IBM Developer SKILLS NETWORK

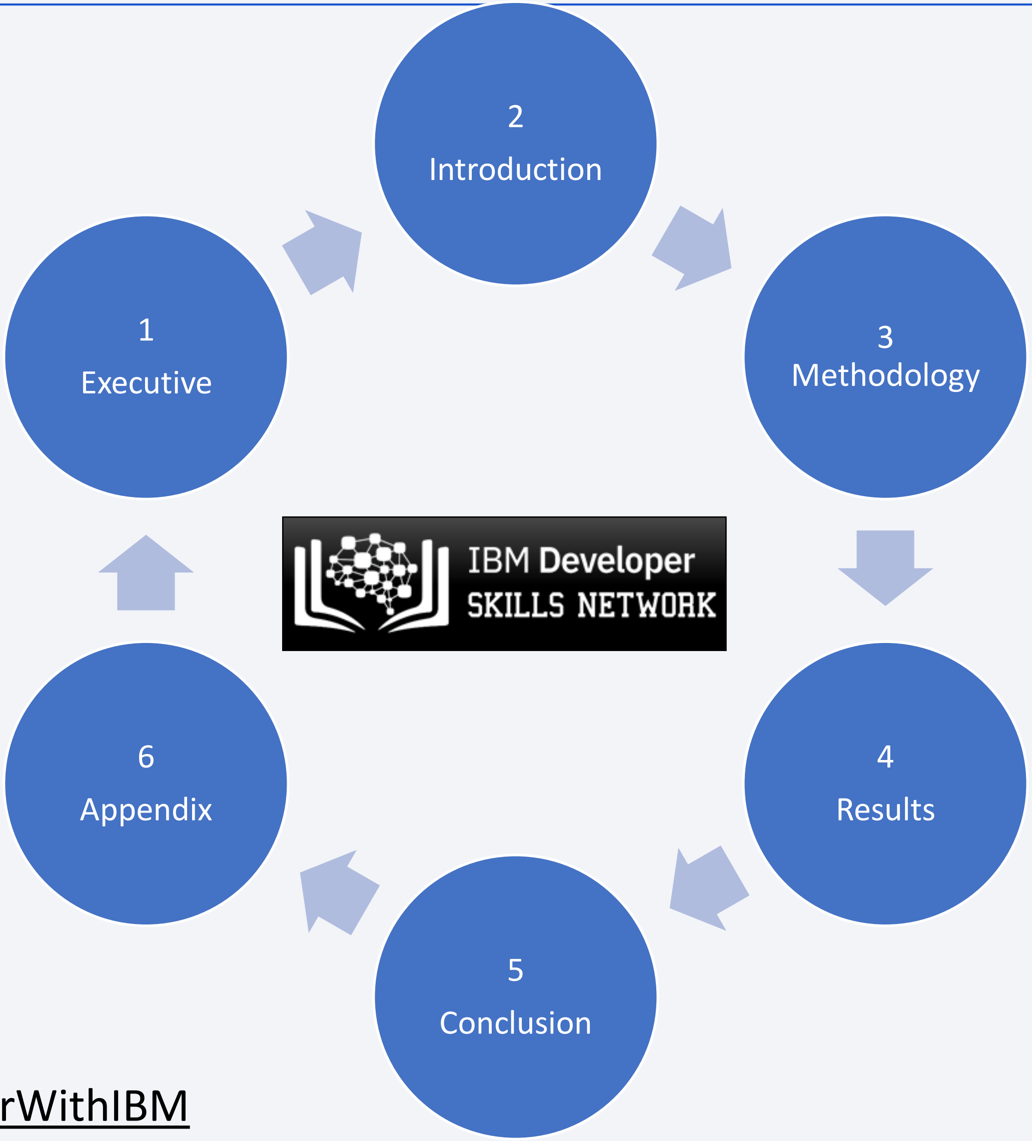
Winning Space Race with Data Science

Dr Lebede Ngartera
PhD Applied Mathematics

12/22/2023

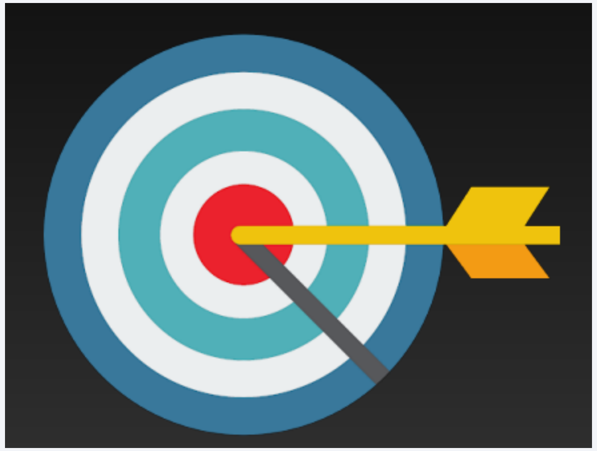
https://github.com/Teraces12/LearnerWithIBM

Outline



https://github.com/Teraces12/LearnerWithIBM

Executive Summary



- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result

https://github.com/Teraces12/LearnerWithIBM

Introduction

Project background and context

- This project centers around the cost-efficient space endeavors of Space X, particularly its Falcon 9 rocket launches priced at $62 million, a stark contrast to competitors' rates exceeding $165 million.
- The core objective is the development of a machine learning pipeline designed to forecast the successful landing of the first stage, a pivotal element for precise launch cost estimation.
- To achieve this goal, the project will delve into identifying influential factors, establishing correlations among them, and determining optimal operating conditions essential for a reliable landing program.

Questions to be answered

- How do variables such as payload mass, launch site, number of flights, and orbits affect the success of the first stage landing?

- Does the rate of successful landings increase over the years?

- What is the best algorithm that can be used for binary classification in this case?

https://github.com/Teraces12/LearnerWithIBM

Section 1

Methodology

Methodology

Overall Methodology:

1-Data Collection, Wrangling, and Formatting:

- Utilizing SpaceX API
- Employing web scraping techniques

2-Exploratory Data Analysis (EDA):

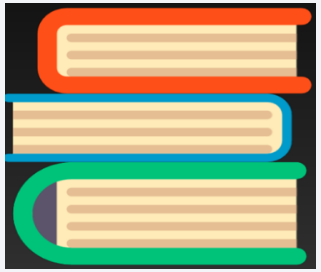
- Leveraging Pandas and NumPy for data manipulation
- Utilizing SQL for querying and analysis

3-Data Visualization:

- Using Matplotlib and Seaborn for static visualizations
- Employing Folium for interactive maps
- Utilizing Dash for interactive web-based visualizations

4-Machine Learning Prediction:

- Implementing Logistic Regression for predictive modeling
- Applying Support Vector Machine (SVM) for classification
- Utilizing Decision Tree for decision-making
- Employing K-Nearest Neighbors (KNN) for pattern recognition



https://github.com/Teraces12/LearnerWithIBM

Data Collection

- **Data collection process** involved a combination of API requests from SpaceX REST API and Web Scraping data from a table in SpaceX's Wikipedia entry.
- We had to use both of these data collection methods in order to get complete information about the launches for a more detailed analysis.
- **Data Columns are obtained by using SpaceX REST API**:

FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude

- **Data Columns are obtained by using Wikipedia Web Scraping:**

Flight No., Launch site, Payload, PayloadMass, Orbit, Customer, Launch outcome, Version Booster, Booster landing, Date, Time

https://github.com/Teraces12/LearnerWithIBM

Data Collection – SpaceX API

The data is available

is the following link:

https://github.com/cestevespr/IBM-final/blob/main/jupyter-labs-spacex-data-collection-api%20(1).ipynb

The data was collected by different methods:

Via SpaceX API: This requires sending an API request to the SpaceX server and getting a JSON response. The process was implemented using the request library.

Via Webscraping: Here, the data was collected by scraping the Wikipedia page for information containing SpaceX rockets. The Beautiful Soup library was employed for scraping, and the scraped HTML data was then parsed using an HTML parser. The necessary data were extracted and used for analysis.

https://github.com/Teraces12/LearnerWithIBM

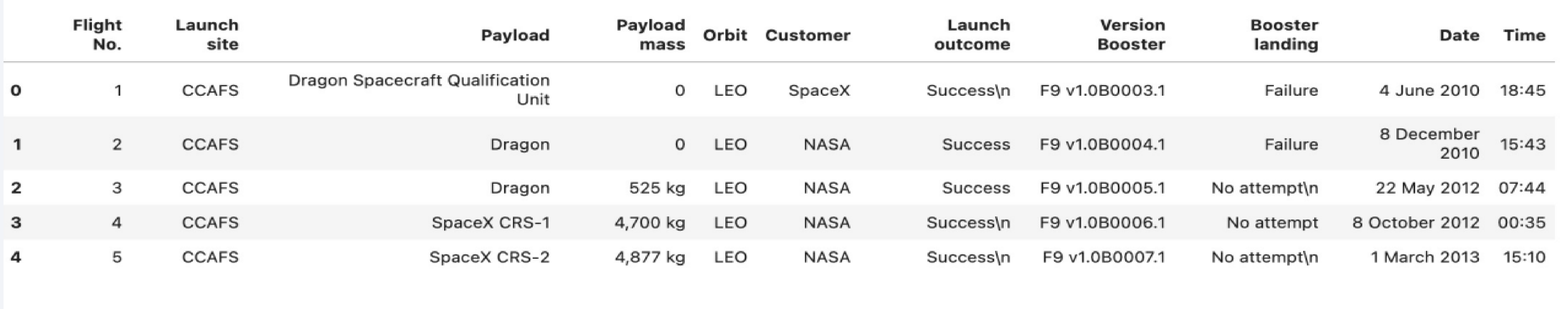
Data Collection – Scraping

1-Data Source: the data is scraped from https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922.

2-Content Overview: the website exclusively contains data about Falcon 9 launches.

3-Data Size: the resulting dataset comprises 121 rows or instances and 11 columns or features.

4-Preview: the picture below provides a snapshot of the first few rows of the collected data.



	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10

Below is the project link :

https://github.com/Teraces12/LearnerWithIBM/blob/9dc401de345afe60ed9038a48766a22f57bfb7d6/Data%20Wrangling.ipynb

Data Wrangling

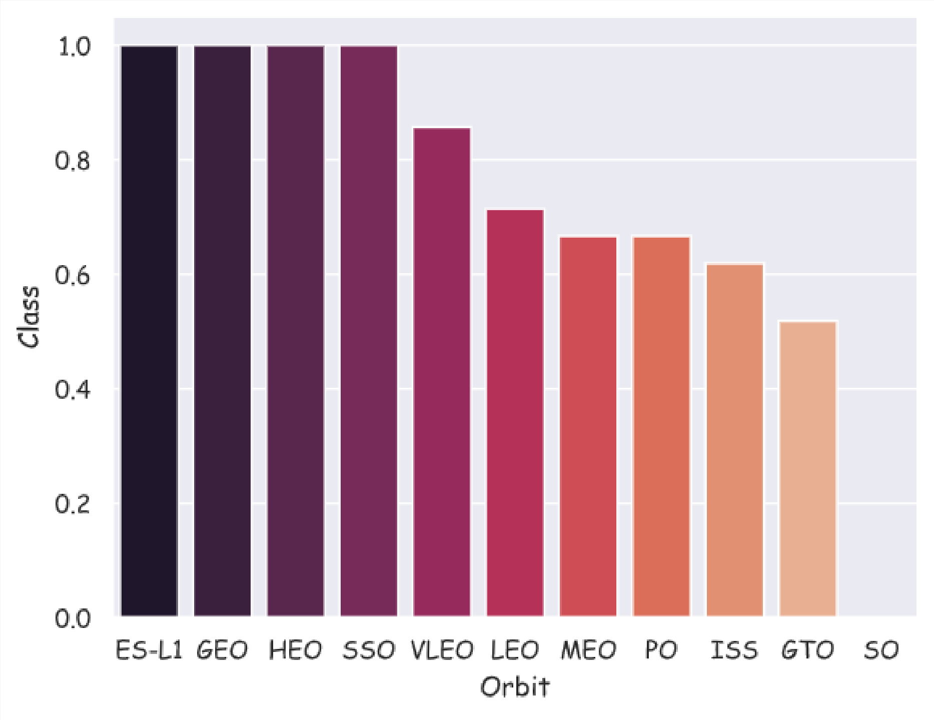
1. **Data Processing:** the data is processed to eliminate missing entries, and categorical features undergo one-hot encoding.
2. **Class Column Addition:** an additional column named 'Class' is introduced to the data frame. The 'Class' column is coded with 0 for failed launches and 1 for successful launches.
3. **Final Dataset:** the processed dataset concludes with 90 rows or instances and 83 columns or features.

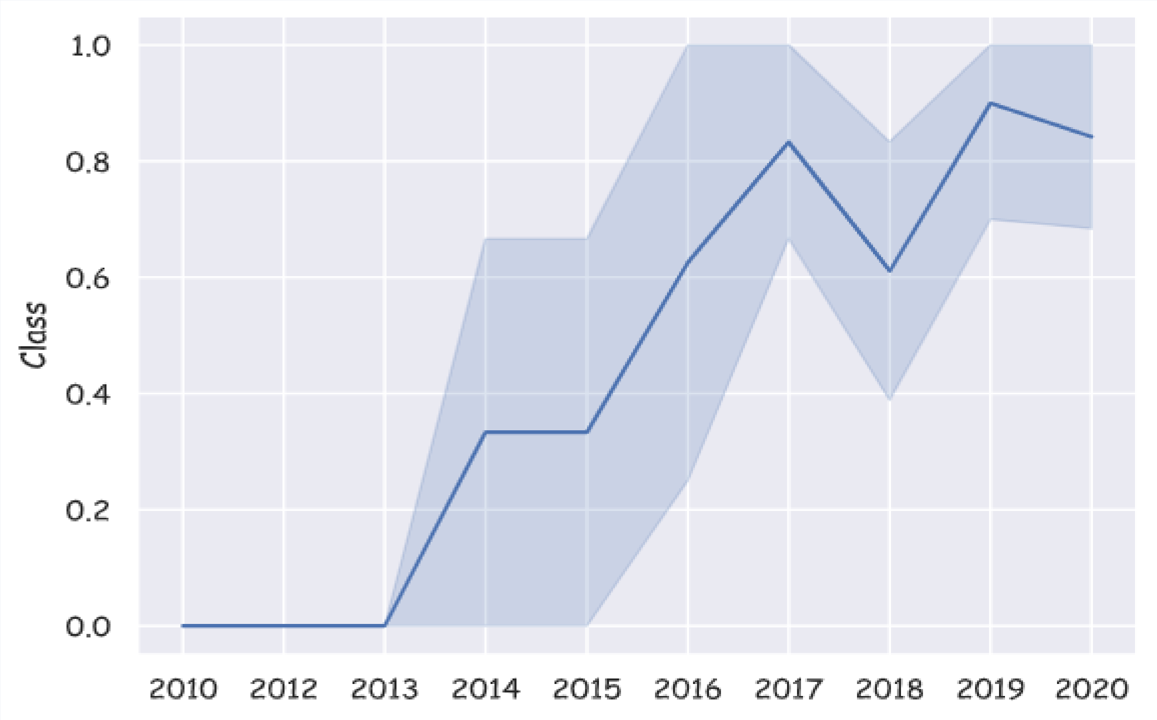
https://github.com/Teraces12/LebedeNgartera.git

EDA with Data Visualization

Bar charts and Line charts were plotted. The bar chart was used to visualize the success rate for each orbit while the line chart was used to show the trend of success rates over the years.

https://github.com/Teraces12/LearnerWithIBM/blob/9dc401de345afe60ed9038a48766a22f57bfb7d6/Data%20Wrangling.ipynb





EDA with SQL

- The data used for the EDA was loaded into SQL Server and then a connection was made to the server using pyodbc module.
- A cursor object was then created from the connection string which was used to run the SQL queries in python.
- The link below would provide a more detailed look into the EDA using SQL

Github:
https://github.com/Teraces12/LearnerWithIBM/blob/9dc401de345afe60ed9038a48766a22f57bfb7d6/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- Using the Folium API, we created interactive maps to visualize the launch sites.
- Additionally, we employed the MarkerCluster object to display all launches at various sites, labeling them as either successful or failed.
- To obtain the coordinates of the coastline, railway, highway, and a city, we utilized the MousePosition feature.
- Subsequently, we calculated the distance between the launch center and these specified locations.

Github:
https://github.com/Teraces12/LearnerWithIBM/blob/9dc401de345afe60ed9038a48766a22f57bfb7d6/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

Predictive Analysis (Classification)

- The data was normalized using a standard scaler before being passed to several classification algorithms.

- Subsequently, the dataset was split into training and testing sets using the train_test_split function.

- After training the data on different algorithms, the accuracy scores of each model were plotted on a bar chart, revealing that the Decision Tree model performed the best in accurately predicting whether the Falcon 9 rocket would land or not.

Github:
https://github.com/Teraces12/LearnerWithIBM/blob/9dc401de345afe60ed9038a48766a22f57bfb7d6/Machine%20Learning%20Prediction.ipynb

Build a Dashboard with Plotly Dash

Dash Implementation:

1. **Interactive Site Generation:** functions from Dash are employed to create an interactive site.
2. **User Input Controls:** the interactive site incorporates a dropdown menu and a range slider, providing users with the ability to toggle inputs.
3. **Visualizations:** utilizing a pie chart and a scatterplot, the interactive site visually represents:

- The total number of successful launches from each launch site.
- The correlation between payload mass and mission outcome (success or failure) for each launch site.

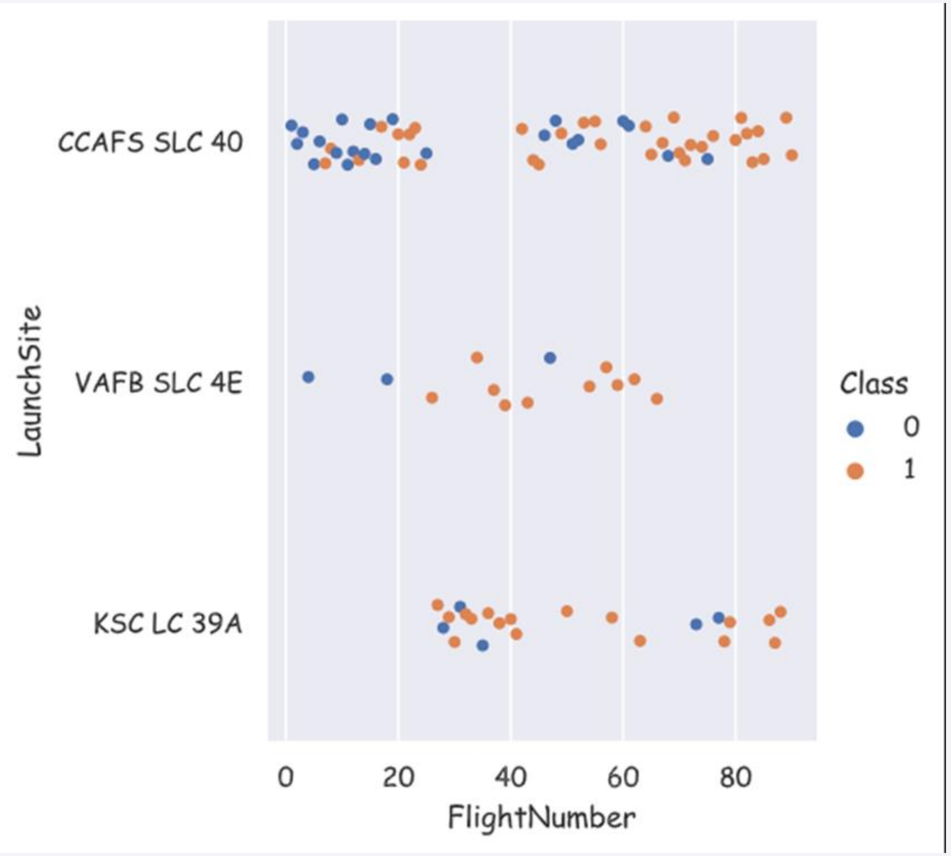
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- We can see clearly from the chart the number of flights that succeeded and failed in each launch site.

- KSC LC 39A and VAFB SLC 4E clearly had lesser number of flights and a higher success rate than CCAFS SLC 40.

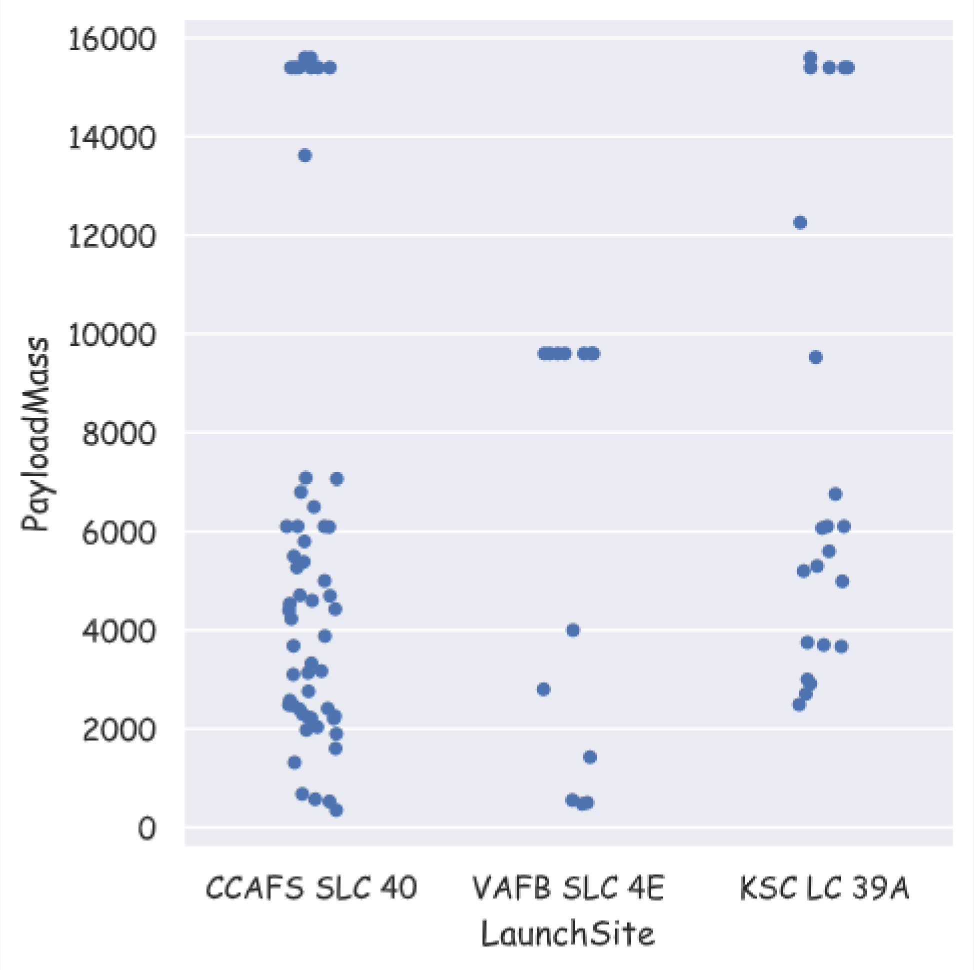


https://github.com/Teraces12/LearnerWithIBM

Payload vs. Launch Site

- We can see clearly from the chart the launch sites and their distribution of payload masses for each launch.

- CCAFS SLC 40 and KSC LC 39A both had flights with payload mass above 15000kg

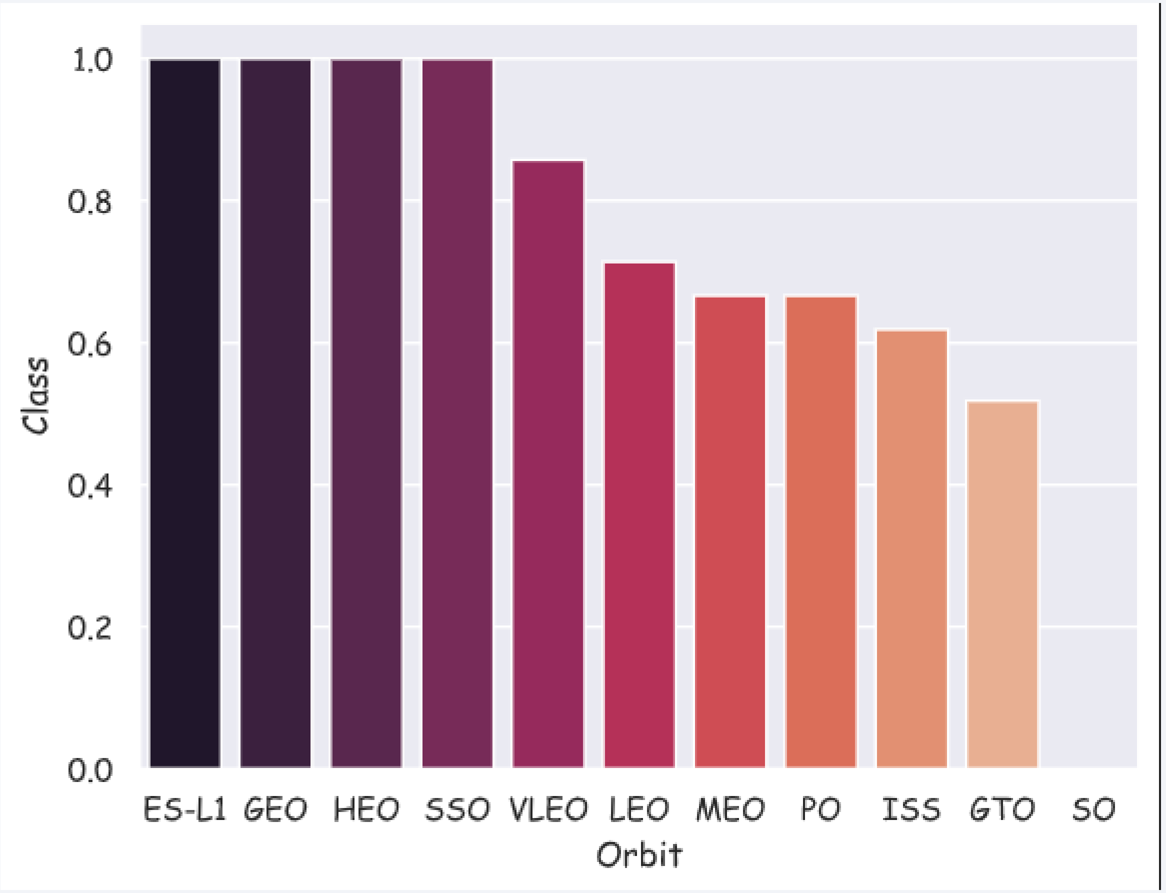


https://github.com/Teraces12/LearnerWithIBM

Success Rate vs. Orbit Type

- This chart shows the success rate for each orbit.

- Orbits ES-L1, GEO, HEO, and SSO has the highest success rates among all orbits.

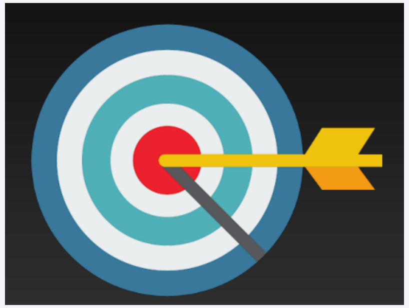


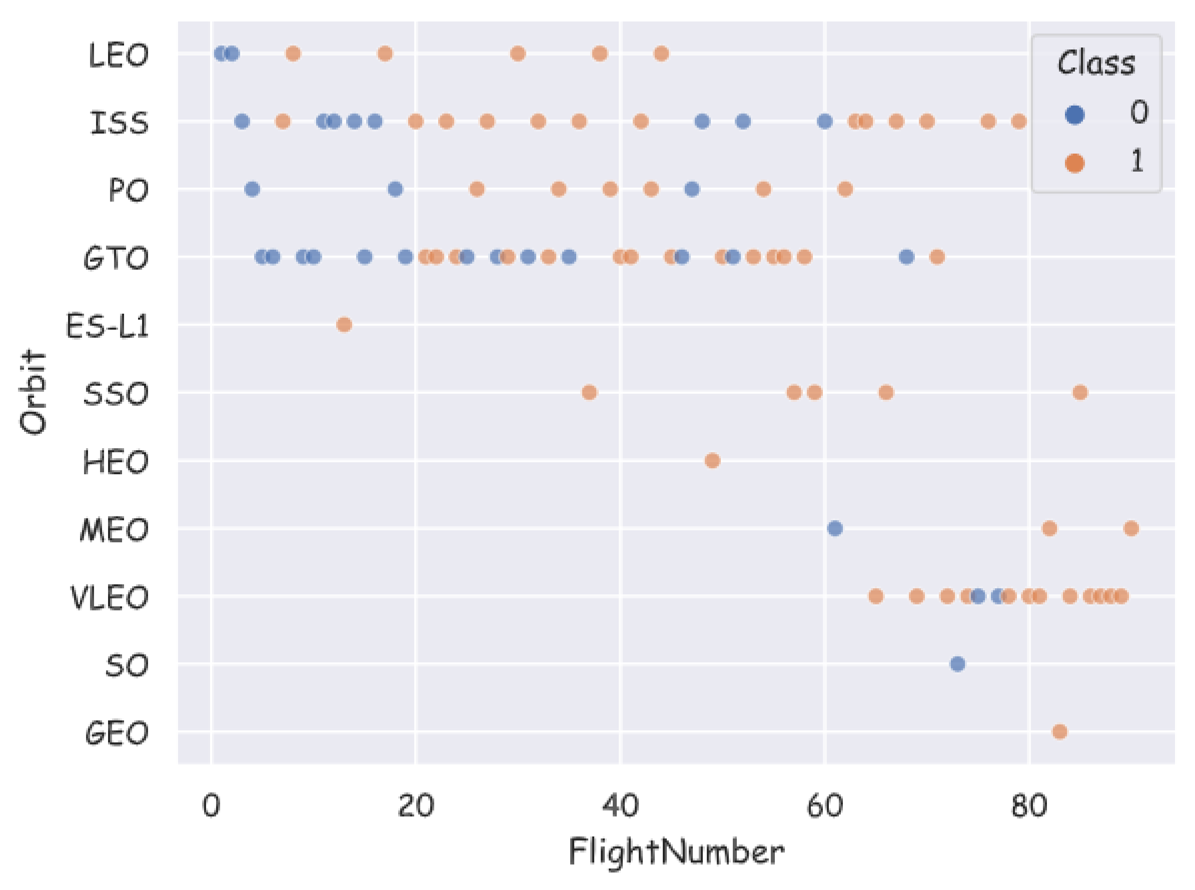
https://github.com/Teraces12/LearnerWithIBM

Flight Number vs. Orbit Type

- The chart here shows the success rate for each orbit.

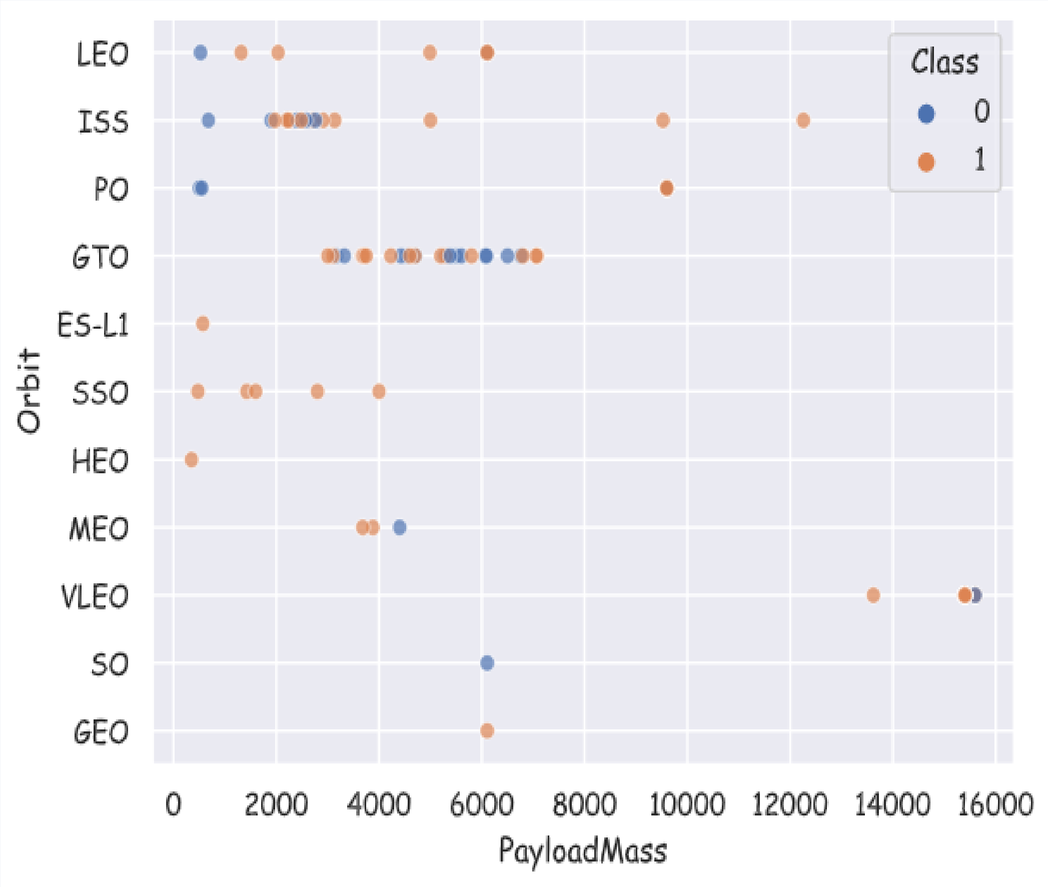
- You would notice that the more the number of flights for each orbit the lesser the success rate.



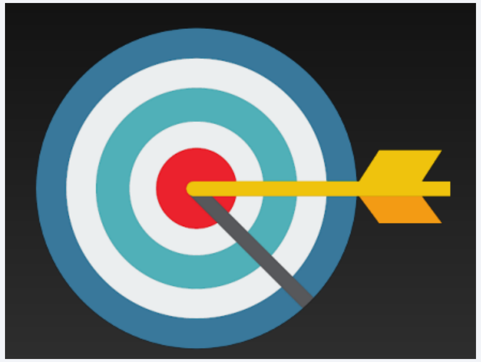


https://github.com/Teraces12/LearnerWithIBM

Payload vs. Orbit Type



This chart shows the payload mass for each orbit type.

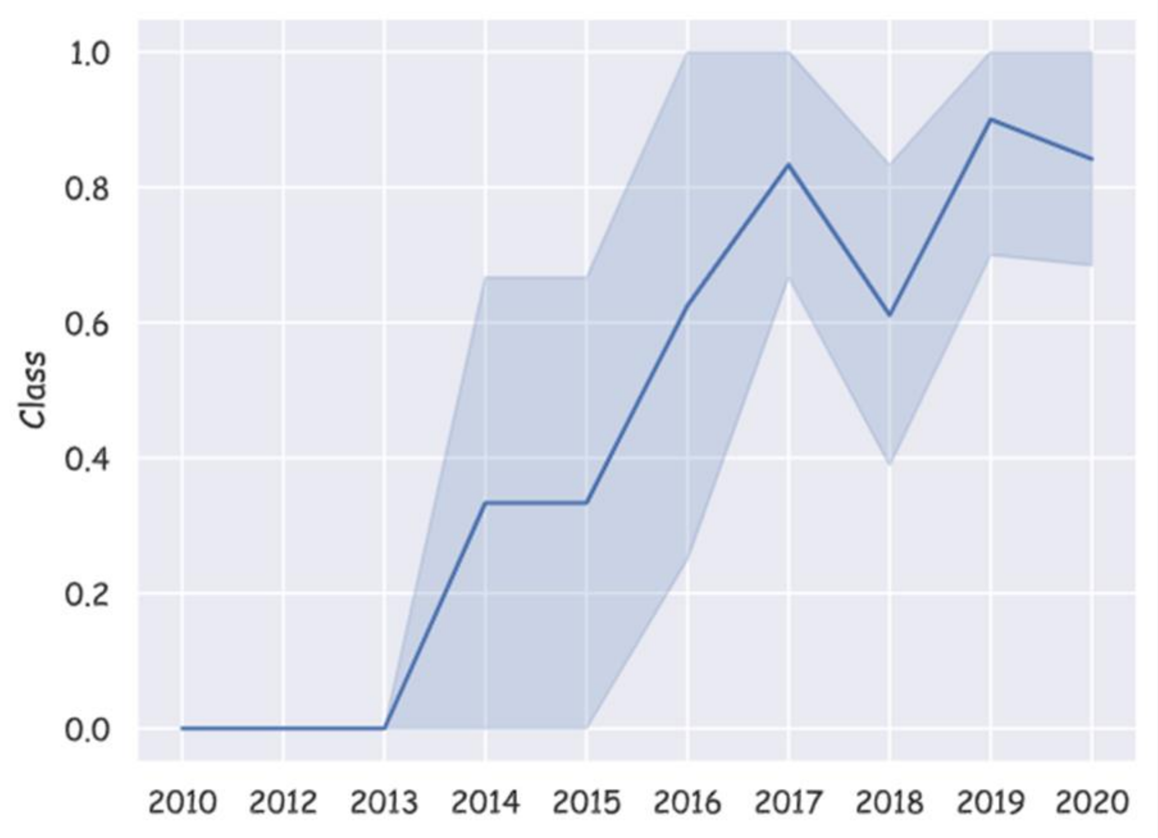


https://github.com/Teraces12/LearnerWithIBM

Launch Success Yearly Trend

- This chart shows the trend of the success and the failure rates for each year.

- We can see a markedly increase of the success rate from 2013 and a sharp drop between 2017 and 2018.

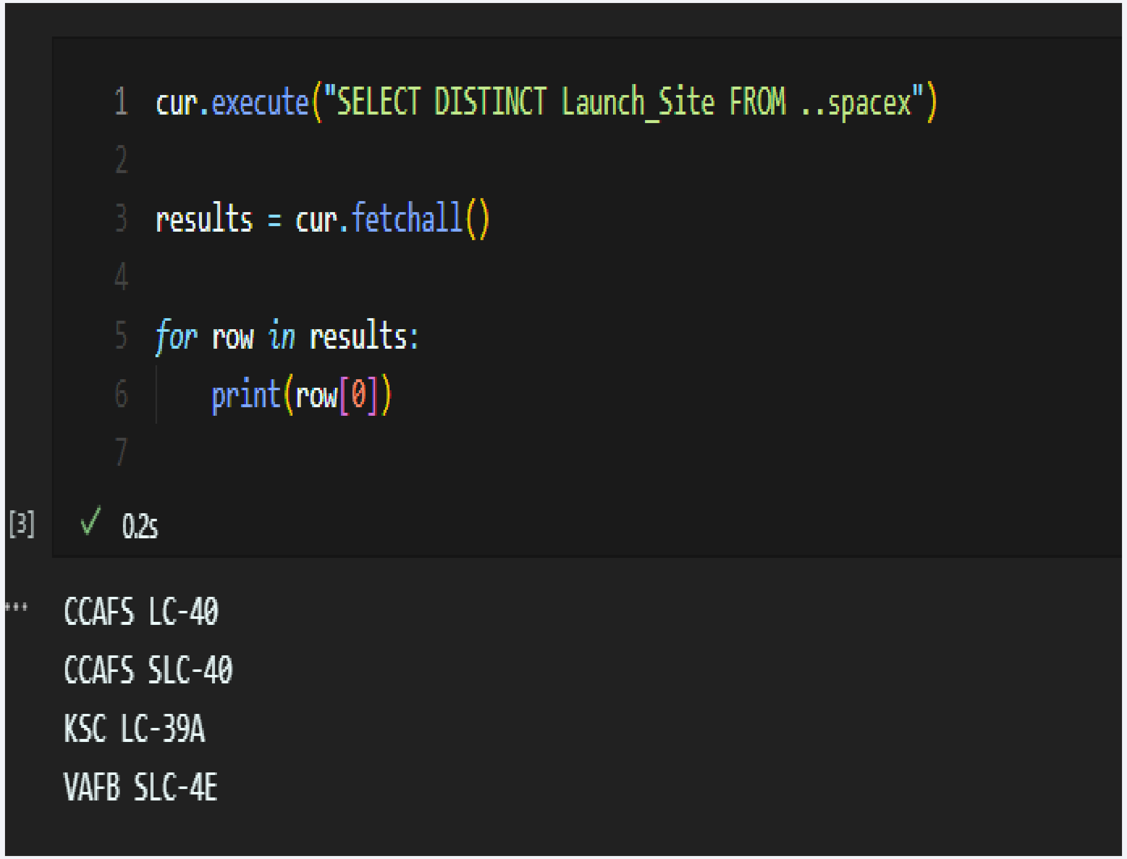


https://github.com/Teraces12/LearnerWithIBM

All Launch Site Names

- The query shows the different sites names.

- The distinct sites names were gotten by using the ‘DISTINCT’ keyword to show the unique launch sites.



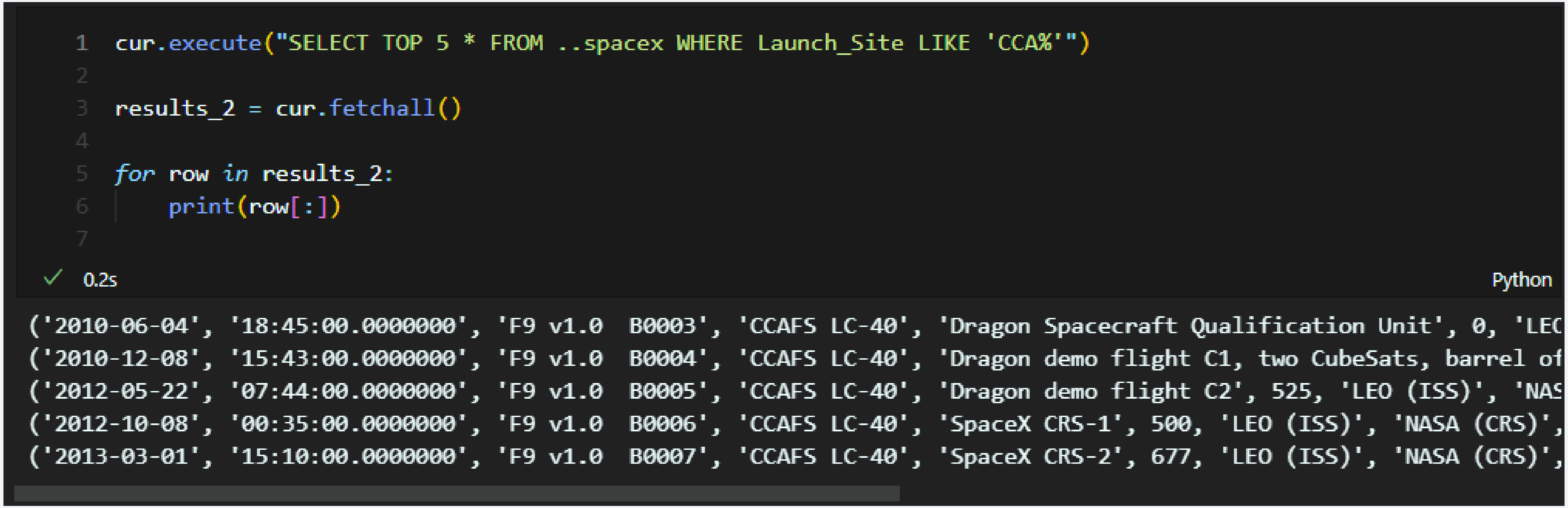
```
cur.execute("SELECT DISTINCT Launch_Site FROM ..spacex")

results = cur.fetchall()

for row in results:
    print(row[0])

```

```
[3]  ✓ 0.2s
```

```
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E
```

https://github.com/Teraces12/LearnerWithIBM

Launch Site Names Begin with 'CCA'



```python
cur.execute("SELECT TOP 5 * FROM ..spacex WHERE Launch_Site LIKE 'CCA%'")

results_2 = cur.fetchall()

for row in results_2:
    print(row[:])

```

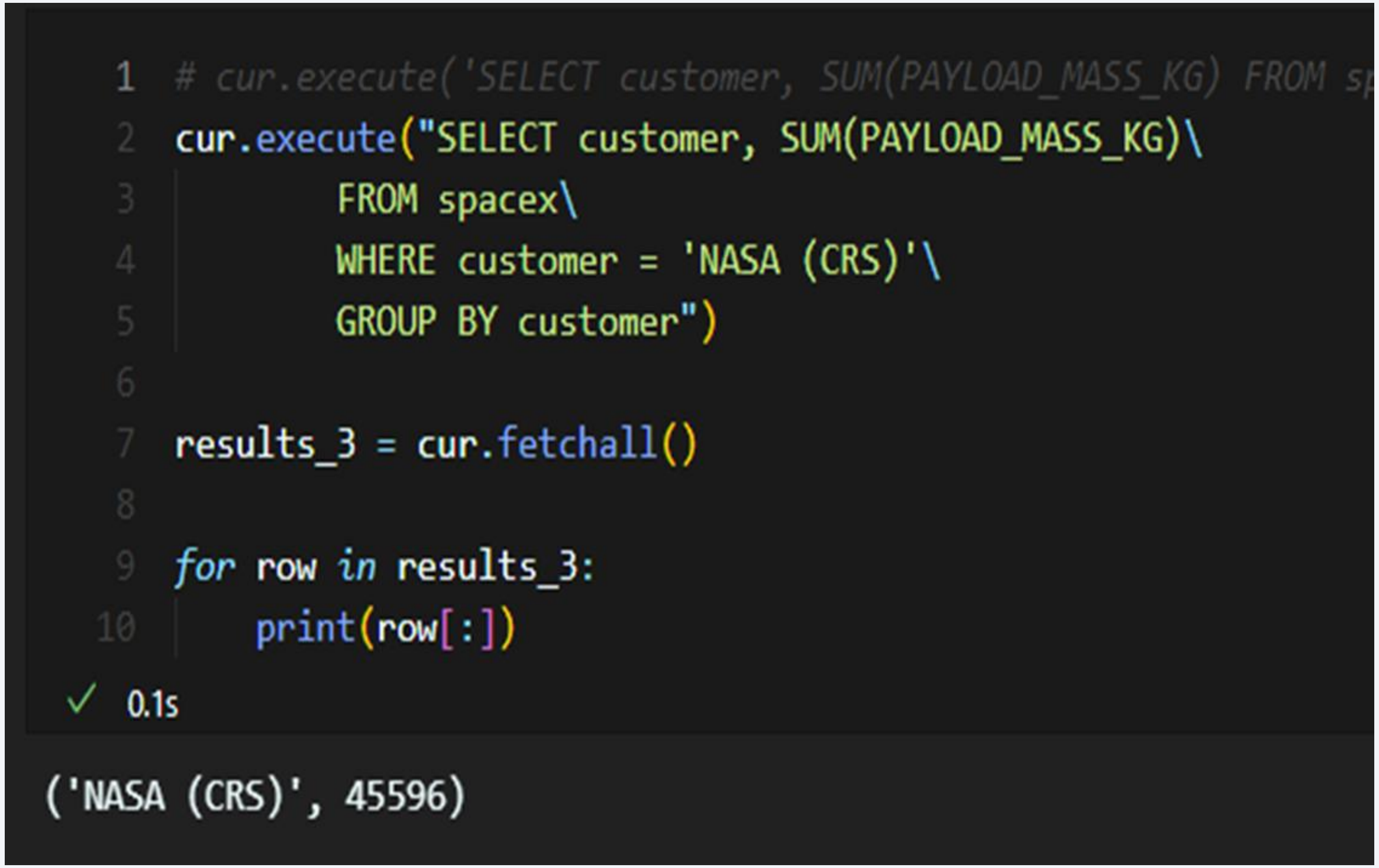
```
('2010-06-04', '18:45:00.0000000', 'F9 v1.0  B0003', 'CCAFS LC-40', 'Dragon Spacecraft Qualification Unit', 0, 'LEO
('2010-12-08', '15:43:00.0000000', 'F9 v1.0  B0004', 'CCAFS LC-40', 'Dragon demo flight C1, two CubeSats, barrel of
('2012-05-22', '07:44:00.0000000', 'F9 v1.0  B0005', 'CCAFS LC-40', 'Dragon demo flight C2', 525, 'LEO (ISS)', 'NAS
('2012-10-08', '00:35:00.0000000', 'F9 v1.0  B0006', 'CCAFS LC-40', 'SpaceX CRS-1', 500, 'LEO (ISS)', 'NASA (CRS)',
('2013-03-01', '15:10:00.0000000', 'F9 v1.0  B0007', 'CCAFS LC-40', 'SpaceX CRS-2', 677, 'LEO (ISS)', 'NASA (CRS)',
```

- Here, you can see the top 5 records of the launch sites that starts with ‘CCA’.
- I used the ‘TOP’, ‘WHERE’, and ‘LIKE’ keywords to get this information.

Total Payload Mass

- This screenshot shows the total payload mass carried by boosters launched by NASA (CRS).

- The total payload mass is 45,596kg.



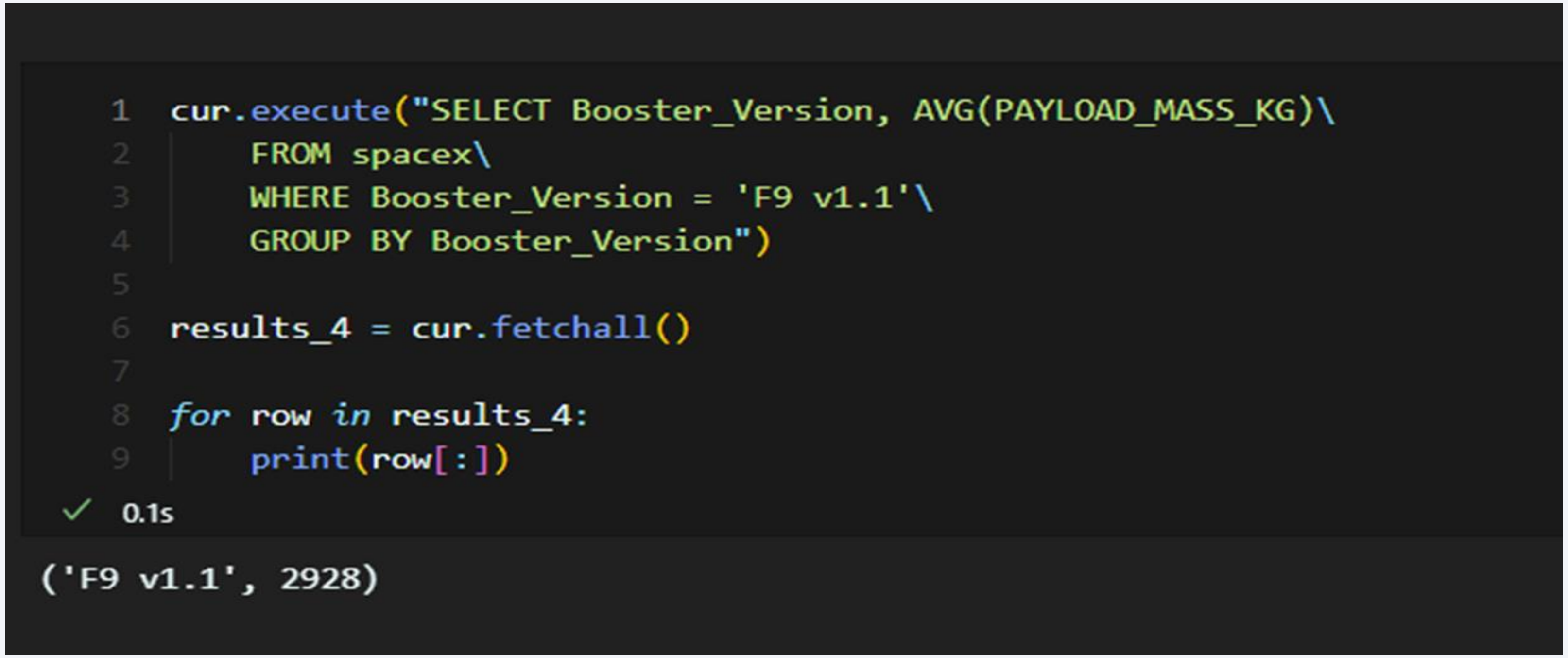
```
# cur.execute('SELECT customer, SUM(PAYLOAD_MASS_KG) FROM sp
cur.execute("SELECT customer, SUM(PAYLOAD_MASS_KG)\
            FROM spacex\
            WHERE customer = 'NASA (CRS)'\
            GROUP BY customer")

results_3 = cur.fetchall()

for row in results_3:
    print(row[:])
```

✓ 0.1s

```
('NASA (CRS)', 45596)
```

https://github.com/Teraces12/LearnerWithIBM

Average Payload Mass by F9 v1.1



```
cur.execute("SELECT Booster_Version, AVG(PAYLOAD_MASS_KG)\
          FROM spacex\
          WHERE Booster_Version = 'F9 v1.1'\
          GROUP BY Booster_Version")

results_4 = cur.fetchall()

for row in results_4:
    print(row[:])
```

✓ 0.1s

```
('F9 v1.1', 2928)
```

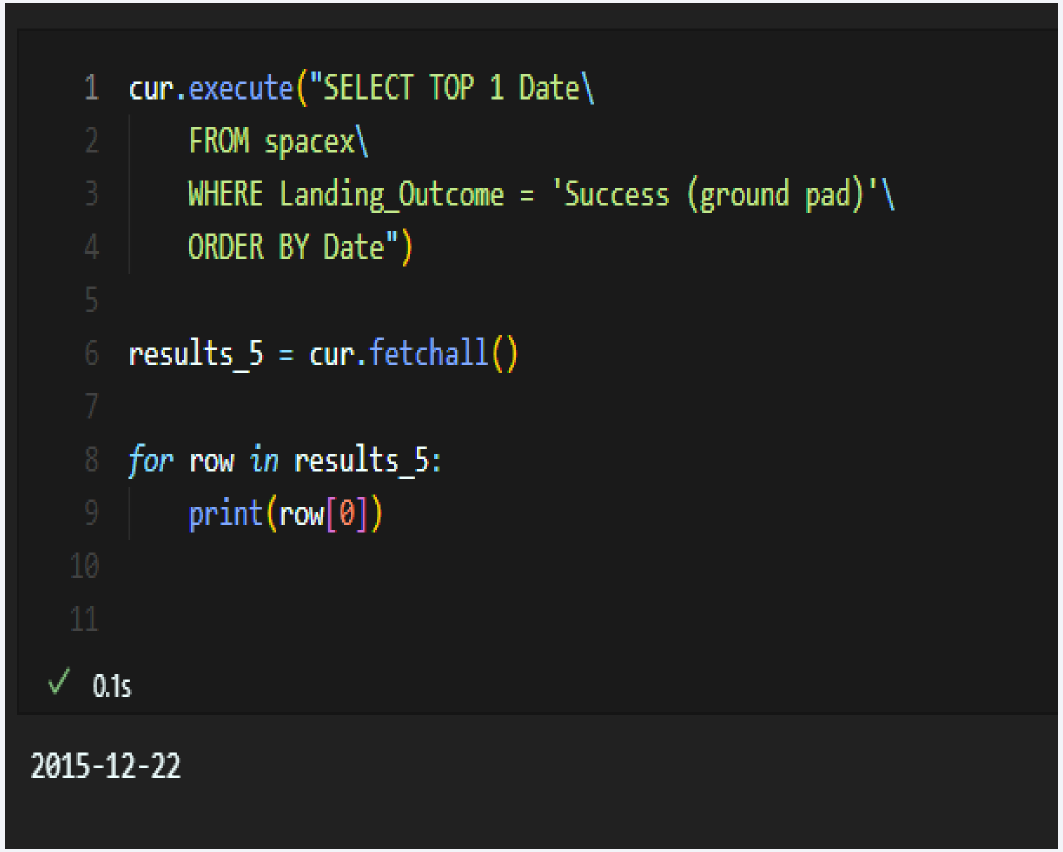
This shows the average payload mass by F9 v1.1 booster which is 2928.

https://github.com/Teraces12/LearnerWithIBM

First Successful Ground Landing Date

- This shows the data for the first successful landing on the ground pad.

- The date is shown to be 2015-12-22



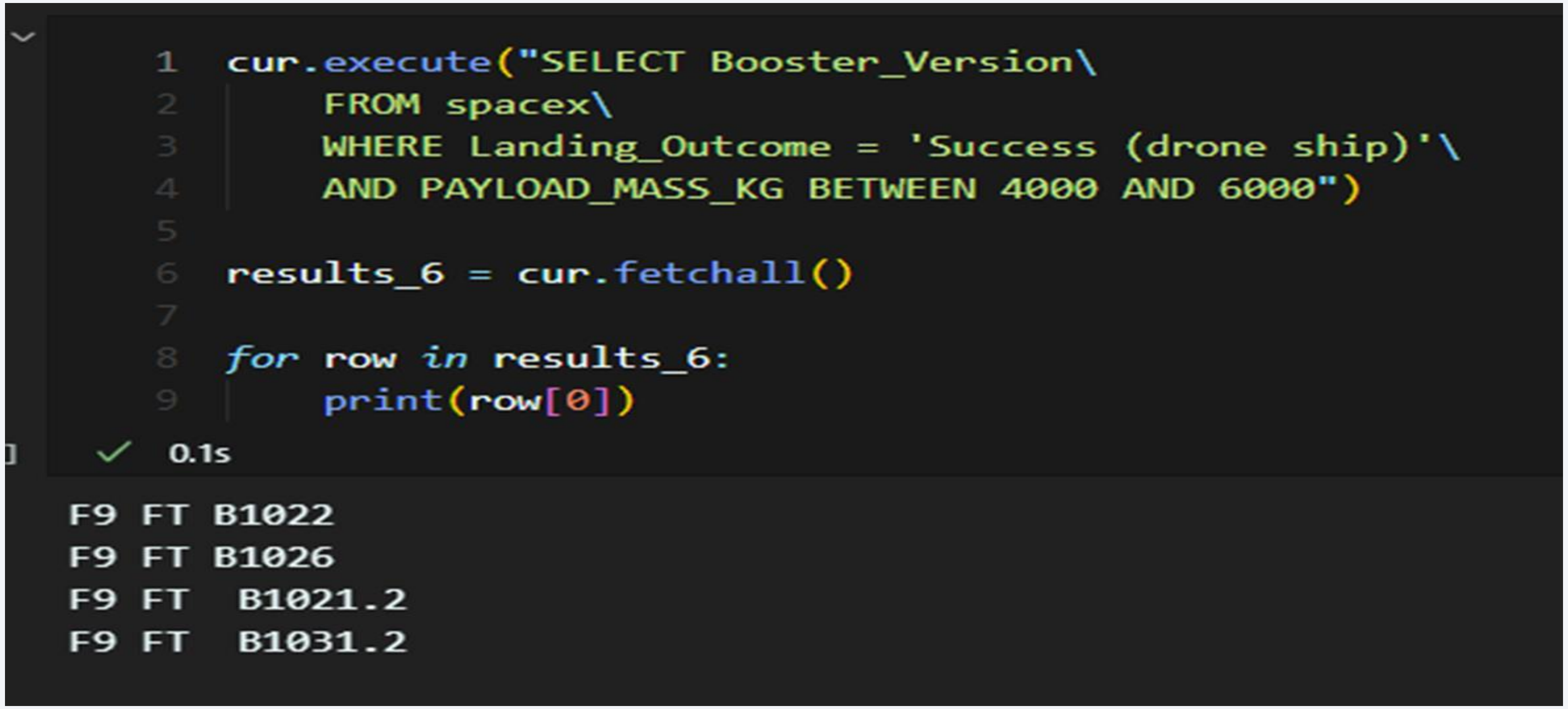
```
cur.execute("SELECT TOP 1 Date\
    FROM spacex\
    WHERE Landing_Outcome = 'Success (ground pad)'\
    ORDER BY Date")

results_5 = cur.fetchall()

for row in results_5:
    print(row[0])
```

✓ 0.1s

```
2015-12-22
```

https://github.com/Teraces12/LearnerWithIBM

Successful Drone Ship Landing with Payload between 4000 and 6000



```
cur.execute("SELECT Booster_Version\
    FROM spacex\
    WHERE Landing_Outcome = 'Success (drone ship)'\
    AND PAYLOAD_MASS_KG BETWEEN 4000 AND 6000")

results_6 = cur.fetchall()

for row in results_6:
    print(row[0])
```

✓ 0.1s

```
F9 FT B1022
F9 FT B1026
F9 FT  B1021.2
F9 FT  B1031.2
```

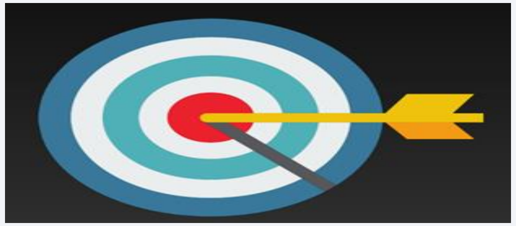
This shows the boosters which have successfully landed on a drone ship with a payload mass between 4000 and 6000.

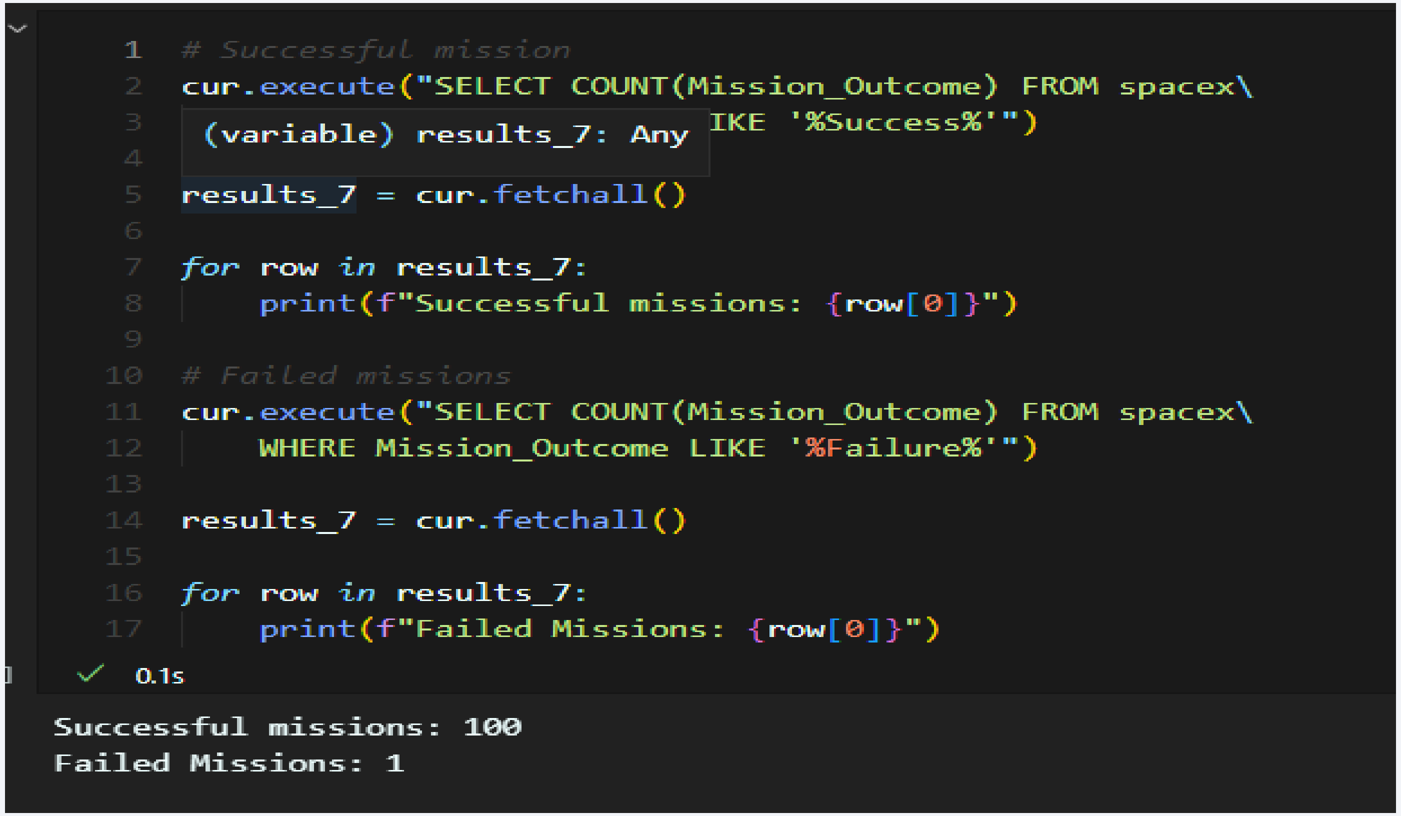
https://github.com/Teraces12/LearnerWithIBM

Total Number of Successful and Failure Mission Outcomes

- This shows the number of successful and failed mission outcomes.

- Here, we have 100 successful missions and just 1 failed missions.





```
# Successful mission
cur.execute("SELECT COUNT(Mission_Outcome) FROM spacex\
 (variable) results_7: Any IKE '%Success%'")

results_7 = cur.fetchall()

for row in results_7:
    print(f"Successful missions: {row[0]}")

# Failed missions
cur.execute("SELECT COUNT(Mission_Outcome) FROM spacex\
    WHERE Mission_Outcome LIKE '%Failure%'")

results_7 = cur.fetchall()

for row in results_7:
    print(f"Failed Missions: {row[0]}")
```

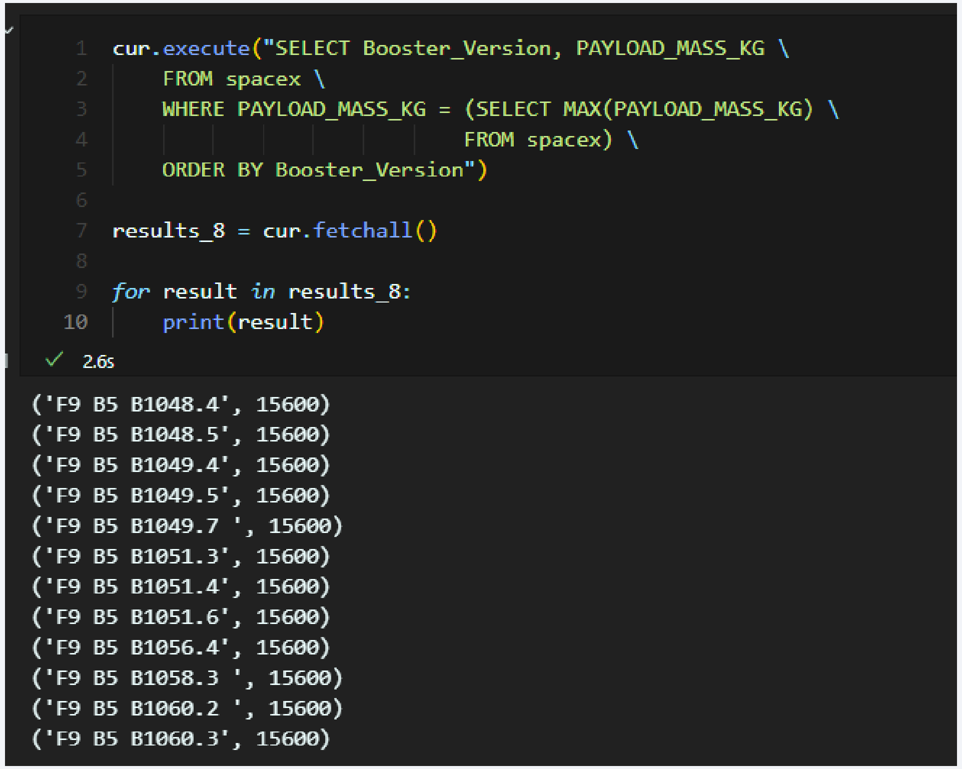
✓ 0.1s

```
Successful missions: 100
Failed Missions: 1
```

https://github.com/Teraces12/LearnerWithIBM

Boosters Carried Maximum Payload

This shows the boosters that carried the maximum payload. We have 12 of them listed.

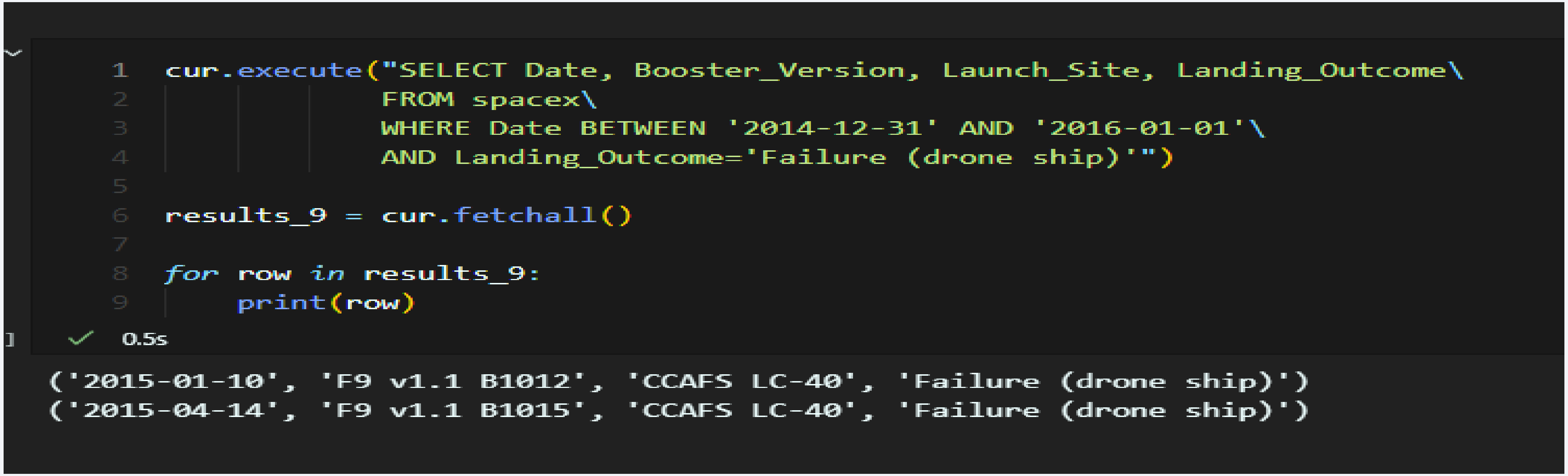


```
cur.execute("SELECT Booster_Version, PAYLOAD_MASS_KG \
    FROM spacex \
    WHERE PAYLOAD_MASS_KG = (SELECT MAX(PAYLOAD_MASS_KG) \
                            FROM spacex) \
    ORDER BY Booster_Version")

results_8 = cur.fetchall()

for result in results_8:
    print(result)
```

```
('F9 B5 B1048.4', 15600)
('F9 B5 B1048.5', 15600)
('F9 B5 B1049.4', 15600)
('F9 B5 B1049.5', 15600)
('F9 B5 B1049.7 ', 15600)
('F9 B5 B1051.3', 15600)
('F9 B5 B1051.4', 15600)
('F9 B5 B1051.6', 15600)
('F9 B5 B1056.4', 15600)
('F9 B5 B1058.3 ', 15600)
('F9 B5 B1060.2 ', 15600)
('F9 B5 B1060.3', 15600)
```

https://github.com/Teraces12/LearnerWithIBM

2015 Launch Records



```
cur.execute("SELECT Date, Booster_Version, Launch_Site, Landing_Outcome\
                FROM spacex\
                WHERE Date BETWEEN '2014-12-31' AND '2016-01-01'\
                AND Landing_Outcome='Failure (drone ship)'")

results_9 = cur.fetchall()

for row in results_9:
    print(row)
```

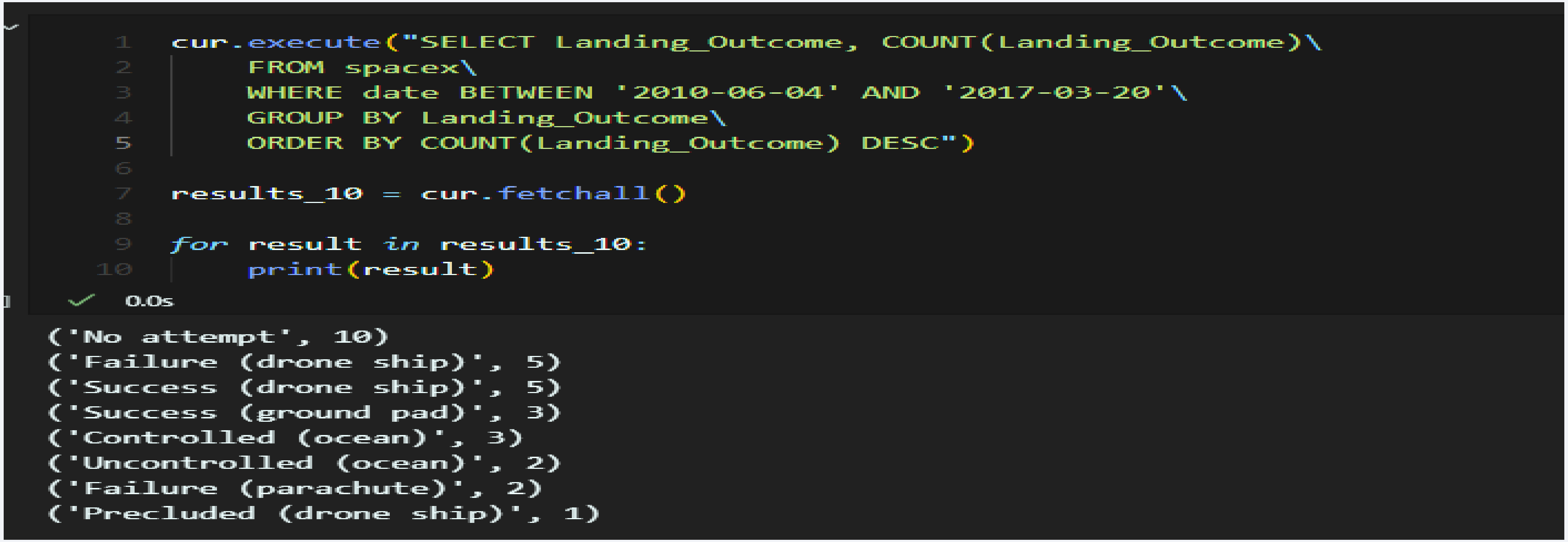
✓ 0.5s

```
('2015-01-10', 'F9 v1.1 B1012', 'CCAFS LC-40', 'Failure (drone ship)')
('2015-04-14', 'F9 v1.1 B1015', 'CCAFS LC-40', 'Failure (drone ship)')
```

This shows the launch records for 2015. Here, we have just 2 records from 2015.

https://github.com/Teraces12/LearnerWithIBM

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20



```
cur.execute("SELECT Landing_Outcome, COUNT(Landing_Outcome)\
    FROM spacex\
    WHERE date BETWEEN '2010-06-04' AND '2017-03-20'\
    GROUP BY Landing_Outcome\
    ORDER BY COUNT(Landing_Outcome) DESC")

results_10 = cur.fetchall()

for result in results_10:
    print(result)
```

✓ 0.0s

```
('No attempt', 10)
('Failure (drone ship)', 5)
('Success (drone ship)', 5)
('Success (ground pad)', 3)
('Controlled (ocean)', 3)
('Uncontrolled (ocean)', 2)
('Failure (parachute)', 2)
('Precluded (drone ship)', 1)
```

This shows the count of successful landing outcomes between 04-06-2010 and 20-03-2017.

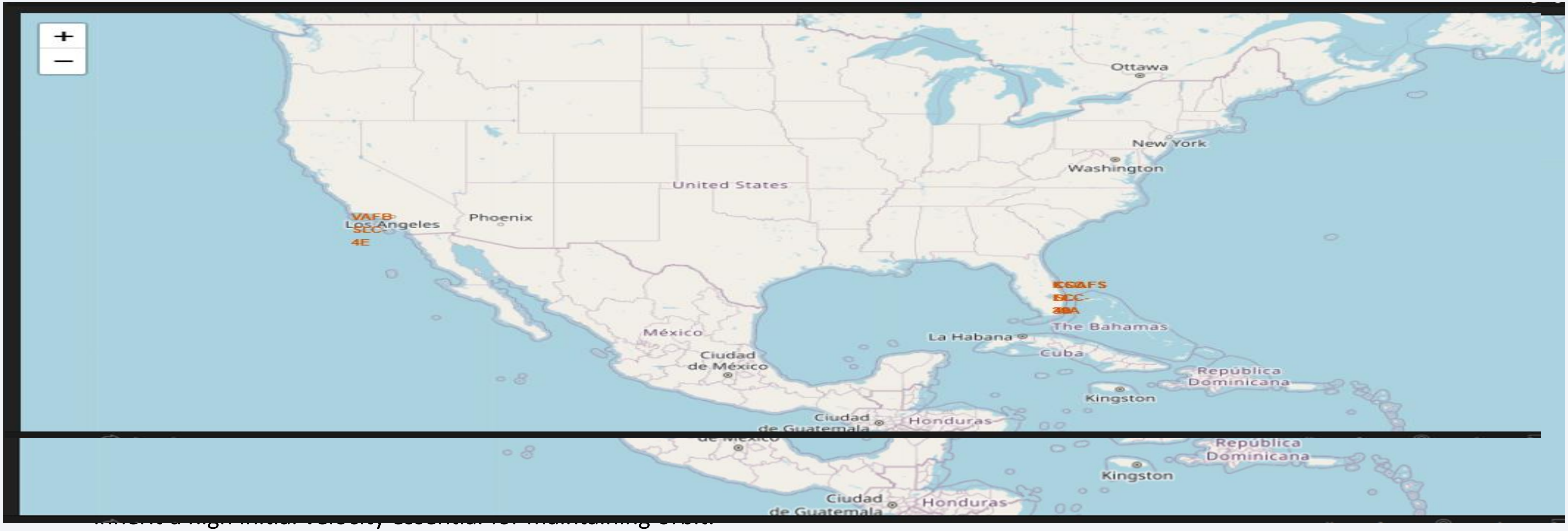
https://github.com/Teraces12/LearnerWithIBM

Section 3

Launch Sites Proximities Analysis

https://github.com/Teraces12/LearnerWithIBM

Launch Sites

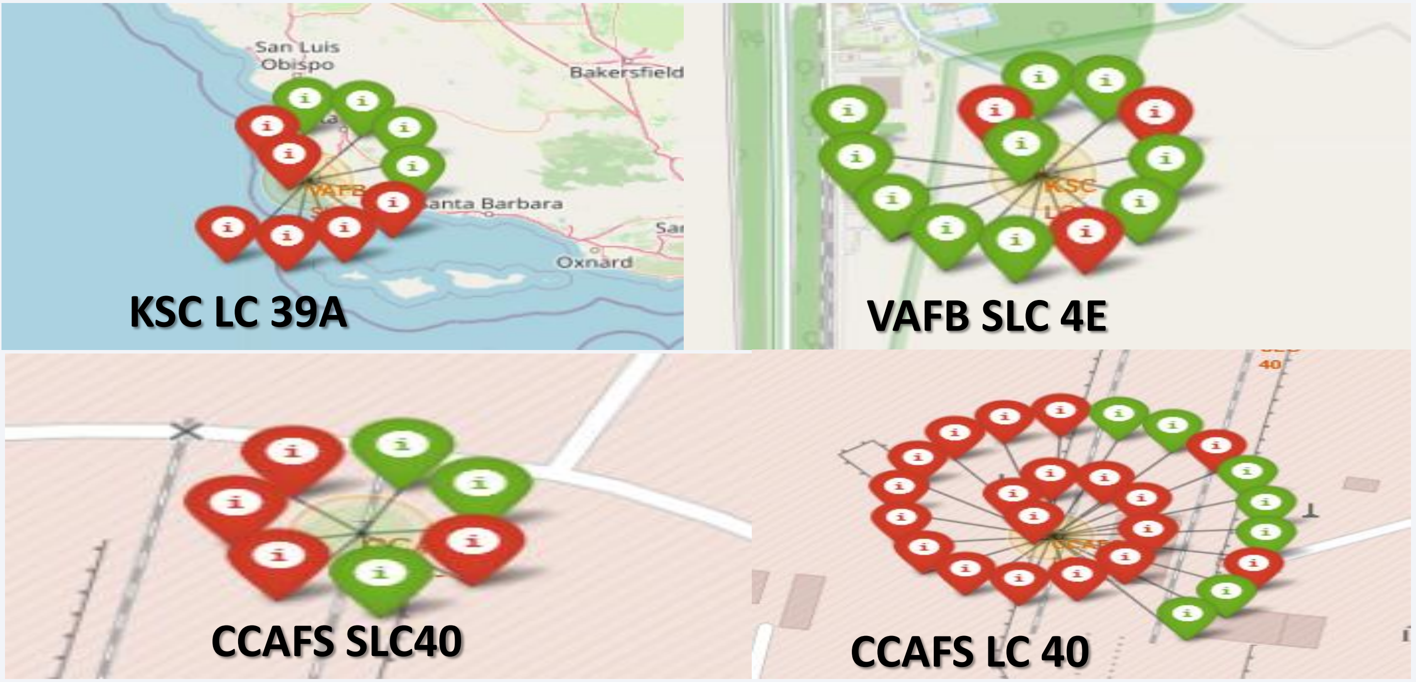


inherit a high initial velocity, essential for maintaining orbit.

- Additionally, situating launch sites close to coastlines minimizes the risk of debris endangering populated areas when rockets are launched over the ocean.

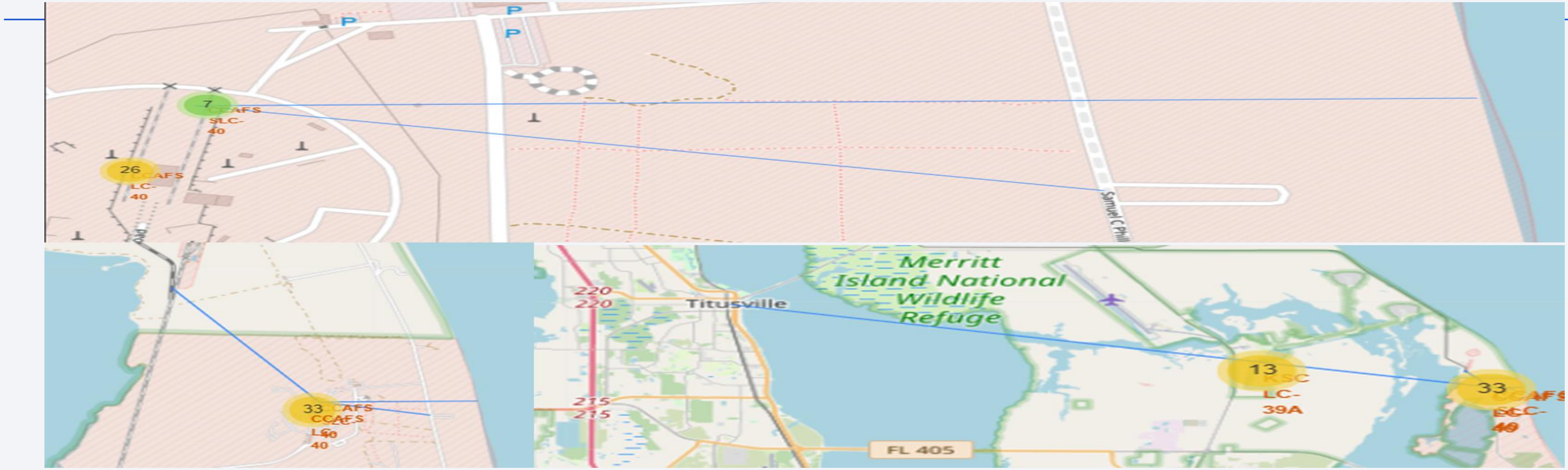
https://github.com/Teraces12/LearnerWithIBM

Successful and Failed Launches at different launch sites



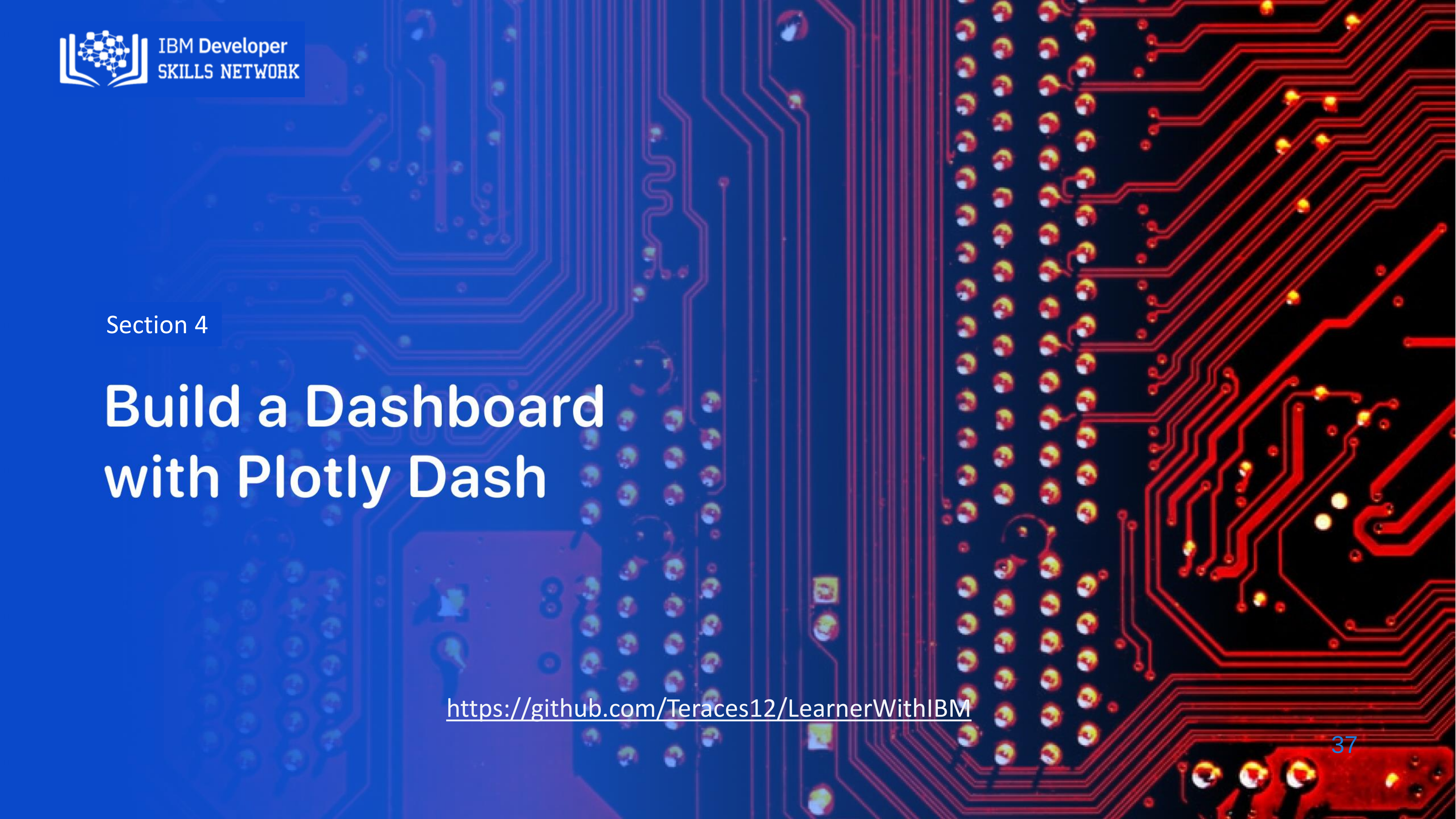
The following screenshots shows the different launch sites with successful launches in green and failed launches in red.

Launch Sites to Markers distance



All launch sites are in very close proximity to the coast and they are also a couple thousand kilometers away from the equator line.

https://github.com/Teraces12/LearnerWithIBM

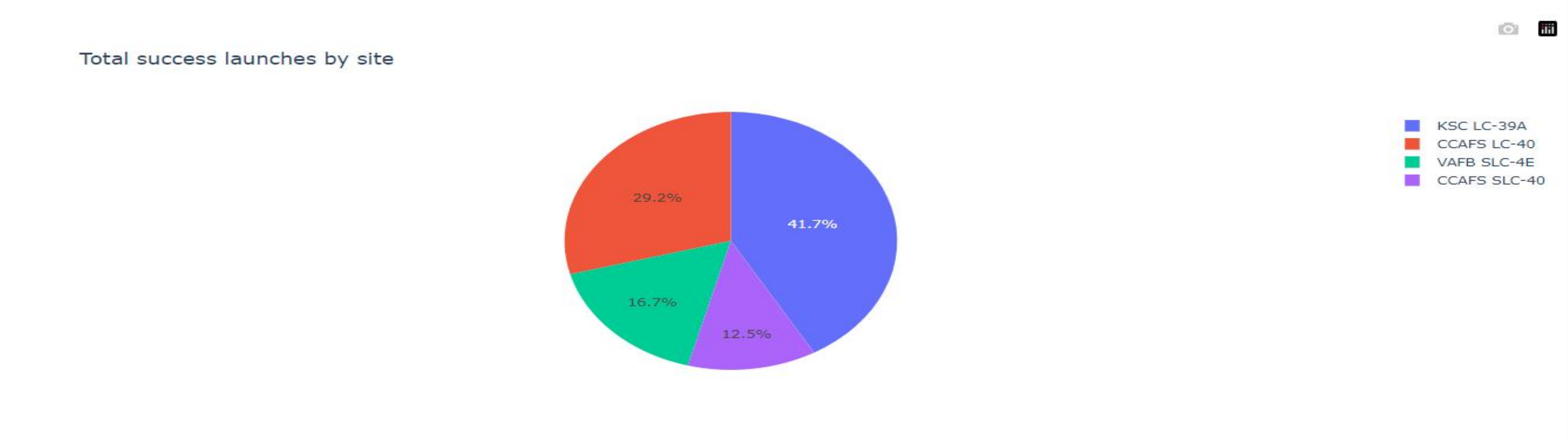


Section 4

Build a Dashboard with Plotly Dash

https://github.com/Teraces12/LearnerWithIBM

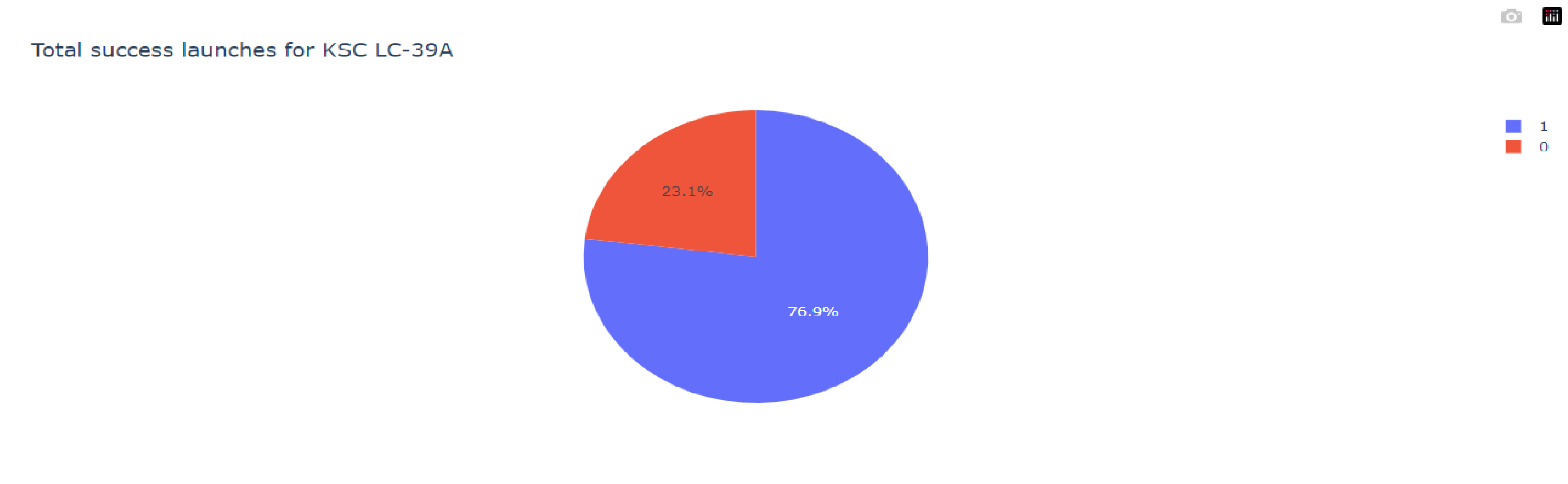
Total success launches by sites



We can see from the plot that Site KSC LC-39A has the largest successful launches a s well the highest launch success rate.

https://github.com/Teraces12/LearnerWithIBM

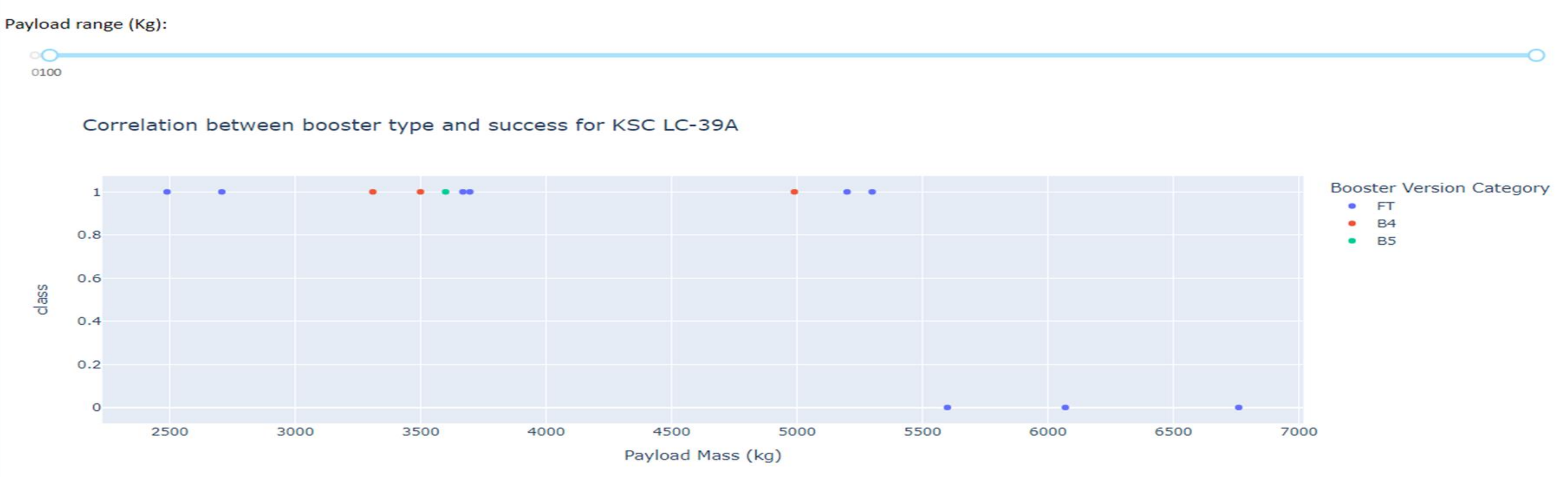
Success ratio for KSC LC-39A



We can see that 76.9% of the total launches at site KSC LC-39A were successful. This is a the highest success rate of all the different launch sites.

https://github.com/Teraces12/LearnerWithIBM

Payload vs Launch outcome scatter plot



It appears that the payload range between 2000 kg and 4000 kg has the highest success rate.

https://github.com/Teraces12/LearnerWithIBM

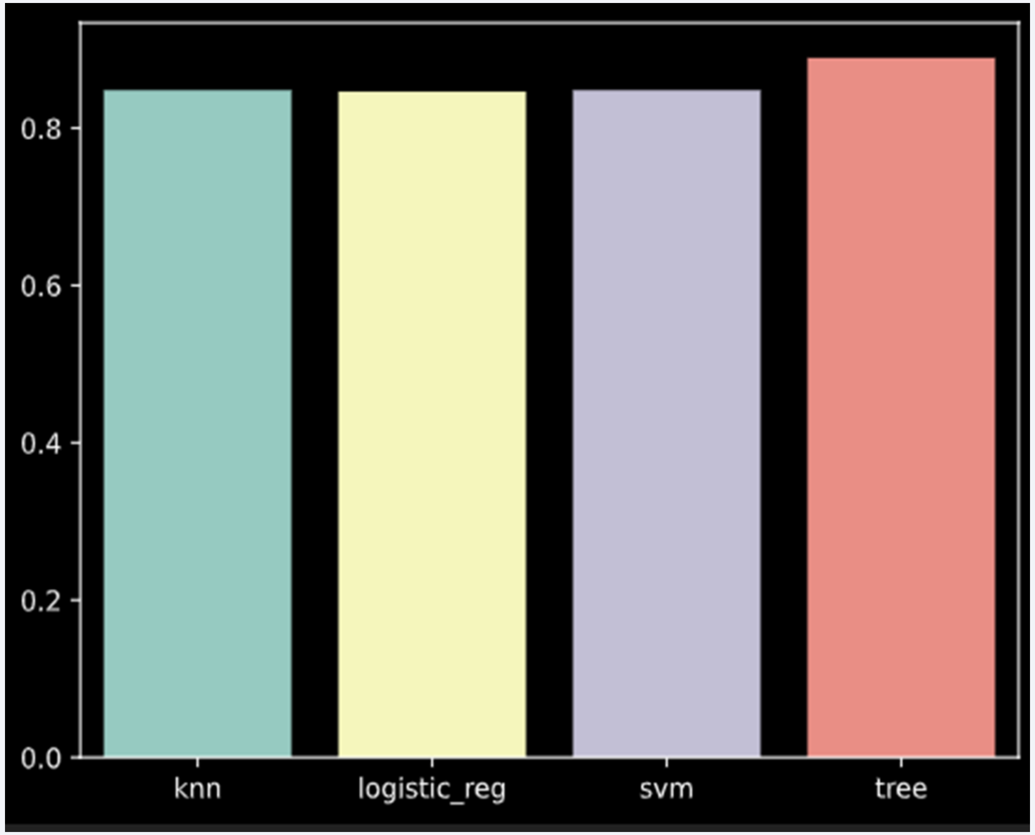
Section 5

Predictive Analysis (Classification)

https://github.com/Teraces12/LearnerWithIBM

Classification Accuracy

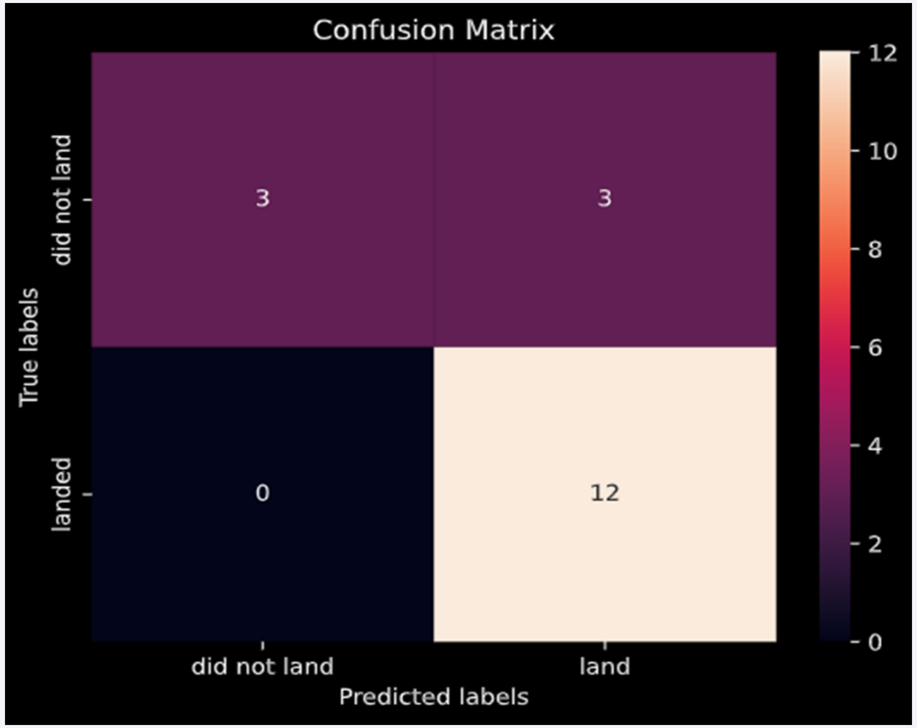
- From the chart, you can see that out of all the classification algorithms, the decision tree algorithm had the best accuracy score.



https://github.com/Teraces12/LearnerWithIBM

Confusion Matrix

- The confusion matrix was plotted for the best performing model (Decision tree classifier) showing the True Positives, False positives, True Negatives and False Negatives.
- True Positives: 12
- True Negatives: 3
- False Positives: 3
- False Negatives: 0

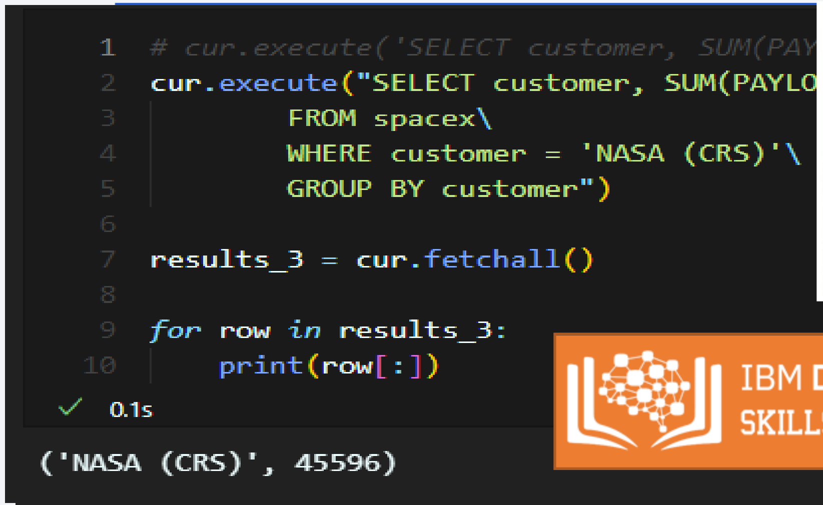


https://github.com/Teraces12/LearnerWithIBM

Conclusions

- The Decision tree classifier is the best machine learning algorithm for this task.
- The launch success rate started to increase from 2013 until 2020.
- Orbits ES-L1, GEO, HEO, SSO, and VLEO had the highest success rates.
- Notably, KSC LC-39A had the most successful launches among all sites.
- Additionally, it's observed that the larger the flight amount at a launch site, the greater the success rate at that specific launch site.

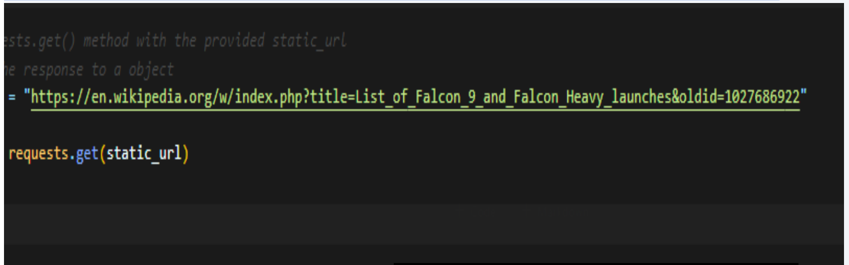
Appendix



```
# cur.execute('SELECT customer, SUM(PAY
cur.execute("SELECT customer, SUM(PAYLO
        FROM spacex\
        WHERE customer = 'NASA (CRS)'\
        GROUP BY customer")

results_3 = cur.fetchall()

for row in results_3:
    print(row[:])
```

✓ 0.1s

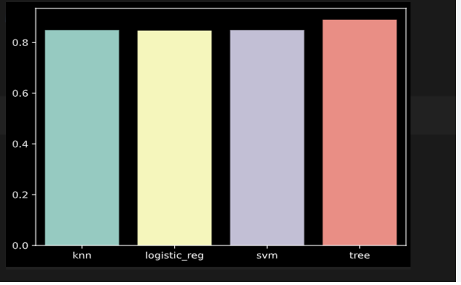
```
('NASA (CRS)', 45596)
```

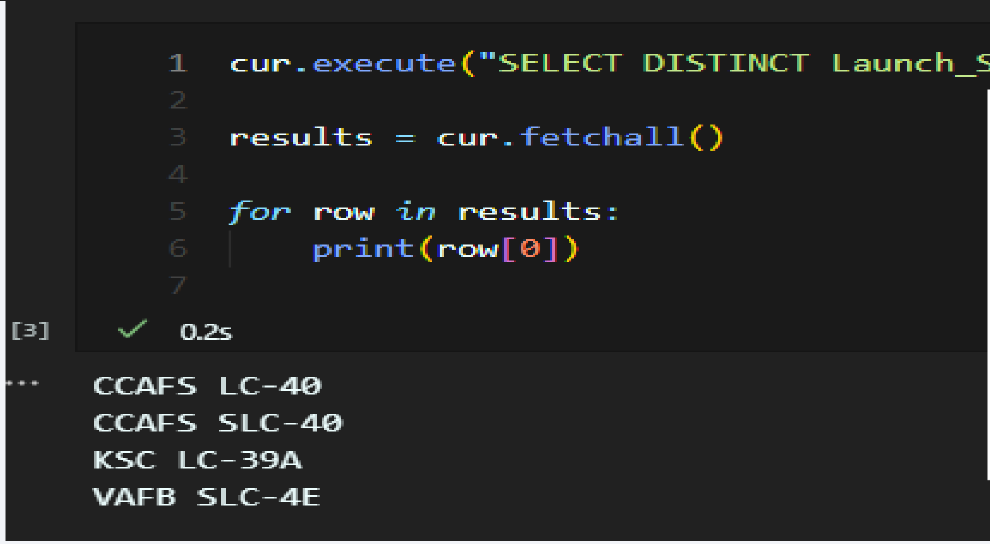


```
ests.get() method with the provided static_url
he response to a object
= "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"

requests.get(static_url)
```

```
ifulSoup() to create a BeautifulSoup object from a
soup = BeautifulSoup(response.text, 'html.parser')
```

```
# Use soup.title attribute
soup.title

html_tables = soup.find_all('table')
```

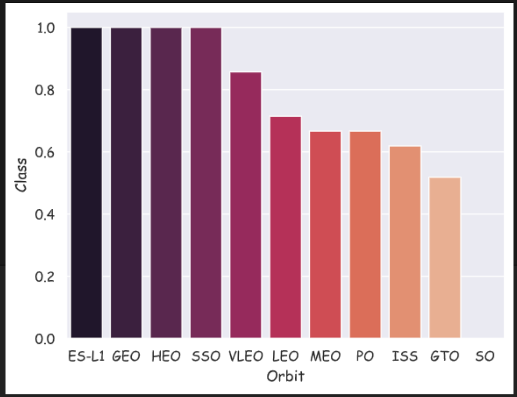


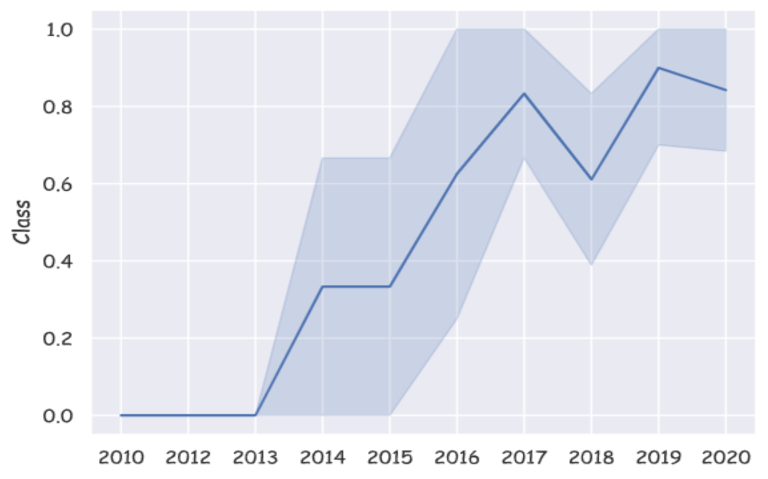


```
cur.execute("SELECT DISTINCT Launch_Site FROM ..spacex")

results = cur.fetchall()

for row in results:
    print(row[0])
```

[3] ✓ 0.2s

```
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E
```





https://github.com/Teraces12/LearnerWithIBM

IBM Developer
SKILLS NETWORK

Thank you!

https://github.com/Teraces12/LearnerWithIBM