



IBM Developer  
SKILLS NETWORK

# Winning Space Race with Data Science

Atharv Vani  
UT Austin, Stats + DS



# Outline

---

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

# Executive Summary

---

- Summary of methodologies
  - Data Collection through API & w/ Webscraping
  - Data Wrangling
  - Exploratory Data Analytics using SQL
  - EDA w/ Data Viz
  - Interactive Visual Analytics using Folium
  - Dashboard using Plotly/Dash
  - Machine Learning Prediction
- Summary of all results
  - EDA Results
  - Interactive Analytics in maps
  - Predictions and Conclusions

# Introduction

---

- In this capstone, we will predict if the Falcon 9 first stage will land successfully. SpaceX 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 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. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch. In this module, you will be provided with an overview of the problem and the tools you need to complete the course.
- Problems you want to find answers
  - Which features are important for determining if the rocket will land successfully?
  - What statistics/key metrics are important to consider when determining accuracy?



Section 1

# Methodology

# Methodology

---

## Executive Summary

- Data collection methodology:
  - Data was collected and extracted through the API and webscraping(BeautifulSoup)
- Perform data wrangling
  - Data was processed through pandas df functions
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Consider feature importance/correlations and one-hot encoding for categorical variables
  - Used scikit-learn to build, train, test, and optimize various models for accuracy

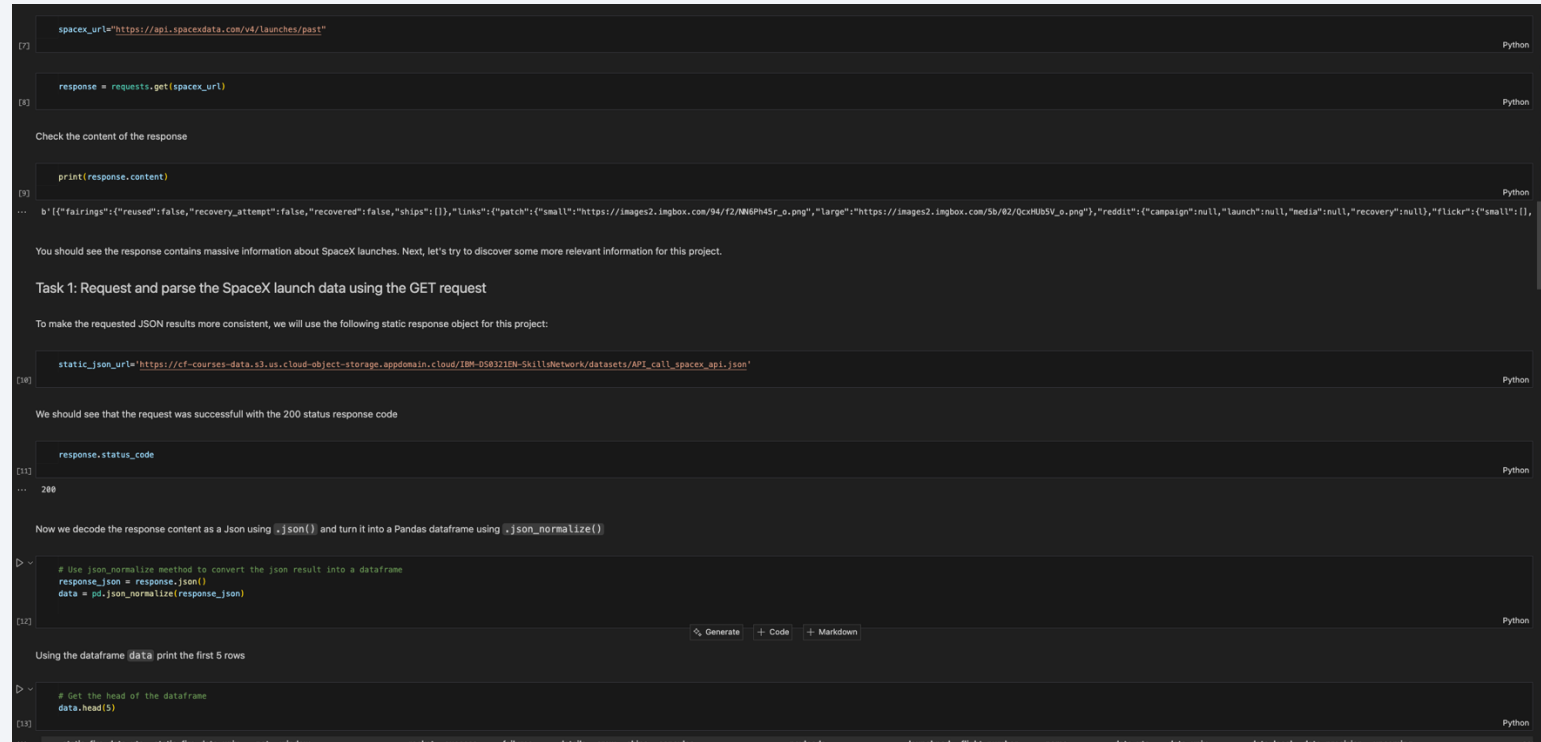
# Data Collection

---

- Data collection was done using get request to the SpaceX API
- Next, we decoded the response content as a Json using `.json()` function call and turn it into a pandas dataframe using `.json_normalize()`
- We then cleaned the data, checked for missing values and fill in missing values where necessary
- In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup
- The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis

# Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.
- Github Notebook - <https://github.com/atharvva/ni464/IBM-Applied-Capstone/blob/main/jupyter-labs-spacex-data-collection-api-v2.ipynb>



```
spacex_url="https://api.spacexdata.com/v4/launches/past"

response = requests.get(spacex_url)

Check the content of the response

print(response.content)

b'{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.com/94/12/N6Ph45r_o.png","large":"https://images2.imgbox.com/5b/02/OcxR0b5V_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[]},

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

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

static_json_url="https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json"

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

response.status_code

200

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

# Use json_normalize method to convert the json result into a dataframe
response_json = response.json()
data = pd.json_normalize(response_json)

Using the dataframe 'data' print the first 5 rows

# Get the head of the dataframe
data.head(5)
```



# Data Collection - Scraping

- Applied web scrapping to web scrap Falcon 9 launch records with BeautifulSoup
- Parsed the table and converted it into a pandas dataframe
- Github Notebook - <https://github.com/atharvvani464/IBM-Applied-Capstone/blob/main/jupyter-labs-webscraping.ipynb>

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"

headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
}

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

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

# use requests.get() method with the provided static_url and headers
# assign the response to a object
response = requests.get(static_url, headers=headers)

Create a BeautifulSoup object from the HTML response

# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text, 'html.parser')

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

# Use soup.title attribute
soup.title

<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about BeautifulSoup, please check the external reference link towards the end of this lab

# Use the find_all function in the BeautifulSoup object, with element type 'table'
# Assign the result to a list called 'html_tables'
html_tables = soup.find_all('table')
```

# Data Wrangling

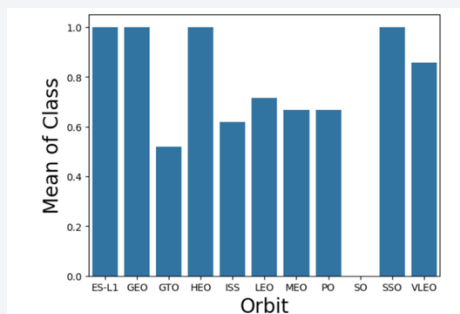
---

- Began by loading in the data and identifying important features and data types for columns
- Counted the number of launches per site and number of types of orbits
- Built a new binary feature representing the outcome of each launch
- Github Notebook: <https://github.com/atharvvani464/IBM-Applied-Capstone/blob/main/labs-jupyter-spacex-Data%20wrangling-v2.ipynb>

# EDA with Data Visualization

---

- Loaded the data using pandas pd and plot the relationship between FlightNumber and PayloadMass, LaunchSite(separate y-axis) to reveal for the VAFB-SLC launch site, there are no rockets launched for heavy payload mass(greater than 10000).
- Plot the success of orbit types, visualize the launch success trend, and encode dummy variables for a categorical column.
- Github Notebook: <https://github.com/atharvvani464/IBM-Applied-Capstone/blob/main/jupyter-labs-eda-dataviz-v2.ipynb>



# EDA with SQL

---

- We loaded the SpaceX dataset into a SQLite database by establishing a connection using a cursor object
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance – the total payload mass carried by boosters launched by NASA (CRS), the names of the unique launch sites in the space mission, and the failed landing outcomes in drone ship, their booster version and launch site names.
- Github Notebook: [https://github.com/atharvvani464/IBM-Applied-Capstone/blob/main/jupyter-labs-eda-sql-coursera\\_sqlite.ipynb](https://github.com/atharvvani464/IBM-Applied-Capstone/blob/main/jupyter-labs-eda-sql-coursera_sqlite.ipynb)

# 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.
- We calculated the distances between a launch site to its proximities. We answered 2 questions:
  - Are launch sites near railways, highways and coastlines
  - Do launch sites keep certain distance away from cities?
- Github Notebook: <https://github.com/atharvvani464/IBM-Applied-Capstone/blob/main/lab-jupyter-launch-site-location-v2.ipynb>



# Build a Dashboard with Plotly/Dash

---

- Built an interactive dashboard using components provided through the frameworks of Plotly/Dash
- Displayed pie charts showing the number of launches per each site
- Plotted a scatter graph showing the relationship of Outcome and Payload Mass (Kg) for the different booster version
- Github Notebook: <https://github.com/atharvvani464/IBM-Applied-Capstone/blob/main/app.py>

# Predictive Analysis (Classification)

---

- Loaded the data, preprocessed, and normalized using pandas/numpy
- Scikit-learn to split the training and testing sets and used GridSearchCV to find optimal hyperparameters for models
- Built and compared predictive performance of several ML methods: Logistic Regression, Support Vector Machine, Decision Tree, K-Nearest Neighbors
- Plotted confusion matrices to identify false predictions

# Results

---

- Exploratory data analysis query results
- Interactive analytics map in notebook
- Predictive analysis results



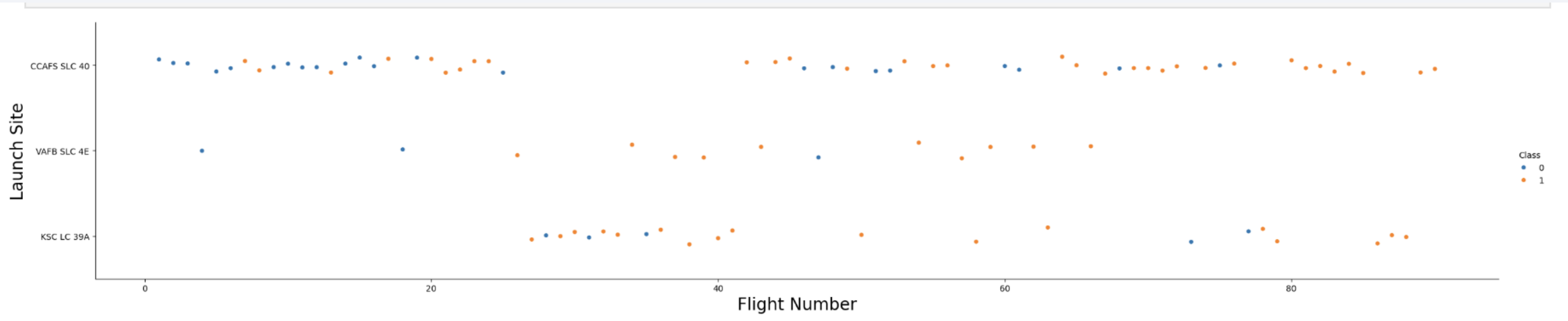


Section 2

# Insights drawn from EDA



# Flight Number vs. Launch Site

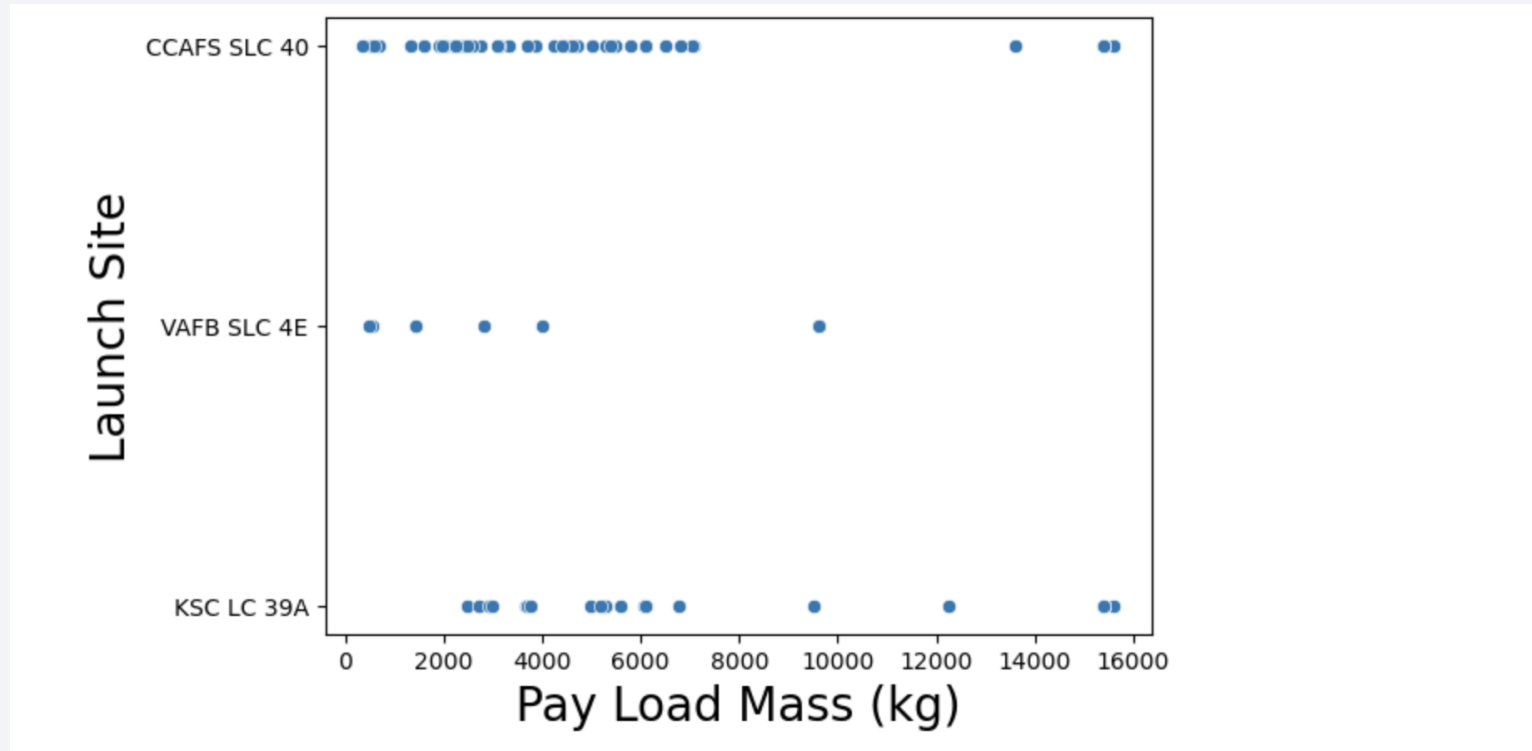


- Conclusion: The larger the flight amount at a launch site, the greater the success rate at a launch site



# Payload vs. Launch Site

---

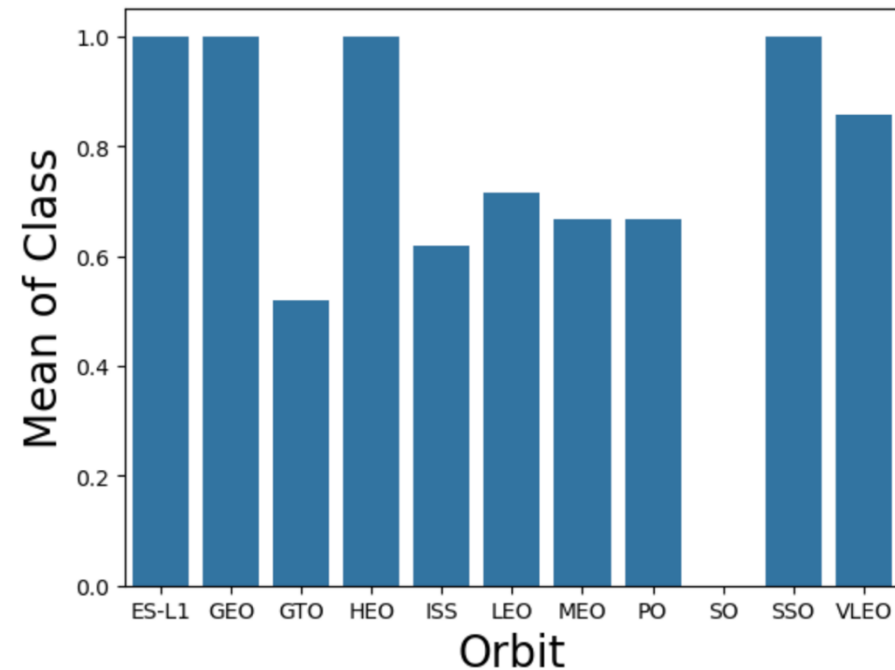


- Conclusion: The greater the payload mass amount for Launch Site CCAFS SLC 40, the higher the success rate for the rocket

# Success Rate vs. Orbit Type

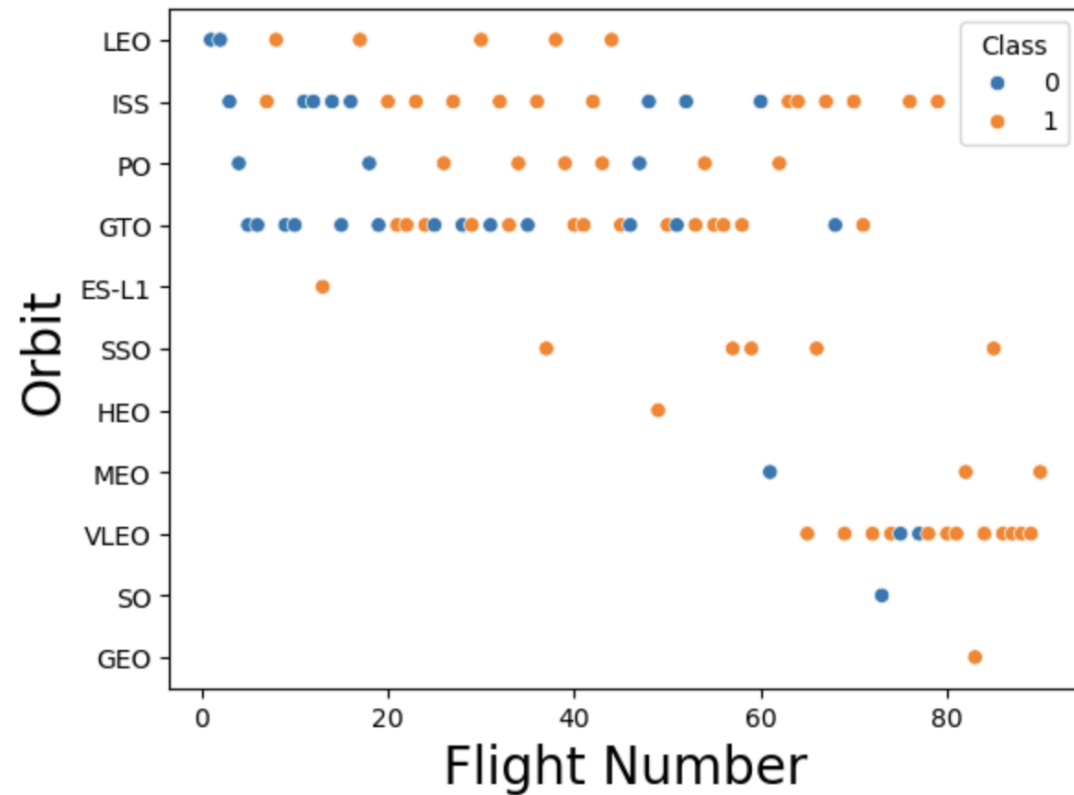
---

- Based on the graph, ES-L1, GEO, HEO, SSO, VLEO had the most success rate(top 5)



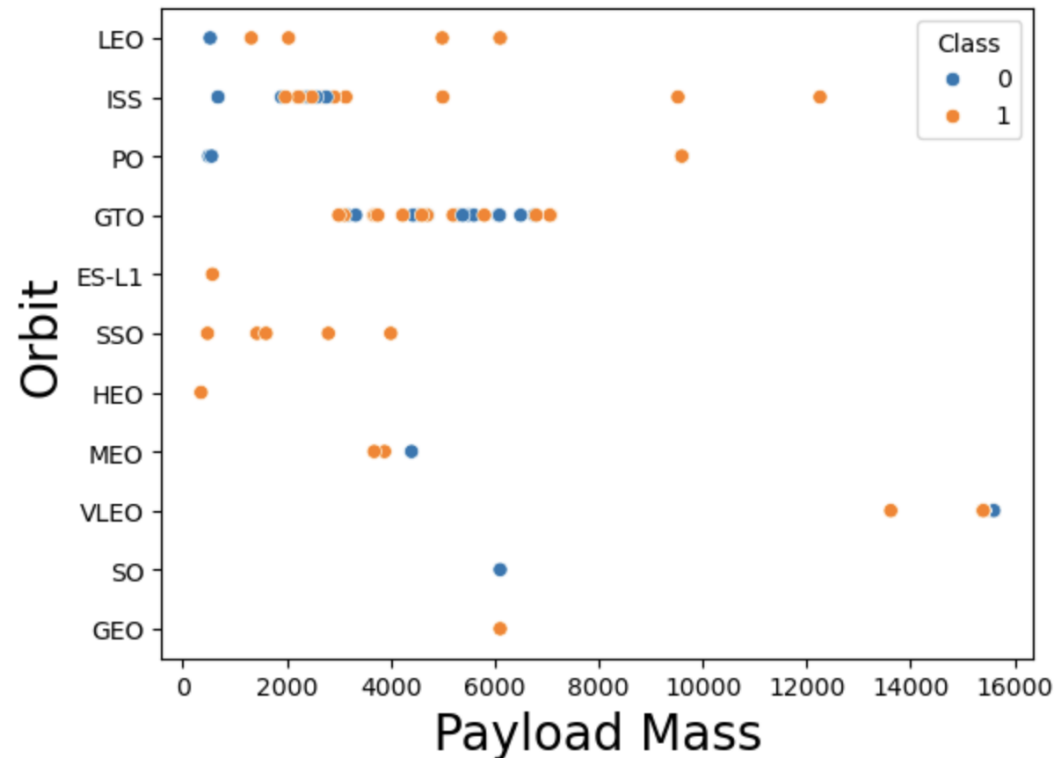
# Flight Number vs. Orbit Type

- In the LEO orbit the Success appears related to the number of flights
- There seems to be no relationship between flight number when in GTO orbit.



# Payload vs. Orbit Type

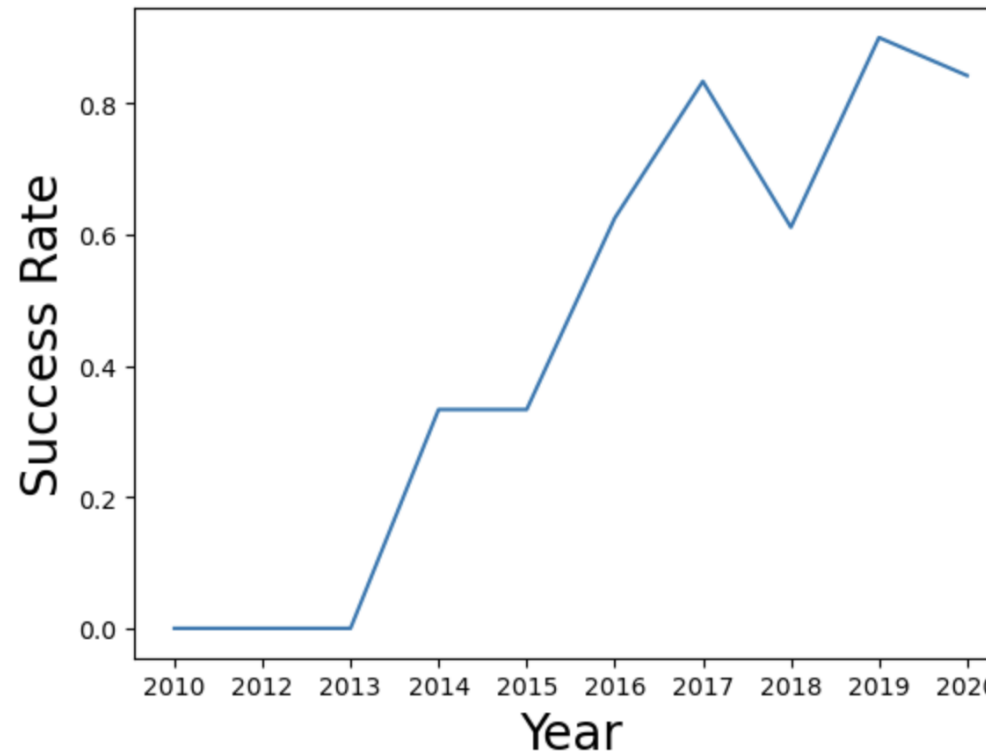
- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO, and ISS.
- However, for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccesful mission) are both there here.



# Launch Success Yearly Trend

---

- The success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.





# All Launch Site Names

---

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

In [11]: `%sql SELECT DISTINCT Launch_Site FROM SPACEXTABLE;`

\* sqlite:///my\_data1.db  
Done.

Out[11]: **Launch\_Site**

CCAFS LC-40

VAFB SLC-4E

KSC LC-39A

CCAFS SLC-40

# Launch Site Names Begin with 'CCA'

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

In [14]: `%sql SELECT * from SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;`

\* sqlite:///my\_data1.db  
Done.

Out[14]:

| Date       | Time (UTC) | Booster_Version | Launch_Site | Payload   | PAYLOAD_MASS__KG_ | Orbit     | Customer        | Mission_Outcome | Landing_   |
|------------|------------|-----------------|-------------|---|-------------------|-----------|-----------------|-----------------|------------|
| 2010-06-04 | 18:45:00   | F9 v1.0 B0003   | CCAFS LC-40 | Dragon Spacecraft Qualification Unit                          | 0                 | LEO       | SpaceX          | Success         | Failure (p |
| 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 (p |
| 2012-05-22 | 7:44:00    | F9 v1.0 B0005   | CCAFS LC-40 | Dragon demo flight C2   | 525               | LEO (ISS) | NASA (COTS)     | Success         | N          |
| 2012-10-08 | 0:35:00    | F9 v1.0 B0006   | CCAFS LC-40 | SpaceX CRS-1  | 500               | LEO (ISS) | NASA (CRS)      | Success         | N          |
| 2013-03-01 | 15:10:00   | F9 v1.0 B0007   | CCAFS LC-40 | SpaceX CRS-2  | 677               | LEO (ISS) | NASA (CRS)      | Success         | N          |

# Total Payload Mass

---

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

```
In [18]: %sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total_Payload_Mass" FROM SPACEXTABLE WHERE Customer = 'NASA (CRS)';
```

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

```
Out[18]: Total_Payload_Mass  
         45596
```

# Average Payload Mass by F9 v1.1

---

## Task 4

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

In [19]: `%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "AVG_Payload_Mass" FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1';`

\* sqlite:///my\_data1.db  
Done.

Out[19]: 

| <u>AVG_Payload_Mass</u> |
|-------------------------|
| 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*

```
%sql SELECT MIN(Date) AS FirstSuccessfull_landing_date FROM SPACEXTABLE WHERE Landing_Outcome LIKE 'Success (ground pad)';
```

[17]

... \* [sqlite:///my\\_data1.db](#)

Done.

... **FirstSuccessfull\_landing\_date**

2015-12-22



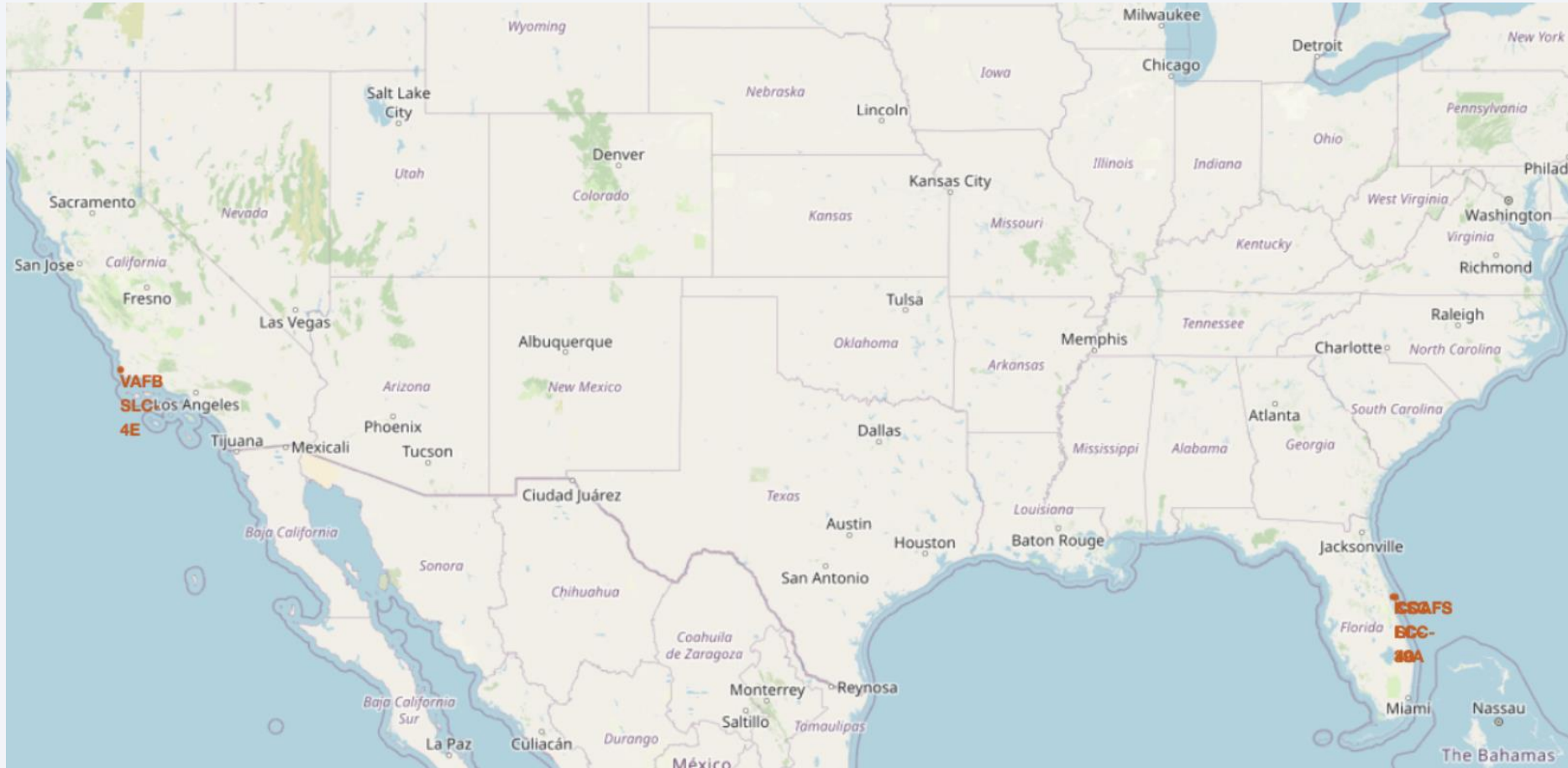
A satellite view of Earth from space, showing the curvature of the planet and city lights at night. The image is a composite of a solid blue background on the left and a satellite photograph of Earth on the right. The Earth's surface is dark, with numerous bright yellow and orange lights representing cities and urban areas. The horizon of the Earth is visible as a curved line separating the dark surface from the deep blue of space.

Section 3

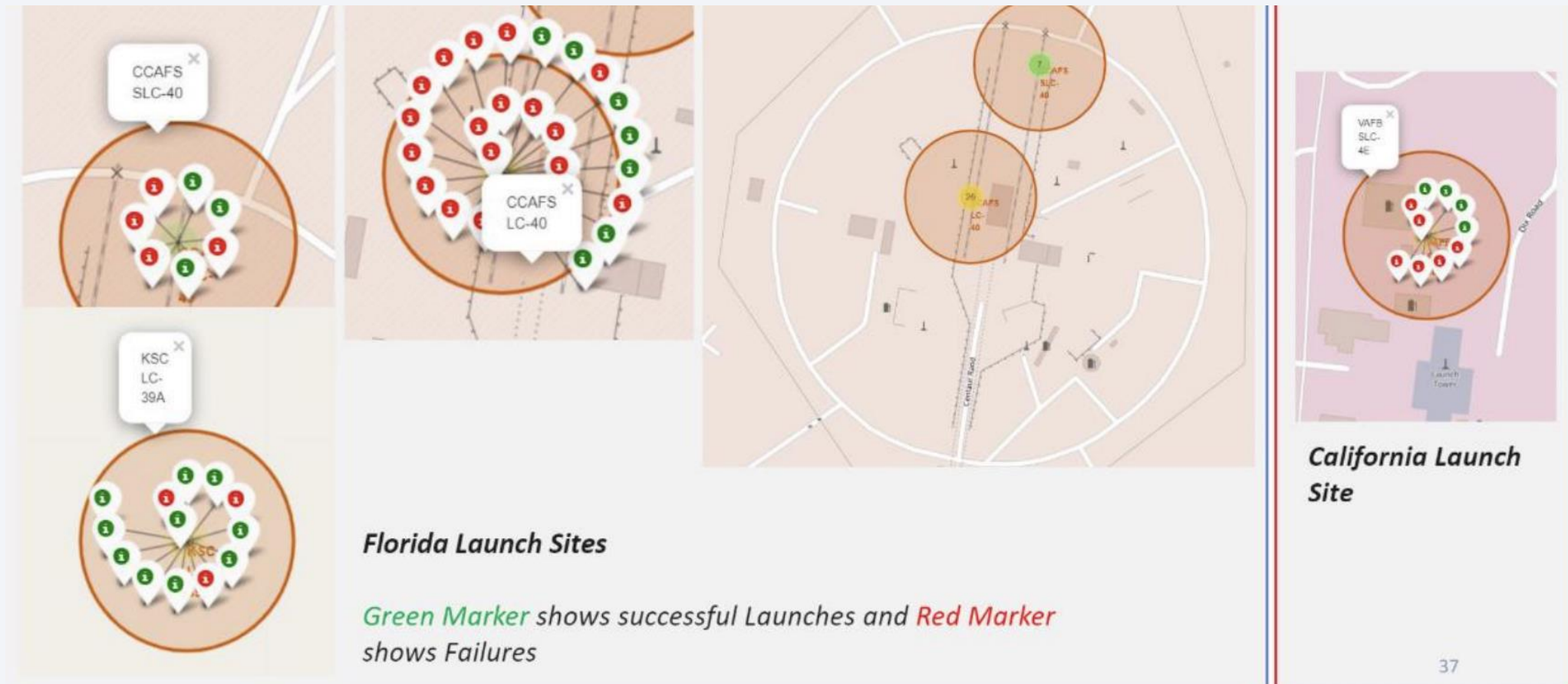
# Launch Sites Proximities Analysis

# Global Launch Sites

---



# Markers showing Launch sites





# Launch Site distances to proximities



- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes



Section 4

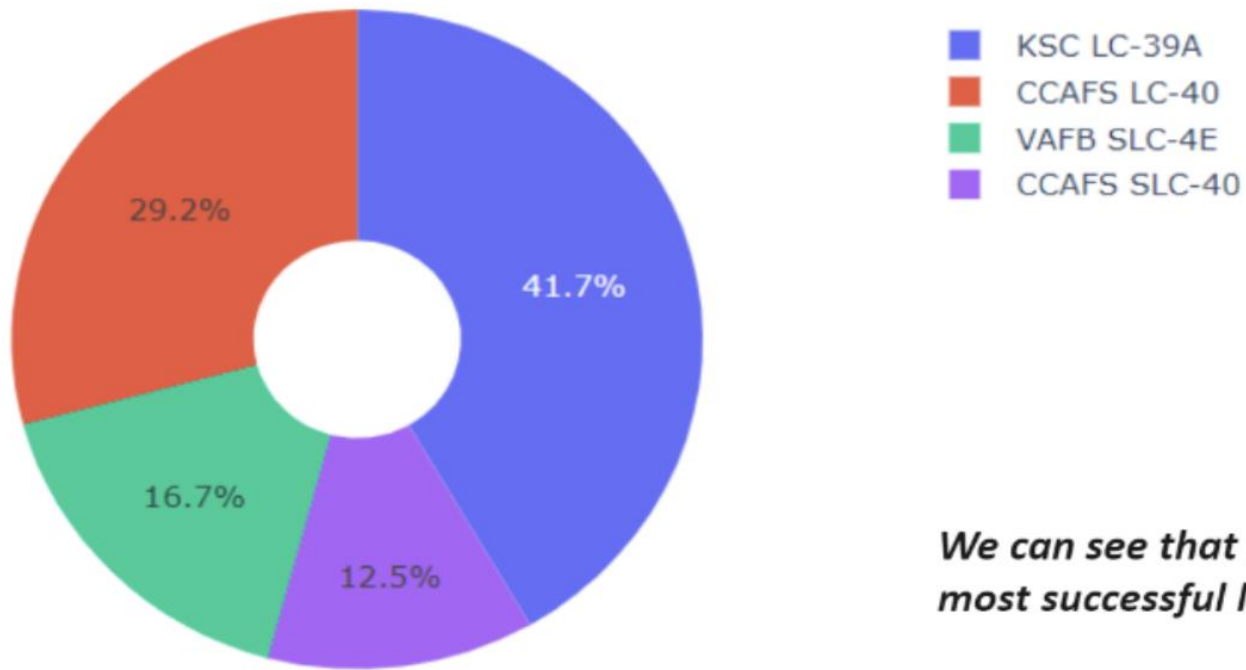
# Build a Dashboard with Plotly Dash



# Success per Launch site

---

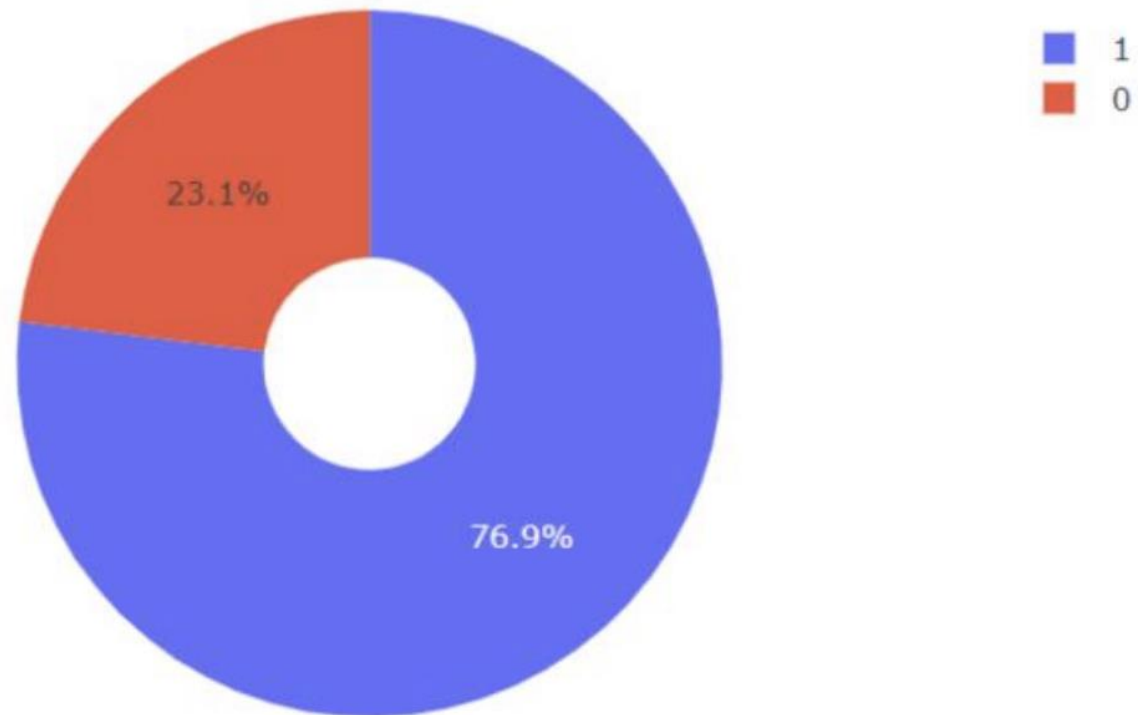
Total Success Launches By all sites



*We can see that KSC LC-39A had the most successful launches from all the sites*

# Launch site with highest success ratio

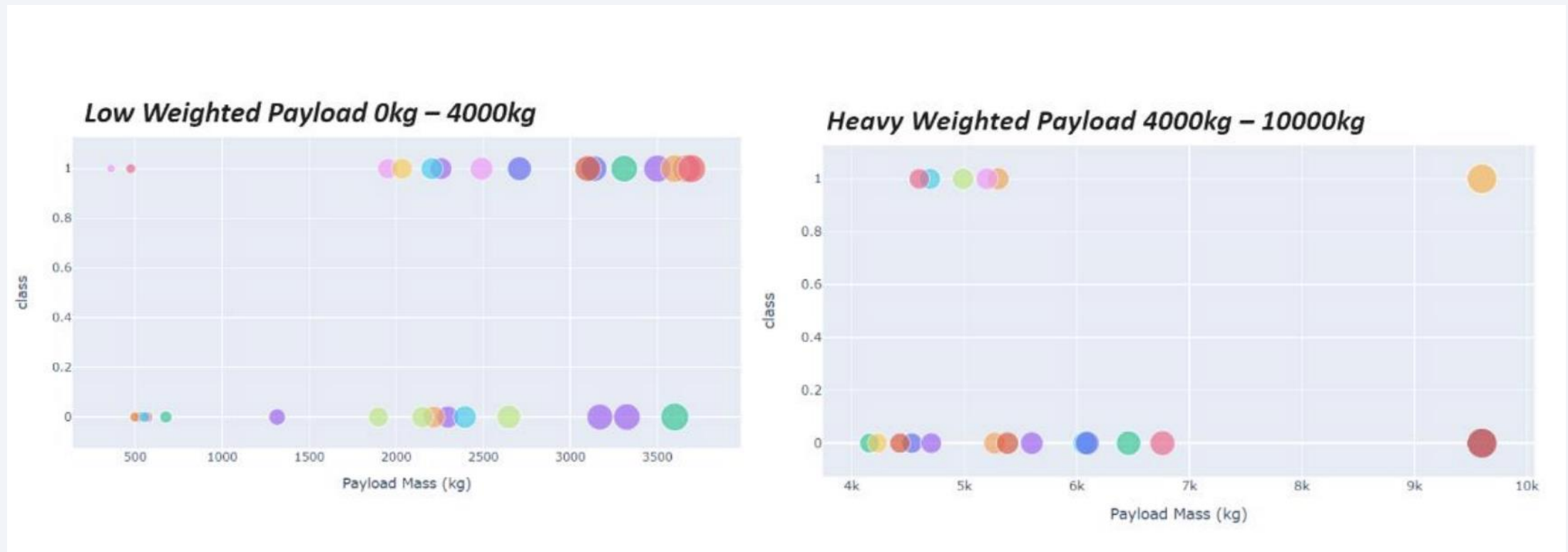
---



***KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate***



# Payload(slidebar) vs Launch Outcomes for sites



Section 5

# Predictive Analysis (Classification)

# Classification Accuracy

- Highest Classification Accuracy: Logistic Regression Model, 84.6%

Create a logistic regression object then create a GridSearchCV object `logreg_cv` with `cv = 10`. Fit the object to find the best parameters from the dictionary `parameters`.

```
parameters = {'C': [0.01, 0.1, 1],
              'penalty': ['l2'],
              'solver': ['lbfgs']}
```

✓ 0.0s Python

```
parameters = {"C": [0.01, 0.1, 1], 'penalty': ['l2'], 'solver': ['lbfgs']} # l1 lasso l2 ridge
logreg = LogisticRegression()
logreg_cv = GridSearchCV(logreg, parameters, cv=10)
logreg_cv.fit(X_train, Y_train)
```

✓ 0.0s Python

```
GridSearchCV
└─ best_estimator_:
   └─ LogisticRegression
      └─ LogisticRegression
```

```
print("tuned hpyerparameters :(best parameters) ", logreg_cv.best_params_)
print("accuracy :", logreg_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters) {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

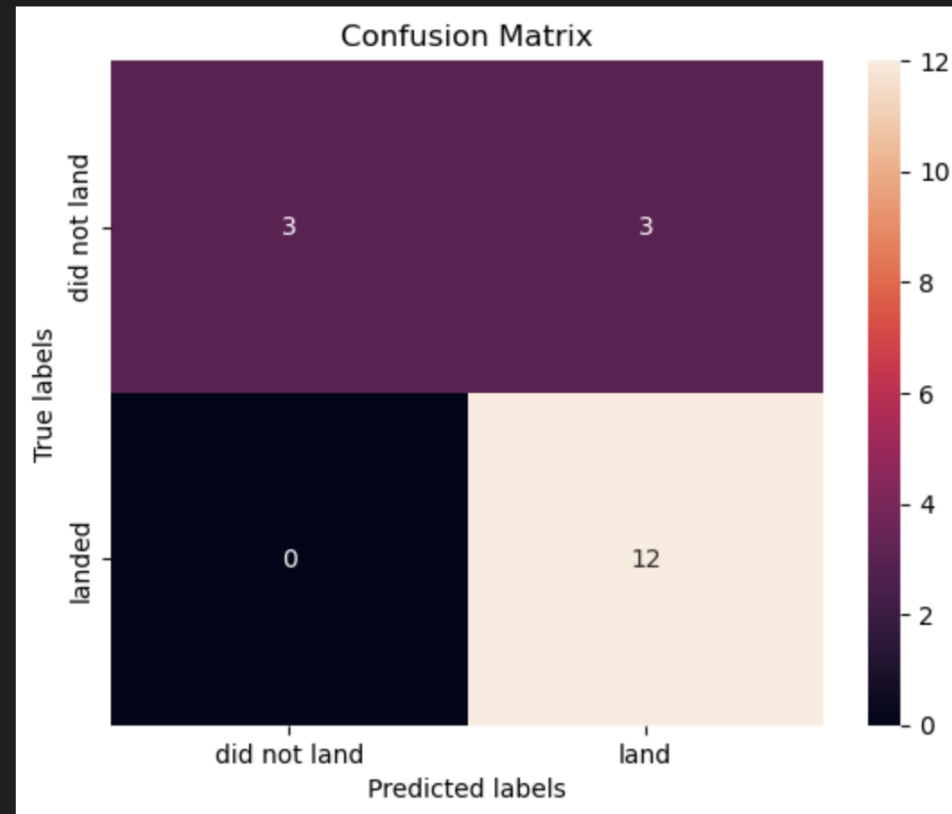
# Confusion Matrix

- The confusion matrix for the logistic regression classifier shows that the classifier can distinguish between the different classes. The problem is mainly the false positives, or unsuccessful landings marked as successful landing by the classifier.

Lets look at the confusion matrix:

```
yhat=logreg_cv.predict(X_test)  
plot_confusion_matrix(Y_test,yhat)
```

✓ 0.0s



# Conclusions and Business Stakeholders

---

- The larger the flight amount at a launch site, the greater the success rate at a launch site
- Launch success rate started to increase in 2013 till 2020
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate
- KSC LC-39A had the most successful launches of any sites.
- The Logistic Regression classifier is the best ML algorithm for this specific domain related task.

# Appendix

---

- Github Project: <https://github.com/atharvvani464/IBM-Applied-Capstone/tree/main>
- Coursera: <https://www.coursera.org/learn/applied-data-science-capstone>



Thank you!

