IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Victor Vargas Fierro
July 2023

Outline

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

Executive Summary

Summary of methodologies
Data collection through API
Data collection with WebScraping
Data Wrangling
Exploratory data analysis with SQL
Exploratory data analysis with Data Visualization
Visual analytics with Follium
Machine Learning
Summary of all results
Data analysis results
Analytics in screenshots
Predictive results

Introduction

- Project background and context
- The biggest problem facing space exploration is the enormous cost of rockets that prevents the generalization of space flights, in this context the Space X company launched the initiative to reuse the first stages of rockets as the key to reducing costs because the most valuable parts are recovered for a new mission. By creating a learning pipeline through Machine Learning to predict whether the landings of this reusable first stage of the rocket will be successful or not, we will be able to determine what the possibilities of financing space travel to remote destinations would be.
- Problems you want to find answers
    - What factors can determine if the Falcon 9 will land successfully?
    - What is the interaction of characteristics that determine the success rate.
    - What are the operating conditions to guarantee a successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

The SpaceX REST API endpoints, or URL, starts with api.spacexdata.com/v4/

api.spacexdata.com/v4/capsules

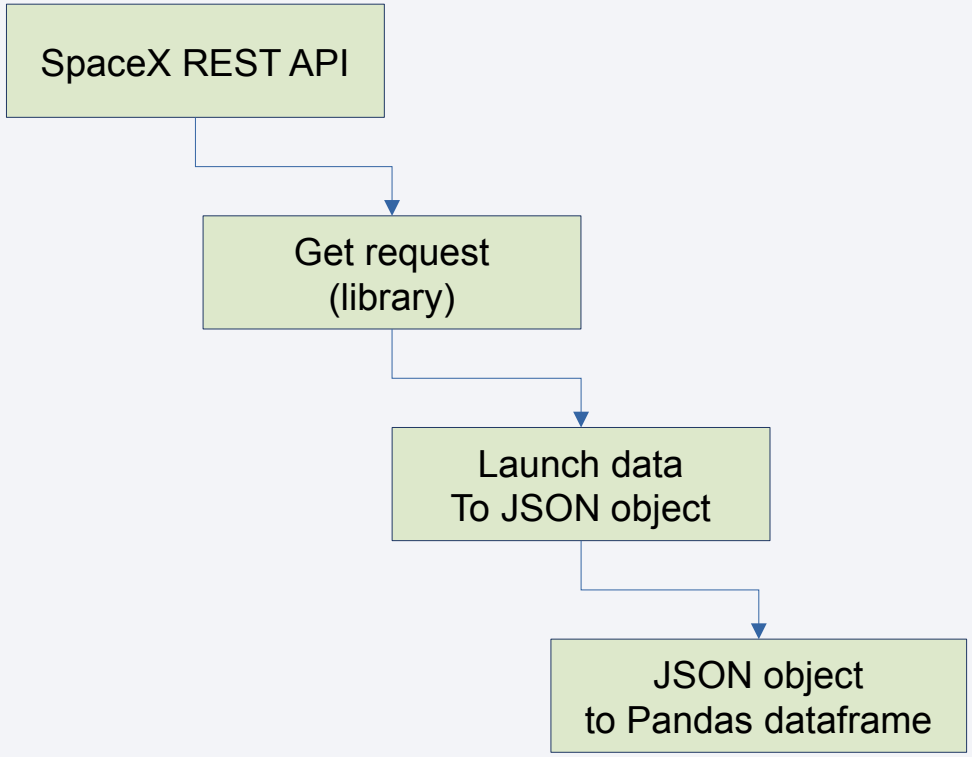
api.spacexdata.com/v4/cores

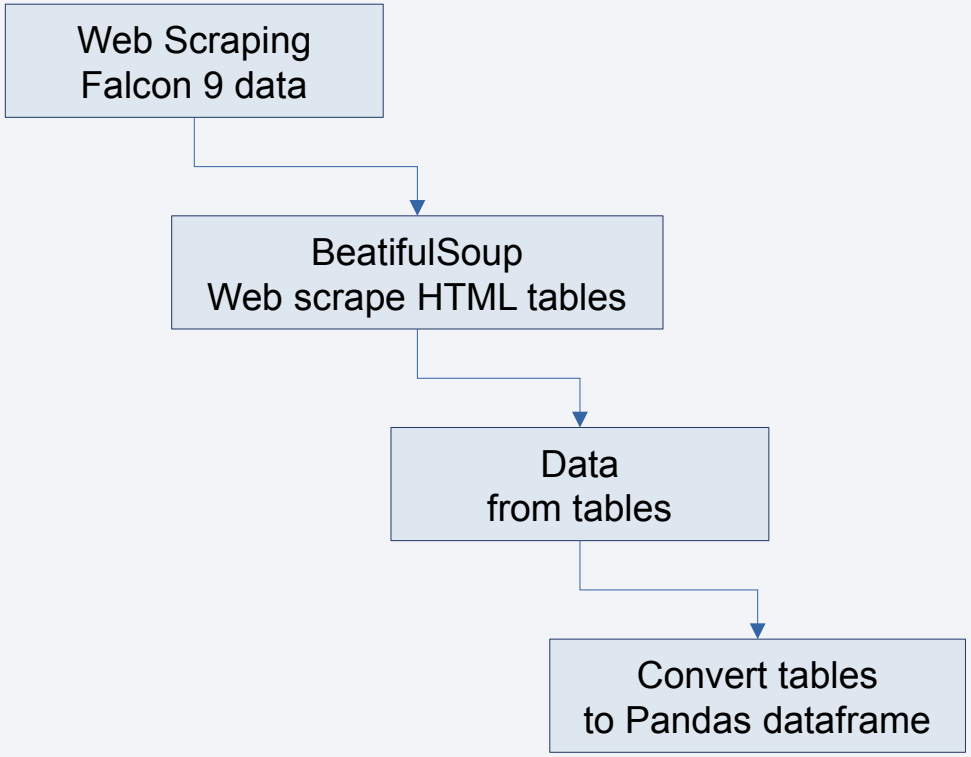
api.spacexdata.com/v4/launches/past

- Perform data wrangling
  - Data wrangling with Wikipedia and Pandas dataframes
- 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

How data sets were collected, the data was collected with the SpaceX REST API and web scraped from Wiki pages.





Data Collection – SpaceX API

Data collection with SpaceX REST API, get request to collect data

URL of the completed SpaceX API calls notebook https://github.com/VictorVargasF/SpaceYcapstone/blob/328348006527522d9afe968f6e56714249bfa52f/jupyter-labs-spacex-data-collection-api.ipynb

Below we will define a series of helper functions that will help us use the API to extract information using identification numbers in the launch data.

From the `rocket` column we would like to learn the booster name.

In [2]:
```
# Takes the dataset and uses the rocket column to call the API and append the data to the list
def getBoosterVersion(data):
    for x in data['rocket']:
       if x:
        response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
        BoosterVersion.append(response['name'])
```

From the `launchpad` we would like to know the name of the launch site being used, the logitude, and the latitude.

In [3]:
```
# Takes the dataset and uses the launchpad column to call the API and append the data to the list
def getLaunchSite(data):
    for x in data['launchpad']:
       if x:
         response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
         Longitude.append(response['longitude'])
         Latitude.append(response['latitude'])
         LaunchSite.append(response['name'])
```

From the `payload` we would like to learn the mass of the payload and the orbit that it is going to.

In [4]:
```
# Takes the dataset and uses the payloads column to call the API and append the data to the lists
def getPayloadData(data):
    for load in data['payloads']:
       if load:
        response = requests.get("https://api.spacexdata.com/v4/payloads/"+load).json()
        PayloadMass.append(response['mass_kg'])
        Orbit.append(response['orbit'])
```

Data Collection - Scraping

Web scraping process

GitHub URL of the web scraping notebook, https://github.com/VictorVargasF/SpaceYcapstone/blob/c6a37e7e076f27a1b0518bc076782da9d39ddd43/jupyter-labs-webscraping.ipynb

In [5]:
```
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 [6]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url).text
```

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

In [7]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response, 'html.parser')
```

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

In [8]:
```
# Use soup.title attribute
print(soup.title)
```

Data Wrangling

Exploratory data analysis and determined the training labels, calculated the number of launches at each site, number and occurrence of each orbits and exported the results to csv.

GitHub URL of data wrangling related notebooks, https://github.com/VictorVargasF/SpaceYcapstone/blob/c6a37e7e076f27a1b0518bc076782da9d39ddd43/labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

In [4]:
```
df=pd.read_csv(dataset_part_1_csv)
df.head(10)
```

Out[4]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPa
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	Na
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	Na
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	Na
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	Na
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	Na
5	6	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	Na
6	7	2014-04-18	Falcon 9	2296.000000	ISS	CCAFS SLC 40	True Ocean	1	False	False	True	Na
7	8	2014-07-14	Falcon 9	1316.000000	LEO	CCAFS SLC 40	True Ocean	1	False	False	True	Na
8	9	2014-08-05	Falcon 9	4535.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	Na
9	10	2014-09-07	Falcon 9	4428.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	Na

Identify and calculate the percentage of the missing values in each attribute

In [5]:
```
df.isnull().sum()/df.shape[0]*100
```

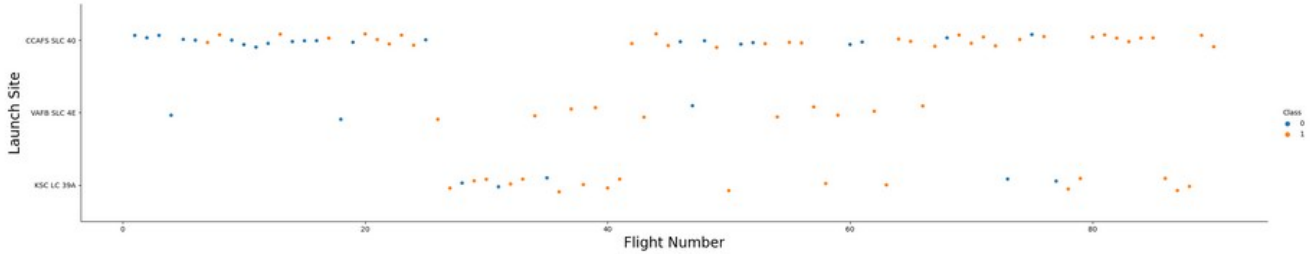
Out[5]:
```
FlightNumber      0.000000
Date              0.000000
BoosterVersion    0.000000
PayloadMass       0.000000
Orbit             0.000000
```

EDA with Data Visualization

Data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.

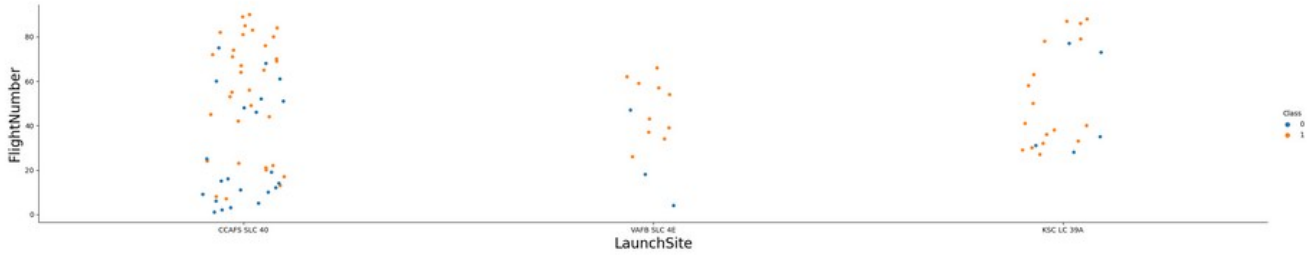
GitHub URL of your completed EDA with data visualization notebook, https://github.com/VictorVargasF/SpaceYcapstone/blob/9b9a1789cd178e872889e0dbdd14e54f26a75d1d/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

5]:
```
### TASK 1: Visualize the relationship between Flight Number and Launch Site
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Use the function `catplot` to plot `FlightNumber` vs `LaunchSite`, set the parameter `x` parameter to `FlightNumber`,set the `y` to `Launch Site` and set the parameter `hue` to `'class'`

6]:
```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the
sns.catplot(y="FlightNumber", x="LaunchSite", hue="Class", data=df, aspect = 5)
plt.xlabel("LaunchSite",fontsize=20)
plt.ylabel("FlightNumber",fontsize=20)
plt.show()
```



EDA with SQL

EDA with SQL to get insight from the data. We wrote queries to find out for instance.

Github link to the notebook is, https://github.com/VictorVargasF/SpaceYcapstone/blob/ef7fcfee4211511533622aa8872e66701ab07fd1/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Task 1

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

In [7]:
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

Out[7]:

Launch_Sites
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
None

Task 2

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

In [8]:
```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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

Out[8]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	

Build an Interactive Map with Folium

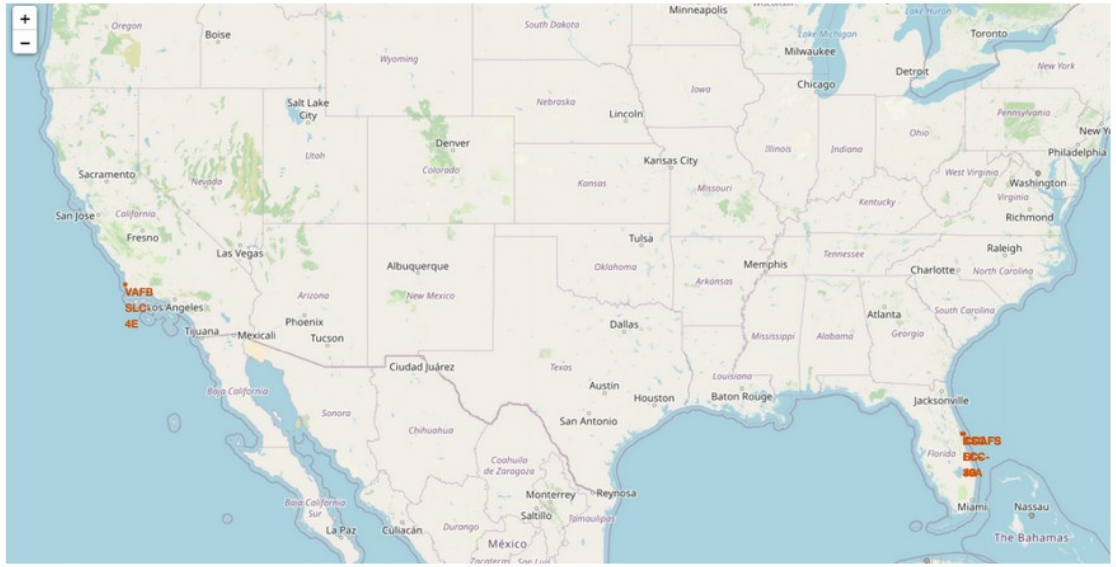
Marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.

GitHub URL interactive map with Folium map, https://github.com/VictorVargasF/SpaceYcapstone/blob/c6cc6e9401716550135a4571d51ddee8784bfbf2/lab_jupyter_launch_site_location.jupyterlite.ipynb

```
folium.map.Marker(coordinate, icon=DivIcon(icon_size=(20,20),icon_anchor=(0,0), html='<div style="font-size:
12; color:#d35400;"><b>%s</b></div>' % 'label', ))
```

```
# Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
# For each launch site, add a Circle object based on its coordinate (Lat, Long) values. In addition, add Launch s
```

The generated map with marked launch sites should look similar to the following:



Build a Dashboard with Plotly Dash

Plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

GitHub URL Plotly Dash lab, https://github.com/VictorVargasF/SpaceYcapstone/blob/4e9c570473041af09270851b617549ed643a9645/PlottyDash.py

Predictive Analysis (Classification)

Import: numpy, pandas, matplotlib, seaborn; to sklearn: preprocessing, train_test_split, GridSearchCV, LogisticRegression, SVC, DecisionTreeClassifier, KneighborsClassifier.

Transformed data and split data into training for testing.

GitHub URL of predictive analysis lab, https://github.com/VictorVargasF/SpaceYcapstone/blob/daad4e0ee83baabcc7178e73574ef69d2fa2fd93/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

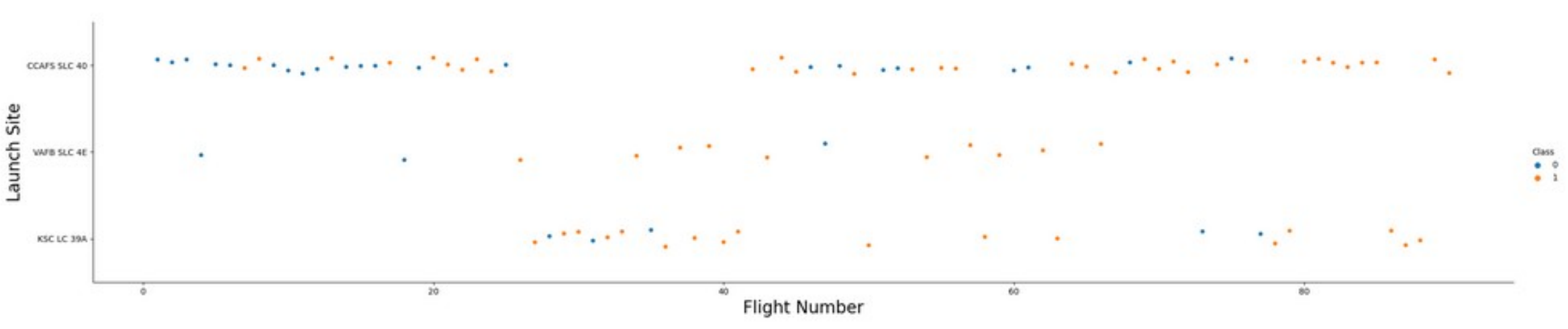
Results

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

Section 2

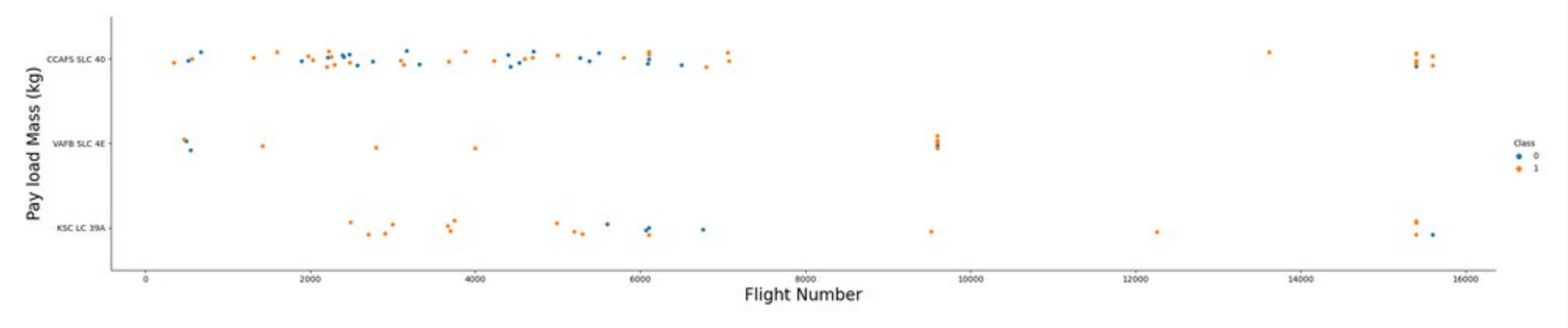
Insights drawn from EDA

Flight Number vs. Launch Site



The Flight Number vs. Launch Site, the larger the flight amount at a launch site, the greater the success rate at a launch site.

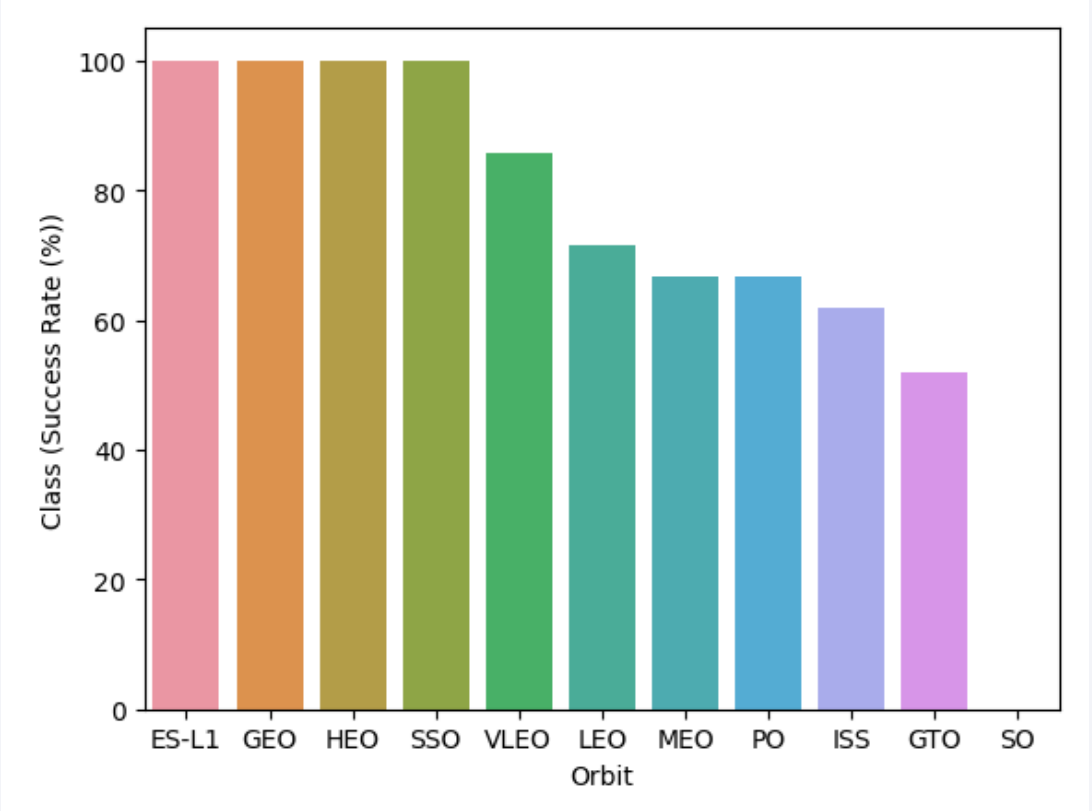
Payload vs. Launch Site



Payload vs. Launch Site, for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass (greater than 10000).

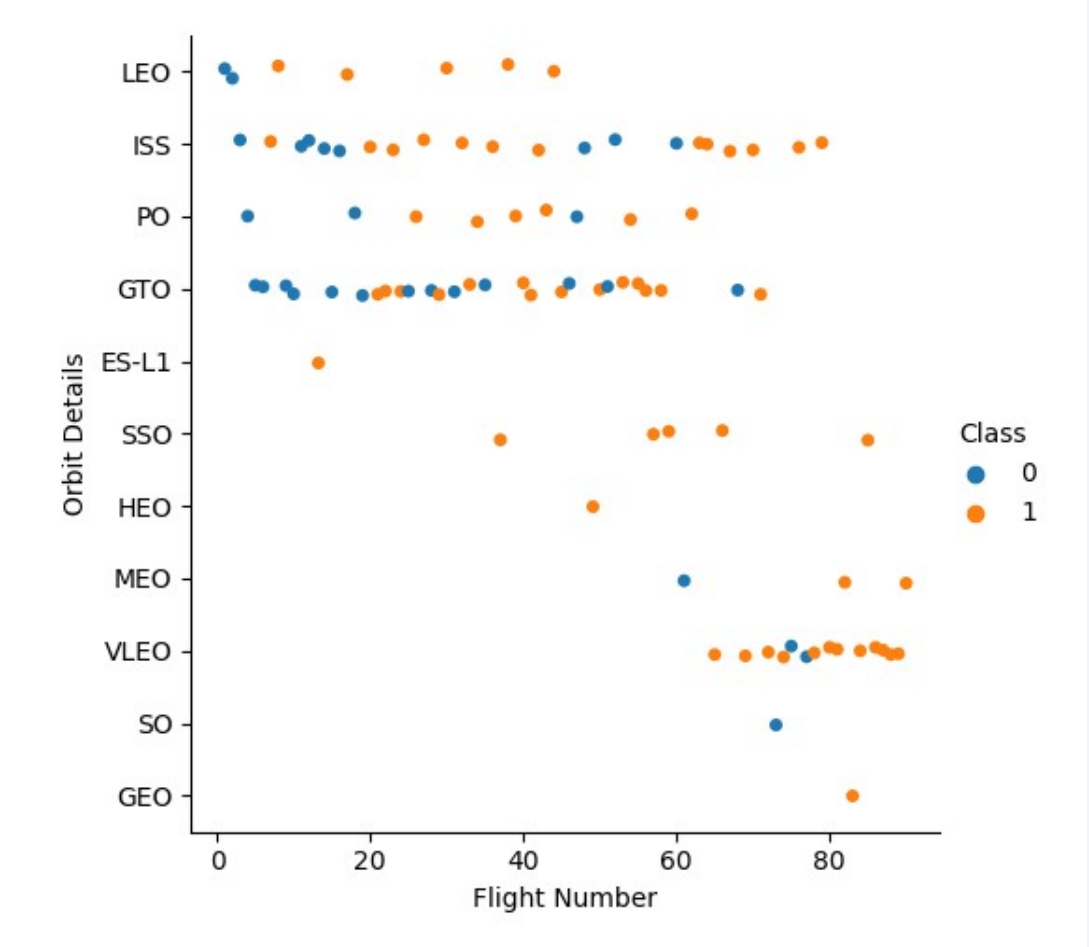
Success Rate vs. Orbit Type

we can see which orbit had the highest success rate



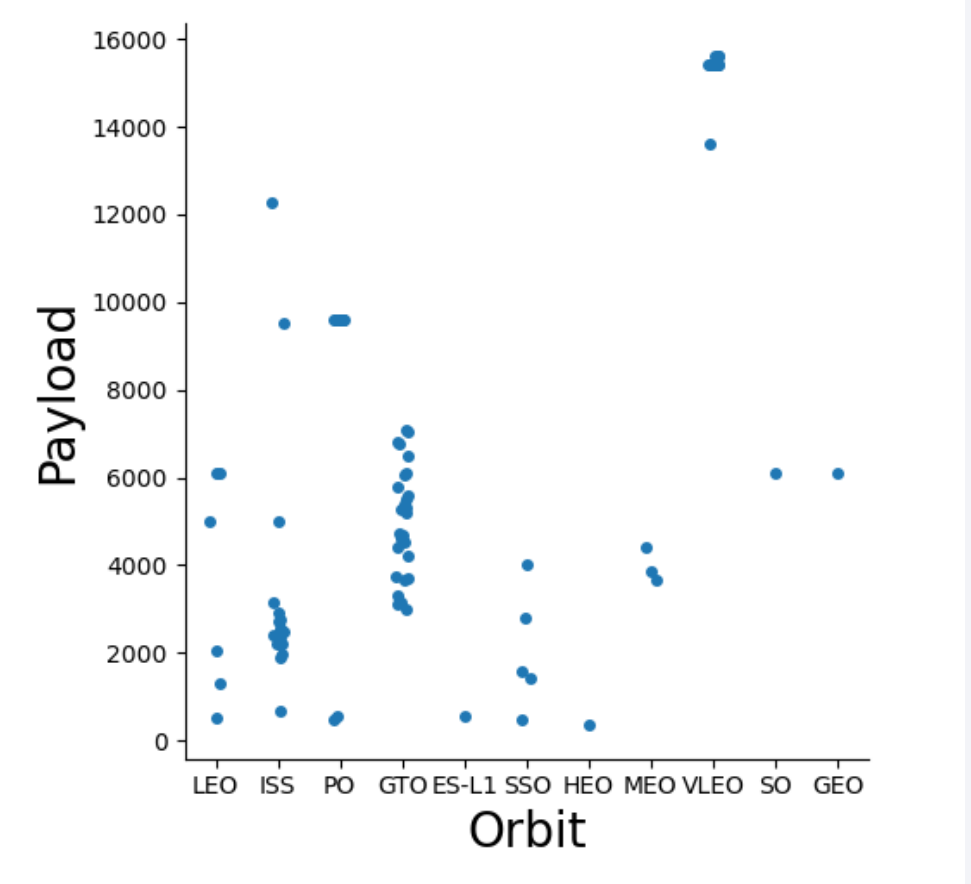
Flight Number vs. Orbit Type

Flight number vs. Orbit type, the LEO orbit success is related to the number of flights, GTO orbit no relationship between flight number.



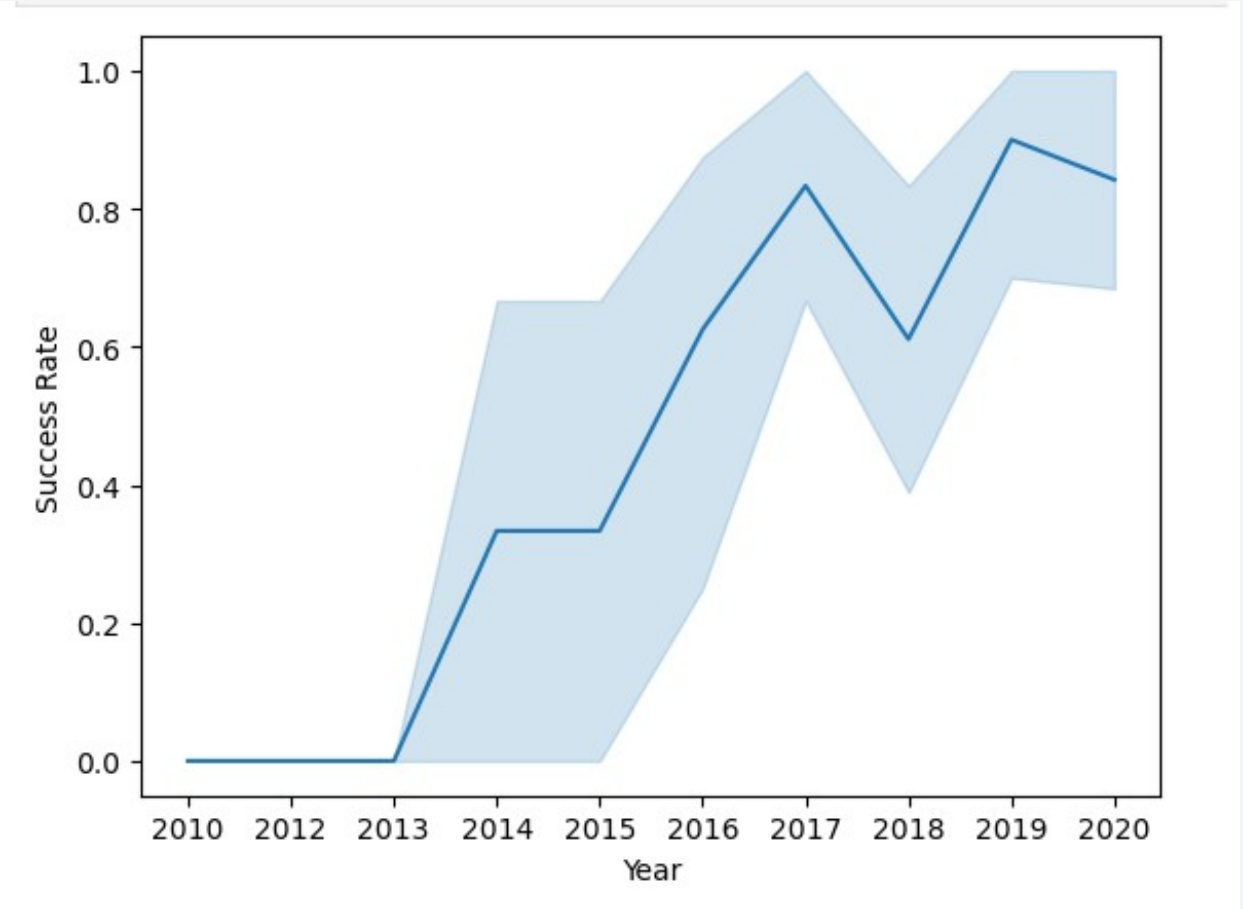
Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type
- Show the screenshot of the scatter plot with explanations



Launch Success Yearly Trend

Yearly average success rate,
Since 2013 the success rate
shows a increase until 2020.



All Launch Site Names

Task 1

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

In [7]:
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

Out[7]:

Launch_Sites
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
None

Launch Site Names Begin with 'CCA'

Task 2

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

In [8]:
```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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

Out[8]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	

Total Payload Mass

Task 3

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

In [11]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA
```

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

Out[11]:

Total Payload Mass(Kgs)	Customer
45596.0	NASA (CRS)

Average Payload Mass by F9 v1.1

Task 4

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

In [12]:
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booste
```

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

Out[12]:

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

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

In [42]:
```
%sql SELECT MIN(Date) FROM SPACEXTBL WHERE Landing_Outcome = "Success (ground pad)";
```

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

Out[42]:

MIN(Date)
01/08/2018

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

In [41]:
```
%sql SELECT Booster_Version,Payload FROM SPACEXTBL WHERE Landing_Outcome = "Success (drone ship)" AND "PAYLOAD_MA
```

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

Out[41]:

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

Task 7

List the total number of successful and failure mission outcomes

In [26]:
```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

Out[26]:

Mission_Outcome	Total
None	0
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

Task 8

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

In [37]:
```
%sql SELECT Booster_Version,Payload,PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAY
```

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

Out[37]:

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600.0
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600.0
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600.0
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600.0
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600.0
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600.0
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600.0
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600.0
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600.0
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600.0
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600.0
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600.0

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.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

In [45]:
```
%sql SELECT substr(Date,7,4), substr(Date, 4, 2),Booster_Version,Launch_Site,Payload,PAYLOAD_MASS__KG_,Mission_Ou
```

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

Out[45]:

substr(Date,7,4)	substr(Date, 4, 2)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	Landing_Outcome
2015	10	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395.0	Success	Failure (drone ship)
2015	04	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898.0	Success	Failure (drone ship)

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

Task 10

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.

In [47]:
```
%sql SELECT * FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017
```

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

Out[47]:

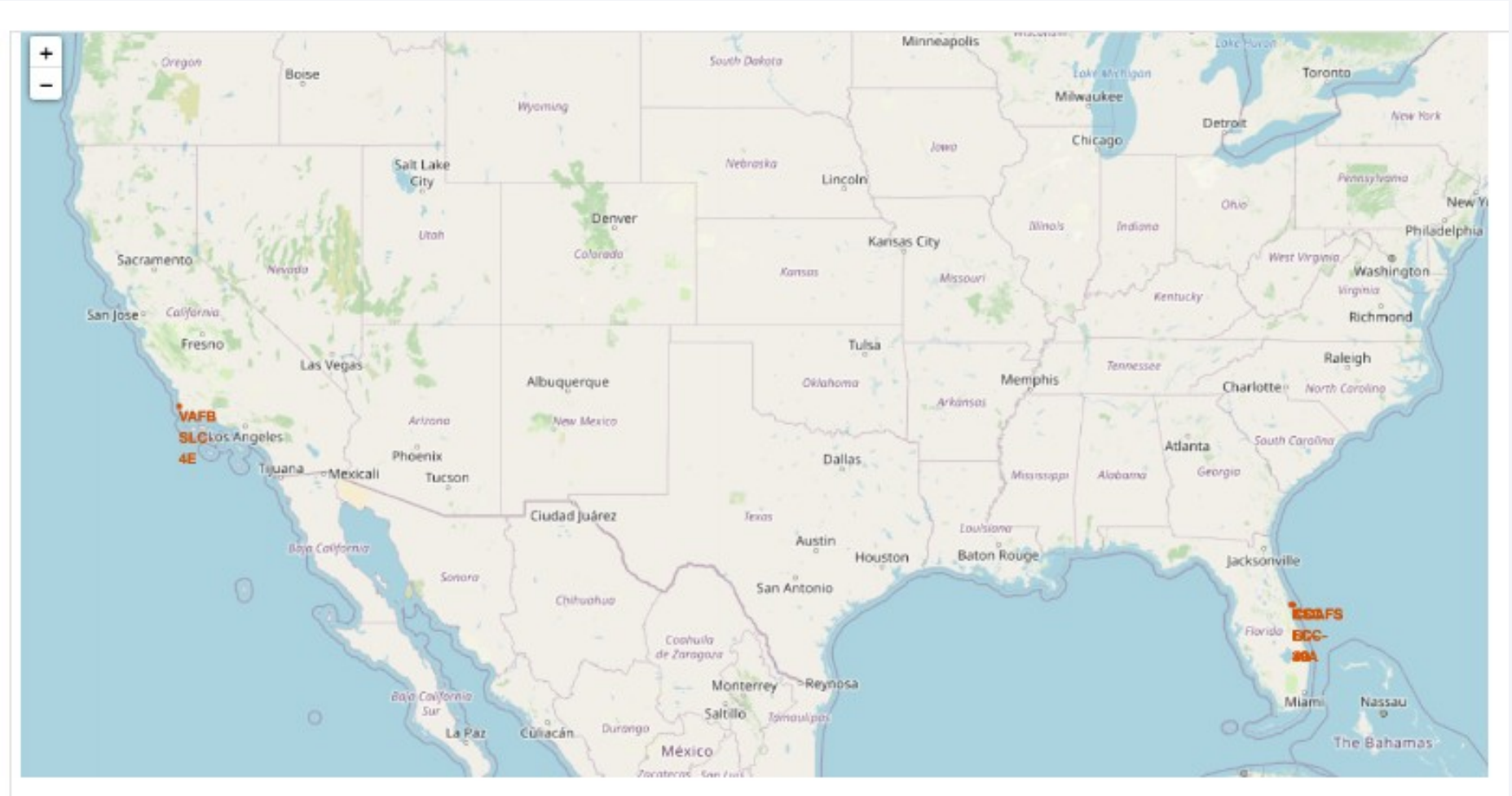
Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome
19/02/2017	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490.0	LEO (ISS)	NASA (CRS)	Success
18/10/2020	12:25:57	F9 B5 B1051.6	KSC LC-39A	Starlink 13 v1.0, Starlink 14 v1.0	15600.0	LEO	SpaceX	Success
18/08/2020	14:31:00	F9 B5 B1049.6	CCAFS SLC-40	Starlink 10 v1.0, SkySat-19, -20, -21, SAOCOM 1B	15440.0	LEO	SpaceX, Planet Labs, PlanetIQ	Success
18/07/2016	4:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257.0	LEO (ISS)	NASA (CRS)	Success
18/04/2018	22:51:00	F9 B4 B1045.1	CCAFS SLC-40	Transiting Exoplanet Survey Satellite (TESS)	362.0	HEO	NASA (LSP)	Success
17/12/2019	0:10:00	F9 B5 B1056.3	CCAFS SLC-40	JCSat-18 / Kacific 1, Starlink 2 v1.0	6956.0	GTO	Sky Perfect JSAT, Kacific 1	Success
16/11/2020	0:27:00	F9 B5B1061.1	KSC LC-39A	Crew-1, Sentinel-6 Michael Freilich	12500.0	LEO (ISS)	NASA (CCP)	Success
15/12/2017	15:36:00	F9 FT B1035.2	CCAFS SLC-40	SpaceX CRS-13	2205.0	LEO (ISS)	NASA (CRS)	Success
15/11/2018	20:46:00	F9 B5 B1047.2	KSC LC-39A	Es hail 2	5300.0	GTO	Es hailSat	Success
14/08/2017	16:31:00	F9 B4 B1039.1	KSC LC-39A	SpaceX CRS-12	3310.0	LEO (ISS)	NASA (CRS)	Success

Section 3

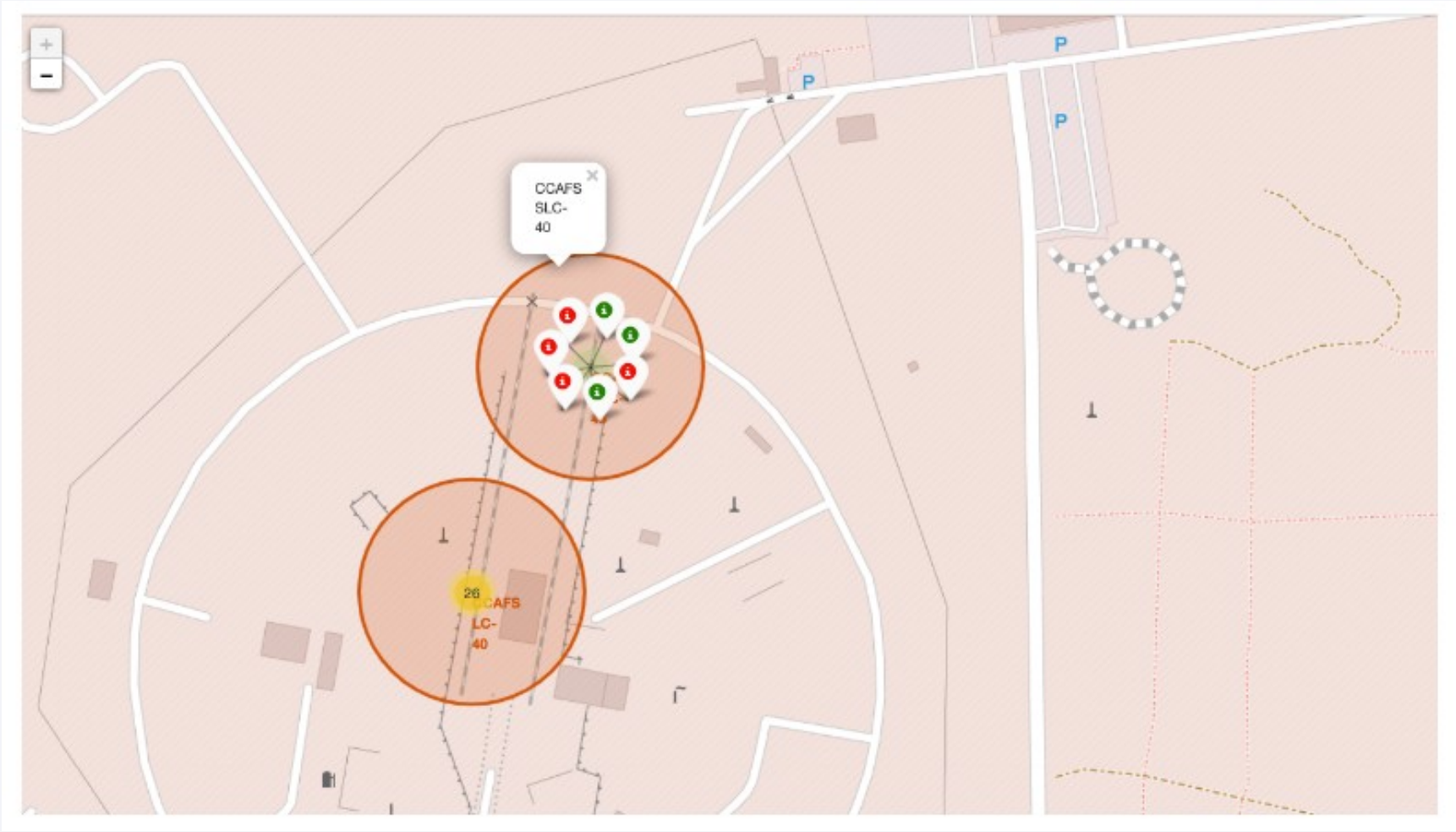
Launch Sites Proximities Analysis

SpaceX Launch Sites on United State of America

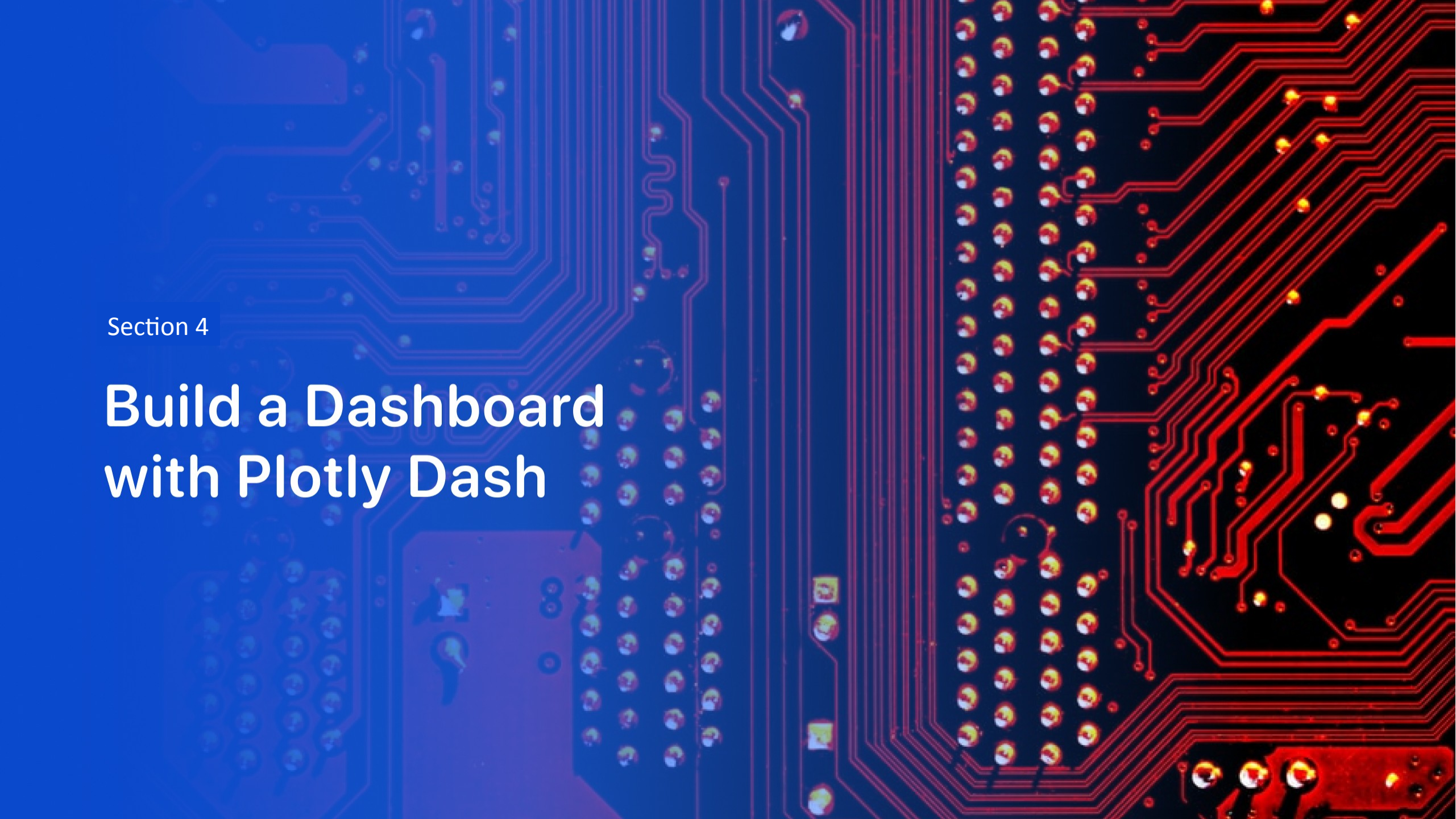
Launch sites to locate: The VAFB SLC – 4E in California, the rest in Florida.



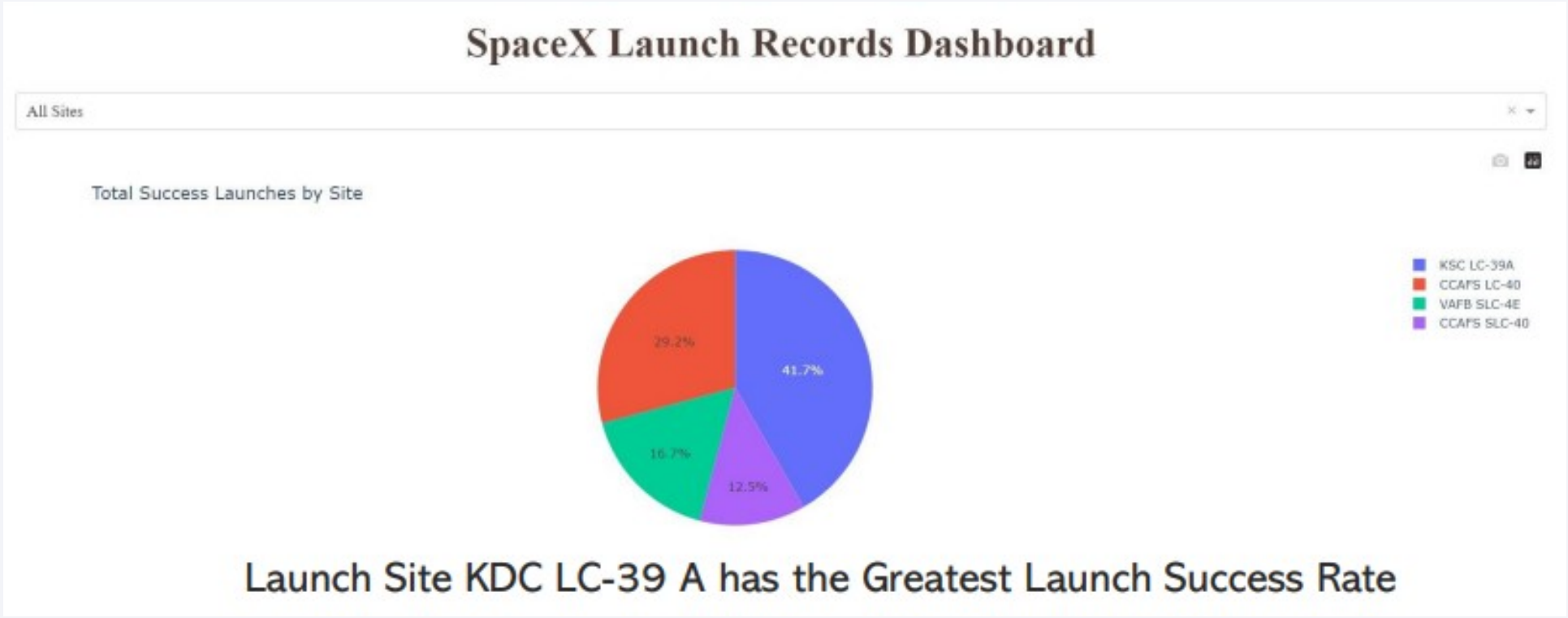
Success/Failed Launches for GCAFS SL40



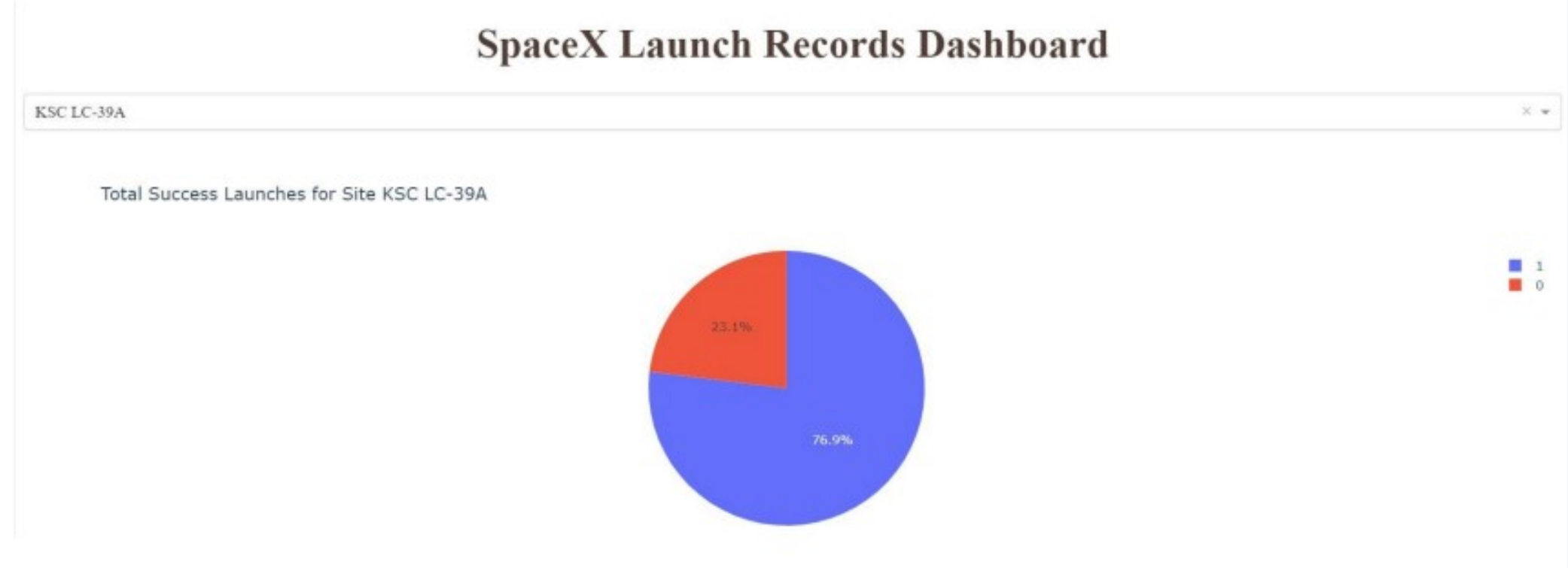
Launch site GCAFS SL40 proximity from Coastline


Section 4
Build a Dashboard with Plotly Dash

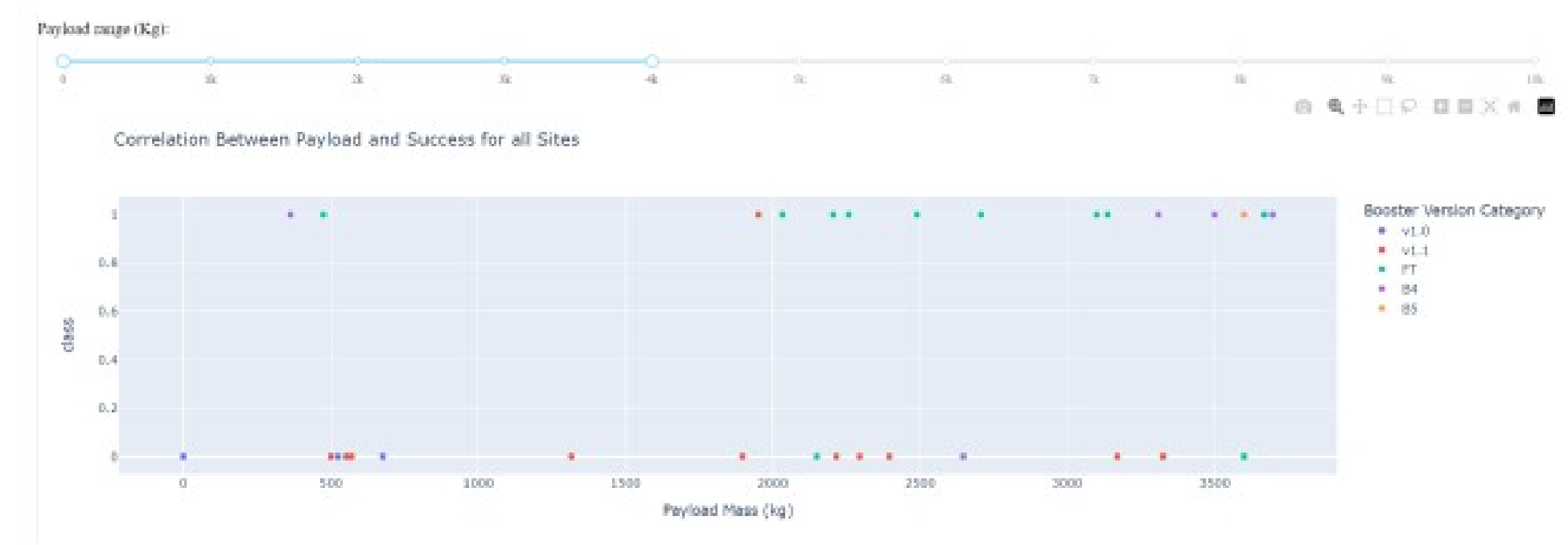
Total Success Launches by all Sites



Success/Failure Launch Rate



Payload Weight



Section 5

Predictive Analysis (Classification)

Classification Accuracy

TASK 12

Find the method performs best:

In [41]:

```
Report = pd.DataFrame({'Method' : ['Test Data Accuracy']})
knn_accuracy=knn_cv.score(x_test, y_test)
Decision_tree_accuracy=tree_cv.score(x_test, y_test)
SVM_accuracy=svm_cv.score(x_test, y_test)
Logistic_Regression=logreg_cv.score(x_test, y_test)
Report['Logistic_Reg'] = [Logistic_Regression]
Report['SVM'] = [SVM_accuracy]
Report['Decision Tree'] = [Decision_tree_accuracy]
Report['KNN'] = [knn_accuracy]
Report.transpose()
```

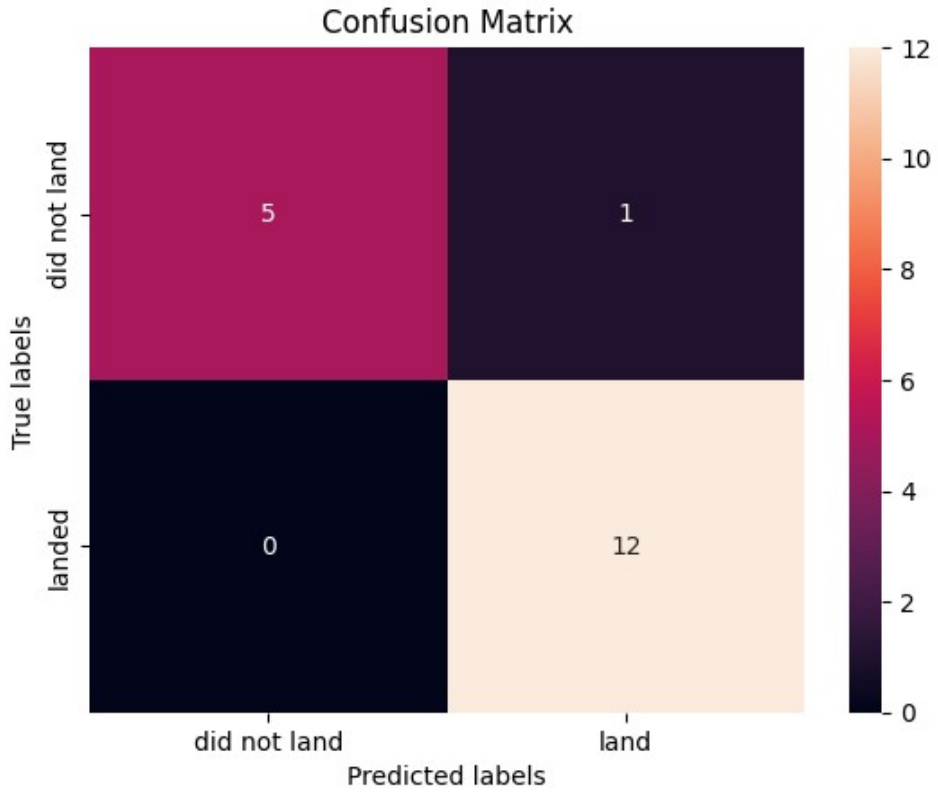
Out[41]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.944444
KNN	0.833333

Decision Tree classifier has the highest classification accuracy of 0.944

Confusion Matrix

```
In [34]: yhat = tree_cv.predict(x_test)
         plot_confusion_matrix(y_test,yhat)
```



the best performing model with an explanation

Conclusions

The more flights at a launch site, the higher the success rate at that site.

ES-L1, GEO, HEO and SSO are the orbits with the highest success rate.

With heavier payloads, success increases for LEO, ISS and Polar orbits.

Launch success rate increased from 2013 to 2020 with minor fluctuations.

The KDC LC-39 A launch site has the highest success rate.

The Decision Tree Classifier is the best machine learning algorithm for predicting whether the first stage of SpaceX's Facon 9 rocket will land successfully.

Thank you!