Winning Space Race with Data Science

Cihan Kurter
Nov 7, 2023

Outline

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

Executive Summary

- Summary of methodologies for SpaceX data
  - Data Collection via SpaceX API
  - Data Collection by using Web-scraping
  - SpaceX Data Wrangling
  - Exploratory Data Analysis (EDA) with SQL
  - EDA by utilizing Data Visualization Methods
  - Interactive Analyses by creating Dashboards
  - Visual Analyses by utilizing Folium
  - Analyses with Machine Learning Predictions
- Summary of all results
  - Exploratory data analyses results
  - Interactive and visual analytics results by using Dashboards and Folium
  - Predictive analysis results by using ML algorithms

Introduction

- Project background and context

Falcon 9 is a reusable, two-stage rocket designed and manufactured by SpaceX for the reliable and safe transport of people and payloads into Earth orbit and beyond. Falcon 9 is the world's first orbital class reusable rocket; which saves considerable money. Reusability allows SpaceX to re-fly the most expensive parts of the rocket (such as the first stage), which in turn drives down the cost of space access. If we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

- Problems you want to find answers

We would like to predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its website by using various data science methods.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collection with via SpaceX API and Web-scraping from relevant Wikipedia page.
- Perform data wrangling
  - Converting raw data into a usable form and preparing data for some Exploratory Data Analysis to find some patterns in the data
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Compare various classification models based on Machine learning algorithms

Data Collection

The data set is collected with two different methods

- By using a relevant REST API; SpaceX API

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

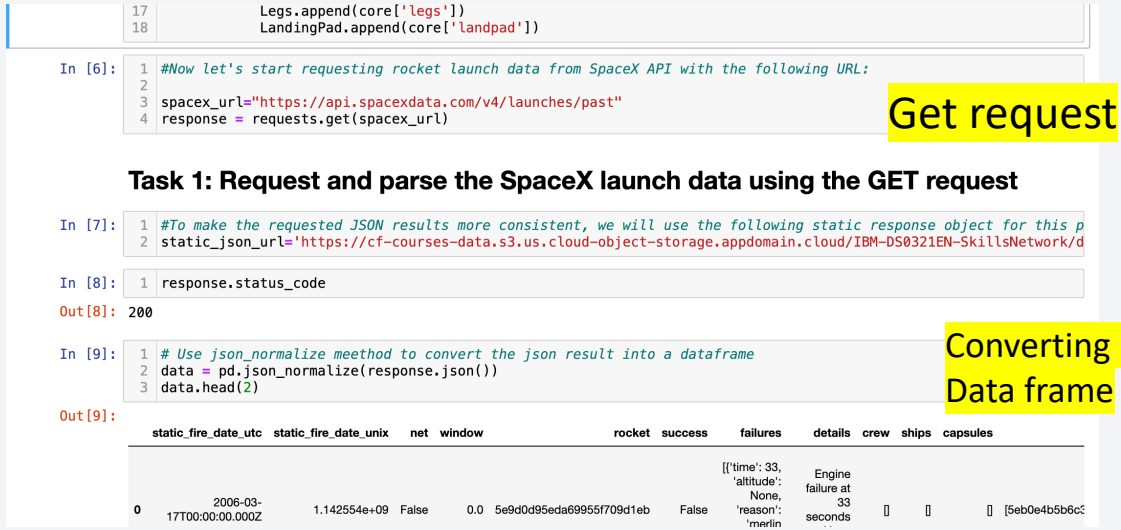
- Via Web-scraping from a relevant wiki page

https://en.wikipedia.org/wiki/List_of_Falcon_9 _and_Falcon_Heavy_launches

Data Collection – SpaceX API

- To collect data via SpaceX REST API, we performed a “get request” to the API first. The response is in the form of json. Then to be able convert the json format into a data frame we utilized the python’s pandas library. See the chart on the right.
- The GitHub URL of the completed SpaceX API calls notebook is here:

https://github.com/ckurter/Applied_Data_Science_Capstone/blob/main/Week1_SpaceX%20Data%20Collection%20with%20API.ipynb



```
17              Legs.append(core['legs'])
18              LandingPad.append(core['landpad'])
```

In [6]:
```
#Now let's start requesting rocket launch data from SpaceX API with the following URL:

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

Get request

Task 1: Request and parse the SpaceX launch data using the GET request

In [7]:
```
#To make the requested JSON results more consistent, we will use the following static response object for this p
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/d
```

In [8]:
```
response.status_code
```

Out[8]: 200

In [9]:
```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
data.head(2)
```

Converting into Data frame

Out[9]:

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	capsules	
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin	Engine failure at 33 seconds	[]	[]	[]	[5eb0e4b5b6c3

Cleaning and modifying the data
Saving it into a csv file

Data Collection - Scraping

- Here we used python's BeautifulSoup package to webscrape some html tables from SpaceX wiki page. Then we parsed the data and converted into a dataframe. See the parts of the script on the right.
- the GitHub URL of the completed web scraping notebook is here:

https://github.com/ckurter/Applied_Data_Science_Capstone/blob/main/Week1_SpaceX%20Data%20Collection%20with%20Web%20Scraping.ipynb

```
In [3]:
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=102768
```

TASK 1: Request the Falcon9 Launch Wiki page from its URL

```
In [4]:
#Next, request the HTML page from the above URL and get a response object

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

Get request

```
In [5]:
#Create a BeautifulSoup object from the HTML response
soup = BeautifulSoup(response, 'html5lib')
```

Using BeautifulSoup Package

```
In [6]:
#Print the page title to verify if the BeautifulSoup object was created properly

#print("the page title:", soup.title.string)
print("the page title:", soup.title)
```

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

Create a data frame by parsing the launch HTML tables
Export to csv file

Data Wrangling

We wanted to do an initial EDA and find some patterns in the data . For that reason, we first converted the raw data into useful (labeled) form through data wrangling.

- calculated the number of launches on each site
- calculated the number and occurrence of each orbit
- calculated the number and occurence of mission outcome of the orbits
- finally converted landing outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

- the GitHub URL of the completed data wrangling related notebook:

https://github.com/ckurter/Applied_Data_Science_Capstone/blob/main/Week1_SpaceX%20Data%20Wrangling.ipynb

EDA with Data Visualization

- Data Visualization is a great way to do EDA and see the initial patterns in the data for further applications of ML models
  - visualized relationship between Flight Number and Launch Site through categorical scatter plots
  - visualized relationship between Payload and Launch Site through categorical scatter plots
  - visualized the relationship between success rate of each orbit type through bar chart
  - visualized the relationship between Flight Number and Orbit type with scatter plots
  - visualized the relationship between Payload and Orbit type through a scatter plot
  - visualized the launch success yearly trend with a line plot
- the GitHub URL of the completed EDA with data visualization notebook:

https://github.com/ckurter/Applied_Data_Science_Capstone/blob/main/Week2_SpaceX%20EDA%20with%20Data%20Visualization.ipynb

EDA with SQL

- Following tasks are completed with SQL queries:
  - Display the names of the unique launch sites in the space mission
  - Display 5 records where launch sites begin with the string 'CCA’
  - Display the total payload mass carried by boosters launched by NASA (CRS)
  - Display average payload mass carried by booster version F9 v1.1
  - List the date when the first succesful landing outcome in ground pad was acheived.
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - List the total number of successful and failure mission outcomes
  - List the names of the booster_versions which have carried the maximum payload mass by using a subquery
  - List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
  - Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

- the GitHub URL of your completed EDA with SQL notebook:

https://github.com/ckurter/Applied_Data_Science_Capstone/blob/main/Week2_SpaceX%20EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- Interactive Maps with Folium is helpful to find an optimal location for building a launch site
  - We marked all launch sites on a map by using folium.Circle and folium.Marker
  - We marked the successful and failed launches for each site on the map by utilizing marker_cluster
  - We marked down a point on the closest coastline using MousePosition and calculate the distance between the coastline point and the launch site
- We utilized above objects since we wanted to mark the location of the launch sites and see if they are close to the equator line, coasts, railway, highways, or cities
- the GitHub URL of your completed interactive map with Folium map:

https://github.com/ckurter/Applied_Data_Science_Capstone/blob/main/Week3_SpaceX%20Launch%20Sites%20Locations%20Analysis%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

To be able to display correlations between payload and launch success interactively we created a dashboard

- added a dropdown list with an id of “site-dropdown” to enable Launch Site selection
- added a “pie chart” showing the total successful launches count for all sites
- created a callback function for “site-dropdown” as input, “pie-chart” as output
- added a slider to select payload range with an id of “payload-slider”
- added a scatter chart to show the correlation between payload and launch success
- created a callback function for “site-dropdown” and “payload-slider”as inputs; scatter chart as output
- the GitHub URL of your completed Plotly Dash lab:

https://github.com/ckurter/Applied_Data_Science_Capstone/blob/main/Week3_SpaceX%20Dashboard%20with%20Plotly.ipynb

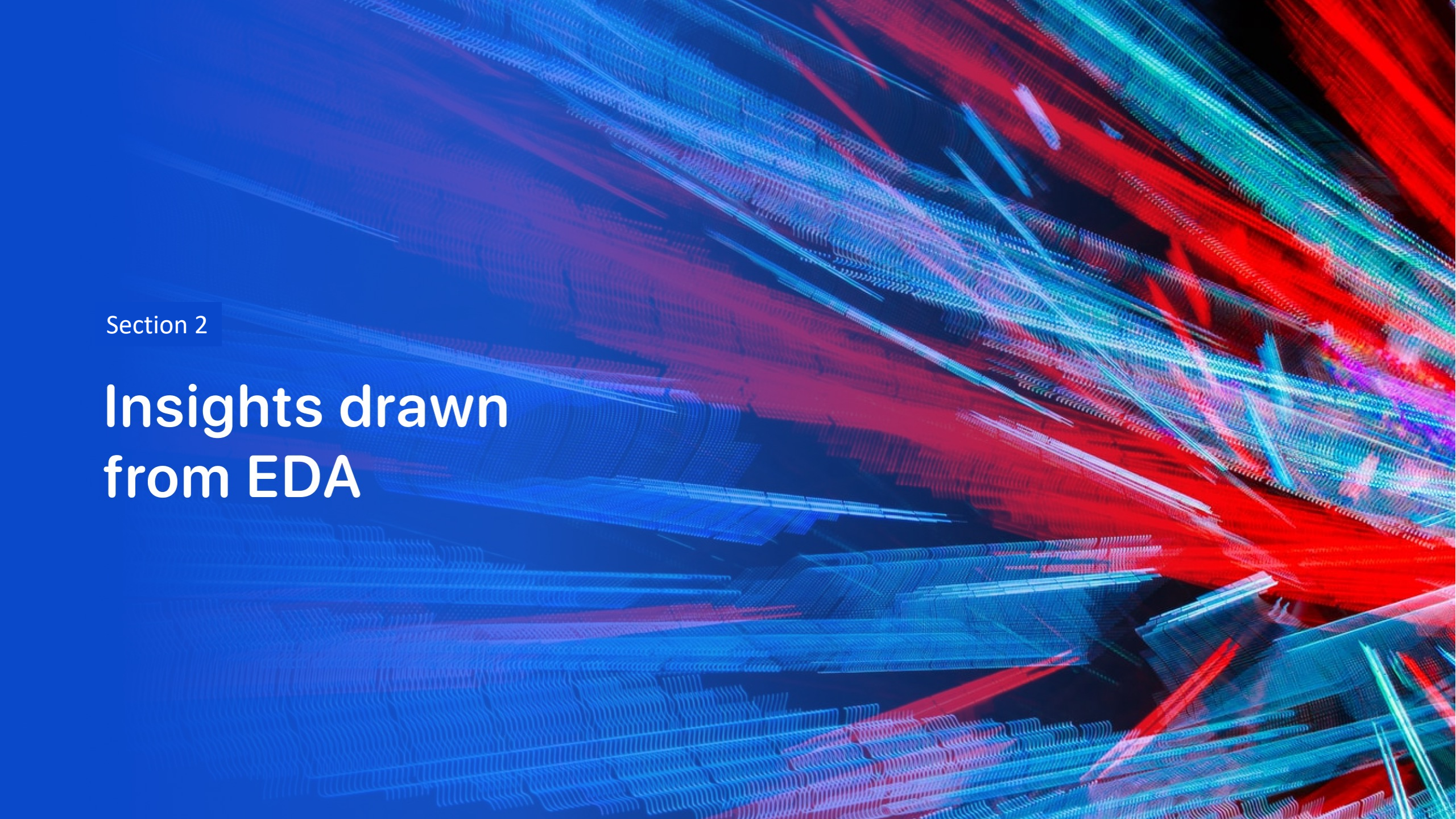
Predictive Analysis (Classification)

- Data wrangling: Created a column for the “class” for data ; Standardized the data X
- Split data set into training and test data
- Applied ML algorithms to fit the training data such as Logistic regression, SVM, Decision Tree and KNN
- Calculated the accuracy of the test data for each model on the same data
- Selected the right model depending on the accuracy score
- the GitHub URL of your completed predictive analysis lab:

https://github.com/ckurter/Applied_Data_Science_Capstone/blob/main/Week4_SpaceX%20ML%20prediction.ipynb

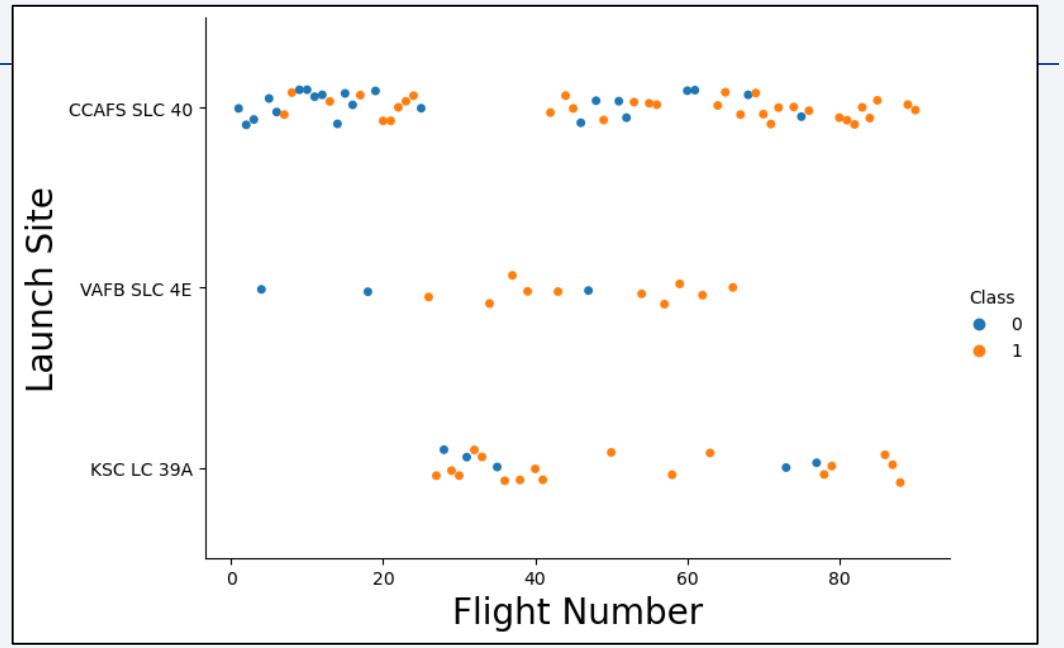
Results

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


Section 2
Insights drawn from EDA

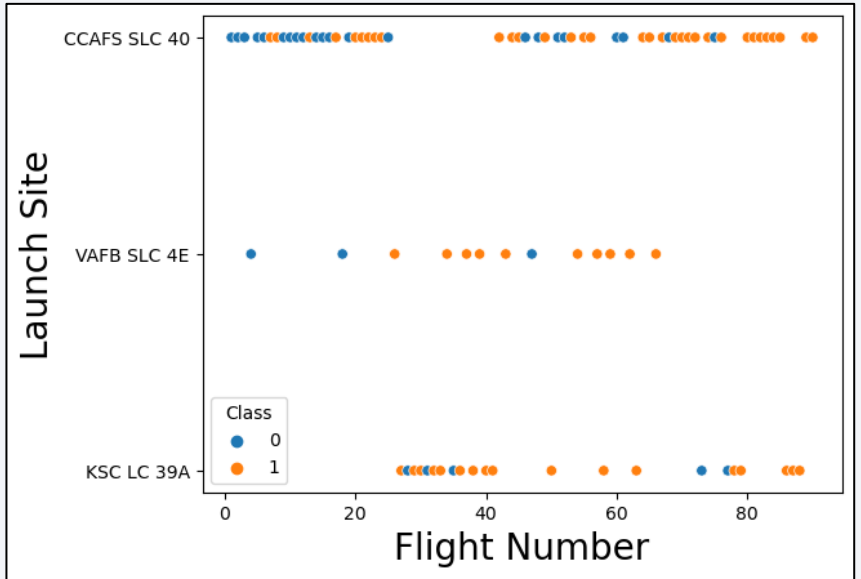
Flight Number vs. Launch Site

- a scatter plot of Flight Number vs. Launch Site



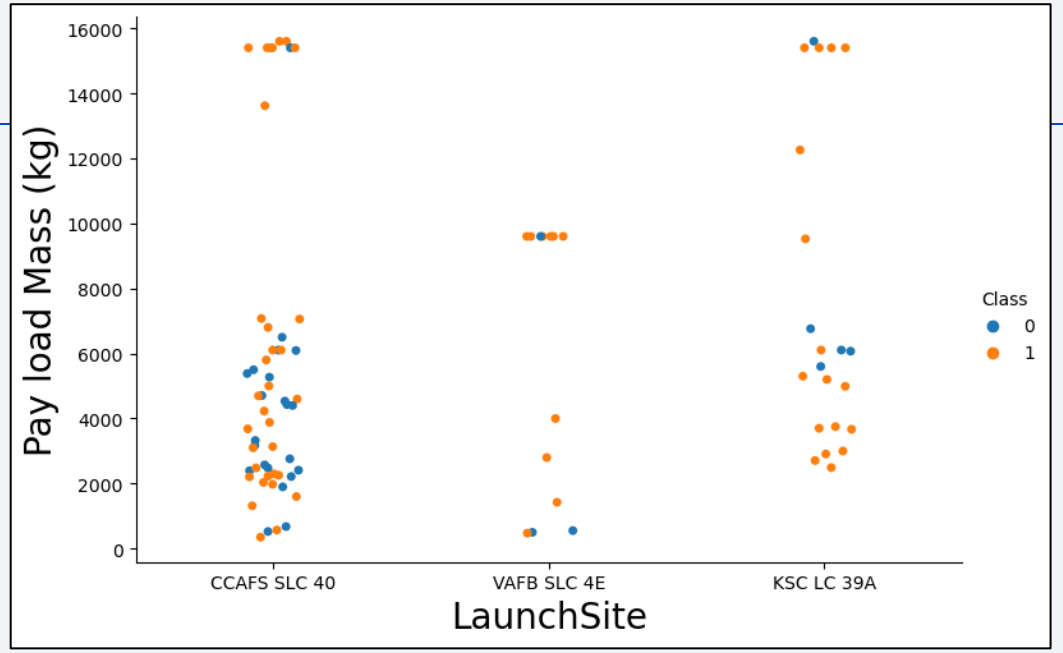
- the scatter plot with explanations:

As the flight number increases in each of the 3 launch sites, so does the success rate. For instance, the success rate for the VAFB SLC 4E launch site is 100% after the Flight number 50. Both KSC LC 39A and CCAFS SLC 40 have a 100% success rates after 80th flight.



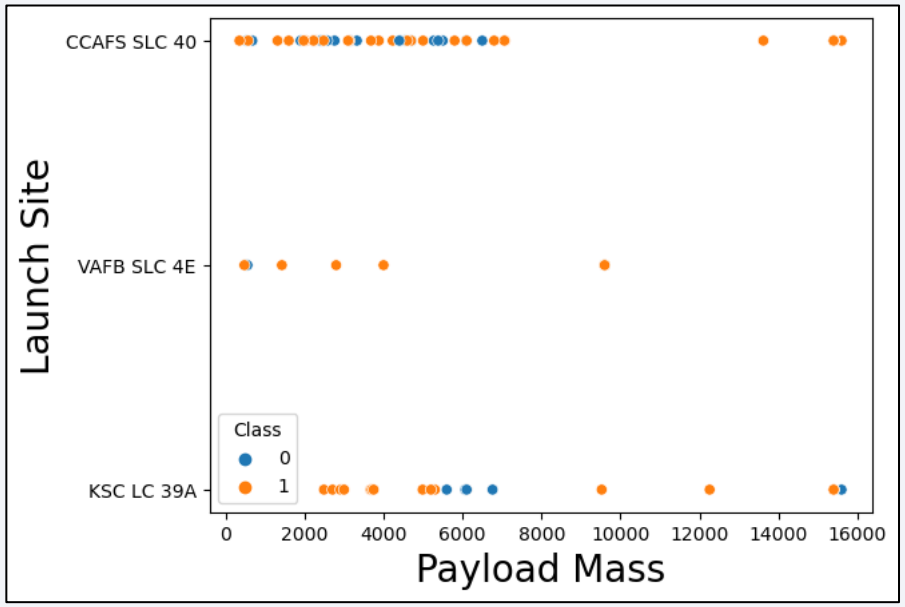
Payload vs. Launch Site

- a scatter plot of Payload vs. Launch Site



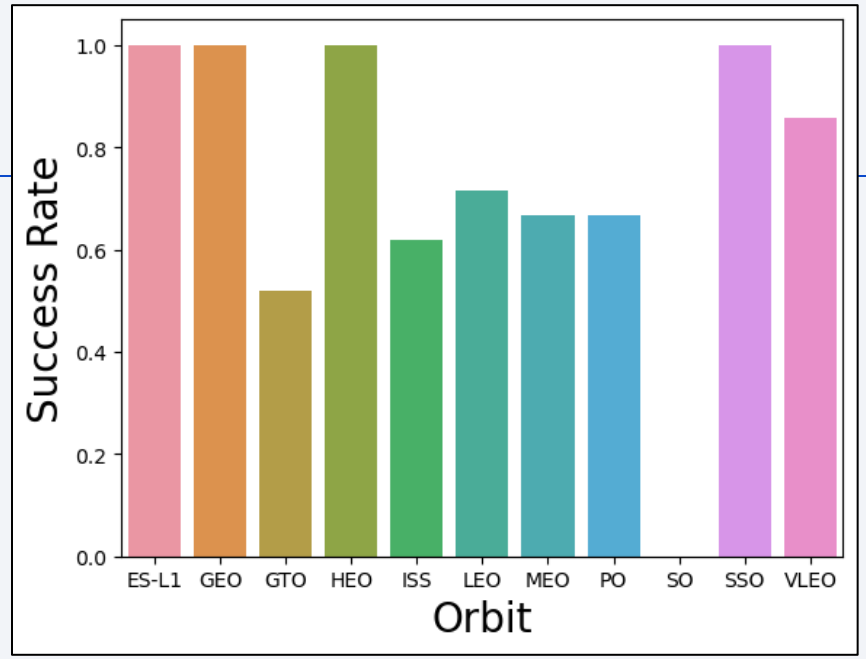
- the screenshot of the scatter plot with explanations:

For the VAFB-SLC launch site there are no rockets launched for heavy payload mass (greater than 10000).



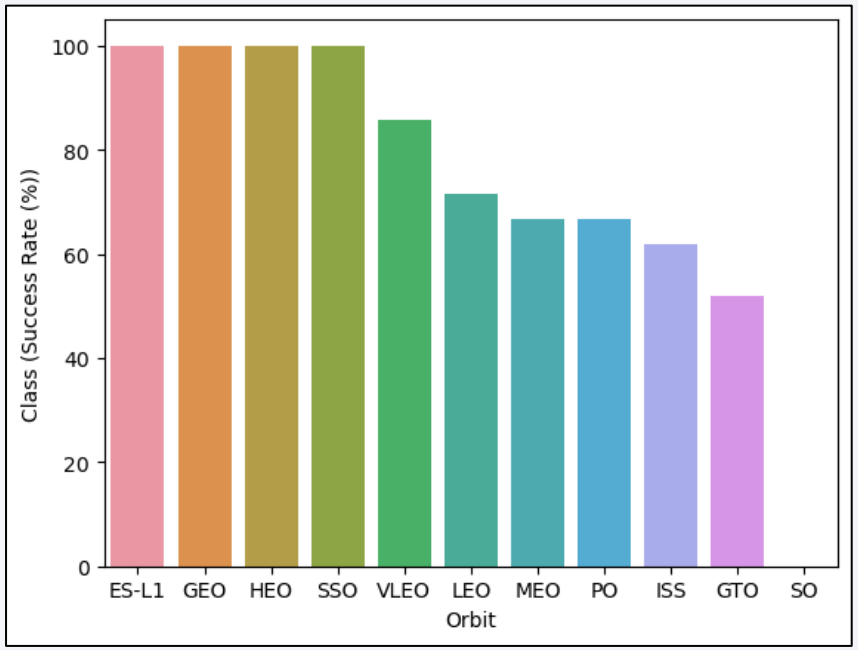
Success Rate vs. Orbit Type

- a bar chart for the success rate of each orbit type



- the screenshot of the scatter plot with explanations:

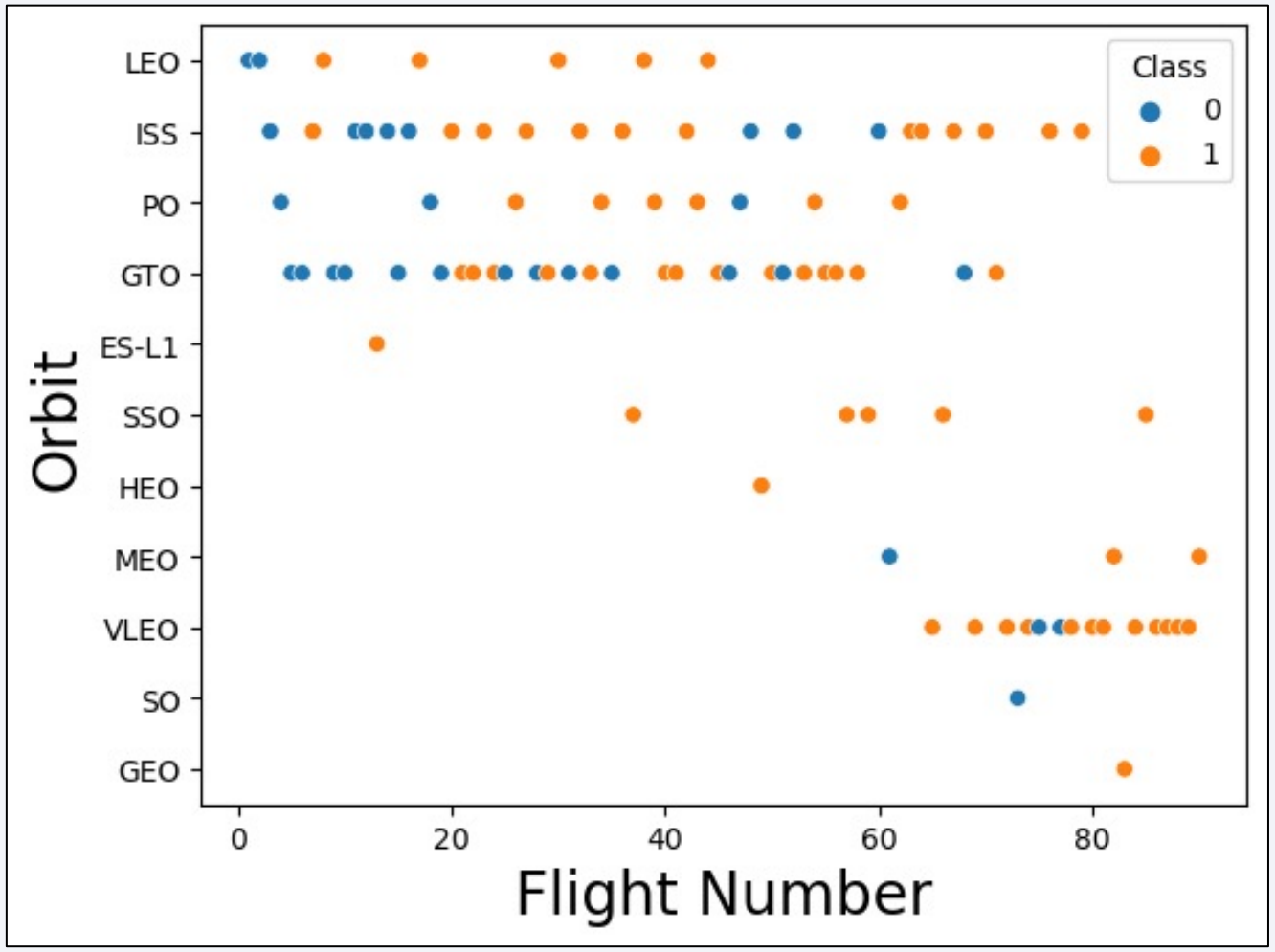
Orbits ES-L1, GEO, HEO & SSO have the highest success rates at 100%, with GTO orbit having the lowest success rate at ~50%. Orbit SO has 0% success rate.



Flight Number vs. Orbit Type

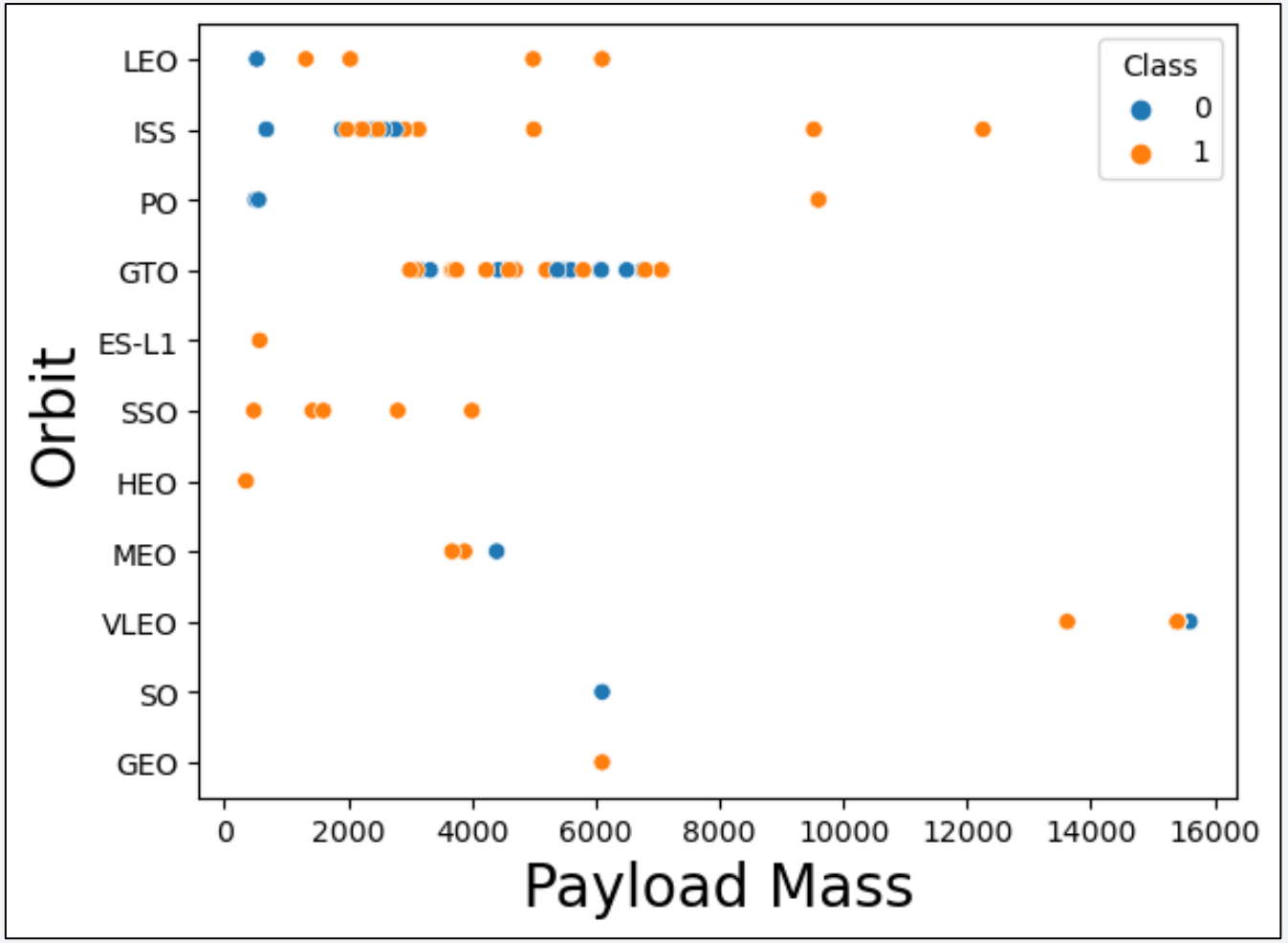
- the screenshot of the scatter plot with explanations

The success appears to be related to the number of flights in the LEO or ISS orbits; on the other hand, there seems to be no relationship between flight number and success of the outcome in GTO orbit.



Payload vs. Orbit Type

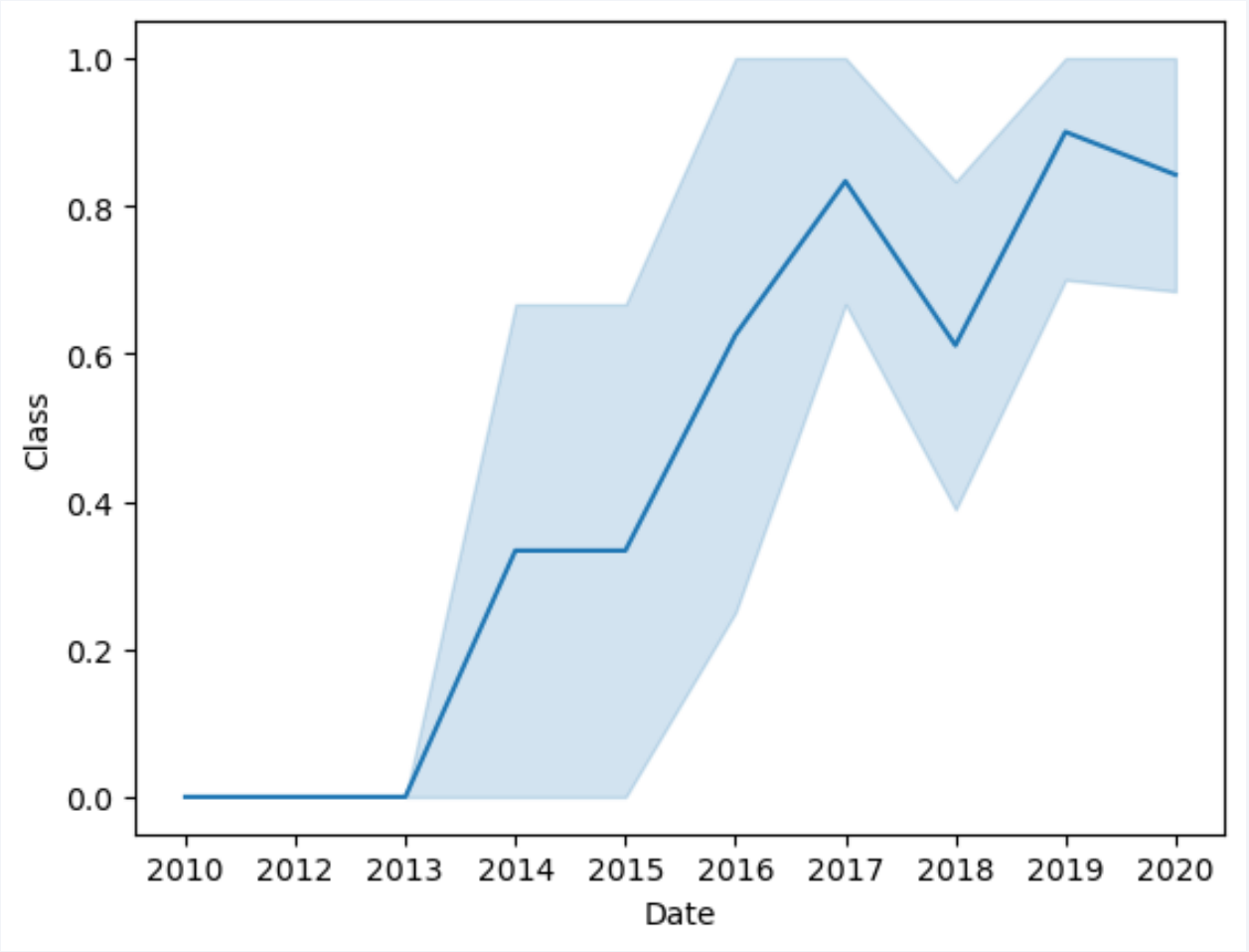
- the screenshot of the scatter plot with explanations
- The successful landing or positive landing rate are more frequent with heavy payloads for Polar, LEO and ISS.
- However, for orbit GTO we cannot distinguish this well as both successful and unsuccessful outcomes are available in the same payload mass regime.



Launch Success Yearly Trend

- the screenshot of the scatter plot with explanations

The success rate kept increasing since 2013 till 2020



All Launch Site Names

- Find the names of the unique launch sites
- Present your query result with a short explanation here

In [117]:
```
1 %sql select distinct Launch_Site from SPACEXTBL;
```

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

Out[117]:

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

"DISTINCT" statement allows us to select unique sites from Launch_site column

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- Present your query result with a short explanation here

Task 2:

Display 5 records where launch sites begin with the string 'CCA'

In [132]:
```
%sql select * from SPACEXTBL where Launch_Site like 'CCA%' limit 5;
```

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

Out[132]:

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

We use “where ” clause and “like” statement to get launch sites begin with `CCA`, and limit 5 gets “5 records”

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- Present your query result with a short explanation here

Task 3:

Display the total payload mass carried by boosters launched by NASA (CRS)

In [133]:

```
%sql select Customer, SUM(PAYLOAD_MASS__KG_) AS "total payload mass" from SPACEXTBL where \
Customer = "NASA (CRS)" \
group by Customer
```

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

Out[133]:

Customer	total payload mass
NASA (CRS)	45596

Total payload is calculated via SUM() function and written into a new column called “total payload mass”

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here

Task 4:

Display average payload mass carried by booster version F9 v1.1

In [135]:
```
%sql select Booster_Version, avg(PAYLOAD_MASS__KG_) as "average payload mass" from SPACEXTBL \
where Booster_Version = "F9 v1.1" group by Booster_Version;
```

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

Out[135]:

Booster_Version	average payload mass
F9 v1.1	2928.4

The average payload is calculated by AVG() function and written into a new column called “ average payload mass”

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Present your query result with a short explanation here

Task 5:

List the date when the first succesful landing outcome in ground pad was acheived.

In [138]:
```
%sql select Landing_Outcome, min(Date) from SPACEXTBL where Landing_Outcome = "Success" ;
```

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

Out[138]:

Landing_Outcome	min(Date)
Success	2018-03-12

The function min(Date) allows the get the earliest time/date for the successful landing outcome

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- Present your query result with a short explanation here

Task 6:

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [13]:
```
%sql select Booster_Version, Landing_Outcome, PAYLOAD_MASS__KG_ from SPACEXTBL \
where Landing_Outcome= "Success (drone ship)" and PAYLOAD_MASS__KG_ >4000 and \
PAYLOAD_MASS__KG_ < 6000;
```

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

Out[13]:

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

The payload mass is specified by conditions PAYLOAD_MASS__KG_ >4000 and PAYLOAD_MASS__KG_ < 6000;

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here

In [105]:
```
%sql select Mission_Outcome, count(Mission_Outcome) as "total number" from SPACEXTBL group by Mission_Outcome
```

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

Out[105]:

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

The total number of successful and failure mission outcomes is obtained by COUNT() function

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Present your query result with a short explanation here

Task 8:

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

In [15]:
```
%sql select Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTBL where \
PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

Out[15]:

Booster_Version	PAYLOAD_MASS__KG_
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	15600

The max() function is used to get the list of boosters which have carried the maximum payload mass

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- Present your query result with a short explanation here

Task 9:

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

In [16]:
```
%sql SELECT substr(Date, 6,2) as month, Date, Booster_Version,Launch_Site, Landing_Outcome FROM SPACEXTBL \
where Landing_Outcome like "Fail%" and substr(Date,0,5)='2015'
```

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

Out[16]:

month	Date	Booster_Version	Launch_Site	Landing_Outcome
10	2015-10-01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

To get the months/years from the Date column, we used substr(Date, 6,2) and substr(Date,0,5)

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

- 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
- Present your query result with a short explanation here

In [17]:
```
%sql select Date, 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.
```

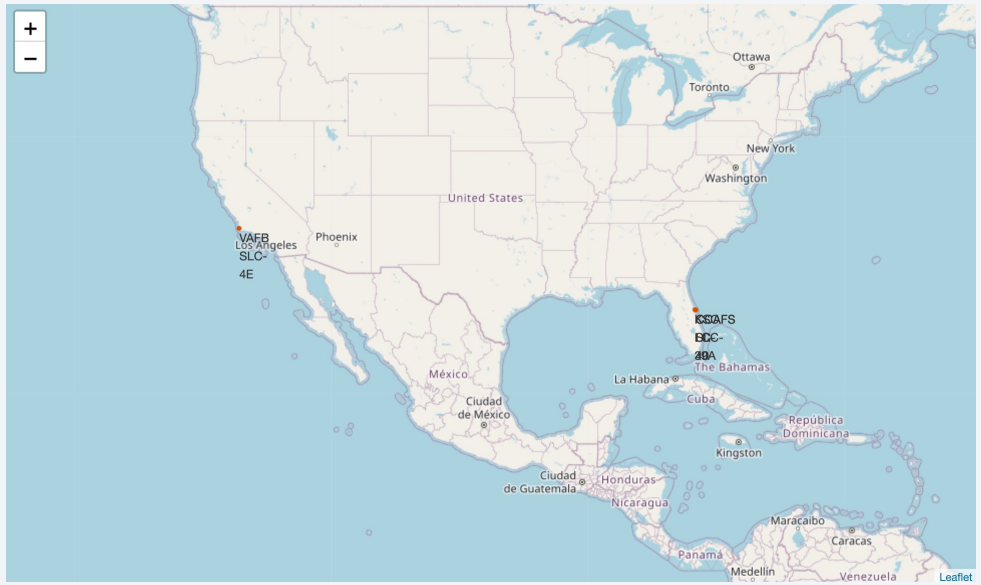
Out[17]:

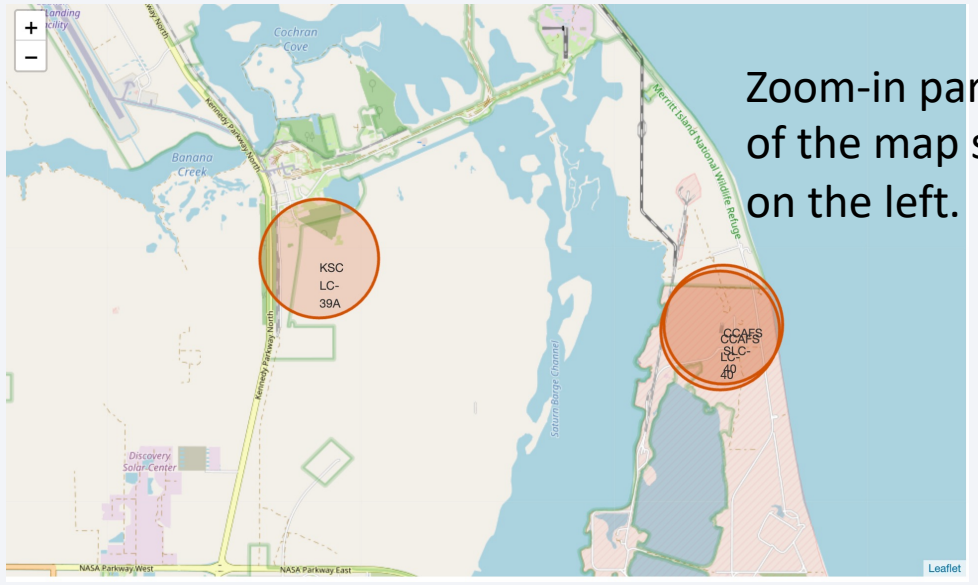
Date	Landing_Outcome	count_outcomes
2012-05-22	No attempt	10
2015-12-22	Success (ground pad)	5
2016-08-04	Success (drone ship)	5
2015-10-01	Failure (drone ship)	5
2014-04-18	Controlled (ocean)	3
2013-09-29	Uncontrolled (ocean)	2
2015-06-28	Precluded (drone ship)	1
2010-08-12	Failure (parachute)	1

We used a condition here for Date column as following:
Date between "2010-06-04" and "2017-03-20"


Section 3
Launch Sites
Proximities Analysis

Launch Site Location Analysis with Folium





Zoom-in part of the map shown on the left.

From the previous analyses, we found out that there are 4 distinctive launch sites as in the following table.

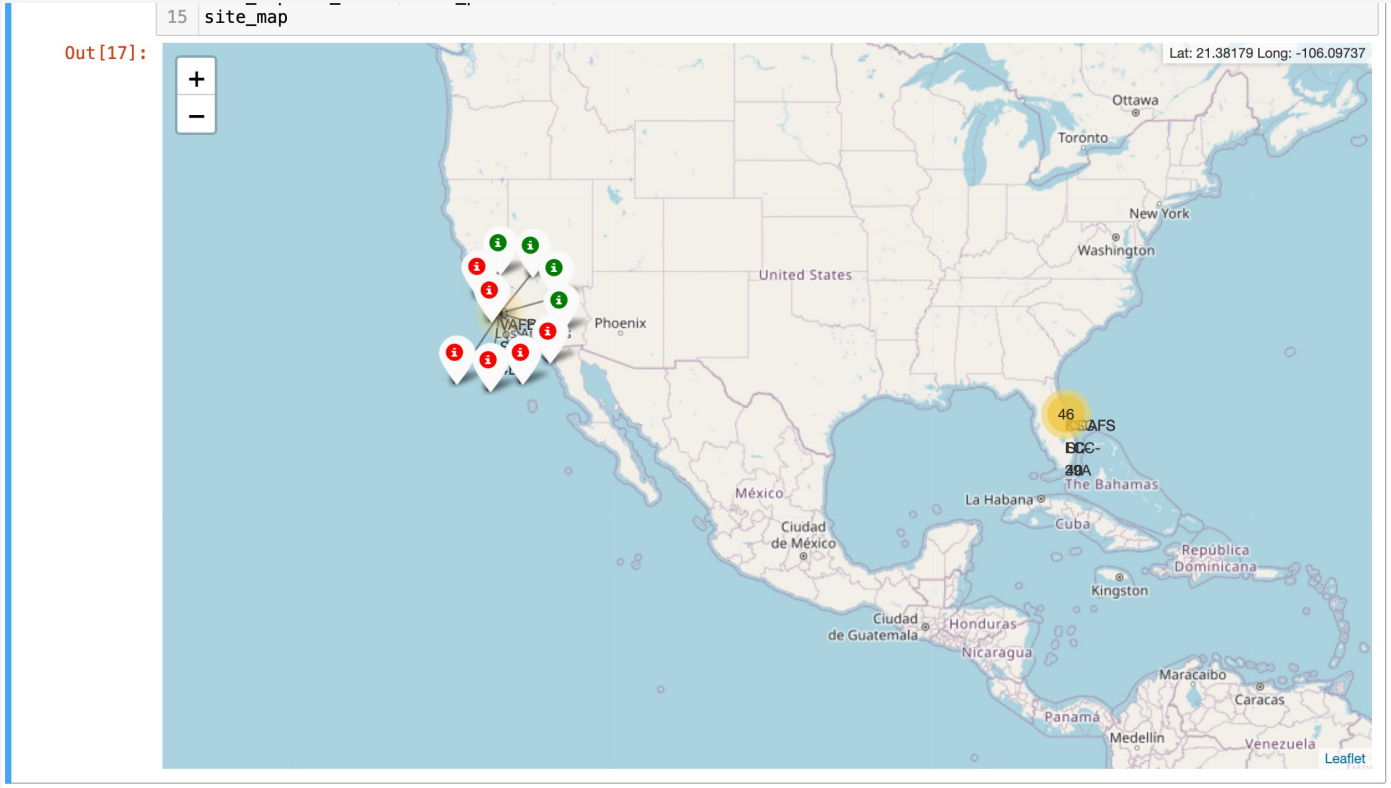
[17]:

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

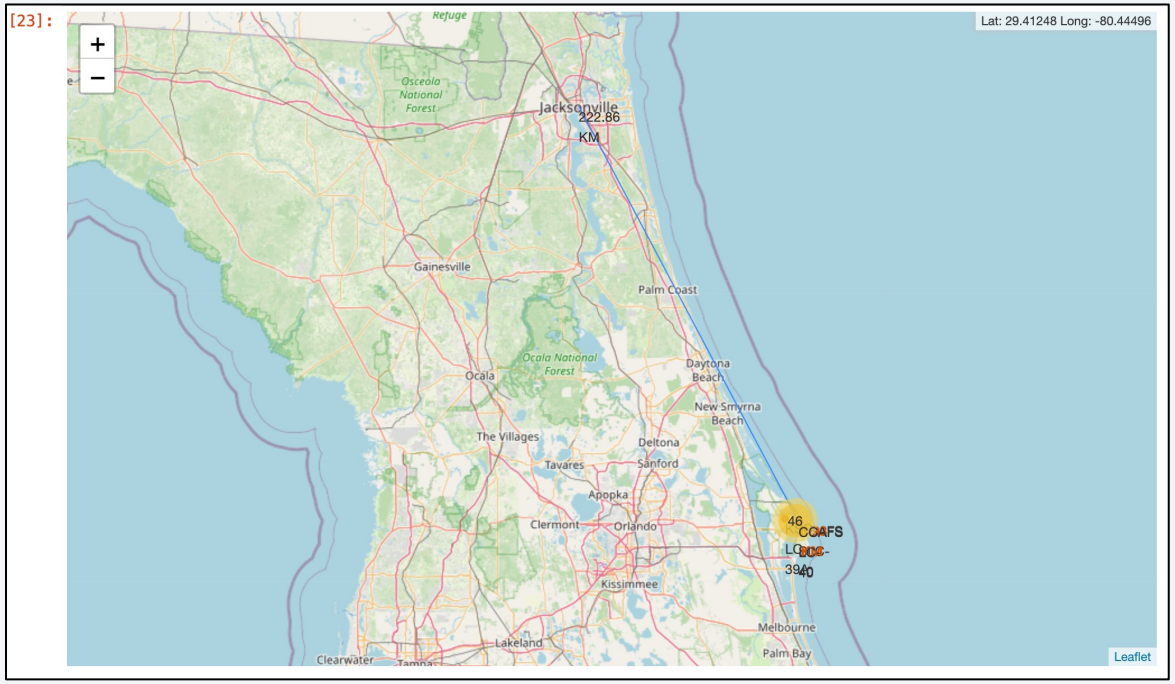
With Folium, we marked the locations of these launch sites, VAFB is on the west coast, but the other 3 sites are on the south-east coast and quite close to each other. To be able to see the markers, we need to zoom in, the zoom in map is shown on the right.

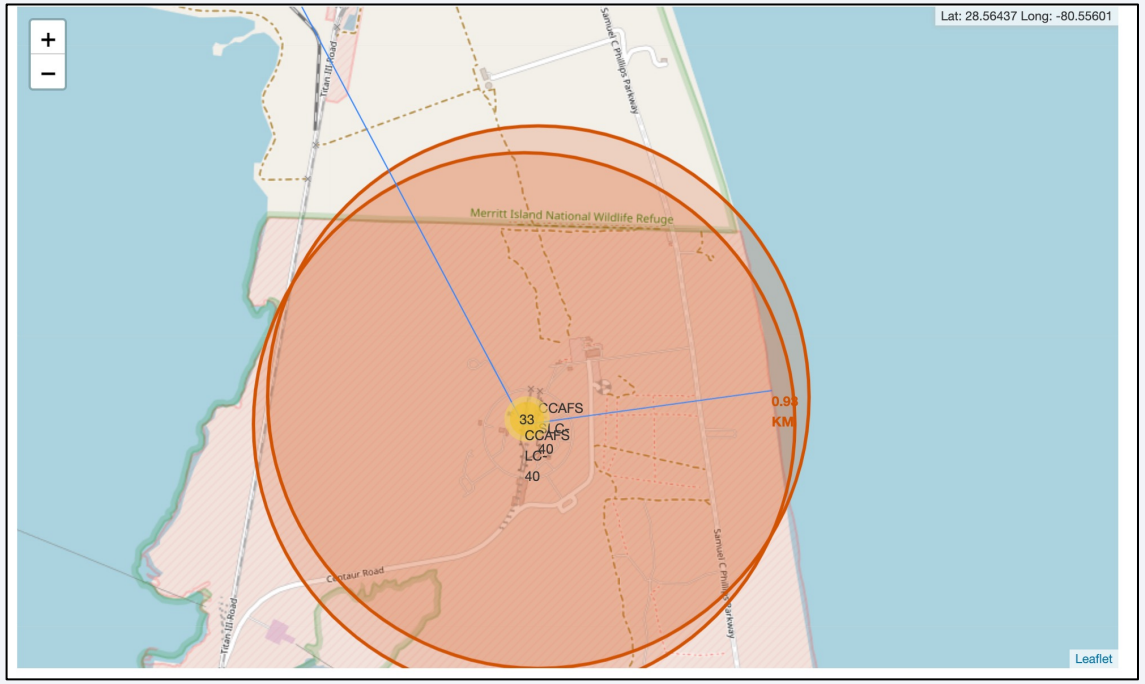
Launch outcomes of different sites

We marked the success/failed launches for each site on the map. We assigned a red marker if a launch was failed, and assigned a green marker if the launch was successful. The screenshot shows the marked outcomes at the VAFB SLC 4E launch site.



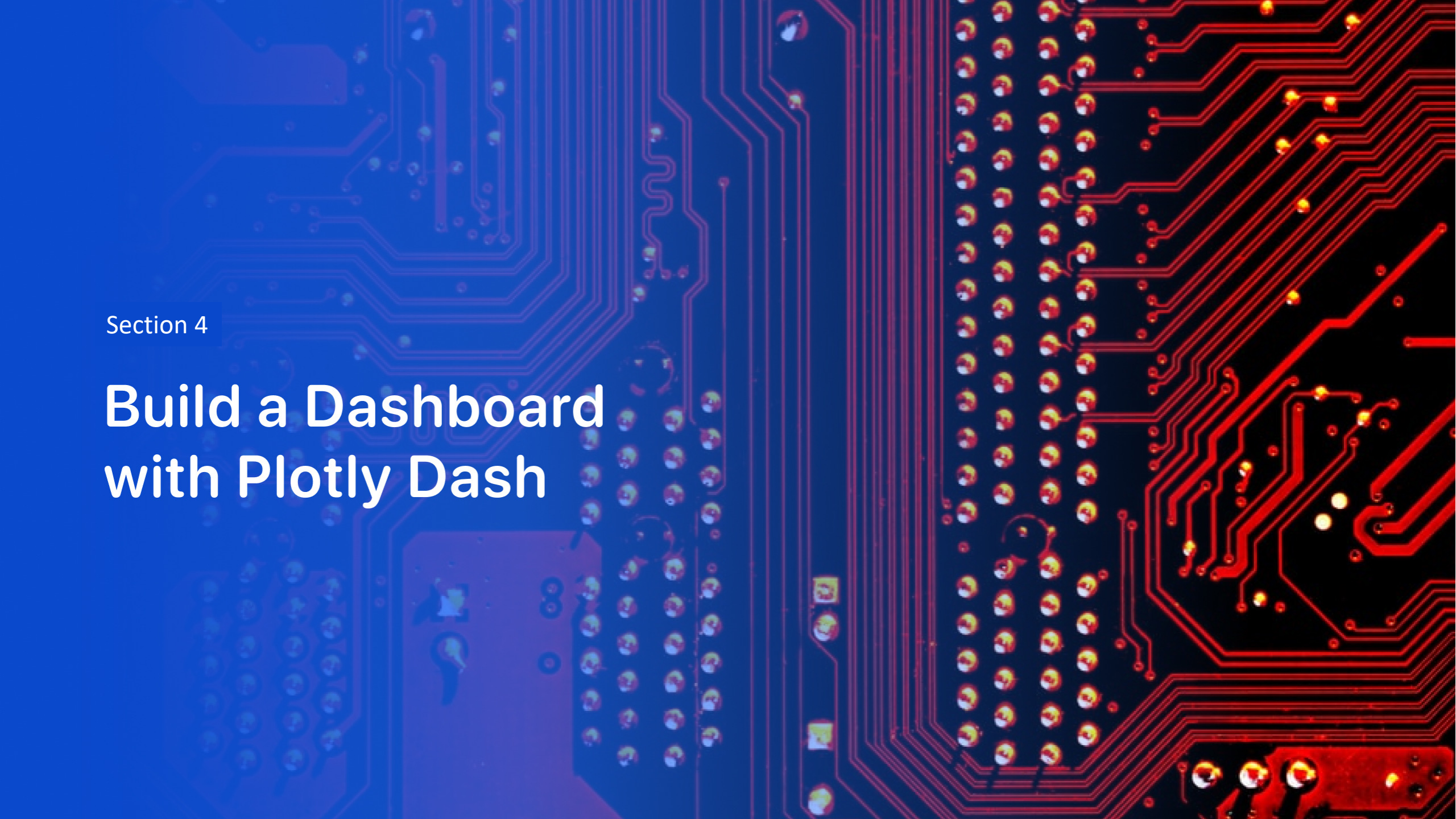
Calculating the distances between a launch site to its proximities



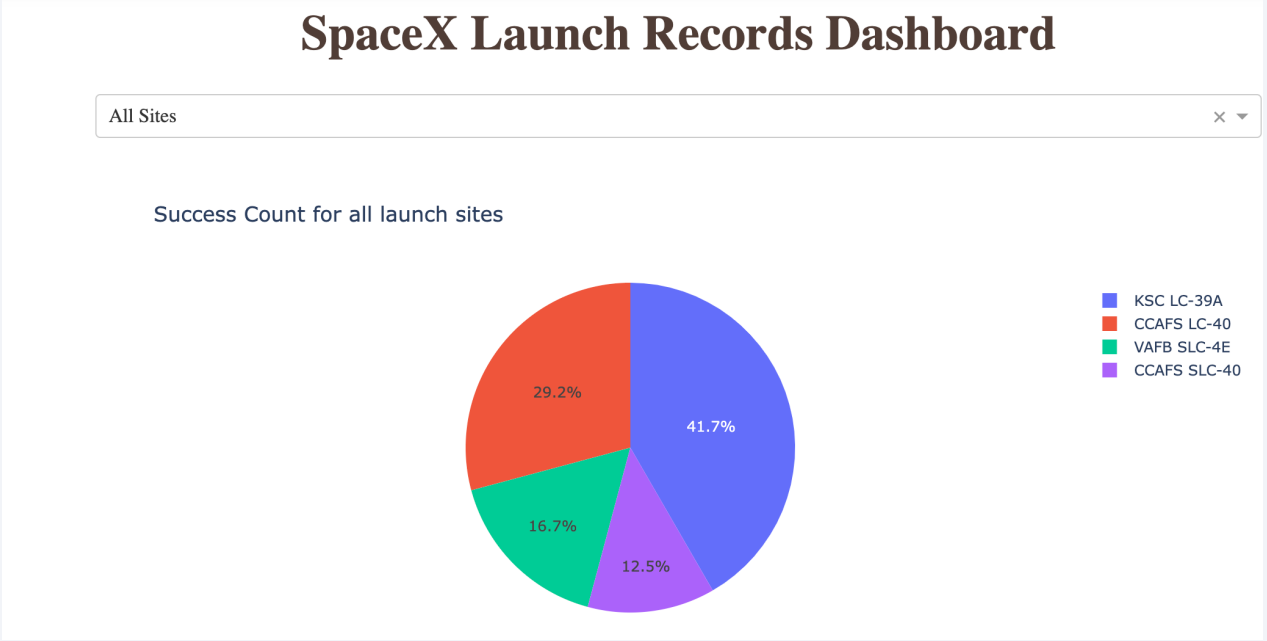


We also explored and analyzed the proximities of launch sites. We calculated and plotted distance lines to the proximities of the launch sites such as railways, highways, coastlines and even nearby cities.

The screen shot shows the distance from CCAFS LC-40 launch site to Jacksonville as 222.86 km and to the coastline as 0.93 km

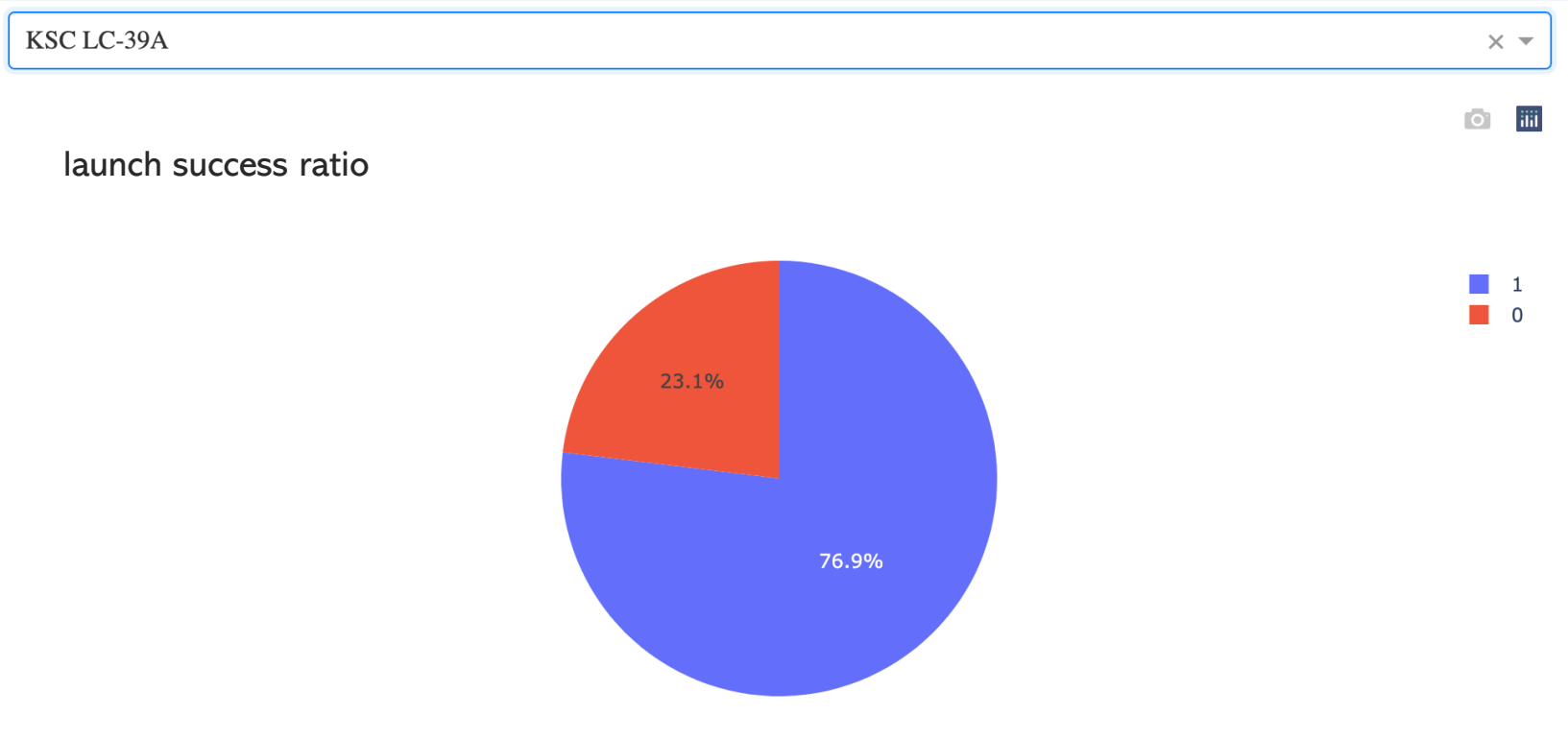

Section 4
Build a Dashboard with Plotly Dash

Success Count for All Launch Sites



- Dashboard demonstrates the pie chart showing the success rate for all the launch sites
- Launch site KSC LC-39A has the highest launch success rate at 42% followed by CCAFS LC-40 at 29%, VAFB SLC-4E at 17% and CCAFS SLC-40 at 13%

Highest launch success ratio



The highest success rate is achieved at the site KSC LC-39 A, the success rate is 77%, only 23 % of the attempts were failed.

Payload vs. Launch Outcome scatter plot for all sites

The scatter plot demonstrates the correlation between outcome and payload mass for all sites. Particularly booster version FT seems to have a large success rate for payload mass >2000 kg

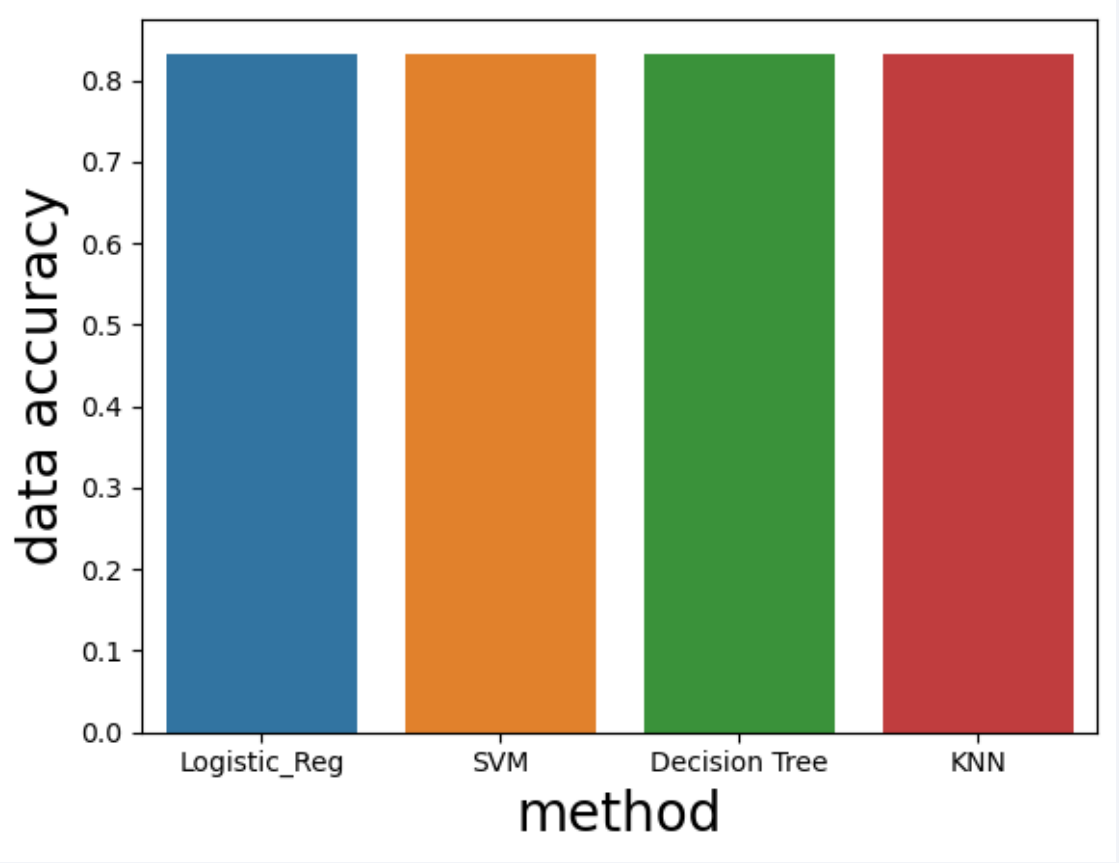
Section 5

Predictive Analysis (Classification)

Classification Accuracy

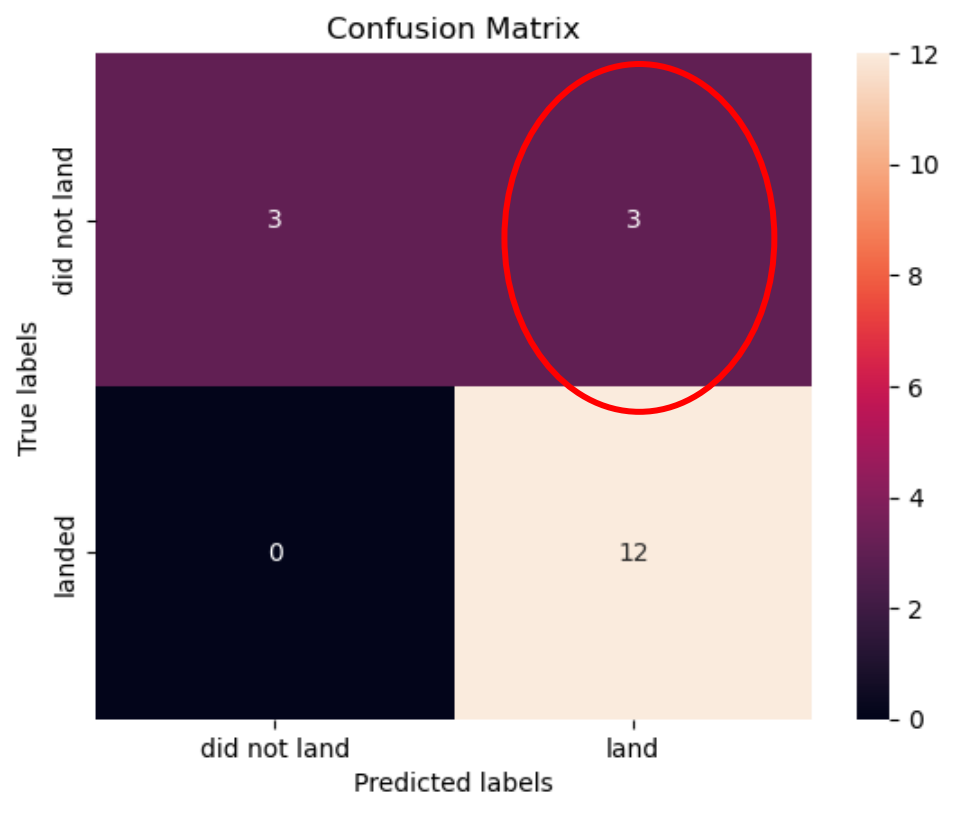
- Visualization of the built model accuracy for all built classification models, in a bar chart

- All models have the same accuracy of 0.833 on the same test data.



Confusion Matrix

- The confusion matrix for one of the best performing models; KNN. There seems to be a False Negative problem, shown in red circle below.



Conclusions

- As the flight number increases in each of the three launch sites, so does the success rate.
- Success rate depends on obits, certain orbits such as ES-L1, GEO, HEO & SSO have the highest success rate, whereas there are orbits such as SO with 0 success rate
- For certain orbits such as Polar, LEO and ISS, the successful landing is observed more frequently with heavy payloads
- The success rate kept increasing since 2013 till 2020
- all launch sites are in proximity to the Equator line and coasts
- All the machine learning models yielded same accuracy on the test data

Appendix

- The relevant Github repository including all the notebooks can be found here:

https://github.com/ckurter/Applied_Data_Science_Capstone/tree/main

- The course webpage:

https://www.coursera.org/learn/applied-data-science-capstone/

Thank you!