IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Steven L. Stolze
5 December 2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection
    - API
    - Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis
    - SQL
    - Data Visualization
  - Interactive Visual Analitcs through Folium Dashboard
  - Machine Learning Predictions
- Summary of all results
  - Data Analysis
  - Predictive Analytics

Introduction

- Project background and context
  - By introducing the ability to reuse the first stage booster, SpaceX has shown to be a more cost-effective means of rocket launches by utilizing the Falcon 9 and reducing the cost to nearly 1/3 of the competitions saving customers over 100 million dollars per launch. SpaceY is at the heels of SpaceX in similar feats, so if we are able to conclude the price of each one of our launches, we can compete more effectively. By utilizing machine learning models, we can determine the likelihood of SpaceX being able to reuse the first stage.

- Problems you want to find answers
  - Determinants of first stage landing success
  - What the determinant conditions need to be for a success

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
- Perform data wrangling
- 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

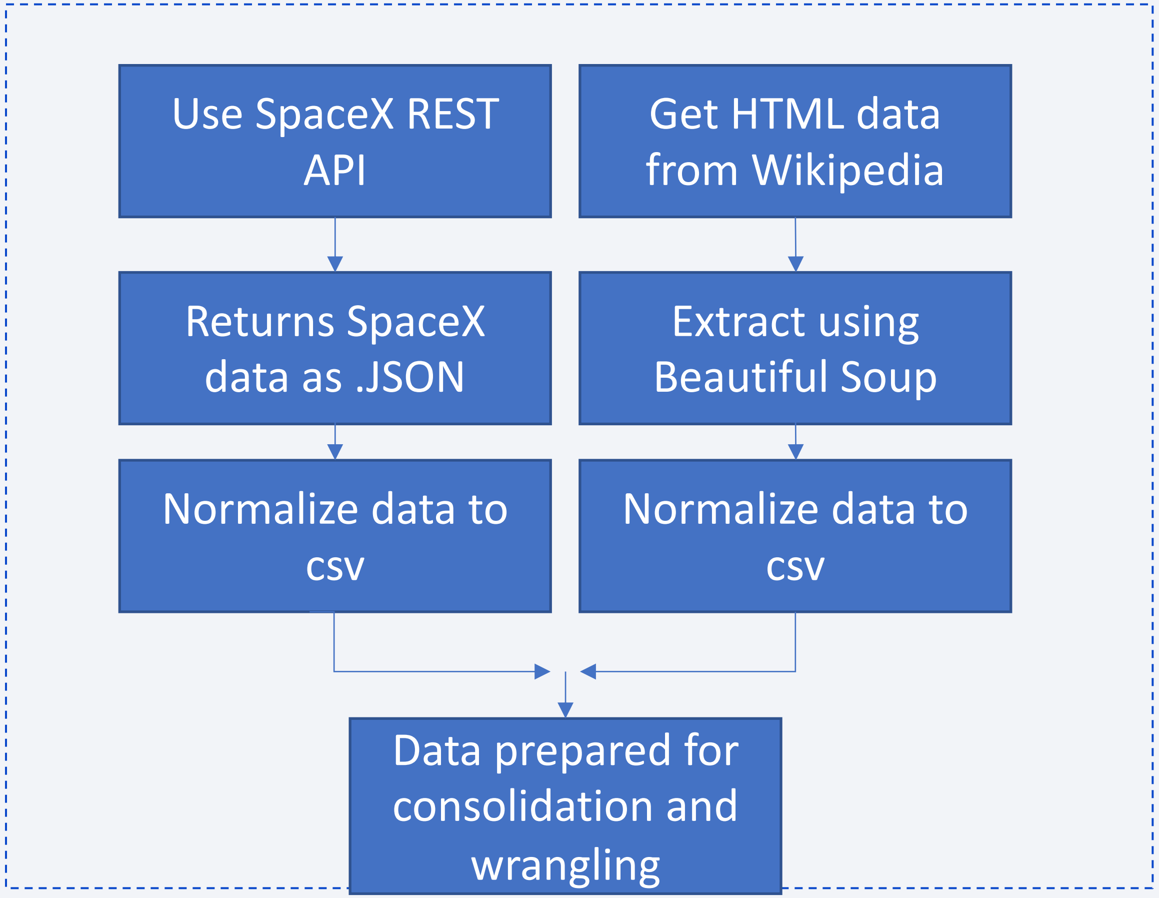
Data Collection

- The SpaceX API was utilized with the get request and decoded as a Json so we can convert the data into a pandas data frame. Missing values were normalized, and the data was cleaned.
- BeautifulSoup was utilized through web scraping of Falcon 9 launch records from Wikipedia so we could convert from HTML to a pandas data frame.

Data Collection

- We've gathered information from both the SpaceX REST API and Wikipedia.
- For the REST API, we extract it and as a .JSON file ands normalize the data into a .csv file.
- For the Wikipedia data, we web scrape utilizing HTML and extract it using Beautiful Soup to normalize it as a .csv file.
- From there we take both sets and consolidate it for data wrangling.

- https://github.com/slstolze/DS_Space_Race_Capstone.git



Data Collection – SpaceX API

- 1. Get Response from API
- 2. Assign response as Object and convert to .JSON file
- 3. Clean data
- 4. Assign list to dictionary
- 5. Assign dictionary to data frame
- 6. Filter data frame
- 7. save data fame to .CSV

1

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

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

2

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
```

3

```
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}
```

4

5

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

```
data_falcon9=df[df['BoosterVersion']!='Falcon 1']
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
```

6

7

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```


- https://github.com/slstolze/DS_Space_Race_Capstone.git

Data Collection - Scraping

- 1. Get Response from HTML
- 2. Assign response as Object
- 3. Create Beautiful Soup Object
- 4. Find Tables
- 5. Get column names
- 6. Create dictionary
- 7. Add data to keys
- 8. Convert dict to dataframe
- 9. DF to .CSV

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

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

```
soup = BeautifulSoup(data, 'html5lib')
```

```
html_tables=soup.find_all('table')
```

```
column_names = []

for row in first_launch_table.find_all('th'):
    name=extract_column_from_header(row)
    if (name!=None and len(name)>0):
        column_names.append(name)
```

```
launch_dict= dict.fromkeys(column_names)

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']=[]
```

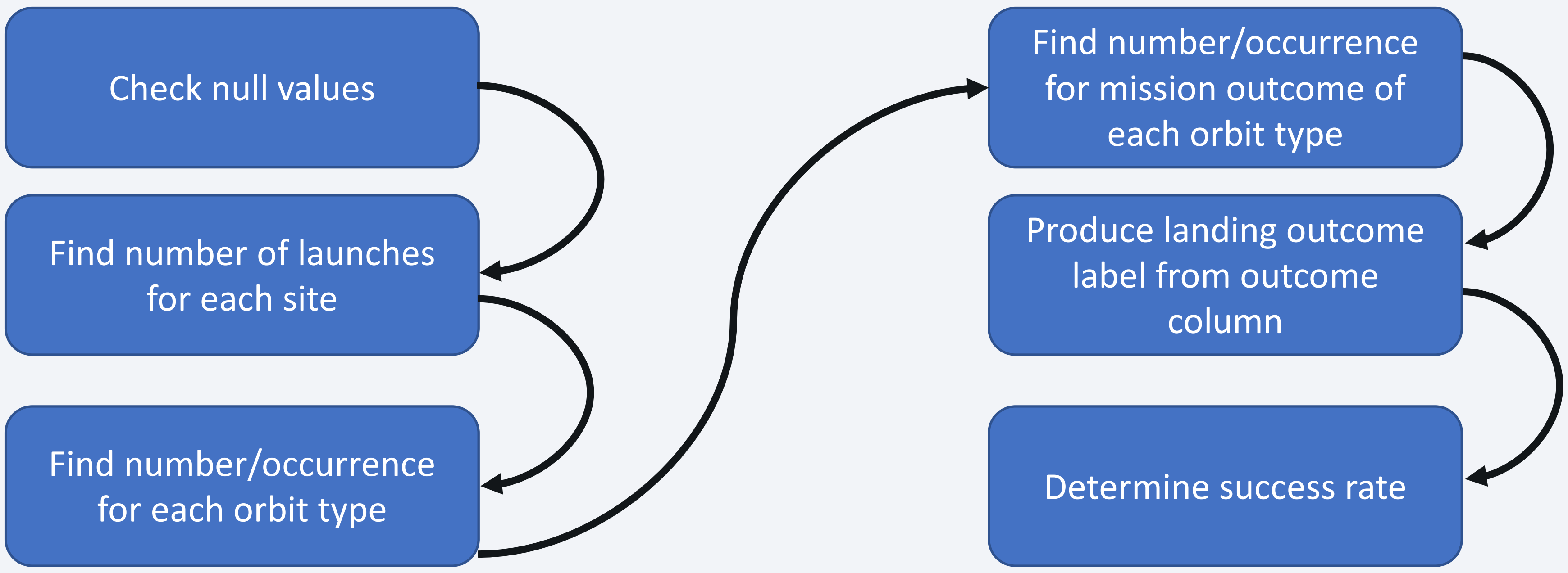
```
extracted_row = 0
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
    for rows in table.find_all("tr"):
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        row=rows.find_all('td')
        if flag:
            extracted_row += 1
            datatimelist=date_time(row[0])
            date = datatimelist[0].strip(',')
            time = datatimelist[1]
            bv=booster_version(row[1])
            if not(bv):
                bv=row[1].a.string
            print(bv)
            launch_site = row[2].a.string
            payload = row[3].a.string
            payload_mass = get_mass(row[4])
            orbit = row[5].a.string
            customer = row[6].a.string
            launch_outcome = list(row[7].strings)[0]
            booster_landing = landing_status(row[8])
```

```
df=pd.DataFrame(launch_dict)
```

```
df.to_csv('spacex_data.csv', index=False)
```

- https://github.com/slstolze/DS_Space_Race_Capstone.git

Data Wrangling

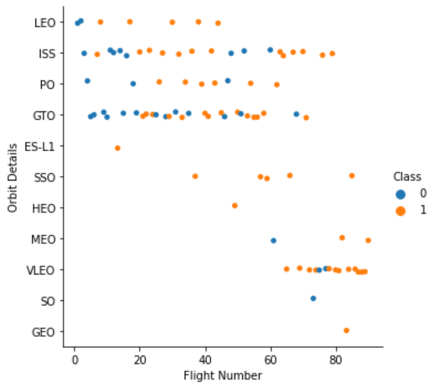


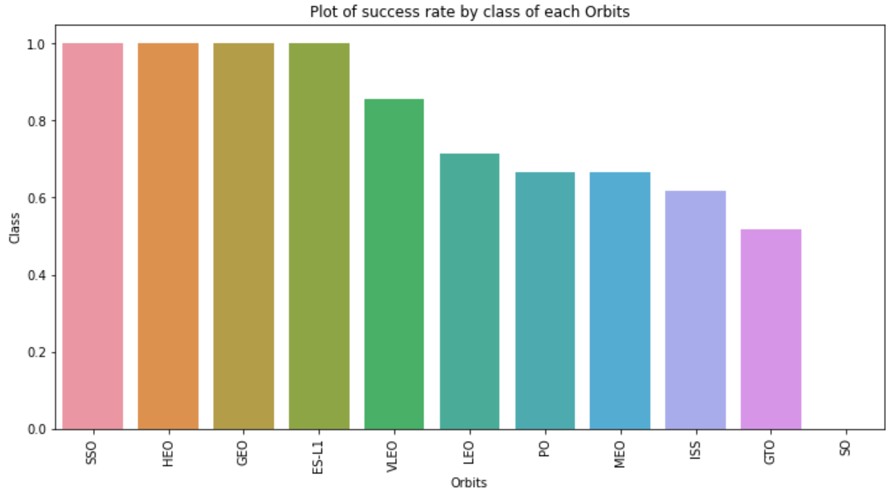
- https://github.com/slstolze/DS_Space_Race_Capstone.git

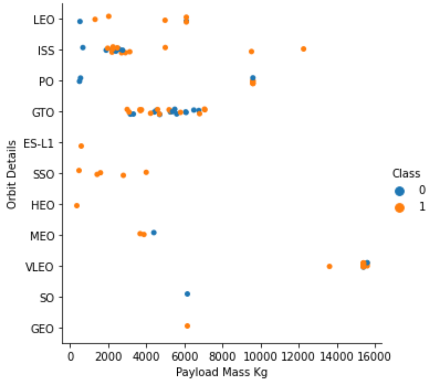
EDA with Data Visualization

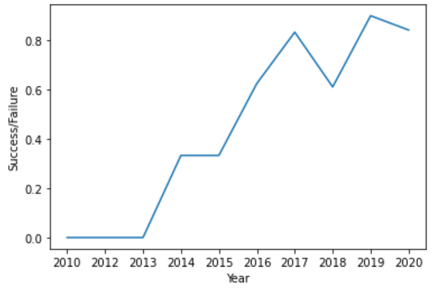
- We explore the patterns between
  - launch site/flight number,
  - launch site/payload
  - Orbit type/Payload
  - Orbit type/Flight Number
- We see the success rates by each class of orbit type and the success trend from 2010 to 2020.

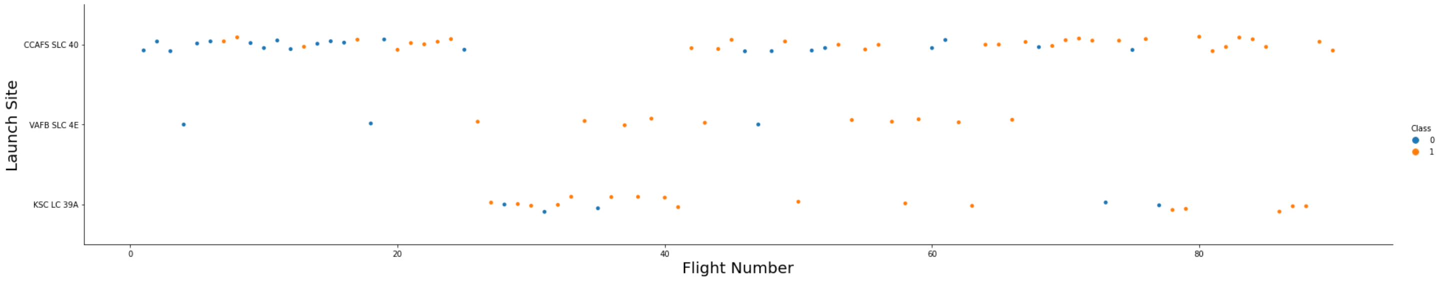
https://github.com/slstolze/DS_Space_Race_Capstone.git











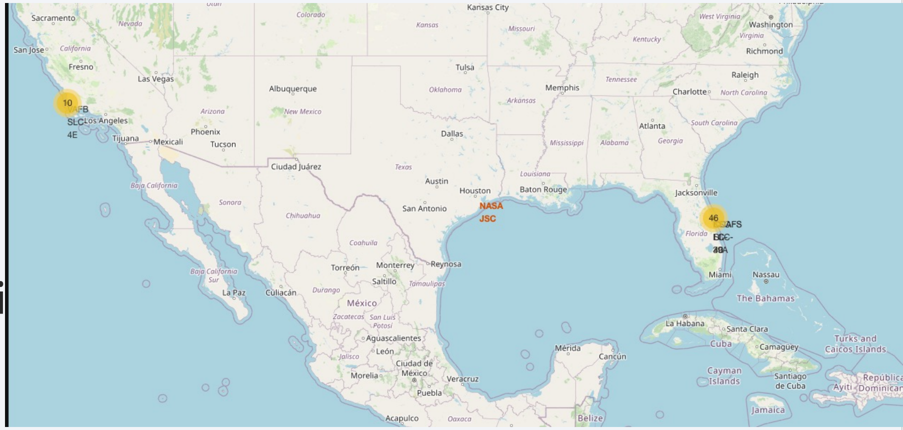
EDA with SQL

- The SQL queries we performed were:
  - Names of launch sites
  - Displaying 5 records where launch sites begin with “CCA”
  - Total Payload Mass carried by boosters launched by NASA
  - Average payload mass carried by booster version F9 v1.1
  - First successful landing date
  - Names of the boosters with success in drone ship w/payload >4K but <6K lbs.
  - Total # of successful and failure mission outcomes.
  - Names of the booster versions that had the max payload mass
  - List of records that display Month, failure, booster version, launch site in 2015
  - Rank the count of successful landing outcomes form 04/06/2010-20/03/2017

- https://github.com/slstolze/DS_Space_Race_Capstone.git

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
- We marked all launch sites on a map with circles denoting success/failed launches for each site for quick visualization.
- We then used marker clusters for high success rates.
- Finally we determined the distance between launch sites
  - This is to determine if there are any correlations with logistics of locati

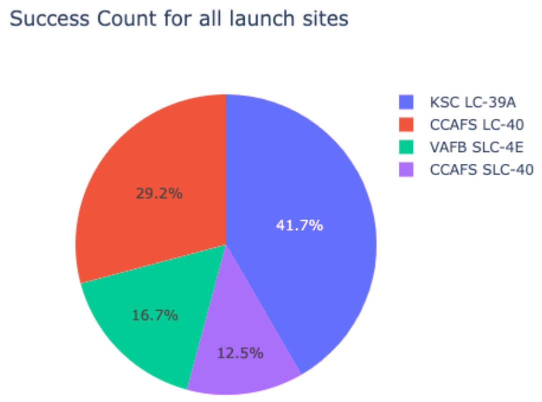


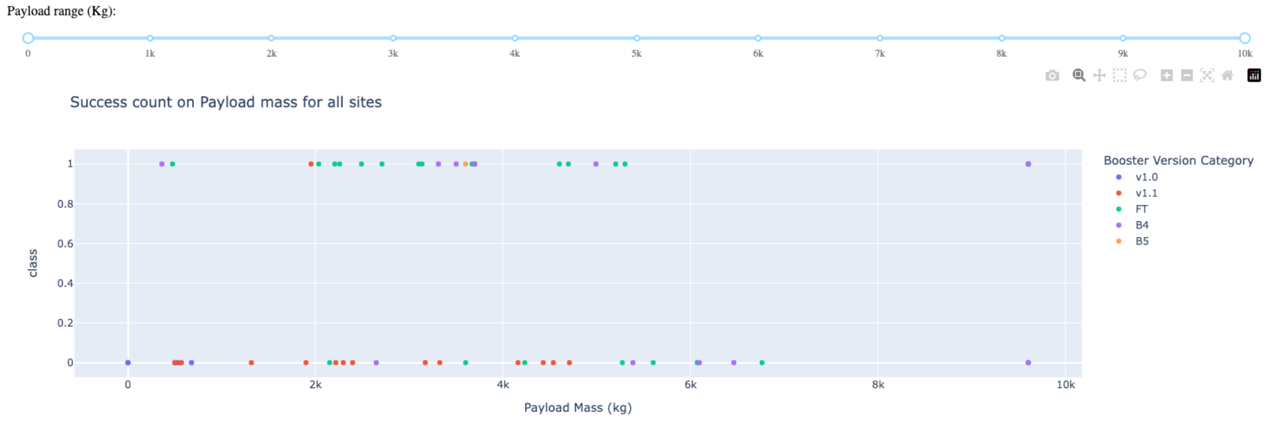
- https://github.com/slstolze/DS_Space_Race_Capstone.git

Build a Dashboard with Plotly Dash

- Plotly dash was used to build an interactive dashboard
    - Pie charts with launch totals by sites
        - Determine the site with the highest success count
    - Scatter graph between success vs. payload mass.
        - Determine the payload mass with the highest success rate/booster version

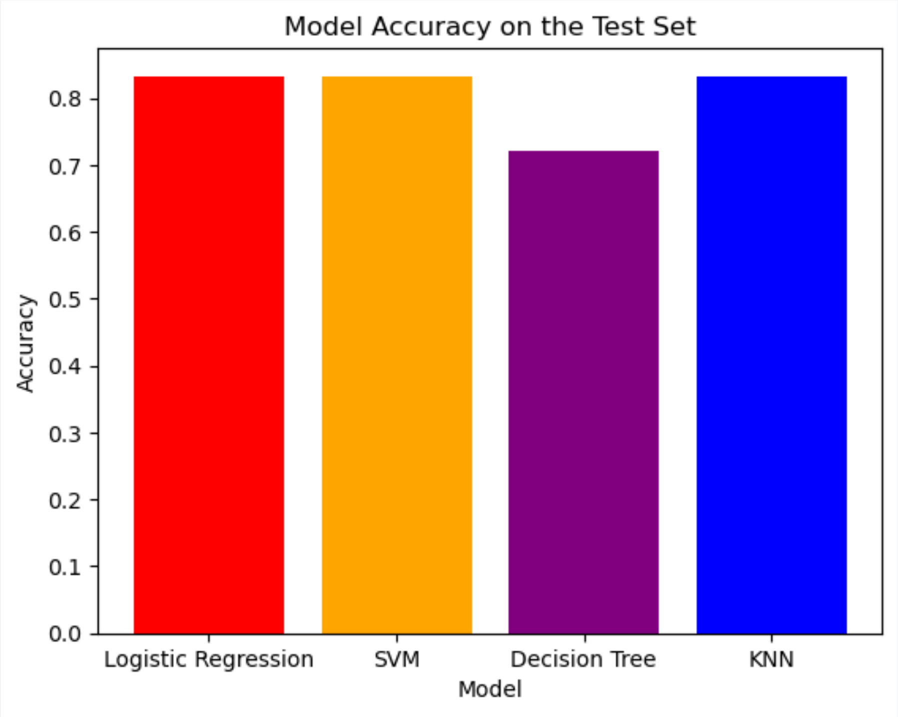
- https://github.com/slstolze/DS_Space_Race_Capstone.git

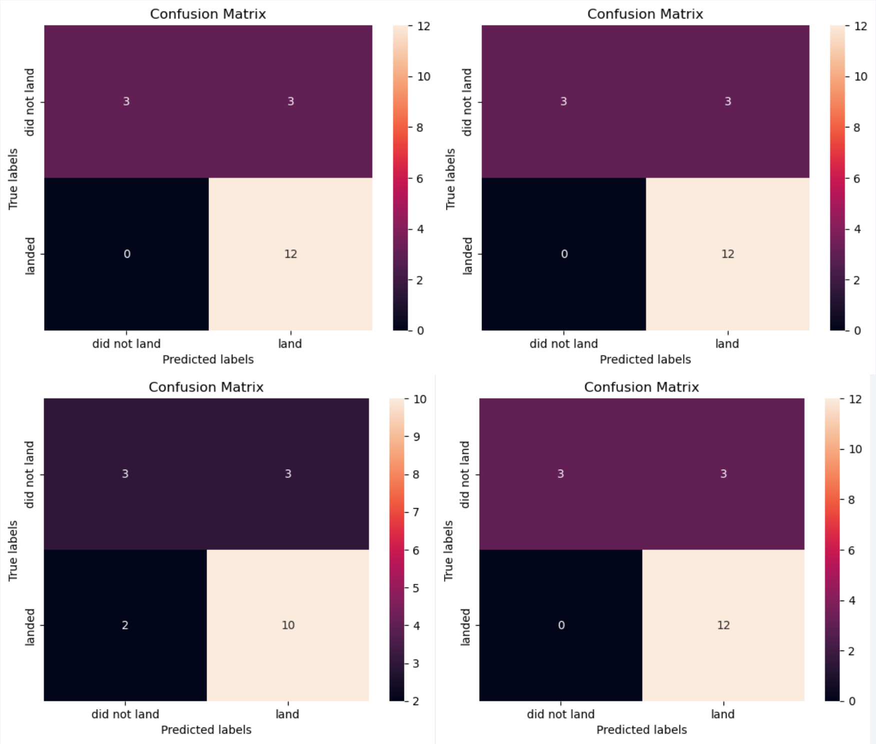




Predictive Analysis (Classification)

- The Logistic Regression, SVM and KNN all had the simmilar accuracy with 83.3% with the Decision Tree having the lowest with 72.2%





https://github.com/slstolze/DS_Space_Race_Capstone.git

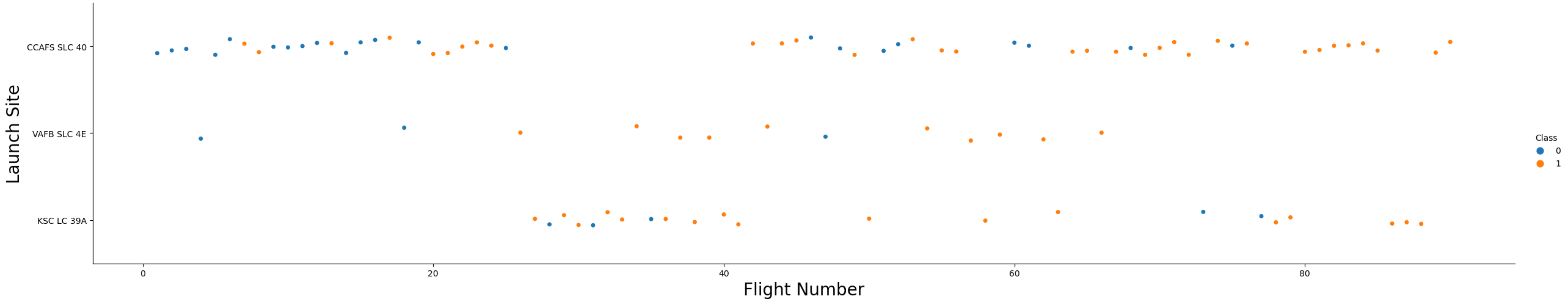
Results

- The Logistic Regression, KNN, and SVM models had the best accuracy with the Decision Tree coming in slightly lower.
- Payloads with lower weights performed better
- KSC LC 39A had the most successful launces
- Orbit GEO, HEO, SSEO, ES L1 had the best success rate.

Section 2

Insights drawn from EDA

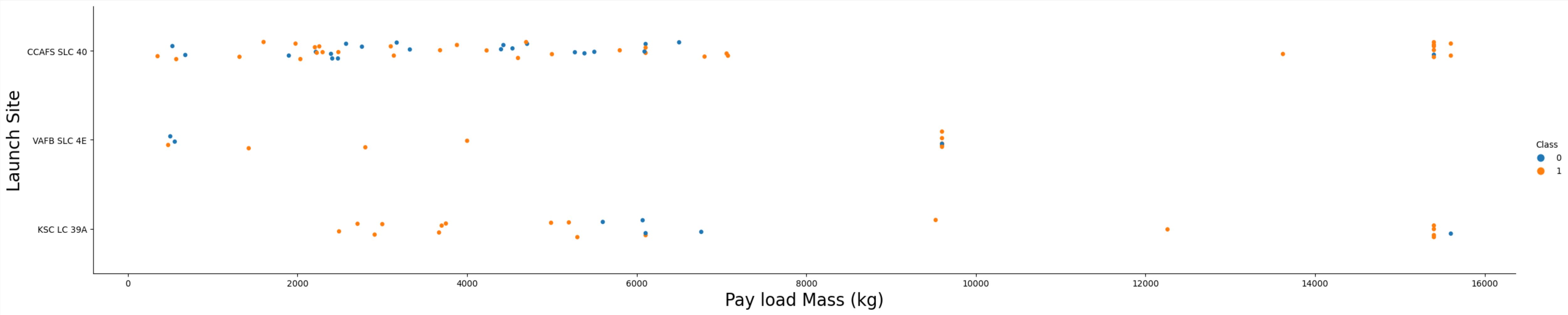
Flight Number vs. Launch Site



- The launches from CCAFS SLC 40 shows a higher flight number than other sites

Payload vs. Launch Site

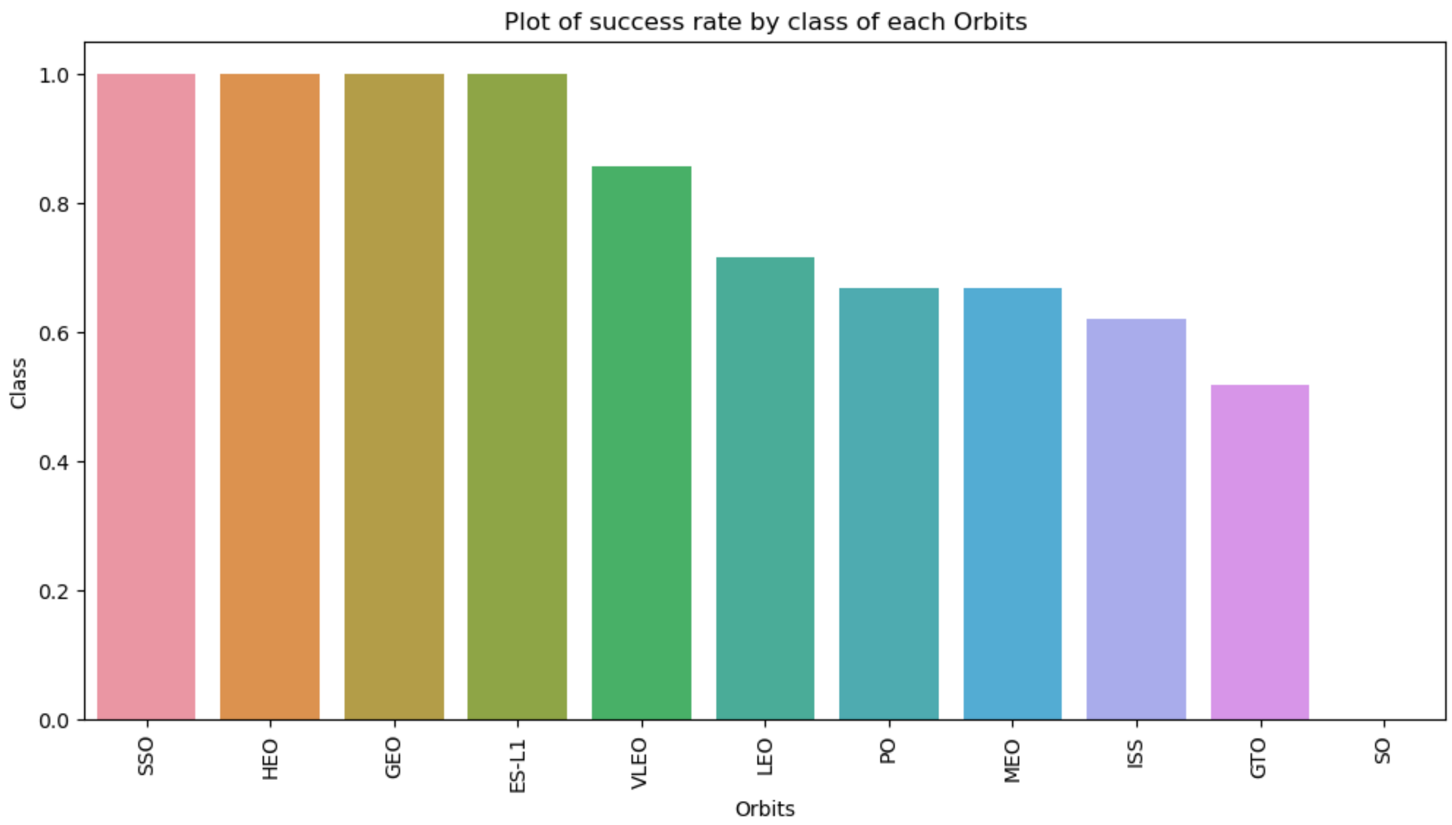
- Pay loads with a lower mass mostly come from CCAFS SLC 40.



Success Rate vs. Orbit Type

- SSO, HEO, GEO and ES-L1 have the highest success rate.

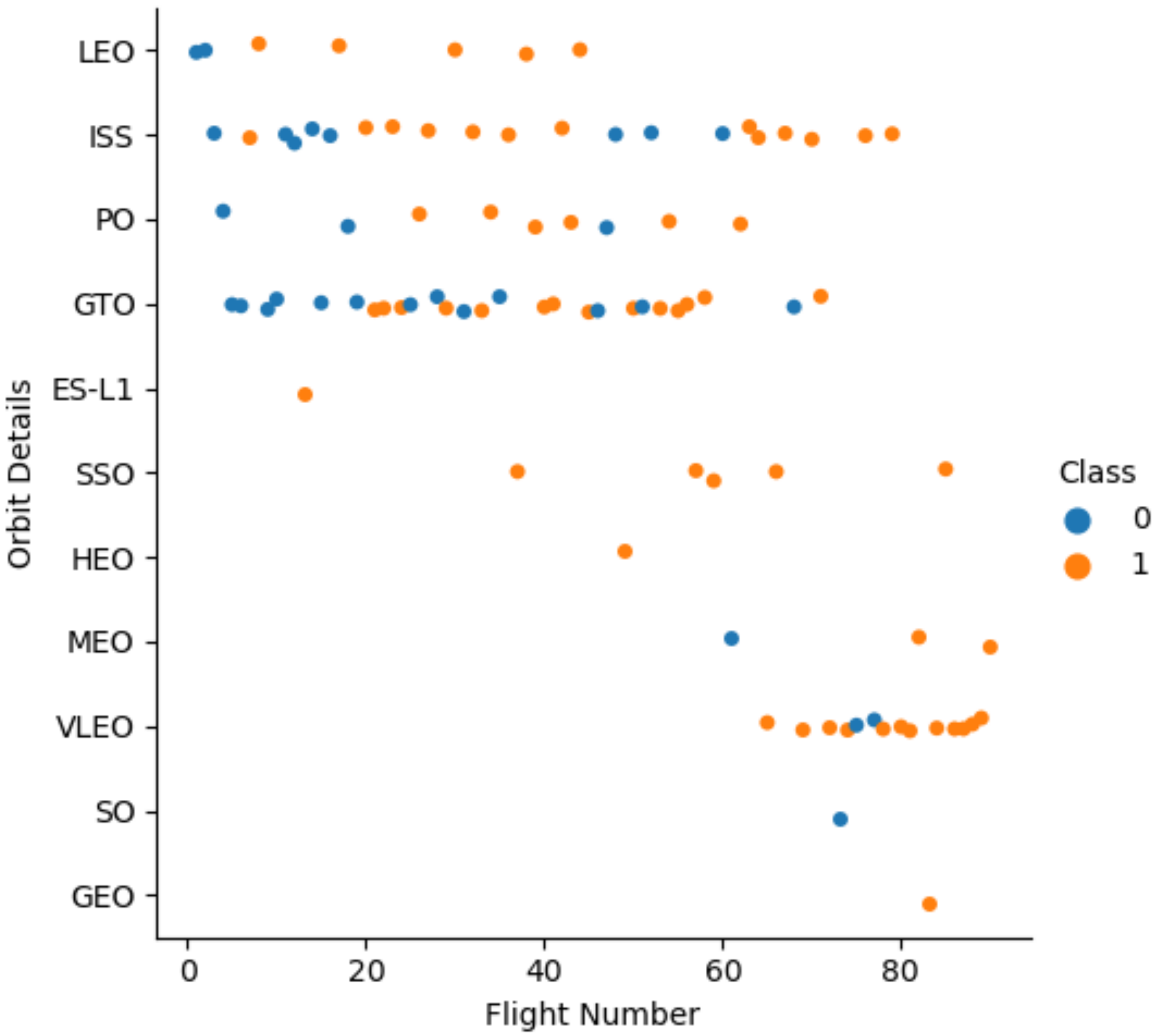
- Show the screenshot of the scatter plot with explanations



Flight Number vs. Orbit Type

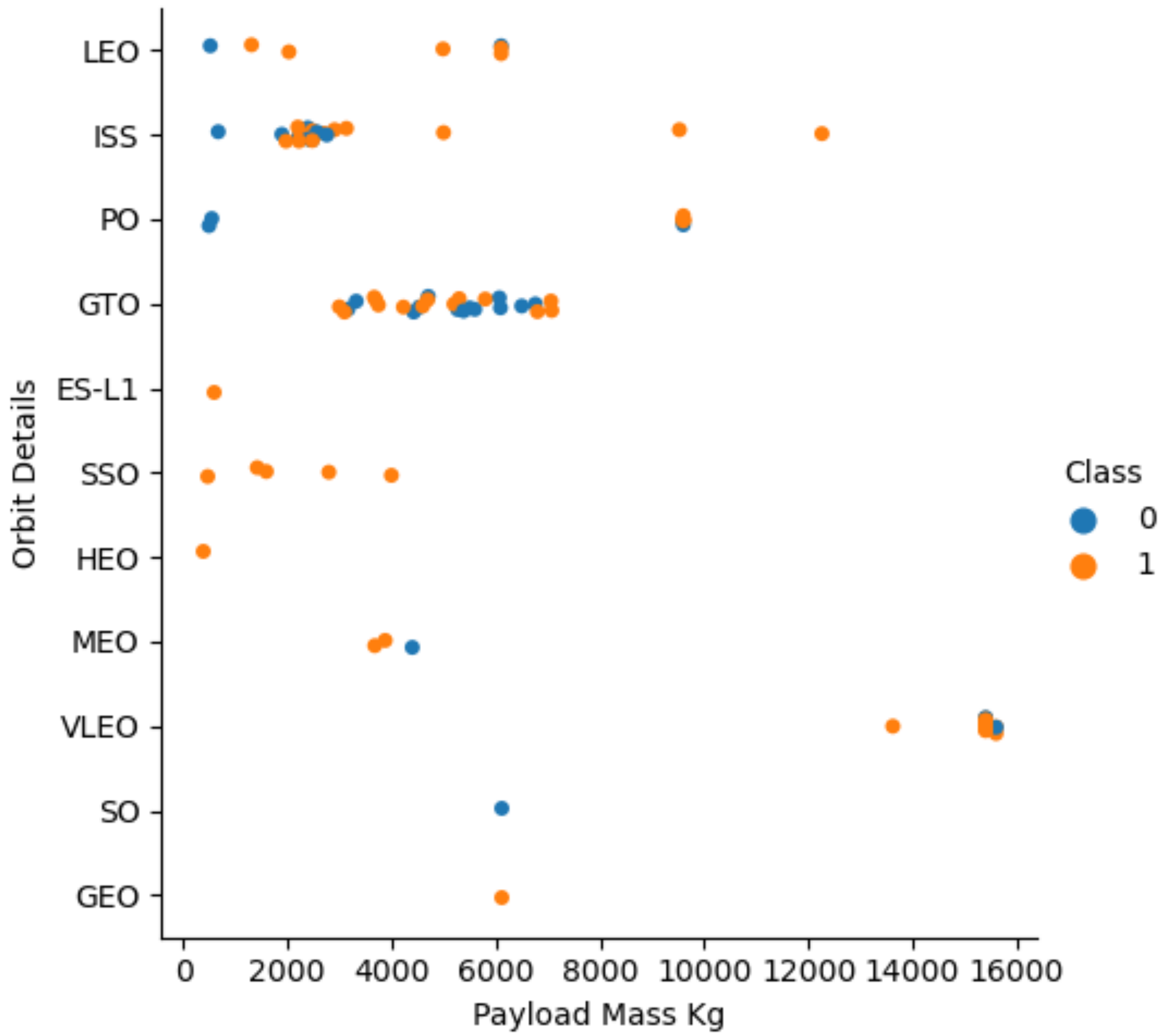
- LEO has a relationship with higher success with higher flight numbers.

- SSO has the best success rate



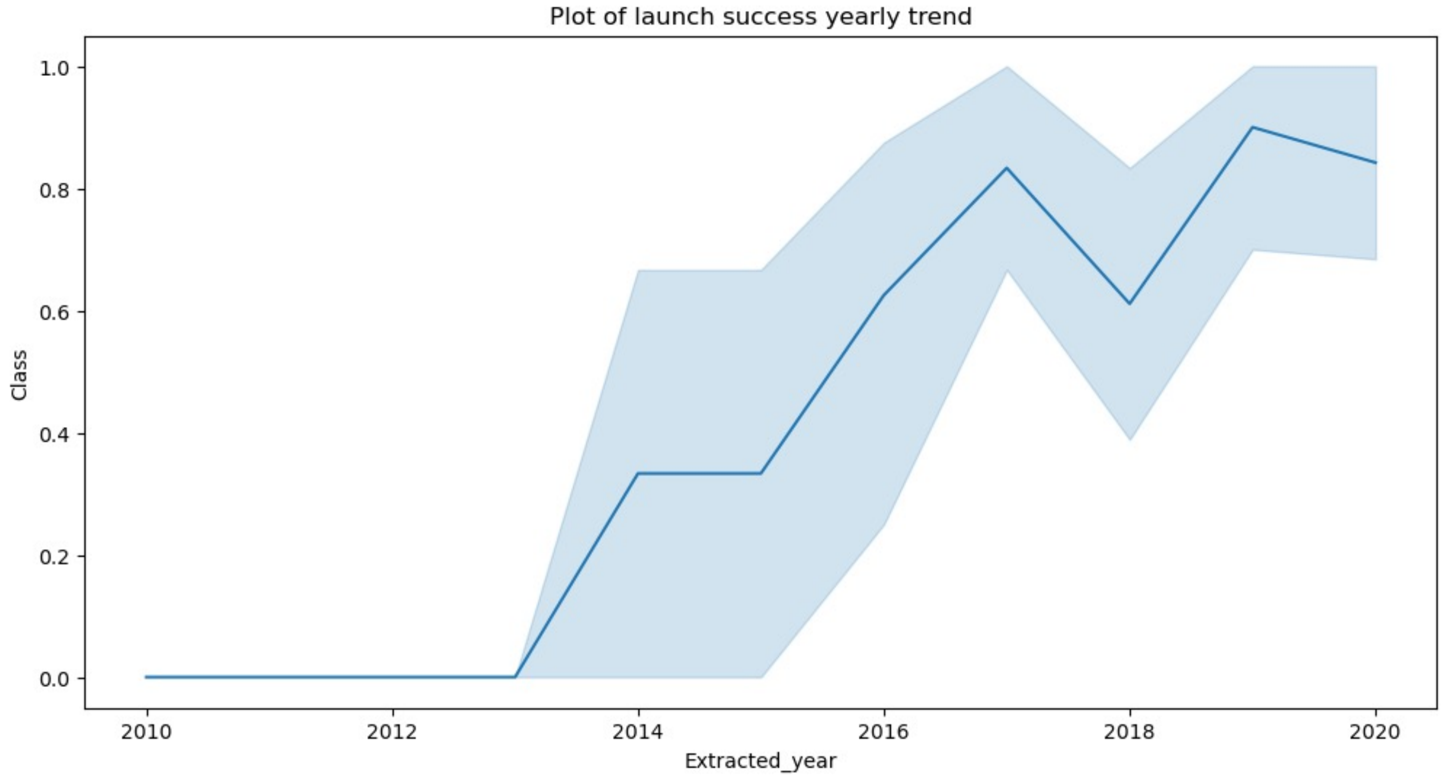
Payload vs. Orbit Type

- Heavier payload success is seen with LEO and ISS
- Low payload success with SSO and HEO is seen



Launch Success Yearly Trend

- Success rates have increased since 2013.
- The increasing trend has become more stabilized since 2017.



All Launch Site Names

- %sql select Unique (LAUNCH_SITE) from SPACEXTBL;

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

Launch Site Names Begin with 'CCA'

- %sql SELECT * from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;

```
%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
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

- %sql select sum(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;

```
%sql select sum(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;

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

payloadmass
619967

Average Payload Mass by F9 v1.1

- %sql select avg(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;

```
%sql select avg(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;

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

payloadmass
6138.287128712871

First Successful Ground Landing Date

- select min(DATE) from SPACEXTBL Where Landing__Outcome ='Success (ground pad)';

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- %sql select BOOSTER_VERSION from SPACEXTBL where LANDING__OUTCOME='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000;

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

Total Number of Successful and Failure Mission Outcomes

- %sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS OUTCOME FROM SPACEXTBL GROUP BY MISSION_OUTCOME;

```
%sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS OUTCOME FROM SPACEXTBL GROUP BY MISSION_OUTCOME;
```

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

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

Boosters Carried Maximum Payload

- %sql select BOOSTER_VERSION as boosterversion from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL);

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

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

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

- %sql SELECT DATE,MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE FROM SPACEXTBL where EXTRACT(YEAR FROM DATE)='2015' order by date desc;

DATE	MISSION_OUTCOME	BOOSTER_VERSION	LAUNCH_SITE
2015-12-22	Success	F9 FT B1019	CCAFS LC-40
2015-06-28	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
2015-04-27	Success	F9 v1.1 B1016	CCAFS LC-40
2015-04-14	Success	F9 v1.1 B1015	CCAFS LC-40
2015-03-02	Success	F9 v1.1 B1014	CCAFS LC-40
2015-02-11	Success	F9 v1.1 B1013	CCAFS LC-40
2015-01-10	Success	F9 v1.1 B1012	CCAFS LC-40

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

- **%sql** SELECT LANDING__OUTCOME, COUNT(*) AS COUNT_LAUNCHES FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY LANDING__OUTCOME ORDER BY COUNT_LAUNCHES DESC;

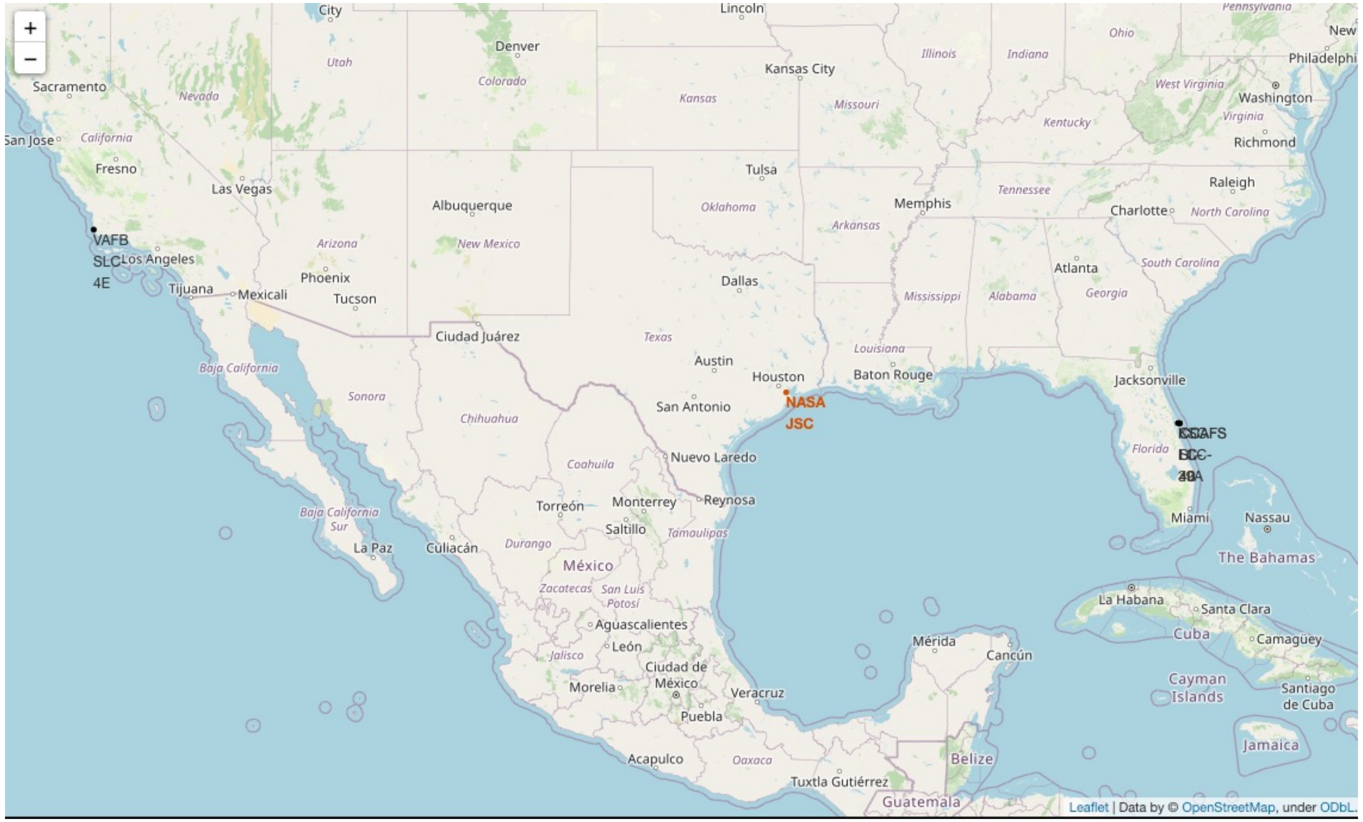
LANDING__OUTCOME	COUNT_LAUNCHES
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

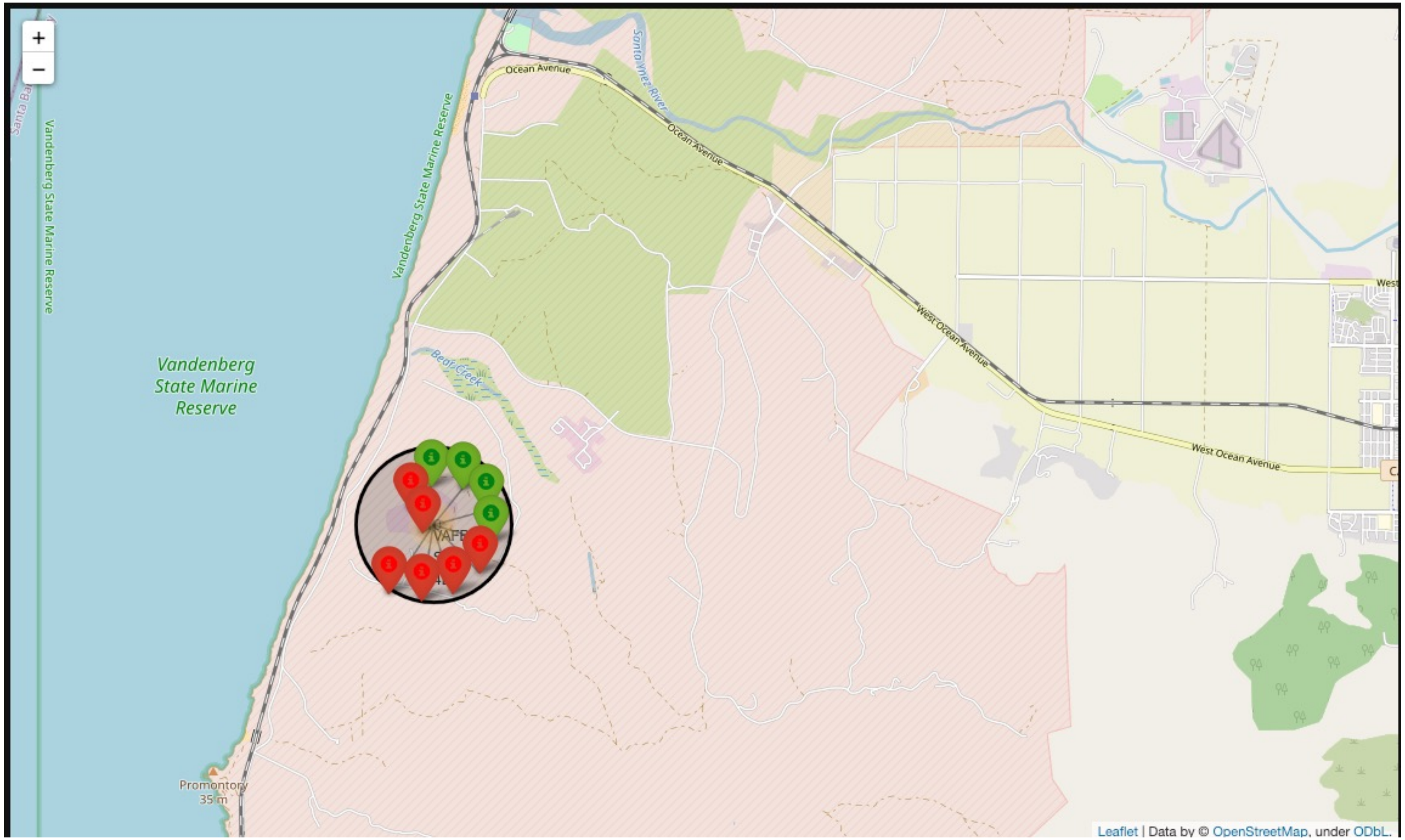
NASA Johnson Space Center and Launch Site Locations

- Key locations identified on the folium map
- JSC located in TX
- Vandanburg AFB located on west coast
- Kennedy Space Center/Cape Canaveral AFS on East Coast



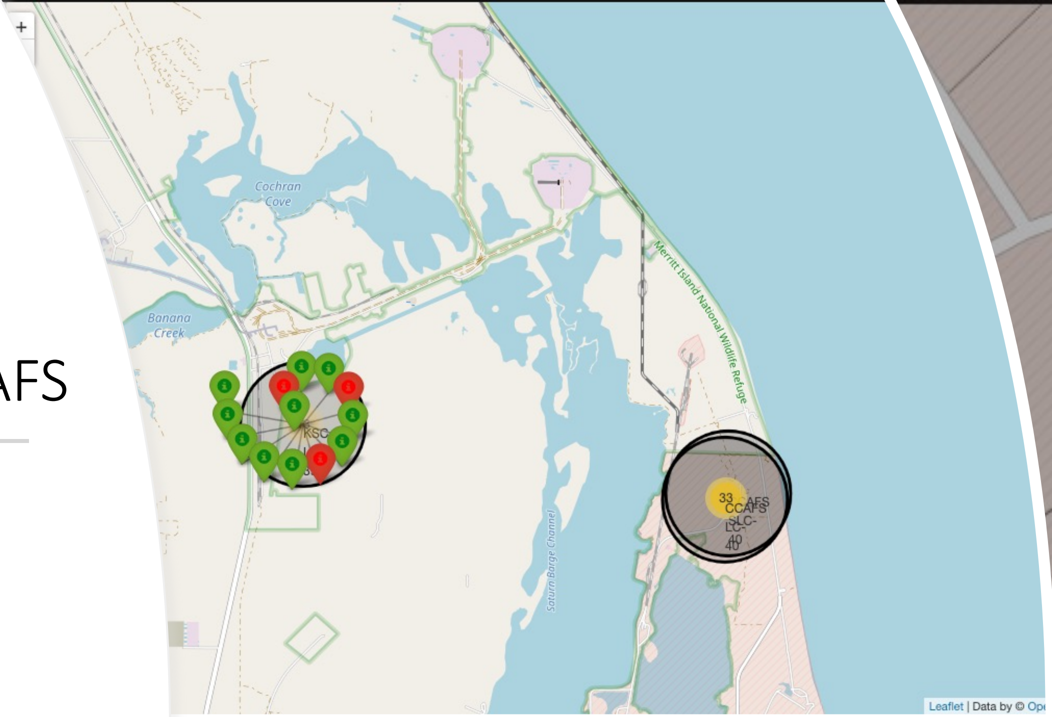
Vandenburg AFB Launch results

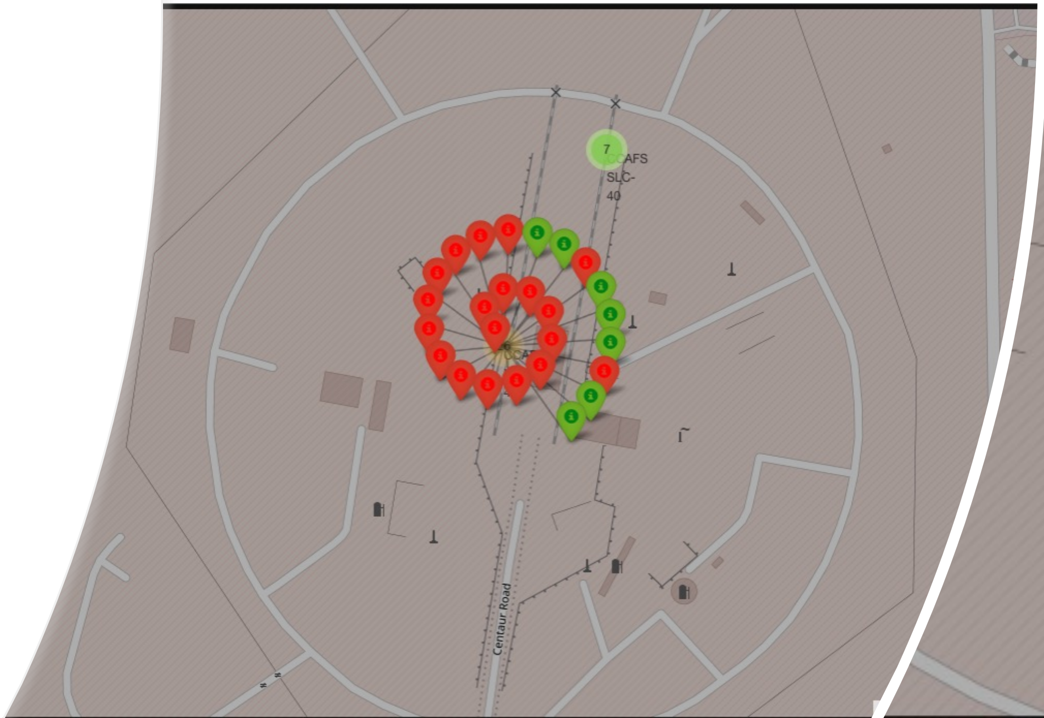
- Green = Successful Launch
- Red = Unsuccessful Launch



Launch results
Top Left: KSC
Bottom Left/Right: CCAFS

- Green = Successful Launch
- Red = Unsuccessful Launch

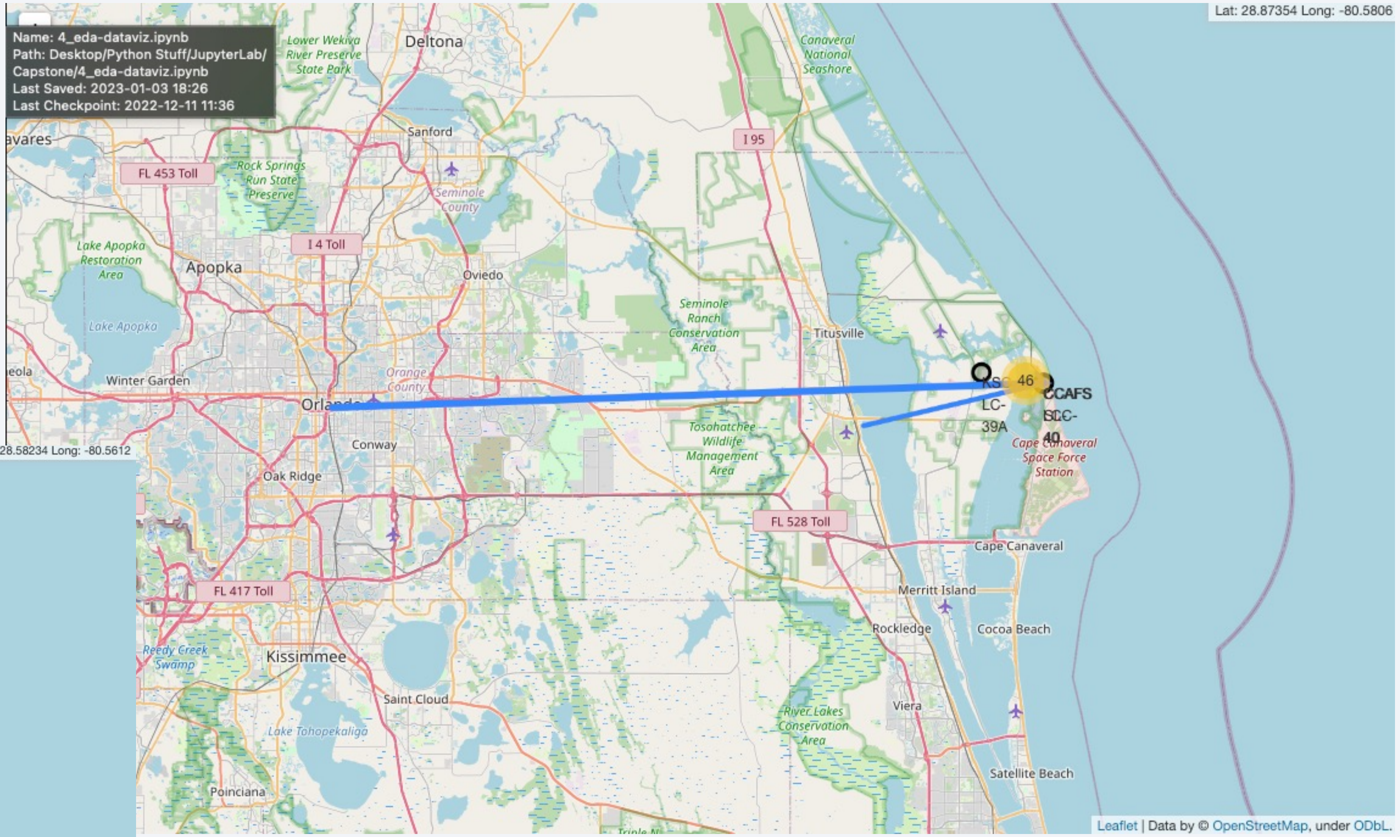


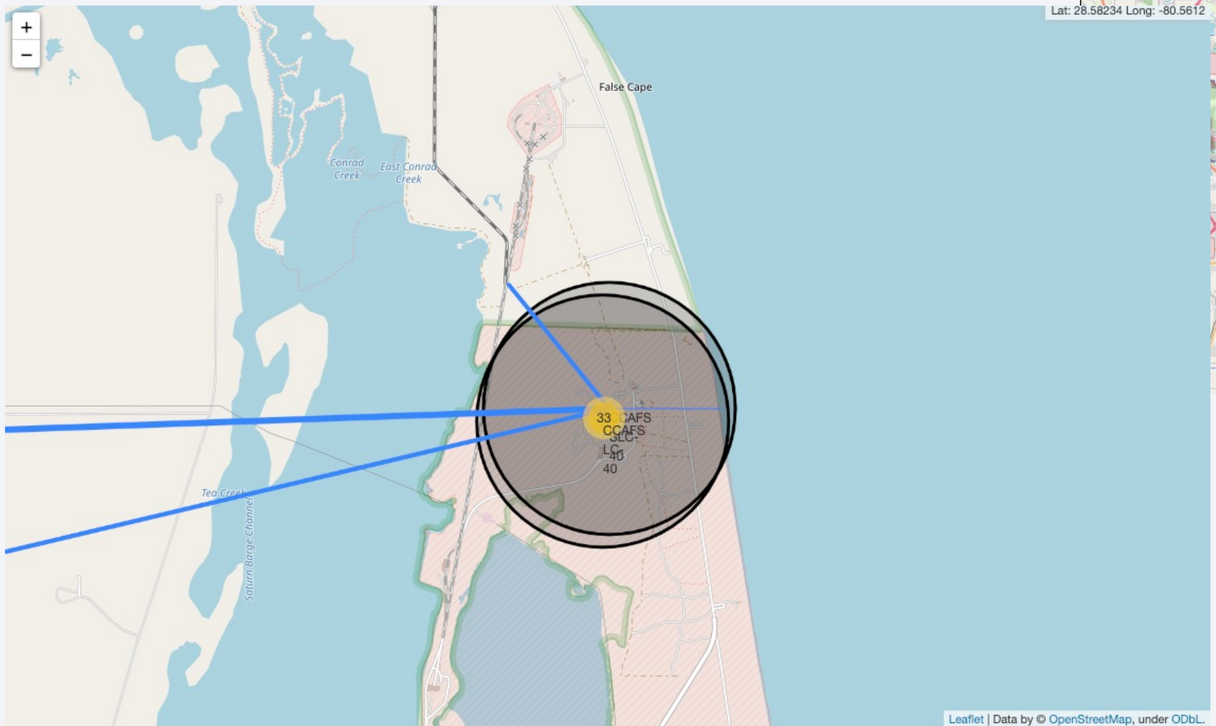


Distance to significant areas from Launch Site CCAFS

- Distance in Miles

```
('From Coastline', 0.8762983388668405),
('From Orlando', 78.41446868981808),
('From Hightway', 20.400323649726467),
('From Railway', 1.2638386820575733))
```



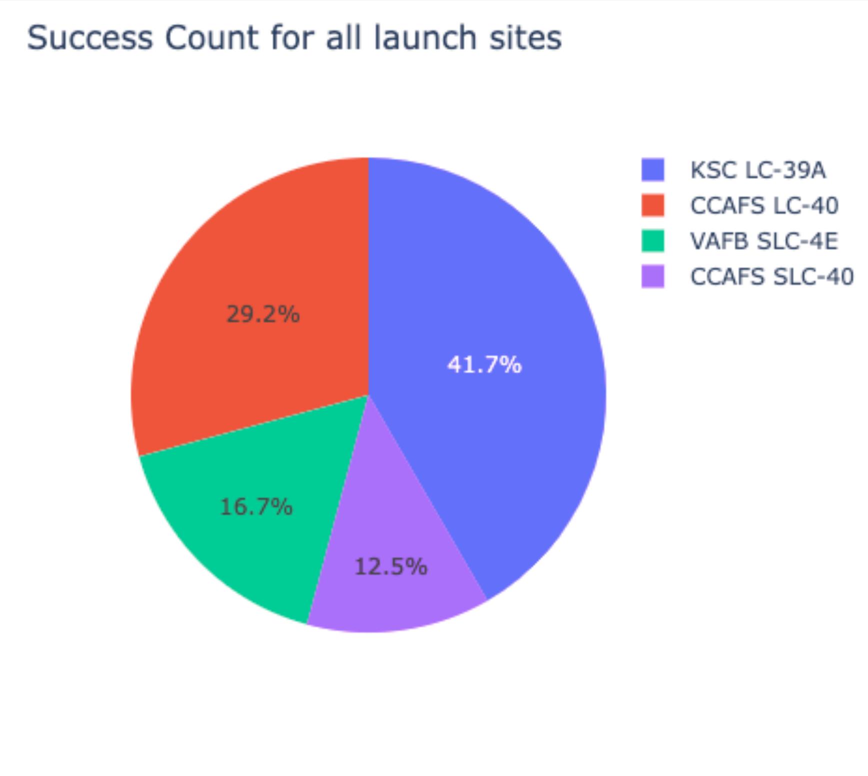


Section 4

Build a Dashboard with Plotly Dash

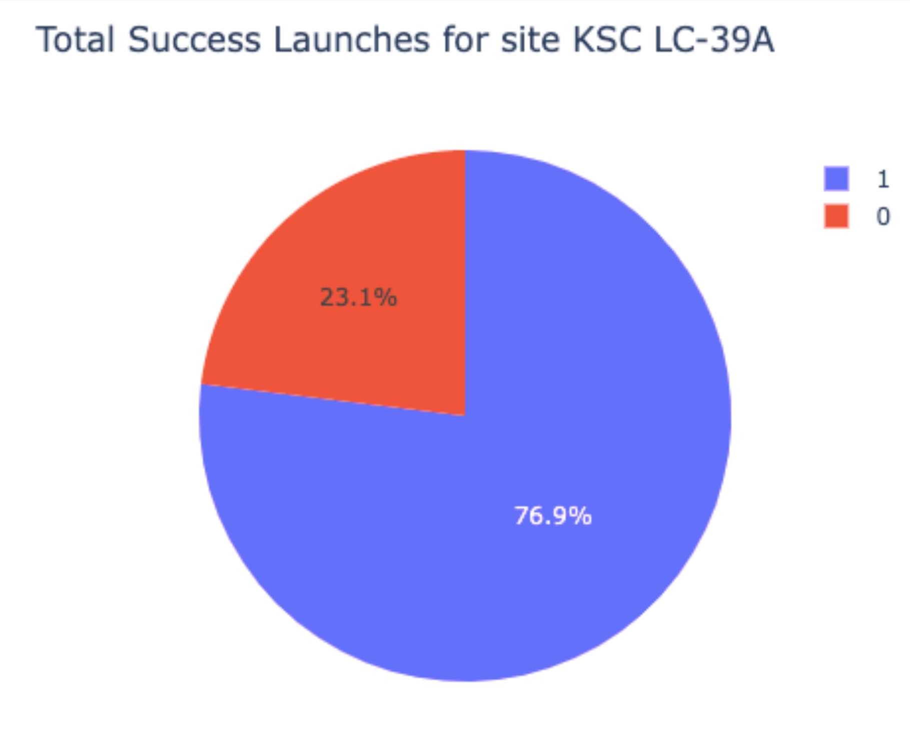
Success rate by launch site

- This pie chart highlights the significance in success rates between launch sites



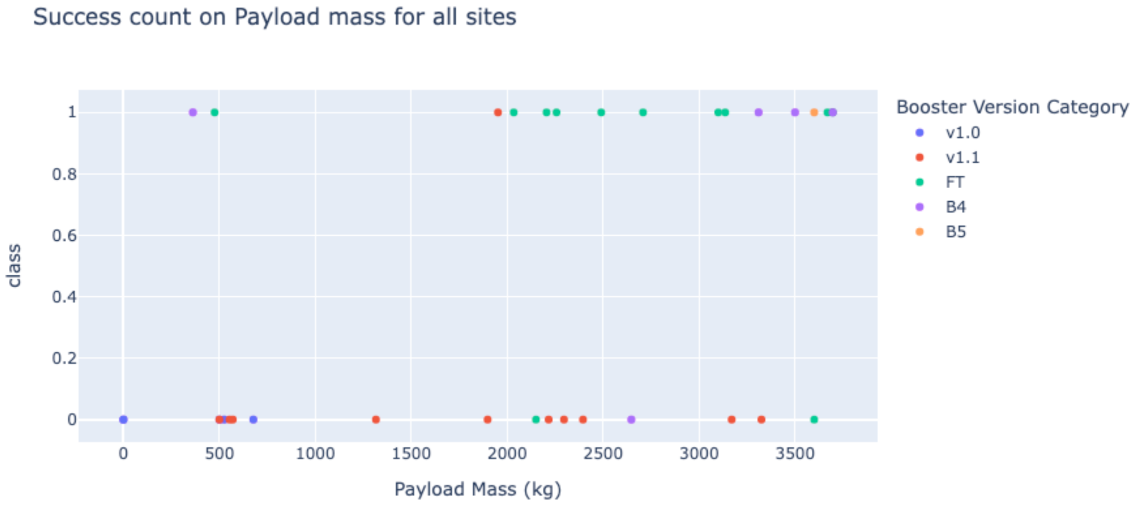
KSC LC-38A

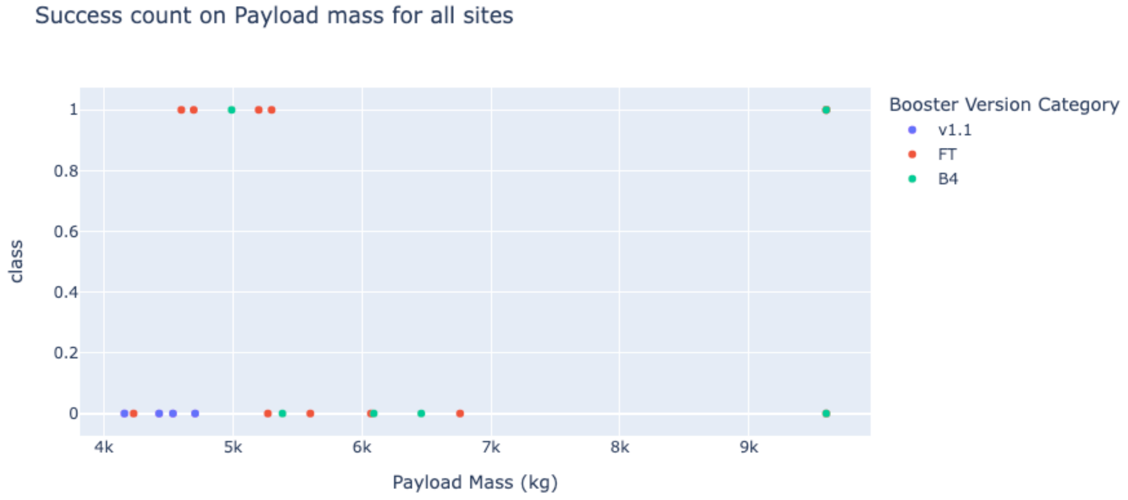
- KSC LC-39A had the highest success rate, and it is broken down in this pie chart even further.



Payload Mass Success by Class

- These two charts are the success count per class and by breaking them down between payload weights, we can see that the lower payload weights had a more significant success corilation.



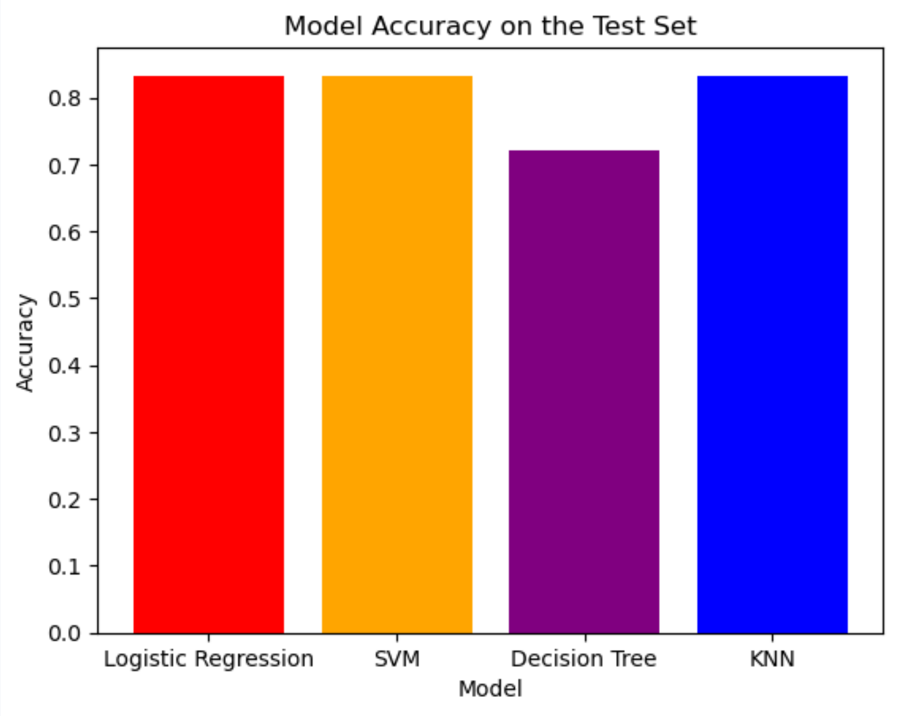


Section 5

Predictive Analysis (Classification)

Classification Accuracy

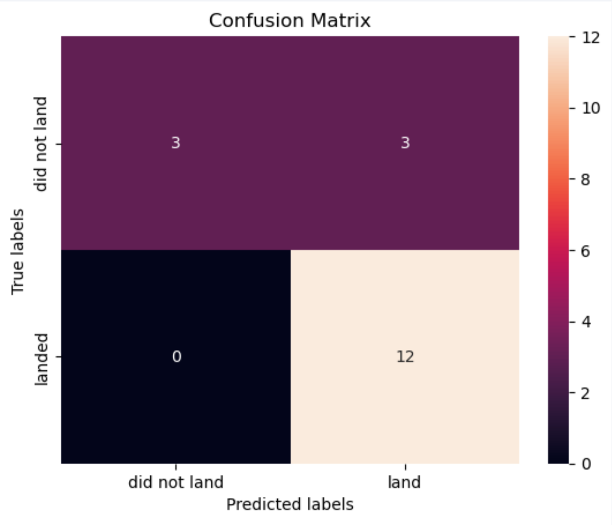
- The Logistic Regression and SVM had the highest accuracy with 94.4% while the KNN had 88.8% and the Decision Tree had the lowest with 83.3%



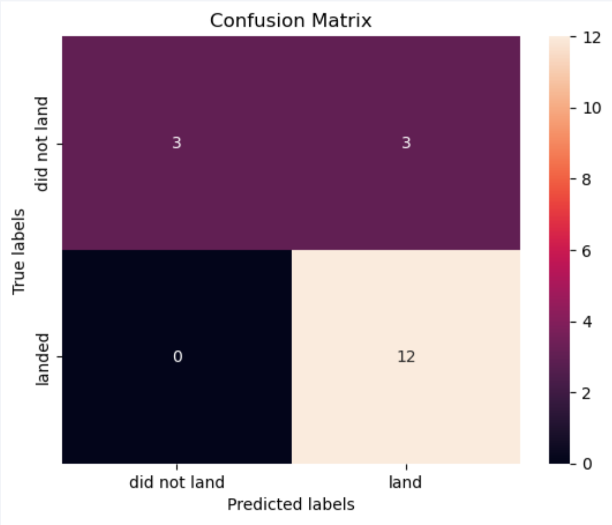
Confusion Matrix

- We can see that the Logistic Regression, SVM, and KNN can distinguish between the different classes, and that all models have a high false positive. The Decision tree was the most inacurate having higher false negatives.

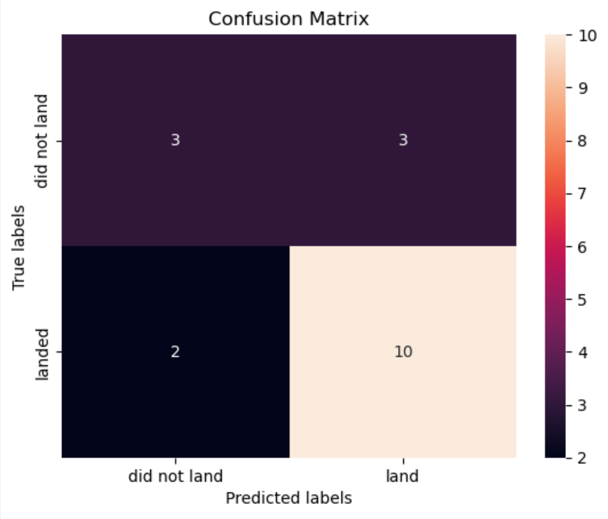
Log Reg



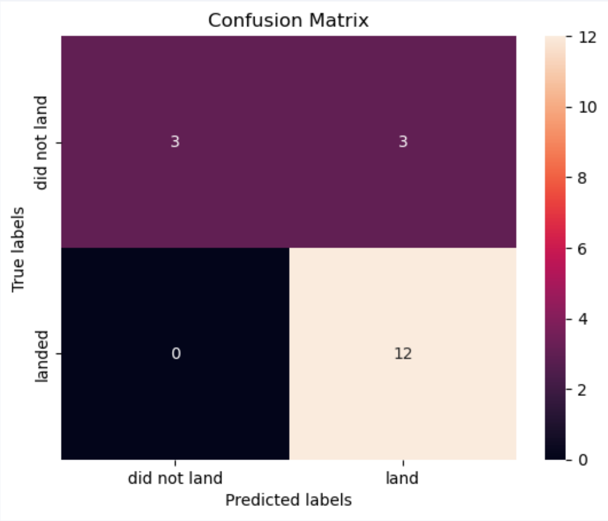
SVM



Decision Tree



KNN



Conclusions

- Orbit GEO, HEO, SSO, ES L1 have the best success rates
- Launch success has become more successful since 2017 and the trend could continue upward with further advances
- KSC LC 39A had the most successful launches
- Low payload mass has a higher success rate than heavier payload masses
- The SVM, KNN, and Logistic Regression models seem to do best with accuracy

Thank you!