IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sujatro Majumdar
29 August 2024

Outline

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

Executive Summary

- Summary of methodologies

→ Data Collection

→ Data Wrangling

→ EDA with Data Visualisation and SQL

→ Folium: Interactive Map

→ Plotly Dash: Dashboard

→ Predictive Analysis

- Summary of all results

→ EDA Results

→ Interactive Analytics

→ Predictive Analysis

Introduction

- This project is the conclusion of a lengthy learning journey, the culmination of everything we've learned from IBM Data Science. It is now time to put our learned skills to the test.
- In this project, we will help SpaceY, our rival to SpaceX, make informed decisions on rocket launches based on historical knowledge and our expertise.

Premise: SpaceX advertises its Falcon 9 rocket at a cost of 62 million dollars. The market standard stands at 165 million dollars, the lower cost of SpaceX owing to the fact that it can reuse its first stage. Therefore by determining whether this first stage will land or not, we can determine the cost of a launch. This information can be used by a competitor seeking to bid against SpaceX (like us).

Our job is to predict if the Falcon 9 first stage will land successfully using historical data.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describes how data was collected
- Perform data wrangling
  - Describes how data was processed
- 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

- Data first collected using SpaceX API (a RESTful API) via request.

→ Done by first defining a series of helper functions using identification numbers in the launch data, then requesting rocket launch data from the relevant url.

→ To make the requested JSON results more consistent, launch data was requested and parsed using GET request. Response content was then decoded as a JSON, which was then finally converted into a Pandas data frame.

- Additionally performed webscraping to collect Falcon 9 historical launch records from Wikipedia, the launch records themselves stored as HTML. Using BeautifulSoup and request Libraries, the HTML table records were parsed and converted into a Pandas data frame

Data Collection – SpaceX API

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/datasets/API_call_spacex_api.json'
```

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

```
response.status_code
```

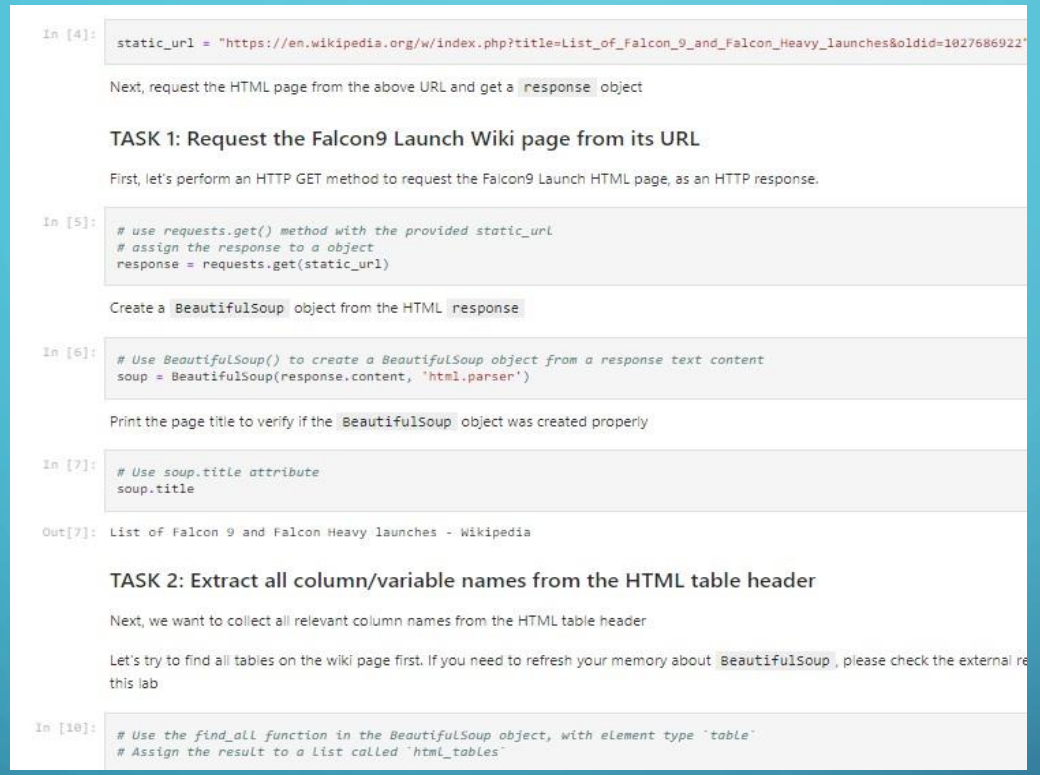
```
200
```

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

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

https://github.com/Majic-al/Capstone-DatSci/blob/main/1.Data%20Collection%20API.ipynb

Data Collection – SpaceX WebScraping



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

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.

In [5]:
```
# 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`

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, 'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

In [7]:
```
# Use soup.title attribute
soup.title
```

Out[7]:
```
List of Falcon 9 and Falcon Heavy launches - Wikipedia
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external re
this lab

In [10]:
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
```

- https://github.com/Majic-al/Capstone-DatSci/blob/main/2.Web%20scraping.ipynb

Data Wrangling

- Data was then filtered using BoosterVersion column to only keep Falcon 9 launches, and missing data values in the LandingPad and PayloadMass columns were cleared. For instance, with PayloadMass, missing data values were replaced using mean values.
- Also performed Exploratory Data Analysis (EDA) to find patterns in data and determine what would be the label for training supervised models
- https://github.com/Majic-al/Capstone-DatSci/blob/main/3.Data%20Wrangling.ipynb

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is variable `landing_class`:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
df['Class'] = df['Outcome'].apply(lambda x: 0 if x in bad_outcomes else 1)
df['Class'].value_counts()
```

```
1    60
0    30
Name: Class, dtype: int64
```

This variable will represent the classification variable that represents the outcome of each launch. If t
first stage landed Successfully

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

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

EDA with Data Visualization

→ Performed Data Analysis with Features using Pandas and Matplotlib

→ Preparing Features

→Used scatter plots, bar charts, and line plots to visualize relationship between several relevant variables.

→ https://github.com/Majic-al/Capstone-DatSci/blob/main/5.EDA%20DataVis.ipynb

EDA with SQL

- The following SQL queries were performed for EDA
- Display names of unique launch sites

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

- Display 5 instances where launch sites begin with the string 'CCA‘

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

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

EDA with SQL (1)

- List date when first successful landing outcome on ground pad was accomplished

```
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";
```

- 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 DISTINCT Booster_Version, Payload  FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS_KG__ >  4000 AND PAYLOAD_MASS_KG__ < 6000;
```

- List the total number of successful and failure mission outcomes

```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

- https://github.com/Majic-al/Capstone-DatSci/blob/main/4.EDA%20Using%20SQL.ipynb

Build an Interactive Map with Folium

- Created map to mark all launch sites, and created map objects to indicate success or failure of launches for each site.
- Created a launch set outcomes (failure=0, success=1).

- https://github.com/Majic-al/Capstone-DatSci/blob/main/6.Folium.ipynb

Build a Dashboard with Plotly Dash

Built an interactive dashboard application with Plotly dash by:

- Adding Launch Site Drop-down Input Component
- Adding callback function to render success pie-chart based on selected site dropdown
- Adding a Range Slider to Select Payload
- Adding callback function to render success-payload scatter plot

- https://github.com/Majic-al/Capstone-DatSci/blob/main/7.%20Ploty%20Dashboard.py

Predictive Analysis (Classification)

https://github.com/Majic-al/Capstone-DatSci/blob/main/8.%20ML%20Prediction.ipynb

After loading data as Pandas Dataframe, Exploratory Data Analysis was performed and Training Labels determined by:

- Creating a NumPy array from column Class in data by applying method to_numpy(), then assigned it to variable Y as outcome variable.
- Standardized feature dataset (x) by transforming it using preprocessing.StandardScaler() function from Sklearn.
- Data was split into training and testing sets using the function train_test_split from sklearn.model_selection with test_size parameter setto 0.2 and random_state to 2.

Predictive Analysis (1)

In order to find the best ML model/ method that would performs best using test data between SVM, Classification Trees, k nearest neighbors and Logistic Regression;

- First created object for each algorithm, then created a GridSearchCV object and assigned them a set of parameters (each model).

For each model under evaluation, GridsearchCV object was created with cv=10, then fit training data into GridSearch object, for each to Find best Hyperparameter.

- After fitting training set, output GridSearchCV object for each model, then displayed best parameters using data attribute best_params_ and accuracy on the validation data using data attribute best_score_.

Finally, using method score to calculate the accuracy on the test data for each model, and plotted confusion matrix for each using test and predicted outcomes.

Predictive Analysis (2)

Out[68]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

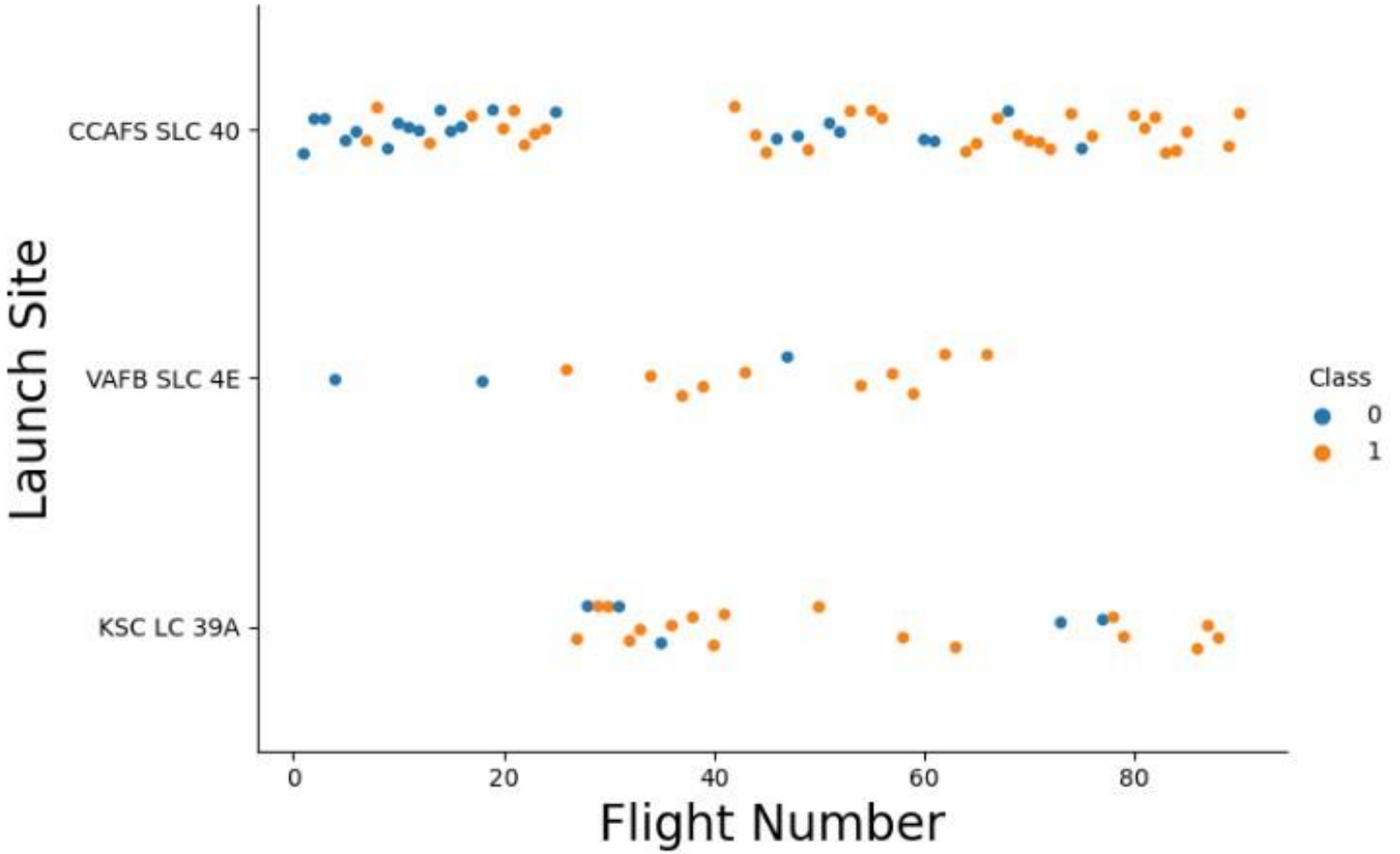
Results

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

Section 2

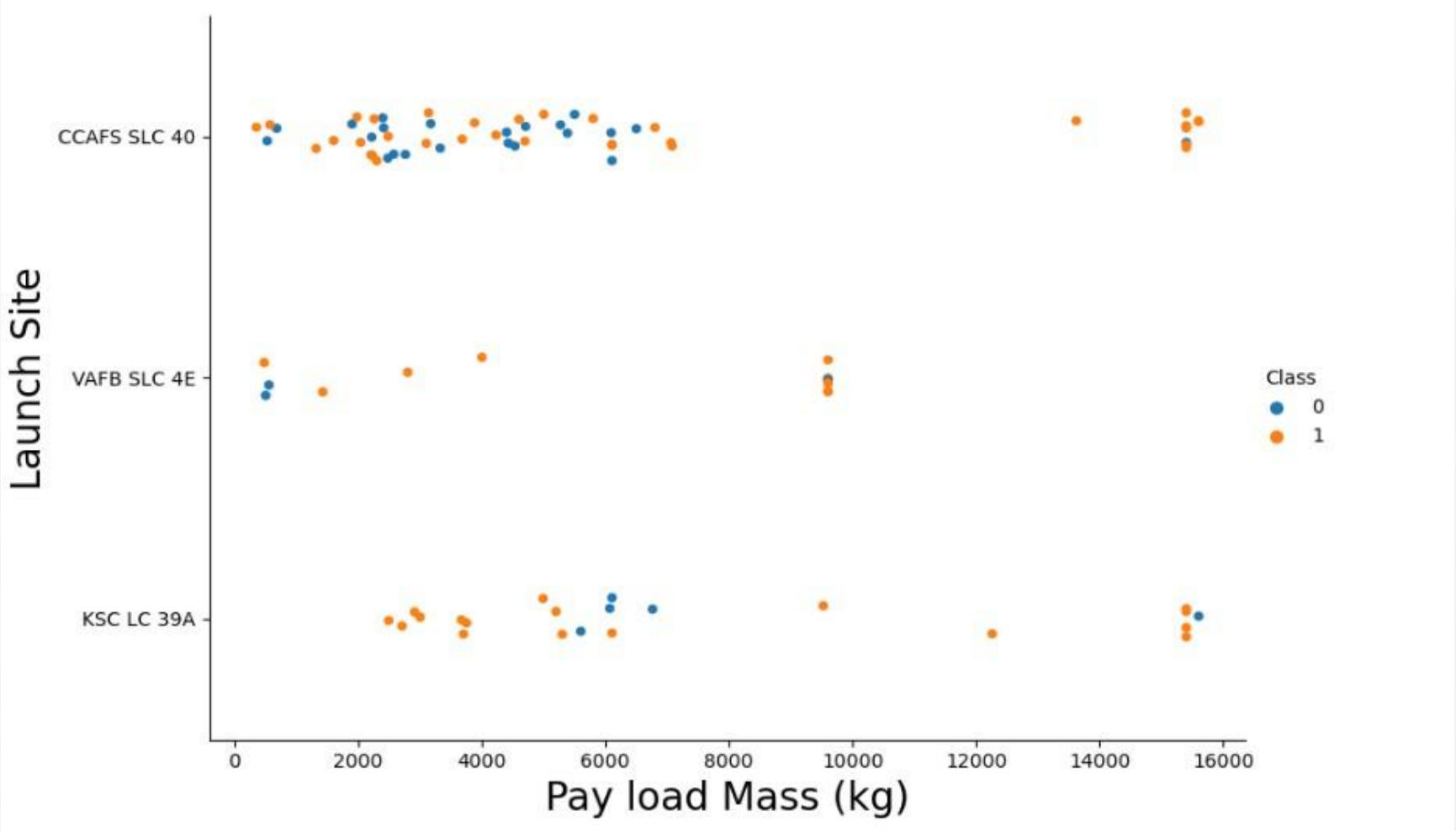
Insights drawn from EDA

Flight Number vs. Launch Site



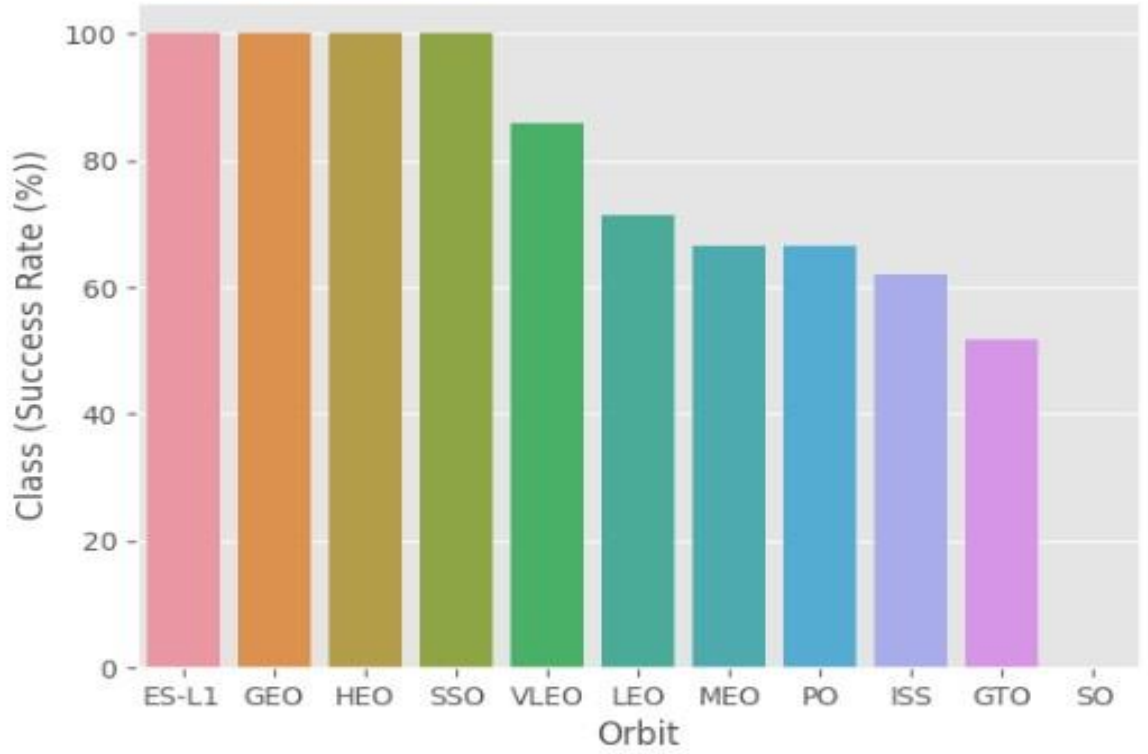
We can deduce that, as the flight number increases in each of the 3 launcg sites, so does the success rate. For instance, the success rate for the VAFB SLC 4E launch site is 100% after the Flight number 50. Both KSC LC 39A and CCAFS SLC 40 have a 100% success rates after 80th flight.

Payload vs. Launch Site



Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

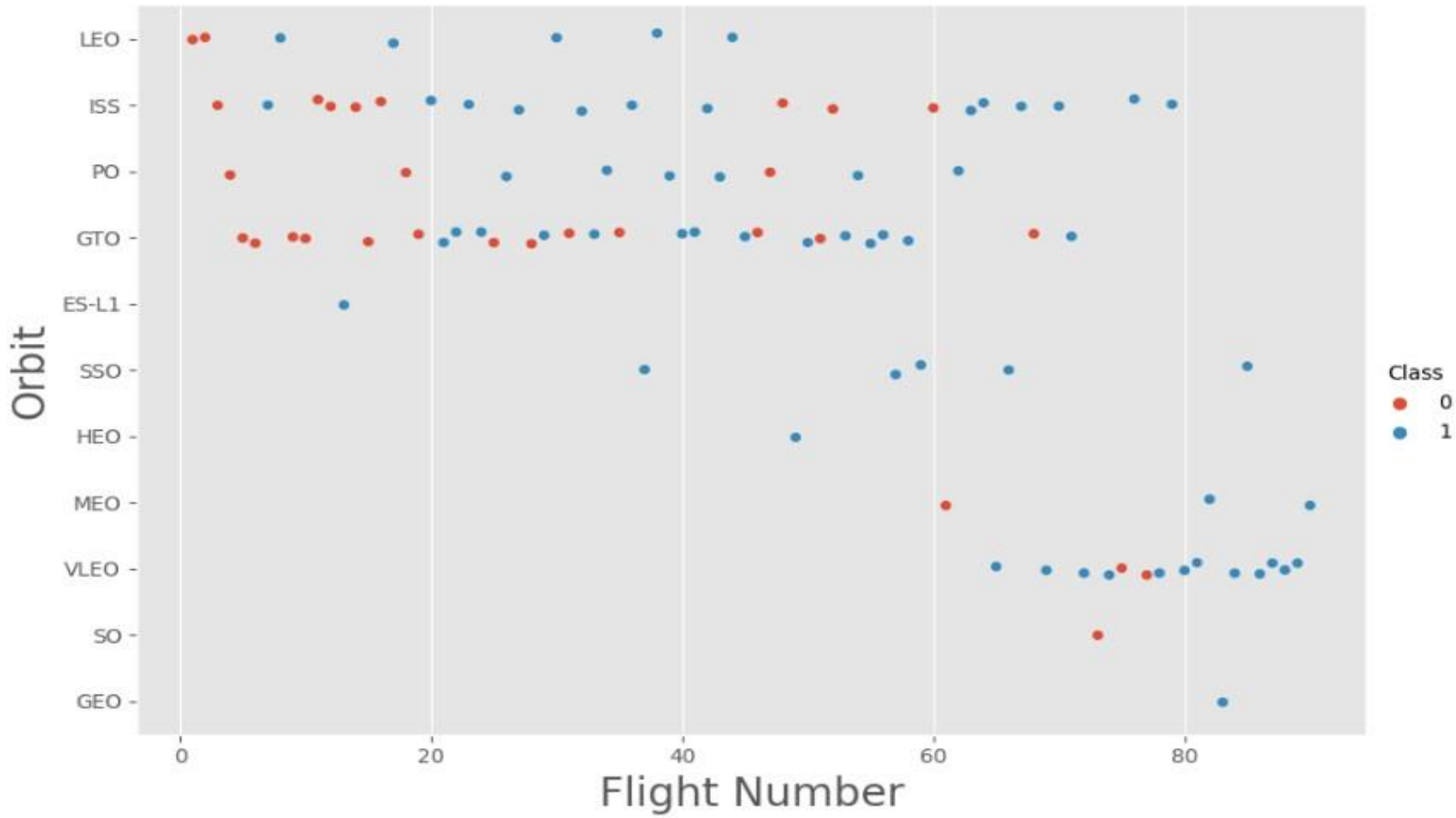
Success Rate vs. Orbit Type



Analyze the ploted bar chart try to find which orbits have high sucess rate.

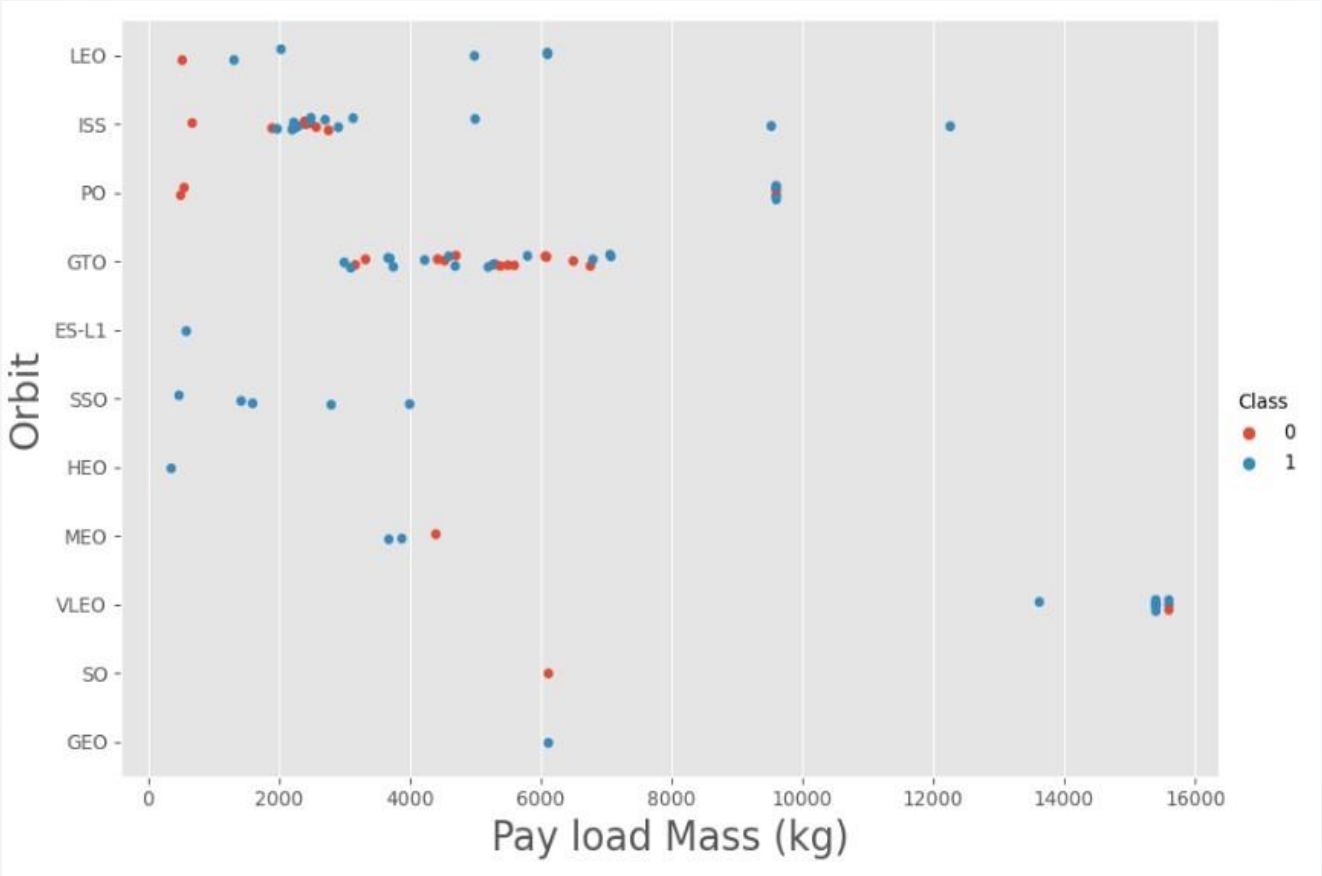
Orbits ES-L1, GEO, HEO & SSO have the highest success rates at 100%, with SO orbit having the lowest success rate at ~50%. Orbit SO has 0% success rate.

Flight Number vs. Orbit Type



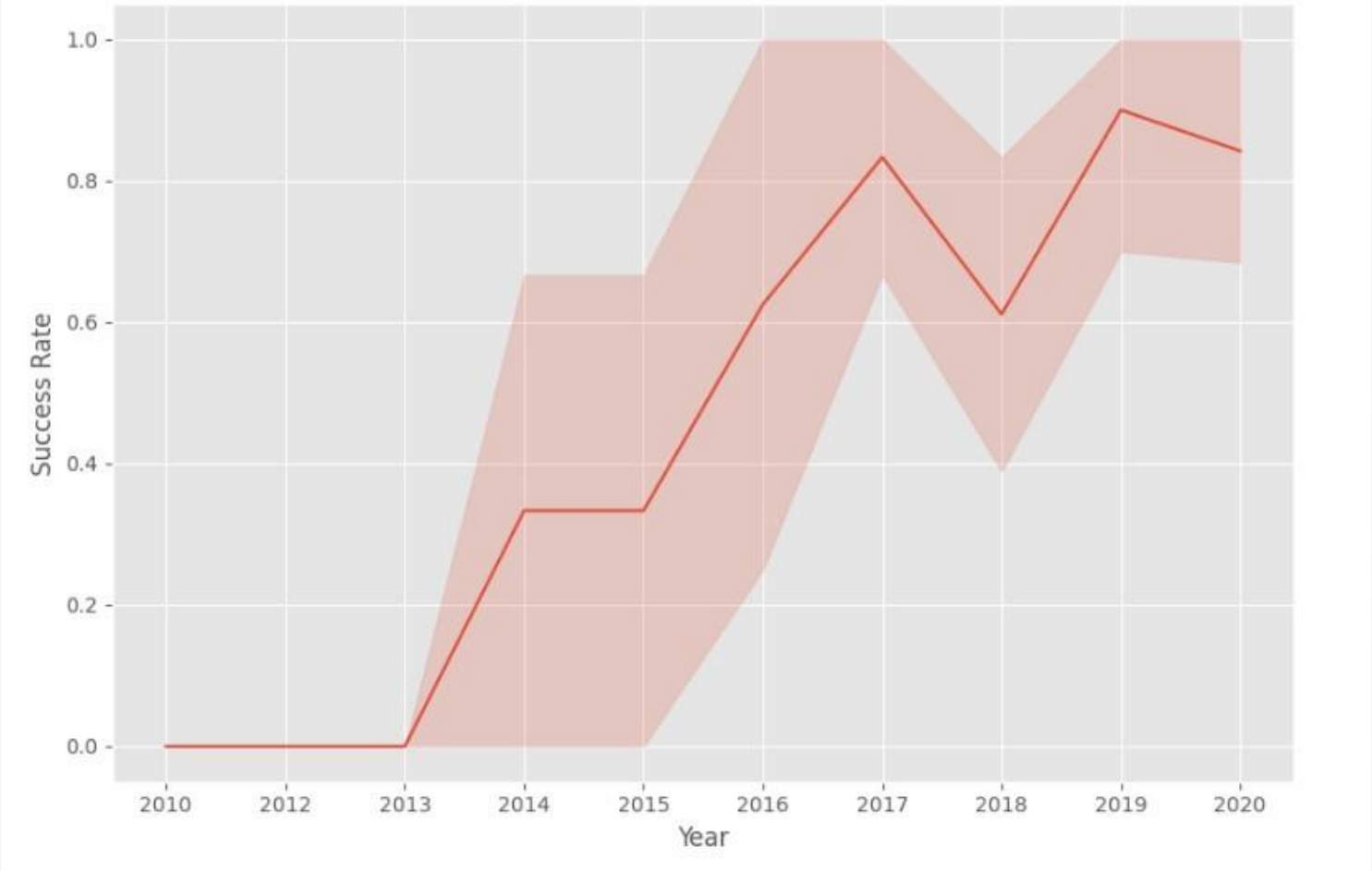
You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type



- POLAR, LEO, ISS more success landing at higher payloads, GTO inconclusive

Launch Success Yearly Trend



- Rising success rate 2013-2020, with 2018 an outlier

All Launch Site Names

Task 1

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

In [31]:
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

Out[31]:

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

- Used 'distinct' query (SQL) to only have unique sites returned

Launch Site Names Begin with 'CCA'

Task 2

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

In [72]:
```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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

Out[72]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- Used 'like' query (SQL) to return all entries with string %CAA returned

Total Payload Mass

Task 3

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

In [17]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

Out[17]:

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

- Used ‘sum’ query (SQL) to add total mass, also specified customer has to be NASA (CRS) using ‘where’

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 "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

- Used ‘avg’ query (SQL) to average mass, also specified booster has to be F9v1.1 using ‘where’ and ‘like’ to identify string ‘F9 v1.1’

First Successful Ground Landing Date

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 "Landing _Outcome" = "Success (ground pad)";
```

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

MIN(DATE)
01-05-2017

- Used ‘min’ query (SQL) to find lowest returned value, adding in ‘where’ “Success (ground pad)” is landing outcome

Successful Drone Ship Landing with Payload between 4000 and 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 * FROM 'SPACEXTBL'
```

```
%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOA
```

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

Booster_Version	Payload
F9 FT B1022	JCSAT-14
F9 FT B1026	JCSAT-16
F9 FT B1021.2	SES-10
F9 FT B1031.2	SES-11 / EchoStar 105

- Used ‘distinct’ query (SQL) to isolate booster version and payload combination ‘where’ (SQL) landing outcome successful on drone ship and mass within specified boundaries

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

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

- Used ‘count’ and ‘group by’ statements (SQL) to return each outcome categorized

Boosters Carried Maximum Payload

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

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

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

- Used ‘max’ in subquery to isolate all booster types that have carried maximum payload 15600kg

2015 Launch Records

Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

```
%sql SELECT substr(Date,7,4), substr(Date, 4, 2),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing _Outcome"
```

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

substr(Date,7,4)	substr(Date, 4, 2)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	Landing _Outcome
2015	01	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	Success	Failure (drone ship)
2015	04	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	Success	Failure (drone ship)

- Used the 'subsrt()' in select statement to get month and year from date column for year 2015 and included the categories seen above as columns to be on display in 'select' query

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

Task 10

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDER BY Date DESC;
```

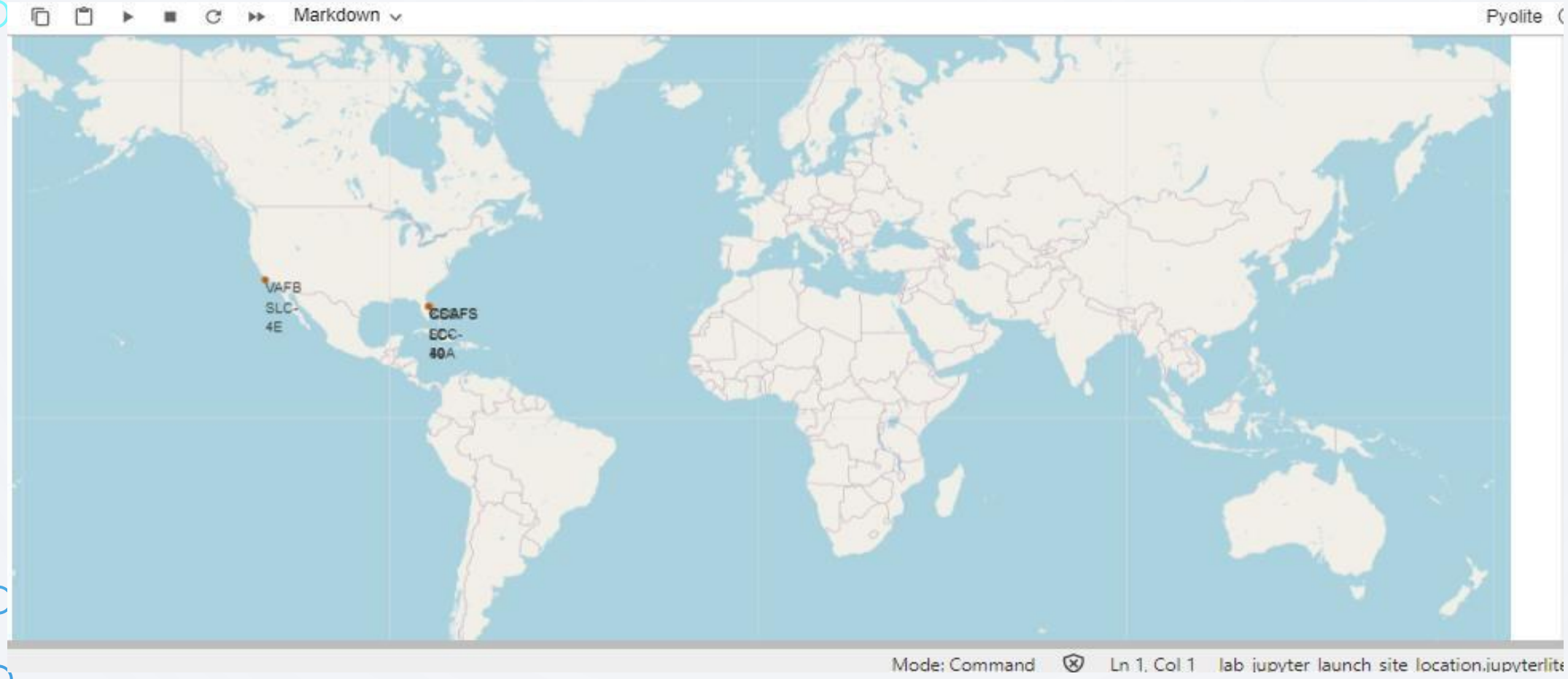
```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
19-02-2017	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-10-2020	12:25:57	F9 B5 B1051.6	KSC LC-39A	Starlink 13 v1.0, Starlink 14 v1.0	15600	LEO	SpaceX	Success	Success
18-08-2020	14:31:00	F9 B5 B1049.6	CCAFS SLC-40	Starlink 10 v1.0, SkySat-19, -20, -21, SAOCOM 1B	15440	LEO	SpaceX, Planet Labs, PlanetIQ	Success	Success
18-07-2016	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-04-2018	22:51:00	F9 B4 B1045.1	CCAFS SLC-40	Transiting Exoplanet Survey Satellite (TESS)	362	HEO	NASA (LSP)	Success	Success (drone ship)

Section 3

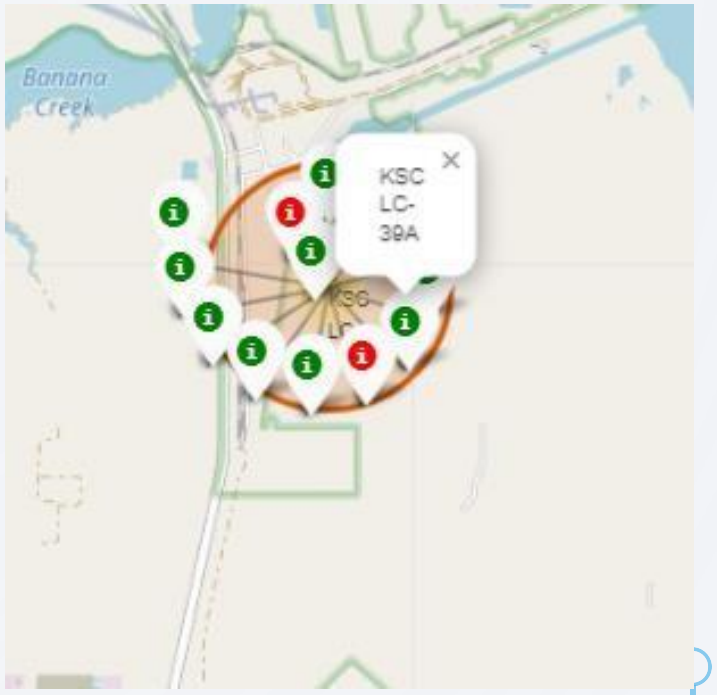
Launch Sites Proximities Analysis

Coastal/Equitorial Inclination of SpaceX Launch Sites



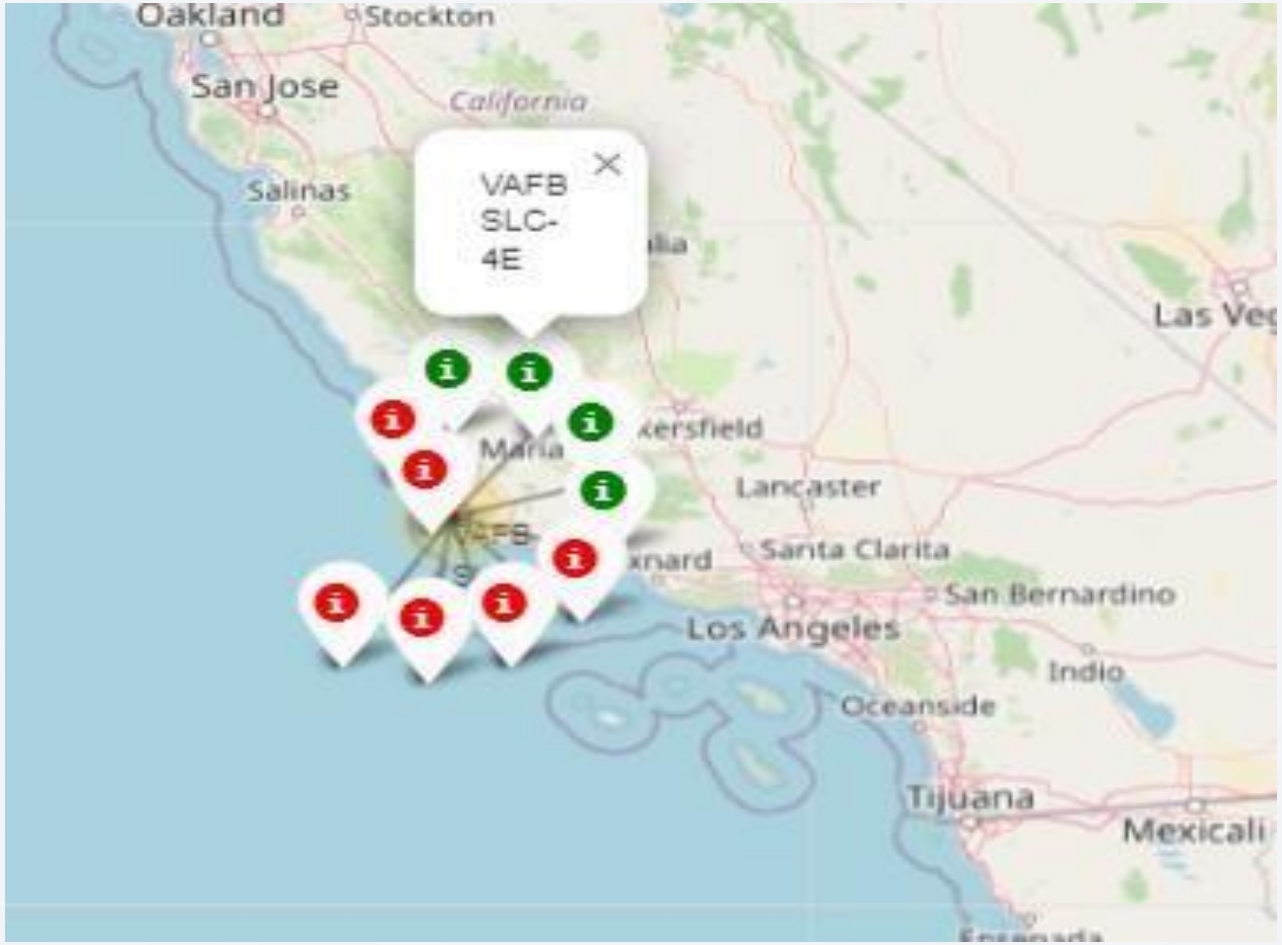
- All sites in proximity to coast, as well as nearer to the equator (south of U.S)

Launch Outcomes (Florida sites)



- Site KSCLC-39A has higher relative success rate compared to the two CCAFS sites

Launch Outcomes (California site)

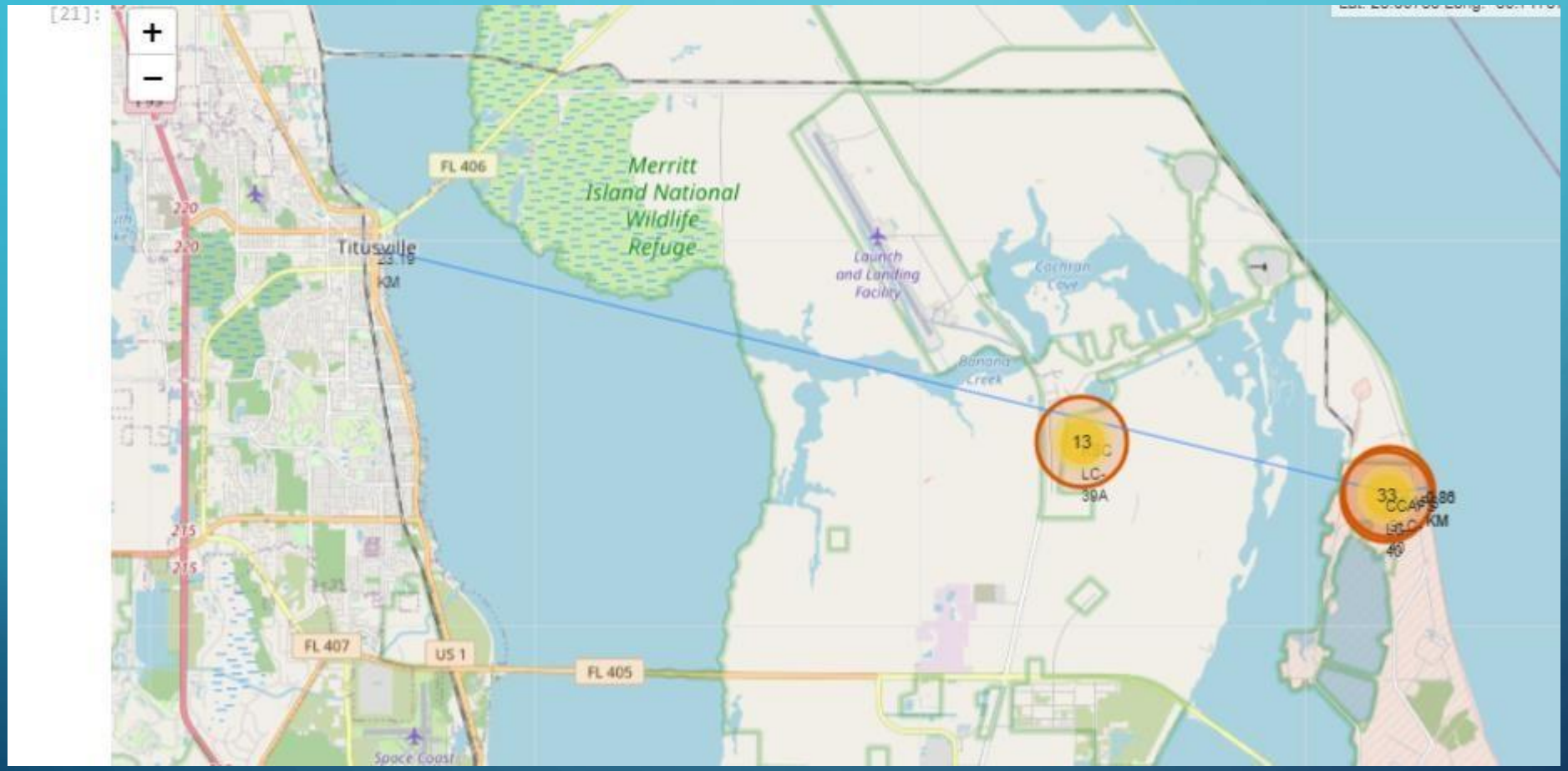


- Similar success rate as CCAFS sites, but lower than KSCLC

DISTANCE BETWEEN LAUNCH SITE AND COAST

- Launch site CCAFS SLC-40 proximity to coastline is 0.86km (closest of surveyed examples)

DISTANCE BETWEEN LAUNCH SITE AND HIGHWAY

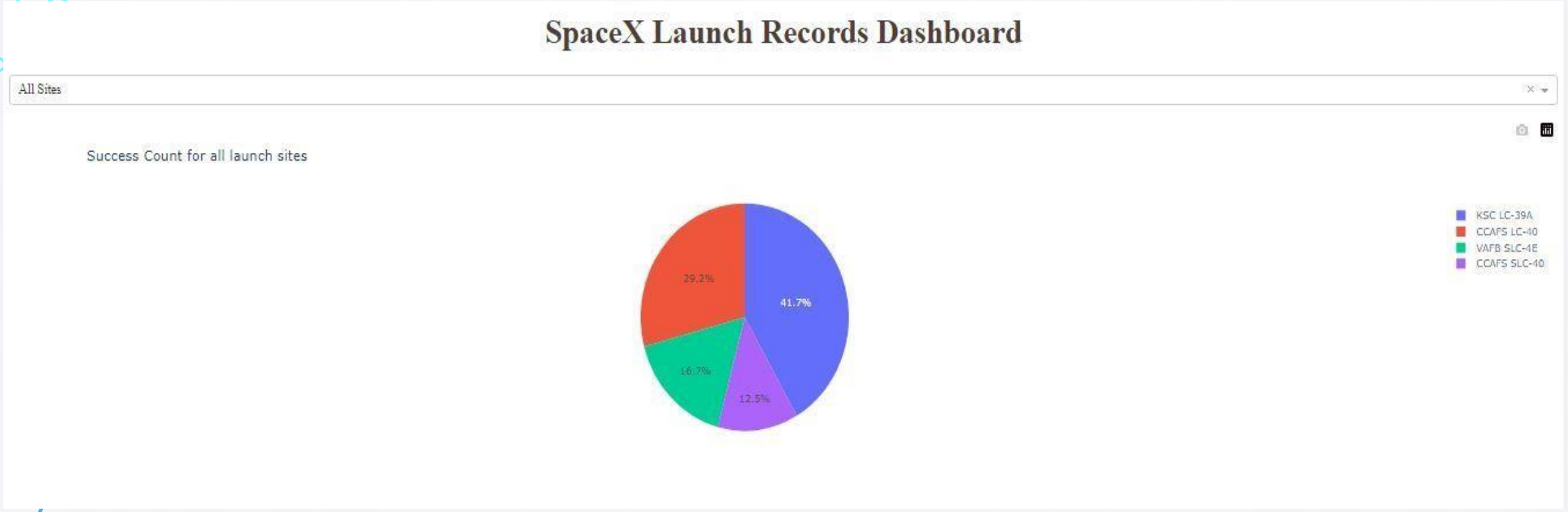


- Launch site CCAFS SLC-40 closest to highway, at 23.19km

Section 4

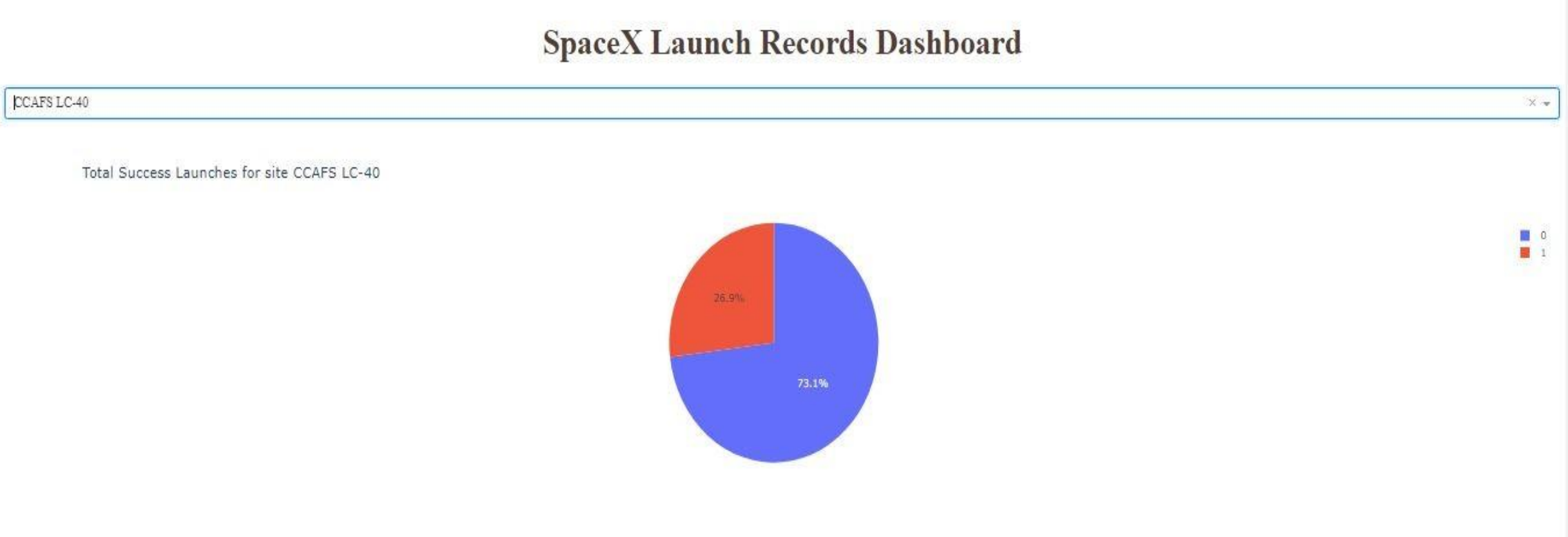
Build a Dashboard with Plotly Dash

All Sites' Success Rate as part of total



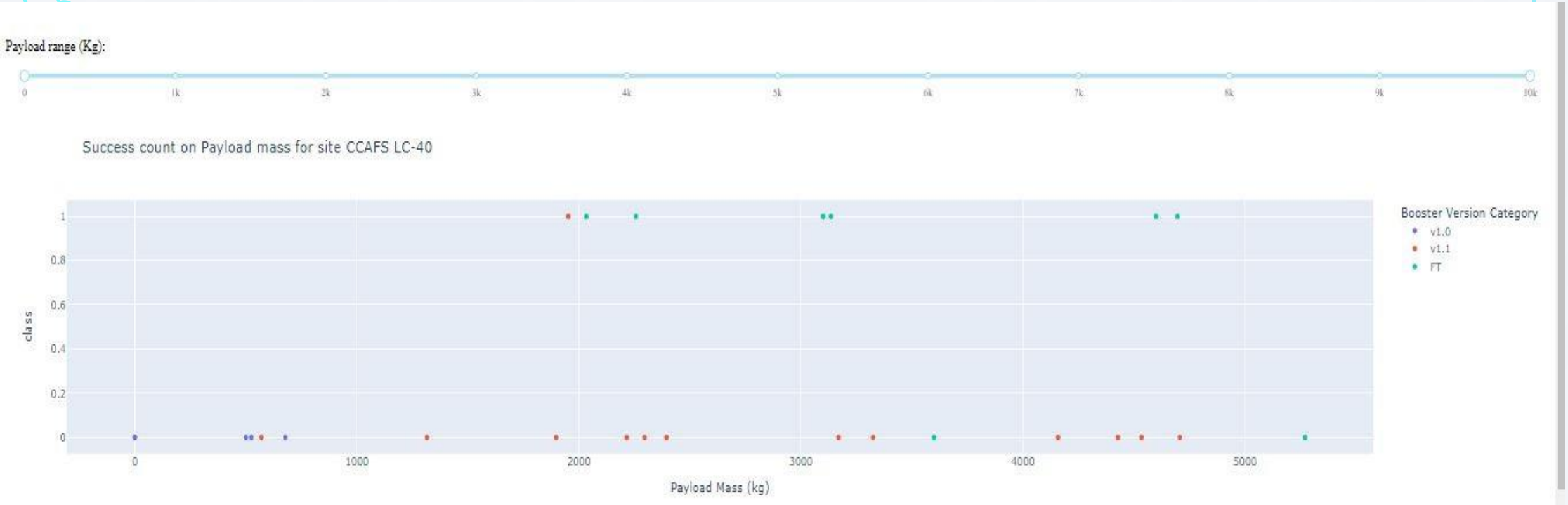
- KSC highest, followed by CCA (LC), then VAFB (California), and lastly CCA (SLC)

Success Ratio



- Above is the rate of success of CCAPS LC-40, second highest ratio of successes, as demonstration of ratio displays

Payload vs Launch Outcomes



- Using CCAFS LC-40 as example, we can see how payload measures against success rate, which in this instance indicates rising success rate above payload 2000kg

Section 5

Predictive Analysis (Classification)

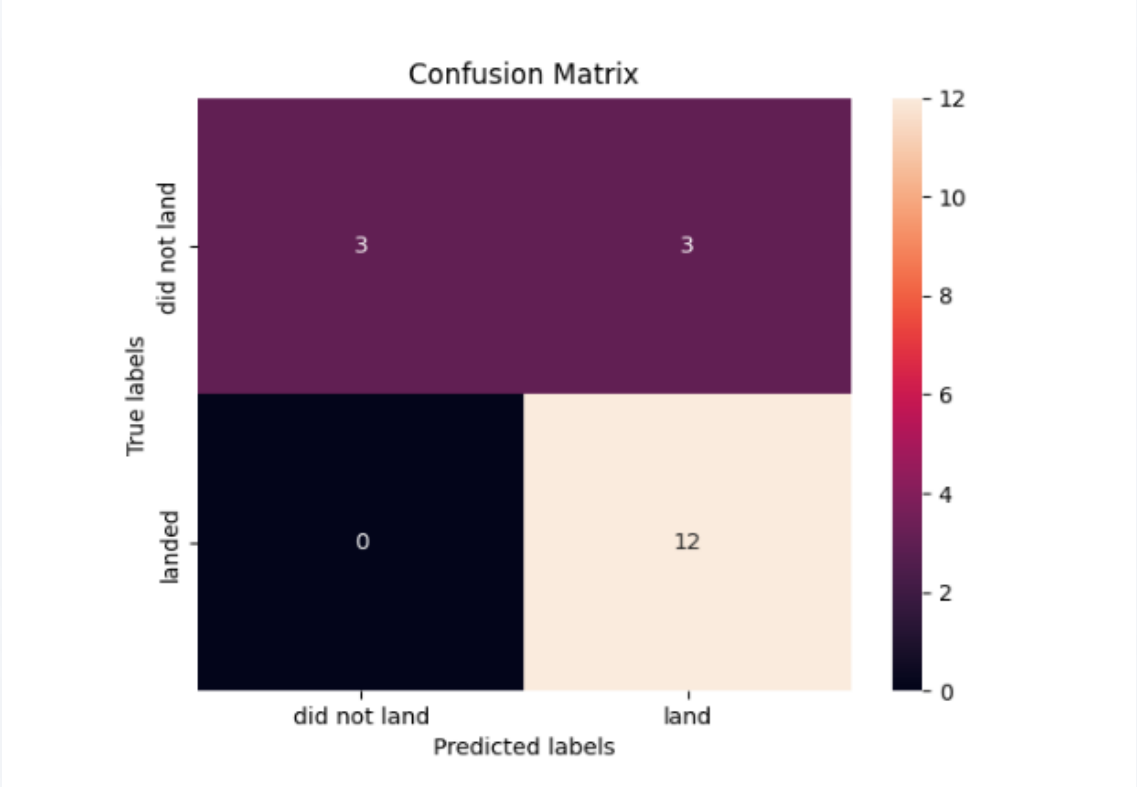
Classification Accuracy

Out[68]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

All the methods perform equally on the test data: i.e. They all have the same accuracy of 0.833333 on the test Data

Confusion Matrix



- All four models had similar matrices, being able to differentiate classes. False positives a problem to be considered in all models

Conclusions

Different launch sites exhibit varying success rates. CCAFS LC-40 has a 60% success rate, while KSC LC-39A and VAFB SLC-4E have rates of 77%. As the number of flights increases at each site, the success rate also tends to improve. For example, VAFB SLC-4E has a 100% success rate after the 50th flight, while both KSC LC-39A and CCAFS LC-40 achieve 100% success after the 80th flight.

The Payload vs. Launch Site scatter chart shows that VAFB SLC-4E has not launched rockets with heavy payloads (over 10,000 kg). Orbits ES-L1, GEO, HEO, and SSO all have a 100% success rate, whereas orbit SO has the lowest success rate at around 50% and a 0% success rate in some cases.

For LEO orbit, success rates appear to correlate with the number of flights, but there is no such correlation for GTO orbit.

Thank you!