IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Steven London Morris
April 22, 2024

Outline

EXECUTIVE SUMMARY

INTRODUCTION

METHODOLOGY

RESULTS

CONCLUSION

APPENDIX

Executive Summary

- Data was collected through both HTTP Web scraping and calling the SpaceX API. Many of the launch data is made available through both Wikipedia and the SpaceX API.
- Exploratory Data Analysis (EDA) was performed via SQL and Data Visualization primarily using Seaborn library.
- We further visualized some of the data using both a Ploty Dashboard on launch data, and also using Folium to map out launch sites
- Using Predictive Analysis we see that the Decision Tree classifier from Scikit-Learn is most accurate
- We also learn some other conclusions such as:
    - Most successful Launch Sites
    - Most successful Orbits
    - Effect of Payloads
    - What booster versions handle higher payloads
    - What year(s) did launches start to grow successful

Introduction

SpaceX is at the forefront of the Space Age. They perform a variety of rocket ship launches at different launch sites. While their launches are far more inexpensive than other competitors, they still cost millions of dollars.

Our main objective is to look at SpaceX launch data to determine:

- What factors contribute most to a successful launch
- The most successful launch sites via data visualization
- Predict if the first stage of a launch will be successful using predictive analytics

Doing so can assist in saving costs

Section 1

Methodology

Methodology

- Executive Summary
- Data collection methodology:
  - Request and parse SpaceX launch data through a GET request
  - Webscrape HTML data from SpaceX Launch Data from Wikipedia
- Perform data wrangling
  - Convert NULL values in Payload to zero
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Logisitc Regression, SVM, Decision Tree, KNN

Data Collection

- Data collection used a variety of methods:
  - We started by using a GET request of the SpaceX API
  - We then used .json_normalize() to turn the data into a Pandas dataframe
  - We cleaned the data including:
    - Standardizing date structures
    - Removed rows with multiple cores
    - Filtered the data to only work with Falcon9 launches
    - Filtered out missing values for Payload Mass

Data Collection – SpaceX API

- We used a GET request to retrieve the data from the static_json_url
- We then normalized the data
- Using the normalized data, we then used the API again to get information about the launches using the IDs given for each launch

- Source

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and
  # rows that have multiple payloads in a single rocket.
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 extract the single value in the list and replace the feature.
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 and then extracting the date leaving the time
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)]
```

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datas
```

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_norm`

```
# Use json_normalize method to convert the json result into a dataframe
response = requests.get(static_json_url).json()   #decode using .json
data = pd.json_normalize(response) #convert to pandas dataframe
```

Data Collection - Scraping

- We performed an HTTP Get method to request Falcon9 launch data
- We then created a BeautifulSoup object
- Then we extracted column names from the HTML table headers
- Then we created a dataframe by parsing launch HTML tables
- **Source**

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&old
```

Next, request the HTML page from the above URL and get a `response` object

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

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

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data)
```

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

rows = first_launch_table.find_all('th')

for row in rows:
    column_name = extract_column_from_header(row)
    if column_name is not None and len(row) > 0:
        column_names.append(column_name)
-----
```

Data Wrangling

- First we calculated the number of launches of each site
- Then we calculated the number and occurrence of each orbit
- Afterwards we calculated number and occurrence of mission outcome of the orbits
- Using this information, we created a landing outcome label of the Outcome column
- Source

```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

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

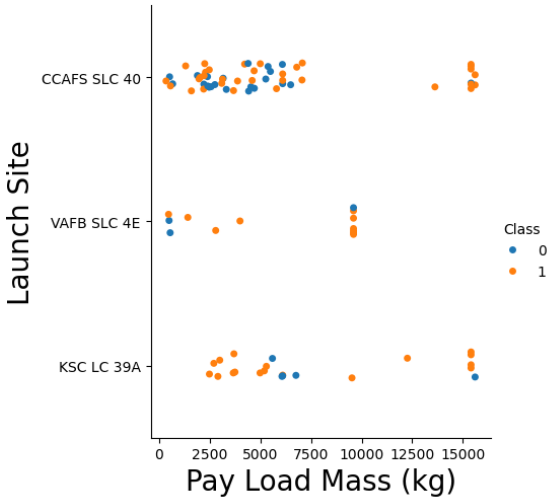
```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

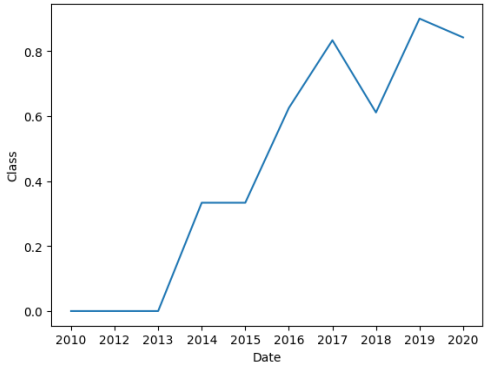
```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

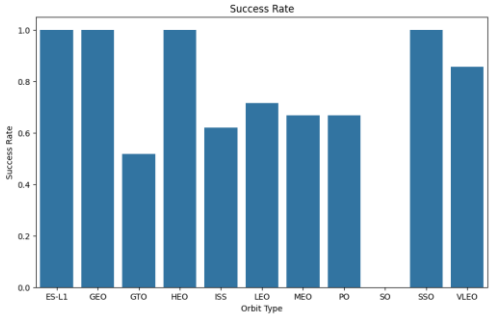
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise

landing_class = [] #create list landing_class
for landing_outcome in df['Outcome']:
    if landing_outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

EDA with Data Visualization

- For data visualization, we used 3 main types of plots:
  - Scatterplots
  - Bar plots
  - Line charts
- Amongst the 3, scatterplots were used the most. The plot is useful in comparing the relationship between 2 variables to see if they cause success
- **Source**

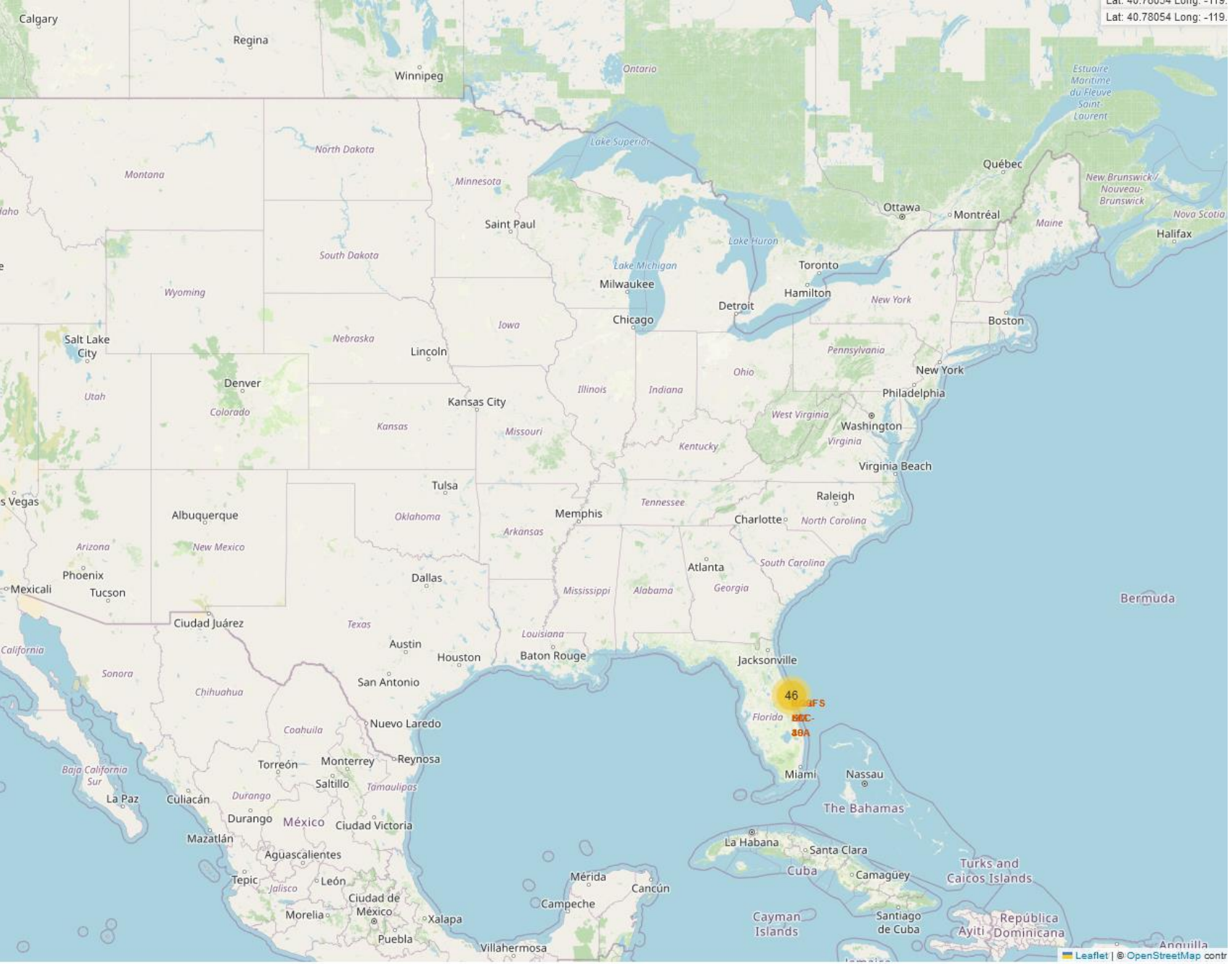






EDA with SQL

- We performed a variety of queries on the SPACEXTBL, such as:
  - Displaying all the distinct launch sites
  - Displaying 5 records beginning with ‘CCA’
  - Displaying the total payload mass carried by boosters launched by NASA
  - Displaying average payload from booster version F9 v1.1
  - Finding date of first successful landing outcome
  - Naming the boosters with success in drop ship and having a mass between 4k & 6k kg
  - Listing the ottal number of successful and failed mission outcomes
  - Naming booster versions carrying maximum payload mass
  - Listing failed launches from 2015
  - Ranking the count of landing outcomes between June 2010 & March 2017
- **Source**

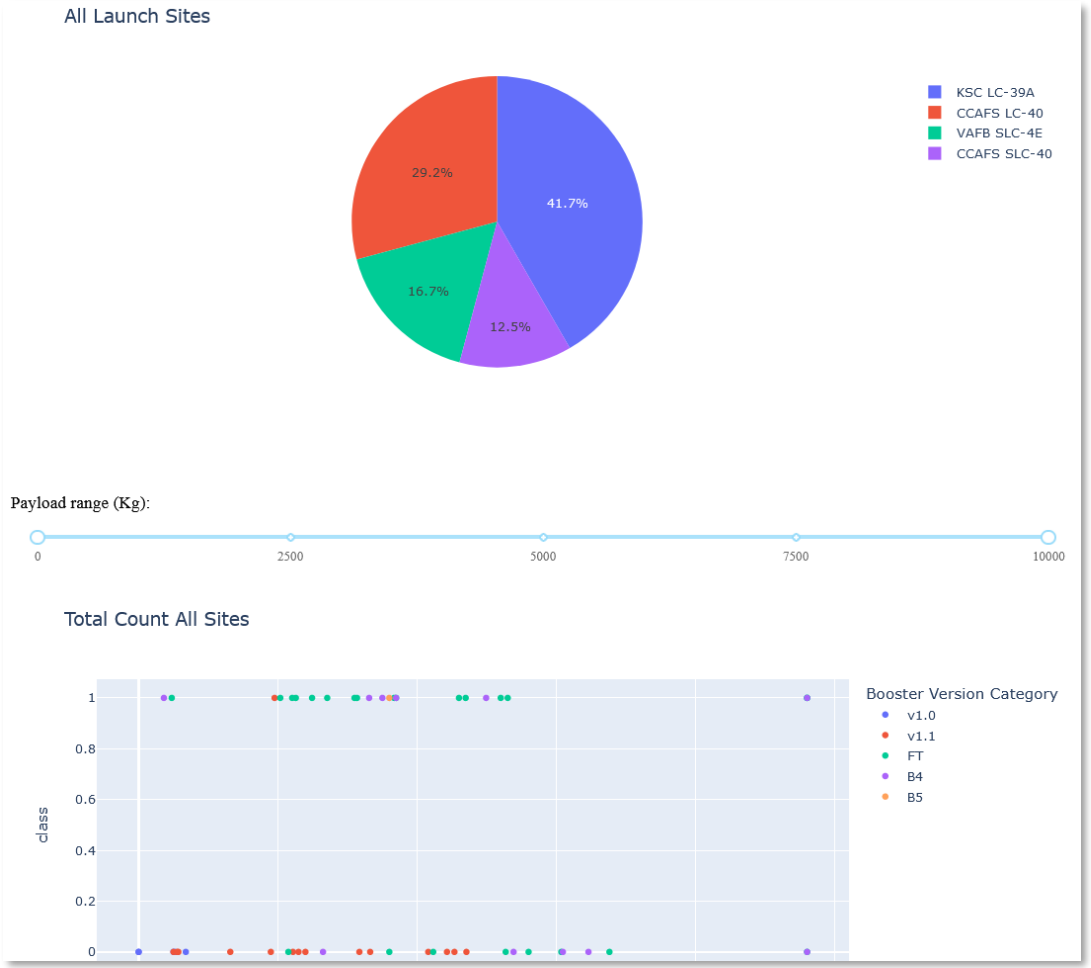


Build an Interactive Map with Folium

- Using Folium we created identified all of the SpaceX launch sites in America.
- Afterwards we created marker clusters at each launch site, to show both successful and failed launches
- Using the VAFB SLC-4E launch site location we then analyzed some of the surrounding area including:
  - Nearby coastlines
  - Nearby cities
  - Nearby railways
- We also calculated the distance of the above in (KM) to see how far they are from the launch site
- **SOURCE (NB VIEWER)**

Build a Dashboard with Plotly Dash

- We created a dashboard using Ploty
- The dashboard includes:
  - An a drop-down menu that handles information for all launch sites & individual sites
  - A pie chart that shows successful launch percentages
  - A slider that filters through payload values
  - A scatterplot that shows the correlation between payload and the various classes
  - The values on the scatterplot will vary depending on the payload slider
- **SOURCE**

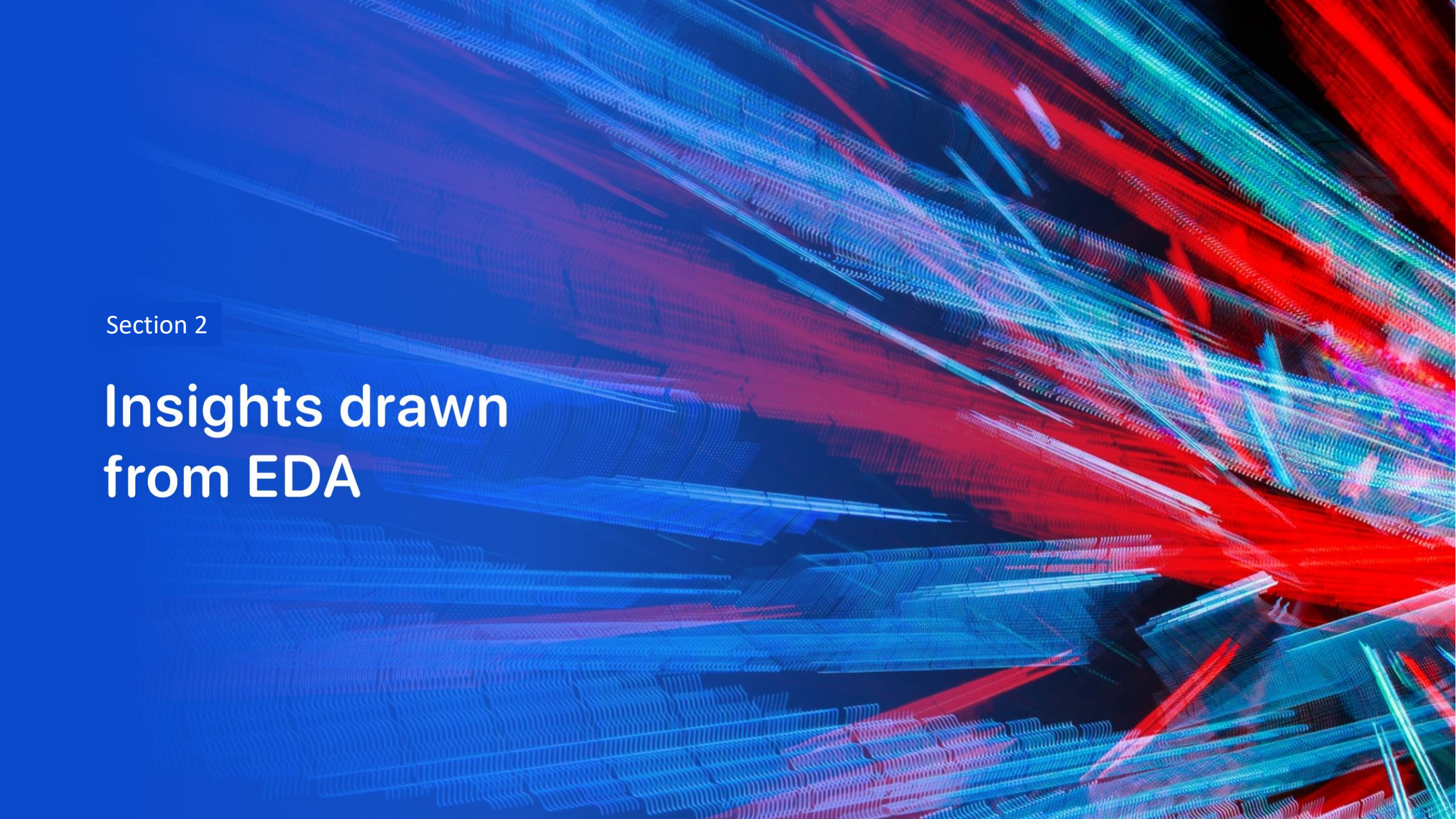


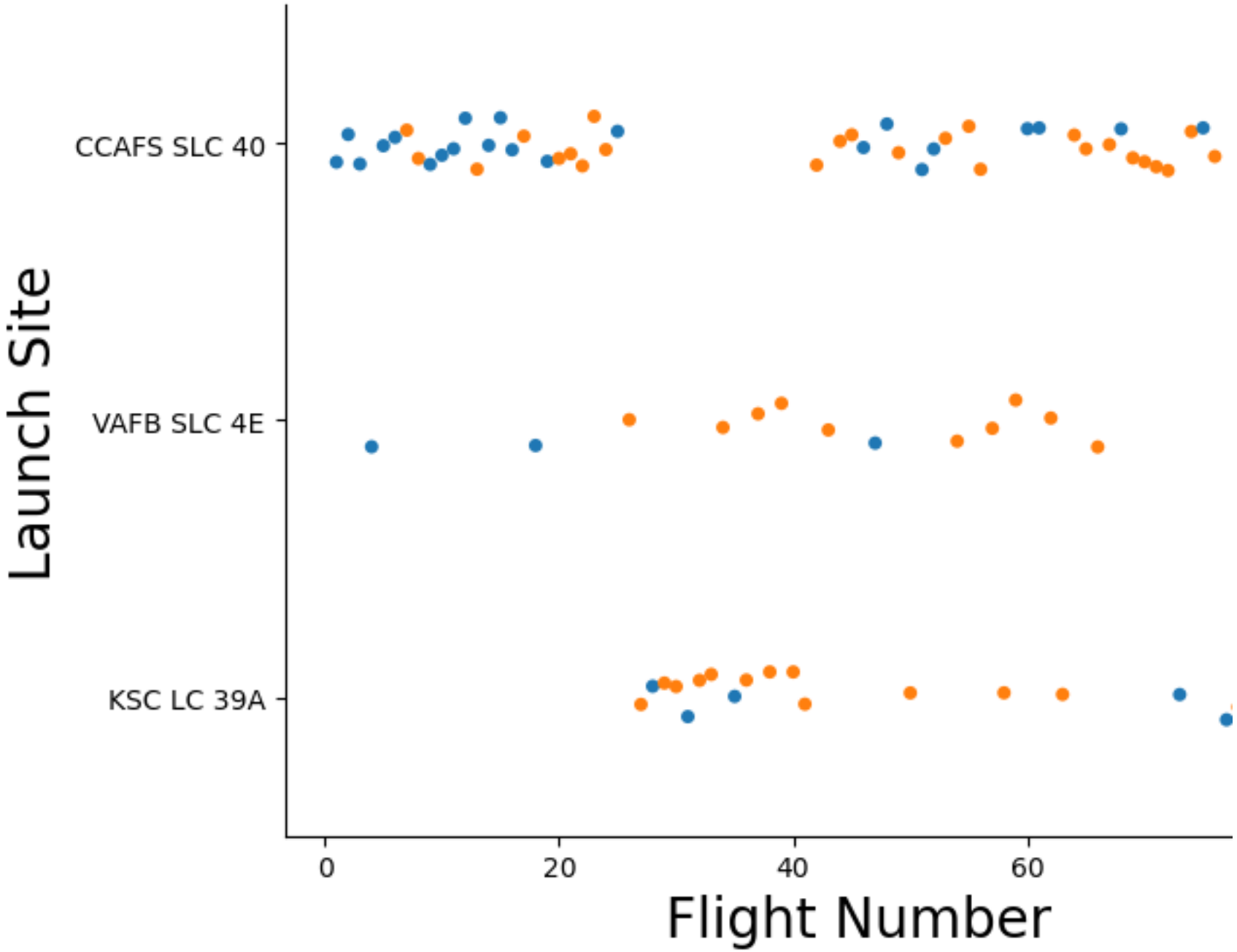
Predictive Analysis (Classification)

- We used Numpy and Pandas to transform the data.
- We then split the data between training data and testing data.
- Afterwards we used GridSearchCV on a variety of machine learning models such as:
  - Decision Trees
  - KNN
  - SVM
  - Logisitc Regression
- We determined that Decision Trees had the best score accuracy compared to other models
- SOURCE

Results

- Some of our results included in the following slides will contain:
  - Exploratory data analysis results via SQL and Data Visualization methods
  - Interactive analytics demo in screenshots
  - Predictive analysis results


Section 2
Insights drawn from EDA

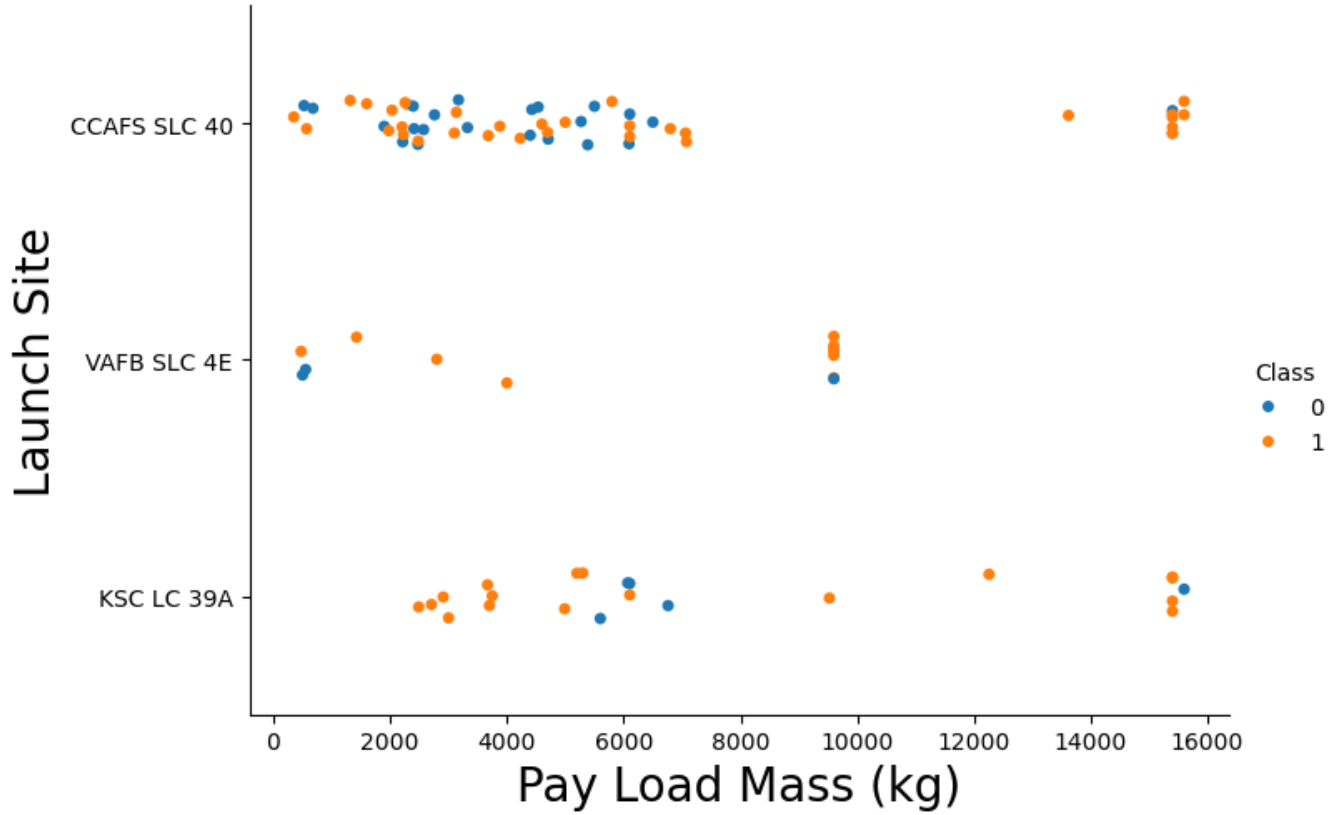


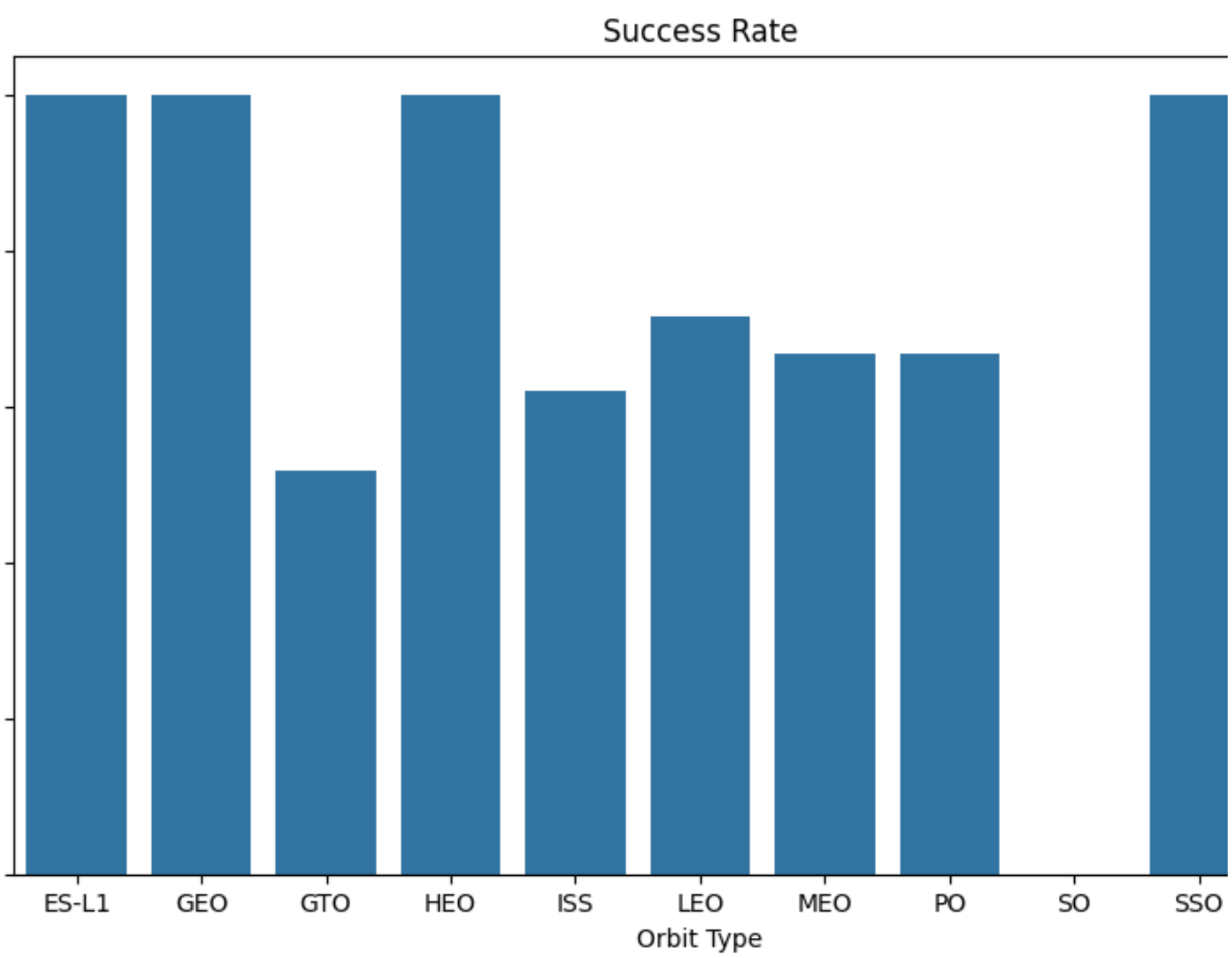
Flight Number vs. Launch Site

- CCAFS SLC 40 has the largest number of flights and the most amount of failed launches compared to VAFB SLC 4E and KSC LC 39A

Payload vs. Launch Site

VAFB SLC-4E is the only launch site to avoid payloads greater than 10,000 kg



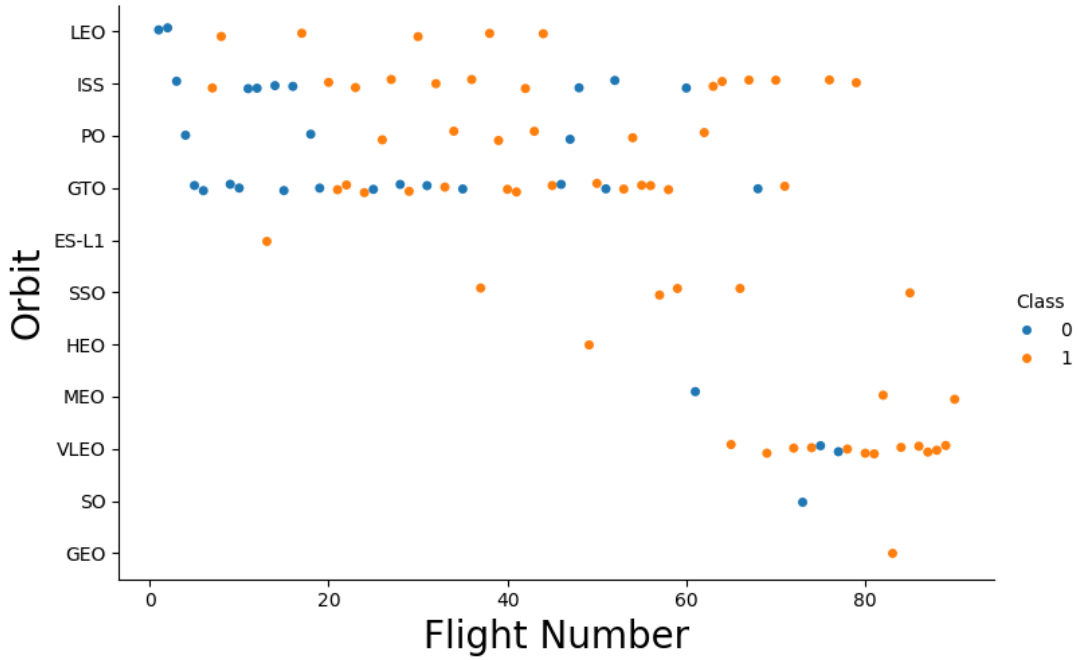


Success Rate vs. Orbit Type

- ES-l1, GEO, HEO, and SSO have the greatest success rates of launches compared to the other Orbit Types

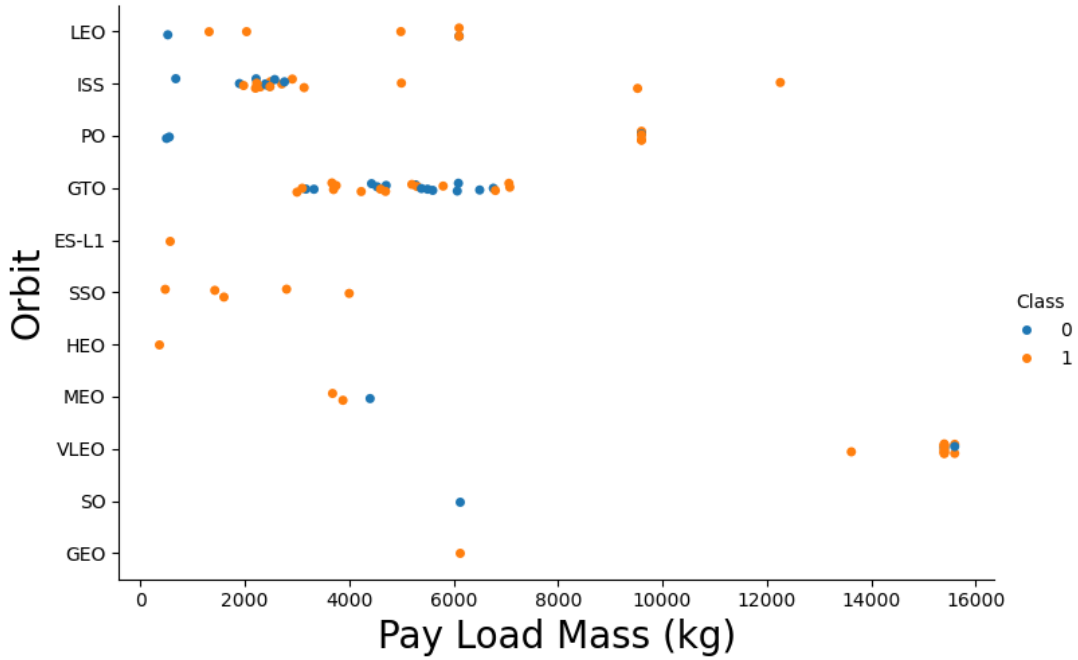
Flight Number vs. Orbit Type

- LEO seems to have greater success on higher flight numbers, but that trend isn't the same for all orbits



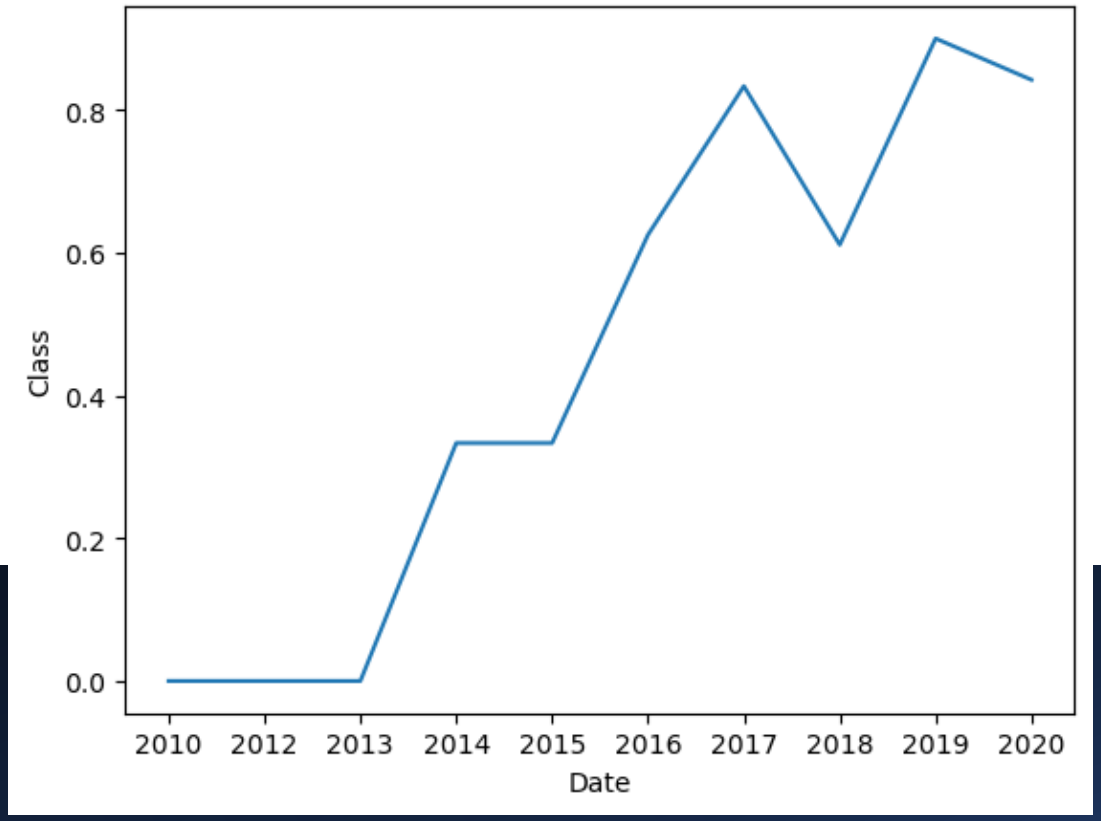
Payload vs. Orbit Type

- Higher payloads seem to have higher success rates, but they’re less frequent.



Launch Success Yearly Trend

The launches have gotten more successful from 2013 to 2020



All Launch Site Names

- The unique launch sites are:
  - CCAFS LC-40
  - VAFC SLC-4E
  - KSC LC-39A
  - CCAFS SLC-40

Task 1

Display the names of the unique launch sites in the space mission ¶

```
%sql SELECT DISTINCT(Launch_Site) FROM SPACEXTBL
```

```
 * sqlite:///my_data1.db
Done.
```

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

Launch Site Names Begin with 'CCA'

Task 2

Display 5 records where launch sites begin with the string 'CCA'

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

```
 * sqlite:///my_data1.db
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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

Included is a query displaying some of the launches from launch sites beginning with CCAFS

Total Payload Mass

Task 3

Display the total payload mass carried by boosters launched by NASA (CRS)

```
%sql SELECT Customer, SUM(PAYLOAD_MASS__KG_) as  Sum_Payload_Mass_KG FROM SPACEXTBL WHERE Customer = 'NASA (CRS)' GROUP BY Customer
```

```
 * sqlite:///my_data1.db
Done.
```

Customer	Sum_Payload_Mass_KG
NASA (CRS)	45596

The total payload carried by boosters launched by NASA (CRS) is 45,596 KG

Average Payload Mass by F9 v1.1

Task 4

Display average payload mass carried by booster version F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as Avg_Payload_Mass_KG FROM SPACEXTBL WHERE Booster_Version = 'F9 v1.1'
```

```
 * sqlite:///my_data1.db
Done.
```

Avg_Payload_Mass_KG
2928.4

The average of payload mass carried by booster version F9 v1.1 is 2928.4 KG

First Successful Ground Landing Date

The date of the first successful landing outcome is June 4, 2010

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
%sql SELECT MIN(DATE) FROM SPACEXTBL WHERE Mission_Outcome = 'Success'
```

```
 * sqlite:///my_data1.db
Done.
```

MIN(DATE)
2010-06-04

Successful Drone Ship Landing with Payload between 4000 and 6000

- Included in the picture are a list the names of boosters that have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

Task 6 ¶

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
%sql SELECT Booster_Version FROM SPACEXTBL WHERE Mission_Outcome = 'Success' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

```
 * sqlite:///my_data1.db
Done.
```

Booster_Version
F9 v1.1
F9 v1.1 B1011
F9 v1.1 B1014
F9 v1.1 B1016
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1030
F9 FT B1021.2
F9 FT B1032.1
F9 B4 B1040.1
F9 FT B1031.2
F9 FT B1032.2
F9 B4 B1040.2
F9 B5 B1046.2
F9 B5 B1047.2
F9 B5 B1046.3
F9 B5 B1048.3
F9 B5 B1051.2
F9 B5B1060.1
F9 B5 B1058.2
F9 B5B1062.1

Total Number of Successful and Failure Mission Outcomes

We had 100 successful mission comes, and 1 failure mission outcome

Task 7

List the total number of successful and failure mission outcomes

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) as Mission_Outcomes FROM SPACEXTBL GROUP BY Mission_Outcome
```

```
 * sqlite:///my_data1.db
Done.
```

Mission_Outcome	Mission_Outcomes
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- Amongst Booster Versions, many of the versions F9 B5 B between versions F9 B5 B1048.4 and F9 B5 1060.3 can carry the maximum payload mass

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
%sql SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

```
 * sqlite:///my_data1.db
Done.
```

Booster_Version
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

```
%sql SELECT substr(Date, 6,2) as month , Landing_Outcome, Booster_Version, Launch_Site
FROM SPACEXTBL WHERE substr(Date,0,5)='2015' AND Landing_Outcome = 'Failure (drone ship)'
```

```
 * sqlite:///my_data1.db
Done.
```

month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- In 2015, there were 2 failed launches at CCAFS LC-40. They carried Booster Versions F9 v1.1 B1012 and F9 v1.1 B1015

2015 Launch Records

```
%sql SELECT Landing_Outcome, COUNT(Landing_Outcome) as Landing_Outcome_Count FROM SPACEXTBL  WHERE DATE
BETWEEN '2010-06-04' and '2017-03-20'  GROUP BY Landing_Outcome ORDER BY Landing_Outcome_Count DESC
```

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

- Included are the result of a query used to rank 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

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

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

- Included are the result of a query used to rank 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

```
%sql SELECT Landing_Outcome, COUNT(Landing_Outcome) as Landing_Outcome_Count FROM SPACEXTBL  WHERE DATE
BETWEEN '2010-06-04' and '2017-03-20'  GROUP BY Landing_Outcome ORDER BY Landing_Outcome_Count DESC
```

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

Section 3

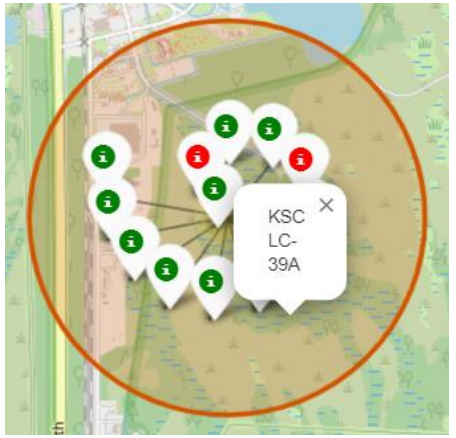
Launch Sites Proximities Analysis

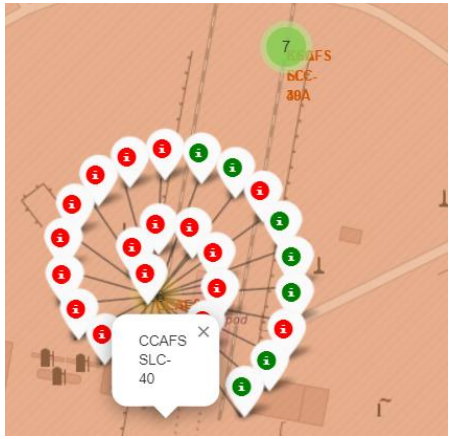
- There are 4 SpaceX launch sites. 3 of them fall on the east coast in Florida, and one site is in California on the west coast. They’re all in America.

Global Launch Site Locations

Color Coded Launch Site Outcomes

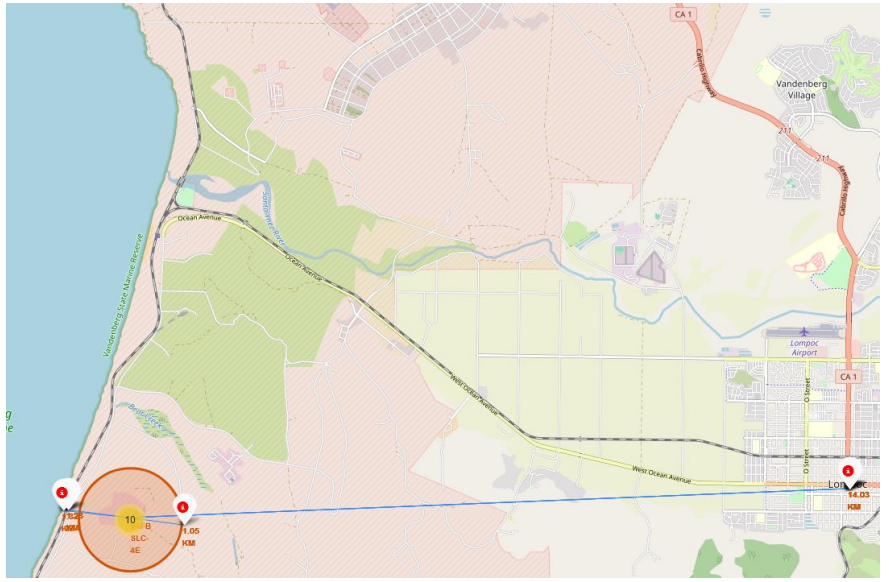
- Amongst the different launch sites, it seems KSC LC-39A has the highest success vs failed outcomes. CCAFS SLC-40 has more launches, but also more failures

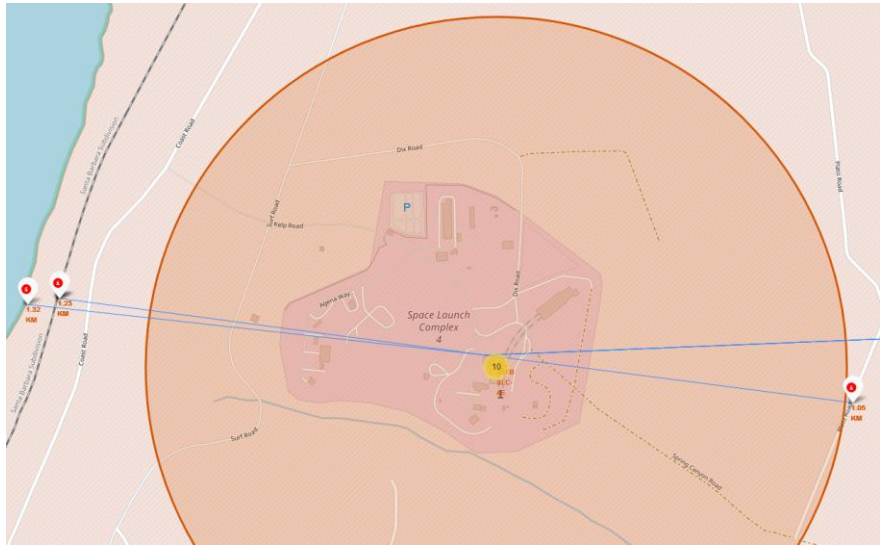


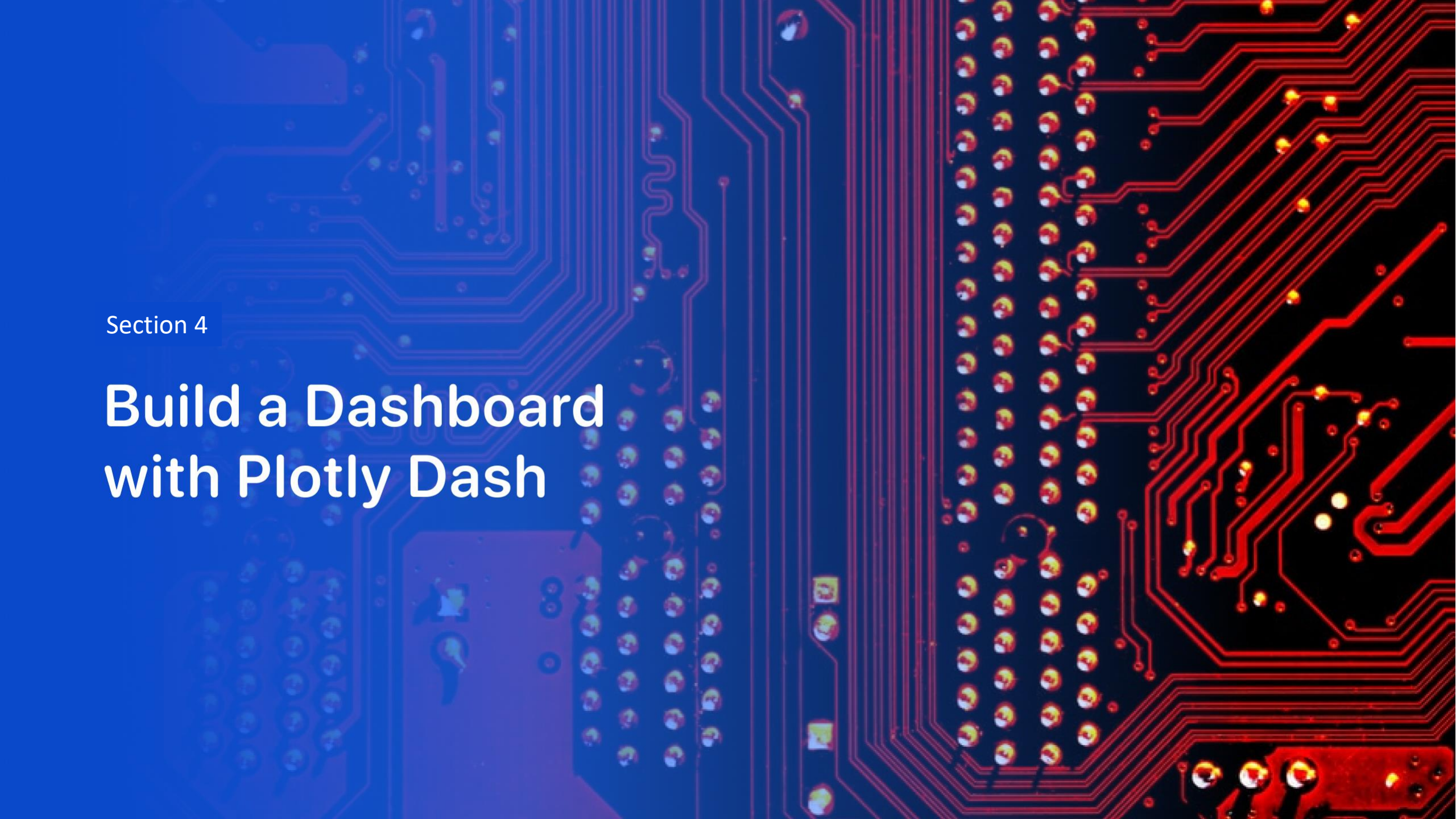


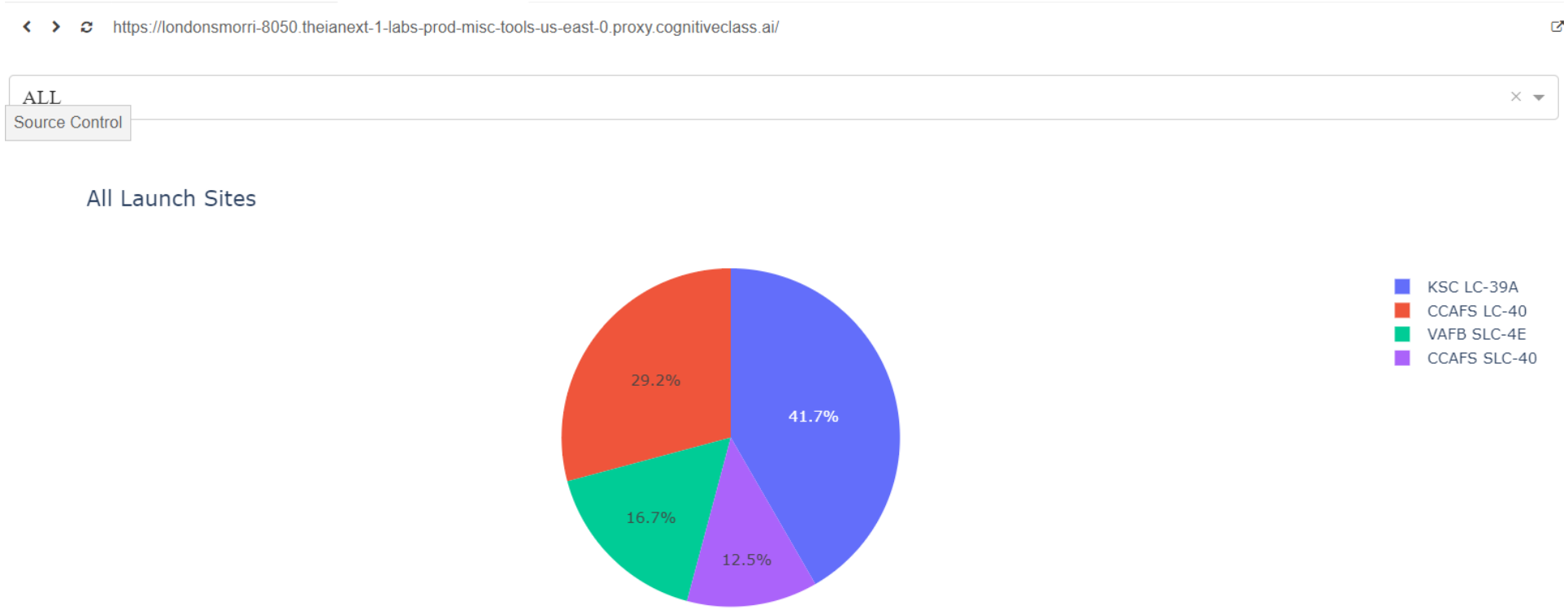
Features in Proximity to VAFB SLC-4E

- VAFB SLC-4E is in close proximity to:
  - Plato Rd (1.05 KM)
  - Santa Barbara Subdivision (1.23 KM)
  - The coastline (1.32 KM)
- It also has a 14.03 KM distance from the city of Lompoc



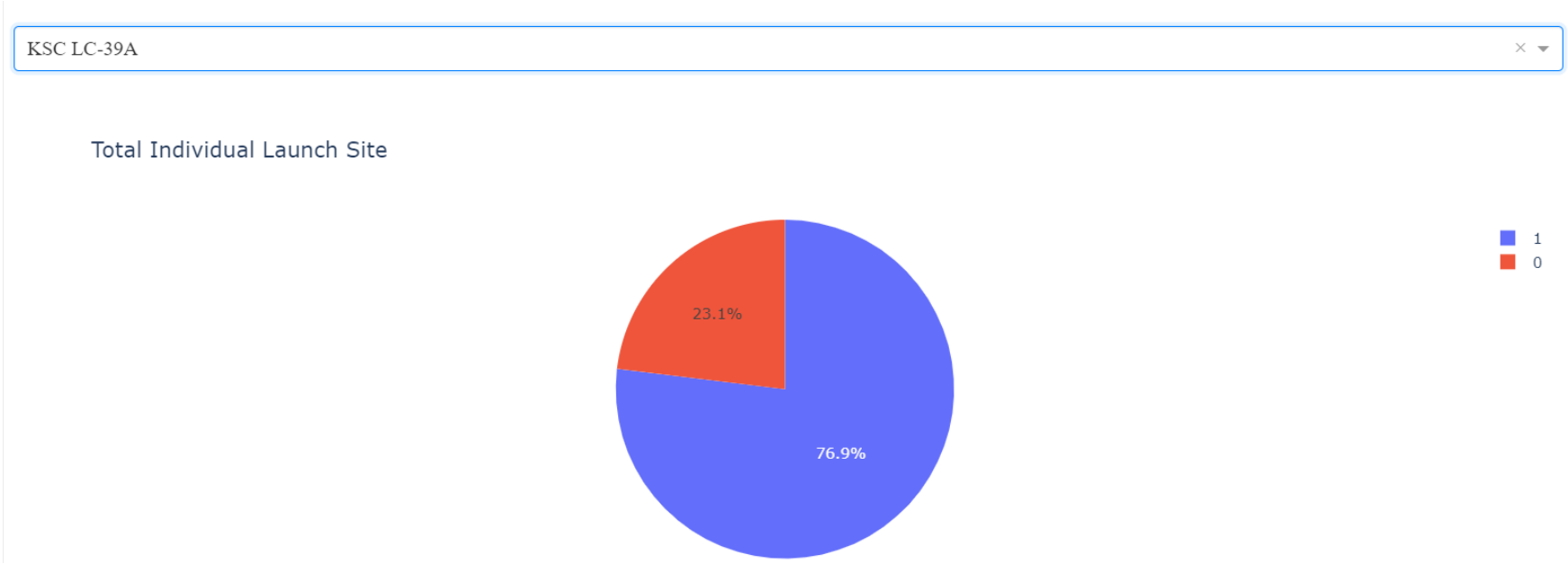



Section 4
Build a Dashboard with Plotly Dash



- Based on the pie chart data, KSC LC-39A has the most successful launches (41.7%)

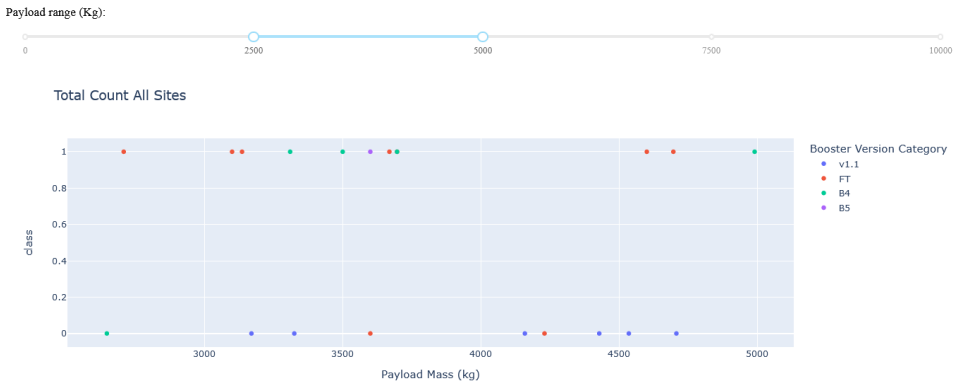
Success Rates of All Launch Sites

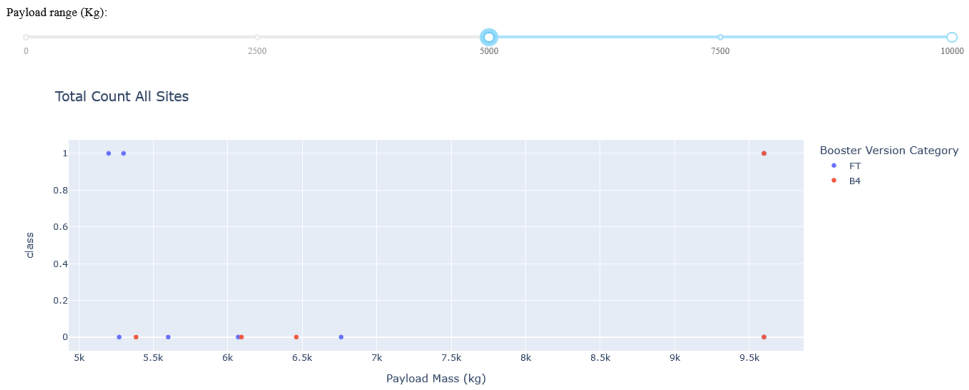


- KSC LC-39A has a success rate of 76.9%, making it more successful than the other sites with lower rates.

KSC LC-39A Success Rate

Payload Slider and Booster Version





- At higher payloads, there seem to be fewer Booster Versions. At payloads above 5000 KG , we only see Booster Version Categories B4 and FT.

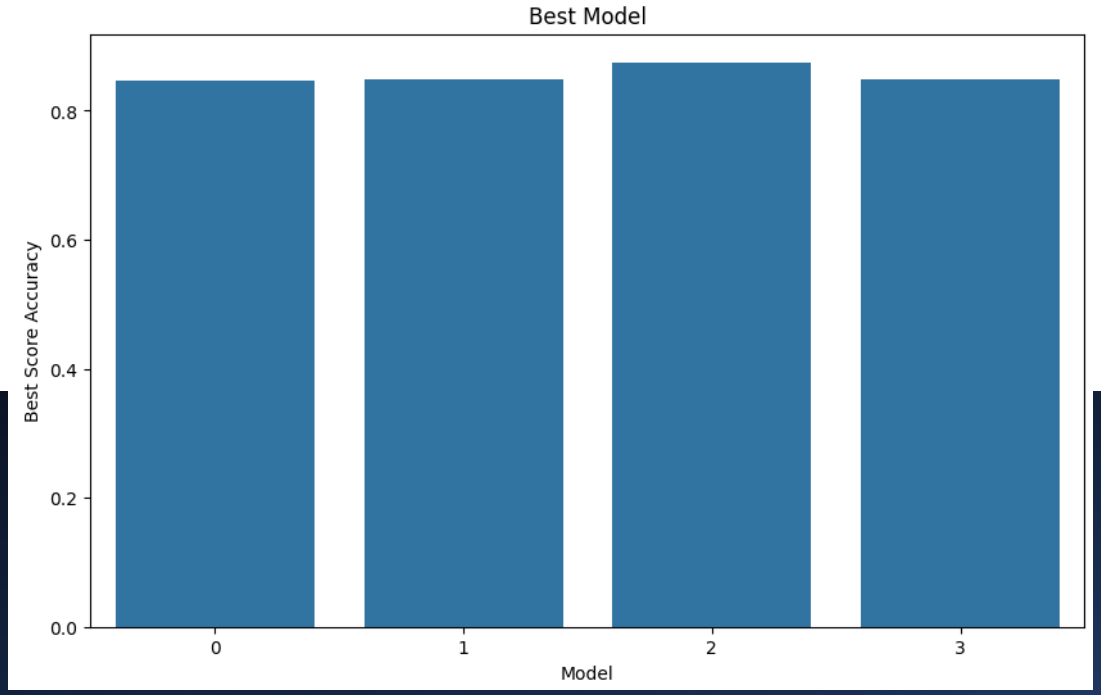
Section 5

Predictive Analysis (Classification)

Classification Accuracy

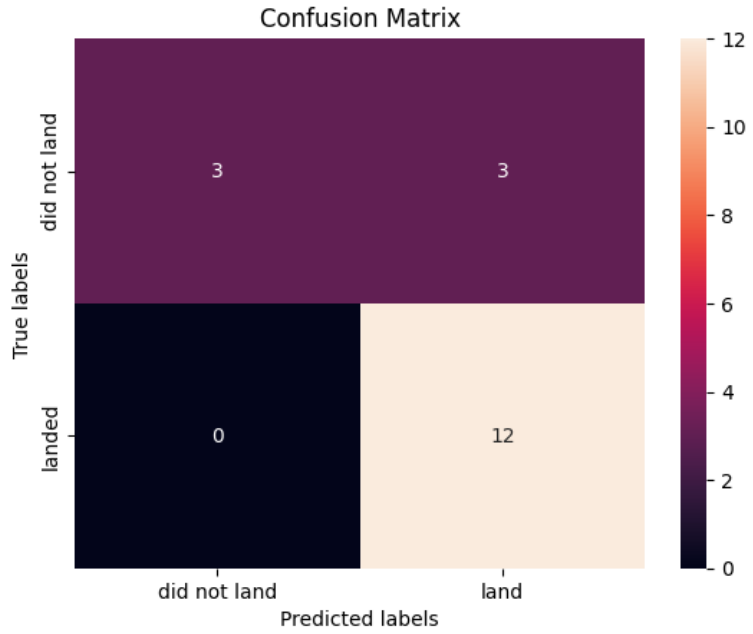
Amongst the various classifications, Decision Trees (Bar #3) has the highest Best Score Accuracy

```
best_score_data = [logreg_cv.best_score_, svm_cv.best_score_, tree_cv.best_score_, knn_cv.best_score_]
model_type = ['Logistic Regression', 'SVM', 'Decision Tree', 'KNN']
plt.figure(figsize=(10, 6))
sns.barplot( best_score_data )
plt.xlabel('Model')
plt.ylabel('Best Score Accuracy')
plt.title('Best Model')

plt.show()
```



Confusion Matrix

- The confusion model does show a degree of accuracy in showing landings.
- However, there does seem to be a few cases of false positive landings (did not land on y-axis vs land on the x-axis)



Conclusions

- Launch Site KSC LC-39 has the most successful launches
- The most success Orbits are ES-l1, GEO, HEO, and SSO
- In terms of machine learning, using a Decision Tree will provide the most accuracy
- Launch sites that can handle higher payloads such as KSC LC-39 also tend to have them be successful
- Booster versions FT and B4 also have experience with handling launches at higher payloads
- Between 2013 and 2020, the success rate of launches sharply compared to a couple years prior

Appendix

- **Github Links:**
  - SpaceX Data Collection API Lab
  - SpaceX Lab Webscraping HTML Lab
  - SpaceX Data Wrangling Lab
  - SpaceX SQL Lab
  - SpaceX Data Visualization (EDA) Lab
  - SpaceX Folium Map Lab NBViewer
    - SpaceX Folium Map Lab GitHub
  - SpaceX Ploty Dashboard Lab
  - SpaceX Machine Learning Predictions Lab

Thank you!