IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Timothy Green
9/7/2022

Outline

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

Executive Summary

- Summary of methodologies
  - In this exercise, we will be utilizing web scraping, data wrangling, SQL queries, machine learning models, geographical mapping, and other summary methodologies to describe the attributes related to the Pace X dataset
- Summary of all results
  - In general, the results of the following analysis show that, though there are numerous unsuccessful launches, the number of successful launches has been trending upwards overtime. If this trend continues, the likelihood that the next Space X launch will be successful should continue to rise.

Introduction

- Project background and context
  - This project considers data sets that pertain to the space flight company Space X. The following analysis will examine attributes of Space X launches and provide visualizations for those attributes
- Problems you want to find answers
  - We would like to determine if, with the help of machine learning, we can examine past launch data in order to predict the outcomes of future launches.

Section 1

Methodology

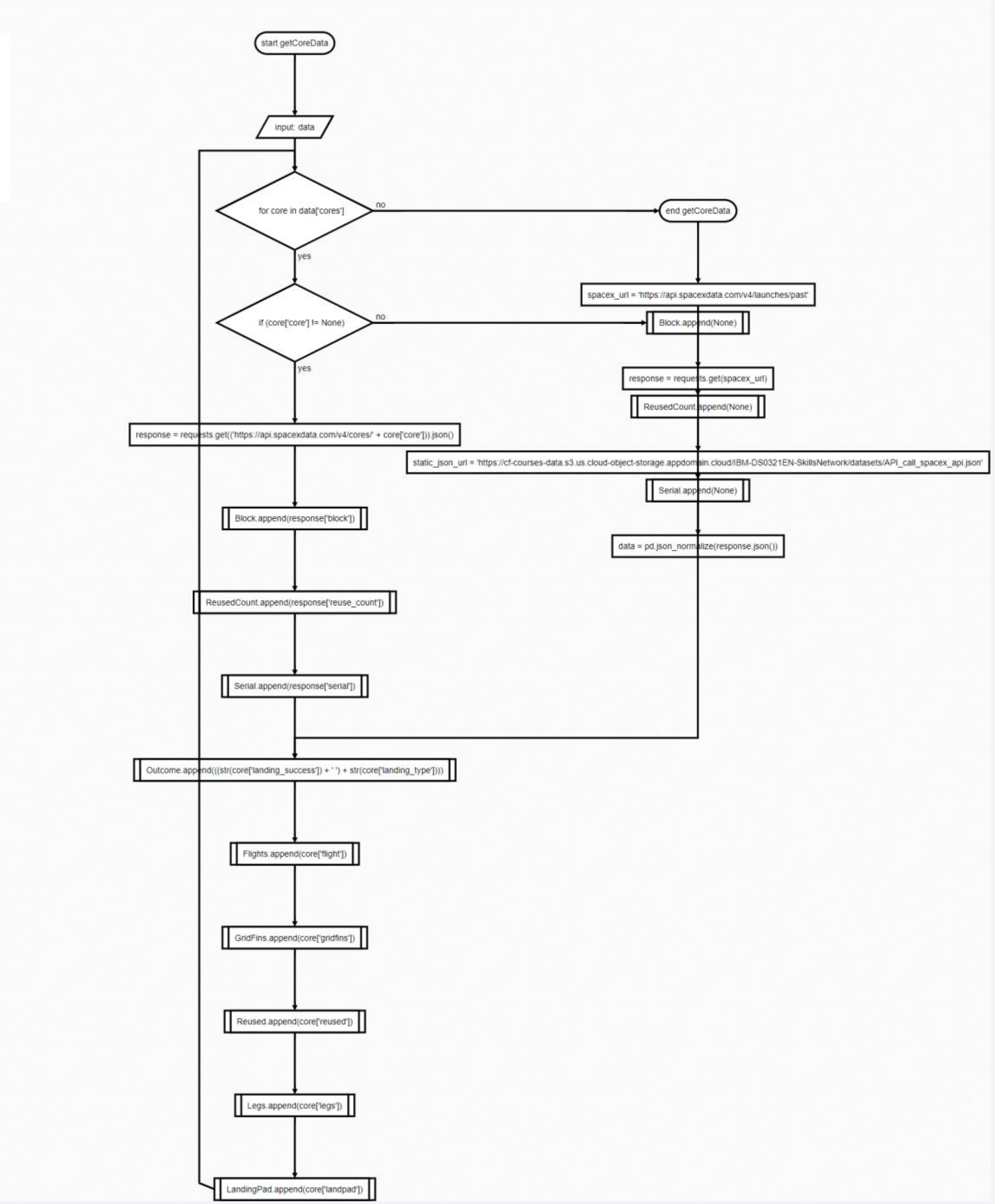
Methodology

Executive Summary

- Data collection methodology:
  - We will be dealing with SpaceX launch data obtained through an API, namely the SpaceX REST API. This will supply us with information on launches such as the rocket version used, payload mass delivered, launch specifications, landing specifications, and landing outcomes. Our goal is to use this information to predict whether or not SpaceX will attempt to land a rocket.
- Perform data wrangling
  - To process the data, we defined the attributes of the result set and calculated some summary values for reference, including a landing outcome label, the number and occurrence of mission outcome per orbit type, the number and occurrence of each orbit, and the number of launches on each site
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models by standardizing the data, splitting the data into training and testing sets, finding the best hyperparameters for each model, and calculating the accuracy of the test data.

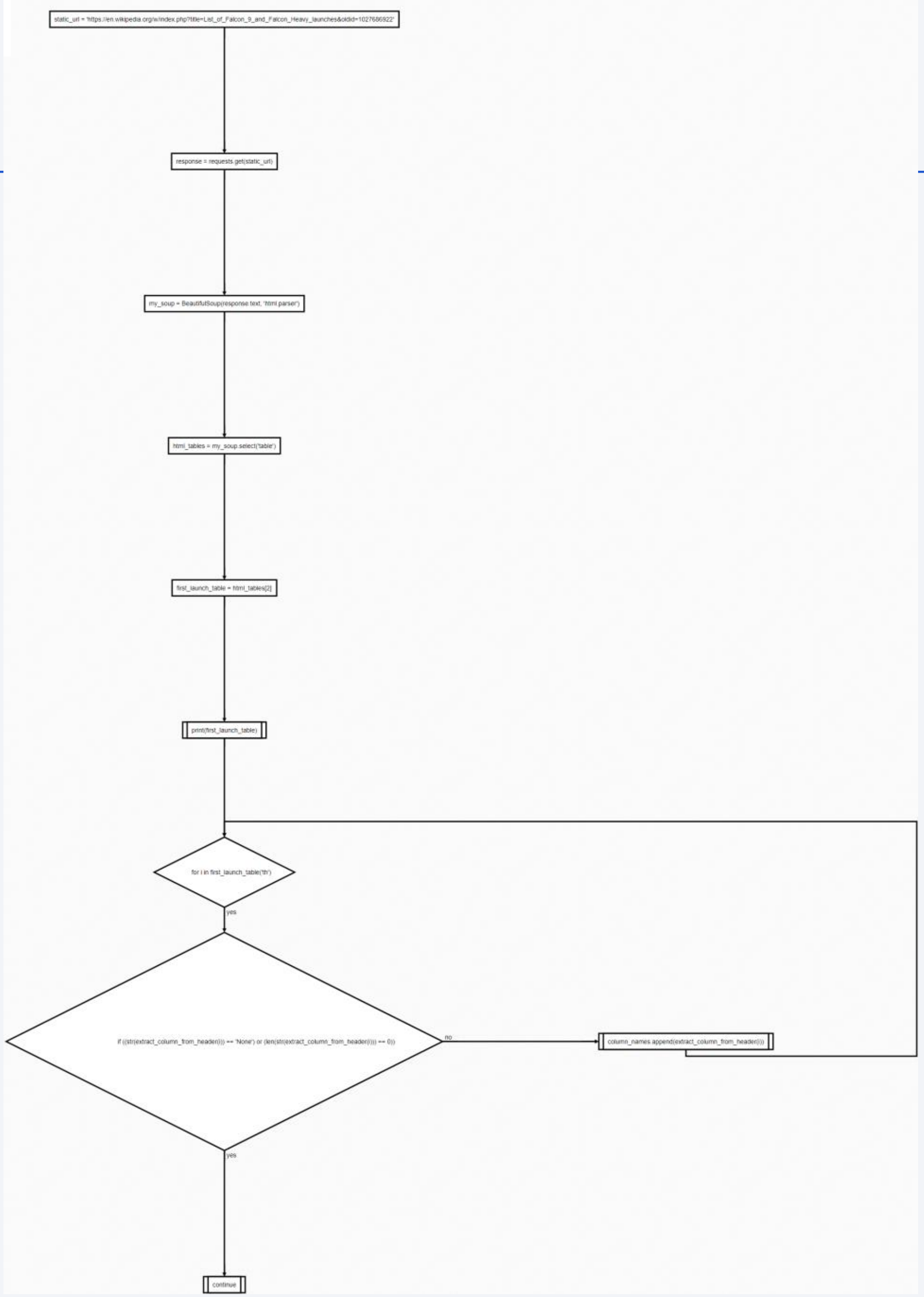
Data Collection – SpaceX API

- [GitHub URL of the completed SpaceX API calls notebook]



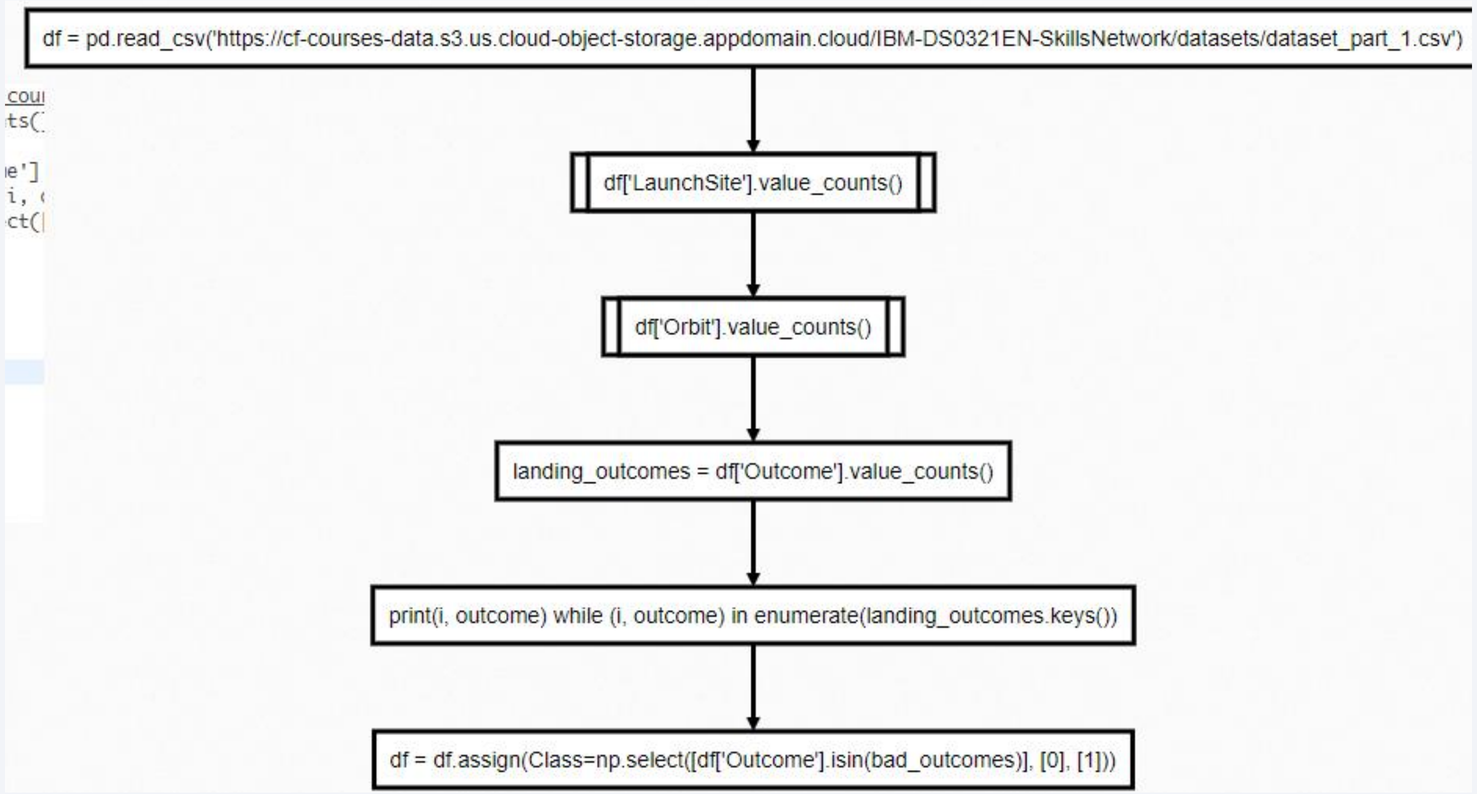
Data Collection - Scraping

- [GitHub URL of the completed web scraping notebook]()



Data Wrangling

- [GitHub URL of your completed data wrangling related notebooks]()



EDA with Data Visualization

- In this workbook, we utilized the scatter plot and bar chart to visualize the relationships between various attributes, including payload size in kilograms, orbit type, yearly trends, flight numbers, and launch sites.
- GitHub URL of your completed EDA with data visualization notebook

EDA with SQL

- General purpose of SQL Queries used:
  - Display the names of the distinct launch sites
  - Display 5 records where the launch site name starts with the letters 'CCA’
  - Display the total payload mass in kilograms carried by boosters launched by the National Aeronautics and Space Administration customer
  - Display average payload mass in kilograms hauled by booster version number F9 v1.1
  - The date of the first successful landing on a ground pad
  - Names of the booster versions with at least one successful landing on a drone ship and which carried a payload mass between 4,000 and 6,000 kilograms
- GitHub URL of your completed EDA with SQL notebook

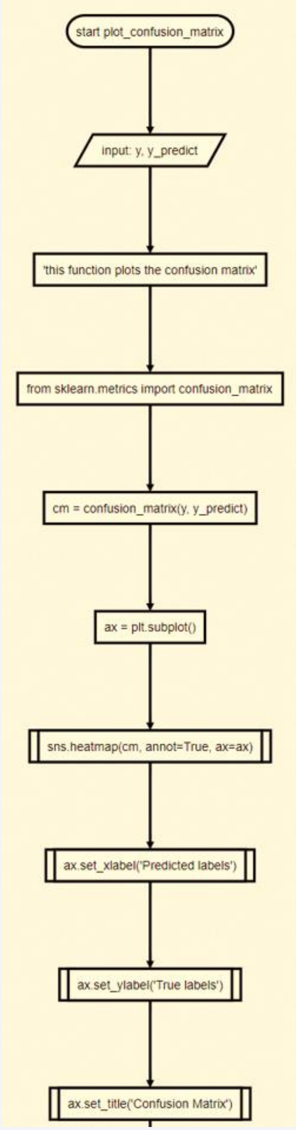
Build an Interactive Map with Folium

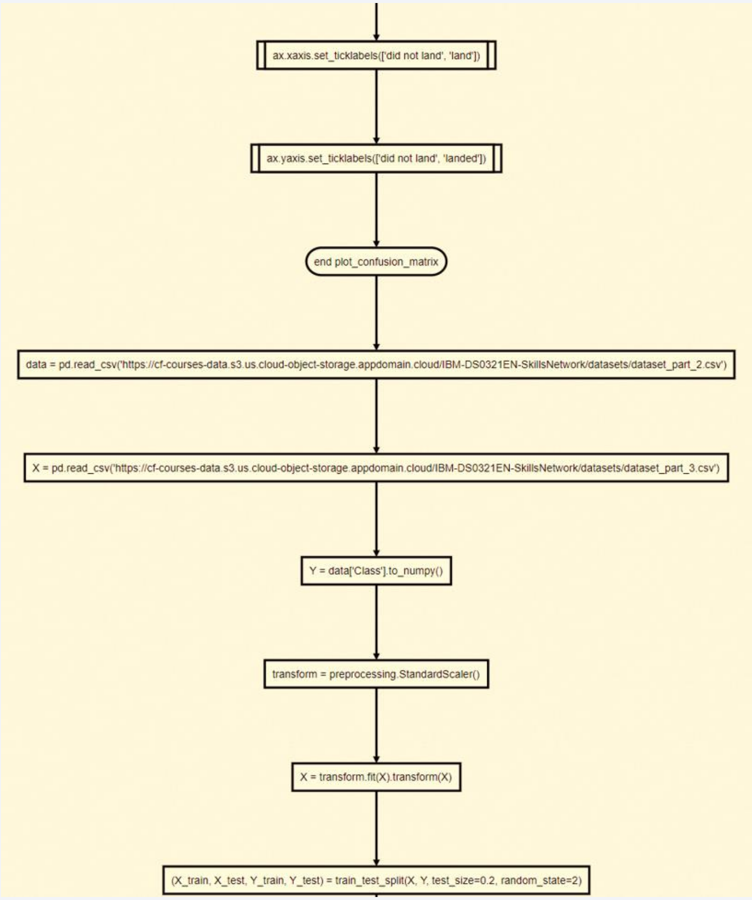
- This workbook contains folium map objects which denote the number and location of launch sites, makers to denote the number of distinct launches with color codes that denote whether they were successful or unsuccessful, and map lines denoting the distance between selected launch sites and geographical features near them
- GitHub URL of your completed interactive map with Folium map

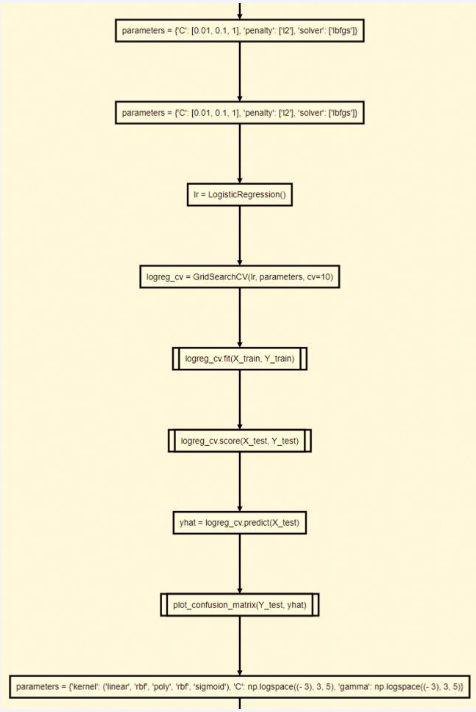
Build a Dashboard with Plotly Dash

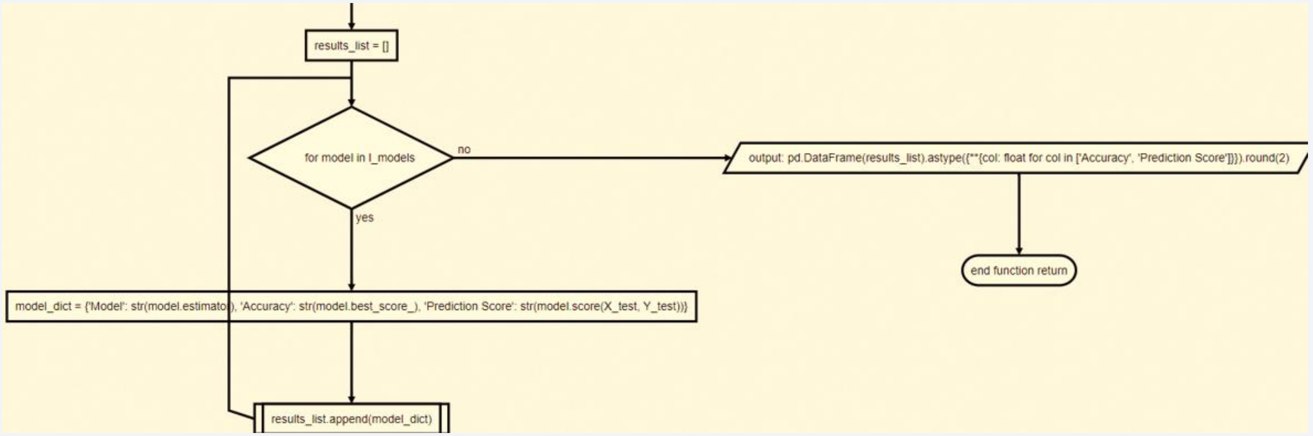
- This Plotly dashboard includes the cod needed to produce a pie chart and scatter chart for use in analyzing the relationships between the number of successful launches from each launch site and the payload mass, flight number, and success rate of different booster versions
- GitHub URL of your completed Plotly Dash lab

Predictive Analysis (Classification)





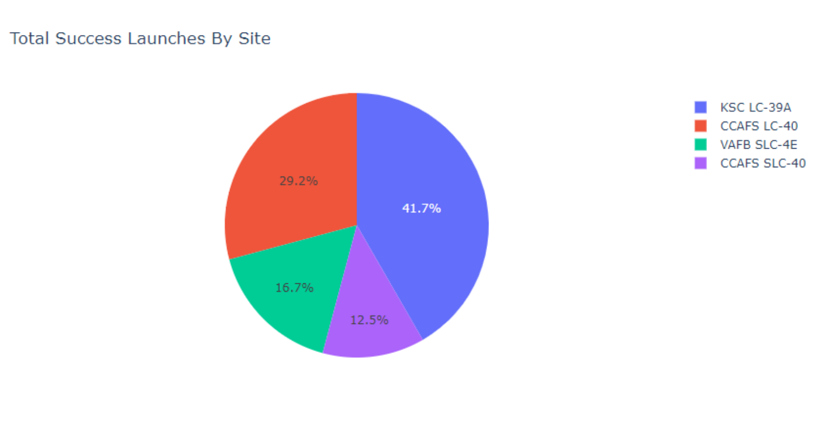




- [GitHub URL of your completed predictive analysis lab]()

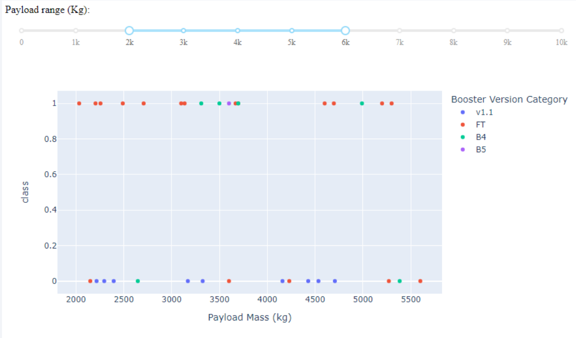
Results

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



Out[45]:

	Model	Accuracy	Prediction Score
0	LogisticRegression(C=1.0, class_weight=None, d...	0.85	0.83
1	SVC(C=1.0, cache_size=200, class_weight=None, ...	0.85	0.83
2	DecisionTreeClassifier(class_weight=None, crit...	0.88	0.78
3	KNeighborsClassifier(algorithm='auto', leaf_si...	0.85	0.83



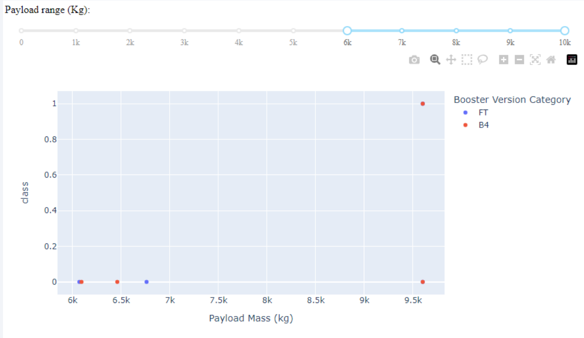
In [83]:

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

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

Out[83]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS_KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt



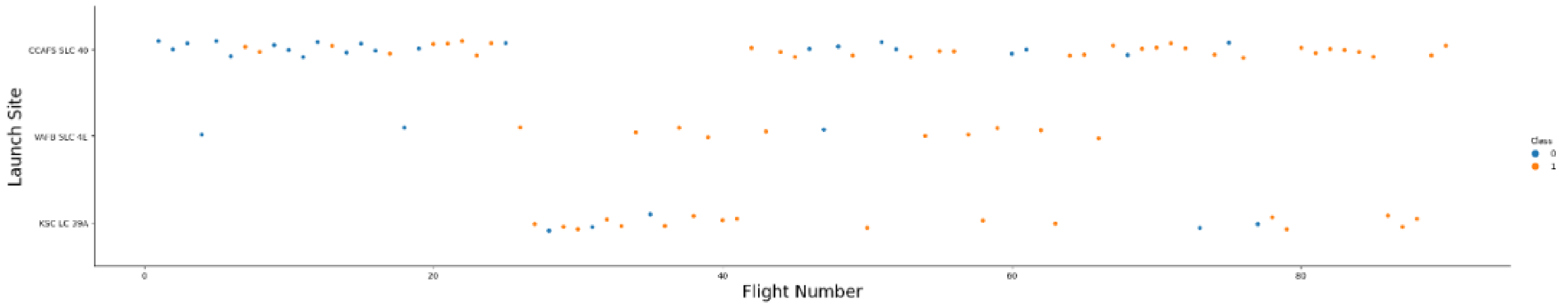
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

In [4]:

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

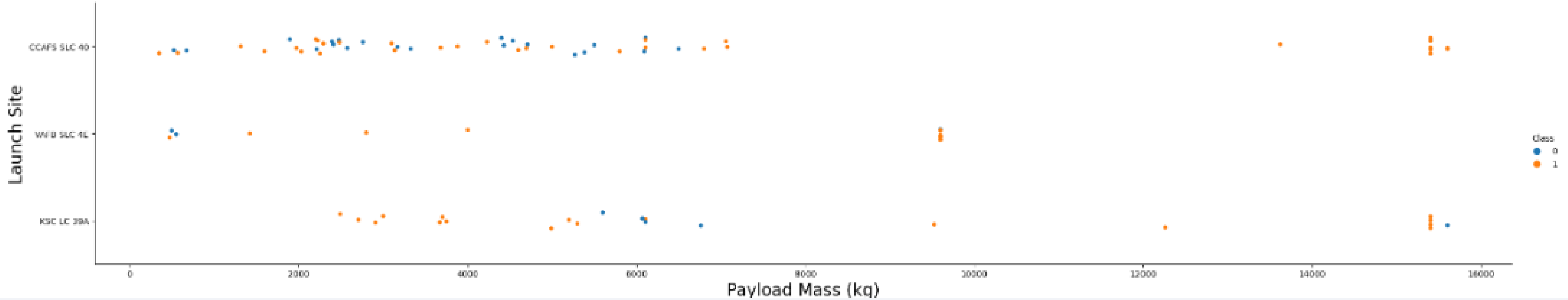


- This scatter plot represents the relationship between Flight Number and Launch Site, where blue markers are failed landing attempts and orange markers are successful landing attempts. It would appear that CCAFS SLC 40 had the most attempts overall, with the most successful and failed attempts as well.

Payload vs. Launch Site

In [5]:

```
# 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("Payload Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



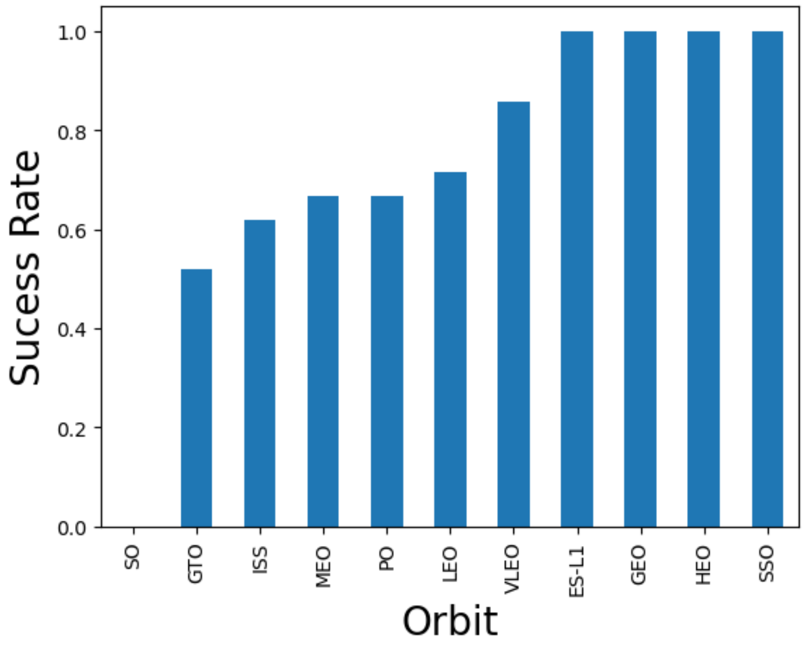
- This scatter plot represents the relationship between Payload Mass in kilograms and Launch Site, where blue markers are failed landing attempts and orange markers are successful landing attempts. It would appear that CCAFS SLC 40 has most of its successes and failures at between 0 and 9000 kg.

Success Rate vs. Orbit Type

- This bar chart represents the Success Rate for each type of Orbit. We can see that ES-L1, GEO, HEO, and SSO orbits are tied at a 100% success rate.

In [10]:

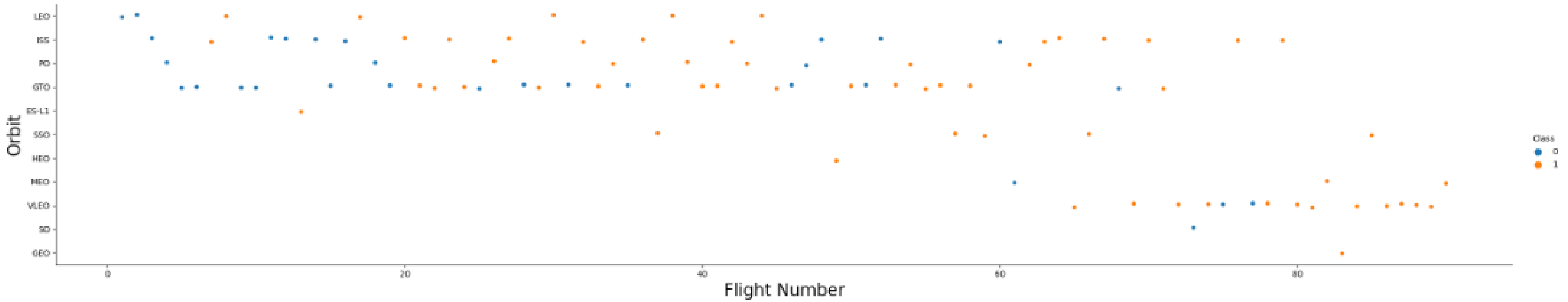
```
# HINT use groupby method on Orbit column and get the mean of Class column
(
    df
        .groupby(['Orbit'])
        .mean()['Class']
        .sort_values()
        .plot(kind='bar')
)

plt.xlabel("Orbit", fontsize=20)
plt.ylabel("Sucess Rate", fontsize=20)
plt.show()
```



Flight Number vs. Orbit Type

In [14]:

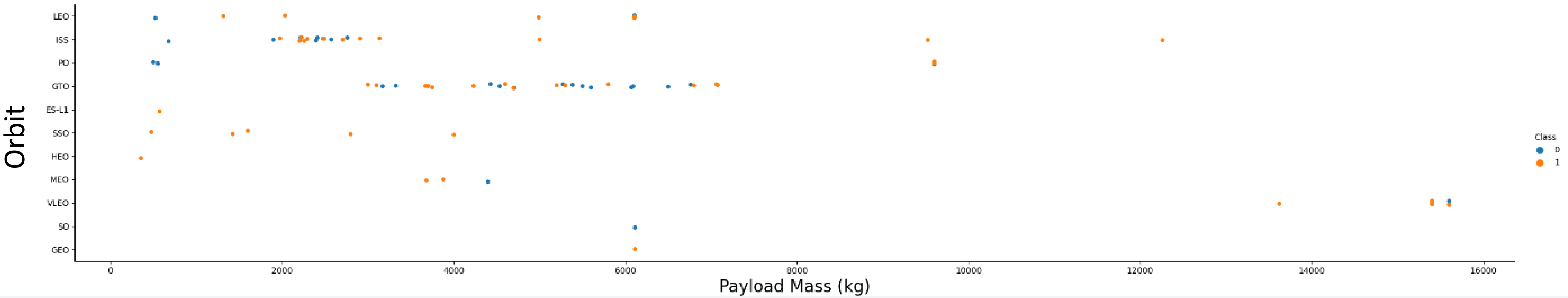
```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue='Class', data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



- This scatter plot show the relationship between flight number and orbit type name. We can see that many of the launches with higher flight number belong to orbit VLEO.

Payload vs. Orbit Type

```
[9]: # Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue='Class', data=df, aspect = 5)
plt.xlabel("Payload Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



- This scatter plot shows the payload mass in kilograms in relation to the orbit name. We can see that GTO has the most launches overall.

Launch Success Yearly Trend

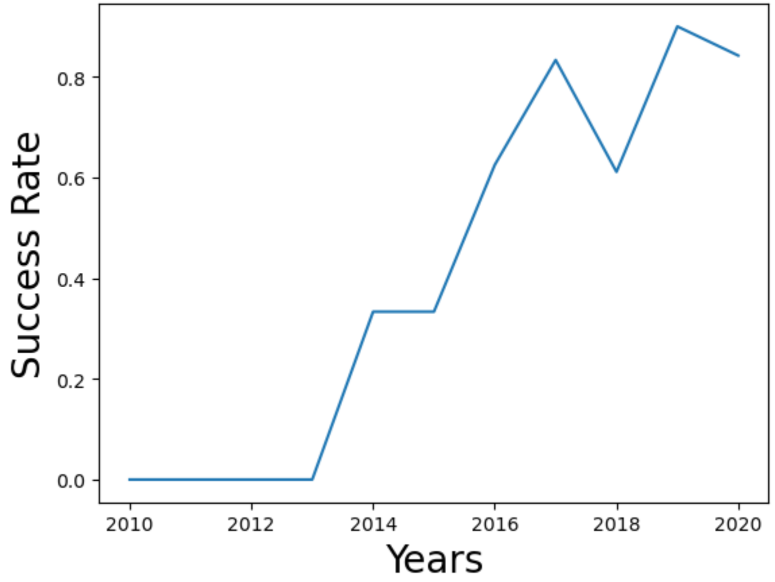
- This line chart represents the total success rate over the span of years from 2010 to 2020. We can see that the success rate has been trending upwards over time.

In [16]:

```python
# A function to Extract years from the date
year=[]
def Extract_year(date):
    for i in df["Date"]:
        year.append(i.split("-")[0])
    return year
```

In [18]:

```python
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
df['Year'] = pd.DataFrame(Extract_year(df['Date'])).astype('int')

x_val = df['Year'].unique()
y_val = df.groupby(['Year'])['Class'].mean()

sns.lineplot(x = x_val , y = y_val)
plt.xlabel("Years",fontsize=20)
plt.ylabel("Success Rate",fontsize=20)
plt.show()
```



All Launch Site Names

- This SQL query results in a list of the distinct launch site names.

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

```
In [7]: %sql SELECT DISTINCT Launch_Site FROM SPACEXTBL

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

Out[7]:

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

Launch Site Names Begin with 'CCA'

- This SQL query results in a list of five records where the launch site name starts with the letters CCA.

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

In [83]:
```
%sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5
```

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

Out[83]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- This SQL query represents the total payload mass in kilograms that have been carried by the National Space and Aeronautics Administration customer.

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

```
In [9]: %sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD_MASS FROM SPACEXTBL WHERE Customer LIKE 'NASA (CRS)%'
```

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

Out[9]:

TOTAL_PAYLOAD_MASS
48213

Average Payload Mass by F9 v1.1

- This SQL query results in the average payload mass in kilograms carried by the F9 v1.1 booster version.

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

In [10]:
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD_MASS, Booster_Version FROM SPACEXTBL WHERE Booster_Version LIKE 'F9 v1.1%'
```

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

Out[10]:

AVG_PAYLOAD_MASS	Booster_Version
2534.6666666666665	F9 v1.1 B1003

First Successful Ground Landing Date

- This SQL query results in the date in which the first successful ground pad landing occurred.

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

In [11]:
```
%sql SELECT MIN(Date), `Landing _Outcome` FROM SPACEXTBL WHERE `Landing _Outcome` LIKE '%Success (ground pad)%'
```

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

Out[11]:

MIN(Date)	Landing _Outcome
01-05-2017	Success (ground pad)

Successful Drone Ship Landing with Payload between 4000 and 6000

- This SQL query results in a list of booster version numbers that have had at least one successful landing on a drone ship while carrying a payload mass in kilograms of between 4000 and 6000 kg.

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [12]:

```
%%sql booster_names <<

SELECT
    Booster_Version,
    Payload,
    `Landing _Outcome`,
    PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE
    `Landing _Outcome` LIKE '%Success (drone ship)%'
    AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

```
 * sqlite:///my_data1.db
Done.
Returning data to local variable booster_names
```

In [13]:

```
booster_names
```

Out[13]:

Booster_Version	Payload	Landing _Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	JCSAT-14	Success (drone ship)	4696
F9 FT B1026	JCSAT-16	Success (drone ship)	4600
F9 FT B1021.2	SES-10	Success (drone ship)	5300
F9 FT B1031.2	SES-11 / EchoStar 105	Success (drone ship)	5200

Total Number of Successful and Failure Mission Outcomes

- This SQL query results in a count of the total number of successful and failure missions.

List the total number of successful and failure mission outcomes

In [14]:
```
%%sql mission_outcomes <<

SELECT DISTINCT
    (SELECT COUNT(Mission_Outcome) FROM SPACEXTBL WHERE Mission_Outcome LIKE '%Success%') AS SUCCESSFUL_MISSIONS,
    (SELECT COUNT(Mission_Outcome) FROM SPACEXTBL WHERE Mission_Outcome LIKE '%Failure%') AS FAILURE_MISSIONS
FROM SPACEXTBL
```

```
 * sqlite:///my_data1.db
Done.
Returning data to local variable mission_outcomes
```

In [15]:
```
mission_outcomes
```

Out[15]:

SUCCESSFUL_MISSIONS	FAILURE_MISSIONS
100	1

Boosters Carried Maximum Payload

- This SQL query lists the distinct booster version numbers that have carried the max payload mass in kilograms of all in the set.

```
In [16]:
%%sql max_mass <<

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

 * sqlite:///my_data1.db
Done.
Returning data to local variable max_mass

In [17]:
max_mass

Out[17]:
```

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

2015 Launch Records

- This SQL query utilizes a common table expression to derive the month name and year of failed drone ship landing in 2015

In [18]:

```
%%sql sql_query <<

WITH
    MONTH_TABLE(MONTH_NO, MONTH_NAME) AS (
        VALUES
            /* Silicon Ranch */
            ('01', 'January'),
            ('02', 'February'),
            ('03', 'March'),
            ('04', 'April'),
            ('05', 'May'),
            ('06', 'June'),
            ('07', 'July'),
            ('08', 'August'),
            ('09', 'September'),
            ('10', 'October'),
            ('11', 'November'),
            ('12', 'December')
        )
SELECT
    M.MONTH_NAME,
    SUBSTR(X.Date,7,4) AS YEAR,
    X.DATE,
    X.Booster_Version,
    X.Launch_Site,
    X.`Landing _Outcome`
FROM SPACEXTBL AS X
JOIN MONTH_TABLE AS M
    ON SUBSTR(X.Date, 4, 2) = M.MONTH_NO
WHERE
    `Landing _Outcome` LIKE '%Failure (drone ship)%'
    AND SUBSTR(X.Date,7,4) = '2015'
```

```
 * sqlite:///my_data1.db
Done.
Returning data to local variable sql_query
```

In [19]:

```
sql_query
```

Out[19]:

MONTH_NAME	YEAR	Date	Booster_Version	Launch_Site	Landing _Outcome
January	2015	10-01-2015	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
April	2015	14-04-2015	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- This SQL query details the ranked count of each outcome category, with the results sorted in descending order.

In [81]:

```
%%sql landing_ranks <<

SELECT
    `Landing _Outcome`,
    COUNT(`Landing _Outcome`),
    RANK() OVER(
        ORDER BY `Landing _Outcome`
    ) AS R1
FROM SPACEXTBL
WHERE
    `Landing _Outcome` LIKE '%Success%'
    AND Date BETWEEN '04-06-2010' AND '20-03-2017'
GROUP BY `Landing _Outcome`
ORDER BY RANK() OVER(
        ORDER BY `Landing _Outcome` ASC
    ) DESC
```

```
 * sqlite:///my_data1.db
Done.
Returning data to local variable landing_ranks
```

In [82]:

```
landing_ranks
```

Out[82]:

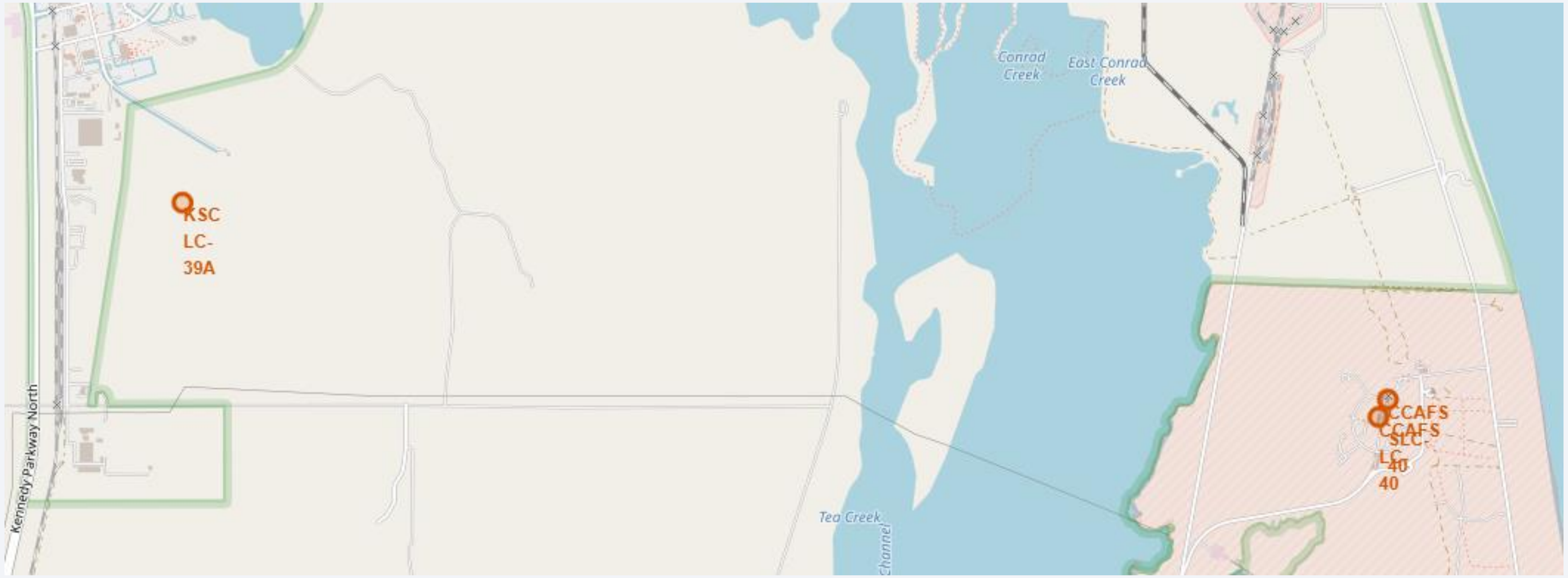
Landing _Outcome	COUNT(`Landing _Outcome`)	R1
Success (ground pad)	6	3
Success (drone ship)	8	2
Success	20	1

Section 3

Launch Sites Proximities Analysis

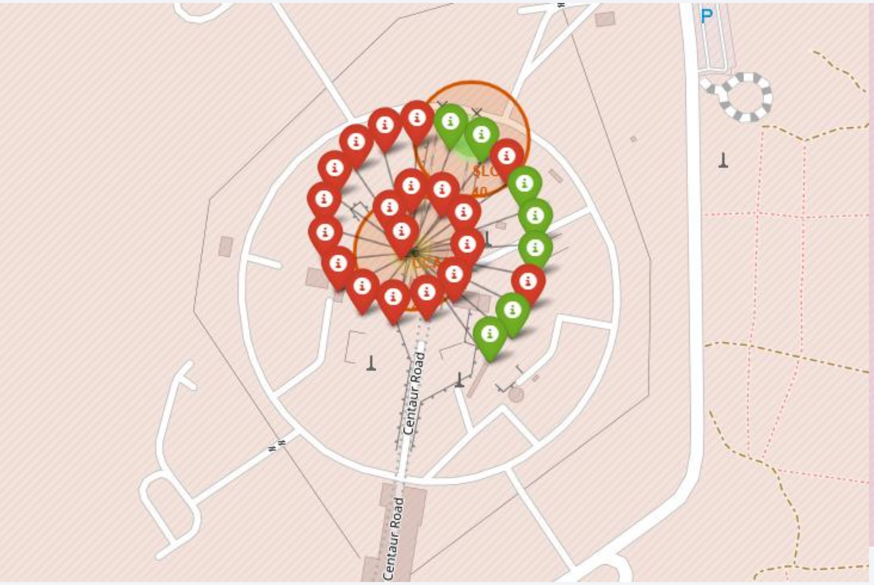
Launch Sites Location Markers

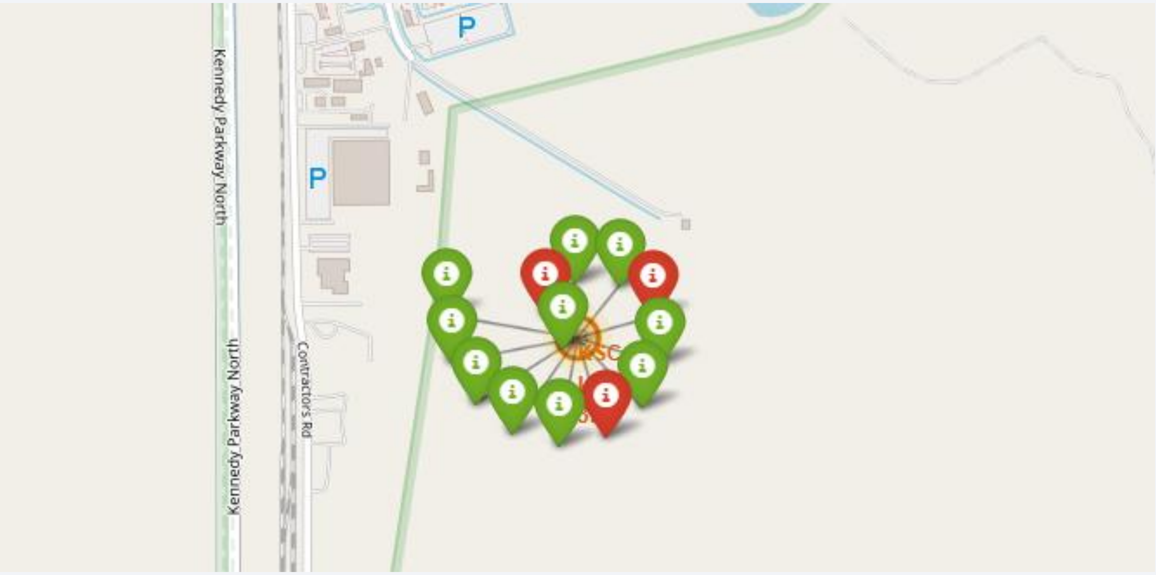
- These figures show an orange marker for each launch site cluster, all of which are contained in the continental USA.

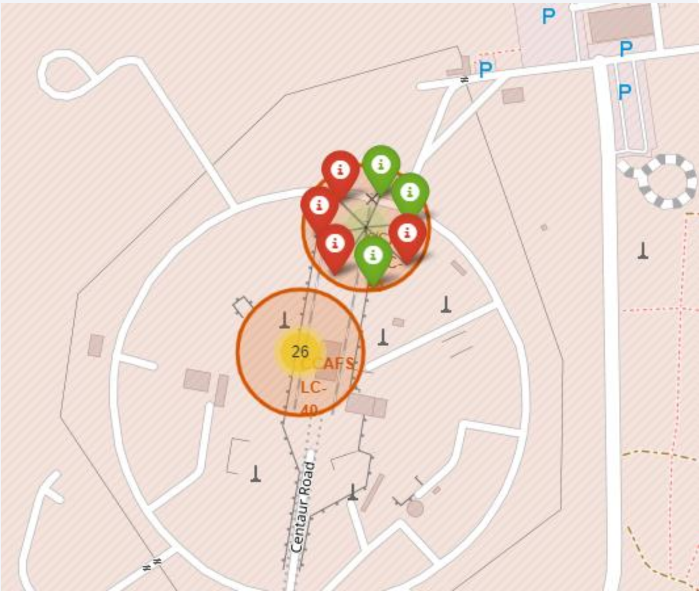


Color-labeled Launch Outcomes

- These figures show colored markers for launch types at each launch site, green being successful launches and read being failed launches.

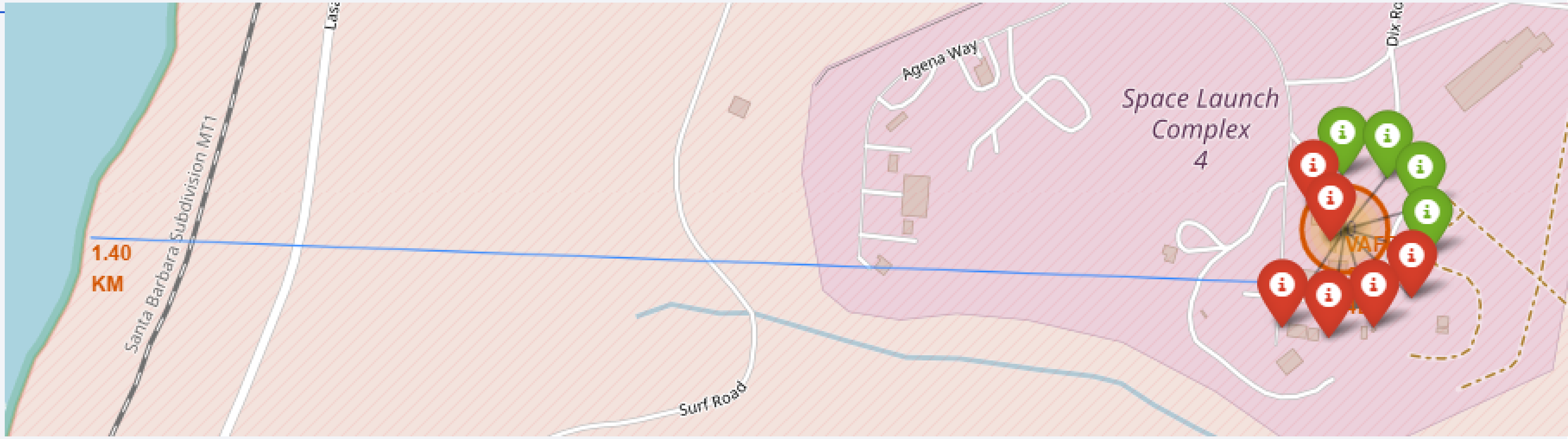






Selected Launch Site To Its Proximities

- These map figures represent the distance from a launch site cluster to the nearest ocean front and to the nearest creek.

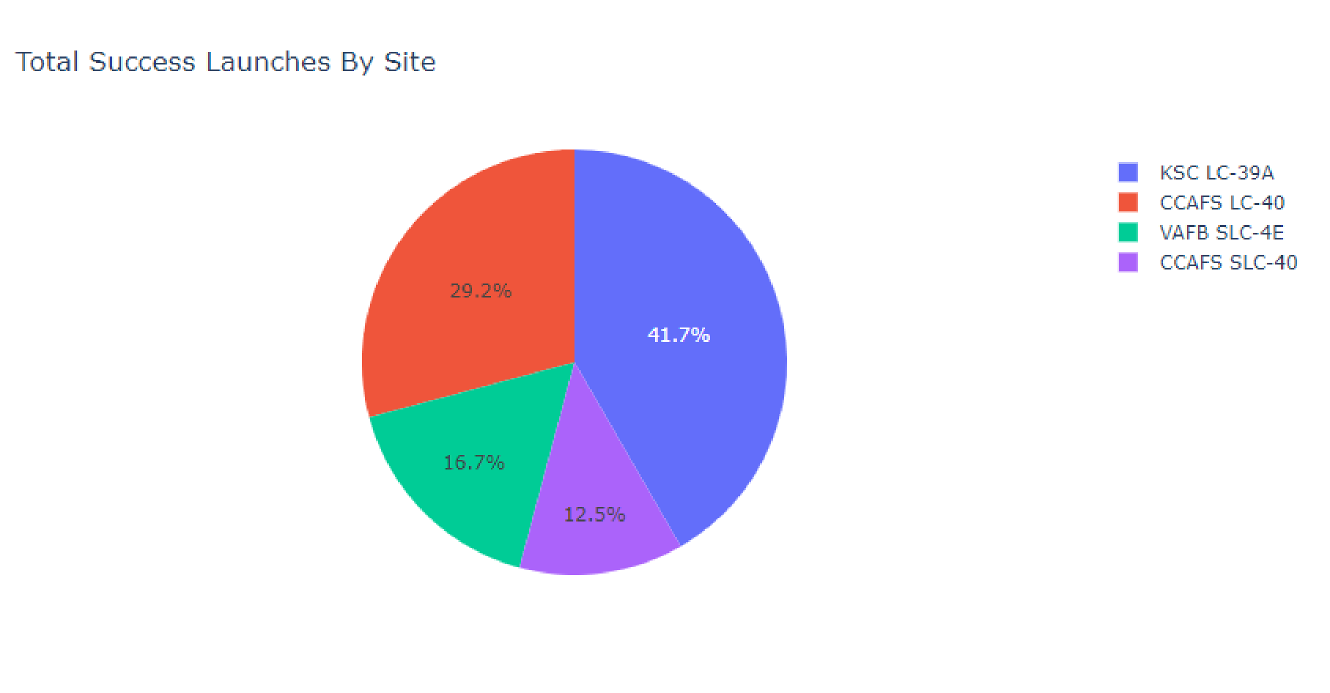


Section 4

Build a Dashboard with Plotly Dash

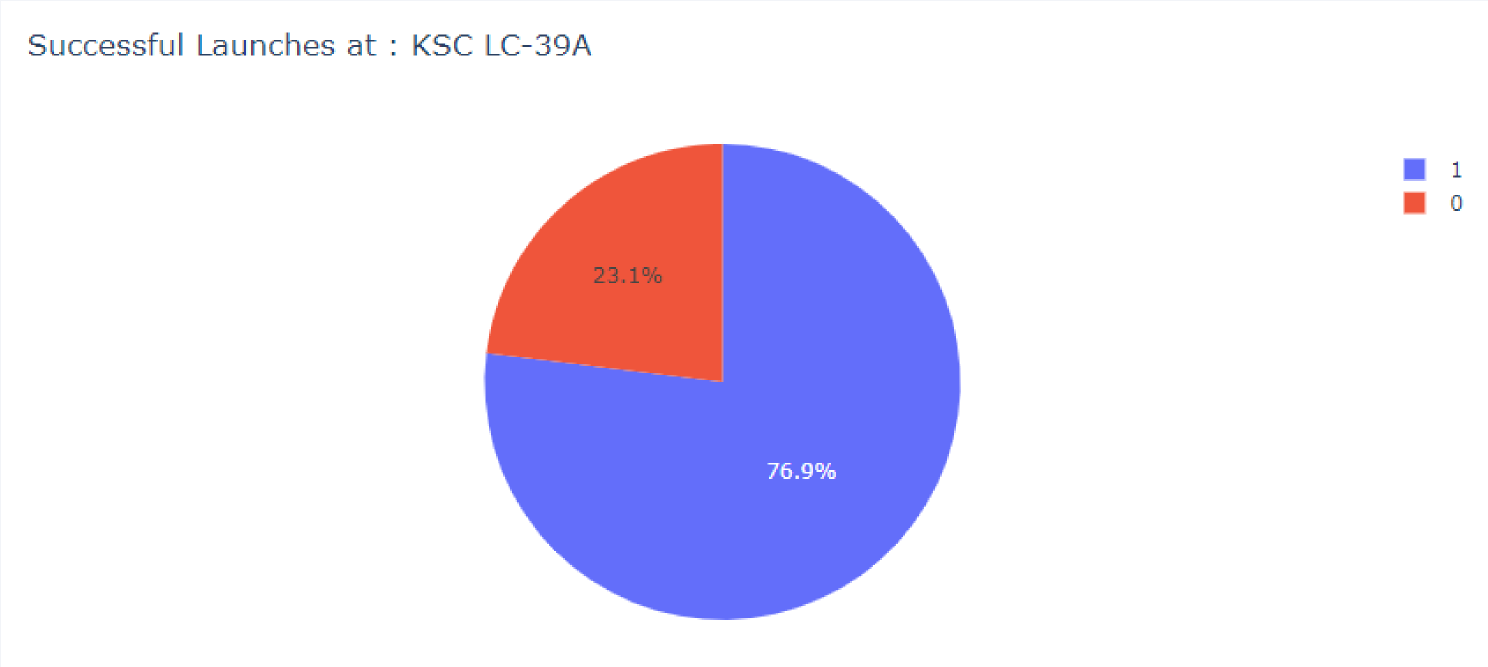
Pie Chart Launch Success Count For All Sites

- This figure represents the ratio of successful launches at each launch site in relation to every other site with successful launches.



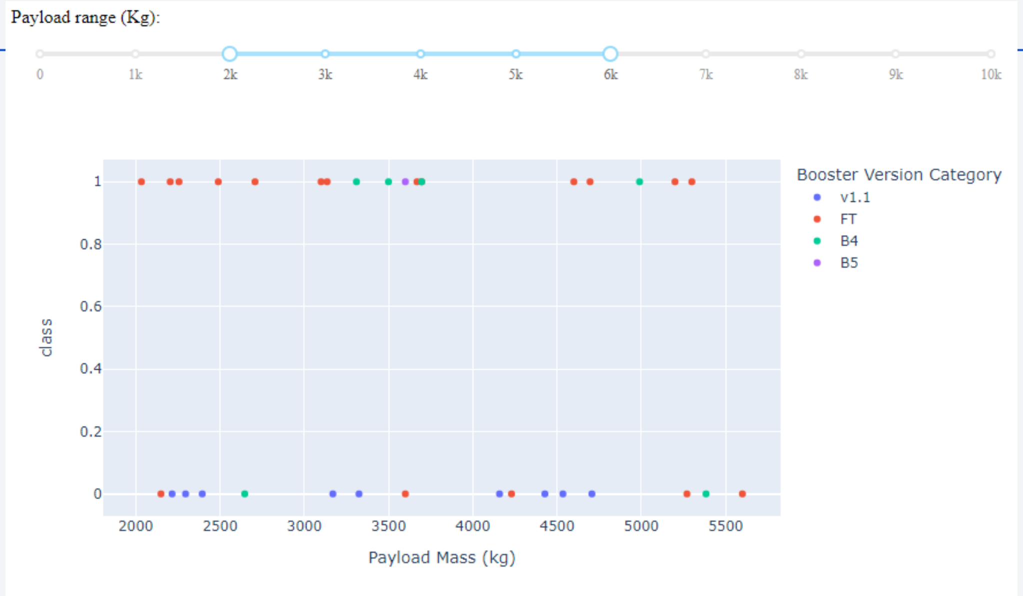
Pie Chart Launch Site With Highest Launch Success Ratio

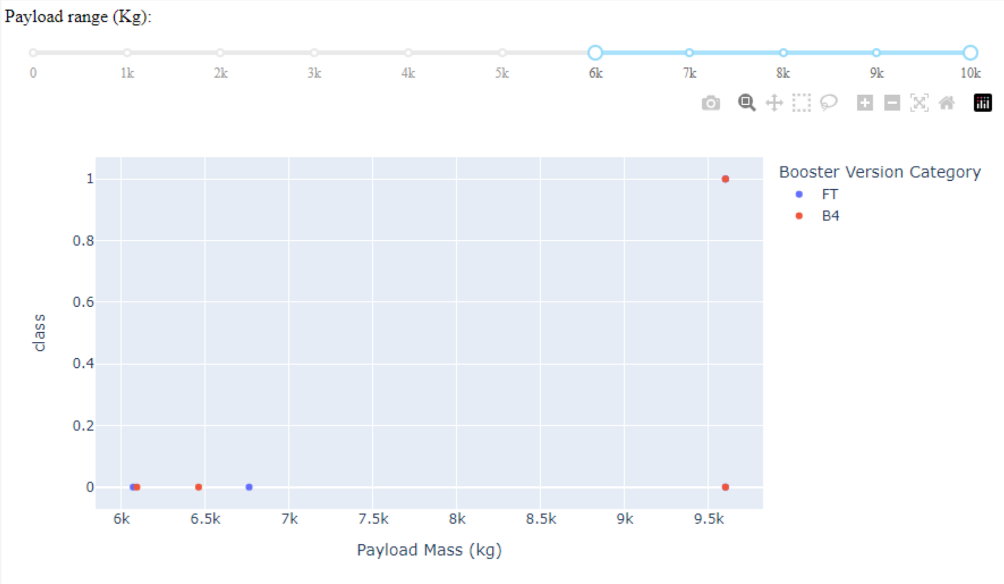
- This pie chart represents the ratio of failed launches to successful launches at the launch site with the highest success rate of all sites.



Payload vs. Launch Outcome

- These figures represent the success classes of different booster version categories in relation to their payload mass in kilograms, with 1 being a successful landing and 0 being a failed landing.





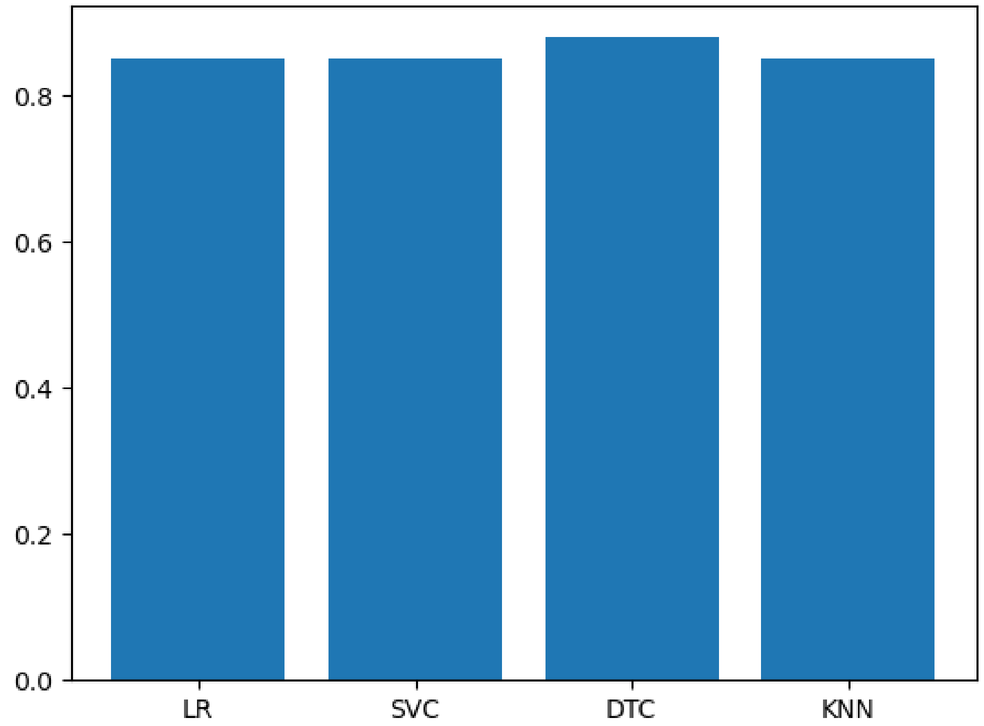
Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The predictive model with the highest accuracy as the Decision Tree classifier, with an accuracy of 88%.

[37]: <BarContainer object of 4 artists>

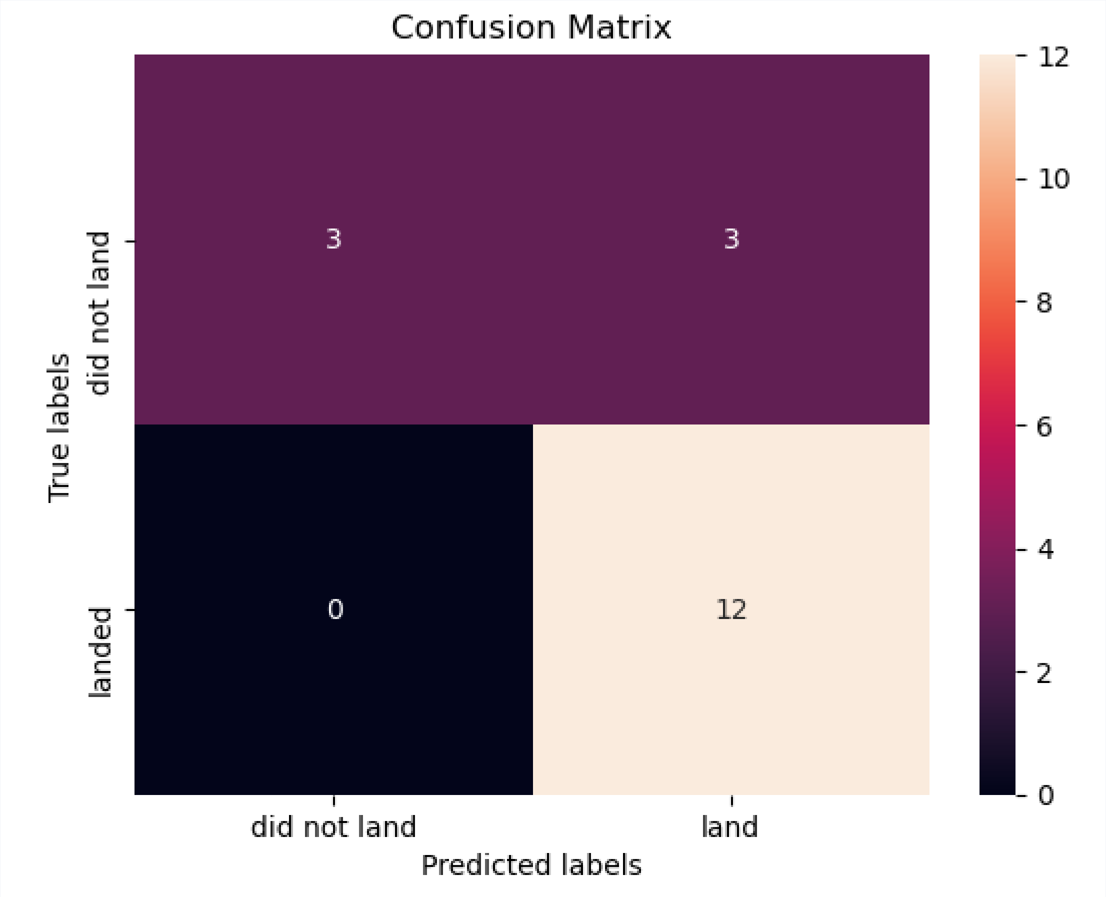


[30]:

	Model	Accuracy	Prediction Score
0	LogisticRegression(C=1.0, class_weight=None, d...	0.85	0.83
1	SVC(C=1.0, cache_size=200, class_weight=None, ...	0.85	0.83
2	DecisionTreeClassifier(class_weight=None, crit...	0.88	0.72
3	KNeighborsClassifier(algorithm='auto', leaf_si...	0.85	0.83

Confusion Matrix

- This confusion matrix represents the predictive model with the highest accuracy tested, which was the decision tree classifier at 88% accuracy.



Conclusions

- The data sets provided are of a sufficient quality as to server as the basis for this exercise
- The number of successful launches is increasing over time and is likely to continue trending upwards.
- It is likely that, with machine learning models, we can estimate the chances of a successful future launch with increasing accuracy as time progresses

Appendix

- app_data_science_capstone_coursera/Data Collection API Lab.ipynb at main · delgreen/app_data_science_capstone_coursera (github.com)
- app_data_science_capstone_coursera/SpaceX_Machine Learning Prediction_Part_5.ipynb at main · delgreen/app_data_science_capstone_coursera (github.com)
- app_data_science_capstone_coursera/jupyter-labs-eda-dataviz.ipynb at main · delgreen/app_data_science_capstone_coursera (github.com)
- app_data_science_capstone_coursera/jupyter-labs-eda-sql-coursera_sqllite.ipynb at main · delgreen/app_data_science_capstone_coursera (github.com)
- app_data_science_capstone_coursera/jupyter-labs-webscraping.ipynb at main · delgreen/app_data_science_capstone_coursera (github.com)
- app_data_science_capstone_coursera/lab_jupyter_launch_site_location.ipynb at main · delgreen/app_data_science_capstone_coursera (github.com)
- app_data_science_capstone_coursera/labs-jupyter-spacex-Data wrangling.ipynb at main · delgreen/app_data_science_capstone_coursera (github.com)
- app_data_science_capstone_coursera/spacex_dash_app.py at main · delgreen/app_data_science_capstone_coursera (github.com)

Thank you!