IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Dominique Dayato
December 23rd, 2023

Outline

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

Executive Summary

- Summary of methodologies:
    - Data Collection with SpaceX API
    - Data Collection with Web Scrapping
    - Data Wrangling
    - Exploratory Data Analysis with SQL
    - Exploratory Data Analysis with Visualization
    - Interactive Visual Analytics with Folium
    - Interactive Dashboard with Plotly
    - Machine Learning Prediction
- Summary of all results:
    - SQL Table Results
    - Graphs (Scatter Plots, Line plot, Bar Chart)
    - Maps
    - Dashboards
    - Categorical Prediction

Introduction

- Project background and context:
  - Welcome to the commercial space age, where companies are making space travel affordable for everyone. Today we are going to focus on SpaceX for our studies. Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.
- Problems you want to find answers
  - If we can determine if the first stage will land, we can determine the cost of a launch.
  - 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.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- 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

Data Collection

- How was Data collected?
  - We worked with SpaceX launch data that was gathered from the SpaceX REST API. This API gives us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
  - Another way we obtained Falcon 9 Launch data is web scraping related Wiki pages. We used the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records. Then we parsed the data from those tables and converted them into a Pandas data frame for further visualization and analysis.

Data Collection – SpaceX API

- We will perform a get request using the requests library to obtain the launch data, which we will use to get the data from the API. This result can be viewed by calling the .json() method. Our response will be in the form of a JSON, specifically a list of JSON objects. To convert this JSON to a data frame, we can use the json_normalize function. This function will allow us to “normalize” the structured json data into a flat table.
- Here is the GitHub URL of the completed SpaceX API calls notebook: https://github.com/dominiquedayato/Capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb.

To make the requested JSON results more consistent, we will use the following static response object for this project:

[9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwor
```

We should see that the request was successfull with the 200 status response code

[10]:
```
response.status_code
```

[10]: 200

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

[11]:
```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

[12]:
```
# Get the head of the dataframe
data.head()
```

[12]:

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine failure'}]	Engine failure at 33 seconds and loss of vehicle
								Successful first stage burn and transition

Data Collection with SpaceX Launch REST API

Data Collection - Scraping

- First, we performed an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.
- We created a BeautifulSoup object from the HTML response.
- Next, we collected all relevant column names from the HTML table header.
- We created an empty dictionary with keys from the extracted column names in the previous task, then converted this dictionary into a Pandas data frame.
- Here is the GitHub URL of the completed web scraping notebook: https://github.com/dominiquedayato/Capstone/blob/main/jupyter-labs-webscraping.ipynb

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

In [9]:
```
# use requests.get() method with the provided static_url
# assign the response to a object

data  = requests.get(static_url).text
```

Create a `BeautifulSoup` object from the HTML `response`

In [12]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data, 'html5lib')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

In [13]:
```
# Use soup.title attribute
soup.title
```

Out[13]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

In [17]:
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

Data Collection With Web Scrapping

Data Wrangling

- Describe how data were processed:
    - In the data set, there are several different cases where the booster did not land successfully. We mainly converted those different outcomes into Training Labels with 1 meaning the booster successfully landed and 0 meaning it was unsuccessful.
    - We calculated the number of launches on each site, the number and occurence of each orbit, and the number of occurence of mission outcome of the orbit.
    - And finally, we created a landing outcome label from outcome column.
- Here is the GitHub URL of the completed data wrangling related notebooks: https://github.com/dominiquedayato/Capstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

In [18]:
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

In [19]:
```
df['Class']=landing_class
df[['Class']].head(8)
```

Out[19]:

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

In [20]:
```
df.head(5)
```

Out[20]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPa
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	Na
		2012-										

Converting landing outcomes to classes.

EDA with Data Visualization

- We have used scatter plots to visualize the relationship between flight number and launch site, payload and launch site, flight number and orbit type, and payload and orbit type.
- We have used a bar chart to visualize the relationship between success rate of each orbit type.
- We have used a line plot to visualize the launch success yearly trend.
- Here is the GitHub URL of the completed EDA with data visualization notebook: https://github.com/dominiquedayato/Capstone/blob/main/jupyter-labs-eda-dataviz.ipynb.

EDA with SQL

- We have performed several SQL queries to:
  - 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.
  - List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months 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.
- Here is the GitHub URL of the completed EDA with SQL notebook: https://github.com/dominiquedayato/Capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We have created a folium map and added map objects such as markers, circles, marker clusters, mouse positions, and distance markers.
- We added circles to mark all launch sites.
- We added green markers for successful launches and red markers for failed launches.
- We added mouse positions and distances lines to show the distances between launch sites and its proximities.
- Here is the GitHub URL of the completed interactive map with Folium map: https://github.com/dominiquedayato/Capstone/blob/main/lab_jupyter_launch_site_location.jupyterlite%20(1).ipynb

Build a Dashboard with Plotly Dash

- We created a dashboard application that contains input components such as a dropdown list and a range slider to interact with a pie chart and a scatter point chart.
- The goal is to find more insights from the SpaceX dataset more easily than with static graphs.
- Here is the GitHub URL of the completed Plotly Dash lab: https://github.com/dominiquedayato/Capstone/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- We built a machine learning pipeline to predict if the first stage of the Falcon 9 lands successfully. This includes: Preprocessing, allowing us to standardize our data, and Train_test_split, allowing us to split our data into training and testing data.
- We trained the model and perform Grid Search, allowing us to find the hyper-parameters that allow a given algorithm to perform best. Using the best hyper-parameter values, we determined the model with the best accuracy using the training data.
- We tested Logistic Regression, Support Vector machines, Decision Tree Classifier, K-nearest neighbors, and we created a confusion matrix.
- Here is the GitHub URL of the completed predictive analysis lab: https://github.com/dominiquedayato/Capstone/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

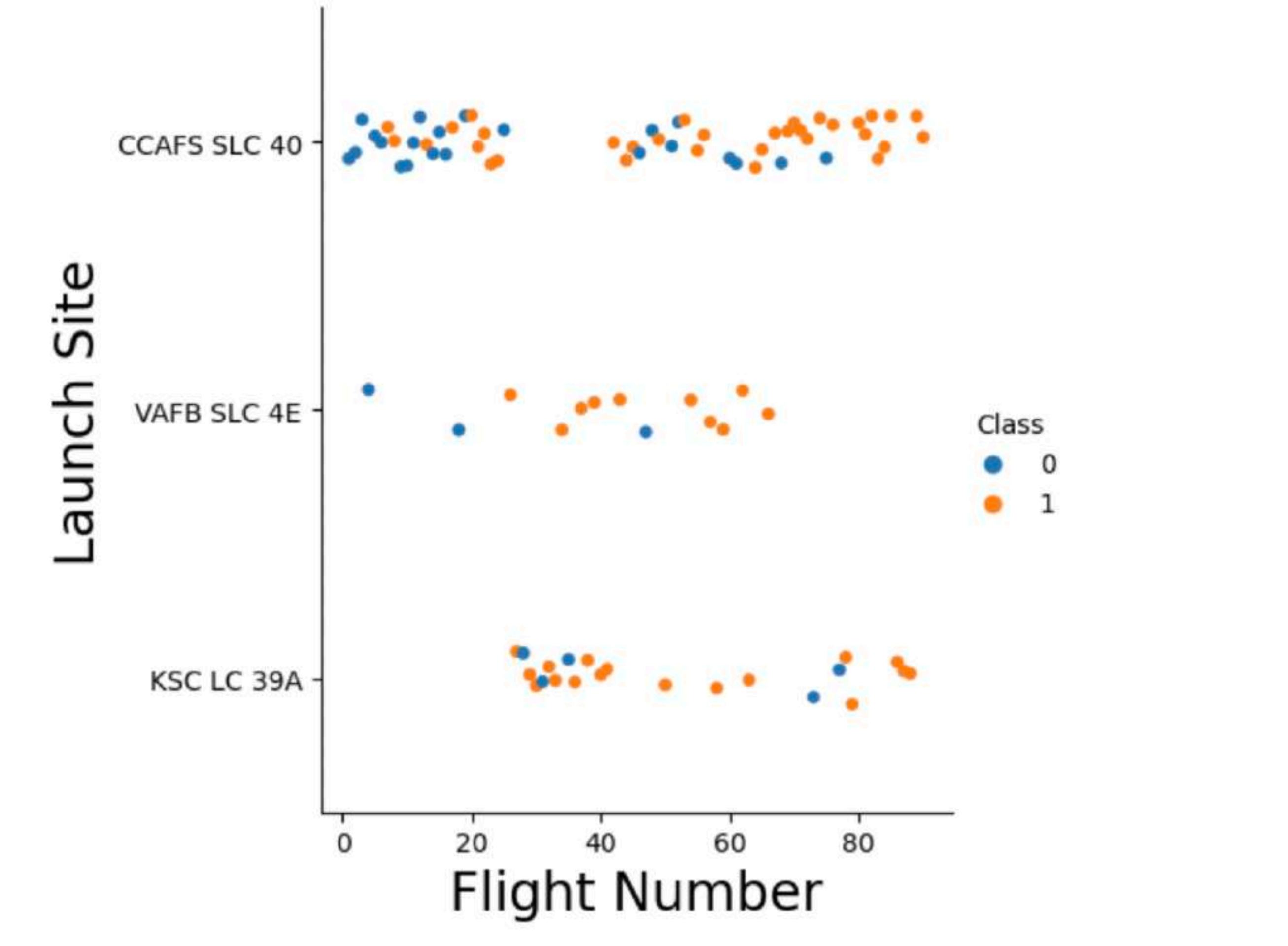
Results

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

Section 2

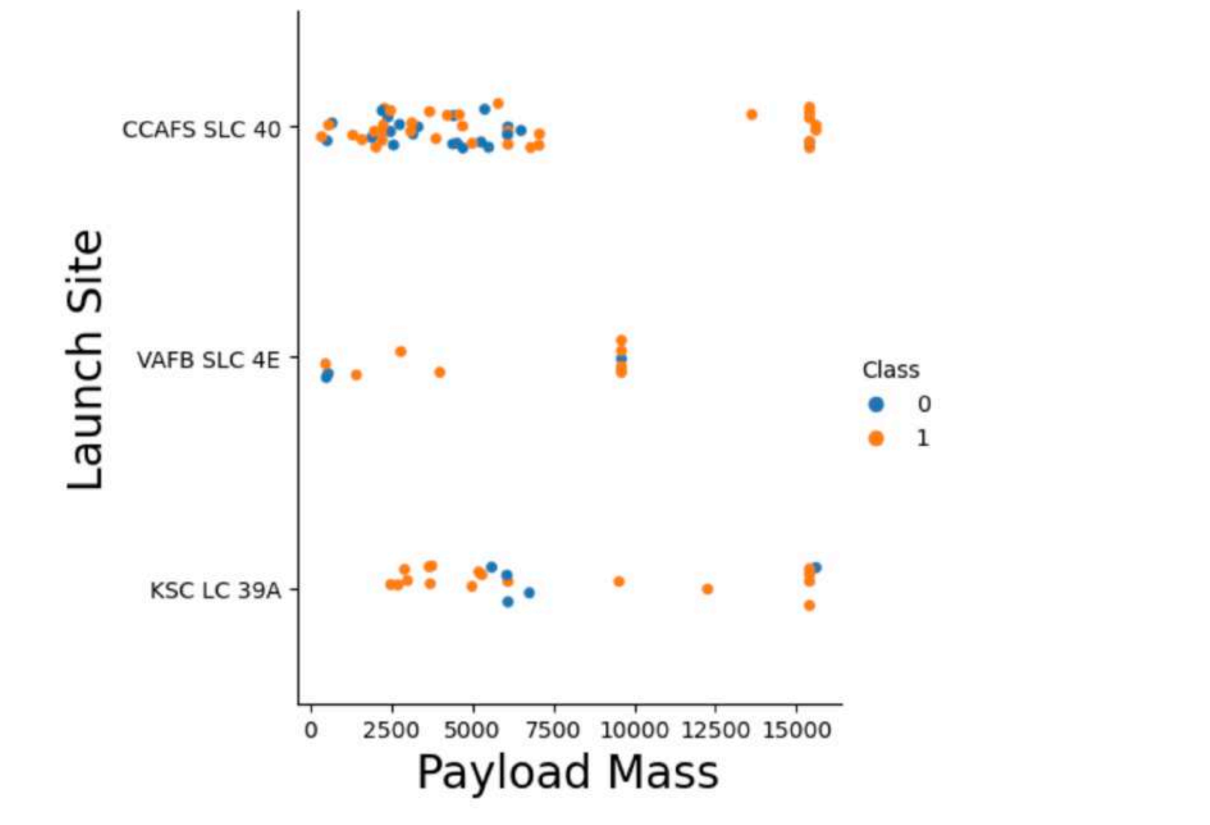
Insights drawn from EDA

Flight Number vs. Launch Site



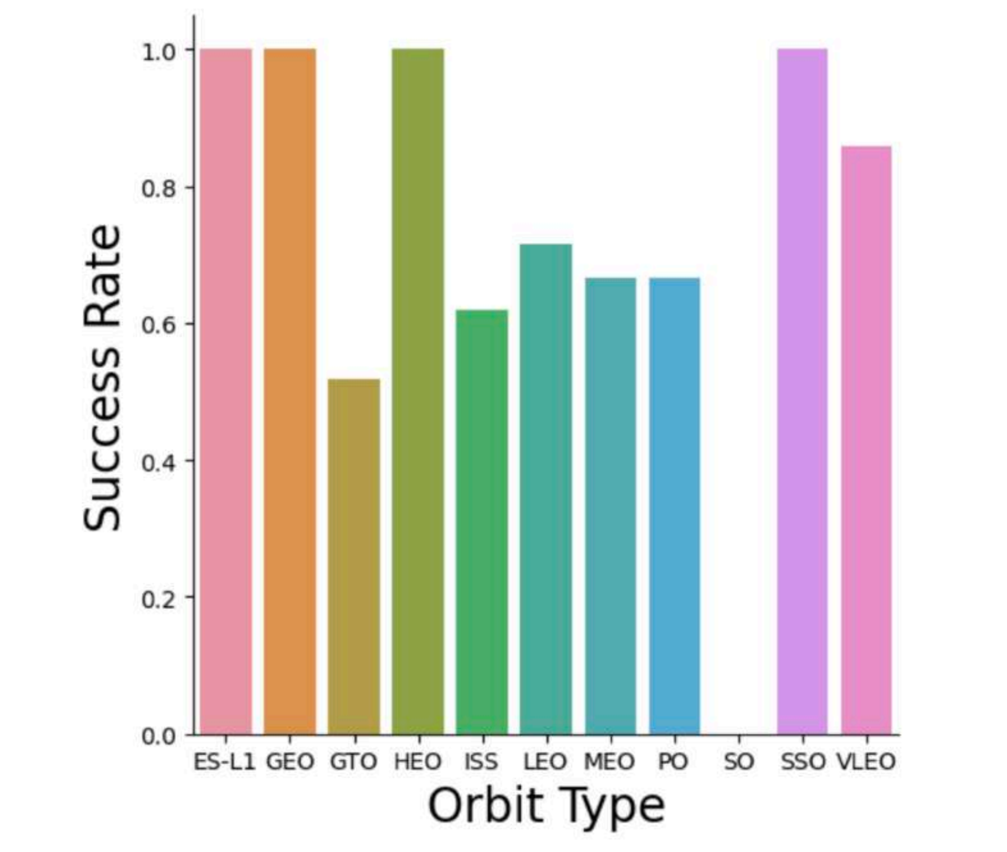
We can notice that as the flight number increases for all launch sites, the success rates increases as well.

Payload vs. Launch Site



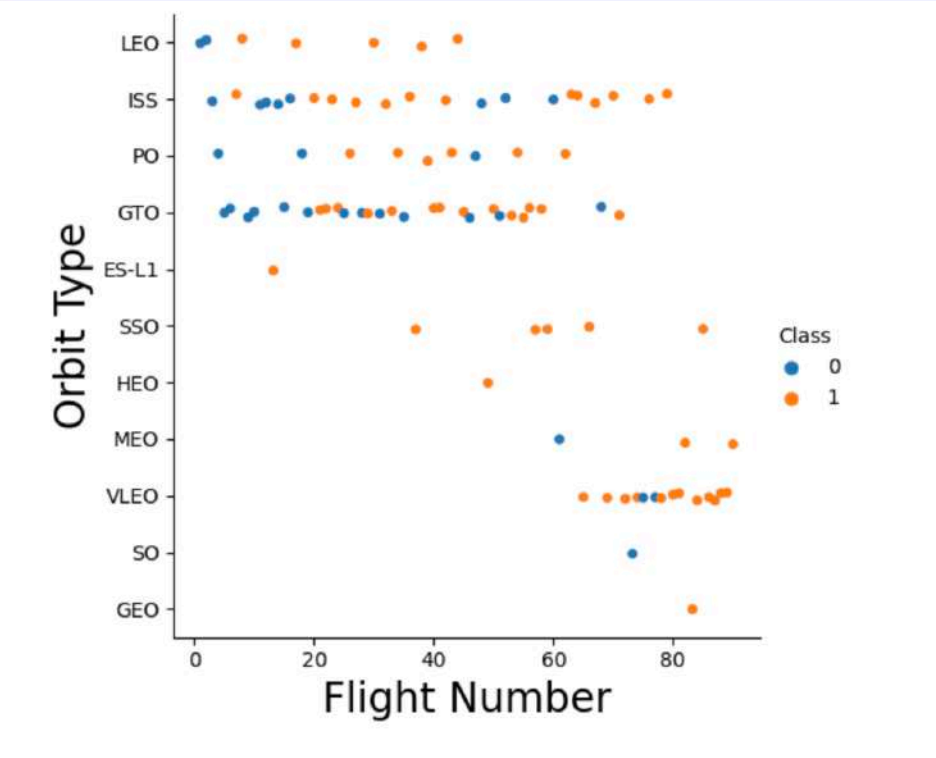
Here we can notice that as the Payload Mass goes up, the success rate also goes up.

Success Rate vs. Orbit Type



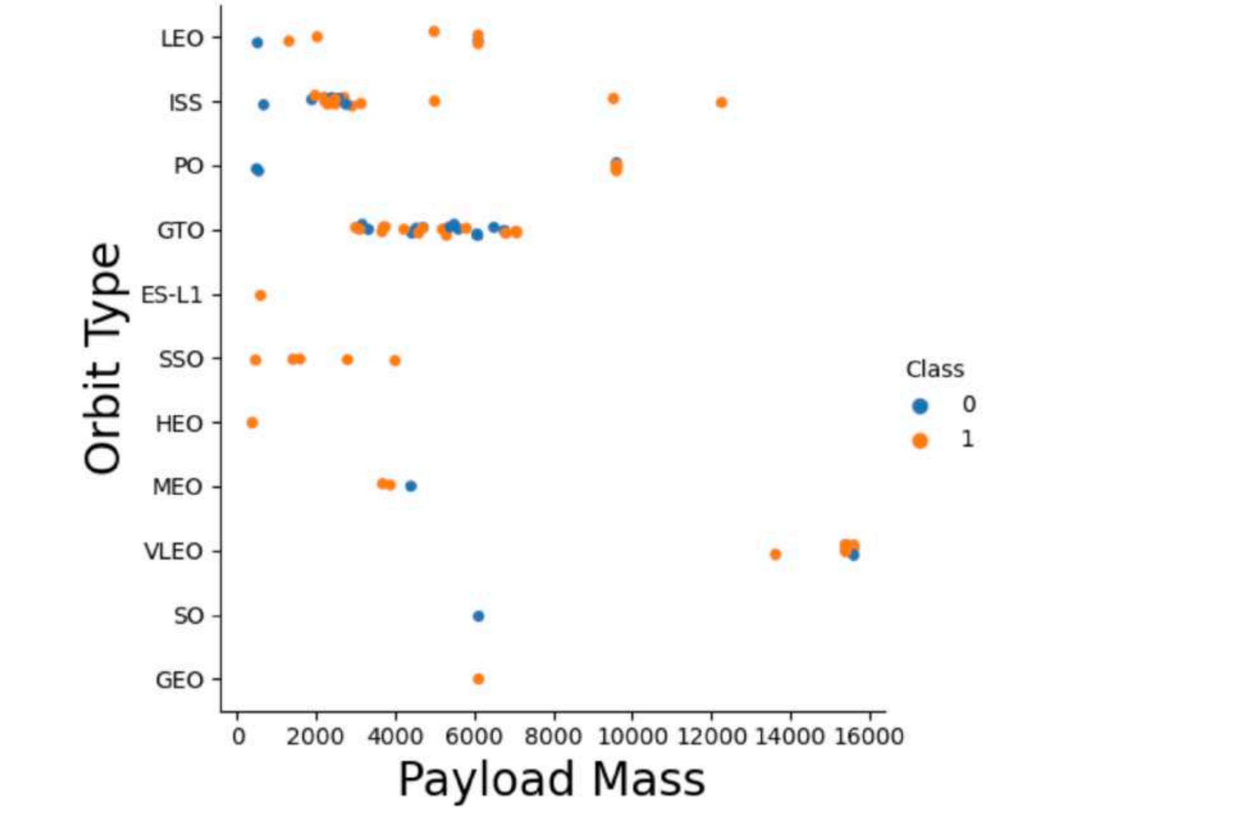
This bar chart shows us that the success rates for launches are at their highest with Orbits ES-L1, GEO, HEO, HEO, and SSO.

Flight Number vs. Orbit Type



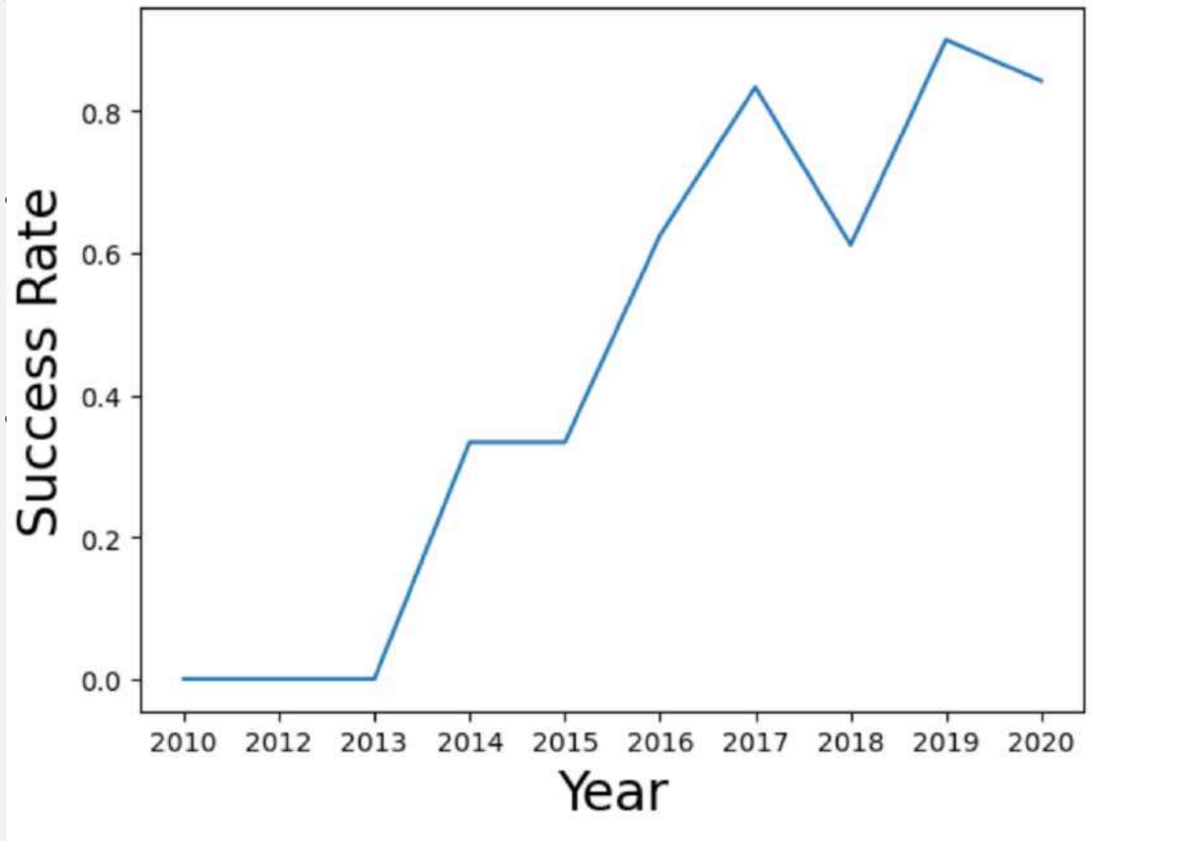
This scatter plot shows us the success rate increases as the flight number increases at Orbit LEO.

Payload vs. Orbit Type



This graph shows us the success rate increases as the payload mass increases at Orbit LEO

Launch Success Yearly Trend



This graph shows us the success rate kept going up over the years from 2010 to 2016, declined a little in 2017 and restarted to go back up in 2018.

All Launch Site Names

Task 1

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

```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL
```

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

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

We have used a SELECT DISTINCT query to get the names of the unique launch sites.

Launch Site Names Begin with 'CCA'

In [15]:
```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

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

Out[15]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (p
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 (p
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	N
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	N
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	N

Here we made a query to get 5 records where launch sites begin with ‘CCA’.

Total Payload Mass

Task 3

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_), CUSTOMER FROM SPACEXTBL WHERE Customer = 'NASA (CRS)'
```

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

SUM(PAYLOAD_MASS__KG_)	Customer
45596	NASA (CRS)

Here with a SELECT SUM query, we were able to calculate the total payload mass carried by boosters launched by NASA

Average Payload Mass by F9 v1.1

Task 4

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

In [20]:
```
%sql SELECT AVG(PAYLOAD_MASS__KG_), BOOSTER_VERSION FROM SPACEXTBL WHERE Booster_Version LIKE 'F9 v1.0%'
```

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

Out[20]:

AVG(PAYLOAD_MASS__KG_)	Booster_Version
340.4	F9 v1.0 B0003

Here with a SELECT AVG query, we were able to calculate the average payload mass by booster F9 v1.1

First Successful Ground Landing Date

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

Hint:Use min function

[22]:
```
%sql SELECT MIN(Date), LANDING_OUTCOME FROM SPACEXTBL WHERE Landing_Outcome = 'Success (ground pad)'
```

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

[22]:

MIN(Date)	Landing_Outcome
2015-12-22	Success (ground pad)

Here with a SELECT MIN query we were able to find out the first successful ground landing date.

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

In [26]:
```
%sql SELECT BOOSTER_VERSION, LANDING_OUTCOME, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE Landing_Outcome = 'Success
```

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

Out[26]:

Booster_Version	Landing_Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

Here is the query we used : **%sql** SELECT BOOSTER_VERSION, LANDING_OUTCOME, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_< 6000

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

In [27]:
```
%sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS TOTAL_NUMBER FROM SPACEXTBL GROUP BY MISSION_OUTCOME
```

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

Out[27]:

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

Here we used a SELECT COUNT AS TOTAL_NUMBER to find the total number of successful and failure mission outcomes.

Boosters Carried Maximum Payload

Task 8

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

In [30]:
```
%sql SELECT DISTINCT BOOSTER_VERSION, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYL
```

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

Out[30]:

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

We used a subquery and here it is: **%sql** SELECT DISTINCT BOOSTER_VERSION, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_)FROM SPACEXTBL);

2015 Launch Records

Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

In [31]:
```
%sql SELECT substr(Date,6,2) as month, DATE,BOOSTER_VERSION, LAUNCH_SITE, LANDING_OUTCOME FROM SPACEXTBL where La
```

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

Out[31]:

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)

Here is the query we used: **%sql** SELECT substr(Date,6,2) as month, DATE,BOOSTER_VERSION, LAUNCH_SITE, LANDING_OUTCOME FROM SPACEXTBL where Landing_Outcome = 'Failure (drone ship)' and substr(Date,0,5)='2015'

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

Task 10

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.

In [32]:

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

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

Out[32]:

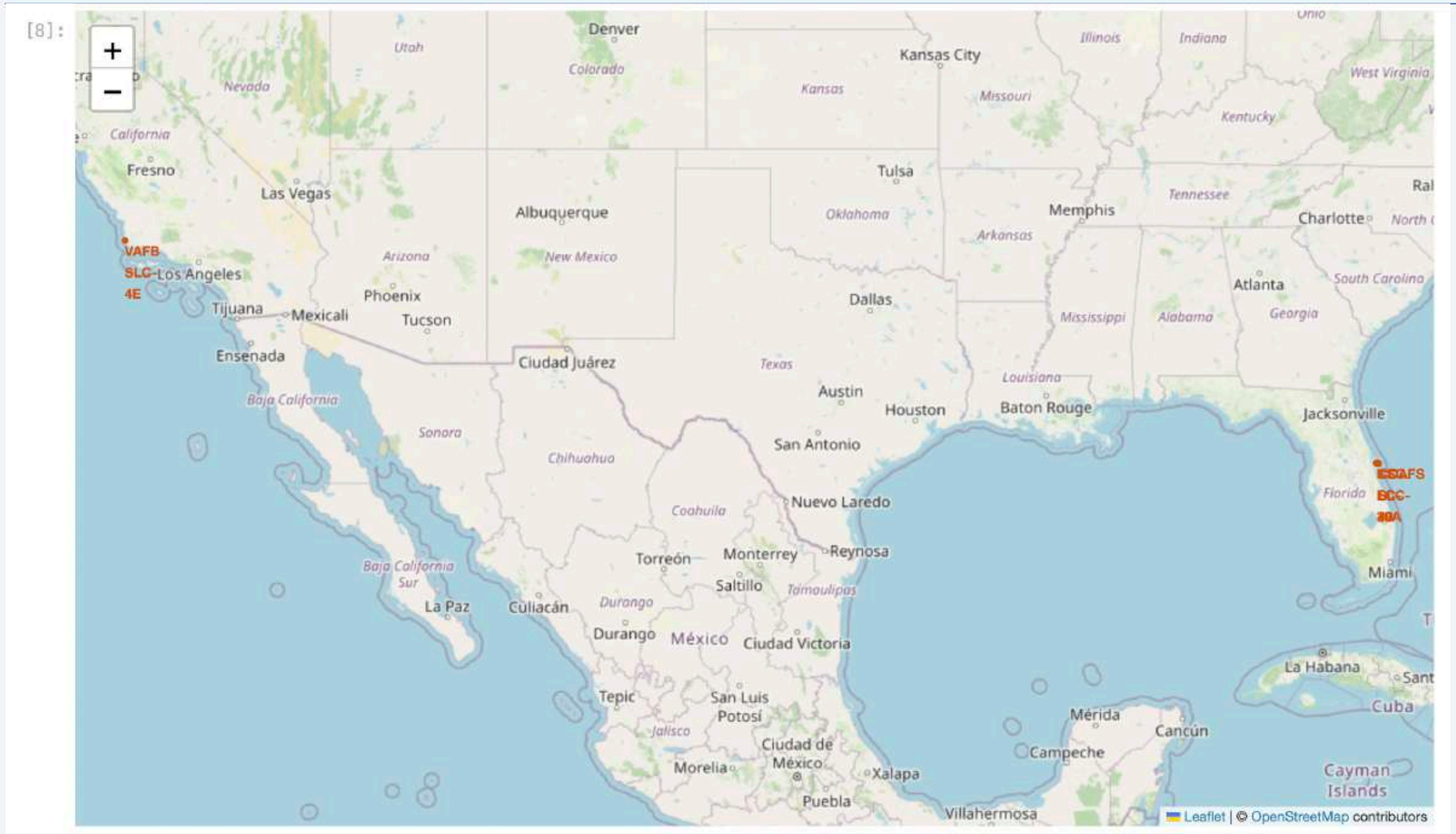
Landing_Outcome	TOTAL_NUMBER
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Here we used a SELECT, COUNT, GROUP BY and ORDER BY query to rank landing outcomes between 2010-06-04 and 2017-03-20

Section 3

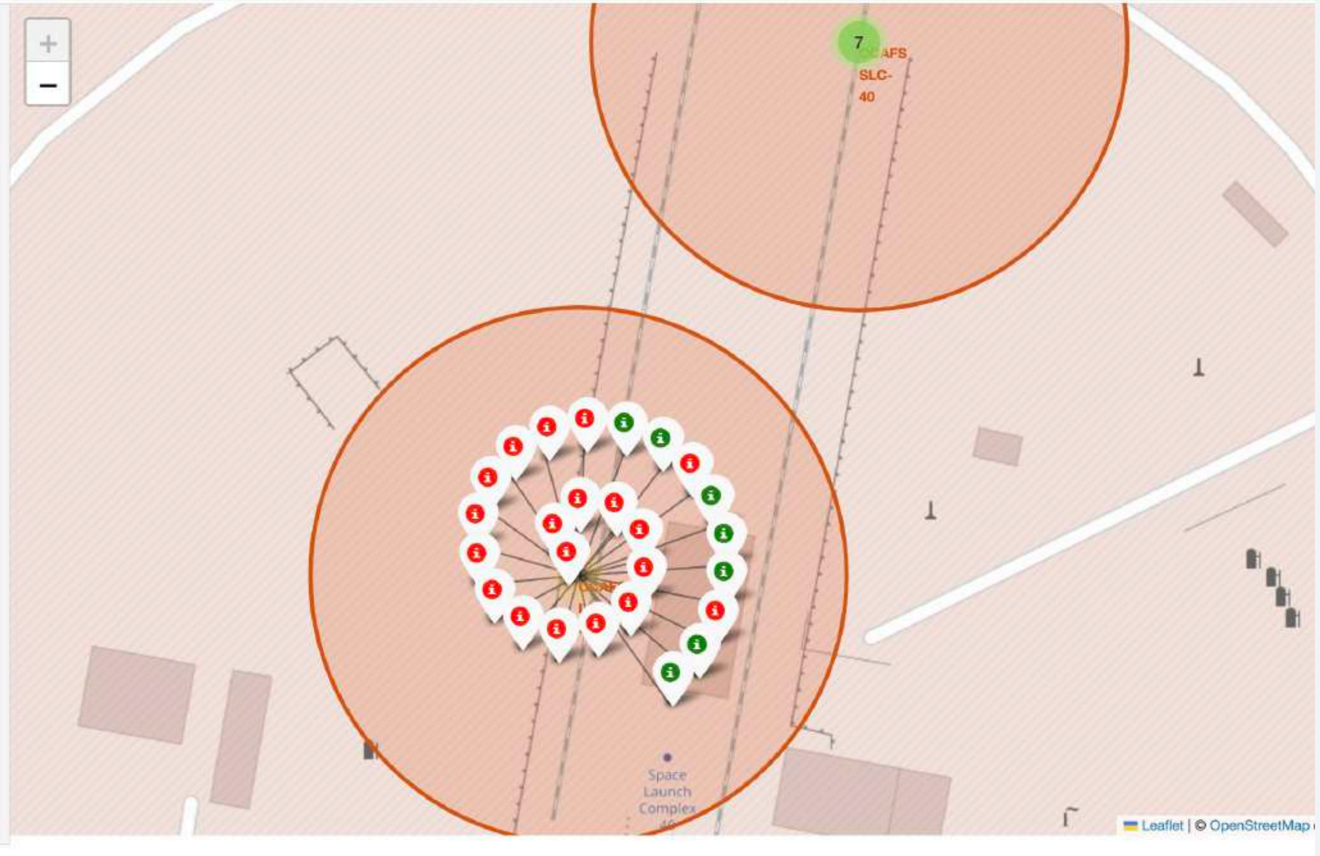
Launch Sites Proximities Analysis

Map of all launch sites



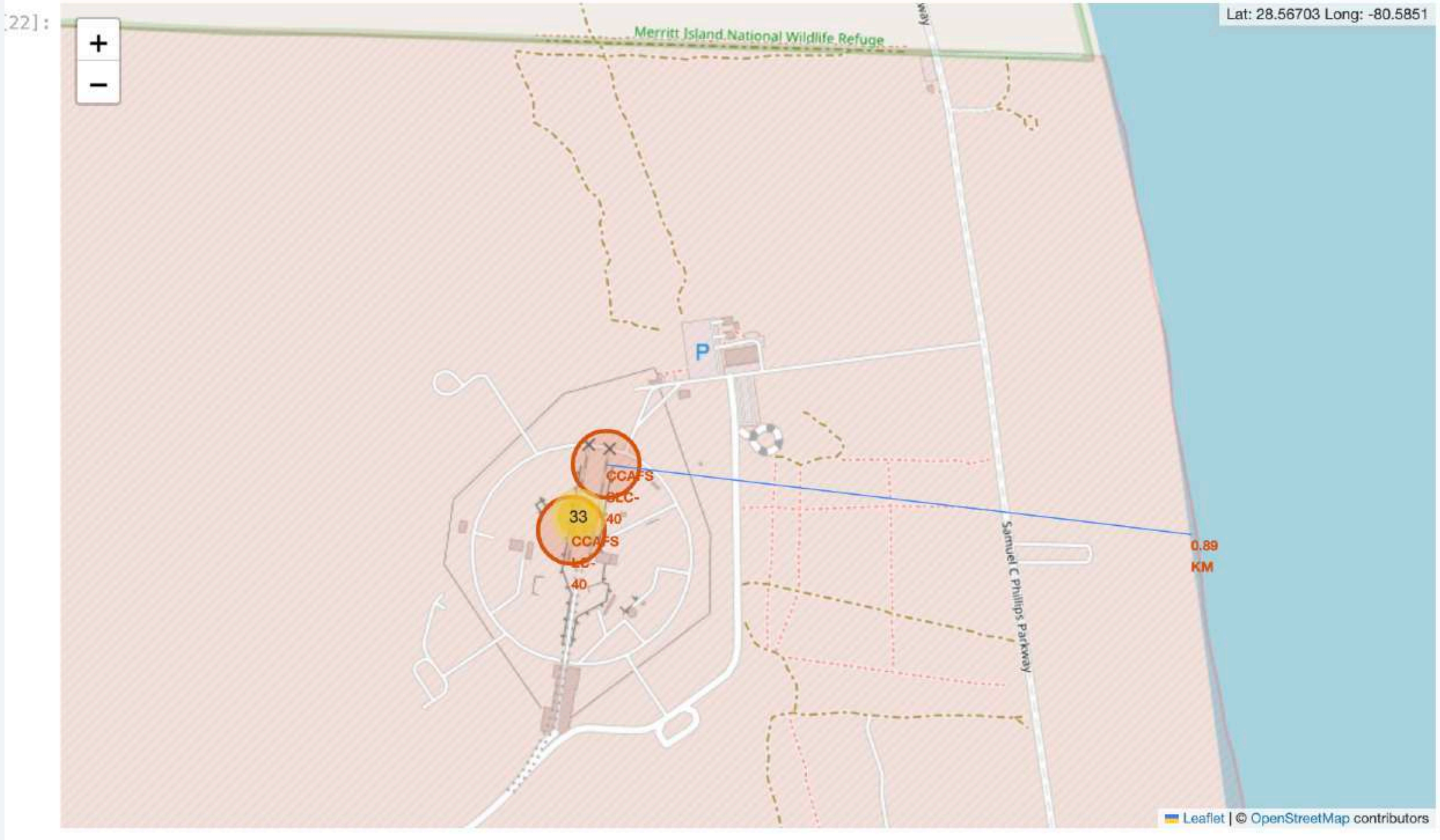
Here is a map of all SpaceX launch sites. We notice that most of the launch sites are located on the East coast.

Map with launch outcomes for each site.



Here on this zoomed in version of the map, we can see all the launch outcomes for site CCAFS LC - 40. Green markers represent a successful launch and red markers represent failed ones.

Map of distances between a launch site and its proximities

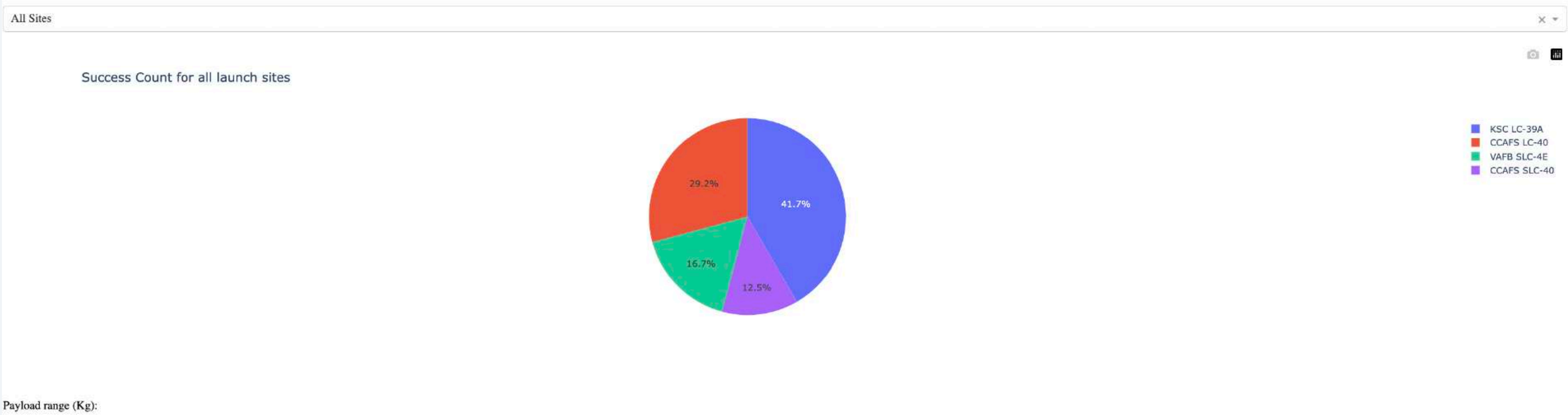


Here on this map we can see the distance between site CCAFS SLC-40 and the coastline.

Section 4

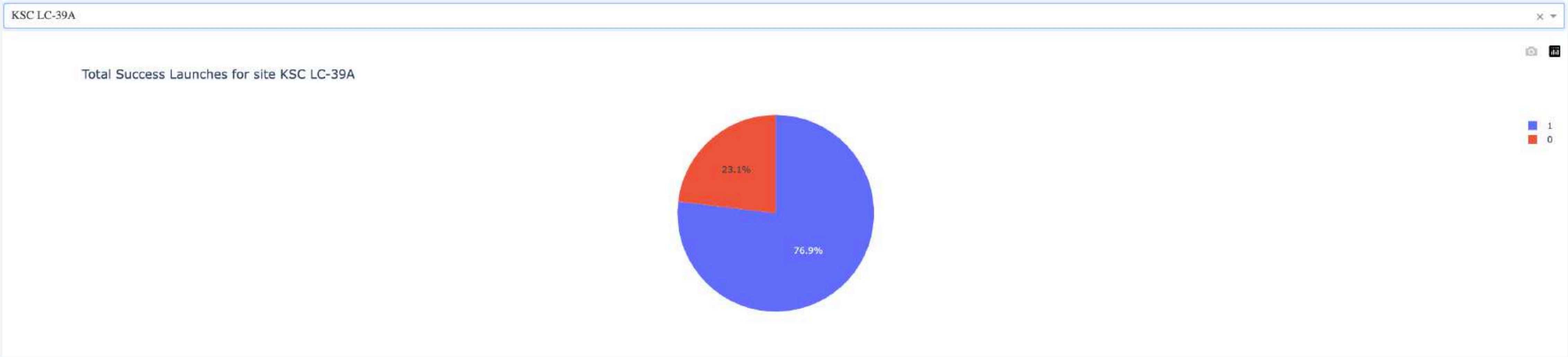
Build a Dashboard with Plotly Dash

Piechart of launch success count for all sites.



On this pie chart, we can see that KSC LC-39A launch site has the most successful launches out of all sites.

Pie chart for KSC LC-39A



On this chart, we can see that the launch site KSC LC-39A has a 76.9% success rate.

Payload vs Launch Outcome scatter plots for all sites.

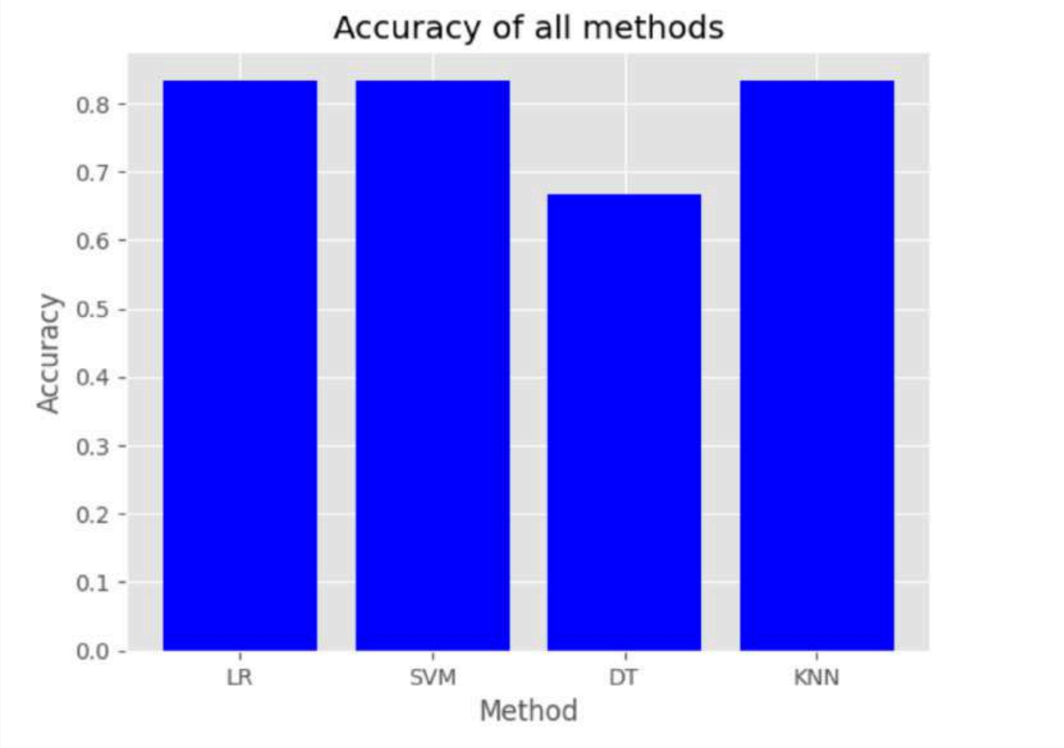
Here we see from 5k to 10k range for the payload mass, there is no relationship between the success rate and the booster category.

Here we see that from 2k to 5k for the payload mass, booster FT has a very high success rate.

Section 5

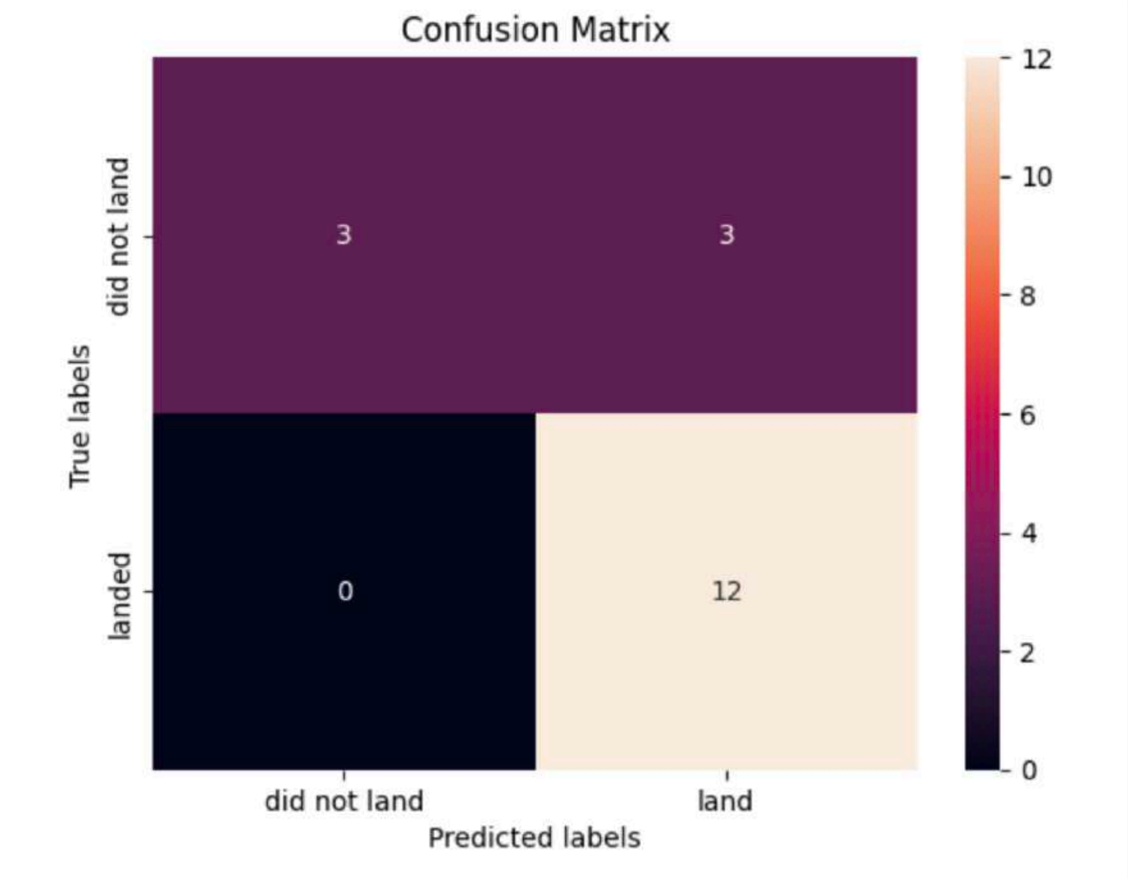
Predictive Analysis (Classification)

Classification Accuracy



On this bar chart, we can see that the decision tree is the least accurate method. All other methods have the same accuracy which is above 0.8.

Confusion Matrix



The confusion matrix for the logistic regression, the KNN method, and the SVM method shows us we have a very good amount of true positive which is great for prediction.

Conclusions

- As the flight number increases for all launch sites, the success rates increases as well.
- As the payload mass goes up, the success rate also goes up.
- The success rates for launches are at their highest with Orbits ES-L1, GEO, HEO, HEO, and SSO.
- All launch sites are located on a coast and very closed to coastlines
- Launch site KSC LC-39A has the most successful launches out of all sites with a success rate of almost 77%.
- The logic regression, the support vector machine, and the k nearest neighbors are the best methods for prediction.

Thank you!