IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Name: Yashar Janparvar Javdani
Date:16-January-2023

Outline

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

Executive Summary

- Summary of methodologies
  1. Data Collection Rest API
  2. Data Collection
  3. Data Wrangling
  4. Exploratory Data Analysis with SQL
  5. Exploratory Data Analysis with Data Visualization
  6. Interactive Data Visualization with Folium and Ploty Dash
  7. Machine Learning Prediction
- Summary of all results
  1. Collect Data and Analysis result
  2. Drawing analytical charts
  3. Use Machine Learning algorithm and predict result

Introduction

- Project background and context

  SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards 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

- Problems you want to find answers

  I. The interaction amongst various features that determine the success rate of a successful landing.

  II. For a given set of features about a Falcon 9 rocket launch, will the first stage of the rocket land successfully?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API and web scraping List of SpaceX Falcon 9 launches wiki article
- Perform data wrangling
  - Assigning appropriate data types and dealing with NaN values appropriately
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Built several classification models and evaluated the accuracy of each to determine the best model

Data Collection

I. First get data from url with this method requests.get(spacex_url)

II. Convert raw data to json data with this method json_data=response.json()

III. After that normalized json data and converted to data frame data=pd.json_normalize(json_data)

IV. Extract Useful Columns and create dict

V. Create data from a dict with this method dataDict=pd.DataFrame(launch_dict)

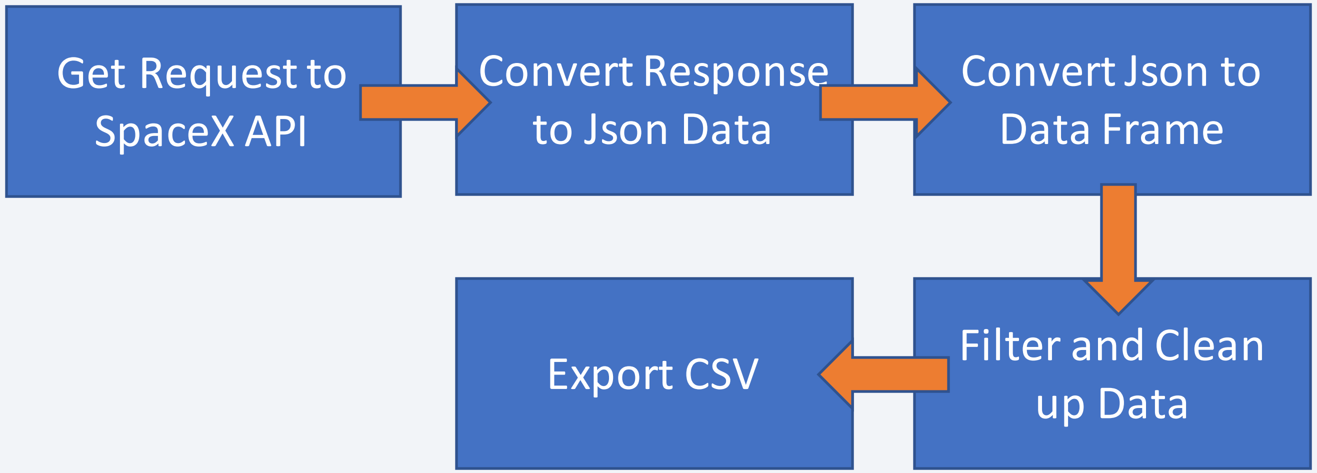
VI. Filter dataframe to only include Falcon 9 launches

VII. Calculate the mean value of PayloadMass column with this method x=data_falcon9['PayloadMass'].mean()

VIII. Handle missing values with this method data_falcon9['PayloadMass'].replace(np.nan,x, inplace=True)

IX. Export to CSV file

Data Collection – SpaceX API



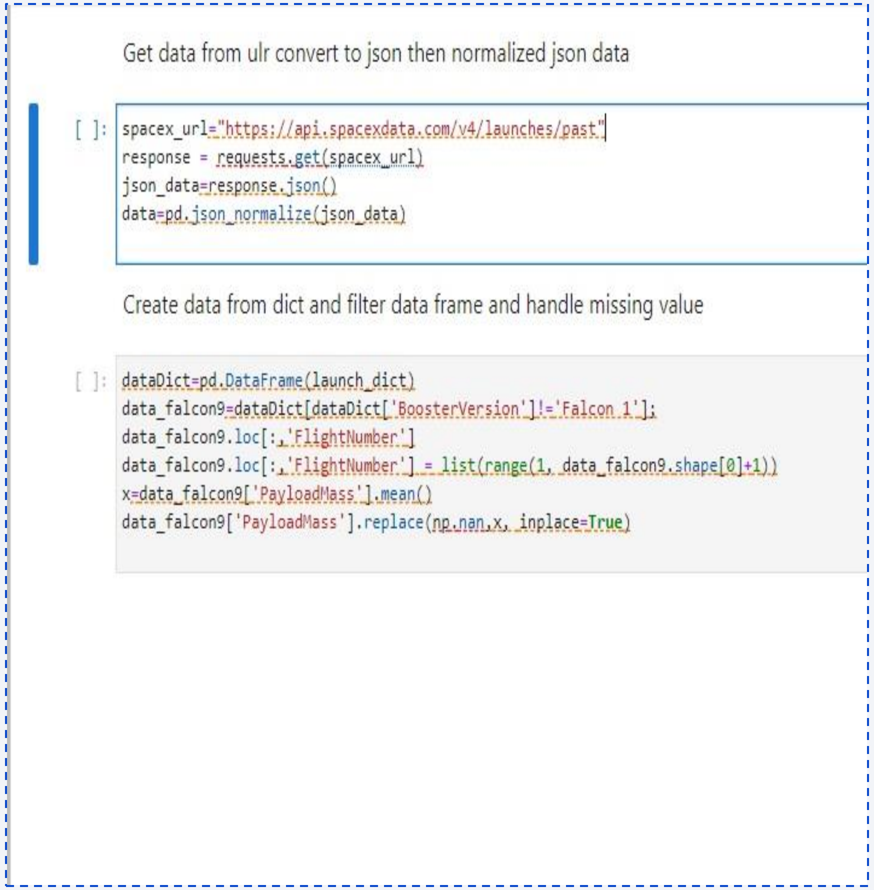
Get Request to SpaceX API

Convert Response to Json Data

Convert Json to Data Frame

Filter and Clean up Data

Export CSV



Get data from ulr convert to json then normalized json data

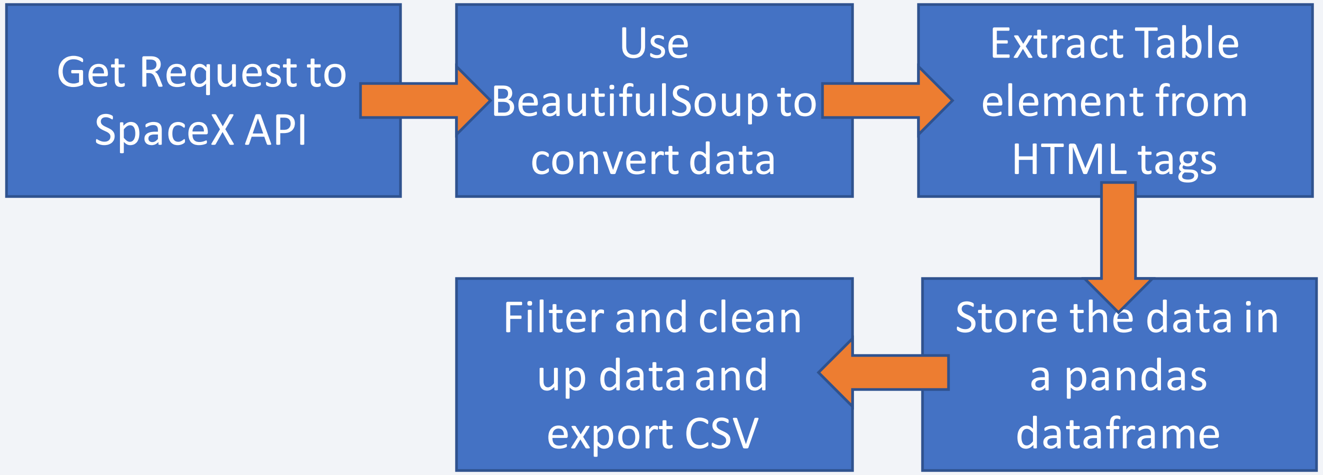
```
[ ]: spacex_url="https://api.spacexdata.com/v4/launches/past"
     response = requests.get(spacex_url)
     json_data=response.json()
     data=pd.json_normalize(json_data)
```

Create data from dict and filter data frame and handle missing value

```
[ ]: dataDict=pd.DataFrame(launch_dict)
     data_falcon9=dataDict[dataDict['BoosterVersion']!='Falcon 1'];
     data_falcon9.loc[:,'FlightNumber']
     data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
     x=data_falcon9['PayloadMass'].mean()
     data_falcon9['PayloadMass'].replace(np.nan,x, inplace=True)
```

- GitHub URL: Collecting the data

Data Collection - Scraping



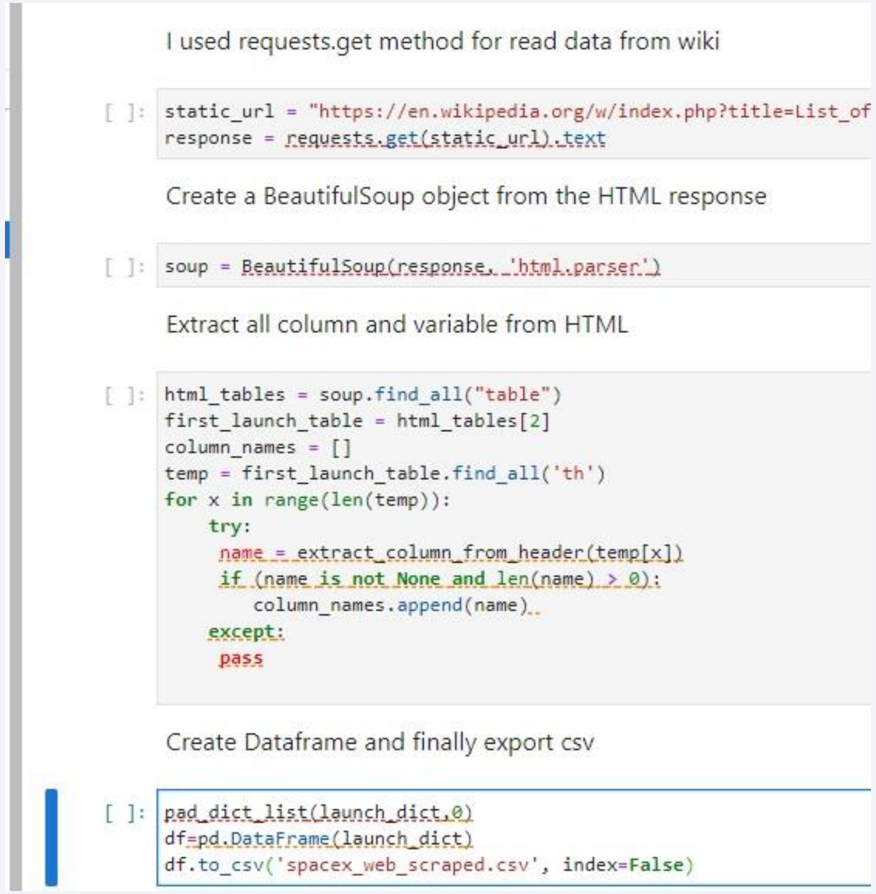
Get Request to SpaceX API

Use BeautifulSoup to convert data

Extract Table element from HTML tags

Store the data in a pandas dataframe

Filter and clean up data and export CSV



I used requests.get method for read data from wiki

```
[ ]: static_url = "https://en.wikipedia.org/w/index.php?title=List_of
     response = requests.get(static_url).text
```

Create a BeautifulSoup object from the HTML response

```
[ ]: soup = BeautifulSoup(response, 'html.parser')
```

Extract all column and variable from HTML

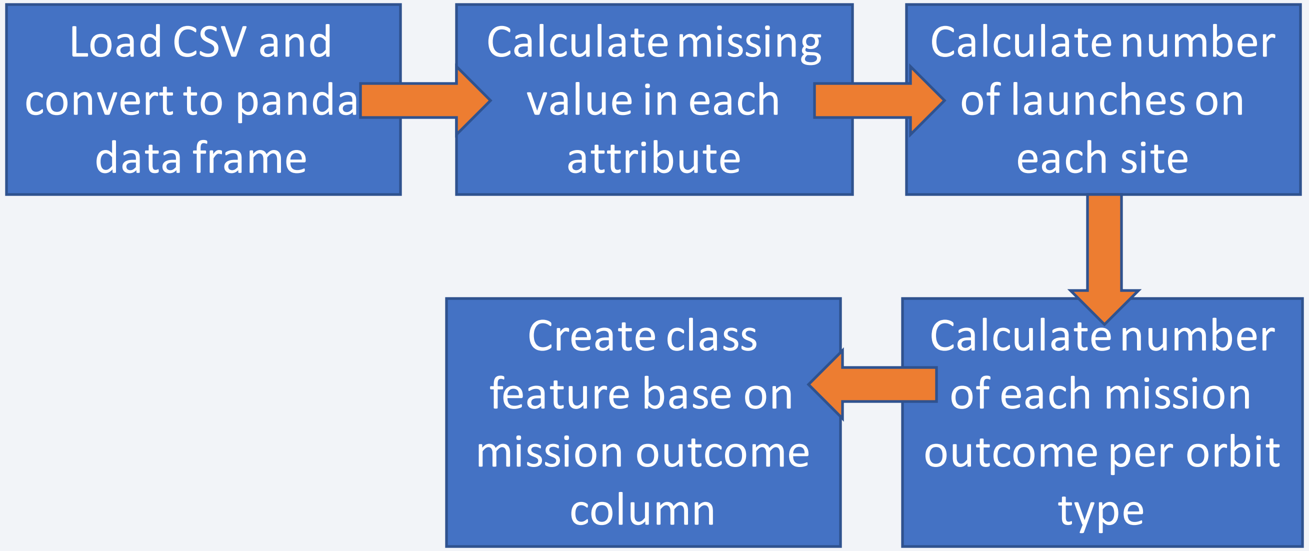
```
[ ]: html_tables = soup.find_all("table")
     first_launch_table = html_tables[2]
     column_names = []
     temp = first_launch_table.find_all('th')
     for x in range(len(temp)):
         try:
          name = extract_column_from_header(temp[x])
          if (name is not None and len(name) > 0):
             column_names.append(name)
         except:
          pass
```

Create Dataframe and finally export csv

```
[ ]: pad_dict_list(launch_dict,0)
     df=pd.DataFrame(launch_dict)
     df.to_csv('spacex_web_scraped.csv', index=False)
```

- GitHub URL: Data Collection -Scraping

Data Wrangling



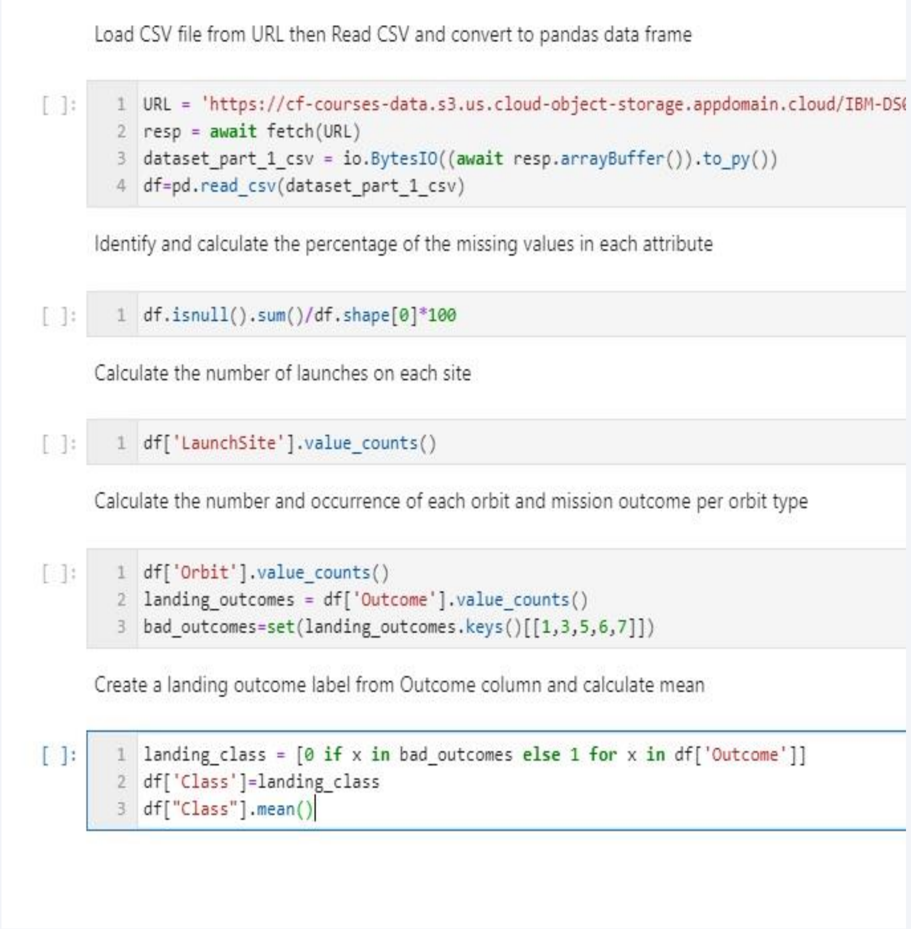
Load CSV and convert to panda data frame

Calculate missing value in each attribute

Calculate number of launches on each site

Calculate number of each mission outcome per orbit type

Create class feature base on mission outcome column



Load CSV file from URL then Read CSV and convert to pandas data frame

```
[ ]: URL = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS
     resp = await fetch(URL)
     dataset_part_1_csv = io.BytesIO((await resp.arrayBuffer()).to_py())
     df=pd.read_csv(dataset_part_1_csv)
```

Identify and calculate the percentage of the missing values in each attribute

```
[ ]: df.isnull().sum()/df.shape[0]*100
```

Calculate the number of launches on each site

```
[ ]: df['LaunchSite'].value_counts()
```

Calculate the number and occurrence of each orbit and mission outcome per orbit type

```
[ ]: df['Orbit'].value_counts()
     landing_outcomes = df['Outcome'].value_counts()
     bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
```

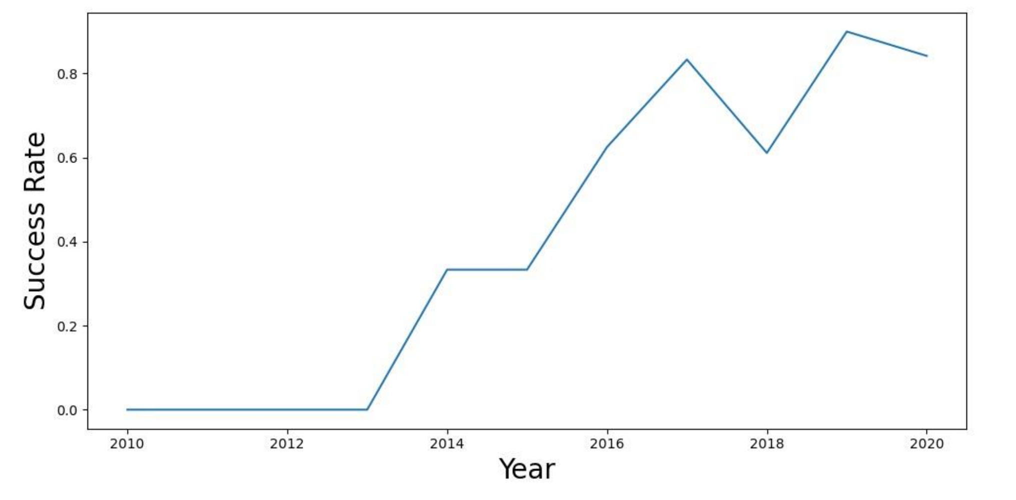
Create a landing outcome label from Outcome column and calculate mean

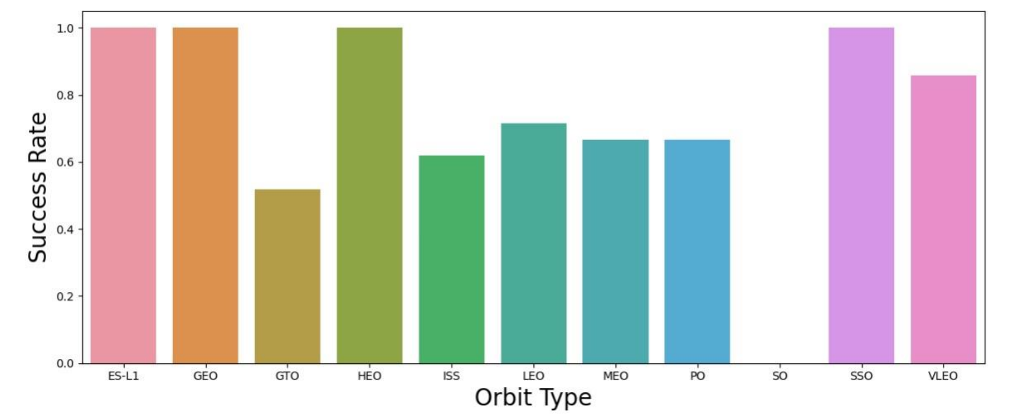
```
[ ]: landing_class = [0 if x in bad_outcomes else 1 for x in df['Outcome']]
     df['Class']=landing_class
     df["Class"].mean()
```

- GitHub URL: Data wrangling

EDA with Data Visualization

- Scatter plots: Scatter plots were used to represent the relationship between two variables. Different sets of features were compared such as Flight Number vs. Launch Site, Payload vs. Launch Site, Flight Number vs. Orbit Type and Payload vs. Orbit Type.
- Landing success rate versus Orbit type – to see if orbit type influenced the landing success rate
- Bar chart: Bar charts were used makes it easy to compare values between multiple groups at a glance. The x-axis represents a category and the y-axis represents a discrete value. Bar charts were used to compare the Success Rate for different Orbit Types
- GitHub URL: Exploring and Preparing Data





EDA with SQL

- SELECT distinct(LAUNCH_SITE) from SPACEXTBL
  - Display unique launch site in the space mission
- SELECT * from SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
  - Display 5 records where launch sites begin with the string 'CCA'
- SELECT SUM(PAYLOAD_MASS__KG_) from SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)'
  - Display the total payload mass carried by boosters launched by NASA (CRS)
- SELECT DATE, "Landing _Outcome", BOOSTER_VERSION, LAUNCH_SITE from SPACEXTBL WHERE "Landing _Outcome" = 'Failure (drone ship)' AND substr(Date,7,4) LIKE '%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(use substr because of sqllite)
- GitHub URL: EDA SQL

Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- Red and green marker clusters were added to indicate successful and failed landings at each launch site
- Line markers were added to indicate distance of launch site from coastline, nearest city, highway, and railway
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
  - Are launch sites near railways, highways and coastlines.
  - Do launch sites keep certain distance away from cities.
- GitHub URL: Launch Sites Locations Analysis with Folium

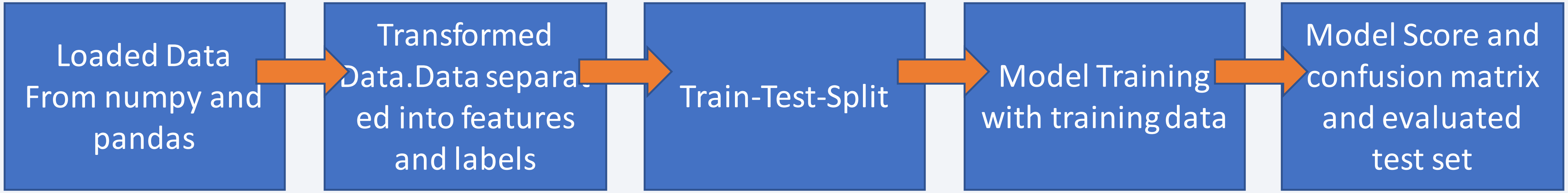
Build a Dashboard with Plotly Dash

- Pie chart showing success rate at the user chosen site. If all sites were picked, then the percentage of total successes at each site is shown as a pie chart – to quickly visualize success rates at each site

- The dashboard takes two inputs, namely the site(s) and payload mass. User input payload mass range slider – user can change the range of payload mass for which they can see the plot of success class versus payload mass graph (for the user chosen launch site), color coded by booster version – to see if class can be separated at a critical payload mass

- GitHub URL: Ploty Dash

Predictive Analysis (Classification)



Loaded Data From numpy and pandas

Transformed Data.Data separated into features and labels

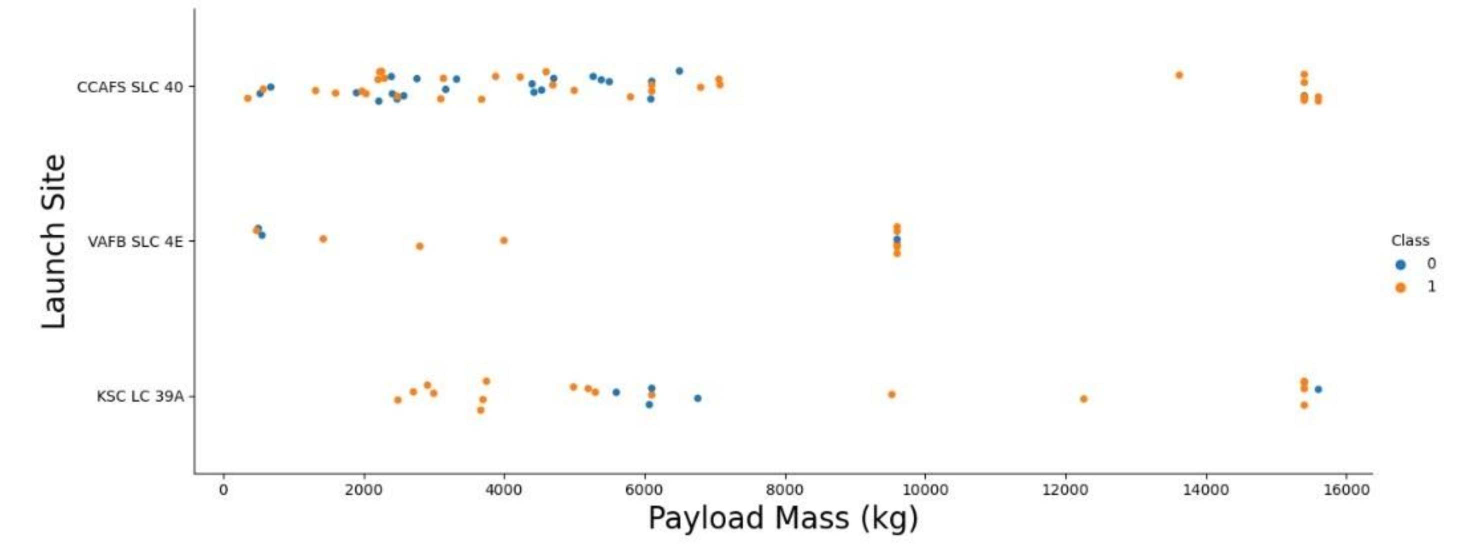
Train-Test-Split

Model Training with training data

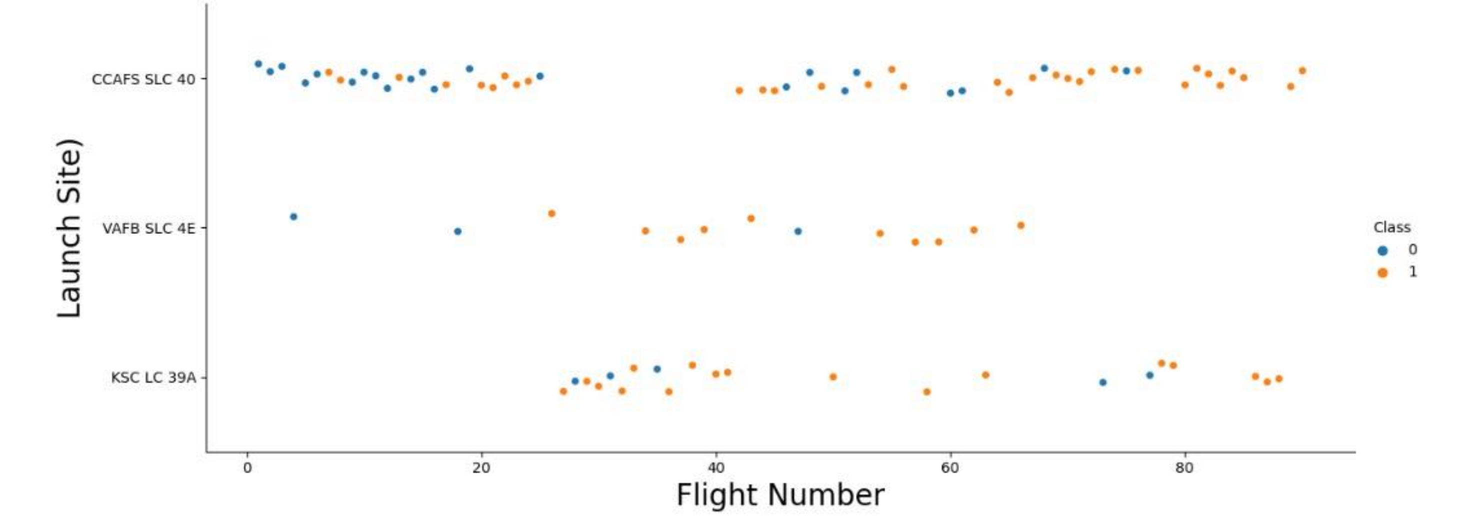
Model Score and confusion matrix and evaluated test set

- All above steps is repeated for all classification model.
- We built different machine learning models and tune different hyperparameters using GridSearchCV
- We used This Classification Model: Logistic Regression, Support Vector Machine, Decision Tree, K Nearest Neighbours
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- GitHub URL: Machine Learning Prediction

Results EDA

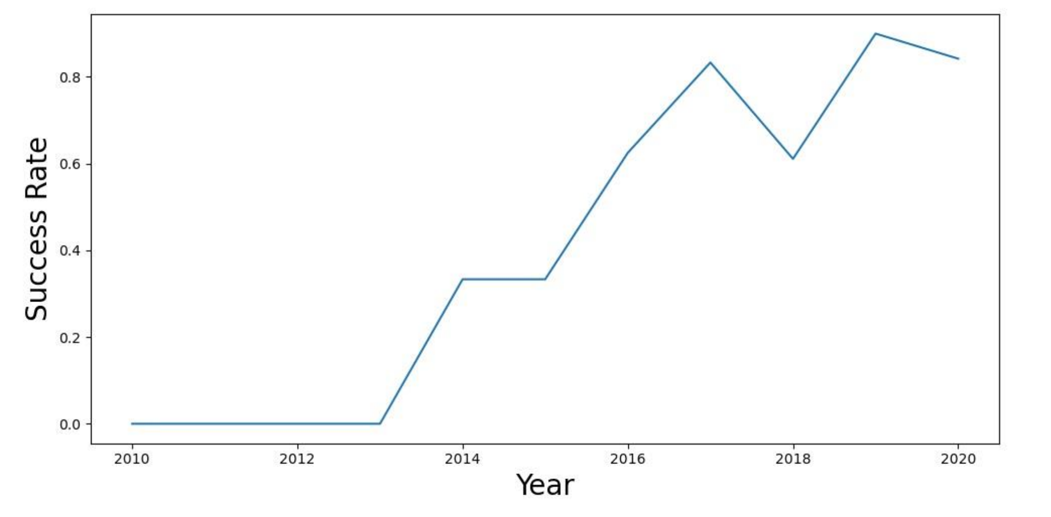


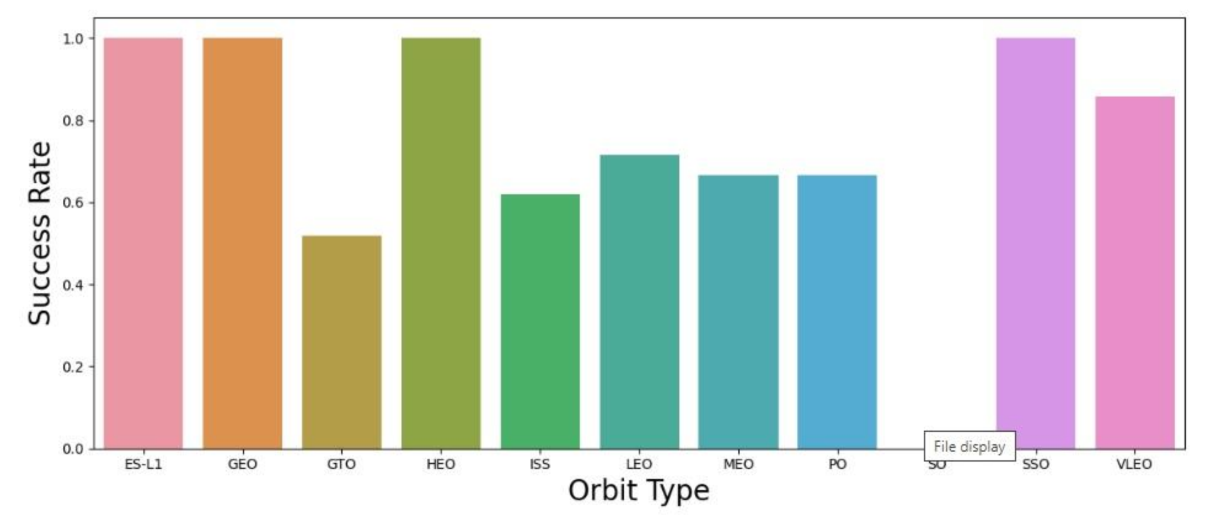
CCAFS SLC 40
VAFB SLC 4E
KSC LC 39A
Launch Site
0
2000
4000
6000
8000
10000
12000
14000
16000
Payload Mass (kg)
Class
0
1

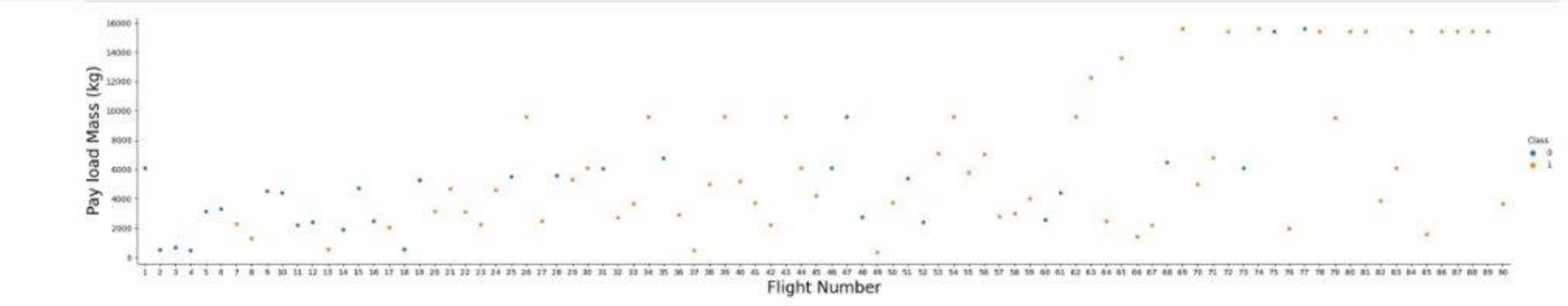


CCAFS SLC 40
VAFB SLC 4E
KSC LC 39A
Launch Site)
0
20
40
60
80
Flight Number
Class
0
1

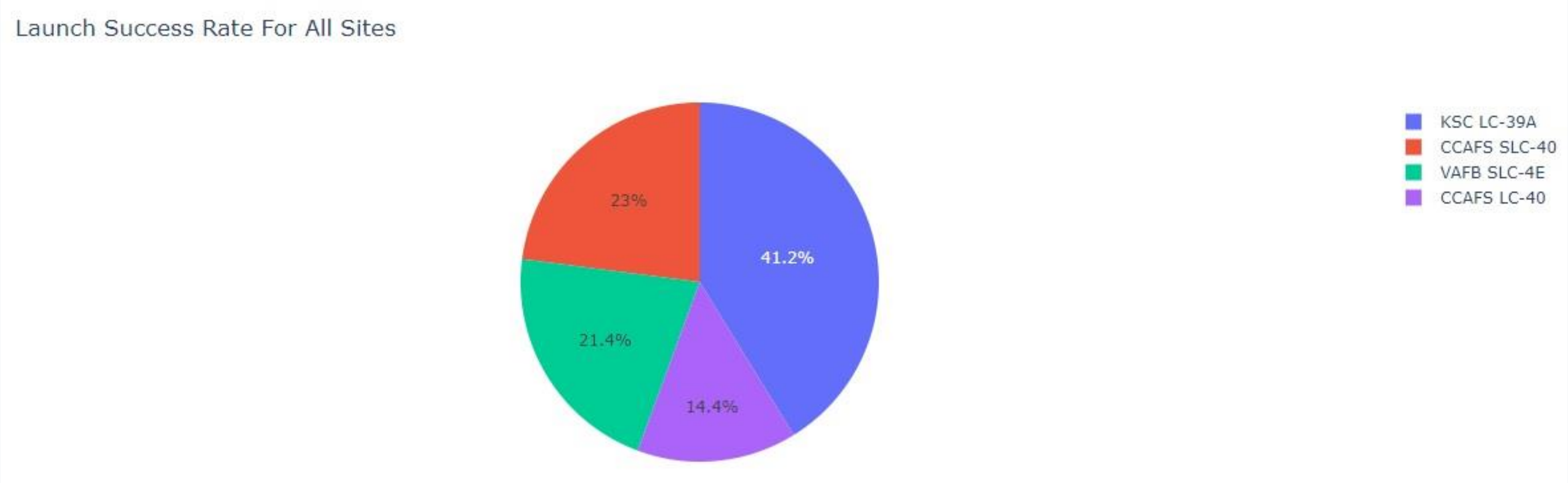
Results EDA

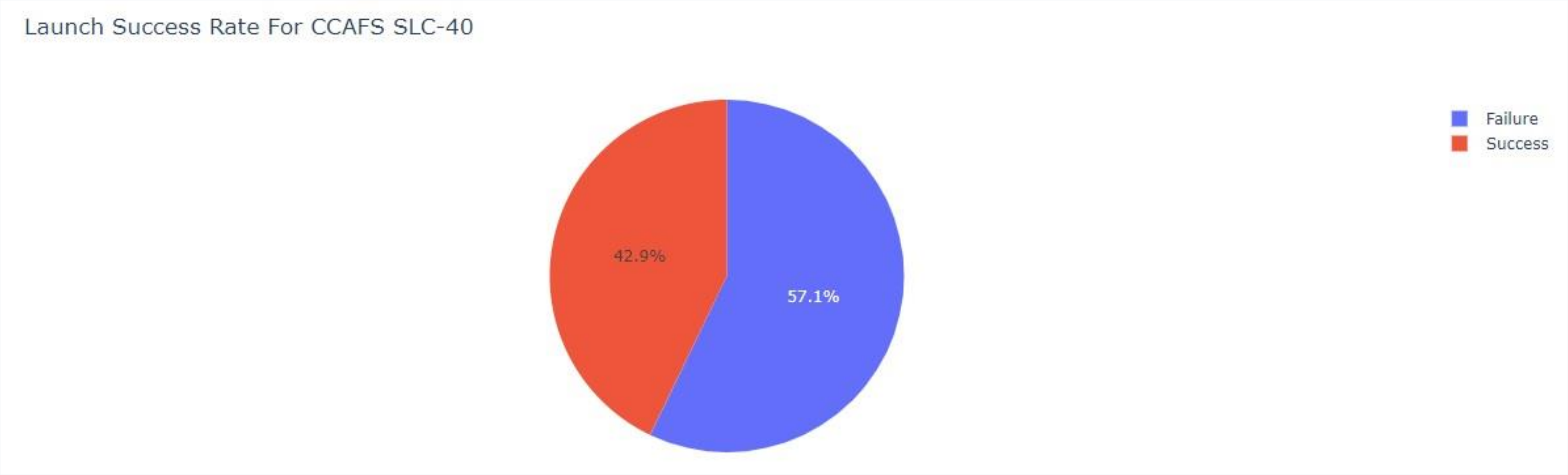

Success Rate
0.0
0.2
0.4
0.6
0.8
2010
2012
2014
2016
2018
2020
Year


Success Rate
0.0
0.2
0.4
0.6
0.8
1.0
ES-L1
GEO
GTO
HEO
ISS
LEO
MEO
PO
SO
SSO
VLEO
Orbit Type
File display

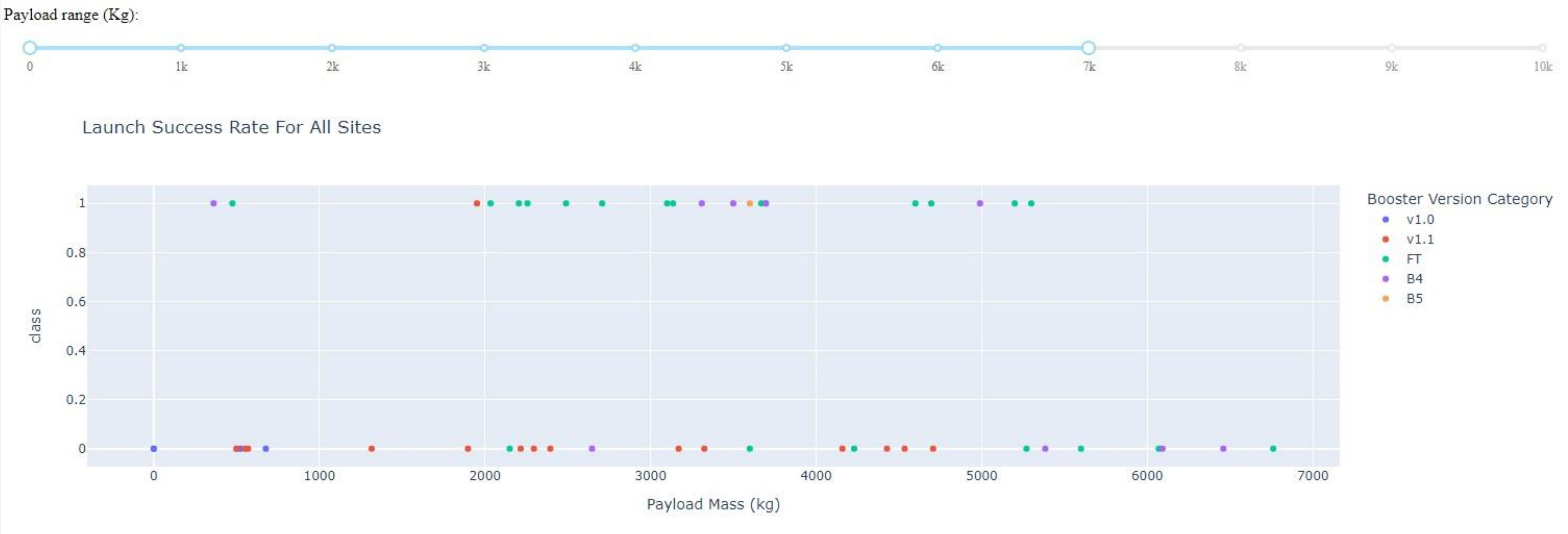

Pay load Mass (kg)
0
2000
4000
6000
8000
10000
12000
14000
16000
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90
Flight Number
Class
0
1

Results Dashboard

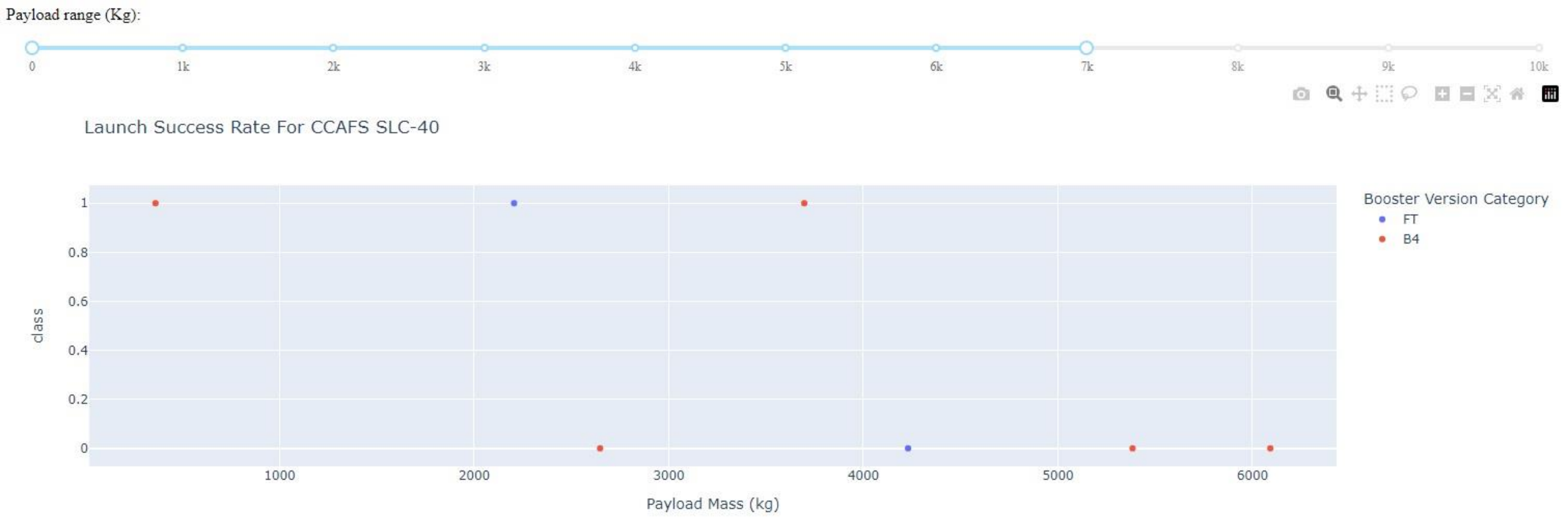

Launch Success Rate For All Sites


Launch Success Rate For CCAFS SLC-40

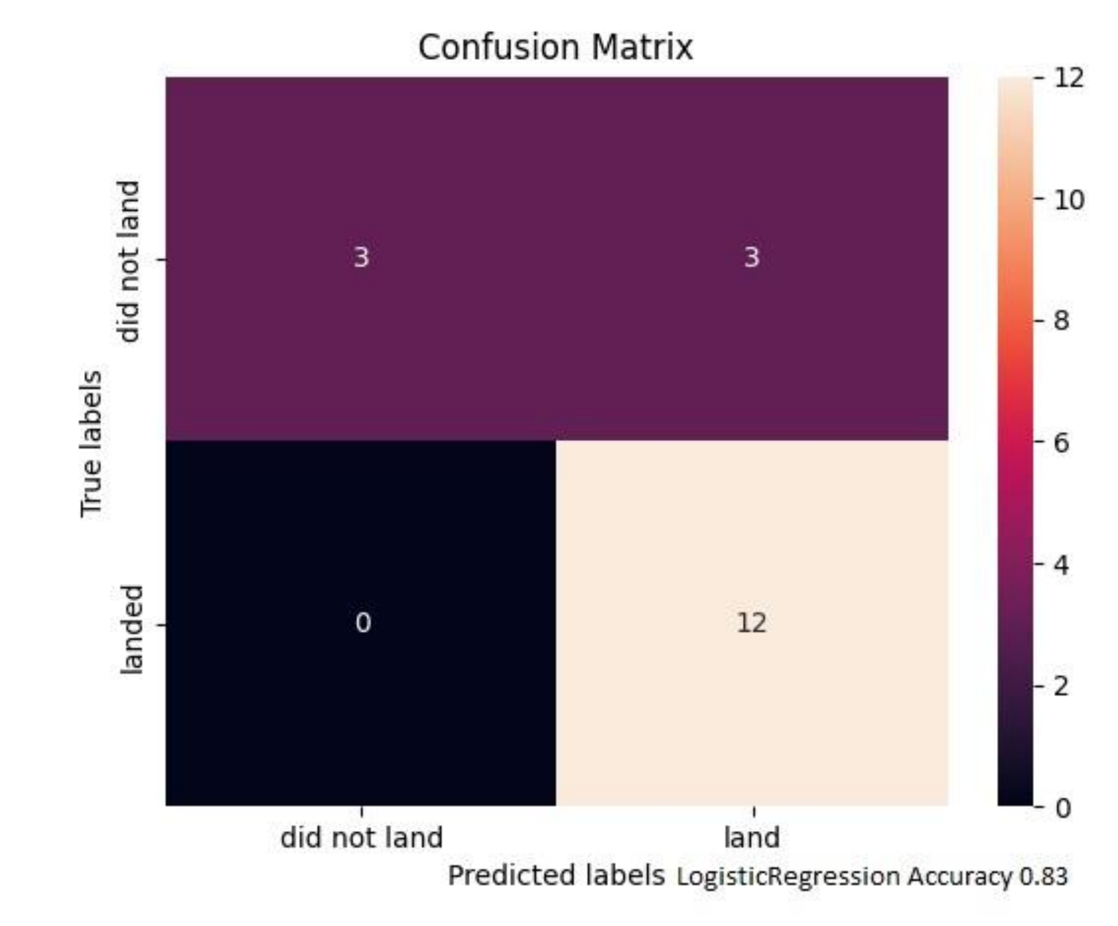
Results Dashboard



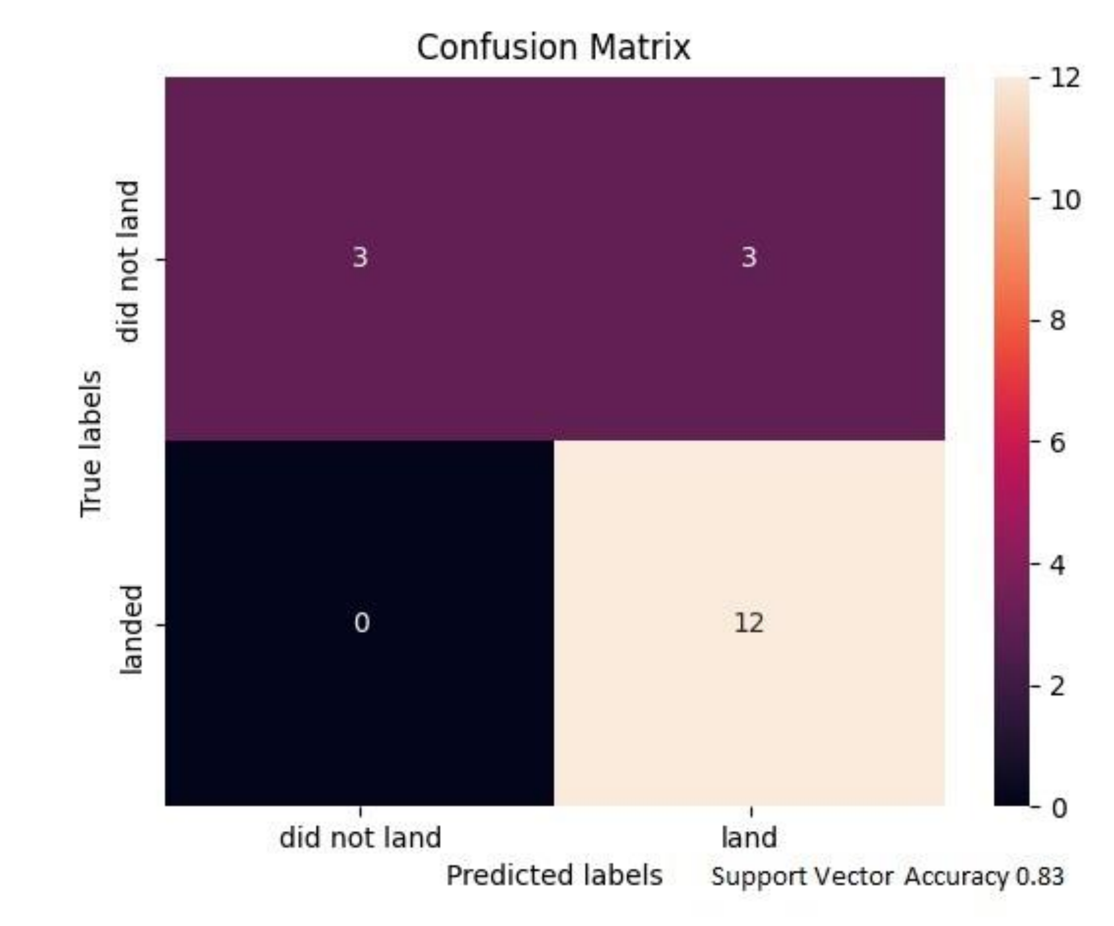
Results Dashboard



Results Predictive Analysis

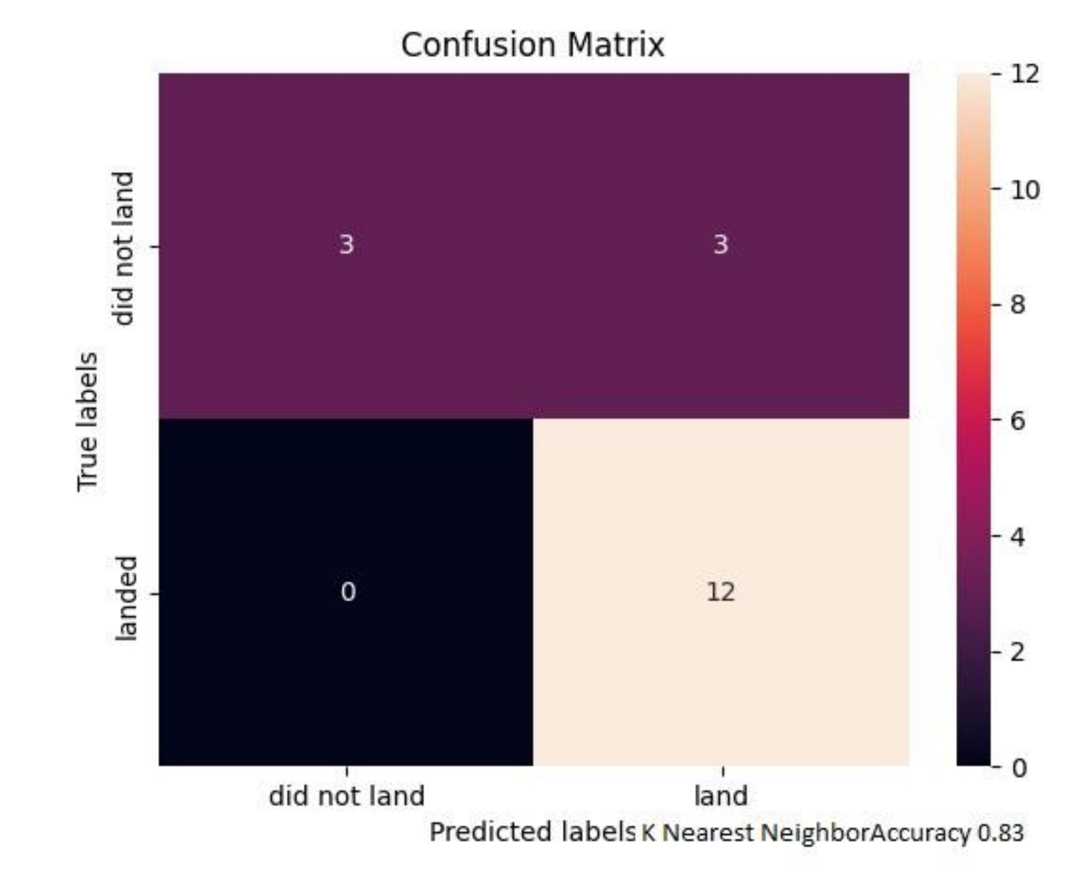


LogisticRegression Accuracy 0.83

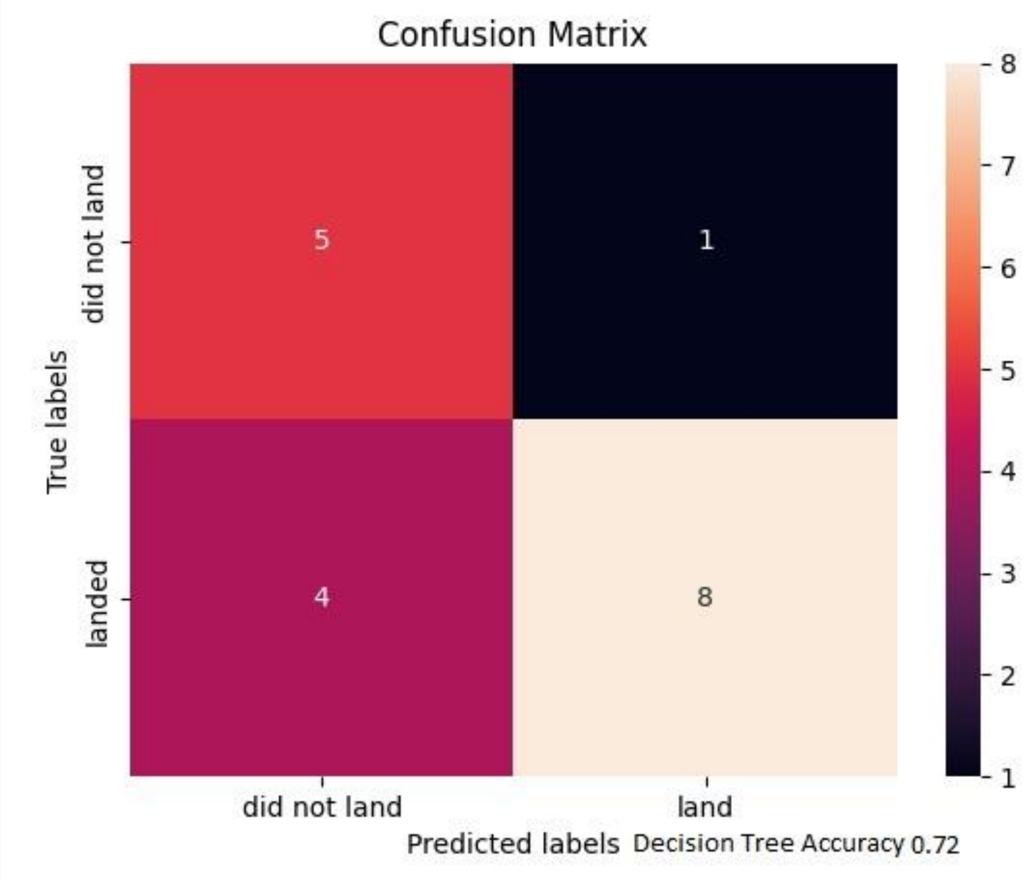


Support Vector Accuracy 0.83

Results Predictive Analysis



K Nearest NeighborAccuracy 0.83

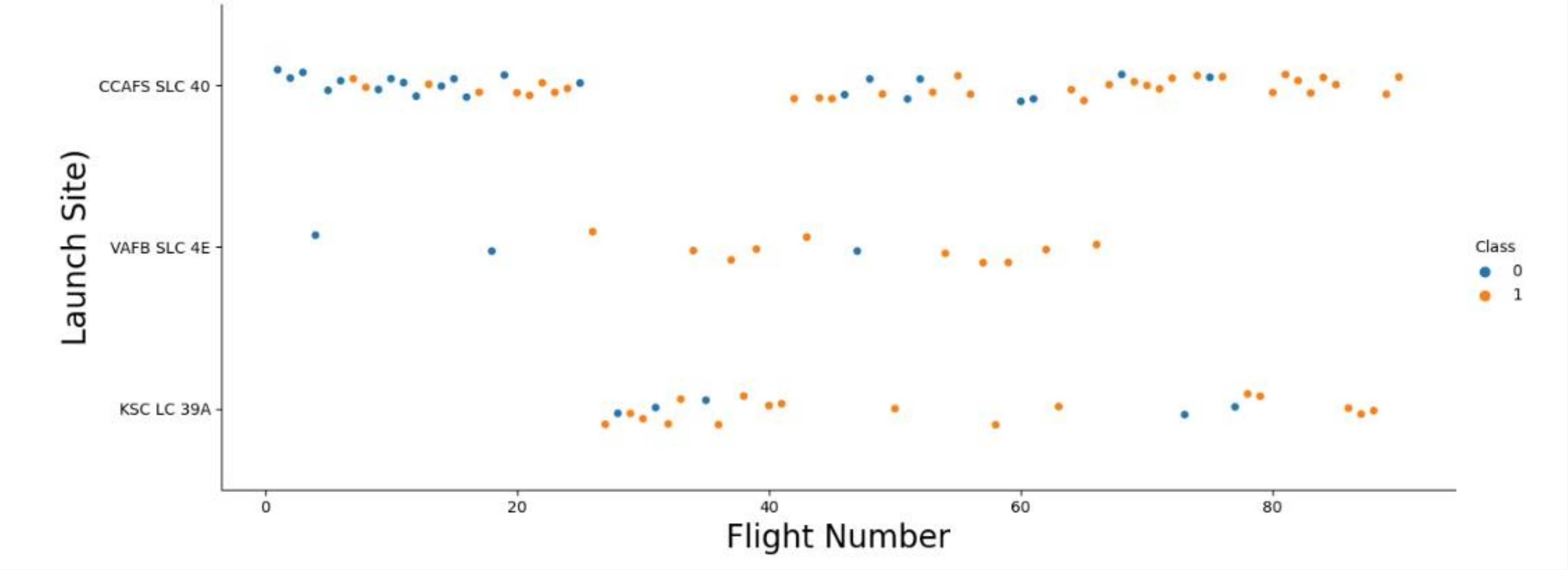


Decision Tree Accuracy 0.72

Section 2

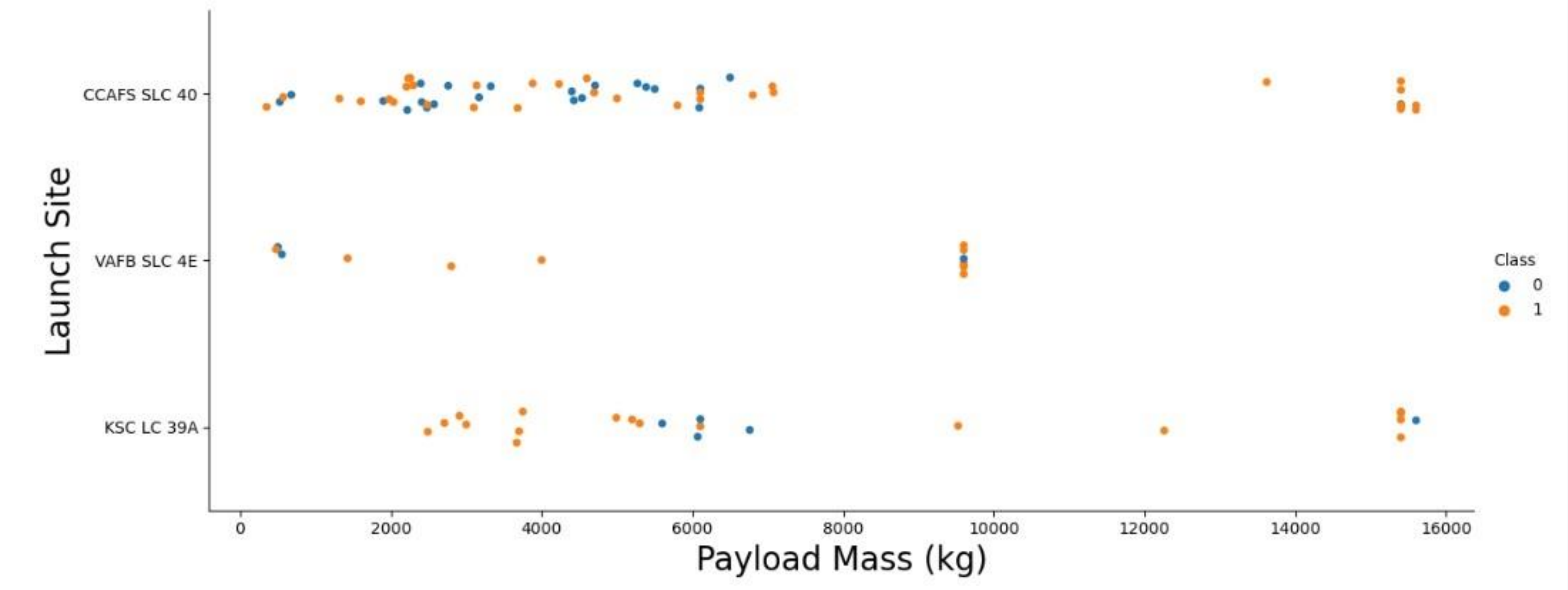
Insights drawn from EDA

Flight Number vs. Launch Site



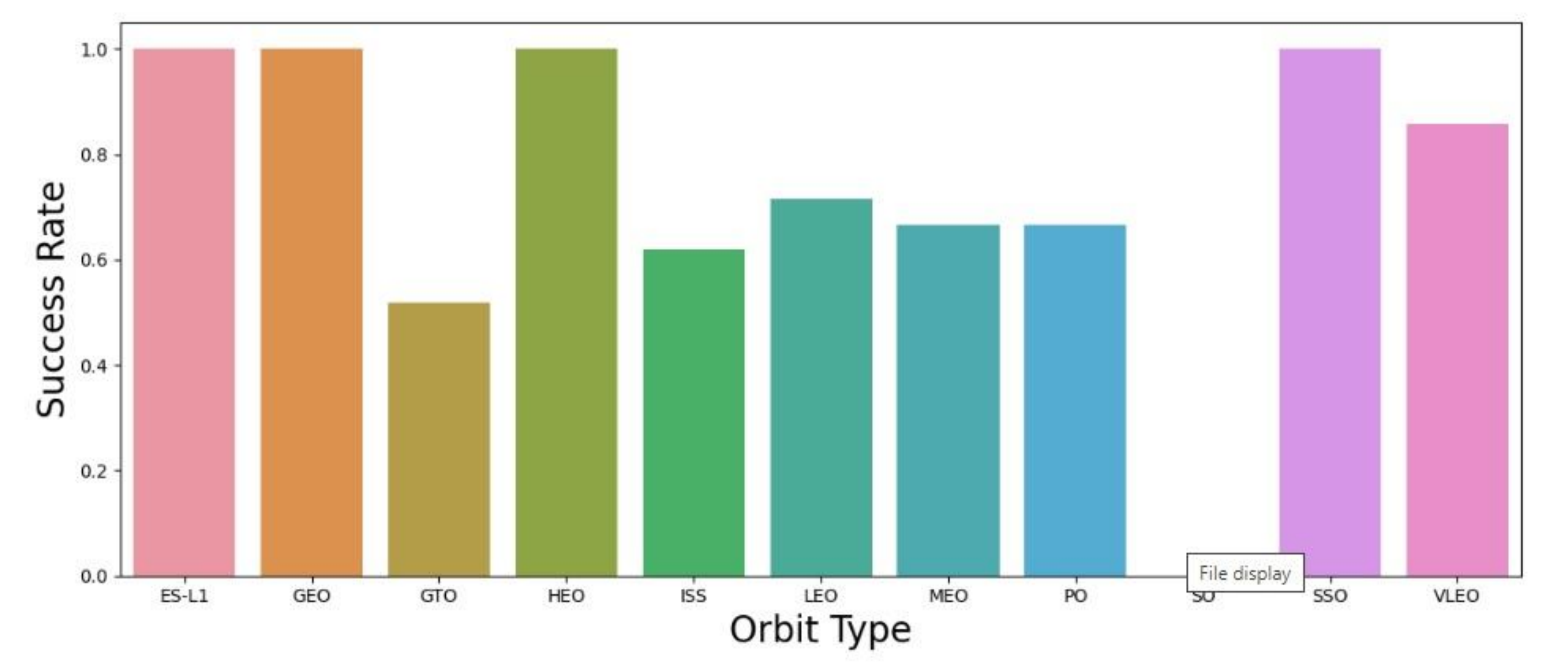
- CCAFS SLC-40 site was not used for flight numbers between 25 and 40 – KSC LC 39A was used for those numbers instead
- This figure shows that the success rate increased as the number of flights increased.
- VAFB SLC 4E is the least used launch site whereas CCAFS SLC 40 is the most used launch site
- There seems to be an increase in successful flights after the 40th launch.

Payload vs. Launch Site



- At VAFB SLC 4E the payload mass is always less than 10k
- The blue dots represent the successful launches while the red dots represent unsuccessful launches.

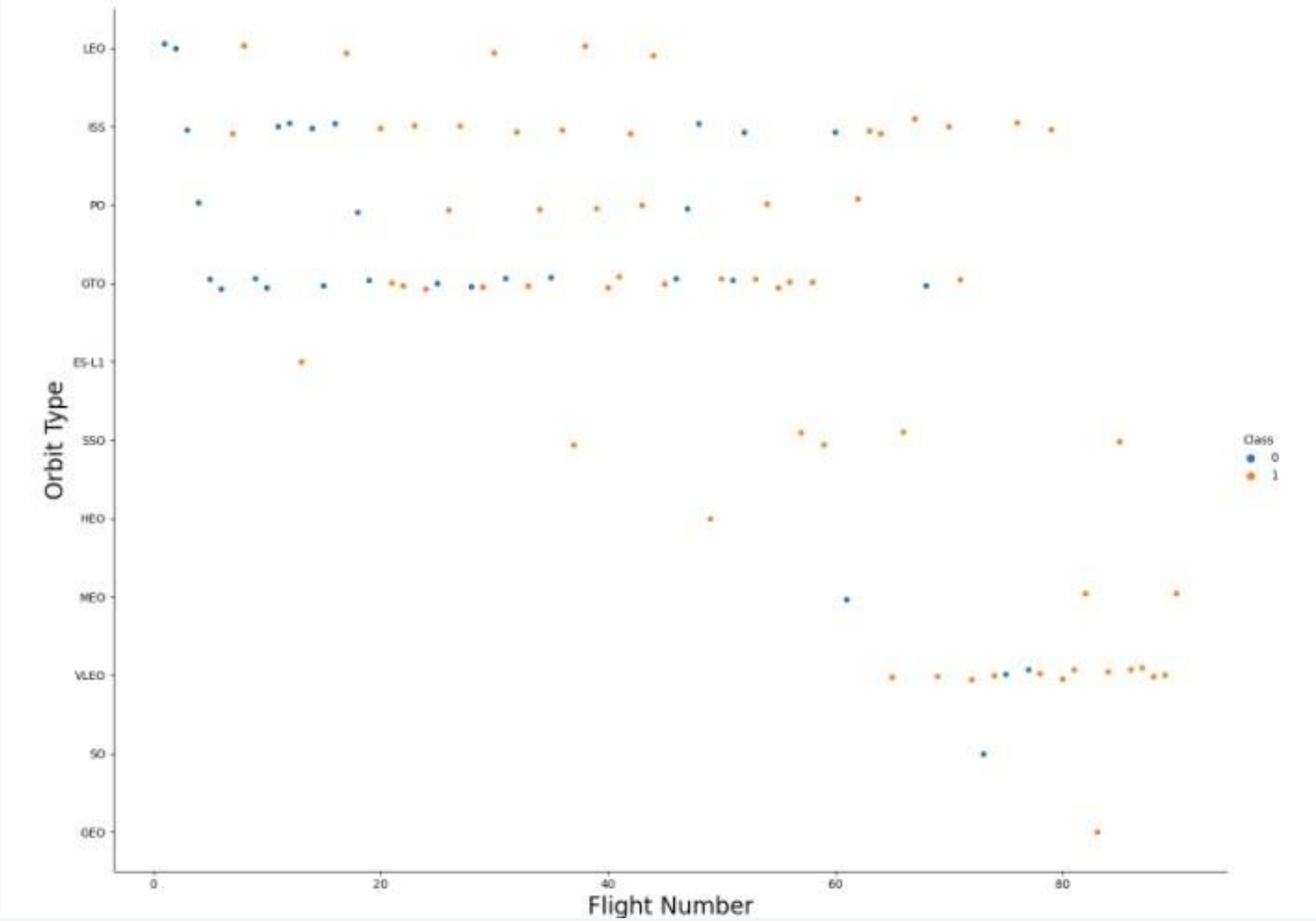
Success Rate vs. Orbit Type



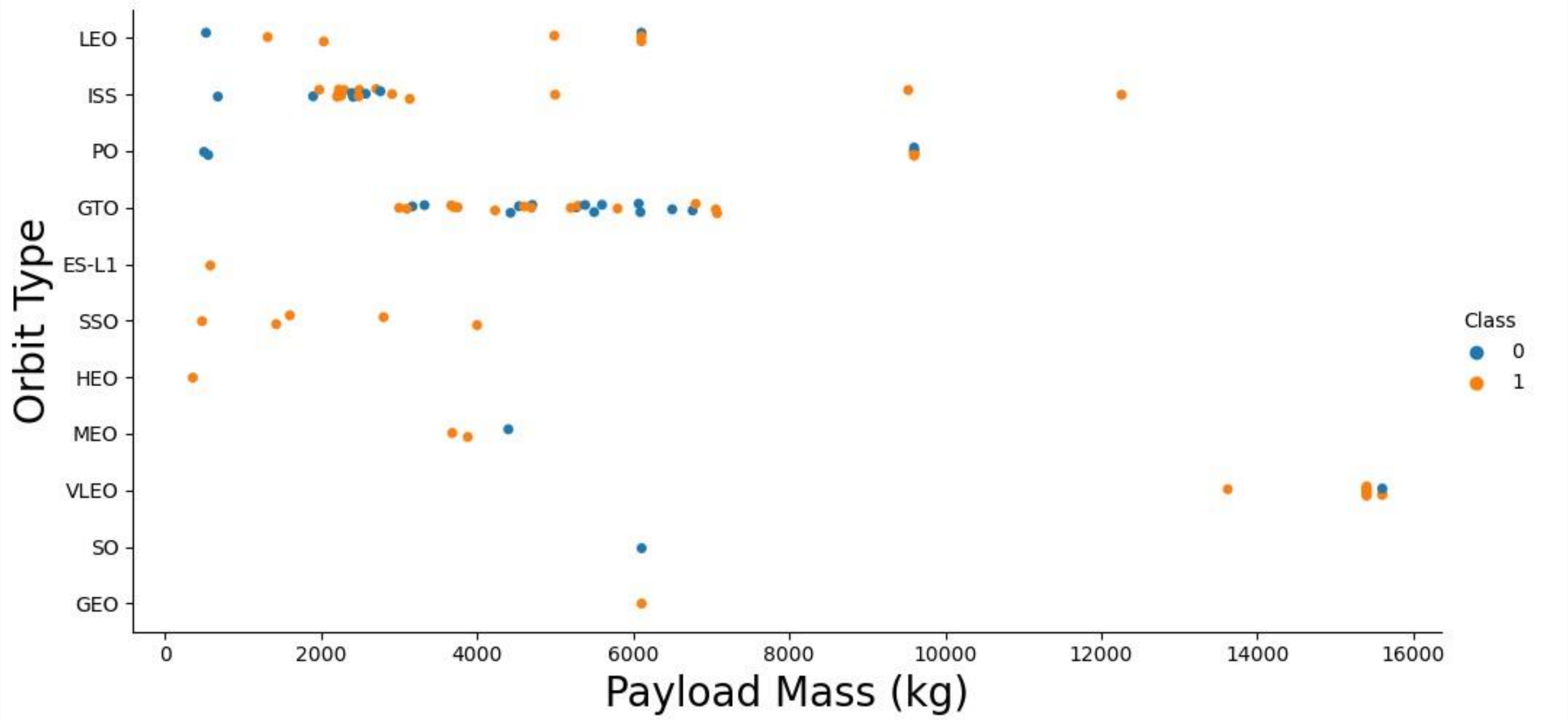
- From the plot, we can see that ES L1, GEO, HEO, SSO, VLEO had the most success rate.
- SO orbit did not have any successful launches with a 0% success rate.

Flight Number vs. Orbit Type

- In the LEO orbit, the success is positively correlated to the number of flights.
- The SSO orbit has a 100% success rate however with fewer flights than the other orbits
- ISS and GTO are more popular than the rest

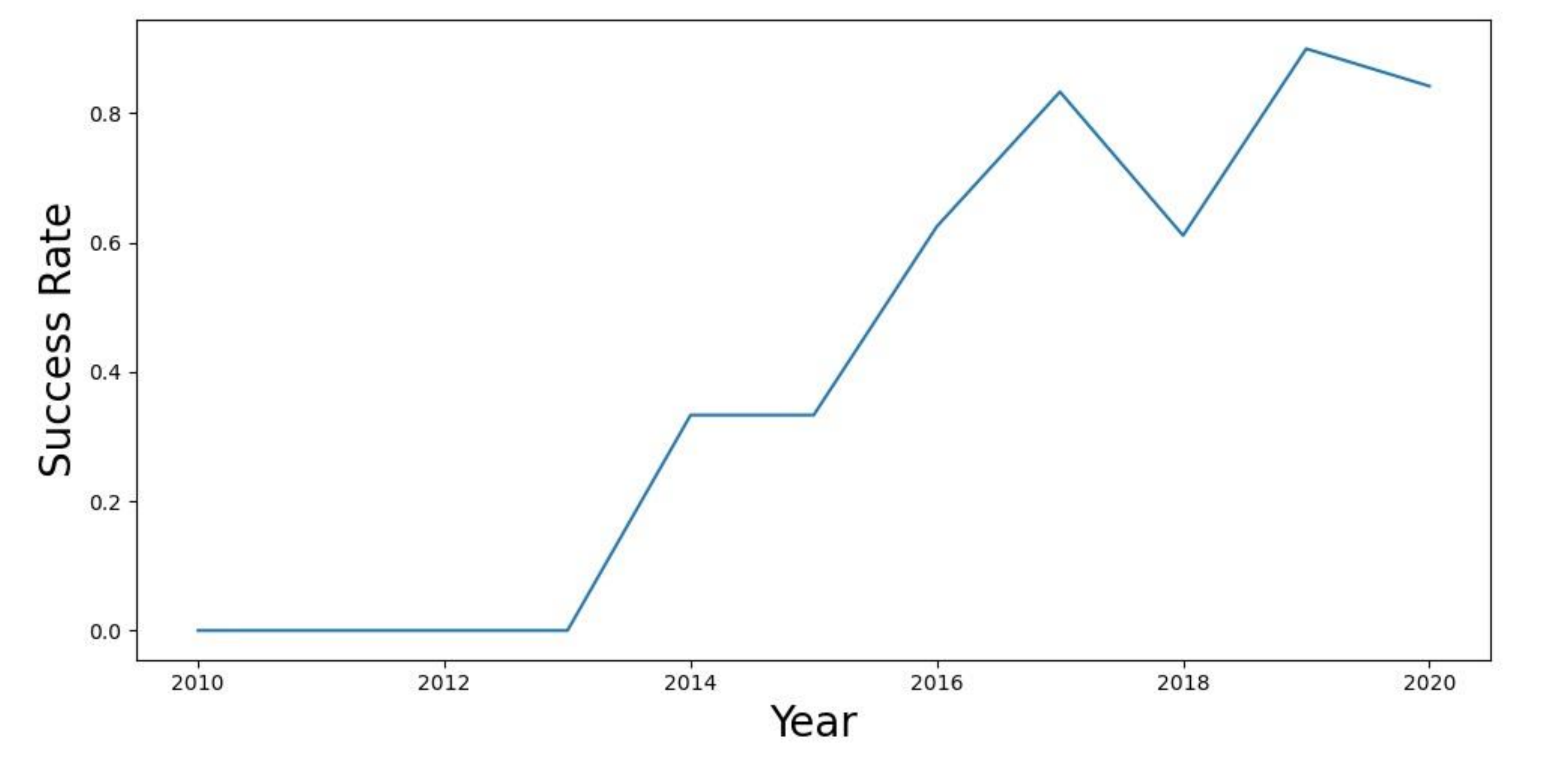


Payload vs. Orbit Type



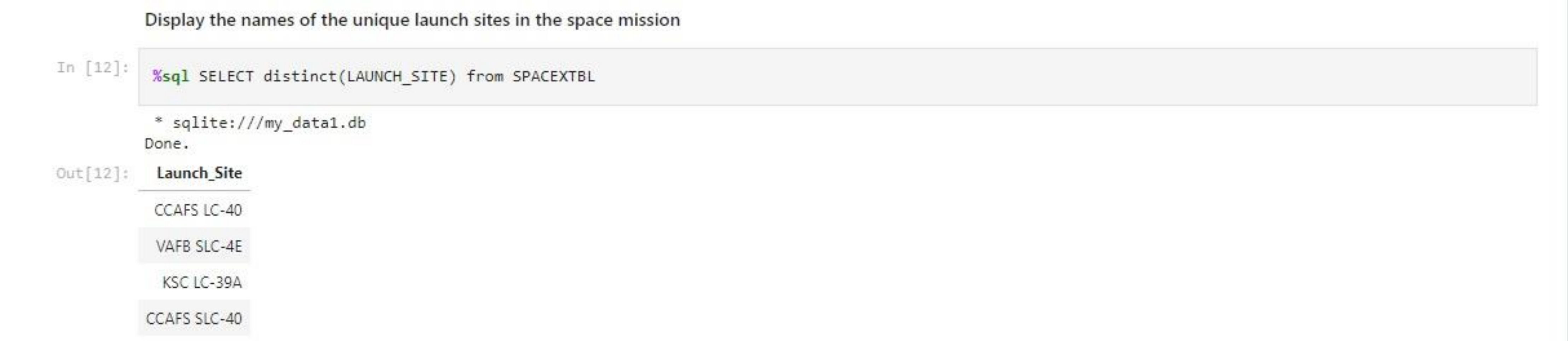
- For PO and LEO, landing fails at low payload mass while it is successful at higher payload mass
- For VLEO the payload is the heaviest
- For LEO, SSO, HEO, ES-L1 the payload is light

Launch Success Yearly Trend



- The average success rate increases over the years from 2013

All Launch Site Names



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

```
In [12]: %sql SELECT distinct(LAUNCH_SITE) from SPACEXTBL
```

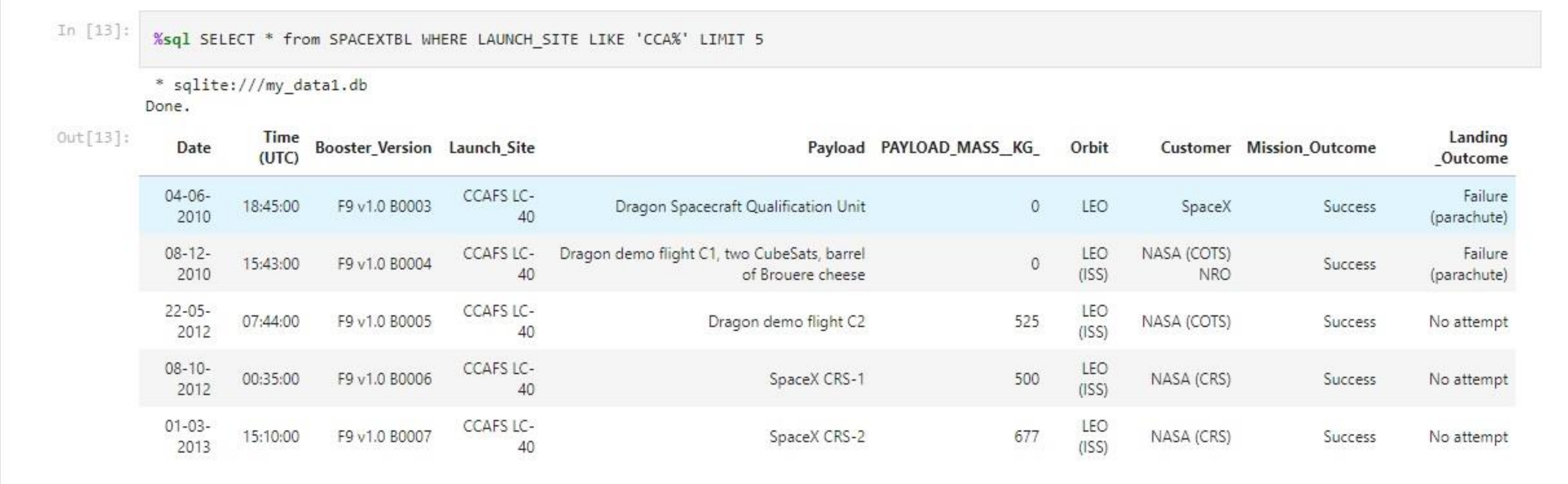
```
 * sqlite:///my_data1.db
Done.
```

Out[12]:

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

- We used the key word DISTINCT to show only unique launch sites from the SpaceX data.

Launch Site Names Begin with 'CCA'



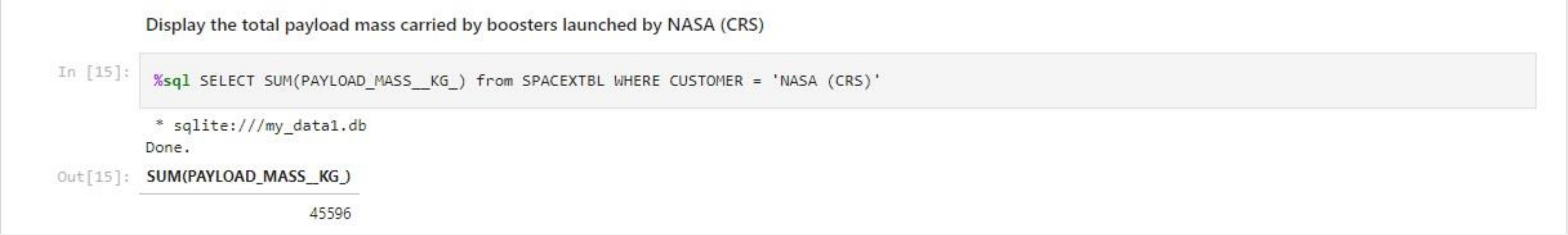
```
In [13]: %sql SELECT * from SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5

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

Out[13]:

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

- We used the query above to display 5 records where launch sites begin with CCA

Total Payload Mass

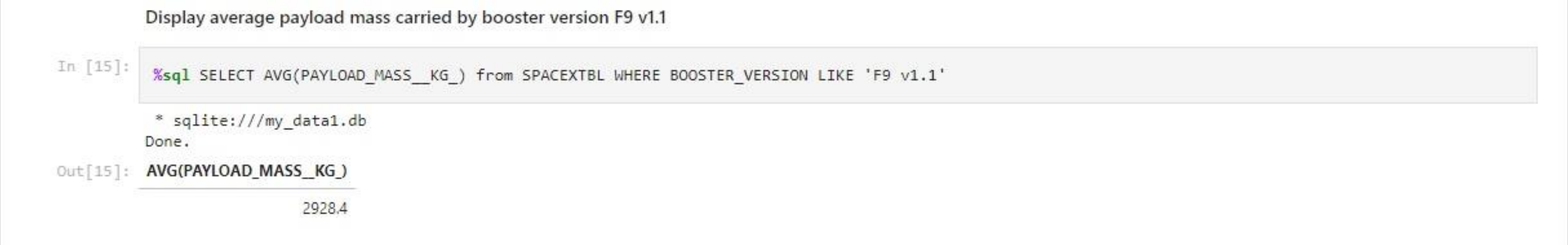


Display the total payload mass carried by boosters launched by NASA (CRS)

```
In [15]: %sql SELECT SUM(PAYLOAD_MASS__KG_) from SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)'

 * sqlite:///my_data1.db
Done.

Out[15]: SUM(PAYLOAD_MASS__KG_)
                         45596
```

- We calculated the total payload carried by boosters from NASA as 45596 using the query below

Average Payload Mass by F9 v1.1



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

```
In [15]: %sql SELECT AVG(PAYLOAD_MASS__KG_) from SPACEXTBL WHERE BOOSTER_VERSION LIKE 'F9 v1.1'
```

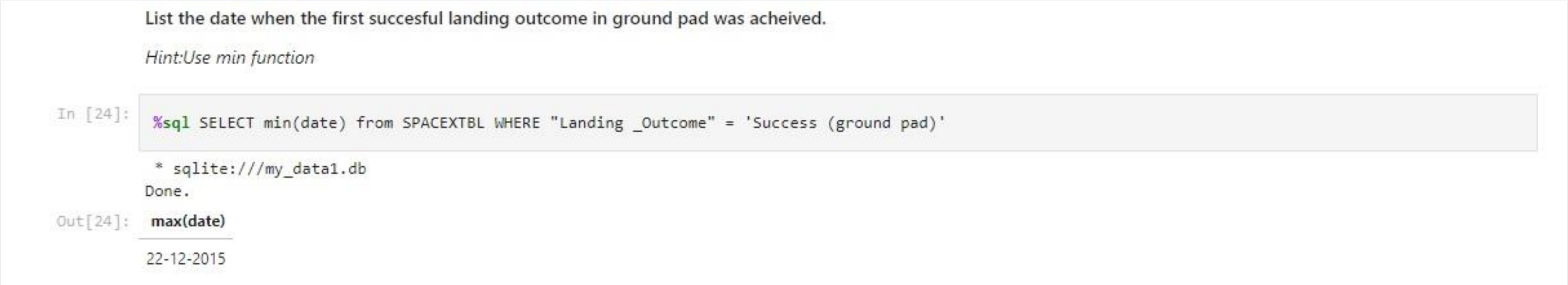
```
 * sqlite:///my_data1.db
Done.
```

Out[15]:

AVG(PAYLOAD_MASS__KG_)
2928.4

- We calculated the average payload mass carried by booster version F9 v1.1 as 2928.4

First Successful Ground Landing Date



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

Hint:Use min function

In [24]:

```
%sql SELECT min(date) from SPACEXTBL WHERE "Landing _Outcome" = 'Success (ground pad)'
```

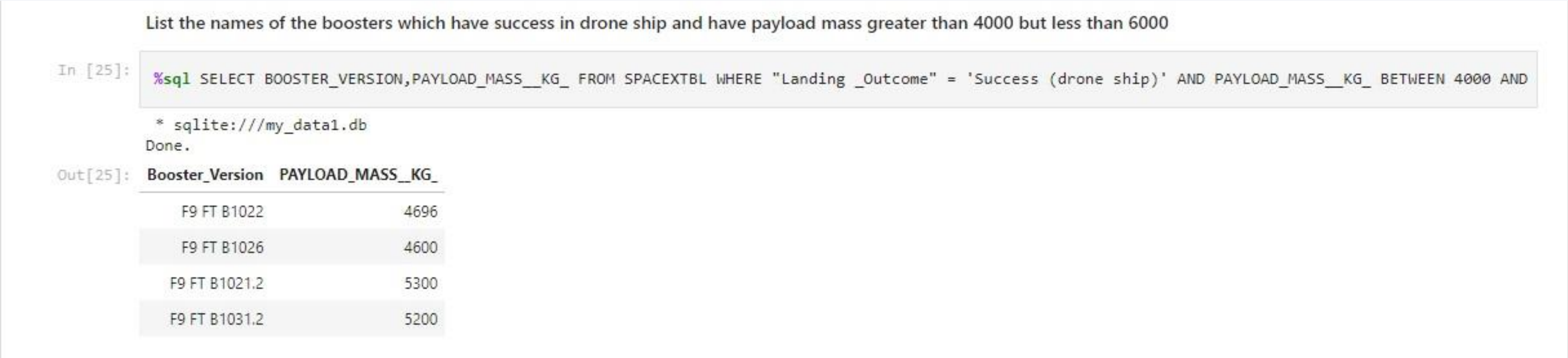
```
 * sqlite:///my_data1.db
Done.
```

Out[24]:

max(date)
22-12-2015

- We observed that the dates of the first successful landing outcome on ground pad was 22 December 2015

Successful Drone Ship Landing with Payload between 4000 and 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 [25]:

```
%sql SELECT BOOSTER_VERSION,PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE "Landing _Outcome" = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND
```

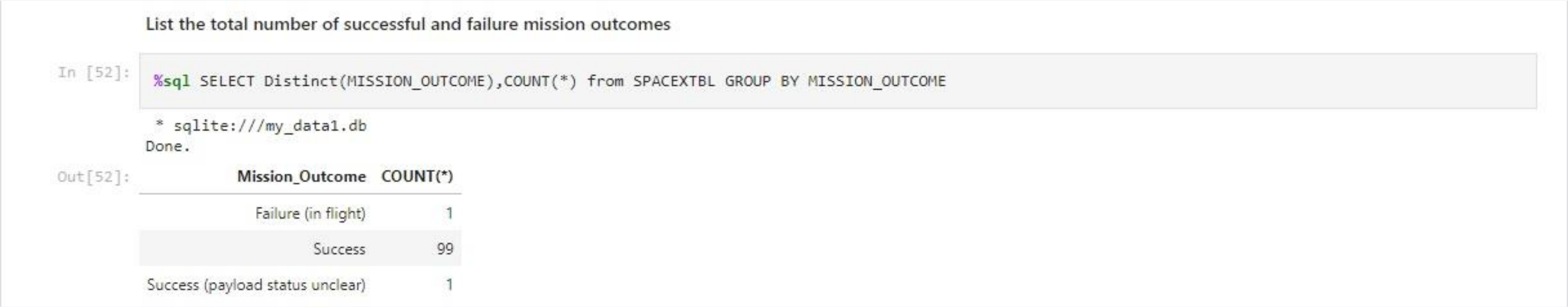
```
 * sqlite:///my_data1.db
Done.
```

Out[25]:

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

- We used the WHERE clause to filter for boosters which have successfully landed on drone ship and applied the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes



List the total number of successful and failure mission outcomes

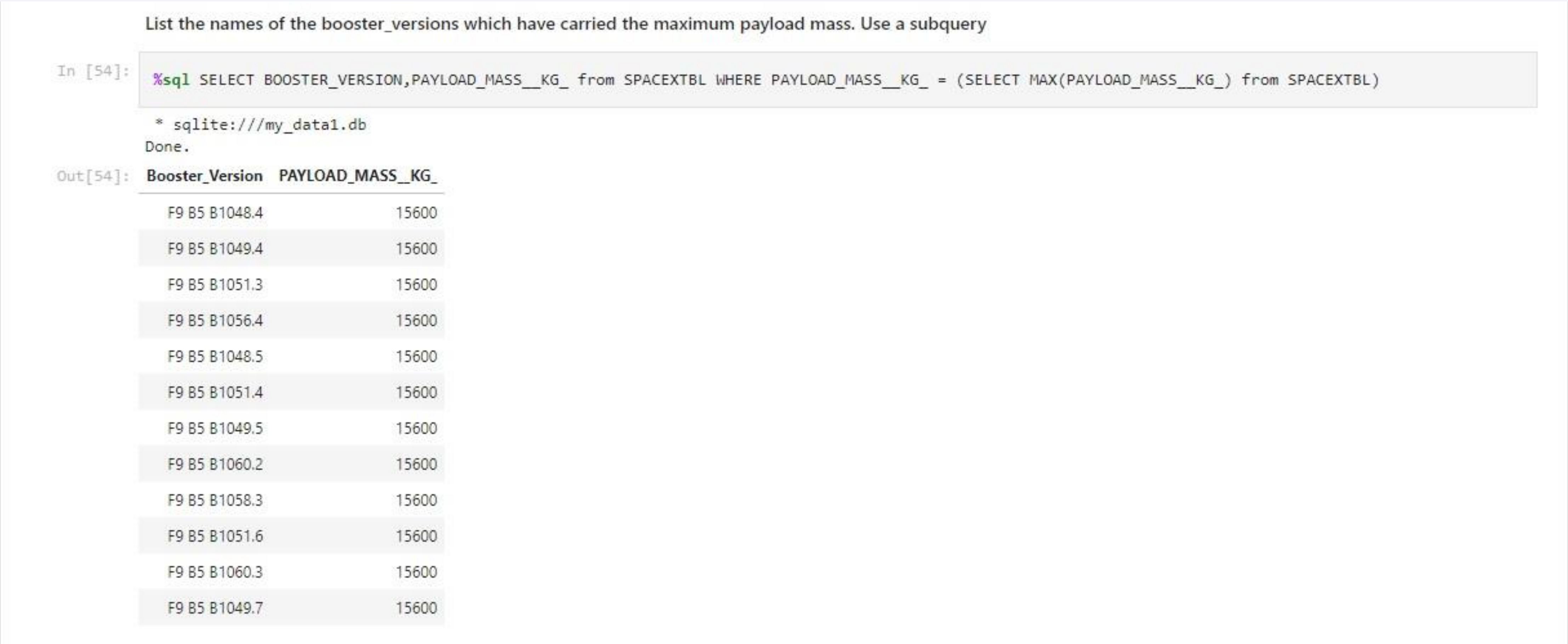
```
%sql SELECT Distinct(MISSION_OUTCOME),COUNT(*) from SPACEXTBL GROUP BY MISSION_OUTCOME
```

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

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

- The COUNT() function is used to count the number of occurences of different mission outcomes with the help of the GROUPBY clause applied to the 'mission_outcome' column. A list of the total number of successful and failure mission outcomes os returned.

Boosters Carried Maximum Payload



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

```
In [54]: %sql SELECT BOOSTER_VERSION,PAYLOAD_MASS__KG_ from SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

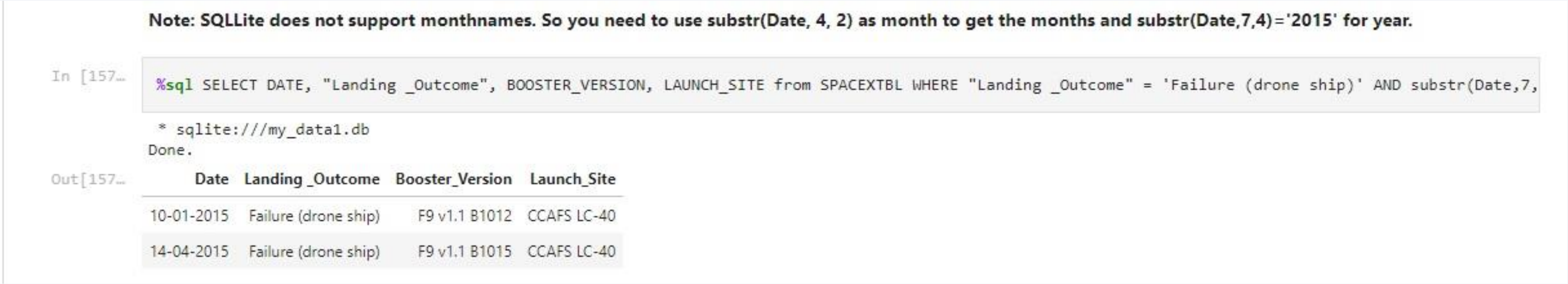
```
 * sqlite:///my_data1.db
Done.
```

Out[54]:

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

- We determined the booster that have carried the maximum payload using a subquery in the WHERE clause and the MAX() function.

2015 Launch Records



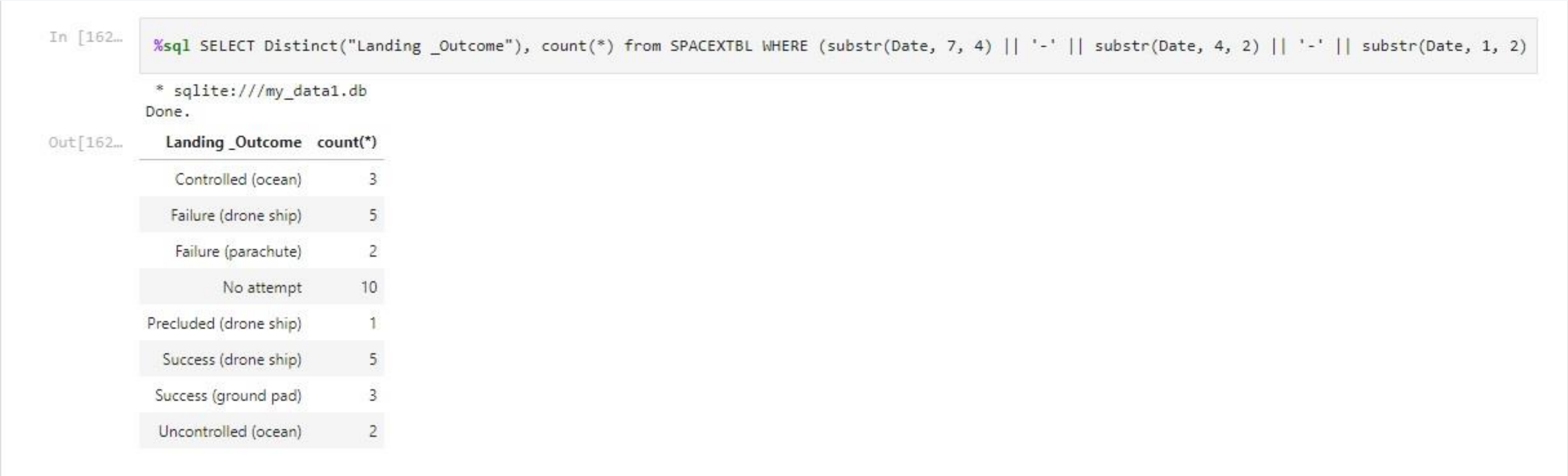
Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

```
In [157…  %sql SELECT DATE, "Landing _Outcome", BOOSTER_VERSION, LAUNCH_SITE from SPACEXTBL WHERE "Landing _Outcome" = 'Failure (drone ship)' AND substr(Date,7,
```

```
 * sqlite:///my_data1.db
Done.
Out[157…
      Date        Landing _Outcome       Booster_Version   Launch_Site
  10-01-2015   Failure (drone ship)    F9 v1.1 B1012    CCAFS LC-40
  14-04-2015   Failure (drone ship)    F9 v1.1 B1015    CCAFS LC-40
```

- We used a combinations of the WHERE clause, LIKE , AND , and BETWEEN conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

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



```
%sql SELECT Distinct("Landing _Outcome"), count(*) from SPACEXTBL WHERE (substr(Date, 7, 4) || '-' || substr(Date, 4, 2) || '-' || substr(Date, 1, 2)
```

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

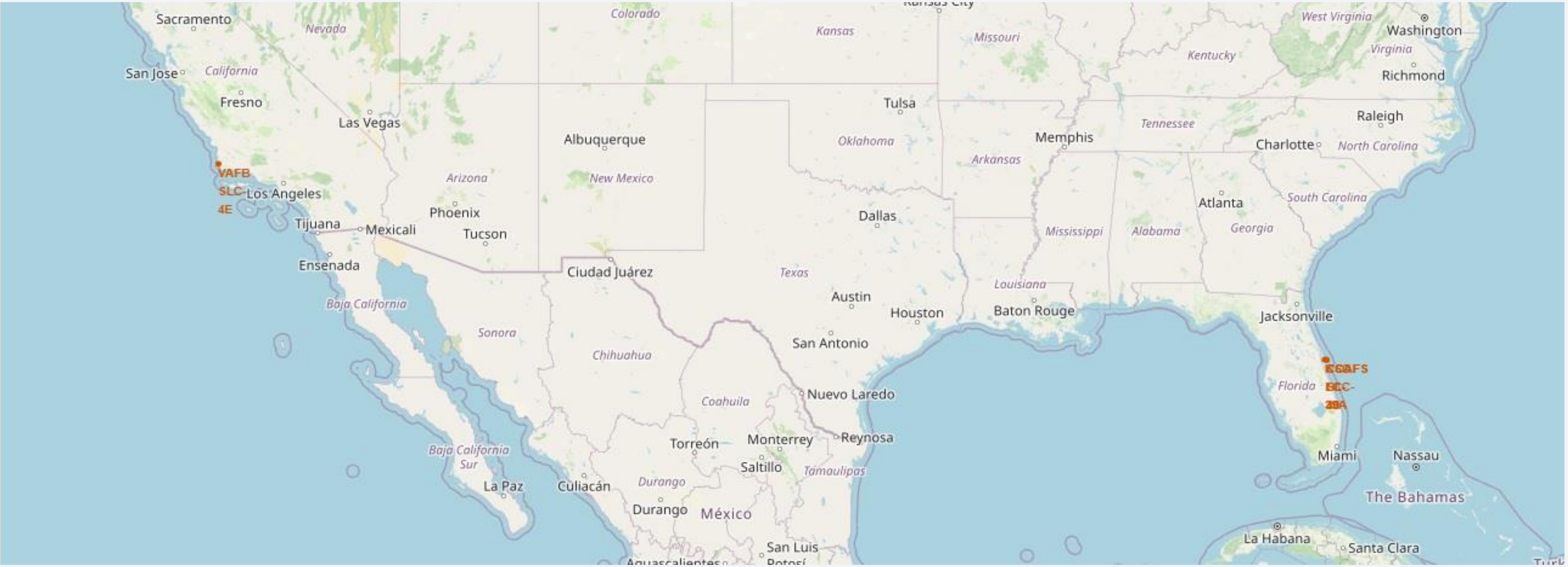
Landing _Outcome	count(*)
Controlled (ocean)	3
Failure (drone ship)	5
Failure (parachute)	2
No attempt	10
Precluded (drone ship)	1
Success (drone ship)	5
Success (ground pad)	3
Uncontrolled (ocean)	2

- We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN 2010 06 04 to 2010 03 20.

Section 3

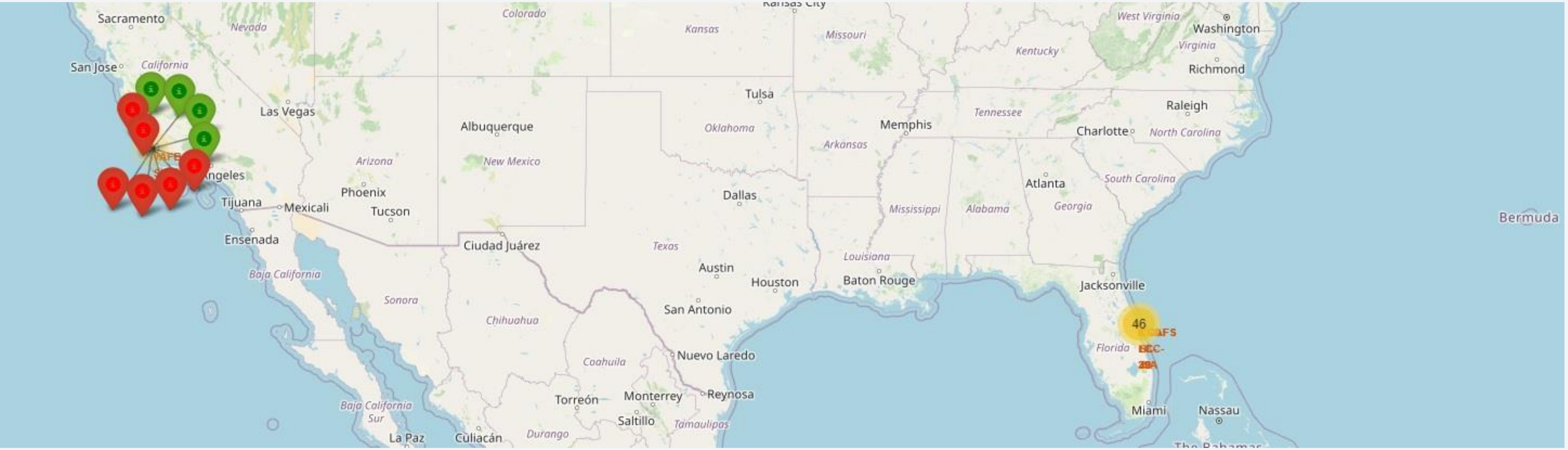
Launch Sites Proximities Analysis

Falcon 9 launch Sites



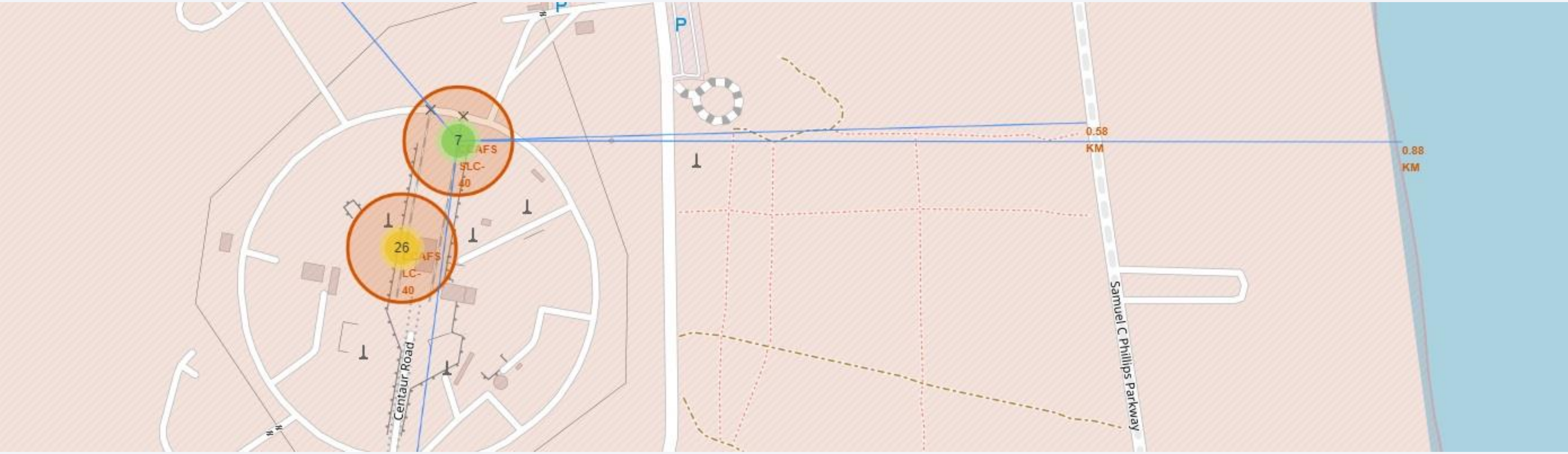
- The launch sites are marked with a red circle and labelled in red font

Launch Outcomes at VAFB SLC 4E

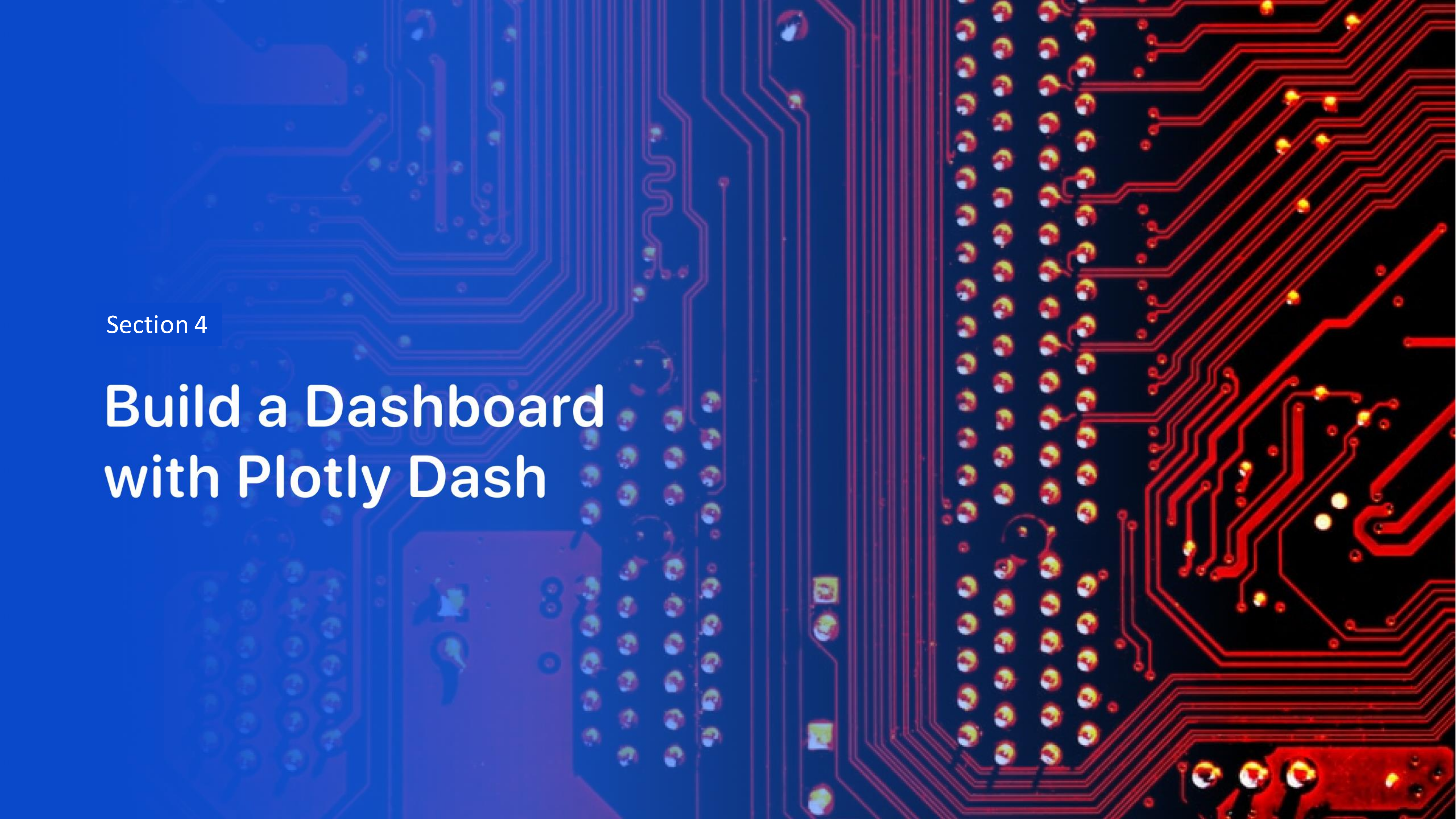


- Red markers indicate failed outcomes whereas green ones indicate successful outcomes

Distance of CCAFS SLC-40 from Amenities



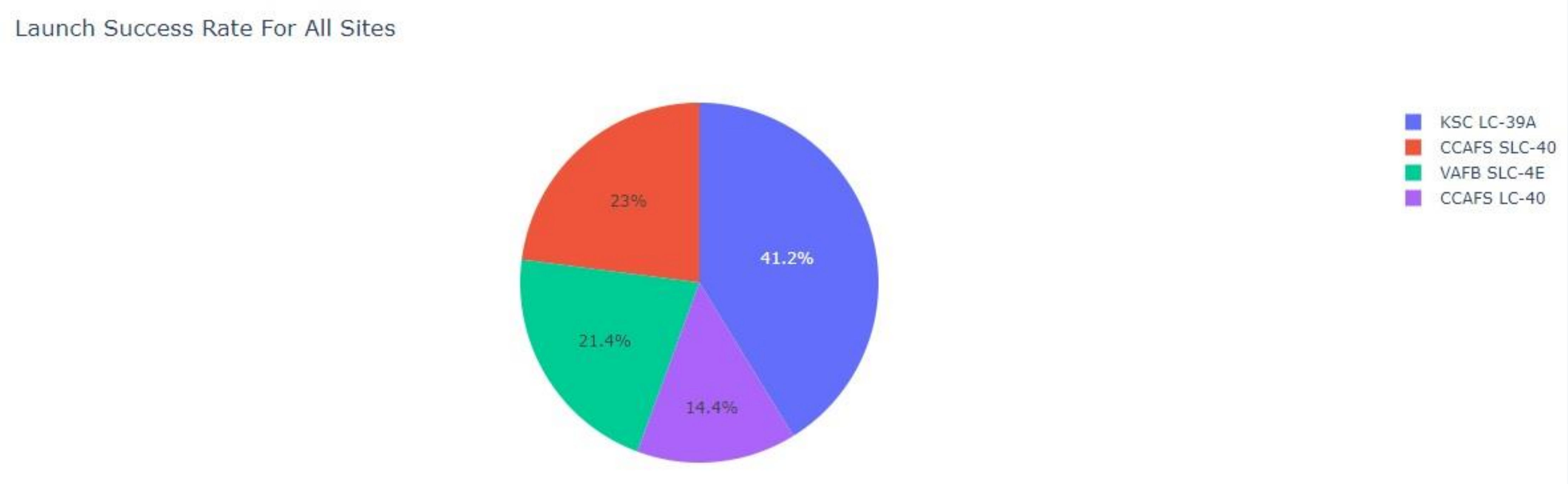
- Distance to coastline: 0.88km, melbourne: 51.43km, highway: 0.58km



Section 4

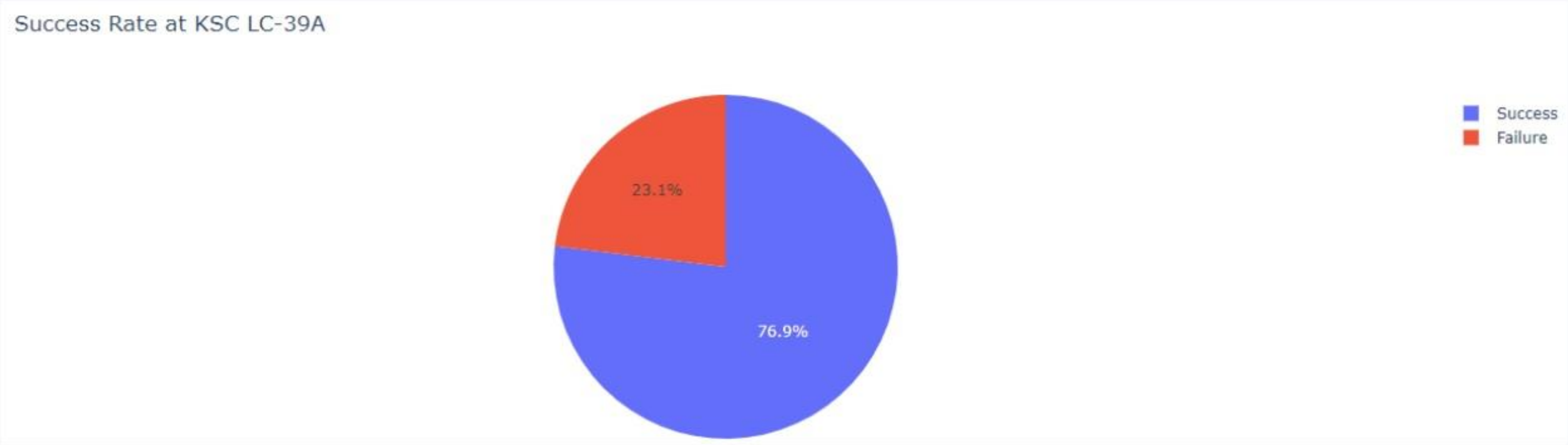
Build a Dashboard with Plotly Dash

Proportion of Total Successes at each Launch Site

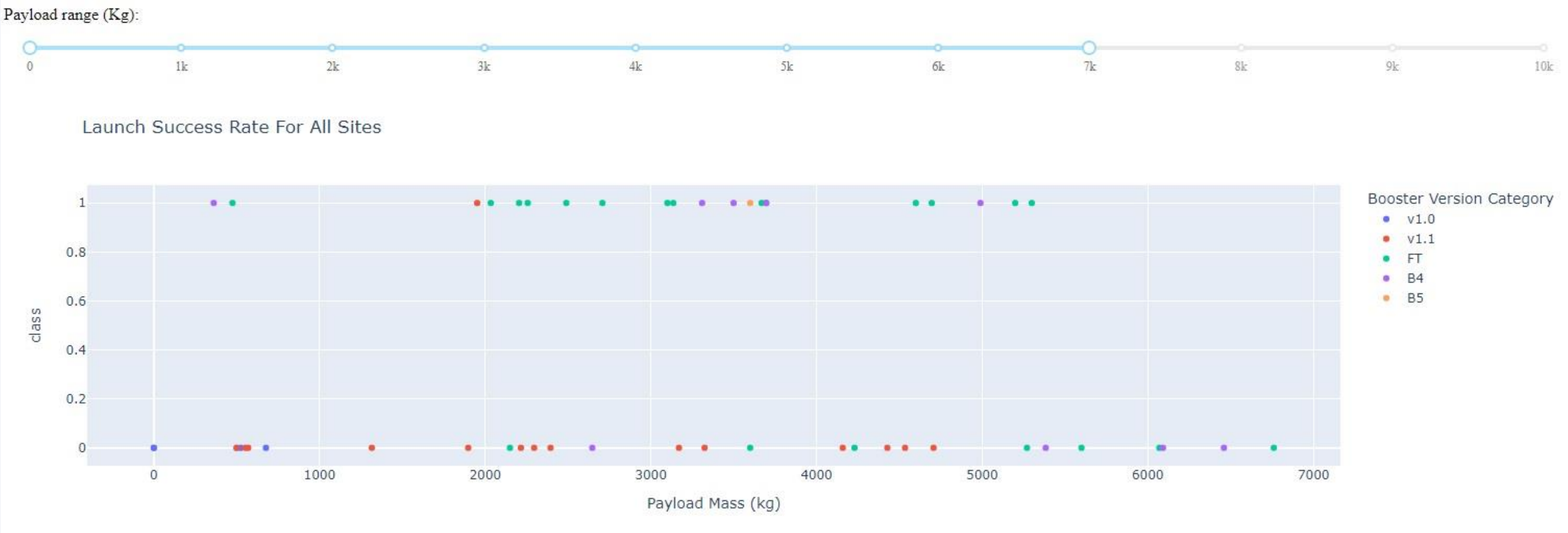


- First position of the successes are from KSC LC-39A launch site and last position is CCAFS LC-40

Launch Site with the Highest Proportion of Success



Launch Outcome versus Payload Mass for All Sites

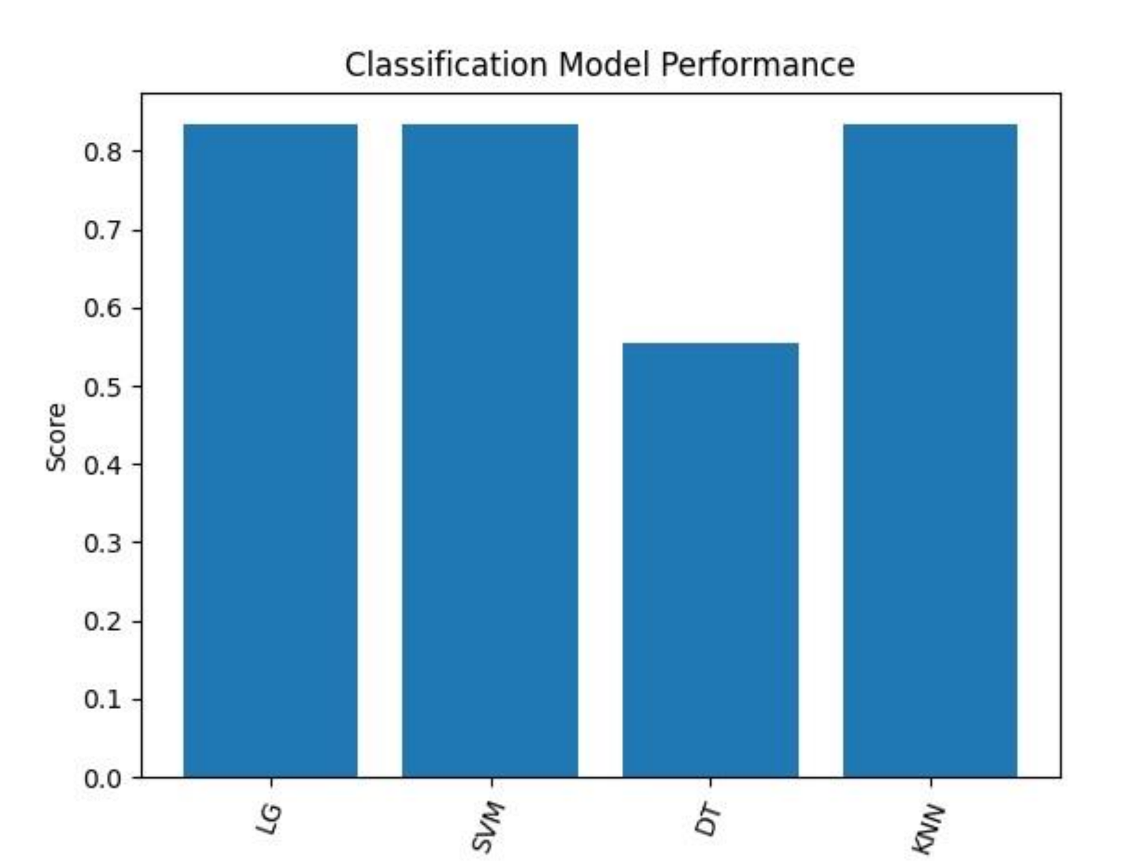


- The payload mass range is between 0 and 7 kg FT has the highest success rate and v1.1 has the lowest

Section 5

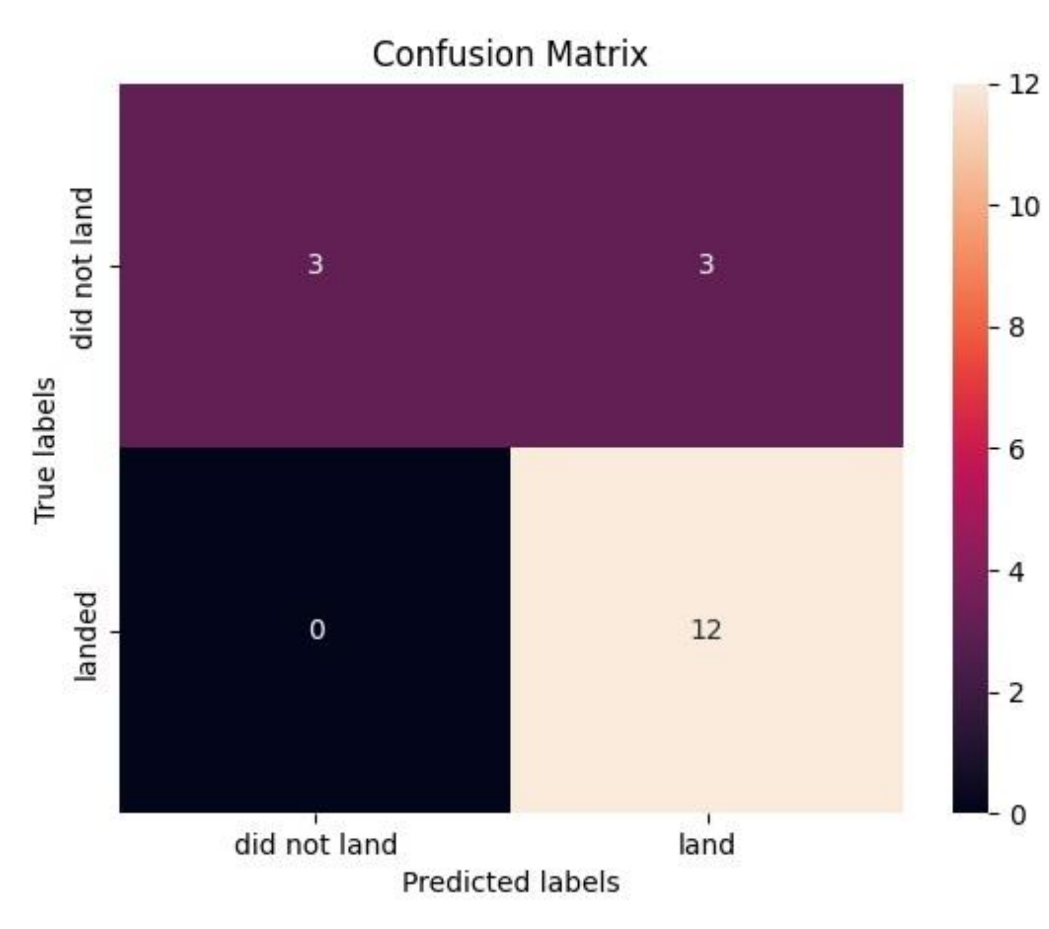
Predictive Analysis (Classification)

Classification Accuracy



- Logistics Regression : 0.8333333333333334
- Support Vector Machine : 0.8333333333333334
- Decision tree method: 0.5555555555555556
- K nearest neighbors : 0.8333333333333334

Confusion Matrix



- The Logistic Regression model has no problem identifying cases where the first stage truly landed

- Logistics Regression : 0.8333333333333334

Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- KSC LC 39A is the launch site where the likelihood of first stage recovery is the highest
- There are certain orbits like SSO, HEO, GEO, and ES-L1 where launches were the most successful.
- It is possible to predict the fate of the first stage of Falcon 9 with a reasonably high degree of accuracy, given feature information such as launch site, payload mass, booster version, orbit type etc, using a logistic regressor
- The Logistics Regression classifier is the best machine learning algorithm for this task.
- As more launches take place, more data points could be added to the master data set and the regressor can be trained to be better at predicting recovery

Appendix

- Coursera Link : https://www.coursera.org/learn/applied-data-science-capstone/home/week/5
- My GitHub Link : https://github.com/yashar-javdani/Yashar-Applied-Data-Science

Thank you!