IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Christopher P
2023/12/20

Outline

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

Executive Summary

❖**Summary of methodologies**

- Data Collection with API and Web Scraping
- Data Wrangling
- Exploratory Data Analysis with SQL
- Data Visualization
- Interactive Visual Analysis with Folium
- Prediction with Machine Learning

❖**Summary of all results**

Introduction

❖Project background and context

- Exploration of SpaceX's notable achievements and contributions to space exploration
- Introduction to the Falcon 9 rocket and its groundbreaking feature—reusability
- Falcon 9 launches priced at 62 million dollars, in contrast to competitors, which normally charge more than 165 million
- Understanding the need to predict the success of Falcon 9 first stage landings

❖Problems you want to find answers

- What features affect the rocket's likelihood of a successful first stage landing
- What models best predict the successfulness of first stage landings
- Performance insights for different elements

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Gather data with SpaceX API and web scraping from wiki
- Perform data wrangling
  - One-hot encoding for 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
  - Find best parameters with cross validation
  - Implement model with the best accuracy score

Data Collection

- Describe how data sets were collected

1. Web Scraping with SpaceX API
2. Using JSON functions to decode the response content and convert to pandas dataframe
3. Data Cleaning
4. Implemented Beautiful Soup for launch records of Falcon 9

Data Collection – SpaceX API

- GitHub URL: https://github.com/Chrispy1204/IBM_data_science_capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

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)
```

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:

In [9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/d
```

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

In [10]:
```
response.status_code
```

Out[10]: 200

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

In [15]:
```
# Use json_normalize meethod to convert the json result into a dataframe
json_data = response.json()
data = pd.json_normalize(json_data)

```

In [17]:
```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Data Collection - Scraping

- GitHub URL
- Using Beautiful Soup for web scraping and gathered Falcon 9 launch records. Also extracted the columns name we needed.

In [4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=102768
```

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.

In [6]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

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

In [7]:
```
# 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

In [8]:
```
# Use soup.title attribute
print("Page Title:", soup.title.text)
```

```
Page Title: List of Falcon 9 and Falcon Heavy launches - Wikipedia
```

In [11]:
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

th_elements = first_launch_table.find_all('th')
for th_element in th_elements:
        name = extract_column_from_header(th_element)
        if name is not None and len(name) > 0:
            column_names.append(name)
```

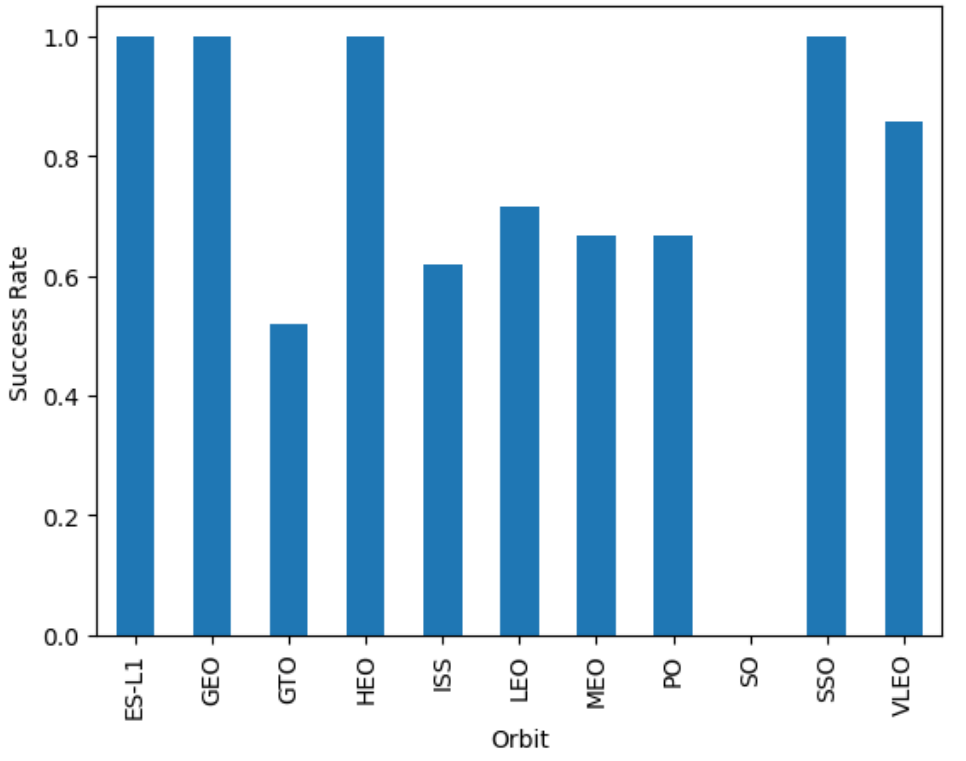
Data Wrangling

- From the findings of our Exploratory Data Analysis, we identified the labels for our dataframe
- Calculated the number and occurrence of each orbit and the mission outcome
- Added a ’Class’ column to determine successful landing
- GitHub

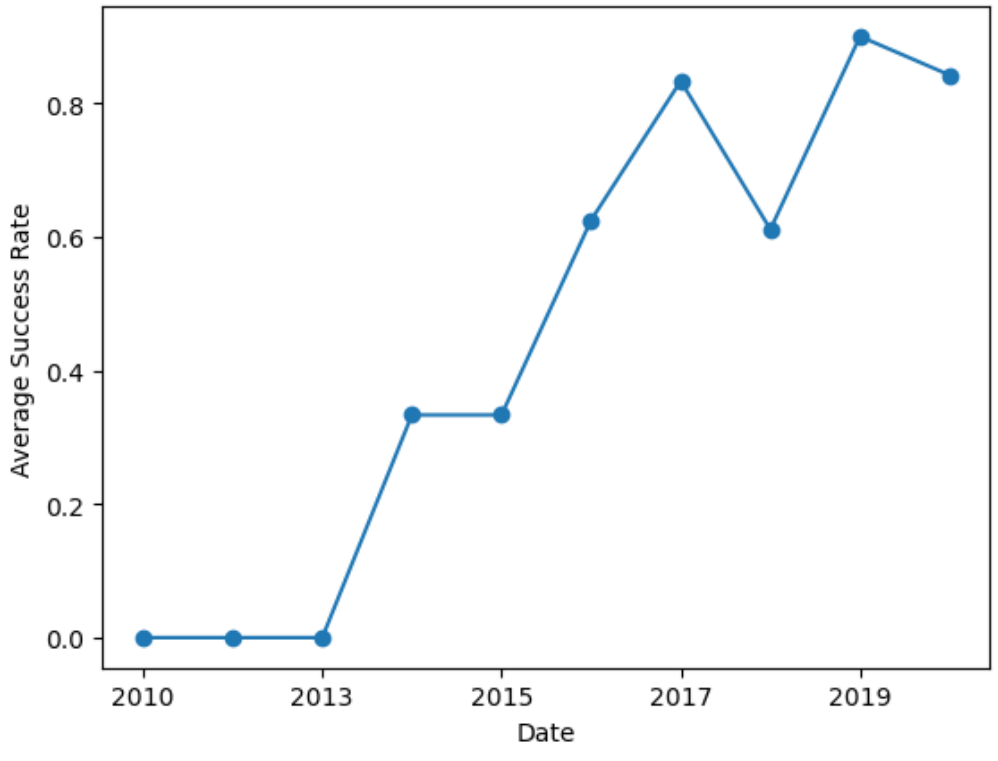
EDA with Data Visualization

- GitHub

Plot for Success Rate vs Orbit



Plot for Average Successful Rate vs Year

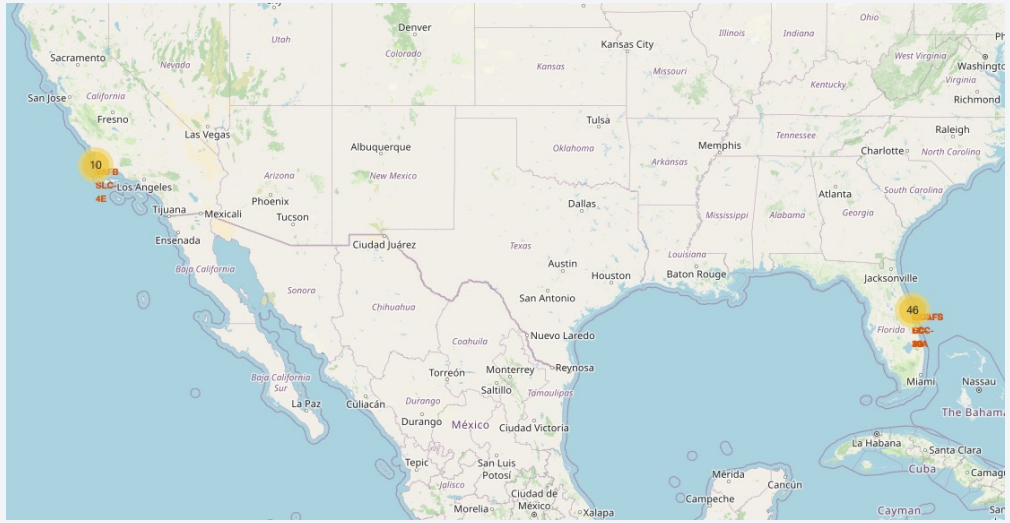


EDA with SQL

- We used SQL queries to extract more information from the SpaceX dataset:
  - Distinct launch site names
  - Total Mass Payload that NASA CRS boosters carried
  - The F9 v1.1 rocket's average payload mass
  - Total number of outcomes from missions that were successful or unsuccessful
  - Name of launch sites and booster version with Unsuccessful Drone Ship landing results
- GitHub

Build an Interactive Map with Folium

- Initialize map and add folium circle and marker for launch sites using NASA coordinates
- Measure distance between launch sites and coastlines and its surroundings, including cities, highways, trains, and beaches.
- Green stands for successful launches and red stands for failures
- GitHub

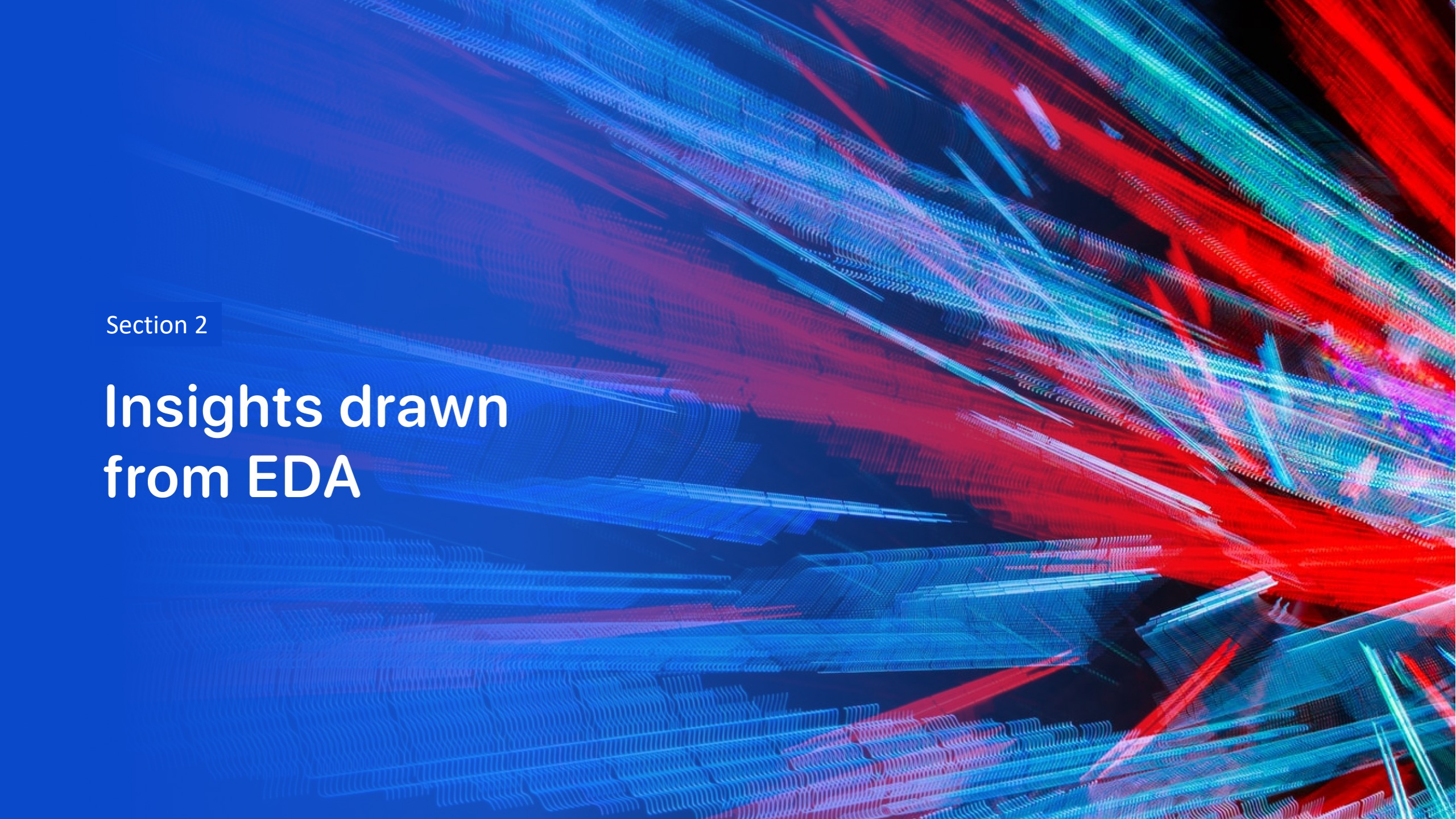


Predictive Analysis (Classification)

- Download dataset
- Data Preprocessing
  - Data cleaning, train_test_split
- Parameter Tuning
  - GridSearchCV, feature selection
- Model Implementation
  - Model fit, accuracy score, confusion matrix
- GitHub

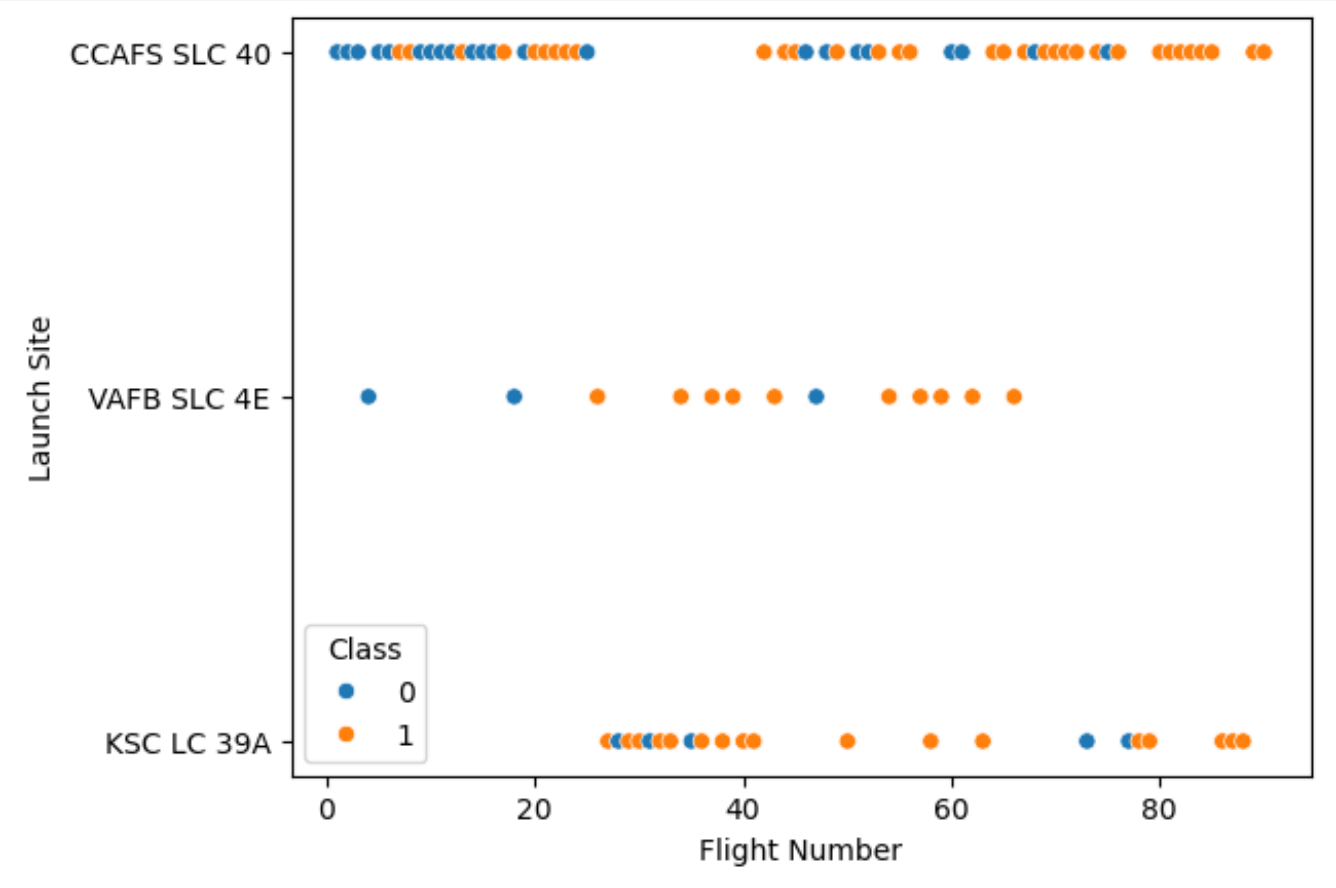
Results

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


Section 2
Insights drawn from EDA

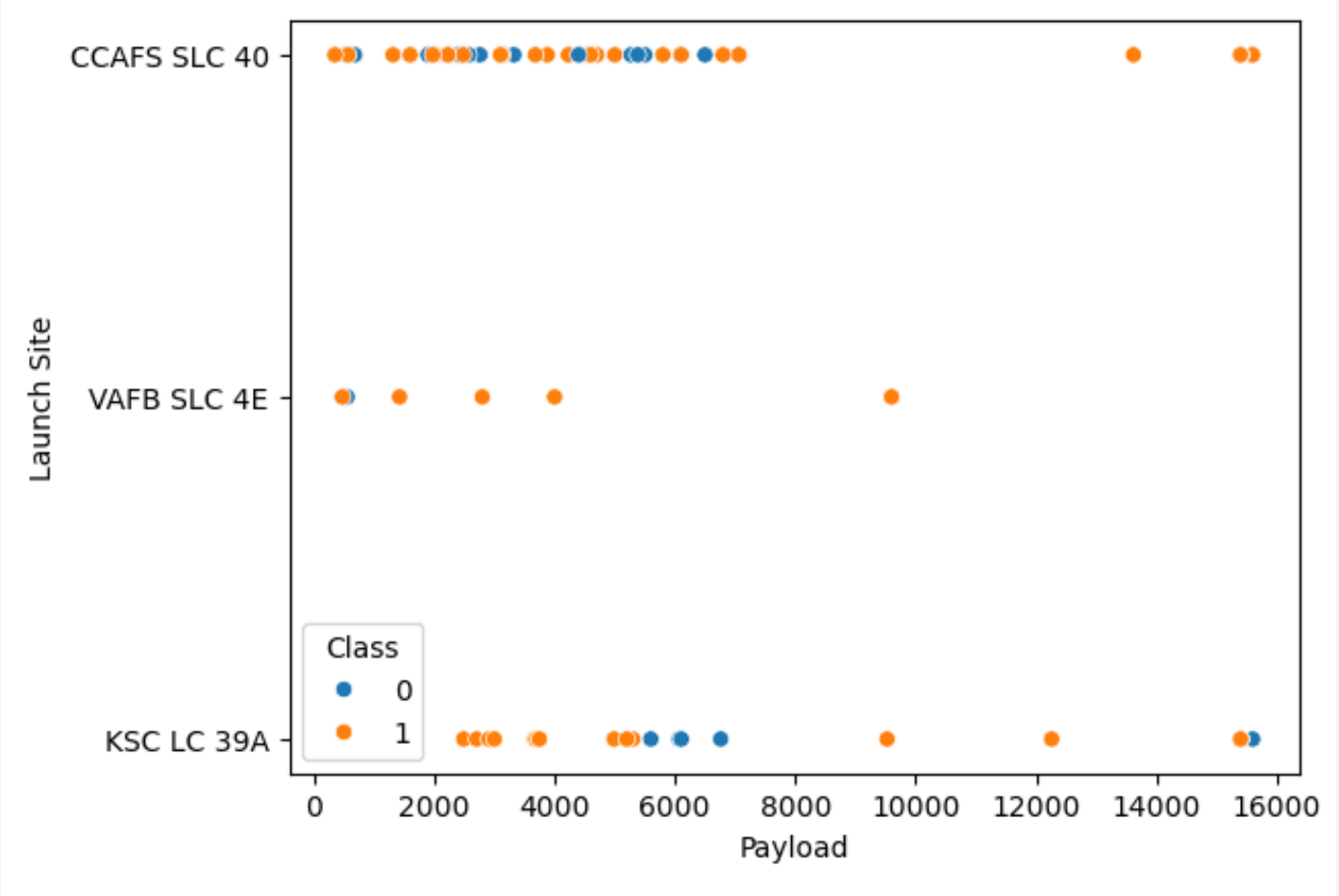
Flight Number vs. Launch Site

- The higher the number of flights, the more successful are the landings
- CCAFS SLC 40 has the most flights



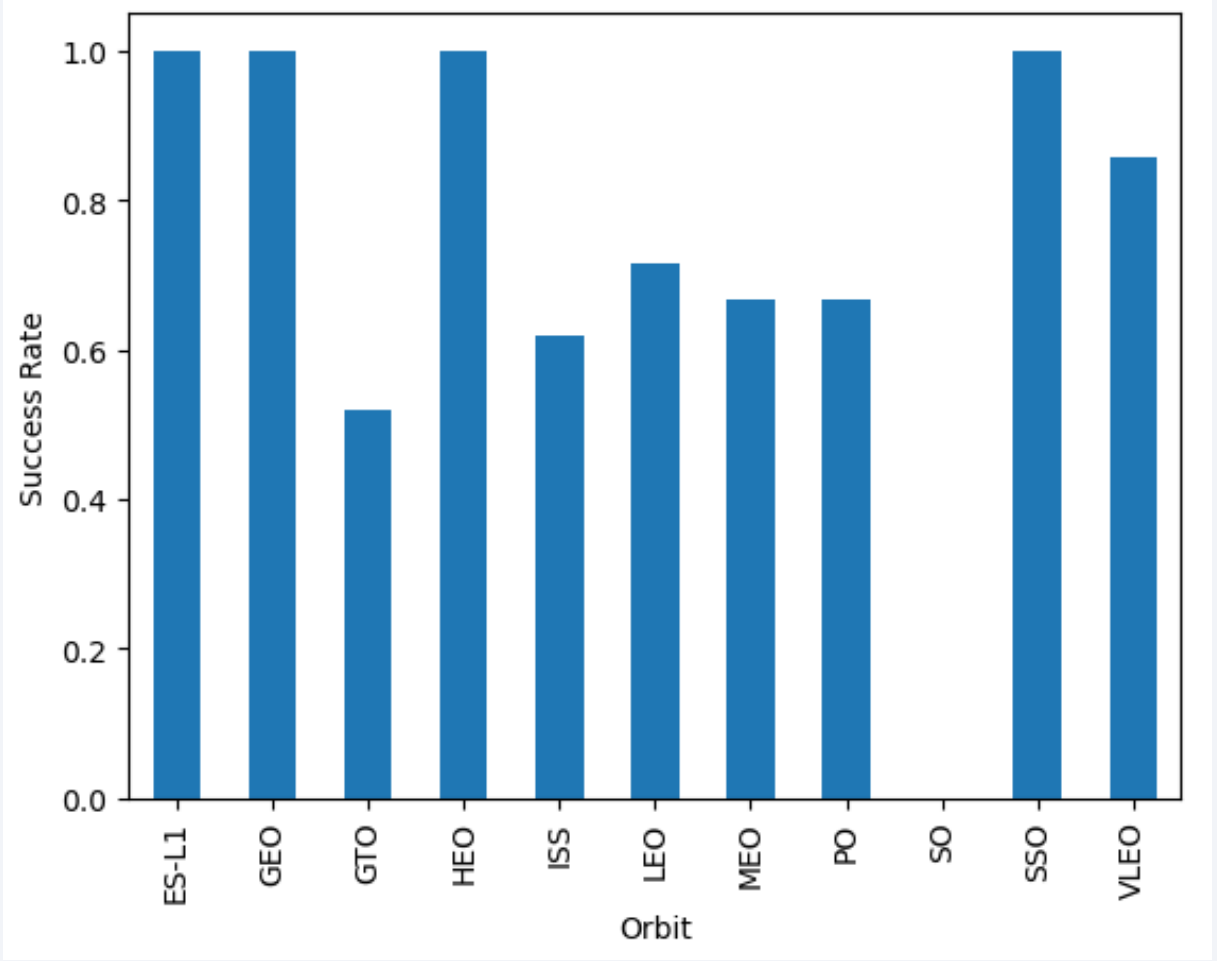
Payload vs. Launch Site

- There is a rise in success rate as the payload mass increases
- No rocket with a payload mass more than 10,000 has been launched from the VAFB SLC 4E launch site



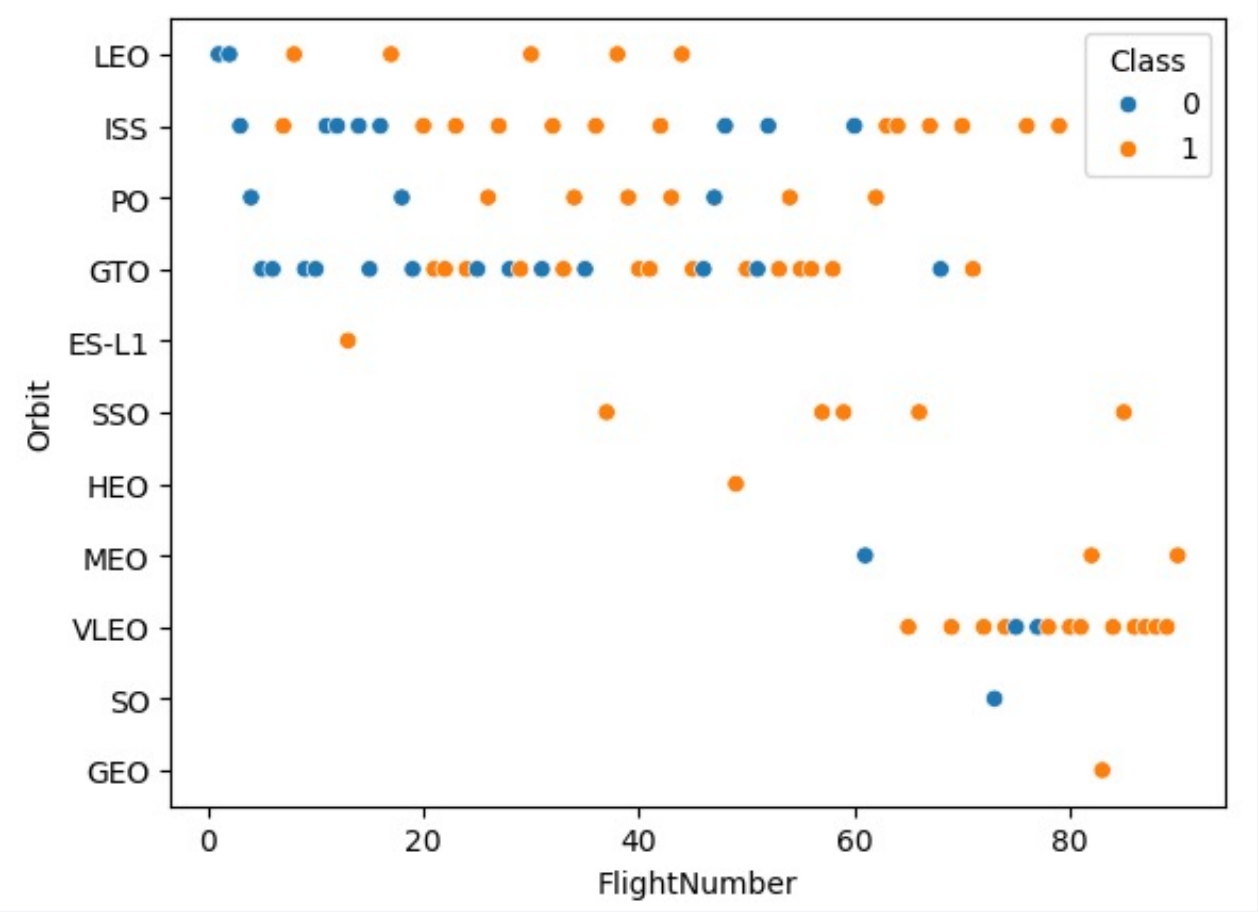
Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, SSO all have the highest success rate
- SO has 0 success rate



Flight Number vs. Orbit Type

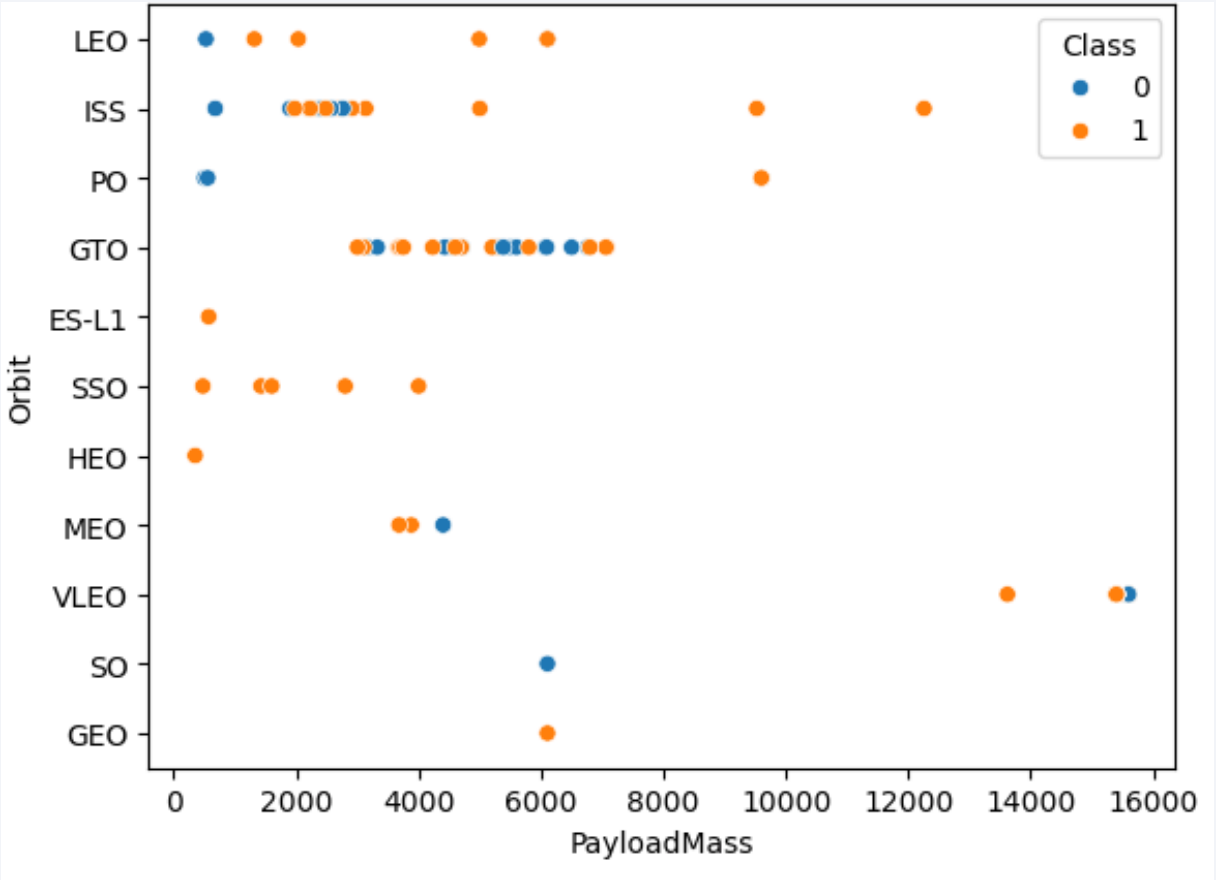
- For the LEO orbit, the higher the flight number the more success landings
- For the other orbits, there isn't a significant relationship inbetween



Payload vs. Orbit Type

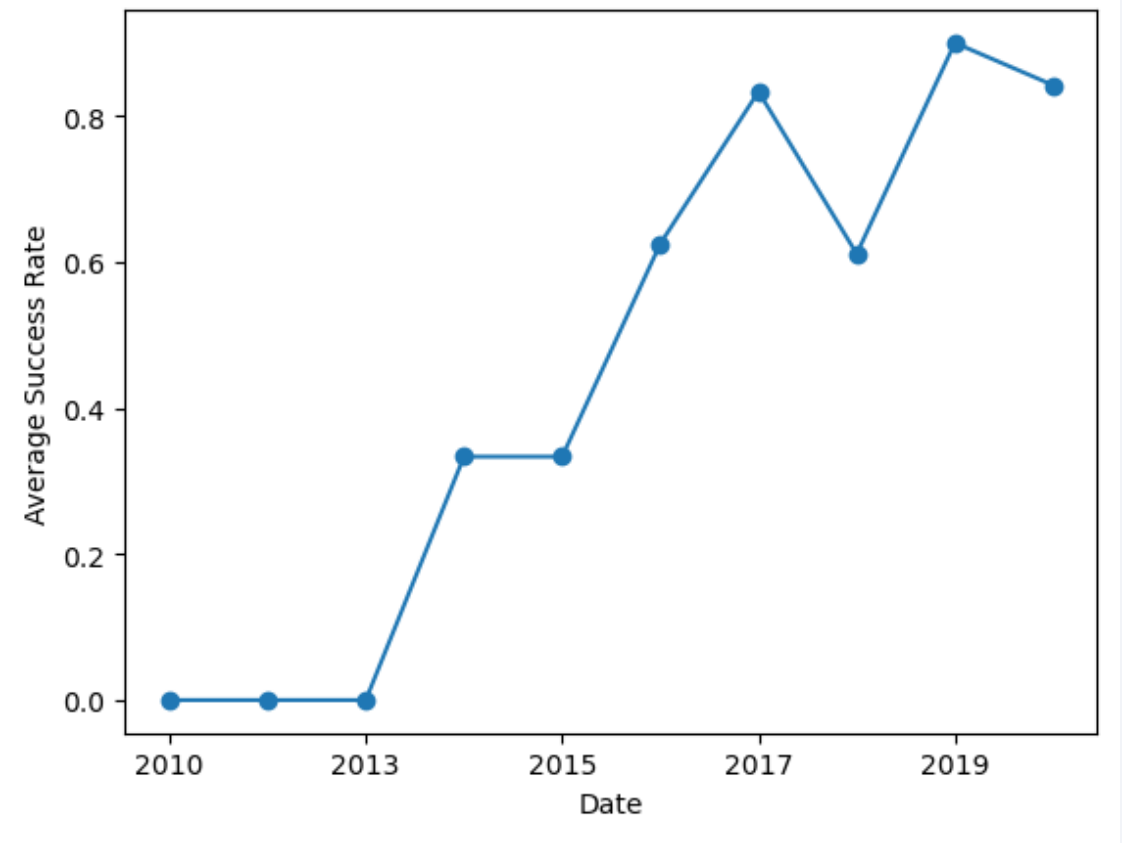
- GTO and ISS does not show a strong relationship with payload mass

- LEO, ISS, PO have high successful landings with larger payload mass



Launch Success Yearly Trend

- The Average Successful Rate increases through years
- However, there is a decline in 2018 that needs to be investigated



All Launch Site Names

```
In [12]:  1  %sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

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

Out[12]:

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

Launch Site Names Begin with 'CCA'

Task 2

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

In [15]:
```
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
```

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

Out[15]:

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

Total Payload Mass

Task 3

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

In [16]:
```
%sql SELECT SUM("PAYLOAD_MASS__KG_") AS "Total Payload Mass" FROM SPACEXTABLE WHERE "Customer" LIKE 'NASA (CRS)%
```

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

Out[16]:

Total Payload Mass
48213

Average Payload Mass by F9 v1.1

Task 4

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

In [17]:
```
1 %sql SELECT AVG("PAYLOAD_MASS__KG_") AS "Average Payload Mass" FROM SPACEXTABLE WHERE "Booster_Version" LIKE 'F9
2
```

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

Out[17]:

Average Payload Mass
2928.4

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 [18]:
```
%sql SELECT MIN("Date") AS "First Successful Landing Date" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (
```

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

Out[18]:

First Successful Landing Date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [19]:
```
%sql SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "PA
```

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

Out[19]:

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

- Only 4 boosters have success in drone ship and have payload mass between 4000 and 6000

Total Number of Successful and Failure Mission Outcomes

Task 7 ¶

List the total number of successful and failure mission outcomes

In [20]:
```
1 %sql SELECT "Mission_Outcome", COUNT(*) AS "Count" FROM SPACEXTABLE GROUP BY "Mission_Outcome";
2
```

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

Out[20]:

Mission_Outcome	Count
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

- Success vs Failure = 100:1

Boosters Carried Maximum Payload

Task 8

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

In [21]:
```
%sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM
```

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

Out[21]:

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 [22]:
```
%sql SELECT SUBSTR("Date", 6, 2) AS "Month", "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTABL
```

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

Out[22]:

Month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

Task 10

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.

In [23]:
```
%sql SELECT "Landing_Outcome", COUNT(*) AS "Count", RANK() OVER (ORDER BY COUNT(*) DESC) AS "Rank" FROM SPACEXTA
```

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

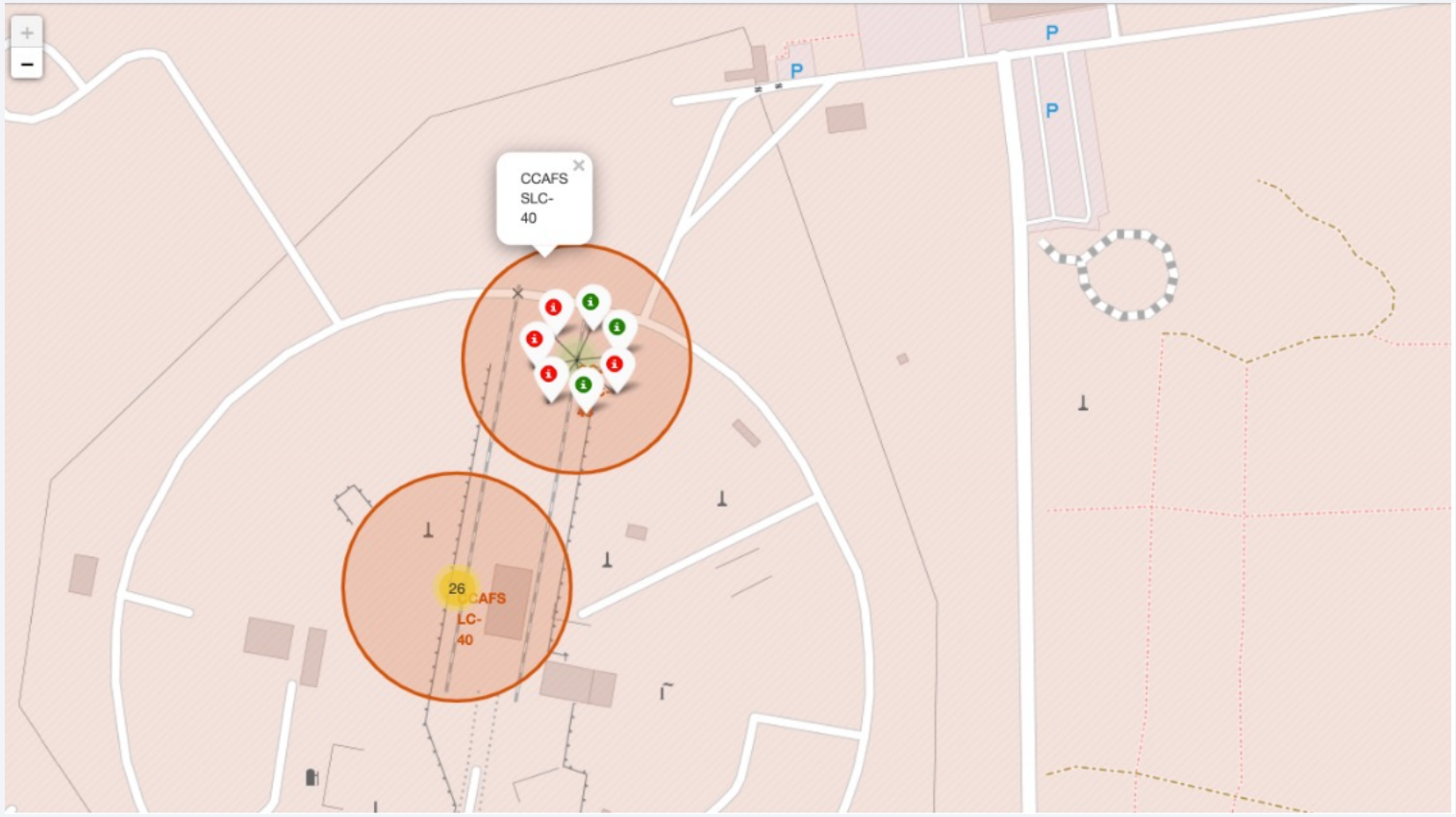
Out[23]:

Landing_Outcome	Count	Rank
No attempt	10	1
Success (drone ship)	5	2
Failure (drone ship)	5	2
Success (ground pad)	3	4
Controlled (ocean)	3	4
Uncontrolled (ocean)	2	6
Failure (parachute)	2	6
Precluded (drone ship)	1	8


Section 3
Launch Sites
Proximities Analysis

Marking All Launch Sites on Map

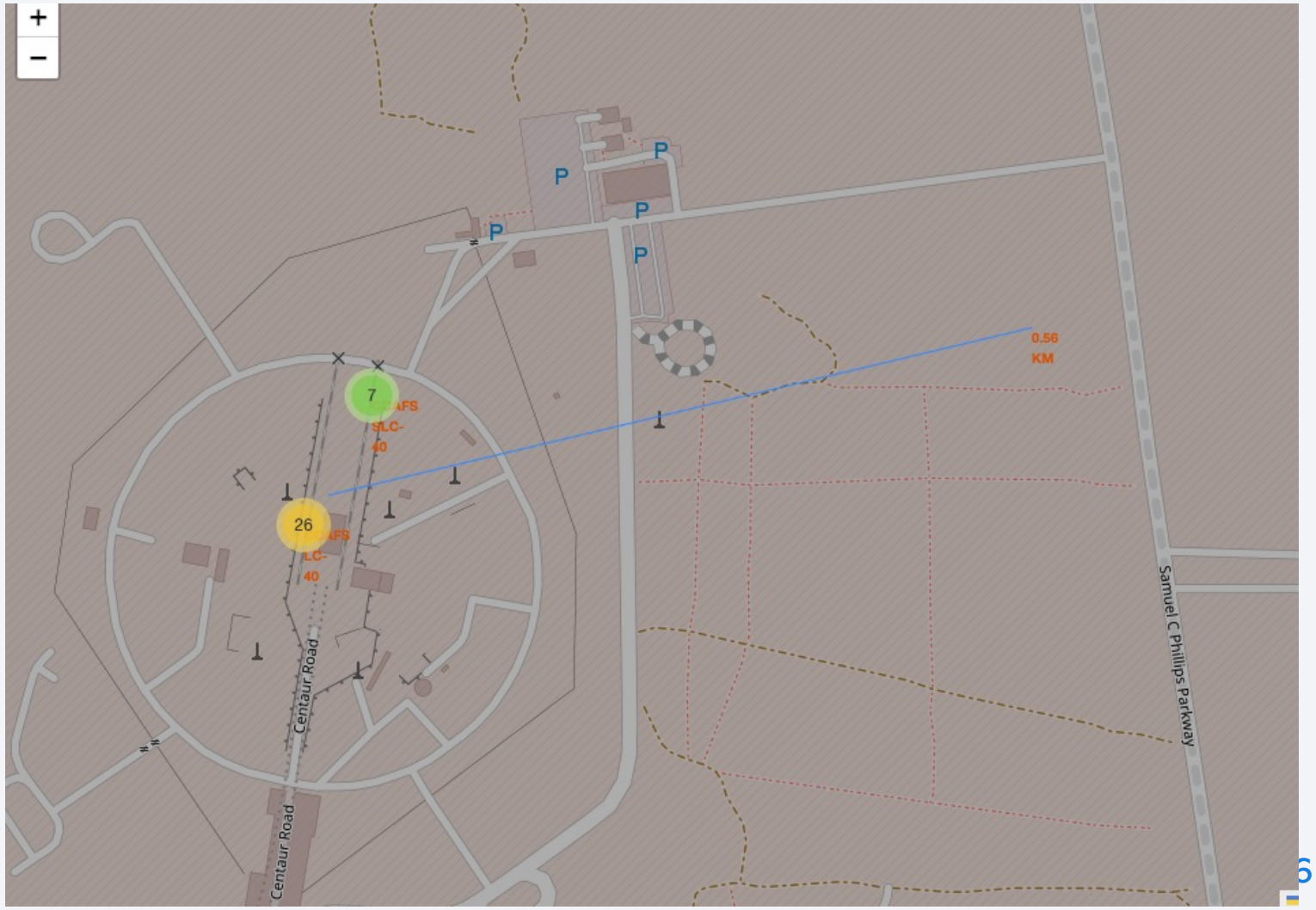
Color Labeled Launch Outcomes



- Green stands for successful launches, red stands for failure

Launch Sites Distance

- Blue line stands for the distance between the launch site and the destination

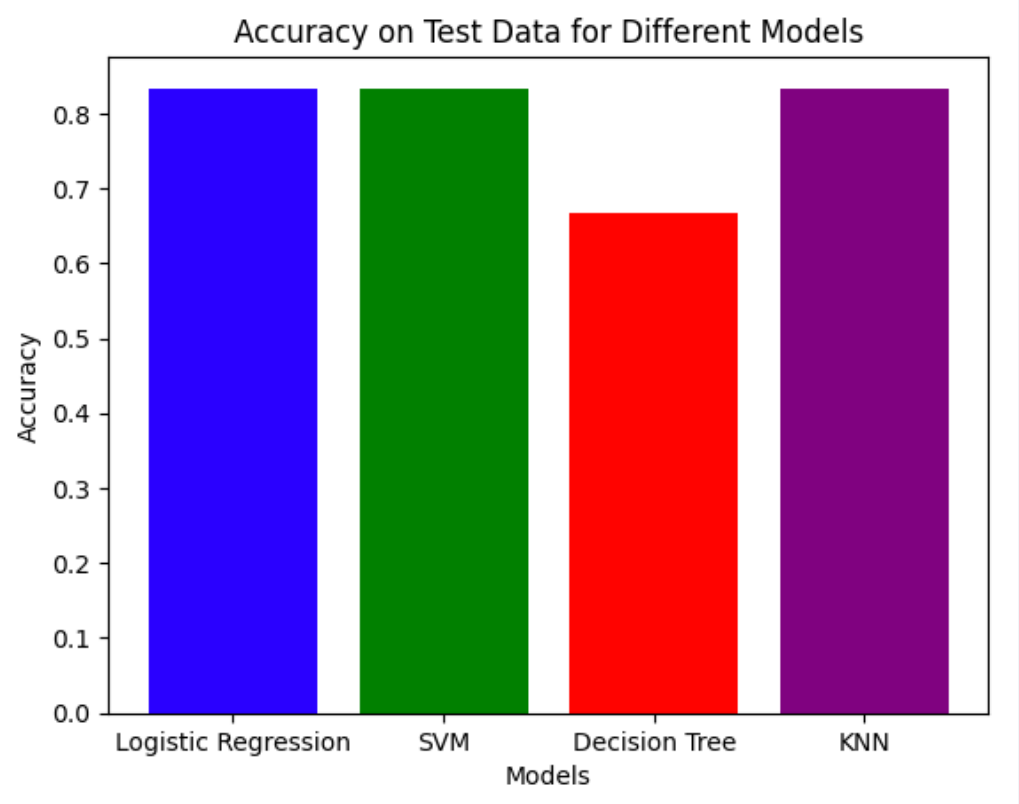


Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Logistic Regression, SVM, and KNN all have 0.8333 accuracy score, which is higher than the Decision Tree model



TASK 12

Find the method performs best:

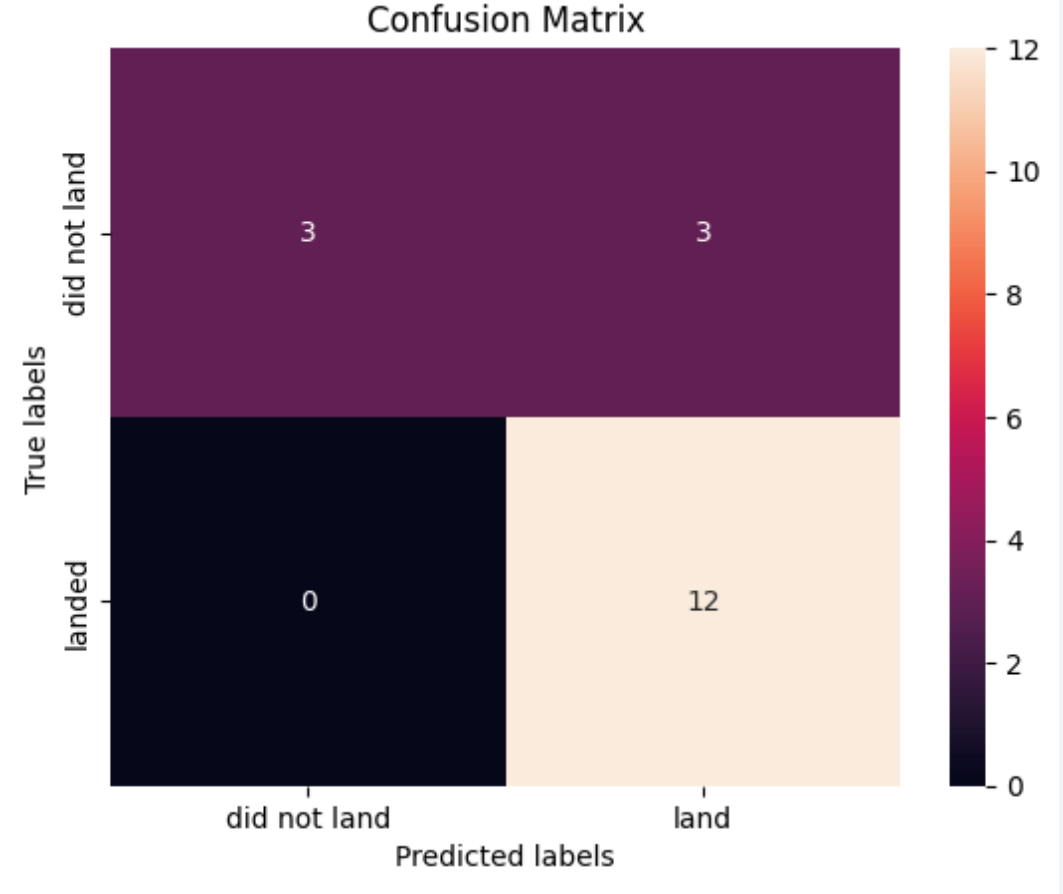
Toggle output scrolling

```
[44]: print("Accuracy on test data for Logistic Regression: ", accuracy_test)
      print("Accuracy on test data for SVM: ", accuracy_svm_test)
      print("Accuracy on test data for Decision Tree: ", accuracy_tree_test)
      print("Accuracy on test data for KNN: ", accuracy_knn_test)
```

```
Accuracy on test data for Logistic Regression:  0.8333333333333334
Accuracy on test data for SVM:  0.8333333333333334
Accuracy on test data for Decision Tree:  0.6666666666666666
Accuracy on test data for KNN:  0.8333333333333334
```

Confusion Matrix

- Logistic Regression, SVM, and KNN all have the same confusion matrix



Conclusions

- The launch successful is increasing, we are working in the right direction!
- From our EDA, we learned that the number of flights had an impact on landing or not
- Orbits with the highest successful rate are ES-L1, GEO, HEO, SSO, and VLEO
- Logistic Regression, Support Vector Machine, and KNN have good performance for prediction. To further decide which model to use, we should check the precision or recall score.
- With the implement of our machine learning model, we get an accuracy score of 83.33% to forecast whether the first-stage launch would land successfully or not

Appendix

- GitHub Link: https://github.com/Chrispy1204/IBM_data_science_capstone

Thank you!