IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<HafizRoosly>
<10 October 2021>

Outline

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

Executive Summary

•Collected data from public SpaceX API and SpaceX Wikipedia page. Created labels column ‘class’ which classifies successful landings. Explored data using SQL, visualization, folium maps, and dashboards. Gathered relevant columns to be used as features. Changed all categorical variables to binary using one hot encoding. Standardized data and used GridSearchCV to find best parameters for machine learning models. Visualize accuracy score of all models.

•Four machine learning models were produced: Logistic Regression, Support Vector Machine, Decision Tree Classifier, and K Nearest Neighbors. All produced similar results with accuracy rate of about 83.33%. All models over predicted successful landings. More data is needed for better model determination and accuracy.

Introduction

Background:

- Commercial Space Age is Here
- Space X has best pricing ($62 million vs. $165 million USD)
- Largely due to ability to recover part of rocket (Stage 1)
- Does the current setting of shuttle/rocket are at the optimise

Problem:

- To predict successful Stage 1 recovery by training the machine learning.

Section 1

Methodology

Methodology

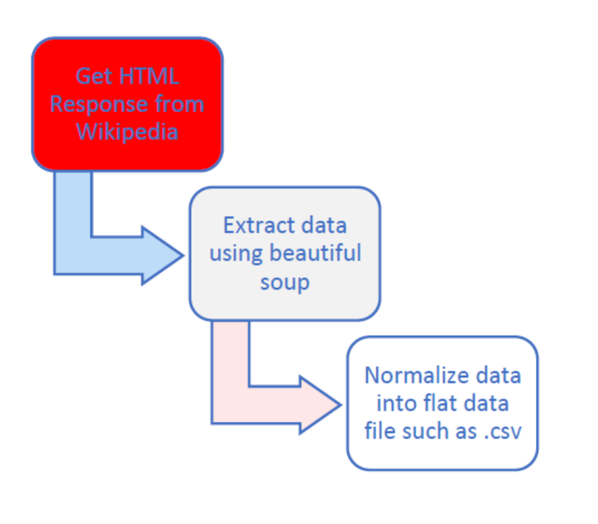
Study Process should be divide into Three Main Part.

1. Data Collecting and Wrangling.
2. Exploratory Data Analysis.
3. Train and Testing(Machine Learning)

Data Collecting

- In this Study we will mainly use web scraping and API.

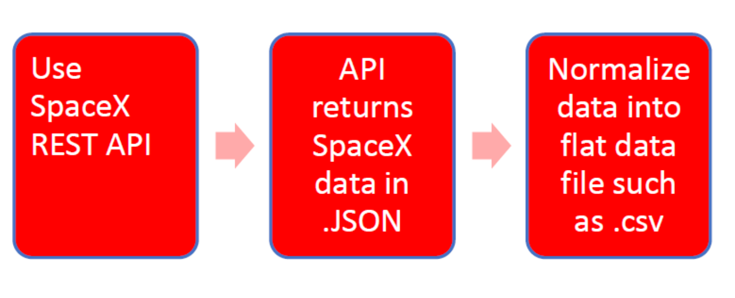
Web scraping Flowchart


Get HTML Response from Wikipedia

Extract data using beautiful soup

Normalize data into flat data file such as .csv

SPACEX API V4 Flowchart


Use SpaceX REST API

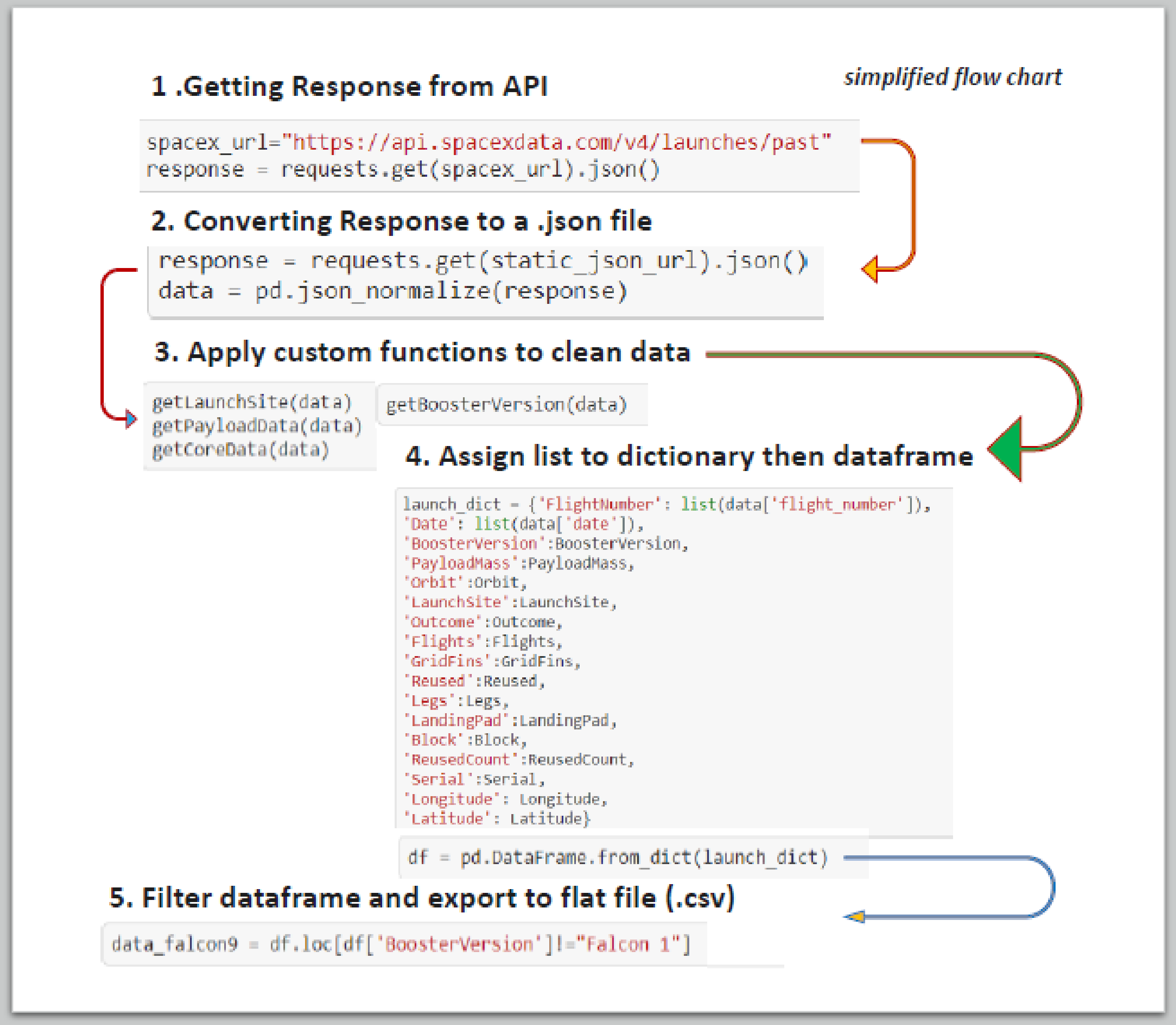
API returns SpaceX data in .JSON

Normalize data into flat data file such as .csv

Data Collection – SpaceX API

- Data collected from past launches will be save into normalize
- A series of function are introduced to clean data
- Data the reassign and export for future easy access

- GitHub Reference



simplified flow chart

1 .Getting Response from API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url).json()
```

2. Converting Response to a .json file

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Apply custom functions to clean data

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
```

```
getBoosterVersion(data)
```

4. Assign list to dictionary then dataframe

```
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}
```

```
df = pd.DataFrame.from_dict(launch_dict)
```

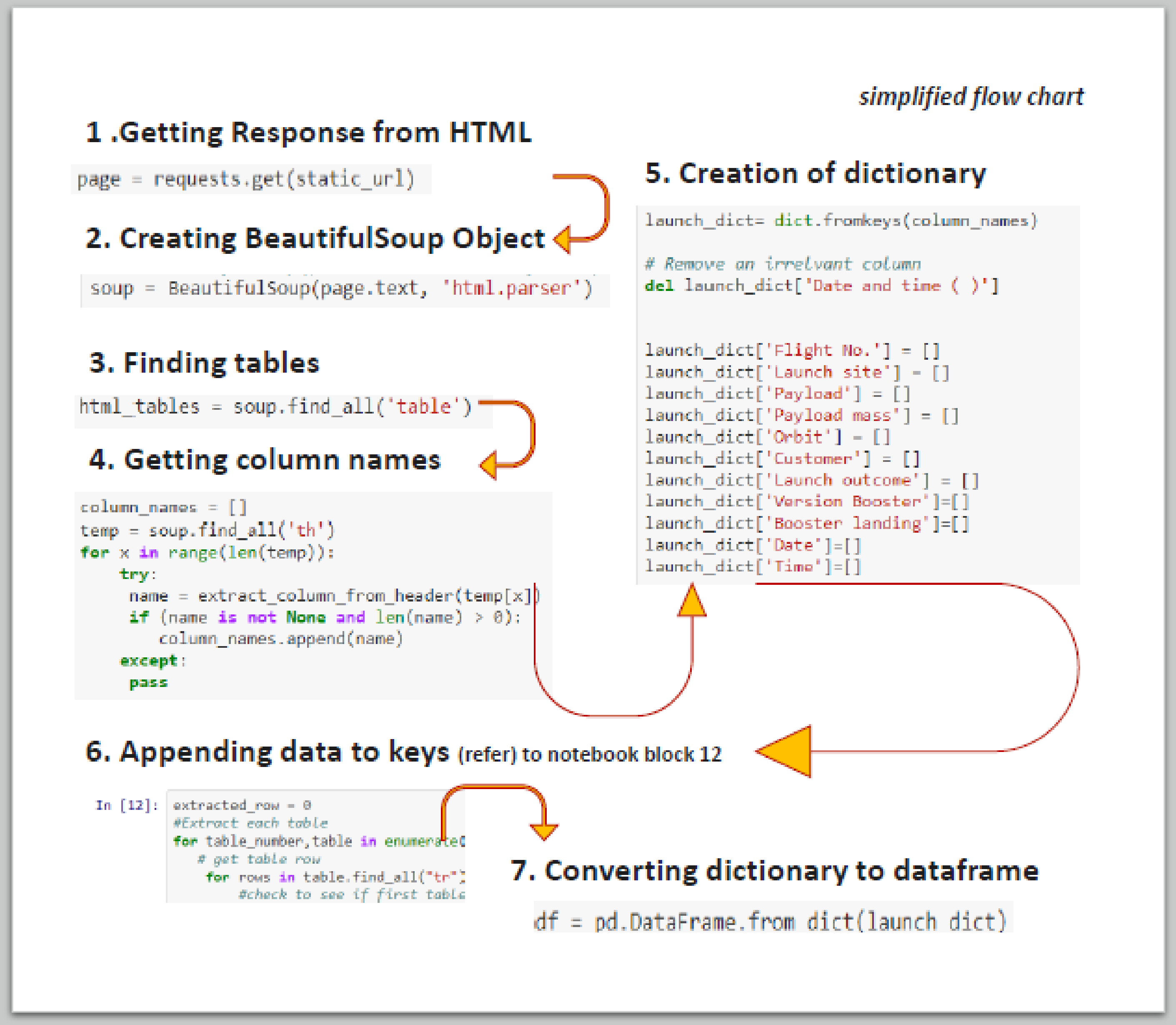
5. Filter dataframe and export to flat file (.csv)

```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
```

Data Collection - Scraping

- Data Collected from Wikipedia are parse into soup.
- Then table should be label and clean using iteration
- Clean table then will be appended to data frame using pandas

- GitHub Reference



simplified flow chart

1 .Getting Response from HTML

```
page = requests.get(static_url)
```

2. Creating BeautifulSoup Object

```
soup = BeautifulSoup(page.text, 'html.parser')
```

3. Finding tables

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

4. Getting column names

```
column_names = []
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

5. Creation of dictionary

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']


launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. Appending data to keys (refer) to notebook block 12

```
In [12]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(
             # get table row
              for rows in table.find_all("tr")
                  #check to see if first table
```

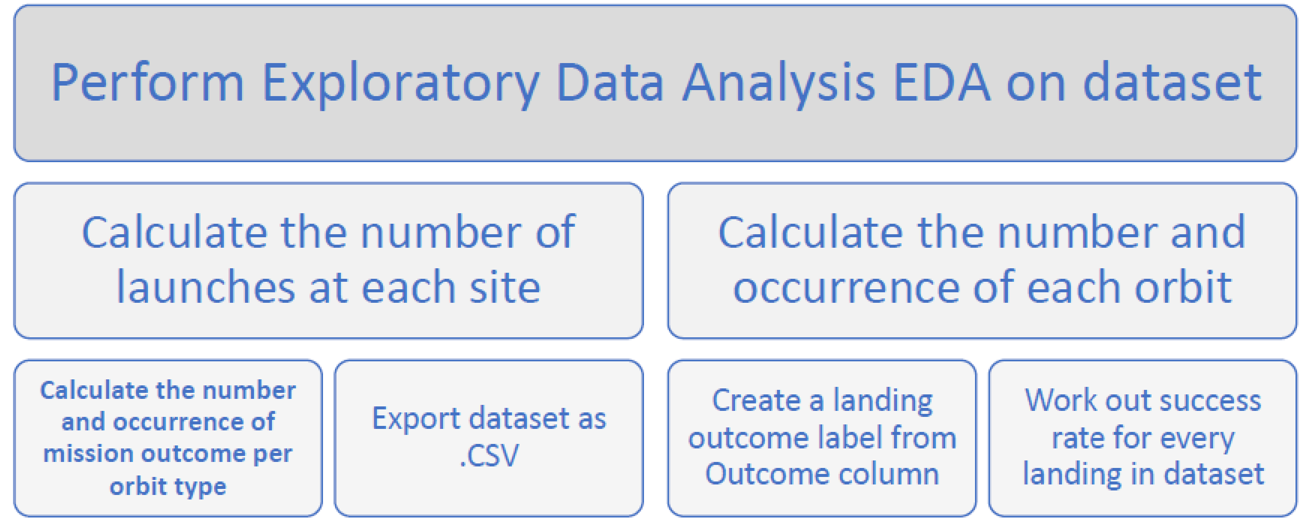
7. Converting dictionary to dataframe

```
df = pd.DataFrame.from_dict(launch_dict)
```

Data Wrangling

- In the data set, there are several different cases where the booster did not land successfully.
- True Ocean, RTLS ,RTLS ,ASDS means the mission outcome was successfully
- False Ocean , RTLS ,RTLS ,ASDS means the mission outcome was unsuccessfully
- We mainly convert those outcomes into Training Labels with 1 means the booster successfully
- landed 0 means it was unsuccessful.

Process



Perform Exploratory Data Analysis EDA on dataset

Calculate the number of launches at each site

Calculate the number and occurrence of each orbit

Calculate the number and occurrence of mission outcome per orbit type

Export dataset as .CSV

Create a landing outcome label from Outcome column

Work out success rate for every landing in dataset

- GitHub Reference

EDA with Data Visualization

Exploratory Data Analysis performed on variables :

Flight Number, Payload Mass, Launch Site, Orbit, Class and Year.

Plots Used:

Flight Number vs. Payload Mass, Flight Number vs. Launch Site, Payload Mass vs. Launch Site, Orbit vs. Success Rate, Flight Number vs. Orbit, Payload vs Orbit, and Success Yearly Trend

Scatter plots, line charts, and bar plots were used to compare relationships between variables to decide if a relationship exists so that they could be used in training the machine learning model

- GitHub Reference

EDA with SQL

- Loaded data set into IBM DB2Database.
- Queried using SQL Python integration.
- Queries were made to get a better understanding of the dataset.
- Queried information about launch site names, mission outcomes, various pay load sizes of customers and booster versions, and landing outcomes

- GitHub Reference

Build an Interactive Map with Folium

- Folium maps mark Launch Sites, successful and unsuccessful landings, and a proximity example to key locations: Railway, Highway, Coast, and City.
- This allows us to understand why launch sites may be located where they are. Also visualizes successful landings relative to location.

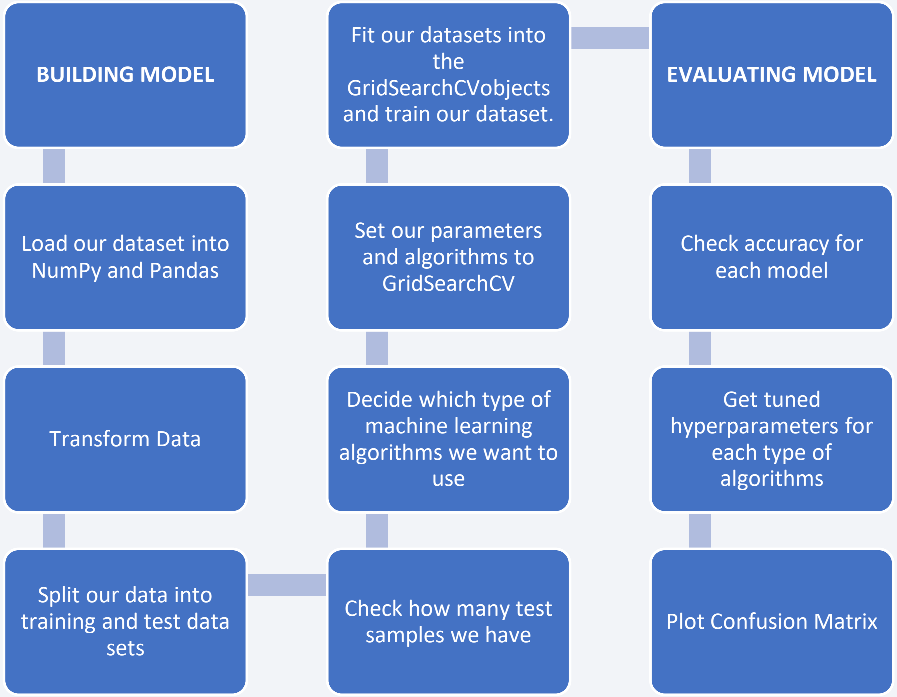
- GitHub Reference

Build a Dashboard with Plotly Dash

- Dashboard includes a pie chart and a scatterplot.
- Pie chart can be selected to show distribution of successful landings across all launch sites and can be selected to show individual launch site success rates.
- Scatter plot takes two inputs: All sites or individual site and payload mass on a slider between 0 and 10000kg.
- The pie chart is used to visualize launch site success rate.
- The scatter plot can help us see how success varies across launch sites, payload mass, and booster version category.

- GitHub Reference

Predictive Analysis (Classification)



BUILDING MODEL
Load our dataset into NumPy and Pandas
Transform Data
Split our data into training and test data sets
Check how many test samples we have
Decide which type of machine learning algorithms we want to use
Set our parameters and algorithms to GridSearchCV
Fit our datasets into the GridSearchCVobjects and train our dataset.
EVALUATING MODEL
Check accuracy for each model
Get tuned hyperparameters for each type of algorithms
Plot Confusion Matrix

IMPROVING MODEL

- Feature Engineering
- Algorithm Tuning

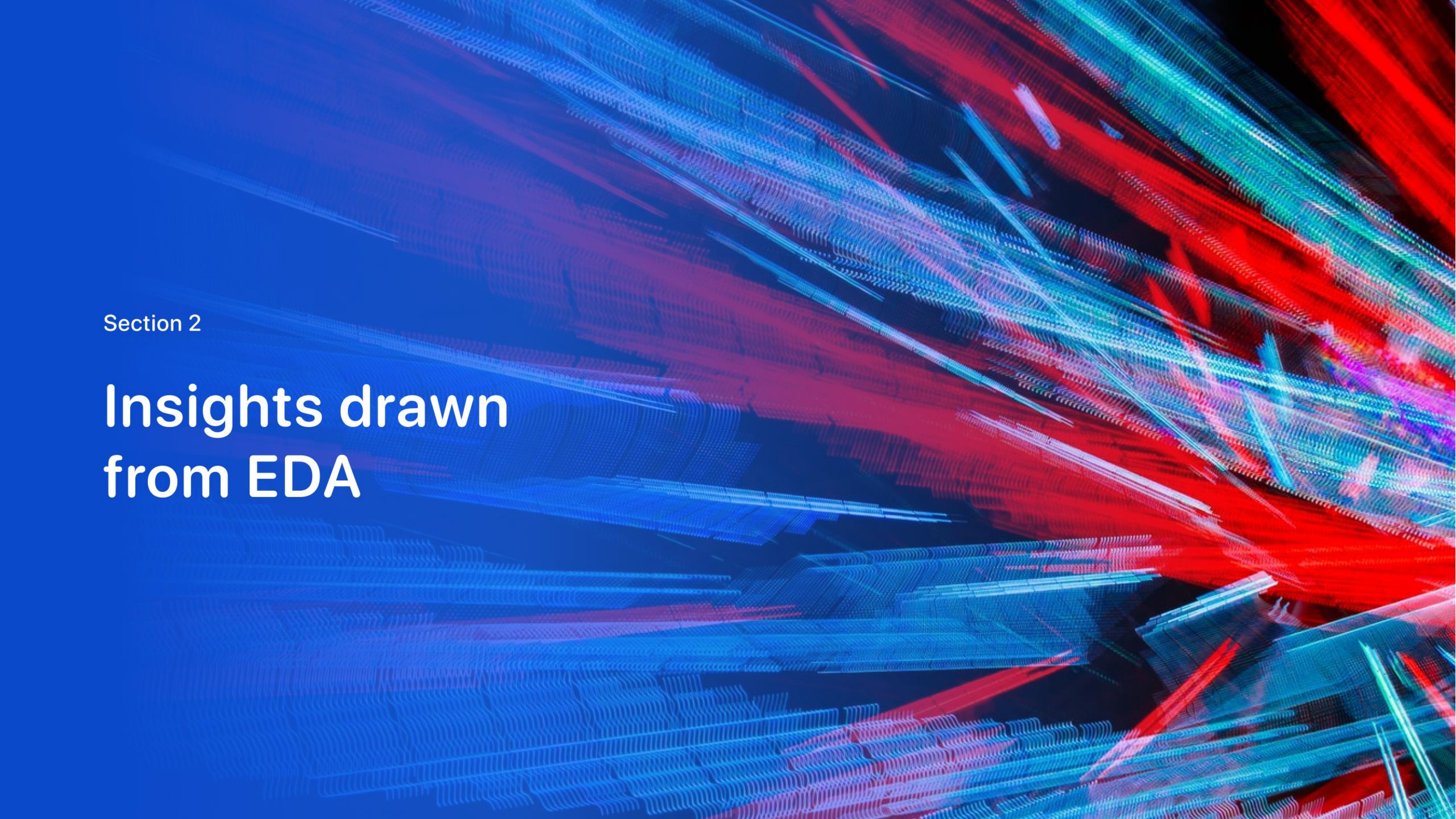
FINDING THE BEST PERFORMING CLASSIFICATION MODEL

- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

- GitHub Reference

Results

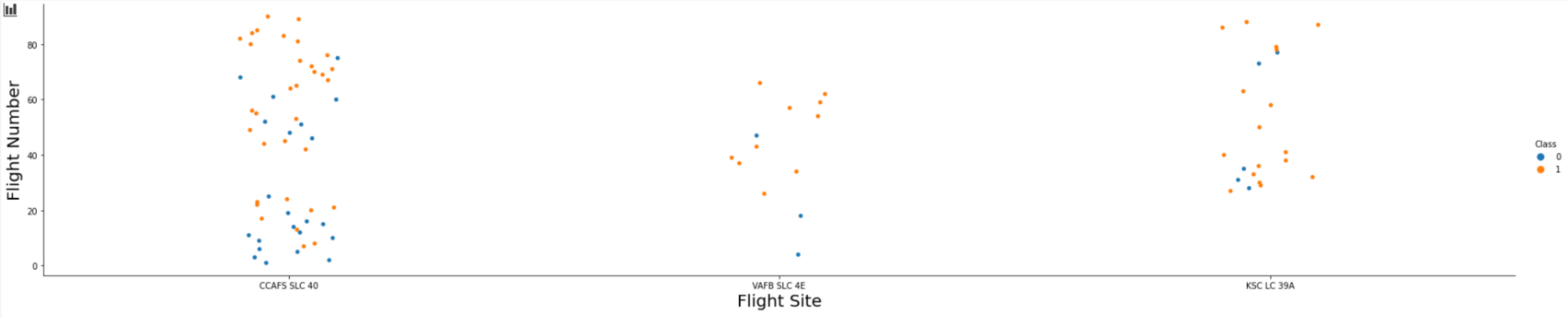
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results



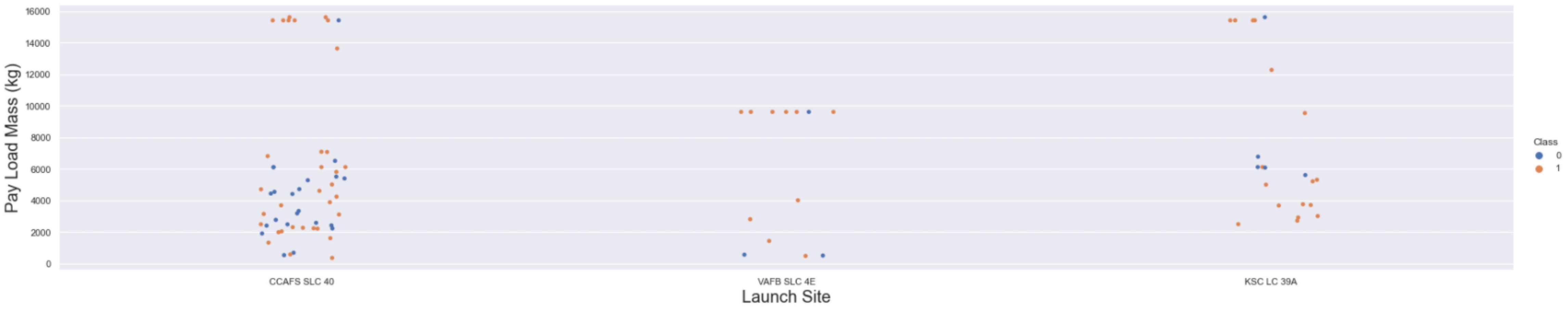
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site



The greater number of flights at a launch site the greater the success rate.

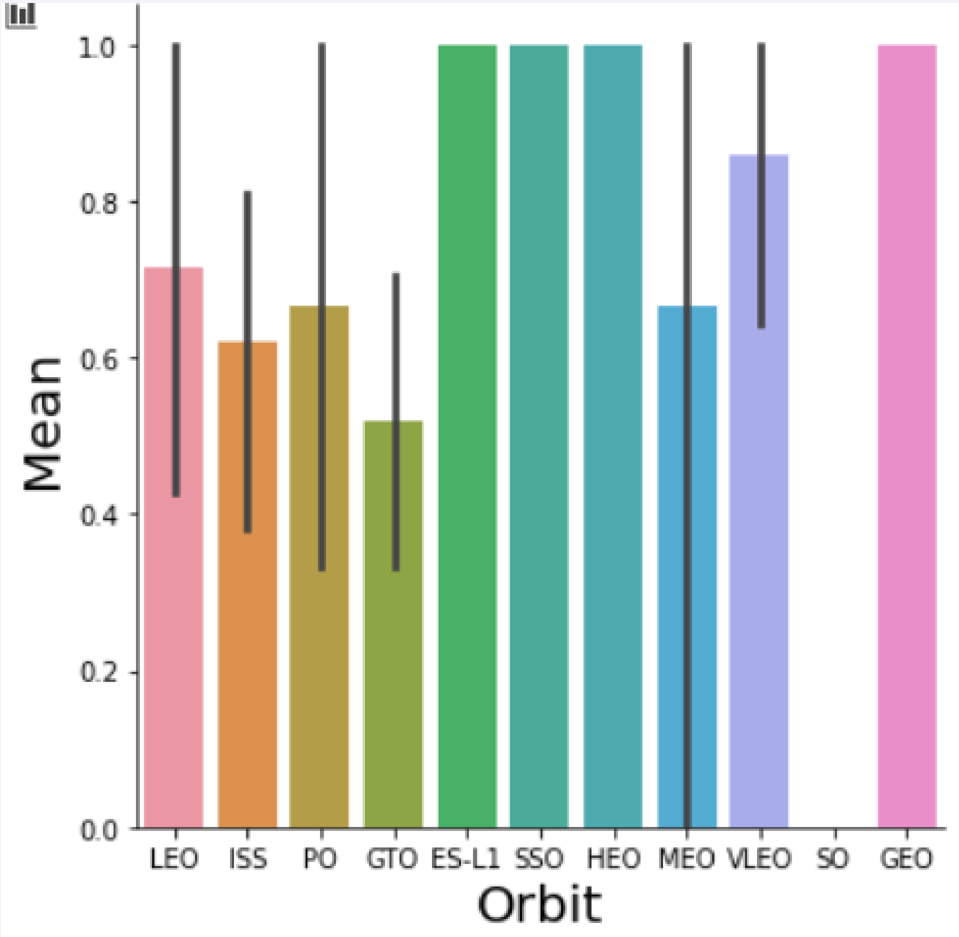


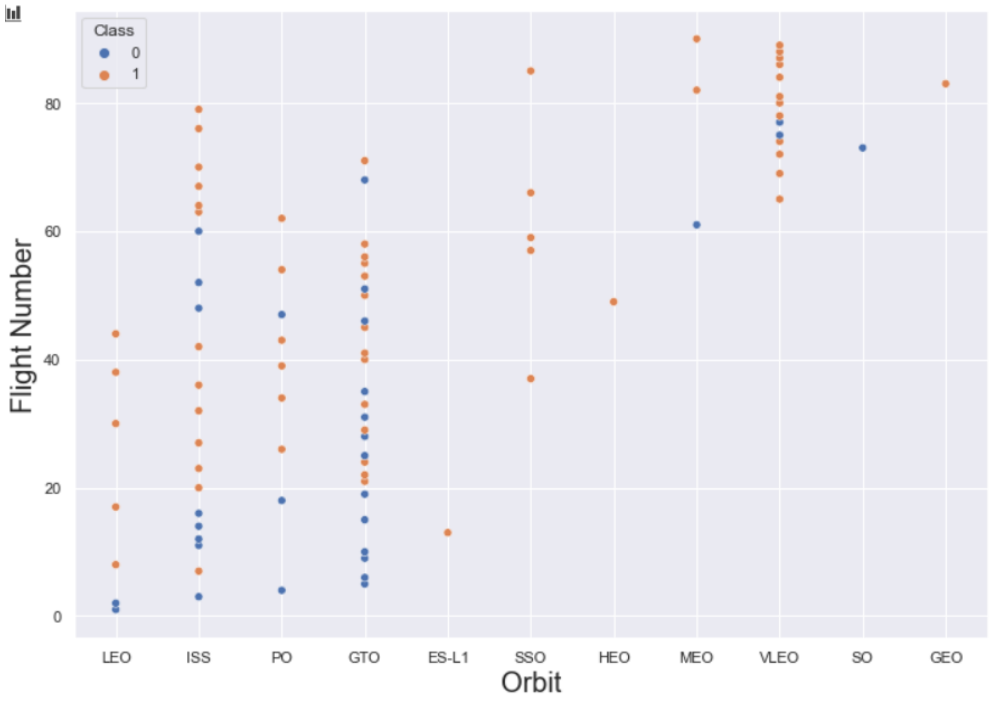
Payload vs. Launch Site

- Using this visualization, no clear pattern to we can conclude if the Launch Site is dependant on Pay Load Mass for a success launch.
- Few detail we can se for CCAFS SLC 40,lThe greater the payload mass the higher the success rate for the Rocket.

Success Rate vs. Orbit Type

- Success Rate for ES-L1,SSO,HEO and GEO have the best success rate while GTO has the lowest.

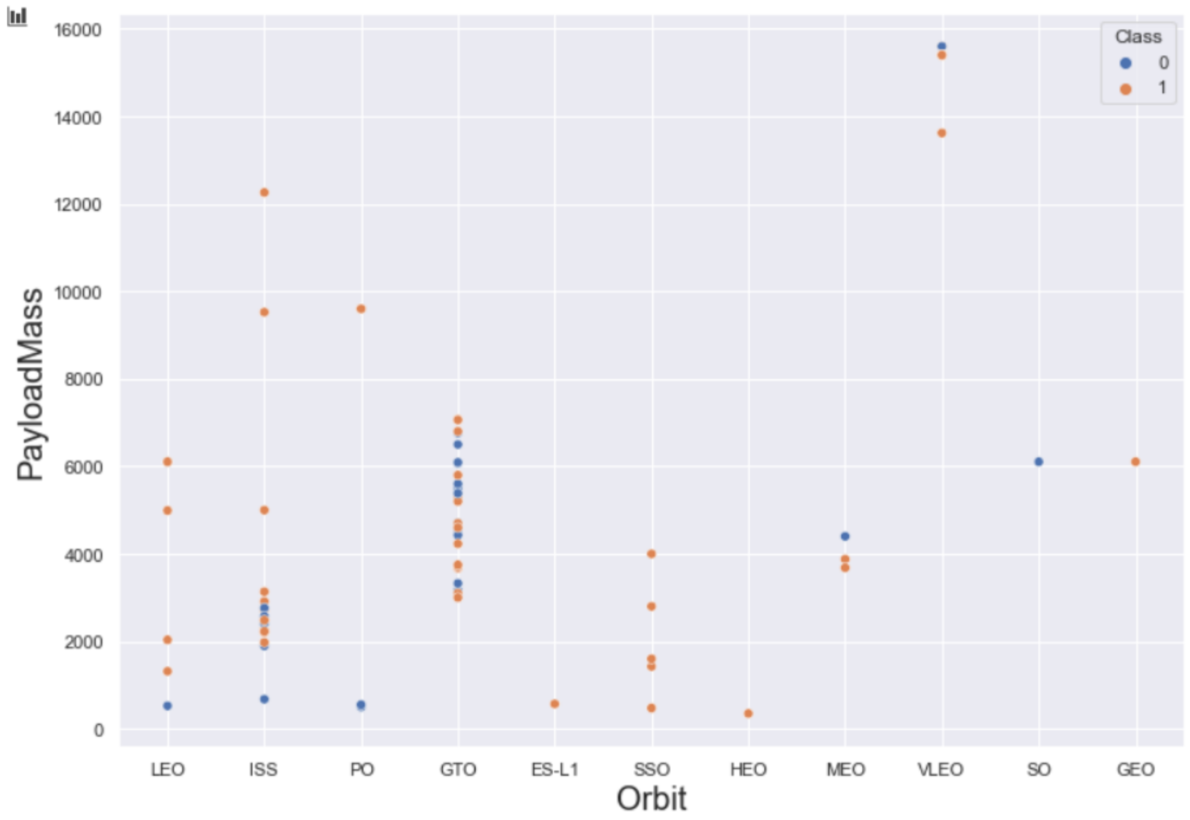




Flight Number vs. Orbit Type

- In LEO orbit, we can clearly see that number of flight increase.
- on the other hand, there seems to be no relationship between flight number when in GTO orbit.

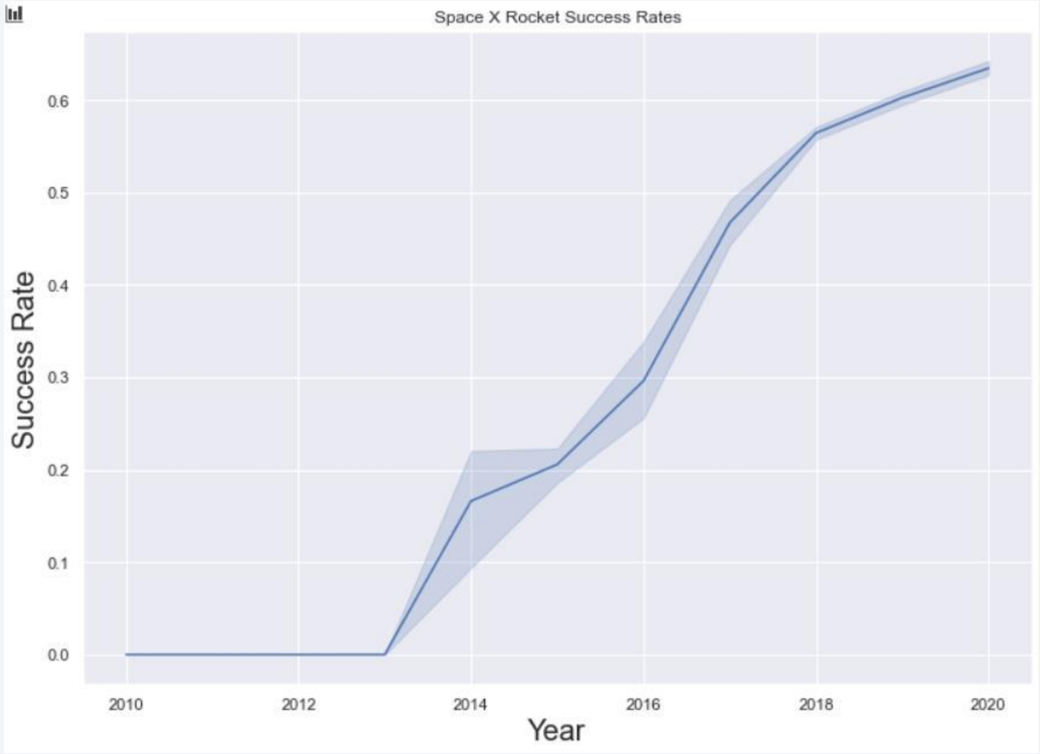
Payload vs. Orbit Type

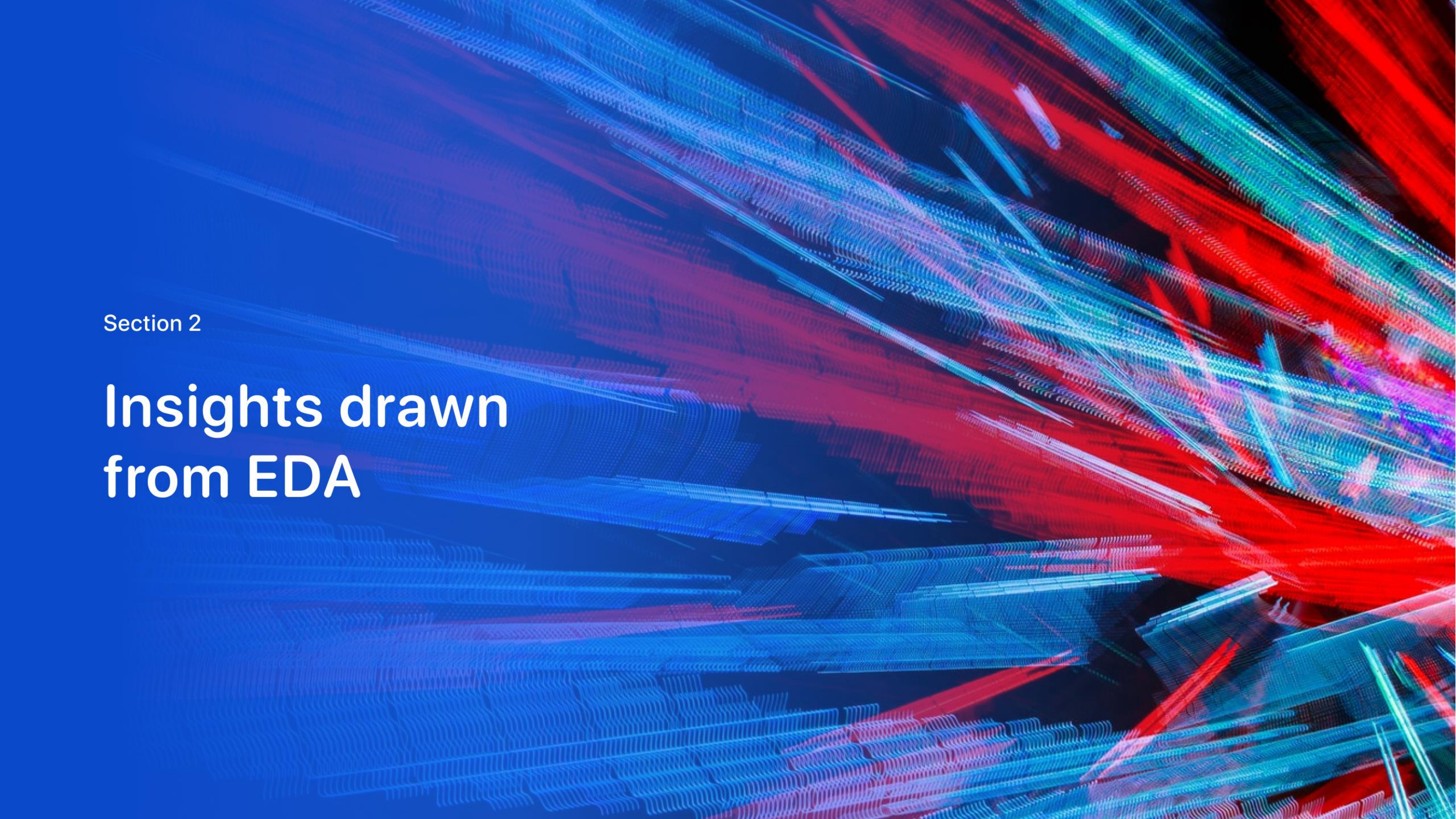


- You should observe that Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

Launch Success Yearly Trend

- you can observe that the success rate since 2013 kept increasing till 2020



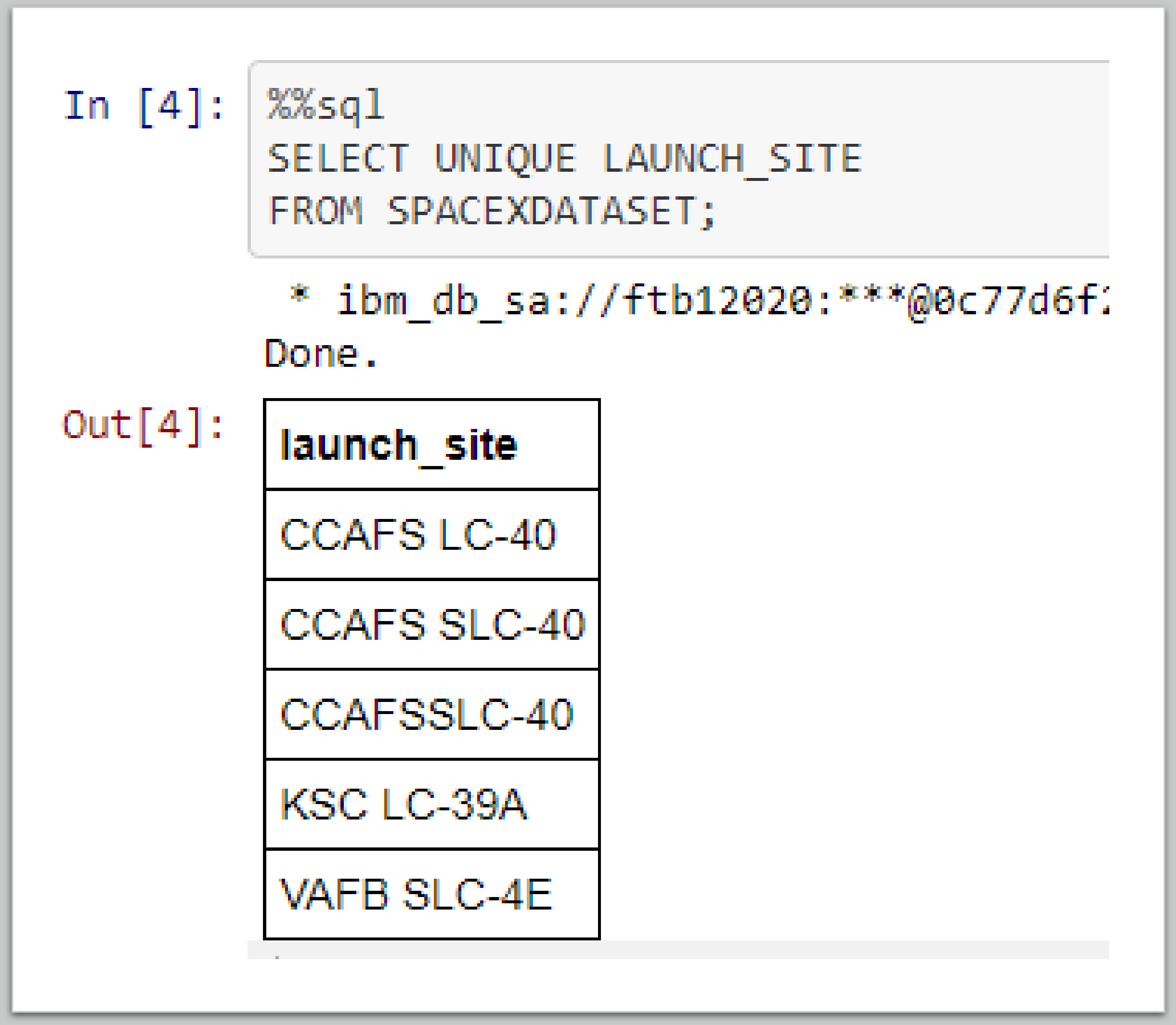


Section 2

Insights drawn from EDA

All Launch Site Names

- Query unique launch site names fromdatabase.
- CCAFS SLC-40 and CCAFSSLC-40 likely all represent thesame
- launch site with data entryerrors.
- CCAFS LC-40 was the previous name. Likely only 3 unique launch_site values: CCAFS SLC-40, KSC LC-39A,VAFB SLC-4E



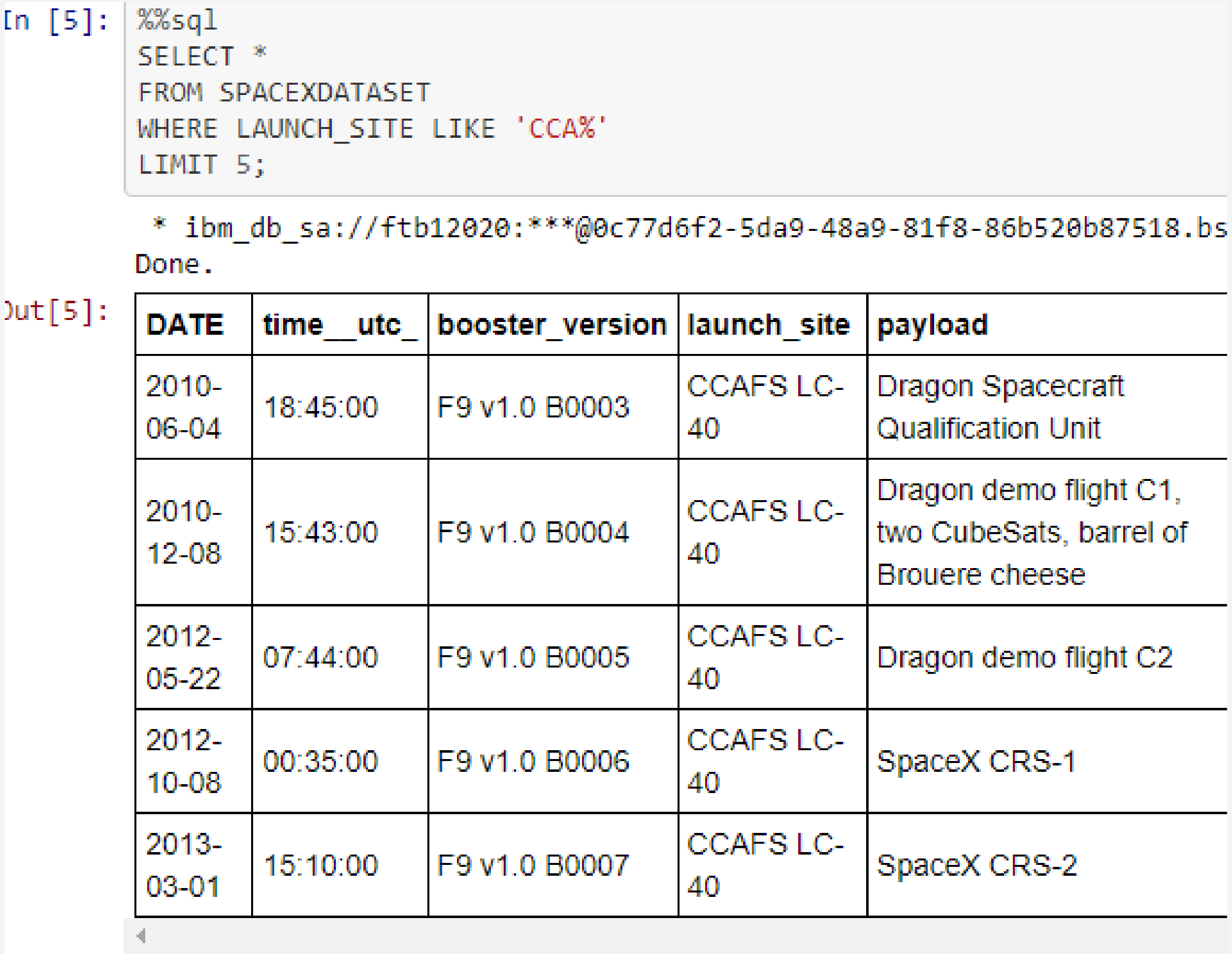
```
In [4]: %%sql
        SELECT UNIQUE LAUNCH_SITE
        FROM SPACEXDATASET;

 * ibm_db_sa://ftb12020:***@0c77d6f
Done.
```

Out[4]:

launch_site
CCAFS LC-40
CCAFS SLC-40
CCAFSSLC-40
KSC LC-39A
VAFB SLC-4E

Launch Site Names Begin with 'CCA'



```
[n [5]: %%sql
        SELECT *
        FROM SPACEXDATASET
        WHERE LAUNCH_SITE LIKE 'CCA%'
        LIMIT 5;

         * ibm_db_sa://ftb12020:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs
        Done.
```

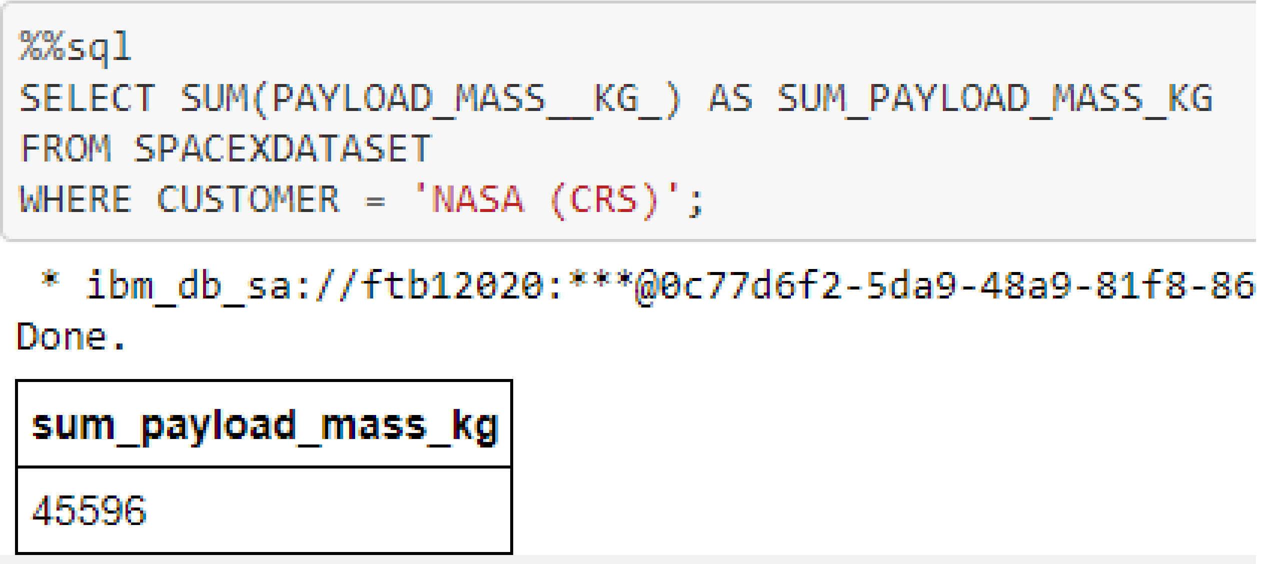
)ut[5]:

DATE	time__utc_	booster_version	launch_site	payload
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2

First five entries in database with Launch Site name beginning with CCA.

Total Payload Mass

- This query sums the total payload mass in kg where NASA was the customer.
- CRS stands for Commercial Resupply Services which indicates that these payloads were sent to the International Space Station (ISS).



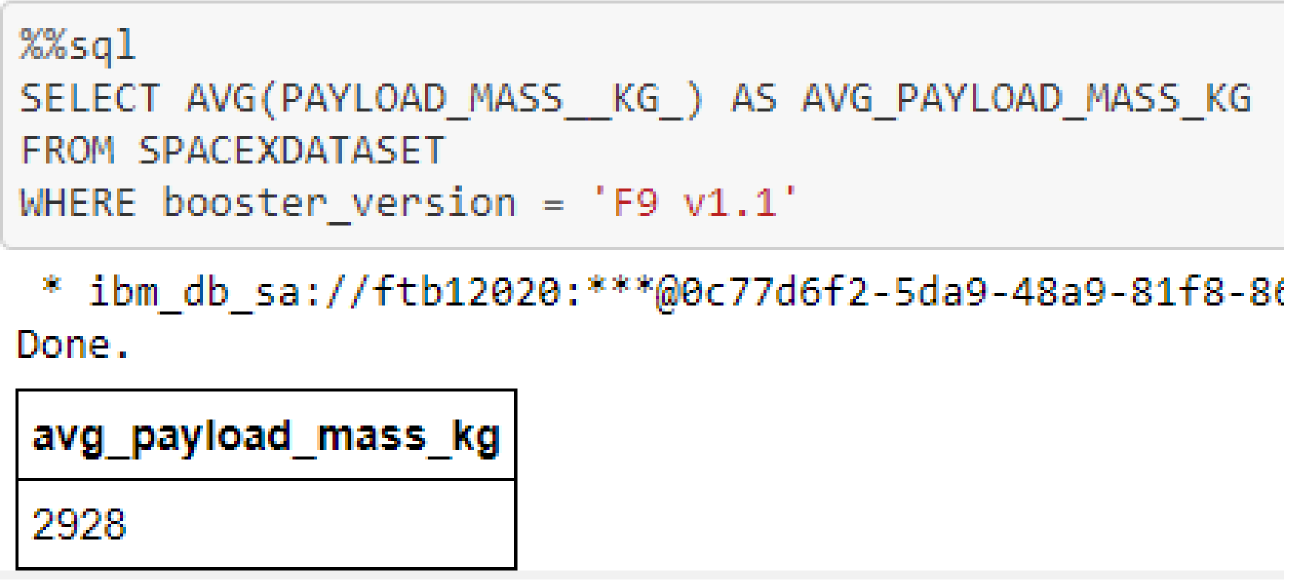
```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) AS SUM_PAYLOAD_MASS_KG
FROM SPACEXDATASET
WHERE CUSTOMER = 'NASA (CRS)';
```

```
 * ibm_db_sa://ftb12020:***@0c77d6f2-5da9-48a9-81f8-86
Done.
```

sum_payload_mass_kg
45596

Average Payload Mass by F9 v1.1

- Using the function ***AVG*** works out the average in the column ***PAYLOAD_MASS_KG_***
- The ***WHERE clause*** filters the dataset to only perform calculations on ***Booster_version F9 v1.1***

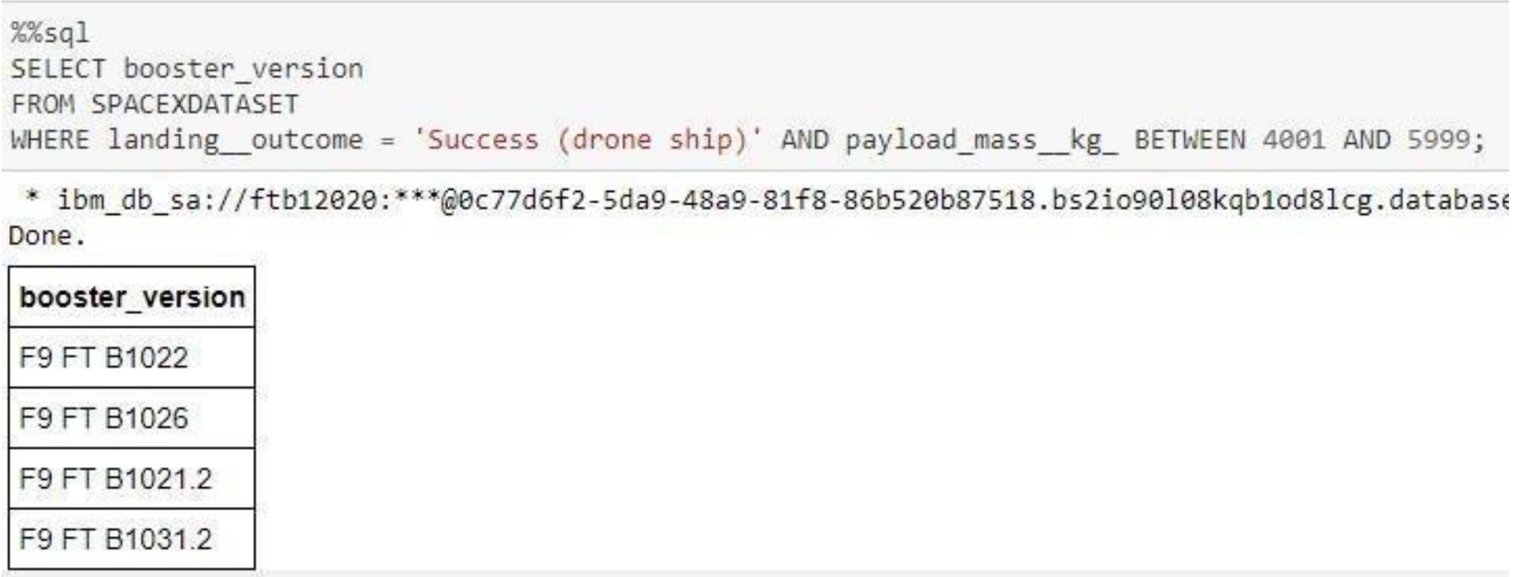


```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD_MASS_KG
FROM SPACEXDATASET
WHERE booster_version = 'F9 v1.1'
```

```
 * ibm_db_sa://ftb12020:***@0c77d6f2-5da9-48a9-81f8-8
Done.
```

avg_payload_mass_kg
2928

First Successful Ground Landing Date



```
%%sql
SELECT booster_version
FROM SPACEXDATASET
WHERE landing__outcome = 'Success (drone ship)' AND payload_mass__kg_ BETWEEN 4001 AND 5999;
```

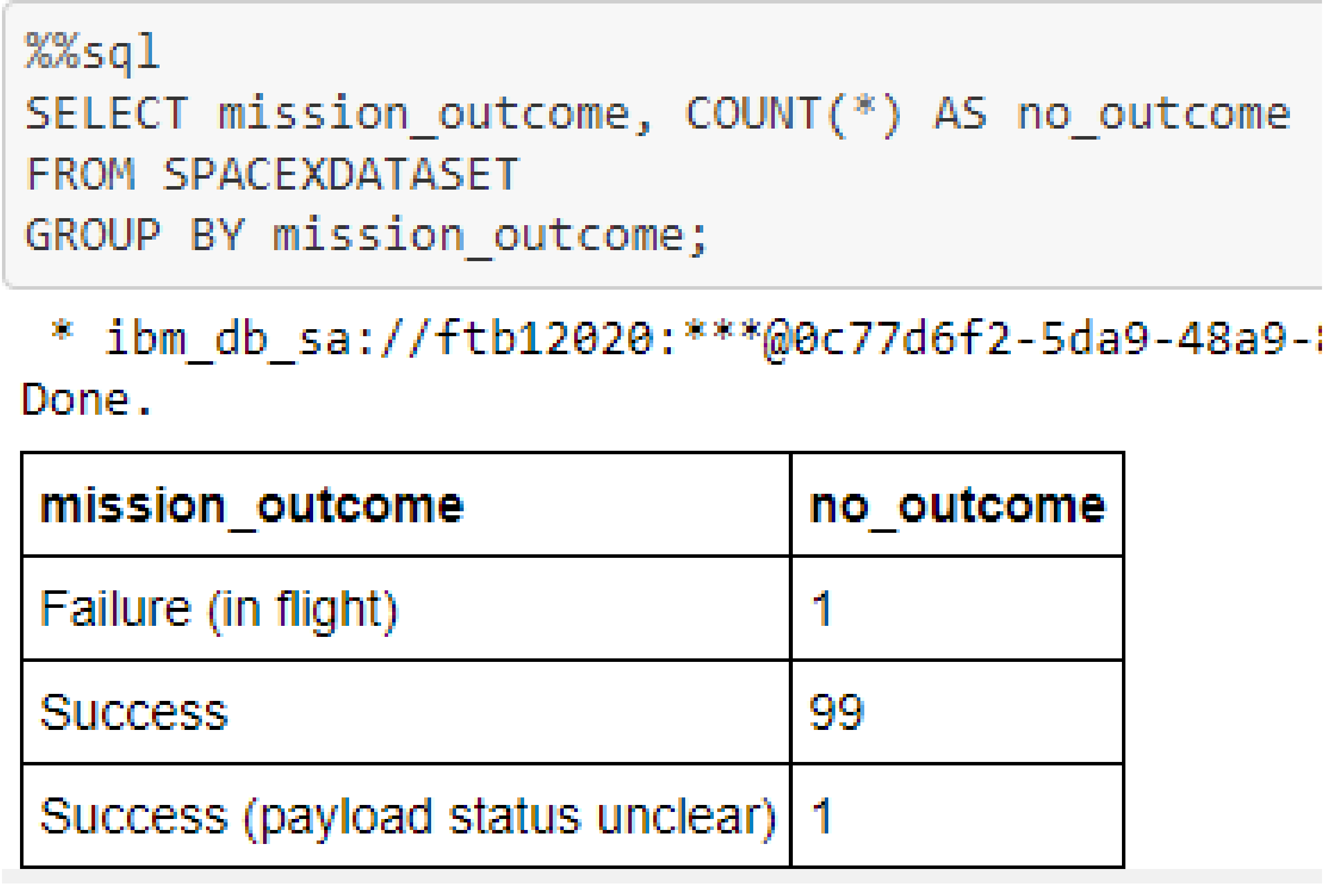
```
 * ibm_db_sa://ftb12020:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databas
Done.
```

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

- Using the function ***MIN*** works out the minimum date in the column ***Date***
- The ***WHERE*** clause filters the dataset to only perform calculations on ***Landing_Outcome Success (drone ship)***

Successful Drone Ship Landing with Payload between 4000 and 6000

- The ***WHERE*** clause filters the dataset to ***Landing_Outcome = Success (drone ship)***
- The ***AND*** clause specifies additional filter conditions
- ***Payload_MASS_KG_***>4000AND***Payload_MASS_KG_<6000***



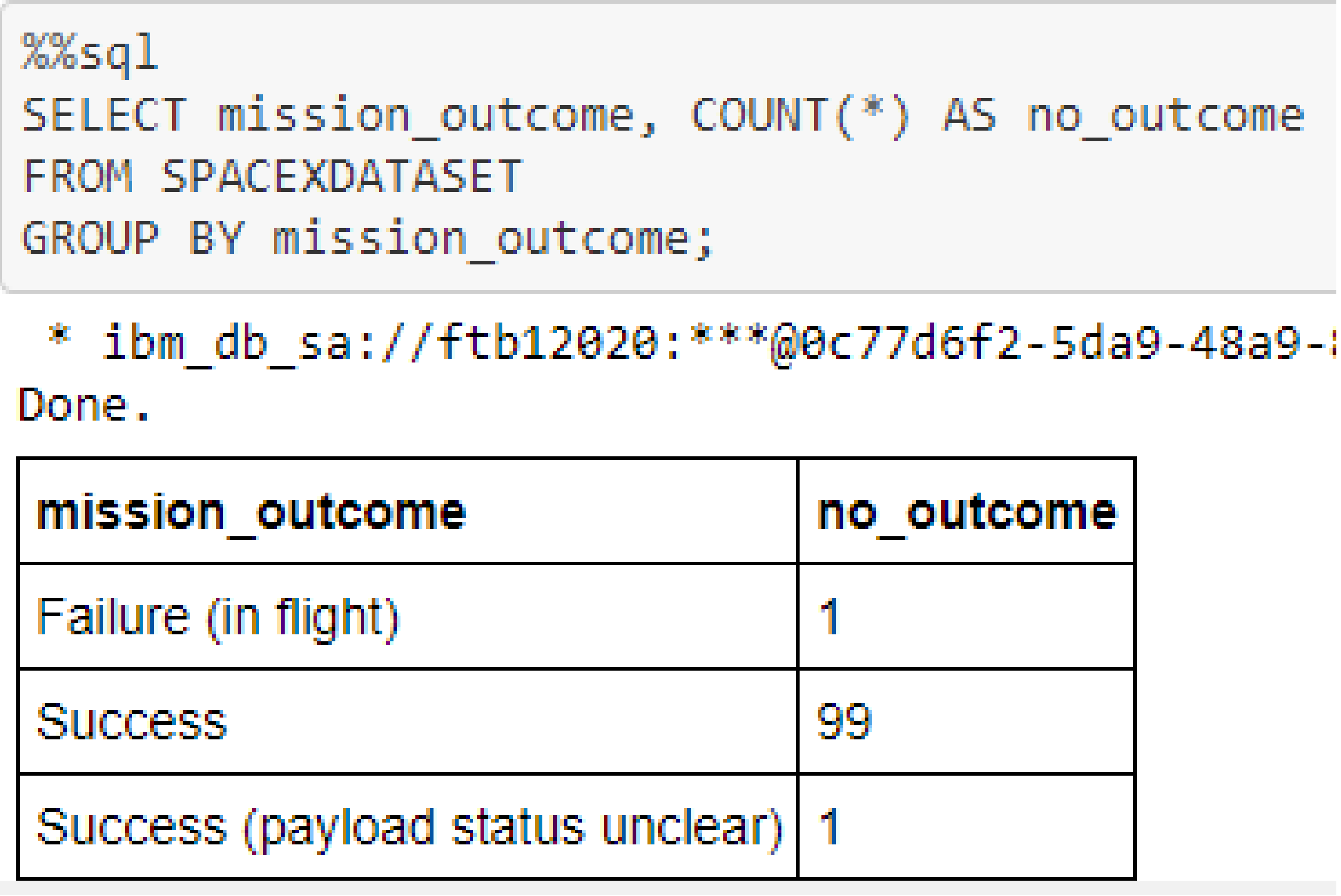
```
%%sql
SELECT mission_outcome, COUNT(*) AS no_outcome
FROM SPACEXDATASET
GROUP BY mission_outcome;
```

```
 * ibm_db_sa://ftb12020:***@0c77d6f2-5da9-48a9-
Done.
```

mission_outcome	no_outcome
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Total Number of Successful and Failure Mission Outcomes

- This query returns a count of each mission outcome.
- SpaceX appears to achieve its mission outcome nearly 99% of the time.
- This means that most of the landing failures are intended.
- Interestingly, one launch has an unclear payload status and unfortunately one failed inflight..



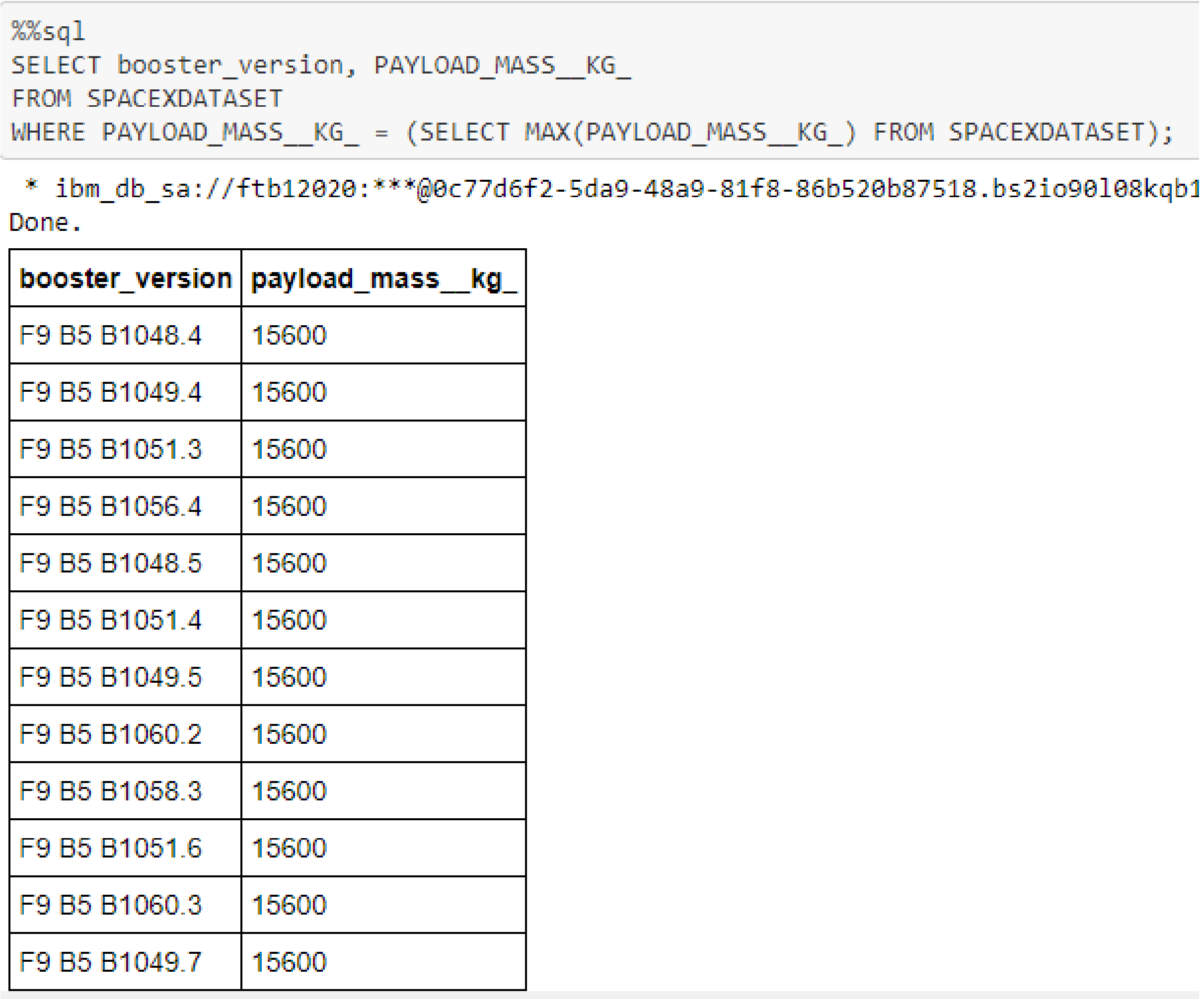
```
%%sql
SELECT mission_outcome, COUNT(*) AS no_outcome
FROM SPACEXDATASET
GROUP BY mission_outcome;
```

```
 * ibm_db_sa://ftb12020:***@0c77d6f2-5da9-48a9-
Done.
```

mission_outcome	no_outcome
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

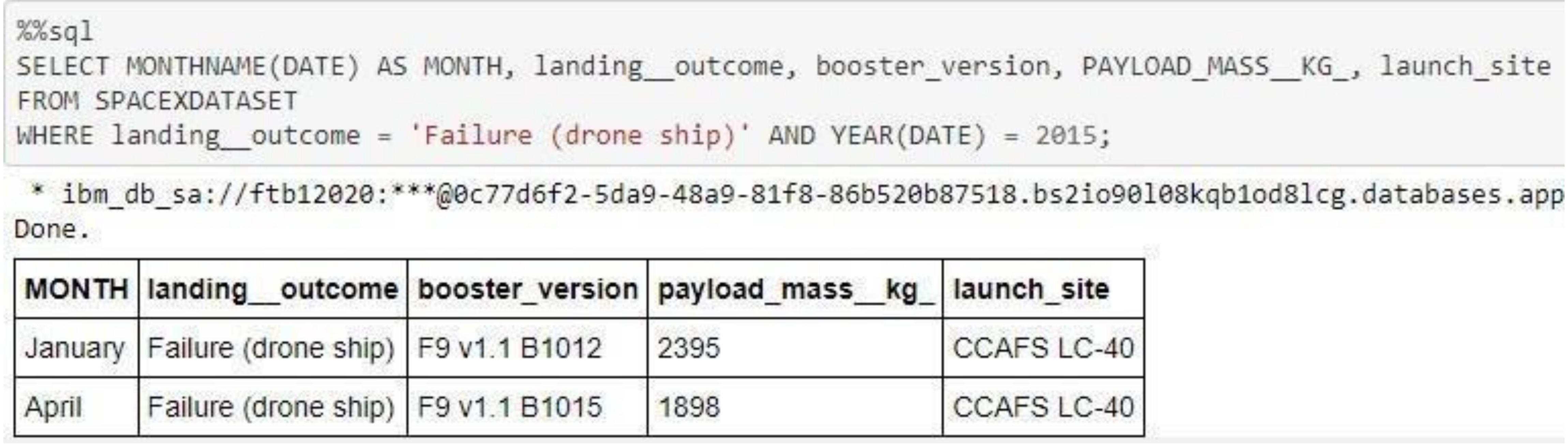
- This query returns the booster versions that carried the highest payload mass of 15600 kg.
- These booster versions are very similar, and all are of the F9 B5 B10xx.xvariety.
- This likely indicates payload mass correlates with the booster version that issued.



```
%%sql
SELECT booster_version, PAYLOAD_MASS__KG_
FROM SPACEXDATASET
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXDATASET);
```

```
 * ibm_db_sa://ftb12020:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1
Done.
```

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600



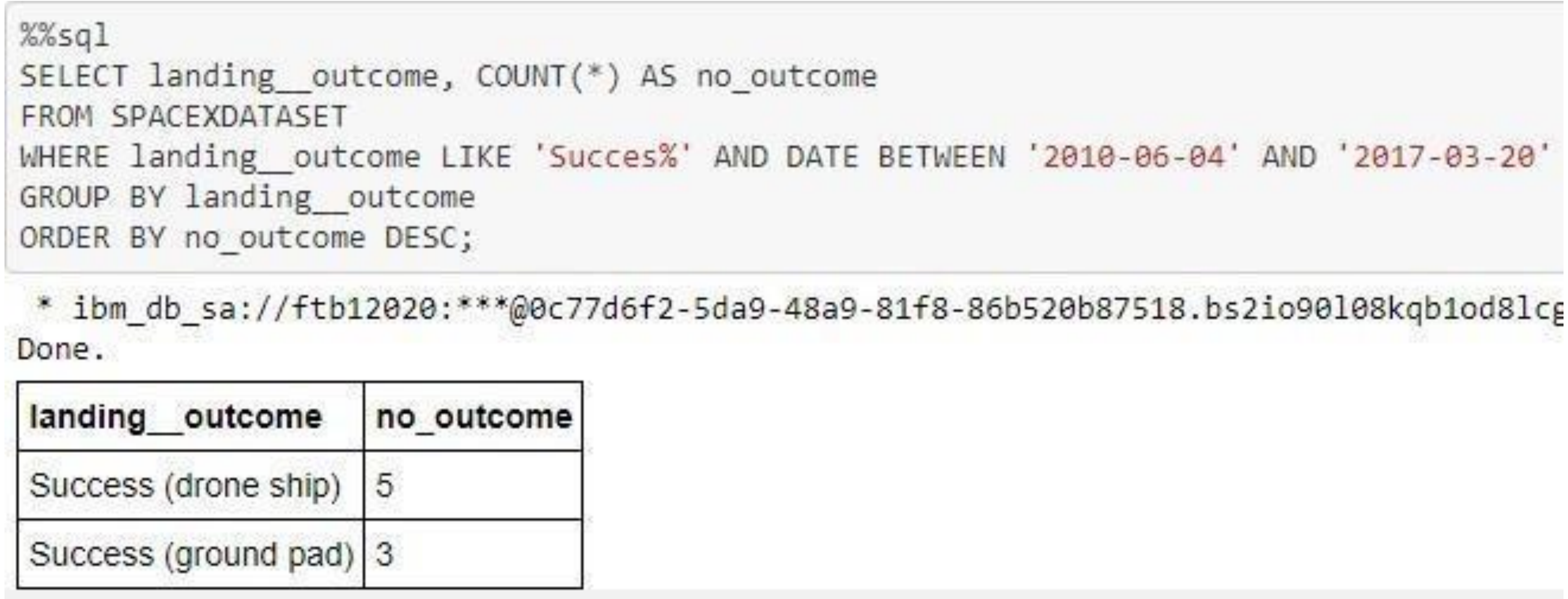
```
%%sql
SELECT MONTHNAME(DATE) AS MONTH, landing__outcome, booster_version, PAYLOAD_MASS__KG_, launch_site
FROM SPACEXDATASET
WHERE landing__outcome = 'Failure (drone ship)' AND YEAR(DATE) = 2015;
```

```
 * ibm_db_sa://ftb12020:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.app
Done.
```

MONTH	landing__outcome	booster_version	payload_mass__kg_	launch_site
January	Failure (drone ship)	F9 v1.1 B1012	2395	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	1898	CCAFS LC-40

2015 Failed Drone Ship Landing Record

- This query returns the Month,Landing Outcome, Booster Version, Payload Mass (kg), and Launch site of 2015 launches where stage 1 failed to land on a droneship.
- There were two suchoccurrences.



```
%%sql
SELECT landing__outcome, COUNT(*) AS no_outcome
FROM SPACEXDATASET
WHERE landing__outcome LIKE 'Succes%' AND DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY landing__outcome
ORDER BY no_outcome DESC;
```

```
 * ibm_db_sa://ftb12020:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg
Done.
```

landing__outcome	no_outcome
Success (drone ship)	5
Success (ground pad)	3

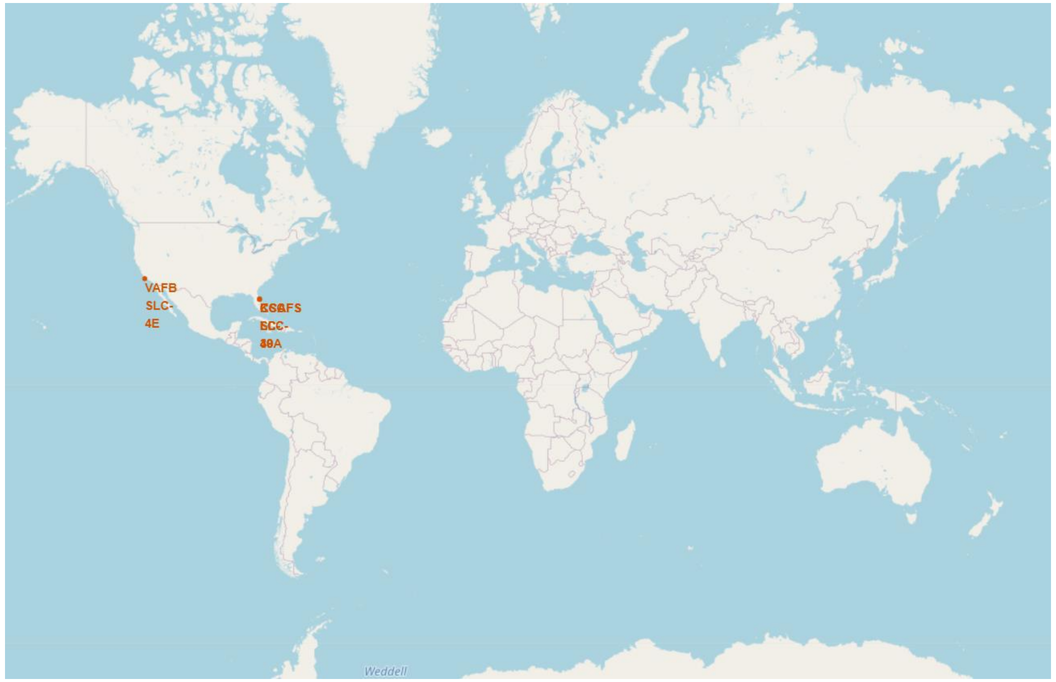
- This query returns a list of successful landings and between 2010-06-04 and 2017-03-20 inclusively.
- There are two types of successful landing outcomes: drone ship and ground pad landings.
- There were 8 successful landings in total during this time period

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

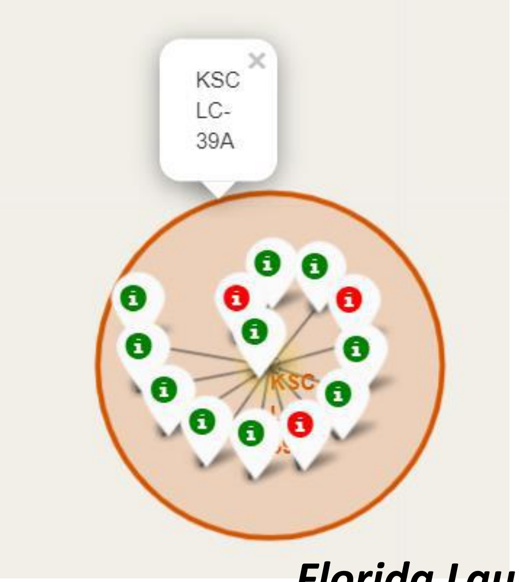
Section 4

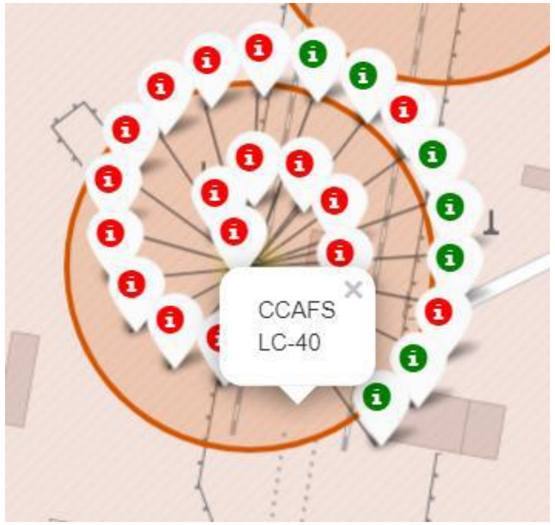
Launch Sites Proximities Analysis

Launch Site Location



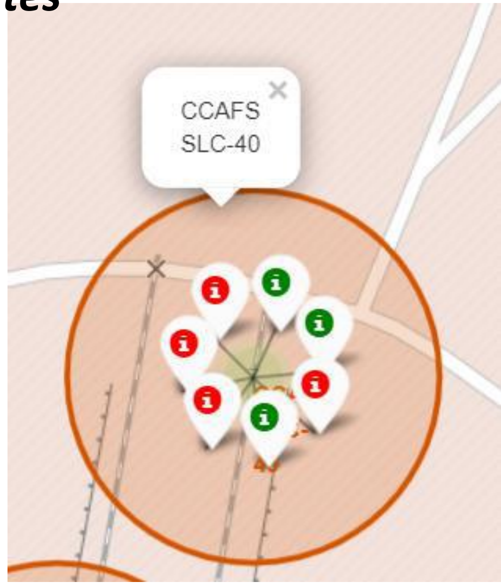
- ***We can see that the SpaceX launch sites are in the United States of America coasts.Florida and California***



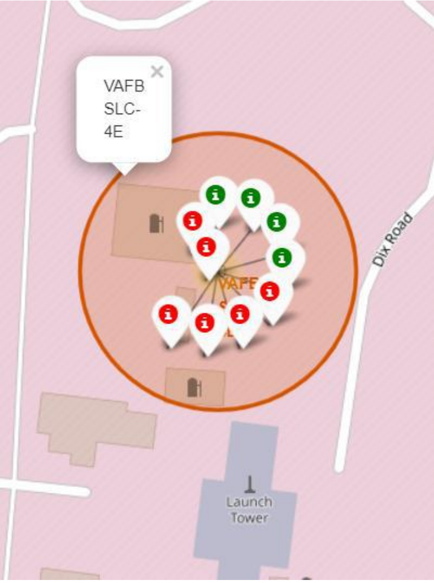


Florida Launch Sites

7
CAFS
SLC-
40
26
CAFS
LC-
40
Centaur Road

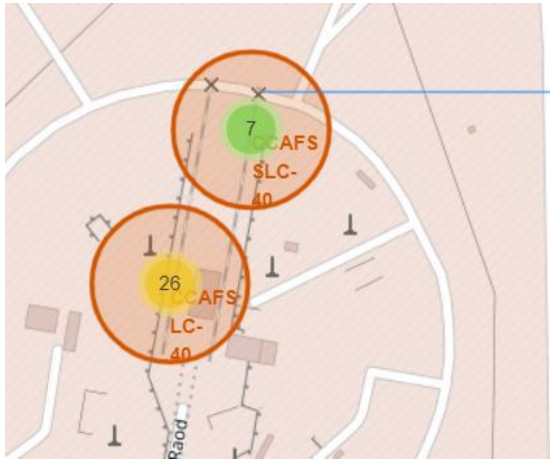


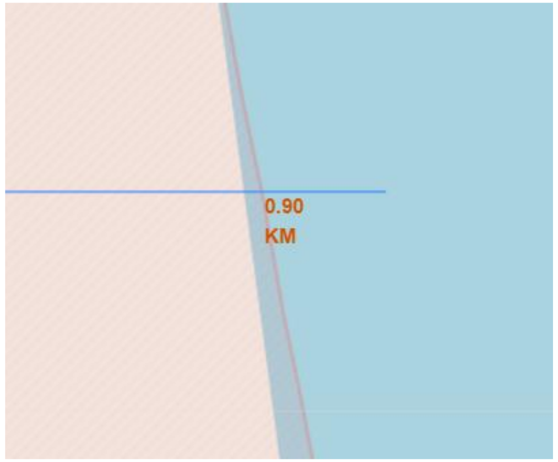
California Launch Site

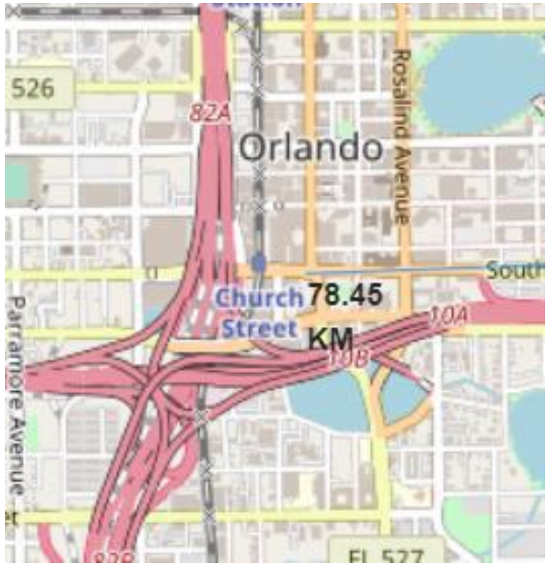


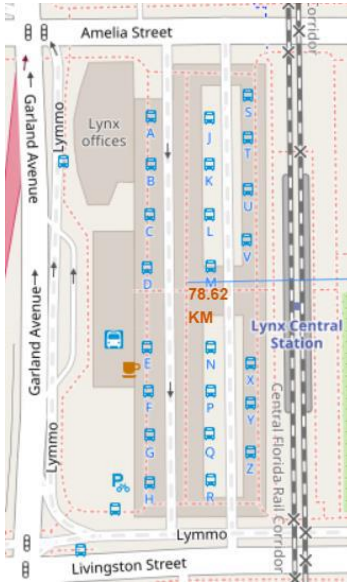
Color-Coded Launch Markers

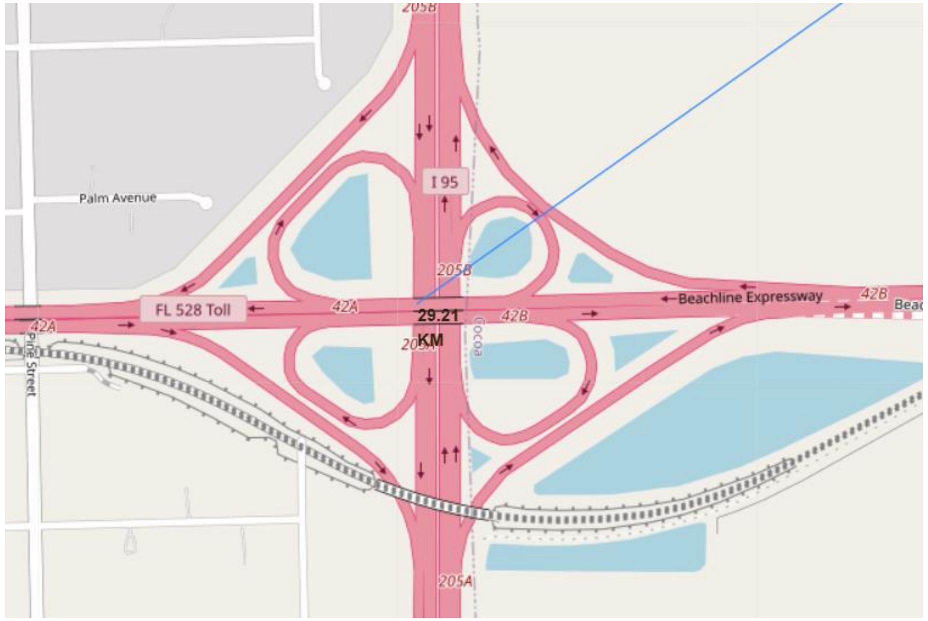
Green Marker shows successful Launches andRed Markershows Failures









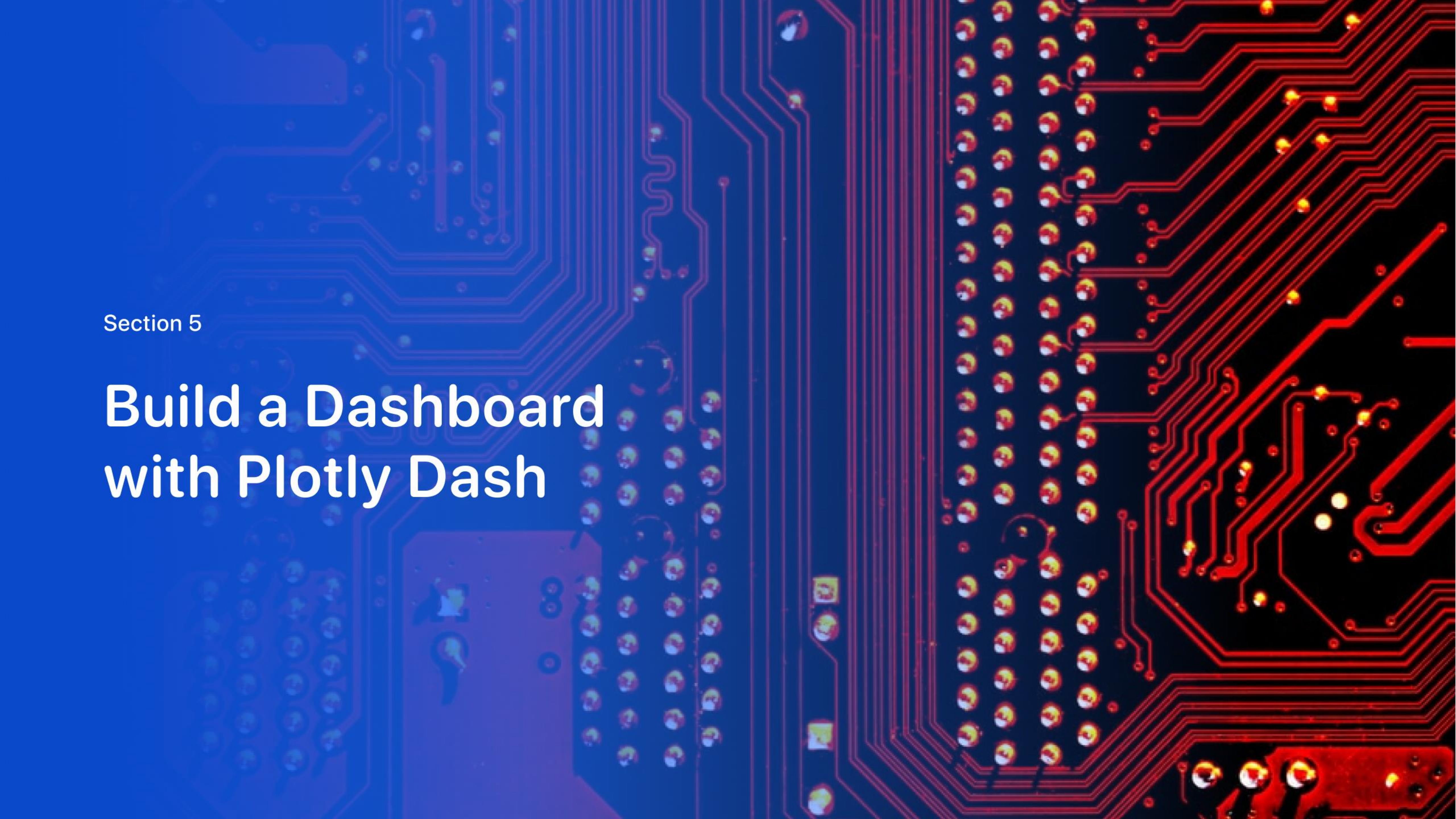


Key Location Proximities

- 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

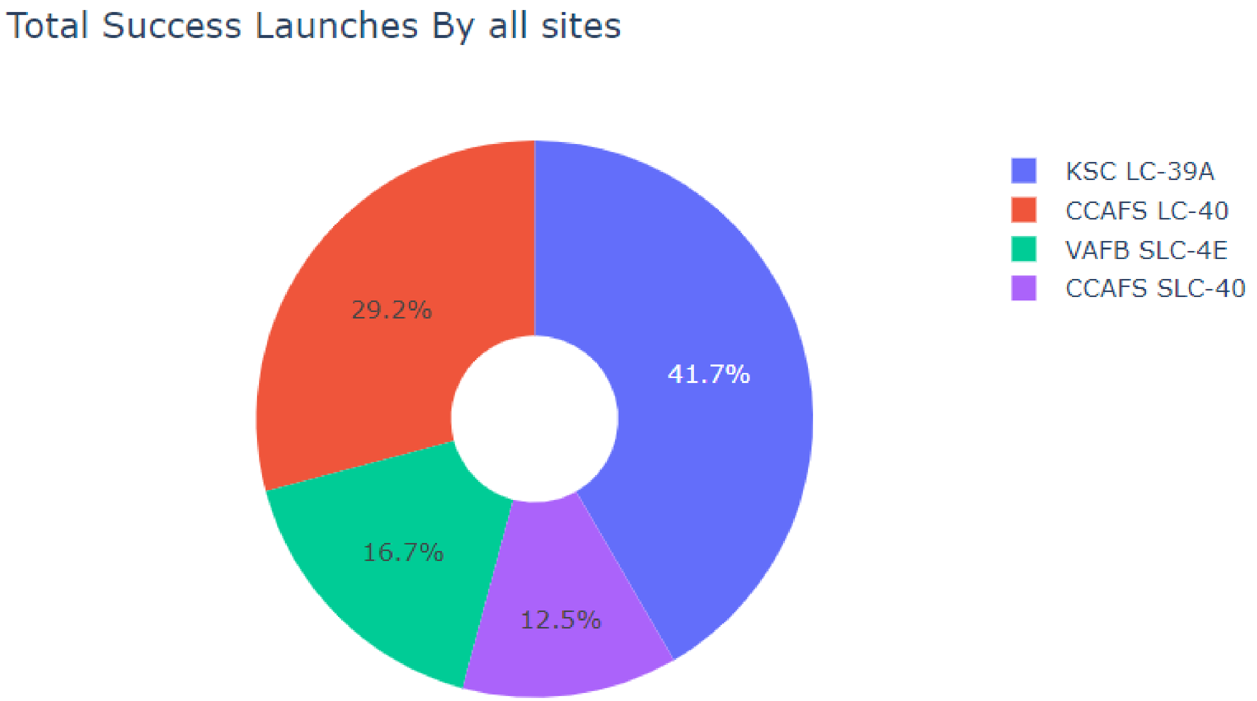
Section 5

Build a Dashboard with Plotly Dash



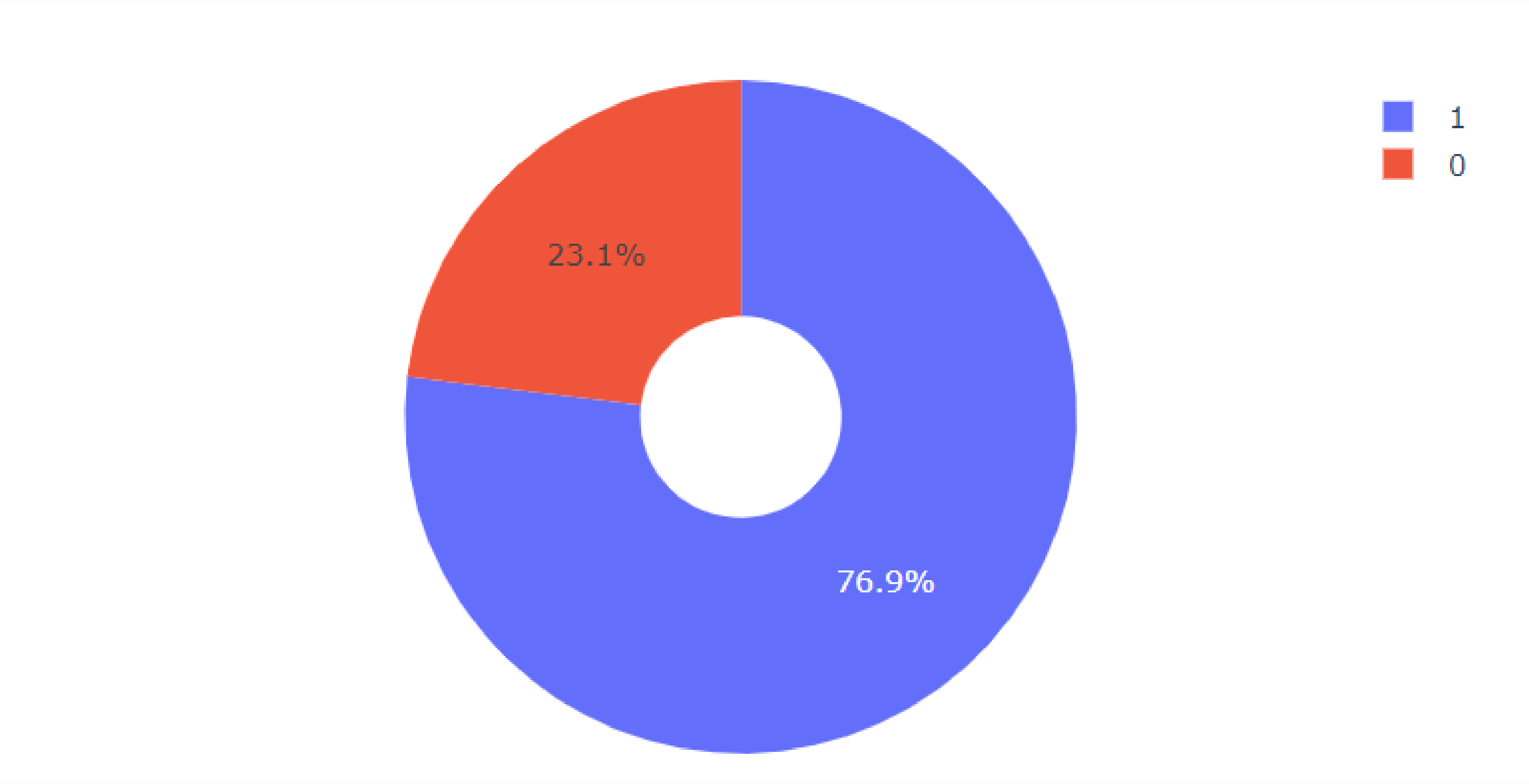
Successful Launches Across Launch Sites

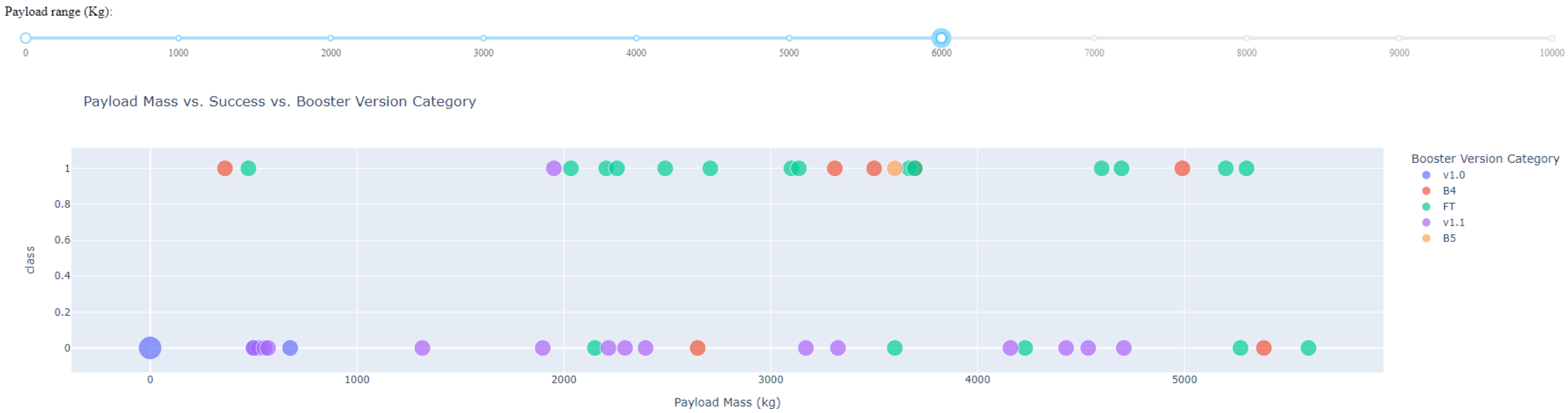
- ***We can see that KSC LC-39A had the***
- ***most successful launches from all the sites***



Highest Success Rate Launch Site

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





Payload Mass vs. Success vs. Booster Version Category

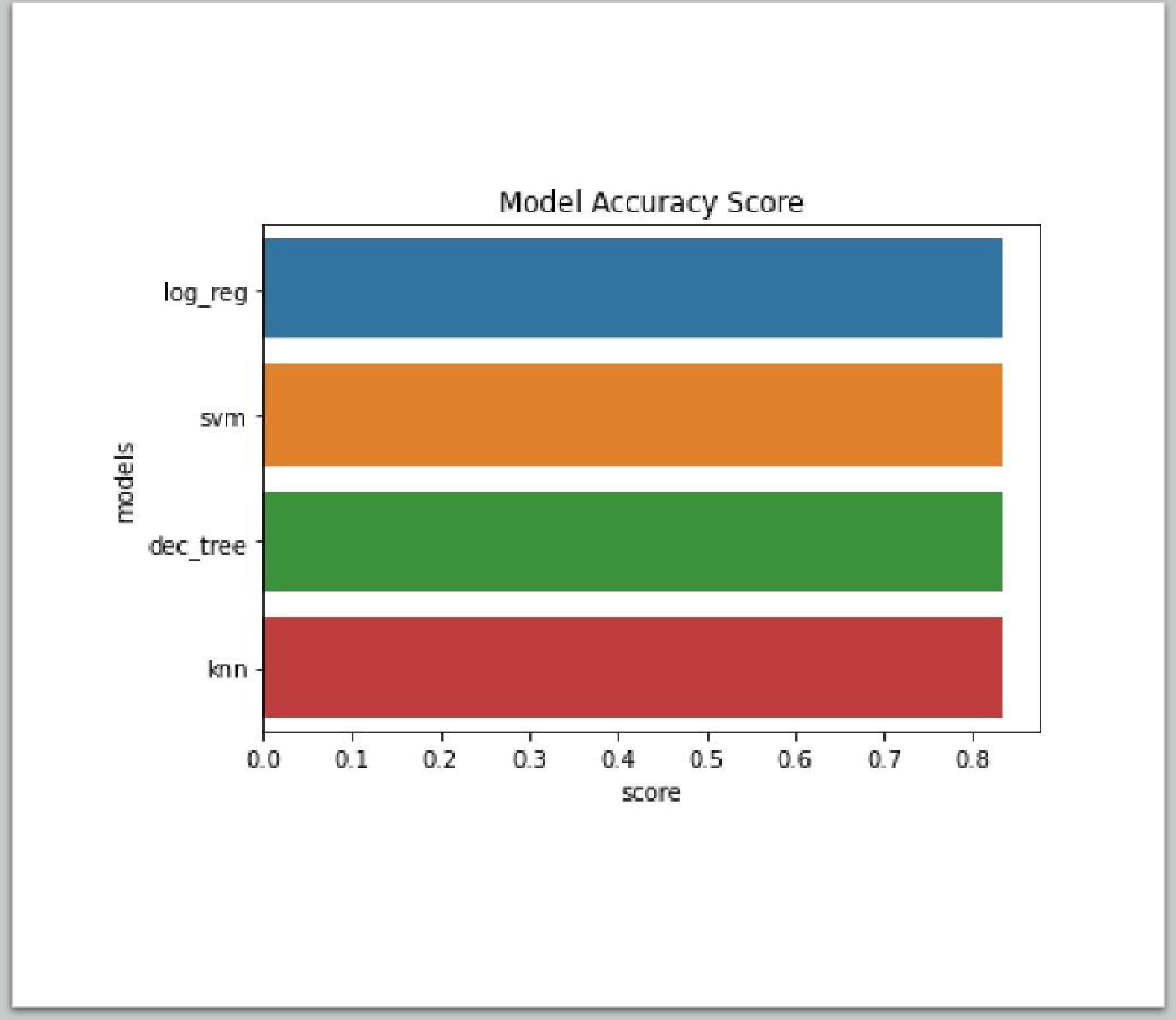
- Plotly dashboard has a Payload range selector. However, this is set from 0-10000 instead of the max Payload of 15600. Class indicates 1 for successful landing and 0 for failure. Scatter plot also accounts for booster version category in colour and number of launches in point size. In this particular range of 0-6000, interestingly there are two failed landings with payloads of zero.

Section 6

Predictive Analysis (Classification)

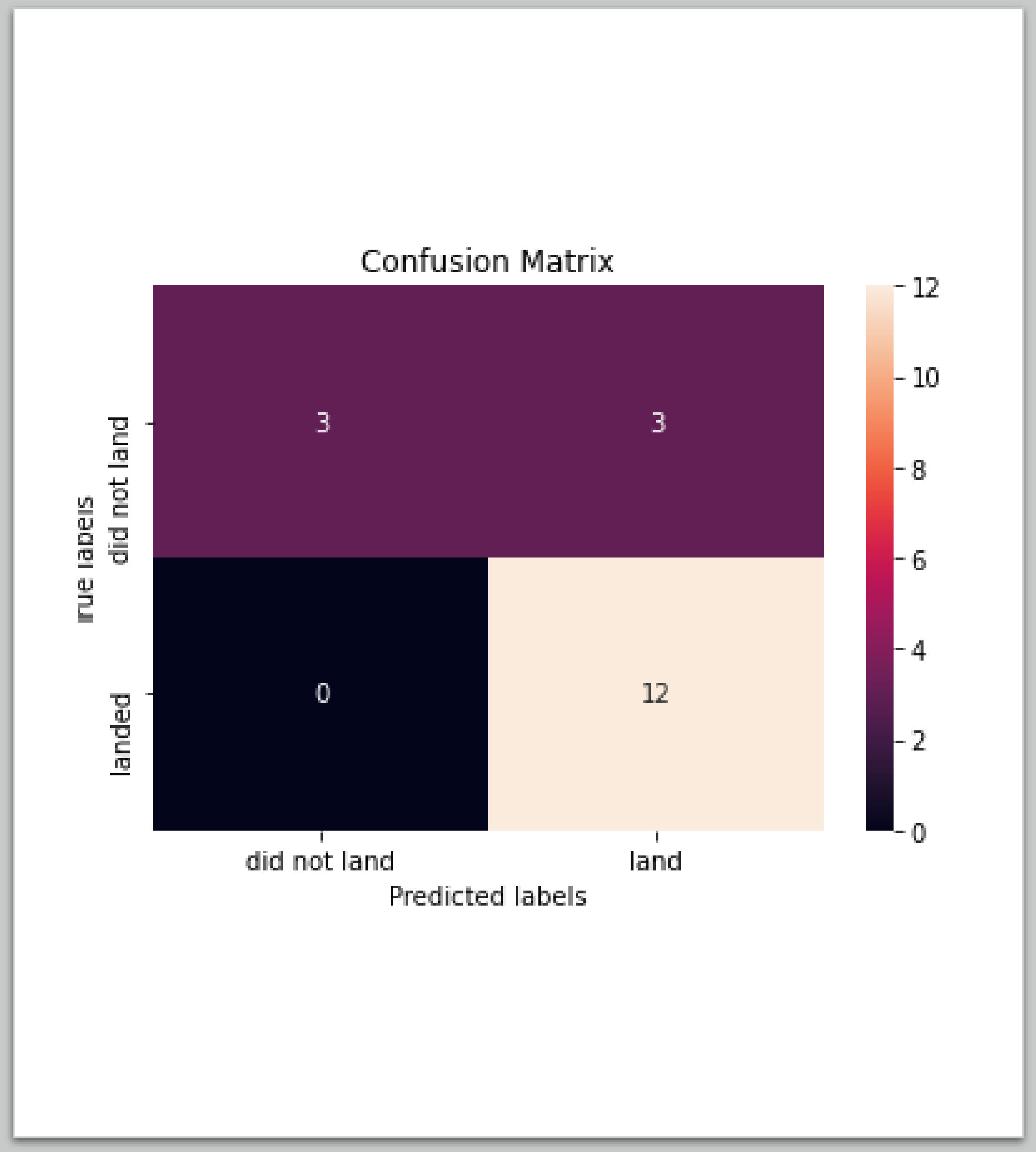
Classification Accuracy

- All models had virtually the same accuracy on the test set at 83.33% accuracy. It should be noted that test size is small at only sample size of18.
- This can cause large variance in accuracy results, such as those in Decision Tree Classifier model in repeatedruns.
- We likely need more data to determine the bestmodel.



Confusion Matrix

- Since all models performed the same for the test set, the confusion matrix is the same across all models. The models predicted 12 successful landings when the true labelwas successful landing.
- The models predicted 3 unsuccessful landings when the true label was unsuccessfullanding.
- The models predicted 3 successful landings when the true label was unsuccessful landings (false positives). Our models over predict successfullandings.



Conclusions

- The Tree Classifier Algorithm is the best for Machine Learning for this dataset
- Low weighted payloads perform better than the heavier payloads
- The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches
- We can see that KSC LC-39A had the most successful launches from all the sites
- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

Appendix

GitHub repository url:

- https://github.com/hafizroosly/IBM-SPACEX-CAPSTONE-

Instructors:

- **Instructors: RavAhuja, Alex Aklson, AijeEgwaikhide, Svetlana Levitan, Romeo Kienzler, PolongLin, Joseph Santarcangelo, Azim Hirjani, HimaVasudevan, SaishruthiSwaminathan, Saeed Aghabozorgi, Yan Luo**

Special Thanks to All Instructors:

- https://www.coursera.org/professional-certificates/ibm-datascience?#instructors

Thank you!