IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Archana Hegde

Outline

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

Executive Summary

- Summary of methodologies
  - Defining objectives of the project.
  - Data collection using spacex api, processed with SQL and web scraping
  - Data wrangling and exploratory analysis and data visualisation
  - Interactive maps with Folium

- Summary of all results
  - Data visualisation to view results
  - Selecting best model to predict results.

Introduction

- Project background and context

  In this project, 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 due to the fact that SpaceX can reuse the first stage.

- Problems you want to find answers
- What factors affect successful rocket landing.
- Predicting first stage will land successful or not.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collected using spacex api
  - Web scraping from Wikipedia
- Perform data wrangling
  - Cleaning, encoding, feature engineering, scaling
  - filtering dataframes to only include Falcon 9 data, replacing missing values with mean values and dropping columns which are unnecessary.
- 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

Data Collection – SpaceX API

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

```
response = requests.get(spacex_url)
```

```
response=requests.get(static_json_url)
```

```
# Calculate the mean value of PayloadMass column
mean_payload_mass = data_falcon9['PayloadMass'].mean()
print(f"Mean payload mass is {mean_payload_mass }")
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].replace(np.nan, mean_payload_mass)
```

Get response from API

parsing spacex Data

Dealing with missing value

Export to a flat file

```
# Use json_normalize meethod to convert the json res
data = response.json()
data = pd.json_normalize(data)
```

```
# Create a data from launch_dict
df = pd.DataFrame(launch_dict)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0

Github

Data Collection - Scraping

Getting response from HTML page

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falco

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"
}
```

Beautifulsoup for parsing

```
# Use BeautifulSoup() to create a BeautifulSoup object from
soup = BeautifulSoup(responses.text, "html.parser")
```

Dataframe by parsing html table

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
```

```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
```

Converting dataframe to csv file

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.07B0003.18	Failure
1	1	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.07B0003.18	Failure
2	2	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.07B0004.18	No attempt\n
3	3	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.07B0005.18	No attempt
4	4	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1.07B0006.18	No attempt\n

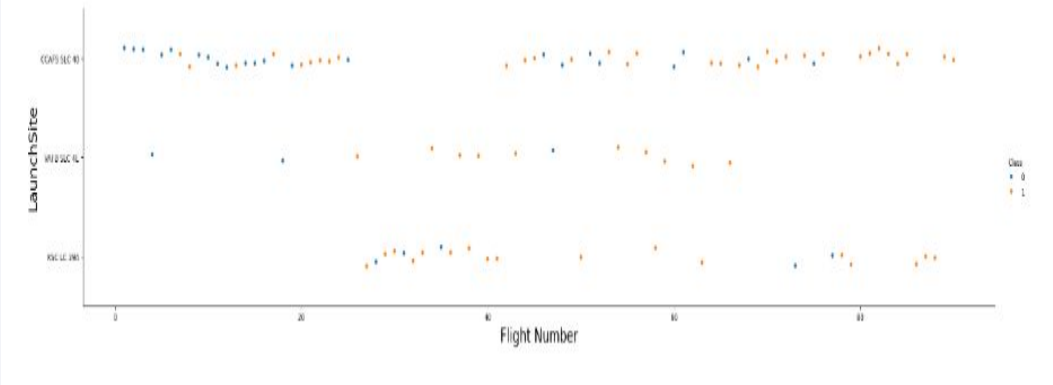
Github

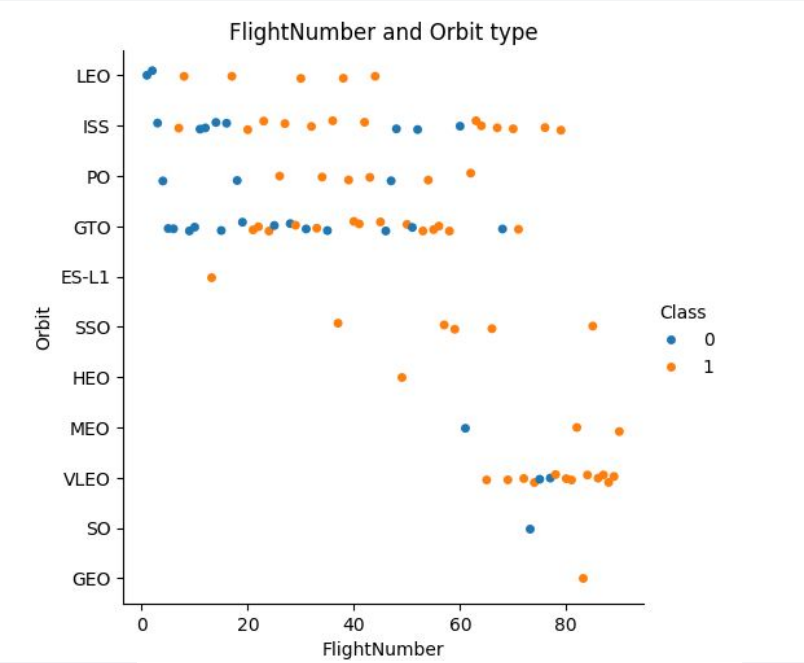
Data Wrangling

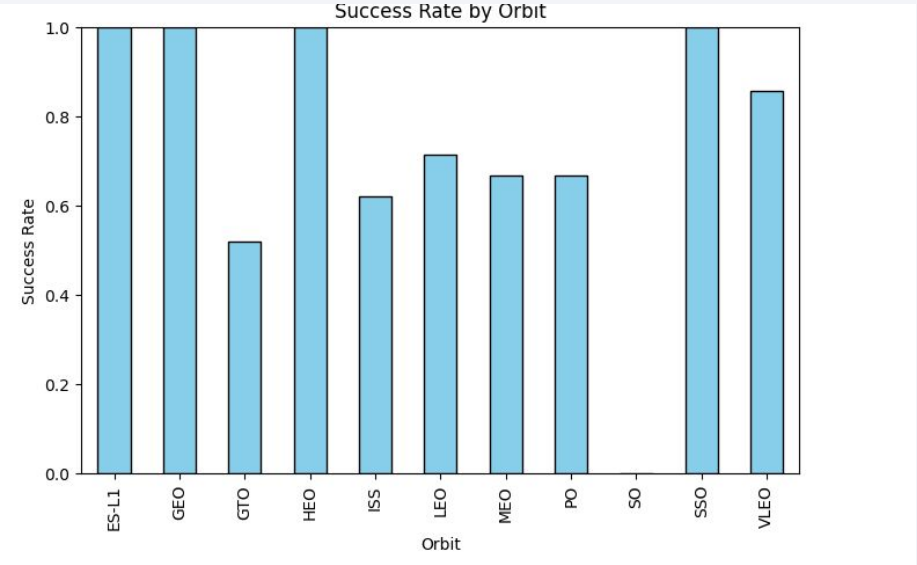
Cleaning the data, calculating number of launches per site, Calculate the number and occurrence of each orbit

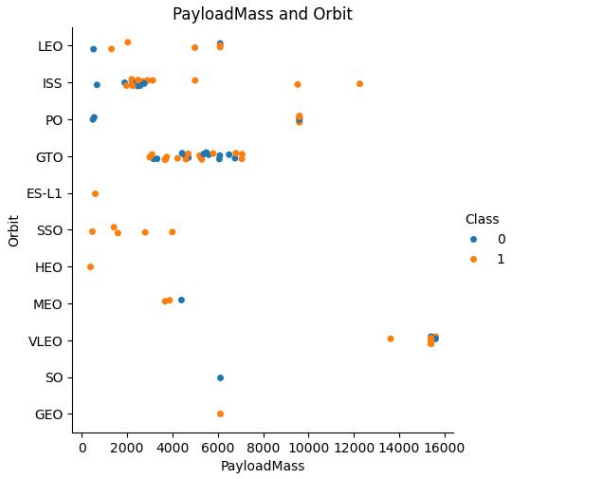
Calculate the number and occurence of mission outcome of the orbits

EDA with Data Visualization









[Github]

EDA with SQL

- Used to filter and display the data

```
%%sql select Booster_Version from SPACEXTBL
WHERE Landing_Outcome is "Success (drone ship)"
AND PAYLOAD_MASS__KG_ between 4000 AND 6000
```

- Total number of successful and failure outcomes

```
%%sql SELECT Mission_Outcome, COUNT(*) AS count
FROM SPACEXTBL
GROUP BY Mission_Outcome
ORDER BY count DESC;
```

- Counting the landing outcomes

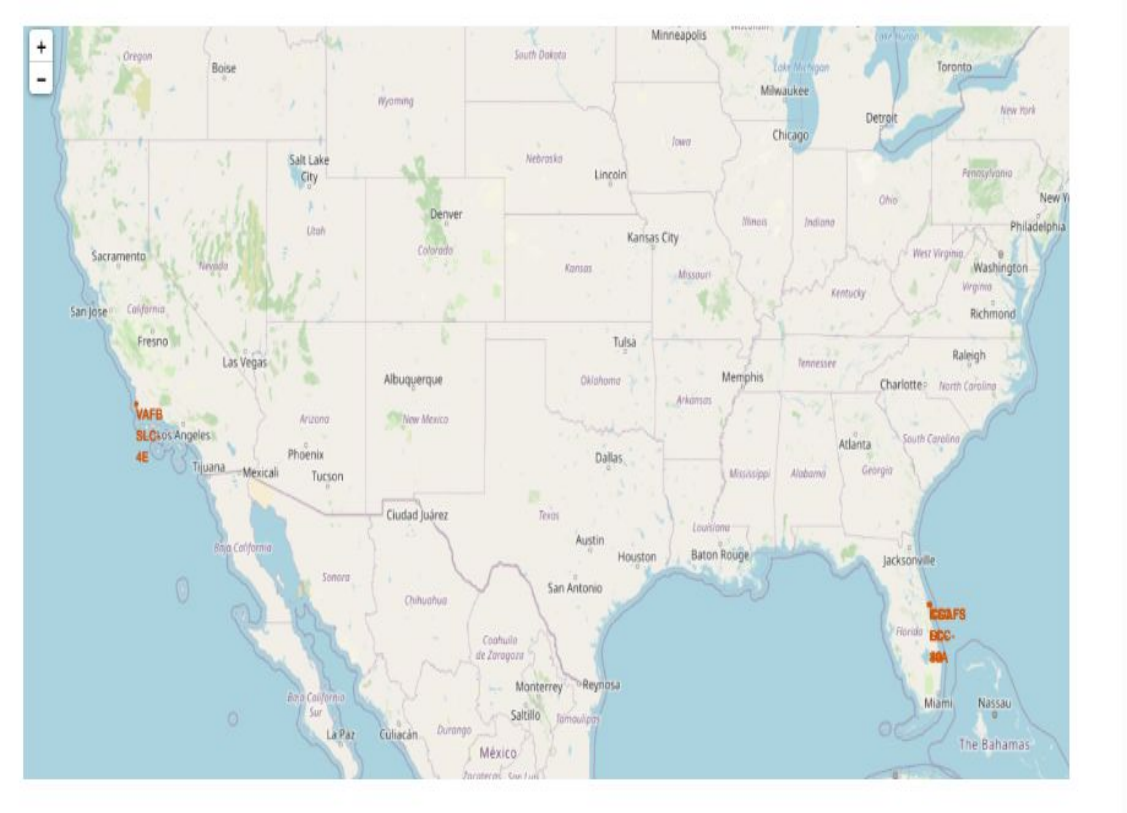
```
%%sql select Date, Landing_Outcome, count(*) as Count from SPACEXTBL
group by Landing_Outcome
order by Count desc
```

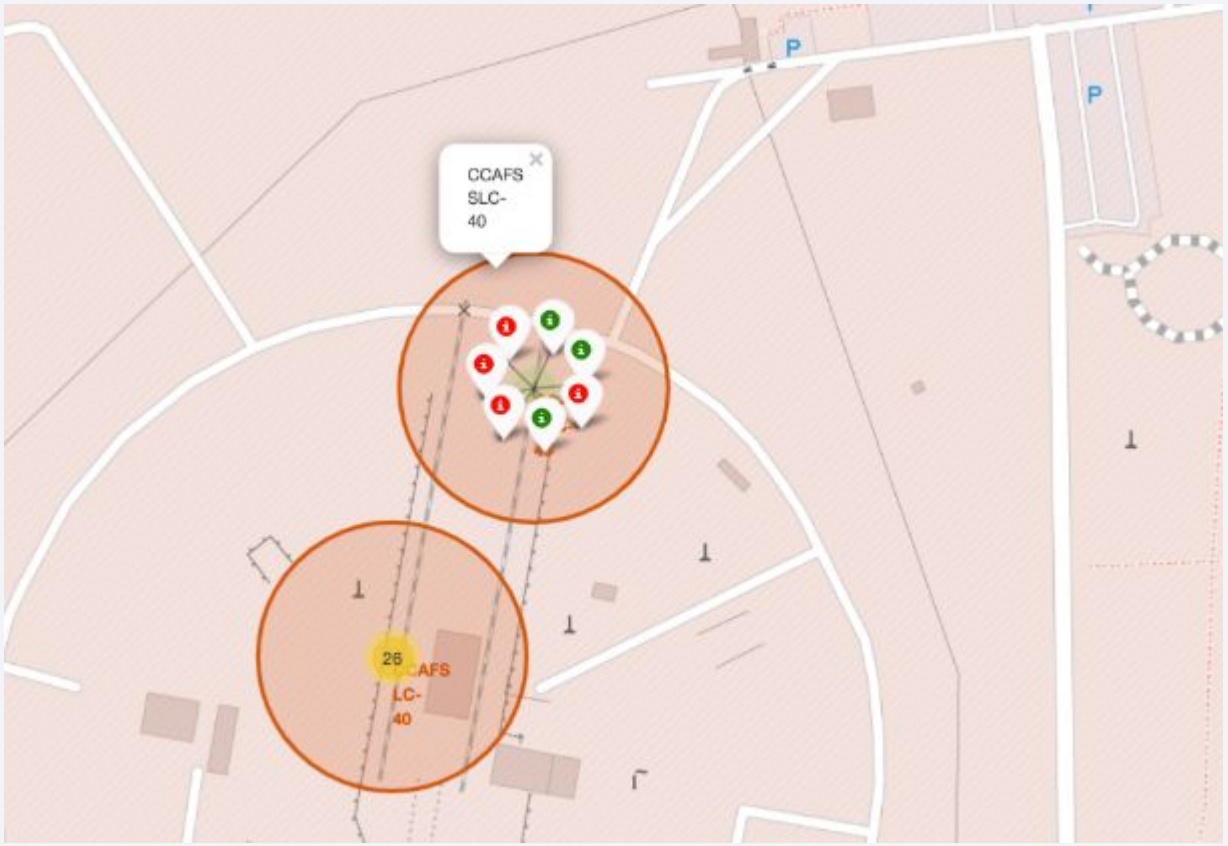
Date	Landing_Outcome	Count
2018-07-22	Success	38
2012-05-22	No attempt	21
2016-04-08	Success (drone ship)	14
2015-12-22	Success (ground pad)	9
2015-01-10	Failure (drone ship)	5
2014-04-18	Controlled (ocean)	5
2018-12-05	Failure	3
2013-09-29	Uncontrolled (ocean)	2
2010-06-04	Failure (parachute)	2
2015-06-28	Precluded (drone ship)	1
2019-08-06	No attempt	1

[github-sql]

Build an Interactive Map with Folium

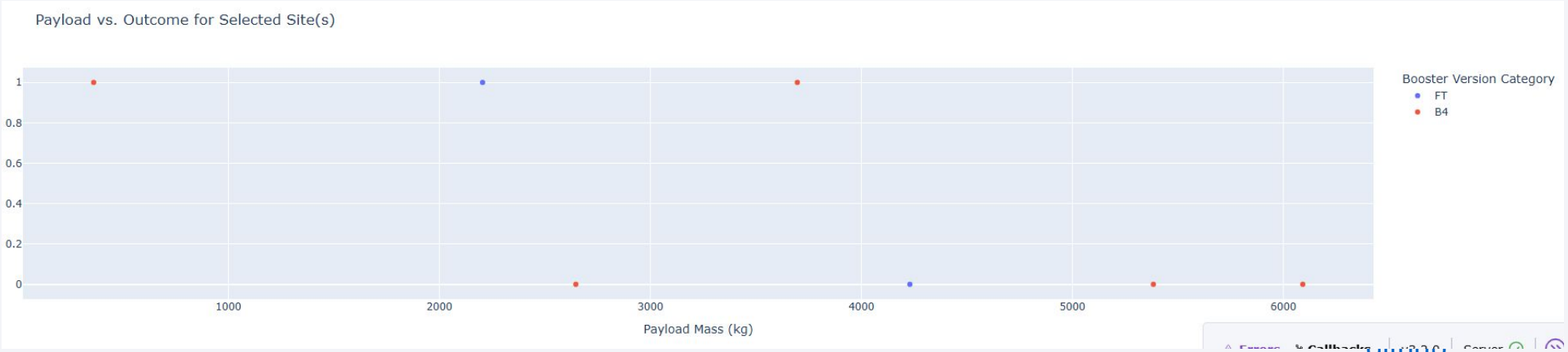
- Folium is used to display launch site on the map





Build a Dashboard with Plotly Dash

- Added different site locations and calculated success rate
- Created scatter plot for different payload masses and outcome



github

Predictive Analysis (Classification)

- Load dataframe
- Create Class array
- Data Standardization
- Split data
- applying different ml models
- Comparing accuracy scores
- select best model

```
Y = data['Class'].to_numpy()
```

```
# students get this
transform = preprocessing.StandardScaler()
```

```
X = transform.fit_transform(X)
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state = 2)
```

```
logreg_cv = GridSearchCV(lr, parameters, cv=10)
logreg_cv.fit(X_train, Y_train)
```

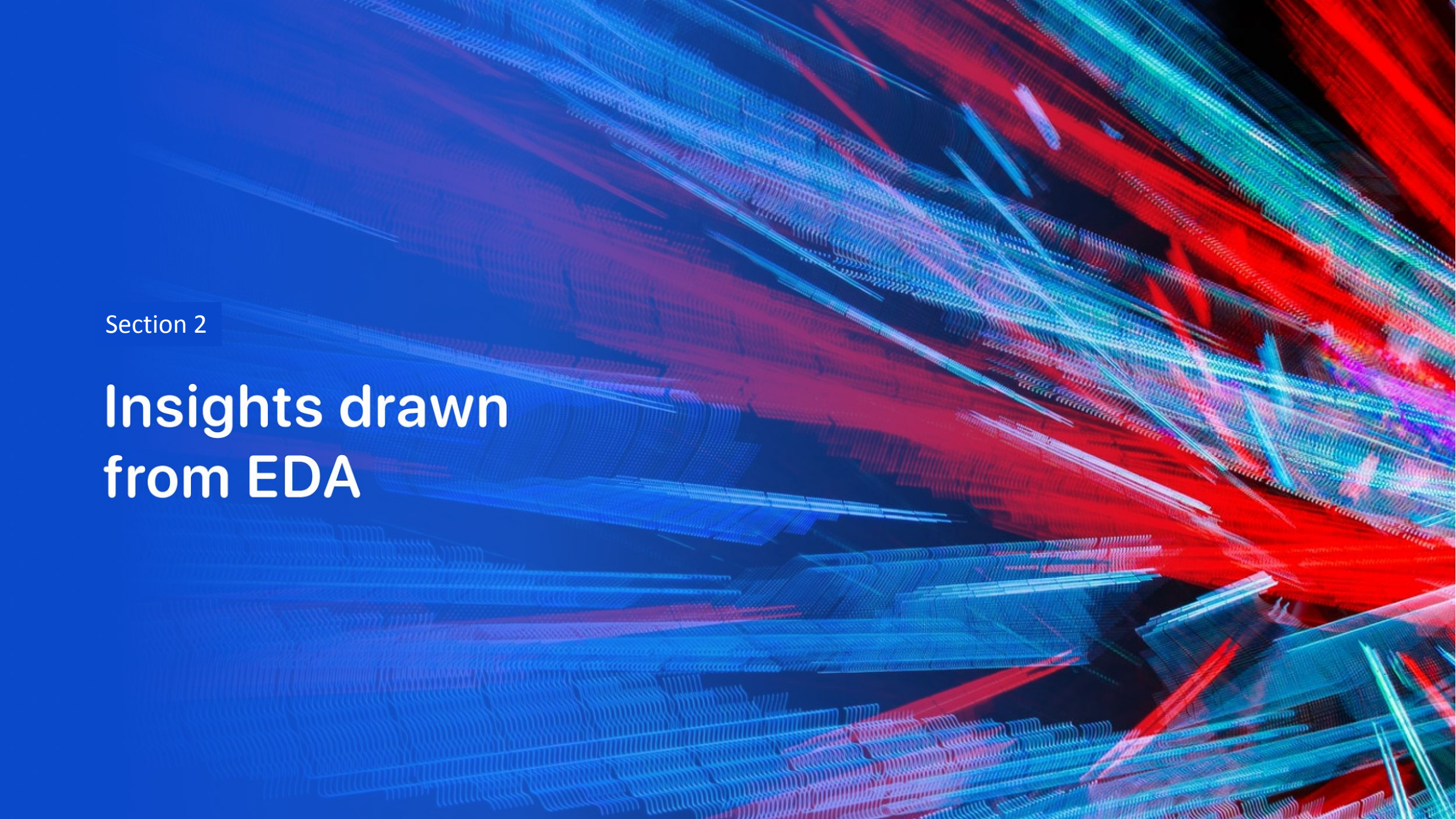
```
print(knn_cv.score(X_test, Y_test))
```

Results

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

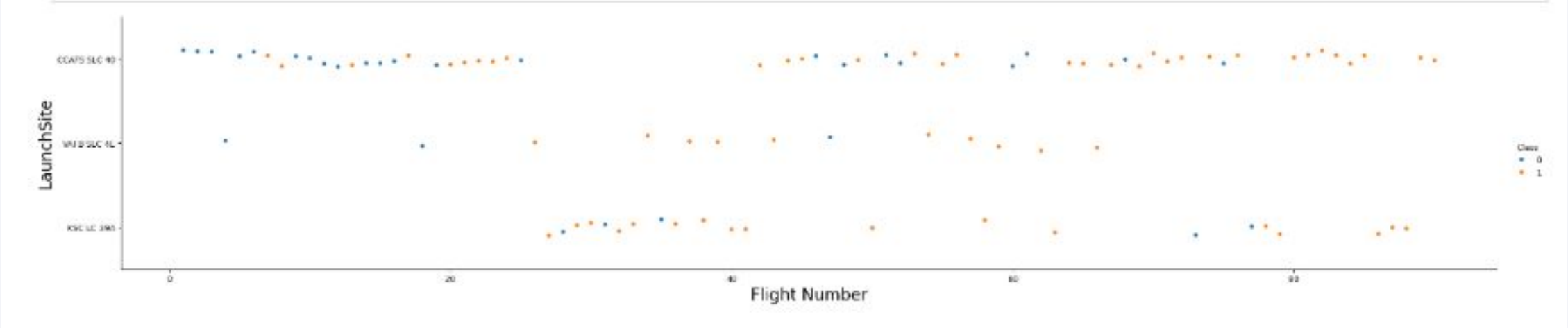
Section 2

Insights drawn from EDA



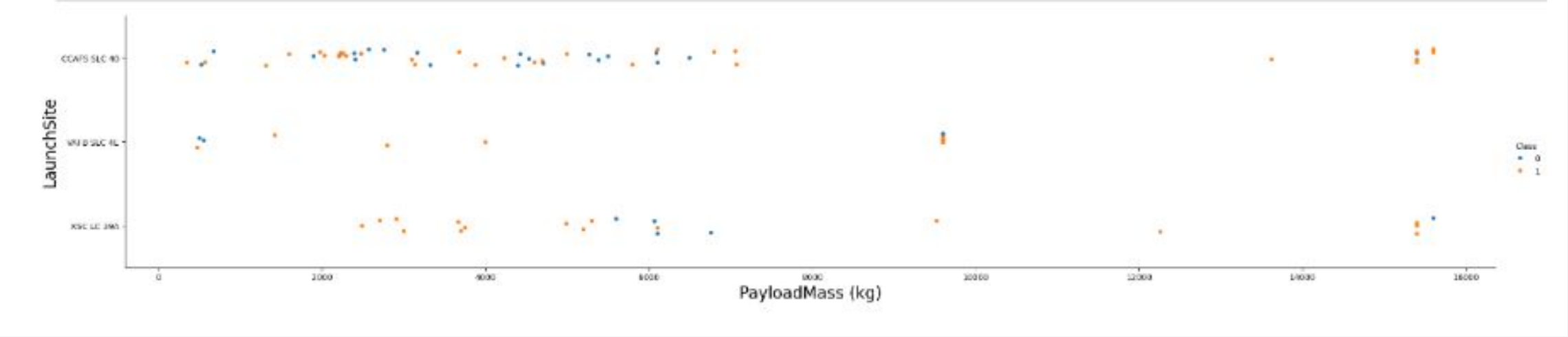
Flight Number vs. Launch Site

- Scatter plot of Flight Number vs. Launch Site



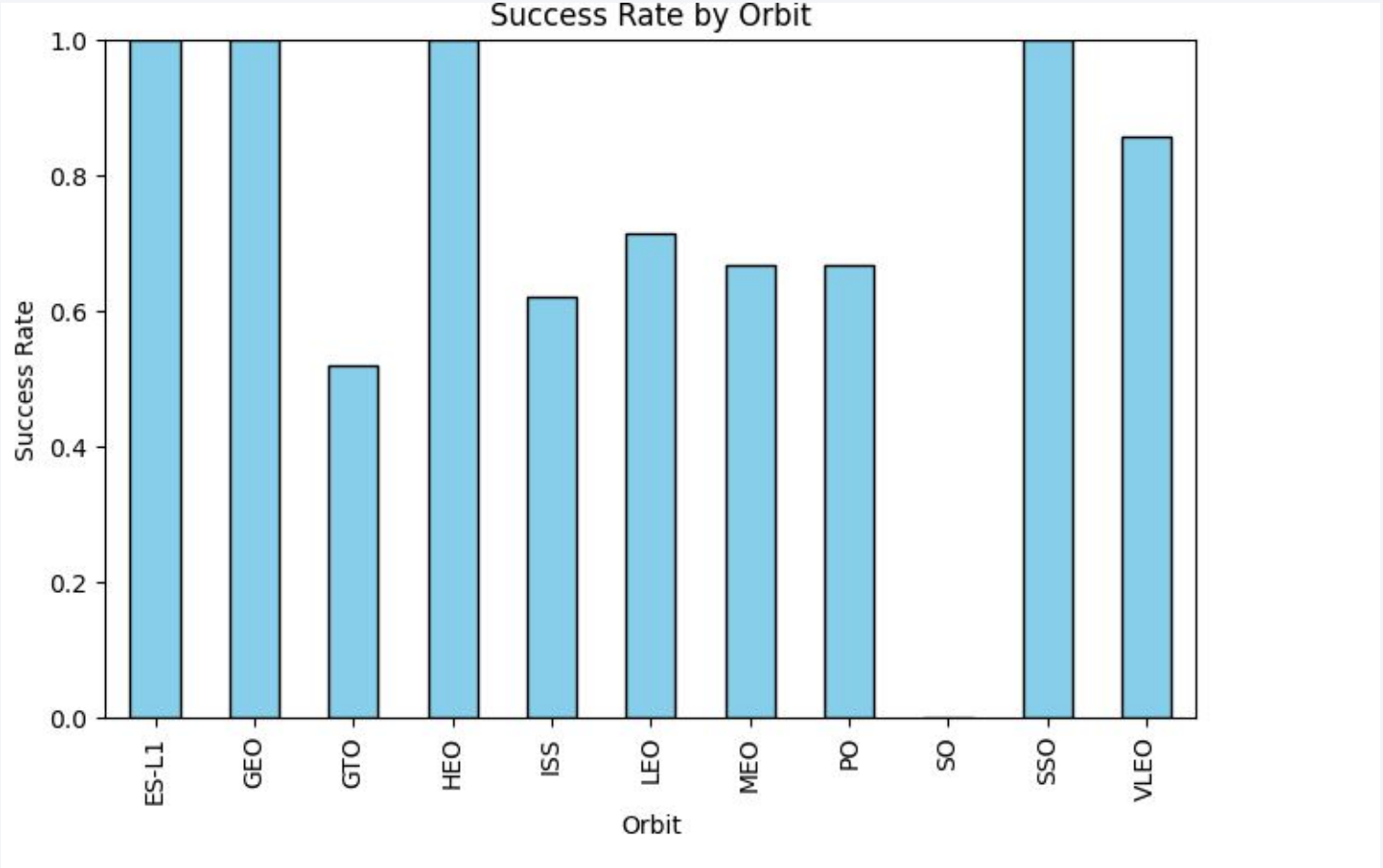
Payload vs. Launch Site

- Scatter plot of Payload vs. Launch Site

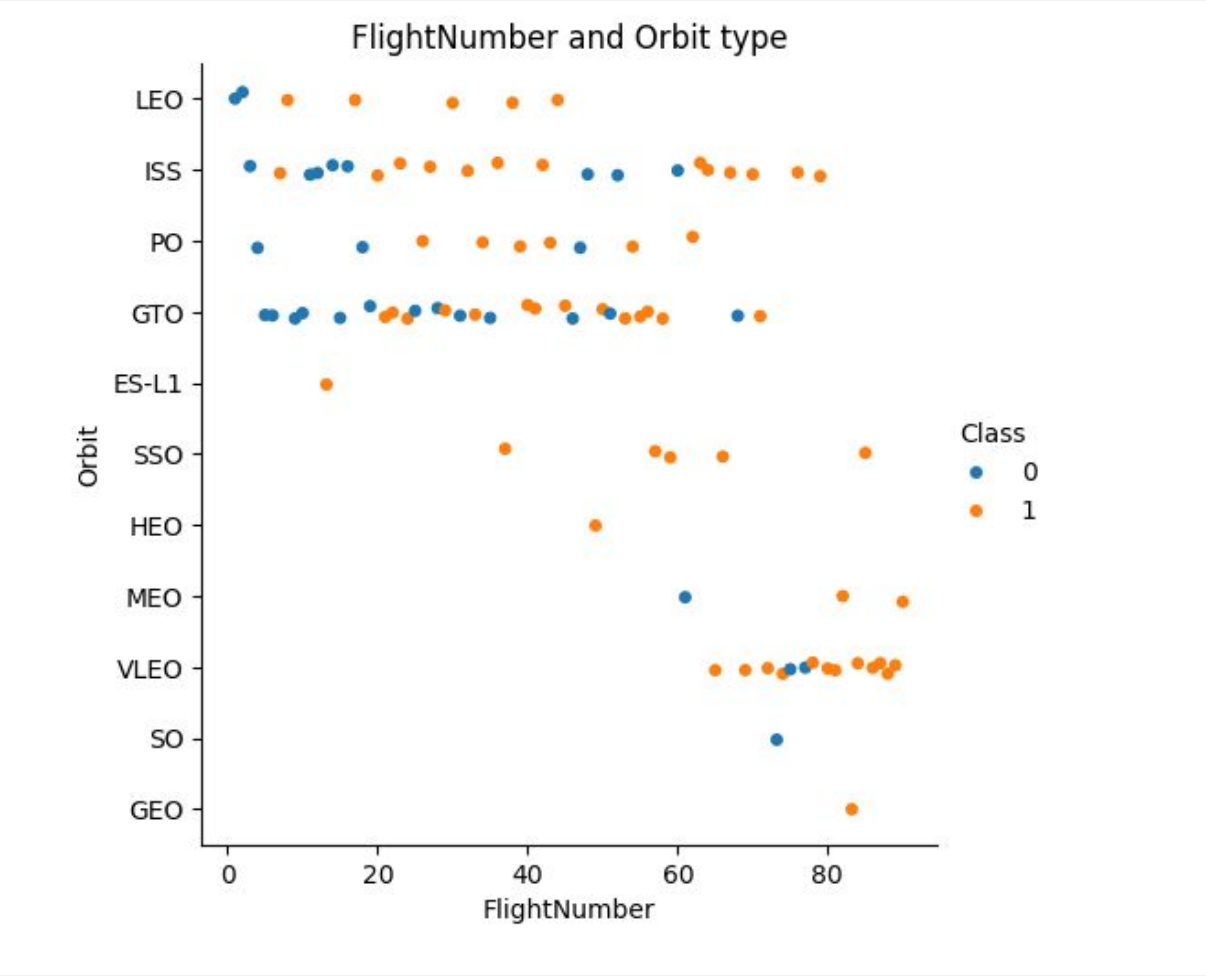


Success Rate vs. Orbit Type

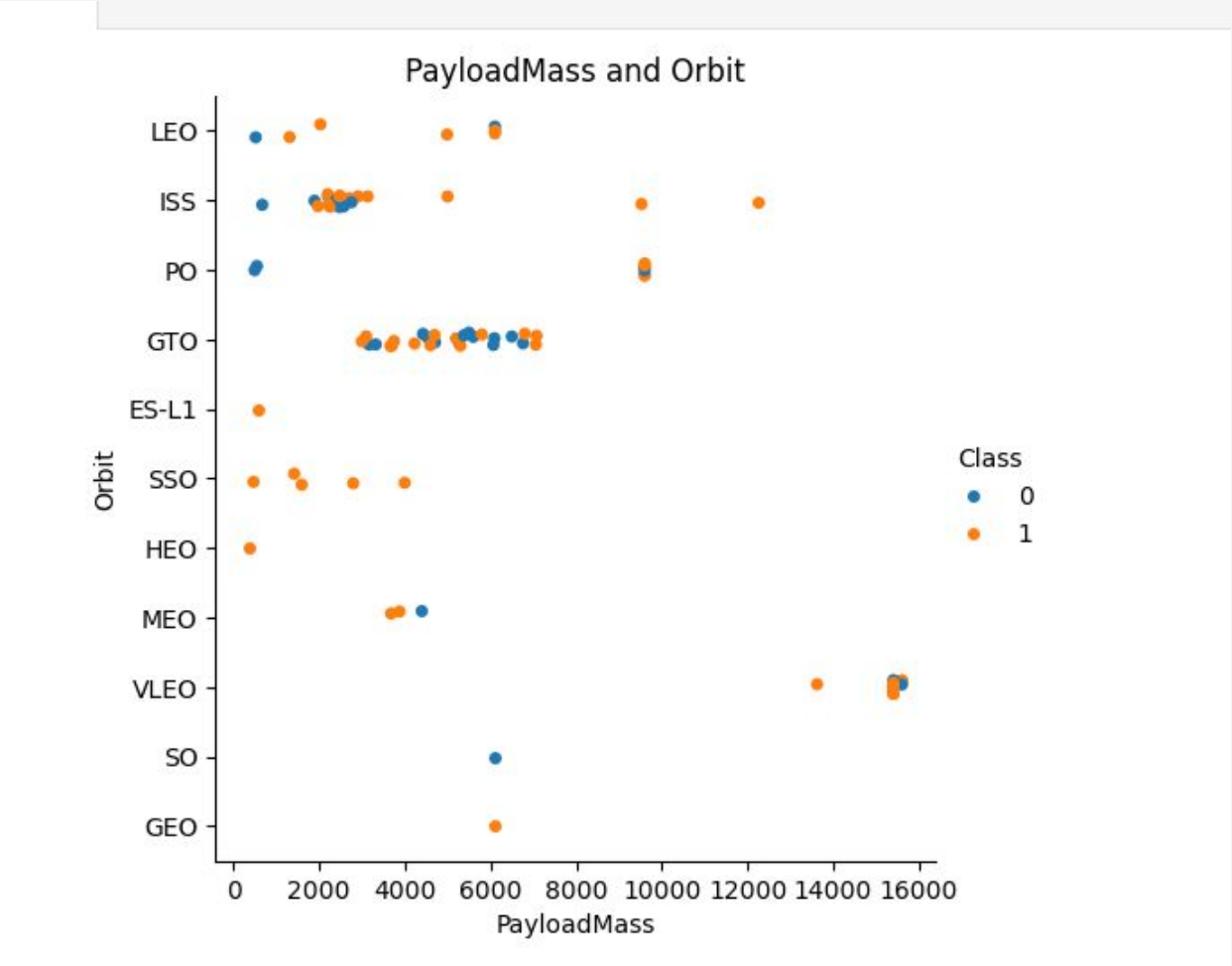
- Bar chart for the success rate of each orbit type



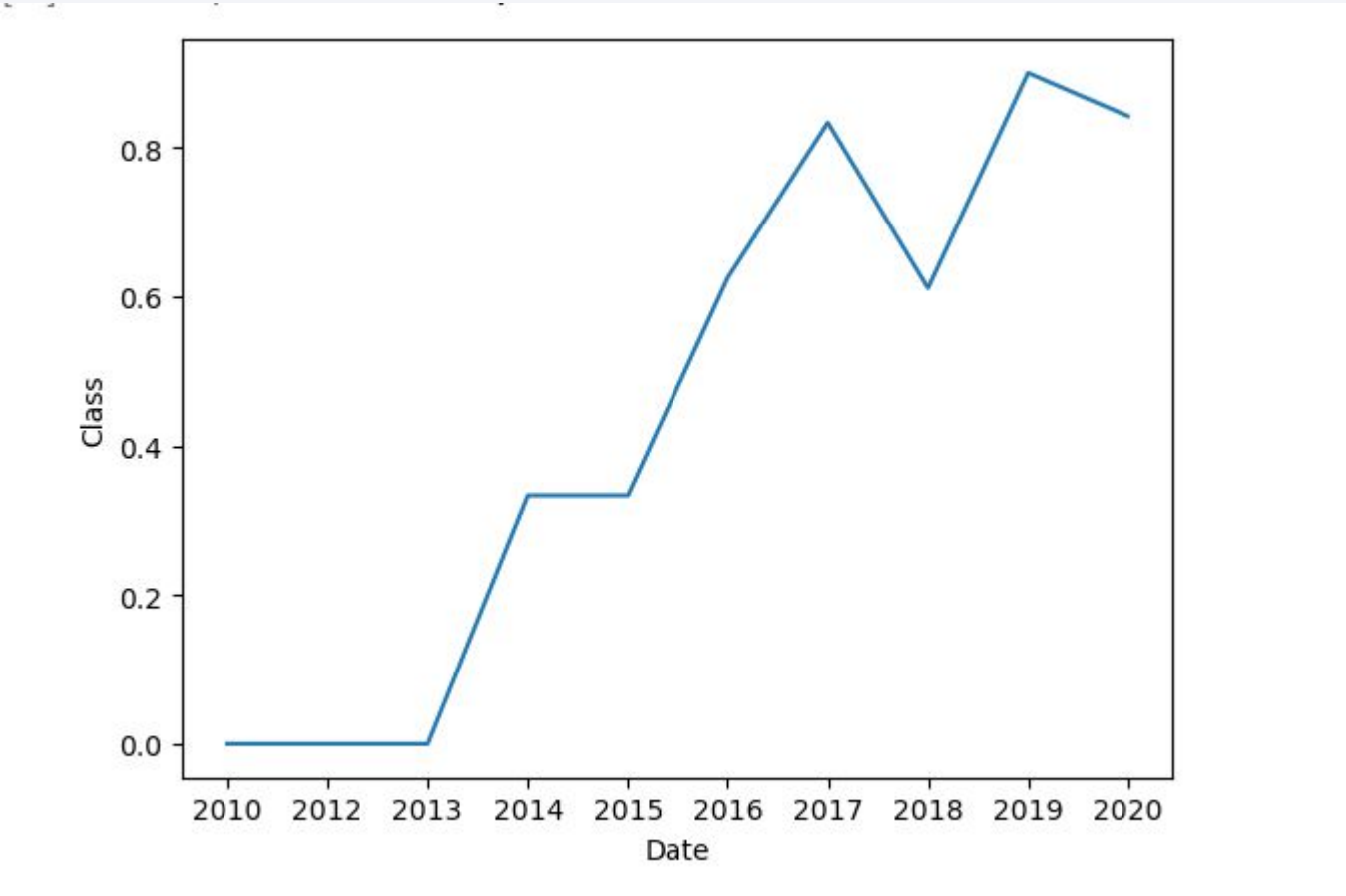
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names

- Names of the unique launch sites

```
%sql select DISTINCT "Launch_Site" from SPACEXTBL
```

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

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

Launch Site Names 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 attemp
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attemp
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attemp

Total Payload Mass

- Total payload carried by boosters from NASA

```
%sql select sum("PAYLOAD_MASS__KG_") from SPACEXTBL where Customer is "NASA (CRS)"
```

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

sum("PAYLOAD_MASS__KG_")
45596

Average Payload Mass by F9 v1.1

- Average payload mass carried by booster version F9 v1.1

```
%sql select avg("PAYLOAD_MASS__KG_") from SPACEXTBL where Booster_Version is "F9 v1.1"
```

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

avg("PAYLOAD_MASS__KG_")
2928.4

First Successful Ground Landing Date

- Dates of the first successful landing outcome on ground pad

```
%sql select min(Date) from SPACEXTBL where Landing_Outcome is "Success (ground pad)"
```

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

min(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%%sql select Booster_Version from SPACEXTBL
WHERE Landing_Outcome is "Success (drone ship)"
AND PAYLOAD_MASS__KG_ between 4000 AND 6000
```

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

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

Total Number of Successful and Failure Mission Outcomes

- Total number of successful and failure mission outcomes

```
%%sql SELECT Mission_Outcome, COUNT(*) AS count
FROM SPACEXTBL
GROUP BY Mission_Outcome
ORDER BY count DESC;
```

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

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

Boosters Carried Maximum Payload

```
%%sql select Booster_version,  PAYLOAD_MASS__KG_ from SPACEXTBL
    where PAYLOAD_MASS__KG_ = ( select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

```
%%sql select substr(Date, 6,2) as month, substr(Date,0,5) as year,
    Booster_Version, Launch_Site, Landing_Outcome from SPACEXTBL
WHERE Landing_Outcome = 'Failure (drone ship)'
```

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

month	year	Booster_Version	Launch_Site	Landing_Outcome
01	2015	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)
01	2016	F9 v1.1 B1017	VAFB SLC-4E	Failure (drone ship)
03	2016	F9 FT B1020	CCAFS LC-40	Failure (drone ship)
06	2016	F9 FT B1024	CCAFS LC-40	Failure (drone ship)

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

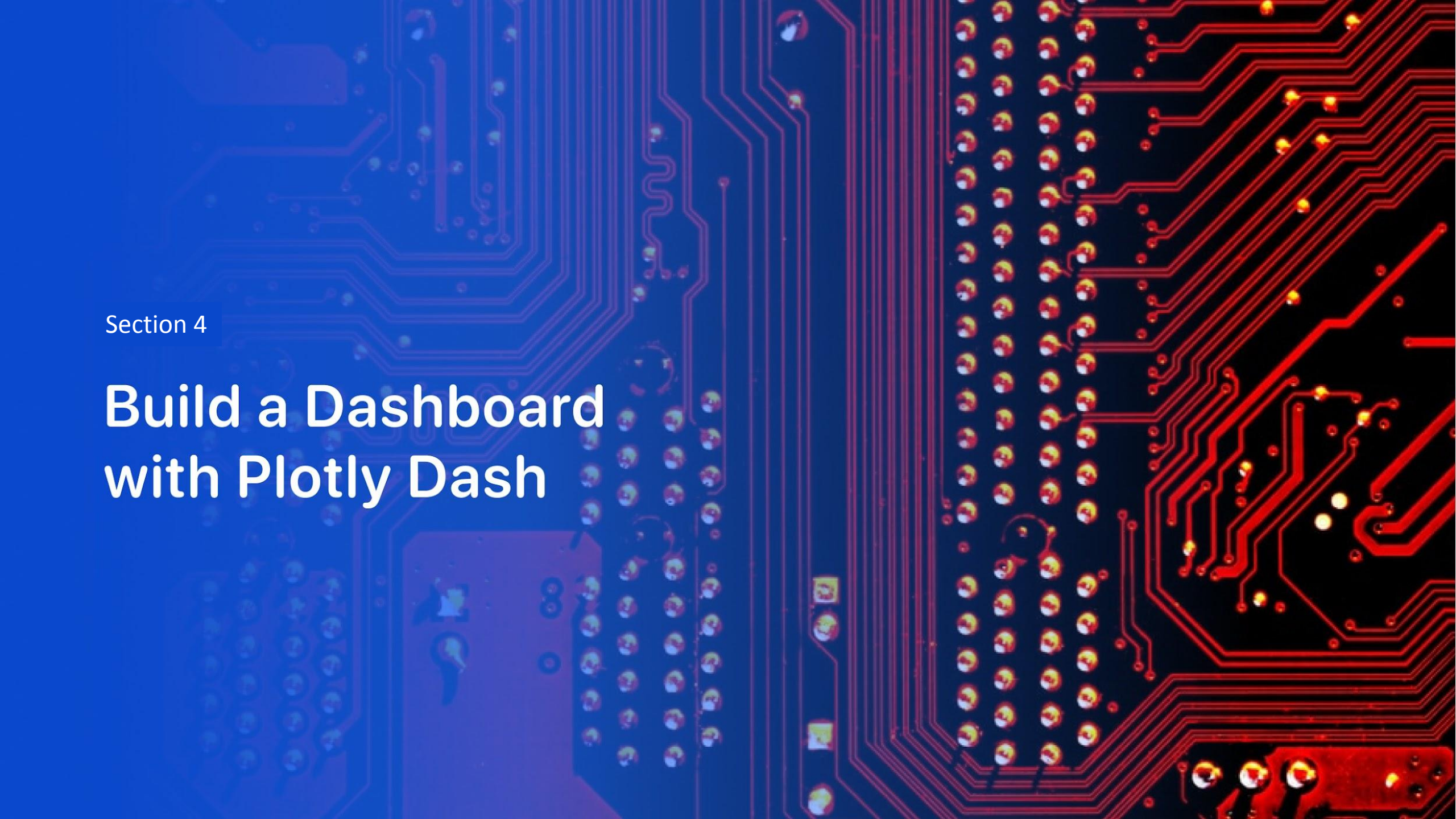
```
%%sql select Date, Landing_Outcome, count(*) as Count from SPACEXTBL
group by Landing_Outcome
order by Count desc
```

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

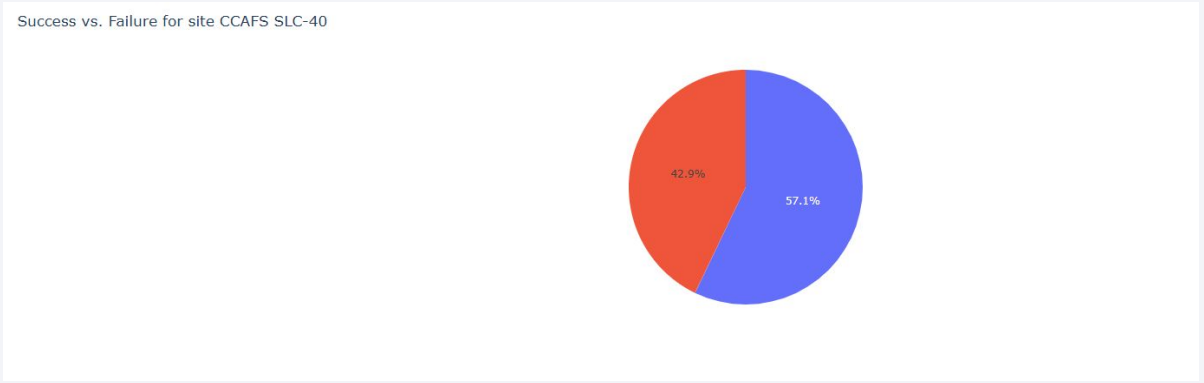
Date	Landing_Outcome	Count
2018-07-22	Success	38
2012-05-22	No attempt	21
2016-04-08	Success (drone ship)	14
2015-12-22	Success (ground pad)	9
2015-01-10	Failure (drone ship)	5
2014-04-18	Controlled (ocean)	5
2018-12-05	Failure	3
2013-09-29	Uncontrolled (ocean)	2
2010-06-04	Failure (parachute)	2
2015-06-28	Precluded (drone ship)	1
2019-08-06	No attempt	1


Section 3
Launch Sites
Proximities Analysis

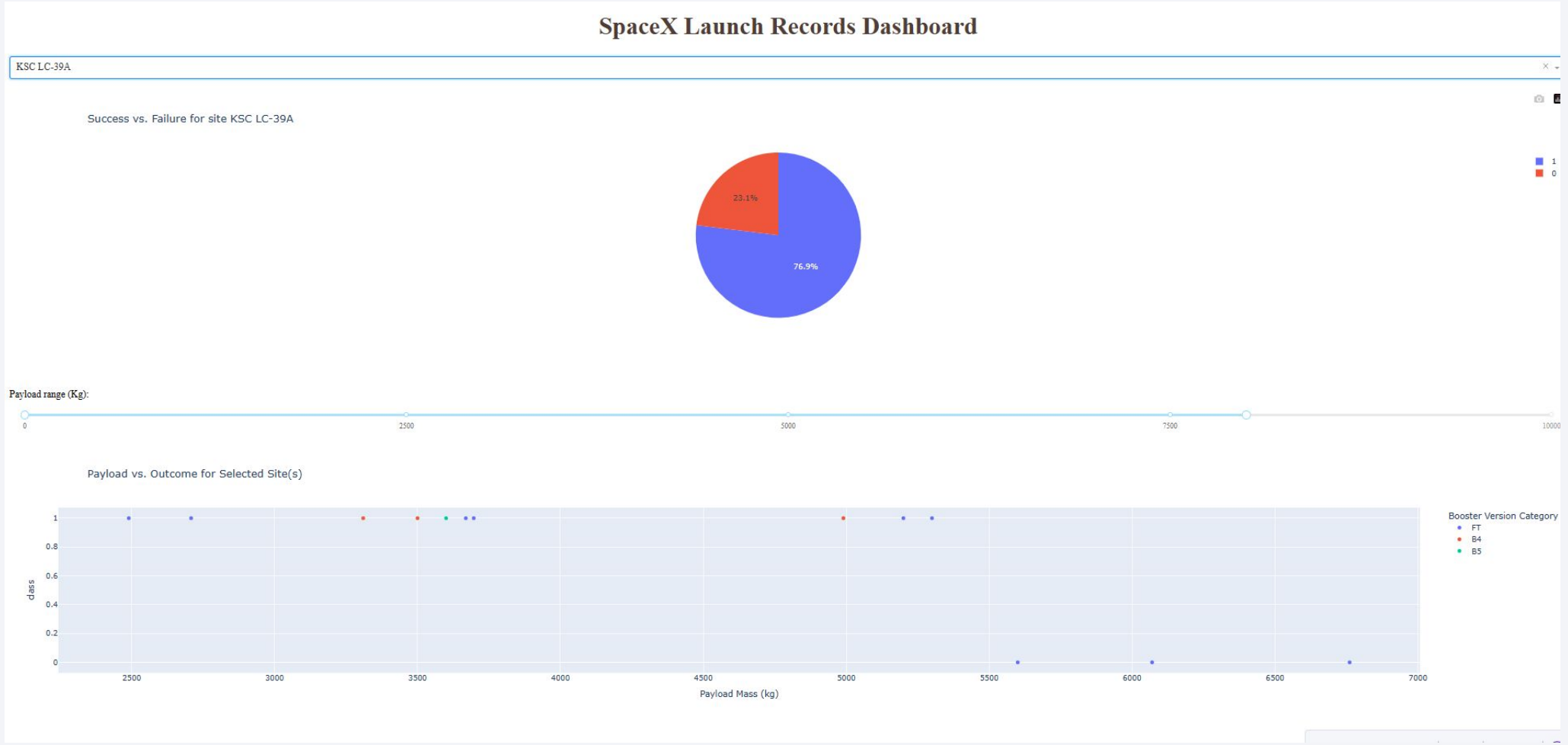
<Folium Map Screenshot 1>


Section 4
Build a Dashboard with Plotly Dash

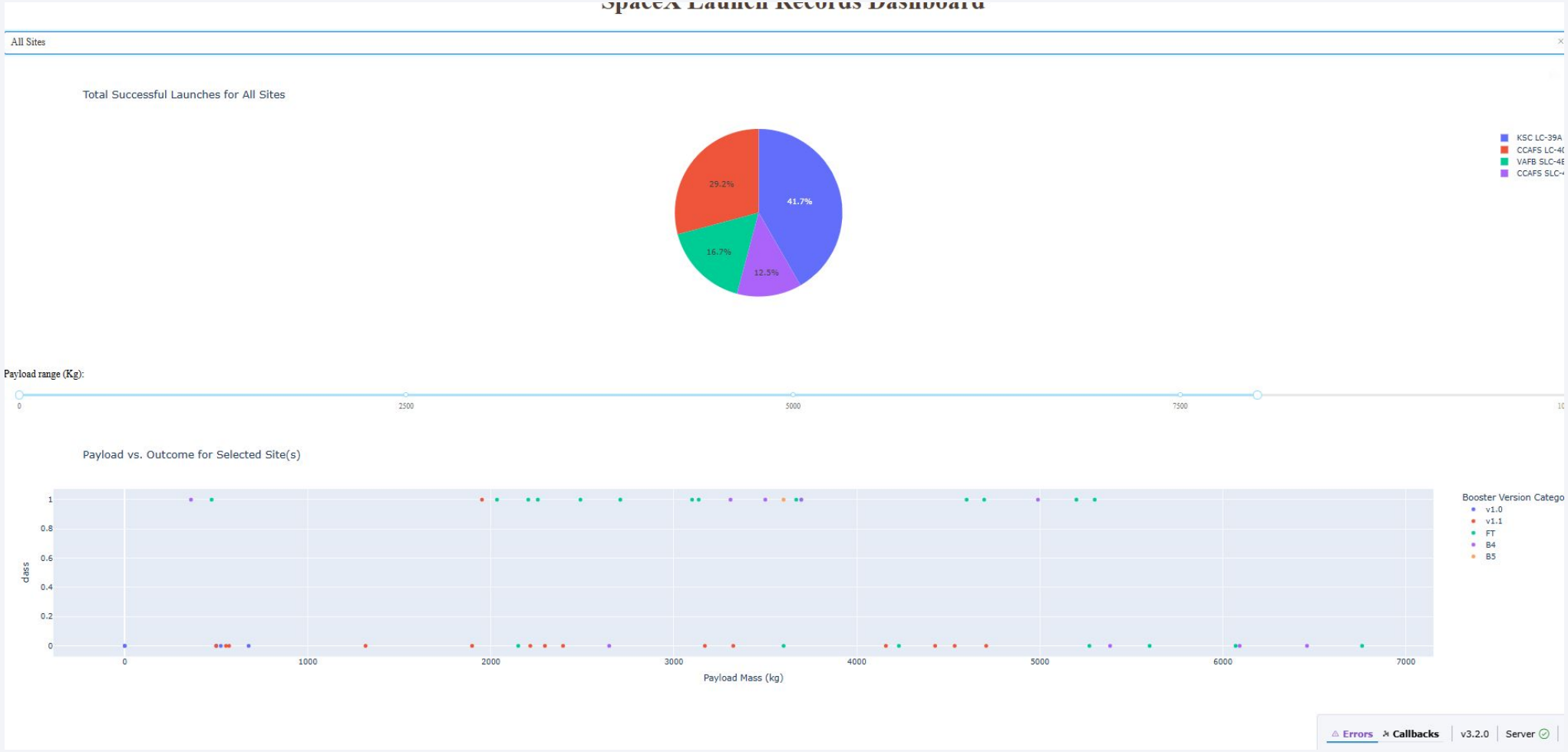
List of Launch sites



Highest success rate site



All sites


All Sites
Total Successful Launches for All Sites
29.2%
41.7%
16.7%
12.5%
KSC LC-39A
Payload range (Kg):
0
2500
5000
7500
Payload vs. Outcome for Selected Site(s)
class
Payload Mass (kg)
Booster Version Catego
v1.0
v1.1
FT
B4
B5
Errors
Callbacks
v3.2.0
Server

Section 5

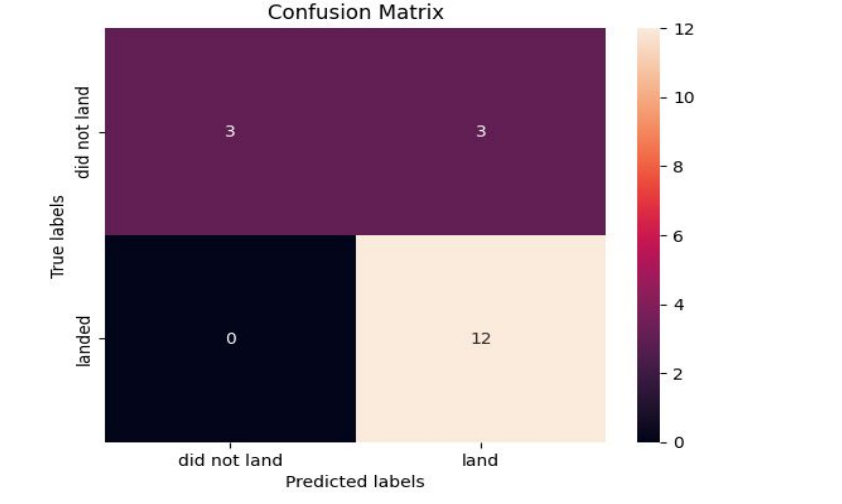
Predictive Analysis (Classification)

Classification Accuracy

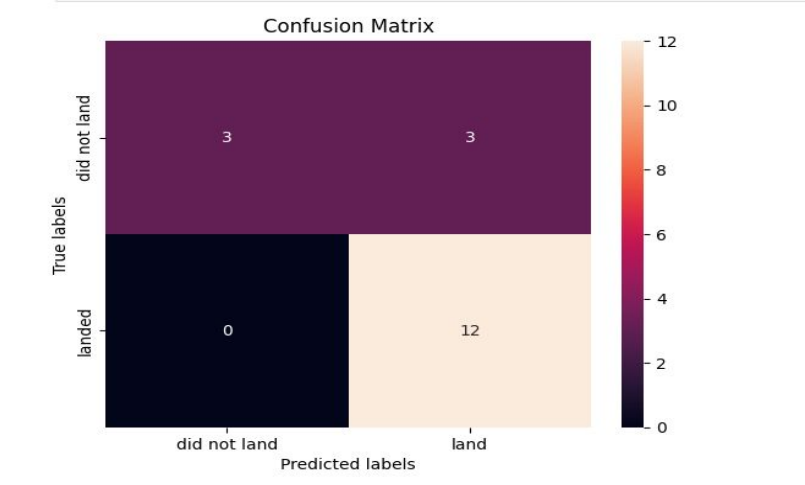
Model	Accuracy
Logistic Regression	0.8464
SVM	0.84821
Decision tree classifier	0.6111
KNN	0.8482

Confusion Matrix

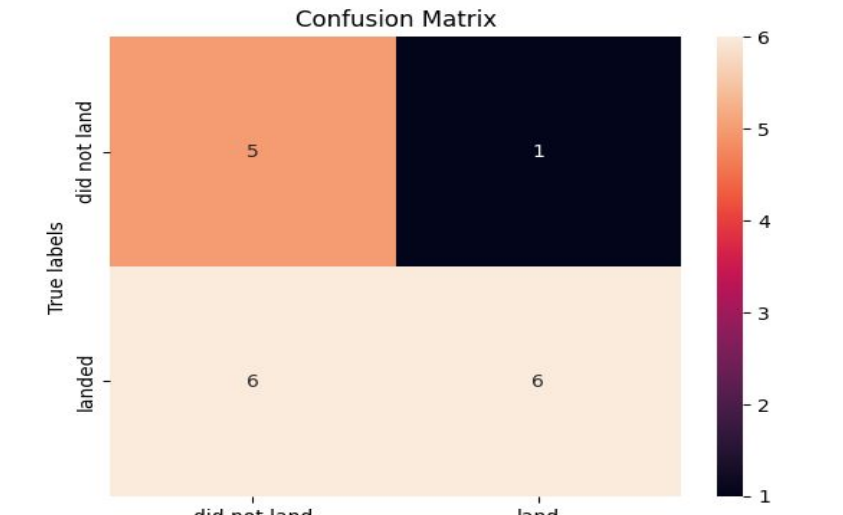
Logistic Regression



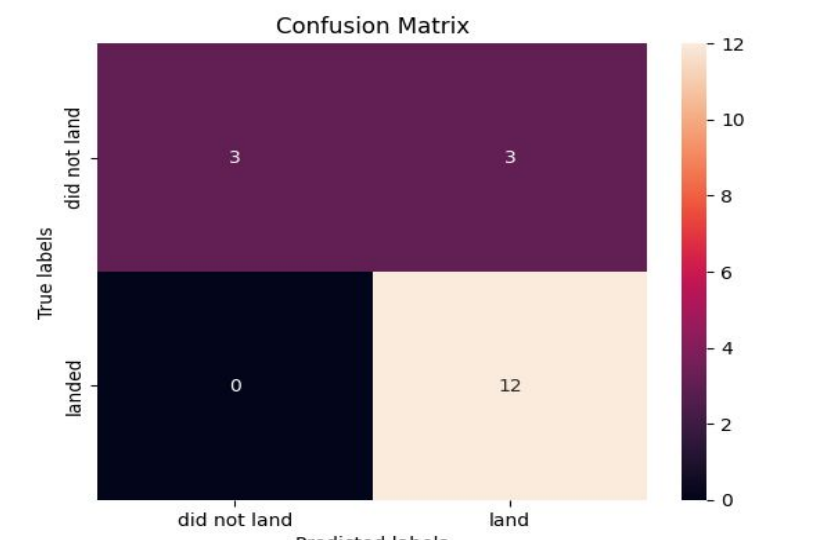
SVM



Tree classifier



KNN classifier



Conclusions

- Success rate for ES-L1, GEO, HEO, SSO is high.
- Success rate for SpaceX rockets is increasing since 2013.
- KNN is best for predicting success rate.

Appendix

- Interactive Plotly.
- SQL magic
- Interactive maps with folium

Thank you!