IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Aakash Bhardwaj>
<25-01-2022>

Outline

EXECUTIVE SUMMARY

INTRODUCTION

METHODOLOGY

RESULTS

CONCLUSION

APPENDIX

Executive Summary

- Summary of methodologies
  - Collecting relevant data and improving the quality by performing data wrangling
  - Exploring Analysis of data using SQL
  - Exploratory Analysis of data using Pandas and Matplotlib
  - Building a dashboard to analyze launch records interactively with Plotly Dash
  - Building an interactive map to analyze the launch site proximity
  - Using machine learning skills to build a predictive model
- Summary of all results

Introduction

- Project background and context

  The commercial space age is here, companies are making space travel affordable for everyone. Many companies are trying to make space travel affordable. The most successful perhaps is SpaceX. SpaceX Falcon 9 rocket launches cost 62 million USD whereas other providers cost upward of 165 million USD each. Much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine the Falcon 9 first stage will land successfully, we can determine the cost of a launch.

- Problems you want to find answers

  - As a data scientist working for a new rocket company called SpaceY, which wants to bid against SpaceX for a rocket launch I am trying to determine the price of each launch.
  - What are the factors that influence the successful landing of the first stage
  - Do launch sites affect the success rate of landing
  - Does the Payload Mass also affect the success rate of landing

Section 1

Methodology

Methodology

Executive Summary

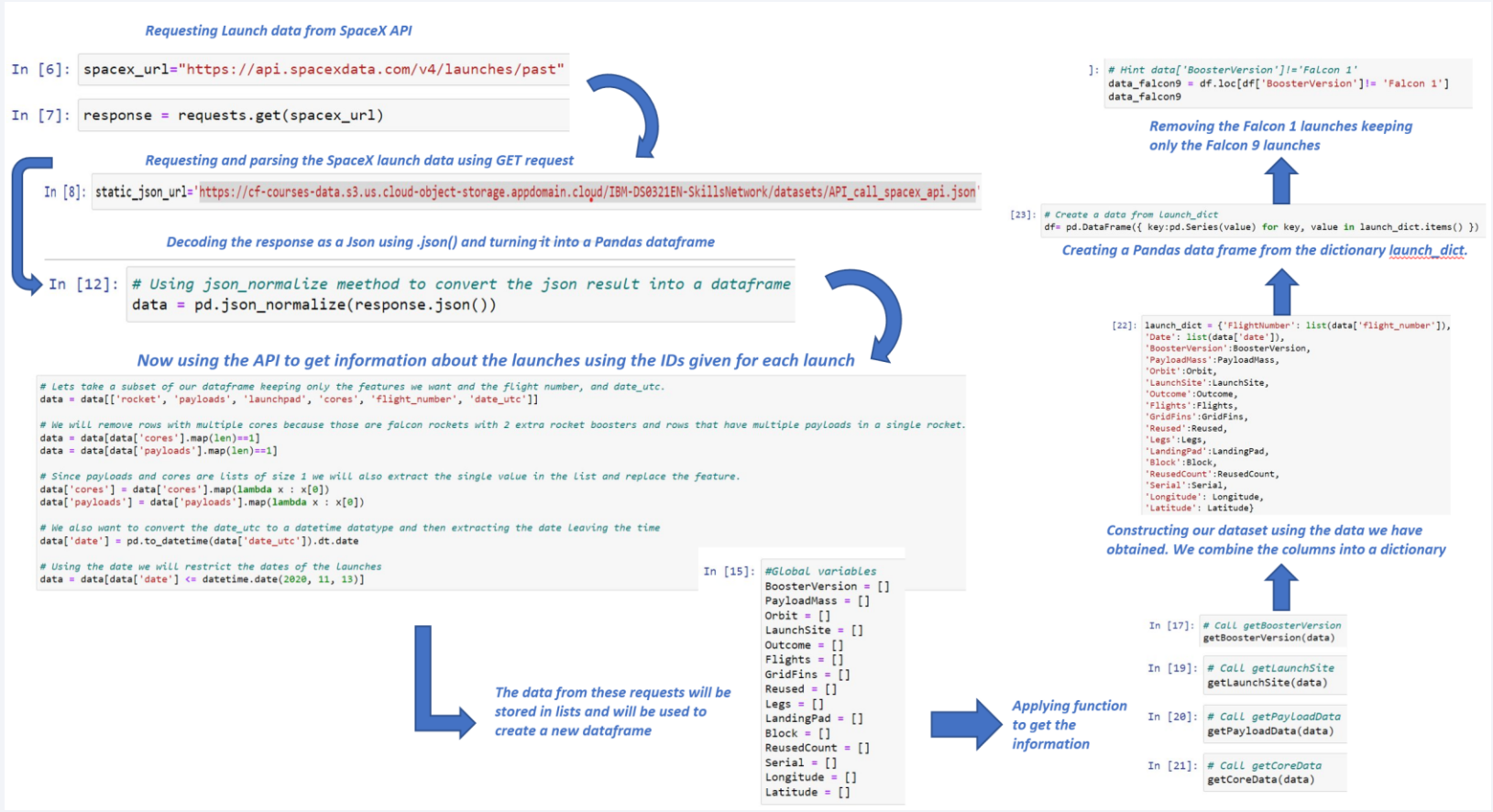
- Data collection methodology:
  - Collected the launch data from SpaceX REST API
  - Web-scraping Wikipedia's Falcon 9 launches page. Link
- Performed data wrangling
  - Null values were either replaced with the average or removed in some cases
  - Categorical variables were converted to binary vectors using One-Hot Encoding
- Performed exploratory data analysis (EDA) using visualization and SQL
- Performed interactive visual analytics using Folium and Plotly Dash
- Performed predictive analysis using classification models
  - Standardizing the data, training the data and finding the best model

Data Collection

- The first source of data collection was from SpaceX REST API.
  - The API gave us data about launches, including information about rocket used, payload delivered, launch specifications, landing specifications and landing outcomes.
- Data was also collected by performing Web-scraping using BeautifulSoup on Wikipedia page titled List of Falcon 9 and Falcon Heavy launches.
  - I was able to extract a Falcon 9 launch HTML table and parse the table and convert it into a Pandas data frame.

Data Collection – SpaceX API

- First we perform a GET request using the requests library to obtain the launch data, which we will use to get the data from the API.
- Our response will be in the form of a JSON, specifically a list of JSON objects.
- Specifically, we have a list of JSON objects which each represent a launch.
- To convert this JSON to a dataframe, we use the json_normalize function.
- This function will allow us to normalize the structured json data into a flat table.



GitHub https://github.com/aakash9107/IBM-/blob/main/Data%20Collection%20API%20%20.ipynb

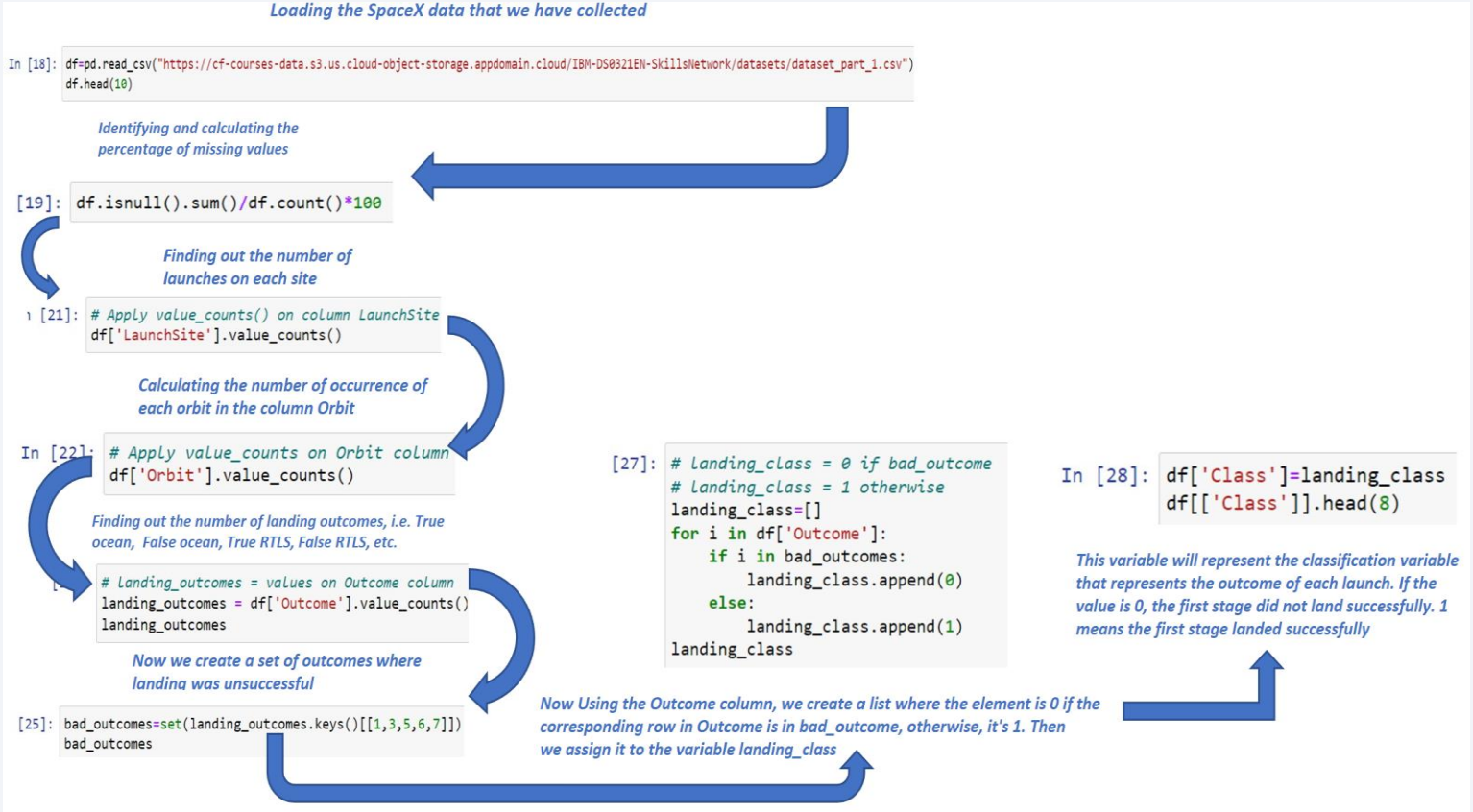
Data Collection – Web Scraping

- We use BeautifulSoup package to web scrape HTML tables that contain valuable Falcon 9 launch records.
- We parse the data from those tables and convert them into a Pandas data frame.
- After converting it into a Pandas data frame we create an empty dictionary and then fill up that dictionary with the extracted records

https://github.com/aakash9107/IBM-/blob/main/Data%20Collection%20with%20Web%20Scraping.ipynb

Data Wrangling

- In the data set, there are several different cases where the booster did not land successfully.
- Sometimes a landing was attempted but failed due to an accident for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean. True RTLS means the mission outcome was successfully landed to a ground pad, False RTLS means the mission outcome was unsuccessfully landed to a ground pad. True ASDS means the mission outcome was successfully landed on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship.
- We accumulate all the successful landings and unsuccessful landings and convert them into training labels.
- 1 represents successful landing, 0 represents unsuccessful landing.



GitHub

https://github.com/aakash9107/IBM-/blob/main/Data%20Wrangling.ipynb

EDA with Data Visualization

- Scatter Plot
  - A scatter plot was first drawn to between Flight number (indicating the number of launch attempts) and Payload Mass to see how these 2 variables affect the outcome.
  - It was used to find out the relationship between Flight Number and Launch Site.
  - Another scatter plot was drawn to observe if there is any relationship between Launch Sites and Payload mass.
  - We also draw a scatter plot to visualize the relationship between Flight Number and Orbit type.
  - Then lastly we draw one to understand the relationship between Payload and Orbit type.
- Bar Chart
  - We draw one bar chart between Success rate and Orbit type to understand which orbits have more successful landings.
- Line Plot
  - We draw one line chart to get the average launch success trend.

https://github.com/aakash9107/IBM-/blob/main/EDA%20with%20Data%20Visualization%20.ipynb

EDA with SQL

- There were multiple SQL queries that we performed to explore the dataset given below:
  - First one to understand the number of launch sites used, i.e. list of unique sites
  - To display 5 records where launch sites begin with the string 'CCA'
  - To learn about the total payload mass carried by booster launched by NASA (CRS)
  - To learn about the average payload mass carried by booster version F9 v1.1
  - To know the date when the first successful landing outcome in ground pad was achieved
  - To know about the names of boosters which have success in drone ship and have payload mass between 4000 – 6000
  - To know the total number of successful and failure mission outcomes
  - To list out the names of booster versions which have carried the maximum payload mass
  - To display the months, failure landing outcome in drone ship, booster version name and launch site for year 2015
  - To rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order

GitHub https://github.com/aakash9107/IBM-/blob/main/EDA%20with%20sqllite%20DB.ipynb

Build an Interactive Map with Folium

- First we got the coordinates of all the launch site and for each launch site added a circle object and the launch site name as a popup label.
- We want to show on the map the success/failed launches for each site for that we enhanced the map by adding red marker for every unsuccessful (class=0) launch and green marker object for all the successful (class=1) launches.
- Finally we wanted to find out the distance between a launch site to its proximities, therefore we drew a polyline between a launch site to its closest coastline, city, railway and highway. We used mouse position to find their coordinates on the map.
- Objective behind all this:
  - To understand if launch sites are in proximity to the equator line
  - Whether they are in close proximity to highways, railways, coastline
  - Are the launch sites far from cities

GitHub https://github.com/aakash9107/IBM-/blob/main/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

Build a Dashboard with Plotly Dash

- So basically our goal was to make a dashboard application that contains the following input components
  - A launch site drop down to select a particular launch site and display the interactive graphs.
  - A callback function to render a pie chart showing the success rate
  - A range slider to select Payload
  - A callback function to render a scatter plot
- Insights that we were looking for
  - To know which site has the largest successful launches
  - To know about the site that has the highest launch success rate
  - To understand which payload has the highest success rate
  - To know about which payload has the lowest success rate
  - To know which F9 booster version has the highest launch success rate

https://github.com/aakash9107/IBM-/blob/main/spacex_dash_app%20(2).py

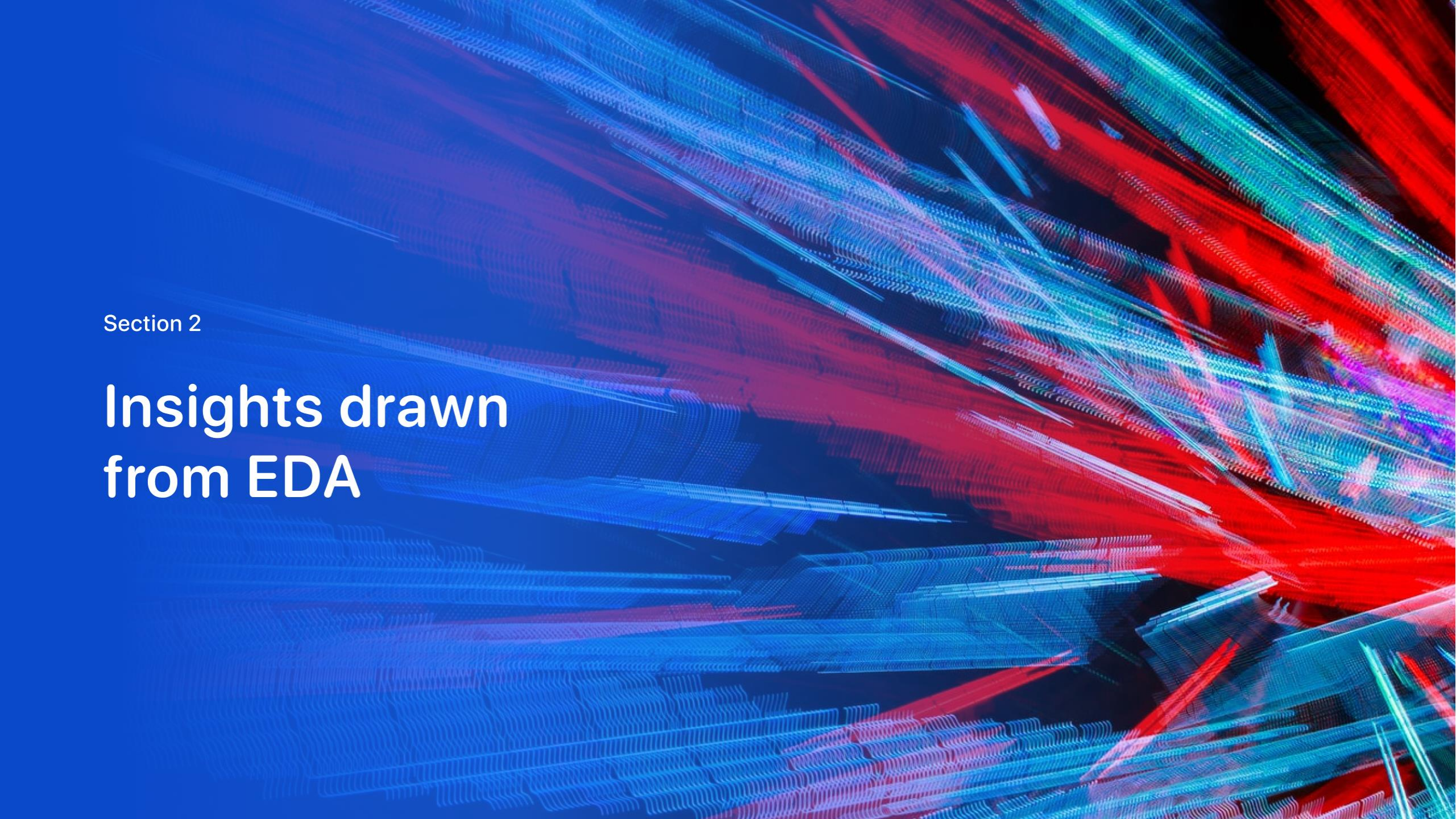
Predictive Analysis (Classification)

- After loading the data we create a NumPy array from the column class and assign it to variable Y
- Then we standardize the data and assign it to X
- We split the data into training set and testing set using the function train_test_split
- The data will be split into 80:20, 80 % training set and 20 % for testing
- Create an object and then create a GridSearchCV object and then find the best parameter for all the classification method
- Plot a confusion matrix
- Finally find the best performing classification model

GitHub https://github.com/aakash9107/IBM-/blob/main/Machine%20Learning%20Prediction%20.ipynb

Results

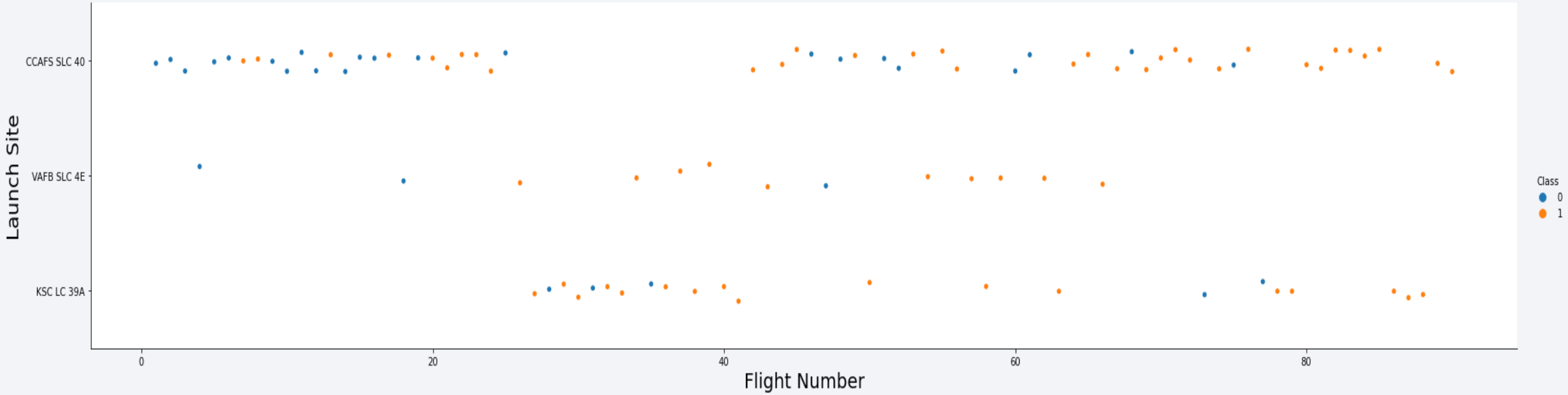
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results



Section 2

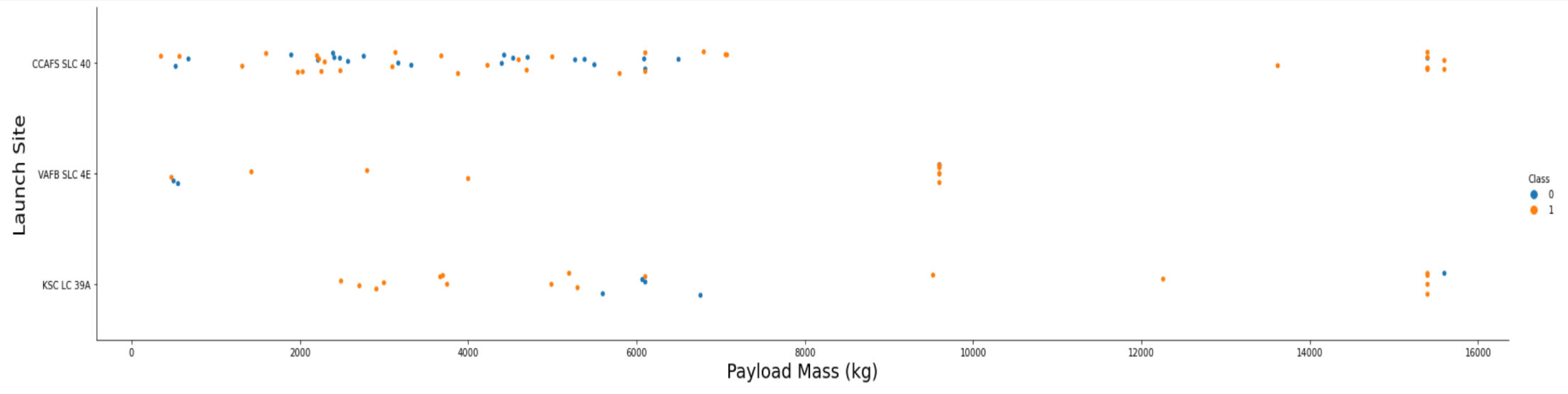
Insights drawn from EDA

Flight Number vs. Launch Site



- If you observe here, we can clearly see that as the flight numbers increase the success rate of the first stage landing also increases.
- Launch site KSC LC 39A has a success rate of 100% when the number of flights are above 78
- Launch site VAFB SLC 40 has the best success rate over all, and when the number of flights are above 55 it's also a 100%
- Launch site CCAFS SLC 40 has a success rate of 100% when flight numbers are above 80

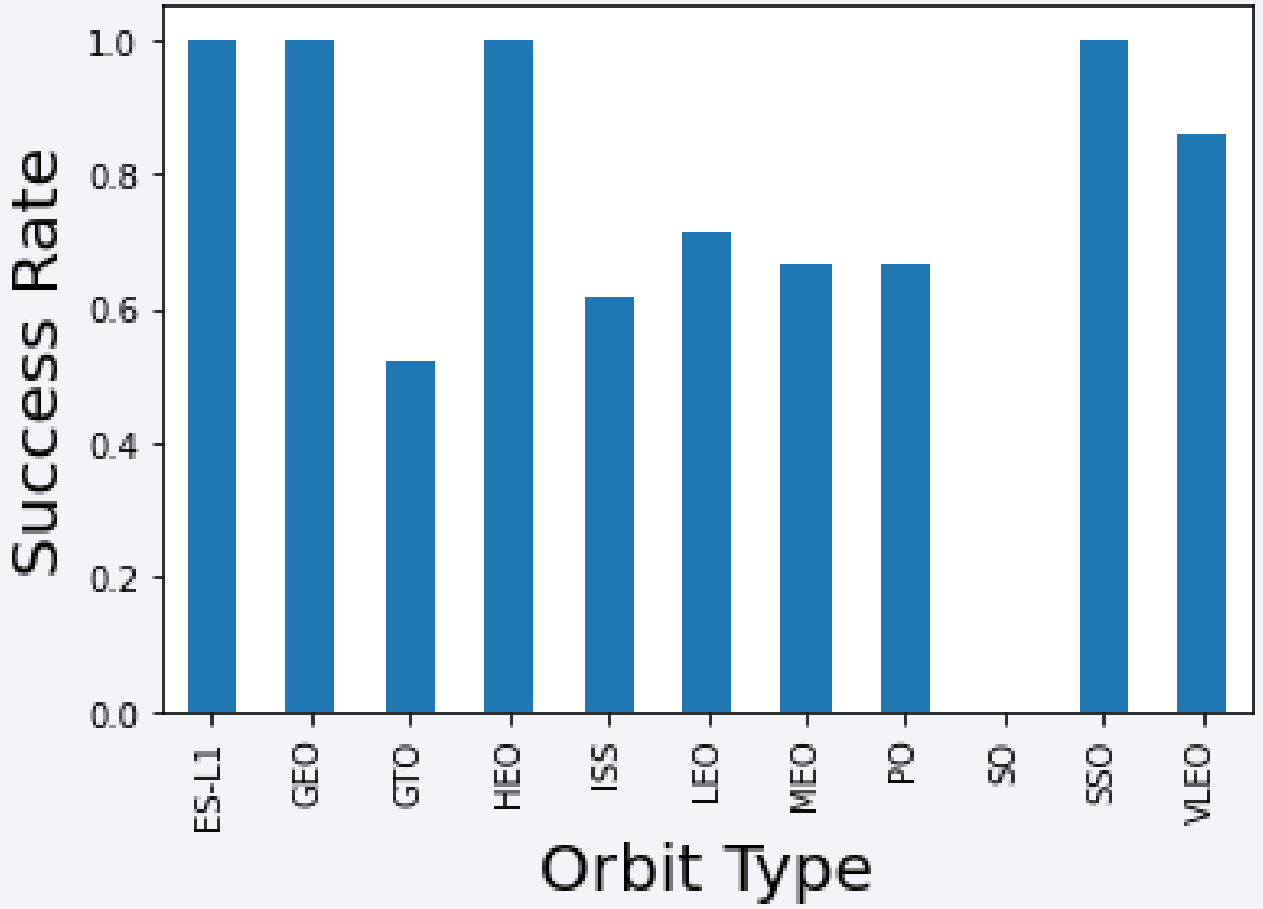
Payload vs. Launch Site



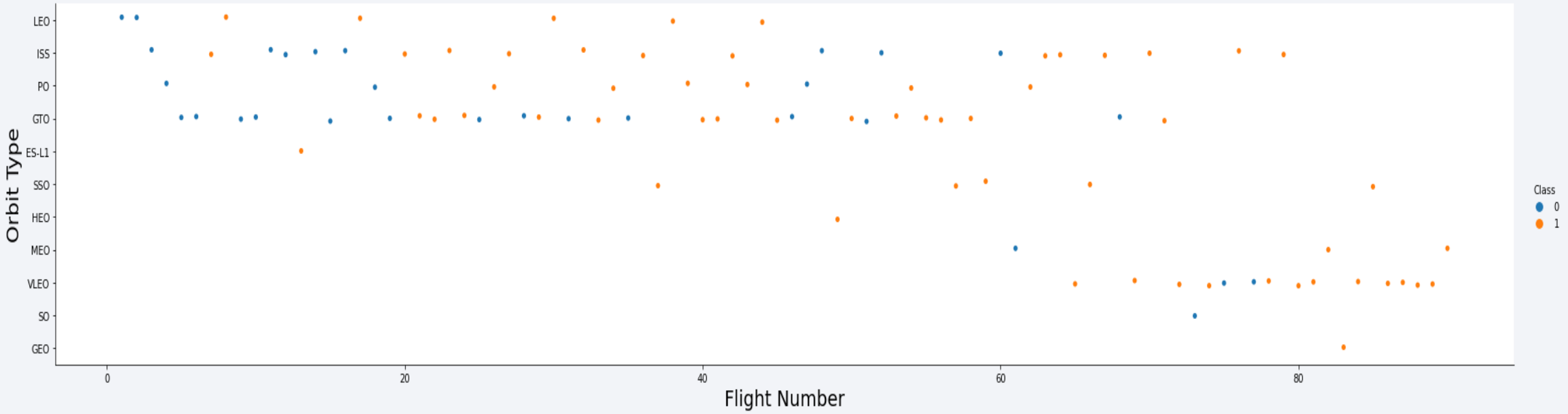
- Now if we observe this scatter point chart we understand that the VAFB SLC 4E launch site has no rockets launched for heavy payload mass (greater than 10,000 kg).
- The launch site CCAFS SLC 40 looks to have low success rate when the payload mass is less, but it's significantly high when the payload mass is above 10,000 kg.

Success Rate vs. Orbit Type

- The following orbits have 100 percent success rate:
  - ES-L1
  - GEO
  - HEO
  - SSO
- We can also see that the orbit SO has a 0% success rate which means that either there any rockets launched in this orbit did not have a safe landing of first stage.
- The Orbit VLEO also has a good success rate of about 90%

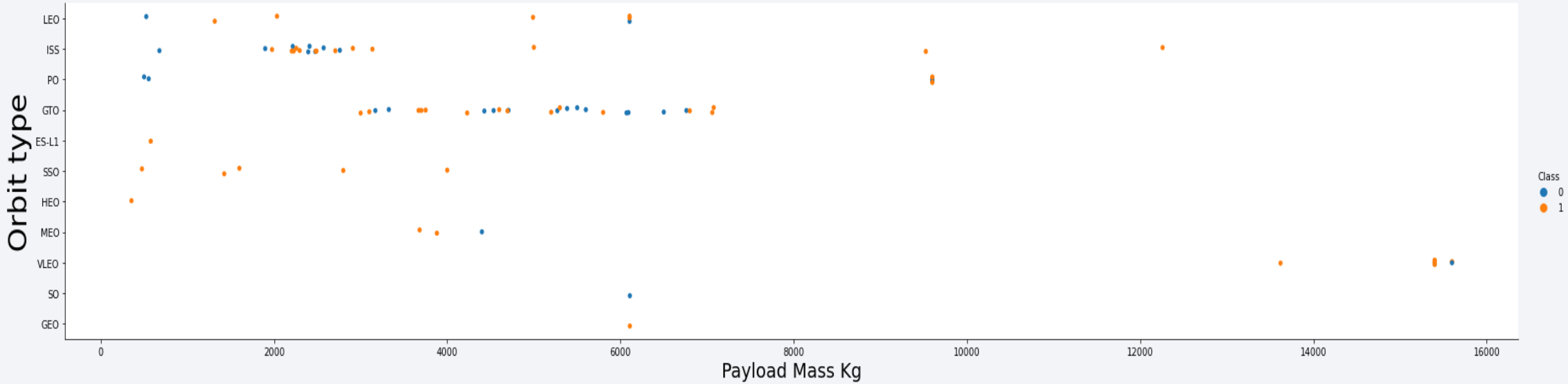


Flight Number vs. Orbit Type



- If we see in LEO orbit the success rate appears related to number of flights
- Similarly in GTO orbit there seems to be no relationship between flight number and success rate
- In VLEO orbit the success rate is 100% when number of flights are more than 78

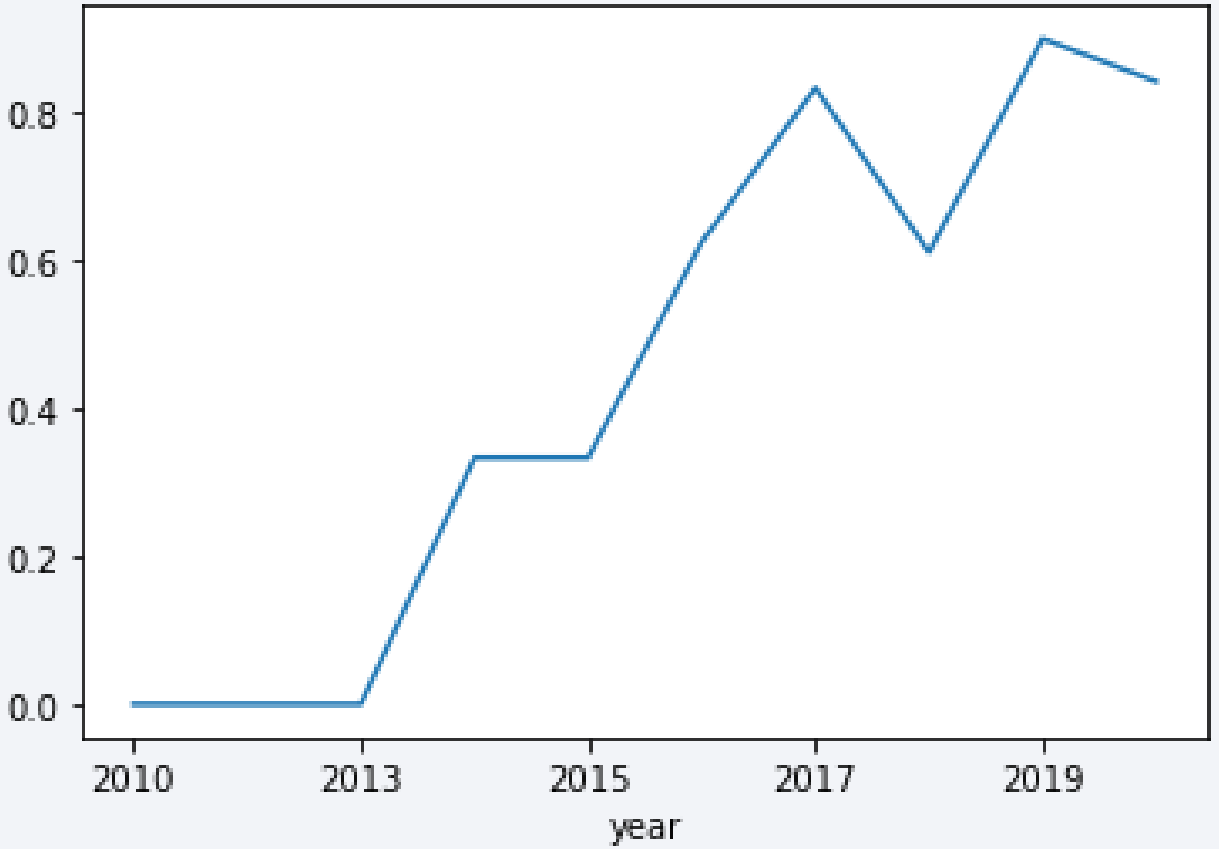
Payload vs. Orbit Type



- The following orbits have more success rate when rockets launched are with high payload:
  - LEO
  - PO
  - ISS
- Similarly when the payload mass is less the above 3 orbits have no successful landing

Launch Success Yearly Trend

In this line chart we can clearly see that the success rate of launches have kept increasing since 2013 till 2020.



All Launch Site Names

- With the motive to list out all the unique launch sites which were used, we perform the SQL query using DISTINCT function.
- This function filters out all the unique values from the Launch_Site column in the SpaceX table.
- After the query is successfully performed we get total of 4 unique launch sites.

```
In [9]: %sql SELECT DISTINCT Launch_Site from SPACEXTBL;
 * sqlite:///my_data1.db
Done.
```

Out[9]:

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

Launch Site Names Begin with 'CCA'

- Now we want to list out all the records from SpaceX table where launch site's name begins with CAA
  - For this SQL query we use the WHERE clause with the like predicate.
  - This like predicate searches for the CCA pattern in the Launch_Site column.
  - The percentage is used to define the missing letters .

```
In [37]: %sql SELECT * FROM SPACEXTBL WHERE Launch_Site like 'CCA%' LIMIT 5;
 * sqlite:///my_data1.db
Done.
```

Out[37]:

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

Total Payload Mass NASA (CRS)

```
In [20]: %sql SELECT Customer, SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass FROM SPACEXTBL WHERE CUSTOMER LIKE 'NASA (CRS)%';
 * sqlite:///my_data1.db
Done.
```

Out[20]:

Customer	Total_Payload_Mass
NASA (CRS)	48213

- Here, to figure out total the total payload mass for customer NASA (CRS) we use the built in SQL function SUM().
  - This function helps us add all the values in the column.
  - Since we only need to find out the total payload mass for NASA(CRS) we include a like predicate in the WHERE clause.
  - We finally have the total payload mass for the customer NASA(CRS) and we have also specified the column name.

Average Payload Mass by F9 v1.1

```
In [22]: %sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD_MASS FROM SPACEXTBL WHERE Booster_Version LIKE 'F9 v1.1%';
 * sqlite:///my_data1.db
Done.
```

Out[22]:

AVG_PAYLOAD_MASS
2534.6666666666665

- To find the average payload mass by F9 v1.1, we use the built in database function called AVG()
  - It basically returns the average of all the values in a column
  - Now since we only want to know the average of Booster version F9 v1.1, we would specify that in the WHERE clause.
  - The result gives us the average as we specify the column name as AVG_PAYLOAD_MASS

First Successful Ground Landing Date

In [66]:
```
#%sql SELECT * FROM SPACEXTBL WHERE Mission_Outcome LIKE 'Success%';
%sql SELECT "Landing _Outcome", MIN(Date) AS First_Successful_Landing FROM SPACEXTBL WHERE "Landing _Outcome" = 'Success (ground pad)';
```

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

Out[66]:

Landing _Outcome	First_Successful_Landing
Success (ground pad)	01-05-2017

- We would like to know the date when first successful landing on a ground pad was achieved
  - For this we use the min() function to ensure that we get the minimum value from the date column, i.e. the first date.
  - In the WHERE clause we would specify that the query should only be performed where Landing_Outcome is Success (ground pad)

Successful Drone Ship Landing with Payload between 4000 and 6000

In [68]:
```
%sql SELECT Booster_Version, "Landing _Outcome" FROM SPACEXTBL WHERE (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000) AND "Landing _Out
come" = 'Success (drone ship)';
```

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

Out[68]:

Booster_Version	Landing _Outcome
F9 FT B1022	Success (drone ship)
F9 FT B1026	Success (drone ship)
F9 FT B1021.2	Success (drone ship)
F9 FT B1031.2	Success (drone ship)

- To find out the successful Drone ship landing with payload between 4000 and 6000 we use the range function in the WHERE clause.
- We also use AND clause in the query to filter out the result restricting it to Successful drone ship landing

Total Number of Successful and Failure Mission Outcomes

```
In [76]: %sql SELECT Mission_Outcome as "Mission Outcome", count(*) as Frequency FROM SPACEXTBL GROUP BY Mission_Outcome;
 * sqlite:///my_data1.db
Done.
```

Out[76]:

Mission Outcome	Frequency
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

- We figure out the total number of successful and failure mission outcomes by adding count(*) to add the values
- We use the GROUP BY clause in the SELECT statement to group the result into subset that has matching values

Boosters Carried Maximum Payload

- To know about the boosters carrying the maximum payload we have to use a sub query.
- Basically what it does is that the second SELECT statement finds out the highest values
- The outer query uses the result of the sub query as the data source and gives us the answer.

```
In [84]: %sql SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
 * sqlite:///my_data1.db
Done.
```

Out[84]:

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

Failed landing outcome in drone ship for year 2015

```
In [92]: %sql SELECT substr(Date, 4,2) AS Month, Booster_Version, Launch_Site, "Landing _Outcome" FROM SPACEXTBL WHERE "Landing _Outcome"
         = 'Failure (drone ship)' AND substr(Date,7,4)= '2015';

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

Out[92]:

Month	Booster_Version	Launch_Site	Landing _Outcome
01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

- To display the month names, failed landing outcomes in drone ship, booster version, launch site for the months in year 2015:
  - We basically start with a select statement to include all the data we are looking for
  - Then in the WHERE clause we restrict the result to look for only failed drone ship landing
  - We use an AND clause to filter the year
  - We use substr(Date,4,2) as month to get the months and substr(Date,7,4)='2015' for year because SQLLite does not support monthnames

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

```
In [100]: %sql SELECT "Landing _Outcome" as Landing_Outcome, COUNT(*) AS Frequency FROM SPACEXTBL WHERE "Landing _Outcome" like 'Success%'
          AND DATE BETWEEN '04-06-2010' AND '20-03-2017' GROUP BY "Landing _Outcome" ORDER BY Frequency desc;

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

Out[100]:

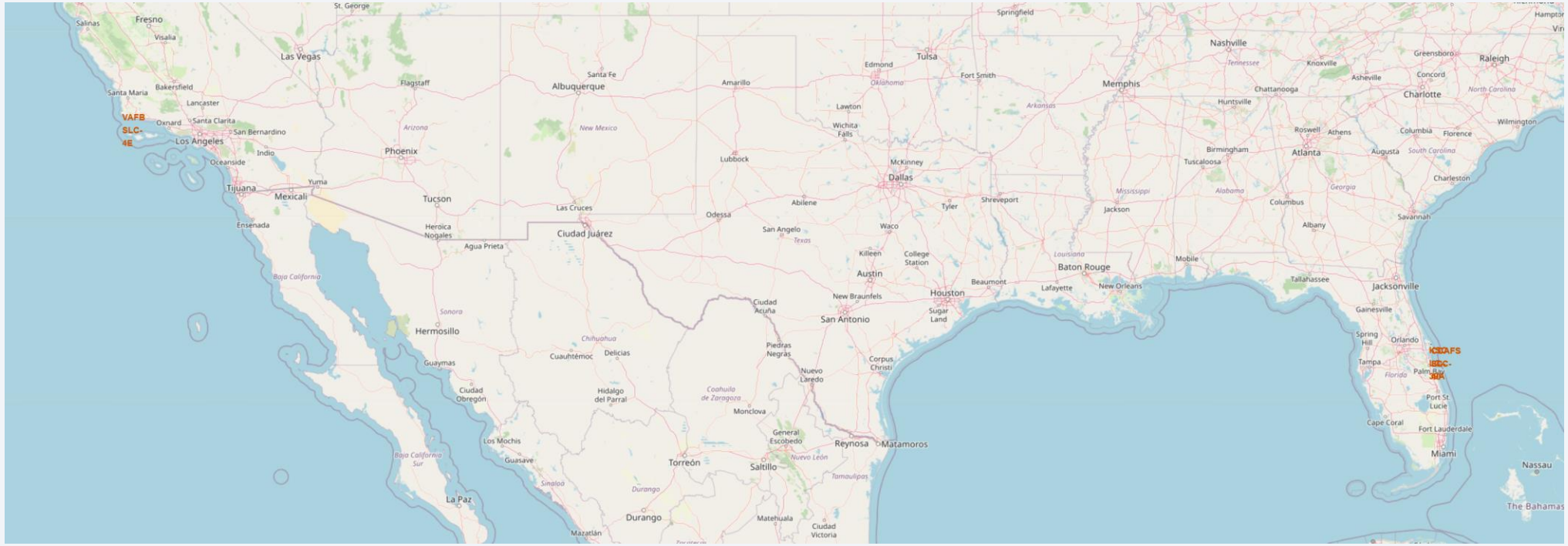
Landing_Outcome	Frequency
Success	20
Success (drone ship)	8
Success (ground pad)	6

- To rank all the successful landings between 2010-06-04 and 2017-03-20:
  - We start our SELECT statement with what we want to find out
  - In the where clause we include a like predicate
  - AND statement is used to define the range as mentioned
  - Group by clause is used to group the result into subsets that has matching values
  - Finally ORDER BY clause is used to ensure the result is in the descending order hence providing us the ranking

Section 4

Launch Sites Proximities Analysis

Marker of all the launch sites on a Global Map



- We do have the coordinates of the launch sites in our data set but one has to be extremely good at geography to gain any kind of insights from that data because they are just plain numbers
- Therefore, to get some insights we visualize the launch sites on a map using Folium
- Here after plotting the launch sites on a map we can clearly see that all of them are extremely close to coastal side and away from the city

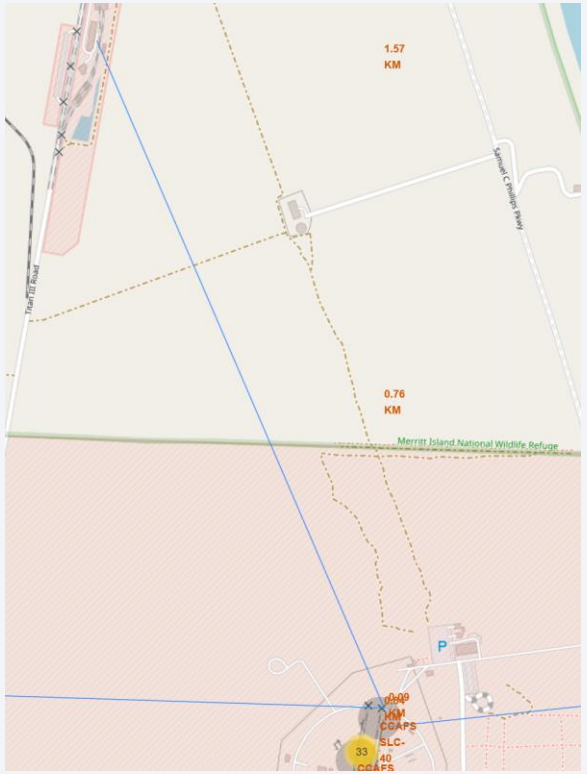
Marking the success/failed launches for each sites

- Now to gain better insights we enhance the map by adding the launch outcome for each site.
- We do that by creating a green marker for every successful landing (class=1), and a red marker for every unsuccessful landing (class=0)
- As we can clearly see, KSA LC-39A has the highest successful landing of stage 1 rocket

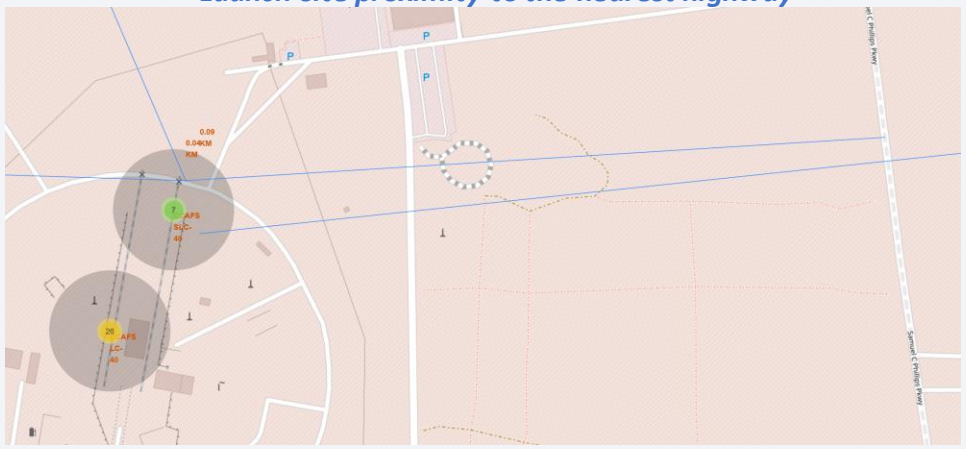
Launch site's distance to its proximities

Launch site proximity to the nearest coastline

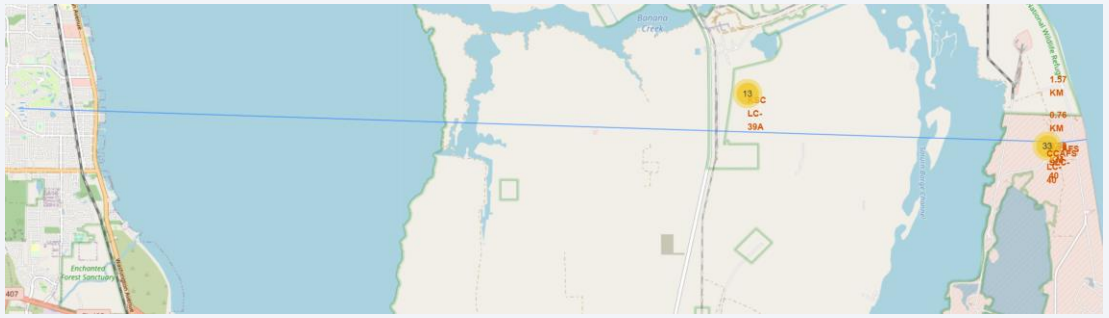
Launch site proximity to the nearest railway station



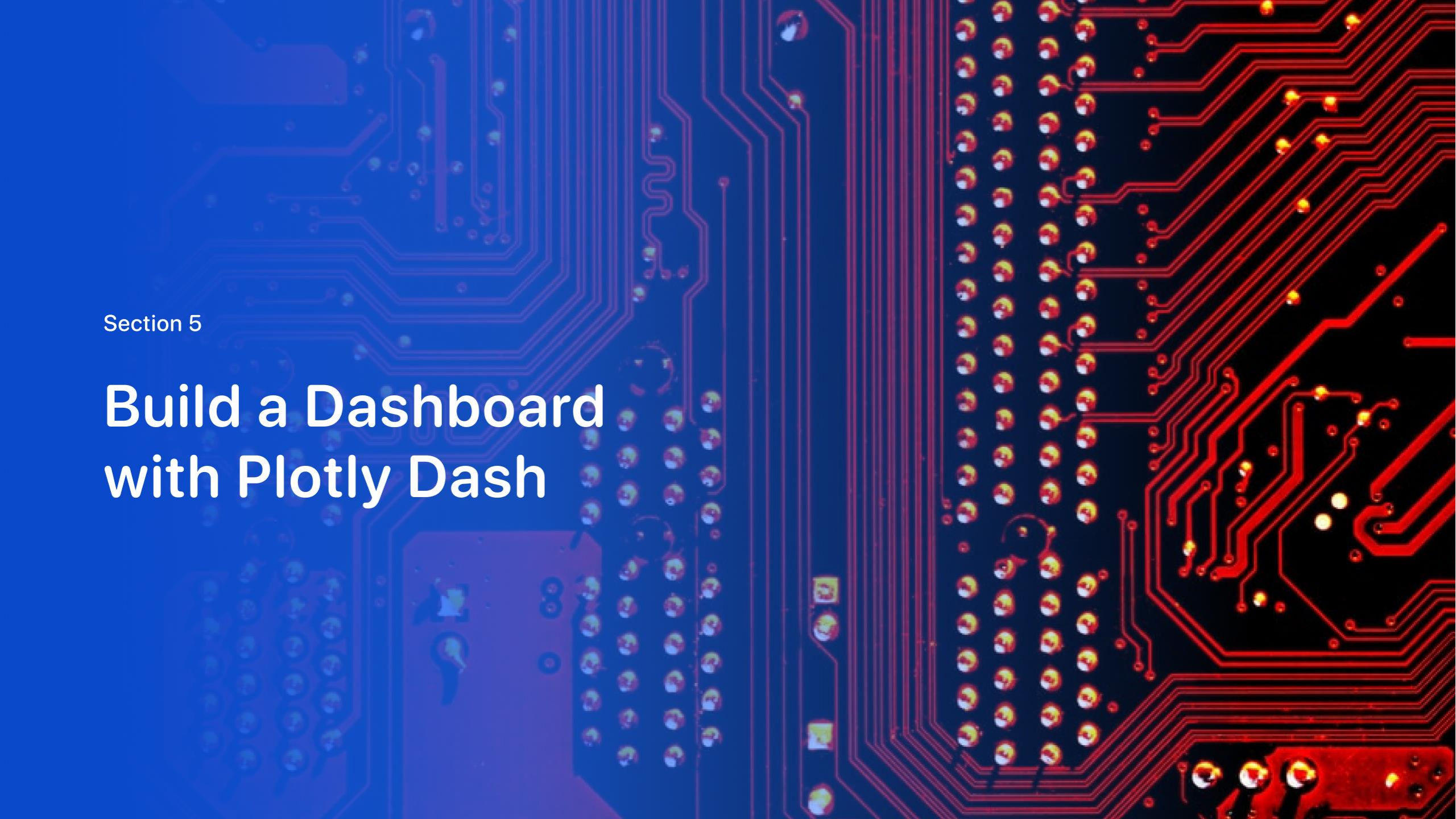
Launch site proximity to the nearest highway



Launch site proximity to the nearest City



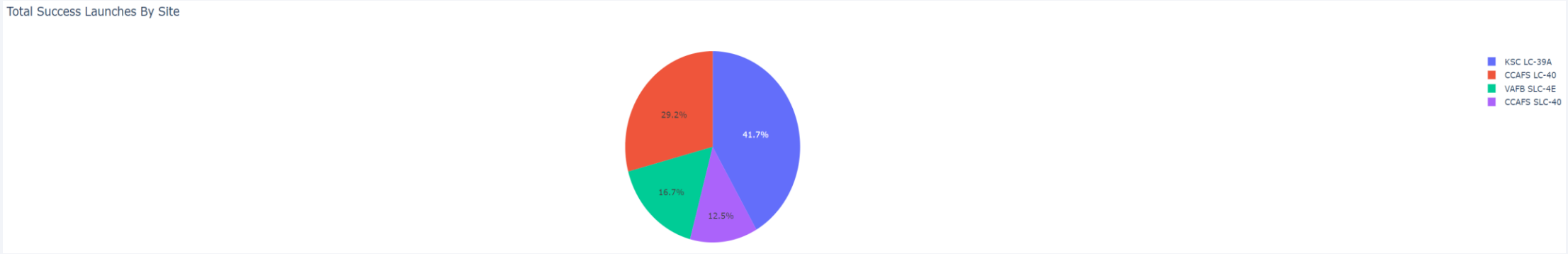
- We can clearly observe that all the launch sites are away from the city
- The launch sites are in close proximity to highways
- The launch sites are deliberately chosen to be far from areas where there is large population



Section 5

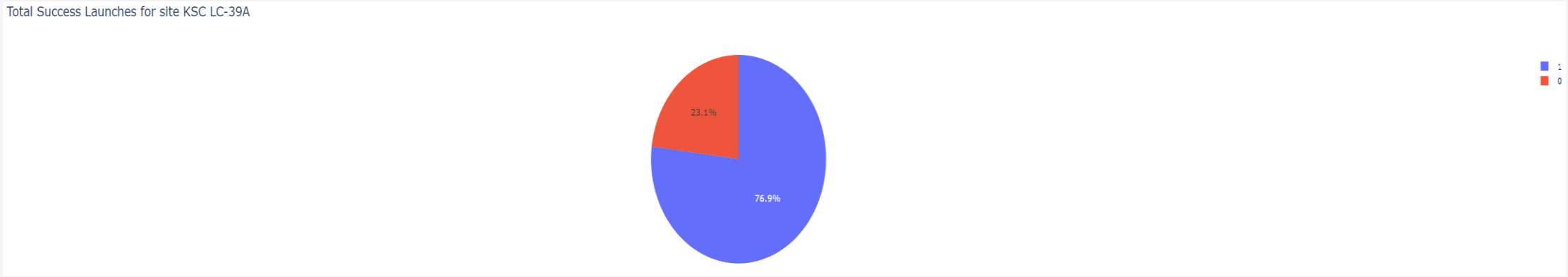
Build a Dashboard with Plotly Dash

Successful launches from all the sites



- Here we see the percentage of successful launches from all the launch sites
- We can clearly understand that launch site KSC LC-39A has the highest percentage of overall successful launches.
- Whereas CCAFS SLC-40 has just 12.5% of successful launches

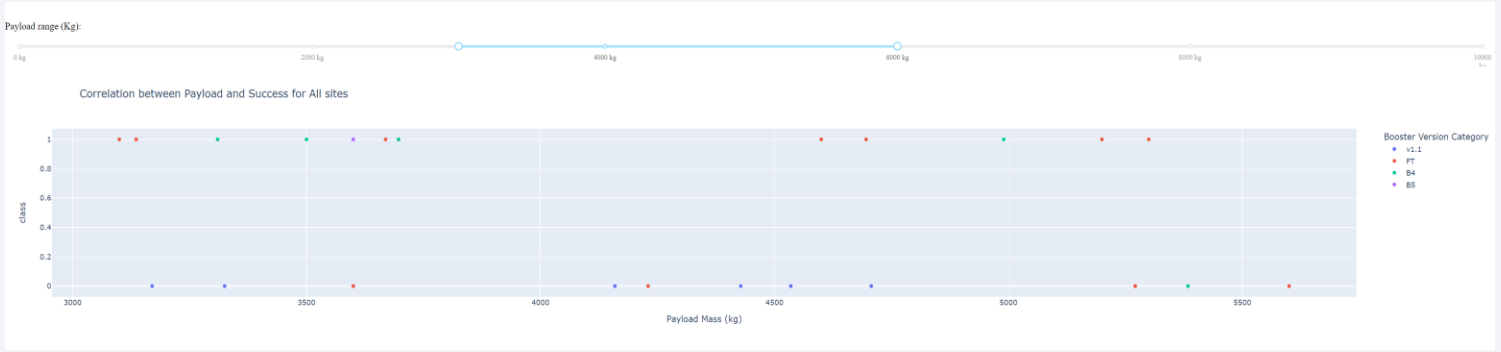
Pie chart of highest landing success ratio



- KSC LC-39A has the highest success ratio whenever a rocket is launched from this site
- It has a success rate of 76.9%
- Therefore, this launch site gives us the maximum success

Payload vs Launch Outcome scatter plot for all sites with different payload selected

- In the first screenshot we can see that the payload selected between 0-3000 kgs



- The second screenshor shows us the success rate for payload mass between 3000kgs to 6000 kgs

- The final screenshot shows us the success rate of first stage landing of rockets that bear the mass above 6000 kgs

Section 6

Predictive Analysis (Classification)

Classification Accuracy on testing data

- The best classification model which has the highest accuracy is Decision tree.
- The accuracy score is 0.93035
- This was achieved on the test data and the model was prepared on the training data.

In [31]:
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('The Best Algorithm from the performed algorithms is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
The Best Algorithm from the performed algorithms is Tree with a score of 0.9303571428571429
Best Params is : {'criterion': 'entropy', 'max_depth': 16, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split':
10, 'splitter': 'best'}
```

Confusion Matrix

Confusion matrix for Decision Tree:

- We can see that the model has good accuracy of predicting the failed landings.
- We can also see that the landing predictions are also quite accurate.
- There are times when the model predicted that the first stage would not land which falls under the false positive category which means that it did land.

Conclusions

- The decision tree is the best performing classification model.
- KSC LC-39A has the highest success ratio of 77% whenever a rocket is launched from this site
- The launch sites are always located away from cities and closest to the coastlines
- The mission outcome is 99% successful
- The average payload mass carried by rockets is 2534.66
- The success rate of launches have kept increasing since 2013 till 2020.
- Orbits LEO, PO and ISS have more success rate when rockets have high payload mass
- The VAFB SLC 4E launch site has no rockets launched for heavy payload mass (greater than 10,000 kg).

Appendix

- GitHub URL of all the Python notebooks I created during this project
  - Data collection API: https://github.com/aakash9107/IBM-/blob/main/Data%20Collection%20API%20%20.ipynb
  - Data collection web-scraping: https://github.com/aakash9107/IBM-/blob/main/Data%20Collection%20with%20Web%20Scraping.ipynb
  - Data Wrangling: https://github.com/aakash9107/IBM-/blob/main/Data%20Wrangling.ipynb
  - EDA Visualization: https://github.com/aakash9107/IBM-/blob/main/EDA%20with%20Data%20Visualization%20.ipynb
  - EDA SQLLite: https://github.com/aakash9107/IBM-/blob/main/EDA%20with%20sqllite%20DB.ipynb
  - Folium: https://github.com/aakash9107/IBM-/blob/main/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb
  - Dash app: https://github.com/aakash9107/IBM-/blob/main/spacex_dash_app%20(2).py
  - ML Prediction: https://github.com/aakash9107/IBM-/blob/main/Machine%20Learning%20Prediction%20.ipynb

https://github.com/aakash9107

Thank you!