IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ridhesh Goti
09.10.2022

Outline

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

Executive Summary

- With main objective of predicting the successful launches for Falcon9 from the SpaceX company, here we start with the data collection using the SpaceX API, and web-scraping from the website, we clean and filter some of the data. After data wrangling, we query the data using SQL to format the data nicely and using the visualisations in Python and Dash app, we find several features which could be good predictors for the successful launch. With finding payload mass, launch site, location, and other rocket components assembly, we create the final dataset. We also visualise the launch site in Folium to get the better understanding of successful locations, and their proximity.
- With modelling in four machine learning models: Logistic regression, Decision tree classifier, SVM, and KNN classifier, we obtain the highest accuracy of 0.8477 on training dataset and 0.833 in the test dataset. We find all the models similar with the same accuracy values.

Introduction

- Nowadays, many space agencies are exploring the space with their successful missions. Agencies like NASA, SpaceX, Blue Origin, ESA, and others are trying to get the optimum result for the mission like cost, budget, weight, and success of the mission. With a lot of circumstances, cost and budget are the main factors for the successful mission. At the moment, SpaceX is the main frontier with low cost and reusable booster agency. There could be numerous factors affecting for this kind of launches. It is important to know about the success of the launches before the final launch.
- We want to land the rocket for reuse and also want the launch successful. To avoid the failures as possible as much, here we try to find the best optimal factors that could lead to the successful launches. With the main aim of predicting the successful rate for launches, here we will model the SpaceX past failures and successes using different variables to predict the future mission successes.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - In this method, we use two ways to collect the data. The first one is the SpaceX API in Python to collect the launch site, payload data, booster version, location data, landing success, etc. The second one is to extract the information from WikiPedia tables related to SpaceX mission like data time, payload data, orbit, landing outcome, booster name, etc.
- Perform data wrangling
  - Data wrangling of the data involves finding the best features to train the model. In this method, we explore some of the features, their datatype, and counting the number of categories. Also, applied the one-hot encoding to categorical features.
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Here we query some of the features with SQL and panda library to know better about the data and visualize the data features to see if there is any correlation between them.
- Perform interactive visual analytics using Folium and Plotly Dash
  - With help of Folium and plotly dash, we create the interactive dashboard to explore the final dataset.
- Perform predictive analysis using classification models
  - We build four possible models KNN, decision tree, SVM, and logistic regression to perform the model on the training sample. With the trained model, we also try to perform the tuning of the hyper parameters for each model using GridSearchCV and find the best possible parameters. The accuracy is obtained from the test sample with these four models.

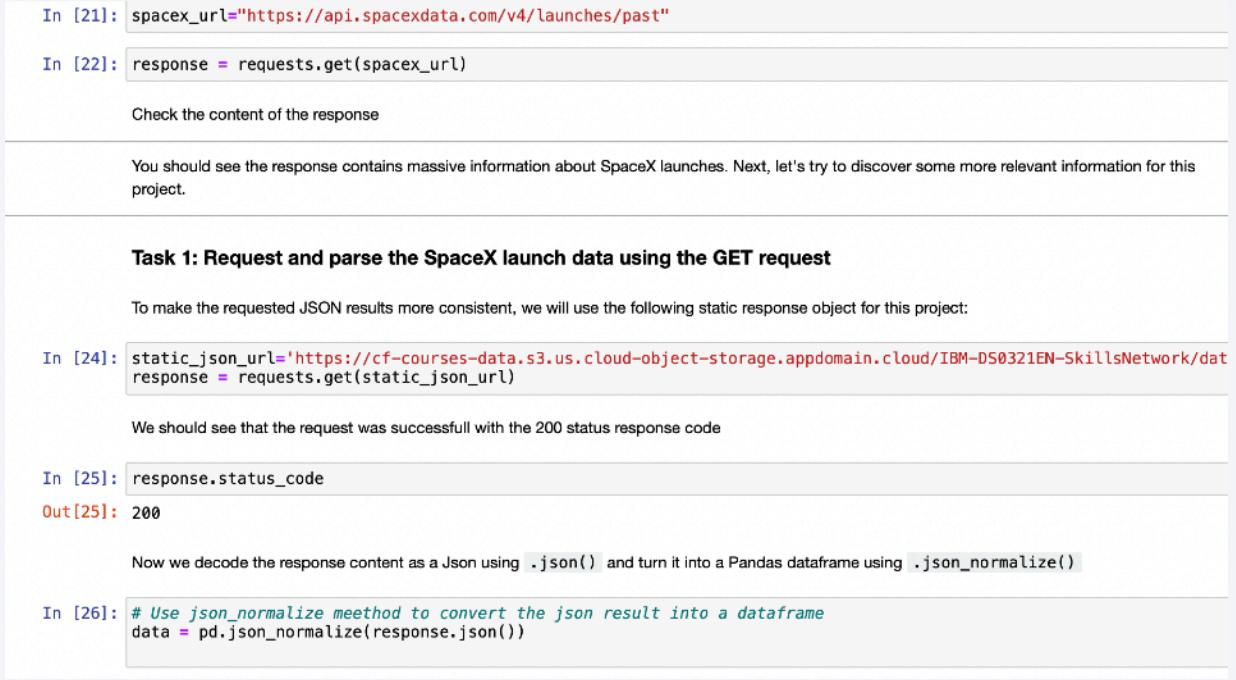
Data Collection

- Using the SpaceX API, we use python to collect the booster version, payload data, launch site, and many other related data. We also use BeautifulSoup to extract the web-based data using the web-scraping in python. The extracted information is success rate for launches, date time of launches, sites, locations, etc.
- Data collection:

1. Call the SpaceX API and decode the JSON information in data frame using .json_normalize().
2. Retrieve the data in a proper format with cleaning from a self-made functions.
3. Then perform the web-scraping from wikipedia table related to SpaceX Falcon9 launches with the use of BeautifulSoup library in python.
4. Convert it to data frame by parsing the html table in the web data soup.

Data Collection – SpaceX API

- Use get request to retrieve the SpaceX API url and .json_normalize to get it into the data frame format.
- Also, using different functions, retrieving the data was easy.



In [21]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [22]:
```
response = requests.get(spacex_url)
```

Check the content of the response

You should see the response contains massive information about SpaceX launches. Next, let's try to discover some more relevant information for this project.

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 [24]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/dat
response = requests.get(static_json_url)
```

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

In [25]:
```
response.status_code
```

Out[25]:
```
200
```

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

In [26]:
```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

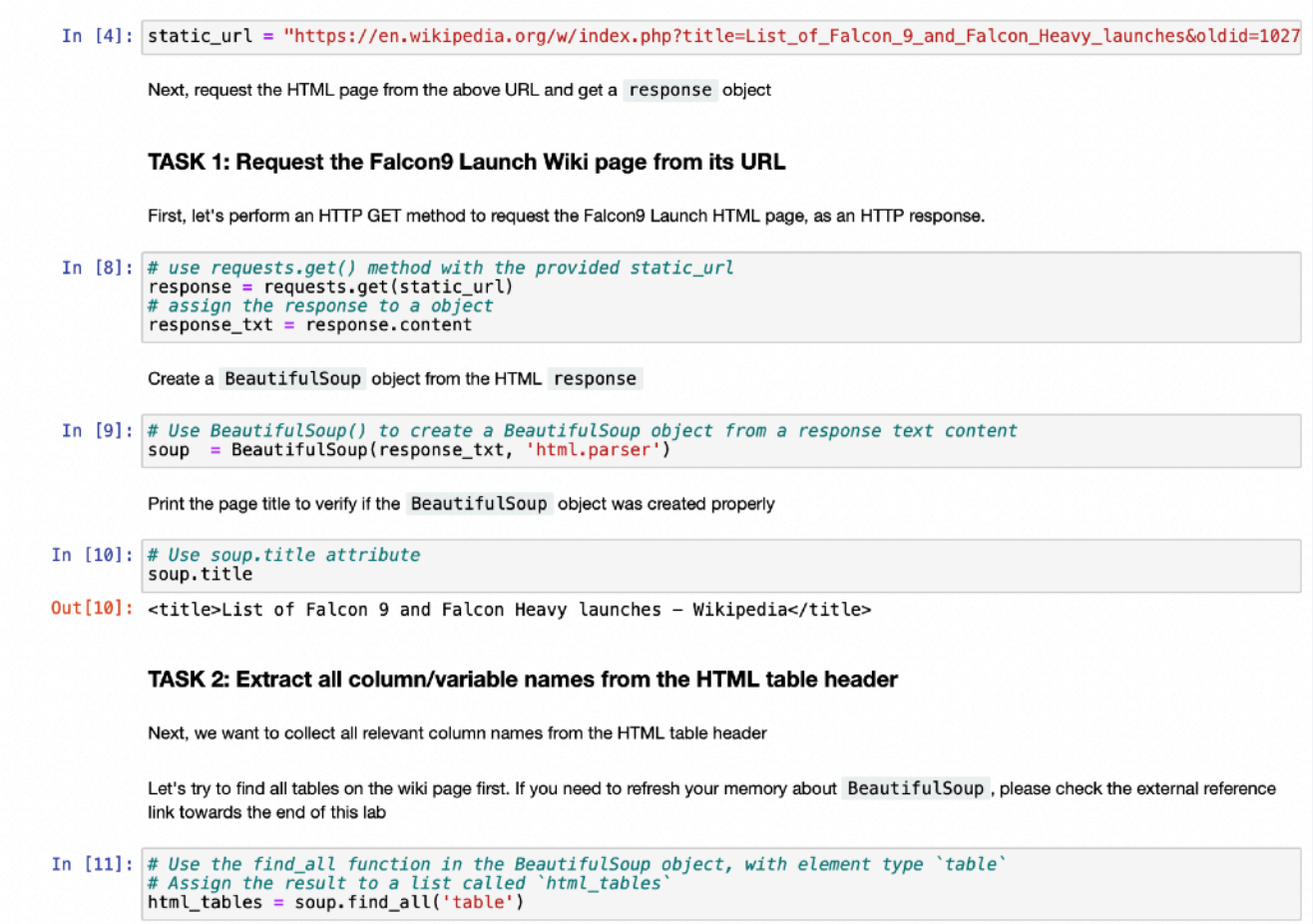
SpaceX API link and request method

- Github link: https://github.com/rid181198/Spacex-Falcon9-launches/blob/759de29455648971bc1eb833116c297bed06af6d/Data_collection.ipynb

Data Collection - Scraping

- Parse the html table from the wikipedia table to BeautifulSoup and retrieve the information using the seld-made function.

- Github link: https://github.com/rid181198/Spacex-Falcon9-launches/blob/759de29455648971bc1eb833116c297bed06af6d/Data_web_scraping.ipynb



```
In [4]: static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027
```

Next, request the HTML page from the above URL and get a `response` object

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.

```
In [8]: # use requests.get() method with the provided static_url
        response = requests.get(static_url)
        # assign the response to a object
        response_txt = response.content
```

Create a `BeautifulSoup` object from the HTML `response`

```
In [9]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
        soup  = BeautifulSoup(response_txt, 'html.parser')
```

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

```
In [10]: # Use soup.title attribute
         soup.title
Out[10]: <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 the external reference link towards the end of this lab

```
In [11]: # 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')
```

Data webscraping using BeautifulSoup method

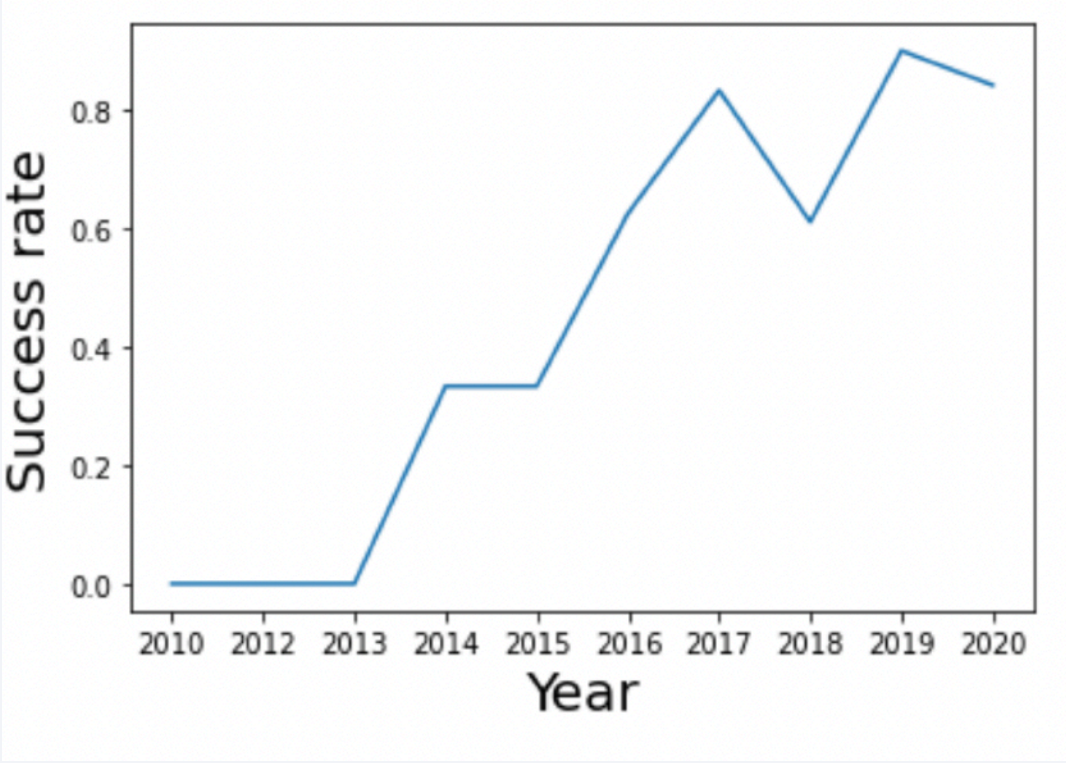
Data Wrangling

- Using the exploratory data analysis, the training features are determined in this part.
- Determined the number of different type of orbits, number of launches at different sites, etc.
- Create the target variable landing outcome for the model.
- Github link: https://github.com/rid181198/Spacex-Falcon9-launches/blob/759de29455648971bc1eb833116c297bed06af6d/Data_wrangling.ipynb

EDA with Data Visualization

- Relations like payload mass with flight number, launch site with flight number, launch site with payload mass, orbit with flight number, and orbit with payload mass help to understand the possible correlation and which payload mass, launch site, orbit type has high landing outcome success.
- Relation between years and success rate gives the idea about progress over the years.

- Github link: https://github.com/rid181198/Spacex-Falcon9-launches/blob/759de29455648971bc1eb833116c297bed06af6d/EDA_python_visualization.ipynb



Relation between year and success rate

EDA with SQL

- Loaded the SQL database to IBM cloud and perform the following SQL queries to get insights of the data.
- Name of the unique launch site with DISTINCT
- Filtered the launch site starting from ‘CCA’ with wild card character
- Total payload mass with SUM function
- Average payload mass with AVG function for one booster
- First date when first success happened for the ground pad using WHERE and MIN function
- With AND logic, filter out payload mass between 4000 and 6000 only for drone ship
- With COUNT, counted the successful and failures missions
- Using a subquery, find the booster with the maximum payload mass
- With YEAR function to date, find information of failed drone ship in the year 2015
- With GROUP BY and DESC, find the landing outcome between two dates
- Github link: https://github.com/rid181198/Spacex-Falcon9-launches/blob/759de29455648971bc1eb833116c297bed06af6d/EDA_sql.ipynb

Build an Interactive Map with Folium

- Mark all the launch site using folium.Circle and folium.Marker with labels and locations.
- With MarkerCluster object, added markers for a cluster of landing outcome for the same launch site.
- With help of distance function, calculate the distance between two locations, and draw a line using folium.PolyLine function.
- This helps to understand the geographic of our data set and which locations have high success using the maps.
- Github link: https://github.com/rid181198/Spacex-Falcon9-launches/blob/759de29455648971bc1eb833116c297bed06af6d/Folium_visualization_locations.ipynb

Build a Dashboard with Plotly Dash

- Build the interactive application to get the pie chart for success rate for each and all launch sites using dropdown method.
- Build the second interactive plot using range slider in dash to get the success for each booster version in different payload mass range.
- This helps to give very compact information about the dataset using the interactive plots.
- Github link: https://github.com/rid181198/Spacex-Falcon9-launches/blob/759de29455648971bc1eb833116c297bed06af6d/spacex_dash_app_visuals.py

Predictive Analysis (Classification)

- Perform the normalisation using StandardScalar to the data and split the data in training and test samples with tratin_test_split with 0.3 sample size
- With GridSeachCV (cv=10), obtain the best hyper parameters for each model: logistic regression, SVM, KNN, Decision Tree.
- Using metrics, calculated the accuracy score for each model.
- Github link: https://github.com/rid181198/Spacex-Falcon9-launches/blob/759de29455648971bc1eb833116c297bed06af6d/Machine_learning_modeling_prediction.ipynb

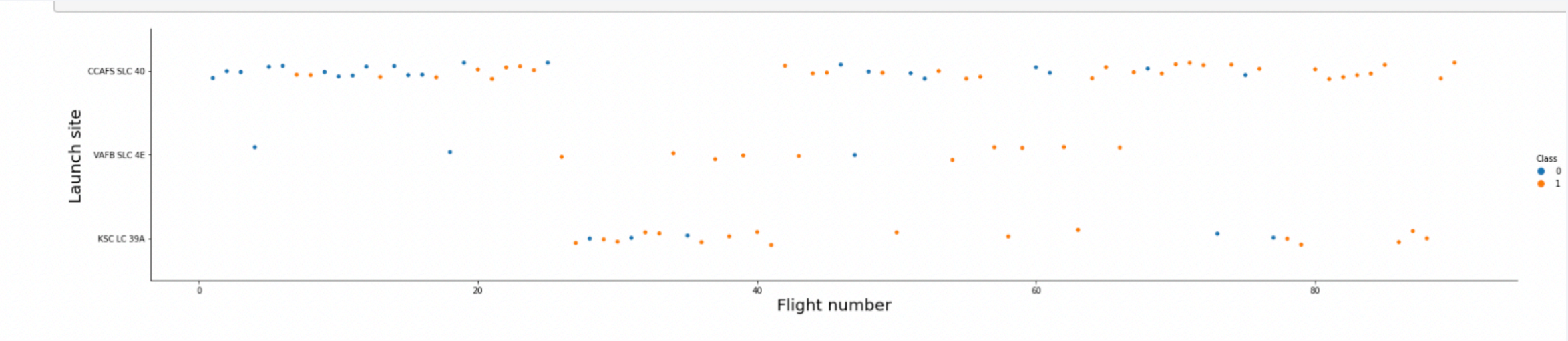
Results

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

Section 2

Insights drawn from EDA

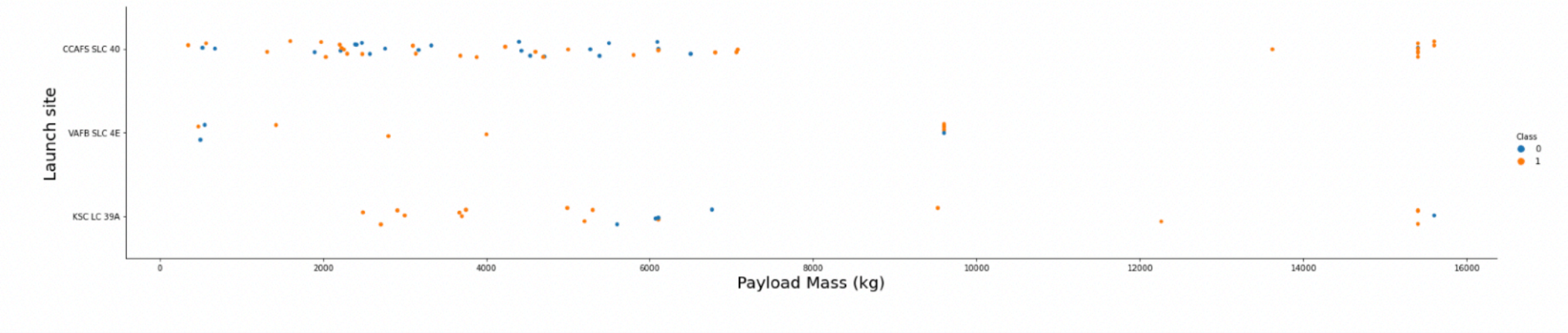
Flight Number vs. Launch Site



Launch site vs Flight number

- From the plot, 0 means fail and 1 means success
- It shows that initially success rate was very low for all launch sites and eventually with increasing flights, success becomes more.

Payload vs. Launch Site

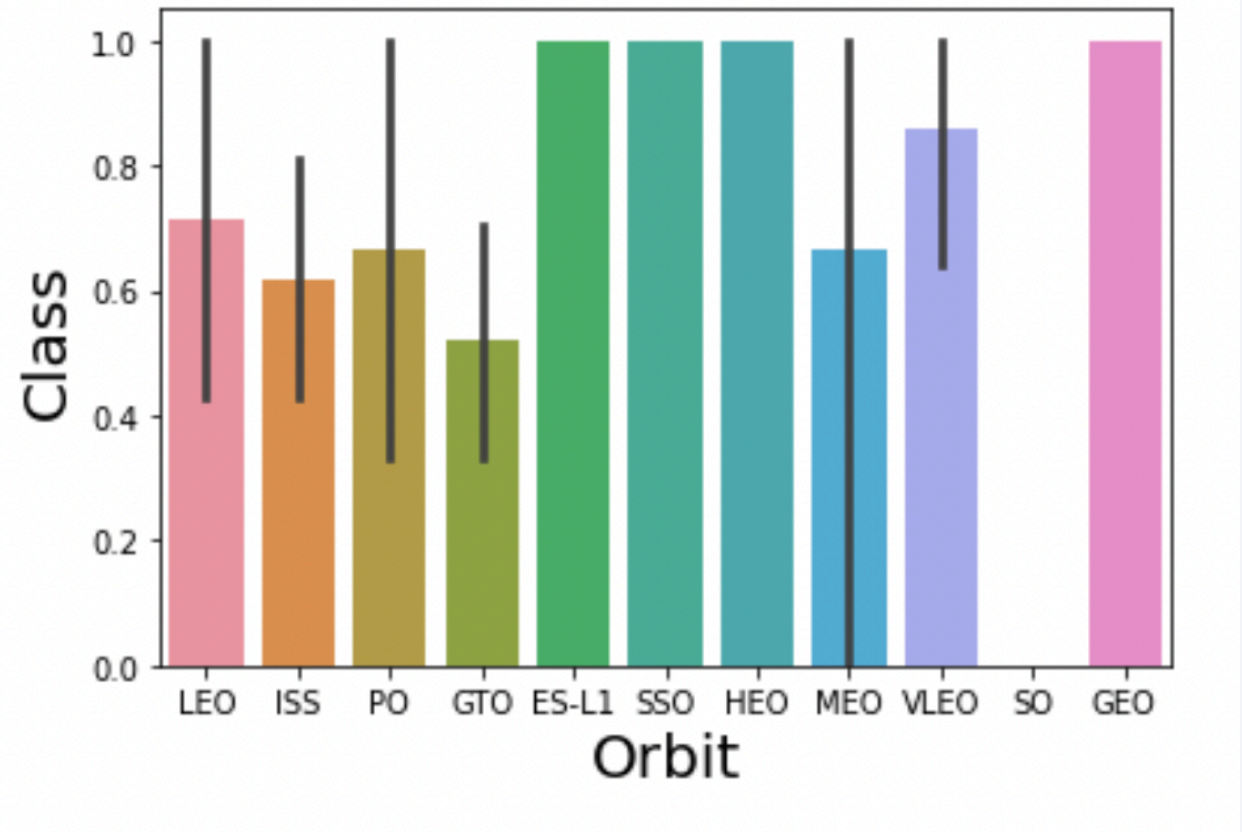


Launch site vs Payload mass

- Conclusion: The higher the payload mass, the greater the success rate would be. Also, only launch site VAFB SLC 4E has low payload mass but success rate is still higher for that case.

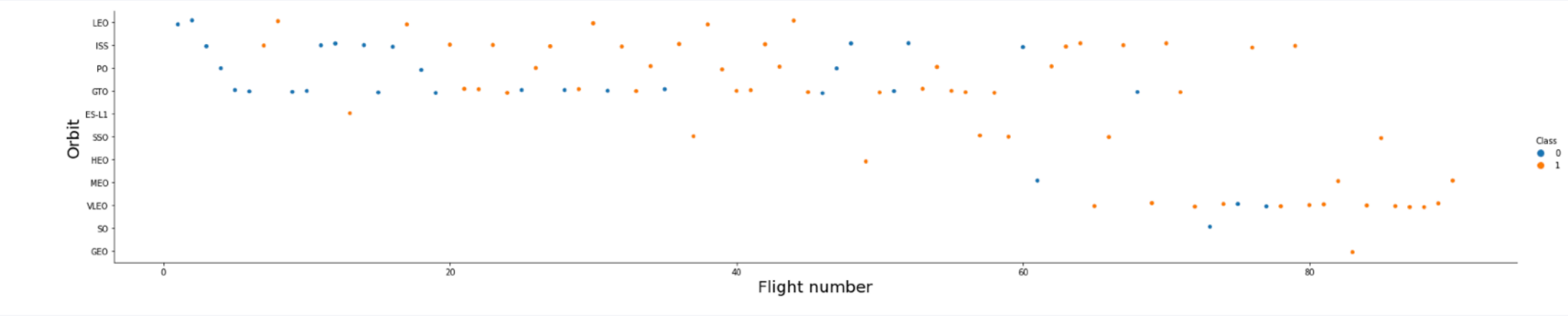
Success Rate vs. Orbit Type

- It shows that four orbits have success rate 1 but their sample size is very low.
- Comparing the orbits with large sample size, VLEO has high success rate.



Success rate bar chart with the Orbital type

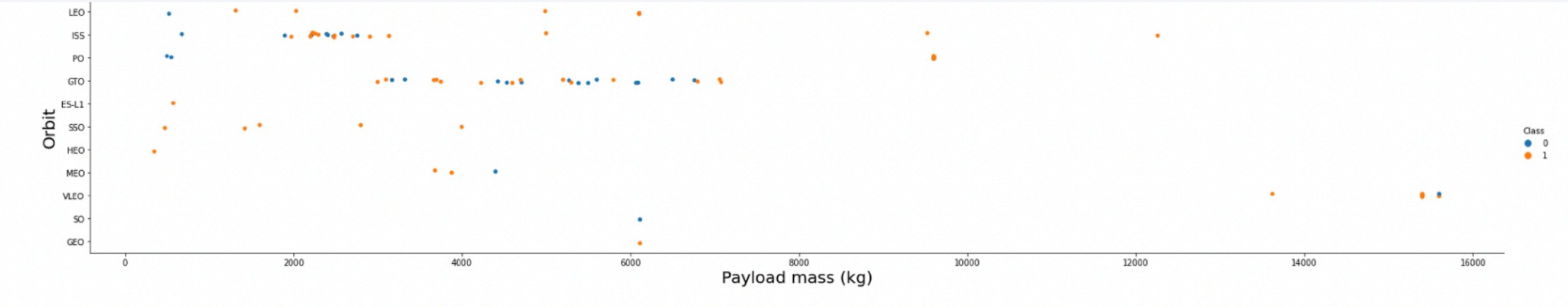
Flight Number vs. Orbit Type



Flight number vs Orbit type

- Some orbits like VLEO, LEO, ISS have high success rate with flight number while GTO shows nothing like that.

Payload vs. Orbit Type

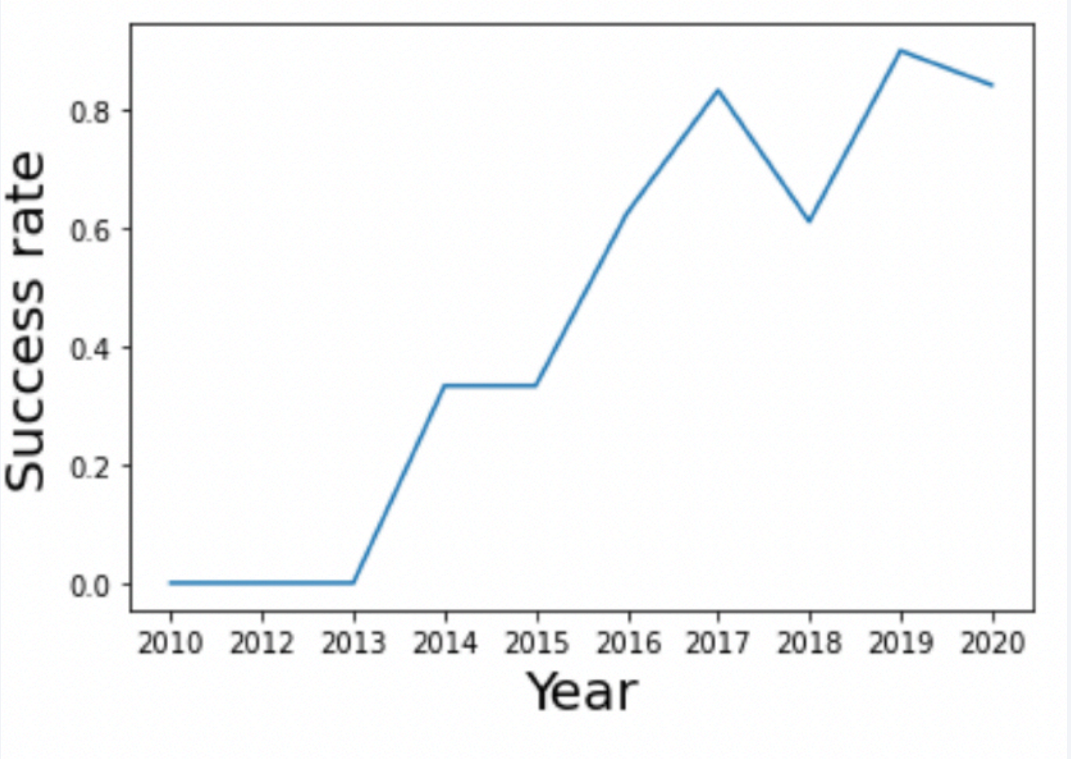


Payload mass vs Orbit

- Success rate is high for heavy payload mass for PO, LEO, and ISS while GTO doesn’t show any trend here.

Launch Success Yearly Trend

- All of a sudden, after 2013, the success rate increases gradually.



Relation between year and success rate

All Launch Site Names

Task 1

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

```
In [17]: %sql select distinct(LAUNCH_SITE) from SPACEX
```

```
 * ibm_db_sa://frq68978:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

Out[17]:

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

Caption

- Use DISTINCT function to find the unique launch site

Launch Site Names Begin with 'CCA'

Task 2

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

```
In [23]: %sql select * from SPACEX where launch_site like 'CCA%' limit 5;
 * ibm_db_sa://frq68978:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

Out[23]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure
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
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	
2012-10-08	00:35:00	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	

Caption

- Use LIKE with wild card character % to find the launch sites starting with ‘CCA’.

Total Payload Mass

Task 3

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

```
In [34]: %sql select sum(payload_mass__kg_) as Total_payload from Spacex where customer = 'NASA (CRS)'
```

```
 * ibm_db_sa://frq68978:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

Out[34]:

total_payload
45596

Caption

- Use SUM function to calculate the total payload mass for NASA CRS

Average Payload Mass by F9 v1.1

Task 4

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

```
In [38]: %sql select avg(payload_mass__kg_) as Average_payload from spacex where booster_version = 'F9
```

```
 * ibm_db_sa://frq68978:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databa
ses.appdomain.cloud:32328/bludb
Done.
```

Out[38]:

average_payload
2928

Caption

- Find 2928 average payload mass using AVG function for 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 [42]:
```
%%sql select min(date) as date_ground_first from
spacex where landing__outcome = 'Success (ground pad)';
```

```
 * ibm_db_sa://frq68978:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databa
ses.appdomain.cloud:32328/bludb
Done.
```

Out[42]:

date_ground_first
2015-12-22

Caption

- Date is 2015/12/22 when the first success was achieved in 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 [43]:
```
%sql select booster_version, landing__outcome, payload_mass__kg_ from spacex \
where landing__outcome = 'Success (drone ship)' and (payload_mass__kg_ between 4000 and 6000)
```

```
 * ibm_db_sa://frq68978:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

Out[43]:

booster_version	landing__outcome	payload_mass__kg_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

Caption

- Applied WHERE clause to filter drone ship success and use AND logic for payload mass between 4000 and 6000. Obtain four booster version with this filter.

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

```
In [46]: %sql select mission_outcome, count(mission_outcome) as count from spacex group by mission_out
```

```
 * ibm_db_sa://frq68978:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databa
ses.appdomain.cloud:32328/bludb
Done.
```

Out[46]:

mission_outcome	COUNT
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Caption

- With COUNT function, calculate the successful and failure of outcomes for all missions. 100 missions have success and 1 has failure.

Boosters Carried Maximum Payload

- Using a subquery, find the maximum payload mass and compare it with payload and obtain the booster version with those values.

Task 8

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

In [62]:
```
%%sql select booster_version, payload_mass__kg_ from
spacex where payload_mass__kg_ = (select max(payload_mass__kg_) from spacex);
```

```
 * ibm_db_sa://frq68978:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.data
ses.appdomain.cloud:32328/bludb
Done.
```

Out[62]:

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

Caption

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 [70]:
```
%%sql select landing__outcome, booster_version, launch_site, date
from spacex where landing__outcome = 'Failure (drone ship)' and year(date) = 2015;
```

```
 * ibm_db_sa://frq68978:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

Out[70]:

landing__outcome	booster_version	launch_site	DATE
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40	2015-01-10
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40	2015-04-14

Caption

- Use YEAR, WHERE function to find dron ship failures in the year 2015.

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

```
In [119]: %%sql select landing__outcome, count(landing__outcome) as counts from
          spacex where date in (select date from spacex where date
                                between '2010-06-04' and '2017-03-20')
          group by landing__outcome order by counts desc;

 * ibm_db_sa://frq68978:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

Out[119]:

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

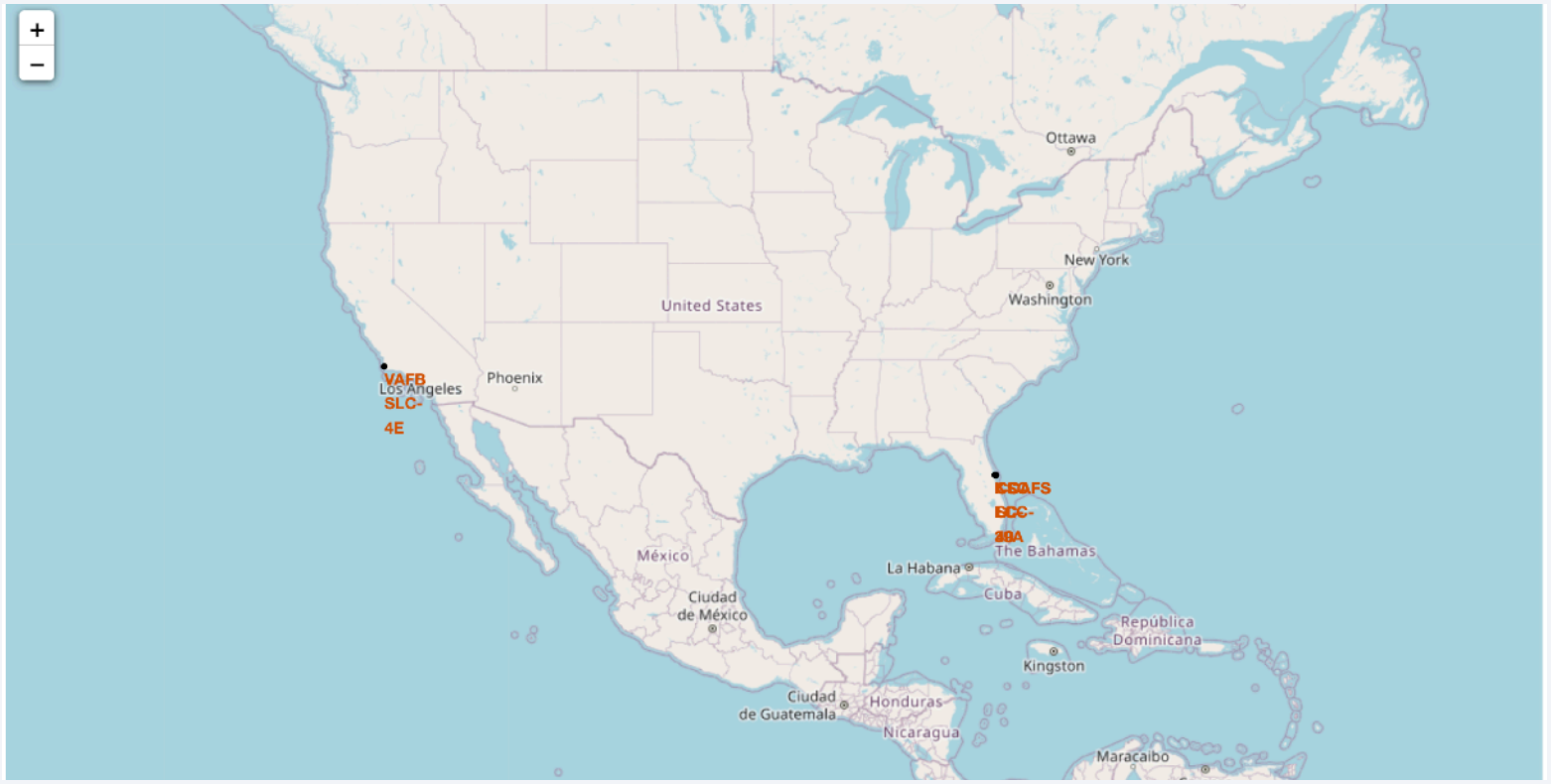
Caption

- Using a subquery, WHERE, COUNT, DESC, find the outcomes between the given dates and ordered in the descending order for each outcome.

Section 3

Launch Sites Proximities Analysis

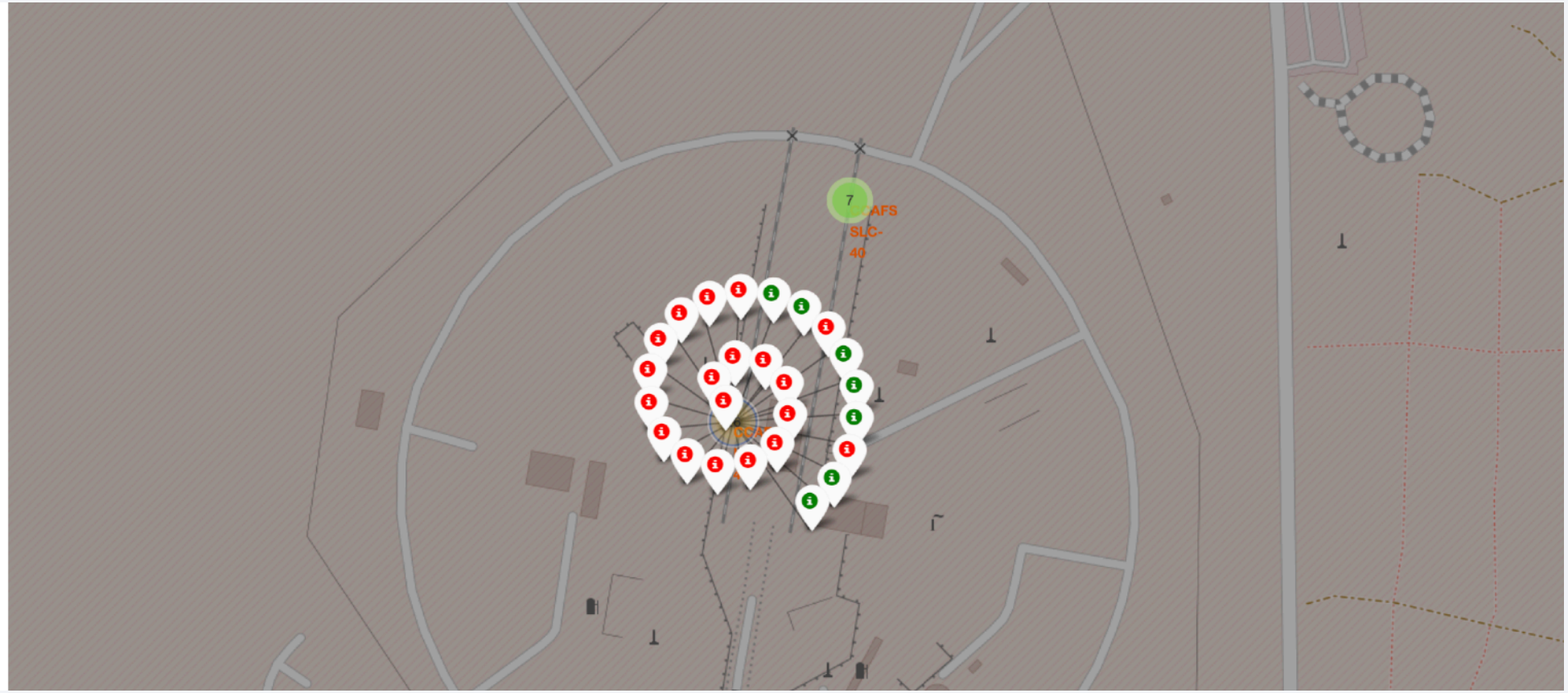
Launch sites in the world map



All launch sites shown in red labels

- Here we use folium.Marker to label the launch sites using the latitude and longitude values.

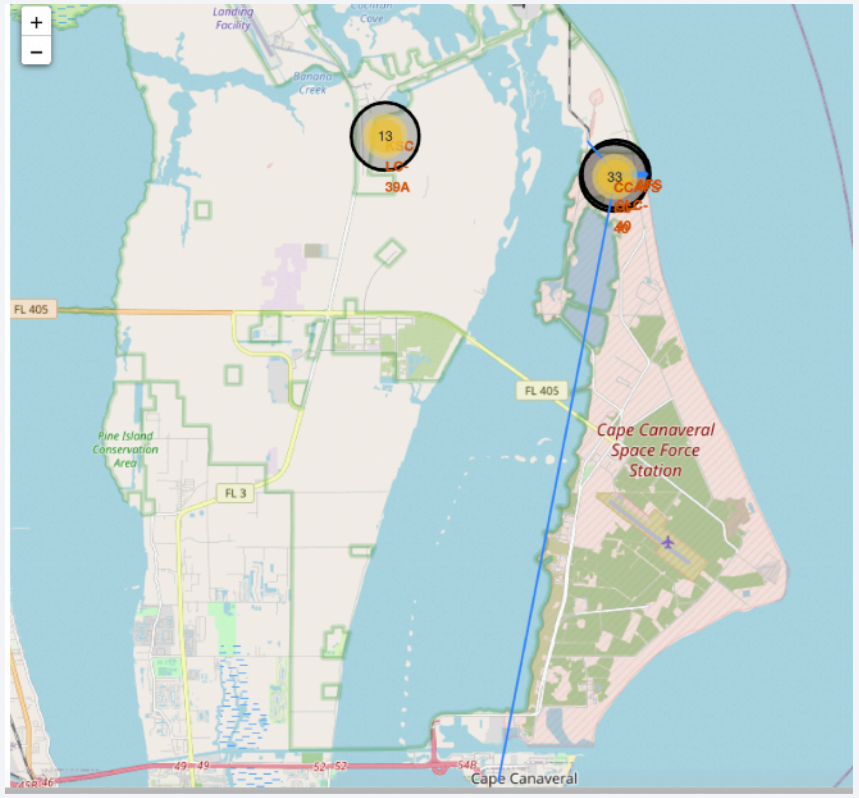
Success and failures for the CCAFS LC-40 launch site



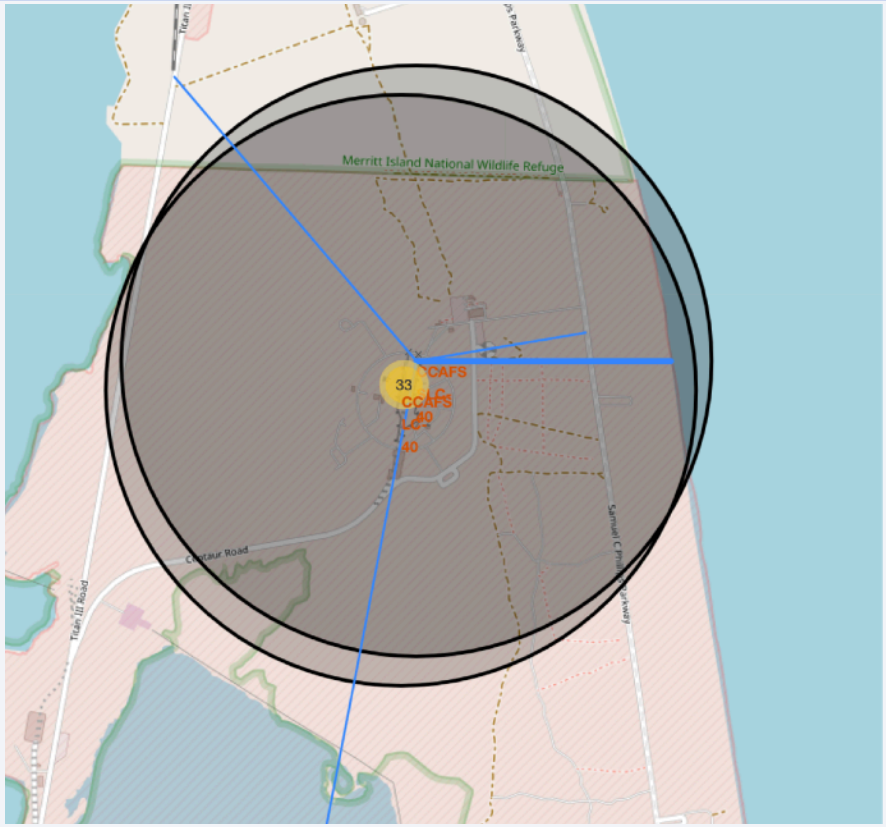
Success rate in green color and failures in red colours for the launch site CCAFS LC-40

- We can see that this launch site has high failures compared to successes.

Proximity blue lines to the CCAFS SLC-40 launch site

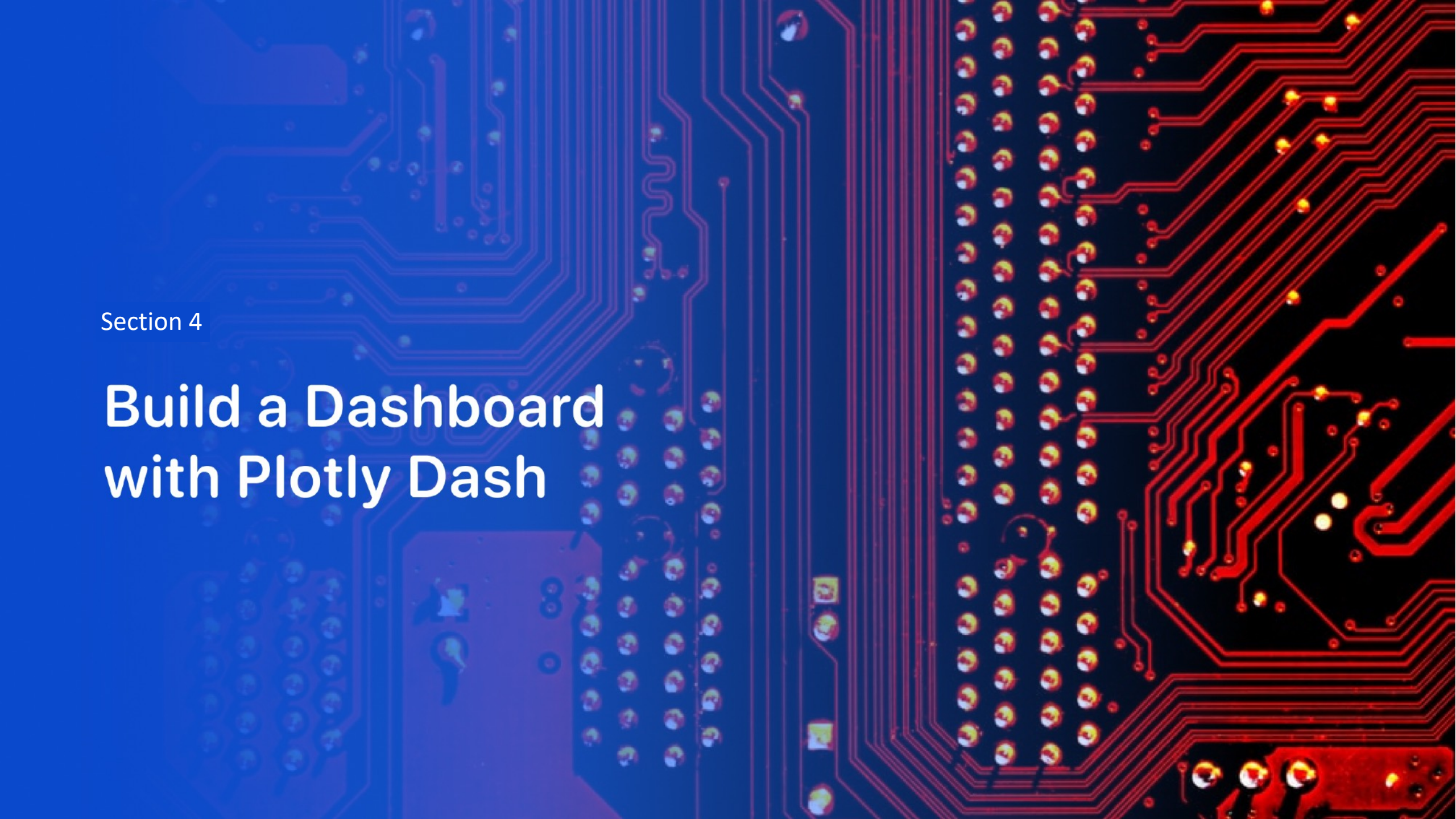


Blue line show the closest city Cape Canaveral from CCAFS SLC-40

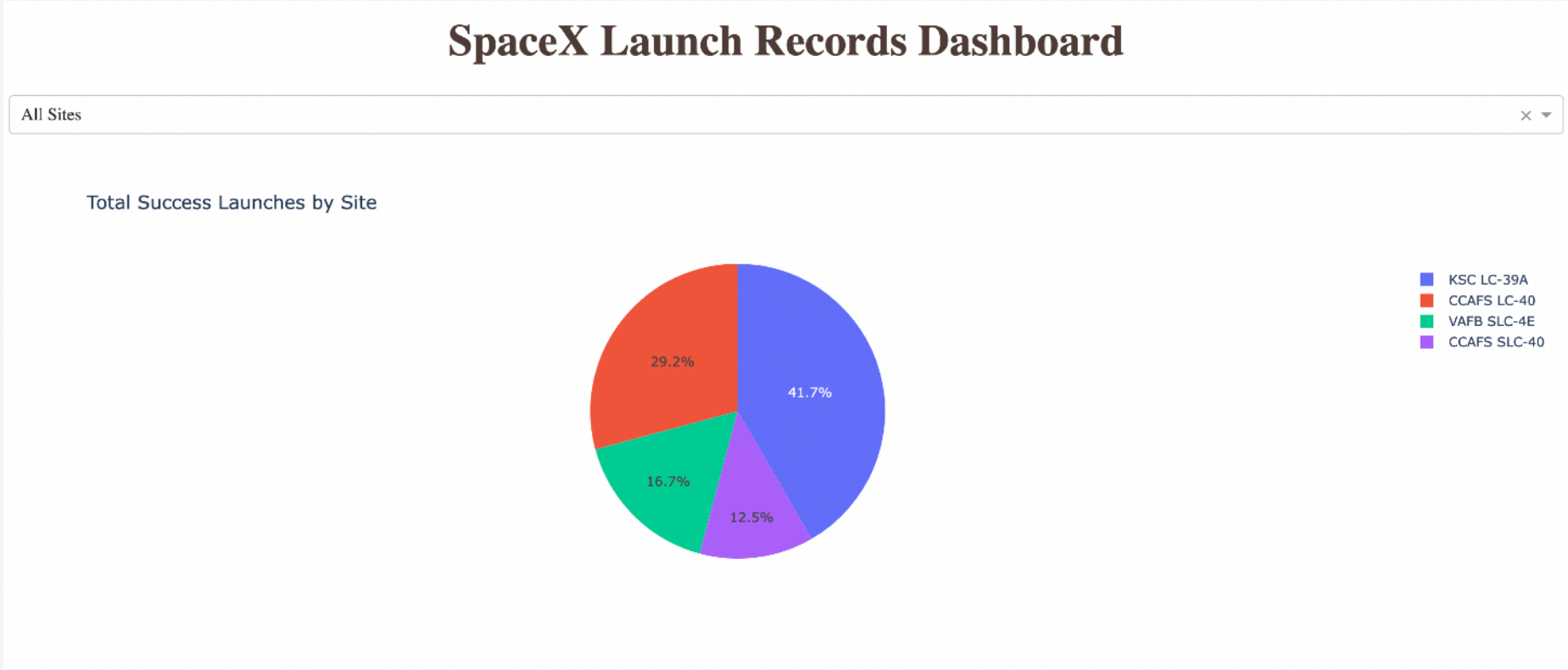


The closest railway track, highway and coastline from the same launch site

- Using folium.PolyLine, create the blue lines to the closest proximities


Section 4
Build a Dashboard with Plotly Dash

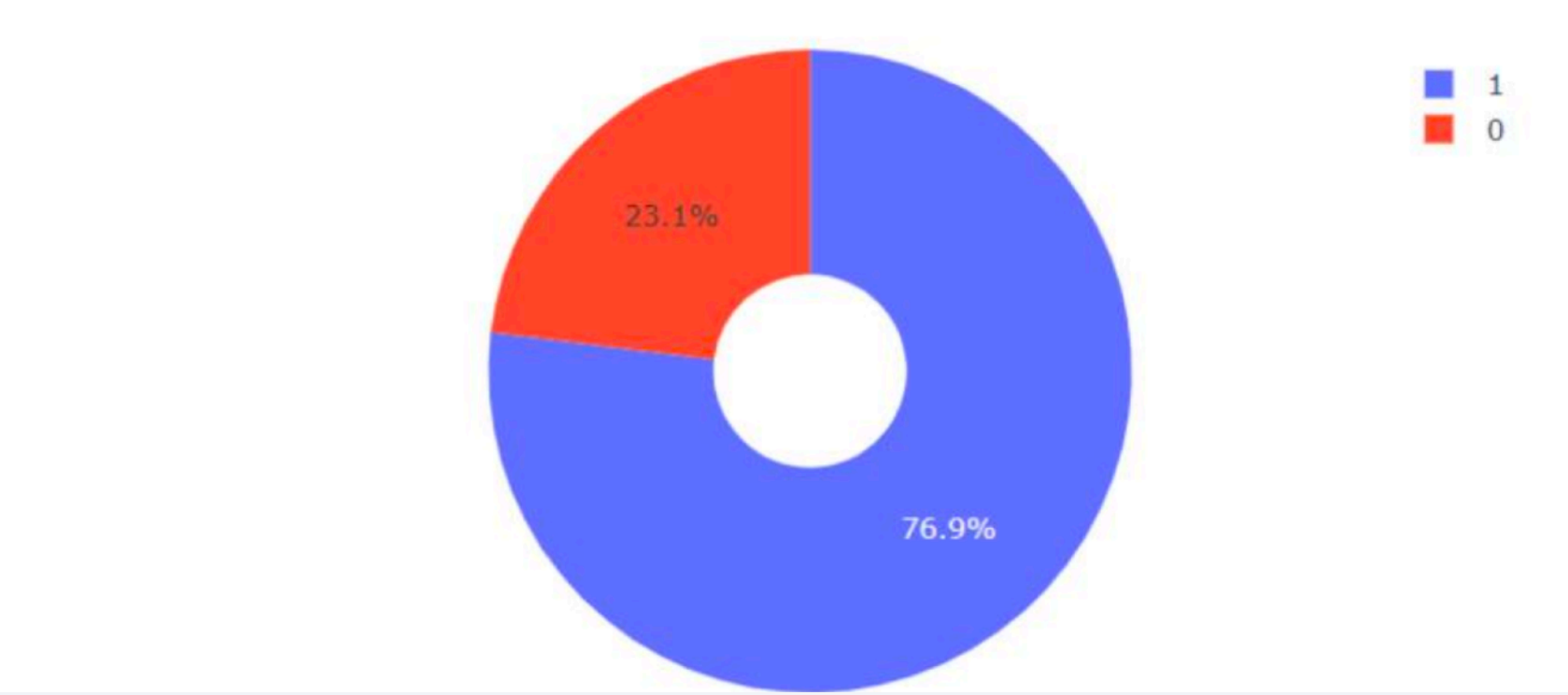
Pie chart with success rate for all launch sites



Caption

- KSC LC-39A has the most success rate compared to other sites.

Pie chart showing the success and failure for the KSC LC39-A launch site



Caption

- KSC LC-39A has the most success rate of 76.9 percent.

Payload vs launch outcome for all launch sites for a particular range of payload mass

Class vs payload with the whole payload mass range

- FT has the highest success rate for the whole payload range

Class vs payload with the payload mass range between 1000 and 7500 kg.

- FT has the highest success rate for the selected range

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- We use four models logistic regression, SVM, decision tree, and KNN.

- We find the decision tree as the best classifier with accuracy of 0.9027.

- The tuned hyper parameters are:

Criterion: ‘gini’

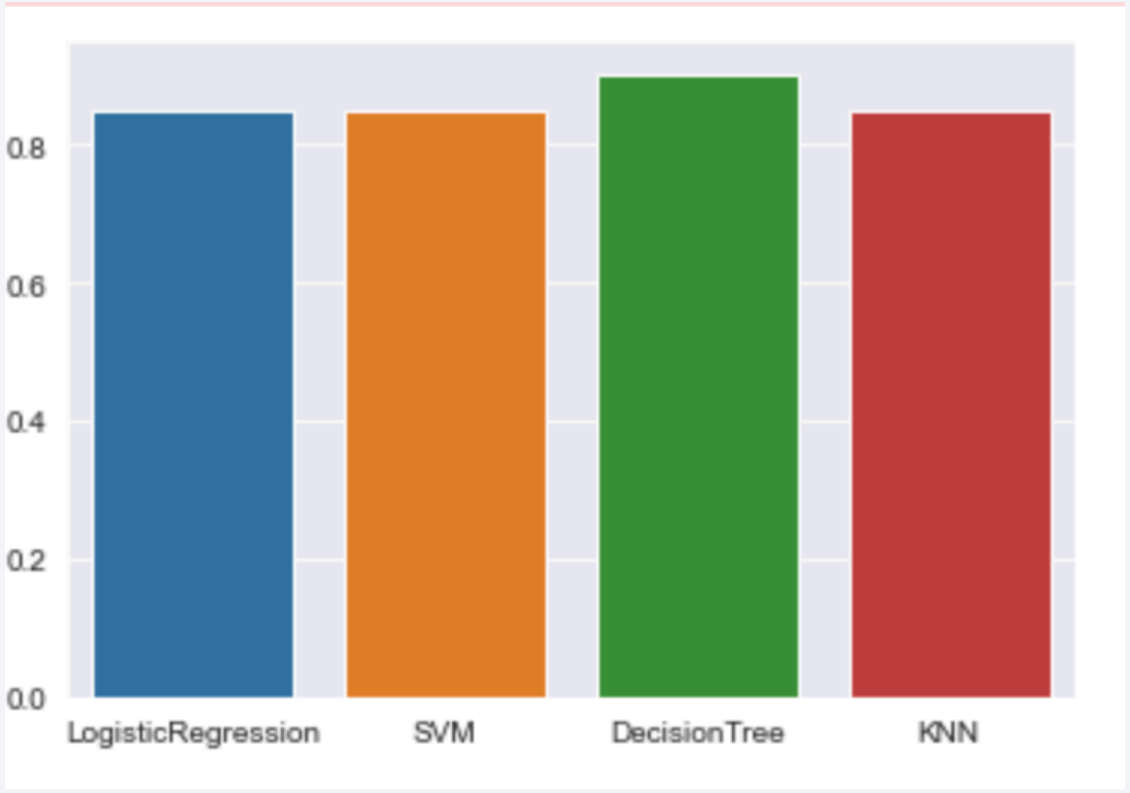
max_depth: 4

max_features: ‘auto’

min_samples_leaf: 1

min_samples_split: 5

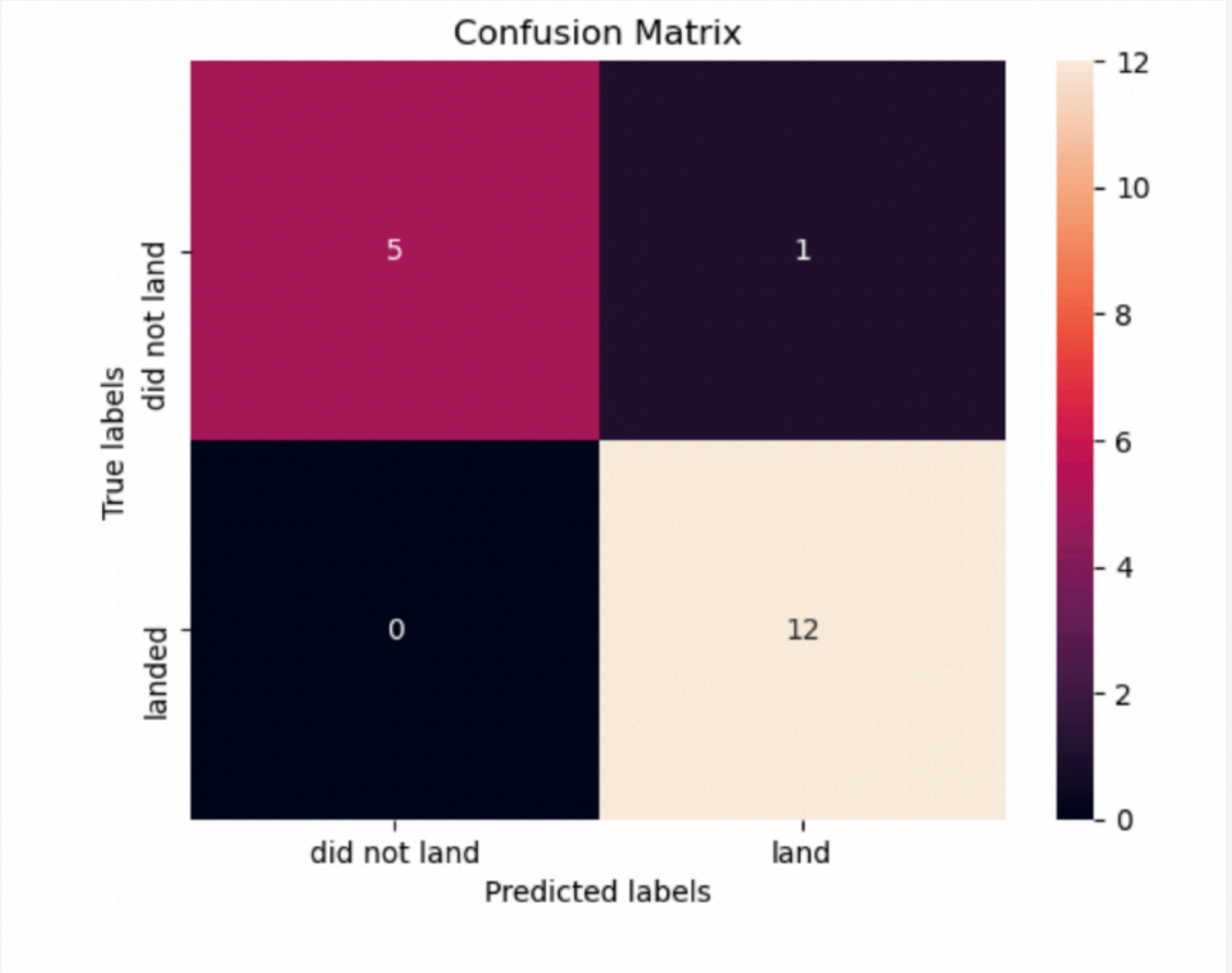
Splitter: ‘random’



Model accuracy with model

Confusion Matrix

- This classifier (Decision tree) classifies the labels accurately compared to other classifiers.
- Here True positive and True negative are high that means model is accurate and precise.



Confusion matrix for decision tree

Conclusions

- With flight number, the success rate increases
- Heavy payload increases the success rate
- LEO, VLEO, GEO, HEO, SSO orbits have the highest success rate
- Launch success rate increases suddenly from 2013 till 2020 gradually
- KSC LC-39A booster has the highest success rate.
- The Decision tree classifier has the best accuracy score compared to other models.

Thank you!