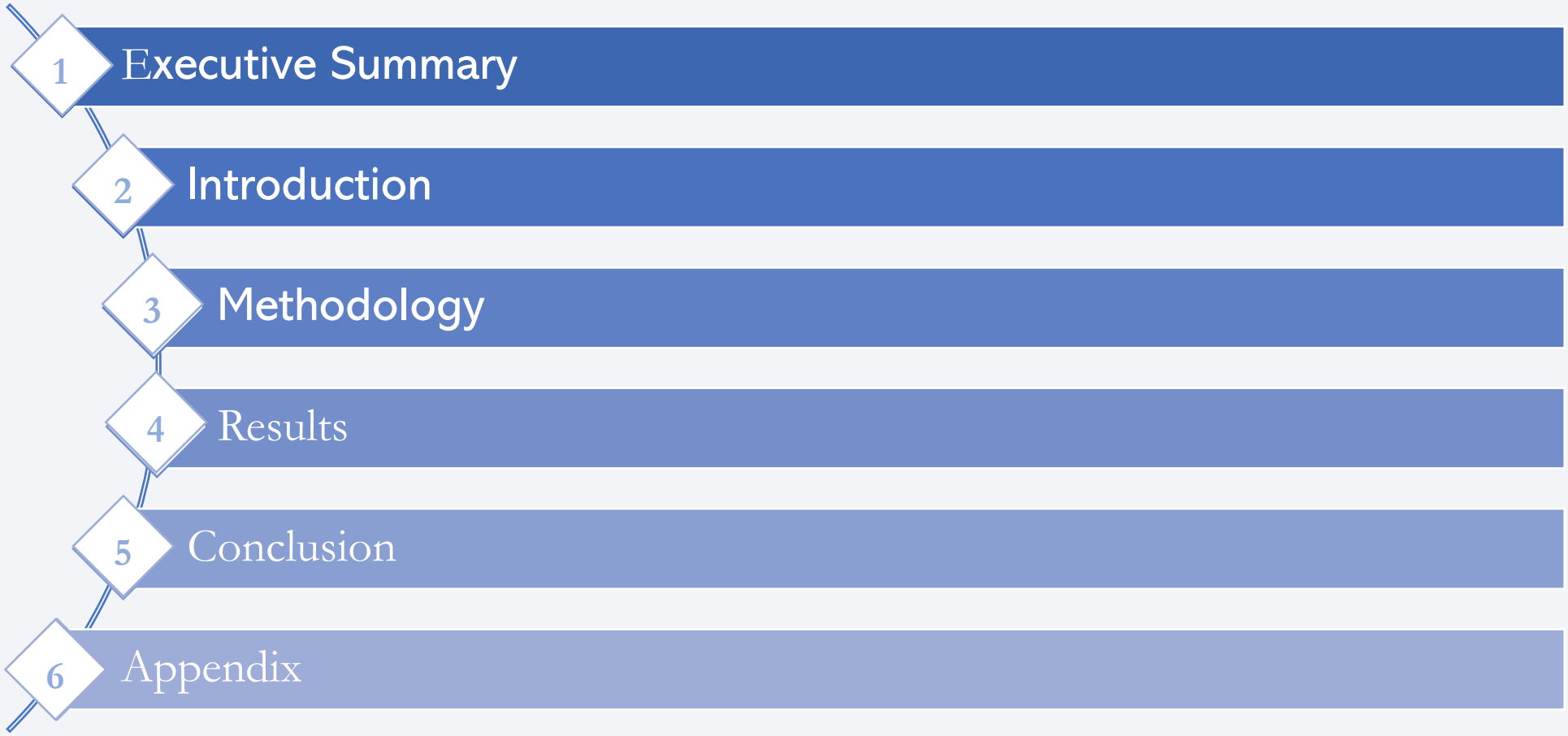
Winning Space Race with Data Science

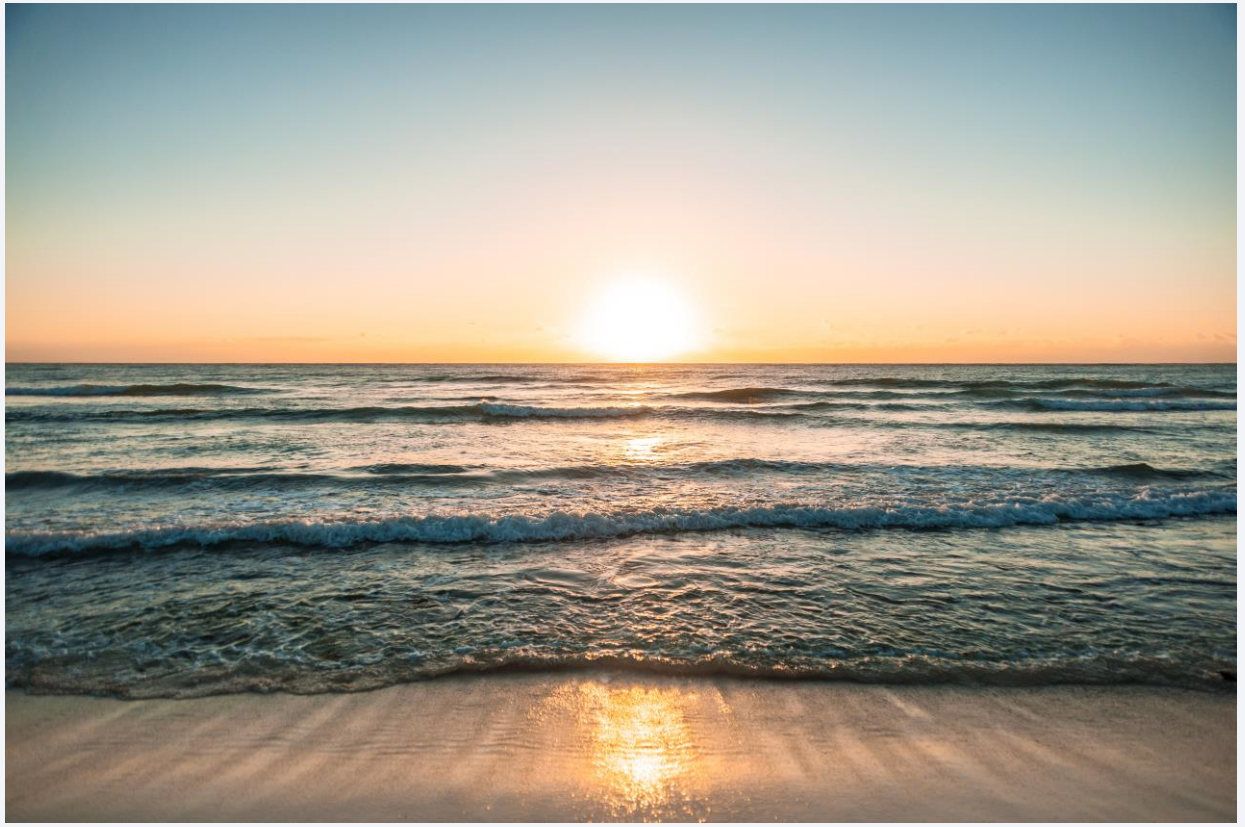
Chad Hucey
April 2022

Outline


1
Executive Summary
2
Introduction
3
Methodology
4
Results
5
Conclusion
6
Appendix

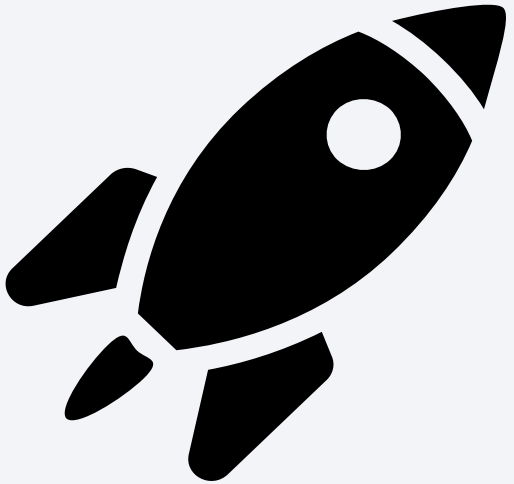
Executive Summary

- Developed a machine learning model to predict successful rocket landings
- Gathered data from SpaceX rocket launches
- Visualized the data
- Determined the sites with most successful landings
- Accurately predicted the success of future landings



Introduction

- Commercial travel to space is rapidity becoming a possibility
- SpaceX is an industry leader as the cost of rocket launches is relatively low ($62,000,000) compared to other competitors (>$150,000,000)
- SpaceX can reuse the first stage of their Falcon 9 rocket for future launches

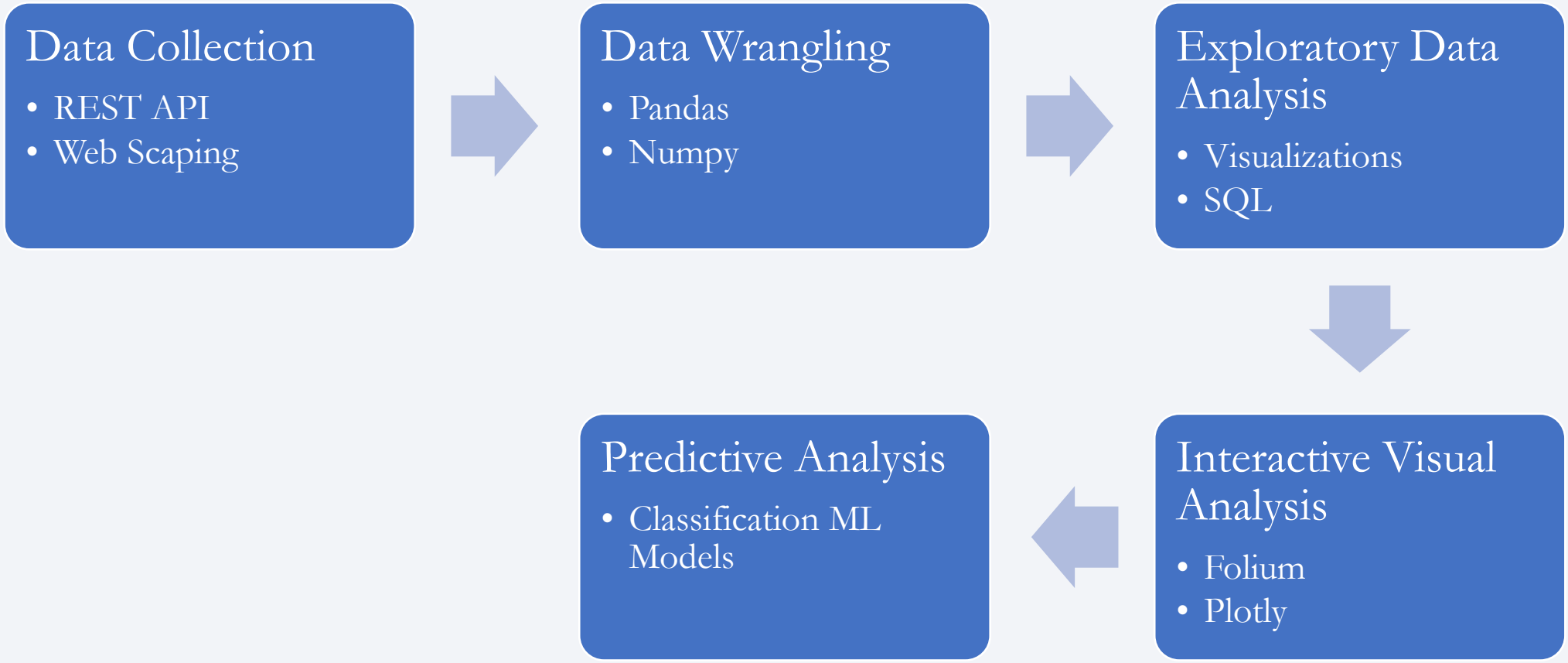


- **Is it possible to predict whether the first stage will land successfully?**
- **If we can determine if the first stage lands successfully, we can determine the cost of a launch**

Section 1

Methodology

Methodology



Data Collection

- Gathered data via:
  - REST API
    - Obtain launch data using the SpaceX API
    - Get data directly from SpaceX
  - Web Scaping
    - Obtain data from Wikipedia
    - Supplement the data obtained from spaceX

Data Collection – SpaceX API

- Sent a request to the SpaceX API
- Received a response
- Parsed and converted the results into a dataframe
- Removed unwanted columns from the dataframe
- Furthered filtered the dataframe
  - Only include falcon 9 rockets
- Click here to access my Github repository

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

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

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

```
data_falcon9 = data[data['BoosterVersion'] =='Falcon 9']
data_falcon9.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedC
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0
5	8	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0
6	10	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0
7	11	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0
8	12	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0

Data Collection - Scraping

- Requested the Falcon Launch HTML page
- Created a BeautifulSoup Object
- Parsed the data and created a dataframe

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

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

```
soup = BeautifulSoup(response, 'html.parser')
```

```
column_names = []

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

```
launch_dict= dict.fromkeys(column_names)
```

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

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX
1	2	CCAFS	Dragon	0	LEO	NASA

- Click here to access my Github repository

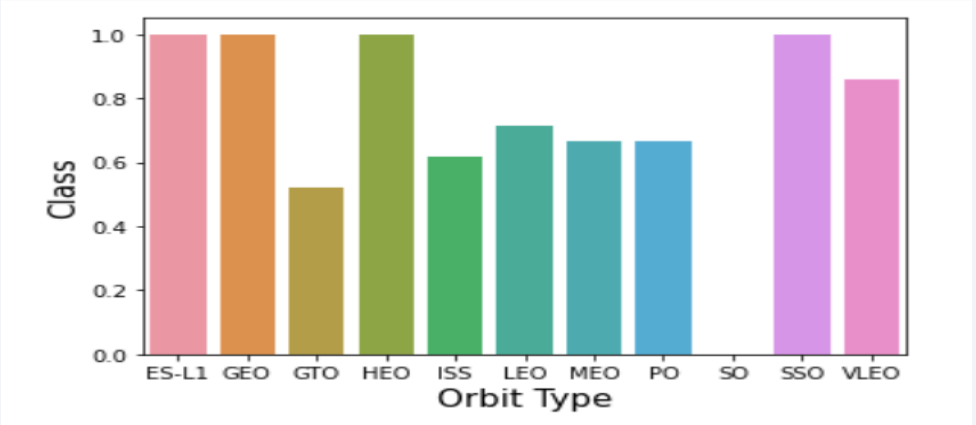
Data Wrangling

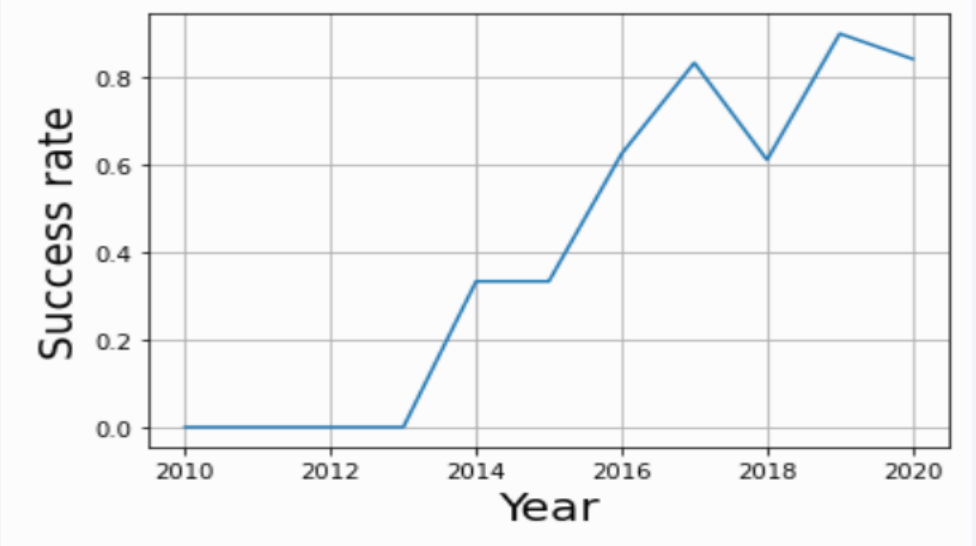
- Initial Wrangling
  - Loaded the data
  - Identified the data types in the data
  - Calculated the % of null values to total vales
- Understanding the data
  - Calculated the total number of launches per site
  - Calculated the number of occurrences of each orbit
  - Calculated the number specific landing outcomes
  - Created a ‘class’ column
    - Classify each landing outcome
      - 1 = successful landing
      - 0 = failed landing
    - Used as the target variable in the predictive model

Click here to access my Github repository

EDA with Data Visualization

- Scatter Plots
  - Flight Number vs Payload Mass
  - Flight Number vs Launch Site
  - Payload vs Launch Site
  - Flight Number vs Orbit Type
  - Payload Mass vs Orbit
- Line graph
  - Year vs success rate
- Bar chart
  - Orbit type vs landing success rate
- Click here to access my Github repository





EDA with SQL

- Found the unique launch sites
- Found the total payload mass carried by boosters launched by NASA
- Found the average payload mass carried by booster version F9 v1.1
- Listed the names of the boosters which had success in drone ship ;landings and had payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- Counted the amount of landing outcomes between the date 2010-06-04 and 2017-03-20, and sorted them in descending order
- Click here to access my Github repository

Build an Interactive Map with Folium

The geographical location of each launch site was represented using folium

- Markers were used to mark the position of the launch site on a map
- Markers were also used to identify a successful or failed landing from the launch site
- Circles were used the cluster close by markers together
- Lines were used to depict the distance from the launch site to various landmarks

Click here to access my Github repository

Build a Dashboard with Plotly Dash

- Dropdown menu
  - Allows the user to select a specific launch site
- Pie Chart
  - If the user selects “All sites” from the dropdown menu, the pie chart shows the contribution of successful landings from each launch site
  - If the user selects a specific launch site, the pie chart shows the contribution of successful and failed landings from that site
- Slider
  - Allows the user to select a payload from a range of values
- Scatter plot
  - Shows successful and failed landings differentiated by payload and booster type

Click here to access my Github repository

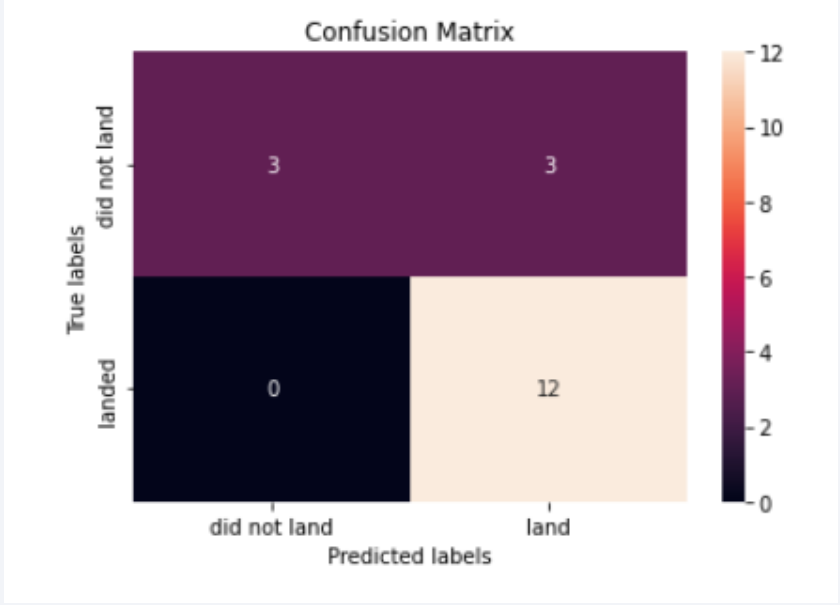
Predictive Analysis (Classification)

- Created independent (X) and target (Y) variables
  - X = orbit type, booster type, payload, flight number, etc
  - Y = class (successful or failed landings)
- Normalized the data
- Split the data into a training set and a testing set
- Built and evaluated four machine learning models using the sklearn library
  - Decision tree
  - Support Vector Machine
  - Logistic Regression
  - K Nearest Neighbour

Tuned hyperparameters of each model

Trained each model

Evaluated each model using the R^2 score

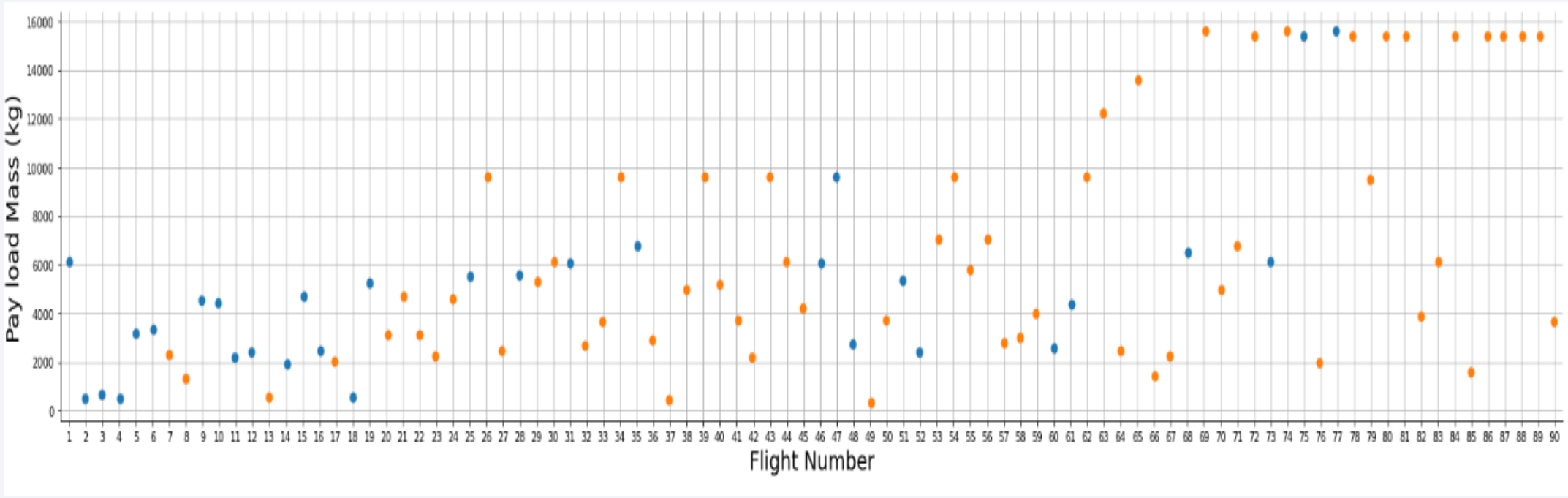


Click here to access my Github repository

Section 2

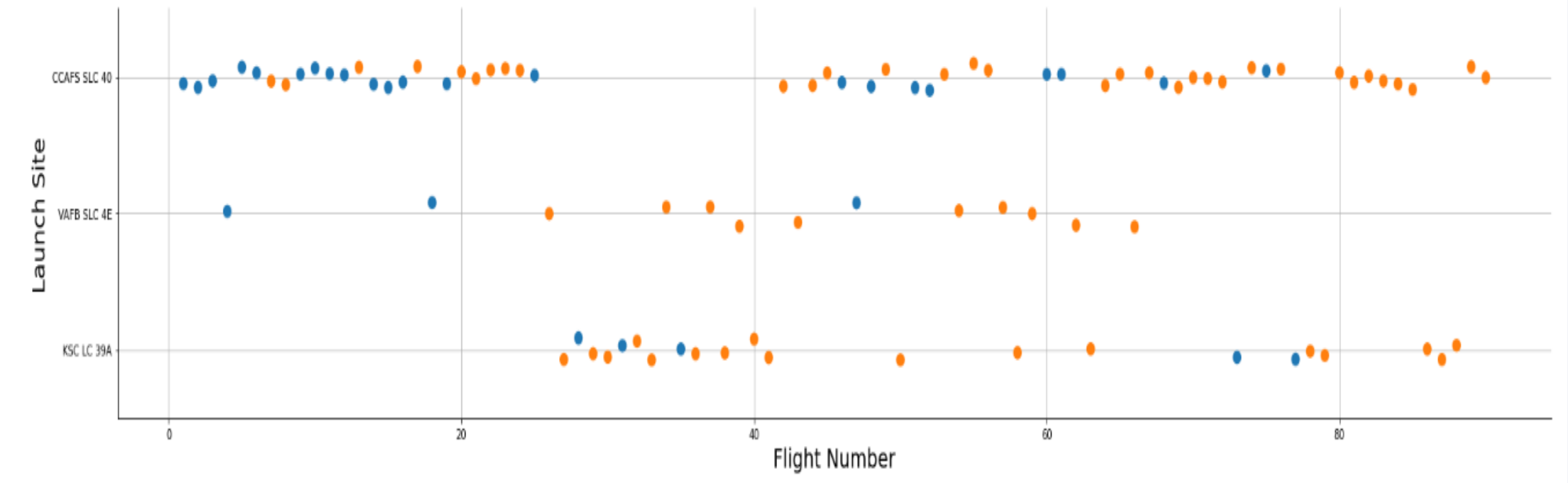
Insights drawn from EDA

Flight Number vs. Launch Site



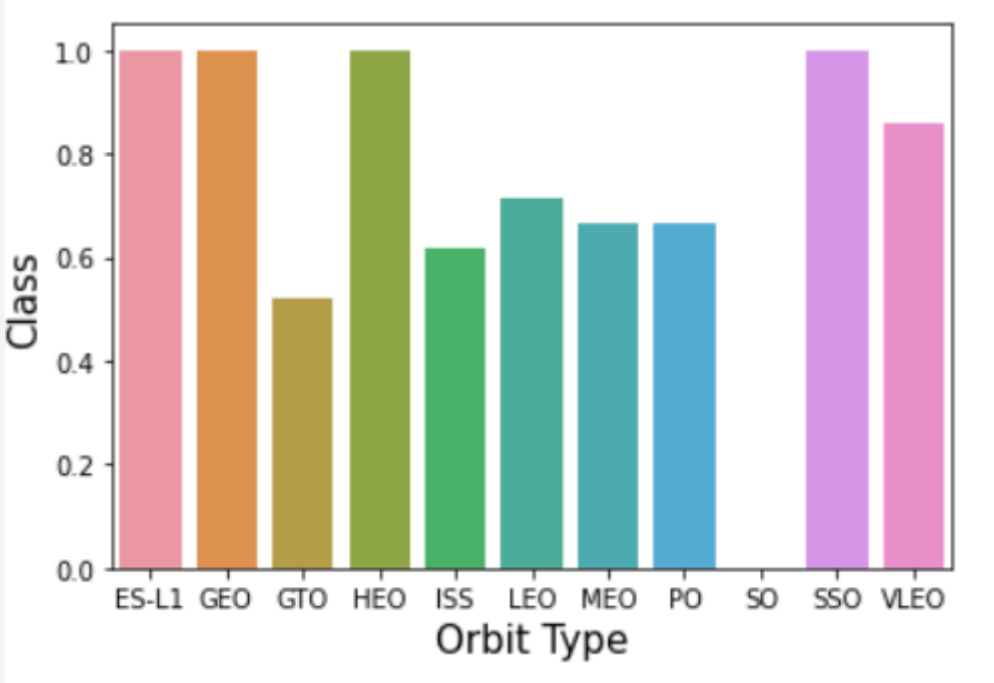
- The likelihood of a successful landing increases with flight number
- The higher the payload, the higher the likelihood of a successful landing

Payload vs. Launch Site



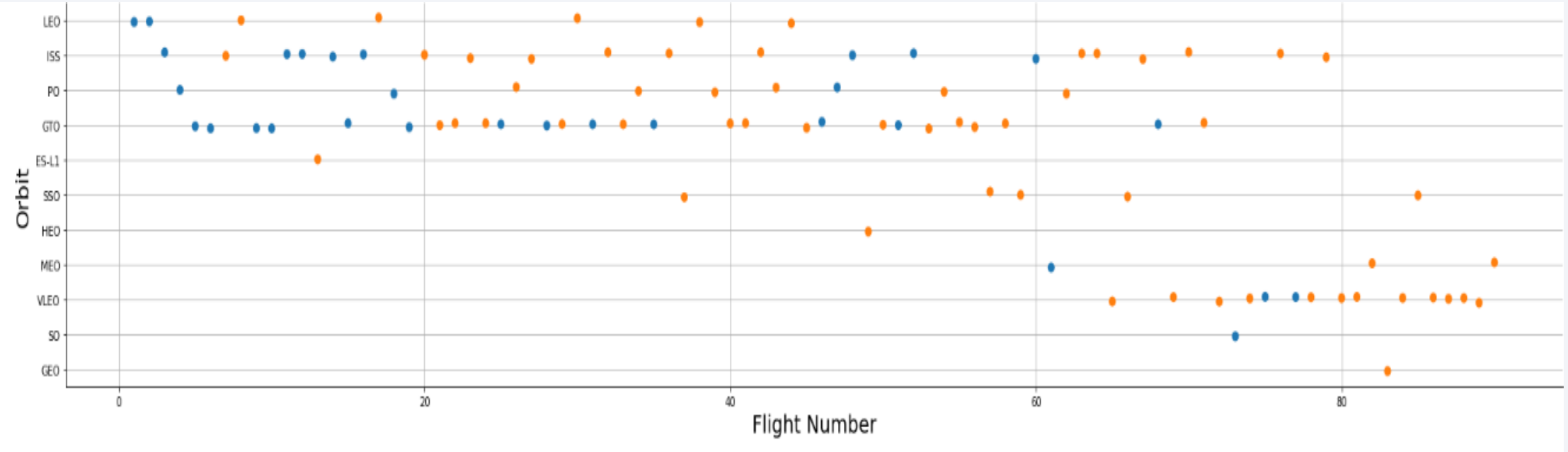
- The likelihood of a successful landing increases with flight number
- There were 3 failed landings at VAFB SLC-4E
- The most launches occurred at CCAFS SLC-40

Success Rate vs. Orbit Type

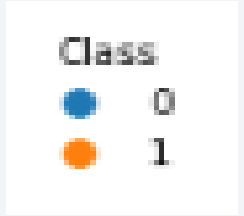


- The chart shows the avg successful landing rate vs orbit type
- ES-L1, GEO, MEO and SSO have the highest success rate
- SO has the lowest success rate
- The other orbit type have moderate to high success rates

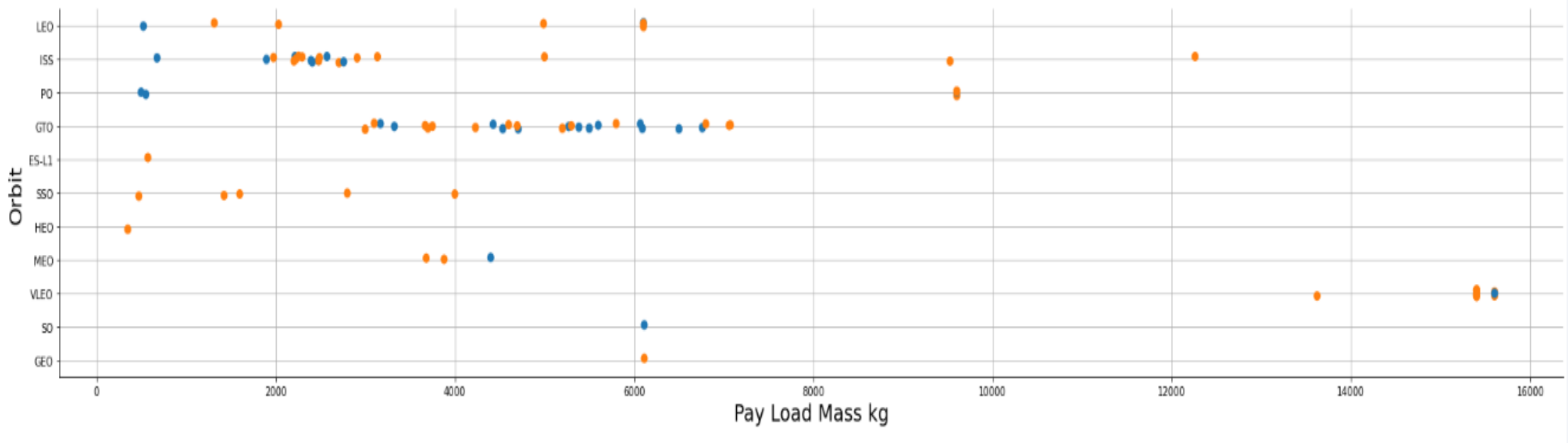
Flight Number vs. Orbit Type



- As flight number increases, the likelihood of a successful landing increases
- Low sample size but high success rate:
  - ES-L1, MEO and GEO had one launch
  - SSO had five launches

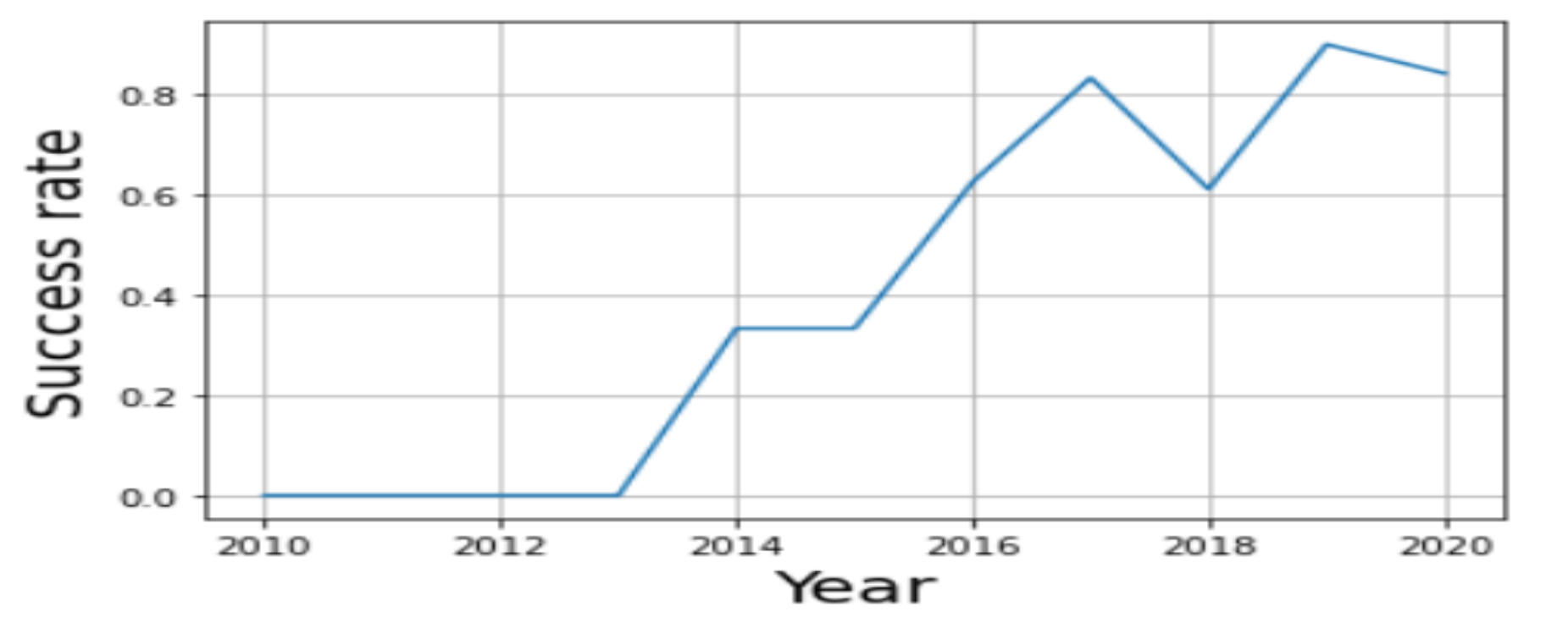


Payload vs. Orbit Type



- As payload increases, the likelihood of a successful landing increases
- Low sample size but high success rate:
  - ES-L1, MEO and GEO had one launch
  - SSO had five launches

Launch Success Yearly Trend



- Success rate increases with year

All Launch Site Names

```
%%sql
select DISTINCT LAUNCH_SITE
FROM SPACEXDATASET
```

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

- Used the “Distinct” argument so that that database return unique values

Launch Site Names Begin with 'CCA'

```
%%sql
select LAUNCH_SITE
FROM SPACEXDATASET
WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

launch_site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

- Find the launch site names that begin with 'CCA'
  - Used the “LIKE” argument to return launch sites that have “CCA” in its name
  - Refined the query by adding a”%” to “CCA” to return entries that start with “CCA”
  - Used the “LIMIT” argument to return 5 records

Total Payload Mass

```
%%sql

SELECT SUM(PAYLOAD_MASS__KG_) AS PAYLOAD_MASS_SUM
FROM SPACEXDATASET
WHERE CUSTOMER = 'NASA (CRS)';
```

payload_mass_sum
45596

- Calculate the total payload carried by boosters from NASA
  - Used the SUM() function to return the total payload
  - Used the “AS” clause to rename the output column to “payload_mass_sum”
  - Used the “WHERE” clause to filter the database and return the total payload for booster from only NASA

Average Payload Mass by F9 v1.1

```
%%sql

SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD_MASS
FROM SPACEXDATASET
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

avg_payload_mass
2928

- Calculate the average payload mass carried by booster version F9 v1.1
  - Used the AVG() function to return the average payload
  - Used the “AS” clause to rename the output column to “Avg_payload_mass”
  - Used the “WHERE” clause to filter the database and return the average payload for the “F9 v1.1 booster

First Successful Ground Landing Date

```
%%sql

SELECT MIN(DATE) AS earliest_date
FROM SPACEXDATASET
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

earliest_date
2015-12-22

- Find the date of the first successful landing outcome on ground pad
  - Used the MIN() function to return the lowest (earliest) date
  - Used the “WHERE” clause to filter the database return the earliest date of a successful ground landing

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%%sql

SELECT BOOSTER_VERSION, LANDING__OUTCOME, PAYLOAD_MASS__KG_
FROM SPACEXDATASET
WHERE LANDING__OUTCOME = 'Success (drone ship)' AND
      PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000 ;
```

booster_version	landing__outcome	payload_mass__kg_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
  - Used “SELECT” to return the booster version, landing outcome and payload columns
  - Used the “WHERE” clause to filter the database for successful drone landings and used the “AND” to add another filter
  - Used “BETWEEN” to filter for results within the specified range

Total Number of Successful and Failure Mission Outcomes

```
%%sql

SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) MISSION_OUTCOME_COUNT
FROM SPACEXDATASET
GROUP BY MISSION_OUTCOME;
```

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

- Calculate the total number of successful and failure mission outcomes
  - Used “SELECT” to return the mission outcome categories
  - Used the COUNT() function to count the number of outcomes
  - Used “GROUP BY” to aggregate each outcome

Boosters Carried Maximum Payload

```
%%sql

SELECT BOOSTER_VERSION
FROM SPACEXDATASET
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXDATASET)
```

- List the names of the booster which have carried the maximum payload mass
  - Used “SELECT” to return the booster version column
  - USED the “WHERE” clause to filter the data
  - USED a subquery within the “WHERE” clause and the MAX() function to find the boosters with the maximum payload

booster_version
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 BOOSTER_VERSION, DATE, LAUNCH_SITE, LANDING__OUTCOME
FROM SPACEXDATASET
WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND
      YEAR(DATE) = 2015;
```

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

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
  - Used “SELECT” to return the booster version, date, launch site and landing outcome columns
  - Used the “WHERE” clause the filter the database for drone ship failures
  - Used the “AND” clause to add another filter criterion
  - Used the YEAR() function with in the “WHERE” clause to filter for dates whose year was 2015

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

```
%%sql

SELECT LANDING__OUTCOME,  COUNT(LANDING__OUTCOME) AS LANDING_OUTCOME_COUNT
FROM SPACEXDATASET
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'

GROUP BY LANDING__OUTCOME
ORDER BY COUNT(LANDING__OUTCOME) DESC;
```

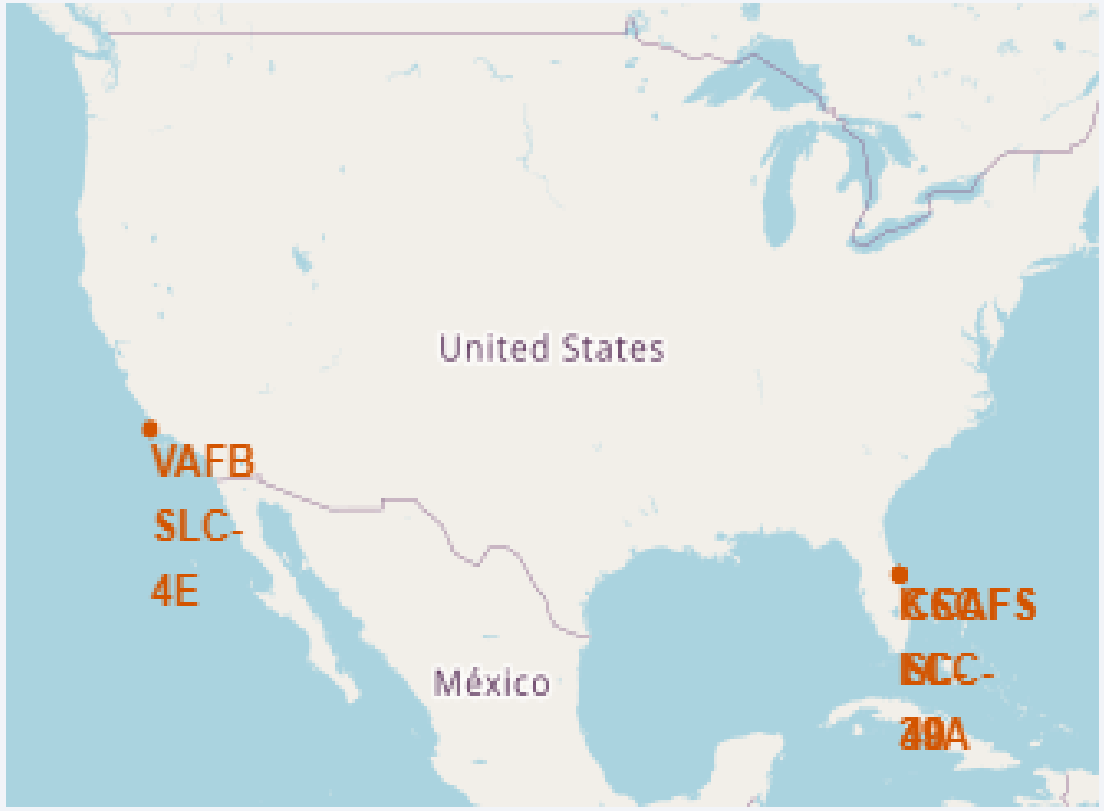
landing__outcome	landing_outcome_count
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

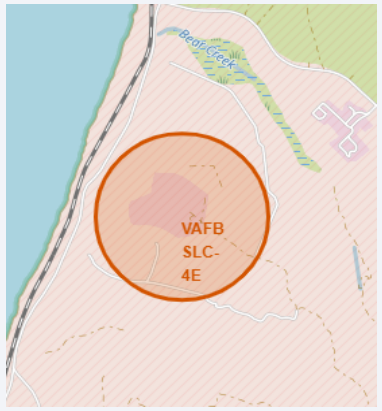
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order
  - Used “SELECT” to return the landing outcome column
  - USED the COUNT() function to count the number of landing outcomes
  - Used “WHERE” and “BETWEEN” to filter the database for dates within the specified range
  - Used “GROUP BY” to aggregate the landing outcomes
  - Used “ORDER BY” and “DESC” to sort the results in descending order

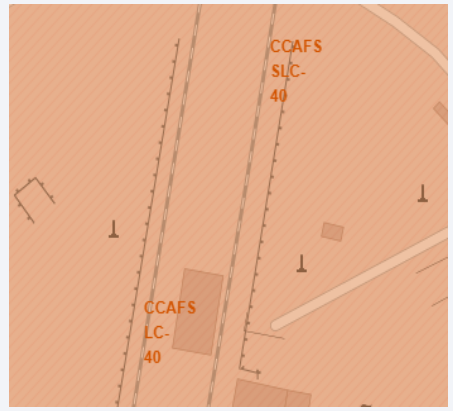
Section 3

Launch Sites Proximities Analysis

Launch Site Locations



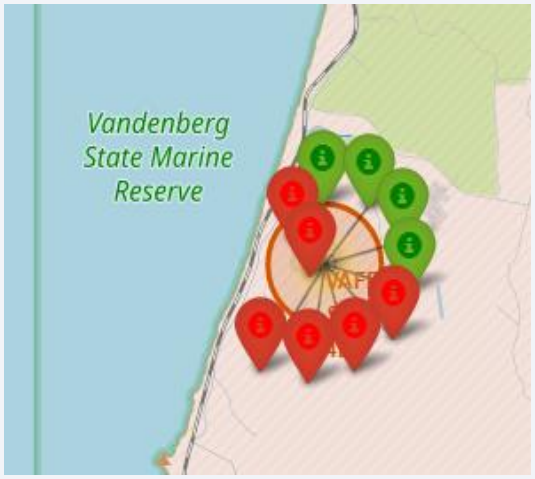




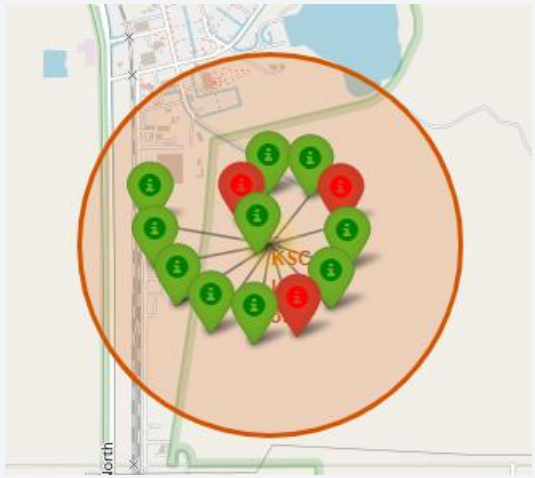
- The launch sites are located in California and Florida
- Launch sites are close to the coastline

Launch Outcomes

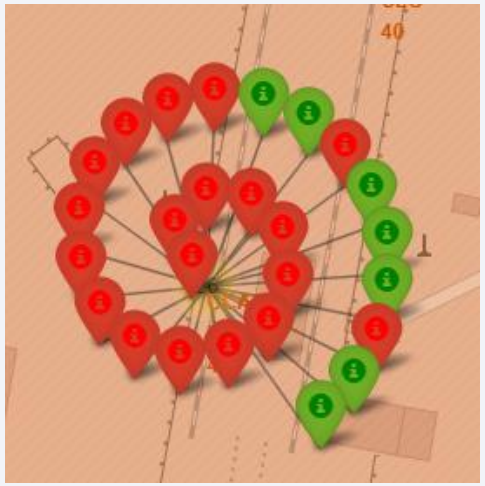
VAFB SLC-4E



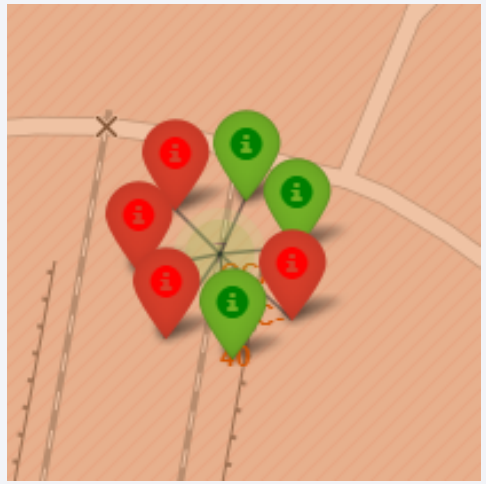
KSC LC-39A



CCAFS LC-40

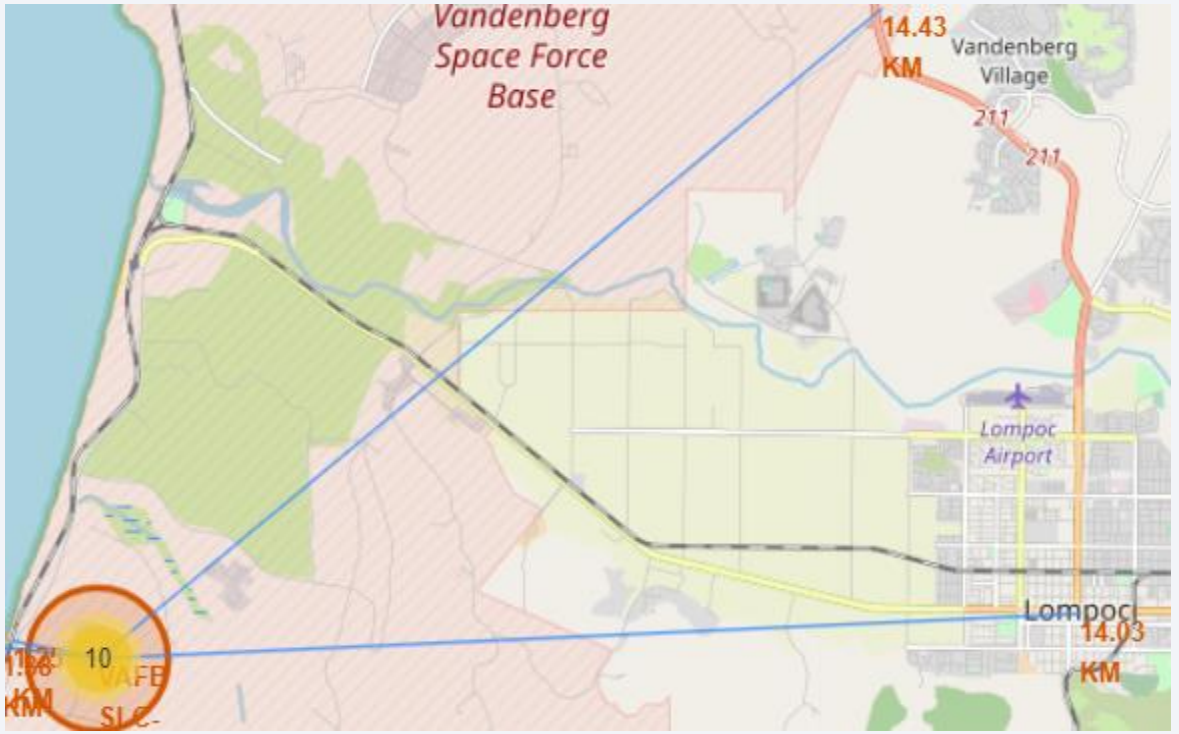


CCAFS SLC-40

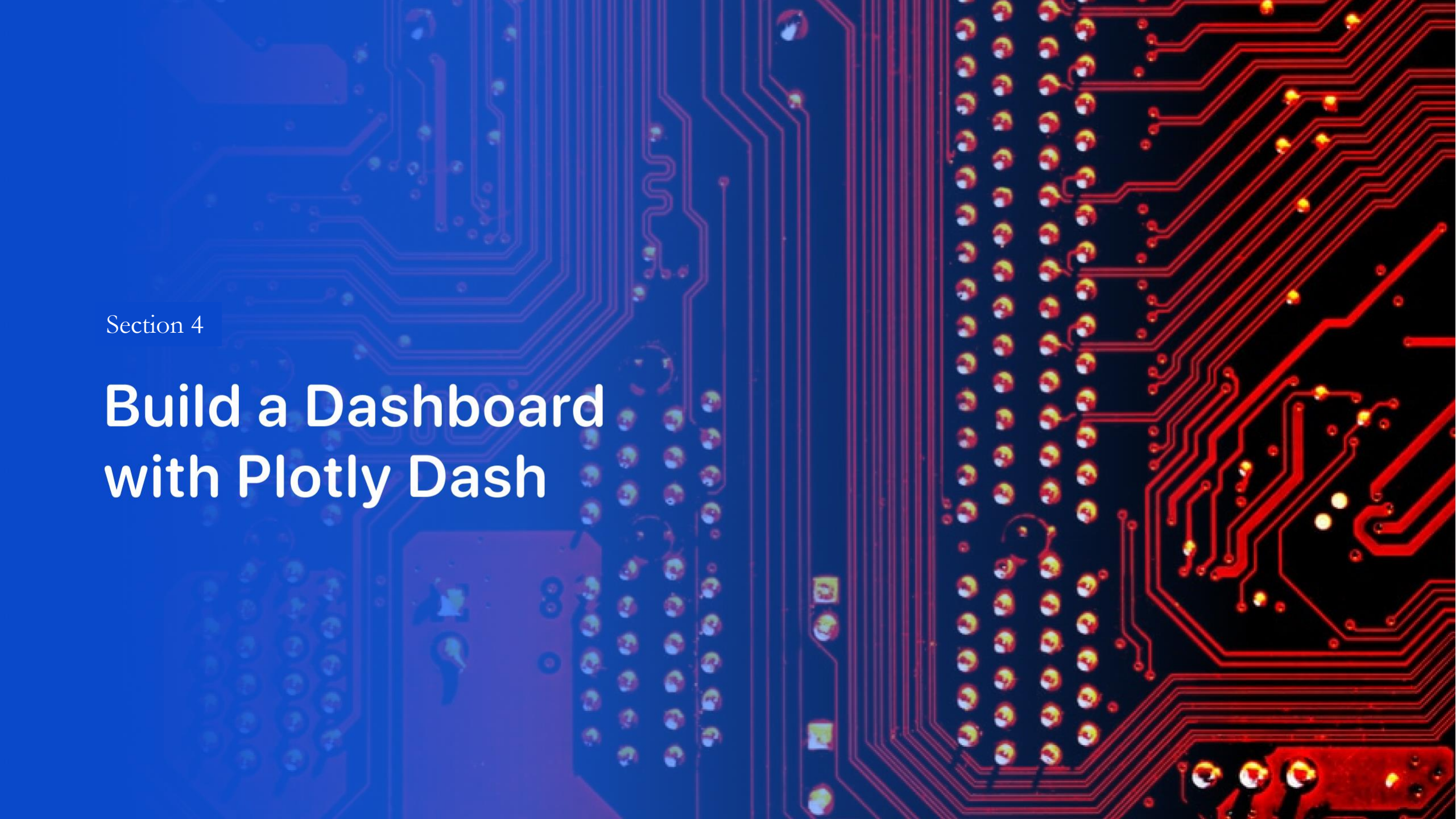


- KSC LC-39A had the highest rate of successful outcomes
- CCAFS LC-40 had the lowest rate of successful outcomes

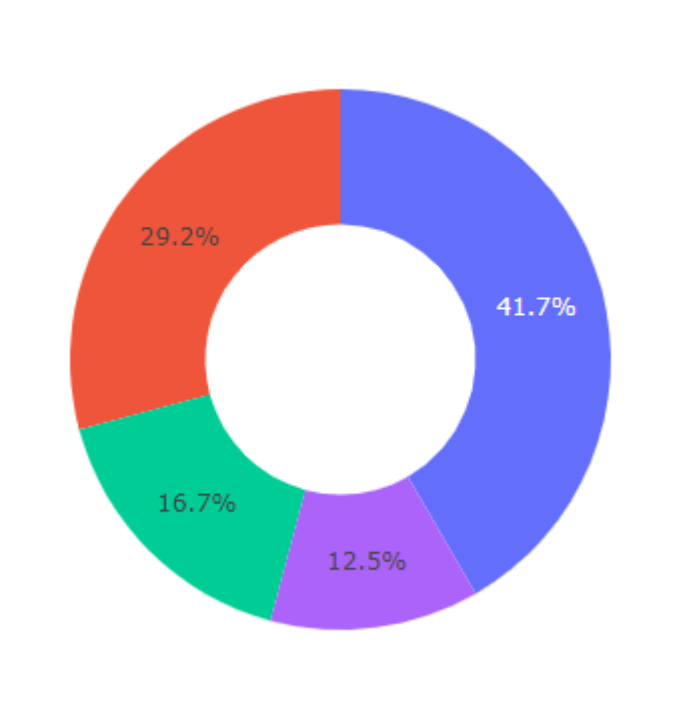
Landmark Proximity – VAFB SLC-4E



- Near to the coast and to a railway
- Far from cities and highways


Section 4
Build a Dashboard with Plotly Dash

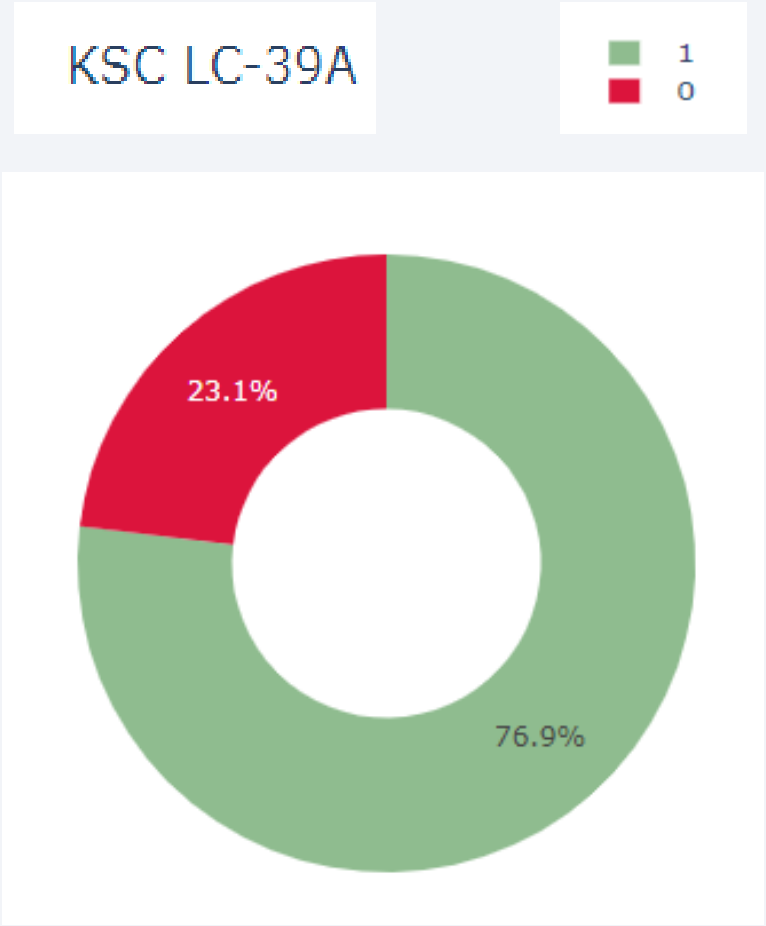
Successful Landings from All sites



Launch Site	Successful landings
KSC LC-39A	10
CCAFS LC-40	7
VAFB SLC-4E	4
CCAFS SLC-40	3

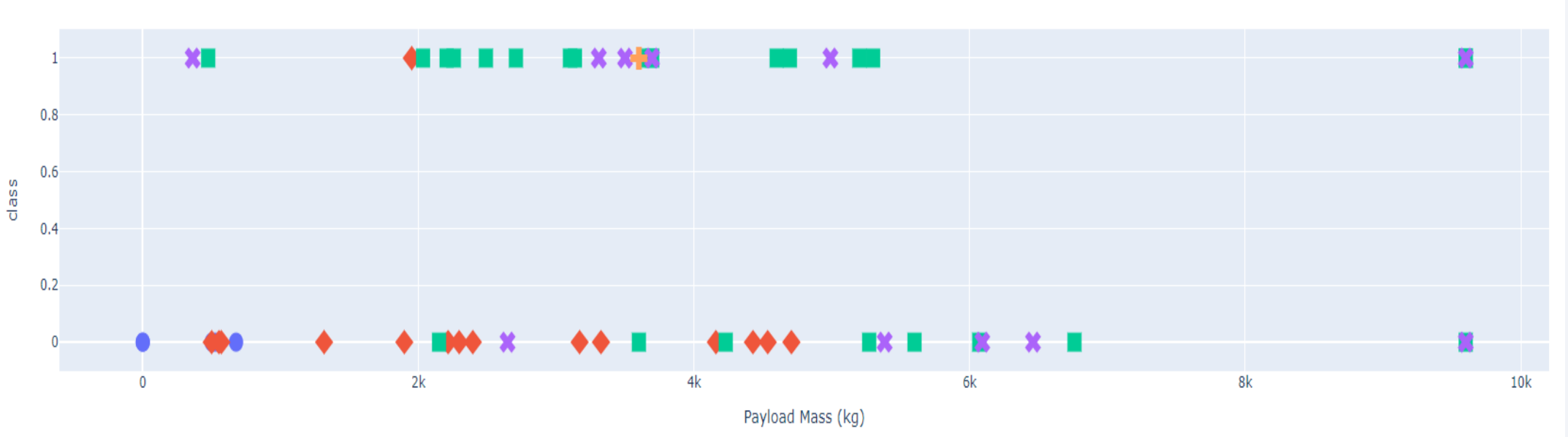
- The most successful landings were from KSC LC-39A with 10 landings
- The second most successful landings were from CCAFS LC-40 with 7 landings

Launch site with the Highest Success Rate



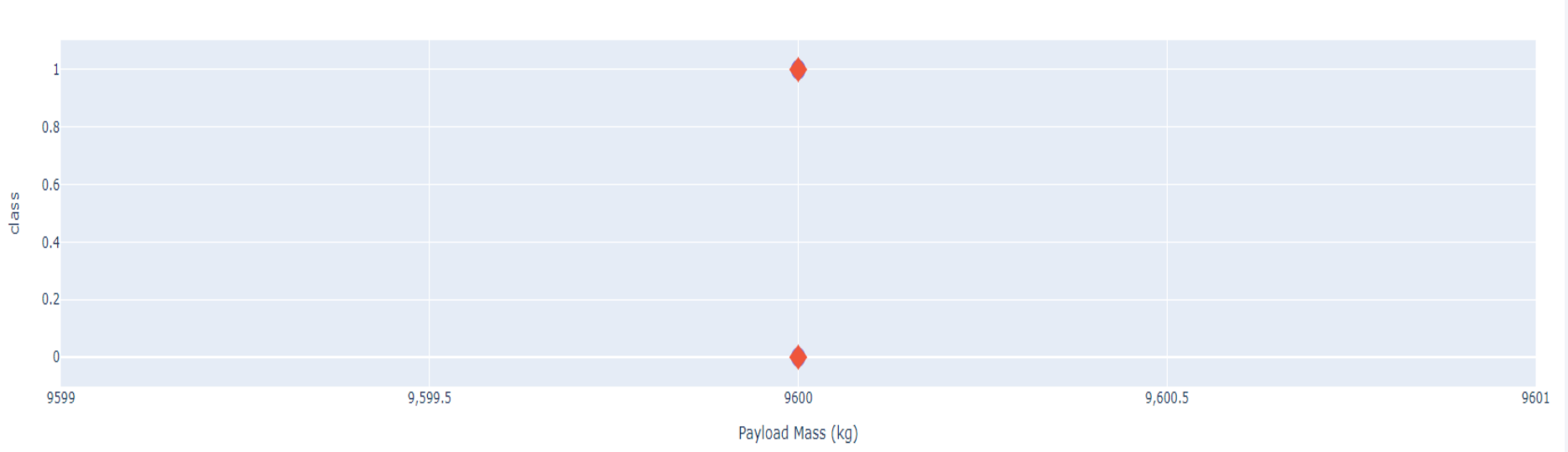
- KSC LC-39A has a success rate of 76.9%
- 10/13 rockets landed successfully at this site
- 3 rockets crashed from this site

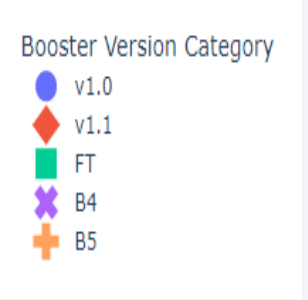
Payload vs Class – All sites



- The most successful landings occurred using the FT booster
- The most successful landings had a payload mass of between 2,000 – 4,000kg
- The most failed landings occurred with the v1.1 booster

Payload vs Class – All sites

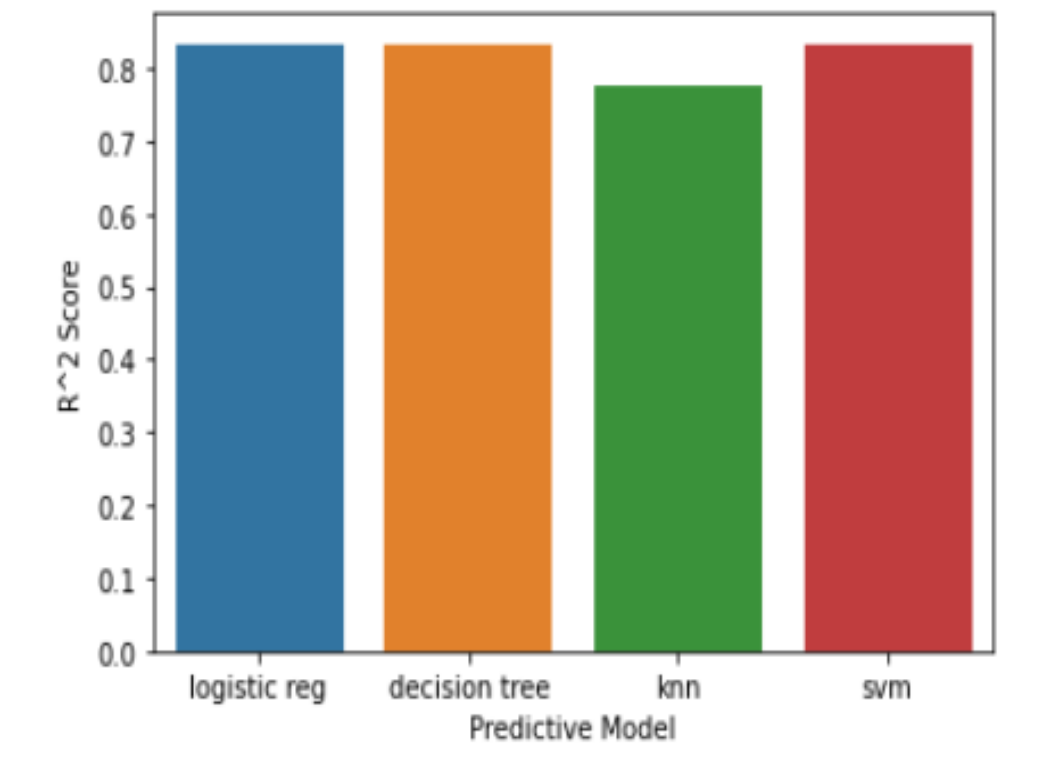




Section 5

Predictive Analysis (Classification)

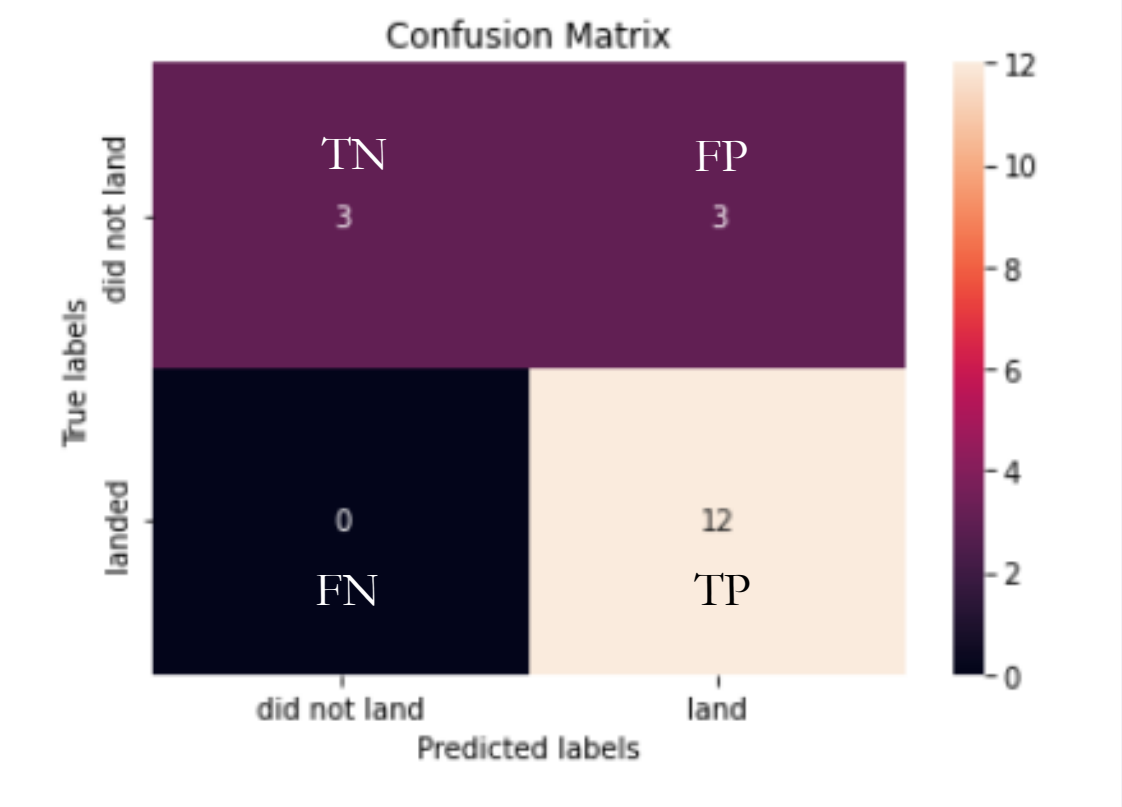
Classification Accuracy



Predictive Model	Score
Logistic Reg	0.83
Decision tree	0.83
KNN	0.77
SVM	0.83

- Logistic reg, decision tree and SVM performed similarly
- Potentially due to a limited dataset

Confusion Matrix



	Precision	Recall	F-1 score
Land	0.8	1	0.89
Did not land	1	0.5	0.67

- Confusion Matrix for the SVM model
- The matrix distinguished between classes

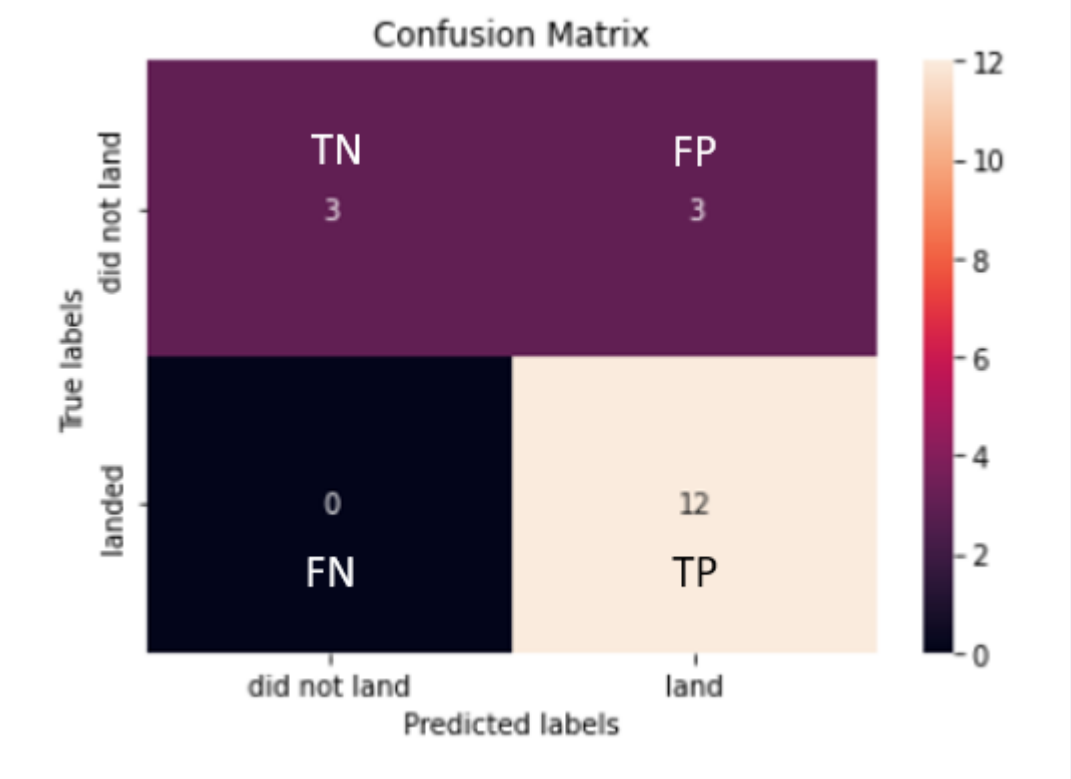
Avg Accuracy = 0.78

Conclusions

- An accurate predictive model was built
- With this model we can predict the cost of a launch
- This model will be more accurate with more data

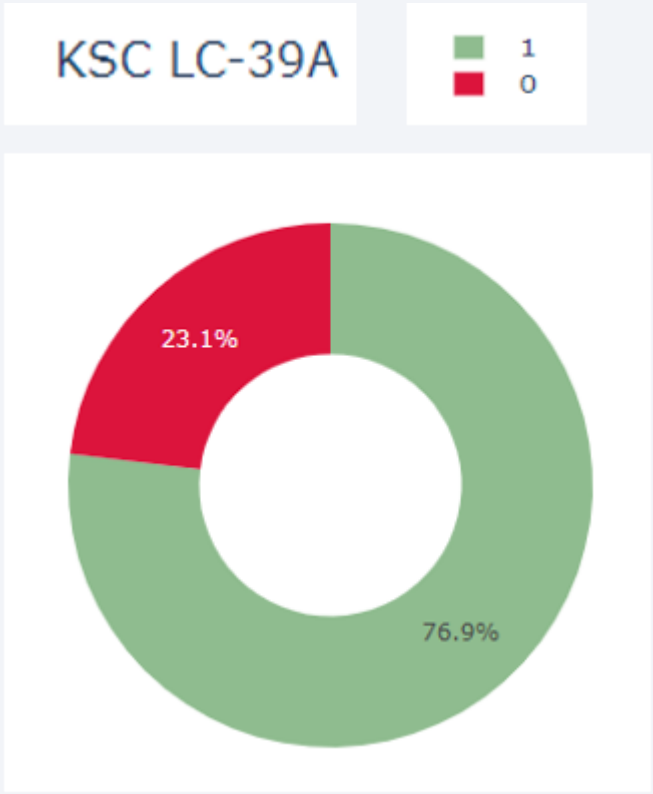
Appendix – Confusion Matrix Calculations

- Precision $= \frac{TP}{TP+FP}$
- Recall $= \frac{TP}{TP+FN}$
- F-1 score $= 2 * \frac{(Precision * Recall)}{(Precision+Recall)}$



Appendix – Plotly Pie Chart Code

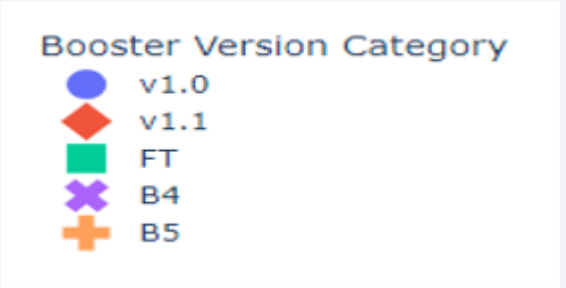
```
filtered_df = spacex_df[spacex_df['Launch Site']==entered_site]
fig = px.pie(filtered_df,
      names= 'class', color = 'class', hole=0.5,
      color_discrete_map={1: '#8FBC8F', 0: '#DC143C'},
      title = entered_site + ' Successful Launch rate' )
```



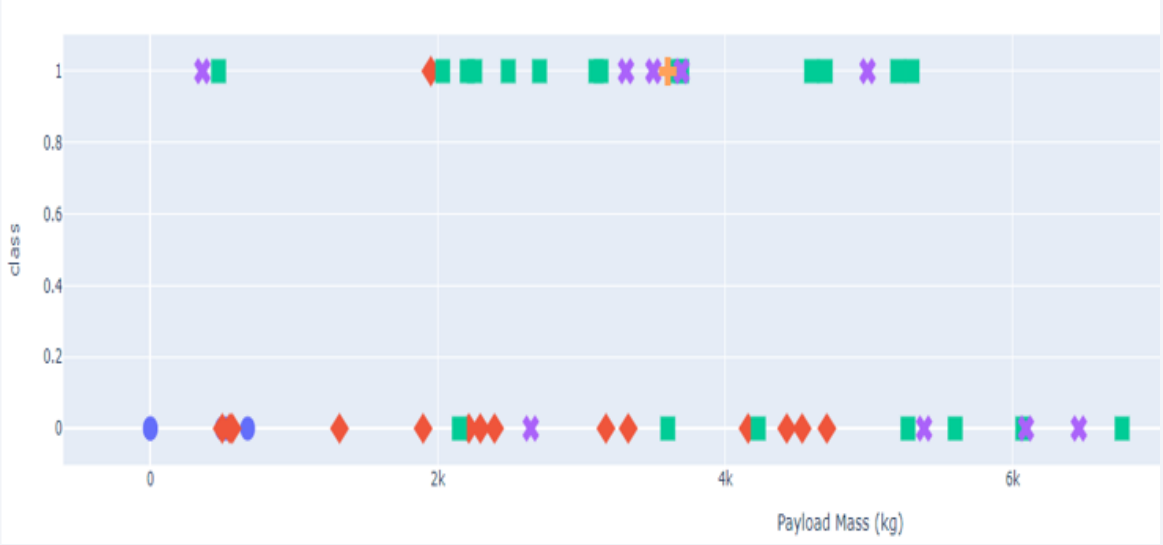
- This code creates a donut chart and changes the color of the classes
- Use the "hole" argument to specify the size of the hole
- Use the "color" and the "color_discrete_map" to change to color displayed on the chart
- Pass a dictionary into the "color_discrete_map" argument and be sure to use CSS color references as the dictionary values
- Refer to the plotly express pie chart documentation for more details

Appendix – Plotly Scatter Plot Code

```
scat_df = spacex_df[(spacex_df['Payload Mass (kg)'] >= payload_mass[0]) &
    (spacex_df['Payload Mass (kg)'] <= payload_mass[1])]
scat_plot = px.scatter(scat_df, x = 'Payload Mass (kg)' , y = 'class',
         color="Booster Version Category", symbol = "Booster Version Category",
         title = "Payload vs class - All sites")
scat_plot.update_traces(marker={'size': 15})
```



- This code creates a scatter plot and changes the color and the shape of the markers
- Use the “color” and “symbol” arguments to specify the color and shape of the markers
- Use the .update_traces() method to the change the size of the markers. This makes the all markers the same size
- Refer to the [plotly express scatter plot documentation] for more details



Thank you!