Winning Space Race with Data Science

Jack Crago

29th October 2023

Outline

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

Executive Summary

Methodology

- Data collection – collected data from the SpaceX REST API as well as from scraping relevant wiki pages
- Data wrangling – dealt with missing values, encoded categorical variables for classification and filtered the data
- Exploratory data analysis – used SQL and Pandas to evaluate and visualize the relationships between the variables
- Mapping the data – used Folium to conduct geospatial analytics
- Interactive dashboard – used Plotly to create an interactive dashboard visualizing launch site success
- Predictive analysis (classification) – standardized the data before training and comparing different classification models

Summary of results

- The rate of success typically increases with more flights from a launch site
- Launch site KSC-LC-39A had the highest success rate with 76.9%
- The best performing classification model was a decision tree with an accuracy of 94%

Introduction

- SpaceX is one of the most successful commercial space age companies
- The SpaceX Falcon 9 rocket is considerably cheaper than other providers (Falcon 9 is advertised at $62m compared to other providers which can cost upwards of $165m)
- These savings can be achieved as SpaceX can reuse the first stage of the launch
- We want to predict whether the first stage will land successfully allowing us to determine the cost of a launch

Section 1

Methodology

Methodology

Data Collection

- Data from the SpaceX REST API using Get requests and from relevant wiki pages through web scraping

Data Wrangling

- Dealt with missing values, encoded categorical variables for classification and filtered the data

Exploratory Data Analysis

- SQL queries allowed manipulation and evaluation of the dataset
- Pandas allowed evaluation and visualization of the relationships between the variables to determine relevant patterns

Data Visualization

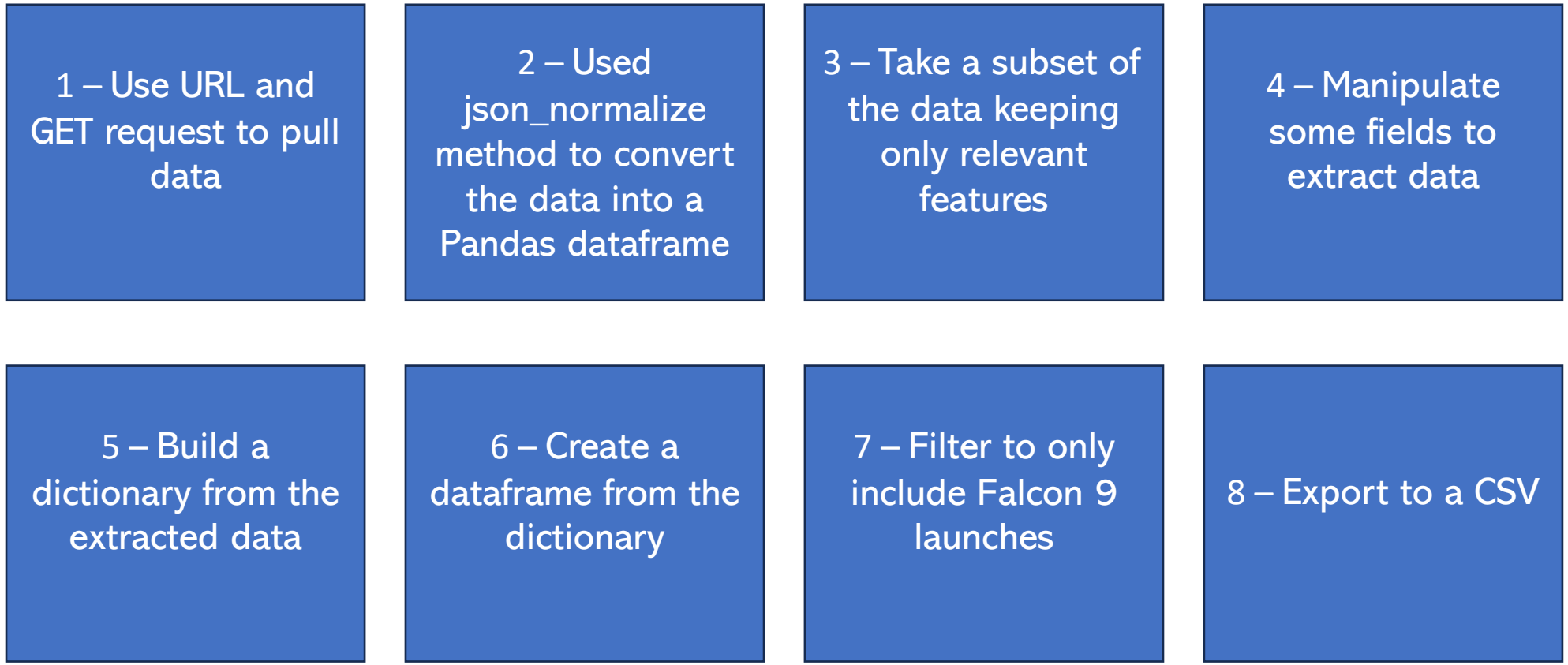
- Folium allowed geospatial analytics to understand the success rates of launch sites and the relationship to their environments
- Plotly allowed creation of an interactive dashboard to visualize launch site success and the relationship with other factors

Predictive Analysis

- Used Scikit-Learn to standardize the data and create and compare different classification models

Data Collection – SpaceX API

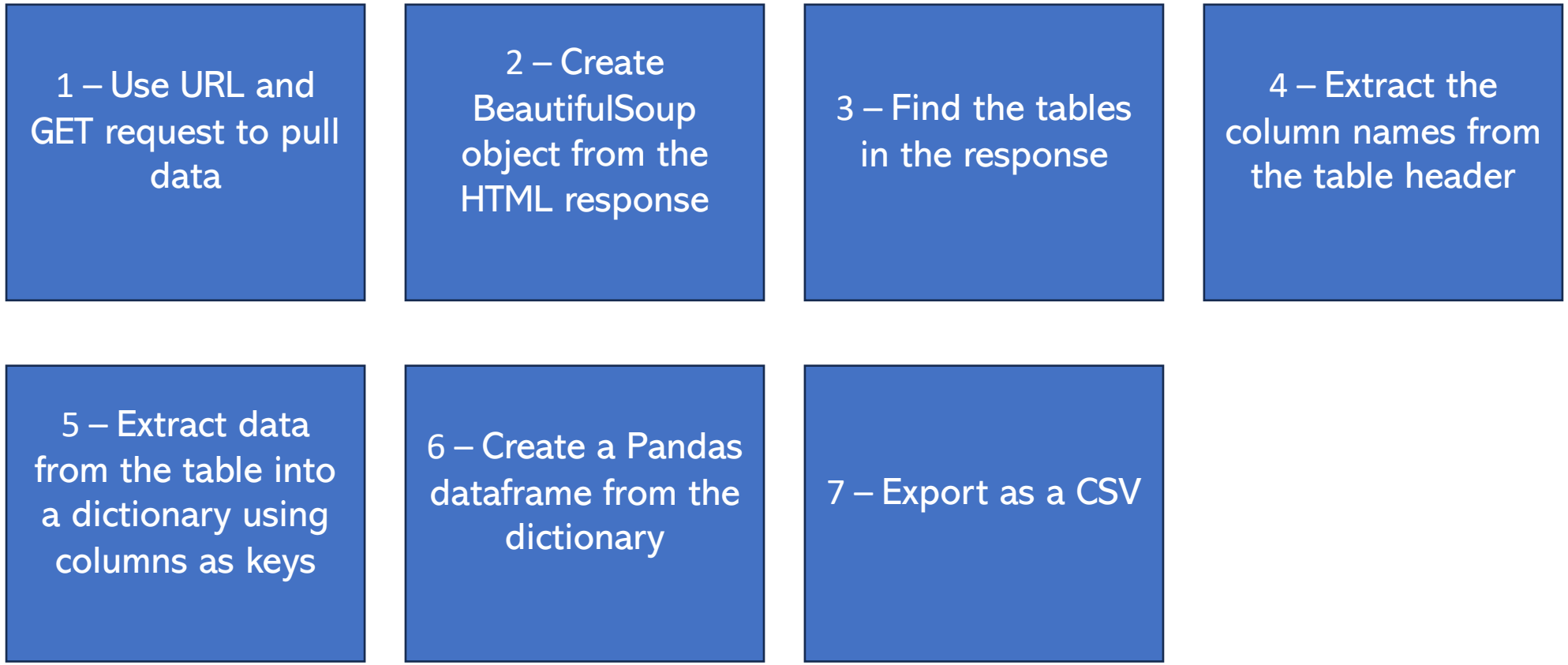
The flow chart shows the steps taken to pull the data from the SpaceX API and manipulate it into a usable format



This link shows the complete data collection via API notebook

Data Collection - Scraping

The flow chart shows the steps taken to extract the data from a webpage



This link shows the complete web scraping notebook

Data Wrangling

- The main objectives of the data wrangling process were to determine the training labels and explore the relationships between the variables
- The list below shows the order of the steps taken in this process

1. Identify the percentage of missing values in each field
2. Understand the number of occurrences of launch sites and orbits
3. Understand the number of occurrences of orbits and mission outcomes
4. Create a landing outcome label for the data set by analyzing the outcome fields
5. Export the manipulated dataset to a CSV file

This link shows the complete data wrangling notebook

EDA with Data Visualization

- The following charts were plotted to explore the data
  - Flight number and payload mass with color showing outcome – we see the outcome is more likely to be positive as flight number increases
  - Launch site vs flight number with color showing outcome – success varies by launch site
  - Launch site vs payload mass – VAFB-SLC launch site does not have heavy payloads
  - Success rate by orbit – ES-L1, GEO, HEO and SSO have highest success rate
  - Flight number vs orbit – some orbits only occur with later flights. Success rate only appears related to flight number for some orbits
  - Payload mass vs orbit – not all orbits have heavy payload mass
  - Success rate by year – success rate generally improves over time
- This link shows the notebook with the EDA visualizations

EDA with SQL

The below SQL queries were used to better understand the data

1. Displayed a list of unique launch site names in the missions
2. Displayed 5 records where launch sites began with the string 'CCA'
3. Displayed the total payload mass carried by boosters launched by NASA (CRS)
4. Displayed the average payload mass carried by booster version F9 v1.1
5. Displayed the first successful ground pad landing
6. Listed the names of the boosters with success in drone ship and with payload mass between 4000 and 6000
7. Showed a count of successful and unsuccessful mission outcomes
8. Listed the booster versions that have carried the heaviest payload mass
9. Listed the records with month names, failure landing outcomes in drone ship ,booster versions, launch site for the year 2015
10. Ranked the count of landing outcomes between the dates 2010-06-04 and 2017-03-20, in descending order.

This link shows the notebook with the above SQL queries

Build an Interactive Map with Folium

The steps below were used to visualize the data on an interactive map

1. Initialize a map with a Folium map object
2. Add a circle and a marker to each launch site to show the location on the map
3. Show the successful and unsuccessful launches for each site on the map
    1. As the launches shared coordinates, it was necessary to cluster the markers
    2. To make the visual clearer, marker colors were used to indicate success and failure
    3. A MarkerCluster object was used to create the cluster
    4. The interactive nature of the map allows the user to explore the data at a higher zoom
4. Explore the distance between launch sites and other points of interest
    1. Using the cursor position we were able to determine the coordinates of other locations (cities, roads, railways and the coast)
    2. Using these coordinates, we could calculate the distance between the launch site and these locations

This link shows the notebook with the interactive maps described above

Build a Dashboard with Plotly Dash

An interactive dashboard was created to explore the data with the below plots

1. Pie chart showing proportion of successful and unsuccessful launches
    1. An interactive control allows the user to select the launch site
    2. The success rate of different launch sites can be compared
    3. This is an important factor when considering predicting a new launch
2. Scatter chart showing the relationship between success, and payload mass and booster type
    1. An interactive slider control allows the user to filter the payload mass
    2. These variables are important to consider when predicting a future launch

This link shows a .py file that can be executed to create the dashboard

Predictive Analysis (Classification)

1 – Created a NumPy array from the column Class

2 – Standardized the data with the StandardScalar method

3 – Split the data into training and test sets

4 – Created a logistic regression object and a GridSearchCV object. Fit the object to find the best parameters

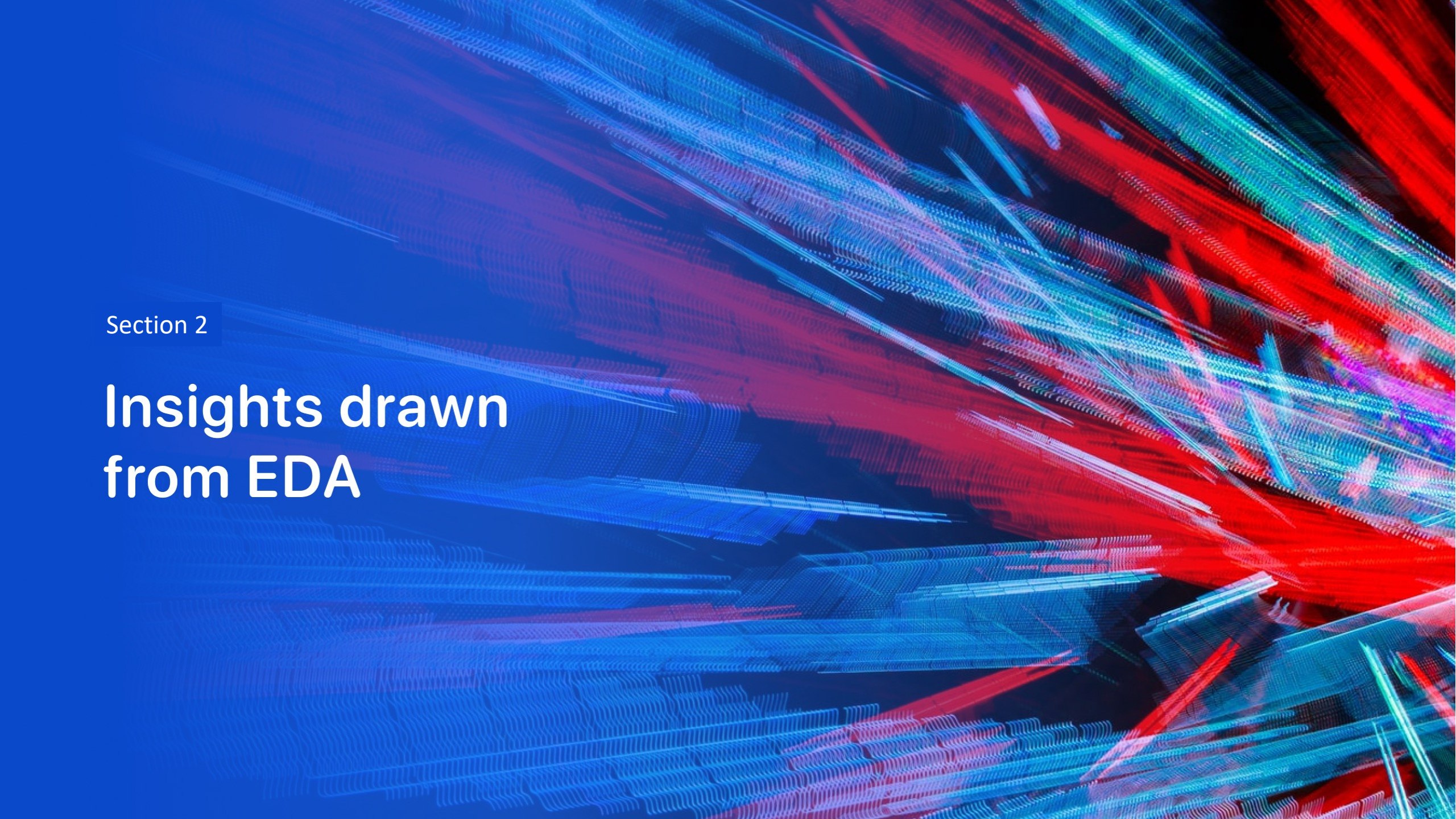
5 – Calculated the accuracy of the classifier on the test set

6 – Repeated steps 4 -5 for SVM, KINN and decision tree models

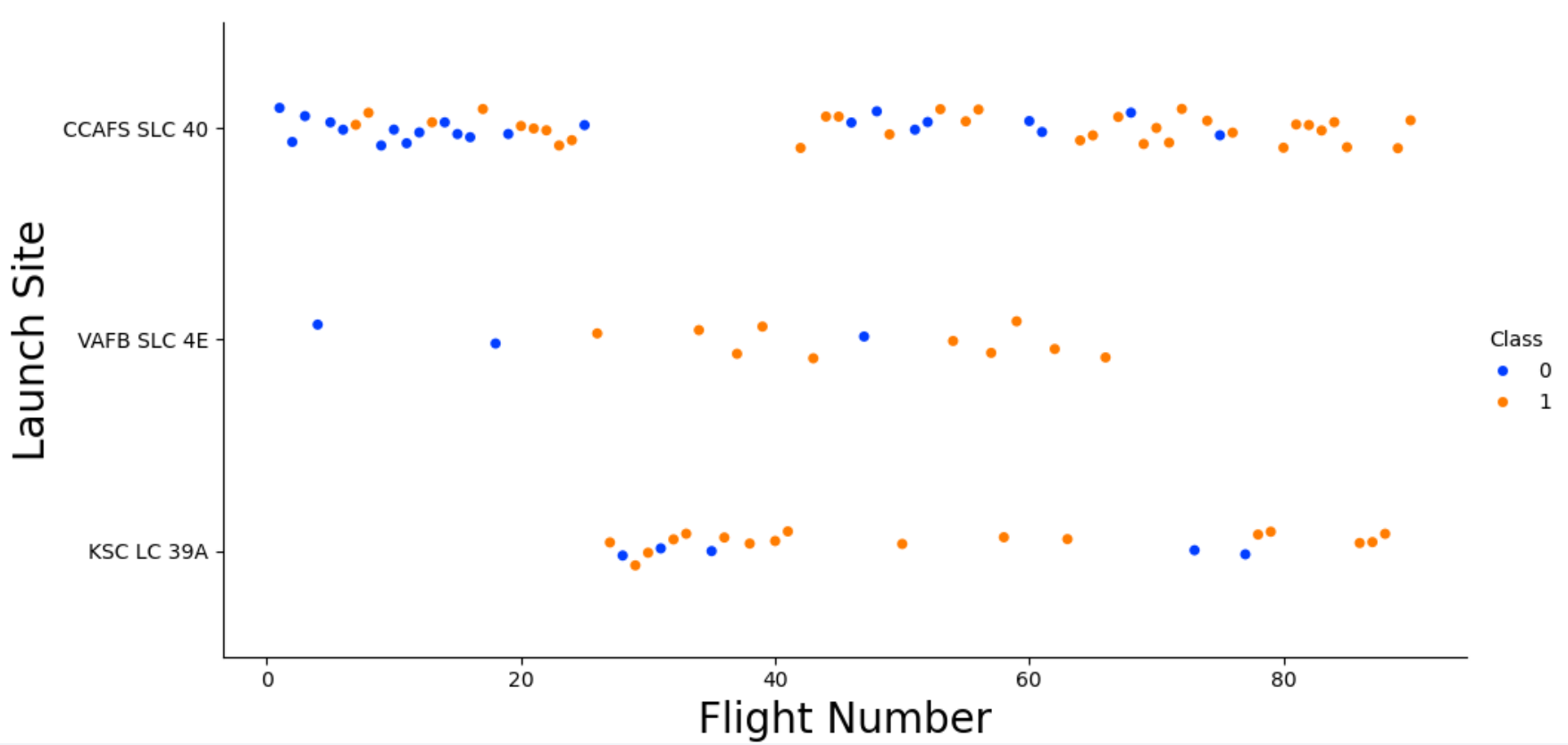
7 – Created a confusion matrix for each model to compare performance on the test set

8 – Compared the accuracy score of each model to determine the best model

This link shows the notebook with the interactive maps described above

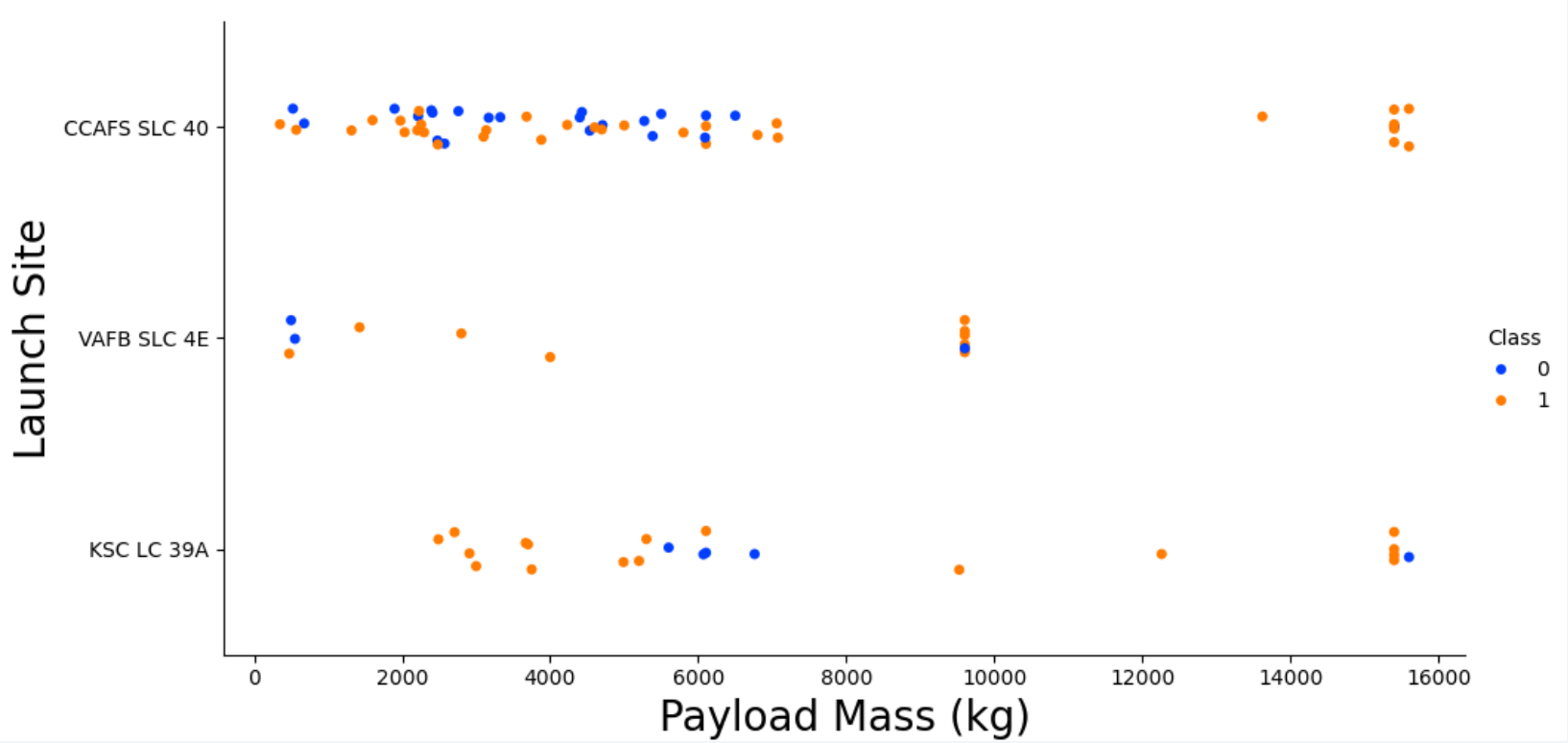

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



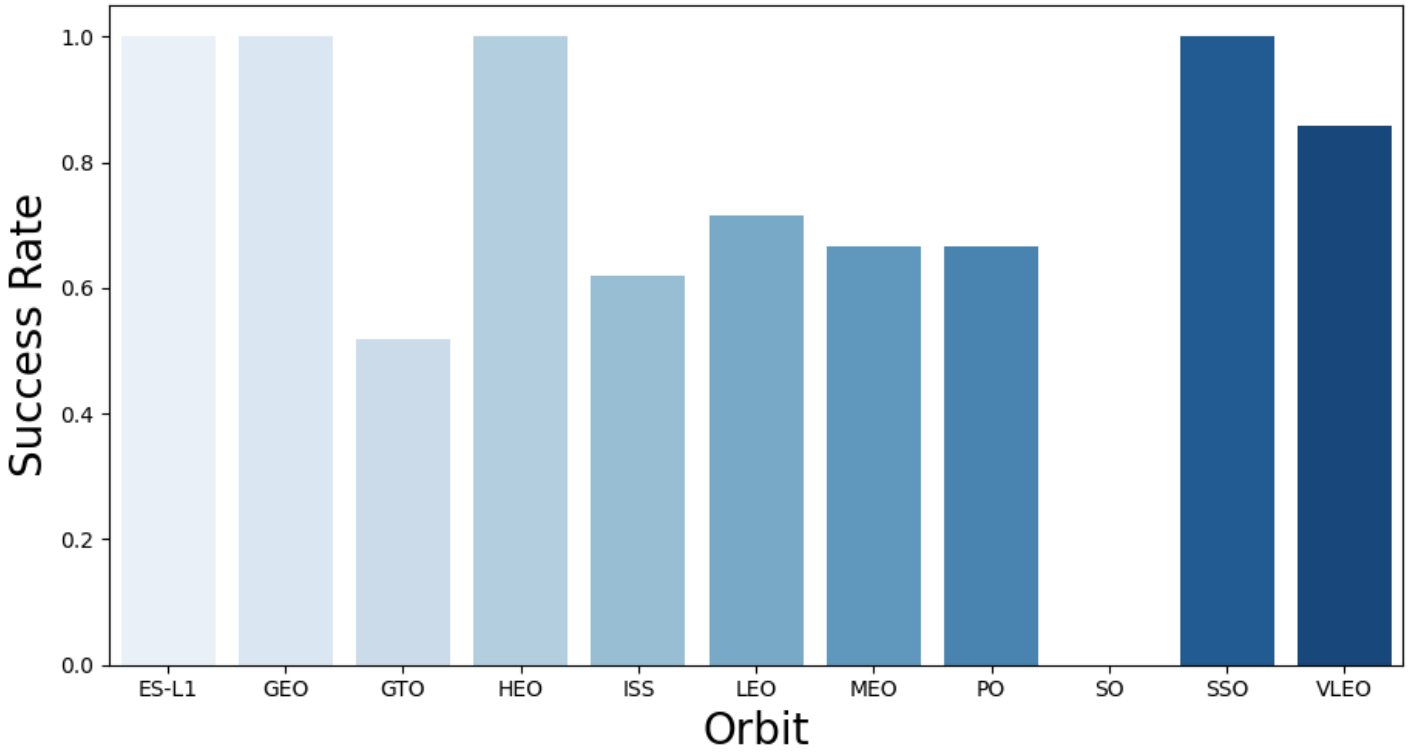
- Success rate increased with the number of flights
- The CCAFS SLC 40 launch site was the most common
- VAFB SLC 4E and KSC LC 39A have higher success rates as they were used for later flights

Payload vs. Launch Site



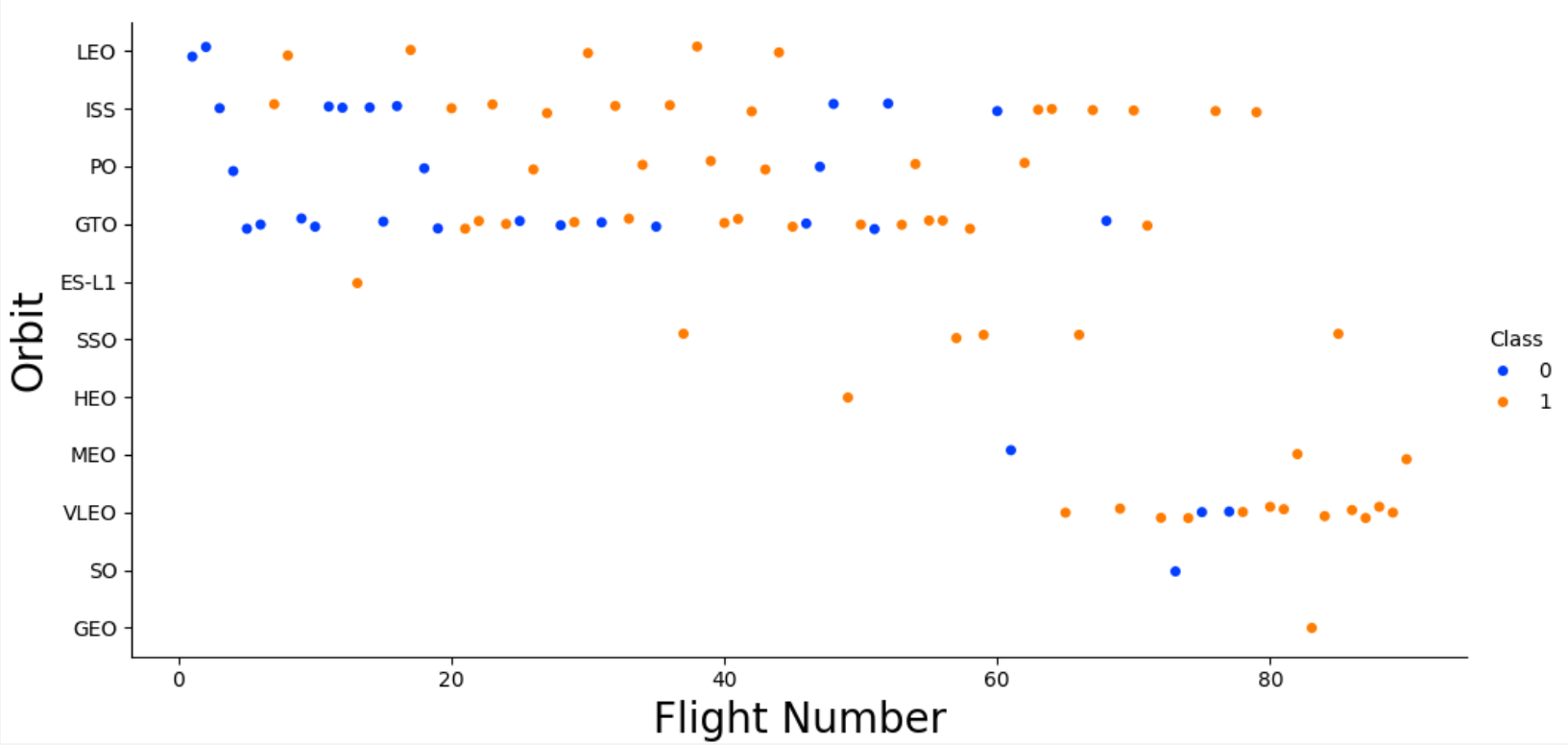
- Higher payload mass have higher success rates
- Most launches with a payload mass over 7000kg were successful
- Launches with a payload mass under 7000kg are more common

Success Rate vs. Orbit Type



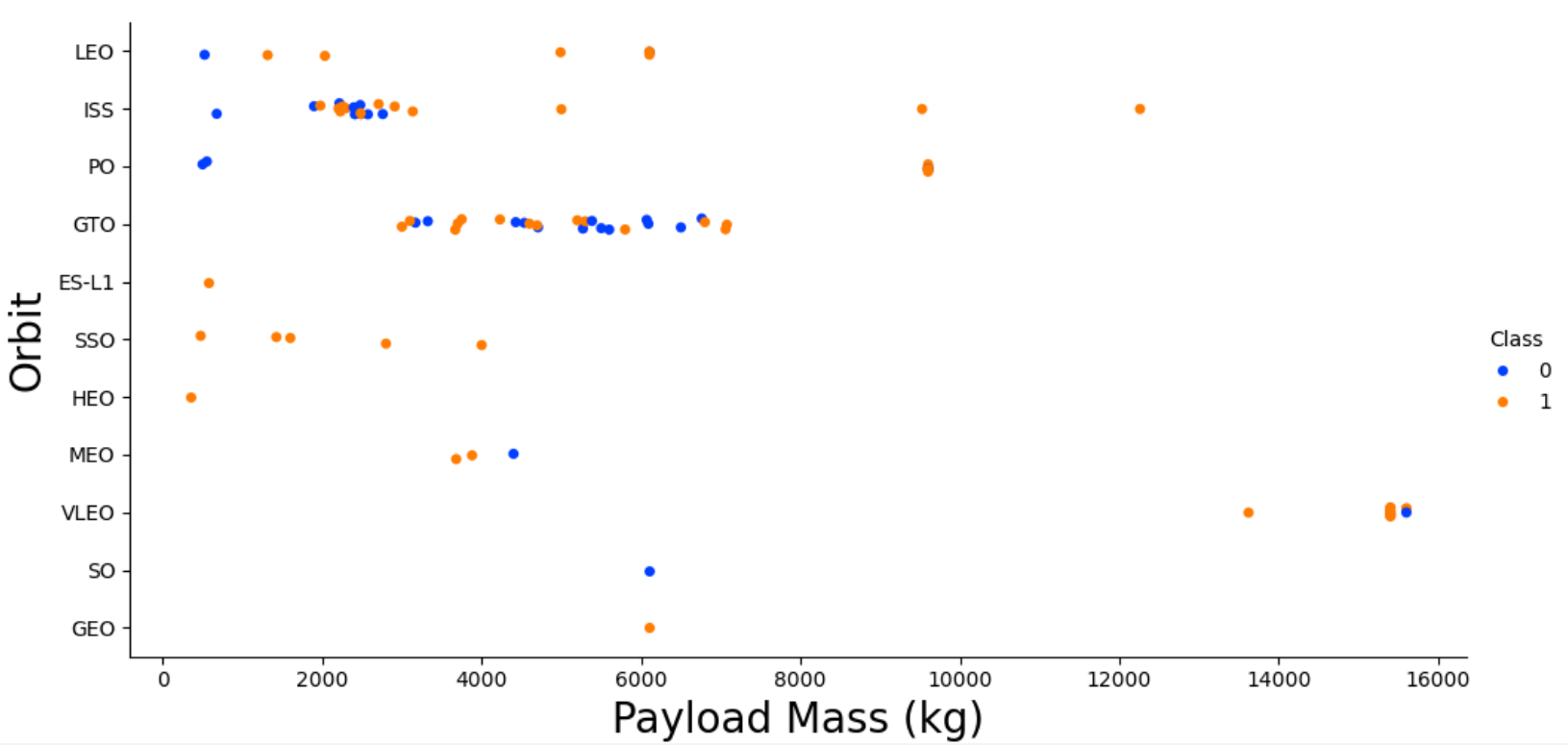
- The ES-L1, GEO, HEO and SSO orbits all have 100% success rate
- The SO orbit has a 0% success rate

Flight Number vs. Orbit Type



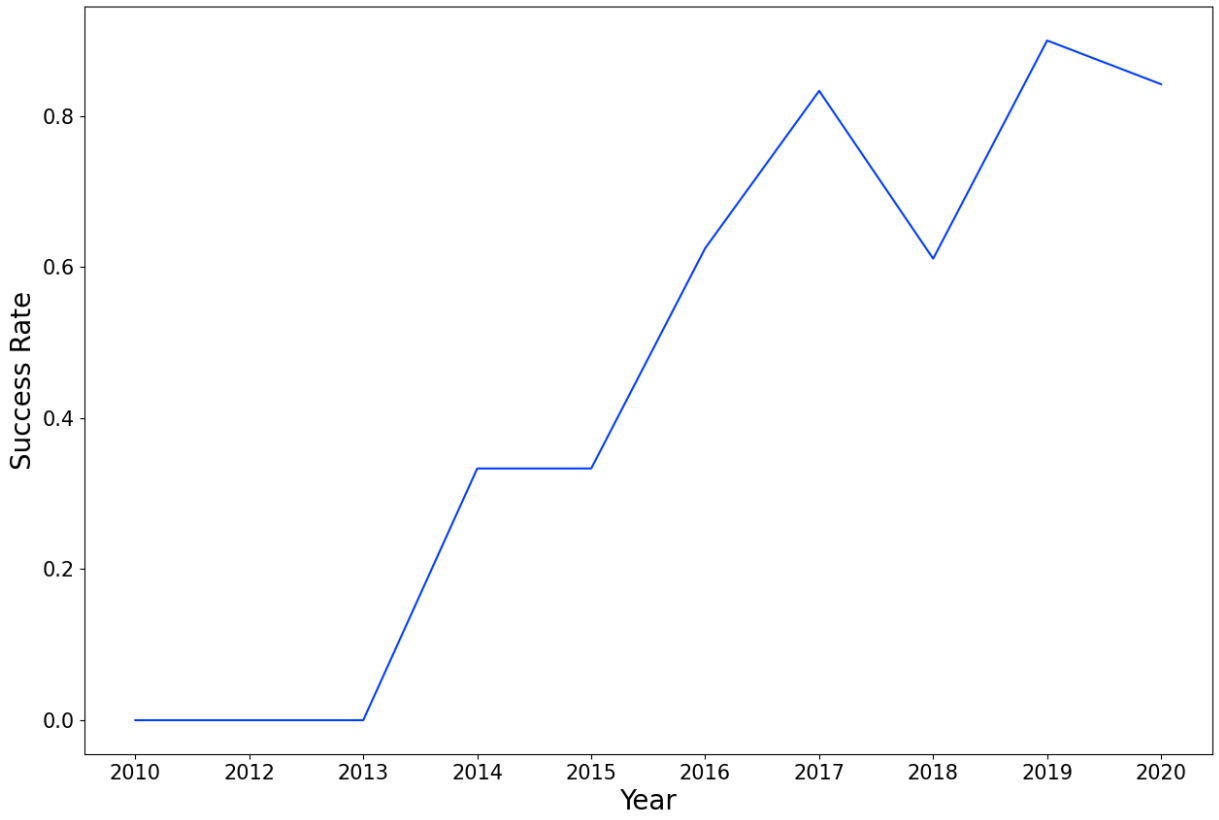
- The scatter plot adds context to the previous slide.
- The 100% success rates of ES-L1, HEO and GEO are a result of a single launch
- The 0% success rate of SO is also the result of a single launch
- The relationship between success and flight number is not present for the GTO orbit
- The pattern of success rate increasing with flight number is present for the other orbits

Payload vs. Orbit Type



- Only ISS, PO and VLEO have payloads over 7000kg
- The heaviest payloads typically have VLEO orbits
- The doesn't appear to be a relationship between payload mass and success for GTO orbits

Launch Success Yearly Trend



- The success rate for 2010 – 2013 was 0%
- After 2013, the success rate increased every year until 2017
- 2018 saw a dip in success rate
- The success rate increased to its highest level in 2019 before a small dip in 2020
- The success rate has been above 50% since 2016

All Launch Site Names

```
[10]: %sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE
 * sqlite:///my_data1.db
Done.
```

[10]:

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

- The DISTINCT keyword in the query returns unique values from the launch site column

Launch Site Names Begin with 'CCA'

```
[11]: %sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5
 * sqlite:///my_data1.db
Done.
```

[11]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- The WHERE and LIKE keywords along with the % symbol returns values that start with ‘CAA’. The use of Limit 5 restricts the result to 5 values

Total Payload Mass

```
[16]: %%sql
      SELECT SUM("PAYLOAD_MASS__KG_") AS total_payload_mass
      FROM SPACEXTABLE WHERE "Customer" LIKE '%NASA%'

       * sqlite:///my_data1.db
      Done.

[16]: total_payload_mass
      ----------------
                107010
```

- The sum keyword is used to calculate the total
- The results are filtered to customers containing ‘NASA’ with the WHERE and LIKE keywords

Average Payload Mass by F9 v1.1

```
[19]: %%sql
      SELECT ROUND(AVG("PAYLOAD_MASS__KG_"),0) AS average_payload_mass
      FROM SPACEXTABLE WHERE "Booster_Version" LIKE 'F9 v1.1%'

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

[19]:

average_payload_mass
2535.0

- The AVG keyword calculates the mean of the payload mass column
- The WHERE keyword is used with the LIKE keyword to filter the results

First Successful Ground Landing Date

```
[22]: %%sql
      SELECT MIN("Date") FROM SPACEXTABLE
      WHERE "Landing_Outcome" = 'Success (ground pad)'

       * sqlite:///my_data1.db
      Done.

[22]: MIN("Date")
      2015-12-22
```

- The MIN keyword returns the smallest value. In the case of dates, this returns the earliest date
- The WHERE keyword filters the results

Successful Drone Ship Landing with Payload between 4000 and 6000

[23]:
```
%%sql
SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (drone ship)' AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_"<6000
```

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

[23]:

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

- The WHERE keyword is used to filter the results. The AND keywords allow us to define multiple filtering criteria
- As before, the DISTINCT keyword returns unique values

Total Number of Successful and Failure Mission Outcomes

```
[24]: %%sql
SELECT "Mission_Outcome", count(*) AS 'missions'
FROM SPACEXTABLE GROUP BY 1
```

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

[24]:

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

- The COUNT(*) keyword returns a count of the rows returned.
- The GROUP BY 1 keyword groups the count by mission outcome

Boosters Carried Maximum Payload

```
[25]: %%sql
SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE
WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTABLE)
```

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

[25]:

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

- A subquery is used to filter the results. The SELECT statement within the brackets returns a value that is then used to filter the main query
- The MAX keyword returns the largest values
- The WHERE keyword then filters the query based on the result of the subquery

2015 Launch Records

[26]:
```
%%sql
SELECT SUBSTR("Date",6,2) AS month, "Booster_Version", "Launch_Site"
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Failure (drone ship)' AND SUBSTR("Date",0,5)='2015'
```

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

[26]:

month	Booster_Version	Launch_Site
10	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

- The SUBSTR keyword is used to take a substring from a longer string
- The WHERE and AND keywords are used to filter the results

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

```
[28]: %%sql
SELECT "Landing_Outcome", COUNT(*) AS missions FROM SPACEXTABLE
WHERE "Date" BETWEEN '2010-06-04' and '2017-03-20'
GROUP BY 1
ORDER BY 2 DESC
```

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

[28]:

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

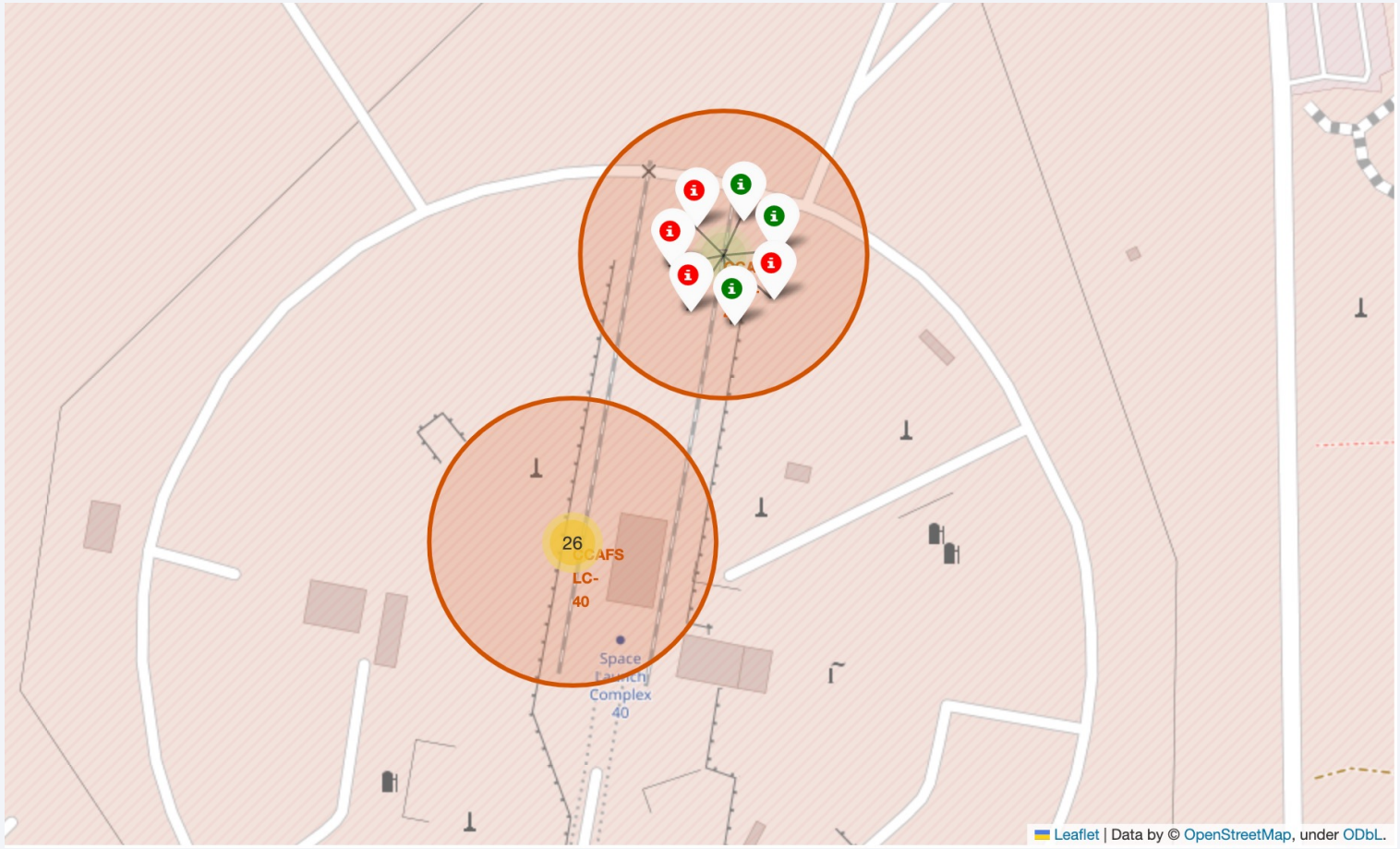
- The ORDER BY 2 DESC keyword is used to sort the results by the count of missions in descending order
- The GROUP BY 1 keyword groups the count by the landing outcome
- The BETWEEN keyword filters for dates within the range


Section 3
Launch Sites
Proximities Analysis

Launch Site Locations

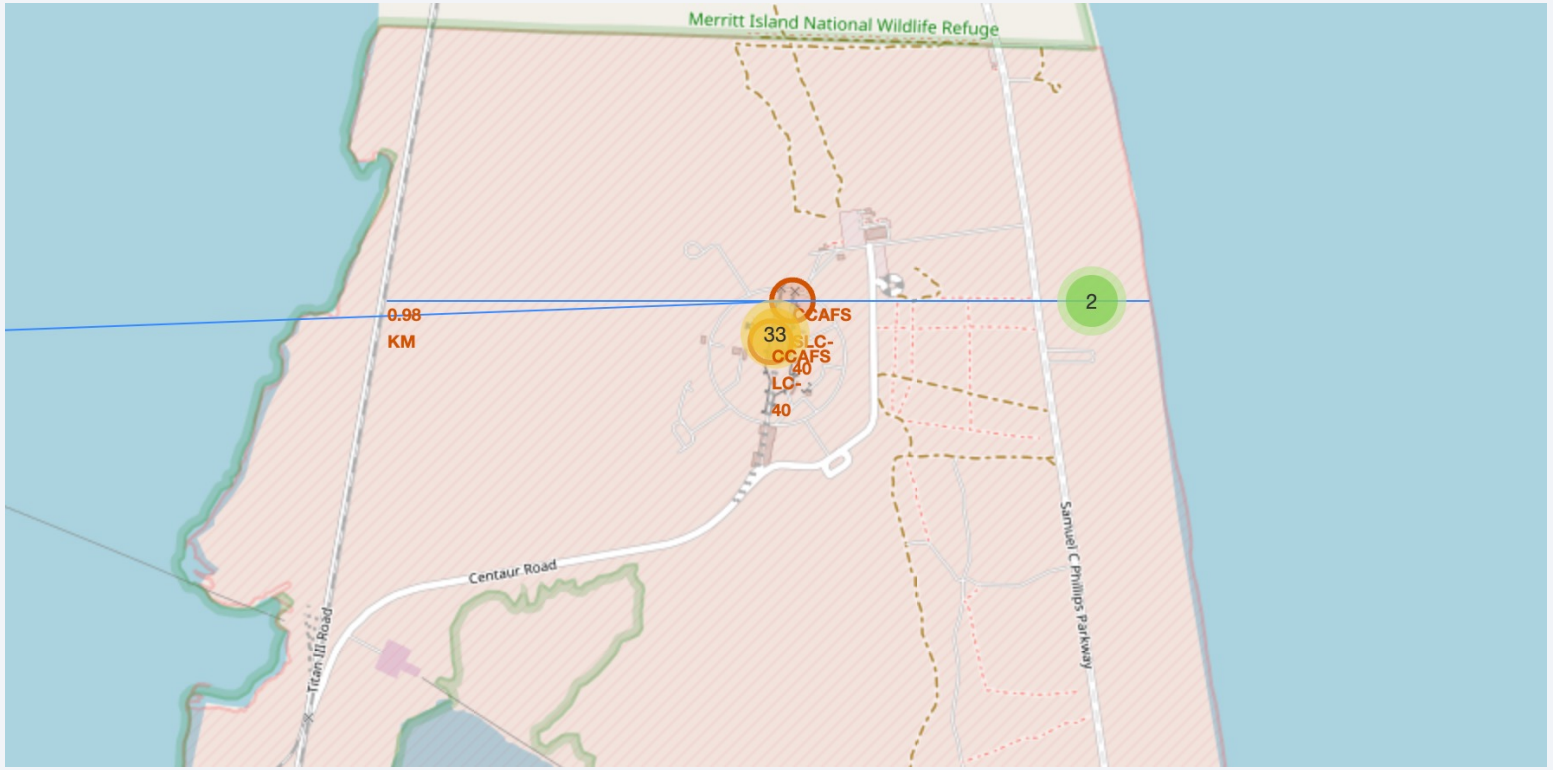
- The launch sites are all on the coast of the USA, specifically in the South East (Florida) and South West (California)

Successful and Failed Launches

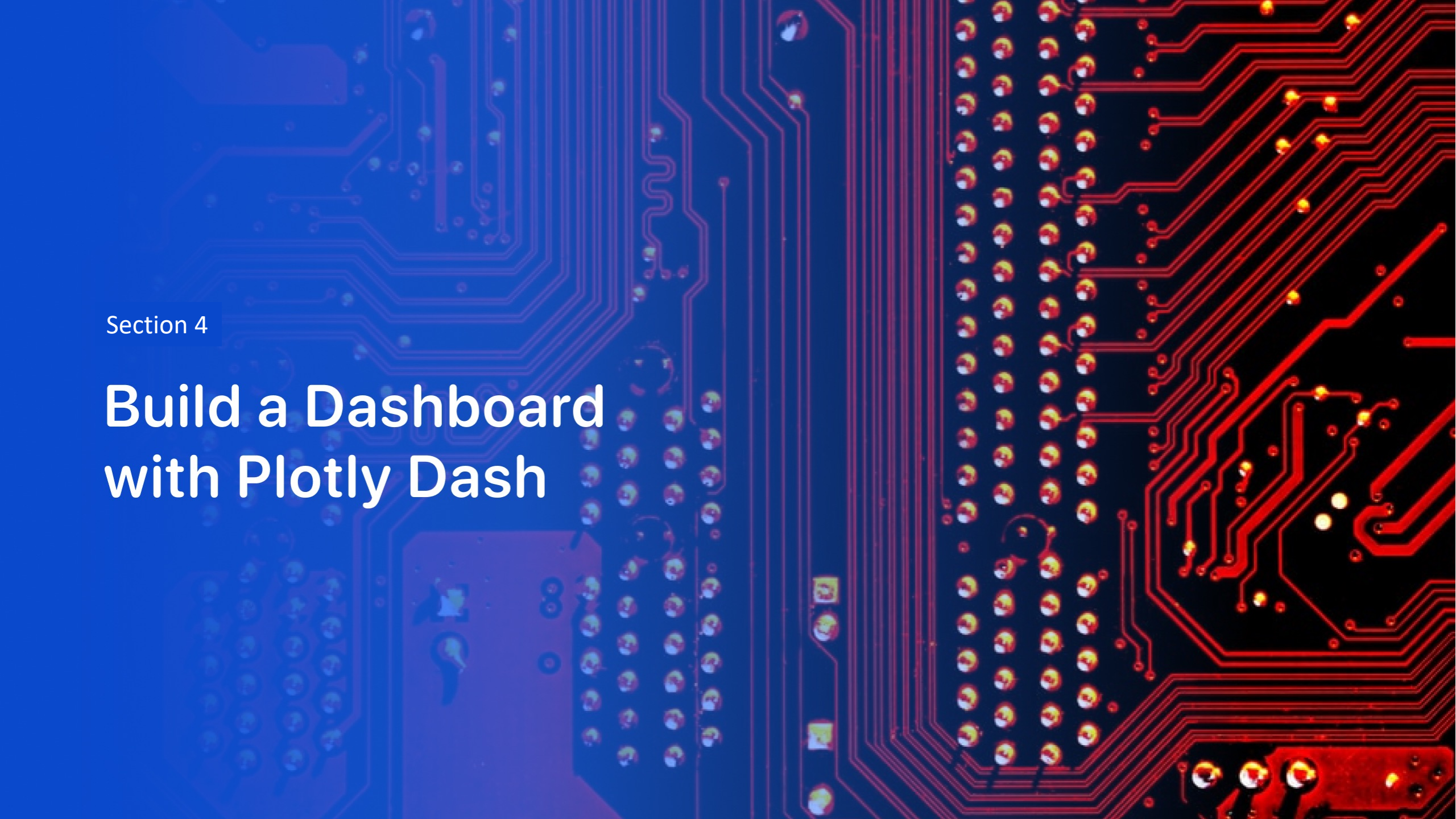


- The screenshot shows a count of successful / failed launches for a sample sight
- Successful launches are colored green and failed launches are colored red
- The circle with 28 indicates there are 28 launches collapsed within the marker

Launch Site Proximity to Points of Interest



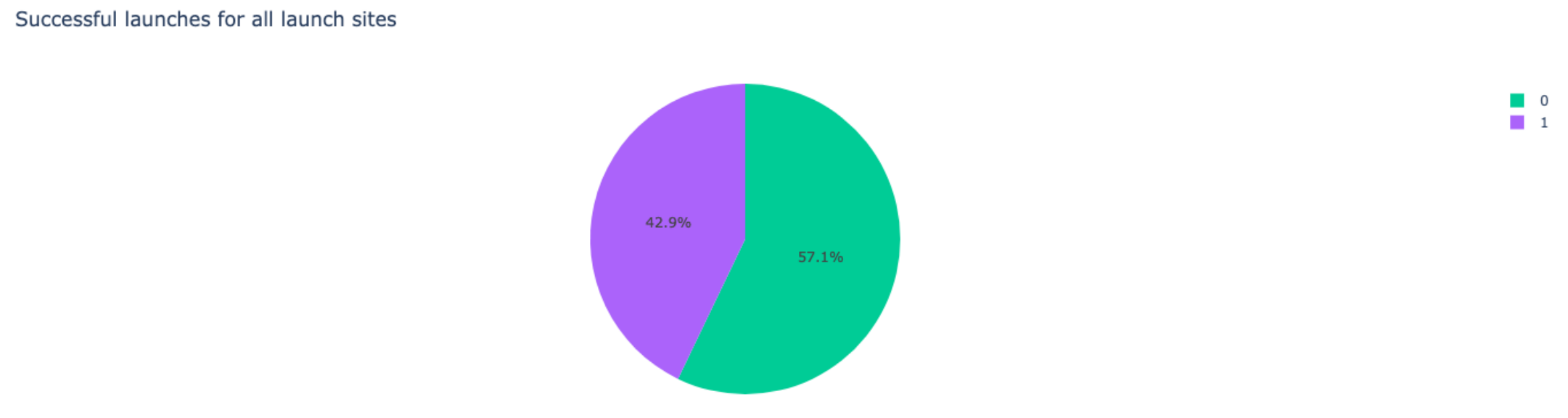
- A sample launch site can be used to show the proximity to other locations
- The coast is 0.86km to the East
- A railway is nearby at 1km away
- A road is nearby at 0.59km away
- The nearest major city of Orlando is further at almost 80km



Section 4

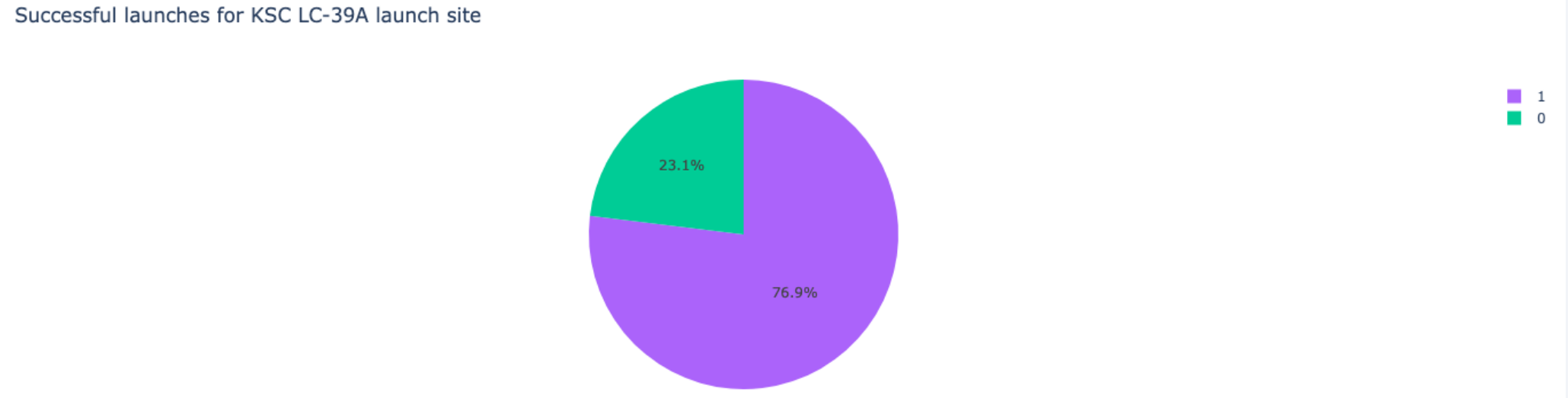
Build a Dashboard with Plotly Dash

Launch Success for All Sites



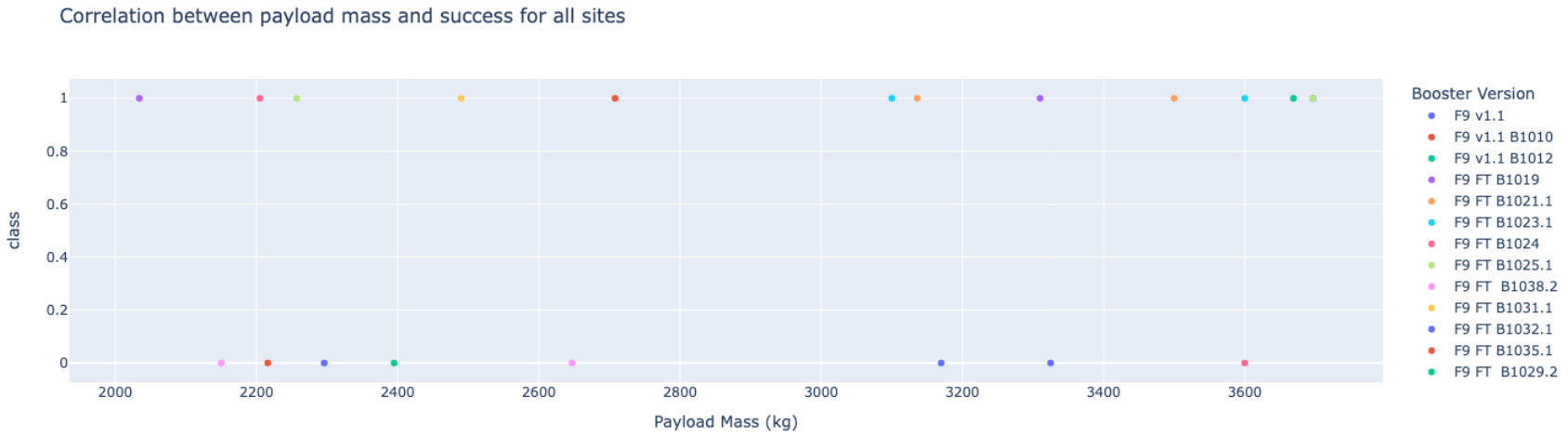
- Only 42.9% of launches have been successful
- The success rate improves with flight number so this is expected to improve with time

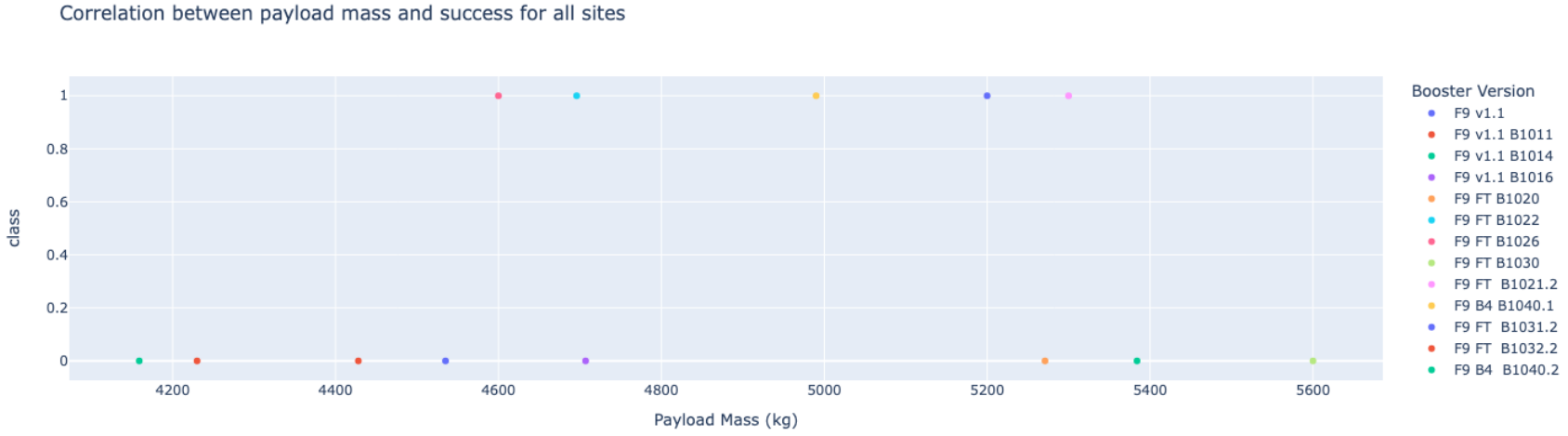
KSC-LC-39A has the highest success ratio



- The KSC-LC-39A has the highest success rate (76.9%)

Success by Payload Mass



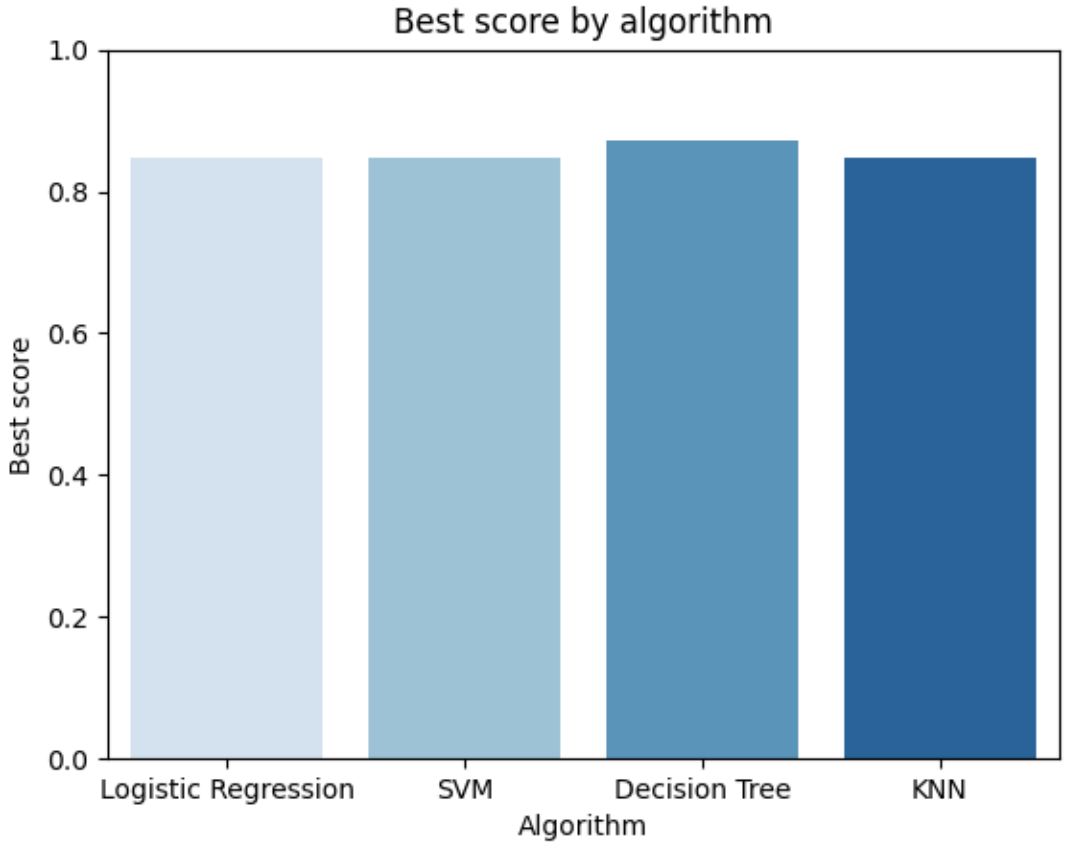


- The two charts show payloads for 2000kg - 4000kg and 4000kg – 6000kg as these are the most common ranges
- The charts show a higher success rate in the 2000kg – 4000kg range

Section 5

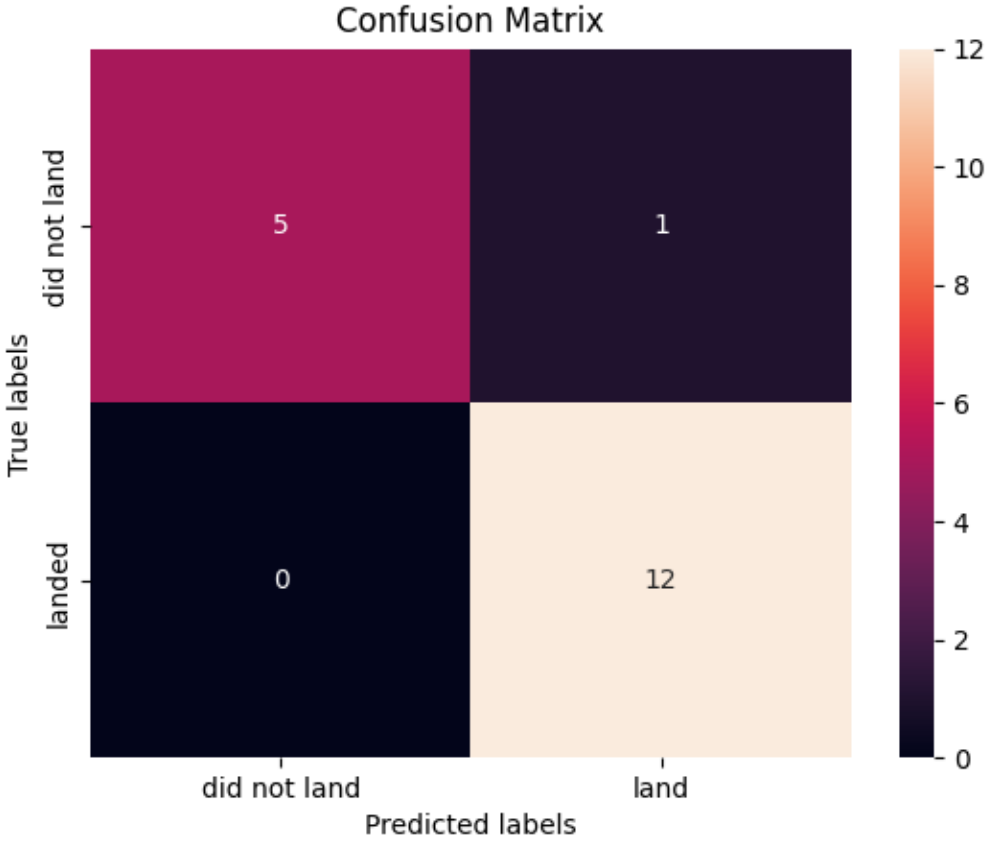
Predictive Analysis (Classification)

Classification Accuracy



- The decision tree has the highest accuracy among the algorithms tested
- The decision tree achieved an accuracy of 94% on the test set
- The other algorithms (KNN, SVM, Logistic Regression) all achieved the same accuracy of 83% on the test set

Confusion Matrix



- The confusion matrix shows the results of the decision tree classifier on the test set
- It correctly predicted 17 of the 18 values
- It incorrectly predicted that one result would land when the correct label was a failure

Conclusions

- The likelihood of success increases with the flight number – both as an overall measure and when considered by launch site. This can been seen with the increasing success rate by year
  - This is expected as later launches will have more experience and data available
- Heavier payloads have fewer data points but a higher success rate
- Launch sites are located on the coast, near to transportation infrastructure (roads and railways) but further from major cities (possibly for safety considerations)
- The decision tree classification model had the best performance with an accuracy of 94% on the test set

Appendix - Web Scraping

- https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches
- Above link was used to collect web scraping data
- Screenshots show useful functions to clean scraped data

```
def date_time(table_cells):
    """
    This function returns the data and time from the HTML  table cell
    Input: the  element of a table data cell extracts extra row
    """
    return [data_time.strip() for data_time in list(table_cells.strings)][0:2]

def booster_version(table_cells):
    """
    This function returns the booster version from the HTML  table cell
    Input: the  element of a table data cell extracts extra row
    """
    out=''.join([booster_version for i,booster_version in enumerate( table_cells.strings) if i%2==0][0:-1])
    return out

def landing_status(table_cells):
    """
    This function returns the landing status from the HTML table cell
    Input: the  element of a table data cell extracts extra row
    """
    out=[i for i in table_cells.strings][0]
    return out


def get_mass(table_cells):
    mass=unicodedata.normalize("NFKD", table_cells.text).strip()
    if mass:
        mass.find("kg")
        new_mass=mass[0:mass.find("kg")+2]
    else:
        new_mass=0
    return new_mass
```

```
def extract_column_from_header(row):
    """
    This function returns the landing status from the HTML table cell
    Input: the  element of a table data cell extracts extra row
    """
    if (row.br):
        row.br.extract()
    if row.a:
        row.a.extract()
    if row.sup:
        row.sup.extract()

    colunm_name = ' '.join(row.contents)

    # Filter the digit and empty names
    if not(colunm_name.strip().isdigit()):
        colunm_name = colunm_name.strip()
        return colunm_name
```

Appendix - Notebooks

- https://github.com/JackCrago/IBMDataScienceCertificateCapstone
- The link above shows the repository with the completed notebooks for this project

Thank you!