IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

MICHAEL VENTURA
07/01/2024

Outline

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

Executive Summary

- Summary of methodologies
  - SpaceX Data Collection using SpaceX API
  - SpaceX Data Collection with Web Scraping
  - SpaceX Data Wrangling
  - SpaceX Exploratory Data Analysis using SQL
  - SpaceX EDA DataViz Using Python Pandas and Matplotlib
  - SpaceX Launch Sites Analysis with Folium-Interactive Visual Analytics and Ploty Dash
  - SpaceX Machine Learning Landing Prediction
- Summary of all results
  - EDA Results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis

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 alternative company wants to bid against SpaceX for a rocket launch.
- Problems you want to find answers
- This capstone has the purpose to predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its website.

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

Describe how data sets were collected.

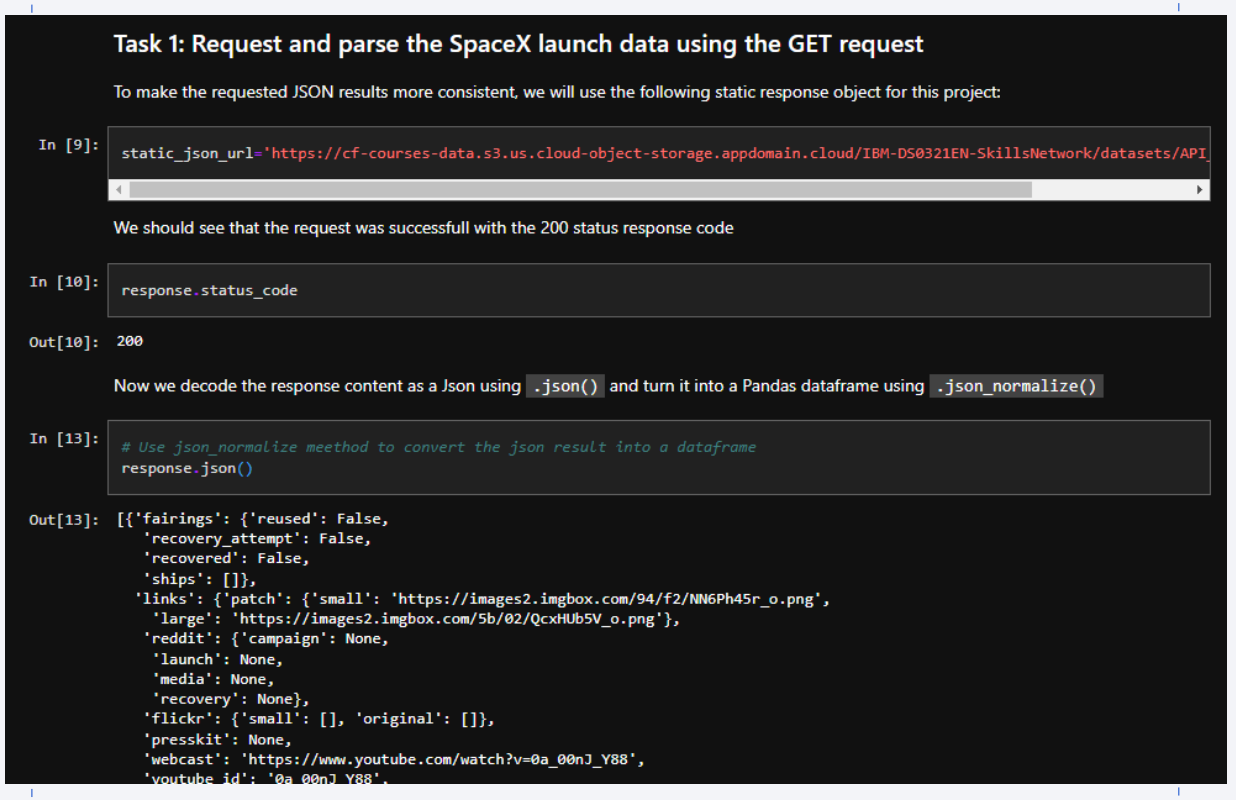
Data was first collected using SpaceX API by making a get request to the SpaceX API. This was done by first defining a series helper functions that would help in the use of the API to extract information using identification numbers in the launch data and then requesting rocket launch data from the SpaceX API url.

Finally to make the requested JSON results more consistent, the SpaceX launch data was requested and parsed using the request and then decoded the response content as a JSON result witch was then converted into a Pandas Data Frame

Data Collection – SpaceX API

- Data Collected using SpaceX 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 JSON result which was then converted into a Pandas Data Frame

- Here is the GitHub URL of the completed SpaceX API calls notebook (https://github.com/michaelventura01/space_x_launch/blob/main/jupyter-labs-spacex-data-collection-api.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:

In [9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_
```

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

In [10]:
```
response.status_code
```

Out[10]: 200

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

In [13]:
```
# Use json_normalize meethod to convert the json result into a dataframe
response.json()
```

Out[13]:
```
[{'fairings': {'reused': False,
   'recovery_attempt': False,
   'recovered': False,
   'ships': []},
  'links': {'patch': {'small': 'https://images2.imgbox.com/94/f2/NN6Ph45r_o.png',
    'large': 'https://images2.imgbox.com/5b/02/QcxHUb5V_o.png'},
   'reddit': {'campaign': None,
    'launch': None,
    'media': None,
    'recovery': None},
   'flickr': {'small': [], 'original': []},
   'presskit': None,
   'webcast': 'https://www.youtube.com/watch?v=0a_00nJ_Y88',
   'youtube_id': '0a_00nJ_Y88',
```

Data Collection - Scraping

- After obtaining and creating a Pandas DF from the collected data, data was filtered using the BoosterVersion column to only keep the Falcon 9 launches, then dealt with the missing data values in the LandingPad and PayloadMass columns. For the PayloadMass, missing data values were replaced using mean value of column.
- Here is the GitHub URL of the completed SpaceX web scraping notebook (https://github.com/michaelventura01/space_x_launch/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb)

Load Space X dataset, from last section.

```
In [5]: df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/data
        df.head(10)
```

Out[5]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block
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
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0

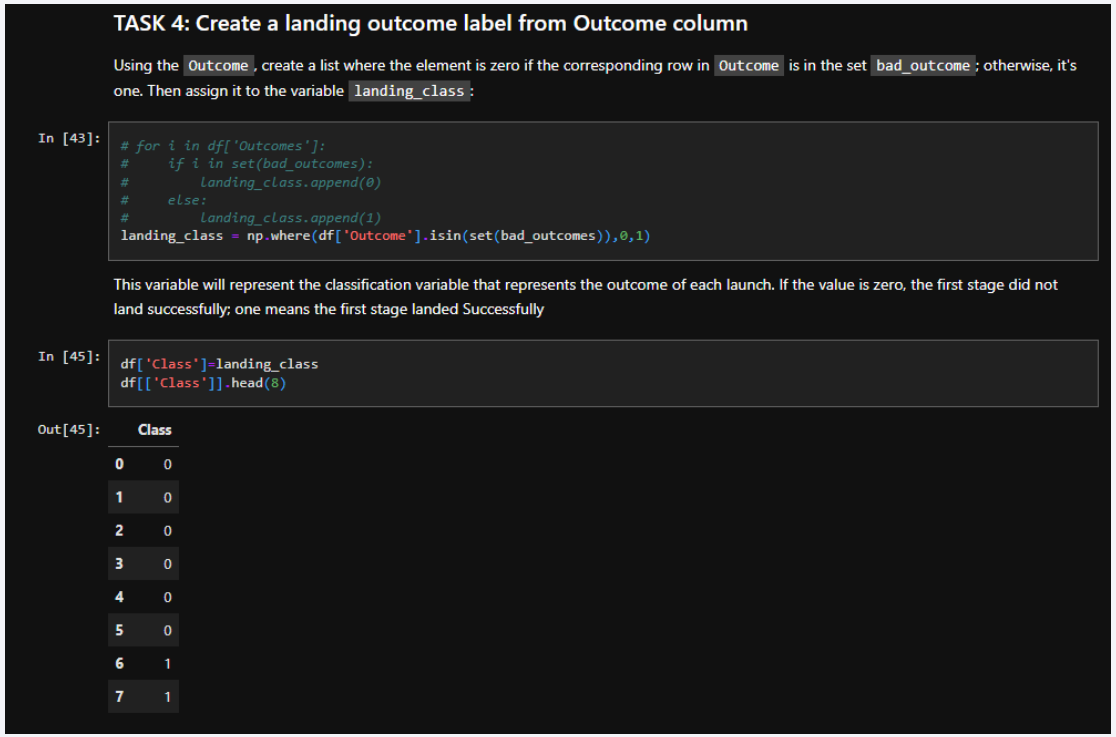
Data Wrangling

After obtaining and creating a Pandas DF from the collected data, data was filtered using the BoosterVersion column to only keep the Falcon 9 launches, then dealt with the missing data values in the LandingPad and PayloadMass columns. For the PayloadMass , missing data values were replaced using mean value of column.

Also performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models

Here is the GitHub URL of the completed SpaceX data wrangling related notebooks, as an external reference and peer-review purpose

https://github.com/michaelventura01/space_x_launch/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb



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`:

In [43]:
```
# for i in df['Outcomes']:
#     if i in set(bad_outcomes):
#         landing_class.append(0)
#     else:
#         landing_class.append(1)
landing_class = np.where(df['Outcome'].isin(set(bad_outcomes)),0,1)
```

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

In [45]:
```
df['Class']=landing_class
df[['Class']].head(8)
```

Out[45]:

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

EDA with Data Visualization

Performed data Analysis and Feature Engineering using Pandas and Matplotlib.i.e.

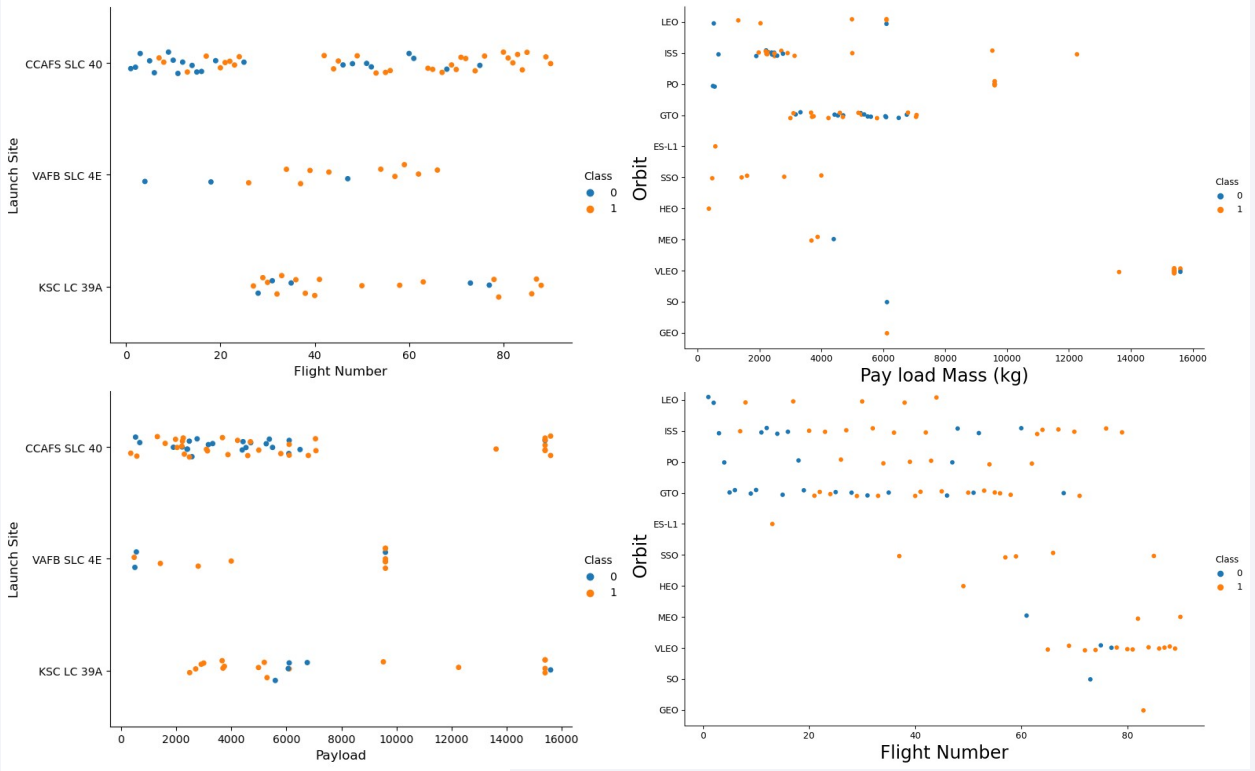
- Exploratory Data Analysis
- Preparing Data Feature Engineering

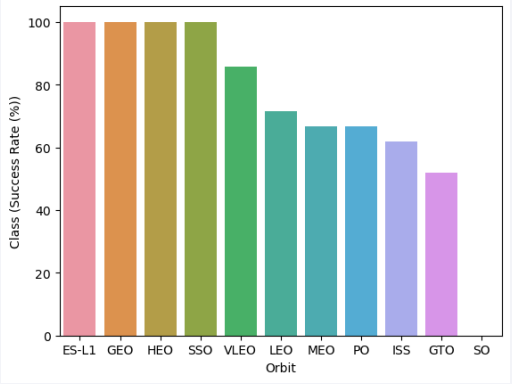
Used scatter plots to Visualize the relationship between Flight Number and Launch Site, Payload and Launch Site, FlightNumber and Orbit type, Payload and Orbit type.

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 SpaceX EDA with data visualization notebook (https://github.com/michaelventura01/space_x_launch/blob/main/edadataviz.ipynb)





EDA with SQL

The following SQL queries were performed for EDA

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

```
%sql select distinct Landing_Outcome from SPACEXTABLE
```

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

```
%sql select * from SPACEXTABLE where Launch_Site like 'CCA%' order by Launch_Site limit 5
```

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

```
%sql select booster_Version,sum(PAYLOAD_MASS__KG_) total_payload_KG from SPACEXTABLE where customer = 'NASA (CRS)'  group b
```

Display average payload mass carried by booster version F9 v1.

```
%sql select booster_Version,avg(PAYLOAD_MASS__KG_) average_payload_KG from SPACEXTABLE where booster_Version like 'F9 v1.1%
```

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.

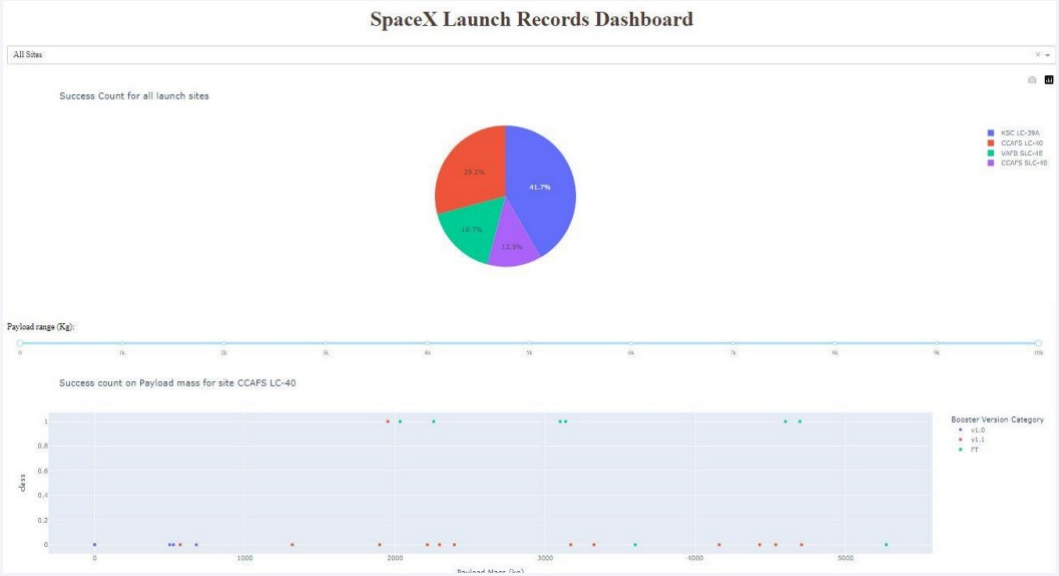
Created a launch set outcomes (failure=0 or success=1)

Here is the GitHub URL of the completed SpaceX of your completed interactive map with Folium map, as an external reference and peer-review purpose (https://github.com/michaelventura01/space_x_launch/blob/main/lab_jupyter_launch_site_location.ipynb)

Build a Dashboard with Plotly Dash

Built an interactive dashboard application with Plotly dash 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
- Addeng a callback function to render the success-payload-scatter-chart scatter plot
- Here is the GitHub URL of the completed SpaceX of your completed Plotly Dash lab, as an external reference and peer-review purpose(https://github.com/michaelventura01/space_x_launch/blob/main/spacex_app/spacex_dash_app.py)



Predictive Analysis (Classification)

Summary of how I built, evaluated, improved, and found the best performing classification model

After loading the data as a Pandas Dataframe, I set out to perform exploratory Data Analysis and determine Training Labels by;

- creating a NumPy array 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.
- After which 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)

In order to find the best ML model/ method that would performs best using the test data between SVM, Classification Trees, k nearest neighbors and Logistic Regression;

- First created an object for each of the algorithms then created a GridSearchCV object and assigned them a set of parameters for each model.
- For each of the models under evaluation, the GridsearchCV object was created with cv=10, then fit the training data into the GridSearch object for each to Find best Hyperparameter.
- After fitting the training set, we output GridSearchCV object for each of the models, then displayed the best parameters using the data attribute best_params_ and the accuracy on the validation data using the data attribute best_score_.
- Finally using the method score to calculate the accuracy on the test data for each model and plotted a confussion matrix for each using the test and predicted outcomes.

Predictive Analysis (Classification)

The table below shows the test data accuracy score for each of the methods comparing them to show which performed best using the test data between SVM, Classification Trees, k nearest neighbors and Logistic Regression;

Here is the GitHub URL of the completed SpaceX of predictive analysis (https://github.com/michaelventura01/space_x_launch/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb)

Find the method performs best:

In [130…

```
Report = pd.DataFrame({'Method' : ['Test Data Accuracy']})

knn_accuracy=knn_cv.score(X_test, Y_test)
Decision_tree_accuracy=tree_cv.score(X_test, Y_test)
SVM_accuracy=svm_cv.score(X_test, Y_test)
Logistic_Regression=logreg_cv.score(X_test, Y_test)

Report['Logistic_Reg'] = [Logistic_Regression]
Report['SVM'] = [SVM_accuracy]
Report['Decision Tree'] = [Decision_tree_accuracy]
Report['KNN'] = [knn_accuracy]

Report.transpose()
```

Out[130…

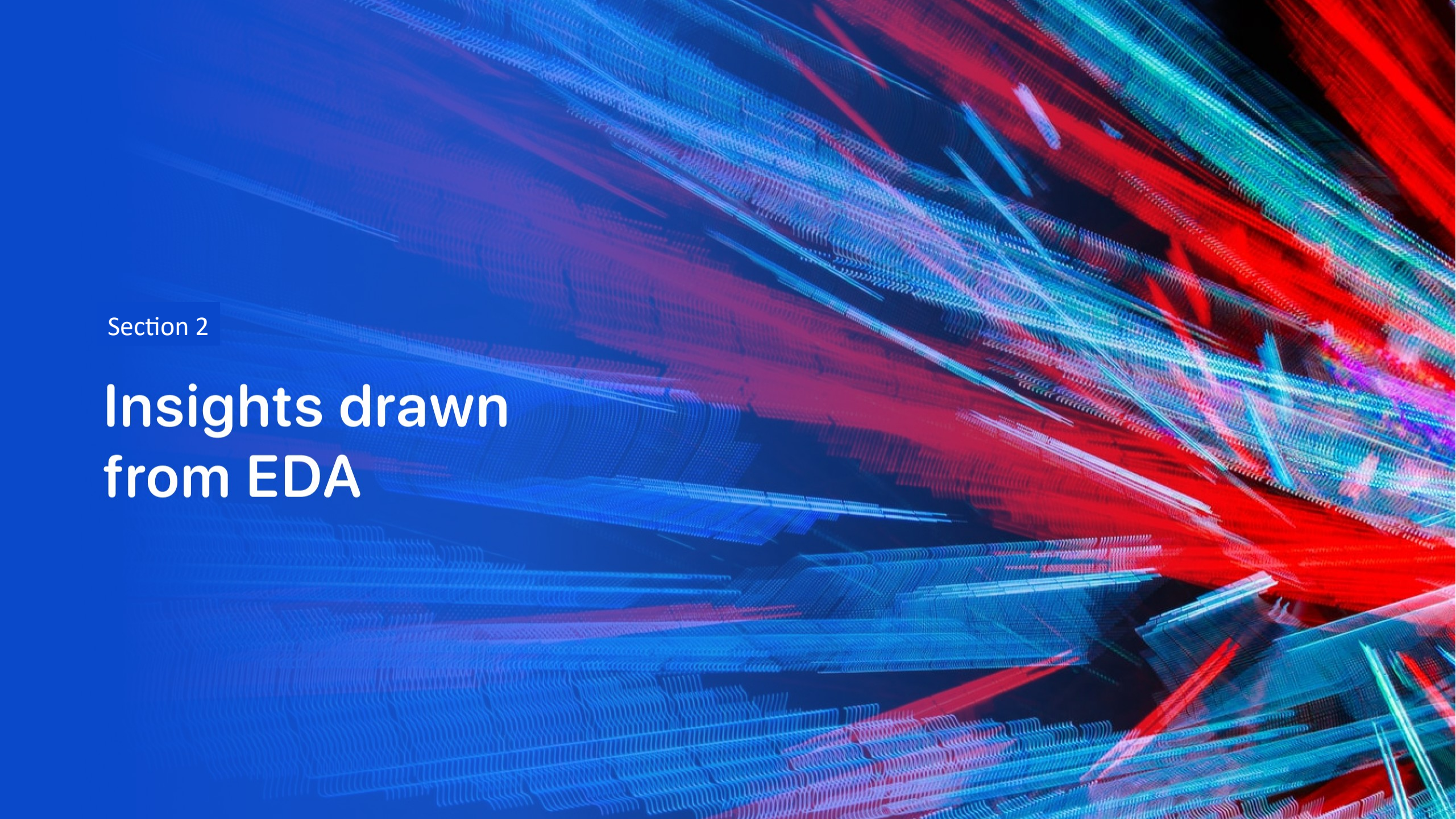
	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.777778
KNN	0.833333

Results

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

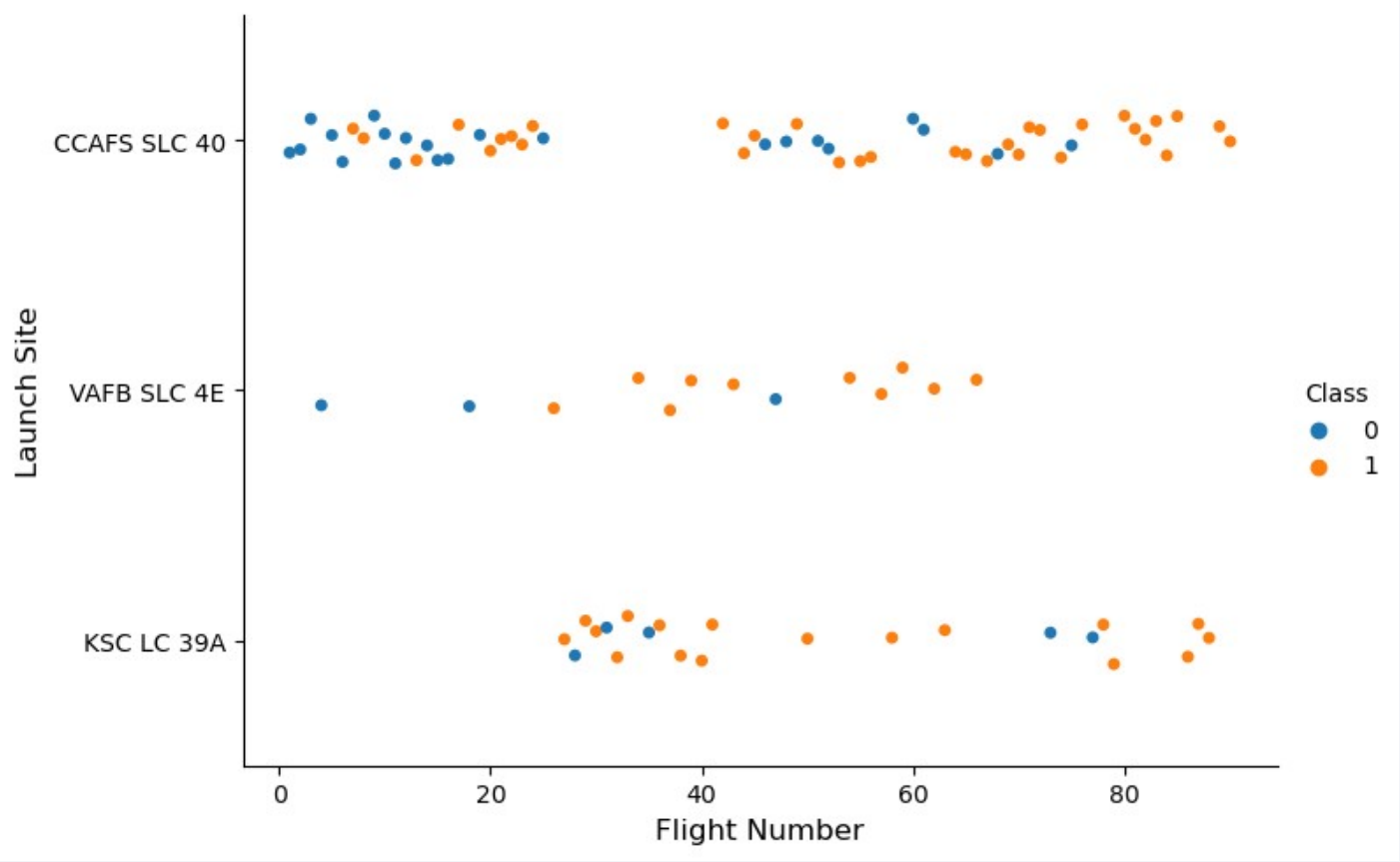
Section 2

Insights drawn from EDA



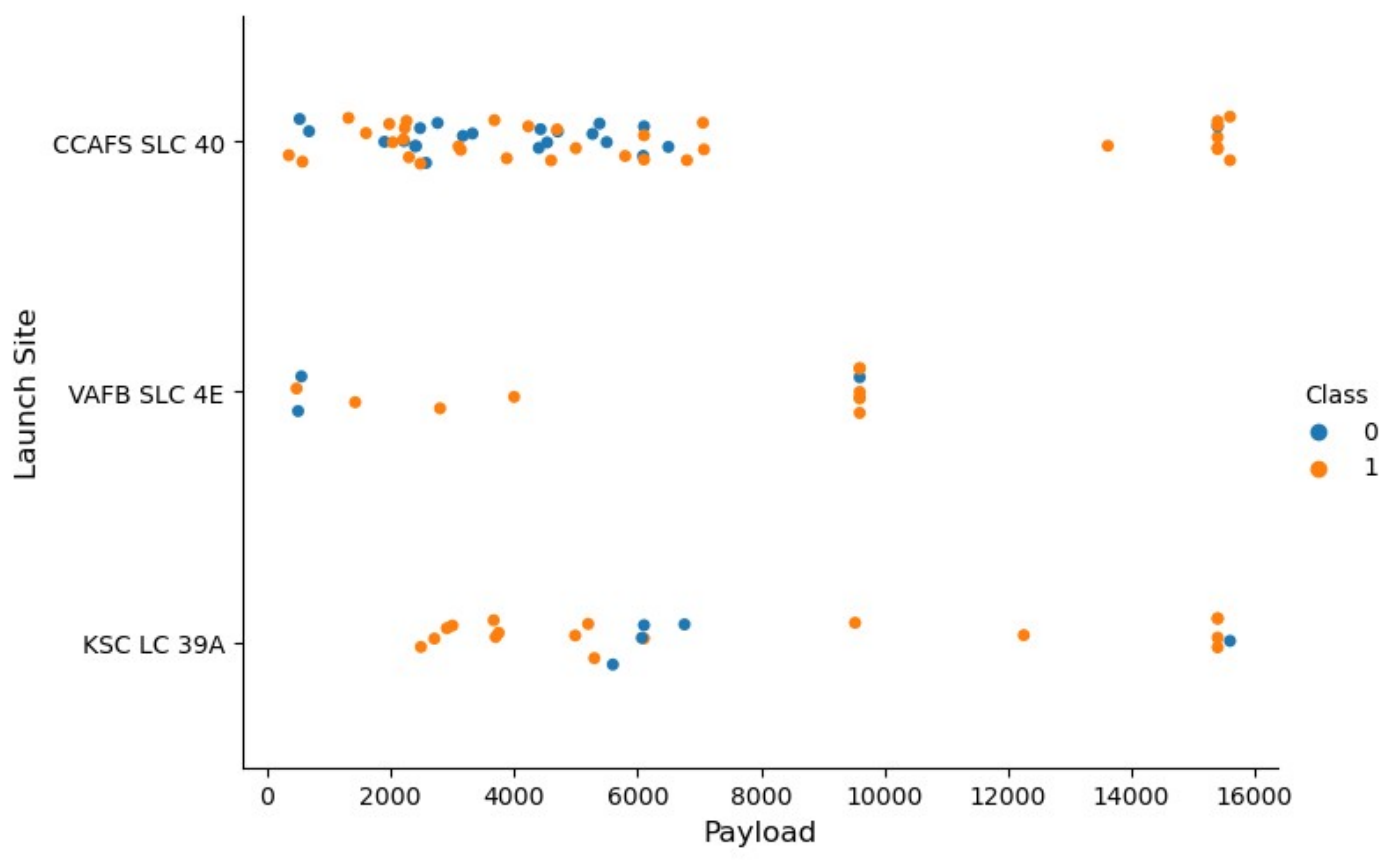
Flight Number vs. Launch Site

Scatter plot of Flight Number vs Launch Site



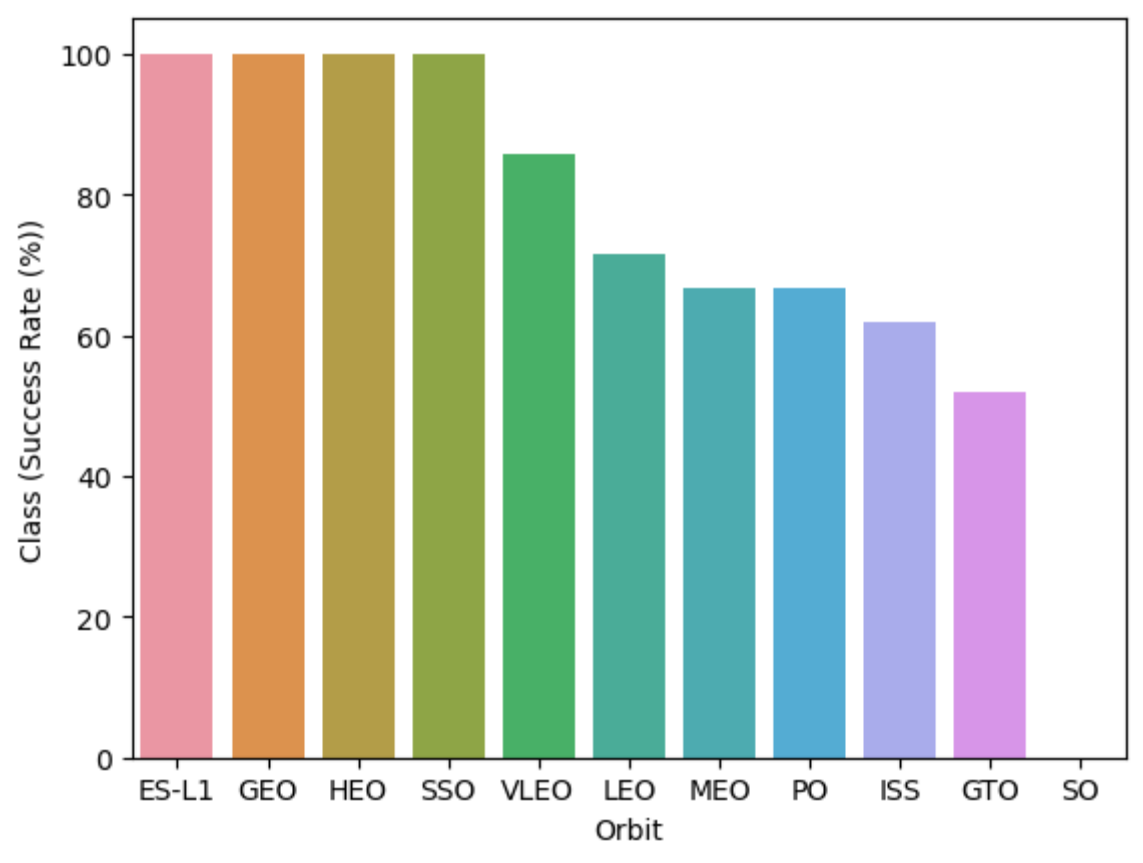
Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site



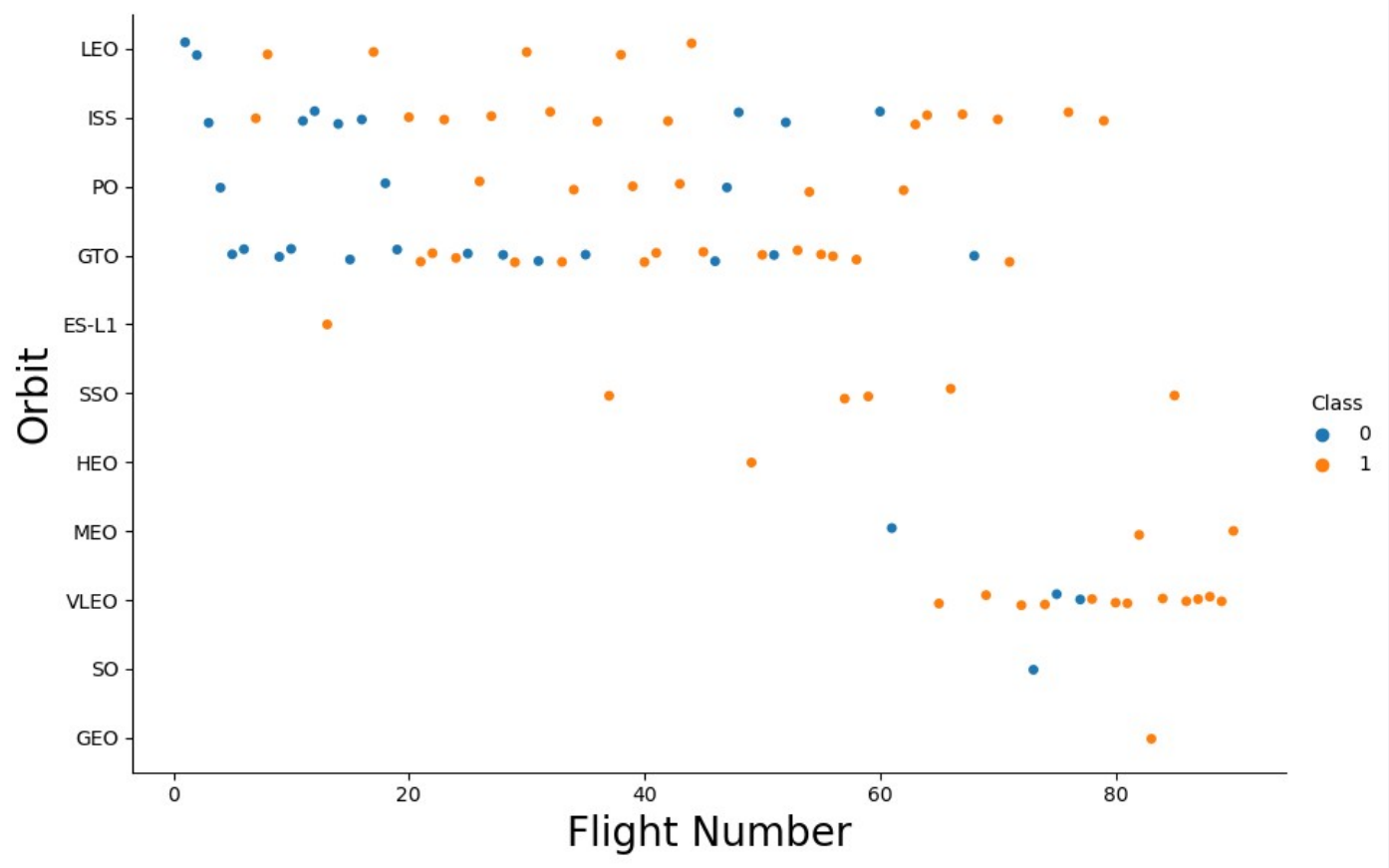
Success Rate vs. Orbit Type

Show a bar chart for the success rate of each orbit type



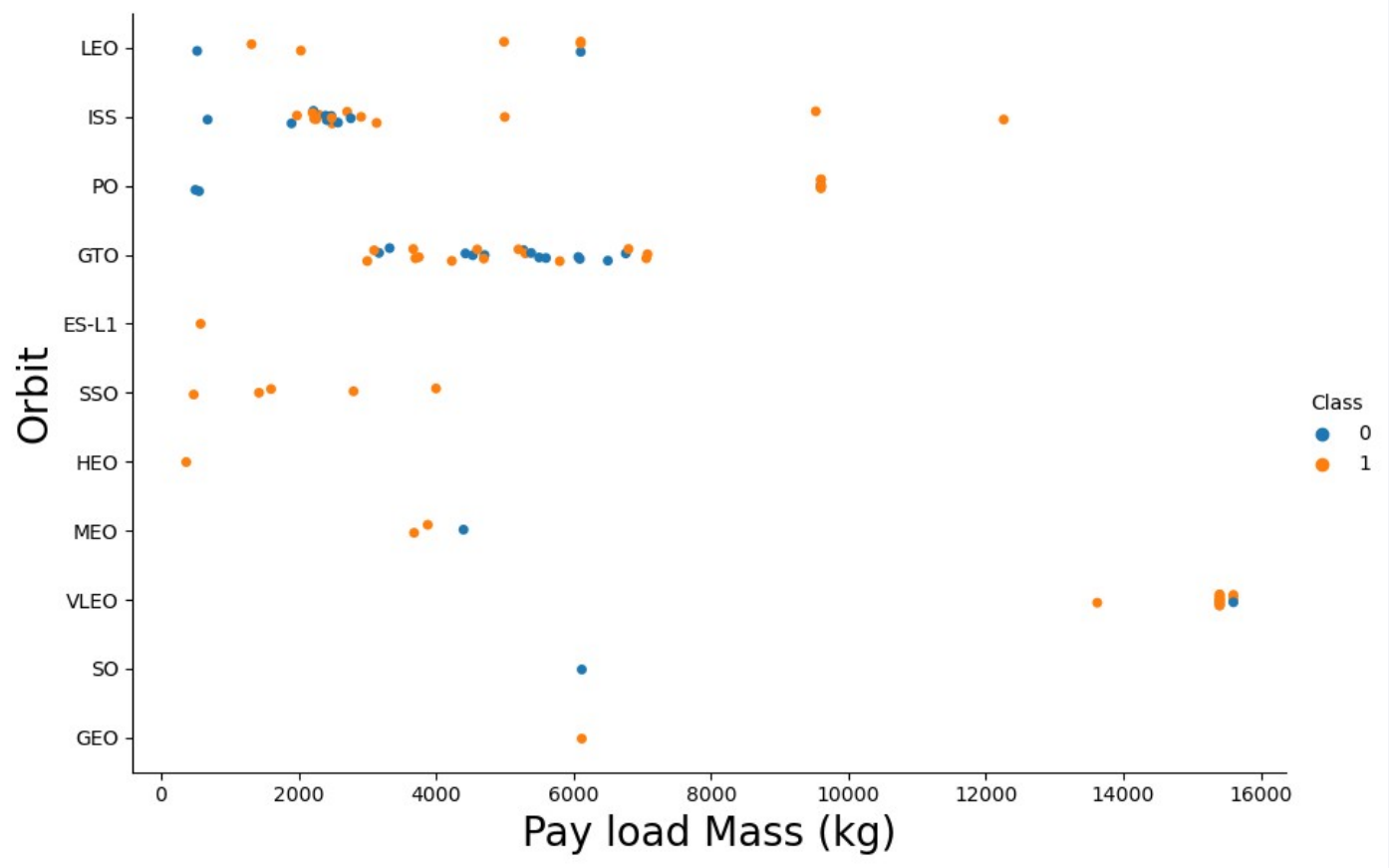
Flight Number vs. Orbit Type

Show a scatter point of Flight number vs. Orbit type



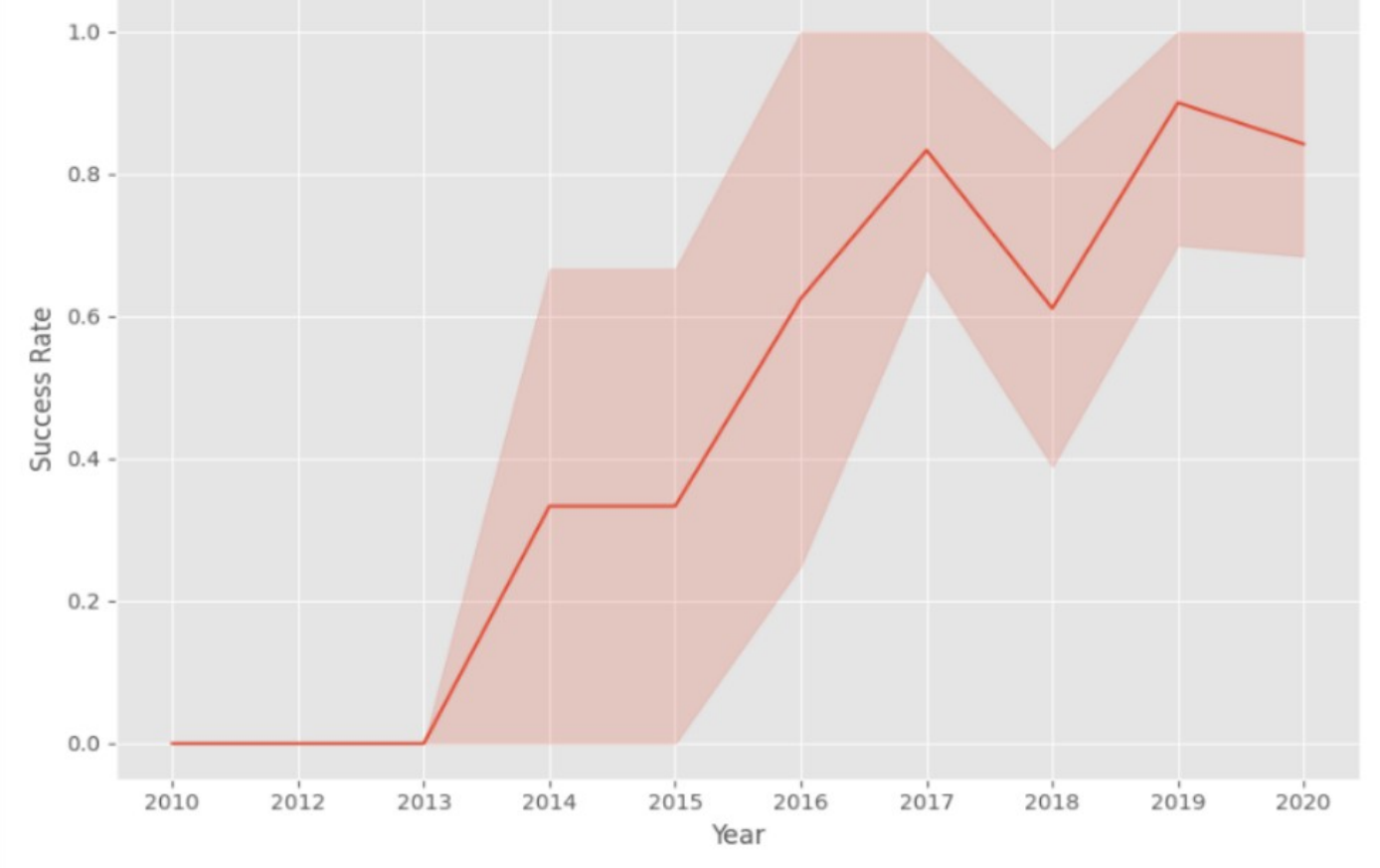
Payload vs. Orbit Type

Show a scatter point of payload vs. orbit type



Launch Success Yearly Trend

Show a line chart of yearly average success rate



All Launch Site Names

Find the names of the unique launch sites

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

```
In [10]: %sql select distinct Landing_Outcome from SPACEXTABLE
 * sqlite:///my_data1.db
Done.
```

Out[10]:

Landing_Outcome
Failure (parachute)
No attempt
Uncontrolled (ocean)
Controlled (ocean)
Failure (drone ship)
Precluded (drone ship)
Success (ground pad)
Success (drone ship)
Success
Failure
No attempt

Launch Site Names Begin with 'CCA'

Find 5 records where launch sites begin with `CCA`

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

```
In [11]: %sql select * from SPACEXTABLE where Launch_Site like 'CCA%' order by Launch_Site limit 5
 * sqlite:///my_data1.db
Done.
```

Out[11]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

Calculate the total payload carried by boosters from NASA

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

```
In [13]: %sql select booster_Version,sum(PAYLOAD_MASS__KG_) total_payload_KG from SPACEXTABLE where customer = 'NASA (CRS)'  group by
```

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

Out[13]:

Booster_Version	total_payload_KG
F9 v1.0 B0006	500
F9 v1.0 B0007	677
F9 v1.1 B1015	1898
F9 v1.1 B1018	1952
F9 B5 B1059.2	1977
F9 FT B1035.2	2205
F9 v1.1 B1010	2216
F9 FT B1025.1	2257
F9 B5 B1056.2	2268
F9 v1.1	2296
F9 v1.1 B1012	2395
F9 FT B1031.1	2490
F9 B5B1056.1	2495
F9 B5B1050	2500
F9 B4 B1039.2	2647
F9 B4 B1045.2	2697
F9 FT B1035.1	2708
F9 B5 B1058.4	2972
F9 FT B1021.1	3136
F9 B4 B1039.1	3310

Average Payload Mass by F9 v1.1

Calculate the average payload mass carried by booster version F9 v1.1

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

In [14]:
```
%sql select booster_Version,avg(PAYLOAD_MASS__KG_) average_payload_KG from SPACEXTABLE where booster_Version like 'F9 v1.1%
```

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

Out[14]:

Booster_Version	average_payload_KG
F9 v1.1 B1003	500.0
F9 v1.1 B1017	553.0
F9 v1.1 B1013	570.0
F9 v1.1	1316.0
F9 v1.1 B1015	1898.0
F9 v1.1 B1018	1952.0
F9 v1.1 B1010	2216.0
F9 v1.1	2296.0
F9 v1.1 B1012	2395.0
F9 v1.1	3170.0
F9 v1.1	3325.0
F9 v1.1 B1014	4159.0
F9 v1.1 B1011	4428.0
F9 v1.1	4535.0
F9 v1.1 B1016	4707.0

First Successful Ground Landing Date

Find the dates of the first successful landing outcome on ground pad

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

Hint:Use min function

In [15]:
```
%sql select * from SPACEXTABLE where landing_outcome = 'Success (ground pad)' order by Date asc limit 1
```

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

Out[15]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

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

```
In [18]: %sql select Booster_Version from SPACEXTABLE where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ between 4
```

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

Out[18]:

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

Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcomes

List the total number of successful and failure mission outcomes

```
In [19]: %sql select Mission_Outcome, COUNT(Mission_Outcome) TOTAL from SPACEXTABLE GROUP BY Mission_Outcome
 * sqlite:///my_data1.db
Done.
```

Out[19]:

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

Boosters Carried Maximum Payload

List the names of the booster which have carried the maximum payload mass

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

```
In [21]: %sql select Booster_Version from SPACEXTABLE where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
 * sqlite:///my_data1.db
Done.
```

Out[21]:

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

2015 Launch Records

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

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

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

In [47]:
```
%sql SELECT substring(cast(DATE as varchar(10)),6,2) date,MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE FROM SPACEXTBL where
```

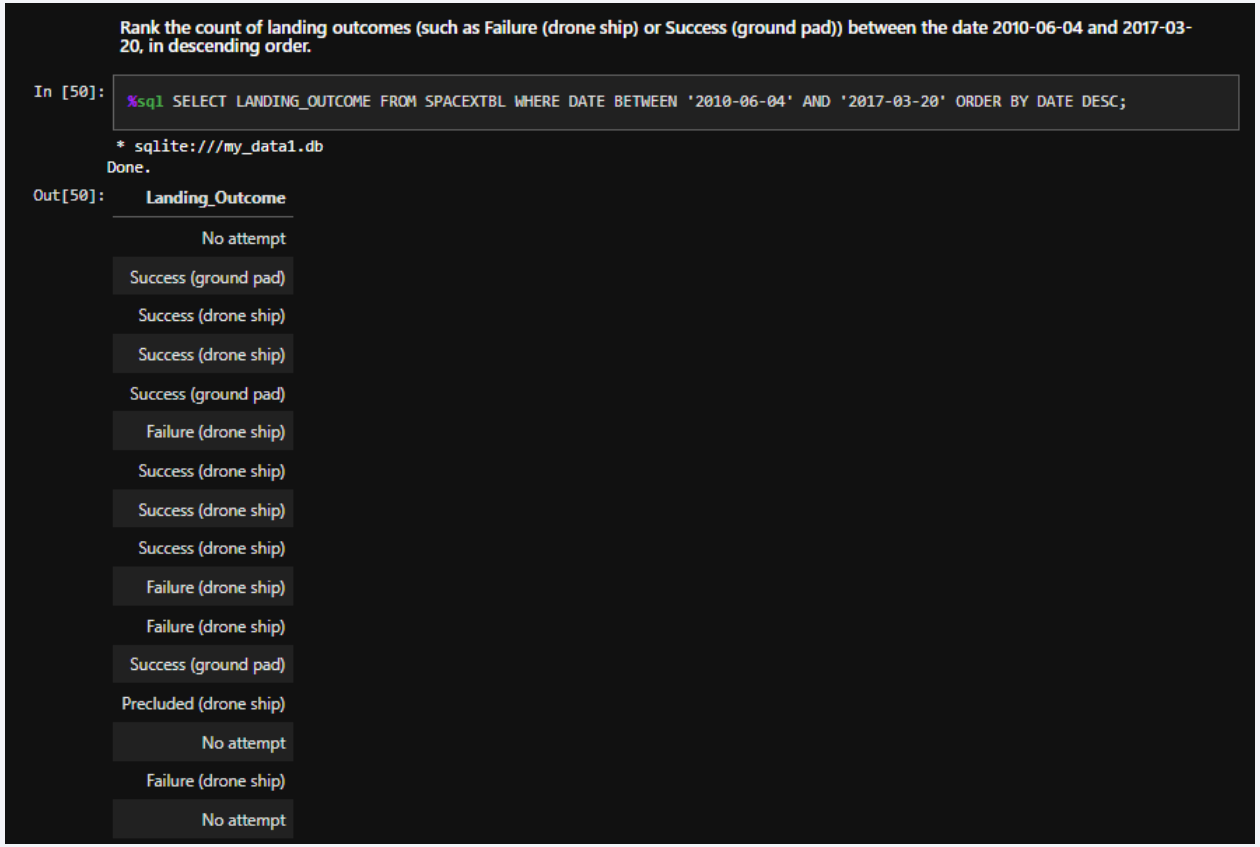
```
 * sqlite:///my_data1.db
Done.
```

Out[47]:

date	Mission_Outcome	Booster_Version	Launch_Site
01	Success	F9 v1.1 B1012	CCAFS LC-40
02	Success	F9 v1.1 B1013	CCAFS LC-40
03	Success	F9 v1.1 B1014	CCAFS LC-40
04	Success	F9 v1.1 B1015	CCAFS LC-40
04	Success	F9 v1.1 B1016	CCAFS LC-40
06	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	CCAFS LC-40

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

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



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 [50]:
```
%sql SELECT LANDING_OUTCOME FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY DATE DESC;
```

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

Out[50]:

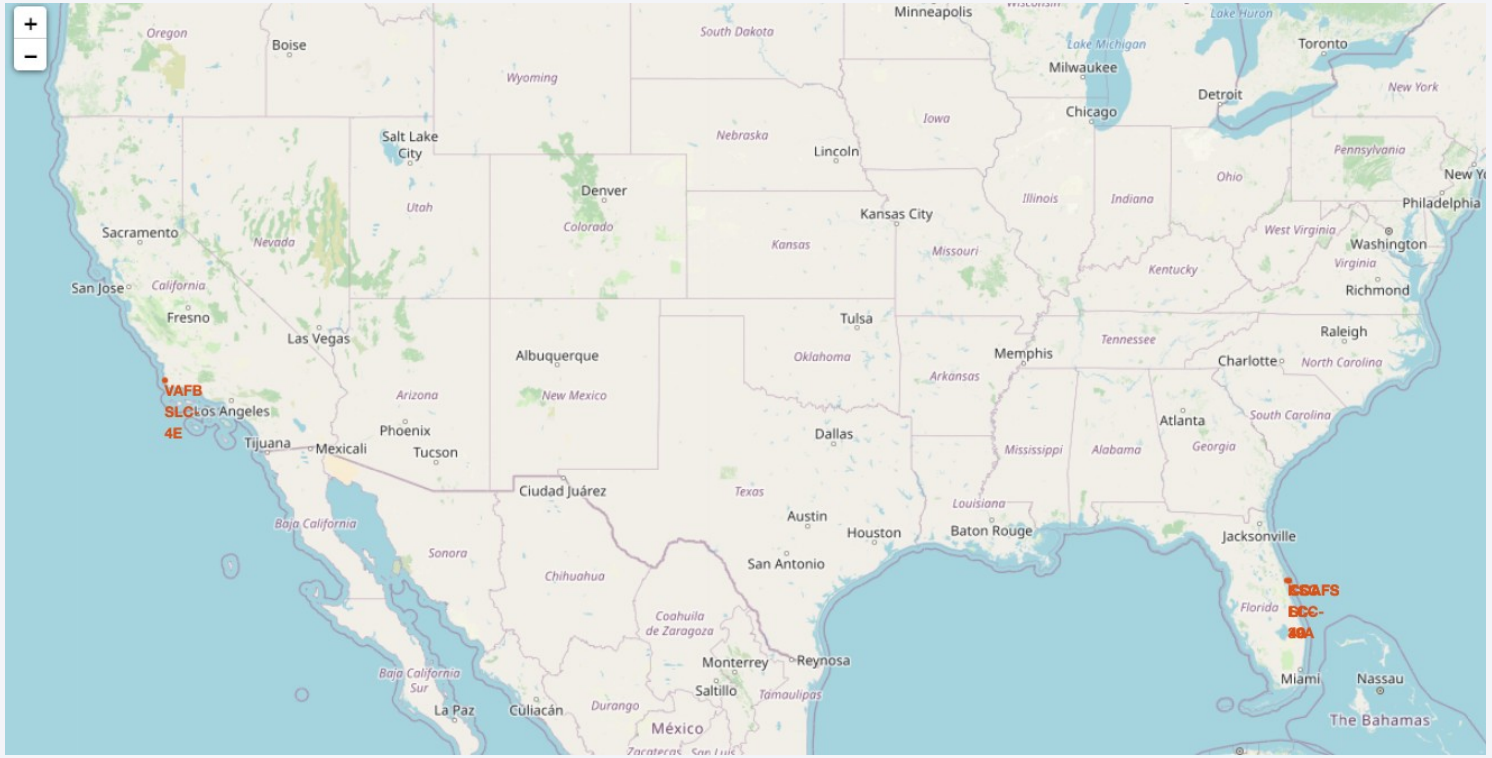
Landing_Outcome
No attempt
Success (ground pad)
Success (drone ship)
Success (drone ship)
Success (ground pad)
Failure (drone ship)
Success (drone ship)
Success (drone ship)
Success (drone ship)
Failure (drone ship)
Failure (drone ship)
Success (ground pad)
Precluded (drone ship)
No attempt
Failure (drone ship)
No attempt

Section 3

Launch Sites Proximities Analysis

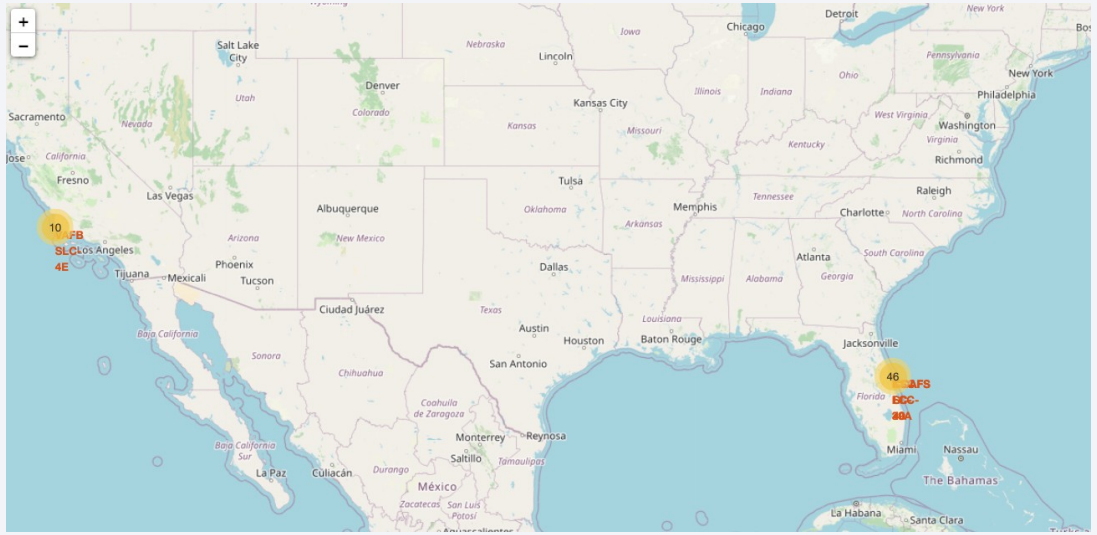
Mark all launch sites on a map

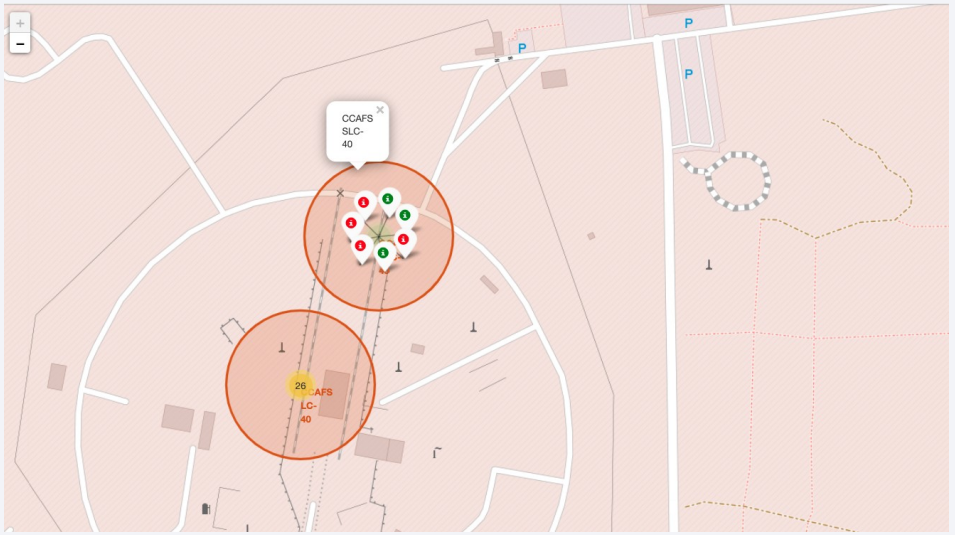
All launch sites are in proximity to the Equator, (located southwards of the US map).
Also all the laumch sites are in very close proximity to the coast.



Mark the success/failed launches for each site on the map

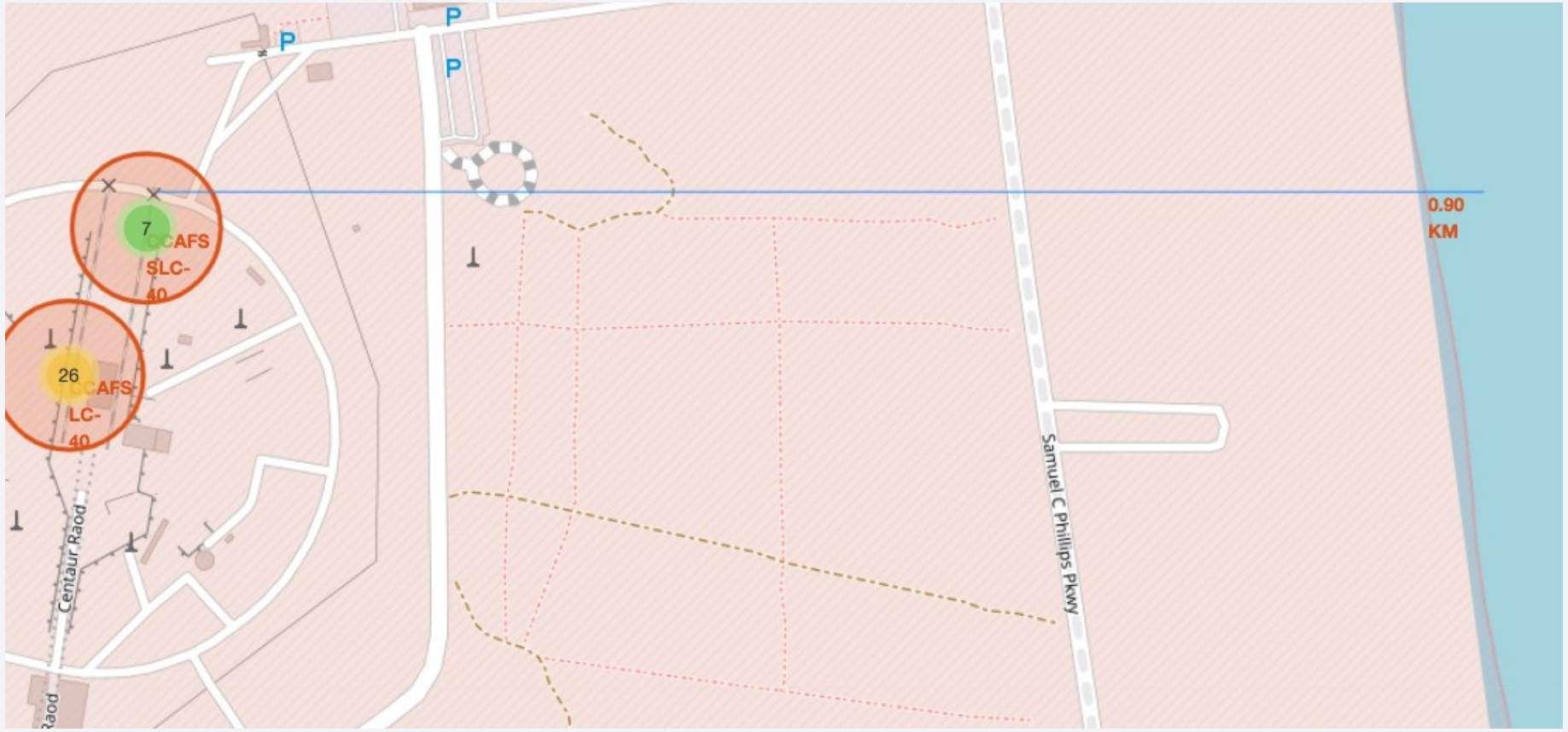
All launch sites are in proximity to the Equator, (located southwards of the US map). Also all the laumch sites are in very close proximity to the coast.





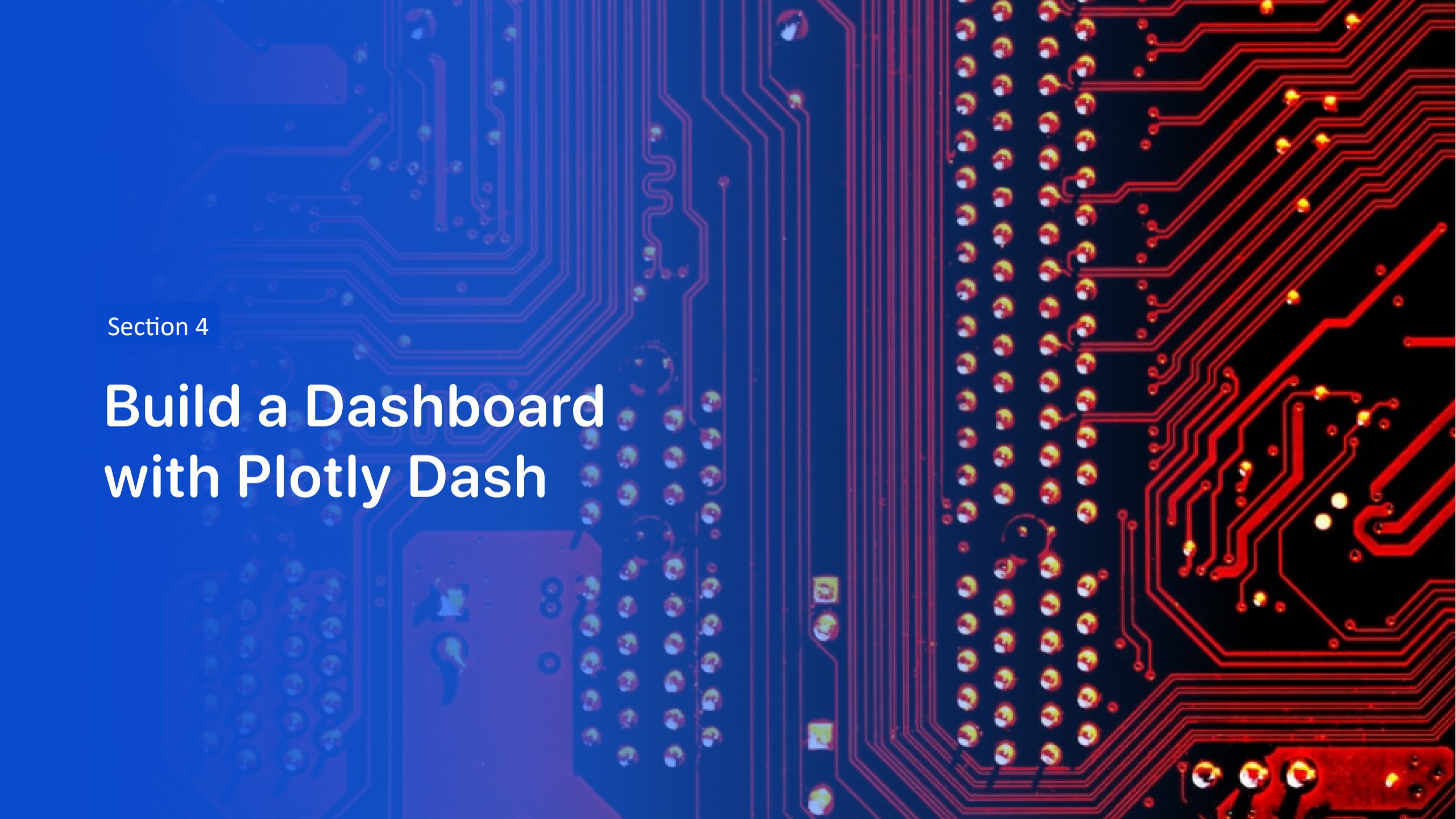
Distances between a launch site to its proximities

In the Eastern coast (Florida) Launch site KSC LC-39A has relatively high success rates compared to CCAFS SLC-40 & CCAFS LC-40.



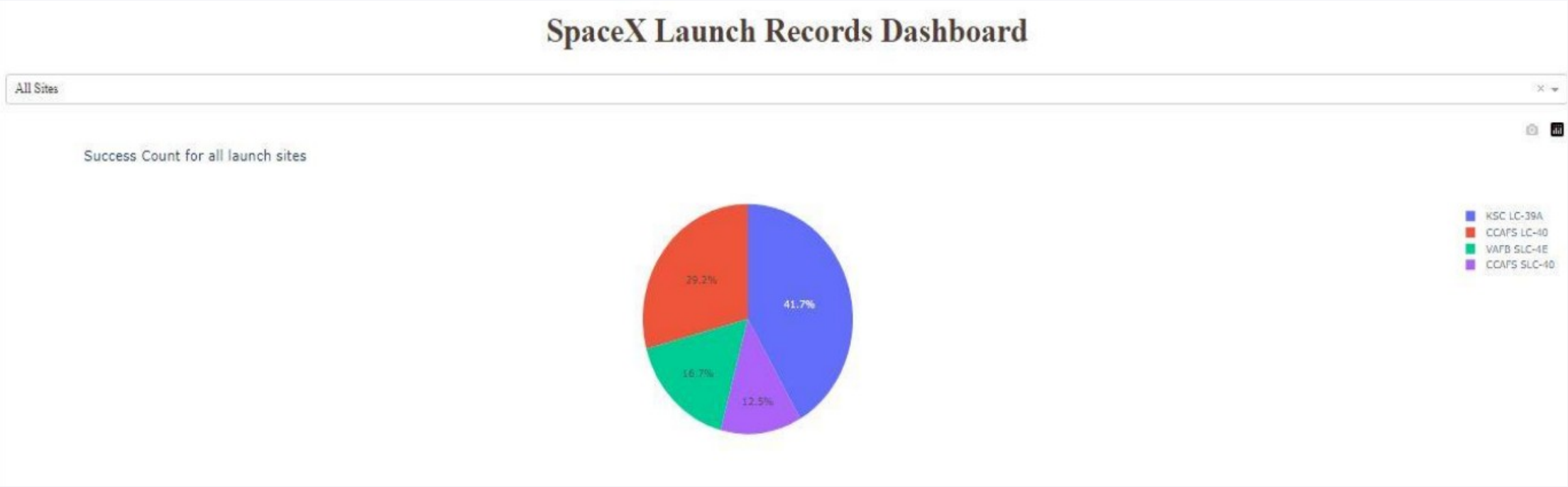
Section 4

Build a Dashboard with Plotly Dash



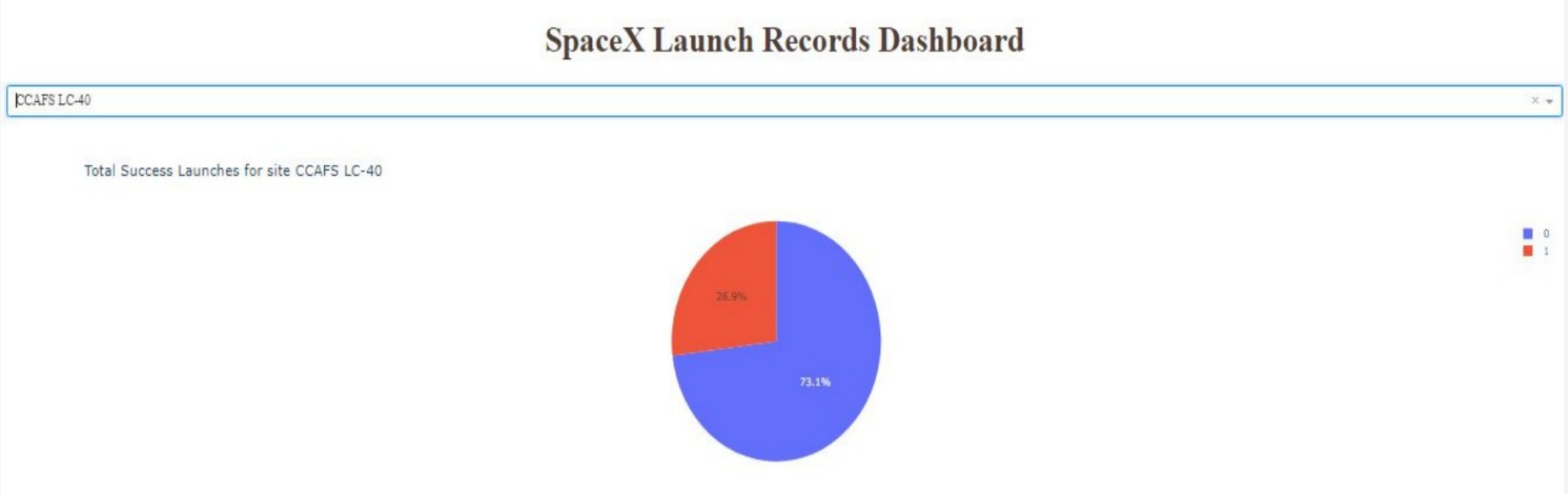
Pie-Chart for launch success count for all sites

Launch site KSC LC-39A has the highest launch success rate at 42% followed by CCAFS LC-40 at 29%, VAFB SLC-4E at 17% and lastly launch site CCAFS SLC-40 with a success rate of 13%



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

Launch site CCAFS LC-40 had the 2nd highest success ratio of 73% success against 27% 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

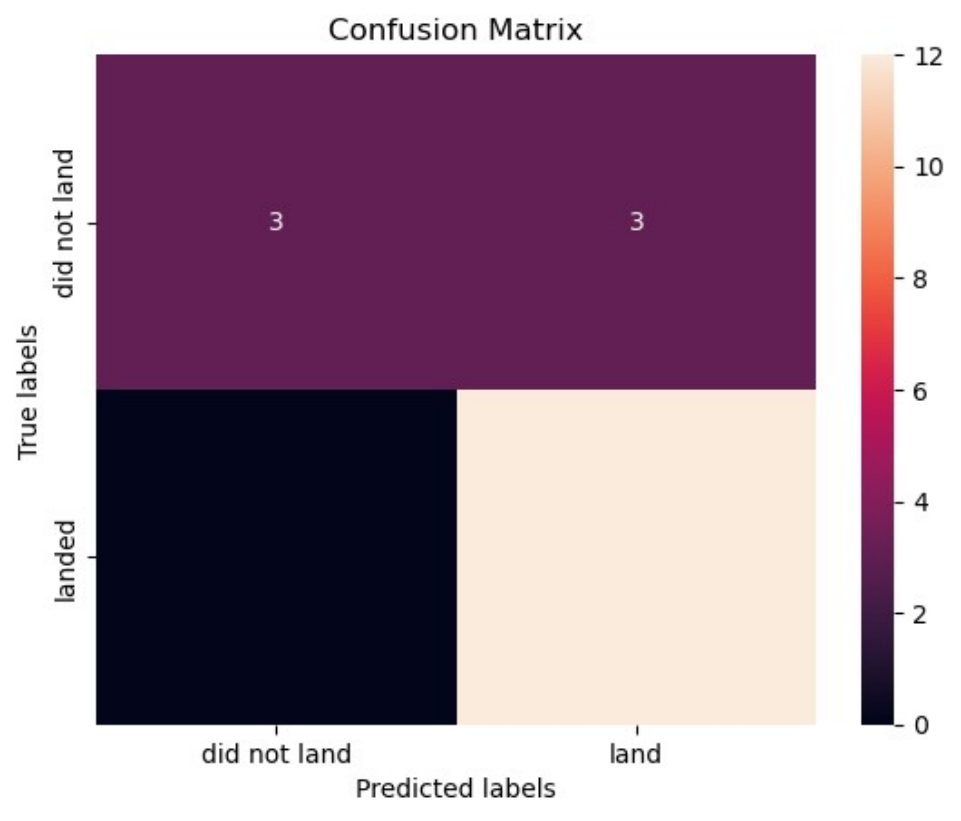
- Visualize the built model accuracy for all built classification models, in a bar chart

Out[130…

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.777778
KNN	0.833333

Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation



Conclusions

Different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.

We can deduce that, as the flight number increases in each of the 3 launcg 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. Both KSC LC 39A and CCAFS SLC 40 have a 100% success rates after 80th flight

If you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

Orbits ES-L1, GEO, HEO & SSO have the highest success rates at 100%, with SO 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

Appendix

With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS. However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here

Anf finally the sucess rate since 2013 kept increasing till 2020.

Thank you!