IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Magnus Urosev
1/1/2021

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection
  - Data wrangling
  - EDA data visualization
  - EDA SQL
  - Building an interactive map with Folium
  - Building a dashboard with Plotly Dash
  - Predictive analysis
- Summary of all results
  - EDA Results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

Introduction

- Project background and context
  - SpaceX has brought the future of commercial space flight into the present. We are trying to predict if their rocket, Falcon 9, will land successfully during its first stage. SpaceX 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 SpaceX can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

- Problems you want to find answers
  - What factors influence the success of a rocket launch?
  - Is there a correlation between each rocket variable and success of the landing rate?
  - What conditions are needed to achieve the best results?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX Rest API
  - Web scarping from [Wikipedia]
- Perform data wrangling
  - One-hot encoding data fields for ML and dropping irrelevant columns
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Plotting of scatter and bar graphs to show patterns between data
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected.
  - Data sets were collected with API requestions from the SpaceX API and web scraping from the previously mentioned Falcon and Falcon Heavy Launches Records Wikepedia page.
- You need to present your data collection process use key phrases and flowcharts

SpaceX API

Request data from the SpaceX API → The API returns data in .JSON → Normalize data into .CSV

Web Scraping

Get HTML response from Wikipedia → Extract data with Beautiful Soup → Normalize data into .CSV

Data Collection – SpaceX API

1. Requesting rocket launch data from SpaceX API with following URL

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

2. Decoding response as a .JSON to turn into Pandas data frame

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

3. Custom functions to clean data

```
# Call getBoosterVersion
getBoosterVersion(data)
```

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

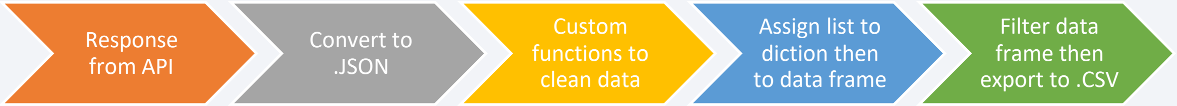
4. Constructing a data frame by combining columns into a dictionary

```
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}
```

5. Filtering data frame and exporting to .CSV

```
data_falcon9 = launch_df[launch_df['BoosterVersion'] == 'Falcon 9']
```

```
data_falcon9.to_csv('dataset_part\_1.csv', index=False)
```



Data Collection API Notebook

Data Collection - Scraping

1. Getting response from HTML

```
html_data = requests.get(static_url).text
```

2. Creating a BeautifulSoup object

```
soup = BeautifulSoup(html_data, 'html5lib')
```

3. Finding tables

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

4. Getting column names

```
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if(name != None and len(name) > 0):
        column_names.append(name)
```

5. Creating dictionary with keys

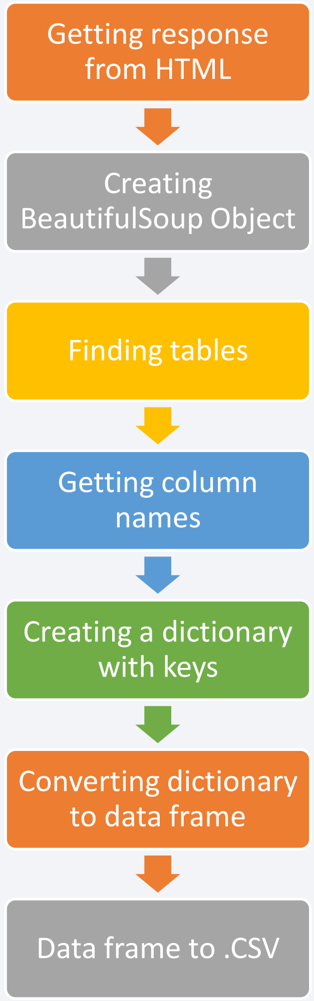
```
# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with
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'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. Appending data to keys (please refer to notebook)

7. Creating data frame then converting to .CSV

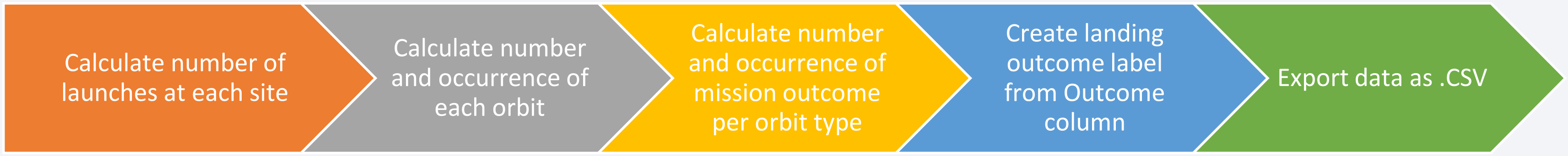
```
df=pd.DataFrame(launch_dict)
df.to_csv('spacex_web_scraped.csv', index=False)
```



Web scraping notebook

Data Wrangling

- There were several cases where the booster had failed to land. Sometimes, it had tried to land but failed due to accident. Below describes results of the missions:
    - True Ocean – Successfully landed in specific ocean region
    - False Ocean – Unsuccessfully landed in specific ocean region
    - True RTLS – Successfully landed on ground pad
    - False RTLS – Unsuccessfully landed on ground pad
    - True ASDS – Successfully landed on drone ship
    - False ASDS - Unsuccessfully landed on drone ship



[Data Wrangling Notebook]

Data Wrangling

1. Calculating the number of launches at each site

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

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

2. Calculating the number of occurrence of each orbit

```
df.Orbit.value_counts()
```

3. Calculating the number and occurrence of mission outcome per orbit type

```
landing_outcomes = df.Outcome.value_counts()
landing_outcomes
```

4. Creating a landing outcome label from outcome column

```
landing_class = []
for outcome in df.Outcome:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

```
df['Class']=landing_class
df[['Class']].head(8)
```

5. Calculating the success rate for every landing in dataset

```
df["Class"].mean()
```

```
0.6666666666666666
```

6. Exporting dataset to .CSV

```
df.to_csv("dataset_part\_2.csv", index=False)
```

EDA with Data Visualization

- Scatter Graphs:
  - Flight Number vs Payload Mass
  - Flight Number vs Launch Site
  - Payload vs Launch Size
  - Orbit vs Flight Number
  - Payload vs Orbit Type
  - Orbit vs Payload Mass

A scatter plot shows how one variable is affected by another. The relationship between two variables is called their correlation. Scatter plots usually consist of a large body of data.

- Bar Charts:
  - Mean vs Orbit

A bar chart makes it easy to compare sets of data between different groups. One axis is a discrete value and the other represent categories. Its purpose is to show relationships between the two axes.

- Line Graph
  - Success Rate vs Year

Line graphs shows data variables and trends very clearly, which can help to make predictions about the results of data not yet recorded.

[Notebook]

EDA with SQL

- **The following queries were used to get information from the dataset:**
    - Displaying the names of the unique launch sites in the space mission.
    - Displaying 5 records where launch sites began with the string ‘CCA.’
    - Displaying the total payload mass carried by boosters launched by NASA (CRS).
    - Displaying the average payload mass carried by booster version F9 v1.1.
    - Listing dates when the first successful landing outcome in ground pad was achieved.
    - Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
    - Listing the total number of successful and failure mission outcomes.
    - Listing the names of the booster_versions which have carried the maximum payload mass.
    - Listing the failed landing_outcomes in drone ship, their booster version, and launch site names for 2015.
    - Ranking the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

Notebook

Build an Interactive Map with Folium

- Visualization of Launch Data in an Interactive Map
  - The coordinates of each launch site was added onto the map as a circle marker with the name and launch site.
  - Colored markers were then added to show successful (green) and failed (red) launches for each site.
  - Lines were added to show the distance from the launch site to different landmarks to find trends.
  - Some trends that were drawn:
    - 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 coastlines? Yes
    - Do launch sites keep certain distances away from cities? Yes

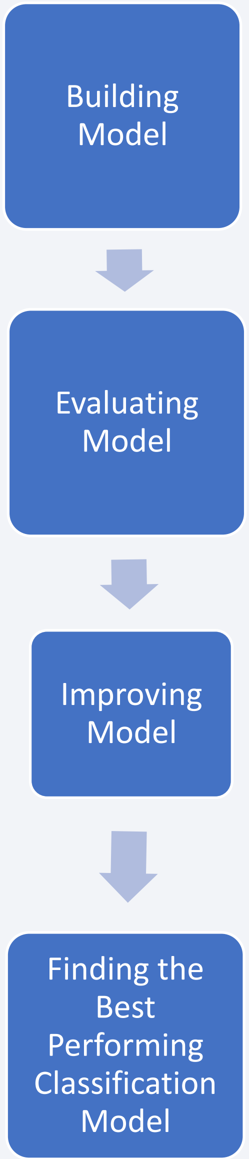
[Notebook and maps]

Build a Dashboard with Plotly Dash

- A dashboard was created to visualize the findings.
  - A pie chart was created to show the number of successful launches by each site. This chart was selected to show the distribution of successful launches across all sites.
  - A scatter plot was created to show the relationship between outcomes and payload mass of the boosters. A scatterplot was chosen to show any correlations or trends between outcomes and payloads.

Predictive Analysis (Classification)

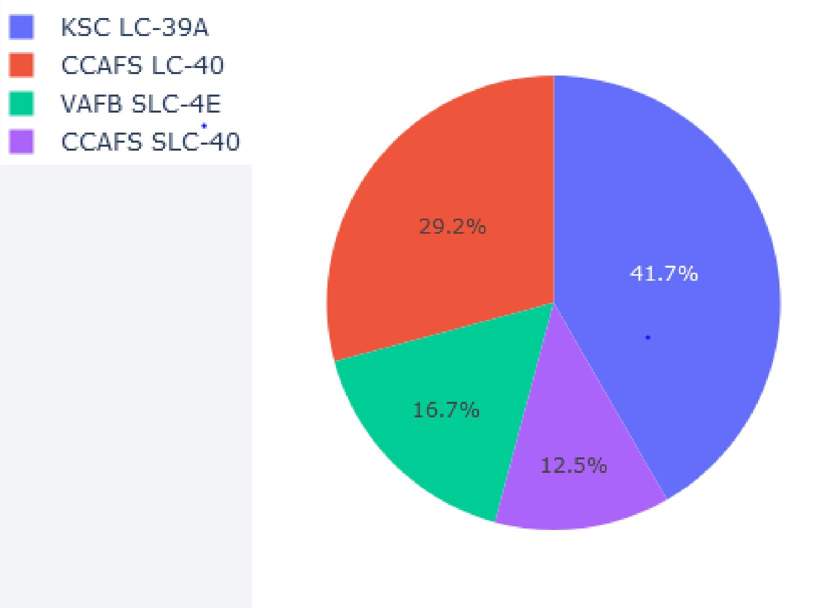
- Building Model
  - Load data set and transform data. Then split data into training and test data sets. Decide which ML algorithms to use. Set parameters then train dataset.
- Evaluating Model
  - Check accuracy of each model and tune hyperparameters. Plot confusion matrix.
- Improving Model
  - Tune algorithm.
- Finding the Best Performing Classification Model
  - Find which method performs the best using the test data.

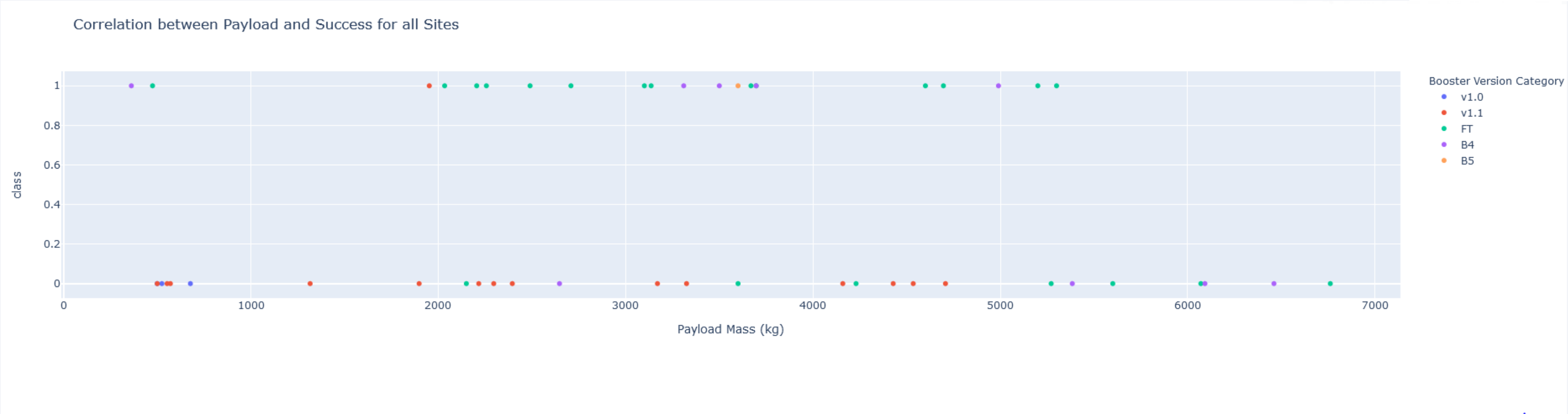


Notebook

Results

- Results of the launch records and correlation between payload and success for all sites. Information taken from the SpaceX Launch Records Dashboard.



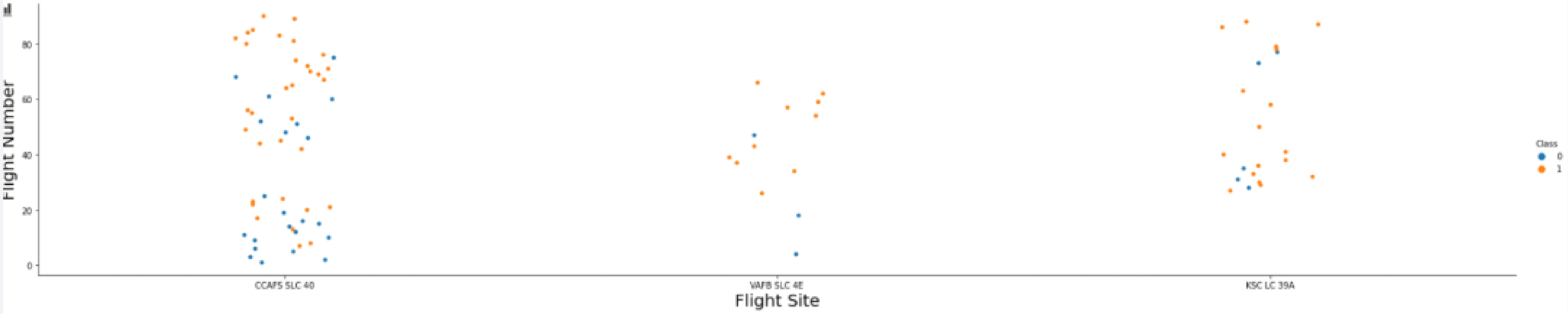


Section 2

Insights drawn from EDA

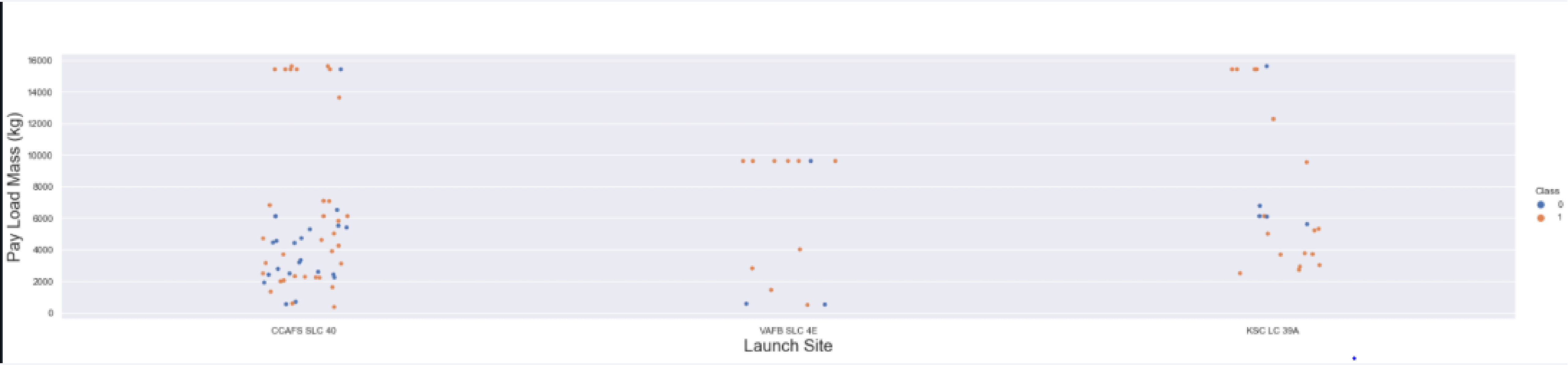
Flight Number vs. Launch Site

- The higher the number of flights at a launch site, there was a greater success rate at the launch site.



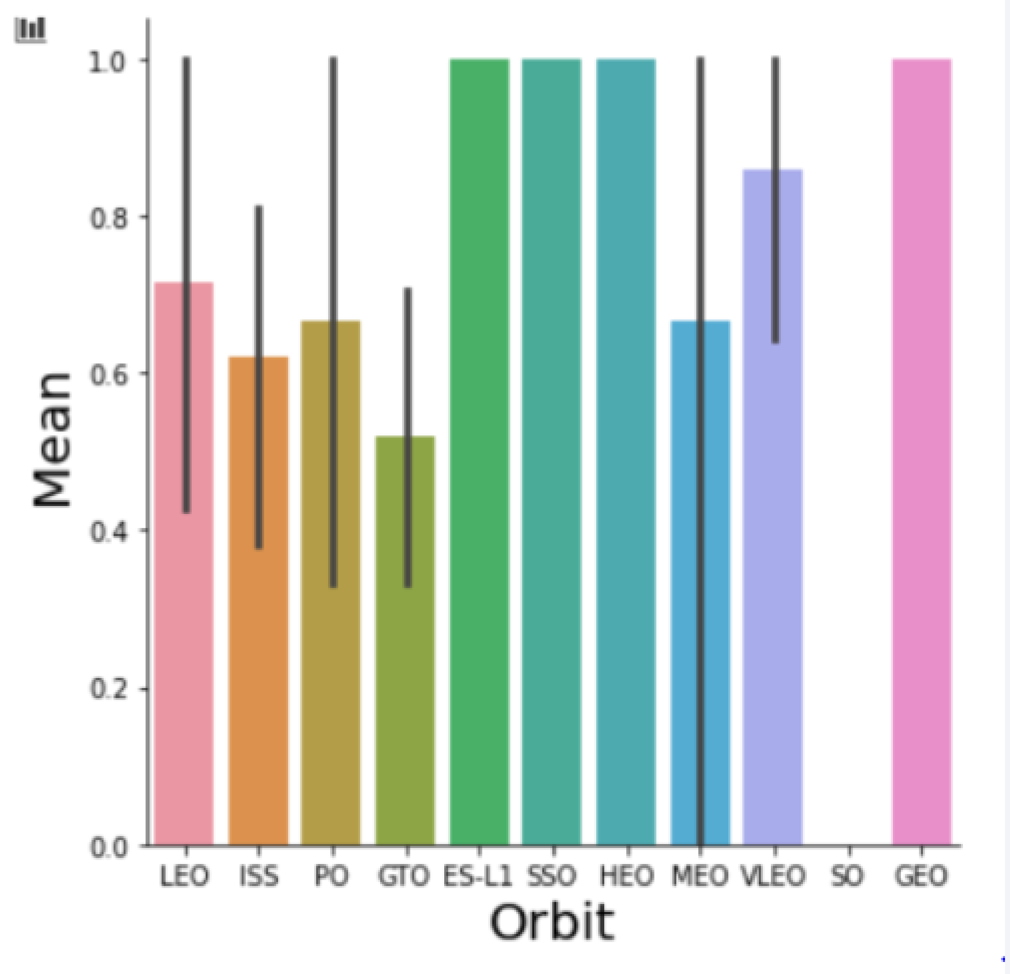
Payload vs. Launch Site

- If the payload had a greater mass at the CCAFS SLC 40 launch site, then there would be a higher success rate for the rocket.



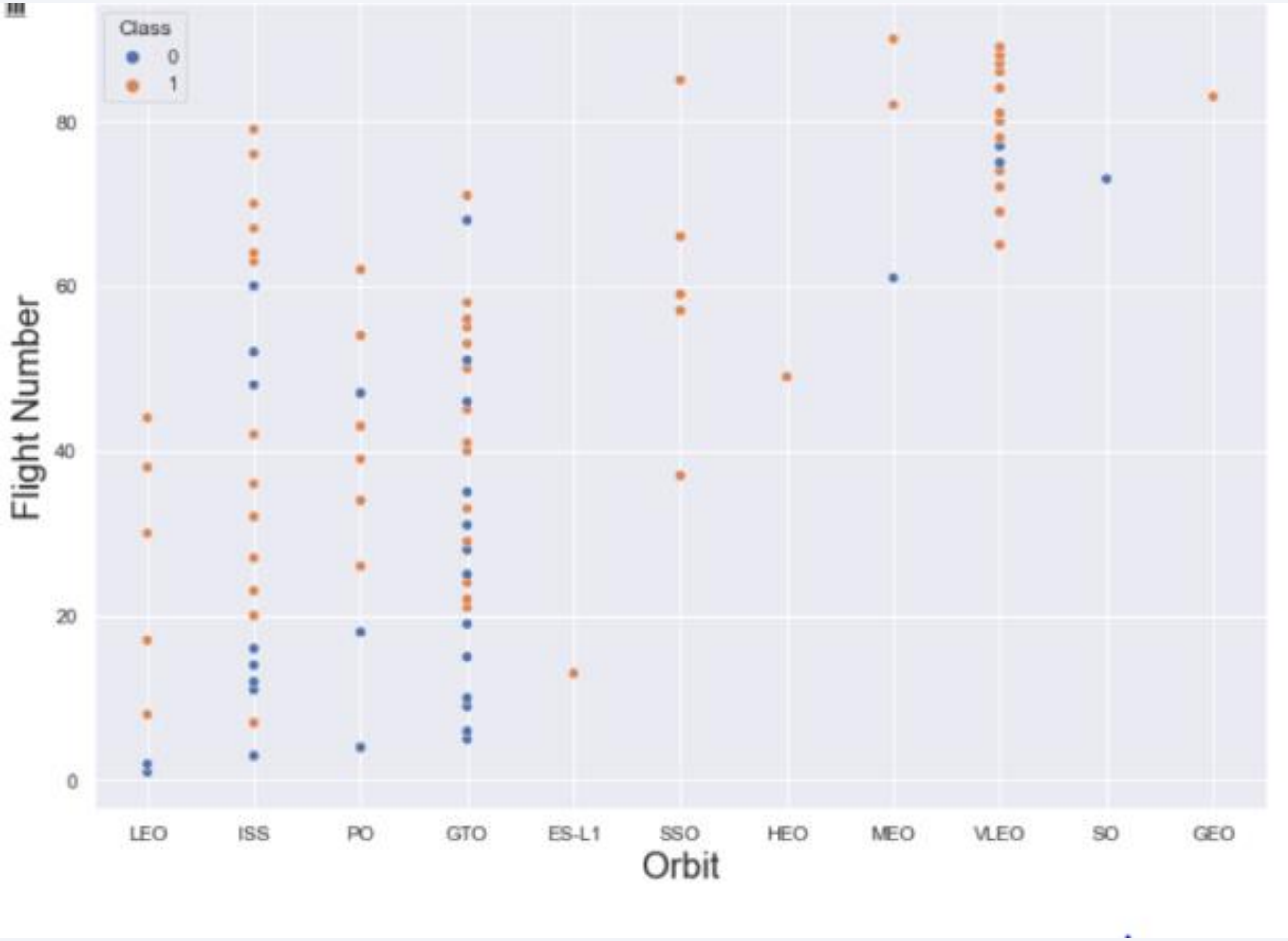
Success Rate vs. Orbit Type

- Orbits: GEO, HEO, SSO, and ES-L1 had the best success rates.



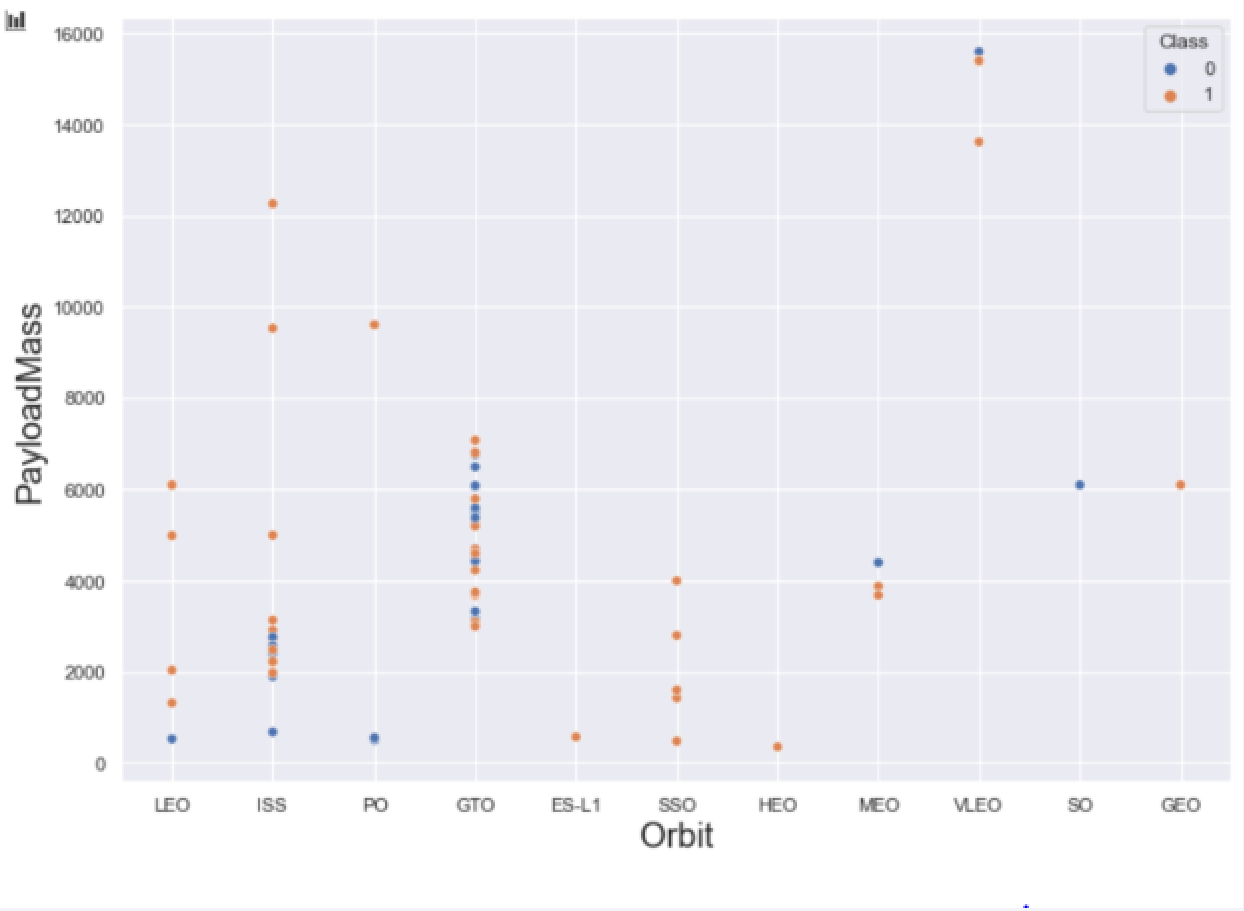
Flight Number vs. Orbit Type

- Blue = Unsuccesful
- Orange = Successful
- When looking at the chart, it seems that LEO's success rate has a correlation with the number of flights.
- For GTO, it appears it has no correlation with number of flights.



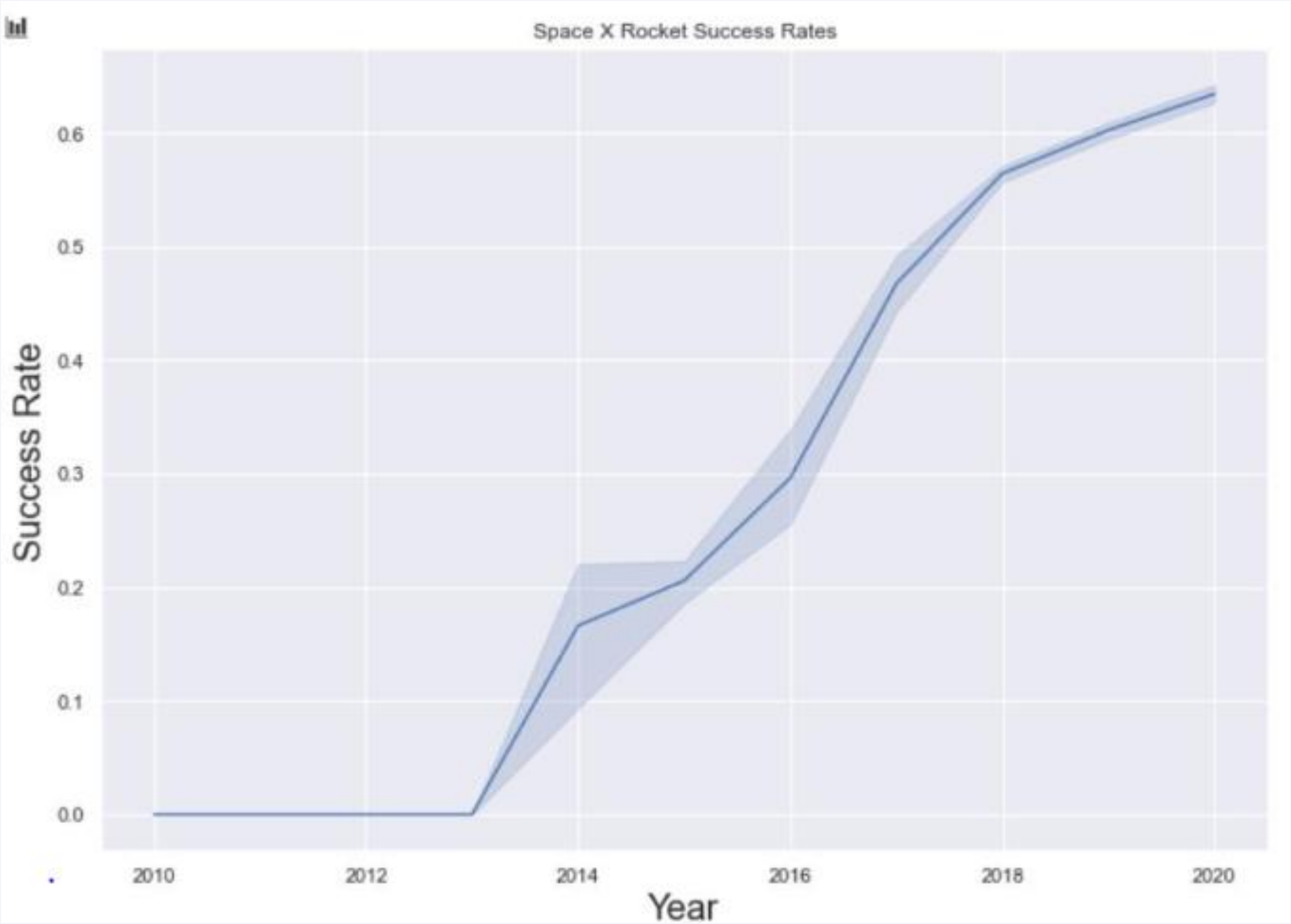
Payload vs. Orbit Type

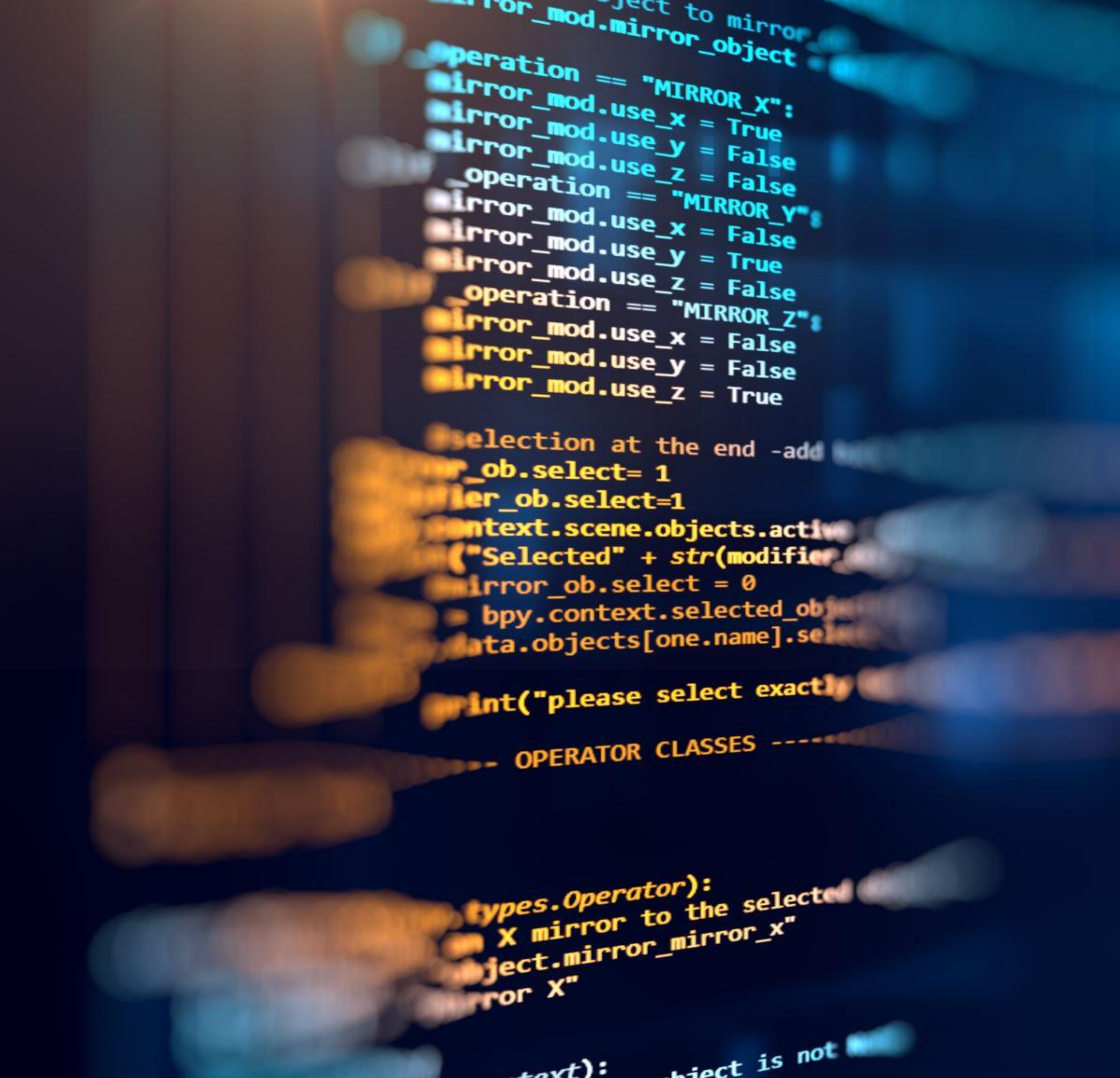
- At a glance, you can see that heavy payloads have a negative relationship with GTO orbits, but a positive relationship with LEO.



Launch Success Yearly Trend

- Launch success had been continually increasing since 2013.





Section 3

EDA with SQL

All Launch Site Names

- Used a SELECT DISTINCT in the query to retrieve unique values from the launch_site column.
- The four unique sites are:
  - CCAFS LC-40
  - CCAFS SLC-40
  - KSC LC-39A
  - VAFB SLC-4E

```
%%sql
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL
```

```
 * ibm_db_sa://gqt78196:***@
nrk39u98g.databases.appdomai
Done.
```

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

Launch Site Names Begin with 'CCA'

- Retrieved only five rows using the LIMIT 5 query then used LIKE along with % to call sites with CAA.

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

```
 * ibm_db_sa://gqt78196:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0ngnrk39u98g.databases.appdomain.cloud:32286/bludb
Done.
```

DATE	time__utc_	booster_version	launch_site	PAYLOADMASS_KG_	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	None	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	None	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	None	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	None	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	None	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Calculated payload mass by using SUM(PAYLOAD_MASS__KG_) with WHERE clause to only include NASA CRS.

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) AS total_payload_mass_kg
FROM SPACEXTBL
WHERE CUSTOMER = 'NASA (CRS)'
```

```
 * ibm_db_sa://gqt78196:***@1bbf73c5-d84a-4bb0-85b9-ab1a
bludb
Done.
```

total_payload_mass_kg
45596

Average Payload Mass by F9 v1.1

- Used AVG(PAYLOAD_MASS__KG_) with WHERE clause to calculate f9 v1.1 mass.

```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) AS avg_payload_mass_kg
FROM SPACEXTBL
WHERE BOOSTER_VERSION = 'F9 v1.1'
```

```
 * ibm_db_sa://gqt78196:***@1bbf73c5-d84a-4bb0-85b9-ab
bludb
Done.
```

avg_payload_mass_kg
2928

First Successful Ground Landing Date

- Used the MIN(DATE) function to select the earliest date along with WHERE Lading-outcome ‘Success’ to find the first successful landing.

```
%%sql
SELECT MIN(DATE) AS first_successful_landing_date
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (ground pad)'
```

```
 * ibm_db_sa://gqt78196:***@1bbf73c5-d84a-4bb0-85b
bludb
Done.
```

first_successful_landing_date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Used the WHERE Landing_outcome = ‘Success (drone ship)’ to only select drone ships along with AND… …BETWEEN 4000 and 6000) to select conditions for payload mass.

```
%%sql
SELECT BOOSTER_VERSION
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (drone ship)'
    AND (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000)
```

```
 * ibm_db_sa://gqt78196:***@1bbf73c5-d84a-4bb0-85b
bludb
Done.
```

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

Total Number of Successful and Failure Mission Outcomes

- Used COUNT(*) to find the total number of columns. Then sued GROUP BY MISSION_OUTCOME to find the total number of each mission outcome.

```
%%sql
SELECT MISSION_OUTCOME, COUNT(*) AS total_number
FROM SPACEXTBL
GROUP BY MISSION_OUTCOME
```

```
 * ibm_db_sa://gqt78196:***@1bbf73c5-d84a-4bb0-85b
bludb
Done.
```

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

Boosters Carried Maximum Payload

- Used MAX() with WHERE to find boosters that carried the maximum payload, which was F9 B5 B10.

```
%%sql
SELECT DISTINCT BOOSTER_VERSION, PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (
    SELECT MAX(PAYLOAD_MASS__KG_)
    FROM SPACEXTBL);
```

```
 * ibm_db_sa://gqt78196:***@1bbf73c5-d84a-4bb0-85b9
bludb
Done.
```

booster_version	payload_mass__kg_
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

2015 Launch Records

- Used WHERE Landing_outcome = ‘Failure (drone ship)’ AND YEAR(DATE) = ‘2015’ to find all drone ship failures in 2015.

```
%%sql
SELECT LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND YEAR(DATE) = '2015'
```

```
 * ibm_db_sa://gqt78196:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cm
bludb
Done.
```

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- Used a WHERE DATE clause to select dates between 2010-06-04 and 2017-03-20 along with GROUP BY Landing_outcome and DESC to list the number of successful and failed landing outcomes and rank them.

```
%%sql
SELECT LANDING__OUTCOME, COUNT(LANDING__OUTCOME) AS total_number
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY LANDING__OUTCOME
ORDER BY total_number DESC
```

```
 * ibm_db_sa://gqt78196:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.
bludb
Done.
```

landing__outcome	total_number
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

Section 4

Launch Sites Proximities Analysis

All Launch Site Locations

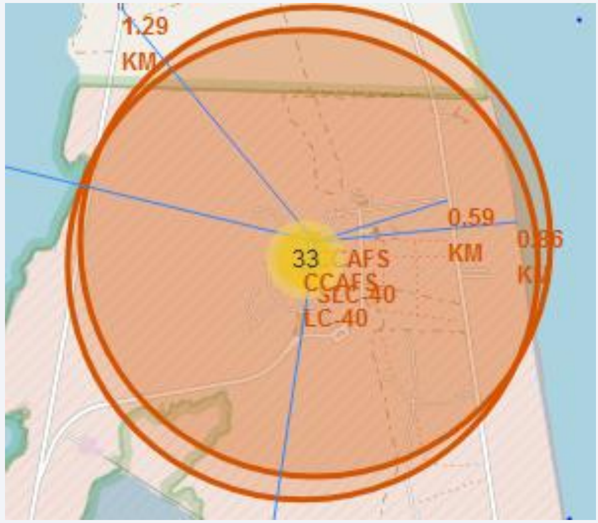
- On the map is all SpaceX launch sites. As you can see, all sites are on the coast.

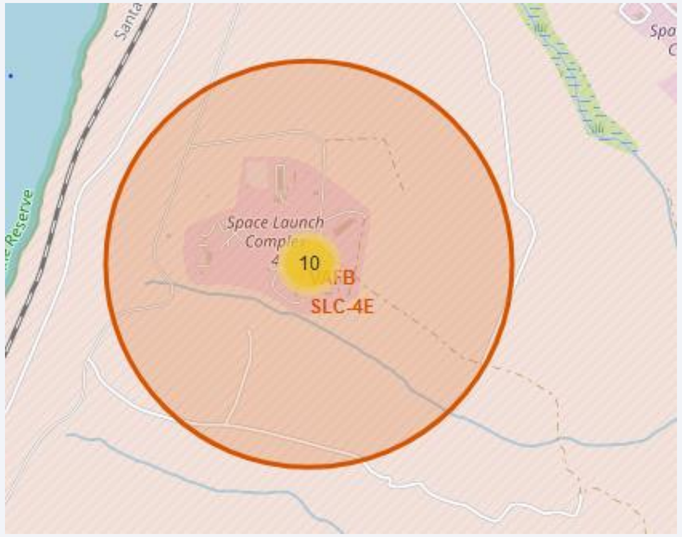
Color Labeled Launch Outcomes

- Each launch was labeled and colored (green = success, red = unsuccessful). Each icon can be clicked to get more information about the launch.

Distance of Launch Sites to Certain Objects

- Each launch site has a proximity that is must keep away from railways, highways and cities. The circle shows the proximity and each line shows the distance it is keeping away from certain objects.



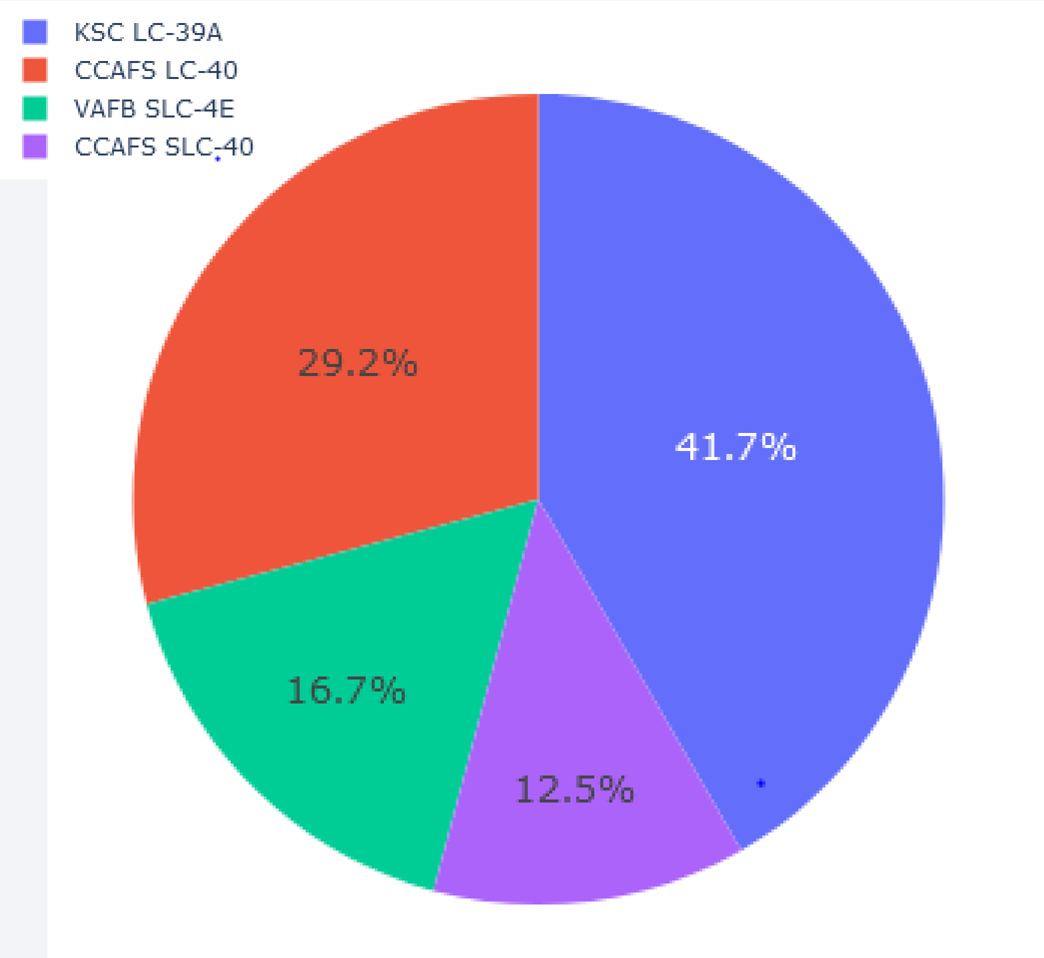


Section 5

Build a Dashboard with Plotly Dash

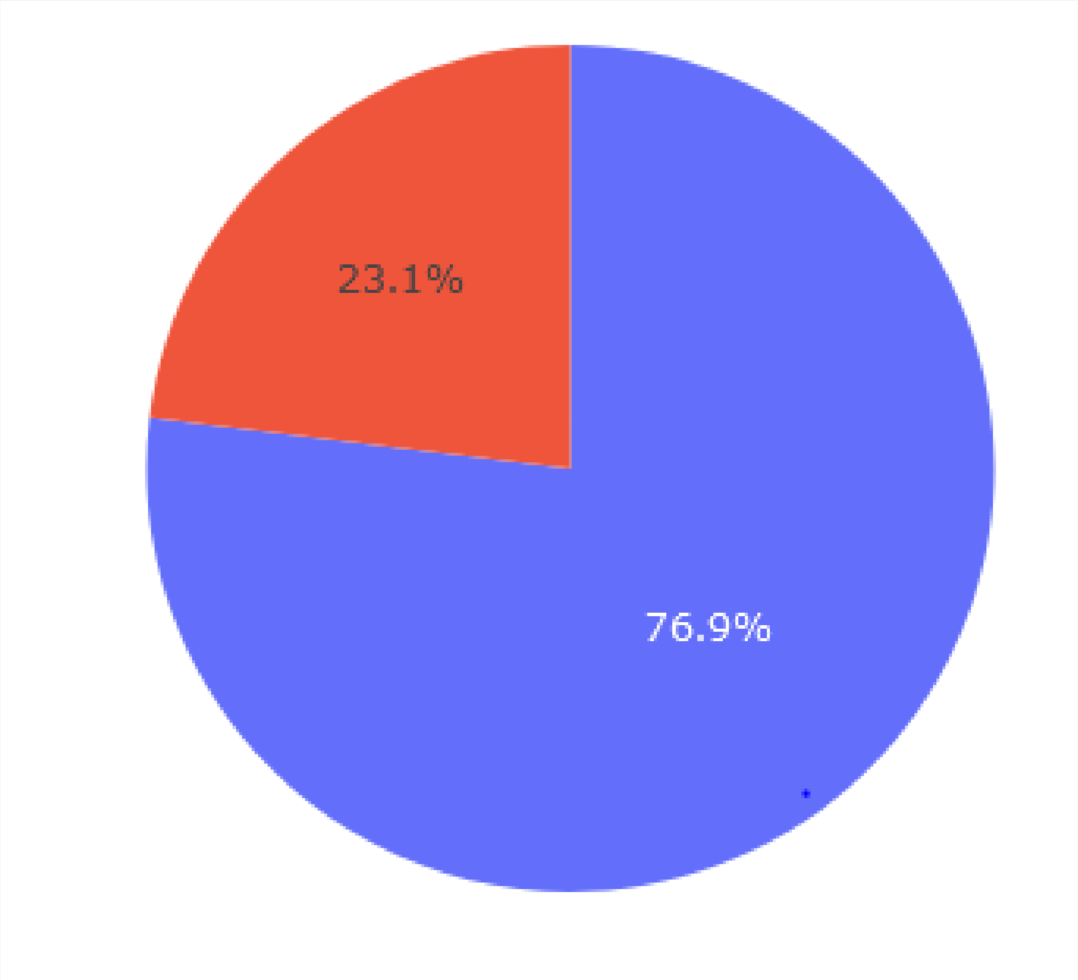
Distribution of Success of all Launch Sites

- KSC LC-39A had the most success out of all the launch sites (41.7%).



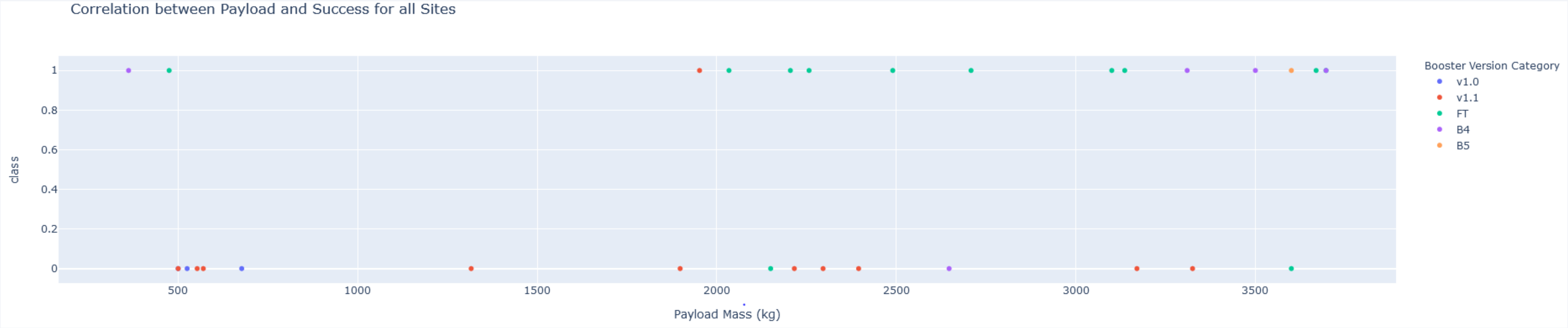
Highest Launch Success Ratio

- KSC LC-39A was successful 76.9% of the time (blue = successful, red = unsuccessful).



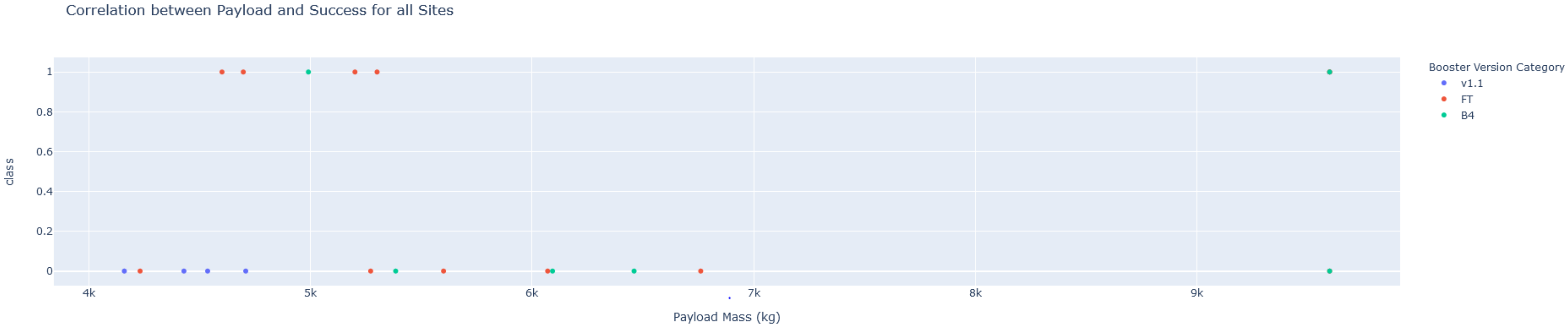
Payload vs Outcome Scatter 0 – 4,000 kg

- The success rate for lighter payloads is higher than heavier payloads.



Payload vs Outcome Scatter 4,000 – 10,000 kg

- The success rate for lighter payloads is higher than heavier payloads.



Section 6

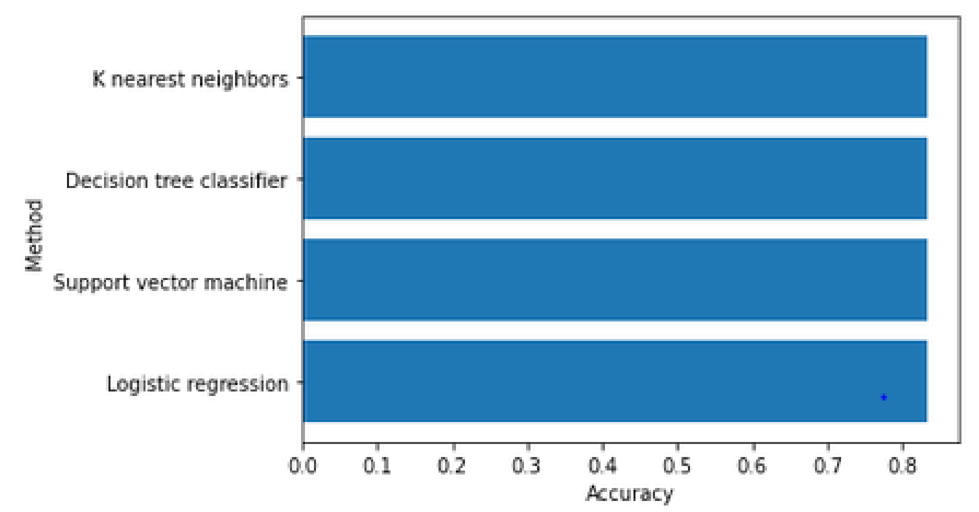
Predictive Analysis (Classification)

Classification Accuracy

- All models were the same at 83.33% This may be due to the small test size. More testing may be needed to find the best model.

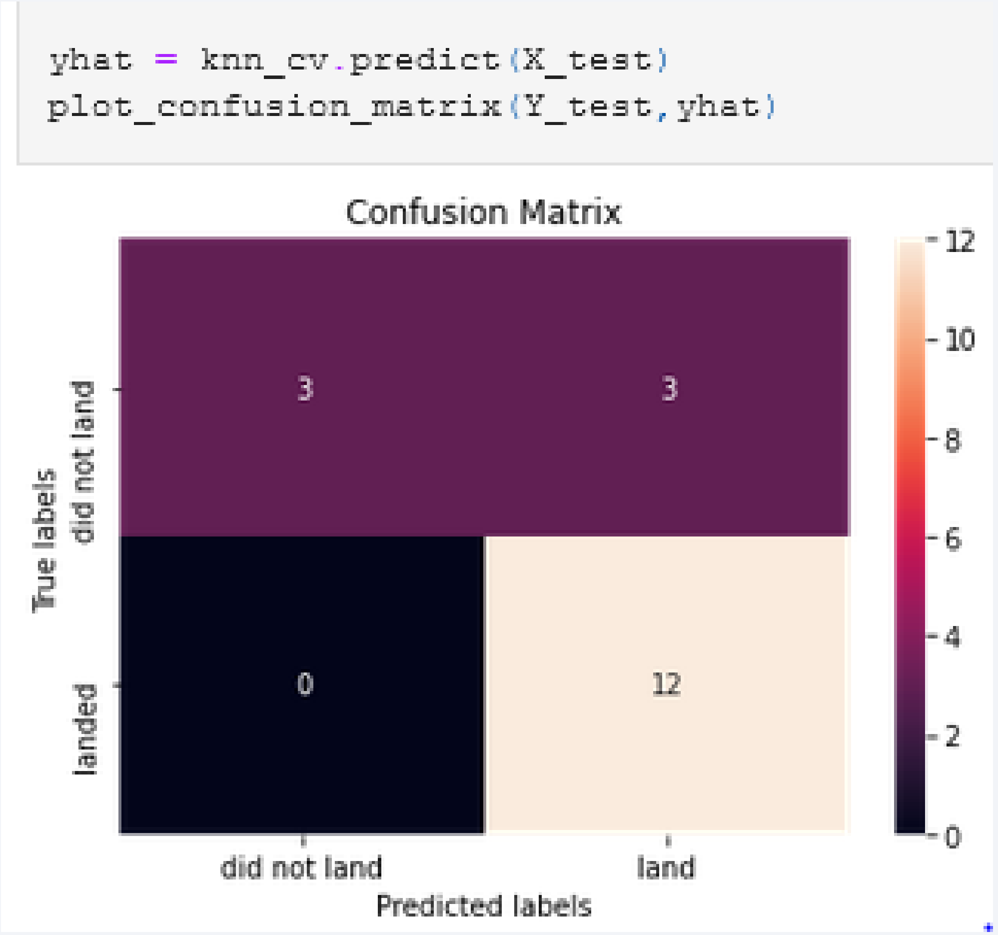
```python
import numpy as np
import matplotlib.pyplot as plt

plt.barh(methods, accuracy)
plt.xlabel('Accuracy')
plt.ylabel('Method')
plt.show()
```



Confusion Matrix

- The confusion matrix was the same for all the models. From this, it had predicted 12 successful landings, 3 failed landings, and 3 successful landings which were false positives.



Conclusions

- The success rates increased with the number of flights.
- The highest success rates were with the SSO, HEO, GEO, and ES-L1 orbits.
- KSLC-39A had the best success among all sites.
- Low weighted payloads had better success than heavier payloads.
- All models seemed to have had the same accuracy due to the small data set.

Appendix

- Notebook repository:
- https://github.com/murosev/IBM-SpaceX-Capstone-Project

Thank you!