IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Darnell Hernandez
July 6, 2024

Outline

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

Executive Summary

- Summary of methodologies
  - SpaceX Data Collection Using SpaceX Api
  - SpaceX Data Collection with Web Scraping
  - SpaceX Data Wrangling
  - SpaceX Exploratory Data Analysis using SQL
  - SpaceX EDA Datavix Using Python Pandas and Matplotib
  - SpaceX Launch Site Analysis with Folium-Interactive Visual Analytics and Polty Dash
  - SpaceX Machine Learning Landing Prediction
- Summary of all results
  - EDA results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis (Classification)

Introduction

- Project background and context
  - SpaceX advertises Falcon 9 rocket lanches on ites 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 the launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.
- Problems you want to find answers
  - We will predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its website.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- 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

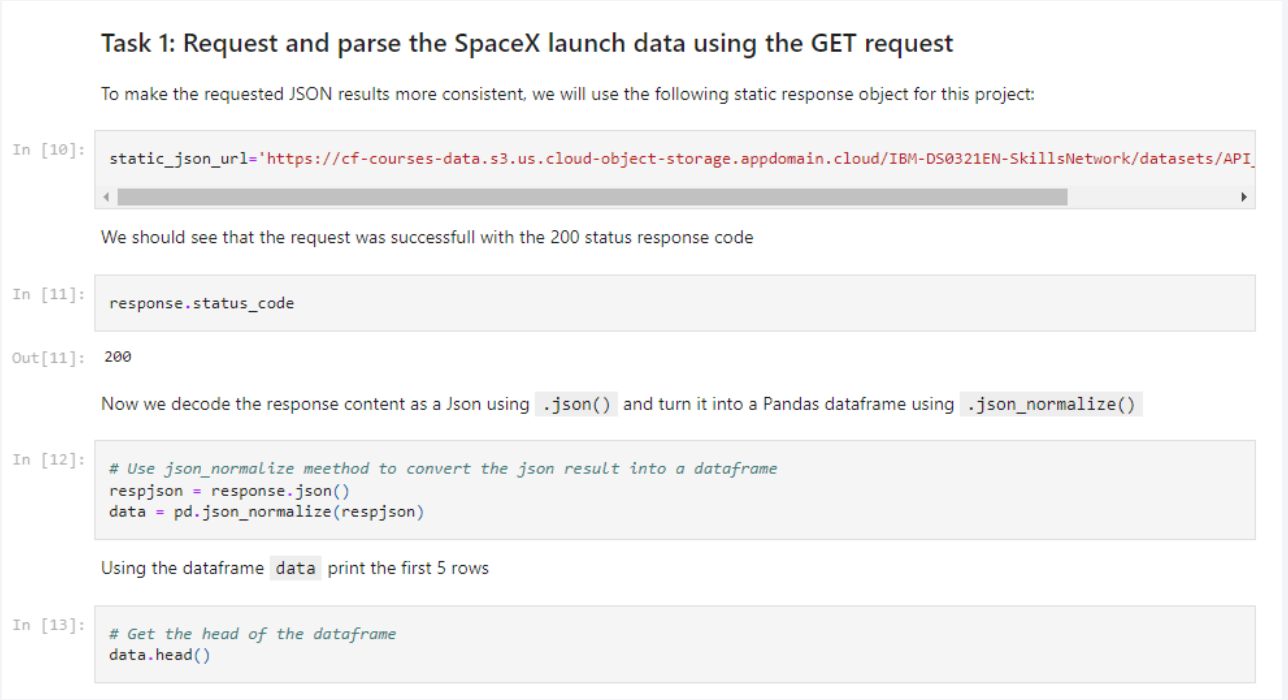
Data Collection

- The Data was collected with an API, specifically the SpaceX REST API. This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
- Another popular way to gather data was through Webscrapping related to Wiki pages.

Data Collection – SpaceX API

- Data collected using SpaceX API
  - We will use this URL (api.spacexdat.com/v4/).
  - This URL is to target a specific endpoint of the API to get past launch data.
  - We will perform a get request using the requests library to obtain the launch data, which we will use to get the dta from the API.
  - This result is then viewed by called the .json() method
  - Our response will be in the form of a JSON, specifically a list of JSON objects.
  - Then we convert JSON to a dataframe through json_normalize function. This function will allow use to "normalize" the structured json data into a flat table.
- Here is the Github URL of the completed SpaceX API.

https://github.com/darnell793/SpaceX-Falcon-9-first-stage-Landing-Prediction-Lab-1-Collecting-the-data/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



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 [10]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_
```

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

In [11]:
```
response.status_code
```

Out[11]: 200

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

In [12]:
```
# Use json_normalize meethod to convert the json result into a dataframe
respjson = response.json()
data = pd.json_normalize(respjson)
```

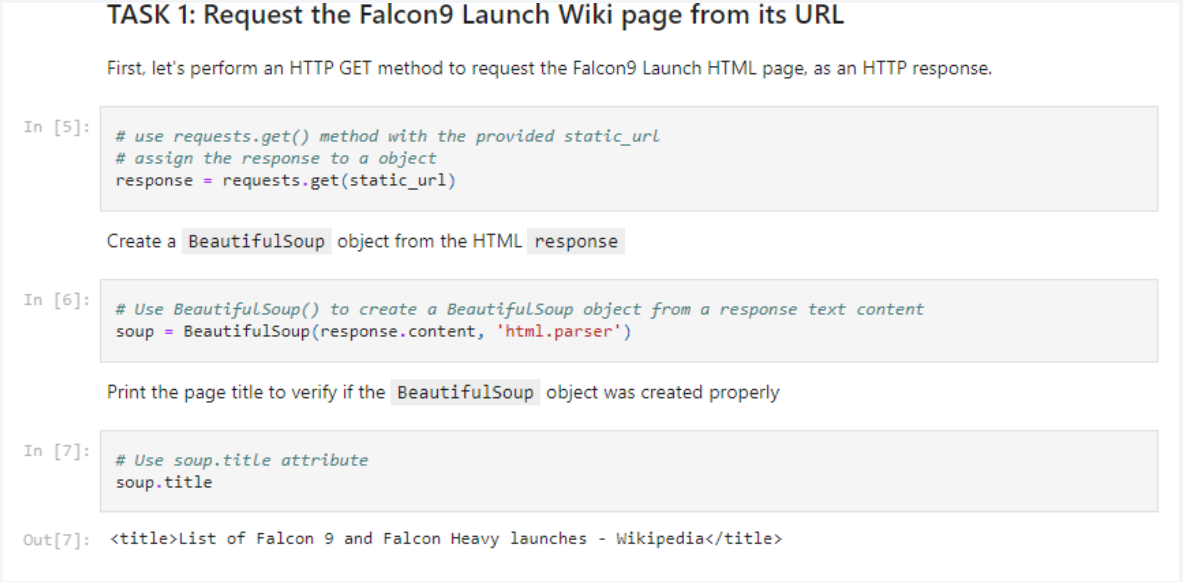
Using the dataframe `data` print the first 5 rows

In [13]:
```
# Get the head of the dataframe
data.head()
```

Data Collection – Webscraping

- Webscrapping to collect Falcon 9 historical launch records from a Wikipedia using BeautifulSoup and request.
- This is to extract the Falcon 9 launch records from the HTML table of the Wikipedia page, then create a dataframe by parsing the launch html.
- Here is the Github URL of the completed SpaceX API.

https://github.com/darnell793/Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/jupyter-labs-webscraping.ipynb



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 [5]:
```
# 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 [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, 'html.parser')
```

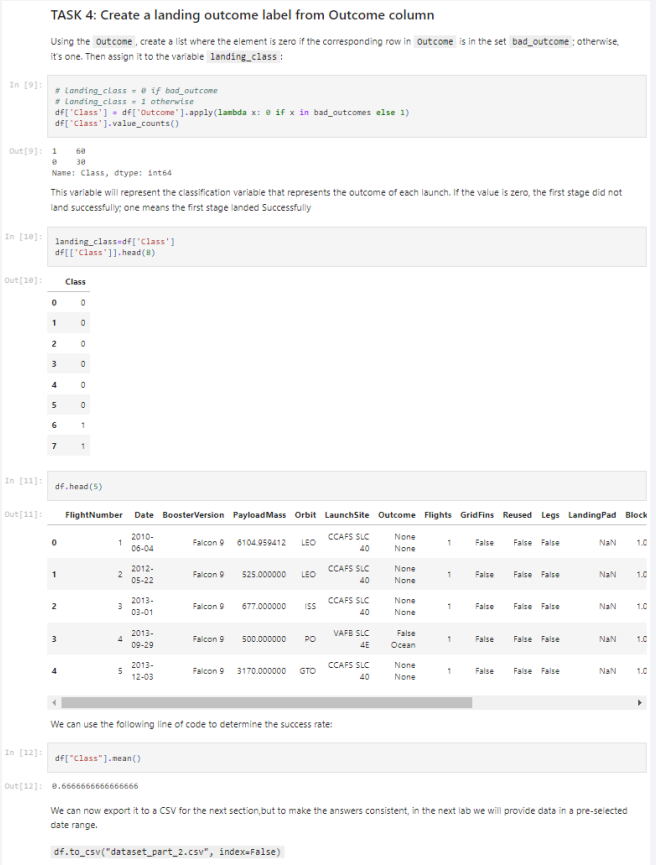
Print the page title to verify if the `BeautifulSoup` object was created properly

In [7]:
```
# Use soup.title attribute
soup.title
```

Out[7]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

Data Wrangling

- We create a Pandas DF from the data we colelcted.
- The data is filtered using the Booster column to only keep the Falcon 9 launches
- Then we resolve the missing data values in the LandingPad and PayloadMass columns. For PayloadMass, missing data values were replaced using mean value of column.



TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

In [9]:
```
# Landing_class = 0 if bad_outcome
# Landing_class = 1 otherwise
df['Class'] = df['Outcome'].apply(lambda x: 0 if x in bad_outcomes else 1)
df['Class'].value_counts()
```

Out[9]:
```
1    60
0    30
Name: Class, dtype: int64
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

In [10]:
```
landing_class=df['Class']
df[['Class']].head(8)
```

Out[10]:

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

In [11]:
```
df.head(5)
```

Out[11]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0

We can use the following line of code to determine the success rate:

In [12]:
```
df["Class"].mean()
```

Out[12]:
```
0.6666666666666666
```

We can now export it to a CSV for the next section,but to make the answers consistent, in the next lab we will provide data in a pre-selected date range.

```
df.to_csv("dataset_part_2.csv", index=False)
```

- Here is the Github URL of the completed SpaceX API.

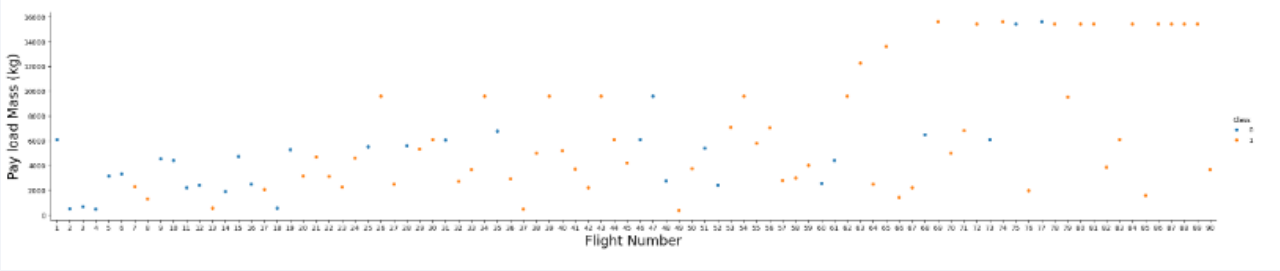
https://github.com/darnell793/Space-X-Falcon-9-First-Stage-Landing-Prediction-Lab-2-Data-wrangling

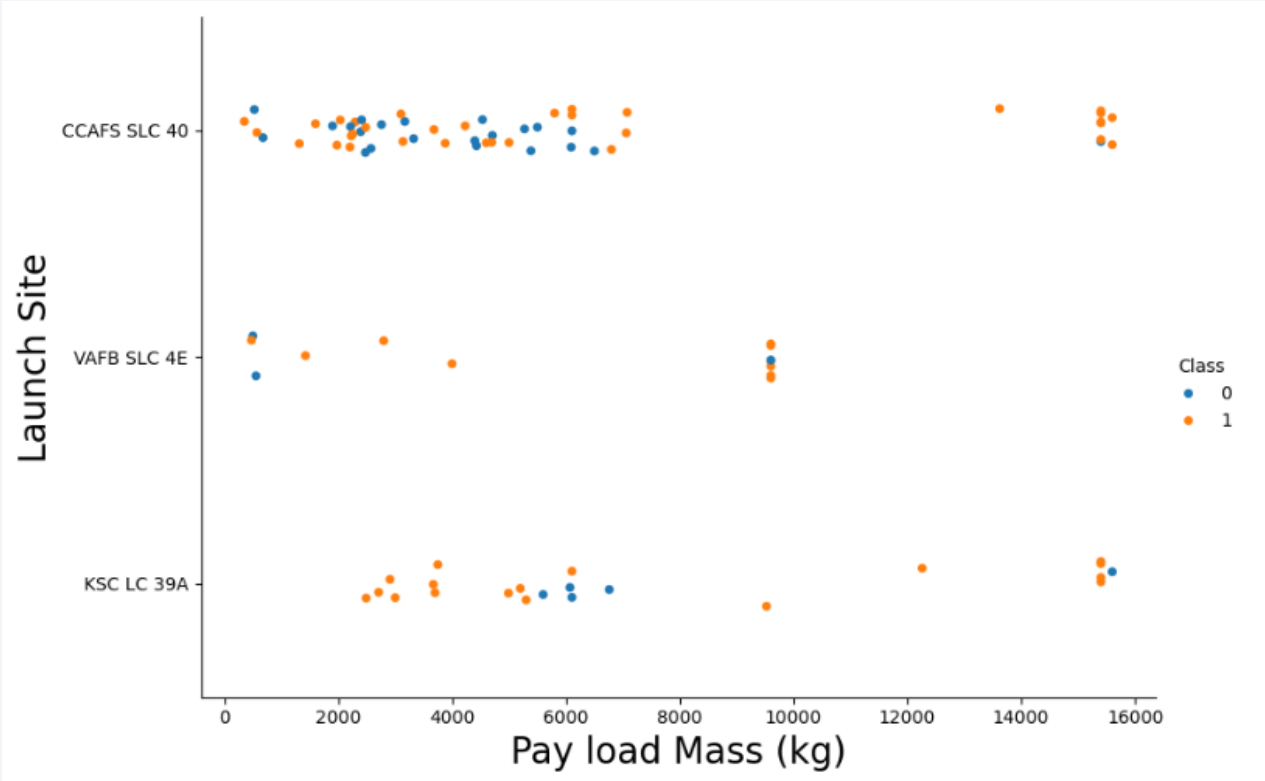
EDA with Data Visualization

- Perfomed Data Analysis and Feature Engineering
- We used Scatterplot, Bar Chart and Line Plot.
  - Scatterplot was to visualize the relationship between Flight Number and Launch Site and Payload and Launch Site
  - Bar charts to visualize the relationship between Success Rate of each Orbit
  - Line plot to visualize the relationship between Flight Number and Orbit and Payload Mass and Orbit
- Here is the Github URL for EDA with Data Visualization

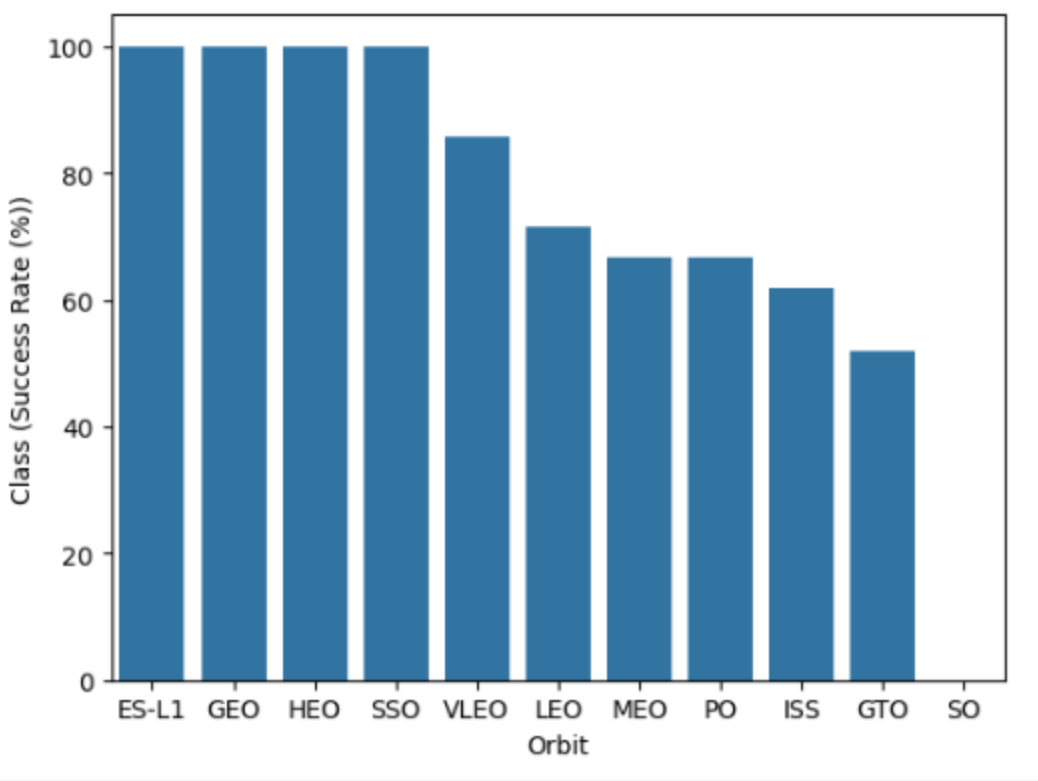
https://github.com/darnell793/SpaceX-Falcon-9-First-Stage-Landing-Prediction-Assignment-Exploring-and-Preparing-Data/blob/main/edadataviz%20(3).ipynb

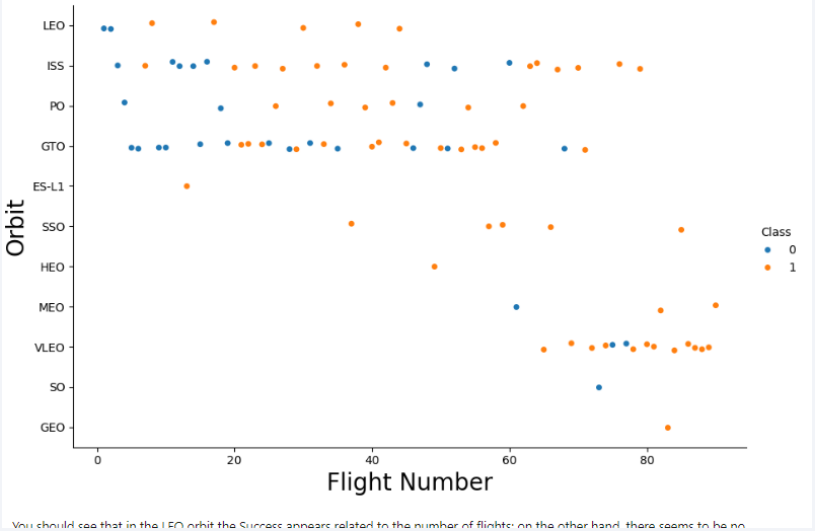
EDA with Data Visualization



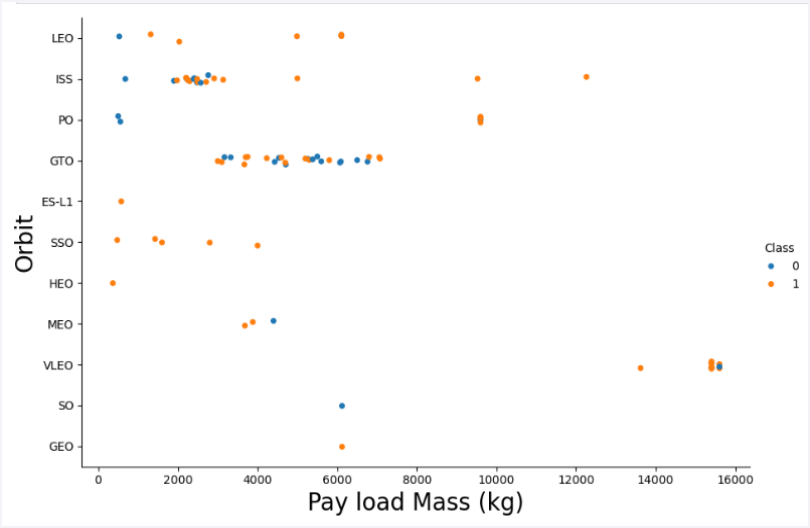


EDA with Data Visualization





You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no



EDA with SQL

- **SQL queries were performed for EDA**
  - Display the names of unqiue launch sites in the space mission
    - **%sql** SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
  - Display 5 records where launch sites begin with the string 'CCA'
    - **%sql** SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
  - Display the total payload mass carried by boosters launched by NASA (CRS)
    - **%sql** SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
  - Display average payload mass carried by booster version F9 v1.1
    - **%sql** SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
    -

EDA with SQL

- List the data when the first successful landing outcome in ground pad was achieved
  - **%sql** SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - **%sql** SELECT * FROM 'SPACEXTB
- List the total number of successful and failure mission outcomes
  - **%sql** SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
- List the names of the booster_versions which have carried the maximum payload mass
  - **%sql** SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL);

EDA with SQL

- List the records which will display the month names, failure of landing, outcomes in drone ship, booster versions, launch_site for the months in year 2015
  - **%sql** SELECT substr(Date,7,4), substr(Date, 4, 2),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing _Outcome" FROM SPACEXTBL WHERE substr(Date,7,4)='2015' AND "Landing _Outcome" = 'Failure (drone ship)';
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 annd 2017-03-20 in descending order.
  - **%sql** SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDER BY Date DESC;

- Here is the Github URL of EDA with SQL

https://github.com/darnell793/-Complete-the-EDA-with-SQL/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We created a folium map to mark all the launch sites.
- Then we created map objects such as folium circles, folium markers, and marker cluster
- Additionally, we create lines to mark success or failure of launches for each launch site
- Here is the Github URL for Interactive Map with Folium

https://github.com/darnell793/Space-X-Launch-Sites-Locations-Analysis-with-Folium-Interactive-Visual-Analytics

Build a Dashboard with Plotly Dash

- We built an interactive dashboard application with Plotly Dash
- Here is the Github URL for Dashboard with Plotly Dash
- https://github.com/darnell793/Module-10-SpaceX-Build-an-Interactive-Dashboard-with-Ploty-Dash

Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
- We loaded the data into a Pandas dataframe, then we used Exploratory Data Analysis and determined Training Labels
    - We first created NumPy array from the column Class in data, by applying the method to_numpy(), then assigned it to the variable Y
    - Next, we standardize the data in X then reassign it to the variable X using the transform StandardScaler method
    - After, we use the function train_test_split to split the data X and Y into training and test data. We set the parameter test_sdize to 0.2 and random_state to 2.

Predictive Analysis (Classification)

- Our next step was to find the best Machine Learning Model that would perform best using the test data between Logistic Regression, SVM, Classification Trees and K-nearest Neighbors
  - First, we created an object for each algorithm to be used
  - Next, we created GridSearchCV object and gave them parameters
  - Each model under review, we set the GridSearchCV object was created with a cv=10.
  - Then we fir the object to find the best parameters
  - After fitting the training set, we output GridSearchCV object for each model and display the best parameters using the data attribute best_params_ and the accuracy on the validation data using the data attribute best_score_
  - Finally, we use the method score to calculate the accuracy on the test data for each model and plotted a Confusion Matrix for each using the test and predicted outcomes

Predictive Analysis (Classification)

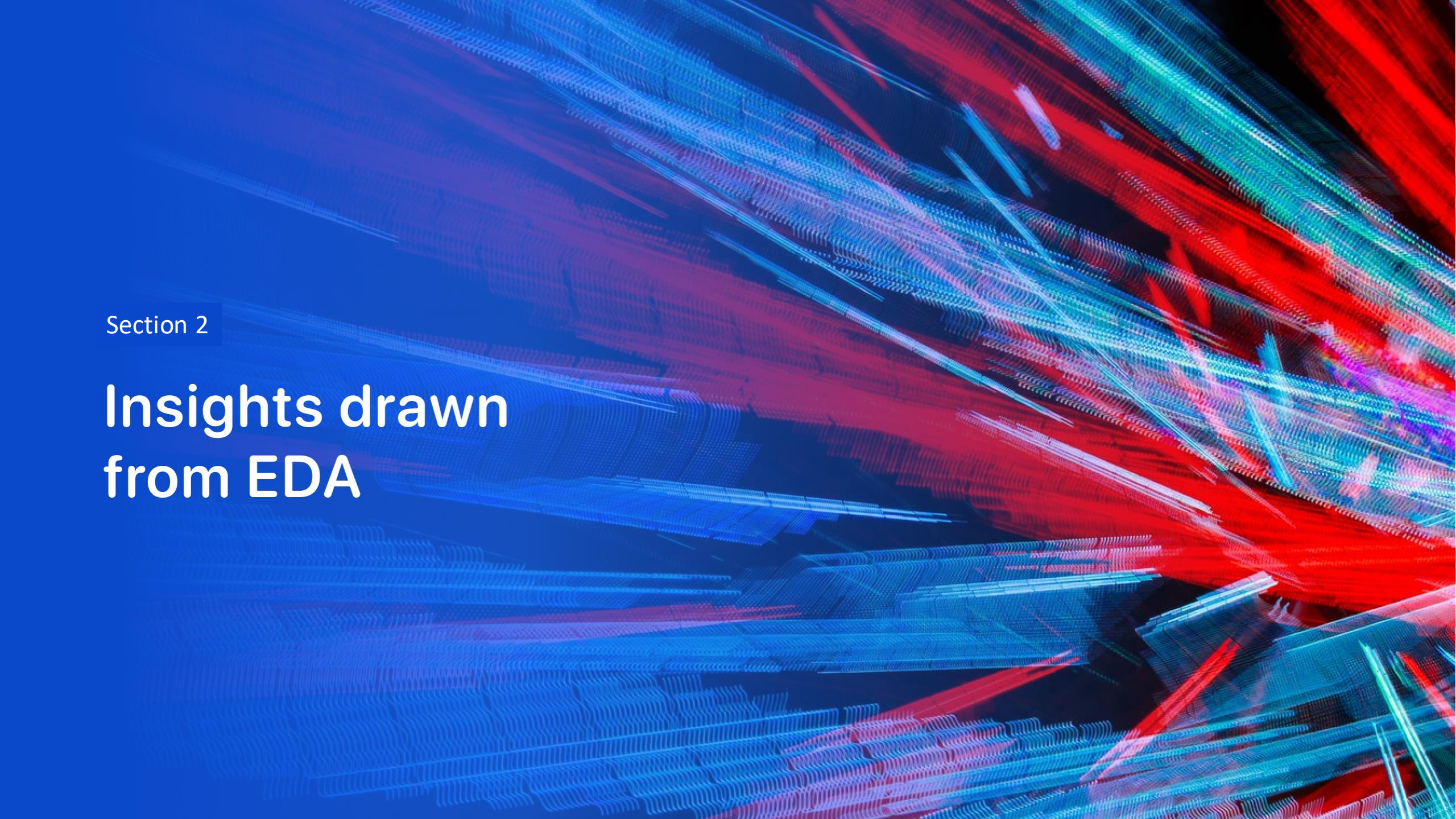
- Our results are a table that shows the test accuracy score for each method comparing them.

Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.944444
KNN	0.833333

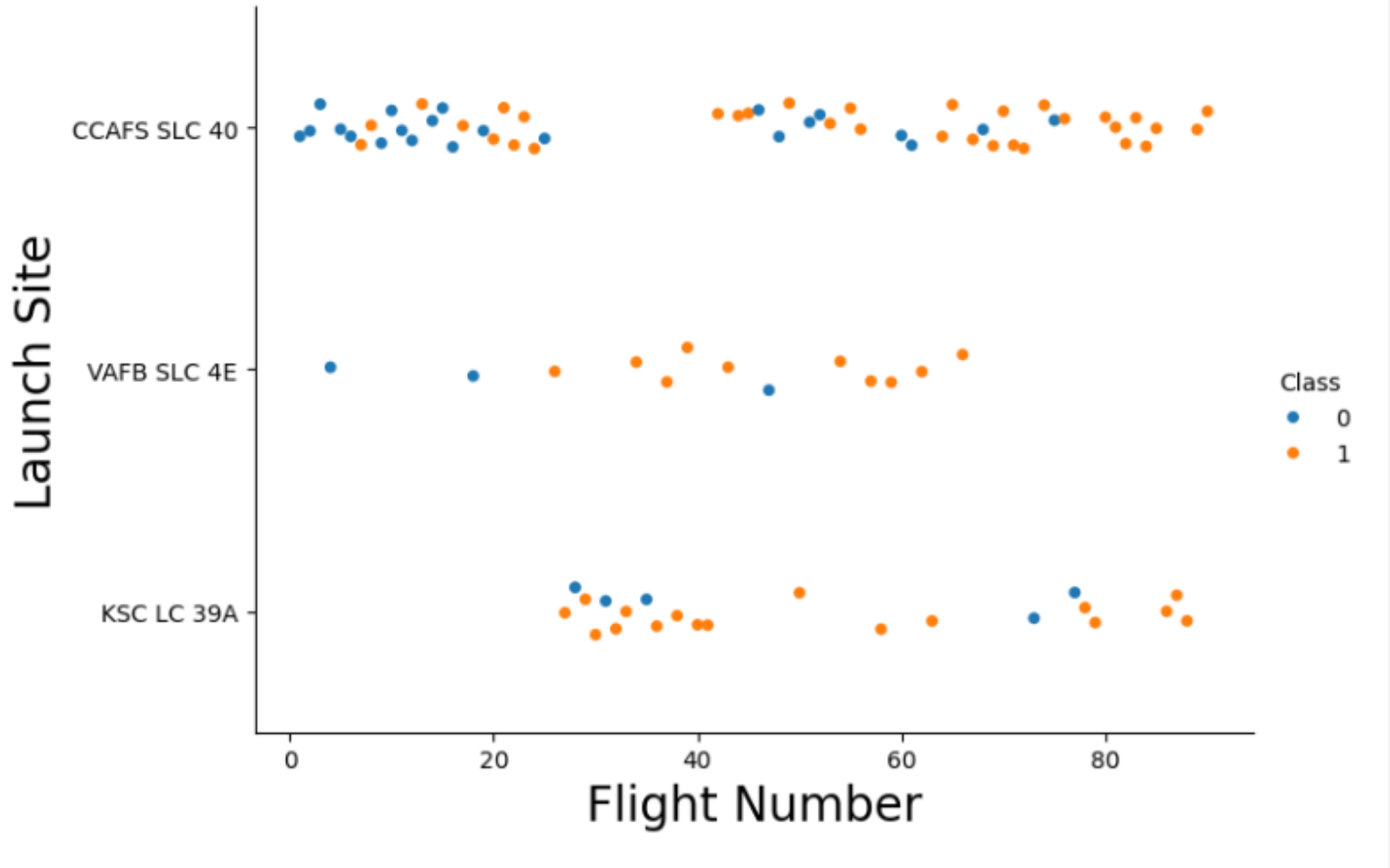
- Here is the Github URL for Predictive Analysis
- https://github.com/darnell793/Space-X-Falcon-9-First-Stage-Landing-Prediction-Machine-Learning-Prediction-/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5%20(1).ipynb

Results

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


Section 2
Insights drawn from EDA

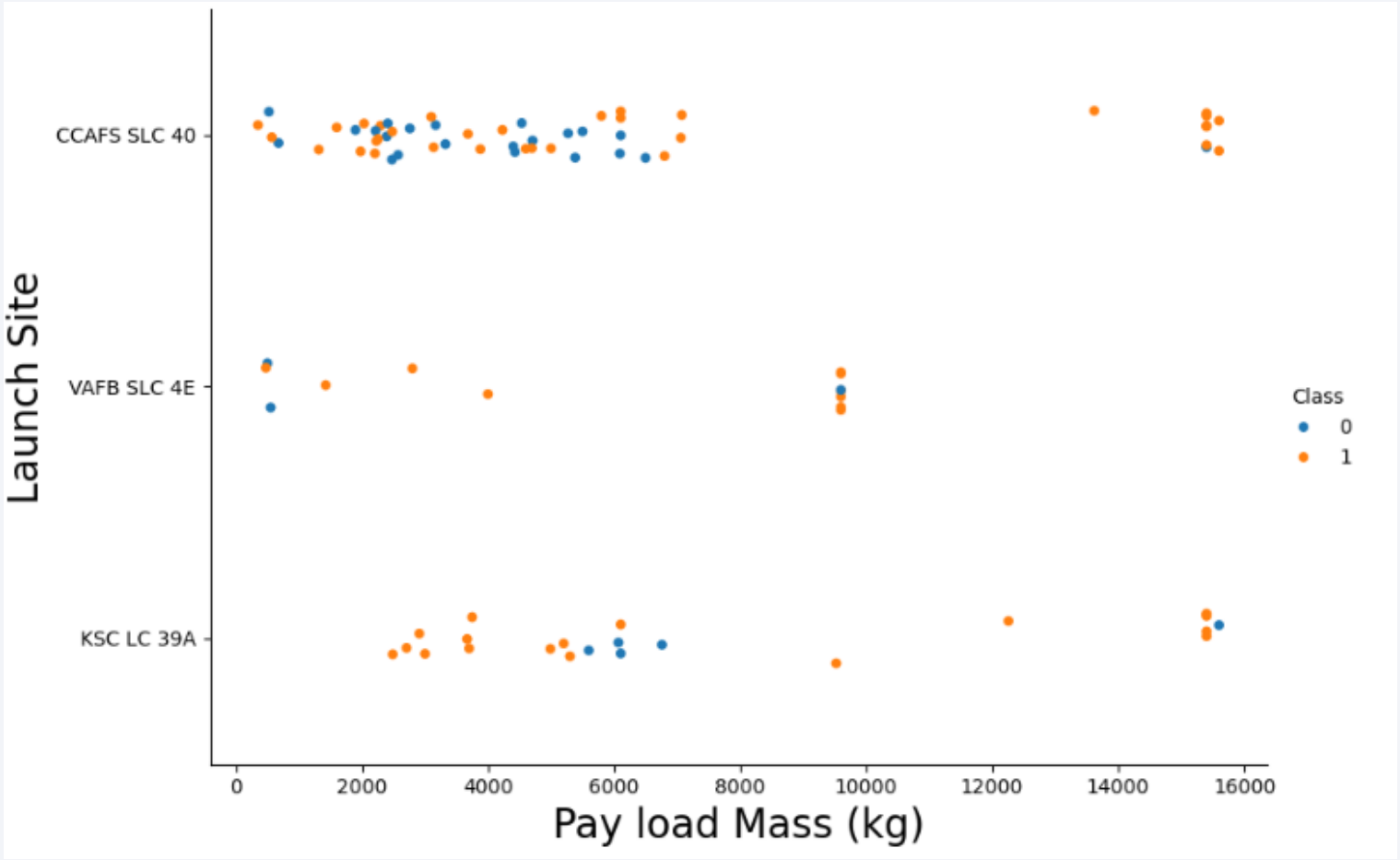
Flight Number vs. Launch Site



From the plot, we can determine as the flight number is directly proportional to the number of success rate.

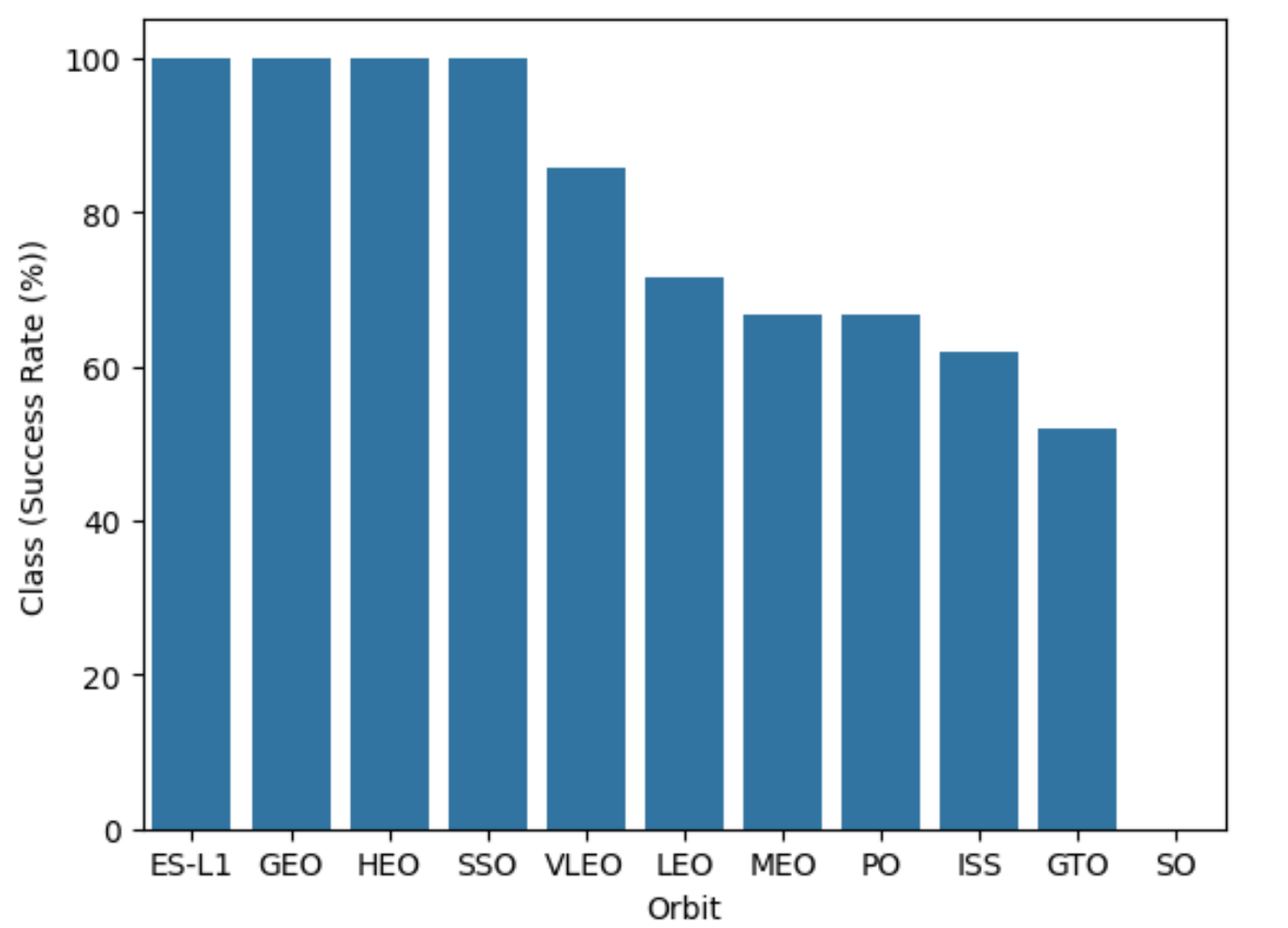
For example, we can see VAFB SLC 4E is 100% after the Flight number is over 50.

Payload vs. Launch Site



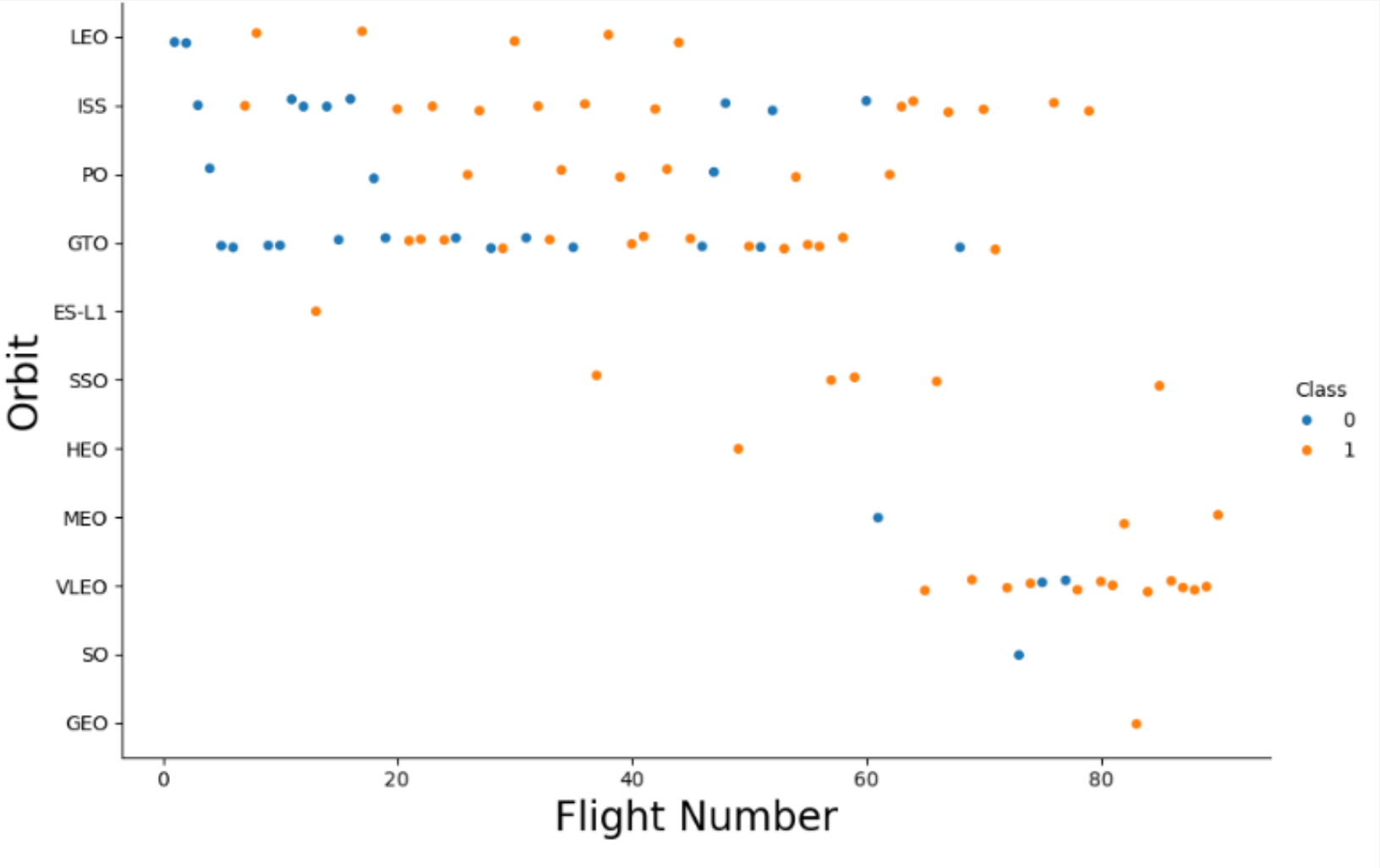
From the plot, we can see that the VAFB-SLC Launch Site there are no rockets launched for Pay Load Mass (greater than 10000).

Success Rate vs. Orbit Type



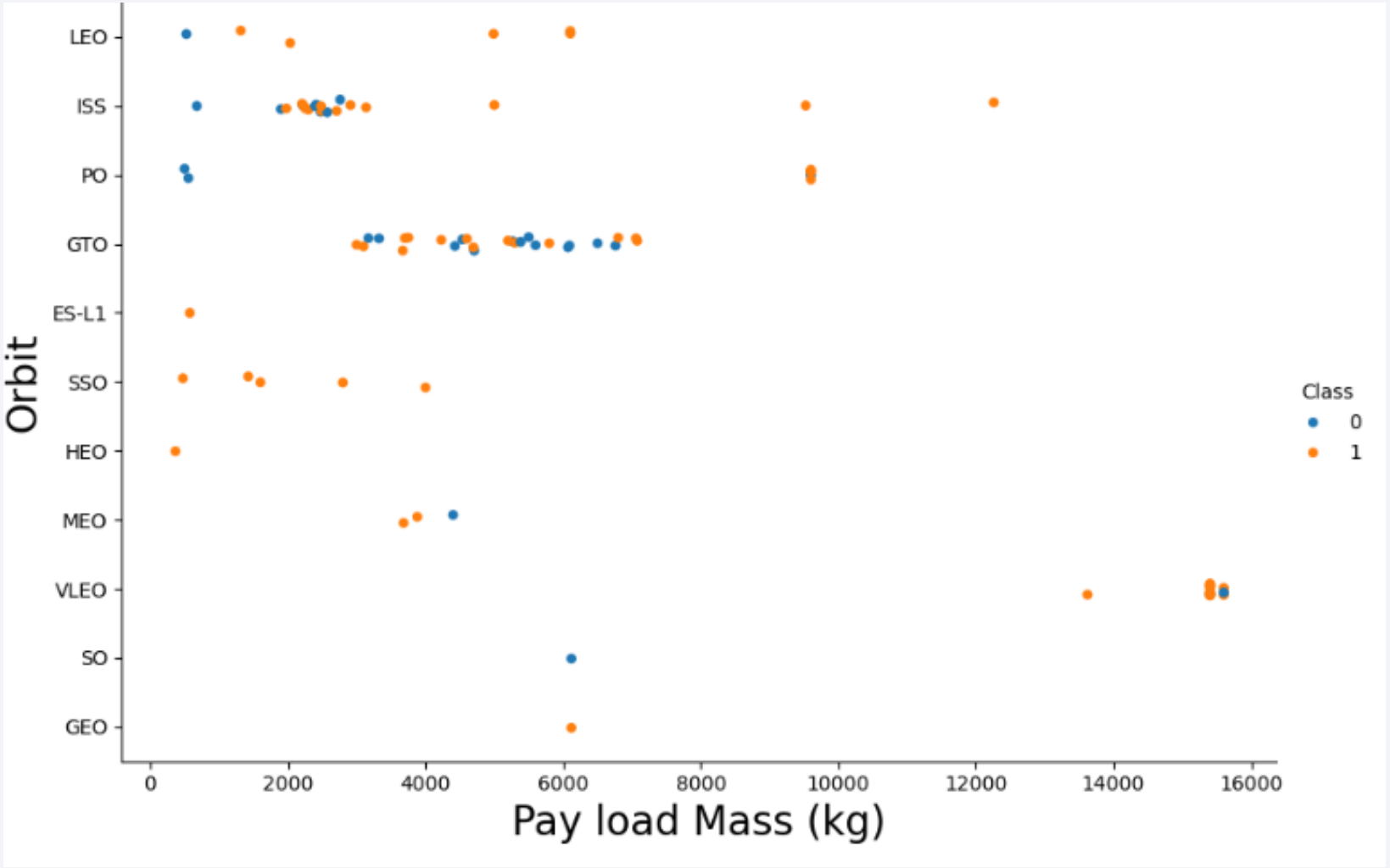
- From the bar plot, we see that the ES-LI, GEO, HEO and SSO have the highest success rate at 100%, while SO has the lowest success rate at 0%.

Flight Number vs. Orbit Type



- A scatter plot between the orbits and flight numbers.
- We can see that more flight numbers leads to more success.
- Although, GTO does not have that relationship with flight numbers

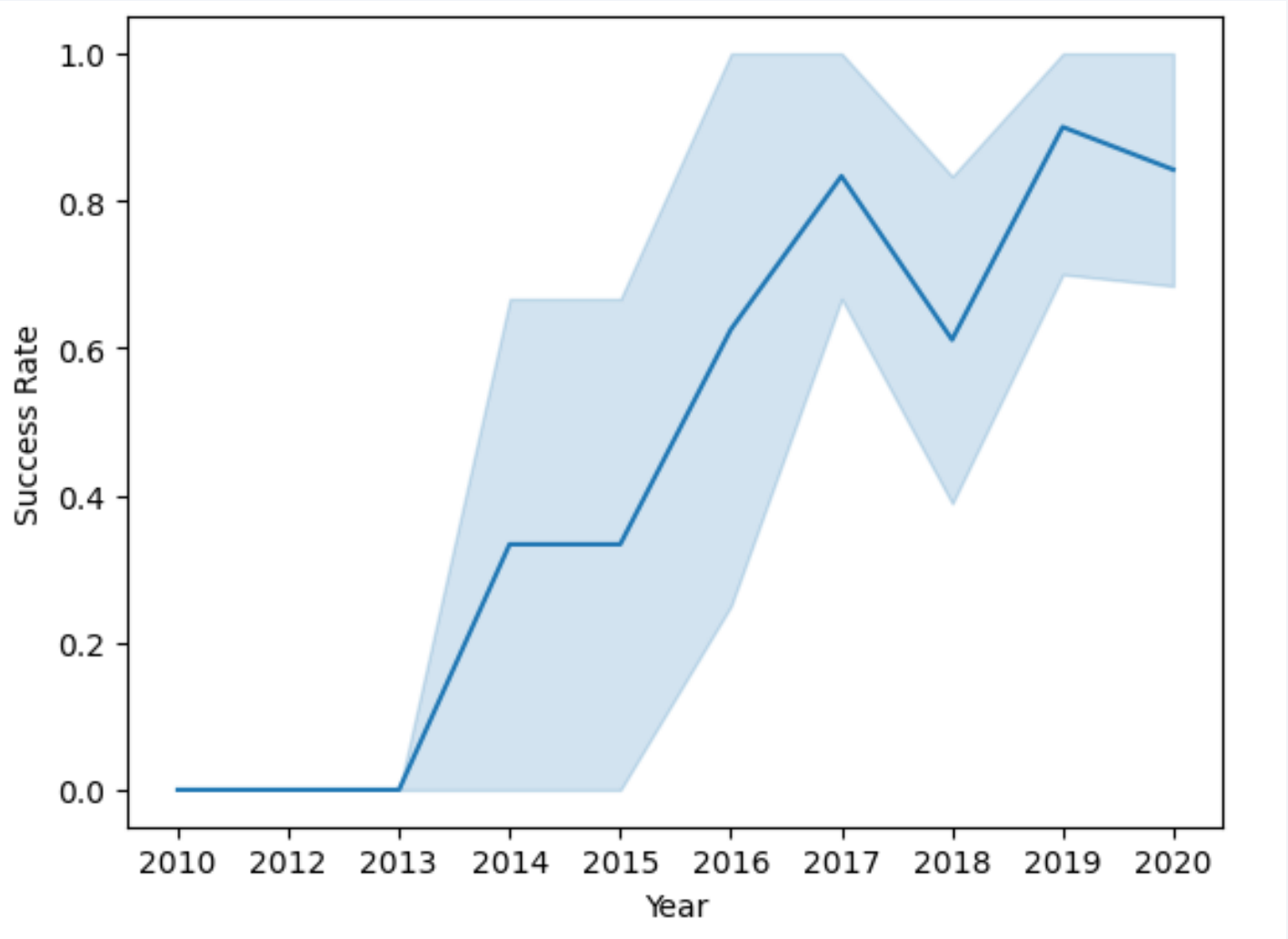
Payload vs. Orbit Type



From the plot, we see that ISS, PO and SO have more successful landings or positive landings.

Although, GTO is still can not determine if there is a relationship.

Launch Success Yearly Trend



From the plot, we see that starting from 2013, the success rate has increased.

All Launch Site Names

Finds the names of the unique launch sites

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

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

Launch Site Names Begin with 'CCA'

Finds 5 records where launch sites begin with `CCA`

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

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

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

The total payload carried by boosters from NASA

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

Average Payload Mass by F9 v1.1

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

Task 4

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

```
G(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

First Successful Ground Landing Date

The dates of the first successful landing outcome on ground pad

```
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";
```

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

MIN(DATE)
None

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

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

```
%sql SELECT * FROM 'SPACEXTB
```

```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) unrecognized token: "'SPACEXTB"
[SQL: SELECT * FROM 'SPACEXTB]
(Background on this error at: http://sqlalche.me/e/e3q8)
```

Total Number of Successful and Failure Mission Outcomes

The total number of successful and failure mission outcomes

```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

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

Boosters Carried Maximum Payload

List the names of the booster which have carried the maximum payload mass

- `%sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL);`

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

```
%sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_
```

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

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

2015 Launch Records

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
%sql SELECT substr(Date,7,4), substr(Date, 4, 2),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Ou
```

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

substr(Date,7,4)	substr(Date, 4, 2)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	"Landing_Outcome"

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

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

```
%sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDE
```

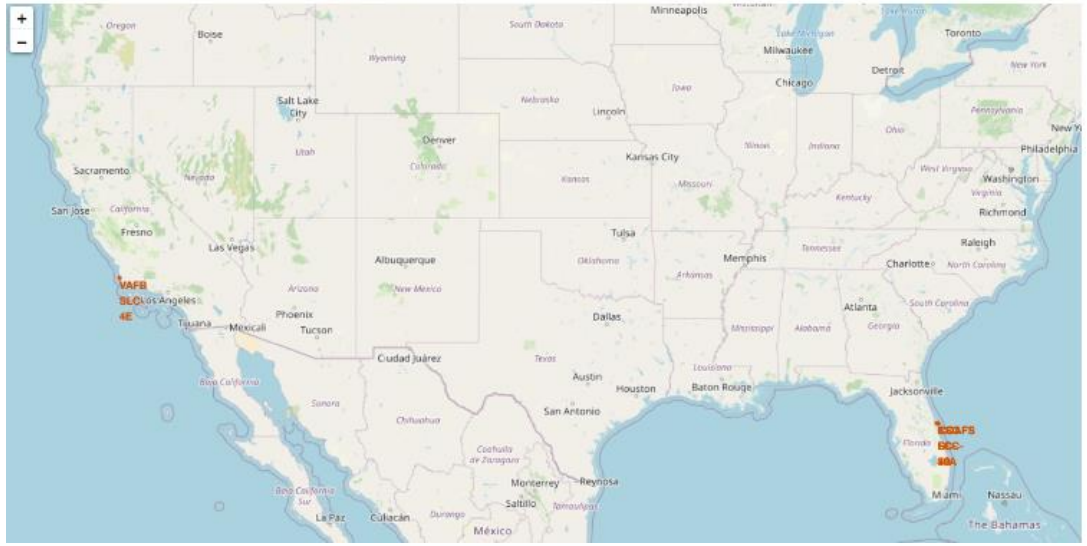
```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome

Section 3
Launch Sites
Proximities Analysis

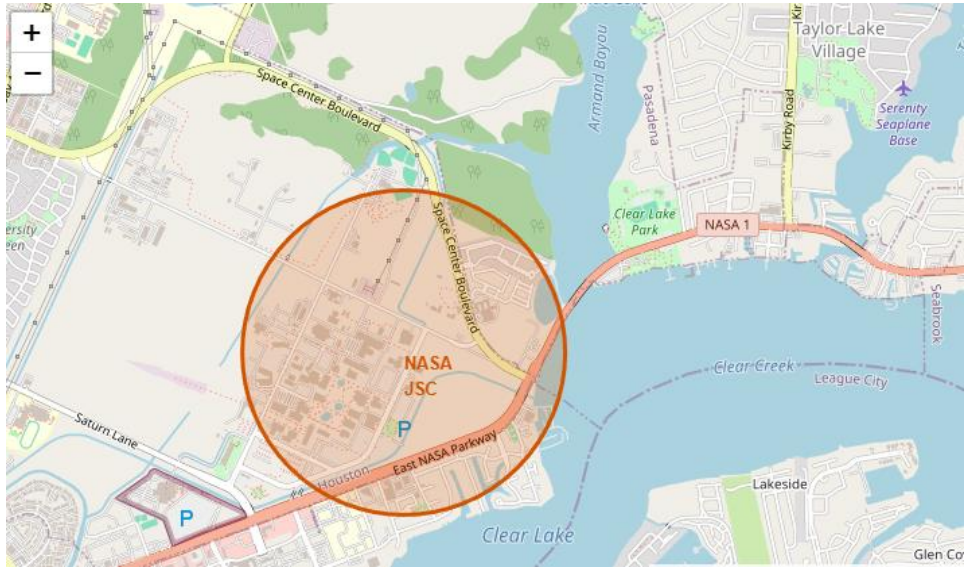
Launch site on Global Map

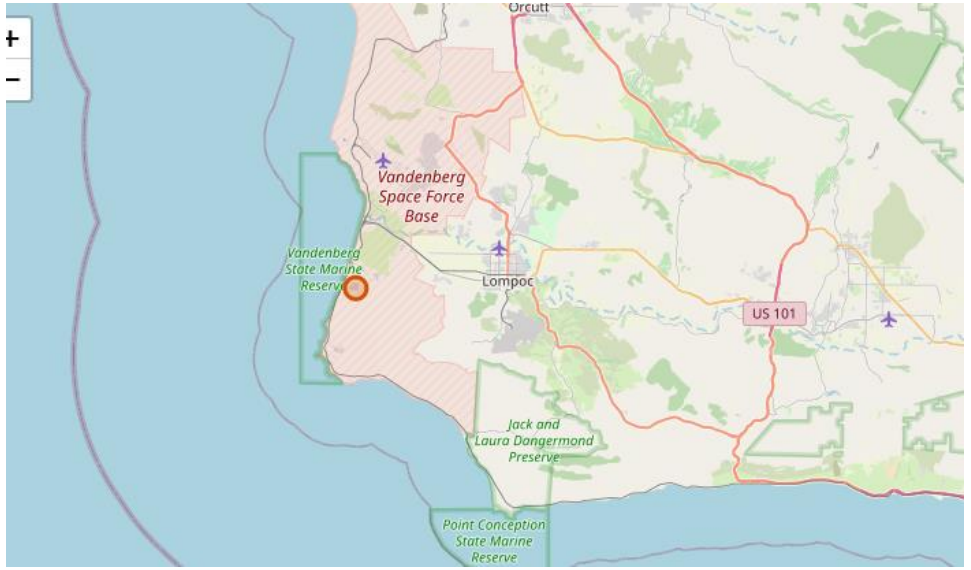
- All launch sites are in proximity to the Equator and near the coast.



Launch site on Map

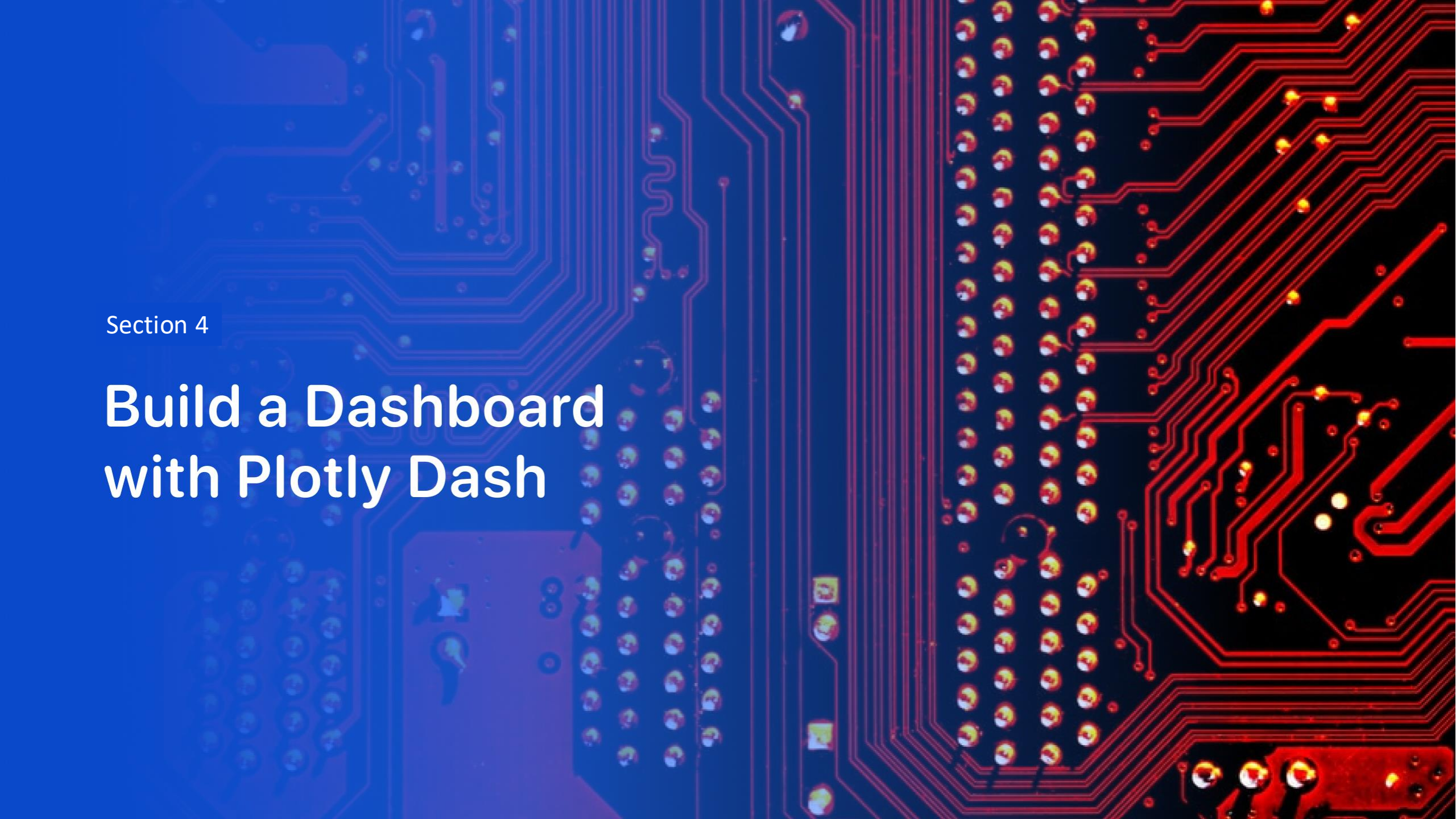
- Closer look into launch sites





Launch outcomes for Each site with Colour Markers

- All launch sites are in proximity to the Equator and near the coast.


Section 4
Build a Dashboard with Plotly Dash

<Dashboard Screenshot 1>

- Replace <Dashboard screenshot 1> title with an appropriate title
- Show the screenshot of launch success count for all sites, in a piechart
- Explain the important elements and findings on the screenshot

<Dashboard Screenshot 2>

- Replace <Dashboard screenshot 2> title with an appropriate title
- Show the screenshot of the piechart for the launch site with highest launch success ratio
- Explain the important elements and findings on the screenshot

<Dashboard Screenshot 3>

- Replace <Dashboard screenshot 3> title with an appropriate title
- Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider
- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.

Section 5

Predictive Analysis (Classification)

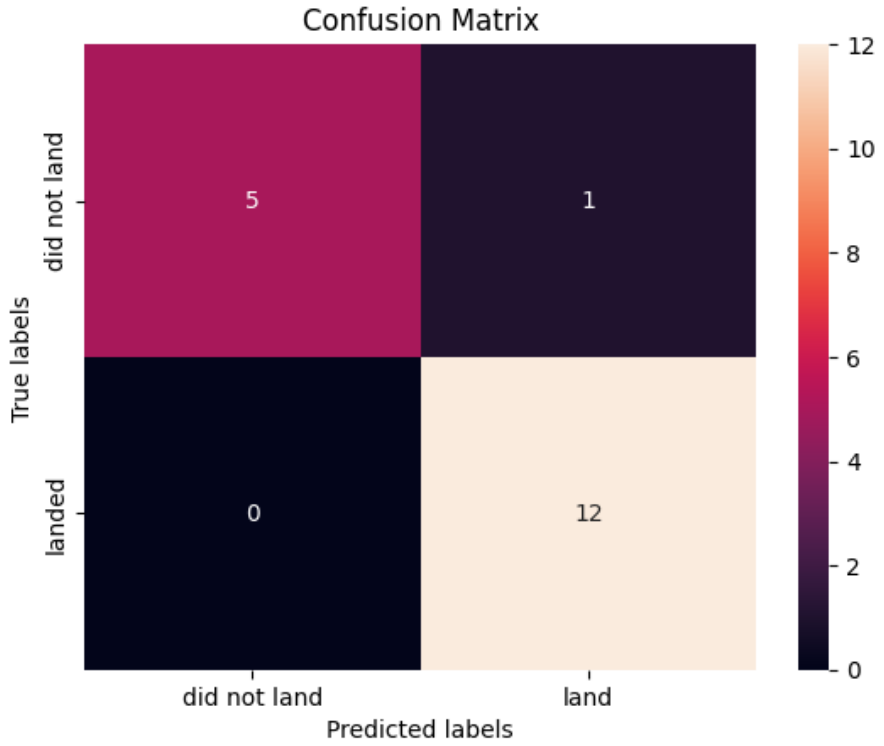
Classification Accuracy

Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.944444
KNN	0.833333

- Decision Tree had the highest Test Data Accuracy with 0.944444

Confusion Matrix

- This is the confusion matrix for decision trees. We can see that only have 1 false positive, while the rest are true positives for did not land and land.



Conclusions

- Our conclusion is that different sites have different success rates. We can see that the number of flights is directly proportional to the number of success rates. We can see that VAFB SCL 4E launch site is 100% after the flight 50, while both KSC LC and CCAFS SLC 40 have a 100% success rate after 80th flight. Additionally, we can see that heavy payload masses greater than 10000, there are no rockets for VAFB-SLC.
- Orbits ES-L1, GEO, HEO and SSO have the highest success rates at 100%, while SO has a 0 % success rate. Additionally, the success rate for LEO is seen to be related to the number of flights, while GTO has no relationship with it. For heavy payloads, ISS, PO and SO have the more successful landings or positive landings, while we can not make any meaningful relationship for GTO for positive and negative landings.
- Finally, our success rate has been increasing by year starting from 2013

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!