Winning Space Race with Data Science

Brian Yu
Jan 2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection performed via both SpaceX API and Web scraping
  - EDA performed using standard and interactive charts, and map visualization
  - Predictive machine learning models with four different algorithms

- Summary of all results
  - Decision Tree algorithm generates most accurate predictive model
  - There are correlations between the success rate and payload mass, flight number, and year

Introduction

- Project background and context
  - The recent success of commercial space journey has provoked competition on making space travel affordable for everyone. Companies such as Virgin Galactic and Blue Origin are racing on their development of suborbital spaceflights and reusable rockets.
  - Among all the spaceflight service companies, SpaceX has been the most successful one because its rocket launches are relatively inexpensive. Falcon 9 rocket launch, as SpaceX advertises on its website, cost around 62 million dollars while other providers cost upward of 165 million dollar each, much of the saving is because SpaceX can reuse the first stage.
  - The competing company would like to bid against SpaceX by knowing when the first stage will land successfully in order to determine the price for each launch
- Problems you want to find answers
  - Under what condition will the rocket land successfully
  - The relationship each parameter plays with the outcome of the landing

Section 1

Methodology

Data Collection – SpaceX API

1. Request information from SpaceX API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"

response = requests.get(spacex_url)
```

2. Read and parse json results

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

3. Call custom function to get info from API

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

4. Filter out irrelevant information

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

5. Deal with missing values

```
# Calculate the mean value of PayloadMass column
meanPayLoadMass = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9.loc[:, 'PayloadMass'] = data_falcon9['PayloadMass'].replace(np.nan, meanPayLoadMass)
```

6. Save and export

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
4	1	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857
5	2	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.561857
6	3	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007	-80.577366	28.561857
7	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0	B1003	-120.610829	34.632093
8	5	2013-12-03	Falcon 9	3170.000000	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B1004	-80.577366	28.561857
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
89	86	2020-09-03	Falcon 9	15600.000000	VLEO	KSC LC 39A	True ASDS	2	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	9	B1060	-80.603956	28.608058
90	87	2020-10-06	Falcon 9	15600.000000	VLEO	KSC LC 39A	True ASDS	3	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	9	B1058	-80.603956	28.608058
91	88	2020-10-18	Falcon 9	15600.000000	VLEO	KSC LC 39A	True ASDS	6	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	10	B1051	-80.603956	28.608058

GitHub URL

Data Collection - Scraping

1. Request information from Web

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
response = requests.get(static_url)
```

2. Read and parse into BeautifulSoup

```
soup = BeautifulSoup(response.text, 'html5lib')
```

3. Find the target information from table

```
html_tables = soup.find_all('table')
```

4. Create dataframe from parsed HTML table

```
launch_dict= dict.fromkeys(column_names)
df=pd.DataFrame(launch_dict)
```

5. Save and export

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10
...	...	...	...	...	...	...	...	...	...	...	...
116	117	CCSFS	Starlink	15,600 kg	LEO	SpaceX	Success\n	F9 B5B1051.10	Success	9 May 2021	06:42
117	118	KSC	Starlink	~14,000 kg	LEO	SpaceX	Success\n	F9 B5B1058.8	Success	15 May 2021	22:56
118	119	CCSFS	Starlink	15,600 kg	LEO	SpaceX	Success\n	F9 B5B1063.2	Success	26 May 2021	18:59
119	120	KSC	SpaceX CRS-22	3,328 kg	LEO	NASA	Success\n	F9 B5B1067.1	Success	3 June 2021	17:29
120	121	CCSFS	SXM-8	7,000 kg	GTO	Sirius XM	Success\n	F9 B5	Success	6 June 2021	04:26

GitHub URL

Data Wrangling

1. Calculate the number of launches on each site

2. Calculate the number and occurrence of each orbit

3. Calculate the number and occurrence of mission outcome per orbit type

4. Create a landing outcome label from Outcome column

5. Save and export

```
# Apply value_counts() on column LaunchSite
df.LaunchSite.value_counts()
```

```
CCAFS SLC 40     55
KSC LC 39A       22
VAFB SLC 4E      13
```

```
# Apply value_counts on Orbit column
df.Orbit.value_counts()
```

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

```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise

landing_class = [0 if outcome in bad_outcomes else 1 for outcome in df['Outcome']]
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitu
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.5773
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.5773
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.5773

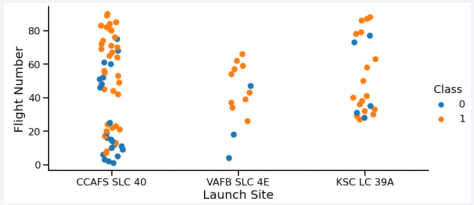
GitHub URL

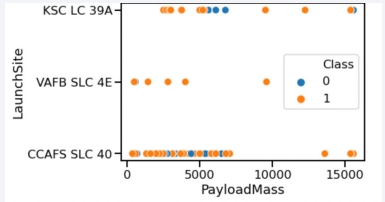
EDA with Data Visualization

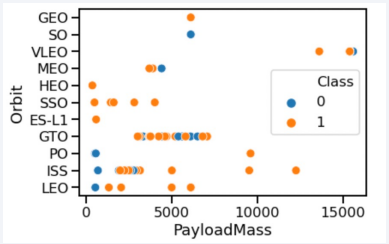
Scatter Graph Generated:

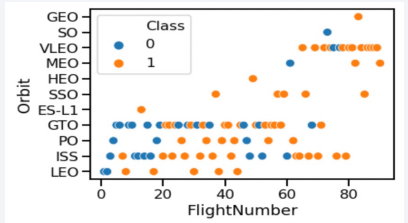
- Payload Mass vs. Flight Number
- Launch Site vs. Flight Number
- Payload Mass vs. Launch Site
- Flight Number vs. Orbit Type
- Payload Mass vs. Orbit Type

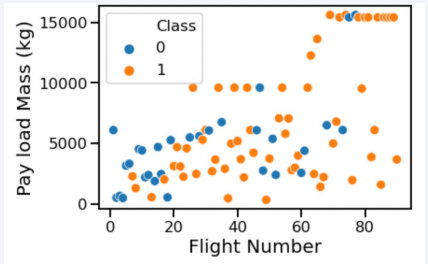
Scatter plots provides an easy visualization of the correlation between two attributes which helps us in determining the dependent parameters for the machine learning models built later in the project









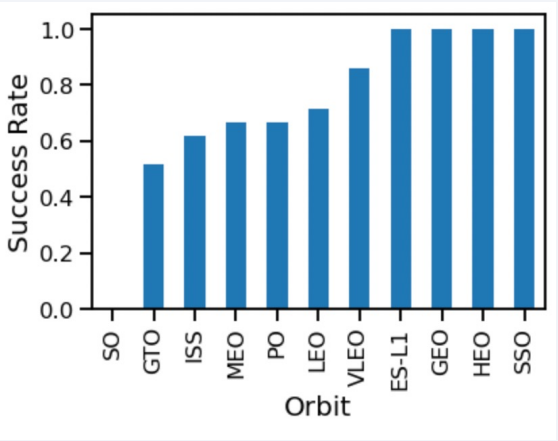


GitHub URL

Bar Graph Generated:

- Orbit Type vs. Success Rate

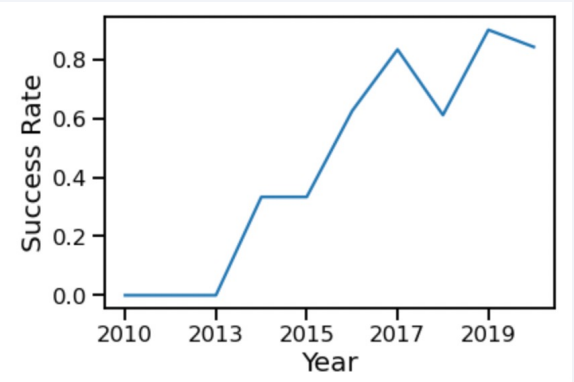
Bar plot is helpful when comparing different types of a certain attribute against a certain criterion. From the figure below, we can conclude that ES-L1, GEO, HEO, and SSO have highest success rate



Line Graph Generated:

- Year of Launch vs. Success Rate

Line plot is beneficial to see how a certain attribute change over another sequential attribute. This can help predicting the trend in the future.



EDA with SQL

SQL Queries Performed:

- Display the names of the unique launch sites in the space mission.
- Display 5 records where launch sites begin with the string 'CCA’.
- Display the total payload mass carried by boosters launched by NASA (CRS).
- Display average payload mass carried by booster version F9 v1.1.
- List the date when the first successful landing outcome in ground pad was achieved.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
- List the total number of successful and failure mission outcomes.
- List the names of the booster versions which have carried the maximum payload mass. Use a subquery.
- List the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015.
- 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.

GitHub URL

Build an Interactive Map with Folium

Map Objects	Code	Explanation
Map	folium.Map()	Initialize a map
Circle	folium.Circle()	Place a circle marker at given coordinate
Marker	folium.map.Marker()	Place a marker at given coordinate
Icon	folium.features.DivIcon()	Create an icon at given coordinate
MarkerCluster	folium.plugins.MarkerCluster()	Combine nearby markers into cluster and simplify the display
PolyLine	folium.PolyLine()	Create a line that connects the given coordinates

GitHub URL

Build a Dashboard with Plotly Dash

Objects	Explanation
Dropdown menu	Dropdown menu allows the user to switch between whether they want to see a general summary or focus on a specific launch site
Pie Chart	Pie chart is connected to the dropdown menu and automatically update to the right information when user makes a selection. It either displays percentage of successful launches by site or percentage of successful and unsuccessful launch, which displays the relationship between launch site and rate of success
Slider	Slider allows the user to select the range of payload they want to be focus at
Scatter Plot	The scatter plot is connected to the slider and automatically update to only include launches that has payload within the interested range to show correlation between success rate and booster version

GitHub URL

Predictive Analysis (Classification)

- Load and convert the data
- Standardize the data
- Split to train, test sets
- Determine learning algorithm
- Run the model with GridSearchCV

```
X = pd.read_csv()
Y = data['Class'].to_numpy()
X = preprocessing.StandardScaler().fit_transform(X)
```

- Train the model with train dataset
- Determine accuracy
- Generate Confusion Matrices

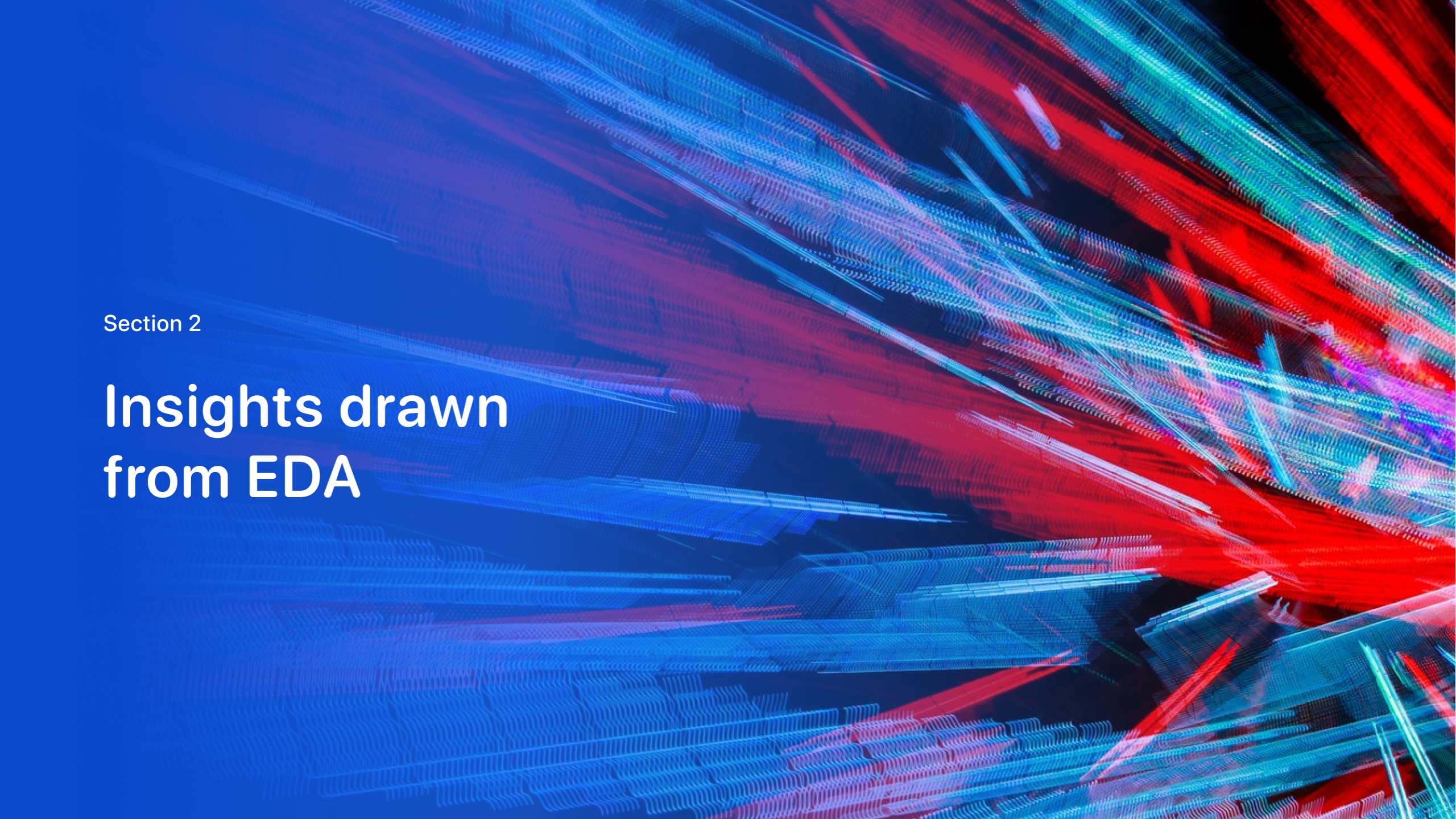
```
logreg_cv = GridSearchCV(lr, parameters)
logreg_cv.fit(X_train, Y_train)
logreg_cv.score(X_test, Y_test)
plot_confusion_matrix(Y_test,yhat)
```

- Determine best hyperparameter combination for each algorithm

```
logreg_cv.best_params_
```

- Final evaluation of each algorithm with best hyperparameter

GitHub URL

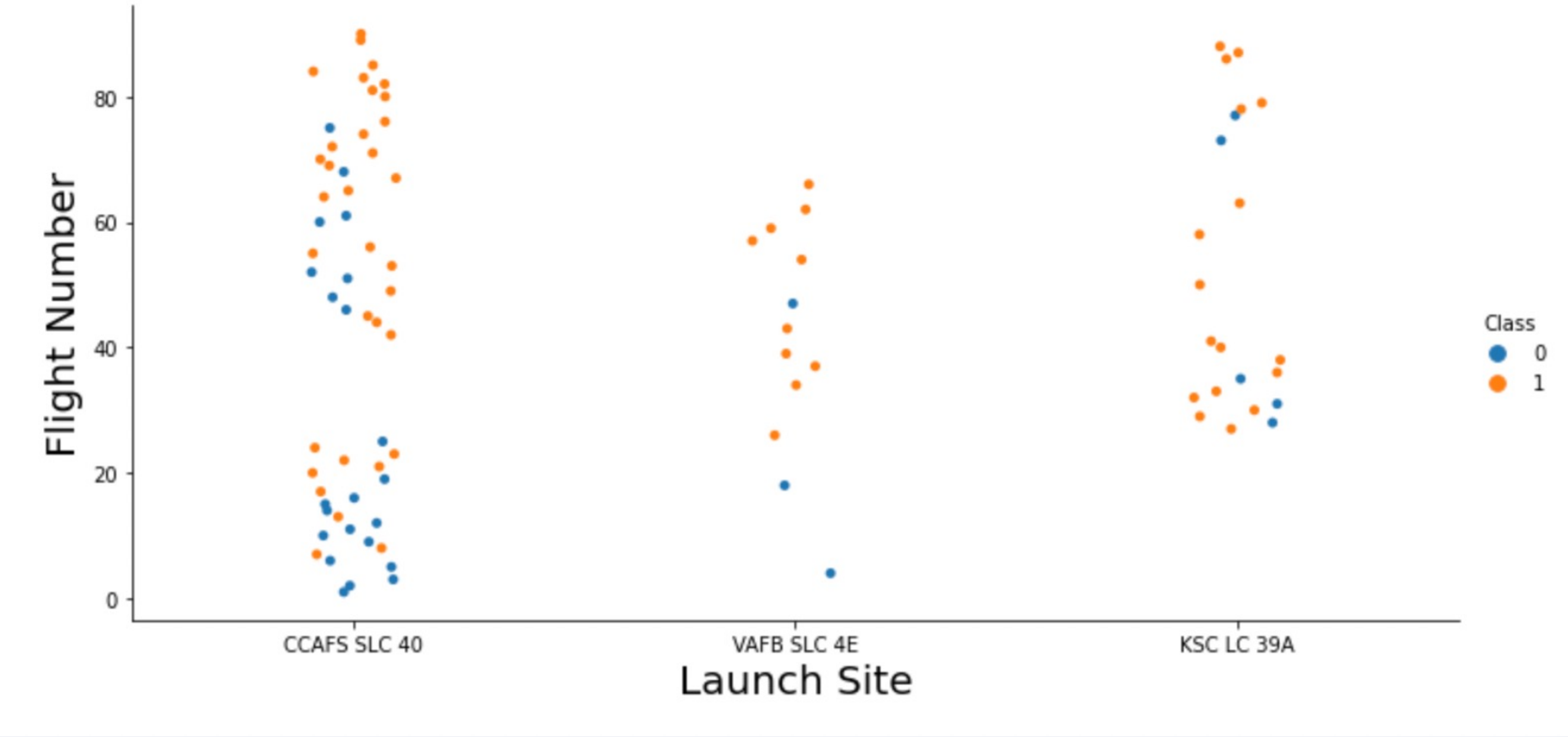


Section 2

Insights drawn from EDA

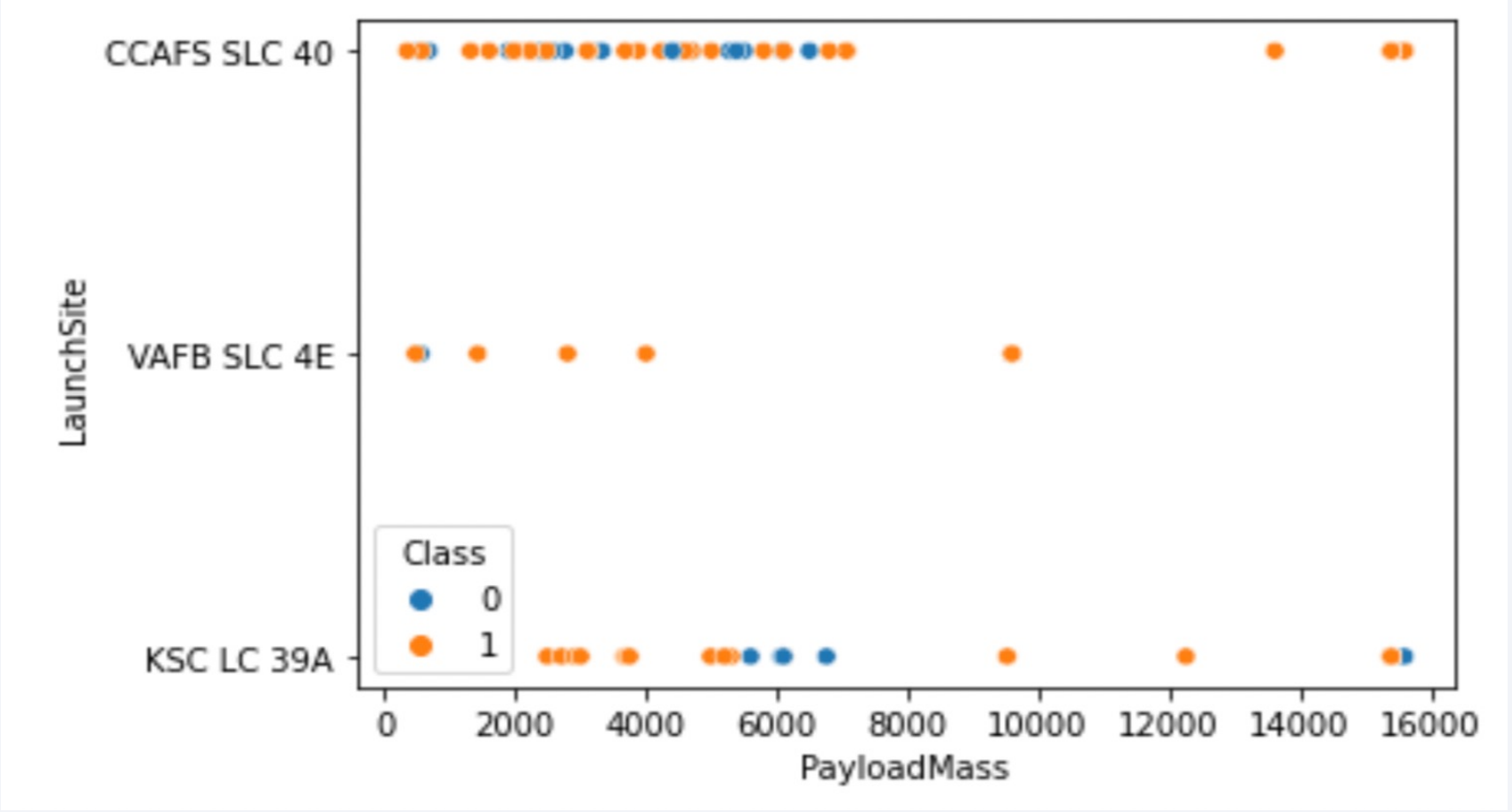
Flight Number vs. Launch Site

- As flight number increases, the possibility of a launch being successful also increases.



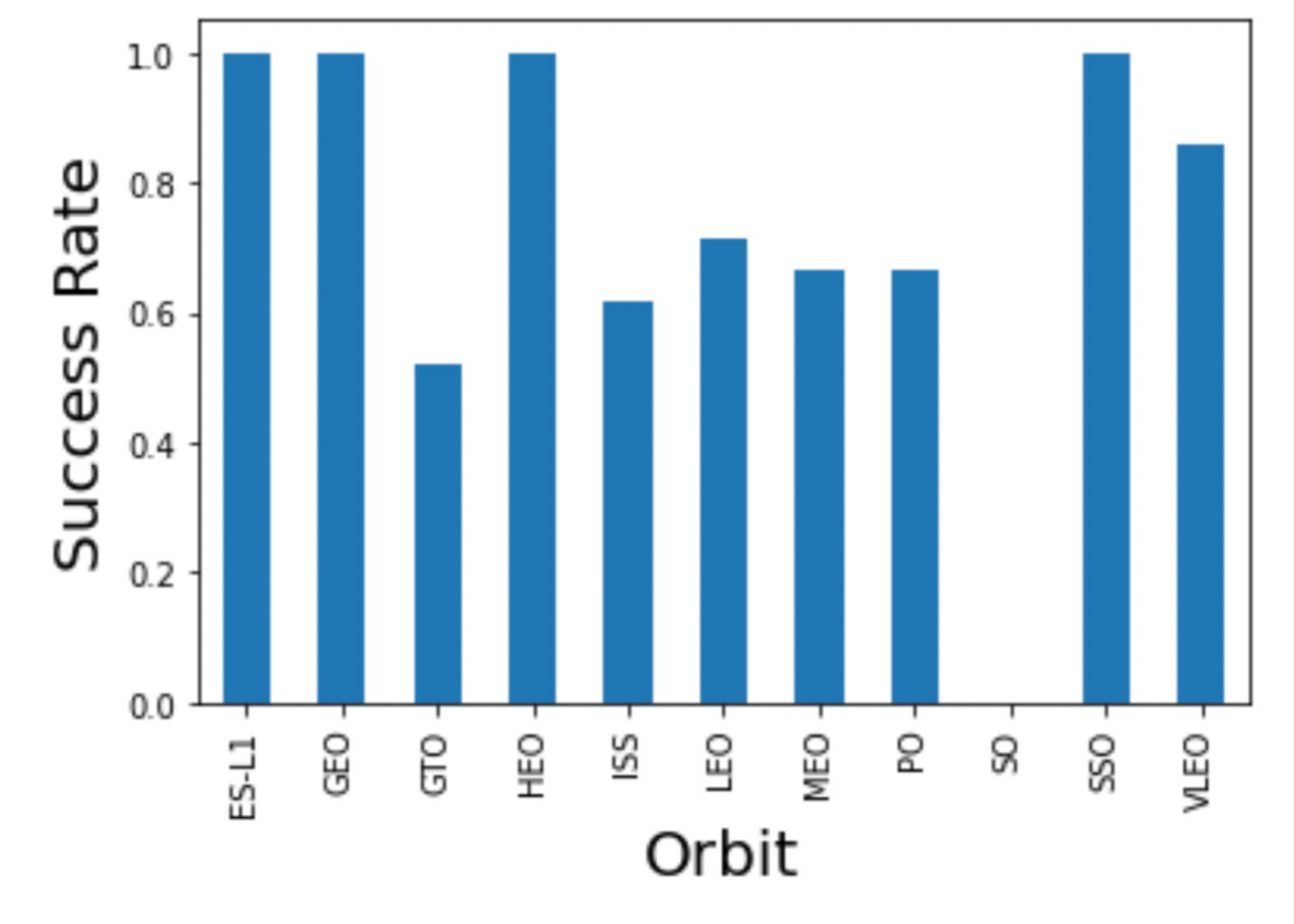
Payload vs. Launch Site

- CCAFS SLC 40: There is a higher success rate when the payload is heavy
- VAFB SLC 4E: This launch site did not launch rockets with high payload
- KSC LC 39A: Lower payload has more success rates



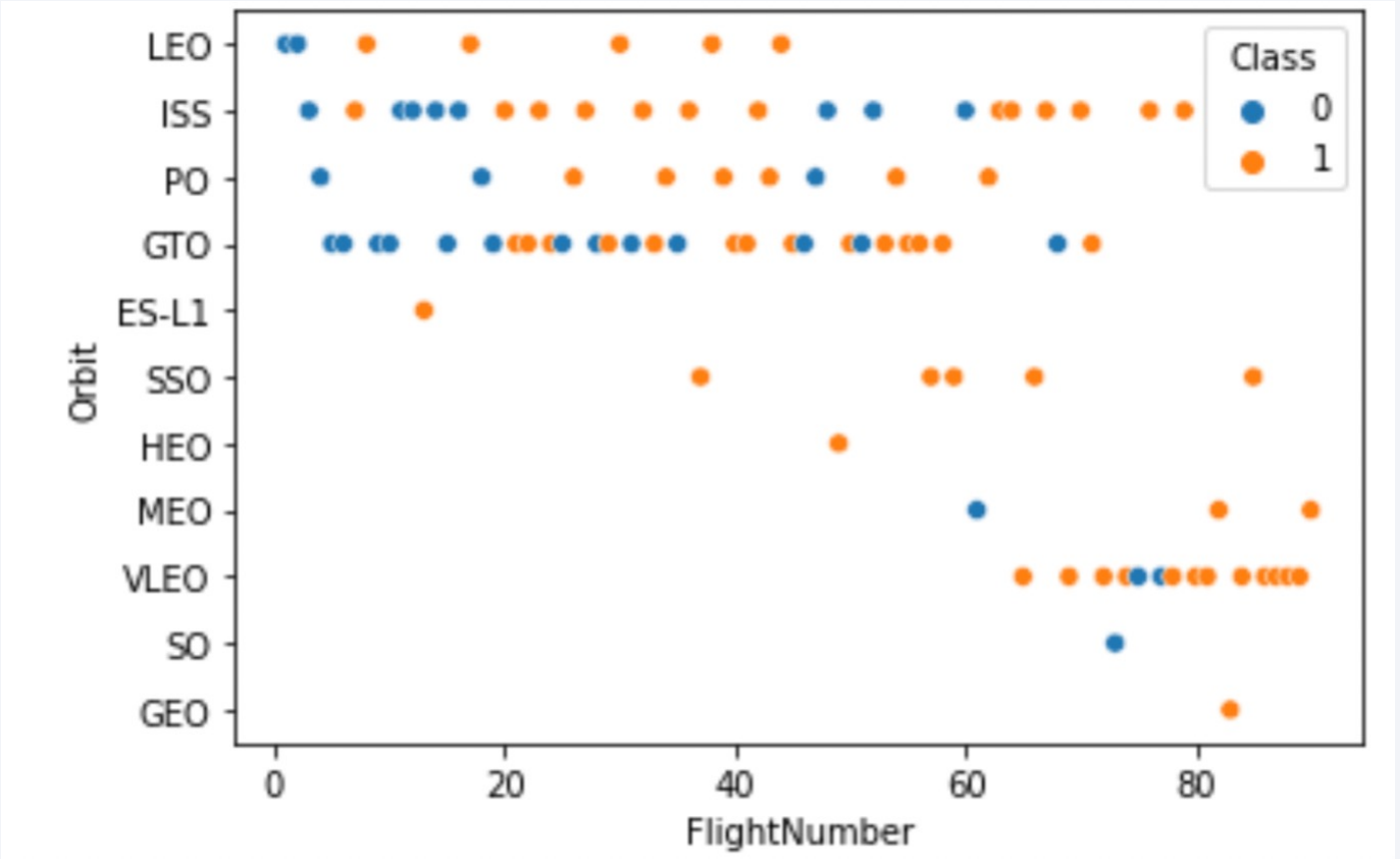
Success Rate vs. Orbit Type

- The top four orbit types with 100% success rate are ES-L1, GEO, HEO, and SSO



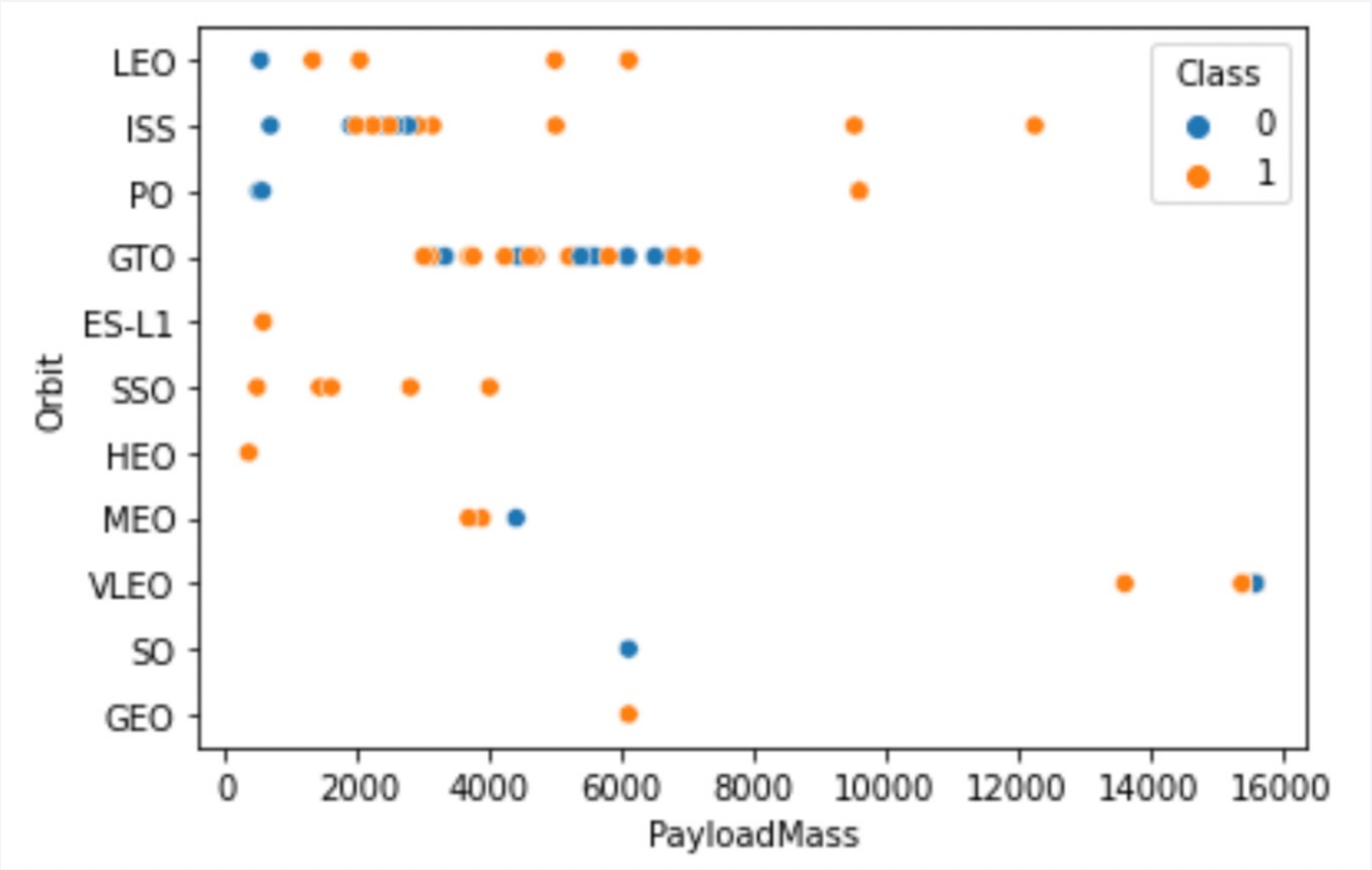
Flight Number vs. Orbit Type

- For orbit types such as LEO and PO, the success rate gradually increase as the flight number increase
- GTO, however, has no relationship between flight number and success rate



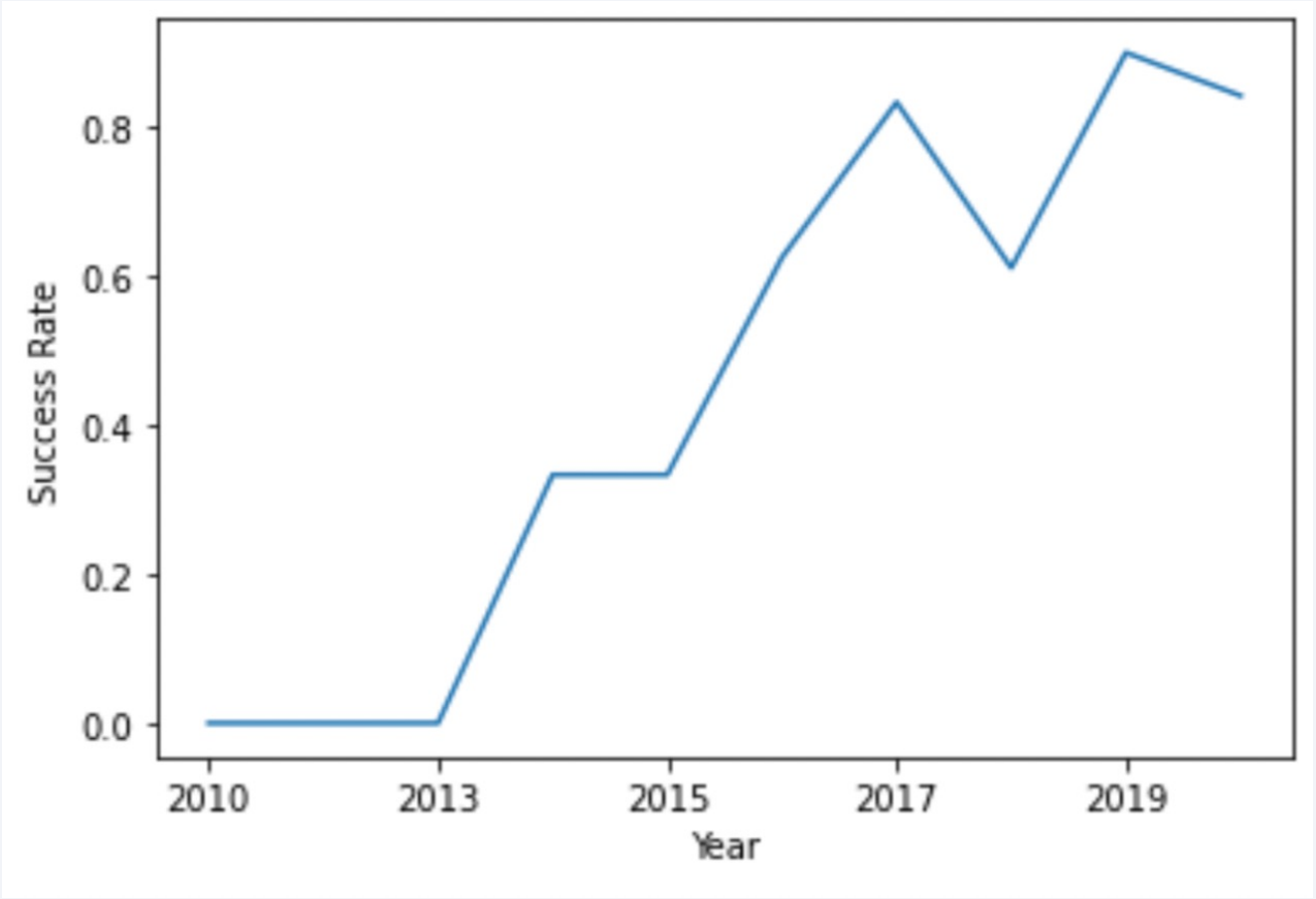
Payload vs. Orbit Type

- Orbit type such as LEO, ISS, and PO have higher success rate when the payload is heavier. However, GTO does not exhibit a strong relation between payload mass and success rate



Launch Success Yearly Trend

- The launch success rate has been gradually increasing since 2013



All Launch Site Names

Command:

```
%sql SELECT UNIQUE launch_site FROM SPACEXTBL
```

Output:

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

Explanation:

- This command query the launch site name from the SpaceX table
- Key word ”UNIQUE” was used to remove duplicates

Launch Site Names Begin with 'CCA'

Command:

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

Output:

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

Explanation:

- Key word ‘LIKE’ is used to use regular expression for where clause condition
- Key word ’LIMIT’ is used to limit only display 5 entries

Total Payload Mass

Command:

```
%sql SELECT SUM(payload_mass__kg_) FROM SPACEXTBL WHERE customer = 'NASA (CRS)'
```

Output:

1
45596

Explanation:

- SUM function is called to sum up all the values defined by the where clause

Average Payload Mass by F9 v1.1

Command:

```
%sql SELECT AVG(payload_mass__kg_) FROM SPACEXTBL WHERE booster_version = 'F9 v1.1'
```

Output:

1
2928

Explanation:

- AVG function is called to average all the values defined by the where clause

First Successful Ground Landing Date

Command:

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

Output:

1
2015-12-22

Explanation:

- MIN function is called to get the first date object satisfying the where clause

Successful Drone Ship Landing with Payload between 4000 and 6000

Command:

```
%sql SELECT UNIQUE booster_version FROM SPACEXTBL WHERE (mission_outcome = 'Success') AND (payload_mass__kg_ BETWEEN 4000 AND 6000)
```

Output:

booster_version
F9 B4 B1040.2
F9 B4 B1040.1
F9 B5 B1046.2
F9 B5 B1046.3
F9 B5 B1047.2
F9 B5 B1048.3
F9 B5 B1051.2
F9 B5 B1058.2
F9 B5B1054
F9 B5B1060.1
F9 B5B1062.1
F9 FT B1021.2
F9 FT B1031.2
F9 FT B1032.2
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1030
F9 FT B1032.1
F9 v1.1
F9 v1.1 B1011
F9 v1.1 B1014
F9 v1.1 B1016

Explanation:

- UNIQUE key word used to eliminate repeat value
- Parenthesis to separate the two where clause condition
- BETWEEN key word used to find value in between two values

Total Number of Successful and Failure Mission Outcomes

Command:

```
%sql SELECT mission_outcome, COUNT(*) as count FROM SPACEXTBL GROUP BY mission_outcome
```

Output:

mission_outcome	COUNT
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Explanation:

- COUNT function is called to return the number of occurrence

Boosters Carried Maximum Payload

Command:

```
%sql SELECT booster_version FROM SPACEXTBL where payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL)
```

Output:

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

Explanation:

- Subquery is used to find the maximum load which is then used in the where clause

2015 Launch Records

Command:

```
%sql SELECT landing__outcome, Date, booster_version, launch_site FROM SPACEXTBL WHERE (landing__outcome = 'Failure (dr
one ship)') and (YEAR(DATE) = 2015)
```

Output:

landing__outcome	DATE	booster_version	launch_site
Failure (drone ship)	2015-01-10	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	2015-04-14	F9 v1.1 B1015	CCAFS LC-40

Explanation:

- YEAR function is called on date object to return the year of that date

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

Command:

```
%sql SELECT landing__outcome, COUNT(*) AS COUNT FROM SPACEXTBL where Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP
BY landing__outcome ORDER BY COUNT DESC
```

Output:

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

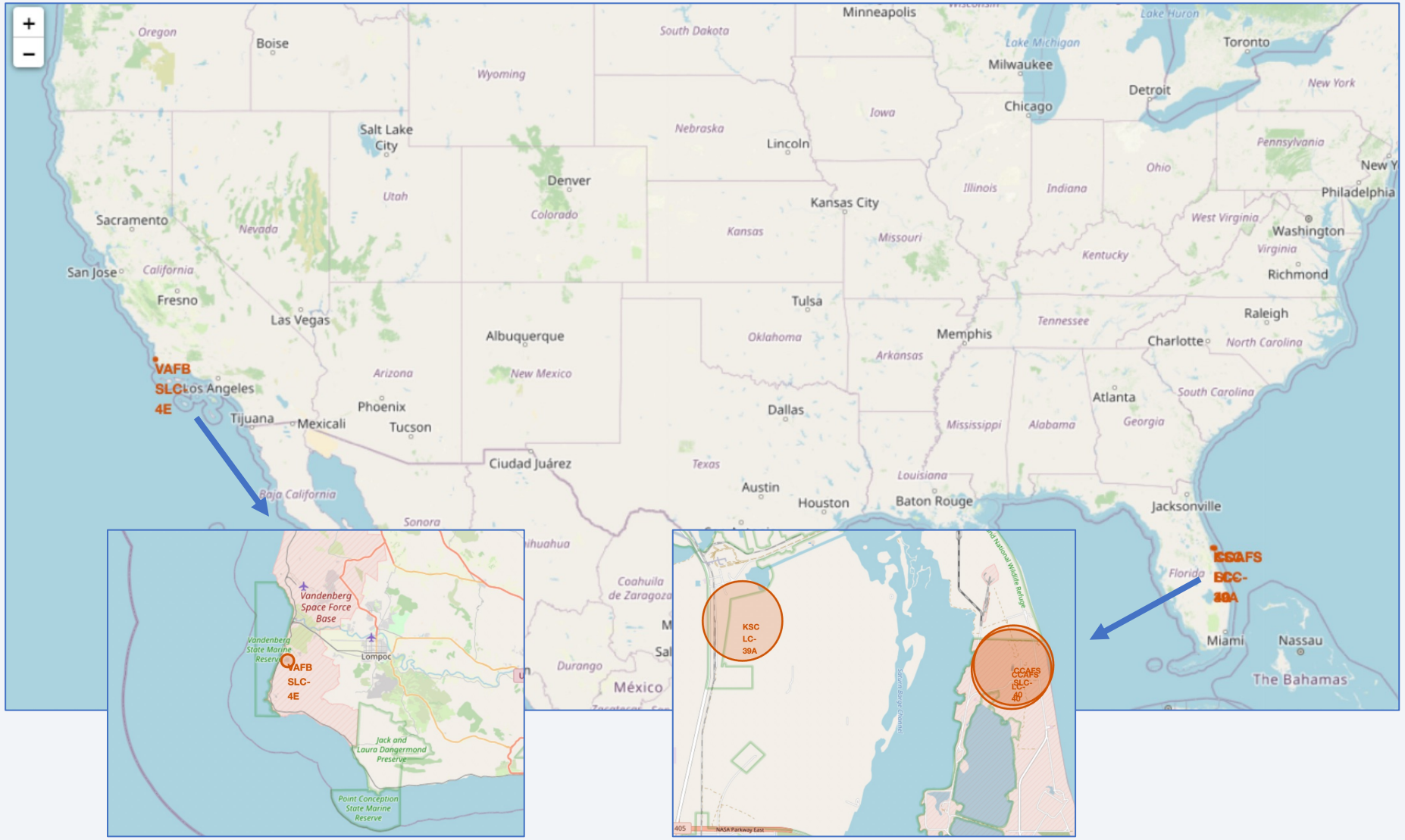
Explanation:

- COUNT function is called get the number of occurrence
- BETWEEN key word is called to evaluate if the date is in the range
- GROUP by is used to aggregate based on a certain attribute
- ORDER BY is used to sort based on a certain attribute

Section 4

Launch Sites Proximities Analysis

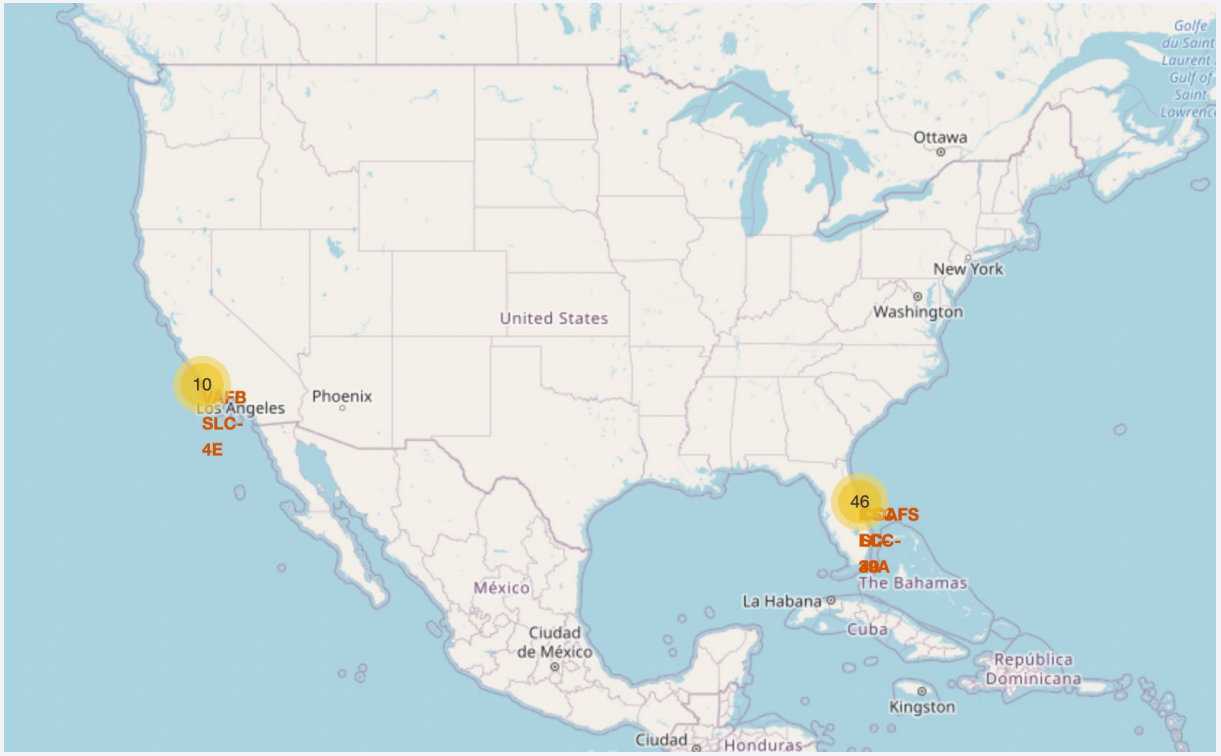
Launch Site Map



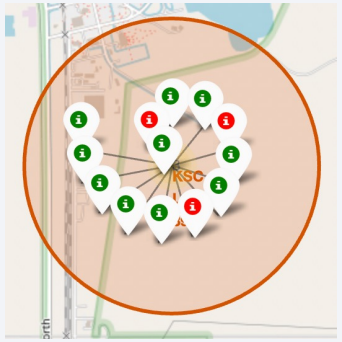
Most launch site in U.S. are located near the coasts. (California and Florida in the map shown in the left)

Launch Record Map

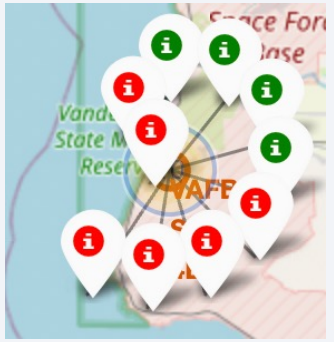
Based on the icon, we can conclude that KSC LC-39A has the highest success rate compared to the other three launch sites



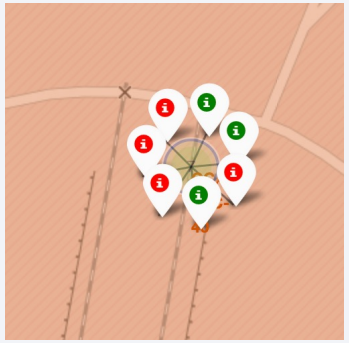
KSC LC-39A



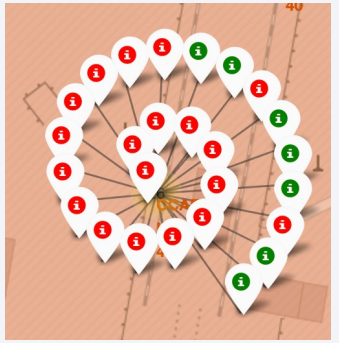
VAFB SLC-4E



CCAFS SLC-40

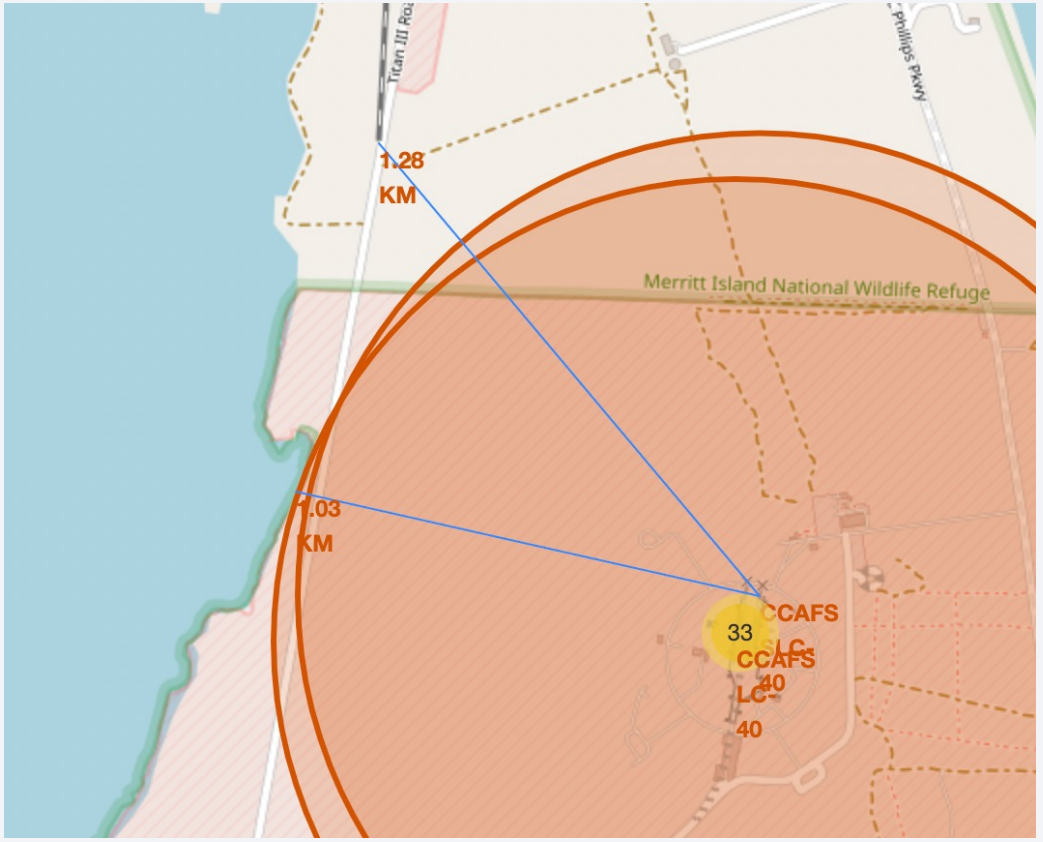


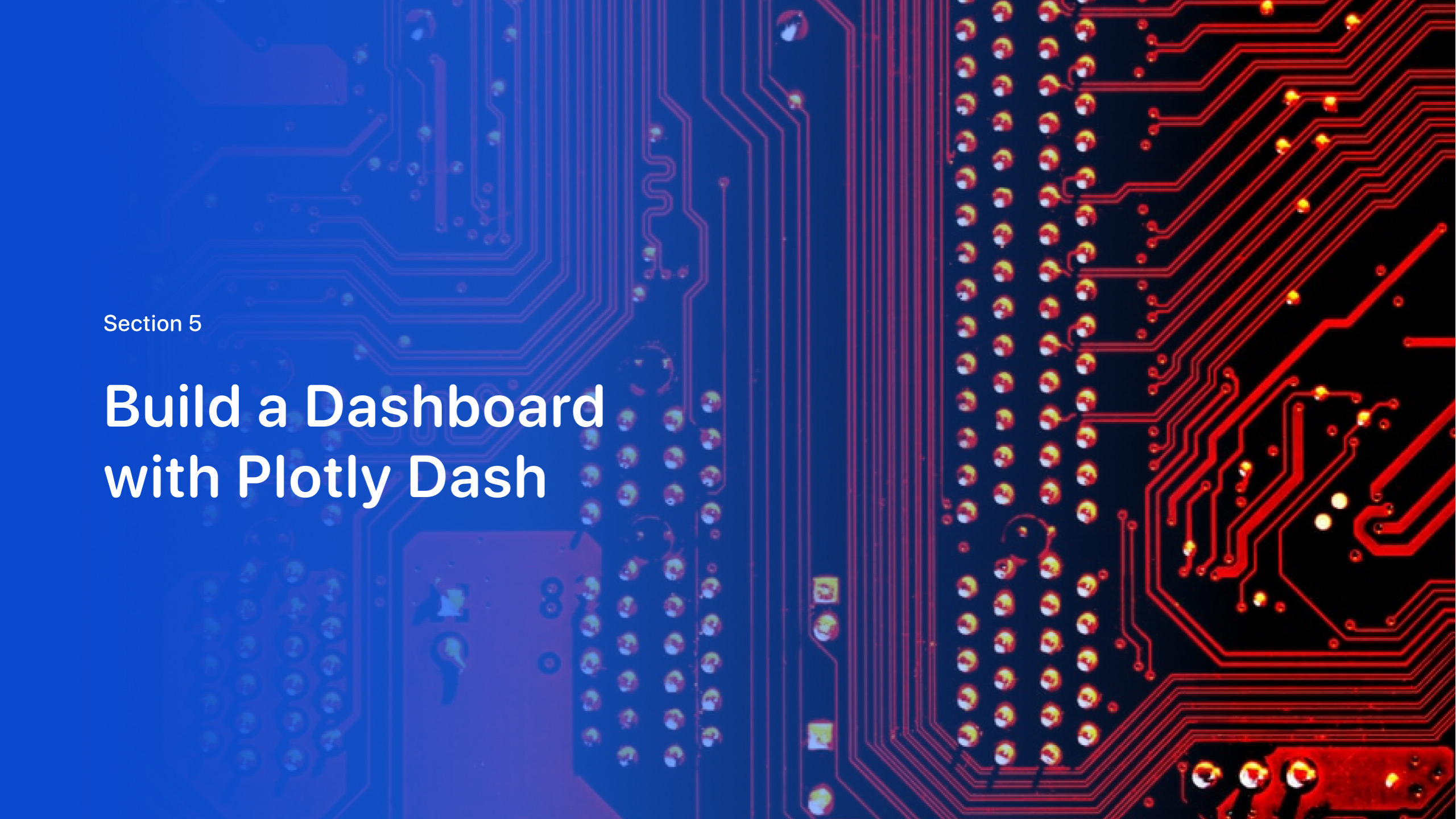
CCAFS LC-40



Launch Site Vicinity Distance Map

The launch site CCAFS SLA-40 is about 1 km away from coast and 1.28 km away from railroad based on this approximation

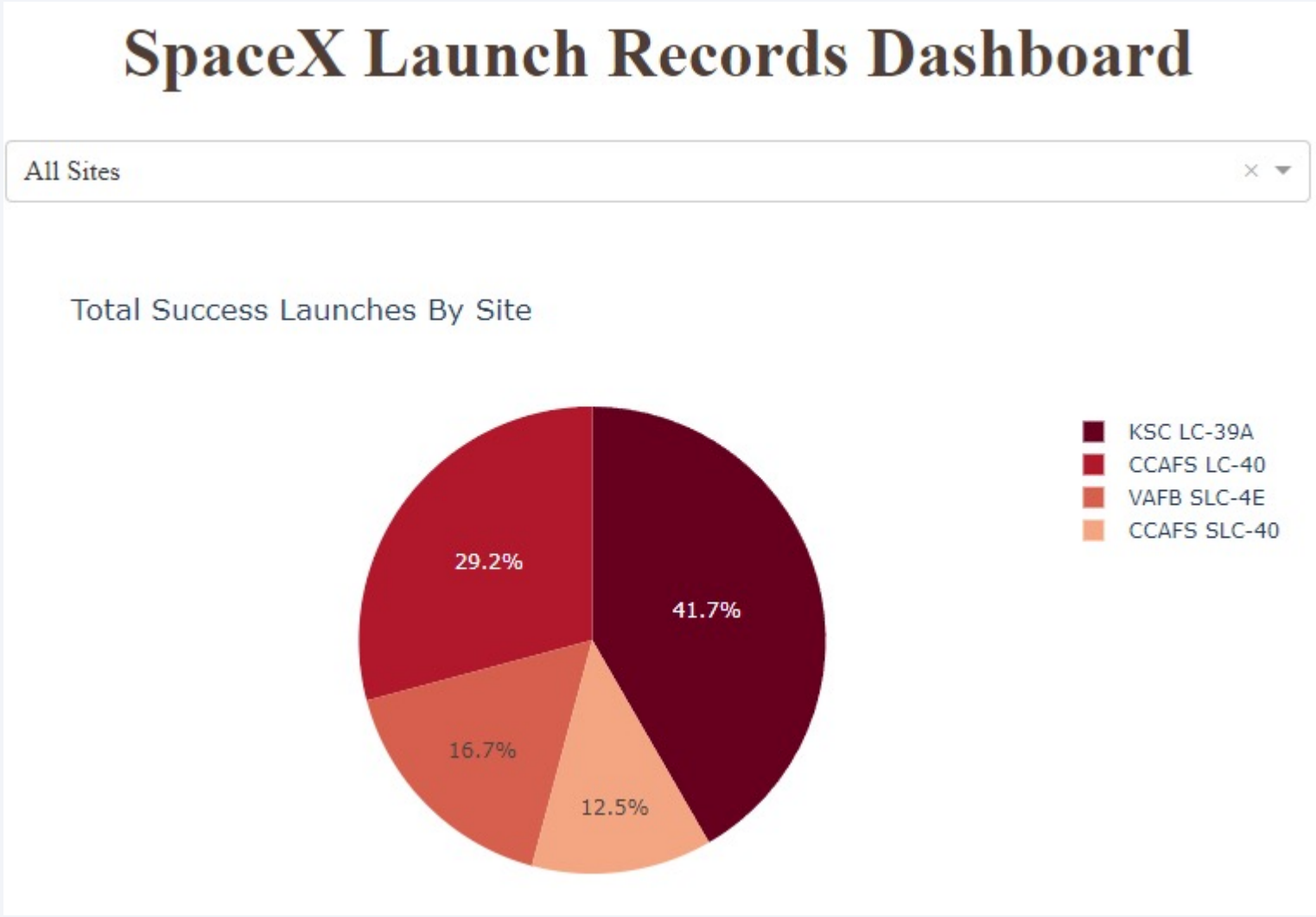




Section 5

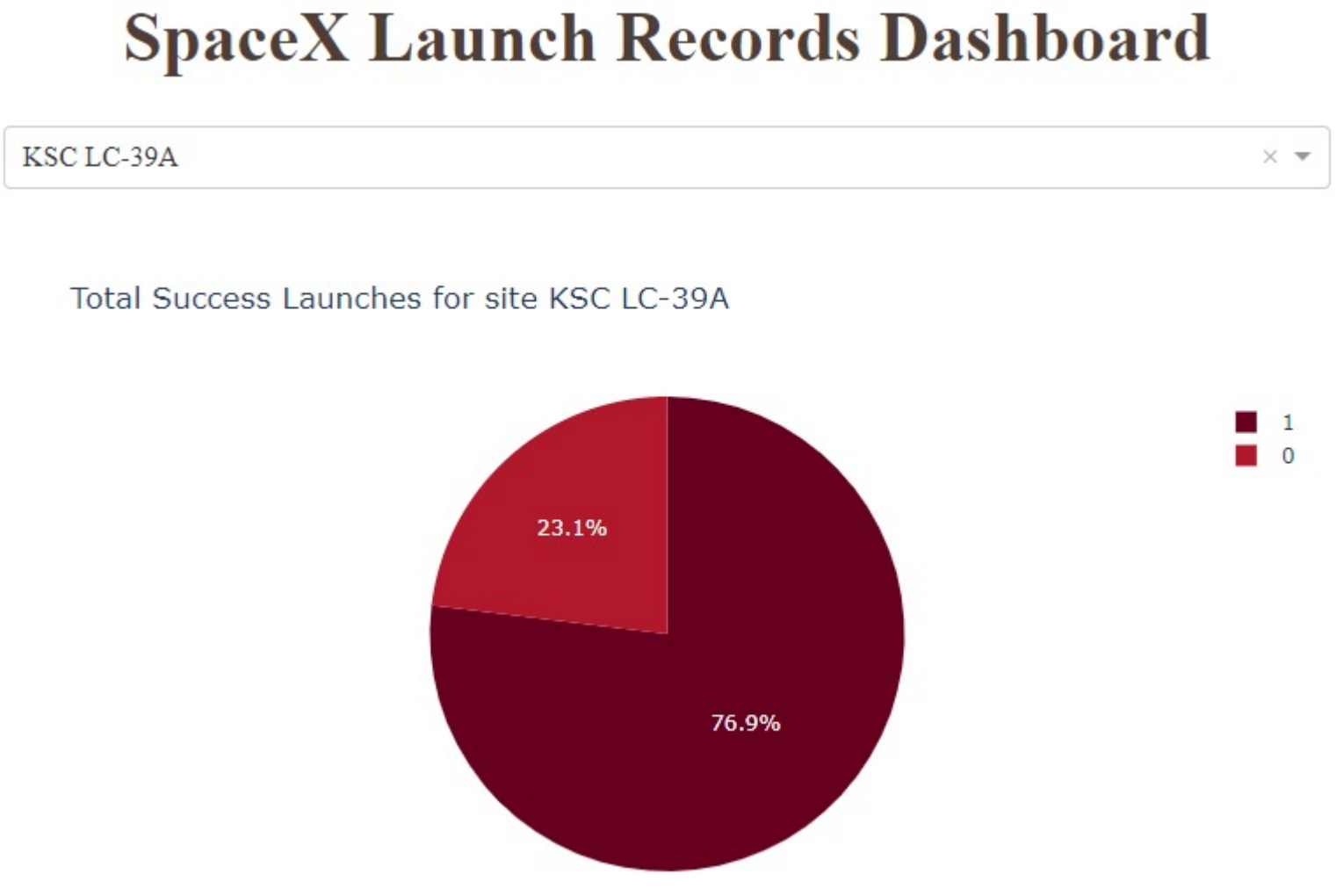
Build a Dashboard with Plotly Dash

Total Success Launches by All Sites



- KSC LC-39A had the most amount of successful launches
- CCAFS SLA-40 has the least amount of successful launches

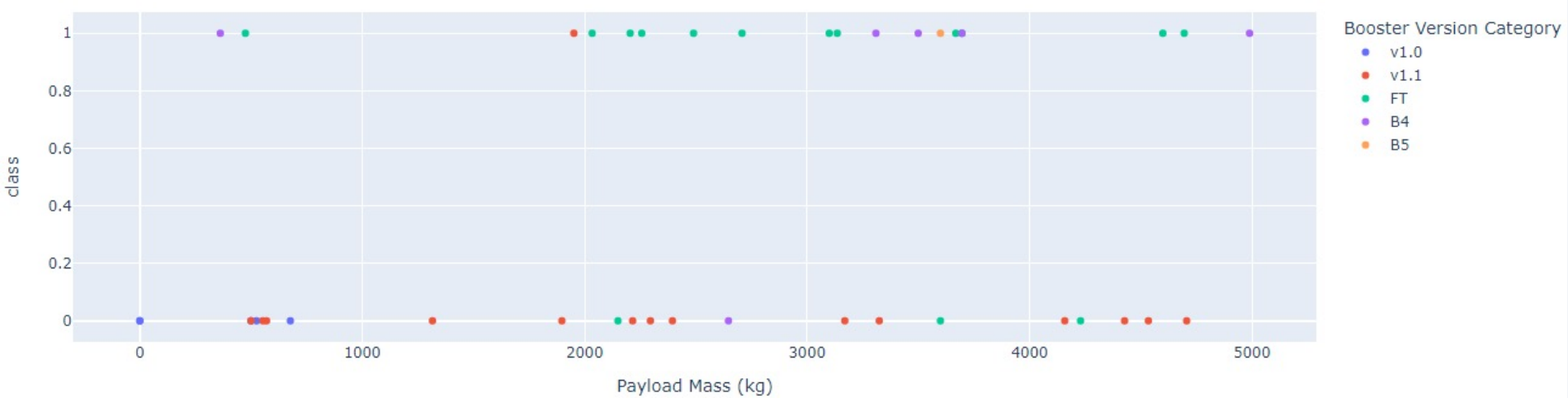
Launch Site with Highest Launch Success Ratio



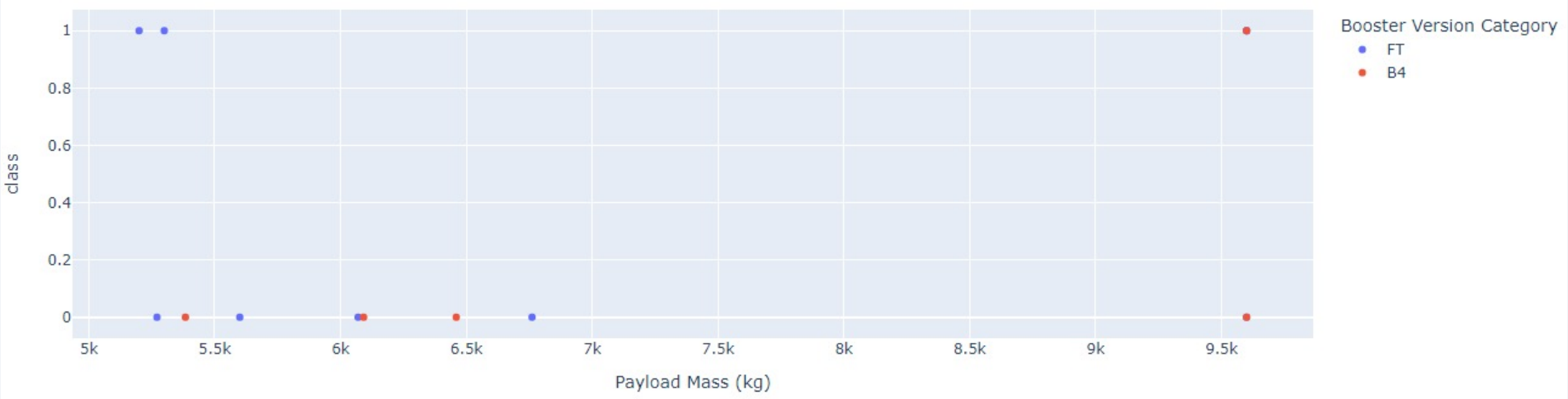
KSC LC-39A has a success rate of 76.9% and a failure rate of 23.1%

Payload vs. Launch Outcome for All Sites

0 – 5000kg Payload:



5000 - 10000kg Payload:



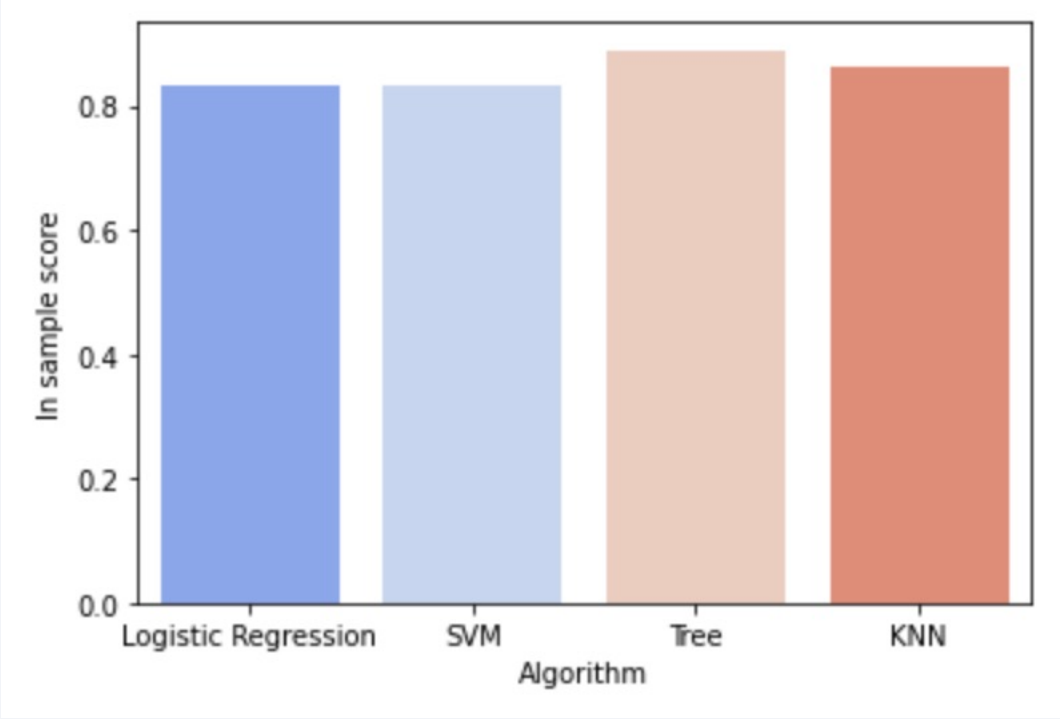
- The success rate is higher when the payload range is between 2000 and 5000
- FT booster has the largest success rate

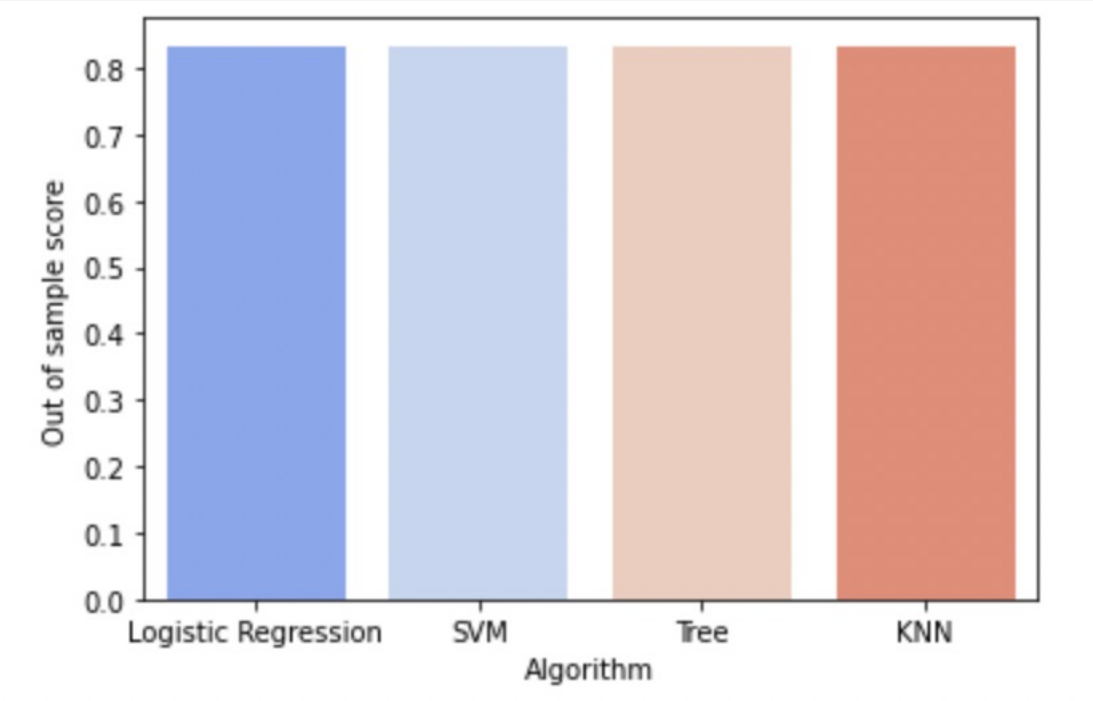
Section 6

Predictive Analysis (Classification)

Classification Accuracy

- In terms of in sample score performance, the decision tree algorithm is the best algorithm with a score of 0.889.
- In terms of out of sample score performance, all four algorithms are tied with a score of 0.833.

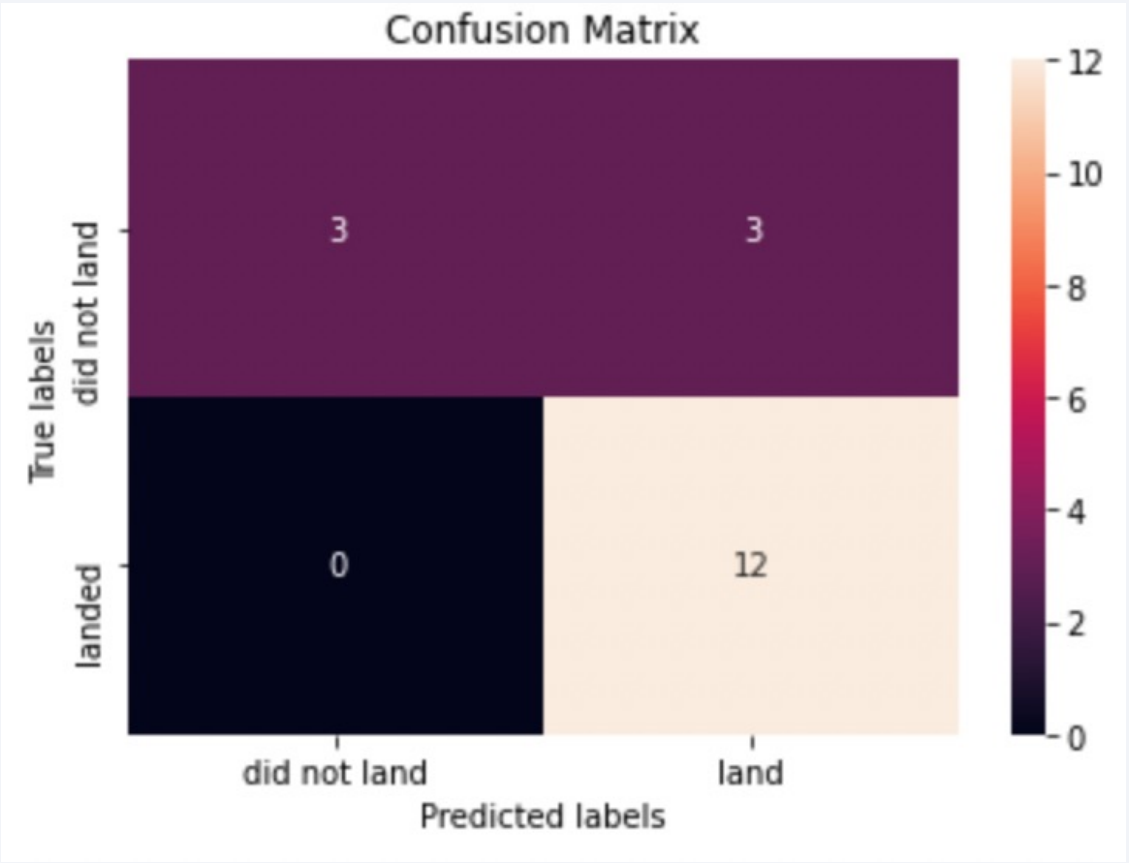




Confusion Matrix

- The algorithm was able to correctly classify all the cases when the first stage rocket landed and predict half of the case correct when the first stage rocket did not land

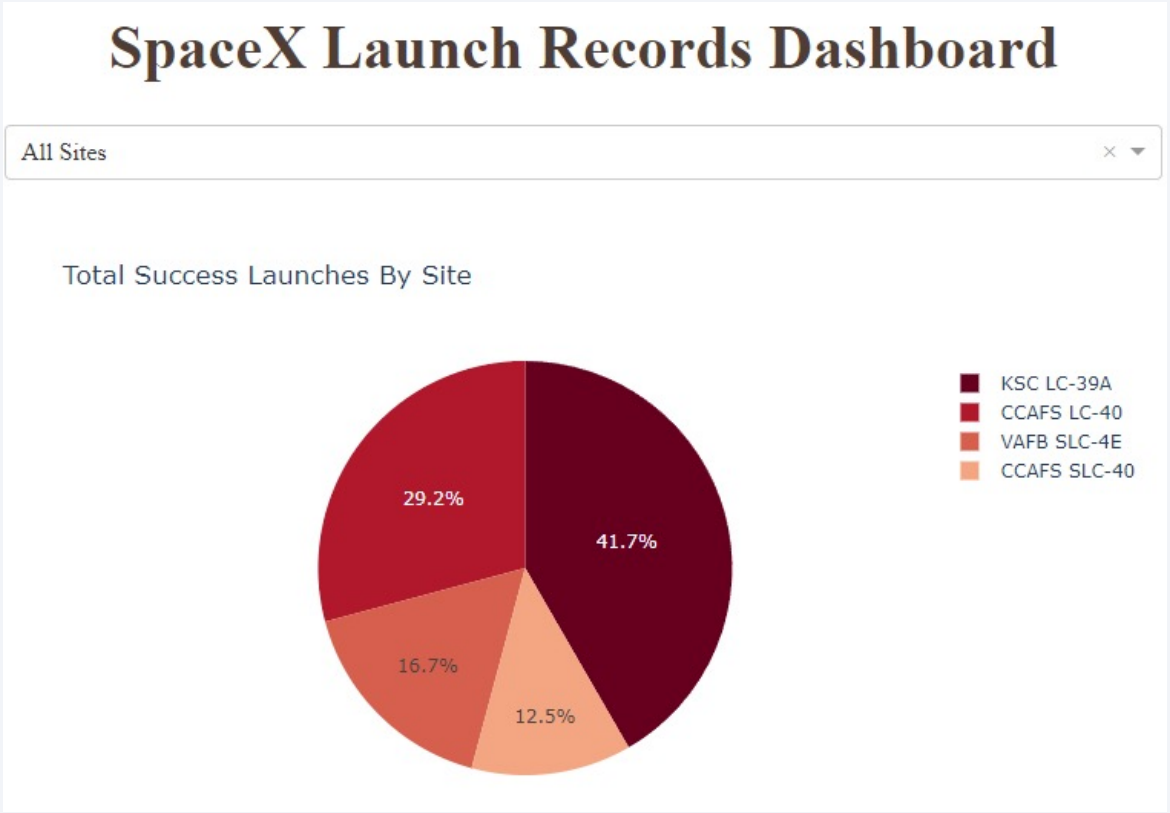
Decision Tree

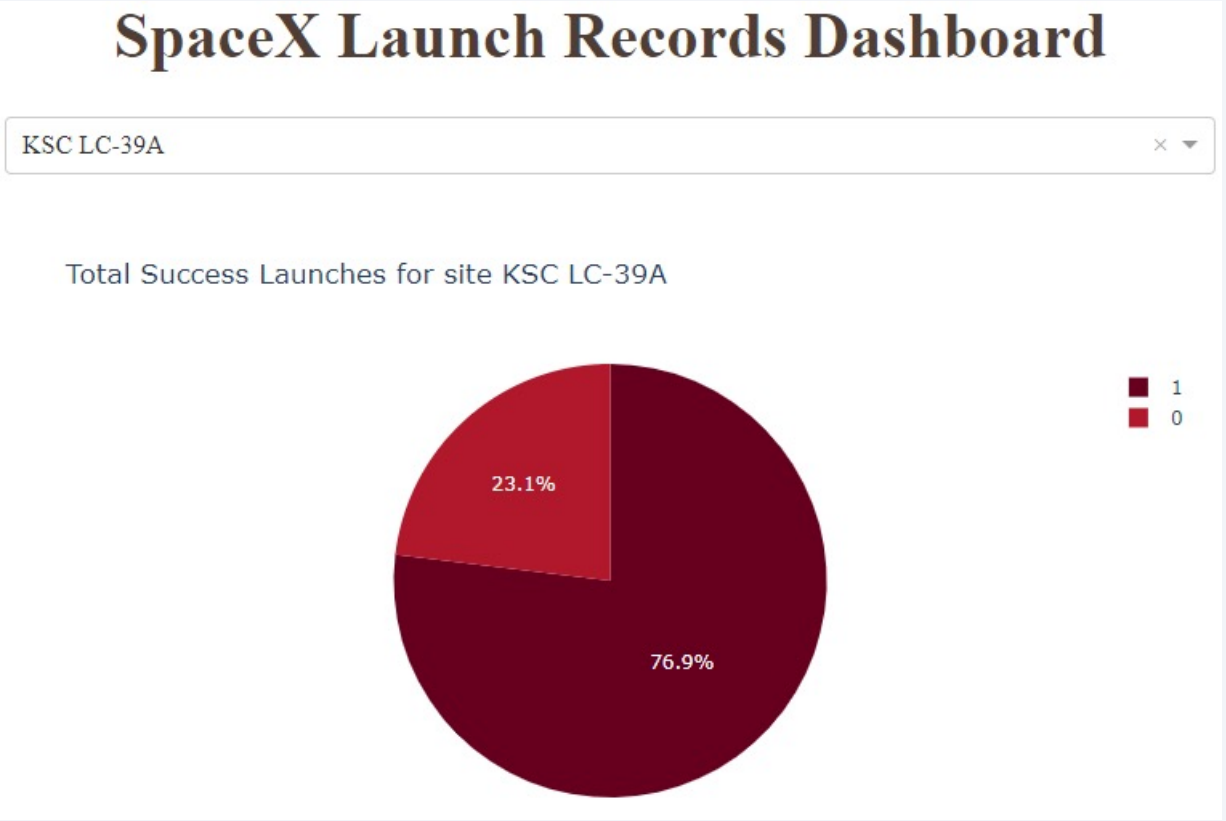


Conclusions

- KSC LC-39A had the highest success rate and the failure cases happened when the payload is either around 6000 or above 16000
- The orbit types with highest success rate are ES-L1, GEO, HEO, and SSO
- The success rate for SpaceX launches have been gradually increasing since 2013 and is sitting at around 80% at 2020
- Best machine learning model for the current dataset is Decision Tree algorithm

Appendix


SpaceX Launch Records Dashboard
All Sites
Total Success Launches By Site
29.2%
41.7%
16.7%
12.5%
KSC LC-39A
CCAFS LC-40
VAFB SLC-4E
CCAFS SLC-40


SpaceX Launch Records Dashboard
KSC LC-39A
Total Success Launches for site KSC LC-39A
23.1%
76.9%
1
0

Thank you!