IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

José Carlos Caldas da Silva
December 22th , 2022

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- SpaceX designs, manufactures and launches advanced rockets and spacecraft.
- This company was founded in 2002 to revolutionize space technology.
- Using public information, as well machine learning tools, we are predicted the best place to make launches.

Section 1

Methodology

Methodology

- **Data collection:**
    - SpaceX API (link);
    - Web Scrapping from Wikipedia (link).
- **Data wrangling:**
    - Filtering the data;
    - Dealing with missing values;
    - To name the data label based on outcome data after summarizing and analyzing features.
- **Perform exploratory data analysis (EDA) using visualization and SQL;**
- **Perform interactive visual analytics using Folium and Plotly Dash;**
- **Perform predictive analysis using classification models;**

[Click here – Notebook link]

Data Collection – SpaceX API

i. To get answer from API and convert the result to json file:

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.clo
response.status_code
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url).json()
df = pd.json_normalize(response)
```

Python

ii. Cleaning data:

```
# Call getLaunchSite
getLaunchSite(data)
```

[38] ✓ 50.6s

```
# Call getPayloadData
getPayloadData(data)
```

[39] ✓ 48.6s

```
# Call getCoreData
getCoreData(data)
```

[40] ✓ 48.5s

SpaceXAPI: https://api.spacexdata.com/v4/rockets/

[Click here – Notebook link](#)

Data Collection – SpaceX API

iii. Converting list to a dataframe:

Finally lets construct our dataset using the data we have obtained. We we combine the columns into a dictionary.

```python
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

iv. Filtering the dataframe and converting it to a .csv file:

```python
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

✓ 0.1s

Click here – Notebook link

Data Collection - Scraping

i. To get a response from HTML address:

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

[12] ✓ 1.2s Python

```
200
```

ii. To create a BeautifulSoup Object:

Create a BeautifulSoup object from the HTML response

```
# Use BeautifulSoup() to create a BeautifulSoup obje
soup = BeautifulSoup(page.text, 'html.parser')
```

[13] ✓ 1.3s Python

iii. To find the tables of interest:

```
# Use the find_all function in the BeautifulSoup obj
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

[15] ✓ 0.9s Python

iv. To get the columns info:

```
column_names = []

# Apply find_all() function with `th` element on fir
# Iterate each th element and apply the provided ext
# Append the Non-empty column name (`if name is not 
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
        name = extract_column_from_header(temp[x])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

✓ 0.1s Python

HTML Address: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

[Click here – Notebook link]

Data Collection - Scraping

v. Creating a dictionary :

```python
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to b
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

✓ 0.7s Python

vi. Appending data from all keys, converting to a dataframe and saving in .csv:

```python
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowhead
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch
        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')
        #if it is number save cells in a dictonary
        if flag:
```

```python
headings = []
for key,values in dict(launch_dict).items():
    if key not in headings:
        headings.append(key)
    if values is None:
        del launch_dict[key]

def pad_dict_list(dict_list, padel):
    lmax = 0
    for lname in dict_list.keys():
        lmax = max(lmax, len(dict_list[lname]))
    for lname in dict_list.keys():
        ll = len(dict_list[lname])
        if  ll < lmax:
            dict_list[lname] += [padel] * (lmax - ll)
    return dict_list

pad_dict_list(launch_dict,0)

df = pd.DataFrame.from_dict(launch_dict)
df.head()
df.to_csv('spacex_web_scraped.csv', index=False)
```

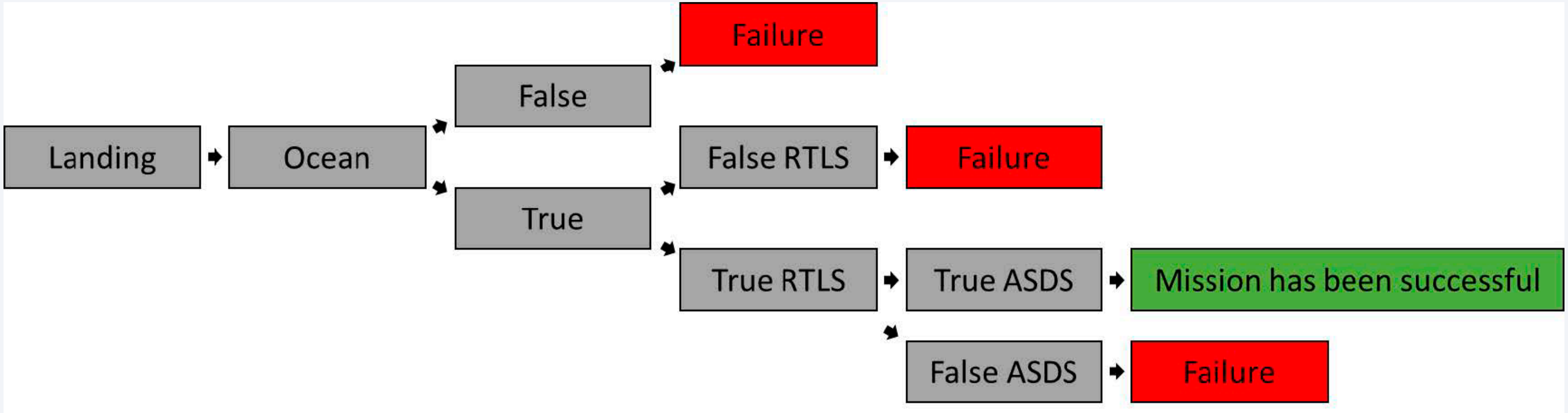
✓ 0.1s

HTML Address: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

Click here – Notebook link

Data Wrangling

Regarding the data analysis process, there are several cases in which the booster did not land successfully:



[Click here – Notebook link]

Data Wrangling

i. Number of launches:

```
# Apply value_counts() on column LaunchSite
df["LaunchSite"].value_counts()
```

```
[34] ✓ 0.1s
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

ii. Number occurrence of Earth orbit:

```
# Apply value_counts on Orbit column
df["Orbit"].value_counts("Orbit")
```

```
[35] ✓ 0.1s
GTO      0.300000
ISS      0.233333
VLEO     0.155556
PO       0.100000
LEO      0.077778
SSO      0.055556
MEO      0.033333
HEO      0.011111
SO       0.011111
ES-L1    0.011111
GEO      0.011111
Name: Orbit, dtype: float64
```

iii. Number mission outcome:

```
# landing_outcomes = values on Outcome column
landing_outcomes = df["Outcome"].value_counts()
landing_outcomes
```

```
[44] ✓ 0.9s
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
None ASDS       2
False Ocean     2
False RTLS      1
Name: Outcome, dtype: int64
```

iv. Creating an outcome label:

```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
[37] ✓ 0.1s
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 None ASDS
6 False Ocean
7 False RTLS
```

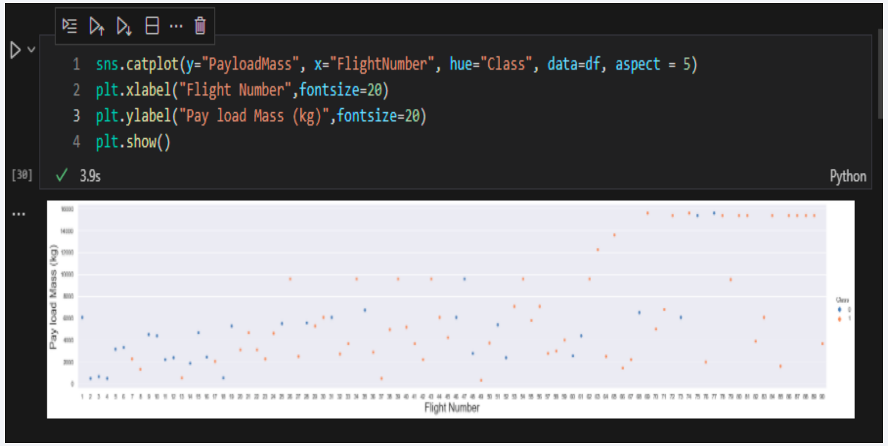
[Click here – Notebook link]

EDA with Data Visualization

Scatter Plots

Scatter plots showed the relationship between two variables (correlation):

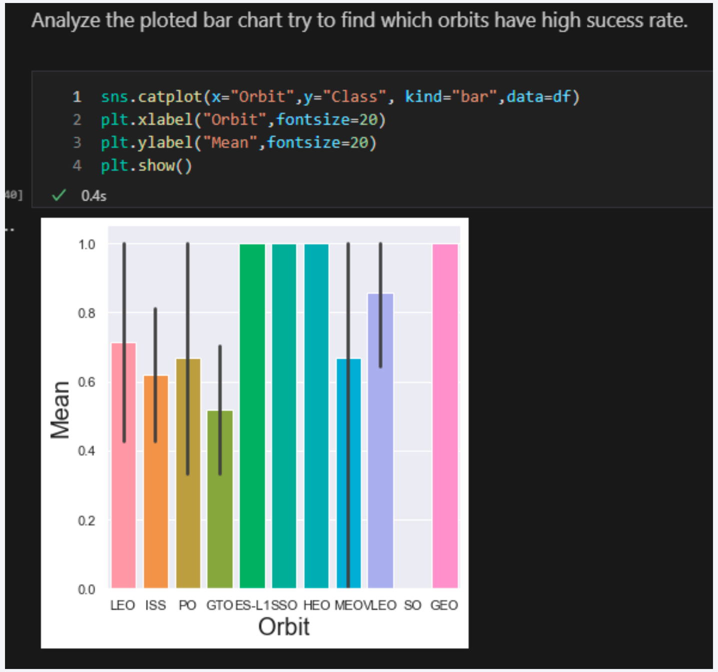
- Flight Number VS. Payload Mass;
- Flight Number VS. Launch Site;
- Payload VS. Launch Site;
- Orbit VS. Flight Number;
- Payload VS. Orbit Type;
- Orbit VS. Payload Masster plot.



Bar Plots

Mean VS Orbt

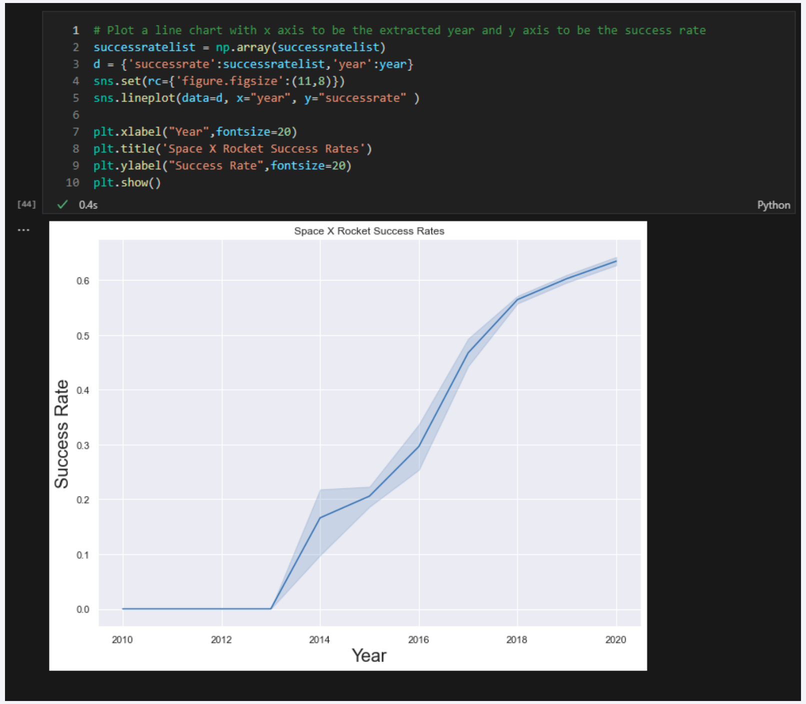
A bar diagram makes it easy to compare sets of data between different groups.



Line Plots

Success Rate VS Year

Line graphs are useful in that they show data variables and trends during the time.



Click here – Notebook link

EDA with SQL

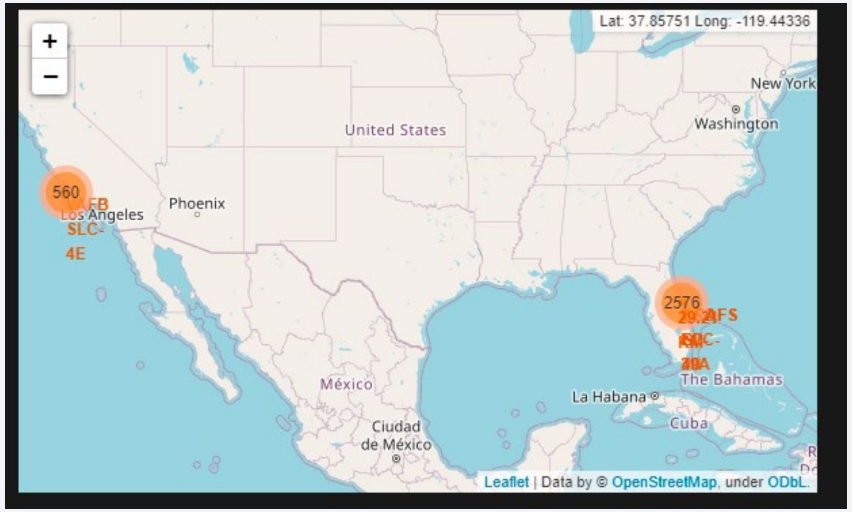
SQL queries performed:

- Display the names of the unique launches sites in the space mission;
- Display 5 records where launch sites begin with 'CCA';
- Display the total payload mass carried by boosters launched by NASA;
- Display the average of the amount paid by booster version F9 v1.1;
- List of the dates of the first successful landing outcome;
- List of the names of the boosters which have success in drone ships and have payloads between 4,000 and 6,000;
- The total od number of successful and failure mission outcomes;
- List of the names of the booster version which have carried the maximum payload mass;
- Listing the records which will display the successful landing outcomes in the ground pad, booster versions, and launch site for the months in the year 2015;
- Ranking the count of successful landing outcomes between the date 2010-06-04 and 2017-03-20.

Click here – Notebook link

Build an Interactive Map with Folium

- To visualize the Launch Data into an interactive map:
  - Using the Latitude and Longitude Coordinates at each launch site and added a Circle Marker around each launch site with a label with the name of the launch site;
  - Dataframe launch_outcomes assigned and converted to classes 0 and 1 with Green and Red, respectively;
  - Red circles at each launch site coordinates with labels showing launch site name.
- Objects created to make easier understanding problem data and showing all launch sites with successful and unsuccessful landings.



Click here – Notebook link

Build a Dashboard with Plotly Dash

- The dashboard has a dropdown, pie chart and scatters plot:
  - The dropdown allows to choose the launch site;
  - The pie chart shows the total of successful and unsuccessful launches sites selected from the dropdowns;
- Scatter Graph establishing the relationship with Outcome and Payload Mass (Kg) for the different Booster Versions:
  - Shows the relationship between two variables;
  - Best to show a nonlinear pattern;
  - The range of data flow, maximum and minimum value, can be determined;
  - Observation and reading easier.

Click here – Notebook link

Predictive Analysis (Classification)

Data Preparation

i. Load our dataset into NumPy and Pandas;
ii. Data transformation;
iii. Split our data into training and test data sets.

Model Evaluation

i. Check the accuracy of each model;
ii. Get tuned hyperparameters for each type of algorithm.

Model Preparation

i. Check quantities test samples;
ii. Decide which better type of machine learning algorithm to use.
iii. Set parameters and algorithms to GridSearchCV;
iv. Training GridSearchCV.

Improving and finding the best classification model

i. Feature Engineering;
ii. Algorithm Tuning;
iii. Comparison between methods;
iv. The model with the best accuracy score wins the best-performing model.

Results

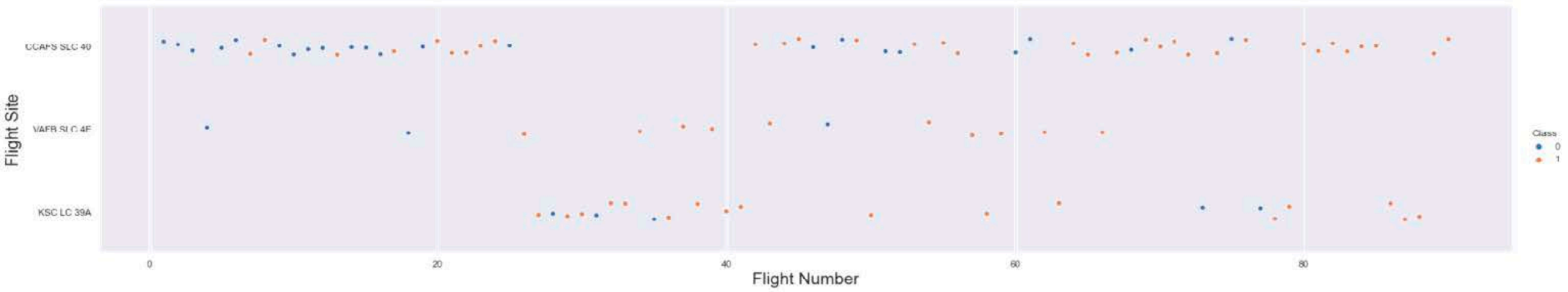
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

[Click here – Notebook link]

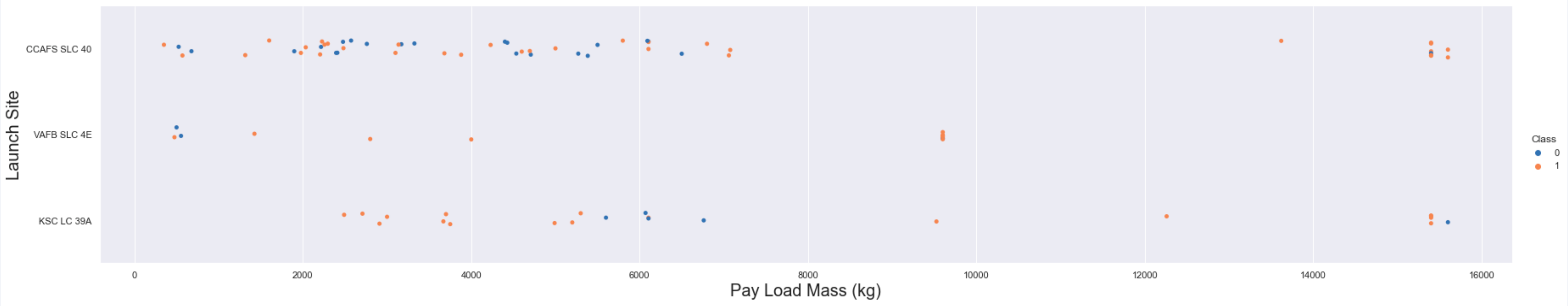
Flight Number vs. Launch Site



It was observed, for each site, the success rate is increasing.

[Click here – Notebook link]

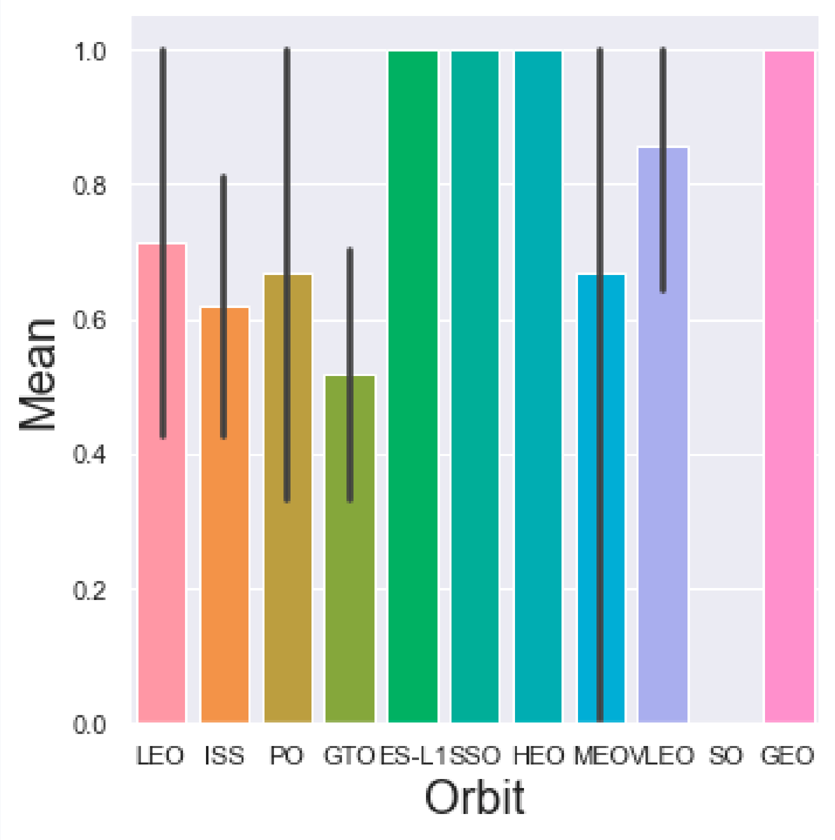
Payload vs. Launch Site

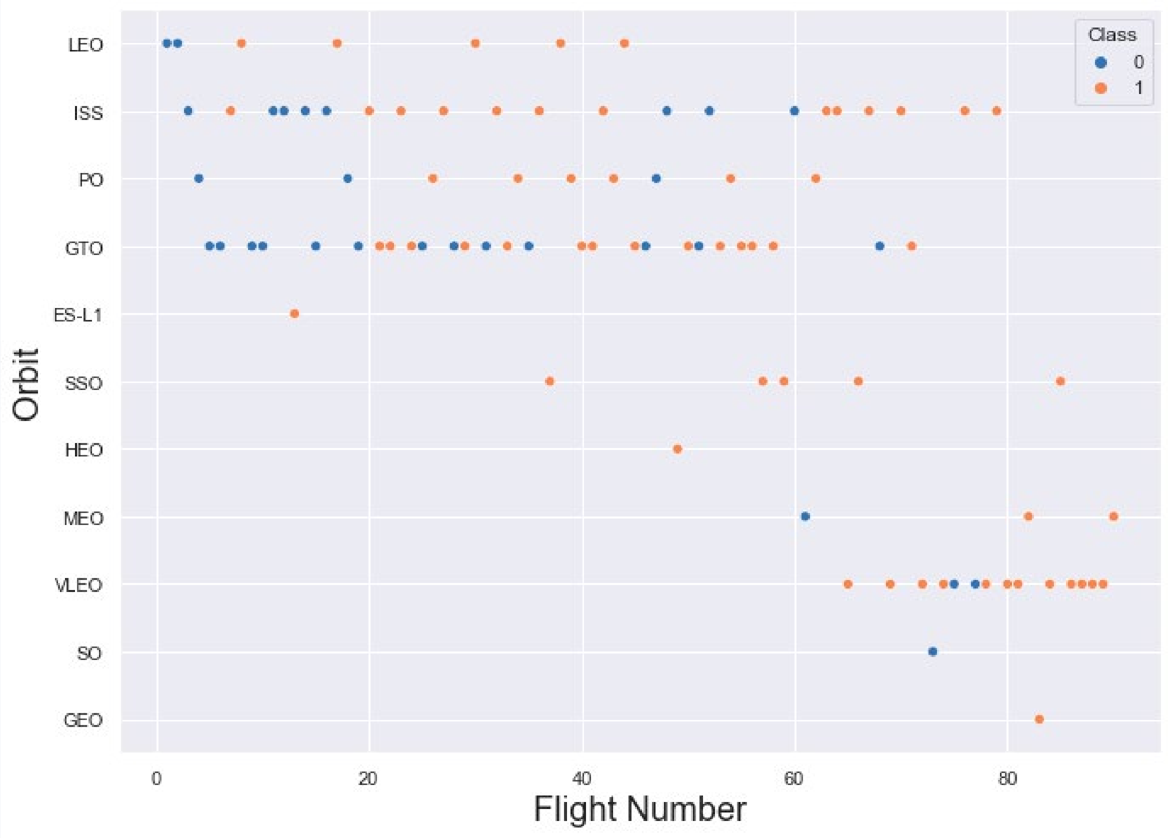


The heavy impacts the launch site. Therefore, a heavier payload may be a consideration for a successful landing.

Click here – Notebook link

Success Rate vs. Orbit Type



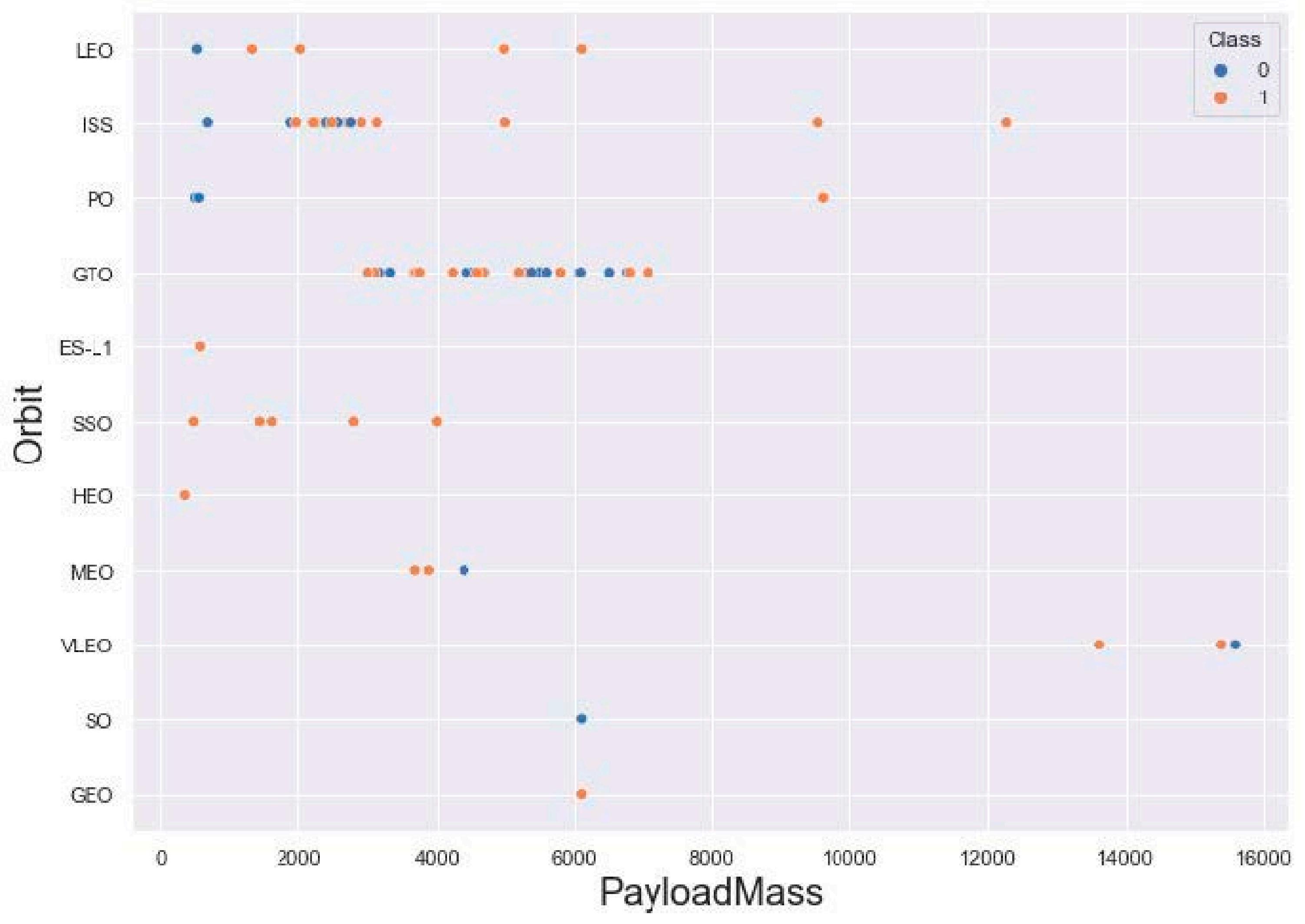


The success rate increases with the number of flights for the LEO orbit.

On the other hand, in other orbits, like GTO, it's not related to success.

Click here – Notebook link

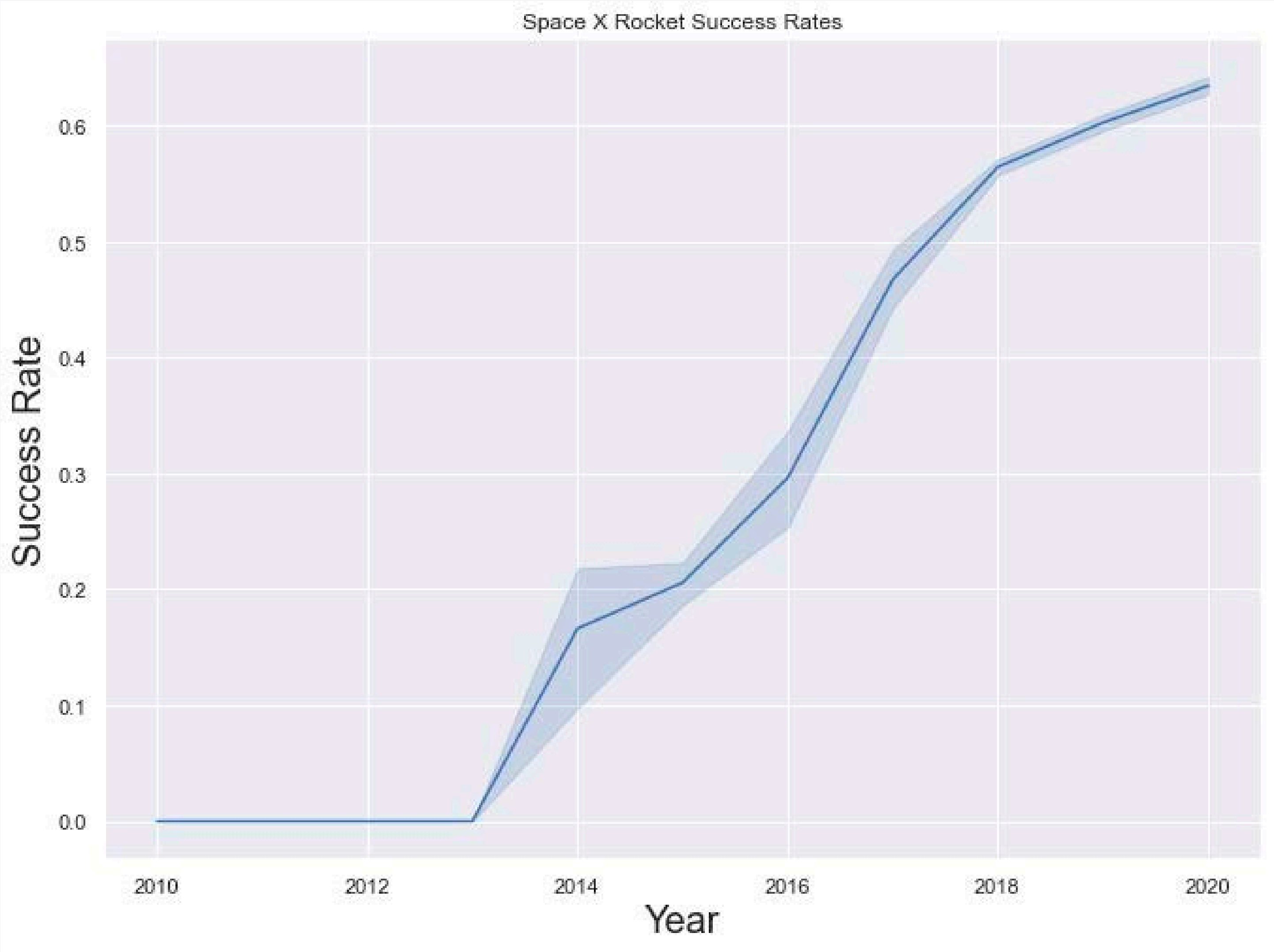
Payload vs. Orbit Type



Heavy payloads have a negative influence on GTO orbits and positive on GTO and polar LEO orbits.

Click here – Notebook link

Launch Success Yearly Trend



Since 2013, the success rate increasing.

[Click here – Notebook link]

All Launch Site Names

```
%sql select DISTINCT launch_site from SpaceX
```

Python

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

Before use DESTINCT in a query, it's necessary to remove all duplicated values.

[Click here – Notebook link]

Launch Site Names Begin with 'CCA'

```
1  %sql select * from SpaceX WHERE launch_site LIKE 'CCA%' limit 5
```

Python

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX
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
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)

WHERE followed by LIKE allows get the launchs that contain subsisting ‘CCA’ and getting only the 5 rows using LIMIT 5.

[Click here – Notebook link]

Total Payload Mass

```python
%sql select SUM(payload_mass__kg_) from SpaceX where Customer = 'NASA (CRS)'
```

Python

1
45596

The query gives the sum of all payload where the customer is equal to NASA (CRS).

[Click here – Notebook link]

Average Payload Mass by F9 v1.1

```
%sql select AVG(payload_mass__kg_) from SpaceX where Booster_Version = 'F9 v1.1'
```

1
2928

WHERE clause filters the dataset to only perform calculations on Booster version F9 v1.1

Click here – Notebook link

First Successful Ground Landing Date

```python
%sql select MIN(DATE) from SpaceX where landing__outcome = 'Success (ground pad)'
```

Python

1
2015-12-22

WHERE clause filters the dataset to only perform calculations on Landing_Outcome = Success (ground pad) and get the first day.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select DISTINCT booster_version from SpaceX where landing__outcome = 'Success
```

Python

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

Click here – Notebook link

The query returns the booster version where landing was successful and payload was between 4k and 6k.

Total Number of Successful and Failure Mission Outcomes

```
1  %sql SELECT mission_outcome, COUNT(*) FROM SpaceX GROUP BY mission_outcome
```

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Click here – Notebook link

Using the function COUNT works out the amount.

[Click here – Notebook link]

Boosters Carried Maximum Payload

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

Using the word SELECT in the query means that it will show values in the Booster Version column from Space.

Click here – Notebook link

2015 Launch Records

DATE	landing__outcome	booster_version	launch_site
2015-01-10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015-04-14	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

DATE LIKE puts the value of 2015.

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

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2016-06-15	14:29:00	F9 FT B1024	CCAFS LC-40	ABS-2A Eutelsat 117 West B	3600	GTO	ABS Eutelsat	Success	Failure (drone ship)
2016-03-04	23:35:00	F9 FT B1020	CCAFS LC-40	SES-9	5271	GTO	SES	Success	Failure (drone ship)
2016-01-17	18:42:00	F9 v1.1 B1017	VAFB SLC-4E	Jason-3	553	LEO	NASA (LSP) NOAA CNES	Success	Failure (drone ship)
2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
2015-01-10	09:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
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)
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)

Function WHERE filters landing_outcome and LIKE (Success or Failure; AND (DATE between) DESC means its arranging the dataset into descending order.

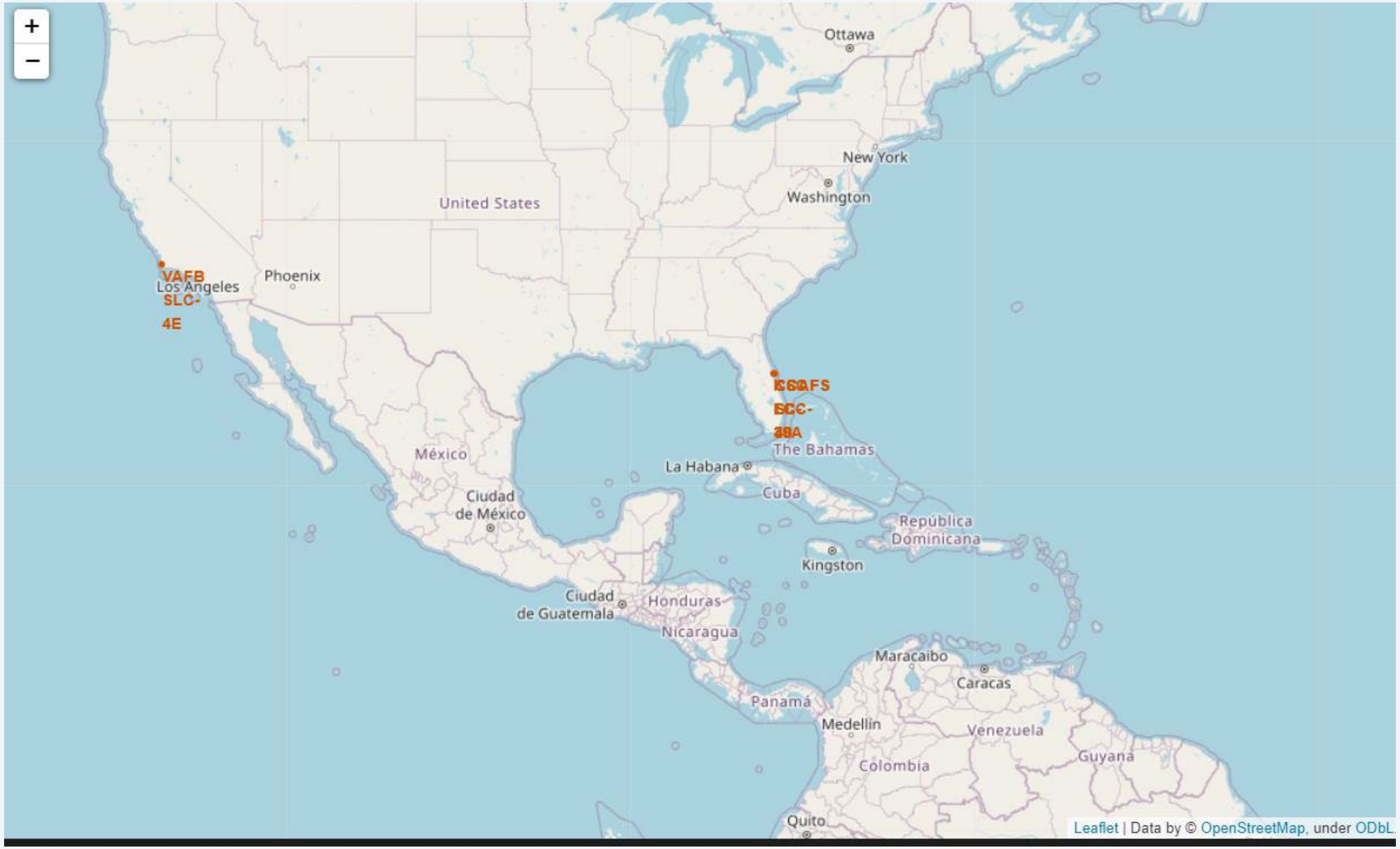
Click here – Notebook link

Section 3

Launch Sites Proximities Analysis

[Click here – Notebook link]

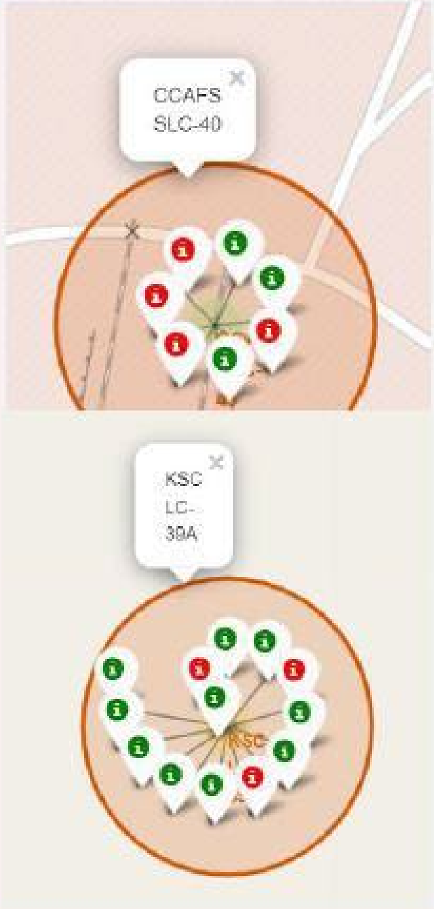
Space X in United States

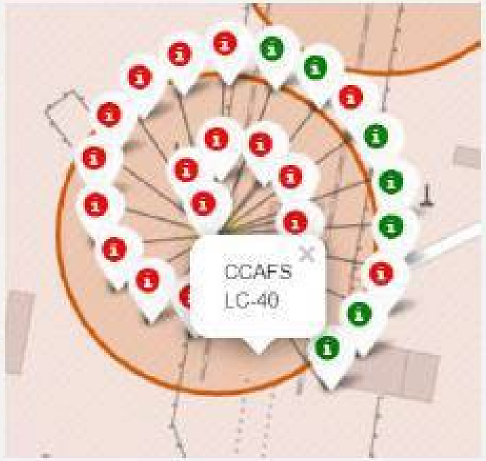


Space X sites.

[Click here – Notebook link]

Color-labeled launch outcomes





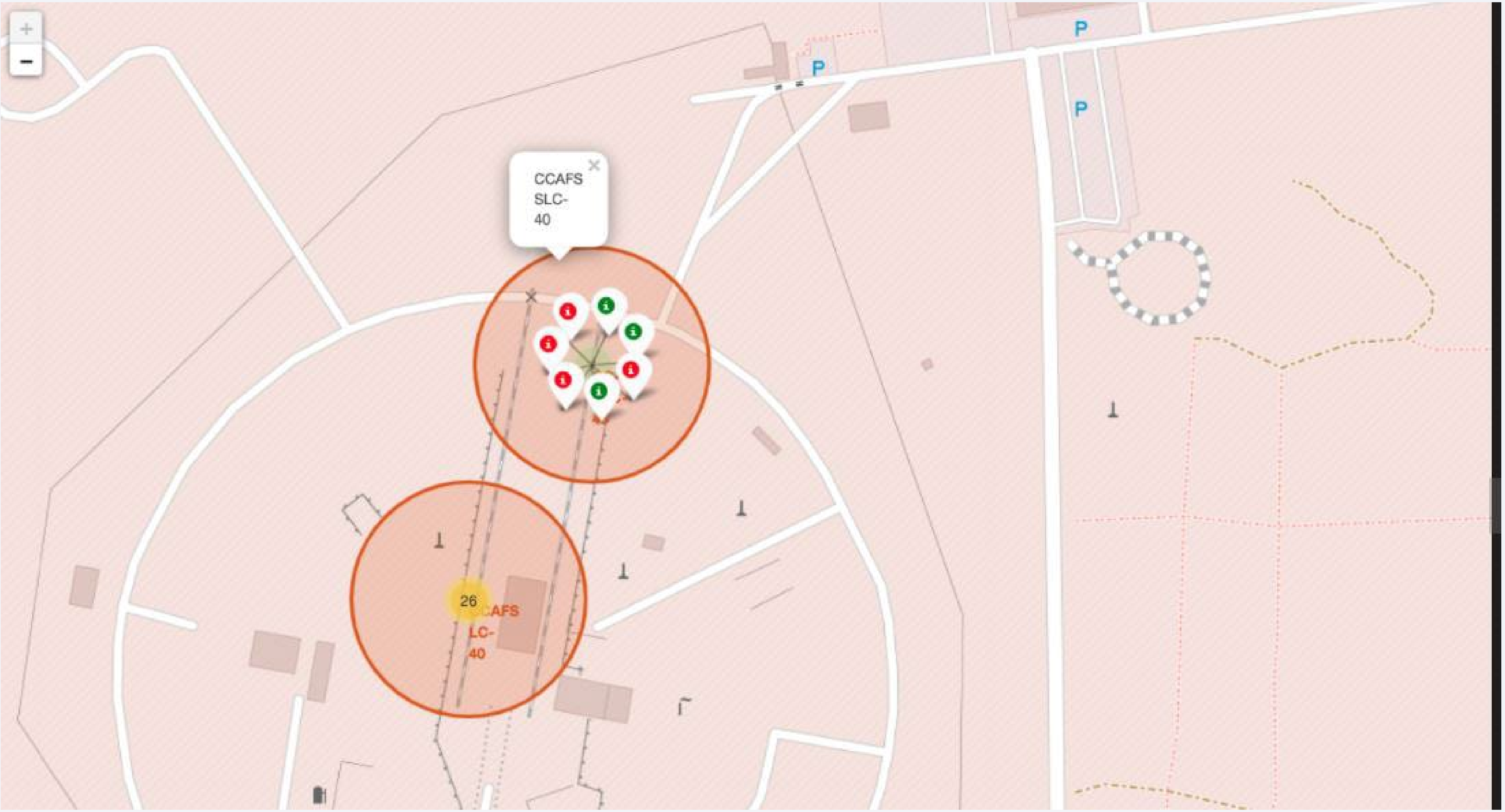
Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

[Click here – Notebook link]

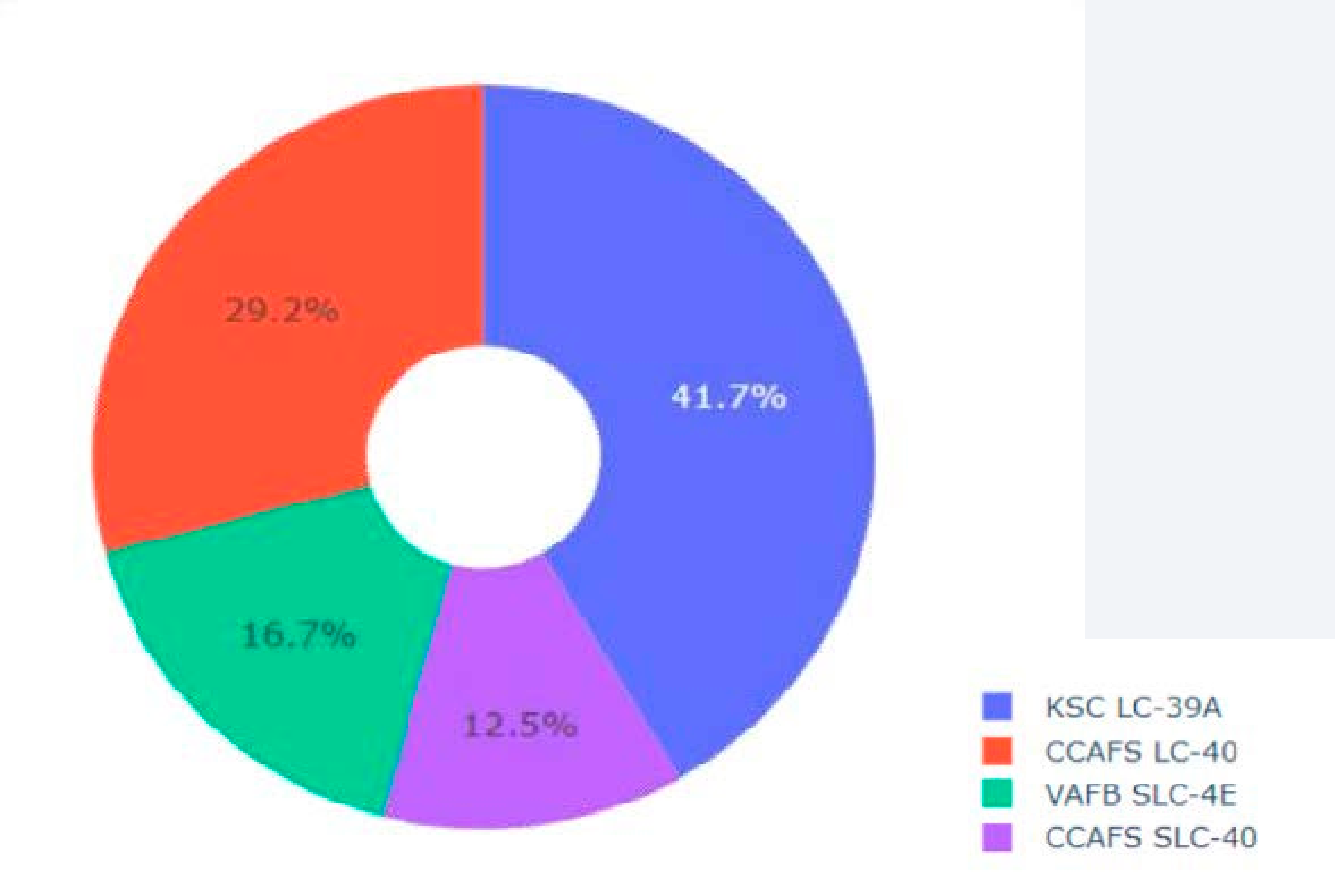
Color-labeled launch



Section 4

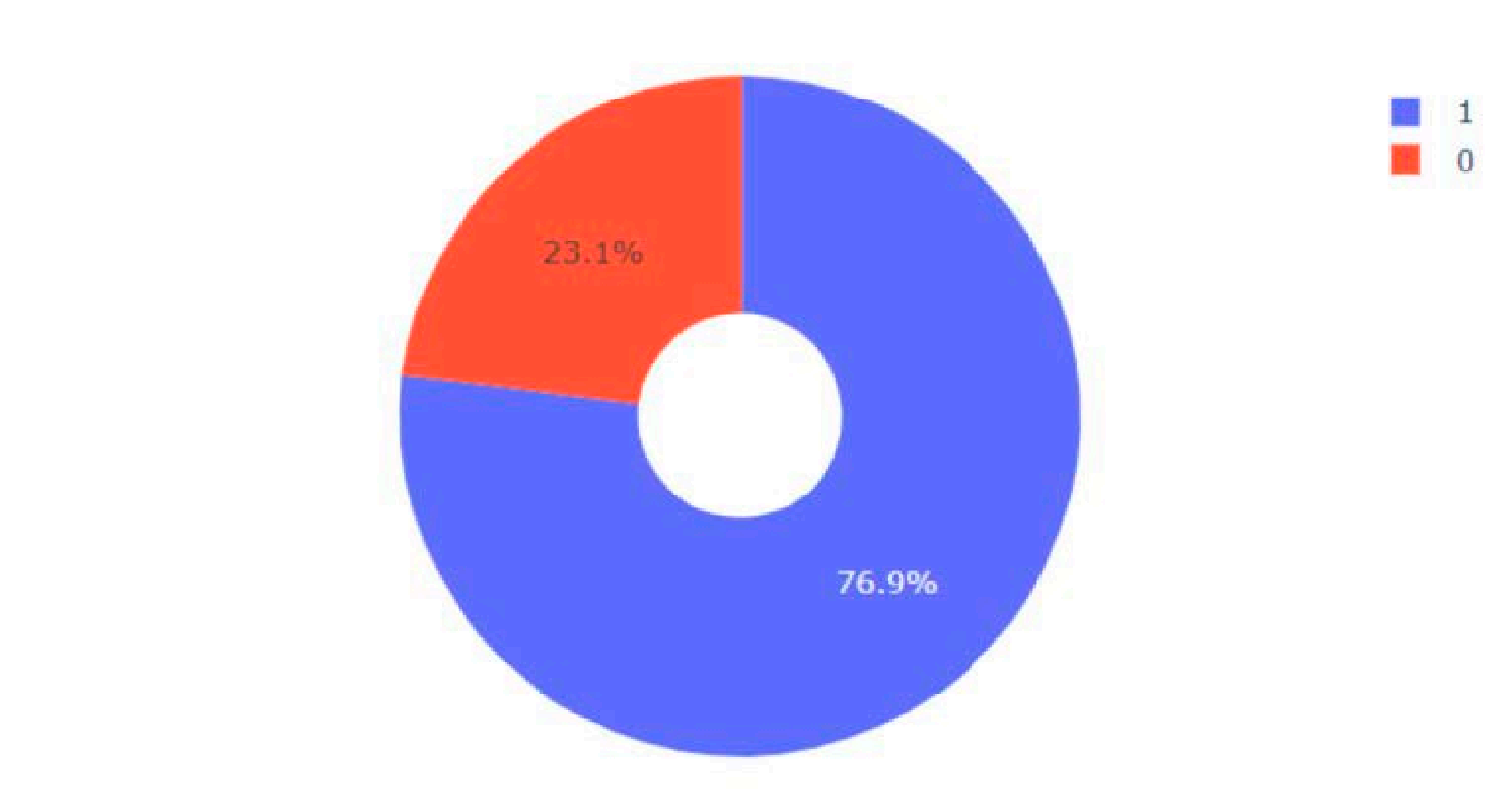
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage



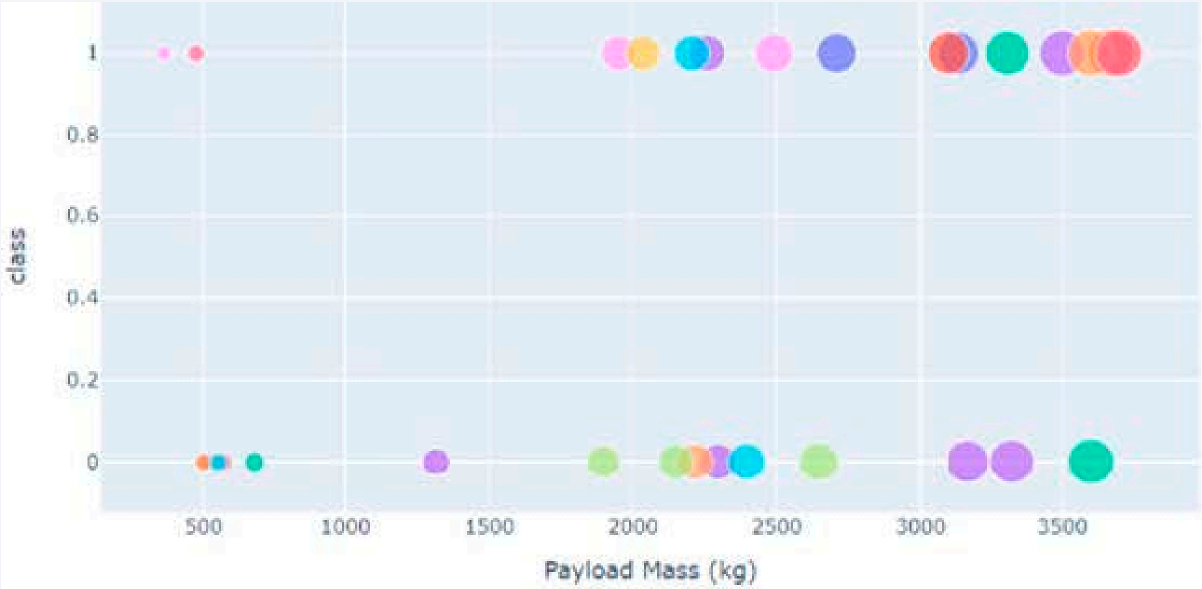
KSC had the most successful launches.

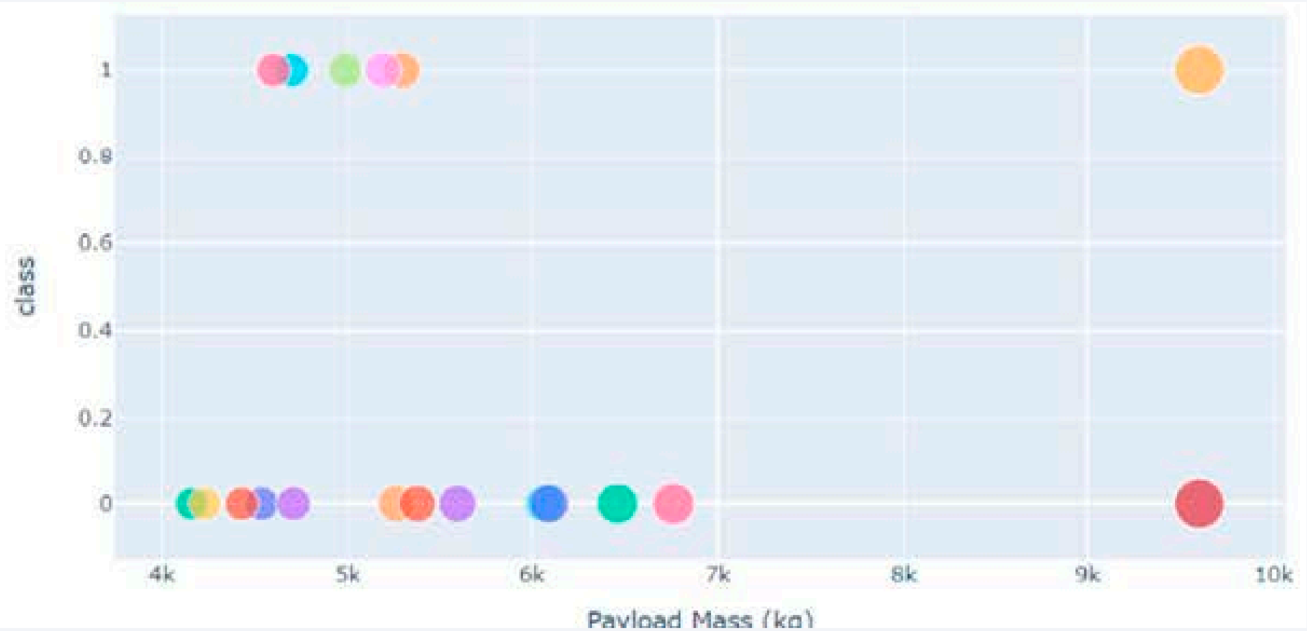
Pie chart showing the Launch site with the highest launch success ratio



KSC LC 39A achieved a 76,9% of success.

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider





The success rates for low weighted payload is higher than the heavier payloads.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

Click here – Notebook link

```python
parameters = {'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]}

tree = DecisionTreeClassifier()
```

[31] ✓ 0.1s Python

```python
tree_cv=GridSearchCV(tree, param_grid=parameters, cv=10)
tree_cv.fit(X_train,Y_train)
```

[32] ✓ 6.2s Python

```
GridSearchCV(cv=10, estimator=DecisionTreeClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [2, 4, 6, 8, 10, 12, 14, 16, 18],
                         'max_features': ['auto', 'sqrt'],
                         'min_samples_leaf': [1, 2, 4],
                         'min_samples_split': [2, 5, 10],
                         'splitter': ['best', 'random']})
```

```python
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

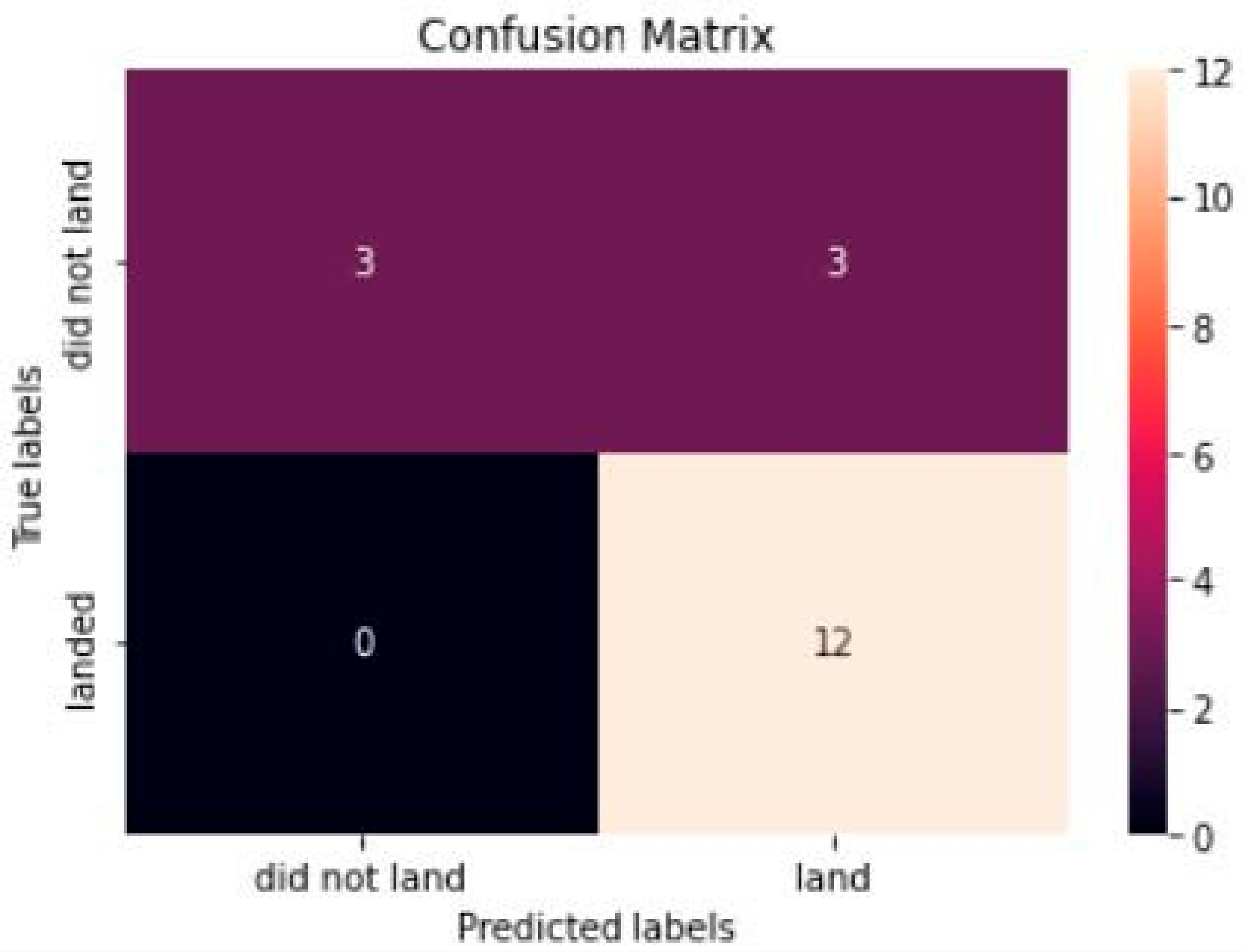
[33] ✓ 0.7s Python

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 1,
'min_samples_split': 2, 'splitter': 'random'}
accuracy : 0.8767857142857143
```

The decision tree was the best model based in the classification accuracy.

[Click here – Notebook link]

Confusion Matrix



The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the classes.

Conclusions

- The larger the flight amount at a launch site;
- The success rates for SpaceX launches is directly proportional time in years;
- Orbits ES L1, GEO, HEO, SSO, VLEO had the most success rate;
- The Decision tree classifier is the best machine learning algorithm for this task;

Thank you!