IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Name>
<Date>

Outline

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

Executive Summary

- Summary of methodologies

  - Data Collection
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis using SQL
  - EDA DataViz Using Python Pandas and Matplotlib
  - Launch Sites Analysis with Folium-Interactive Visual Analytics and PlotyDash
  - Machine Learning Landing Prediction

- Summary of all results
  - EDA Results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis (Classification)

Introduction

- Project background and context

  SpaceX 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 SpaceX can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

- •Problems you want to find answers

  In this capstone, we will predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches and various methods to determine its success rate.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Description of how SpaceX Falcon9 data was collected.

  - First the data is collected using SpaceX API (a RESTful API) by making a get request to the SpaceX API. This was done by first defining a series helper functions that would help in the use of the API to extract information using identification numbers in the launch data and then requesting rocket launch data from the SpaceX API url: *https://api.spacexdata.com/v4/launches/past*.

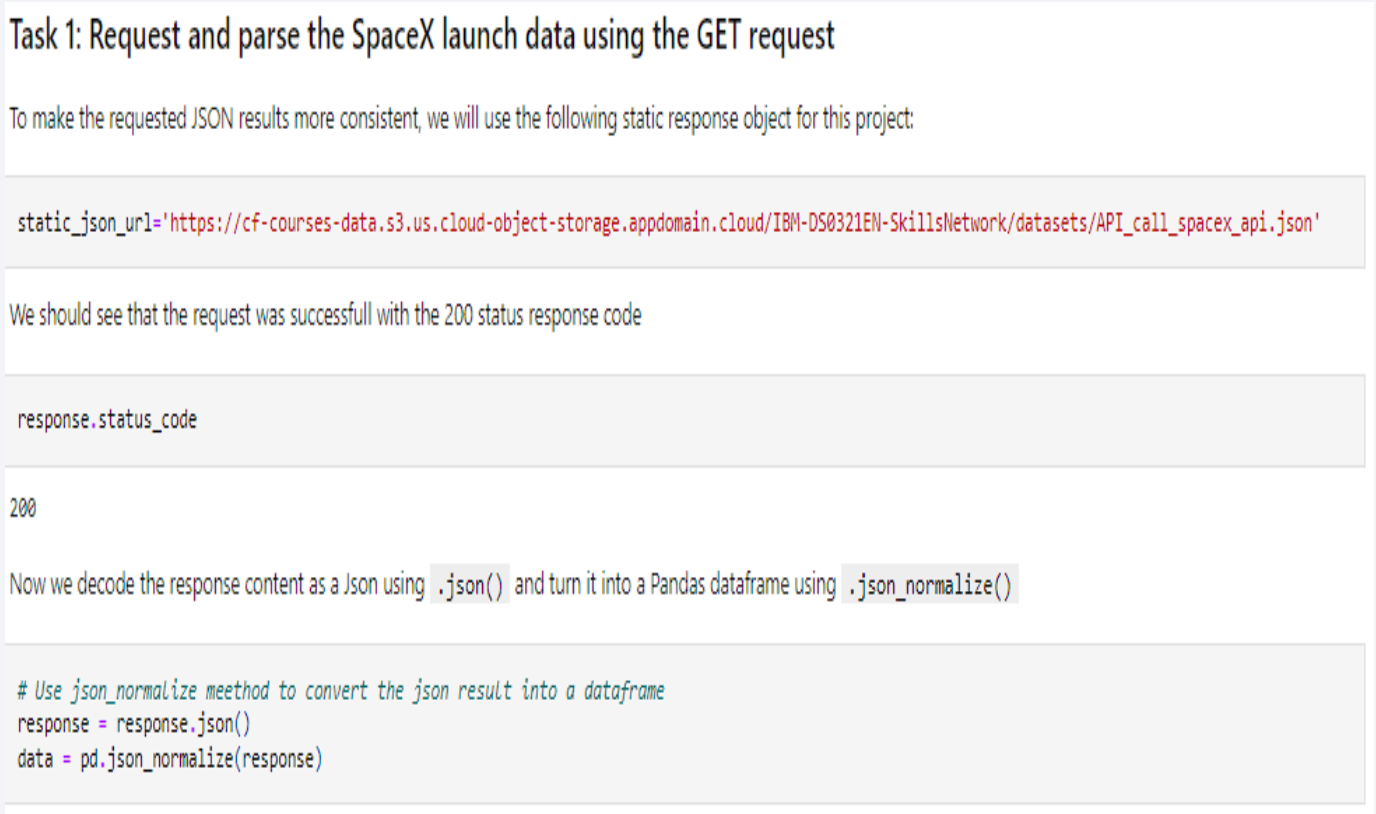
  - Then a to make the JSON results more consistent, the SpaceX launch data was requested and parsed using the GET request and then decoded the response content as a Json result which after being normalized it was then converted into a Pandas data frame.

  - Also performed web scraping to collect Falcon 9 historical launch records from the url: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922 which contains the launch records and are stored in a HTML. Using BeautifulSoup the information was extracted from the Falcon 9 launch HTML. After, the information extracted was parsed into a dictionary created with the columns names and then it was finally converted into a Pandas data frame.

Data Collection – SpaceX API

- Data collected using SpaceX API (a RESTful API) by making a get request to the SpaceX API then requested and parsed the SpaceX launch data using the GET request and decoded the response content as a Jsonresult which was then converted into a Pandas data frame.

- Here is the GitHub URL of the completed Data Collection Notebook.
- URL: https://github.com/turquiz/IMB-Watson/blob/f439051c06d87c0e86583cf0c2594f19cf5c35cc/Final_Assignment.ipynb



Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

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

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

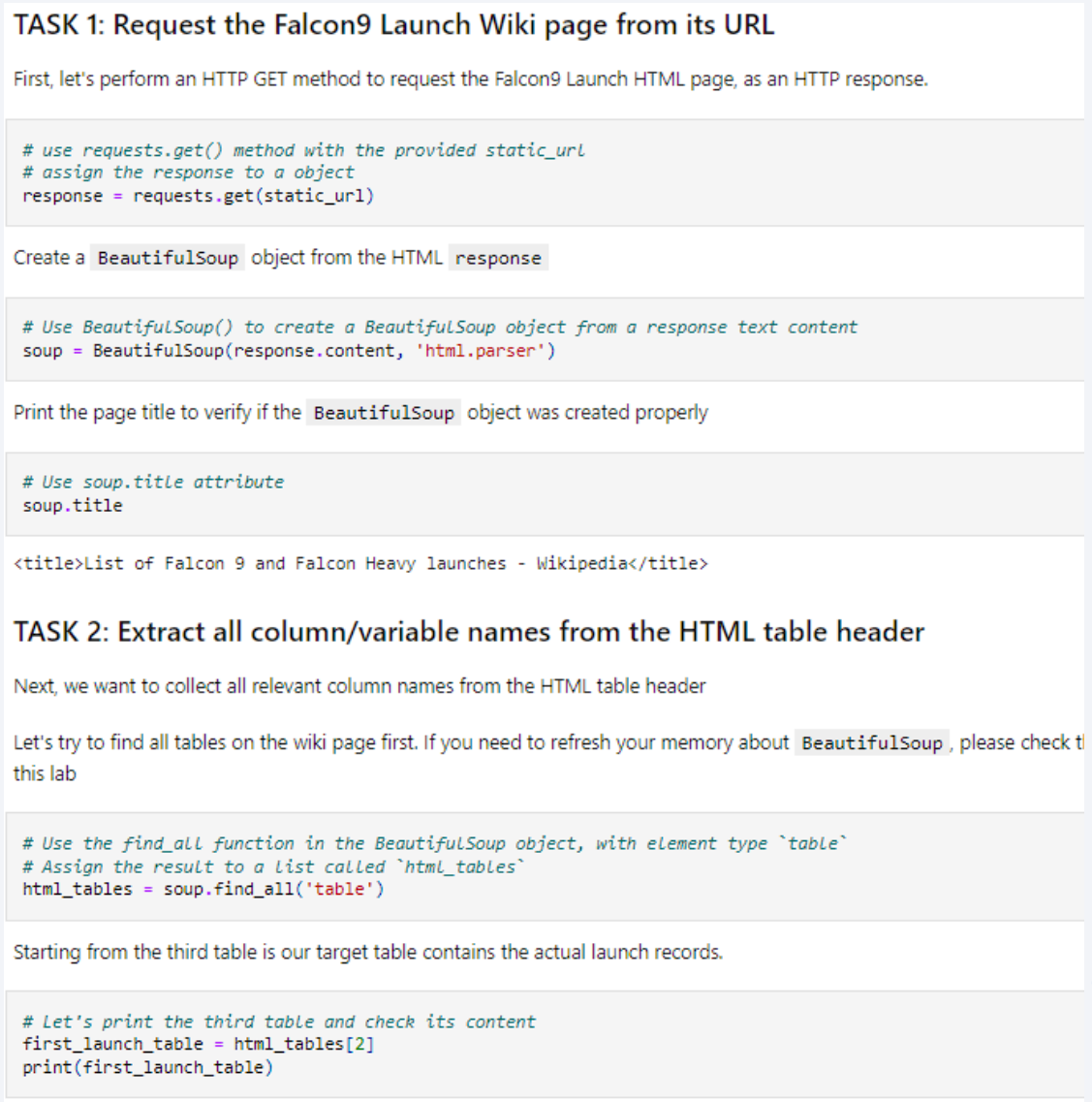
Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
# Use json_normalize meethod to convert the json result into a dataframe
response = response.json()
data = pd.json_normalize(response)
```

Data Collection - Scraping

- To perform the web scrapping, the launch records of Falcon 9 were taken from its Wikipedia page using BeautifulSoup and request in order to extract them from its HTML table, then it was parsed to a dictionary in order to create a Data frame with pandas from it.

- Here is the GitHub URL of the completed Data Scraping Notebook.
- URL: https://github.com/turquiz/IMB-Watson/blob/f439051c06d87c0e86583cf0c2594f19cf5c35cc/Final_Assigment_2.ipynb



TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
# 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`

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

Print the page title to verify if the `BeautifulSoup` object was created properly

```
# Use soup.title attribute
soup.title
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check t
this lab

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

Starting from the third table is our target table contains the actual launch records.

```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

Data Wrangling

•After the process of creating the Pandas data set from the previous points, we determined which where the possible OUTCOME and its occurrence. After that, we stablished which where the bad OUTCOME and created a new class where presented the information in binary to simplify it.

• Here is the GitHub URL of the completed Data Wrangling Notebook.

URL: https://github.com/turquiz/IMB-Watson/blob/145225792786b132a2addda84e35229b317bf5b2/Final_Assigment_3.ipynb

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

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

We create a set of outcomes where the second stage did not land successfully:

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

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
df['Class'] = df['Outcome'].apply(lambda x: 0 if x in bad_outcomes else 1)
df['Class'].value_counts()
```

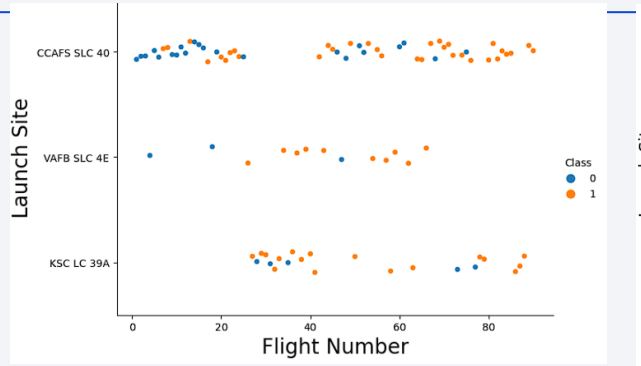
```
1    60
0    30
Name: Class, dtype: int64
```

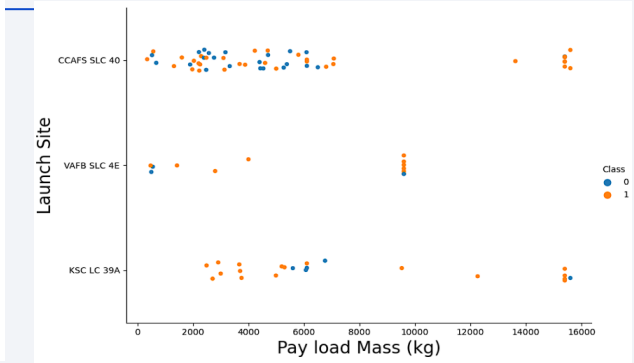
This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

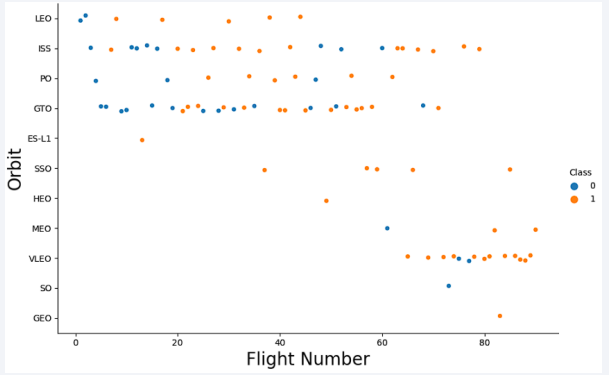
```
landing_class =df['Class']
df[['Class']].head(8)
```

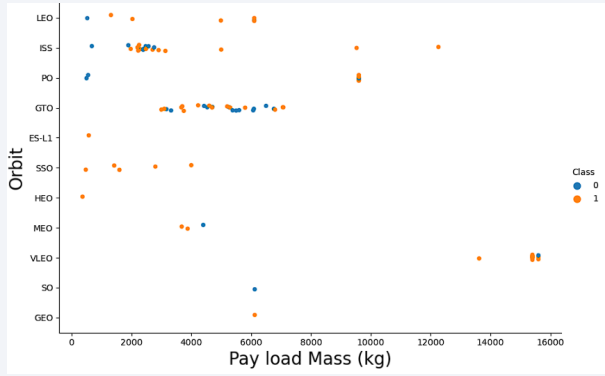
EDA with Data Visualization

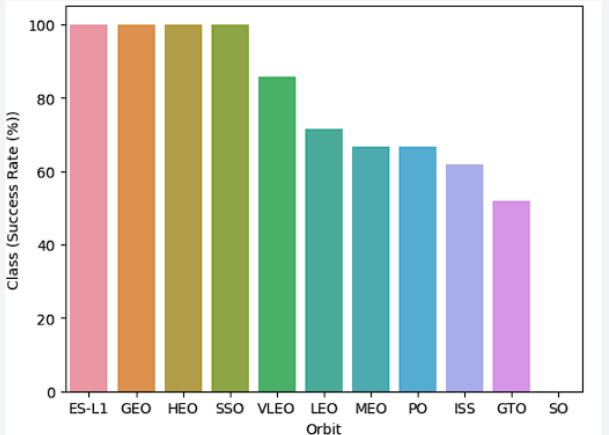
- Performed data Analysis and Feature Engineering using Pandas and Matplotlib
- Used scatter plots to Visualize the relationship between Flight Number and Launch Site, Payload and Launch Site, Flight Number and Orbit type, Payload and Orbit type.
- Used Bar chart to Visualize the relationship between success rate of each orbit type
- Line plot to Visualize the launch success yearly trend.
- Here is the GitHub URL of the completed Data Visualization Notebook.
- URL: https://github.com/turquiz/IMB-Watson/blob/17acd0c969b51d48749df4960dbd5fe19463b1d0/Final_Assigment_5.ipynb

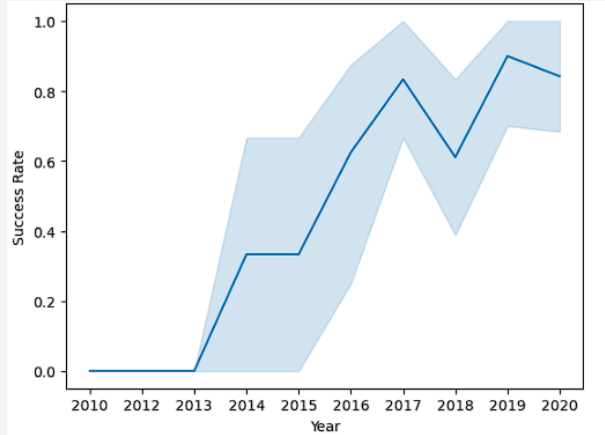












EDA with SQL

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

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

- Here is the GitHub URL of the completed SQL Notebook.
- URL: https://github.com/turquiz/IMB-Watson/blob/17acd0c969b51d48749df4960dbd5fe19463b1d0/Final_Assigment_4.ipynb

Build an Interactive Map with Folium

- Created folium map to marked all the launch sites, and created map objects such as markers, circles, lines to mark the success or failure of launches for each launch site.
- Created a launch set outcomes (failure=0 or success=1) to facilitate the understanding of information.

- Here is the GitHub URL of the completed Data Wrangling Notebook.
- URL: https://github.com/turquiz/IMB-Watson/blob/17acd0c969b51d48749df4960dbd5fe19463b1d0/Final_Assigment_6.ipynb

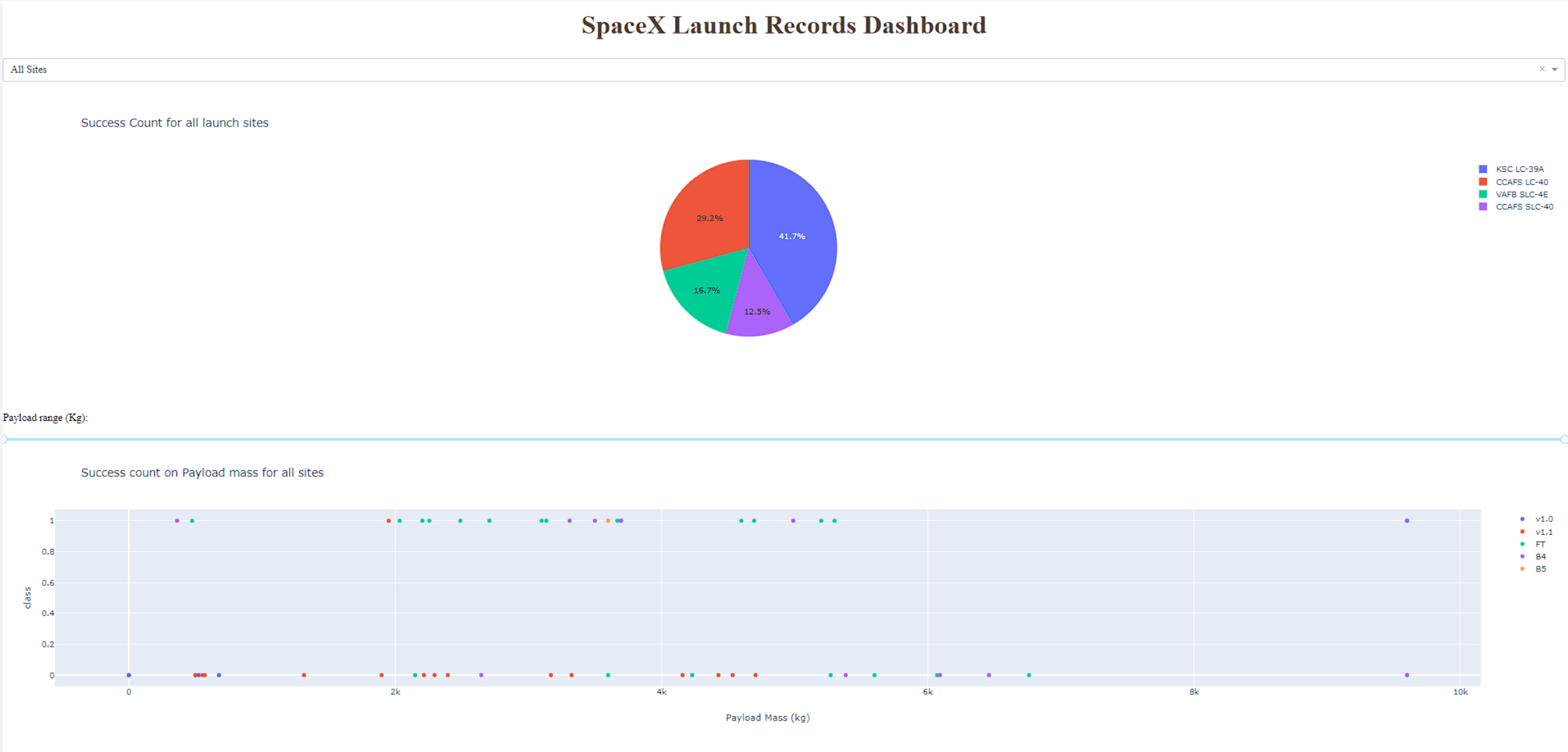
Build a Dashboard with Plotly Dash

- Built an interactive dashboard application with Plotlydash by:
  - Adding a Launch Site Drop-down Input Component
  - Adding a callback function to render success-pie-chart based on selected site dropdown
  - Adding a Range Slider to Select Payload
  - Adding a callback function to render the success-payload-scatter-chart scatter plot

- Here is the GitHub URL of the completed Data Wrangling Notebook.
- URL: https://github.com/turquiz/IMB-Watson/blob/17acd0c969b51d48749df4960dbd5fe19463b1d0/Final_Assigment_7.ipynb

- Dashboard: http://127.0.0.1:8050/

SpaceX Dash App



Predictive Analysis (Classification)

- After the process or collecting and loading the data into a pandas Dataframe, an exploratory data analysis was performed to determine where the parameters of testing and training would be used.

- A **NumPy** array was created from the column **Class** in data, by applying the method **to_numpy()** then assigned it to the variable **Y** as the outcome variable.

- Then standardized the feature dataset (x) by transforming it using **preprocessing.StandardScaler()** function from **Sklearn**.

- The data was split into training and testing sets using the function **train_test_split** from **sklearn.model_selection** with the **test_size** parameter set to 0.2 and **random_state** to 2.

Predictive Analysis (Classification)

- After that we tried different Machine Learning models with the testing and training data to determine with would perform the best. The ML models used where: SVM, Classification Trees, k nearest neighbors and Logistic Regression.
- In order to determine with was the best fitted for prediction, we determine the accuracy of the tested Data and found out that the results where the same for the 4 different ML methods giving a valor of 83.3%

Out[35]:

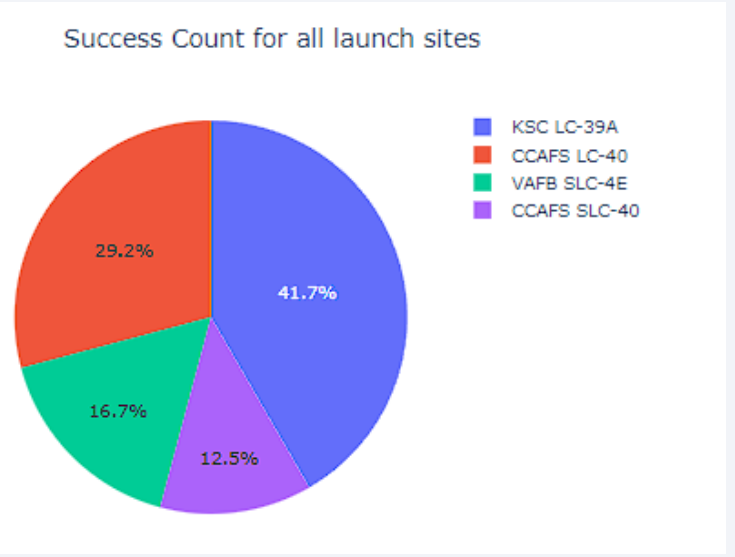
	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

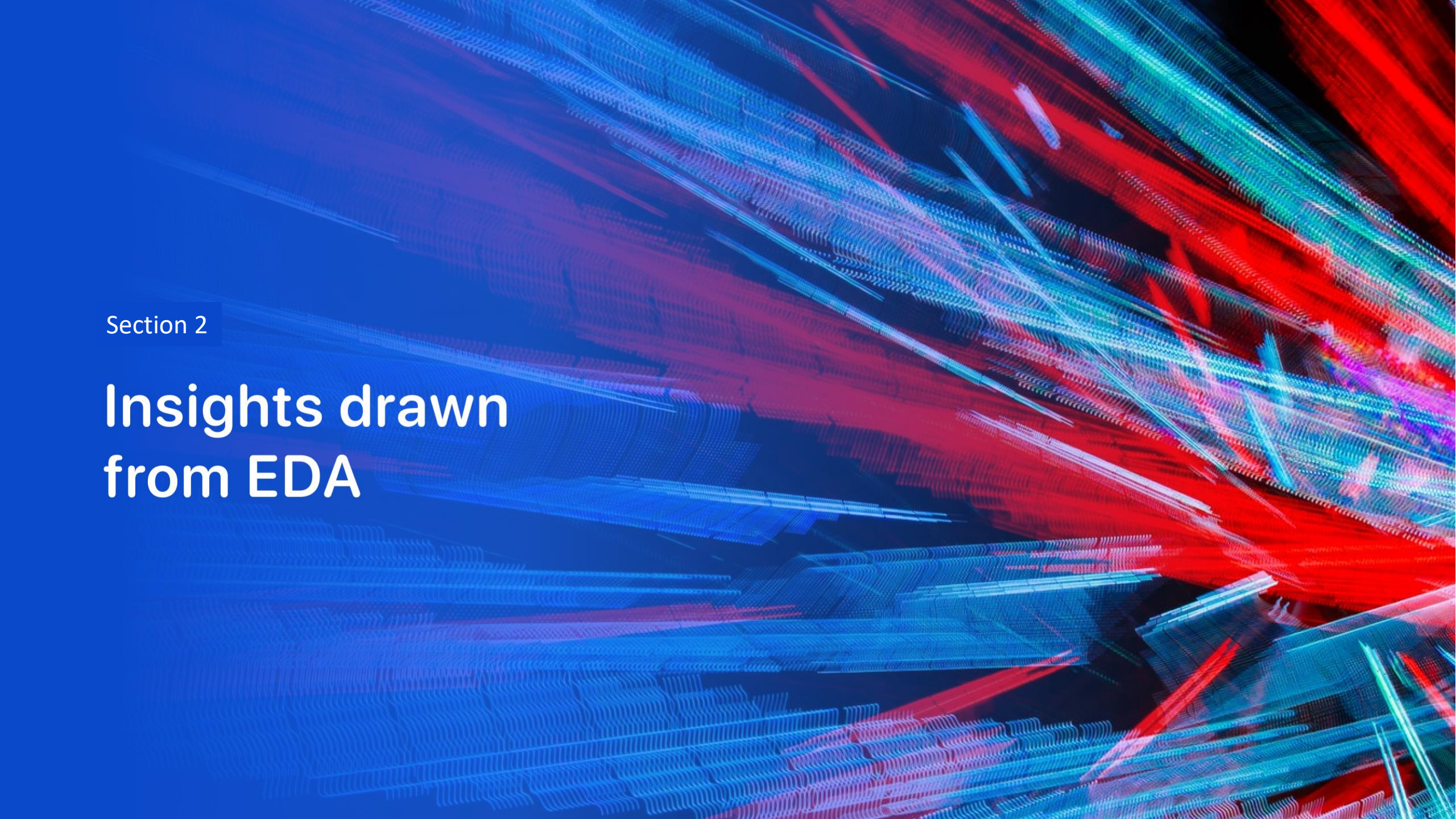
Results

- After applying a exploratory analysis to the Falcon 9 launch database. We concluded that any of the Machine Learning models could be used to predict the possible outcome of the launches. Although the expected accuracy of the data was a high number of 83%, taking into consideration the price to pay for a failed attempt it is recommended we do more studying on possible ways to fix this problem.
- Taking into consideration the Launch Sites. We can conclude that the best launch site to use is the KSC LC-39A located in the Easter Coast on Florida. The results outclass the other launch sites.

Out[35]:

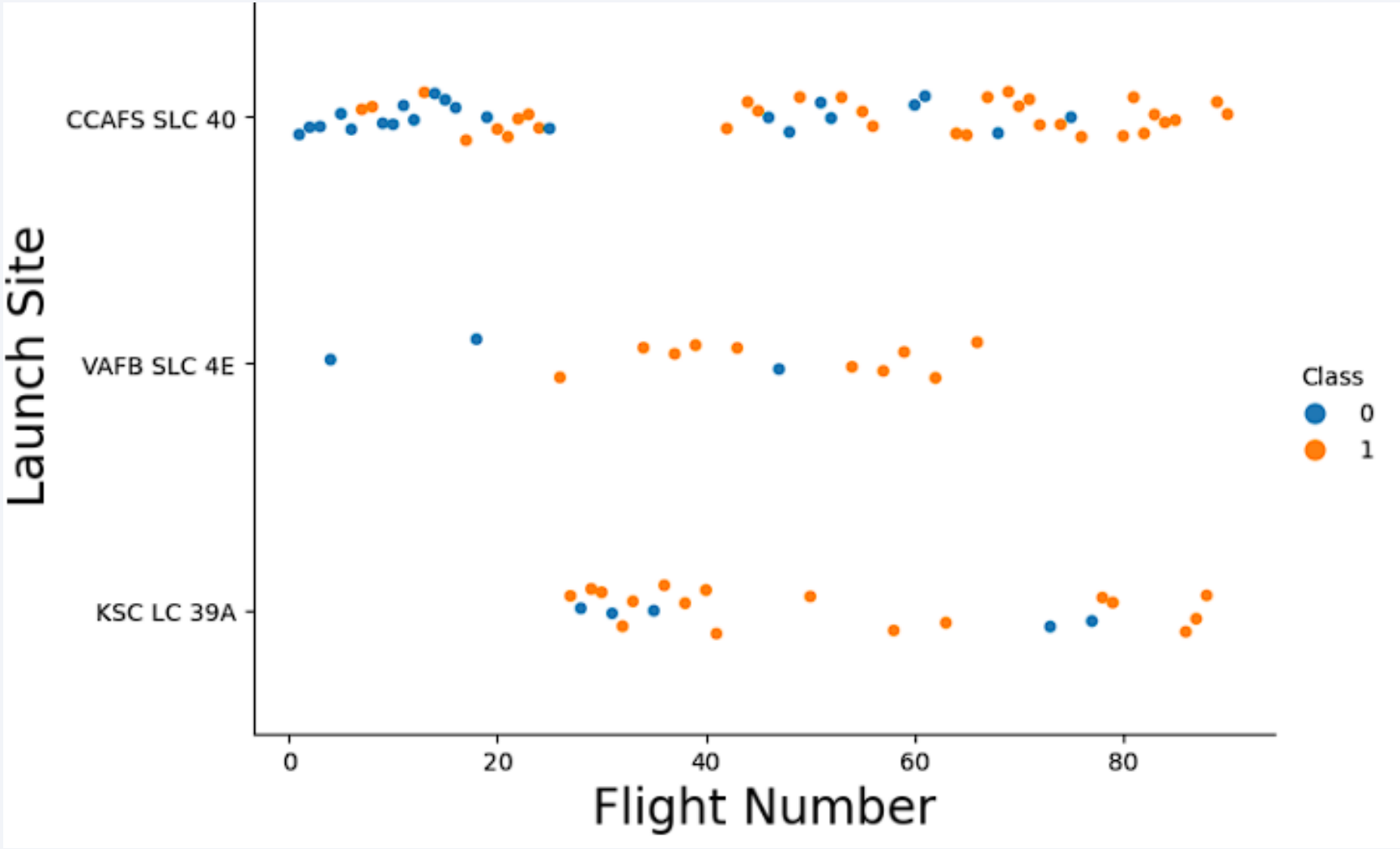
	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333




Section 2
Insights drawn from EDA

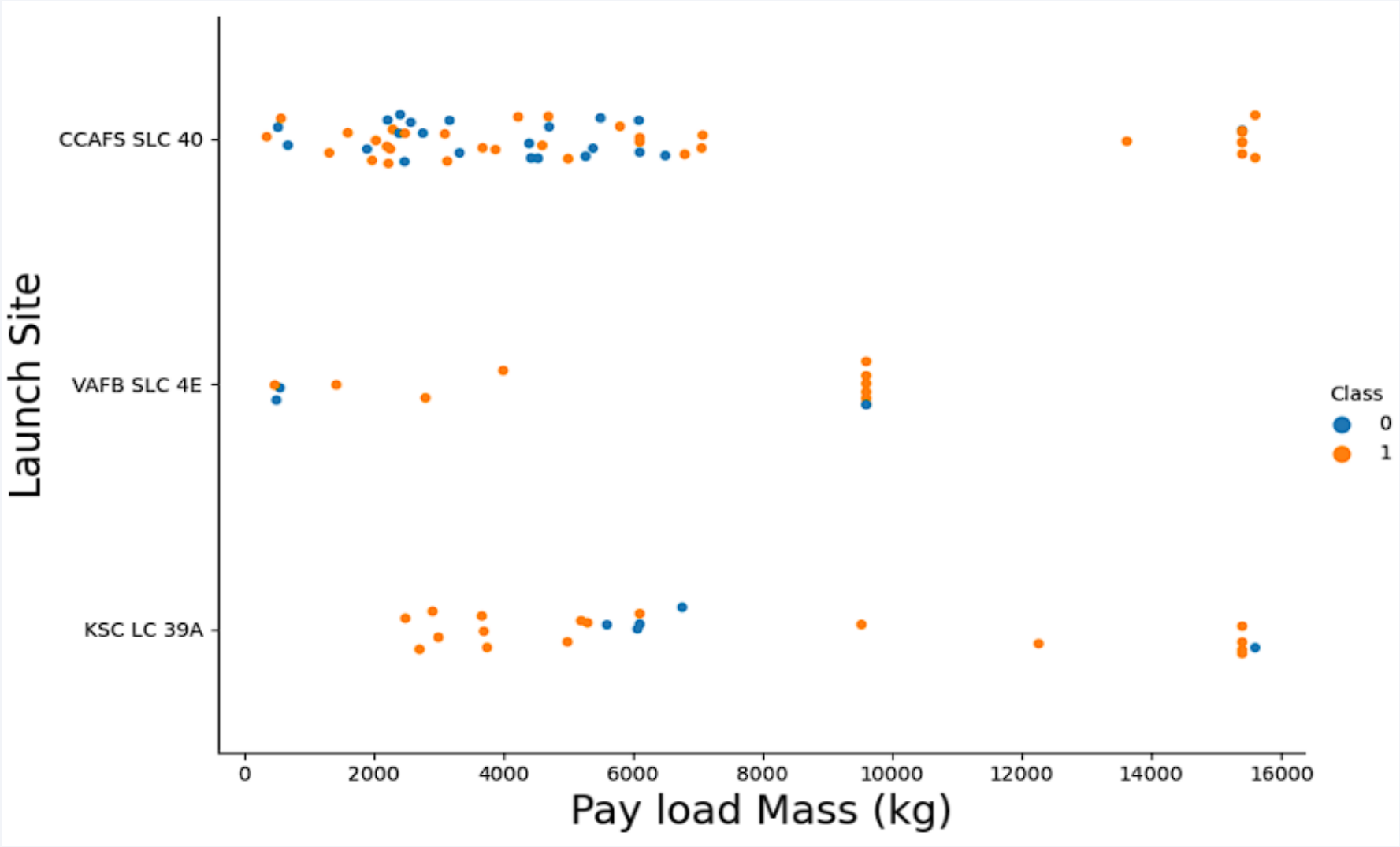
Flight Number vs. Launch Site

- We can find that there is a correlation between number of launches and success rate on each of the sites. On the site VAFB, after they passed the 50th launch their success rate went to 100%. For the other 2 sites you can say the same after their launch 80.



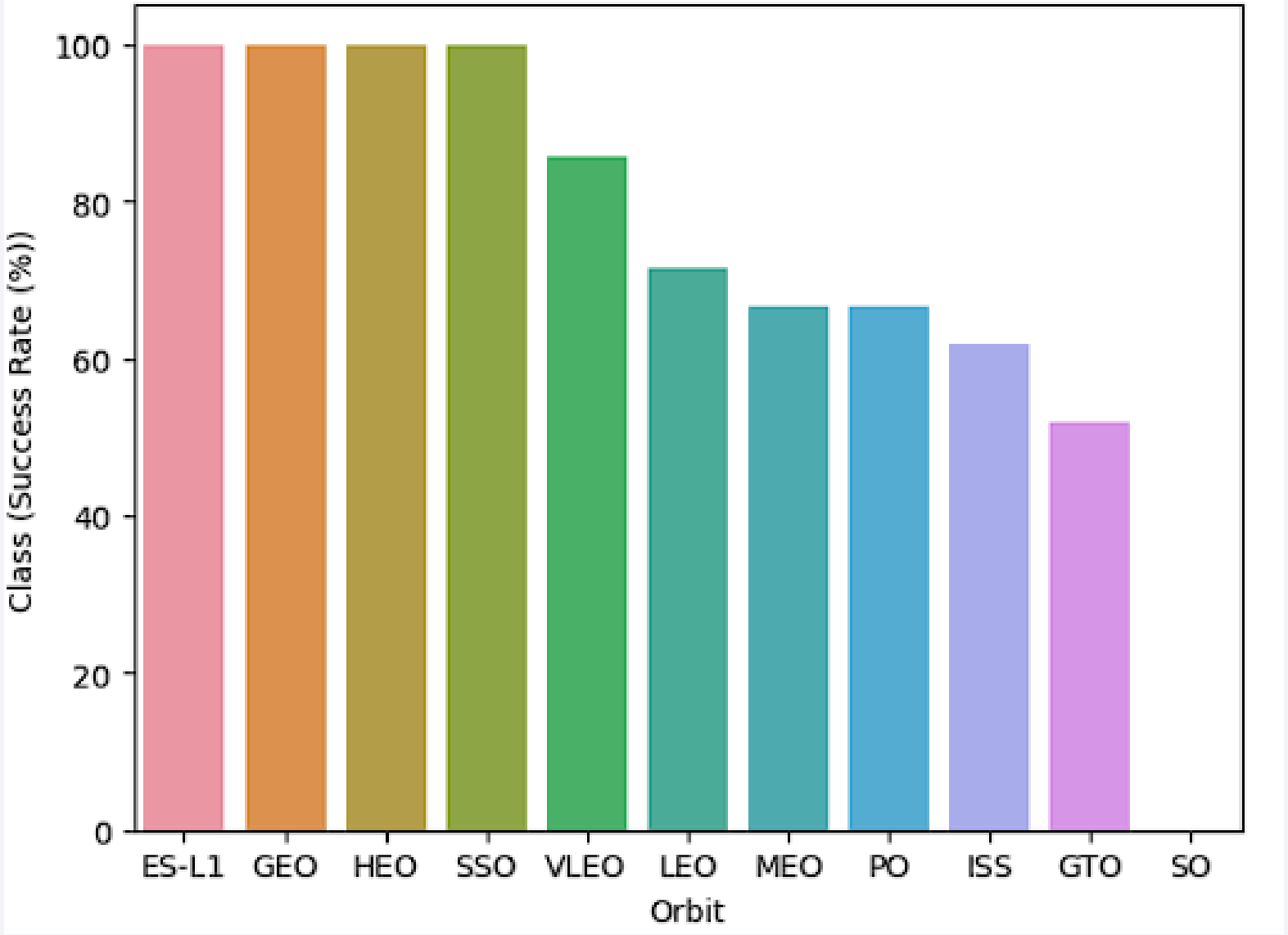
Payload vs. Launch Site

- Taking into consideration the information presented by the chart, we can conclude that the Launch site VAFB can not hold launches where their rockets surpass 10,000kg in weight.



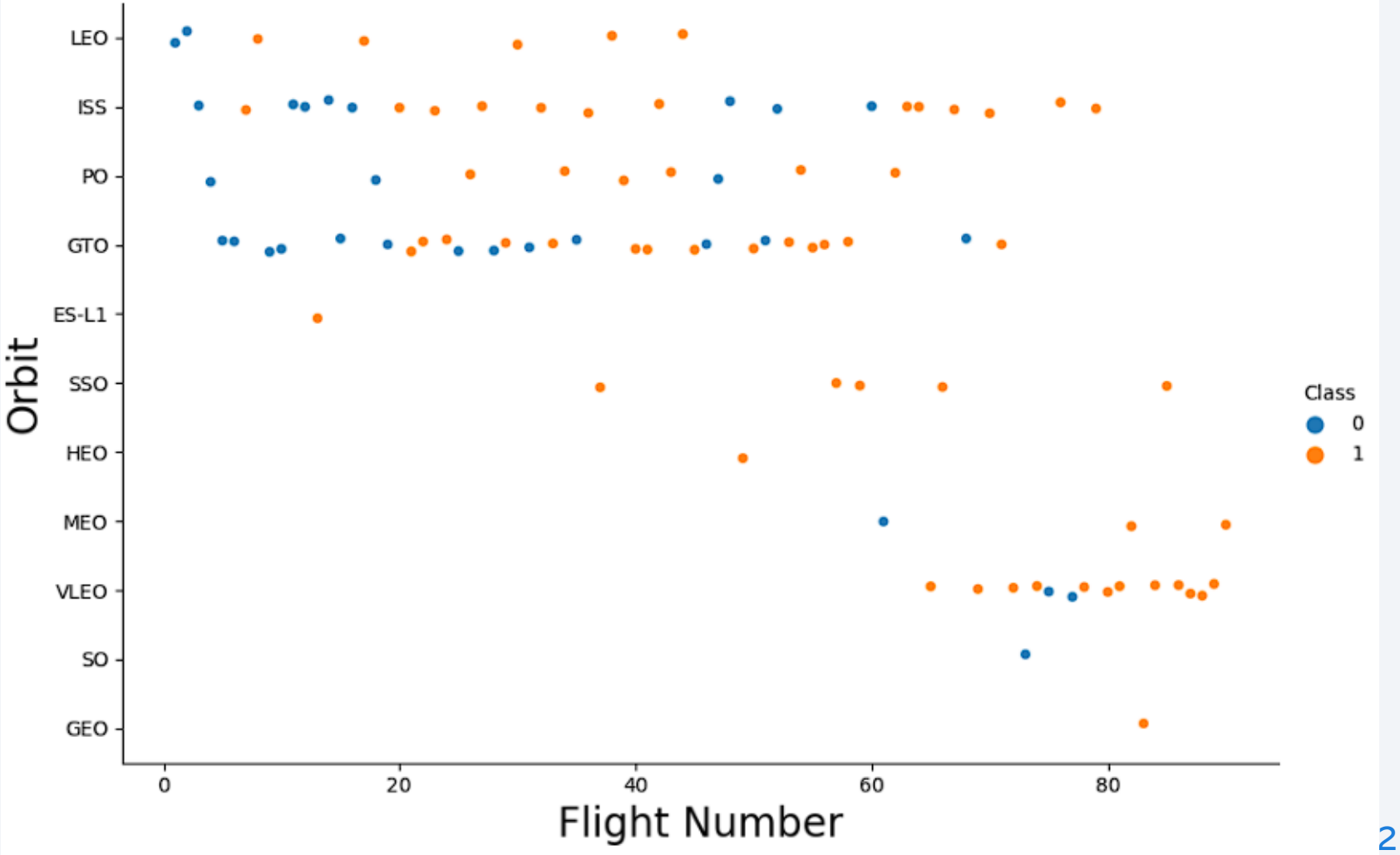
Success Rate vs. Orbit Type

- We can conclude that from orbits ES-L1 to SSO, their success rate has always being 100%.
- SO have not ever had a successful launch.



Flight Number vs. Orbit Type

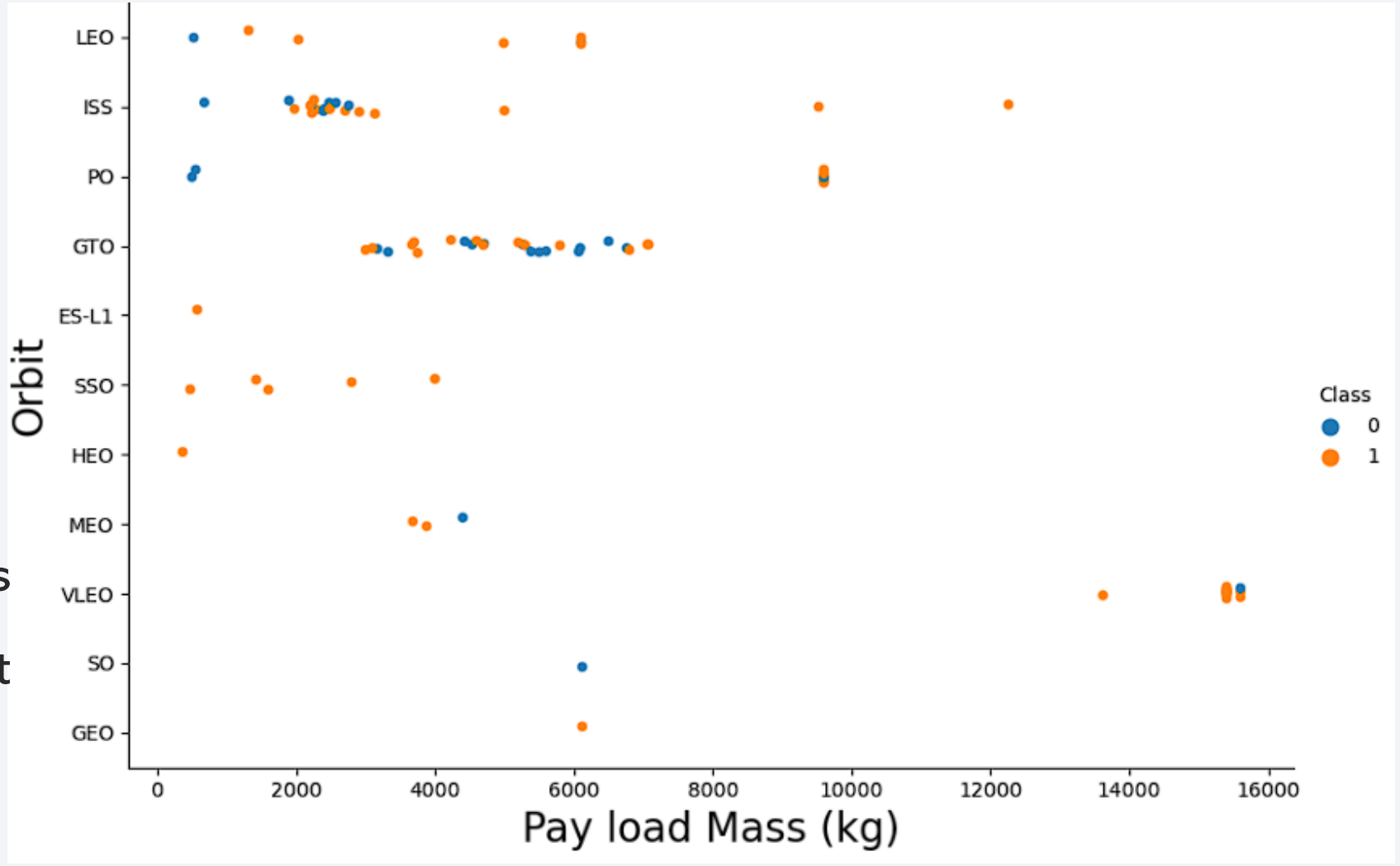
- We can conclude from this graph that LEO orbit have had an increase in success rate after 10 launches where we could not find any correlation on GTO orbit due to it having a random assessment of success.
- SO also increased its % of success after approximately 80 launches.



Payload vs. Orbit Type

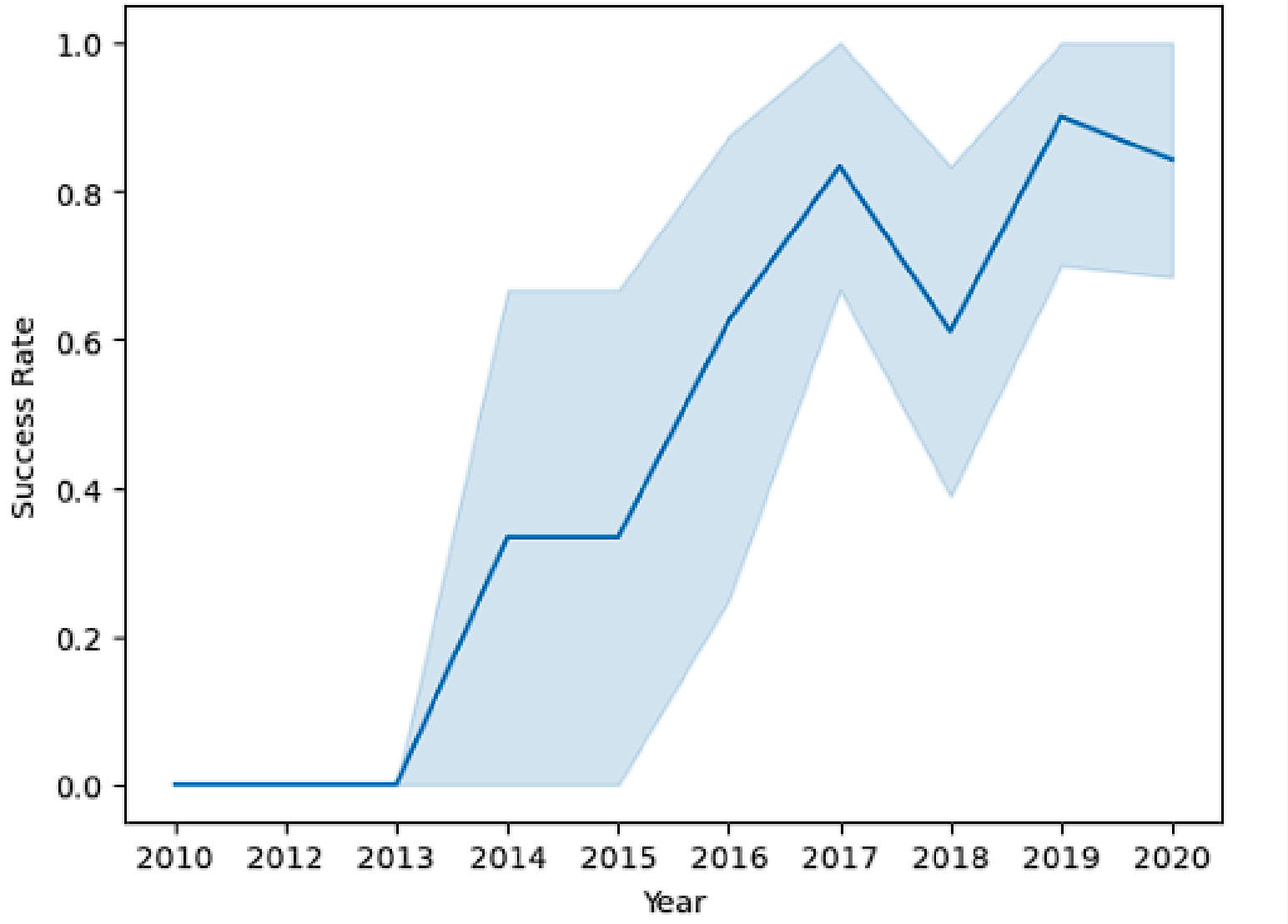
- We can conclude that for LEO, ISS and SSO, the heaver the payload, the more success they have achived on their launches.

- GTO have had a mix of both success and fails which determines that its payload mass can not determine if it will be a successful launch.



Launch Success Yearly Trend

- We can se a correlation with success rating and time passed. It have being on a positive way except for some problems on 2018.



All Launch Site Names

- Here we can select the distinct values from the spacex database and determined there were 4 sites for Falcon 9.

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

In [15]:
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

Out[15]:

Launch_Sites
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 [16]:
```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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

Out[16]:

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

- We used the ‘LIKE’ command with ‘%’ wildcard in ‘WHERE’ clause to select and display a table of all records where launch sites begin with the string 'CCA'

Total Payload Mass

Task 3

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

In [17]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

Out[17]:

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

- We used the 'SUM()' function to return the display of the total sum of **PAYLOAD_MASS_KG** for **NASA** launches

Average Payload Mass by F9 v1.1

Task 4

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

In [18]:
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

Out[18]:

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

- We used the 'AVG()' function to return the display of the average **Payload_Mass_Kgs** carried by the booster version F9 v1.1

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

In [19]:
```
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)"
```

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

Out[19]:

MIN(DATE)
01-05-2017

- We used the ‘MIN()’ function to return the display of first date when successful landing outcome on ground happen (ground pad).

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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

In [22]:
```
%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOA
```

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

Out[22]:

Booster_Version	Payload
F9 FT B1022	JCSAT-14
F9 FT B1026	JCSAT-16
F9 FT B1021.2	SES-10
F9 FT B1031.2	SES-11 / EchoStar 105

- We used the ‘Select Distinct()’ statement to return the list of distinct names of boosters with operators between 4000 and 6000 and having them with a outcome of success.

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

In [23]:
```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

Out[23]:

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

- We used the ‘COUNT()’ function and ‘GROUP BY’ to return the display of the total number of missions outcomes.

Boosters Carried Maximum Payload

Task 8

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

In [24]:
```
%sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL
```

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

Out[24]:

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

- We used a subquery to return the booster version of all launches that had the maximum payload of 15,600kgs

2015 Launch Records

Task 9

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

In [25]:
```
%sql SELECT substr(Date,7,4), substr(Date, 4, 2),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing _Outcome"
```

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

Out[25]:

substr(Date,7,4)	substr(Date, 4, 2)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	Landing _Outcome
2015	01	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	Success	Failure (drone ship)
2015	04	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	Success	Failure (drone ship)

- We used the ‘SUBSTR()’ in the select statement in other to get the month and year of the failure drone ship outcome on mission success from the year 2015.

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

Task 10

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

In [26]:
```
%sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDER BY Date DESC;
```

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

Out[26]:

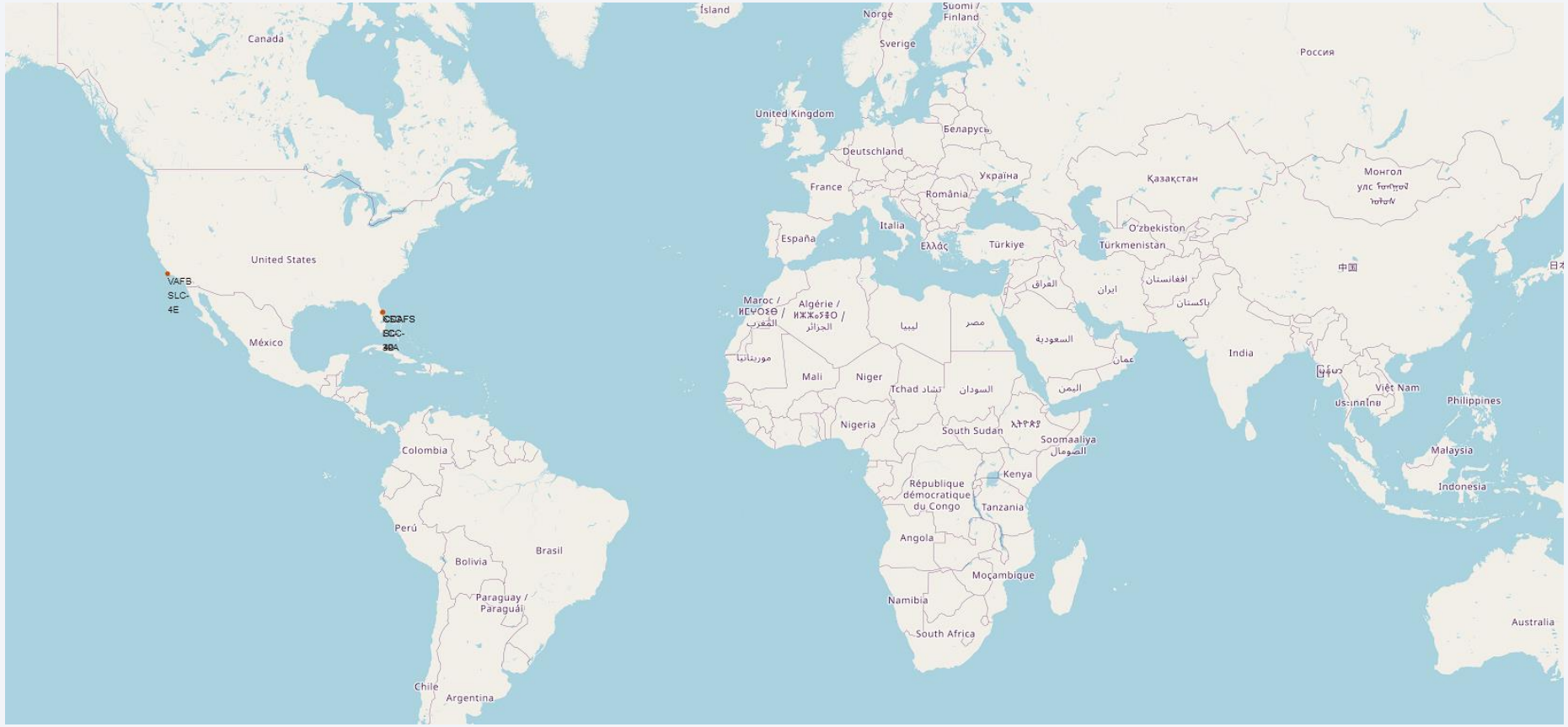
Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
19-02-2017	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-10-2020	12:25:57	F9 B5 B1051.6	KSC LC-39A	Starlink 13 v1.0, Starlink 14 v1.0	15600	LEO	SpaceX	Success	Success
18-08-2020	14:31:00	F9 B5 B1049.6	CCAFS SLC-40	Starlink 10 v1.0, SkySat-19, -20, -21, SAOCOM 18	15440	LEO	SpaceX, Planet Labs, PlanetIQ	Success	Success
18-07-2016	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-04-2018	22:51:00	F9 B4 B1045.1	CCAFS SLC-40	Transiting Exoplanet Survey Satellite (TESS)	362	HEO	NASA (LSP)	Success	Success (drone ship)
17-12-2019	00:10:00	F9 B5 B1056.3	CCAFS SLC-40	JCSat-18 / Kacific 1, Starlink 2 v1.0	6956	GTO	Sky Perfect JSAT, Kacific 1	Success	Success
16-11-2020	00:27:00	F9 B5B1061.1	KSC LC-39A	Crew-1, Sentinel-6 Michael Freilich	12500	LEO (ISS)	NASA (CCP)	Success	Success
15-12-[illegible]	15:36:00	F9 FT B1035.2	CCAFS SLC-[illegible]	SpaceX CRS-13	2205	LEO [illegible]	NASA (CRS)	Success	Success [illegible]

- In this table we can see the rank count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

Section 3

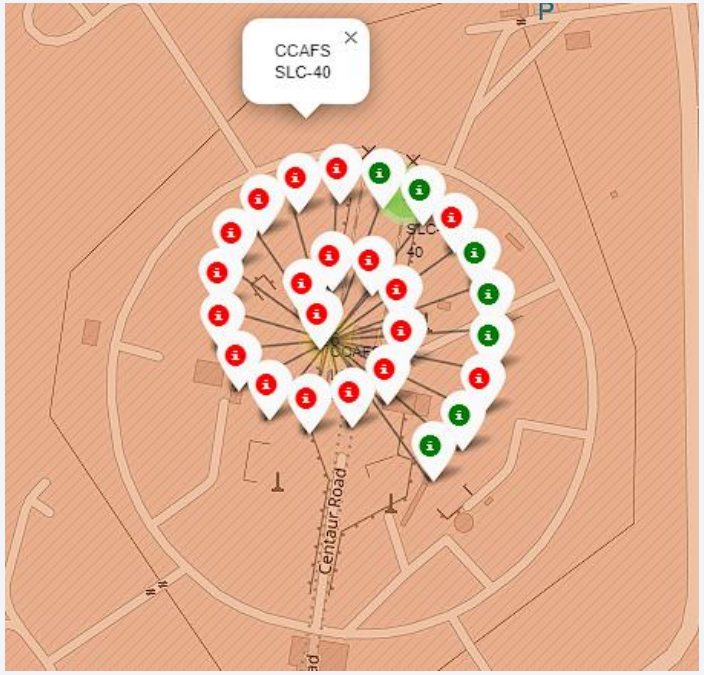
Launch Sites Proximities Analysis

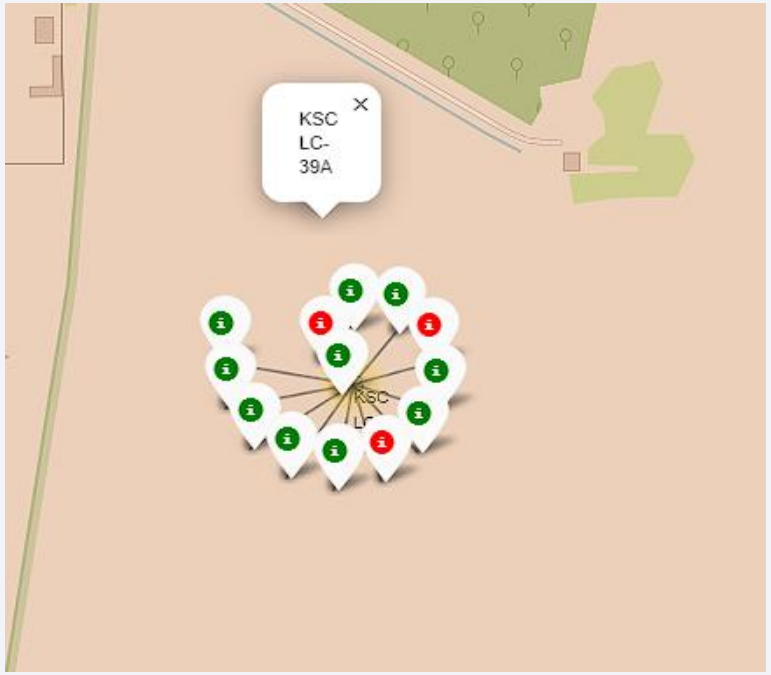
Markers of all launch sites on global map



- All launch sites are in proximity to the coast.

Launch outcomes for each site on the map With Color Markers

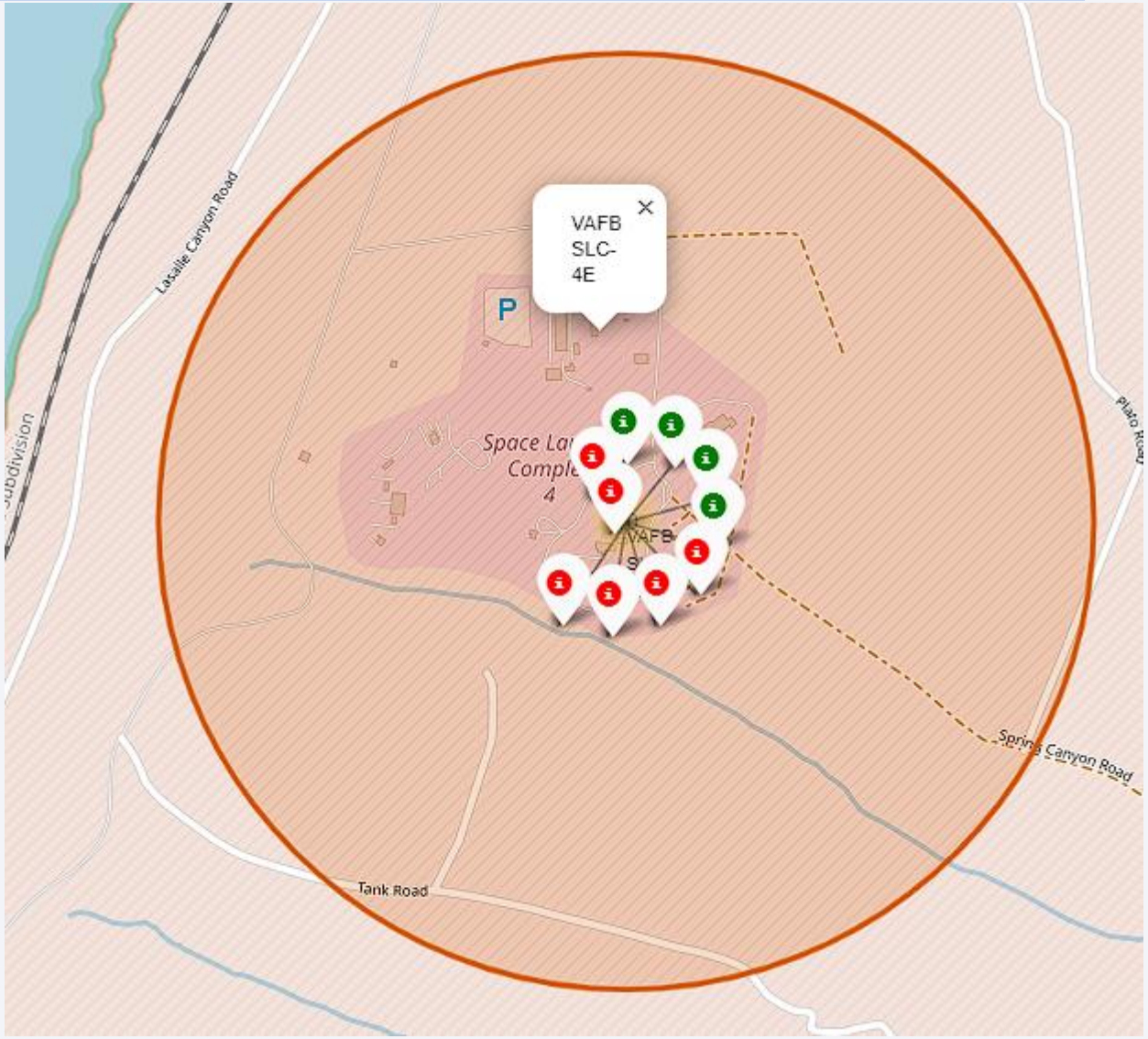




- In the Eastern coast of USA, the site KSC LC-39A as seen in the pictures, has a higher success rate compared to the the site at CCAFS SLC-40 & CCAFS LC-40

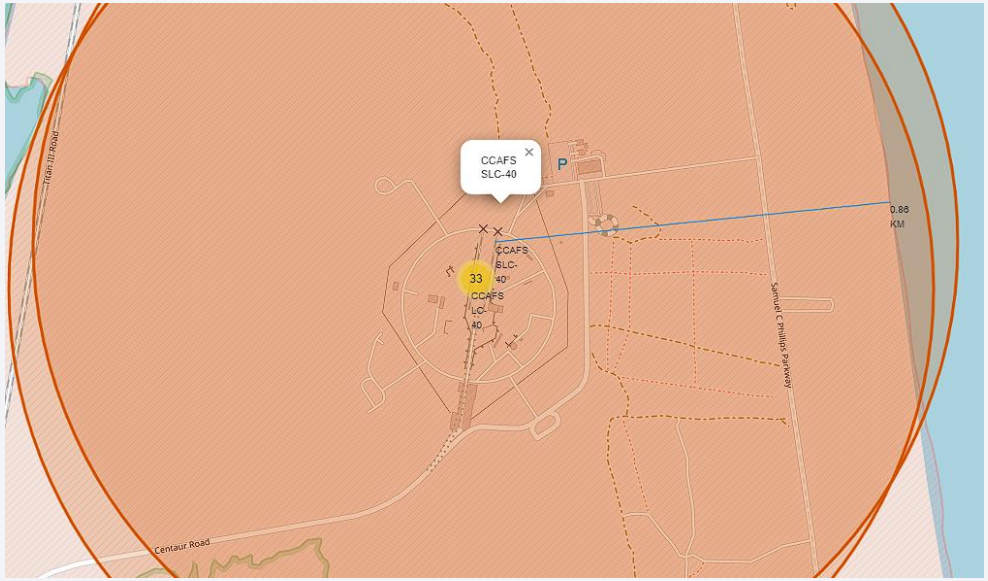
Launch outcomes for each site on the map With Color Markers

- In the Wester coast of USA, the launch site VAFB SLC-4E has relatively lower success rate, having 4 out of 10 successful launches, compared to KSC LC-39A launch site in the Eastern Coast.

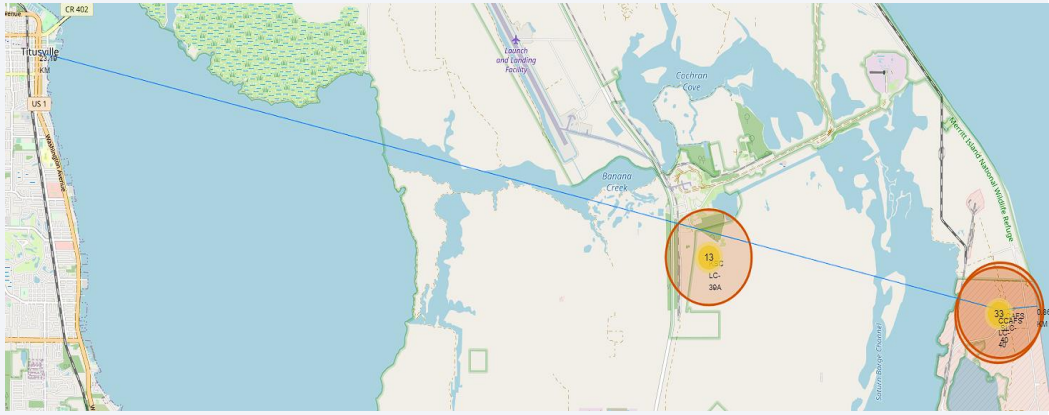


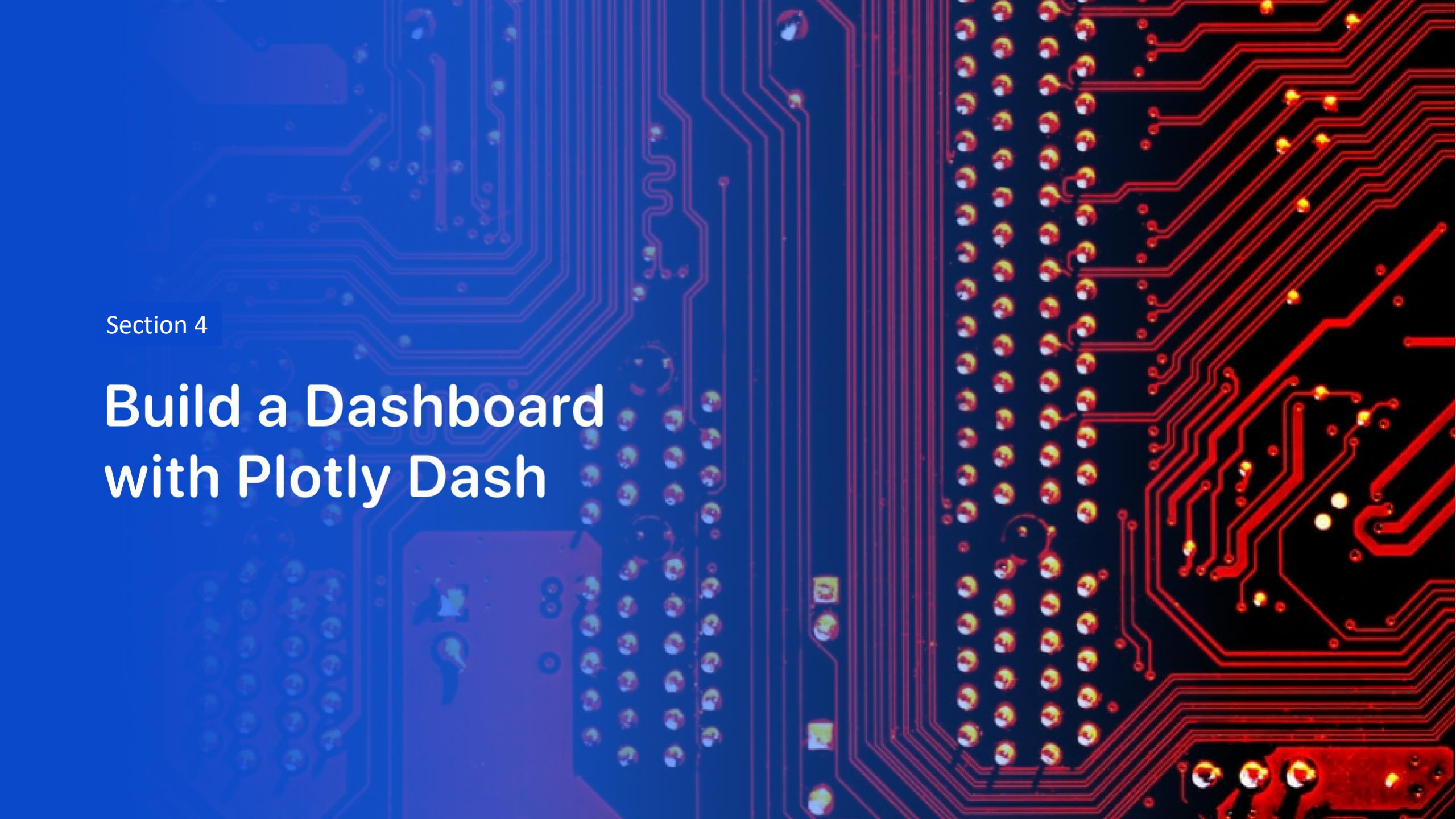
Distances between a launch site to its proximities

- Launch site CCAFS SLC-40proximity to coastline is 0.86km

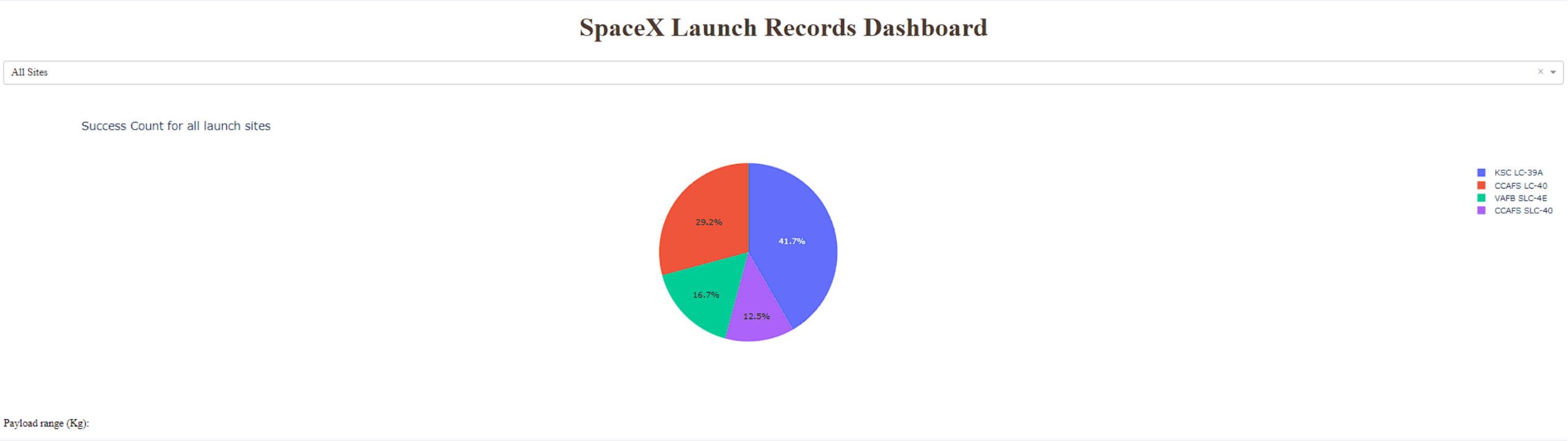


- Launch siteCCAFS SLC-40closest to highway (Washington Avenue) is 23.19km



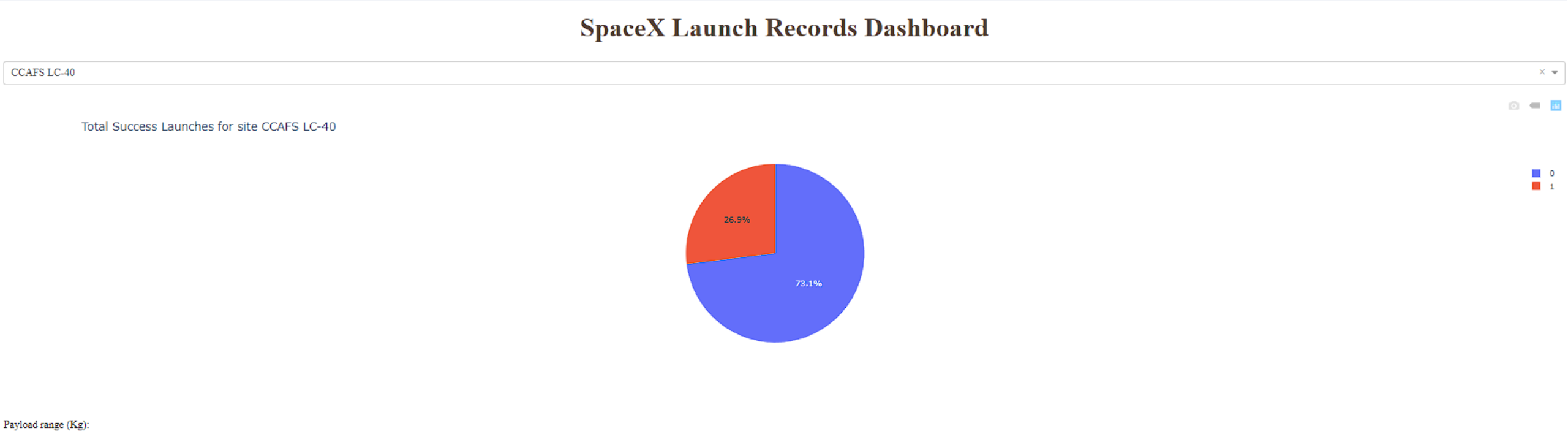

Section 4
Build a Dashboard with Plotly Dash

Pie-Chart for launch success count for all sites



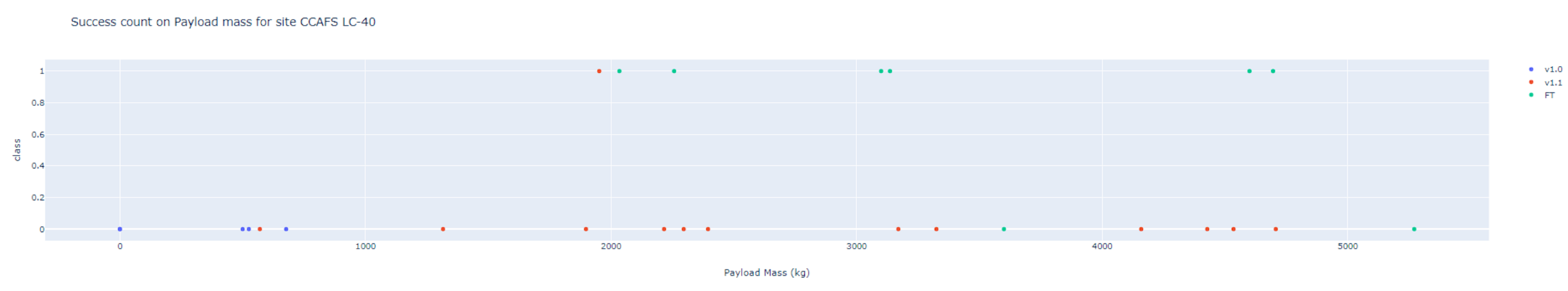
- Launch site KSC LC-39A has the highest launch success rate at 41.7% followed by CCAFS LC-40 at 29.2%, VAFB SLC-4E at 16.7% and lastly launch site CCAFS SLC-40 with a success rate of 12.5%

Pie chart for the launch site with 2nd highest launch success ratio



- Launch site CCAFS LC-40 had the 2ndhighest success ratio of 73.1% success against 26.9% failed launches

Payload vs. Launch Outcome scatter plot for all sites



- For Launch site CCAFS LC-40 the booster version FT has the largest success rate from a payload mass of >2000kg

Section 5

Predictive Analysis (Classification)

Classification Accuracy

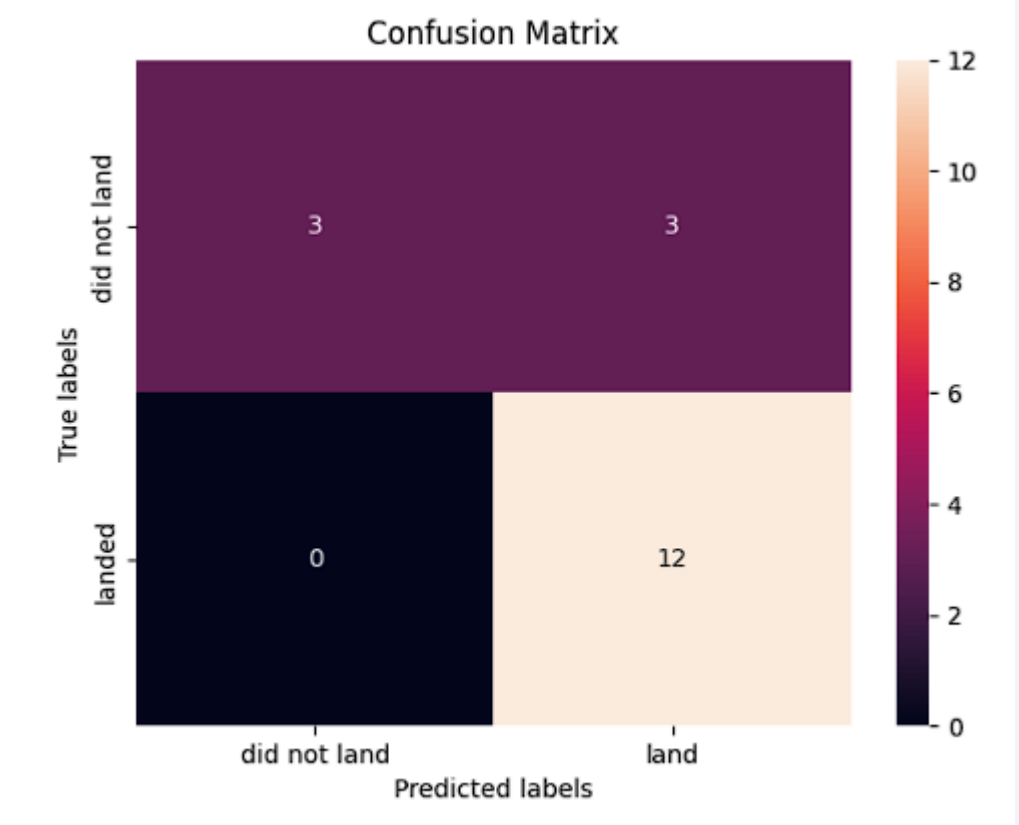
- All methods performed equally on the test data with a value of 83% accuracy.

Out[35]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

Confusion Matrix

- All the 4 classification model had the same confusion matrixes and were able equally distinguish between the different classes. The major problem is false positives for all the models.



Conclusions

- The safest launch site is KSC LC-39A with a success rate of 76.9% which outclasses by a lot the most dangerous place to launch with is CCAFS SLC-40 that have only 57.1%. Due to the price of launches, we can not recommend to keep on using launch cite CCAFS SLC-40 due to its volatility. Further modifications to the site are recommended to improve its success.
- We can deduce that, as the flight number increases in each of the 3 launch sites, so does the success rate. For instance, the success rate for the VAFB SLC 4E launch site is 100% after the Flight number 50^{th} . Both KSC LC 39A and CCAFS SLC 40 have a 100% success rates after 80^{th} launch
- From observing the charts on payload, it can be determined that VAFB-SLC launch site had no infrastructure for rockets launched on heavy payload mass(greater than 10,000kgs).
- Orbits ES-L1, GEO, HEO & SSO have the highest success rates at 100%, with GTO orbit having the lowest success rate at ~50%. Orbit SO has 0% success rate.
- LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.
- We can determine that that for LEO, ISS and SSO, the heaver the payload, the more success they have achieved on their launches. GTO have had a mix of both success and fails which determines that its payload mass can not determine if it will be a successful launch.
- The final conclusion is that as time passes by, we can see a higher success rate in launches. We determine this since how the line of success have changed from 2013 to 2020 in a correlated way.

Thank you!