IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ryan Francway
5/8/2023

Outline

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

Executive Summary

- **Summary of methodologies**
  - Data collection through use of Web Scrapping & APIs
  - Data Wrangling & Analysis through use of SQL & Python
  - Interactive Visual mapping with Folium
  - Predictive Analysis for different models using Machine Learning
- **Summary of all results**
  - Exploratory Data Analysis & Visualization results
  - Predictive Analysis results

Introduction

- **Project background and context:**
  - The overall goal of this project is to predict whether or not the first stage of the Falcon 9 rocket will land successfully. On the SpaceX website, the estimated cost of Falcon 9 rocket launches is advertised to be 62 million dollars; other companies advertise an estimated cost of 165 million dollars or more. Through successful landings of the first stage of the Falcon 9 rocket, SpaceX can reuse this key component for future rockets.

- **Problems to address:**
  - Considering all factors, which ones contribute the most to a successful landing?
  - What degree of impact does each interaction between rocket variables have on the outcome?
  - What operating conditions are favorable for SpaceX rocket launches?

Section 1

Methodology

Methodology

Executive Summary

- **Data collection methodology:**
  - SpaceX Rest API
  - Web Scraping from Wikipedia
- **Perform data wrangling**
  - One-Hot Encoding categorical data fields as binary vectors for machine learning algorithms and dropping null or irrelevant values
- **Perform exploratory data analysis (EDA) using visualization and SQL**
  - Generated scatter and bar plots to analyze patterns between variables
- **Perform interactive visual analytics using Folium and Plotly Dash**
  - Folium & Plotly Dash Visualizations
- **Perform predictive analysis using classification models**
  - Build, tune, evaluate classification models

Data Collection

- **Data collection** is the process of gathering and measuring information on variables of interest, in an established systematic fashion that enables one to answer stated research questions, test hypotheses, and evaluate outcomes.
- For this project, the data sets were collected through SpaceX's API & Web Scraped from the SpaceX Wikipedia Page.

Data Collection – SpaceX API

[Data Collection API URL]

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

response = requests.get(spacex_url)
```

```
# Call getBoosterVersion
getBoosterVersion(data)

# Call getLaunchSite
getLaunchSite(data)

# Call getPayloadData
getPayloadData(data)

# Call getCoreData
getCoreData(data)
```

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = launch_dict_df.loc[launch_dict_df['BoosterVersion']!='Falcon 1'
data_falcon9.head()
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Get response API | Convert response to .JSON file | Apply custom functions to normalized data | Assign list to dictionary | Filter dataframe and export to .csv file type

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

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857
5	8	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.561857
6	10	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007	-80.577366	28.561857
7	11	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0	B1003	-120.610829	34.632093
8	12	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B1004	-80.577366	28.561857

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

```
# Create a data from launch_dict
launch_dict_df = pd.DataFrame(launch_dict)
```

Filtered Dataframe

Data Collection - Scraping

[Data Collection with Web Scraping URL]

Get response from URL

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

Create BeautifulSoup object

```
soup = BeautifulSoup(r.content)
```

Find tables

```
html_tables = soup.find_all('table')
first_launch_table = html_tables[2]
print(first_launch_table)
```

Get column names

```
column_names = []
labels = first_launch_table.find_all('th')
for label in labels:
    name = extract_column_from_header(label)
    if name != None:
        if len(name) > 0:
            column_names.append(name)
```

Create Dictionary and append data

```
launch_dict= dict.fromkeys(column_names)
```

Empty Dictionary with Keys

```
# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

Convert Dictionary to Dataframe

Export to .CSV

	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 (COTS)\nNRO	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA (COTS)	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA (CRS)	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA (CRS)	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10

Data Wrangling

[EDA Data Wrangling Lab URL]

Data wrangling is the process of transforming messy or unorganized data for the purpose of analysis.

In regards to this project, the focus was mainly on:

Calculating amount of launches from each site

```
df['LaunchSite'].value_counts()
```

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

Calculating occurrence for each type of orbit

```
df['Orbit'].value_counts()
```

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

Calculating the number and occurrence of mission outcome of the oribits

```
landing_outcomes = df['Outcome'].value_counts()
```

True ASDS	41
None None	19
True RTLS	14
False ASDS	6
True Ocean	5
False Ocean	2
None ASDS	2
False RTLS	1

Creating training labels based on mission outcome

```
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 Ocear
6 None ASDS
7 False RTLS
```

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

    landing_class.append(i)
```

Finally, Exporting results for further analysis

```
df.to_csv("csvs/dataset_part_2.csv", index=False)
```

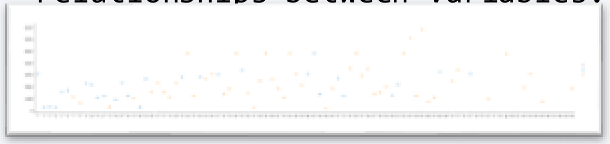
EDA with Data Visualization

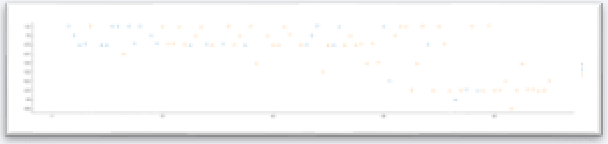
[Exploratory Data Analaysis with Visualization URL]

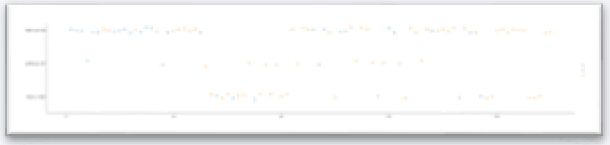
Scatter Plots Drawn:

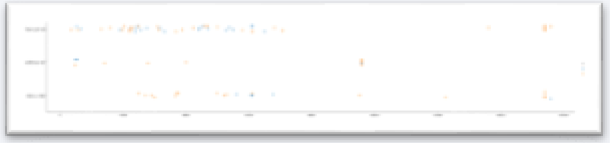
Pay load Mass (kg) and Flight Number
Launch Site and Flight Number
Launch Site and Pay load Mass (kg)
Orbit and Flight Number
Orbit and Pay load Mass (kg)

A scatter plot (aka scatter chart, scatter graph) uses dots to represent values for two different numeric variables. The position of each dot on the horizontal and vertical axis indicates values for an individual data point. Scatter plots are used to observe relationships between variables.





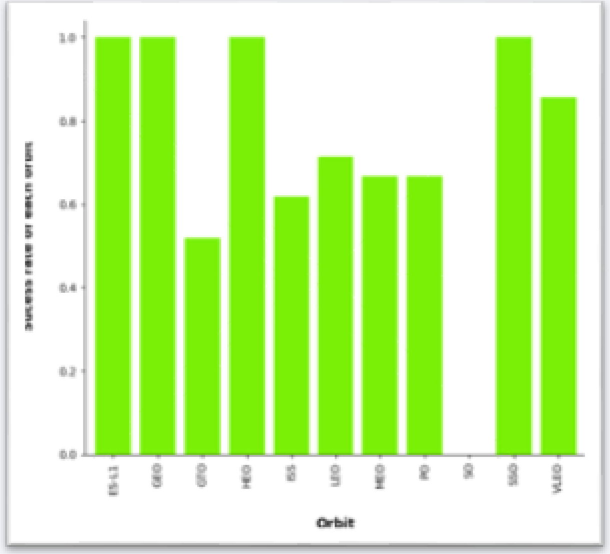




Bar Plot Drawn:

Orbit VS. Class (Success Rate)

A bar chart describes the comparisons between categories. One of the axis of the plot represents the specific categories being compared, while the other axis represents the measured values corresponding to those categories.



Line Plot Drawn:

Launch Success Yearly Trend

Line charts are a fundamental chart type generally used to show change in values across time.

EDA with SQL

[Exploratory Data Analysis with SQL URL]

Structured query language (SQL) is a programming language for storing and processing information in a relational database. A relational database stores information in tabular form, with rows and columns representing different data attributes and the various relationships between the data values. Here we use IBM's Db2 for Cloud, which is a fully managed SQL Database provided as a service.

Connecting to Database and Environment Setup:

```
!pip install --force-reinstall ibm_db==3.1.0 ibm_db_sa==0.3.3
!pip install sqlalchemy==1.3.24
!pip uninstall ipython-sql -y
!pip install ipython-sql==0.4.1
%load_ext sql

%sql ibm_db_sa://my-username:my-password@my-hostname:my-port/my-db-name?security=SSL
%sql <query>
```

SQL queries performed for this assignment:

- 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.
- names of the booster_versions which have carried the maximum payload mass using 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.

Build an Interactive Map with Folium

Interactive Visual Analytics with Folium Lab URL

Folium builds on the data wrangling strengths of the Python ecosystem and the mapping strengths of the leaflet.js library. For this assignment, we used the longitude and latitude coordinates for each launch site that labels each site accordingly. We also added circle markers to make each launch site easily identifiable on the map. The number of Successful/Failure launches is shown within the circle markers as well.

Map Objects	Code	Result
Map Marker	folium.Marker()	Map object to mark coordinates on a map.
Icon Marker	folium.Icon()	Create an icon on map.
Circle Marker	folium.Circle()	Create a circle where Marker is being placed.
PolyLine	Folium.PolyLine()	Create a line between points.
Marker Cluster Object	MarkerCluster()	Simplify a map containing many markers having the same coordinate.
Ant Path	foliums.plugins.AntPath()	Create an animated line between points.

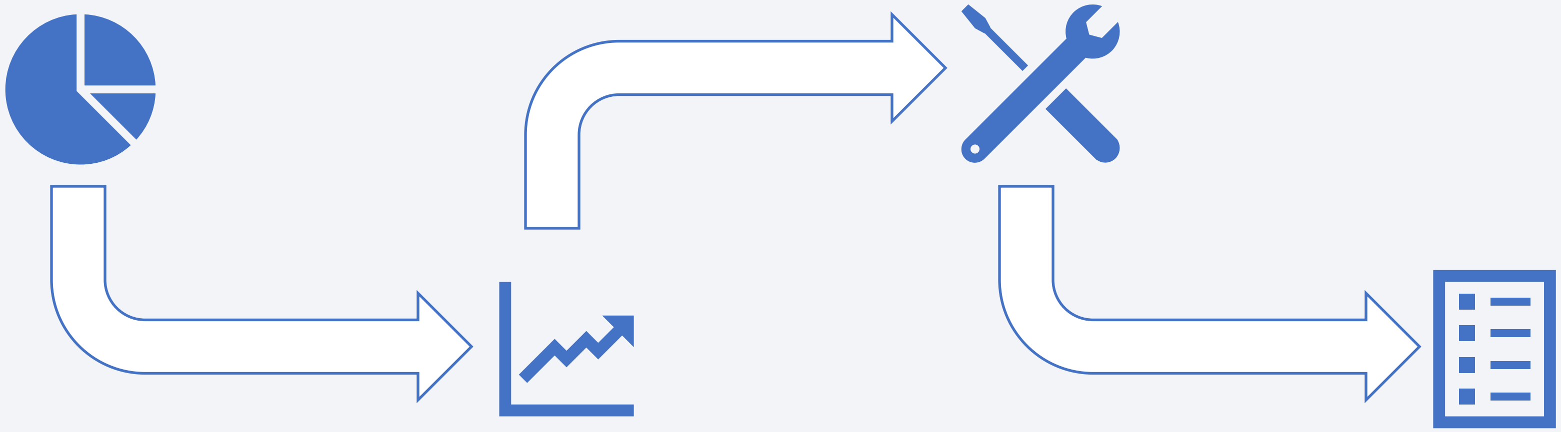
Build a Dashboard with Plotly Dash

[Plotly Dash Lab URL]

During this lab, I built an interactive dashboard with graphs to help analyze the SpaceX data.

Plots/Graphs & Interactions:

- Plotted Pie Charts to show the total number of launches by each site.
- Plotted Scatter Graphs to show the correlation between Outcome and Payload Mass (kg) for the unique booster versions.
- Created a dropdown menu for launch sites
- Created a rangeslider for Payload Mass (kg) range selection



Predictive Analysis (Classification)

[Machine Learning Prediction URL]

- Developing Model
- Import Libraries & define auxillary functions
- Load the dataframes
- Define NumPy arrays using 'Class' Column
- Standardize & transform data
- Split data into training and test sets
- Select Machine Learning Algorithms
  - Logistic Regression
  - Support Vector Machine
  - Decision Tree Classifier
  - K-Nearest Neighbor
- Set parameters and algorithms to GridSearchCV
- Fit dataset into GridSearchCV objects for each algorithm
- Train the models
- Check accuracy for each model
- Gather most optimal hyper parameters for each type of algorithm
- Identify most accurate models

Results

Exploratory Data Analysis Results

Interactive analytics demo in screenshots

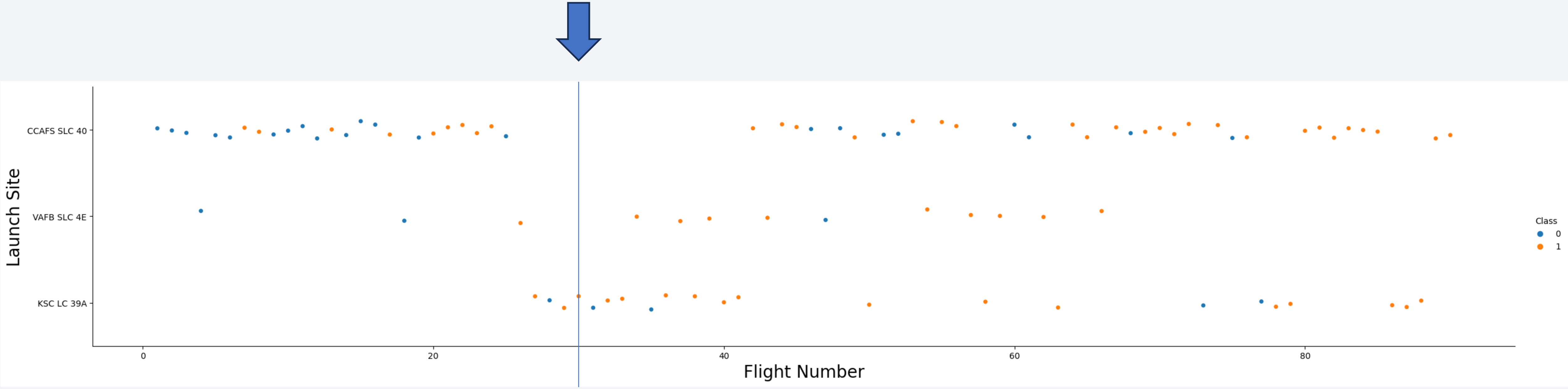
Predictive analysis results

Section 2

Insights drawn from EDA

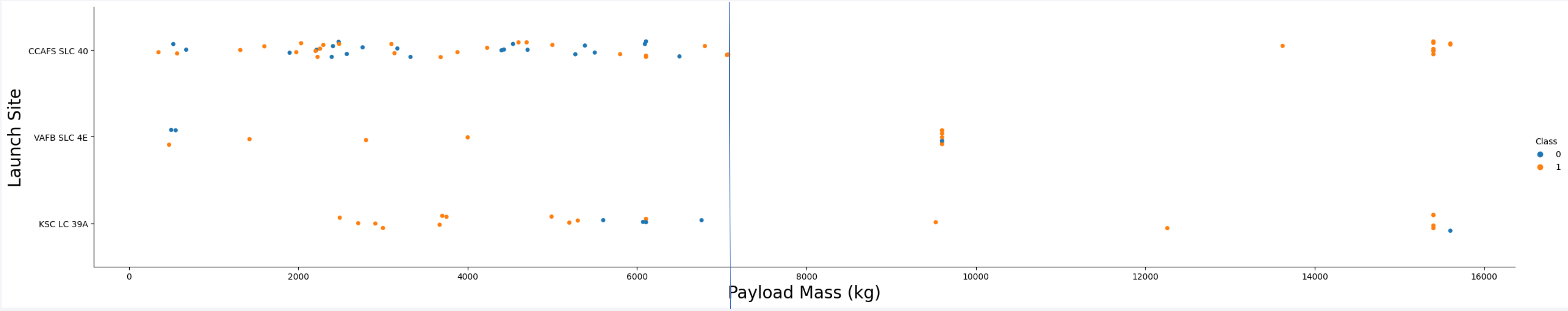
Flight Number vs. Launch Site

With higher flight numbers (greater than 30) the success rate for Falcon 9 rocket is increasing.

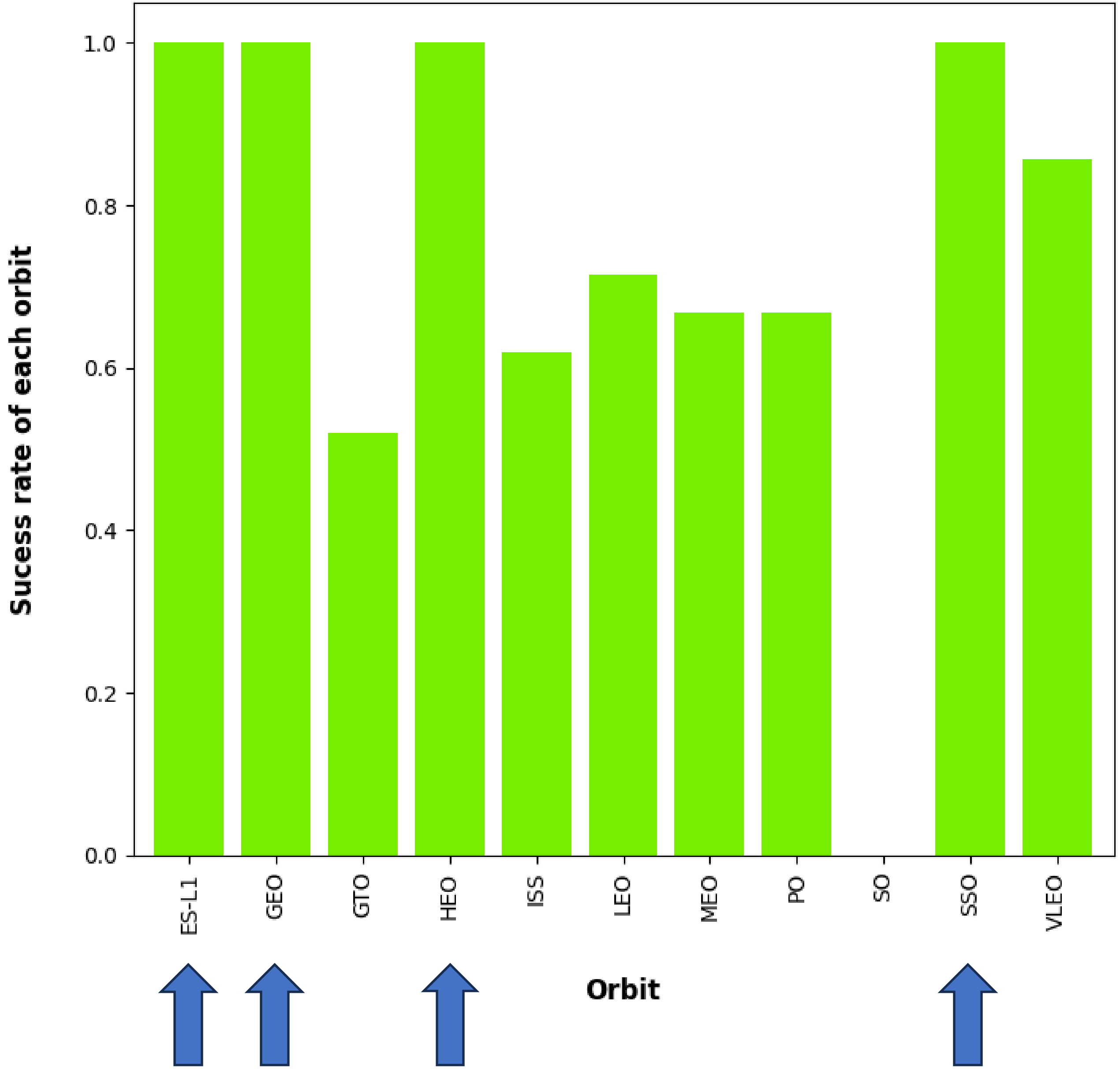


Payload vs. Launch Site

The greater the payload mass (greater than 7000 kg) the higher the success rate for the Falcon 9 rocket. Could likely use more data points to determine whether or not this insight is correct.



Success Rate vs. Orbit Type

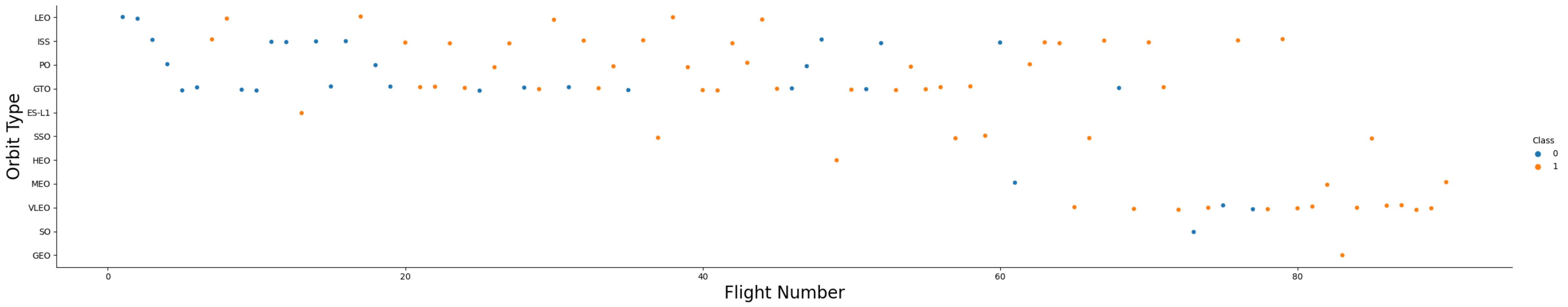


Highest Success rate per orbit type:

- ES-L1
- GEO
- HEO
- SSO

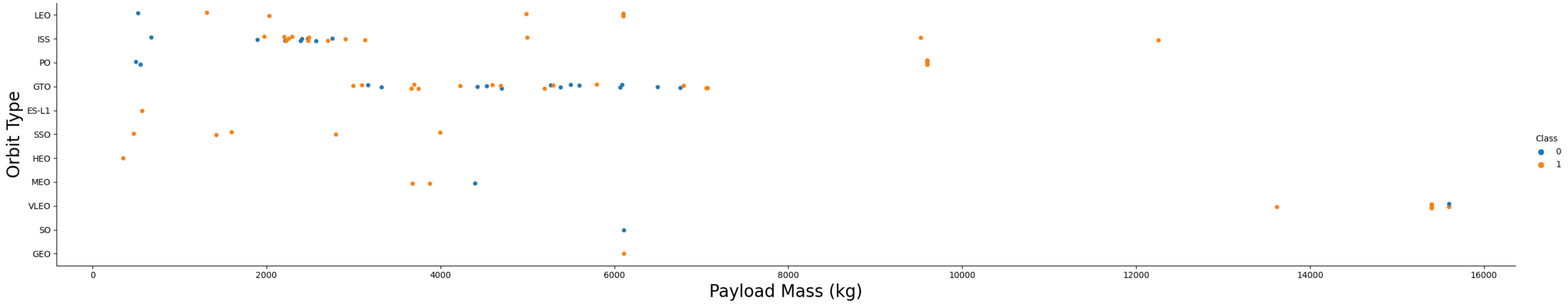
Flight Number vs. Orbit Type

We can identify that the LEO type increases with number of flights, whereas, in regards to GTO type, there is no relationship between number of flights and type of orbit.



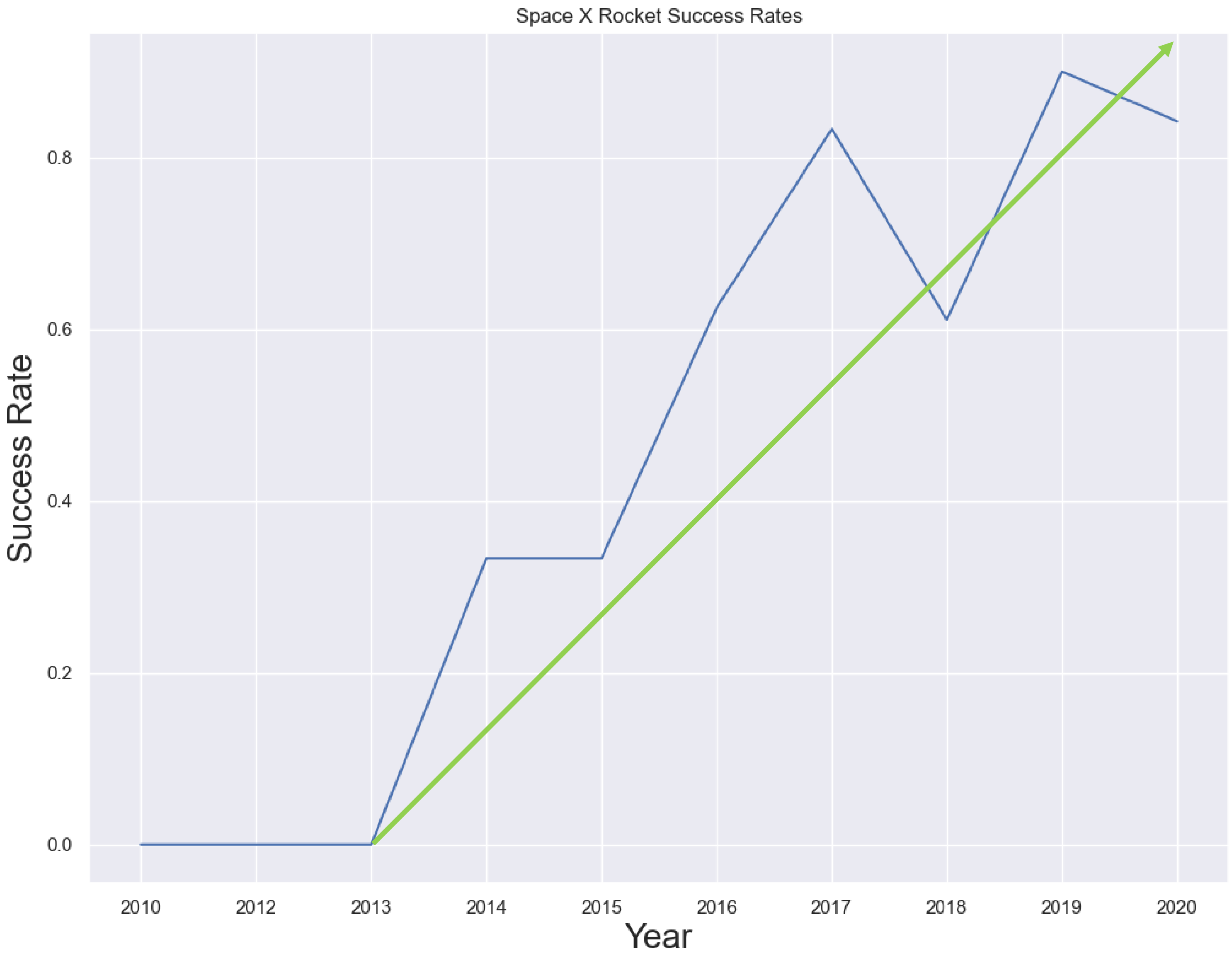
Payload vs. Orbit Type

We can identify that heavier payloads have a positive influence on LEO & ISS types. While heavier payloads, have a negative influence on MEO, GTO, & VLEO types.



Launch Success Yearly Trend

We can identify that success rate since 2013 has constantly increased.



All Launch Site Names

Using UNIQUE() in this query will only return each distinct launch site from the SpaceX table. As shown in the figures below.

```
%sql select Unique(LAUNCH_SITE) from SPACEX;
```

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

Using keyword LIMIT 5 we return the first 5 records from the SpaceX table. With keyword LIKE specifying 'CCA%', we return any launch site that begins with 'CCA.'

```
%sql select LAUNCH_SITE from SPACEX where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	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
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-[illegible]	15:10:00	F9 v1.0 B0007	CCAFS LC-[illegible]	SpaceX CRS-2	677	LEO [illegible]	NASA (CRS)	Success	No attempt

Total Payload Mass

Using SUM() in the select statement we can find the total of the payload_mass_kg_ column. Specifying with CUSTOMER = 'NASA (CRS)' in the where statement will only return the total payload mass where the customer is 'NASA (CRS).'

```
%sql select SUM(PAYLOAD_MASS_KG_) from SPACEX where CUSTOMER = 'NASA (CRS)';
```

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

Using AVG() in the select statement we can find the average of the payload_mass_kg_ column. Specifying with BOOSTER_VERSION = 'F9 v1.1' in the where statement will only return the total payload mass where the booster version is 'F9 v1.1.'

```
%sql select AVG(PAYLOAD_MASS_KG_) from SPACEX where BOOSTER_VERSION = 'F9 v1.1';
```

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

Using MIN() in the select statement we can find the lowest or earliest date in the DATE column. Specifying with LANDING_OUTCOME = 'Success (ground pad)' in the where statement will only return the landing outcome where LANDING_OUTCOME is equal to 'Success (ground pad).'

```
%sql select MIN(DATE) from SPACEX where LANDING_OUTCOME = 'Success (ground pad)';
```

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Selecting only BOOSTER_VERSION, the WHERE clause filters the dataset to LANDING_OUTCOME = Success (drone ship). The AND clause specifies additional filter conditions. In this case, the payload mass being greater than 4000 kg and less than 6000 kg.

```
%sql select BOOSTER_VERSION from SPACEX where LANDING_OUTCOME = 'Success (drone ship)' and PAYLOAD_MASS_KG_ > 4000 and < 6000 ;
```

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

Using COUNT() in the select statement and LIKE in the WHERE clause while specifying 'Success%' or 'Failure%' for each query will return the total number of rows that match that criteria.

```
%%sql
select COUNT(MissionOutcome) AS SuccessOutcome from SPACEX where MissionOutcome LIKE 'Success%';

select COUNT(MissionOutcome) AS FailureOutcome from SPACEX where MissionOutcome LIKE 'Failure%';
```

	successoutcome
0	100

	failureoutcome
0	1

Boosters Carried Maximum Payload

Using a subquery in the WHERE clause can help to identify which booster versions have carried the MAX() of the PAYLOAD_MASS_KG_ column.

```
%sql select BOOSTER_VERSION as boosterversion from SPACEX where PAYLOAD_MASS_KG_=(select max(PAYLOAD_MASS_KG_) from SPACEX);
```

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

Using EXTRACT() in the where clause of the query to select only the year 2015 from the DATE column, we can show the LANDING_OUTCOMES = 'Failure (drone ship)' that happened that year.

```
%sql select LANDING_OUTCOMES, BOOSTER_VERSION, LAUNCH_SITE from SPACEX where LANDING_OUTCOMES = 'Failure (drone ship)' and extract(YEAR from DATE) = 2015;
```

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

Selected Landing Outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 to 2010-03-20.

Applied the GROUP BY clause to group the landing outcomes and the order by clause to order the grouped landing outcome in descending order.

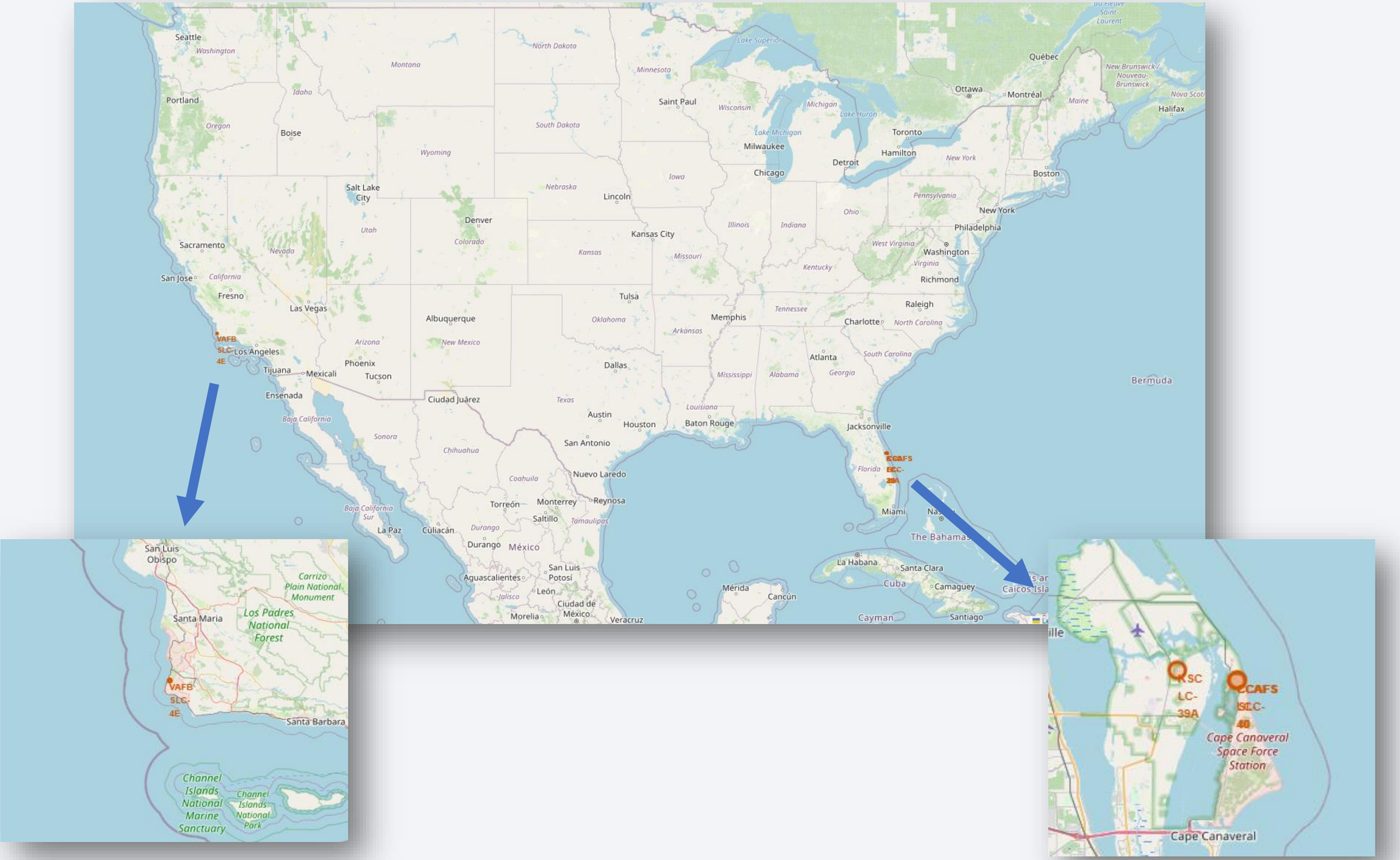
```
%sql SELECT LANDING_OUTCOME, COUNT(LandingOutcome) FROM SPACEX WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'GROUP BY LandingOutcome ORDER BY COUNT(LandingOutcome) DESC;
```

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

Section 3

Launch Sites Proximities Analysis

All Launch Sites Using Folium



Using the Folium map, we can identify that all of the launch sites are located near a coast in the United States.

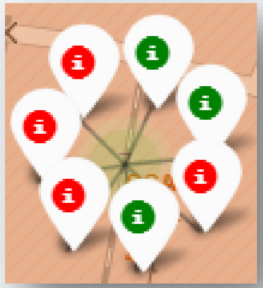
Color-labeled Launch Outcomes

Green Markers indicate successful launches

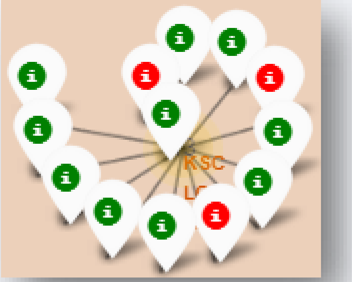
Red Markers indicate failed launches

From the use of the color-labeled Folium map, we can identify that launch site **KSC LC-39A** has the **highest percentage of successful launches.**

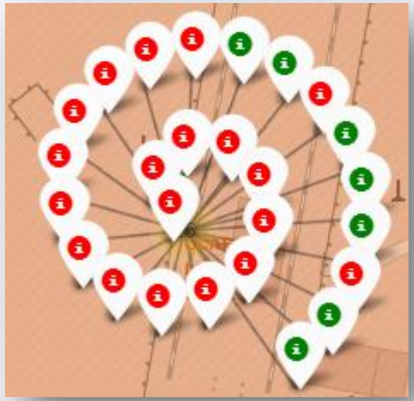
CCAFS SL-40



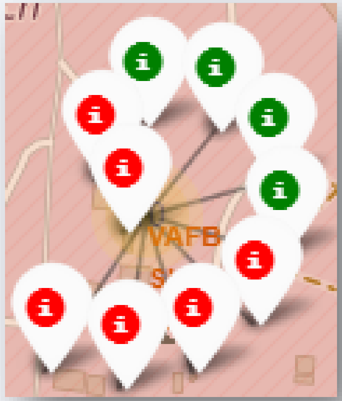
KSC LC-39A



CCAFS LC-40

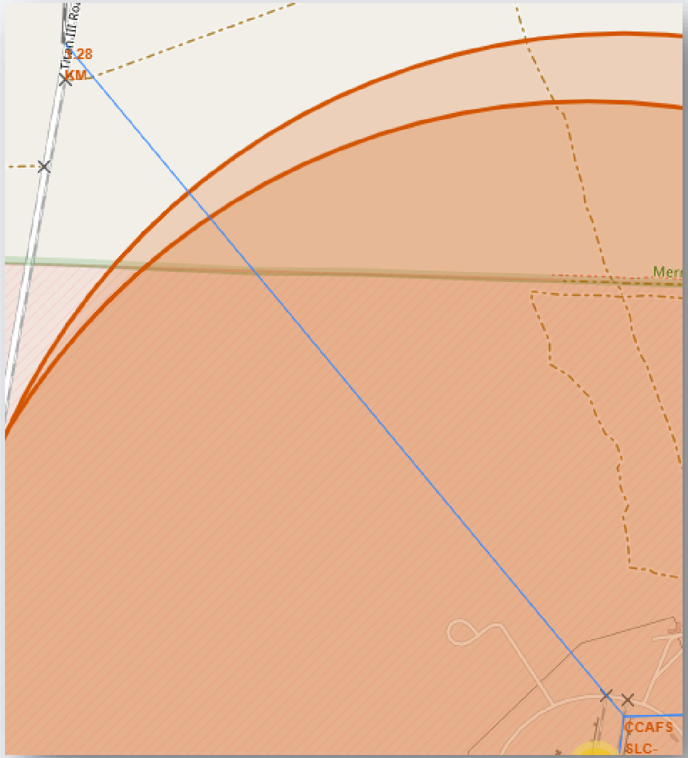


VAFB SLC-4E

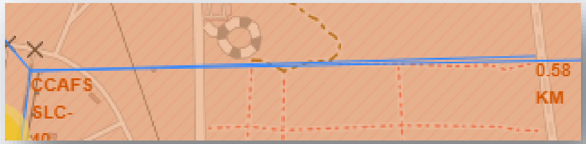


Distance from Important Launch Site Factors

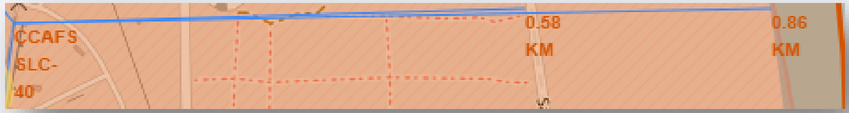
CCAFS SLC-40 distance from nearest railway is 1.28 KM.



CCAFS SLC-40 distance from nearest highway is 0.58 KM.



CCAFS SLC-40 distance from nearest coastline is 0.86 KM.



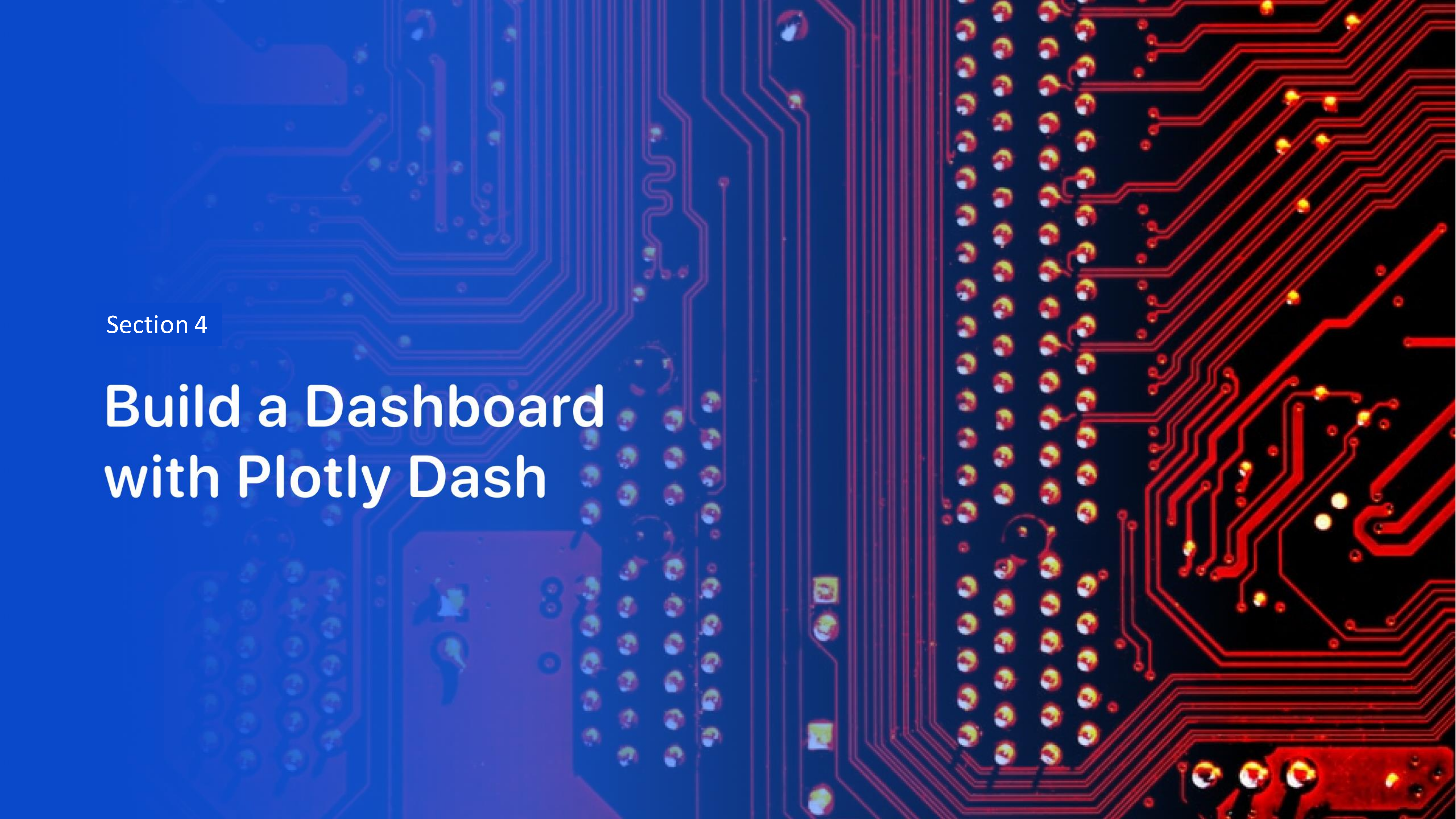
CCAFS SLC-40 distance from the nearest densely populated city is 51.42 KM.

My Findings

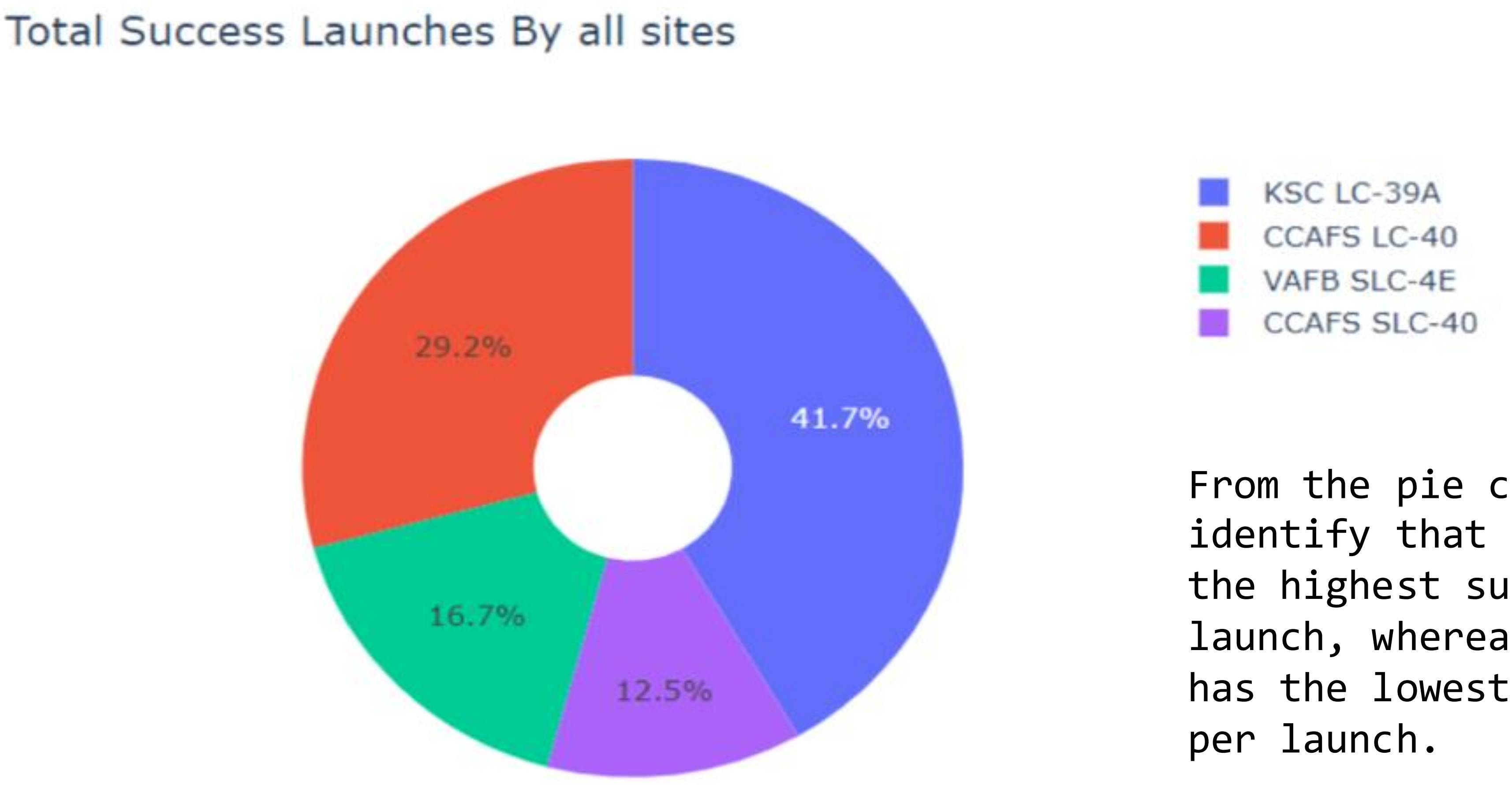
- Launch sites are in close proximity to the equator to minimize fuel consumption by using Earth's ~ 30km/sec eastward spin to help spaceships get into orbit.
- Launch sites are in close proximity to coastlines so they can fly over the ocean during launch, for at least two safety reasons-- (1) crew has option to abort launch and attempt water landing (2) minimize risk to people and property from falling debris.
- Launch sites are in close proximity to highways, which allows for ease of access for required personnel.
- Launch sites are in close proximity to railways, which allows transport for heavy materials or cargo.
- Launch sites are not in close proximity to cities, which minimizes danger to densely populated areas.

Section 4

Build a Dashboard with Plotly Dash

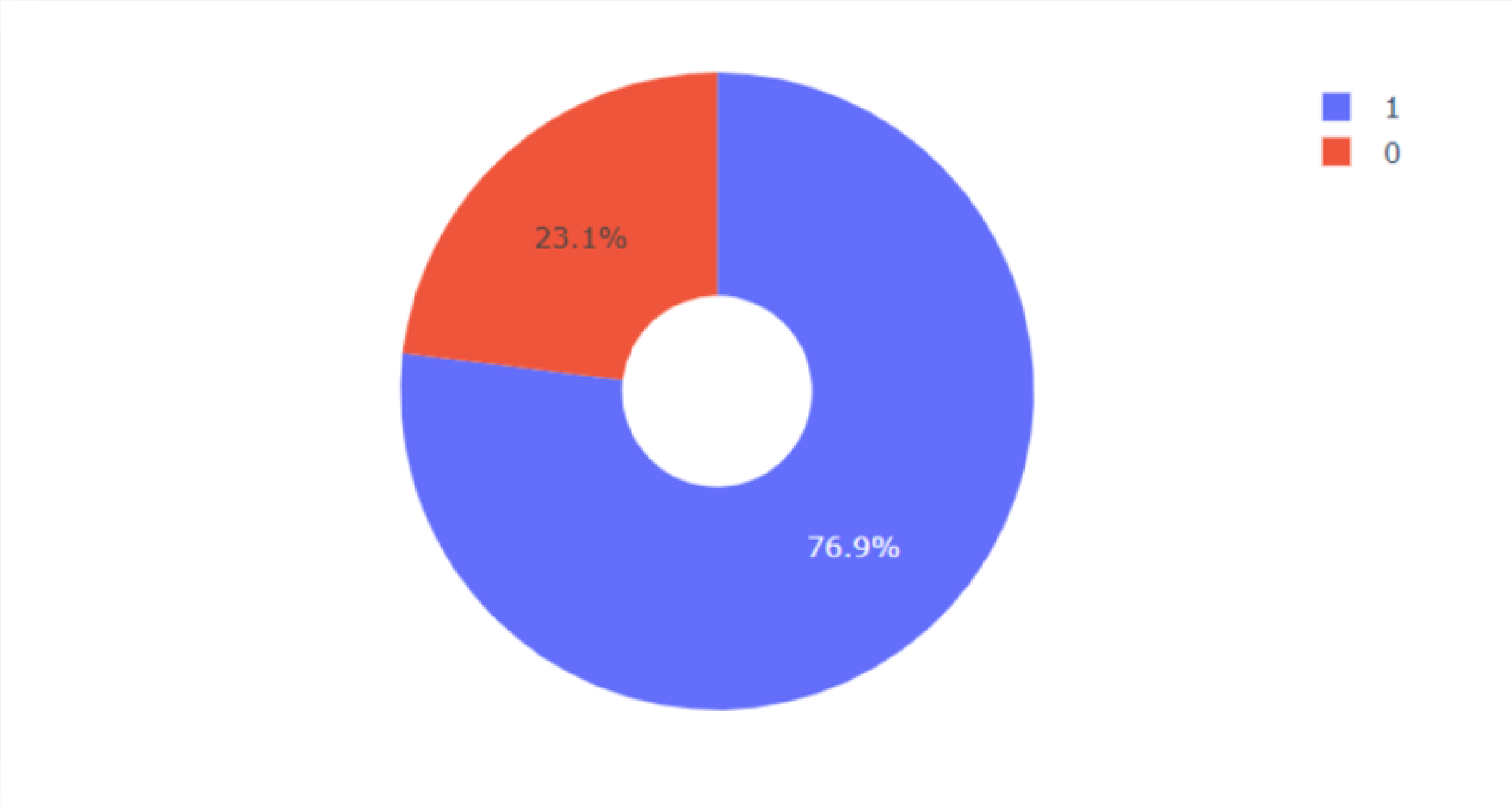


Launch Success for All Sites



From the pie chart, we can identify that **KSC LC-39A** has the highest success rate per launch, whereas, **CCAFS SLC-40** has the lowest success rate per launch.

Highest Success Launch Site

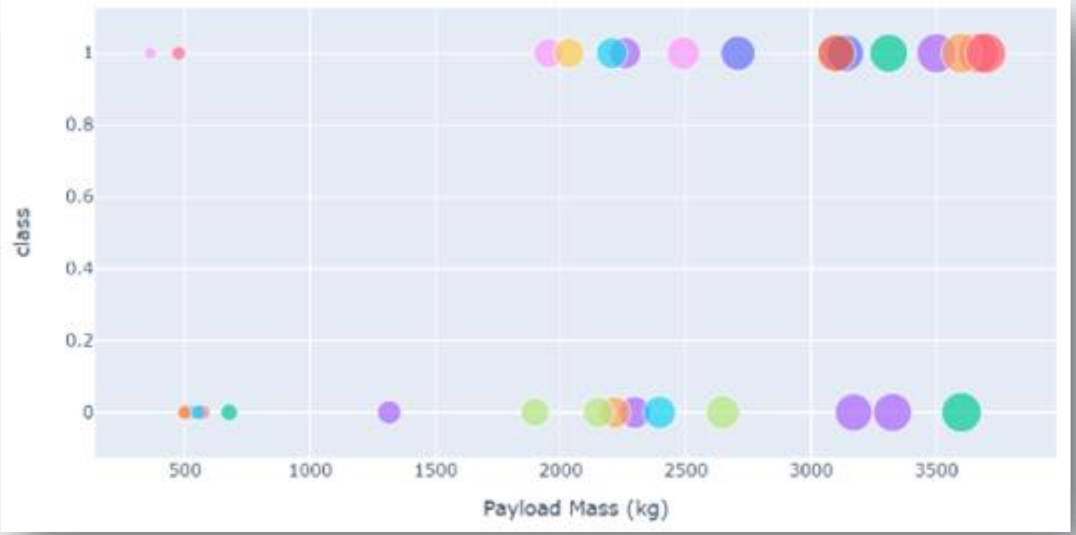


1 = Successful Launch
0 = Failed Launch

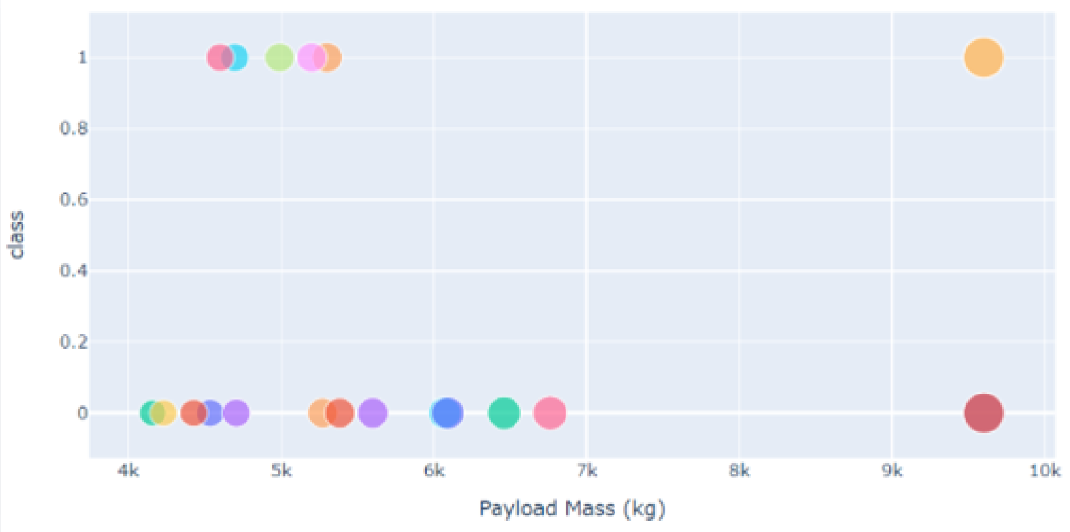
According to the data, **KSC LC-39A** has a **76.9% success rate** and a **23.1% failure rate.**

Payload Mass vs. Launch Outcome

Low Weight Payload Mass (kg) 0kg – 4000kg



High Weight Payload Mass (kg) 4000kg – 10000kg



According to our graphs, the success rates for low-weighted payloads is higher than that of the heavy-weighted payloads.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

Through the use of the machine learning models, we have found that the Decision Tree Classifier is the model with the highest classification accuracy.

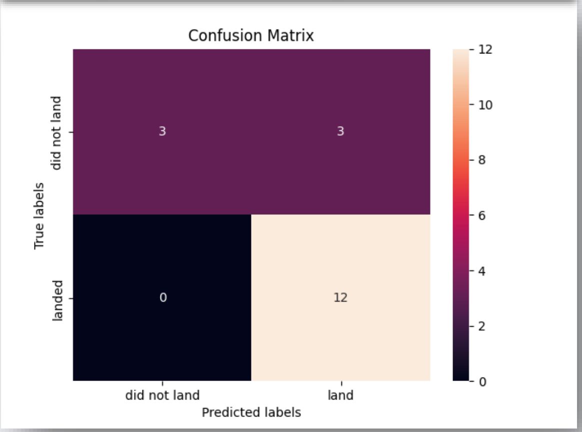
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

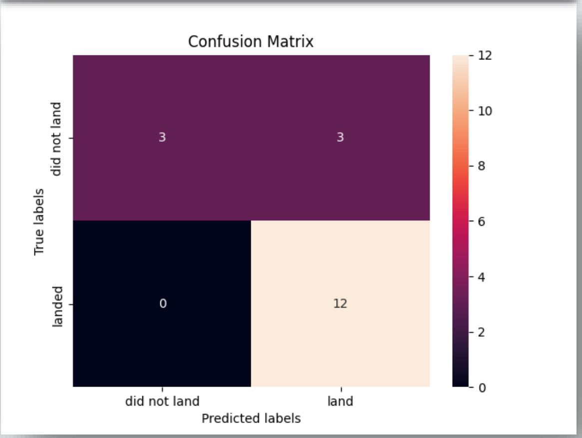
Confusion Matrix

For all of the models, we have the same Confusion Matrix. This could be from using the same limited data for each or all of them having similar parameters during the modeling process so they all produced the same result.

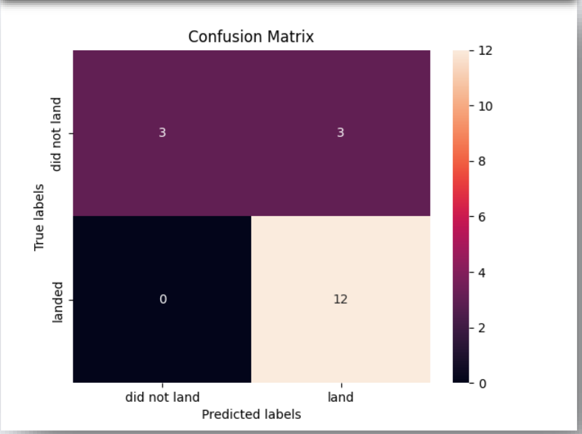
Logistic Regression



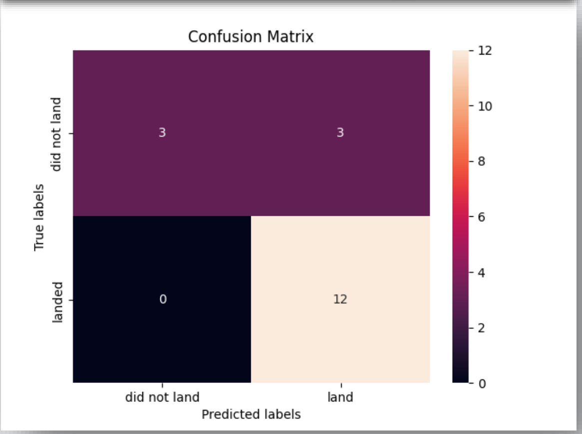
Support Vector Machine



Decision Tree

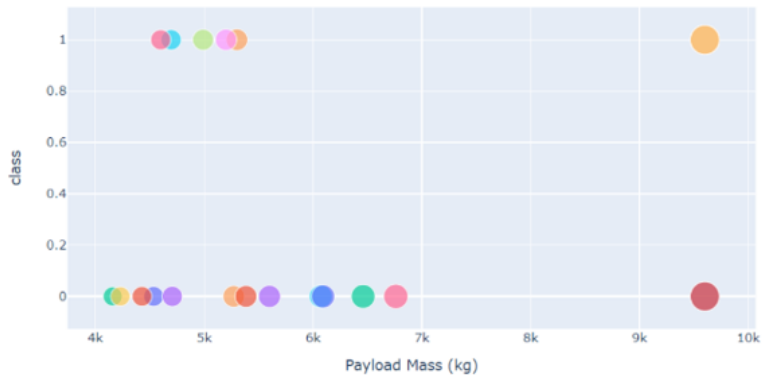


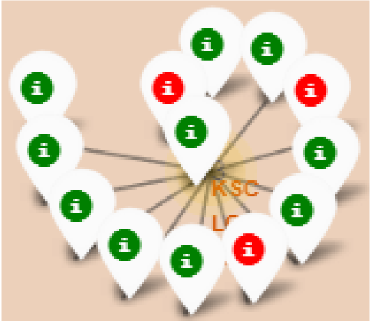
K-Nearest Neighbor

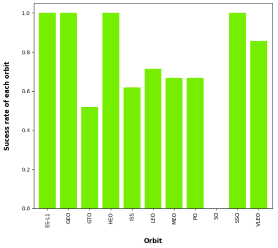


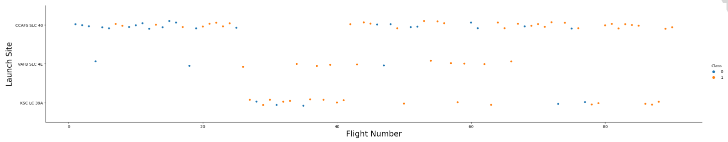
Conclusions

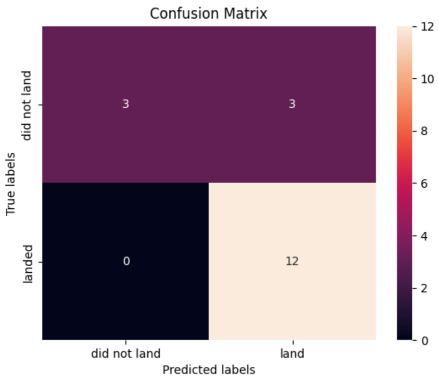
- With higher flight numbers (greater than 30) the success rate for Falcon 9 rocket is increasing.
- ES-L1, GEO, HEO, SSO orbit types have the highest success rates.
- Falcon 9 rocket launches have steadily increased in success rate over time.
- Launch site KSC LC-39A has the most successful launches, but increasing payload mass seems to negatively impact success.
- The Decision Tree Classifier Algorithm provided the highest amount of accuracy.











Appendix

Github SpaceX Project Repository

SpaceX API

SpaceX Wiki Page

Thank you!