IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Akash Verma
26-Nov-2024

Outline

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

Executive Summary

Summary of methodologies

Collected and optimized SpaceX Falcon 9 launch data via API to analyze first-stage landing success rates across different areas. Cleaned data was analyzed using SQL and Python visualization libraries to uncover relationships between launch parameters and outcomes. Launch sites and landing success rates were visualized using maps. A dashboard highlighted the rocket with the highest landing success across varying payload masses. Finally, a predictive model was developed to forecast landing success based on launch site and payload mass.

Summary of all results

Out of 101 landing attempts, 61 were successful, and 40 failed. The "KSC LC-39A" launch site had the highest success rate. Booster versions FT and B4 performed best, while V1.1 had the lowest success rate. Logistic Regression emerged as the most effective predictive model, achieving 93.33% accuracy on test data.

Introduction

Background

Space exploration is no longer limited to governments, with private organizations like SpaceX innovating more efficient rocket launches. The Falcon 9 is renowned for its reusable first stage, but successful landings are not guaranteed. Predicting first-stage landing success can significantly reduce spaceflight costs. This project analyzes Falcon 9 launch data from various sites and develops an effective ML model to identify parameters for a successful landing.

The problem and Key Questions

Which parameters predict a flight's successful launch and landing? How factors like launch site, payload mass, booster version, landing site, orbit, and other conditions impact success? *The goal is to develop a predictive model to assess landing success and optimize flight reliability.*

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX past launch data was collected from SpaceX website using the API.
- Performed data wrangling
  - Data was filtered for falcon 9 launch vehicle and then cleaned, as well as explored using basic exploratory technique with pandas python library.
- Performed exploratory data analysis (EDA) using visualization and SQL
- Performed interactive visual analytics using Folium and Plotly Dash
- Performed predictive analysis using classification models
  - Various predictive model development algorithms/techniques were compared and using GridSearchCV module of Sci-Kit learn python library, and chosen the best fit for final model development.

Data Collection

Get Data	JSON to DataFrame	Data Extraction	Saved to CSV
Python Request library was used to fetch past space launch data from SpaceX website using SpaceX API. The data was received as a JSON object.	JSON object was normalized and converted to to tabular data structure called Dataframe using pandas library. Further, it was explored to understand column and entry data.	Custom functions were built and used to extract data sub-data of launch sites, cores, and Payload. Data was filtered to use on Falcon 9 launch events.	Finally data was stored as csv file, that was layered used to perform further studies.

Data Collection – SpaceX API

Click here!

Link to Github File

```
            response = requests.get("https://api.spacexdata.com/v4/cores/"+core['core']).json()
            Block.append(response['block'])
            ReusedCount.append(response['reuse_count'])
            Serial.append(response['serial'])
        else:
            Block.append(None)
            ReusedCount.append(None)
            Serial.append(None)
        Outcome.append(str(core['landing_success'])+' '+str(core['landing_type']))
        Flights.append(core['flight'])
        GridFins.append(core['gridfins'])
        Reused.append(core['reused'])
        Legs.append(core['legs'])
        LandingPad.append(core['landpad'])
```

Now let's start requesting rocket launch data from SpaceX API with the following URL:

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

Check the content of the response

In [13]:
```
print(response.content)
```


```
b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://imag
es2.imgbox.com/94/f2/NN6Ph45r_o.png","large":"https://images2.imgbox.com/5b/02/QcxHUb5V_o.png"},"reddit":{"campaign":null,"la
unch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.c
om/watch?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.space.com/2196-spacex-inaugural-falcon-1-rocket-los
t-launch.html","wikipedia":"https://en.wikipedia.org/wiki/DemoSat"},"static_fire_date_utc":"2006-03-17T00:00:00.000Z","static
_fire_date_unix":1142553600,"net":false,"window":0,"rocket":"5e9d0d95eda69955f709d1eb","success":false,"failures":[{"time":3
3,"altitude":null,"reason":"merlin engine failure"}],"details":"Engine failure at 33 seconds and loss of vehicle","crew":
[],"ships":[],"capsules":[],"payloads":["5eb0e4b5b6c3bb0006eeb1e1"],"launchpad":"5e9e4502f5090995de566f86","flight_number":
1,"name":"FalconSat","date_utc":"2006-03-24T22:30:00.000Z","date_unix":1143239400,"date_local":"2006-03-25T10:30:00+12:00","d
```


Requests library was installed and imported in the project to request data from SpaceX REST API.

Received status of the response using response.status_code

GET method of Requests library was used to get JSON data from SpaceX API url.

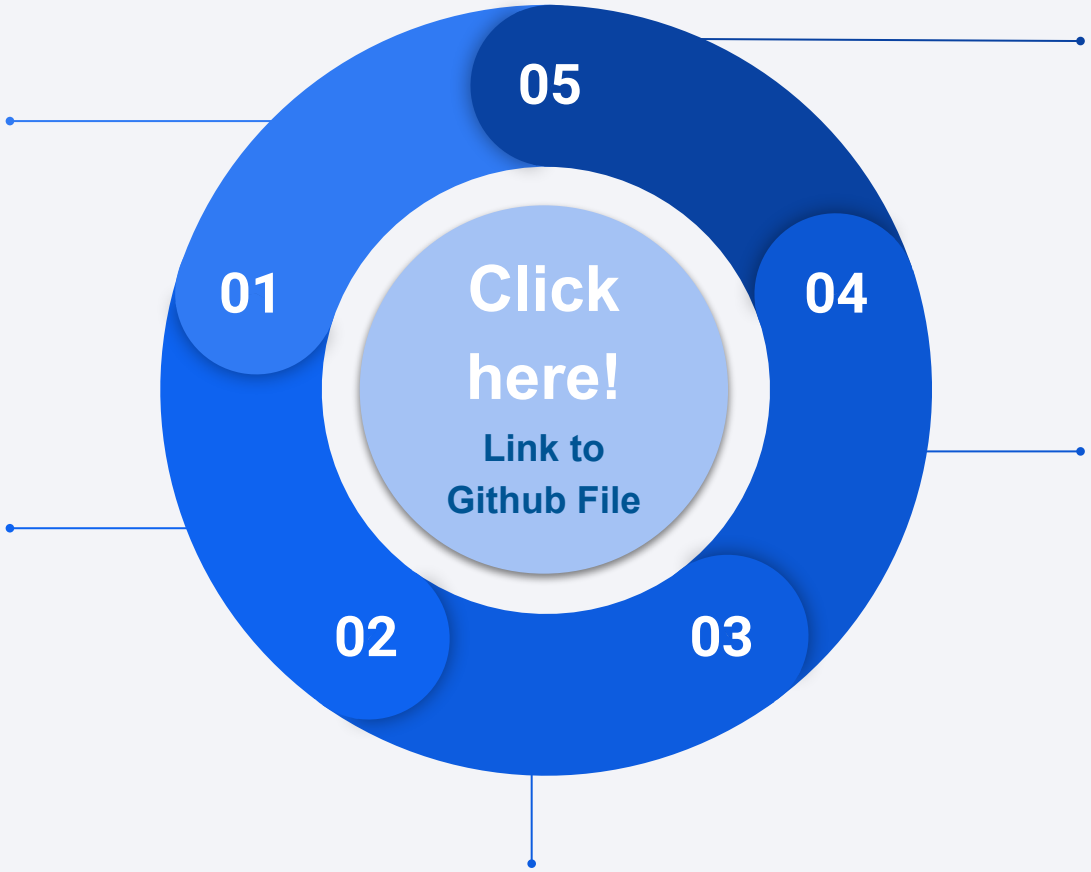
Data Wrangling

Data Analysis

Data was analysed to find missing values in the dataset and data-type of each column (feature).

Launches on each Site

Used .value_counts() method to find number of launches per from each launch site. CCAFS SLC 40 found to be winner.



Launches to each orbit

Found the number launches to each orbit. GTO (a geosynchronous orbit) won the race with 27 launches.

Class of Outcomes

Landing outcomes were classified as “success” and “failure”, finally stored as 0 (failure) and 1 (success) in a column named “class” in the pandas dataframe representing launch data.

Landing Outcomes

Most successful landing outcomes were on drone ship (41/47).

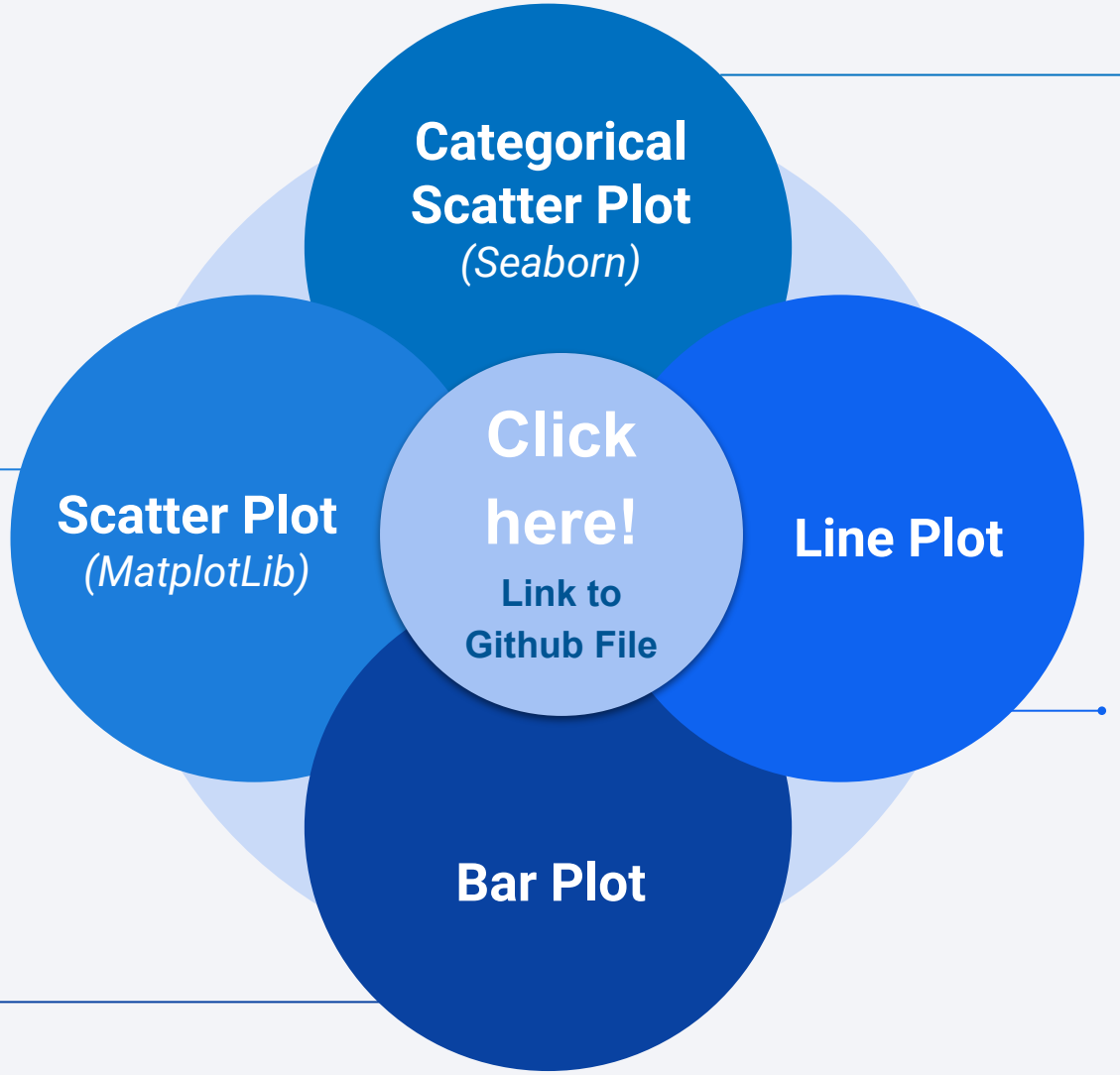
EDA with Data Visualization

Matplotlib Scatter plots were used to observe the frequency of successful landing attempts in different orbits and its combined relationship with:

- Payload mass
- Flight number

High success rate was observed for orbit VLEO with high flight number. Good success was observed for SSO orbit when payload mass is low.

Bar plot was used to visualize the categorical success rate of each orbit.



Explored combined relationships of following features with landing outcome (class):

- Flight Number & Payload mass.
- Flight Number & Launch Site
- Payload Mass & Launch Site

Landing is more successful at higher figh number and payload mass.

launches from KSC LC 39A has more success rates.

The line plot was used to see year-wise trend of the launch success.

Launch success has boosted astonishingly after 2013, with 2019 marked as most successful year for launching successful flights.

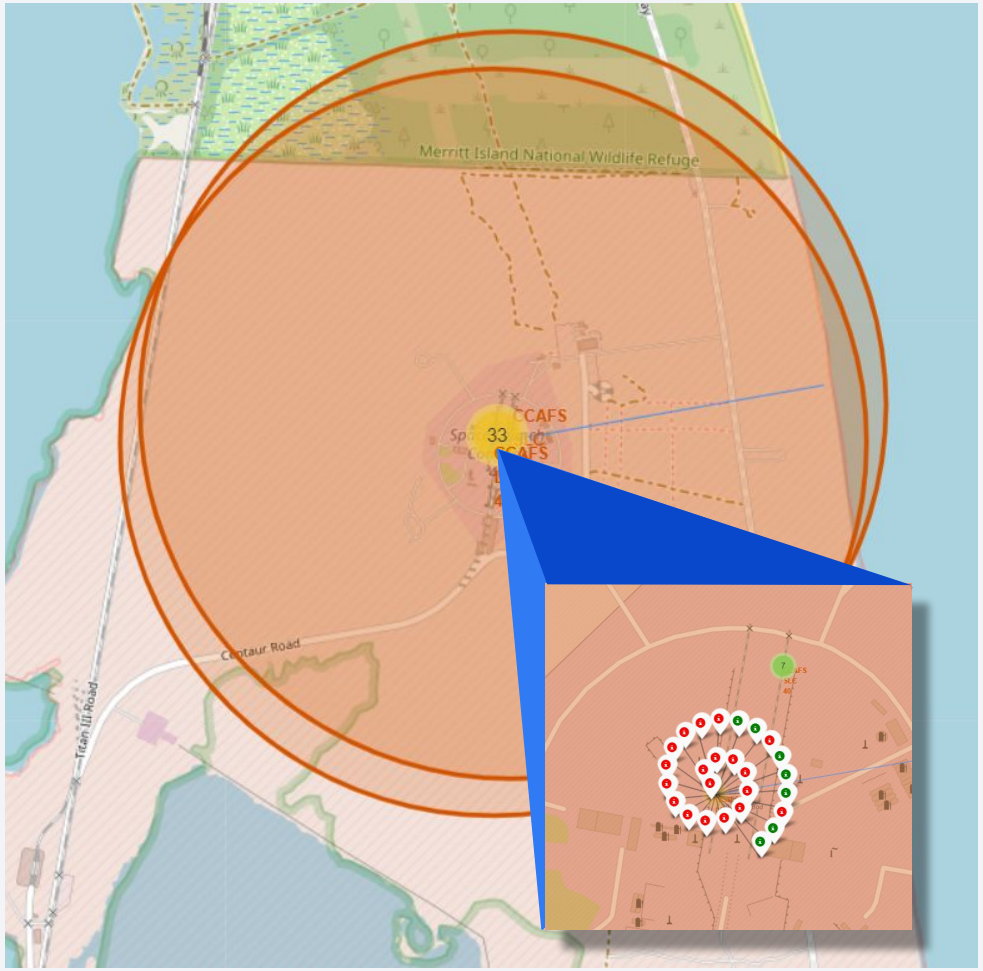
EDA with SQL

Click here!
Link to Github File

	Queries	Key Statements
01	Unique Launch Sites	DISTINCT()
02	Launch sites with keyword CCS	LIKE "CCA%"
03	Average payload mass carried out by F9 V1.1	SUM(), WHERE "Customer" = "NASA (CRS)"
04	Date of first landing success at Ground Pad	AVG("PAYLOAD_MASS__KG_")
05	Total payload mass by NASA (CRS)	MIN("Date")
06	Boosters with success on drone ship and 4000 < payload mass < 6000	BETWEEN 4000 AND 6000
07	Total successful and failed landing outcomes	SUM()
08	Boosters that carried maximum payload mass	MAX("PAYLOAD_MASS__KG_")
09	Month wise landing outcome data	substr("Date", 0, 5) = '2015
10	Ranking Landing Outcomes between dates 2010-06-04 and 2017-03-20	COUNT(), BETWEEN(), GROUP BY, ORDER BY

Build an Interactive Map with Folium

Click here!
Link to Github File



Marker

Markers were used to mark the exact location of launch sites.

Circle

Circles were used to visualize the area around the launch sites.

PolyLine

Lines were used to show distance of nearest coastline, railway line, city, and highway from the launch site location.

Marker Cluster

Marer cluster was used to show landing outcomes on different launch sites.

Build a Dashboard with Plotly Dash

Click here!
Link to Github File

A dashboard was created using plotly dash which has following components:

1. A dropdown

Drop-down to select the launch site for which the data is to be visualized on pie-chart and scatter plot.

2. Pie Chart

The pie-chart shows proportion of successful and failed outcome for selected launch site.

3. Range Slider

Range slider helps in selecting a range of payload mass for which the data is to be visualized on categorical scatter plot.

4. Scatterplot

Categorical scatterplot consider payload as launch site as input to visualize landing outcome for both.

Predictive Analysis (Classification)

Click here!
Link to Github File

Data Preprocessing | Train-Test-Split | Model Development | Model Evaluation | Model Selection

Python Lib/Module

numpy

sklearn:

Preprocessing

Outcomes

A numpy array of landing outcomes stored inside a *Y* variable.

Scaler Standardization of source data and dropped outcome column stored as the data frame *X*.

Python Lib/Module

sklearn:

train_test_split

Outcomes

The data is splitted into *training and validations sets*. Each set contains feature data and outcome data as:

X_train

X_test

Y_train

Y_test

Python Lib/Module

sklearn:

LogisticRegression

SVC

DecisionTreeClassifier

KNeighborsClassifier

GridSearchCV

Outcomes

Data was used to develop different type of ML models and best hyper parameters were selected using *Grid Search* Algorithm.

Python Lib/Module

sklearn

Outcomes

Each model was evaluated by:

Calculating the *accuracy of model* with best parameters (as identified using Grid Search).

Visualizing *confusion matrix.*

Python Lib/Module

Based on the outcome from calculated accuracy and confusion matrix, *Logistic Regression* was identified as most suitable model for to predict landing outcome using various parameters.

Results

EDA

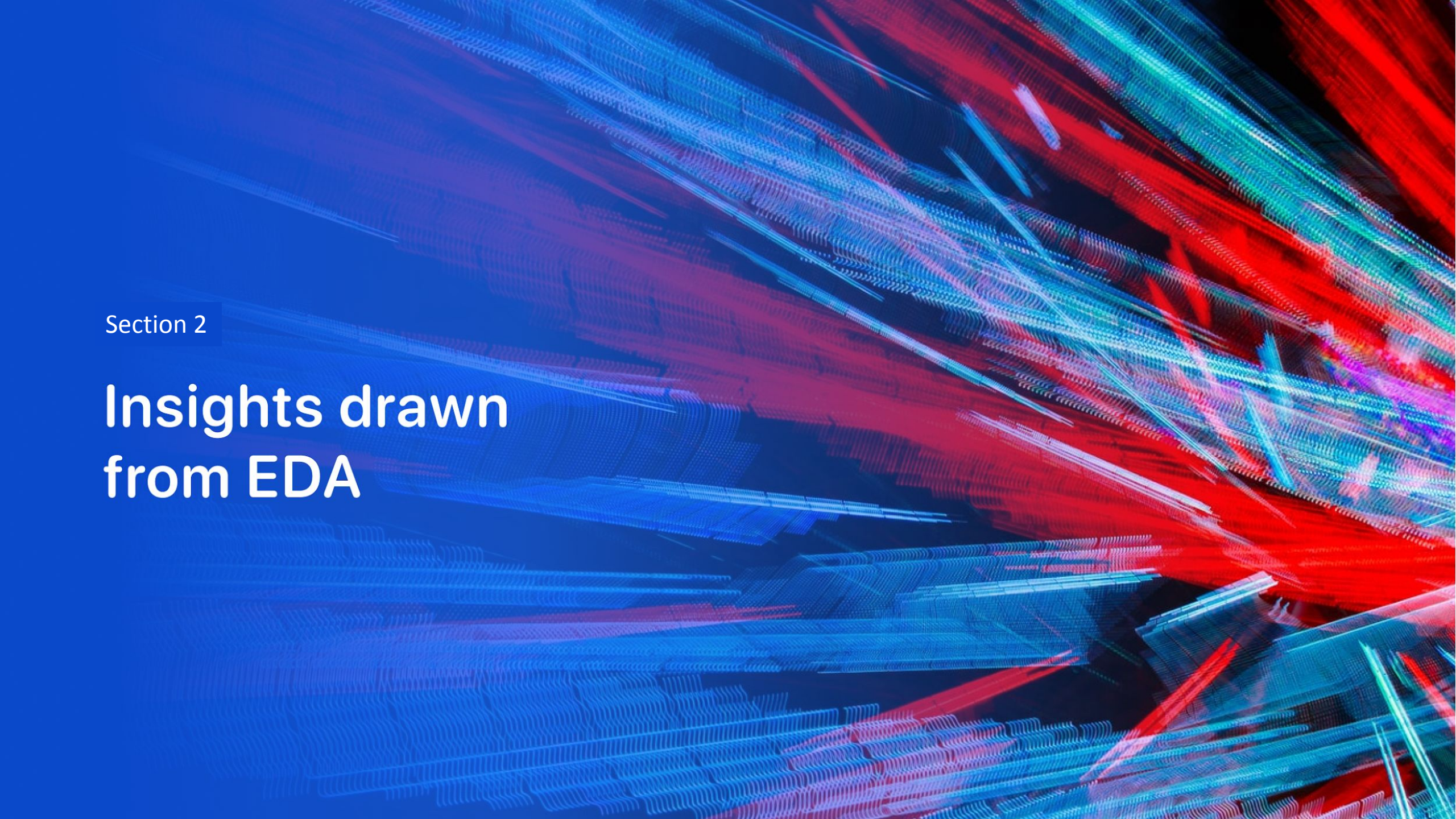
Total successful landings: **61** | Total failures: **40**
As flight number increases, there are more chances of successful landing.
As payload mass increases, chances of successful landing increases.
Successful landings have been increasing amazingly since 2013.

Interactive Analytics using Map and App

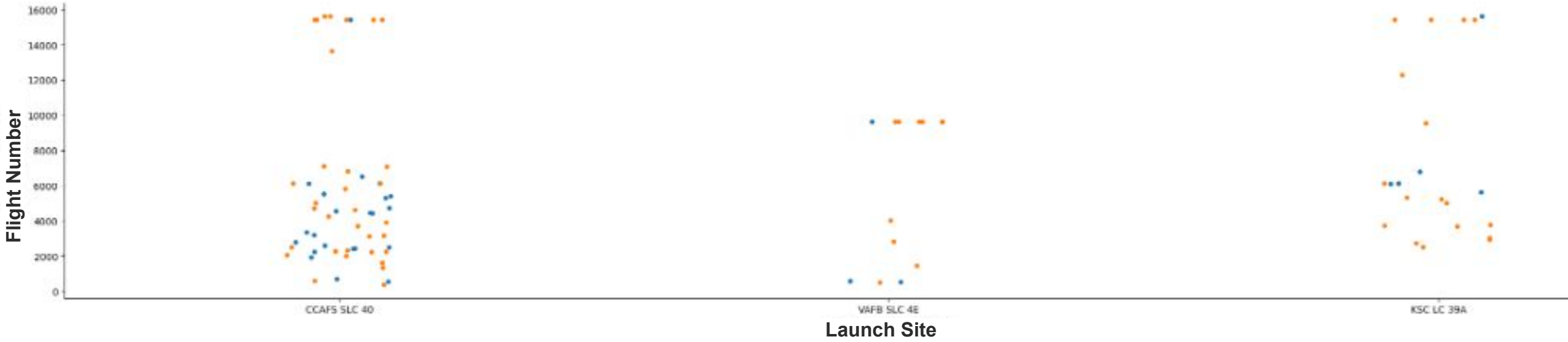
Predictive Analytics

Section 2

Insights drawn from EDA

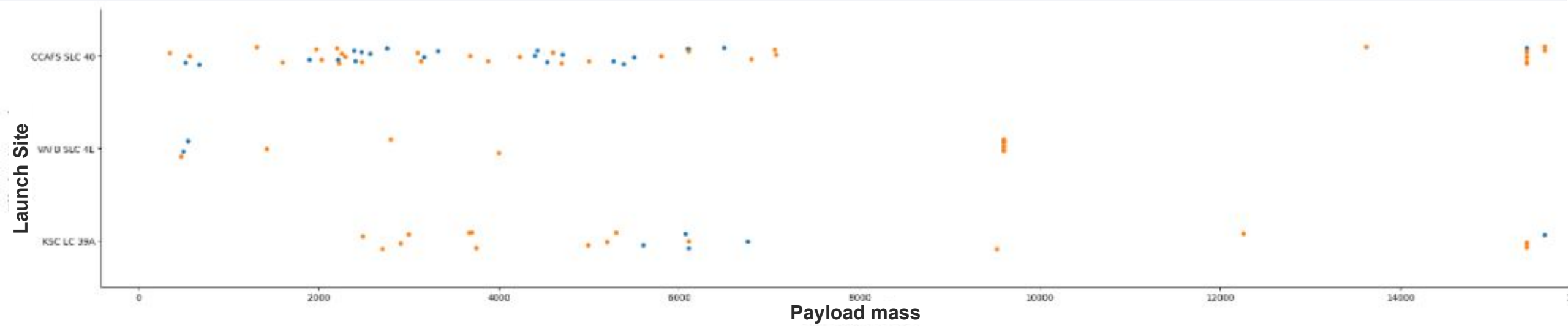


Flight Number vs. Launch Site



As flight number increases, landing success increases for all launch sites.

Payload vs. Launch Site



Landing success is inconsistent for CCAFS SLC 40 launch site, until payload jumps beyond 14000 kg.

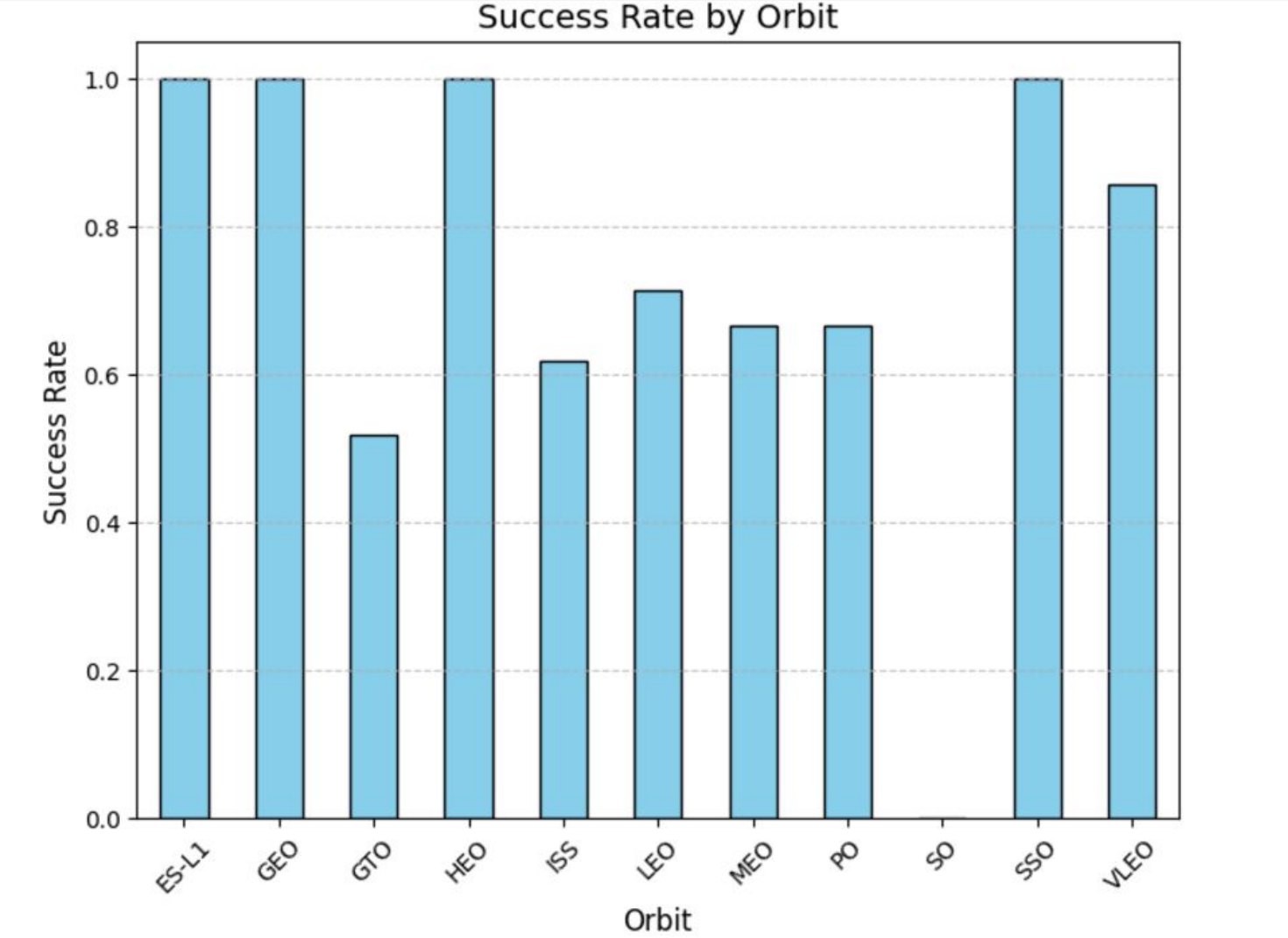
Landing success beyond 500 kg is consistently maintained for VAFB SLC 4E launch site.

Landing success seems to be achievable between 2000 kg to 600 kg payload mass for launch site KSC LC 39A.

Success Rate vs. Orbit Type

Following orbits seems to have high success rates:

- ES-L1
- GEO
- HEO
- SSO
- VLEO



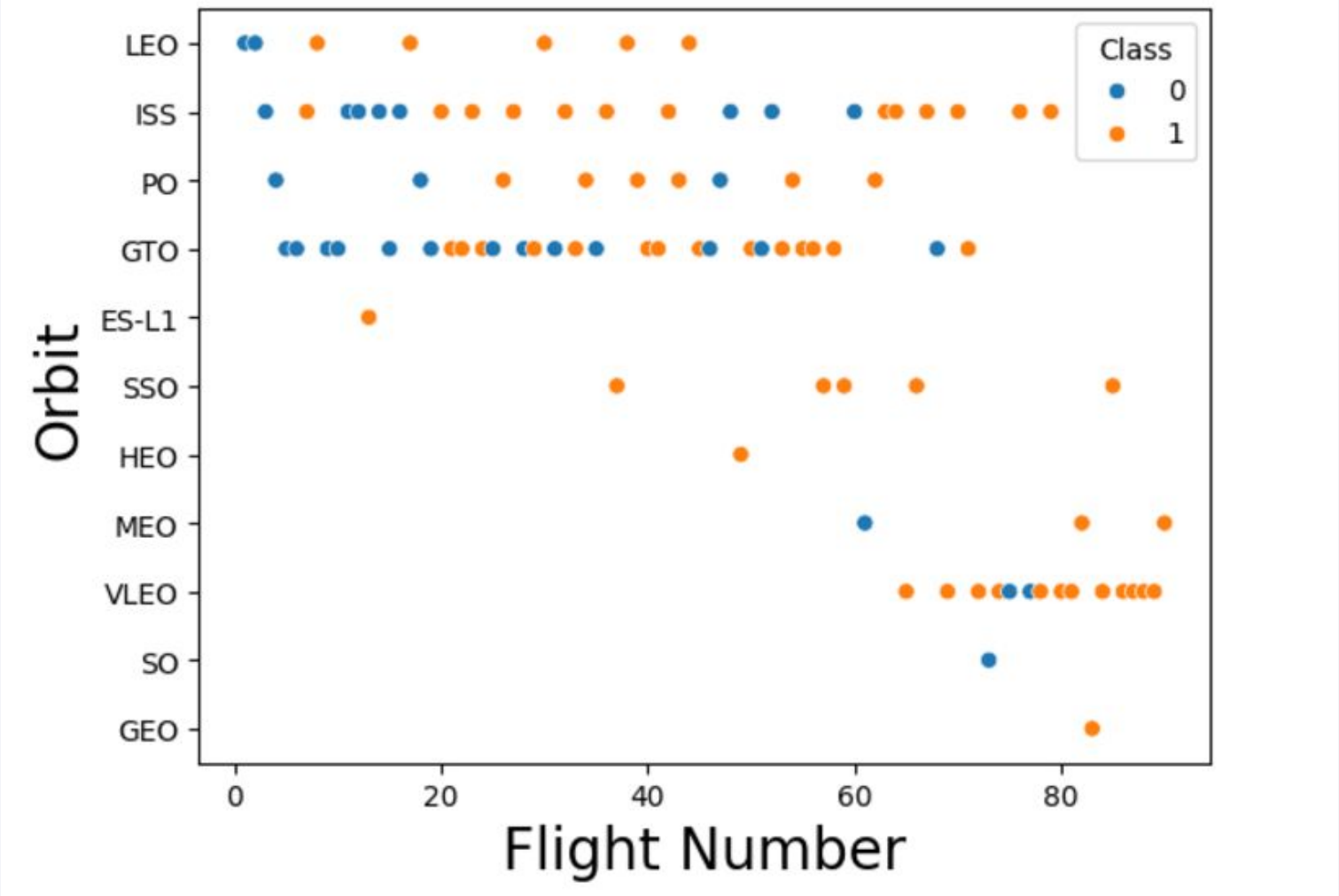
Flight Number vs. Orbit Type

Following orbit's success seems to depend fully or partially on flight number:

- LEO (Fully)
- ISS (20<FN<40, <60)
- VLEO (60<FN<75, <80)
- PO (20<FN<50)

Following orbit's success does not depend on flight number:

- GTO



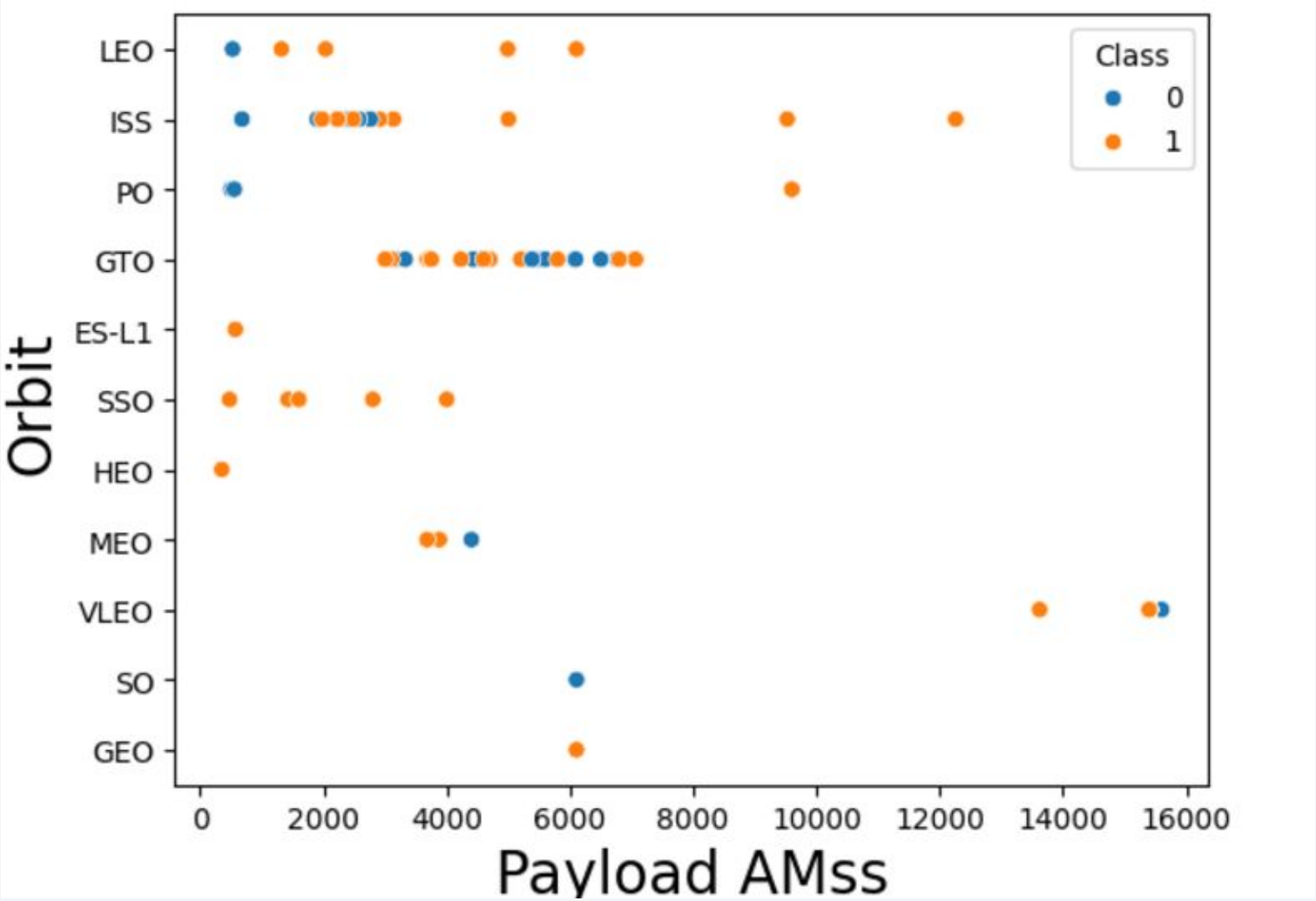
Payload vs. Orbit Type

Following orbit's success seems to depend fully or partially on payload mass:

- LEO (Fully)
- ISS
- PO (Low sample)

Following orbit's success does not depend on flight number:

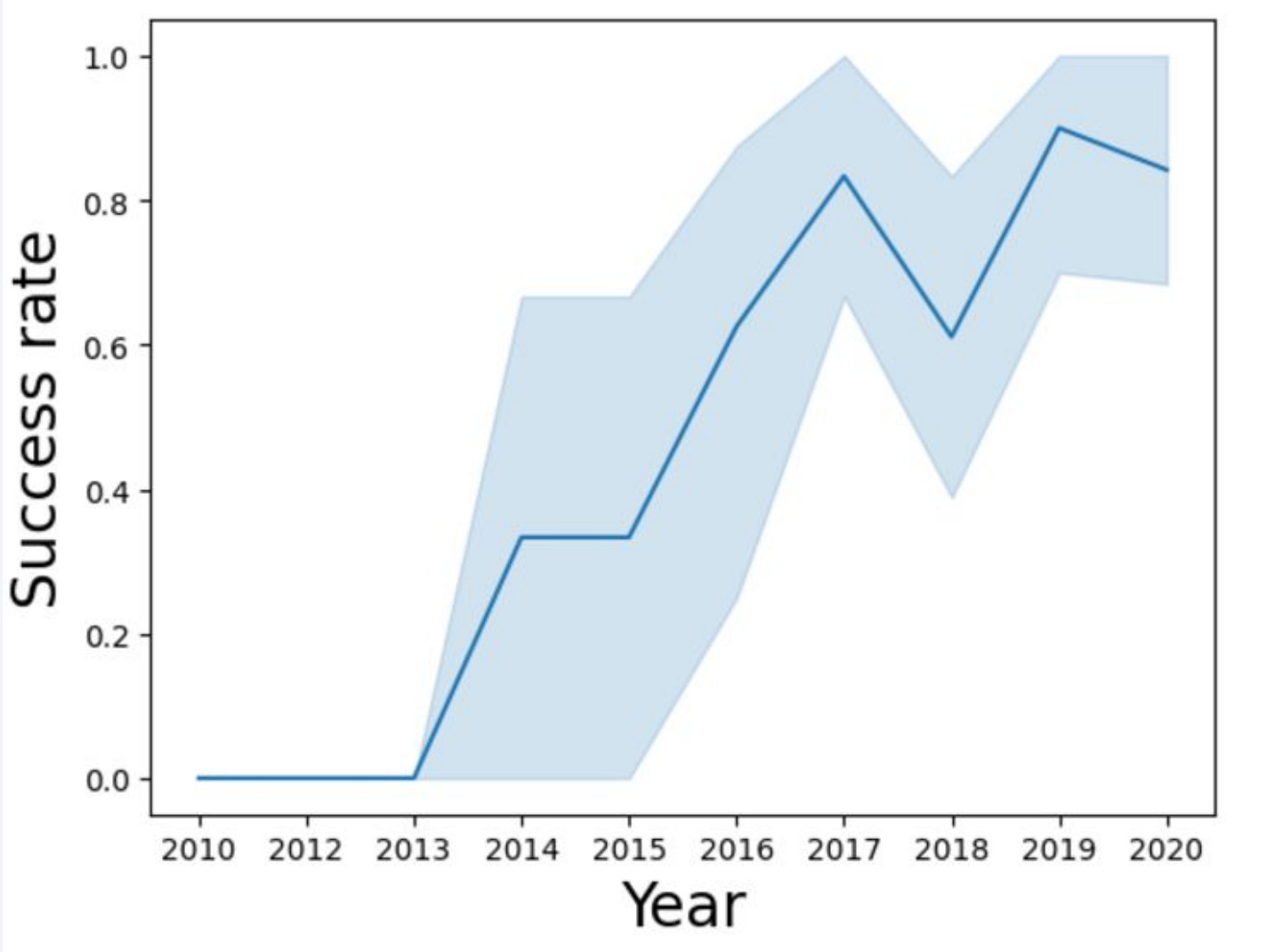
- GTO



Launch Success Yearly Trend

Success in vehicle (stage 1 of Falcon 9) is increasing consistently since 2013.

2017 and 2019 can be seen as local maxima on the plot signifying larger number of attempts of successful landing.



All Launch Site Names

Library used:

csv, sqlite3, prettytable

Key Statement:

DISTINCT

Explanation:

distinct STATEMENT extract unique values from the column “Launch Site” from SPACEXTABLE data.

```
In [15]: %sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE
 * sqlite:///my_data1.db
Done.
```

Out[15]:

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

Launch Site Names Begin with 'CCA'

Library used:

csv, sqlite3, prettytable

Key Statement:

LIKE, LIMIT

Explanation:

LIKE statement is used to find all entries from the column “Launch Site” starting with CCA. The result is limited to 5 records using LIMIT statement.

```
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5
```

✓ 0.0s

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

Date	Time (UTC)	Booster_Version	Launch_Site	
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Sp
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two C
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	

Total Payload Mass

```
%sql SELECT SUM("PAYLOAD_MASS__KG_") AS "Total_Payload_Mass_Kg" FROM SPACEXTABLE WHERE "Customer" = 'NASA (CRS)'
```

[25] ✓ 0.0s

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

Total_Payload_Mass_Kg
45596

Library used:

csv, sqlite3, prettytable

Key Statement:

SUM()

Explanation:

SUM statement is used to calculate total payload mass for Customer name NASA (CRS). Then, it is stored as Total_Payload_Mass_Kg.

Average Payload Mass by F9 v1.1

Task 4

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

In [21]:
```
%sql SELECT AVG("PAYLOAD_MASS__KG_") AS "Average_Payload_Mass_Kg" FROM SPACEXTABLE WHERE "Booster_Version" = "F9 v1.1"
```

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

Out[21]:

Average_Payload_Mass_Kg
2928.4

Library used:

csv, sqlite3, prettytable

Key Statement:

AVG()

Explanation:

AVG statement is used to calculate average payload mass for Booster Version F9 V1.1. Then, it is stored as Average_Payload_Mass_Kg.

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

In [22]:
```
%sql SELECT MIN("Date") FROM SPACEXTABLE WHERE "Landing_Outcome" = "Success (ground pad)"
```

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

Out[22]:

MIN("Date")
2015-12-22

Library used:

csv, sqlite3, prettytable

Key Statement:

MIN()

Explanation:

MIN() statement was used on the “Date” column for the value ‘Success (ground pad)’ in the column Landing_Outcome.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "PAYLOAD_MASS__KG_" BETWEEN 4000 AND 6000
```

[26] ✓ 0.0s

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

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

Library used:

csv, sqlite3, prettytable

Key Statement:

BETWEEN

Explanation:

BETWEEN statement is used to filter the data for Payload mass between 4000 to 6000 kg, and further filtered to select ‘Success (drone ship)’ as value in the column Landing_Outcome.

Total Number of Successful and Failure Mission Outcomes

```
In [25]: %sql SELECT SUM(CASE WHEN "Landing_Outcome" LIKE '%success%' THEN 1 ELSE 0 END) AS total_success, SUM(CASE WHEN "Landing_Out
 * sqlite:///my_data1.db
Done.
```

Out[25]:

total_success	total_failure
61	40

Library used:

csv, sqlite3, prettytable

Key Statement:

SUM(), CASE

Explanation:

CASE is used to detect successes and failures with LIKE function and SUM is used to calculate the total number of Successful and Failed outcomes.

Boosters Carried Maximum Payload

Library used:

csv, sqlite3, prettytable

Key Statement:

MAX()

Explanation:

MAX statement is used to identify the maximum payload mass, and then data is filtered to list all Booster_Versions associated with it the payload mass.

Task 8

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

In [26]:
```
%sql select "Booster_Version" from SPACEXTBL where "PAYLOAD_MASS__KG_" = (select MAX("PAYLOAD_MASS__KG_") from SPACEXTBL)
```

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

Out[26]:

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

2015 Launch Records

Task 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.

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

In [28]:
```
%sql SELECT CASE WHEN substr("Date", 6, 2) = '01' THEN 'January' WHEN substr("Date", 6, 2) = '02' THEN 'February' WHEN subst
```

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

Out[28]:

Month_Name	Landing_Outcome	Booster_Version	Launch_Site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Library used:

csv, sqlite3, prettytable

Key Statement:

substr()

Explanation:

substr() statement is used to get the substring of the values inside column “Date” and extracted month and year from then. Then, the data was filtered with landing_outcome value ‘Failure (drone ship).

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

```
In [32]: %sql SELECT "Landing_Outcome", COUNT("Landing_Outcome") AS outcome_count FROM SPACEXTBL WHERE "Date" BETWEEN '2010-06-04' A
 * sqlite:///my_data1.db
Done.
```

Out[32]:

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

Library used:

csv, sqlite3, prettytable

Key Statement:

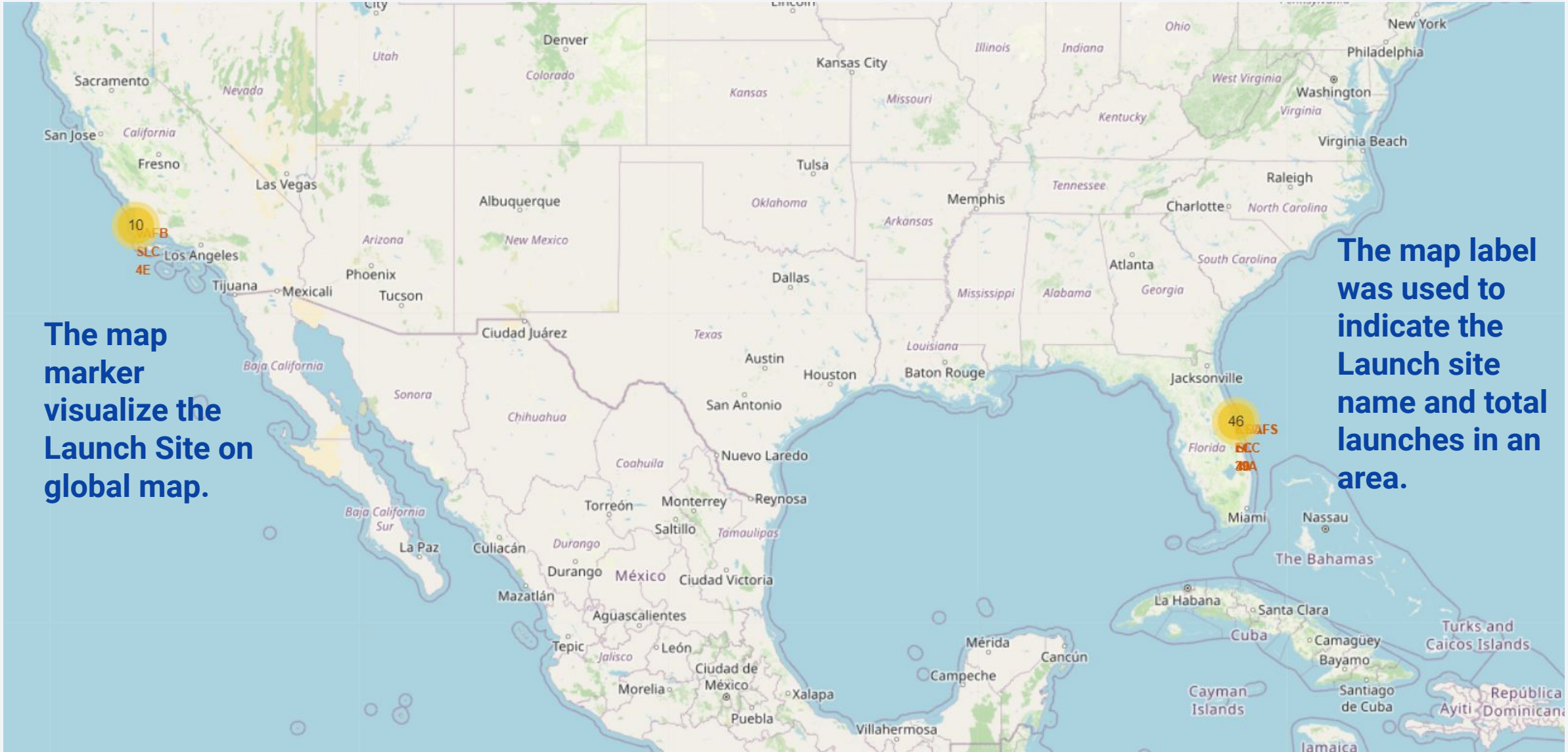
COUNT(), BETWEEN, GROUP BY, ORDER BY

Explanation:

COUNT() statement is used to count all the landing outcomes BETWEEN dates '2010-06-04' AND '2017-03-20', and then data was grouped by column ‘Landing_Outcome’ using GROUP BY statement. Finally data was sorted in descending order using ORDER BY statement.


Section 3
Launch Sites
Proximities Analysis

Launches on Map

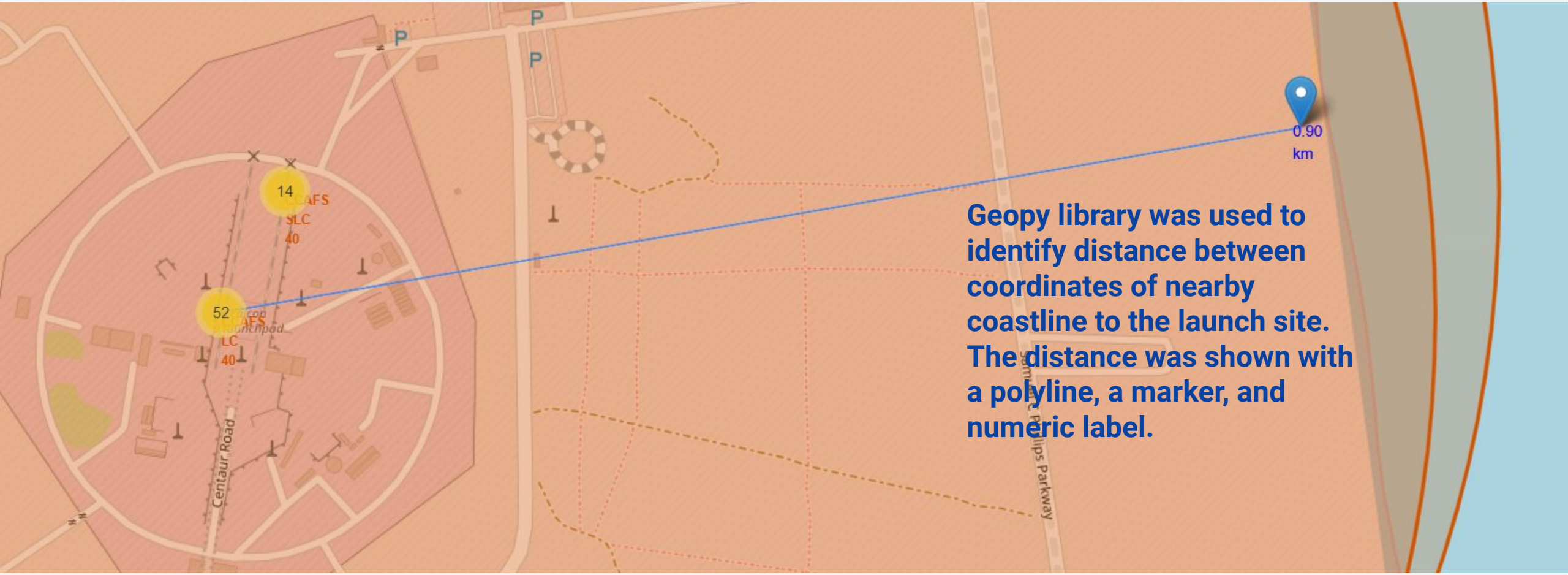


The map marker visualize the Launch Site on global map.

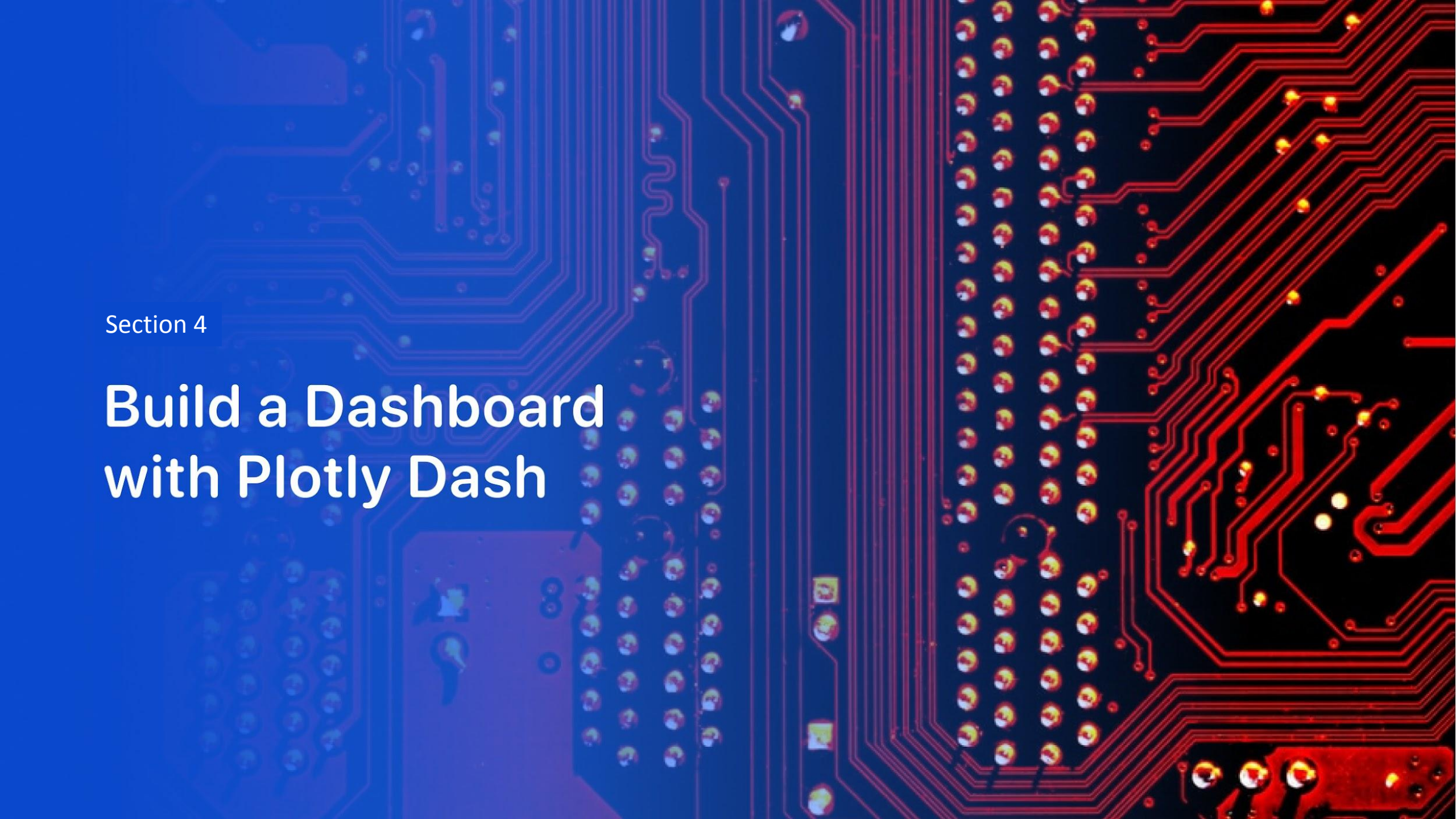
The map label was used to indicate the Launch site name and total launches in an area.

Landing **outcomes** on the Map

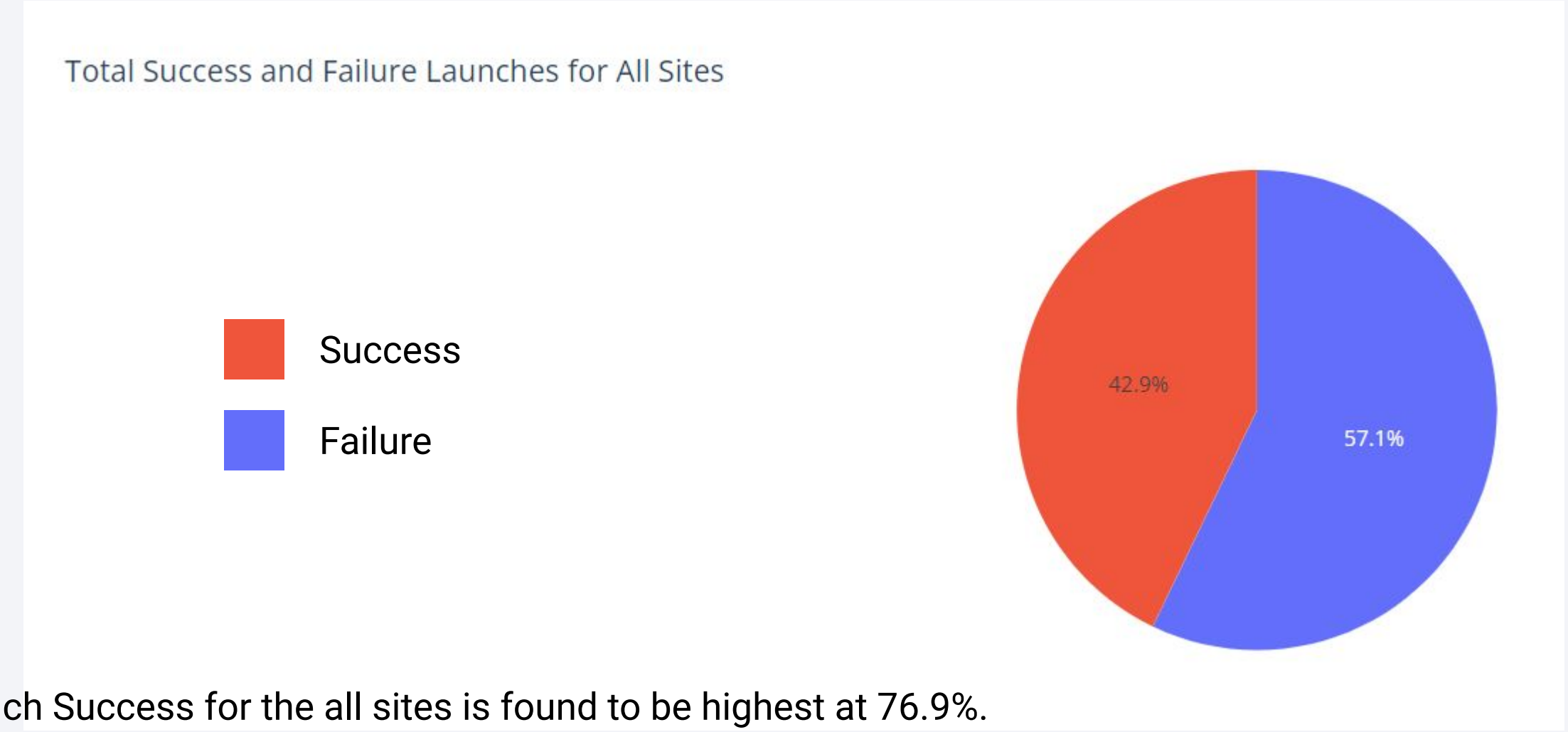
Launch Site from Coastline



Geopy library was used to identify distance between coordinates of nearby coastline to the launch site. The distance was shown with a polyline, a marker, and numeric label.

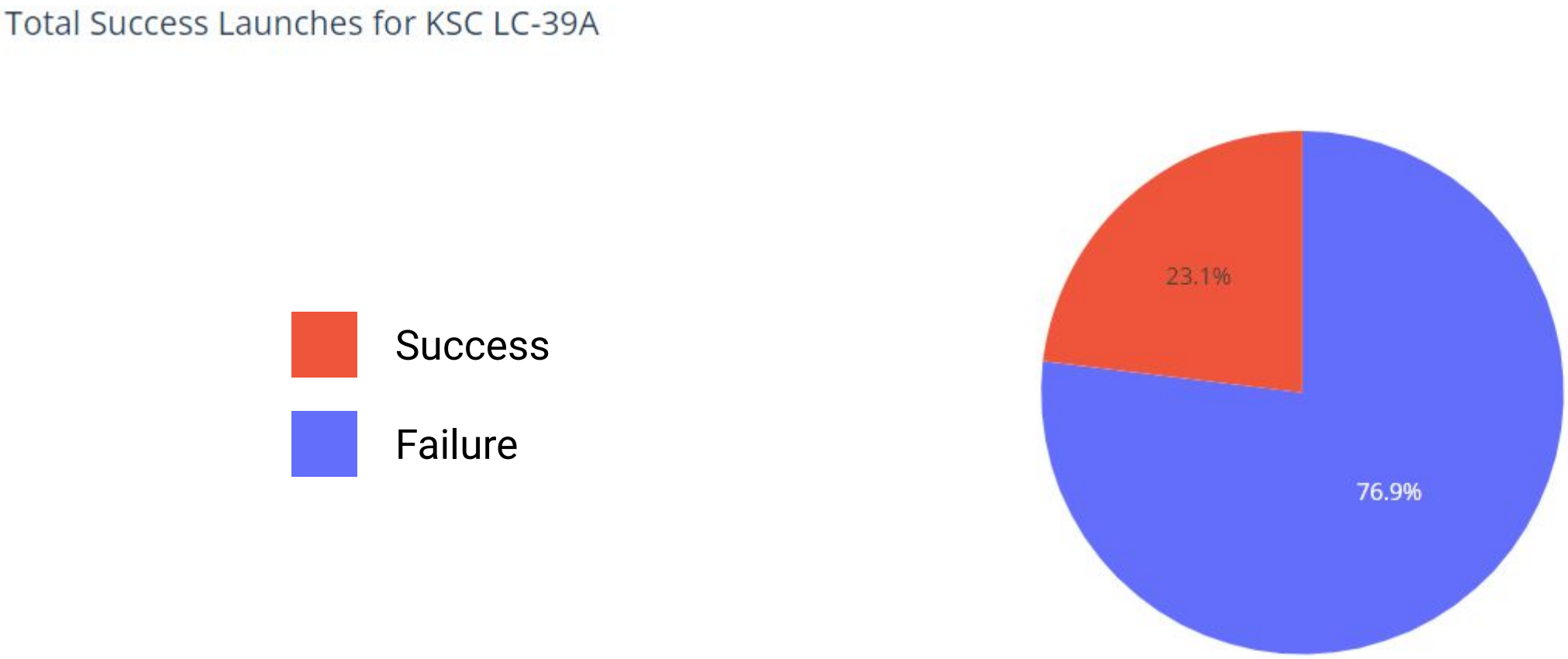

Section 4
Build a Dashboard with Plotly Dash

Success v/s Failure on All Sites



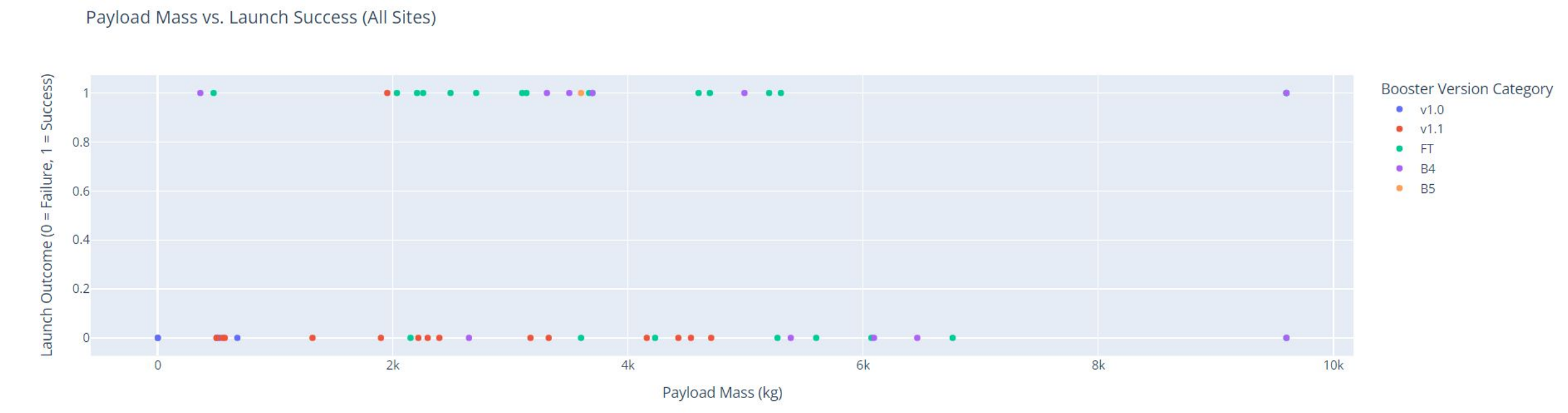
Launch Success for the all sites is found to be highest at 76.9%.

<Dashboard Screenshot 2>



Launch Success for the site KSC LC-39A is found to be 76.9%.

<Dashboard Screenshot 3>



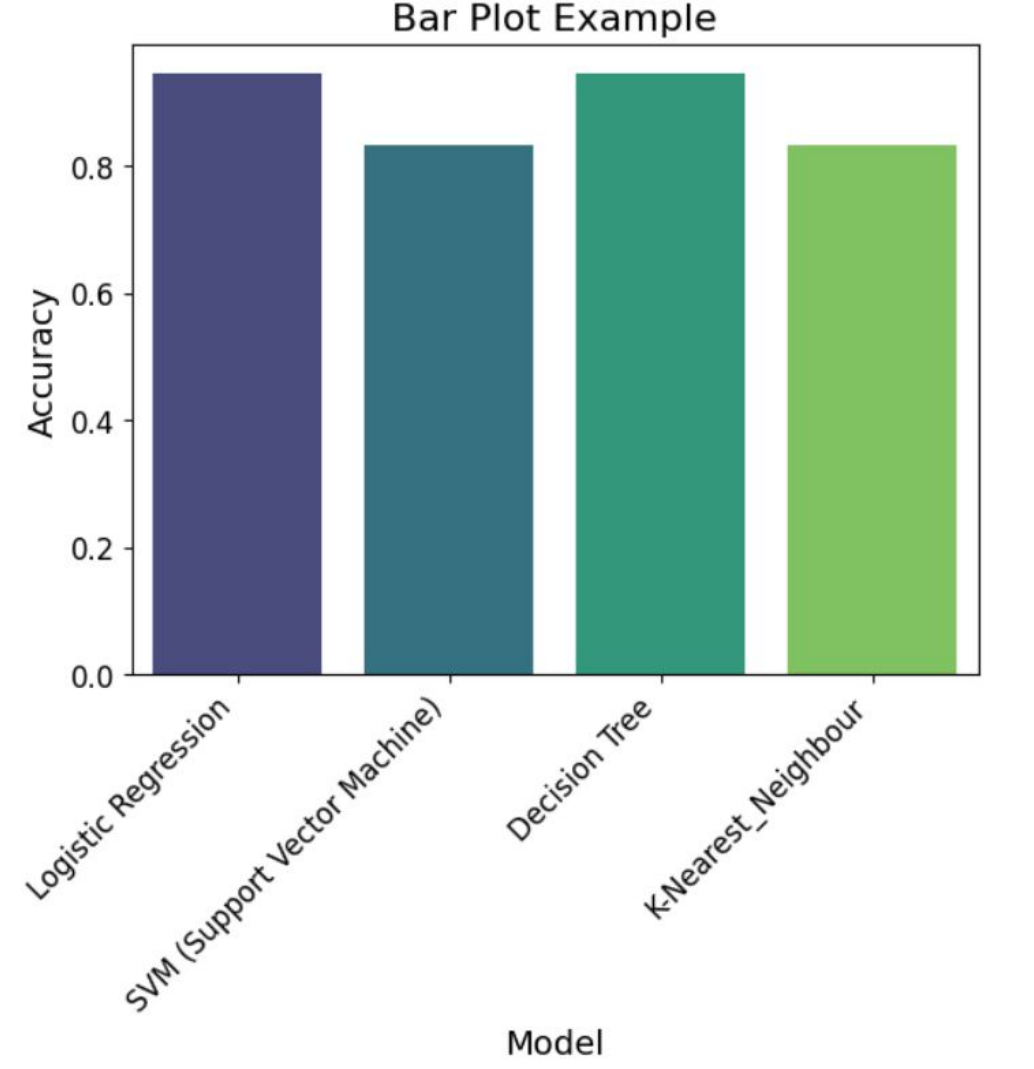
- Booster version FT seems to have high success rate.
- Launch projects with payload mass between 2000 to 6000 kg seems to have more success rate.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

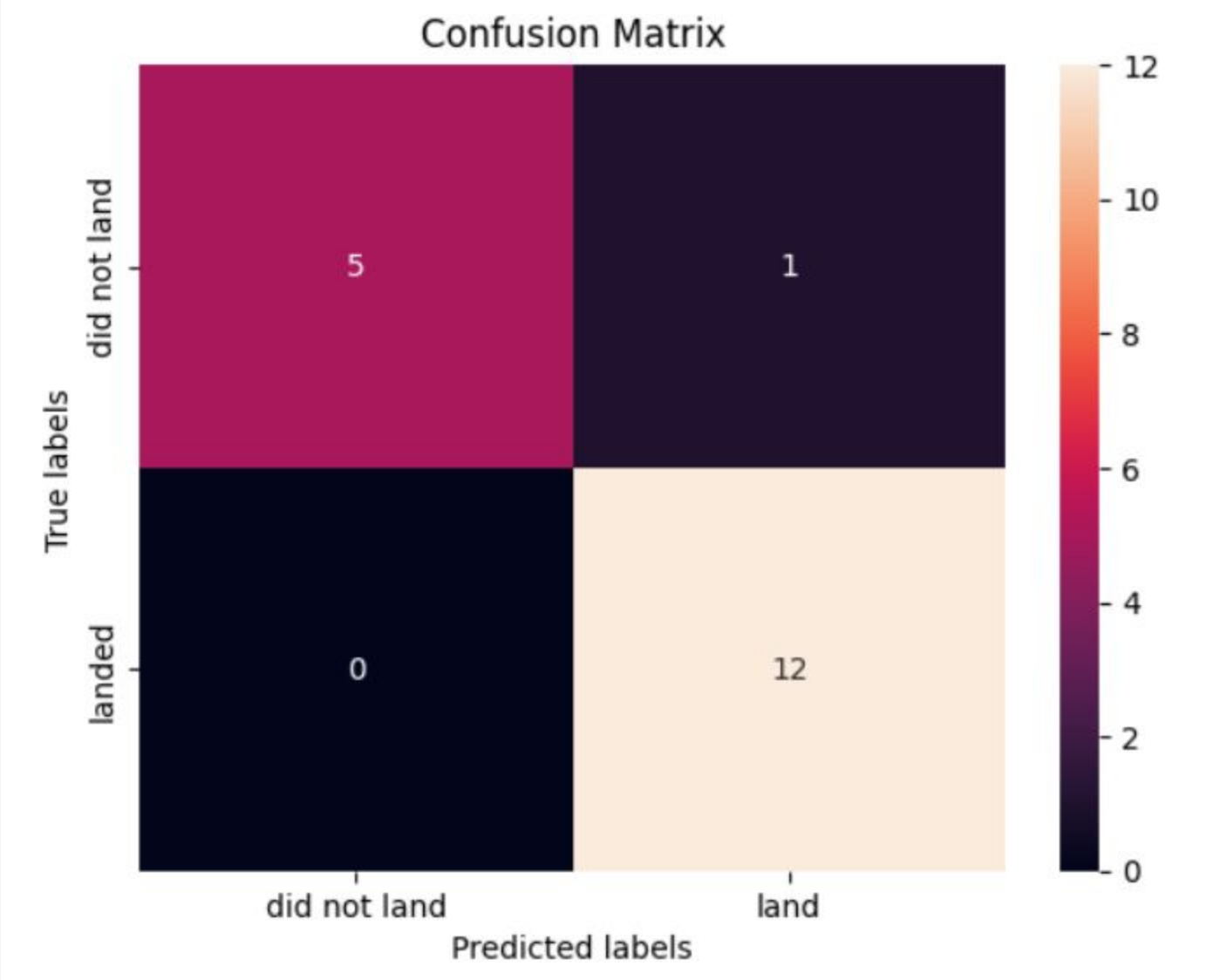
Logistic Regression and Decision Tree has shown the maximum accuracy of 93.33%.



Confusion Matrix

Logistic Regression was observed to be the best model to predict the outcome using optimal parameters.

The confusion matrix shows appropriate outcomes, with only one loss value (false-positive).



Conclusions

- ***Launch Site Performance:*** The "KSC LC-39A" launch site demonstrated the highest success rate, contributing significantly to overall successful landings.
- ***Booster Version Insights:*** Booster versions FT and B4 were the most successful, while F9 V1.1 had the lowest success rate, highlighting the importance of booster design in mission outcomes.
- ***Payload Mass Influence:*** Launches with payload masses between 2,000 kg and 6,000 kg consistently achieved higher success rates, emphasizing an optimal payload range for reliable operations.
- ***Yearly Trends in Success:*** ince 2013, the number of successful landings has increased significantly, with notable peaks in 2017 and 2019.
- ***Modeling Success Prediction:*** Logistic Regression emerged as the most effective model for predicting landing outcomes, achieving an accuracy of 93.33% with minimal misclassification errors as shown in the confusion matrix.

Appendix 1

Github Repository Link: https://github.com/akashverma67/Falcon_Land_Prediction.git

Data Source: https://api.spacexdata.com/v4/launches/past

Python Libraries used in the project with purposes:
requests: Used to fetch data from the SpaceX REST API.
pandas: For data manipulation, exploration, and cleaning.
matplotlib: For creating static plots like scatter and line plots.
seaborn: For enhanced visualizations like bar plots and categorical scatter plots.
folium: For creating interactive maps and adding markers, circles, and polylines.
plotly: For building interactive dashboards (e.g., pie charts, range sliders, scatterplots).
geopy: For calculating distances between geographical coordinates.
SQL Queries and Database Management
sqlite3: For running SQL queries and managing a local SQLite database.
numpy: For numerical computations and handling arrays.

Appendix 2

Python Libraries used in the project with purposes (contd.):

sklearn (Scikit-learn):

SVC: For Support Vector Classification.

DecisionTreeClassifier: For tree-based models.

KNeighborsClassifier: For K-Nearest Neighbors algorithm.

train_test_split: For splitting the dataset into training and test sets.

GridSearchCV: For hyperparameter optimization.

preprocessing: For standardizing and transforming the data.

Interactive Tools and Additional Libraries

prettytable: For tabular display of query results.

csv: For handling CSV files during data saving and export.

Appendix 3

SQL Statements used in the project with purposes:

SELECT DISTINCT: Extracted unique launch site names.

LIKE and LIMIT: Filtered launch sites starting with "CCA" and limited the results to 5.

SUM(): Calculated the total payload mass for NASA (CRS).

AVG(): Found the average payload mass for booster version F9 V1.1.

MAX(): Identified the maximum payload mass and the booster versions associated with it.

MIN(): Found the date of the first successful ground landing.

BETWEEN: Filtered landing outcomes between specific dates.

CASE and SUM(): Classified and calculated total successful and failed mission outcomes.

substr(): Extracted month and year for filtering 2015 launch records.

GROUP BY and ORDER BY: Ranked landing outcomes based on counts within a specific date range.

Thank you!

Akash Verma