IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Gloria SD
July 27, 2024

Outline

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

Executive Summary

Summary of methodologies

We will be collecting data from sources available to the public. After the raw data has been collected, we will improve the quality by performing data wrangling. Then we start exploring the processed data. We will use SQL to query the data and gather insights. To gain further insights into the data, we apply some basic statistical analysis and data visualization. We will try to see directly how variables might be related to each other. We will drill down into finer levels of detail by splitting the data into groups defined by categorical variables or factors in the data. We will build, evaluate, and refine predictive models for discovering more insights.

Executive Summary

Summary of all results

After collecting the data using APIs and Web scraping, we performed data wrangling to improve the quality of the data. We used SQL to query the data and gather insights. Also we used some basic statistical analysis and data visualization to gain further insights on the data and we discovered that flight number, payload mass, launch sites, and orbit type could affect the launch outcome success. The success rate since 2013 kept increasing till 2020. Also, launch sites are usually near the equator. And they are in very close proximity to coastlines.

We developed, evaluated, and refined predictive models for discovering more insights. We use the confusion matrix to test the accuracy of different models. And among the classification models tested, the decision tree model has the highest classification accuracy. And because of the high accuracy of the predictive model we developed and tested, we will be able to predict with confidence if the Falcon 9 first stage will land successfully.

Introduction

The commercial space age is here and companies are making space travel affordable for everyone. Perhaps the most successful company today is SpaceX. One reason SpaceX can do various space travels is the rocket launches are relatively inexpensive. 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 SPACE Y wants to bid against SpaceX for a rocket launch.

In this project we will predict if the Falcon 9 first stage will land successfully.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

  We will be working with SpaceX launch data that is gathered from an API, specifically the SpaceX REST API. This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.

  Another source for obtaining Falcon 9 Launch data is web scraping related Wiki pages. We will be using the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records. Then we need to parse the data from those tables and convert them into a Pandas data frame for further visualization and analysis.

- Perform data wrangling

  We will transform the raw data into a clean dataset which provides meaningful data on the situation we are trying to address: Wrangling Data using an API, Sampling Data, and Dealing with Nulls.

Data Collection

We will be working with SpaceX launch data that is gathered from an API, specifically the SpaceX REST API. This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome..

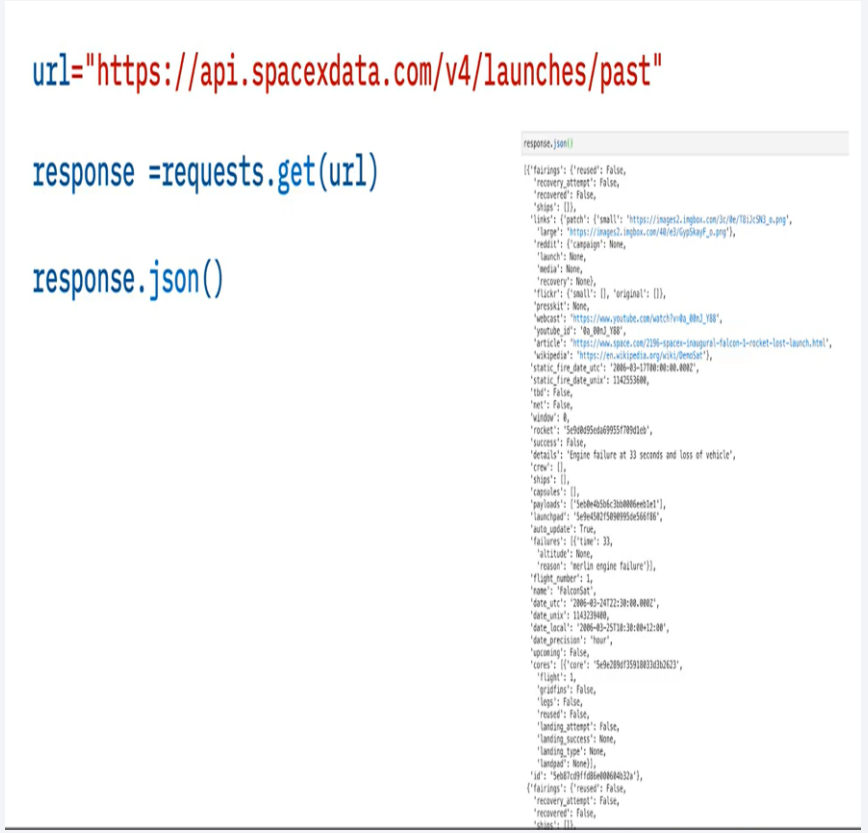
We will also use the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records.

Data Collection – SpaceX API

We will use a URL to target a specific endpoint of the API to get past launch data. We will perform a get request using the requests library to obtain the launch data, which we will use to get the data from the API. This result can be viewed by calling the .json()method. Our response will be in the form of a JSON, specifically a list of JSON objects.

For details, please refer to the following:

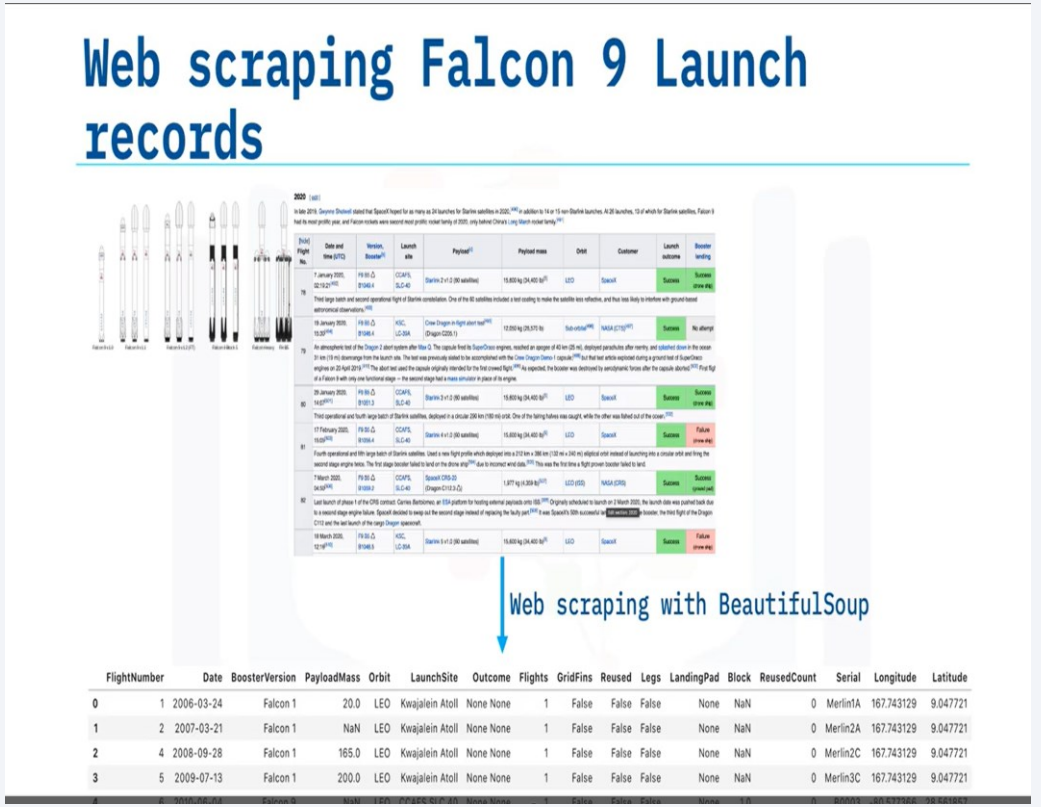
https://github.com/euroamerasia/applied_capstone_gsd/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

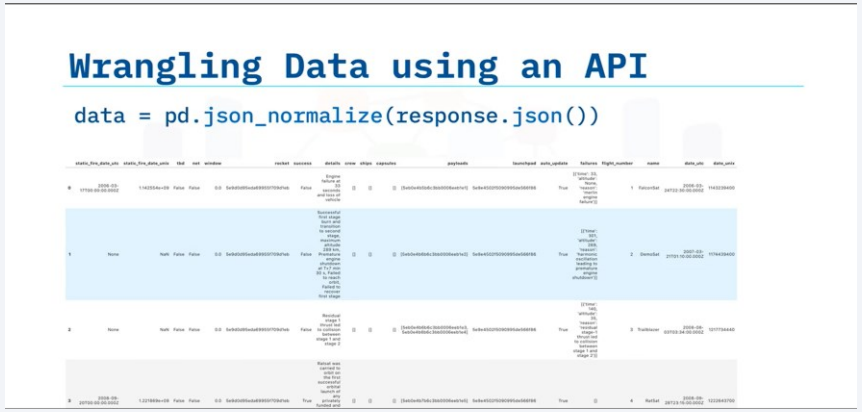
We will be using the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records. Then we need to parse the data from those tables and convert them into a Pandas data frame for further visualization and analysis

For details, please refer to the following:
https://github.com/euroamerasia/applied_capstone_gsd/blob/main/jupyter-labs-webscraping.ipynb

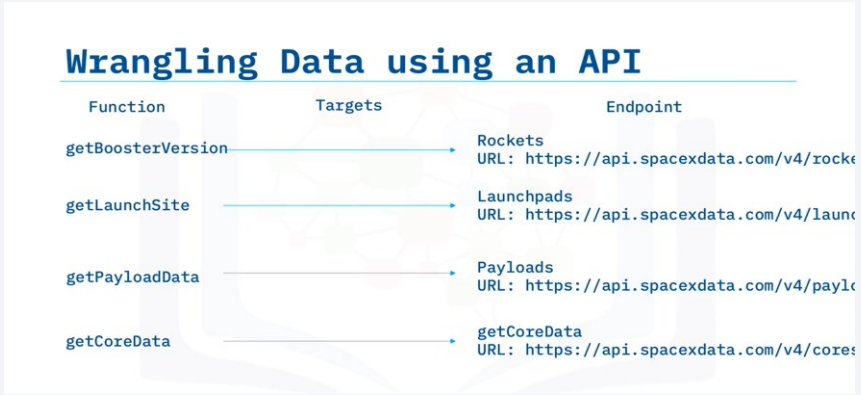


Data Wrangling

Since we are using an API, when we get a response it is in the form of a JSON. Specifically, we have a list of JSON objects which each represent a launch. To convert this JSON to a dataframe, we can use the json_normalize function. This function will allow us to “normalize”the structured json data into a flat table. This is what your JSON will look like in a table form.



In some of the columns, like rocket, we have an identification number, not actual data. This means we will need to use the API again targeting another endpoint to gather specific data for each ID number. We created the corresponding functions to get Booster, Launchpad, payload, and core data. The data will be stored in lists and will be used to create our dataset.



Data Wrangling

Another issue we have is that the launch data we have includes data for the Falcon 1 booster whereas we only want falcon 9, so we need to filter/sample the data to remove Falcon 1 launches.

Finally, we may end up with data that contains NULL values. We must sometimes deal with these null values in order to make the dataset viable for analysis. In this case we will deal with the NULL values by calculating the mean of the data and then replace the null values in column with the mean.

For details, please refer to the following:

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

EDA with Data Visualization

To visually see the relationship of variables against each other and how they would affect the launch outcome, we plot the following charts:

- FlightNumber vs. PayloadMass: A scatter plot chart to see the relationship of flight number and payload mass .
- FlightNumber vs LaunchSite: A scatter plot chart to see the relationship between Flight Number and Launch Site.
- Payload vs. Launchsite: a scatter plot to see the relationship of Payload and Launch site
- Orbit vs Success Rate: A bar chart to see the relationship between Orbit type and Success Rate
- FlightNumber vs Orbit Type: A scatter plot chart to see the relationship between Flight Number and Orbit type.
- Payload vs Orbit: A scatter plot chart to see relationship between Payload and Orbit type.
- Launch Success Yearly: A line chart to see if there is a pattern of success over the years.

For details, please refer to the following: https://github.com/euroamerasia/applied_capstone_gsd/blob/main/edadataviz.ipynb

EDA with SQL

Summary of SQL queries performed: (Part 1)

1. Display the names of the unique launch sites in the space mission: ***%sql select distinct "Launch_Site" from SPACEXTABLE***

2. Display 5 records where launch sites begin with the string 'CCA': ***%sql select * from SPACEXTABLE where "Launch_Site" like 'CCA%' limit 5***

3. Display the total payload mass carried by boosters launched by NASA (CRS): ***%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where "Customer"='NASA (CRS)'***

4. Display average payload mass carried by booster version F9 v1.1: ***%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTABLE where "Booster_Version" = 'F9 v1.1'***

5. List the date when the first succesfull landing outcome in ground pad was achieved.: ***%sql select min(Date) from SPACEXTABLE where "Landing_Outcome" = 'Success (ground pad)'***

6. List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000: ***%sql select "Booster_Version", PAYLOAD_MASS__KG_ from SPACEXTABLE where "Landing_Outcome" = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000***

EDA with SQL

Summary of SQL queries performed: (Part 2)

7. List the total number of successful and failure mission outcomes: ***%sql select count(substr("Mission_Outcome",0,8)) as outcome_count, substr("Mission_Outcome",0,8) as outcome from SPACEXTABLE group by outcome***

8. List the names of the booster_versions which have carried the maximum payload mass: ***%sql select "Booster_Version" , PAYLOAD_MASS__K***G_ from SPACEXTABLE where PAYLOAD_MASS__KG_=(select MAX(PAYLOAD_MASS__KG_) from SPACEXTABLE)

9. 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: ***%sql select substr("Date",0,5) as yearint, substr("Date", 6,2) as monthint,"Landing_Outcome", "Booster_Version", "Launch_Site" from SPACEXTABLE where "Landing_Outcome"='Failure (drone ship)' and substr("Date",0,5)='2015'***

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: ***%sql select "Landing_Outcome", count("Landing_Outcome") as outcome_count from SPACEXTABLE where "Date" between '2010-06-04' and '2017-03-20' group by "Landing_Outcome" order by outcome_count desc***

For details, please refer to the following: https://github.com/euroamerasia/applied_capstone_gsd/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

In this section, we will be performing more interactive visual analytics on the launch sites using Folium.

First, we mark all launch sites on a map. We create a folium map object, and add highlighted circle area with a text label on a specific coordinate. By exploring the map by zoom-in/out the marked areas, we will notice that all the launch sites are near the equator. Also they are in close proximity to coastlines.

Next, we try to enhance the map by adding the launch outcomes for each site, and see which sites have high success rates. We create markers for all launch records. If a launch was successful (class=1), then we use a green marker and if a launch was failed, we use a red marker (class=0). Note that a launch only happens in one of the four launch sites, which means many launch records will have the exact same coordinate. Marker clusters can be a good way to simplify a map containing many markers having the same coordinate.

Lastly, we calculate the distances between a launch site to its proximities(coastline, highway, railroad, etc.). To do this, we first add MousePosition on the map to get coordinate for a mouse over a point on the map. As such, while we are exploring the map, you can easily find the coordinates of any points of interests (such as coastline). Next, we Draw a PolyLine between a launch site to the selected coastline point

For details, please refer to the following:
https://github.com/euroamerasia/applied_capstone_gsd/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

We will be building a Plotly Dash application for users to perform interactive visual analytics on SpaceX launch data in real-time.

This dashboard application contains input components such as a dropdown list and a range slider to interact with a pie chart and a scatter point chart.

First, we add a Launch Site Drop-down Input Component

Second, we add a callback function to render success-pie-chart based on selected site dropdown

Third, we add a Range Slider to Select Payload

Fourth, we add a callback function to render the success-payload-scatter-chart scatter plot

For details, please refer to the following:
https://github.com/euroamerasia/applied_capstone_gsd/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

We will build a machine learning pipeline to predict if the first stage of the Falcon 9 lands successfully.

This will include: Preprocessing, allowing us to standardize our data, and Train_test_split, allowing us to split our data into training and testing data.

We will train the model and perform Grid Search, allowing us to find the hyperparameters that allow a given algorithm to perform best. Using the best hyperparameter values, we will determine the model with the best accuracy using the training data. We will use Logistic Regression, Support Vector machines, Decision Tree Classifier, and K-nearest neighbors.

Finally, we will output the confusion matrix.

For details, please refer to the following:
https://github.com/euroamerasia/applied_capstone_gsd/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

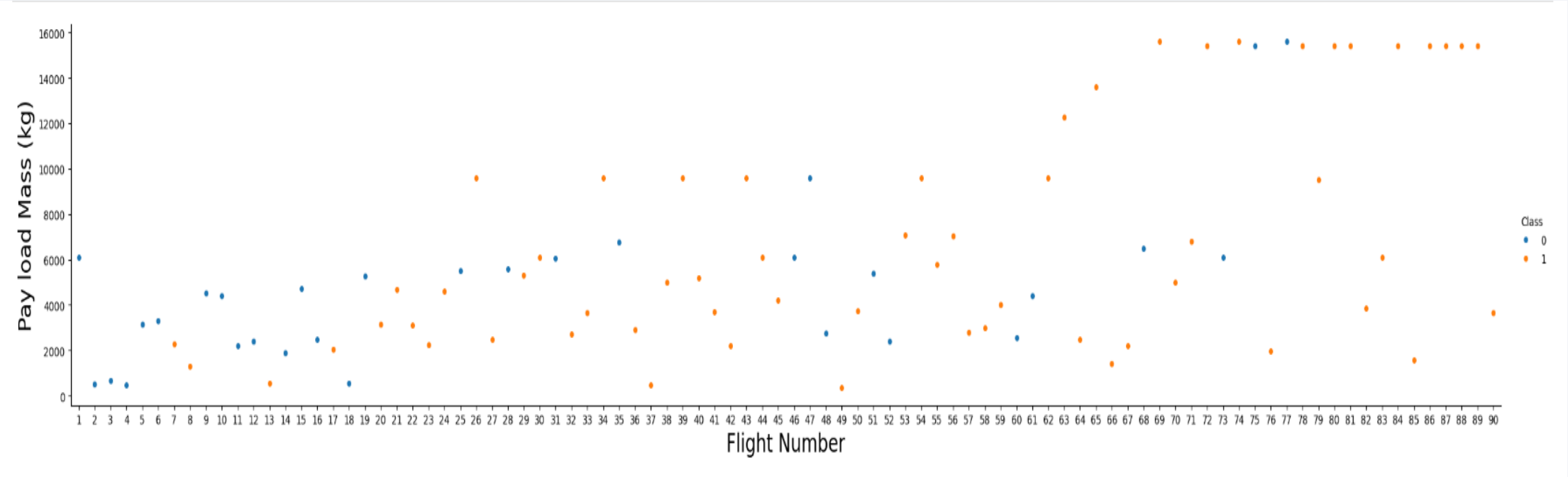
Results

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

Section 2

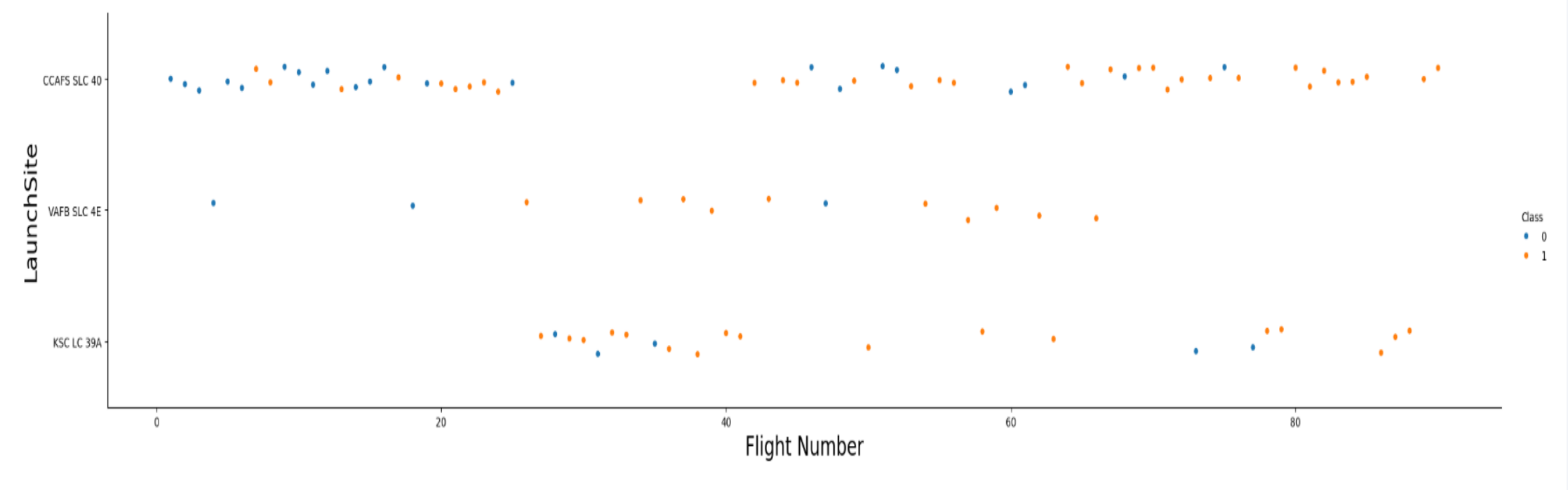
Insights drawn from EDA

Flight Number vs. Payload



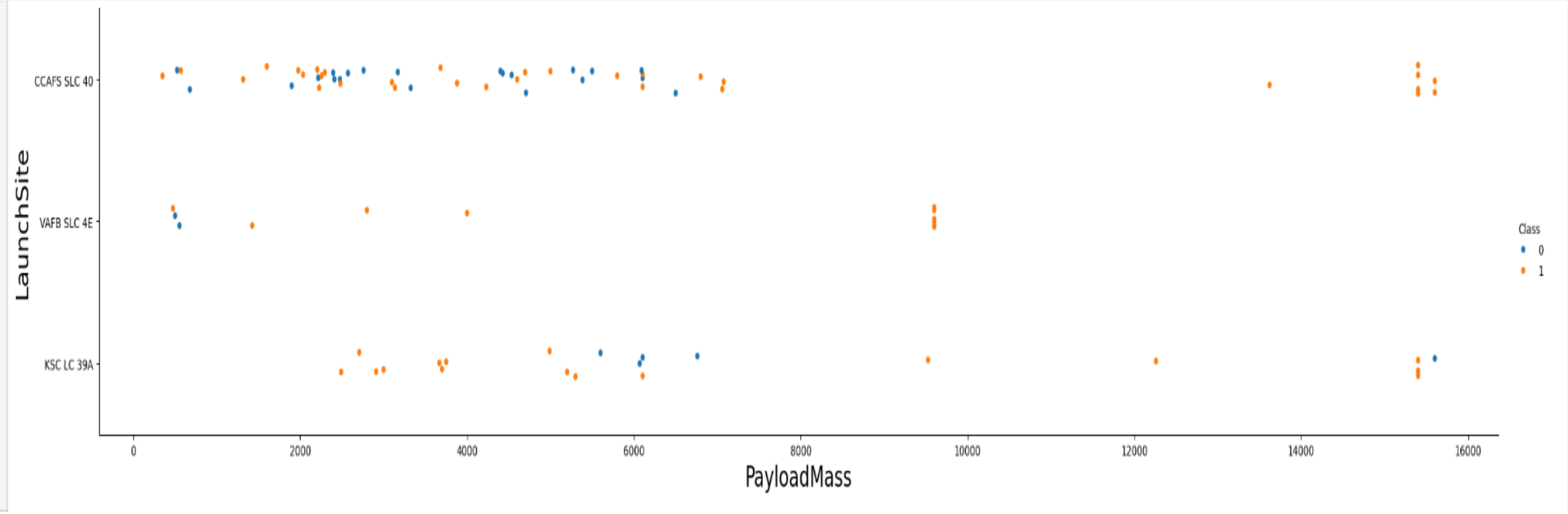
We see that as the flight number increases, the first stage is more likely to land successfully. The payload mass is also important; it seems the more massive the payload, the less likely the first stage will return.

Flight Number vs. Launch Site



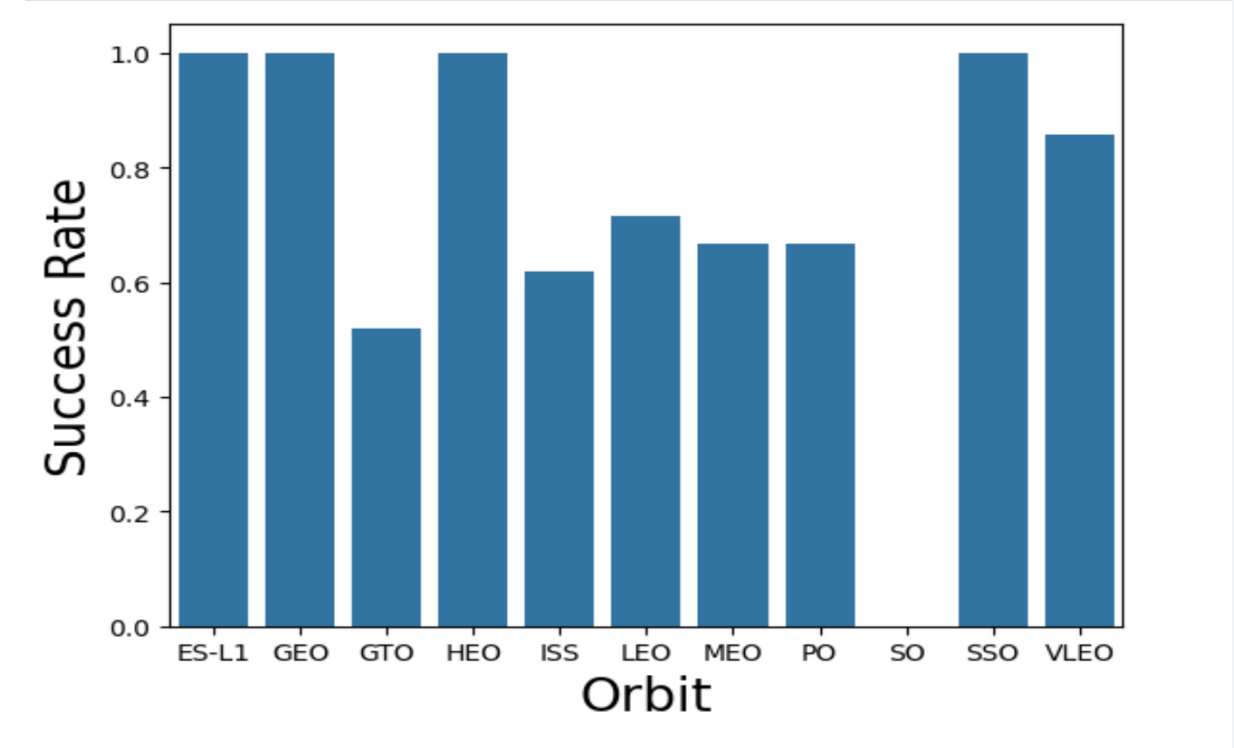
We see that different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E have a success rate of 77%.

Payload vs. Launch Site



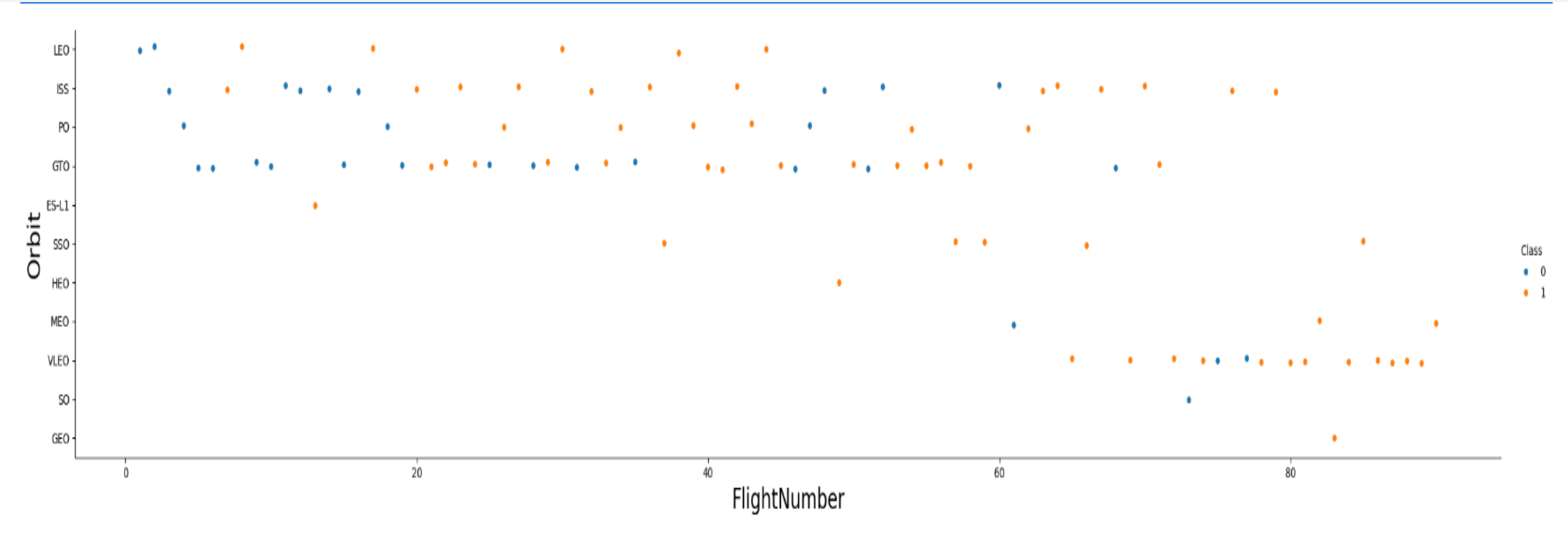
Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).

Orbit Type vs. Success Rate



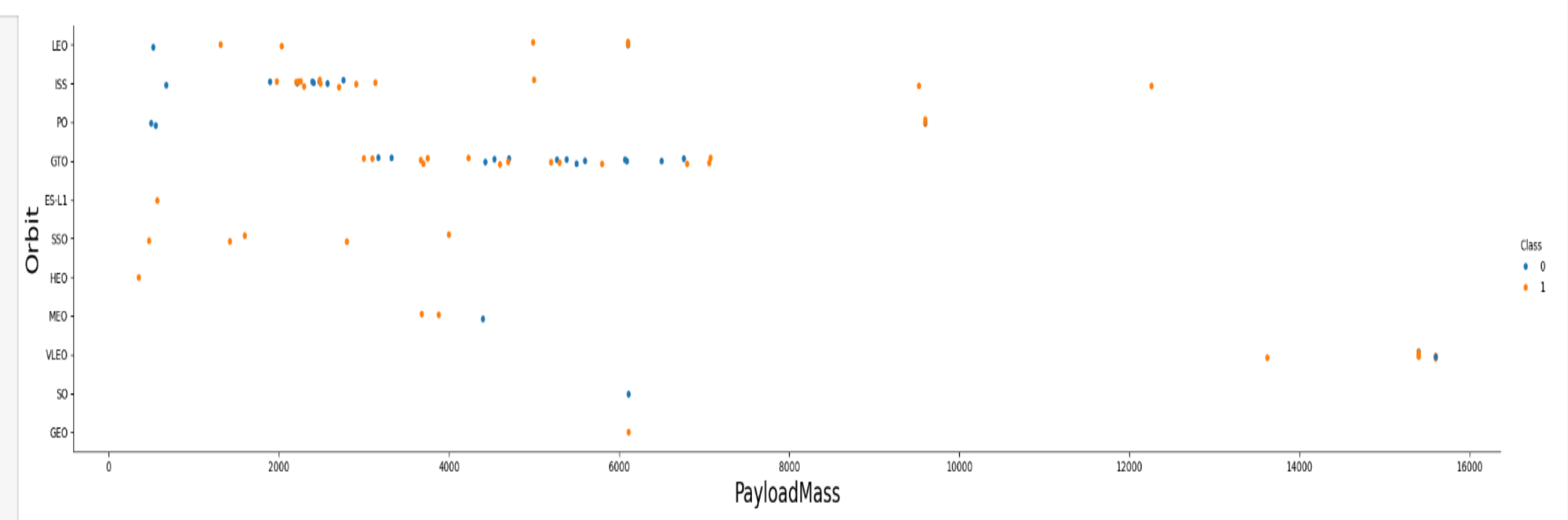
Orbits ES-L1, GEO, HEO, and SSO have 100% success rate.

Flight Number vs. Orbit Type



You should see that in the 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.

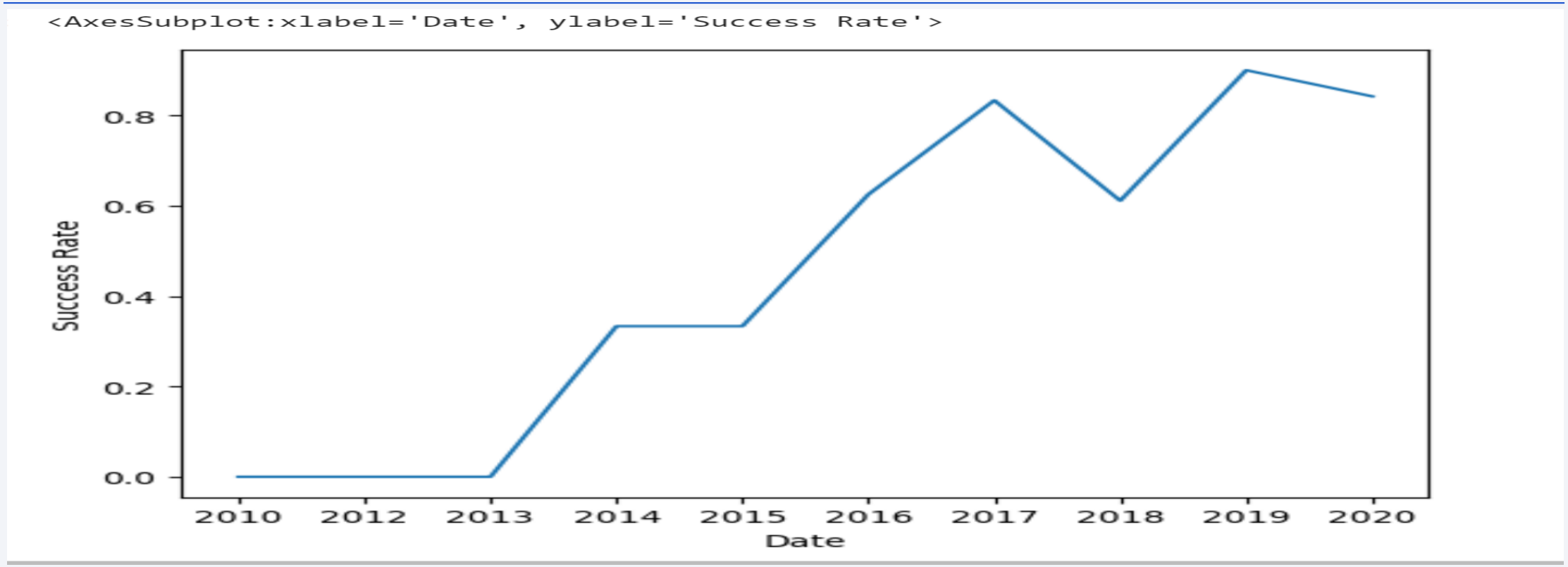
Payload vs. Orbit Type



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 present here.

Launch Success Yearly Trend



The success rate since 2013 kept increasing till 2020.

All Launch Site Names

To find the names of the unique launch sites, we use the following SQL code:

```
%sql select distinct "Launch_Site" from SPACEXTABLE
```

Launch_Site

CCAFS LC-40

VAFB SLC-4E

KSC LC-39A

CCAFS SLC-40

Launch Site Names Begin with 'CCA'

To find 5 records where launch sites begin with `CCA` we use the following SQL code:

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Lar
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Fai
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	Fai
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	

Total Payload Mass

To calculate the total payload carried by boosters from NASA, we use the following SQL code:

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer='NASA (CRS)'
```

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

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

To calculate the average payload mass carried by booster version F9 v1.1, we use the following SQL code:

%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTABLE where "Booster_Version" = 'F9 v1.1'

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

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

To find the date of the first successful landing outcome on ground pad, we use the following SQL code:

```
%sql select min("Date") from SPACEXTABLE where "Landing_Outcome" = 'Success (ground pad)'
```

min(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

To list the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000, we use the following SQL code:

%sql select "Booster_Version", PAYLOAD_MASS__KG_ from SPACEXTABLE where "Landing_Outcome" = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000

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

Booster_Version	PAYLOAD_MASS_KG_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

Total Number of Successful and Failure Mission Outcomes

To calculate the total number of successful and failure mission outcomes, we use the following SQL code:

```
%sql select count(substr("Mission_Outcome",0,8)) as outcome_count,
        substr("Mission_Outcome",0,8) as outcome from SPACEXTABLE group by outcome
```

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

outcome_count	outcome
1	Failure
100	Success

Boosters Carried Maximum Payload

To list the names of the booster which have carried the maximum payload mass, we use the following SQL code:

```
%sql select "Booster_Version" , PAYLOAD_MASS__KG_ from
        SPACEXTABLE where PAYLOAD_MASS__KG_ =
        (select MAX(PAYLOAD_MASS__KG_) from
        SPACEXTABLE)
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

To list the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015, we use the following SQL code:

```
%sql select substr("Date",0,5) as yearint, substr("Date", 6,2) as monthint,"Landing_Outcome",
        "Booster_Version",    "Launch_Site"   from SPACEXTABLE where "Landing_Outcome"='Failure
        (drone ship)' and substr("Date",0,5)='2015'
```

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

yearint	monthint	Landing_Outcome	Booster_Version	Launch_Site
2015	01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015	04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

To 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, we use the following SQL code:

```
%sql select "Landing_Outcome", count("Landing_Outcome") as
outcome_count from SPACEXTABLE group by
"Landing_Outcome"  order by outcome_count desc
```

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

Landing_Outcome	outcome_count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

Launch Sites

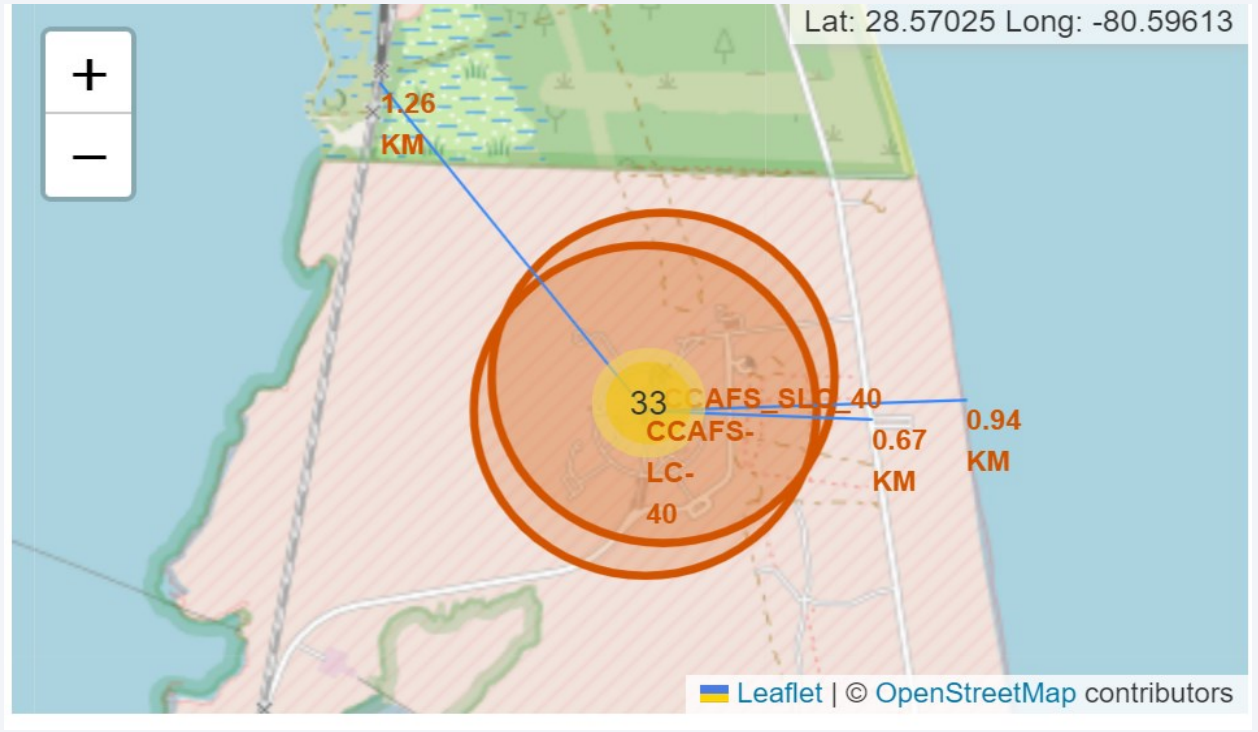
The map is marked where the locations of the all the launch sites. Note that all launch sites are in proximity to the Equator line. Also all launch sites are in very close proximity to the coastline.

Launch Results by Site

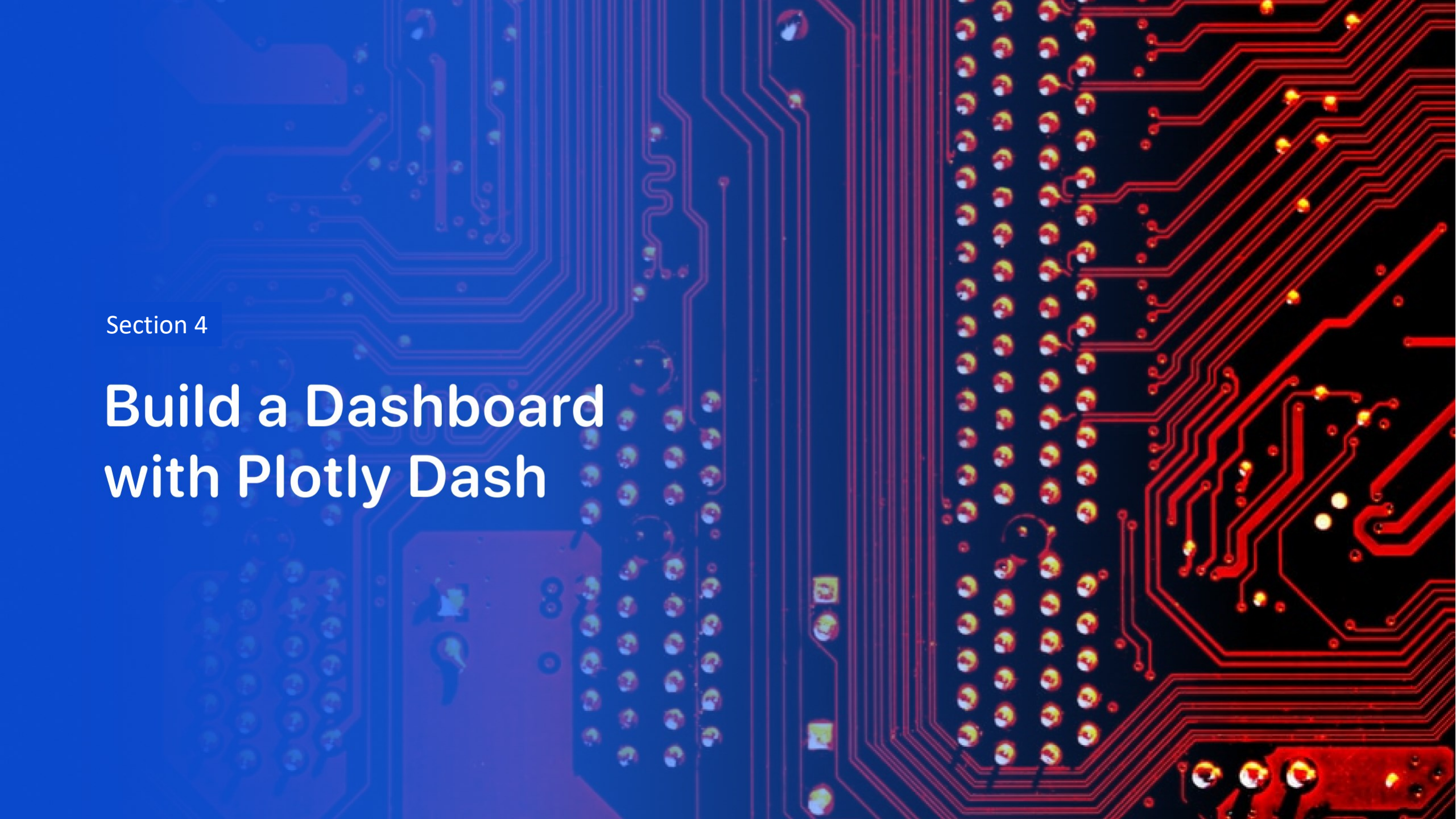
We create markers for all launch records. If a launch was successful (class=1), then we use a green marker and if a launch was failed, we use a red marker (class=0)

Note that a launch only happens in one of the four launch sites, which means many launch records will have the exact same coordinate. Marker clusters can be a good way to simplify a map containing many markers having the same coordinate.

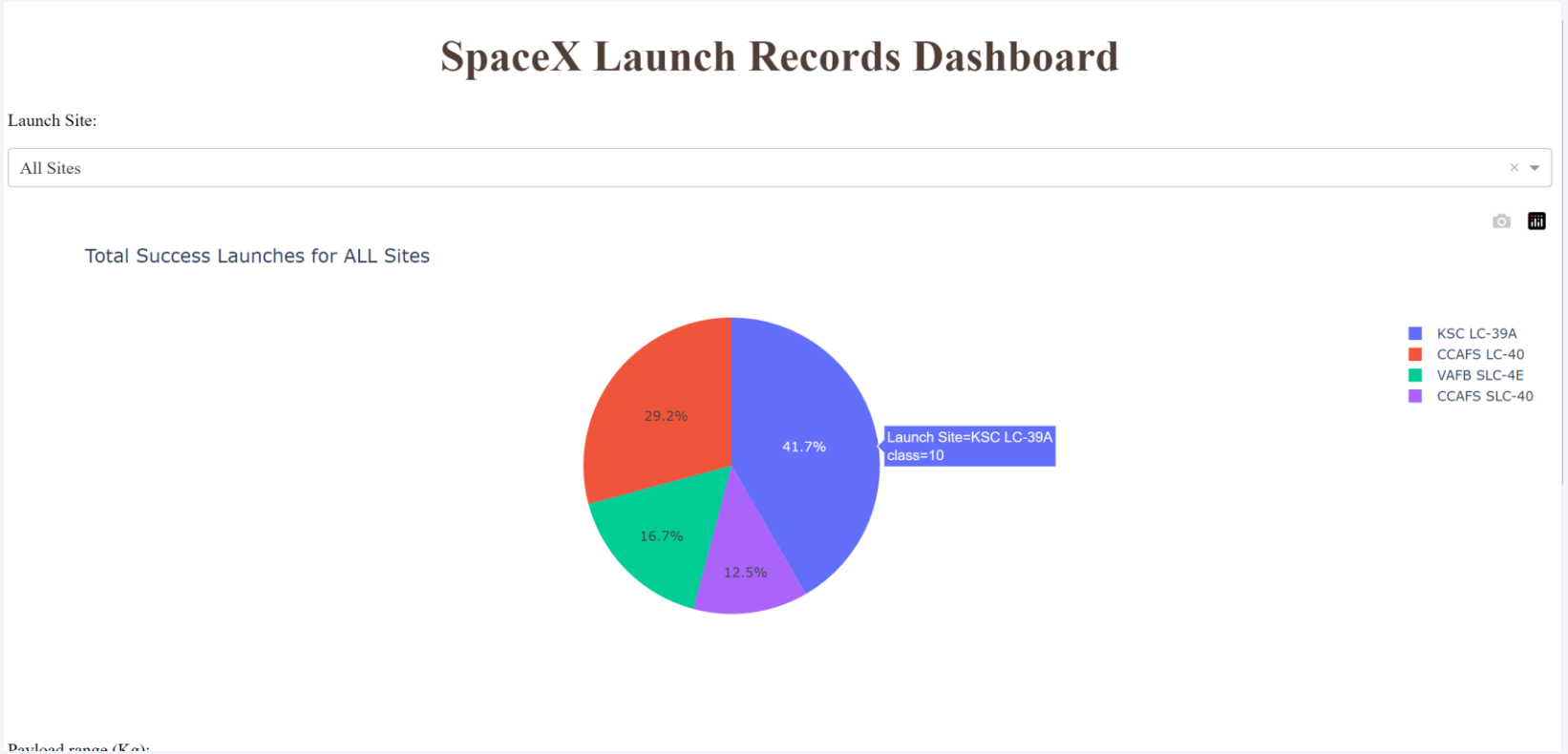
Launch Site Proximity to Coastline, Highway and Railroad



The folium generated map shows a distance line with distance calculated between the launch site and the coastline, between the launch site and highway, and between the launch site and railroad.

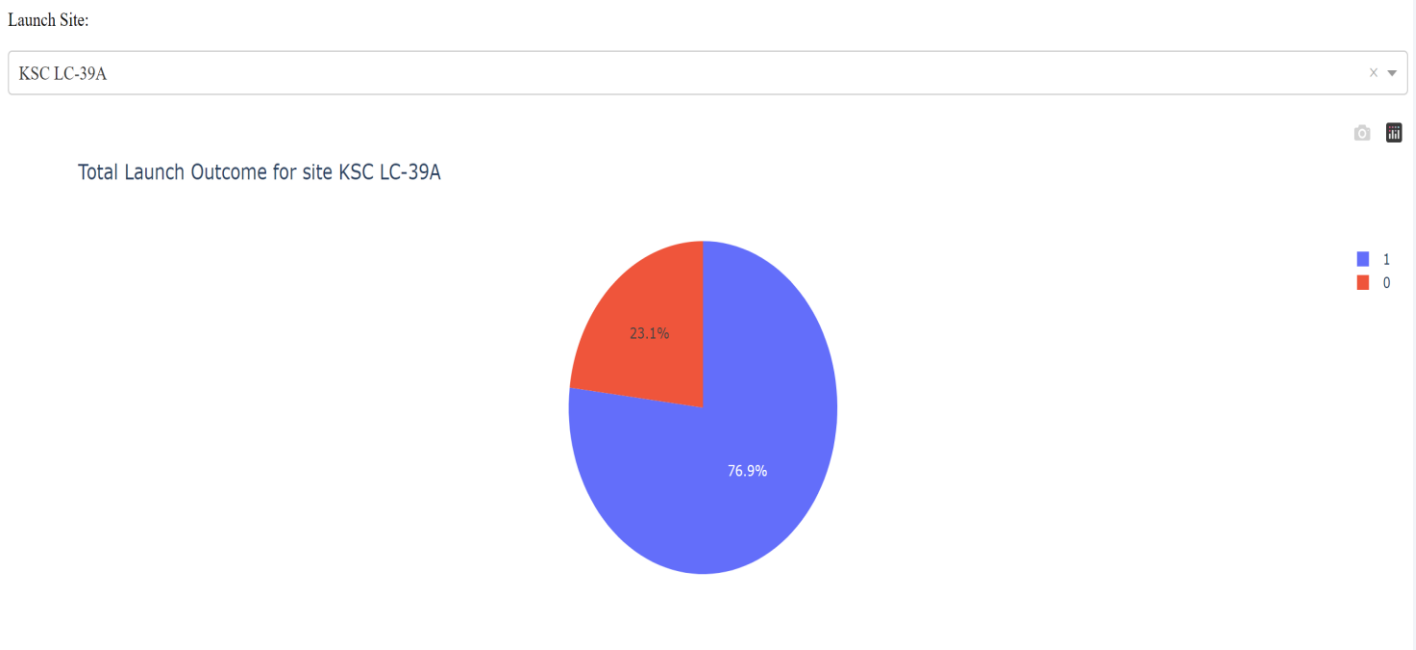

Section 4
Build a Dashboard with Plotly Dash

Launch Success for All Sites



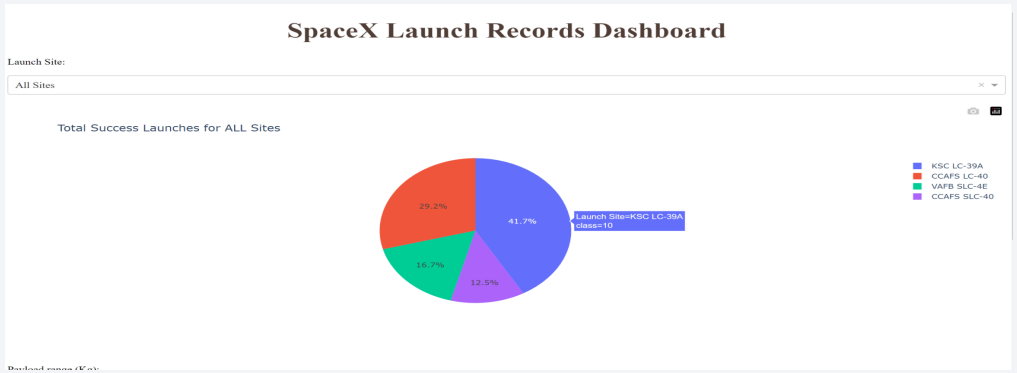
The pie chart shows total successful launches for all sites. Site KSC LC-39A has the highest successful launches among the sites.

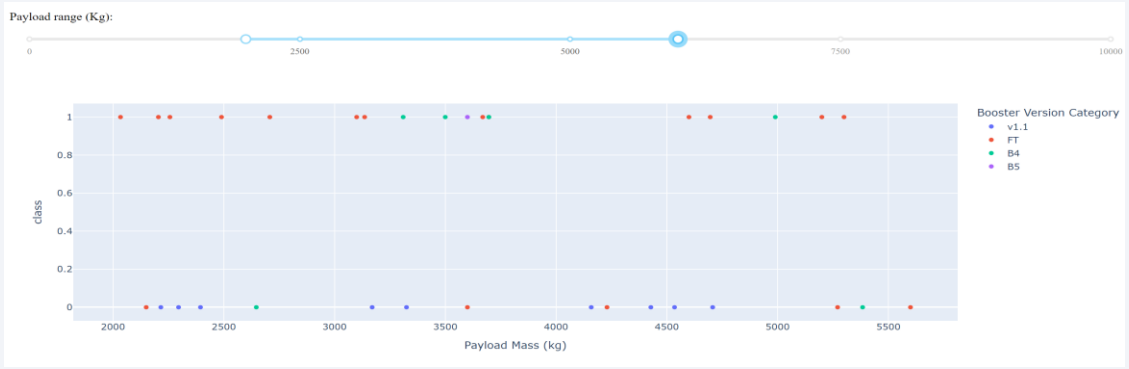
Launch Site with Highest Success

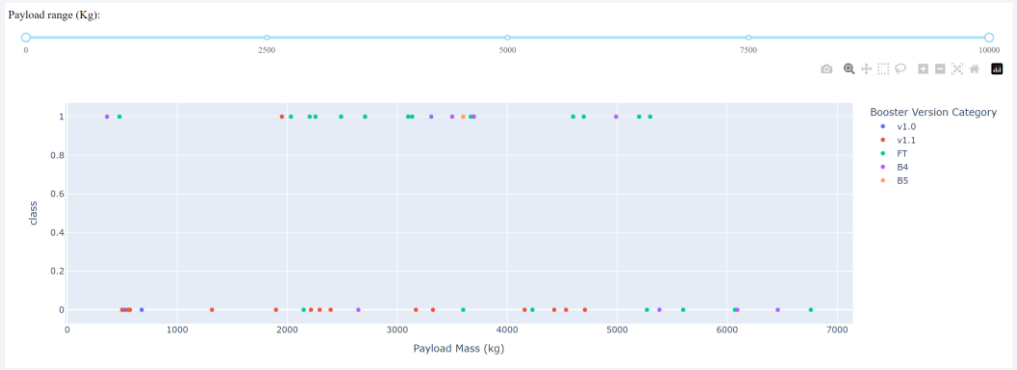


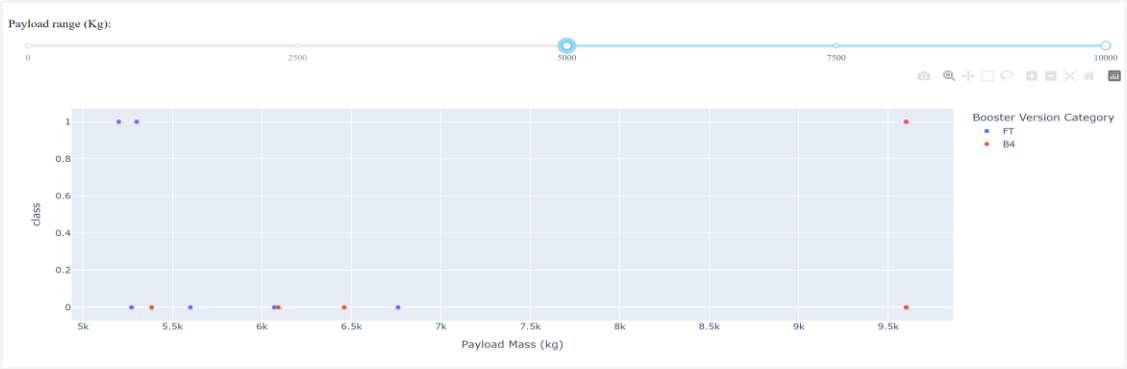
The pie chart shows the total launch outcome for site KSC LC-39A. It has 76.9% success, 23.1% fail.

Payload vs Outcome









Payload range 2000-400 has the highest success rate while the range 5500-7000 has the lowest launch success rate. Among the F9 booster versions, FT has the highest launch success rate.

Section 5

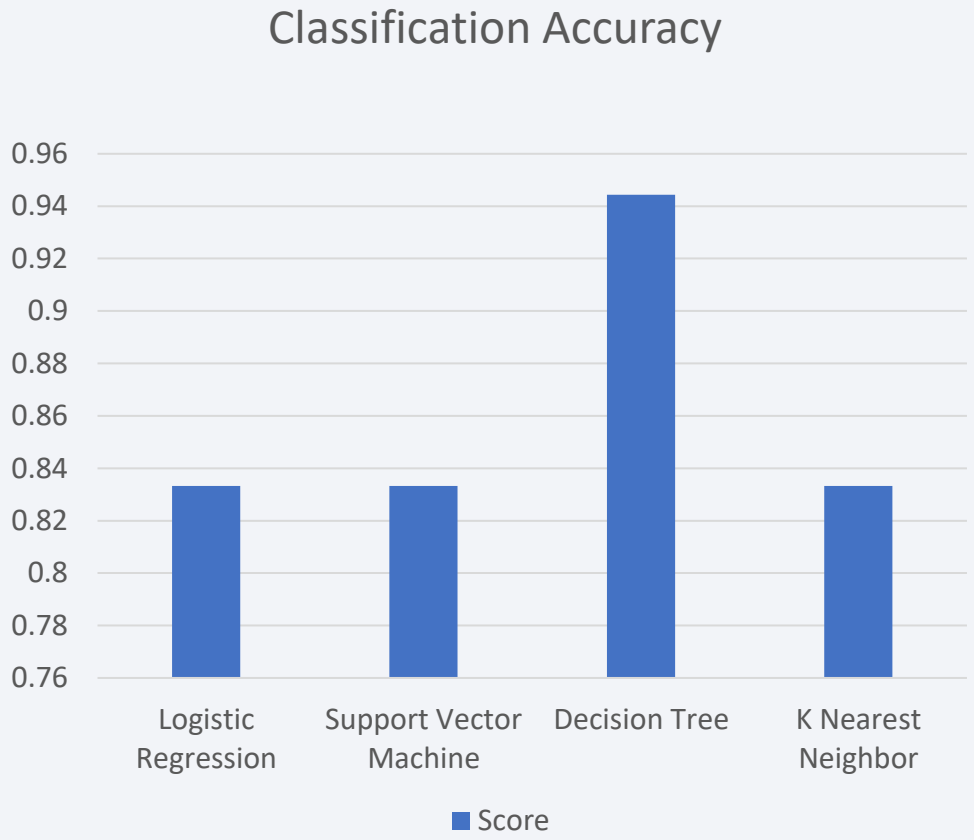
Predictive Analysis (Classification)

Classification Accuracy

We build a machine learning pipeline to predict if the first stage of the Falcon 9 lands successfully.

We standardize our data, and split our data into training and testing data. We train the model and perform Grid Search, to find the hyperparameters that allow a given algorithm to perform best. Using the best hyperparameter values, we determine the model with the best accuracy using the training data. We use Logistic Regression, Support Vector machines, Decision Tree Classifier, and K-nearest neighbors. We use a confusion matrix to measure the accuracy of our model..

Among the classification models tested, the decision tree model has the highest classification accuracy of 0.94444.



Confusion Matrix for Decision Tree model

Decision Tree Model:

best parameters: {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'best'}

accuracy on validation data: 0.875

accuracy on the test data: 0.9444444444444444

Among the classification models tested, the decision tree model has the highest classification accuracy of 0.9444444444444444.

The confusion matrix shows the corrected and wrong predictions in comparison with the actual labels. Each confusion matrix row shows the actual true labels in the test set, and the columns show the predicted labels by the decision tree classifier.

The first row is for actual launch outcome ‘did not land’. The second row is for actual launch outcome ‘landed’. Out of the 18 sample data, 6 did not land and 12 landed. Out of the 6 ‘did not land’ the classifier correctly predicted 5, and incorrectly predicted 1 ‘did not land’. Also out of the 12 ‘landed’ outcome, the classifier correctly predicted all of them as ‘landed’. The confusion matrix shows the model’s high accuracy to correctly predict or separate the classes.

Logistic Regression Model

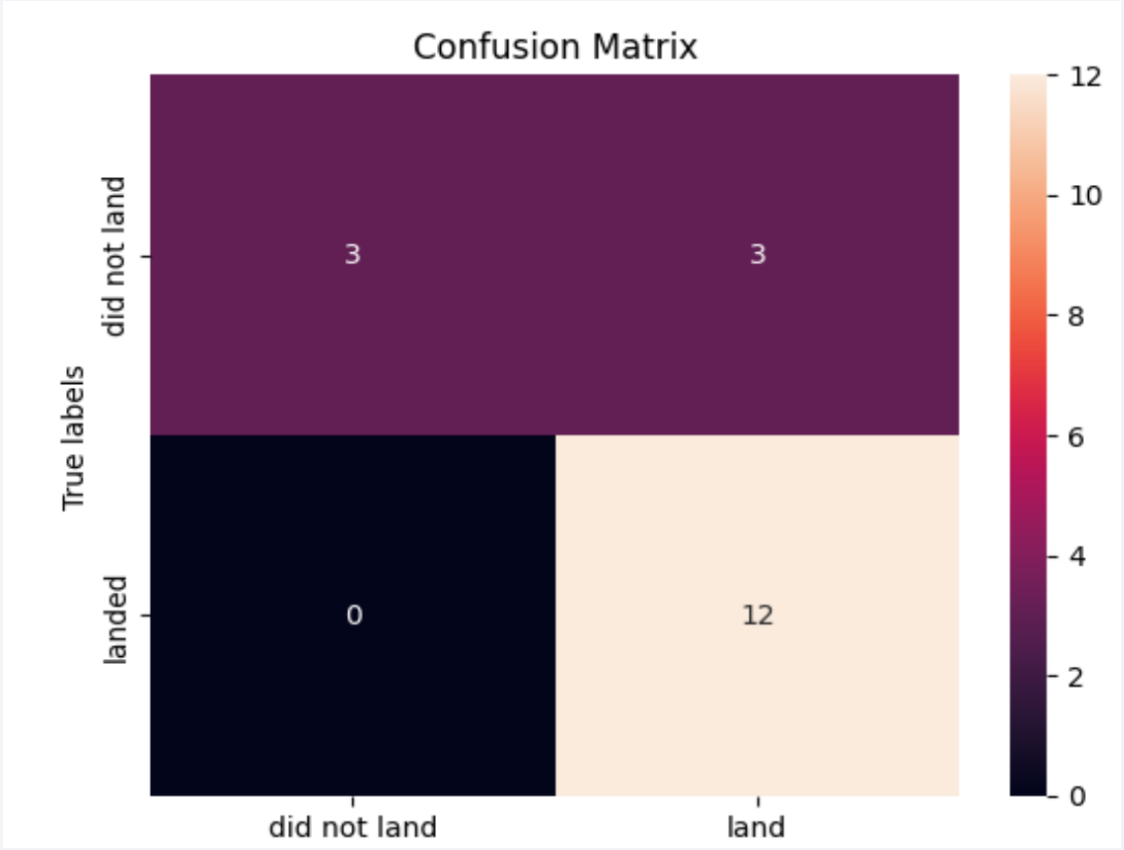
Logistic Regression Model:

best parameters {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}

accuracy on validation data: 0.8464285714285713

accuracy on the test data: 0.8333333333333334

The confusion matrix for Logistic Regression model shows that it doesn't do a good job at predicting outcome 'did not land' because the true label indicates a total of 6 'did not land' but the model correctly predicted 3 'did not laned but the other 3 incorrectly as 'landed', it predicted . But it does a good job at predicting outcome 'landed'.



Support Vector Machine Model

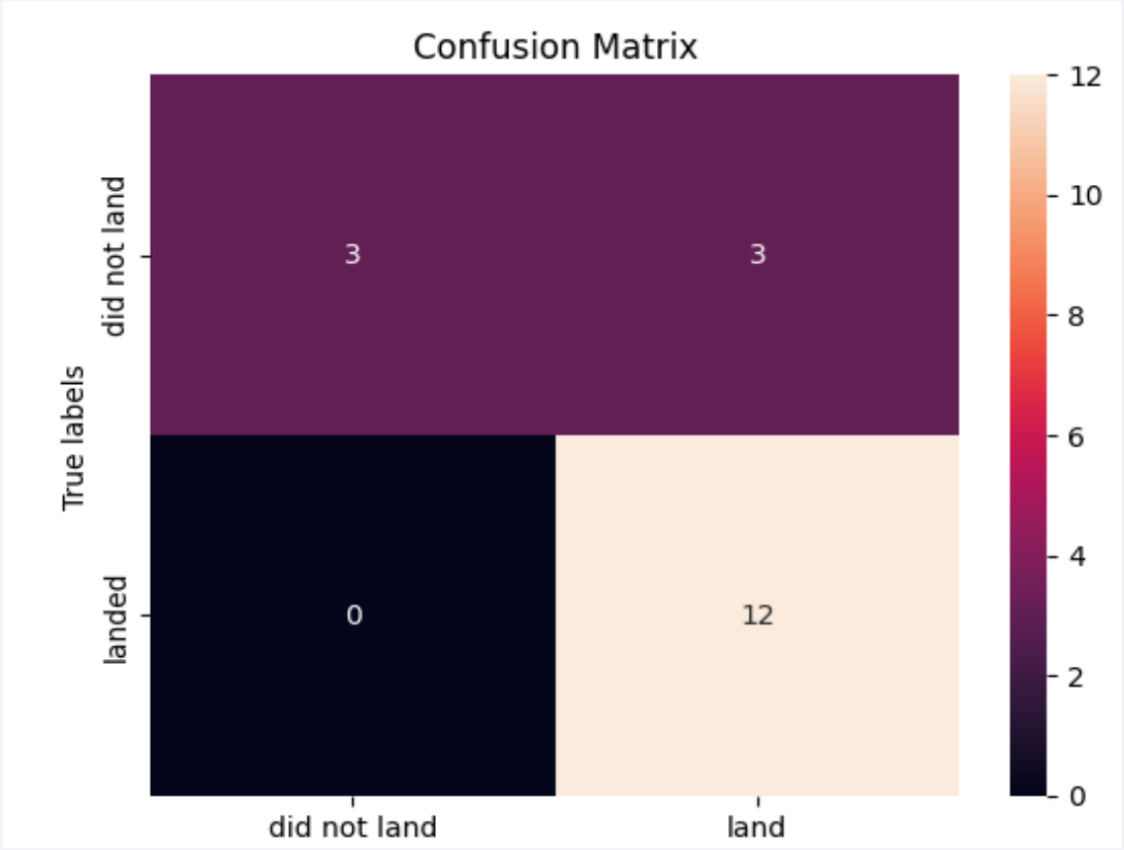
Support Vector Machine Model:

best parameters: {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}

accuracy on validation data: 0.8482142857142856

accuracy on the test data: 0.8333333333333334

The confusion matrix for Support Vector Machine model shows that it doesn't do a good job at predicting outcome 'did not land' because the true label indicates a total of 6 'did not land' but the model correctly predicted 3 'did not laned but the other 3 incorrectly predicted as 'landed' . But it does a good job at predicting outcome 'landed'.



K Nearest Neighbors Model

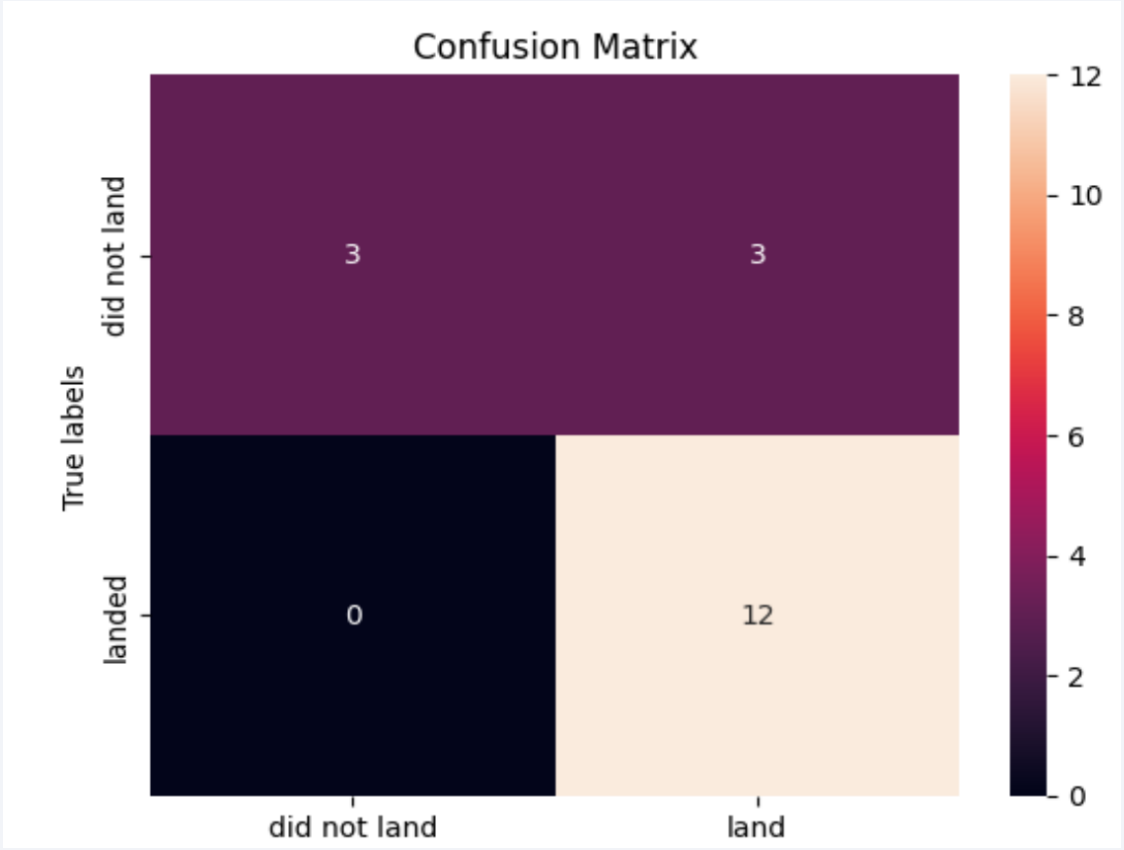
K Nearest Neighbors Model:

best parameters: {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}

accuracy on validation data: 0.8482142857142858

accuracy on the test data: 0.8333333333333334

The confusion matrix for K Nearest Neighbors model shows that it doesn't do a good job at predicting outcome 'did not land' because the true label indicates a total of 6 'did not land' but the model correctly predicted 3 'did not laned but the other 3 incorrectly predicted as 'landed',. But it does a good job at predicting outcome 'landed'.



Conclusions

We can collect data from various sources using APIs or Webscraping. But we need to perform some data wrangling to clean and format the data to make them more useful for analysis.

Using various tools to analyze the data, we discover that:

- Flight number, payload mass, launch sites, and orbit type could affect the launch outcome success
- The success rate since 2013 kept increasing till 2020.
- Launch sites are near the equator. And they are in very close proximity to coastlines.

After training and testing our models, we discover that among the models tested, the decision tree model has the highest classification accuracy. We used the confusion matrix to measure the accuracy of different predictive models.

In conclusion, because of the high accuracy of the predictive model we developed and tested, we will be able to predict with confidence if the Falcon 9 first stage will land successfully.

Appendix

Please refer to the following GitHub repository which contains all the completed labs and related files.

applied_capstone_gsd/README.md at main · euroamerasia/applied_capstone_gsd · GitHub

Thank you!