IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ben Lehmann
8/6/2023

Outline

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

Executive Summary

We collect data using the SpaceX Rest API and use Web Scraping methods using BeautifulSoup4. We will wrangle data to create 1/0 or Success/Failure variables for the data. We will explore the data using Seaborn, Dash, and Plotly for visualization and SQL for statistical calculations and filtering. After this, we will create ML models (KNN,Decision Tree, Logistic Regression, SVM) to find which one is best for prediction of launch.

Results
We found out that ES-L1, GEO, HEO, and SSO have 100% success rate. KSC LC-39A is the best booster, with the best success rate. We also learned with visualization is that the more weight could mean that the launches would be more successful.

Introduction

SpaceX is the leading company in the space industry, with the goal to make space travel available to anyone. SpaceX's Falcon 9 is one the most used type of rocket. In this Final Project, I will be working with SpaceX and will be working to predict if the first stage will land. This will help the company in terms of determining the cost of each launch.

There are some problems that we need to ask

- Do factors such as Launch Site, Number of flights, Orbits make an impact success in first stage launches
- Can we find the accuracy of a landing?
- Can we find the best ML model for predicting a successful landing

Section 1

Methodology

Methodology

- Data collection methodology:
  - Data was collected using SpaceX’s Rest API and Web Scraping
- Perform data wrangling
  - Filling in missing data (NAN, Null), Filtering and Label Encoding
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly and Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models (F1, Accuracy)

Data Collection

1. Use an API Call to gather the data
2. Use JSON to convert to a dataframe using Pandas
3. Update the Data with encoding, filling out null values
4. We need only the Falcon 9, so filter out the rest of the data
5. Convert the Data to CSV File

Data Collection – SpaceX API

- https://github.com/BenLehmann12/Capstone-Project/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

1. API Call

```python
# Takes the dataset and uses the rocket column to call the API and append the data to the list
def getBoosterVersion(data):
    for x in data['rocket']:
        if x:
         response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
         BoosterVersion.append(response['name'])
```

From the `launchpad` we would like to know the name of the launch site being used, the logitude, and the latitude.

```python
# Takes the dataset and uses the launchpad column to call the API and append the data to the list
def getLaunchSite(data):
    for x in data['launchpad']:
        if x:
          response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
          Longitude.append(response['longitude'])
          Latitude.append(response['latitude'])
          LaunchSite.append(response['name'])
```

2. Convert to JSON

```python
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_
```

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

```python
response.status_code
```

```
200
```

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

```python
# 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

```python
# Get the head of the dataframe
data.head(5)
```

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

3. Update Data

```python
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the featu
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

4. Get only Falcon 9's

```python
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = launch_data[launch_data['BoosterVersion']!='Falcon 1']
data_falcon9.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0
5	8	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0
6	10	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0
7	11	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0
8	12	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0

Now that we have removed some values we should reset the FlgihtNumber column

```python
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

5. Convert to CSV

Data Collection - Scraping

- https://github.com/BenLehmann12/Capstone-Project/blob/main/jupyter-labs-webscraping.ipynb

1. Request Web Page and Create BeautifulSoup

```
# 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
```

```
200
```

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text)
```

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

```
# Use soup.title attribute
soup.title
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

2. Find and Get Column Names

```
# Use the find_all function in the BeautifulSoup object, with element
# 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.

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

3. Storing the Dictionary to parse names

```
launch_dict= dict.fromkeys(column_names)

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

# Let's initial the launch_dict with each value to be
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']=[]
```

4. Convert to Pandas Data Frame

```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
df.head(5)
```

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success/n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43

5. Convert to CSV

Data Wrangling

1. Load the Data

```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/data
df.head(10)
```

2. Find the Bad Outcomes

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

3. Convert to Binary

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

4. Find Success Outcome

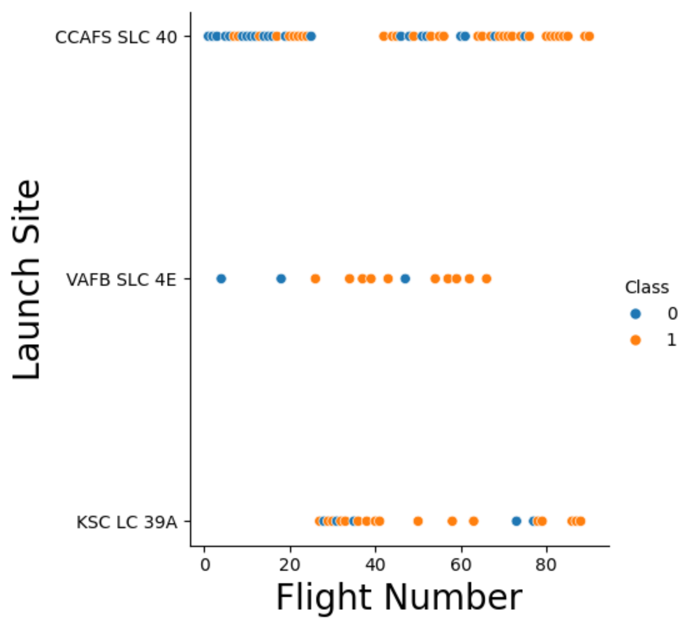
```
df["Class"].mean()
```

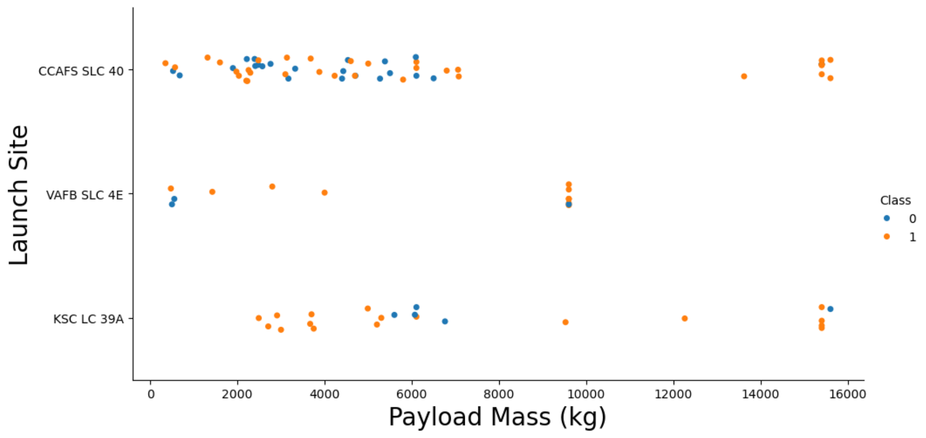
```
0.6666666666666666
```

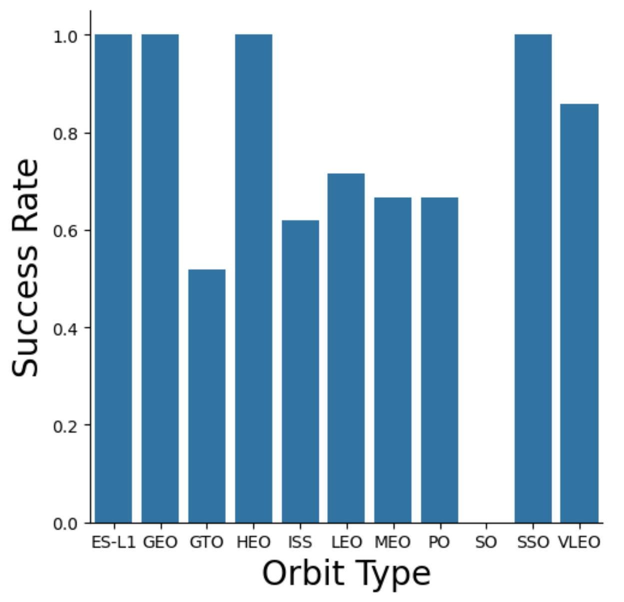
https://github.com/BenLehmann12/Capstone-Project/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

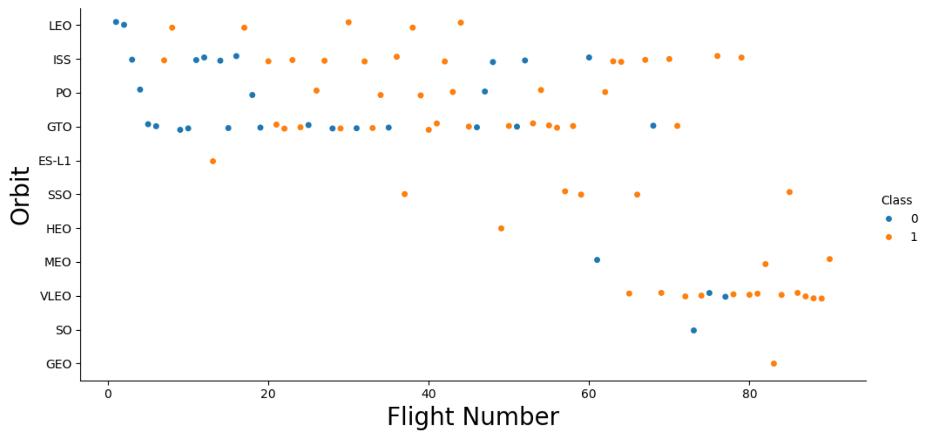
EDA with Data Visualization

https://github.com/BenLehmann12/Capstone-Project/blob/main/edadataviz.ipynb









EDA with SQL

- Show the names of the unique launch sites
- Show the 5 records where the launch site begins with 'CAA'
- Give the Total Payload Mass Carried by boosters that were launched by NASA
- Give the Average Payload Mass Carried by version F9
- Find the Data of 1st Successful landing on the pad
- Names of boosters which had success landing on drone ship and have payload mass greater than 4,000 but less than 6,000
- Total Number of Successful and Failed Missions
- Names of Booster versions that have carried max payload

https://github.com/BenLehmann12/Capstone-Project/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Used Colored Markers to indicate if the launch was a success or a failure
  - Green means Success and Red means Failure
- Used Circles to indicate a Launch Site
- Add a blue line to show the Distance between the Site CCAFS SLC40 and the nearest coastline, city or some highway

https://github.com/BenLehmann12/Capstone-Project/blob/main/Site%20Location%20Graphing.ipynb

Build a Dashboard with Plotly Dash

- We have a dropdown with a list of Launch Sites
- We have Pie Charts showing the proportion of successful launches by each type
- A Scatter Plot of Payload mass vs Success rate by Booster type

- https://github.com/BenLehmann12/Capstone-Project/blob/main/SpaceXDash.ipynb

Predictive Analysis (Classification)

- We Create an Array of Data
- Standardize the Data
- Split the data with train_test_split
- Use GridSearchCV to find the best parameters for Logistic Regression, KNN, Decision Tree, SVM
- Get the Accuracy, F1 score
- Get the Confusion Matrix

- https://github.com/BenLehmann12/Capstone-Project/blob/main/SpaceXMachineLearning.ipynb

Results

- When you keep increasing the weight, the success rate increases
- KSC LC-39A is the best booster, with the best success rate
- ES-L1, GEO, HEO, and SSO have 100% success rate
- In terms of ML Model, they all performed nearly the same, but Decision tree was the model to use

Section 2

Insights drawn from EDA

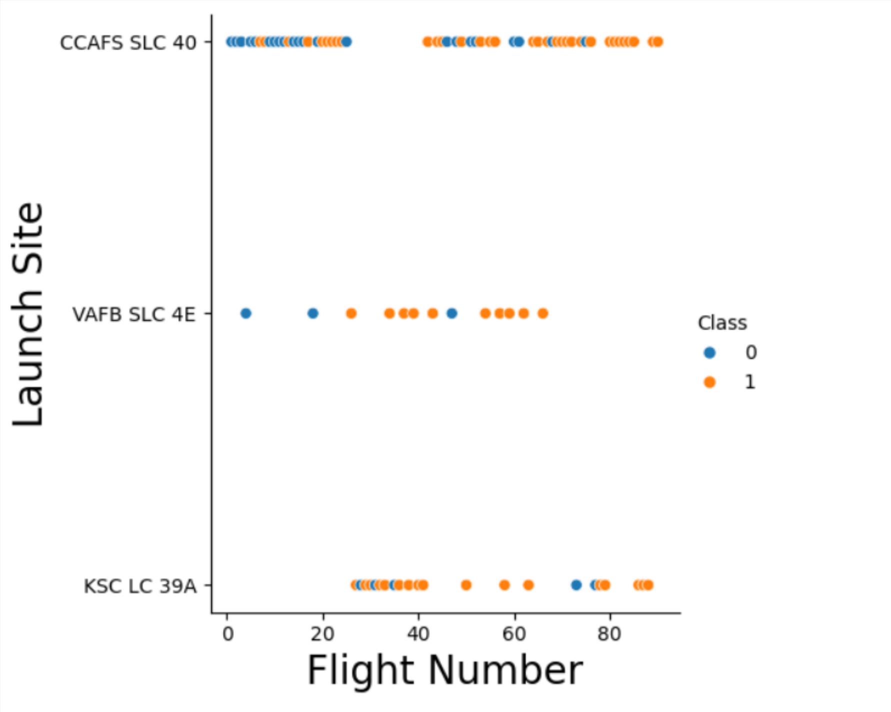
Flight Number vs. Launch Site

There were more failures in the earlier flights, but the success rate improved at the number of flights increased

There were more launches at the CCAFS SLC 40

Blue = 0 = Failure

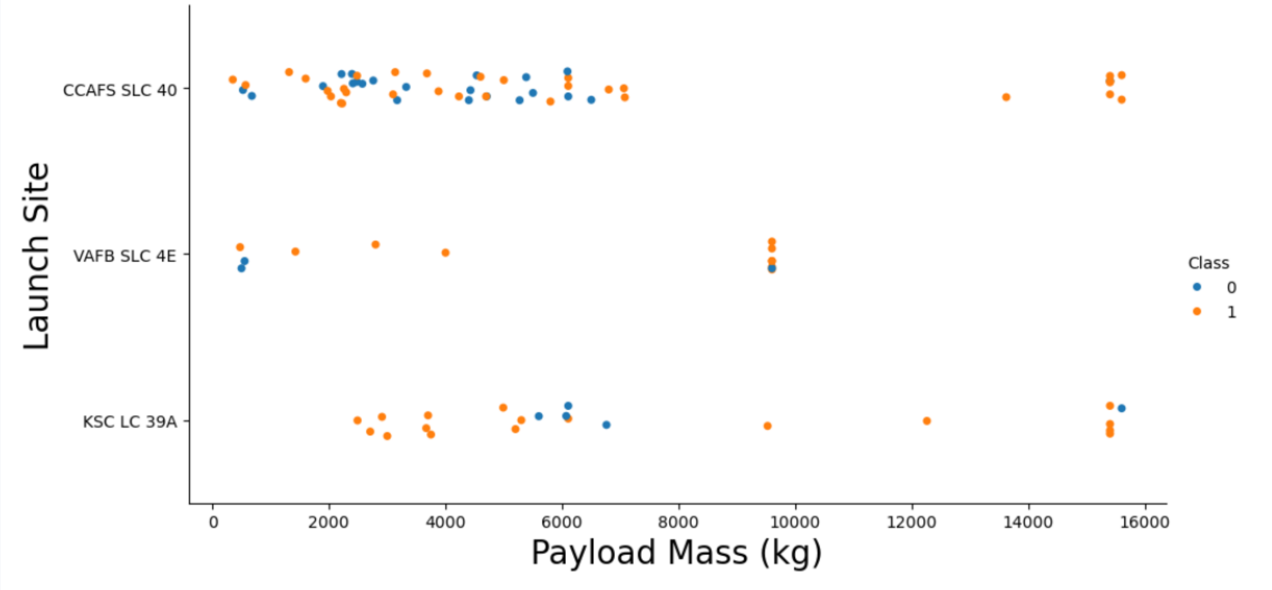
Orange = 1 = Success



Payload vs. Launch Site

One thing that can be seen is that as the weight increases, we see a higher success rate, so we can say the more weight = more success

Any Payload mass over 8,000 KG would have a much more improved Success Rate

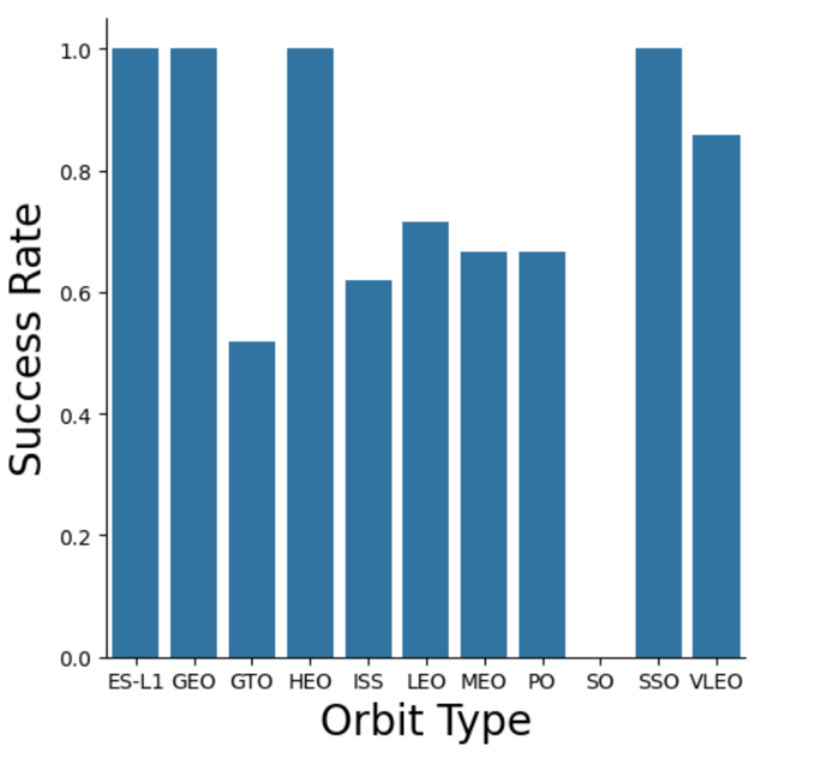


Success Rate vs. Orbit Type

Types ES-L1, GEO,HEO,SSO all had 100% Success Rate

Types GTO, ISS, LEO,MEO,PO all had a range between 40 to 80% of Success Rate

Type SO has 0% Success Rate

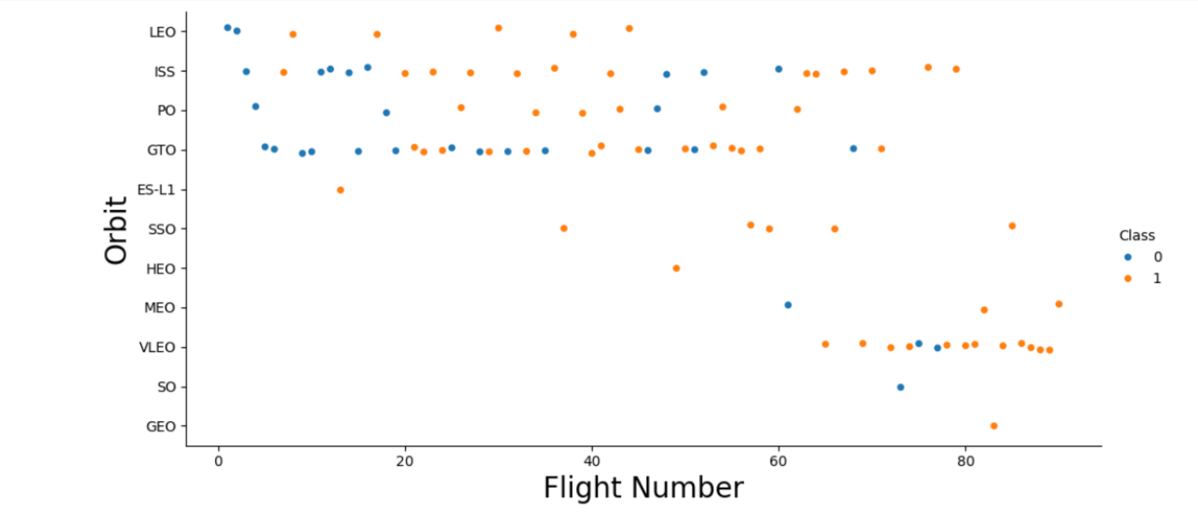


Flight Number vs. Orbit Type

For One Type, LEO, we see failure at the smaller number of flights, but the Success keeps growing at the number of flights increases

The GTO and ISS is a mixed bag, we see failures even in the growing number of flights

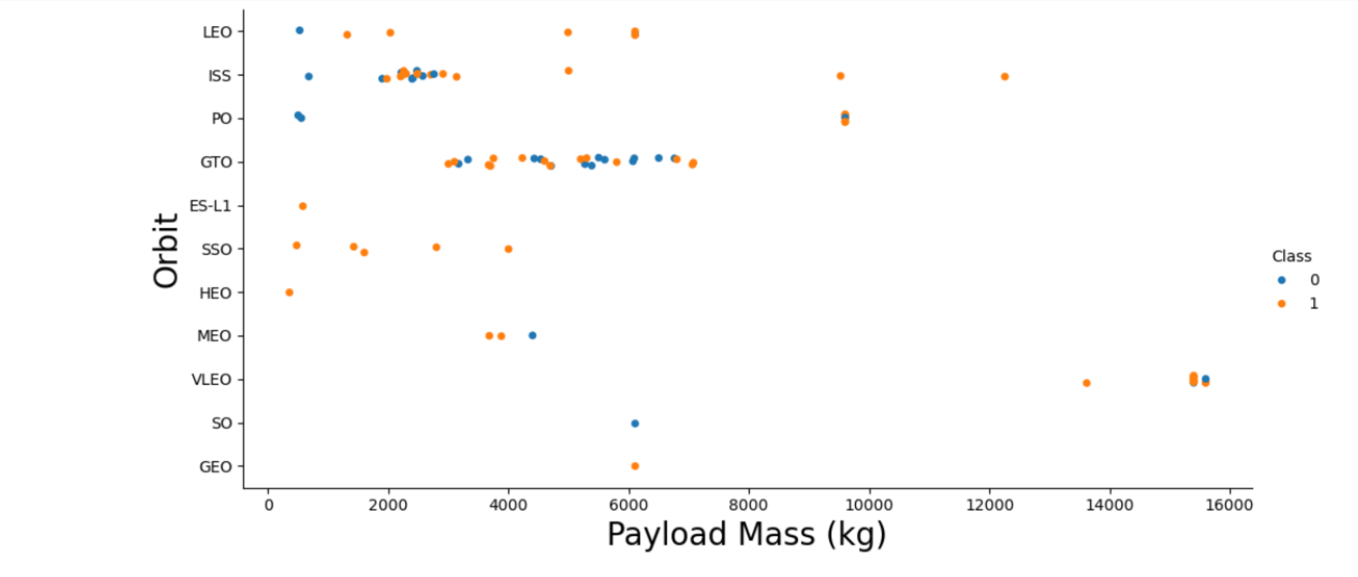
We can still say that there is more success in the more number of flights



Payload vs. Orbit Type

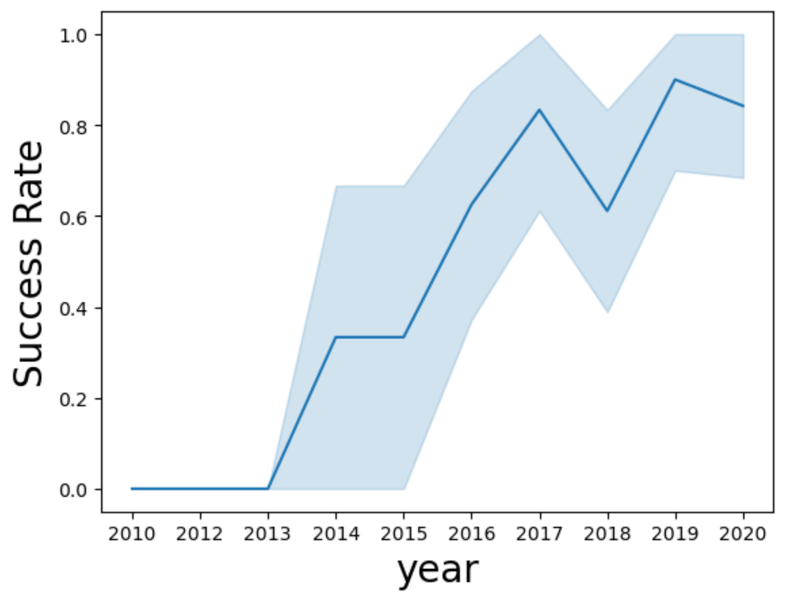
We can see many of the data points for GTO are more compressed towards each other, so we have a reason to believe that the successes are mixed with payload mass.

ISS did a better job than GTO in terms of Success and Weight.



Launch Success Yearly Trend

- The Success Rate increases by each year
- 2017-2018 had a sharp decrease
- 2018-2019 Had a Jump
- 2019-2020 went down again



All Launch Site Names

The Unique Sites include CCAFS LC-40, VAFB SLC-4E, KSC LC-39A and CCAFS SLC-40

```
[16]: %sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL;

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

[16]:

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

Launch Site Names Begin with 'CCA'

- The First Launch Sits Orbit LEO, only 1 is SpaceX and the rest is NASA

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

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

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

Total Payload Mass

- The total Payload is 45596 KG

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

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- The Average Payload Mass Carried is 2928.4 kg

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

[26]:
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1';
```

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

[26]:

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- The First Successful Ground Landing date was on 12/22/2015

```
[27]: %sql SELECT min(date) from SPACEXTBL where landing_outcome = 'Success (ground pad)'

 * sqlite:///my_data1.db
Done.

[27]:
```

min(date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- There are 5 boosters that ship landing with payload that is between 4000 to 6000 kg

```
[28]: %sql select BOOSTER_VERSION from SPACEXTBL where landing_outcome='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000
 * sqlite:///my_data1.db
Done.
```

[28]:

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

Total Number of Successful and Failure Mission Outcomes

There are 100 success and only 1 failure

[17]:
```
%%sql SELECT (SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Success%') AS SUCCESS,
(SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Failure%') AS FAILURE
```

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

[17]:

SUCCESS	FAILURE
100	1

Boosters Carried Maximum Payload

There are 12 boosters with boosters that carried maximum payload.

[35]:
```
%sql select BOOSTER_VERSION as boosterversion from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
```

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

[35]:

boosterversion
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

- I could not get this one

```
61]: ATE,BOOSTER_VERSION, LAUNCH_SITE, (LANDING_OUTCOME) FROM SPACEXTBL WHERE (LANDING_OUTCOME) = 'Failure(drone ship)'and substr(DATE,7,4) = '2015'
```

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

61]:

month	Date	Booster_Version	Launch_Site	Landing_Outcome

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

- No Attempts had 10 launches while Percludes (Drone Ship) had only 1 Launch

```
[37]: UNT(*) AS COUNT_LAUNCHES FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY LANDING_OUTCOME ORDER BY COUNT_LAUNCHES DESC;
 * sqlite:///my_data1.db
Done.
```

[37]:

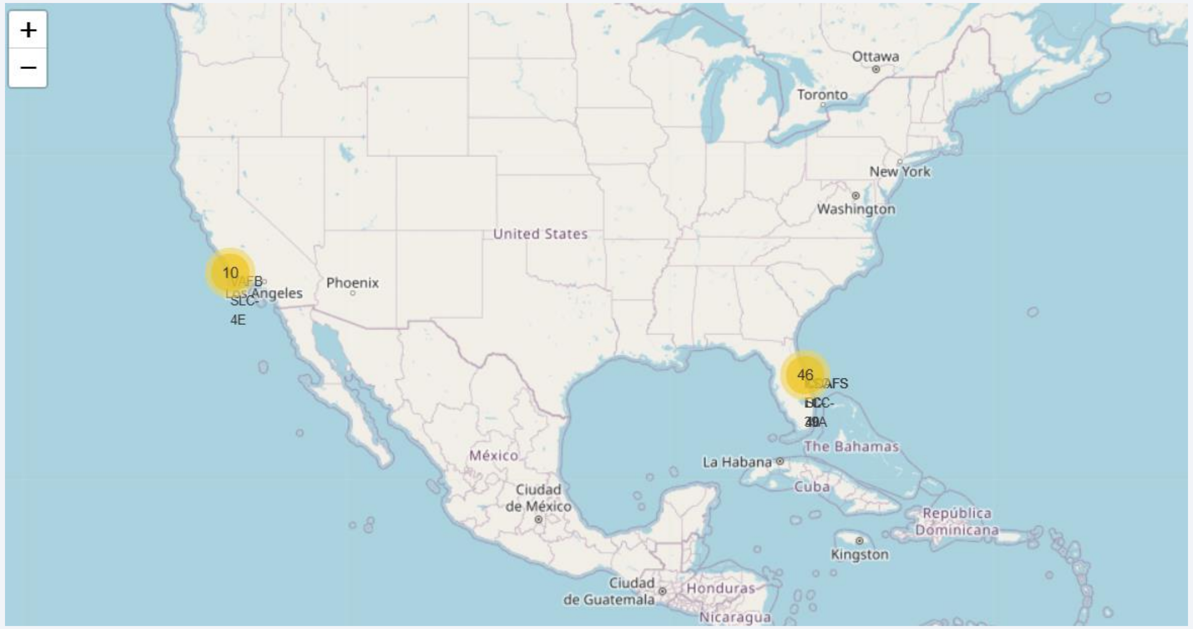
Landing_Outcome	COUNT_LAUNCHES
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

All the Launch Sites

- https://github.com/BenLehmann12/Capstone-Project/blob/main/Site%20Location%20Graphing.ipynb
- All the launch sites are near the equator, has to do with rotation, so being near the equator is the best for launching

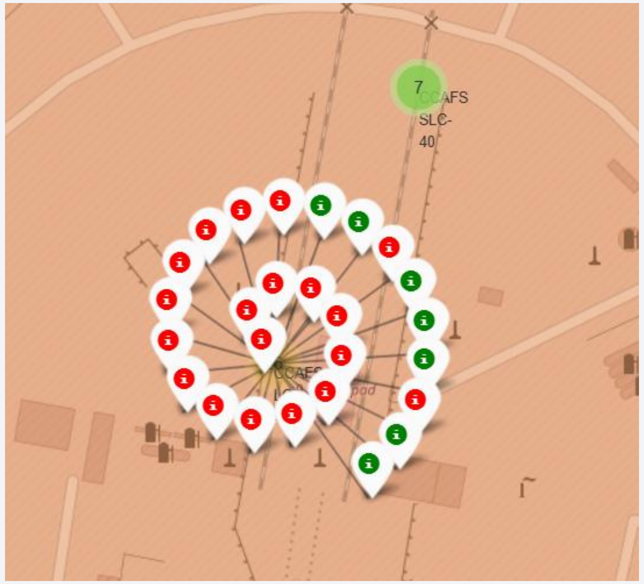


Site Failures and Successes

Green = Success, Red = Failure

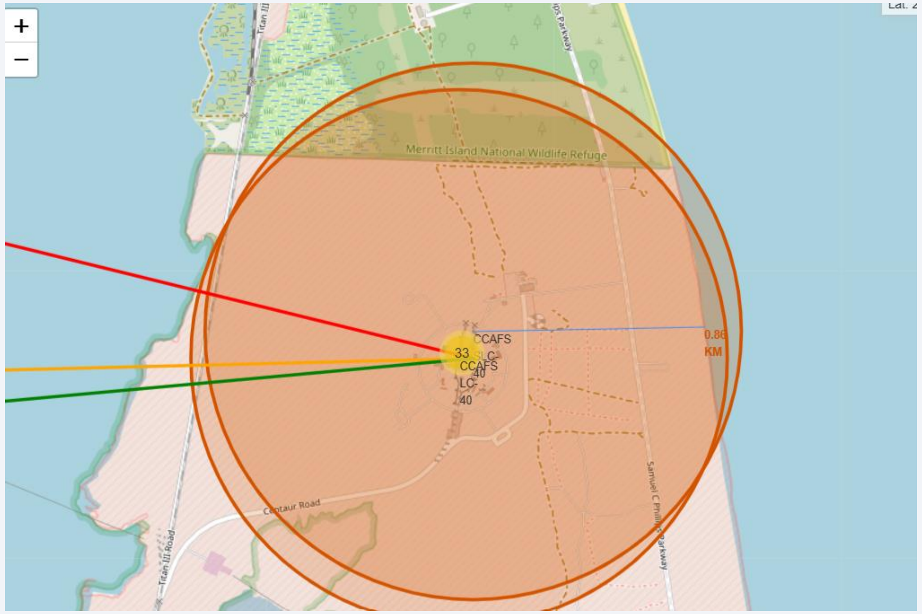
In one area of CCAFS SLC-40, there are 7 launches, only 3 are successful

In the other area there are 26, only 7 are successful



Distances

- CCAFS SLC-40
- Only 0.86 km to the shoreline
- 23.23 to the closest city

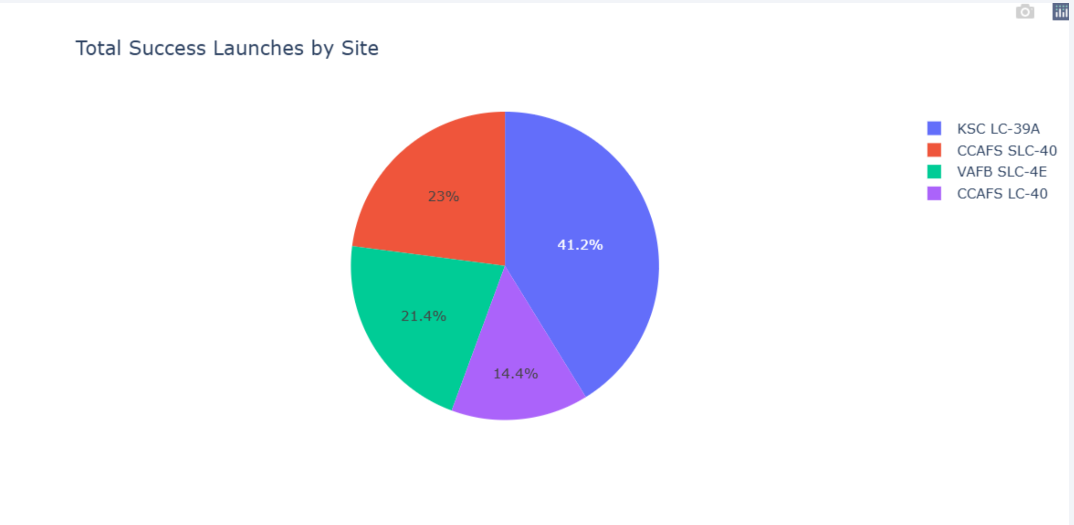


Section 4

Build a Dashboard with Plotly Dash

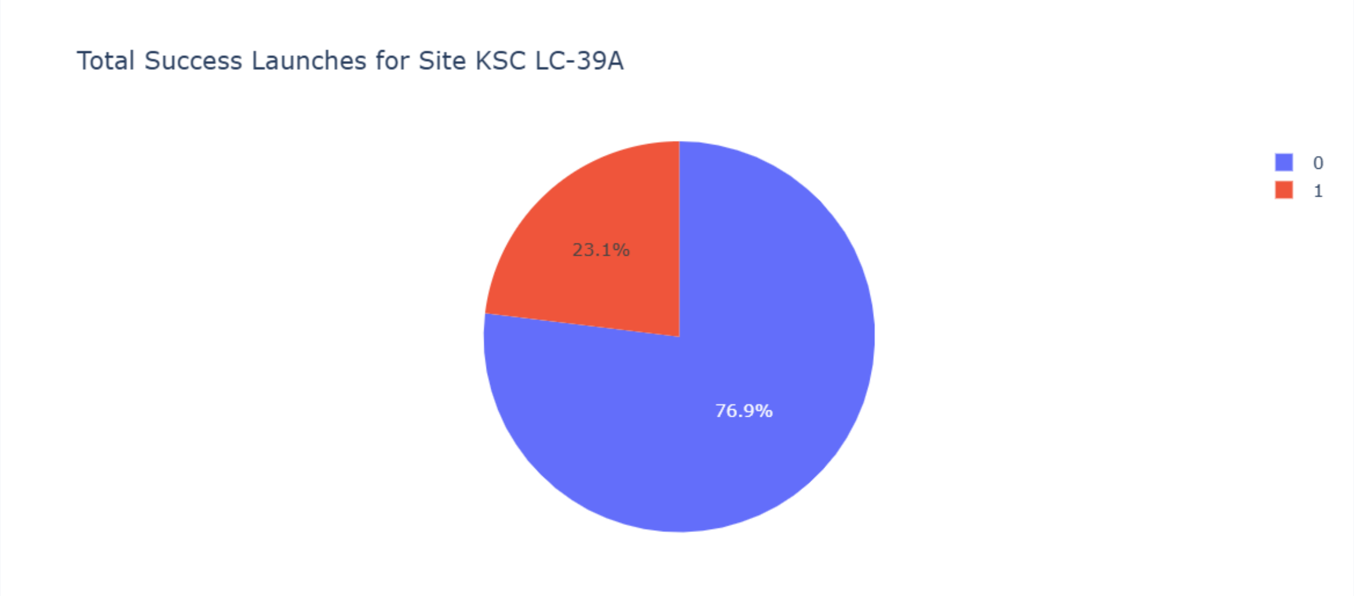
Percentage of Success

- https://github.com/BenLehmann12/Capstone-Project/blob/main/SpaceXDash.ipynb
- KSC-LC-39A had the best success



Highest Launch Success

- Over 76.9% of KSC LC-39A's Launches were successful, only 23.1% were Failures



Correlation Payload and Success

- Payloads between 2000 to 4000 kg have the best success rate

Section 5

Predictive Analysis (Classification)

Classification Accuracy

All the Models tended to perform the best. But it could be determined that the Decision tree would be the best model to use.

We can use the .best_score to find the best score of the model

	LogReg	SVM	Tree	KNN
Jaccard_Score	0.800000	0.800000	0.800000	0.800000
F1_Score	0.888889	0.888889	0.888889	0.888889
Accuracy	0.833333	0.833333	0.833333	0.833333

Confusion Matrix

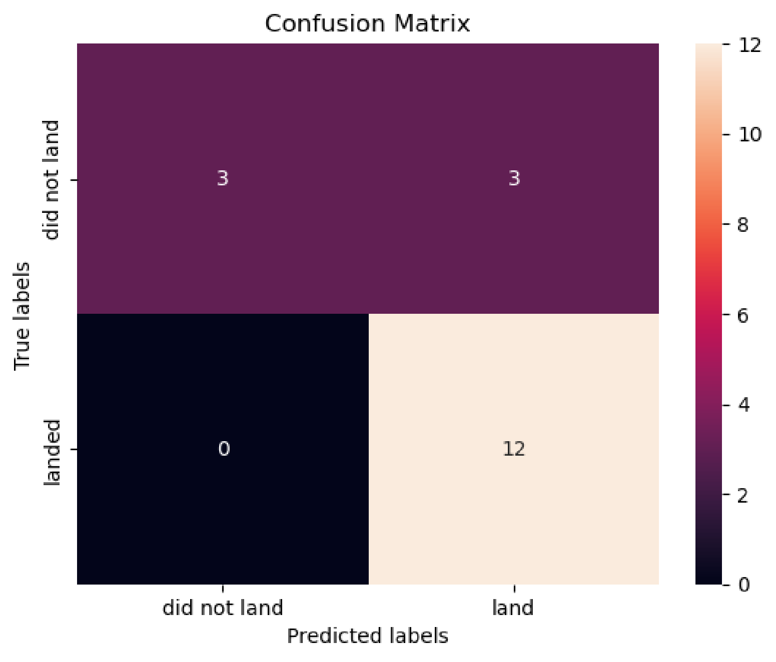
The Confusion Matrix of the Decision Tree

Precision = 80%

Recall = 100%

F1 Score = 89%

Accuracy = 83%



Conclusions

- Models: All the Models performed equally, but the Decision Tree is the way to go.
- Best Orbits: ES-L1, GEO, HEO, SSO had the best success rate with 100%
- Mass: The Larger the Payload Mass, the higher the success rate
- Launch: The Success rate for the Launch increased over each time

Thank you!