Winning Space Race with Data Science

Mitchell Fargher
January 30th, 2022

Outline

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

Executive Summary - Methodologies

Predictive Analysis

3

Explored a variety of predictive analysis algorithms to predict the landing outcomes from the SpaceX dataset.

Exploratory Data Analysis

2

Initially explored the dataset using SQL queries. The data were then visualized with Python, Folium Maps, and an interactive Plotly Dash Web Application.

Data Collection & Wrangling

1

Data were collected using both the SpaceX REST API and scraped from the web using Beautiful Soup. Data were transformed to include a new classification column for success and failure.

Executive Summary - Results

Number of Flights

As the number of flights increased, so did the successful landing outcome.

Payload Mass

Flights that had a higher payload mass did not have as much success.

Launch Site

The location of the launch site was directly related to if the mission was successful.

Predictive Algorithm

The decision tree classifier algorithm was the best method for predicting successful or unsuccessful landing outcomes.

Introduction

Project Background & Context

To predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch.

Problems to Answer

- What influences a successful landing?
- Variables that impact the success rate?
- What conditions are conducive to a successful landing?

View the project notebooks on GitHub

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data were collected both by the SpaceX REST API & web scraping Wikipedia
- Perform data wrangling
  - Data was processed using NumPy & Pandas to add a classification column
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Build, tune, and evaluate classification models

Data Collection

- Data were collected using two methods:
  - SpaceX REST API
    - Decoded the response content as a JSON using .json()
    - Turned the data into a Pandas dataframe using .json_normalize()
    - Filled in missing data in the dataframe using .fillna(.mean())
  - Web Scraping SpaceX Falcon 9 Launch Wikipedia
    - Created a BeautifulSoup object from the HTML response
    - Extracted and parsed the html launch records table
    - Turned the extracted launch records into a Pandas dataframe

Data Collection – SpaceX API

1) Get response from SpaceX API & convert to JSON

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

2) Create a dictionary of column data

```
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}
```

3) Create Pandas dataframe from dictionary

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

4) Filter out Falcon 1 data and replace missing values

```
data_falcon9 = data.loc[data['BoosterVersion'] != 'Falcon 1']
data_falcon9['PayloadMass'].fillna((data_falcon9['PayloadMass'].mean()), inplace=True)
```

Data Collection – Web Scraping

1) Get HTML response & BeautifulSoup object

```
response = requests.get(static_url)
html = response.content
soup = BeautifulSoup(html, 'html5lib')
html_tables = soup.find_all('table')
first_launch_table = html_tables[2]
```

2) Extract column names from header

```
column_names = []
first_launch_table.find_all('th')
for row in first_launch_table.find_all('th'):
    cols=row.find_all('td')
    name=extract_column_from_header(row)
    if name is not None and len(name) > 0:
        column_names.append(name)
```

3) Create a dictionary & fill with extracted data

```
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'] = []
```

4) Create Pandas dataframe & export

```
df=pd.DataFrame(launch_dict)

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

Data Wrangling

- Two main objectives:
  - Exploratory Data Analysis
    - Determined which columns were numerical or categorical
    - Calculate the number of launches from each site
    - Calculate the number and occurrence of each orbit
  - Determine Training Labels
    - Calculate number & occurrence of mission outcomes per orbit type
    - Create landing outcome label from outcome column
    - Append new class column with landing outcome

EDA with Data Visualization

Scatter Plots

- Payload Mass vs Flight Number
- Launch Site vs Flight Number
- Launch Site vs Payload Mass
- Orbit vs Flight Number
- Orbit vs Payload Mass

We use scatter plots to visualize how one variable is affected by another. This relationship helps us determine if the variables are correlated.

Bar Graph

- Mean Class vs Orbit

We use bar graphs to visualize how a single variable differs across a number of categories.

Line Graph

- Mean Class vs Year

We use line graphs to visualize trends. We look at how the average class changes over years.

EDA with SQL

- Load the dataset into Db2
  - Connect to Db2 using SQL Magic
- Explore & understand the database using SQL queries
  - Names of distinct launch sites
  - Total payload mass carried by boosters launched by NASA (CRS)
  - Average payload mass carried by booster version F9 v1.1
  - Date when first successful ground pad landing outcome achieved
  - Total number of successful and failure mission outcomes
  - Booster versions which have carried the maximum payload mass
  - Failed drone ship landing outcomes, booster versions, and launch site names for the year 2015

Building an Interactive Map with Folium

- Latitude and longitude were used to label launch sites and draw a circle marker
- Launch Outcomes were assigned 0 (red) or 1 (green) and placed on the map using MarkerCluster()
- A line was drawn from the launch site to prominent landmarks and labeled with the distance between the two:
  - Coast
  - Highway
  - Railway
  - City
- Determine if launch sites are near the coast, highways, or railways
- Determine if launch sites are kept a particular distance from cities

Build a Dashboard with Plotly Dash

- An interactive web app was built using Plotly Dash
- Graphs displayed in the web app:
  - Pie Chart
    - Total successes by launch site
    - Success and failure percent by selected launch site
  - Scatter Plot
    - Class vs Payload Mass
    - Differentiated by booster version

Predictive Analysis (Classification)

Building Model

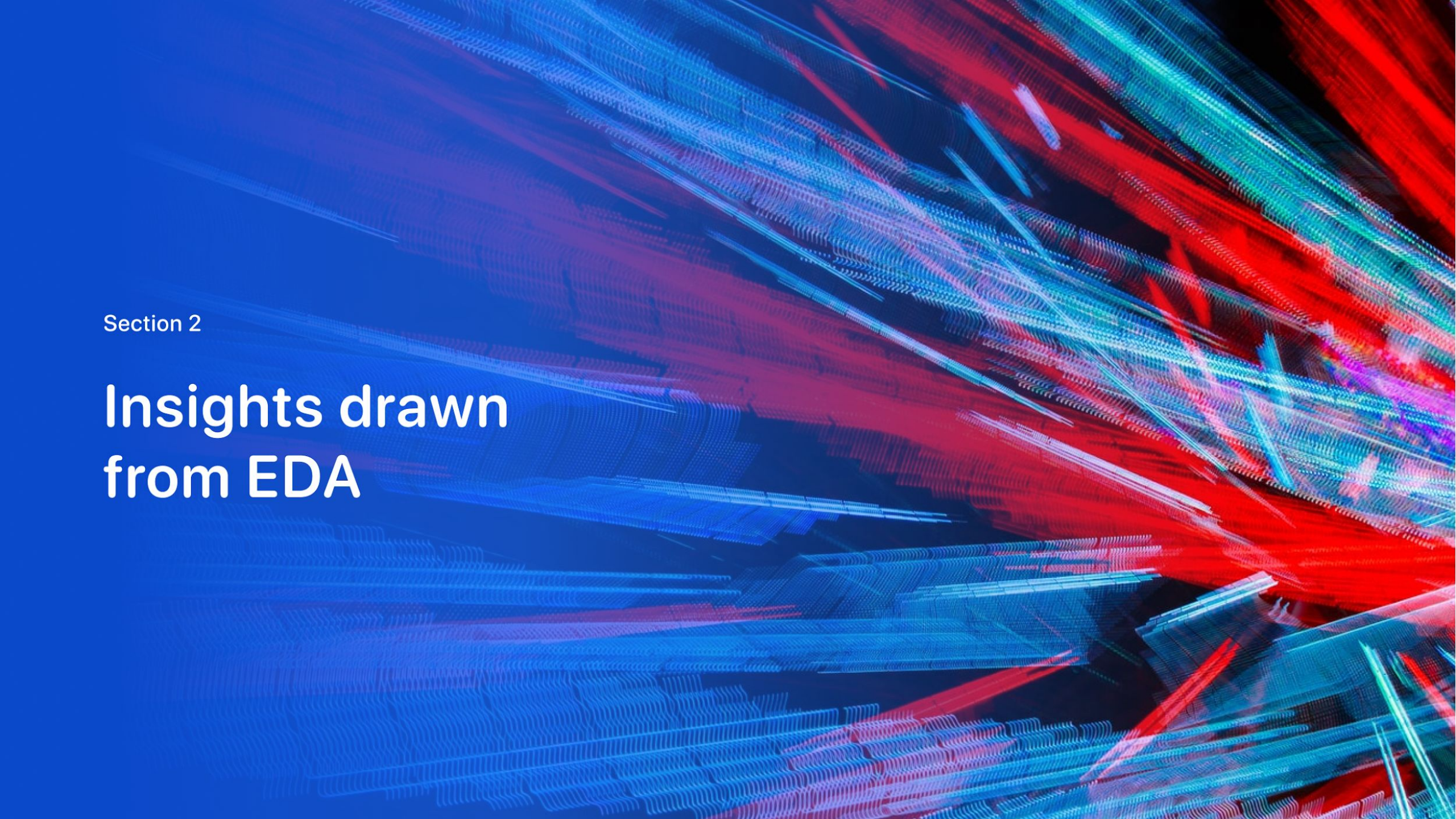
- Loaded data using Pandas
- Created a NumPy array from the class column
- Standardized the data using StandardScaler()
- Split data into Train and Test sets
- Apply four different machine learning algorithms to our train/test split
- Tune hyperparameters using GridSearchCV

Improvement & Evaluation

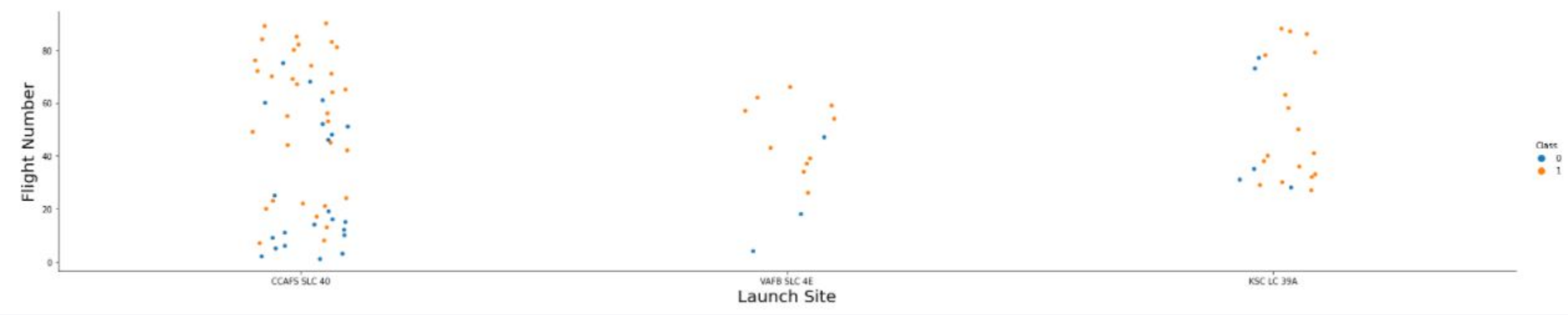
- Tuning & Feature Engineering used to improve models
- Confusion matrix is plotted for each algorithm
- Accuracy is used to determine which model works best

Results

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

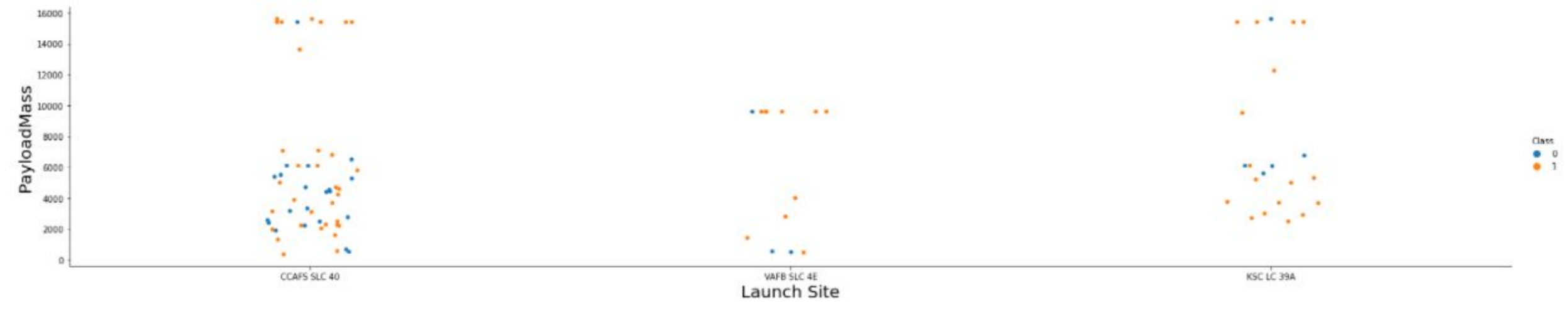

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



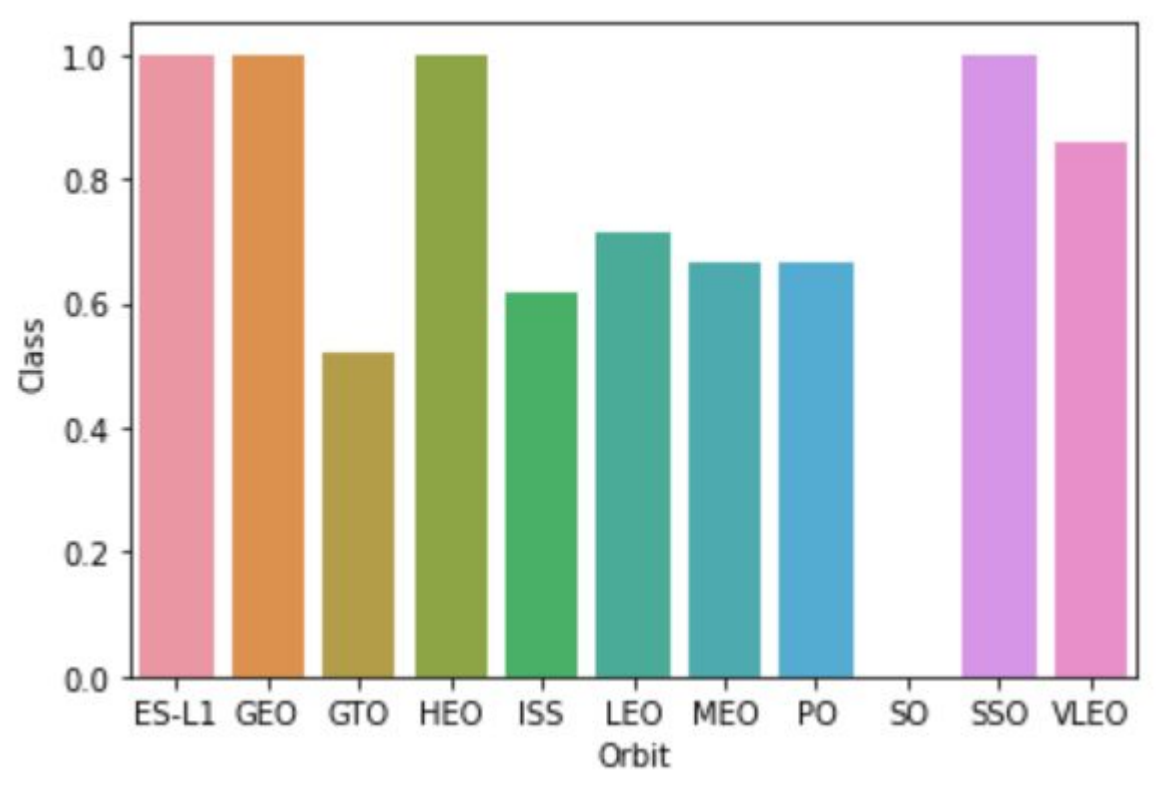
We found a higher success rate when the number of flights launched from a particular launch site increases. This is most visible when looking at CCAFS SLC40. Of the first 15 launches, only 3 were successful. Of the last 15 launches, only 1 was unsuccessful.

Payload vs. Launch Site



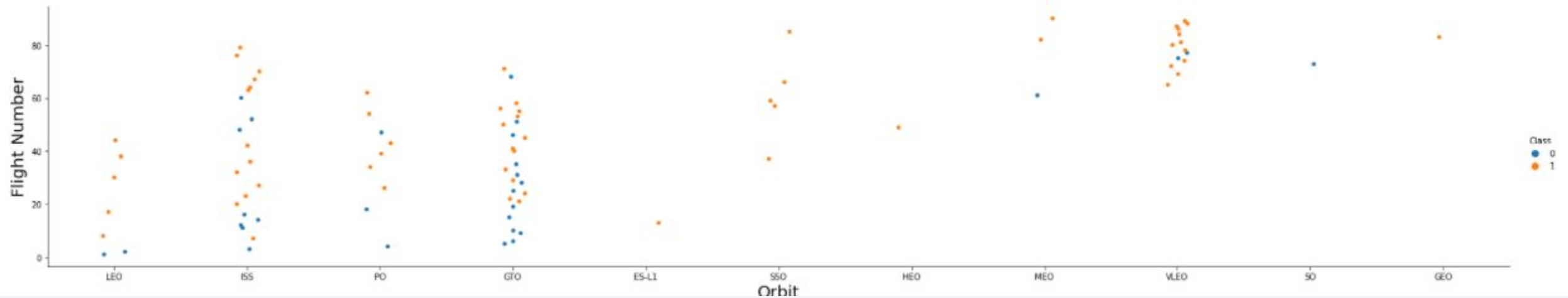
We found a higher success rate for CCAFS SLC 40 when the payload mass increased. Overall, it was unclear if there was an actual relationship between payload mass and launch site.

Success Rate vs. Orbit Type



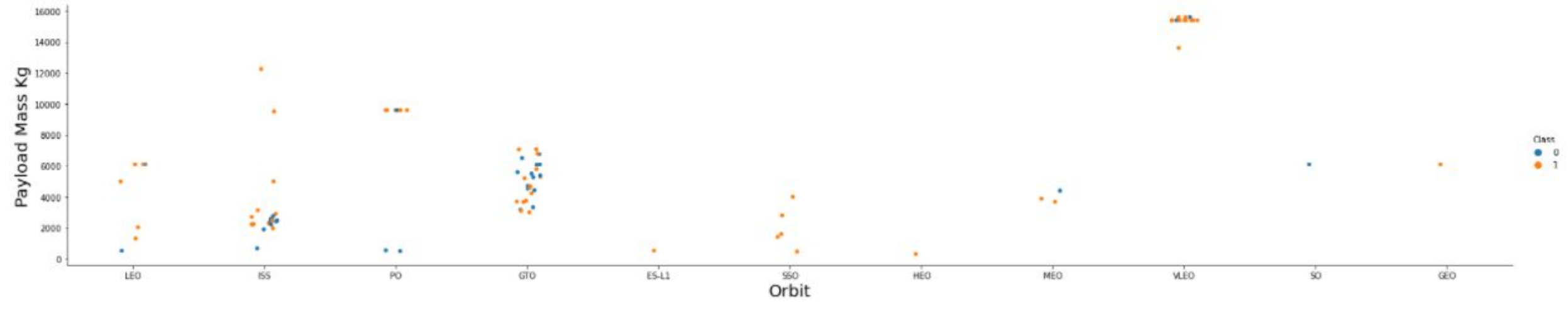
ES-L1, GEO, HEO, and SSO all had the highest success rate.

Flight Number vs. Orbit Type



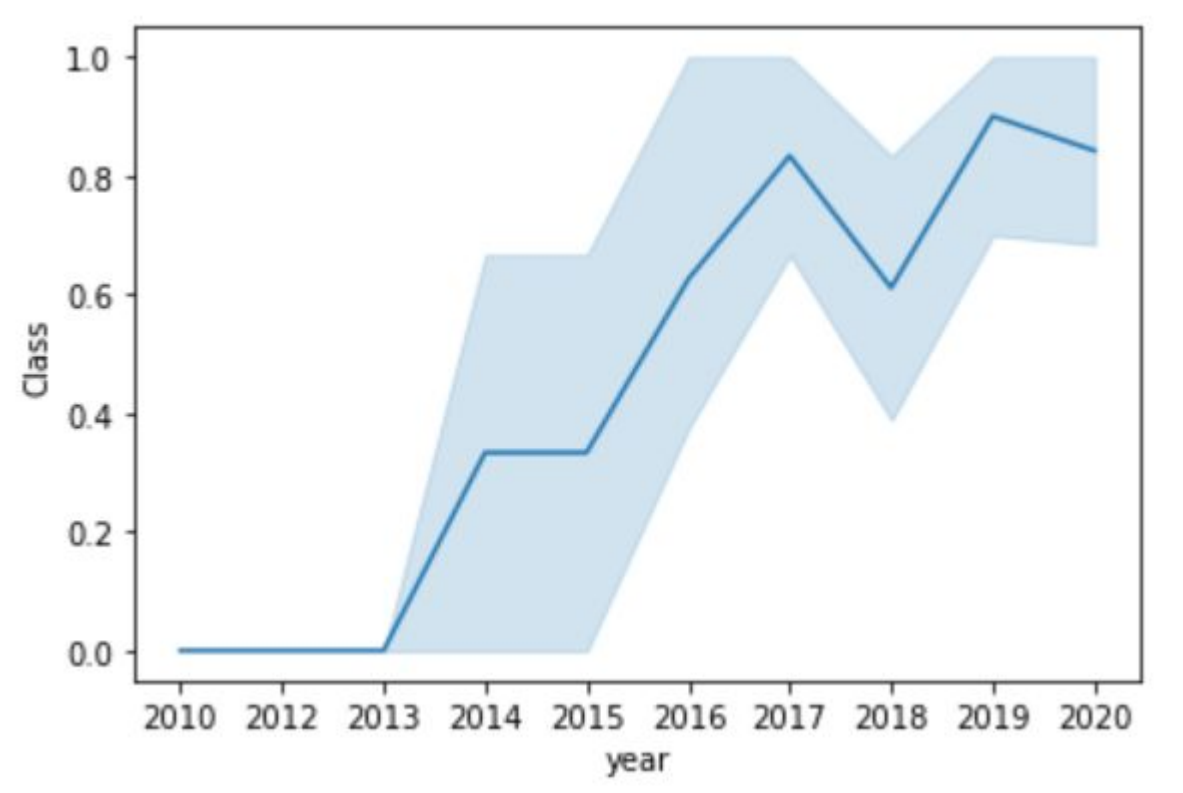
In the LEO orbit, the Success appears to be related to the number of flights; while there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type



For PO, LEO, and ISS, heavier payloads were related to a higher success rate. With GTO, we are unable to distinguish if there is a relationship between payload and success.

Launch Success Yearly Trend



We can see from the trend that since 2013, the success rate has continued to climb year over year with small dips sprinkled in.

All Launch Site Names

```
%sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL
```

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

By adding **DISTINCT**, we are able to only return the unique launch site names from the SPACEXTBL.

Launch Site Names Begin with 'CCA'

```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE '%CCA%' LIMIT 5
```

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

By using **LIKE** and the **'%CCA%'** wildcard, we are able to only select rows where the launch site contains "CCA". We add the **LIMIT 5** to the end so that we only return the first five rows.

Total Payload Mass

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)'
```

Using the **SUM** function on PAYLOAD_MASS_KG_ returns the sum of the column and the **WHERE** clause filters the rows to only contain customers equal to “NASA (CRS)”.

Average Payload Mass by F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1'
```

Using the **AVG** function on PAYLOAD_MASS_KG_ returns the average of the column and the **WHERE** clause filters the rows to only contain booster versions equal to “F9 v1.1”.

First Successful Ground Landing Date

```
%sql SELECT MIN(DATE) FROM SPACEXTBL WHERE LANDING__OUTCOME = 'Success (ground pad)'
```

1
2015-12-22

Using the **MIN** function on DATE returns the minimum (or first) from the column and the **WHERE** clause filters the rows to only contain landing outcomes equal to “Success (ground pad)”.

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_ > 4000 AND PAYLOAD_MASS__KG_ < 6000
```

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

Three statements in the **WHERE** clause are joined together using **AND** to filter the landing outcome by “Success (drone ship)” and payload mass greater than 4000 and payload mass less than 6000.

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT COUNT(MISSION_OUTCOME) FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE '%Success%'
```

1
100

```
%sql SELECT COUNT(MISSION_OUTCOME) FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE '%Failure%'
```

1
1

Using the **COUNT** function on MISSION_OUTCOME returns the count from the column and the **WHERE** clause filters the rows to only contain mission outcomes **LIKE** “Success” or “Failure” wildcard.

Boosters Carried Maximum Payload

```
%sql SELECT PAYLOAD_MASS__KG_, BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ IN (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

payload_mass__kg_	booster_version
15600	F9 B5 B1048.4
15600	F9 B5 B1049.4
15600	F9 B5 B1051.3
15600	F9 B5 B1056.4
15600	F9 B5 B1048.5
15600	F9 B5 B1051.4
15600	F9 B5 B1049.5
15600	F9 B5 B1060.2
15600	F9 B5 B1058.3
15600	F9 B5 B1051.6
15600	F9 B5 B1060.3
15600	F9 B5 B1049.7

Using the **MAX** function on PAYLOAD_MASS__KG in a subquery, we are able to **SELECT** the booster versions that have carried a payload mass equal to this maximum value.

2015 Launch Records

```
%sql SELECT * FROM SPACEXTBL WHERE YEAR(DATE) = 2015 AND LANDING__OUTCOME = 'Failure (drone ship)'
```

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2015-01-10	09:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)

Two statements in the **WHERE** clause are joined together using **AND** to filter the landing outcome by “Failure (drone ship)” and the extracted year from DATE equal to 2015.

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

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

landing__outcome	2
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

The LANDING_OUTCOME and **COUNT(**LANDING_OUTCOME) were selected from SPACEXTBL. In the **WHERE** clause, we **CAST** the date range of interest as datetime. We then **GROUP BY** the LANDING__OUTCOME and **ORDER BY** the count of the LANDING_OUTCOME in **DESC** order.

Section 4

Launch Sites Proximities Analysis

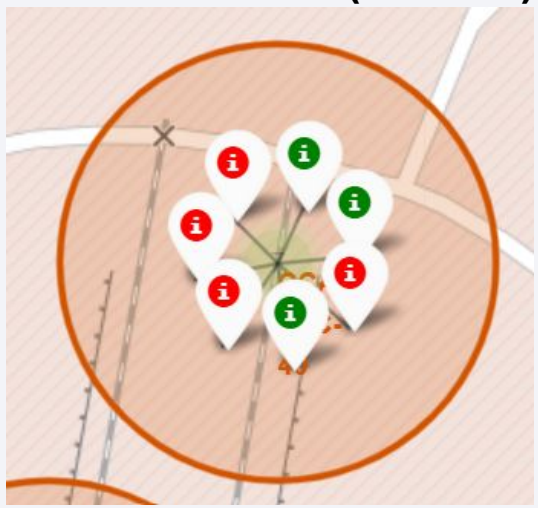
Global Map Markers of Each Launch Site

We see that all launch sites are near the equator and close to the coast in both California and Florida

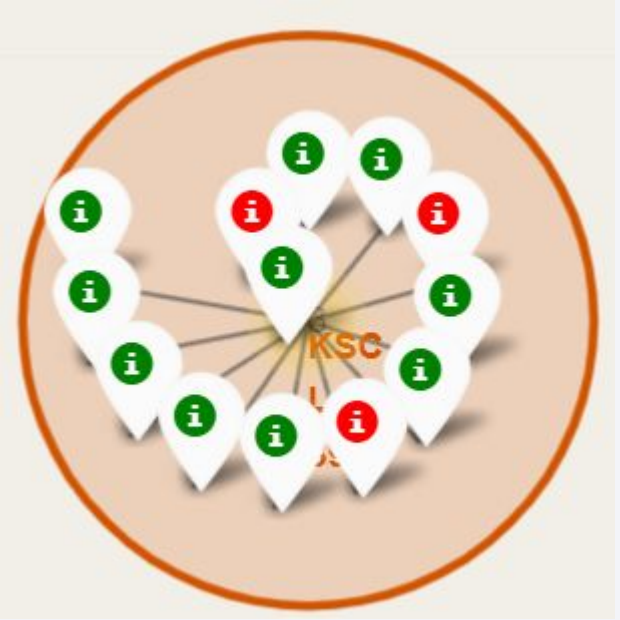
Launch Success & Failure Markers

VAFB SLC 4E (California)

CCAFS SLC-40 (Florida)



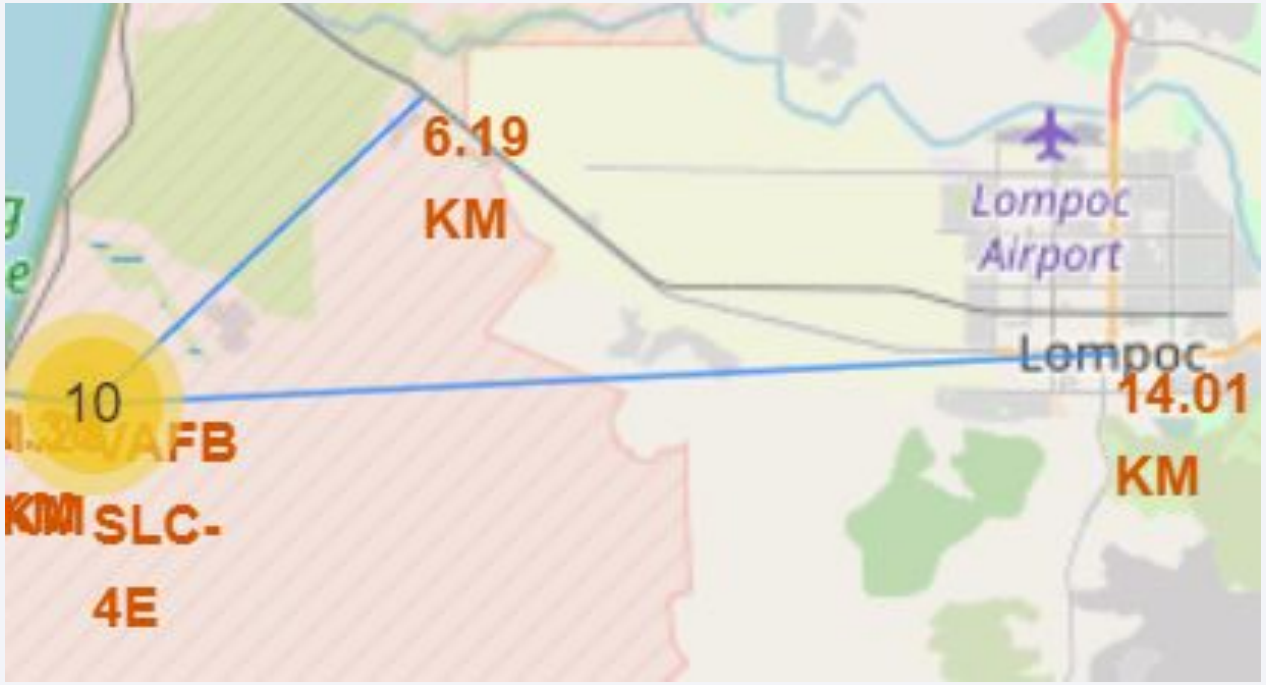
CCAFS LC-40 (Florida)



KSC LC-39A

Green markers indicate a successful missions, while Red markers indicate a failed mission.

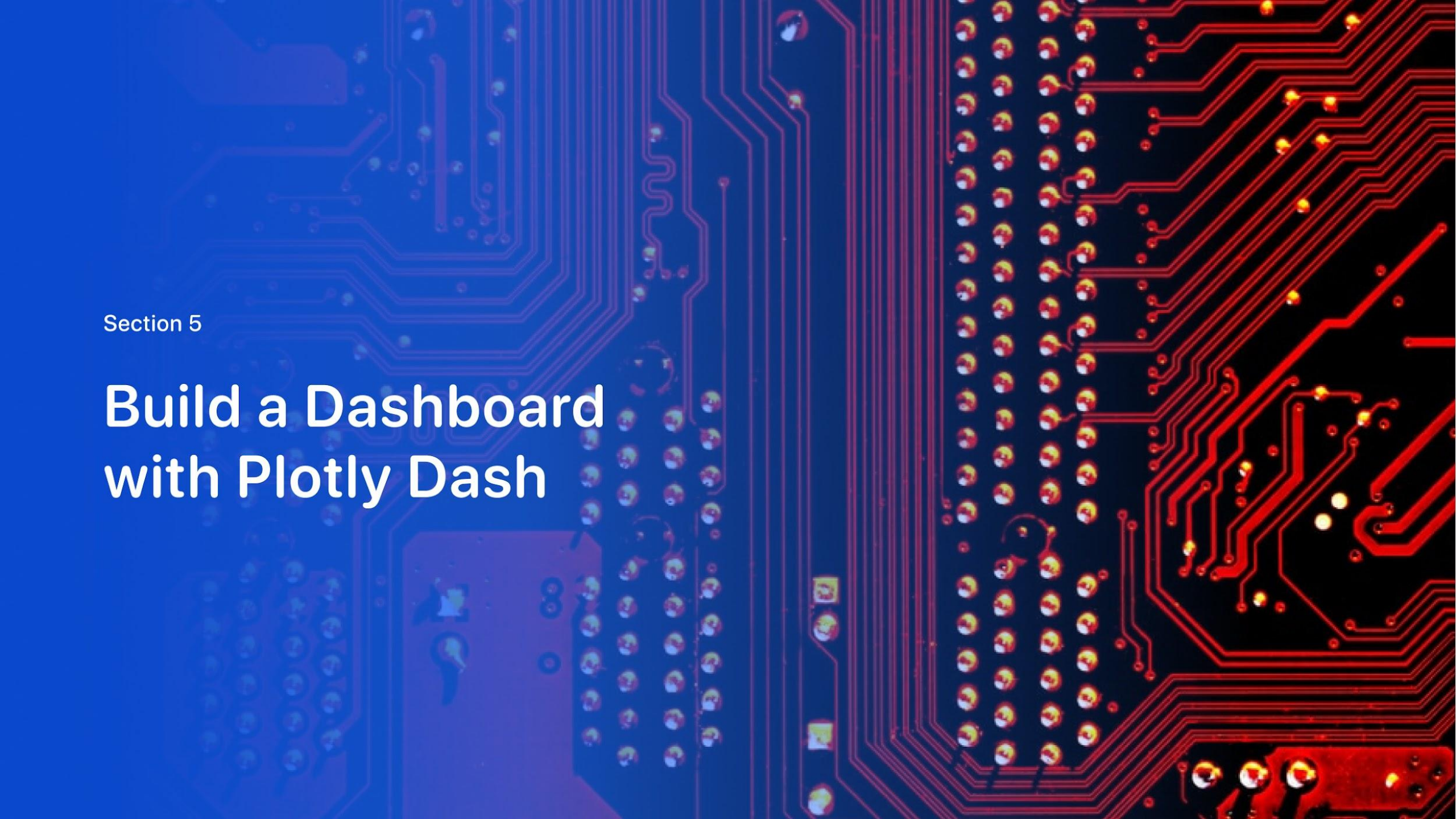
Launch Site Distances to Landmarks



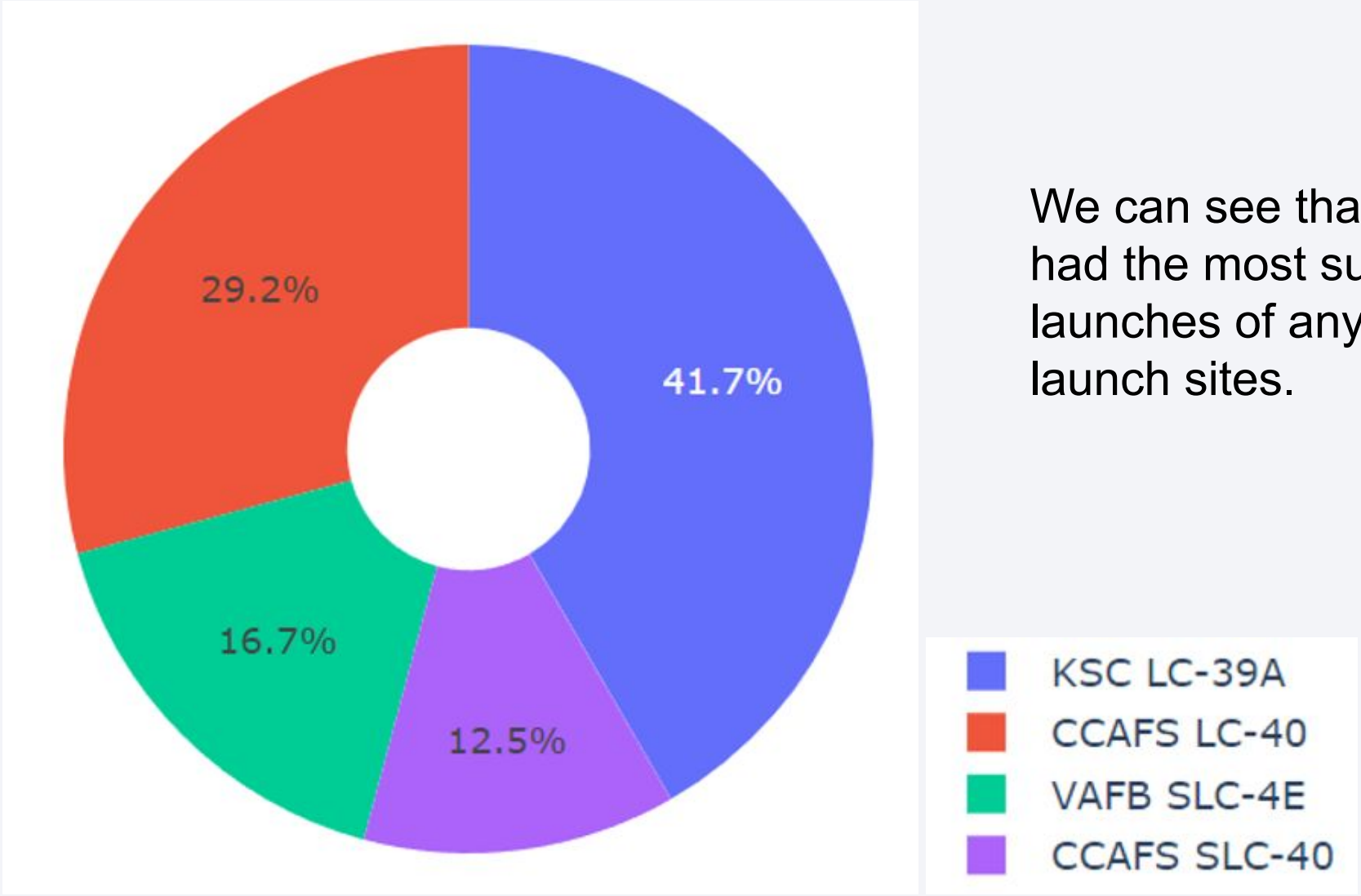
Distance to the coast 1.38KM
Distance to railway 1.28KM
Distance to the highway 6.19KM
Distance to nearest city 14.01KM

Example of VAFB SLC-4E (California)

All of the launch sites follow the same trend. Close to the coast and to railways, but further away from highways and cities.

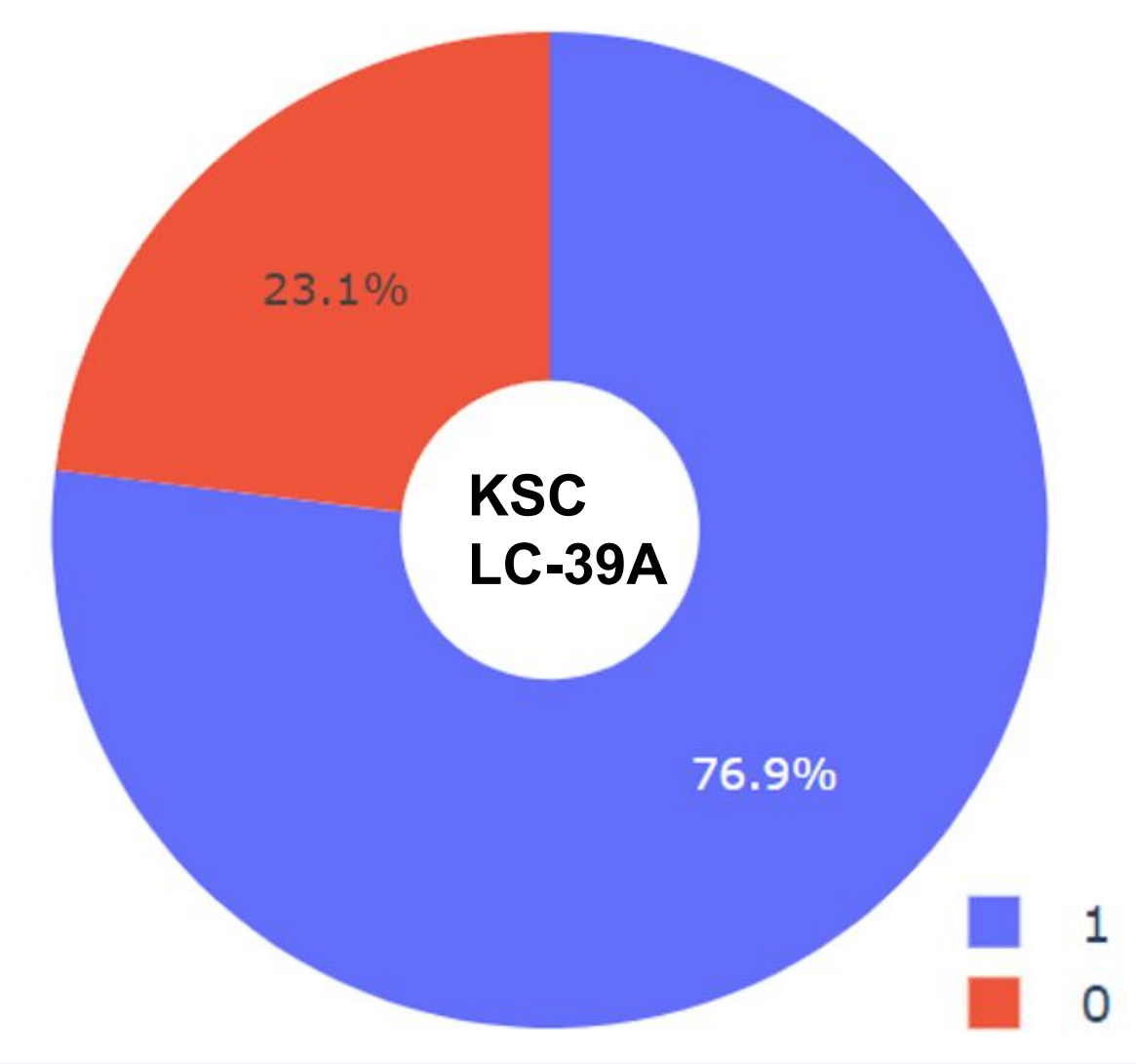

Section 5
Build a Dashboard with Plotly Dash

Plotly Dash - Success Percent Pie Chart



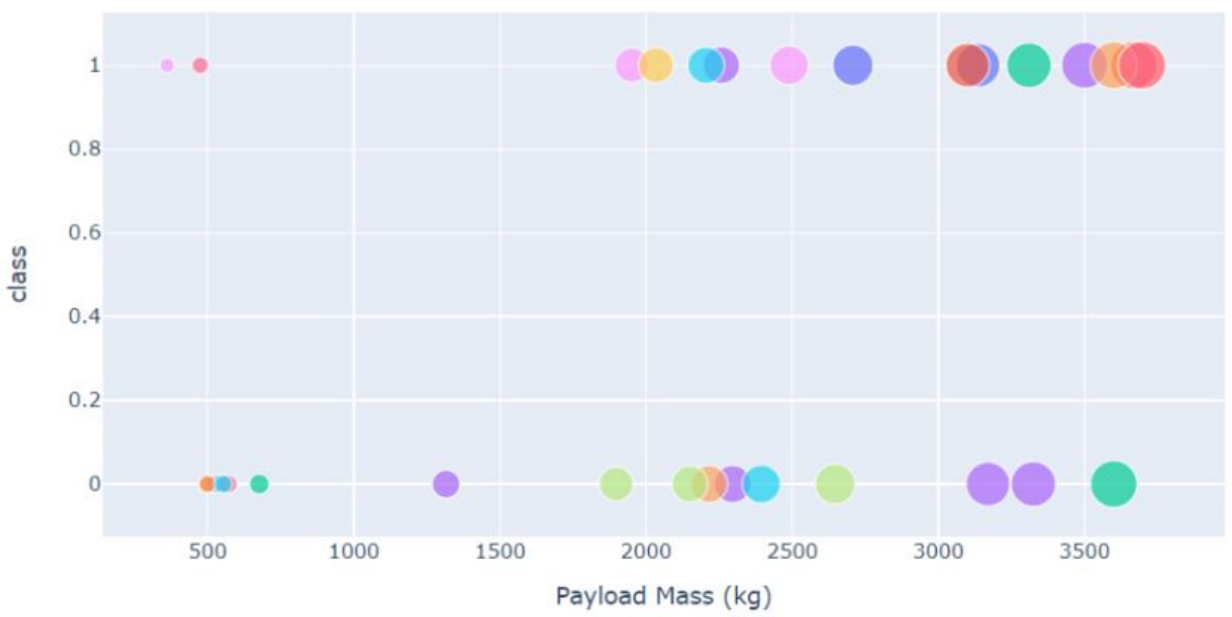
We can see that KSC LC-39A had the most successful launches of any of the different launch sites.

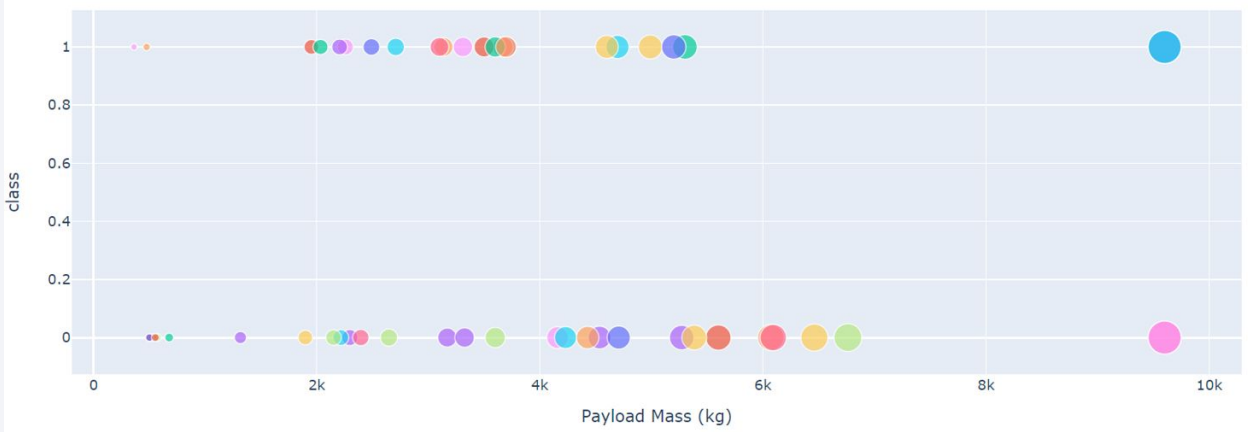
Plotly Dash - Most Successful Launch Site



KSC LC-39A had a success rate of 76.9% and a failure rate of 23.1%.

Plotly Dash - Outcome vs Payload Mass



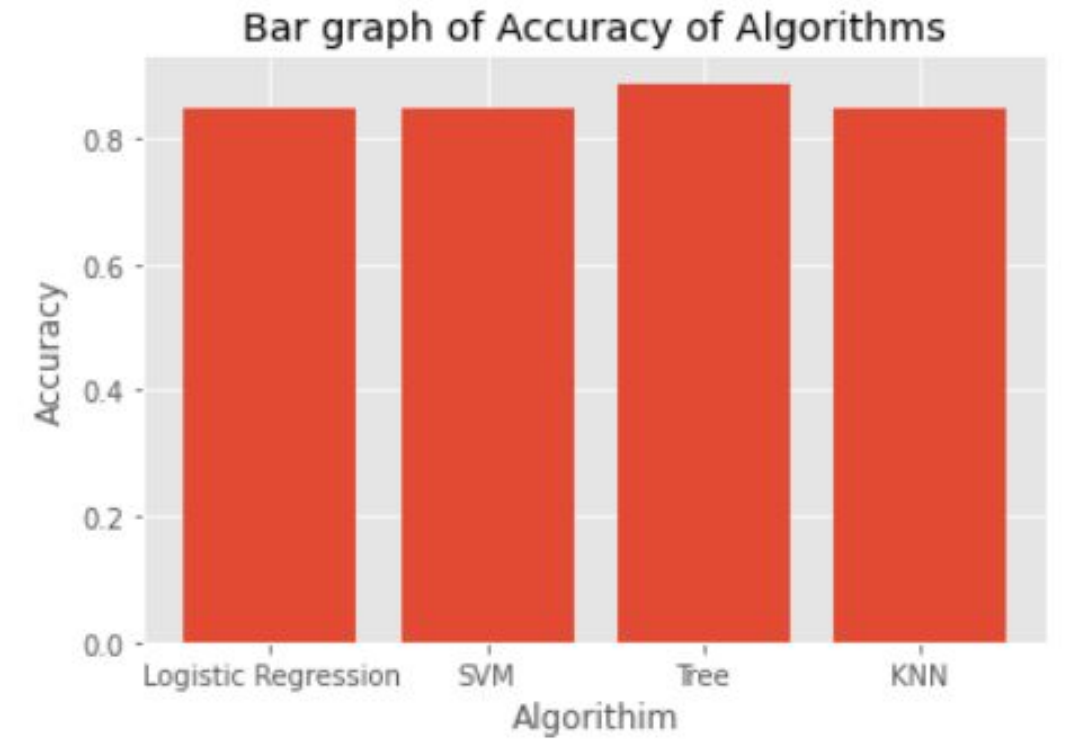


Lighter Payloads had a higher success rate than heavier payloads. This is particularly noticeable for payload masses above 4000 KG.

Section 6

Predictive Analysis (Classification)

Classification Accuracy

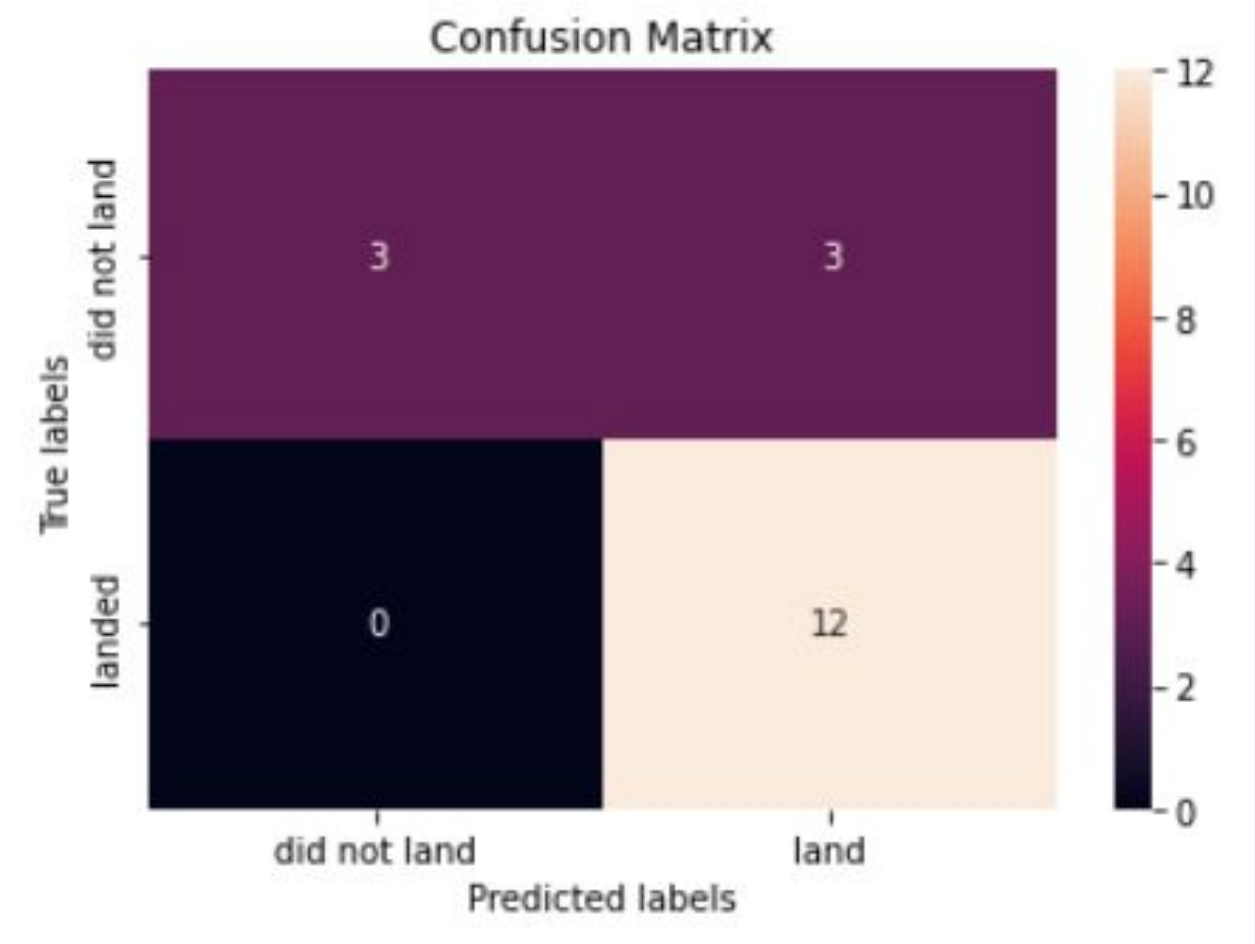


Of the four models we tested, the Tree Classifier had the highest classification accuracy at 0.8875.

```
bestalgorithm = max(algorithms, key=algorithms.get)
```

```
Best Algorithm is Tree with a score of 0.8875
Best Params is : {'criterion': 'entropy', 'max_depth': 12, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
```

Confusion Matrix



The confusion matrix shows the Tree Classifier did a good job predicting the successful and unsuccessful landings. There was an issue with false positives, but there were no false negatives.

Conclusions

- As a launch site launches more flights, the success rate increases
- Launch success has increased since 2013
- ES-L1, GEO, HEO, and SSO were the most successful orbits
- KSC LC-39A had the most successful launches
- Lighter payloads had a higher success rate - particularly below 4000 KG
- The Decision Tree Classifier algorithm had most success predicting outcomes

Appendix

```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_, 'SVM':svm_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
if bestalgorithm == 'SVM':
    print('Best Params is :', svm_cv.best_params_)
```

Code for finding the best algorithm

Appendix

View all of the notebooks and files on GitHub

Thank you!