Winning Space Race with Data Science

Antonio Villar
February 2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection
  - Data Wrangling
  - EDA with Data Visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotty Dash
  - Predictive Analysis

- Summary of all results
  - EDA
  - Interactive Analysis
  - Predictive Analysis

Introduction

- Project background and context
  - SpaceX Falcon 9 costs 62 million dollars while others go over 150 millions
  - The reutilization of first stage is the reason to this amount of savings
  - Instead of using rocket science to determine if the first stage will land successfully, we will train a machine learning model and use public information to predict if SpaceX will reuse the first stage.
  - We will do this by gathering information about Space X and creating dashboards.

- Problems you want to find answers
  - Determine if SpaceX will reuse the first stage.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - We used data from URLs available related to SpaceX launch.
- Perform data wrangling
  - Multiple actions were done to remove unnecessary data and collecting relevant data not present in the initial batch.
- Perform exploratory data analysis (EDA) using visualization and SQL
  - SQL allowed us to better understand the data and interact with them
- Perform interactive visual analytics using Folium and Plotly Dash
  - Maps and dashboards were created to better understand the analysis
- Perform predictive analysis using classification models
  - Different simulations and models were tested to show the best fit result

Data Collection

- We will be working with SpaceX launch data that is gathered from an API, specifically the SpaceX REST API.

- This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.

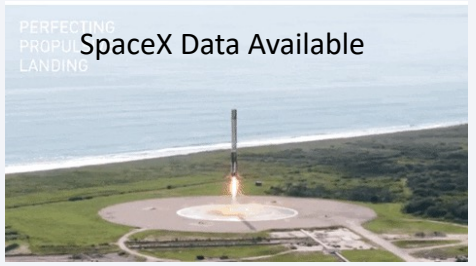
Data Collection – SpaceX API

Retrieving data from SpaceX URL

response = requests.get(https://api.spacexdata.com/v4/launches/past)

Output:

3T05:00:00.000Z","static_fire_date_unix":1620018000,"net":false,"window":0,"rocket":"5e9d0d95eda69973a809d1ec","success":true,"failures":[],"details":"This mission launches the 25th batch of operational Starlink satellites, which are version 1.0, from LC-39A. It is the 26th Starlink launch overall. The satellites will be delivered to low Earth orbit and will spend a few weeks maneuvering to their operational altitude. The booster is expected to land on



Get Data from URL using API

3T05:00:00.000Z","static_fire_date_unix":1620018000,"net":false,"window":0,"rocket":"5e9d0d95eda69973a809d1ec","success":true,"failures":[],"details":"This mission launches the 25th batch of operational Starlink satellites, which are version 1.0, from LC-39A. It is the 26th Starlink launch overall

Initial data comes with a format difficult to understand

- https://github.com/aeRibeiro/CapstoneFinal/blob/main/jupyter-labs-spacex-data-collection-api%20(1).ipynb

Data Collection - Scraping

As seen before we need to add some steps to make the data easier to work

Data Normalization

JSON normalize method used to convert results into a DataFrame

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

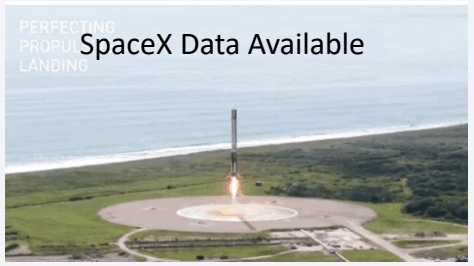
More work was done to retrieve names from data received only with codes like [['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

```
response = requests.get("https://api.spacexdata.com/v4/rockets/
response = requests.get(https://api.spacexdata.com/v4/launchpads/
…
```

Finally, we will remove the Falcon 1 launches keeping only the Falcon 9 launches.

```
data_falcon9 = data[data['BoosterVersion']!='Falcon 1']
```

https://github.com/aeRibeiro/CapstoneFinal/blob/main/jupyter-labs-webscraping.ipynb



Get Data from URL using API

3T05:00:00.000Z","static_fire_date_unix":1620018000,"net":false,"window":0,"rocket":"5e9d0d95eda69973a809d1ec","success":true,"failures":[],"details":"This mission launches the 25th batch of operational Starlink satellites, which are version 1.0, from LC-39A. It is the 26th Starlink launch overall

Initial data comes with a format difficult to understand

Data Normalization

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing
0	333	CCAFS	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure
1	344	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Easy format to read and with relevant feature

Data Wrangling

Our goal is to find some patterns in the data and determine what would be the label for training supervised models.

We will mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful. (class created with 0 or 1)

	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

True (1) means the stage1 successful landed

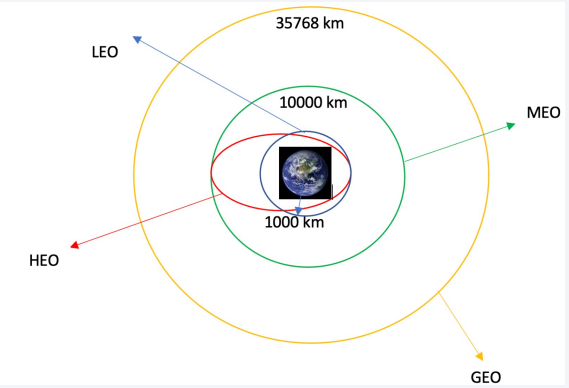
```
0 ('True ASDS',)
1 ('None None',)
2 ('True RTLS',)
3 ('False ASDS',)
4 ('True Ocean',)
5 ('False Ocean',)
6 ('None ASDS',)
7 ('False RTLS',)
```

This phase we found missing values to be replaced or eliminated as well as understand the meaning of some features like orbits, launch sites, successful landing etc.

df.isnull().sum()/len(df)*100

```
FlightNumber      0.000000
Date              0.000000
BoosterVersion    0.000000
PayloadMass       0.000000
Orbit             0.000000
LaunchSite        0.000000
Outcome           0.000000
Flights           0.000000
GridFins          0.000000
Reused            0.000000
Legs              0.000000
LandingPad       28.888889
Block             0.000000
ReusedCount       0.000000
Serial            0.000000
Longitude         0.000000
Latitude          0.000000
```

Rows missing values need to be replaced by the mean value. We found LandingPad and Payload mass with missing values



The understanding of different types of orbits and payload are key to our model

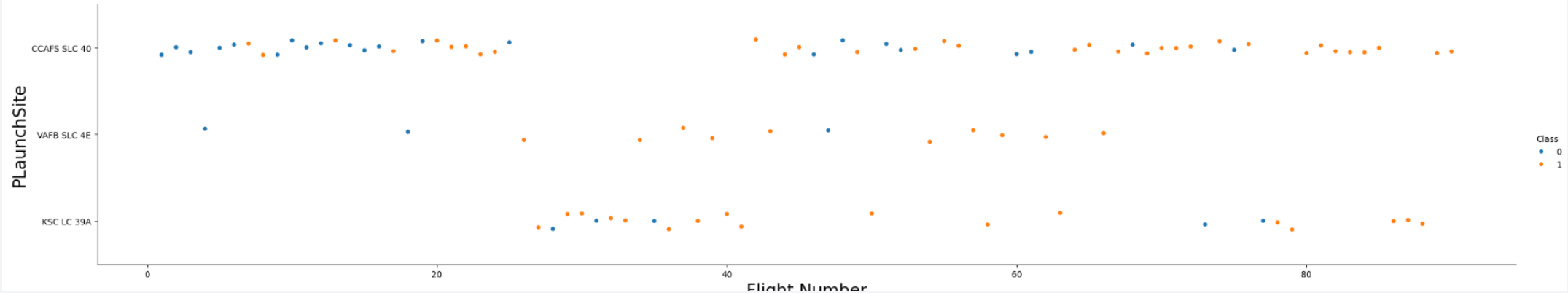
```
[8]: Orbit
     GTO      27
     ISS      21
     VLEO     14
     PO        9
     LEO       7
     SSO       5
     MEO       3
     ES-L1     1
     GEO       1
     HEO       1
     SO        1
```

https://github.com/aeRibeiro/CapstoneFinal/blob/main/labs-jupyter-spacex-Data%20wrangling%20(2).ipynb

EDA with Data Visualization

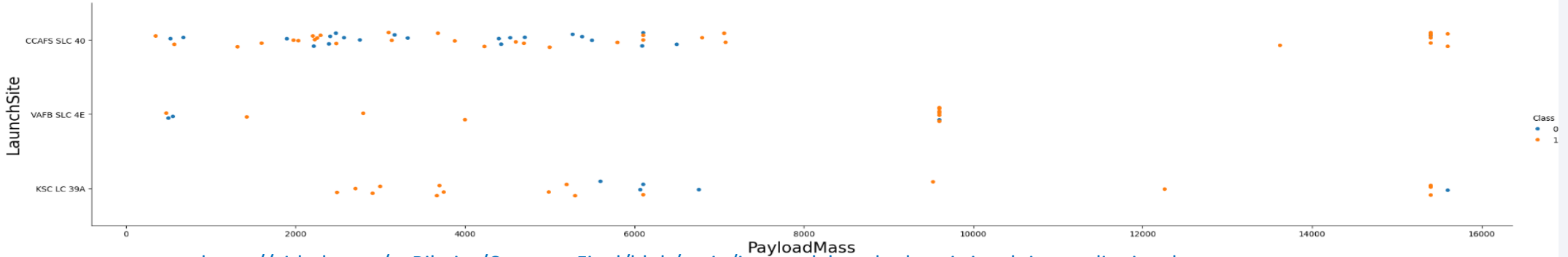
We see that different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.

```
### 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("PLaunchSite",fontsize=20)
plt.show()
```



We also want to observe if there is any relationship between launch sites and their payload mass.

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```

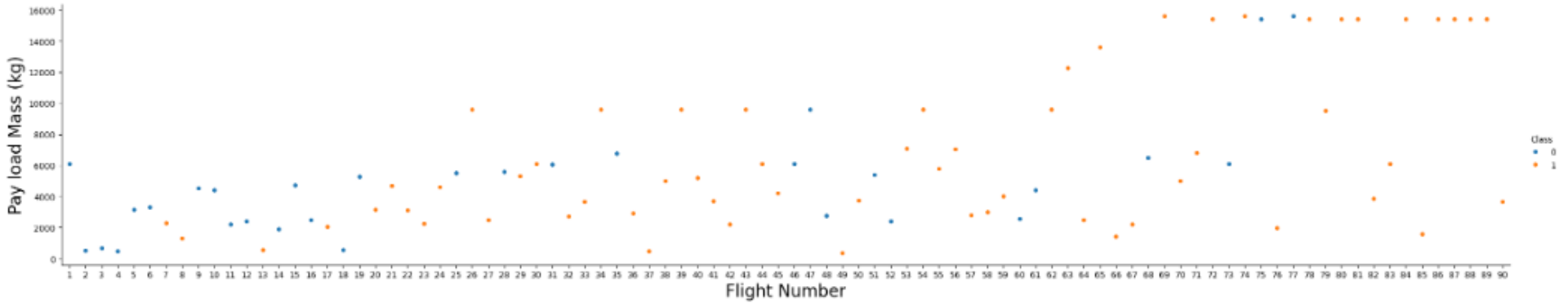


https://github.com/aeRibeiro/CapstoneFinal/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with Data Visualization

- FlightNumber vs. PayloadMass and overlay the outcome of the launch. We see that as the flight number increases, the first stage is more likely to land successfully. The payload mass is also important; it seems the more massive the payload, the less likely the first stage will return.

```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```



https://github.com/aeRibeiro/CapstoneFinal/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

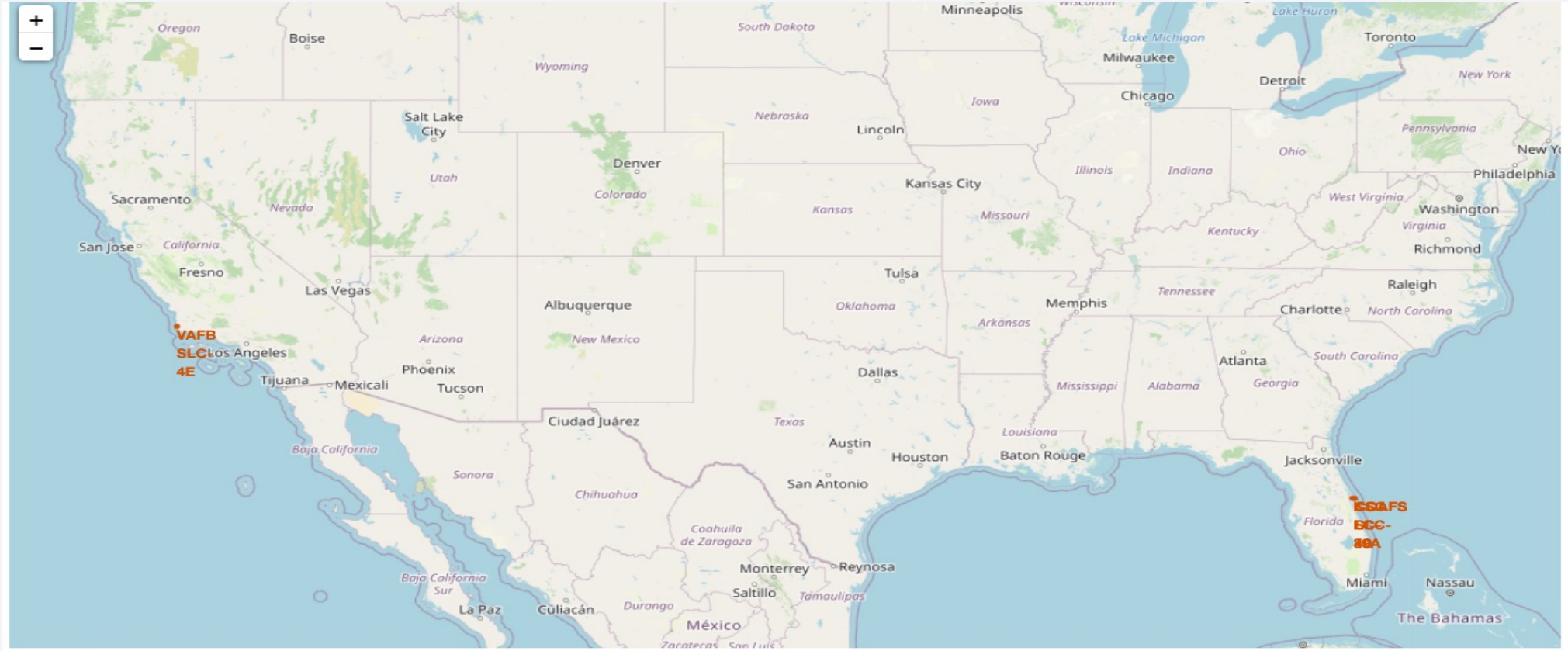
EDA with SQL

SQL queries performed

- SELECT DISTINCT Launch_Site from SPACEXTABLE
- SELECT * from SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5
- SELECT Customer, SUM([PAYLOAD_MASS__KG_]) from SPACEXTABLE WHERE Customer like 'NASA (CRS)'
- SELECT Customer, AVG([PAYLOAD_MASS__KG_]) from SPACEXTABLE WHERE Customer like 'NASA (CRS)'
- SELECT MIN(Date) from SPACEXTABLE WHERE MIssion_Outcome like 'Success'
- SELECT DISTINCT Booster_Version, PAYLOAD_MASS__KG_, Mission_Outcome from SPACEXTABLE WHERE ((PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000) AND Mission_Outcome = "Success")
- SELECT Mission_Outcome, COUNT(Mission_Outcome) from SPACEXTABLE GROUP BY Mission_Outcome
- SELECT Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)

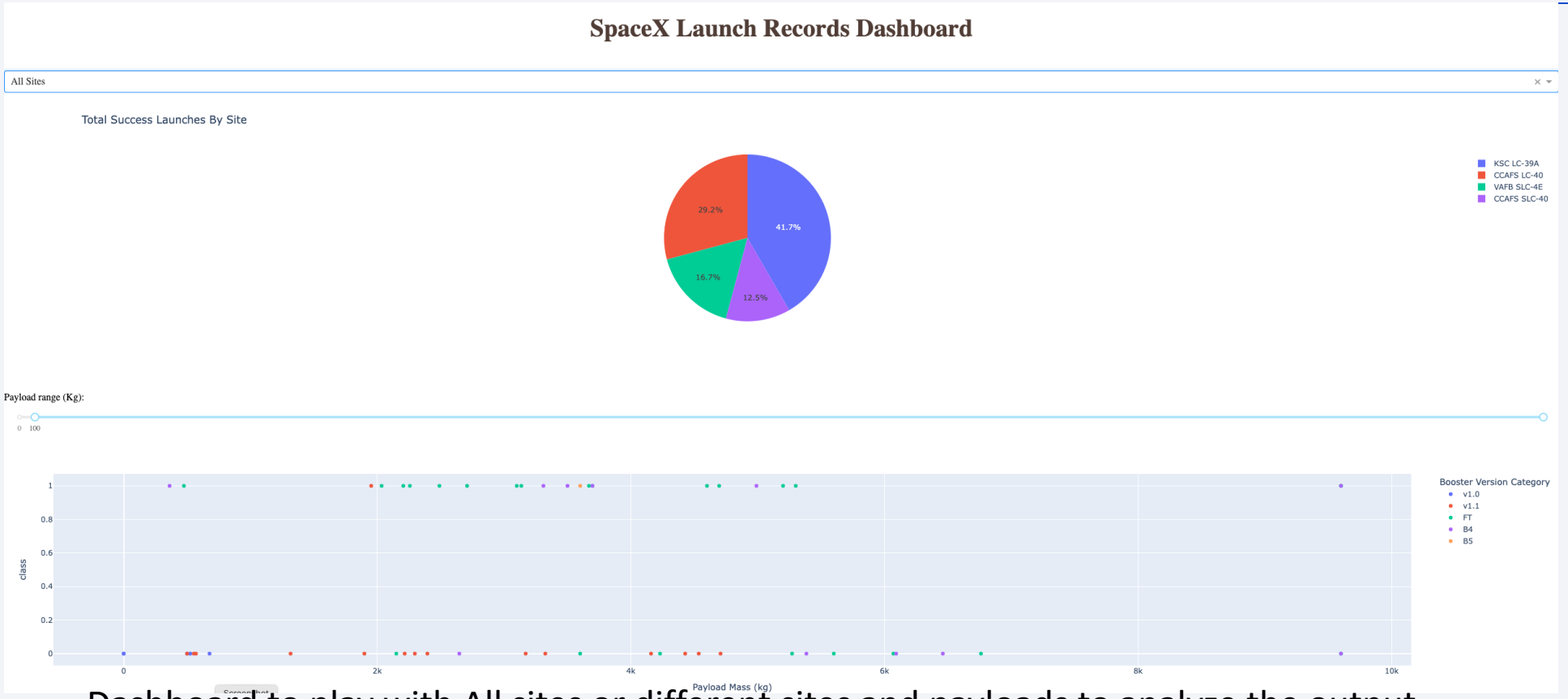
- https://github.com/aeRibeiro/CapstoneFinal/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium



https://github.com/aeRibeiro/CapstoneFinal/blob/main/lab_jupyter_launch_site_location.jupyterlite%20(3).ipynb

Build a Dashboard with Plotly Dash



Dashboard to play with All sites or different sites and payloads to analyze the output

https://github.com/aeRibeiro/CapstoneFinal/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- Create a training and Test Data Set where the class feature informing if the return of the first stage was successful and used as dependent variable
- Decide which model is the best for us based on best score

Logistic Regression

Support Vector Machine

Decision Tree Classifier

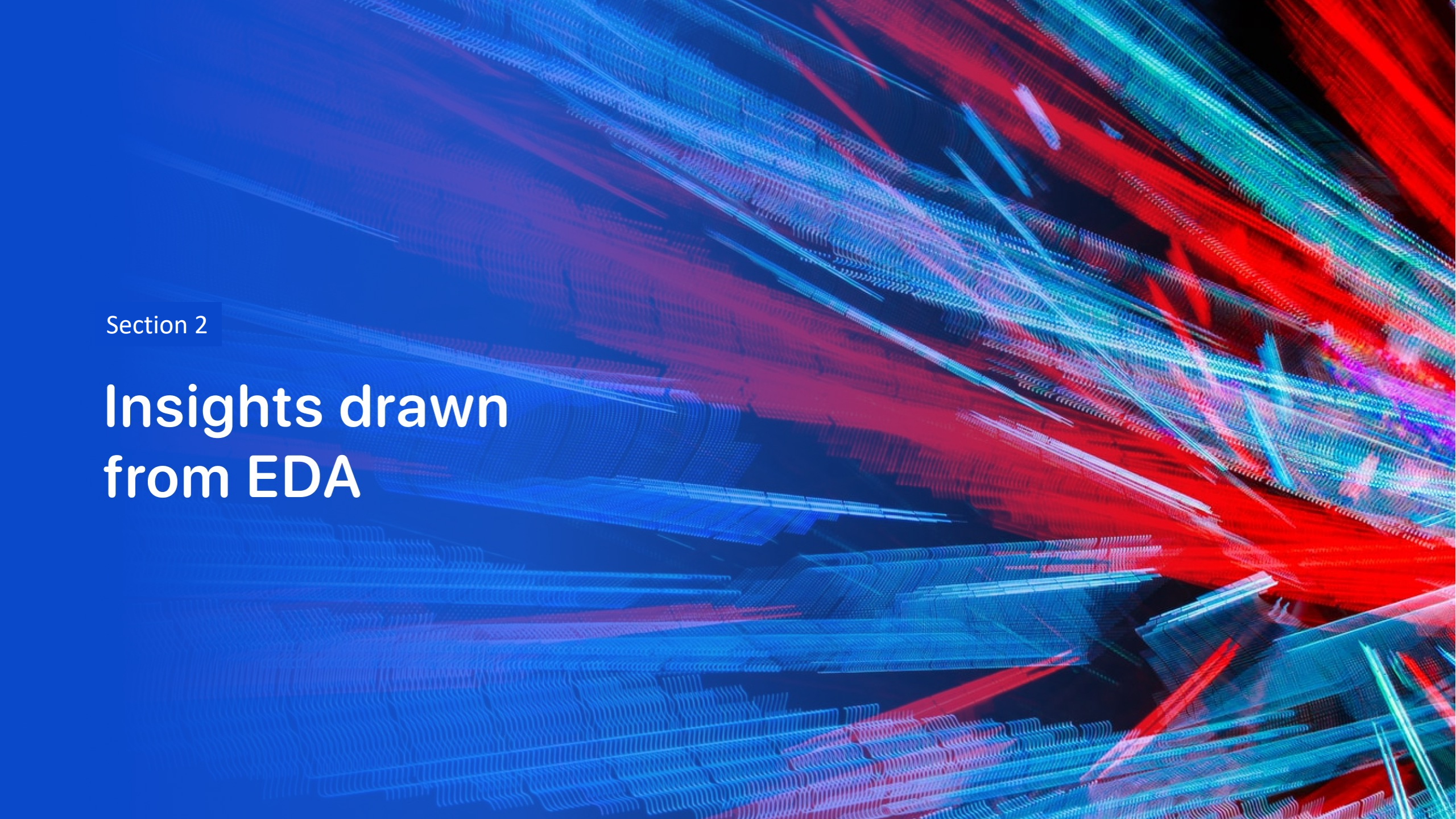
K Nearest neighbors KNN

best_score comparison

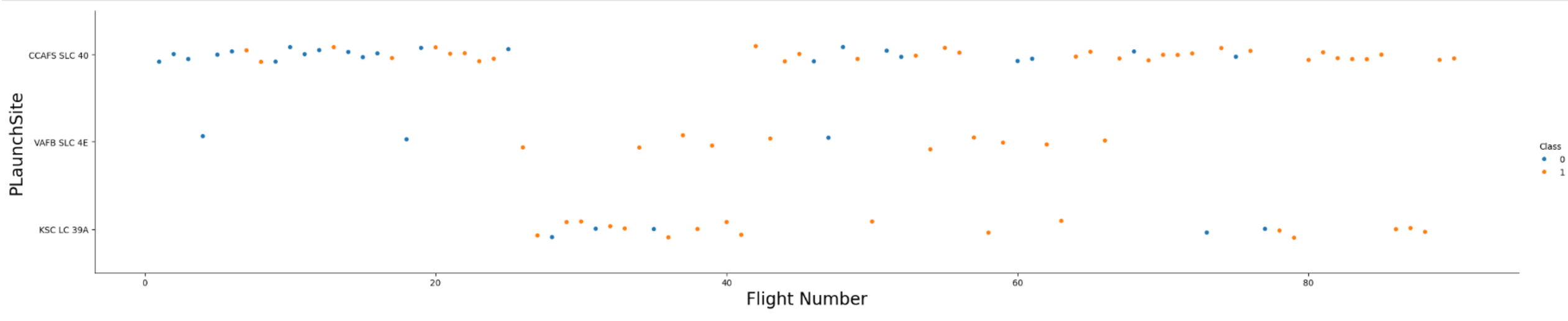
https://github.com/aeRibeiro/CapstoneFinal/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

- Increasing the number of Flight brought better results
- Using less payload cause more success retrieving the first stage
- Launch Sites KSC LC-39A and VAFB SLC 4E had the best success rate of 77%.
- Decision Tree was the best algorithm for training data set with a score of 0.86
- ES L1, GEO, HEO and SSO orbits had the best success rate.


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

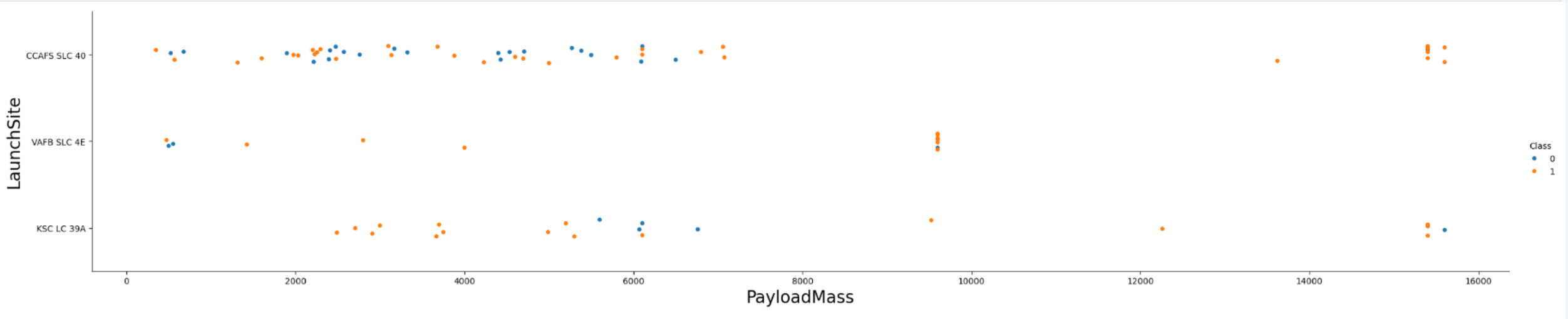


CCAFS LC-40, has a success rate of 60 %, (33 red Success and 22 blue Failed)

KSC LC-39A has a success rate of 77% (10 red Success and 3 blue Failed)

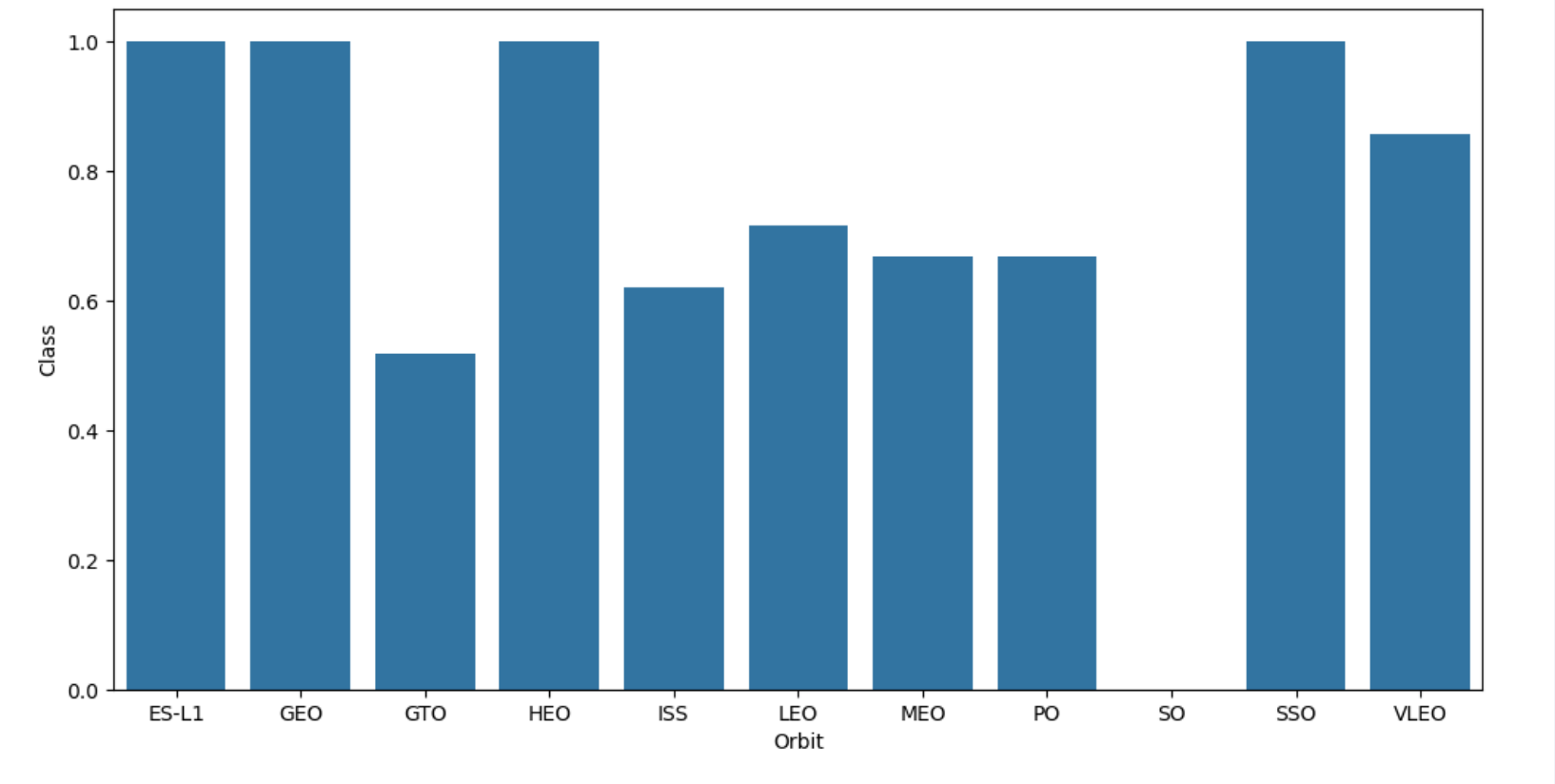
VAFB SLC 4E has a success rate of 77% (17 red Success and 5 blue Failed)

Payload vs. Launch Site



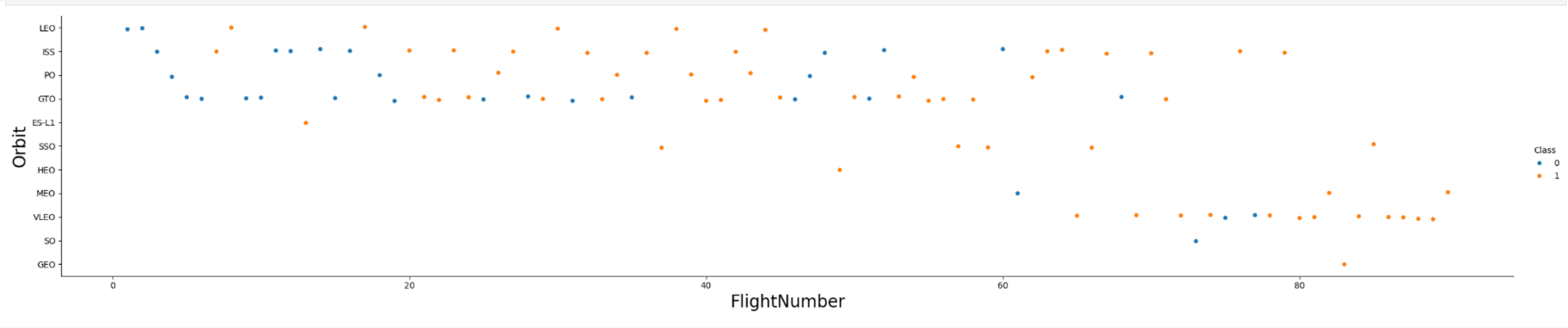
VAFB-SLC launch site there are no rockets launched for heavypayload mass(greater than 10000) and acording to previous chart it was the most successful launch site. This may link the payload with success rate.

Success Rate vs. Orbit Type



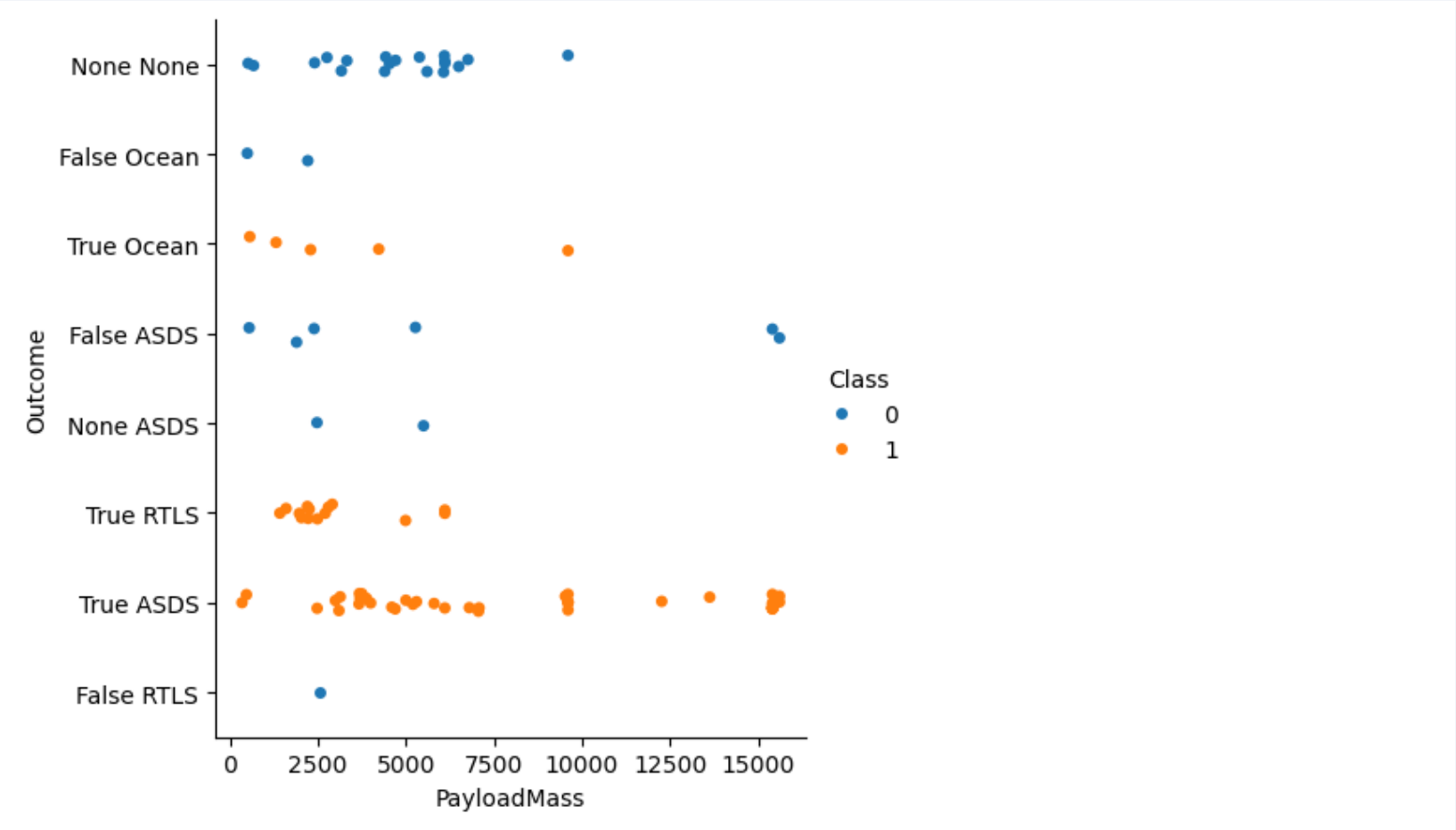
Graphic is showing the success rate for each type of orbit after getting the mean value

Flight Number vs. Orbit Type

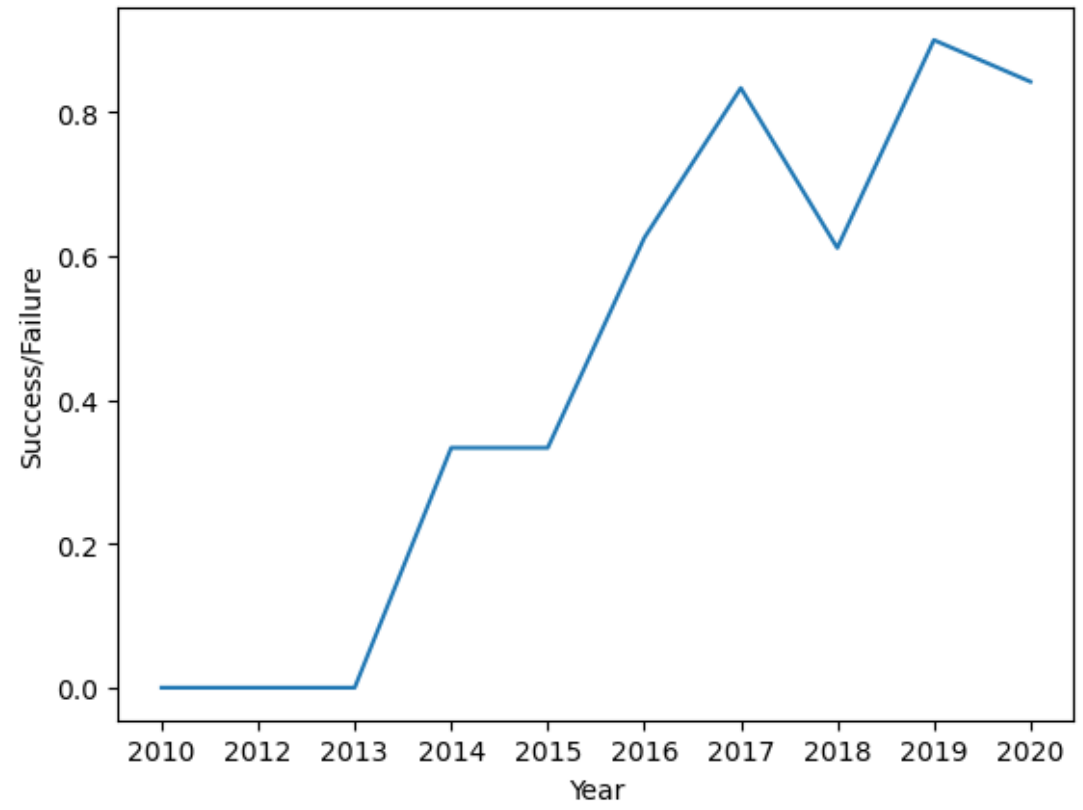


- see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type



Launch Success Yearly Trend



you can observe that the sucess rate since 2013 kept increasing till 2020

- Show the screenshot of the scatter plot with explanations

All Launch Site Names

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

```
%sql SELECT DISTINCT Launch_Site  from SPACEXTABLE
```

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

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

Using SQL makes easier to play with data like checking the Launch Sites

Launch Site Names Begin with 'CCA'

Task 2

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

```
%sql SELECT * from SPACEXTABLE 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

Total Payload Mass

Task 3

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

```
%sql SELECT Customer, SUM([PAYLOAD_MASS__KG_]) from SPACEXTABLE WHERE Customer like 'NASA (CRS)'
```

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

Customer	SUM([PAYLOAD_MASS__KG_])
NASA (CRS)	45596

Average Payload Mass by F9 v1.1

Task 4

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

```
%sql SELECT Customer, AVG([PAYLOAD_MASS__KG_]) from SPACEXTABLE WHERE Customer like 'NASA (CRS)'
```

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

Customer	AVG([PAYLOAD_MASS__KG_])
NASA (CRS)	2279.8

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 SPACEXTABLE WHERE MIssion_Outcome like 'Success'
```

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

MIN(Date)
2010-06-04

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 DISTINCT Booster_Version, PAYLOAD_MASS__KG_, Mission_Outcome from SPACEXTABLE WHERE ((PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000) AND Mission_Outcome = "Success")
```

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

Booster_Version	PAYLOAD_MASS__KG_	Mission_Outcome
F9 v1.1	4535	Success
F9 v1.1 B1011	4428	Success
F9 v1.1 B1014	4159	Success
F9 v1.1 B1016	4707	Success
F9 FT B1020	5271	Success
F9 FT B1022	4696	Success
F9 FT B1026	4600	Success
F9 FT B1030	5600	Success
F9 FT B1021.2	5300	Success
F9 FT B1032.1	5300	Success
F9 B4 B1040.1	4990	Success
F9 FT B1031.2	5200	Success
F9 FT B1032.2	4230	Success
F9 B4 B1040.2	5384	Success
F9 B5 B1046.2	5800	Success
F9 B5 B1047.2	5300	Success
F9 B5 B1048.3	4850	Success
F9 B5 B1051.2	4200	Success
F9 B5B1060.1	4311	Success
F9 B5 B1058.2	5500	Success
F9 B5B1062.1	4311	Success

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) from SPACEXTABLE GROUP BY Mission_Outcome
```

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

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

Boosters Carried Maximum Payload

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

```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

- Sub Query to get the Max payload as requested

2015 Launch Records

```
%sql SELECT substr(Date,6,2) as month, DATE, BOOSTER_VERSION, LAUNCH_SITE, Landing_Outcome FROM SPACEXTBL where (Landing_Outcome LIKE 'Failure%' and substr(Date,0,5)='2015')
```

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

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

We used LIKE to find all Failures in the year specified by Date,0,5)

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.

```
%sql SELECT COUNT(Landing_Outcome) as count_outcomes, Landing_Outcome  FROM SPACEXTBL  WHERE (DATE > '2010-06-04' AND DATE < '2017-030-20')  group by Landing_Outcome order by count_outcomes DESC;
```

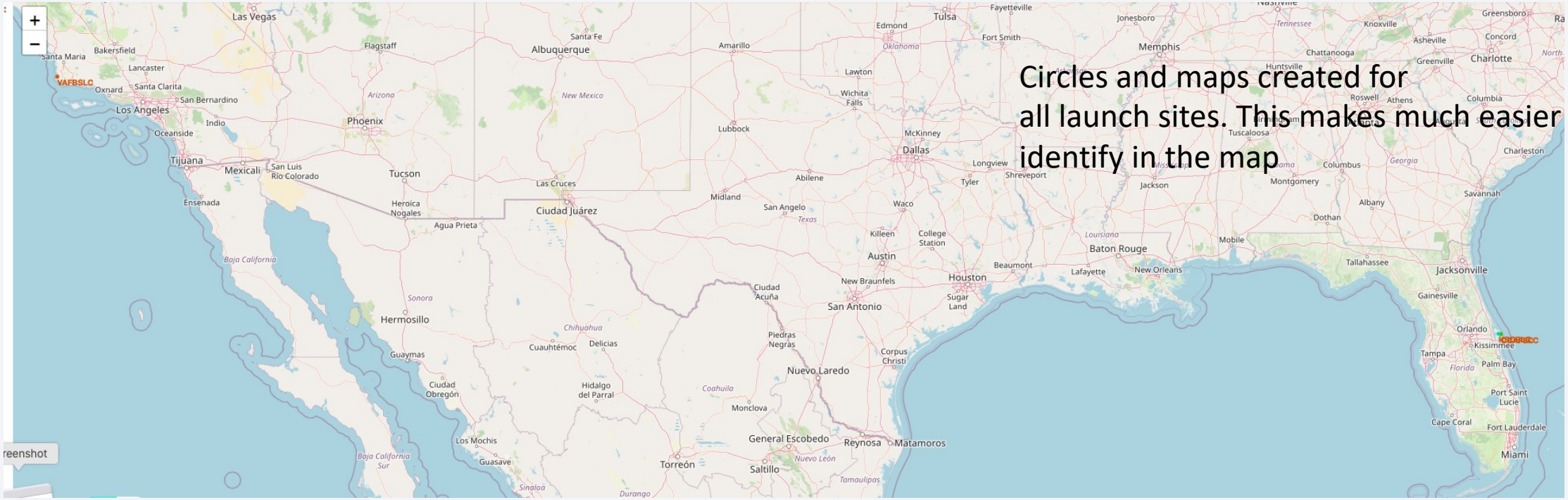
```
 * sqlite:///my_data1.db
Done.
```

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

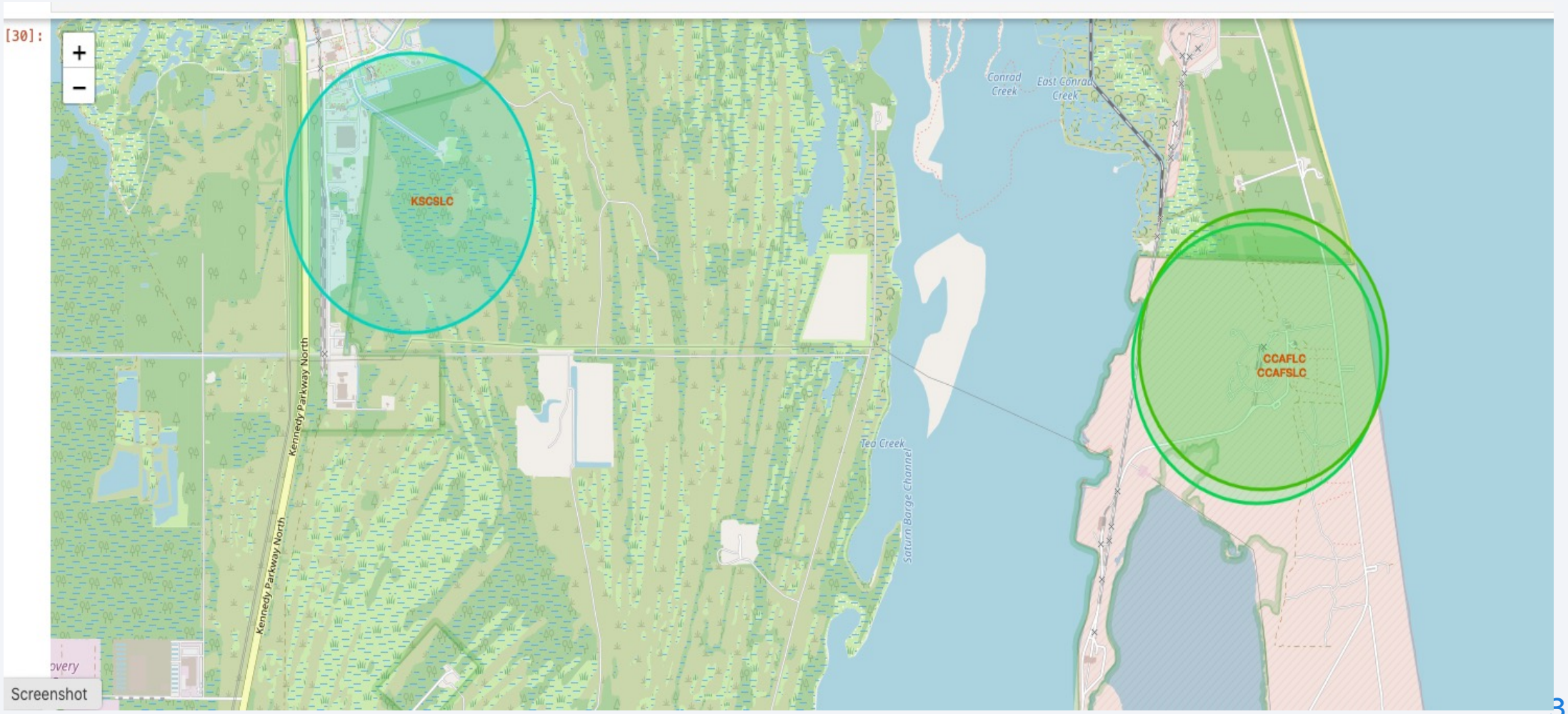
Descending order between dates and gruped by Landing Outcome


Section 3
Launch Sites
Proximities Analysis

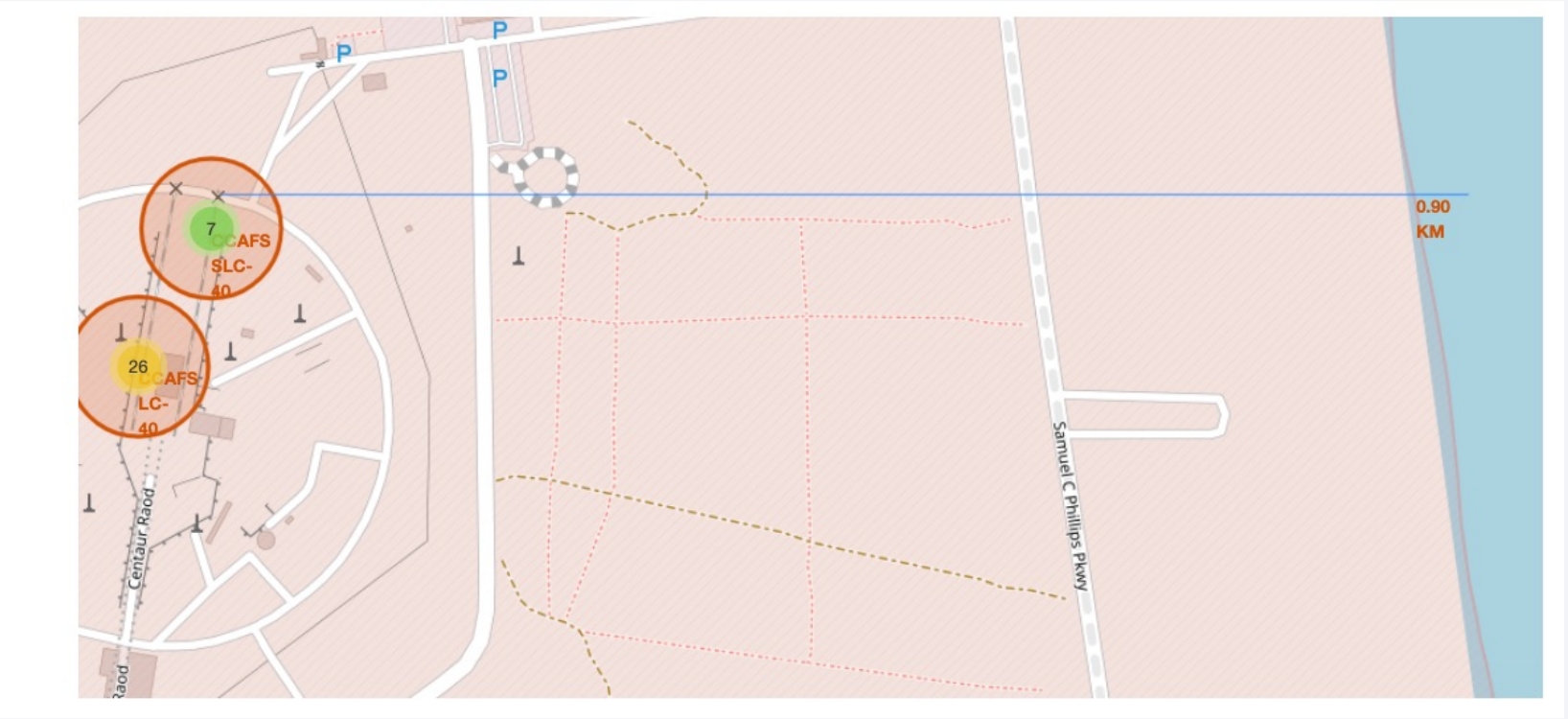
Launch Sites with Circles and Markers

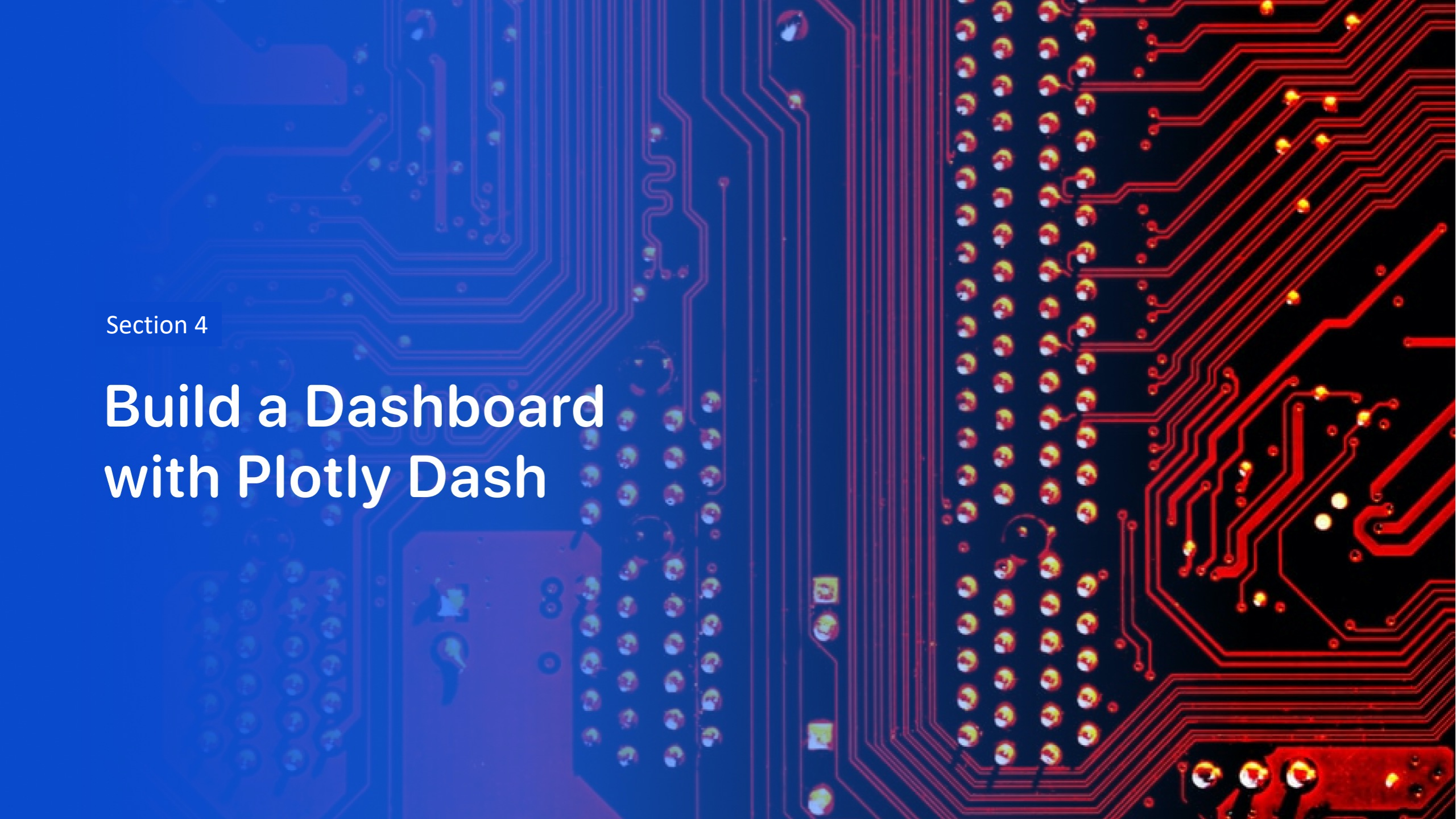


Color Labeled Maps

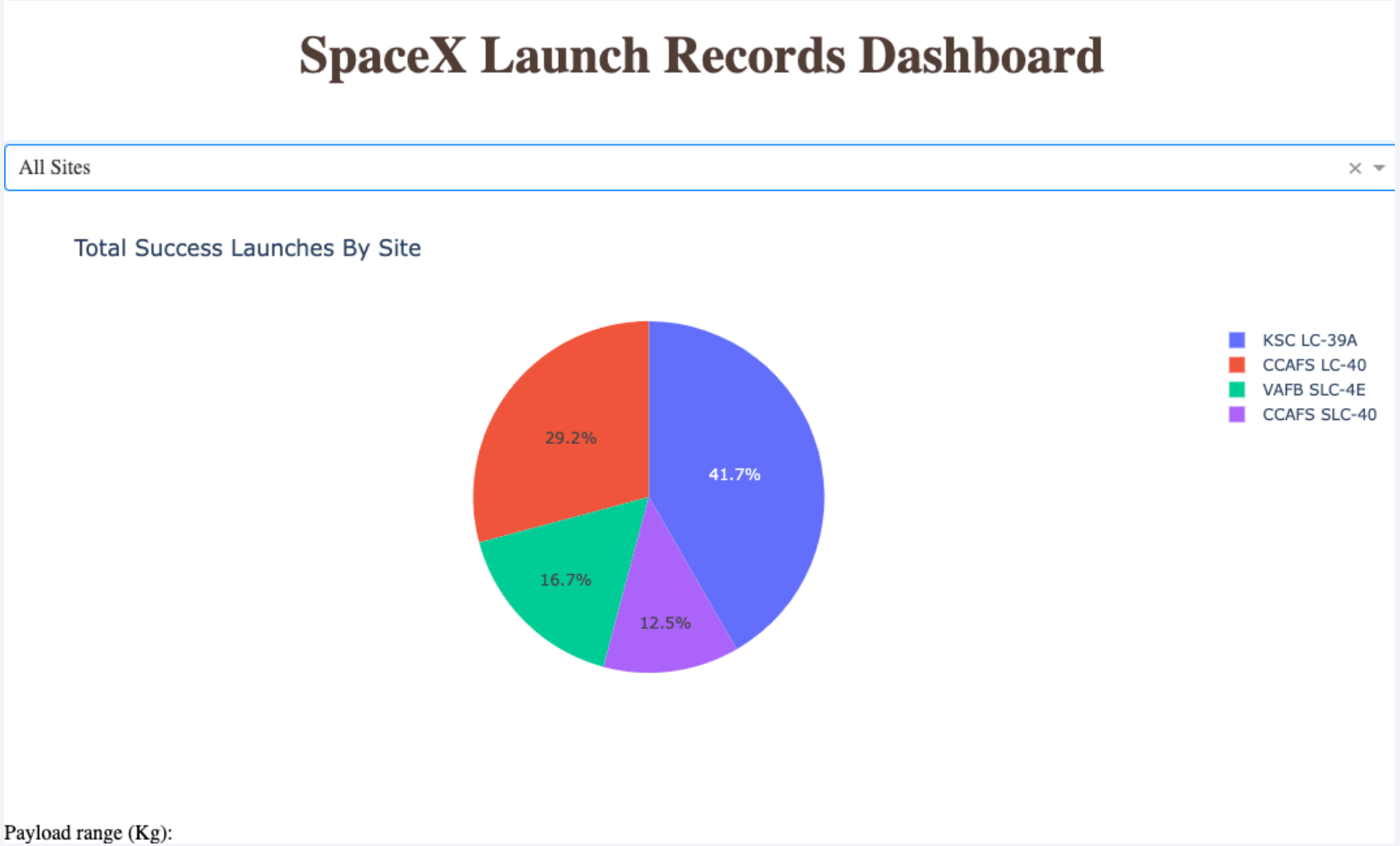


Distance to Proximities



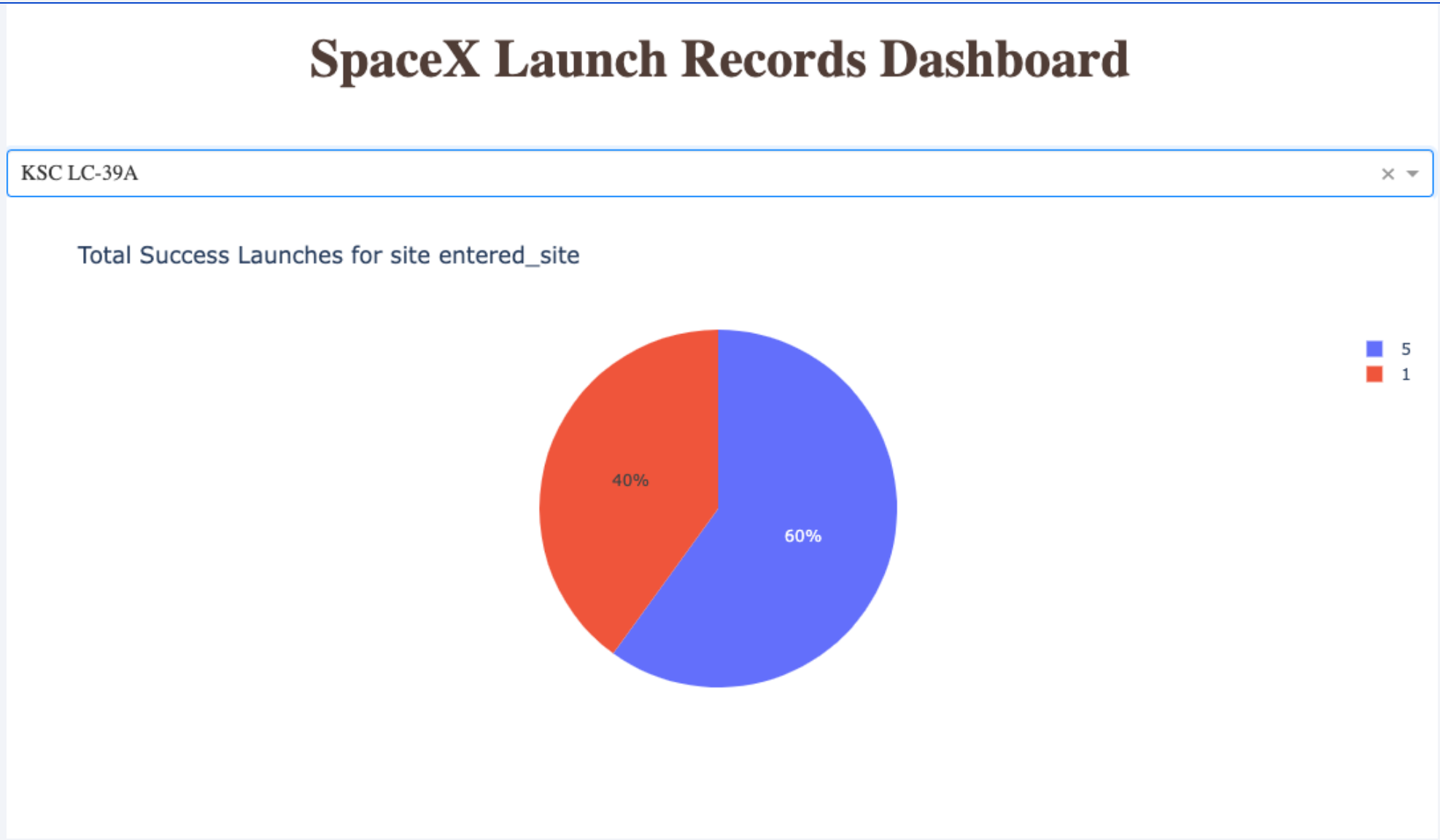

Section 4
Build a Dashboard with Plotly Dash

Launch Success Count for All Sites

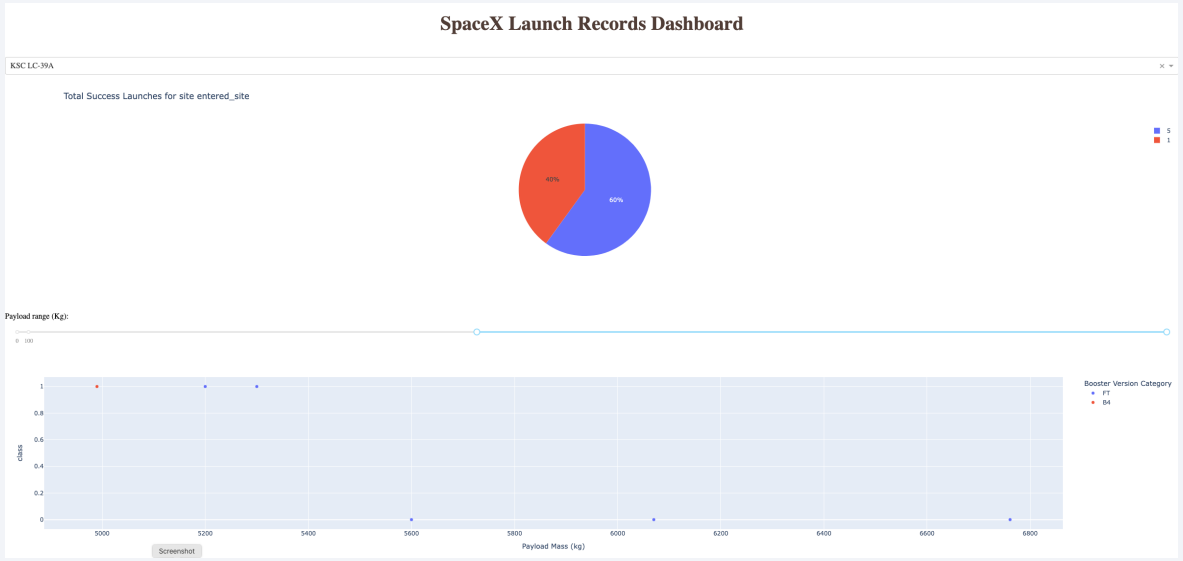


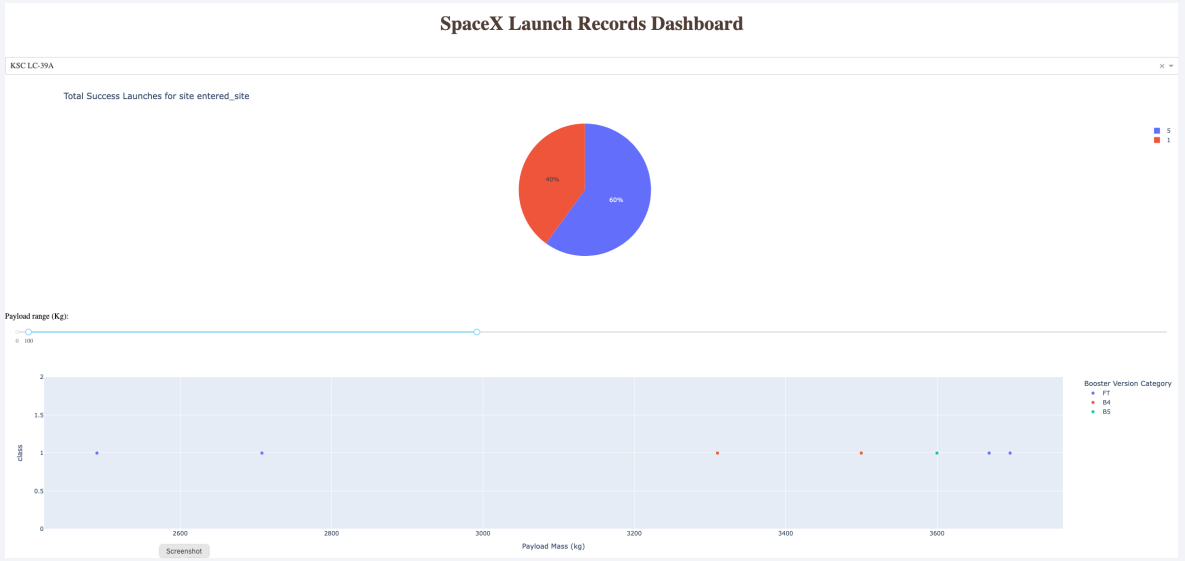
We can visualize sites with best success using a pie chart

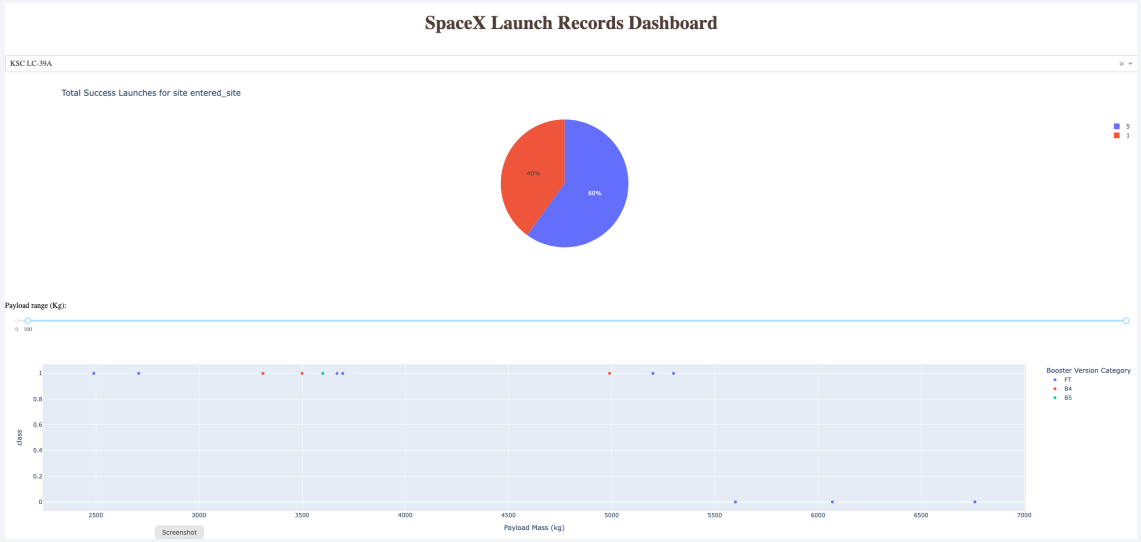
KSC LC-39A success ratio - Highest launch site rate



Payloads vs Launch Outcome







Different payloads selected will change the Class values for different Booster Version Category where 1 is a successfull and 0 a failure. This can be applied to All Sites or specific ones

Section 5

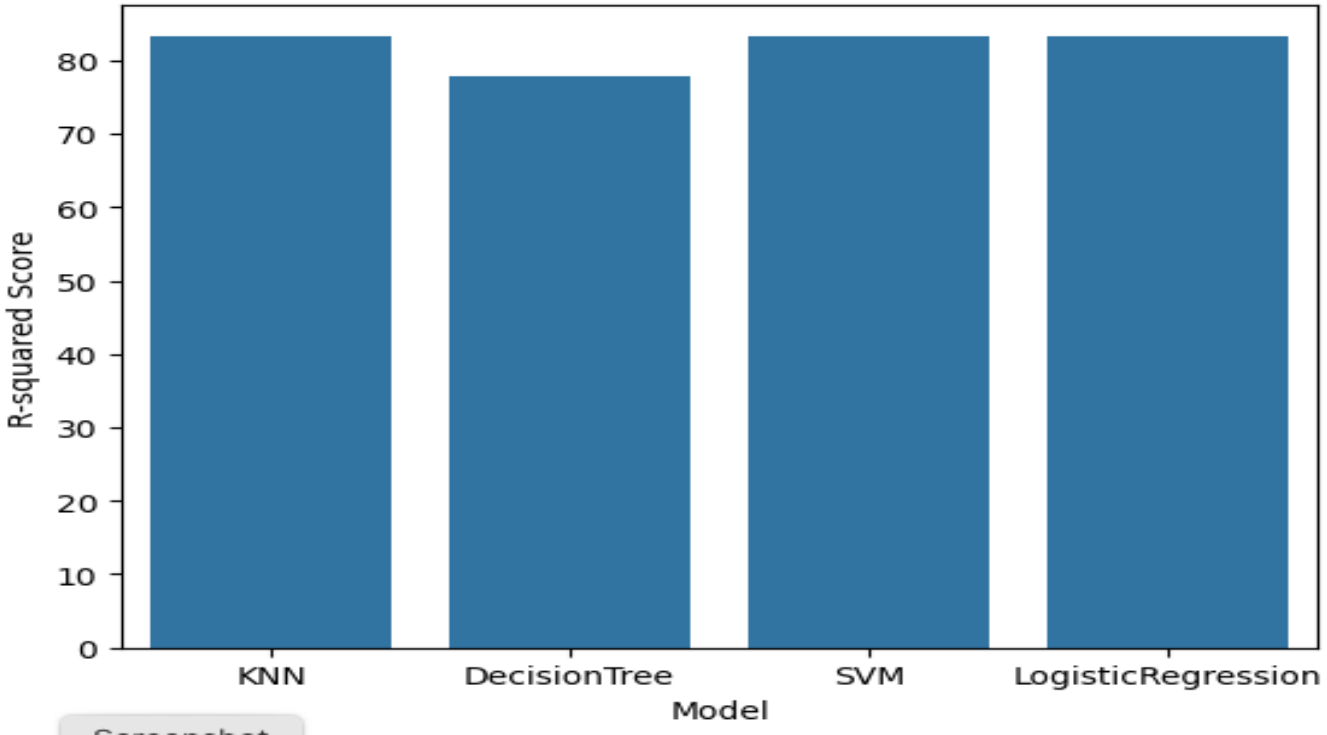
Predictive Analysis (Classification)

Classification Accuracy

	Model	R-squared Score
0	KNN	83.333333
2	SVM	83.333333
3	LogisticRegression	83.333333
1	DecisionTree	77.777778

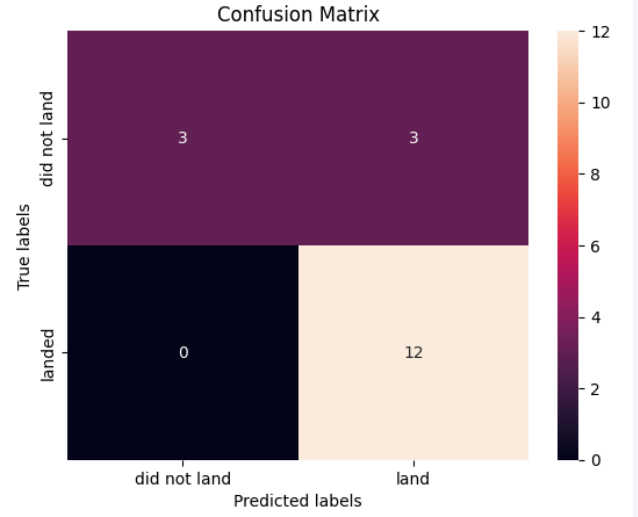
```
sns.barplot(x='Model', y='R-squared Score', data=models)
```

```
<AxesSubplot:xlabel='Model', ylabel='R-squared Score'>
```

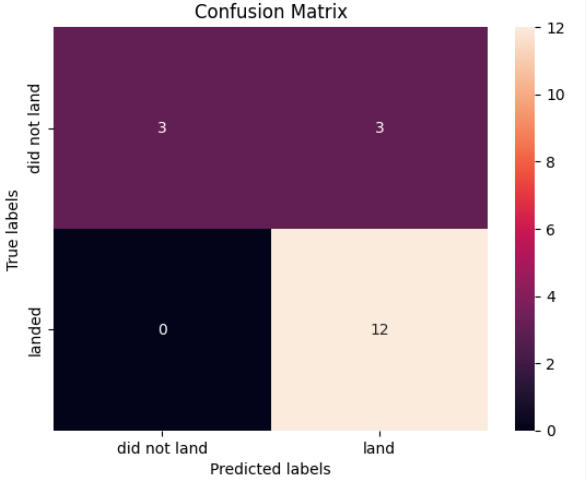


Confusion Matrix

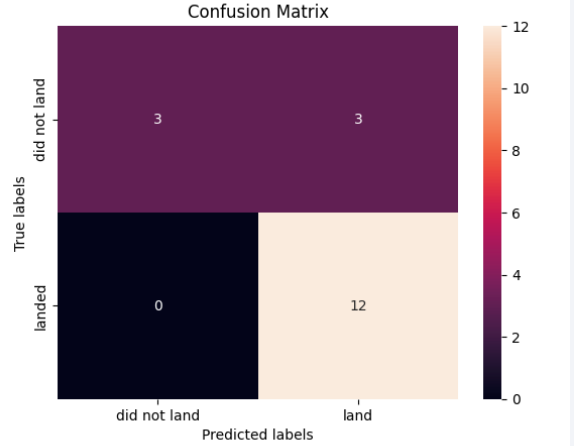
```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



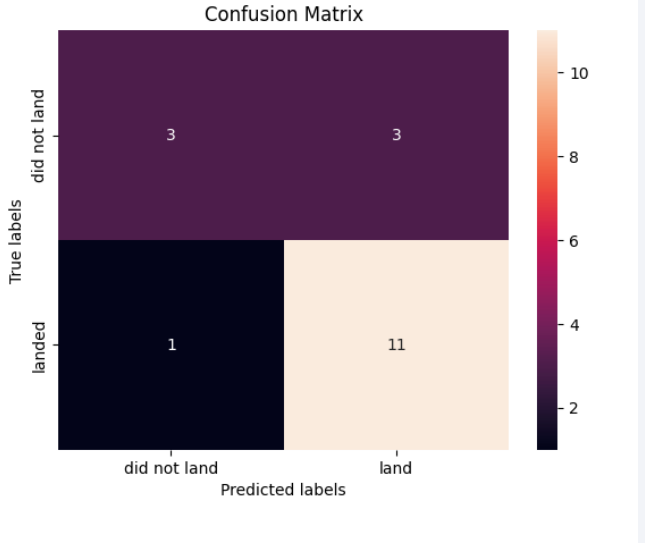

```
yhat=svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```




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




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



Conclusions

Increasing the number of Flight brought better results

Using less payload cause more success retrieving the first stage

Launch Sites KSC LC-39A and VAFB SLC 4E had the best success rate of 77%.

Decision Tree was the best algorithm for training data set with a score of 0.86

ES L1, GEO, HEO and SSO orbits had the best success rate.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!