3)

S3 is a highly scalable, secured, and high performing object storage service offered by Amazon Web Services (AWS). It can be used to store data in any sizes, structures and formats; serving customers of different sizes and needs.

It is believed to be beneficial to store data on a cloud-based storage system instead of a local machine. Taking Amazon S3 as an example, such storage service provider offers different storage classes and storage management, meaning frequently visited data can be stored as Standard Class, while infrequently accessed/archived data can be stored in lower cost class for cost effectiveness. Besides, the Access Management features allows auditing and access management to different data buckets and objects; so that restricted access can be granted to specific personnel accordingly, and unauthorised access can be prevented (Amazon Web Services, 2022). These features and functions can be difficult to replicate by a local machine/server, hence cloud-based storage systems like Amazon S3 are good options when building a Data Architecture.

In this project, Amazon S3 is used in multiple occasions within the ETL. It stores the raw data extracted from APIs, the transformed tables in csv, parquets files, Amazon SageMaker Models, and SQL queries results. It has been considered to use IAM to grant access to the teaching team, however their Amazon account number were unavailable, hence S3 Block Public Access feature is used to prevent unauthorized access with malicious intentions instead. A screenshot of the Amazon S3 bucket has been attached in the following.

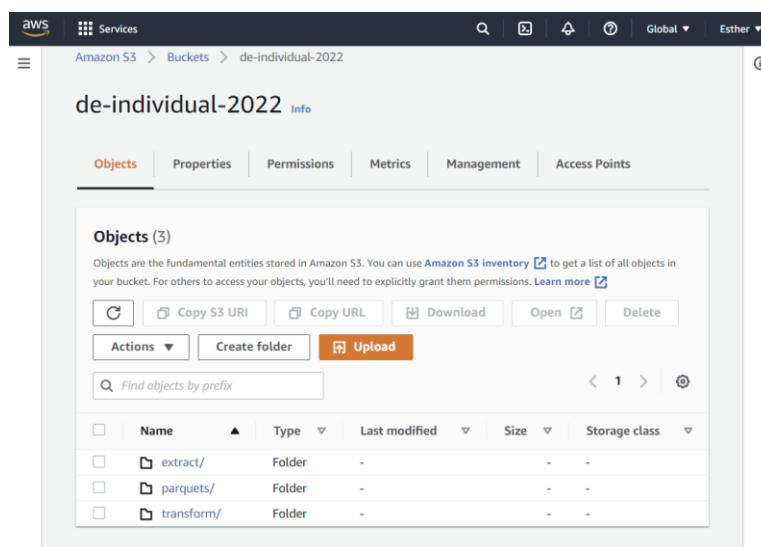


Figure 1: Amazon S3 Bucket Overview

### 2.1.2 Amazon Relational Database Service (RDS)

RDS is a managed SQL database service provided by AWS that facilitate the implementation and maintenance of relational database (Lutkevich, 2021). It assists its user to set up/migrate, store, operate and scale databases in the cloud. It is beneficial to use RDS, as it is incredibly simple to use. No tricky data formatting and processing is needed before storage, and RDS is able to consistently execute storage operation as instructed by users and have the ability to roll back the operation if issues arise during new deployment. These features are highly desirable for industry use.

In this scenario, PostgreSQL has been used to transformed data into a relational database, based on the created schema; and storing it with RDS.

### 2.1.3 Amazon SageMaker Autopilot

Amazon SageMaker Autopilot is a service provided by AWS that automatically builds, trains and fine-tune Machine Learning Models based on the input data. It automatically pre-processes the dataset (E.g., filling missing item, extracting from non-numeric columns etc.) and explore the best Machine Learning Models based on the type of prediction required (Amazon Web Services, 2022). This allows users with little Machine Learning knowledge to build and select the best model.

In this project, Amazon SageMaker Autopilot is used to create and deployed Machine Learning Models, based on the extracted data that are stored in Amazon S3.

## 2.2 Faculty

Faculty is the cloud platform that was used for all code writing of this project. The main reason for using Faculty instead of a local machine, or other cloud-based platform (E.g., Amazon EC2, Google Colab) is that this project requires a minimum of 4 cores CPU server. Free Amazon EC2 or Free Google Colab only offers a maximum of 2 CPU Cores; whilst Faculty is able to supply with sufficient specification.

## 2.3 PySpark

Apache Spark is a data processing framework with the ability to perform data processing on a large dataset in a time efficient manner. Due to its performance speed and ease of use, it is favourable to set up Spark in order to make future data processing more efficient when the database is scaled up (databricks, 2022). PySpark is the Python API for Apache Spark, and this was chosen due to the familiarity to Python Programming Language.

## 3. Data Preparation and Storage

### 3.1 Data Source

There are two major data sources for this project – DataSF API and Yelp Fusion API.

As mentioned previously, DataSF is the official Open Data portal created by the City and County of San Francisco. Within the portal, there are 538 published datasets, within which 2 have been chosen for the purpose of this report – 1) Mobile Food Facility Permit dataset, 2) Mobile Food Schedule dataset.

Mobile Food Facility Permit datasets includes the name, location, type of food sold and permit status of all mobile food vendors in San Francisco (DataSF, 2022); whilst Mobile Food Schedule dataset includes operating day of week, operating hours, and location of mobile food vendors with a valid permit (DataSF, 2022). These datasets are extracted with Socrata Open Data API, which does not require an application token. There are no official throttling limits and users are technically able to get unlimited requests in normal circumstance. As the Airflow DAG would only run once a week, and the datasets extracted are around ~2000 rows each, hence data extraction from DataSF should not be an issue.

The second data source is Yelp Fusion API. Yelp is an online directory of business around the world, with reviews and rating submitted by public. The idea is to combine official permit data with unofficial public data in order to generate one big dataset that containing both for further analysis. In order to extract the most relevant data, a list of Vender name and location were extracted from Mobile Food Schedule dataset, and that was used as search parameter for Yelp Fusion API. A Yelp account needs to be created in order to generate an API key for data extraction, and there is a limit of 5000 API calls

a day (Yelp Fusion, 2022). Due to the API daily limit, no offset was added onto the parameter, resulting the API only returning the top 50 results. This dataset can be expanded rapidly if Yelp Fusion API allows a higher number of daily requests, it will help to populate a higher percentage of the Yelp data in the combined dataframe and make the future analysis more representable.

## 3.2 Schema Design

As previously mentioned, relational database is used for this data architecture. All data are stores in tables with multiple rows and columns. Each table contains a unique identifier as Primary Key, which can be added to another table as Foreign Key. The relationship between tables within a relational database can be identified using the aforementioned Primary and Foreign keys. This linkage allows easy data storage, retrieval and selection, based on the known data which uniquely identify the requested data (Davidson, 2020).

The schema for this relational database will be shown in the following:



Figure 2: Schema Design (Link: <https://dbdiagram.io/d/626ae8f595e7f23c61931e32>)

This database contains a total of 11 tables, with <details> table being the primary table. As the same vendor can have multiple permits to operate in different locations, unique permit number is used as Primary Key; and the table contains applicant (vendor name) and facilitytype. <applicant\_details> table is created with objectid (permit application unique ID) as Primary Key, due to the fact multiple applications can be made under the same permit, which is the Foreign Key (FK). Afterwards, <permit\_application\_details> is created using objectid as Primary Key, and the table contains the details of permit application under certain objectid.

On the other side, <food\_type> table contains data on types of food sold, with permit as Primary Key. <num\_permit\_per\_business> table uses applicant as Primary Key, showing all permit (FK) number under the same vendor, and a newly created no\_permit\_owned\_per\_business, which counts the number of permit under one vendor. <yelp\_info> table uses Yelp\_id as Primary key, containing data extracted from Yelp Fusion API. Name (Vendor Name) is used as Foreign Key to connect to <num\_permit\_per\_business>'s applicant.

<permitted\_biz\_details> table uses schedule\_id as Primary Key, with permit as Foreign Key, as under the same permit, mobile food vendors can operate on multiple days of week, creating multiple schedule\_id. The final four tables <location>, <operating\_day>, <permit\_motification\_details> and <operation\_time> all use schedule\_id as Primary Key, and linked to <permitted\_biz\_details> by schedule\_id.

Creating tables like <applicant\_details>, <permitted\_biz\_details> and <num\_permit\_per\_business> might seem to be repetitive, however they are essential in this relational database setting, in order to create unique Primary Keys and Foreign Keys.

### 3.3 ETL Pipeline and Airflow DAG

As the data being explored are regarding Mobile Food Facilities in San Francisco, the focus would be the establishments with such permits. DataSF being the official database of City and County of San Francisco, it would be a very accurate and reliable data source. Yelp, on the other hand, are unofficial public-contributed data. Combining both could potentially provide much more insight into the mobile food industry in San Francisco.

The ETL pipeline contains 1 DAG and 4 components, they are written in one single python file.

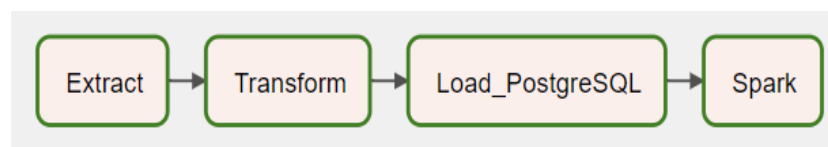


Figure 3: ETL Pipeline

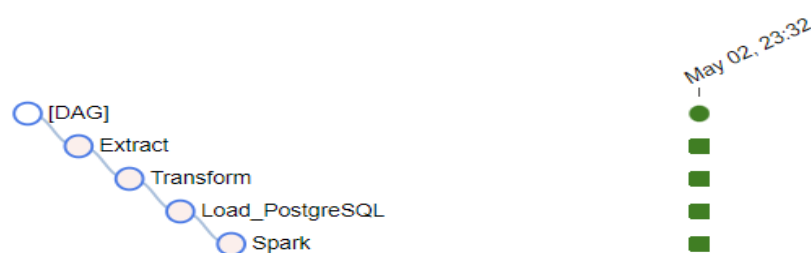


Figure 4: Airflow DAG

### 3.3.1 Extract

Step 1: Using Socrata Open Data API, Mobile Food Permit data and Mobile Food Schedule data are extracted. These data include establishments name, operating schedule, permit details etc.

Step 2: Based on the establishment names and locations from Mobile Food Schedule, codes are written to use those information as search parameter, extracting Yelp Data via Yelp Fusion API. The data extracted from Yelp includes Name, number of reviews, rating, address and business phone number etc. Again, the data is saved as CSV into Amazon S3 bucket.

As DataSF update the two data sources on a weekly basis, new data will be extracted weekly and saved as csv in Amazon S3 bucket.

### 3.3.2 Transform

An aforementioned schema is defined in order to facilitate data transformation.

Step 1: Basic data cleaning and transformation is done on the raw data, in order to remove unwanted columns from Mobile Food Facility Permit dataframe. A schedule ID is also added onto Mobile Food Schedule dataframe in order to identify each unique schedule.

Step 2: Transforms and organises the data into 11 dataframes according to the schema design.

Step 3: Exports the data into 11 csv file:

'applicant\_details\_table.csv', 'details\_table.csv', 'food\_type\_table.csv', 'location\_table.csv',  
'operating\_day\_table.csv', 'operation\_time\_table.csv', 'yelp\_info\_table.csv', '  
permit\_application\_details\_table.csv', 'permit\_motification\_details\_table.csv',  
'premitted\_biz\_details\_table.csv', 'num\_permit\_per\_business\_table.csv'.

These csv are saved in Amazon S3 bucket, under the folder name *<transform>*.

### 3.3.3 Load PostgreSQL

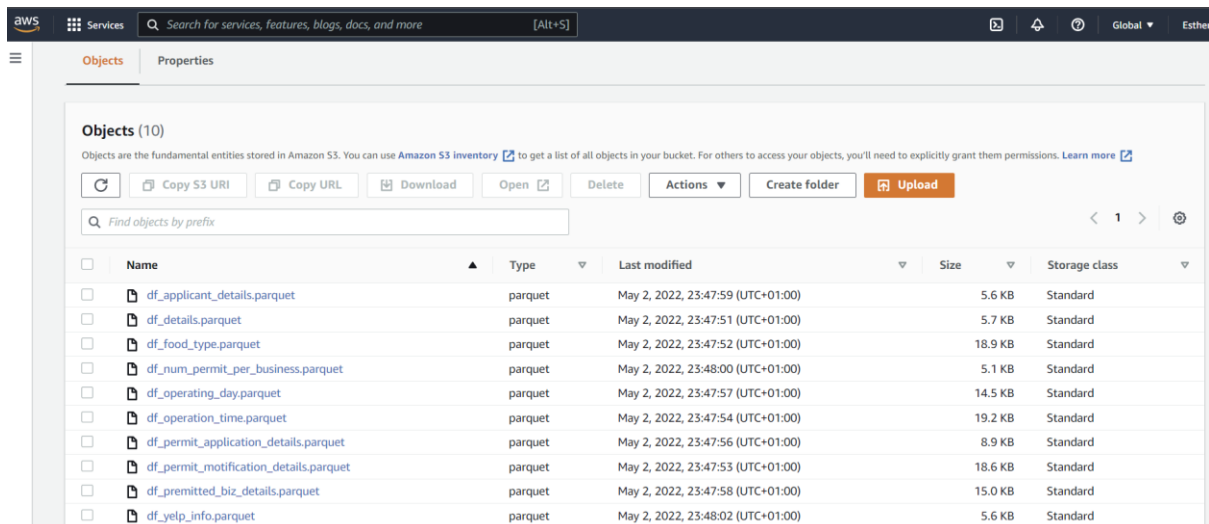
Step 1: Create 11 database tables based on schema with specific Primary and Foreign Keys:

'details', 'premitted\_biz\_details', 'applicant\_details', 'location', 'food\_type',  
'permit\_motification\_details', 'operation\_time', 'permit\_application\_details', 'operating\_day',  
'num\_permit\_per\_business', 'yelp\_info'

Step 2: Convert all dataframes into tuples, using psycopg2 to feed each row of data into respective schema one at a time into PostgreSQL on Amazon RDS.

### 3.3.4 Load for Spark

Basic queries have been executed to better understand the new relational database. PySpark is used to perform SQL queries, and the tables are also saved in Amazon S3 as Parquet format as requested. Parquet files are found to perform much better at complicated data processing and analytics, as it allows efficient retrieval of data columns (Butterfly Thoughts, 2021).



<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	df_applicant_details.parquet	parquet	May 2, 2022, 23:47:59 (UTC+01:00)	5.6 KB	Standard
<input type="checkbox"/>	df_details.parquet	parquet	May 2, 2022, 23:47:51 (UTC+01:00)	5.7 KB	Standard
<input type="checkbox"/>	df_food_type.parquet	parquet	May 2, 2022, 23:47:52 (UTC+01:00)	18.9 KB	Standard
<input type="checkbox"/>	df_num_permit_per_business.parquet	parquet	May 2, 2022, 23:48:00 (UTC+01:00)	5.1 KB	Standard
<input type="checkbox"/>	df_operating_day.parquet	parquet	May 2, 2022, 23:47:57 (UTC+01:00)	14.5 KB	Standard
<input type="checkbox"/>	df_operation_time.parquet	parquet	May 2, 2022, 23:47:54 (UTC+01:00)	19.2 KB	Standard
<input type="checkbox"/>	df_permit_application_details.parquet	parquet	May 2, 2022, 23:47:56 (UTC+01:00)	8.9 KB	Standard
<input type="checkbox"/>	df_permit_motification_details.parquet	parquet	May 2, 2022, 23:47:53 (UTC+01:00)	18.6 KB	Standard
<input type="checkbox"/>	df_permitted_biz_details.parquet	parquet	May 2, 2022, 23:47:58 (UTC+01:00)	15.0 KB	Standard
<input type="checkbox"/>	df_yelp_info.parquet	parquet	May 2, 2022, 23:48:02 (UTC+01:00)	5.6 KB	Standard

Figure 5: Parquet Format files in Amazon S3

### 3.4 Git and Github

Github is online code hosting platform that facilitate version controls and collaboration. It enables multiple developers to work on the same project and keep track of file versions that were updated by different contributors. The version control functions also allow version reversion when necessary.

As this is an individual project, the collaboration function of Github were not fully utilised. Github was cloned to Faculty, where all codes were experimented in Jupyter Notebook, and finally written in Python script. The full DAG Python Script has been pushed to Github for version control after every changes made. The link to the private repository for this project is attached below, with the following Module Teaching Staff's Github account added as collaborators.

Link to private Github Repository



## 4. SQL Queries

High level SQL Queries have been made to test the performance of relational database.

### Top 5 Mobile Food Establishments with the highest share in approved schedule

Findings: The Top 5 establishments with the highest counts of Approved Schedules are: Brazuca Grill, Liang Bai Ping, Park's Catering, Singh Brother Ice Cream and Sohomei, LLC.

Top 5 Mobile Food Establishments with the highest share in approved schedule

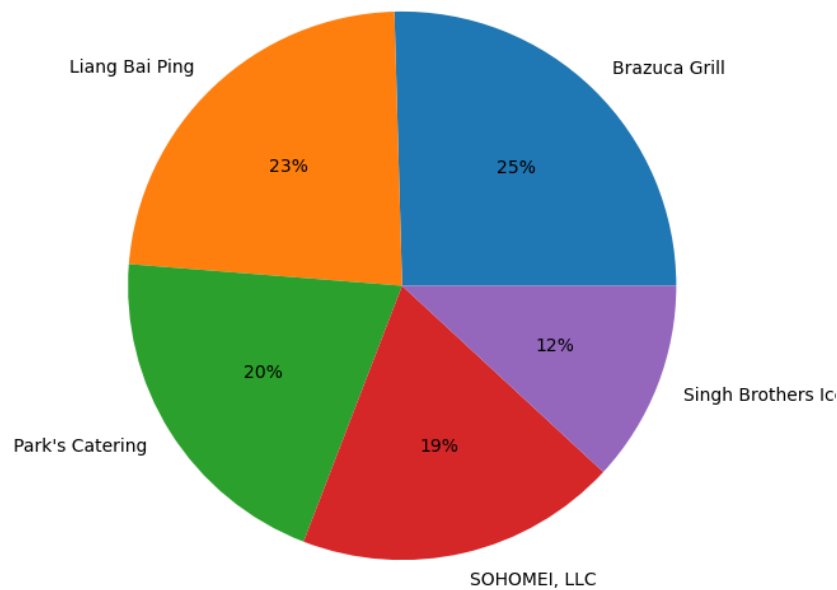


Figure 6: Top 5 Mobile Food Establishments with the highest share in approved schedule

## 5. Amazon SageMaker Autopilot

Amazon SageMaker is a fully managed machine learning services for building and training machine learning model quickly and easily (Amazon Web Services, 2022).

The idea behind incorporating Yelp data into this relational database, is to obtain reviews, rating and price of the Mobile Food Facilities in San Francisco. With such data, machine learning model can be deployed to find out the relationship between more branches (vendor with the same name and multiple permits), popularity (with higher number of reviews), and ratings.

However, due to API limits and technical difficulties that are detailed later, the extracted dataset is too small for Machine Learning Model Training. In order to demonstrate how Amazon SageMaker can be used for Machine Learning Model deployment, the original Mobile Food Facility Schedule dataset has been used instead. The targeted column is set to be "status", referring whether the permit application is approved, rejected, pending or expired. Amazon SageMaker would automatically process and do feature engineering on the data. After exploring hundreds of feasible models, the best one would be fine-tuned and deployed automatically.

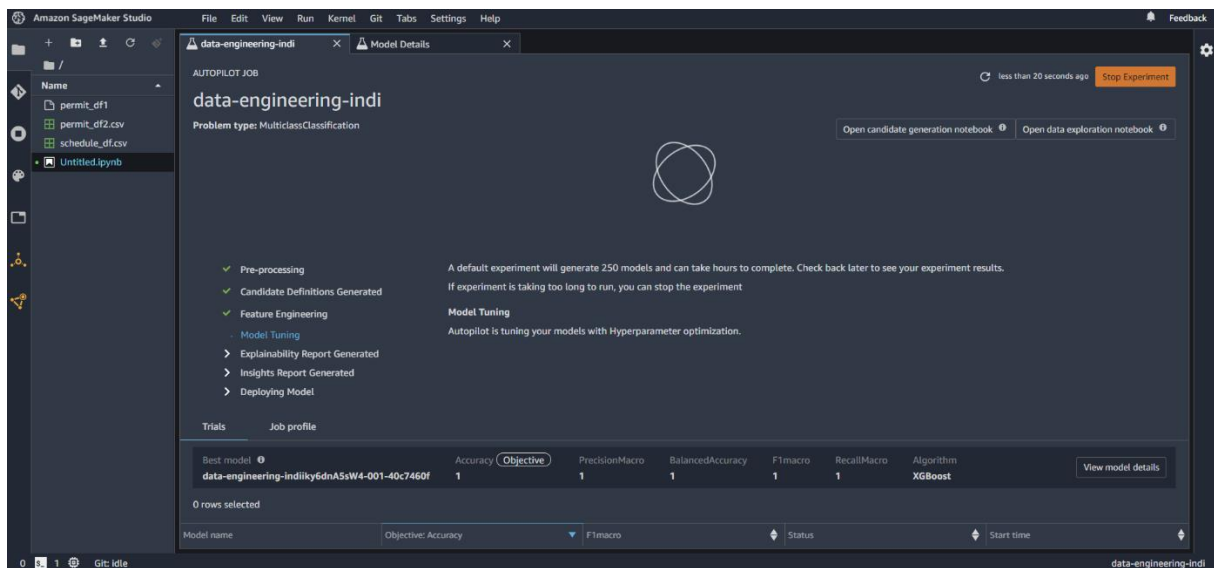


Figure 7: Amazon SageMaker Autopilot Model Training

In this circumstance, the chosen dataset is not suitable for Machine Learning prediction tasks, as Permit Application details (E.g. form submission date etc.) may not be the most relevant to whether the permit is approved or not. Hence after running SageMaker for an hour, the experiment has been stopped, due to the fact that training and hosting are billed by minutes of usage, and it was not worth it to spend resources on training a futile model. This is more to demonstrate the knowledge of SageMaker, and how it could have been incorporated into this project.

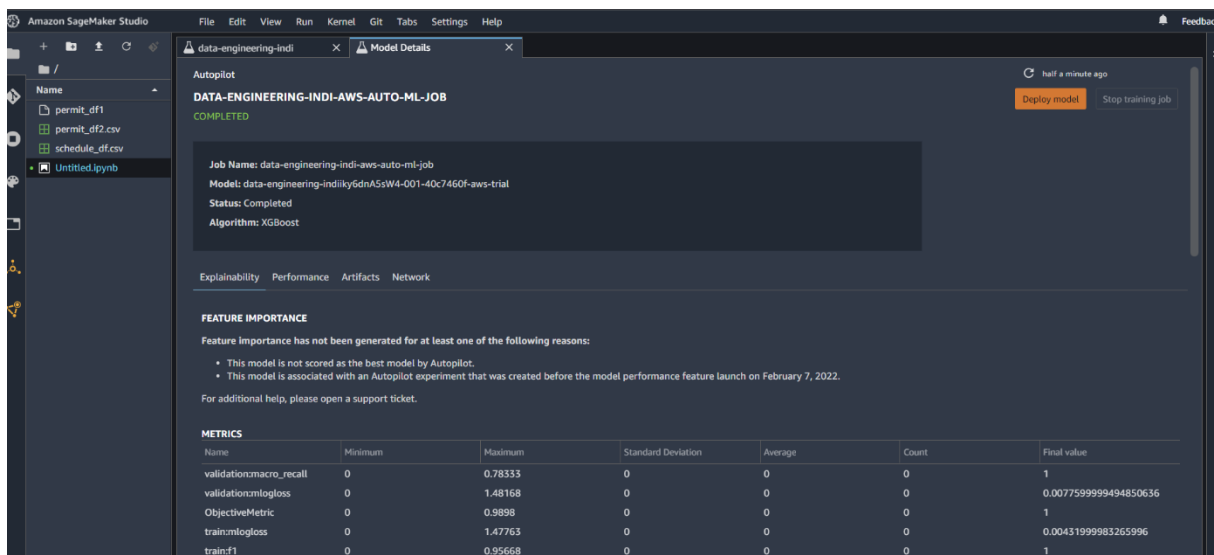


Figure 8: Amazon SageMaker Autopilot Model Details

## **6. Conclusion, Limitation and Outlook**

To conclude, this project has built a data pipeline regarding all Mobile Food Establishments in San Francisco. By using Apache Airflow as the backbone of the data architecture, it enables the creation, scheduling and monitoring of ETL workflow in one place. Apache Airflow extracts data from DataSF and Yelp Fusion API on a weekly basis and saves the unprocessed database into Amazon S3. It is cleaned and transformed into multiple csv tables, then fed into PostgreSQL on Amazon RDS according to the designed relational schema.

It is worth noting that, for industry best practice, Machine Learning Models training, fine-tuning and deployment should also be included in the same Airflow DAG. However due to technical difficulties and time constraints, Machine Learning Model Training is done separately from the DAG. Amazon SageMaker Autopilot is used to for model training and selection, based on the raw data that was extracted and saved in Amazon S3 via Apache Airflow. Going forward, Amazon SageMaker should be incorporated in the DAG, in order to streamline the whole ETL and Machine Learning Process.

Additionally, it is noted that due to the Yelp Fusion API daily limit constraint, only 50 search results were returned for every extraction. The extracted Yelp dataset is then compared to the establishment name and location from Mobile Food Schedule dataset, and only the matched data would remain in the relational database. Currently, only 2 rows of data can be matched between the two datasets. Latitude and Longitude have also been used to try merging the two datasets but in vain. It is noted that the vendors' name appears on DataSF database, may not be the trading name that appears on Yelp, which could be one of the reason for the lack of matched data. However, even with the above issues, it is believed that the main constraint is still the limit search results. If a higher API limit can be obtained, it can surely help to populate the <yelp\_info> table in the relational database, and make the data more representable and insightful.

Going forward, after the aforementioned Yelp database issue is solved, it would be worth it to scrap Yelp Reviews by using Beautiful Soup, a python package that is used for web scraping. The reviews can be stored in the relational database, using Yelp\_id as Primary Key, referencing to <yelp\_info> table. The scraped review can be used for sentimental analysis, fake comment detection etc.

## References

Algolia, 2022. *API Reference - offset*. [Online]

Available at: <https://www.algolia.com/doc/api-reference/api-parameters/offset/?client=python>  
[Accessed 1 April 2022].

Amazon Web Services, 2022. *Amazon SageMaker Autopilot - Automatically create machine learning models with full visibility*. [Online]

Available at: <https://aws.amazon.com/sagemaker/autopilot/>  
[Accessed 29 April 2022].

Amazon Web Services, 2022. *What is Amazon S3?*. [Online]

Available at: <https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html>  
[Accessed 1 April 2022].

Amazon Web Services, 2022. *What Is Amazon SageMaker?*. [Online]

Available at: <https://docs.aws.amazon.com/sagemaker/latest/dg/whatis.html>  
[Accessed 1 May 2022].

Butterfly Thoughts, 2021. *Probably The Best Alternative to CSV Storage: Parquet Data*. [Online]

Available at: <https://geekflare.com/parquet-csv-data-storage/>  
[Accessed 1 April 2022].

databricks, 2022. *Apache Spark is a lightning-fast unified analytics engine for big data and machine learning..* [Online]

Available at: <https://databricks.com/spark/about>  
[Accessed 30 April 2022].

DataSF, 2022. *About SF OpenData*. [Online]

Available at: <https://data.sfgov.org/about>  
[Accessed 15 April 2022].

DataSF, 2022. *Mobile Food Facility Permit*. [Online]

Available at: <https://data.sfgov.org/Economy-and-Community/Mobile-Food-Facility-Permit/rqzj-sfat>  
[Accessed 16 April 2022].

DataSF, 2022. *Mobile Food Schedule*. [Online]

Available at: <https://data.sfgov.org/Economy-and-Community/Mobile-Food-Schedule/jjew-r69b>  
[Accessed 15 April 2022].

Davidson, L., 2020. The Fundamentals. In: *Pro SQL Server Relational Database Design and Implementation: Best Practices for Scalability and Performance*. Antioch, Tennessee: Apress.

Li, W. S., Yang, C., Tong, E. & Momen, S., 2022. *Building a Data Pipeline for League of Legends Challenger-level Pro-players in the EU Region*, London: s.n.

Lutkevich, B., 2021. *Amazon RDS (Relational Database Service)*. [Online]

Available at: <https://www.techtarget.com/searchaws/definition/Amazon-Relational-Database-Service-RDS>  
[Accessed 20 April 2022].

Meloni, J. C., 2003. The Importance of Good Database Design. In: K. Hansing, ed. *Sams Teach Yourself PHP, MySQL® and Apache All in One*. California: Sams Publishing.

Noble Desktop, 2018. *What Is Git & Why Should You Use It?*. [Online]  
Available at: <https://www.nobledesktop.com/blog/what-is-git-and-why-should-you-use-it#:~:text=Git%20is%20the%20most%20commonly,be%20merged%20into%20one%20source.>  
[Accessed 27 March 2022].

snowflake, 2022. *WHAT IS ETL?*. [Online]  
Available at: <https://www.snowflake.com/guides/what-etl>  
[Accessed 21 March 2022].

Yelp Fusion, 2022. *Yelp Fusion*. [Online]  
Available at: [https://www.yelp.com/developers/v3/manage\\_app](https://www.yelp.com/developers/v3/manage_app)  
[Accessed 19 April 2022].

Yelp, 2022. *Yelp*. [Online]  
Available at: <https://www.yelp.co.uk/london>  
[Accessed 16 April 2022].

Zabor, K., 2022. *How to make the most of Airflow ETL?*. [Online]  
Available at: <https://www.n-ix.com/make-the-most-of-airflow-etl/#:~:text=Apache%20Airflow%20for%20ETL%20offers,productivity%2C%20and%20cost%2Doptimization.>  
[Accessed 16 April 2022].