IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Carlo Marziani
28 Nov. 2023

Outline

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

Executive Summary

Summary of methodologies

- Collect and Wrangle Data
- Exploratory Data Analysis
- Create Interactive Visual Analytics
- Perform Predictive Analysis

Summary of all results

- Exploratory Analysis results
- Interactive Dashboard
- Predictive Analysis Models

Introduction

Project background and context

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars, while for other suppliers each rocket costs more than $165 million, much of the savings is because Space X can reuse the first stage.

Therefore, if we can determine if the first stage will land, we can determine the cost of a launch.

This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

Problems you want to find answers

- What factors determine if the rocket will land successfully?
- The interaction amongst various features that determine the success rate of a successful landing.
- What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- **Data collection methodology:**
  - Launch data was scraped from the Wikipedia page for Falcon 9 Launches
  - Additional data for launch specifics was retrieved through the SpaceX API
- **Perform data wrangling**
  - Missing data was replaced with the mean of the existing values
  - We also reduced the eight listed outcomes to a simplified binary result (success/failure)
- **Perform exploratory data analysis (EDA) using visualization and SQL**
- **Perform interactive visual analytics using Folium and Plotly Dash**
- **Perform predictive analysis using classification models**
  - To ensure the best model we considered regression, KNN, SVM and tree classifiers with multiple parameters
  - Final accuracy was calculated with a withheld test dataset

Data Collection

- **SpaceX provides a range of public data about launches**
- **Launch specifics were retrieved through the SpaceX API**
  - API calls yielded our data for rocket, payloads, launchpad, and cores

- **Additional launch data was scraped from the Wikipedia page for Falcon 9 Launches**
  - This site provided information on launch date and location, payload, boosters, orbit, and outcomes

Data Collection – SpaceX API

We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling.

SpaceX API notebook GitHub

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

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

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

```
mean = df['PayloadMass'].mean()
data_falcon9['PayloadMass'].replace(np.NAN, mean, inplace = True)
```

Data Collection - Scraping

- We applied web scrapping to scrape Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.

[Web scraping notebook GitHub]

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
page = requests.get(static_url)
page.status_code
```

```
200
```

```
soup = BeautifulSoup(page.text)
soup.title
```

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

...

Many lines of code to parse the data

...

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

Data Wrangling

- Initially we identified missing values to prep the data for later evaluation
- Next we identified all of the potential outcomes based on web scraping
- Finally, we reduced outcomes to a binary (1/0) result for good outcomes

[Data wrangling notebook GitHub]

```
df.isnull().sum()/len(df)*100
```

```
FlightNumber      0.000000
Date              0.000000
BoosterVersion    0.000000
PayloadMass       0.000000
Orbit             0.000000
LaunchSite        0.000000
Outcome           0.000000
Flights           0.000000
GridFins          0.000000
Reused            0.000000
Legs              0.000000
LandingPad       28.888889
```

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

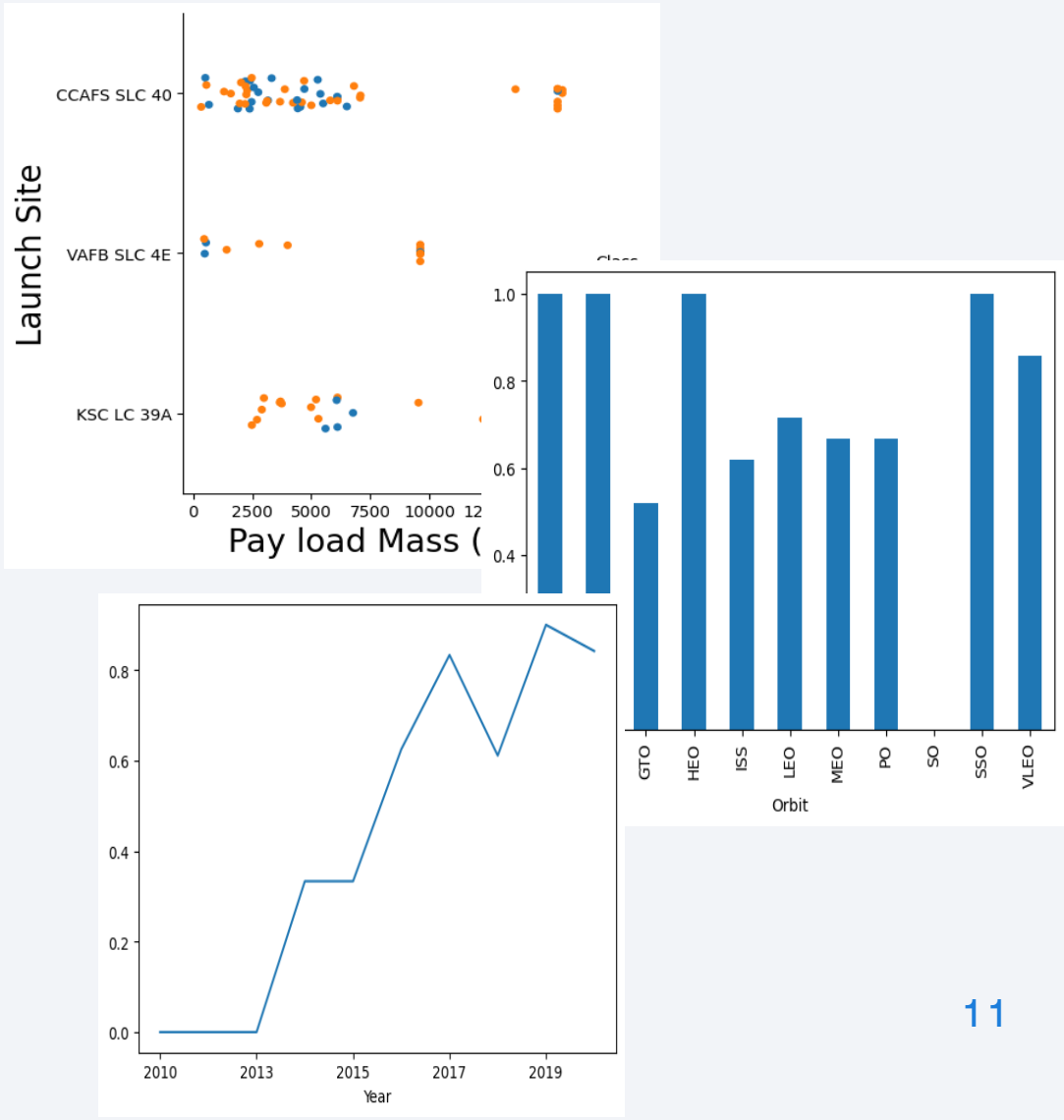
```
landing_class = []

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

EDA with Data Visualization

- **We plotted multiple scatter charts to identify correlation between mass, flight #, and launch site**
  - This chart was chosen due to the fact that each launch represented a distinct data point with X,Y coordinate and success or failure
- **A bar chart was used to display the success rate by orbit**
  - This was an excellent choice for a categorical variable with a single output (percent success)
- **A line plot was used to plot the success rate by year**
  - This is appropriate as year is a progressive variable

[EDA with data visualization GitHub]



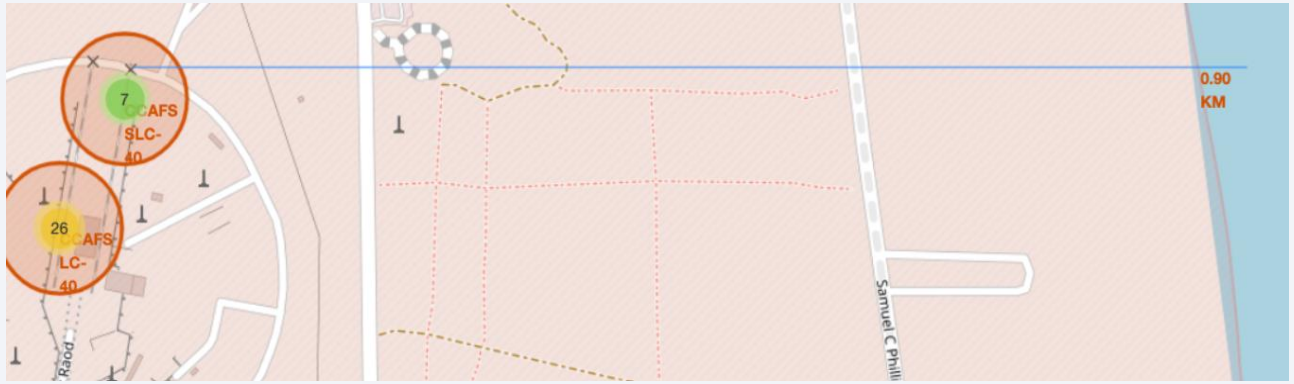
EDA with SQL

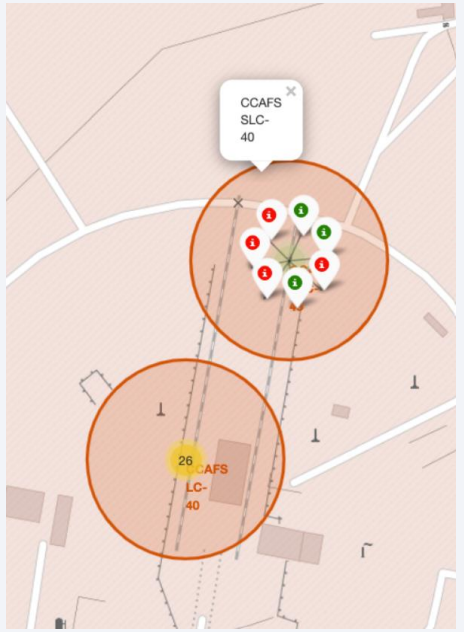
- **We loaded the SpaceX dataset into a PostgreSQL database**
- **We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:**
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.

EDA with SQL notebook GitHub

Build an Interactive Map with Folium

- **We added launch site markers to display count at each location**
  - This provides visual summary of where SpaceX has performed their launches
- **We also added markers for successful or failed launches at that site**
  - This provides additional detail when zooming in on a site about its specific history
- **We calculated the distances between a launch site to its proximities. We answered some question for instance:**
  - Are launch sites near railways, highways and coastlines.
  - Do launch sites keep certain distance away from cities.

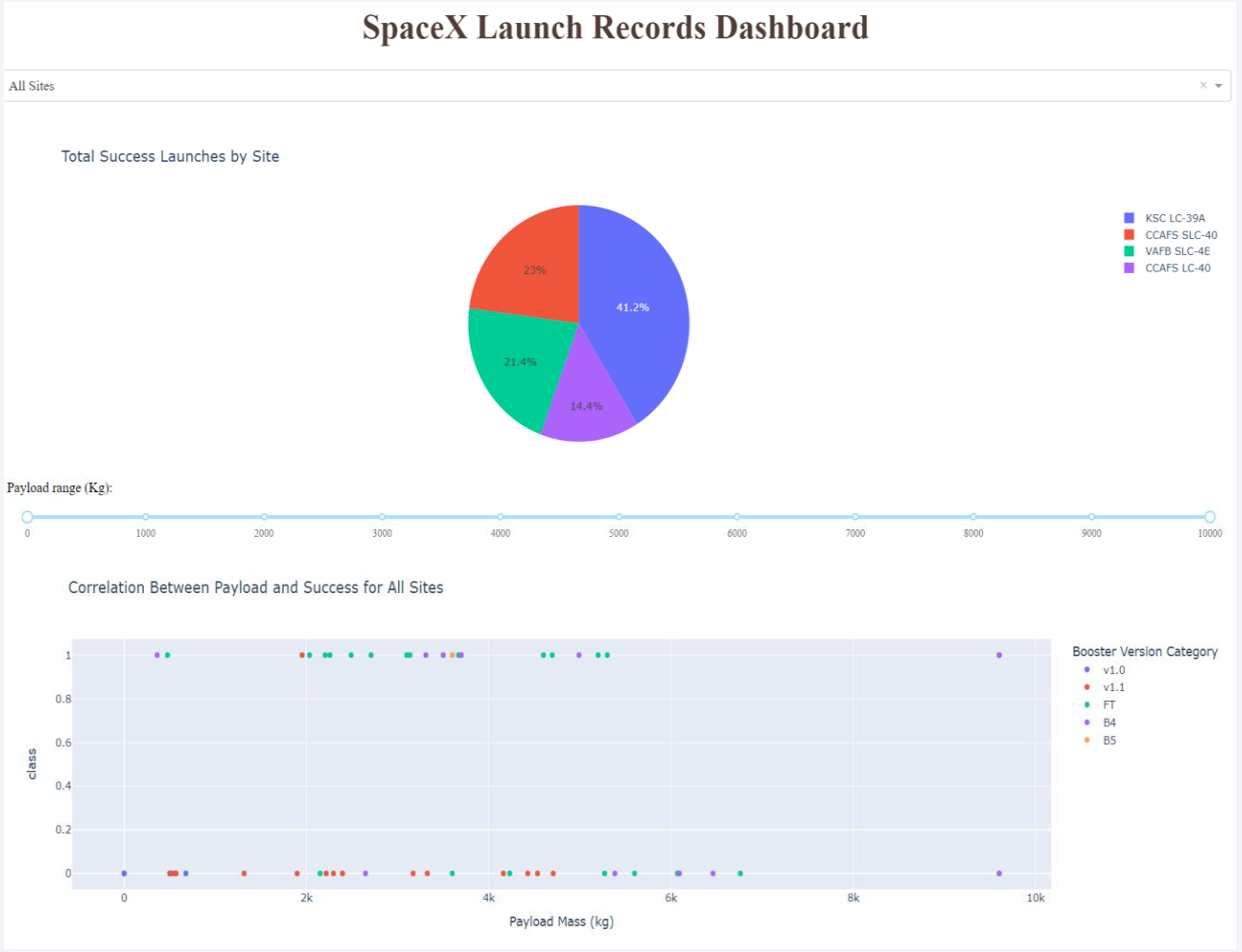




Folium map GitHub

Build a Dashboard with Plotly Dash

- **The first diagram visually represents the successes broken down by site.**
  - Choosing a specific site customizes the diagram to show successes versus failures for that particular site.
  - Using pie charts suits this purpose perfectly, as the cumulative sum will consistently be 100%.
- **Next, we introduced a payload success graph.**
  - The scatterplot allowed us to incorporate details on the booster, distinguishing each point through distinct colors.
  - Additionally, a slider has been implemented that gives users the ability to select more specific payload ranges, when applicable.

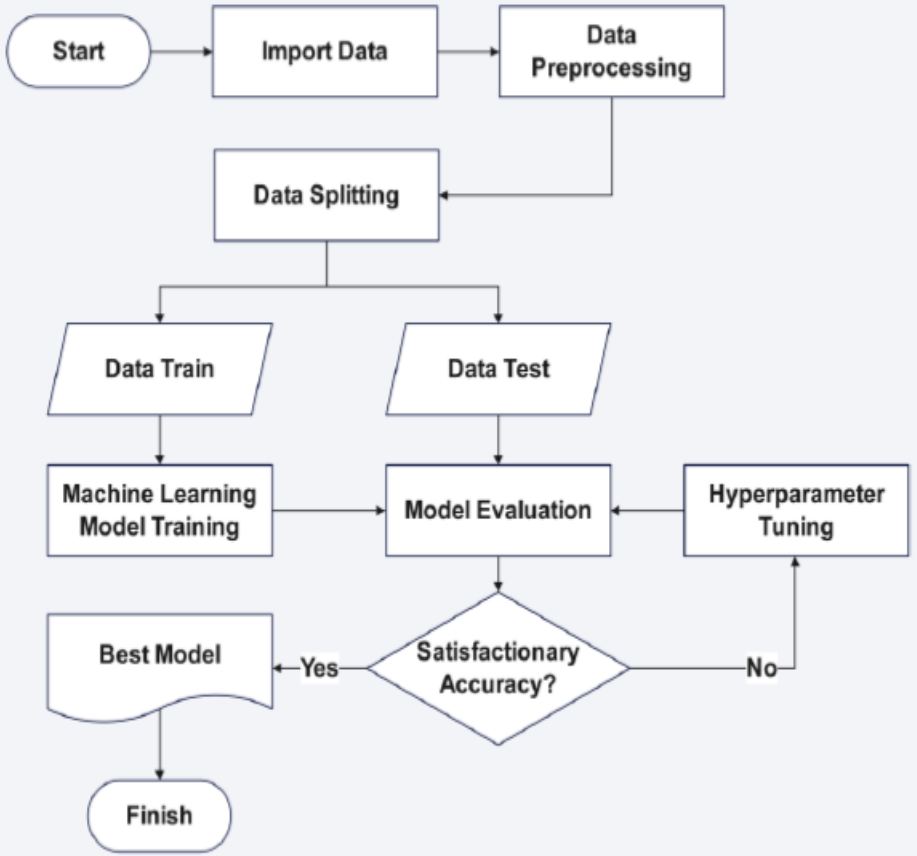


[Plotly Dash GitHub]

Predictive Analysis (Classification)

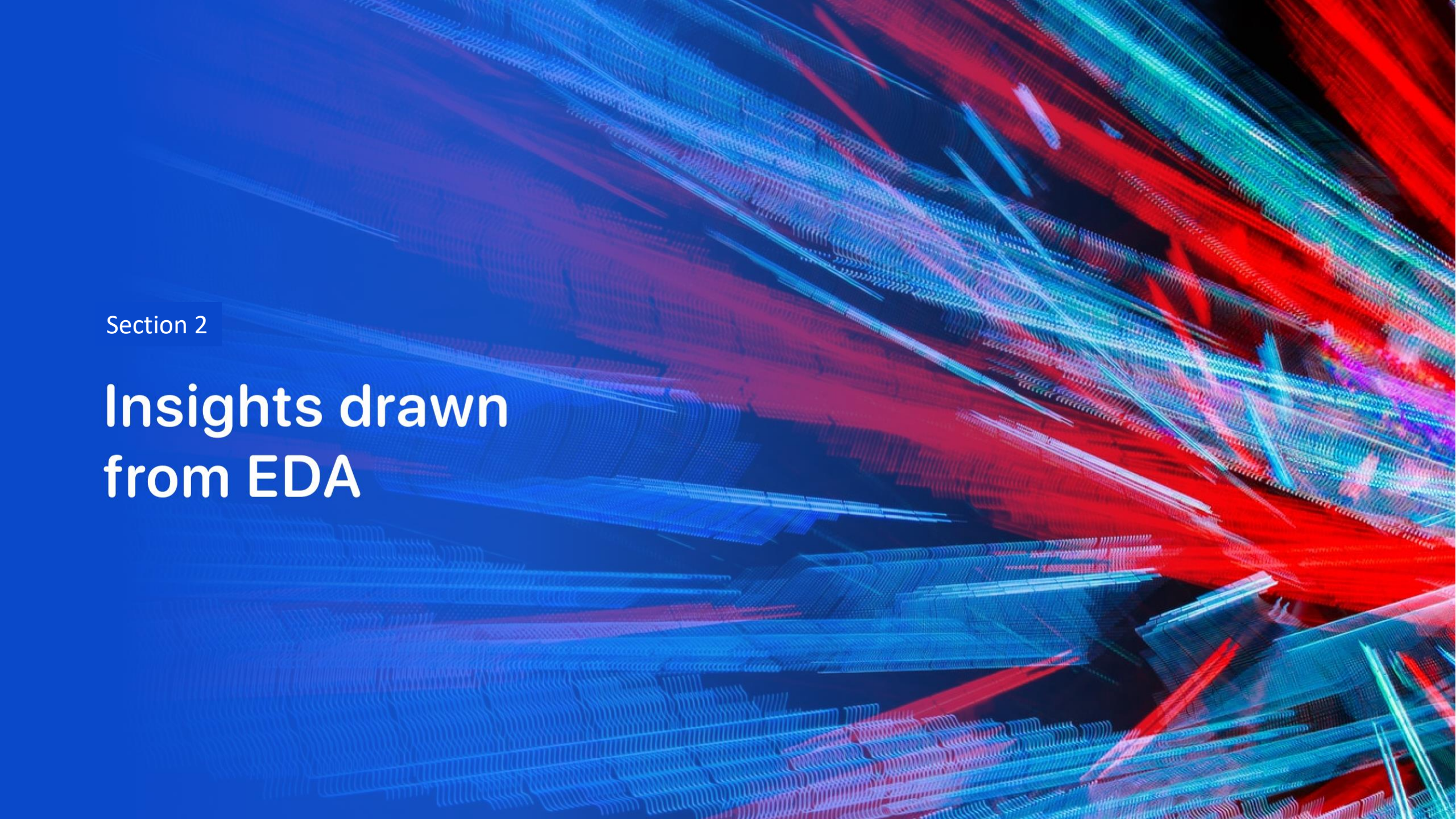
- **Model Creation:** Developed classification models using Logistic Regression, SVM, Decision Tree, and KNN.
- **Hyperparameter Tuning:** Optimized model settings via GridSearchCV to enhance accuracy.
- **Evaluation:** Assessed models using test data, generating confusion matrices and accuracy scores.
- **Best Model:** Decision Tree emerged as the most accurate, achieving approximately 92.9% accuracy.

Predictive Analysis GitHub

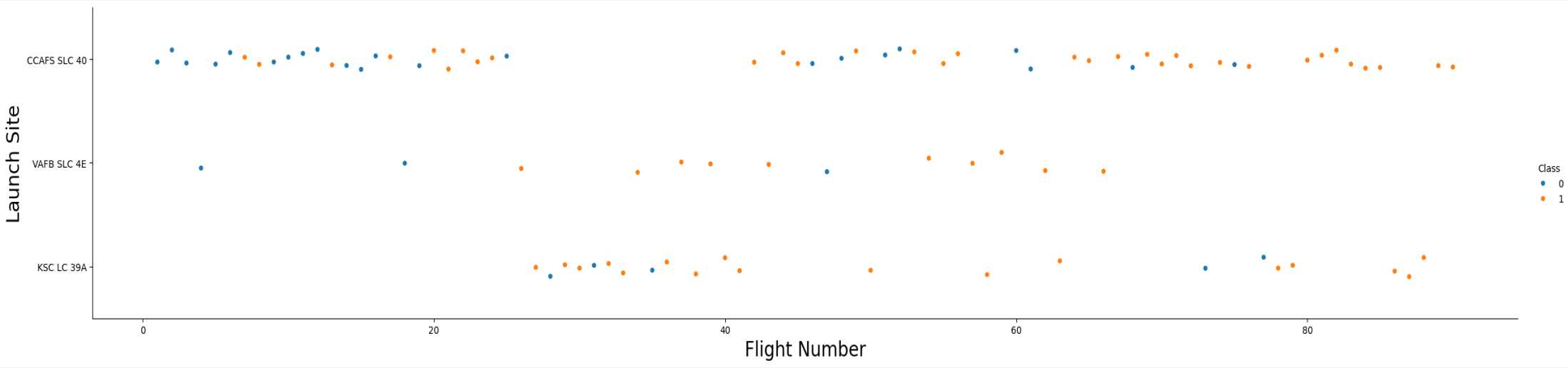


Results

- **Exploratory data analysis results**
  - Launch success has improved over time
  - KSC LC-39A has the highest success rate among landing sites
  - Orbits ES-L1, GEO, HEO, and SSO have a 100% success rate
- **Interactive analytics**
  - Most launch sites are near the equator, and all are close to the coast
- **Predictive analysis results**
  - All models performed similarly on the test set. The decision tree model slightly outperformed when looking at .best_score_

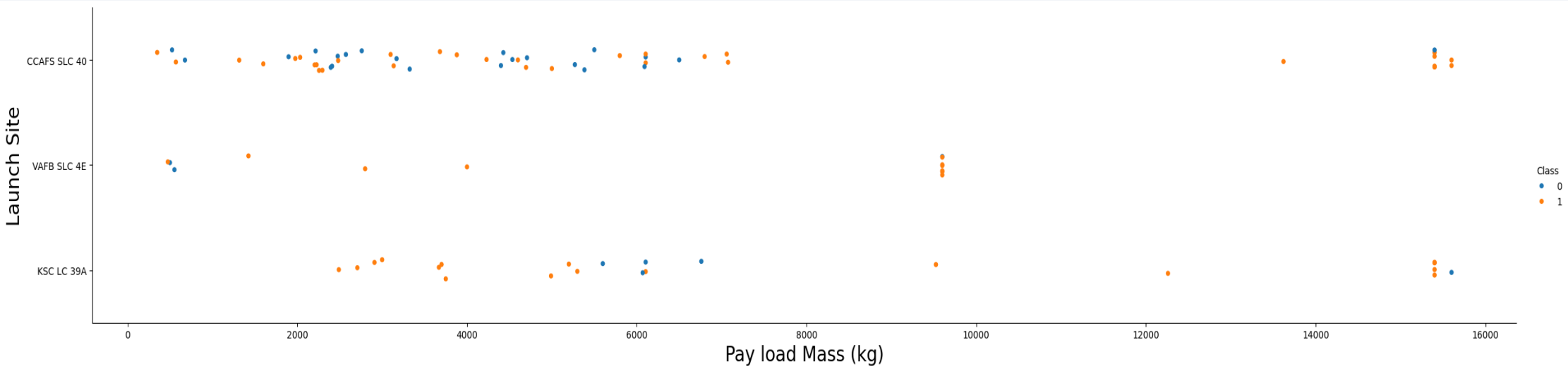

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



- Previous flights show a higher incidence of failures, indicated by the blue dots.
- Initially, the majority of flights came from CCAFS SLC 40, before spreading to all three sites.
- After the twentieth flight, all three sites show a similar high success rate.

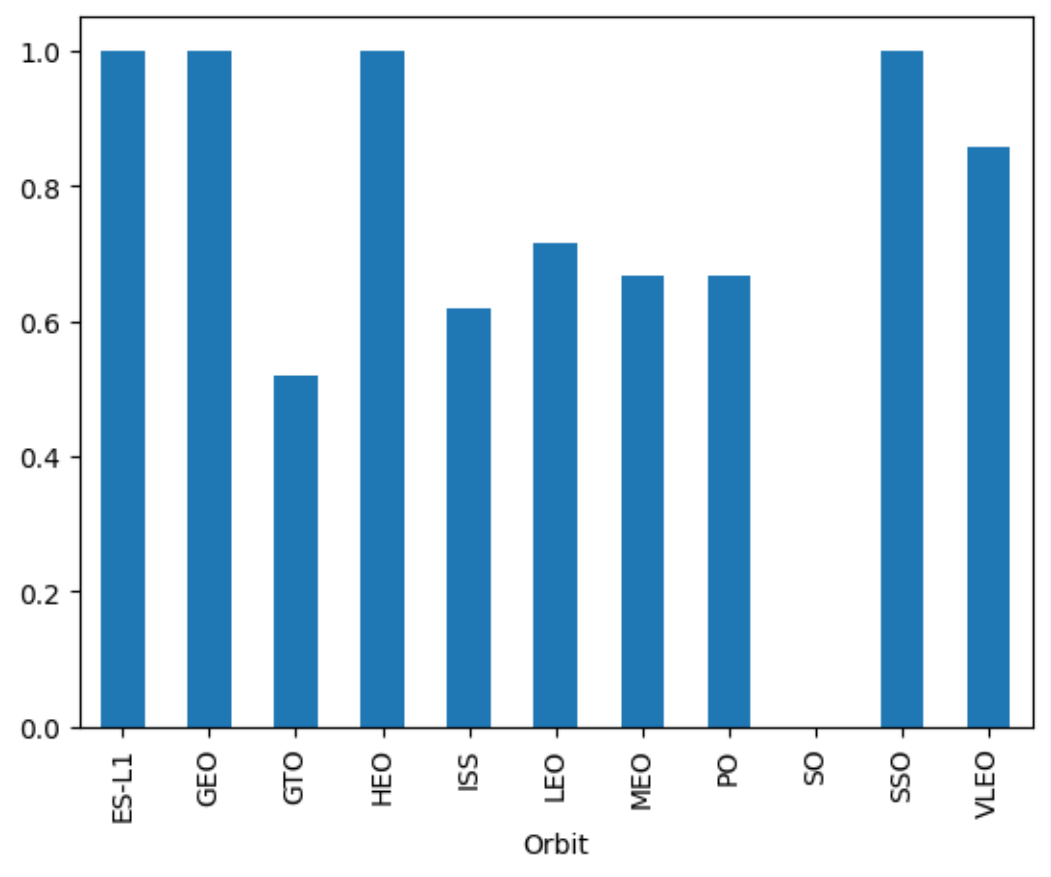
Payload vs. Launch Site



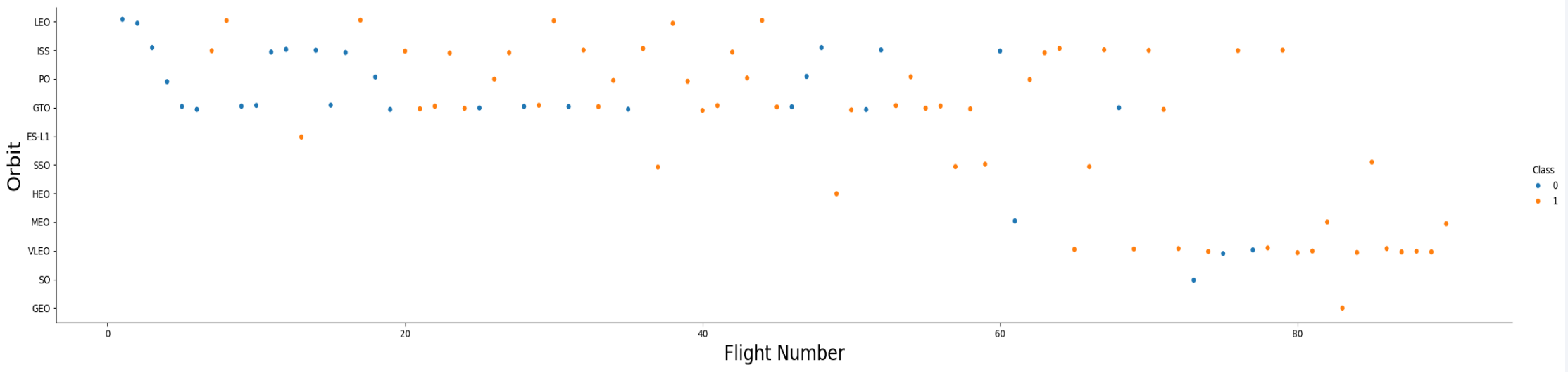
- The vast majority of payloads are less than 8000 kg
- The greater the payload mass, the higher the success rate for the rocket

Success Rate vs. Orbit Type

- Four separate orbits have 100% success:
  - ES-L1
  - GEO
  - HEO
  - SSO
- SO Orbit has yet to be successful
- The remaining orbits have similar 50-80% success rates

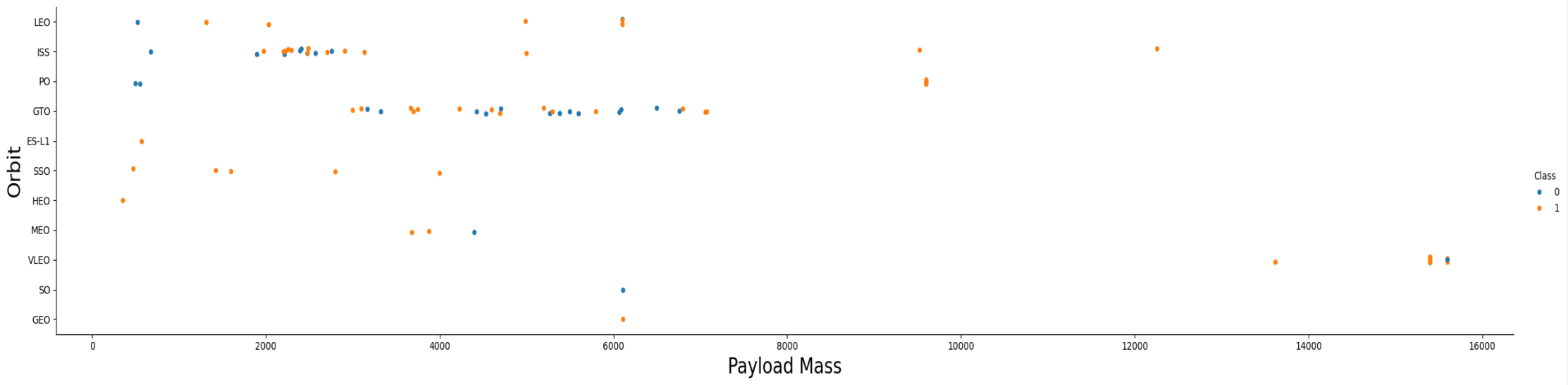


Flight Number vs. Orbit Type



- Early flights fall into LEO, ISS, and GTO
  - All have early failures then consistent successes after #20
- VLEO begins after flight 60 quickly becoming the most frequent
- LEO orbit, success is related to the number of flights

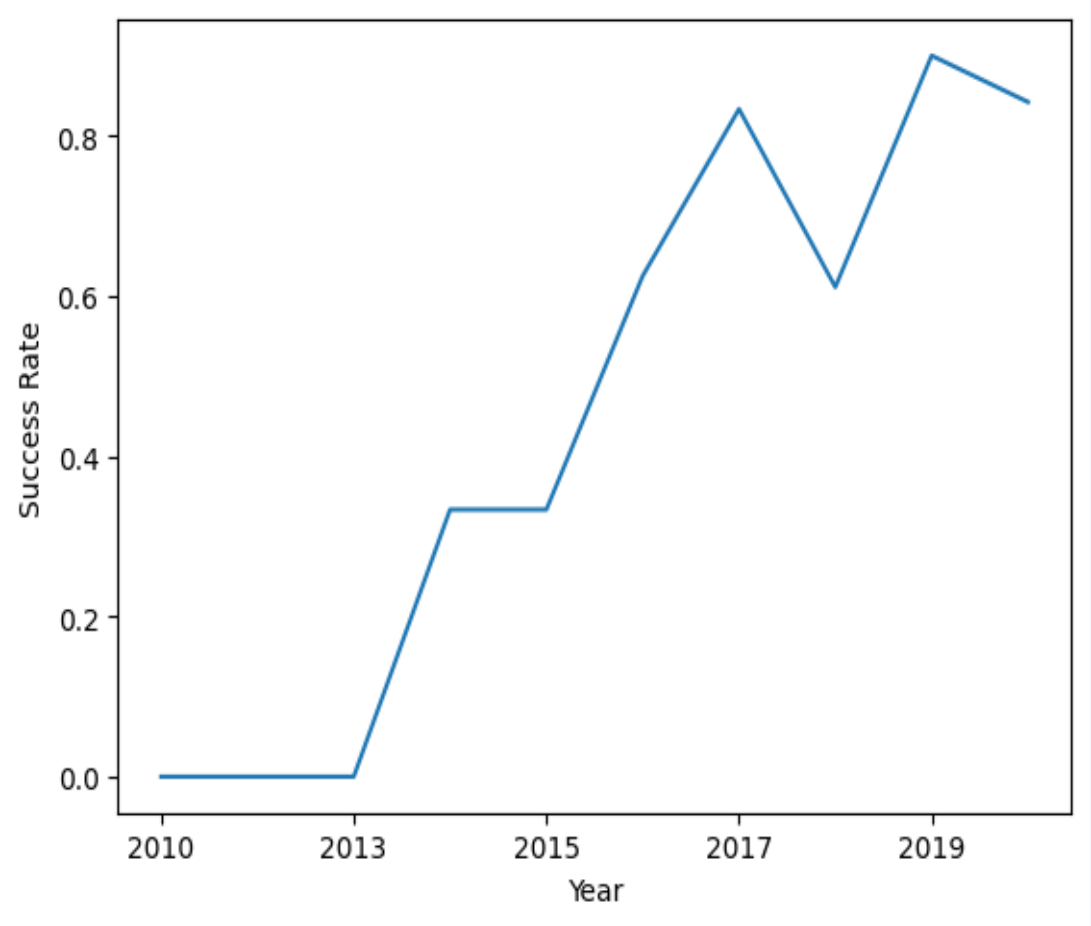
Payload vs. Orbit Type



- VLEO payloads tend to be significantly heavier than any other orbit
- With heavy payloads the successful landing or positive landing rate is higher for PO, VLEO and ISS.
    - For GTO such a relationship is not as clear

Launch Success Yearly Trend

- The first few years saw no success
- Once successes begin in 2013 we see rapid increase
- Since 2017 there has been little increase with some years falling



All Launch Site Names

- We use the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

```
%%sql
SELECT
    DISTINCT "Launch_Site" AS LAUNCH_SITE
FROM
    SPACEXTABLE
```

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

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

Launch Site Names Begin with 'CCA'

- Using the ‘WHERE’ statement we can take only matching results
  - The ‘LIKE’ and % parameters allow us to look for text starting with CCA
- The ‘LIMIT’ parameter allows us to set a number of records we want to see

```
%%sql
SELECT
    *
FROM
    SPACEXTABLE
WHERE
    "Launch_Site" LIKE 'CCA%'
LIMIT
    5;
```

Python

```
 * 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

- We calculate the total payload carried by boosters from NASA using aggregate function ‘SUM’

```
%%sql
SELECT
    SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD_MASS
FROM
    SPACEXTABLE
WHERE
    "Customer" = 'NASA (CRS)';
```

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

TOTAL_PAYLOAD_MASS
45596

Average Payload Mass by F9 v1.1

- In this query we use the aggregate function ‘AVG’
- Again the ‘WHERE’ clause allows us to filter to only Falcon 9 v1.1

```
%%sql
SELECT
    AVG(PAYLOAD_MASS__KG_) AS AVERAGE_PAYLOAD_MASS
FROM
    SPACEXTABLE
WHERE
    "Booster_Version" = "F9 v1.1"
```

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

AVERAGE_PAYLOAD_MASS
2928.4

First Successful Ground Landing Date

- In this case we use the ‘MIN’ function to get the First Successful Ground Landing Date
- The ‘WHERE’ clause limits to only successful ground pad landings

```
%%sql
SELECT
    MIN("Date") AS "DATE_WHICH_FIRST_SUCCESSFUL_LANDING_OUTCOME_IN_GROUND_PAD_WAS_ACHIEVED"
FROM
    SPACEXTABLE
WHERE
    "Landing_Outcome" = 'Success (ground pad)'
```

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

DATE_WHICH_FIRST_SUCCESSFUL_LANDING_OUTCOME_IN_GROUND_PAD_WAS_ACHIEVED
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We use the WHERE clause to filter for boosters which have successfully landed on drone ship AND we apply the BETWEEN condition to determine successful landing with payload mass greater than 4000 but less than 6000

```
%%sql
SELECT
    "Booster_Version" AS BOOSTER_VERSION
FROM
    SPACEXTABLE
WHERE
    "Landing_Outcome" = 'Success (drone ship)' AND
    PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

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

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

Total Number of Successful and Failure Mission Outcomes

- GROUP BY' allows us to group data based on certain criteria. (in this case outcome)
- 'COUNT' function allows us to take the total count of rows regardless of content

```
%%sql
SELECT
    "Mission_Outcome" AS MISSION_OUTCOME, COUNT(*) AS TOTAL_NUMBER
FROM
    SPACEXTABLE
GROUP BY
    "Mission_Outcome"
```

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

MISSION_OUTCOME	TOTAL_NUMBER
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- This query uses two distinct SELECT statements
- In order to filter only the maximum payload, we must first find that value
  - This is the role of the second SELECT statement
  - Setting payload=max(payload) will not work

```
%%sql
SELECT
    "Booster_Version" AS BOOSTER_VERSION
FROM
    SPACEXTABLE
WHERE
    PAYLOAD_MASS__KG_ = (SELECT
                             MAX(PAYLOAD_MASS__KG_)
                         FROM
                             SPACEXTABLE)
```

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

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

- In this query we are going to use the substr() function which extracts the month and year from the 'Date' column.
- We list the failed 'LANDING_OUTCOMES' in drone ship, their booster versions, and launch site names for in year 2015

```
%%sql
SELECT
    substr("Date", 6, 2) AS MONTH,
    "Date" AS DATE,
    "Booster_Version" AS BOOSTER_VERSION,
    "Launch_Site" AS LAUNCH_SITE,
    "Landing_Outcome" AS LANDING_OUTCOME
FROM
    SPACEXTBL
WHERE
    "Landing_Outcome" = 'Failure (drone ship)' AND
    substr("Date", 1, 4) = '2015'
```

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

MONTH	DATE	BOOSTER_VERSION	LAUNCH_SITE	LANDING_OUTCOME
01	2015-01-10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- This query uses a number of previously discussed functions (WHERE, BETWEEN, and GROUP BY)
- Let's add one more instruction 'ORDER BY' which allows us to sort values
  - The addition of 'DESC' parameter tells us we want descending values

```
%%sql
SELECT
    "Landing_Outcome" AS LANDING_OUTCOME,
    COUNT(*) as COUNT_OUTCOMES
FROM
    SPACEXTBL
WHERE
    "Date" BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY
    "Landing_Outcome"
ORDER BY
    COUNT_OUTCOMES DESC
```

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

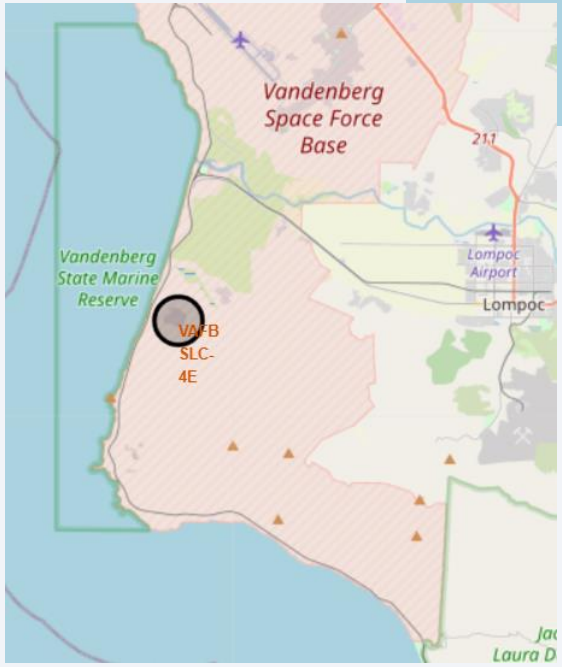
LANDING_OUTCOME	COUNT_OUTCOMES
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

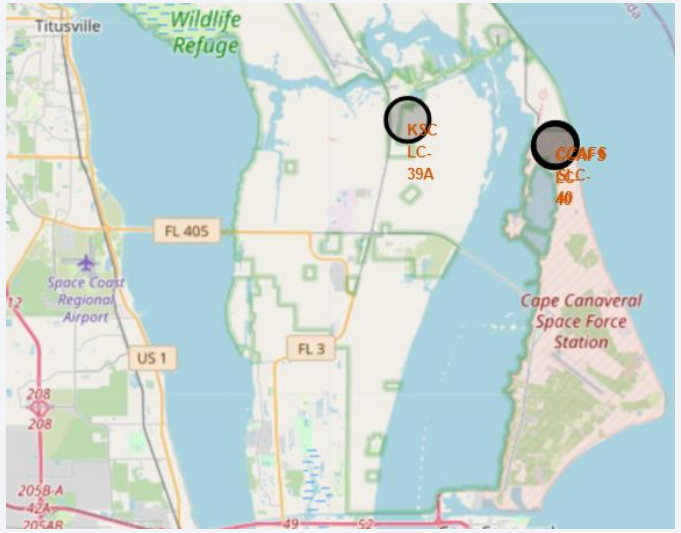
Section 3

Launch Sites Proximities Analysis

Launch Sites

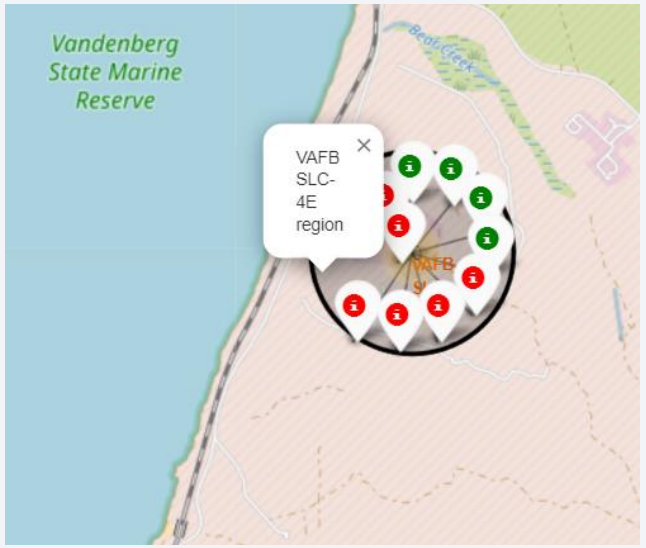
- We see that all launches fall in the southern half of the US in California or Florida
- Launch sites closer to the equator are preferable thus the southern part of U.S.

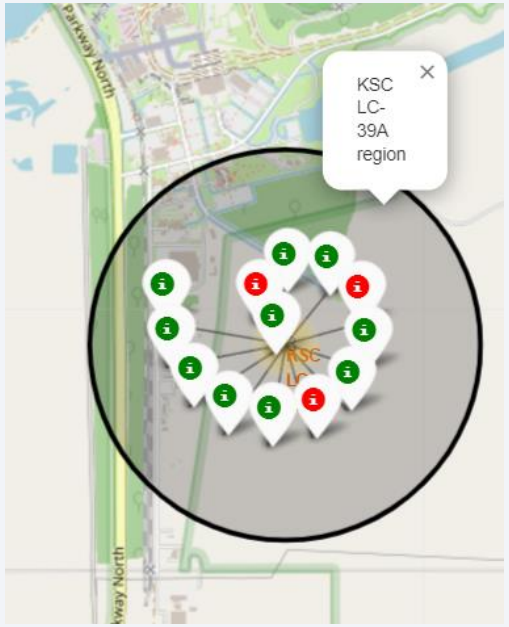


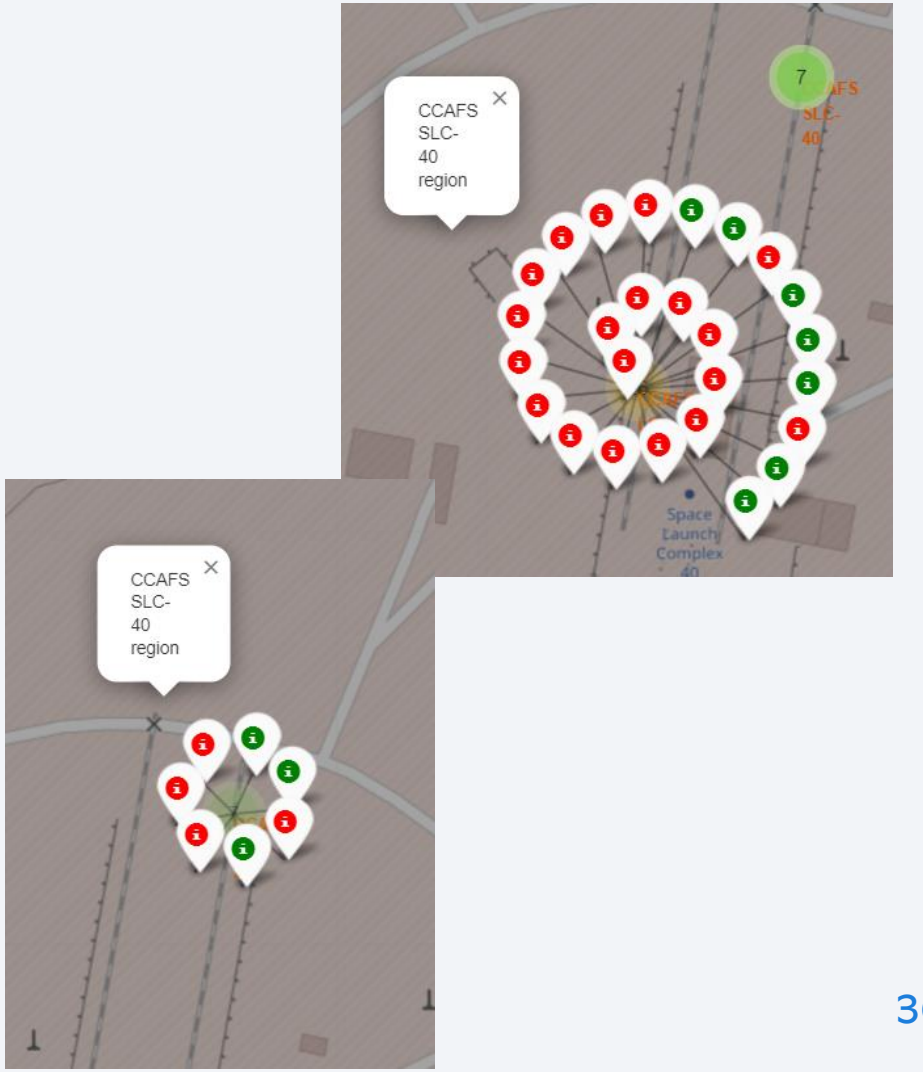


Success and Failed Launches for each site

- Markers have been added:
  - red to indicate failed launches
  - green to indicate successful launches
- The best site is KSC LC-39A





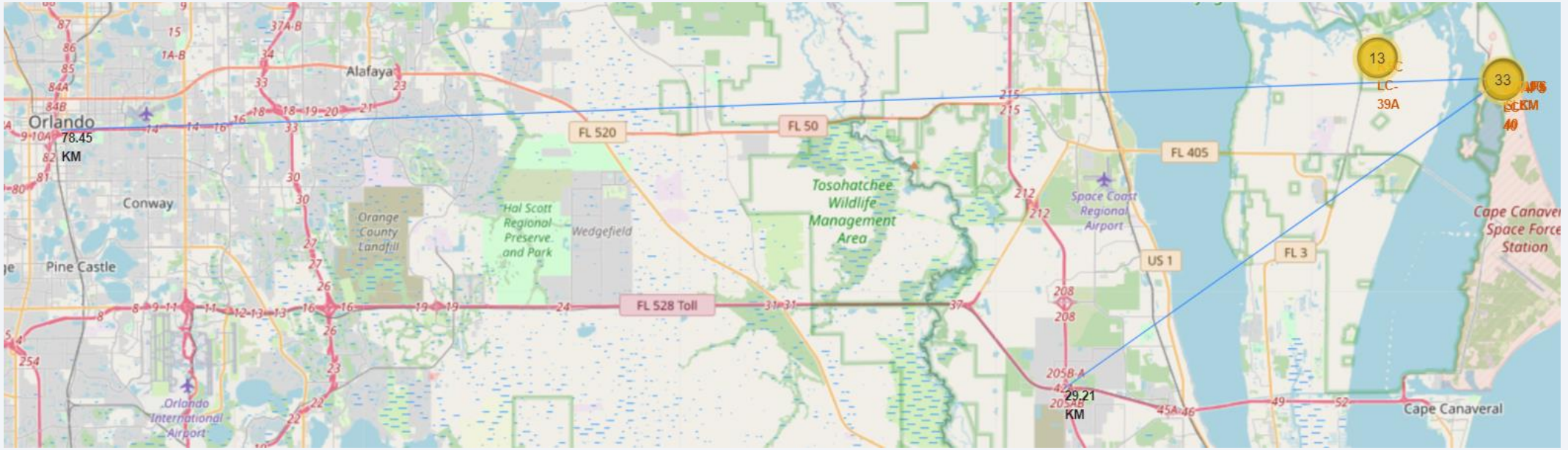


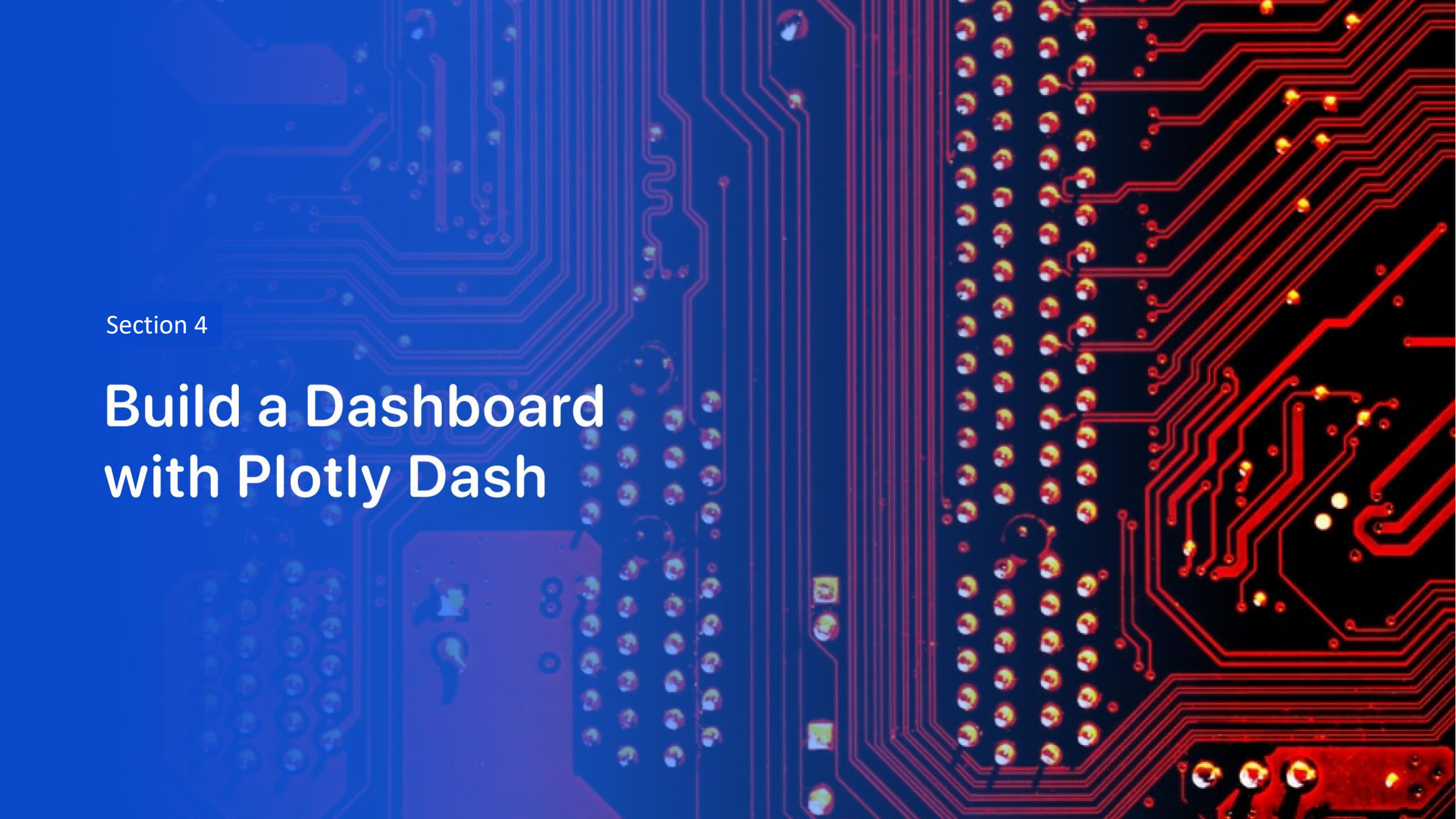
Launch Site distance to landmarks

Examining CCAFS SLC-40 we find that it is quite close to the coast but at a fair distance from both the city and the highway.

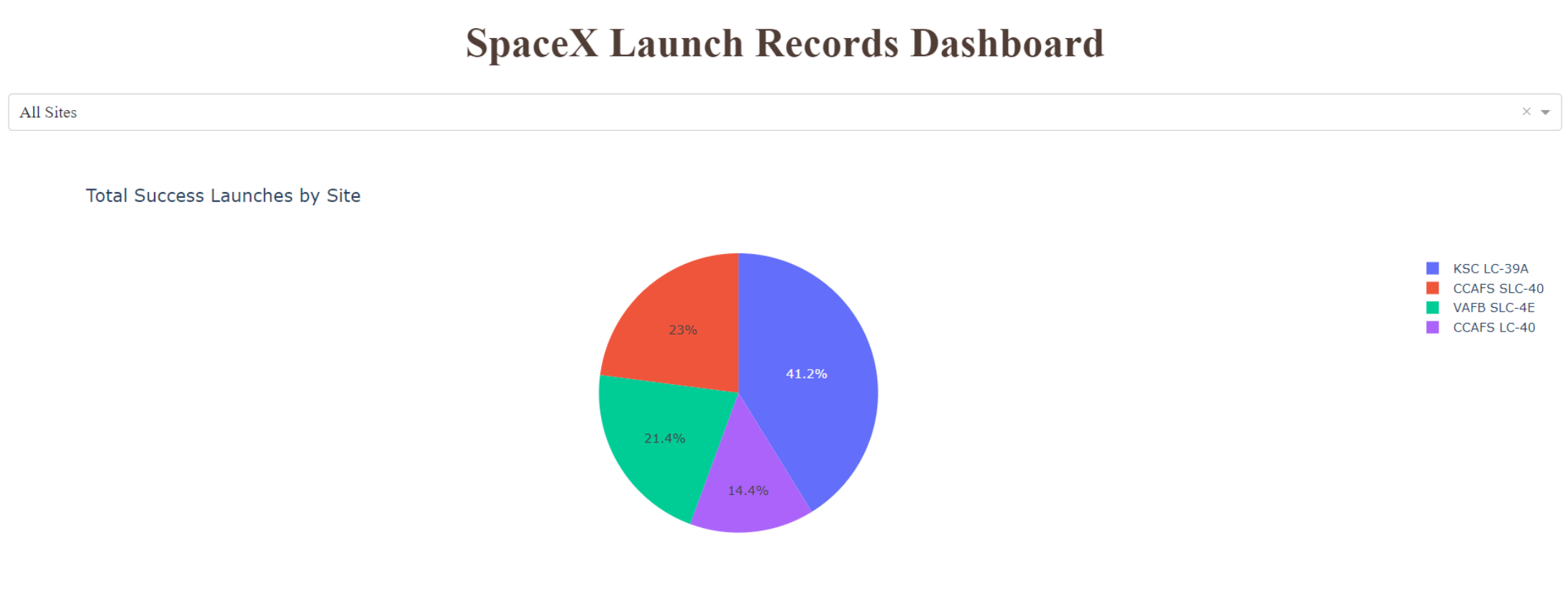
Launches generate an enormous amount of noise and would need to be away from population centers.

Launching close to the coast would also allow for trajectory over water, reducing risk of damage if break up occurs.



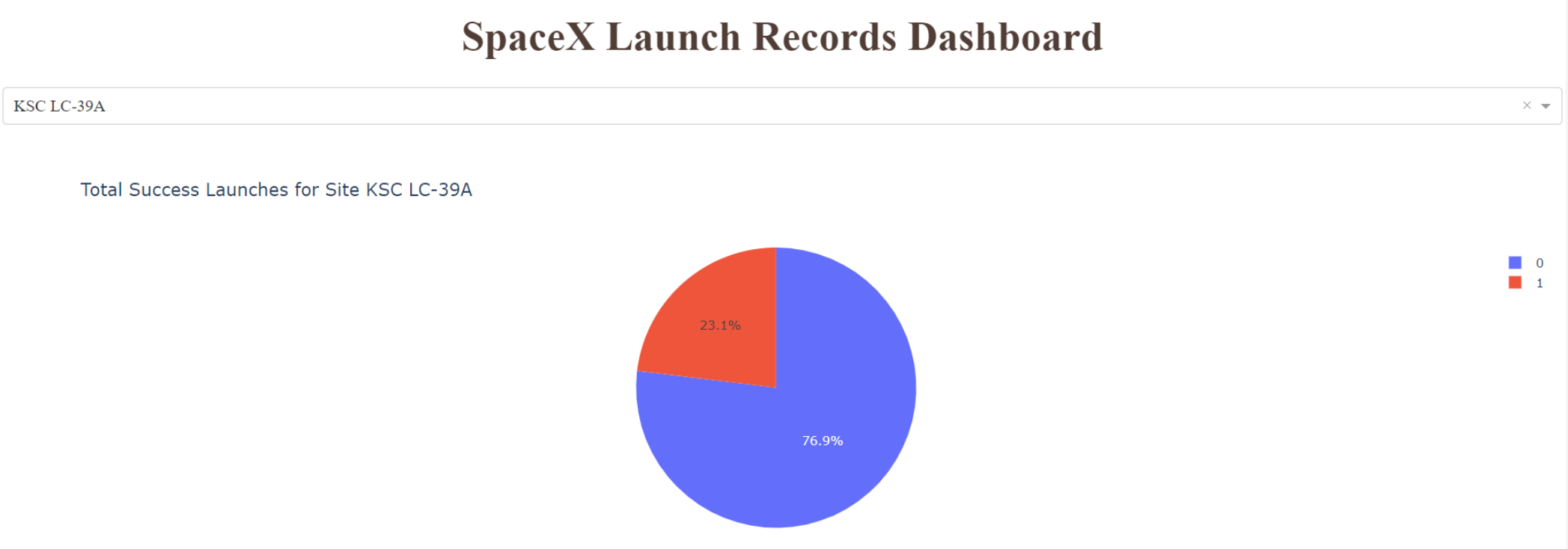

Section 4
Build a Dashboard with Plotly Dash

Distribution of Successful Landings by Site



- The majority of successful launches are from site KSC LC 39A (41.2%).

Success Rate for KSC LC-39A



- By looking at a single site we can show the success and failure rates
- KSC LC 39A has a high success rate with 76.9% successful launches.

Payload vs. Launch Outcome

We reviewed payloads under 5000kg:

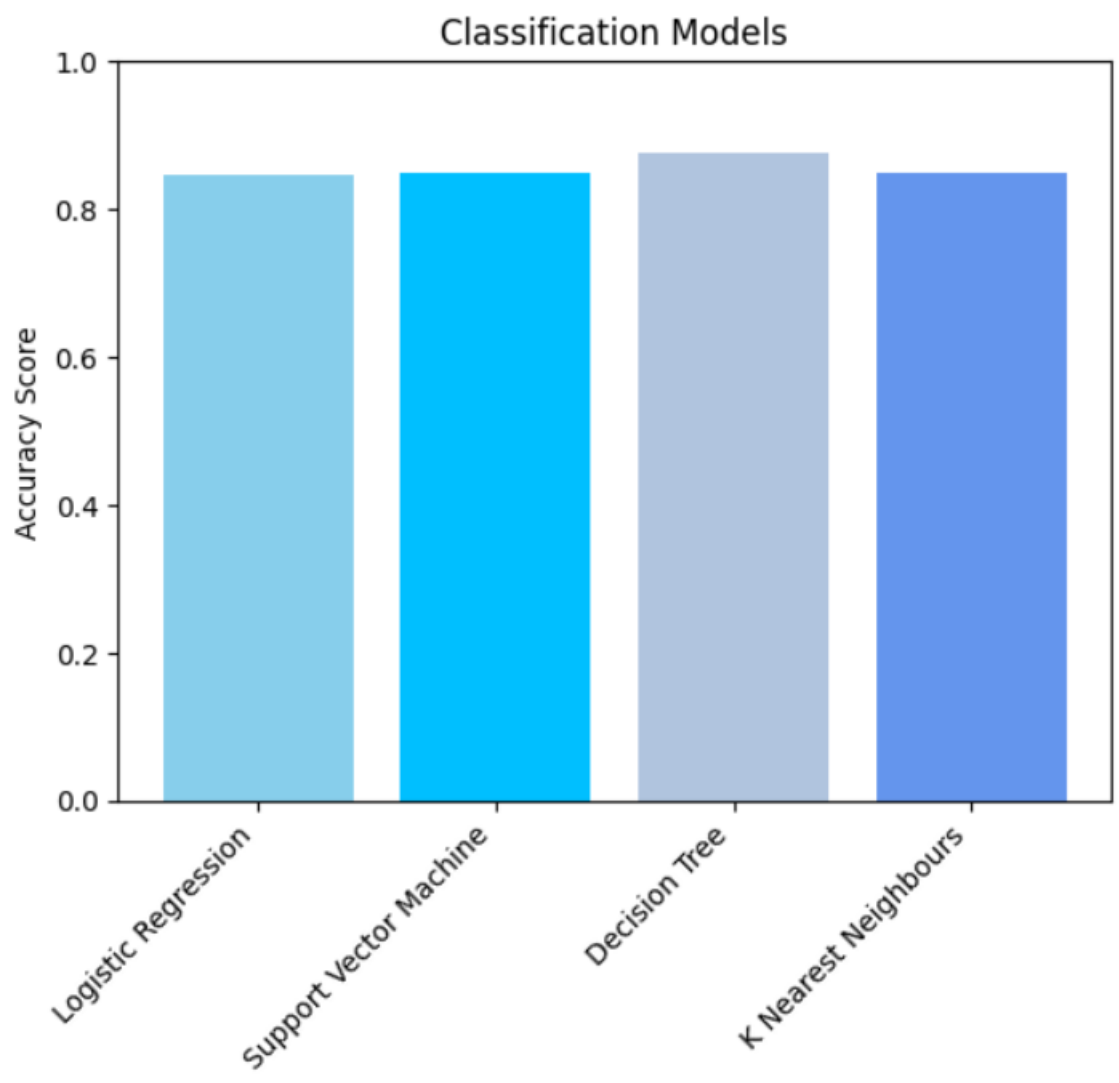
- There is little correlation between payload mass and success
- There does appear to be correlation between booster and success
  - The FT and B4 boosters seem particularly successful and the B5 appears unfailing

Section 5

Predictive Analysis (Classification)

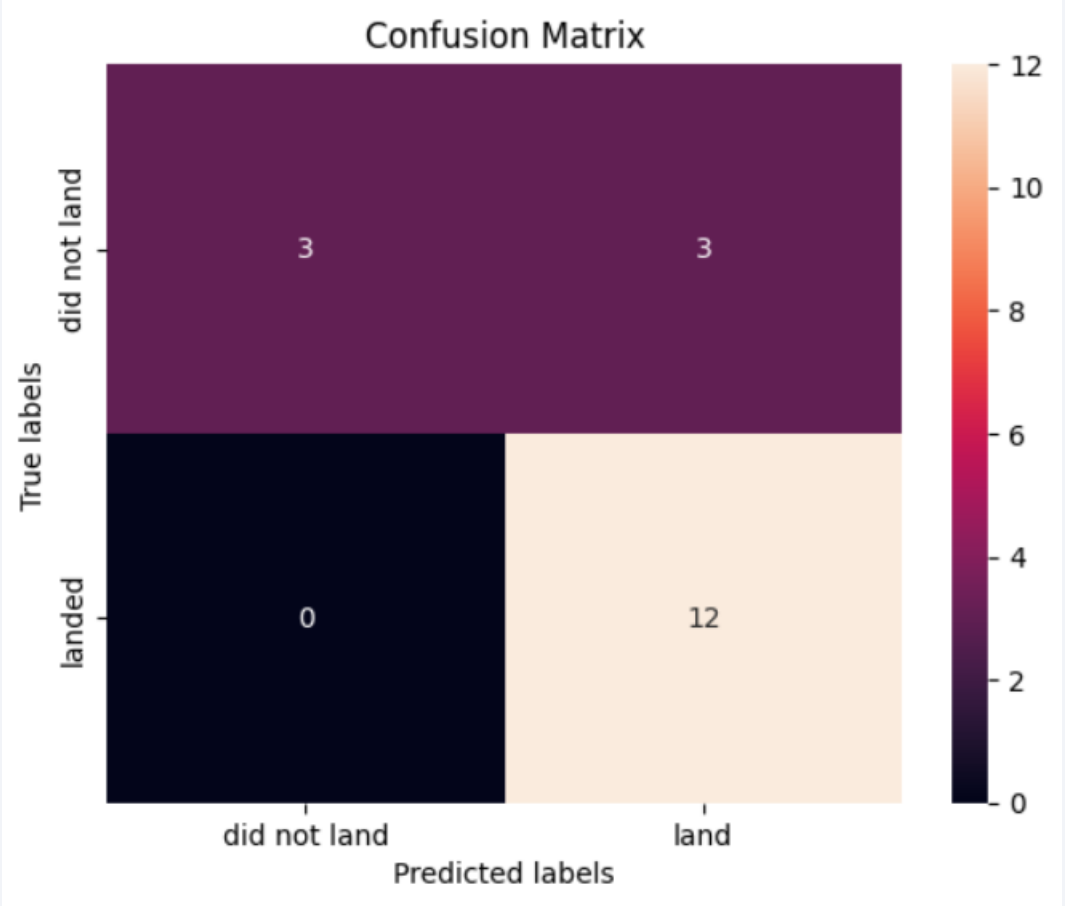
Classification Accuracy

- The optimized models achieve an average accuracy of 80%.
- Decision Tree outperforms other models, boasting the highest accuracy among classifiers, achieving an accuracy of 87.68%.



Confusion Matrix

- False positives are the thing we need to think about.
- In our case there are 3 false positives, i.e. the model predicted that there would be a landing which did not occur



Conclusions

- **Model Performance**: The models performed similarly on the test set with the decision tree model slightly better
- **Equator**: Most of the launch sites are near the equator for an additional natural boost - due to the rotational speed of earth - which helps save the cost of putting in extra fuel and boosters
- **Coast**: All the launch sites are close to the coast
- **Launch Success**: Increases over time
- **KSC LC-39A**: Has the highest success rate among launch sites.
- **Orbits**: ES-L1, GEO, HEO, and SSO have a 100% success rate
- **Payload Mass**: Across all launch sites, the higher the payload mass (kg), the higher the success rate

Appendix

- All code and analysis can be found on my GitHub profile.

Thank you!