




IBM Developer  
SKILLS NETWORK

# Winning Space Race with Data Science

Olena Tsyganok  
04/01/2025



# Outline

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

# Executive Summary

- **Summary of methodologies**
  - Data Collection API
    - Data Collection API with Webscraping
    - Data Wrangling
    - Exploratory Analysis Using SQL
    - Data Visualization
    - Interactive Visual Analytics with Folium
    - Build an Interactive Dashboard with Plotly Dash
    - Interactive Visual Analytics and Dashboard
- **Summary of all results**
  - Exploratory Data Analysis
  - Interactive Analytics with screenshots
  - Predictive Analytic results

# Introduction

- Project background and context
  - Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch.
- Problems you want to find answers
  - What factors determine if the rocket will land successfully
  - The interaction among various features that determine the success rate of successful landing
  - What operating conditions need to be fulfilled to ensure a successful landing program



Section 1

# Methodology

# Methodology

---

- Executive Summary
- Data collection methodology:
  - SpaceX Rest API
  - Web scrapping
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

# Data Collection

---

- Describe how data sets were collected.

Requesting rocket launch data from SpaceX API with the URL

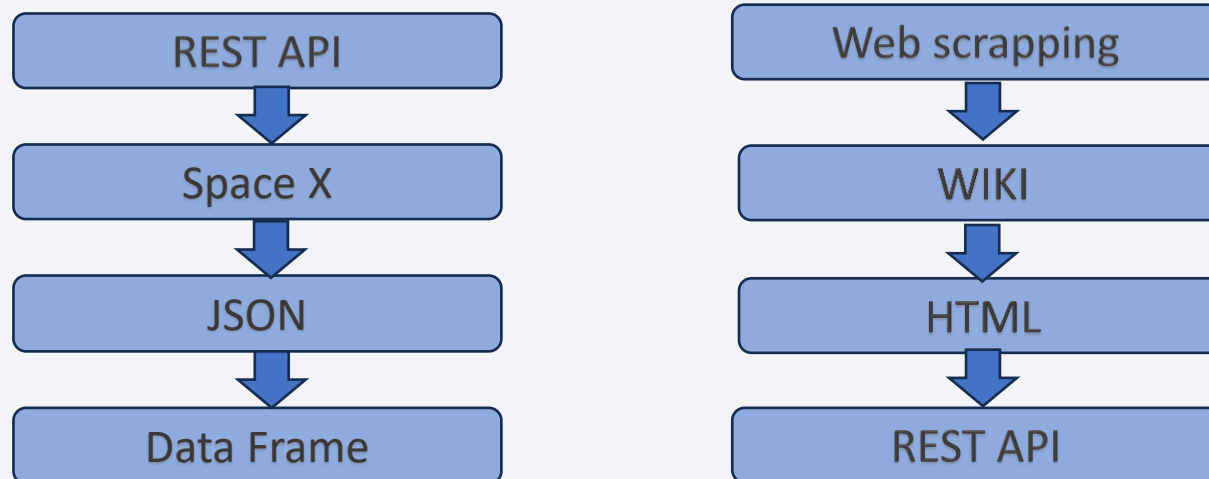
Request and parse the SpaceX launch data using the GET request

Decode the response content as a Json using .json()

Data Wrangling

Dealing with Missing Values

- You need to present your data collection process use key phrases and flowcharts



# Data Collection – SpaceX API

---

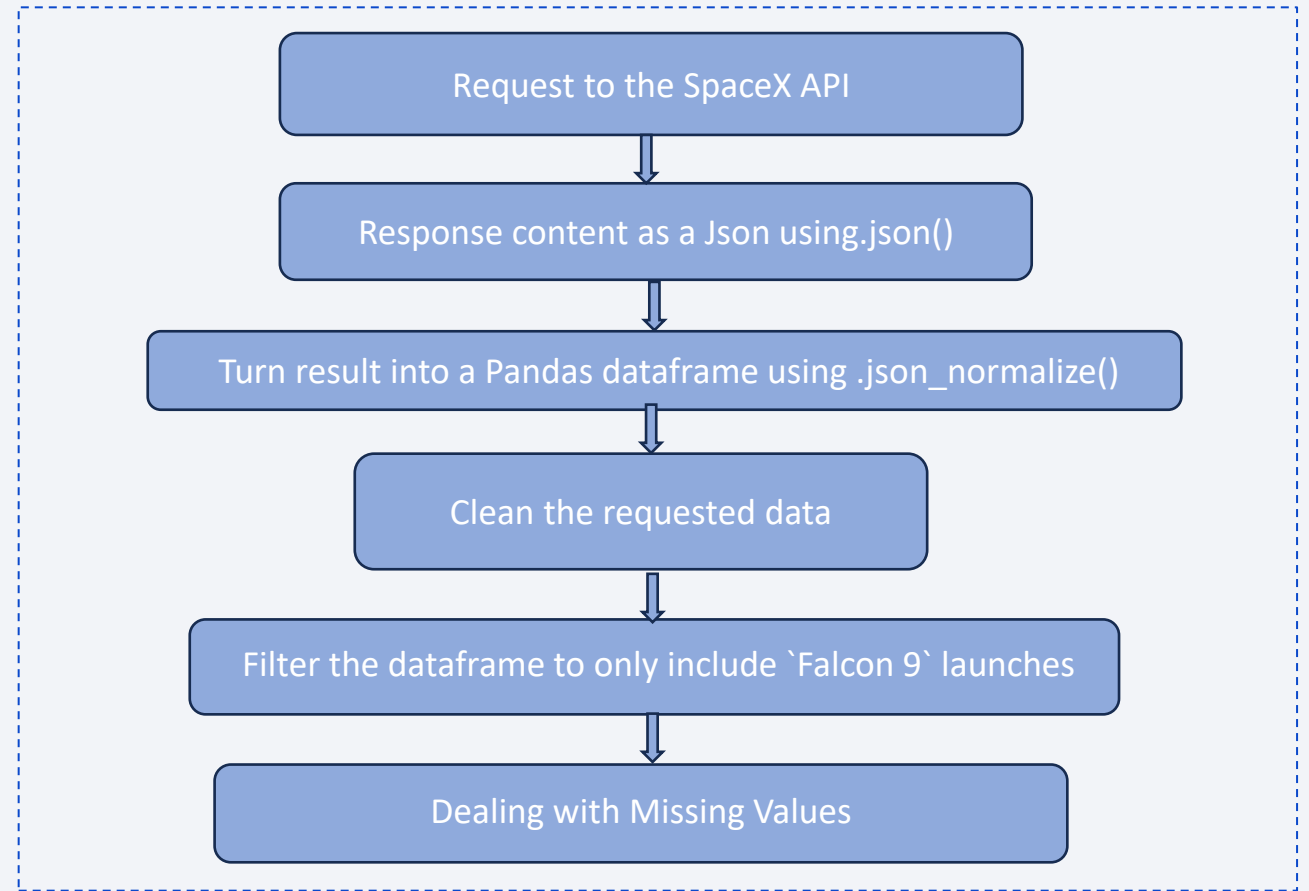
Request to the SpaceX API

Response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

Do some data wrangling

Clean the requested data

[https://github.com/Elygan/DS\\_LAB/blob/main/jupyter-labs-spacex-data-collection-api-v2\\_LAB2.ipynb](https://github.com/Elygan/DS_LAB/blob/main/jupyter-labs-spacex-data-collection-api-v2_LAB2.ipynb)





# Data Collection - Scraping

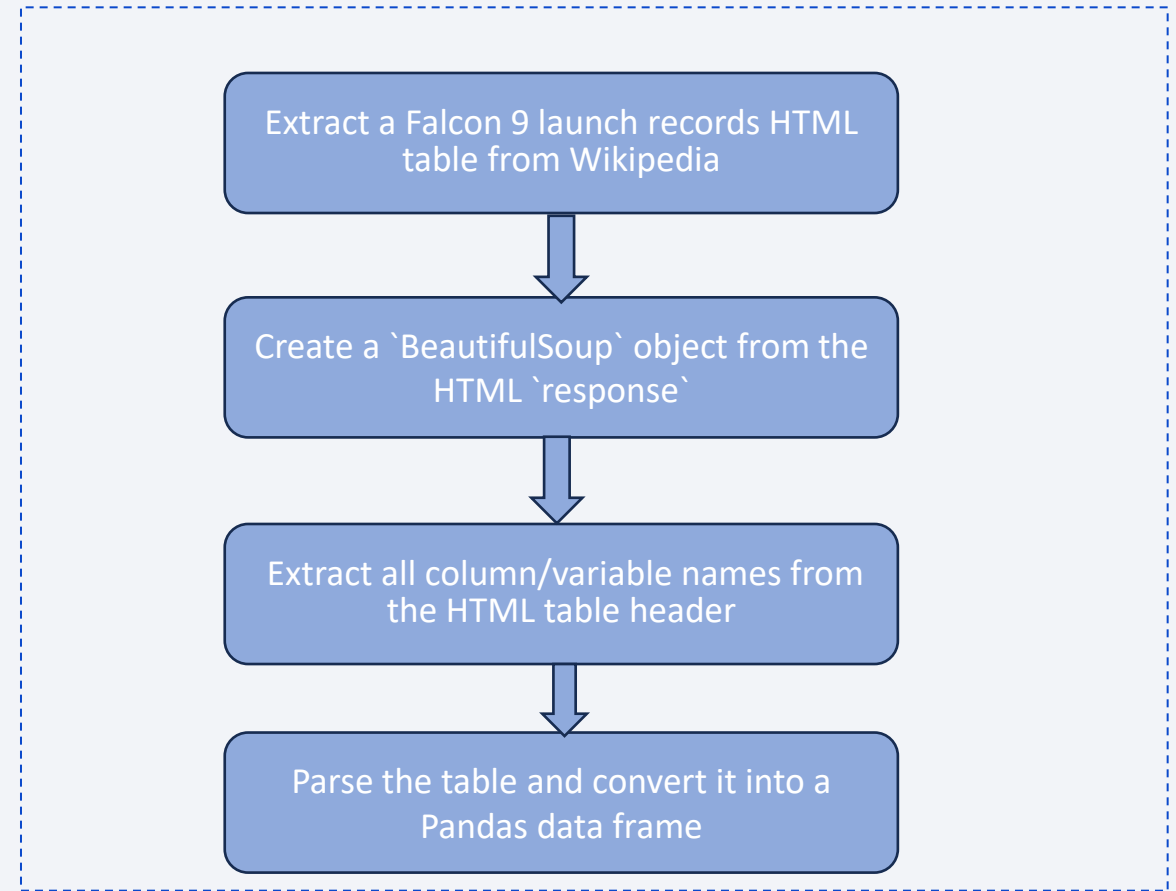
---

Extract a Falcon 9 launch records HTML table from Wikipedia

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

Parse the table and convert it into a Pandas data frame

[https://github.com/Elygan/DS\\_LAB/blob/main/jupyter-labs-webscrapingLAB.ipynb](https://github.com/Elygan/DS_LAB/blob/main/jupyter-labs-webscrapingLAB.ipynb)



# Data Wrangling

---

Calculate the number of launches on each site and occurrence of each orbit use the method `.value_counts()`

Calculate the number and occurrence of mission outcome of the orbits

Create a landing outcome label from Outcome column

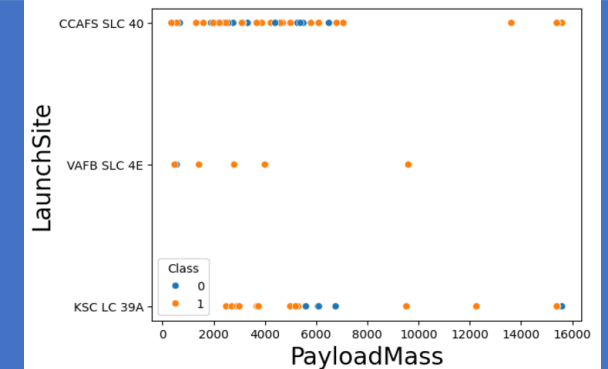
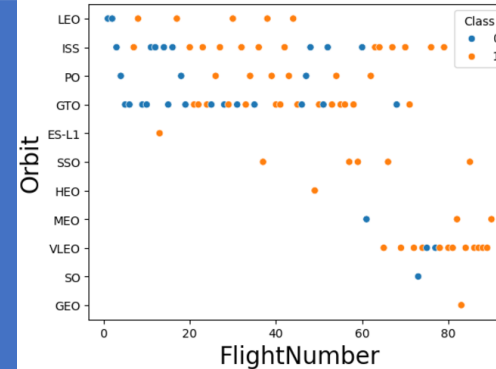
[https://github.com/Elygan/DS\\_LAB/blob/main/labs-jupyter-spacex-Data%20wrangling-v2Lab2.ipynb](https://github.com/Elygan/DS_LAB/blob/main/labs-jupyter-spacex-Data%20wrangling-v2Lab2.ipynb)

# EDA with Data Visualization

## Scatterplot

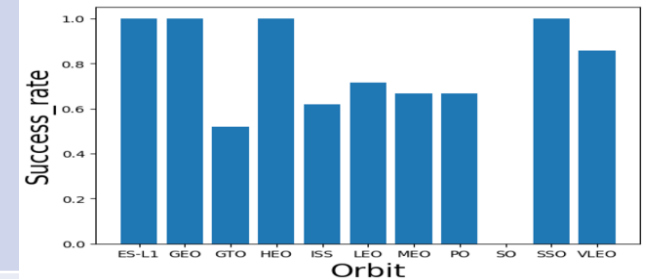
Visualize the relationship between:

- Flight Number and Launch Site
- Payload and Launch Site
- FlightNumber and Orbit type
- Payload and Orbit type



## Bar chart

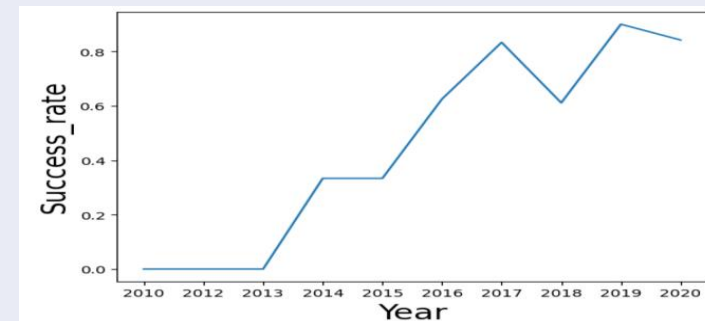
Visualize the relationship between success rate of each orbit type



## Line chart

To get the average launch success trend.

[https://github.com/Elygan/DS\\_LAB/blob/main/jupyter-labs-eda-dataviz-v2LAB.ipynb](https://github.com/Elygan/DS_LAB/blob/main/jupyter-labs-eda-dataviz-v2LAB.ipynb)



# EDA with SQL

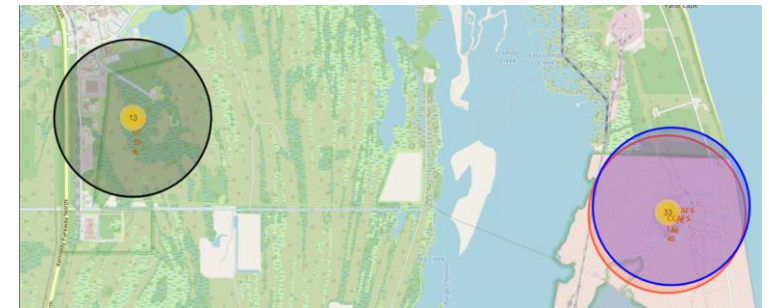
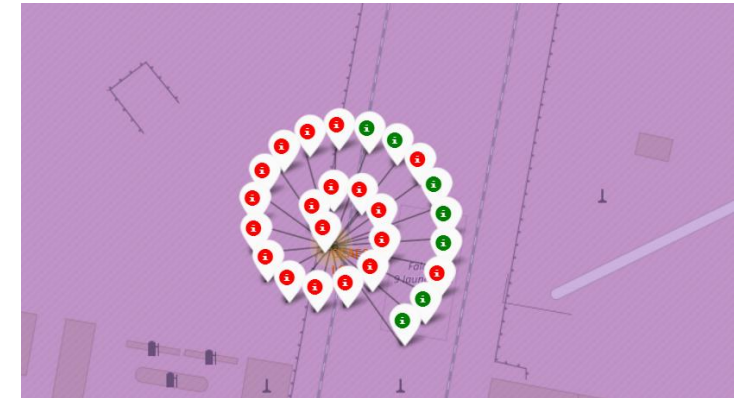
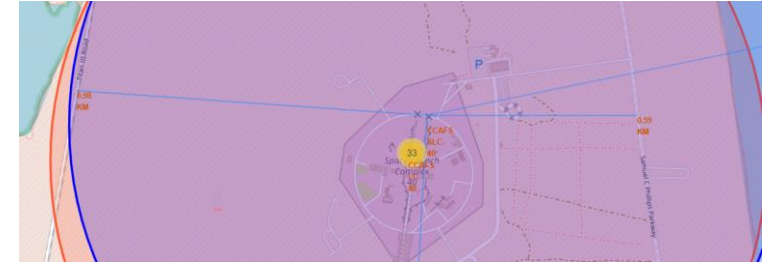
---

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first succesful landing outcome in ground pad was acheived.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster\_versions which have carried the maximum payload mass. Use a subquery
- 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.
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

[https://github.com/Elygan/DS\\_LAB/blob/main/jupyter-labs-eda-sql-coursera\\_sqliteLAB2.ipynb](https://github.com/Elygan/DS_LAB/blob/main/jupyter-labs-eda-sql-coursera_sqliteLAB2.ipynb)

# Build an Interactive Map with Folium

- We Marked all launch sites on a map
  - Add a highlighted circle area with a text label on a specific coordinate
  - Mark the success/failed launches for each site on the map
  - Create markers for all launch records
  - Calculate the distances between a launch site to its proximities
  - Draw a `PolyLine` between a launch site to the selected coastline point
- 
- [https://github.com/Elygan/DS\\_LAB/blob/main/lab-jupyter-launch-site-location-v2LAB.ipynb](https://github.com/Elygan/DS_LAB/blob/main/lab-jupyter-launch-site-location-v2LAB.ipynb)





# Build a Dashboard with Plotly Dash

---

- We Build an Interactive Dashboard with Plotly Dash
- Add a Launch Site Drop-down Input Component
- Add a pie chart to show the total successful launches count for all sites
- Add a slider to select payload range
- Add a callback function

The general idea of this callback function is to get the selected launch site from site-dropdown and render a pie chart visualizing launch success counts. From a dashboard point of view, we want to be able to easily select different payload range and see if we can identify some visual patterns.

[https://github.com/Elygan/DS\\_LAB/blob/main/UIInteractive\\_Dashboard\\_with\\_Ploty\\_Dash.py](https://github.com/Elygan/DS_LAB/blob/main/UIInteractive_Dashboard_with_Ploty_Dash.py)

# Predictive Analysis (Classification)

---

- Load data frame
- Transform data
- Split data on training and testing set
- Train different classification models
- Optimize the Hyperparameter grid search using GridSearchCV object
- Evaluate the model
- [https://github.com/Elygan/DS\\_LAB/blob/main/SpaceX-Machine-Learning-Prediction-Part-5-v1LAB2.ipynb](https://github.com/Elygan/DS_LAB/blob/main/SpaceX-Machine-Learning-Prediction-Part-5-v1LAB2.ipynb)

# Results

---

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



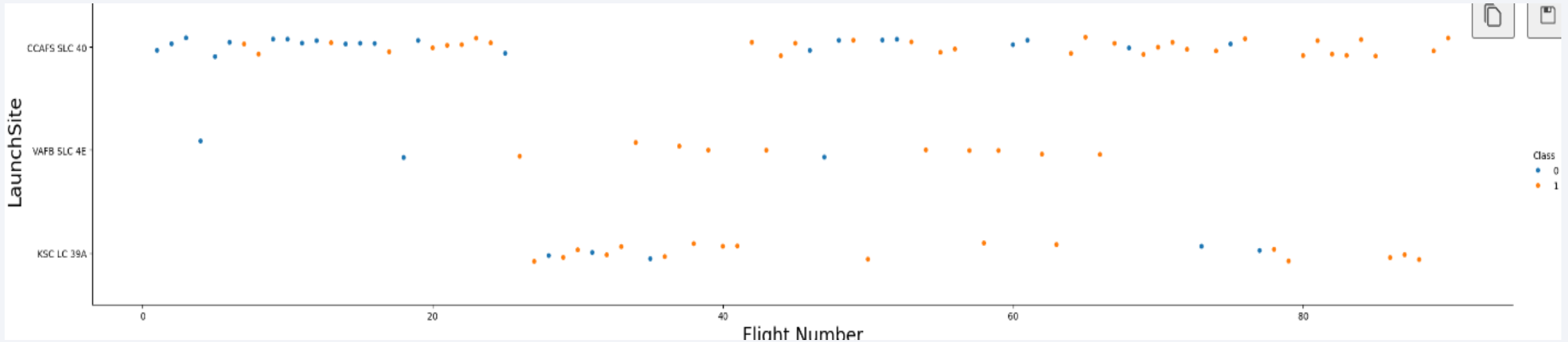
The background of the slide is an abstract composition. It features a dark blue base color. Overlaid on this are numerous diagonal streaks in shades of red and cyan. A faint, light blue grid pattern is also visible, particularly in the lower-left quadrant. The overall effect is dynamic and technological.

Section 2

# Insights drawn from EDA



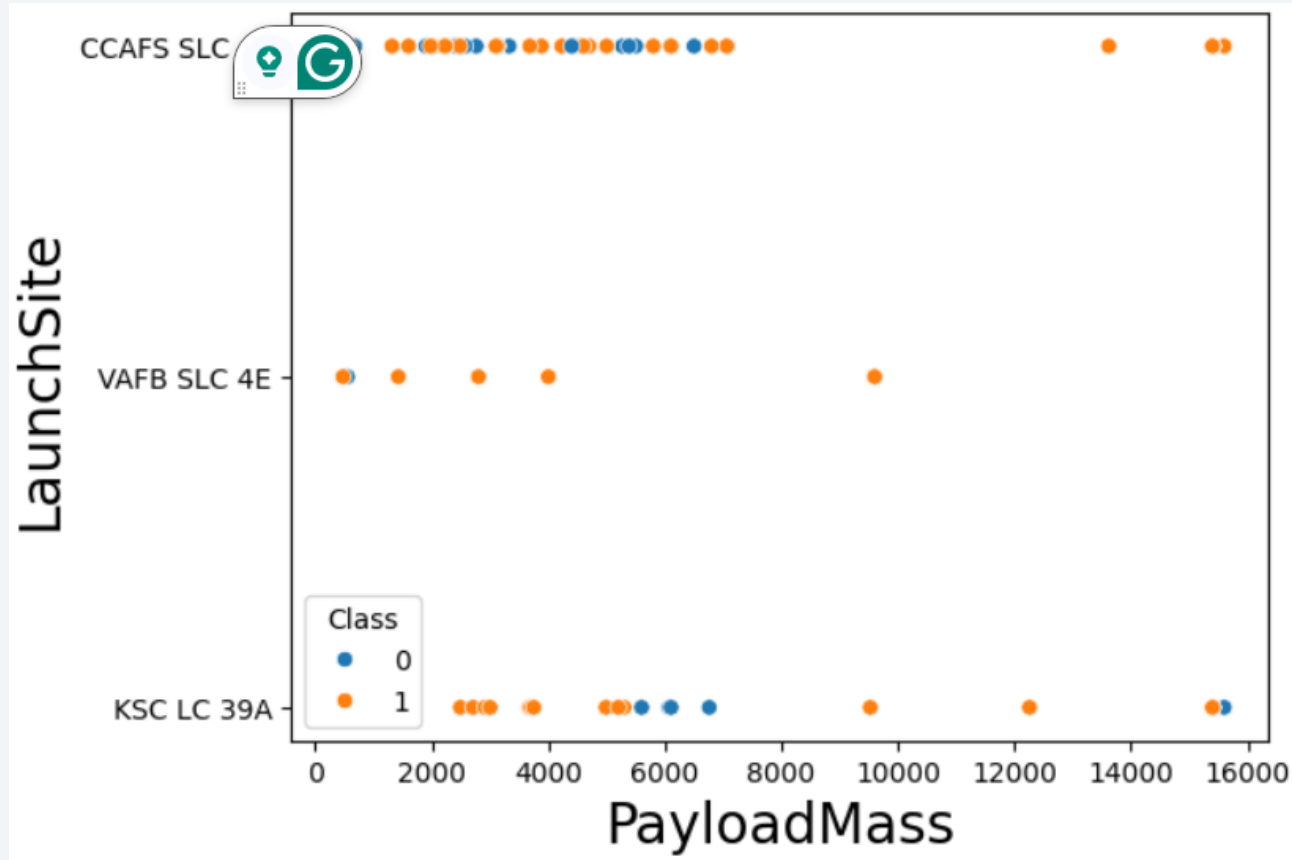
# Flight Number vs. Launch Site



The plot shows that the greater the number of flights, the more successful landing on all launch sites

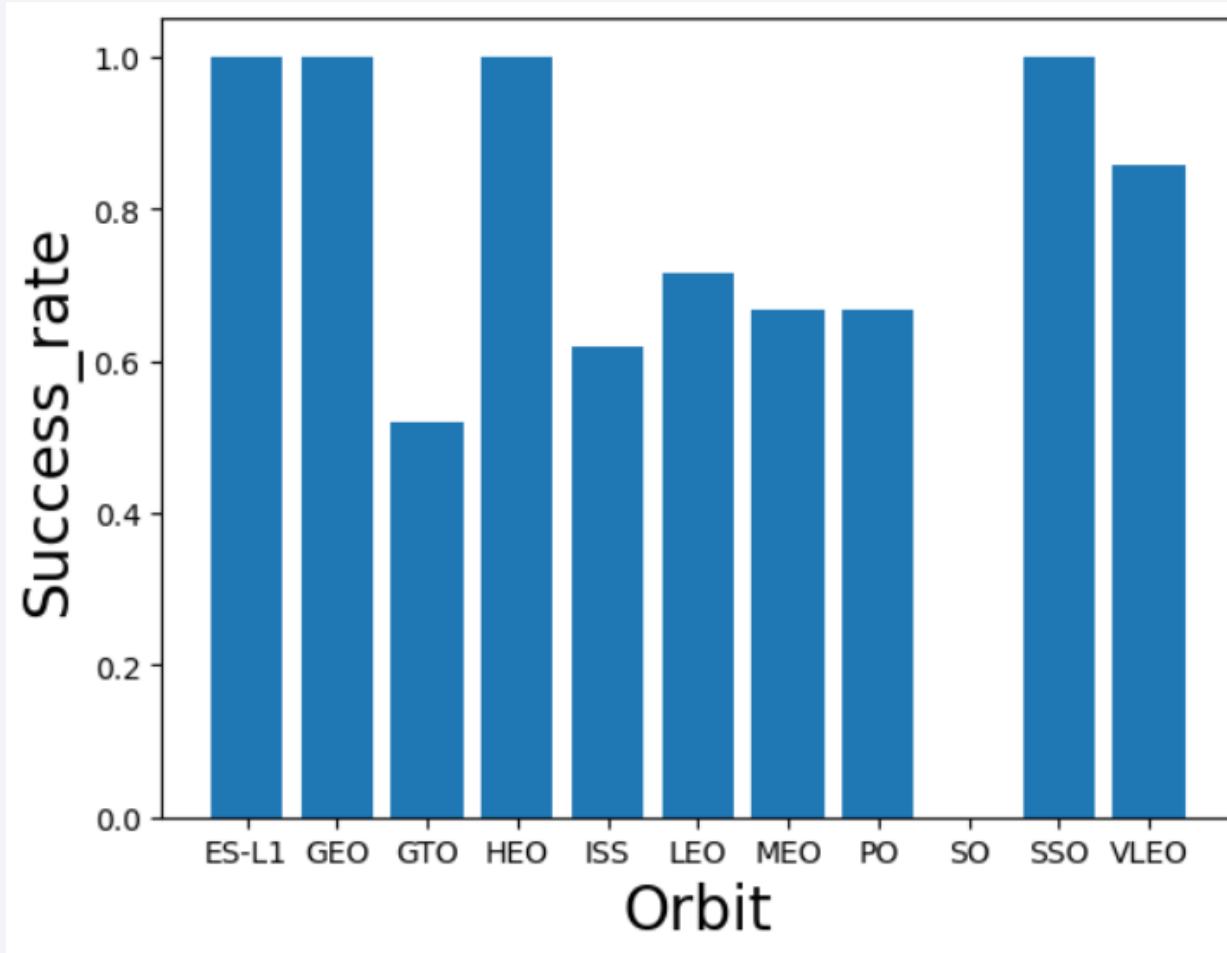


# Payload vs. Launch Site



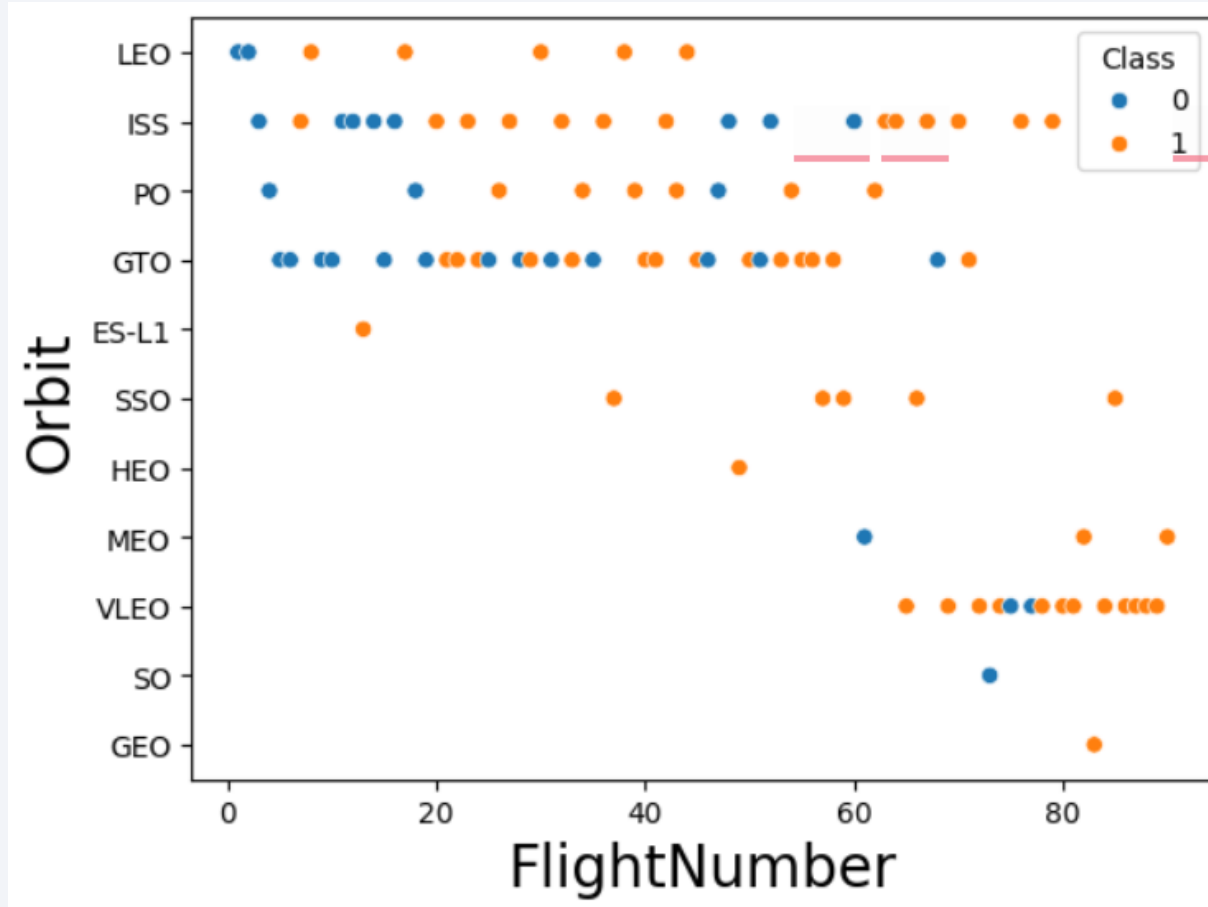
The most common payload mass up to 5800, and it has more successful landing on all launch sites

# Success Rate vs. Orbit Type



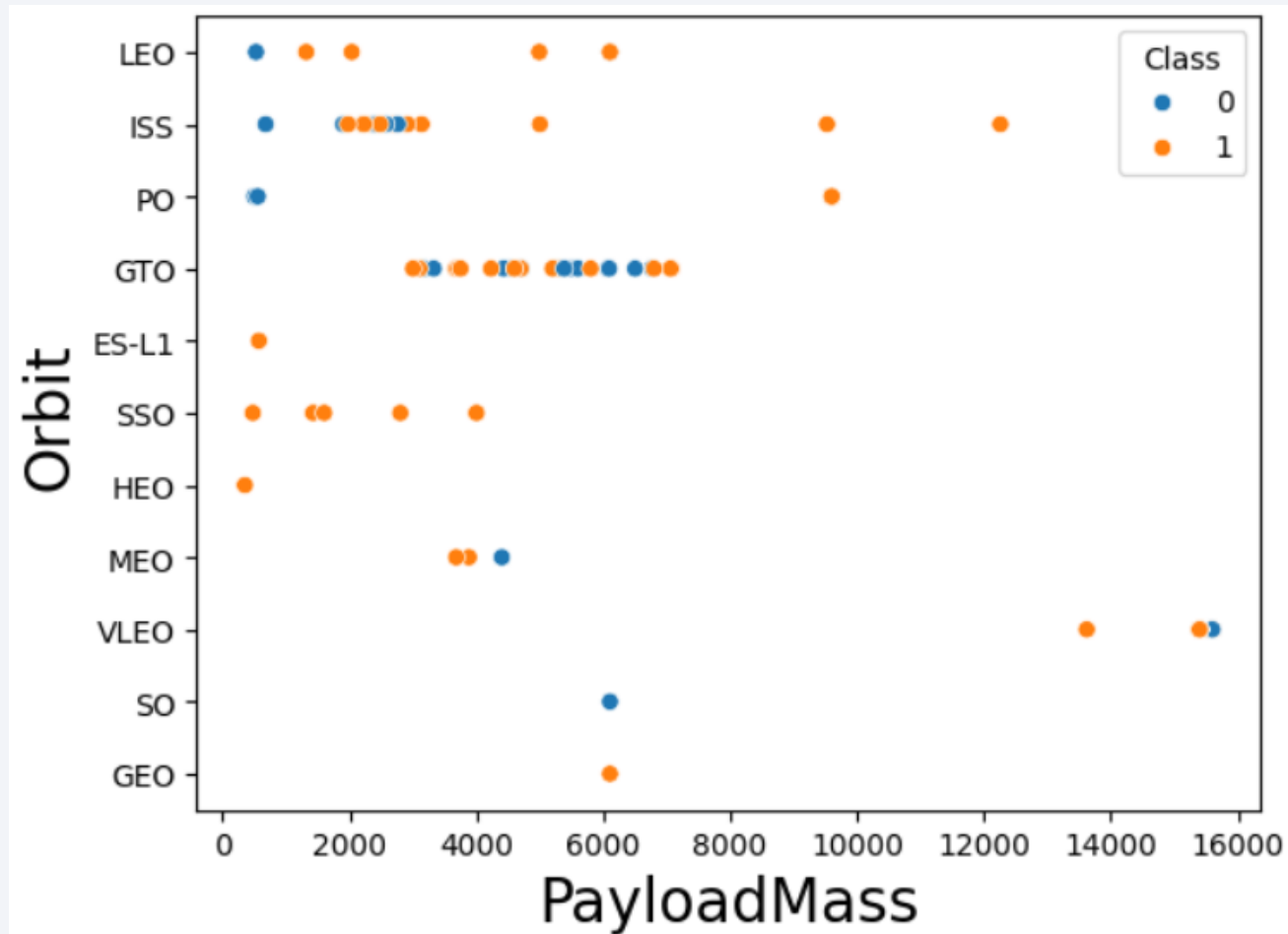
The highest success rate has orbit types: ES-L1, GEO, HEO, SSO and the lowest: GTO, SO

# Flight Number vs. Orbit Type



LEO orbit success appears related to the number of flights. But there is no relationship between flight number when in GTO orbit.

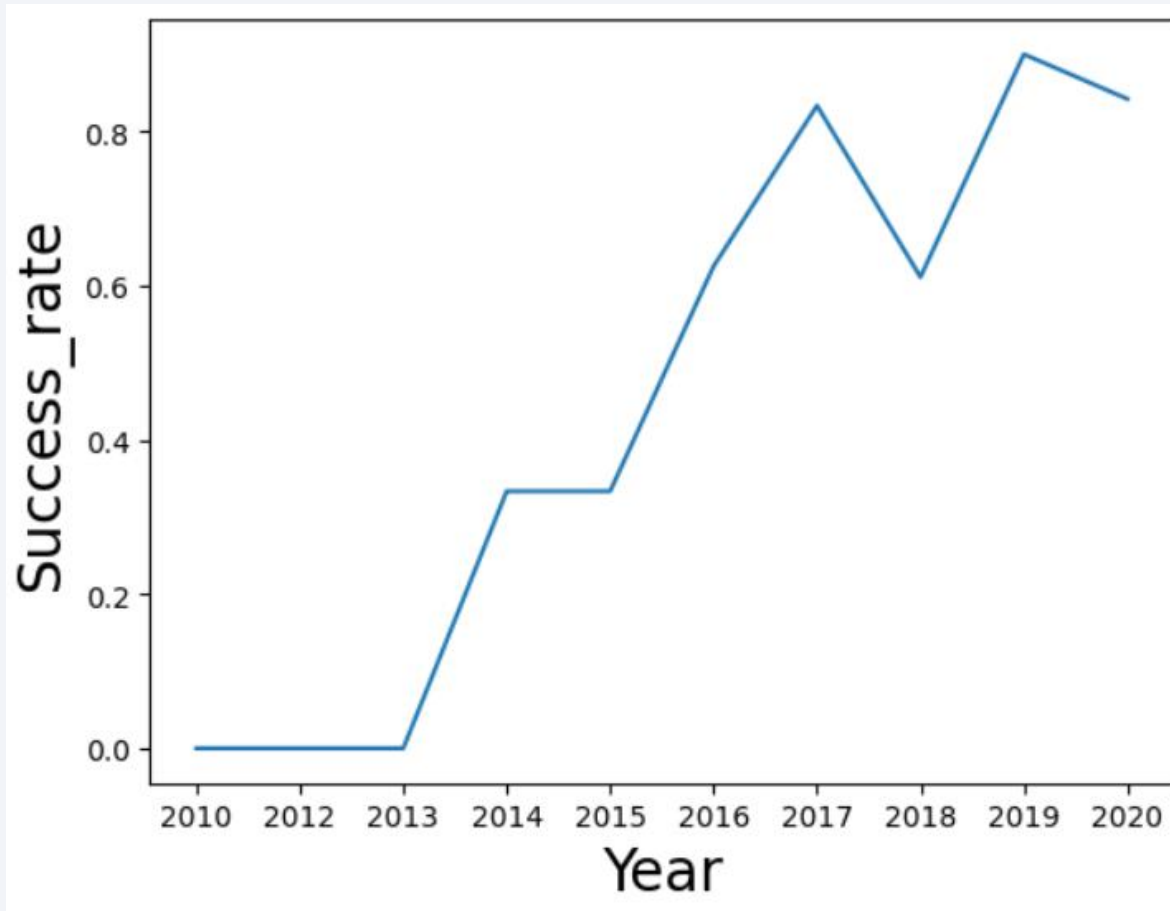
# Payload vs. Orbit Type



With heavy payloads the successful landing or positive landing rate are more for Po, LEO and ISS orbits

# Launch Success Yearly Trend

---



There is observe a rapid increase success rate from 2013 till 2017, stable in 2014 and decrease in 2017-2018



# All Launch Site Names

---

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

```
%sql SELECT DISTINCT Launch_Site FROM SPACEXTABLE;
```

We selected unique Launch sites from SpaceXtable csv file

# Launch Site Names Begin with 'CCA'

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 (parachut
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 (parachut
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attem
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attem
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attem

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

We select all records(5 first rows) from SpaceXtable dataset and use 'CCA%' to display Launch site wich begin with 'CCA'

# Total Payload Mass

---

```
total payload mass
```

```
107010
```

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS 'total payload mass' FROM SPACEXTABLE WHERE Customer LIKE '%NASA%'
```

We used sum() function to calculate total payload mass. And used %NASA% to define all words which contain NASA in column “Customer”

# Average Payload Mass by F9 v1.1

---

Mean payload mass
2928.4

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS 'Mean payload mass' FROM SPACEXTABLE WHERE Booster_Version ='F9 v1.1'
```

We used avg() function to calculate mean payload mass.

# First Successful Ground Landing Date

---

**min(DATE)**

2015-12-22

```
%sql SELECT min(DATE) FROM SPACEXTABLE WHERE Landing_Outcome LIKE '%Success%'
```

We use min(DATE) function to determine the oldest date where landing outcome column contain word 'Success'



## Successful Drone Ship Landing with Payload between 4000 and 6000

---

```
] : Booster_Version  Landing_Outcome
-----
      F9 FT B1022  Success (drone ship)
      F9 FT B1026  Success (drone ship)
      F9 FT B1021.2  Success (drone ship)
      F9 FT B1031.2  Success (drone ship)
```

```
%sql SELECT Booster_Version, Landing_Outcome FROM SPACEXTABLE WHERE Landing_Outcome LIKE '%Success%drone ship%'
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000
```

We select data in column landing\_outcome which contain words 'Success and drone ship' and payload mass must be between 4000 and 6000

# Total Number of Successful and Failure Mission Outcomes

---

Success_Outcome	Failure_Outcome
100	1

```
%sql SELECT SUM(CASE WHEN Mission_Outcome LIKE '%Success%' THEN 1 ELSE 0 END) AS Success_Outcome ,  
SUM(CASE WHEN Mission_Outcome LIKE '%Failure%' THEN 1 ELSE 0 END)  
AS Failure_Outcome FROM SPACEXTABLE
```

We use function `sum()` and `Mission_Outcome` column which contain words 'Success' or 'Failure'

# Boosters Carried Maximum Payload

---

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

```
%sql SELECT Booster_Version, PAYLOAD_MASS_KG_ FROM SPACEXTABLE  
WHERE PAYLOAD_MASS_KG_ = (SELECT MAX(PAYLOAD_MASS_KG_) FROM SPACEXTABLE )
```

We use subquery and max() function to determine maximum payload

# 2015 Launch Records

Month	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

```
%%sql
SELECT
  CASE
    WHEN substr(Date, 6, 2) = '01' THEN 'January'
    WHEN substr(Date, 6, 2) = '02' THEN 'February'
    WHEN substr(Date, 6, 2) = '03' THEN 'March'
    WHEN substr(Date, 6, 2) = '04' THEN 'April'
    WHEN substr(Date, 6, 2) = '05' THEN 'May'
    WHEN substr(Date, 6, 2) = '06' THEN 'June'
    WHEN substr(Date, 6, 2) = '07' THEN 'July'
    WHEN substr(Date, 6, 2) = '08' THEN 'August'
    WHEN substr(Date, 6, 2) = '09' THEN 'September'
    WHEN substr(Date, 6, 2) = '10' THEN 'October'
    WHEN substr(Date, 6, 2) = '11' THEN 'November'
    WHEN substr(Date, 6, 2) = '12' THEN 'December'
  END AS Month,
  Landing_Outcome,
  Booster_Version,
  Launch_Site
FROM SPACEXTABLE
WHERE substr(Date, 0, 5) = '2015'
  AND Landing_Outcome LIKE '%Failure%drone%ship%';
```

We use `substr(Date, 6,2)` as month to get the months and `substr(Date,0,5)='2015'` for year.

And in column `Landing_Outcome` select records which contain 'Failure, drone, ship'

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

]:

Landing_Outcome	RANK
-----------------	------

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
------------------------	---

```
%%sql
SELECT Landing_Outcome, COUNT(Landing_Outcome) AS RANK FROM SPACEXTABLE WHERE DATE between '2010-06-04' and '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY RANK DESC
```

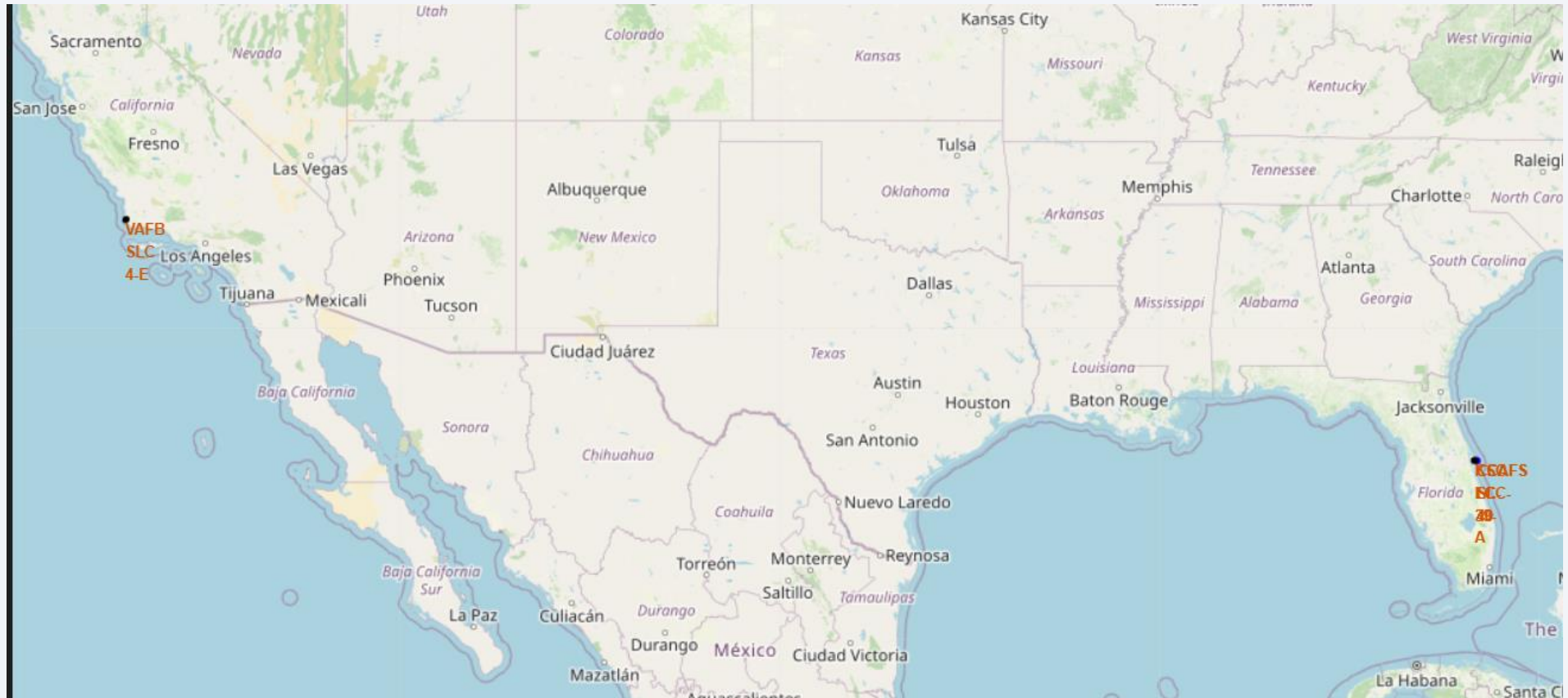
We use function count() to find quantity of landing outcomes and sort in decsending order

A satellite view of Earth from space, showing the curvature of the planet and city lights at night. The background is a deep blue gradient.

Section 3

# Launch Sites Proximities Analysis

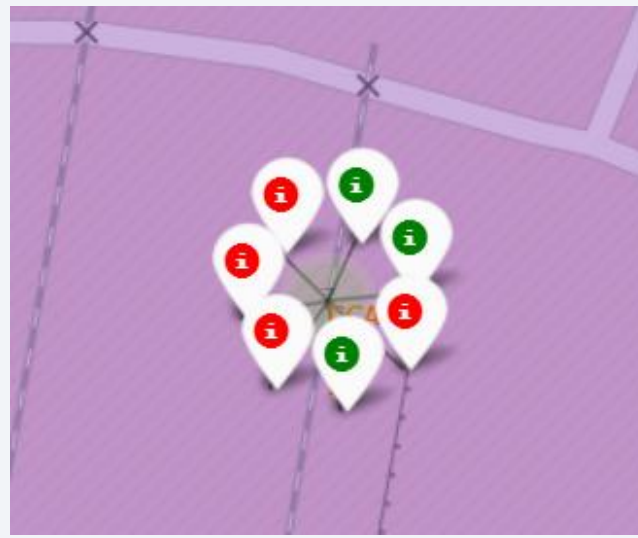
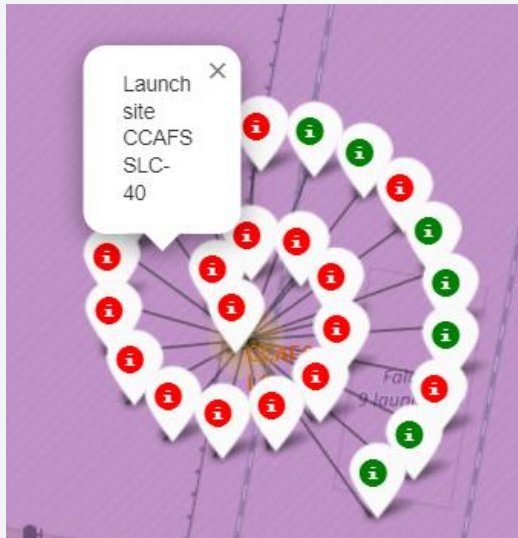
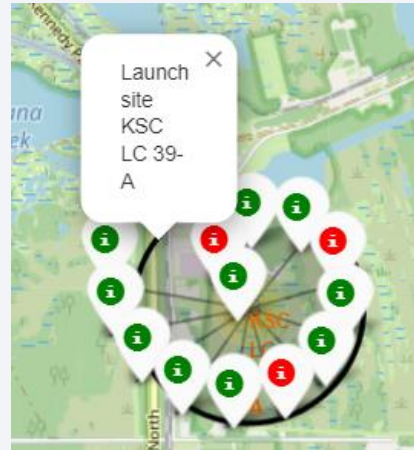
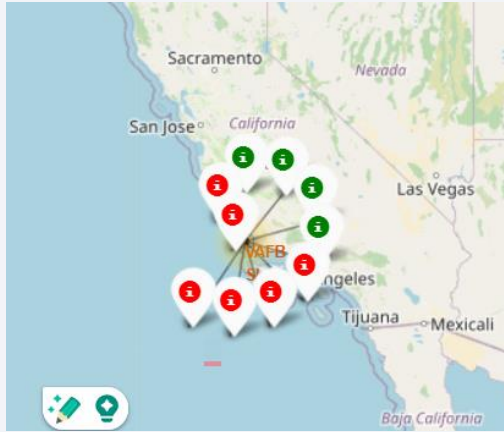
# Launch sites



All launch sites in very close to the coast/ Three of them on Cape Canaveral and one near Los Angeles



## Success/failed launches for each launch site

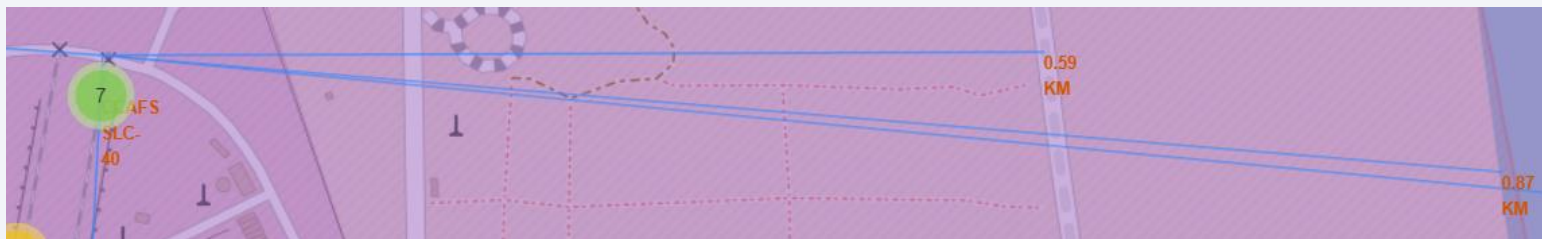
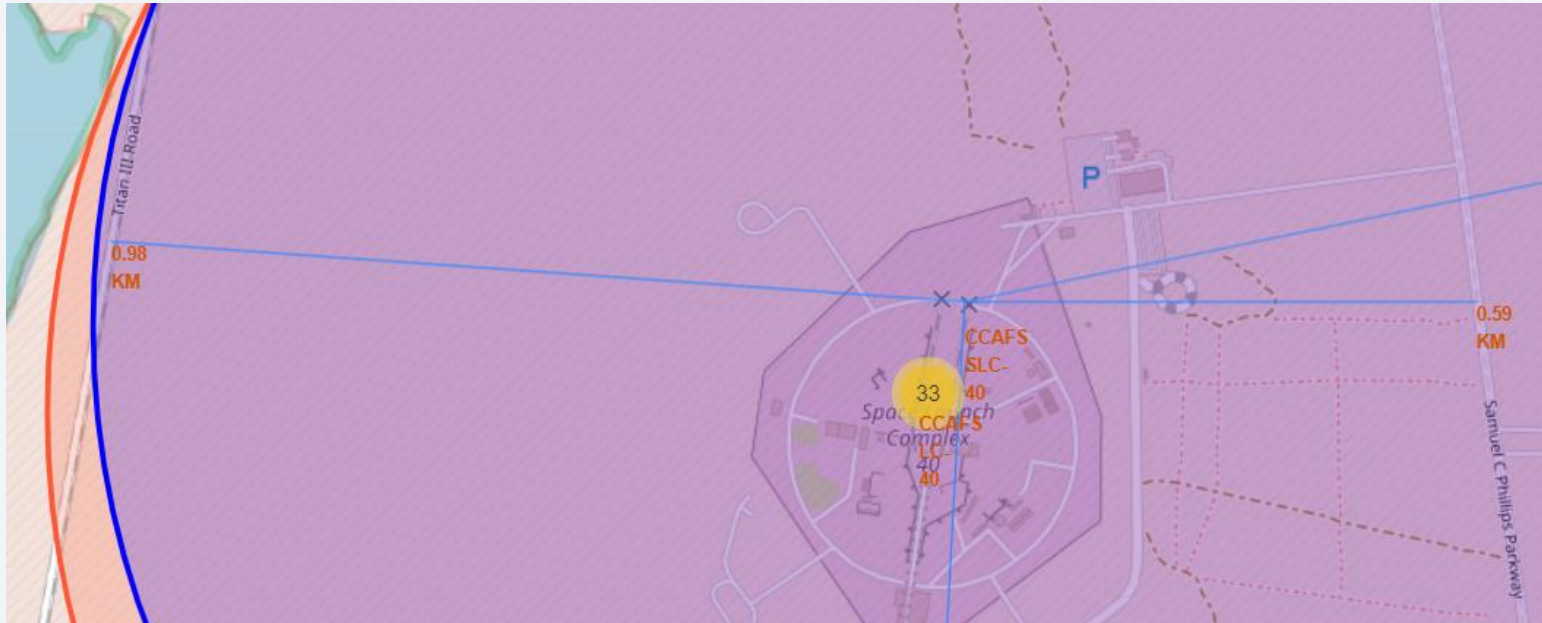


Green marker color represent success launches, red one failed launches

KSC LC 39-A launch site has the highest success rate

CAAFS SLC-40 launch site has the lowest success rate

## ‘PolyLine` and distance between a launch site and the selected point



The distance between CCAFS SLC-40 launch site and:  
railways – 0,59km  
highway – 0,98 km  
Melbourne city – 54,08 km  
coastline – 0,87 km

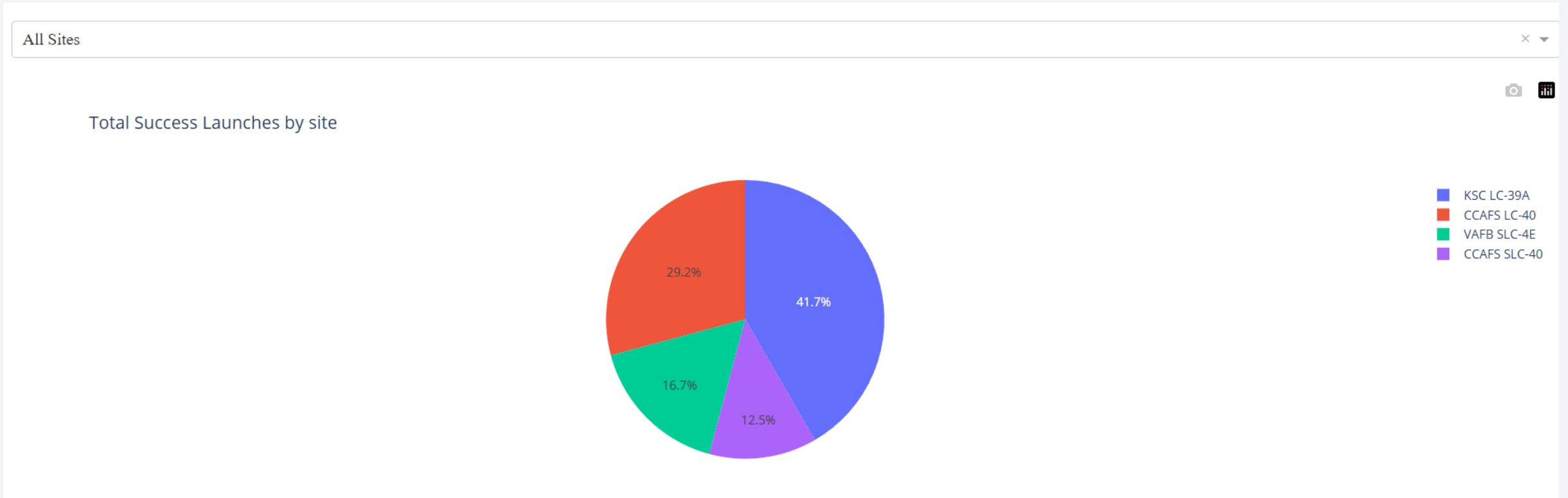




Section 4

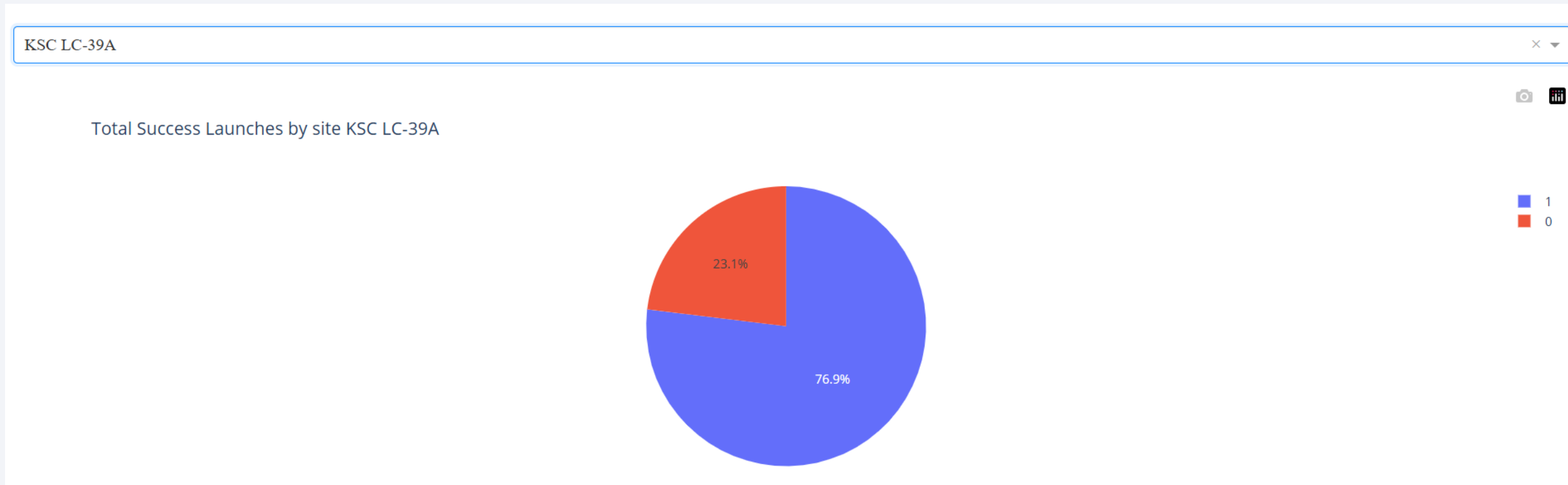
# Build a Dashboard with Plotly Dash

# Launch success count for all sites



KSC LC 39-A launch site has the highest success rate  
CCAFS SLC-40 launch site has the lowest success rate

# The launch site with highest launch success ratio



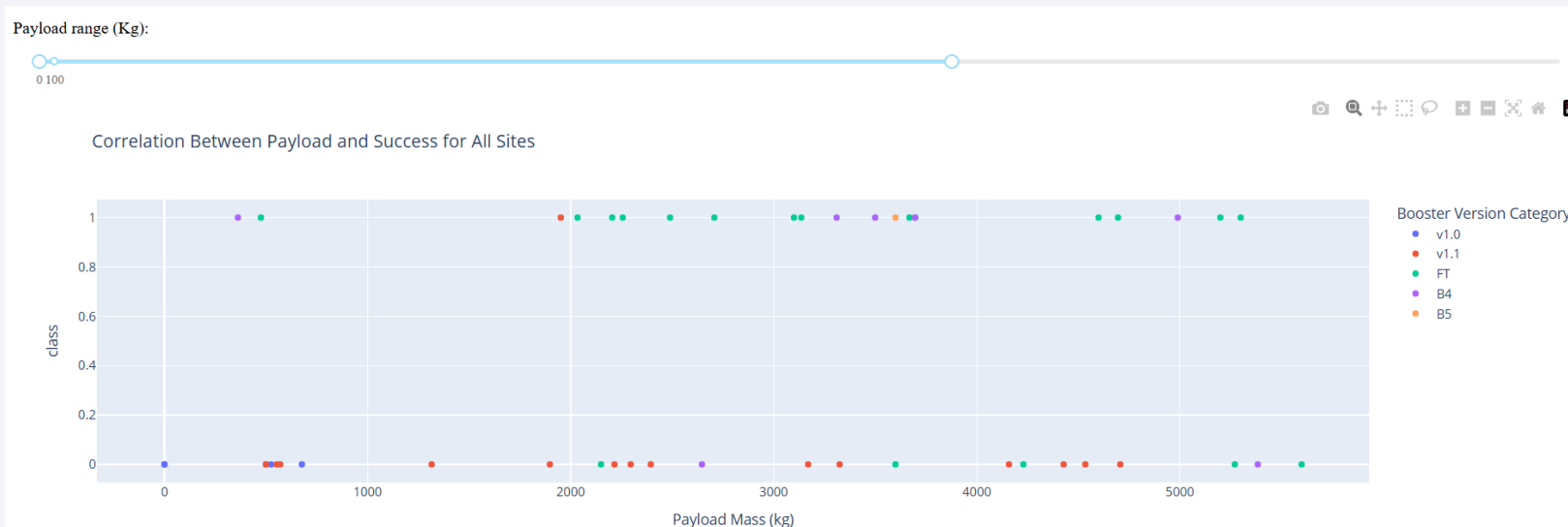
KSC LC 39-A launch site has 79,6% success launches and 23,1% failed launches

# Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider



The range from 2000kg to 5000kg payload mass has the most successful rate

Booster version FT has more successful rate



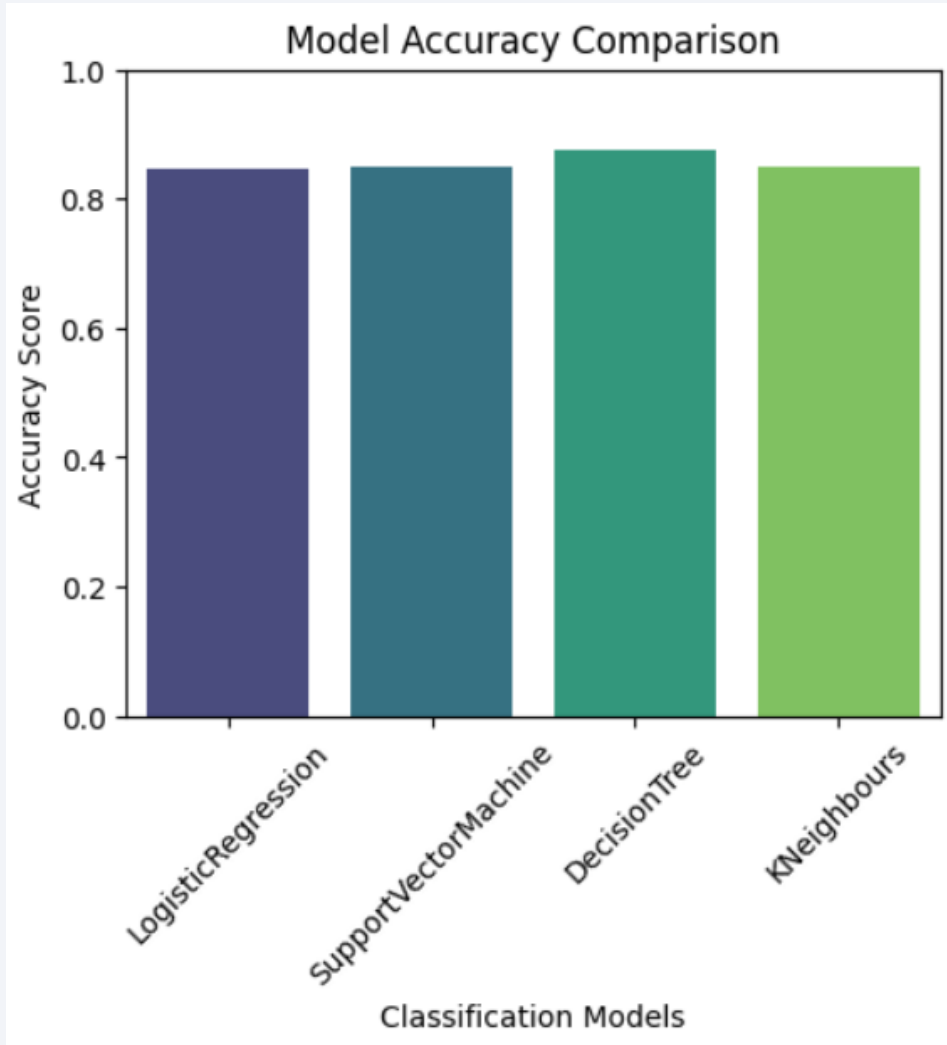
Section 5

# Predictive Analysis (Classification)



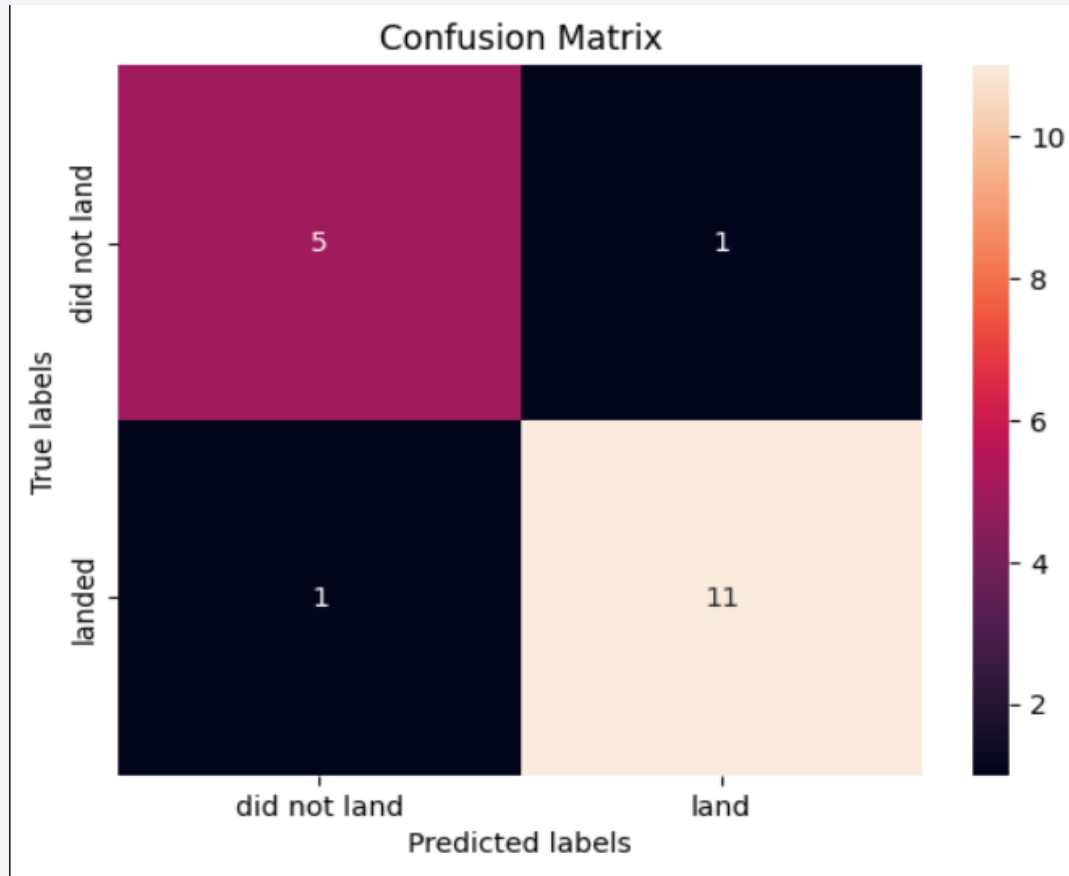
# Classification Accuracy

---



Decision Tree model have the highest accuracy score

# Confusion Matrix



Decision tree classifier model has the highest accuracy 88,8 % compared to other model accuracy 83,3%

# Conclusions

---

- The Decision tree classifier is the best model of machine learning for this dataset
- KSC LC 39-A launch site has the highest success rate
- The highest success rate has orbit types: ES-L1, GEO, HEO, SSO
- The greater the number of flights, the more successful the landing at all launch sites
- The most common payload mass up to 5800, and it has more successful landing on all launch sites

# Appendix

---

- Visual studio code
- Jupyter light
- `labs.cognitiveclass.ai`

Thank you!

