IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Kai Huang
2024-05-28

Outline

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

Executive Summary

- Summary of methodologies
  - The report explores various data science methodologies include:
    - Data API provided by SpaceX
    - Web scraping and data wrangling
    - Data Visualization and dashboard
    - Machine Learning
- Summary of all results
  - Train a machine learning model and predict the launch class

Introduction

- Project background and context
  - The space industry is rapidly evolving, and data science has emerged as a powerful tool to drive innovation and success in this field.
  - This report presents an in-depth analysis of how data science methodologies can be leveraged to gain a competitive edge with the SpaceX launch data.
- Problems you want to find answers
  - Can we use the launch data to predict the successful landing of the rocket

Section 1

Methodology

Methodology

Executive Summary

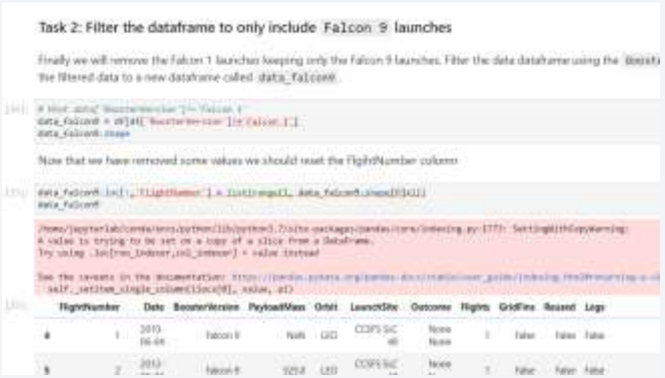
- Data collection methodology: using SpaceX data api and web scraping
- Perform data wrangling:
  - Calculate the number of launches on each site, Calculate the number and occurrence of each orbit, Calculate the number and occurence of mission outcome of the orbits, Create a landing outcome label from Outcome column
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - By using gridcv to find the best model

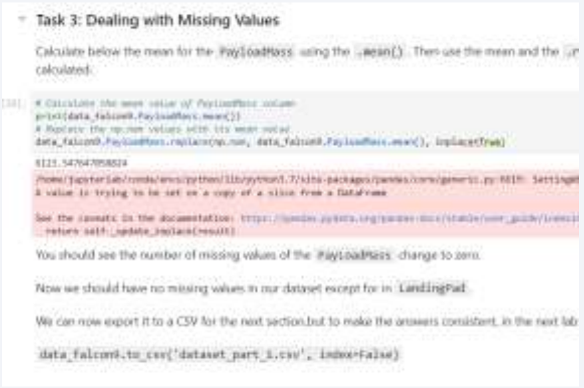
Data Collection

- Describe how data sets were collected.
  - Request and parse the SpaceX launch data using the GET request
  - Filter the dataframe to only include Falcon 9 launches
  - Dealing with Missing Values
  - Request the Falcon9 Launch Wiki page from its URL
  - Extract all column/variable names from the HTML table header
  - Create a data frame by parsing the launch HTML tables

Data Collection – SpaceX API

- https://github.com/huangkai31/Coursera_IBM_Applied-Data-Science-Capstone





Data Collection - Scraping

- https://github.com/huangkai31/Coursera_IBM_Applied-Data-Science-Capstone

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.text)
```

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>
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all("table")
len(html_tables)
```

```
25
```

Starting from the third table is our target table contains the actual launch records.

```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

...

You should able to see the columns names embedded in the table header elements `<th>` as follows:

TASK 3: Create a data frame by parsing the launch HTML tables

We will create an empty dictionary with keys from the extracted column names in the previous task. Later, this dictionary will be converted i

```
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'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

Next, we just need to fill up the `launch_dict` with launch records extracted from table rows.

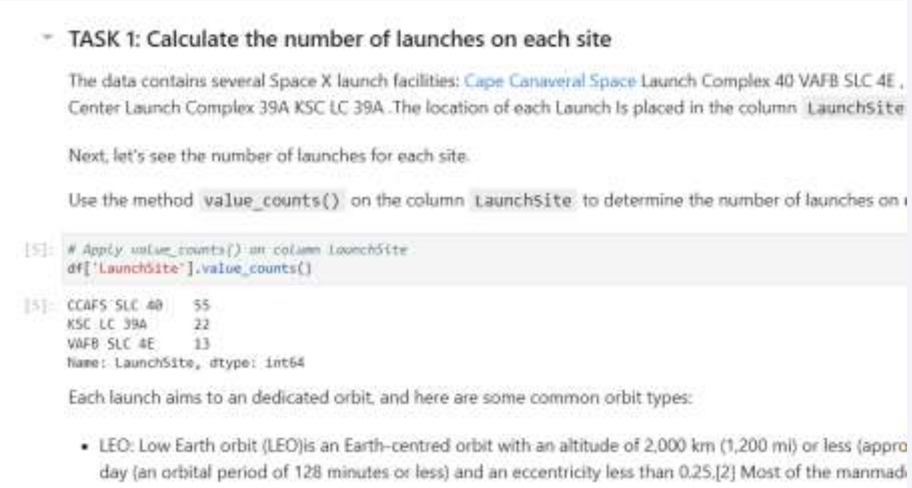
Usually, HTML tables in Wiki pages are likely to contain unexpected annotations and other types of noises, such as reference links `B0004.1` formatting, etc.

To simplify the parsing process, we have provided an incomplete code snippet below to help you to fill up the `launch_dict`. Please compl you can choose to write your own logic to parse all launch tables

```
extracted_row = 0
#Extract each table
```

Data Wrangling

- https://github.com/huangkai31/Coursera_IBM_Applied-Data-Science-Capstone



TASK 1: Calculate the number of launches on each site

The data contains several Space X launch facilities: Cape Canaveral Space Launch Complex 40 VAFB SLC 4E , Center Launch Complex 39A KSC LC 39A .The location of each Launch Is placed in the column `LaunchSite`

Next, let's see the number of launches for each site.

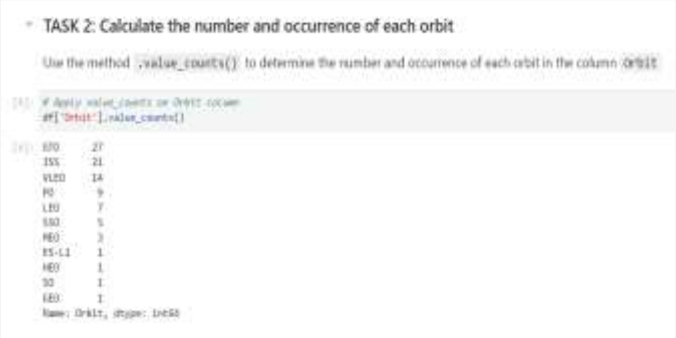
Use the method `value_counts()` on the column `LaunchSite` to determine the number of launches on

```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

Each launch aims to an dedicated orbit, and here are some common orbit types:

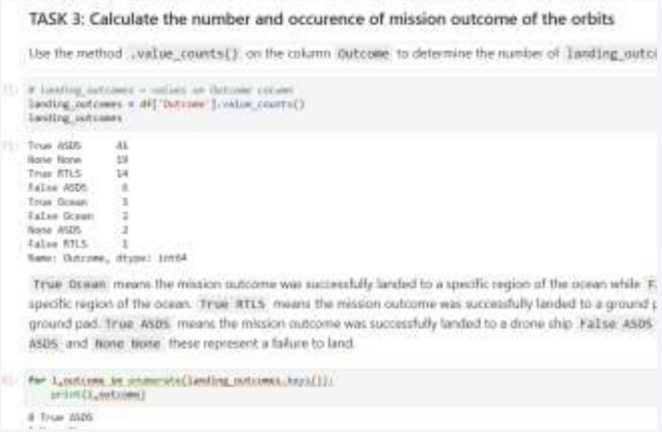
- LEO: Low Earth orbit (LEO)is an Earth-centred orbit with an altitude of 2,000 km (1,200 mi) or less (appro day (an orbital period of 128 minutes or less) and an eccentricity less than 0.25.[2] Most of the manmad



TASK 2: Calculate the number and occurrence of each orbit

Use the method `.value_counts()` to determine the number and occurrence of each orbit in the column `Orbit`

```
ISS     21
VLEO    14
ES-L1    1
```



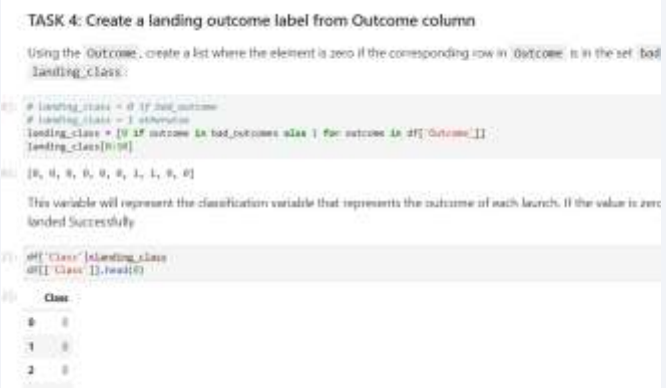
TASK 3: Calculate the number and occurence of mission outcome of the orbits

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outc`

```
landing_outcomes
```

```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

`True Ocean` means the mission outcome was successfully landed to a specific region of the ocean while specific region of the ocean. `True RTLS` means the mission outcome was successfully landed to a ground ground pad. `True ASDS` means the mission outcome was successfully landed to a drone ship `False ASDS` `ASDS` and `None None` these represent a failure to land.



TASK 4: Create a landing outcome label from Outcome column

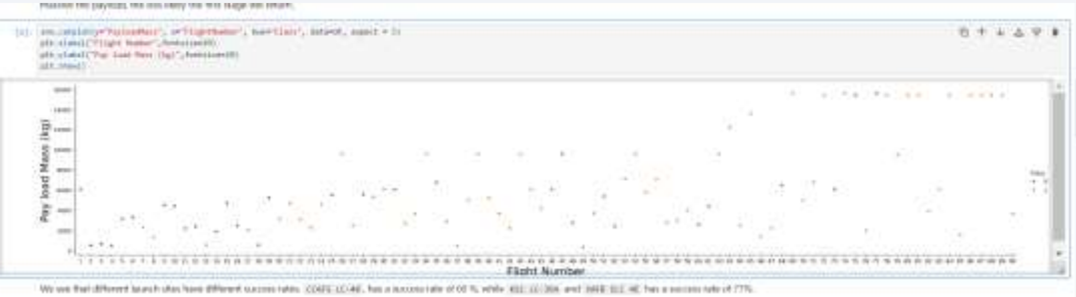
Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad` `landing_class`:

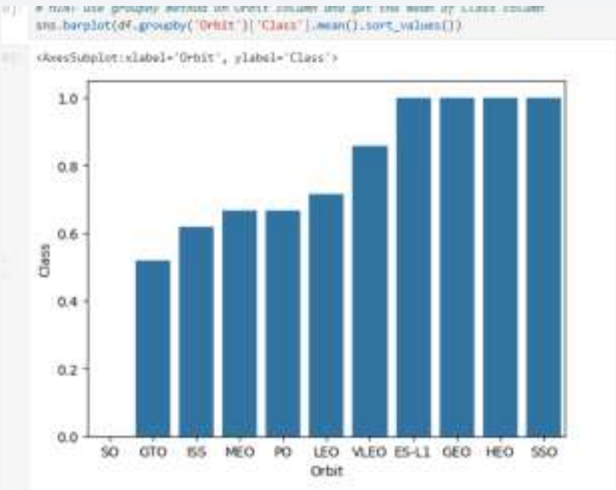
This variable will represent the classification variable that represents the outcome of each launch. If the value is zero landed Successfully

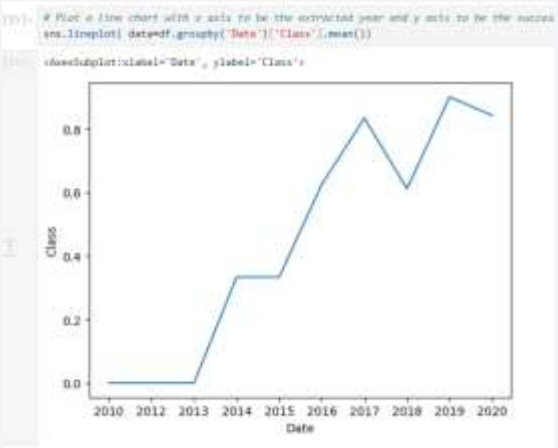
	Class
0	
1	
2	

EDA with Data Visualization

- https://github.com/huangkai31/Coursera_IBM_Applied-Data-Science-Capstone

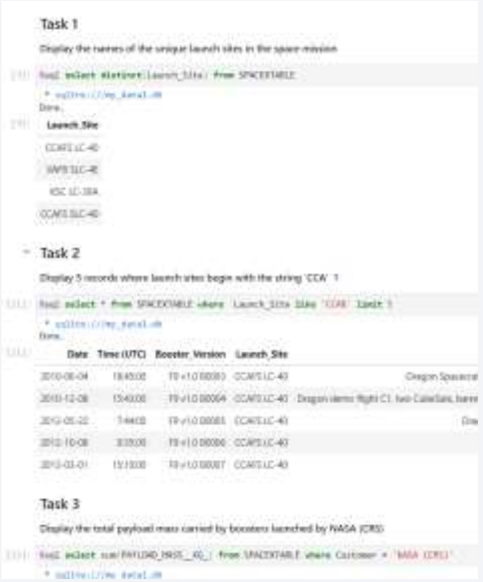






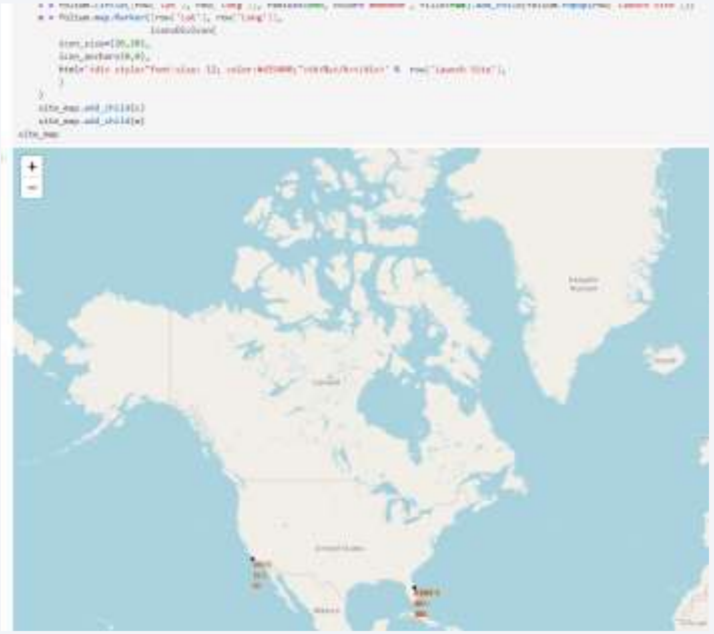
EDA with SQL

- https://github.com/huangkai31/Coursera_IBM_Applied-Data-Science-Capstone

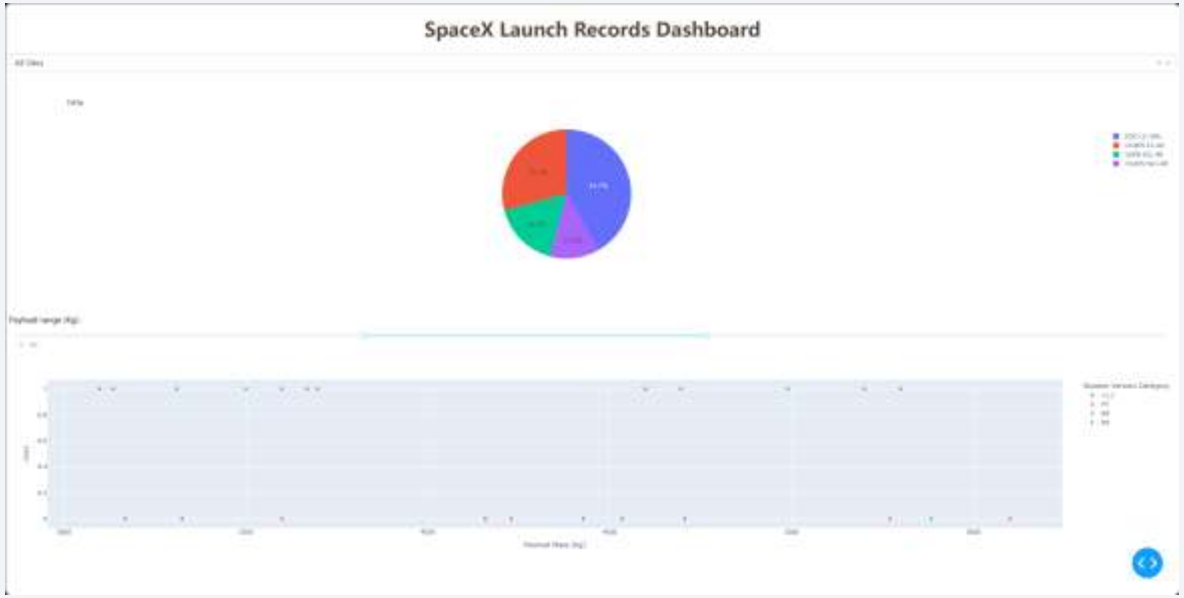


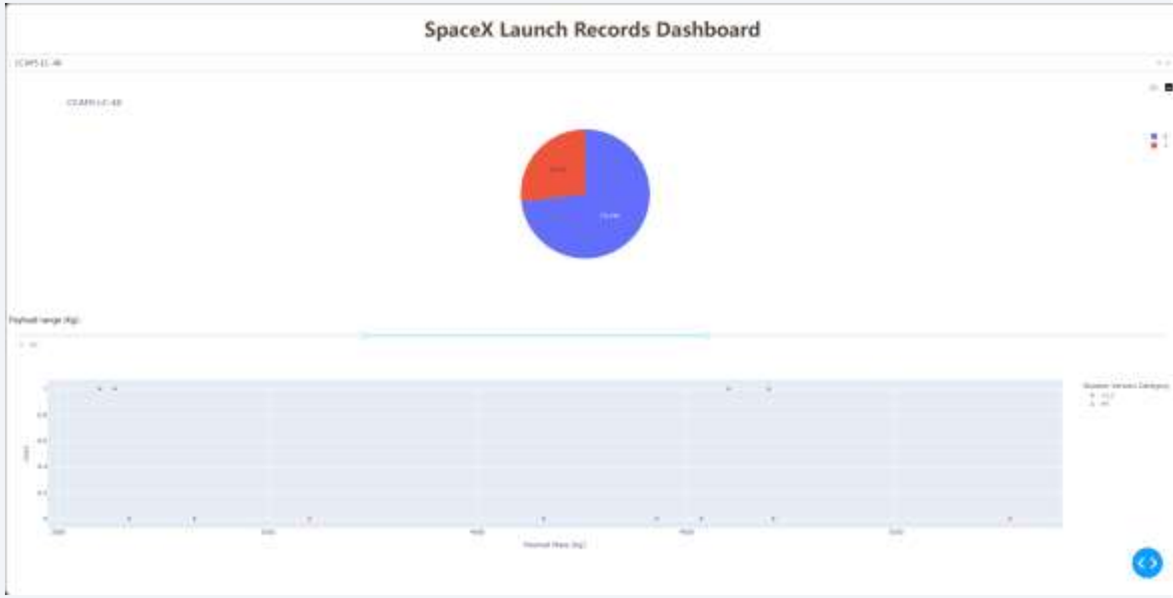
Build an Interactive Map with Folium

- https://github.com/huangkai31/Coursera_IBM_Applied-Data-Science-Capstone



Build a Dashboard with Plotly Dash





Predictive Analysis (Classification)

TASK 11

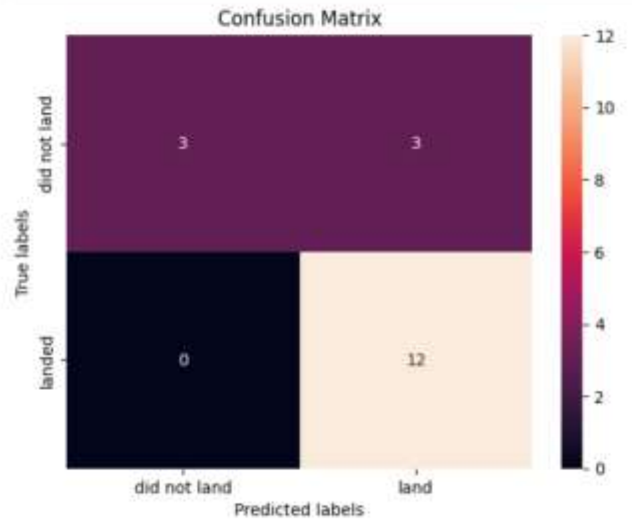
Calculate the accuracy of knn_cv on the test data using the method `score` :

```
[39]: knn_cv.score(X_test, Y_test)
```

```
[39]: 0.8333333333333334
```

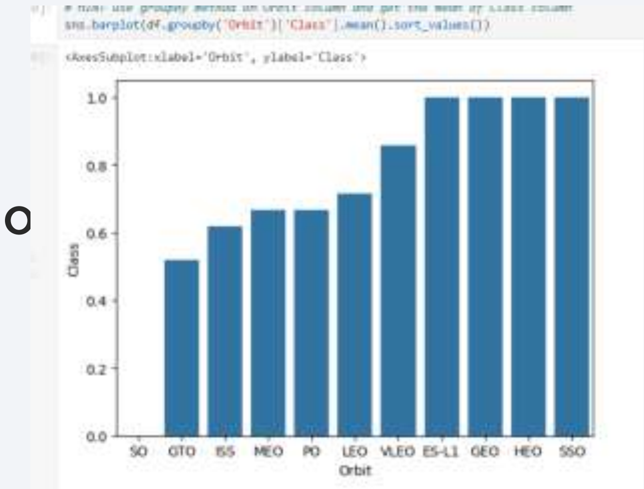
We can plot the confusion matrix

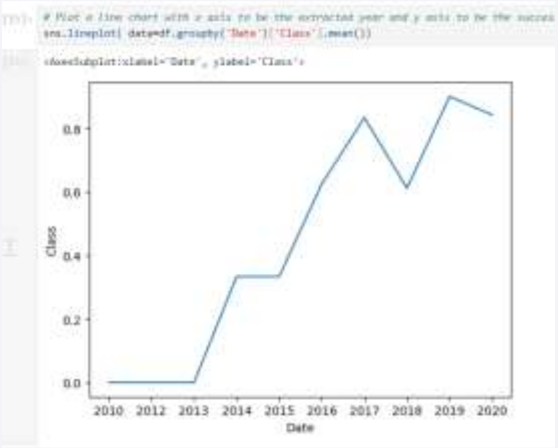
```
[40]: yhat = knn_cv.predict(X_test)
      plot_confusion_matrix(Y_test,yhat)
```



Results

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

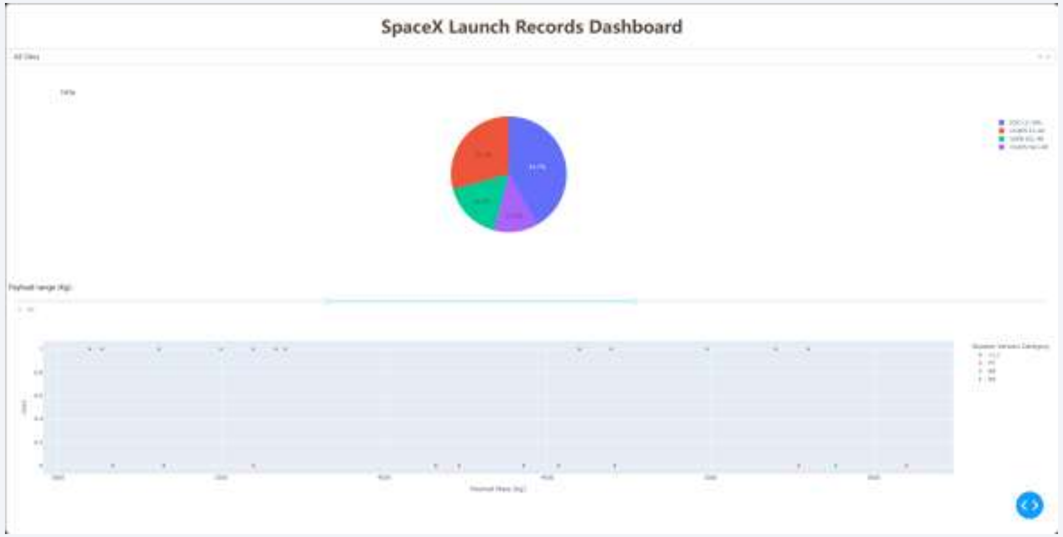




TASK 12

Find the method performs best:

```
53]: tree_cv.best_score_
53]: 0.875
```



Section 2

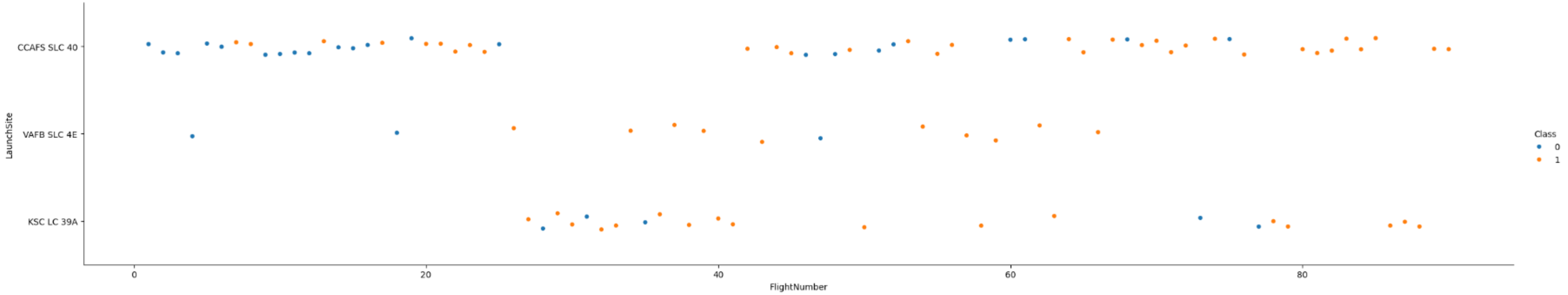
Insights drawn from EDA

Flight Number vs. Launch Site

```
### TASK 1: Visualize the relationship between Flight Number and Launch Site
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)

#
```

```
<seaborn.axisgrid.FacetGrid at 0x6316c10>
```

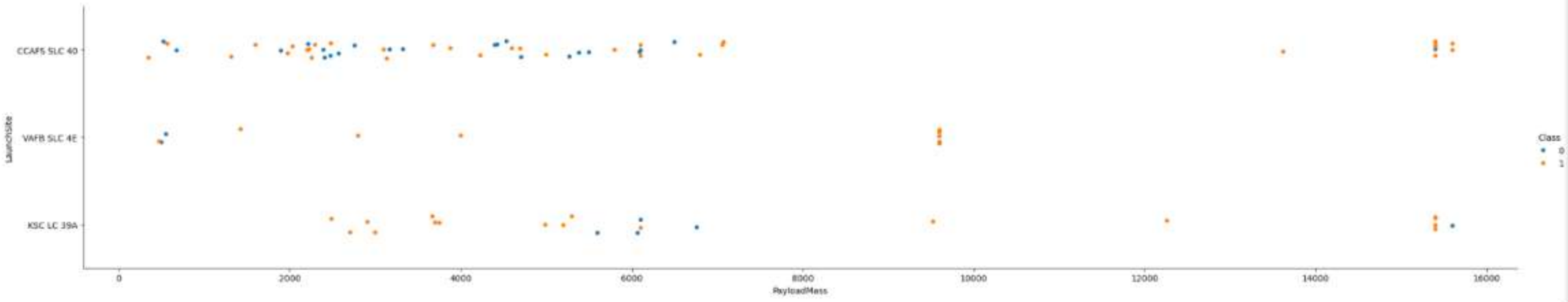


The SLC-40 fails more on lower flight numbers than others

Payload vs. Launch Site

```
[6]: ### TASK 2: Visualize the relationship between Payload and Launch Site
     sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
```

```
[6]: <seaborn.axisgrid.FacetGrid at 0x7c2b2c0>
```

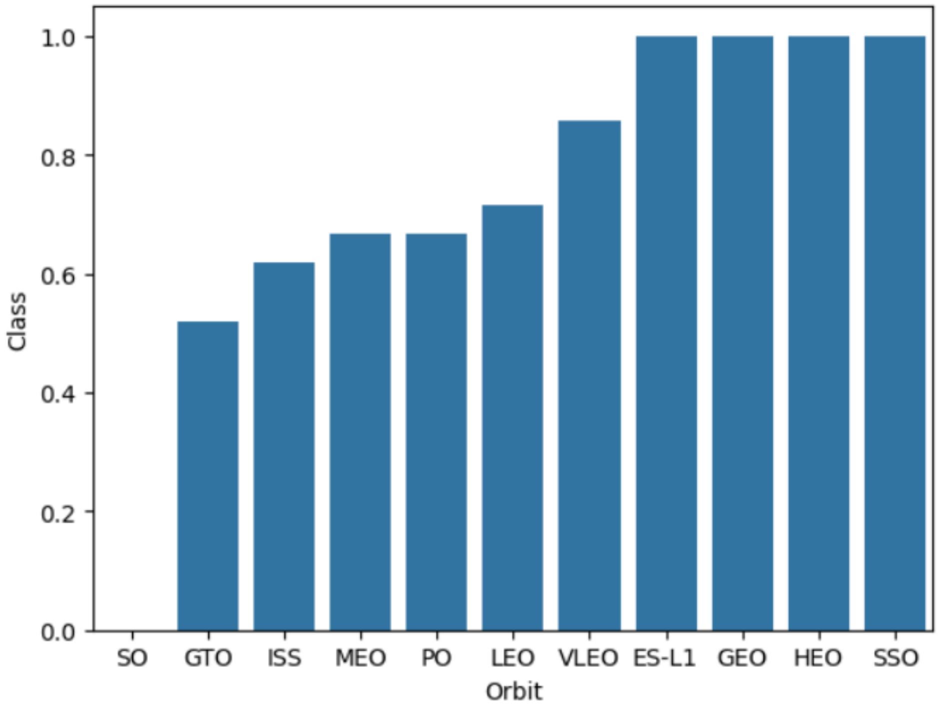


seems the more massive the payload, the less likely the first stage will return

Success Rate vs. Orbit Type

```
]: # HINT use groupby method on Orbit column and get the mean of Class column
   sns.barplot(df.groupby('Orbit')['Class'].mean().sort_values())
```

```
]: <AxesSubplot:xlabel='Orbit', ylabel='Class'>
```

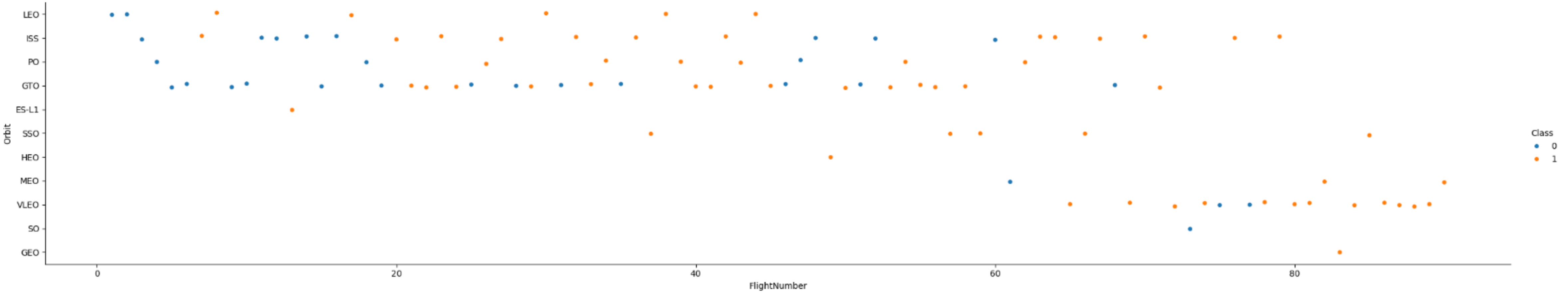


GTO has lower success rate while
ES-L1/GEO/HEO/SSO has high success rate

Flight Number vs. Orbit Type

```
11]: ### TASK  4: Visualize the relationship between FlightNumber and Orbit type
     sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)

11]: <seaborn.axisgrid.FacetGrid at 0x678eb48>
```

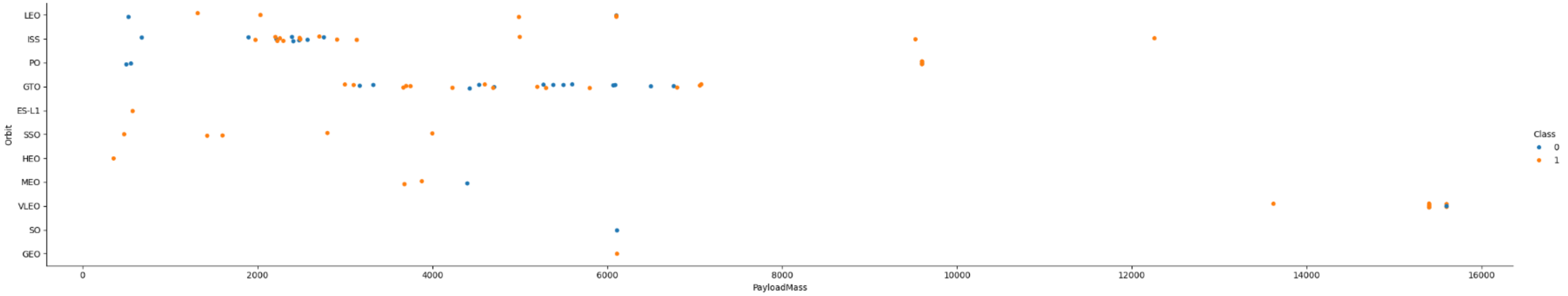


No relation found between flight number and orbit type

Payload vs. Orbit Type

```
### TASK  5: Visualize the relationship between Payload and Orbit type
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
```

```
<seaborn.axisgrid.FacetGrid at 0x7ced408>
```

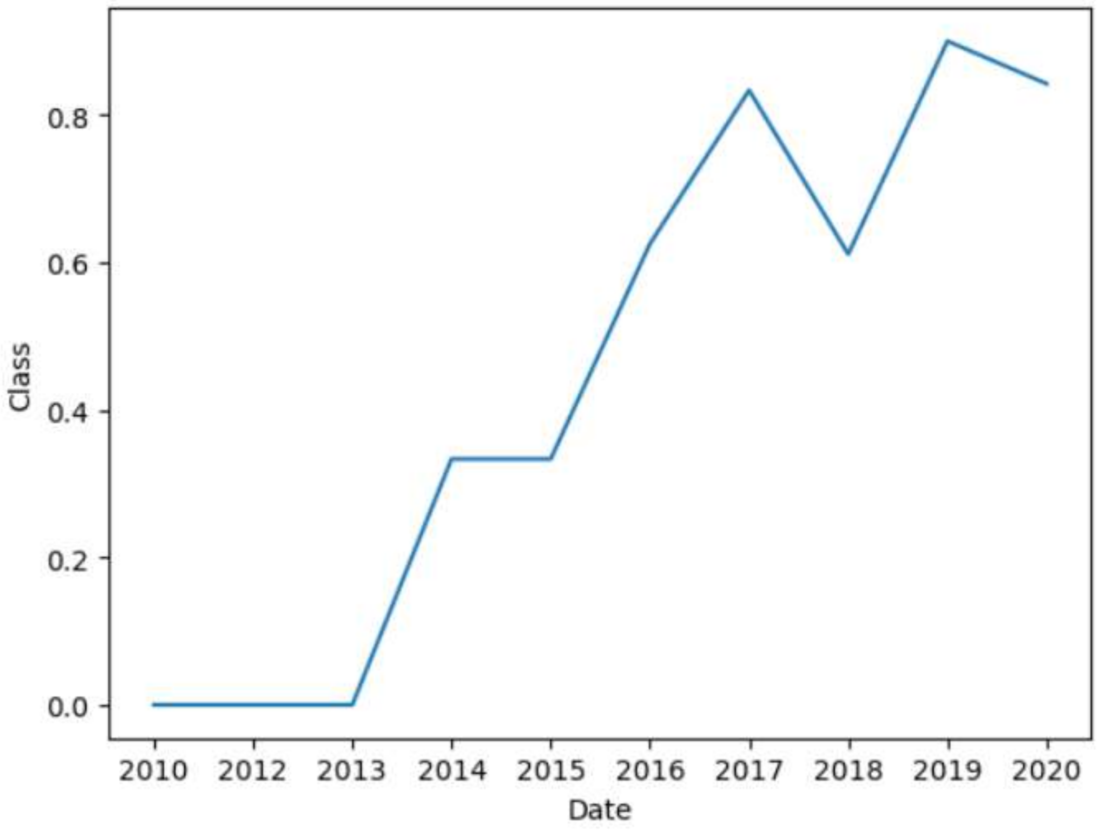


No relation found between payload and orbit type

Launch Success Yearly Trend

5]:
```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
sns.lineplot( data=df.groupby('Date')['Class'].mean())
```

5]:
```
<AxesSubplot:xlabel='Date', ylabel='Class'>
```



you can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

Task 1

Display the names of the unique launch sites in the space mission

```
%sql select distinct(Launch_Site) from SPACEXTABLE
```

```
 * 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'

Task 2

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

[11]:
```
%sql select * from SPACEXTABLE where  Launch_Site like 'CCA%' limit 5
```

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

[11]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MA:
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	

Total Payload Mass

Task 3

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

```
[12]: %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer = 'NASA (CRS)'
```

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

[12]:

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

Task 4 ¶

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

[13]:
```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTABLE where Booster_Version ='F9 v1.1'
```

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

[13]:

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

[17]:
```
%sql select min(Date) from SPACEXTABLE where Landing_Outcome = 'Success (ground pad)'
```

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

[17]:

min(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

[19]:
```
%sql select distinct(Booster_Version) from SPACEXTABLE where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_
```

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

[19]:

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

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

[20]:
```
%sql select count(*), Mission_Outcome from SPACEXTABLE  group by Mission_Outcome
```

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

[20]:

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

Boosters Carried Maximum Payload

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

[22]:
```
%sql select Booster_Version,PAYLOAD_MASS__KG_ from SPACEXTABLE where PAYLOAD_MASS__KG_ = (select max(PAYLOA
```

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

[22]:

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

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.

3]:
```
%sql select * from SPACEXTABLE where substr(Date,0,5)='2015' and  Landing_Outcome = 'Failure (drone ship)'
```

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

3]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_C
2015-01-10	9:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failur
2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	LEO (ISS)	NASA (CRS)	Success	Failur

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.

```
%sql select count(*) co , Landing_Outcome from SPACEXTABLE  where Date>= '2010-06-04' and Date <= '2017-03-20' group by
```

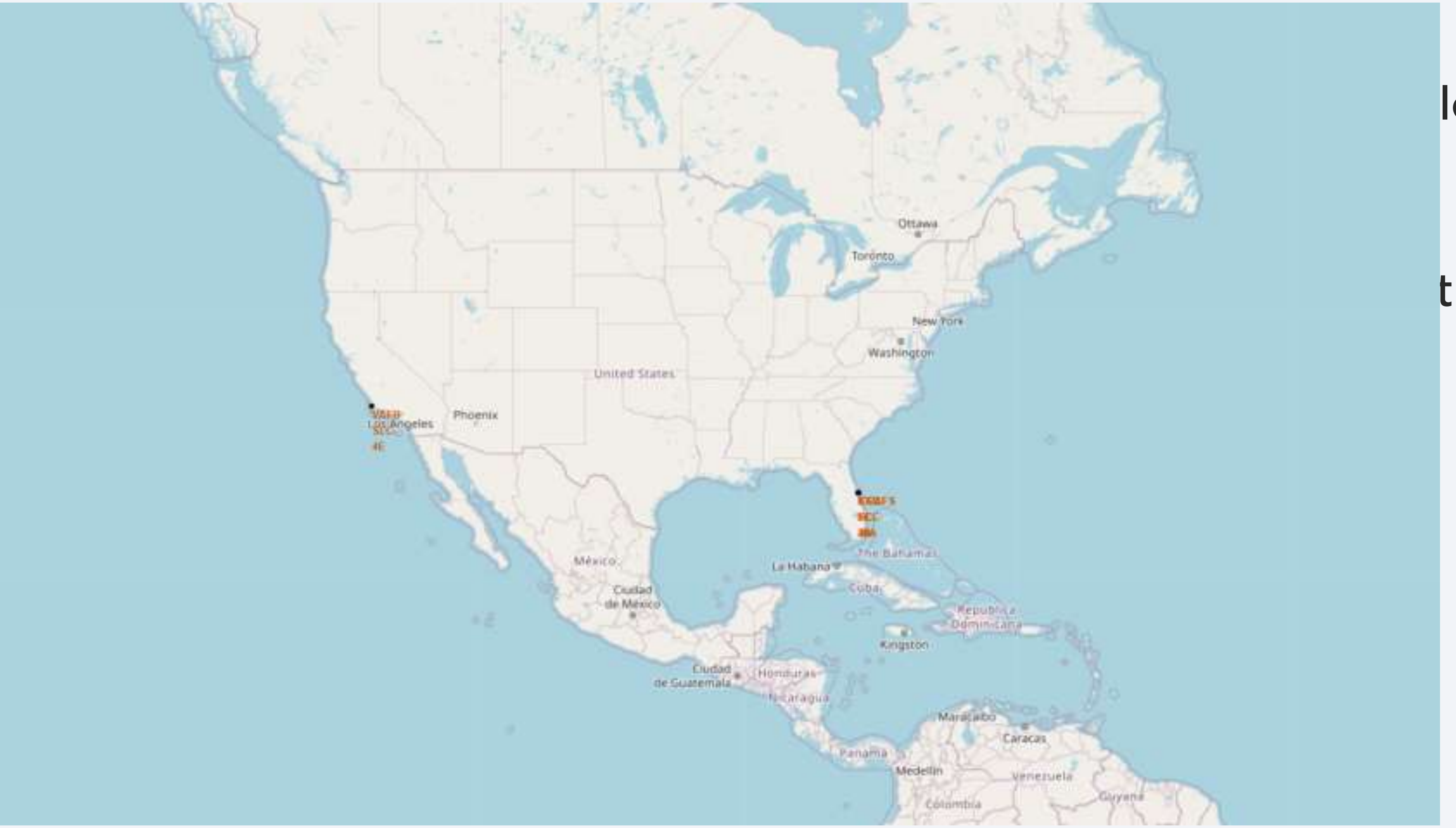
```
 * sqlite:///my_data1.db
Done.
```

co	Landing_Outcome
10	No attempt
5	Success (drone ship)
5	Failure (drone ship)
3	Success (ground pad)
3	Controlled (ocean)
2	Uncontrolled (ocean)
2	Failure (parachute)
1	Precluded (drone ship)

Section 3

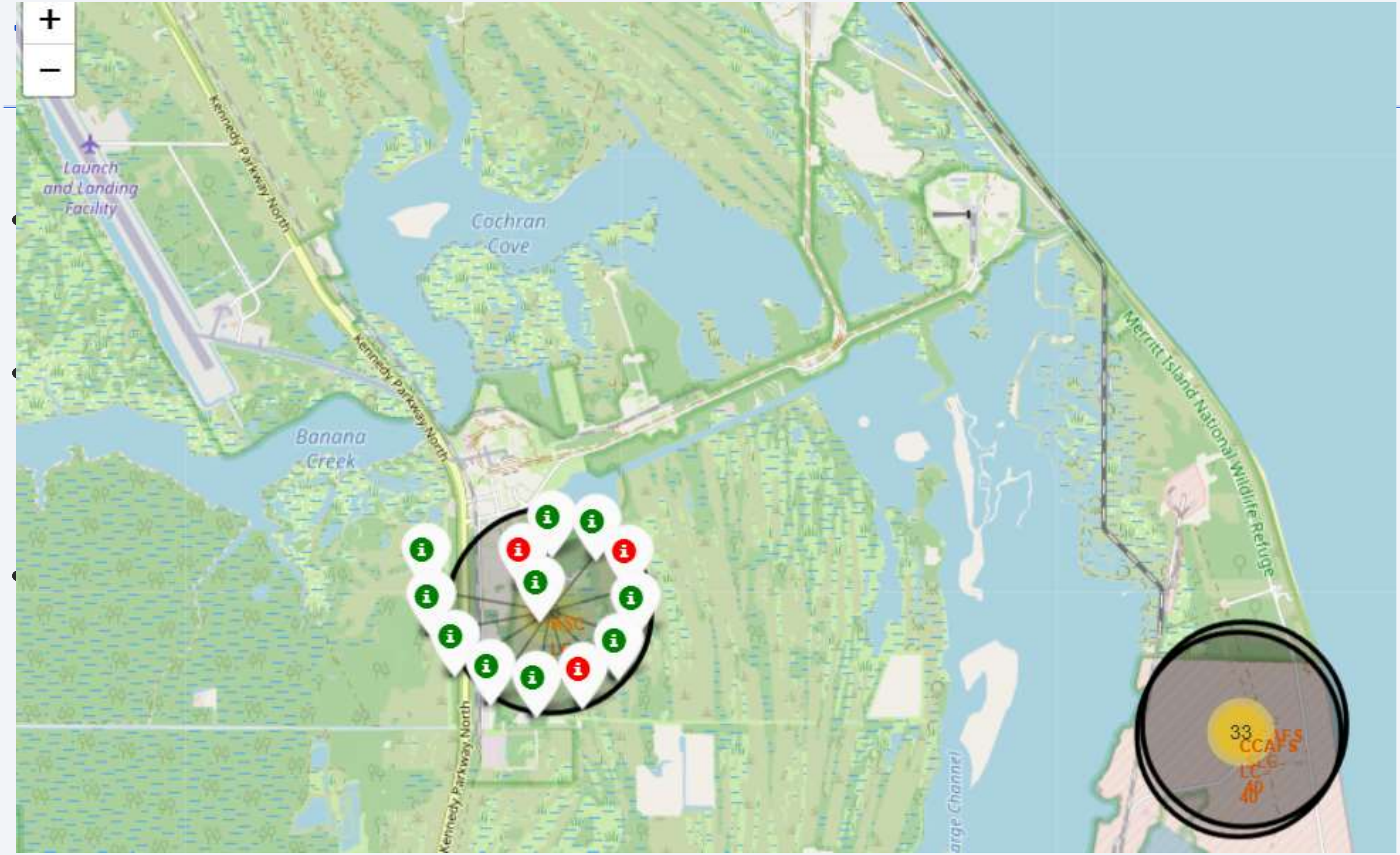
Launch Sites Proximities Analysis

<Folium Map Screenshot 1>



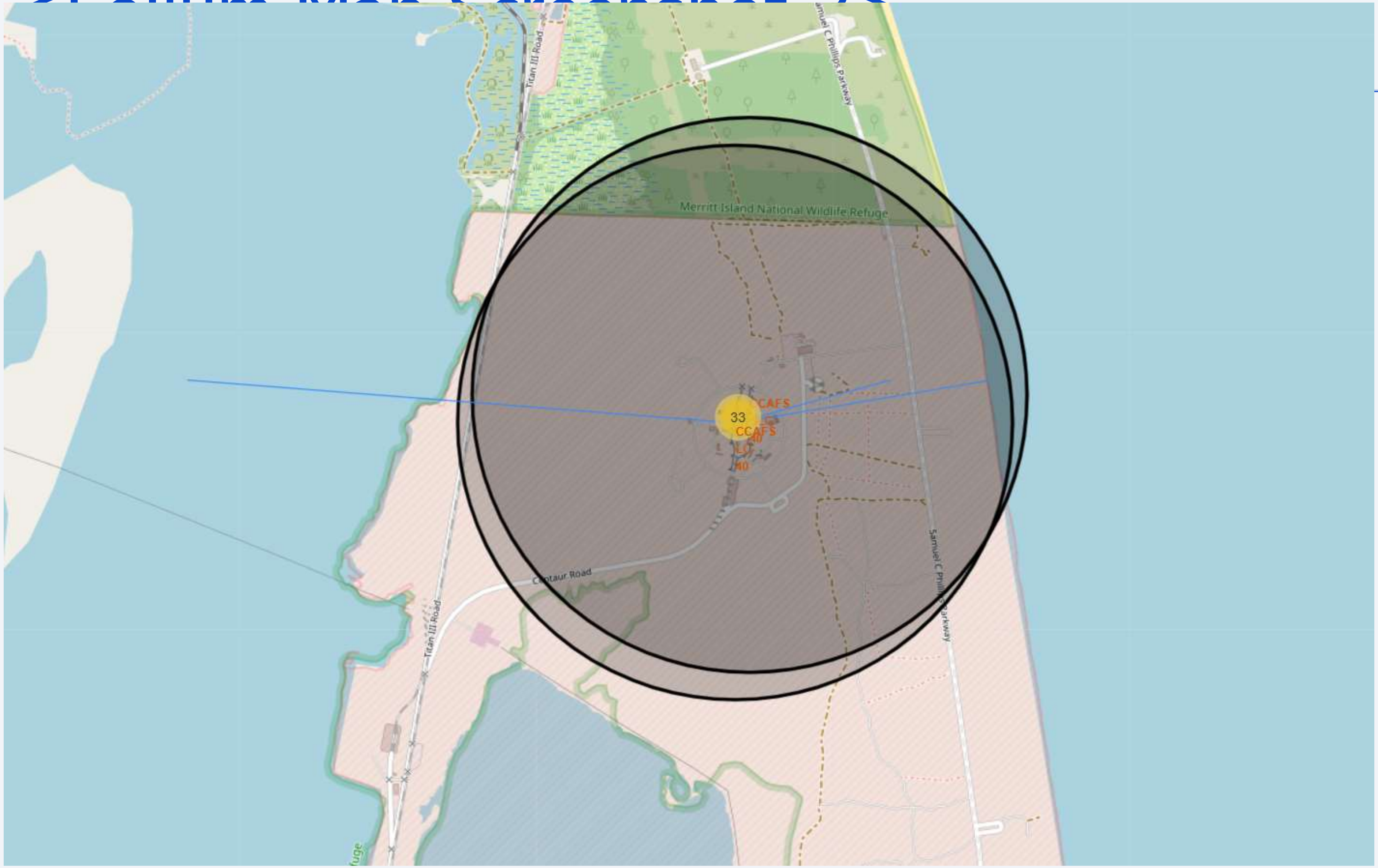
le

t to include



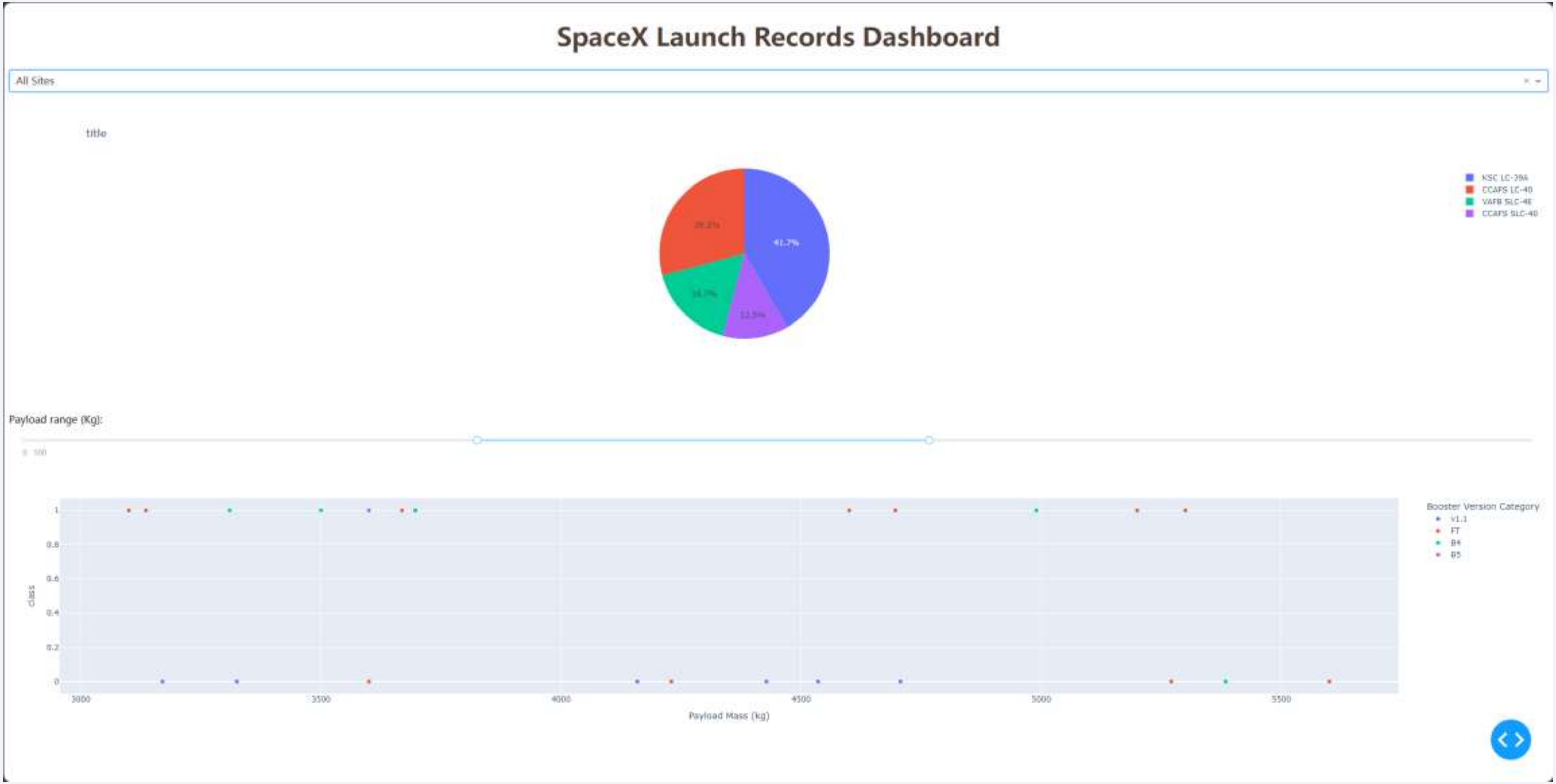
Launch and Landing Facility
Kennedy Parkway North
Cochran Cove
Banana Creek
Merritt Island National Wildlife Refuge
33

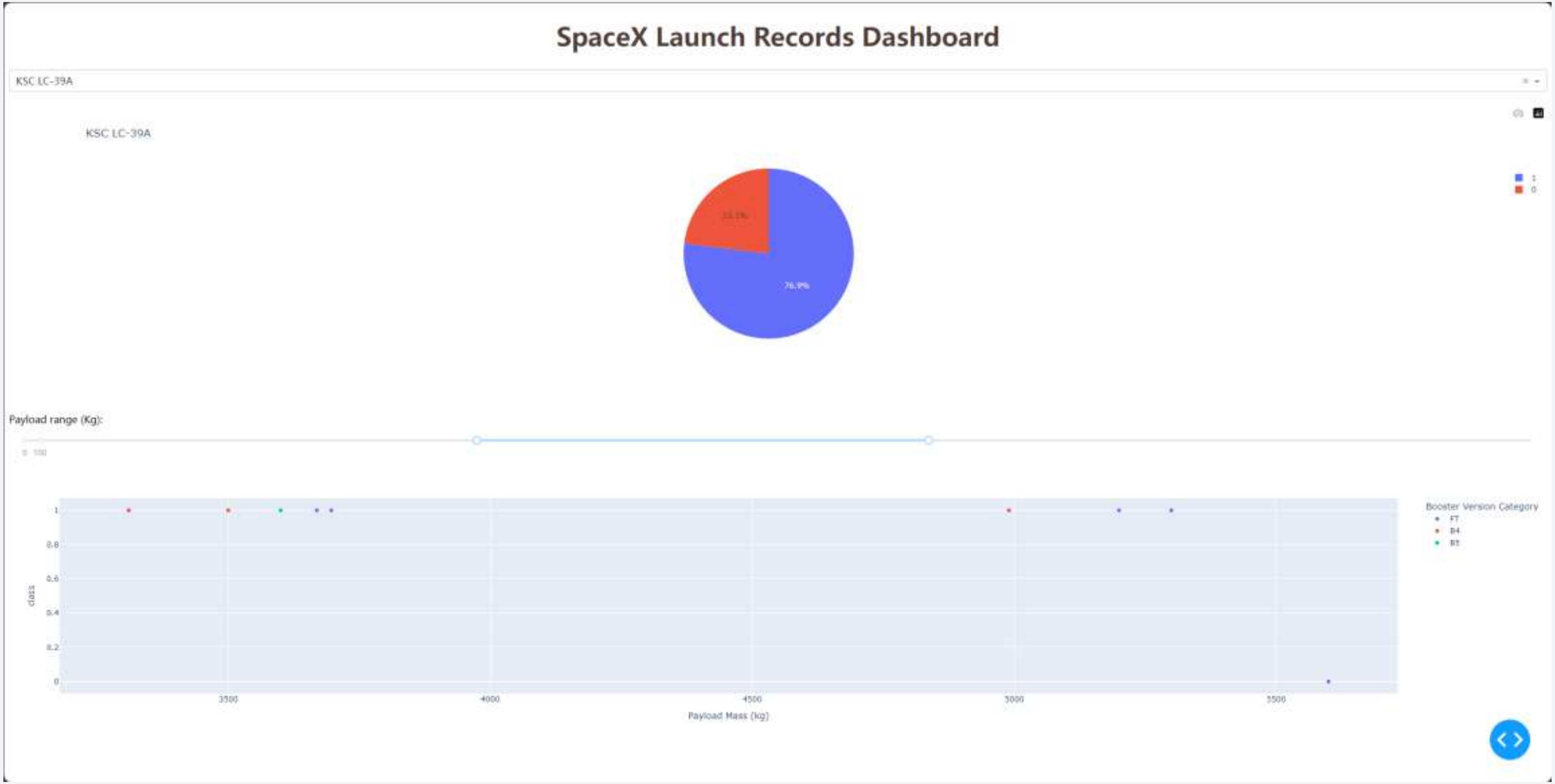
<Folium Map Screenshot 3>



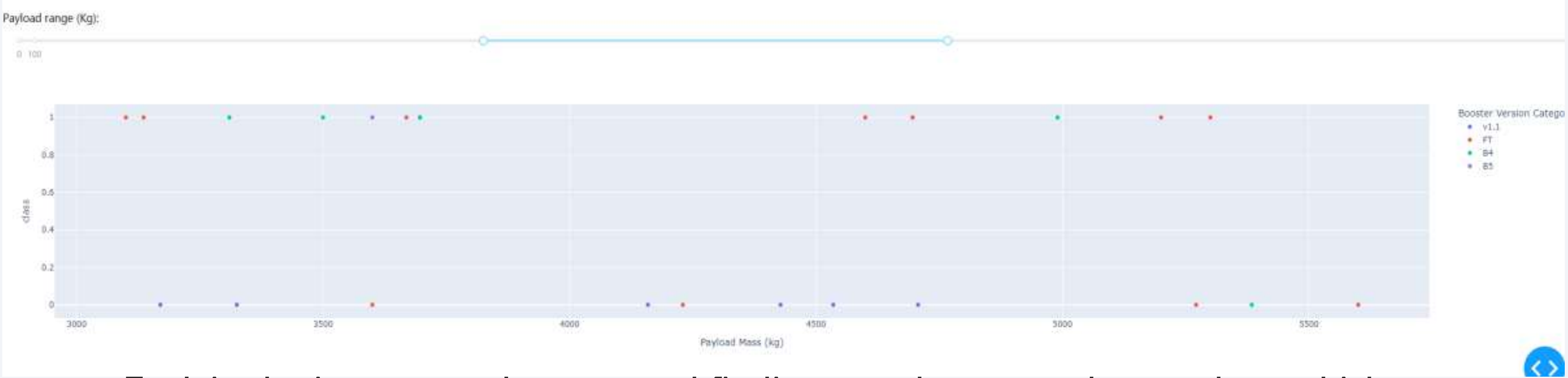
Section 4

Build a Dashboard with Plotly Dash


SpaceX Launch Records Dashboard
All Sites
title
41.7%
KSC LC-39A
CCAFS LC-40
VAFB SLC-4E
CCAFS SLC-40
Payload range (Kg):
class
1
0.8
0.6
0.4
0.2
0
3000
3500
4000
4500
5000
5500
Payload Mass (kg)
Booster Version Category
v1.1
FT
B4
B5


SpaceX Launch Records Dashboard
KSC LC-39A
KSC LC-39A
76.9%
1
0
Payload range (Kg):
3500
4000
4500
5000
5500
Payload Mass (kg)
class
Booster Version Category
FT
B4
B5

<Dashboard Screenshot 3>



- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.

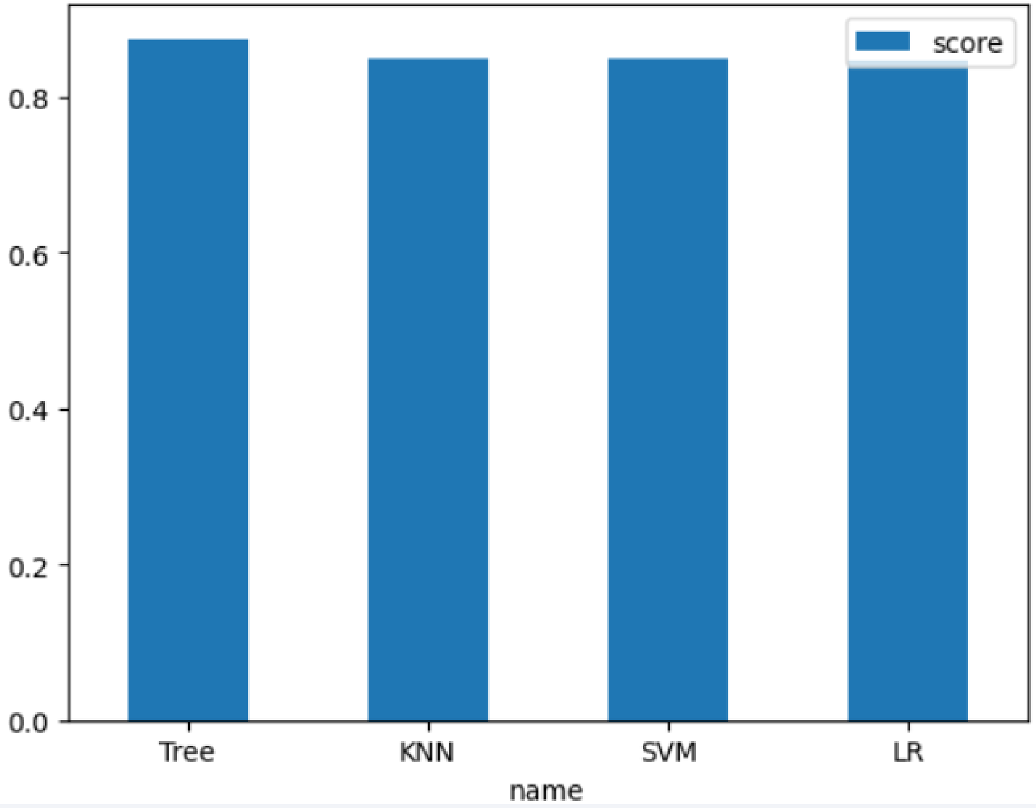
Section 5

Predictive Analysis (Classification)

Classification Accuracy

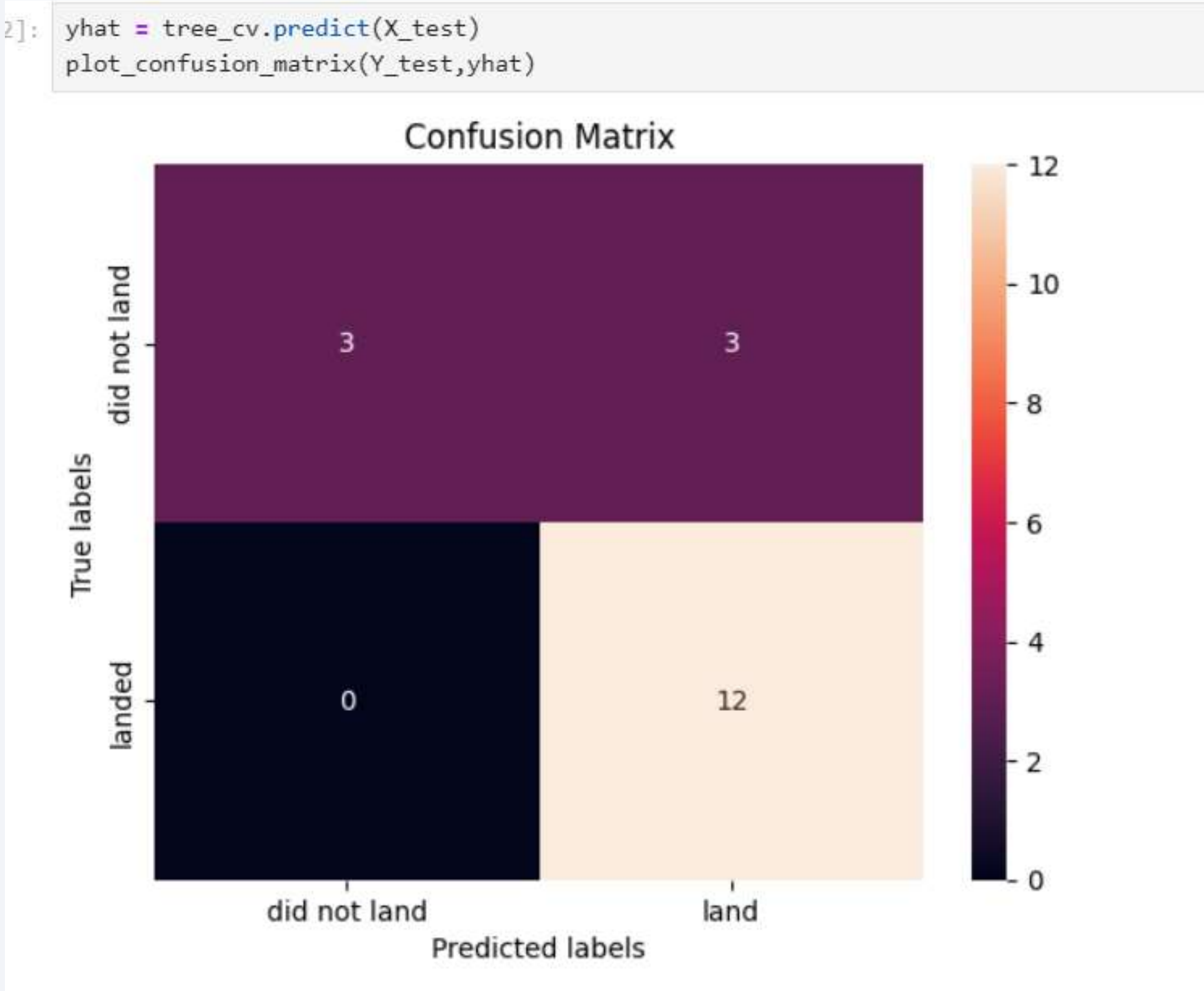
```
import pandas as pd
df = pd.DataFrame({"name":['LR', 'SVM', 'Tree', 'KNN' ], 'score':[logreg_cv.best_score_, svm_cv.best_s
df.sort_values('score', ascending=False).plot.bar(x='name', y='score', rot=0)
```

```
<AxesSubplot:xlabel='name'>
```




```
[48]: tree_cv.best_score_
[48]: 0.875
```

Confusion Matrix



ming model with an

Conclusions

- In the race to conquer space, data science has emerged as a game-changer.
- By embracing data-driven methodologies, organizations can optimize their operations, improve decision-making, and unlock valuable insights that will propel them to victory.
- This report highlights the immense potential of data science in the space industry and serves as a roadmap for organizations looking to leverage its power to win the space race.

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!