IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Johnny Zeng>
<18 December 2023>

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 DataViz Using Python Pandas and Matplotlib
  - SpaceX Launch Sites Analysis with Folium-Interactive Visual Analytics and Plotly Dash
  - SpaceX Machine Learning Landing Prediction
- Summary of all results
  - EDA results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis

Introduction

- Project background and context

  SpaceX states that Flacon 9 rocket launches costs about 62 million dollars while other providers costs up to 165 million dollars each, this is because SpaceX can reuse the first stage and saves a lot of money. If it can be determined if the first stage will land, the cost of a launch can be calculated. This information can be used if another company wants to bid against SpaceX for a launch.

- Problems you want to find answers

  This capstone project will predict if the Falcon 9 first stage will successfully land based on 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

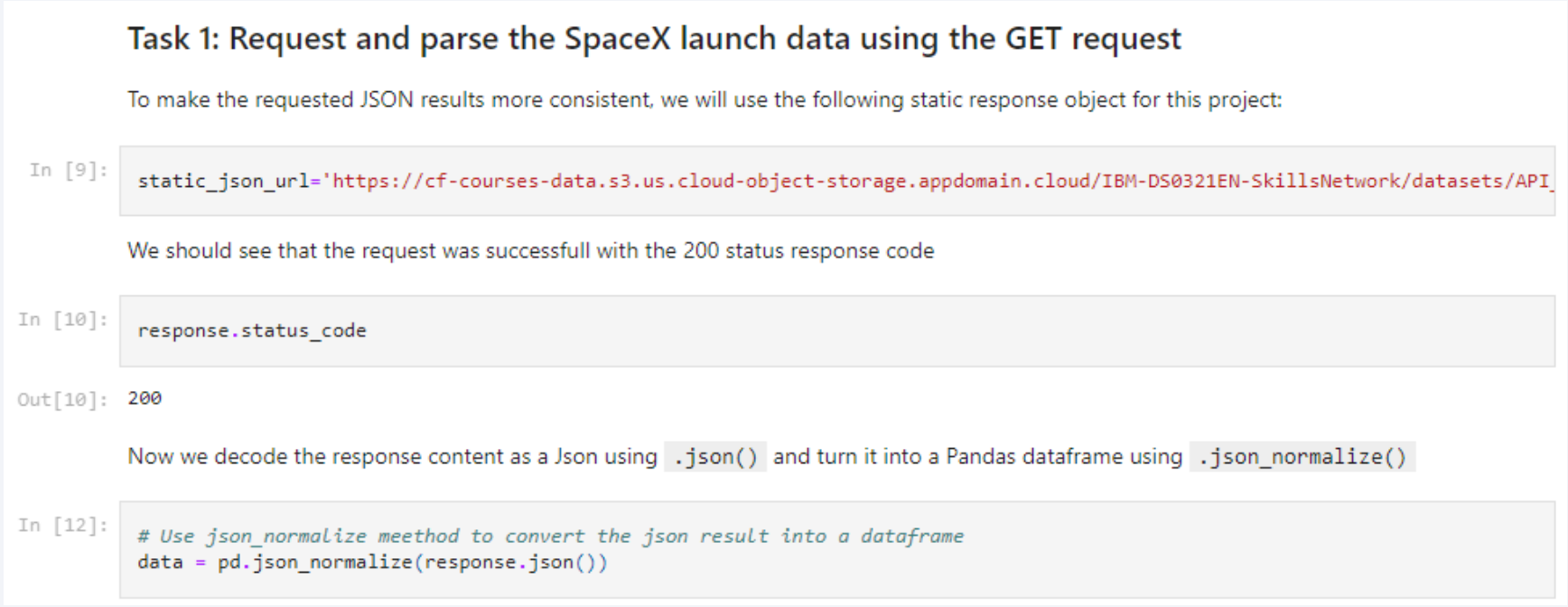
Data Collection

- How the data sets were collected

  Data is first collected using SpaceX API by making a get request.

  The SpaceX launch data was requested and parsed using the GET request and the JSON was converted into a Pandas data frame.

Data Collection – SpaceX API



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/datasets/API_
```

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

- Data was collecting using the SpaceX API by making a get request then decoding the response content and turning it into a Pandas dataframe.
- This is the following Github URL of the complete code:

https://github.com/johnny-zeng-analysis/Coursera-Python-Capstone-Project/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

- Web scraped to collect Falcon 9 launch record from Wikipedia using BeautifulSoup and request to extract records.
- This is the following Github URL of the complete code:

https://github.com/johnny-zeng-analysis/Coursera-Python-Capstone-Project/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.

```
# 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`

```
# 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

```
# Use soup.title attribute
soup.title
```

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

Data Wrangling

- After creating a Pandas data frame from the data, it was filtered, missing data values were replaced with mean value of the column, and an outcome column was created.
- This is the following Github URL of the complete code:

https://github.com/johnny-zeng-analysis/Coursera-Python-Capstone-Project/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

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`:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise

# Create a set of outcomes where the second stage did not land successfully
bad_outcomes = set(landing_outcomes.keys()[[1, 3, 5, 6, 7]])

# Create a new column 'landing_class' based on conditions
df['landing_class'] = np.where(df['Outcome'].isin(bad_outcomes), 0, 1)
```

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

```
#df['Class']=landing_class
#df[['Class']].head(8)

df[['Outcome', 'landing_class']].head(8)
```

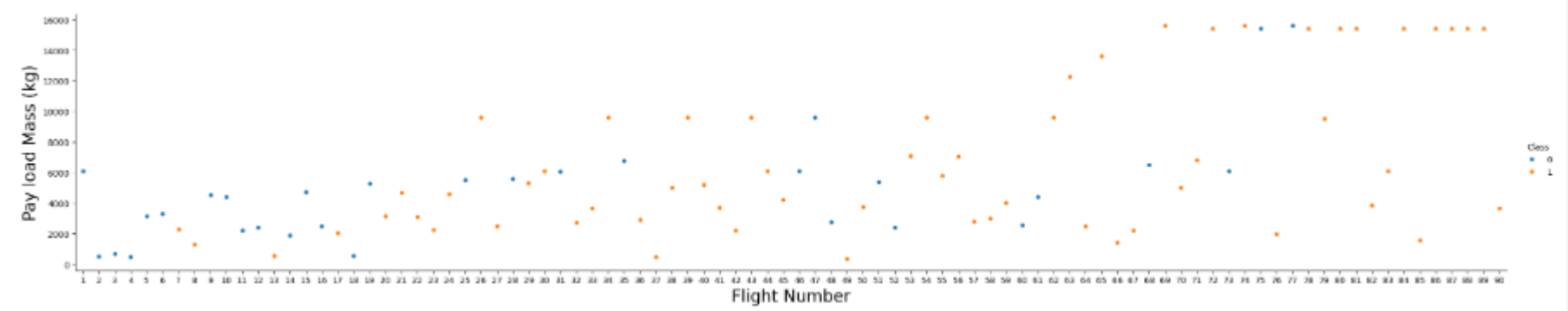
	Outcome	landing_class
0	None None	0
1	None None	0
2	None None	0
3	False Ocean	0
4	None None	0
5	None None	0
6	True Ocean	1
7	True Ocean	1

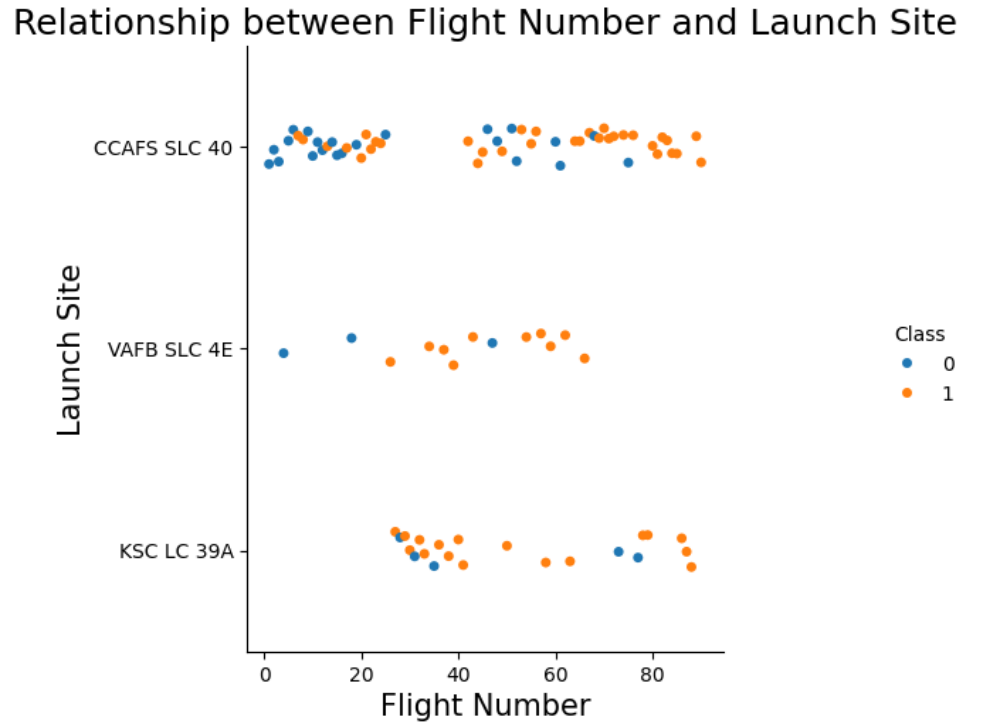
EDA with Data Visualization

- Data analysis was performed using Pandas and Matplotlib
- Scatter plots, bar charts, and line charts were used to visualize the data
- This is the following Github URL of the complete code:

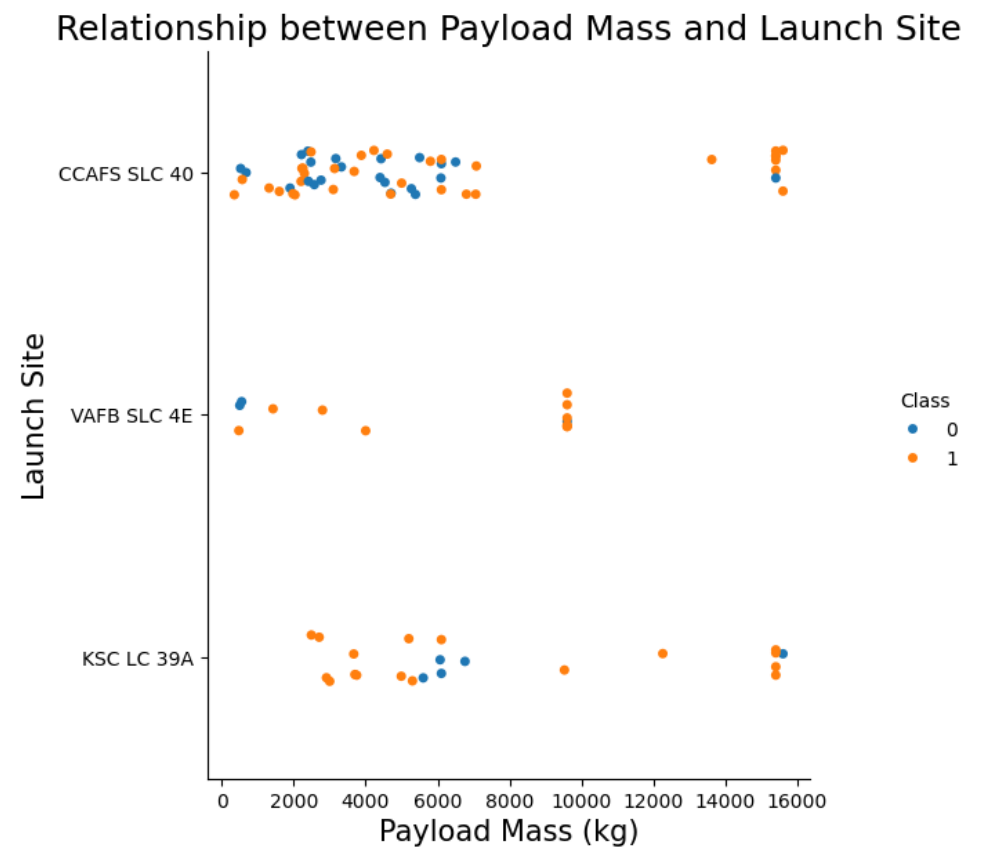
https://github.com/johnny-zeng-analysis/Coursera-Python-Capstone-Project/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

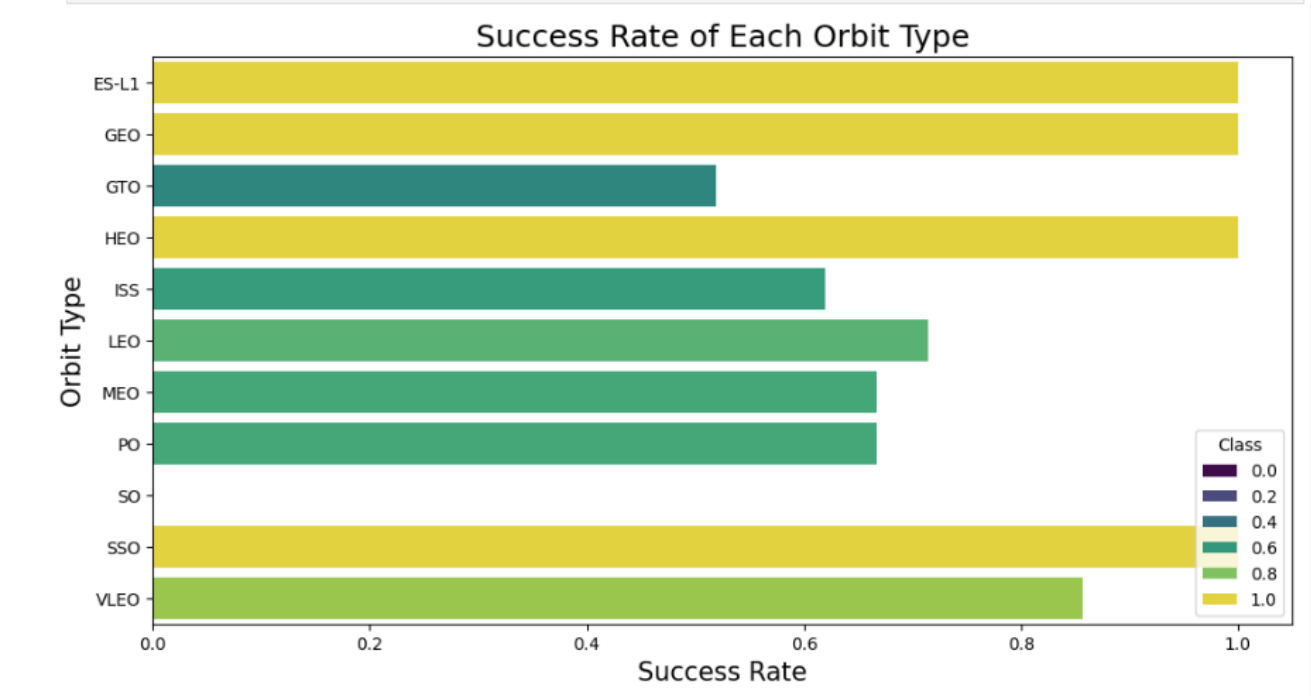
EDA with Data Visualization



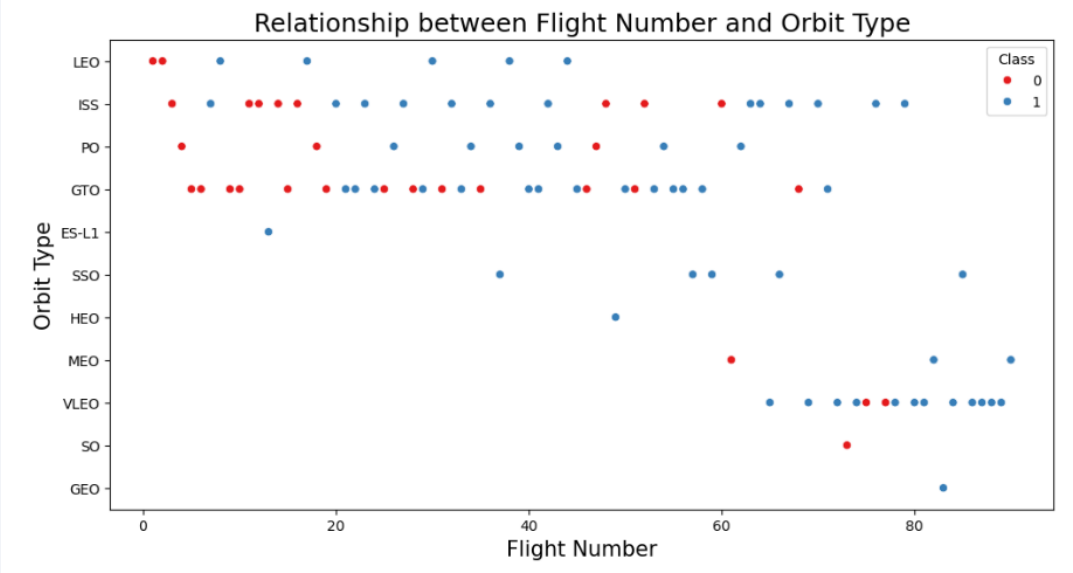


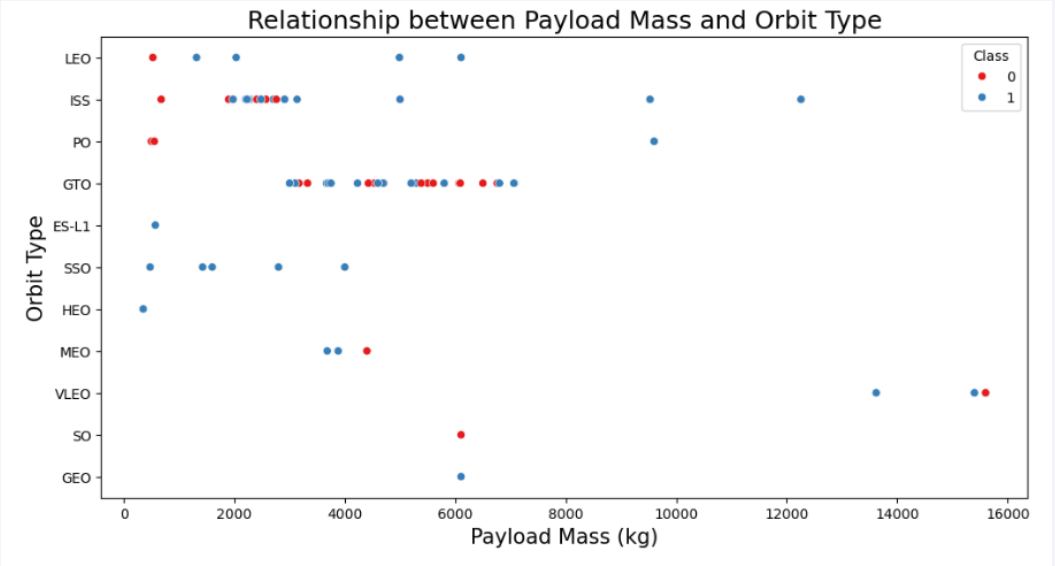
EDA with Data Visualization

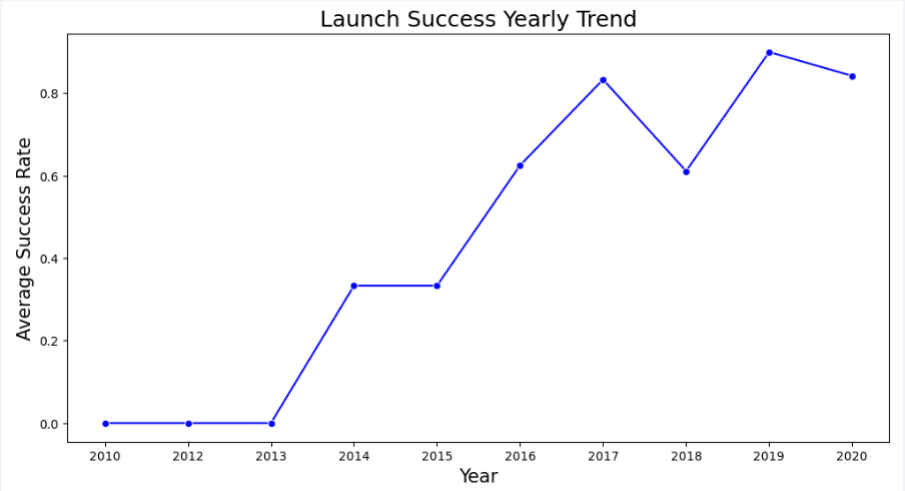




EDA with Data Visualization







EDA with SQL

- SQL was used to perform various queries such as, names of unique launch sites, launch sites beginning with specific strings, total payload mass carried by boosters, average payload mass carried by booster version F9, dates of successful landing outcomes, and many more.
- This is the following Github URL of the complete code:

https://github.com/johnny-zeng-analysis/Coursera-Python-Capstone-Project/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Folium map with marked launch sites with markers, circles was created.
- This is the following Github URL of the complete code:

https://github.com/johnny-zeng-analysis/Coursera-Python-Capstone-Project/blob/main/lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- An interactive dashboard application was created using Plotly dash which includes:
  - Launch Site Drop-down
  - Callback function to render success-pie-chart based on site dropdown selected
  - Range slider to select payload
  - Callback function to render success-payload-scatter-chart scatter plot

This is the following Github URL of the complete code:

https://github.com/johnny-zeng-analysis/Coursera-Python-Capstone-Project/blob/main/labs_jupyter_dashboard_application_plotly_dash.py

Predictive Analysis (Classification)

1- Loading the data as a Pandas Dataframe was the first step before performing exploratory Data analysis.

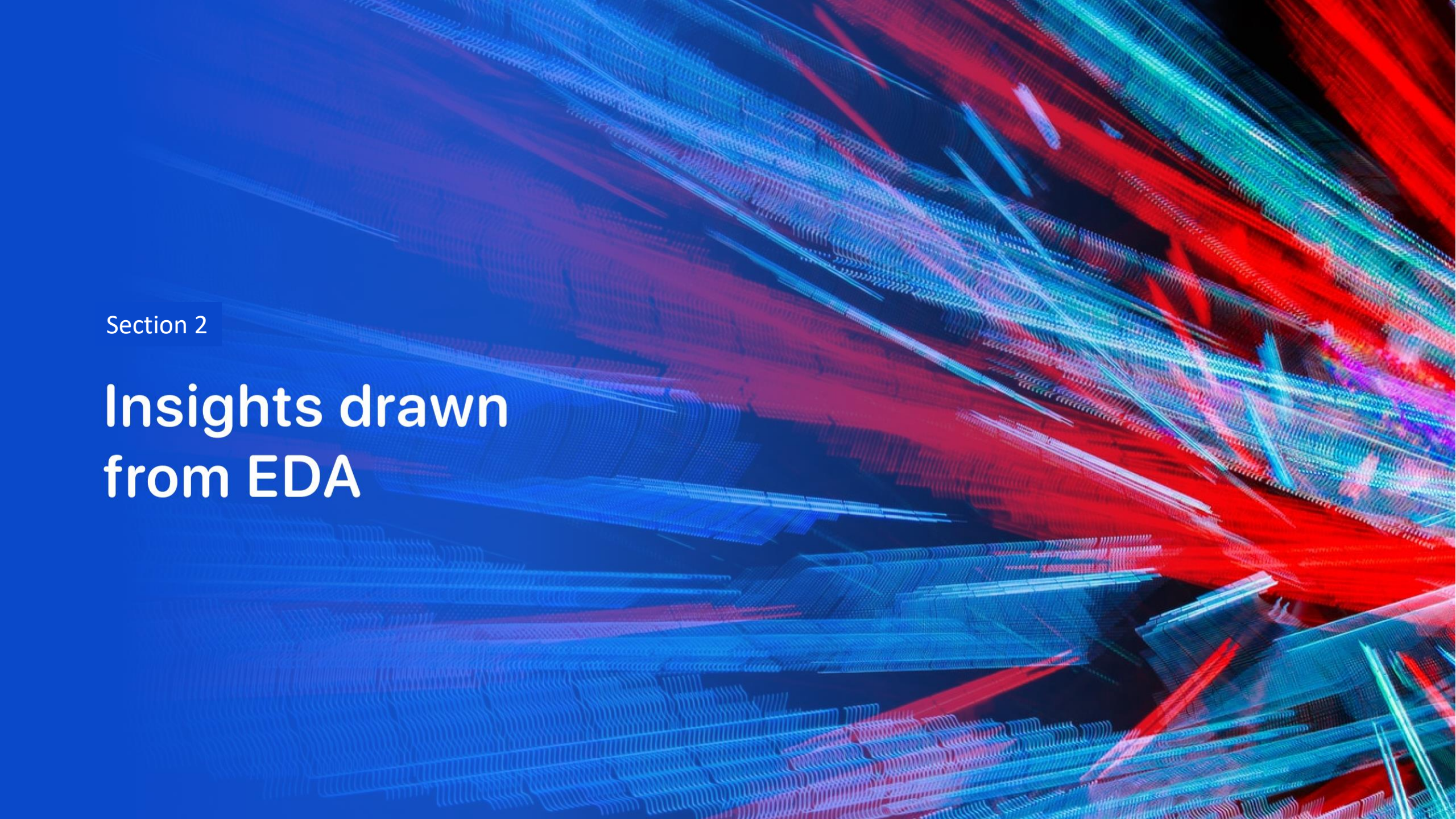
2- I then proceeded to create a Numpy array from the column class in data

3- Then I standardized the dataset (x) by transforming it using preprocessing.StandardScaler() from SkIearn

4- The data was then split into training and testing sets used for analysis

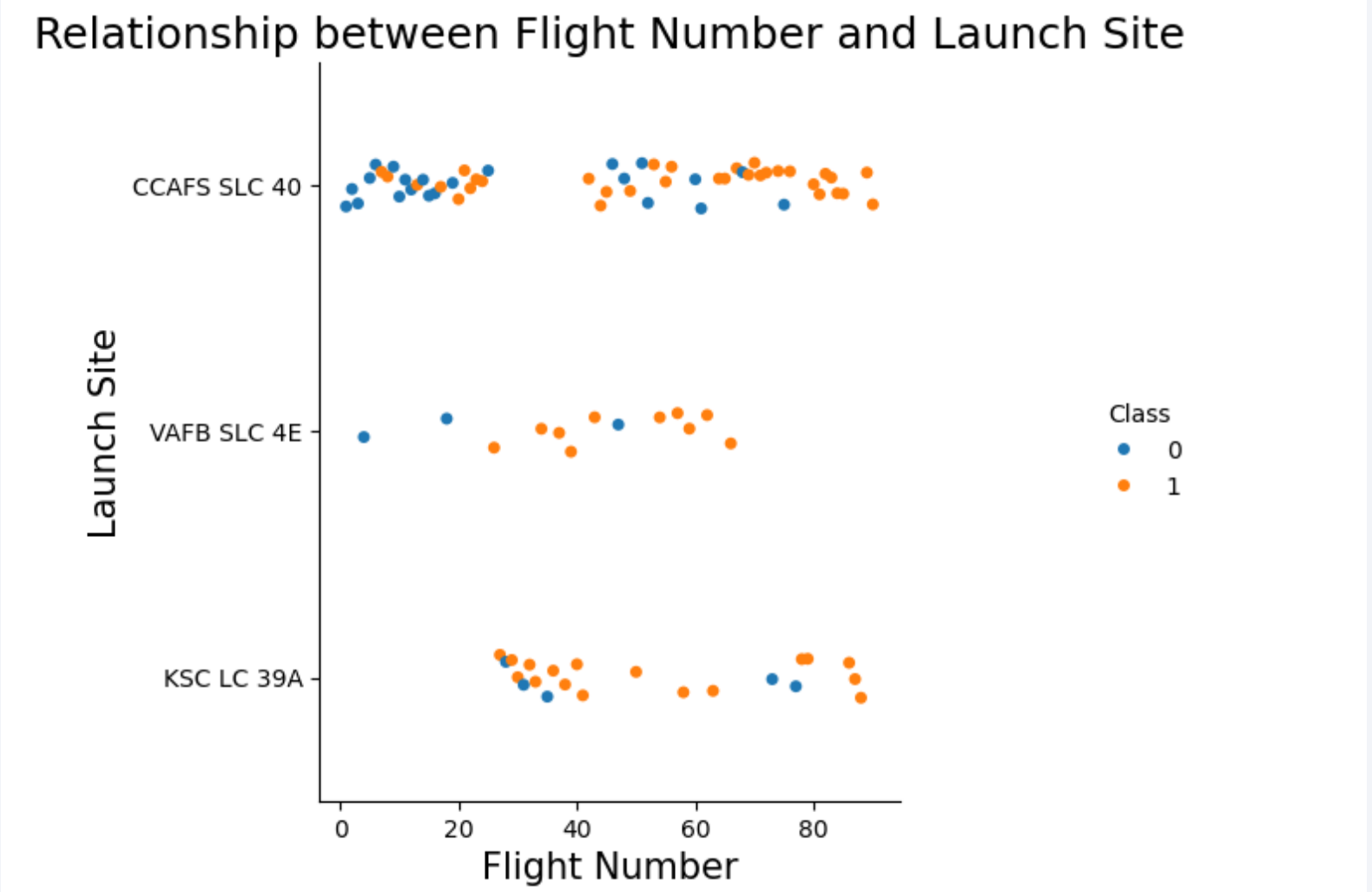
Results

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


Section 2
Insights drawn from EDA

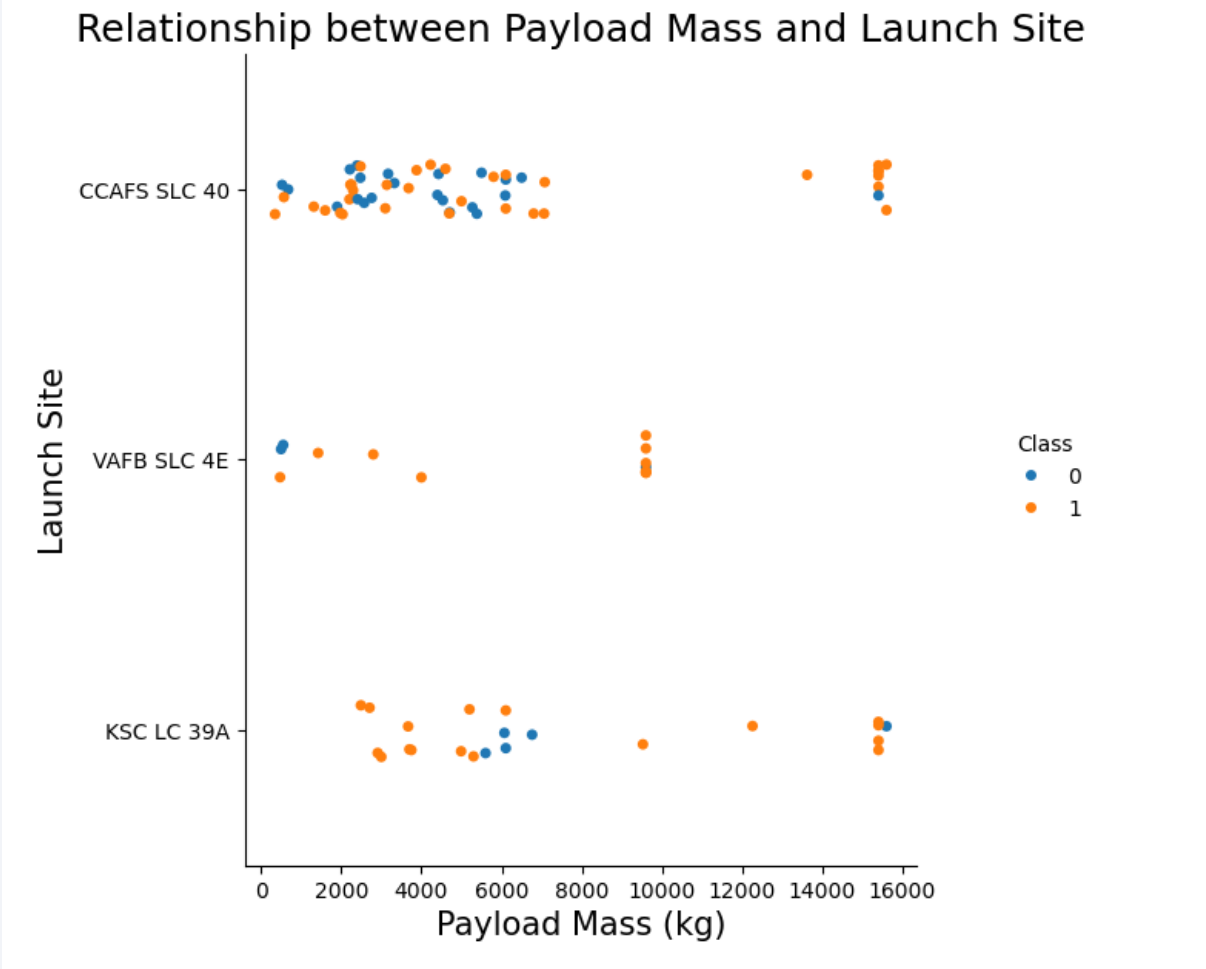
Flight Number vs. Launch Site

The graphs shows as the flight number increases in each of the launch sites, the success rate also increases



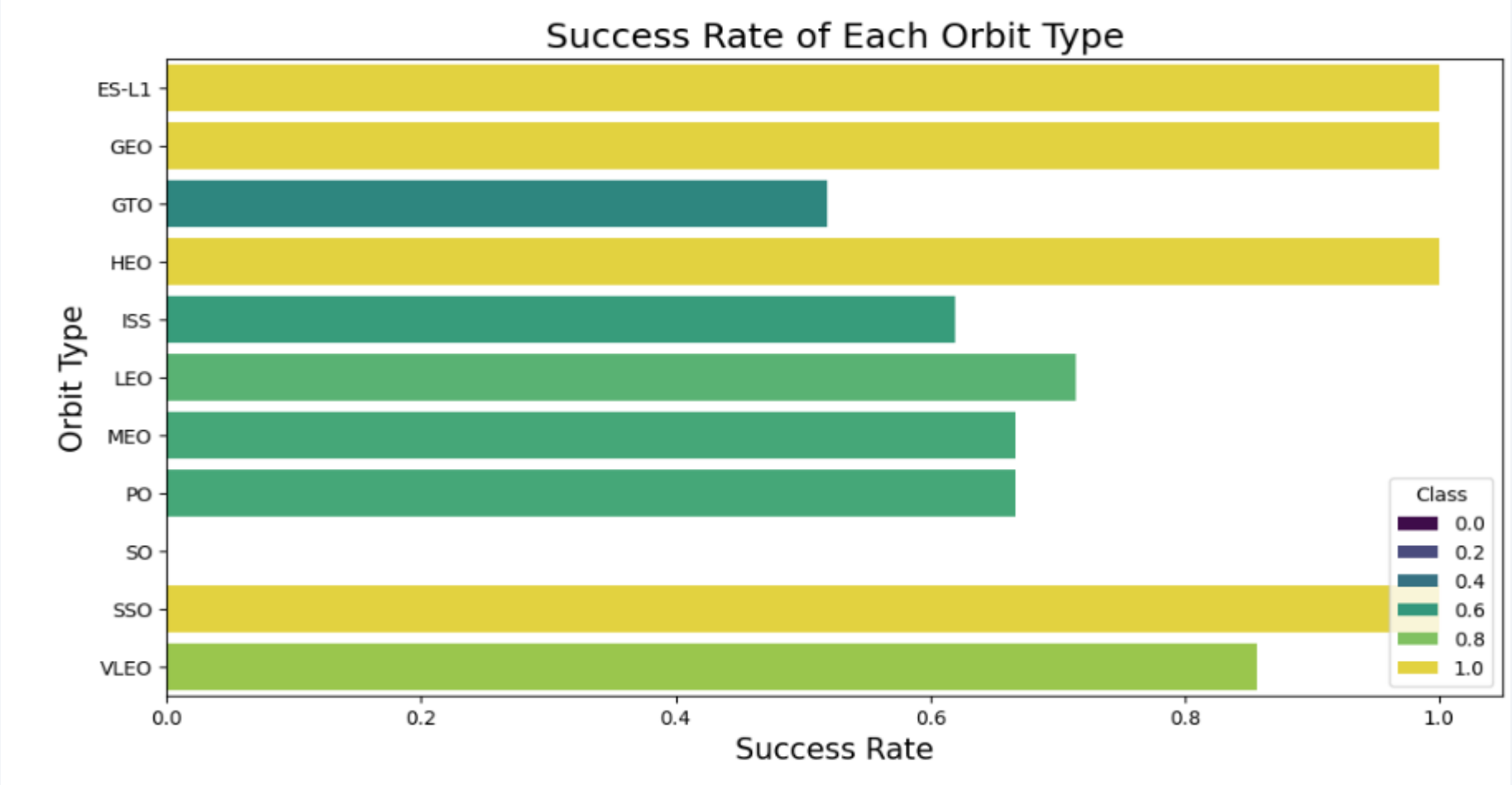
Payload vs. Launch Site

- This chart shows that VAFB SLC 4E launchsite has no rockets launched with a payload mass greater than 10,000 kg



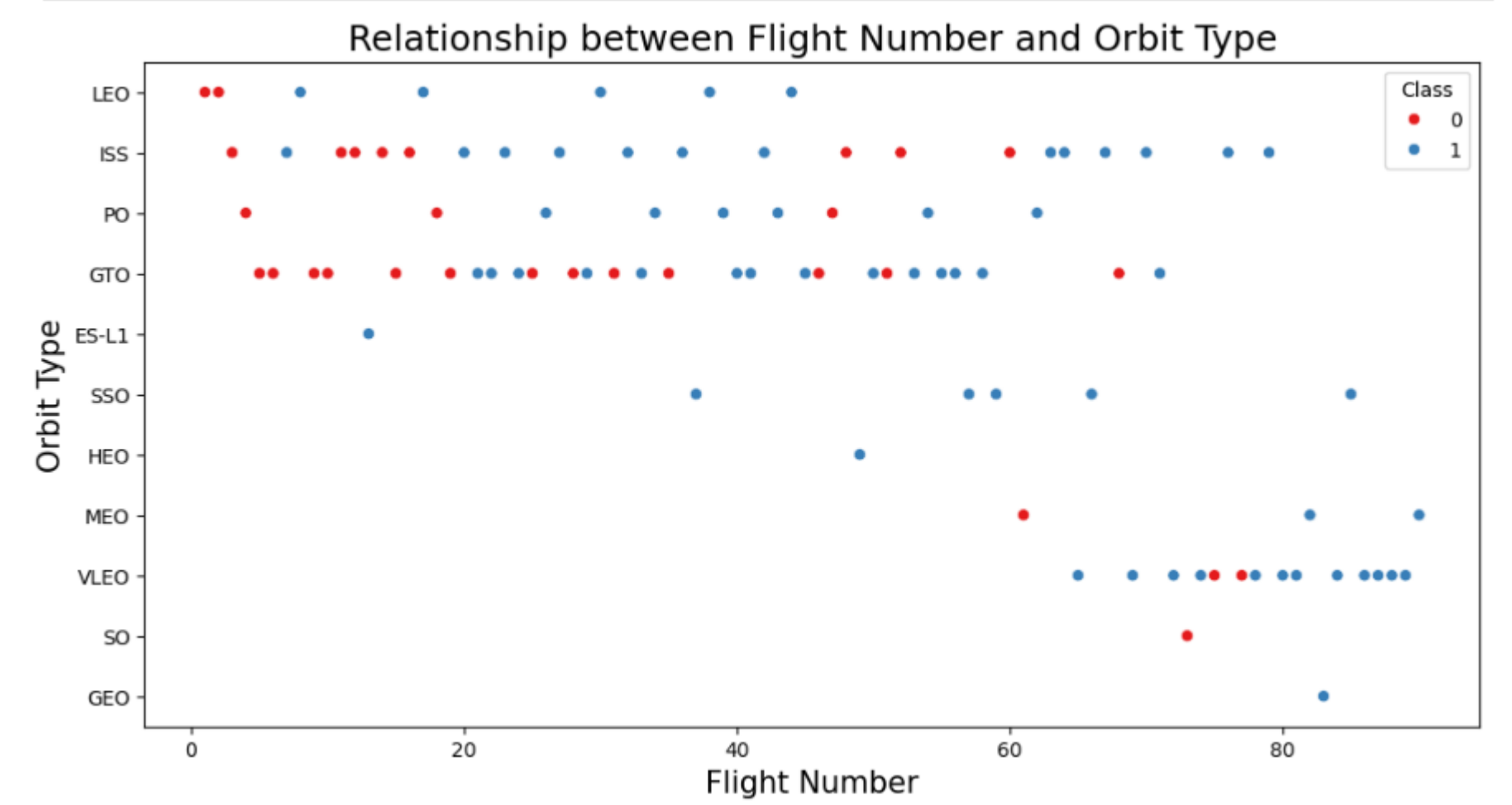
Success Rate vs. Orbit Type

- This chart shows 4 orbit types has a success rate of 100% with one having 0% success rate.



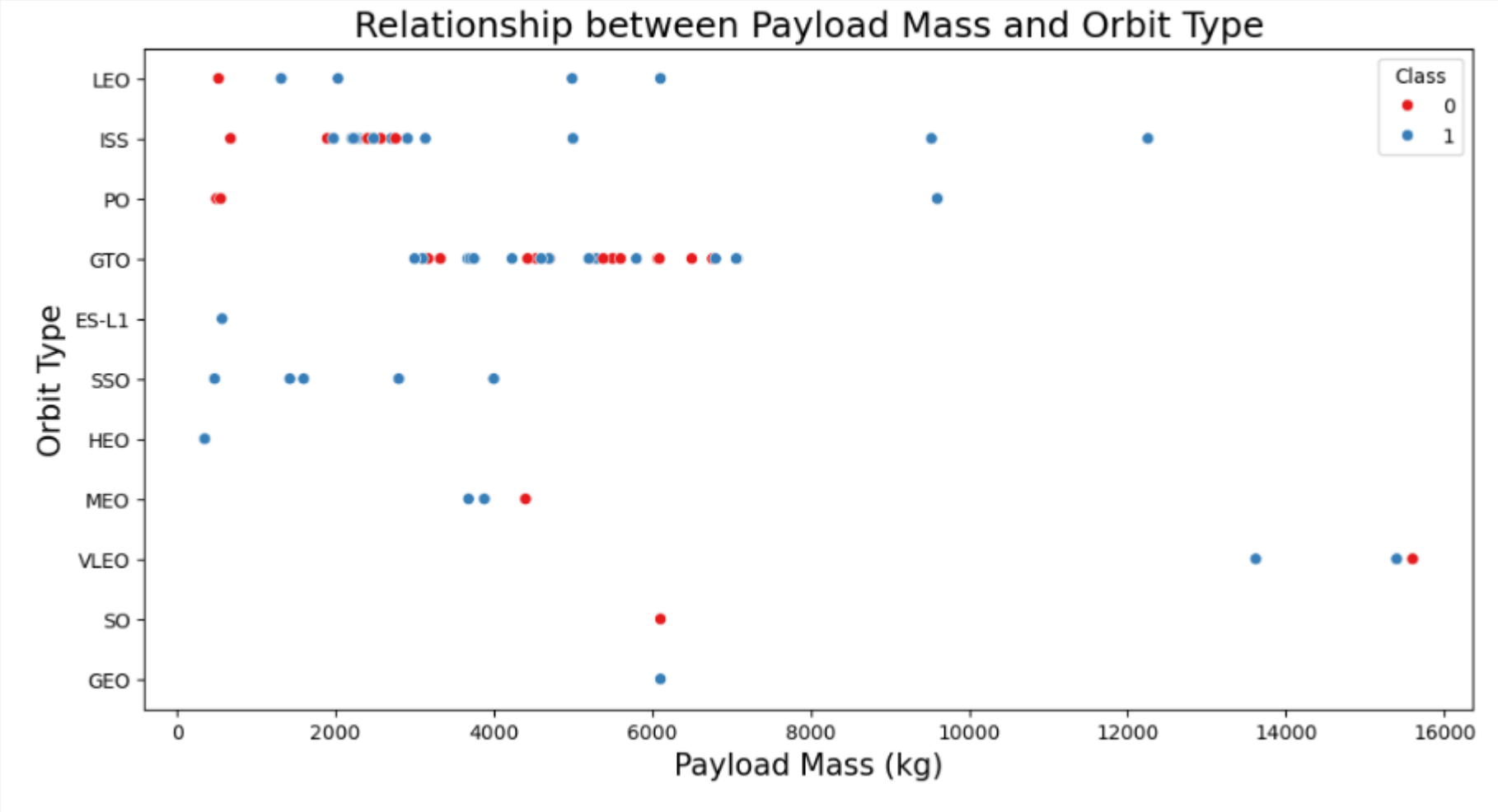
Flight Number vs. Orbit Type

- This chart shows that LEO has more success with more flights whereas GTO seems to have no pattern between flight numbers and success rate.



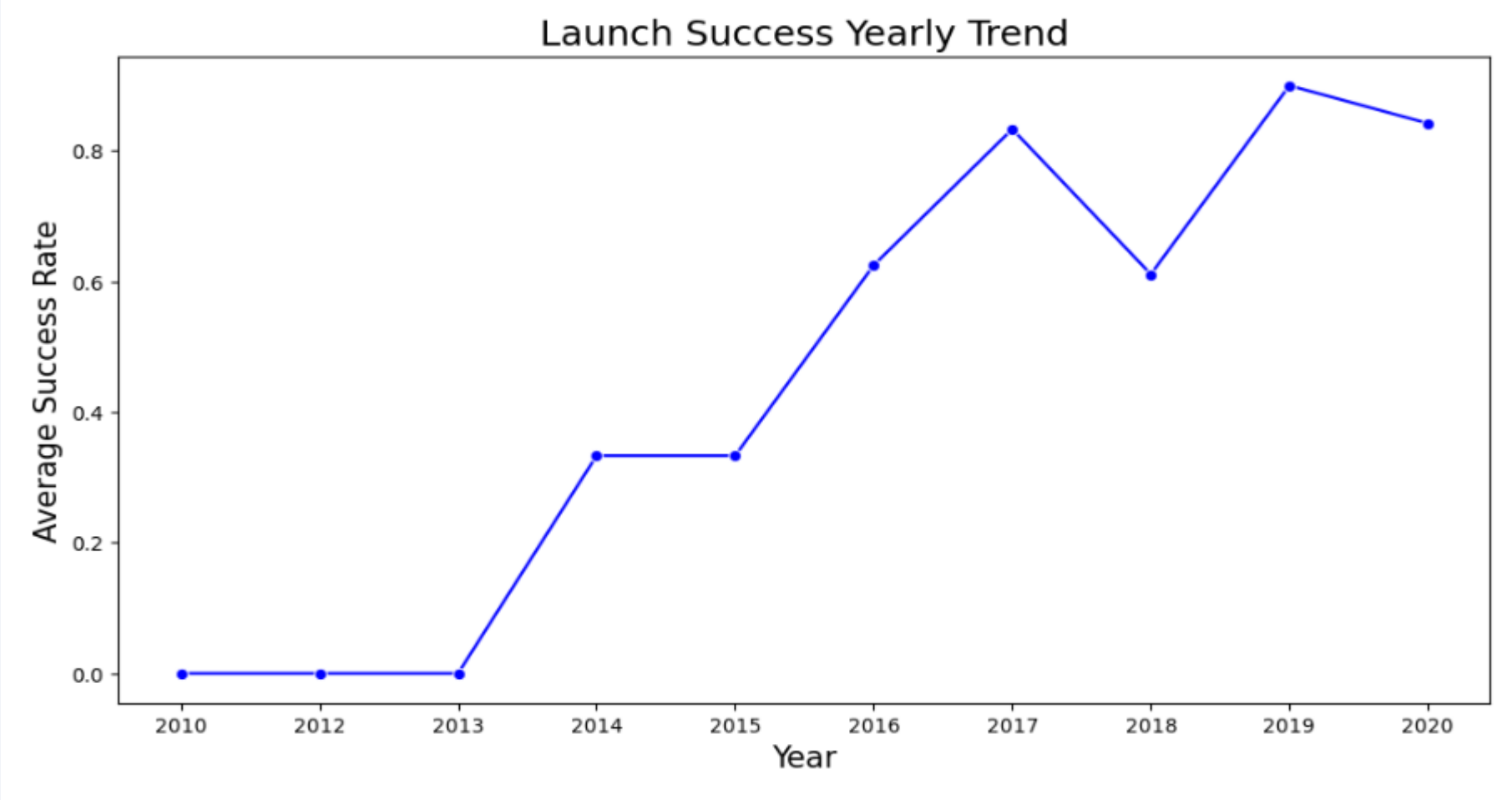
Payload vs. Orbit Type

- This chart shows heavier payload mass have a high chance of successfully landing.



Launch Success Yearly Trend

- This chart shows after 2013 success rate of launches has gone up drastically every year except for years 2017-2018 and 2019-2020.



All Launch Site Names

- Find the names of the unique launch sites
- Used SELECT DISTINCT to return 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'

- Find 5 records where launch sites begin with `CCA`
- Used LIKE % to display records with CCA string

Task 2

Display 5 records where launch sites begin with the string '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

Task 3

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
Done.
```

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

- Calculate the total payload carried by boosters from NASA
- Used SUM to return and display total sum of PAYLOAD_MASS__KG_ column

Average Payload Mass by F9 v1.1

Task 4

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
```

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

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

- Calculate the average payload mass carried by booster version F9 v1.1
- Used AVG() to return average payload mass carried by F9 v1.1

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

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

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

MIN(DATE)
2015-12-22

- Find the dates of the first successful landing outcome on ground pad
- Used MIN() to display first date of successful landing on ground pad

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

```
%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing_Outcome" = "Success (drone ship)" AND PAYLOAD_M
```

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

Booster_Version	Payload
F9 FT B1022	JCSAT-14
F9 FT B1026	JCSAT-16
F9 FT B1021.2	SES-10
F9 FT B1031.2	SES-11 / EchoStar 105

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- Used Select Distinct to return unique names of boosters between 4000 and 6000 kg mass that were success in drone ship

Total Number of Successful and Failure Mission Outcomes

Task 7

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";
```

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

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

- Calculate the total number of successful and failure mission outcomes
- Used COUNT() with GROUP BY to return total number of missions outcomes

Boosters Carried Maximum Payload

Task 8

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

- List the names of the booster which have carried the maximum payload mass
- Used subquery to return Max payload

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.

```
%sql SELECT substr(Date,6,2) as Month, substr(Date, 0, 5) as Year, "Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS_
```

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

Month	Year	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	Landing_Outcome
01	2015	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	Success	Failure (drone ship)
04	2015	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	Success	Failure (drone ship)

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- Used substr() to extract months in year 2015 where landing_outcome were failure (drone ship)

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

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.

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

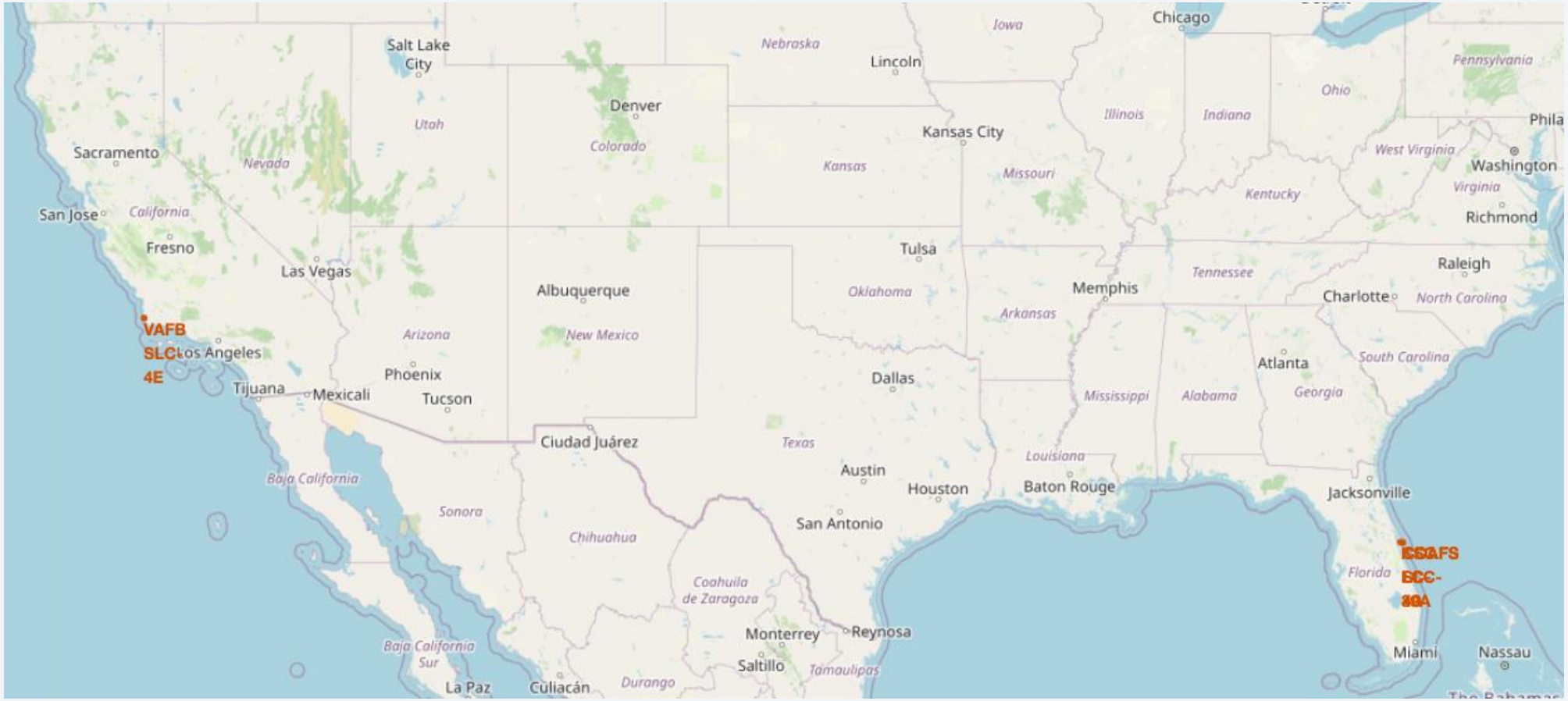
```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outc
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (gro
2017-01-14	17:54:00	F9 FT B1029.1	VAFB SLC-4E	Iridium NEXT 1	9600	Polar LEO	Iridium Communications	Success	Success (dr
2016-08-14	5:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (dr
2016-07-18	4:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (gro
2016-05-27	21:39:00	F9 FT B1023.1	CCAFS LC-40	Thaicom 8	3100	GTO	Thaicom	Success	Success (dr
2016-05-06	5:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (dr
2016-04-08	20:43:00	F9 FT B1021.1	CCAFS LC-40	SpaceX CRS-8	3136	LEO (ISS)	NASA (CRS)	Success	Success (dr
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (gro

Section 3

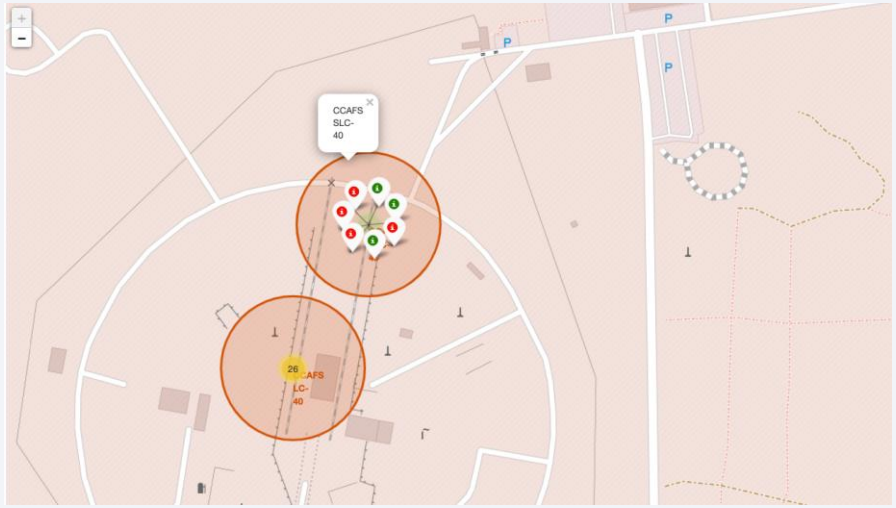
Launch Sites Proximities Analysis

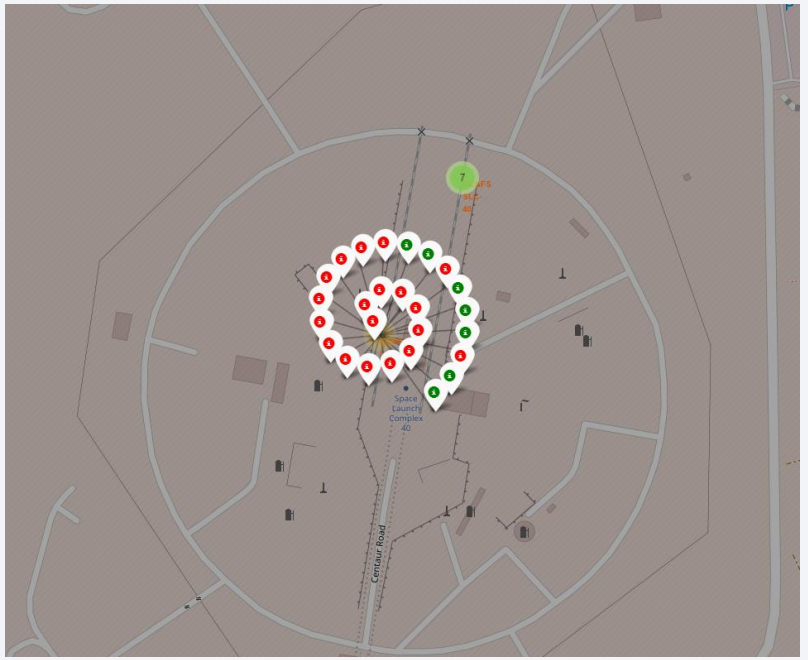
Markers of launch sites



The launch sites are located in the United States as shown by the markers

Launch outcomes for each launch site with markers

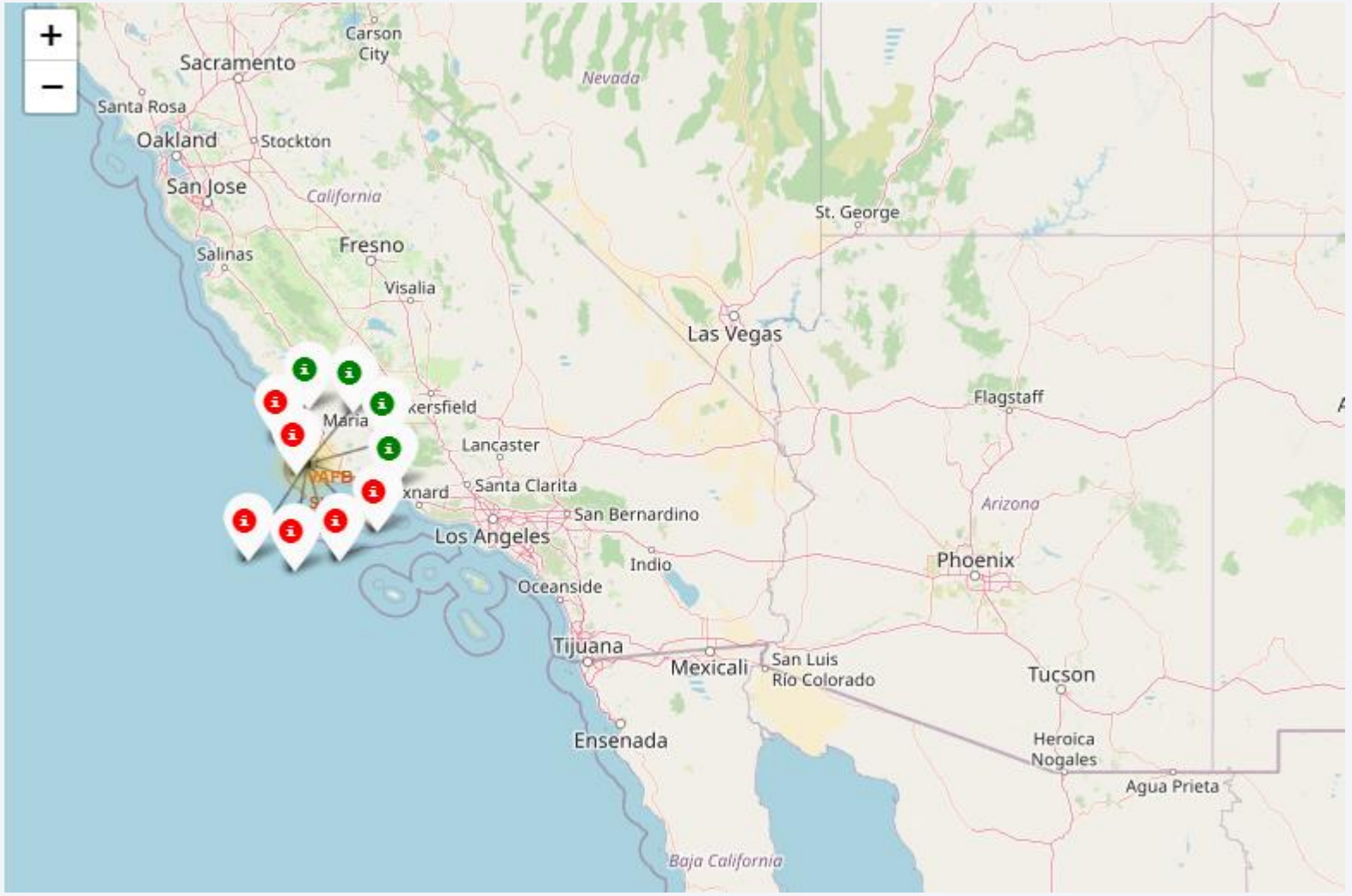


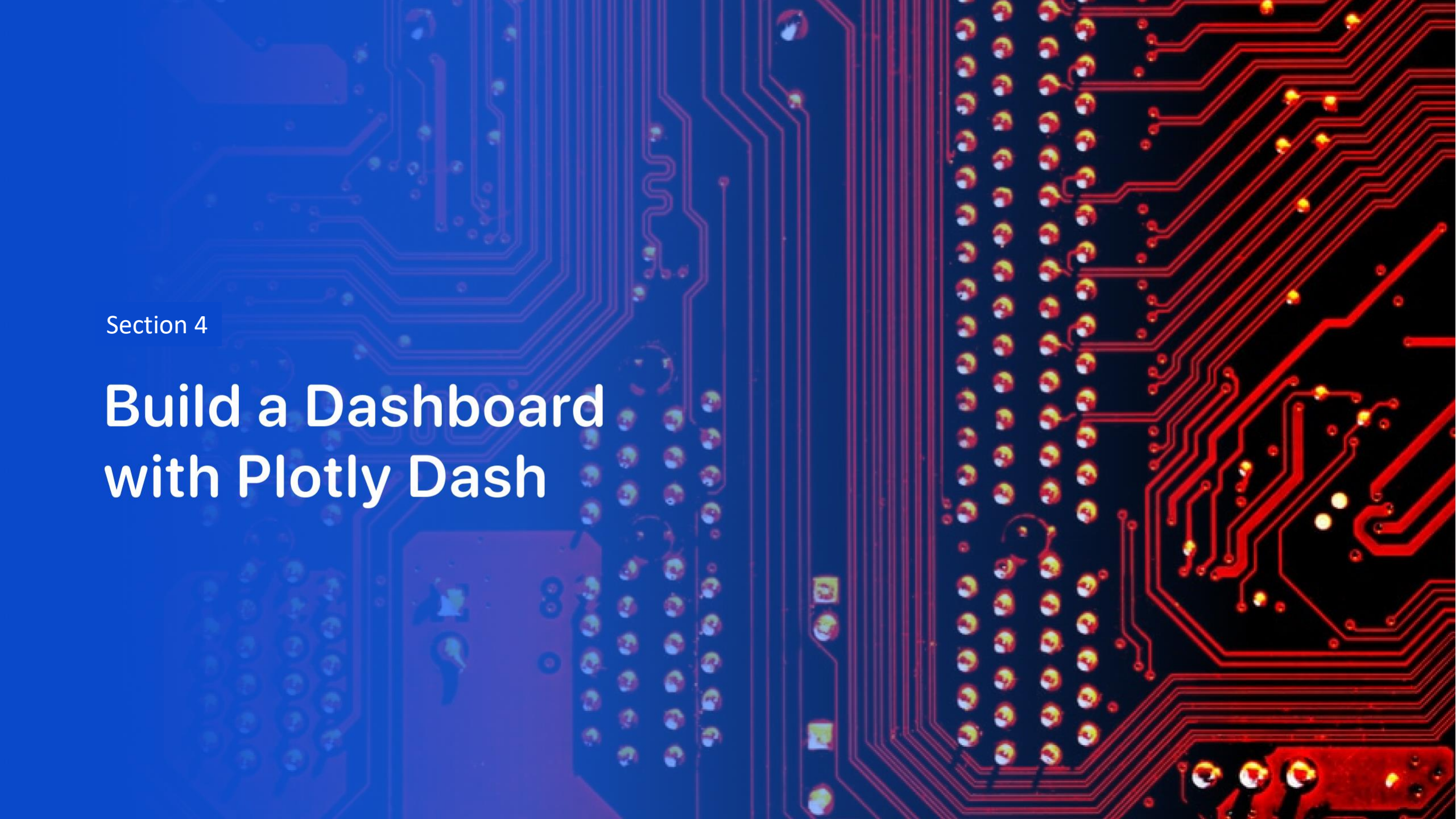


- This shows the success of the launches from each site

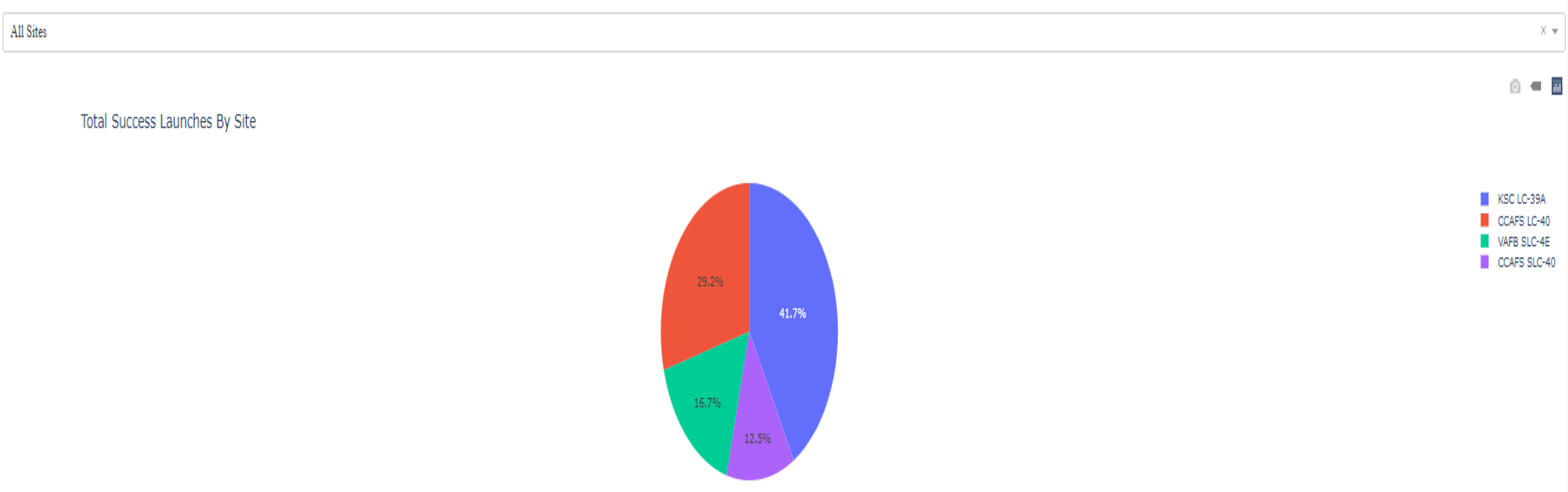
Launch outcomes for California site

- This shows the success of the launch outcomes on the California launch site




Section 4
Build a Dashboard with Plotly Dash

Pie Chart Total Successes by Site



- This shows the total successes for all 4 launch sites in a pie chart

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- All methods have the same accuracy of 0.833333

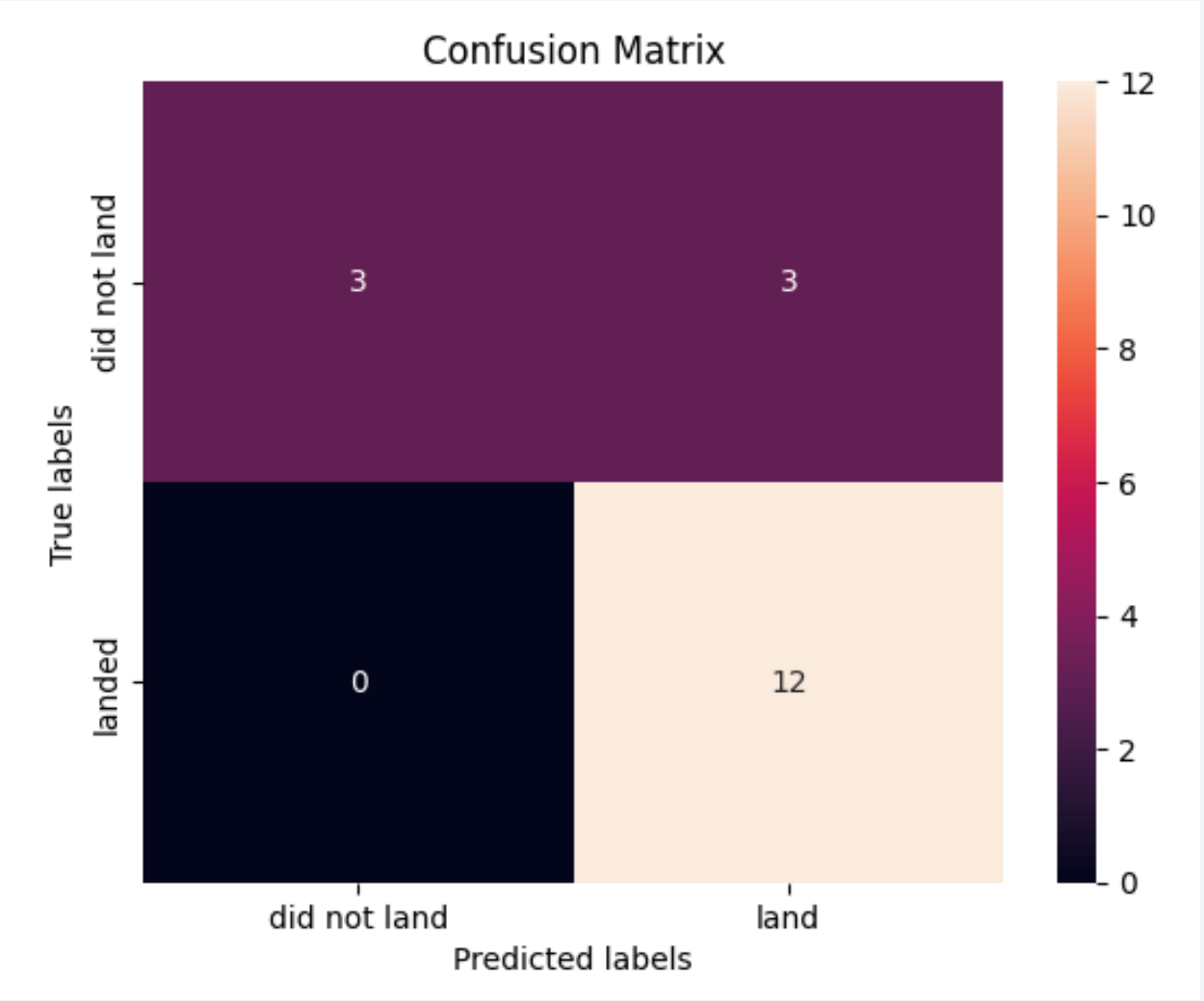
Find the method performs best:

```
Report = pd.DataFrame({'Method' : ['Test Data Accuracy']})

knn_accuracy=knn_cv.score(X_test, Y_test)
Decision_tree_accuracy=tree_cv.score(X_test, Y_test)
SVM_accuracy=svm_cv.score(X_test, Y_test)
Logistic_Regression=logreg_cv.score(X_test, Y_test)

Report['Logistic_Reg'] = [Logistic_Regression]
Report['SVM'] = [SVM_accuracy]
Report['Decision Tree'] = [Decision_tree_accuracy]
Report['KNN'] = [knn_accuracy]



Report.transpose()
```

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

Confusion Matrix

- All 4 of the classification models had the same confusion matrixes



Conclusions

- Different launch sites have different success rates
- For 3 of the 4 launch sites, when number of flights increases, so does the success rate of the launches.
- For VAFB-SLC, they do not have launches of payload mass greater than 10000 kg
- GEO, SSO, HEO, and ES-L1 orbits have 100% success rates.
- GTO orbit seems to have no correlation between flight number and success rate

Thank you!