IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Anas Aamoum
12/03/2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection
  - Data Wrangling
  - EDA with data visualization
  - EDA with SQL
  - Interactive map with Folium
  - Dashboard with Plotly Dash
  - Predictive analysis
- Summary of all results
  - EDA results
  - Interactive analysis (Plotly)
  - Predictive analysis results

Introduction

- Project background and context
  - Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used by our company Space-Y so that we can bid against space X for a rocket launch.
- Problems you want to find answers
  - Predict whether the first stage of the Falcon 9 rocket will land successfully.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX Rest API
  - Web Scrapping from the Wikipedia page of Space X launches
- Perform data wrangling
  - Data was processed using one hot encoding for categorical 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
  - Several models were built and evaluated: Linear Regression, KNN, SVM, Decision Tree

Data Collection

- Data about Space X launches was request from the Space X API.
- The HTTP response was converted to a JSON object.
- The information contained in the JSON included: rocket, payload mass, success/fail, etc…
- The information was then converted to a dataframe and saved to a csv.
- Another way used to collect data is by scraping data from Wikipedia using BeautifulSoup.

Data Collection – SpaceX API

- Data collection from the Space X REST API
- Notebook : click to open [Data-Collection-Jupyter-Notebook]

1- GET request to endpoint:

Now let's start requesting rocket launch data from SpaceX API with the following URL:

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

2- Transform JSON to Dataframe

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
data = pd.json_normalize(response.json())
```

3- Clean data (check notebook 13-28)

4- Export to CSV

In [30]:
```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection - Scraping

- Collect data by web scrapping from Wikipedia
- Notebook : click to open [Web scrapping notebook]()

1- GET request and receive a HTML response. Then, make a soup of it:

```
In [30]: # 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`

```
In [31]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
         soup = BeautifulSoup(response.content, "html.parser")
```

2- Find the tables

```
In [33]: # 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')
```

3- Get column names

```
rows = first_launch_table.find_all('th', {"scope" : "col"})


for row in rows:
  col = extract_column_from_header(row)
  column_names.append(col)
```

4- Create the dataframe

5- Export the dataframe to csv

Data Wrangling

Check null values

Calculate number of launches on each site

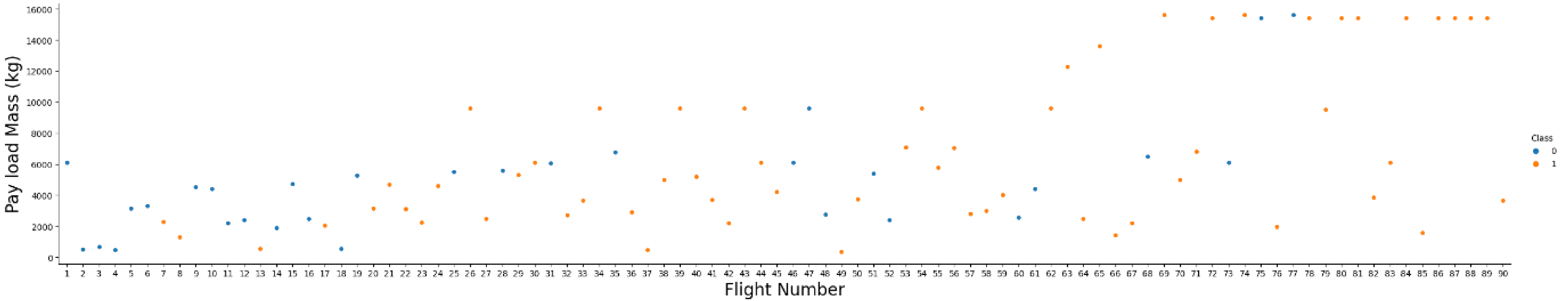
Calculate the number and occurrence of each orbit

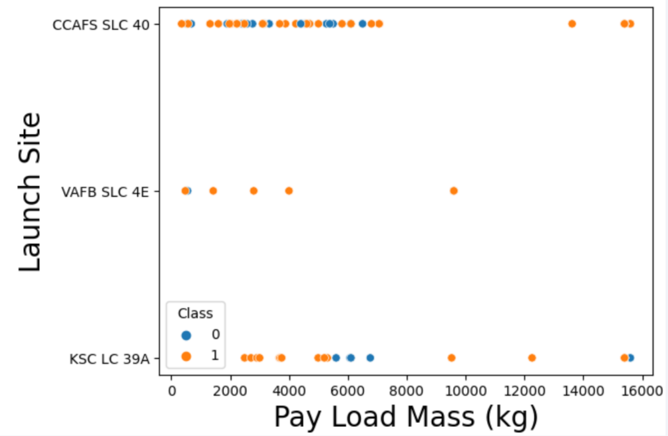
Calculate the number and occurrence of mission outcome per orbit type

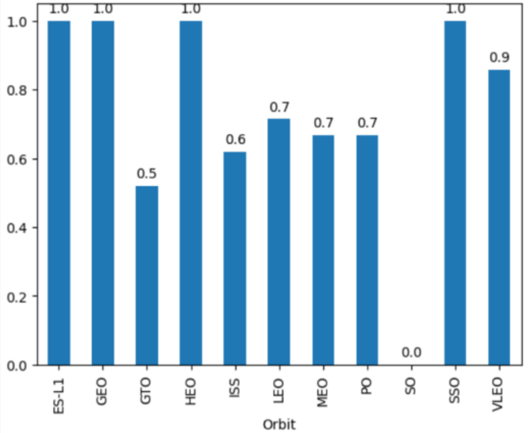
Create a landing outcome label from Outcome column

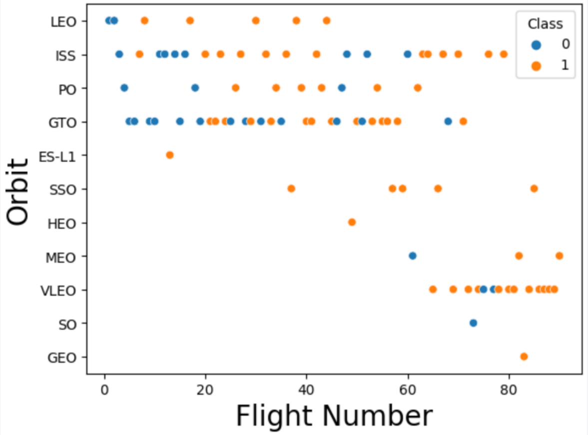
Notebook : click to open [Data wrangling notebook]

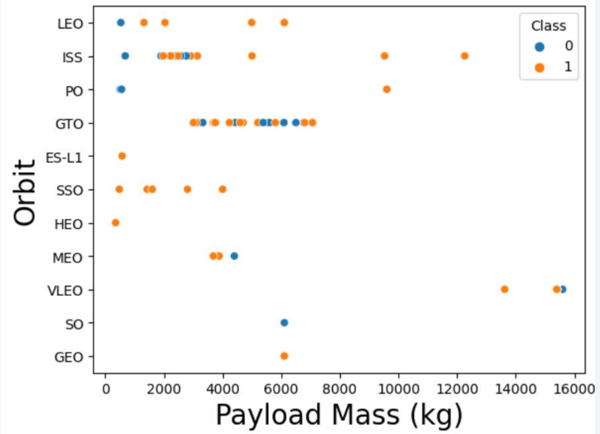
EDA with Data Visualization











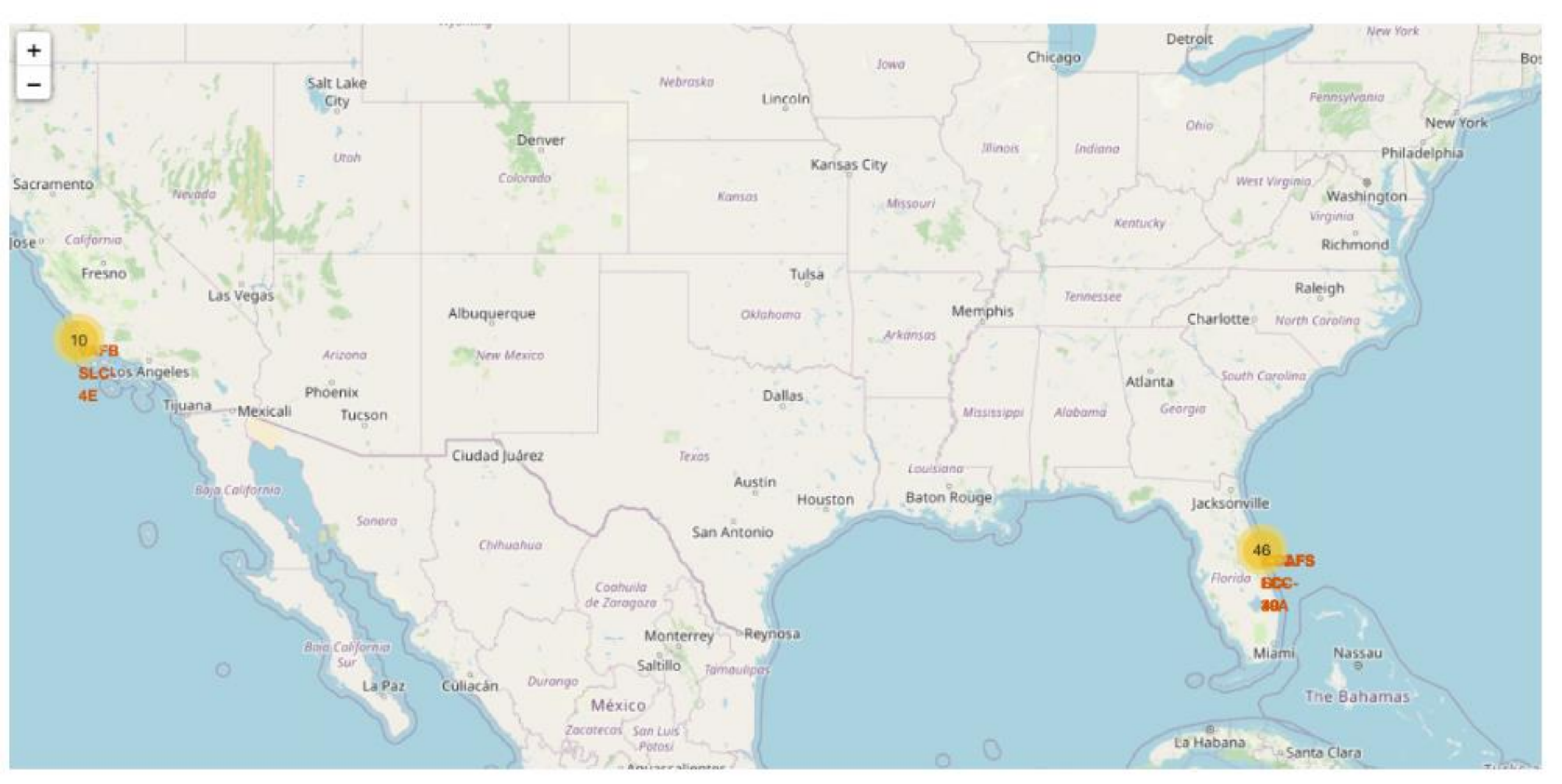
Notebook : click to open [Data visualization notebook]

EDA with SQL

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

Notebook : click to open EDA with SQL notebook

Build an Interactive Map with Folium



Map markers added in order to help in finding the optimal location for a launch site.

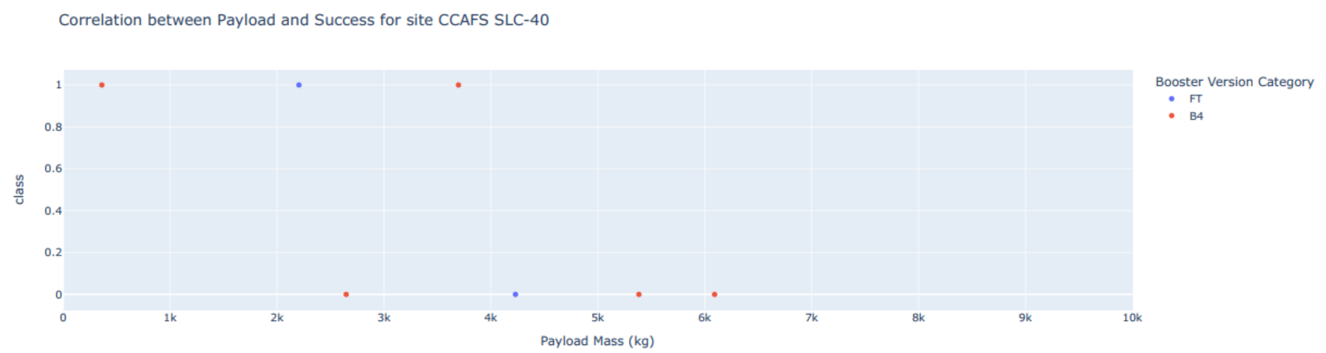
Notebook : click to open [Folium map notebook]

Build a Dashboard with Plotly Dash

This plot shows the how successful launches were split among launch sites.

Python Script : click to open [Plotly Dashboard]()

CCAFS LC-40 had 73.1% successful launches

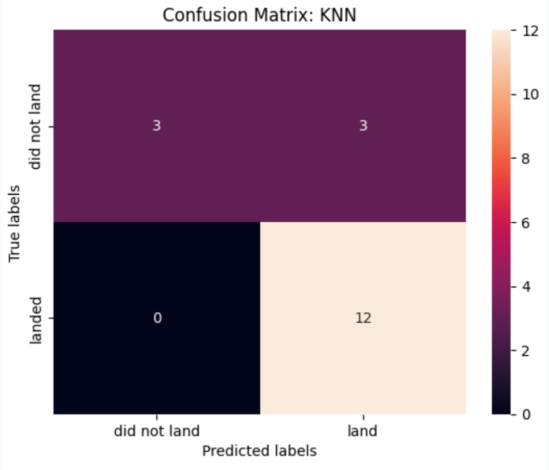


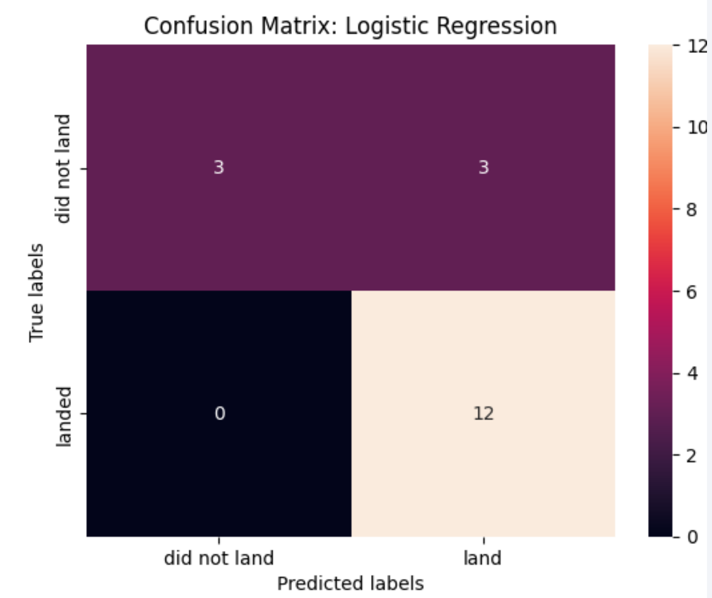
Predictive Analysis (Classification)

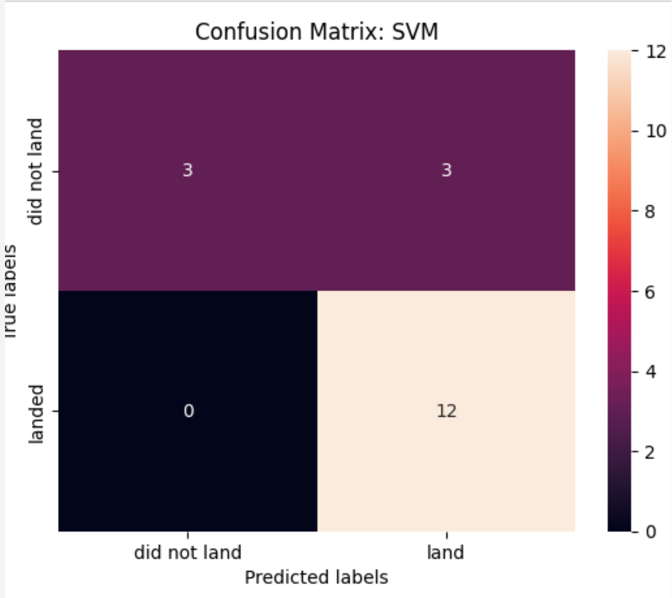
- 4 models were built: SVM, KNN, LR and DT. The accuracy of each model on the test set was calculated.

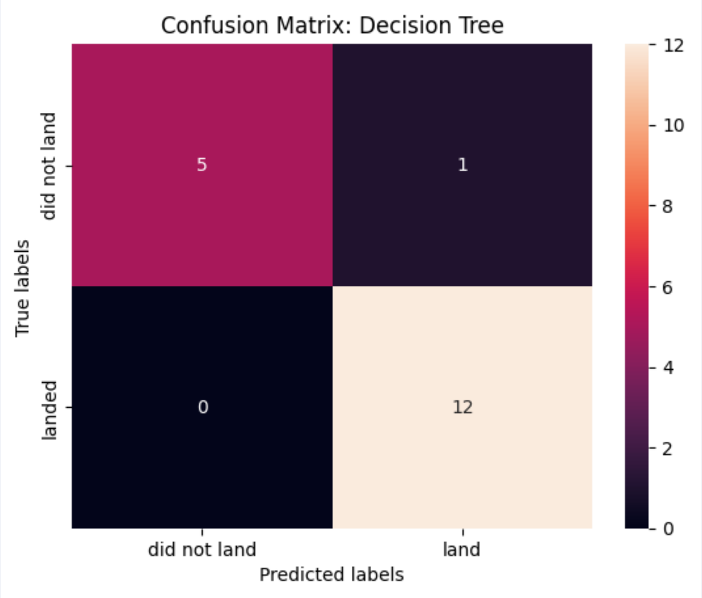
```
In [35]: print('Accuracy for Logistics Regression method:', logreg_cv.score(X_test, Y_test))
         print( 'Accuracy for Support Vector Machine method:', svm_cv.score(X_test, Y_test))
         print('Accuracy for Decision tree method:', tree_cv.score(X_test, Y_test))
         print('Accuracy for K nearsdt neighbors method:', knn_cv.score(X_test, Y_test))

Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.7222222222222222
Accuracy for K nearsdt neighbors method: 0.8333333333333334
```









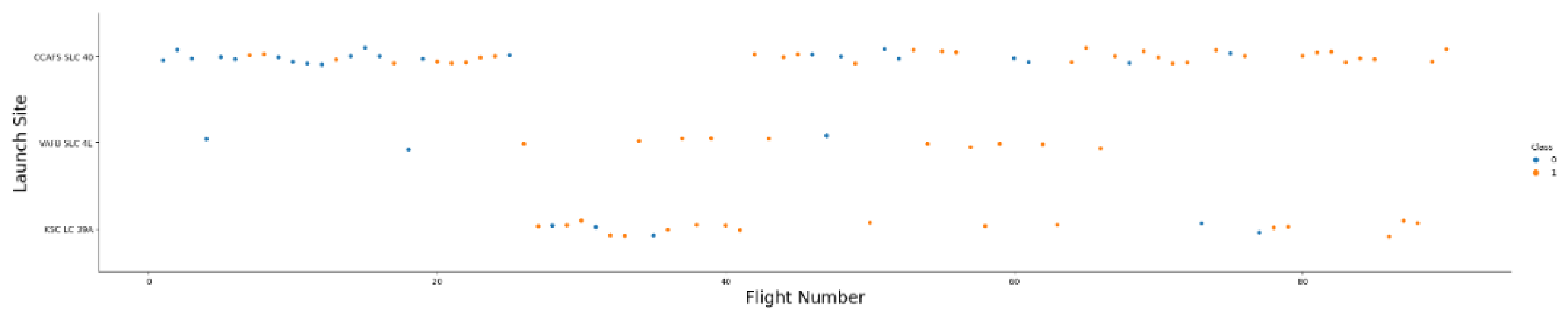
Results

- SVM, KNN and logistic regression have performed the best in predicting the outcome of a launch. The decision trees performed the worst.
- The success rate increased with the years. The payload also increased with the years.
- Orbits GEO, HEO, SSO and ES L1 have the best success rate of launching.

Section 2

Insights drawn from EDA

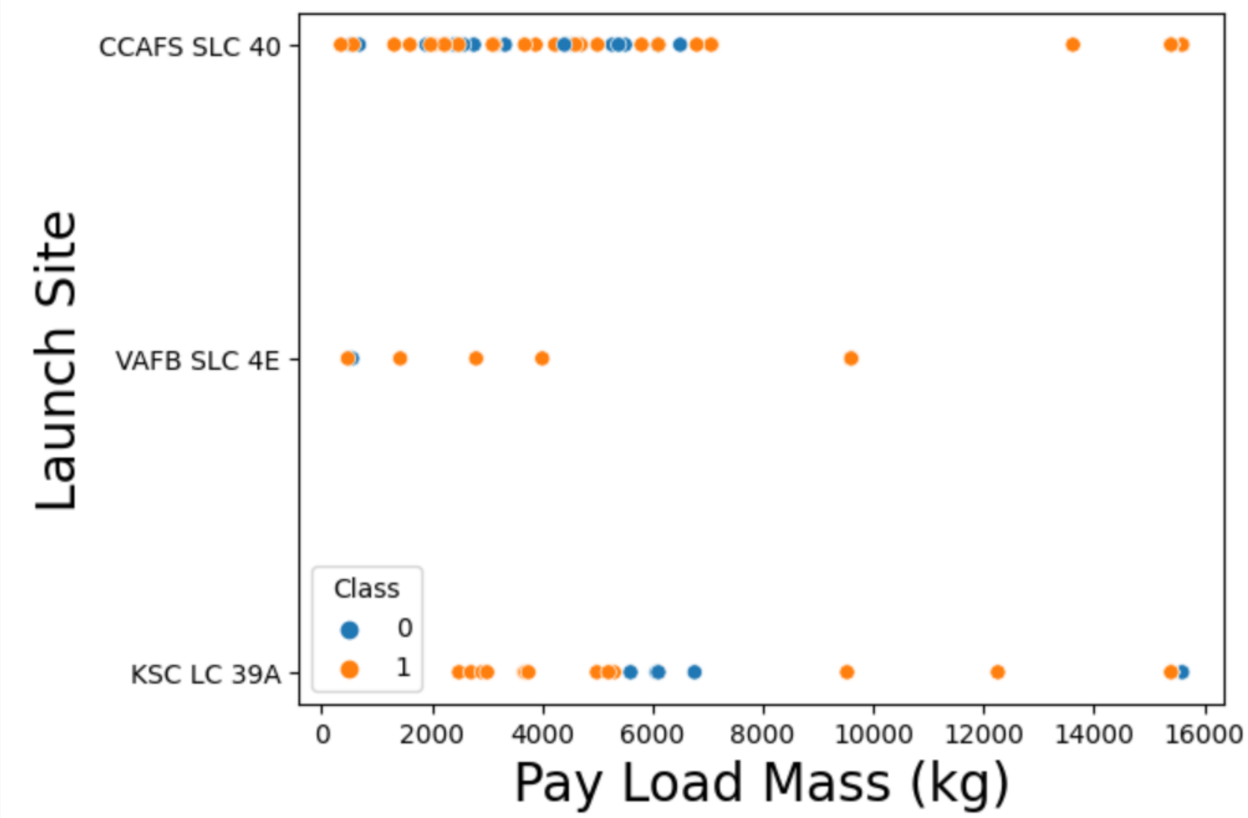
Flight Number vs. Launch Site



- Most of the initial launches happened in CCAFS SLC 40.
- Most of the initial launches were unsuccessful.

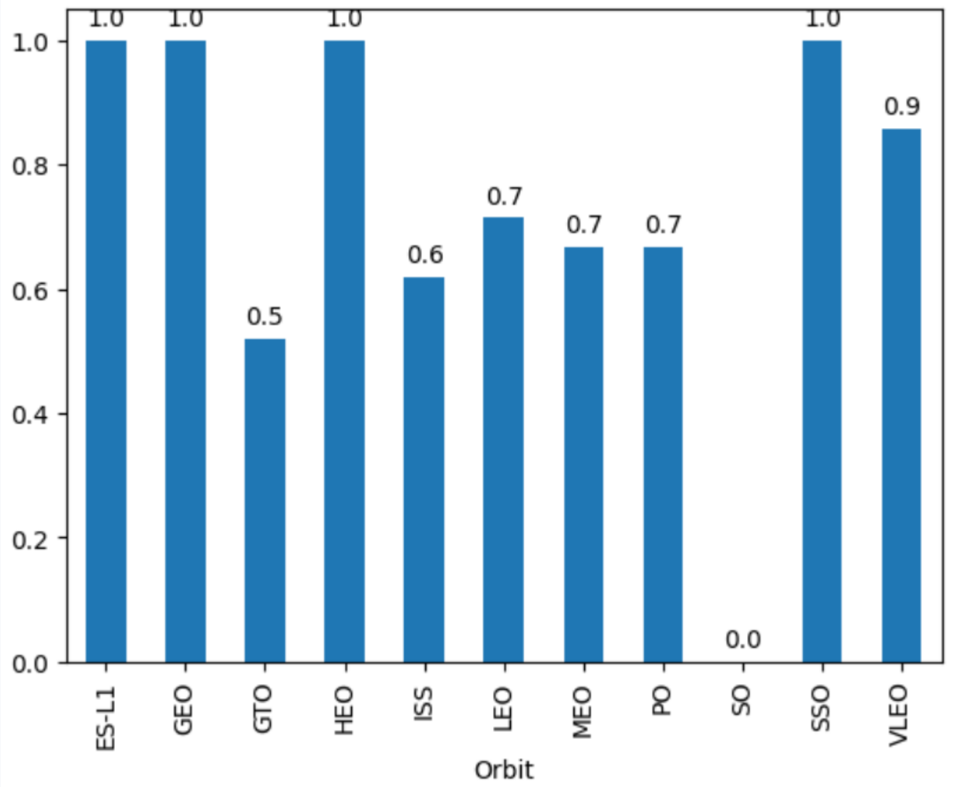
Payload vs. Launch Site

- High payload launches took place in CCAFS SLC 40 and KSC LC 39A



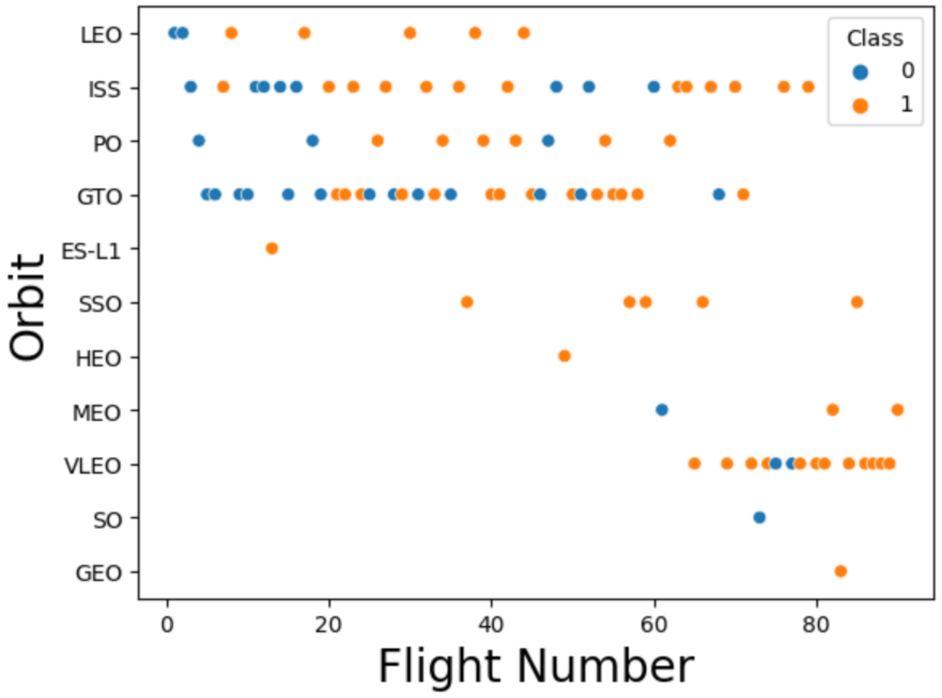
Success Rate vs. Orbit Type

- Orbits GEO, HEO, SSO and ES L1 have the best success rate of launching.
- GTO and SO orbits have the lowest success rate.



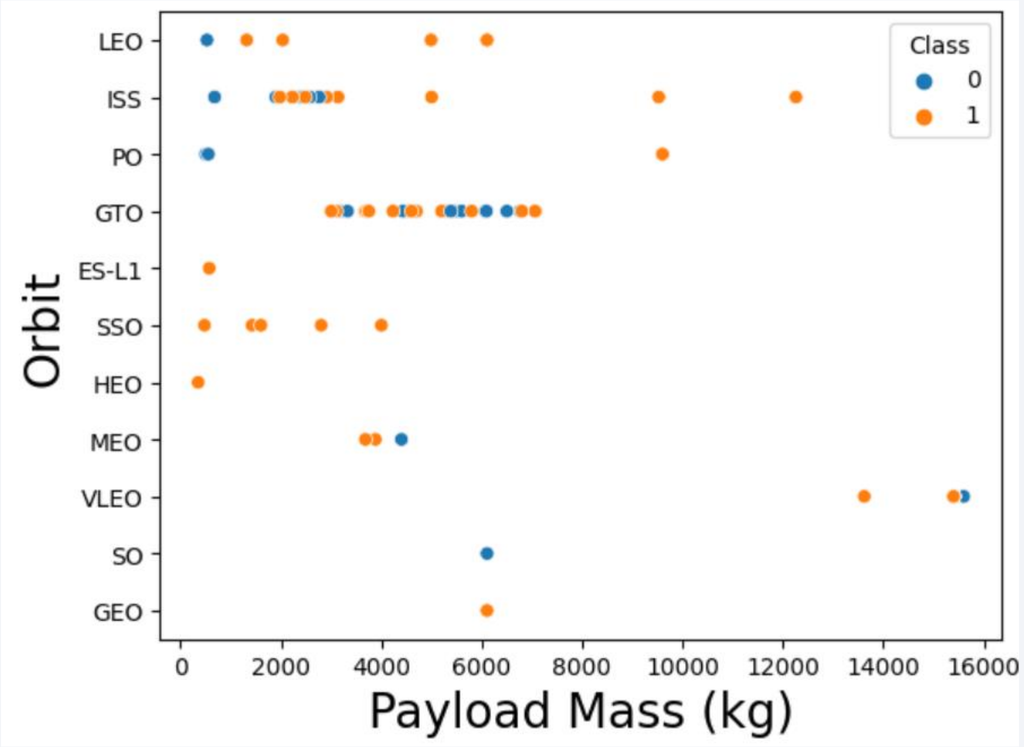
Flight Number vs. Orbit Type

- Some orbits have happened only in the recent years: VLEO, SO, GEO, MEO.



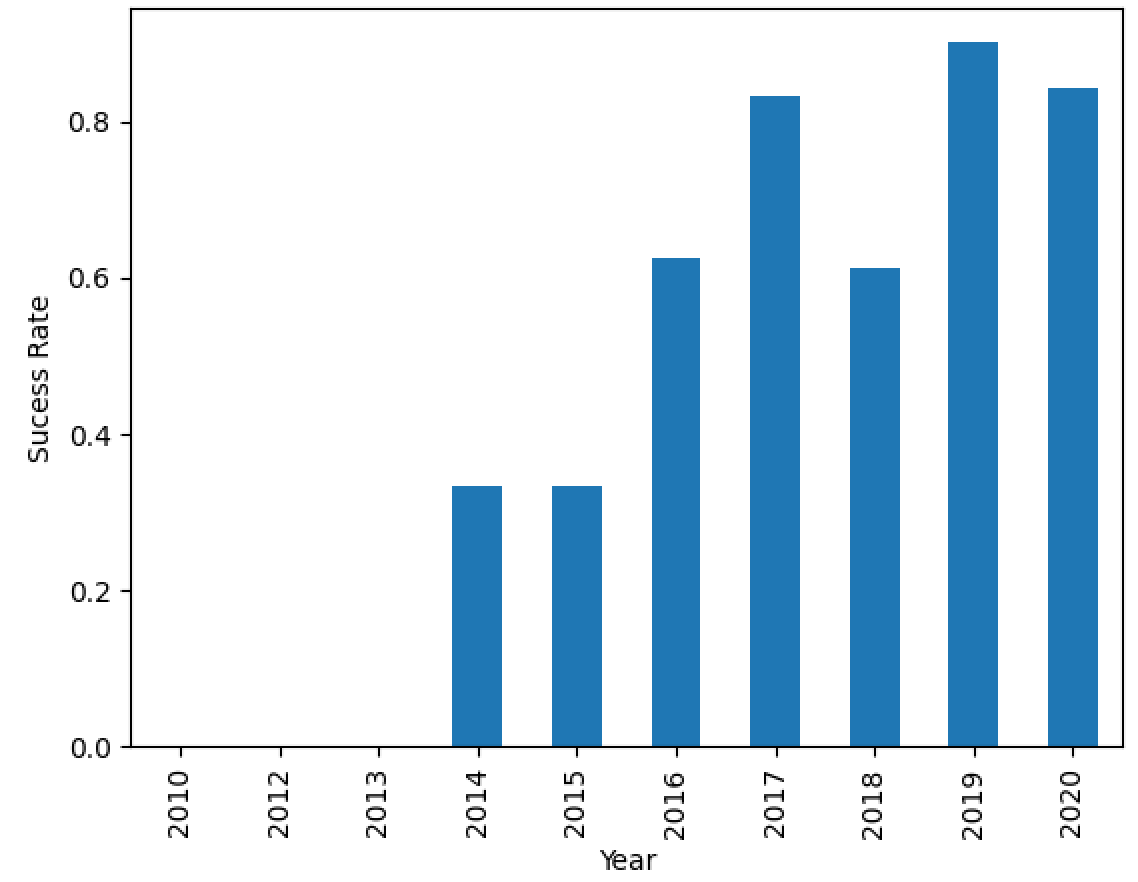
Payload vs. Orbit Type

- Some orbits have either heavy payloads or light payloads.
- Example: VLEO orbits has heavy payloads while SSO orbit has light payloads.



Launch Success Yearly Trend

- Succes rate generally kept increasing with the years.



All Launch Site Names

- The query selects the name of launch site with the keyword `distinct` to get the names once.

```
In [7]: %%sql
        select distinct Launch_Site
          from SPACEXTBL

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

Out[7]:

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

Launch Site Names Begin with 'CCA'

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

In [8]:

```
%%sql
select *
  from SPACEXTBL
where Launch_Site LIKE '%CCA%'
limit 5;
```

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

Out[8]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

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

In [22]:
```
%%sql
select customer, sum(PAYLOAD_MASS__KG_)
from SPACEXTBL
where customer like '%NASA (CRS)';
```

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

Out[22]:

Customer	sum(PAYLOAD_MASS__KG_)
NASA (CRS)	45596

Average Payload Mass by F9 v1.1

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

In [10]:
```
%%sql
select Booster_Version, avg(PAYLOAD_MASS__KG_)
from SPACEXTBL
WHERE Booster_Version like '%F9 v1.1%';
```

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

Out[10]:

Booster_Version	avg(PAYLOAD_MASS__KG_)
F9 v1.1 B1003	2534.6666666666665

First Successful Ground Landing Date

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

Hint:Use min function

In [11]:
```
%%sql
select *
from SPACEXTBL
where "Landing _Outcome" like '%Success (ground pad)%'
AND (substr(Date,7,4) || substr(Date,4,2) || substr(Date,1,2)) = (
  select min(substr(Date,7,4) || substr(Date,4,2) || substr(Date,1,2))
  from SPACEXTBL
  where "Landing _Outcome" like '%Success (ground pad)%'
)
```

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

Out[11]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
22-12-2015	01:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

First successful ground landing was on 22/12/2015

Successful Drone Ship Landing with Payload between 4000 and 6000

In [23]:
```
%%sql
select Booster_Version
from SPACEXTBL
where "Landing _Outcome" like '%Success (drone ship)%'
and 4000 <= PAYLOAD_MASS__KG_
and PAYLOAD_MASS__KG_ <= 6000;
```

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

Out[23]:

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

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

In [16]:
```
%%sql
select "Mission_Outcome", count(*)
from SPACEXTBL
GROUP BY "Mission_Outcome"
```

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

Out[16]:

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

Boosters Carried Maximum Payload

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

In [17]:
```
%%sql
select  distinct(Booster_Version), PAYLOAD_MASS__KG_
from SPACEXTBL
where "PAYLOAD_MASS__KG_" = (
  select max(PAYLOAD_MASS__KG_)
  from SPACEXTBL
)
```

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

Out[17]:

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

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

In [18]:

```
%%sql
select substr(Date, 4, 2) as month, Booster_Version, "Landing _Outcome"
from SPACEXTBL
where substr(Date,7,4)='2015'
and "Landing _Outcome" like '%Failure (drone ship)%'
```

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

Out[18]:

month	Booster_Version	Landing _Outcome
01	F9 v1.1 B1012	Failure (drone ship)
04	F9 v1.1 B1015	Failure (drone ship)

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

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

In [19]:

```
%%sql
select count(*)
from SPACEXTBL
WHERE "Mission_Outcome" like '%Success%'
and substr(Date,7,4) || substr(Date,4,2) || substr(Date,1,2) >= '20100604'
and substr(Date,7,4) || substr(Date,4,2) || substr(Date,1,2) <= '20170320'
```

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

Out[19]:

count(*)
30

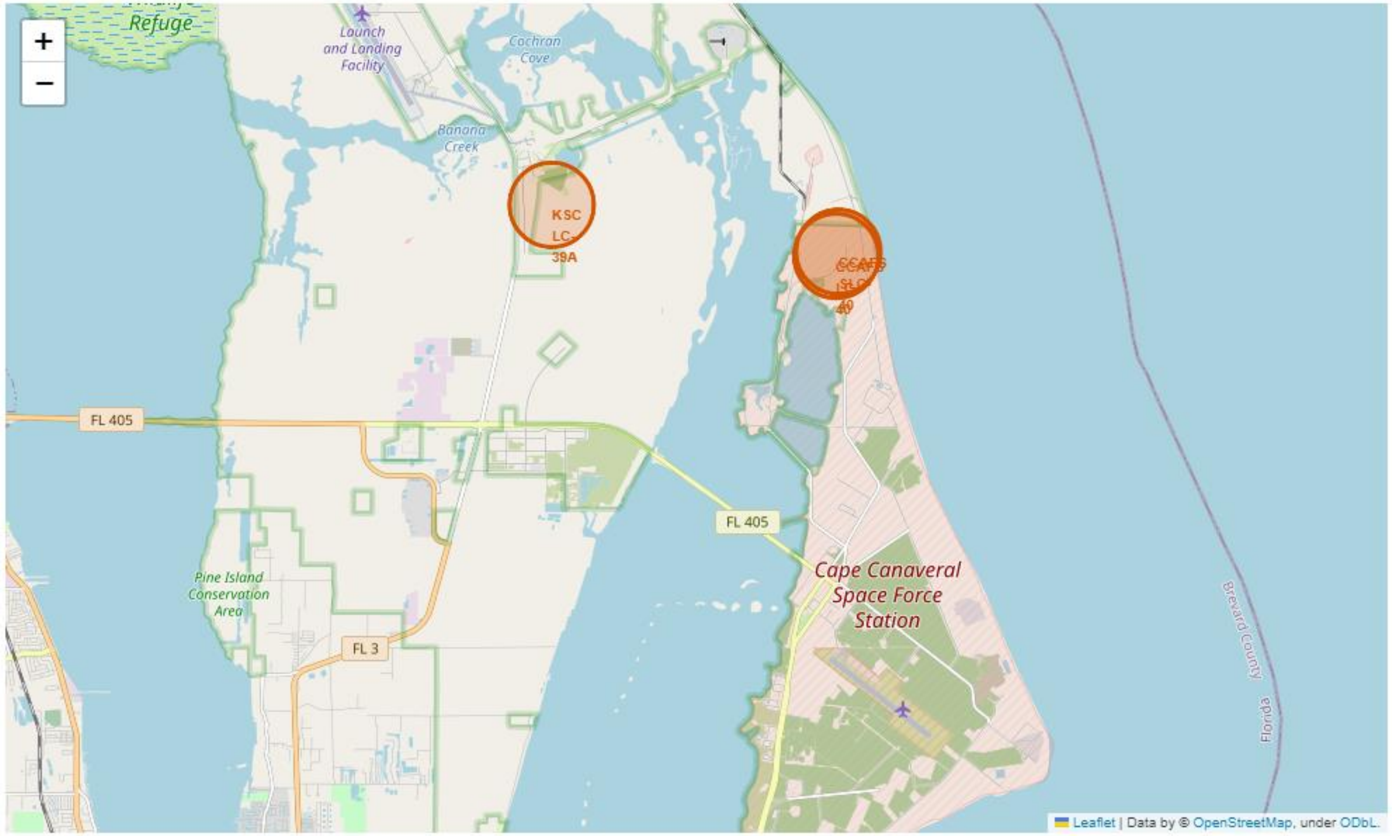
Section 3

Launch Sites Proximities Analysis

Launch site marked on a map

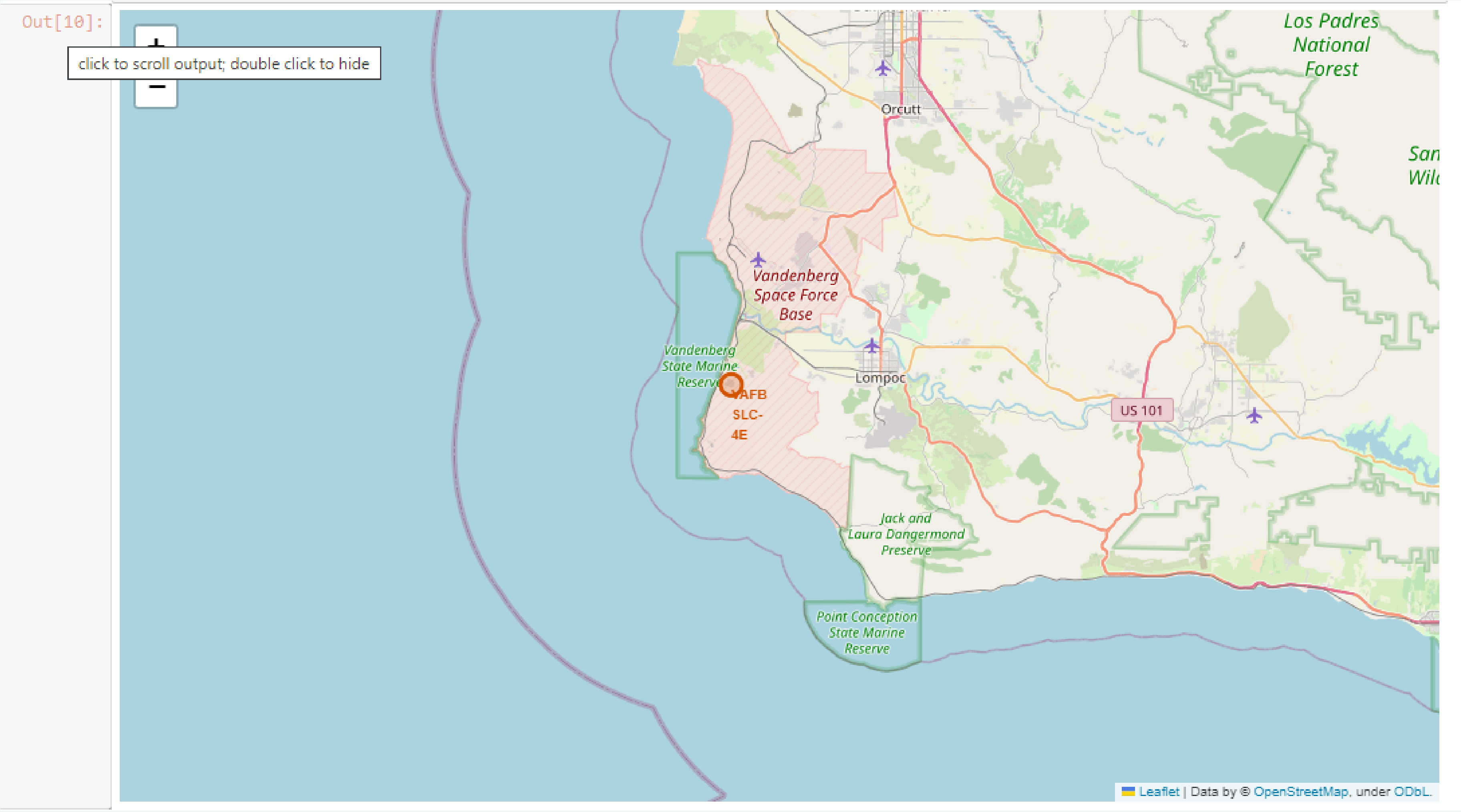
- 3 launch sites on the east coast

Out[10]:



Launch site marked on a map

- 1 launch site on the west coast



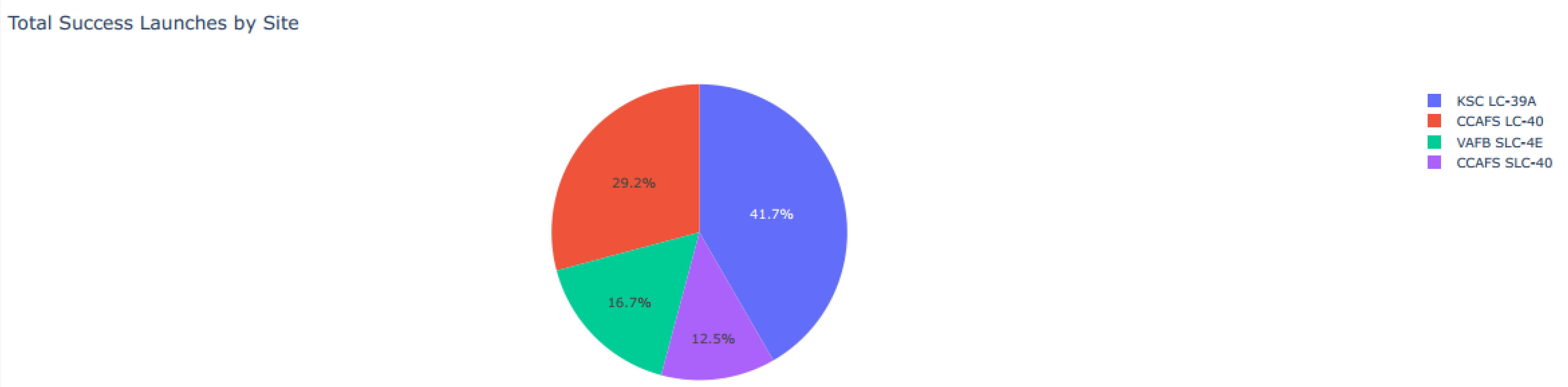
Launch sites, Explanation

- The launch sites are close to the equator line since it requires less energy to send a rocket to space the closer the launch site is to the equator.
- The launch site are near the coast for safety reasons.
- Launch sites are close to railways in order to be able to transport cargo needed to the site.
- Launch sites are close to highways in order to be able to transport personnel
- Launch sites are far from cities for safety reasons.

Section 4

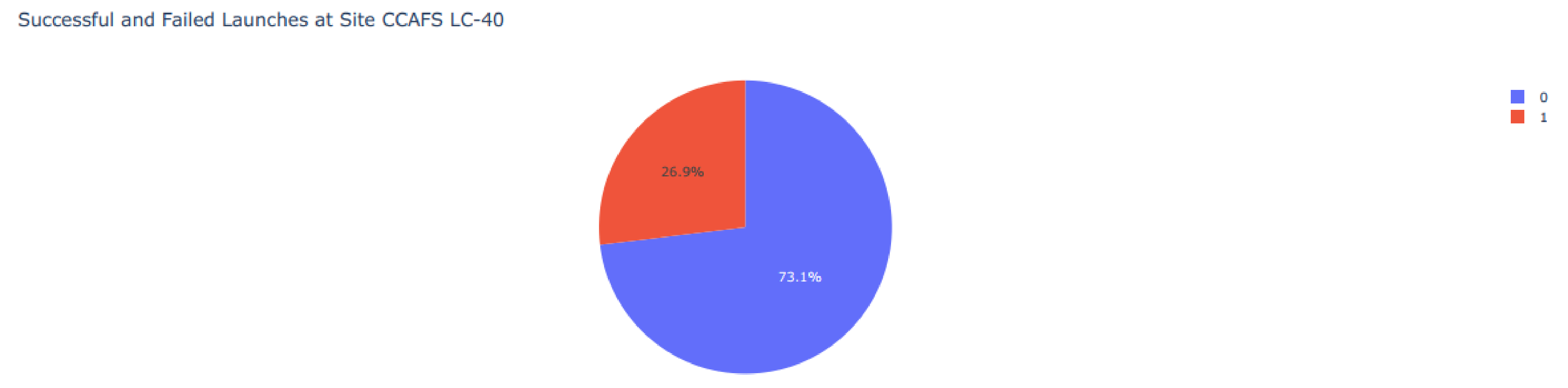
Build a Dashboard with Plotly Dash

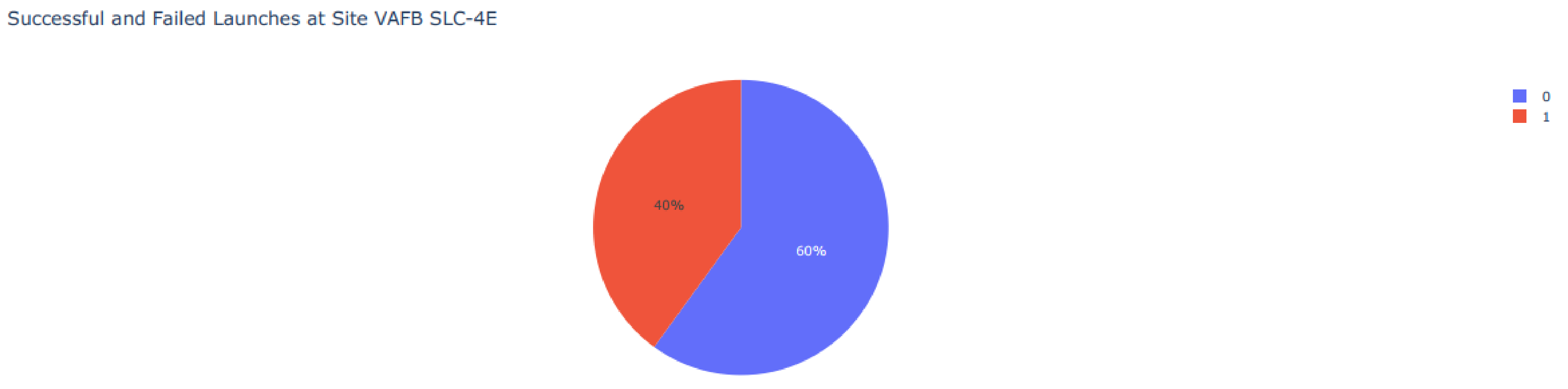
Success launches among launch sites



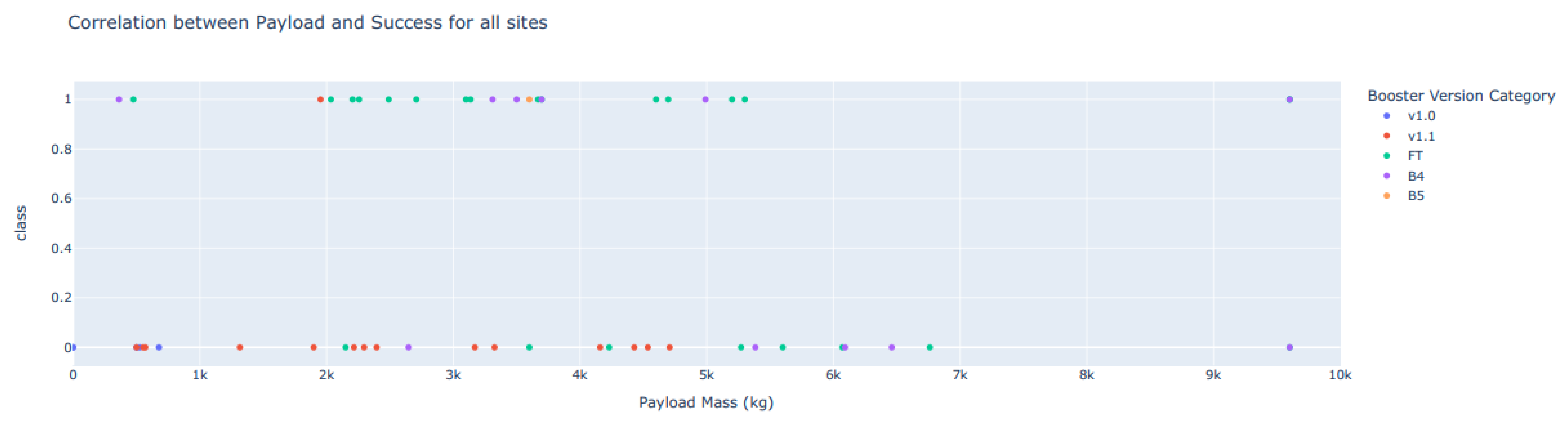
Most launch sites happened in KSC LC-39A

Success / Failure by launch site





Launch outcome Vs. Payload



Section 5

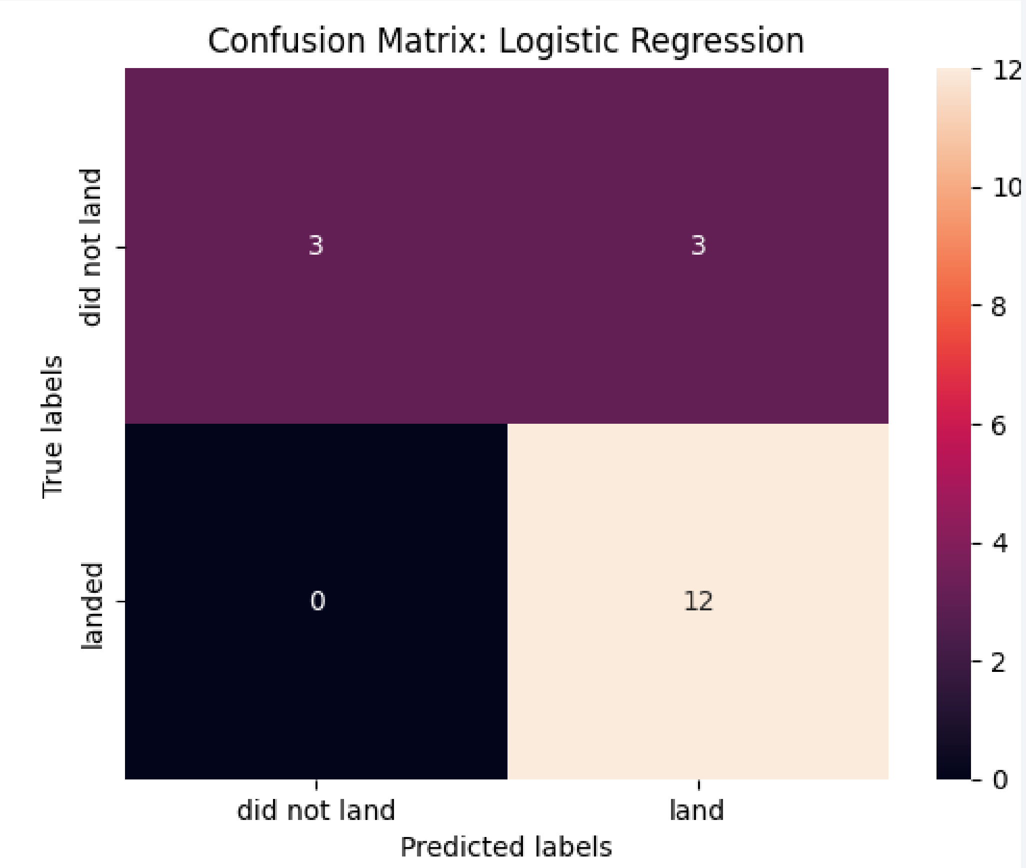
Predictive Analysis (Classification)

Classification Accuracy

```
In [35]: print('Accuracy for Logistics Regression method:', logreg_cv.score(X_test, Y_test))
         print( 'Accuracy for Support Vector Machine method:', svm_cv.score(X_test, Y_test))
         print('Accuracy for Decision tree method:', tree_cv.score(X_test, Y_test))
         print('Accuracy for K nearsdt neighbors method:', knn_cv.score(X_test, Y_test))

         Accuracy for Logistics Regression method: 0.8333333333333334
         Accuracy for Support Vector Machine method: 0.8333333333333334
         Accuracy for Decision tree method: 0.7222222222222222
         Accuracy for K nearsdt neighbors method: 0.8333333333333334
```

Confusion Matrix



- The highest accuracy achieved among all models was 83%

	precision	recall	f1-score	support
0	1.00	0.50	0.67	6
1	0.80	1.00	0.89	12
accuracy			0.83	18
macro avg	0.90	0.75	0.78	18
weighted avg	0.87	0.83	0.81	18

Conclusions

- SVM, KNN and logistic regression have performed the best in predicting the outcome of a launch. The decision trees performed the worst.
- The success rate increased with the years. The payload also increased with the years.
- Orbits GEO, HEO, SSO and ES L1 have the best success rate of launching.

Appendix

- All Jupyter notebooks, Python scripts and CSV files are in the following GitHub repository:
- https://github.com/anasLearn/data-science-capstone

Thank you!