
# Winning Space Race with Data Science

Zhanna Santybayeva
10 May 2022

# Outline

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

# Executive Summary

- Summary of methodologies
  - Data sourcing:
    - SpaceX REST API
    - Web scraping
  - Data wrangling
  - Exploratory Data Analysis
    - EDA with SQL
    - EDA with Visualization
  - Data Visualization:
    - Dataviz with Folium maps
    - Interactive visual Analysis and Dashboarding
  - Machine Learning:
    - Predictive analysis with Machine Learning
- Summary of all results
  - EDA: relationship between different features established
  - Interactive dashboards: Launch sites with highest success rate and relationship between Payload mass and launch outcome established
  - Predictive analysis: the best model to predict launch outcome was found

# Introduction

- Project background and context

In 2015, Falcon 9 became the first rocket with *reusable* first-stage boosters to land successfully after delivering a payload into orbit. This makes Falcon 9 a financially more attractive option for space flights with a cost of *only* 67 million USD*, compared to its counterparts by other companies with flight costs reaching double or triple this price.

The priority in commercializing space flights is to ensure their safety. We aim to predict the flight outcome with Machine Learning models depending on the features of the rocket, payload, launch site etc.

Problems you want to find answers

- What are decisive **factors** for successful launches and landings?
- What is an **optimal** combination of such factors as engine type, payload mass, orbit type, and launch site?
- Can we predict launch **outcome** based on existing data?

* https://www.spacex.com/media/Capabilities&Services.pdf as of March 2022

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:

  Data was collected with SpeceX REST API (requests) and web scraping from Wikipedia page (BeautifulSoup).

- Perform data wrangling

  Data collected with API and web scraping was cleaned, parsed and converted to a Pandas dataframe.

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

# Data Collection

Data was collected with two methods: using REST API provided by SpaceX and Web scraping from the Wikipedia page.

## 1. SpaceX REST API:

- Send GET requests to spacex_url = "https://api.spacexdata.com/v4/launches/past"
- Convert response object to json .json() and convert it to a Pandas dataframe with .json_normalize(jresponse)

## 2. Web scraping:

- Extract a table from the Wikipedia page as a text https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922
- Use BeautifulSoup package to parse the text and to extract useful data by relevant HTML tags

# Data Collection – SpaceX API

- 1. Send a GET request to the SpaceX REST API url
- 2. Convert to json and to Pandas dataframe
- (the resulting response is not standardized for our use)
- 3. Select useful data and convert to a dictionary
- 4. Convert back to dataframe
-

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

```
response = requests.get(spacex_url)
```

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

```
# Get the head of the dataframe
data.head()
```

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

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
```

```
# Create a data from launch_dict
launch_data=pd.DataFrame.from_dict(launch_dict)
```

```
# Show the head of the dataframe
launch_data.head()
```

| | FlightNumber | Date | BoosterVersion | PayloadMass | Orbit | LaunchSite | Outcome | Flights | GridFins | Reused | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2006-03-24 | Falcon 1 | 20.0 | LEO | Kwajalein Atoll | None None | 1 | False | False | |

GitHub URL of the completed SpaceX API calls notebook

# Data Collection - Scraping

1. Extract a Falcon 9 launch records in a table from Wikipedia

2. Parse the HTML text with BeautifulSoup

3. Convert it into a Pandas dataframe by finding relevant columns by their tags.

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

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response, "html.parser")
```

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

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

```
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
    # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
```

```
df=pd.DataFrame(launch_dict)
df.head()
```

| | Flight No. | Launch site | Payload | Payload mass | Orbit | Customer | Launch outcome | Version Booster | Booster landing | Date | Time |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | CCAFS | Dragon Spacecraft Qualification Unit | 0 | LEO | SpaceX | Success\n | F9 v1.0B0003.1 | Failure | 4 June 2010 | 18:45 |
| 1 | 2 | CCAFS | Dragon | 0 | LEO | NASA | Success | F9 v1.0B0004.1 | Failure | 8 December 2010 | 15:43 |

GitHub URL of the completed web scraping notebook

# Data Wrangling

Pre-processing for both data sets:
- only relevant data was retained,
- NaN and missing values were replaced by mean values,
- Boolean values turned to numeric Booleans (0 and 1)

Data wrangling mainly concerned counting certain column values like number of launches per Orbit type or launch outcomes.

A new column was added representing a classification of these outcomes as 0 and 1.

GitHub URL of the completed data wrangling notebook

# EDA with Data Visualization

From obtained data, we need to establish relationship between different features. It is often very easily done by plotting two of the features of interest against each other in a scatter plot.

Following relationships were examined:

- Flight Number vs Launch Site
- Payload vs Launch Site
- Success rate of each Orbit type
- Flight Number vs Orbit type
- Payload vs Orbit type
- Launch success yearly trend

Moreover, categorical data was encoded using a OneHotEncoder

GitHub URL of the completed EDA with data visualization notebook

# EDA with SQL

SQL queries summary:

The dataset was loaded as a table on the DB2 service of the IBM Cloud

Connecting to the DB2 with ibm_db and ibm_db_sa

We use SQLAchemy and sql magics to perform queries in the SQL native form and a prefix **%%sql**

In total 10 queries were made:

1) Display the names of the unique launch sites in the space mission
2) Display 5 records where launch sites begin with the string 'CCA'
3) Display the total payload mass carried by boosters launched by NASA (CRS)
4) Display average payload mass carried by booster version F9 v1.1
5) List the date when the first successful landing outcome in ground pad was acheived.
6) List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7) List the total number of successful and failure mission outcomes
8) List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
9) List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
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

GitHub URL of the completed EDA with SQL notebook

# Build an Interactive Map with Folium

Folium library allows visualizing Python data on Leaflet.js maps with the help of map objects such as markers, circles, lines, etc.

Launch sites were marked with circles and names.

For each launch site additional colour-coded markers were added to indicate successful (green) and failed (red) launches.

Distances from one of the launch sites to the nearest coastline, hightway, railroad and city were found and displayed on the line connecting the site with these landmarks.

GitHub URL of the completed interactive Folium maps notebook

# Build a Dashboard with Plotly Dash

We use Dash package to build and start locally an app with interactive Plotly.express plots, ”SpaceX Launch Records Dashboard”.

The launch sites can be chosen with a dropdown menu and results are presented in a pie chart of launch outcomes for a given site.

A dynamic slider allows to choose payload and display a relationship between a payload mass and a launch site with respective booster versions.

GitHub URL of the completed interactive Plotly Dash app notebook

# Predictive Analysis (Classification)

We explore scikit-learn and its 4 machine learning models, SVM, Classification Trees and Logistic Regression, by finding their best hyperparameters.

First, we standardize data with StandardScaler.

Then we split data to train and test sets with the size of the latter of 0.2 and a random state 2.

We then create  GridSearchCV object for each of the models with given parameters and compare scores.

We finally plot confusion matrices and find the best performing model wit Jacard, F1 and model scores.

GitHub URL of the completed predictive analysis notebook

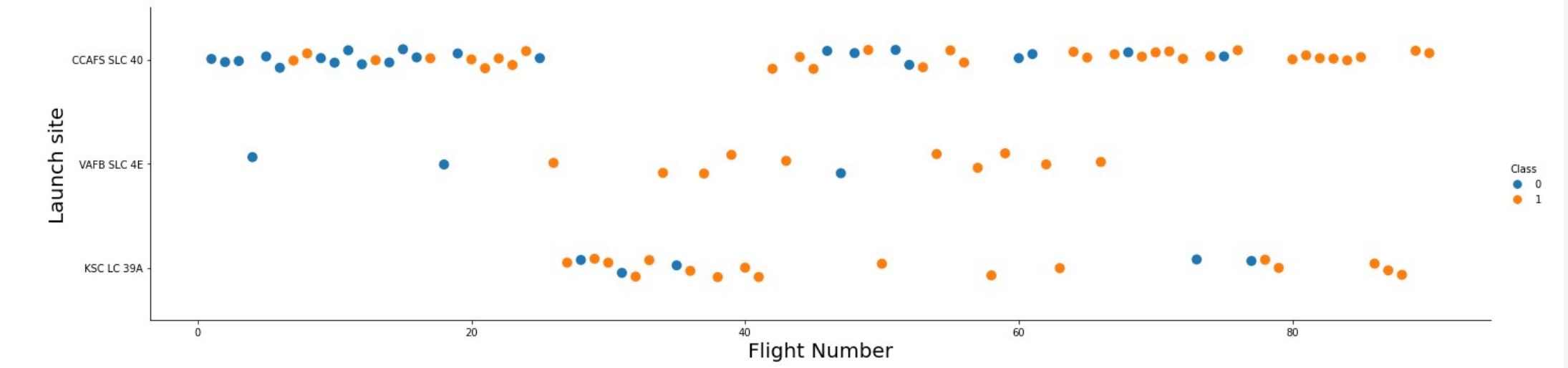
# Results

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

Section 2

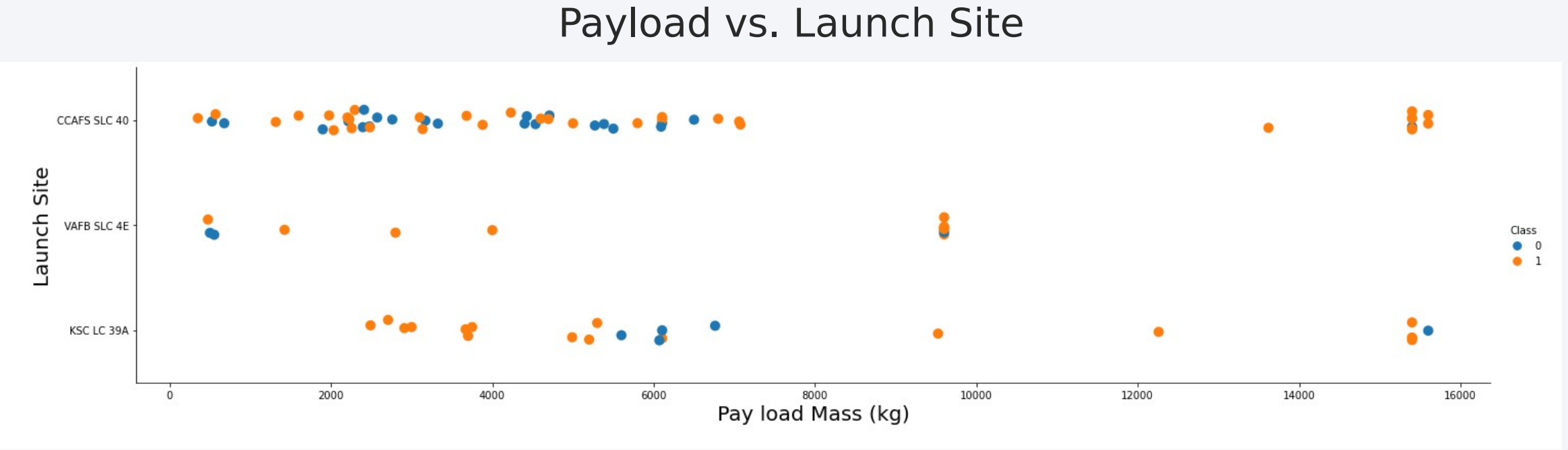
# Insights drawn from EDA

# Flight Number vs. Launch Site



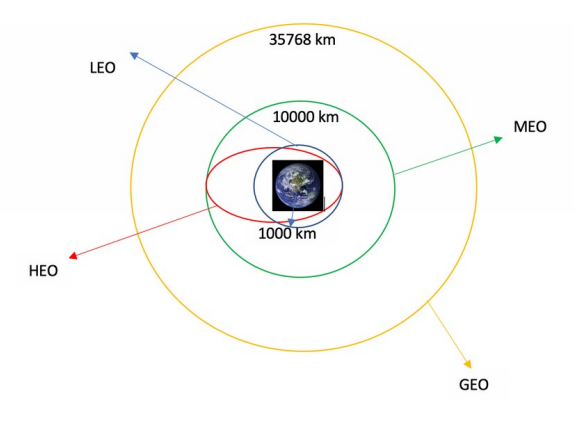

There are more successful flights with time which is reasonable as technology advances.

# Payload vs. Launch Site



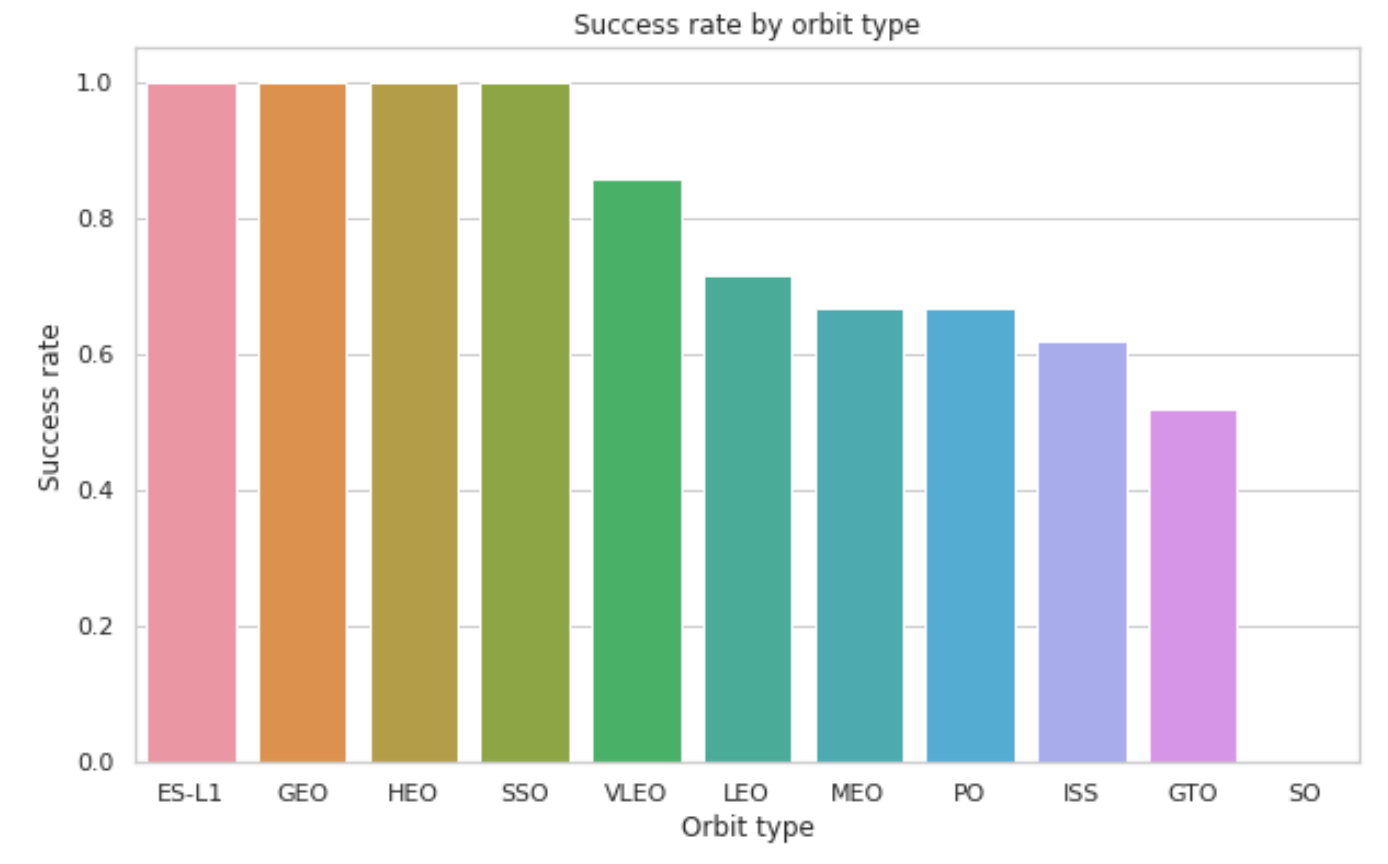

CCAFS SLC 40 rockets carry <8000 kg and very high loads ~16000 kg.
VAFB SLC 4E lanches rockets with payload <10000 kg.
KSC LC 39A rockets carry payload >2000 kg.

# Success Rate vs. Orbit Type

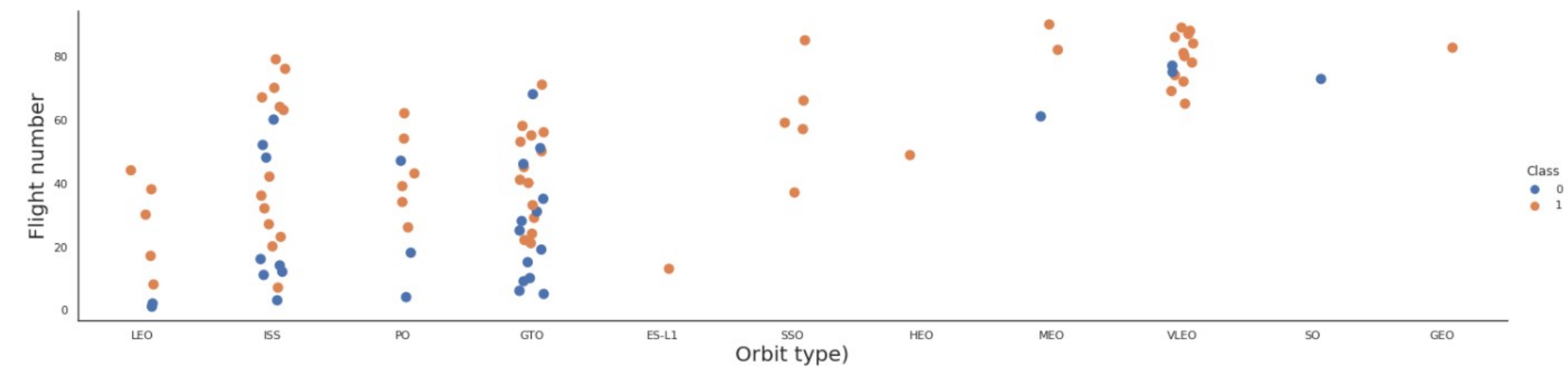



Launches to further orbits ES-L1 and GEO as well as HEO, SSO and a lower VLEO were highly successful.

- Launches to lower orbits (LEO, ISS, MEO) and further GTO and PO were successful in 50% to 60% of cases.

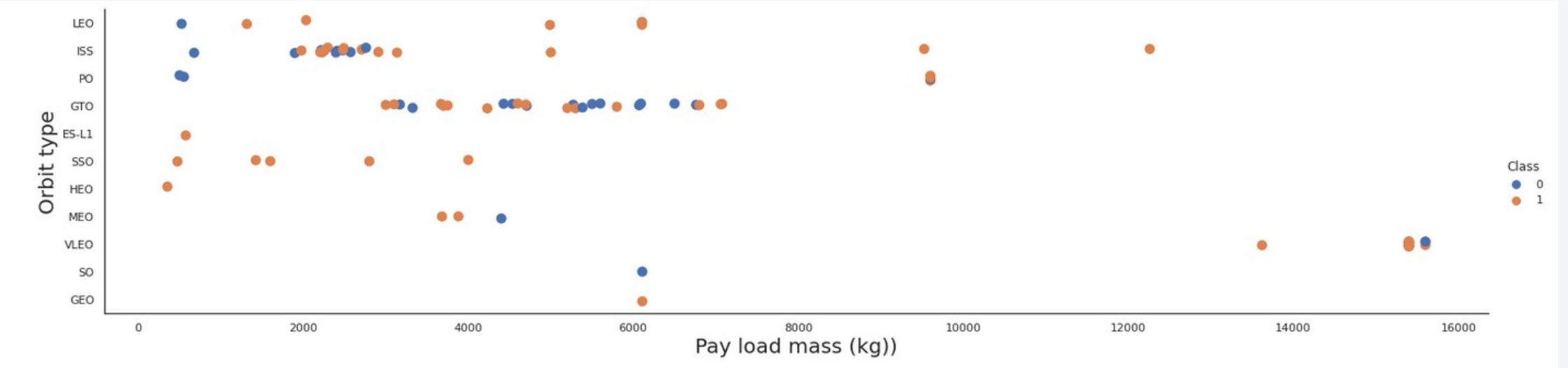


# Flight Number vs. Orbit Type




- The success rate in the first third of the flights was leaning towards failures. It has straightened to more successes in the last two thirds except for launches to GTO where success rate stayed around 0.6.

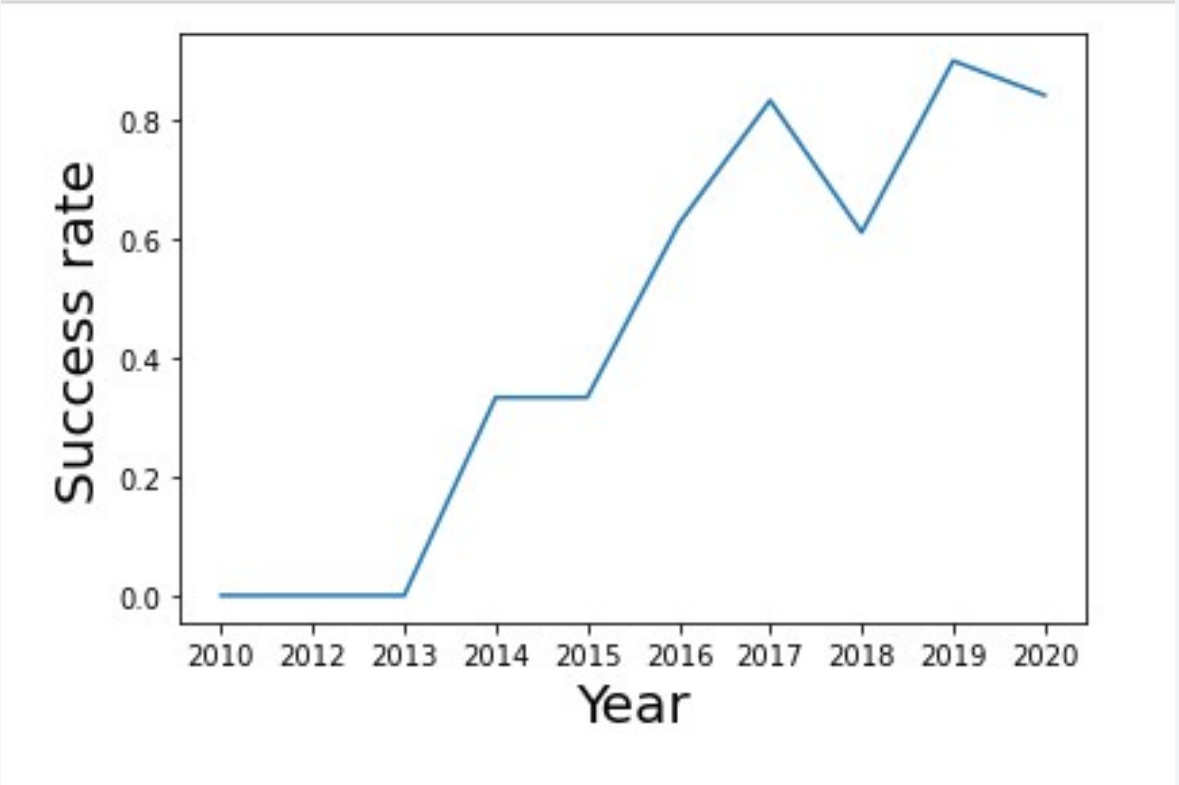
# Payload vs. Orbit Type




- Heavy payloads (>8000 kg) were successfully sent to ISS, PO and VLEO orbits.

# Launch Success Yearly Trend

- Starting 2015 with the first successful landing of Falcon 9 the success rate of launches has been growing.



# All Launch Site Names

```
%%sql
SELECT DISTINCT launch_site FROM SpaceX
```

```
 * ibm_db_sa://xsp69881:***@3883e7e4-18f5-
8/bludb
Done.
```

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

Selecting **DISTINCT** launch sites returns unique site names.

# Launch Site Names Begin with 'CCA'

```
%%sql
SELECT * FROM SpaceX
WHERE launch_site LIKE 'CCA%'
LIMIT 5
```

```
 * ibm_db_sa://xsp69881:***@1dd14d0c-1b52-4f63-a606-53ecba28771d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31498/bludb;security=SSL
Done.
```

9]:

| 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 |

To find records where launch sites begin with `CCA`, we use **LIKE** and a wildcard **%** for the rest of the name ‘CCA%’.

# Total Payload Mass

```
|: %%sql
   SELECT SUM(payload_mass__kg_) as total_payload_kg FROM SpaceX
   WHERE customer='NASA (CRS)'

    * ibm_db_sa://xsp69881:***@1dd14d0c-1b52-4f63-a606-53ecba287
   Done.

40]:  total_payload_kg
       45596
```

We calculate the total payload carried by boosters from NASA using **SUM** and setting the condition on a *customer* column

# Average Payload Mass by F9 v1.1

```
%%sql
SELECT AVG(payload_mass__kg_) as average_payload_kg FROM SpaceX
WHERE booster_version='F9 v1.1'
```

```
 * ibm_db_sa://xsp69881:***@1dd14d0c-1b52-4f63-a606-53ecba2877
Done.
```

| average_payload_kg |
|---|
| 2928 |

We calculate the average payload mass carried by booster version F9 v1.1 using **AVG** and setting the condition on the *booster_version* column

# First Successful Ground Landing Date

```
%%sql
SELECT MIN(DATE) as first_success FROM SpaceX
WHERE landing__outcome='Success (ground pad)'
```

```
 * ibm_db_sa://xsp69881:***@1dd14d0c-1b52-4f6
Done.
```

| first_success |
| --- |
| 2015-12-22 |

We can find the date of the first successful landing outcome on ground pad using **MIN** on the *date* column and setting the condition on the *landing_outcome* column.

# Successful Drone Ship Landing with Payload between 4000 and 6000

```
: %%sql
SELECT booster_version,payload_mass__kg_ FROM SpaceX
WHERE payload_mass__kg_>4000 AND payload_mass__kg_<6000
AND landing__outcome='Success (drone ship)'
```

```
 * ibm_db_sa://xsp69881:***@1dd14d0c-1b52-4f63-a606-53e
Done.
```

43]:

| booster_version | payload_mass__kg_ |
|---|---|
| F9 FT B1022 | 4696 |
| F9 FT B1026 | 4600 |
| F9 FT B1021.2 | 5300 |
| F9 FT B1031.2 | 5200 |

Here, we list the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 by setting three conditions joined by **AND** setting limits on the *payload_mass__kg* column and setting condition on the *landing_outcome*.

# Total Number of Successful and Failure Mission Outcomes

```
%%sql
SELECT COUNT(CASE WHEN mission_outcome LIKE '%Failure%' THEN 1 END ) as failure,
       COUNT(CASE WHEN mission_outcome LIKE '%Success%' THEN 1 END ) as success
FROM SpaceX
```

```
 * ibm_db_sa://xsp69881:***@1dd14d0c-1b52-4f63-a606-53ecba28771d.bs2io90l08kqb1od8l
498/bludb;security=SSL
Done.
```

| failure | success |
|---|---|
| 1 | 100 |

We calculate the total number of successful and failure mission outcomes by using **CASE WHEN x THEN 1 END** construct and naming the new columns appropriately.

# Boosters Carried Maximum Payload

```
%%sql
SELECT booster_version FROM SpaceX
WHERE payload_mass__kg_=(SELECT MAX(payload_mass__kg_) FROM SpaceX)
```

```
 * ibm_db_sa://xsp69881:***@1dd14d0c-1b52-4f63-a606-53ecba28771d.b
Done.
```

5]:

| 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 |

We list the names of the booster which have carried the maximum payload mass by making a nested query with **MAX** of the *payload_mass__kg.*

# 2015 Launch Records

```
%%sql
SELECT booster_version,launch_site,landing__outcome FROM SpaceX
WHERE landing__outcome='Failure (drone ship)' AND YEAR(DATE)=2015
```

```
   ibm_db_sa://xsp69881:***@1dd14d0c-1b52-4f63-a606-53ecba28771d.bs2
 * ibm_db_sa://xsp69881:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2
Done.
```

2]:

| booster_version | launch_site | landing__outcome |
|---|---|---|
| F9 v1.1 B1012 | CCAFS LC-40 | Failure (drone ship) |
| F9 v1.1 B1015 | CCAFS LC-40 | Failure (drone ship) |

We list the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015 by setting a condition on the *landing_outcome* and selecting only **YEAR** on the date column to equal 2015.

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

```
7]: %%sql
SELECT landing__outcome, COUNT(landing__outcome) as outcome_type_count FROM SpaceX
WHERE DATE BETWEEN '2010-06-04' and '2017-03-20'
Group BY landing__outcome
ORDER BY outcome_type_count DESC
```

```
 * ibm_db_sa://xsp69881:***@1dd14d0c-1b52-4f63-a606-53ecba28771d.bs2io90l08kqb1od8lcg
Done.
```

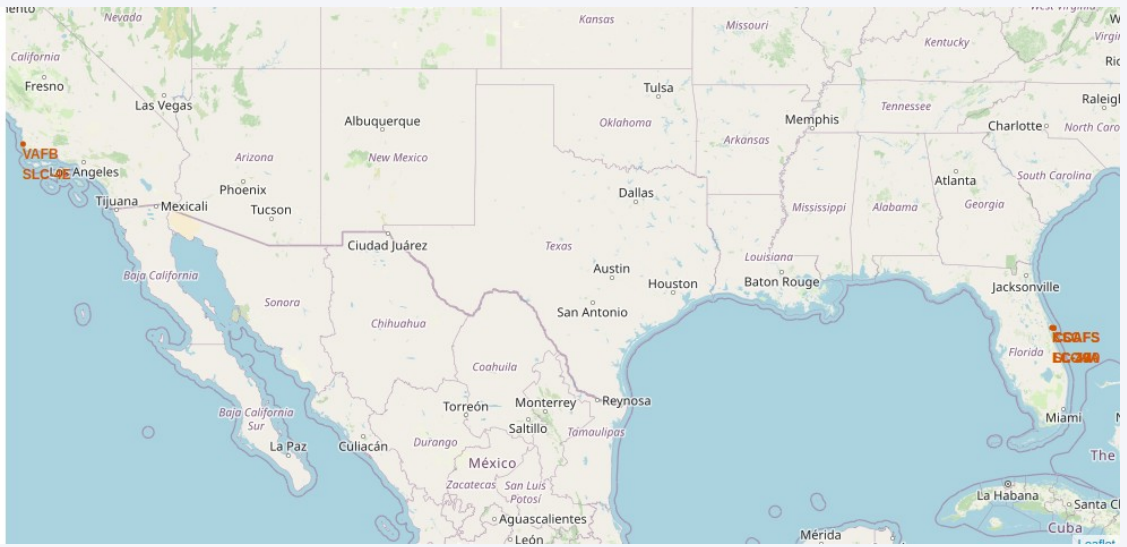
:[47]:

| landing__outcome | outcome_type_count |
|---|---|
| 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 |

We can 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 by setting a condition on the date **BETWEEN** and using **GROUP BY** and **ORDER BY** in **DESC** order.

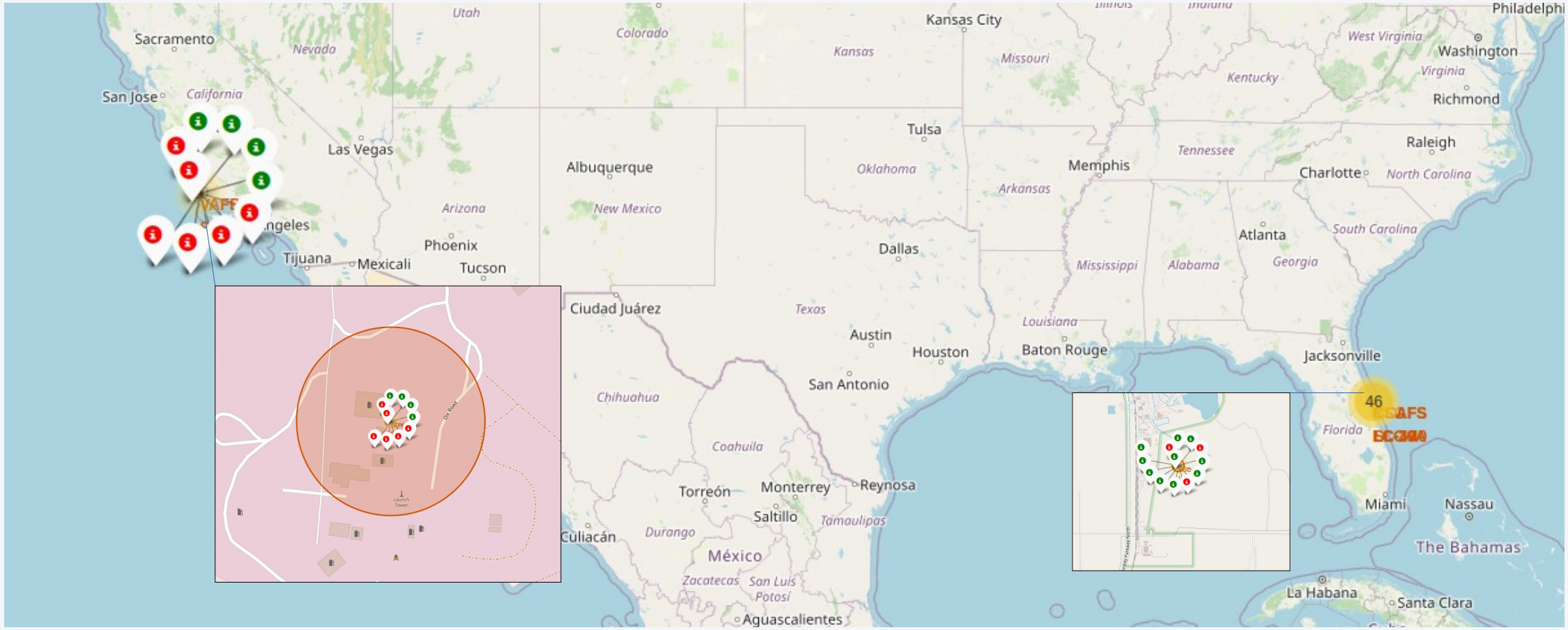

Section 3
Launch Sites
Proximities Analysis

# Map of all SpaceX launch sites

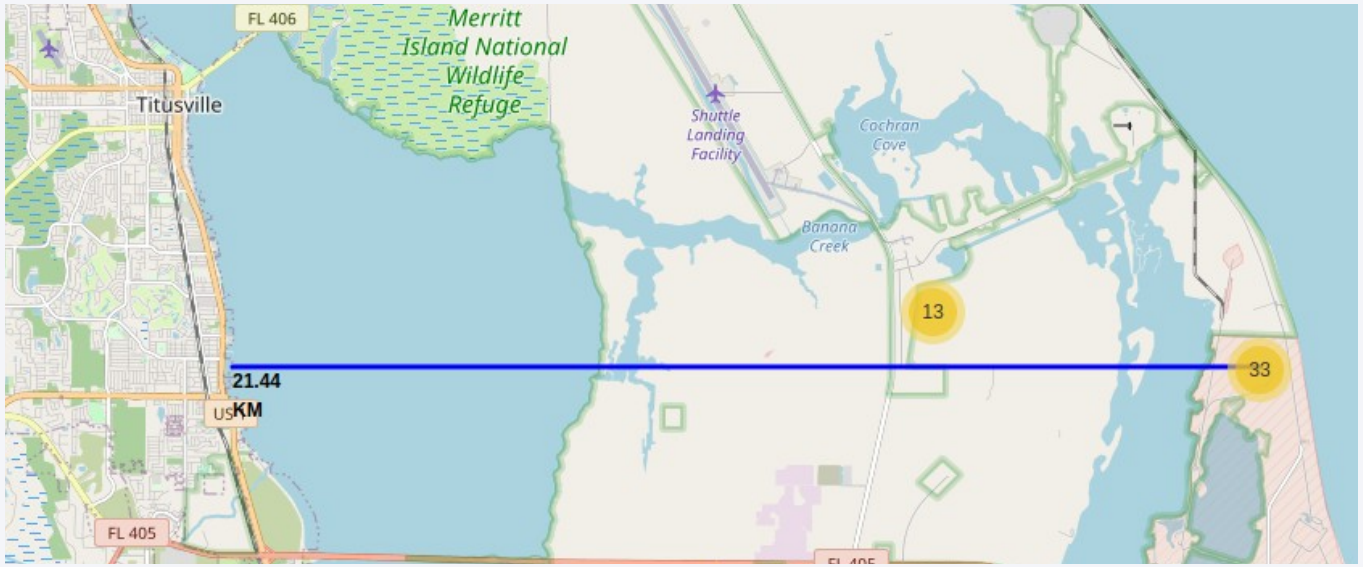

Folium map with location markers of all launch sites of SpaceX rockets. The sites are located as close as possible to the equator to facilitate liftoff and further away from inhabited places to minimize collateral damage.

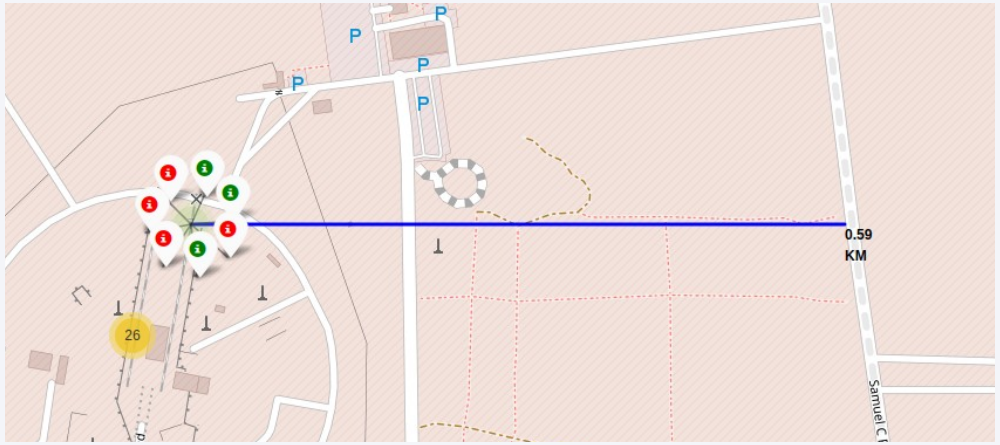
# Colour-labelled launch outcomes



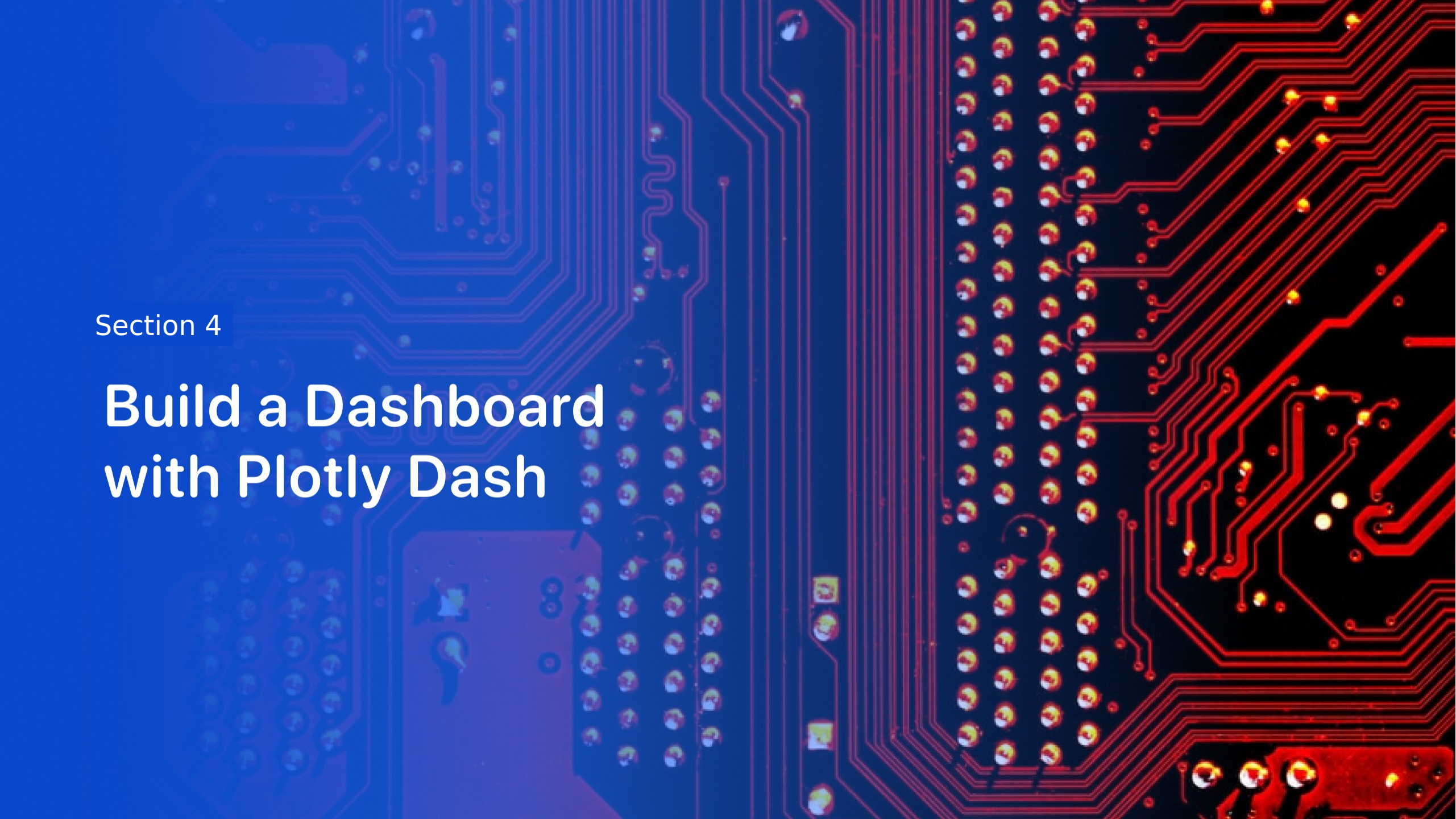

Interactive map visualization of launch sites and their success and failure outcomes gives a better and a faster overview in the geospatial context.

# CCAFS CLC-40 launch site to its proximities

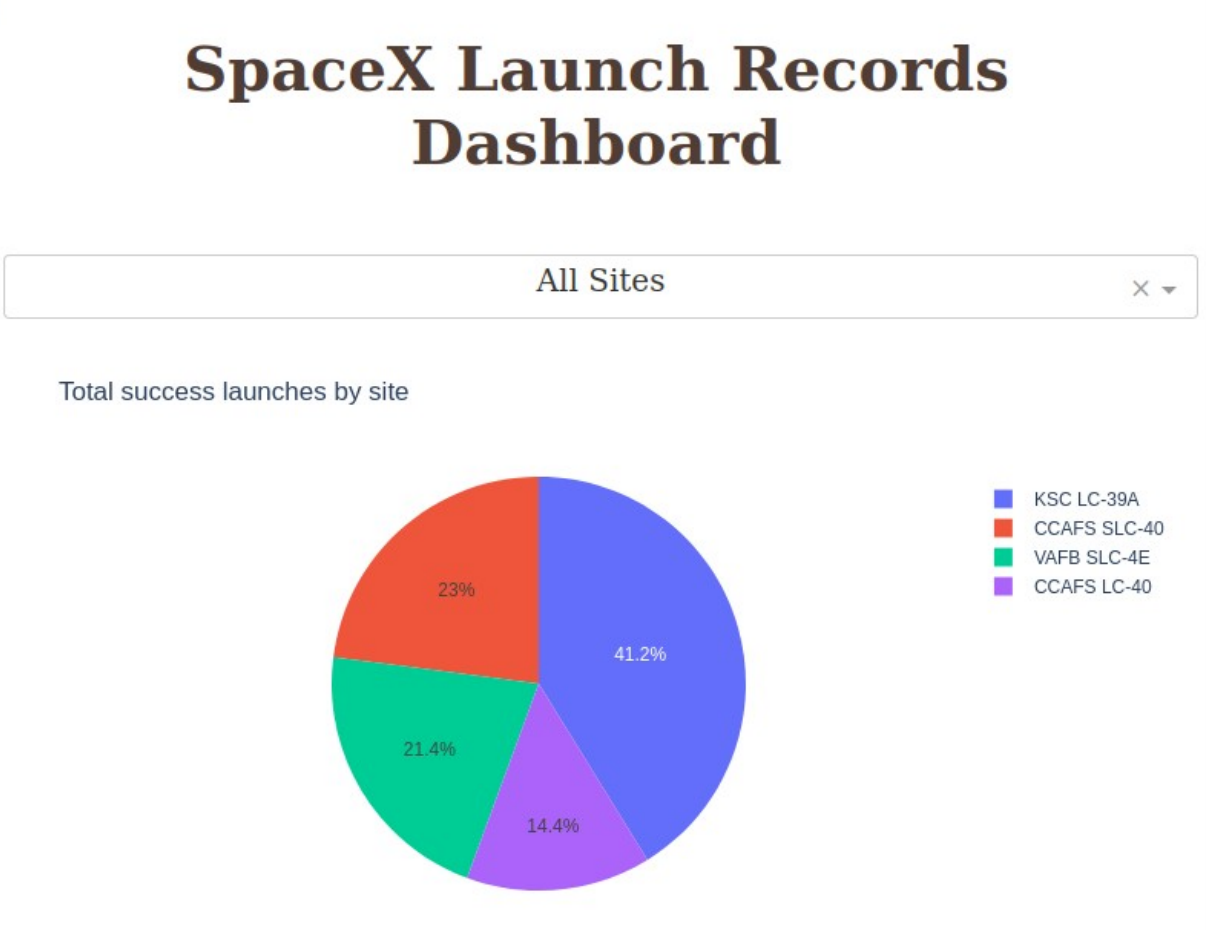




The launch sites should be away from settlements to avoid collateral damage from falling parts, thus proximity to the coastline. However, transport infrastructure should be easily accessible to supply necessary equipment and transport staff.

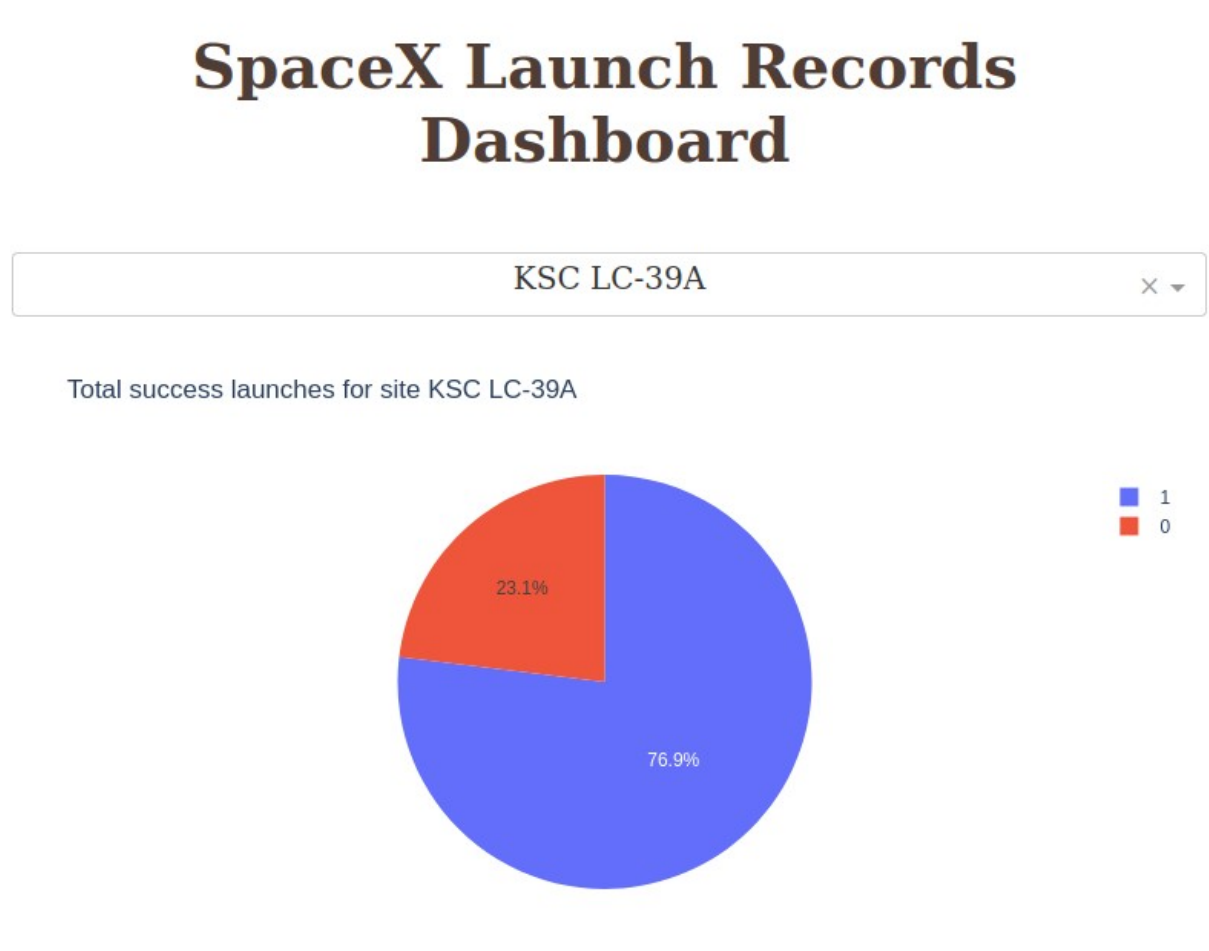

Section 4
Build a Dashboard with Plotly Dash

# Total success launches by site



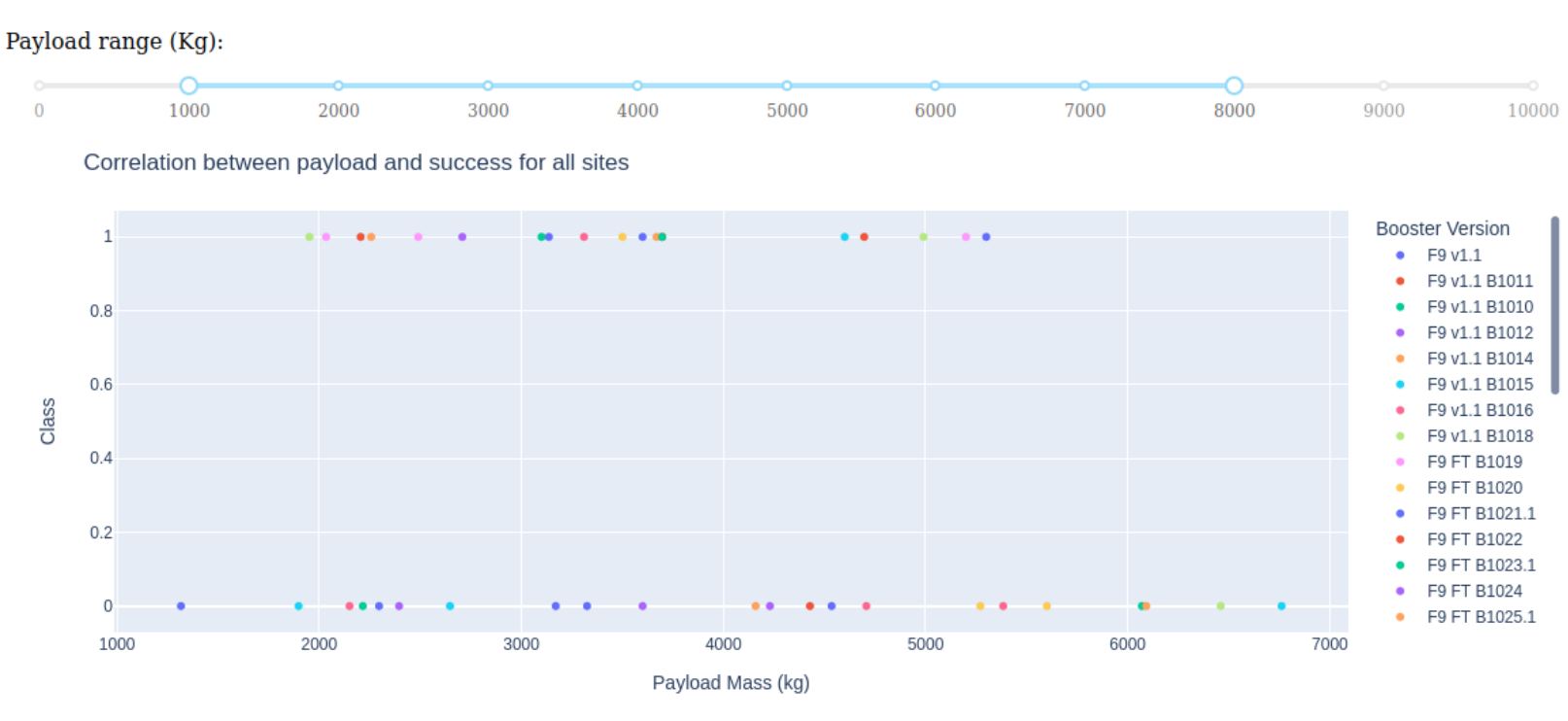

The highest number of successful launches is observed at the John F. Kennedy Space Center, KSC LC-39A site, Florida.

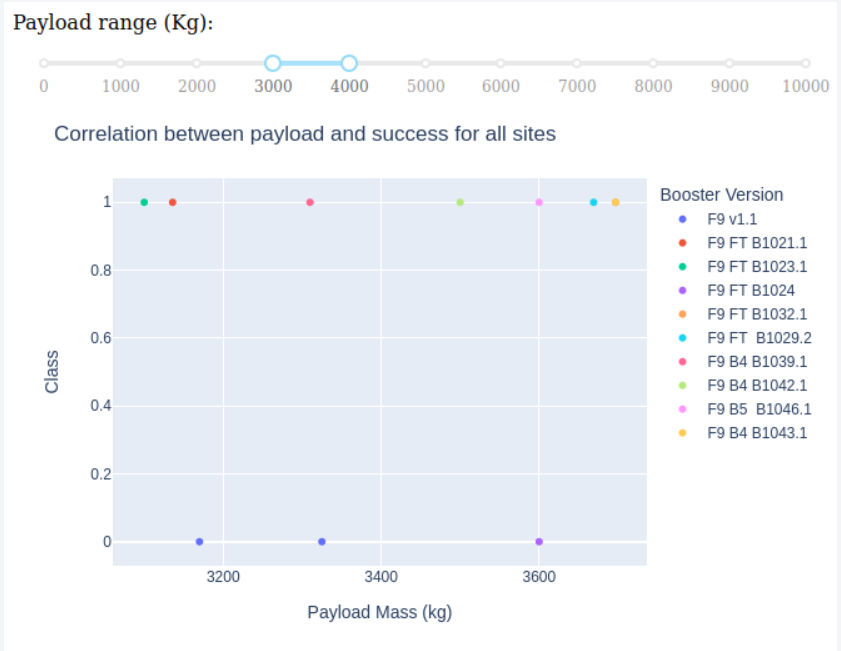
# Site with the highest launch success ratio




Out of 13 launches there have been 10 successful launches or 76.9%.

# Correlation between payload and success rate for all sites




Flights with payload masses between 3000 kg and 4000 kg have a highest success rate for various booster versions as well as around 5000 kg. Flights with payloads higher than 5300 kg were all unsuccessful. The ratio of success/failure for payloads between 2000 kg and 3000 kg is equal.

Section 5

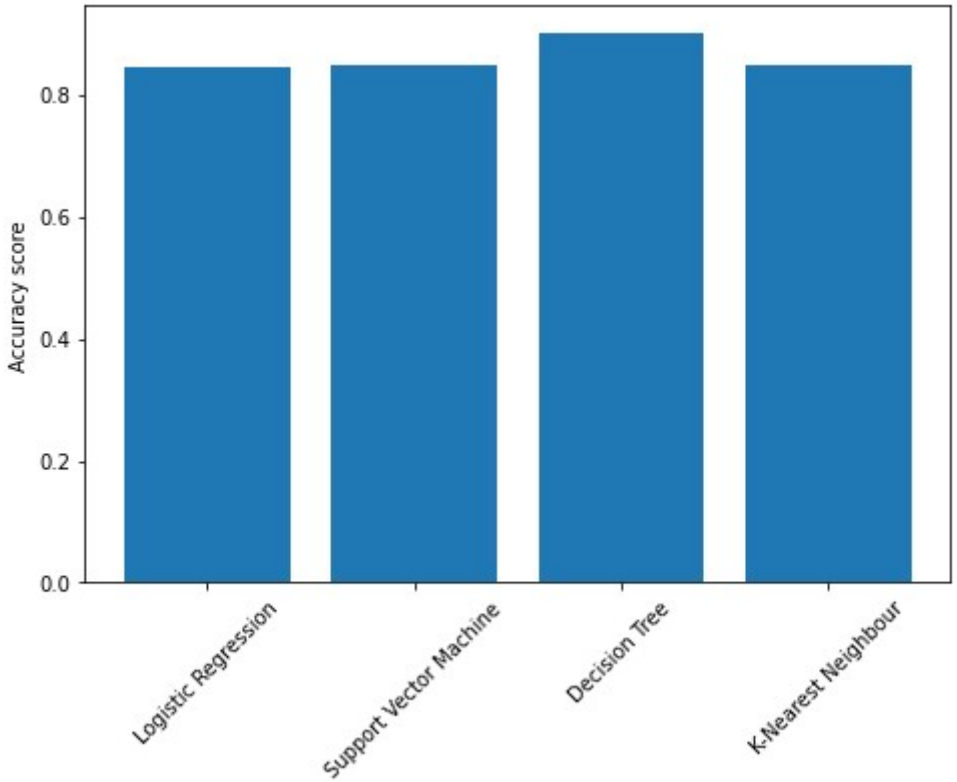
# Predictive Analysis (Classification)

# Classification Accuracy

| | Algorithm | Jaccard | F1-score | Best params |
|---|---|---|---|---|
| 0 | Logistic Regression | 0.800000 | 0.814815 | 0.846429 |
| 1 | Support Vector Machine | 0.800000 | 0.814815 | 0.848214 |
| 2 | Decision Tree | 0.923077 | 0.943030 | 0.876786 |
| 3 | K-Nearest Neighbour | 0.800000 | 0.814815 | 0.848214 |

```
best_score=max([x.best_score_ for x in models.values()])
for key,value in models.items():
    if best_score==value.best_score_:
        print('The best performing model is', key, 'with parameters:
```

```
The best performing model is Decision Tree with parameters: {'criter
ion': 'entropy', 'max_depth': 4, 'max_features': 'auto', 'min_sample
s_leaf': 2, 'min_samples_split': 2, 'splitter': 'best'}
```

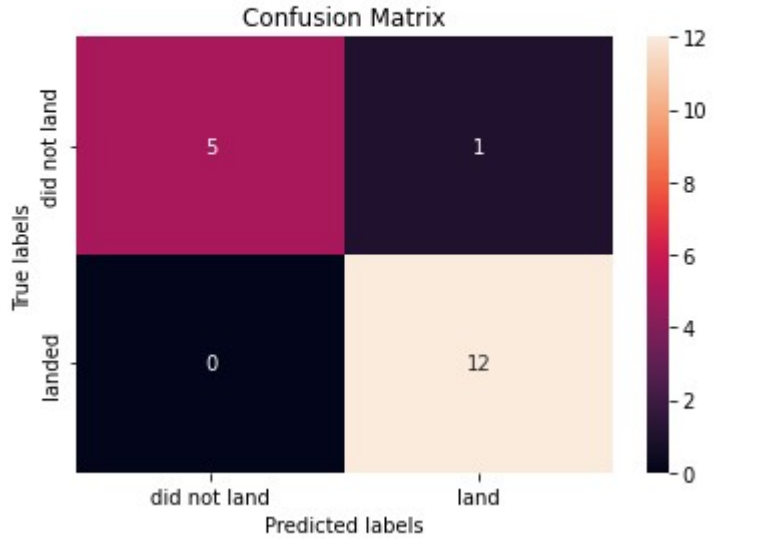



- DecisionTree has the best accuracy and is the best performing model in this case.

# Confusion Matrix

- DecisionTree model showed a better performance and scores. The confusion matrix shows correctly predicted true positive (12) and true negative (5) outcomes and only one false negative predicted outcome.

```
]: tree_yhat = tree_cv.predict(X_test)
   plot_confusion_matrix(Y_test,tree_yhat)
```



# Conclusions

The most successful launch site has so far been KSC LC-39A, Florida.

Payload mass does not play a significant in launch outcomes (KSC LC-39A launches a wide range of payloads)

Most successful launches were to ES-L1, GEO, HEO, SSO, and VLEO orbits.

22 December 2015 was the first successful landing of the Falcon 9 booster.

Since then number of successful launches skyrocketed.

The best performing ML model for predicting launch outcome is DecisionTree with accuracy score of 0.877.

# Appendix

Since github does not support  Leaflet.js map visualization (something to do with javascript security issues), one can use https://nbviewer.org to see notebook with maps.

All datasets are included in the repository here

In some of the notebooks, I used my own functions or methods to perform a task. For example, using list comprehensions was faster than proposed fnctions (in this case twice as slow) :

```
%timeit spacex_df['marker_color']=['red' if x==0 else 'green' for x in spacex_df['class']]
spacex_df.tail(10)
```

```
102 µs ± 4.36 µs per loop (mean ± std. dev. of 7 runs, 10000 loops each)
```

# Thank you!