IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Andy M. Méndez
30-Nov-2022

Outline

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

Executive Summary

- **SUMMARY OF METHODOLOGIES**
  - Data Collection
  - Data Wrangling
  - Exploratory Analysis using SQL
  - Exploratory Analysis using Pandas and Matplotlib
  - Interactive Visual Analytics with Folium lab
  - Interactive Visual Analytics with Ploty Dash
  - Predictive Analysis (Classification)
- **SUMMARY OF ALL RESULTS**
  - Exploratory Analysis
  - Interactive Visual Analytics
  - Predictive Analytics

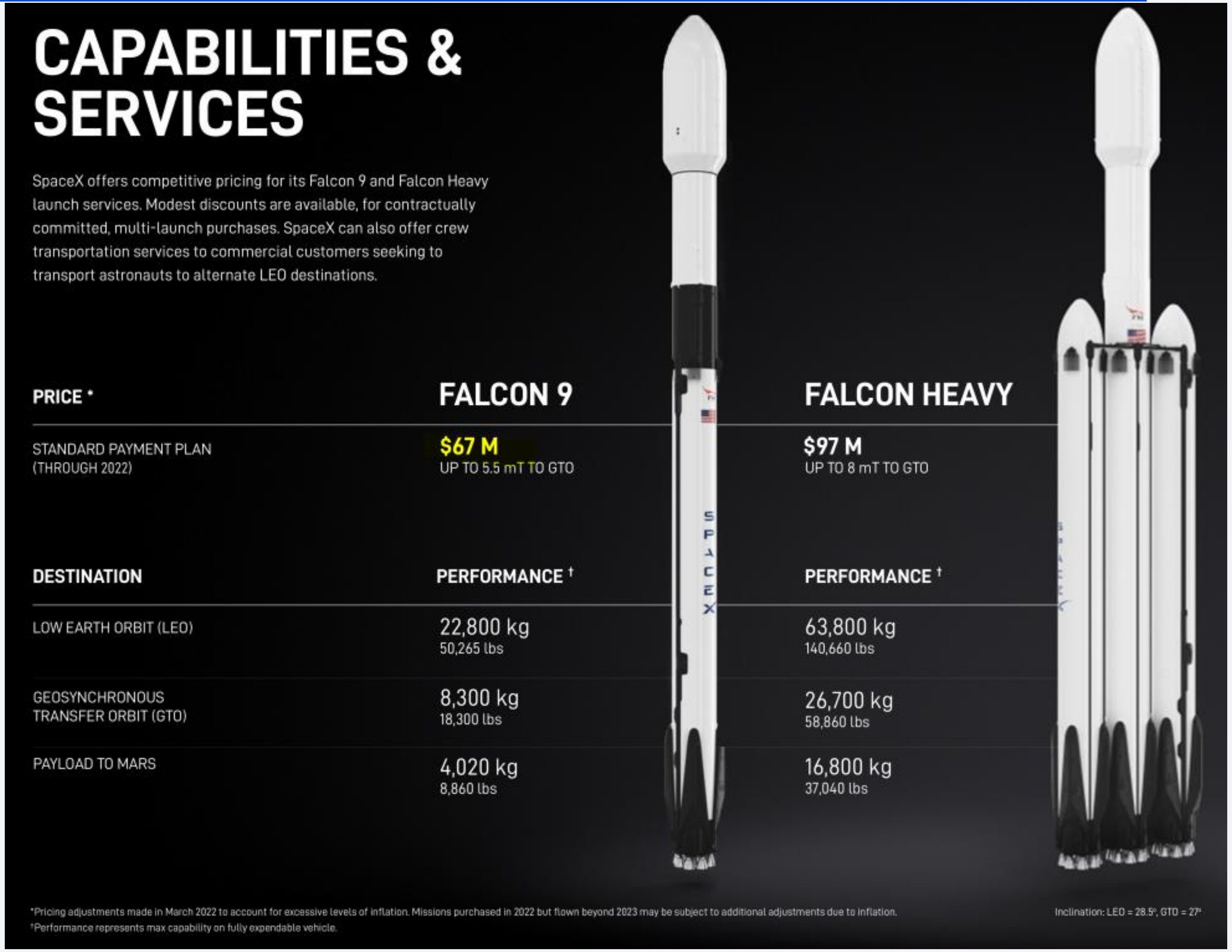
Introduction

- Project background and context

SpaceX advertises Falcon 9 rocket launches cost 62 million dollars(pre-inflation); other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.

- Problems you want to find answers

In this project will predict if the Falcon 9 first stage will land successfully defining the accuracy of the landing.



https://www.spacex.com/vehicles/falcon-9/

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX Rest API   (“ spacex_url=https://api.spacexdata.com/v4/launches/past)
  - Web Scrapping
- Perform data wrangling
  - The information collected incorporates data from both successful and unsuccessful launches, at this point the data is labeled prior to applying the machine learning models.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Training and test data to find the best Hyperparameter for SVM, Classification Trees, and Logistic Regression.

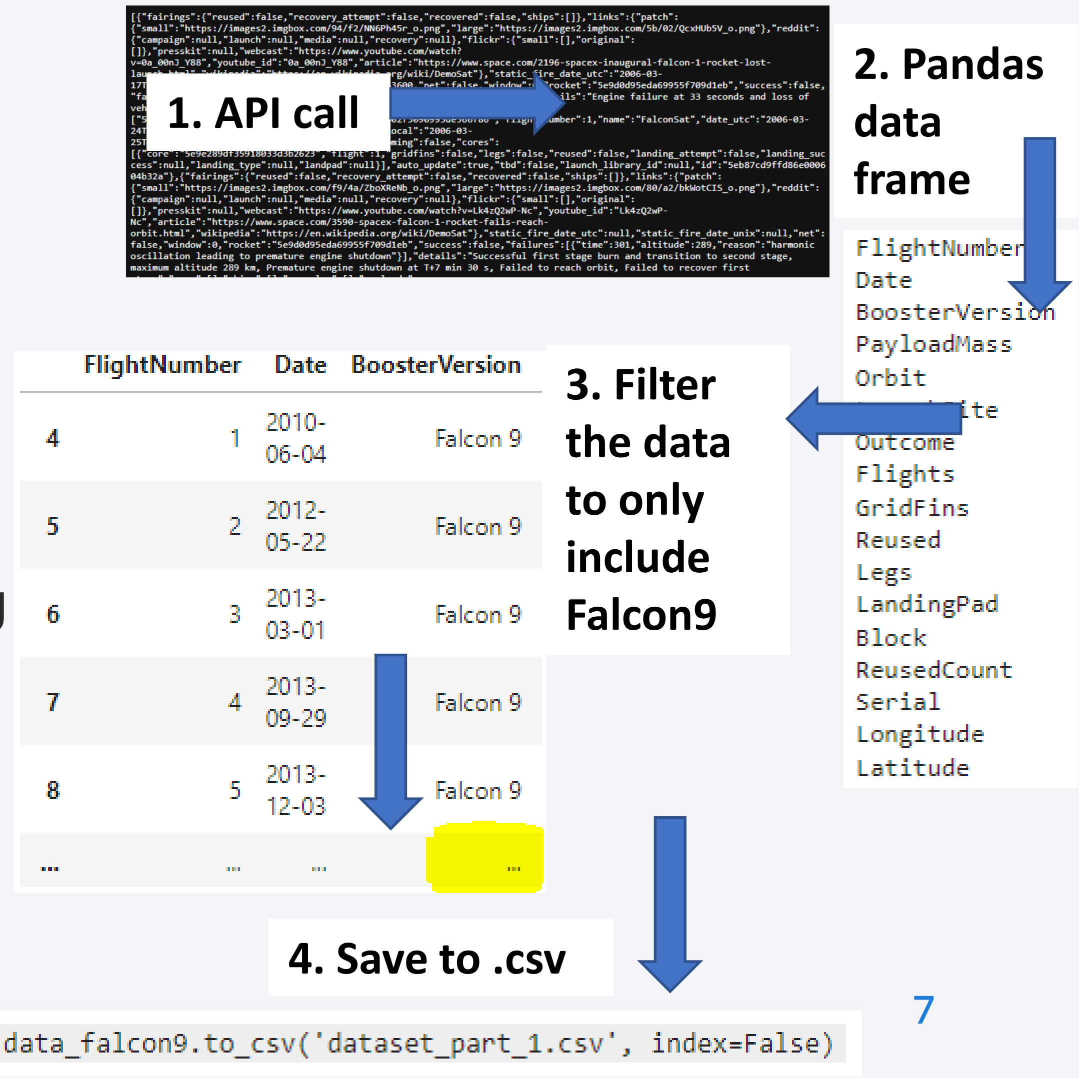
Data Collection

Describe how data sets were collected.

1. SpaceX launch data is collected from the REST API (https://api.spacexdata.com/v4/launches/past)

2. The REST API include enough information referring

   FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude



Data Collection – SpaceX API

1. Data Collection

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

response = requests.get(spacex_url)

print(response.content)
```

b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.com/94/f2/NN6Ph45r_o.png","large":"https://images2.imgbox.com/5b/02/QcxHUb5V_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.space.com/2196-spacex-inaugural-falcon-1-rocket-lost-launch.html","wikipedia":"https://en.wikipedia.org/wiki/DemoSat"},"static_fire_date_utc":"2006-03-17T00:00:00.000Z","static_fire_date_unix":1142553600,"net":false,"window":0,"rocket":"5e9d0d95eda69955f709d1eb","success":false,"failures":[{"time":33,"altitude":null,"reason":"merlin engine failure"}],"details":"Engine failure at 33 seconds and loss of vehicle","crew":[],"ships":[],"capsules":[],"payloads":["5eb0e4b5b6c3bb0006eeb1e1"],"launchpad":"5e9e4502f5090995de566f86","flight_number":1,"name":"FalconSat","date_utc":"2006-03-24T22:30:00.000Z","date_unix":1143239400,"date_local":"2006-03-25T10:30:00+12:00","date_precision":"hour","upcoming":false,"cores":[{"core":"5e9e289df35918033d3b2623","flight":1,"gridfins":false,"legs":false,"reused":false,"landing_attemp

2. Convert the .json() result into a dataframe

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

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	capsules	payloads	
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	[]	[]	[]	[5eb0e4b5b6c3bb0006eeb1e1]	5e9e45

3. First Data Processing

```python
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]
```

```python
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that hav
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

4. Filter to only keep the features needed for Falcon 9 Launches

```python
data_falcon9=df[df['BoosterVersion']!='Falcon 1']
```

	FlightNumber	Date	BoosterVersion
4	1	2010-06-04	Falcon 9
5	2	2012-05-22	Falcon 9
6	3	2013-03-01	Falcon 9
7	4	2013-09-29	Falcon 9
8	5	2013-12-03	Falcon 9
...	...	...	...
89	86	2020-09-03	Falcon 9
90	87	2020-10-06	Falcon 9
91	88	2020-10-18	Falcon 9
92	89	2020-10-24	Falcon 9
93	90	2020-11-05	Falcon 9

90 rows × 17 columns

GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/62f406d4d6ac4f466aa89e0fcafbe7c26f1e7bfa/jupyter-labs-spacex-data-collection-api%20(Lab1).ipynb

Data Collection - Scraping

1. Requests.get method

```
# use requests.get() method with the provided static_url
page = requests.get(static_url)
```

2. Create BeautifulSoup Object

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

3. Assign the result to a list (html_tables)

```
# 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')
```

4. Get column names

```
# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

column_names = []
table_headers = first_launch_table.find_all('th')
# print(table_headers)
for j, table_header in enumerate(table_headers):
    name = extract_column_from_header(table_header)
    if name is not None and len(name) > 0:
        column_names.append(name)
print(column_names)
```

```
['Flight No.', 'Date and time ( )', 'Launch site', 'Payload', 'Payload mass', 'Orbit', 'Customer', 'Launch outcome']
```

5. Create an empty dictionary and convert into a Pandas dataframe

```
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 be an empty list
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']=[]
```

6. Export to CSV

```
df.to_csv('spacex_web_scraped.csv', index=False)
```


GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/07a1d8b5e3b5f29b1179559fc6b4a37b885a7582/jupyter-labs-webscraping(lab1.2).ipynb

Data Wrangling

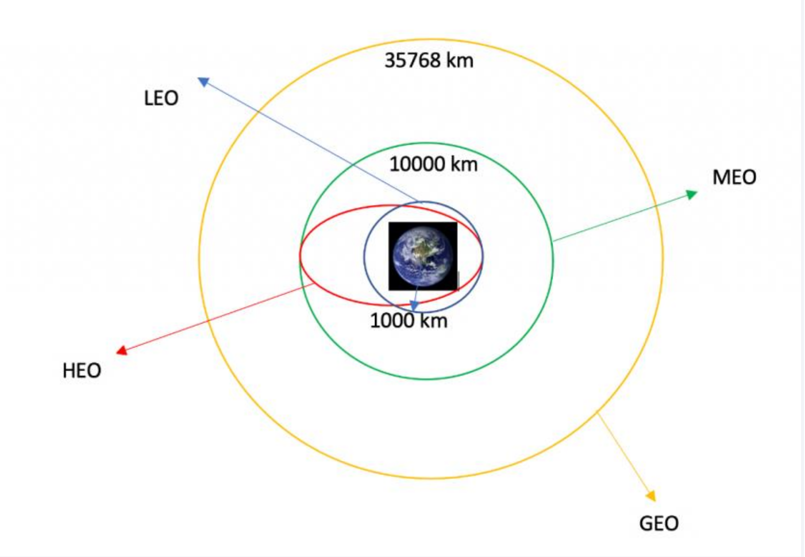
1. Calculate the number of launches on each site

```
# Apply value_counts() on column LaunchSite
df.LaunchSite.value_counts()
```

2. Calculate the number and occurrence of each orbit

```
# Apply value_counts on Orbit column
df.Orbit.value_counts()
```

GTO	27
ISS	21
VLEO	14
PO	9
LEO	7
SSO	5
MEO	3
ES-L1	1
HEO	1
SO	1
GEO	1



3. Calculate the number and occurrence of each orbit

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

True ASDS	41
None None	19
True RTLS	14
False ASDS	6
True Ocean	5
False Ocean	2
None ASDS	2
False RTLS	1

4. Create a landing outcome label from outcome column

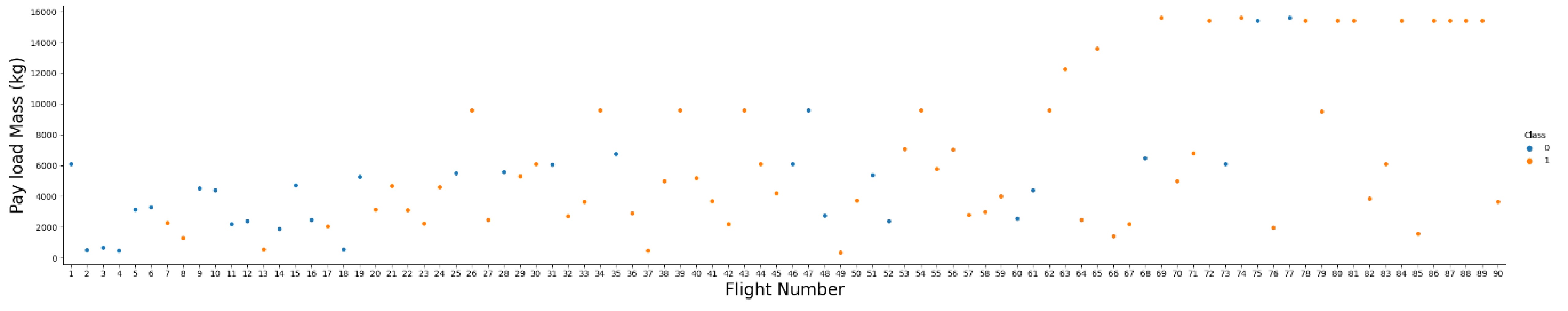
```
# landing_class = 0 if bad_outcome
landing_class = [0 if x in bad_outcomes else 1 for x in df['Outcome']]
df['Class']=landing_class
print(df[['Class']].head(8))
print(df["Class"].mean())  # probability of positive outcome 2/3
print(df.head(5))

# landing_class = 1 otherwise
```

5. Export to CSV

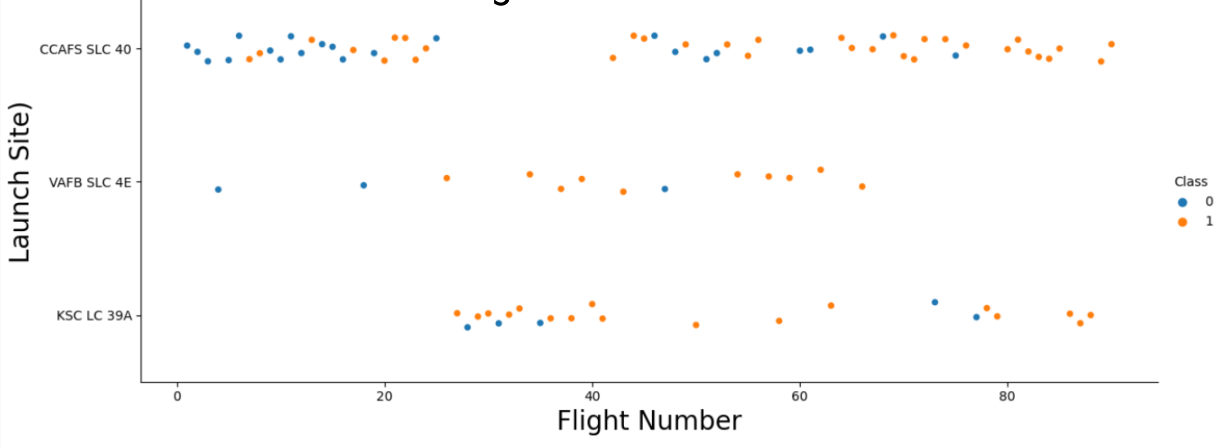
GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/07a1d8b5e3b5f29b1179559fc6b4a37b885a7582/labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite%20(Lab2).ipynb

EDA with Data Visualization

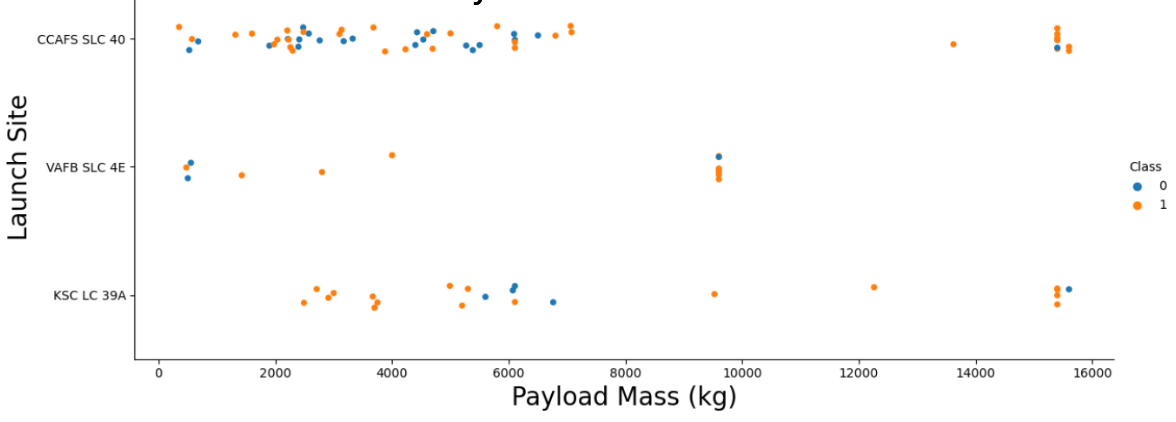
According to the Flight numbers are increasing the Payload Mass (kg)



Relation between Flight number and Launch Site



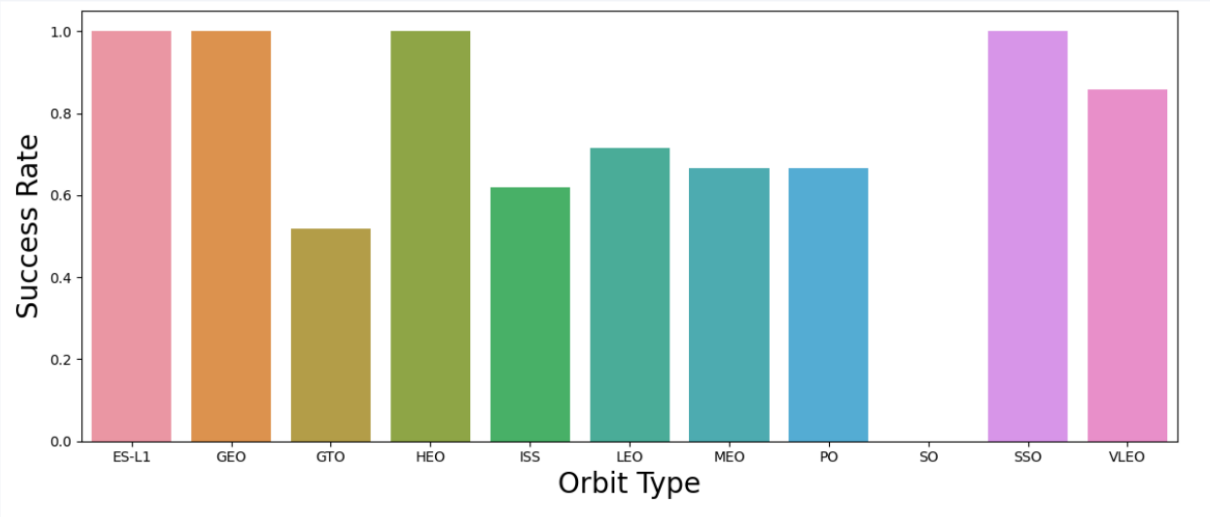
Relation between Payload Mass and Launch Site

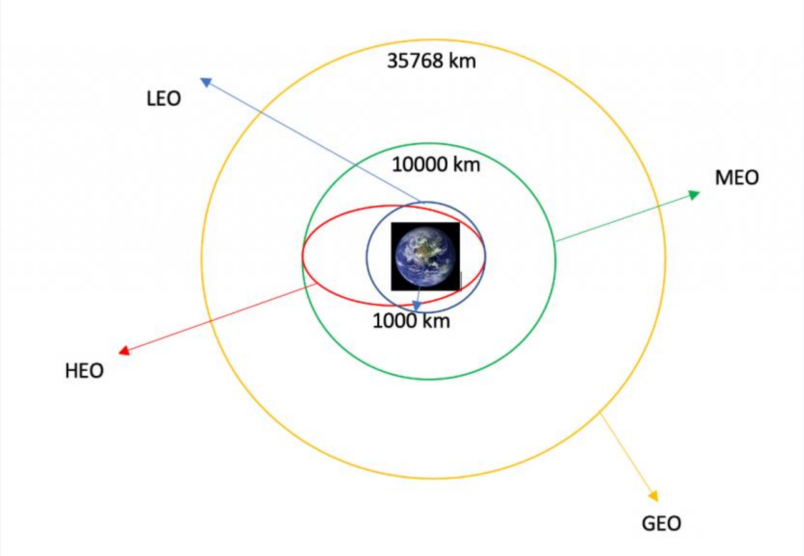


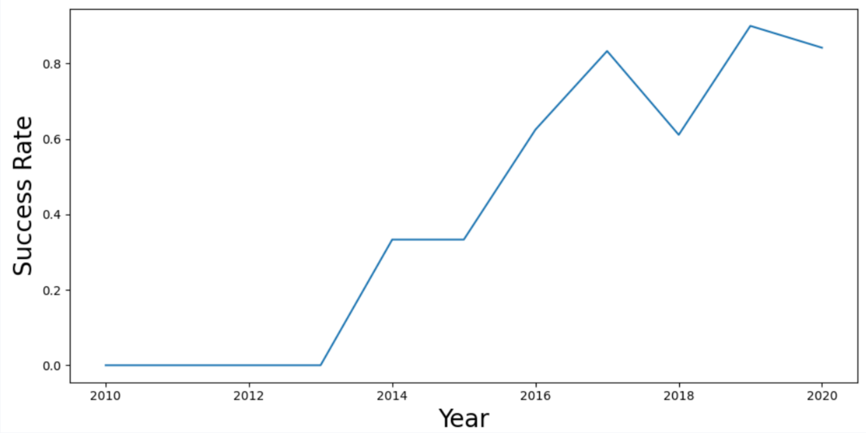
GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite(lab5).ipynb

EDA with Data Visualization

Relation between Success Rate according to each orbit







Launch success yearly trend
– increasing remarkably -

GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite(lab5).ipynb

EDA with SQL

Display the names of the unique launch sites in the space mission

```
conn = sqlite3.connect(':memory:')  # in memory database
df.to_sql(name="spacexdata", con=conn, if_exists="replace")
```

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

```
q = pd.read_sql("select * from spacexdata where Launch_Site like 'CCA%' limit 5", conn)
```

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

```
q = pd.read_sql("select sum(PAYLOAD_MASS__KG_) from spacexdata where Customer='NASA (CRS)'", conn)
```

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

```
q = pd.read_sql("select avg(PAYLOAD_MASS__KG_) from spacexdata where Booster_Version='F9 v1.1'", conn)
```

List the date when the first successful landing outcome in ground pad was achieved.

```
q = pd.read_sql("select min(Date) from spacexdata where Landing__Outcome='Success (ground pad)'", conn)
```

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
q = pd.read_sql("select distinct Booster_Version from spacexdata where Landing__Outcome='Success (drone ship)'
```

List the total number of successful and failure mission outcomes

```
q = pd.read_sql("select substr(Mission_Outcome,1,7) as Mission_Outcome, count(*) from spacexdata  group by 1", conn)
```

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

```
q = pd.read_sql("select distinct Booster_Version from spacexdata where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from spacexdata)", conn)
```

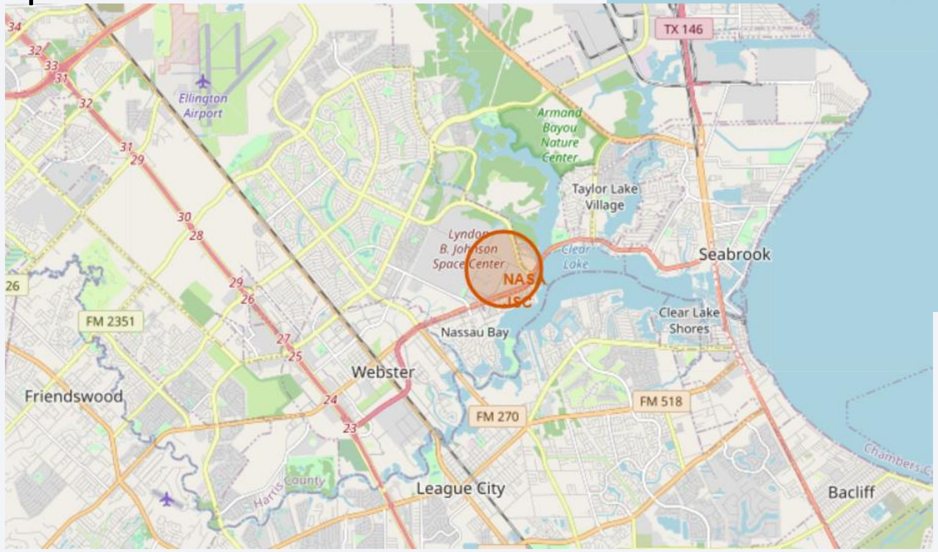
List the records which will display the month names, failure landing outcomes in drone ship ,booster versions, launch site for the months in year 2015.

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

GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/07a1d8b5e3b5f29b1179559fc6b4a37b885a7582/jupyter-labs-eda-sql-coursera_sqllite%20(lab4).ipynb

Build an Interactive Map with Folium

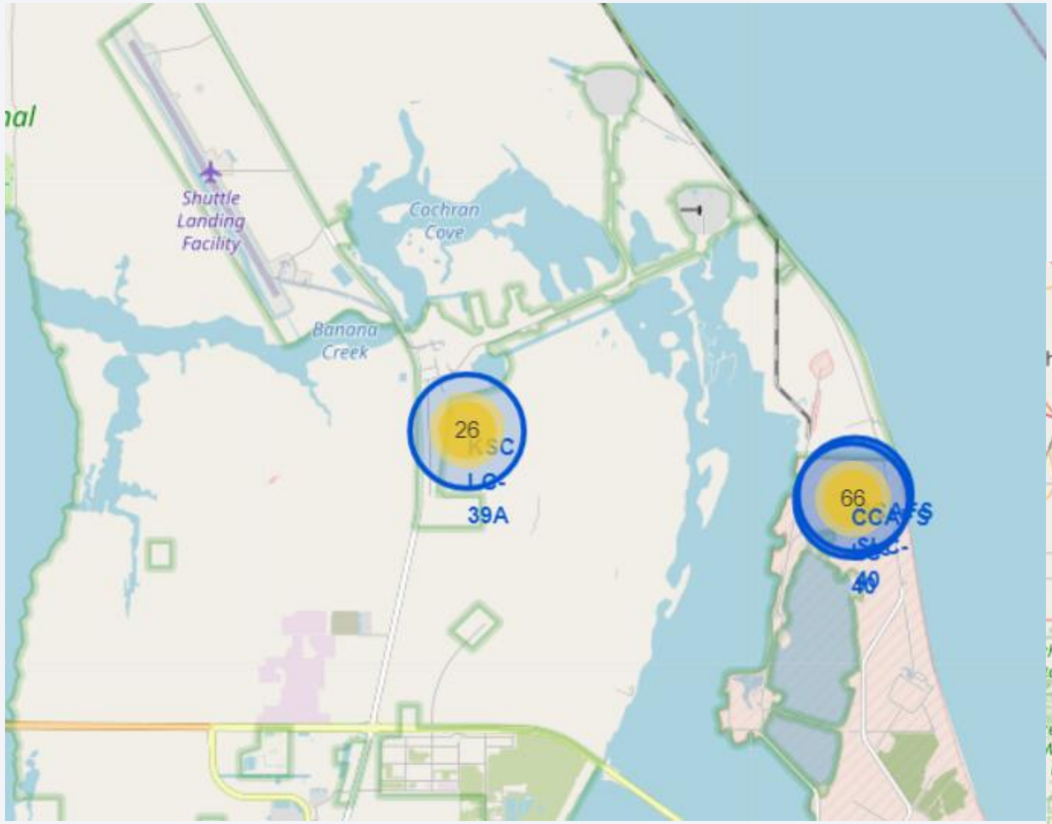
Blue circle at NASA Johnson Space Center's coordinate



Coordinate (Lat, Long) for a mouse over on the map

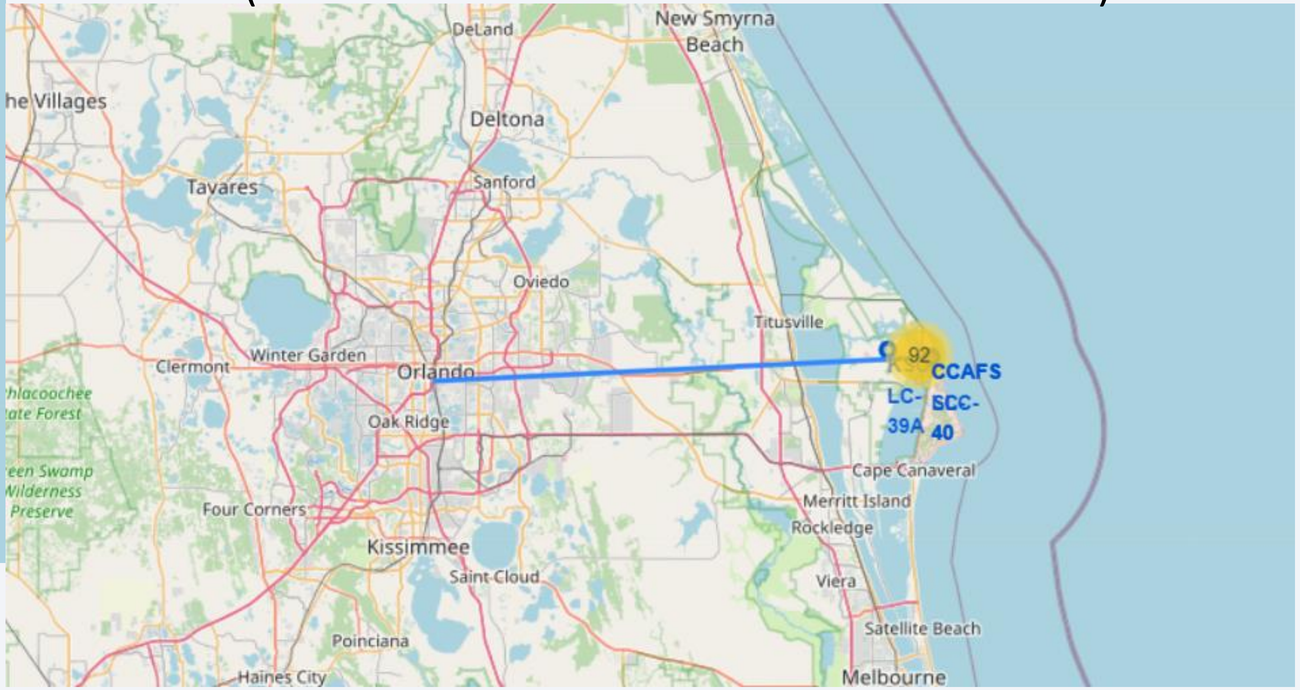
GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/132fd31036d0e7176fa171082042dfccf7c60e80/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

Build an Interactive Map with Folium



`folium.PolyLine` object using the coastline coordinates and launch site coordinate

Marker with distance to a closest city.
(Line between the marker to the launch site)

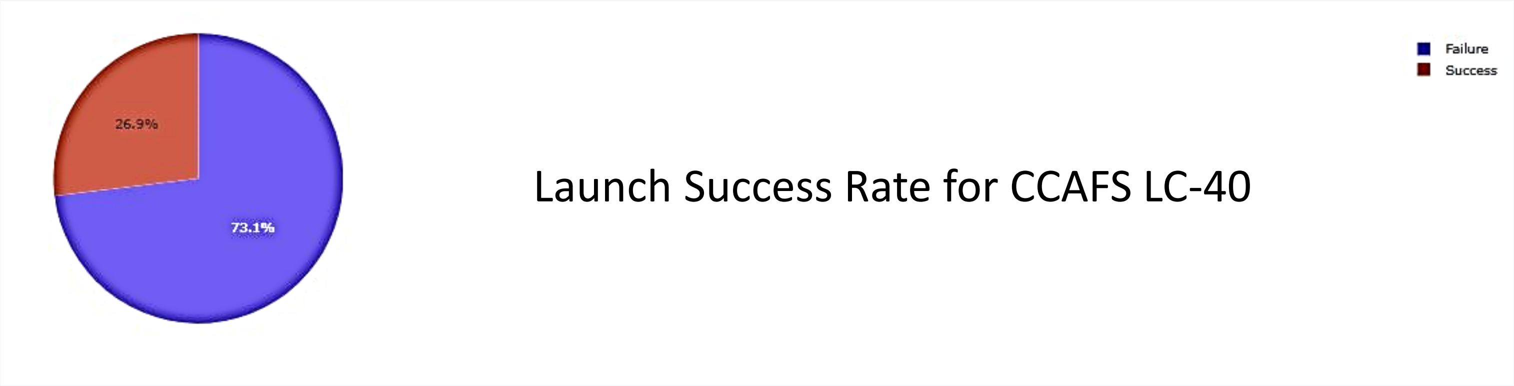


GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/132fd31036d0e7176fa171082042dfccf7c60e80/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

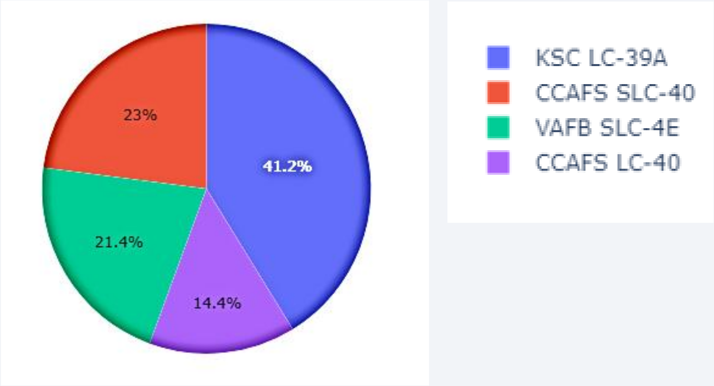
Build a Dashboard with Plotly Dash

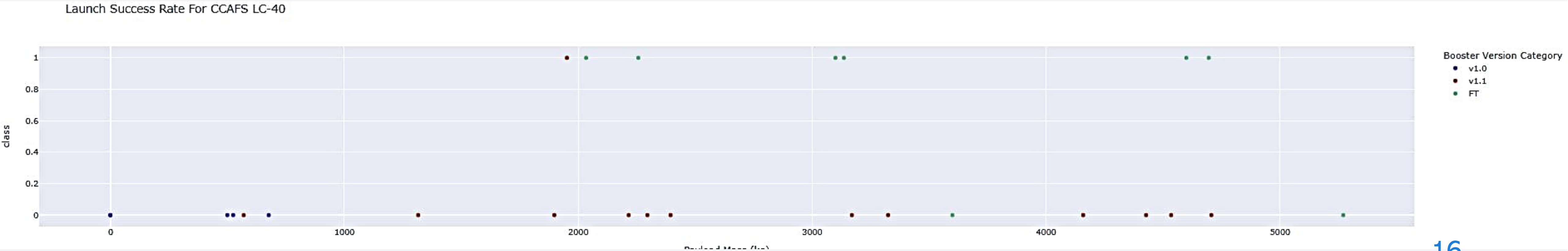
SpaceX Launch Records Dashboard

Cape Canaveral Launch Complex 40 (CAFS LC-40)



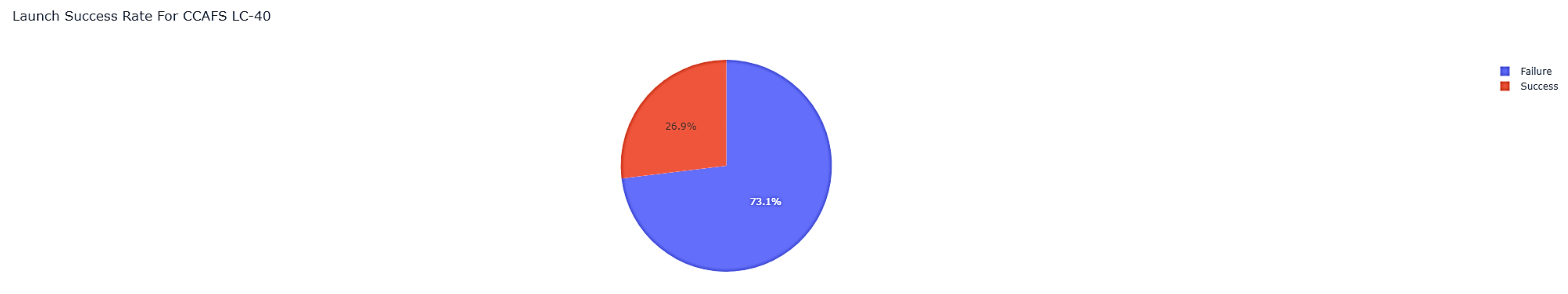
Launch Success Rate For All Sites

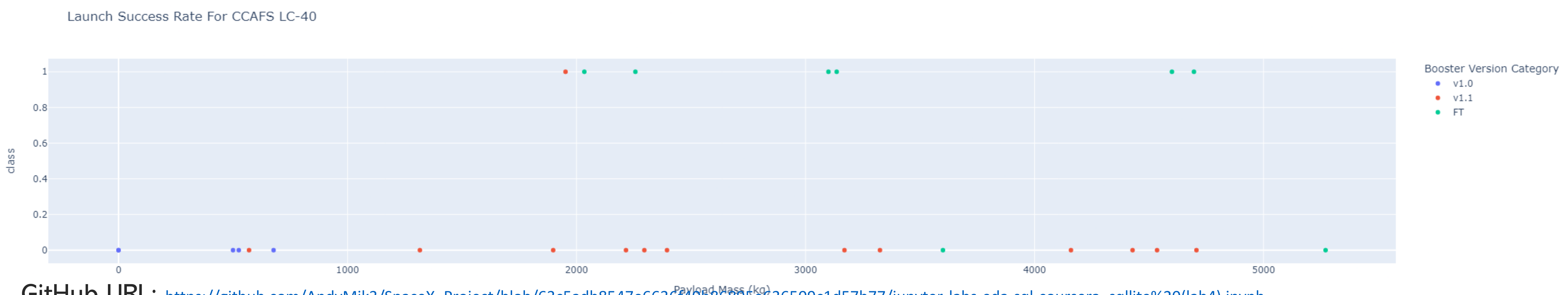




GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/63c5adb8547e6636f40b86895e636509c1d57b77/jupyter-labs-eda-sql-coursera_sqllite%20(lab4).ipynb

Build a Dashboard with Plotly Dash





GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/63c5adb8547e6636f40b86895e636509c1d57b77/jupyter-labs-eda-sql-coursera_sqllite%20(lab4).ipynb

Predictive Analysis (Classification)

	method	accuracy
0	Logistic regression	0.833333
1	Support vector machine	0.833333
2	Decision tree classifier	0.777778
3	K nearest neighbors	0.833333

1. Create a column for the Class
2. Standardize the data
3. Split into training data and test data
4. Find best Hyperparameter for SVM, Classification Trees and Logistic Regression
5. Find the method that performs best using test data

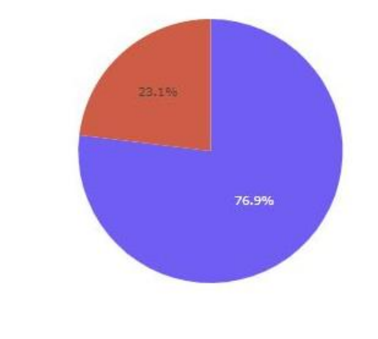
GitHub URL: https://github.com/AndyMik3/SpaceX_Project/blob/63c5adb8547e6636f40b86895e636509c1d57b77/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

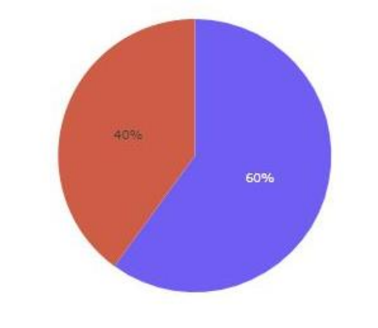
The Support vector machine, logistic regression and K-nearest neighbors share the same value of accuracy

	method	accuracy
0	Logistic regression	0.833333
1	Support vector machine	0.833333
2	Decision tree classifier	0.777778
3	K nearest neighbors	0.833333

The success rate for KSC LC-39A was 76.9%



The success rate for VAFB SLC-4E was 60%

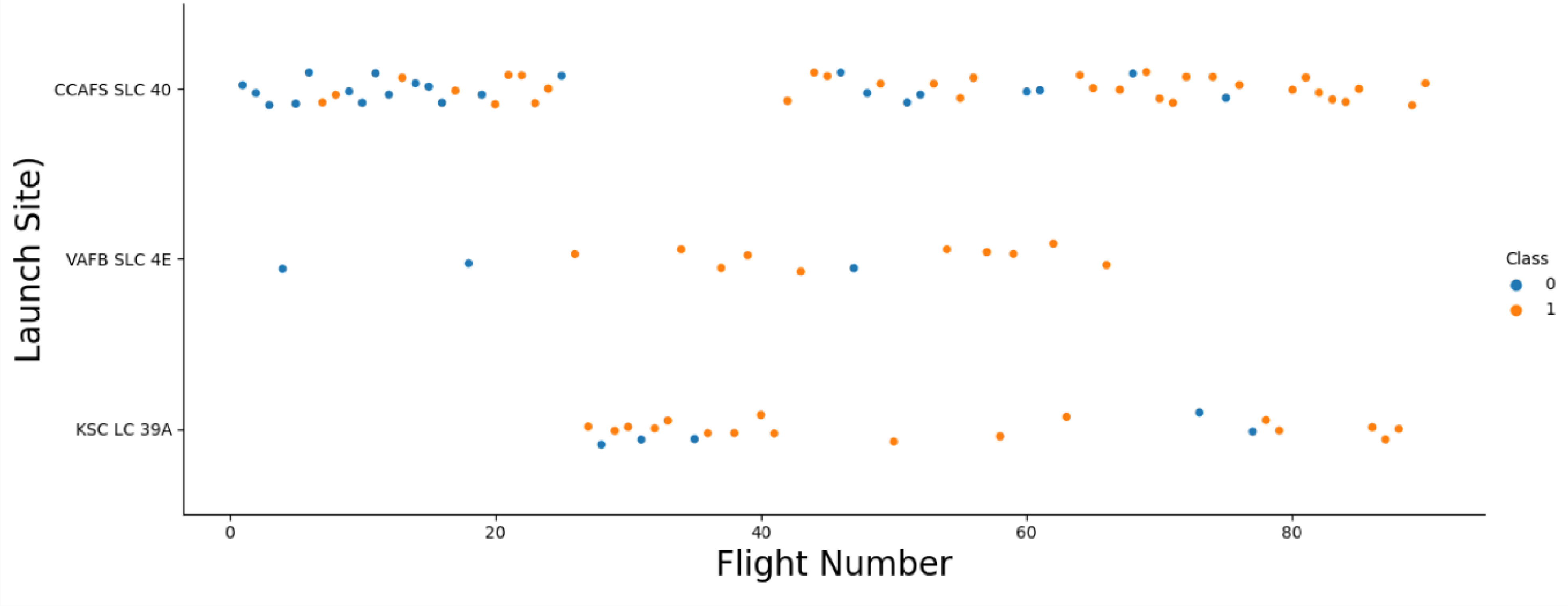


Section 2

Insights drawn from EDA

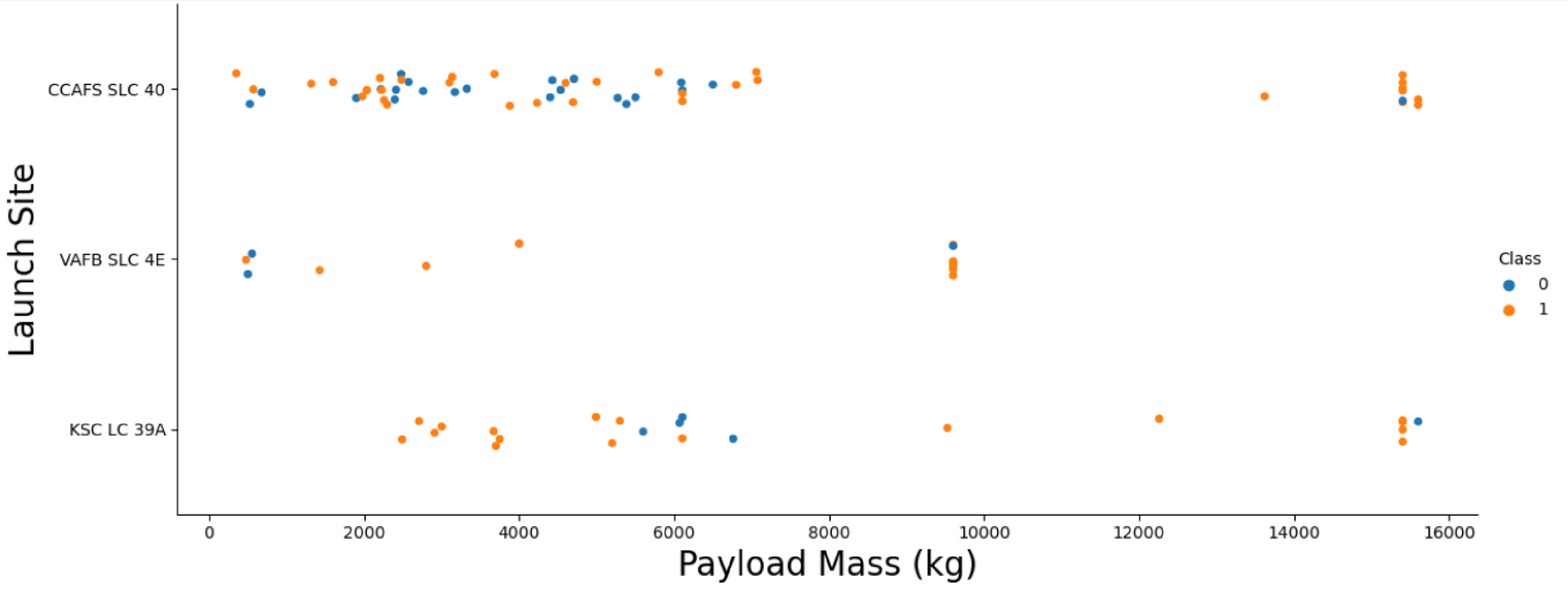
Flight Number vs. Launch Site

According to the flight number the CCAFS SLC 40 is notably the launch site most used

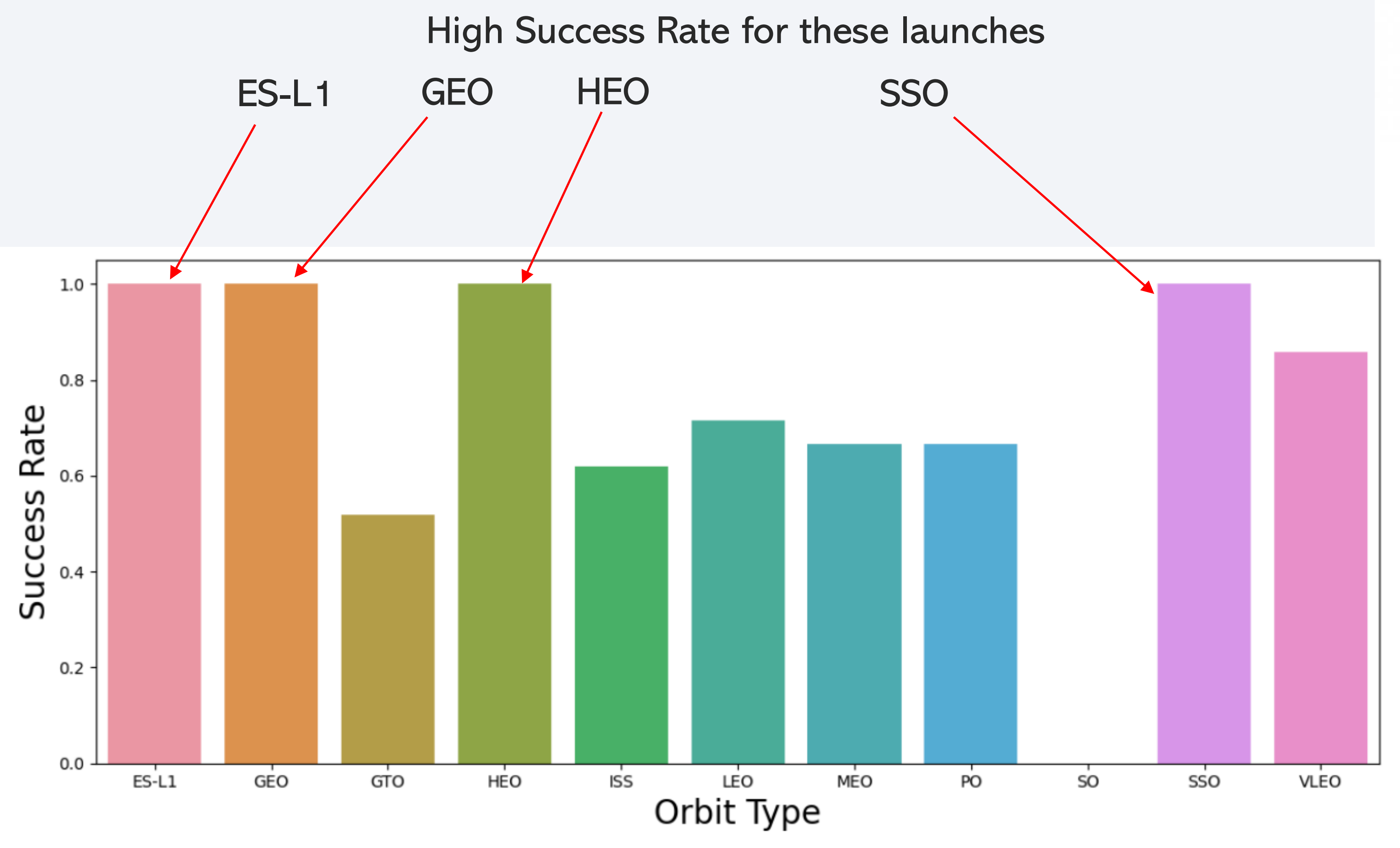


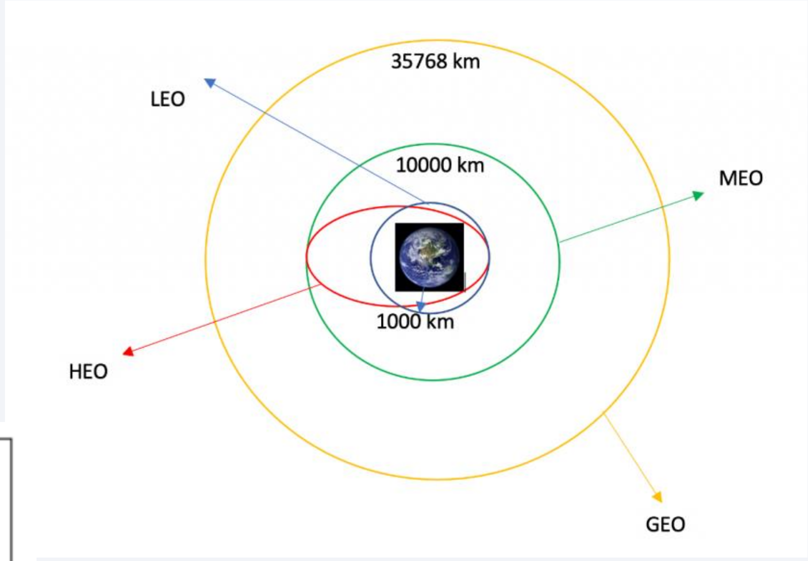
Payload vs. Launch Site

The payload mass is sent with a mass between 1000kg to 8000kg with few very specific launches close to 15000kg

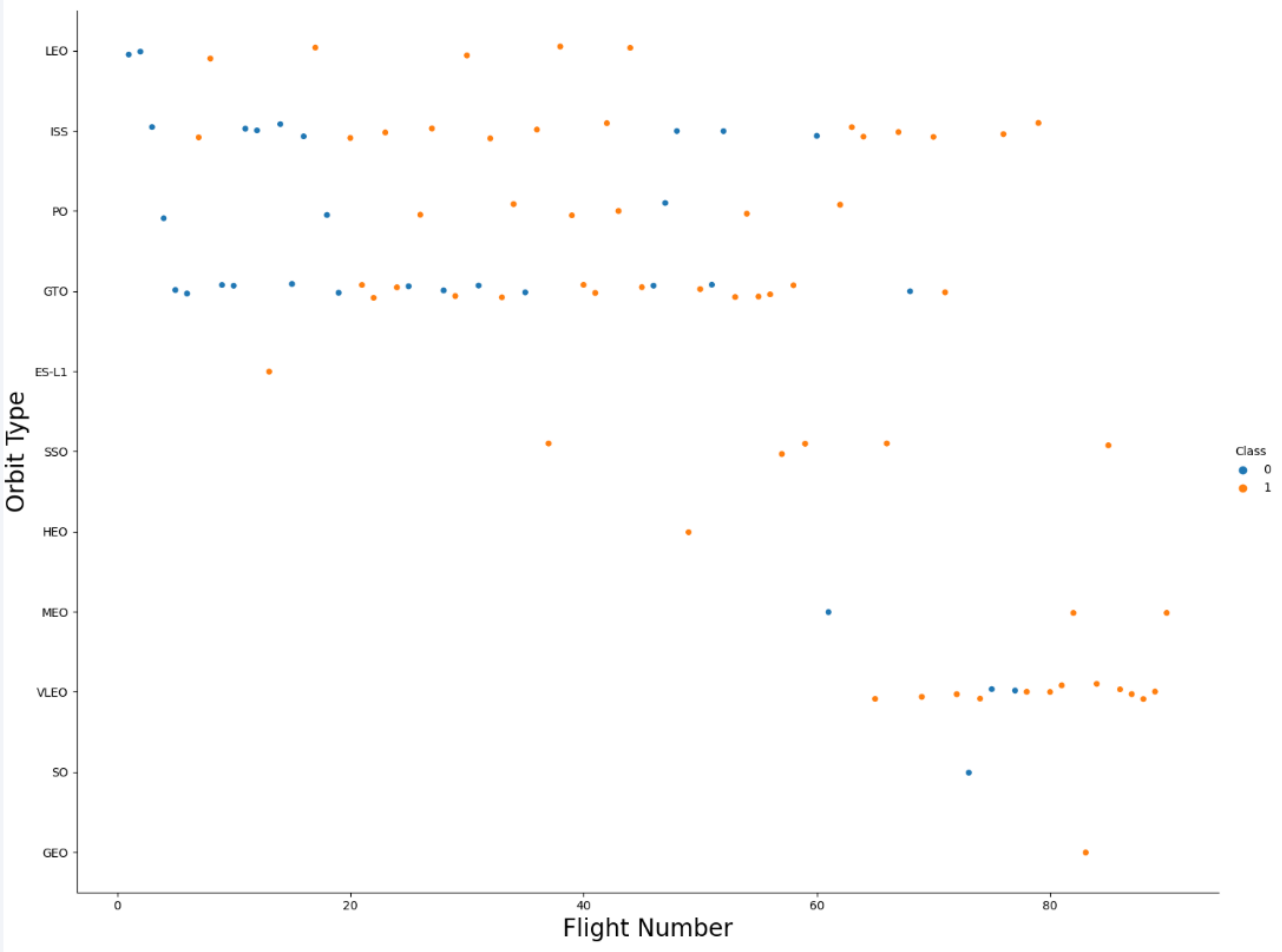


Success Rate vs. Orbit Type





Flight Number vs. Orbit Type

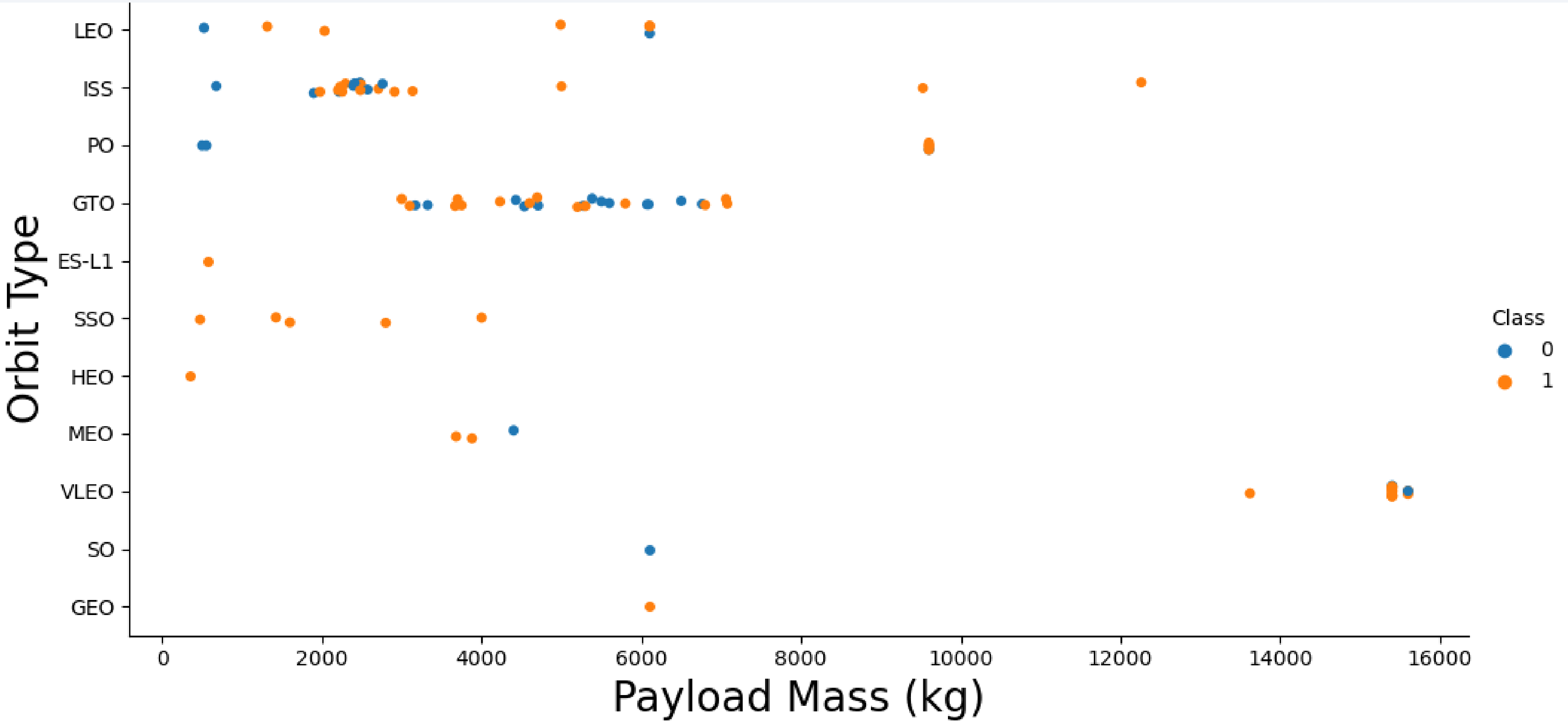


The launches made initially were indistinct for LEO, ISS, PO, GTO orbits, as the number of flights increased, the range of the launch orbits also increased.

The last flights have been up to the VLEO orbit.

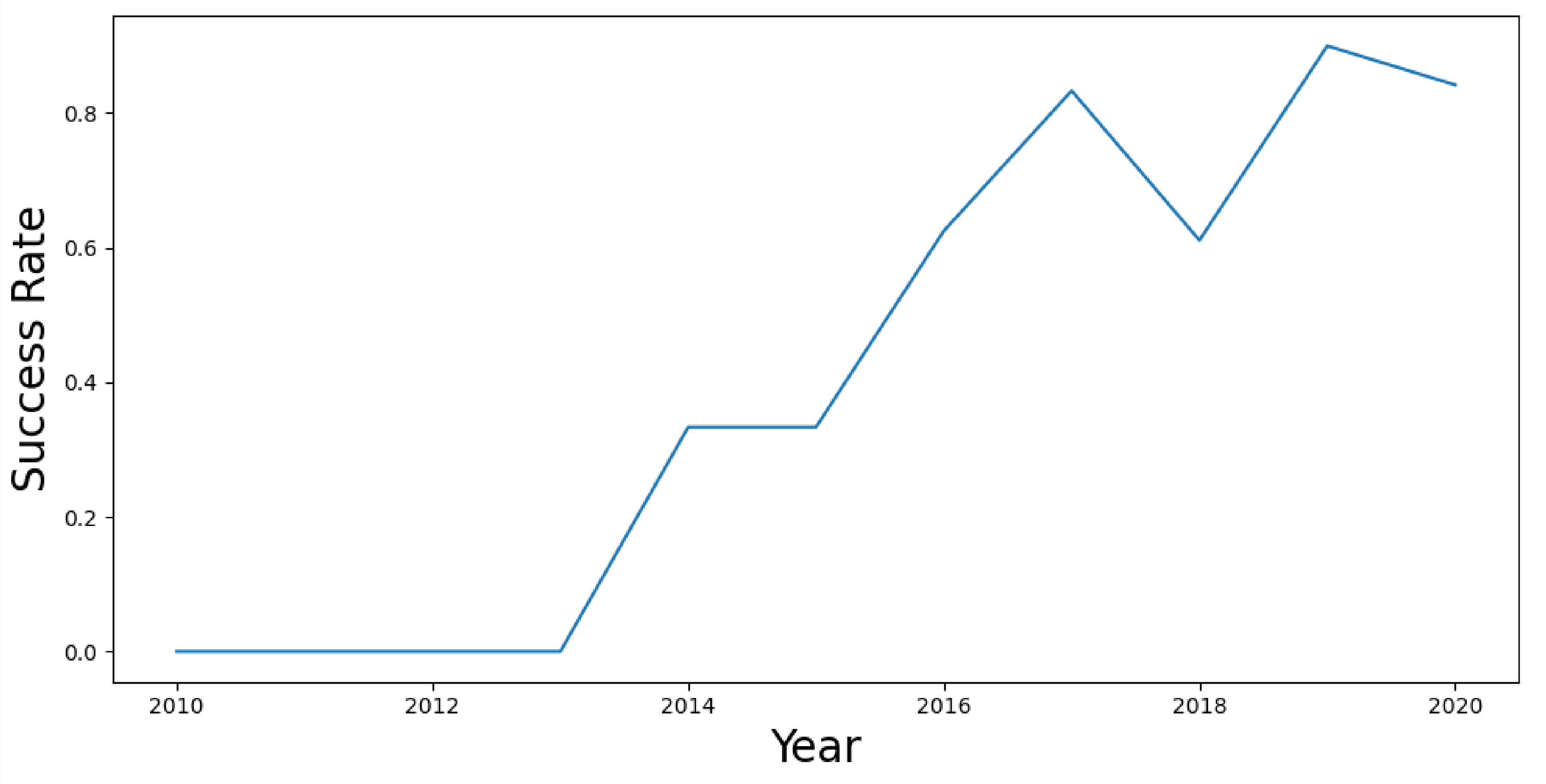
Payload vs. Orbit Type

The relationship between the payload mass and the launch orbit is clear.
When the payload mass is approximately 2000kg to 3000kg, the launch is to the ISS orbit.
When the payload mass is in a range between 3000kg to 7000kg the orbit is the GTO.



Launch Success Yearly Trend

The success rate has increased remarkably over the years.



All Launch Site Names

Launch sites are presented with their latitude and longitude:

	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610745

Launch Site Names Begin with 'CCA'

```
q = pd.read_sql("select * from spacexdata where Launch_Site like 'CCA%' limit 5", conn)
q
```

	index	Date	Time_(UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing__Outcome
0	0	2010-06-04 00:00:00	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	1	2010-12-08 00:00:00	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2	2012-05-22 00:00:00	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	3	2012-10-08 00:00:00	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	4	2013-03-01 00:00:00	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

```
q = pd.read_sql("select sum(PAYLOAD_MASS__KG_) from spacexdata where Customer='NASA (CRS)'", conn)
q
```

	sum(PAYLOAD_MASS__KG_)
0	45596

Average Payload Mass by F9 v1.1

```
q = pd.read_sql("select avg(PAYLOAD_MASS__KG_) from spacexdata where Booster_Version='F9 v1.1'", conn)
q
```

	avg(PAYLOAD_MASS__KG_)
0	2928.4

First Successful Ground Landing Date

```
q = pd.read_sql("select min(Date) from spacexdata where Landing__Outcome='Success (ground pad)'", conn)
q
```

	min(Date)
0	2015-12-22 00:00:00

Successful Drone Ship Landing with Payload between 4000 and 6000

```
q = pd.read_sql("select distinct Booster_Version from spacexdata where Landing__Outcome='Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and
q
```

	Booster_Version
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

```
q = pd.read_sql("select substr(Mission_Outcome,1,7) as Mission_Outcome, count(*) from spacexdata  group by 1", conn)
q
```

	Mission_Outcome	count(*)
0	Failure	1
1	Success	100

Boosters Carried Maximum Payload

```
q = pd.read_sql("select distinct Booster_Version from spacexdata where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from spacexdata)", conn)
q
```

	Booster_Version
0	F9 B5 B1048.4
1	F9 B5 B1049.4
2	F9 B5 B1051.3
3	F9 B5 B1056.4
4	F9 B5 B1048.5
5	F9 B5 B1051.4
6	F9 B5 B1049.5
7	F9 B5 B1060.2
8	F9 B5 B1058.3
9	F9 B5 B1051.6
10	F9 B5 B1060.3
11	F9 B5 B1049.7

2015 Launch Records

```
q = pd.read_sql("select distinct Landing__Outcome, Booster_Version, Launch_Site from spacexdata where Landing__Outcome='Failure (drone ship)'", conn)
q
```

	Landing__Outcome	Booster_Version	Launch_Site
0	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
1	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40
2	Failure (drone ship)	F9 v1.1 B1017	VAFB SLC-4E
3	Failure (drone ship)	F9 FT B1020	CCAFS LC-40
4	Failure (drone ship)	F9 FT B1024	CCAFS LC-40

List of the failed landing_outcomes in drone ship, booster versions, and launch site names in year 2015

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

	Landing__Outcome	count(*)
0	No attempt	10
1	Success (drone ship)	5
2	Failure (drone ship)	5
3	Success (ground pad)	3
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1

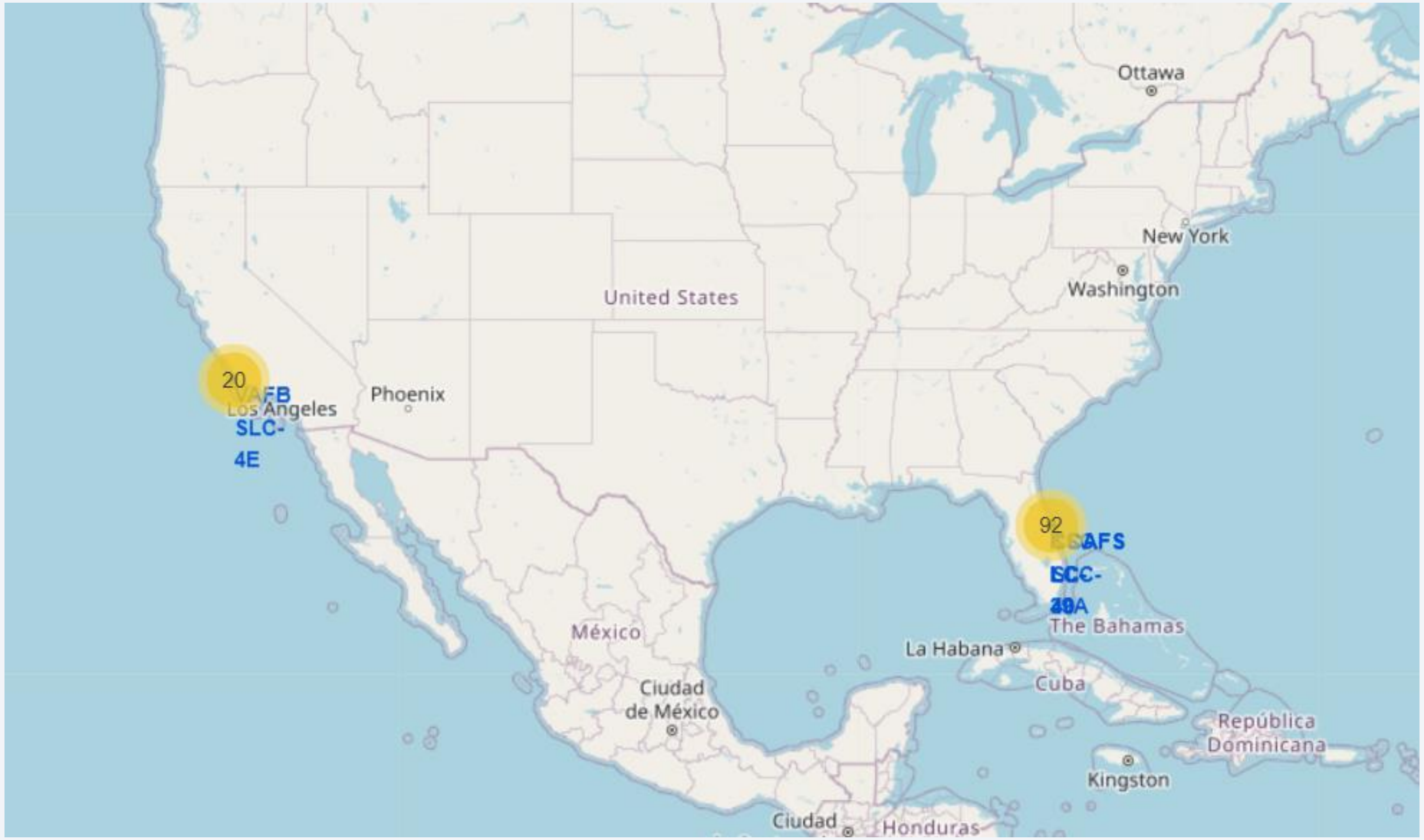
Rank of landing outcomes between 2010-06-04 and 2017-03-20 for:

- Failure (drone ship)
- Success (ground pad)

Section 3

Launch Sites Proximities Analysis

SpaceX Launch Sites Locations (Folium map)



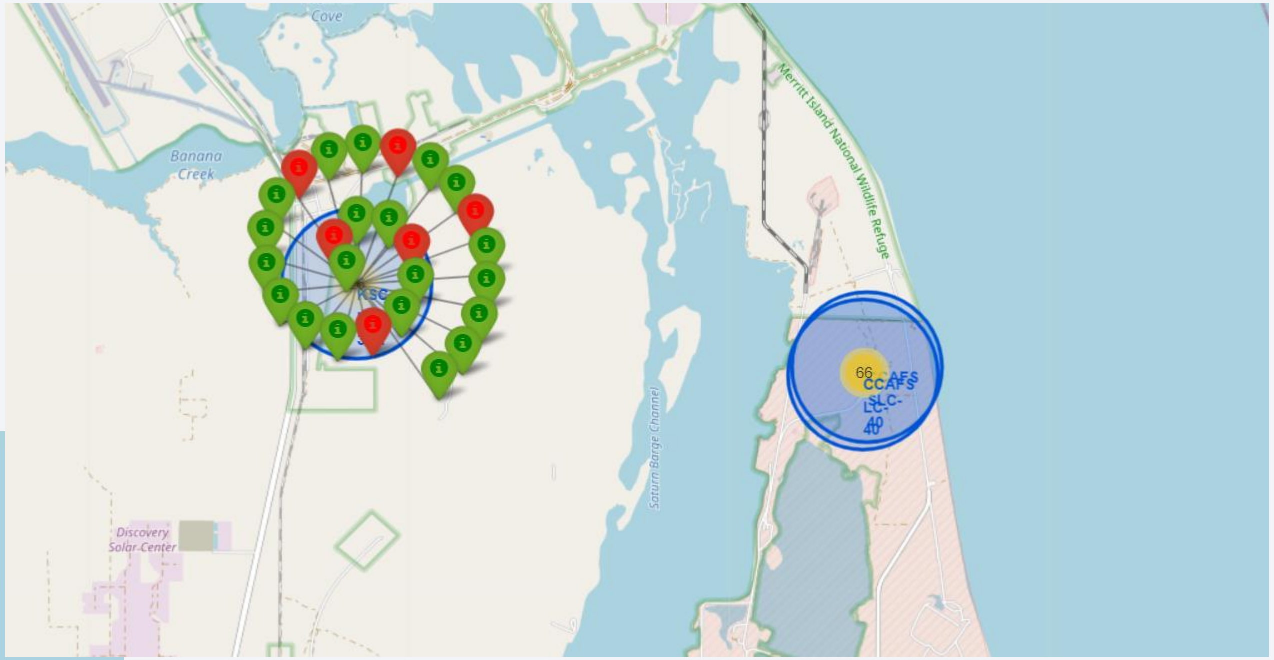
All the launch sites are near to the coast

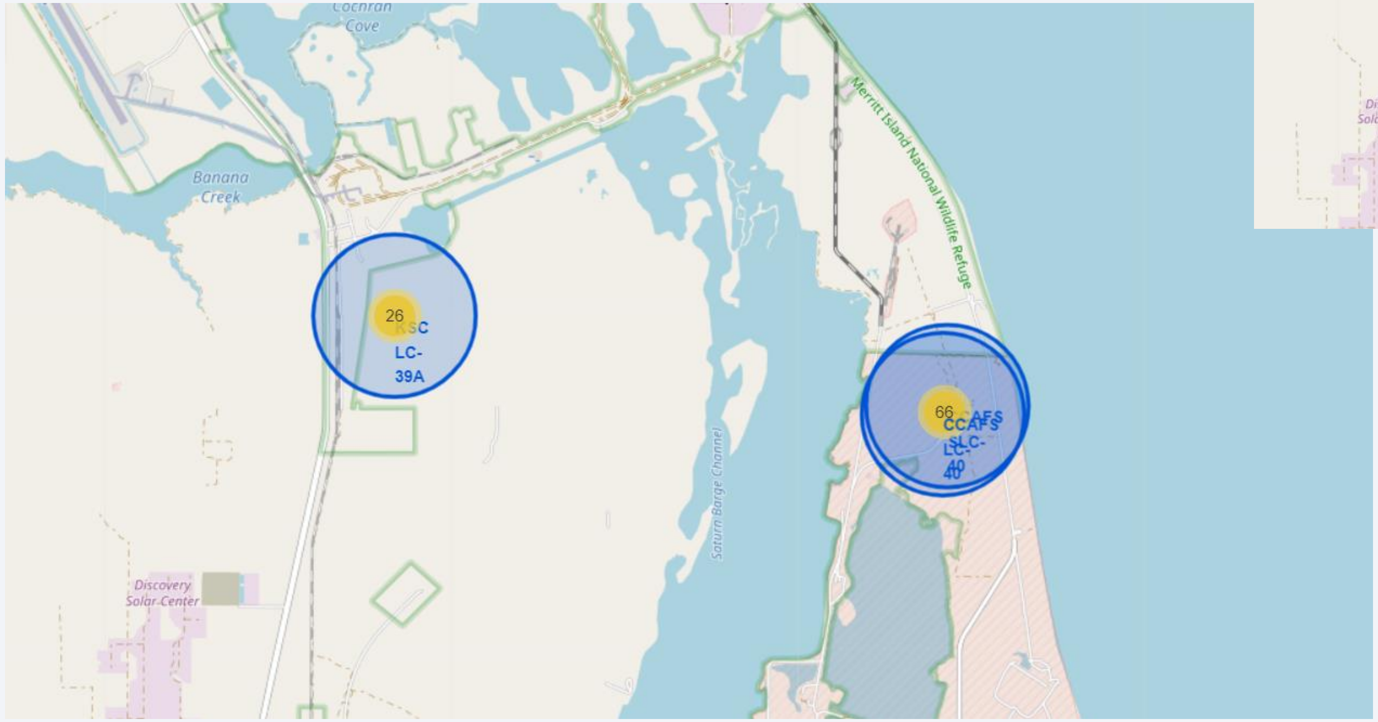
Success and Failed launches

Successful launches are shown in green and failed launches in red.

In the visual map it is possible to zoom in on the study area to learn more about the launches made.

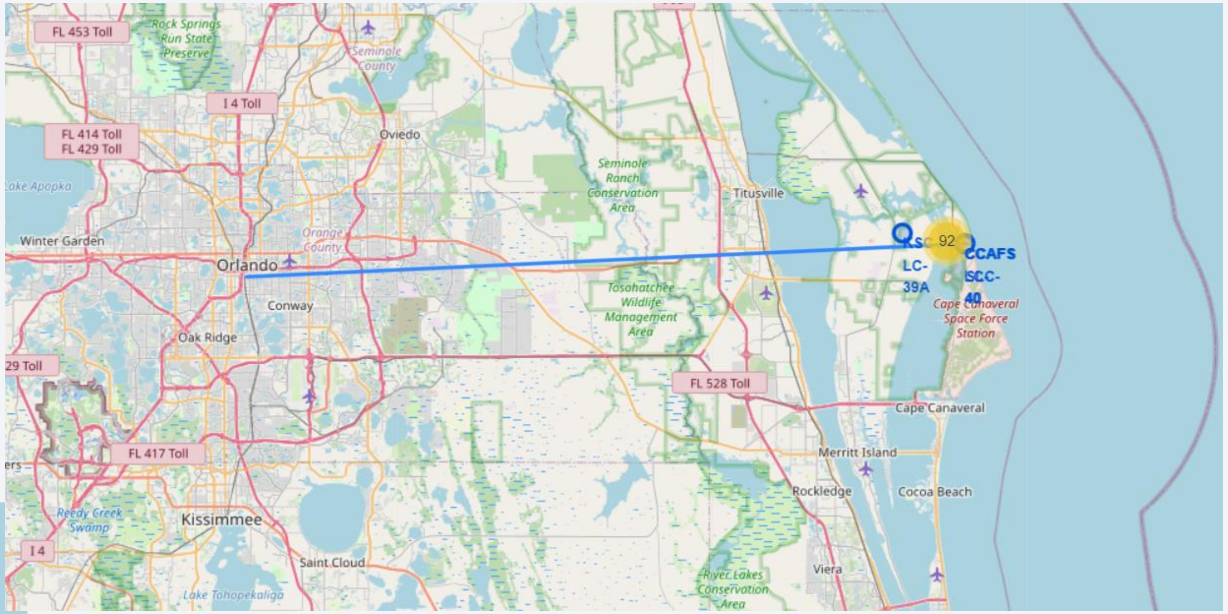
On the East coast there are bases with 26 and 66 pitches

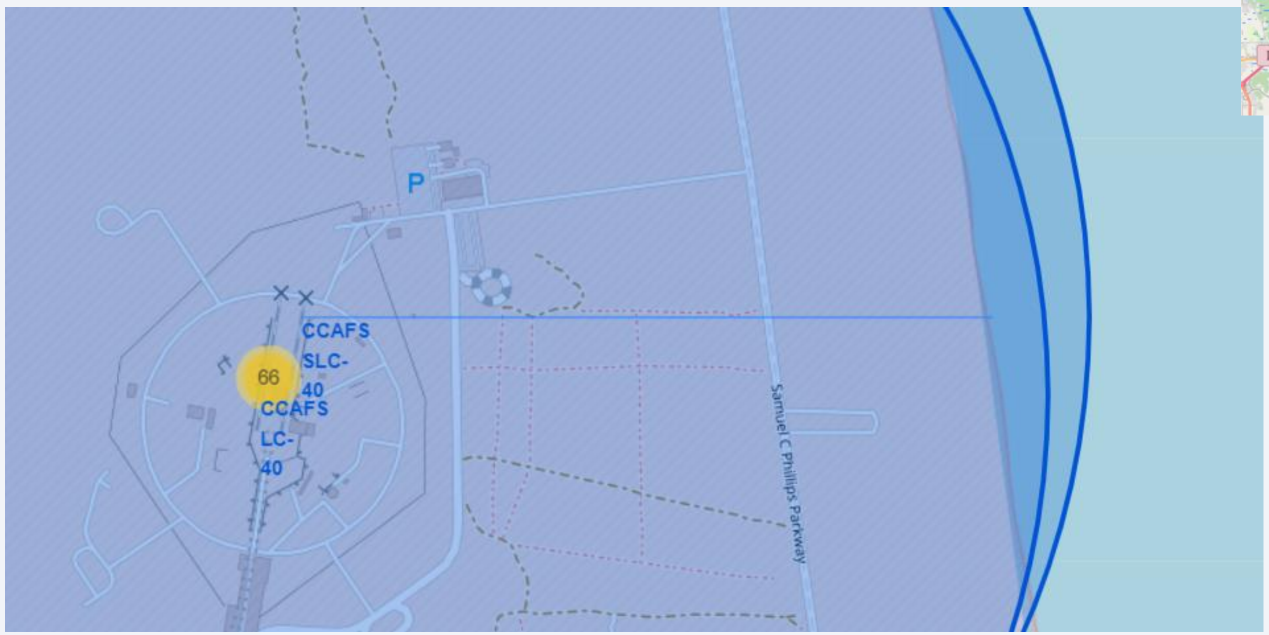




Distance to proximities

The closest distances to different points of interest are presented with a line.
In the images the distance to the coastline is represented

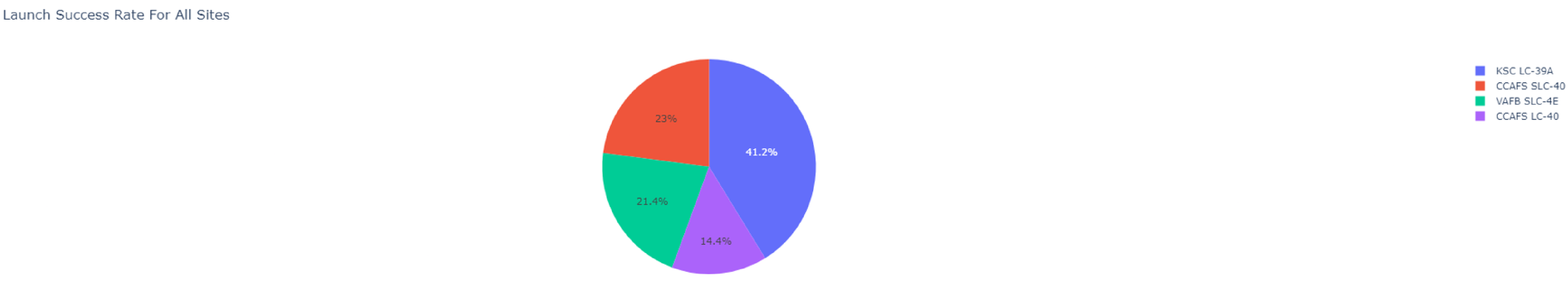




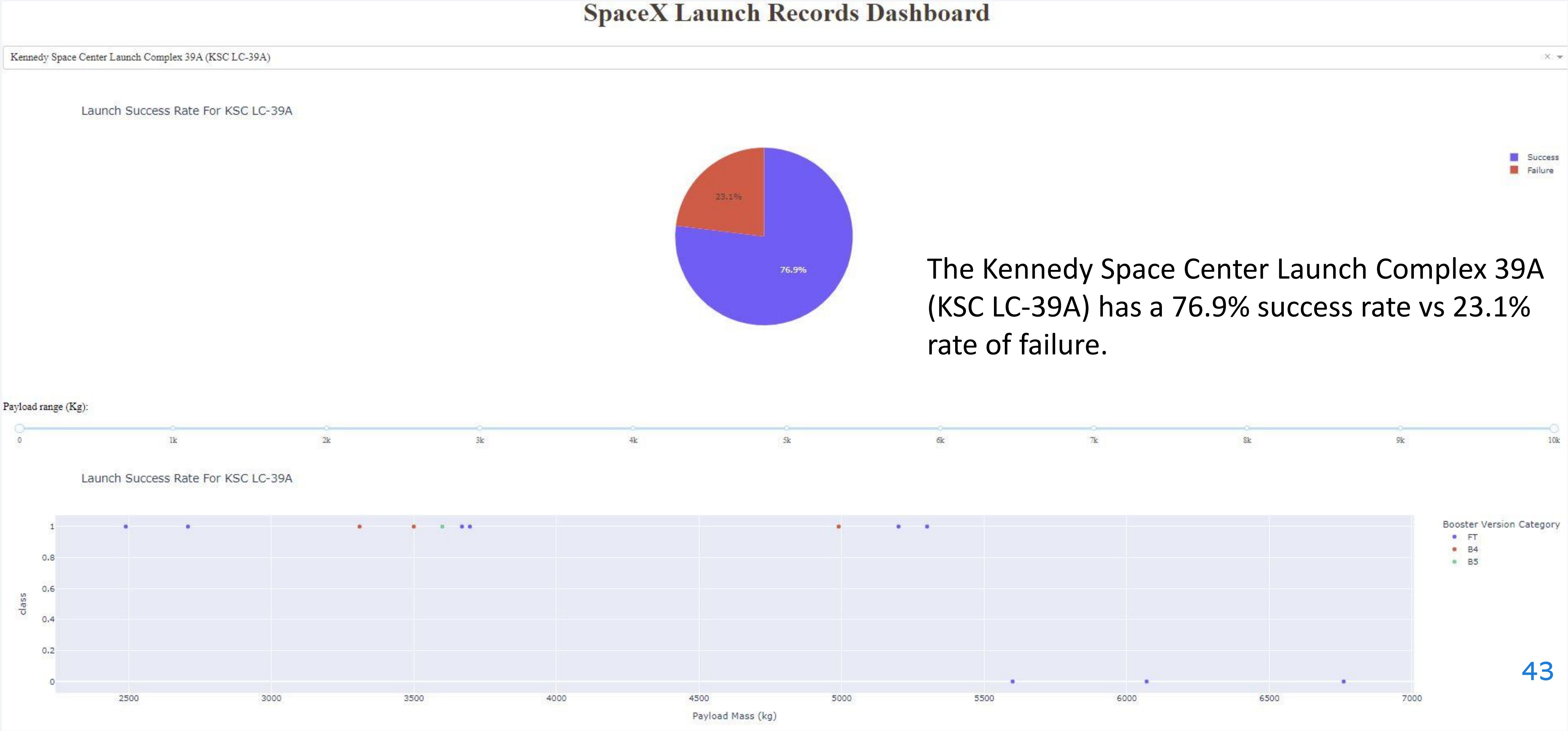
Section 4

Build a Dashboard with Plotly Dash

Launch Success Rate For All Sites

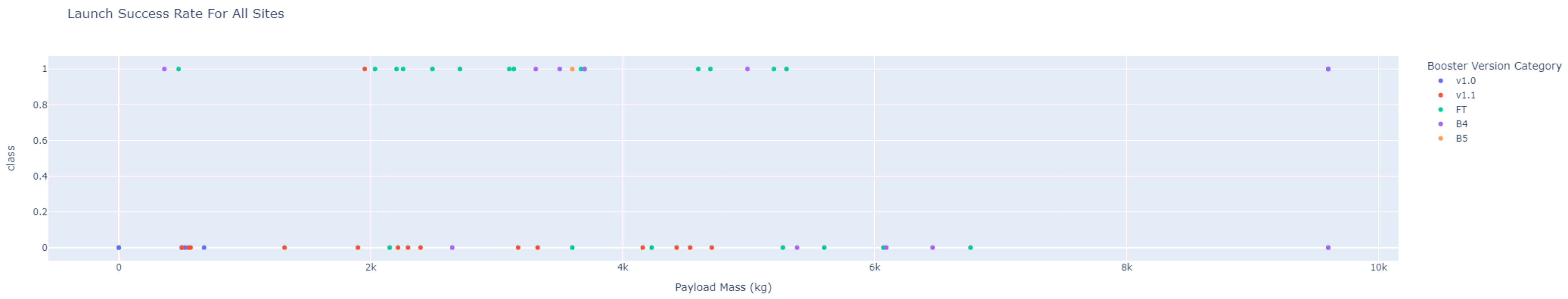


Success rate for the highest launch site



The Kennedy Space Center Launch Complex 39A (KSC LC-39A) has a 76.9% success rate vs 23.1% rate of failure.

Payload Mass



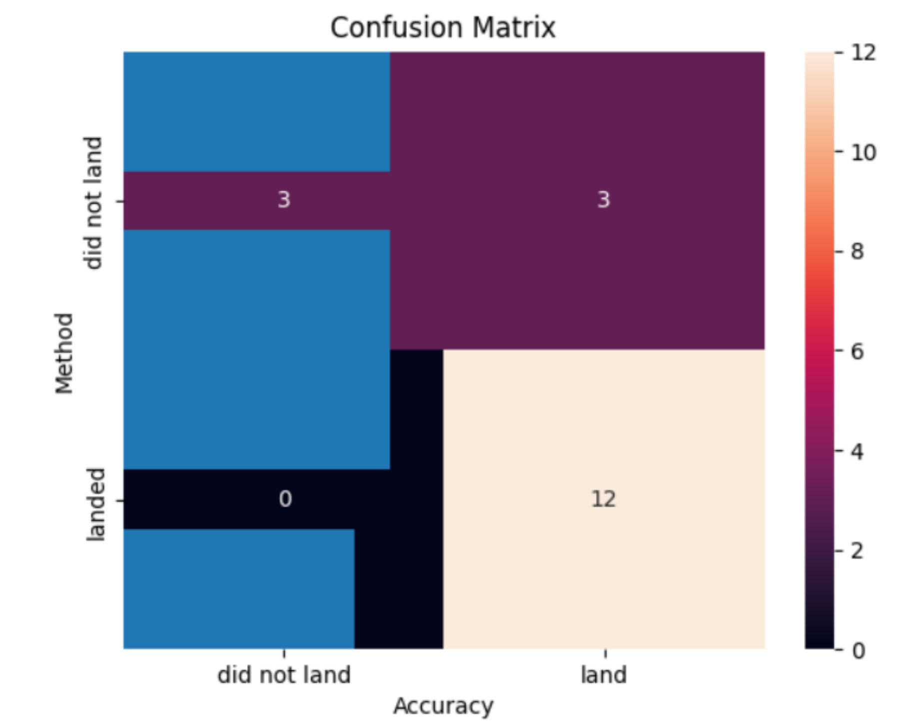
Payload mass (kg) for launch success rate in all sites
There are a high number of launches when the payload mass is between 2k to 4k kg.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The logistic regression, Support vector machine and K-nearest neighbors share a high percentage of accuracy with 0.83



	method	accuracy
0	Logistic regression	0.833333
1	Support vector machine	0.833333
2	Decision tree classifier	0.777778
3	K nearest neighbors	0.833333

Conclusions

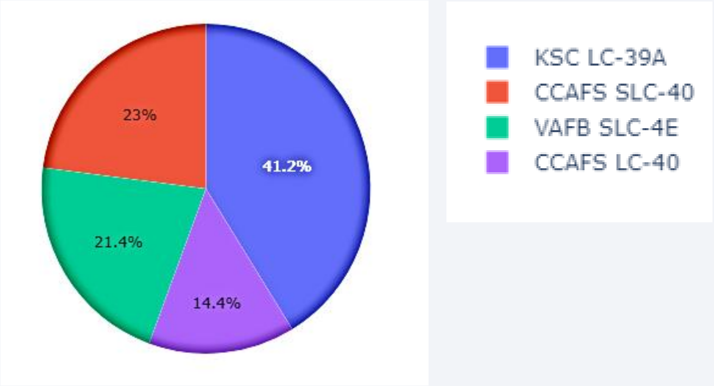
All the launch sites are near to the coastline for transport purposes and security

The Kennedy Space Center Launch Complex 39A (KSC LC-39A) has a 76.9% success rate vs 23.1% rate of failure.

The KSC LC-39A has the most success rate of all the launch sites.

The last flights have been up to the VLEO orbit.

The Support Vector Machine, Logistic Regression and K-nearest neighbors share the same value of accuracy equal to 0.84



Appendix

https://github.com/AndyMik3/SpaceX_Project.git

Thank you!