IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Brent Carcamo
09-06-2023

Outline

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

Executive Summary

Methodology

- In this project, I trained four different machine learning models to predict whether the first stage of the Falcon 9 SpaceX rocket will successfully land. Each of the (4) models were optimized using a range of hyperparameters; I then compared the accuracy of each model to find the most useful model.
- The four classification algorithms I tested:
  - Logistic Regression
  - Support Vector Machines
  - Decision Tree Classifier
  - K-Nearest Neighbors

Results

I found that the best model in terms of accuracy was tied between Logistic Regression, SVM, and KNN. The Decision Tree classifier model seemed to be significantly less accurate than the other three models, and therefore the least useful.

In [42]:

```
#Task 12
#Find the method that performs best
print('Model accuracy')
print('Logreg: ', logreg_cv.score(X_test, Y_test))
print('SVM: ', svm_cv.score(X_test, Y_test))
print('Decision Tree: ', tree_cv.score(X_test, Y_test))
print('KNN accuracy: ', knn_cv.score(X_test, Y_test))

#The decision tree model performs best
```

```
Model accuracy
Logreg:  0.8333333333333334
SVM:  0.8333333333333334
Decision Tree:  0.7222222222222222
KNN accuracy:  0.8333333333333334
```

Introduction

Project background and context

SpaceX can launch rockets relatively inexpensively in comparison to others in the industry. This is because SpaceX's Falcon 9 rocket can be reused if it lands without crashing during the first stage. It doesn't always land successfully, but sometimes it does. If we can predict whether a SpaceX rocket with land successfully without crashing, we can better understand SpaceX's estimated costs, and, in turn, how much to bid against SpaceX for government projects. The goal of this project is to develop and train a machine learning model that can determine if the first stage of SpaceX's Falcon 9 rocket will land successfully or not.

Key questions

- What are the key independent variables that determine whether SpaceX will reuse the first stage?
- Which machine learning models work best for predicting whether the first stage will land successfully?
- Do some launch sites have greater success rates than others?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology
- Perform data wrangling
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

SpaceX API Request

- The first way I collected data was through the publicly available SpaceX API. The SpaceX API contains historical launch data with key independent variables needed to train my machine learning model.
- The API was called via the requests library through a GET request. The API returned a JSON file, which I parsed and read into a Pandas DataFrame for further analysis.

Web Scraping Wiki Page

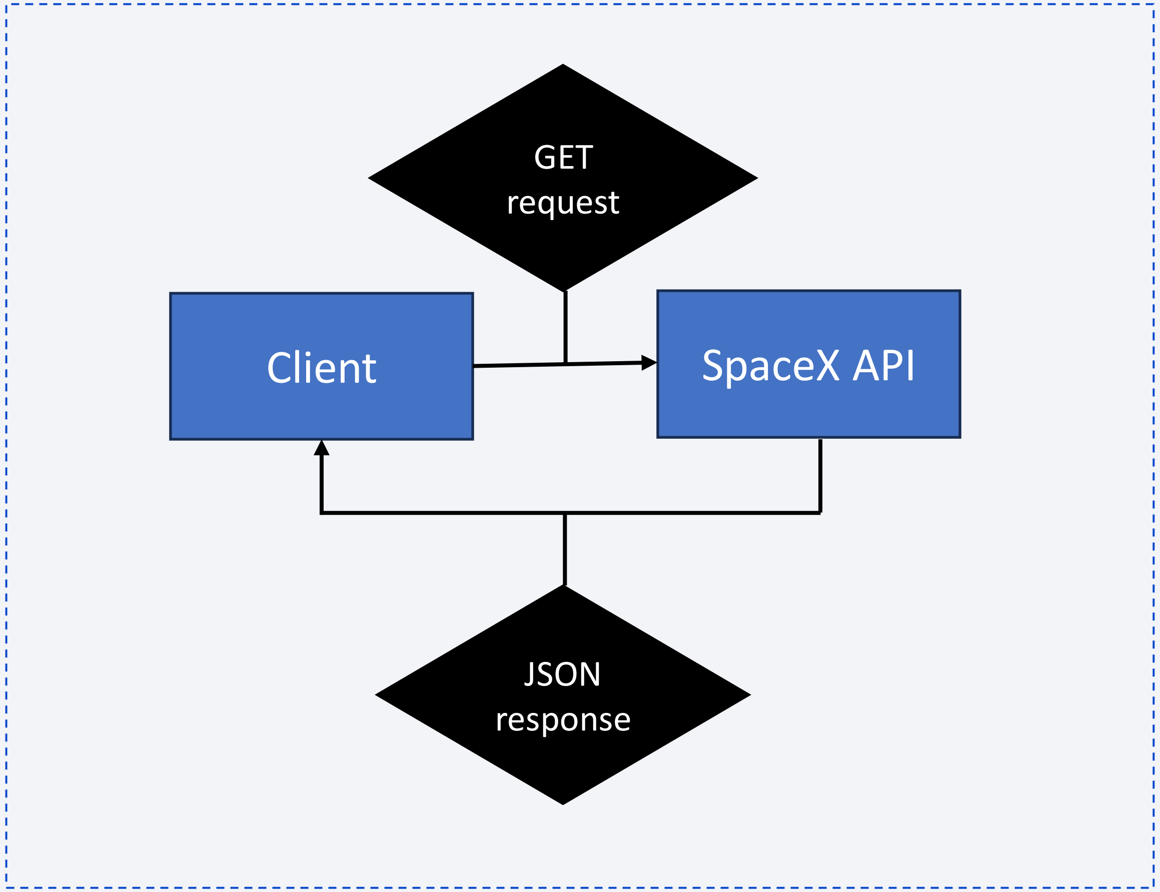
- Data was also collected via web scraping a wiki page using the Beautiful Soup library and the requests library. The requests library was used to scrape rocket launch data from the wiki page, specifically the launch tables. I then used the Beautiful Soup library to parse through the HTML and load the data into my Pandas DataFrame.

Data Collection – SpaceX API

- I used the GET requests to request and parse the SpaceX API, which responds with a JSON file. I then used the pandas json_normalize() method on the JSON response, and read it into a Pandas DataFrame.

- GitHub URL: https://github.com/bcarcamo91/IBM-Capstone/blob/0f5a66cd85d5cd0dbea9dd9e80bf08da5b0d716a/F9%20Data%20Collection%20W1.ipynb

Flowchart



Data Collection – SpaceX API cont.

In [5]:

```
#Request and parse SpaceX launch data using the GET request
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
print(response.content)
```

In [6]:

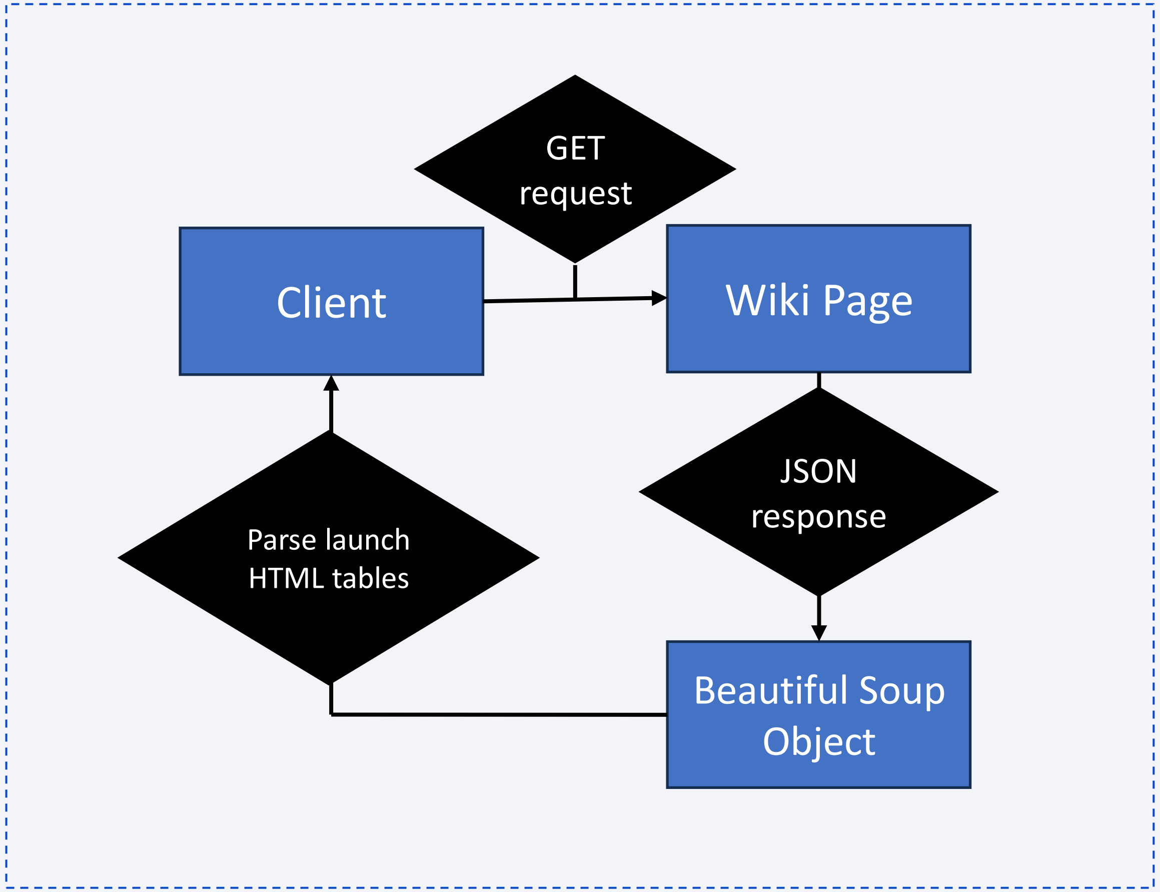
```
#Request and parse the SpaceX launch data using the GET request
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM

#Response code 200 indicates success
response.status_code

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

#Head of dataframe
data.head(5)
```

Data Collection - Scraping

- I used the requests library to get a response from the wiki URL containing the historical rocket launch data, and then used Beautiful Soup to parse it. I created a Beautiful Soup object and used various methods (such as the .find('title')) method to extract the data I needed.

- GitHub URL: https://github.com/bcarcamo91/IBM-Capstone/blob/main/F9%20Web%20Scraping%20W1.ipynb

Flowchart



Data Collection – Scraping cont.

In [4]:
```python
#Scrape data
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

In [5]:
```python
#Use requests.get() method with the provided static_url and assigning response to an object
r = requests.get(static_url)
```

In [6]:
```python
#Create a beautiful soup object
soup = BeautifulSoup(r.text, 'html.parser')
```

In [17]:
```python
#Simplifying the parsing process
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')
```

Data Wrangling

1. Data was first loaded into a Pandas DataFrame
2. Calculated the percentage of missing values in each attribute

```
In [3]:  #Calculate percentage of missing values in each attribute
         df.isnull().sum()/df.shape[0]*100

Out[3]:  Date              0.000000
         BoosterVersion    0.000000
         PayloadMass       0.000000
         Orbit             0.000000
         LaunchSite        0.000000
         Outcome           0.000000
         Flights           0.000000
```

Data Wrangling cont.

3. I then used the value_counts() method to calculate the number of launches on each site.
4. Afterward, I calculated the number and occurrence of each orbit using the same value_counts() method.

5. Next, I calculated the number and occurrence of missing outcome per orbit type, and created a set of outcomes where the second stage did not land successfully.

In [12]:
```
#Calculate the number and occurrence of missing outcome per orbit type
landing_outcomes = df['Outcome'].value_counts()
print(landing_outcomes)
```

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

In [13]:
```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

Data Wrangling cont.

```
In [8]: #Create a set of outcomes where the second stage did not land succesfully
        bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
        bad_outcomes

Out[8]: {'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

6. Finally, I created a landing outcome label from the Outcome column. This is the independent I used for the machine learning models.

GitHub URL:
https://github.com/bcarcamo91/IBM-Capstone/blob/0f5a66cd85d5cd0dbea9dd9e80bf08da5b0d716a/F9%20Data%20Wrangling%20W1.ipynb

```
In [9]: #Create a landing outcome label from Outcome column
        #landing_class = 0 if bad_outcome
        #landing_class = 1 otherwise
        landing_class = []
        for key,value in df["Outcome"].items():
              if value in bad_outcomes:
                 landing_class.append(0)
              else:
                 landing_class.append(1)

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

        #Save file to CSV
        #df.to_csv("dataset_part_2.csv", index=False)
```

EDA with Data Visualization

- I tried to identify correlations and relationships between independent variables and successful landings to get a feel for the day and begin to understand which variables are strong predictor variables.
- Charts included scatter plots of Payload Mass vs. Flight Number, Launch Site vs. Flight Number, Launch Site vs. Payload Mass, Orbit vs. Payload Mass, and others. I plotted a line plot to visualize the launch success yearly trend as well.
- The purpose of the visualization was to get a feel for variable relationships and tendencies. The scatter chart of Launch Site vs. Flight Number showed me which launch sites had a higher percentage of successful launches. This information is useful when building the machine learning model in later stages.
- Github URL: https://github.com/bcarcamo91/IBM-Capstone/blob/0f5a66cd85d5cd0dbea9dd9e80bf08da5b0d716a/F9%20Exploring%20and%20Preparing%20Data%20W2.ipynb

EDA with SQL

Summary of SQL queries performed

- Displayed the names of the unique launch sites in the space mission to get an exhaustive list of all the possible launch sites
- Displayed 5 records where launch sites begin with the string 'CCA'
- Displayed the distinct Booster Versions where launch sites begin with the string 'CCA'
- Displayed the average payload mass carried by booster version F9 v1.1
- Listed the date when the first successful landing outcome in ground pad was achieved
- Listed the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- Listed the total number of successful and failure mission outcomes
- Listed the names of the booster versions which have carried the maximum payload mass using a subquery.
- Listed the count of landing outcomes between dates covering the range of around 7 years.
- Github URL: https://github.com/bcarcamo91/IBM-Capstone/blob/0f5a66cd85d5cd0dbea9dd9e80bf08da5b0d716a/F9%20SQL%20EDA%20W2.ipynb

Build an Interactive Map with Folium

- I created a Folium map with various markers, circles, and lines that could help visualize the geographical locations of SpaceX's launch sites. The Folium map I created included the following:
  - The NASA Johnson Space Center marked with a circle and marker.
  - All launch sites from the SpaceX dataset marked with a circle and marker. Pop up text includes the respective launch site's name.
  - All successful/failed launches marked on the map in marker clusters. Successful launches are colored green, and failed launches are colored in red.
  - Two proximal mouse positions – a coastline coordinate and a railroad coordinate – marked on the map, with text displaying their relative distance to one of the four launch sites. Polylines displaying the distance between the launch site coordinates and the coastline/railroad coordinates were also marked on the map.
- These objects were added to make it easier to identify any relationship between launch sites and likelihood of success/failure. Indeed, we were able to identify a specific launch site that had a higher likelihood of successful landings than its peer launch sites.
- Github URL: https://github.com/bcarcamo91/IBM-Capstone/blob/0f5a66cd85d5cd0dbea9dd9e80bf08da5b0d716a/F9%20Launch%20Sites%20Viz%20with%20Folium%20W3.ipynb

Build a Dashboard with Plotly Dash

- Two interactive visualizations were developed with Plotly Dash to help understand the data.
  - The first is a **pie chart** containing all launch sites and the success of each one. You can further filter the pie chart and select a specific launch site selected from a dropdown, where the pie chart will reflect the number of successful and failed launches for selected site
  - The second is **scatter plot** with Payload Mass (kg) in the x-axis and landing outcome in the y-axis. This plot is dynamic and changes based on the selected launch site from the dropdown, as well as the Payload range (kg) selected in the slider. The payload range can range from 0 to 10,000.
- These visualizations help me understand which launch site had the most successful launches, as well as what effect, if any, Payload Mass (kg) may be having on the landing outcome.
- Github URL: https://github.com/bcarcamo91/IBM-Capstone/blob/main/F9%20Plotly%20Interactive%20Dashboard%20W3.py

Predictive Analysis (Classification)

- To begin the predictive analysis portion, I first loaded my data sets into variables. I stored the dependent variable ‘class’ in a series, and the independent variables in a pandas DataFrame.
- I then used the StandardScaler method to standardize the data set before I prepared it to be split into testing and training sets. Once the data was standardized, I split the dependent and independent variables into testing and training sets using the train_test_split function. I set the parameter test_size = 0.2 which meant 20% of the data went to testing, and 80% went to training. The outcome of this step produced a total of four new variables.
- For the predictive analysis portion, I built four different machine learning models to try to find the most useful model with the highest accuracy in predicting landing outcomes. For each model, I used GridSearchCV to find the best tuned hyperparameters for each model programmatically. The four models that I trained included the following:
  - Logistic Regression
  - Support Vector Machines
  - Decision Tree classifier
  - K-Nearest Neighbors
- Github URL: https://github.com/bcarcamo91/IBM-Capstone/blob/main/F9%20Spacex%20ML%20Predictions%20W4.ipynb

EDA results

There were quite a few initial insights I gathered after going through exploring data analysis exercise.

- The first is that certain launch sites tended to have higher rates of success than other launch sites which were less fortunate. For example, VAFB SLC 4E barely had any unsuccessful landings, while CCAFS SLC40 tended to see a lot more failures.
- There also seemed tobe a relationship between payload and launch site when it came to successful landings. I noticed that CCAFS SLC 40 and KSC LC 39A had very successful landings above a payload of 14,000 kg.
- I looked at the relationship between orbit and the mean of the class column, which determines success/failure. For some orbits, like ES-L1, and GEO, the mean was 1.0, which meant that on average they always landed successfully. Others had less appealing means, such as GTO with a mean of ~0.5. Certainly the orbit could be a meaningful independent variable in the machine learning modeling exercise.
- Finally, I noticed that the date had a strong relationship with the success rate. The success rate really took off in 2013 and has hovered around the 0.8 mark since 2018. It seems that the more recent the rocket launch, the more likely it will land successfully after the first stage. This makes sense as SpaceX is presumably improving their records over time.

Interactive Analytics Results

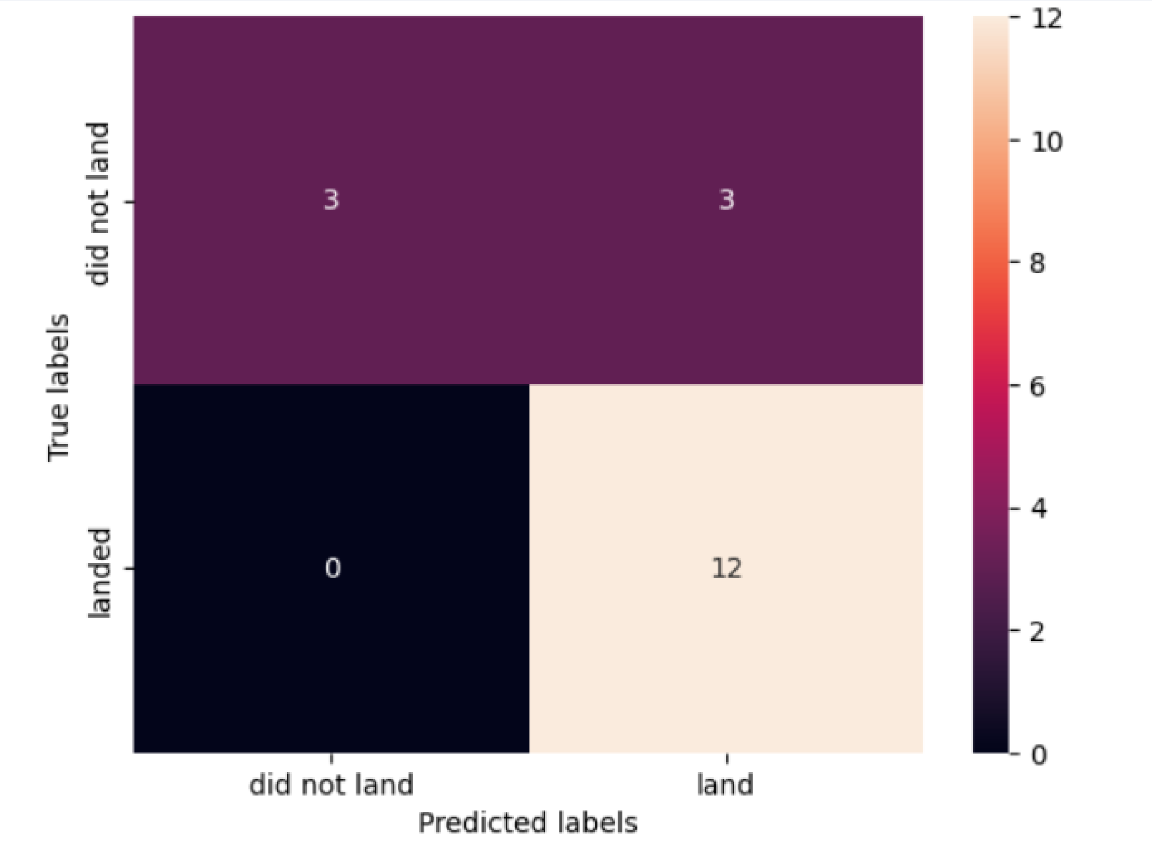
- What I found going through the interactive analytics results in Plotly was that certain launch sites had a higher rate of success than others. The KSC launch site was hovering above 75% success rate, while the worse launch site was closer to 60%. This felt like a significant difference and, therefore, I knew that launch site would be relevant in predicting landing success rate.
- Furthermore, by looking at the payload scatter chart, we found that some booster version categories had higher success rates than others when looking at specific payload ranges and launch sites. Therefore, there appeared to be some interaction between some of these variables in being able to predict success outcomes.

Predictive analysis results

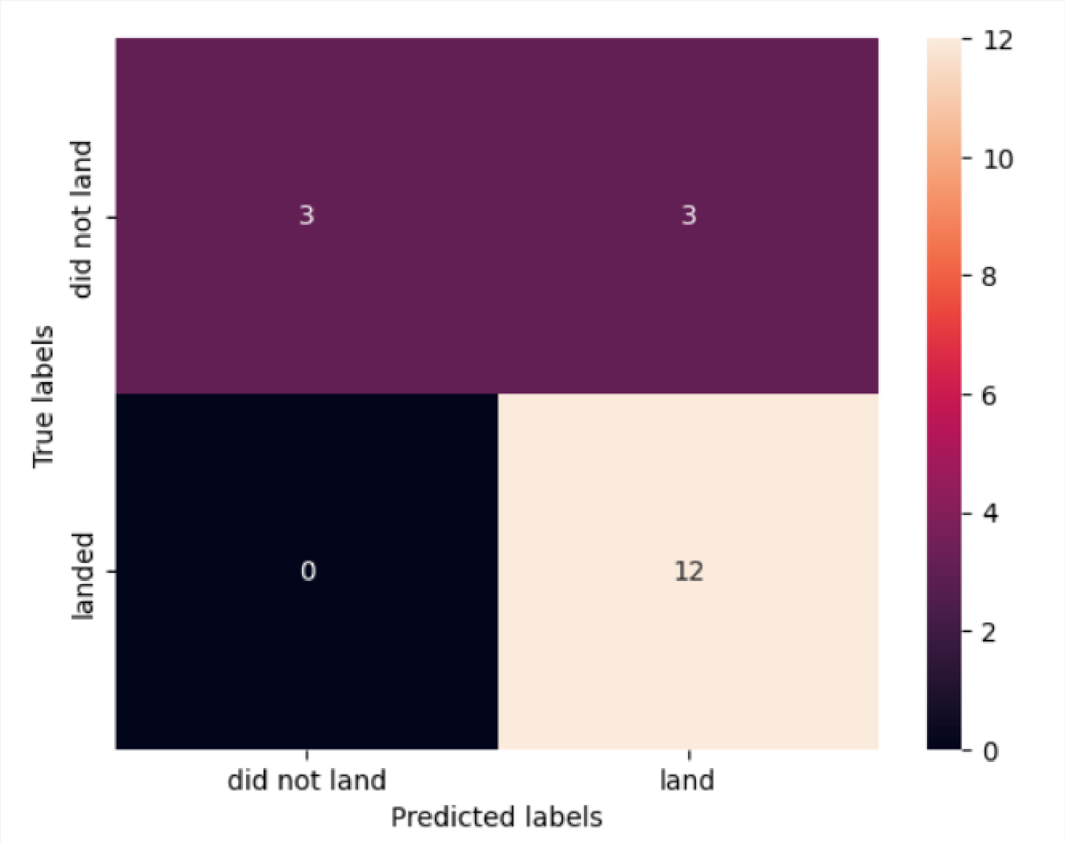
- Once the four models were built and the best hyperparameters were found for each model, I selected the model that had the highest accuracy.
- Interestingly, the Decision Tree Classifier had a significant lower accuracy than the other three models. In fact, I found that Logistic Regression, Support Vector Machines, and K-Nearest Neighbors performed equally well when the accuracy was compared. Therefore, I concluded that either of the three could be used to effectively predict successful Falcon 9 rocket landings.
- The model to steer away from using as a predictive model is the Decision Tree classifier model, with an accuracy of 0.72.

Predictive analysis confusion matrices

Logistic Regression

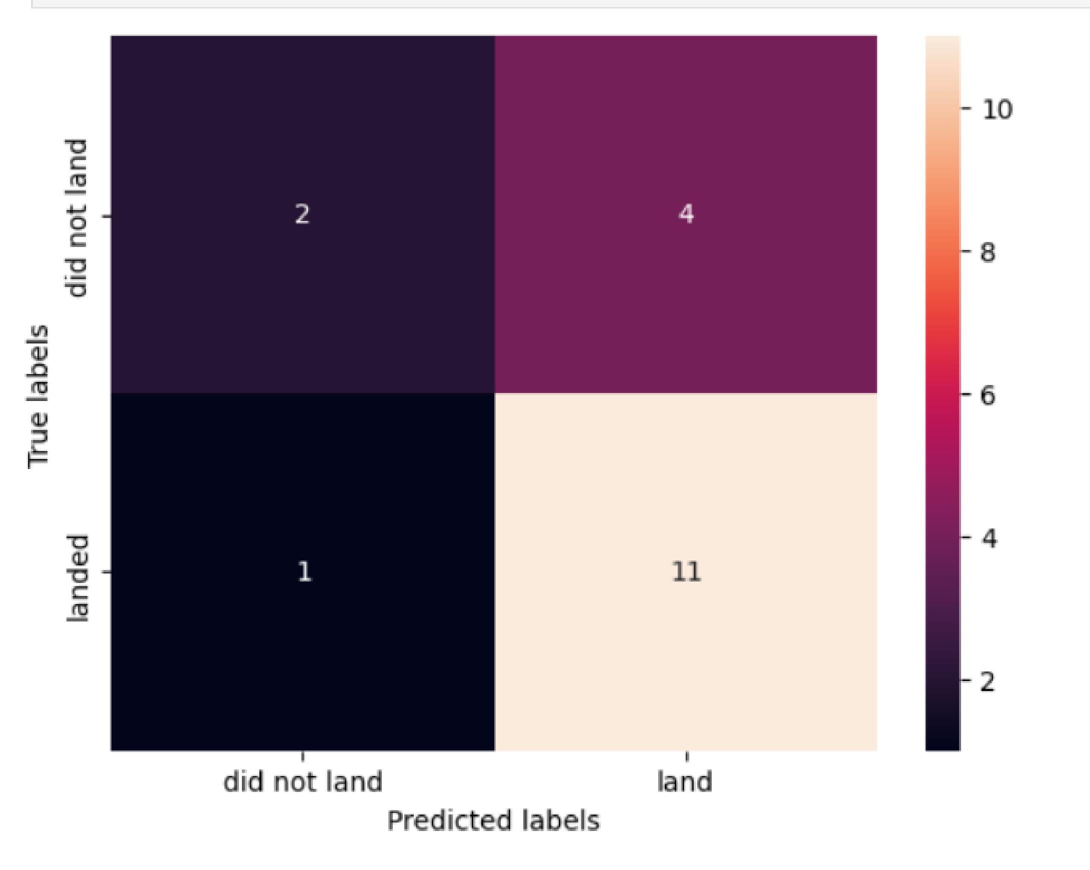


Support Vector Machines

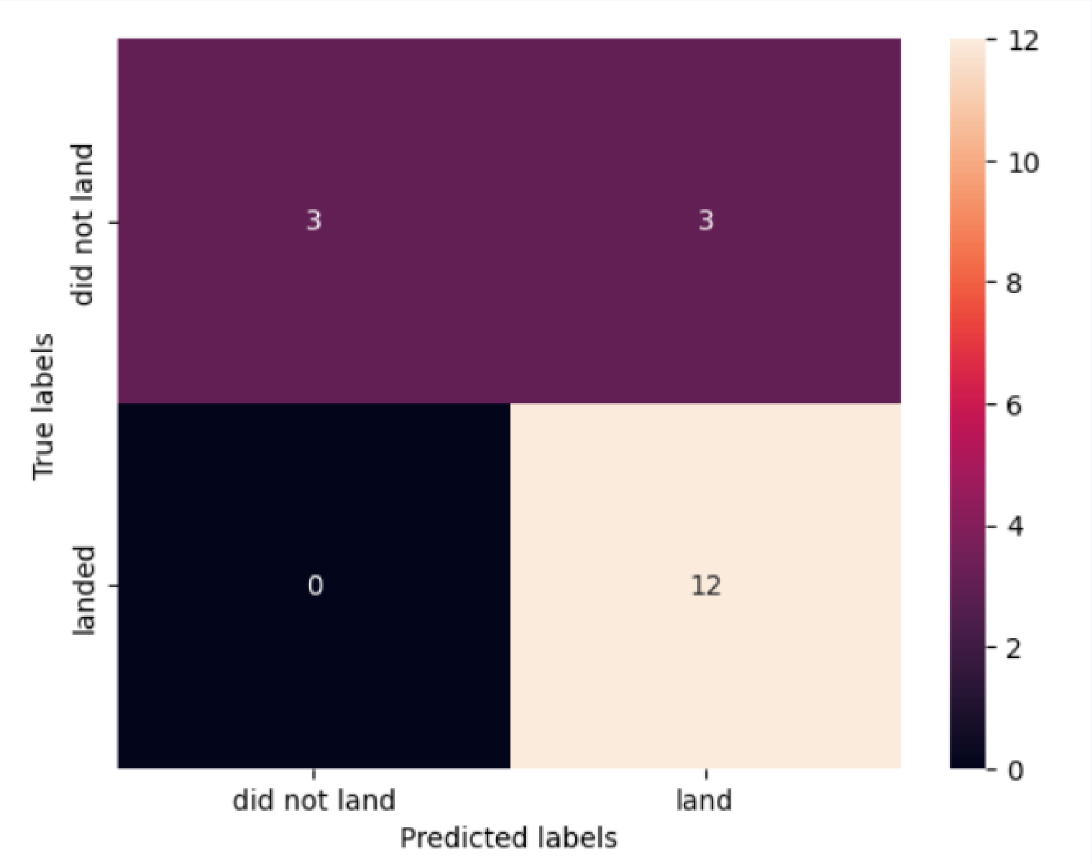


Predictive analysis confusion matrices cont.

Decision Tree Classifier Confusion Matrix



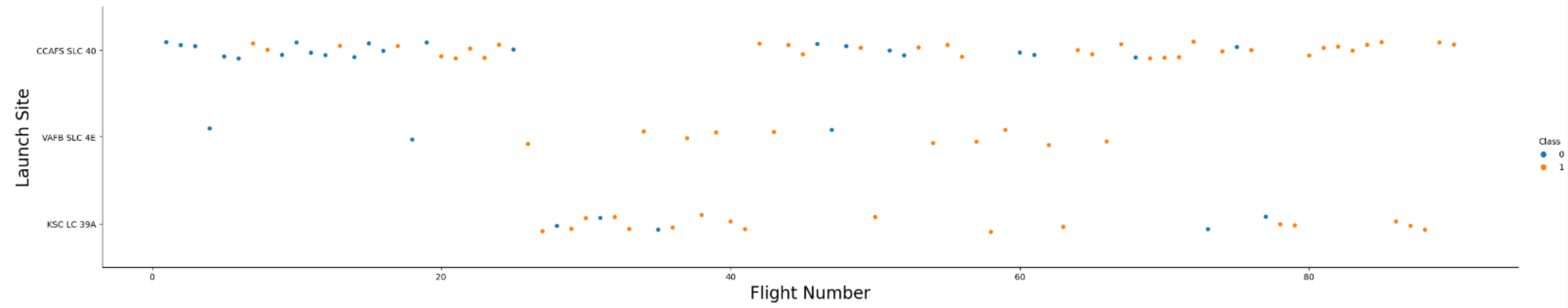
K-Nearest Neighbors



Section 2

Insights drawn from EDA

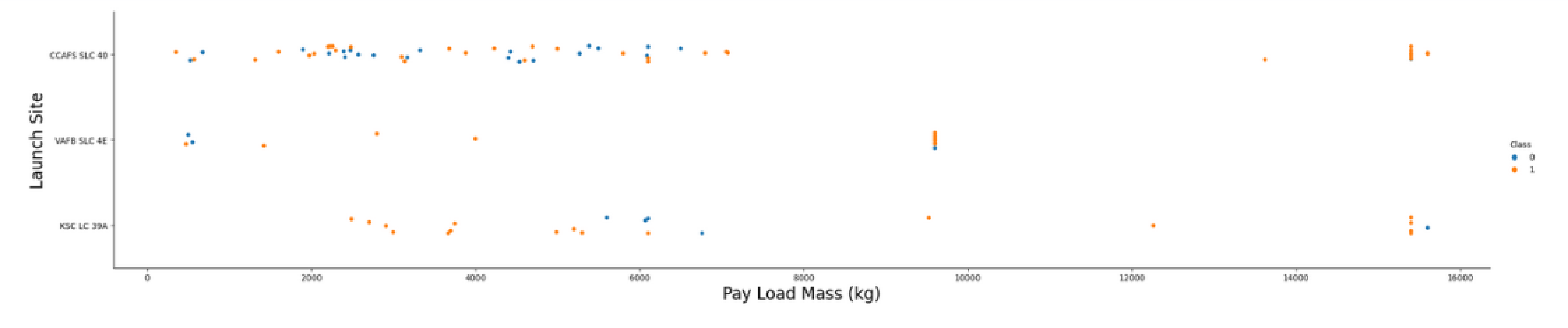
Flight Number vs. Launch Site



We can see that VAFB stopped running launches after the 60th flight number. We can also see that CCAFS had a higher success rate later on than in the beginning. This is noted by how may orange circles there are in higher flight numbers relative to smaller flight numbers.

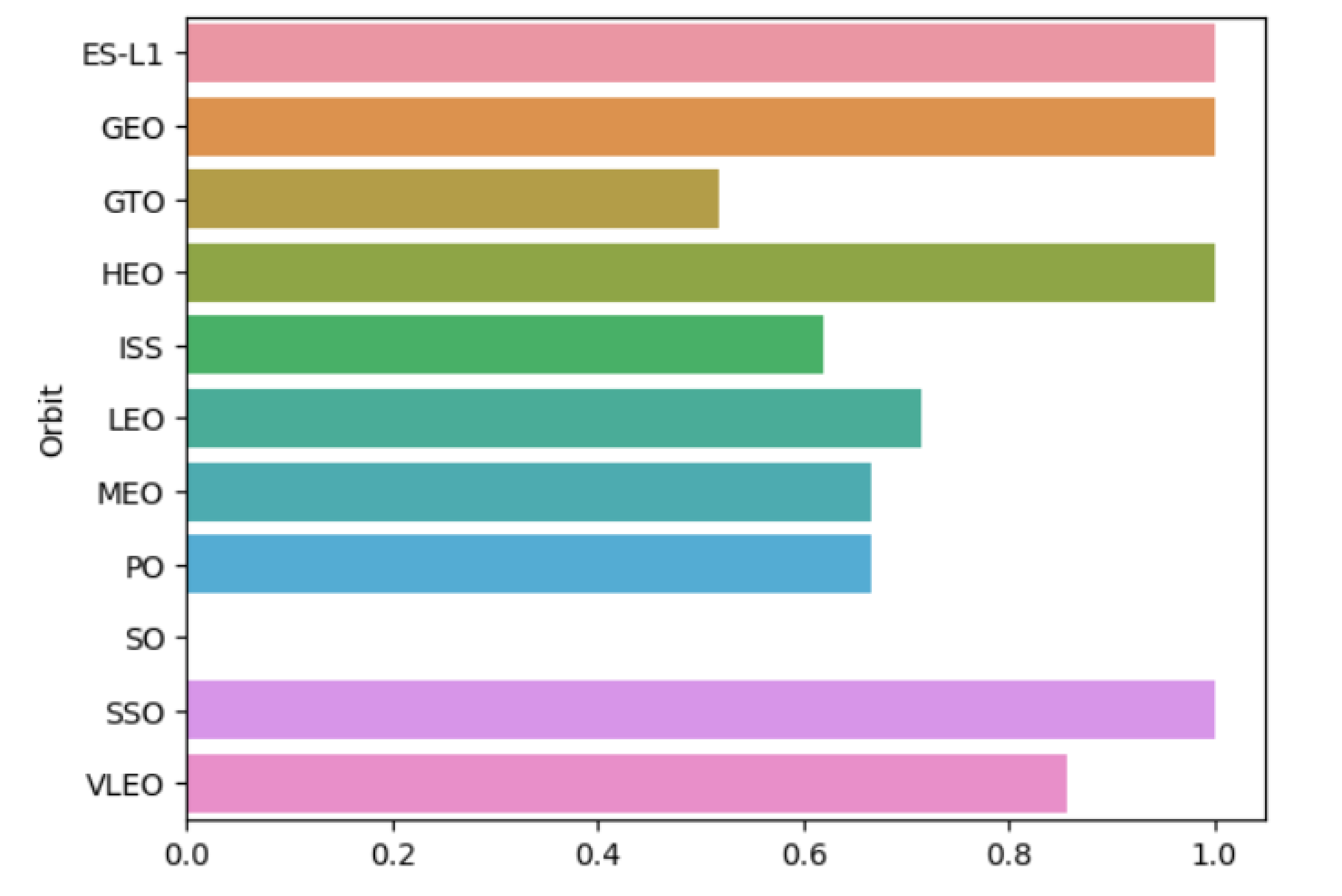
Payload vs. Launch Site

It appears that when the payload is below 8,000, first stage landings are more likely to fail then when the payload is above it. The second insight from this chart is that rockets with a payload higher than 10,000 do not fly out of certain launch sites.



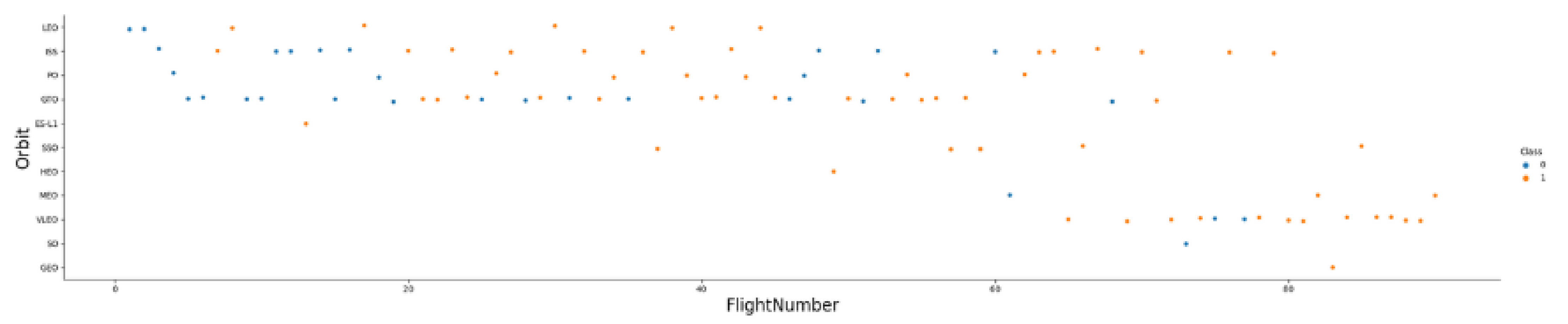
Success Rate vs. Orbit Type

There appears to be a strong relationship between success rate and orbit types. Some orbit types have a 100% success rate (average shown in chart), whereas other orbit types are much more fallible. This difference seems significant and worth testing in the ML classification models.



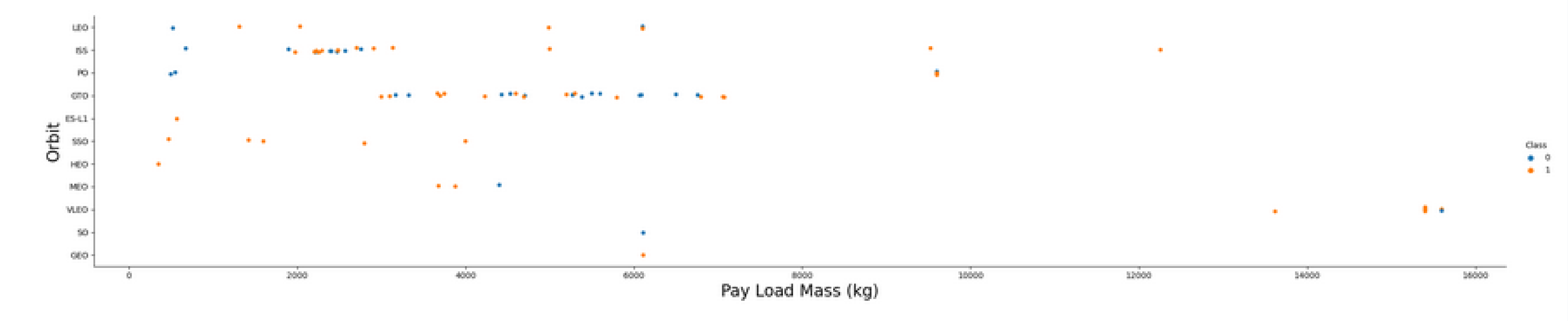
Flight Number vs. Orbit Type

In the chart below, the orbits are represented in the rows. We can see that some of the orbits closer to the x-axis have high success rates past a certain flight number (bottom right of the chart). Most of the data points there are orange, which indicate a successful landing. On the flip side, some of the orbits in the higher rows tend to see a lot more failures. In this case, a lower flight number is associated with an earlier date. Therefore, the date seems to be related to the success rate. The earlier the date, the lower the success rate across various orbits, it appears. The later the date, the more successful stage 1 landings, though the orbits seem to change.



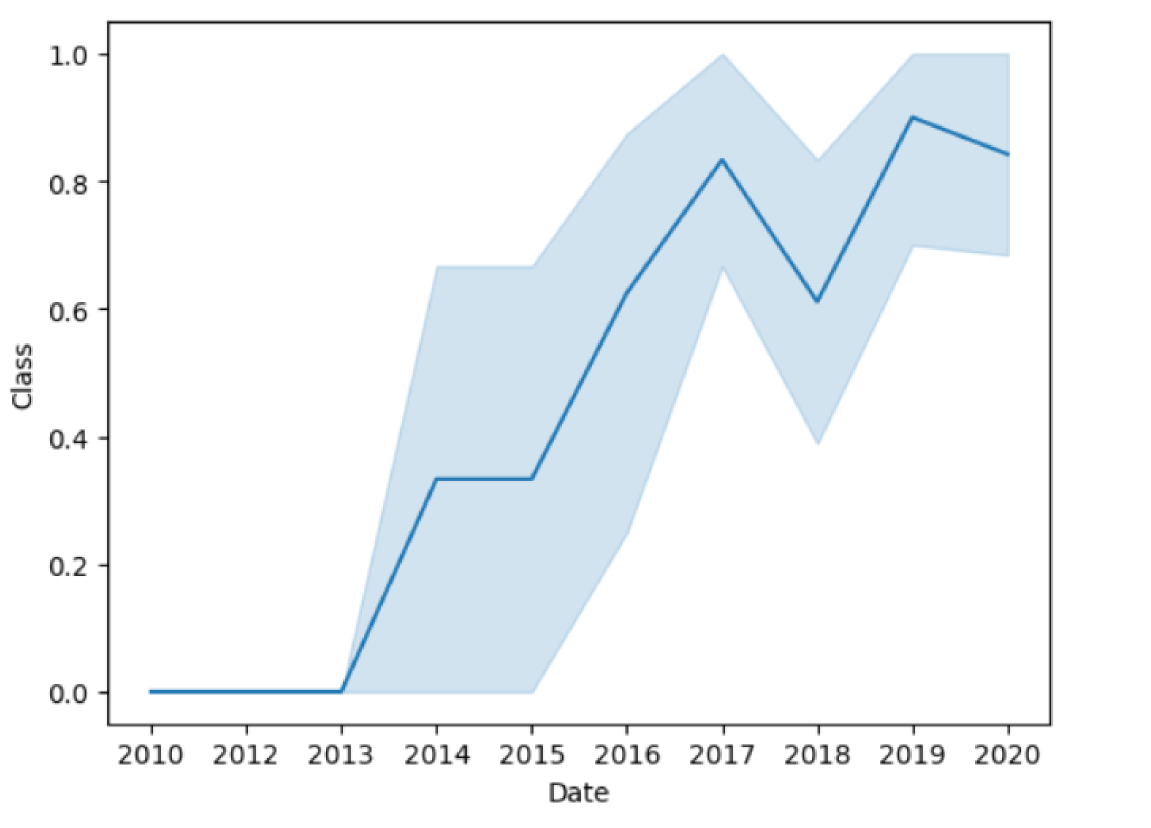
Payload vs. Orbit Type

There is only one orbit with a payload mass past 12,000 kg, and for the most part, it tends to have successful stage 1 landings. Most of those data points are colored orange. Furthermore, some orbits tend to have lower success rates within a specific payload range, in this case between 4,000 to 6,000 kg, where we see a higher concentration of blue data points. Lastly, a third insight is that some orbits have only ever had successful stage 1 landings, which is noteworthy.



Launch Success Yearly Trend

The yearly trend of success rate plotted against the date in a line graph is extremely insightful. This shows how the success rate has increased dramatically since its lowest around 2013, where the success rate was hovering between around 0.2. It reached its peak in 2019 at around 0.9, which means 9/10 Falcon 9 rockets that were launched landed successfully after the first stage, compared to about 3/10 Falcon 9 rockets that were launched in 2013. That's a 300% increase in the success rate over a 6-year time frame.



All Launch Site Names

This query lets us see all the unique launch site names in the data set. We achieve this primarily by using the DISTINCT operator which takes all the unique values in any particular column and returns them. In this case, our column is "Launch_Site".

In [15]:
```
%%sql SELECT DISTINCT "Launch_Site"
FROM SPACEXTABLE
```

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

Out[15]:

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

Launch Site Names Begin with 'CCA'

This screenshot shows the query selecting all launches with launch site beginning with CCA. I limited the results to the first 5 rows.

In [16]:

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

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

Out[16]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outc
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (paracl
2010-08-12	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 (paracl
				Dragon					

Total Payload Mass

This query takes the sum of the payload mass column and displays it in a column renamed to TOTAL_PAYLOAD when only looking at NASA as a customer. We can see that the total payload of the rockets launched in this data set is 45,596 KG for NASA.

In [12]:
```
%%sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD
FROM SPACEXTABLE
WHERE Customer = 'NASA (CRS)'
```

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

Out[12]:

TOTAL_PAYLOAD
45596

Average Payload Mass by F9 v1.1

This query uses the AVG operator to get the average payload mass from the payload mass column, while filtering the booster version for F9 V1.1. Wee see that the average for this booster version is 2534.66kg.

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

In [33]:

```
%%sql SELECT AVG(PAYLOAD_MASS__KG_)
FROM SPACEXTABLE
WHERE Booster_Version LIKE 'F9 v1.1%'
```

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

Out[33]:

AVG(PAYLOAD_MASS__KG_)
2534.6666666666665

First Successful Ground Landing Date

This query selects the minimum of the date column to find the earliest date in which a rocket landed successfully after the fist stage. It occurred on April 6, 2010.

In [34]:
```
%%sql SELECT MIN(DATE)
from SPACEXTABLE
WHERE Mission_Outcome = 'Success'
```

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

Out[34]:

MIN(DATE)
2010-04-06

Successful Drone Ship Landing with Payload between 4000 and 6000

We can see that successful drone ship landings with payload between 4000 and 6000 came from either launch site CCAFS LC 40, or KSC LC 39A. The payloads appeared to be random.

In [35]:

```
%%sql SELECT *
from SPACEXTABLE
where Landing_Outcome = "Success (drone ship)" AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

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

Out[35]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_
2016-06-05	05:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	
2016-08-14	05:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	
2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	
2017-11-10	22:53:00	F9 FT B1031.2	KSC LC-39A	SES-11 / EchoStar 105	5200	GTO	SES EchoStar	

Total Number of Successful and Failure Mission Outcomes

Here we can see the number of successful and failure mission outcomes. As we can see, there was only one recorded failed mission outcome, and the rest were successes. Therefore, mission outcome is unlikely to be a strong predictor of landing outcome.

In [39]:

```
%%sql SELECT Mission_Outcome, COUNT(*) AS TOTAL
FROM SPACEXTABLE
GROUP BY Mission_Outcome
ORDER BY Mission_Outcome
```

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

Out[39]:

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

Boosters Carried Maximum Payload

This query shows the booster versions that carried the maximum payload. Based on this query result, it looks like there's only one booster version, the B5, which can carry the maximum payload.

```
In [40]: %%sql SELECT Booster_version
         FROM SPACEXTABLE
         WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)

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

Out[40]:

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

2015 Launch Records

This query selects the date, month, landing outcome, booster version, and launch site where the landing outcome was a drone ship failure, and the year = 2015. We can see two failures, both from the same booster version F9 v1.1 and launch site.

```
In [41]: %%sql SELECT Date, substr(Date, 6, 2) as MONTH, Landing_Outcome, Booster_Version, Launch_Site
         FROM SPACEXTABLE
         WHERE Landing_Outcome LIKE "Failure (drone ship)" AND substr(Date, 1, 4) = '2015'

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

Out[41]:

Date	MONTH	Landing_Outcome	Booster_Version	Launch_Site
2015-10-01	10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015-04-14	04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

This groups the count of landing outcomes by landing outcome category in descending order. The most frequent landing outcome was “No attempt” at 10, followed by two Success landing outcomes (5 each), and then a drone ship failure. We see that most landing outcomes over those 7 years were “No attempt”, but a fair portion oft hose who did attempt, landed successfully either by ground pad or drone ship.

In [43]:
```
%%sql SELECT Landing_Outcome, COUNT(Landing_Outcome)
FROM SPACEXTABLE
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY COUNT(Landing_Outcome) DESC
```

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

Out[43]:

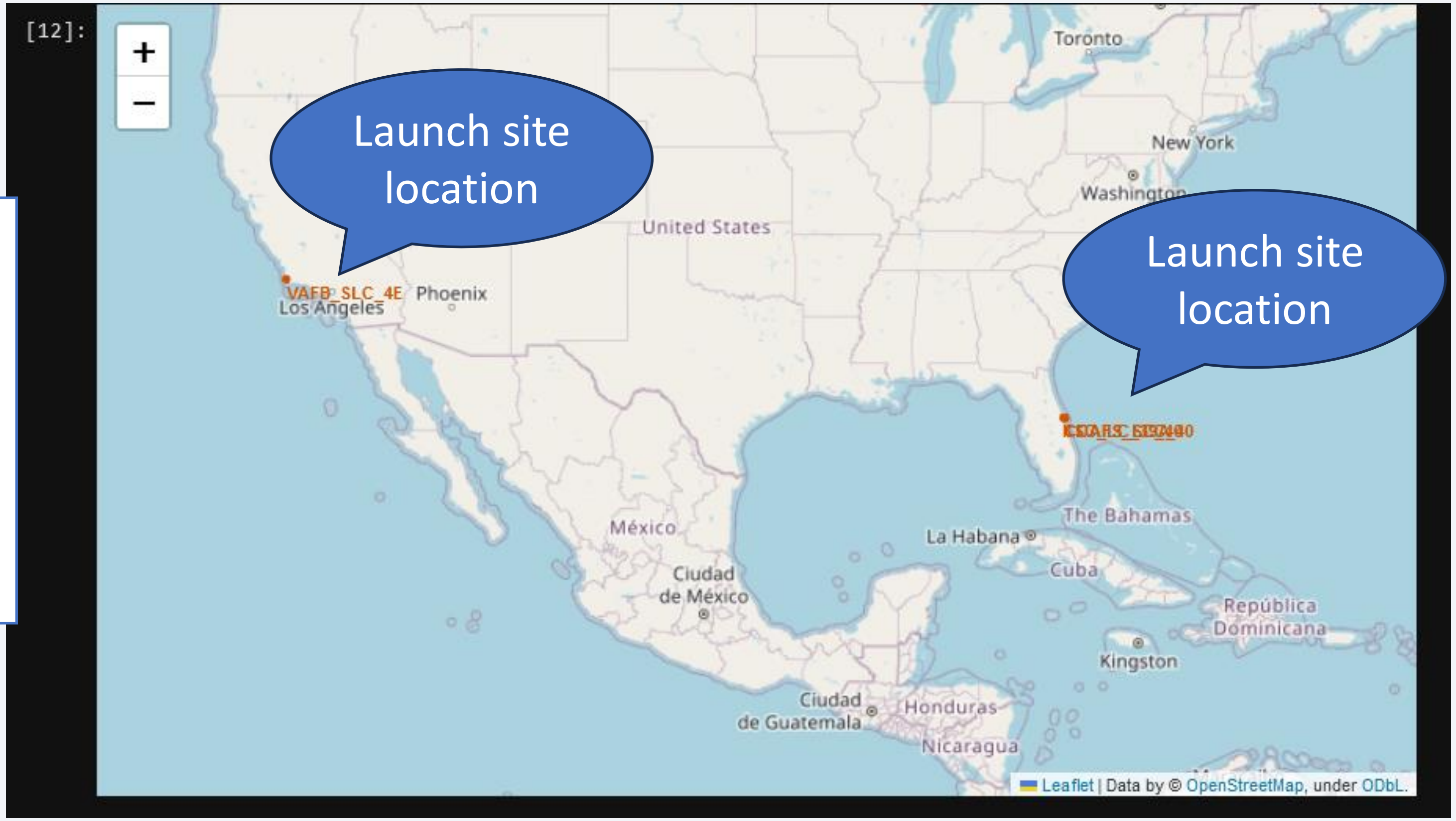
Landing_Outcome	COUNT(Landing_Outcome)
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

Section 3

Launch Sites Proximities Analysis

SpaceX Launch Site Locations - Folium

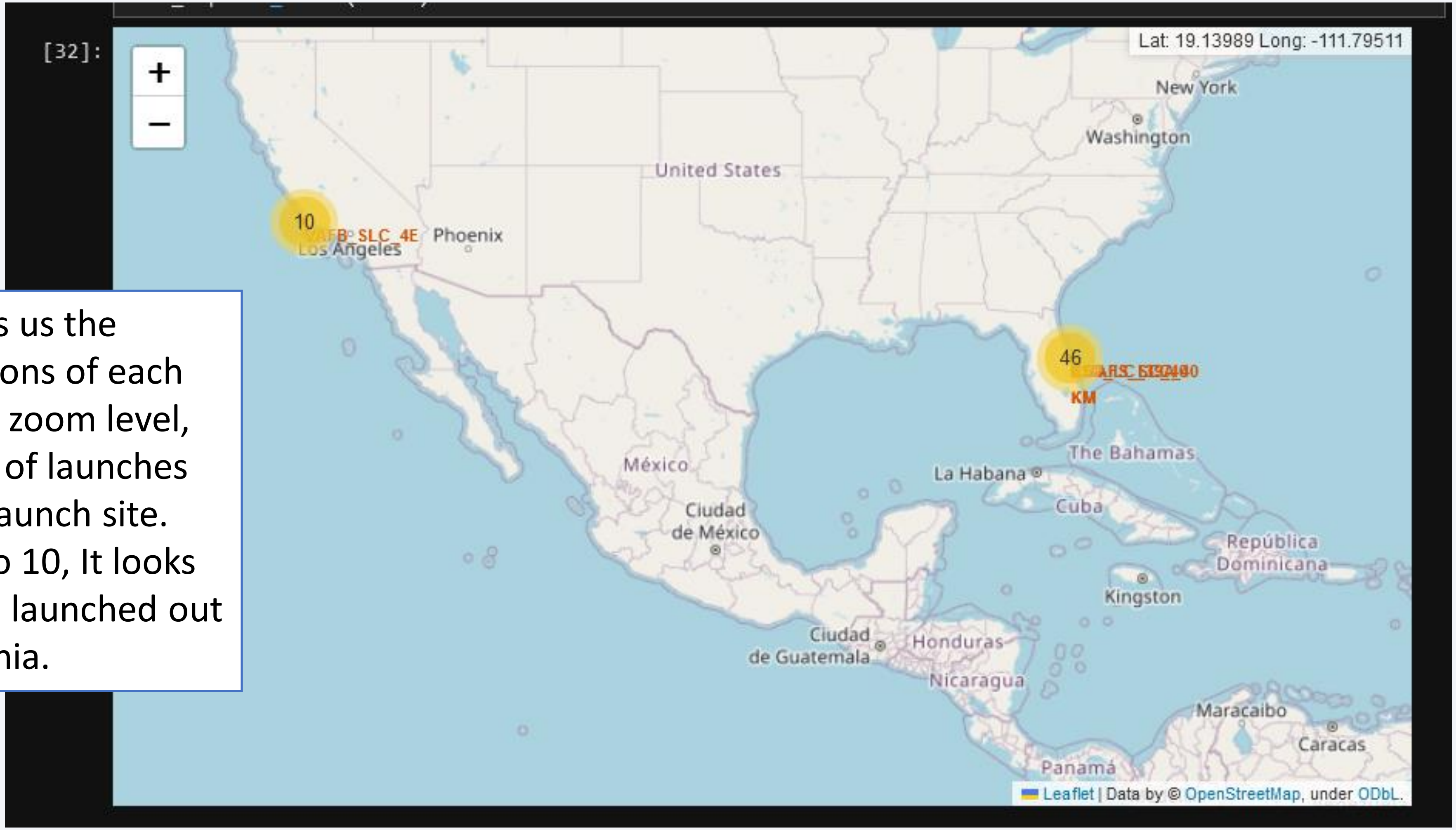
This Folium map shows the launch site locations that SpaceX used to launch its Falcon 9 rockets.
You can see that some of them were launched near Los Angeles, but most launch site locations are located in Florida.



SpaceX Launch Site Locations - Folium

When zooming in on the state of Florida, we can see three launch site locations labeled with a red circle on the map. Two of them are stacked on top of each other on the right side of the Folium map.

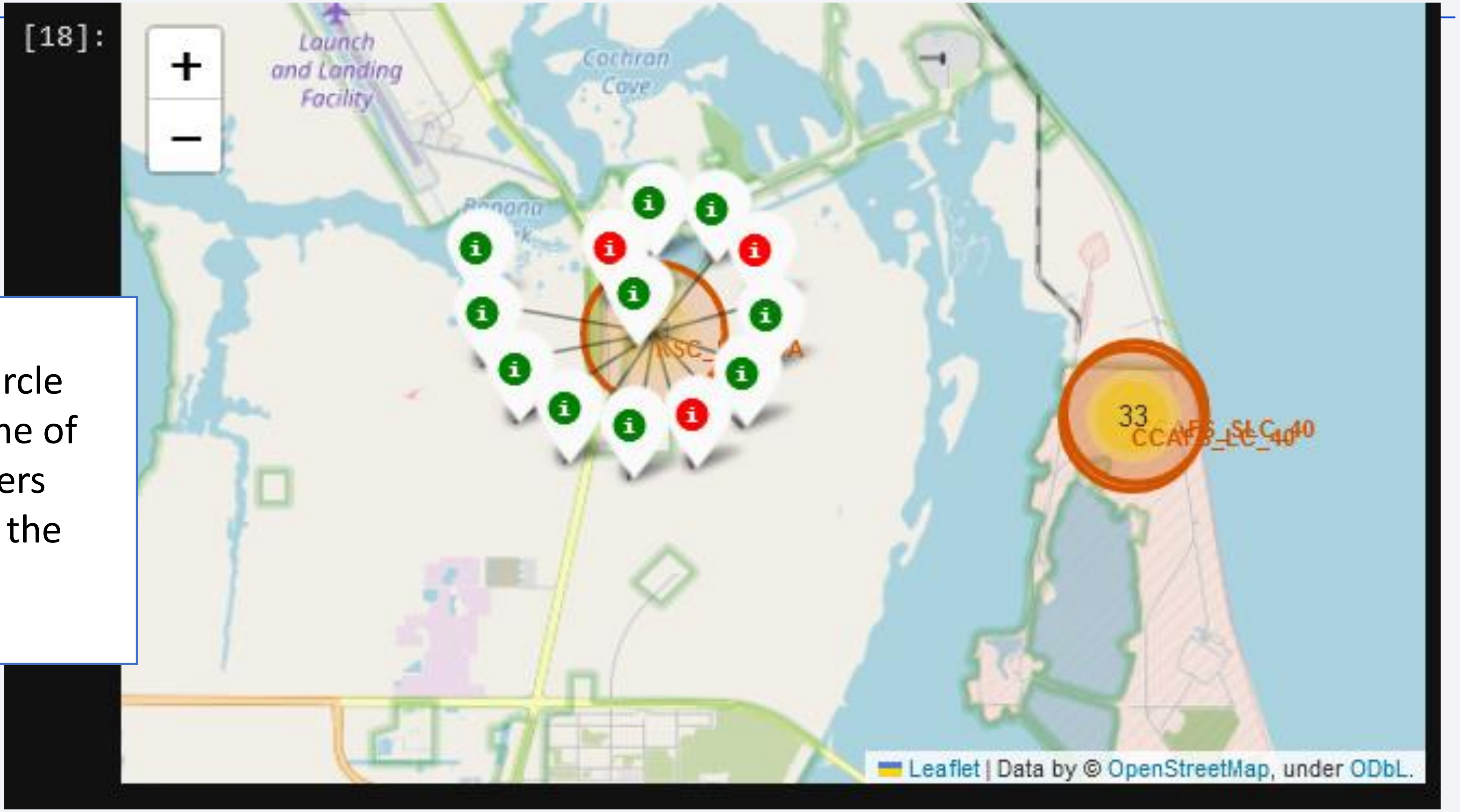
SpaceX Launch Outcome Locations - Folium



This folium map shows us the launch outcome locations of each launch. At this current zoom level, you can see the count of launches near each respective launch site. When comparing 46 to 10, It looks like most rockets were launched out of Florida than California.

SpaceX Launch Outcome Locations – Folium cont.

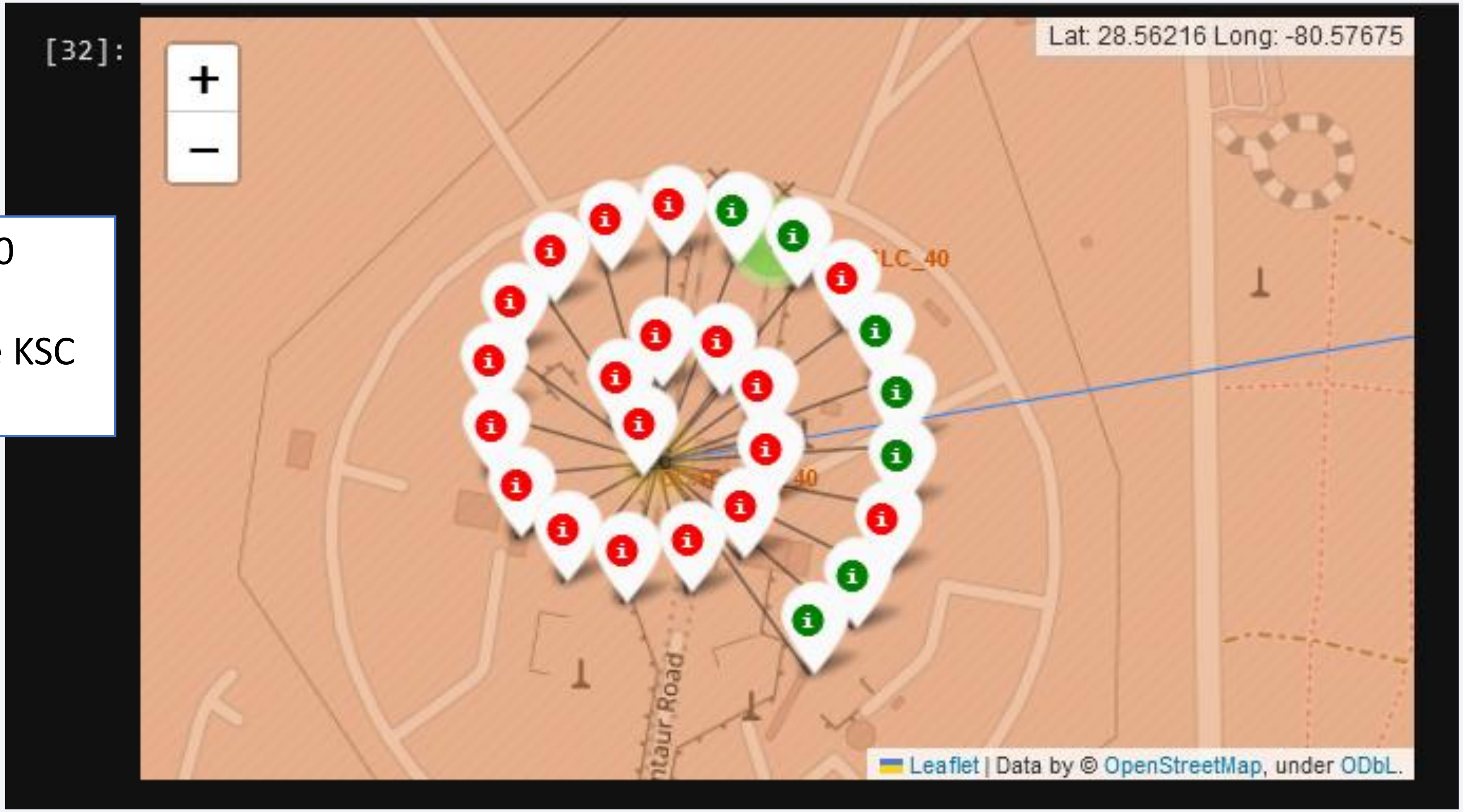
Zooming in a little further on Florida, we can click on the circle and reveal the launch outcome of each launch. The green markers indicate a successful landing, the red markers indicate an unsuccessful landing.



Launch outcome markers displayed for KSC LC 39A launch site

SpaceX Launch Outcome Locations – Folium cont.

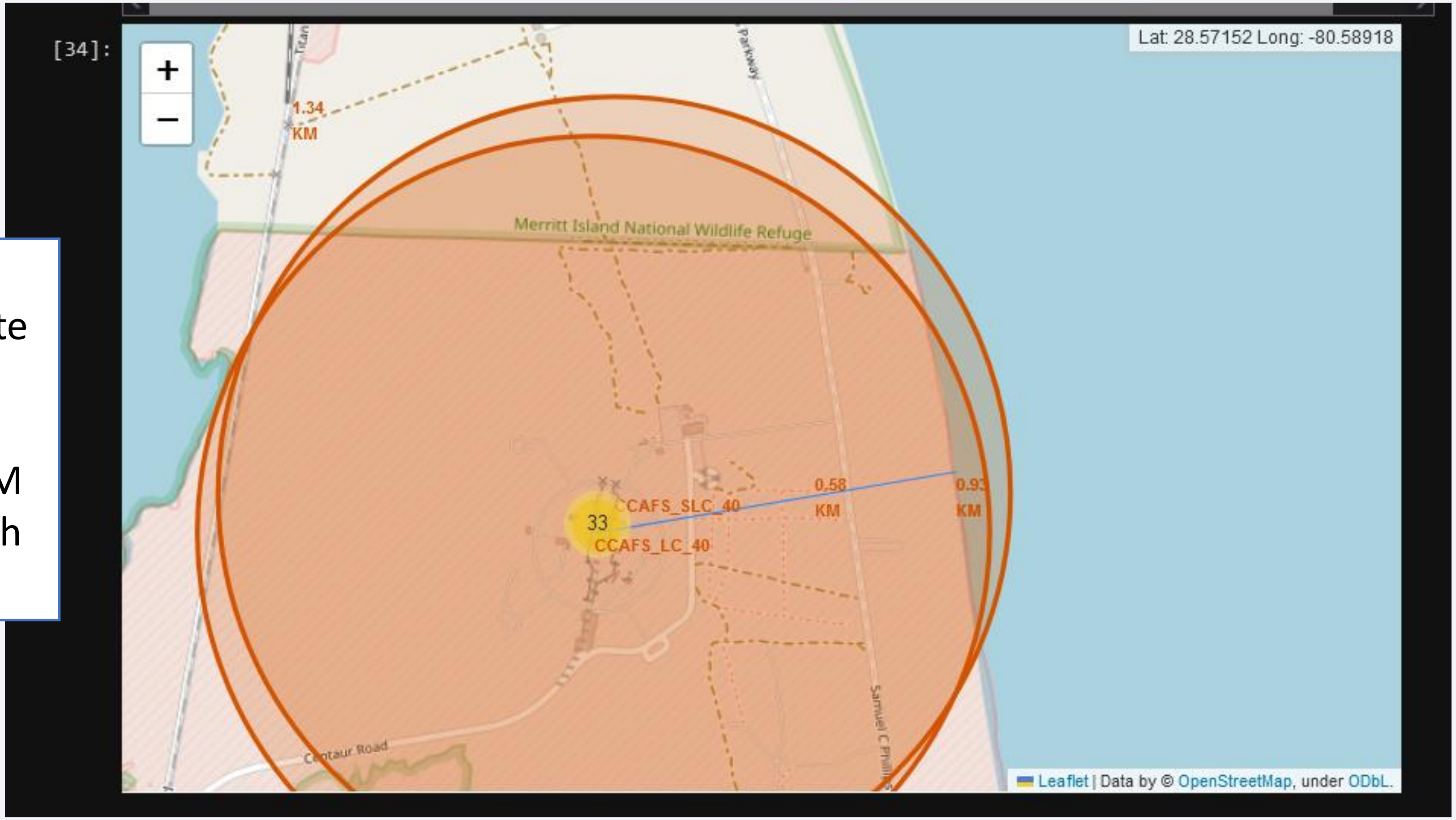
It appears that the CCAFS LC 40 launch site has much more unsuccessful landings than the KSC LC 39A launch site.



Launch outcome markers displayed for CCAFS LC 40 launch site

SpaceX Launch Site Proximity to Coastline - Folium

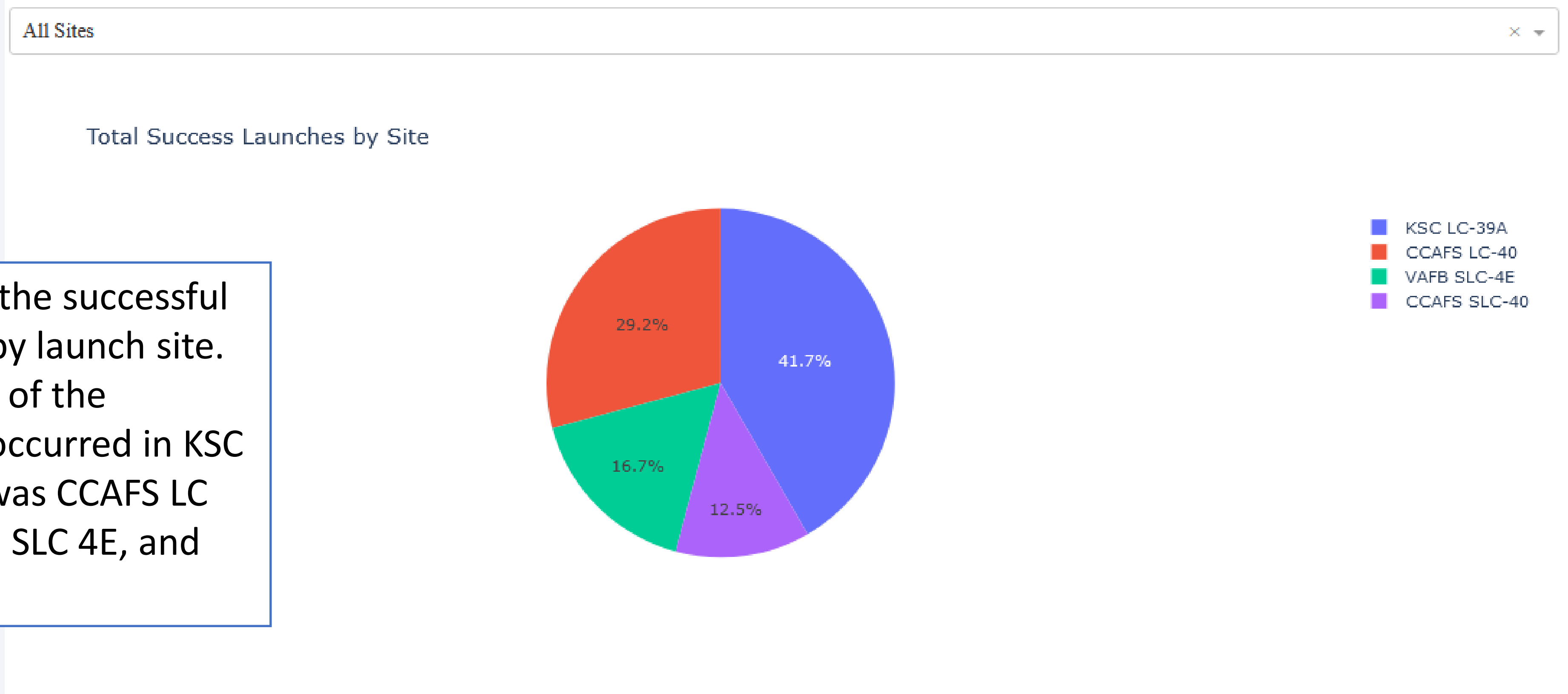
In this map, we can see a blue polyline drawn from the launch site CCAFS SLC 40 to a coastline coordinate nearby. The coastline coordinate shows that it is 0.93 KM away, not far away from the launch site.



Section 4

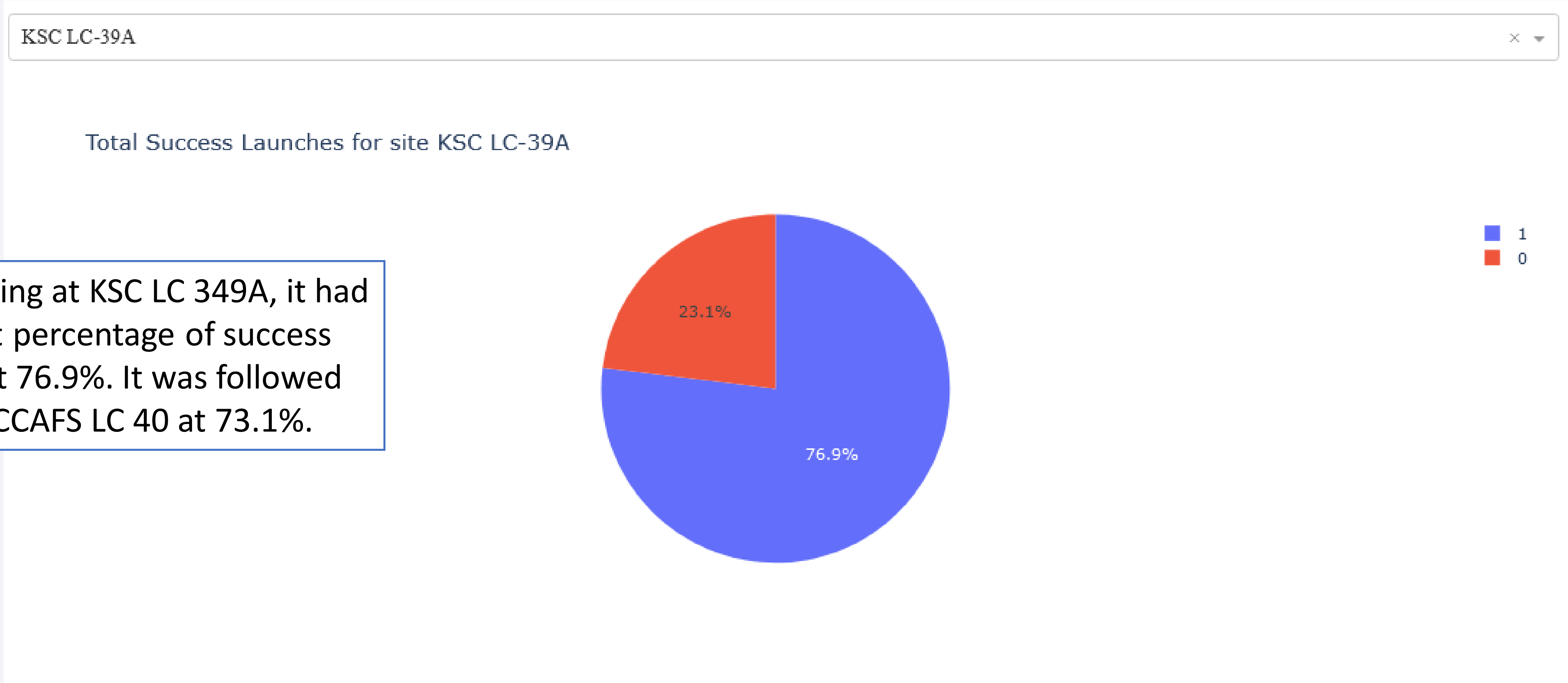
Build a Dashboard with Plotly Dash

SpaceX Launch Success Count for All Sites



This pie chart shows the successful launches broken up by launch site. As you can see, most of the successful launches occurred in KSC LC 39A. Next in line was CCAFS LC 40, followed by VAFB SLC 4E, and lastly, CCAFS SLC 40.

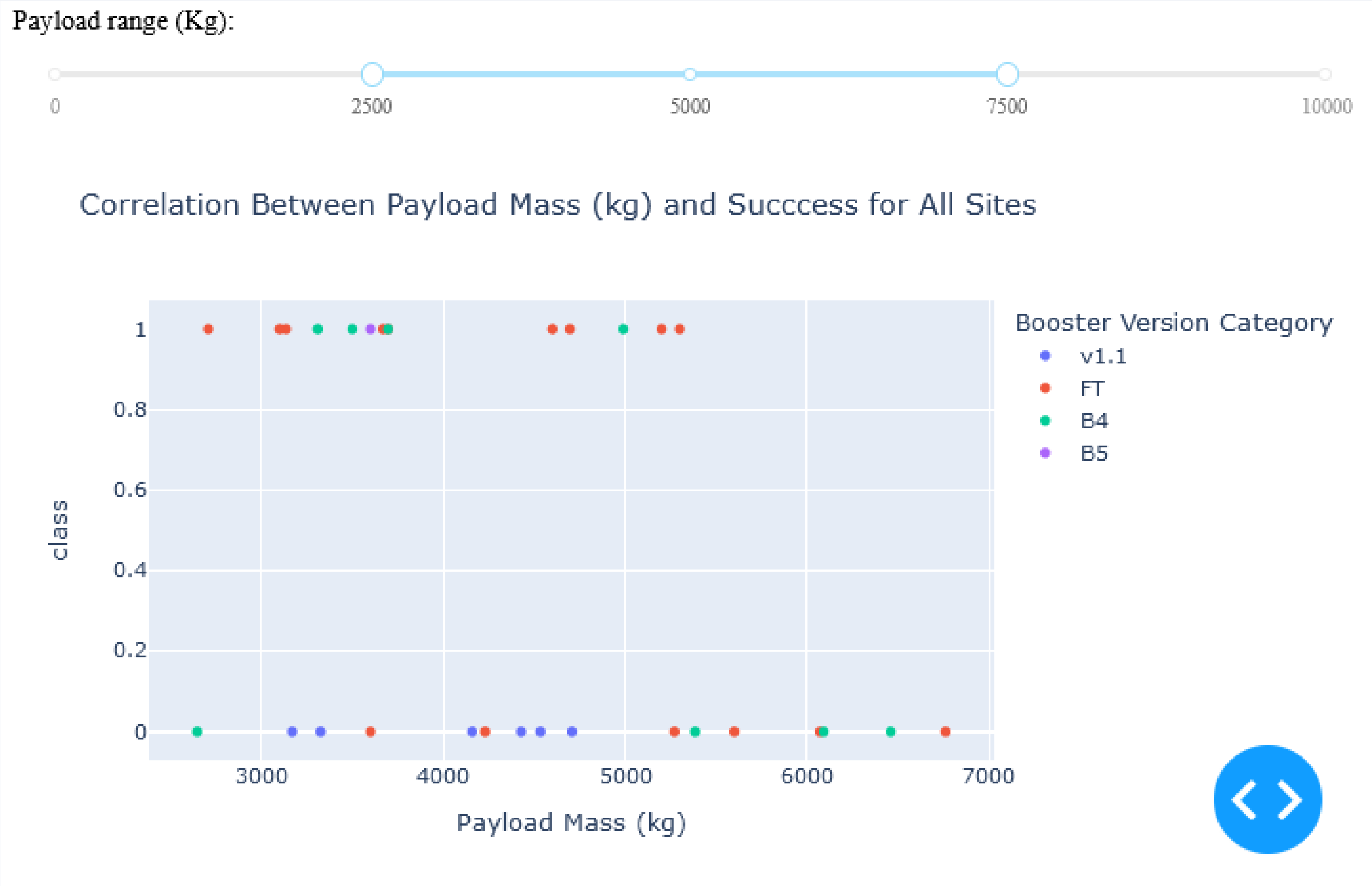
SpaceX Launch Success Count for KSC LC 39A



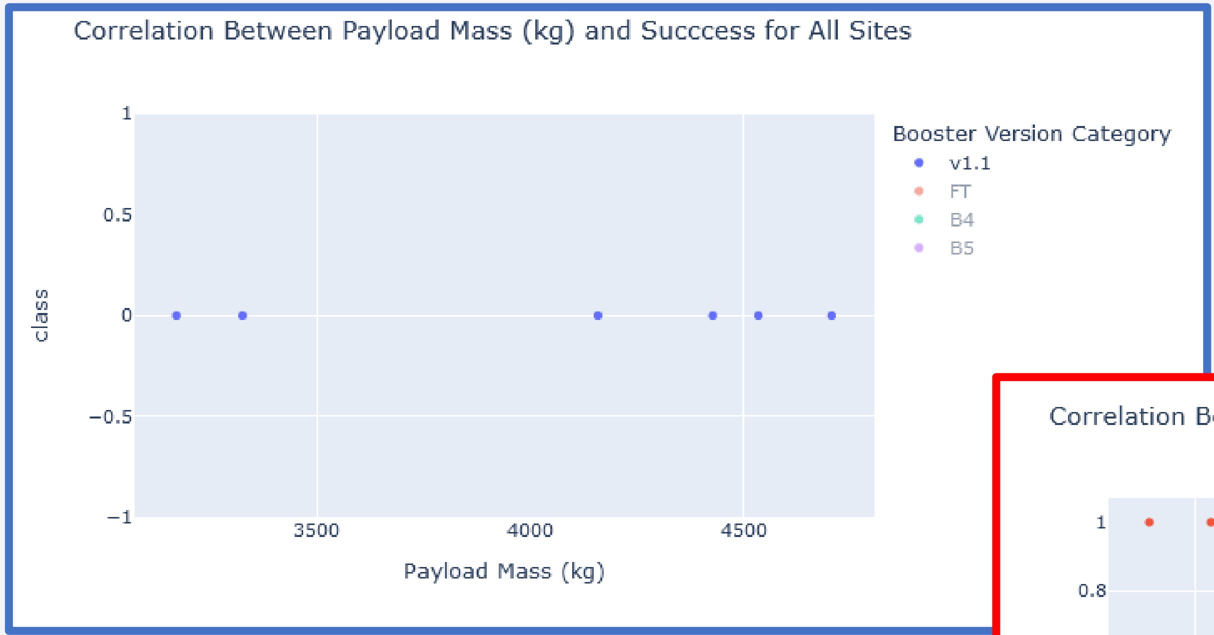
When looking at KSC LC 349A, it had the highest percentage of success launches at 76.9%. It was followed closely by CCAFS LC 40 at 73.1%.

SpaceX Payload Mass x Outcome scatter by BVC

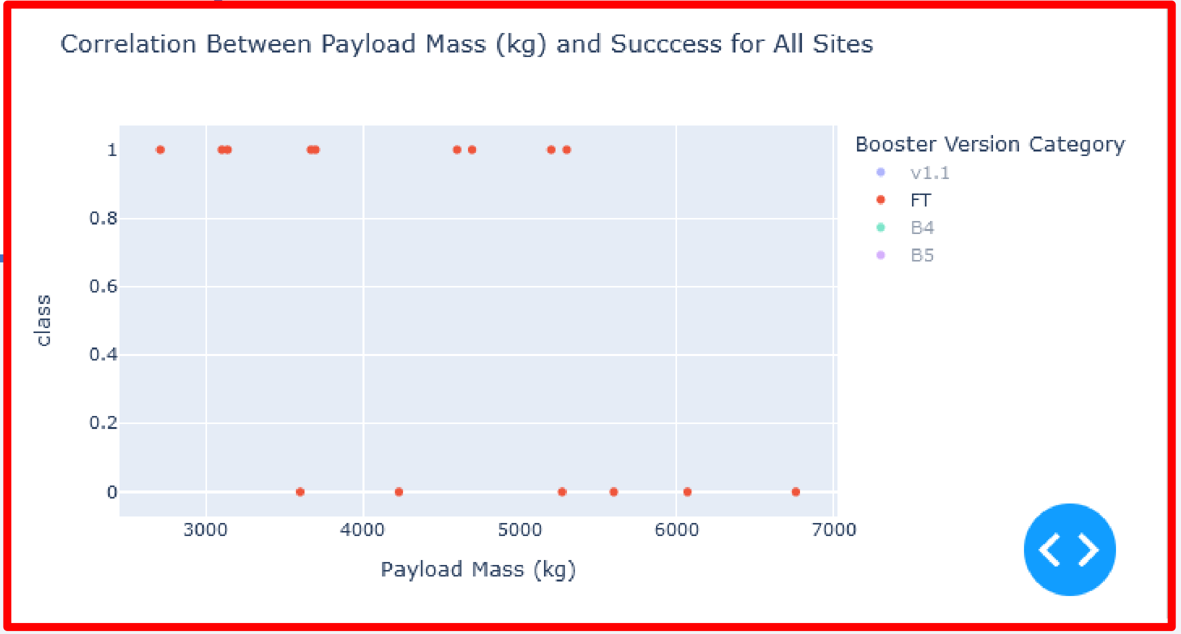
This scatter chart shows the launch outcomes by payload mass for the various booster version categories. We can see that we tended to se fewer successful outcomes across all booster version categories the higher the payload.



SpaceX Payload Mass x Outcome scatter by BVC

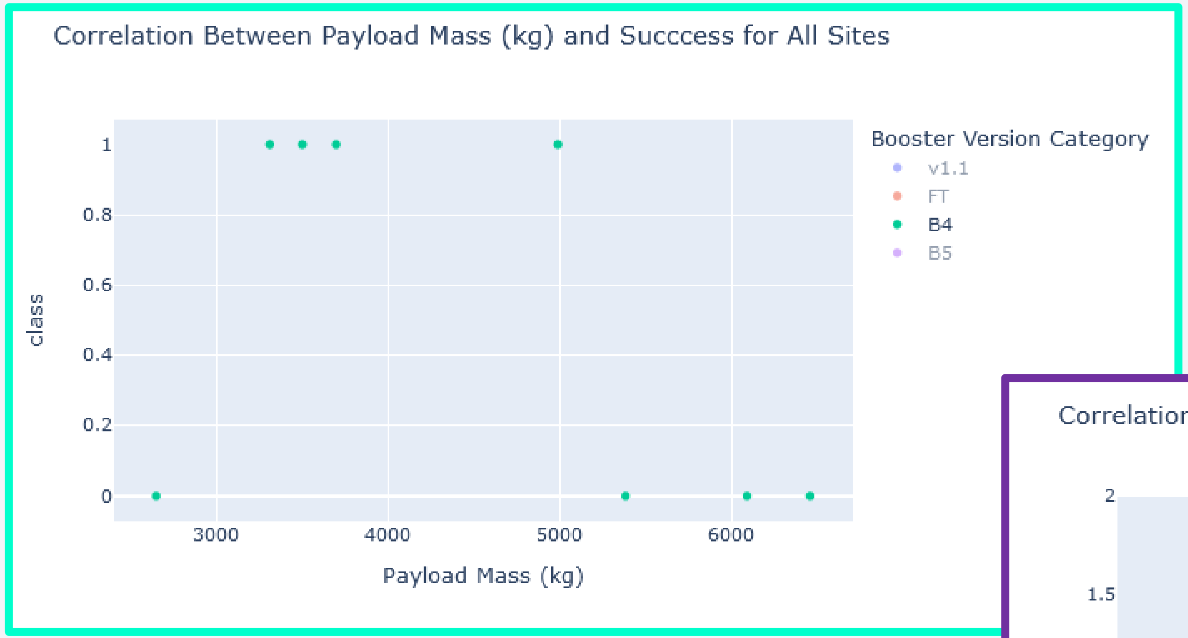


BVC v1.1 did not have *any* successful launches, regardless of payload

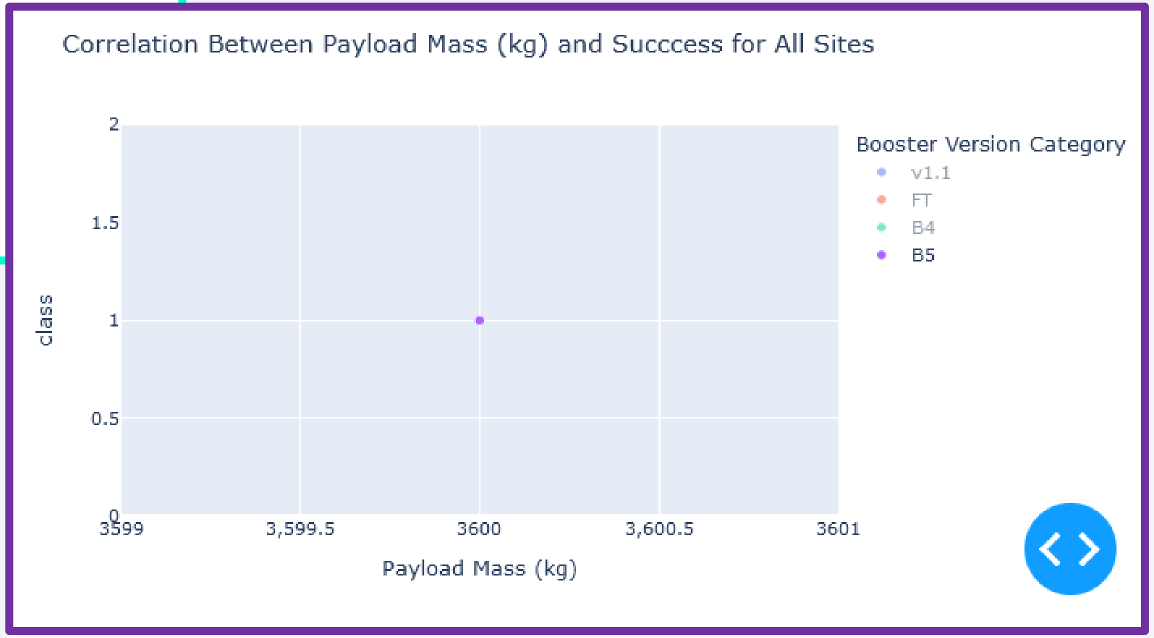


BVC FT both successful and unsuccessful launches, but seemed to see less success the higher the payload

SpaceX Payload Mass x Outcome scatter by BVC



BVC v1.1 did not have *any* successful launches



BVC B5 only had one successful launch when payload is between 2500 and 7500

Section 5

Predictive Analysis (Classification)

Classification Accuracy

```
[71]: import plotly.express as px
fig = px.bar(df2, x='Model', y='Accuracy', title='Model Accuracy', text_auto=True)
fig.show()
```

When viewing the model accuracy across models, the logreg, svm, and KNN models are tied. The decision tree model comes in last with an accuracy of 0.72. Therefore, I would recommend using any of the three models listed, and deprioritize using the Decision tree model based on its accuracy on the testing data set.

Classification Accuracy cont.

Accuracy calculated using the .score() method

```
#Task 12
#Find the method that performs best
print('Model accuracy')
print('Logreg: ', logreg_cv.score(X_test, Y_test))
print('SVM: ', svm_cv.score(X_test, Y_test))
print('Decision Tree: ', tree_cv.score(X_test, Y_test))
print('KNN accuracy: ', knn_cv.score(X_test, Y_test))

#The decision tree model performs best
```

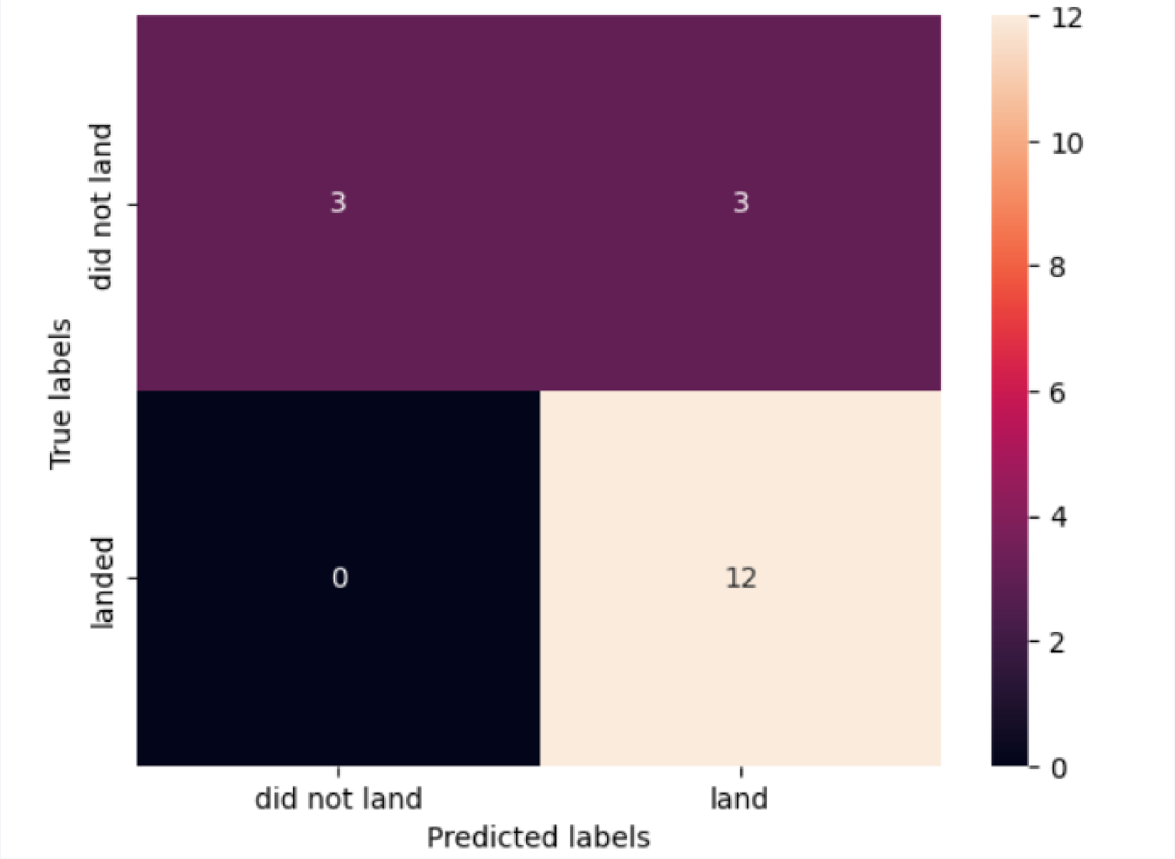
```
Model accuracy
Logreg:  0.8333333333333334
SVM:  0.8333333333333334
Decision Tree:  0.7222222222222222
KNN accuracy:  0.8333333333333334
```

Winners

Confusion Matrix

This is the confusion matrix of the logistic regression model, which is one of the three models that I identified as having the highest accuracy.

The rows indicate the true labels, and the columns indicate the predicted labels.



Logistic Regression confusion matrix

Conclusions

I'm able to predict successful SpaceX F9 landings with any of the four models generated: logreg, SVM, decision tree, KNN. However, Logreg, SVM, and KNN are clearly the winners, each with 0.83 accuracy.

```
•[74]: logreg_cv.best_params_

 [74]: {'algorithm': 'auto', 'n_neighbors': 9, 'p': 1}

 [76]: svm_cv.best_params_

 [76]: {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}

•[75]: tree_cv.best_params_

 [75]: {'criterion': 'gini',
        'max_depth': 18,
        'max_features': 'sqrt',
        'min_samples_leaf': 1,
        'min_samples_split': 10,
        'splitter': 'best'}

 [77]: knn_cv.best_params_

 [77]: {'algorithm': 'auto', 'n_neighbors': 9, 'p': 1}
```

The best hyperparameters for each of the four models:

Appendix

Github Links:

- Project Repository (all files)
    - F9 Data Collection W1
    - F9 Web Scraping W1
    - F9 Data Wrangling W1
    - F9 Exploring and Preparing Data W2
    - F9 SQL EDA W2
    - F9 Launch Sites Viz with Folium W3
    - F9 Plotly Interactive Dashboard W3
    - F9 Spacex ML Predictions W4-V2
    - dataset_part_1.csv
    - spacex_launch_geo_dataset_w3.csv
    - spacex_ml_dataset2_w4.csv
    - spacex_ml_dataset3_w4.csv
    - spacex_w2.csv

Thank you!