IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

T. ASWIIN
09/06/2024

Outline

Executive Summary

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars much of the savings is because SpaceX can reuse the first stage. Therefore if landing of first stage is determined, launch cost can also be determined.

This report provides answers to the below questions:

1. How *"Orbit", "LaunchSite", "LandingPad", "Serial"* play an important role in determining landing outcome?

1. Does success outcome increase with increase in payload mass.

2. How often were most unsuccessful landings seen.

This information can be used if an alternate company wants to bid against SpaceX for a rocket launch by gathering information about Space X and creating dashboards.

Introduction

SpaceX has been most successful in space programs. SpaceX's accomplishments include:

- ❑ Sending spacecraft to the International Space Station.
- ❑ Starlink, a satellite internet constellation providing satellite Internet access.
- ❑ Sending manned missions to Space.

One reason SpaceX can do this is by reusing the first stage but the landing of the first stage is not always certain. This stage is quite large and expensive.

Problem Statement

To determine the price of each launch and also if SpaceX will reuse the first stage by training machine learning model, using public information and creating dashboards.

Section 1

Methodology

Methodology

Executive Summary

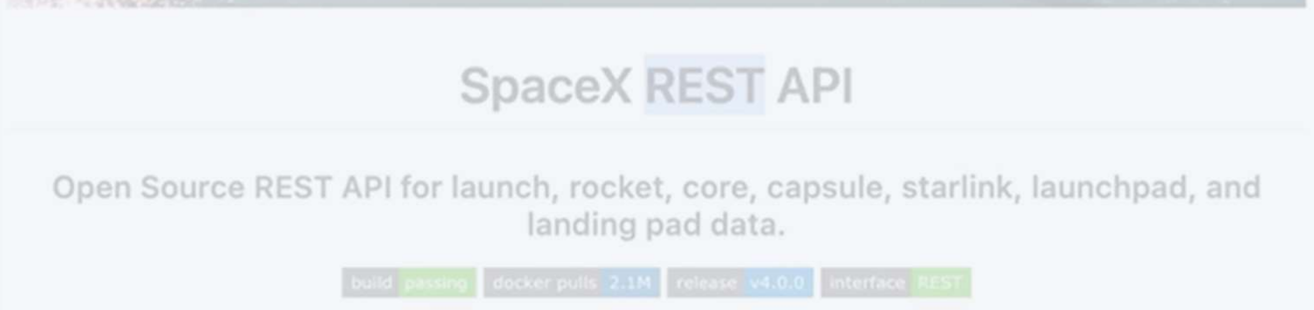
- Data collection methodology
- Perform data wrangling
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

SpaceX launch data is gathered from an API, specifically the SpaceX REST API and the endpoint: api.spacexdata.com/v4/launches/past

This URL is used to target a specific endpoint of the API and request to get past launch data. The response will be in the form of a JSON json_normalize function will normalize the structured json data into a flat table.

Another popular data source for obtaining Falcon 9 Launch data is web scraping related Wiki pages that contain valuable Falcon 9 launch records using the Python BeautifulSoup package.



Data Collection – SpaceX API

Import Libraries and Define Auxiliary Functions that will use the API to extract information using identification numbers in the launch data.

↓

Request and parse the SpaceX launch data from SpaceX API using the GET request on: https://api.spacexdata.com/v4/launches/past

↓

Make the requested JSON results more consistent using a static response object: https://cf-courses-data.s3.us.cloud-objectstorage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json

↓

Decode the response content as a Json and turn it into a Pandas dataframe using json_normalize()

↓

Store the outputs from Auxiliary Functions in lists that will be used to create a new dataframe.

↓

Filter the dataframe to only include `Falcon 9` launches and replace

np.nan values in the data with the mean value.

Data Collection – SpaceX API

.json into DataFrame using .json_normalise()

```
data.head()
```

rocket	success	failures	details	crew	ships	capsules	payloads	launchpad	flight_number	name	date_utc
d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine failure'}]	Engine failure at 33 seconds and loss of vehicle	[]	[]	[]	[5eb0e4b5b6c3bb0006eeb1e1]	5e9e4502f5090995de566f86	1	FalconSat	2006-03-24T22:30:00.000Z

Final DataFrame including Auxiliary Function Results

```
launch_df.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin1A	167.743129	9.047721
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2A	167.743129	9.047721
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2C	167.743129	9.047721
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin3C	167.743129	9.047721

GitHub Repository Link: https://github.com/aswiint/SpaceX-Landing-Prediction/blob/main/SpaceX_Data_Collection_API.ipynb

Data Collection – Scraping

Import libraries and define auxiliary functions to process web scraped HTML table.

Perform a HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP *response* from: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

Create a BeautifulSoup object from the HTML *response*

Starting from the third table (target table) iterate through the <th> elements and apply extract_column_from_header() to extract column name

Create a dictionary with key, value as column names and parsed launch record values and store it in dataframe.

Data Collection – Scraping

Extracted column names through the <th> elements

```
print(column_names)
```

```
['Flight No.', 'Date and time ( )', 'Launch site', 'Payload', 'Payload mass', 'Orbit', 'Customer', 'Launch outcome']
```

Parsed launch records

```
1
4 June 2010
18:45
F9 v1.0B0003.1
CCAFS
Dragon Spacecraft Qualification Unit
Dragon Spacecraft Qualification Unit
LEO
<td><a href="/wiki/SpaceX" title="SpaceX">SpaceX</a>
</td>
Success

Failure
2
8 December 2010
15:43
F9 v1.0B0004.1
CCAFS
Dragon
Dragon
LEO
```


GitHub Repository Link: https://github.com/aswiint/SpaceX-Landing-Prediction/blob/main/SpaceX_WebScraping.ipynb

Data Wrangling

Import libraries and load the dataset

↓

Use value_counts() on *LaunchSite* column to determine the number of launches on each site.

↓

Use value_counts() on *Orbit* column to determine the occurrence of each orbit.

↓

Use value_counts() on *Outcome* column to determine the number of mission outcomes.

↓

Create a set of failed outcomes and create a list where the element is 0 if outcome is in the failed set else 1.

↓

Load the list to the dataframe as *Class* column and determine the overall success rate.

Data Wrangling

Set of bad outcomes

```
bad_outcomes
```

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

Final DataFrame containing Class – Launch outcome

```
df.head(5)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

Average Success Rate

```
df["Class"].mean()
```

```
0.6666666666666666
```


GitHub Repository Link: https://github.com/aswiint/SpaceX-Landing-Prediction/blob/main/SpaceX_Data_Wrangling.ipynb

EDA with Data Visualization

The following graphs were plotted to perform Exploratory Data Analysis:

- ❑ Flight Number vs Payload Mass / Launch Site Cat Plot
- ❑ Payload Mass vs Launch Site Scatter Point Chart
- ❑ Success Rate by each Orbit type Bar Plot
- ❑ Orbit vs Flight Number / Payload Mass Scatter Point Chart
- ❑ Year vs Average Success Rate Line Chart

It can be inferred how each important variable would affect the success rate. Only those features that will be used in **Success Prediction** is selected and **One-Hot Encoded** to contain only numerical data

GitHub Repository Link: https://github.com/aswiint/SpaceX-Landing-Prediction/blob/main/SpaceX_EDA_Viz.ipynb

EDA with SQL

The following graphs were plotted to perform Exploratory Data Analysis:

- ❑ Unique launch sites and those starting with “CCA”.
- ❑ Total payload mass launched by NASA (CRS), average payload by F9 v1.1
- ❑ To list the date when the first successful landing outcome in ground pad was achieved.
- ❑ Boosters that carried maximum payload, success in drone ship (4000-6000 kg).
- ❑ Total mission outcomes, ranking mission outcomes between 4/6/10 – 20/3/17
- ❑ Records for months in 2015 with failure in drone ship.

GitHub Repository Link: https://github.com/aswiint/SpaceX-Landing-Prediction/blob/main/SpaceX_EDA_SQL.ipynb

Build an Interactive Map with Folium

A dataframe is loaded with coordinates of each launch sites and a Folium *"Map"* object is created. Highlighted circle area, a popup label and an icon as text label is added for each launch site on the map using *folium.Circle, folium.Popup* and *folium.map.Marker*

Marker clusters is a good way to simplify a map containing many markers having the same coordinate. From the colour labelled markers in marker clusters, it is easy identify which launch sites have relatively high success rates.

The coordinates of railways, highway, coastline and cities are marked to calculate its distance and a *"PolyLine" is drawn* from launch sites.

The map reveals that all launch sites are in proximity to the Equator line and in very close proximity to the coast.

*** NB Viewer Link:** https://nbviewer.org/github/aswiint/SpaceX-Landing-Prediction/blob/main/SpaceX_Folium.ipynb

Build a Dashboard with Plotly Dash

Plotly Dash application allows users to perform interactive visual analytics on SpaceX launch data in real-time.

- ❑ Dropdown menu lets user select different launch sites.
- ❑ Dash callback function is auto called when clicked or selected.
- ❑ Callback function renders success-pie-chart based on selected launch site from site-dropdown.
- ❑ Range slider selects different payload range to identify visual patterns.
- ❑ Callback function renders success-payload-scatter-chart based on selected launch site. This visualizes correlation of payload with mission outcomes and booster version.

GitHub Repository Link: https://github.com/aswiint/SpaceX-Landing-Prediction/blob/main/SpaceX_Dash.py

Predictive Analysis (Classification)

Create a NumPy array from the *"Class"* column and convert it into a Pandas Series.

Standardize the data and split the data into training and testing data using the function train_test_split().The training data is divided into validation data, a second set used for training data; then the models are trained and hyperparameters are selected using the function GridSearchCV

The ML models used for training the data are:

- ❑ Logistic Regression
- ❑ Support Vector Machine
- ❑ Decision Tree Classifier
- ❑ K Nearest Neighbours

Create these model objects and create a GridSearchCV object and fit the object to find the best parameters, accuracy on the validation data and confusion matrix of test data.

GitHub Repository Link: https://github.com/aswiint/SpaceX-Landing-Prediction/blob/main/SpaceX_Predictive_Analysis.ipynb

Results

"Orbit", "LaunchSite", "LandingPad", "Serial“ are important variables and are selected as features for success prediction.

There are 4 launch sites in the dataset:

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

Total Payload Mass carried by **NASA (CRS)** is *45596*.

Average Payload Mass carried by **F9 v1.1** booster is *2534.667*.

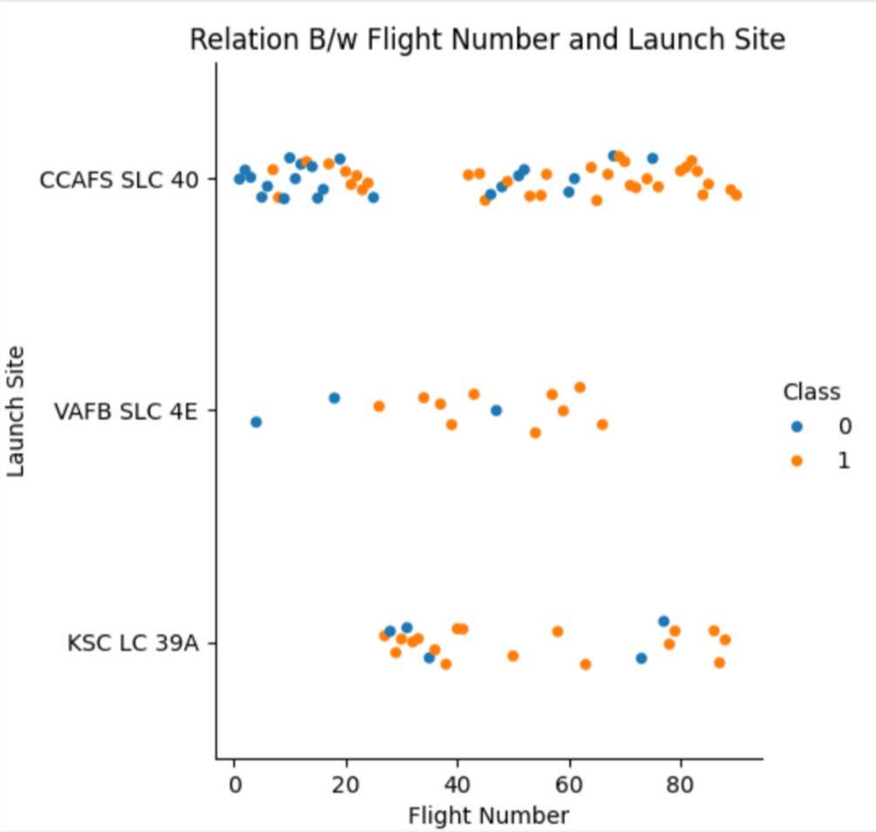
First successful ground landing occurred on *22-02-2015*.

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

```
sns.catplot(data=df, x="FlightNumber", y="LaunchSite", hue="Class")
plt.xlabel("Flight Number")
plt.ylabel("Launch Site")
plt.title ("Relation B/w Flight Number and Launch Site")
plt.show()
```

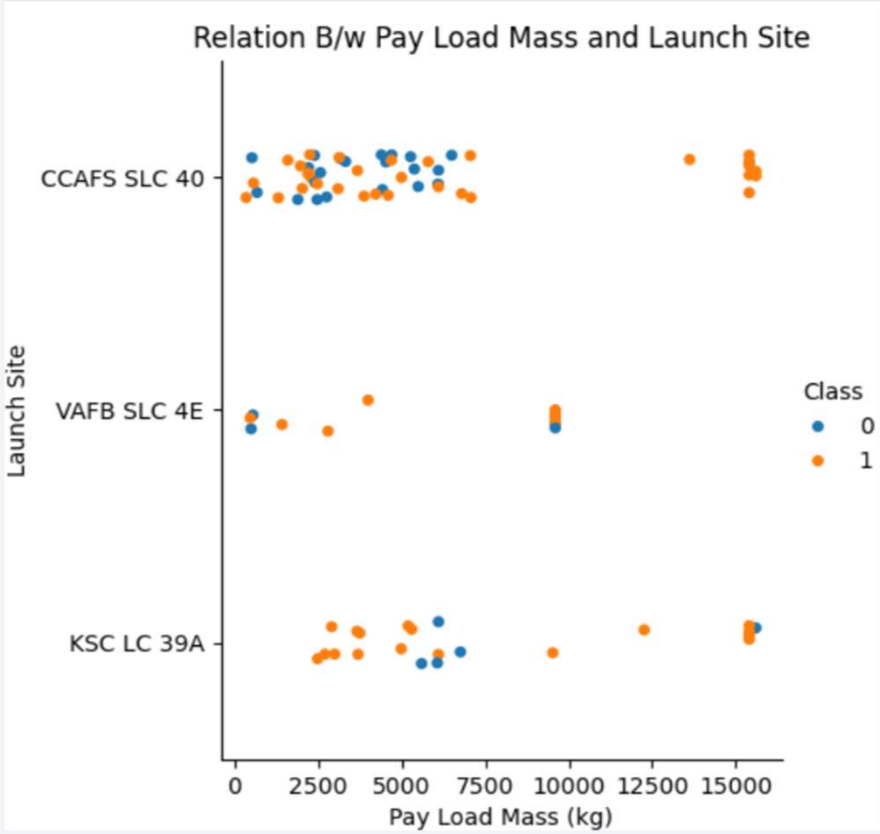


From the graph we can observe the following patterns:

- CCAFS SLC 40 and KSC LC 39A show success landing with many flight launches.
- VAFB SLC 4E has the least total flight numbers as they are the least populated in the plot.
- CCAFS SLC 40 has the most unsuccessful landings of all. They were seen most during initial launches.

Payload vs. Launch Site

```
sns.catplot(data=df, x="PayloadMass", y="LaunchSite", hue="Class")
plt.xlabel("Pay Load Mass (kg)")
plt.ylabel("Launch Site")
plt.title ("Relation B/w Pay Load Mass and Launch Site")
plt.show()
```

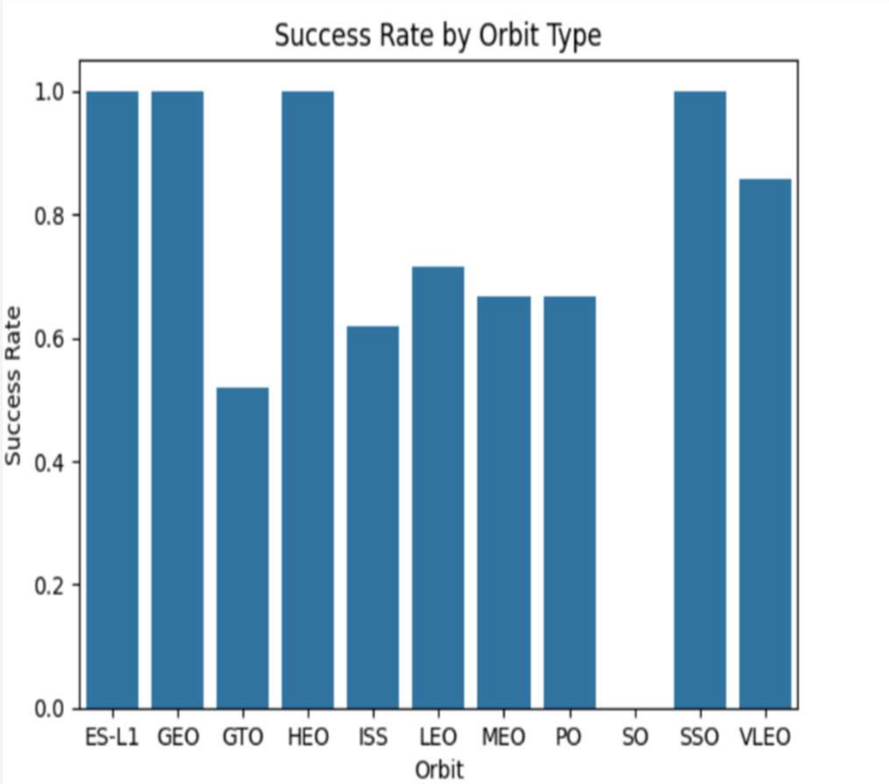


From the graph we can observe the following patterns:

- There is increase in success outcome with increase in payload mass.
- There are no rockets launched at VAFB SLC 4E for heavy payload mass (>10000).
- CCAFS SLC 40 show complete successful landings for heavy payload mass (>10000).

Success Rate vs. Orbit Type

```
success_orbit = pd.DataFrame(df.groupby("Orbit")["Class"].mean().reset_index())
sns.barplot(data = success_orbit, x="Orbit", y="Class")
plt.xlabel("Orbit")
plt.ylabel("Success Rate")
plt.title ("Success Rate by Orbit Type")
plt.show()
```

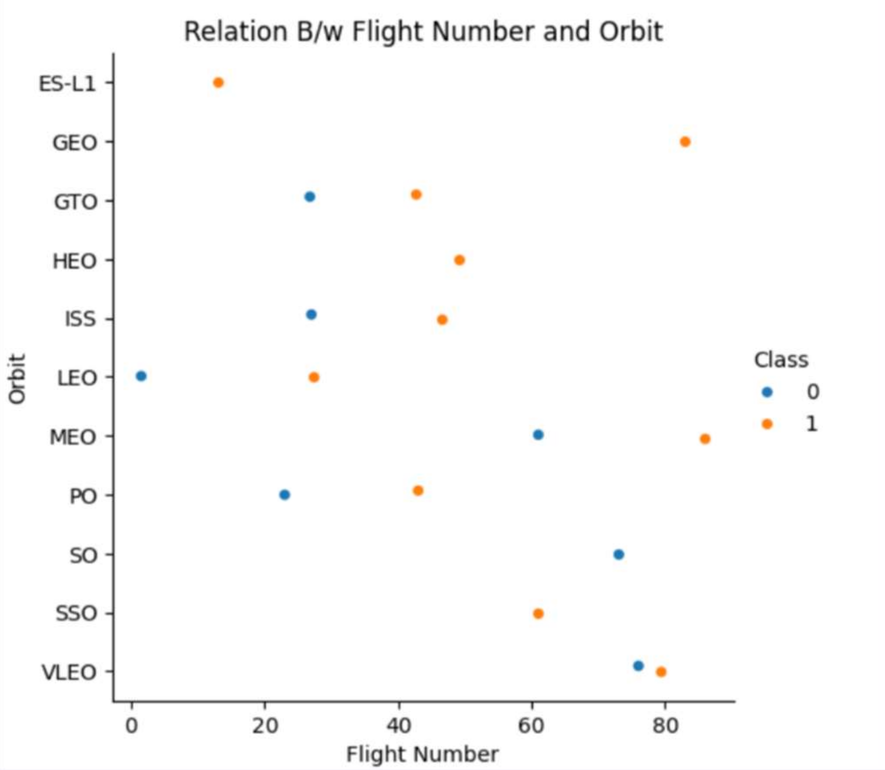


From the graph we can observe the following patterns:

- 4 orbits- ES-L1, GEO, HEO, SSO promise 100% success rate.
- After the top 4 orbits, VLEO followed by LEO have the next highest success rates.
- MEO and PO orbits have similar success rates.
- SO does not provide any outcome.

Flight Number vs. Orbit Type

```
flight_orbit = pd.DataFrame(df.groupby(["Orbit", "Class"])["FlightNumber"]
                            .mean().reset_index())
sns.catplot(data=flight_orbit, x="FlightNumber", y="Orbit", hue="Class")
plt.xlabel("Flight Number")
plt.ylabel("Orbit")
plt.title("Relation B/w Flight Number and Orbit")
plt.show()
```

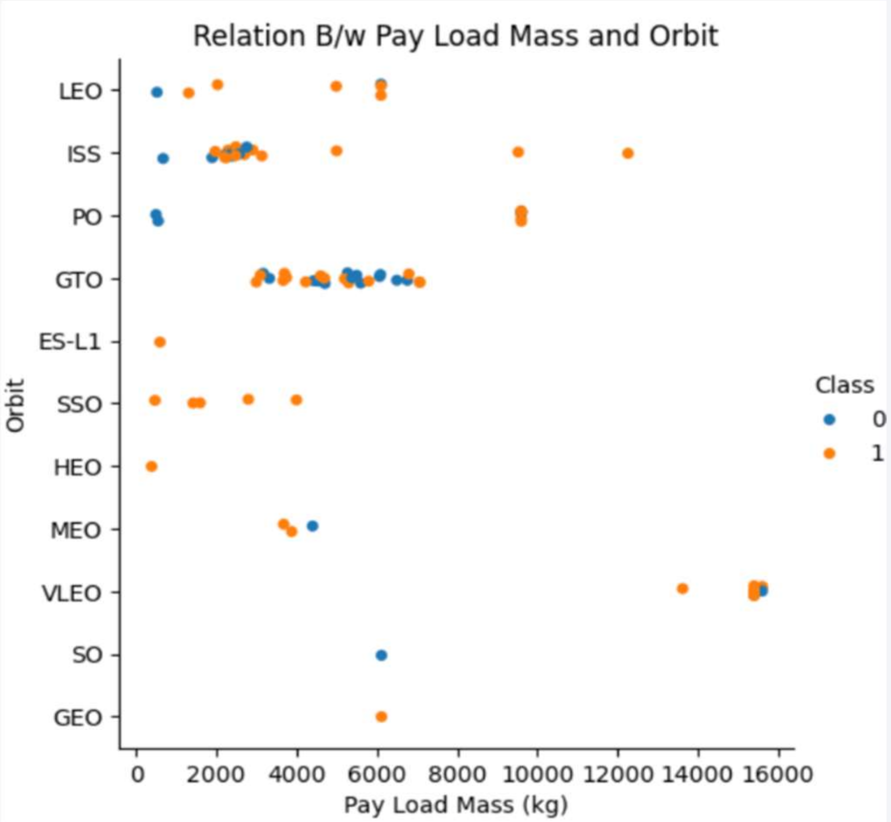


From the graph we can observe the following patterns:

- In the LEO orbit the success appears related to the number of flights.
- In SO orbit, there is no success as flight number increases.
- VLEO orbit does give a clear picture on its relation with flight number.

Payload vs. Orbit Type

```
sns.catplot(data=df, x="PayloadMass", y="Orbit", hue="Class")
plt.xlabel("Pay Load Mass (kg)")
plt.ylabel("Orbit")
plt.title("Relation B/w Pay Load Mass and Orbit")
plt.show()
```

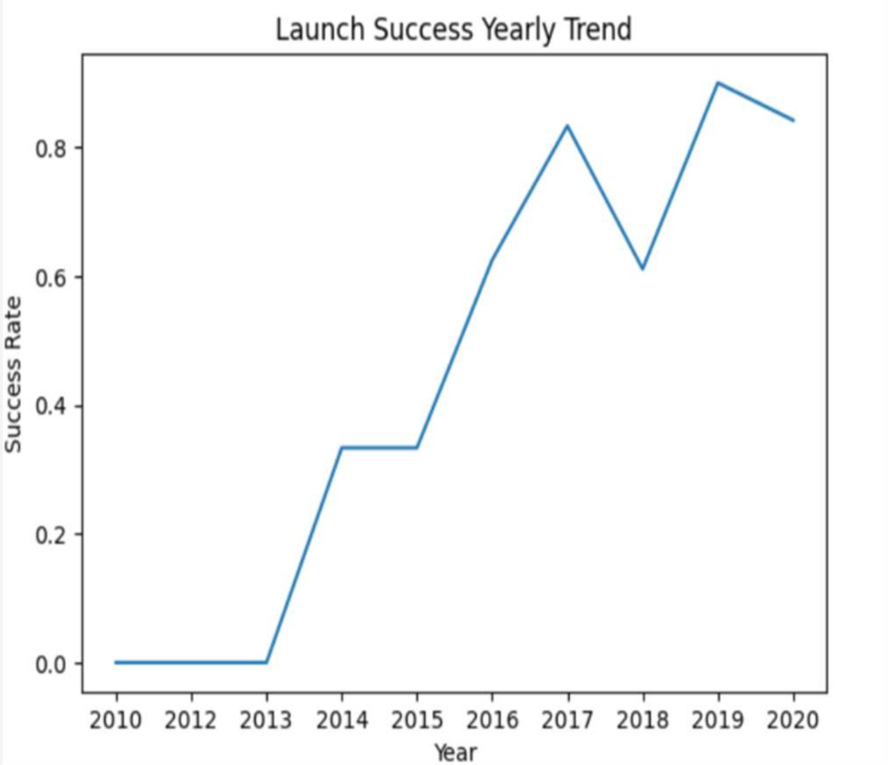


From the graph we can observe the following patterns:

- With heavy payloads, the successful landing or positive landing rate are more for Polar, LEO and ISS.
- Both positive and negative landing are present for GTO, it cannot be distinguished well.
- SO has negative landing rate while GEO has a positive landing rate for the same 6000 kg payload.

Launch Success Yearly Trend

```
yearly_trend = pd.DataFrame(df.groupby("Date")["Class"].mean().reset_index())
sns.lineplot(data=yearly_trend, x="Date", y="Class")
plt.xlabel("Year")
plt.ylabel("Success Rate")
plt.title("Launch Success Yearly Trend")
plt.show()
```



From the graph we can observe the following patterns:

- Between 2010-2013 the line shows constant negative rate. Success rate increased since 2013 until 2020.
- 2017-2018 experienced 20% of decrease in success rate.
- 2019 has the highest positive success rate reaching upto 80%.

All Launch Site Names

```
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTBL
```

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

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

SELECT DISTINCT "Launch_Site" FROM SPACEXTBL

SELECT DISTINCT statement returns unique records in the column "*Launch_Site*" from the table **SPACEXTBL**

Launch Site Names Begin with 'CCA'

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

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

SELECT * FROM SPACEXTBL WHERE “Launch_Site” LIKE “CCA%” LIMIT 5

SELECT * statement returns all records from the table **SPACEXTBL WHERE** “*Launch_Site*” begins with the letters “*CCA*”.

“%” in **“CCA%”** represents one or multiple characters in a string.

Total Payload Mass

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass By NASA(CRS)" FROM SPACEXTBL WHERE "Customer" = "NASA (CRS)"
```

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

Total Payload Mass By NASA(CRS)
45596

SELECT SUM(PAYLOAD_MASS_KG) AS "Total Payload Mass By NASA(CRS)" FROM SPACEXTBL WHERE "Customer" = "NASA (CRS)"

SUM(PAYLOAD_MASS_KG) returns the total sum of records in payload column that have *"Customer"* as **NASA (CRS)** from the table **SPACEXTBL**

The result is displayed in a column named **"Total Payload Mass By NASA(CRS)"**

Average Payload Mass by F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass By F9 v1.1" FROM SPACEXTBL WHERE "Booster_Version" LIKE "F9 v1.1%"
```

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

Average Payload Mass By F9 v1.1
2534.6666666666665

SELECT AVG(PAYLOAD_MASS_KG) AS "Average Payload Mass By F9 v1.1" FROM SPACEXTBL WHERE "Booster_Version" LIKE "F9 v1.1%"

AVG(PAYLOAD_MASS_KG) returns the mean of records in payload column that have *"Booster_Version"* **LIKE "F9 v1.1%"** from the table **SPACEXTBL**

"%" in **"F9 v1.1%"** represents one or multiple characters in a string.

First Successful Ground Landing Date

```
%sql SELECT MIN("Date") AS "First Successful Landing Date" FROM SPACEXTBL WHERE "Landing_Outcome" LIKE "Success (ground pad)"
```

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

First Successful Landing Date
2015-12-22

SELECT MIN("Date") AS "First Successful Landing Date" FROM SPACEXTBL WHERE "Landing_Outcome" LIKE "Success (ground pad)“

MIN("Date") returns the minimum/smallest date in the column *"Date"* from the table **SPACEXTBL.** This date is the earliest date in the column.

SPACEXTBL is filtered to include only those records that have *"Landing_Outcome"* **LIKE "Success (ground pad)"**

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT "Booster_Version" FROM SPACEXTBL WHERE "Landing_Outcome" LIKE "Success (drone ship)" AND ("PAYLOAD_MASS__KG_">4000 AND "PAYLOAD_MASS__KG_"<600
```

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

Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

SELECT ”Booster_Version” FROM SPACEXTBL WHERE “Landing_Outcome” LIKE “Success (drone ship)” AND (“PAYLOAD_MASS__KG_” > 4000 AND “PAYLOAD_MASS__KG_” < 6000)

SELECT *“Booster_Version”* statement returns all booster version from the table **SPACEXTBL WHERE** “*Launch_Site*” is **“Success (drone ship)”**

<, > operator filters *“Payload_Mass__KG_”* between 4000 and 6000.

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT "Mission_Outcome" As "Mission Outcome", COUNT(*) AS "Count" FROM SPACEXTBL GROUP BY "Mission_Outcome"
```

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

Mission Outcome	Count
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

SELECT “Mission_Outcome” AS “Mission Outcome”, COUNT(*) AS “Count” FROM SPACEXTBL GROUP BY “Mission_Outcome”

GROUP BY *“Mission_Outcome”* groups together all distinct mission outcomes from the table **SPACEXTBL** and **COUNT(*) AS “Count”** returns the individual count of every distinct missions.

Boosters Carried Maximum Payload

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

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

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

SELECT "Booster_Version" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL

SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL returns maximum payload value from the table SPACEX.

SELECT "Booster_Version" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = subquery returns only those records whose payload is equal to the maximum payload obtained from subquery.

2015 Launch Records

```
%sql SELECT SUBSTR("Date", 6, 2) AS "Month", "Landing_Outcome", "Booster_Version", "Launch_Site"  FROM SPACEXTBL WHERE "Landing_Outcome" = "Failure (drone ship)" AND "Date" LIKE "2015%"
```

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

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

SELECT SUBSTR("Date", 6, 2) AS "Month", "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTBL WHERE "Landing_Outcome" = "Failure (drone ship) AND "Date" LIKE "2015%"

SUBSTR("Date", 6, 2) returns 2 characters from the 6th character of every record in the column *"Date"* with **"2015"** year and *"Landing_Outcome"* as **"Failure (drone ship)"** from the table **SPACEXTBL**

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

```
%sql SELECT "Landing_Outcome", COUNT("Landing_Outcome") AS "Count" FROM SPACEXTBL GROUP BY "Landing_Outcome" HAVING "Date" BETWEEN "2010-06-04" AND "2017-03-20" ORDER BY "Count" DESC
```

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

Landing_Outcome	Count
No attempt	21
Success (drone ship)	14
Success (ground pad)	9
Failure (drone ship)	5
Controlled (ocean)	5
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

SELECT "Landing_Outcome", COUNT("Landing_Outcome") AS "Count" FROM SPACEXTBL GROUP BY "Landing_Outcome" HAVING "Date" BETWEEN "2010-06-04" AND "2017-03-20" ORDER BY "Count" DESC

GROUP BY *"Landing_Outcome"* groups together all distinct landing outcomes from the table **SPACEXTBL** and **COUNT ("Landing_Outcome")** returns the individual count of every landing missions.

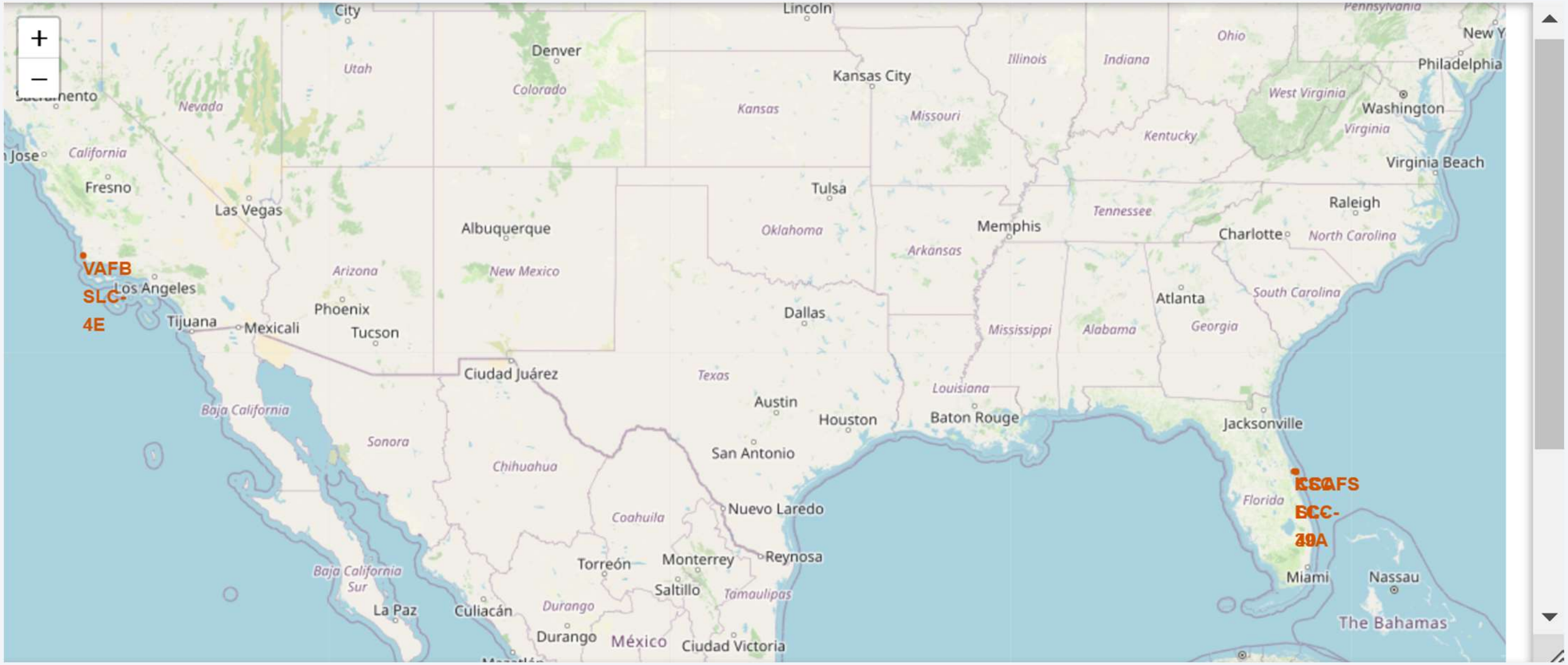
HAVING *"Date"* **BETWEEN "2010-06-04" AND "2017-03-20"** filters the record between these dates after they have been grouped based on landing outcomes.

ORDER BY *"Count"* **DESC** ranks the retrieved records in descending order of their count of landing outcomes.

Section 3

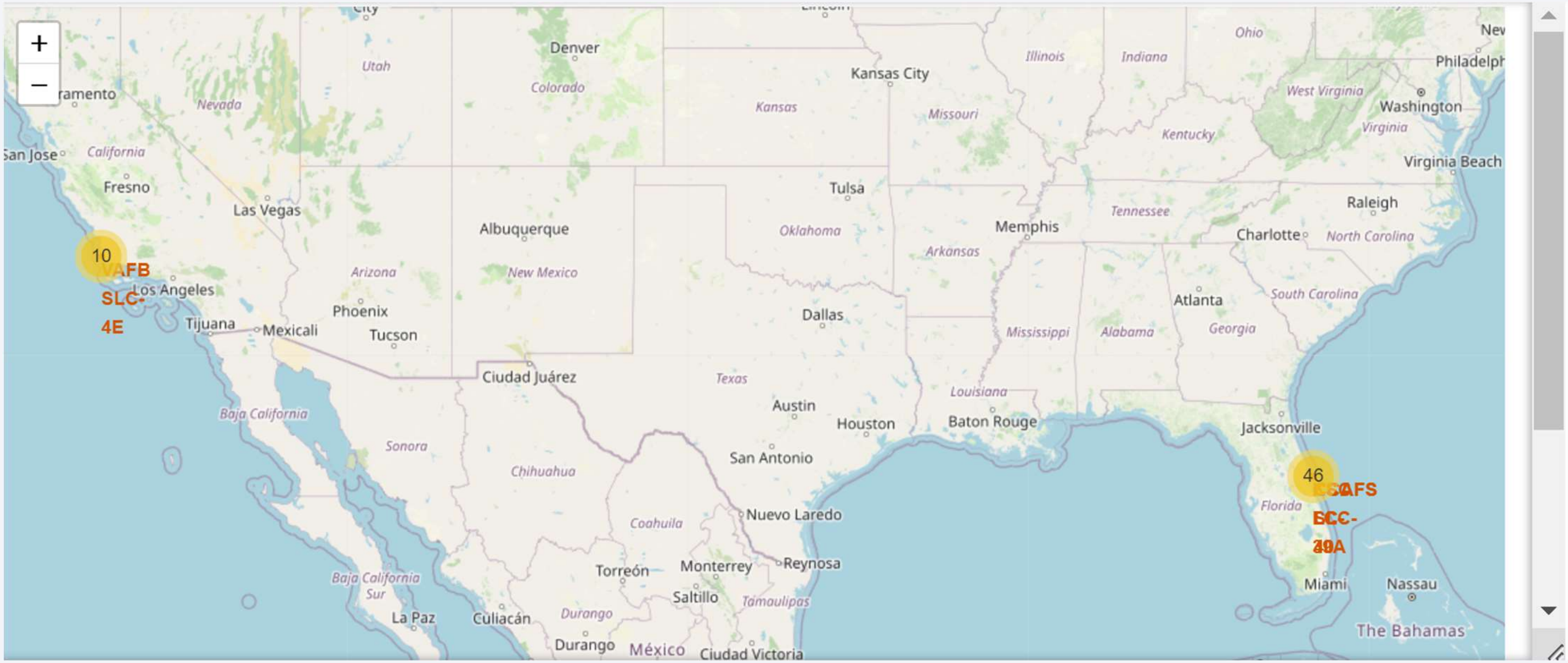
Launch Sites Proximities Analysis

All Launch Sites



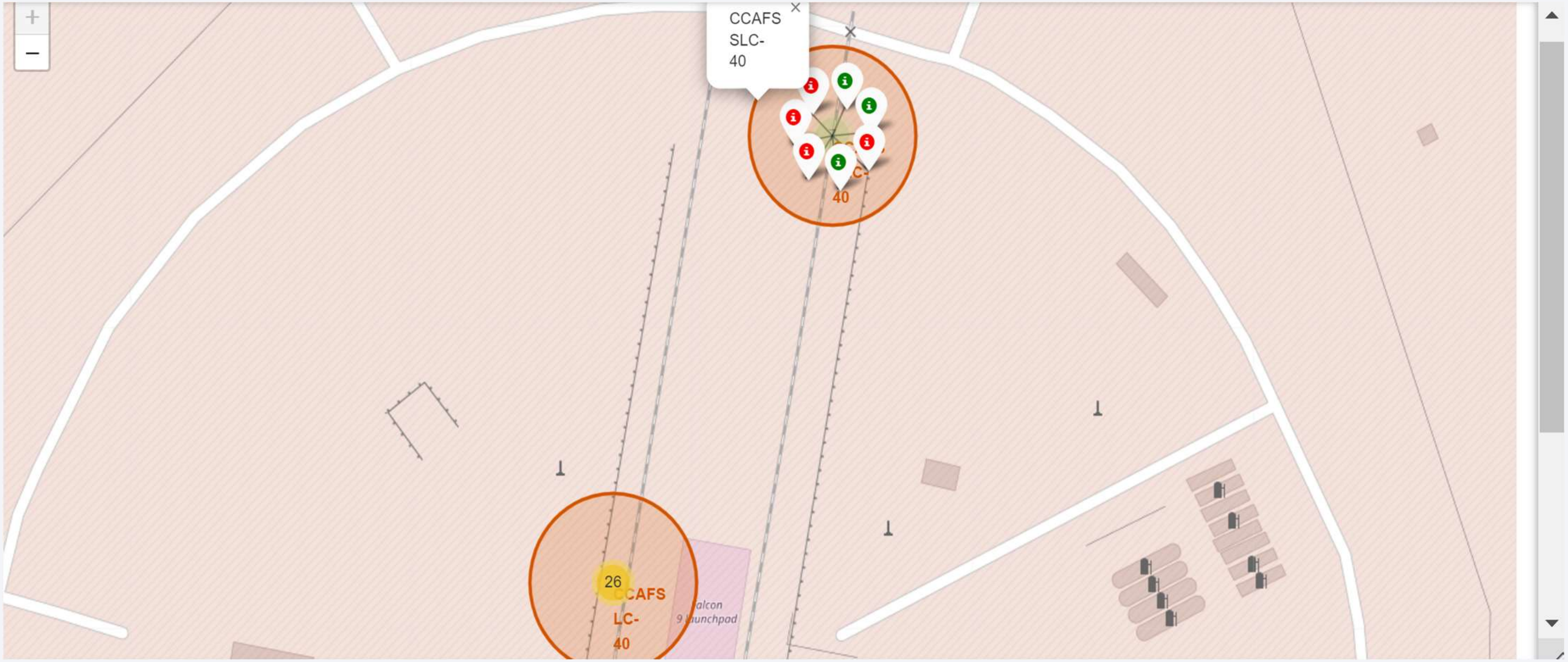
VAFB SLC-4E is located in *California* while **KSCLC-39A, CCAFS SLC-40/LC–40** are located in *Florida*.

Success/Failed Launches For Each Site



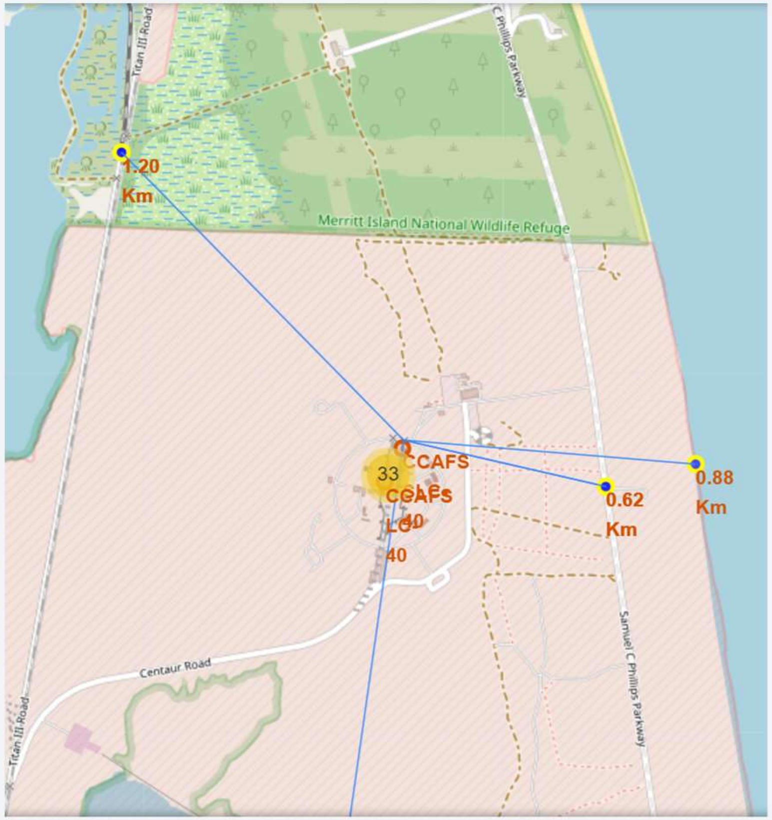
There were *10* launches at **VAFB SLC-4E** and *46* at **KSCLC-39A, CCAFS SLC-40/LC–40**

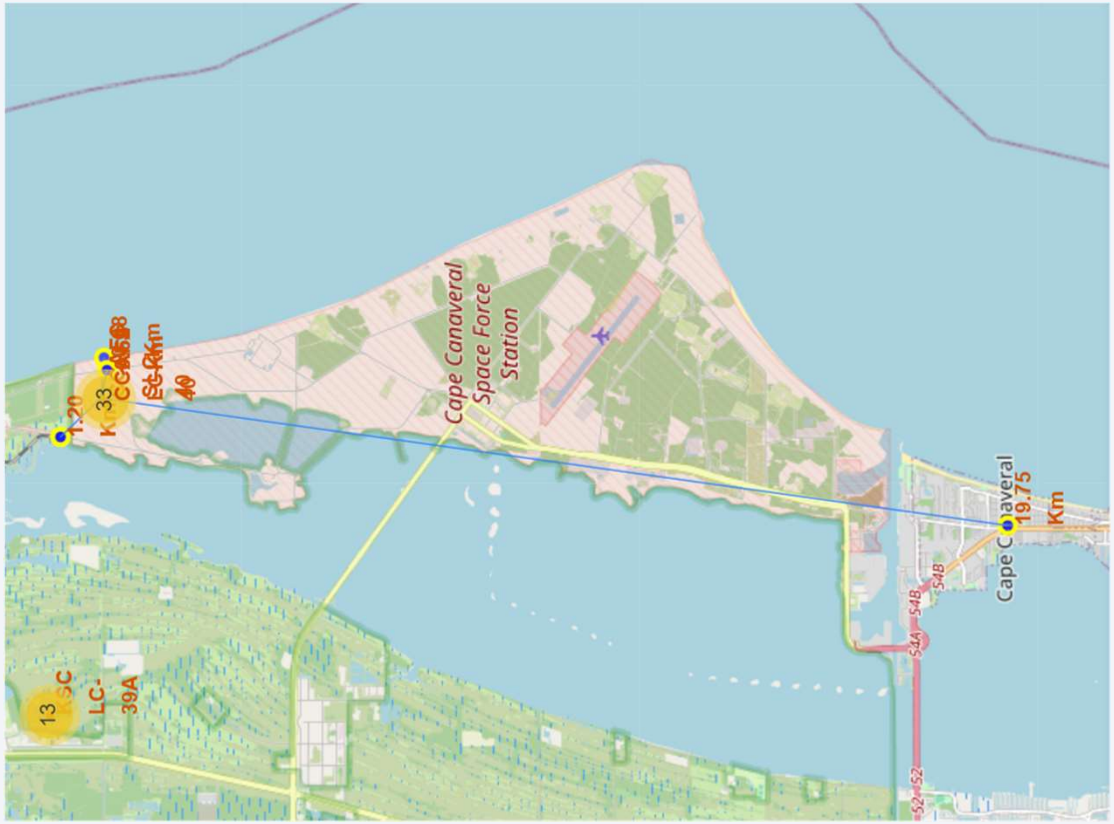
Success/Failed Launches For Each Site



7 were launched at **CCAFS SLC-40** of which *3* succeeded and *4* failed.

Launch Site CCAFS SLC-40 and its Proximities





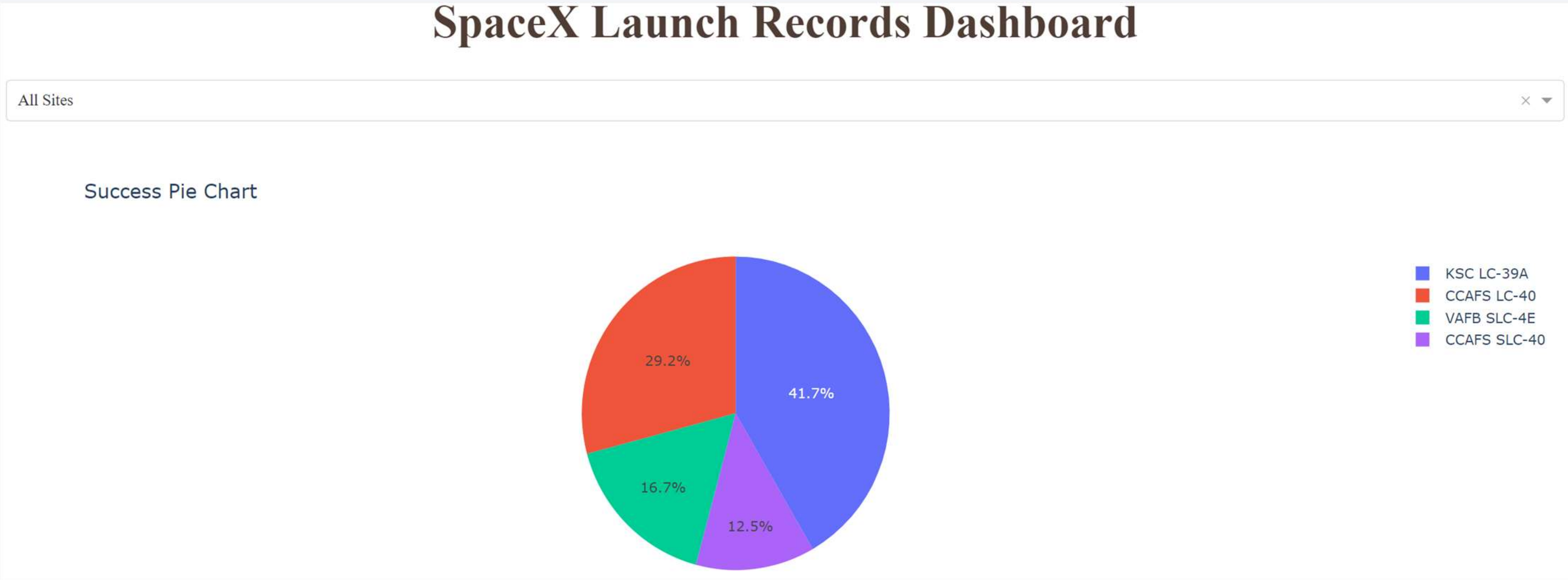
Nearest Coastline: *0.88 Km*
Nearest Rail line: NASA Rail Road, *1.20 Km*

Nearest Highway: Samuel Phillips Parkway, *0.62 Km*
Nearest City: Cape Canaveral, *19.75 Km*

Section 4

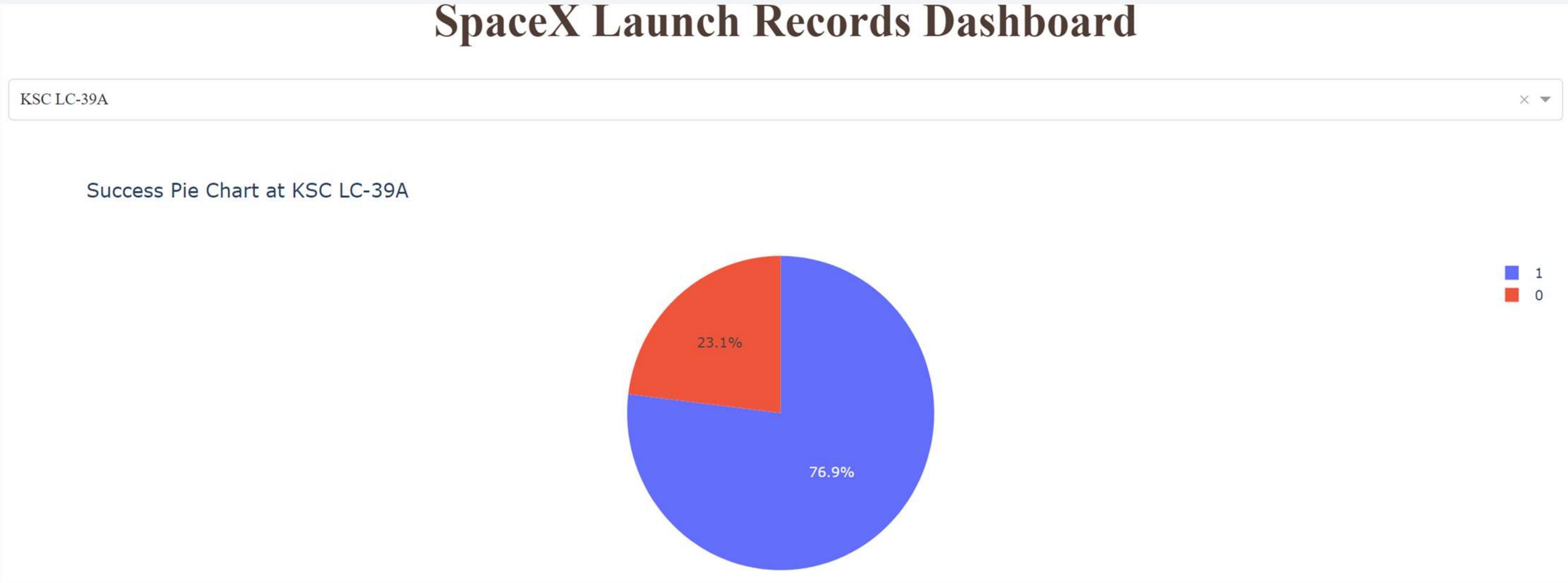
Build a Dashboard with Plotly Dash

Success Pie Chart for All Launch Sites



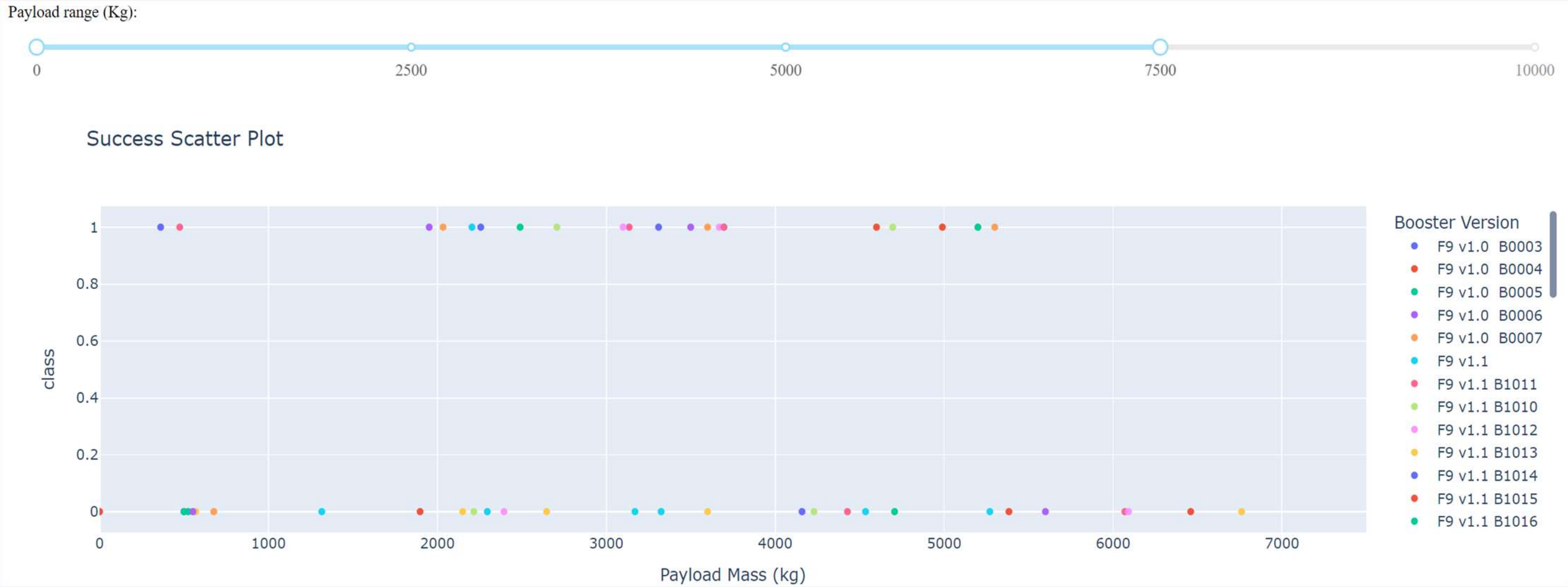
KSCLC-39A has the highest success rate of *41.7%* followed by **CCAFS LC-40** *29.2%.* **CCAFS SLC-40** has the lowest success rate of *12.3%*

Launch Site with Highest Success



KSCLC-39A has the highest success rate. *76.9%* of its launches landed successfully while *23.1%* of them failed.

Payload vs. Launch Outcome



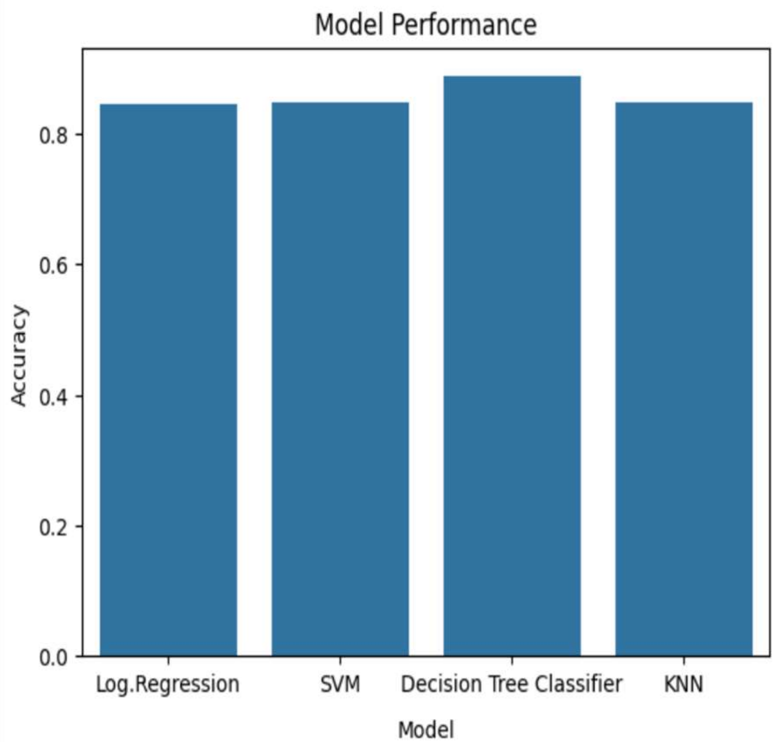
F9 v1.1 B1010 has the most success launches. **F9 v1.1** has the most failed launches for payload in range 0-6000 Kg.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

```
accuracy_df = pd.DataFrame({"Models": ["Log.Regression", "SVM", "Decision Tree Classifier", "KNN"],
                            "Accuracy": [logreg_cv.best_score_, svm_cv.best_score_, tree_cv.best_score_, knn_cv.best_score_]})
sns.barplot(data=accuracy_df, x="Models", y="Accuracy")
plt.xlabel("Model", labelpad=10)
plt.title("Model Performance")
plt.show()
```

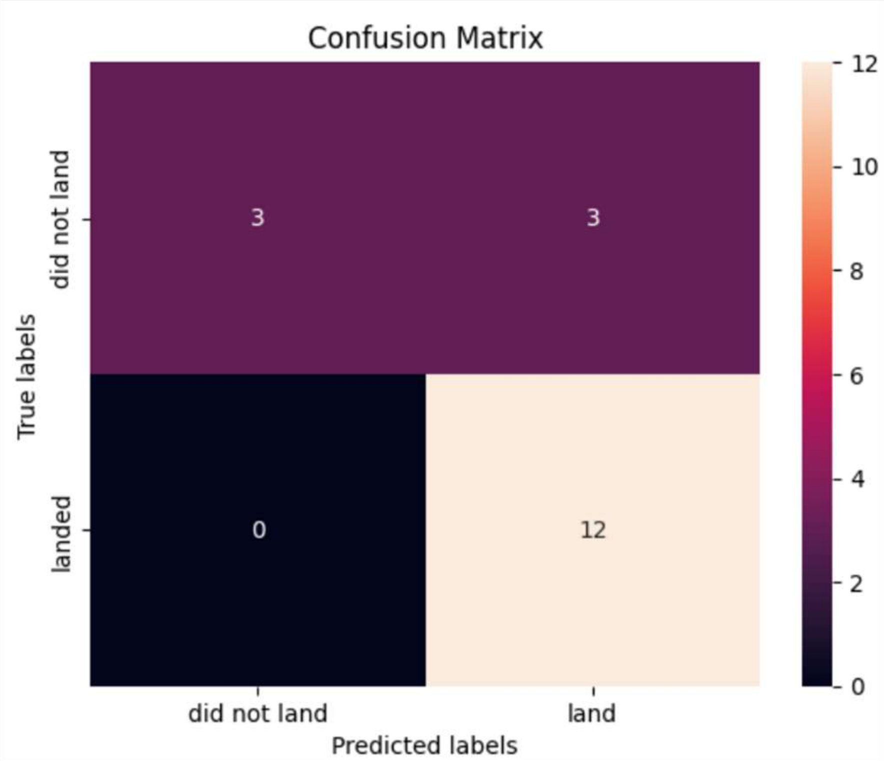


Decision Tree Classifier performed best with accuracy of *0.8875*

Logistic Regression, Support Vector Machine, K-Nearest Neighbours all performed at an equal accuracy of *0.84*

Decision Tree Classifier Confusion Matrix

```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



True Positive: 3
False Negative: 3
True Negative: 12
False Positive: 0

Precision = 1
TP/(TP+FP)

Recall = 0.5
TP/(TP+FN)

Accuracy = 0.833
TP+TN/(TP+FP+TN+FN)

F1 Score = 0.667
2*(Precision*Recall)/(Precision+Recall)

Conclusions

SpaceX should consider following cases for successful landing:

- ❑ Falcon9 launches from **KSCLC -39A** launch site.
- ❑ First Stage lands on a **drone ship**.
- ❑ Falcon9 launches into **LEO, Polar or ISS** orbits.
- ❑ Falcon9 launches into **LEO** orbit when carrying larger payload.
- ❑ **F9 B5 B10** booster versions is used for carrying larger payload.

On successful landing Flacon9 can reuse the First Stage and thereby reduce cost.

Appendix

Falcon9 launch records was Webscraped from the following Wikipedia page:
https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

2020 [edit]

In late 2019, Gwynne Shotwell stated that SpaceX hoped for as many as 24 launches for Starlink satellites in 2020,[490] in addition to 14 or 15 non-Starlink launches. At 26 launches, 13 of which for Starlink satellites, Falcon 9 had its most prolific year, and Falcon rockets were second most prolific rocket family of 2020, only behind China's Long March rocket family.[491]

[hide] Flight No.	Date and time (UTC)	Version, Booster[b]	Launch site	Payload[c]	Payload mass	Orbit	Customer	Launch outcome	Booster landing
78	7 January 2020, 02:19:21[492]	F9 B5 ♺ B1049.4	CCAFS, SLC-40	Starlink 2 v1.0 (60 satellites)	15,600 kg (34,400 lb)[5]	LEO	SpaceX	Success	Success (drone ship)
	Third large batch and second operational flight of Starlink constellation. One of the 60 satellites included a test coating to make the satellite less reflective, and thus less likely to interfere with ground-based astronomical observations.[493]								
79	19 January 2020, 15:30[494]	F9 B5 ♺ B1046.4	KSC, LC-39A	Crew Dragon in-flight abort test[495] (Dragon C205.1)	12,050 kg (26,570 lb)	Sub-orbital[496]	NASA (CTS)[497]	Success	No attempt
	An atmospheric test of the Dragon 2 abort system after Max Q. The capsule fired its SuperDraco engines, reached an apogee of 40 km (25 mi), deployed parachutes after reentry, and splashed down in the ocean 31 km (19 mi) downrange from the launch site. The test was previously slated to be accomplished with the Crew Dragon Demo-1 capsule;[498] but that test article exploded during a ground test of SuperDraco engines on 20 April 2019.[419] The abort test used the capsule originally intended for the first crewed flight.[499] As expected, the booster was destroyed by aerodynamic forces after the capsule aborted.[500] First flight of a Falcon 9 with only one functional stage — the second stage had a mass simulator in place of its engine.								
80	29 January 2020, 14:07[501]	F9 B5 ♺ B1051.3	CCAFS, SLC-40	Starlink 3 v1.0 (60 satellites)	15,600 kg (34,400 lb)[5]	LEO	SpaceX	Success	Success (drone ship)
	Third operational and fourth large batch of Starlink satellites, deployed in a circular 290 km (180 mi) orbit. One of the fairing halves was caught, while the other was fished out of the ocean.[502]								
81	17 February 2020, 15:05[503]	F9 B5 ♺ B1056.4	CCAFS, SLC-40	Starlink 4 v1.0 (60 satellites)	15,600 kg (34,400 lb)[5]	LEO	SpaceX	Success	Failure (drone ship)

*** Maps in Jupyter Notebook cannot be viewed in GitHub. NB Viewer allows interaction with maps in Jupyter Notebook.**


Thank you!