Winning the space race with Data Science

Peter Soosalu
18 November, 2022

Section 1


Executive Summary
Introduction
Methodology
Results
Conclusion
Appendix

Executive Summary

- The following methods were used during this analysis
  - Data collection (webscrape & API)
  - Data wrangling
  - EDA with SQL
  - EDA (Matplotlib, Seaborn)
  - Interactive maps with Folium
  - Dash Plotly Dashboard
  - Predictive Analysis using Alogrithms
- Summary of all results
  - Exploratory results
  - Interactive analytics
  - Predictive Analysis

Introduction

- Space travel is becoming more common, and more affordable – SpaceX advertises that it can launch a rocket for $62m, considerably lower than its competitors because it reuses the first section of its rocket. To financially compete with SpaceX, successfully returning the rocket is crucial, and with some simple analysis of past successful launches, we want to guide a competing company toward launching rockets into space.
- The aim is to cross-examine multiple aspects of SpaceX's Falcon 9 rocket to determine what the final projected cost will be for Space Y.

Methodology

- Data collection methodology:
  - SpaceX Rest API
  - Webscraping from Wikipedia
- Perform data wrangling – cleaning & preprocessing
  - One Hot Encoding data field for further analysis, averaging and dropping columns where necessary
- Perform exploratory data analysis (EDA) using SQL requests using visualization (scatter, bar, and line graphs)
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

- Gathered SpaceX launch data using their REST API and webscraping via BeautifulSoup to give information about launches:
  - Rockets used
  - Payload delivered
  - Launch & landing specifications
  - Landing outcome
- SpaceX REST API: api.spacexdata.com/v4/
- The REST API returns a JSON object that was normalized into flat data using pandas, while the BeautifulSoup HTML returned a flat structure that could already be used alongside the API data.

Data Collection – SpaceX API

Getting response from SpaceX API

Converting to a JSON file

Cleaning the data

Assignment lists to dictionary, and then a dataframe

Filter and export to a .csv file

```
spacex = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-S
response = requests.get(spacex).json()
```

```
data = pd.json_normalize(response)
```

```
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}
```

```
getBoosterVersion(data)
BoosterVersion[0:5]
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
```

```
data_falcon9.to_csv('c:\\Users\peters\Downloads\dataset_part_1.csv', index=False)
```

Github

Data Collection - Scraping

Getting response (& check that it is correct!)

Create BeautifulSoup Object

Find the tables & create the dictionary

Append the data to keys (see Falcon9Webscrape)

Converting to Dataframe, and then to a .csv

Add the GitHub URL

```
page = requests.get(static_url)
print(page.status_code)
```

```
soup = BeautifulSoup(page.text, 'html.parser')
```

```
launch_dict= dict.fromkeys(column_names)

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

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'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

```
df = pd.DataFrame.from_dict(launch_dict)
print(df.head())

df.to_csv('C:\\Users\peters\Downloads\spacex_web_scraped.csv', index=False)
```

Github

Data Wrangling

- Several challenges were faced, including unsuccessful booster landings, bad rocket launches, or unsuccessful landings on the ground pad (for example)
- The process included calculating:
  - Perform initial EDA on the dataset
  - Number of launches
  - Number and occurrence of each orbit
  - Number and occurrence of mission outcome per orbit type
  - Create a landing outcome label from the outcome column
  - Work out success rate for every landing in the dataset

- Github

EDA with Data Visualization

- Scatter graphs drawn
  - Flight Number vs Payload Mass
  - Flight Number vs Launch Site
  - Payload vs Launch Site
  - Orbit vs Flight Number
  - Payload vs Orbit Type
  - Orbit vs Payload Mass
- Bar graph drawn
  - Mean vs Orbit
- Line graph drawn
  - Success Rate vs Year
- Github

EDA with SQL

- Several SQL queries were performed on the dataset, including:
  - Finding names of the unique launch sites
  - Displaying records where launch sites begin with the string ‘CCA’
  - Finding the total payload mass carried by booster launched by NASA
  - Average payload mass carried by booster version f9 v1.1
  - First successful landing outcome to ground pad
  - Names of successful booster in drone ship with payload mass > 4000kg & < 6000kg
  - Number of successful and failure mission outcomes
  - Names of booster version with max payload mass
  - Failed landing outcomes on drone ship with their booster versions and launch site
  - Rank of landing outcomes as success of failure from 2010-06-04 to 2017-03-20
- Github

Interactive Map with Folium

- Latitude and Longitude Coordinates from each site were added with a circle marker with a label naming the launch site
- Assigned the dataframe launch_outcomes to green and red markers on the map with MarkerCluster() to visualize where the best chance of success will be geographically
- Calculated distance using Haversine's formula to various landmarks to see if any patterns existed
- Github

Build a Dashboard with Plotly Dash

- Created an interactive dashboard with Plotly Dash to enable others to further explore the data
- Pie chart added to show the total launches of a certain site and all launch sites
- Scatter graph shows the relationship of Outcome and Payload Mass (kg) for the different booster versions to visualize booster with the best chance of success
- Github

Predictive Analysis (Classification)

- Model build
  - Load (Numpy and Pandas) and transform data
  - Split data into training and testing sets
  - Based on test & training sizes, decide which Machine Learning algorithm to use
  - Set parameters, fit the dataset, train the dataset
- Model Evaluation
  - Check model accuracy against the test set
  - Get tuned hyperparameters for each algorithm type
  - Plot confusion matrix

- Model Improvement
  - Algorithm tuning
- Finding the best-performing model
  - Most accurate model used
  - Notebook contains summary of algorithms with scores for each

- Github

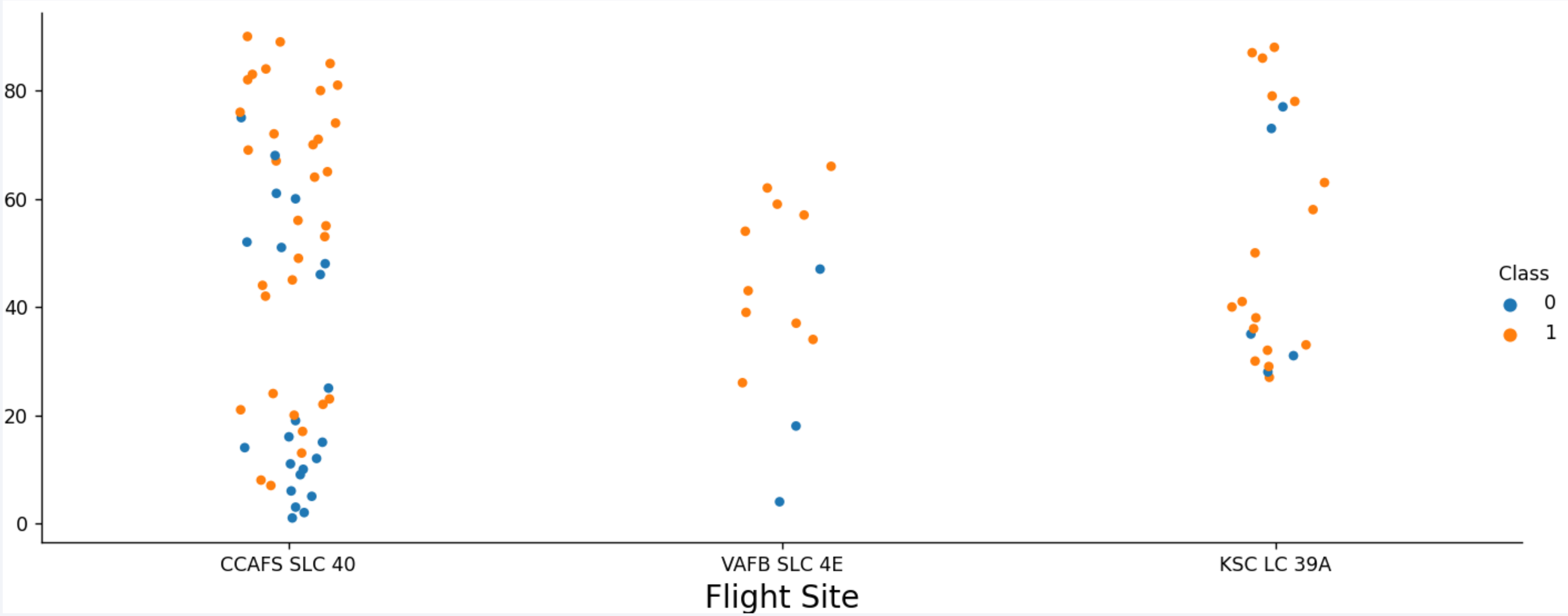
Results

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

Section 2

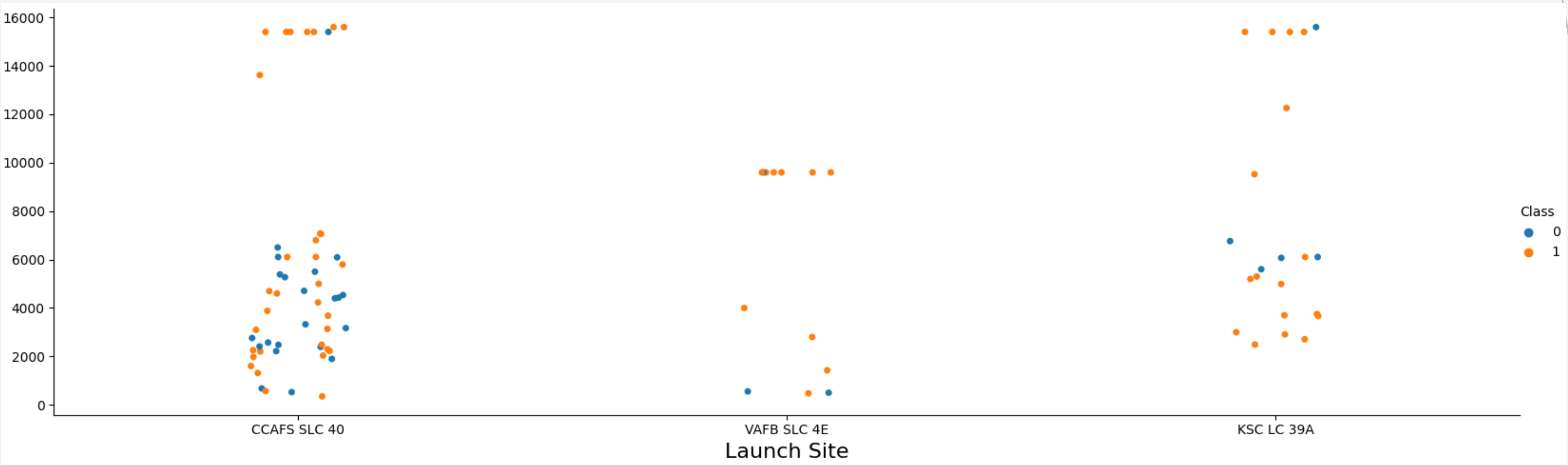
Insights Drawn from EDA

Flight Number vs. Launch Site



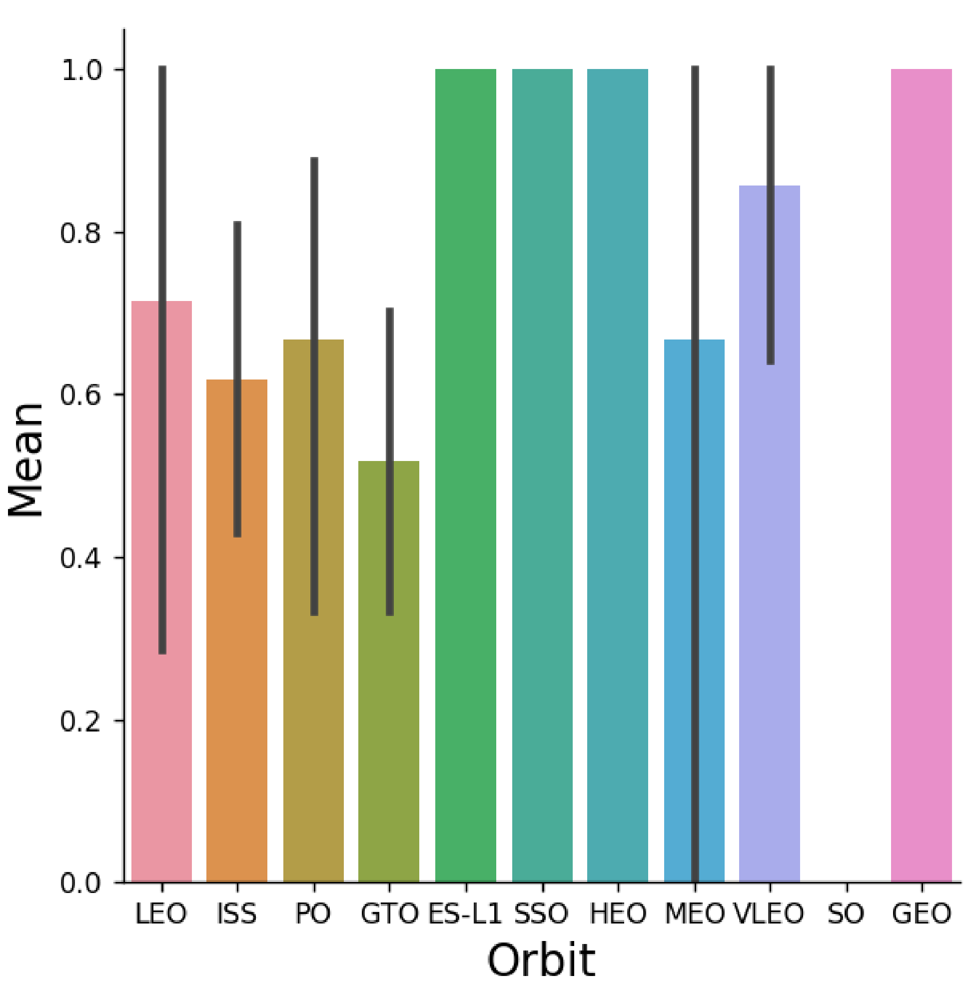
- 0 is not successful, 1 is successful
- As the flight numbers increased, the percentage of success increased
- Launch site axis label included in the source code, but hard to crop

Payload vs. Launch Site



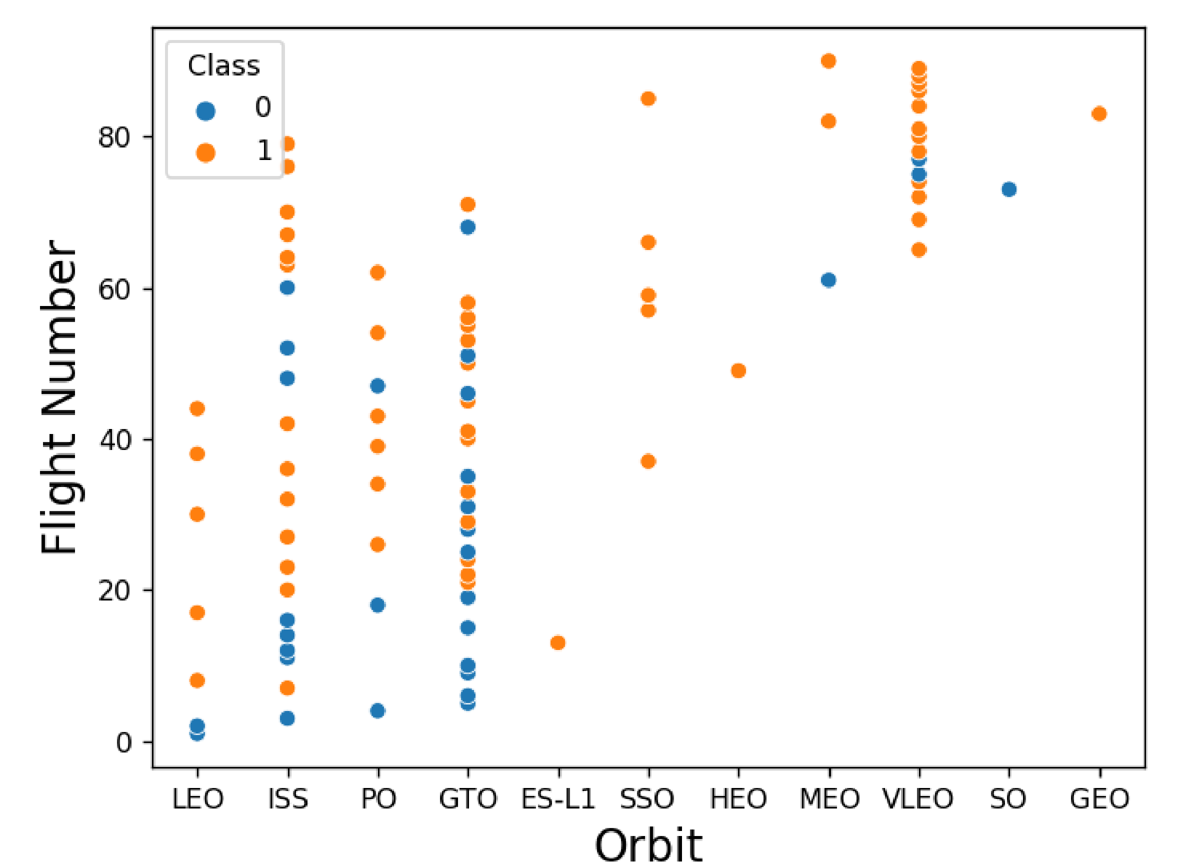
- As the payload (kg) increases, the chance of success appears to slightly increase
- Payload axis label included in the source code, but hard to crop

Success Rate vs. Orbit Type



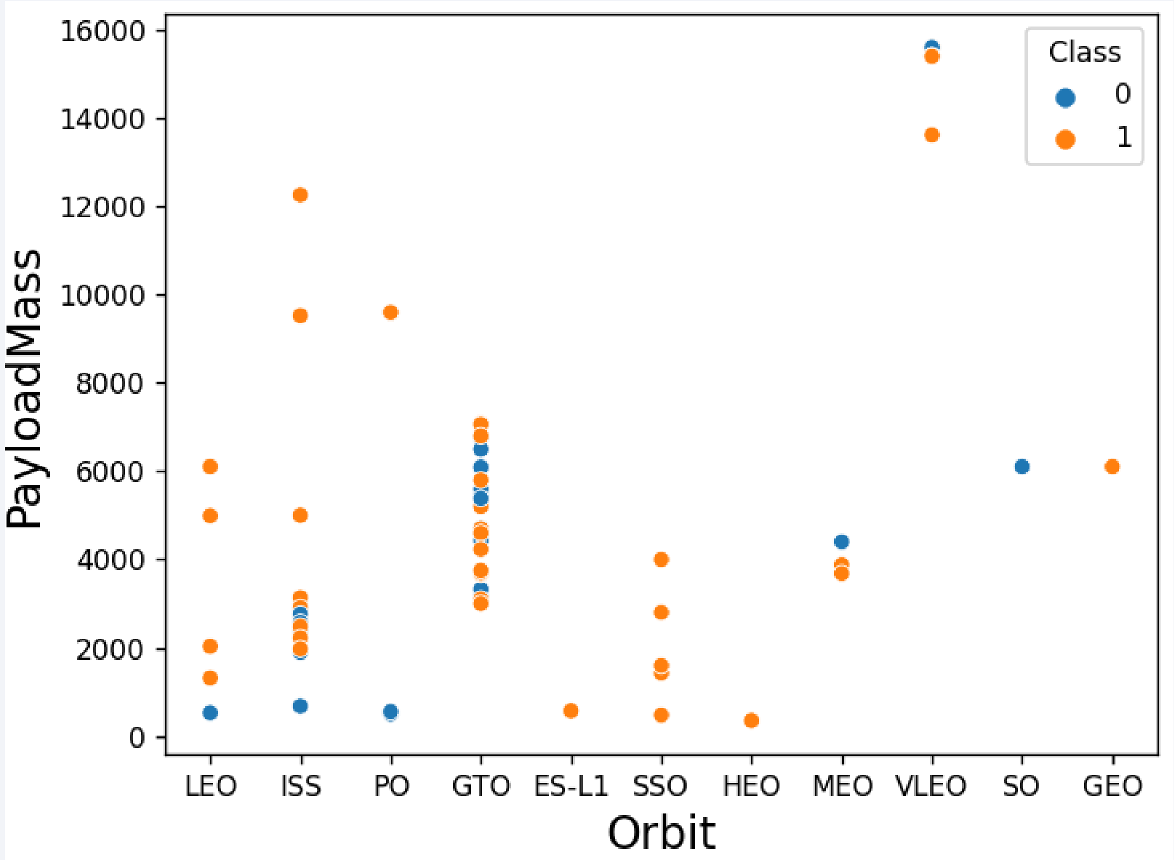
- ES-L1, SSO, HEO, and GEO had the best success rates overall
- Large deviation from MEO and LEO, with a ceiling that includes a success rate as good as rockets mentioned above

Flight Number vs. Orbit Type



- Success in LEO appears to be related to the number of flights
- Very low test sample for ES-L1, HEO, SO and GEO
- Flight number not indicative of success in VLEO, ISS, or GTO

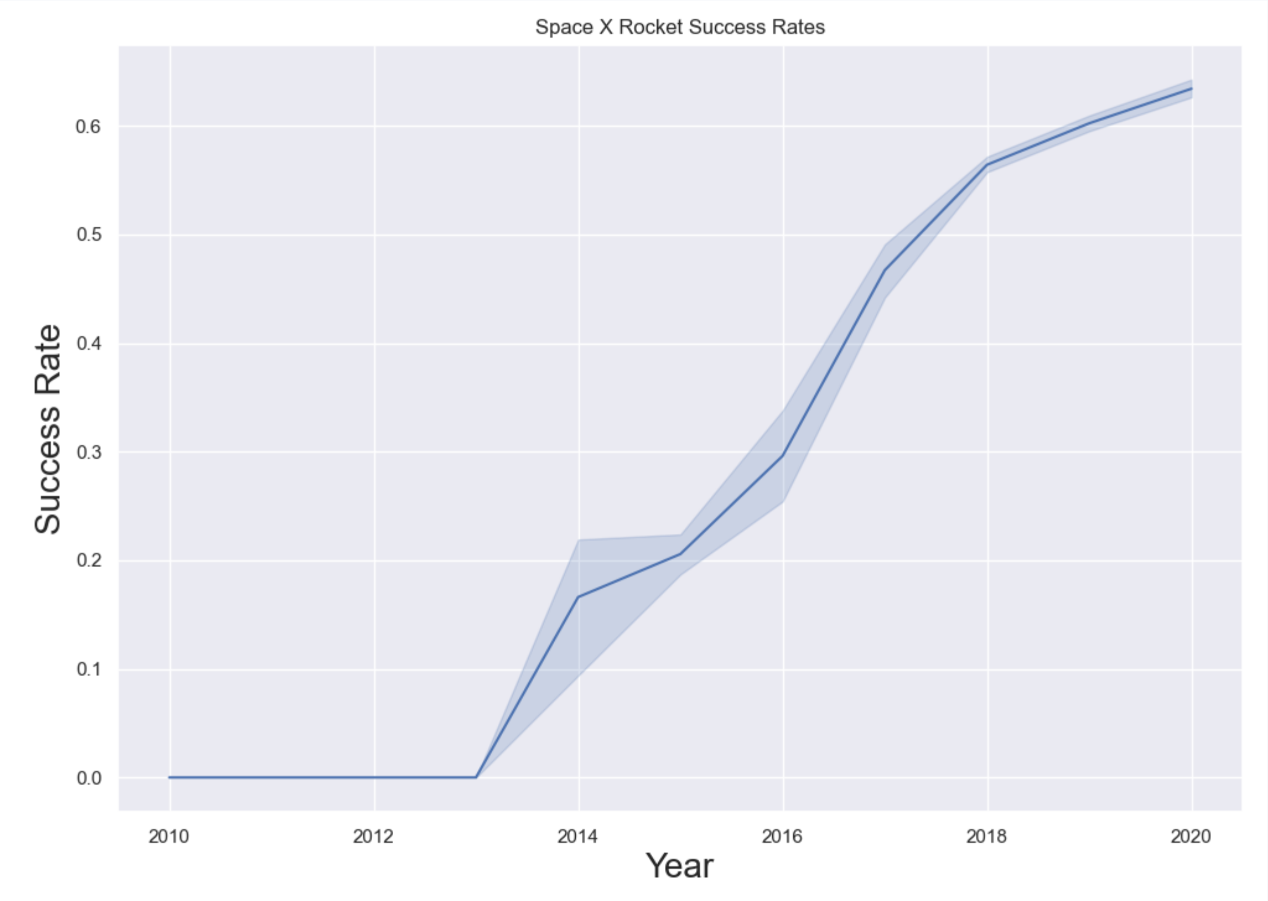
Payload vs. Orbit Type



- Light payloads have a slightly positive influence on GTO orbits

- Heavy payloads have a positive influence on LEO and ISS orbits

Launch Success Yearly Trend



- Success rate has been increasing every year

All Launch Site Names

Input

- ‘SELECT DISTINCT Launch_site FROM df’, ‘Launch_Site’

Output

- Unique launch sites
  - CCAFS LC-40
  - CCAFS SLC-40
  - KSC LC-39A
  - VAFB SLC-4E

- Using the word DISTINCT means you will only shows unique values from the Launch_Site column

Launch Site Names Begin with 'CCA'

Input

- ‘SELECT TOP 5 * FROM df WHERE Launch_Site LIKE ‘CCA%’

Output

- Using TOP 5 and LIKE keywords, alongside ‘CCA%’ means the launch site name must begin with CCA.

Total Payload Mass

Input

- "SELECT SUM(PAYLOAD_MASS_KG_) TotalPayloadMass FROM df"

Output

	Total Payload Mass
0	45596

- SUM sums the total

Average Payload Mass by F9 v1.1

Input

- "SELECT AVG(PAYLOAD_MASS_KG_) AveragePayloadMass FROM df WHERE Booster_Version = 'F9 v1.1'",'AveragePayloadMass'

Output

	Average Payload Mass
0	2928

- AVG averages the total, with a WHERE statement to only use ‘F9 v1.1’ booster information

First Successful Ground Landing Date

Input

- "SELECT MIN(Date) SLO FROM df WHERE Landing_Outcome = 'Success (drone ship)'",'SLO'

Output

	Date which first Successful landing outcome in drone ship was acheived.
0	06-05-2016

- MIN works out the minimum date value WHERE the landing outcome was successful

Successful Drone Ship Landing with 4000 < Payload > 6000

Input

- "SELECT Booster_Version FROM df WHERE Landing_Outcome = 'Success (ground pad)' AND Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 6000",'Booster_Version')"

Output

	Date which first Successful landing outcome in drone ship was acheived.
0	F9 FT B1032.1
1	F9 B4 B1040.1
2	F9 B4 B1043.1

- Only taking booster version that was successful, AND was between the minimum and maximum payload

Number of Successful & Failure Mission Outcomes

Input

- "SELECT(SELECT Count(Mission_Outcome) FROM df WHERE Mission_Outcome LIKE '%Success%') AS Successful_Mission_Outcomes,(SELECT Count(Mission_Outcome) FROM tblSpaceX where Mission_Outcome LIKE '%Failure%') AS Failure_Mission_Outcomes"

Output

	Successful_Mission_Outcomes	Failure_Mission_Outcomes
0	100	1

- Like and a subquery were required to narrow this down. A bit more tricky to get this one.

Boosters Carried Maximum Payload

Input

- "SELECT DISTINCT Booster_Version, MAX(PAYLOAD_MASS_KG_) AS [Maximum Payload Mass] FROM df GROUP BY Booster_Version ORDER BY [Maximum Payload Mass] DESC"

Output

	Booster_Version	Maximum Payload Mass
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
...	...	...
92	F9 v1.1 B1003	500
93	F9 FT B1038.1	475
94	F9 B4 B1045.1	362
95	F9 v1.0 B0003	0
96	F9 v1.0 B0004	0

97 rows × 2 columns

- Only want boosters that were the maximum payload – search for distinct boosters, narrow to max payload, and then group them up to see the completed list

2015 Launch Records

Input

- "SELECT DateName (month , DateAdd(month , MONTH (CONVERT(date,Date, 105)) , 0) - 1) as Month, Booster_Version, Launch_Site, Landing_Outcome FROM df WHERE (Landing_Outcome LIKE N'%Success%') AND YEAR(CONVERT(date,Date, 105)) = '2015'"

- Still not having 100% success with this one, but did get it closer to grabbing the 2015 records.

2015 Launch Records

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

- Present your query result with a short explanation here

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

Input

- "SELECT COUNT(Landing_Outcome) AS sl FROM df WHERE (Landing_Outcome LIKE '%Success%') AND (Date >'04-06-2010') AND (Date < '20-03-2017')",'sl'

Output

	Successful Landing Outcomes Between 2010-06-04 and 2017-03-20
0	34

- COUNT the records, and filter the data using a WHERE statement, and use AND and LIKE to further filter

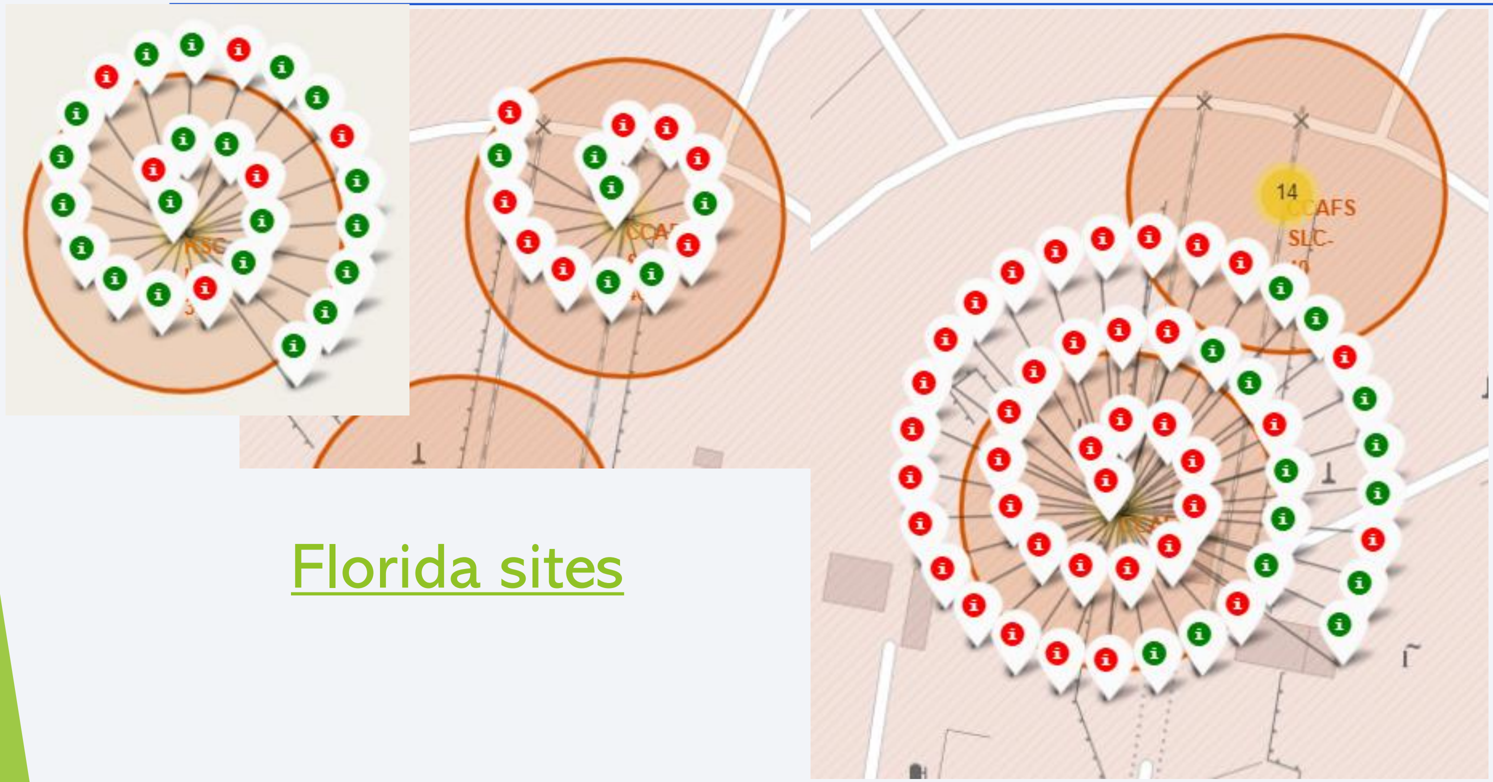
Section 3

Launch Sites Proximity Analysis

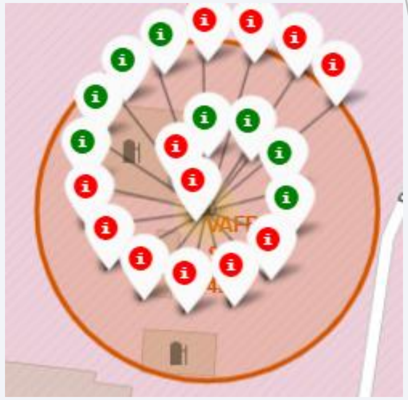
All launch sites' location

- You can see the launch sites are near the coast, located in the southern United States

California and Florida launch sites



Florida sites



California site

- Success is indicated in green, and red indicates a failed launch

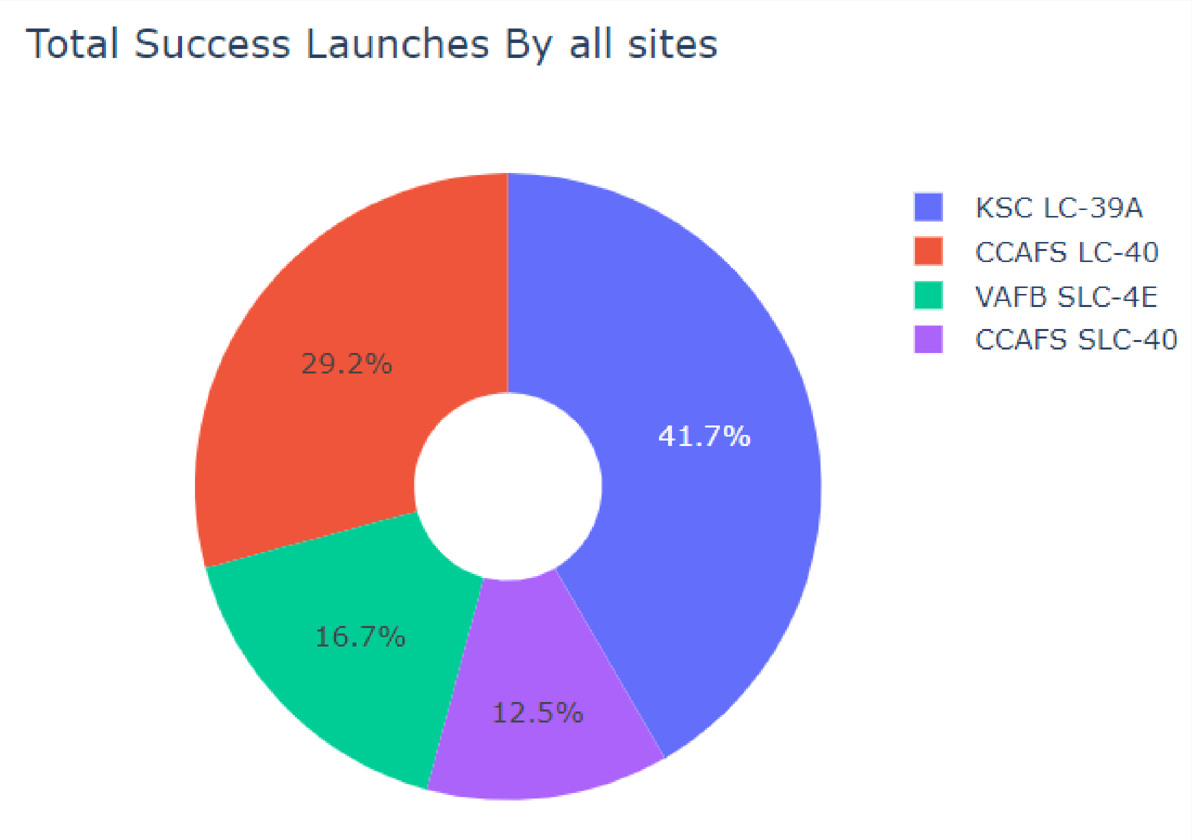
Proximity to points of interest

- The launch sites are quite close to the coastline, but are far away from major highways, cities, and railway stations

Section 4

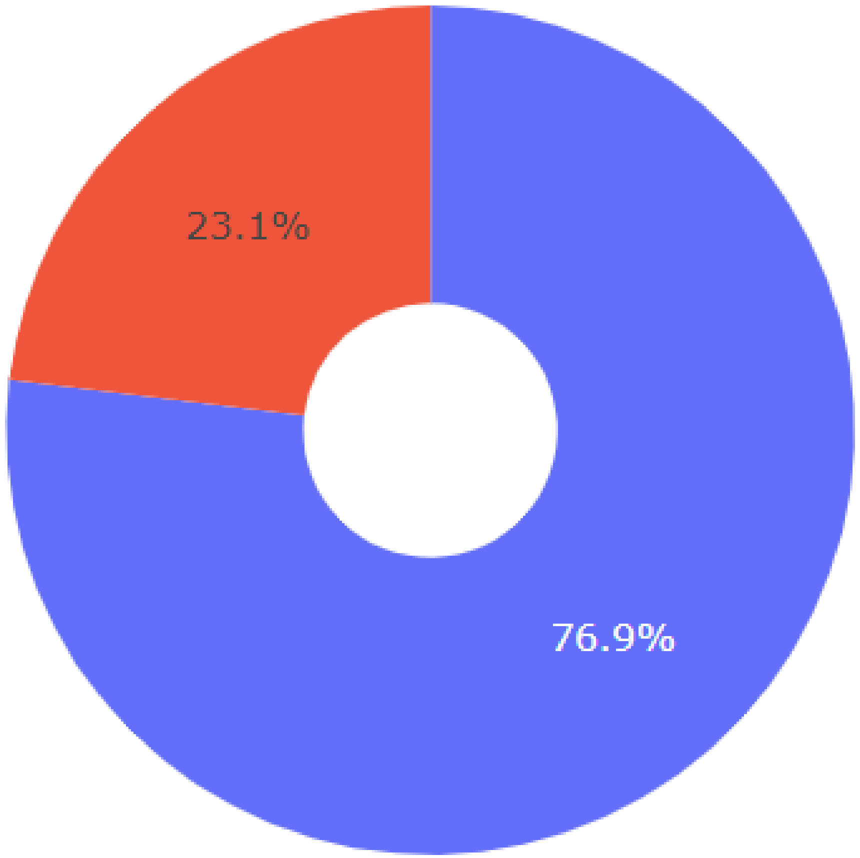
Plotly Dash Dashboard

All sites – Successful Launches



- KSC LC-39A had the most success launches across all sites

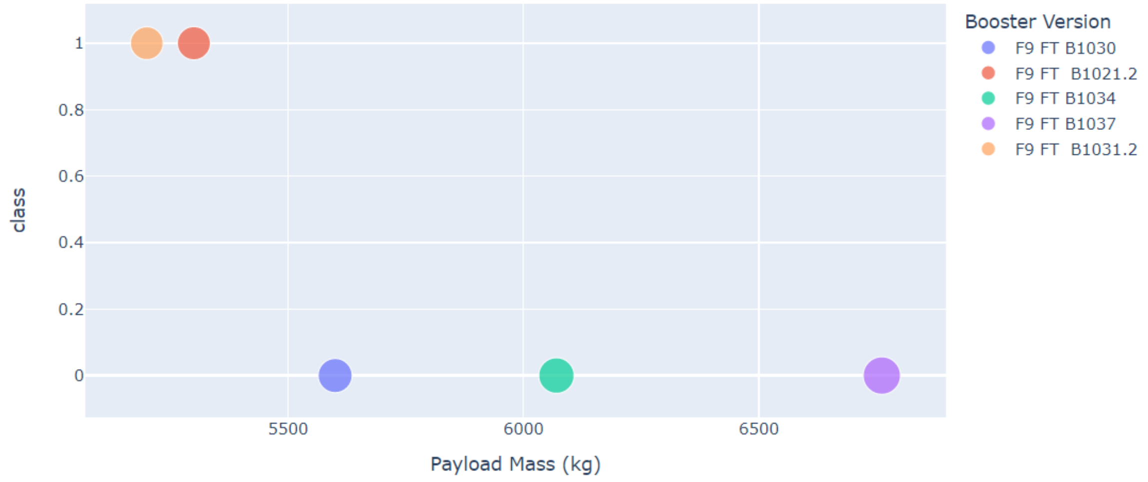
KSC LC-39A

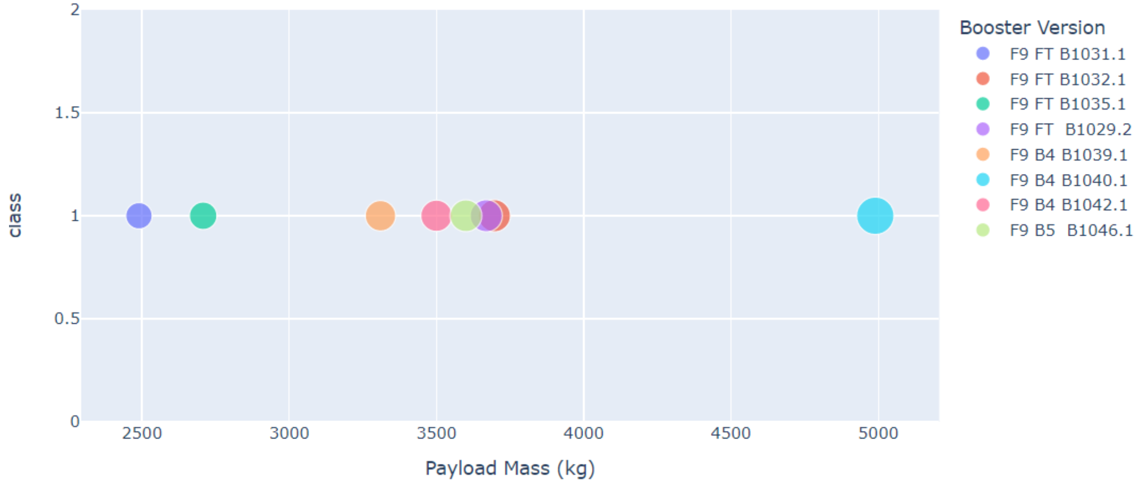


- Success rate of the KSC LC-39A site was 76.9%, and a failure rate of 23.1%

Payload vs launch outcome across all weights

- Success rate for low payloads is better than high weighted payloads

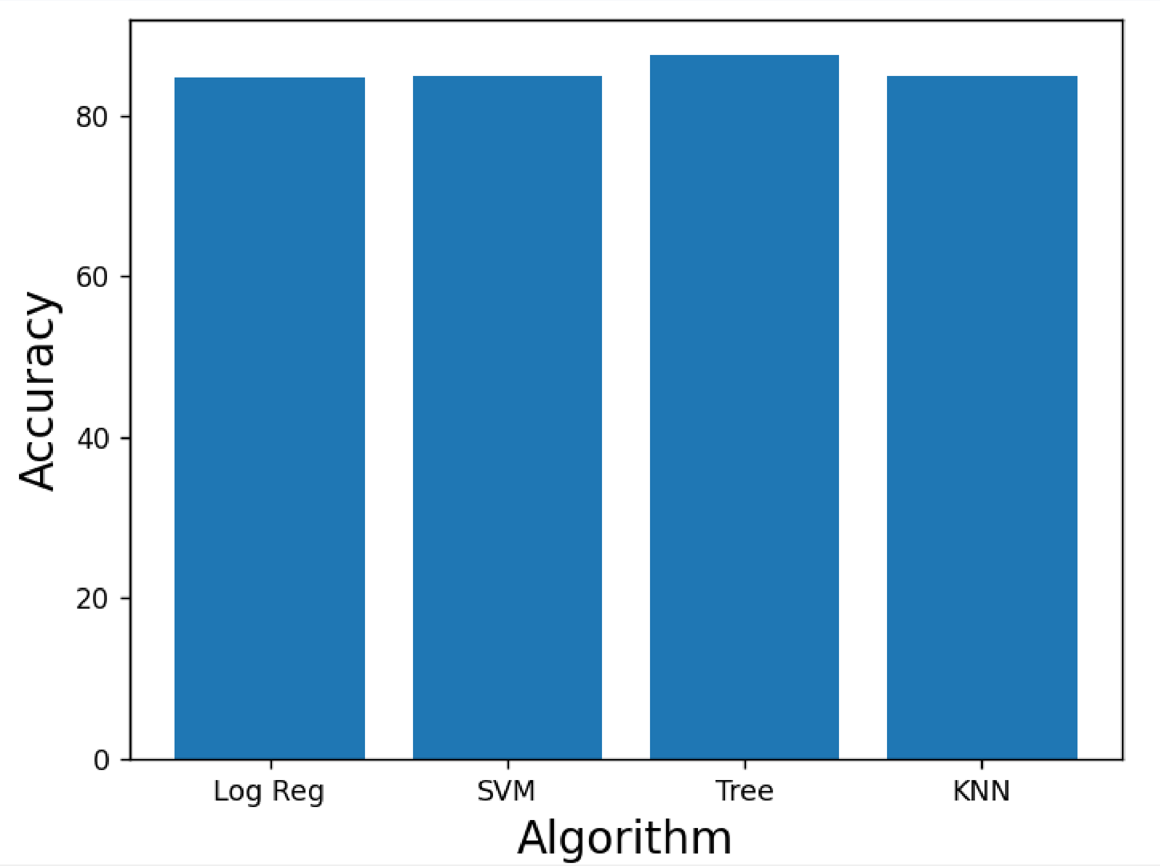




Section 5

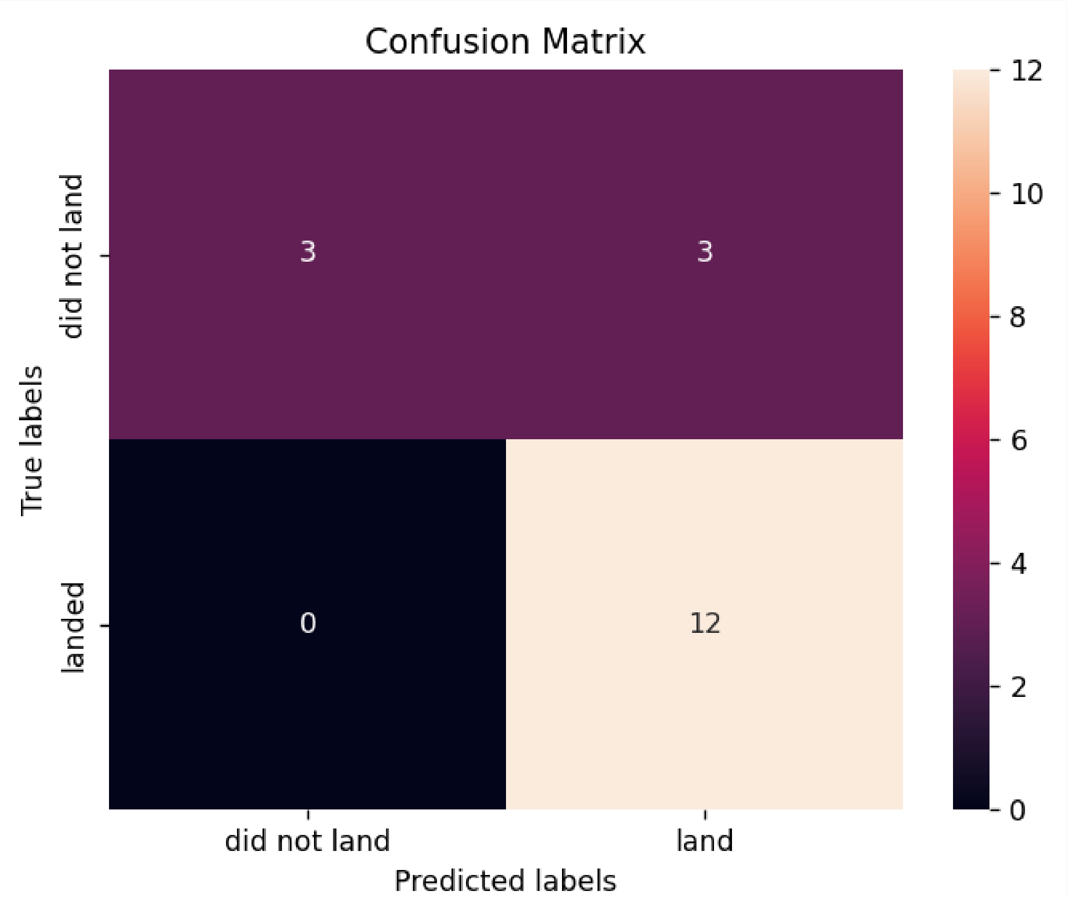
Predictive Analysis (Classification)

Classification Accuracy



- Results were close between the different models (84-88% accuracy)
- Tree proved to be the most accurate algorithm in the end

Confusion Matrix



- The tree algorithm proved to be the most reliable. The biggest issue the algorithm ran into was the values of rockets that did not land

Conclusions

- Low weighted payloads performed much better overall than heavier payloads
- KSC LC-39A has the most successful launches overall
- SpaceX is becoming more successful at launches overall – with the successful launch rate consistently increasing over time
- ES-L1, SSO, HEO, and GEO had the best success rates overall

Thank you!