IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Gonzalo Ríos
Feb 12th, 2024

Outline

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

Executive Summary

- The main competitive advantage of SpaceX in terms of cost per launch is determined by their ability to recover the booster, which implies considerable savings compared to the competition.
- Being able to predict the successful or unsuccessful recovery of the booster was set as the main objective of this project.
- After collecting data from different sources and working with it, we were able to better understand the main factors driving the outcome, their interactions and trends.
- Several models were trained and tested using the available data, finding one with 94.4% accuracy and good classifying performance.

Introduction

- **Context:**
  - On its website, SpaceX says Falcon9 rocket launches cost 62 MUSD whereas other providers have costs over 165 MUSD.
  - The main reason for the reduced cost is the reutilization of the first stage of the rocket, which, after launch and detachment from the rest of the body, is retrieved, worked on and reused for subsequent launches. The successful landing of the first stage depends on a series of different factors.
  - Being so relevant for the overall cost, being able to predict whether or not the first stage will land or not is of paramount importance. This is the main objective of this project.

- **Main topics of interest**:
  - Understanding the main factors related to the success or failure of the landing.
  - Understanding the relationship between the different features available.
  - Understanding the practical conditions to maximize the success of the landing.

Section 1

Methodology

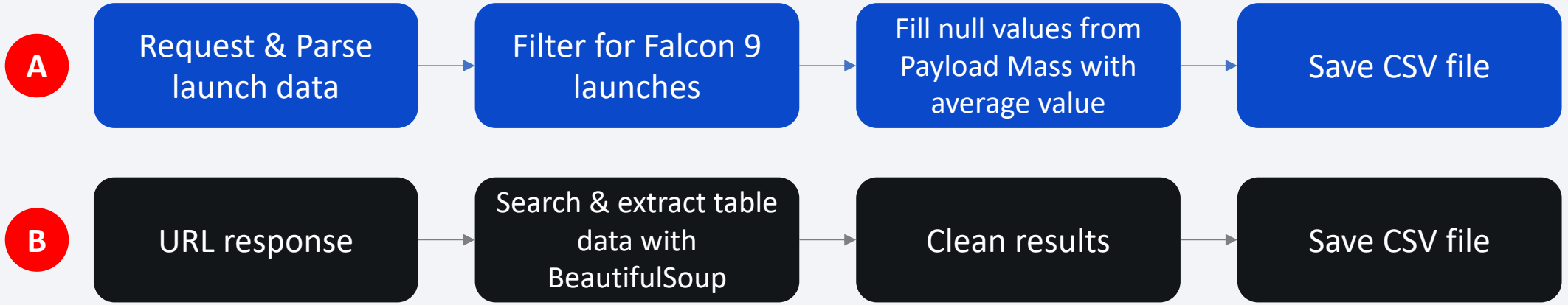
Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API
  - Web Scrapping (Wikipedia)
- Perform data wrangling
  - Filtering, cleaning, null value treatment, data type setting, feature selection, one-hot encoding.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Different models, GridSearch CV for hyperparameter selection and validation.

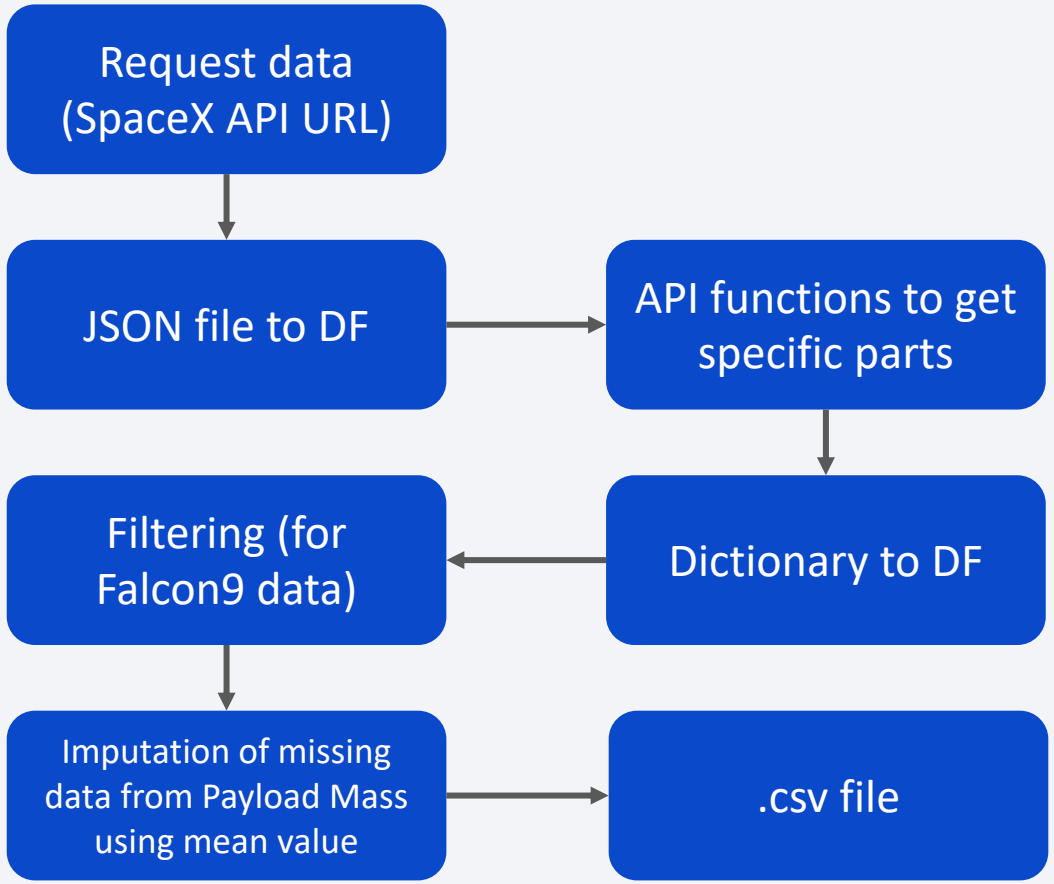
Data Collection

- The data for this project was collected from two main sources:
  - SpaceX REST API **A**
  - Wikipedia page (Web Scrapping) **B**



Data Collection – SpaceX API

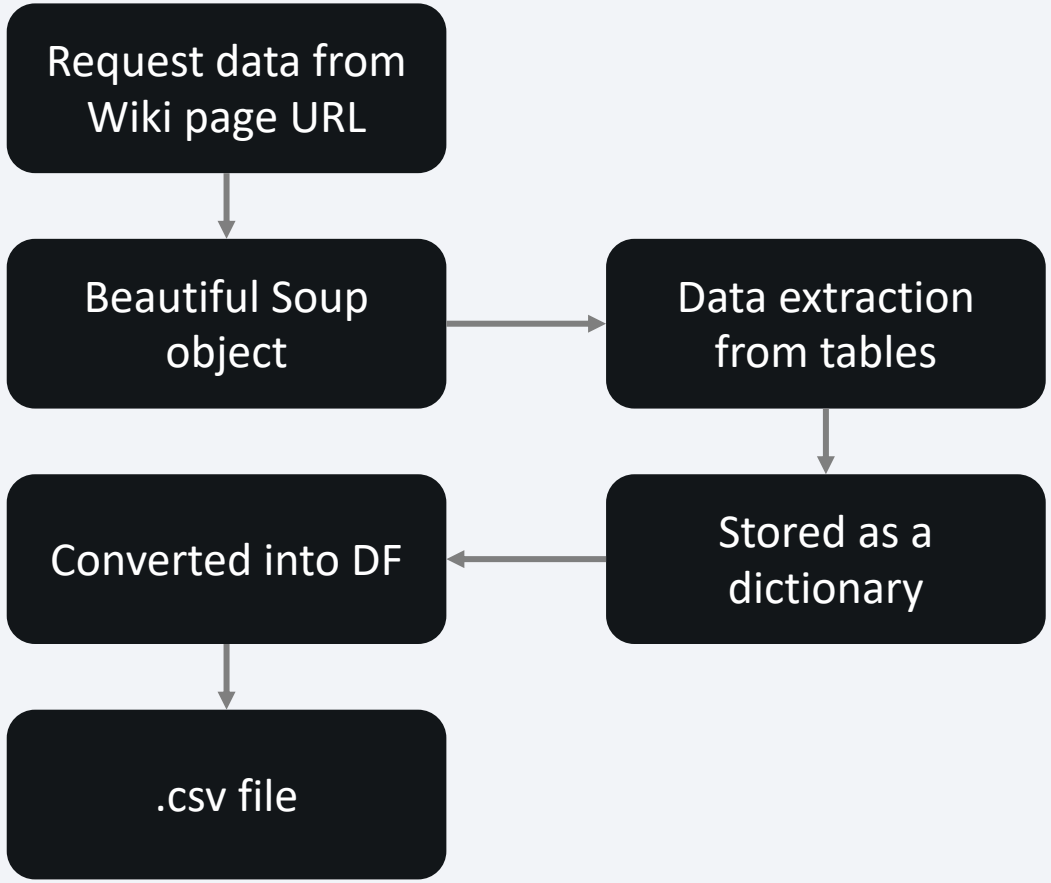
- Request made using SpaceX API URL.
- Separate functions defined in order to get specific features of interest
  - getBoosterVersion
  - getLaunchSite
  - getPayloadData
  - getCoreData



GitHub ref: https://github.com/GRiosG/Public/blob/main/IBM_DataScience_Certificate_Capstone_Project/Capstone_Project_Files/1_Data_Collecting_Spacex_API.ipynb

Data Collection - Scraping

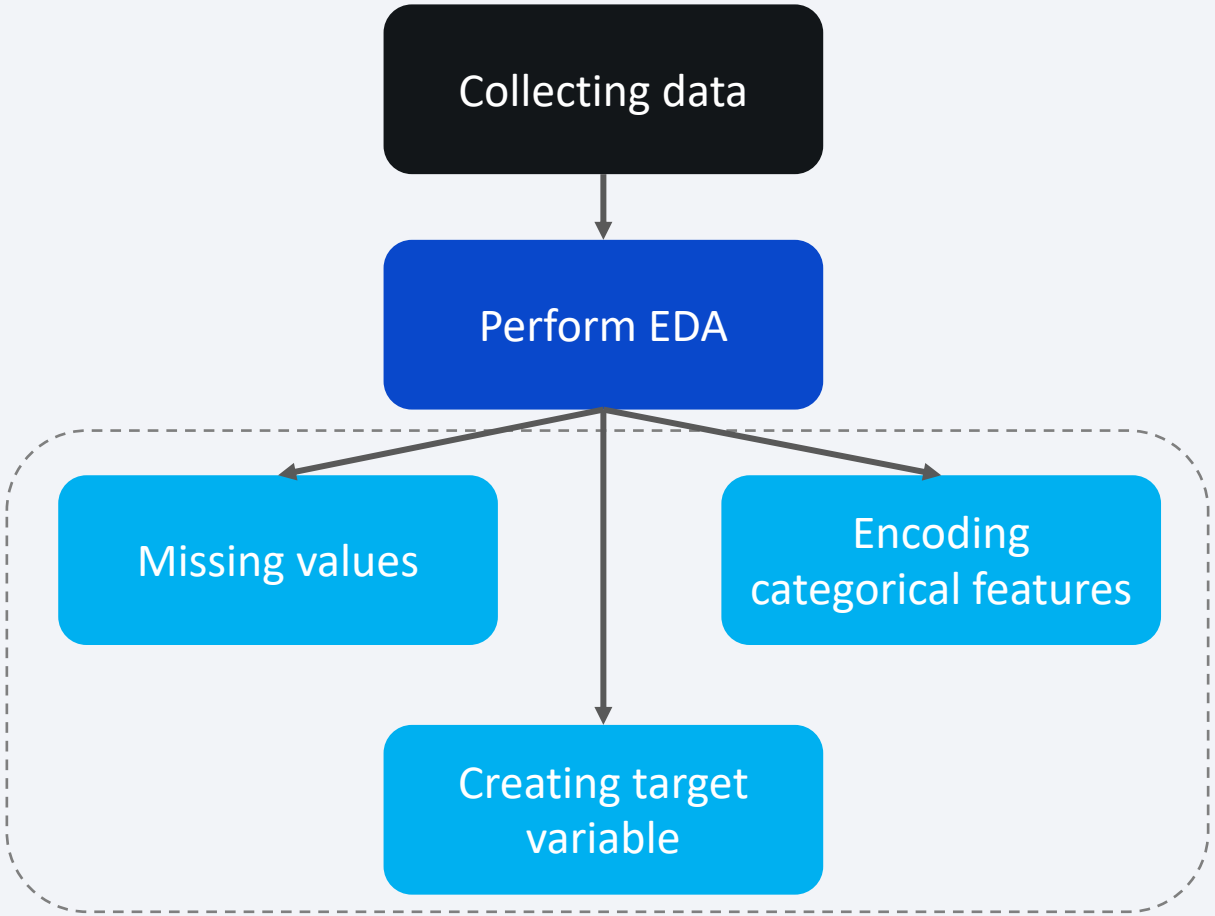
- Web scrapping stage used the URL of Wikipedia page with historical launch data.
- Beautiful Soup was used to parse and extract information from tables.
- The resulting data was finally converted into a dataframe and saved for later use.



GitHub ref: https://github.com/GRiosG/Public/blob/main/IBM_DataScience_Certificate_Capstone_Project/Capstone_Project_Files/2_Data_Collecting_Web_Scrapping.ipynb

Data Wrangling

- Main actions:
  - General data exploration
  - Identifying and dealing with missing values
  - Using 1-hot encoding to transform categorical features
  - Creating a target variable (landing_class) to be used for model training later:
    - 1 if the landing was classified as successful
    - 0 if unsuccessful



GitHub ref: https://github.com/GRiosG/Public/blob/main/IBM_DataScience_Certificate_Capstone_Project/Capstone_Project_Files/3_Data_Wrangling.ipynb

EDA with Data Visualization

- Several plots were made in order to understand the relationship between the main features of the dataset.
- Scatter plots:
  - Flight number and Payload
  - Flight number and Launch Site
  - Payload and Launch Site
  - Flight number and Orbit Type
  - Payload and Orbit Type
- Bar plots:
  - Success rate for each orbit type
- Line plots:
  - Yearly trend of successful landings

GitHub ref: https://github.com/GRiosG/Public/blob/main/IBM_DataScience_Certificate_Capstone_Project/Capstone_Project_Files/4_EDA%20Dataviz.ipynb

EDA with SQL

- The following SQL queries were performed:
  - %sql select * from SPACEXTBL;
  - %sql select distinct Launch_Site from SPACEXTBL;
  - %sql select * from SPACEXTBL where Launch_Site like 'CCA%' limit 5;
  - %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where Customer = 'NASA (CRS)';
  - %sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where Booster_Version like 'F9 v1.1%';
  - %sql select min(Date) from SPACEXTBL where Landing_Outcome is 'Success (ground pad)';
  - %sql select distinct Booster_Version from SPACEXTBL where Landing_Outcome is 'Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000;
  - %sql select Mission_Outcome, count(*) as Count from SPACEXTBL group by Mission_Outcome order by Count desc;
  - %sql select count(Booster_Version) from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
  - %sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL);

GitHub ref: https://github.com/GRiosG/Public/blob/main/IBM_DataScience_Certificate_Capstone_Project/Capstone_Project_Files/5_EDA%20SQL_v2.ipynb

EDA with SQL (cont.)

- The following SQL queries were performed:
  - select
    case **substr**(Date, 6, 2)
    when '01' then 'January'
    when '02' then 'February'
    when '03' then 'March'
    when '04' then 'April'
    when '05' then 'May'
    when '06' then 'June'
    when '07' then 'July'
    when '08' then 'August'
    when '09' then 'September'
    when '10' then 'October'
    when '11' then 'November'
    when '12' then 'December'
    end as MonthName,
    Landing_Outcome,
    Booster_Version,
    Launch_Site
    from SPACEXTBL
    where **substr**(Date, 0, 5) = '2015'
    and Landing_Outcome is 'Failure (drone ship)';
  - %sql select Landing_Outcome, count(*) as Count from SPACEXTBL where Date between '2010-06-04' and '2017-03-20' group by Landing_Outcome order by Count desc;

GitHub ref: https://github.com/GRiosG/Public/blob/main/IBM_DataScience_Certificate_Capstone_Project/Capstone_Project_Files/5_EDA%20SQL_v2.ipynb

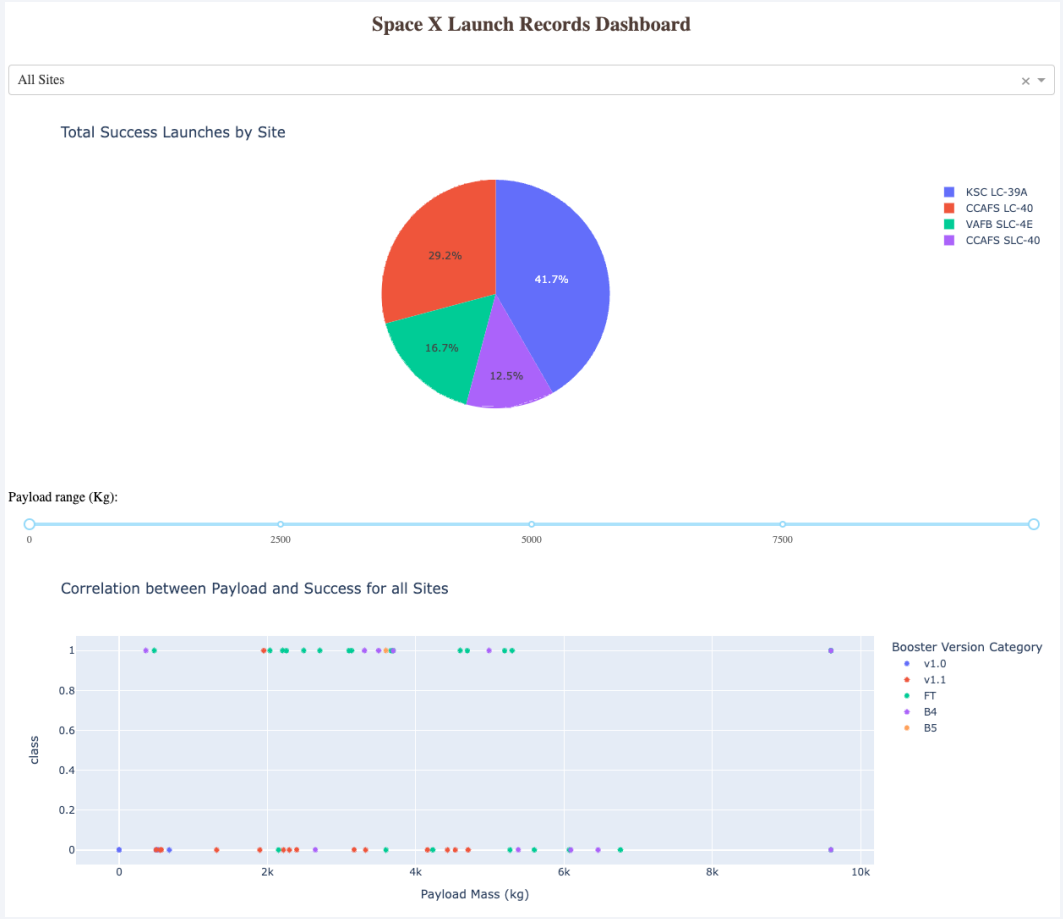
Build an Interactive Map with Folium

- Map objects created and added to the folium map:
  - folium.Circle / folium.Marker: For indicating and labeling specific areas or the map (launch sites).
  - MarkerCluster: to create a cluster of points on the map, facilitating the viewing when markers were close to each other.
  - MousePosition: used to get coordinates from the map based on the position of the cursor.
  - folium.PolyLine: used to draw a line between two points (coordinates obtained from MousePosition).

GitHub ref: https://github.com/GRiosG/Public/blob/main/IBM_DataScience_Certificate_Capstone_Project/Capstone_Project_Files/6_Folium_Maps.ipynb

Build a Dashboard with Plotly Dash

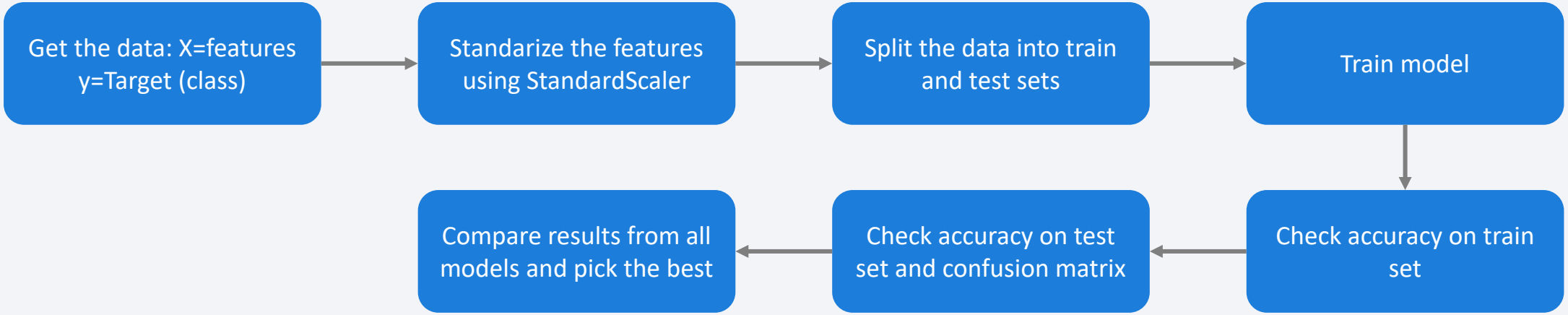
- An interactive dashboard was built using Plotly and Dash libraries.
- It includes:
  - A dropdown selector of launch sites (to select all or a specific one)
  - A pie chart indicating the total success launches by site (when all sites are selected), or ratio of successful launches by site.
  - A range selector (for selecting payload)
  - And a scatter plot showing the relationship between the payload mass, and the class (successful or unsuccessful), depending on the launch site selected and the payload range.



GitHub ref: https://github.com/GRiosG/Public/blob/main/IBM_DataScience_Certificate_Capstone_Project/Capstone_Project_Files/7_spacex_dash_app.py

Predictive Analysis (Classification)

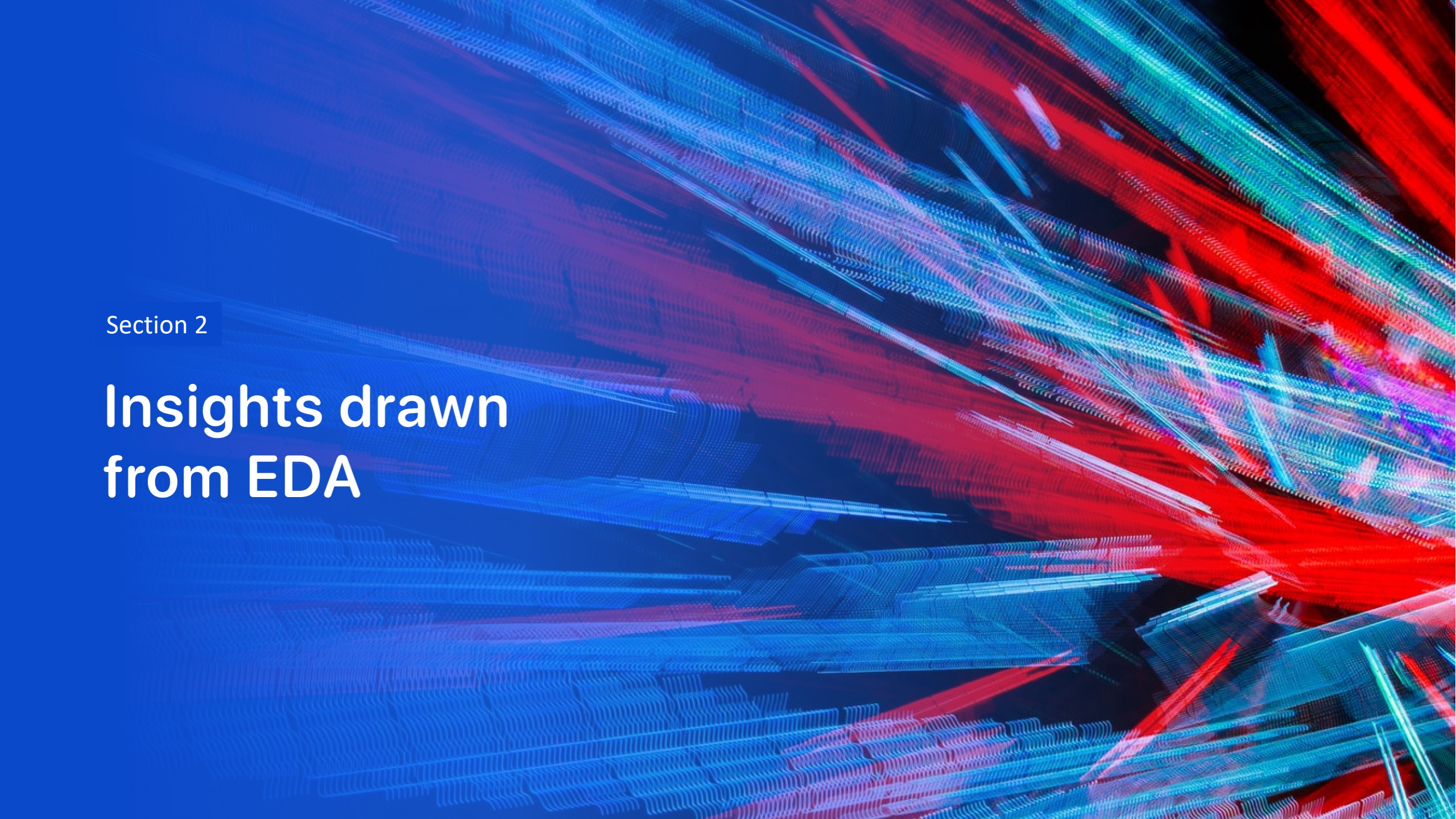
- Several models were trained and tested:
  - SVM, Decision Tree, Logistic Regression, KNN
- The overall process was the following:



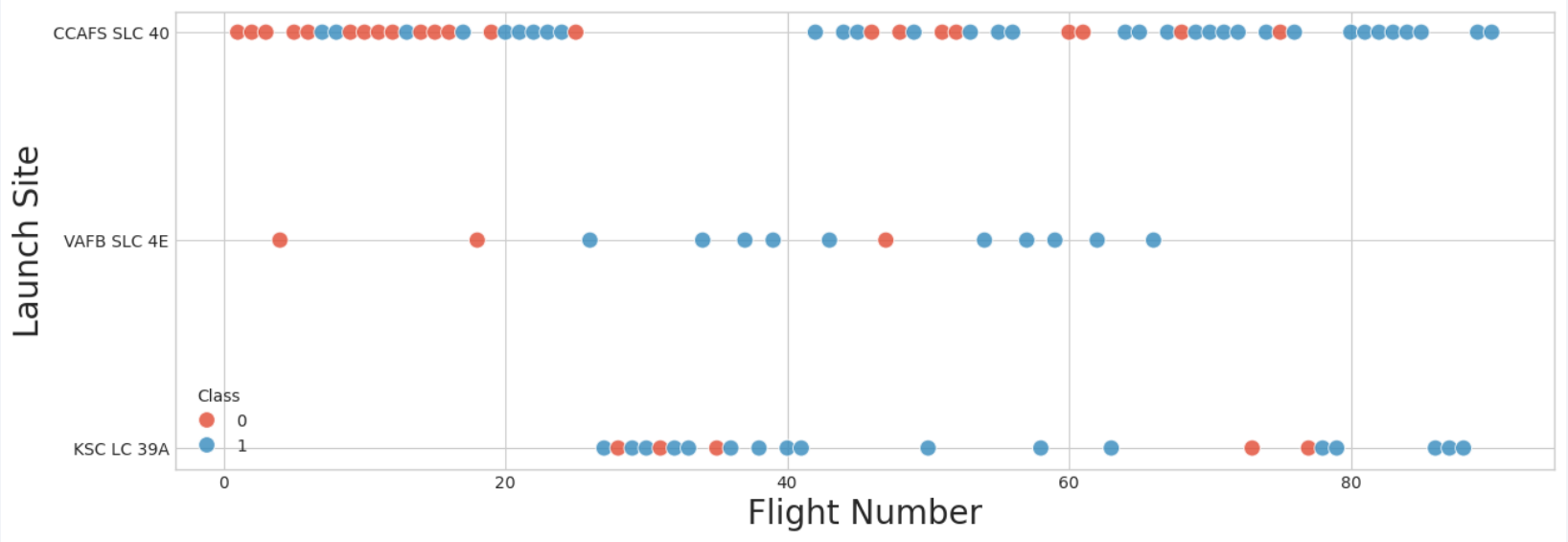
GitHub ref: https://github.com/GRiosG/Public/blob/main/IBM_DataScience_Certificate_Capstone_Project/Capstone_Project_Files/8_ML%20Prediction.ipynb

Results

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

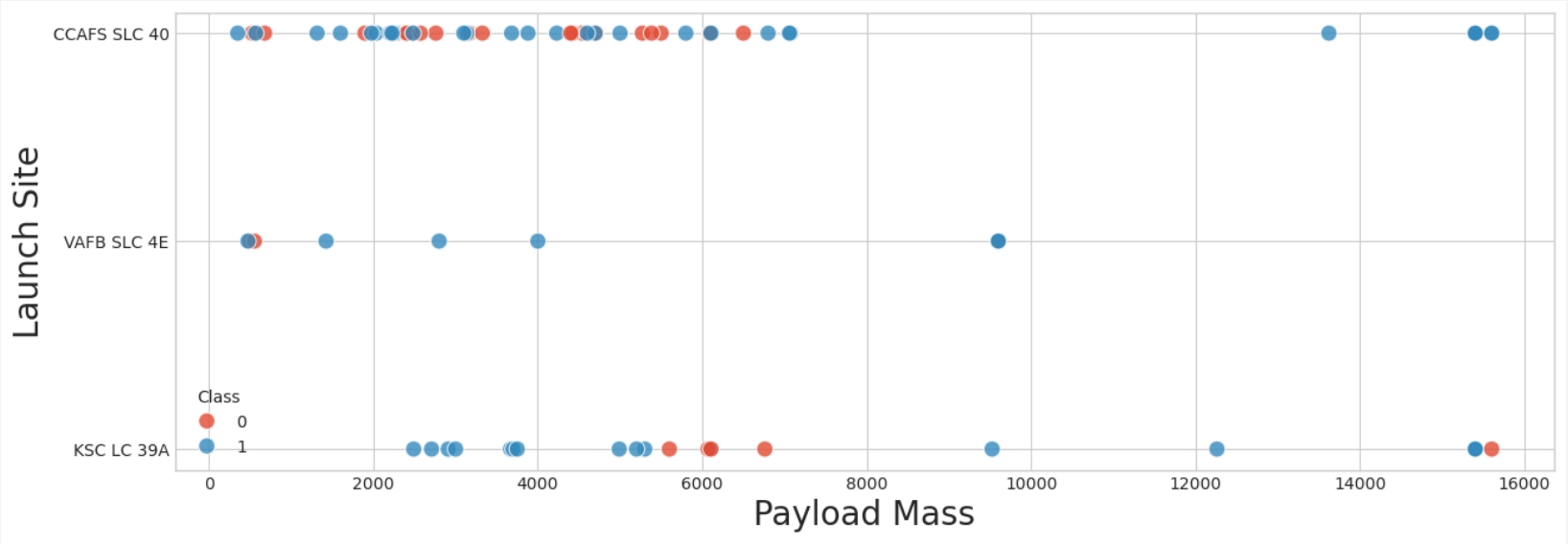

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



- CCADF-SLC is the most used launch site. It visibly concentrates most of the failures, particularly during the first quarter of the project. Its success rate improves overtime.
- VAFB-SLC is the one with the least amount of flights.
- VAFB-SLC and KSC LC 39A present fewer failed landings, but also have much less launches compared to CCADF-SLC.

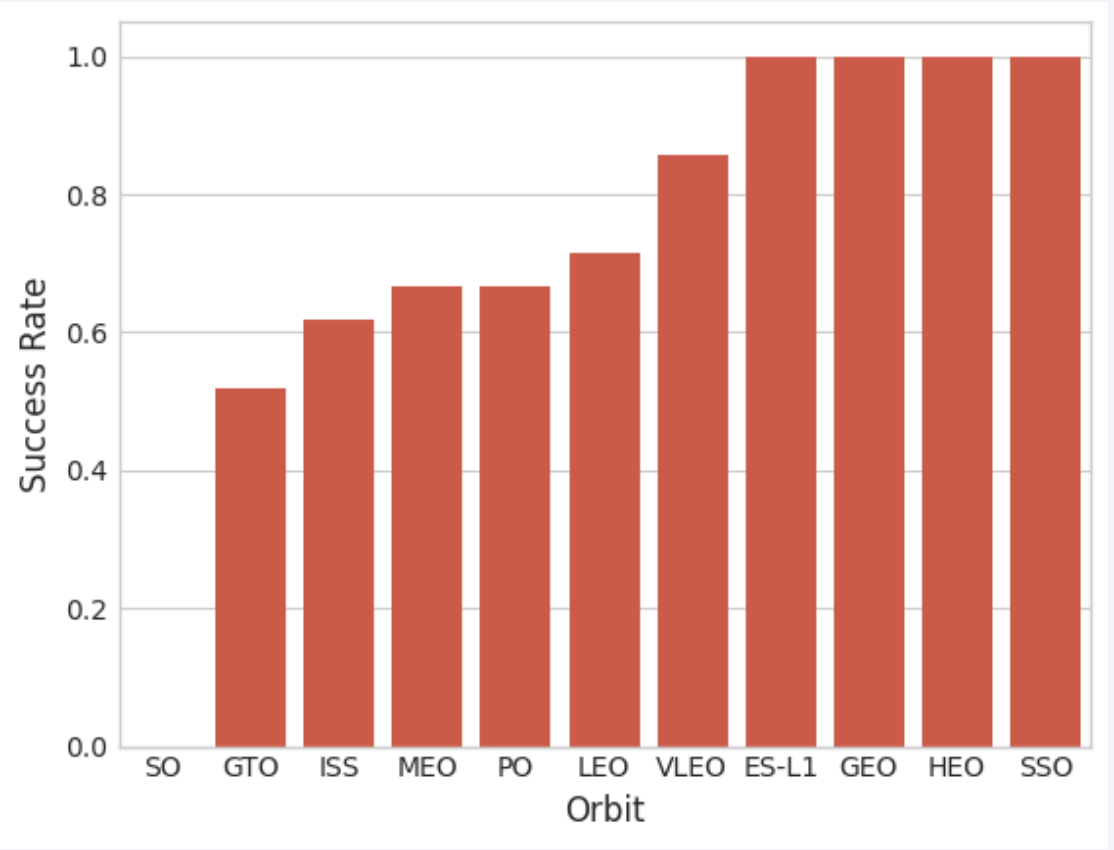
Payload vs. Launch Site



- The chart brings additional info regarding Payload mass per launch site, as well as success/failure.
- Apparently higher payloads have higher success rate, particularly above 8000 Kg. This of course could be due to other factors, so this is just an observation.

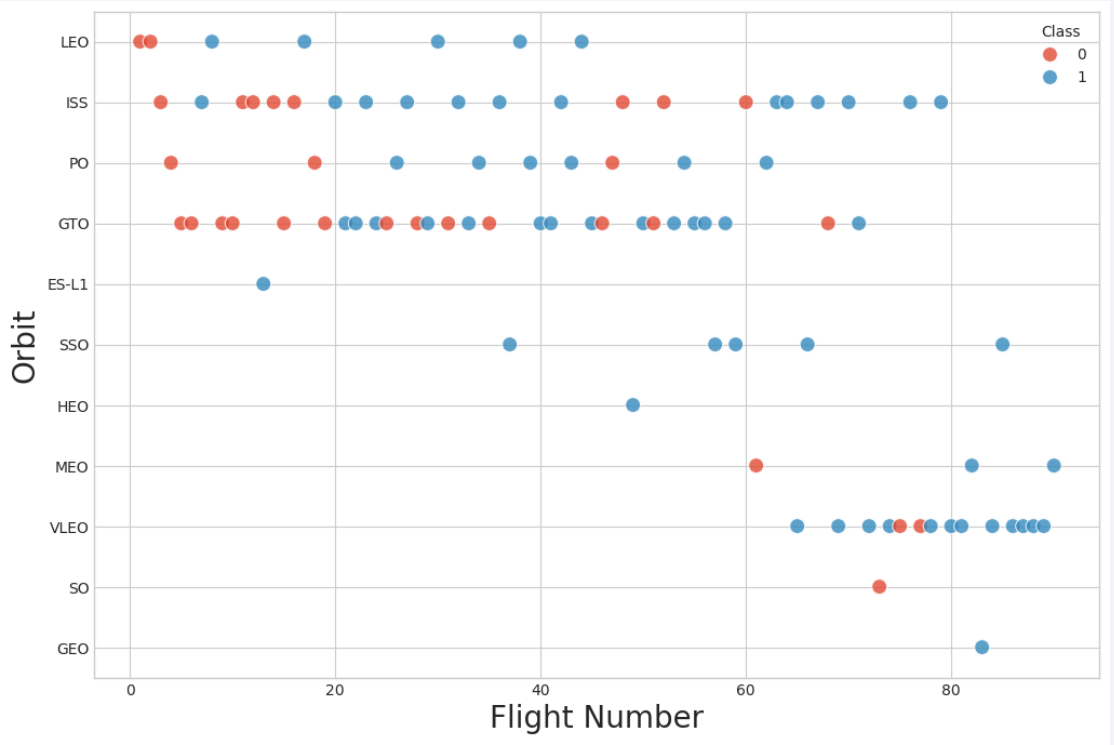
Success Rate vs. Orbit Type

- Ignoring the number of attempts (which is relevant and not visible from this plot):
  - SO has zero percent success rate (but only 1 launch)
  - ES-L1, GEO, HEO and SSO present the highest success rate among all types.



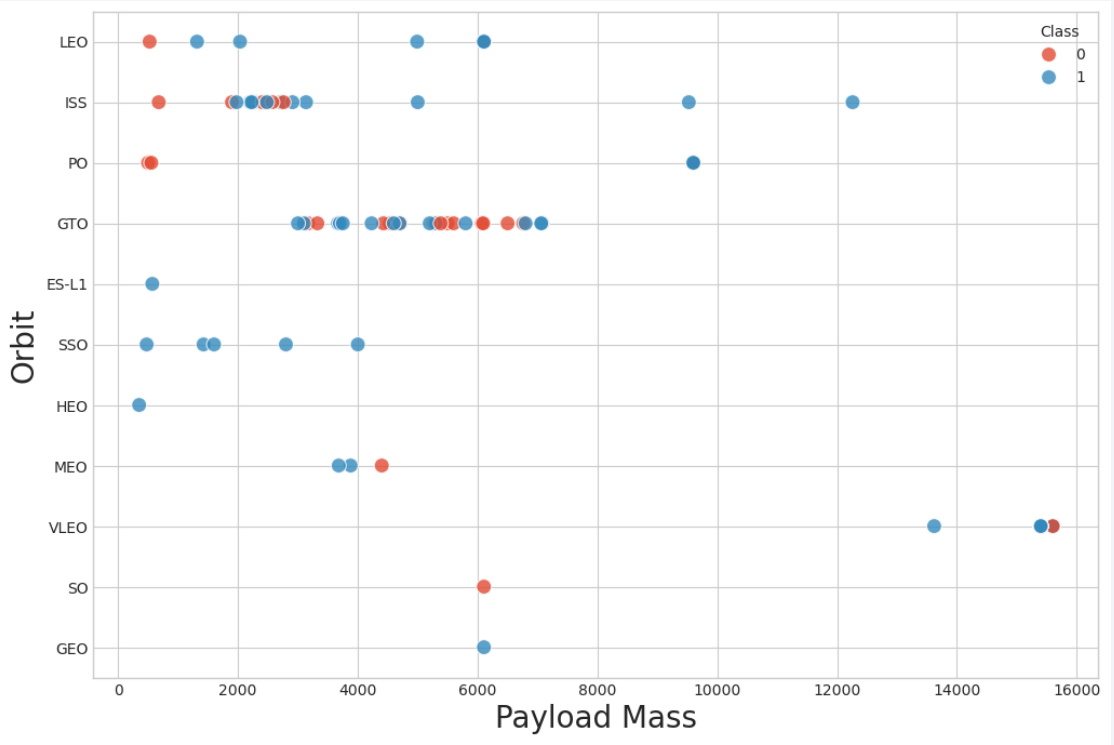
Flight Number vs. Orbit Type

- GEO, SO, HEO, ESL-1 and MEO have a low number of attempts, so the results are not representative necessarily.
- Most of the unsuccessful missions tend to occur at the first half of the flight number counter for each orbit (which is to be expected)
- Overall, the more flights, the more experience and higher probability of success.
- ISS and GTO concentrate most of unsuccessful attempts.



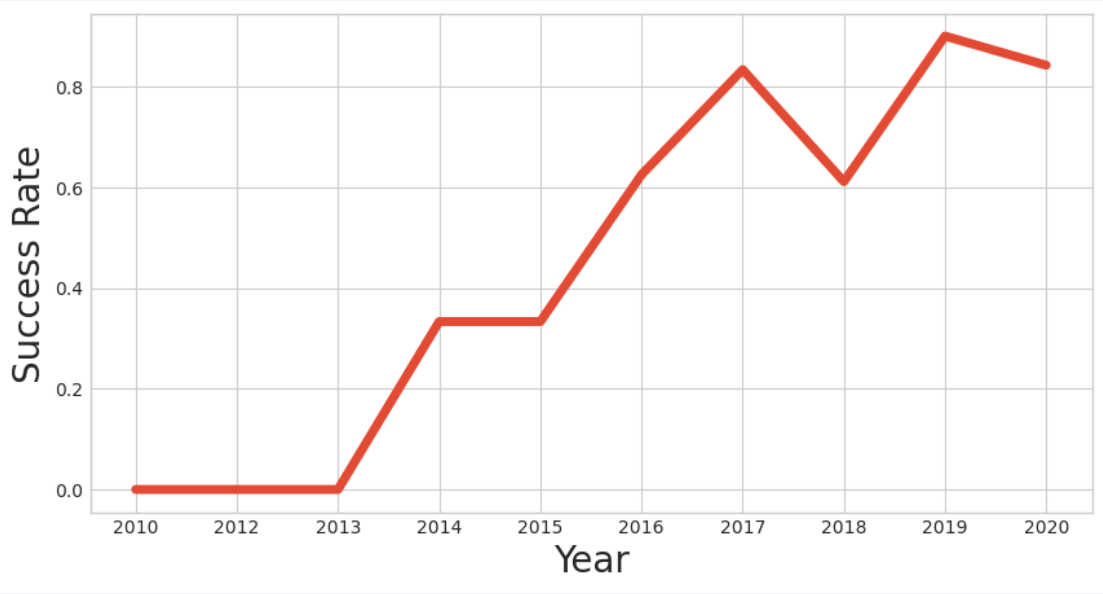
Payload vs. Orbit Type

- There's a clear concentration of under-8000 kg payload mass, and has a comparatively lower success rate vs higher payloads for all orbits overall.
- Only 3 orbit types have attempts with more than 8000 kg payloads.



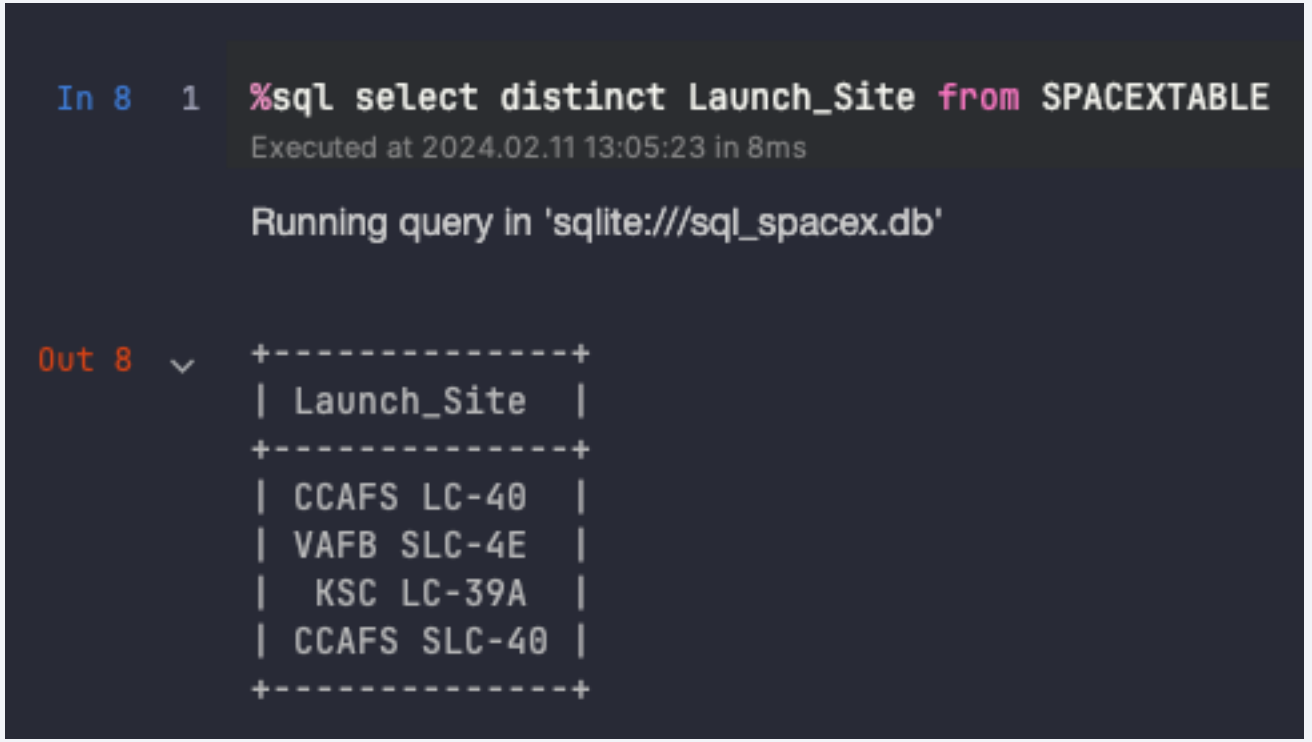
Launch Success Yearly Trend

- Success rate significantly improves over the years from 2013 onwards (having only unsuccessful attempts until then).
- The 50% rate is reached between 2015 and 2016
- The last two years exhibit a high success rate, well above 80%.
- The overall trend is ascending, with a small dip in 2018.



All Launch Site Names

- Selected all distinct launch sites (from the Launch_Site column).
- Used select and distinct.
- 4 Different sites in the data



Launch Site Names Begin with 'CCA'

```
%sql select * from SPACEXTABLE where Launch_Site like 'CCA%' limit 5
Executed at 2024.02.11 13:17:42 in 10ms

Running query in 'sqlite:///sql_spacex.db'
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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 (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- Selected all records where launch sites begin with ‘CAA’ using like and limiting the results to 5 using limit.

Total Payload Mass

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer is 'NASA (CRS)'
Executed at 2024.02.11 13:20:03 in 8ms

Running query in 'sqlite:///sql_spacex.db'


+-------------------------+
| sum(PAYLOAD_MASS__KG_) |
+-------------------------+
|          45596          |
+-------------------------+
```

- Selected the total payload mass carried by boosters launched by NASA (CRS) using sum(PAYLOAD_MASS__KG_) and where Customer is 'NASA (CRS)'
- 45596 total

Average Payload Mass by F9 v1.1

```
%sql select round(avg(PAYLOAD_MASS__KG_),2) as 'Average Payload Mass by F9 v1.1' from SPACEXTABLE where Booster_Version like 'F9 v1.1%'
Executed at 2024.02.11 13:23:30 in 12ms

Running query in 'sqlite:///sql_spacex.db'


+---------------------------------+
| Average Payload Mass by F9 v1.1 |
+---------------------------------+
|             2534.67            |
+---------------------------------+
```

- Selected the average payload mass carried by booster version F9 v1.1
- Used an alias to display the name as ‘Average Payload Mass by F9 v1.1 using as
- Rounded the result to be more legible using round with 2 decimal points
- Matched the required Booster_Version using like

First Successful Ground Landing Date

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

Executed at 2024.02.11 13:26:41 in 8ms

Running query in 'sqlite:///sql_spacex.db'


+-------------+
| min(Date)   |
+-------------+
| 2015-12-22 |
+-------------+
```

- Selected the date when the first successful landing outcome in ground pad was achieved
- Used the min function on the Date
- Used where Landing_Outcome is ‘Success (ground pad)’ to match the required outcome.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select Booster_Version from SPACEXTABLE where PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000 and Landing_Outcome is
 "Success (drone ship)"
Executed at 2024.02.11 13:29:35 in 22ms

Running query in 'sqlite:///sql_spacex.db'


+----------------+
| Booster_Version |
+----------------+
|   F9 FT B1022   |
|   F9 FT B1026   |
|  F9 FT  B1021.2 |
|  F9 FT  B1031.2 |
+----------------+
```

- Selected the names of the boosters which have success in drone ship and have payload mass greater between 4000 and 6000 kg.
- Used the operators `>` and `<` to limit the range of payload mass
- Used `and` to add the condition for Landing_Outcome being ‘Success (drone ship)’

Total Number of Successful and Failure Mission Outcomes

```
%sql select Mission_Outcome, count(*) as Count from SPACEXTBL group by Mission_Outcome order by Count desc;
Executed at 2024.02.11 13:44:22 in 8ms

Running query in 'sqlite:///sql_spacex.db'


+----------------------------------+-------+
|         Mission_Outcome          | Count |
+----------------------------------+-------+
|             Success              |   98  |
| Success (payload status unclear) |   1   |
|             Success              |   1   |
|       Failure (in flight)        |   1   |
+----------------------------------+-------+
```

- Selected the total number of successful and failure mission outcomes
- Used the count function and group by Mission_Outcome
- Ordered the results using desc for better visualization.
- Almost all of the results are positive ones, except for one.

Boosters Carried Maximum Payload

- Selected the names of the booster_versions which have carried the maximum payload mass.
- Used a subquery, count and max functions.
- 12 results but only 10 displayed (on my notebook)

```
%sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
Executed at 2024.02.11 13:47:36 in 24ms

Running query in 'sqlite:///sql_spacex.db'

+-----------------+
| 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  |
+-----------------+
Truncated to displaylimit of 10.
```

```
%sql select count(Booster_Version) from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
Executed at 2024.01.30 19:00:45 in 10ms

Running query in 'sqlite:///DDBB_1.db'

+------------------------+
| count(Booster_Version) |
+------------------------+
|           12           |
+------------------------+
```

2015 Launch Records

```
%%sql
select
case substr(Date, 6, 2)
when '01' then 'January'
when '02' then 'February'
when '03' then 'March'
when '04' then 'April'
when '05' then 'May'
when '06' then 'June'
when '07' then 'July'
when '08' then 'August'
when '09' then 'September'
when '10' then 'October'
when '11' then 'November'
when '12' then 'December'
end as MonthName,
Landing_Outcome,
Booster_Version,
Launch_Site
from SPACEXTBL
where substr(Date, 0, 5) = '2015'
and Landing_Outcome is 'Failure (drone ship)';
```

Executed at 2024.02.11 13:54:07 in 17ms

Running query in 'sqlite:///sql_spacex.db'

MonthName	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

- Selected the month names, failure landing_outcomes in drone ship, booster versions, launch_site for the months in the year 2015.
- Since SQLLite doesn't support monthnames, I used the substr function y also used case to display the actual names of the months, instead of their numbers (for better legibility of results).

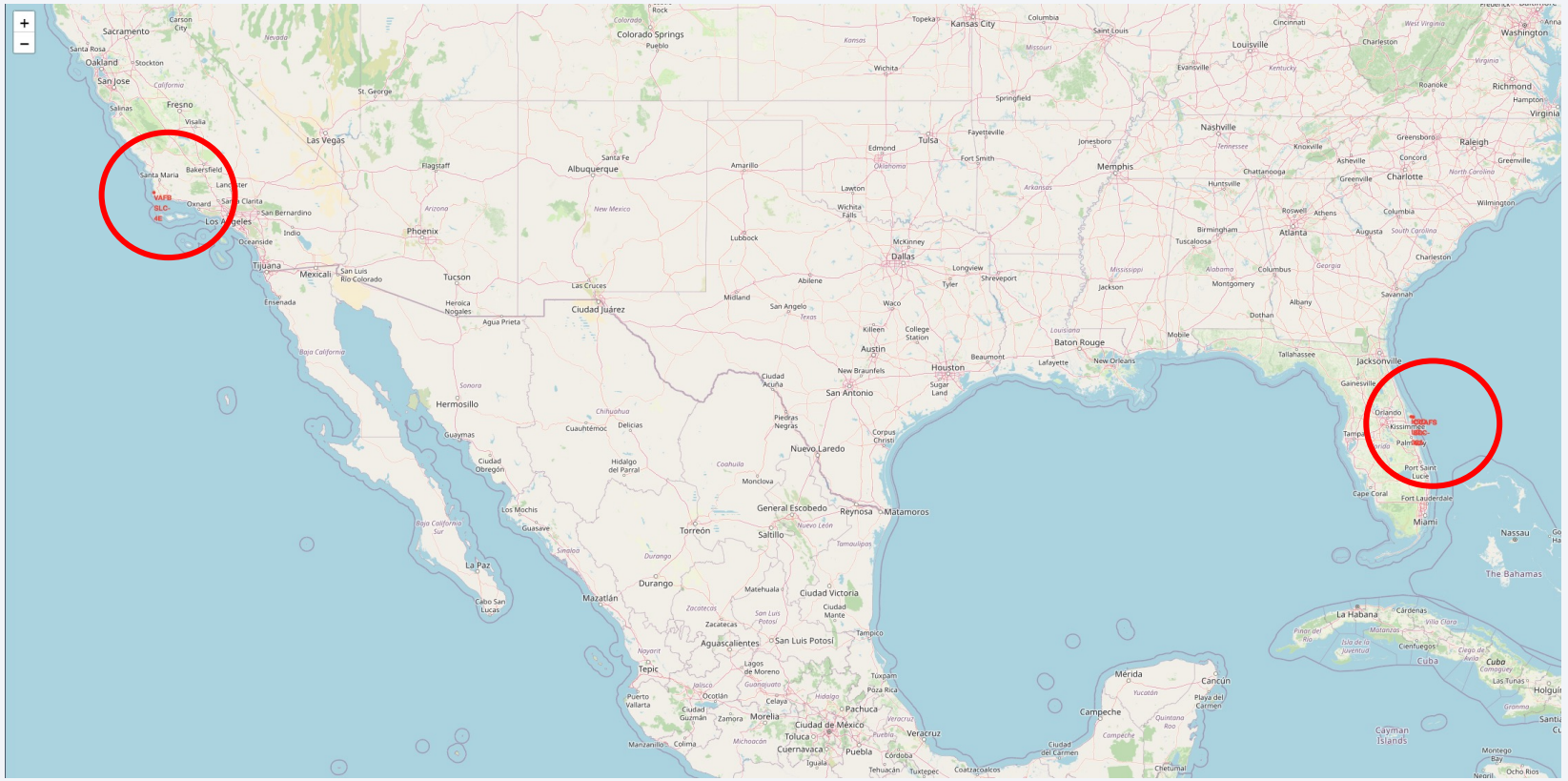
Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

```
%sql select Landing_Outcome, count(*) as Count from SPACEXTBL where Date between '2010-06-04' and '2017-03-20' group by
Landing_Outcome order by Count desc;
Executed at 2024.02.11 13:59:09 in 22ms

Running query in 'sqlite:///sql_spacex.db'


+-----------------------+-------+
|    Landing_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   |
+-----------------------+-------+
```

- Selected the ranking for count of landing outcomes between 2010-06-04 and 2017-03-20 in descending order.
- count function to display the counts required as a column
- Used group by Landing_Outcome
- where function to filter the results using between to limit the range
- Used order by Count desc to present the results in descending order.

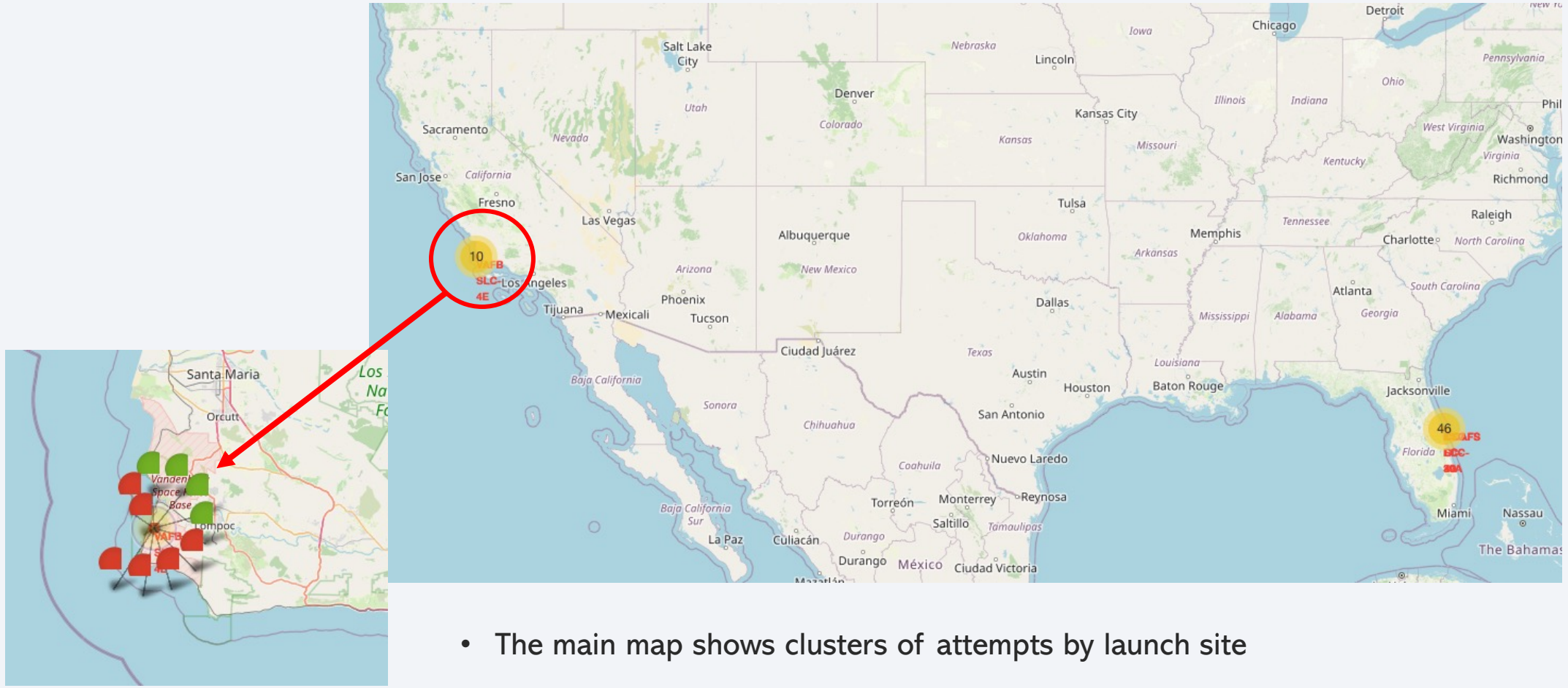

Section 3
Launch Sites
Proximities Analysis

All Launch Sites



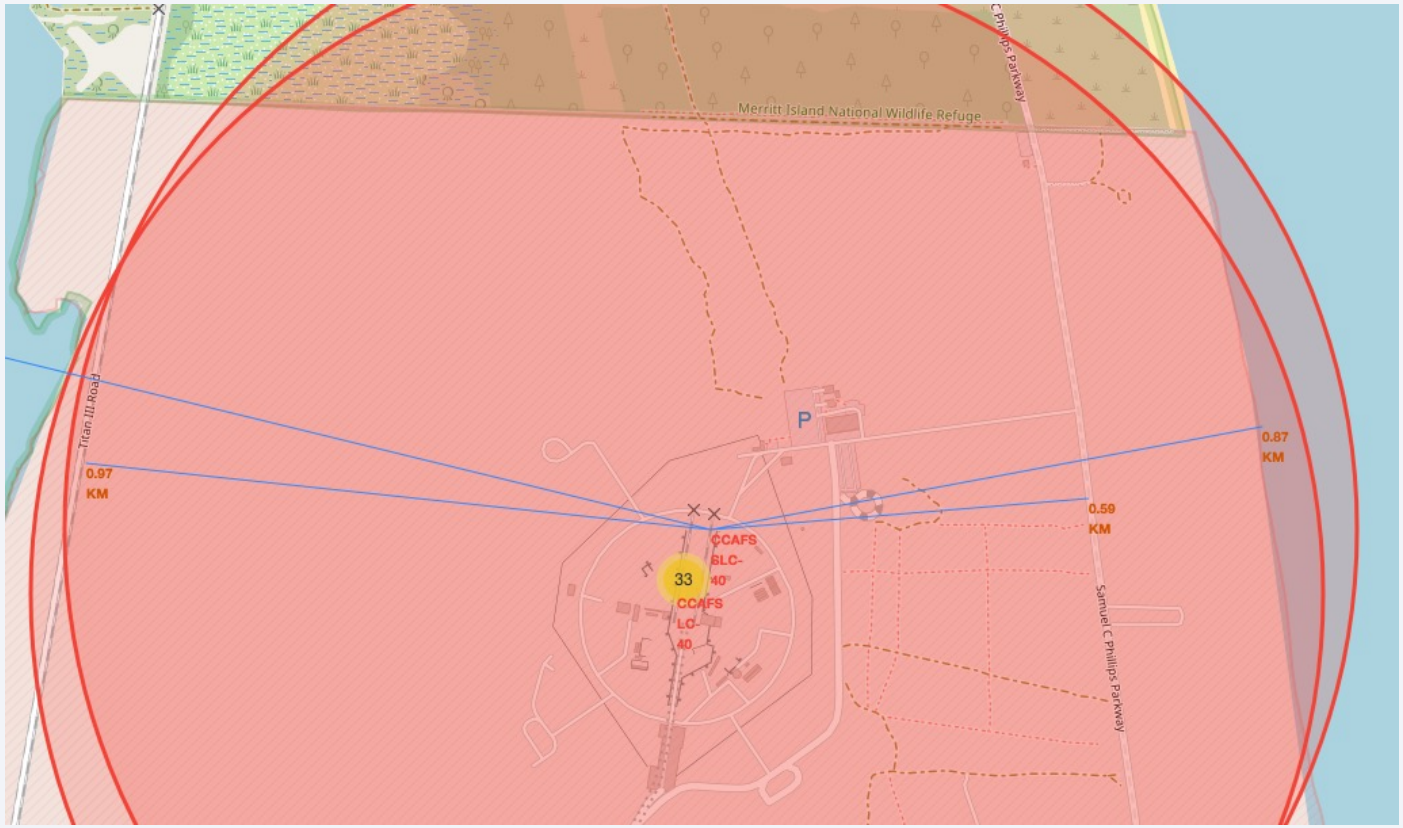
- All launch sites are very close to the coast
- CCAFS SLC-40 and CCAFS LC-40 are closer to the equator

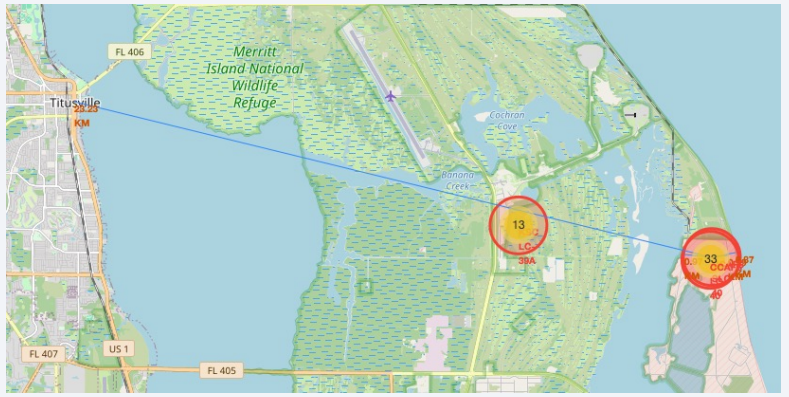
Successful and Unsuccessful attempts by Launch Site



- The main map shows clusters of attempts by launch site
- Upon closer inspection, green and red markers were added to indicate successful or unsuccessful attempts.

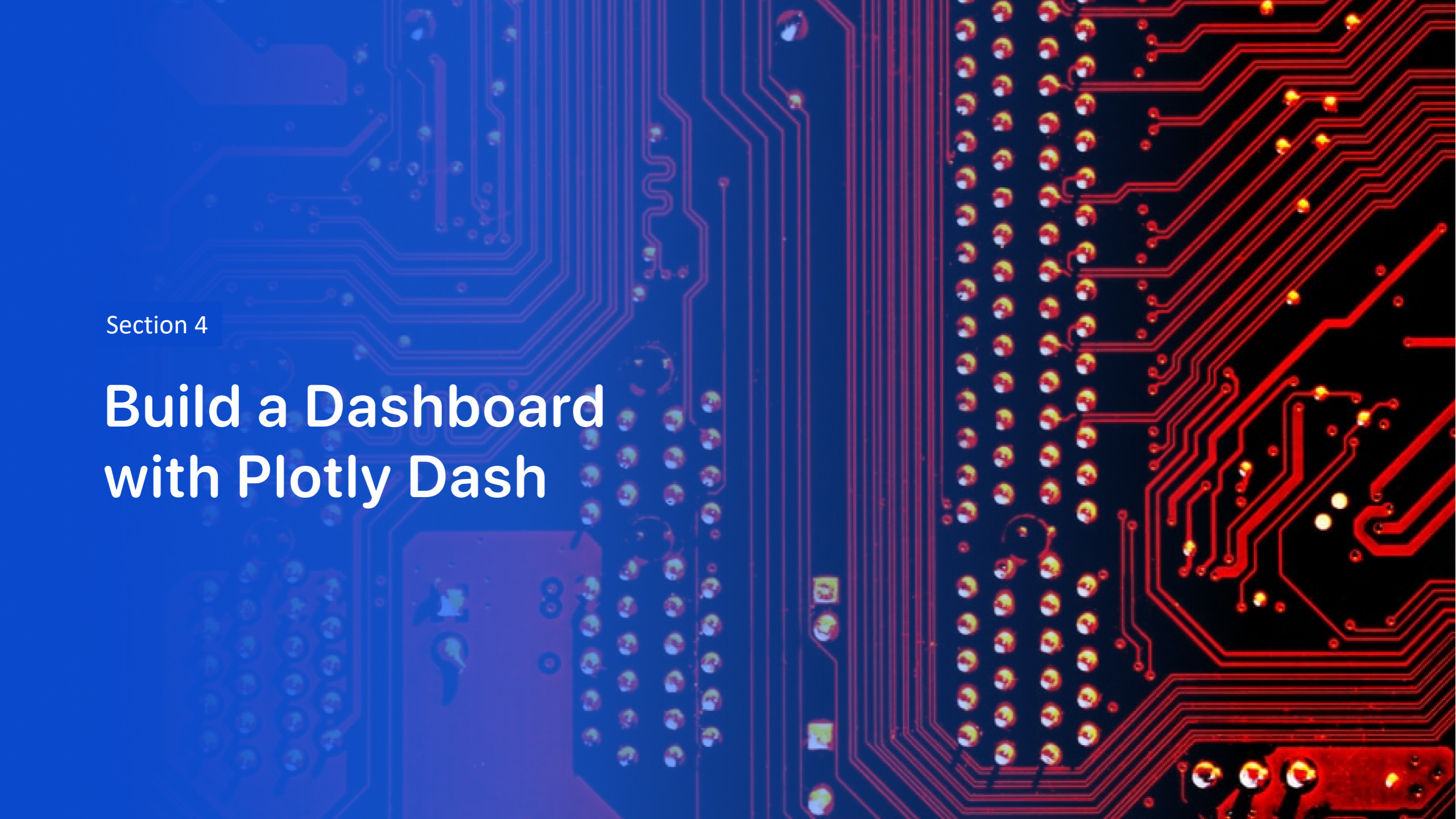
Launch Sites vs proximities



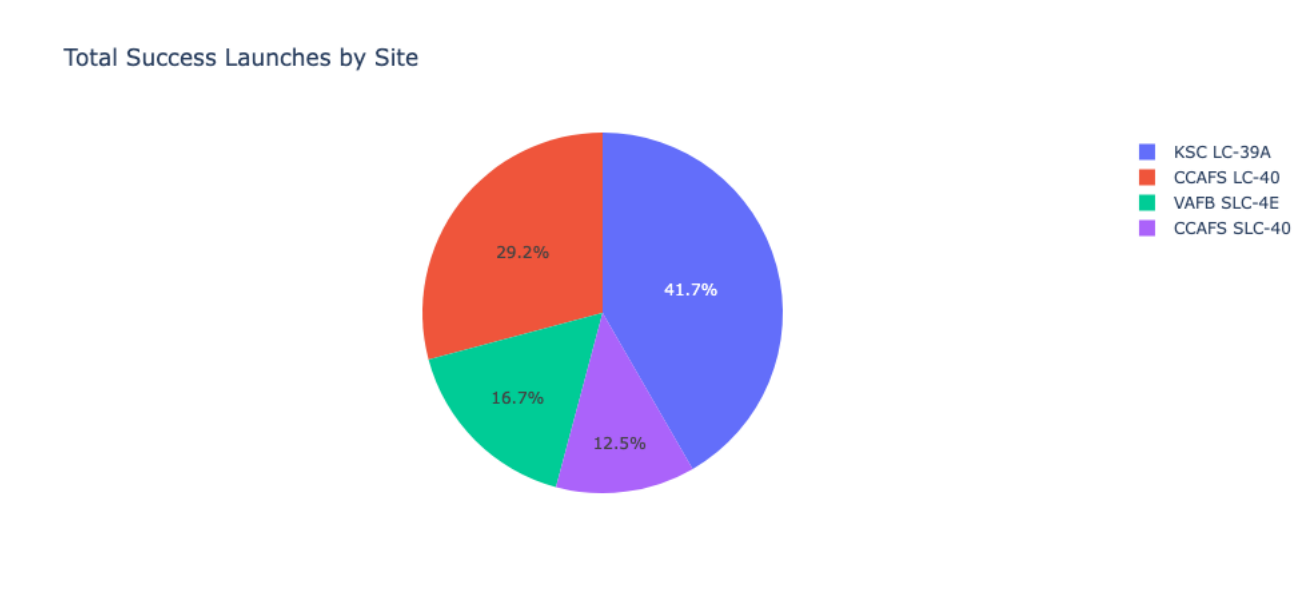


- All sites are located within close distance to the coastline, roads, railroads for logistics. Main cities are also relatively close, but not as close (risks, noise, etc.).

- For CCAFS SLC-40:
  - 870 meters from the coastline, 590 meters from road, 970 meters from railroad
  - 23.2 Km away from a main city.

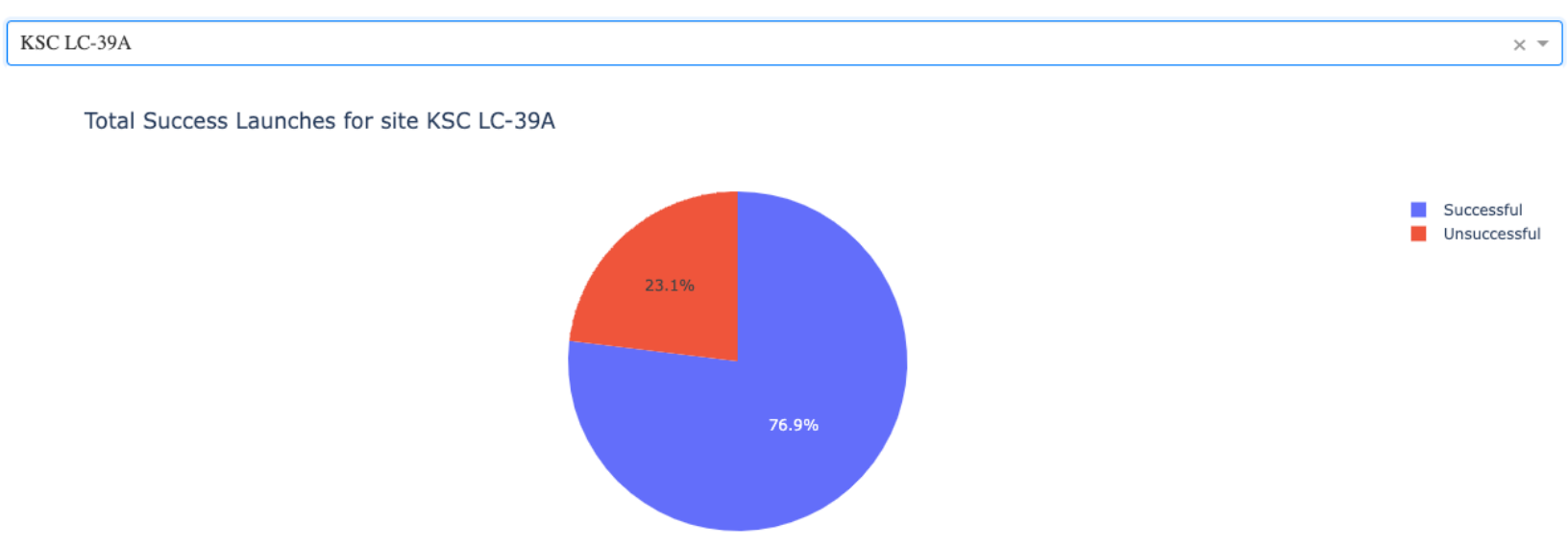

Section 4
Build a Dashboard with Plotly Dash

Success Count for all Launch Sites



- The launch site with the highest successful launches by quite a margin is KSC LC-39A, followed by CCAFS LC-40.

Launch Site with highest success ratio



- KSC LC-39A has the highest ratio, with a 76.9% (Successful) to 23.1% (Unsuccessful).
- CCAFS LC-40 presents the second highest ratio, with 73.1% to 26.9%.

Payload Mass vs Class (outcome)

- The 2000 kg to 5500 kg range concentrates the majority of successful outcomes, although not necessarily with the best ratio.
- After the 5500 kg mark almost all attempts are unsuccessful, except for one with 9600 kg using B4 Booster.
- Overall, FT and B5 Boosters seem to be the most successful ones (although B5 only has 1 attempt), while the rest have generally low success ratio.

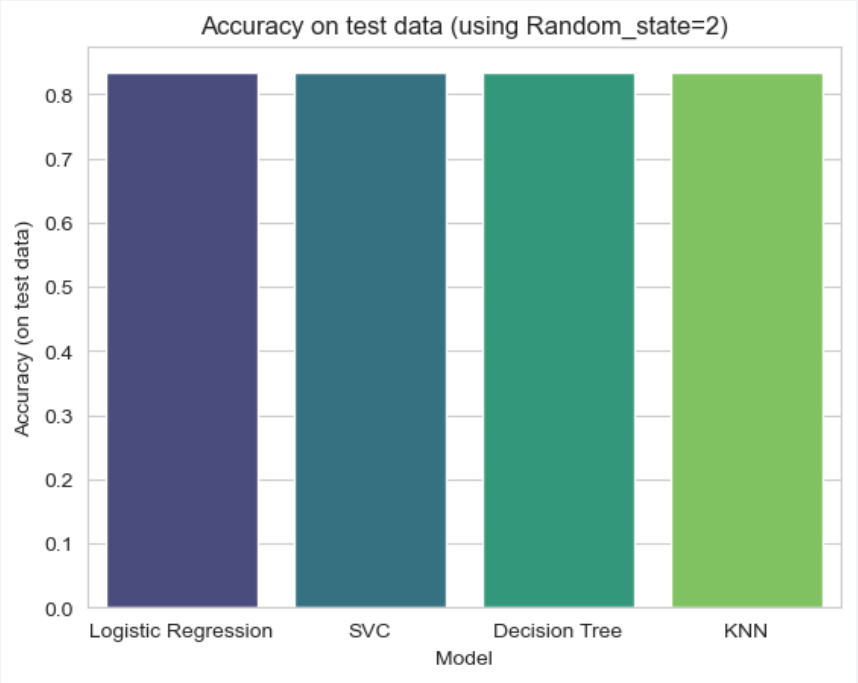
Section 5

Predictive Analysis (Classification)

Classification Accuracy: Random_state=2

- The dataset is really small (90 observations), and as such, too sensitive to the train-test split. Using que original proposed Random_state=2, all models performed exactly the same (boring results and not representative of real situations).

	Accuracy (on test data)
Logistic Regression	0.833333
SVC	0.833333
Decision Tree	0.833333
KNN	0.833333

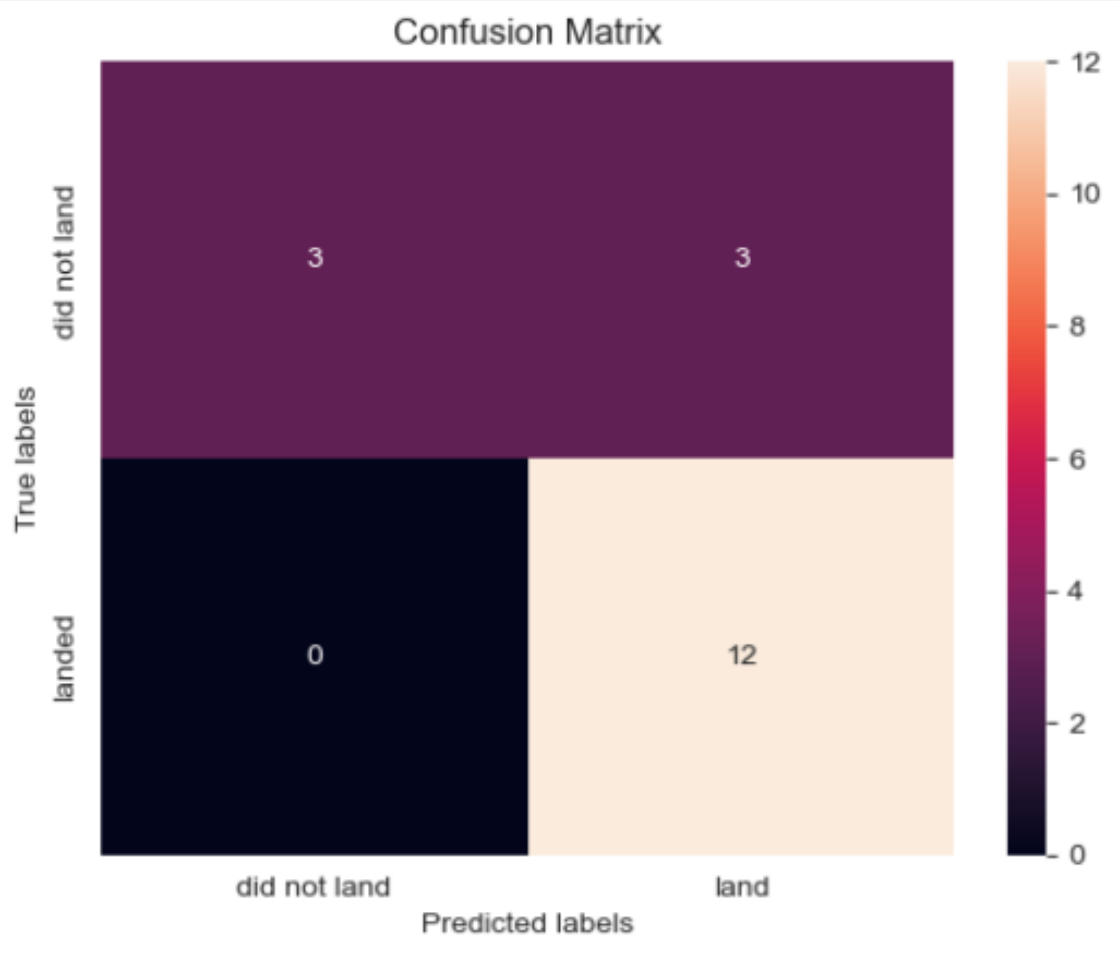


Classification Accuracy: Random_state=3

- Using a random_state=3 (shuffling the observations for train/test), K-Nearest Neighbours model was found to be the most accurate, with 94.4%

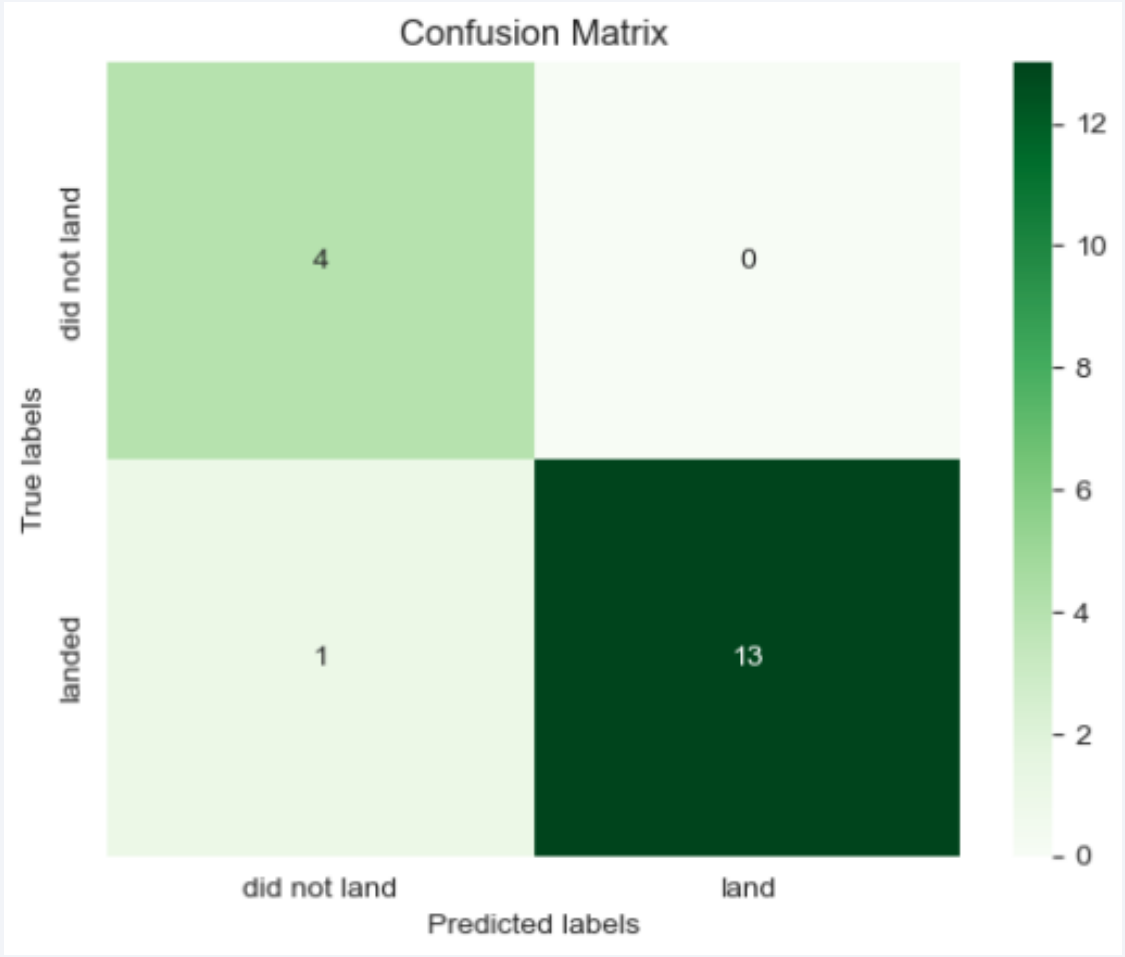
	Accuracy_on_Test_Data
Logistic Regression	0.888889
SVM	0.888889
Decision Tree	0.888889
KNN	0.944444

Confusion Matrix: Random_state=2



- Using random_state=2 all models performed exactly the same, with 3 true negatives, 12 true positives, 3 false positives and 0 false negatives.
- It is not a bad result, but for the purpose of this project, having 3 false positives could be dangerous (predicting a successful landing, but actually failing).

Confusion Matrix: Random_state=3



- Using random_state=3, KNN was the best performing model, with 4 true negatives, 13 true positives, 0 false positives and 1 false negatives.
- The main improvement here is the better prediction of true positives and true negatives, as being able to correctly predict whether the attempt will be successful or not is the most important thing.
- Even with the 1 false negative, it could be argued that for this case, it is better to predict a failure and get a success, rather than predict a success and get a failure.

Conclusions

- The main objective of the project was to predict the success of the recovery process of Falcon 9 booster.
- For this purpose, data was collected, explored, worked on and finally used to train a predictive model.
- After testing different classification models, KNN was found to be capable of predicting launch outcome with a 94.4% accuracy.
- Although the results obtained from the model are good, the small available data used for training and testing is not enough to be confident about the consistency of the model.

Appendix

- Link for all of the notebooks and presentation:
- https://github.com/GRiosG/Public/tree/main/IBM_DataScience_Certificate_Capstone_Project

Thank you!