IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Cole Moebius
9 September 2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection: Web Scraping and API
  - Data Wrangling
  - EDA with SQL
  - Data Visualization
  - Folium Maps
  - ML Prediction

- Summary of all results
  - Exploratory Data Analysis Results
  - Folium Maps
  - Machine Learning Predictions

Introduction

SpaceX is a privately owned space faring company, offering rocket launches with the flagship rocket model Falcon 9. SpaceX offers a much cheaper launch than competitors, as low as $62,000,000, in part due to the reuse of the first stage launch parts. In this project, I will assume the role of a Data Scientist for a rival company SpaceY, with the goal to create a data ingestion pipeline which will be later used to predict the landing outcome of first stage launches.

The problems I seek to address:

- What factors influence landing outcome?
- What is relationship between these factors?
- What are ideal conditions of a successful landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using the SpaceX Rest API and web scraping on Wikipedia.
- Perform data wrangling
  - Data processed via one-hot encoding for launch success/fail
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

The data collection process can be summarized by gathering information on specific metrics called variables, which allows for insights that answer questions related to data.

The data was collected using REST API, by pulling from a get request, decoding the response into Json and converting that into a pandas data frame, and finally by filling in missing values.

Web Scraping from Wikipedia was also done using BeautifulSoup, to parse HTML tables into a data frame.

Data Collection – SpaceX API

1. Get launch data from API request
2. Convert json result to data frame
3. Perform data cleaning and account for missing values using mean.

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

response = requests.get(spacex_url)
```

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

```
# Lets take a subset of our dataframe keeping only the featur
data = data[['rocket', 'payloads', 'launchpad', 'cores',
            'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are f
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also e
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Source: https://github.com/colemoebius2000/SpaceXAnalysis/blob/24117b3854e1fab04c3ba994aae1303a26ada8c5/API.ipynb

Data Collection - Scraping

1. Request Falcon9 Wiki page from url.
2. Create BeatifulSoup object from HTML response.
3. Extract column and variable names from header.

Source: https://github.com/colemoebius2000/SpaceXAnalysis/blob/24117b3854e1fab04c3ba994aae1303a26ada8c5/WebScraping.ipynb

```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

Create a `BeautifulSoup` object from the HTML `response`

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

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            launch_dict['Flight No.'].append(flight_number)
            #print(flight_number)
            datatimelist=date_time(row[0])
```

Data Wrangling

Data wrangling is the process of cleaning and combining messy data in order to conduct exploratory analysis.

First, we calculate the number of launches on each site and then the occurrences of mission outcomes per orbit type.

A landing outcome label is created in the data frame using one hot encoding to assist in the machine learning process.

```
num_launch = df['LaunchSite'].value_counts()
print(num_launch)
```

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

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for o in df['Outcome']:
    print(o)
    if o in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
print(landing_class)
```

Source:
https://github.com/colemoebius2000/SpaceXAnalysis/blob/24117b3854e1fab04c3ba994aae1303a26ada8c5/Data%20Wrangling.ipynb

EDA with Data Visualization

Through categorical scatterplots, bar charts, and line plots, we are able to find the relationship between factors such as:

- Payload and Flight Number
- Flight Number and Launch Site
- Flight Number and Orbit Type
- Payload and Orbit Type
- Payload and Launch Site

Using these plots, we can determine which factors lead to the most successful launches.

Source: https://github.com/colemoebius2000/SpaceXAnalysis/blob/24117b3854e1fab04c3ba994aae1303a26ada8c5/Pandas%20EDA.ipynb

EDA with SQL

Using SQL queries, we want to continue exploring the data set.
We find:

- The names of unique Launch Sites.
- The launch sites beginning with ‘CCA’
- The total payload mass
- The average payload mass carried by booser F9 v1.1
- The date of first successful landing outcome.
- The names of boosters with successful landings with a payload mass between 4000 and 6000 kg.
- And many more.

Source: https://github.com/colemoebius2000/SpaceXAnalysis/blob/24117b3854e1fab04c3ba994aae1303a26ada8c5/SQL%20EDA.ipynb

Build an Interactive Map with Folium

We then use Folium to visualize the launch data with a map. Taking the longitude and latitude of each launch site, we add markers and a label of each name. The prior one hot encoding of launch outcomes then colors the markers red or green based on success or failure at that launch site.

Using these maps we hope to discover:

- How close launch sites are to coastlines, highways, railways and cities?
- If they are, then why?

Source: https://github.com/colemoebius2000/SpaceXAnalysis/blob/24117b3854e1fab04c3ba994aae1303a26ada8c5/Folium.ipynb

Build a Dashboard with Plotly Dash

- To display real time information and analysis, we create an interactive dashboard using Plotly Dash.
- Pie charts are used to show the popularity of launch sites.
- Scatter plots are used to show the relationship between Landing Outcome and Payload Mass (kg) for each booster version.

Source: https://github.com/colemoebius2000/SpaceXAnalysis/blob/24117b3854e1fab04c3ba994aae1303a26ada8c5/spacex_dash_app.py

Predictive Analysis (Classification)

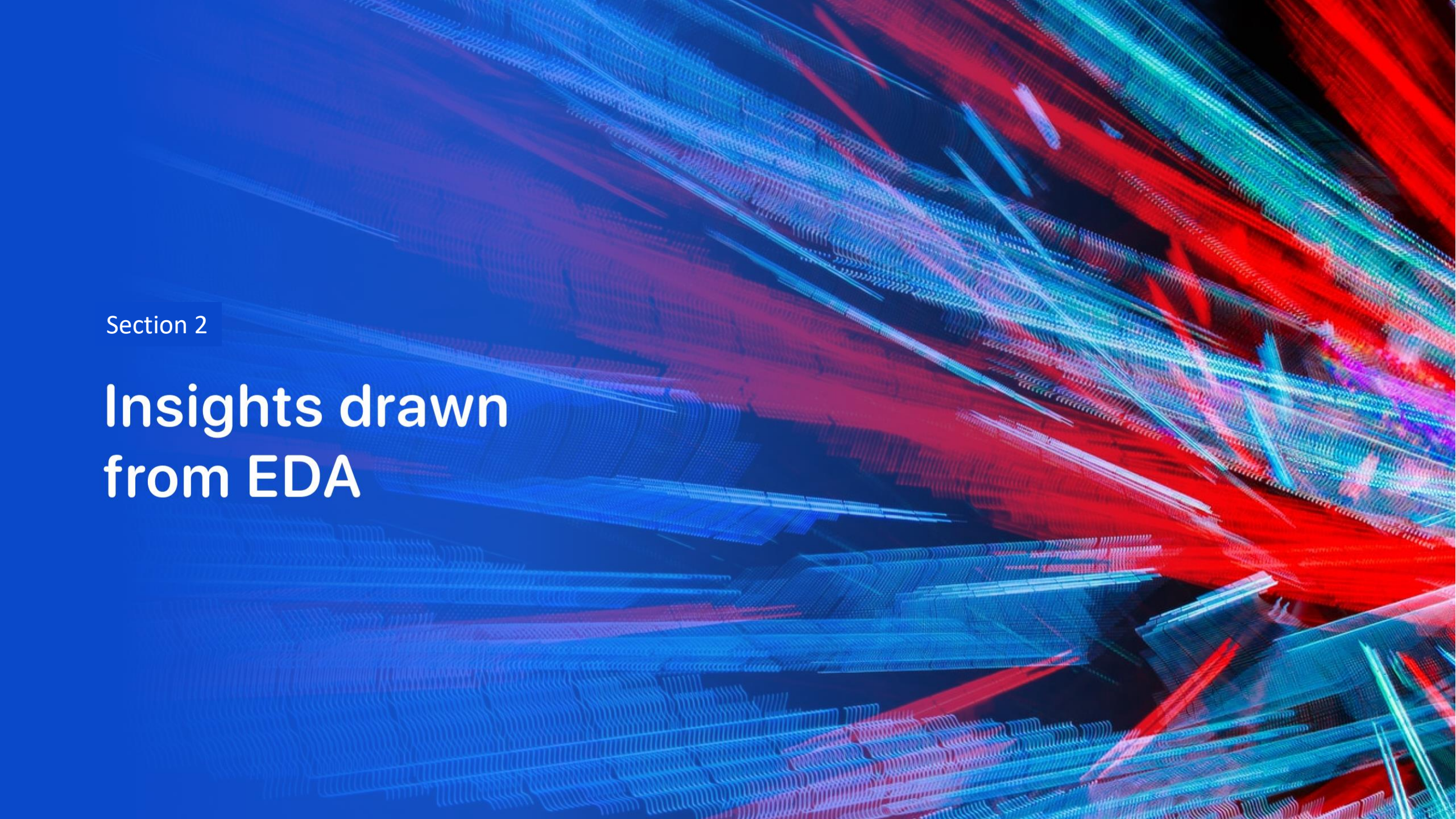
1. Building the Model:
    - Load dataset into Pandas
    - Transform data and split into train and test sets
    - Select one of four potential ML algorithms
2. Evaluating Model:
    - Display model accuracy using model.Score()
    - Tune hyperparameters
    - Plot and interpret confusion matrix
3. Improving Model:
    - Feature and algorithm tuning
4. Choose the best Model:
    - Plot and compare each confidence score.

Source: https://github.com/colemoebius2000/SpaceXAnalysis/blob/24117b3854e1fab04c3ba994aae1303a26ada8c5/Machine%20Learning%20Predictions.ipynb

Results

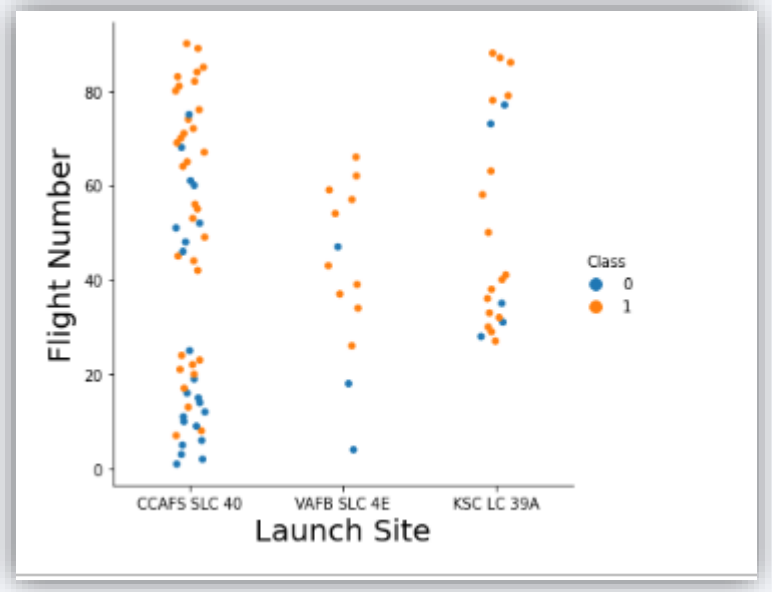
The results will be presented in 3 parts:

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


Section 2
Insights drawn from EDA

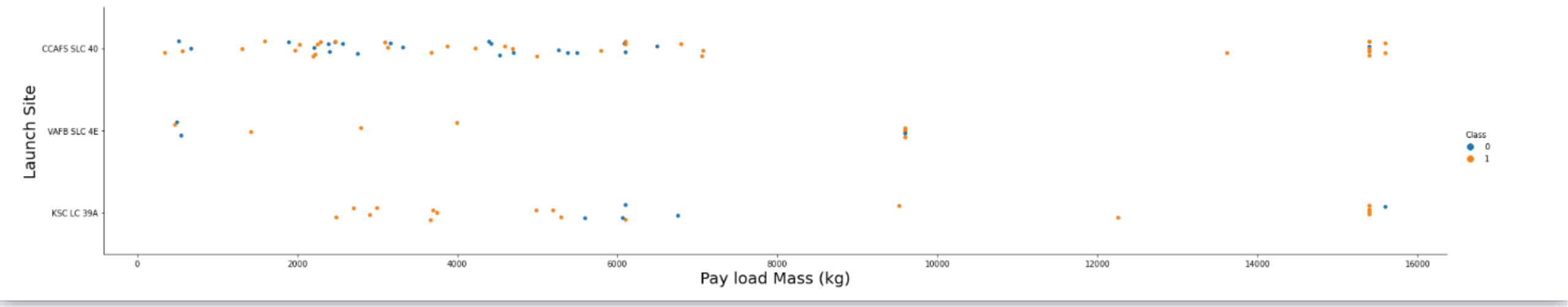
Flight Number vs. Launch Site

The scatter plot shows that higher flight numbers typically correlates to a successful launch.



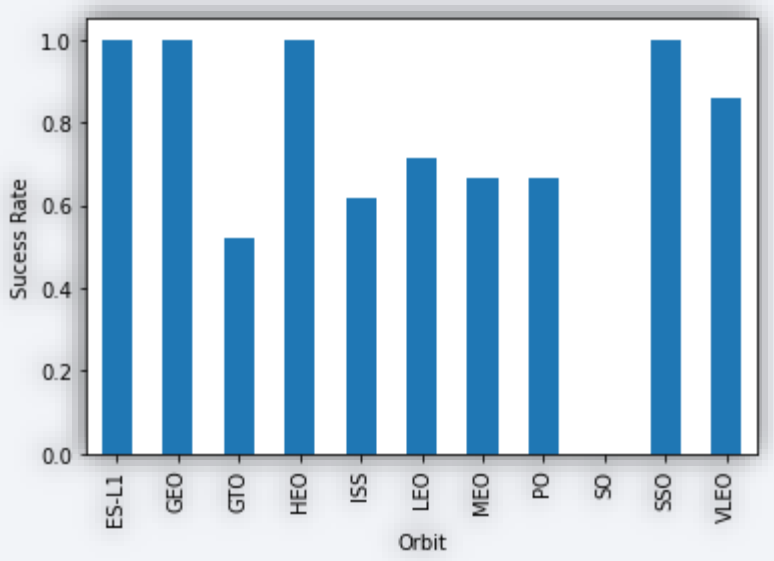
Payload vs. Launch Site

This plot shows that once mass exceeds 7000 kilograms, the probability of successful launch increases. However, there isn't a clear pattern from just this.



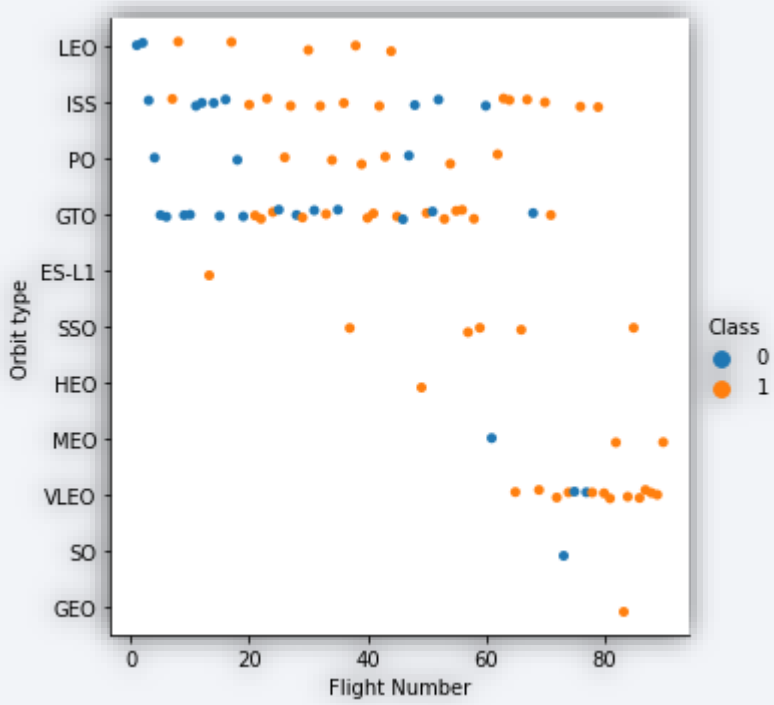
Success Rate vs. Orbit Type

This plot shows the success rates of landings based on Orbits. ES-L1, GEO, HEO, and SSO have a 100% chance of success, while the SO orbit will certainly fail.



Flight Number vs. Orbit Type

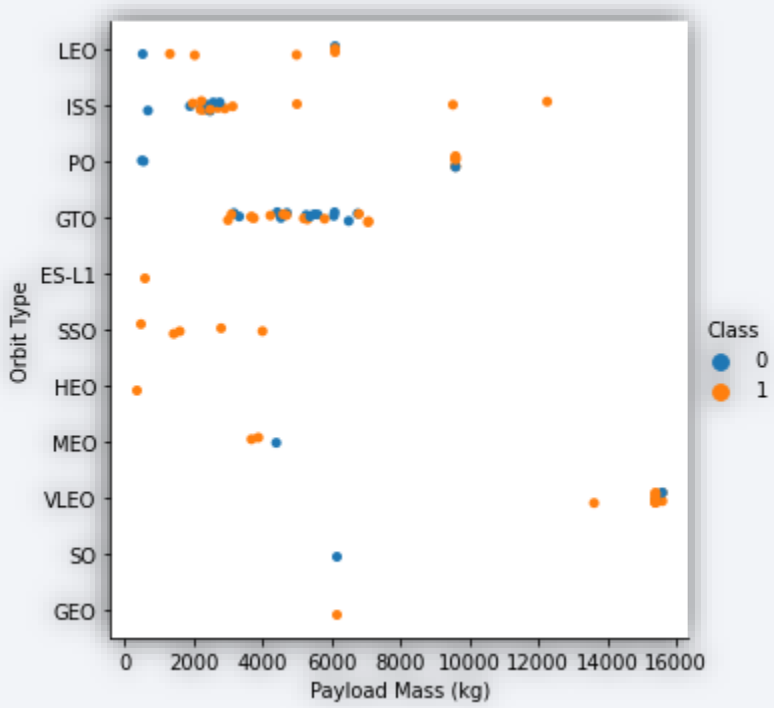
The plot shows that the larger flight number leads to greater success rates in most cases. GTO orbit indicates no relationship, and the orbits with 1 or less occurrences are excluded from this conclusion.



Payload vs. Orbit Type

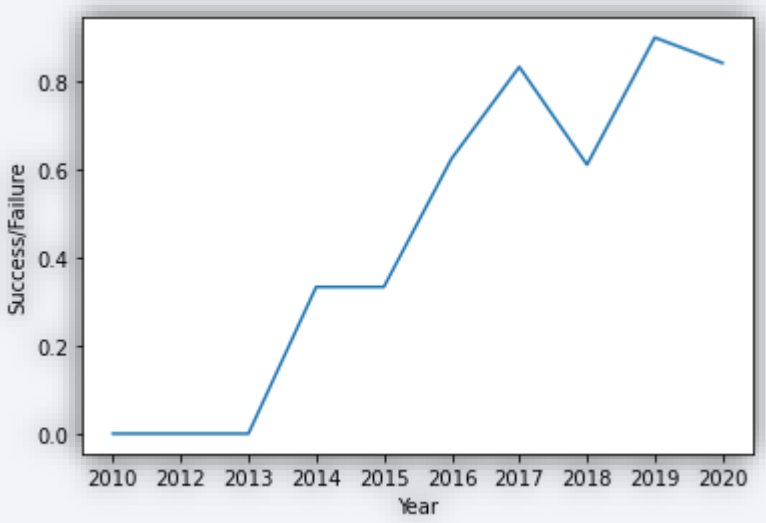
There is no distinguishable relationship between Payload and orbit type in most orbit types overall.

ISS and LEO seems to be positively affected by increasing Payload Mass, while having the opposite effect in orbits MEO or SSO.



Launch Success Yearly Trend

The plot shows a strong positive correlation between year and success rate. We would expect the success rate to level out after some year, which could be what we are observing in 2019-2020.



All Launch Site Names

Using keyword DISTINCT we can show all unique launch site names from the table.

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

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

Launch Site Names Begin with 'CCA'

The percentage symbol “%” allows us to query for Launch Sites that begin with CCA and have trailing characters afterwards.

```
%sql SELECT LAUNCH_SITE from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

launch_site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

Using SUM and SELECTING from we can calculate the total of the Payload Mass in KG for all Payloads carried by NASA CRS.

```
%sql select sum(PAYLOAD_MASS__KG_) as "payloadmass" from SPACEXTBL where (CUSTOMER) = 'NASA (CRS)';
```

```
 * ibm_db_sa://ydg19394:***@19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30699/bludb
Done.
```

payloadmass
45596

Average Payload Mass by F9 v1.1

Using WHERE and AVG we find that the average payload mass carried by F9 v1.1 is 2928.

```
%sql select avg(PAYLOAD_MASS__KG_) as "payloadmass" from SPACEXTBL where (BOOSTER_VERSION) = 'F9 v1.1';
```

```
 * ibm_db_sa://ydg19394:***@19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30699/bludb
Done.
```

payloadmass
2928

First Successful Ground Landing Date

We can use the min() function to find the earliest date.

The first success was in 2015.

```
%sql select min(DATE) from SPACEXTBL where (LANDING__OUTCOME) = 'Success (ground pad)' ;
```

```
 * ibm_db_sa://ydg19394:***@19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30699/bludb
Done.
```

7]:

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Using BETWEEN we can see this result:

```
%sql select BOOSTER_VERSION from SPACEXTBL where LANDING__OUTCOME='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000;
```

```
 * ibm_db_sa://ydg19394:***@19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30699/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

Using count() and GROUP BY we can see there are 1 failure and 99 successful missions.

```
%sql select count(MISSION_OUTCOME) as missionoutcomes from SPACEXTBL GROUP BY MISSION_OUTCOME;
```

```
 * ibm_db_sa://ydg19394:***@19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30699/bludb
Done.
```

]:

missionoutcomes
1
99

Boosters Carried Maximum Payload

Using max() we can select the booster version that carried the largest payload.

```
%sql select BOOSTER_VERSION as boosterversion from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
```

```
 * ibm_db_sa://ydg19394:***@19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30699/bludb
Done.
```

]:

boosterversion
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

2015 Launch Records

Extracting for the year 2015, we can see the launch sites, outcomes, and booster versions for the year of 2015.

```
%sql SELECT MONTH(DATE),MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE FROM SPACEXTBL where EXTRACT(YEAR FROM DATE)='2015';
```

```
 * ibm_db_sa://ydg19394:***@19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30699/bludb
Done.
```

1	mission_outcome	booster_version	launch_site
1	Success	F9 v1.1 B1012	CCAFS LC-40
2	Success	F9 v1.1 B1013	CCAFS LC-40
3	Success	F9 v1.1 B1014	CCAFS LC-40
4	Success	F9 v1.1 B1015	CCAFS LC-40
4	Success	F9 v1.1 B1016	CCAFS LC-40
6	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	CCAFS LC-40

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

Selecting both landing outcome and their count, filtering by dates using where, grouping by outcome and ordering by its count, we can rank the different landing outcomes. Using the keyword DESC, we can sort this into descending order. From this query, we can see that the most common outcome between 2010 and 2017 was No Attempt, followed by drone ship failure.

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

```
 * ibm_db_sa://ydg19394:***@19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30699/bludb
Done.
```

3]:

landing__outcome	2
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 3

Launch Sites Proximities Analysis

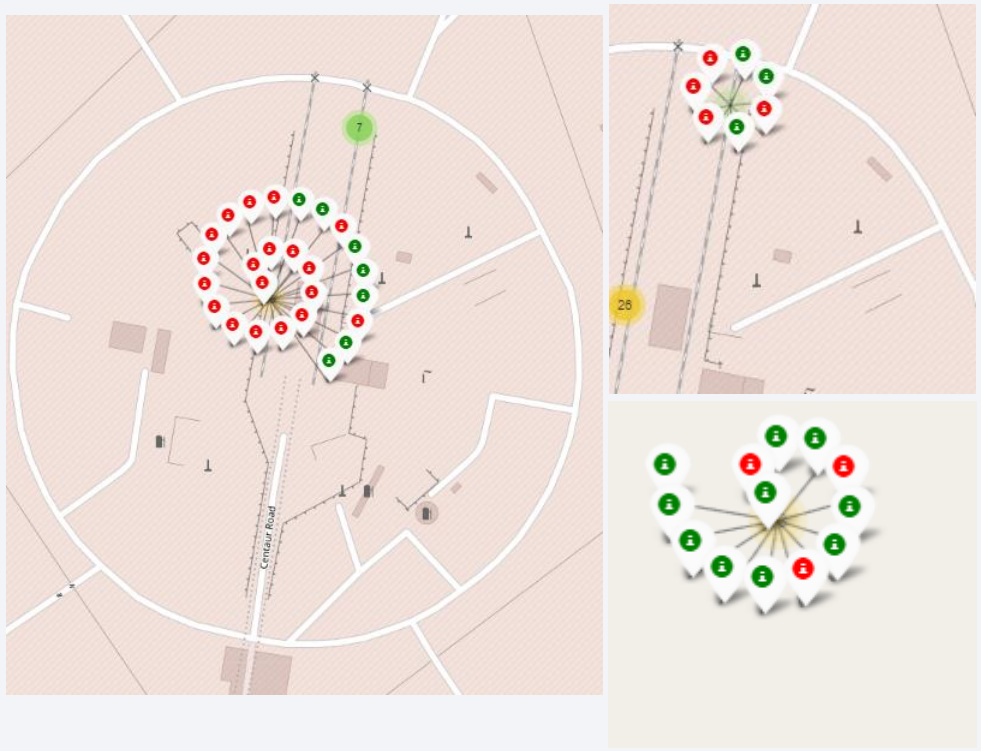
Location of Launch Sites

All SpaceX launch sites are located within the US.

Cluster Markers of successful or failed launches

California Launch Site:

Florida Sites:



Green Markers indicate successful launches. Red indicates failures.

Site Proximity to Landmarks

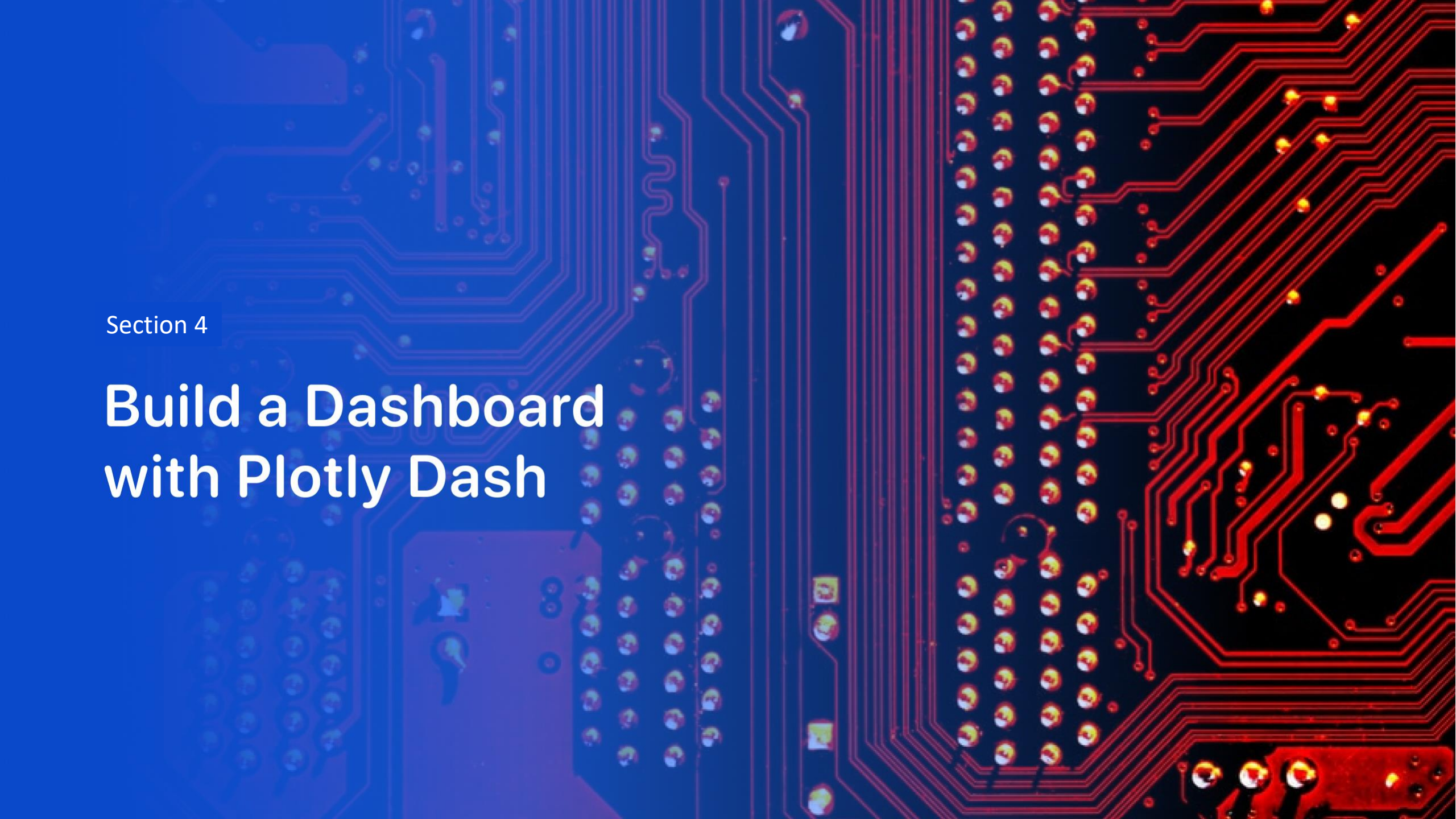
Findings:

Launch sites are close to the coastline because the option for water landings.

Launch sites are close to highways to transport large pieces of equipment.

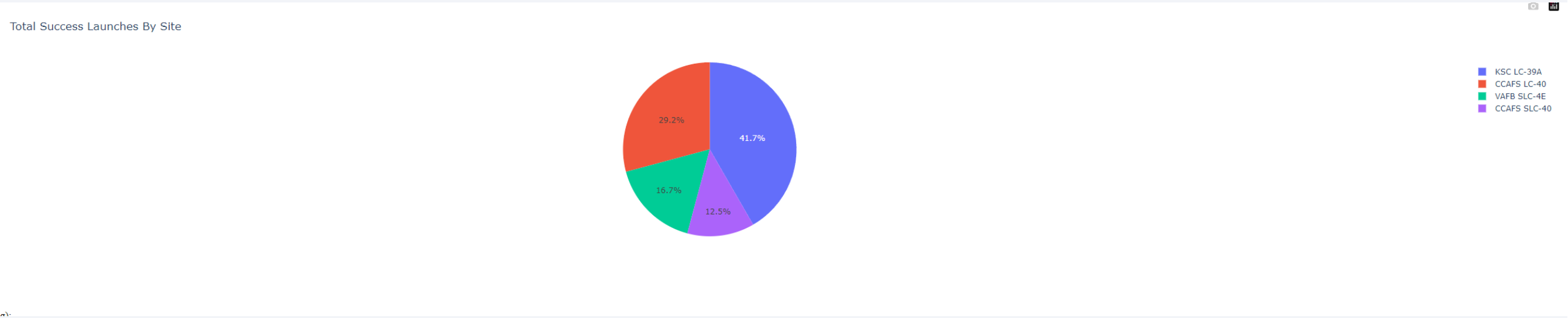
Launch sites are close to railways for the same reason.

Launch sites are far from major cities to minimize risks to the concentrated population.


Section 4
Build a Dashboard with Plotly Dash

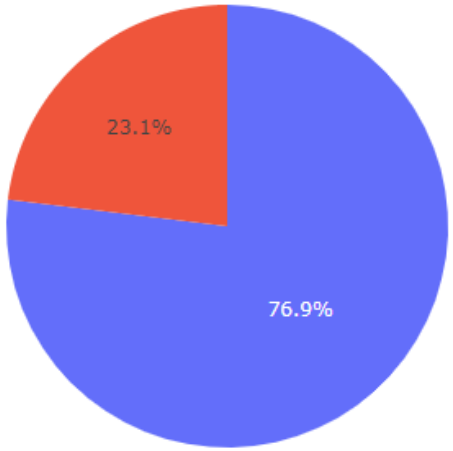
Success Rates of All Sites

KSC LC-39A is the most successful launch site.



Further Success Analysis of KSC LC-39A

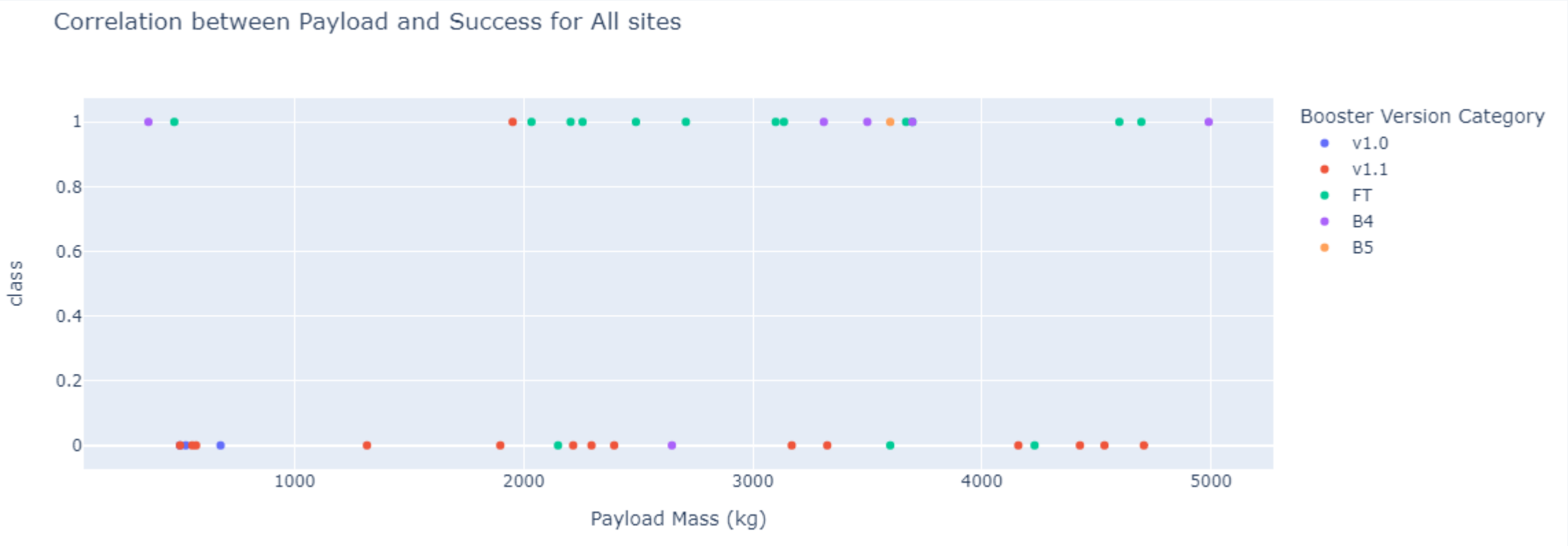
76.9% of launches from KSC LC-39A are successful.



Low Payload vs Launch Outcome

We can see that lower payloads result in a greater chance of success.

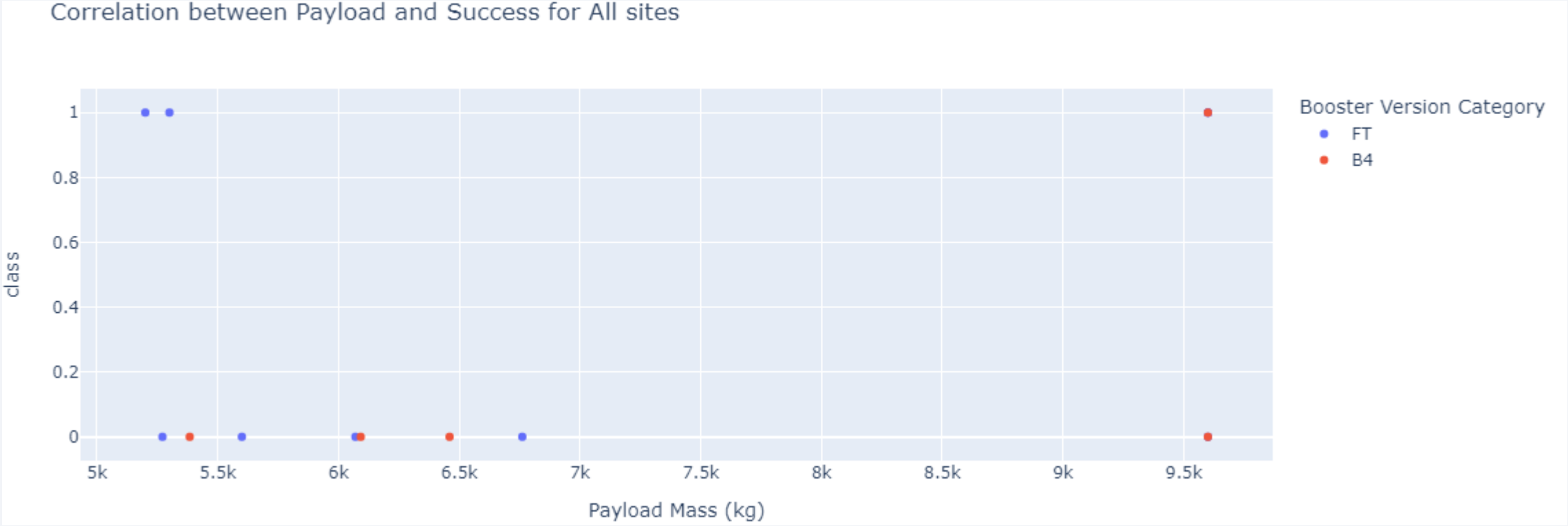
Low payload: (0 kg-5000 kg)



High Payload vs Launch Outcome

We can see that lower payloads result in a greater chance of success.

High payload: (5000kg-10000kg)



Section 5

Predictive Analysis (Classification)

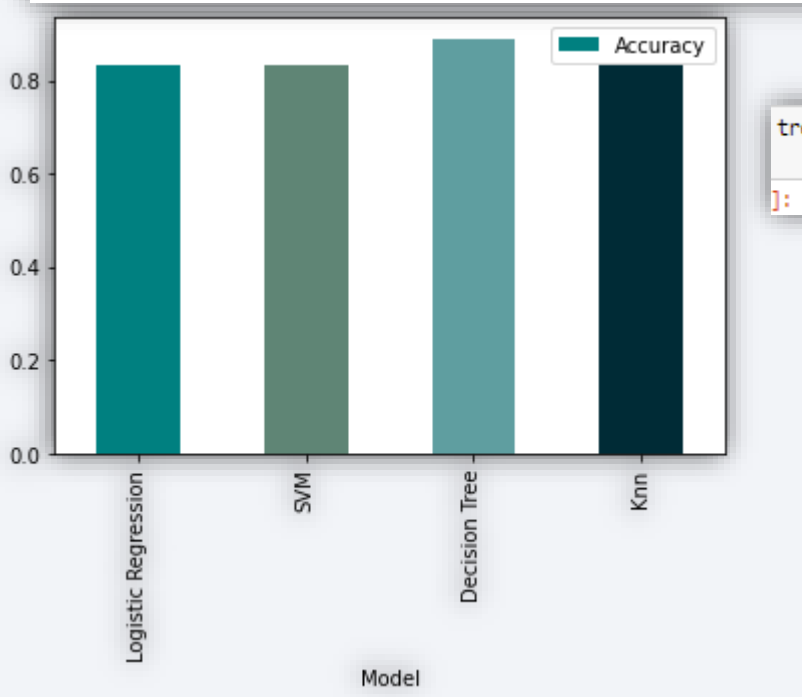
Classification Accuracy

From the bar chart, we can tell that the model with the highest accuracy is the decision tree, with an accuracy of 88%.

Source:
https://github.com/colemoebius2000/SpaceXAnalysis/blob/24117b3854e1fab04c3ba994aae1303a26ada8c5/Machine%20Learning%20Predictions.ipynb

```
lr_confidence = logreg_cv.score(X_test, Y_test)
svm_confidence = svm_cv.score(X_test, Y_test)
dt_confidence = tree_cv.score(X_test, Y_test)
knn_confidence = knn_cv.score(X_test, Y_test)

names = ['Logistic Regression', 'SVM', 'Decision Tree', 'Knn']
columns = ['Model', 'Accuracy']
scores = [lr_confidence, svm_confidence, dt_confidence, knn_confidence]
alg_vs_score = pd.DataFrame([[x, y] for x, y in zip(names, scores)], columns=columns)
alg_vs_score.plot(kind='bar', x='Model', y='Accuracy', color=['#008080', '#5F8575', '#5F9EA0', '#002b36'])
```

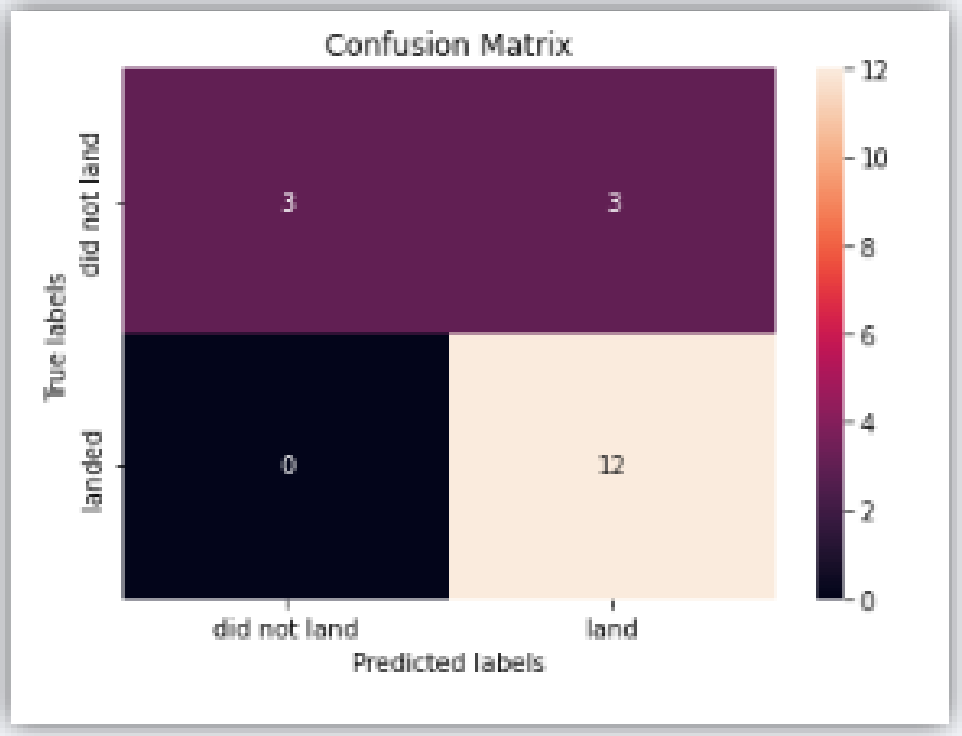



```
tree_cv.score(X_test, Y_test)

]: 0.8888888888888888
```

Confusion Matrix

The confusion matrix for the decision tree classifier gives us two insights. First, that the classifier can distinguish between the classes. Secondly, that the classifier has the most trouble with false positives (unsuccessful landings marked as successful).



Source:
https://github.com/colemoebius2000/SpaceXAnalysis/blob/24117b3854e1fab04c3ba994aae1303a26ada8c5/Machine%20Learning%20Predictions.ipynb

Conclusions

- Lighter payloads have more successful launches.
- As time goes on, SpaceX launches will continue to have greater success.
- KSC LC-39A site succeeds 76.9% of the time.
- SSO Orbit has the greatest launch success.
- The decision tree was most applicable ML algorithm, with a confidence score of .8888.

Thank you!