IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Ruby Kalha>
<30/9/2022>

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary

- Summary of methodologies:
  - Data Collection through API,Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL and Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction

- Summary of all results
  - Exploratory Data Analysis (EDA) results
  - Data Analysis with screenshots of visualizations
  - Predictive Analytics result

Introduction

- Project background and context:

  Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. Our goal is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers:
  - What factors ensure a successful landing of the rocket?
  - The relationship between various features that determine the success rate of a landing.
  - What conditions need to be in place to achieve the best results?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

  Data was collected using:

  - SpaceX API
  - web scraping from Wikipedia.

- Perform data wrangling
  - One-hot encoding (to categorical features)
  - Dropping irrelevant columns
- Perform exploratory data analysis (EDA) using visualization and SQL:
  - Scatter Plots
  - Bar Graphs

- Perform interactive visual analytics using:
  - Folium
  - Plotly Dash
- Perform predictive analysis using classification models
  - Build, tune and evaluate classification models

Data Collection

- Data Collection meaning:

  It is the procedure of collecting, measuring and analyzing accurate insights for research using standard validated techniques to answer questions and evaluate outcomes.

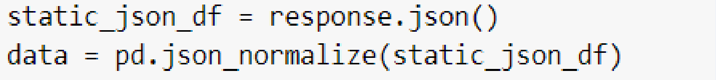
- Data Collection basic steps:

  1. Collect data from API and web pages.
  2. Create dataframe of it.
  3. Clean and filter the dataframe.
  4. Export to flat file (csv).

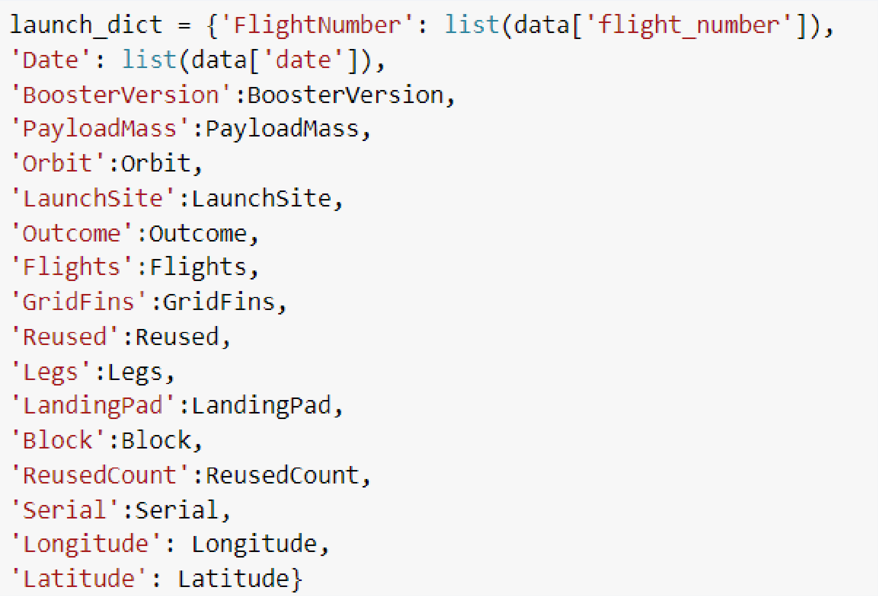
Data Collection – SpaceX API

Our objective is to extract the launch records as HTML table, parse the table, and convert it to a pandas dataframe for future analysis.

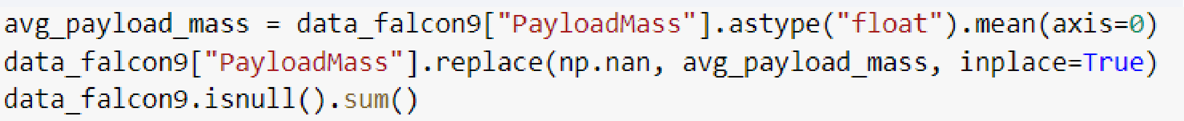
- Collect the data using get request to the SpaceX API.
- Convert the response content as a Json file using .json() function call and turn it into a pandas dataframe using.json_normalize().
- Clean the data, check for missing values and fill in missing values where necessary.
- Ceate the dataframe using dictionary.
- Filter the dataframe and export it to a CSV file.
- Notebook: https://github.com/rubylouai/final/blob/main/Collecting%20the%20Data-%20Collection%20API%20Lab.ipynb.ipynb



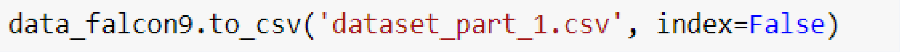
```
static_json_df = response.json()
data = pd.json_normalize(static_json_df)
```



```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```



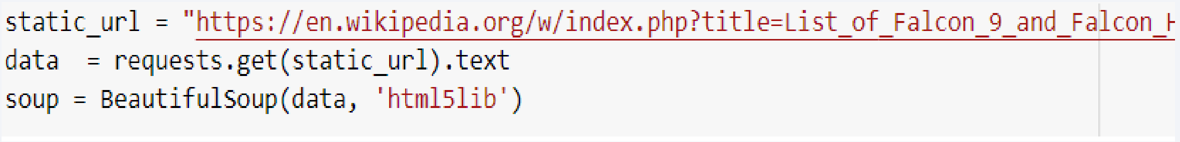
```
avg_payload_mass = data_falcon9["PayloadMass"].astype("float").mean(axis=0)
data_falcon9["PayloadMass"].replace(np.nan, avg_payload_mass, inplace=True)
data_falcon9.isnull().sum()
```



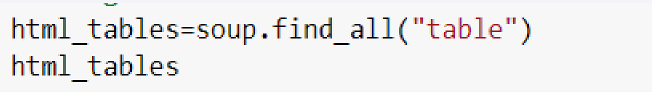
```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection - Scraping

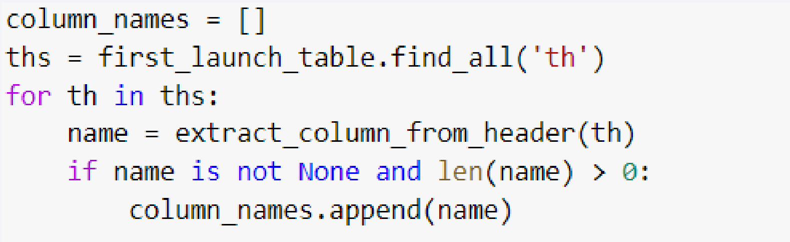
- Apply web scrapping to Falcon 9 launch records from Wikipedia with BeautifulSoup.
- Get response from HTML
- Create a BeautifulSoup object and sparse tables.
- Create dictionary with the column names and appending data to them.
- Convert dictionary to dataframe and export it to csv file.
- Notebook: https://github.com/rubylouai/final/blob/main/webscraping%20(1).ipynb



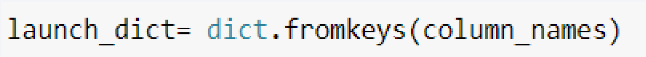
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_H
data  = requests.get(static_url).text
soup = BeautifulSoup(data, 'html5lib')
```



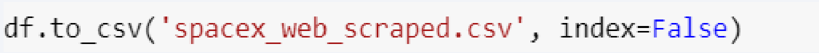
```
html_tables=soup.find_all("table")
html_tables
```



```
column_names = []
ths = first_launch_table.find_all('th')
for th in ths:
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0:
        column_names.append(name)
```



```
launch_dict= dict.fromkeys(column_names)
```

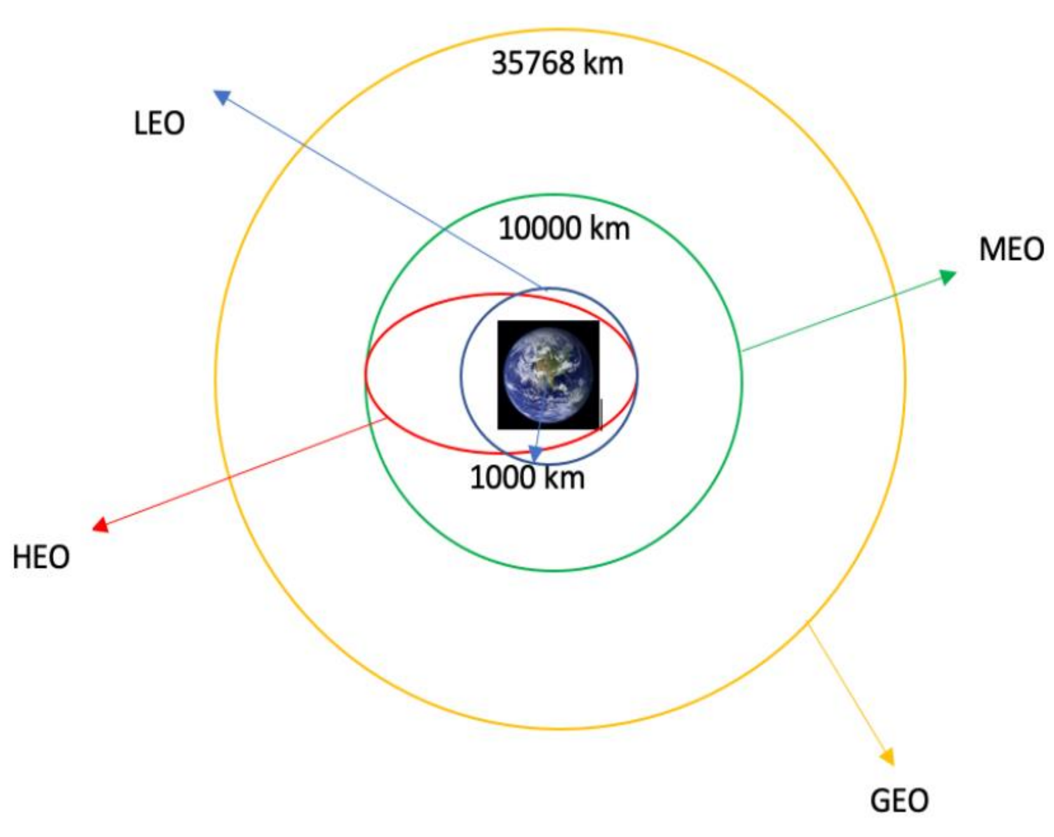


```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

- Data Wrangling Basic steps:
  - Load Data.
  - Make Dataframe from it.
  - Clean Data.
  - Covert it to Boolean values.
  - Export to flat file.
  - Determine training labels.

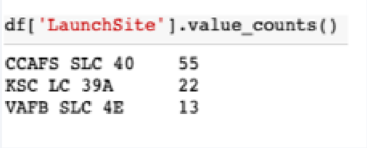
  - Notebook: https://github.com/rubylouai/final/blob/main/Data%20Wrangling%20-%20EDA%20lab.ipynb



Data Wrangling

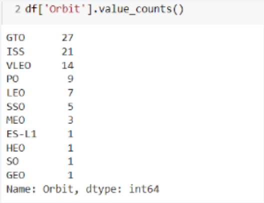
- Data Wrangling steps:

1. Calculate the number of launches at each site.



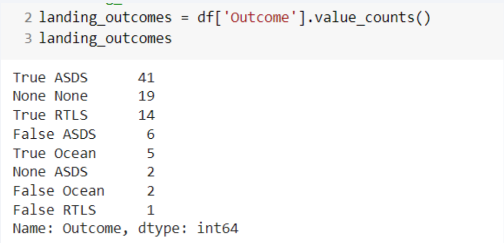
```
df['LaunchSite'].value_counts()

CCAFS SLC 40     55
KSC LC 39A       22
VAFB SLC 4E      13
```

2. Calculate the number of launches at each orbit.



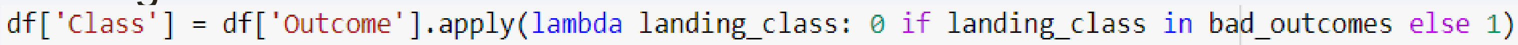
```
df['Orbit'].value_counts()

GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

3. Calculate the number and occurrence of mission outcomes per orbit type.



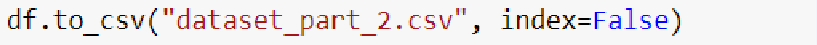
```
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes

True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
None ASDS       2
False Ocean     2
False RTLS      1
Name: Outcome, dtype: int64
```

4. Create landing outcome label from outcome column



```
df['Class'] = df['Outcome'].apply(lambda landing_class: 0 if landing_class in bad_outcomes else 1)
```

5. export the results to CSV file.

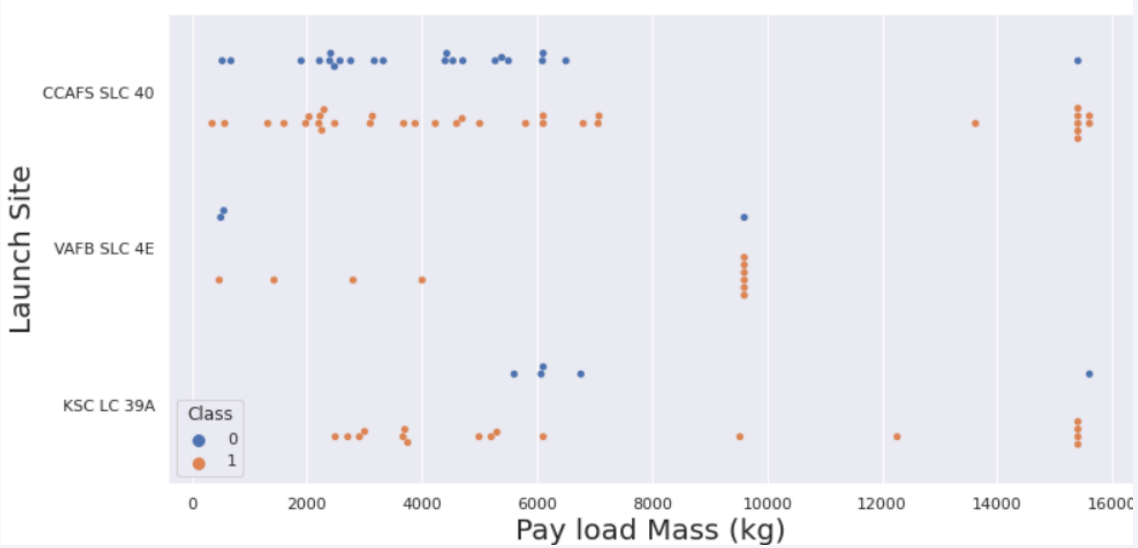


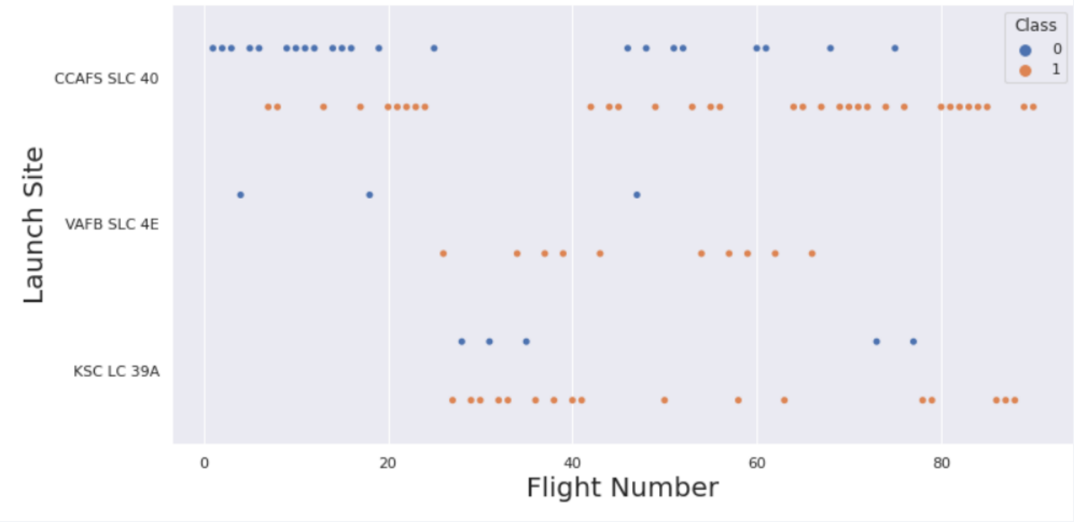
```
df.to_csv("dataset_part_2.csv", index=False)
```

EDA with Data Visualization

- Scatter Plots between:

Payload and Flight number/ Flight number and launch site/ payload and launch site / Flight number and Orbit type/ Payload and Orbit type





EDA with Data Visualization

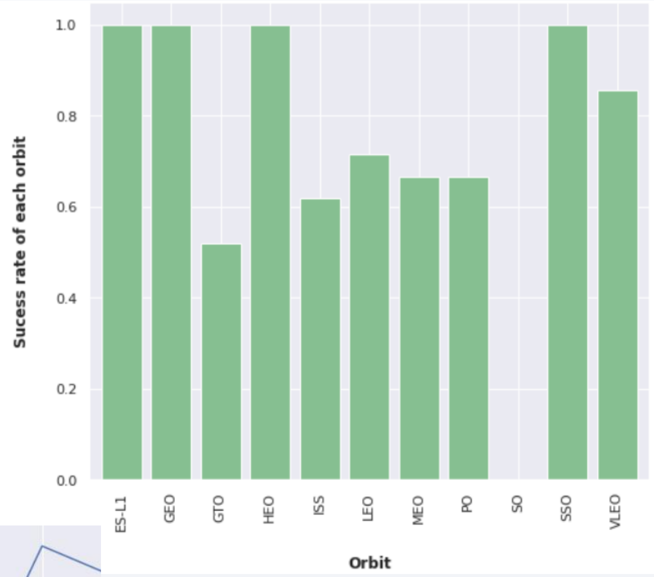
- Bar Graph between:

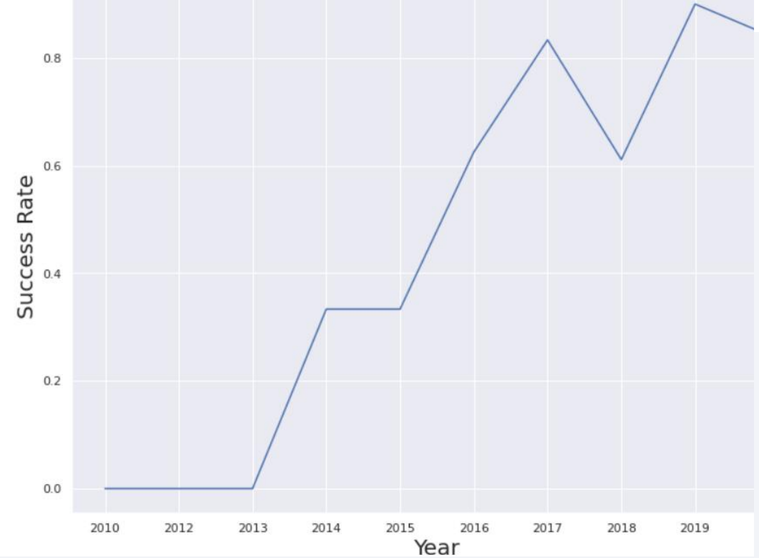
  Success Rate and Orbit type.

- Line Graph:

  Applied to Launch Success yearly trend.

- Notebook: https://github.com/rubylouai/final/blob/main/EDA%20with%20Visualization%20lab.ipynb





EDA with SQL

- **SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relational database.** Here we loaded the SpaceX dataset into a PostgreSQL database.
- We performed SQL queries to get insight from the data:
  - Display the names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- Notebook: https://github.com/rubylouai/final/blob/main/Exploratory%20Data%20Analysis%20Using%20SQL.ipynb

Build an Interactive Map with Folium

- **Folium makes it easy to visualize data that's been manipulated in Python on an interactive leaflet map.**
- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities.
- Notebook: https://github.com/rubylouai/final/blob/main/Interactive%20Visual%20Analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- Notebook: https://github.com/rubylouai/final/blob/main/plotly.py

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- Notebook: https://github.com/rubylouai/final/blob/main/Machine%20Learning%20Prediction%20Lab.ipynb

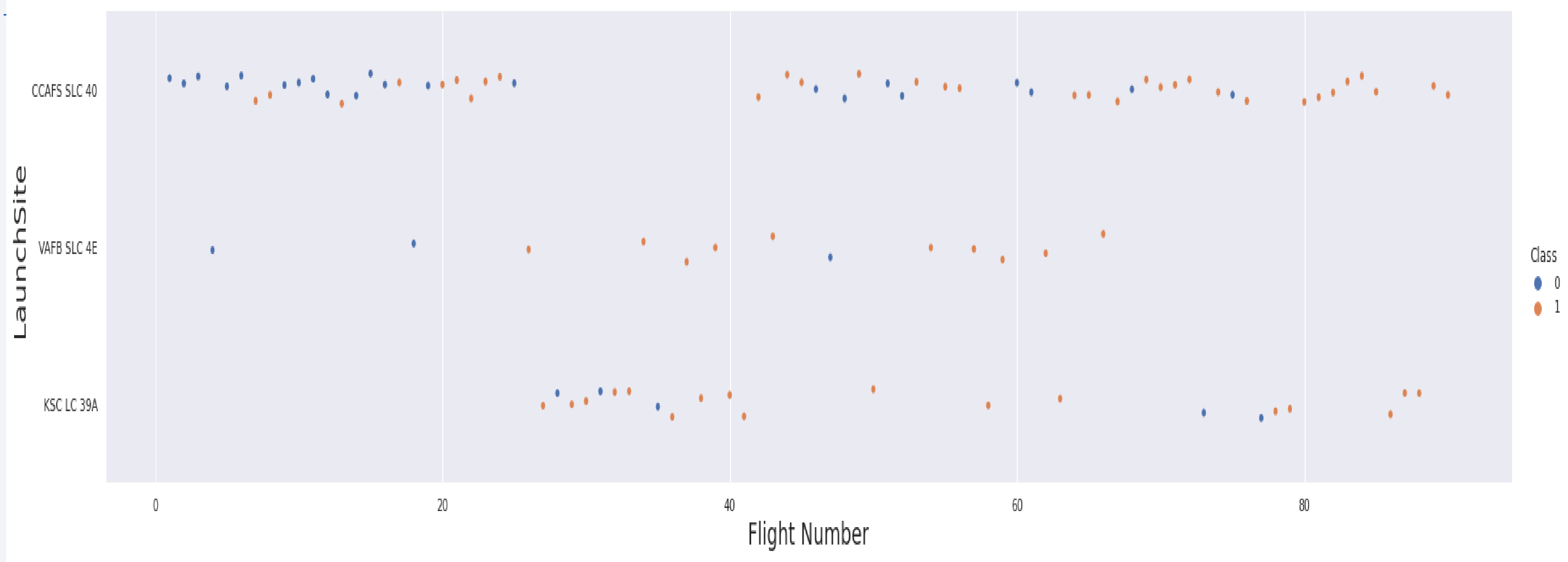
Results

- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

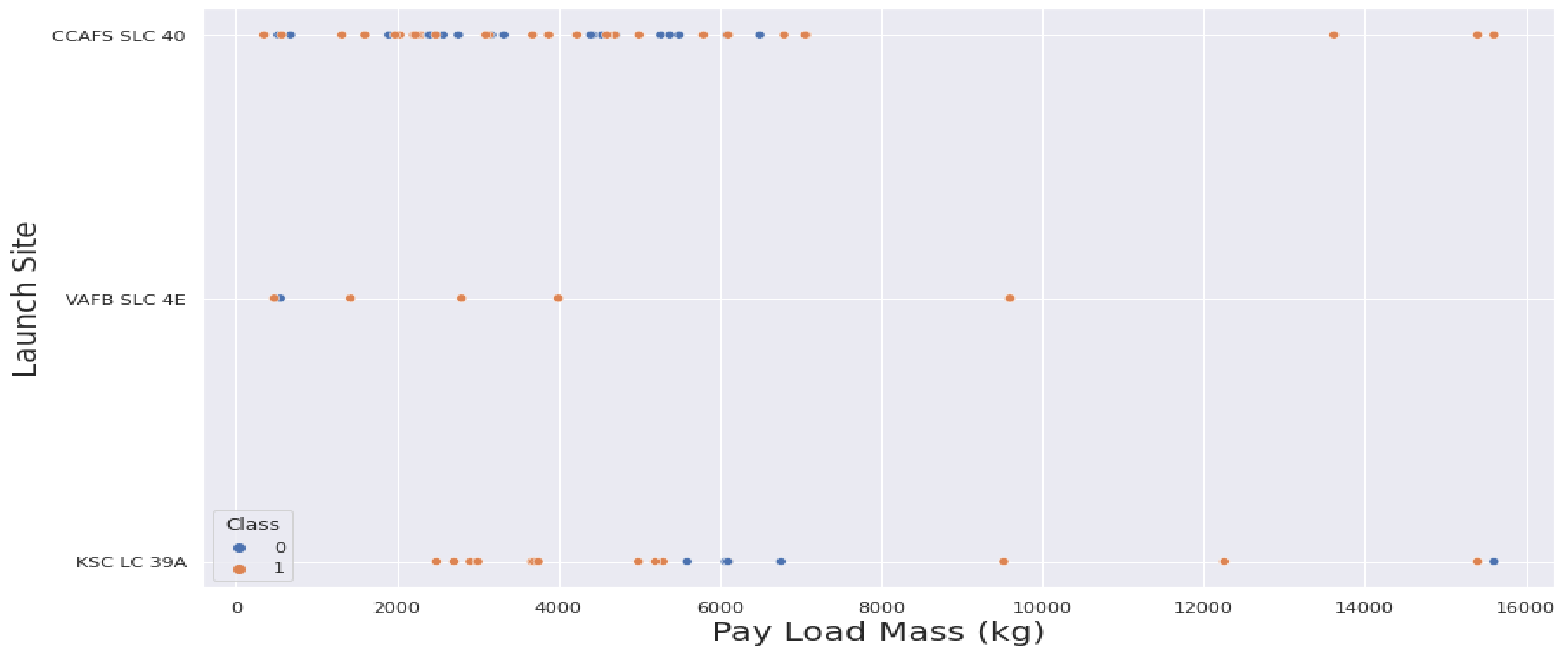
Insights drawn from EDA

Flight Number vs. Launch Site



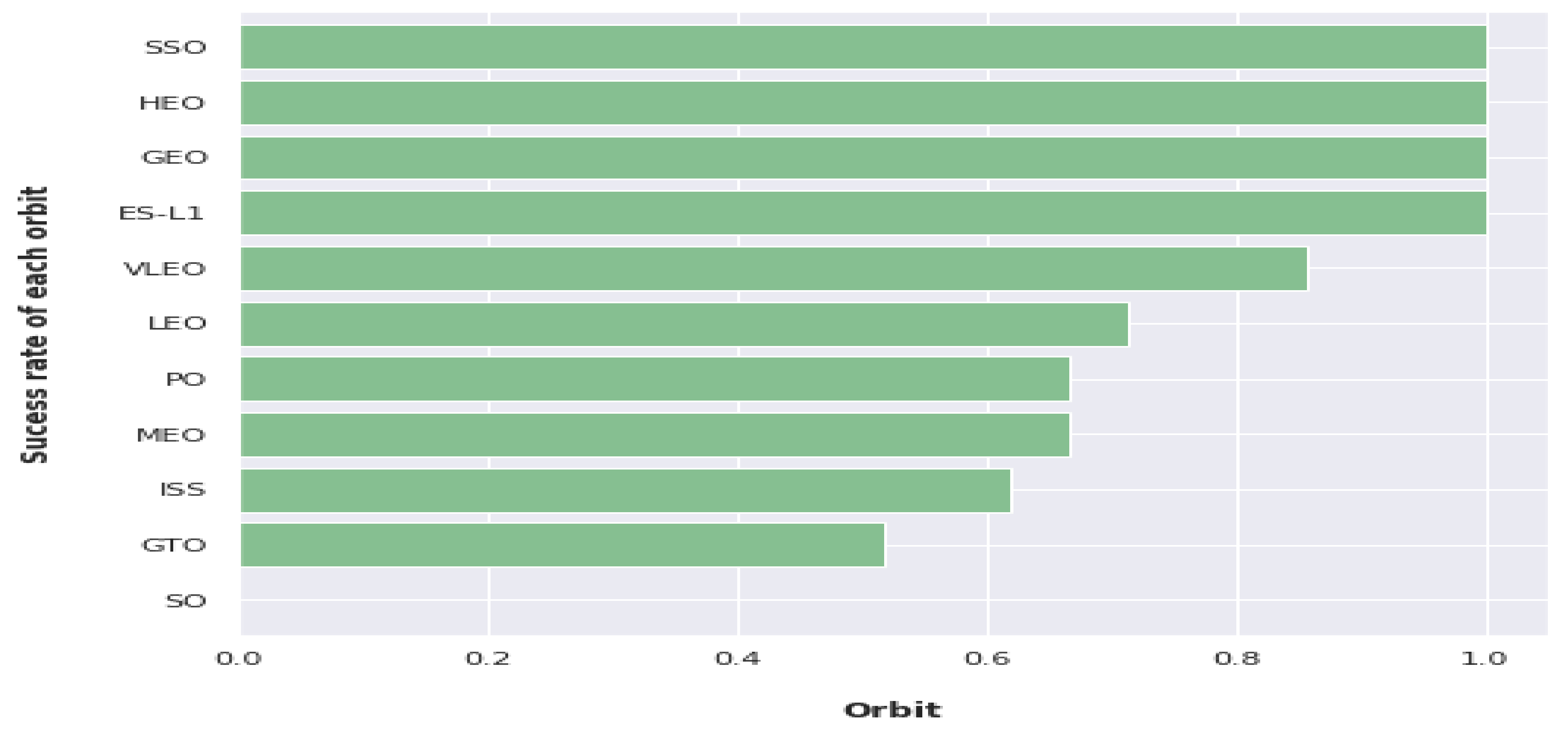
The higher the Flight number the success rate for the Rocket increases

Payload vs. Launch Site



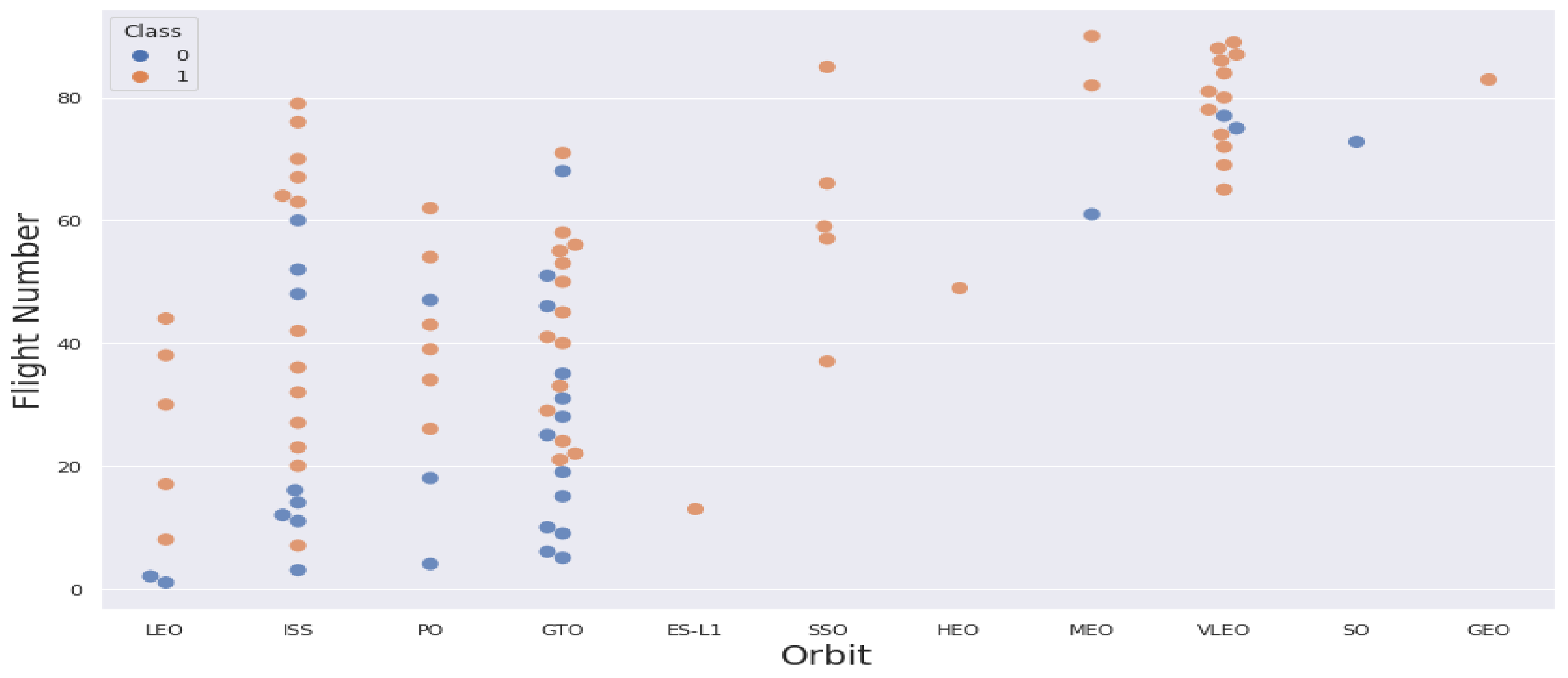
The greater the payload mass (greater than 7000 kg) higher the success rate of the rocket. But there’s no clear pattern to take a decision if the launch site is dependent on Pay Load Mass for a successful launch.

Success Rate vs. Orbit Type



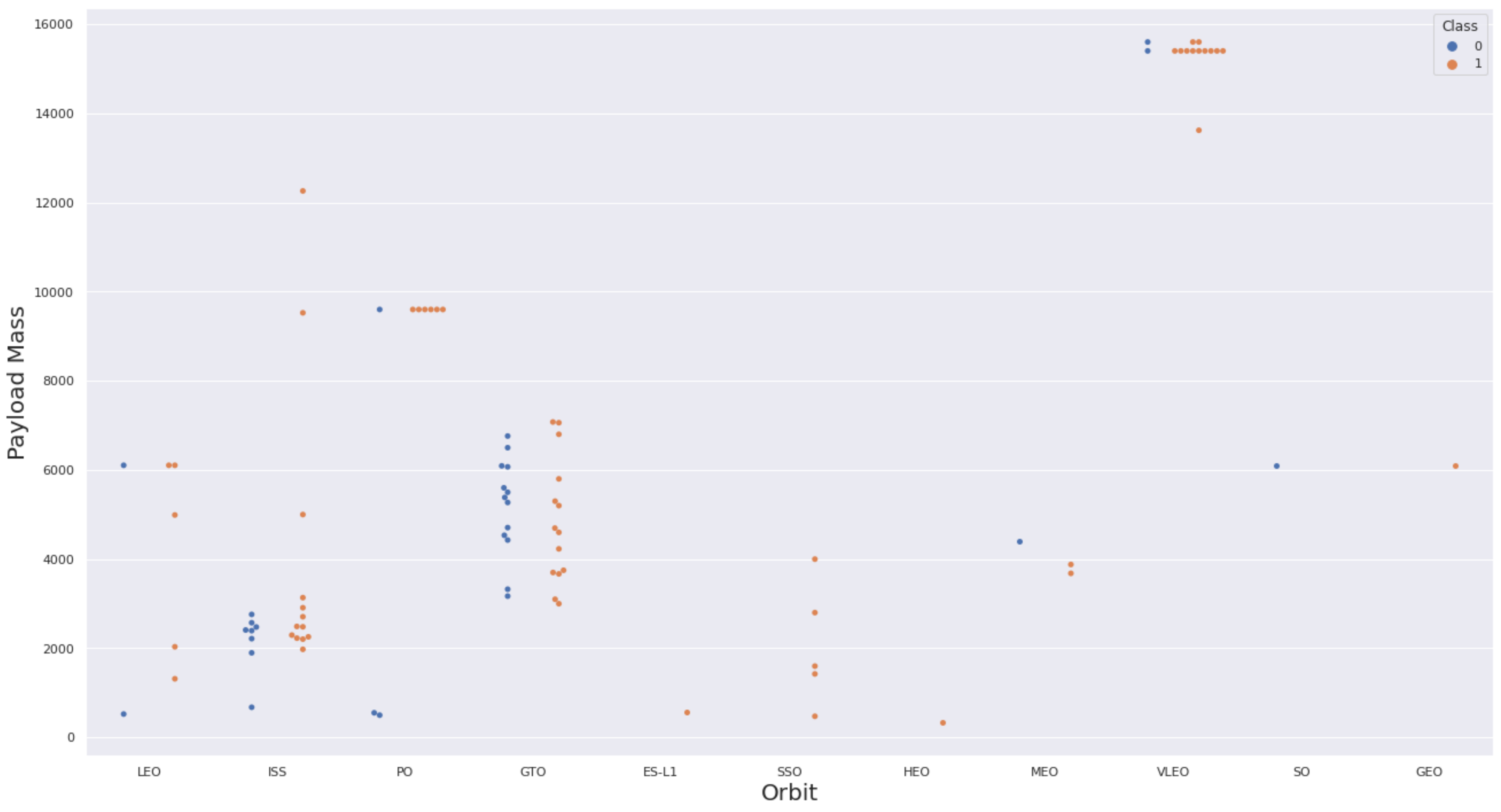
ES-L1 ,GEO,HEO,SSO has the highest Success rates

Flight Number vs. Orbit Type



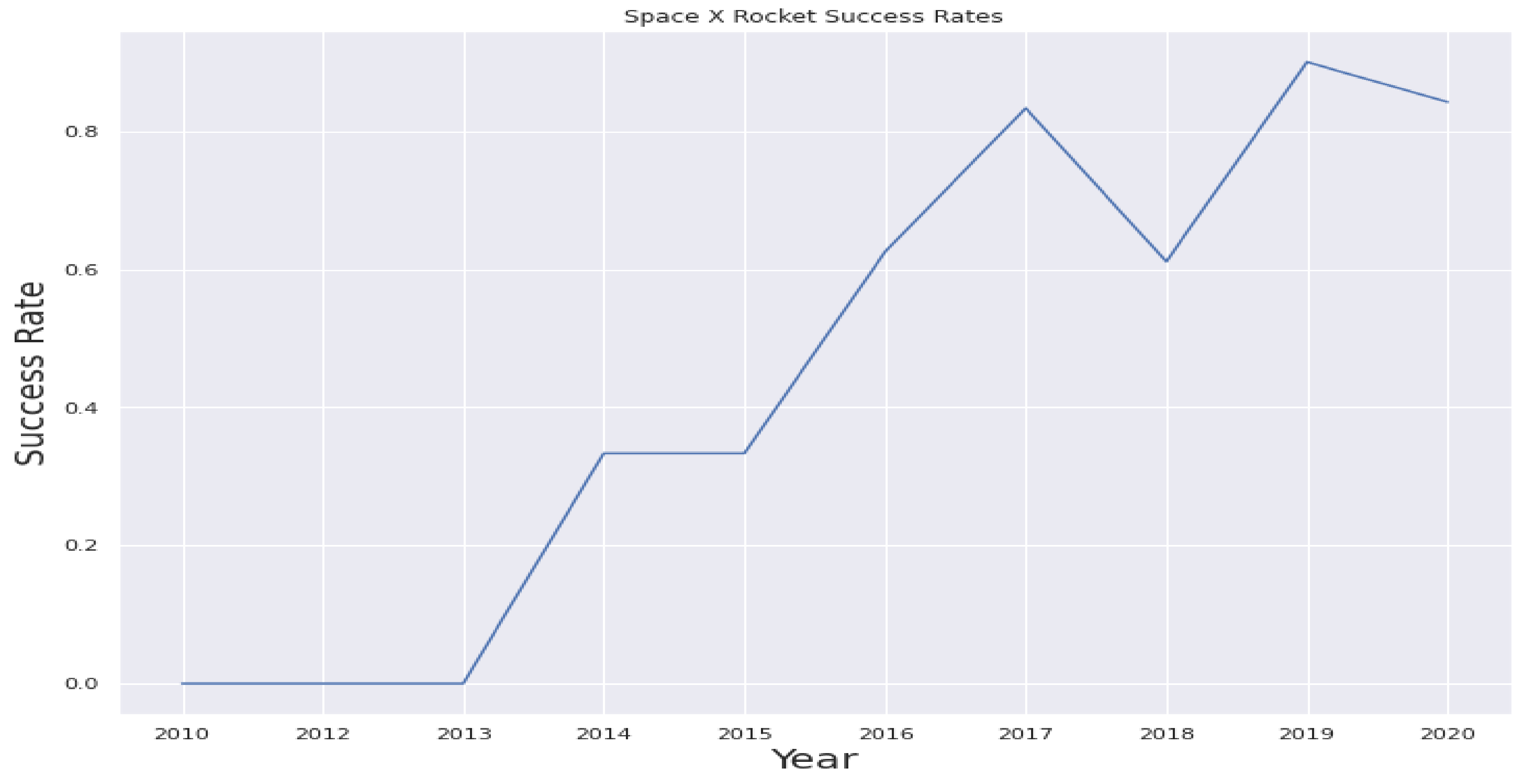
- We see that for Leo orbit the success increases with the number of the flights
- On the other hand ,there seems to be no relationship between flight number and GTO orbit

Payload vs. Orbit Type



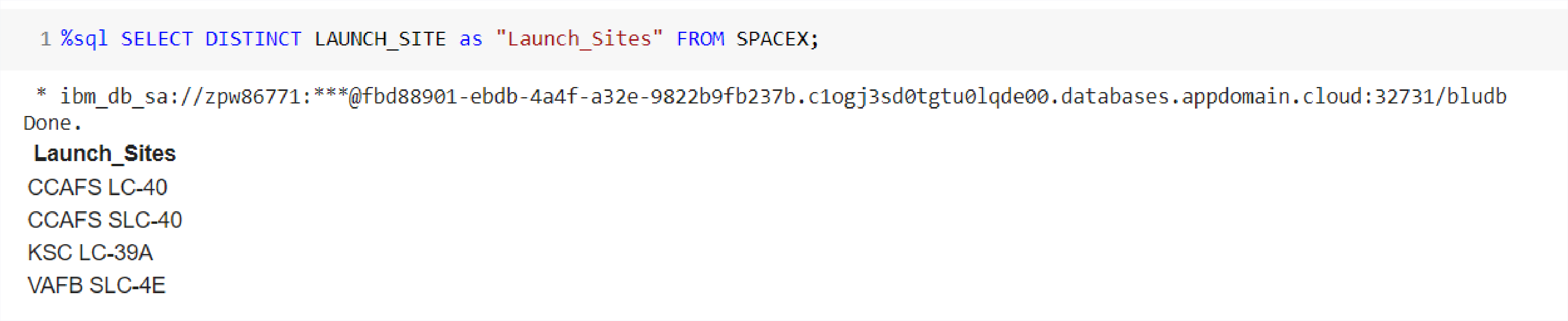
- We observe that heavy payloads have a negative influence on MEO,GTO ,VLEO Orbits
Positive on LEO,ISS Orbit

Launch Success Yearly Trend



We can observe that the success rate since 2013 kept increasing relativity through there is a slight dip after 2019

All Launch Site Names



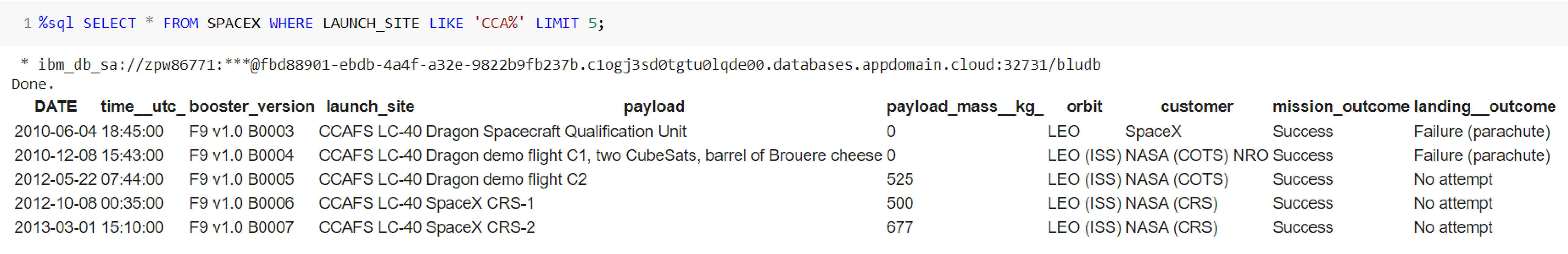
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEX;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

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

We used the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

Launch Site Names Begin with 'CCA'



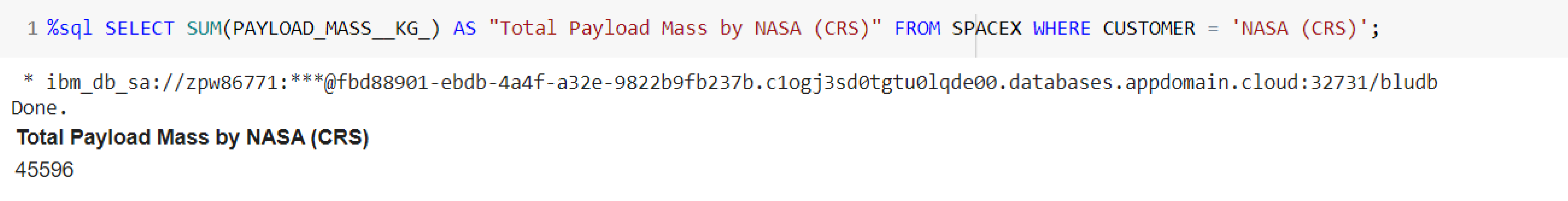
```
%sql SELECT * FROM SPACEX WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

We used the query above to display 5 records where launch sites begin with \`CCA\`

Total Payload Mass



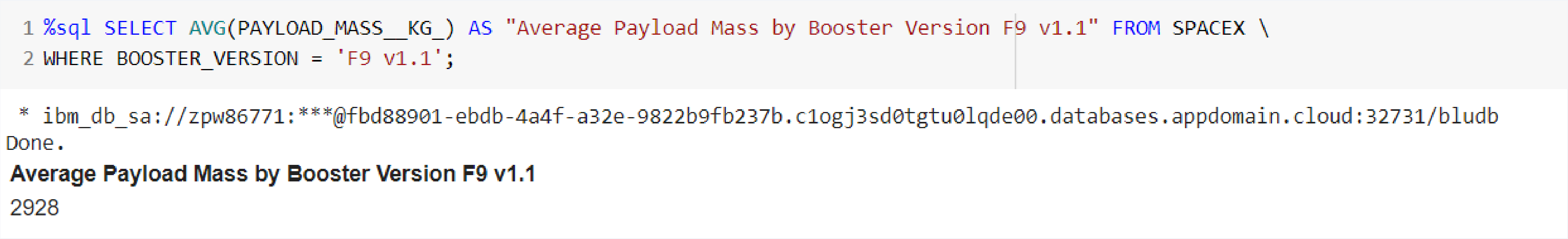
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)" FROM SPACEX WHERE CUSTOMER = 'NASA (CRS)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Total Payload Mass by NASA (CRS)
45596

We calculated the total payload carried by boosters from NASA as 45596

Average Payload Mass by F9 v1.1



```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster Version F9 v1.1" FROM SPACEX \
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

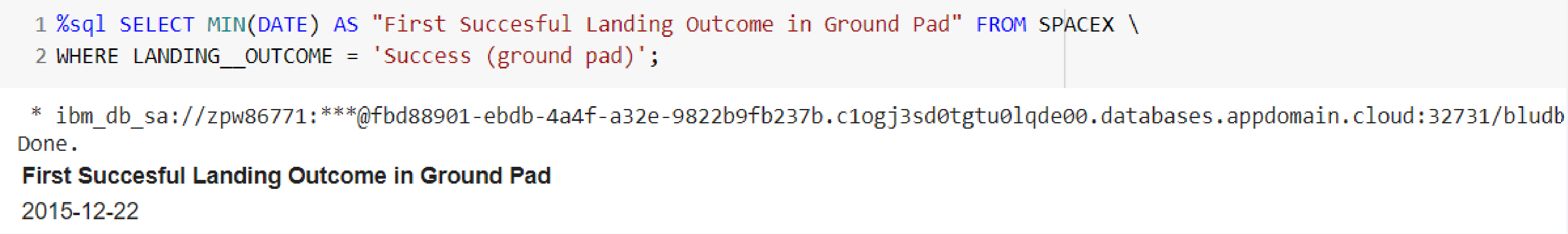
```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Average Payload Mass by Booster Version F9 v1.1

2928

We calculated the average payload mass carried by booster version F9 v1.1 as 2928

First Successful Ground Landing Date



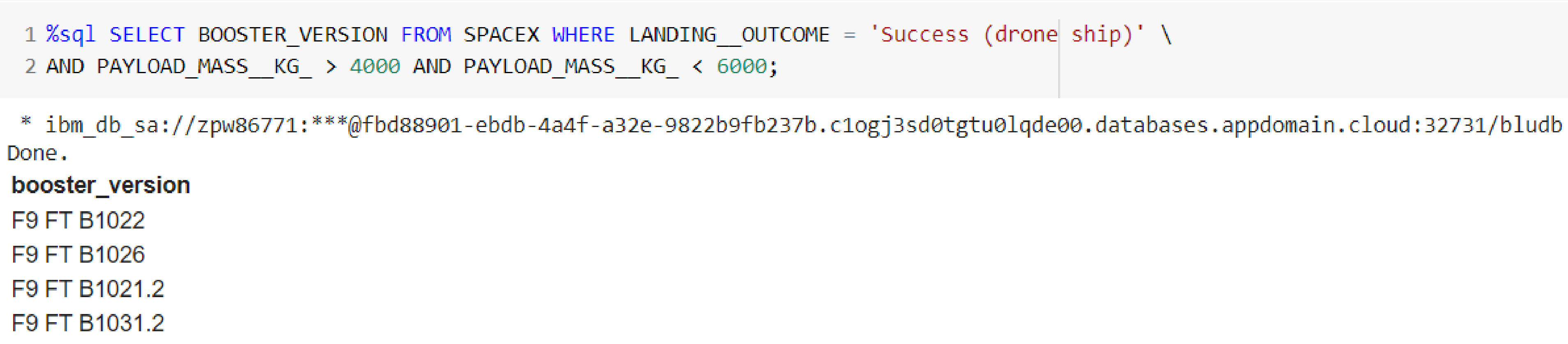
```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad" FROM SPACEX \
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

Successful Drone Ship Landing with Payload between 4000 and 6000



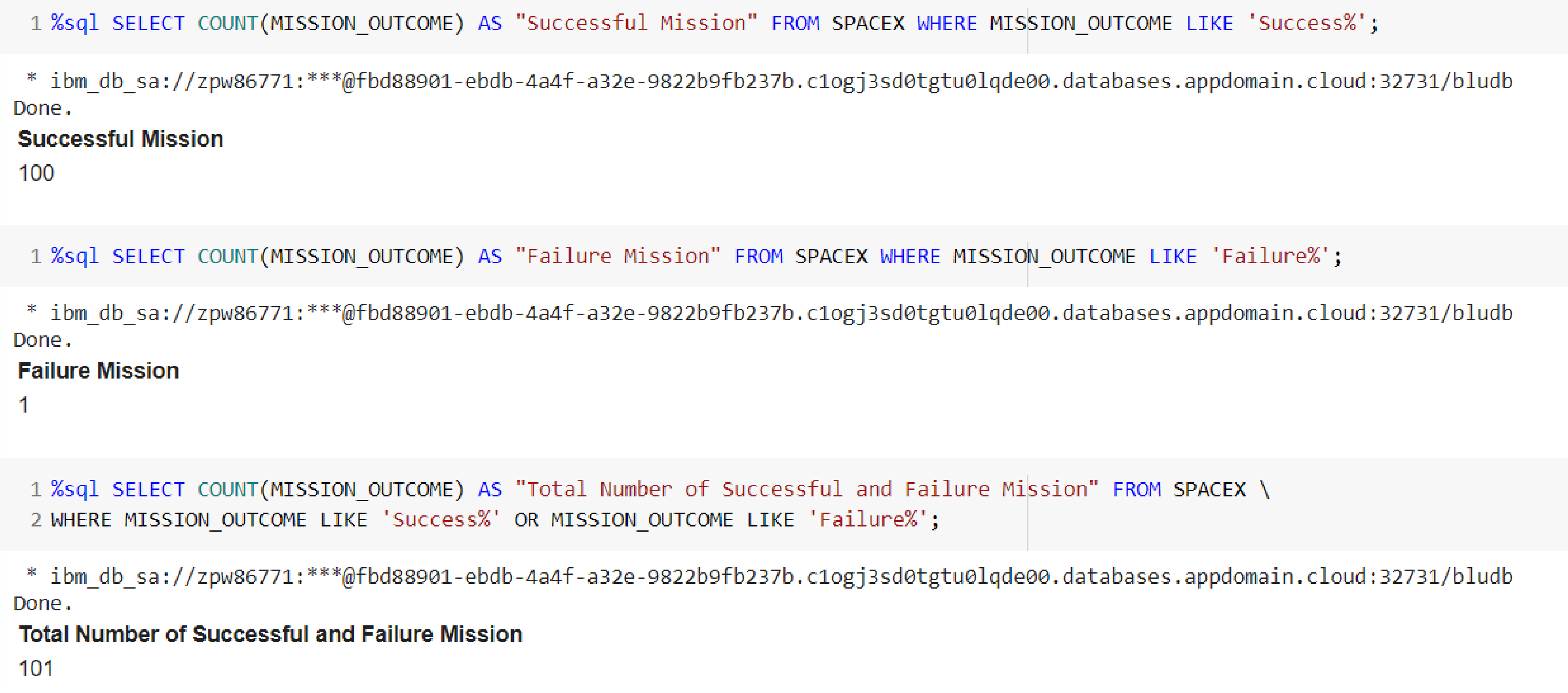
```
%sql SELECT BOOSTER_VERSION FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

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

We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes



```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Successful Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Success%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Successful Mission
100

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Failure Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Failure%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Failure Mission
1

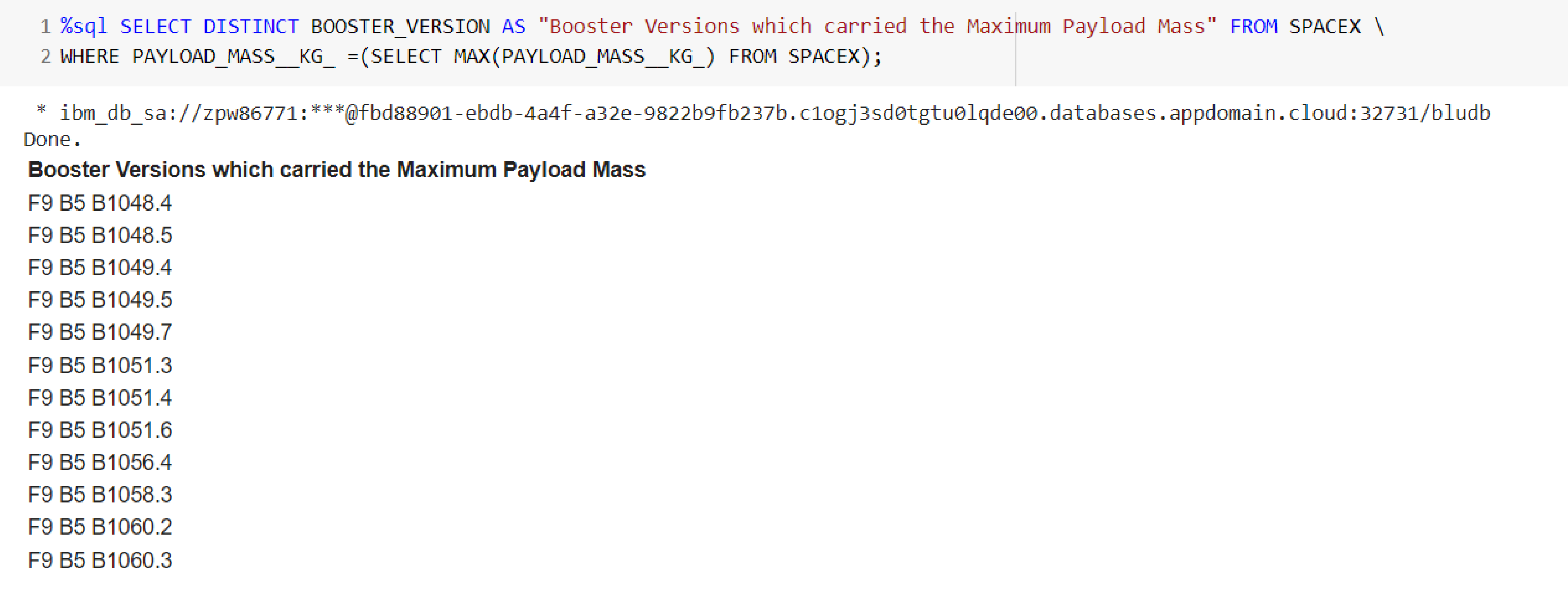
```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Total Number of Successful and Failure Mission" FROM SPACEX \
WHERE MISSION_OUTCOME LIKE 'Success%' OR MISSION_OUTCOME LIKE 'Failure%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Total Number of Successful and Failure Mission
101

We used wildcard like ‘%’ to filter for **WHERE** MissionOutcome was a success or a failure.

Boosters Carried Maximum Payload



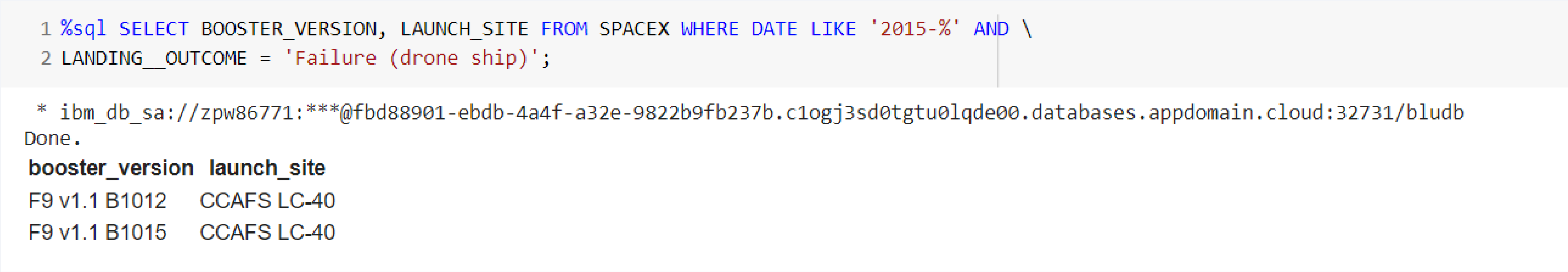
```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEX \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX);
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Booster Versions which carried the Maximum Payload Mass
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

- We determined the booster that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

2015 Launch Records



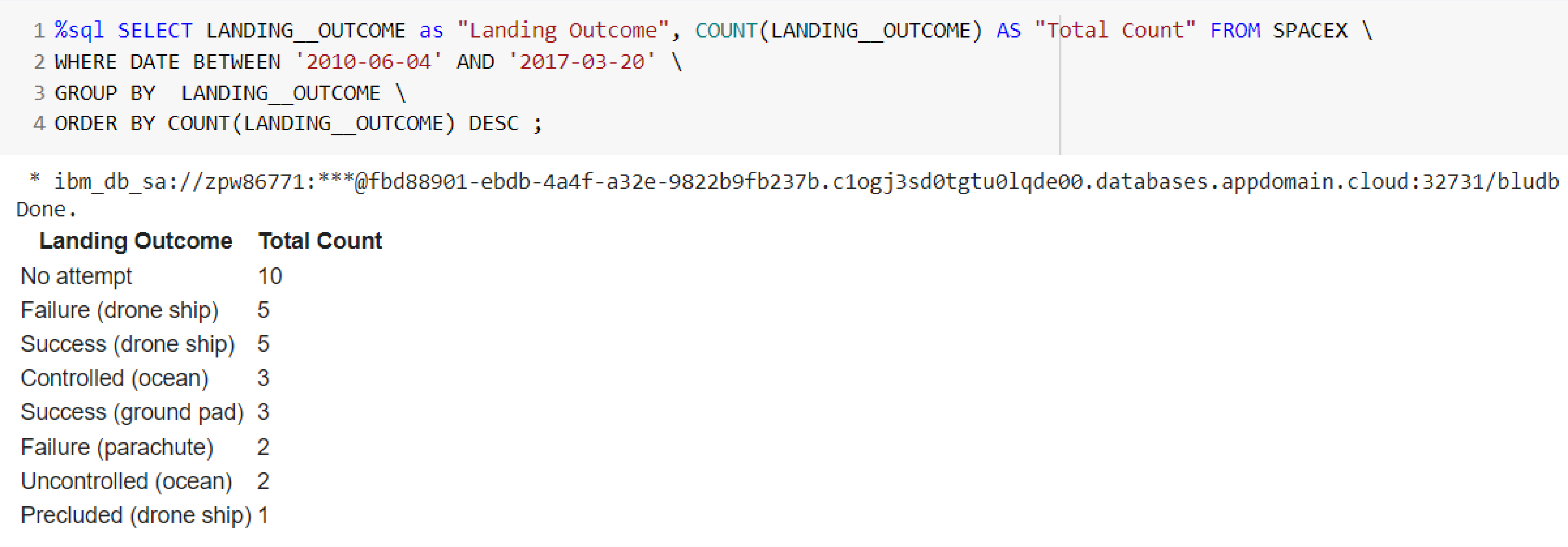
```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE DATE LIKE '2015-%' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

- We used a combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

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



```
%sql SELECT LANDING__OUTCOME as "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" FROM SPACEX \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  LANDING__OUTCOME \
ORDER BY COUNT(LANDING__OUTCOME) DESC ;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

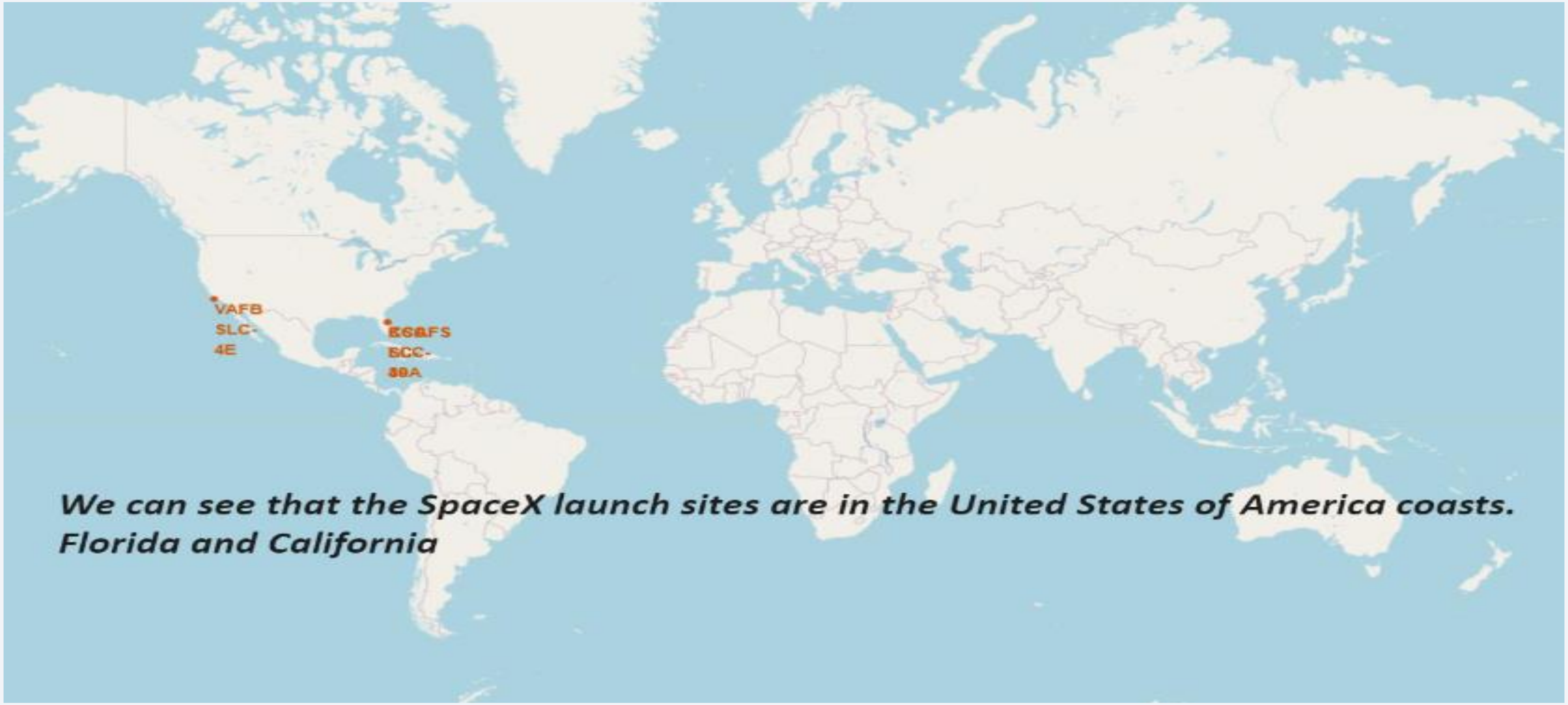
Landing Outcome	Total Count
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

Section 3

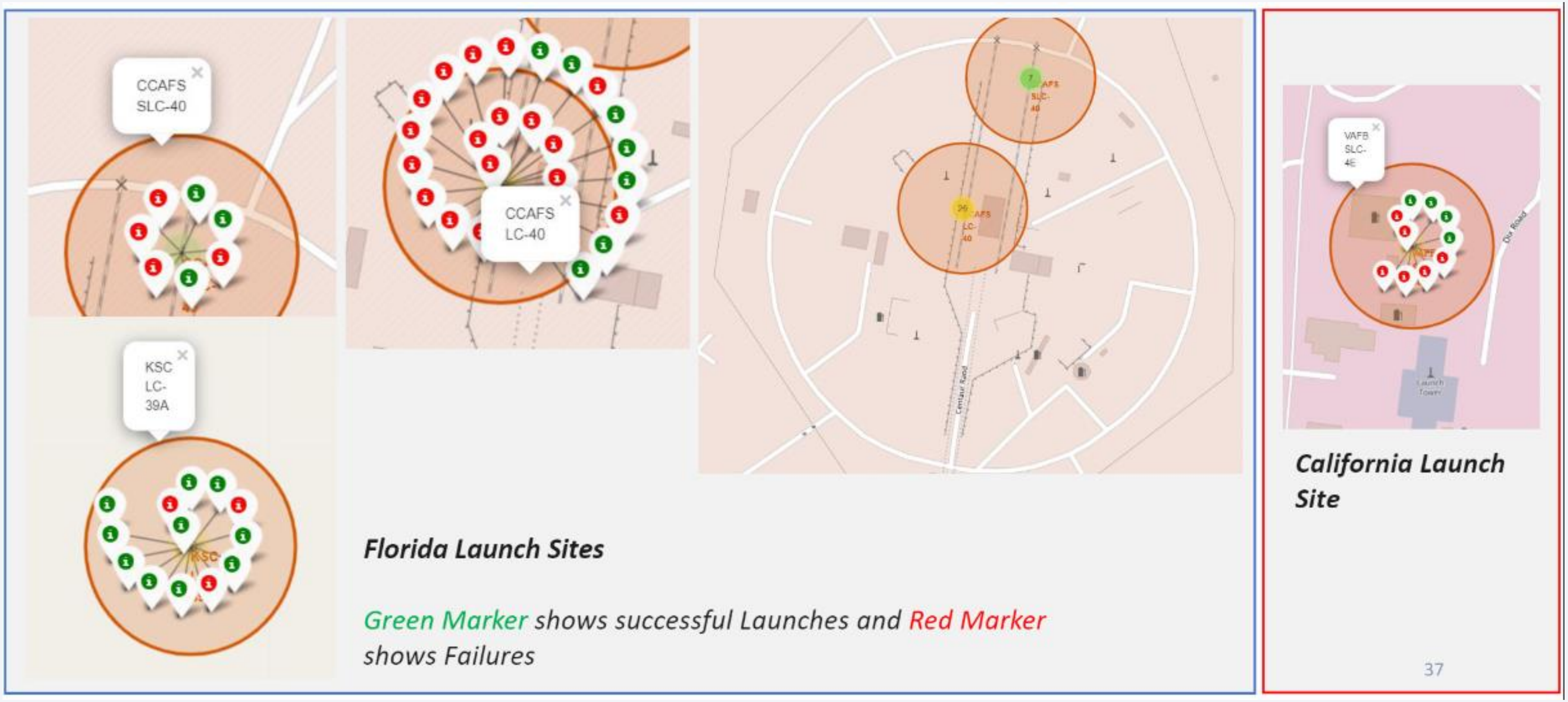
Launch Sites Proximities Analysis

All launch sites global map markers



We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

Markers showing launch sites with color labels



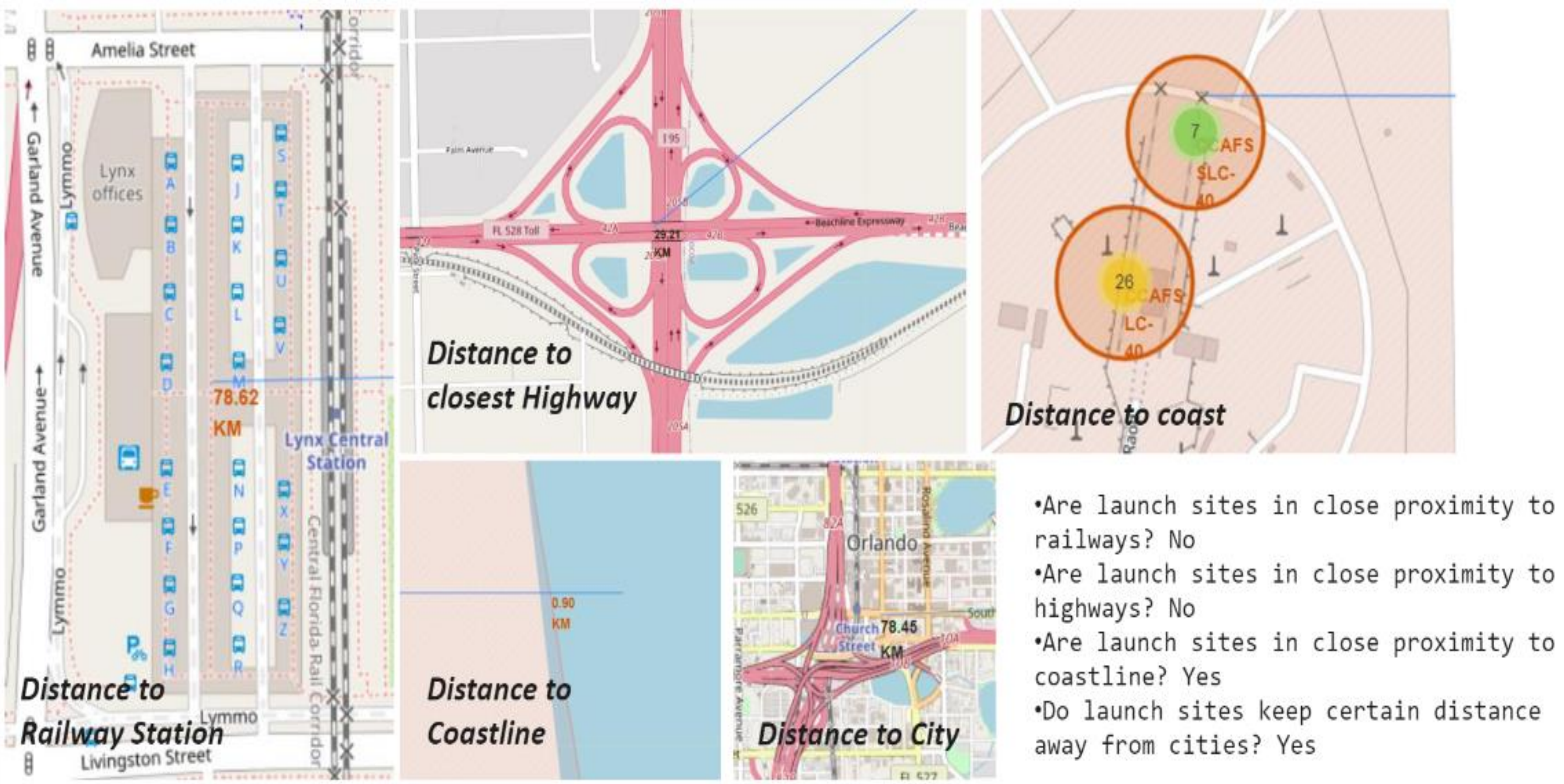
Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

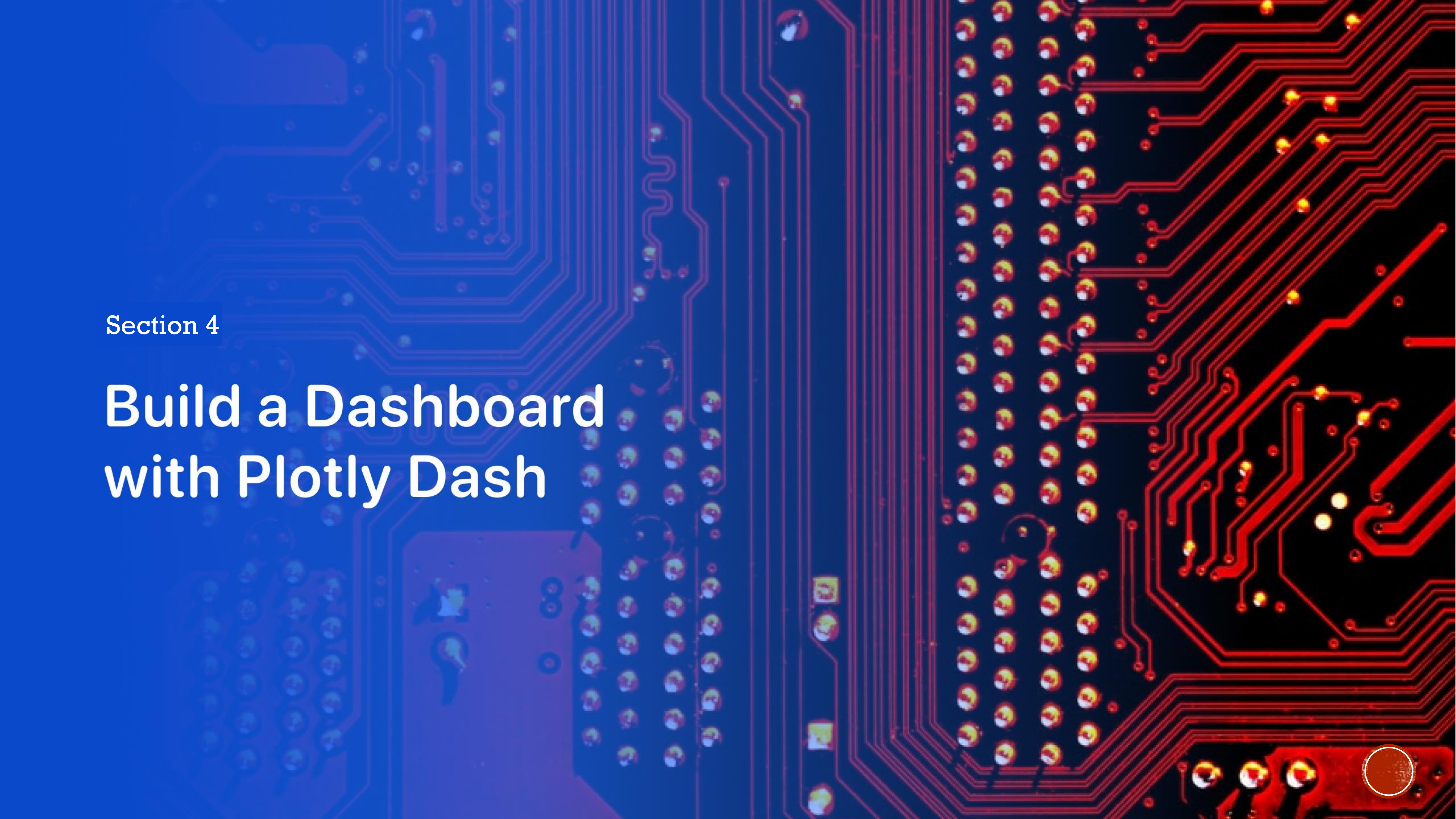
California Launch Site

37

Launch Site distance to landmarks



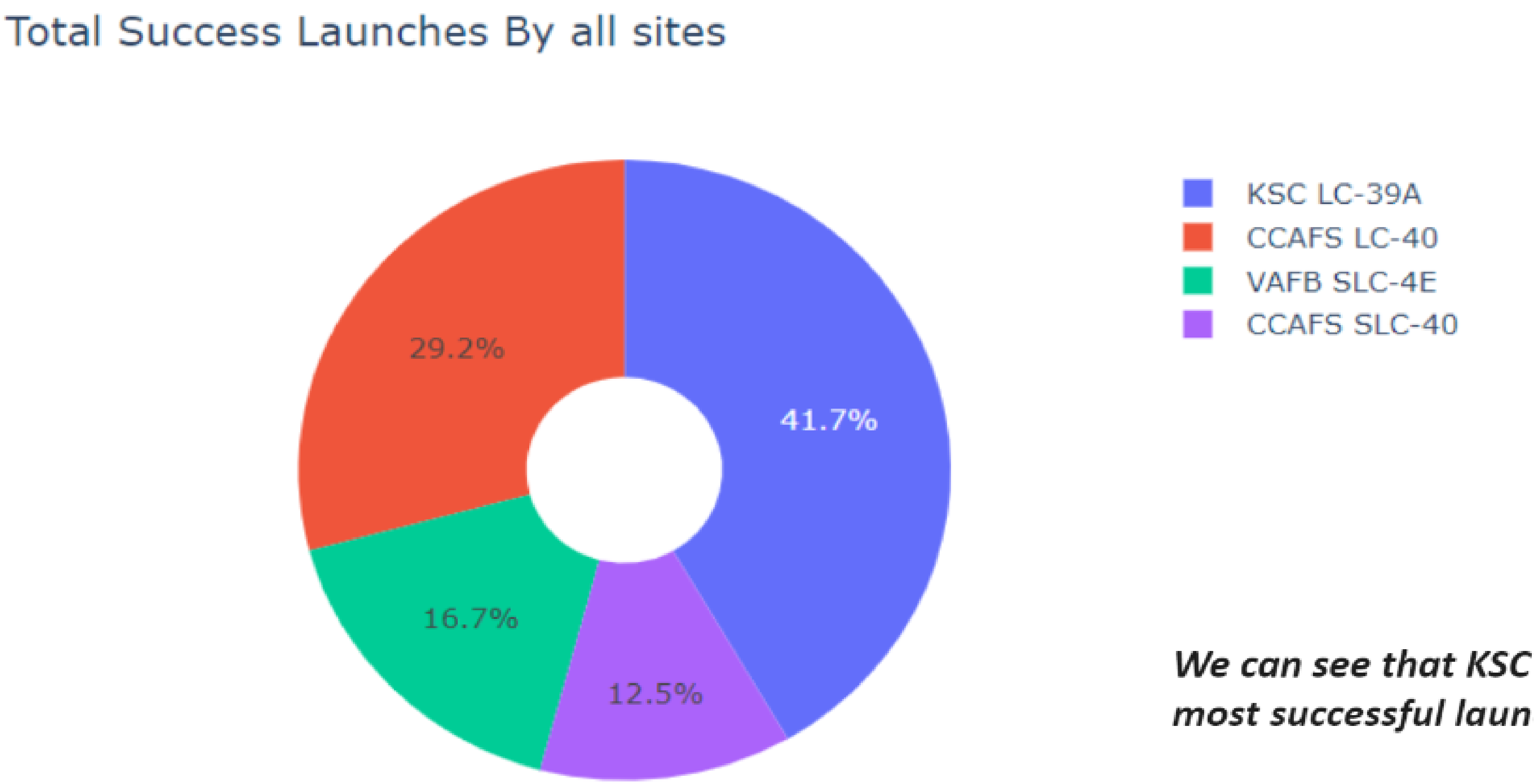
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes



Section 4

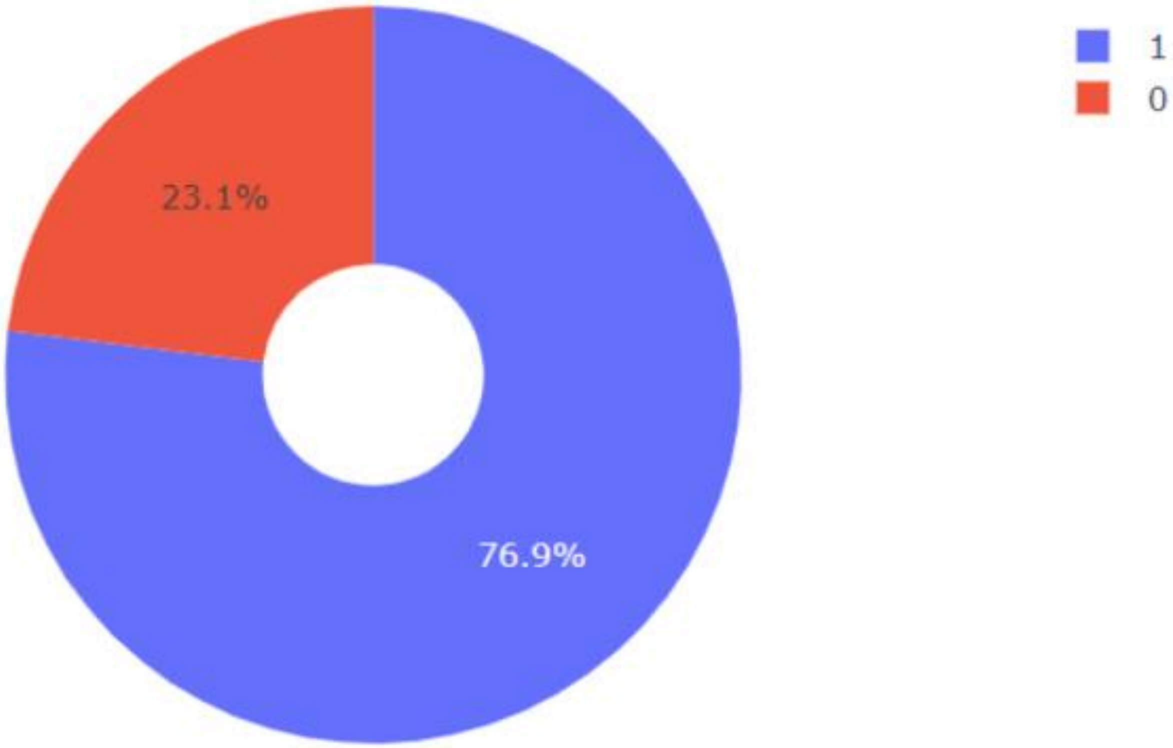
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



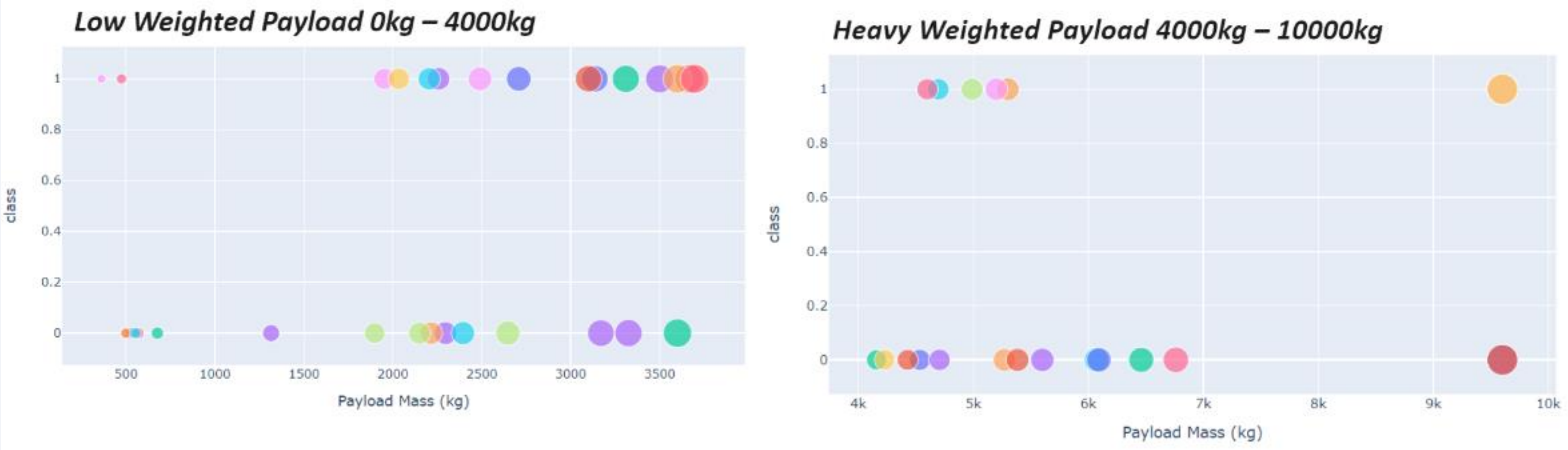
We can see that KSC LC-39A had the most successful launches from all the sites

Pie chart showing the Launch site with the highest launch success ratio



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider

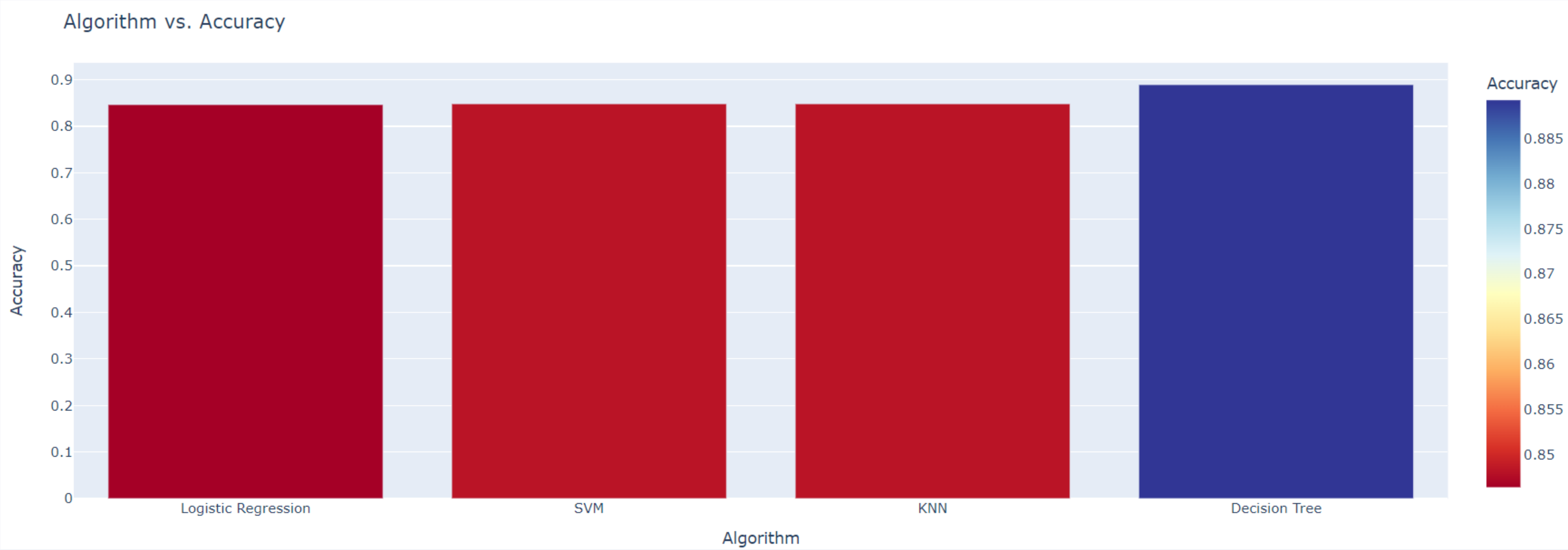


We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

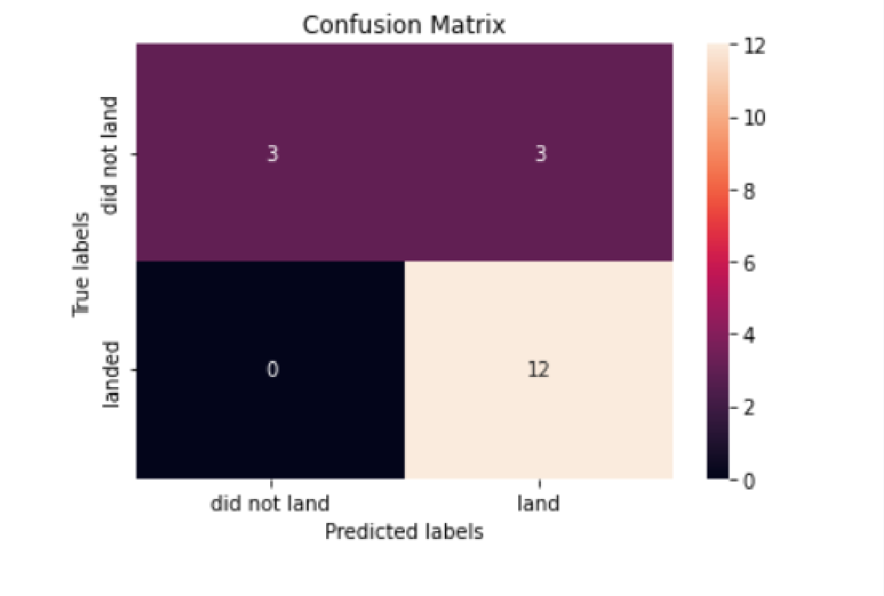
Predictive Analysis (Classification)

Classification Accuracy



As shown Dicision Tree has the highest classification accuracy

Confusion Matrix



The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.

Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the highest success rate.
- KSC LC-39A had the most successful launches amongst other sites, although the increasing payload mass appeared to have a negative impact on success.
- The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!