IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Name>
<Date>

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling (Data Cleaning)
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Dash and Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive Analyctis
  - Predictive Analystics result

Introduction

- Project background and context:

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

- Problems you want to find answers:

If we can determinate if the first stage will land, we can determinate the cost of a launch.

Section 1

Methodology

Methodology

Executive Summary

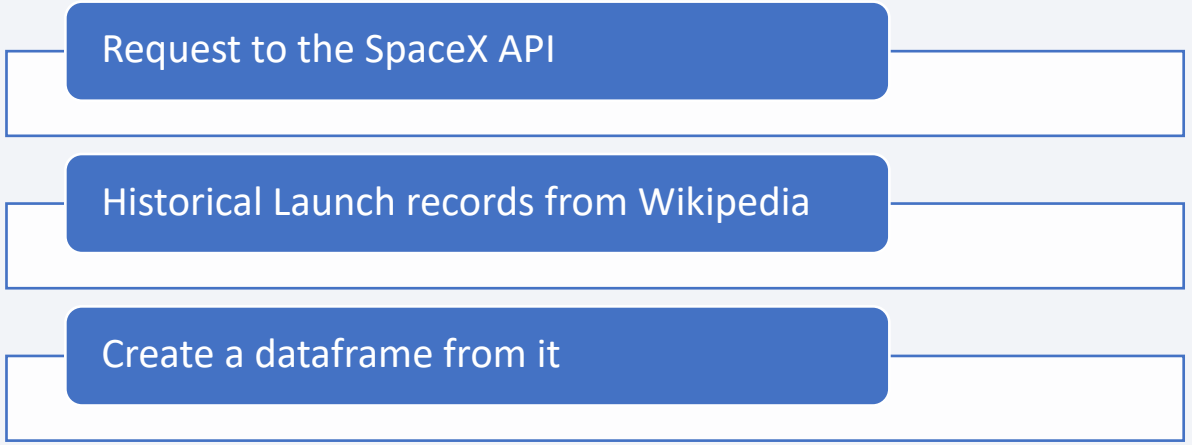
- Data collection methodology:
  - Request to the SpaceX API and Web Scraping
- Perform data wrangling
  - Data cleaning
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected:

From: https://api.spacexdata.com/v4/rockets/

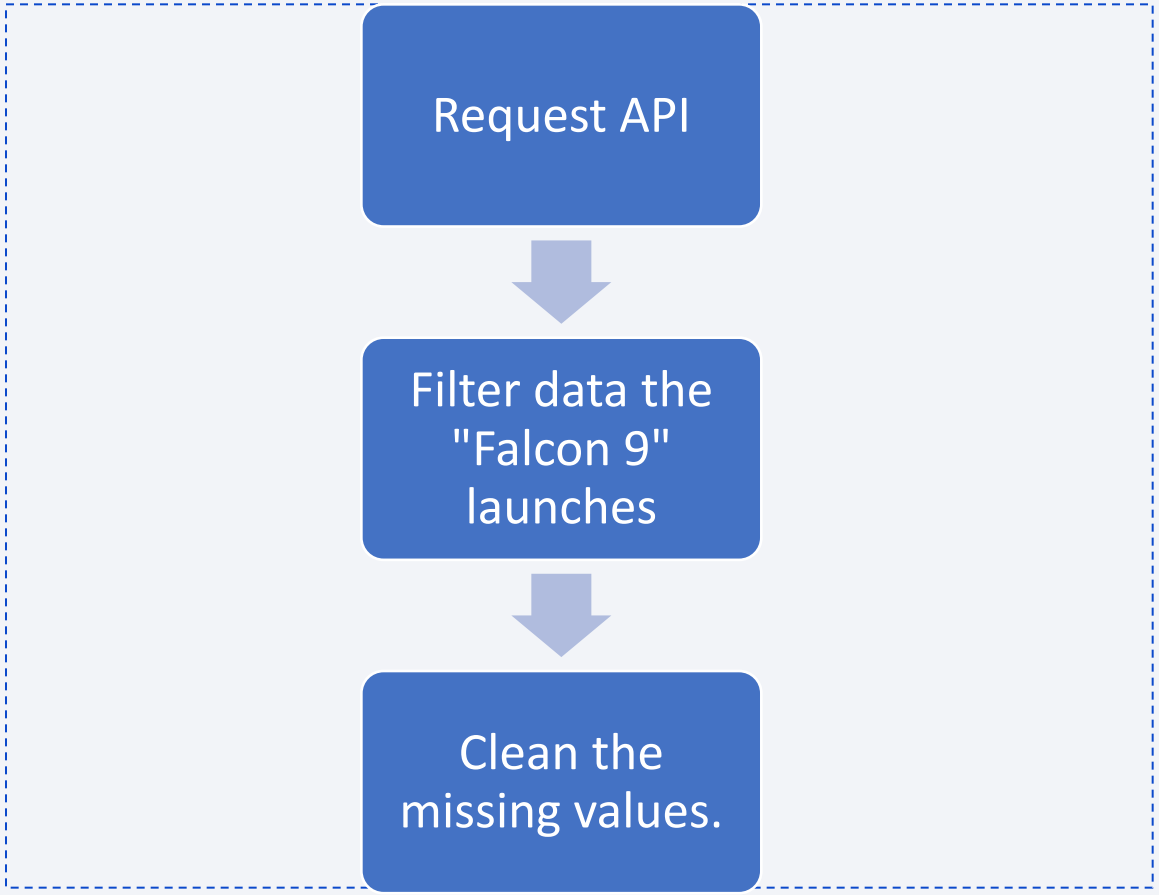
From https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches



Data Collection – SpaceX API

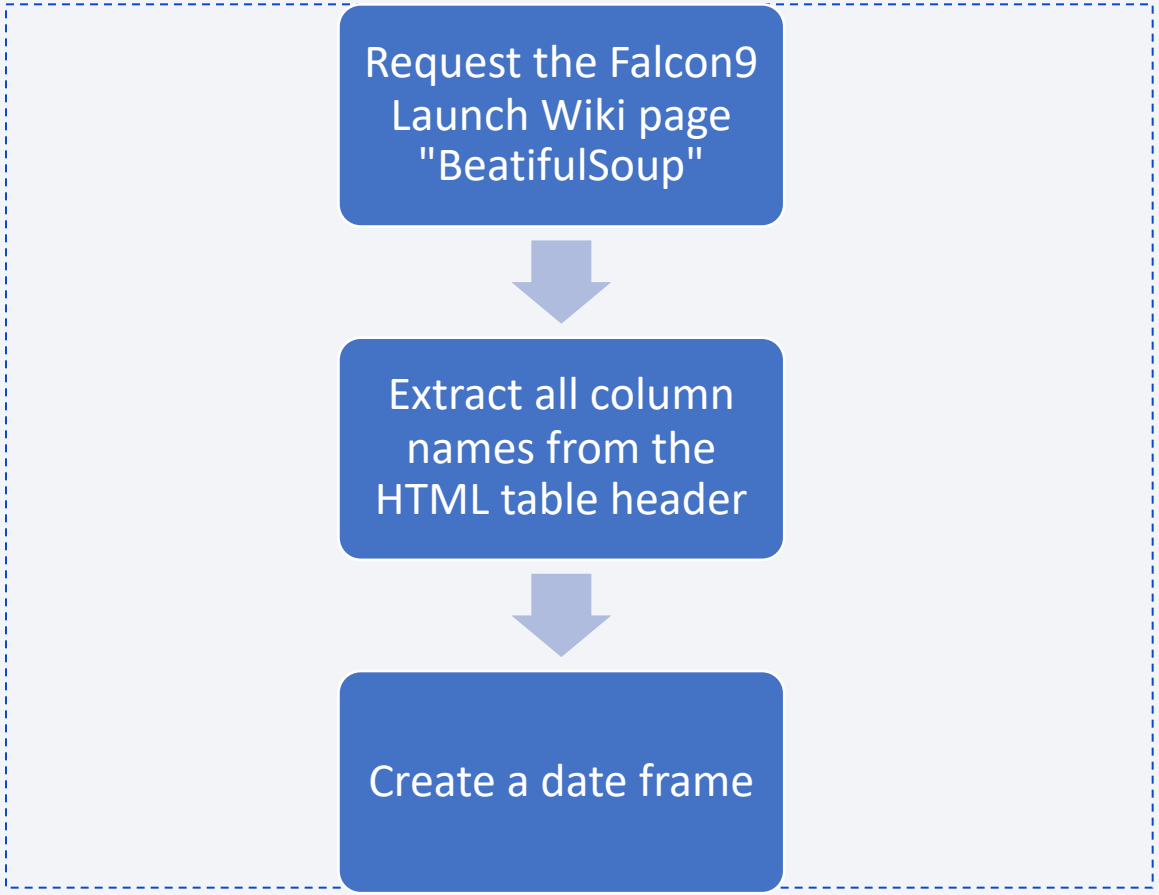
- Space X have a API from where data can be obtained.

- Source Code: https://github.com/IlkaQ02/Cursos/blob/main/Week%201%20-%202.jupyter-labs-webscraping.ipynb



Data Collection - Scraping

- Request the Falcon9 form URL.
- Create BeatifulSoup object from HTML response.
- Source Code: https://github.com/IlkaQ02/Cursos/blob/main/Week%201%20-%202.jupyter-labs-webscraping.ipynb



Data Wrangling

- Perform exploratory Data Analysis and determine Training Labels

- Source Code:

https://github.com/IlkaQ02/Cursos/blob/main/Week%201%20-%203.labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with SQL

- SQL queries to solve the assignment tasks, to find out for instance:
  - Names of unique launch sites.
  - Total payload mass carried by boosters
  - The average payload mass carried by booster version F9 V1.1
  - Number of successful and failure missin outcomes
  - Failed landing outcomes in drone ship, booster and launch site names.
- Code Source: https://github.com/IlkaQ02/Cursos/blob/main/Week%202%20-%201.jupyter-labs-eda-sql-coursera_sqllite.ipynb

EDA with Data Visualization

- Exploratoring and Preparing Data
  - Scatter plot FlightNumber vs PayloadMass
  - Scatter plot FlightNumer vs Launch Site
  - Scatter plot Pay Load Mass (kg) vs Launch Site
  - Bar Chart Orbit vs Class
  - Scatter Flight Number vs Orbit
  - Scatter plot Payload vs Orbit
- Code Source: https://github.com/IlkaQ02/Cursos/blob/main/Week%202.%202.edadataviz.ipynb

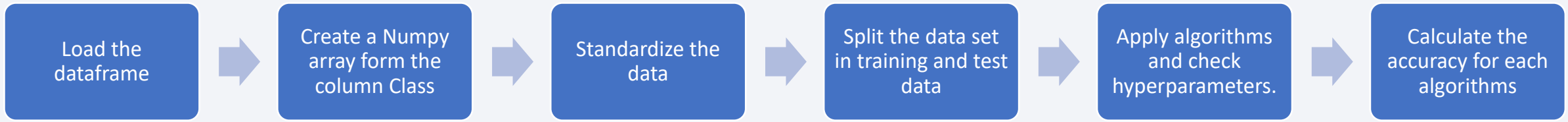
Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
  - Markers indicate points like launch sites
  - Circles indicate areas for specific coordinates
  - Markers clusters are groups of events
  - Linea are the distance between two coordinates.
- Code Source: https://github.com/llkaQ02/Cursos/blob/main/Week%203%20-%201.lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- Code Source: https://github.com/IlkaQ02/Cursos/blob/main/Week%203%20-%203.spacex_dash_app.py

Predictive Analysis (Classification)

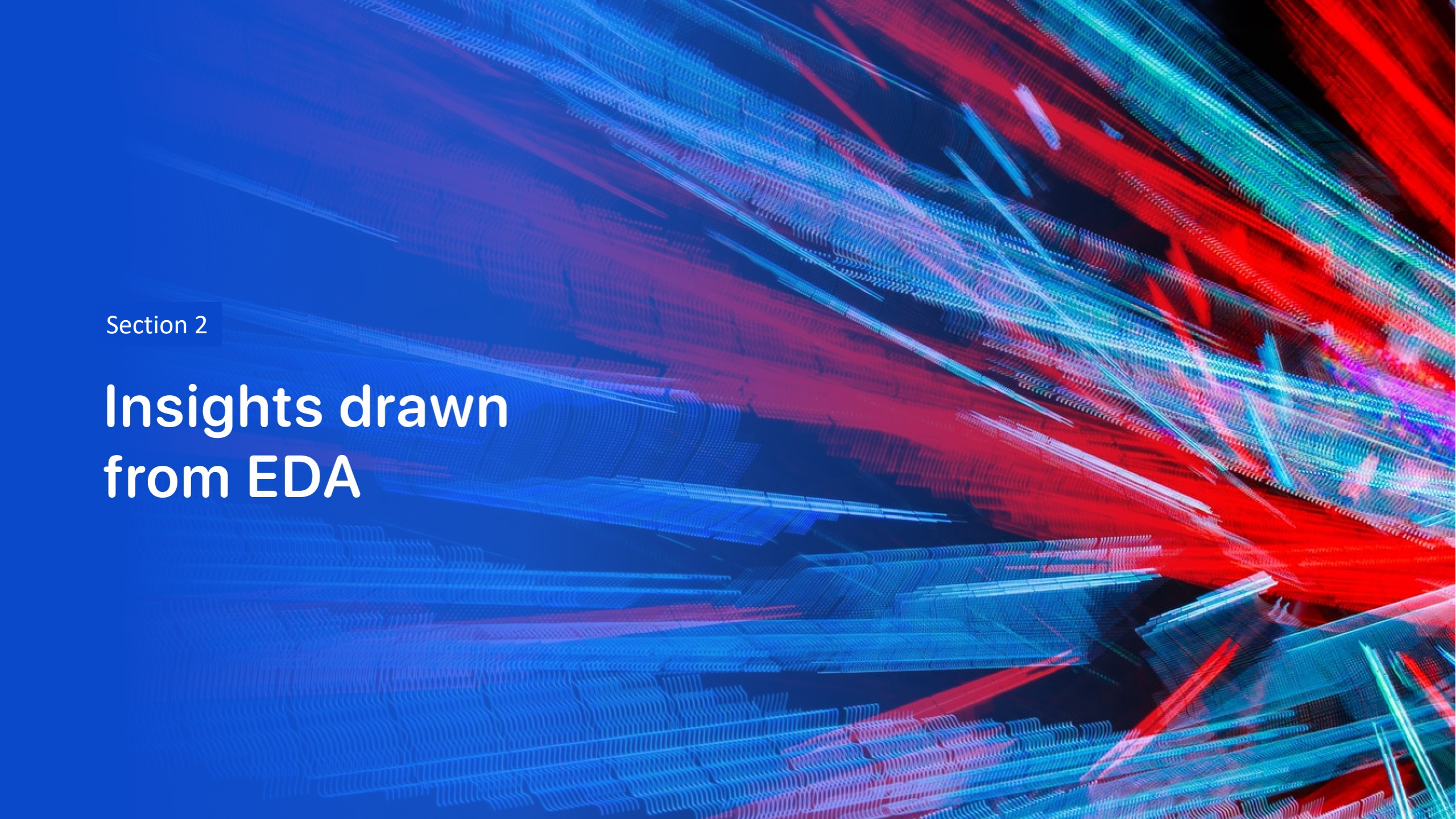


Select the best algorithms. The more importante is select the model with more interpretability

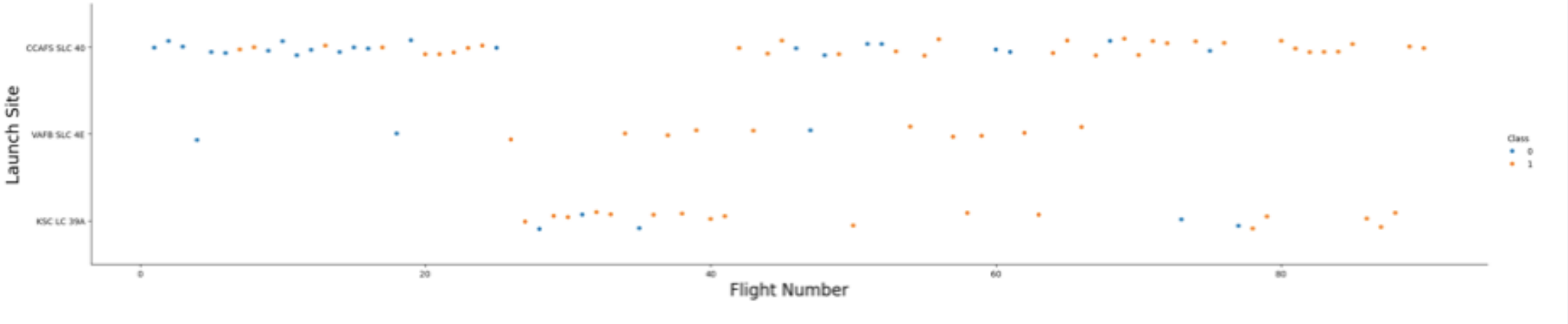
- Code Source: https://github.com/IlkaQ02/Cursos/blob/main/Week%204%20-%20SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

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

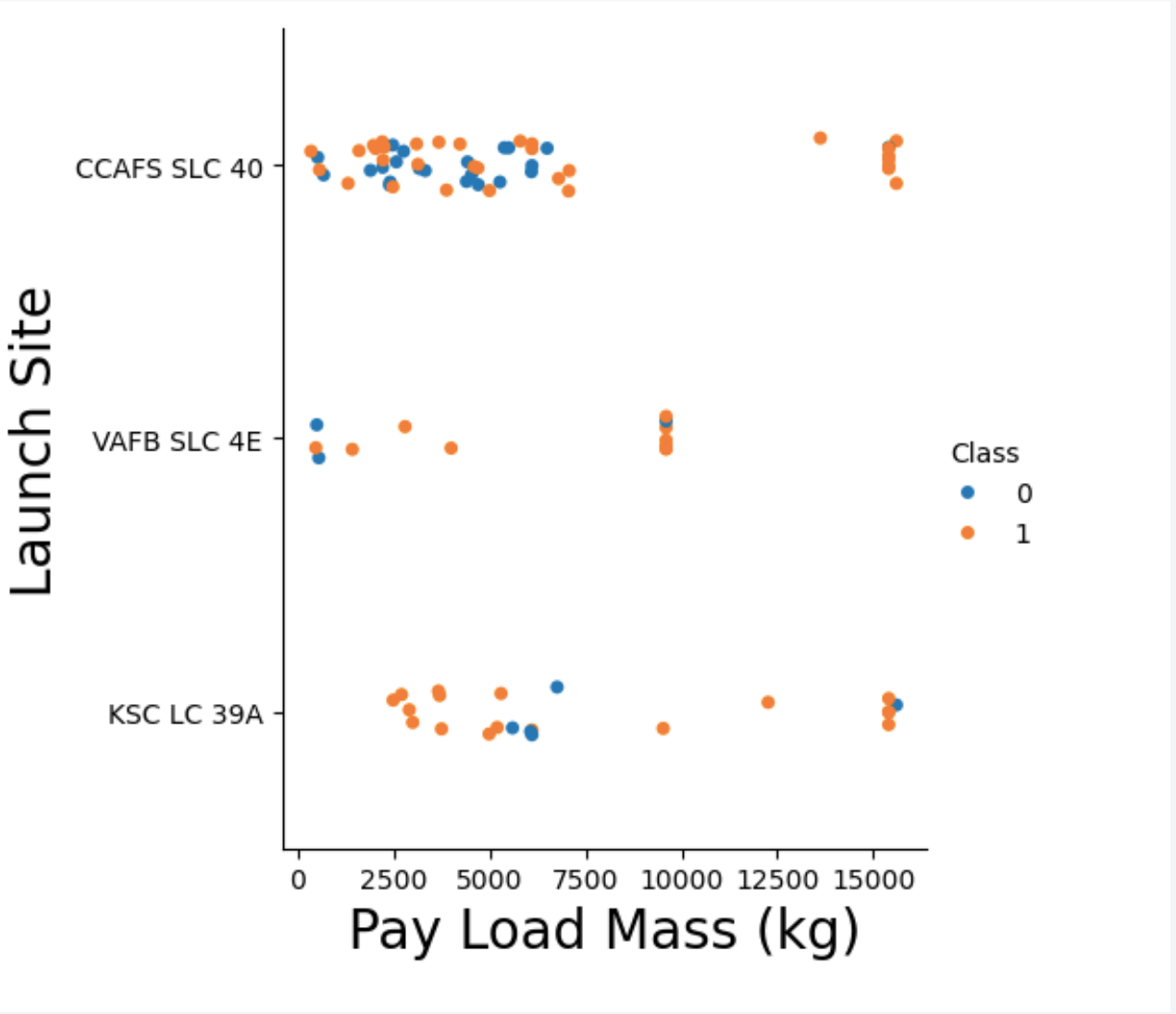

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



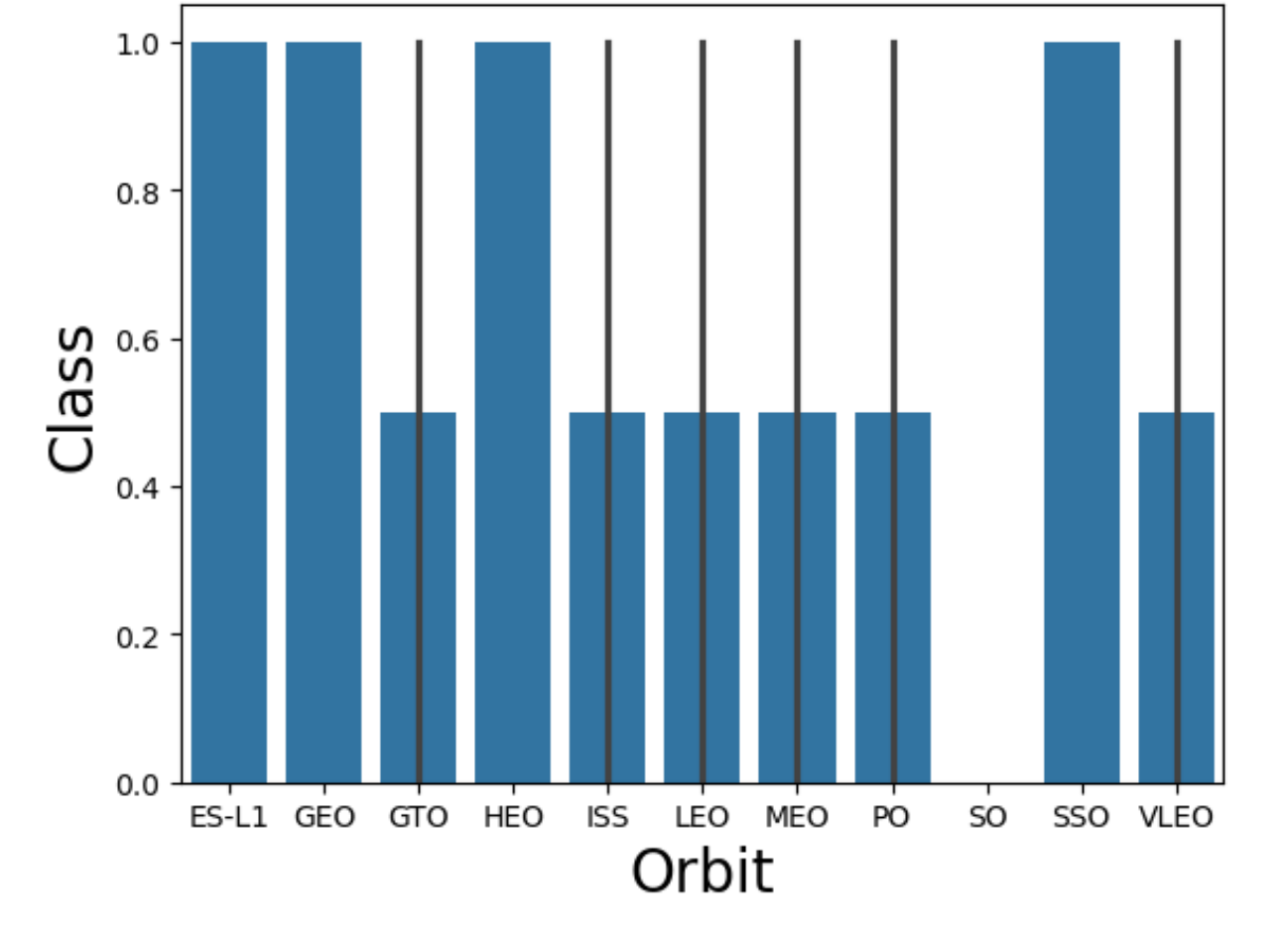
CCAFS SLC 40 is the more used Launch Site.

Payload vs. Launch Site



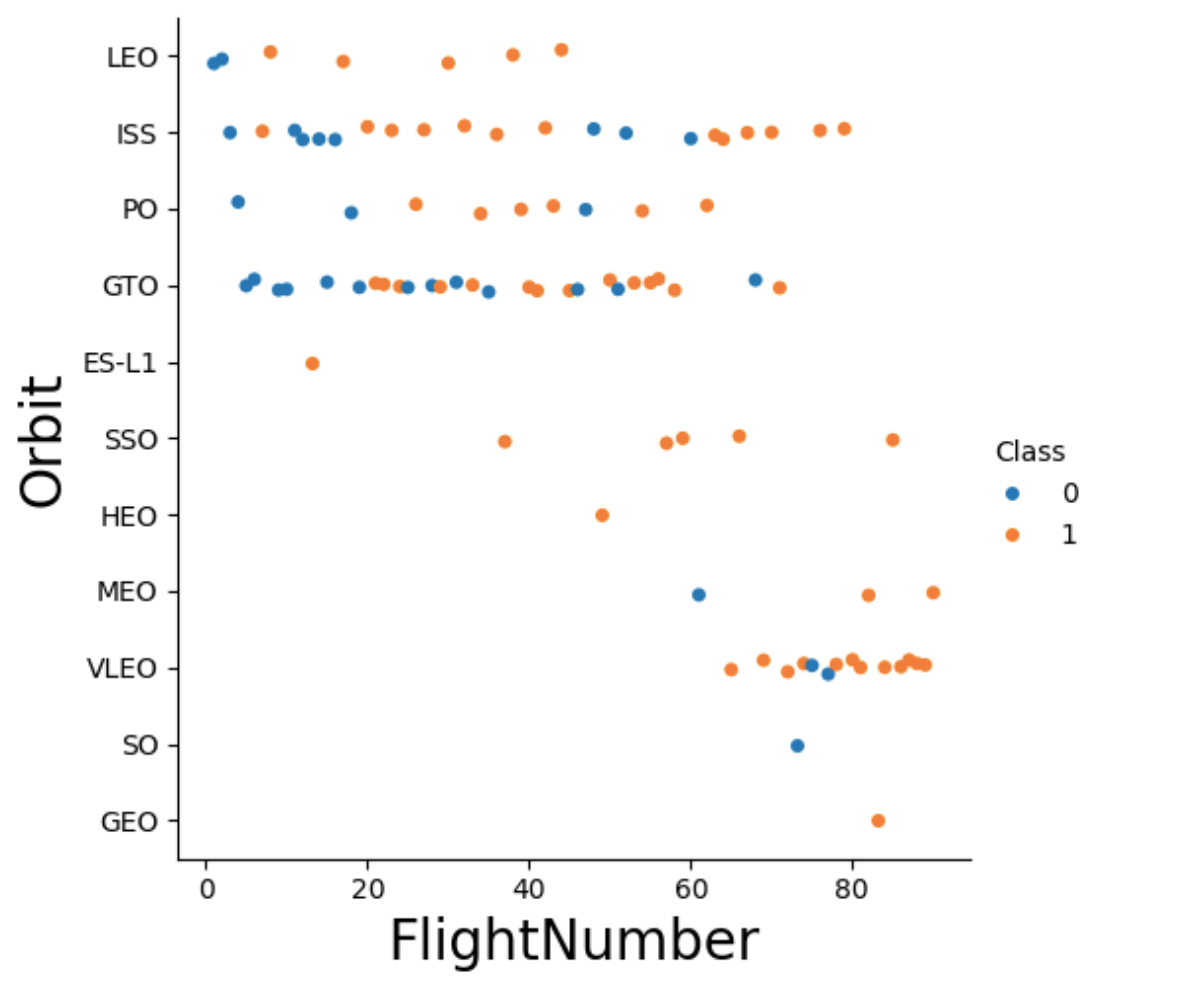
- Launches with heavier payloads have more success, and the success rate at CCAFS SLC 40 with heavier payloads is 100%.

Success Rate vs. Orbit Type



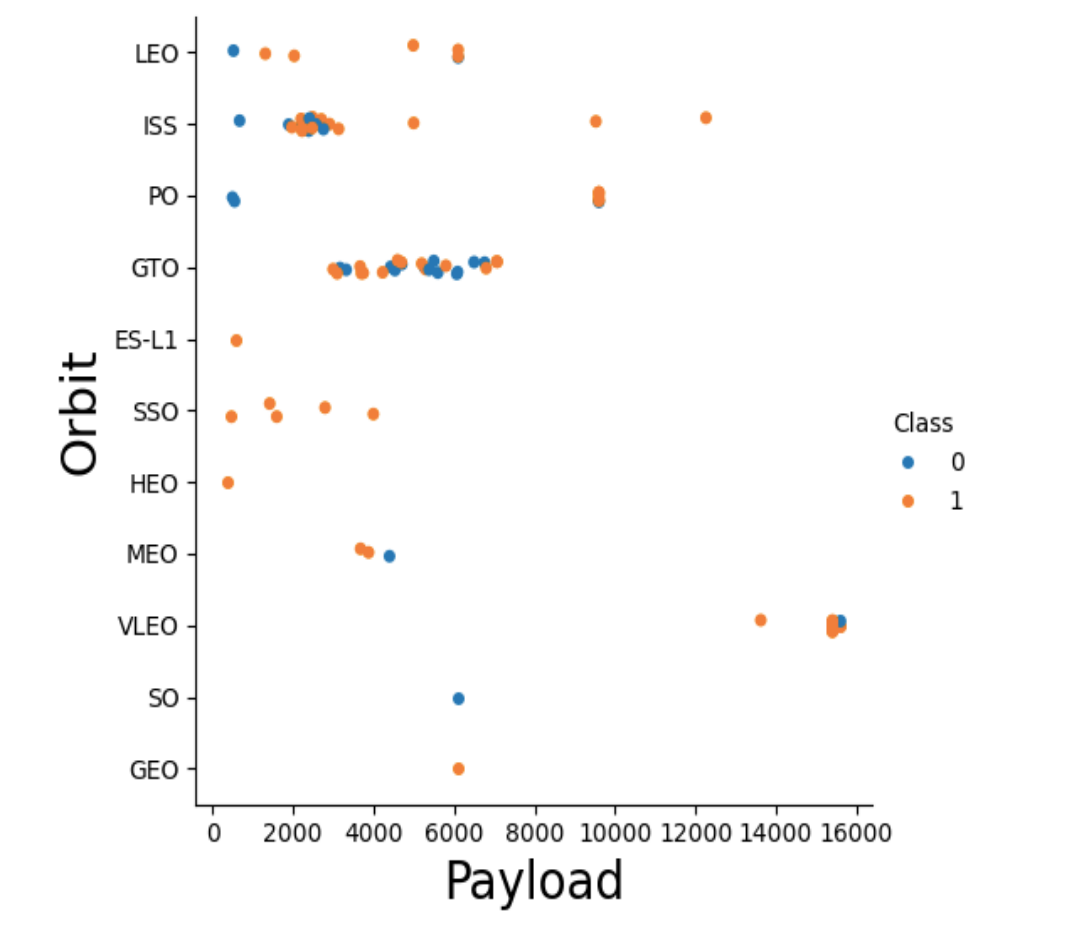
- ES-L1, GEO, HEO and SSD had the most success rate.

Flight Number vs. Orbit Type



- Show the screenshot of the scatter plot with explanations

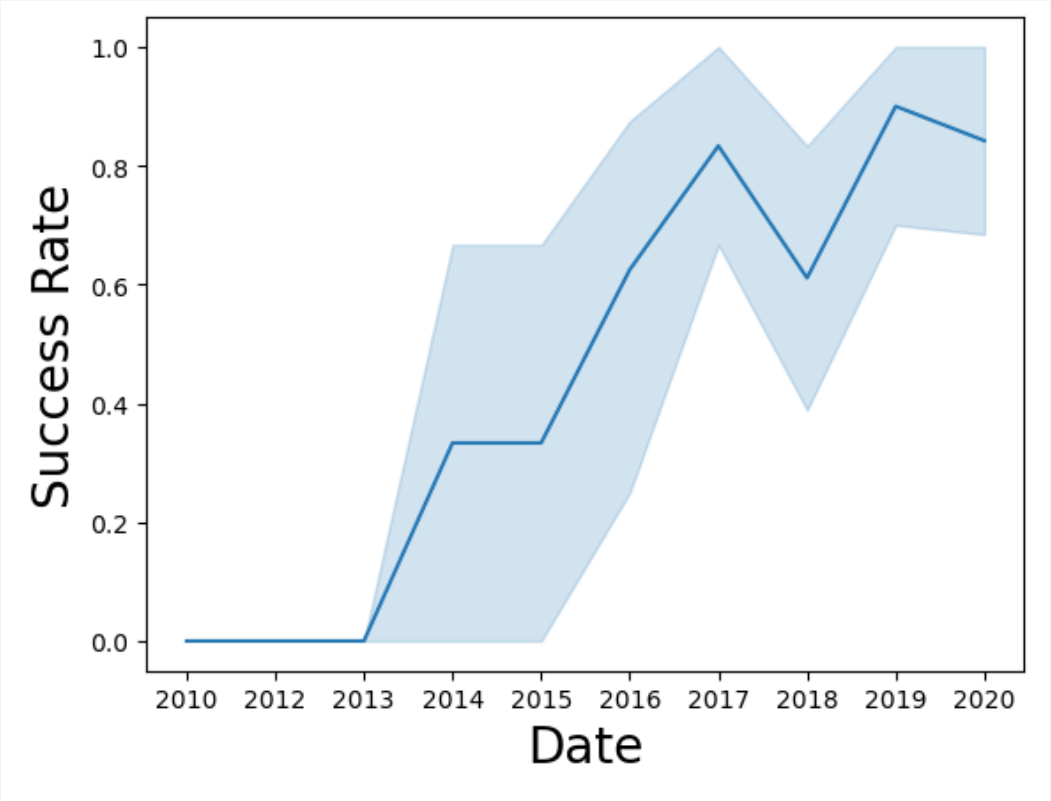
Payload vs. Orbit Type



- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

Launch Success Yearly Trend

- Since 2013, success has increased, but in 2018, there is a decrease in launch success. Following 2018, there are attempts to increase the success rate, but it does not return to the level of success seen in 2017



All Launch Site Names

In [10]:

```
%sql SELECT DISTINCT Launch_Site from SPACEXTBL LIMIT 5
```

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

Out[10]:

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 [12]:

```
%sql SELECT DISTINCT Launch_Site FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%'LI
```

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

Out[12]:

Launch_Site
CCAFS LC-40
CCAFS SLC-40

Total Payload Mass

Task 3

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

In [15]:

```
%sql SELECT SUM (PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS
```

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

Out[15]:

SUM (PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

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

In [35]:

```
%sql SELECT AVG (PAYLOAD_MASS__KG_) from SPACEXTBL WHERE Booster_Version LIK
```

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

Out[35]:

AVG (PAYLOAD_MASS__KG_)
2534.6666666666665

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 [36]:

```
%sql SELECT min (date) from SPACEXTBL WHERE Mission_Outcome = 'Success'
```

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

Out[36]:

min (date)
2010-06-04

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 [44]:

```
%sql SELECT DISTINCT Booster_Version  from SPACEXTBL WHERE Mission_Outcome = '
```

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

Out[44]:

Booster_Version
F9 v1.0 B0003
F9 v1.0 B0004
F9 v1.0 B0005

Total Number of Successful and Failure Mission Outcomes

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 [72]:

```
%%sql sqlite:///my_data1.db
SELECT
    Landing_Outcome,
    COUNT(*) AS Outcome_Count
FROM
    SPACEXTBL
WHERE
    Date BETWEEN '2010-06-04' AND '2017-03-20'
    AND Landing_Outcome IN ('Failure (drone ship)', 'Success (ground pad)')
GROUP BY
    Landing_Outcome
ORDER BY
    Outcome_Count DESC;
```

Done.

Out[72]:

Landing_Outcome	Outcome_Count
Failure (drone ship)	5
Success (ground pad)	3

Boosters Carried Maximum Payload

Task 8

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

65]:
```
%%sql sqlite:///my_data1.db


    SELECT MAX (PAYLOAD_MASS__KG_)
    FROM SPACEXTBL
```

Done.

65]:

MAX (PAYLOAD_MASS__KG_)
15600

In [67]:
```
%%sql sqlite:///my_data1.db


    SELECT DISTINCT Booster_Version
    FROM SPACEXTBL
    WHERE PAYLOAD_MASS__KG_ == '15600'
```

Done.

Out[67]:

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

the months and substr(Date,0,5)='2015' for year.

In [71]:

```
%%sql sqlite:///my_data1.db

SELECT
    CASE
        WHEN substr(Date, 6, 2) = '01' THEN 'January'
        WHEN substr(Date, 6, 2) = '02' THEN 'February'
        WHEN substr(Date, 6, 2) = '03' THEN 'March'
        WHEN substr(Date, 6, 2) = '04' THEN 'April'
        WHEN substr(Date, 6, 2) = '05' THEN 'May'
        WHEN substr(Date, 6, 2) = '06' THEN 'June'
        WHEN substr(Date, 6, 2) = '07' THEN 'July'
        WHEN substr(Date, 6, 2) = '08' THEN 'August'
        WHEN substr(Date, 6, 2) = '09' THEN 'September'
        WHEN substr(Date, 6, 2) = '10' THEN 'October'
        WHEN substr(Date, 6, 2) = '11' THEN 'November'
        WHEN substr(Date, 6, 2) = '12' THEN 'December'
        ELSE 'Unknown'
    END AS Month_Name,
    Landing_Outcome,
    Booster_Version,
    Launch_Site,
    Mission_Outcome
FROM SPACEXTBL
WHERE substr(Date, 0, 5) = '2015'
    AND Mission_Outcome LIKE 'Failure%';
```

Done.

Out[71]:

Month_Name	Landing_Outcome	Booster_Version	Launch_Site	Mission_Outcome
June	Precluded (drone ship)	F9 v1.1 B1018	CCAFS LC-40	Failure (in flight)

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

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.

[72]:
```
%%sql sqlite:///my_data1.db
SELECT
    Landing_Outcome,
    COUNT(*) AS Outcome_Count
FROM
    SPACEXTBL
WHERE
    Date BETWEEN '2010-06-04' AND '2017-03-20'
    AND Landing_Outcome IN ('Failure (drone ship)', 'Success (ground pad)')
GROUP BY
    Landing_Outcome
ORDER BY
    Outcome_Count DESC;
```

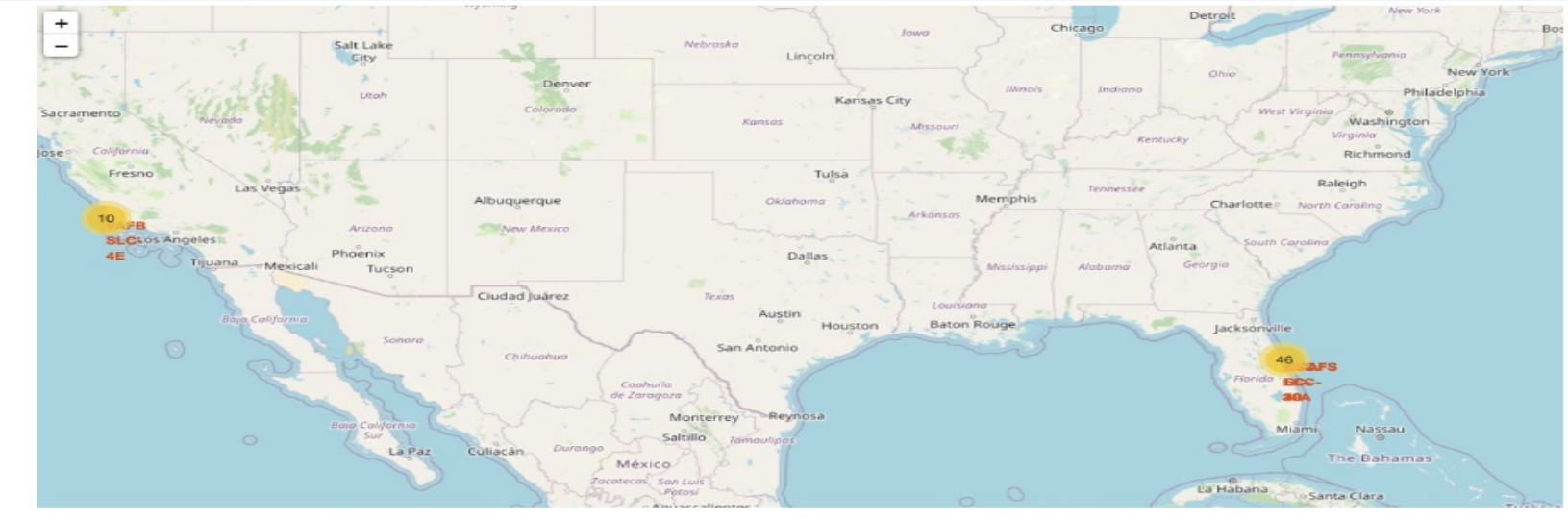
Done.

t[72]:

Landing_Outcome	Outcome_Count
Failure (drone ship)	5
Success (ground pad)	3


Section 3
Launch Sites
Proximities Analysis

LAUNCH RECORDS



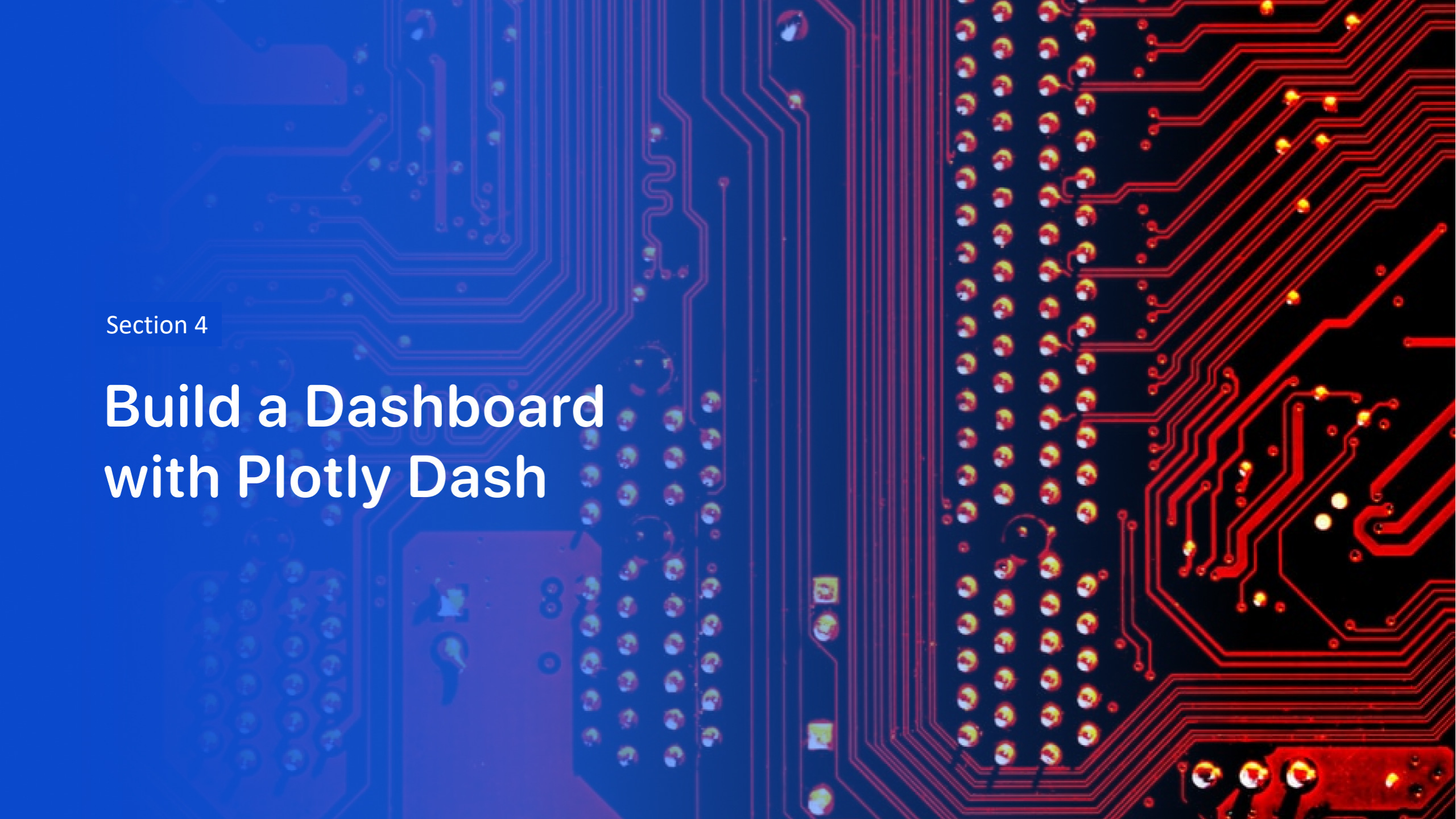
- Note that a launch only happens in one of the four launch sites, which means many launch records will have the exact same coordinate. Marker clusters can be a good way to simplify a map containing many markers having the same coordinate.

LABELED LAUNCHES OUTCOMES

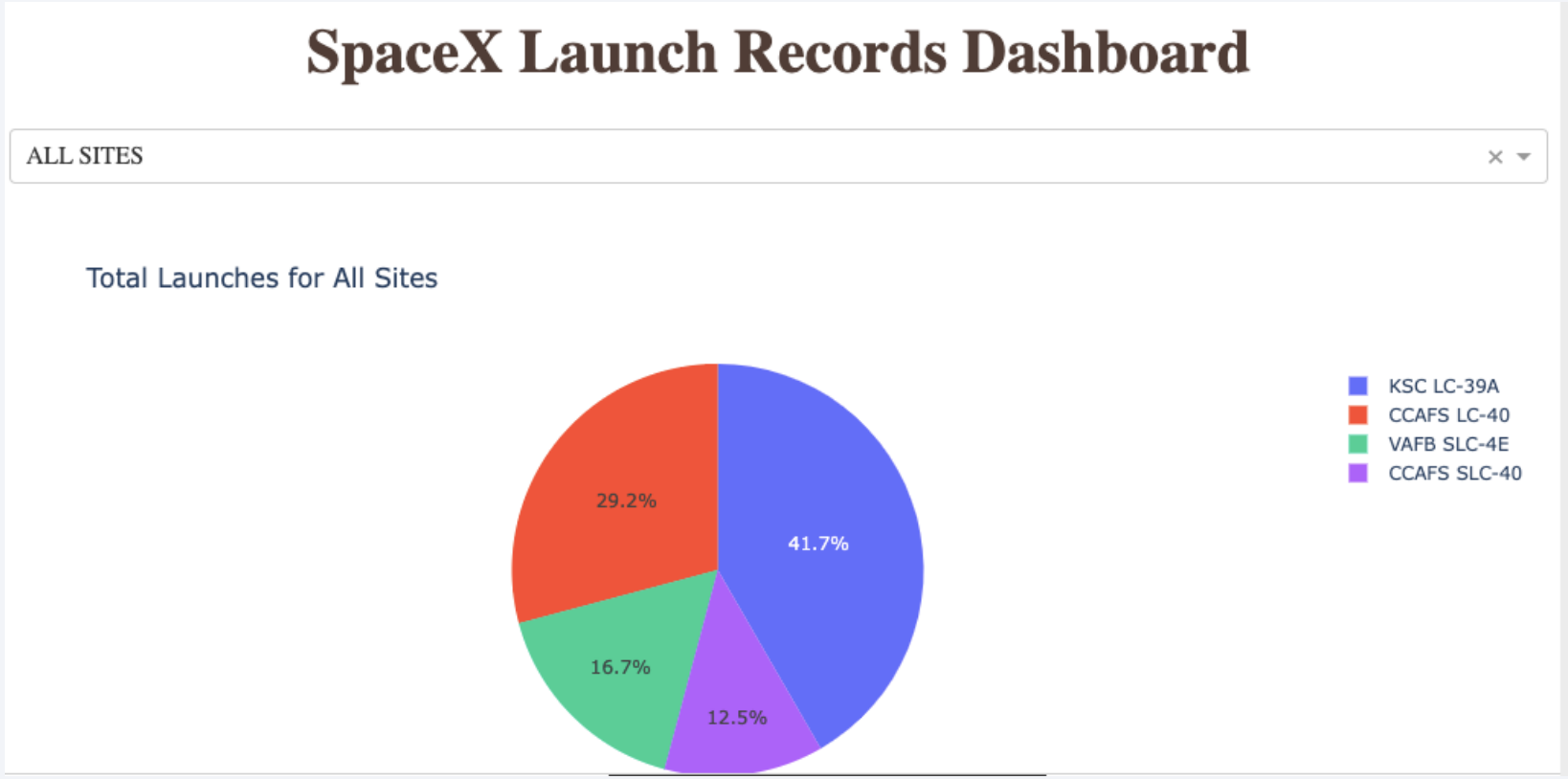
- Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map

Proximities: railway, highway, coastline

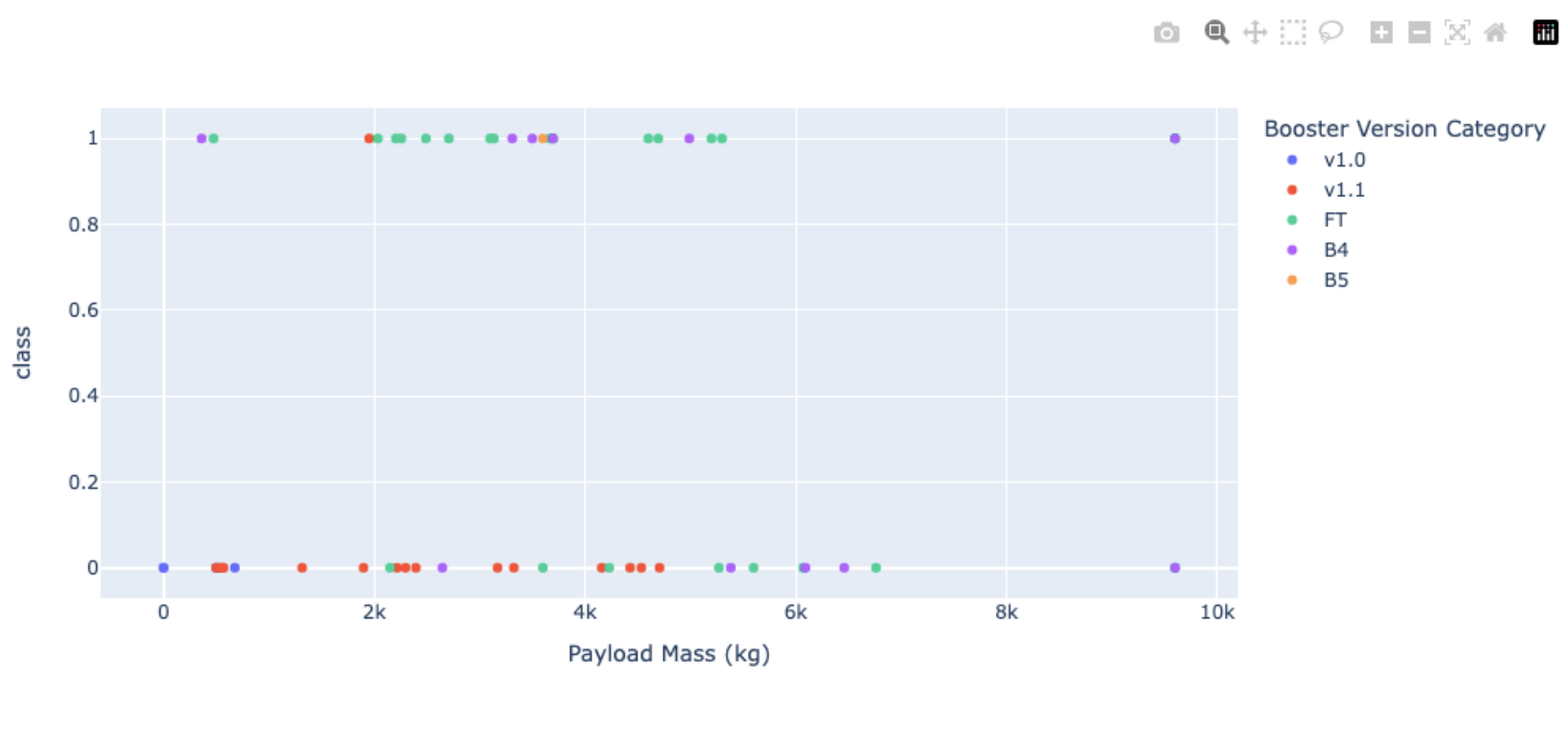
- Close to railway.


Section 4
Build a Dashboard with Plotly Dash

SpaceX Launch Records Dashboard



SpaceX Launch Records Dashboard



Section 5

Predictive Analysis (Classification)

Classification Accuracy

TASK 4

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`.

[3]:
```
parameters ={'C':[0.01,0.1,1],
             'penalty':['l2'],
             'solver':['lbfgs']}
```

[4]:
```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}# l1 lasso l2 ridge
lr=LogisticRegression()
logreg_cv = GridSearchCV(lr, param_grid=parameters,scoring='accuracy', cv=10)
logreg_cv.fit(X_train, Y_train)
logreg_cv.best_params_
```

[4]:
```
{'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
```

We output the `GridSearchCV` object for logistic regression. We display the best parameters using the data attribute `best_params_` and the accuracy on the validation data using the data attribute `best_score_`.

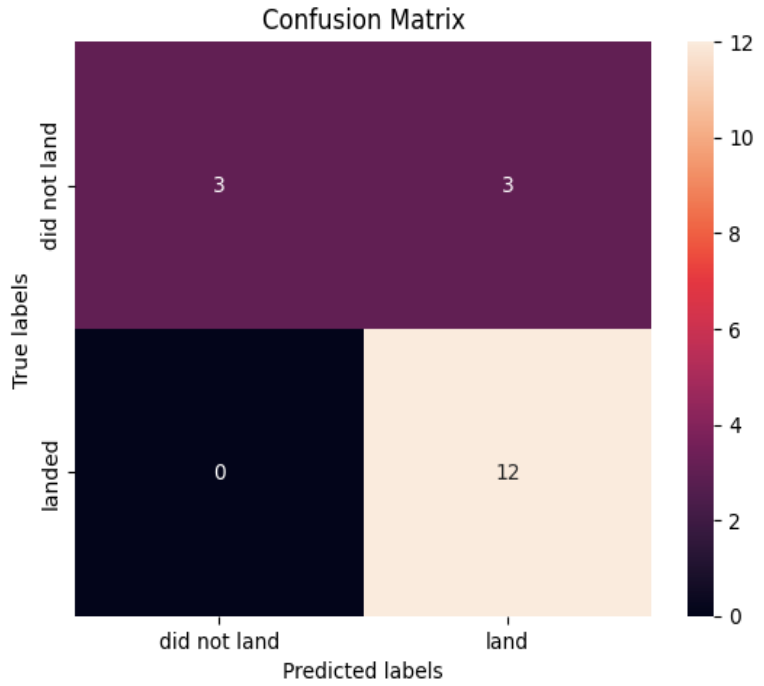
[5]:
```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

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

Confusion Matrix

Lets look at the confusion matrix:

In [17]:
```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



- High precision, except for false positives.

Conclusions

Different algorithms were used, such as logistic regression, support vector machine, decision tree, and KNN. The precision among all the methods was similar.

However, in this case, I would choose logistic regression as the best method because one of the most important aspects of the initial model selection is how easily it can be interpreted and put into production for testing on real data.

Thank you!