IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Liliana Adriana Mendoza
Agosto 26

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive analysis (Classification)
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis result

Introduction

- Project background and context

SpaceX is the only private company ever to return a spacecraft from low-earth orbit, which it first accomplished in December 2010.

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

- Problems you want to find answers

The difference in cost is due to the recovery and reuse of the first stage of the rocket.

This is why we want to **predict if the Falcon 9 first stage will land successfully**

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Request to the SpaceX API
    - We use SpaceX rocket launch data contained in the URL: https://api.spacexdata.com/v4/launches/past
    - We will define a series of helper functions that helped us use an API to extract technical information related to the rockets, payloads, launch pad and cores
    - Using the **GET request** we decode the response content as a Json using .json() and turn it into a Pandas dataframe using .json_normalize()

Methodology

Executive Summary

- Data collection methodology:
  - Request to the SpaceX API
    - Using **BeautifulSoup()** we collect Falcon 9 historical launch records from a Wikipedia page titled "List of Falcon 9 and Falcon Heavy launches" contained in the URL:
    - https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922#Past_launches
    - We create a Pandas data frame with keys from the extracted column names of the launch HTML tables.

Methodology

Executive Summary

- Data collection methodology:
  - Clean the requested data
    - Clean the requested data
    - These missing values of the LandingPad column were retain None values to represent when landing pads were not used.
    - To case the PayloadMass, we calculate the mean for the PayloadMass using to replace the missing value in the data. We use functions **.mean() .replace() and np.nan**

Methodology

Executive Summary

- Coordinates of the launches' infrastructure

  Dataset with latitude and longitude added for each site

  - From URL:

  https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_geo.csv'

Methodology

Executive Summary

- Perform data wrangling
  - Exploratory Data Analysis
    - We identify and calculate the percentage of the missing values in each attribute.
    - We calculate the number of launches on each site, it means in

      1Launch Complex 40 VAFB SLC 4E ,

      2 -Vandenberg Air Force Base Space Launch Complex 4E (SLC-4E),

      3-Kennedy Space Center Launch Complex 39A KSC LC 39A
    - We calculate the number and occurrence of mission outcome of the orbits in order to create a landing outcome label from Outcome column.
  - Determine Training Labels
    - We represent the classification variable as the outcome of each launch. If the value is zero, the first stage did not land successfully, then one means the first stage landed Successfully.

Methodology

Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL
    - We work by database to understand the Spacex DataSet using function **sqlite3()**
    - We connect to the database by **%sql sqlite:///my_data1.db**
    - We load the dataset into the corresponding table in a Db2 database
    - We execute SQL queries by %sql to explorer DataSet by **%sql**

Methodology

Executive Summary

- Perform interactive visual analytics using Folium and Plotly Dash
  - We use the library and function folium.Map() to work on map interactives.
  - We create and mark all launch sites on a map by funtions folium.Circle() , folium.map.Marker() and site_map.add_child()
  - We mark the success/failed launches for each site on the mapusing site's latitude and longitude coordinates and launch outcome.
  - We calculate the distances between a launch site to its proximities as railway, highway and coastline using finction MousePosition() and buildind a funtion to calculate distances.

Methodology

Executive Summary

- Perform predictive analysis using classification models
  - We create a column for the class from URL : "https://api.spacexdata.com/v4/launches/past"
  - We standardize the data by library an function **preprocessing.StandardScaler()**
  - We plit into training data and test data by **train_test_split()**
  - We build the logistic regression, SVC, decision Tree, and KNN algorithms using **LogisticRegression(), SVC(), DecisionTreeClassifier() and KNeighborsClassifier()** function, respectively. and train them by **fit()** function.
  - we got the accuracy on the validation data using the data attribute **best_score_** to select find bestest hyperparameters. And,
  - We select the algorithm with best accuracy by **.score()** function .

Data Collection

- The data sets were collected
  - Booster Version, Payload mass, Orbit, Launch site, Outcome , Flights, Grid fins, Reused, Legs, Landing pad, Block, Reused count, Serial, Longitude, Latitude,
  - The outcome of the landing, the type of the landing, number of flights with that core, whether gridfins were used, whether the core is reused, whether legs were used, the landing pad used, the block of the core which is a number used to separate version of cores, the number of times this specific core has been reused, and the serial of the core.
  - Finally, we will keep only the Falcon 9 launches, by filtering the data dataframe using the 'BoosterVersion' column.

Data Collection – SpaceX API

The GitHub URL of the completed SpaceX API calls notebook purpose.

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:

```
[17]: 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

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

```
[18]: # Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url)
response.status_code
```

```
[18]: 200
```

```
[19]: data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

```
[20]: # Get the head of the dataframe
#data.head(5)
```

```
[21]: data.shape
```

```
[21]: (107, 42)
```

SpaceX REST API → API Returns SpaceX in .Json → Normalize data json_normalize()

Data Collection Scraping

The GitHub URL of the completed web scraping notebook.

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

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

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response, '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('tr')
```

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:

```
<tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time"
title="Coordinated Universal Time">UTC</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of
Falcon 9 first-stage boosters">Version,<br/>Booster</a> <sup class="reference"
```

```
            launch_dict["Version Booster"].append(bv)
-------------
            # Launch Site
            # Append the bv into launch_dict with key `Launch Site`
            launch_site = row[2].a.string
            launch_dict["Launch site"].append(launch_site)
            #print(launch_site)
-------------
            # Payload
            # Append the payload into launch_dict with key `Payload`
            payload = row[3].a.string
            launch_dict["Payload"].append(payload)
            #print(payload)
-------------
            # Payload Mass
            # Append the payload_mass into launch_dict with key `Payload mass`
            payload_mass = get_mass(row[4])
            launch_dict["Payload mass"].append(payload_mass)
            #print(payload)
-------------
            # Orbit
            # Append the orbit into launch_dict with key `Orbit`
            orbit = row[5].a.string
            launch_dict["Orbit"].append(orbit)
            #print(orbit)
-------------
            # Customer
            # Append the customer into launch_dict with key `Customer`
            customer = row[6]
            launch_dict["Customer"].append(customer)
            #print(customer)
-------------
            # Launch outcome
            # Append the launch_outcome into launch_dict with key `Launch outcome`
            launch_outcome = list(row[7].strings)[0]
            launch_dict["Launch outcome"].append(launch_outcome)
            #print(launch_outcome)
-------------
            # Booster landing
            # Append the launch_outcome into launch_dict with key `Booster landing`
            booster_landing = landing_status(row[8])
            launch_dict["Booster landing"].append(booster_landing)
            #print(booster_landing)
-------------
```

After you have fill in the parsed launch record values into `launch_dict`, you can create a dataframe from it.

```
df=pd.DataFrame(launch_dict)
df.head(5)
```

	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], \n]	Success\n	F9 v1.07B0003.18	Failure	4 June 2010	18:45

Ger HTML from Wikipedia by requests.get()

Extract data by BeautifulSoup()

Selection data by .find_all('tr')

Create a dicctionary to set up data frame

Data Wrangling

Identify and calculate the percentage of the missing values → Identify which columns are numerical and categorical → Identify the number and occurrence of mission outcome of the orbits, launches on each site, and orbit → Create a landing outcome

The GitHub URL of the completed data wrangling related notebooks.

EDA with Data Visualization

- Scatter point plot to depict the relationship between Flight number and Launch site
- Scatter plot to depict the relationship between Payload and. Launch site
- Scatter plot to depict the relationship between Payload and Launch site
- Scatter point to depict the relationship between flight number and orbit type
- Scatter point to depict the relationship between payload and orbit type
- The line chart of yearly average success rate to see success rate evolution in the time.

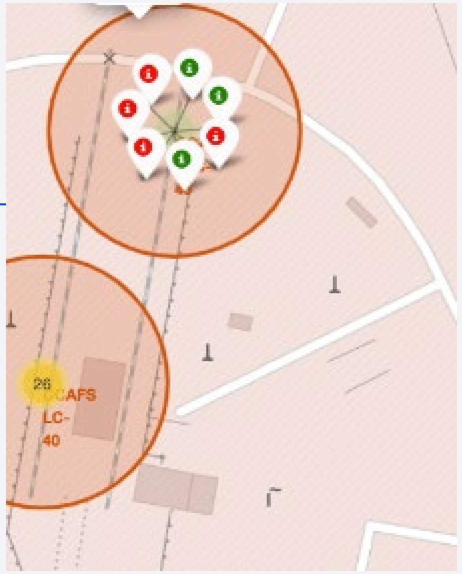
The GitHub URL of the completed EDA with data visualization notebook

EDA with SQL

- Records where launch sites begin with the string 'CCA'
- the total payload mass carried by boosters launched by NASA (CRS)
- Average payload mass carried by booster version F9 v1.1
- Sate when the first succesful landing outcome in ground pad was acheived.
- Names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- Total number of successful and failure mission outcomes
- Names of the booster_versions which have carried the maximum payload mass.
- Records which display the month names, failure landing_outcomes in drone ship , booster versions, launch_site for the months in year 2015.
- 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 ordered.

The [GitHub URL](#) of the completed EDA with SQL notebook

Interactive Map with Folium



- We add map objects such as markers, circles and lines to illustrate strategic procession launches sites.
  - Add each launch site's location on a map using site's latitude and longitude coordinates, sites as : CCAFS SLC-40 KSC LC-39A and VAFB SLC-4E.
  - Add each launch site's location their popup label on a map for showing its name.
  - Add each launch sites circle with one km to radius on a map for underlining its localization.
  - Add white mark the success and failed launches for each site on the map to stand out the outcome of the each site. green to success and red to failed.
  - Add blues lines and red label to illustrate the distances between a launch site to its proximities such as: railway, highway, and coastline.

The GitHub URL of the completed interactive map with Folium map.

Dashboard with Plotly Dash

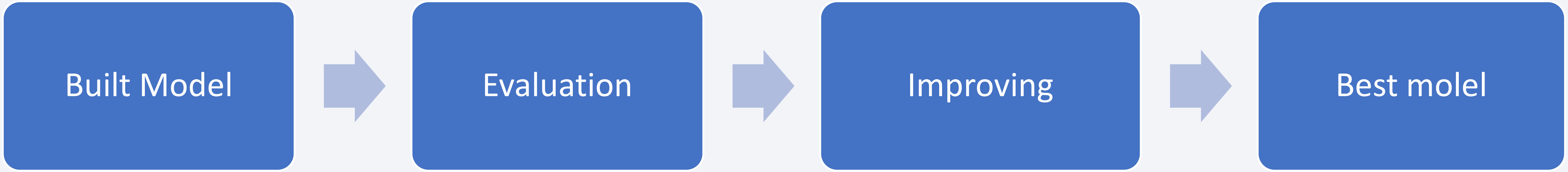
We add a dropdown to choose to Launch site and plot the following plots:

- Scatter plot to illustrate the correlation Between Payload and launch site, or to total aggregated.
- Pie plot with the total Success Launches and launch site, or to total aggregated.

We add a dropdown to limit the Payload mass in the scatter plot with the correlation Between Payload and launch site.

The GitHub URL of the completed Plotly Dash lab.

Predictive Analysis (Classification)



- To built, we use a data set, that standardize of numerical data and dummise categorical data.
- Split data set in test an train sample wit a proportion 20-80, respectively.
- To evaluate, we analyze the score and confusion matrix
- To improve, we use algorithms tuning by GridSearcheCV function.
- To found the best performing classification model, we test a combination to parameters in each algorithms.

The GitHub URL of the completed predictive analysis lab.

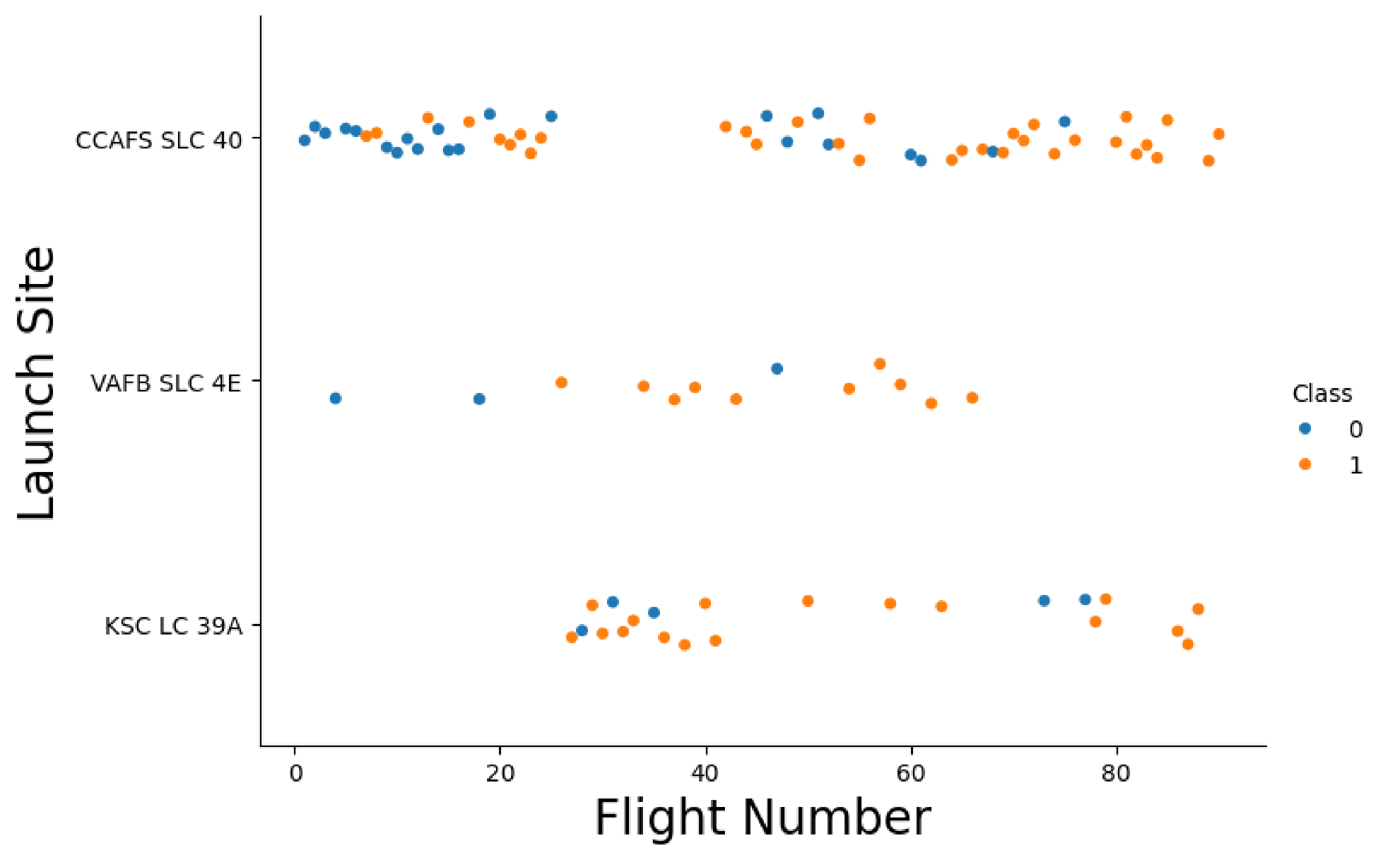
Next, the results

- Exploratory data analysis results (EDA)
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

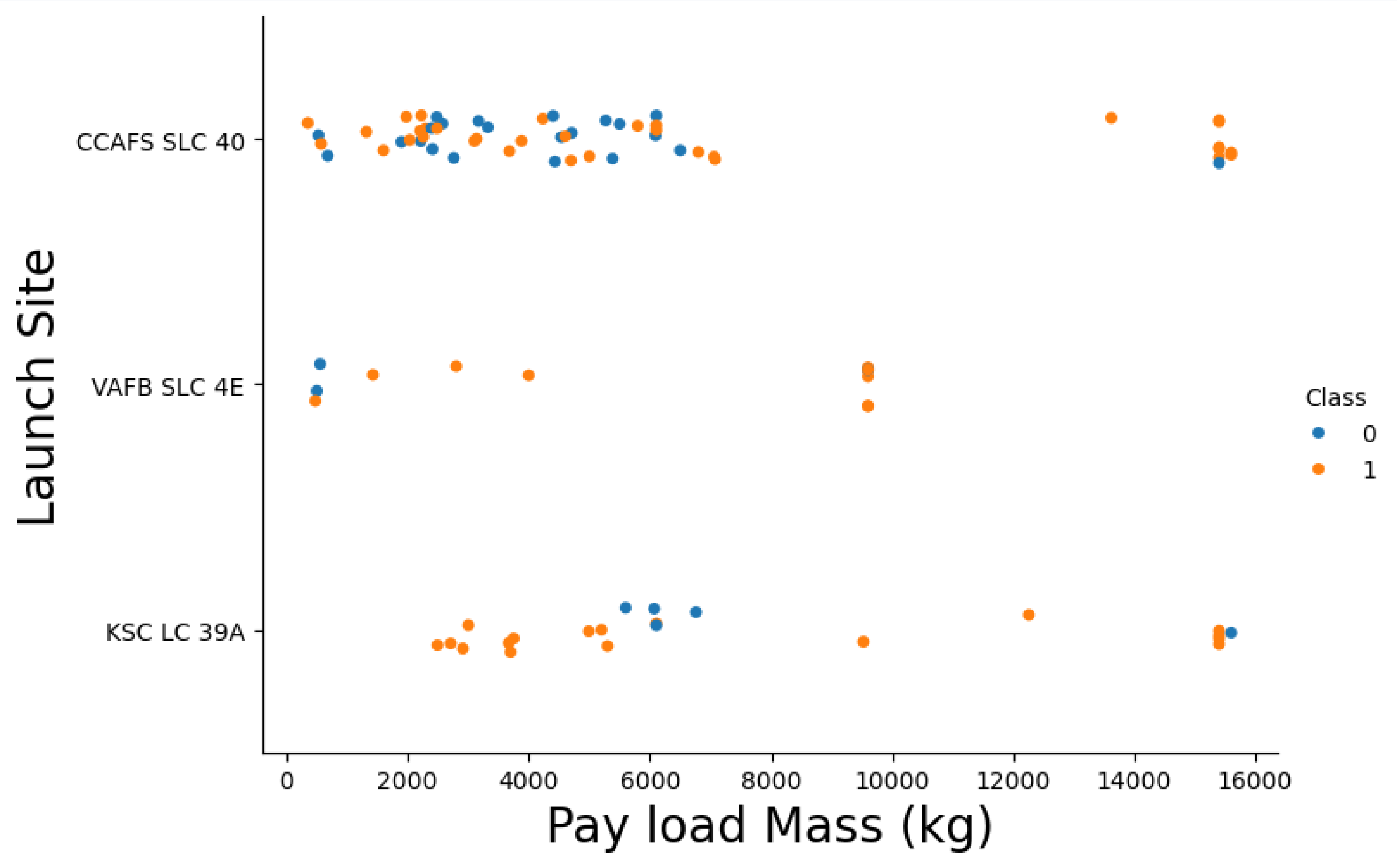
Insights drawn from EDA

Flight Number vs. Launch Site



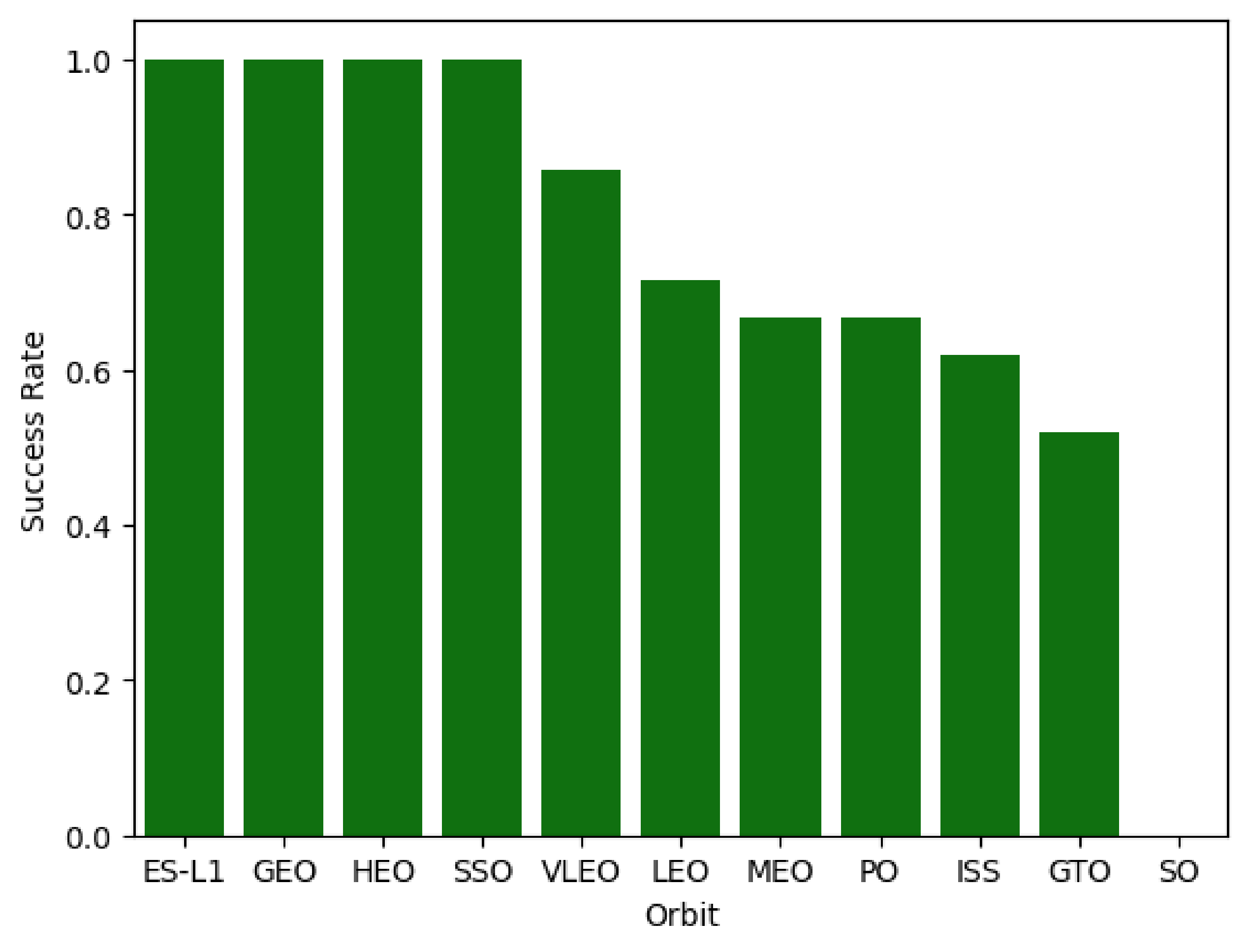
The launch site have different proportions in the flight number. The CCAFS SLC-40 has the most flight number and WFB SLC 4E has the lest.

Payload vs. Launch Site



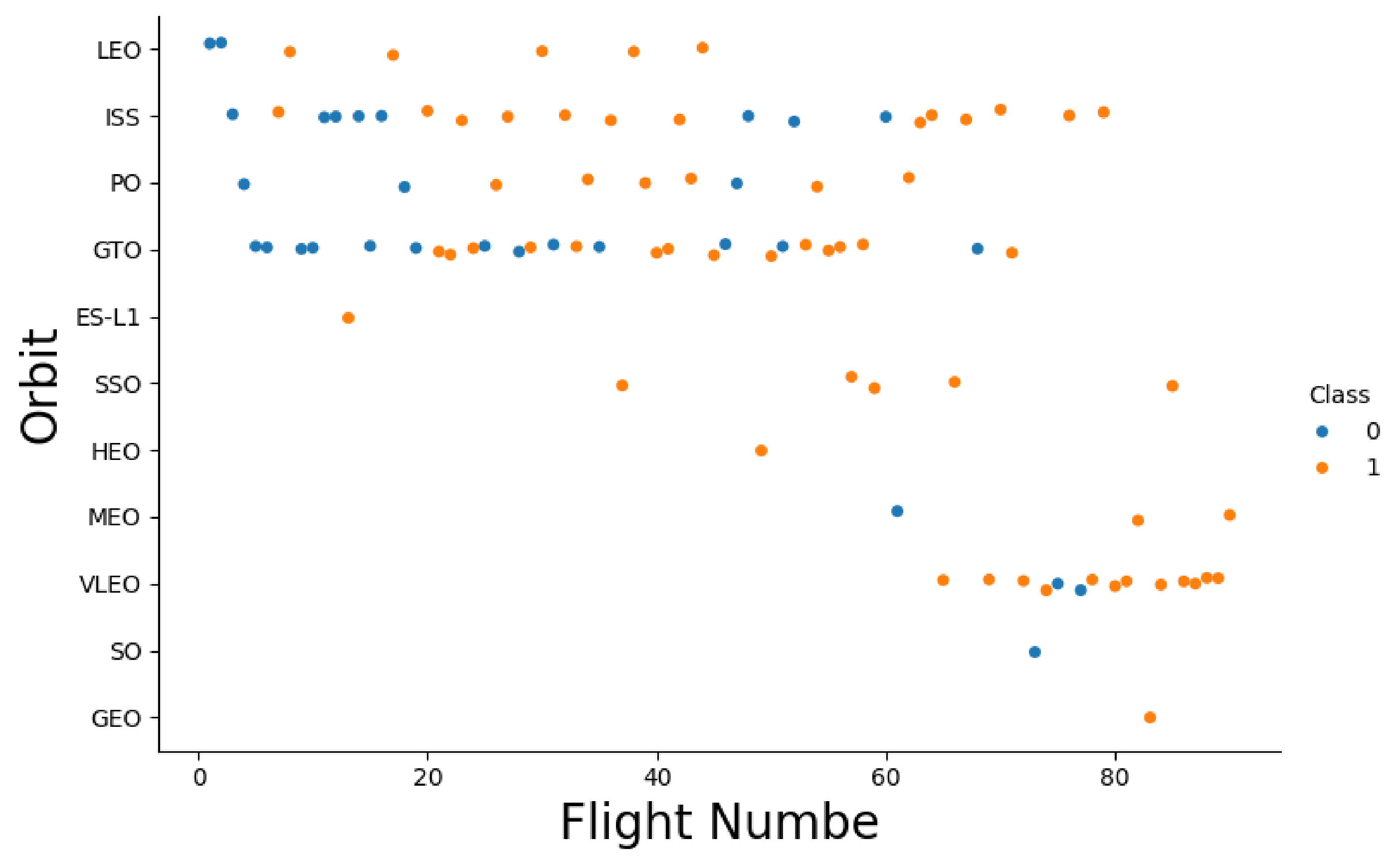
The probability of success increases with increasing payload mass kg.

Success Rate vs. Orbit Type



- Orbits have more than 50% success rate. For example: ES-L1, GEO, HEO and SSO orbits have had 100% successful landings.

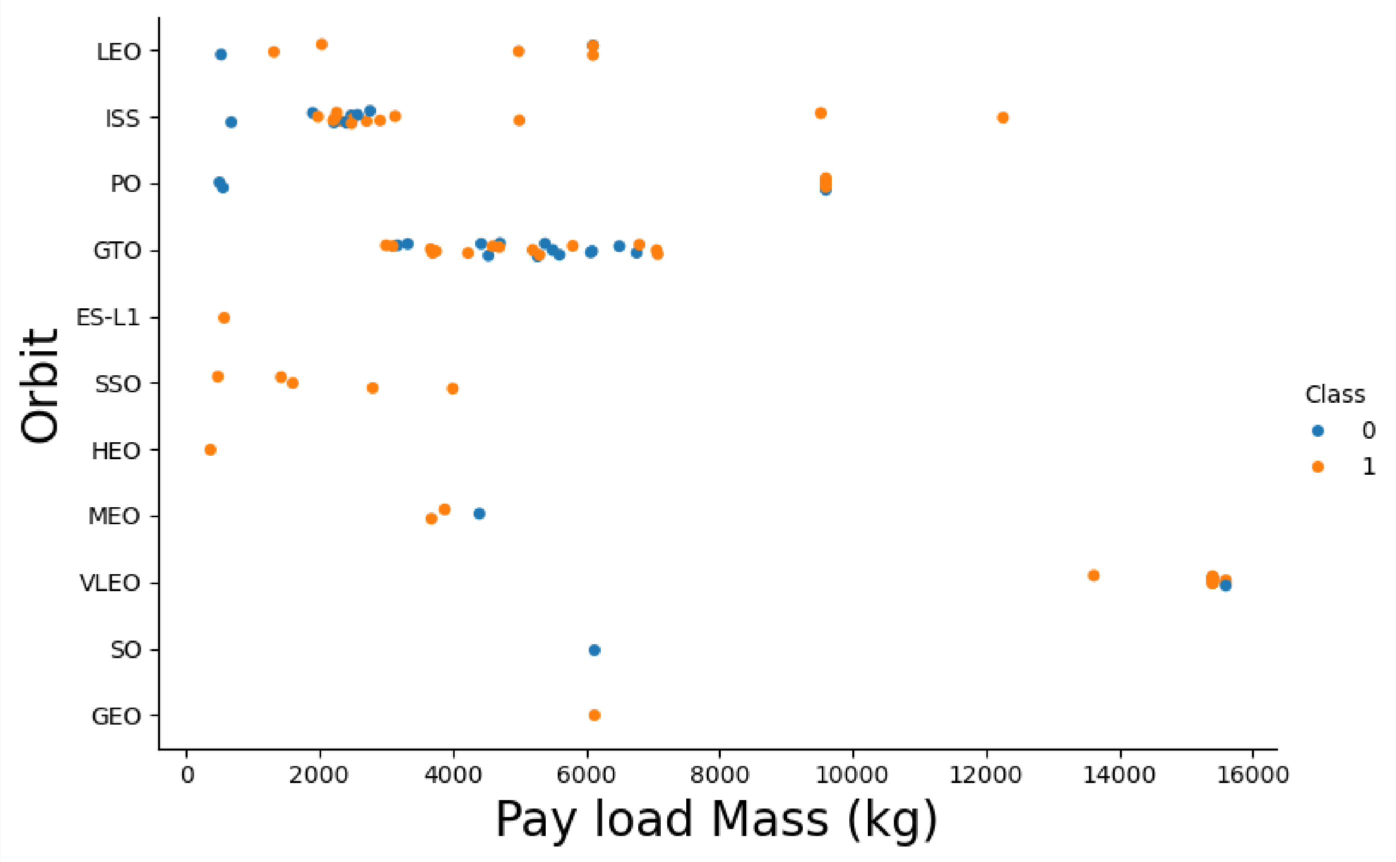
Flight Number vs. Orbit Type



Orbits with fewer flights show a higher success rate. The VLEO orbit has been operating since after flight 60 and has the highest flight traffic and the highest success rate.

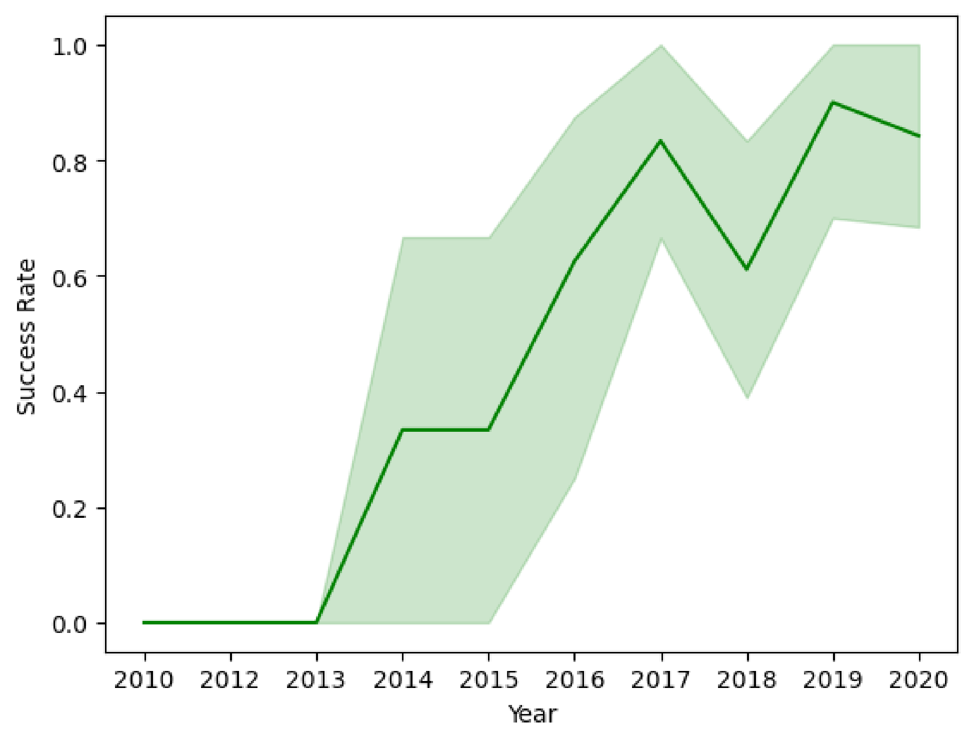
Over time, orbits with higher success rates show improvements in success rates.

Payload vs. Orbit Type



In most orbits, flights have weights of less than 8,000 kg. The VLEO orbit operates with flights of more than 14,000 kg.

Launch Success Yearly Trend



Since 2016, the probability of success has been higher than that of failure. The evolution of success in landings is growing every year, by 2020 the success rate was close to 0.8. That is, 8 out of 10 landings were successful. Except for 2015, when the rate dropped to 6 out of 10.

All Launch Site Names

- There are four names of the unique launch sites

```
Select
distinct(Launch_Site
) from  SPACEXTABLE
```

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

Launch Site Names Begin with 'CCA'

The Launch Site Names Begin with 'CCA' is CCAFS LC-40

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

```
select * from SPACEXTABLE
where Launch_Site like
"CCA%" limit 5
```

Total Payload Mass

The total payload carried by boosters from NASA is 48,213 kg.

It means that, between 2010 and 2023, NASA has invested nearly 48 tons of technical equipment in building and sending it into space.

```
select
sum(PAYLOAD_MASS__KG_) as
"Total payload mass" from
SPACEXTABLE where Customer
like "NASA (CRS)%"
```

Total payload mass
48213

Average Payload Mass by F9 v1.1

- The average payload mass carried by the F9 v1.1 booster version is 2,928.9 kg. This means that it can carry nearly 3 tons on each flight. This load is put at risk on each landing.

```
select * from (select
Booster_Version ,
avg(PAYLOAD_MASS__KG_) AS Average
from SPACEXTABLE group by
Booster_Version)    where
Booster_Version = 'F9 v1.1'
```

Booster_Version	Average
F9 v1.1	2928.4

First Successful Ground Landing Date

The dates of the first successful landing outcome on ground pad is 8 April 2016.

```
select min(Date) as "fisrt
succesful" from SPACEXTABLE
where Landing_Outcome = "Success
(drone ship)"
```

fisrt succesful
2016-04-08

Successful Drone Ship Landing with Payload between 4000 and 6000

There no is a boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000.

```
select Booster_Version
from SPACEXTABLE where
PAYLOAD_MASS__KG_ in
(4000,6000) and
Landing_Outcome =
"Success (drone ship)"
```

Booster_Version

Total Number of Successful and Failure Mission Outcomes

Successful_Mission_Outcomes	Failure_Mission_Coutcomes
100	1

```
select (SELECT Count(Mission_Outcome) from SPACEXTABLE where Mission_Outcome
LIKE '%Succ%' ) as Successful_Mission_Outcomes, (SELECT Count( Mission_Outcome)
from SPACEXTABLE where Mission_Outcome LIKE '%Failu%') as
Failure_Mission_Coutcomes
```

The total number of successful is 100 and failure mission outcomes is 1

Boosters Carried Maximum Payload

- There are 7 registers in the names of the booster which have carried the maximum payload mass.

```
select  Booster_Version ,
PAYLOAD_MASS__KG_
from  SPACEXTABLE where
PAYLOAD_MASS__KG_ = (select
max(PAYLOAD_MASS__KG_) as
maximum from SPACEXTABLE) ;
```

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

2015 Launch Records

month	Booster_Version	launch_site	Landing_Outcome
01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

The failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015 has two Booter versions registers

```
select *  from  (select substr(Date, 6,2) as
month , Booster_Version , launch_site ,
Landing_Outcome   from  SPACEXTABLE where
substr(Date,0,5) = "2015") where
Landing_Outcome = "Failure (drone ship)";
```

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

```
select Landing_Outcome ,
count(distinct(Landing_Out
come)) as count from
(SELECT FORMAT(Date,
"yyyy-MM-dd") AS date ,
Landing_Outcome ,
Failure_Success from
SPACEXTABLE) WHERE date >
"2010-06-04" AND date <
"2017-03-20"
```

Landing_Outcome	count
Failure (parachute)	8

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 is 8 Failures

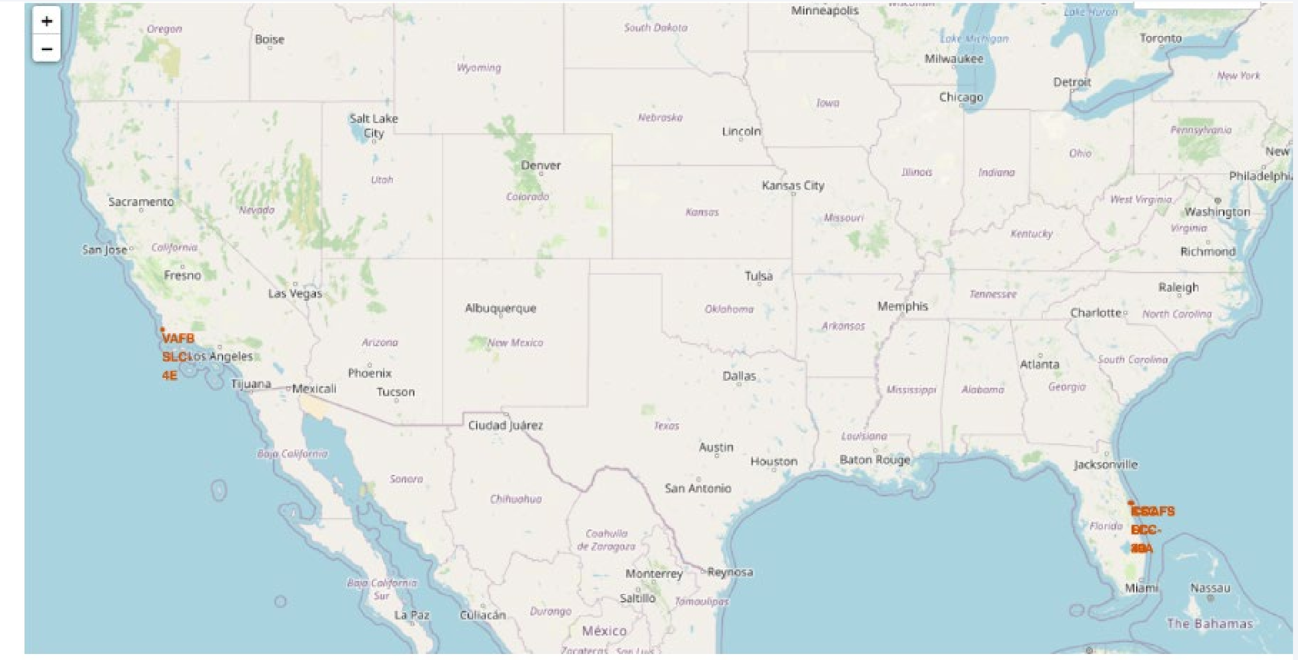
Section 3

Launch Sites Proximities Analysis

Location of the launch sites in the USA

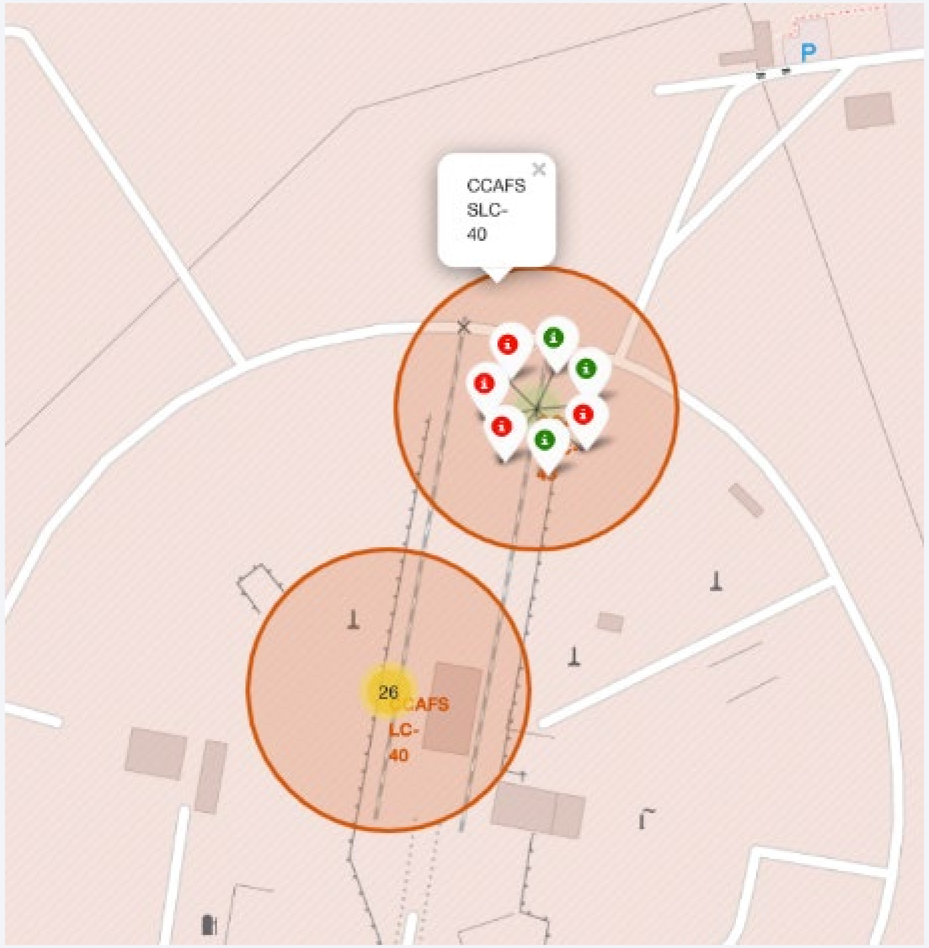
The sites established for launches are identified as being divided both on the Pacific coast and on the Atlantic coast, on the southernmost ends of the United States.

Although the launch platforms are located far from populated areas, they have multiple access routes that guarantee transport logistics.



Location of landing sites according to the result

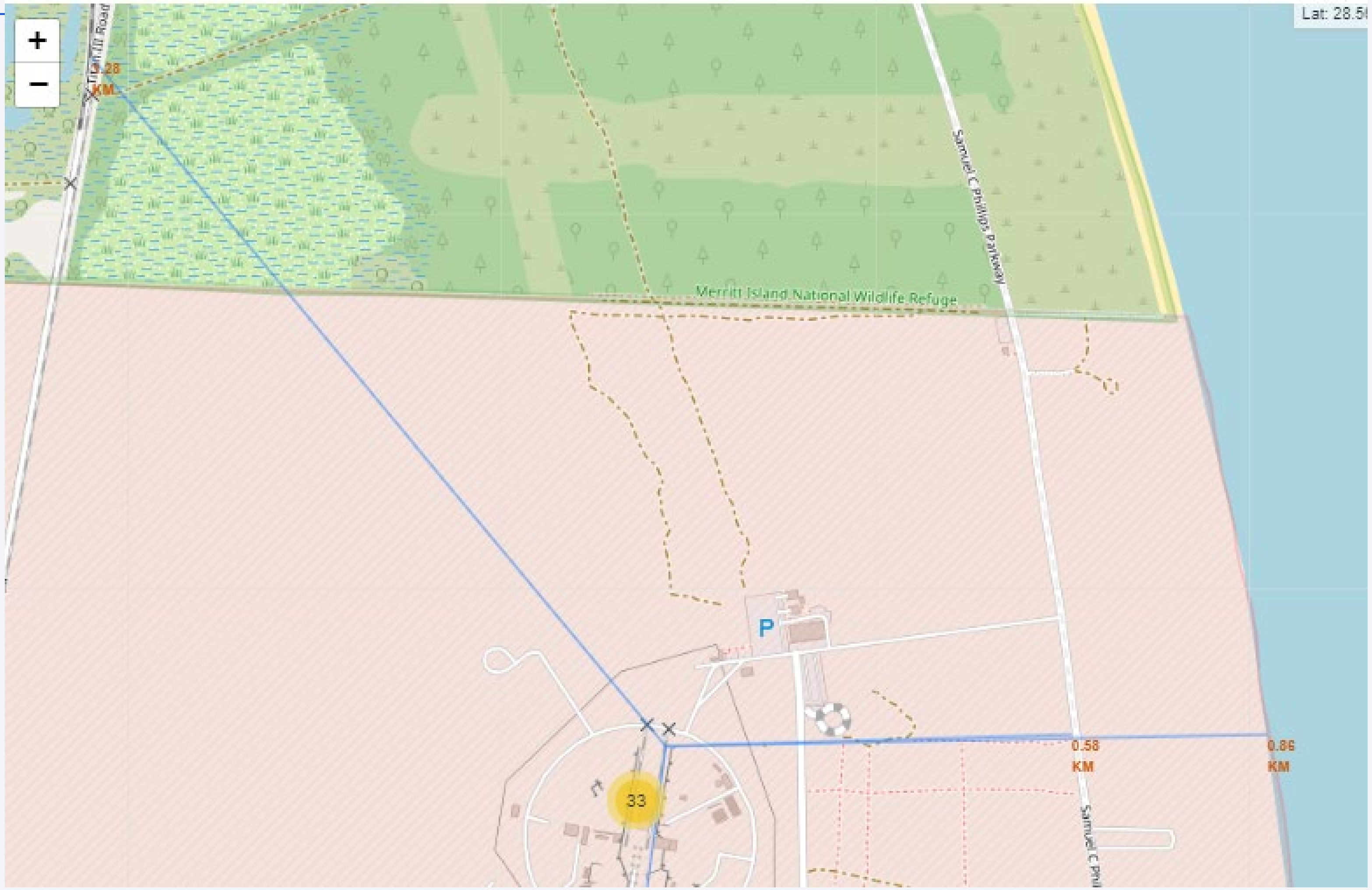
For example, at CCAFS SLC-40 where 7 launches took place. With 4 failed landings and 3 successful landings.



launch site to its proximities such as railway, highway, coastline

For example, in the CCAFA SLC-40 where 7 launches are carried out with an efficiency of 3/7.

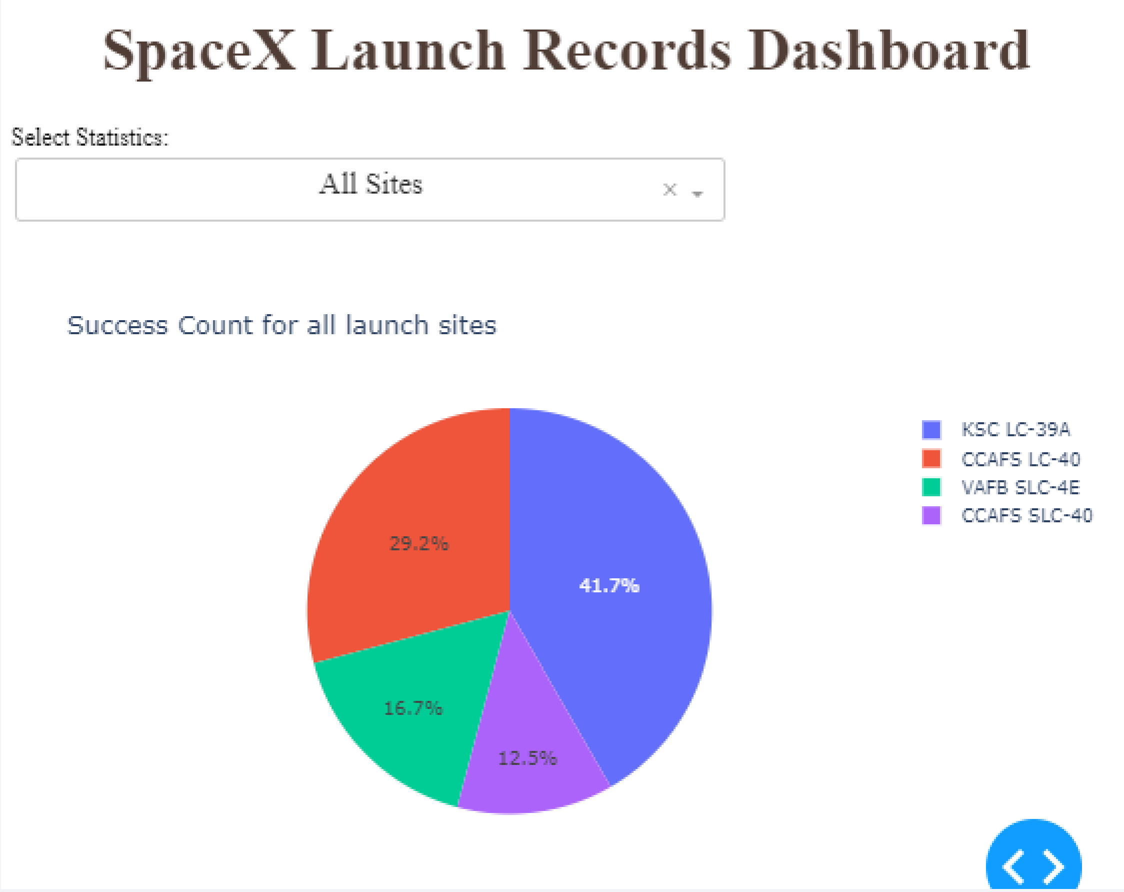
Although it is located 51 km from the nearest city, it is 0.58 km from a motorway, 1.28 km from a railway and 1.28 km from sea routes, i.e. the Atlantic coast.



Section 4

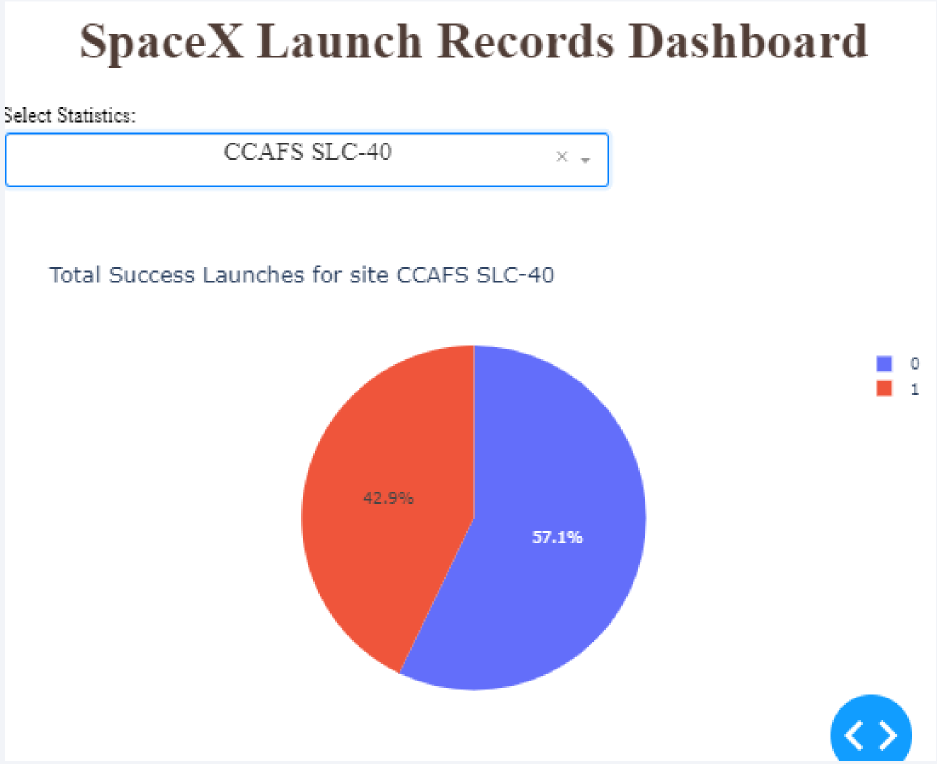
Build a Dashboard with Plotly Dash

Interactive pie chart for landing success rate by SpaceX



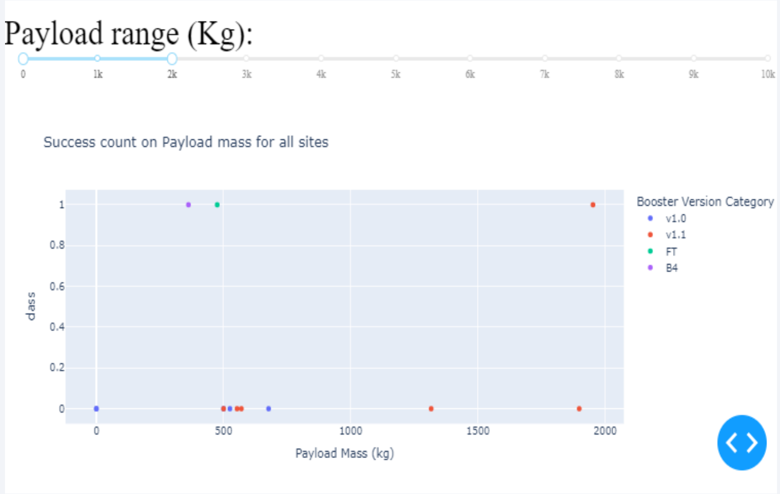
41% of landings are at KSL LC 39

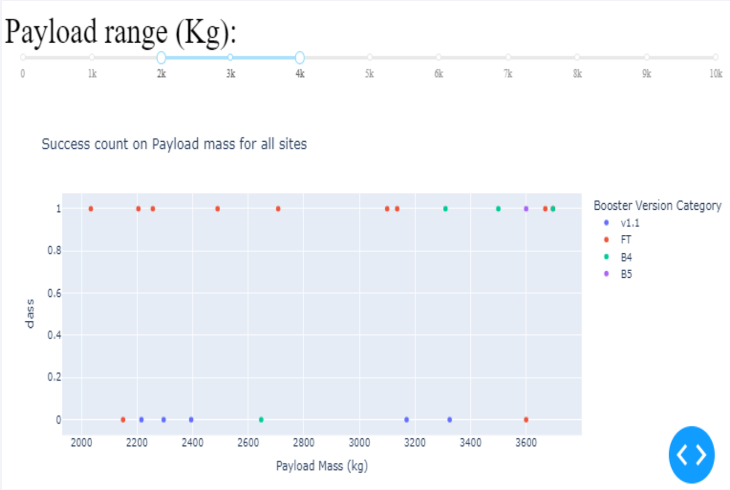
Interactive pie chart for success rate by launch site

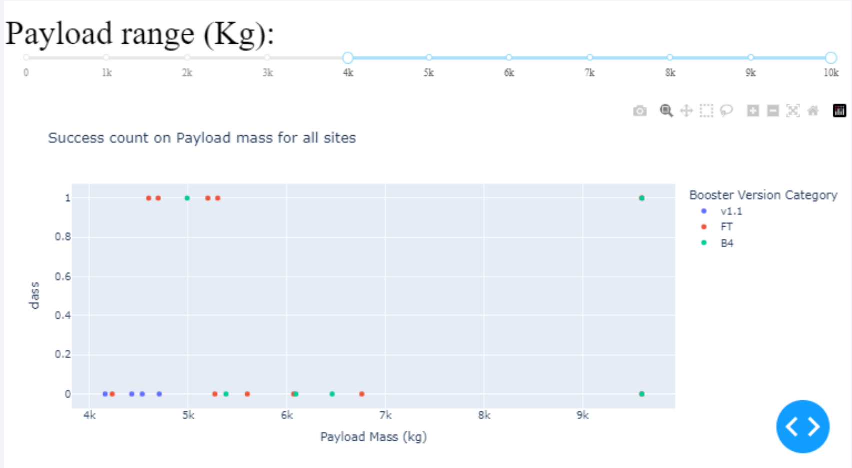


CCAFS SLC-04 has the highest launch success rate, with about 43%

Comparison of interactive scatter plots according to the Payload range







With a Payload range of less than 2 or more than 6 tons the success rate is lower. Conversely, with a Payload range between 2 and 4 tons the success rate increases.

Section 5

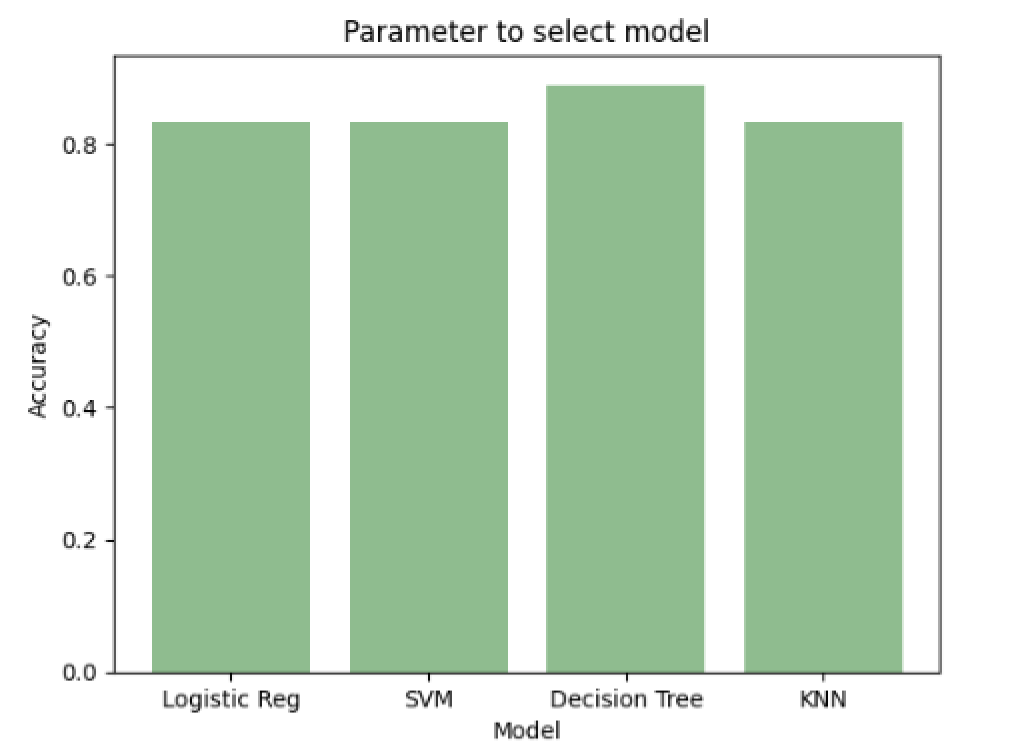
Predictive Analysis (Classification)

Classification Accuracy

The model has the highest classification accuracy is Tree decision with 0,88 of accuracy.

With the following parameters;

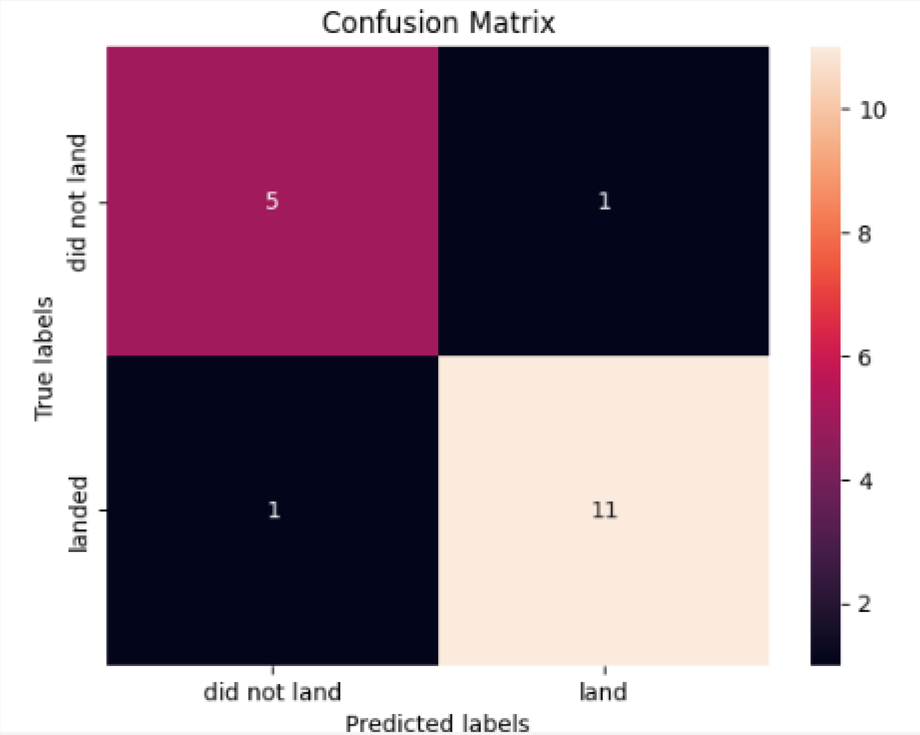
```
'criterion': 'entropy', 'max_depth': 6,
'max_features': 'sqrt', 'min_samples_leaf': 2,
'min_samples_split': 5, 'splitter': 'random'
```



Confusion Matrix of the tree decision model

- Precision = 11/12 = 0,92
- Recall = 11/12 = 0,92

- The tree decision model estimates both successful and failed landings with great precision.



Conclusions

- The VLEO orbit is more crowded and has a higher probability of success than the others.
- Weight is a determinant of landing success.
- Over time, SpaceX has improved its practices and the success rate is higher every year.
- However, the growth rate is stabilizing at 0.8, so the growth rate in the coming years will be decreasing.

Thank you!