Winning Space Race with Data Science

Jen Patrick Nataba
October 25, 2024

Outline

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

Executive Summary

- This project will focus on predicing if the SpaceX Falcon 9 will successfully land using classification machine learning.

- The primary steps in this project involve:
  - Data collection using an API
  - Web scraping relevant information
  - Cleaning all the gathered for analyzation
  - Data visualization
  - Examining launch site data
  - Dashboarding using Plotly
  - Machine Learning Prediction

- This will lead to:
  - Results from initial analysis
  - What do these data mean?
  - Predictive results (Decision Tree)

Introduction

- Rocket launches have been one of the pinnacle of human advancements. These activities give us much information about the universe, giving us a better understanding of how we came to be. However, these launches cost much money. SpaceX Falcon 9 is advertised to cost around 62 million dollars much cheaper than other providers that offer 165 million dollars per launch. This is because SpaceX can reuse the first stages of their rocket launch. With that, how can we leverage this info to apply it to other rocket launches?

- We will answer:
  - Determine the price of each launch
  - Determine if first stage is reusable
  - Use machine learning to determine if SpaceX will reuse the first stage
  - Use this data to determine successful landing conditions

Section 1

Methodology

Methodology

We will go through this step by step:

1. Data collection methodology:
   - Data is collected through the SpaceX API and Web scraping data from wikipedia
2. Perform data wrangling
   - Data is cleaned using pandas library in python, applying one-hot encoding to categorical features
3. Perform exploratory data analysis (EDA) using visualization and SQL
   - Pandas
   - Numpy
   - SQL
4. Data Visualization using:
   - Matplotlib, Seaborn, Folium
   - Plotly
5. Classification prediction using machine learning:
   - Logistic Regression
   - K-Nearest-Neighbor
   - Decision Tree
   - Support Vector Machine

Data Collection

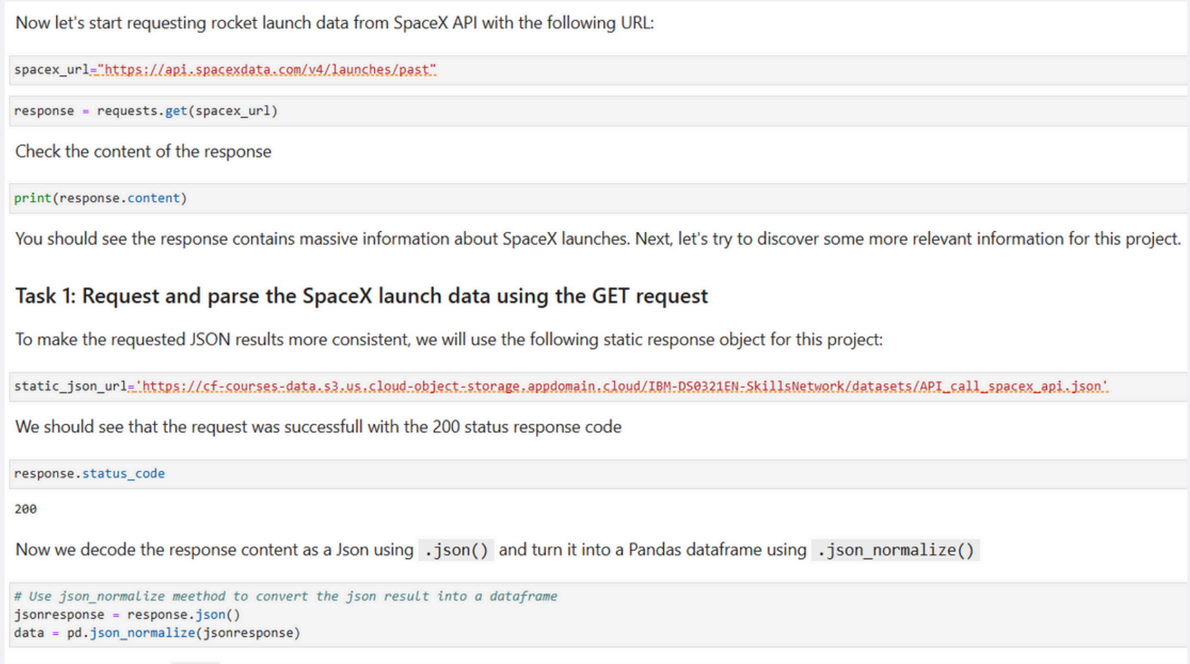
- How was the data collected?
  - Data was collected through the use of an API
  - Since we need more necessary data, web scraping was utilized on wikipedia sites
  - In order to read the data, it is decoded using .json() and then coverted into a dataframe using .json_normalize()
  - Data is then cleaned to fill in missing values using the mean and categorical values are converted using one-hot encoding

- Why do this?
  - These data is crucial for further analysis and machine learning
  - The aim is to extract the data from an API, use web scraping to extract launch data as HTML table, parse the table and then turn it into a pandas dataframe

Data Collection – SpaceX API

- We will start by using the get request to the given link, use a static url, then decode the data using .json, and finally turn it into a pandas dataframe using pd.json_normalize()

- Link to the notebooks: https://github.com/cytojen/applied-data-science-capstone.git



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)
```

Check the content of the response

```
print(response.content)
```

You should see the response contains massive information about SpaceX launches. Next, let's try to discover some more relevant information for this project.

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

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

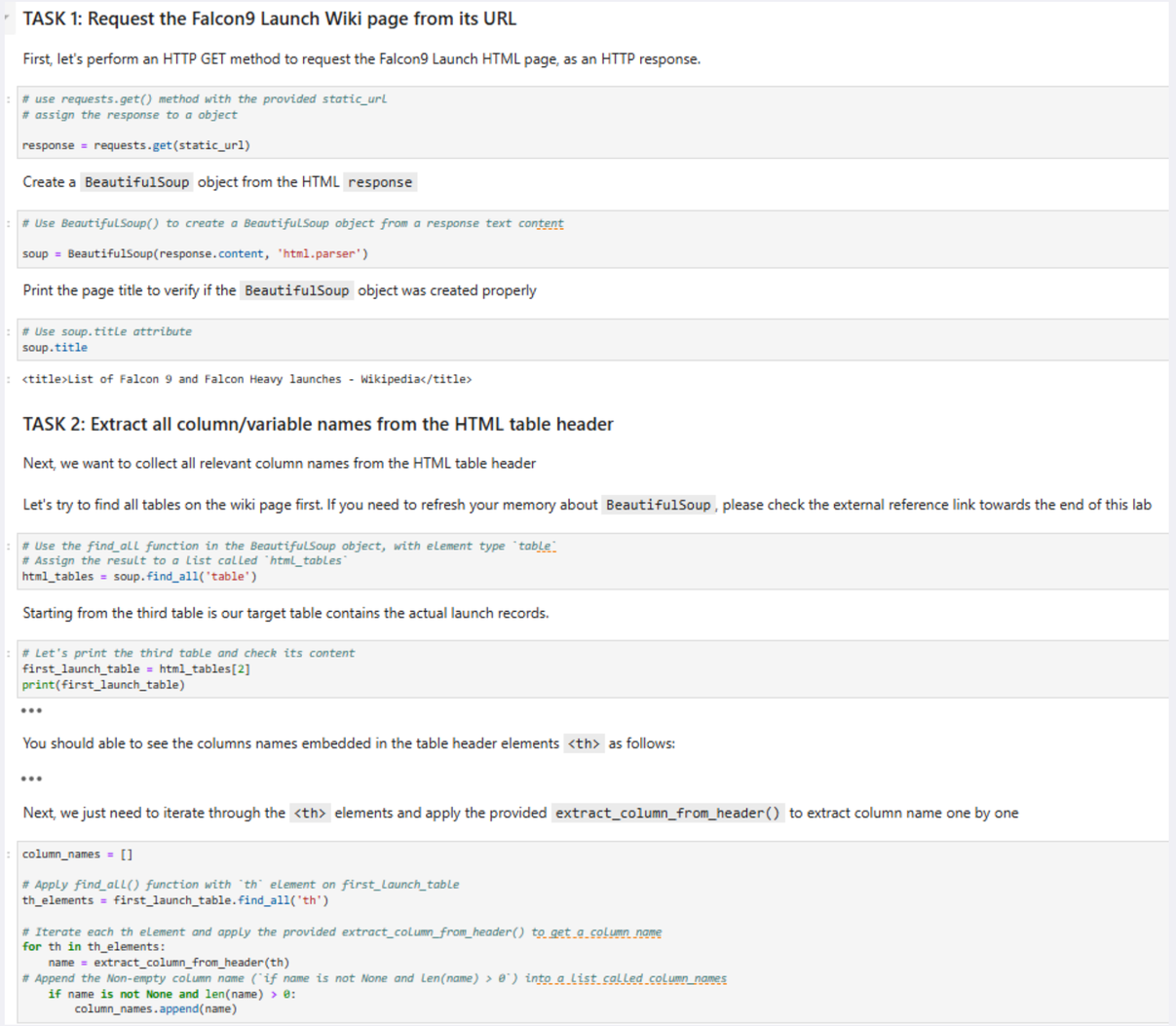
Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

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

Data Collection - Scraping

- We use a request get to the given url and use BeautifulSoup to create an object with the response content as the parameter, find all tables, and extract necessary data

- Link to the notebook: https://github.com/cytojen/applied-data-science-capstone/blob/main/2%20jupyter-labs-webscraping.ipynb



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

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
# use requests.get() method with the provided static_url
# assign the response to a object

response = requests.get(static_url)
```

Create a `BeautifulSoup` object from the HTML `response`

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content

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

Print the page title to verify if the `BeautifulSoup` object was created properly

```
# Use soup.title attribute
soup.title
```

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

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

Starting from the third table is our target table contains the actual launch records.

```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

...

You should able to see the columns names embedded in the table header elements `<th>` as follows:

...

Next, we just need to iterate through the `<th>` elements and apply the provided `extract_column_from_header()` to extract column name one by one

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
th_elements = first_launch_table.find_all('th')

# Iterate each th element and apply the provided extract_column_from_header() to get a column name
for th in th_elements:
    name = extract_column_from_header(th)
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
    if name is not None and len(name) > 0:
        column_names.append(name)
```

Data Wrangling

- Perform EDA on the dataset
- Determine the number of launces per site on each orbit
- Find if the launch is a success or not
- Use pd.get_dummies to do one-hot encoding on succesful and failed launches

- Link to the notebook: https://github.com/cytojen/applied-data-science-capstone/blob/main/3%20labs-jupyter-spacex-Data%20wrangling.ipynb

```
df.head(5)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

We can use the following line of code to determine the success rate:

```
df["Class"].mean()
```

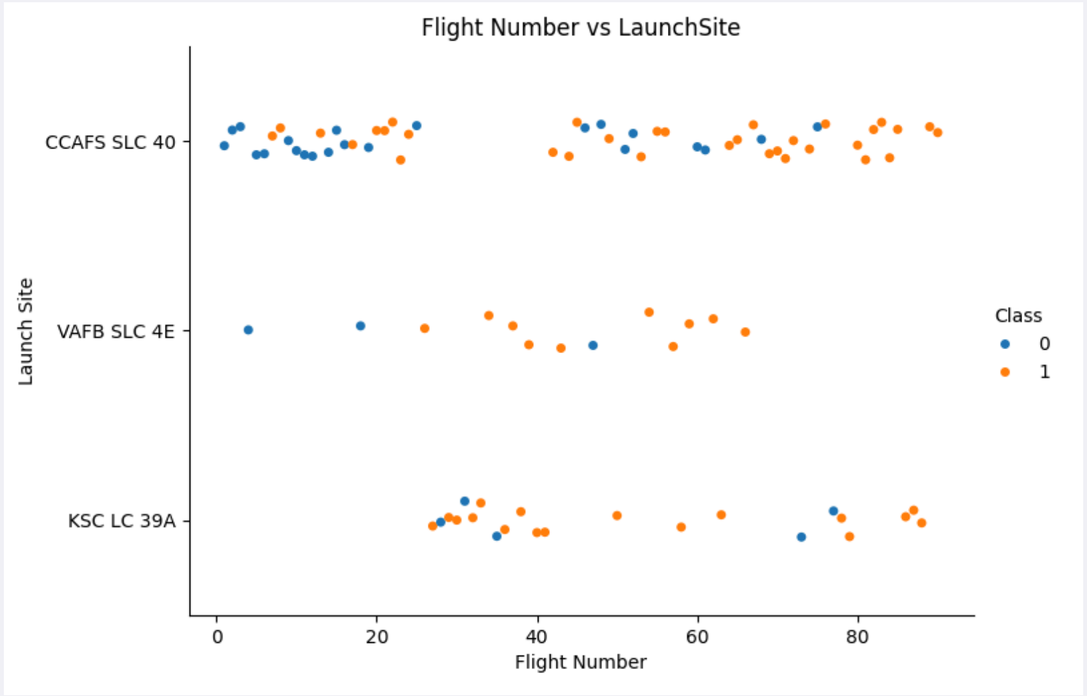
```
np.float64(0.6666666666666666)
```

We can now export it to a CSV for the next section,but to make the answers consistent, in the next lab we will provide data in a pre-selected date range.

```
df.to_csv("dataset_part_2.csv", index=False)
```

EDA with Data Visualization

- From the data gathered, this section explores the relationships between flight number and launch site, payload mass and launch site, success rate for each orbit type, payload mass vs orbit type and success rate over the years
- For more info visit: https://github.com/cytojen/applied-data-science-capstone/blob/main/5%20edadataviz.ipynb



EDA with SQL

- Using SQL, this section explores unique launch sites, payload mass carried by different variables, first successful landing outcome, number of successful mission outcomes, etc.
- For more information visit: https://github.com/cytojen/applied-data-science-capstone/blob/main/4%20jupyter-labs-eda-sql-coursera_sqllite.ipynb

Task 10

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.

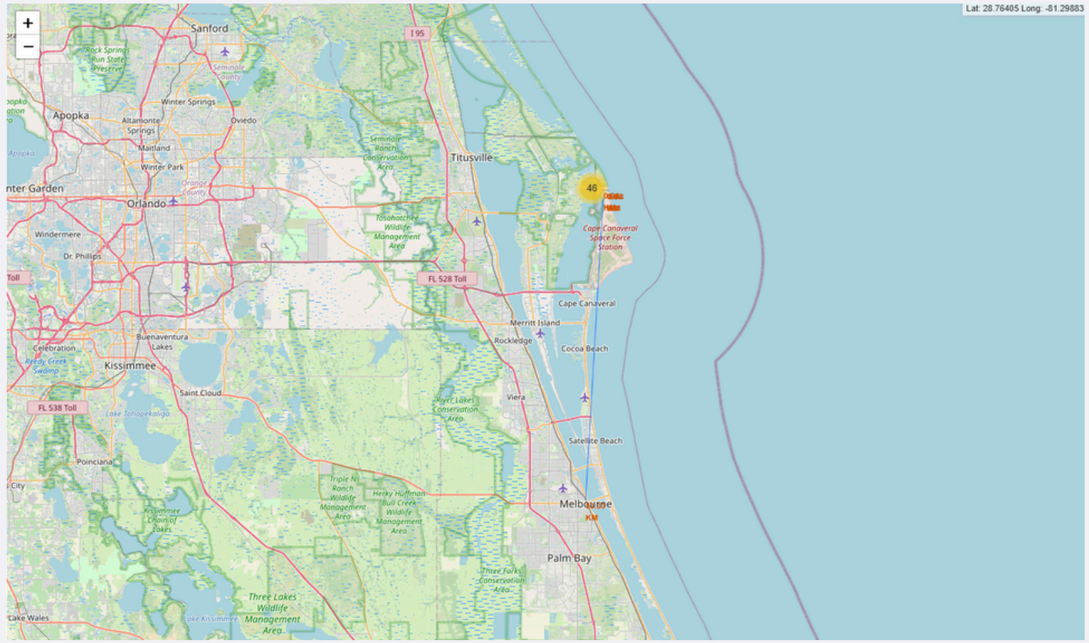
```
%%sql
    SELECT Landing_Outcome, COUNT(Landing_Outcome) AS Outcome_Count
    FROM SPACEXTBL
    WHERE (Landing_Outcome LIKE 'Success%' OR Landing_Outcome LIKE 'Failure%')
    AND Date BETWEEN '2010-06-04' AND '2017-03-20'
    GROUP BY Landing_Outcome
    ORDER BY Outcome_Count DESC;
```

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

Landing_Outcome	Outcome_Count
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Failure (parachute)	2

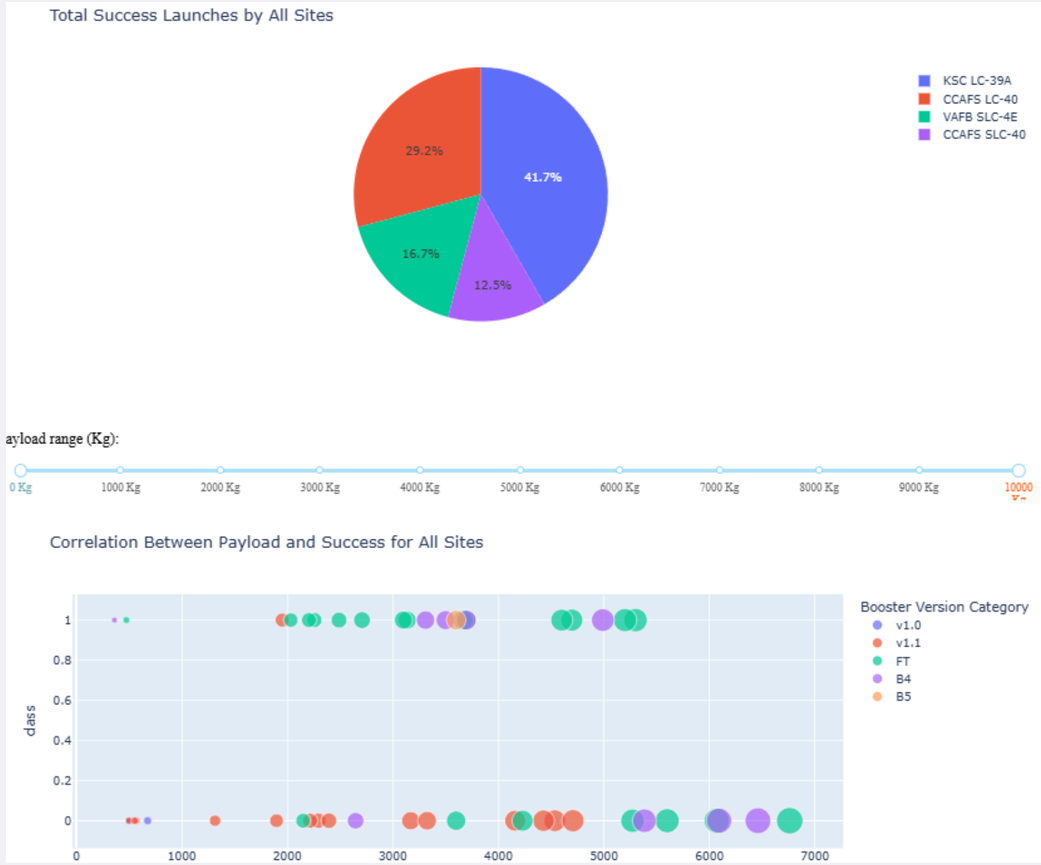
Build an Interactive Map with Folium

- In this section, I utilized folium to visualize the data better. Here we can see all launch sites, success/failed launches, distances between launch sites, and distance of launches with respect to cities, railroads, and coastlines
- I also utilized cluster markers to indentify the launch sites and their success rate
- For more info: https://github.com/cytojen/applied-data-science-capstone/blob/main/6%20lab_jupyter_launch_site_location.ipynb



Build a Dashboard with Plotly Dash

- I utilized Pie Charts and Scatter plots in displaying the data, these provide an easy way for normal people to understand the data quickly.
- Pie Charts for total success launches by all sites
- Scatter plot for Success vs Payload Mass
- For more info visit: https://github.com/cytojen/applied-data-science-capstone/blob/main/spacex_dash_app.py



Predictive Analysis (Classification)

- Different classification machine learning models were built and evaluated using GridSearchCV to find the best and appropriate hyperparameter.
- These models are improved using feature engineerin, accuracy tests, and analyzing confusion matrices.
- In preparing the data for machine learning, I followed a series of steps:
  - Standardize the data
  - Split the data into training and testing data
  - Creating the machine learning models
  - Fitting the train x and train y data
  - Using GridSearchCV to find the best hyperparameter
  - Evaluating the model based on accuracy scores and confusion matrices

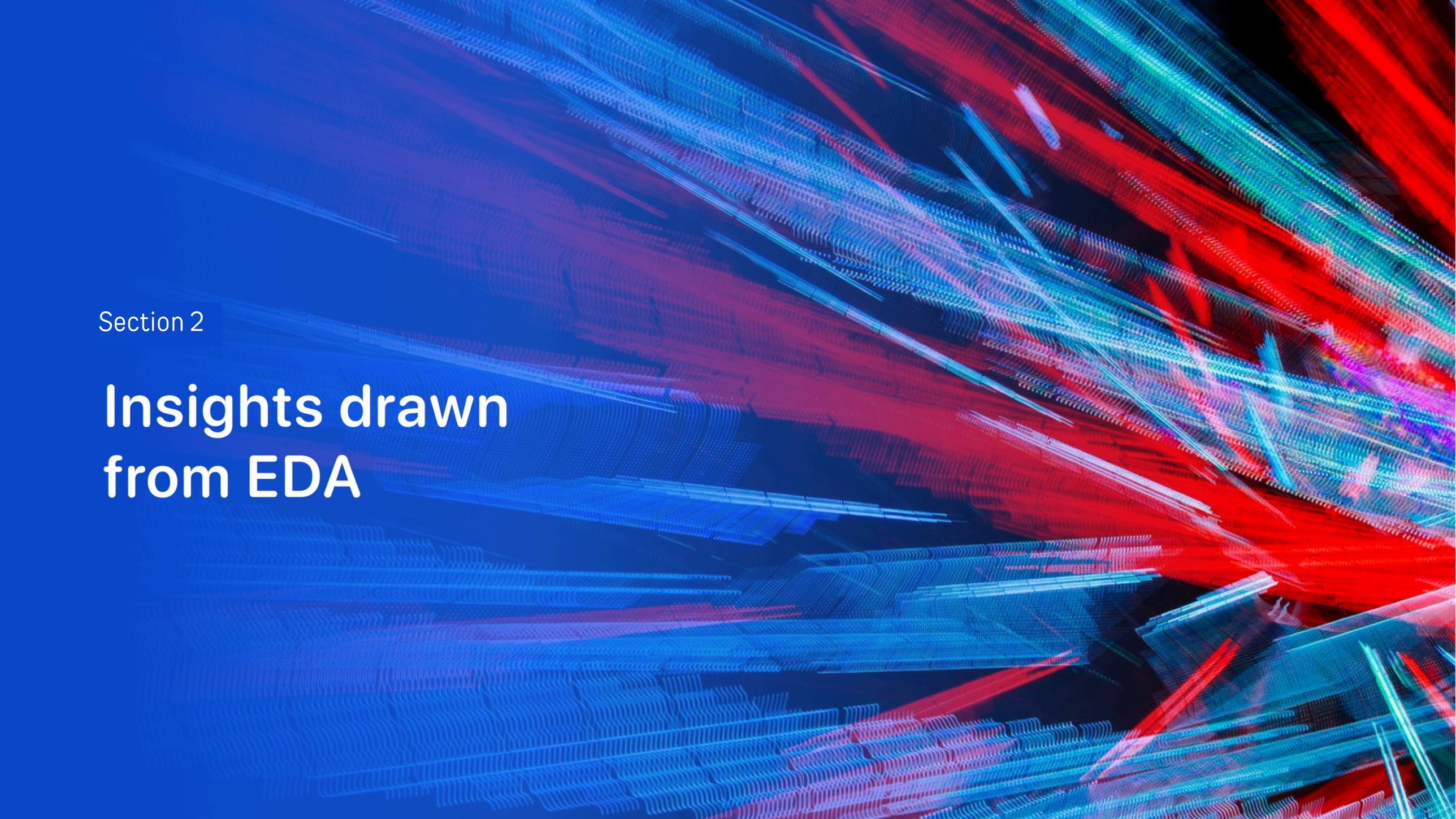
Results

- In summary:
  - SQL, Matplotlib, Seaborn, Folium, Dash, Machine Learning are used for analyzing and predicing results
  - Success rate of launches have dramatically improved
  - KSC LC-39A boasts the highest success rate of all landing sites.
  - It is ideal to launch rockets into orbits ES-LI, GEO, HEO and SSO as they have a 100% success rate

- Based on the folium map, most launch sites are near the equator and coastline. For safety measures, these launch sites are far away from infrastructures (city, railroads, highway)
- Out of all the models, decision search trees had the highest accuracy (87%) of determining wether a rocket launche’s first stage will land

Section 2

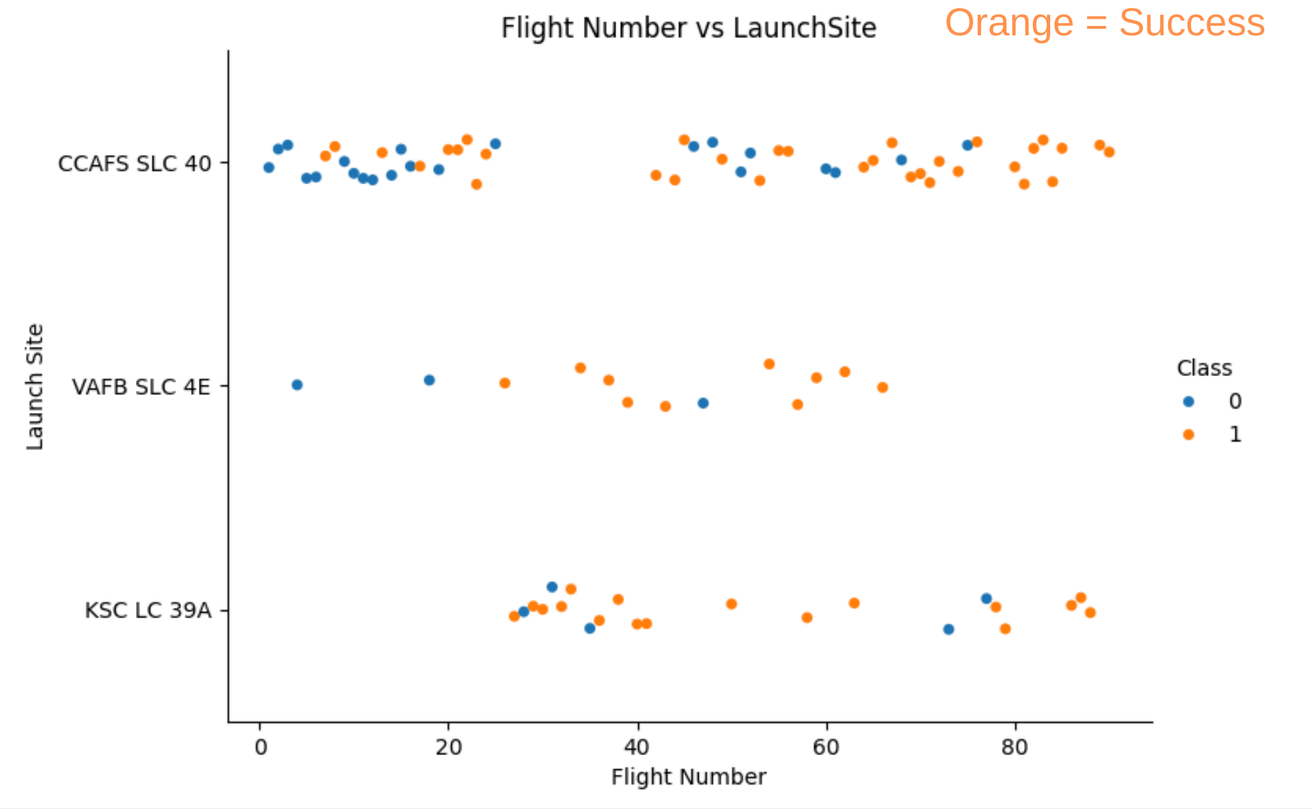
Insights drawn from EDA



Flight Number vs. Launch Site

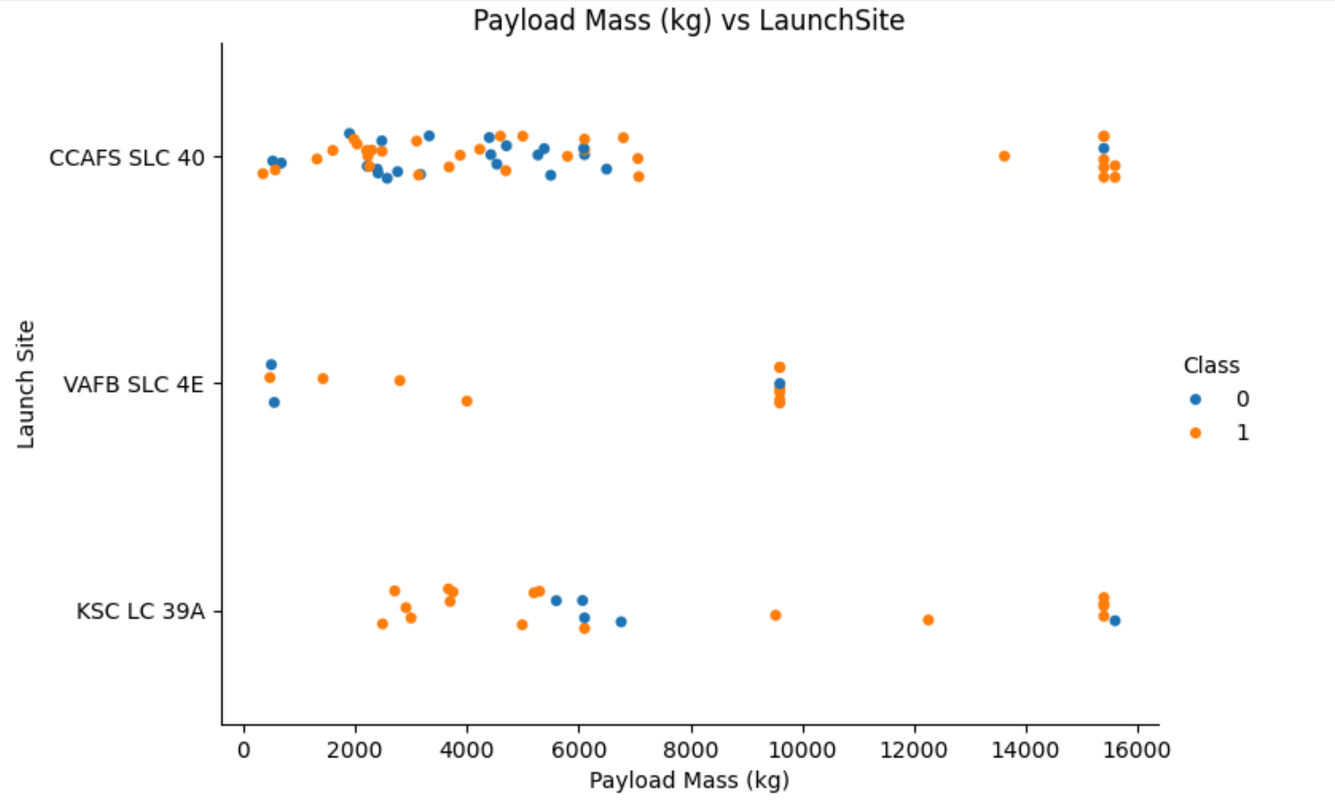
- Flights from earlier eras had a **lower success rate**
- Newer flights had a **higher success rate**
- CCAFS SLC 40 Launch site had the highest number of flights while VAFB SLC 4E is the opposite
- VAFB SLC 4E has the highest success rate
- CCAFS SLC 40 has a 100% success rate recently.
- Therefore, **Newer flights** have a **higher success rate**

Blue = Fail
Orange = Success



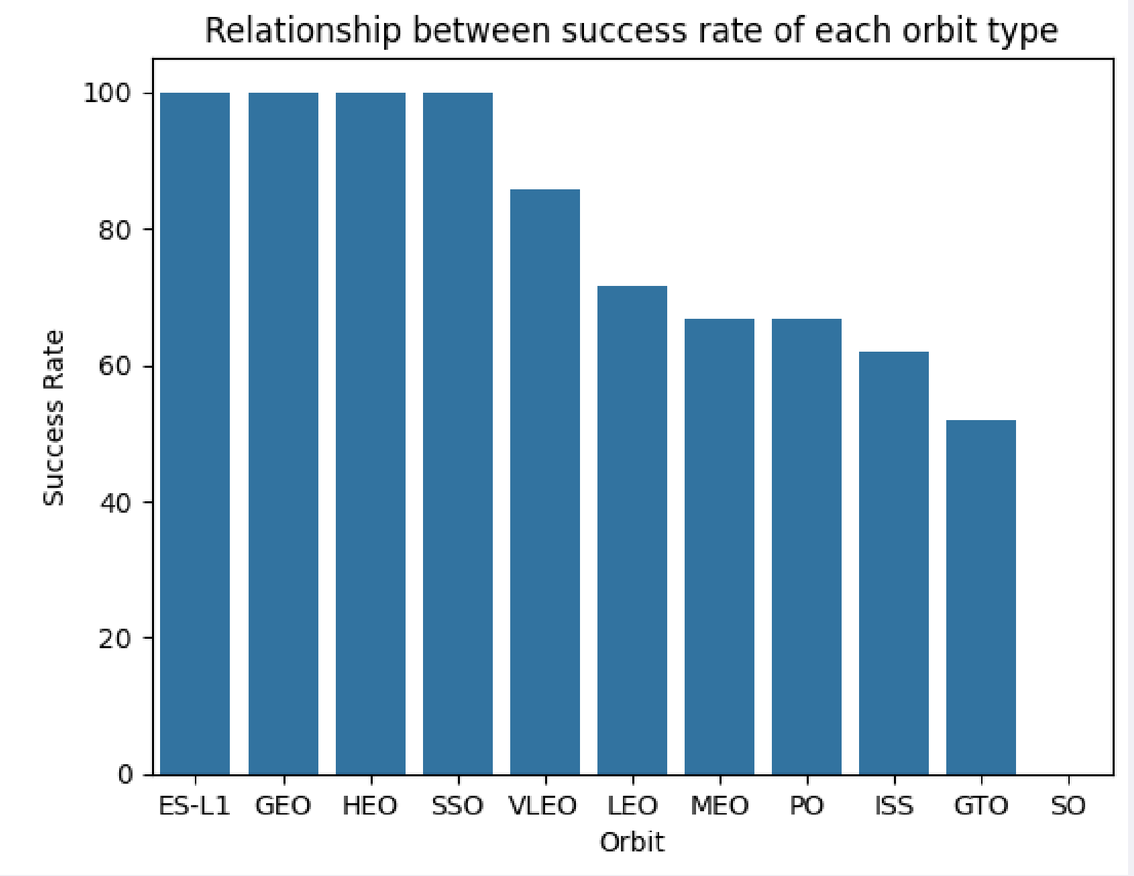
Payload vs. Launch Site

- CCAFS SLC 40 has the most number of flights with a payload mass of under 8000 kg
- **KSC LC 39A** has a **100% success rate** on flight's payload mass **below ~5300 kg**
- There is a **high chance of success** on payload masses of around **14000-16000 kg**
- VAFB SLC 4E has not exceeded 10000 kg payload mass when launching
- **Most flights** of payload mass **> 7000 kg** has a **high success rate**



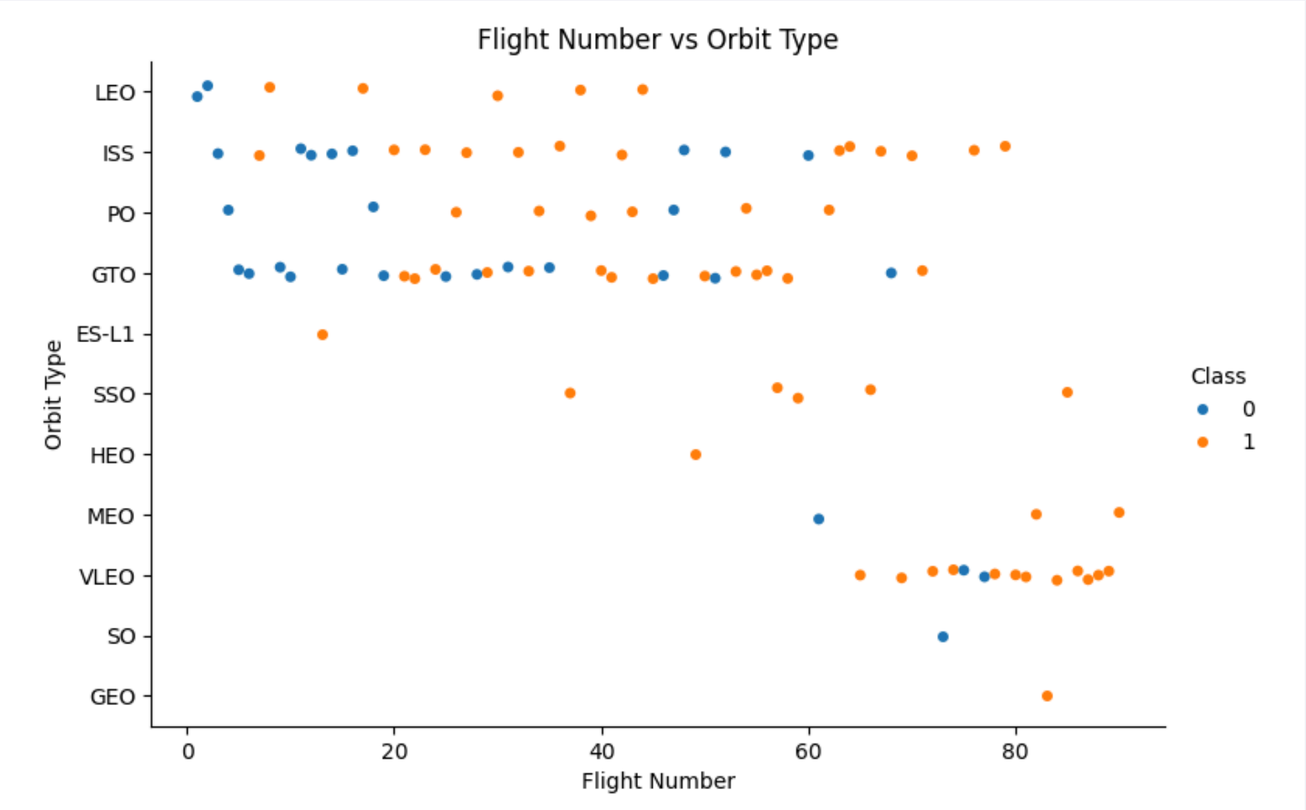
Success Rate vs. Orbit Type

- **ES-L1, GEO, HEO, SSO** has a **100% success rate**
- VLEO, MEO, PO, ISS, GTO has success rates of around 50%-80%
- SO has a 0% success rate
- Safe to say, **launching on orbits ES-L1, GEO, HEO, SSO** is a **good idea**



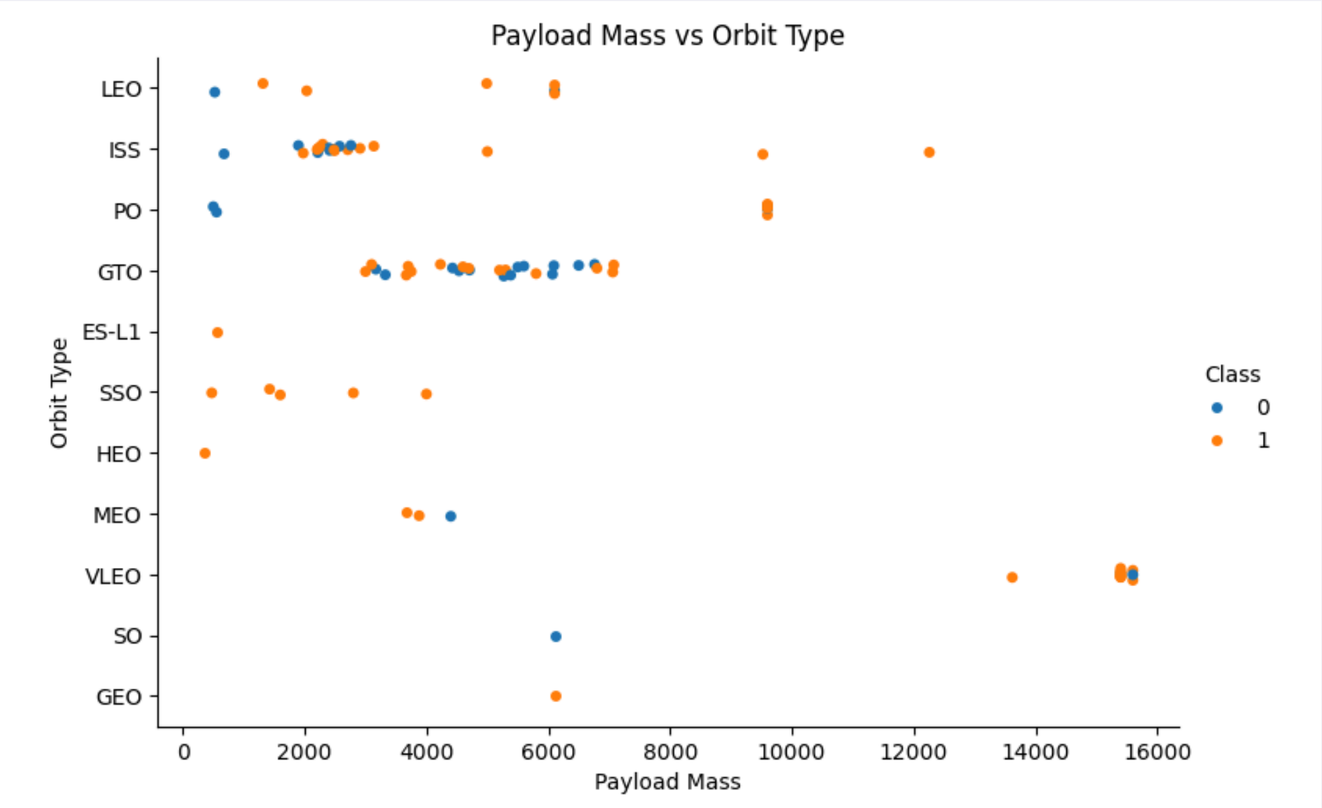
Flight Number vs. Orbit Type

- **ES-L1 and GEO** only had **1 flight** which was a **success**
- SO had 1 flight but failed
- GTO and ISS had the most number of flights
- VLEO has a high success rate
- **As flight number increases, so does the success rate**



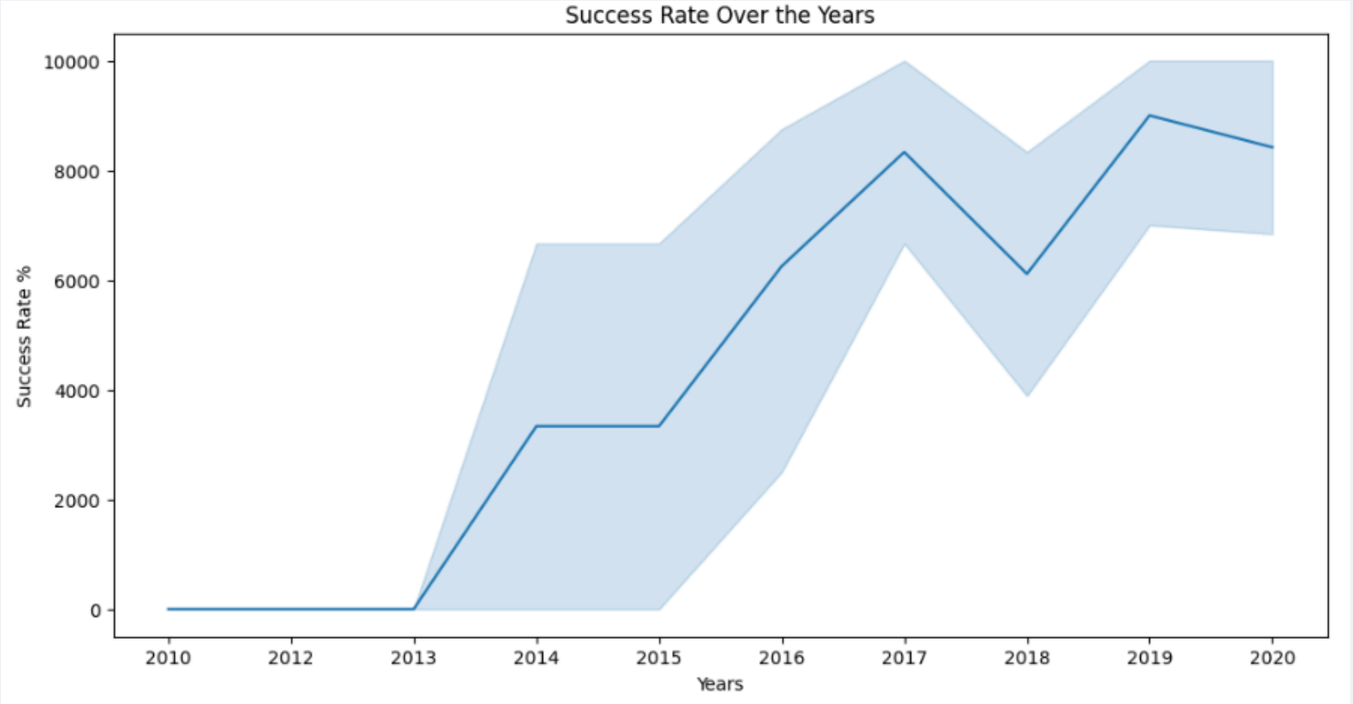
Payload vs. Orbit Type

- SSO only have flight's where payload mass is less than ~4500 kg
- **VLEO** has the **heaviest** payload mass flight around 15000 kg
- **GTO** does not exceed 8000 kg payload mass and has **mixed outcomes**
- **Heavier payloads** tend to **succeed more**



Launch Success Yearly Trend

- As time goes on, launch success rate **improves dramatically**
- Success rate decreased on 2018 and 2020
- Overall, **launch success** has **improved** over the years



All Launch Site Names

- Using SQL, this shows all the launch sites:
  - CCAFS LC-40
  - VAFB SLC-4E
  - KSC LC-39A
  - CCAFS SLC-40

```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL;
```

```
 * 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'

- This query finds launch sites that begin with ‘CCA’ using wildcard and limiting the results by 5
  - CCAFS LC-40

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

```
 * 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

- This query calculates the total payload mass from NASA (CRS)
- There is a total of **45,596 Kg** carried by boosters launched by **NASA (CRS)**

Task 3

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

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

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

SUM_PAYLOAD_MASS_KG	Customer
45596	NASA (CRS)

Average Payload Mass by F9 v1.1

- This query displays the average payload mass carried by booster version F9 v1.1
- An average of **2534.67 Kg** payload mass was carried by **booster version F9 v1.1**

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD_MASS_KG, Booster_Version FROM SPACEXTBL WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

AVG_PAYLOAD_MASS_KG	Booster_Version
2534.6666666666665	F9 v1.1 B1003

First Successful Ground Landing Date

- This query finds the first ever successful ground landing date
- The **first successful ground landing** was recorded at **December 22, 2015** by orbcomm on orbit LEO carrying satellites and a booster of **F9 FT B1019**.
- This happened on launch site **CCAFS LC-40**.

```
%sql SELECT * FROM SPACEXTBL WHERE Date = (SELECT MIN(Date) FROM SPACEXTBL WHERE Landing_Outcome = 'Success (ground pad)');
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

Successful Drone Ship Landing with Payload between 4000 and 6000

- This query lists all the names of boosters which have successfully landed on drone ship and had payload mass **greater than 4000 but less than 6000**

- There are only **4 booster versions** each with their respective payloads who have landed on a drone ship with the given condition.

```
%sql SELECT Booster_Version, Payload FROM SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' AND (PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000);
```

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

Booster_Version	Payload
F9 FT B1022	JCSAT-14
F9 FT B1026	JCSAT-16
F9 FT B1021.2	SES-10
F9 FT B1031.2	SES-11 / EchoStar 105

Total Number of Successful and Failure Mission Outcomes

- This query displays all the failed and succesfsul flight missions
- There are a total of **99 success** flight missions, **1 failure** (in flight), and **1 Success** (payload status unclear)

```
%sql SELECT MISSION_OUTCOME, COUNT(*) FROM SPACEXTBL GROUP BY Mission_Outcome;
```

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

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

Boosters Carried Maximum Payload

- This query lists the names of the booster which have carried the maximum payload mass
- We can see that the results are mostly **starlink payloads** with maximum payload mass of **15600 Kg**

```
%sql SELECT Booster_Version, Payload, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

2015 Launch Records

- This query returns the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015
- There are two fails in 2015
  - **January 2015**, drone ship fail on launch site CCAFS LC-40 carrying F9 v1.1 B1012 boosters
  - **April 2015**, drone ship fail on launch site CCAFS LC-40 carrying F9 v1.1 B1015 boosters

```
%%sql SELECT substr(Date, 6, 2), substr(Date, 0, 5), Landing_Outcome, Booster_Version, Launch_Site
    FROM SPACEXTBL
    WHERE substr(Date, 0, 5) = '2015' AND Landing_Outcome = 'Failure (drone ship)';
```

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

substr(Date, 6, 2)	substr(Date, 0, 5)	Landing_Outcome	Booster_Version	Launch_Site
01	2015	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	2015	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- This query returns 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
- **Drone ships** has the **highest succes** but also the **highest failure** in landing outcomes
- **Ground pad landing** outcome has **3 successful** attempts and **parachute** has **2 failed attempts**

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.

```
%%sql
    SELECT Landing_Outcome, COUNT(Landing_Outcome) AS Outcome_Count
    FROM SPACEXTBL
    WHERE (Landing_Outcome LIKE 'Success%' OR Landing_Outcome LIKE 'Failure%')
    AND Date BETWEEN '2010-06-04' AND '2017-03-20'
    GROUP BY Landing_Outcome
    ORDER BY Outcome_Count DESC;
```

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

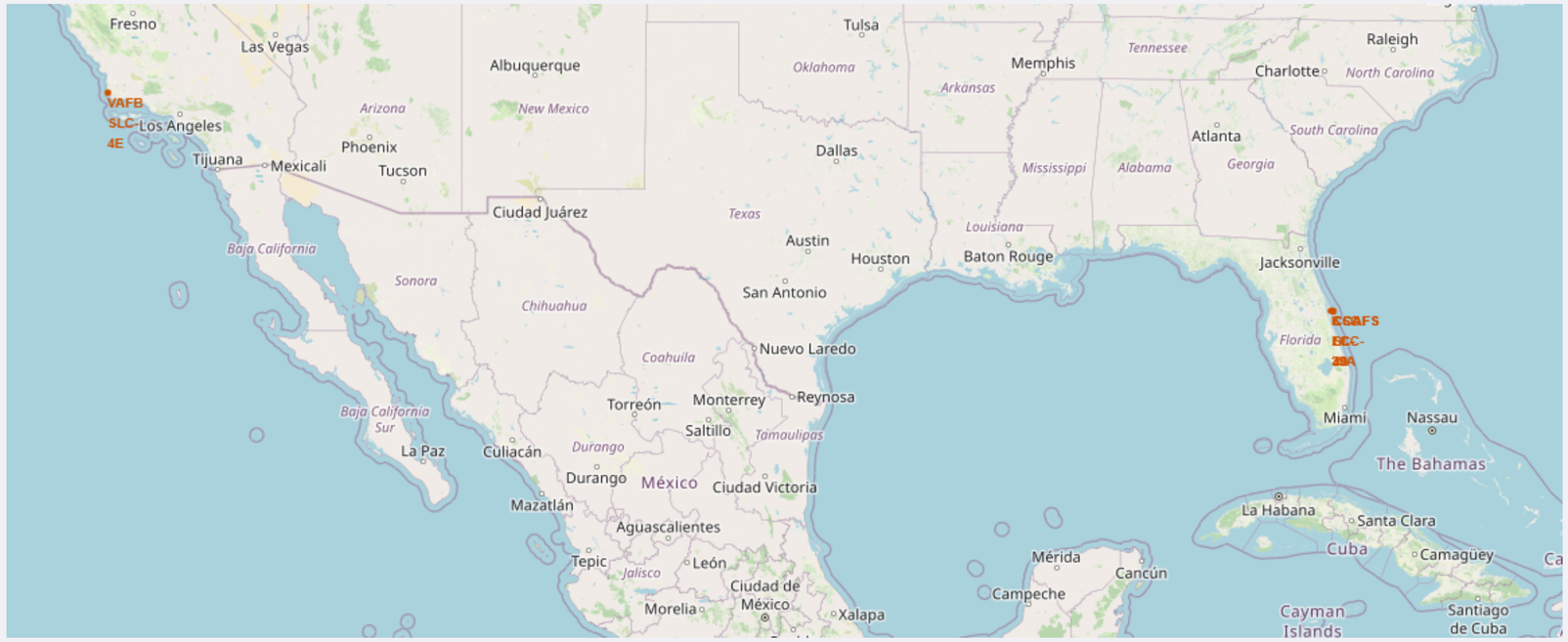
Landing_Outcome	Outcome_Count
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Failure (parachute)	2

Section 3

Launch Sites Proximities Analysis

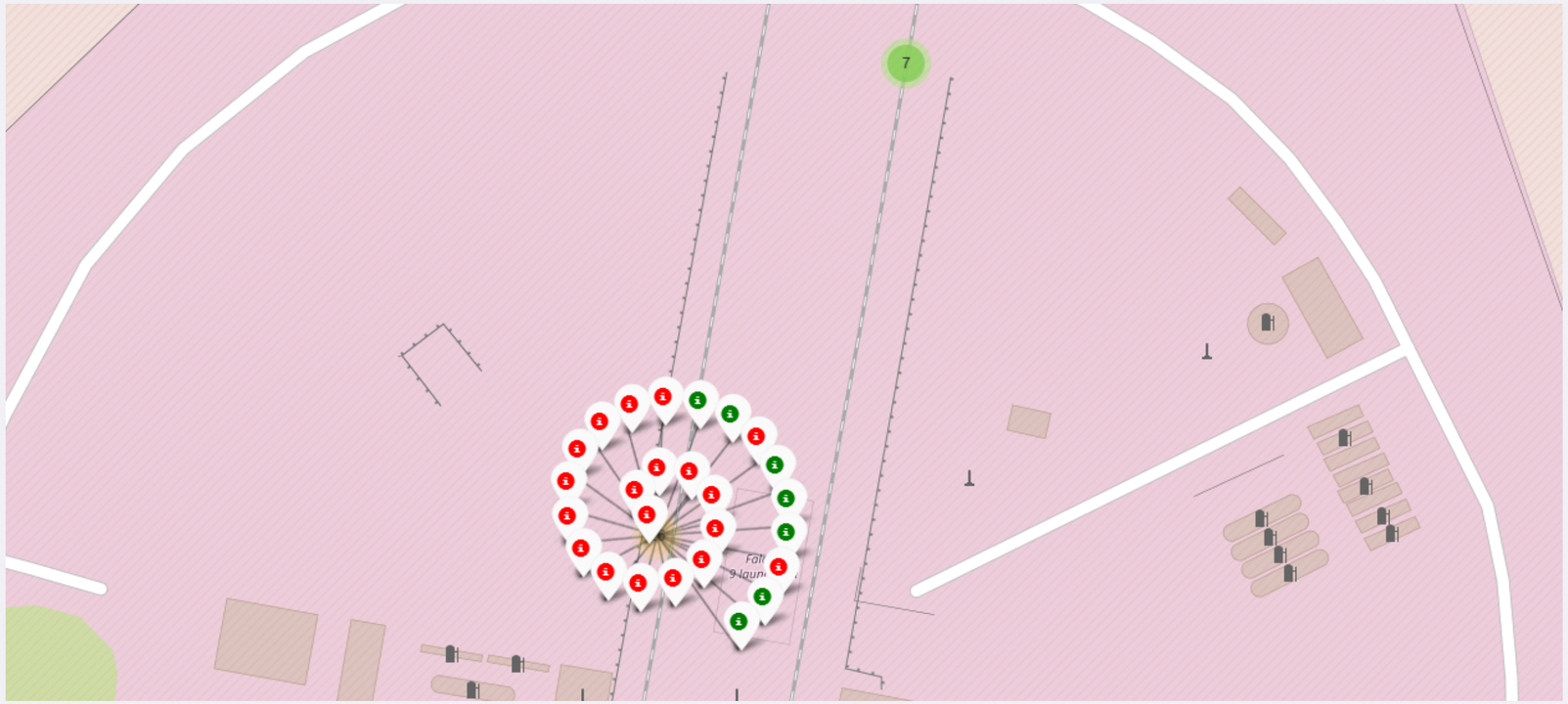
All SpaceX launch sites

- This folium map marks all the SpaceX launch sites
- There is one launch site in **Florida** and one in **Los Angeles**

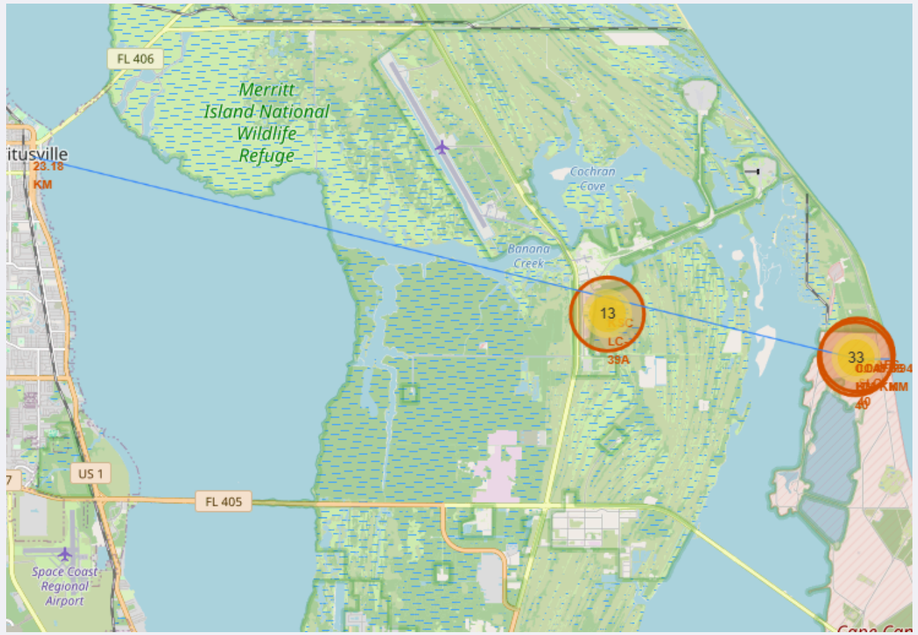


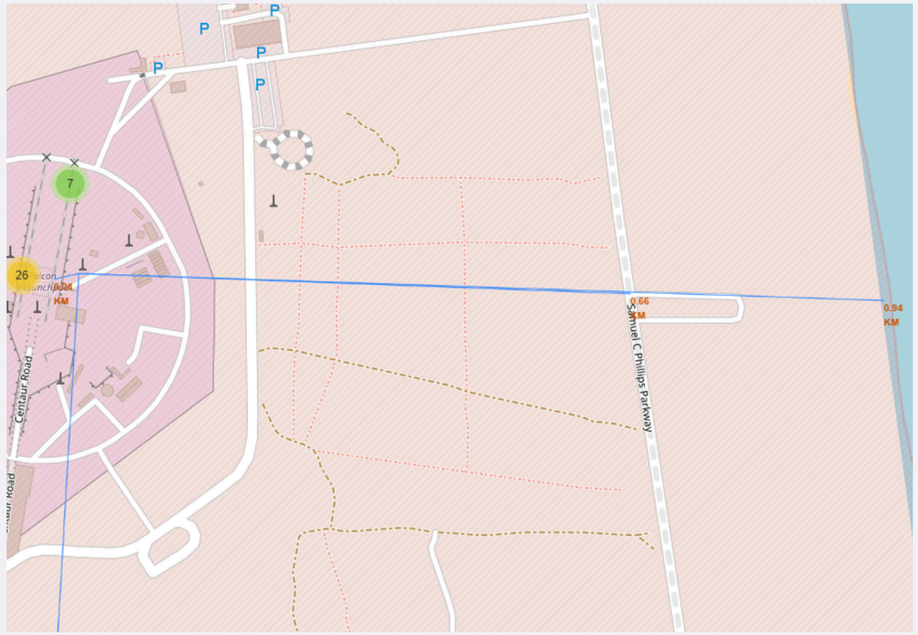
Success/failed launches for each site

- This folium map marks the succesful and failed launches for each site
- **RED** indicates a **failed launch**
- **GREEN** indicates a **successful launch**

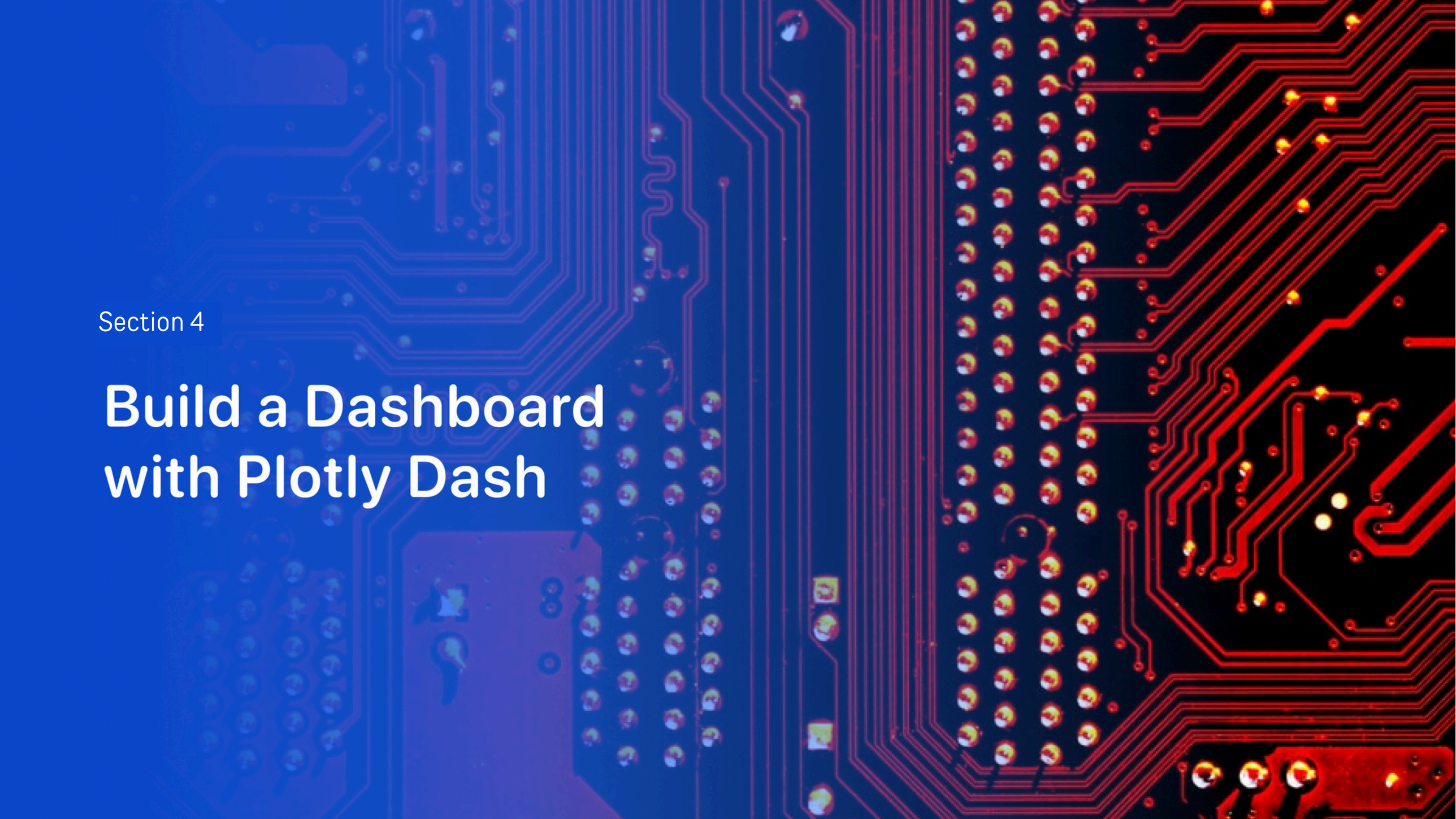


Distance Proximities





- In launch site **CCAFS LC 40**, it is **0.94 KM** away from the **coastline** and **0.66 KM** away from a **highway**
- It is also about **23.18 KM** away from the city of **Titusville**
- A **railroad** is very close to it (0.04 KM) making it very **easy to transport** necessary materials around the site

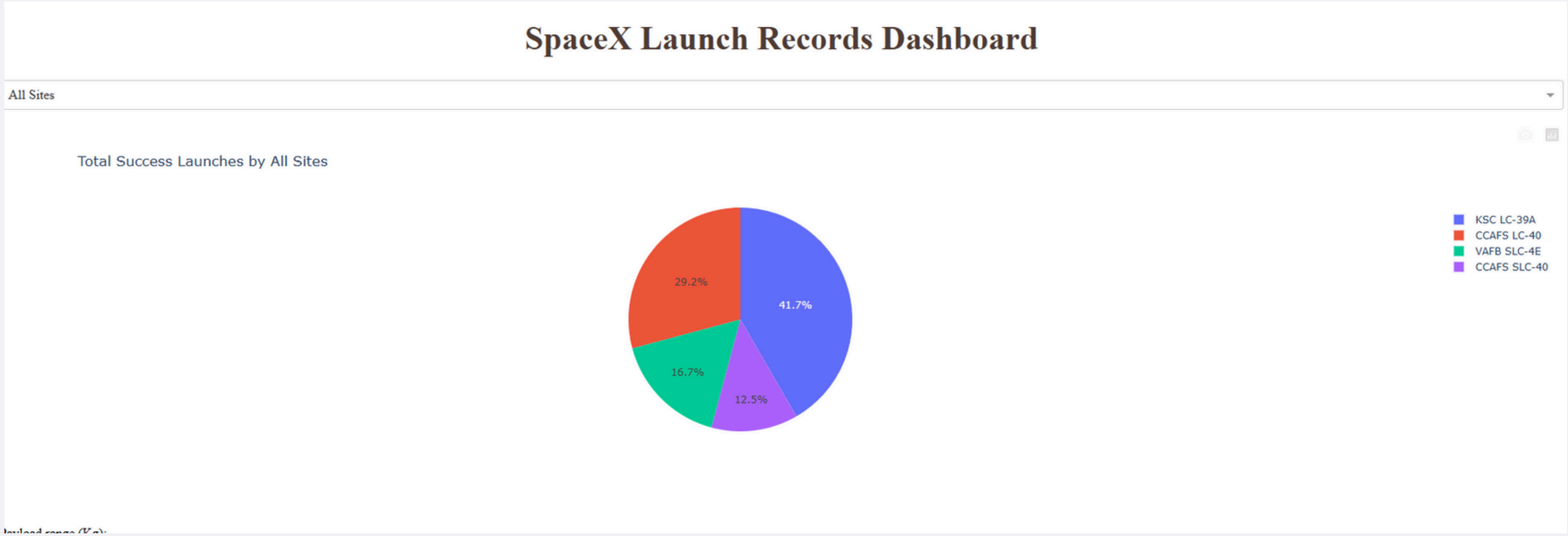


Section 4

Build a Dashboard with Plotly Dash

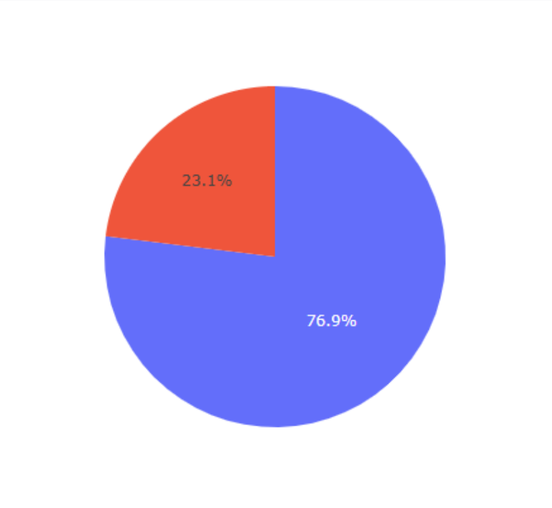
Total Succes Launches by All Sites

- In this pie chart, we can see that KSC LC-39A has the most number of succesful launch with CCAFS LC-40 following second
- **CCAFS SLC-40** has the **least number** of **succesful** launches

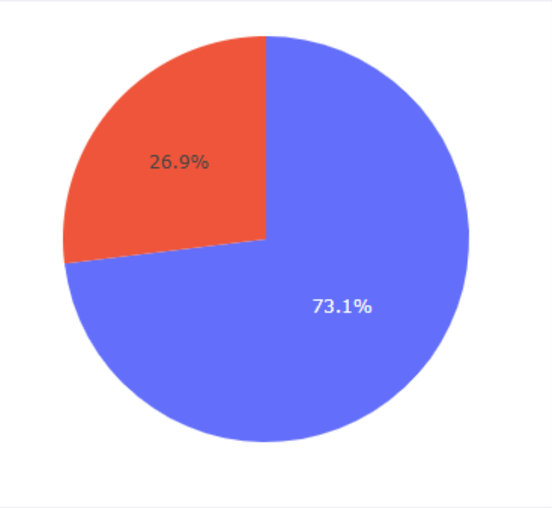


Launch Success vs Fails for each site

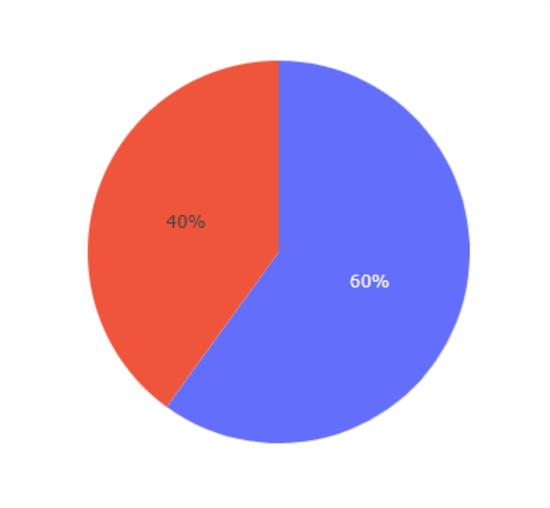
RED indicates Fail
GREEN indicates Success



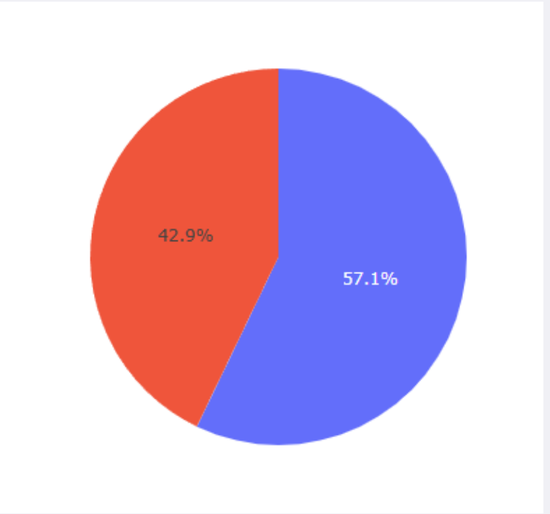
KSC LC-39A



CCAFS LC-40



VAFB SLC-4E

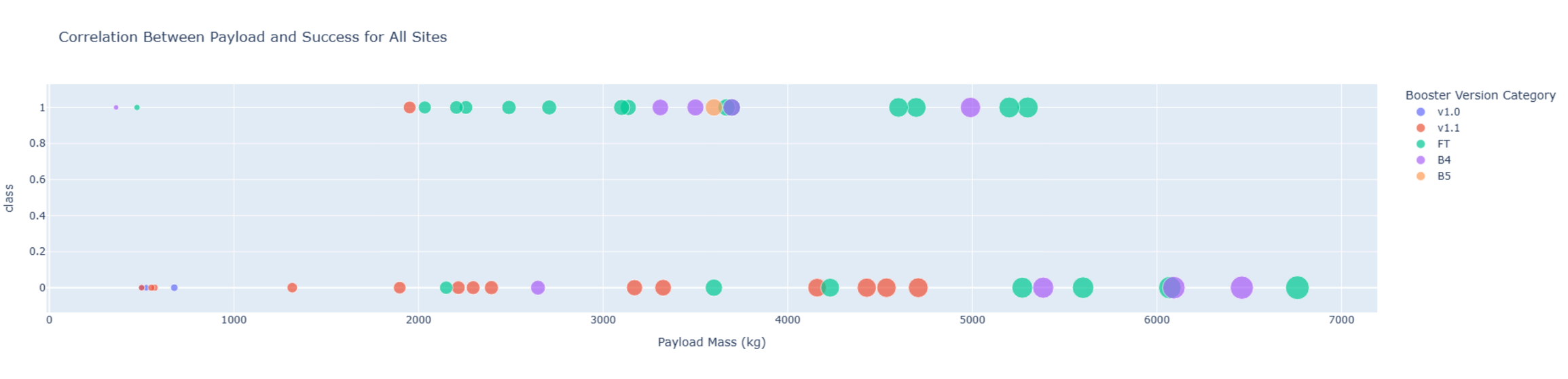


CCAFS SLC-40

- This shows that **KSC LC-39A** has the **highest success rate** among the 4 launch sites and the lowest failure rate too

Payload Mass (Kg) vs Success Rate

- **Booster Version FT** has a **high success rate** when payload mass is less than **5500 Kg**
- **Booster Version v1.1** has a **high failure rate** below a payload mass of **4800 kg**, this booster does not exceed beyond that weight
- There are **more failures** than success



Section 5

Predictive Analysis (Classification)

Classification Accuracy

- From all the models that we tested, with their best hyperparameters chosen using GridSearchCV, **Decision Tree has the Highest Accuracy**

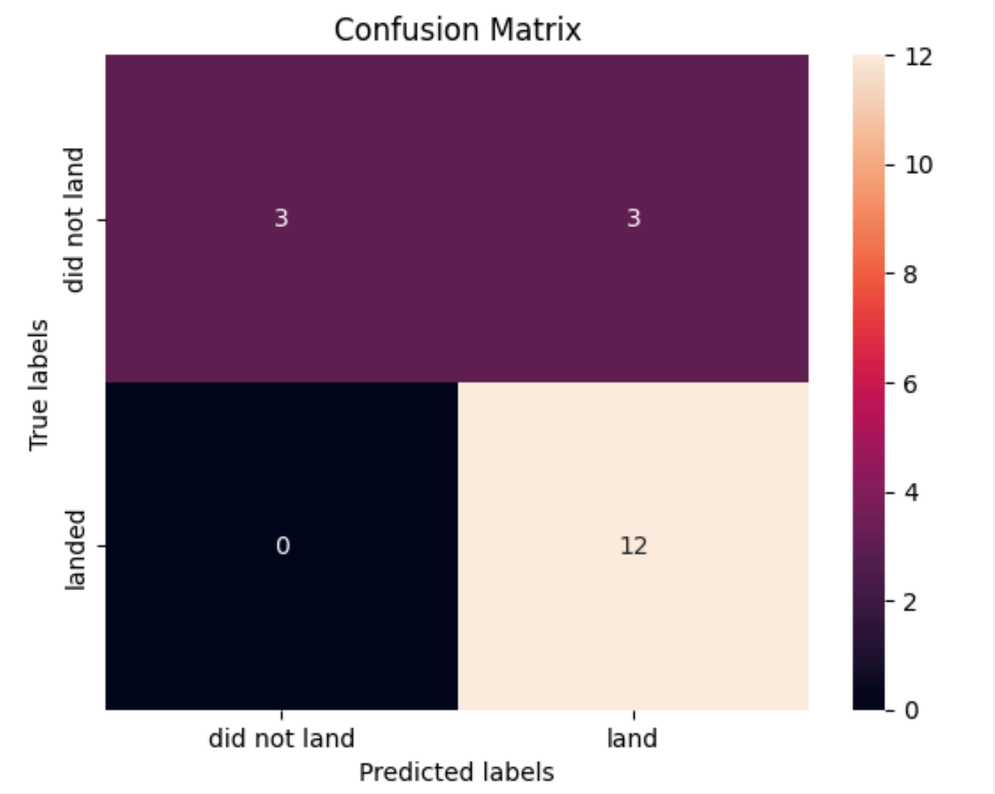
- All 4 models have scores **close to each other** but decision tree **outperforms** all the other models by **~3%**

```
score_list = {
    'Logistic regression': [logreg_cv.best_score_],
    'SVM': [svm_cv.best_score_],
    'Decision tree': [tree_cv.best_score_],
    'KNN': [knn_cv.best_score_]
}

score_list = pd.DataFrame(score_list)

score_list = score_list.transpose()

score_list.columns = ['Score']

score_list
```

	Score
Logistic regression	0.846429
SVM	0.848214
Decision tree	0.876786
KNN	0.848214

Confusion Matrix

- From this confusion matrix, the model outputs:
  - 3 True Negatives
  - 3 False Postives
  - 0 False Negative
  - 12 True Positive
- This means the model is **not that effective** at **predicting** instances of **successful landings**



Conclusions

- A **higher payload mass** (Kg) has a **higher chance** of **success rate**.
- **Most launch sites** are **built near the equator**, because the Earth's rotation provides an additional velocity boost, making it easier and more efficient to launch rockets into space
- **All launch sites** are **near coastlines** to ensure safety during launches, allowing rockets to ascend over water and reducing the risk to populated areas in case of a failure.
- **Launch success rate** has **risen** over the years indicating technological advancements.
- The **SSO orbit** has a **100% success rate**, making it an **excellent** candidate for launching.
- The **VLEO orbit** also has a **high success rate**, both these orbits already have **numerous launches** as compared to the other ones who have a few.
- **KSC LC 39-A** has the **highest success rate** out of all the launch with a **100% success rate** for launches less than **5,500 Kg**.

Recommendations

- Since the decision tree classification model had a high accuracy but it also had 3 false positives, this calls for improvement.
    - Use **XGBoost**, an advanced boosting algorithm known for its high performance and ability to reduce errors in classification tasks. (Or **Random Forest**)
- Add **more comments** to the notebook
- **Use time-series analysis**: If applicable, explore trends and seasonality in the data to enhance predictive accuracy.
- A **larger dataset** with **fewer missing values**

Thank you!