IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

James Boise
June 27, 2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection Using SpaceX APIs
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis Using SQL
  - EDA DataViz Using Python Pandas and Matplotlib
  - Launch Sites Analysis with Folium Interactive Visual Analytics and Plotly Dash
  - Machine Learning Landing Prediction
- Summary of all results
  - EDA Results
  - Interactive Visual Analytics and Dashboards
  - PredictAnalysis (Classification)

Introduction

- Project background and context
  - SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch.
- Problems you want to find answers
  - Can we determine if the first stage of the Falcon 9 rocket will land successfully using data from Falcon 9 launches advertised on the SpaceX website?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - API and Webscraping
- Perform data wrangling
  - Using Pandas dataframe
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Machine Learning, confusion matrix

Data Collection

- Description of how data sets were collected.
  - Data was collected by making a request to the SpaceX API. This was done by defining a series of helper functions that would help to extract information using identification numbers in the launch data. The request was then made to the SpaceX API in order to extract past launch data.
    - In order to make the requested JSON results more consistent and readable, the data was converted into a Pandas data frame.
  - Additionally, web scraping was performed on the Falcoln 9 Launch Wikipedia page using an HTTP GET method and BeautifulSoup object.
    - The extracted Falcon 9 Launch HTML table from Wikipedia was then parsed and converted into a Pandas data frame.

Data Collection – SpaceX API

- Data collected from the SpaceX API by making a GET request to the SpaceX API. The JSON response content was decoded using .json() and turned into a Pandas data frame using .json_normalize().

- The GitHub URL of the completed SpaceX API calls notebook https://github.com/jboise9/SpaceY/blob/main/Lab%201%20Collecting%20the%20Data.ipynb

Task 1: Request and parse the SpaceX launch data using the GET request

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

In [9]:

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

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

In [10]:

```
response.status_code
```

Out[10]:

```
200
```

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

In [16]:

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

Data Collection - Scraping

- First, an HTTP GET method was used to request the Falcoln9 Luanch HTML as an HTTP response. Then, a BeautifulSoup object was created from the HTML response.

- The GitHub URL https://github.com/jboise9/SpaceY/blob/main/Web%20scraping%20Falcon%209%20and%20Falcon%20Heavy%20Launch%20Record%20from%20Wikipedia.ipynb

In [4]:

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_
```

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]:

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

Data Wrangling

- After creating a Pandas DF from the collected data, data was filtered to only keep the Falcon9 launches using the *BoosterVersion* column. Missing data values in the *LandingPad* and *PayloadMass* by calculating the mean value of the column
- GitHub URL https://github.com/jboise9/SpaceY/blob/main/Data%20Wrangling.ipynb

TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

In [12]:

```
# 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()
```

Out[12]:

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

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

In [16]:

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

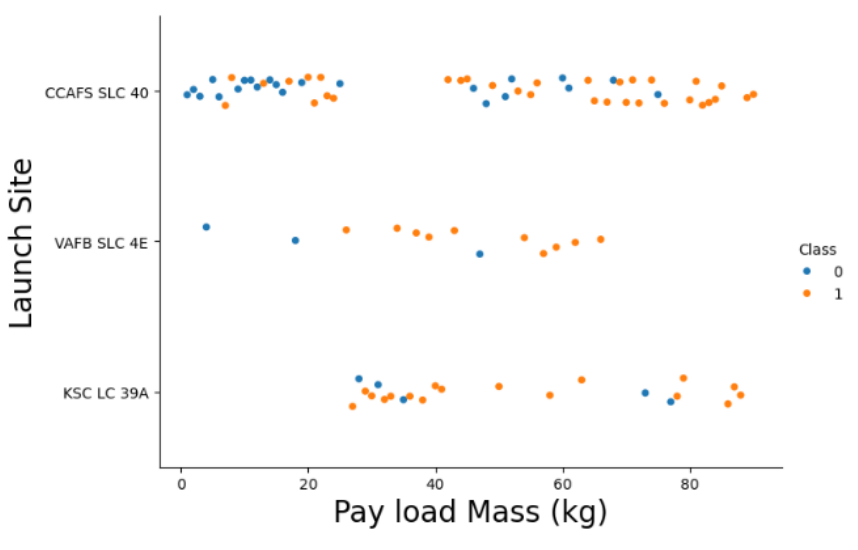
Out[16]:

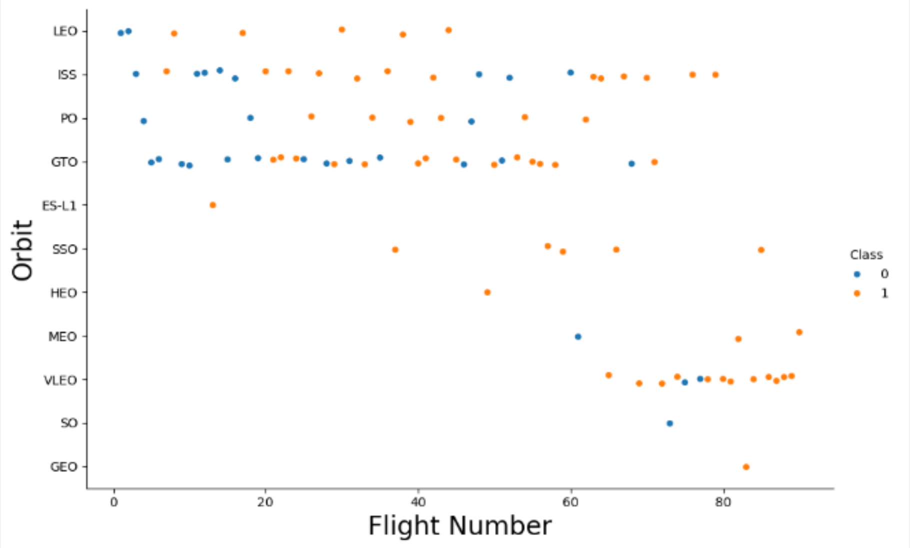
	Class
0	0
1	0
2	0

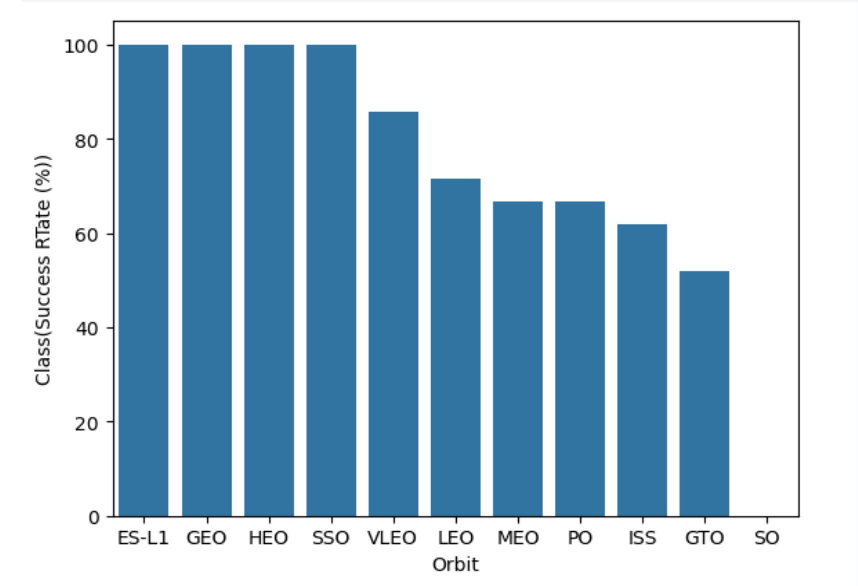
EDA with Data Visualization

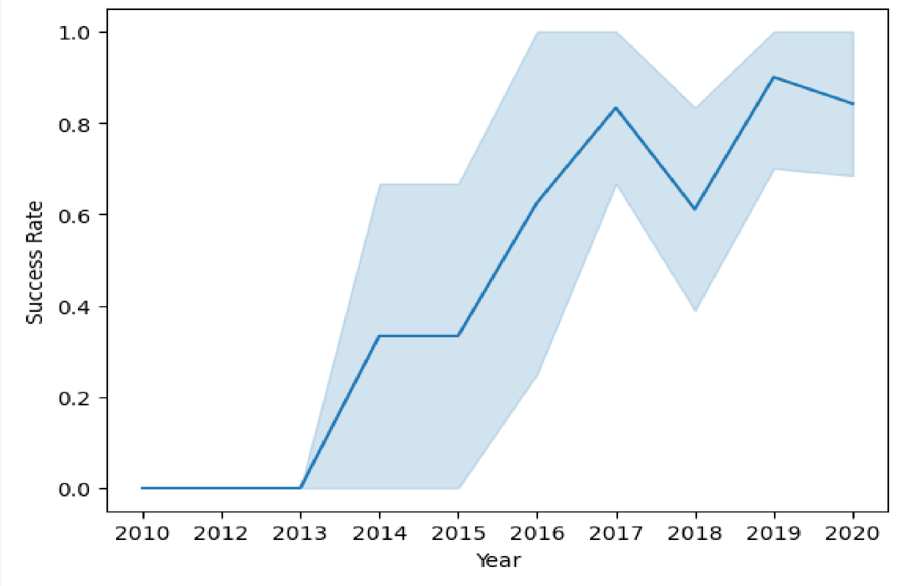
- Performed exploratory Data Analysis and Feature Engineering using Pandas and Matplotlib
    - Exploratory Data Analysis
    - Preparing Data Feature Engineering
- Used scatter plots to chart the relationship between *FlightNumber* and *LaunchSite*, *Pay Load Mass (kg)* and *LaunchSite*, *FlightNumber* and *Orbit*, as well as *Pay load Mass (kg)* and *Orbit*
- Created a bar chart to display *Orbit* and *Class (Success Rate (%))*
- Created a line chart to display *Year* and *Success Rate*
- GitHub URL
  https://github.com/jboise9/SpaceY/blob/main/EDA%20with%20Visualization%20Lab.ipynb

EDA with Data Visualization (continued…)









EDA with SQL

- The following SQL queries were performed for EDA
  - Display the names of the unique launch sites in the space mission

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

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

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

  - Display the 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
```

  - List the date when the first successful landing outcome in ground pad was achieved.

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

EDA with SQL (continued...)

- The following were performed for EDA
  - 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_M
```

  - 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";
```

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

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

  - 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_Ou
```

EDA with SQL (continued...)

- The following were performed for EDA
  - 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 * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDE
```

- GitHub URL
  https://github.com/jboise9/SpaceY/blob/main/EDA%20Using%20SQL.ipynb

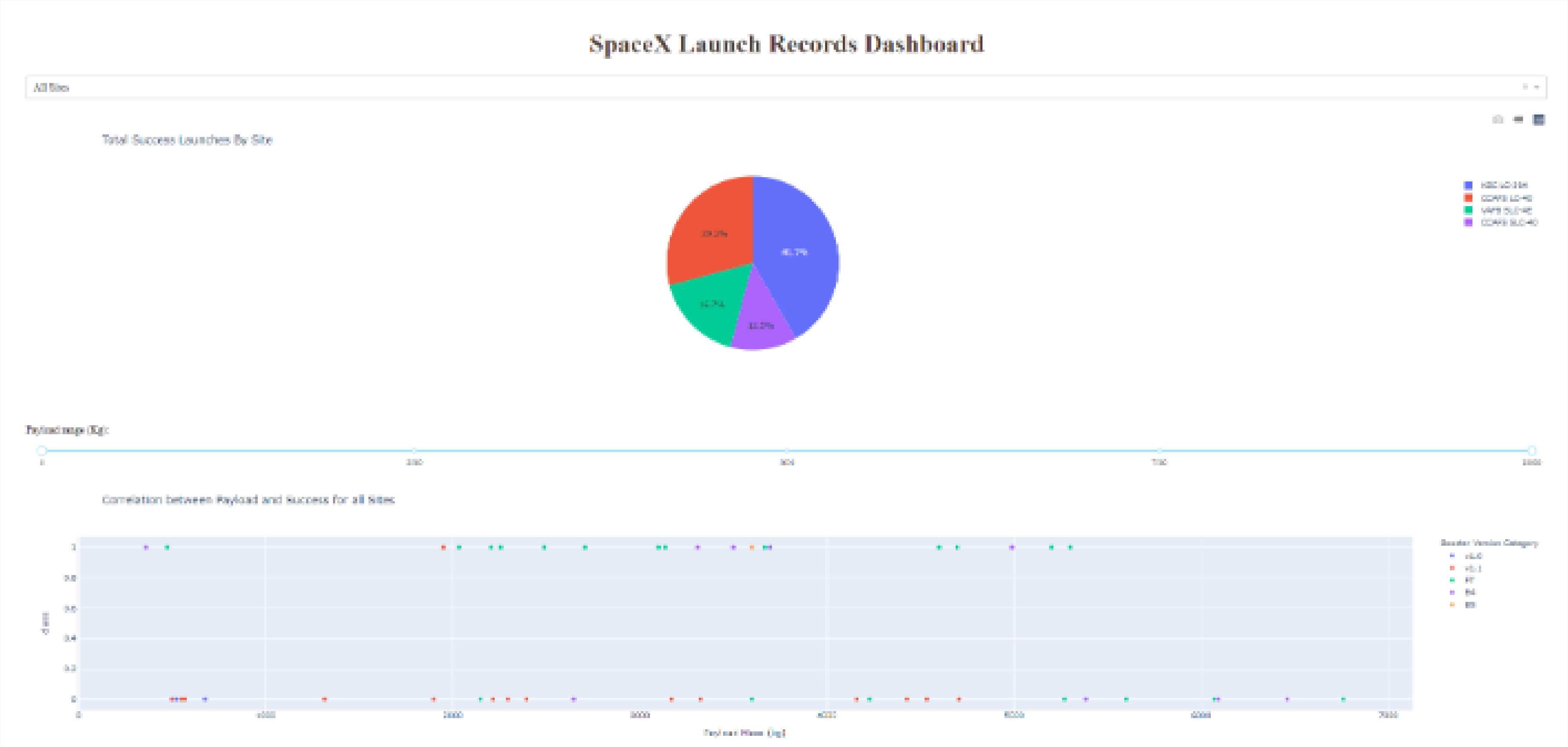
Build an Interactive Map with Folium

- Created a folium map to mark all of the launch sites.
- Created map objects such as markers, circles, lines to mark the success or failure of launches for each launch site.
- Created a launch set outcomes (fail;ure=0, success=1).
- GitHub URL https://github.com/jboise9/SpaceY/blob/main/Interactive%20Visual%20Analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- Built an interactive dahsboard with Plotly Dash by:
  - Adding a Lauch Site drop-down input component
  - Adding a callback function to render success-pie-chart based on select site dropdown
  - Adding a range slider to select payload
  - Adding a callback function to render the success-payload scatter plot
- GitHub URL https://github.com/jboise9/SpaceY/blob/main/dash_interactive

Build a Dashboard with Plotly Dash



Predictive Analysis (Classification)

- The objectives were to perform exploratory data analysis and determine training labels by creating a column for class, standardizing the data, and splitting the data into training and testing segments
  - After loading the data frame, a Numpy array, Y, was created by applying the *to_numpy()* function to column *Class*
  - The data was then standardized and reassigned to X using *.standardscaler* and *.fit_transform*
  - Next, the data was split into training and testing data using *train_test_split'*
  - A logistic regression object and a GridSearchCV object *logreg_cv* were then created with cv=10. After fitting the training set, we output the GridSearchCV object for each of the models, then displayed the best parameters.
  - Finally, using the method score to calculate the accuracy of the test data for each model, a confusion matrix was plotted for each using the test and predicted outcomes.

Results

- The table shows the test data accuracy score for each of the methods and compares them to show which performed best using the test data between Logistic Regression, SVM, Decision Tree, and KNN.

Out[63]:

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

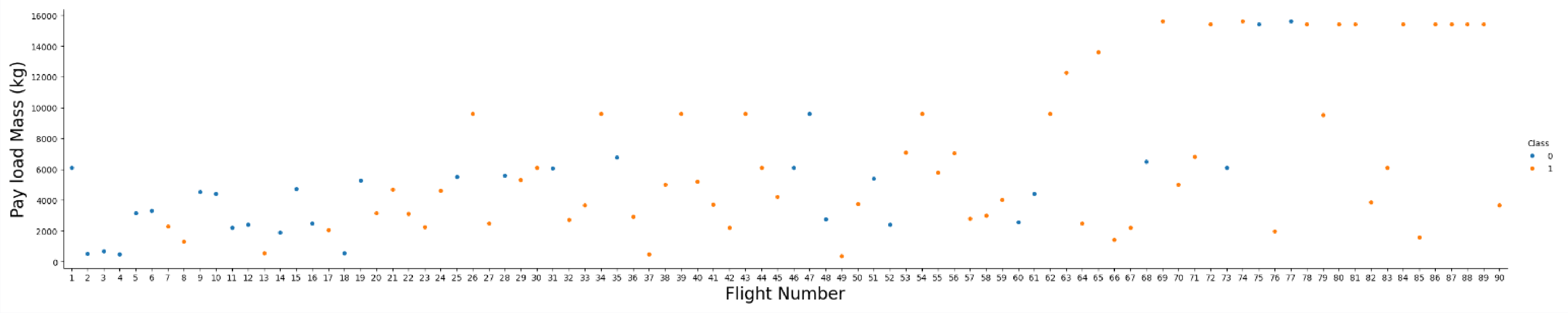
- GitHub URL https://github.com/jboise9/SpaceY/blob/main/Machine%20Learning%20Prediction.ipynb

Section 2

Insights drawn from EDA

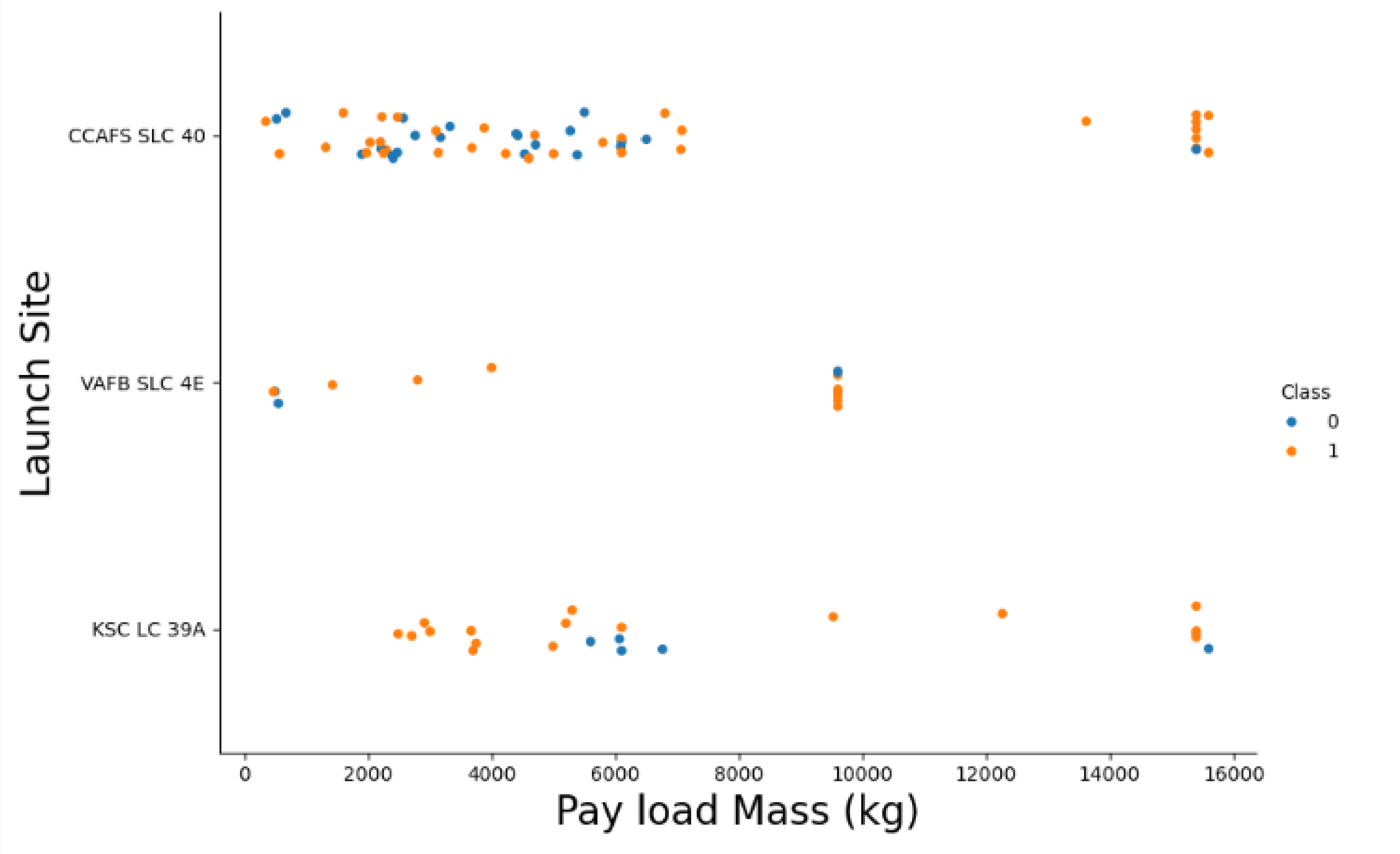
Flight Number vs. Launch Site

Scatter plot of Flight Number vs. Launch Site



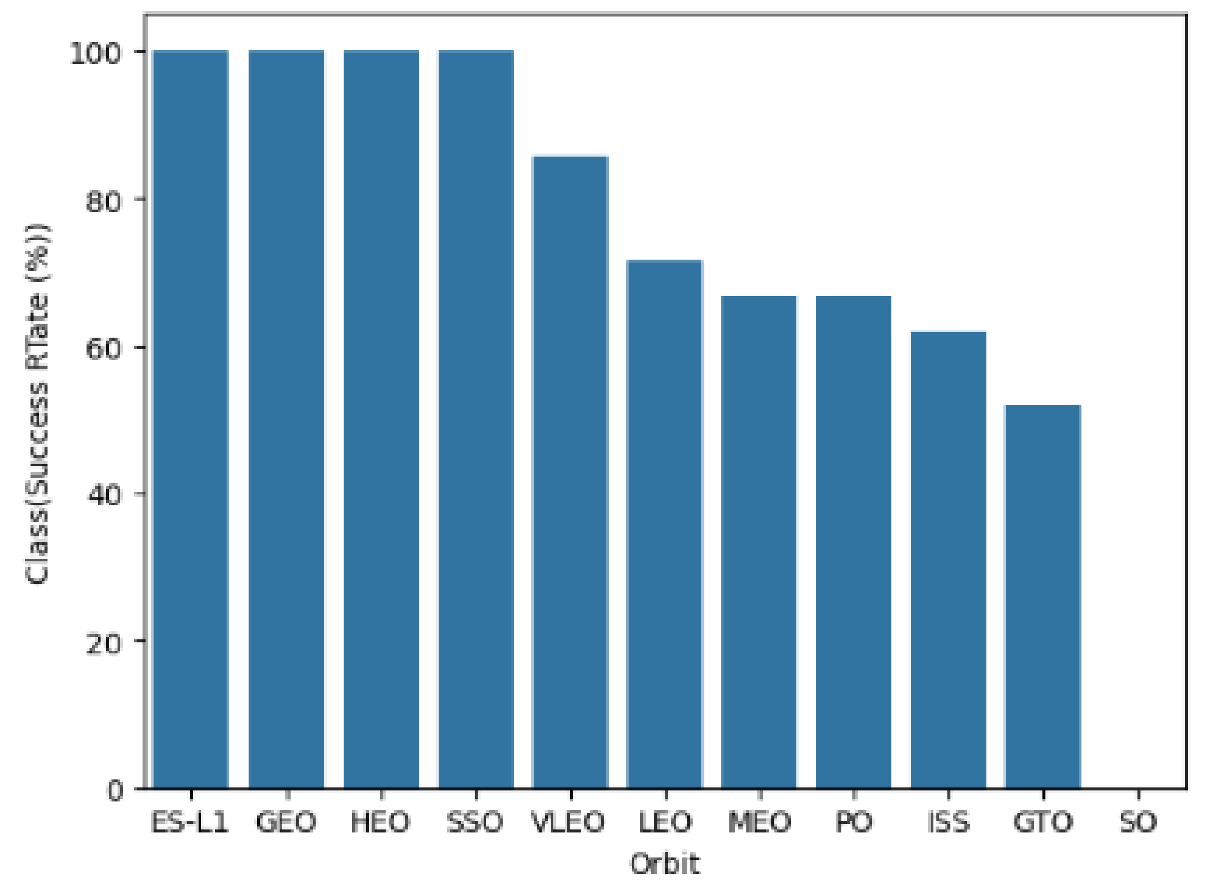
Payload vs. Launch Site

Scatter plot of Payload vs. Launch Site



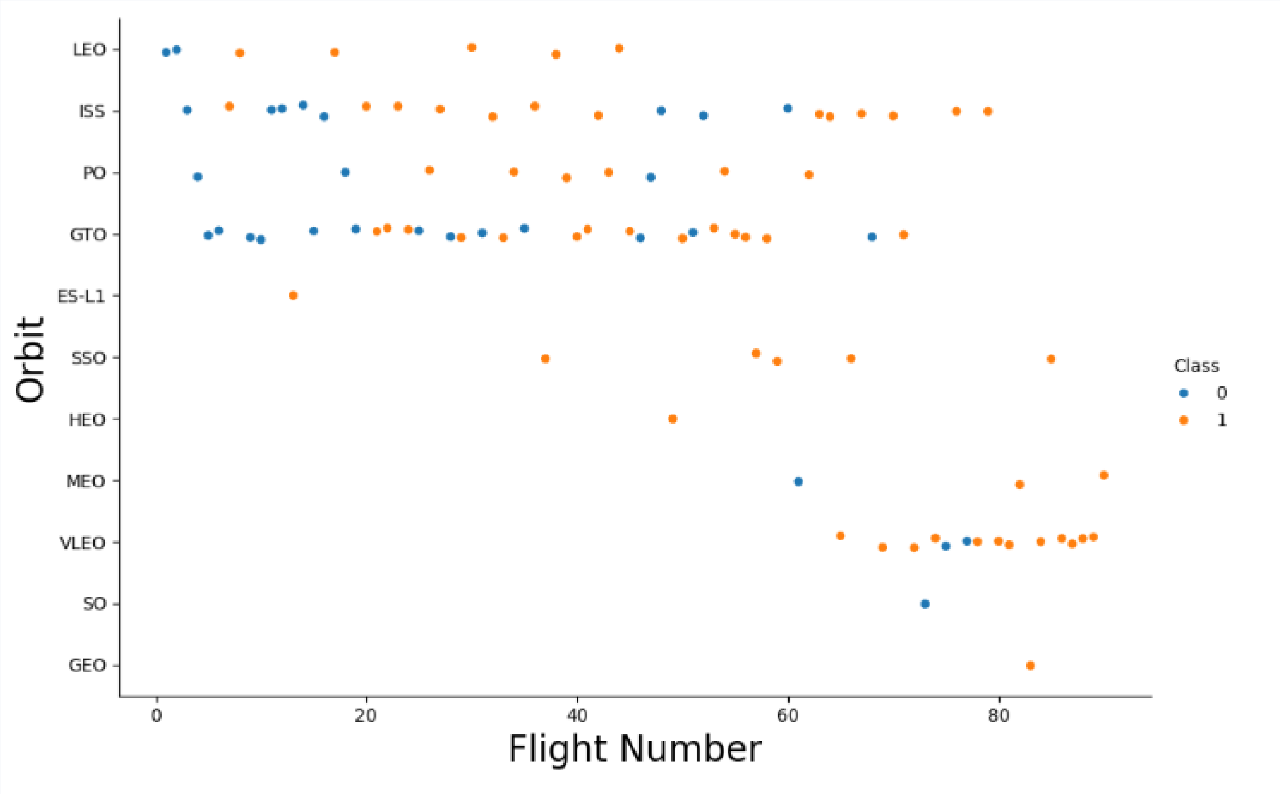
Success Rate vs. Orbit Type

Bar chart for the success rate of each orbit type



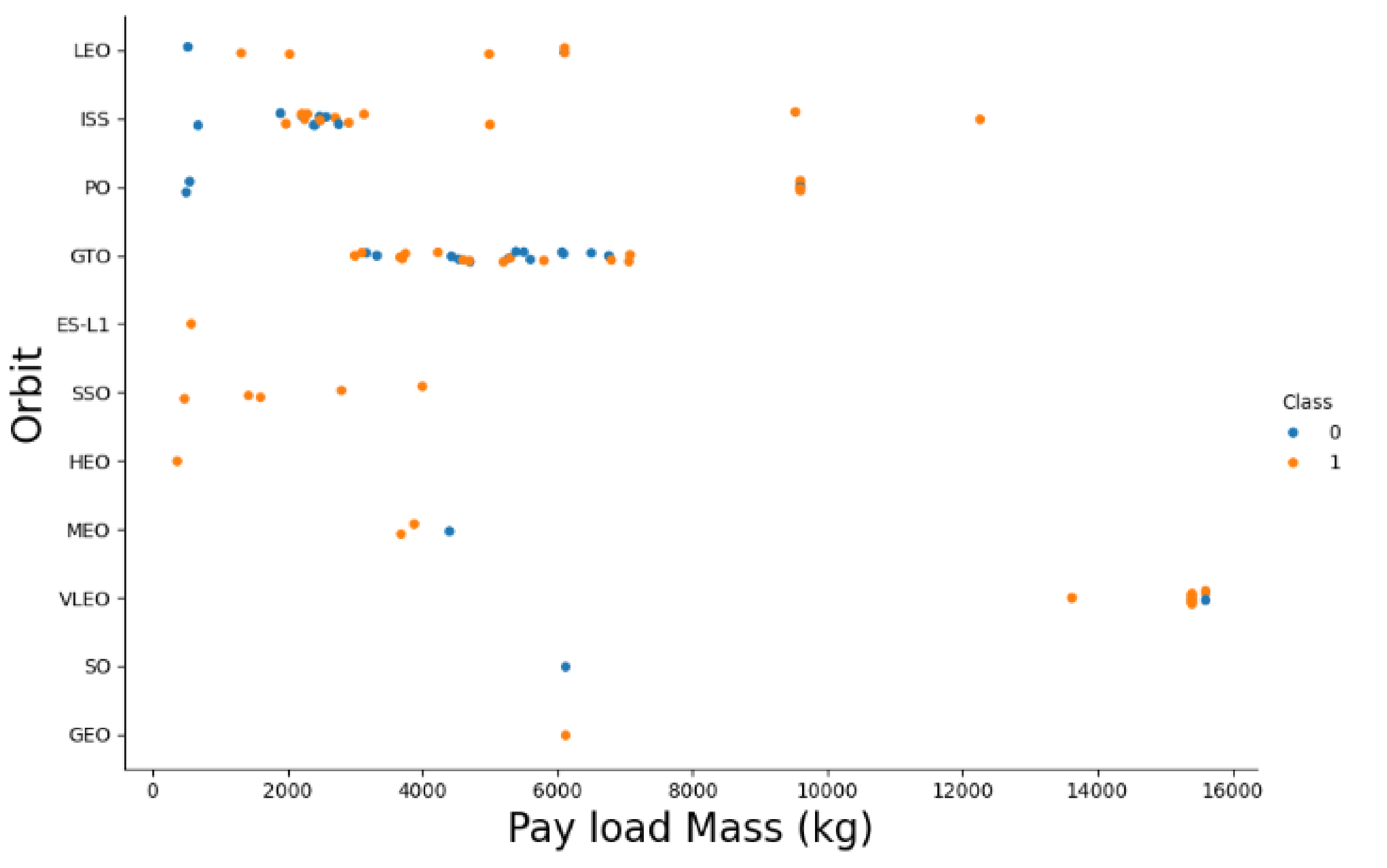
Flight Number vs. Orbit Type

Scatter point of Flight number vs. Orbit type



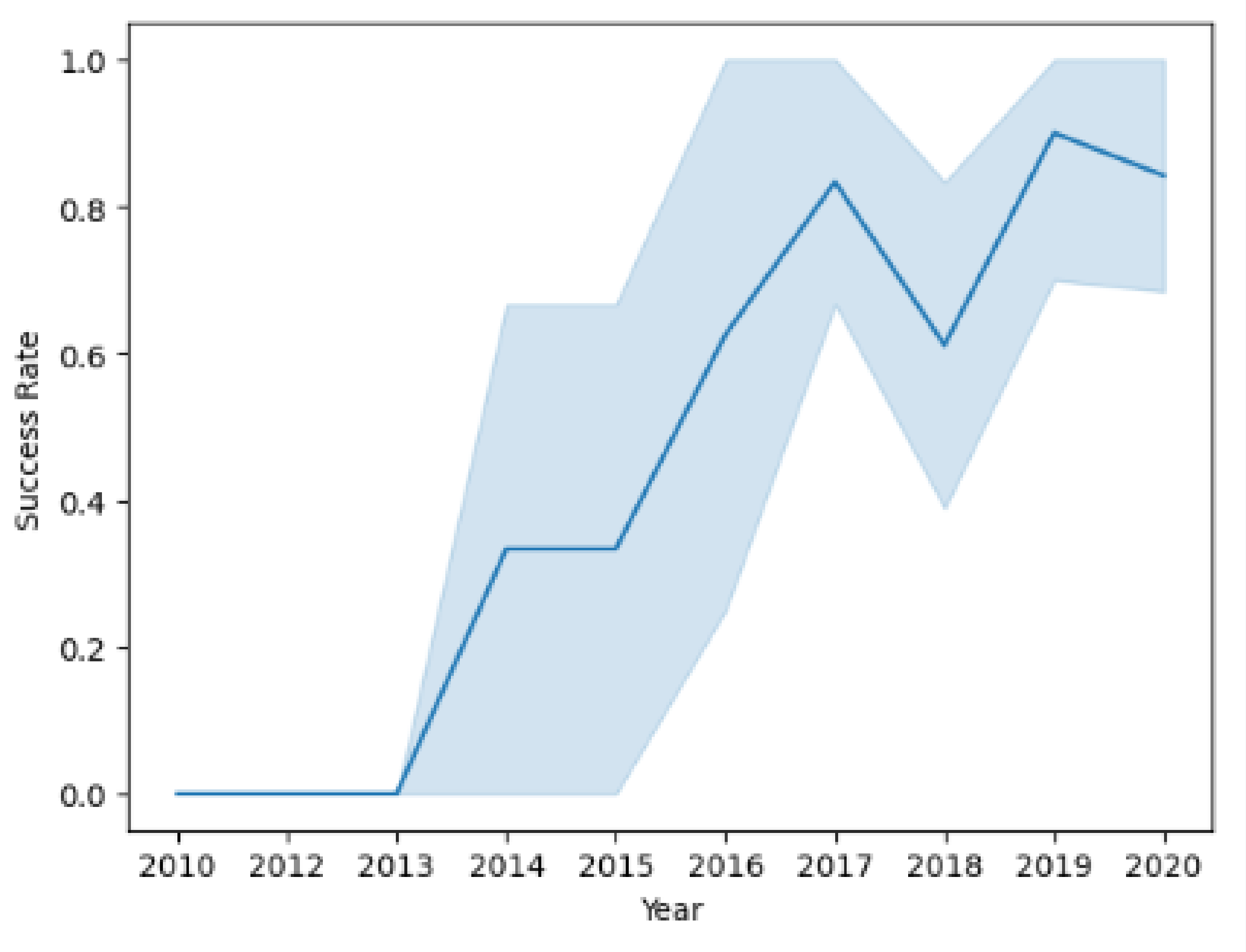
Payload vs. Orbit Type

Payload vs. orbit type



Launch Success Yearly Trend

Line chart of yearly average success rate



All Launch Site Names

- Find the names of the unique launch sites
- Used 'SELECT DISTINCT' statement to return only the unique launch sites from the 'LAUNCH _SITE' column of the SPACEXTBL

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

In [9]:

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

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

Out[9]:

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

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- Used 'LIKE' command with '%' in 'WHERE' to select and display a table of all records where launch sites begin with CCA

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

In [11]:

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

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

Out[11]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX
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
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- Used the 'SUM()' function to return the total sum of 'PAYLOAD_MASS_KG column for Customer 'NASA (CRS)'

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

In [12]:

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(kgs)", Customer FROM 'SPACEXTBL'
```

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

Out[12]:

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

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Used the 'AVG()' function to return the average payload mass carried by booster version F9 v1.1 B1003

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

In [13]:

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM '
```

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

Out[13]:

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

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Used the 'MIN()' function to return the oldest date that a successful landing occurred on the ground pad.

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

Hint:Use min function

```
In [14]:
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing_Outcome" = "Success (ground pad)";
```

```
 * sqlite:///my_data1.db
Done.
Out[14]:
```

MIN(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

- Used 'SELECT DISTINCT' statement to return and list the unique names of boosters with operators >4000 and <6000 to only list booster with payloads between 4000-6000 with landing outcome of 'Success (drone ship)'

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

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Used the 'COUNT()' with the 'GROUP BY' statement to return total number of mission outcomes

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

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Using a subquerry to return and pass the max payload to list all the boosters that have carried the Max payload of 15600 kgs

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

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

```
 * 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

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

- Used the 'substr()' in the select statement to get the month and year from the date column. Landing_outcome was 'Failure (drone ship)'.

List the records which will displav 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)

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

- 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 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;
```

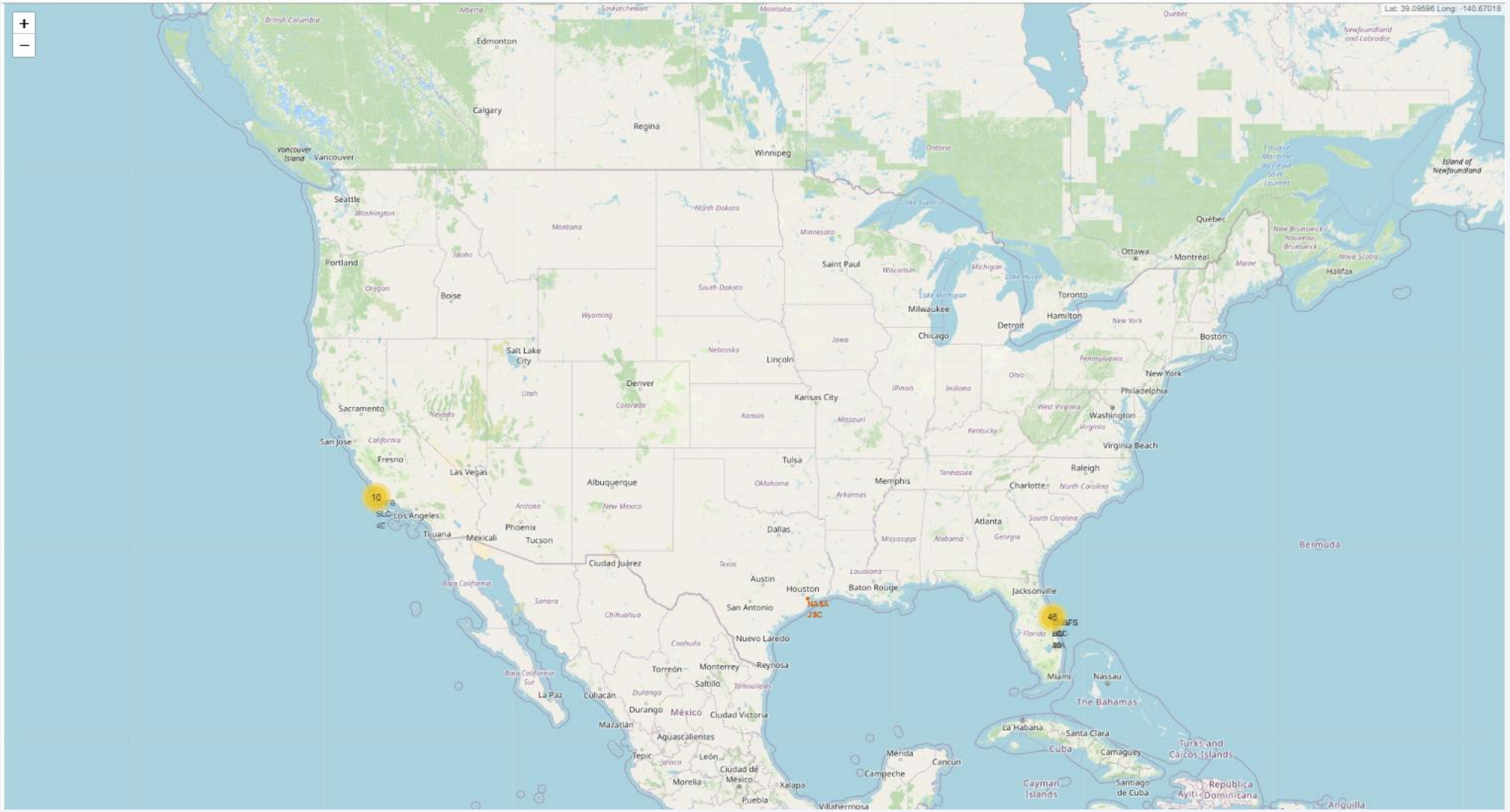
```
 * sqlite:///my_data1.db
Done.
```

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

Markers of all US launch sites

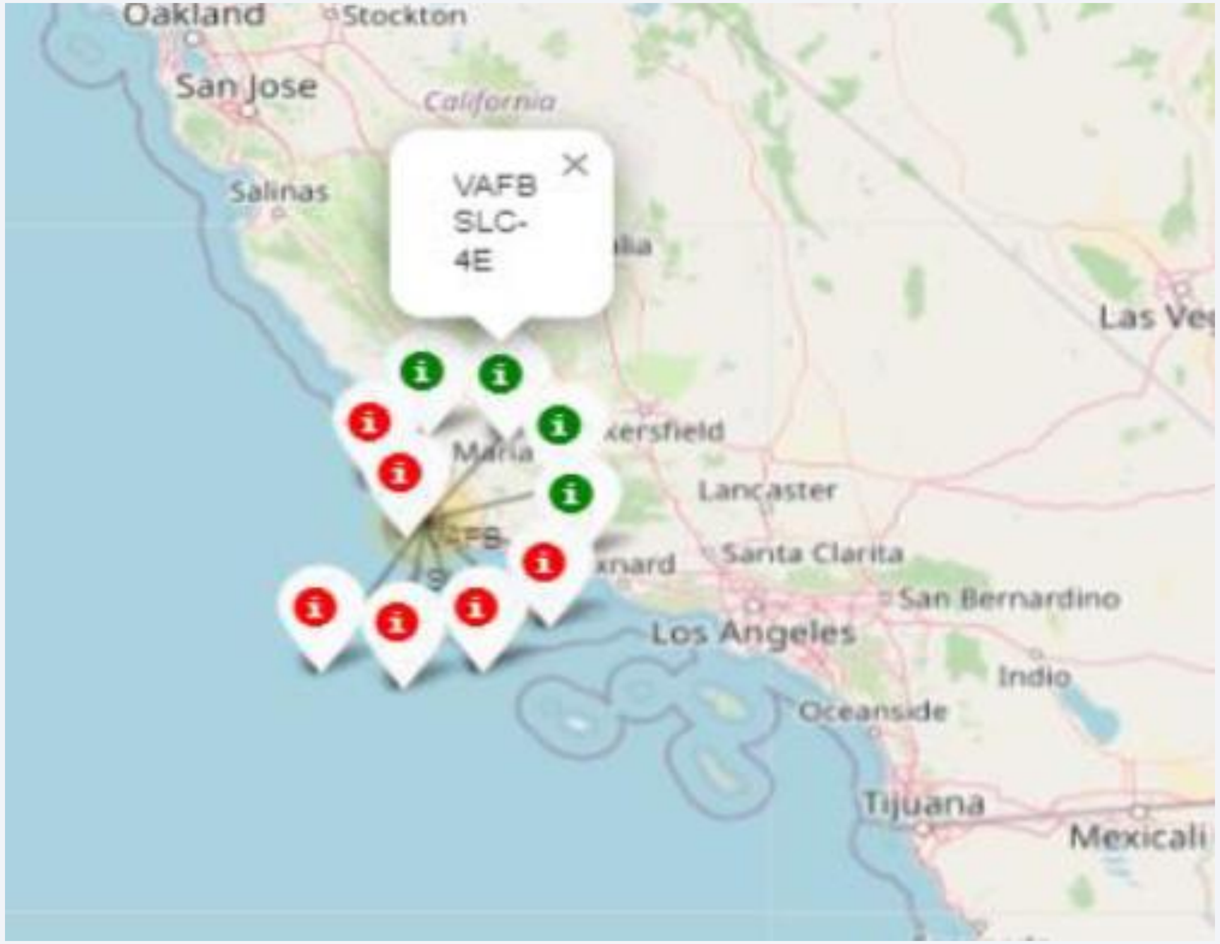


All launch sites are in the Southern tier of the United States and close to the coast

Launch outcomes for Florida

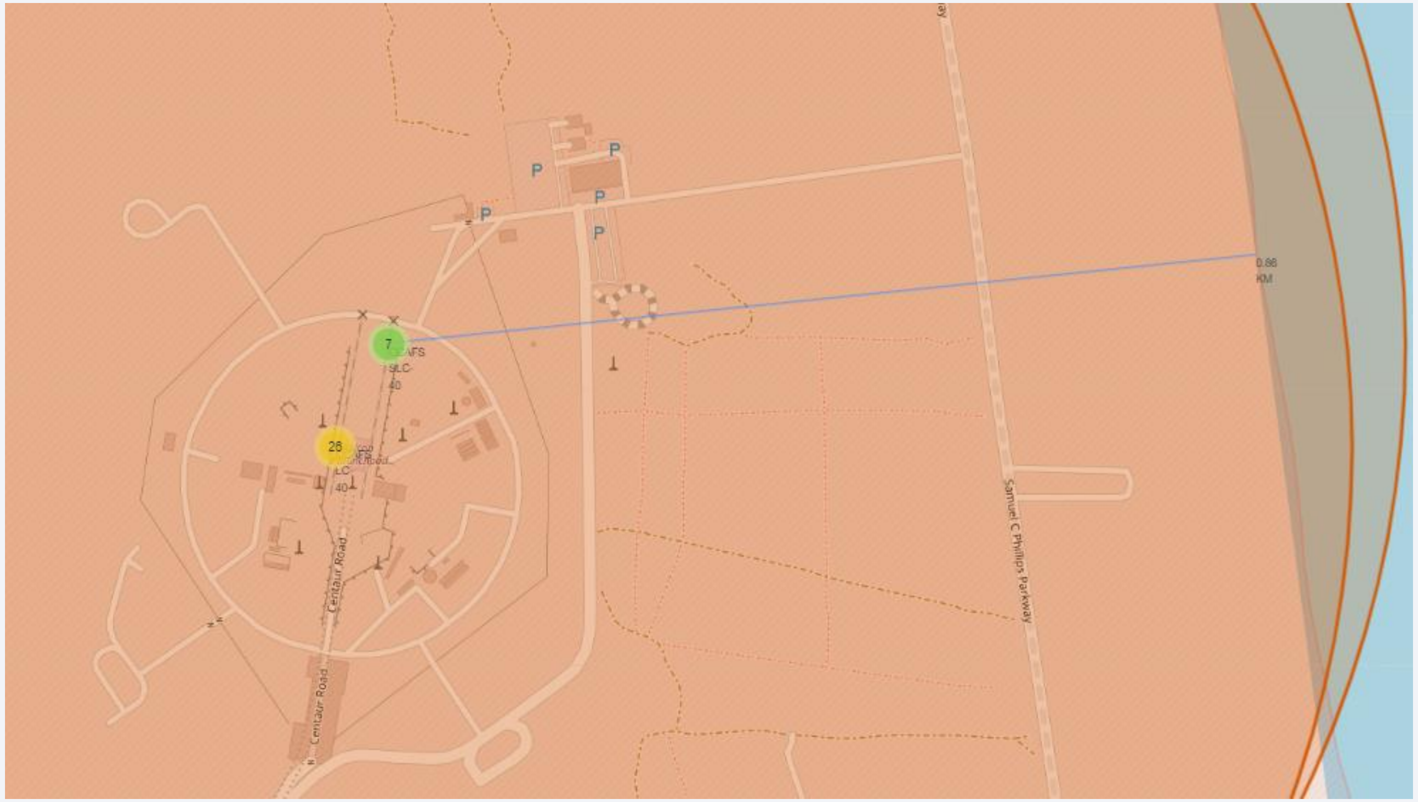
Florida launch site KSC LC-39A has a relatively high success rates compare to CCAFS SLC-40 and CCAFS LC-40

Launch outcomes for California



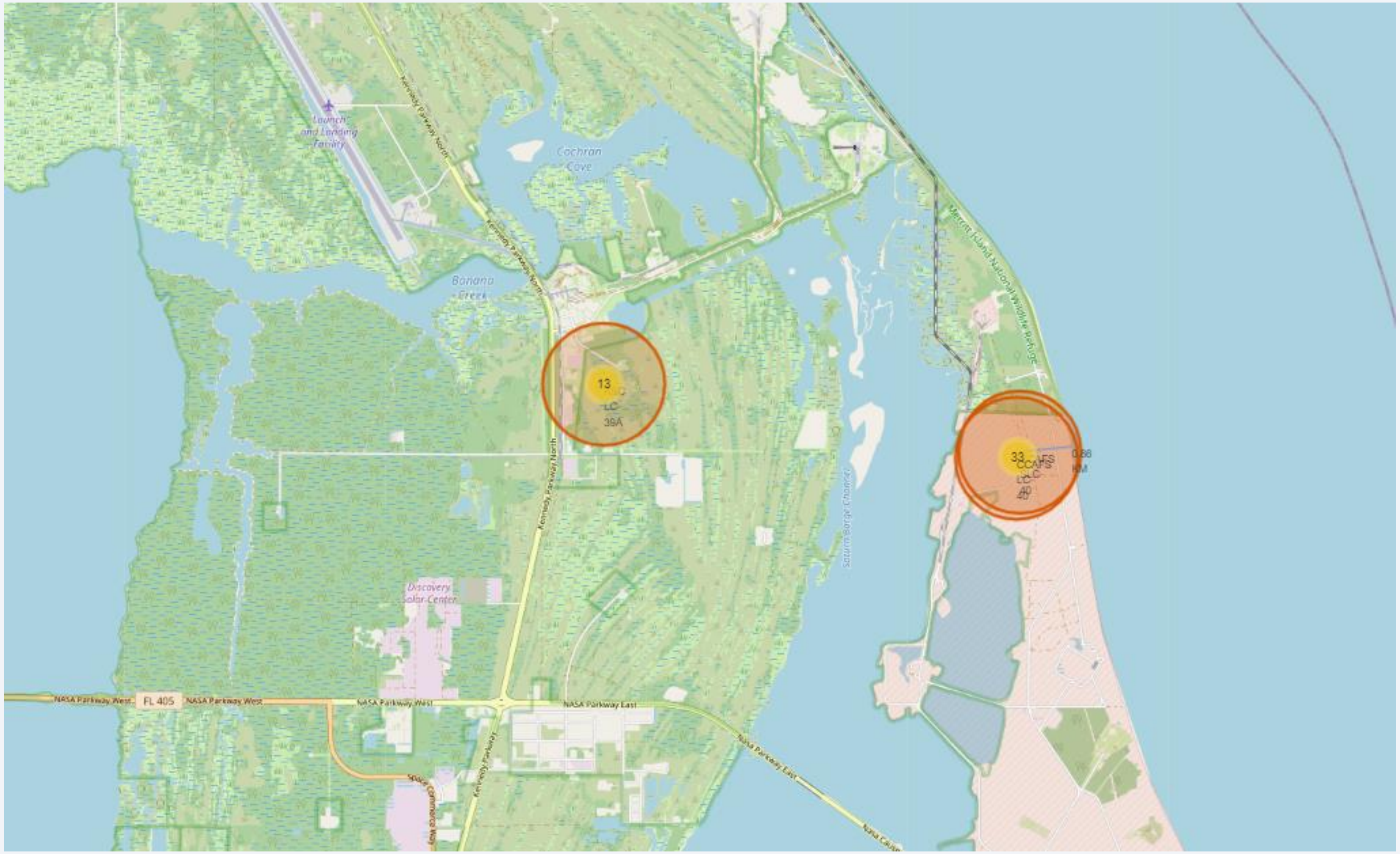
California launch site VAFB SLC-4E has lower success rates compared to Florida's KSC LC-39A launch site.

Distances between a launch site to its proximities



- Launch site CCAFS SLC-40 proximity to coastline is 0.86

Distances between a launch site to its proximities

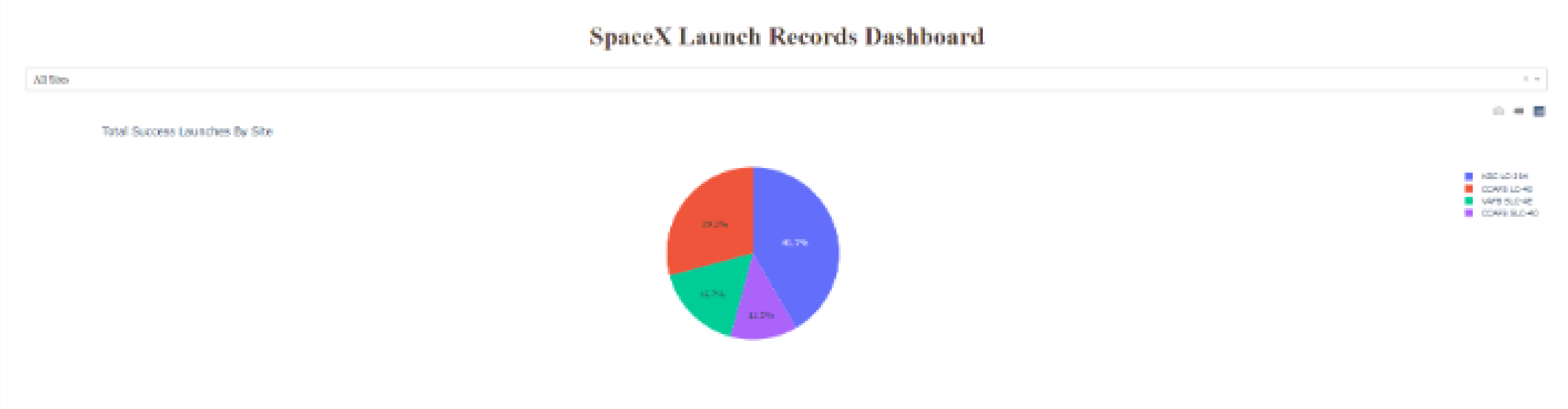


Launch site CCAFS SLC-40 closest to highway

Section 4

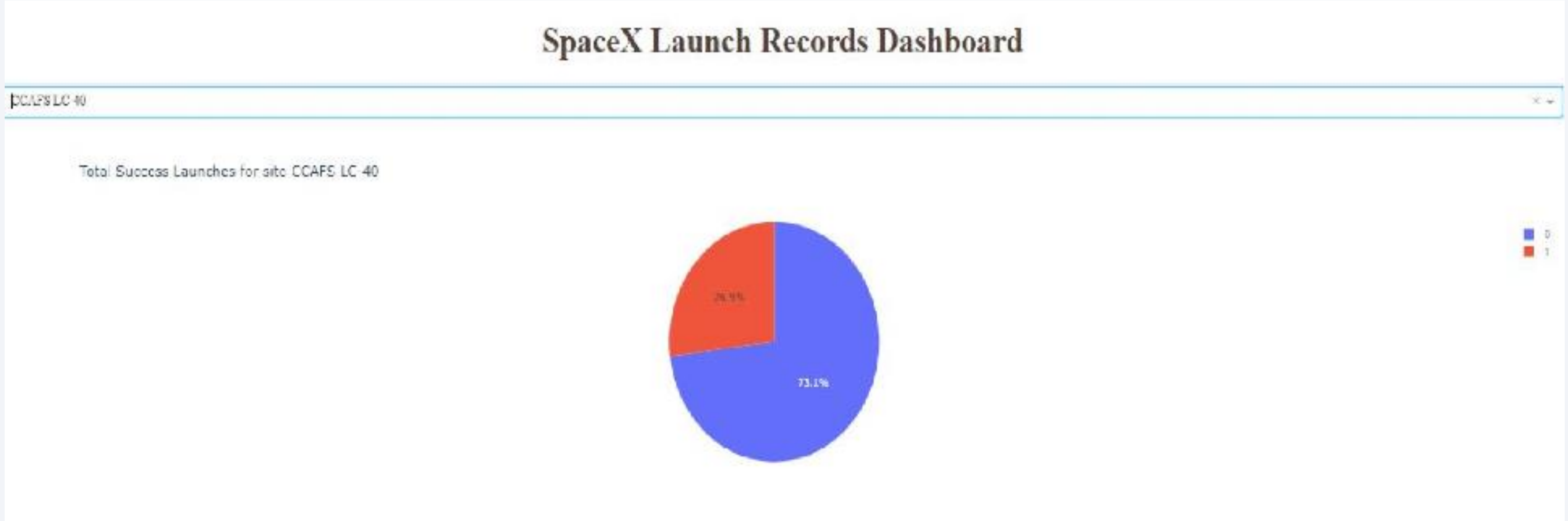
Build a Dashboard with Plotly Dash

<Dashboard Screenshot 1>



- Launch site KSC LC-39A has the highest launch success rate at 42%
- CCAFS LC-40 has the second highest launch success rate at 29%
- VABF SLC-4E has the third highest launch success rate at 17%
- CCAFS SLC-40 has the lowest success rate at 17%

<Dashboard Screenshot 2>



- Launch site CCAFS LC-40 had the 2nd highest success ratio at 73%

<Dashboard Screenshot 3>

- For launch site CCAFS LC-40 the booster version FT has the largest success rate from a payload mass fo >2000kg

Section 5

Predictive Analysis (Classification)

Classification Accuracy

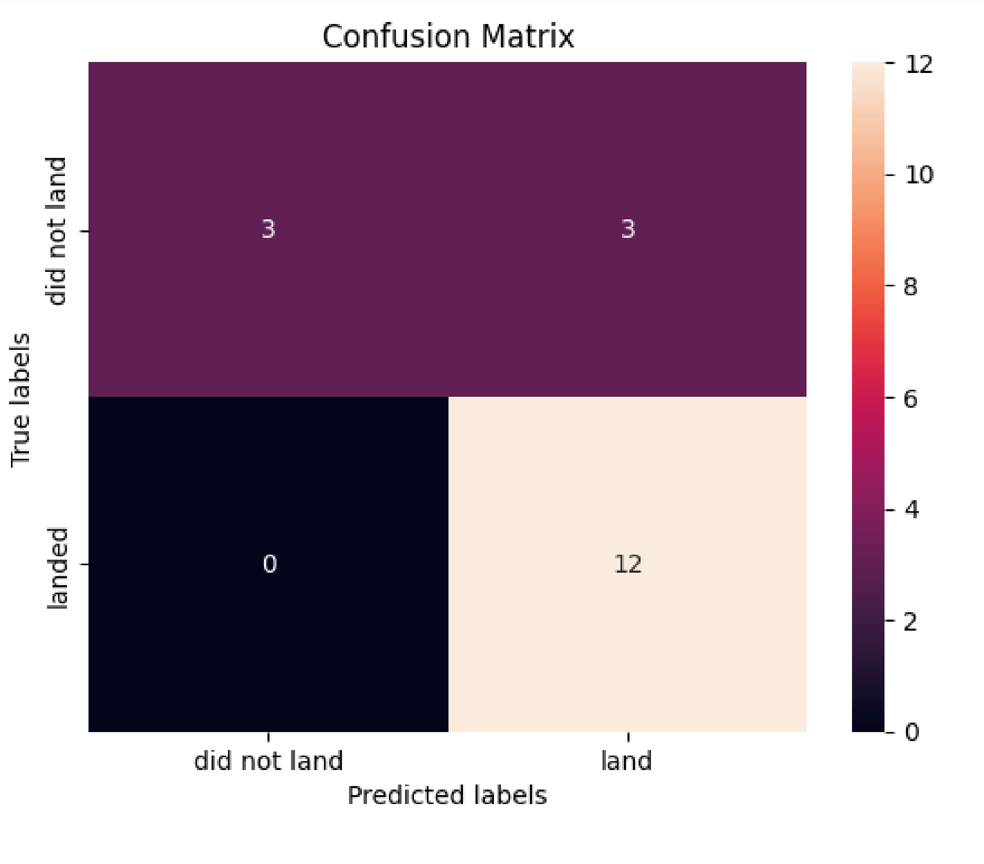
Out[68]:

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

- All methods report an accuracy of 0.833333

Confusion Matrix

- All four classification models returned the same confusion matrices.
- False positives is the major problem for all models.



Conclusions

- Different launch sites have different success rates
- As the flight number increases, so does the success rate
- The Payload vs Launch Site scatter plot you will find for the VAFB-SLC launch site there are no rockets launched for heavypayload mass greater than 10,000
- Orbits ES-L1, GEO, HEO, and SSO have the highest success rates at 100%, while SO has the lowest success rate at approximately 50%
- LEO orbit success appears related to the number of flights. However, there seems to be no relation between flight number when in the GTO orbit.
- With heavy payloads a successful landing is more likely for LEO and ISS.
- The success rate since 2013 has been rising.

Thank you!