IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Luisa Folle

February 10th, 2022

Outline

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

Executive Summary

Summary of methodologies

- Data Collection (‘get request’ to API)
- Data Collection with Web Scraping (Wikipedia)
- Exploratory Data Analysis (EDA):
  - Data Wrangling
  - SQL
  - Data Visualization
- Interactive Visual Analytics with Folium
- Machine Learning Prediction

Summary of all results

- Exploratory Data Analysis results
- Interactive Analytics
- Predictive Analysis results

Introduction

Aerospace companies cost for rocket launches can reach US$ 165 million each, whereas SpaceX advertises Falcon 9 launches with a cost of US$62 million. That difference is greatly due to SpaceX reusing the first stage. SpaceX is the only private company ever to return a spacecraft from low-Earth orbit. Determining if the first stage will land can help determine the cost of a launch, which is relevant information when an alternate company wants to bid against SpaceX for a rocket launch.

Goals:

- Predict if SpaceX will achieve a successful landing;
- Identify variables influencing success landing rate;

Section 1

Methodology

Methodology

- Data collection methodology:
- - Request to the SpaceX API
- - Web scraping (Wikipedia)
- Perform data wrangling
  - - Exploratory Data Analysis
  - - Determine Training Labels
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - - Create machine learning pipeline (build, tune, evaluate classification models)

Data Collection – SpaceX API

https://github.com/luisafolle/Applied-Data-Science-Capstone-IBM--SpaceX/blob/d1c7ffc2bd449879ecbf9cb081bb139f5f712616/data-collection-api.ipynb

- SpaceX REST API
- Data returned in .JSON
- Clean data
- Create dataframe

Data Collection - API

Requesting rocket launch data from SpaceX API:

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

Define helper functions

```
def getLaunchSite(data):
    for x in data['launchpad']:
        response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
```

```
def getBoosterVersion(data):
```

```
def getPayloadData(data):
```

```
def getCoreData(data):
```

Normalize the data and convert into dataframe:

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

Construct dataset, create dictionary:

```
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}
```

Create Pandas data frame:

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

Data Collection - Scraping

https://github.com/luisafolle/Applied-Data-Science-Capstone-IBM---SpaceX/blob/d1c7ffc2bd449879ecbf9cb081bb139f5f712616/webscraping.ipynb

Scrape	Scrape the data	`static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"`
Request	Request page from URL	`launch_data = requests.get(static_url).text`
Create	Create BeautifulSoup object	`soup = BeautifulSoup(launch_data, 'html5lib')`
Extract	Extract all variable names from HTML table	`html_tables = soup.find_all('table')`
Create	Create dataframe	`launch_dict= dict.fromkeys(column_names)`
Export	Export to CSV	`df.to_csv('spacex_web_scraped.csv', index=False)`

Data Collection - Wrangling

https://github.com/luisafolle/Applied-Data-Science-Capstone-IBM---SpaceX/blob/d1c7ffc2bd449879ecbf9cb081bb139f5f712616/Data%20wrangling.ipynb

Launches	Determine number of launches on each site:	`df['LaunchSite'].value_counts()`
Orbit	Determine the number and occurrence of each orbit:	`df['Orbit'].value_counts()`
Landing Outcomes	Calculate the number and occurrence of mission outcome per orbit type	`landing_outcomes = df['Outcome'].value_counts()`
Bad Outcomes	Create a set of outcomes where the second stage did not land successfully	`bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])`
Create	Create list where the element is zero if the corresponding row in is in the set	see code below
Export	Export to CSV	`df.to_csv("dataset_part_2.csv", index=False)`

```
landing_class = []
for outcome in df['Outcome']:
    if outcome in set(bad_outcomes):
        landing_class.append(0)
    else:
        landing_class.append(1)
```

Data Wrangling

Number of launches on each site:

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

Out[6]: CCAFS SLC 40    55
        KSC LC 39A      22
        VAFB SLC 4E     13
        Name: LaunchSite, dtype: int64
```

EDA with Data Visualization

Scatter point charts were used to visualize:

Flight Number vs Payload Mass	Flight Number and Launch Site	Flight Number and Orbit Type	Payload and Launch Site	Payload and Orbit Type

Bar chart was used to visualize:

Success rate of each orbit

Line chart was used to visualize:

Success Yearly Trend (Launch success)

https://github.com/luisafolle/Applied-Data-Science-Capstone-IBM---SpaceX/blob/d1c7ffc2bd449879ecbf9cb081bb139f5f712616/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

https://github.com/luisafolle/Applied-Data-Science-Capstone-IBM---SpaceX/blob/a1c4d852e9e41f513e115b8bf38b510f64404bee/jupyter-labs-eda-sql-coursera%20(2).ipynb

Download	Download dataset;
Connect	Connect to database;
Display	Display unique launch sites
Display	Display total payload mass carried by boosters;
List	List first successful landing outcome in ground pad;
List	List total number of success and failure mission outcomes;
Rank	Rank landing outcomes

Build an Interactive Map with Folium

https://github.com/luisafolle/Applied-Data-Science-Capstone-IBM---SpaceX/blob/a1c4d852e9e41f513e115b8bf38b510f64404bee/Launch%20Sites%20Location%20Analysis%20with%20Folium.ipynb

01

Mark all launch sites on map;

02

Mark successful (green) and failed (red) launches for each site on map;

03

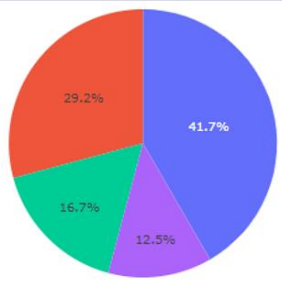
Calculate distances between launch site to its proximities;

Build a Dashboard with Plotly Dash

https://github.com/luisafolle/Applied-Data-Science-Capstone-IBM---SpaceX/blob/a1c4d852e9e41f513e115b8bf38b510f64404bee/dash_interactivity.py

Pie-charts and scatterplots were used to perform interactive visual analytics on SpaceX launch data in real-time;

1. Add a Launch Site Drop-down Input Component;
2. Add a callback function to render pie chart based on selected site dropdown;
3. Add a Range Slider to Select Payload;
4. Add a callback function to render the scatter plot (payload);

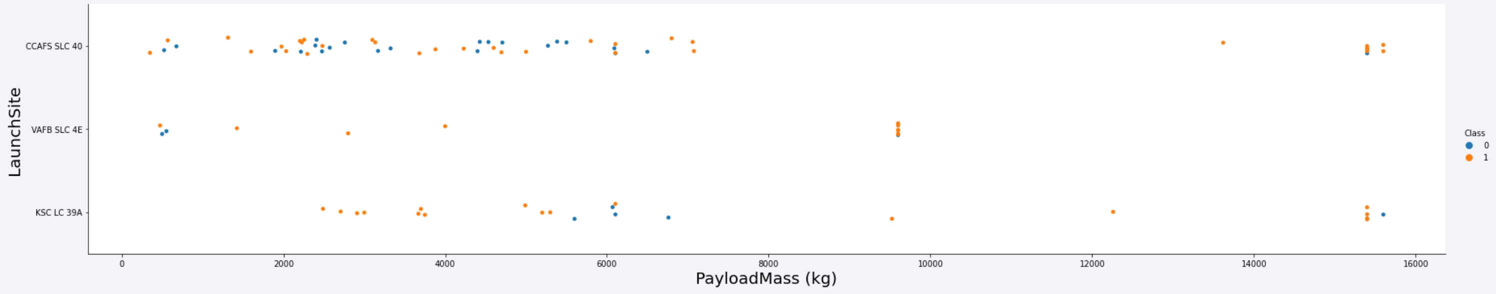


Predictive Analysis (Classification)

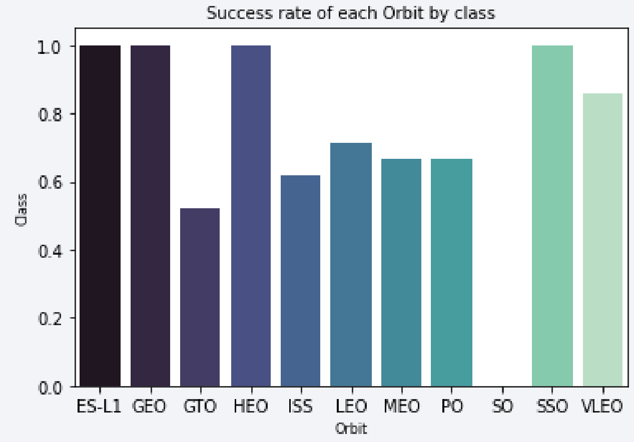
Load	Create	Standardize	Split	Objects	Accuracy
Load the dataframe	Create a NumPy array from the column Class in datafra me	Standardize the data in X then reassign it to the variable X	split the data X and Y into training and test data	Create logistic regression, SVM, decision-tree, KNN objects	Calculate the accuracy on the test data

https://github.com/luisafolle/Applied-Data-Science-Capstone-IBM---SpaceX/blob/a1c4d852e9e41f513e115b8bf38b510f64404bee/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

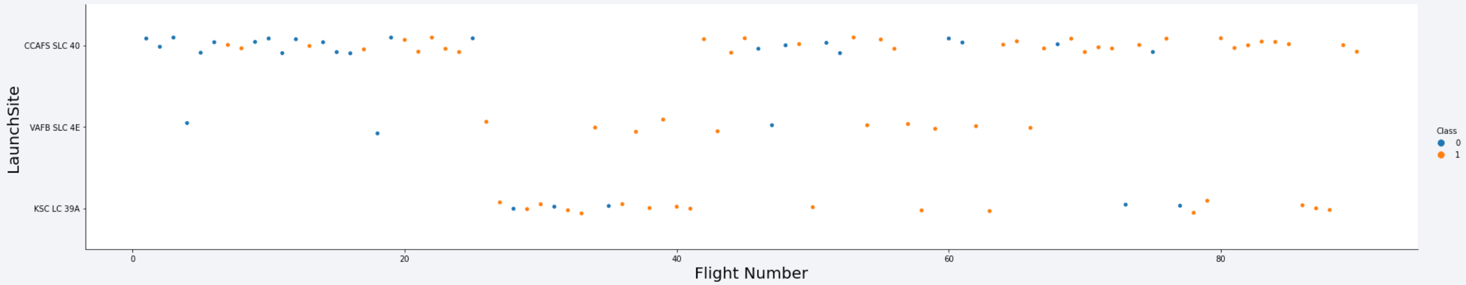
Results



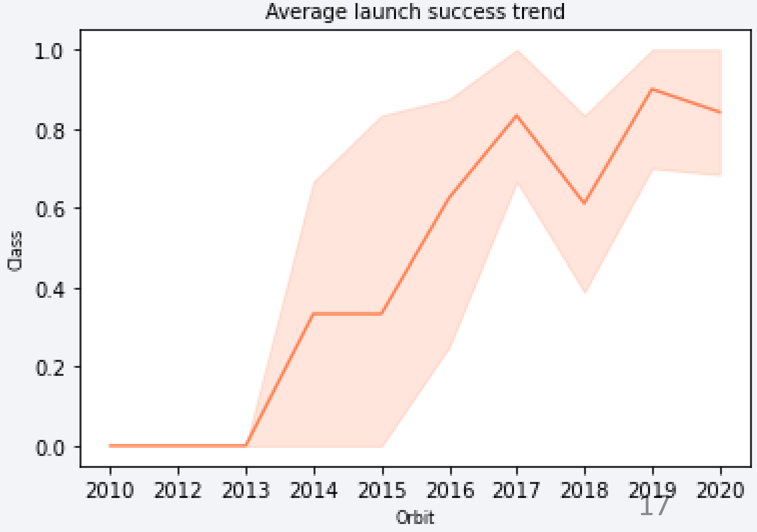
Launch Site KSC LC 39A does better with payloads up to 6000 kg, whereas CCAFS SLC 40 does better with heavier payloads (>10000kg)



Highest sucess rates: ES-L1, GEO, HEO, and SSO, followed closely by VLEO. GTO sits in the middle. SO has the least successful rates of all



As flight number increases, so does success rate



Results

```python
print('Accuracy for Logistics Regression - Test: ' + str(logreg_cv.score(X_test, Y_test)))
print('Accuracy for Logistics Regression - Train: ' + str(logreg_cv.score(X_train, Y_train)))
print('Accuracy for Support Vector Machine - Test: '+ str(svm_cv.score(X_test, Y_test)))
print('Accuracy for Support Vector Machine: '+ str(svm_cv.score(X_train, Y_train)))
print('Accuracy for Decision-Tree - Test: ' + str(tree_cv.score(X_test, Y_test)))
print('Accuracy for Decision-Tree - Train: ' + str(tree_cv.score(X_train, Y_train)))
print('Accuracy for K-Nearest Neighbors - Test: ' + str(knn_cv.score(X_test, Y_test)))
print('Accuracy for K-Nearest Neighbors - Train: ' + str(knn_cv.score(X_train, Y_train)))
```

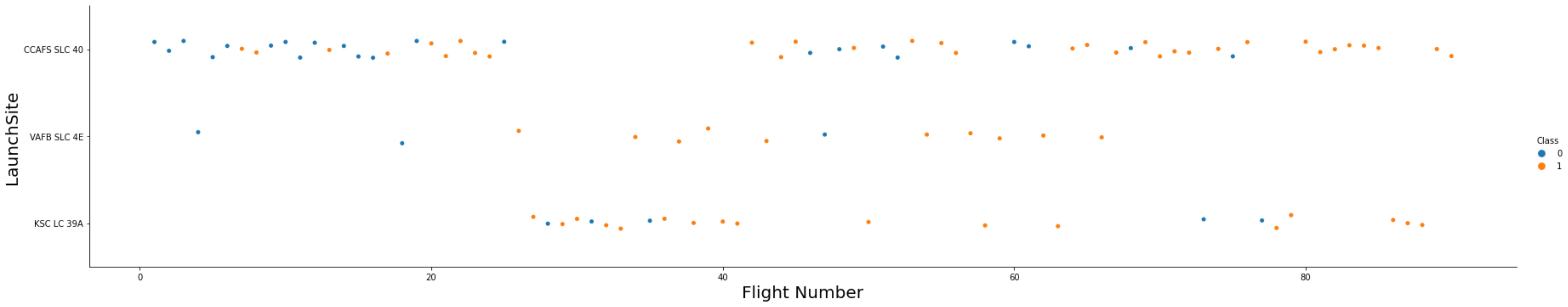
```
Accuracy for Logistics Regression - Test: 0.8333333333333334
Accuracy for Logistics Regression - Train: 0.875
Accuracy for Support Vector Machine - Test: 0.8333333333333334
Accuracy for Support Vector Machine: 0.8888888888888888
Accuracy for Decision-Tree - Test: 0.9444444444444444
Accuracy for Decision-Tree - Train: 0.8611111111111112
Accuracy for K-Nearest Neighbors - Test: 0.8333333333333334
Accuracy for K-Nearest Neighbors - Train: 0.8611111111111112
```

Scores for train and test data are very close, indicating that over-fitting was avoided. Decision-Tree method performed the best.

Section 2

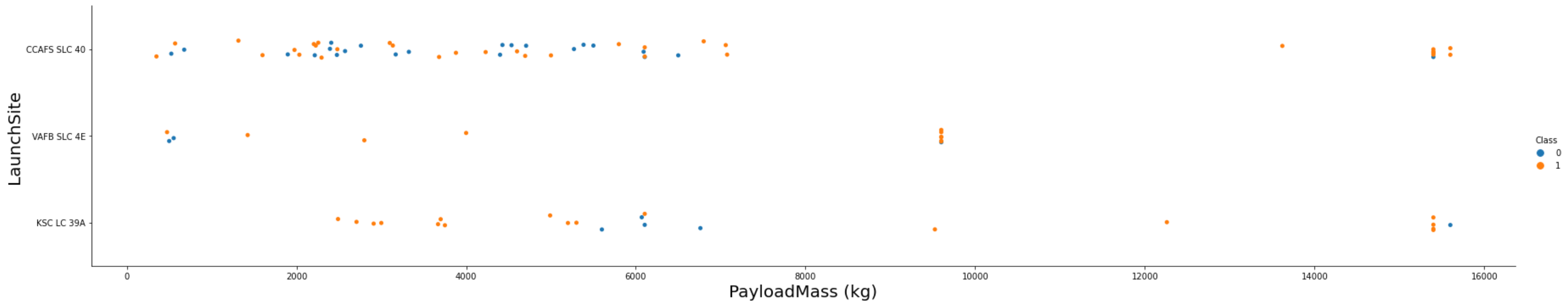
Insights drawn from EDA

Flight Number vs. Launch Site



As Flight Number increases, so does success rate.

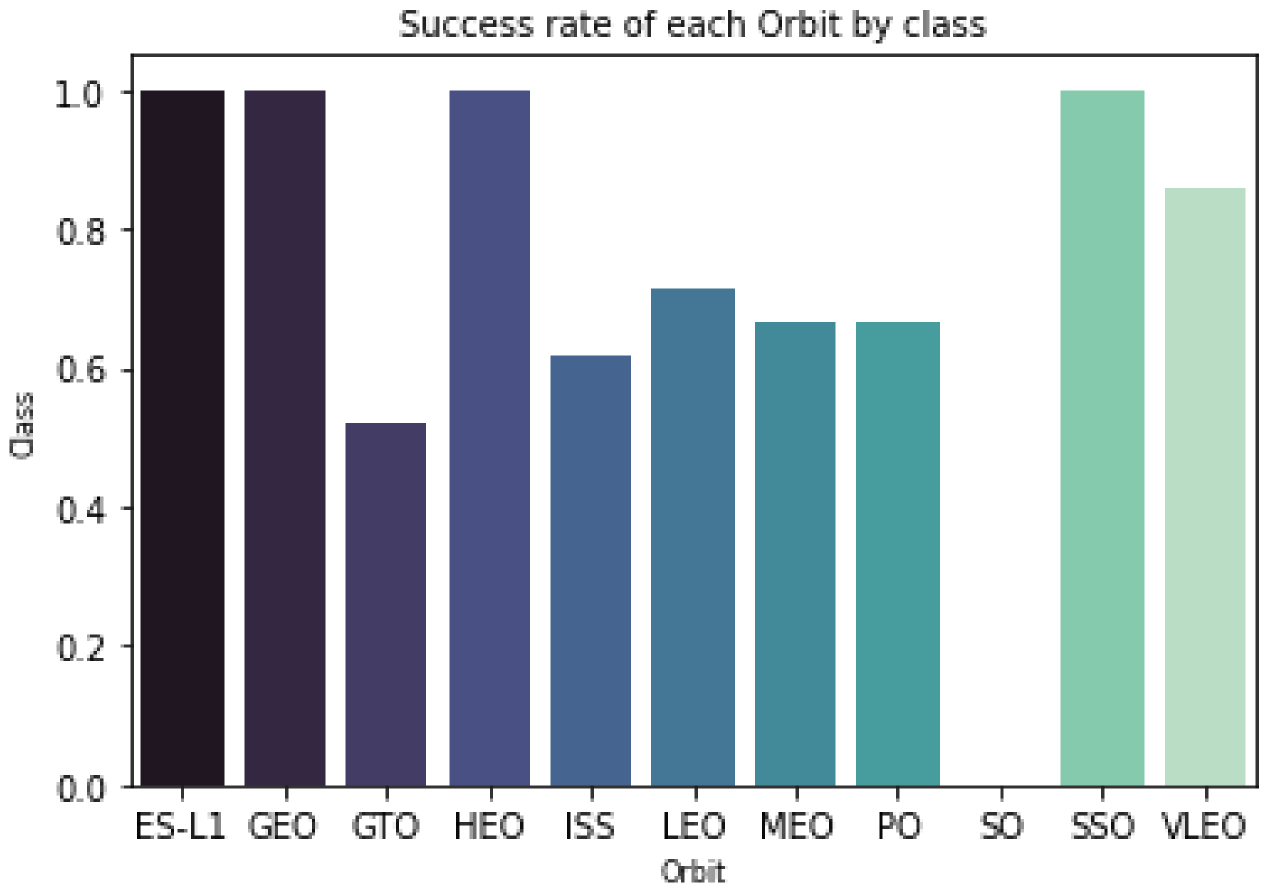
Payload vs. Launch Site



Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

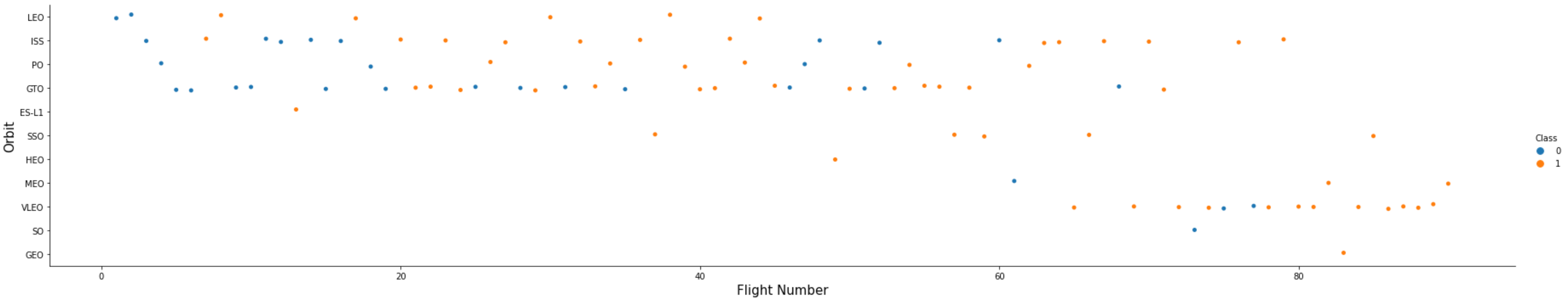
Launch Site KSC LC 39A does better with payloads up to 6000 kg, whereas CCAFS SLC 40 does better with heavier payloads (>10000kg)

Success Rate vs. Orbit Type



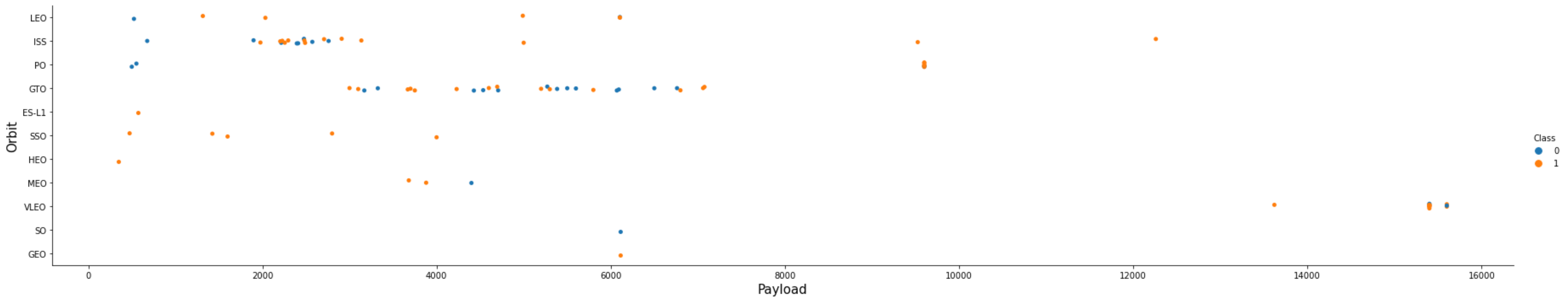
The highest sucess rates are for ES-L1, GEO, HEO, and SSO, followed closely by VLEO. GTO is in the middle, around half as the most successful ones. SO has the least successfull rates for all the Orbits.

Flight Number vs. Orbit Type



You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

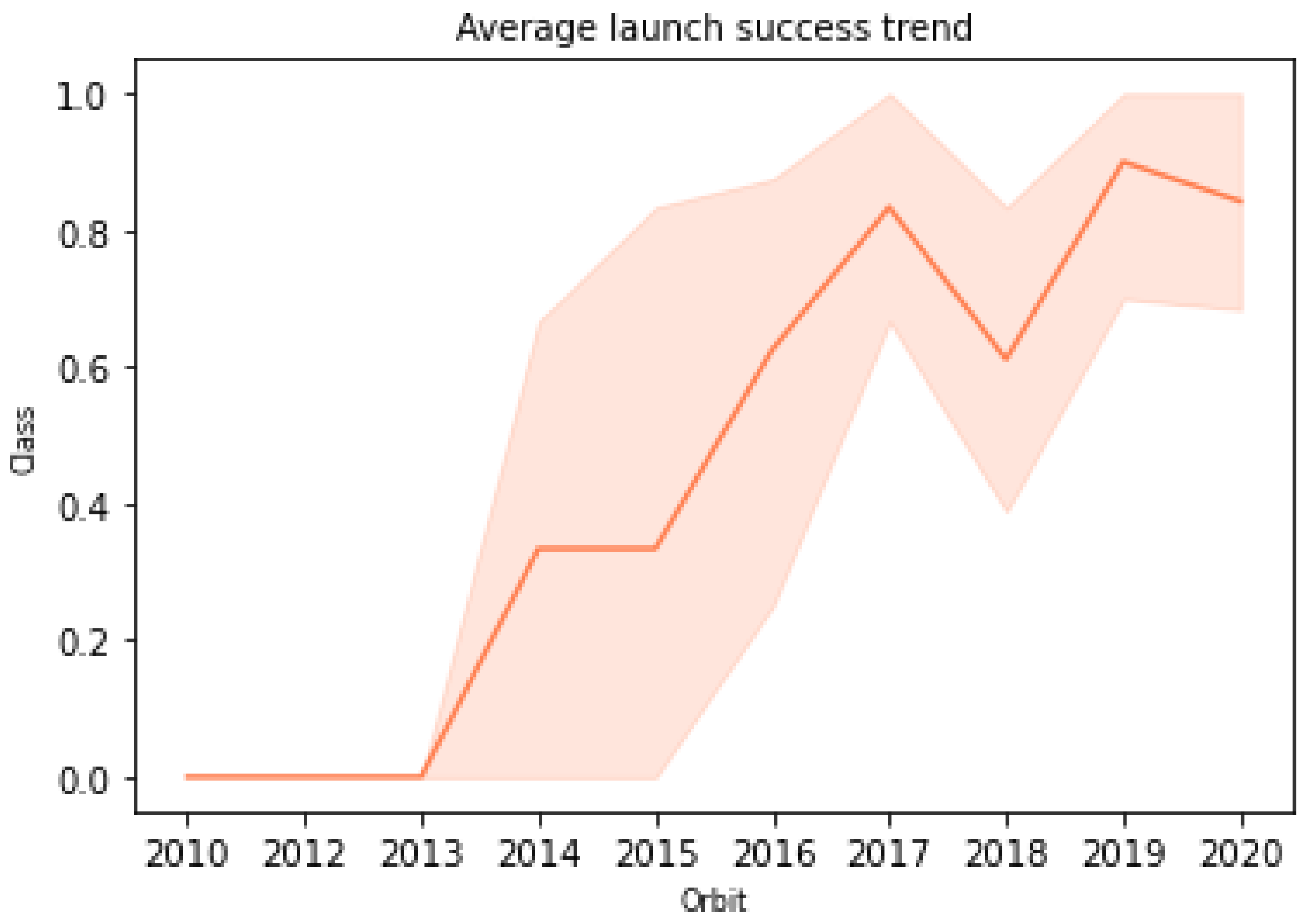
Payload vs. Orbit Type



With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend



The sucess rate starts increasing in 2013 and keeps the pace, increasing until 2020. It is stagnant in 2014-2015 and then drops 2017-2018 and again in 2020

All Launch Site Names

```python
SpaceX_df = pd.read_csv('SpaceX.csv')
pysqldf = lambda q: sqldf(q, globals())

unique_launch_site = pysqldf("SELECT DISTINCT(Launch_site) FROM SpaceX_df")
df_uls = pd.DataFrame(unique_launch_site)
df_uls
```

	Launch_Site
0	CCAFS LC-40
1	VAFB SLC-4E
2	KSC LC-39A
3	CCAFS SLC-40

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

SELECT DISTINCT will only select unique values from Launch Site

Launch Site Names Begin with 'CCA'

LIKE “CCA%” will only return values starting with those letters

```
pysqldf("SELECT *FROM SpaceX_df WHERE Launch_Site LIKE 'CCA%' LIMIT 5")
```

	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	Payload_mass_kg	Orbit	Customer	Mission_Outcome	Landing_Outcome
0	04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	08-12-2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	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
3	08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	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

```
SpaceX_df = SpaceX_df.rename({"Landing _Outcome":"Landing_Outcome","PAYLOAD_MASS__KG_":"Payload_mass_kg"}, axis='columns')
display(list(SpaceX_df.columns.values))
```

```
['Date',
 'Time (UTC)',
 'Booster_Version',
 'Launch_Site',
 'Payload',
 'Payload_mass_kg',
 'Orbit',
 'Customer',
 'Mission_Outcome',
 'Landing_Outcome']
```

```
pysqldf("SELECT SUM (Payload_mass_kg) AS Total_payload_mass FROM SpaceX_df WHERE Customer LIKE 'NASA (CRS)' ")
```

	Total_payload_mass
0	45596

Total payload mass from all NASA

Average Payload Mass by F9 v1.1

```
pysqldf("SELECT AVG (Payload_mass_kg) AS Avg_payload_mass, Booster_Version FROM SpaceX_df WHERE Booster_Version LIKE 'F9 v1.1'")
```

	Avg_payload_mass	Booster_Version
0	2928.4	F9 v1.1

Calculates the average of payload mass for booster version selected with the LIKE clause

First Successful Ground Landing Date

```
pysqldf("SELECT MIN(Date) AS First_success_landing, Launch_site FROM SpaceX_df WHERE Landing_Outcome LIKE 'Success (ground pad)'
```

	First_success_landing	Launch_Site
0	$\boxed{01-05-2017}$	KSC LC-39A

First successful landing outcome on ground pad was on 01-05-2017 on Launch site KSC LC-39A

Successful Drone Ship Landing with Payload between 4000 and 6000

```
pysqldf("SELECT Booster_Version, Landing_Outcome, Payload_mass_kg from SpaceX_df WHERE Landing_Outcome = 'Success (drone ship)' A
```

	Booster_Version	Landing_Outcome	Payload_mass_kg
0	F9 FT B1022	Success (drone ship)	4696
1	F9 FT B1026	Success (drone ship)	4600
2	F9 FT B1021.2	Success (drone ship)	5300
3	F9 FT B1031.2	Success (drone ship)	5200

pysqldf("SELECT Booster_Version, Landing_Outcome, Payload_mass_kg from SpaceX_df WHERE Landing_Outcome = 'Success (drone ship)' AND Payload_mass_kg > 4000 AND Payload_mass_kg < 6000 ")

Boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 (WHERE AND statement)

Total Number of Successful and Failure Mission Outcomes

```
pysqldf("SELECT (SELECT COUNT(Mission_Outcome) FROM SpaceX_df where Mission_Outcome LIKE '%Success%') AS Success_Mission_Outcomes
```

	Success_Mission_Outcomes	Fail_Mission_Outcomes
0	100	1

Total of successful and failure mission outcomes using COUNT

pysqldf("SELECT (SELECT COUNT(Mission_Outcome) FROM SpaceX_df where Mission_Outcome LIKE '%Success%') AS Success_Mission_Outcomes,(SELECT Count(Mission_Outcome) FROM SpaceX_df where Mission_Outcome LIKE '%Failure%') AS Fail_Mission_Outcomes ")

Boosters Carried Maximum Payload

```
, Payload_mass_kg FROM SpaceX_df WHERE Payload_mass_kg = (SELECT MAX(Payload_mass_kg) FROM SpaceX_df) ORDER BY Booster_Version ")
```

pysqldf("SELECT Booster_Version, Payload_mass_kg FROM SpaceX_df
WHERE Payload_mass_kg = (SELECT MAX(Payload_mass_kg)
FROM SpaceX_df) ORDER BY Booster_Version ")

	Booster_Version	Payload_mass_kg
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
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

SELECT MAX will return the maximum value on that column

2015 Launch Records

```
_Version, Landing_Outcome, Launch_Site from SpaceX_df WHERE Date LIKE '%2015%' AND Landing_Outcome LIKE 'Failure (drone ship)' ")
```

pysqldf("SELECT Booster_Version, Landing_Outcome, Launch_Site from SpaceX_df WHERE Date LIKE '%2015%' AND Landing_Outcome LIKE 'Failure (drone ship)' ")

	Booster_Version	Landing_Outcome	Launch_Site
0	F9 v1.1 B1012	Failure (drone ship)	CCAFS LC-40
1	F9 v1.1 B1015	Failure (drone ship)	CCAFS LC-40

Using WHERE + LIKE + AND to find a specific date and Landing Outcome

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

```
pysqldf("SELECT COUNT(Landing_Outcome) AS Count_landing, Landing_Outcome, Launch_Site FROM SpaceX_df WHERE Date BETWEEN '04-06-20
```

pysqldf("SELECT COUNT(Landing_Outcome) AS Count_landing, Landing_Outcome, Launch_Site FROM SpaceX_df WHERE Date BETWEEN '04-06-2010' AND '20-03-2017' GROUP BY Landing_Outcome ORDER BY COUNT (Landing_Outcome) DESC ")

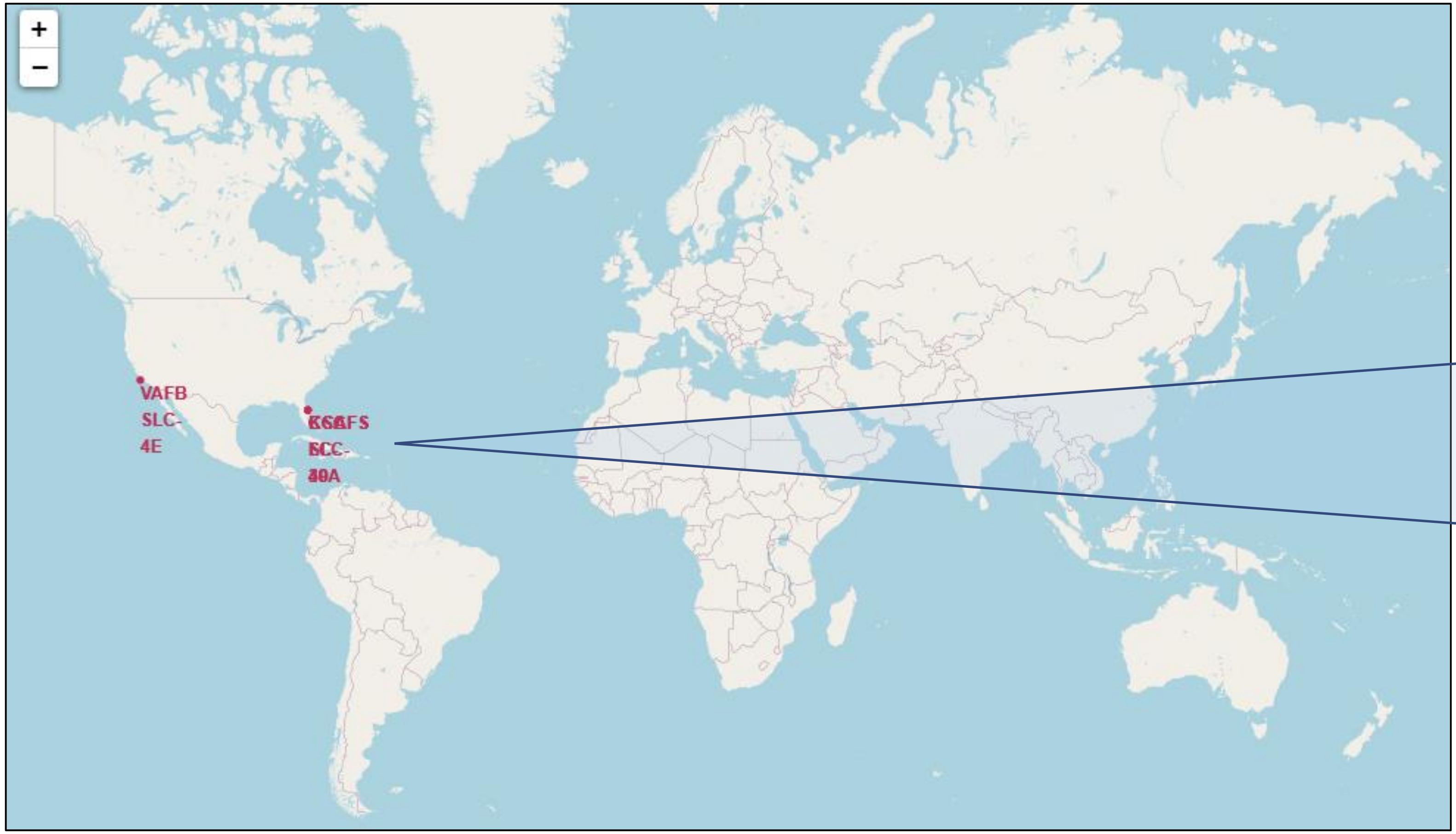
	Count_landing	Landing_Outcome	Launch_Site
0	20	Success	CCAFS SLC-40
1	10	No attempt	CCAFS LC-40
2	8	Success (drone ship)	CCAFS LC-40
3	6	Success (ground pad)	CCAFS LC-40
4	4	Failure (drone ship)	CCAFS LC-40
5	3	Failure	CCAFS SLC-40
6	3	Controlled (ocean)	CCAFS LC-40
7	2	Failure (parachute)	CCAFS LC-40
8	1	No attempt	CCAFS SLC-40

Using COUNT to sum up the outcomes between the dates specified.

Section 3

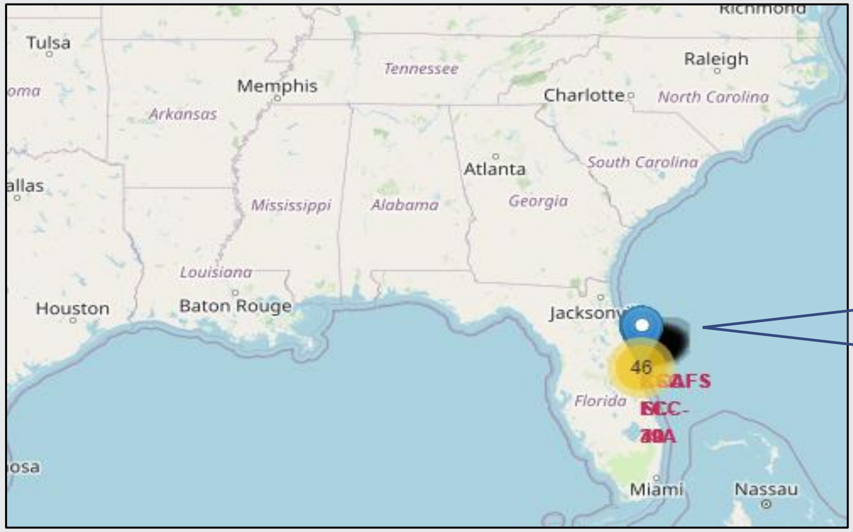
Launch Sites Proximities Analysis

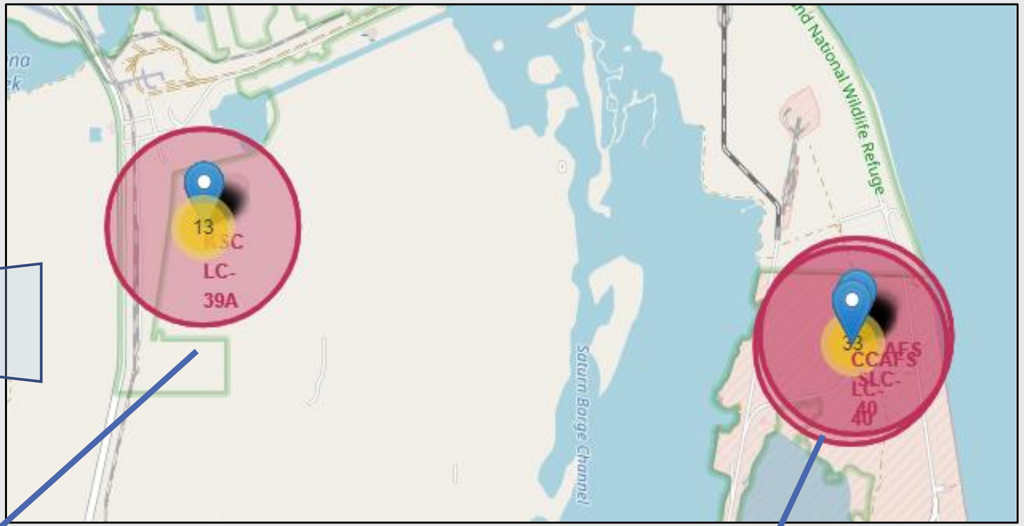
Launch Sites



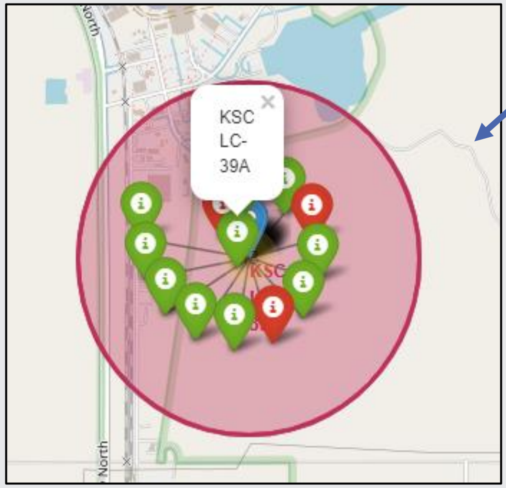
Launch sites are in the United States, close to the Equator line (get an additional natural boost) and also very close to the coast (if anything goes wrong in their ascent, it minimizes the risk to human life as it would fall in the open sea)

Marker Clusters



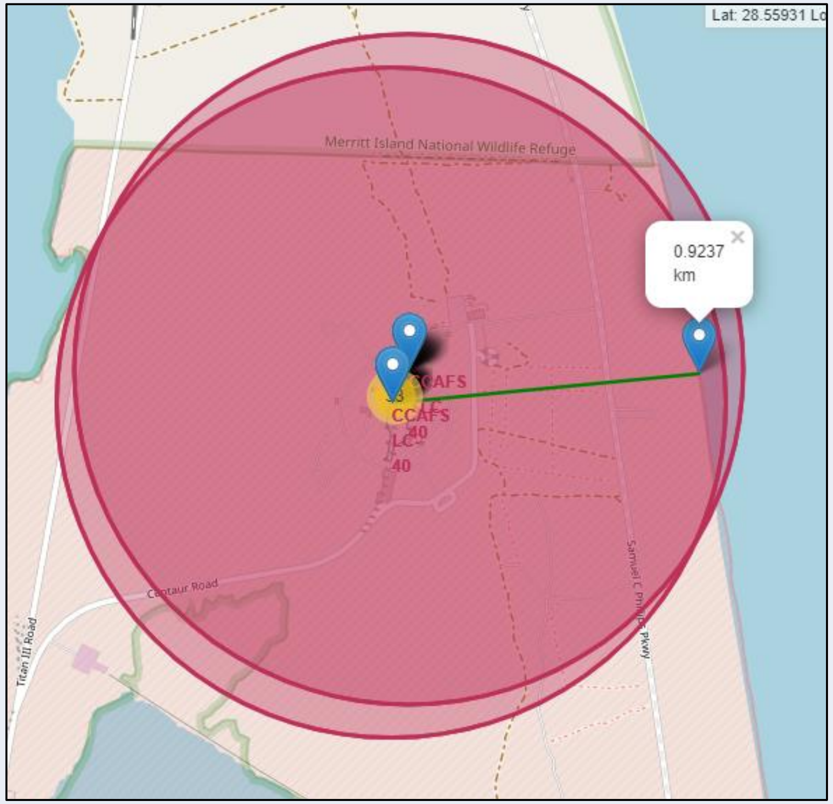


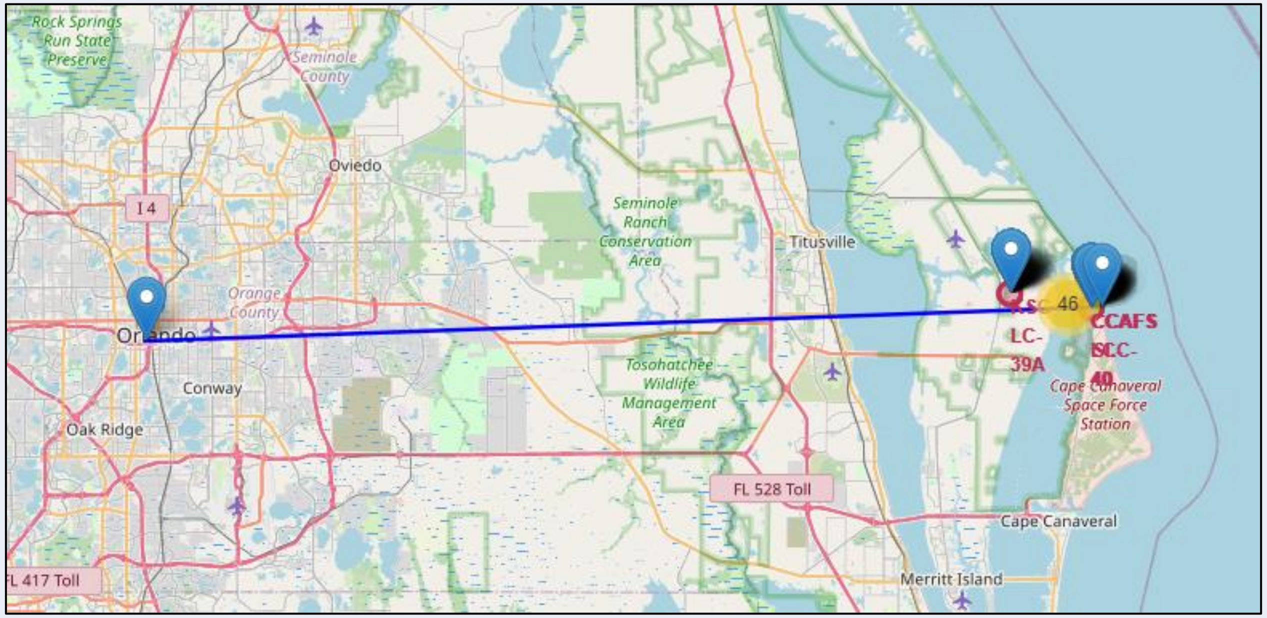
Green markers show successful outcomes



Red markers show unsuccessful outcomes

Marker Clusters – distance



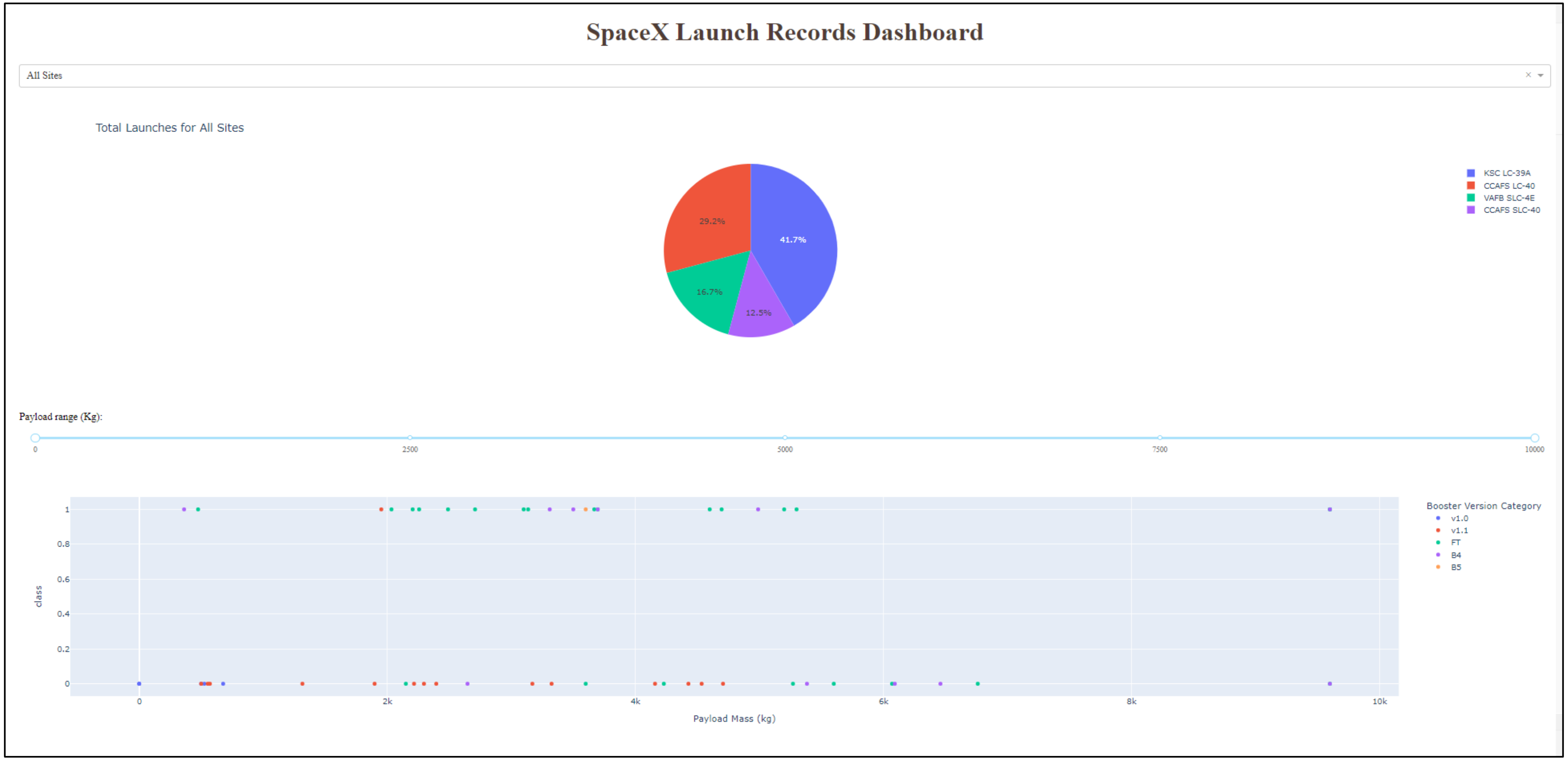


Launch Sites are close to railways, highways and coastline. Rail/highways will provide rapid and easy access to bring in raw material, equipment, etc and also provide easy way out in case finished parts need to be moved. Launch sites are close to the coastline and away from cities for if something goes wrong, debris or anything will not put people's lives at risk but fall in the ocean.

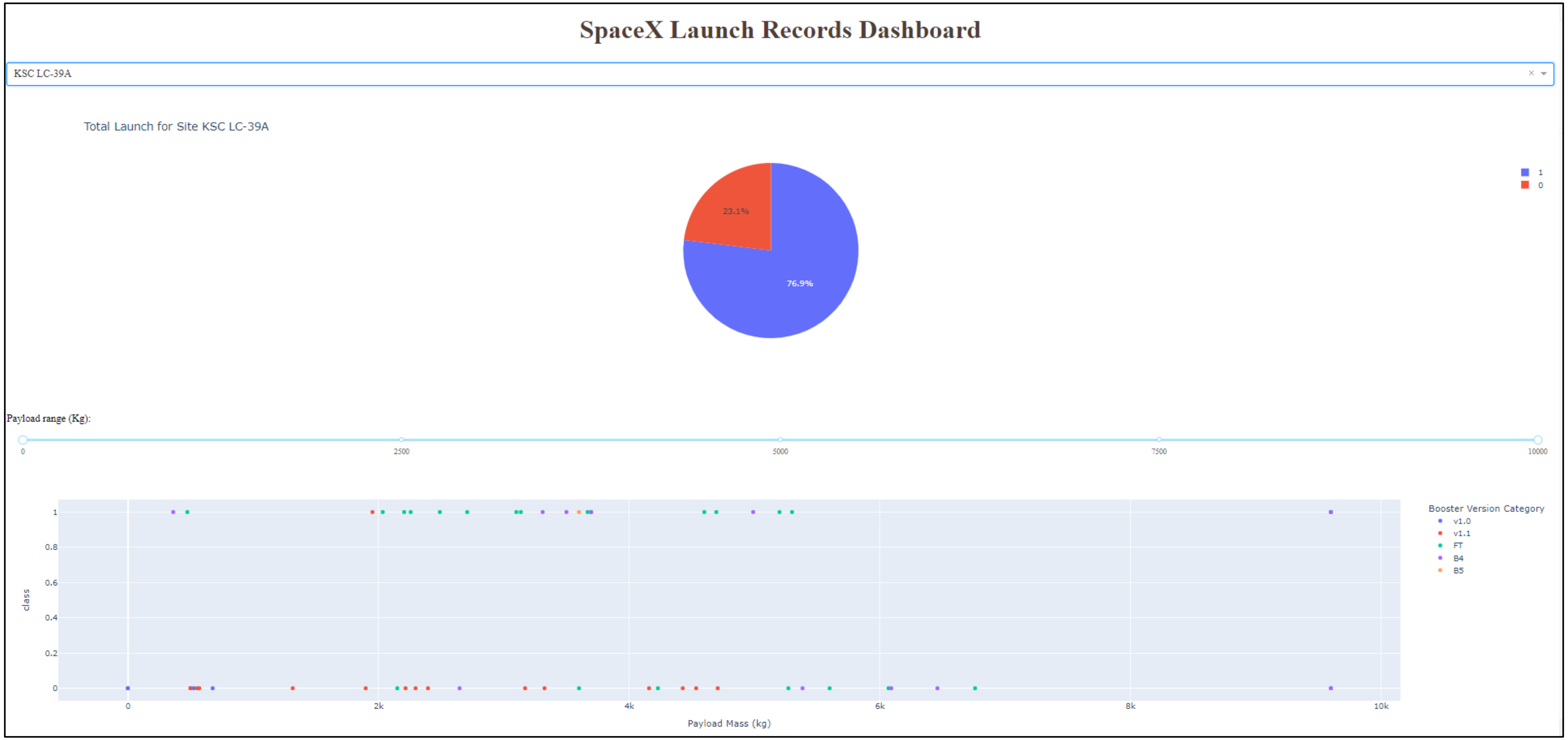
Section 4

Build a Dashboard with Plotly Dash

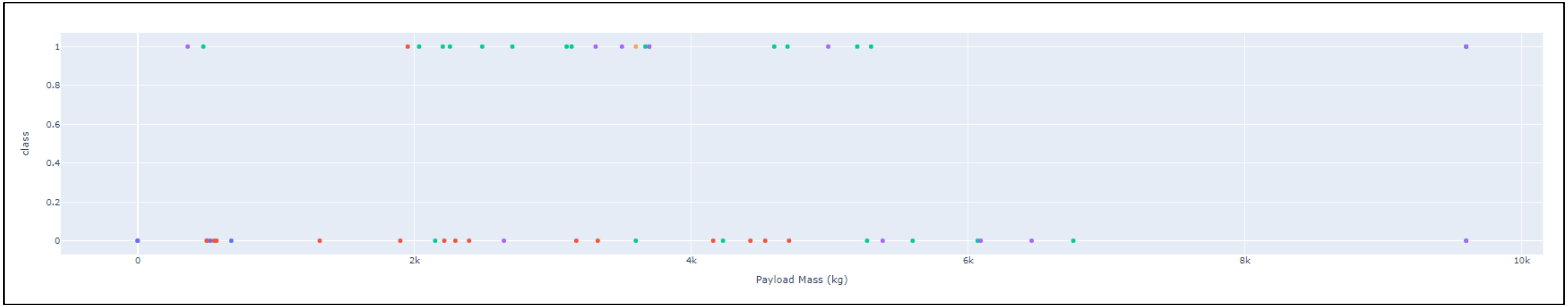
SpaceX Launch Records – All Sites



SpaceX Launch Records – Most successful



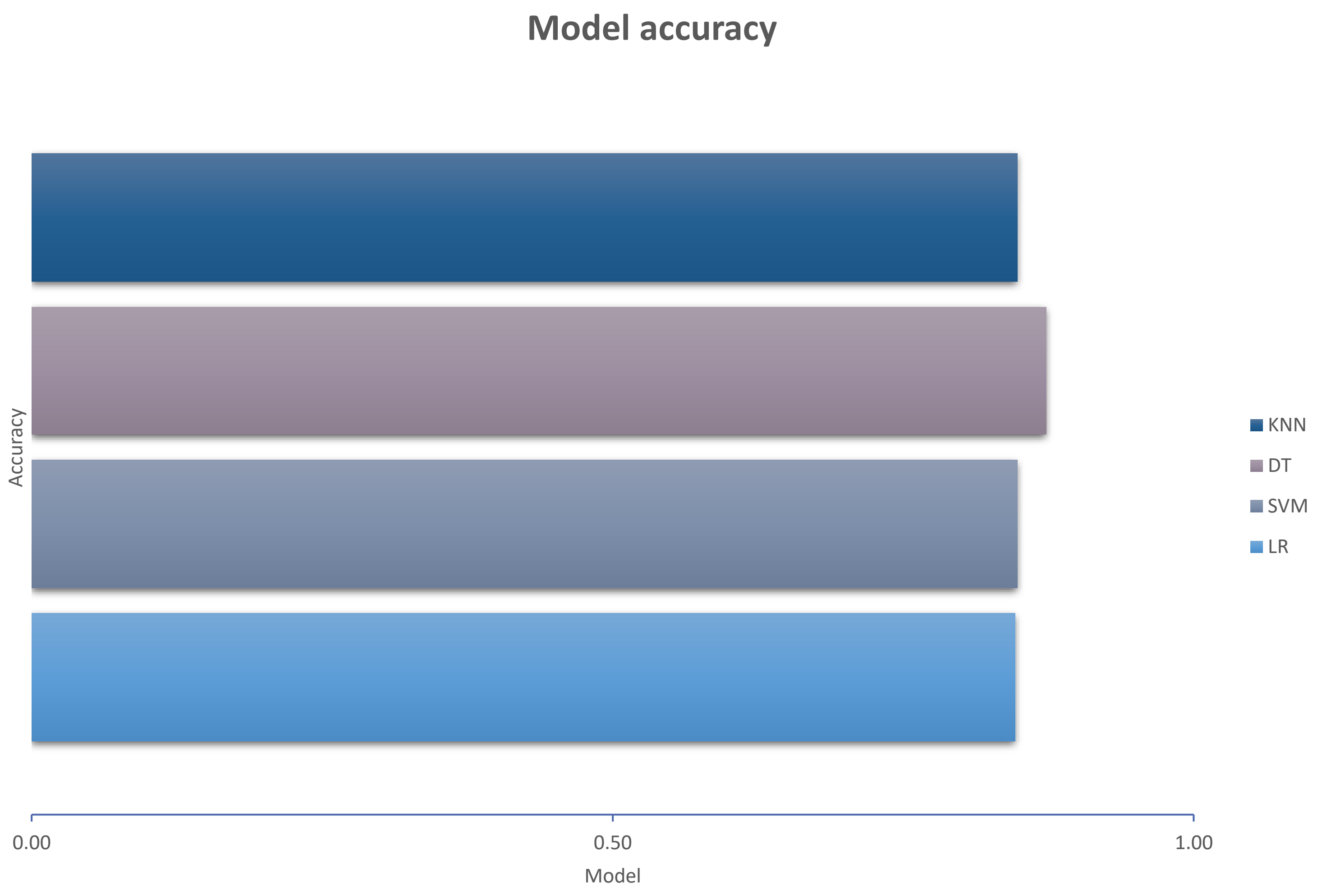
Payload vs Launch Outcome – all sites



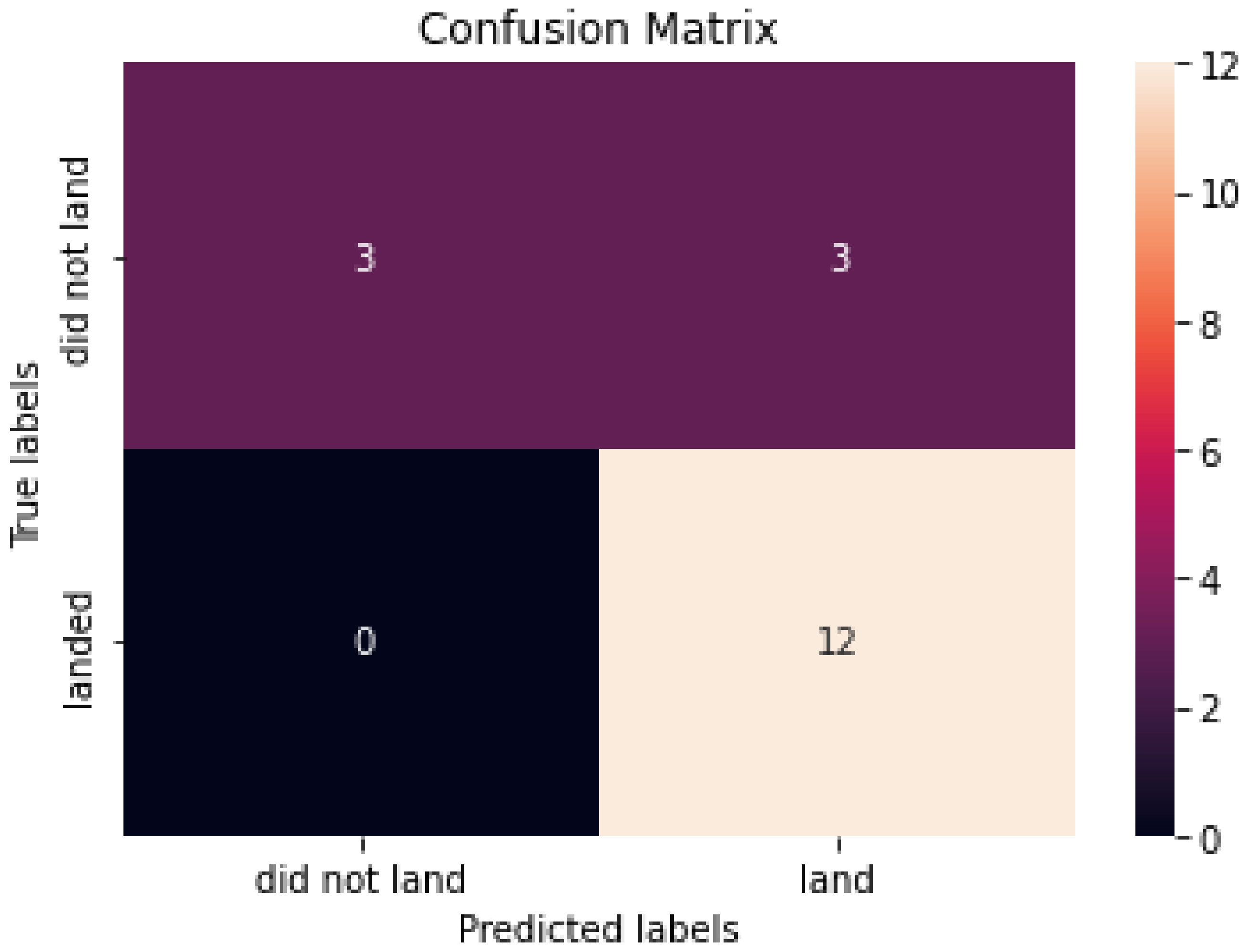
Section 5

Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix



The algorithm can distinguish between the different classes, but a problem is false positives.

Conclusions

KSC LC-39A has the most successful launches of all sites

As flight number increases, first stage is more likely to land successfully

Launch Site KSC LC 39A does better with payloads up to 6000 kg, whereas CCAFS SLC 40 does better with heavier payloads (>10000kg)

The highest sucess rates are for ES-L1, GEO, HEO, and SSO. SO has the least successful rates for all the Orbits.

For heavy payload, successful landing rate are higher for Polar, LEO and ISS Orbits.

Sucess rate starts increasing in 2013 and keeps the pace, increasing until 2020

Decision Tree is the best method for prediction

Thank you!