IBM Developer
SKILLS NETWORK

https://github.com/elengler11/AppliedDataScienceCapstone.git

Winning Space Race with Data Science

Ellen Engler
May 2023

Outline

https://github.com/elengler11/AppliedDataScienceCapstone.git

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

Executive Summary

Summary of methodologies

Data was collected using the SpaceX API using http get requests. The Json responses were stored in Pandas dataframes. Missing values were replaced with column means, and Falcon 9 data was extracted for this study. Several plot types were used to visualize relationships between variables, and database queries were made in an exploratory data analysis. Launch site locations, associated landing outcomes, and transport feature proximities were mapped using Folium. Users may access the constructed Plotly Dash dashboard to display selected data visualizations. Four types of landing success predictive classification machine learning models were trained, tested, and compared for accuracy.

Executive Summary

Summary of results

- KNN and SVM ML models were most accurate for predicting landing outcomes.
- Landing success has generally increased year-by-year.
- Variables that appear to effect landing outcomes include the following: Booster version, payload mass, orbit type, grid fin and leg use and reuse.
- Limited sample size precludes identifying the strength of these effects, but further analyses may be helpful. For example, several orbit types have only one attempt.
- Number of prior attempts (later flights have more success) and launch site effect the above listed variables.

Table of Contents

Introduction.....	6
Methodology.....	7
Results: Data Collection, Scraping, and Wrangling.....	10
Results from EDA.....	25
Results: Launch Site Proximity Analysis.....	45
Results: Plotly Dashboard.....	50
Results: Predictive Analysis (Classification).....	57
Discussion.....	61
Conclusion	68
Appendix.....	71

https://github.com/elengler11/AppliedDataScienceCapstone.git

Introduction

- The goal of this project was to predict landing success of the first stage of SpaceX launches. This is important because the ability to reuse the first stage hugely reduces the costs.
- The problem is determining the effect of many variables on landing outcome. These variables include launch site, date of launch, payload mass, orbit type, booster version, grid fin and legs use and reuse.

Section 1

Methodology

https://github.com/elengler11/AppliedDataScienceCapstone.git

Methodology

- Data collection methodology:
  - Data was collected using http get requests to a static url with SpaceX data maintained by the IBM Skills Network, and organized into a Pandas dataframe.
- Perform data wrangling
  - Missing values were replaced with column means, and landing outcomes were separated into landing class sets (1 = success, 0 = fail)
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash

Methodology: Predictive analysis using classification models

- The target (dependent, y) and predictive variables (independent, x) assigned from the data sets.
- Data was standardized (mean of 0 and unit variance via StandardScaler) to allow for meaningful comparisons between variable values of varying magnitudes and variance.
- X and Y data were each split into training sets to train the model, and testing sets for assessing the accuracy of the trained model.
- GridSearchCV was used to determine the best parameters and hyperparameters for each model. Cross-validation (CV) subsets from the data tune the hyperparameters of the model, while limiting overfitting.
- Objects were created for the estimator of each model (linear regression, SVM, KNN, decision tree), parameters stored in a dictionary, and GridSearchCV analyzed the parameters and fitted hyperparameters to maximize the accuracy of each model.
- Models were evaluated with the “score” method, and confusion matrices visualized error types (false positives, false negatives).
- A table was produced to compare accuracy score of the different models.

https://github.com/elengler11/AppliedDataScienceCapstone.git

Section 2

Results: Data Collection, Scraping, and Wrangling

Data Collection Results Overview

Flow chart of data collection methods with code follow in later slides.

- Requests, numpy, pandas, and datetime libraries were imported.
- Functions to call specified data from launch and core data sets and append to selected lists were predefined.
- An http get request used to call SpaceX dataset via the SpaceX api, and a large very large data set was received.
- Data collection was then limited to the given SpaceX url within the IBM Skills Network for consistent results, and the Json response was converted to a Pandas data frame. Many values in this data frame are ids…???
- Data subsets were organized into a data frame: from rocket: booster version, from payload: payload mass and orbit, from launch pad: launch site names and map coordinates, from cores: outcomes, landing type, number of flights with that core, gridfin use, core is reuse, leg use, the landing pad used, the block of the core, and the serial of the core.
- Predefined functions to call this data with http get requests and append to named lists were used, and the lists were compiled into a dictionary that was converted into a Pandas data frame.
- The data frame was then filtered to contain only rows regarding Falcon 9 information and flight numbers were reset.

Data Collection Results – SpaceX API

Import libraries

```
# Requests allows us to make HTTP requests which we will use to get data from an API
import requests
# Pandas is a software library written for the Python programming language for data manipulation and analysis.
import pandas as pd
# NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices,
import numpy as np
# Datetime is a library that allows us to represent dates
import datetime
```

Request via SpaceX API with get requests

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)

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

Data Collection Results – SpaceX API

Convert Json response to dataframe

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

```
data.head()
```

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	capsules
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	[]	[]	[]
1	None	NaN	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 301, 'altitude': 289, 'reason': 'harmonic oscillation leading to	Successful first stage burn and transition to second stage, maximum altitude 289 km, Premature engine shutdown	[]	[]	[]

Data Collection – SpaceX API, IBM Data Set

Data collection limited to the given url for consistency

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

Data was defined for api calls

```
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])
data['date'] = pd.to_datetime(data['date_utc']).dt.date
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Lists made for data that will be requested

```
#Global variables
BoosterVersion = []
Orbit = []
Outcome = []
GridFins = []
Legs = []
PayloadMass = []
LaunchSite = []
Flights = []
Reused = []
LandingPad = []
Latitude = []
Longitude = []
Serial = []
Block = []
ReusedCount = []
```

Data Collection – SpaceX API, IBM Data Set

Request info and append to lists

getBoosterVersion(data)*
getLaunchSite(data)*
getPayloadData(data)*
getCoreData(data)*

Pre-defined functions make these calls:

```
*def get_______(data):
    for x in data['_____']:
        if x:
        response = requests.get("https://api.spacexdata.com/....json()
        _______.append(response['____']
```

Data Collection – SpaceX API, IBM Data Set

Construct dictionary using the data in the lists

```
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}
```

Data Collection – SpaceX API, IBM Data Set

Construct the new dataframe from the dictionary

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

```
df.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	Reus
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	

Data Collection – SpaceX API, IBM Data Set

Extract only Falcon 9 launches and save in a new dataframe.

```
# get indices for which column BoosterVersion has value 'Falcon 1'
index_names = df[ df['BoosterVersion'] == 'Falcon 1' ].index

# drop these rows from dataFrame
df.drop(index_names, inplace = True)

data_falcon9 = df

print(data_falcon9.shape)
df.head()
```

Reset flight numbers for continuity.

```
data_falcon9['FlightNumber'] = range(1,len(data_falcon9)+1)
```

Data Collection Results – Scraping Data from Wikipedia

URL with HTML data table

```
static_url =
"https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches
&oldid=1027686922"
```

GET request for the HTML data table, response stored in response object (r).

```
r=requests.get(static_url)
r[0:100] #output check
```

Create BeautifulSoup object (soup) from response Object.

```
soup = BeautifulSoup(r.text, 'html.parser’)
print(soup.title) #output check
```

Data Collection Results – Scraping Data from Wikipedia

Use find.all method to extract data in html tables into a list.

```
html_tables = soup.find_all('table')
```

View data from the third html table which contains the desired information.

```
first_launch_table = html_tables[2]
print(first_launch_table)
```

Extract column names to a list.

```
column_names = []

cols = first_launch_table.find_all('th')
for col in cols:
    name = extract_column_from_header(col)
    if name is not None and len(name) > 0:
        column_names.append(name)
```

Data Collection Results – Scraping Data from Wikipedia

Remove irrelevant column and create dictionary with column names as keys. Values from the HTML table will be added in the next step.

```
launch_dict= dict.fromkeys(column_names)

del launch_dict['Date and time ( )']

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']=[]
```

Data Collection Results – Scraping Data from Wikipedia

Continue to parse the HTML table and load the dictionary with value.

```
extracted_row = 0
for table_number,table in enumerate(soup.find_all('table',"wikitable
plainrowheaders collapsible")):
for rows in table.find_all("tr"):
        if rows.th:
                if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
                flag=False
        row=rows.find_all('td’)
        if flag:
                extracted_row += 1
                launch_dict['Flight No'] ='flight_number'
                 print(flight_number)
                datatimelist=date_time(row[0])
```

Convert dictionary to dataframe.

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

Data Wrangling Results

- Missing values were replaced with the mean of the values present in each column.
- Failed landing separated into a set.
- Code used follows in later slides.

Data Wrangling Results

Determine types and number of landing outcomes.

```
landing_outcomes=df['Outcome'].value_counts()
landing_outcomes
```

View landing outcomes as a list of dictionary keys.

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

Separate failed landings as a subset.

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

Replace Boolean values with integers (0,1)

```
landing_class=[]
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

Section 3

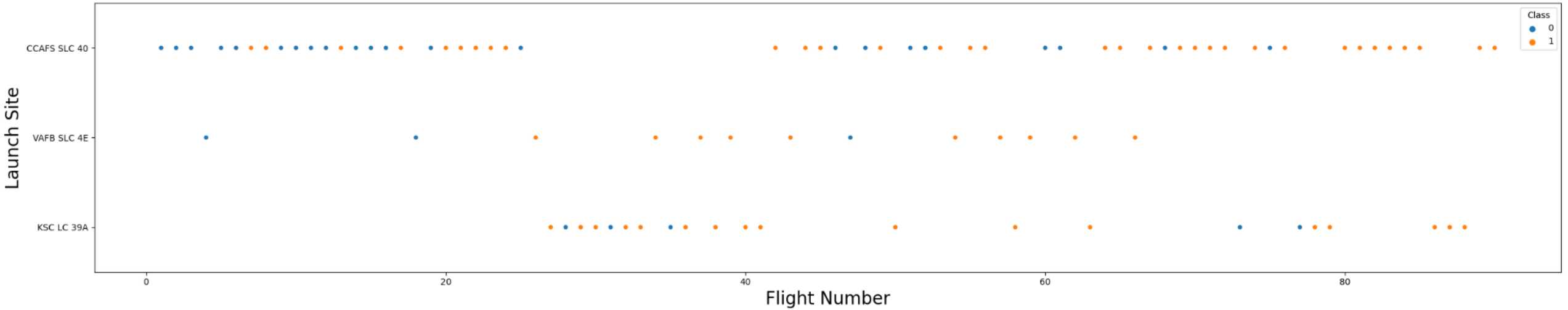
Results from EDA

https://github.com/elengler11/AppliedDataScienceCapstone.git

EDA with Data Visualization Results Overview

- Matplotlib and Seaborn libraries were imported.
- A variety of plots including scatter, bar, and line plots were constructed based on the data frame to explore relationships between the variable…
- **Flight number v. Payload Mass** (scatterplots): Explore how payload mass changed over time, and how each of these variables affects the landing outcome.
- **Flight number v. Launch Site** (scatterplots): Explore launch site usage over time along with landing success.
- **Launch Site v. Payload Mass** (scatterplot, box plot): Explore distribution of different payload masses at each launch site along with landing outcome.
- **Orbit v. Landing Outcome** (bar plot, histogram): Explore relationship between landing success and orbit type.
- **Orbit v. Flight Number** (scatterplot): Explore how chosen orbits type changed over time and examine success of landing.
- **Success Rate by Year** (line plot): Examine how the landing success rate has changed over time.

Flight Number vs. Launch Site

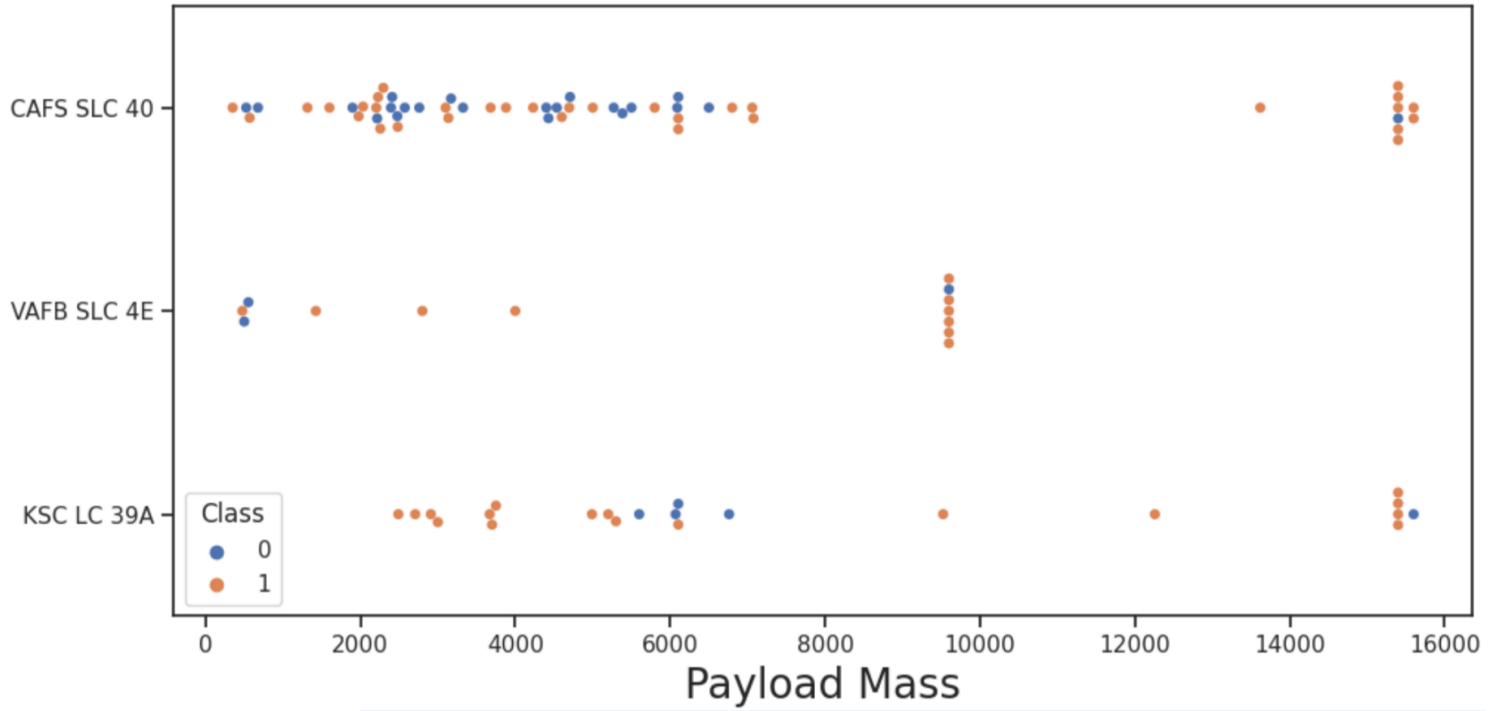


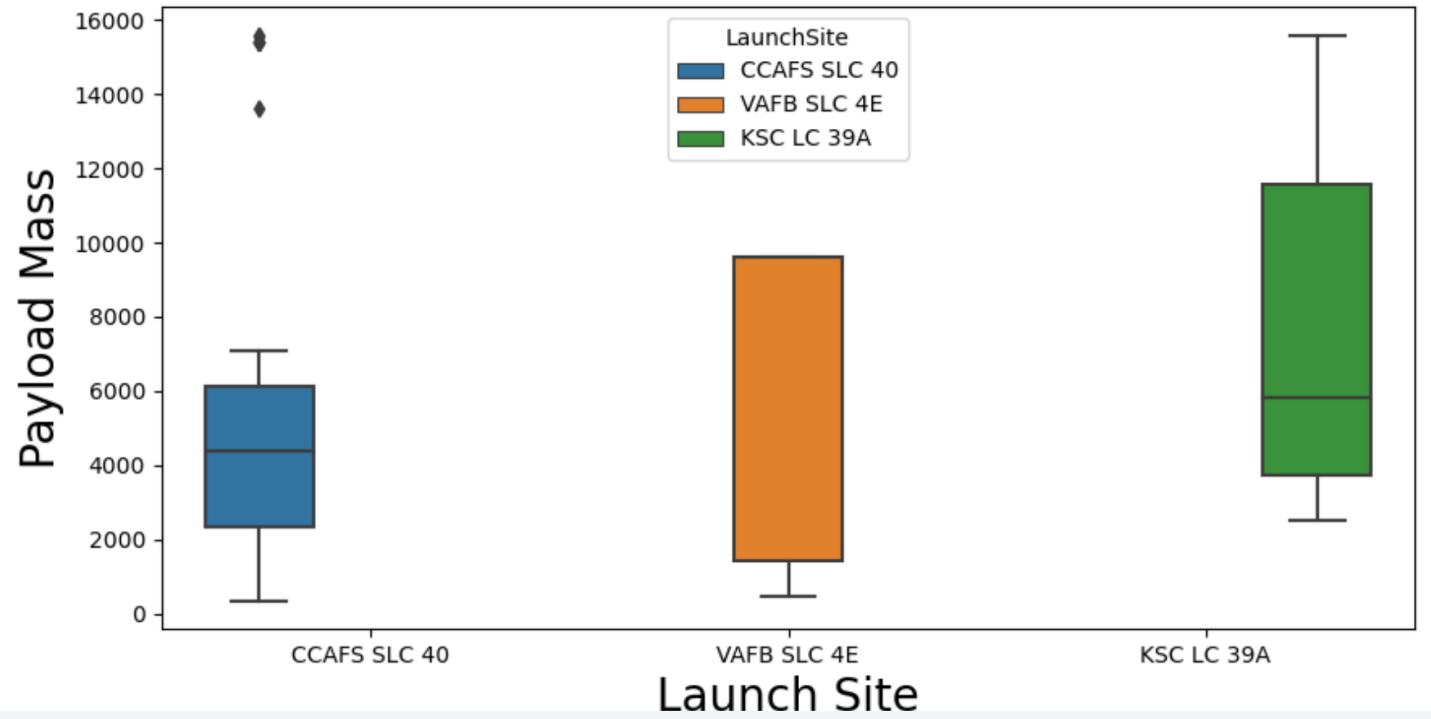
- Flights 1-20 had the lowest success rate and were almost all at CCAFS SLC 40.
- The success rate of all sites increases with later flights.
- KSC LC 39A was inactive until flight 24.
- VAFB SLC 4E became inactive after flight 64.

Payload vs. Launch Site

These plots show the relationship between payload mass, launch site and landing success.

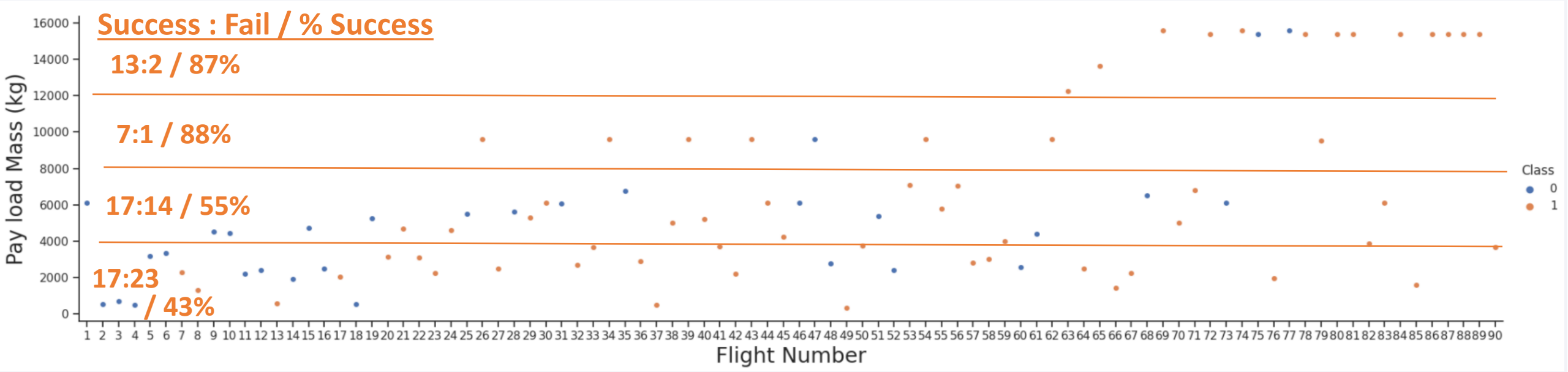
- KSC has a more evenly distributed range of payloads than the other sites.
- VAFB has half its attempts at or below 4000 kg while the remaining half equaled 10,000 kg.
- CAFS has the largest number of payloads above 14,000 kg.





- The box plot clarifies the information from the scatter plot by showing means, quartiles, range and outliers.

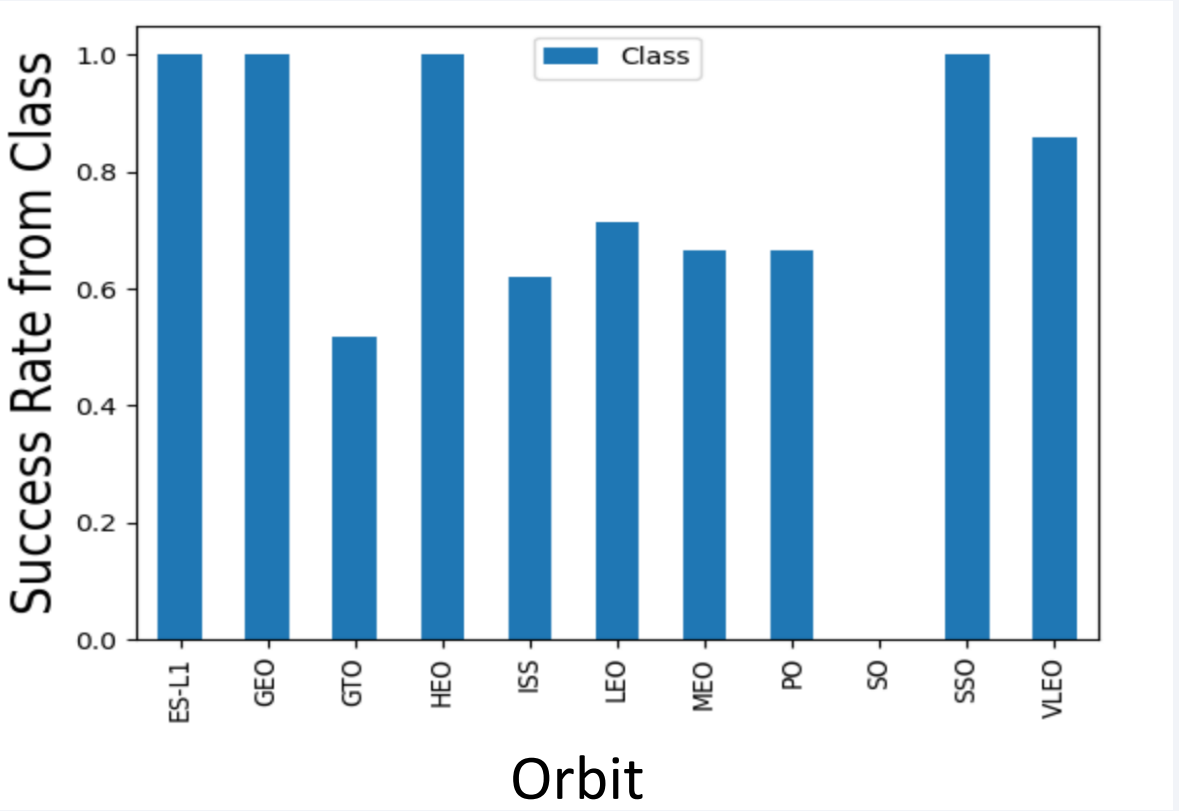
Payload vs. Flight Number

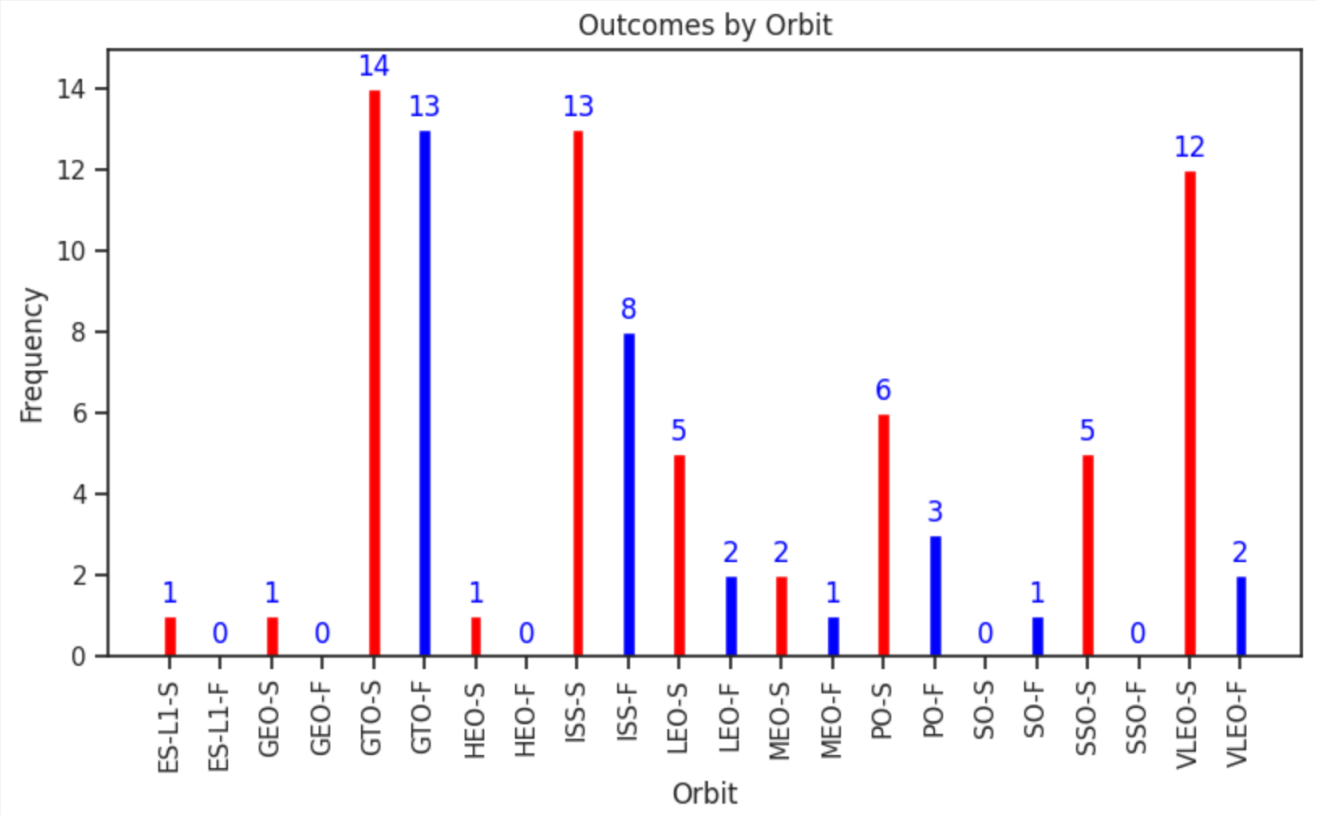


- Most attempts have a payload below 8000 kg.
- Larger payloads generally have been on later flights.
- Landing success generally increases with payload.
- The success rate levels off past 12,000 kg.

Success Rate vs. Orbit Type

- The upper bar chart shows that ESL1, GEO, HEO, and SSO orbits have 100% success.
- GTO has the lowest success and the largest number of attempts.

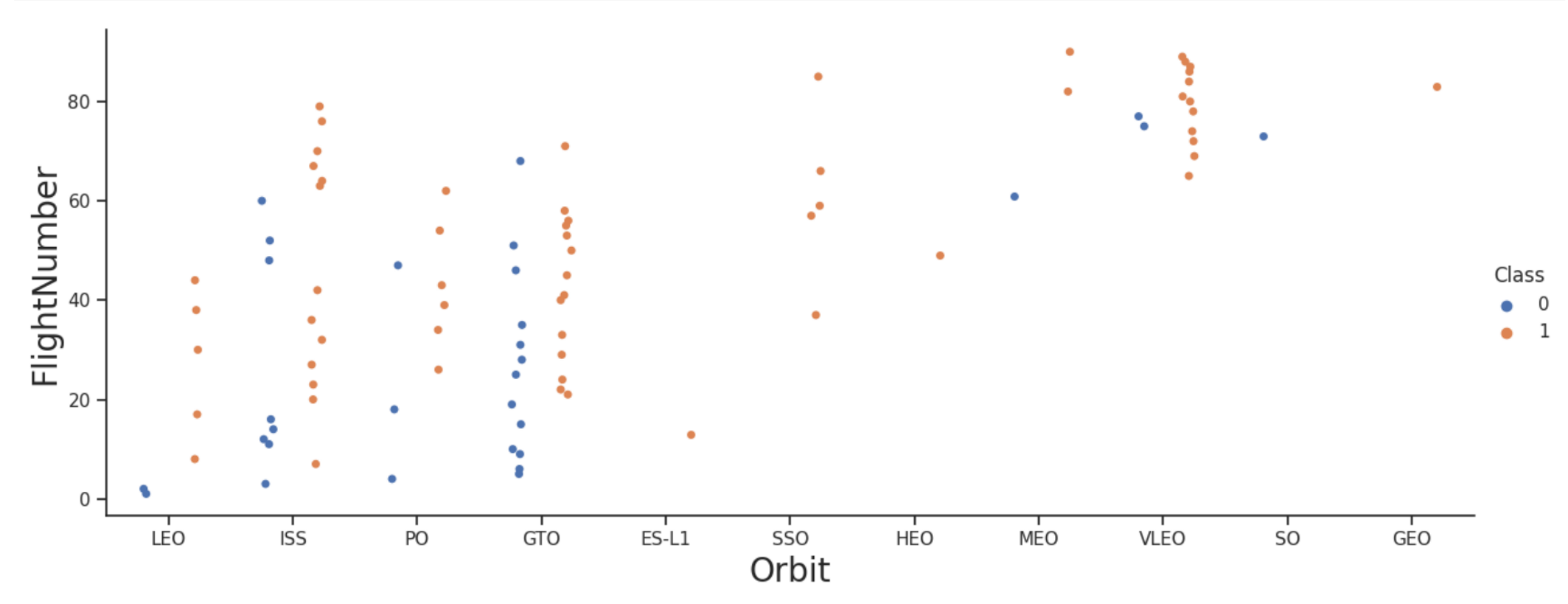




- The histogram shows the number of successful and failed attempts per orbit.
- SSO is the only orbit with a 100% success rate that has more than one attempt.
- VLEO has the highest success rate for any orbit with more than five attempts.

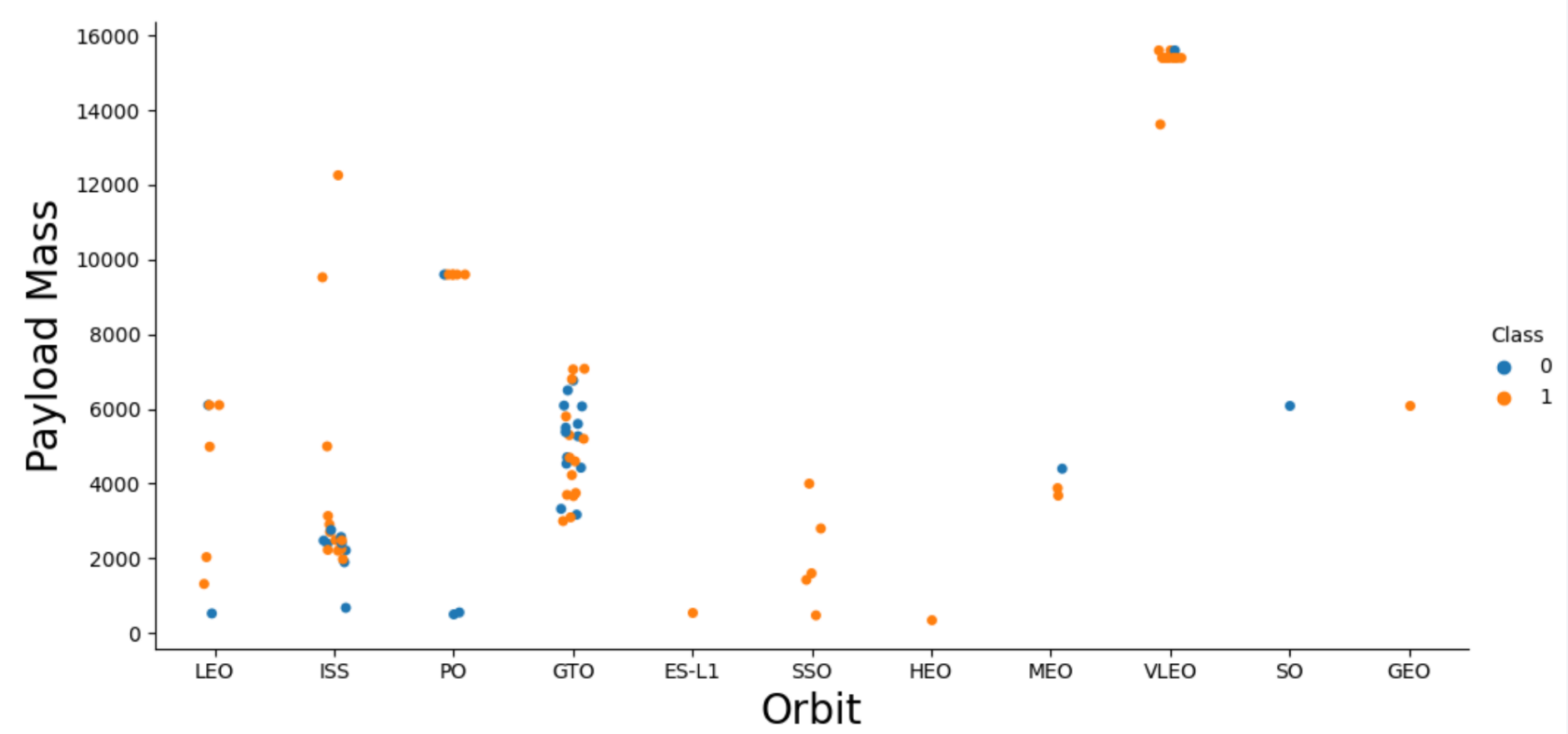
Flight Number vs. Orbit Type

- LEO, ISS, PO, & GTO orbits success rate appear to increase along with flight number/later flights.
- Fewer attempts in the LEO, ISS, PO, and GTO orbits are made in flight numbers greater than 60, while other orbit type increase with the exception of ES-L1.



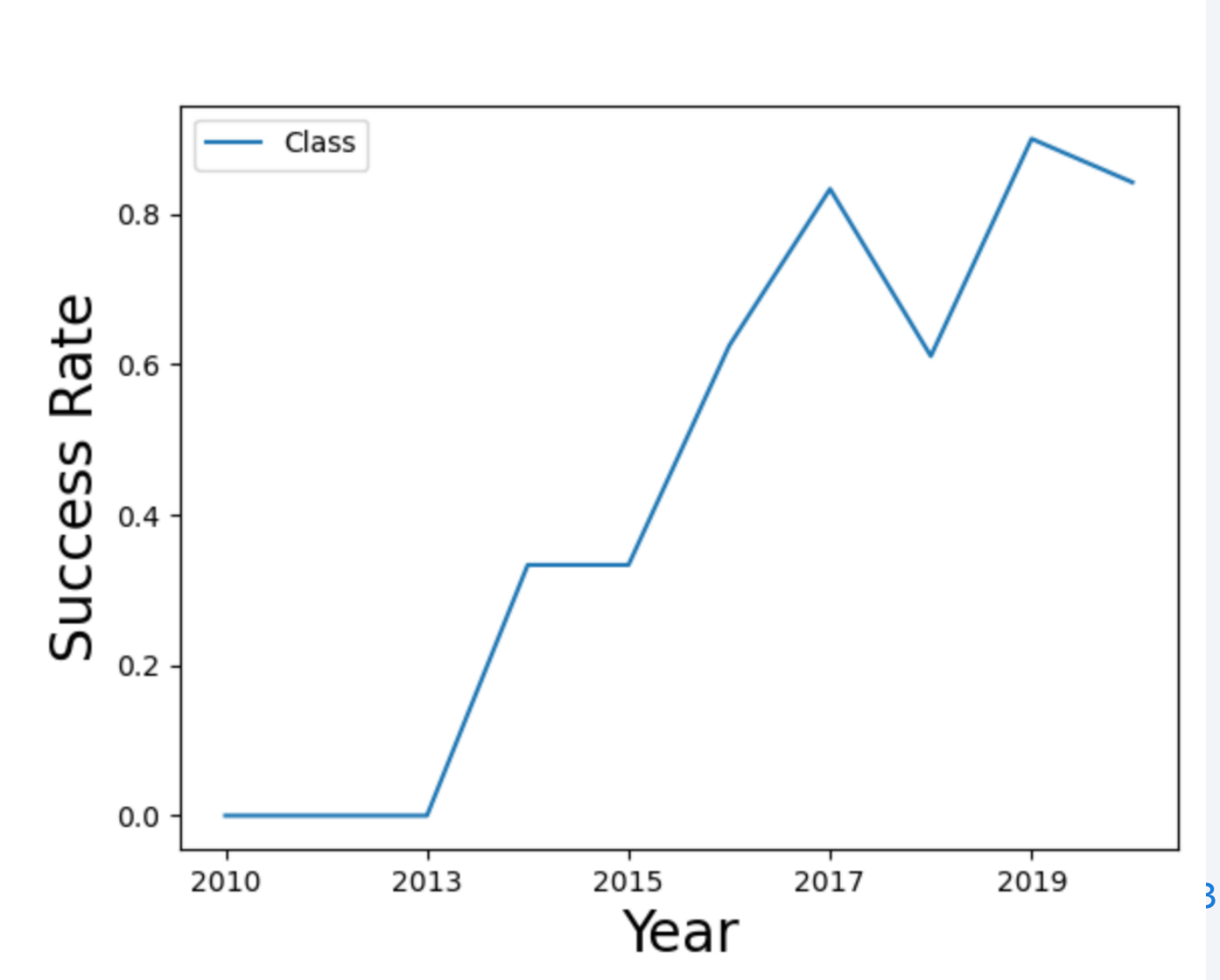
Payload vs. Orbit Type

- LEO, ISS, and PO orbits appear to have more successful landings with increasing payload mass, but no other trends are apparent.



Launch Success Yearly Trend

The success rate for landings generally increases over time.



EDA with SQL Overview

- sqlite3 and the data in csv format were imported.
- The dataset was loaded into a corresponding table in a Db2 database
- Execute SQL queries were performed to learn about the data set.
- Examples with code and results follow in later slides.
- Queries made include the following:
  - ✓ Names of the unique launch sites in the space mission
  - ✓ Five records where launch sites begin with the string 'CCA'
  - ✓ Total payload mass carried by boosters launched by NASA (CRS)
  - ✓ Average payload mass carried by booster version F9 v1.1
  - ✓ Date when the first successful landing outcome in ground pad was achieved.
  - ✓ Names of boosters with successful drone ship landing and payload mass between 4000 and 6000 kg.
  - ✓ Total number of successful and failure mission outcomes
  - ✓ Names of the booster_versions which have carried the maximum payload mass.
  - ✓ Records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015
  - ✓ Ranked count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

All Launch Site Names

Names of the unique launch sites in the space mission

Unique (“distinct”) launch site names were called from the SpaceX table using the select function.

```
#%sql select * from SPACEXTBL;
```

```
%sql select distinct "Launch_Site" from SPACEXTBL;
```

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

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
None

Launch Site Names Begin with 'CCA'

Five records where launch sites begin with the string 'CCA' were called from the SpaceX table using clauses beginning with "like" to select CCA_ and "limit" to show only five results.

```
%sql select * from SPACEXTBL where "Launch_Site" like "CCA%" limit 5;
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (parachute)
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No attempt
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No attempt
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

Total payload mass carried by boosters launched by NASA (CRS) was calculated by selecting payload mass values from the SpaceX table where the associated customer value was restricted to NASA (CRS) and adding these values together.

```
%sql select SUM ("PAYLOAD_MASS__KG_") FROM SPACEXTBL where "Customer" = "NASA (CRS)";
```

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

SUM ("PAYLOAD_MASS__KG_")
45596.0

Average Payload Mass by F9 v1.1

Average payload mass carried by booster version F9 v1.1 was calculated from payload mass values associated with the booster version value F9 v1.1 on the SpaceX table.

```
%sql select AVG ("PAYLOAD_MASS__KG_") FROM SPACEXTBL where "Booster_Version" = "F9 v1.1";
```

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

AVG ("PAYLOAD_MASS__KG_")
2928.4

First Successful Ground Landing Date

Date when the first successful landing outcome in ground pad was achieved by using the select function to call date values from the chronologically ordered SpaceX table and limiting the result to the first value.

```
%sql select "Date" FROM SPACEXTBL where "Landing_Outcome" = "Success (ground pad)" limit 1;
```

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

Date
22/12/2015

Successful Drone Ship Landing with Payload between 4000 and 6000

Names of the boosters which have success in drone ship and have payload mass between 4000 and 6000 were selected from the SpaceX table from rows where the landing outcome value was “Success (drone ship)” and where the payload mass value had to be within the 4000 to 6000 kg range.

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

```
%sql select "Booster_Version" FROM SPACEXTBL where "Landing_Outcome" = "Success (drone ship)" and "PAYLOAD_MASS__KG_" between 4000 and 60
```

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

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

Boosters Carried Maximum Payload

```
%sql SELECT "Booster_Version" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT max ("PAYLOAD_MASS__KG_") FROM SPACEXTBL);
```

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

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

Names of the booster_versions which have carried the maximum payload mass were selected from the SpaceX table by restricting the associated payload mass values with a subquery set to call the maximum payload value.

Total Number of Successful and Failure Mission Outcomes

Total number of successful and failure mission outcomes

%sql SELECT COUNT ("Mission_Outcome") FROM SPACEXTBL group by "Mission_Outcome";

```
%sql SELECT COUNT ("Mission_Outcome" ) FROM SPACEXTBL group by "Mission_Outcome";
 * sqlite:///my_data1.db
Done.
```

COUNT ("Mission_Outcome")
0
1
98
1
1

2015 Launch Records

Records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015 were selected from the SpaceX table with restrictions for date and landing outcome in the where clause.

```
%sql select substr(Date, 4, 2), "Landing_Outcome", "Booster_Version", "Launch_Site" from SPACEXTBL WHERE substr(Date,7,4)='2015'
and "Landing_Outcome" == "Failure (drone ship)";
```

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

Done.
```

substr(Date, 4, 2)	Landing_Outcome	Booster_Version	Launch_Site
(October) 10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
(April) 04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

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

```
%sql select "Date", "Mission_Outcome" from SPACEXTBL where substr(Date,7,4) between '2010' and '
and ("Mission_Outcome" == "Success") order by substr(Date,7,4) desc;
```

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

Date	Mission_Outcome
14/01/2017	Success
19/02/2017	Success
16/03/2017	Success
30/03/2017	Success
05/01/2017	Success
15/05/2017	Success
06/03/2017	Success
23/06/2017	Success
25/06/2017	Success
07/05/2017	Success
14/08/2017	Success
24/08/2017	Success
09/07/2017	Success
10/09/2017	Success

Section 4

Results: Launch Site Proximity Analysis

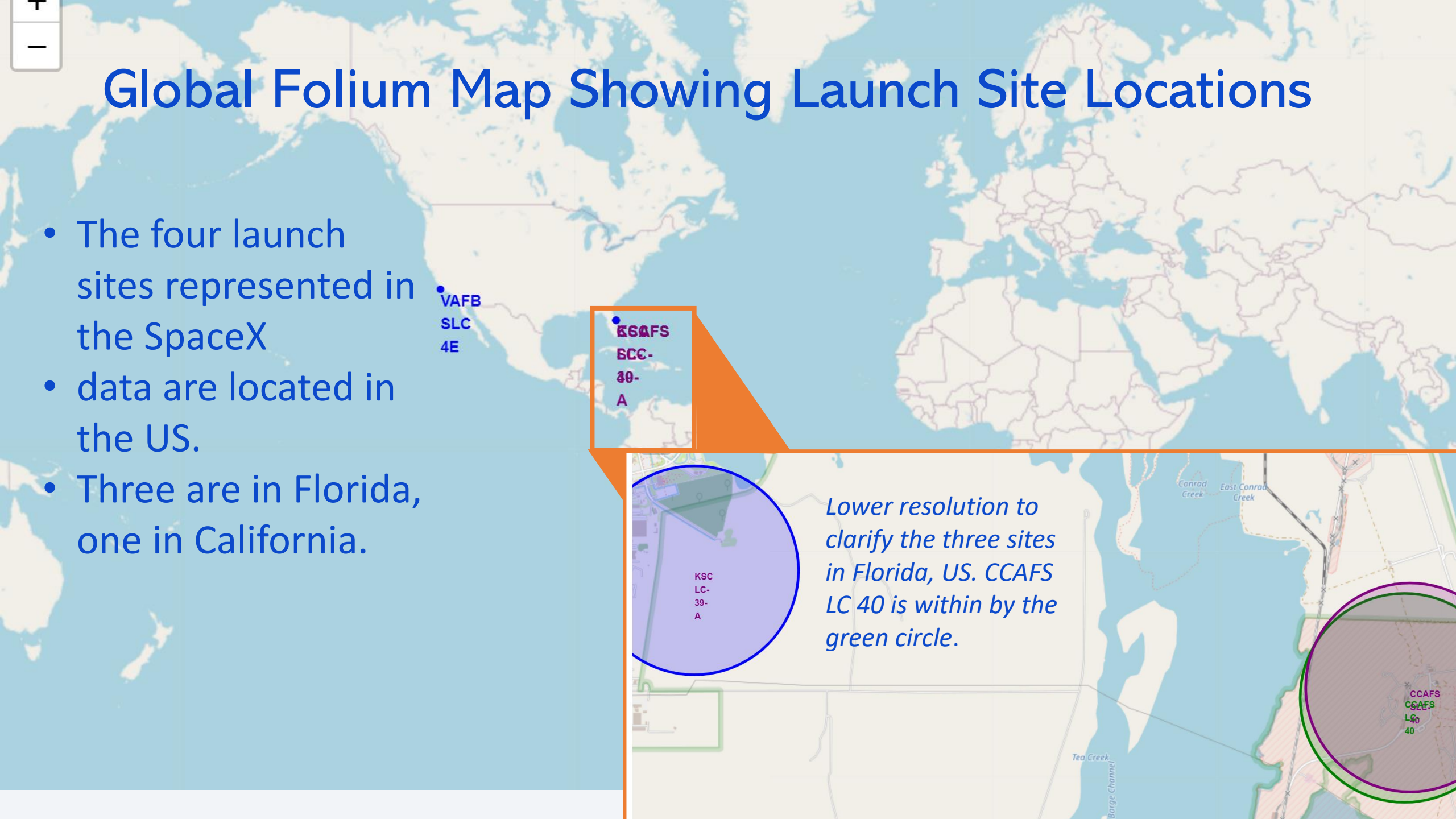
https://github.com/elengler11/AppliedDataScienceCapstone.git

Interactive Map with Folium Results

- Launch sites were located with map coordinates and marked with Folium circles, markers, and text.
- Marker clusters were used to visualize each individual successful and failed attempt at each launch site.
- Markers and polylines were placed to calculate distances from launch sites to transport features allowing for exploration of these cost components.

Global Folium Map Showing Launch Site Locations

- The four launch sites represented in the SpaceX
- data are located in the US.
- Three are in Florida, one in California.

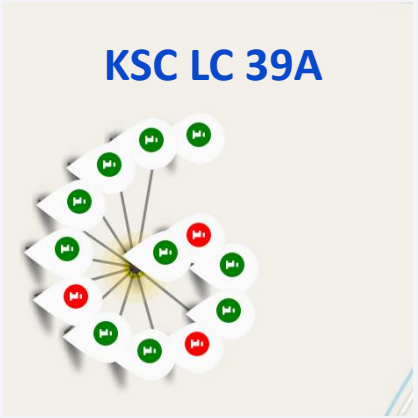


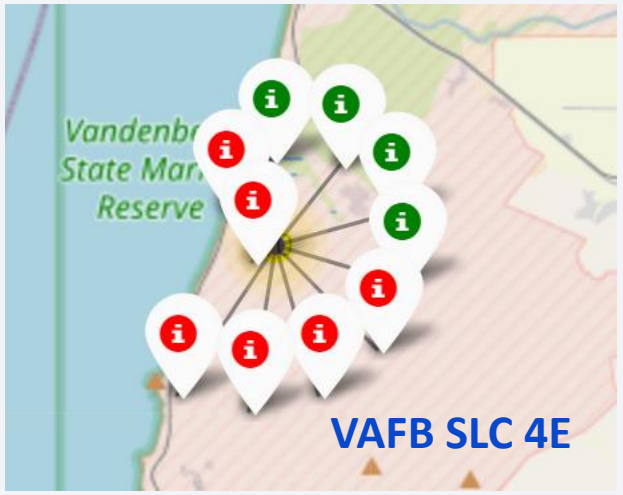
Launch Outcomes by Site

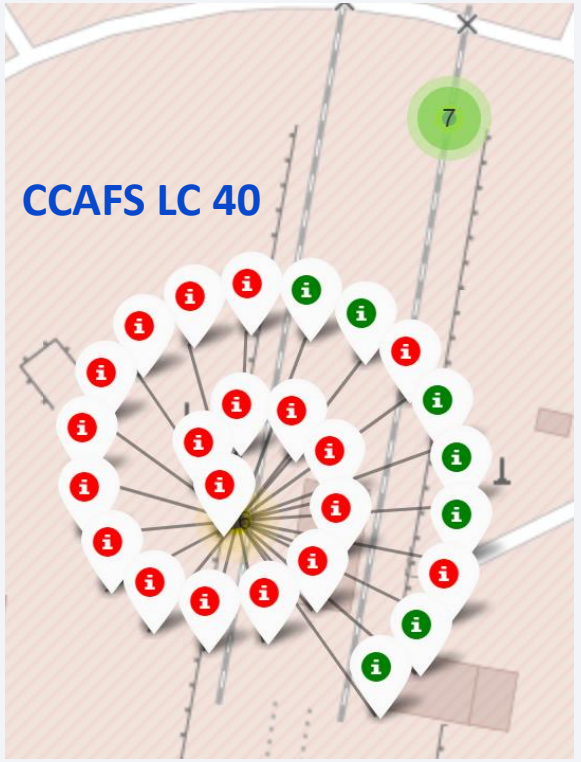
** Folium marker clusters were used to identify locations of all successful (green) and failed (red) launches.*

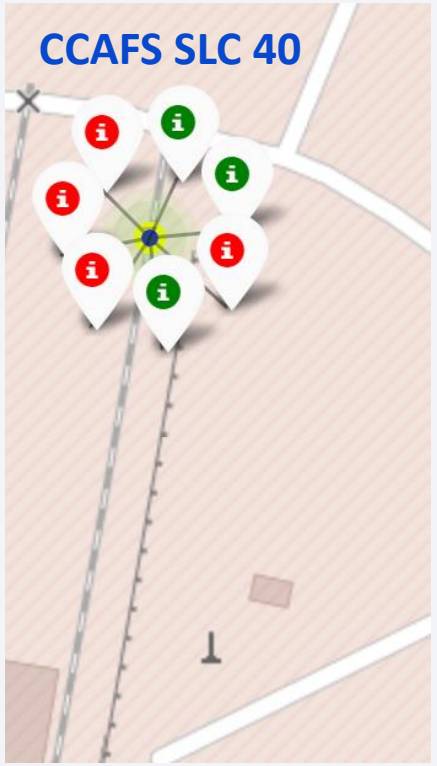
Launch Site	Total Attempts	Success: Fail Ratio	% Successful Landings
CCAFS LC 40	26	7:19	27%
VAFB SLC 4E	10	4:6	40%
KSC LC 39A	13	10:3	77%
CCAFS SLC 40	7	3:4	43%

- CCAFS LC 40 has the most attempts (twice more than any other site) and the lowest success rate.
- VAFB SLC 4E and CCAFS SLC 40 have similar success rates and number of attempts.
- KSC LC 39A has half the attempts of CCAFS LC 40 and a much higher success ratee than any other site.









Launch Site Proximity to Key Transport Feature

- Example of a distance calculation via Folium.
- Distances to key transport features are a useful cost analysis component.

Section 5

Results: Plotly Dashboard

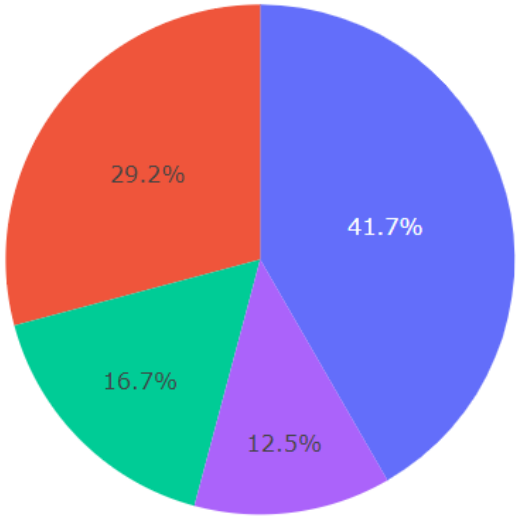
https://github.com/elengler11/AppliedDataScienceCapstone.git

Dashboard with Plotly Dash Results

Scatter plot graphs showing the relationship between payload mass and launch success can be filtered by the dropdown menu for launch site, and a range slider for payload mass. Data point color indicate the booster version used in each launch.

These interactive tools allows users to investigate how payload mass and rocket booster version appear to affect lnding outcomes at each, or all, launch site.

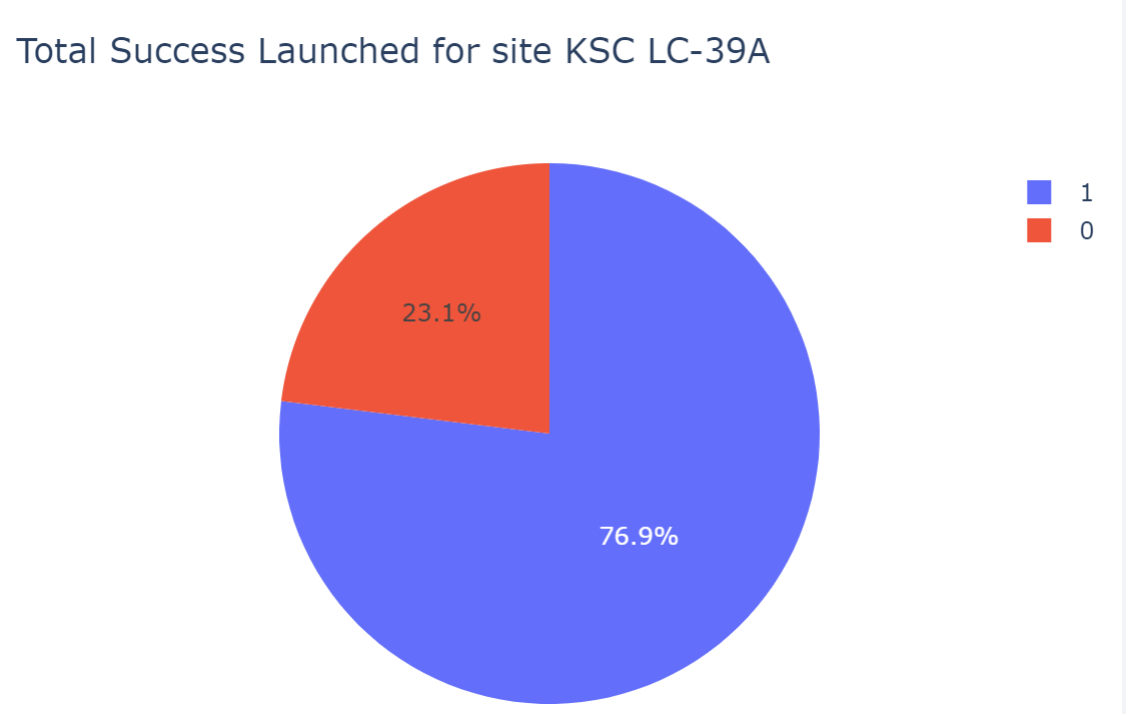
Success Rate by Launch Site



- This pie chart shows that KSC LC-39A has the greatest percentage of successful launches.
- CCAFS SLC-40 has the lowest.
- Other factors, such as number and timing if flights, payload mass, orbit, and booster version are also important components of launch success.

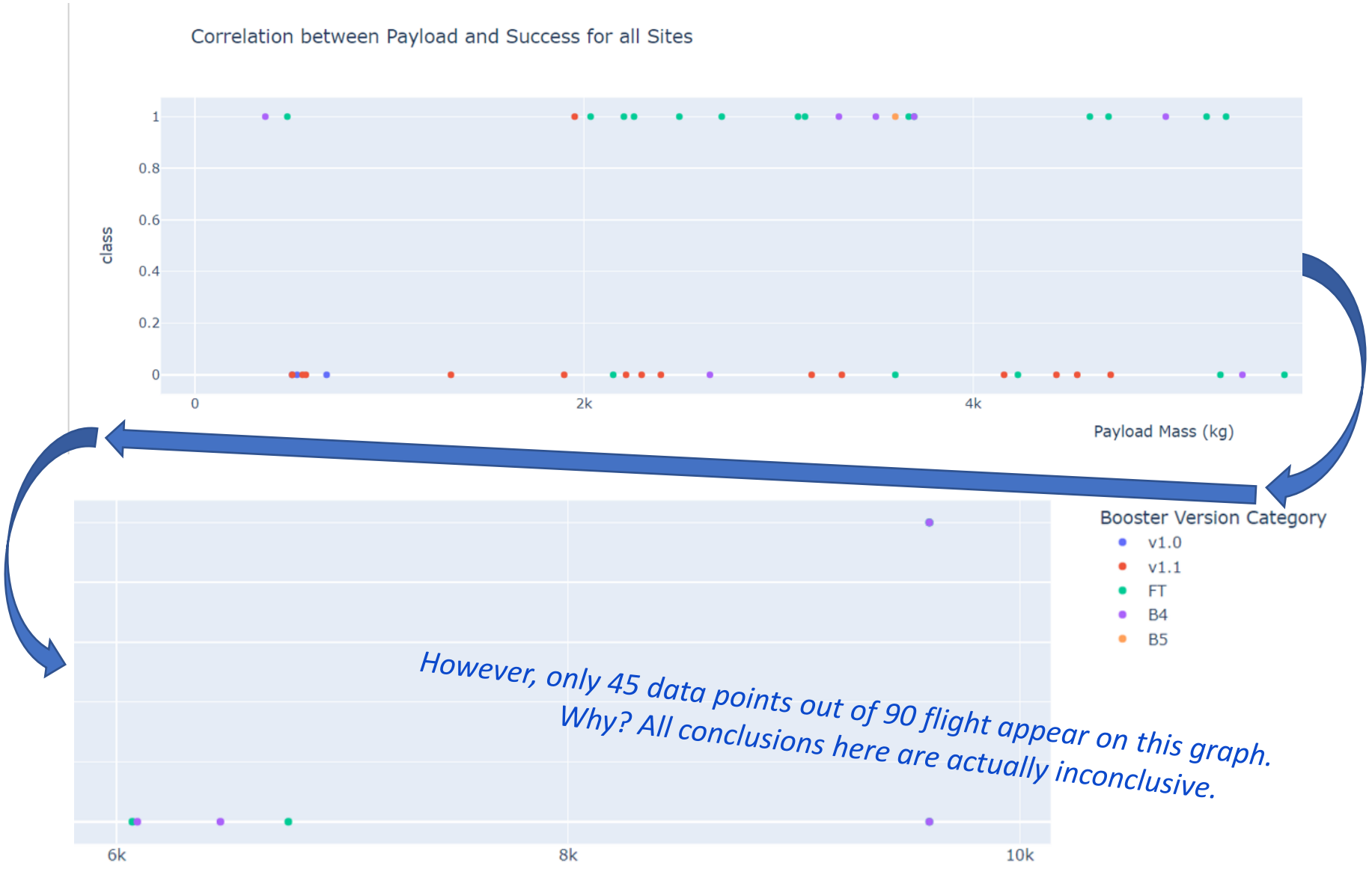
KSC LC-39A has the Highest Success Rate

- KSC LC-39A has the highest launch success rate for all sites.
- More than 75% of flights from KSC LC-39A are successful.

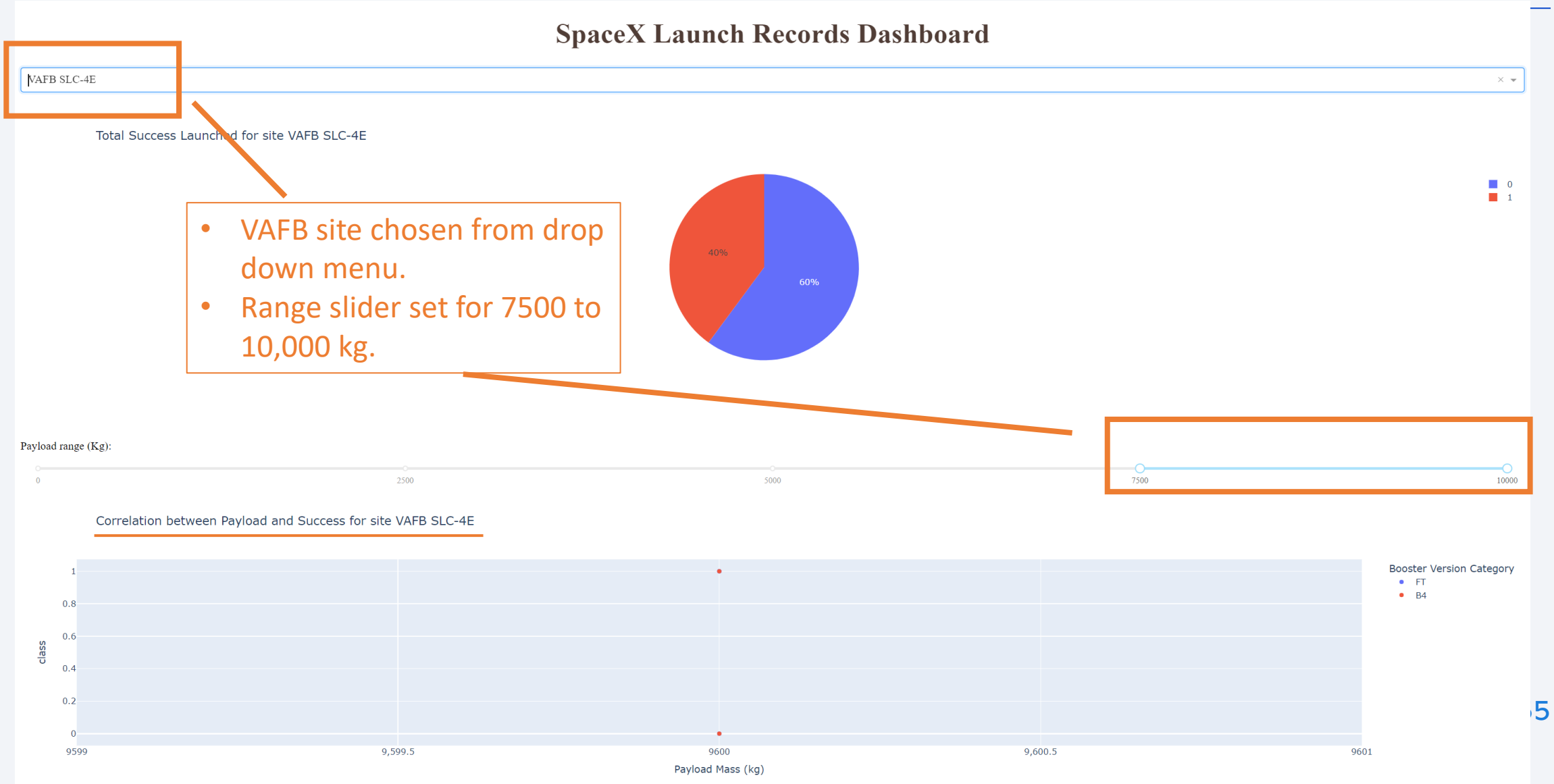


How Does Payload Mass and Booster Version Appear to Effect Launch Success?

- Payload mass is most commonly between 2000 and 6000 kilograms.
- Payload mass has little effect on launch success within this range.
- Payloads outside of this range have lower success, but this effect cannot be segregated from the effect of the booster version.
- V1.0 and V1.1 appear to have very low success.
- FT and B4 appear to be more successful with heavier payloads.



Effect of Payload Mass and Booster Version on Launch Success

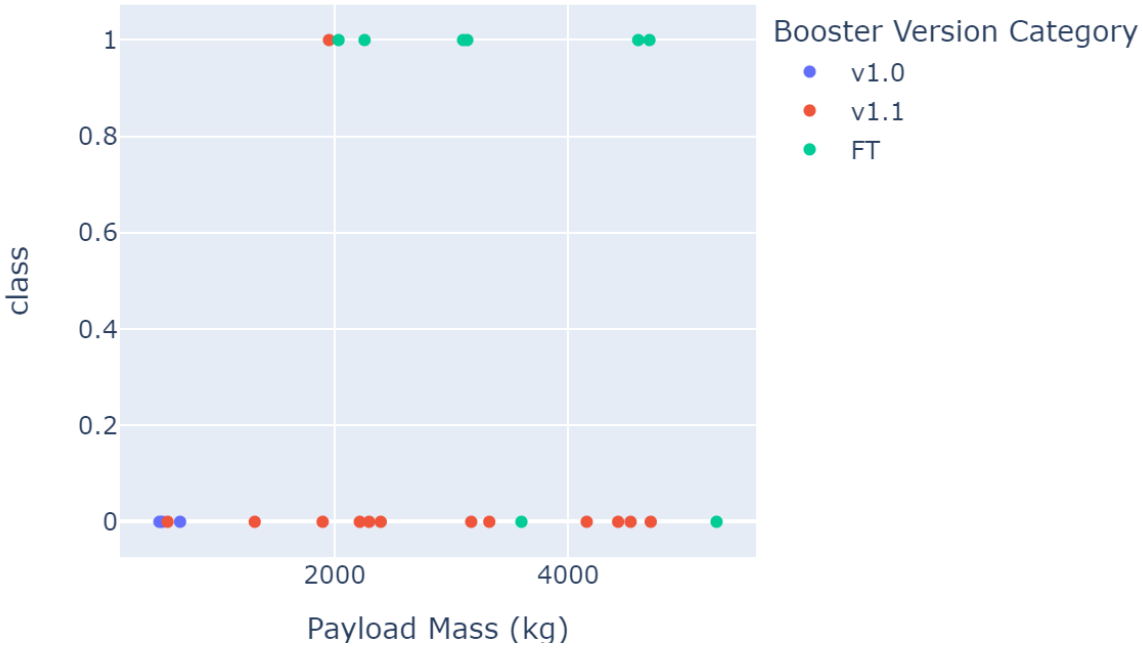


- VAFB site chosen from drop down menu.
- Range slider set for 7500 to 10,000 kg.

CCAFS SLC-40, 0 to almost 6000 kg.

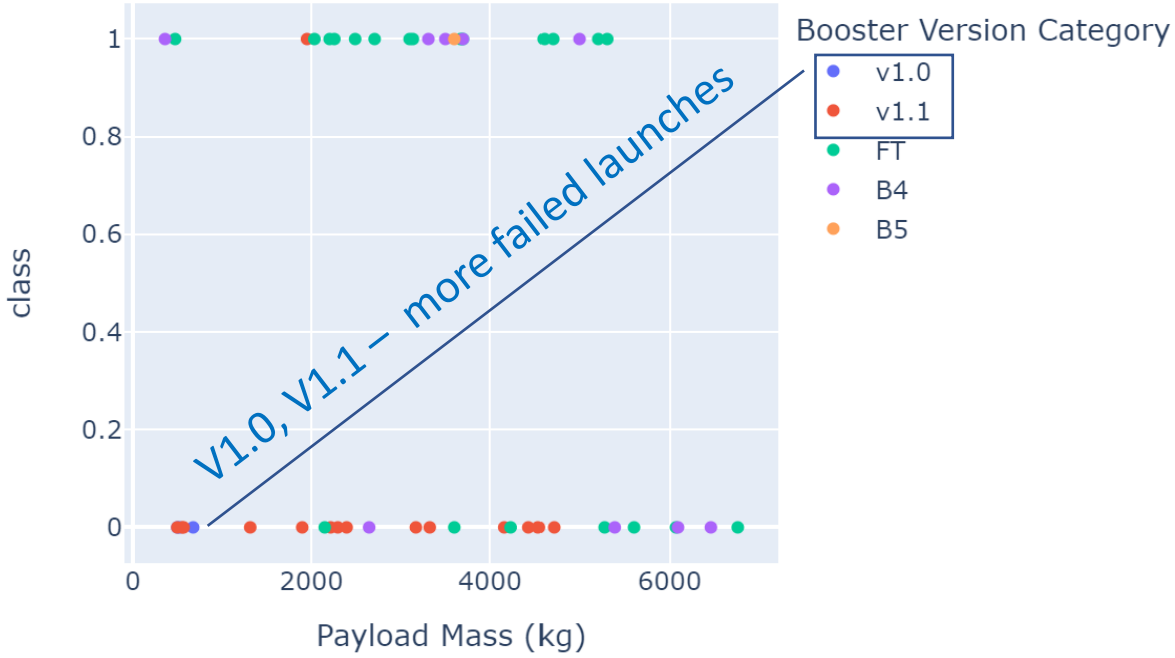
Payload range (Kg):

Correlation between Payload and Success for site CCAFS LC-40



All Sites, 0 to about 7000 kg.

Correlation between Payload and Success for all Sites



Section 6

Results: Predictive Analysis (Classification)

https://github.com/elengler11/AppliedDataScienceCapstone.git

Results: Predictive Analysis (Classification)

The **Dataset** contains the values for each variable and is split into…

Training and Test Data which are subsets of the data set.

train_test_split

The…

Training data is used to fit model parameters or train the model.

estimator object

Parameters are components that are available to a model, such as a penalty in regression, that can be weighted with coefficients or hyperparameters.

Cross validation allows for parameter optimization by using iterations of randomly selected training data subsets to be used as test data. This results in the…

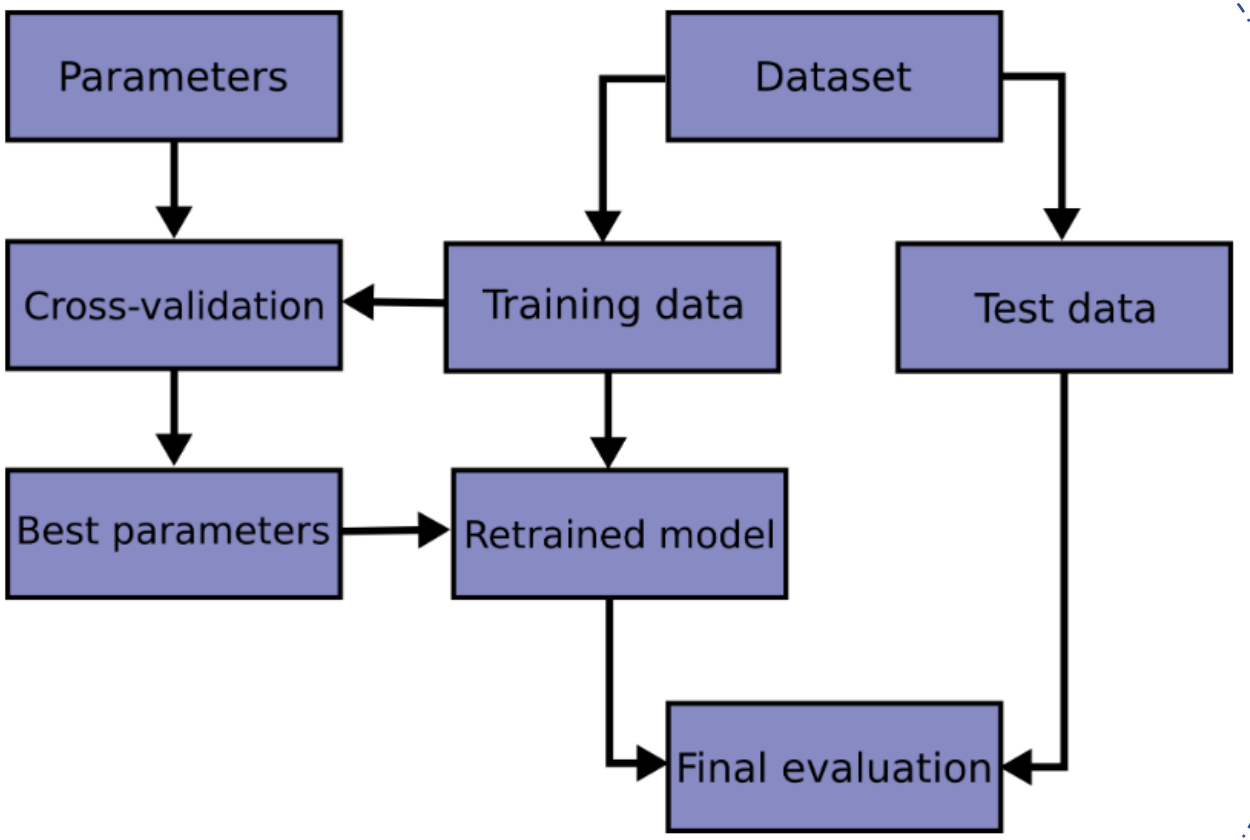
GridSearchCV

Best Parameters for model accuracy in prediction and the model is retrained with these parameters. The…

Retrained Model undergoes…

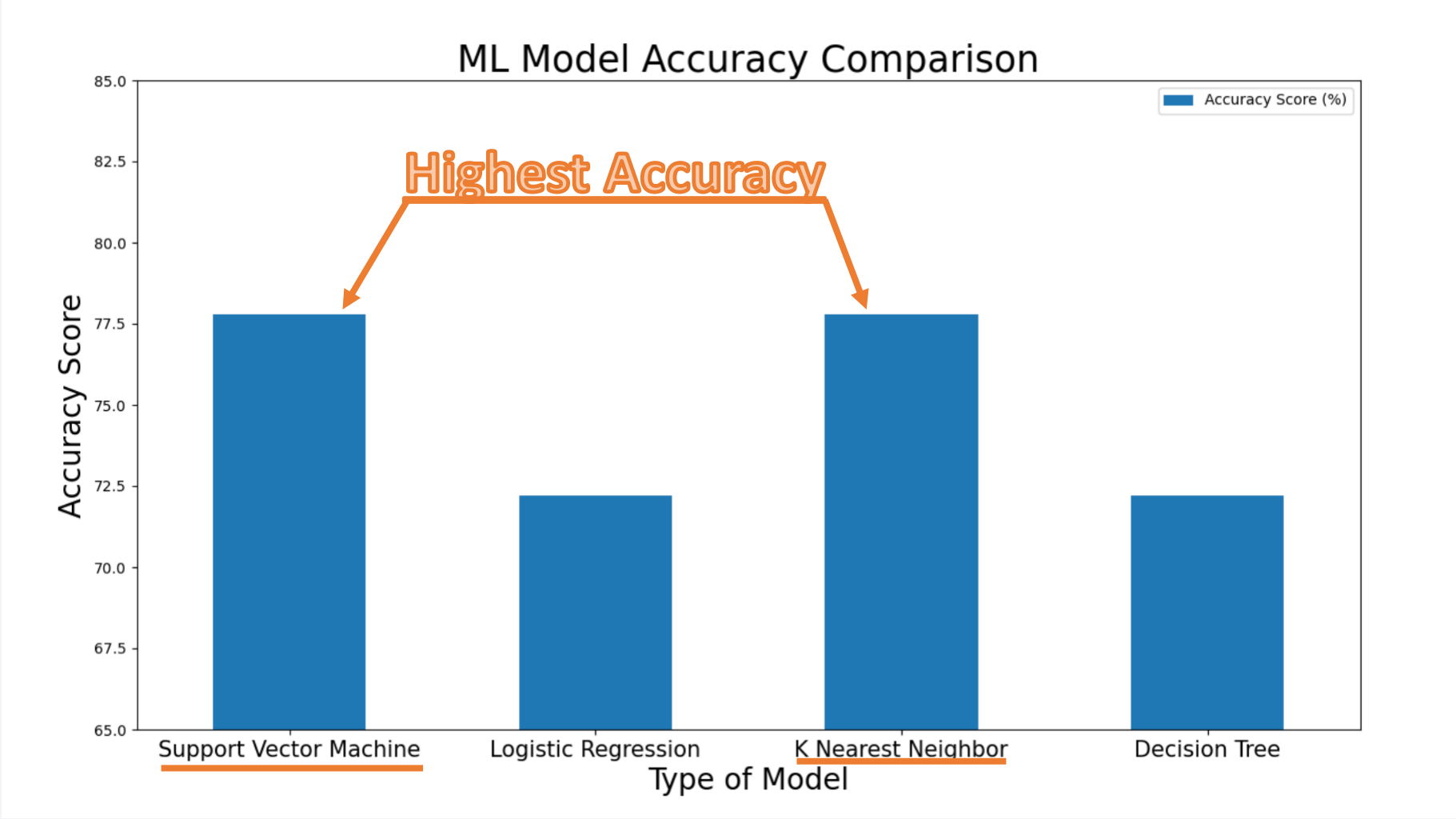
Final Evalution by comparing predicted to actual target values.

confusion matrix, accuracy scores



https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation

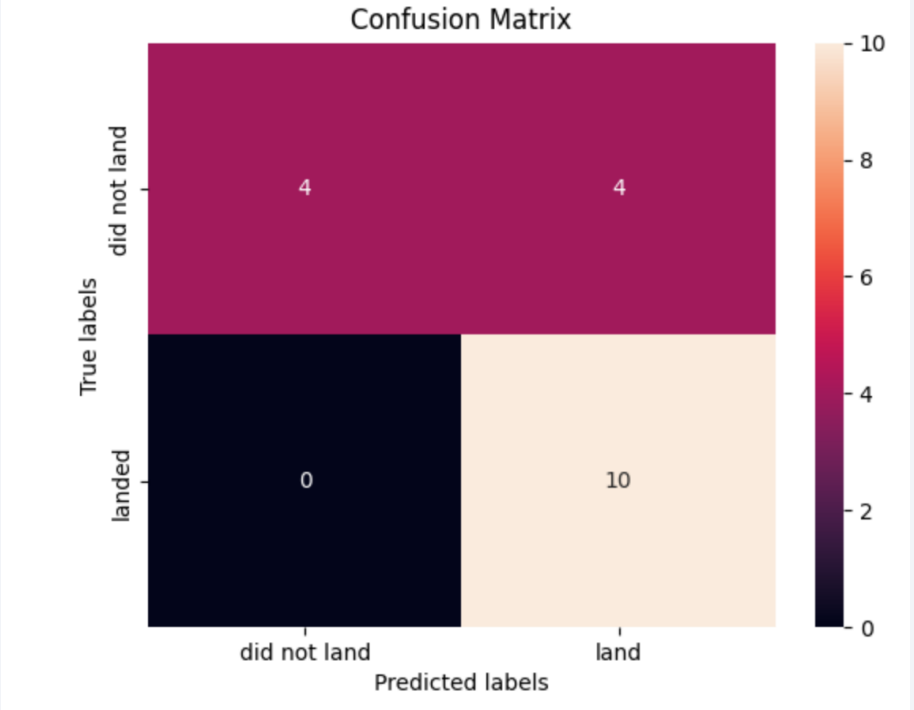
Classification Accuracy



Confusion Matrix Most: Accurate Prediction

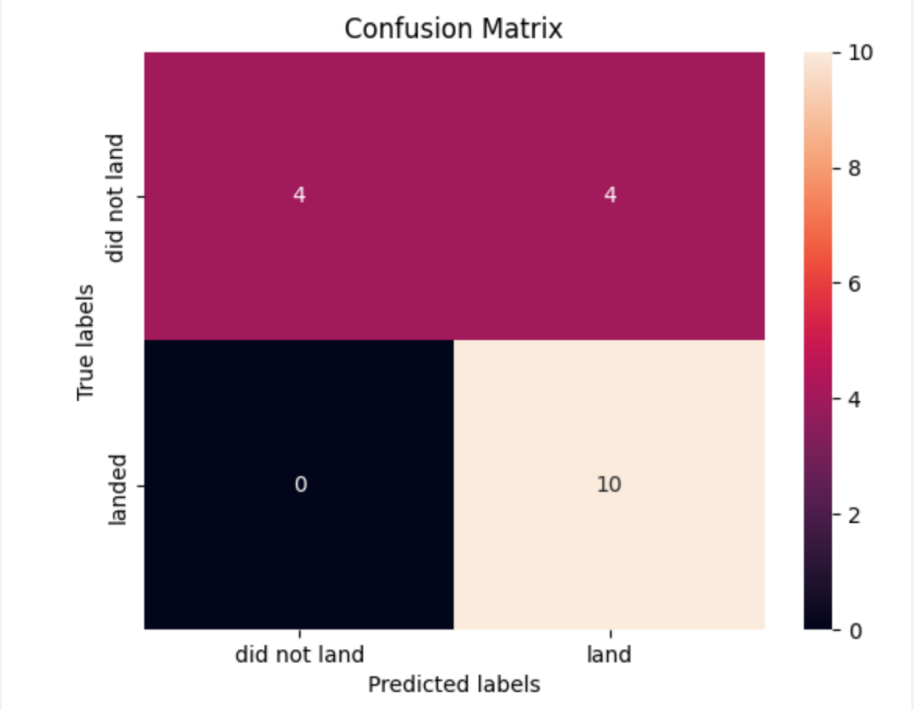
Support Vector Machine

```
yhat=svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



K-Nearest Neighbor

```
yhat = knn_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



A confusion matrix compared actual values compared to predicted values showing corresponding values in upper left and lower right quadrants and disparate values in the lower left (false negative) and upper right (false positive) quadrants. SVM and KNN models have identical confusion matrices and are more accurate than the other two models. The main weakness of these models is false positives (value = 4).

Section 7

Discussion (all analyses)

https://github.com/elengler11/AppliedDataScienceCapstone.git

Discussion

EDA Data Visualization

Launch Site, Flight Number, and Landing Outcome

- Flights 1-20 had the lowest success rate and were almost all at CCAFS SLC 40.
- The success rate of all sites increases with later flights.
- KSC LC 39A was inactive until flight 24.
- VAFB SLC 4E became inactive after flight 64.

Payload, Launch Site, and Landing Outcome

- KSC has a more evenly distributed range of payloads than the other sites.
- VAFB has half its attempts at or below 4000 kg while the remaining half equaled 10,000 kg.
- CAFS has the largest number of payloads above 14,000 kg.

Flight Number, Payload, and Landing Outcome

- Most attempts have a payload below 8000 kg.
- Larger payloads generally have been on later flights.
- Landing success generally increases with payload.
- The success rate levels off past 12,000 kg.

Orbit Type and Landing Outcome

- ESL1, GEO, HEO, and SSO orbits have 100% success, but only SSO had more than 1 attempt..
- GTO has the lowest success and the largest number of attempts.
- VLEO has the highest success rate for any orbit with more than five attempts.

Discussion

EDA Data Visualization (continued)

Orbit Type and Flight Number

- EO, ISS, PO, & GTO orbits success rate appear to increase along with flight number/later flights.
- Fewer attempts in the LEO, ISS, PO, and GTO orbits are made in flight numbers greater than 60, while other orbit type increase with the exception of ES-L1.

Orbit Type and Payload Mass

- LEO, ISS, and PO **orbits** appear to have more successful landings with increasing payload mass. Several orbit types have only one attempt limiting analyses.

Success Rate by Year

- Landing success rate generally increases by year.

Discussion

EDA SQL database queries

- ✓ Names of the unique launch sites: CCAFS LC 40, CCAFS SLC 40, VAFB SLC 4E, KSC LC39A
- ✓ Total payload mass carried by boosters launched by NASA (CRS): 45,596 kg.
- ✓ Average payload mass carried by booster version F9 v1.1: 2928 kg
- ✓ Date when the first successful landing outcome in ground pad was achieved: Dec. 22, 2015
- ✓ Names of boosters with successful drone ship landing and payload mass between 4000 and 6000 kg.:
  F9 FT B10…22, 26, 21.2, 31.2
- ✓ Names of the booster_versions which have carried the maximum payload mass:
  F9 FT B10…48.4, 49.4, 51.3, 56.4, 48.5, 51.4, 49.5, 60.2, 58.3, 51.6, 60.3, 49.7

***See results section for addional queries and responses*

Discussion

Folium Maps

- Three launch sites are located in Florida near Cape Canaveral and one is located in southern California.
- Marker clusters provide location and each marked success/failed landing attempt.
- Distances can be calculated to key transport features for cost analysis, and can be marked on the map with polylines.

Discussion

Plotly Dash Interactive Dashboard

- The dashboard allows users to filter launch site locations with a drop-down menu and payload mass with a range slider.
- The pie chart responds to user input with landing success rate by each or all launch sites.
  - The success rates are displayed without the number of launches per site and these values vary greatly.
- The scatterplot responds to user input with data points indicating payload mass, booster version, and landing outcome.

Discussion

First Stage Landing Prediction Machine Learning Models

- The ML models (Logistic Regression, Decision Tree, SVM, and KNN) included variables excluded from other analysis such as grid fin and leg use and reuse and serial numbers. Landing site was not included.
- The prediction accuracy scores calculated with the test data subsets was highest for both the SVM and KNN models (77%).
- The major weakness with these models is the occurrence of false positive landing attempt predictions (4 out of 18 samples or 22%).
- The logistic regression and decision tree models both had prediction accuracy scores of 72%.
- The logistic regression and decision tree models confusion matrices both had 5 out 18 false positives (28%).
- Additional analyses, and perhaps data, are required to build a more accurate model for predicting successful first stage landings.

Section 8

Conclusions (from all analyses)

https://github.com/elengler11/AppliedDataScienceCapstone.git

Conclusion

Overall landing success (Falcon 9) has increased year-by-year. Reasons for this appear to include the following:

- **Payload mass** is most commonly between 2000 and 6000 kilograms and has little effect on launch success within this range. Payloads outside of this range have lower success, but this effect cannot be segregated from the effect of the booster version.
  - V1.0 and V1.1 **boosters** appear to have very low overall success, while FT and B4 appear to be more successful with heavier payloads.
- LEO, ISS, and PO **orbits** appear to have more successful landings with increasing payload mass. Several orbit types have only one attempt limiting analyses. SSO orbit is the most successful overall (100%), but only has five attempts. ISS and GTO orbits have the most attempts and lowest success rates.
- **Launch site** and **flight number** variables are appear to be confounded by other variables including but not limited to booster version, payload mass, and orbit type.

Conclusions: First Stage Landing Prediction Models

- The ML models (Logistic Regression, Decision Tree, SVM, and KNN) included additional variables including grid fin and leg use and reuse and serial numbers. Landing site was not included.
- The prediction accuracy scores calculated with the test data subsets was highest for both the SVM and KNN models (77%).
- The major weakness with these models is the occurrence of false positive landing attempt predictions (4 out of 18 samples or 22%).
- The logistic regression and decision tree models both had prediction accuracy scores of 72%.
- The logistic regression and decision tree models confusion matrices both had 5 out 18 false positives (28%).
- Additional analyses, and perhaps data, are required to build a more accurate model for predicting successful first stage landings.

Section 9

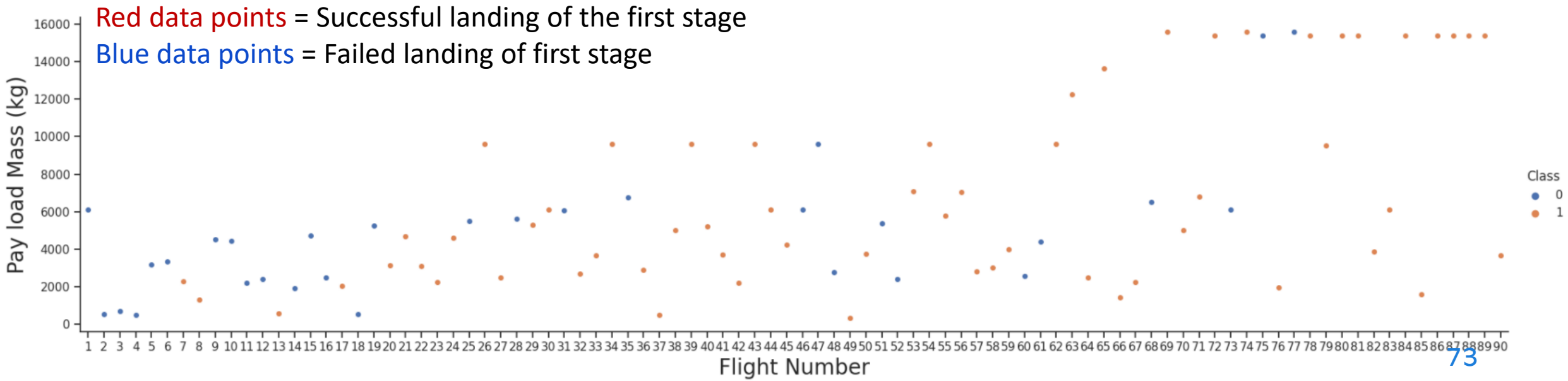
Appendix

Appendix

EDA with Data Visualization: Flight # vs. Payload Mass

- ✓ Successful landings of the first stage increase with the number of attempts and payload mass.

```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect = 4)
sns.set_style("ticks")
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```

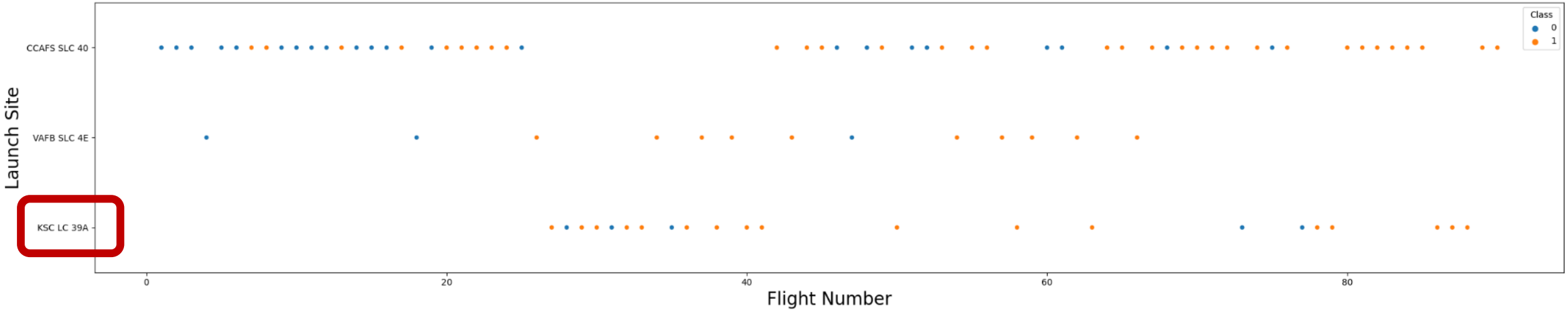


EDA with Data Visualization: Flight # vs. Launch Site

- ✓ KSC LC 39A has the highest success rate.

Launch Site	Success Rate
CCAFS SLC 40	60.000000
VAFB SLC 4E	76.923077
KSC LC 39A	77.272727

```
sns.swarmplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

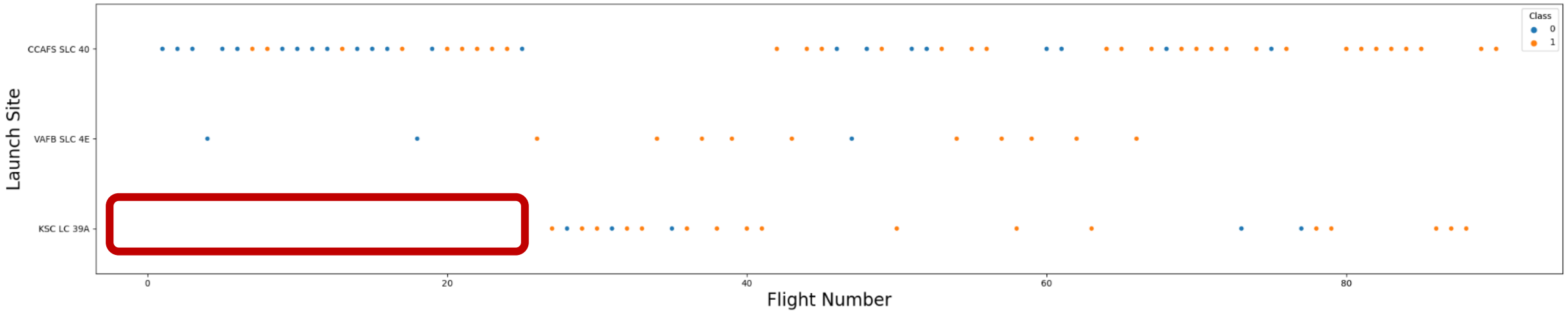


EDA with Data Visualization: Flight # vs. Launch Site

- ✓ However, KSC LC 39A was inactive for about the first third of flight attempts.

Launch Site	Success Rate
CCAFS SLC 40	60.000000
VAFB SLC 4E	76.923077
KSC LC 39A	77.272727

```
sns.swarmplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

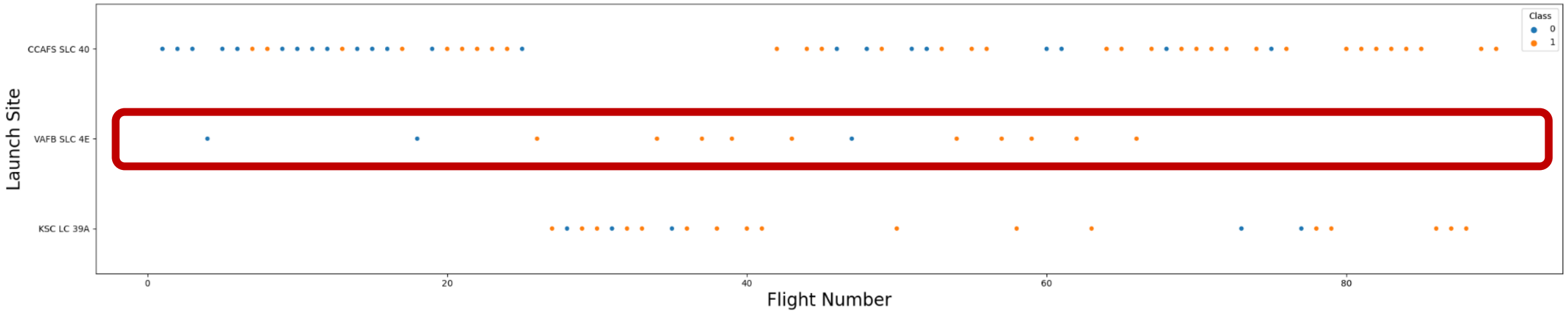


EDA with Data Visualization: Flight # vs. Launch Site

- ✓ VAFB SLC 4E had the fewest attempts while CCAFS SLC 40 had the most.
- ✓ VAFB SLC 4E did not launch after about 70 flight attempts.

Launch Site	Success Rate
CCAFS SLC 40	60.000000
VAFB SLC 4E	76.923077
KSC LC 39A	77.272727

```
sns.swarmplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

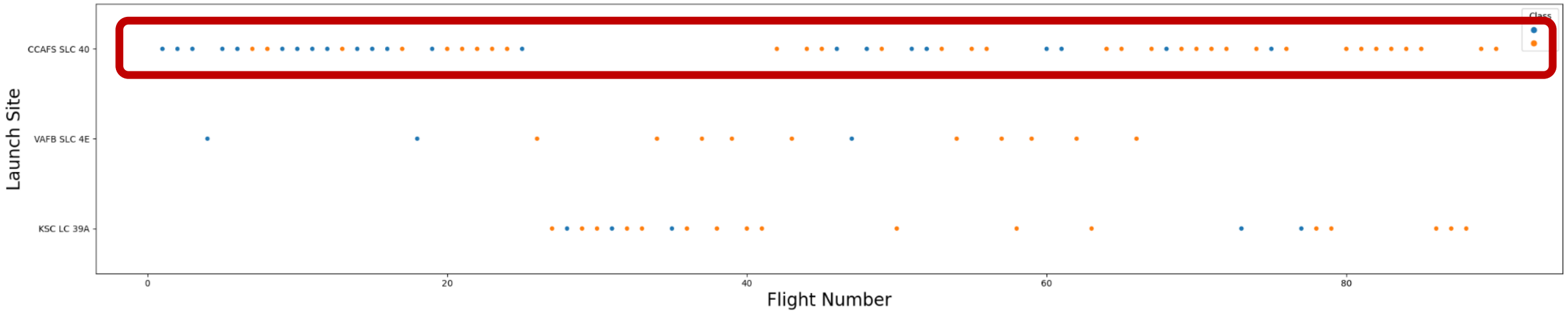


EDA with Data Visualization: Flight # vs. Launch Site

- ✓ CCAFS SLC 40 has the lowest success rate.
- ✓ CCAFS SLC 40 carried out almost all of the earliest 30 flight attempts.

Launch Site	Success Rate
CCAFS SLC 40	60.000000
VAFB SLC 4E	76.923077
KSC LC 39A	77.272727

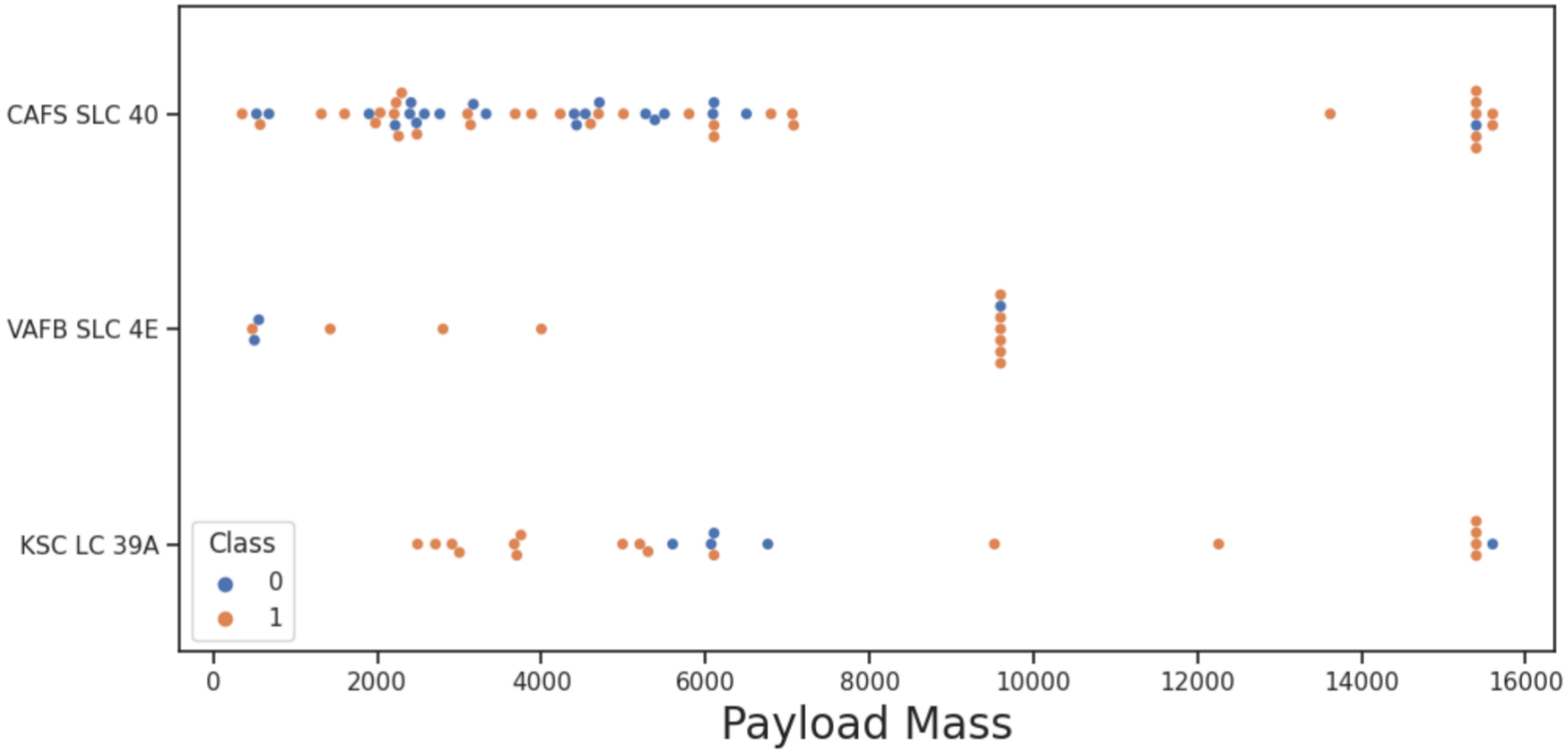
```
sns.swarmplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



EDA with Data Visualization: Launch Site v. Payload Mass

- ✓ CCAFS SLC 40 appears to have lighter payloads while KSC LC 39A seems to have heavier payloads.

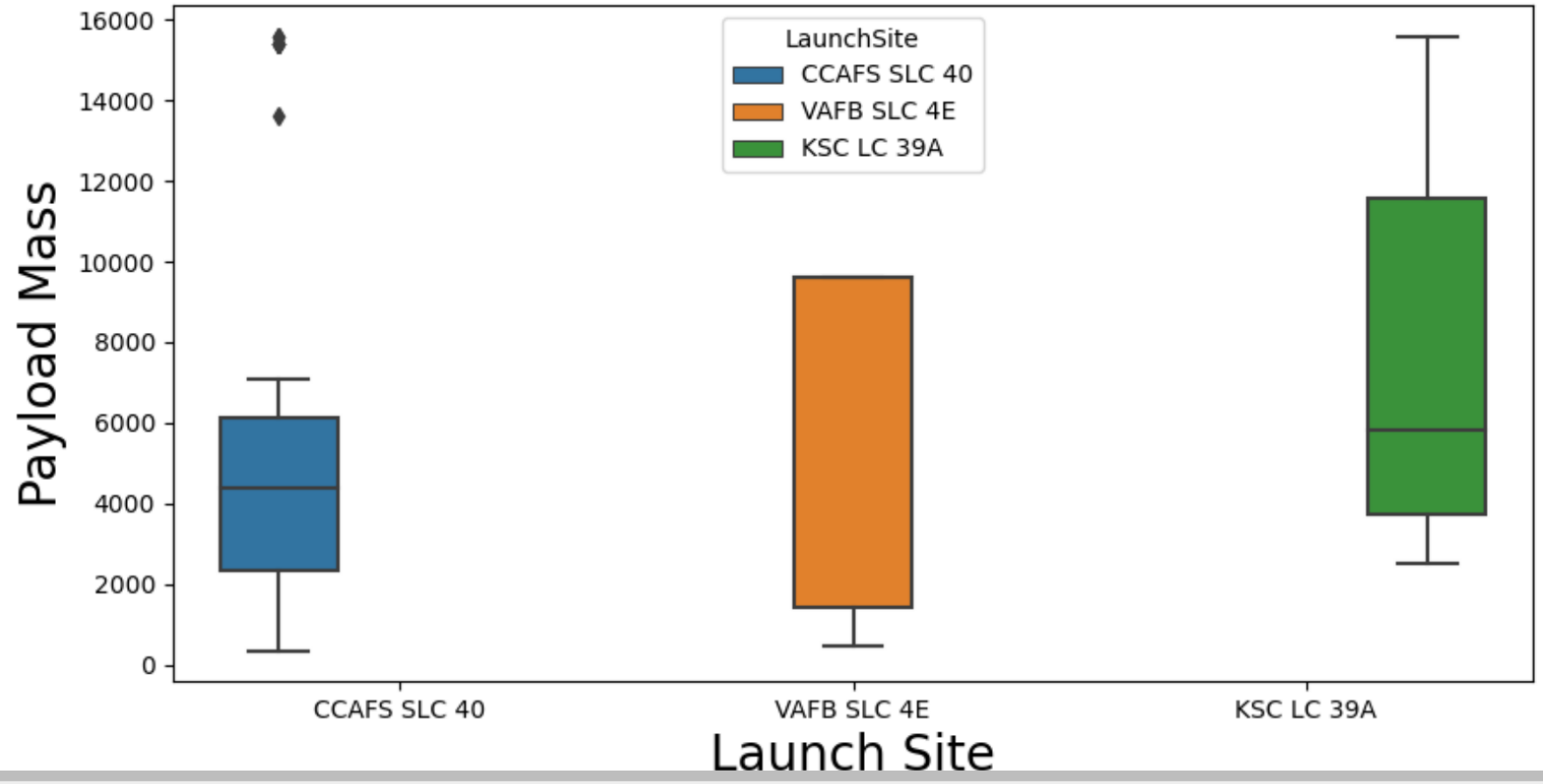
```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the clas
sns.swarmplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df)
plt.xlabel("Payload Mass",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



EDA with Data Visualization: Launch Site v. Payload Mass

- ✓ This box plot confirms the speculation drawn from the scatter plot on the previous slide.

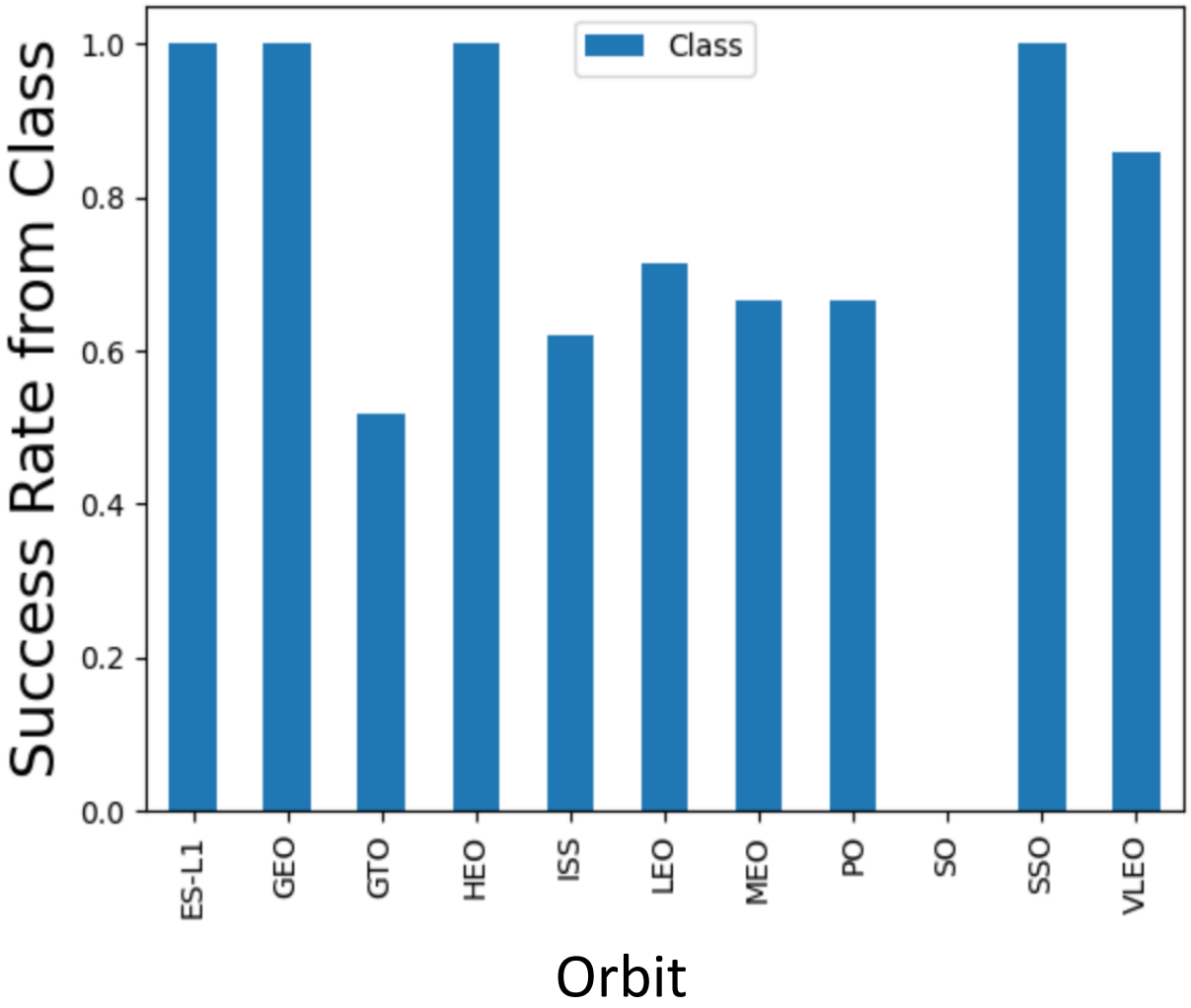
```
sns.boxplot(x="LaunchSite", y="PayloadMass", hue="LaunchSite", data=df)
plt.ylabel("Payload Mass",fontsize=20)
plt.xlabel("Launch Site",fontsize=20)
plt.show()
```



EDA with Data Visualization: Orbit v. Success Rate

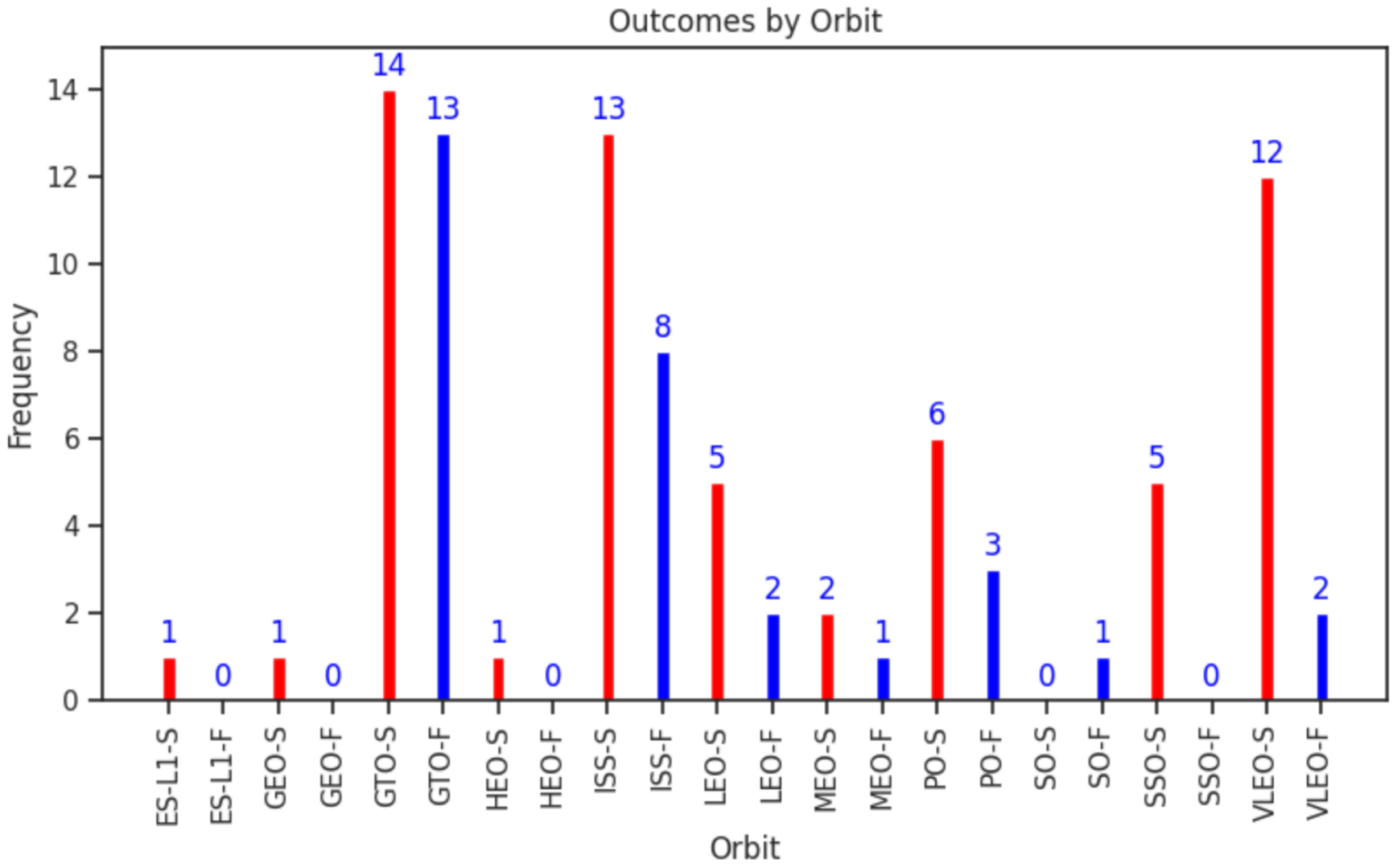
- ✓ ES-L1, GEO, HEO, SSO have 100% success.
- ✓ GTO has the lowest success rate at 52%.
- ✓ The average success rate is 66%.

```
df_group = df[['Orbit','Class']]
df_group = df_group.groupby(df["Orbit"],as_index=True).mean()
print(df_group)
df_group.plot(kind='bar')
plt.title = 'Class v. Orbit'
plt.xlabel("Orbit",fontsize=20)
plt.ylabel("Success Rate from Class",fontsize=20)
plt.show()
```



EDA with Data Visualization: Orbit v. Success Rate

- ✓ The sample size for each orbit is now displayed.
- ✓ Several orbits have a sample size(n) of one.
- ✓ Considering orbits with n>10, VLEO has the highest success rate while GTO is lowest.



EDA with Data Visualization: Orbit v. Success Rate

```
barcolor=[]
orbits = ('ES-L1 S', 'ES-L1 F', 'GEO S', 'GEO F', 'GTO S', 'GTO F', 'HEO S','HEO F', 'ISS S', 'ISS F', 'LEO S', 'LEO F', 'MEO S',
for s in orbits:
    if 'F' in s:
        barcolor.append('blue')
    else:
        barcolor.append('red')
print(barcolor)
```

```
['red', 'blue', 'red', 'blue', 'red', 'blue', 'red', 'blue', 'red', 'blue', 'red', 'blue', 'red', 'blue', 'red', 'blue', 'red',
'blue', 'red', 'blue', 'red', 'blue']
```

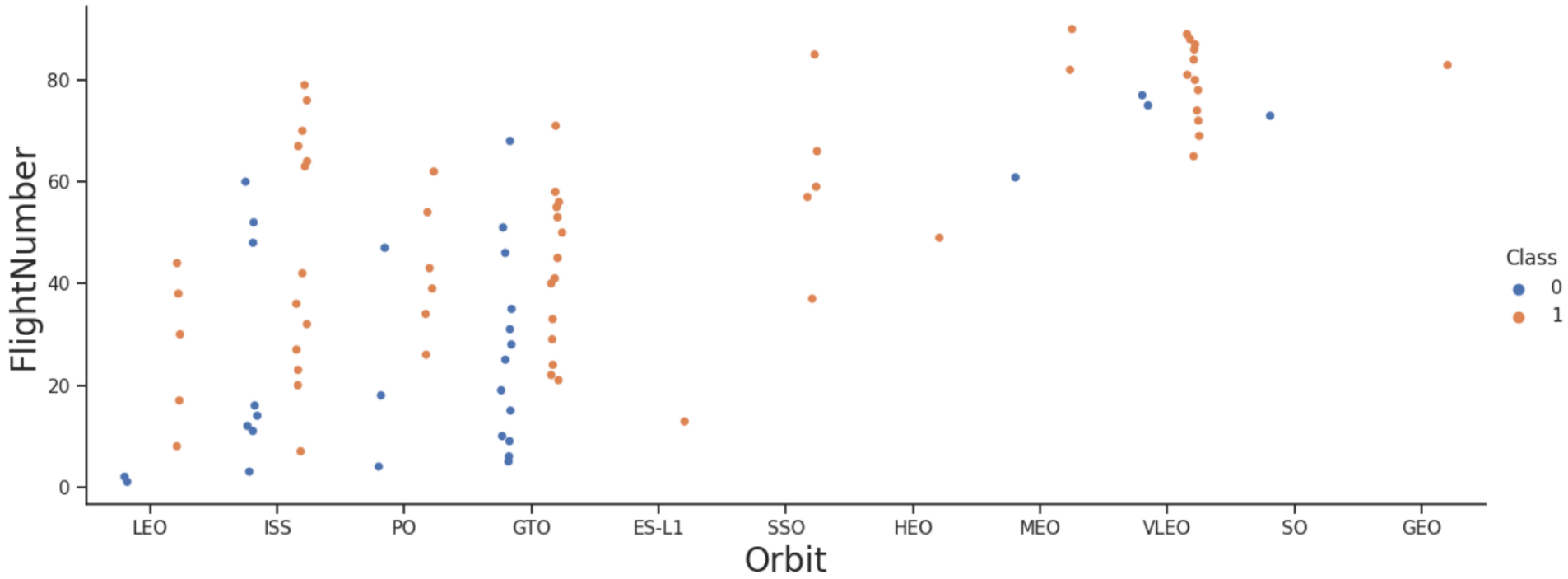
```
orbits = ('ES-L1-S', 'ES-L1-F', 'GEO-S', 'GEO-F', 'GTO-S', 'GTO-F', 'HEO-S','HEO-F', 'ISS-S', 'ISS-F', 'LEO-S', 'LEO-F', 'MEO-S',
SorF = {'Frequency': (1, 0, 1, 0, 14, 13, 1, 0, 13, 8, 5, 2, 2, 1, 6, 3, 0, 1, 5, 0, 12, 2)}

x = np.arange(len(orbits))  # the label locations
width = 0.25  # the width of the bars
multiplier = 0
fig, ax = plt.subplots(figsize=(8,5), layout='constrained')
for attribute, Frequency in SorF.items():
    offset = width * multiplier
    rects = ax.bar(x + offset, Frequency, width, color=barcolor)
    ax.bar_label(rects, padding=3, color=color)
    multiplier += 1
ax.set_ylabel('Frequency')
ax.set_xlabel('Orbit')
ax.set_title('Outcomes by Orbit')
ax.set_xticks(x, orbits, rotation=90)
ax.set_ylim(0, 15)

plt.show()
```

✓ Code for previous slide

EDA with Data Visualization: Orbit v. Flight Number

- ✓ LEO, ISS, PO, & GTO orbits success rate appear to increase along with flight number/later flights.

```
### TASK  4: Visualize the relationship between FlightNumber and Orbit type
sns.catplot(y="FlightNumber", x="Orbit", hue="Class", data=df, aspect = 2.5, dodge=True)
plt.xlabel("Orbit",fontsize=20)
plt.ylabel("FlightNumber",fontsize=20)
plt.show()
```

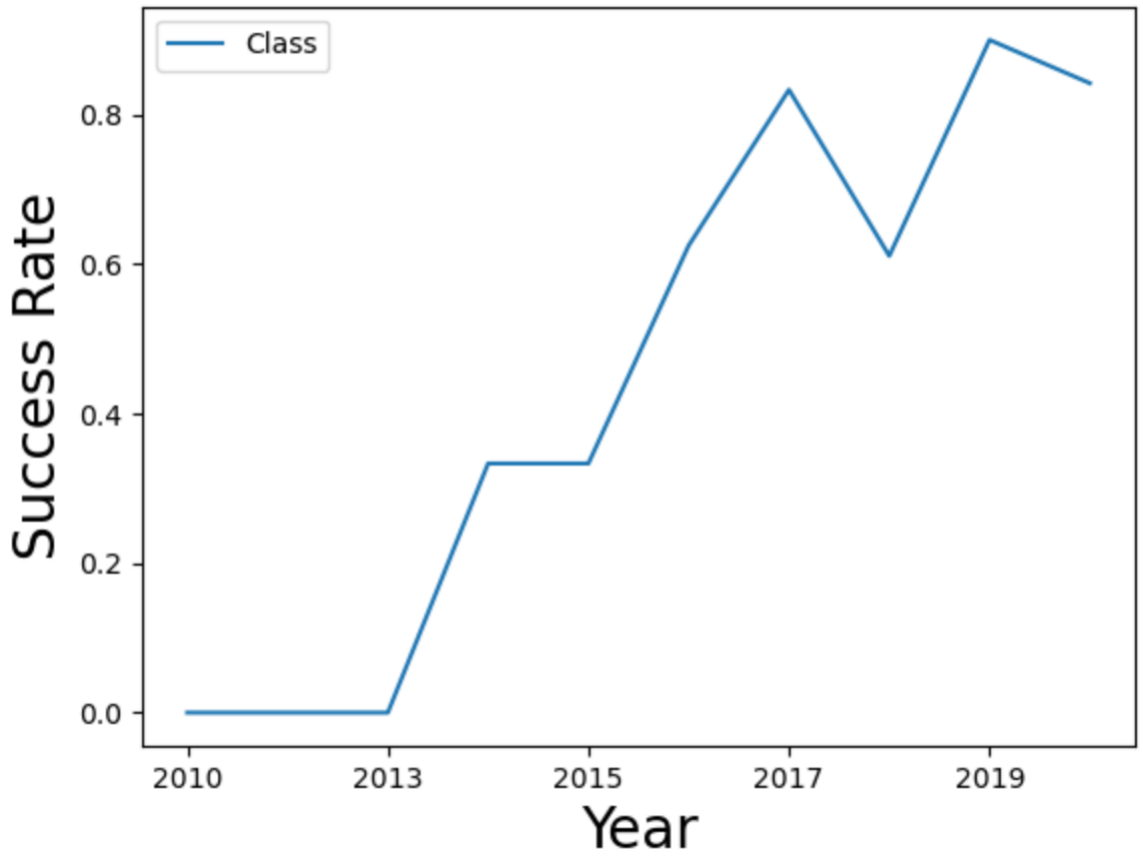


EDA with Data Visualization: Success Rate by Year

✓The success rate generally increases over time.

```
# A function to Extract years from the date
year=[]
def Extract_year():
    for i in df["Date"]:
        year.append(i.split("-")[0])
    return year
Extract_year()
df['Date'] = year
df.head()
```

```
# Plot a line chart with x axis to be the extracted year and y axis
#to be the success rate
df_groupline = df[['Date', 'Class']]
df_groupline = df_groupline.groupby(df['Date'],as_index=True).mean()
print(df_groupline)

df_groupline.plot(kind='line')
plt.title = 'Year v. Success Rate'
plt.xlabel("Year",fontsize=20)
plt.ylabel("Success Rate",fontsize=20)
plt.show()
```



EDA with SQL

- **Names of the unique launch sites in the space mission**

%sql select distinct "Launch_Site" from SPACEXTBL;

- **Five records where launch sites begin with the string 'CCA'**

%sql select * from SPACEXTBL where "Launch_Site" like "CCA%" limit 5;

- **Total payload mass carried by boosters launched by NASA (CRS)**

%sql select SUM ("PAYLOAD_MASS__KG_") FROM SPACEXTBL where "Customer" = "NASA (CRS)";

- **Average payload mass carried by booster version F9 v1.1**

%sql select AVG ("PAYLOAD_MASS__KG_") FROM SPACEXTBL where "Booster_Version" = "F9 v1.1";

- **Date when the first successful landing outcome in ground pad was achieved.**

%sql select "Date" FROM SPACEXTBL where "Landing_Outcome" = "Success (ground pad)" limit 1;

- **Names of the boosters which have success in drone ship and have payload mass between 4000 and 6000**

%sql select "Booster_Version" FROM SPACEXTBL where "Landing_Outcome" = "Success (drone ship)" and "PAYLOAD_MASS__KG_" between 4000 and 6000;

EDA with SQL

- **Total number of successful and failure mission outcomes**

%sql SELECT COUNT ("Mission_Outcome") FROM SPACEXTBL group by "Mission_Outcome";

- **Names of the booster_versions which have carried the maximum payload mass.**

%sql SELECT "Booster_Version" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT max ("PAYLOAD_MASS__KG_") FROM SPACEXTBL);

- **Records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015**

%sql select substr(Date, 4, 2), "Landing_Outcome", "Booster_Version", "Launch_Site" from SPACEXTBL WHERE substr(Date,7,4)='2015' and "Landing_Outcome" == "Failure (drone ship)";

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

%sql select "Date", "Mission_Outcome" from SPACEXTBL where ("Mission_Outcome" == "Success") and substr(Date,7,4) between '2010' and '2017;

Build an Interactive Map with Folium

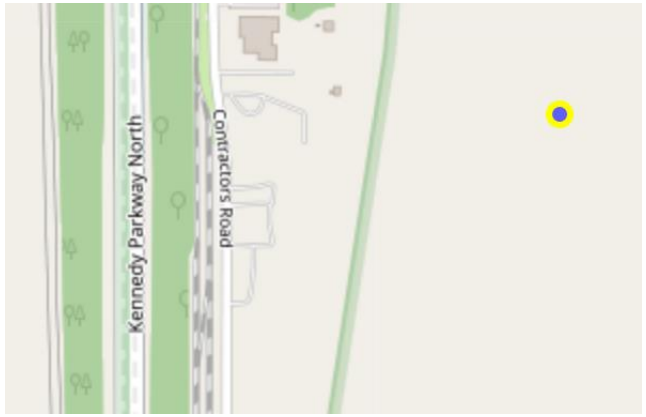
TASK 1: Mark all launch sites on a map.

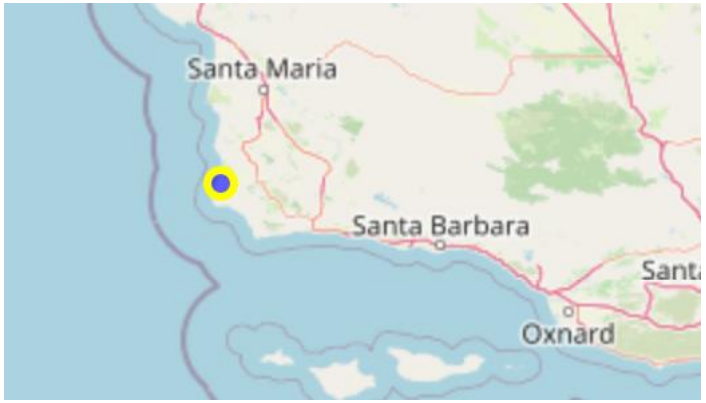
Why? To see the locations of each launch site.

Center and set zoom level of the folium map to show the continental US

Instantiate a feature group for the launches in the dataframe and locate on the folium map with latitude and longitude coordinates.

Loop through the launches and add each to the incidents feature group with a yellow and blue circle marker with a radius of 5.





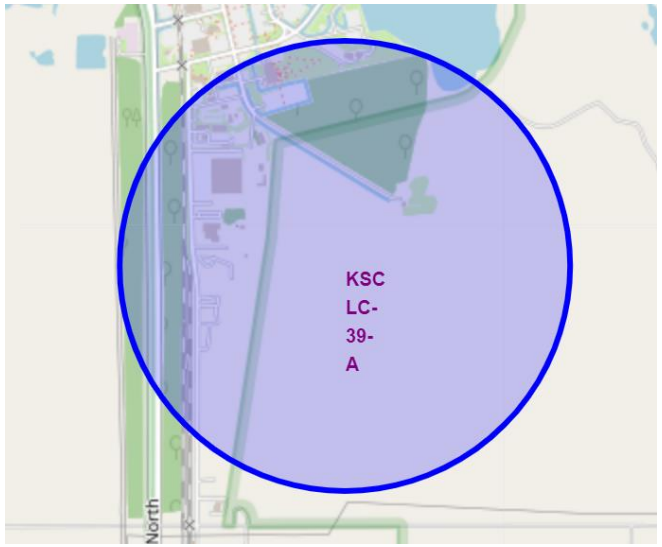
Map with Folium

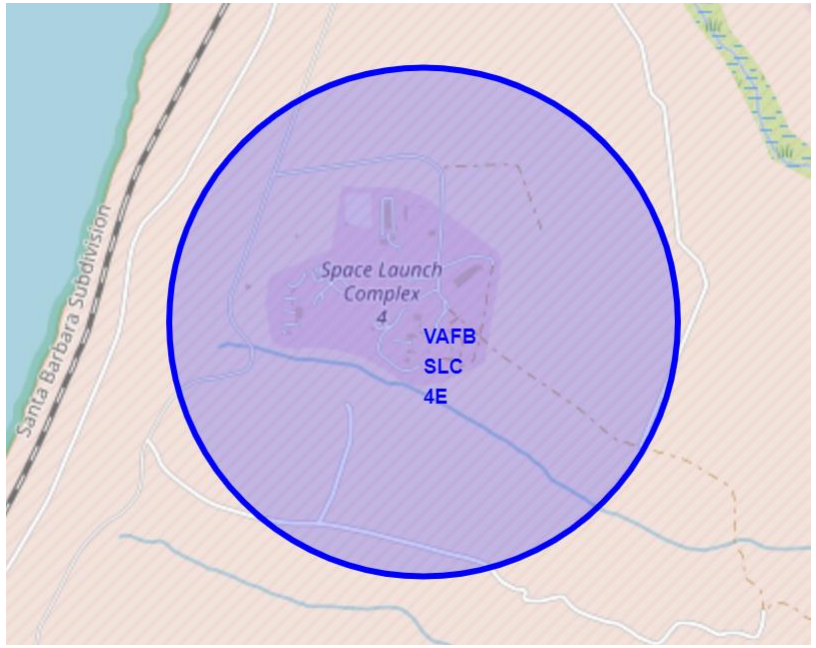
TASK 1: Mark all launch sites on a map.

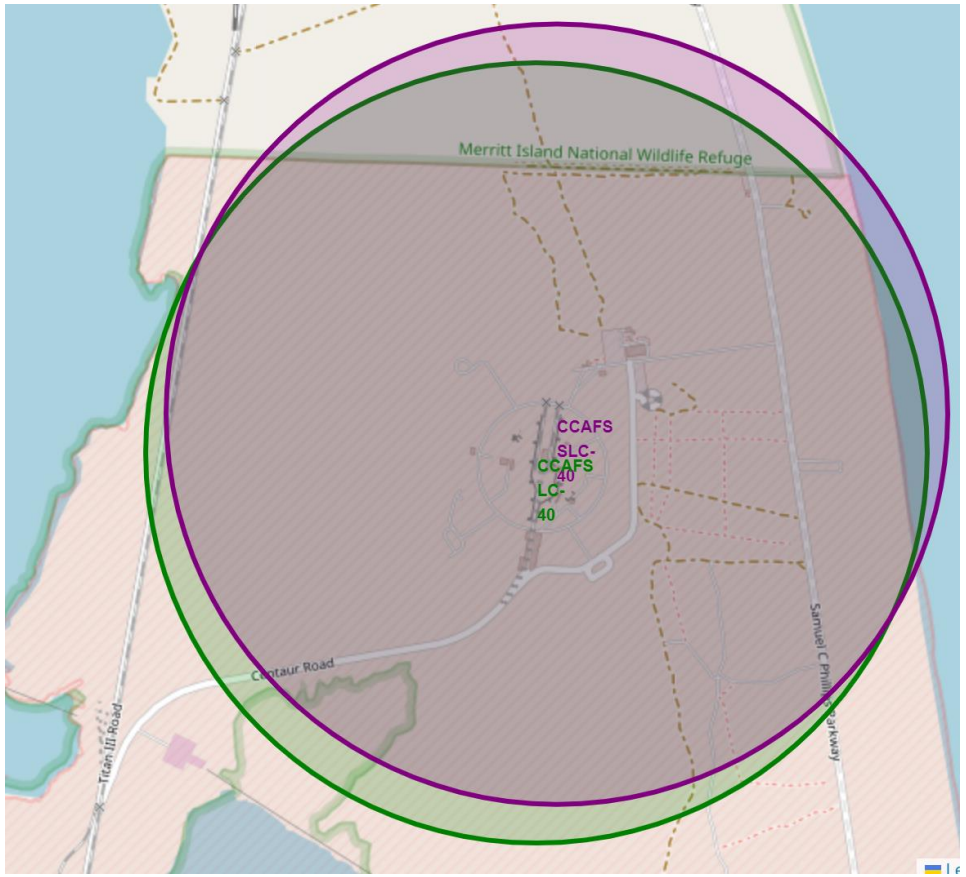
Why? To see the locations of each launch site.

Folium circles and markers then used to mark the site with different colors and text labels.

```
circle = folium.Circle(CCAFS_LC_40_Coordinate, radius=1000, color='green',
fill=True).add_child(folium.Popup('CCAFS LC-40'))
marker = folium.map.Marker(
    CCAFS_LC_40_Coordinate,
icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:green;"><b>%s</b></div>' % 'CCAFS
LC-40',
        )
    )     # for each launch site
```





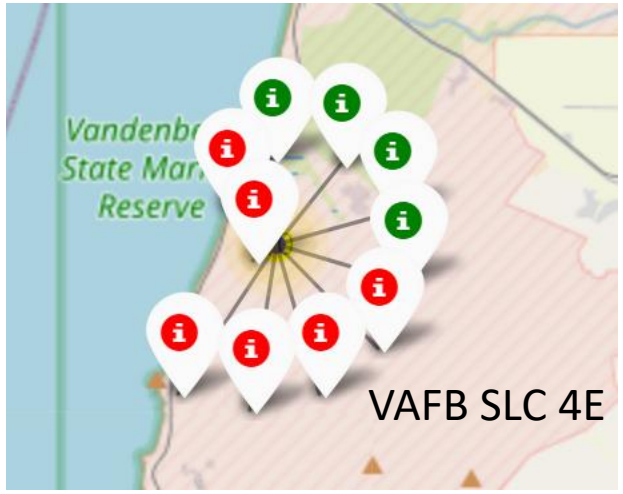


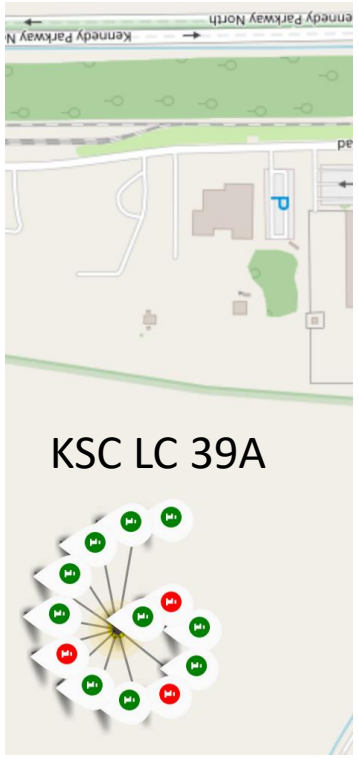
Build an Interactive Map with Folium

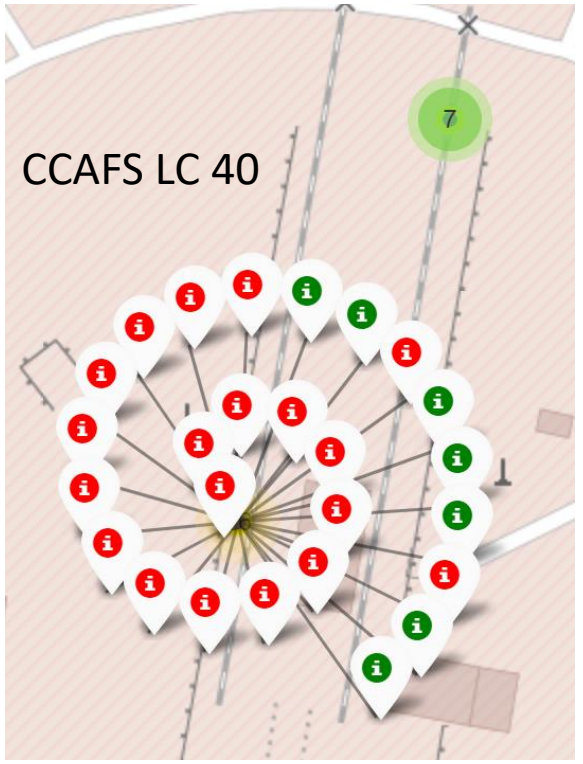
TASK 2: Mark the success/failed launches for each site on the map.

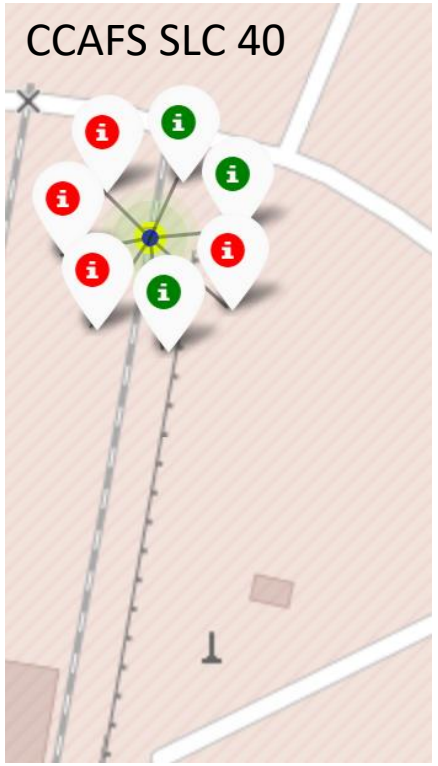
Why? Visualize launch success rates and relative use of each site.

Folium marker clusters were used to identify locations of all successful (green) and failed (red) launches.









Build an Interactive Map with Folium

TASK 3: Calculate the distances between a launch site to its proximities.

- Why? Cost analysis component for materials transport.

Distances were calculated with a mouse_position function where a clicked point is resolved to lat/long coodinates, and a distance function based on trigonometry. **Markers, polylines, and text** were added to indicate proximity to mapped features such as coastlines and rail-lines.

```
distance_marker = folium.Marker(
    coastline_coordinate,
    icon=DivIcon(icon_size=(20,20), icon_anchor=(0,0),
    html='<div style="font-size: 12;
color:#d35400;"><b>%s</b></div>' % "{:10.2f}
KM".format(distance_coastline),))
lines=folium.PolyLine(locations=
(coastline_coordinate,CCAFS_SLC_40Coordinate),
weight=1)
spacex_map.add_child(lines)
```

SpaceX Launch Records Dashboard

All Sites

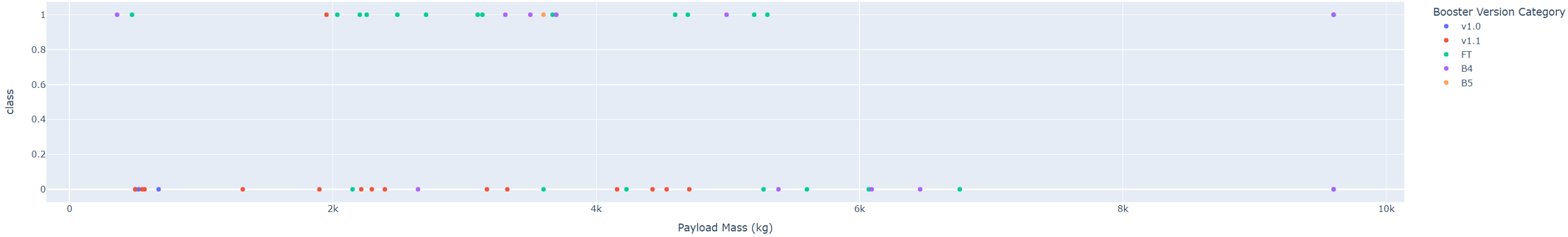
Build a Dashboard with Plotly Dash

Total Success Launches By Site

Payload range (Kg):

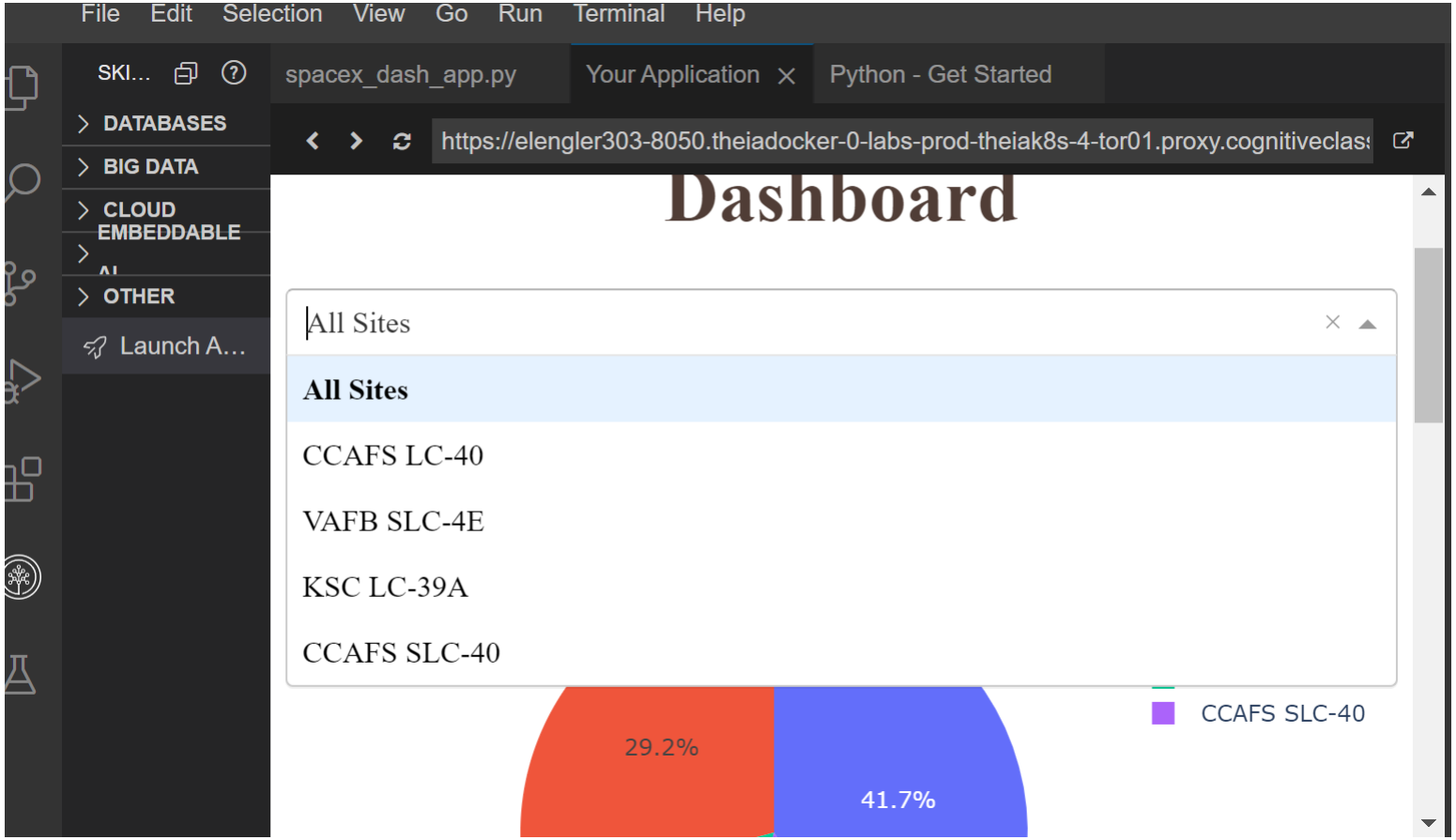
0 2500 5000 7500 10000

Correlation between Payload and Success for all Sites

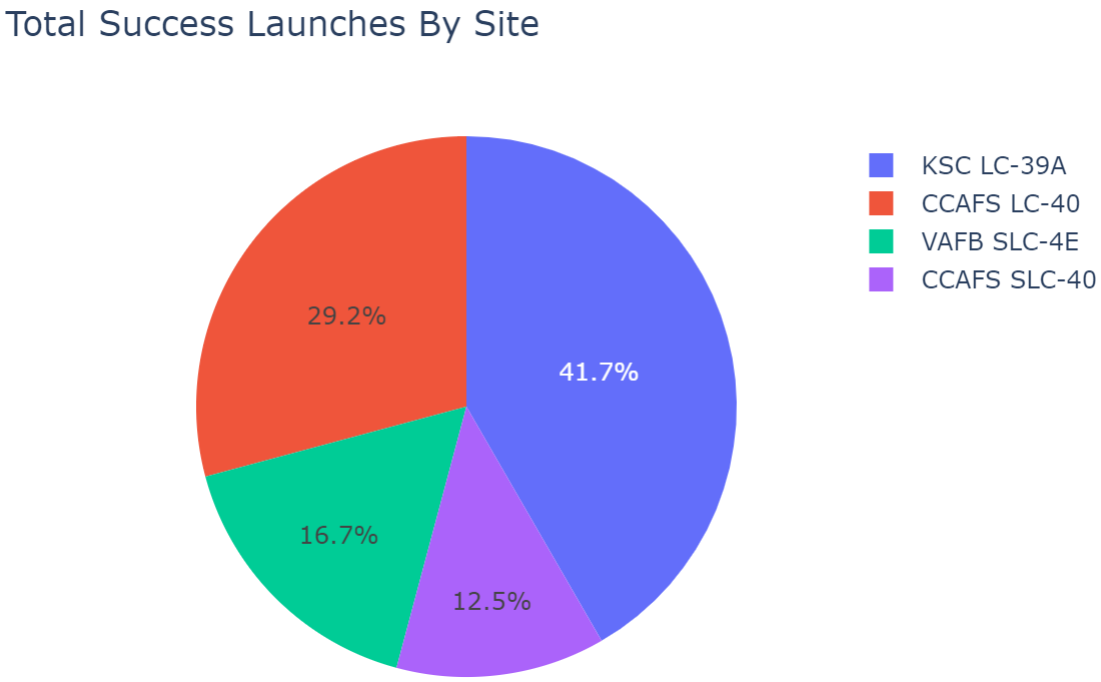


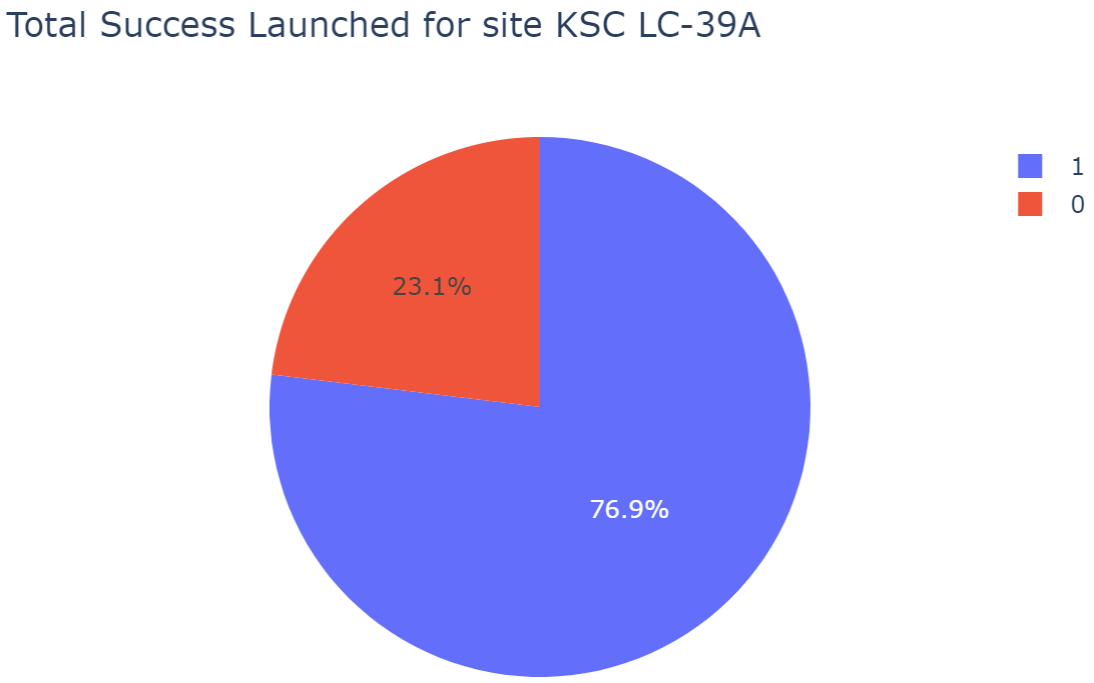
Build a Dashboard with Plotly Dash

A dashboard with a dropdown menu for selecting launch sites was made using Plotly Dash allowing users to filter data by launch site.



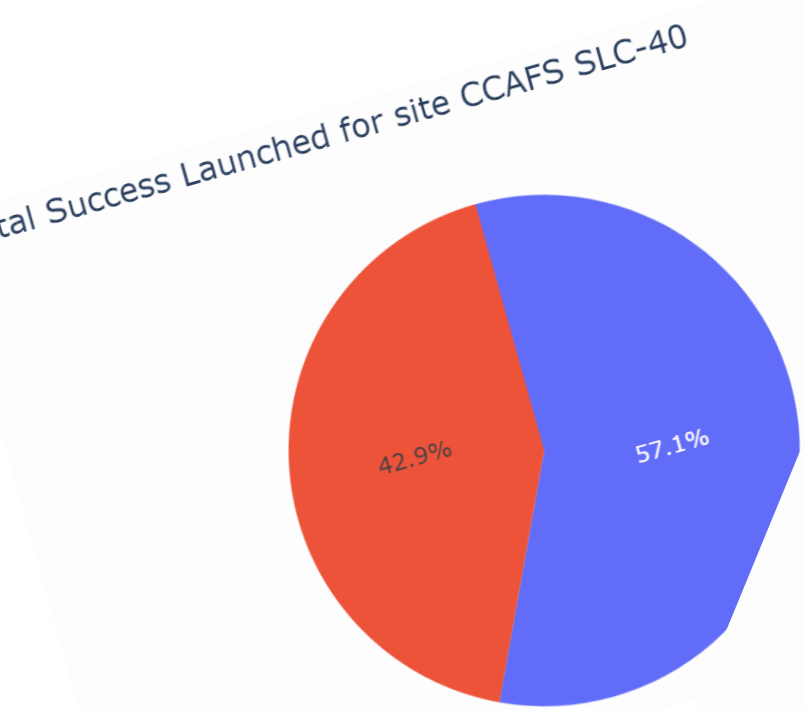
Build a Dashboard with Plotly Dash

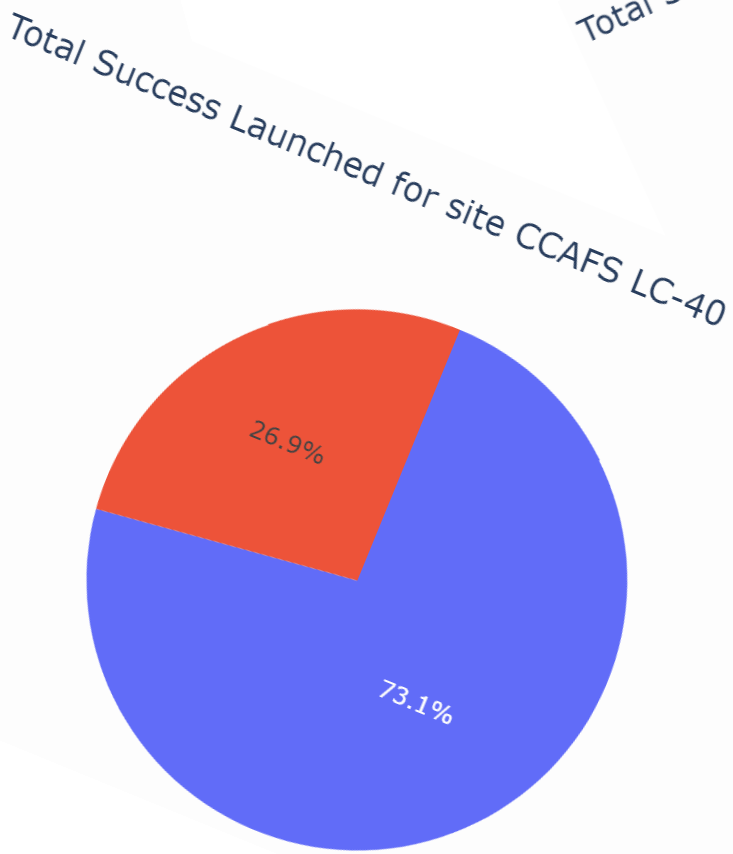


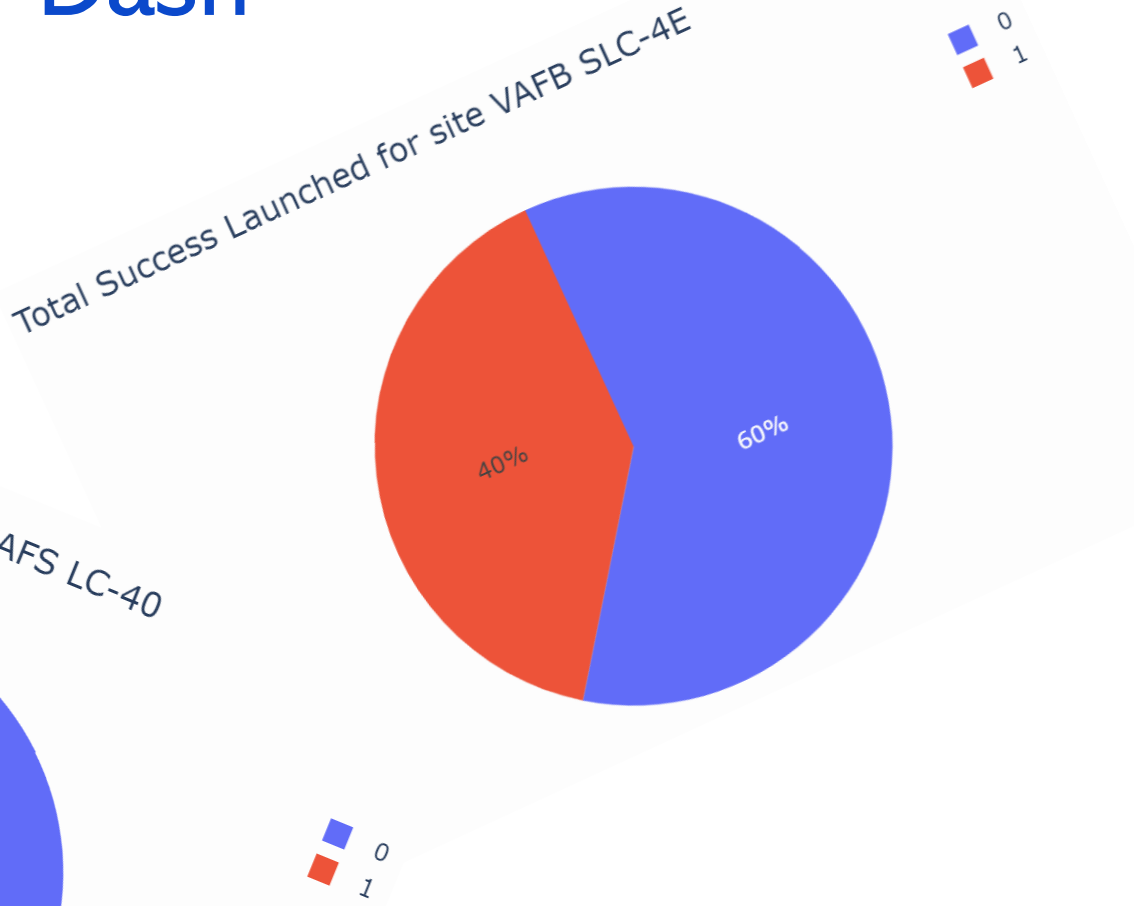


Pie charts showing percent of successful and failed launches are called by the dropdown menu. Next slide also…

Build a Dashboard with Plotly Dash







***See slides 22-25 to review how launch sites differ in number and timing of launch attempts.*

Libraries Imported for Predictive Analysis

```
# Pandas is a software library written for the Python programming language for data manipulation and analysis.
import pandas as pd
# NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along
import numpy as np
# Matplotlib is a plotting library for python and pyplot gives us a MatLab like plotting framework. We will use this in our plott
import matplotlib.pyplot as plt
#Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive ar
import seaborn as sns
# Preprocessing allows us to standarsize our data
from sklearn import preprocessing
# Allows us to split our data into training and testing data
from sklearn.model_selection import train_test_split
# Allows us to test parameters of classification algorithms and find the best one
from sklearn.model_selection import GridSearchCV
# Logistic Regression classification algorithm
from sklearn.linear_model import LogisticRegression
# Support Vector Machine classification algorithm
from sklearn.svm import SVC
# Decision Tree classification algorithm
from sklearn.tree import DecisionTreeClassifier
# K Nearest Neighbors classification algorithm
from sklearn.neighbors import KNeighborsClassifier
```

Predictive Analysis: Confusion Matrix Plotting Function

```
def plot_confusion_matrix(y,y_predict):
    "this function plots the confusion matrix"
    from sklearn.metrics import confusion_matrix

    cm = confusion_matrix(y, y_predict)
    ax= plt.subplot()
    sns.heatmap(cm, annot=True, ax = ax); #annot=True to annotate cells
    ax.set_xlabel('Predicted labels')
    ax.set_ylabel('True labels')
    ax.set_title('Confusion Matrix');
    ax.xaxis.set_ticklabels(['did not land', 'land']); ax.yaxis.set_ticklabels(['did not land', 'landed'])
    plt.show()
```

Predictive Analysis: Data Source and Acquisition

```
from js import fetch
import io

URL1 = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_2.csv"
resp1 = await fetch(URL1)
text1 = io.BytesIO((await resp1.arrayBuffer()).to_py())
data = pd.read_csv(text1)
```

```
data.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	Reus
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	

```
URL2 = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_3.csv'
resp2 = await fetch(URL2)
text2 = io.BytesIO((await resp2.arrayBuffer()).to_py())
X = pd.read_csv(text2)
```

```
X.head(100)
```

	FlightNumber	PayloadMass	Flights	Block	ReusedCount	Orbit_ES-L1	Orbit_GEO	Orbit_GTO	Orbit_HEO	Orbit_ISS	...	Serial_B1058	Serial
0	1.0	6104.959412	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
1	2.0	525.000000	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	

Thank you!