IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Juan Jose Echevarria Aguirrezabal
November 2022

Outline

Executive Summary

- Summary of methodologies
  - Data Collection API and Webscraping
  - Data Wrangling
  - Data Analysis with SQL
  - Data Visualization
  - Interactive Visual Analytics
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis Result
  - Interactive Visual Analytics Result
  - ML Prediction Result

Introduction

- Project background and context

SpaceX 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. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

Problems you want to find answers

1. What characteristics are present in a Falcon 9 first stage successful landing
2. How does Payload Mass, Booster Version and Landing Outcome affect a successful mission
3. What Launch Sites and Orbits have higher success rates
4. How to determine the best location for a Launch Site

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - The data was collected thru the SpaceX API and Webscraping from Wikipedia
- Perform data wrangling
  - The data was cleaned, removed NULL values and Falcon 1 data; finally, created dummy variables for categorical columns
- 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

- First we called the SpaceX API for different variables like LaunchPad, Rocket, PayLoads and Core.
- We received a JSON file and transformed it to a Pandas DataFrame after normalizing the data with the json_normalize() function.
- Then we filtered the data to include only information for Falcon 9 missions.
- After checking the data we had some missing values in the PayLoadMass variable, we filled this values with the mean of PayLoadMass

Data Collection – SpaceX API

- With these code with obtained the data thru the SpaceX API

- You can check the full code in the following Jupyter Notebook: https://github.com/juanjoetxebarria/DSCapstone/blob/main/Data%20Collection%20API.ipynb

```
In [6]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [7]: response = requests.get(spacex_url)
```

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

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

```
In [10]: response.status_code
Out[10]: 200
```

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

```
In [12]: # Use json_normalize meethod to convert the json result into a dataframe
static_json_df = response.json()
data = pd.json_normalize(static_json_df)
```

```
[28]: # Create a data from launch_dict
launch_df = pd.DataFrame(launch_dict)
```

```
In [53]: # Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = launch_df[launch_df['BoosterVersion'] != 'Falcon 1']
data_falcon9
```

Data Collection - Scraping

- We used BeatifulSoup to extract the data and Pandas to import a DataFrame

- You can check the full code in the following Jupyter Notebook: https://github.com/juanjoetxebarria/DSCapstone/blob/main/Data%20Wrangling.ipynb

```
In [4]: static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_
```

Next, request the HTML page from the above URL and get a `response` object

```
In [5]: # use requests.get() method with the provided static_url
        # assign the response to a object
        html_data = requests.get(static_url)
        html_data.status_code

Out[5]: 200

In [6]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
        soup = BeautifulSoup(html_data.text, 'html.parser')

In [8]: # 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')
```

Starting from the third table is our target table contains the actual launch records.

```
In [9]: # Let's print the third table and check its content
        first_launch_table = html_tables[2]
        print(first_launch_table)

In [14]: df=pd.DataFrame(launch_dict)
         df.head()
```

Data Wrangling

- We calculated how many launches happened from each Launch Site
- And the orbits they were trying to reach
- Followed by the landing outcomes
- Finally, we created a new variable called Class that shows if the landing was successful (1) or unsuccessful (0)
- You can check the full code in the following Jupyter Notebook: https://github.com/juanjoetxebarria/DSCapstone/blob/main/Data%20Wrangling.ipynb

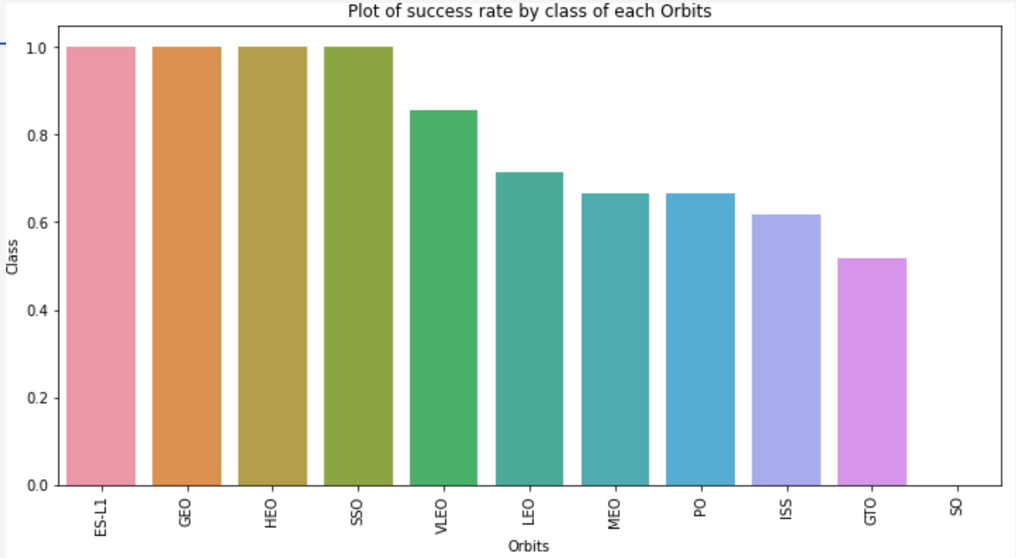
```
5]: # Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()

5]: CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

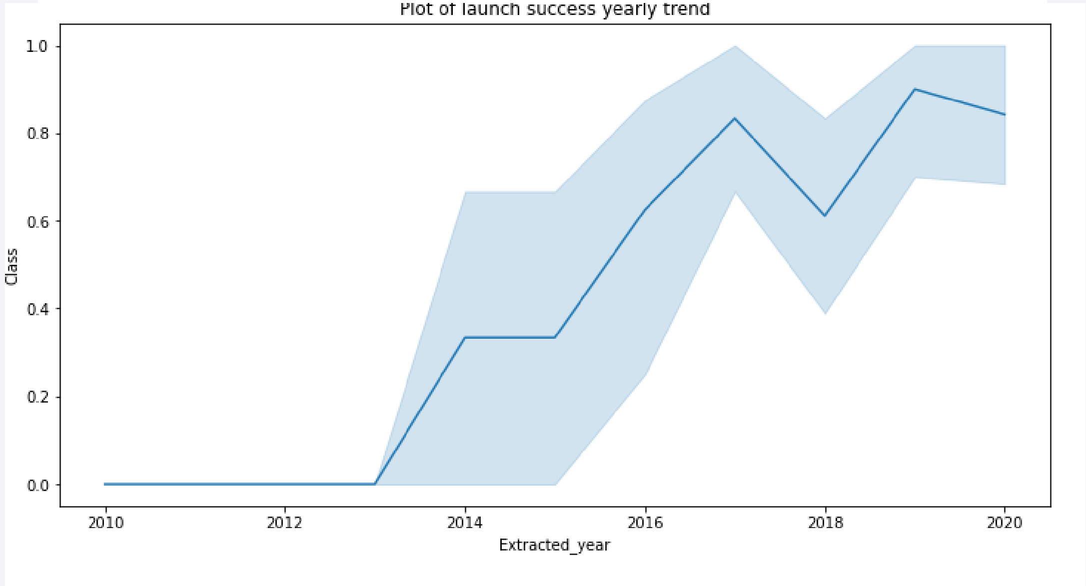
```
In [6]: # Apply value_counts on Orbit column
df['Orbit'].value_counts()

Out[6]: GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

```
In [9]: # landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes

Out[9]: True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

EDA with Data Visualization

- We plotted the success rate by Orbit and the trend of success throught the years so we could tell what kind of missons are more successful and how the company is getting better at launches.
- You can check the full code in the following Jupyter Notebook:https://github.com/juanjoetxebarria/DSCapstone/blob/main/EDA%20with%20data%20Visualization.ipynb





EDA with SQL

- The following queries were applied to the data using SQL Magic:
  - Names of the unique Launch Sites
  - Total and Average of Payload Mass carried by Falcon 9
  - Names of Boosters with certain Payload Mass
  - Totals by mission outcome
  - Ranking of missions by landing utcome
- You can check the full code in the following Jupyter Notebook: https://github.com/juanjoetxebarria/DSCapstone/blob/main/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- We marked the interactive map with the following:
  - The launch sites using longitude and latitude
  - We coded the result of the mission with colors green (successful, type 1) or red (unsuccessful, type 0)
  - We calculated the distance to relevant points like the shoreline, highways of cities
- All of this to try and show the characteristics of a successful Launch pad location
- You can check the full code in the following Jupyter Notebook: https://github.com/juanjoetxebarria/DSCapstone/blob/main/Interactive%20Visual%20Analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- Using Plotly Dash we created a dashboard to show the successful and unsuccessful missions by launch site and the relationship between the booster used in launch and the total weight the rocket was launching and the actual result of the launch
- We used this graphs to prove which Launch Sites have a higher success rate and how the booster and the payload mass affect the result of the mission
- You can check the full code of the python app at: https://github.com/juanjoetxebarria/DSCapstone/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- After accessing the data, we separate the variable Class in an Numpy array and the rest of the data in a Pandas DataFrame.
- Then, we need to split the data into training and testing datasets, we keep 80% of the data for training and 20% for testing.
- After that, we use the function GridSearchCV to find the best available hyperparameters for each of our 4 models (Logistic Regression, SVM, Decision Tree and K Nearest Neighbor) using accuracy as the metric to determine the best available fit.
- You can check the full code in the following Jupyter Notebook: https://github.com/juanjoetxebarria/DSCapstone/blob/main/Machine%20Learning%20Prediction.ipynb

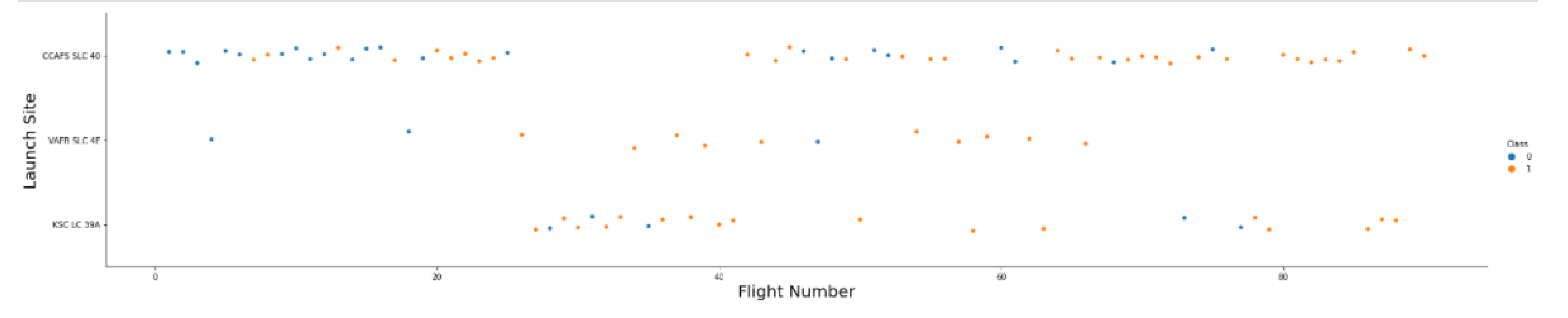
Results

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

Section 2

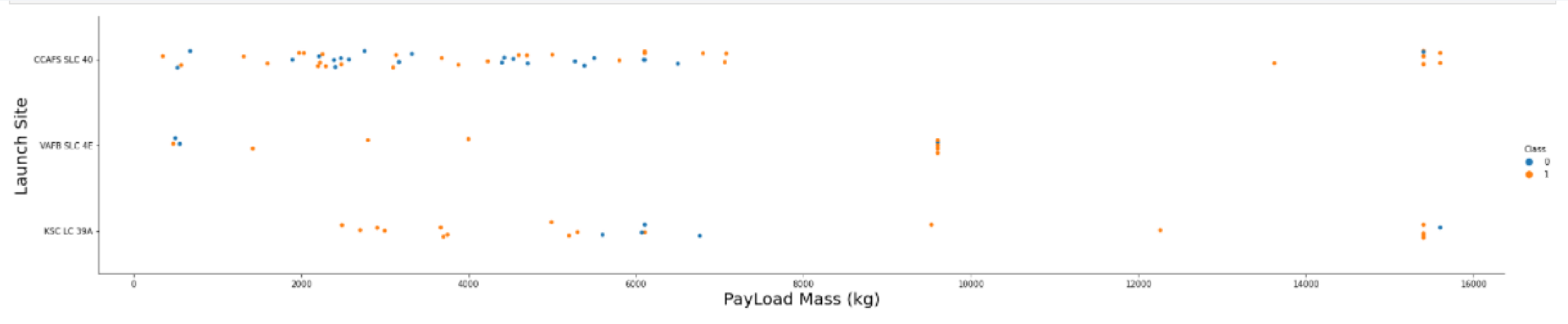
Insights drawn from EDA

Flight Number vs. Launch Site



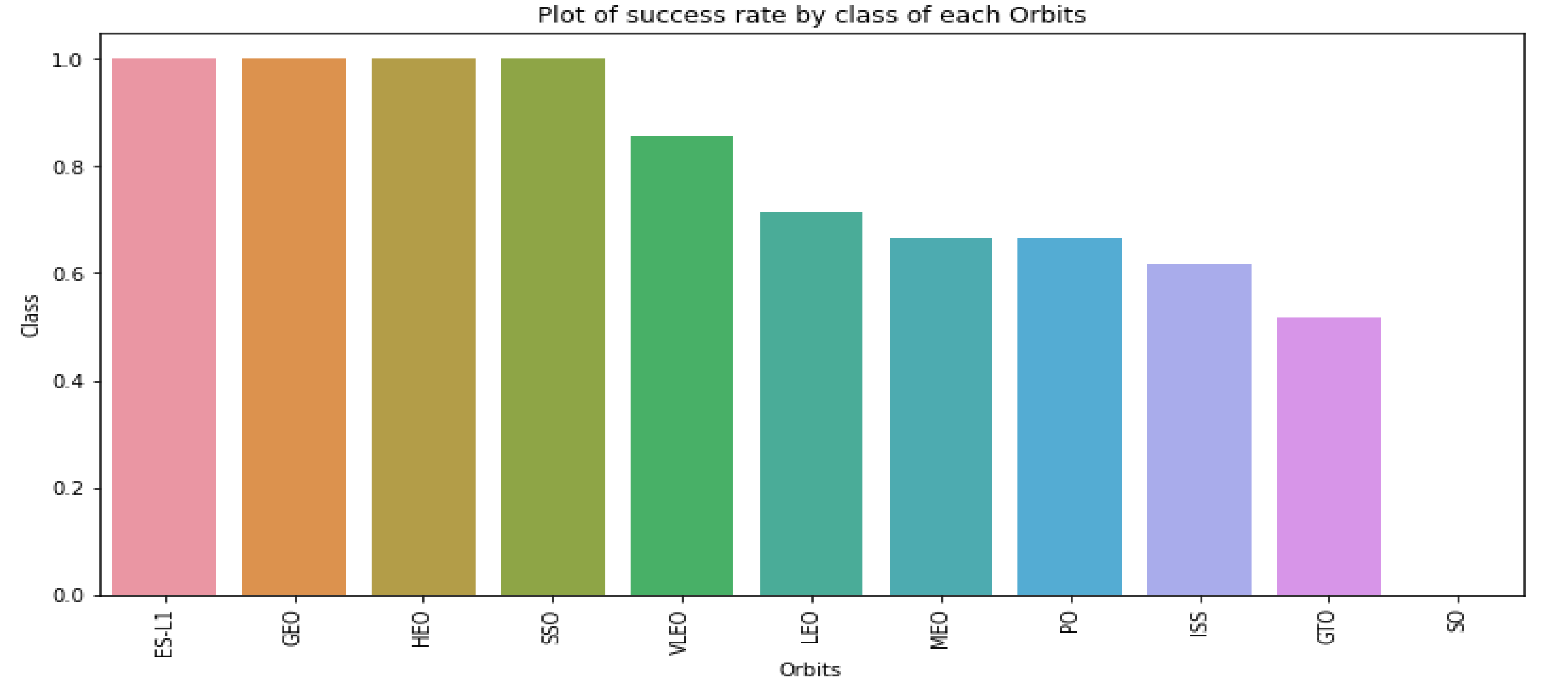
- We can see that as the company gains experience in each Launch Site the successful mission are more common, ie. There are less blue dots or unsuccessful missions

Payload vs. Launch Site



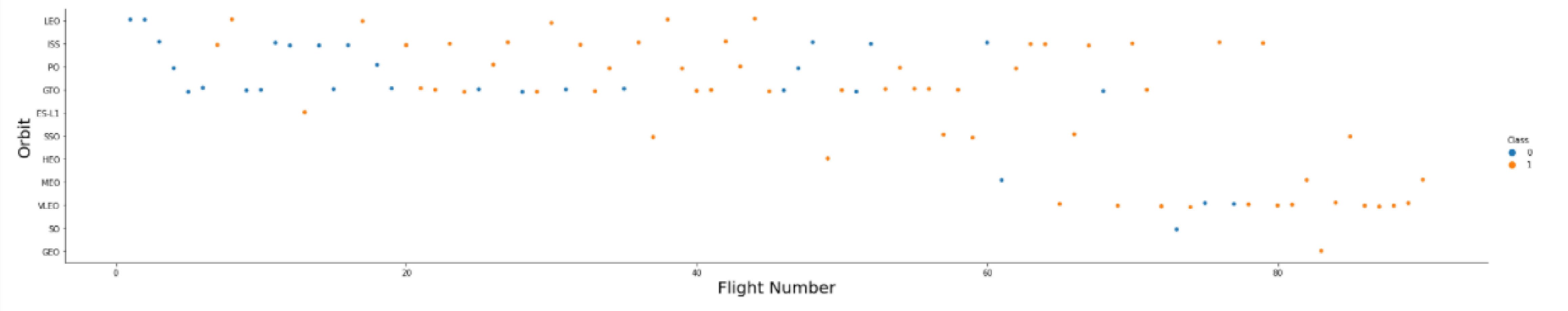
- Launch Sites CCAPS and KSC can handle the largest Payload without affecting the result of the mission. Launch site VAFB hasn´t had launches with Payload higher than 10,000 Kg

Success Rate vs. Orbit Type



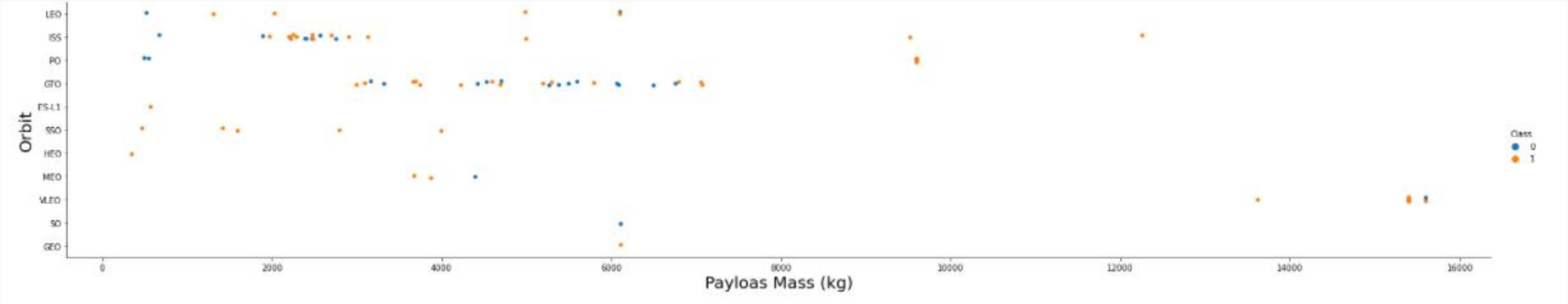
- ES-L1, GEO, HEO and SSO have the highest success rate, while GTO and SO have the lowest

Flight Number vs. Orbit Type



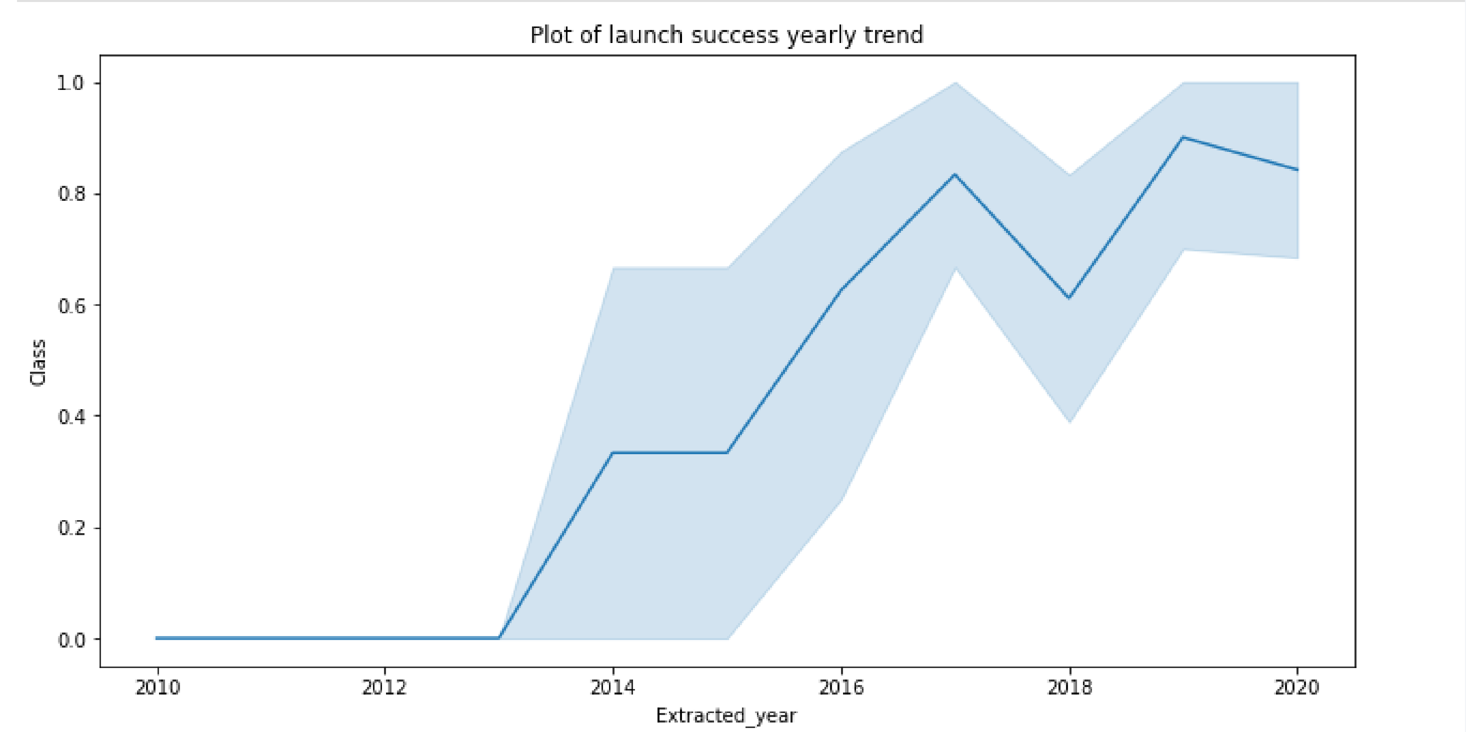
- With more experience comes better results in the launches. In time, there are less and less blue dots for failed launches.

Payload vs. Orbit Type



- It appears like orbit GTO struggles with Payload Masses between 4,000 and 8,000 Kg. Orbit VLEO is the one that can handle the larger Payload Mass

Launch Success Yearly Trend



- As explained before, there is constant uptrend in success as the company obtains more experience with each launch

All Launch Site Names

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

```
In [5]: %sql select launch_site from spacex group by launch_site

 * ibm_db_sa://kjn80744:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/bludb
Done.
```

Out[5]:

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

- Using SQL Magic we can find all the launch site names using GROUP BY

Launch Site Names Begin with 'CCA'

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

```
In [6]: %sql select * from spacex where launch_site like '%CCA%' limit 5
```

```
 * ibm_db_sa://kjn80744:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/bludb
Done.
```

Out[6]:

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

- We can look up the first 5 launches from launch sites thant include CCA using LIKE and LIMIT 5

Total Payload Mass

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

In [8]:
```
%sql select sum(PAYLOAD_MASS__KG_) from spacex where customer like 'NASA (CRS)'
```

```
 * ibm_db_sa://kjn80744:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/bludb
Done.
```

Out[8]:

1
45596

- By selecting sum in a column we can get the total Payload Mass for customer NASA

Average Payload Mass by F9 v1.1

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

```
In [9]: %sql select avg(PAYLOAD_MASS__KG_) from spacex where booster_version like 'F9 v1.1%'
 * ibm_db_sa://kjn80744:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/bludb
Done.
Out[9]:
```

1
2534

- Using LIKE and the function AVG we can calculate the average Payload that the booster F9 v1.1 is carrying

First Successful Ground Landing Date

```
In [12]: %sql select min(date) from spacex where Landing__Outcome like 'Success (ground pad)'

 * ibm_db_sa://kjn80744:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/bludb
Done.
Out[12]:
```

1
2015-12-22

- In December 22 of 2015 was the first successful landing outcome in a ground pad

Successful Drone Ship Landing with Payload between 4000 and 6000

```
In [14]: %sql select booster_version from spacex where Landing__Outcome like 'Success (drone ship)' and PAYLOAD_MASS__KG_>4000 and PAYLOAD_MASS__KG_<6000

 * ibm_db_sa://kjn80744:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/bludb
Done.
```

Out[14]:

booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

- With these 4 booster versions there was a successful landing in a drone ship with Payloads between 4,000 and 6,000 Kg

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

In [16]:
```
%sql select mission_outcome,count(*) from spacex group by mission_outcome
```

```
 * ibm_db_sa://kjn80744:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/bludb
Done.
```

Out[16]:

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

- Of the 101 missions, 99 were considered a success with only one failure and a success with Payload status unclear

Boosters Carried Maximum Payload

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

In [19]:
```
%sql select booster_version,PAYLOAD_MASS__KG_ from spacex where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from spacex) order by booster_version
```

```
 * ibm_db_sa://kjn80744:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/bludb
Done.
```

Out[19]:

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

- These 12 Boosters achieved the maximum of 15,600 Payload Mass

2015 Launch Records

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [21]:
```
%sql select Landing__Outcome, booster_version, launch_site, date from spacex where year(date)=2015 and Landing__Outcome='Failure (drone ship)'
```

```
 * ibm_db_sa://kjn80744:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/bludb
Done.
```

Out[21]:

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

- Only 2 launches failed to land in a drone ship, they happened in January and April of 2015

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

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

In [23]:
```
%sql select Landing__Outcome, count(*) as count_landing from spacex where date between '2010-06-04' and '2017-03-20' group by Landing__Outcome order by count_landing de
```

```
 * ibm_db_sa://kjn80744:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/bludb
Done.
```

Out[23]:

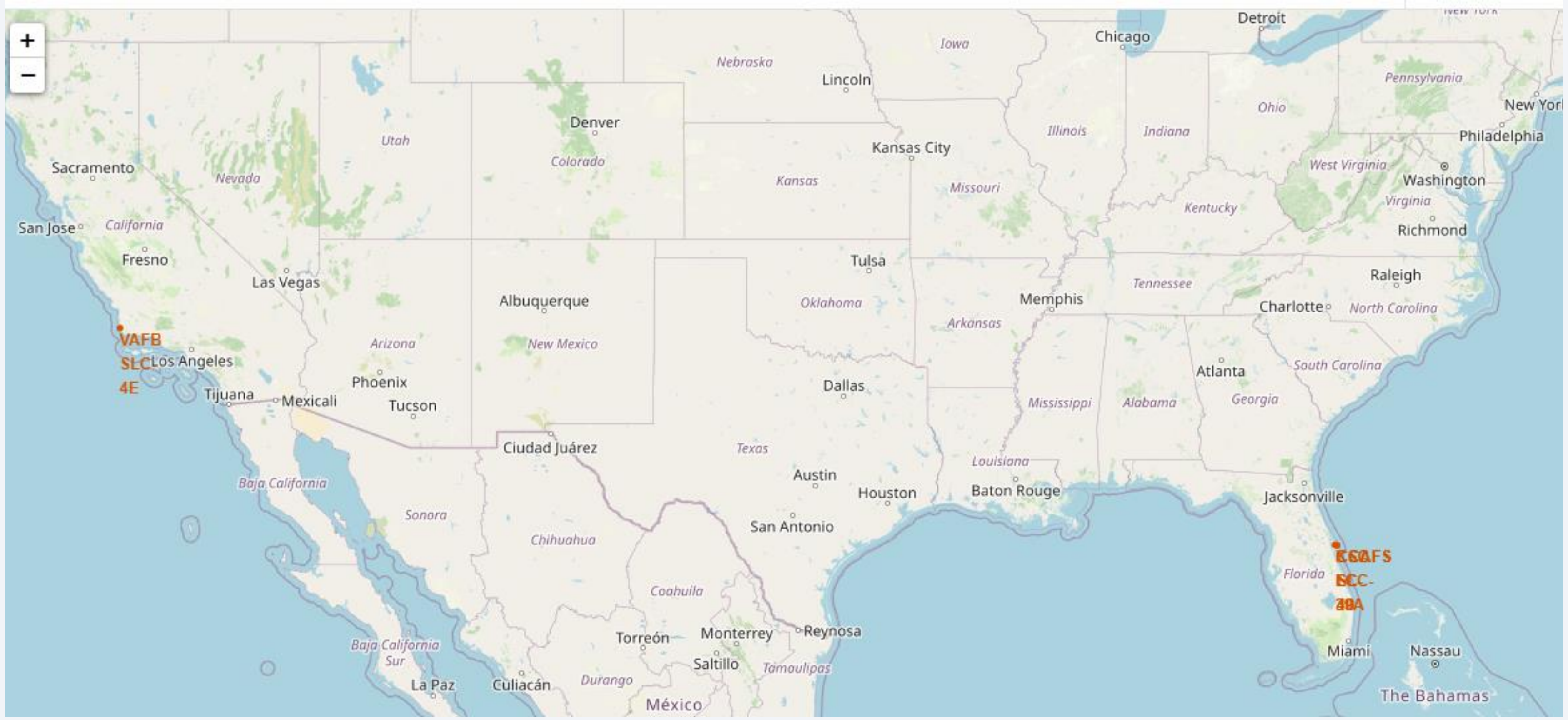
landing__outcome	count_landing
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

- Between June of 2010 and March of 2017, there were 31 launches, with 10 where the was no attempt to land, 10 that could be considered failures and 11 successful landings.

Section 3

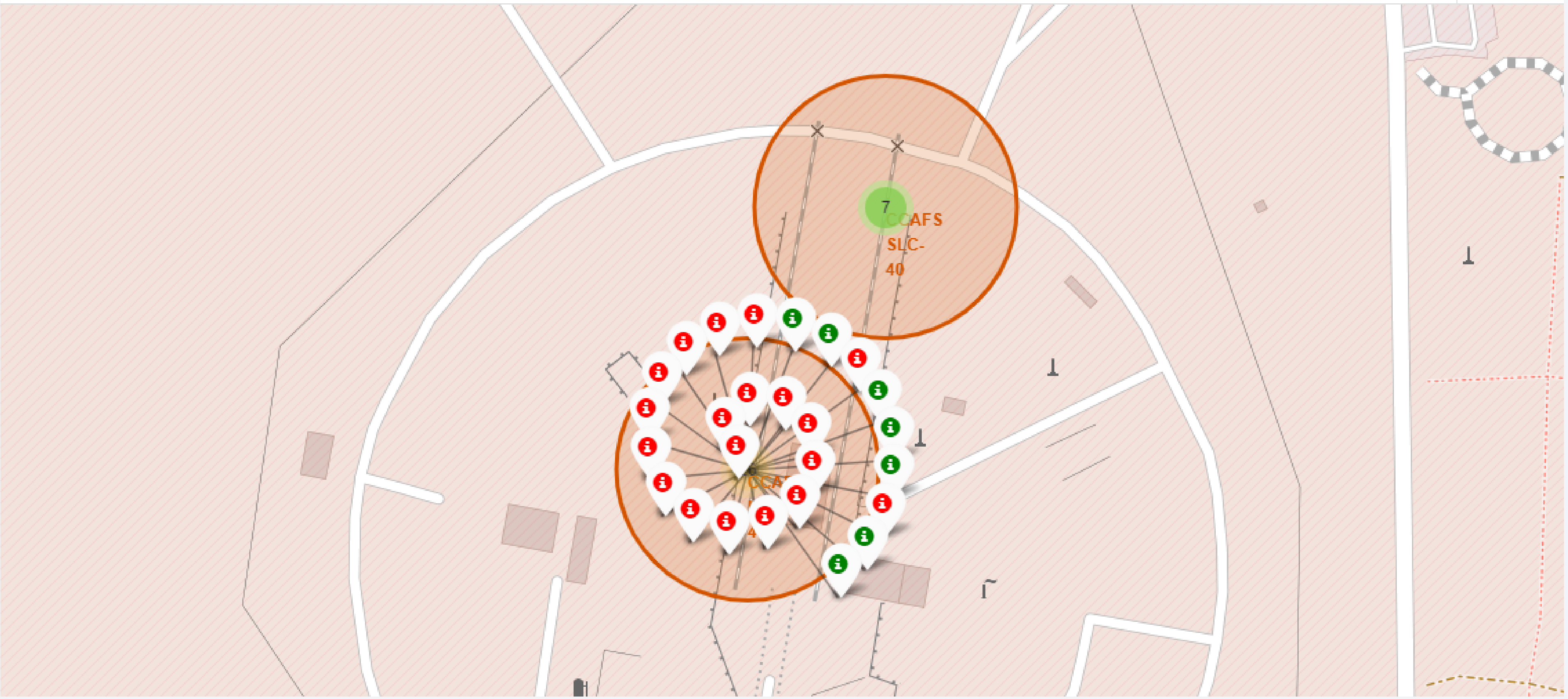
Launch Sites Proximities Analysis

Location of the Launch Sites



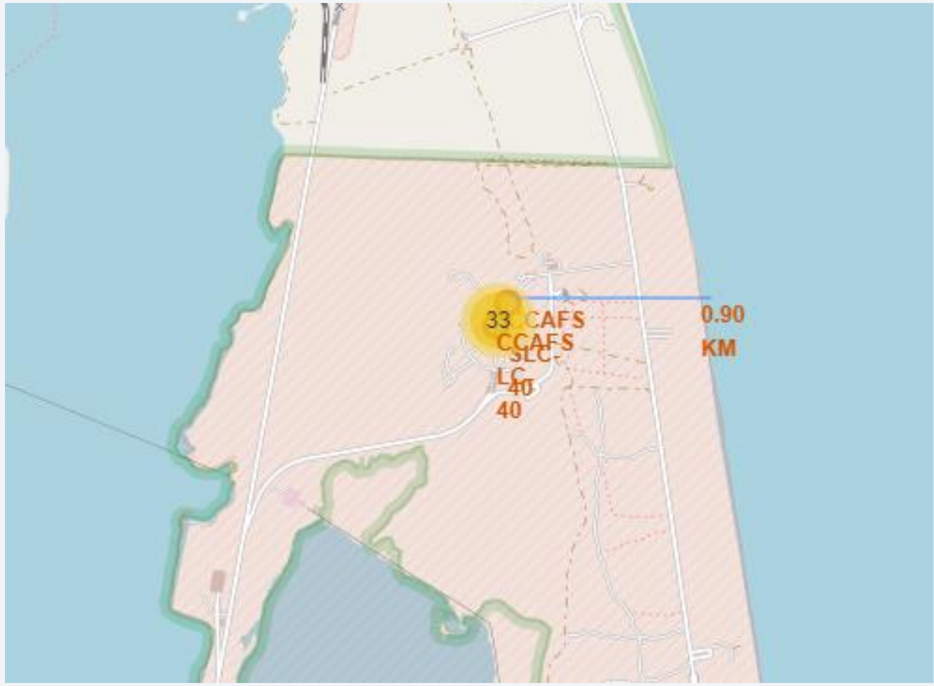
- The launch sites are located in Florida and California near the shoreline

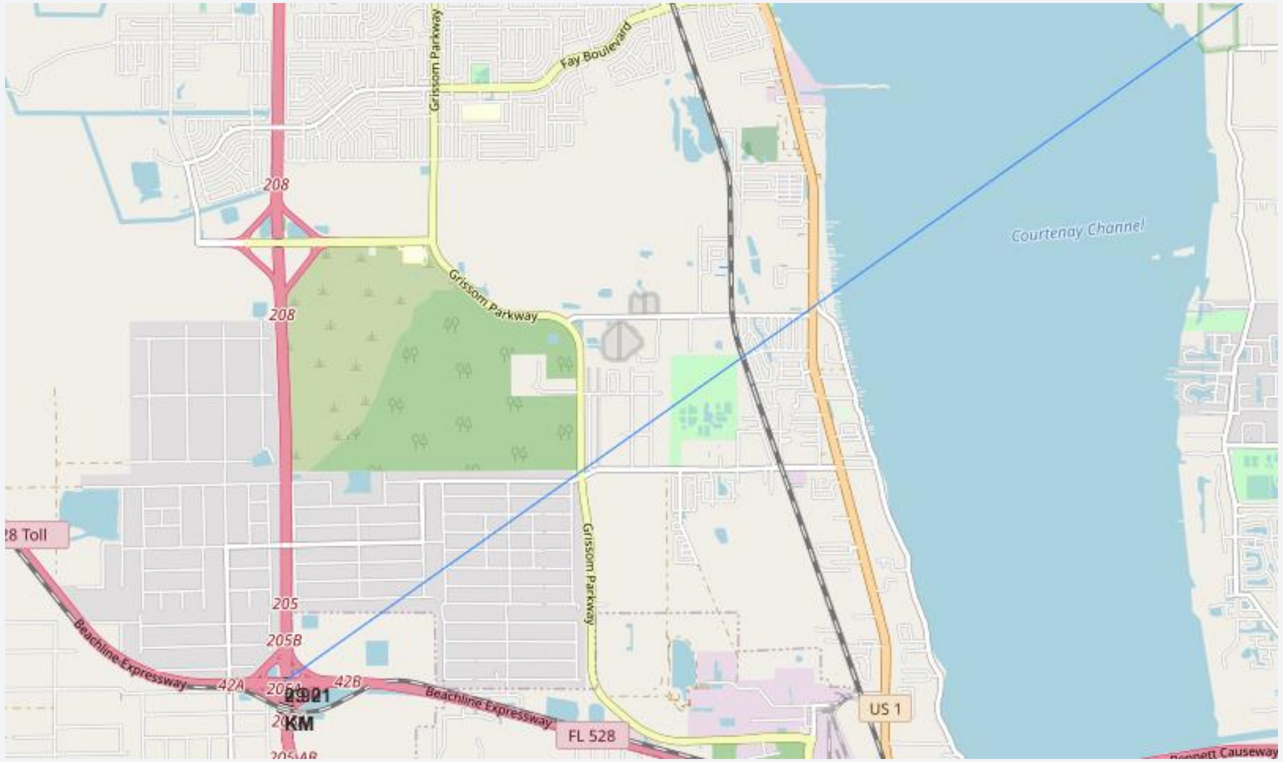
Launch Results by Location in Launch Site



- As we can see, at this Launch Site there were a few failures until the company found the “sweet spot” to launch successfully

Distance from Launch Site to its proximities



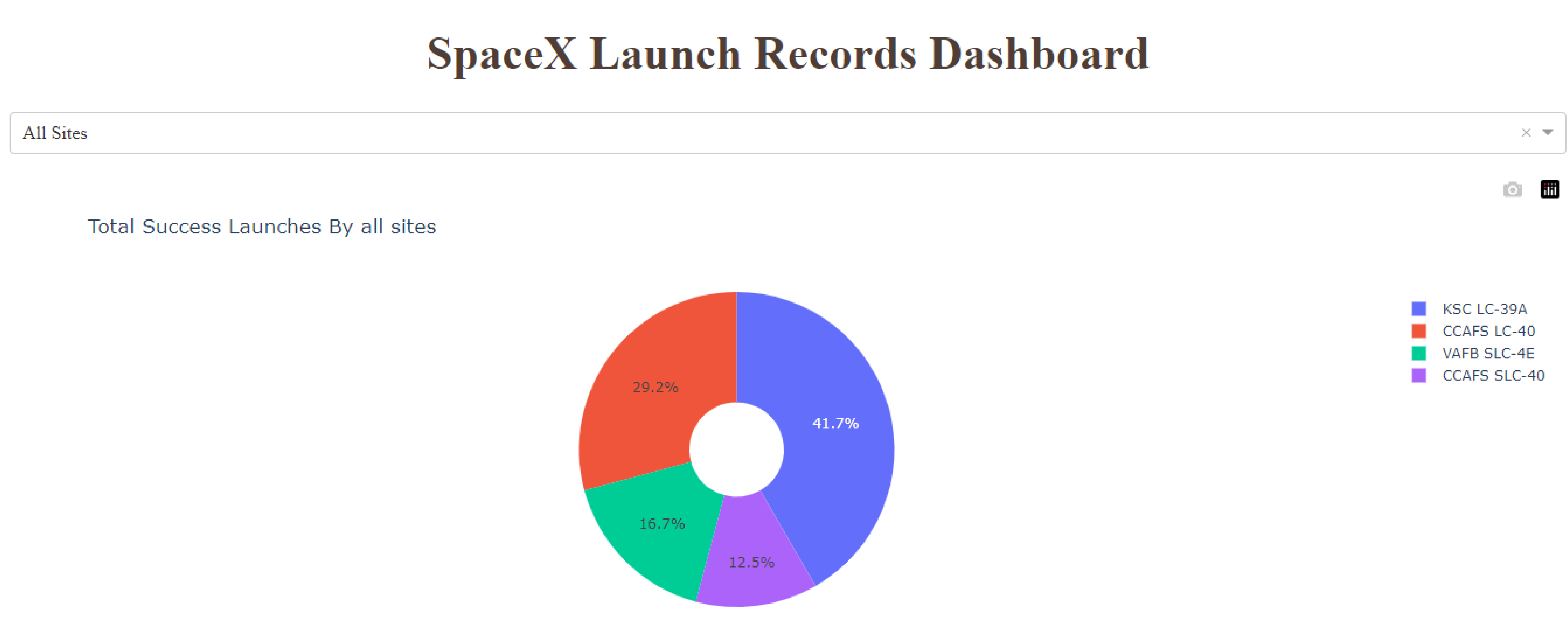


- Launch Sites need to be near the shoreline and away from highly populated areas like highways for safety reasons

Section 4

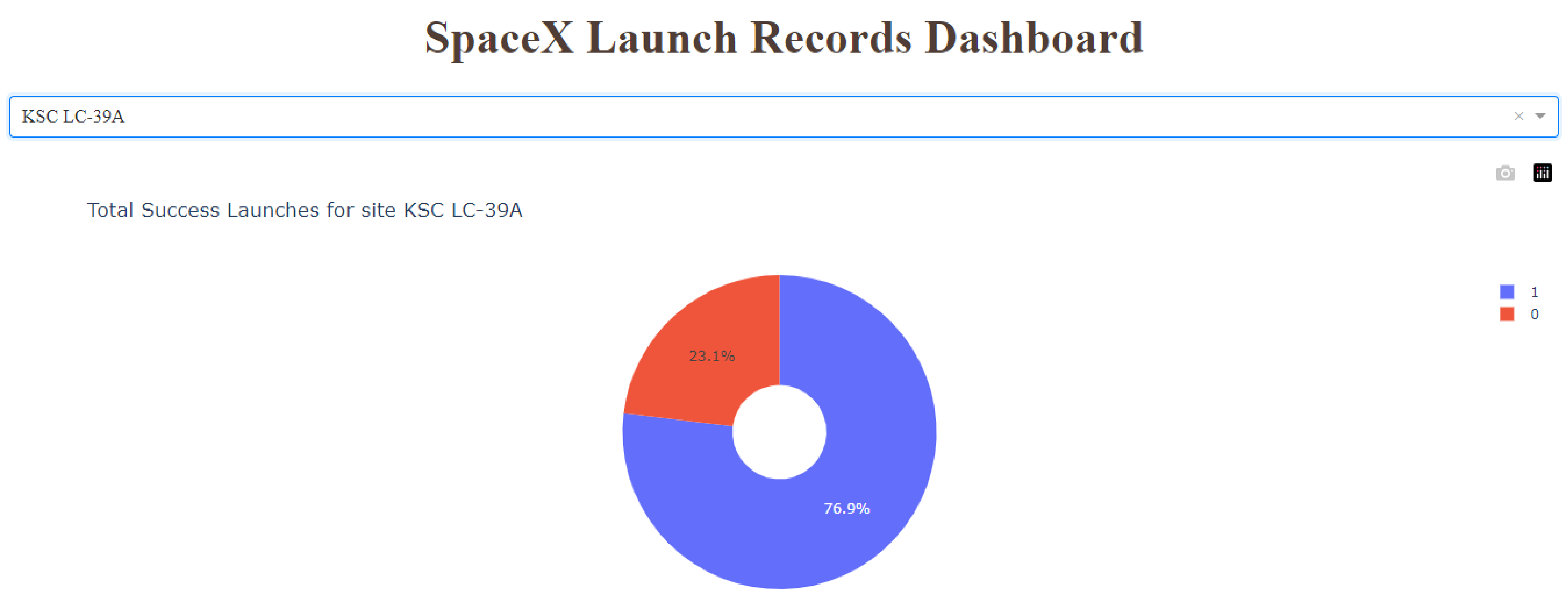
Build a Dashboard with Plotly Dash

SpaceX Launch Records by Launch Site



- KSC has the highest number of successful launches while CCAFS has the lowest

Most successful Launch Site



- 76.9% of the launches at KSC have been successful

Success Rate of Booster Versions with Payloads over 5,000 Kg

- Very few (3) Booster Versions can handle successfully a payload over 5,000 Kg

Section 5

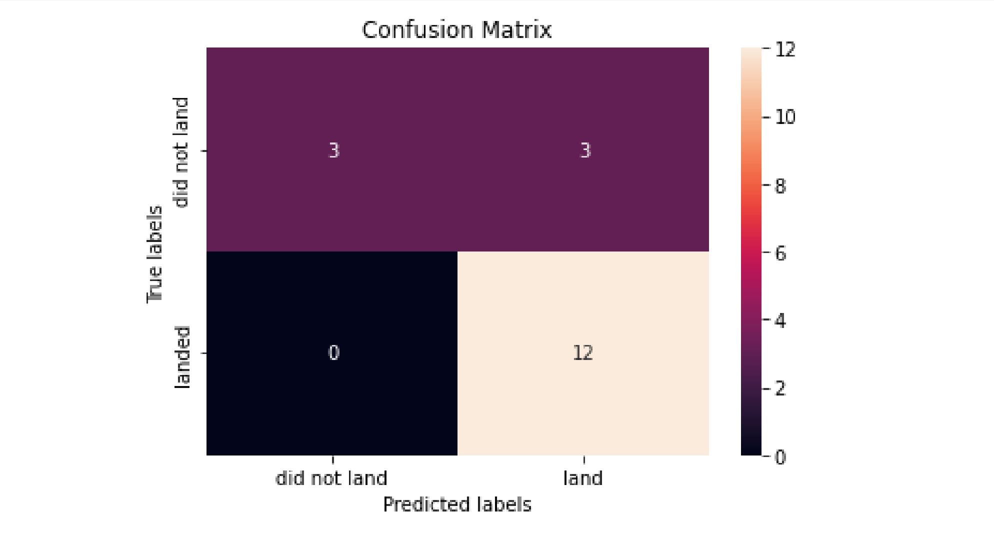
Predictive Analysis (Classification)

Classification Accuracy

```
Best model is DecisionTree with a score of 0.8767857142857143
Best params is : {'criterion': 'gini', 'max_depth': 18, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 2, 'splitter': 'random'}
```

- Decision Tree has the highest score with 0.8767

Confusion Matrix



- The model works great to determine a successful landing but struggles to identify 3 cases where the mission didn't land

Conclusions

- The company has had a positive trend in success rate from 2013 to 2020
- The most successful Orbits are ES-L1, GEO, HEO and SSO
- KSC Launch Site has the most successful missions
- With the newer versions of the Boosters we can have mission with over 5,000 Kg of Payload Mass
- Launch Site locations should be far from populated areas and near shorelines
- With the information given we can predict a successful launch with a confidence of 87.67%

Thank you!