IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Yangle Ma>

<2024-07-31>

Outline

Executive Summary

- **Summary of methodologies**
- Data Collection and Wrangling:
- EDA and Visual analytics
- EDA and SQL
- interactive map with Folium
- Plotly Dash dashboard
- required predictive analysis

- **Summary of all results**
- Both achive the requirements and peform well

Introduction

- **Project background and context**
- 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**
- I would like to figure out whats the differences between the success situations and failure situation and apply changes to improve the launch success rate.

Section 1

Methodology

Methodology

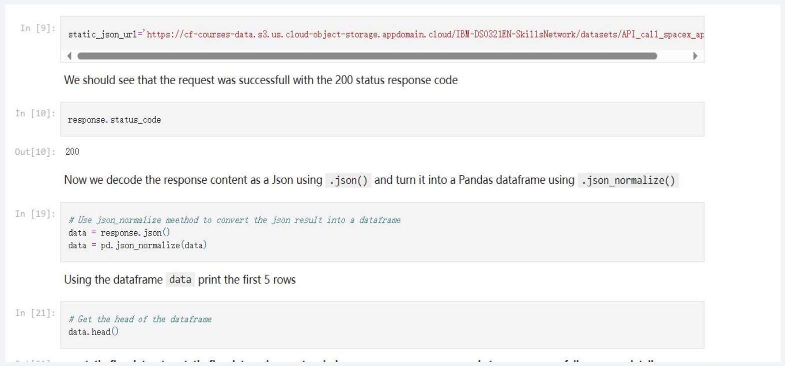
Executive Summary

- Data collection methodology:
  - By importing request and relevant libraries, i collected the requied data by Web scrapling and using the requests function with url successfully.
- Perform data wrangling
  - Clean the missing value by using the mean values to fill. Define the 'class' feature to identify whether succeed or not of the launch
- 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 builing mutiple models,using GridSearchCV to fit best parameters and evaulate the abilities by using score and metrix functions in test set.

Data Collection

- Describe how data sets were collected.
- You need to present your data collection process use key phrases and flowcharts

1. GET request for the Space Launch Site



```
In [9]:  static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_ap
```

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

```
In [10]: response.status_code

Out[10]: 200
```

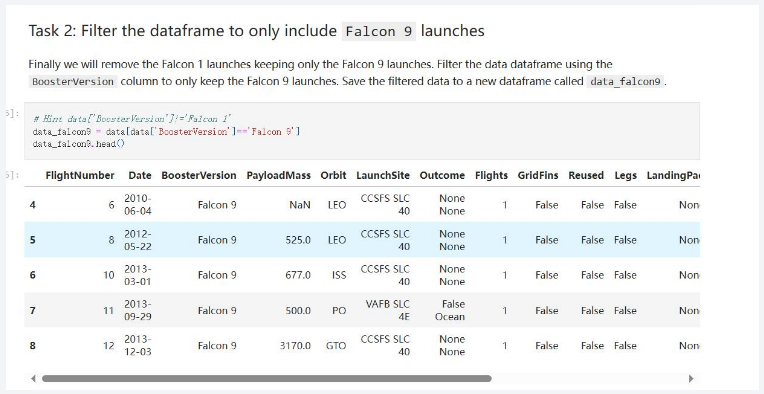
Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
In [19]: # Use json_normalize meethod to convert the json result into a dataframe
         data = response.json()
         data = pd.json_normalize(data)
```

Using the dataframe `data` print the first 5 rows

```
In [21]: # Get the head of the dataframe
         data.head()
```

2. Filter the dataframe to only include Falcon 9 launches



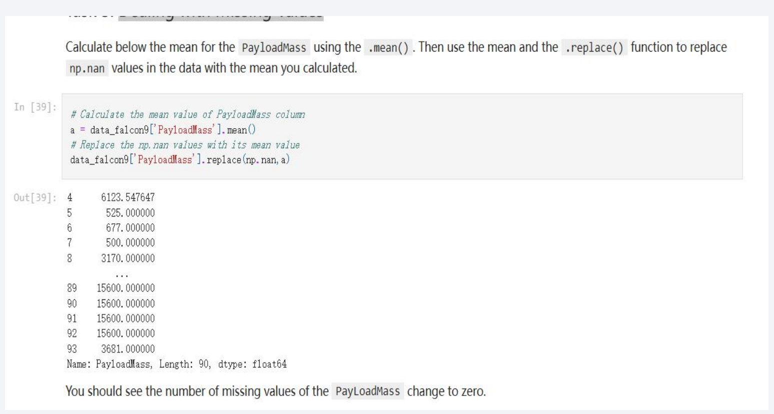
Task 2: Filter the dataframe to only include `Falcon 9` launches

Finally we will remove the Falcon 1 launches keeping only the Falcon 9 launches. Filter the data dataframe using the `BoosterVersion` column to only keep the Falcon 9 launches. Save the filtered data to a new dataframe called `data_falcon9`.

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data[data['BoosterVersion']=='Falcon 9']
data_falcon9.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPa
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	Non
5	8	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	Non
6	10	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	Non
7	11	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	Non
8	12	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False	Non

3. Dealing with Missing Values



Calculate below the mean for the `PayloadMass` using the `.mean()`. Then use the mean and the `.replace()` function to replace `np.nan` values in the data with the mean you calculated.

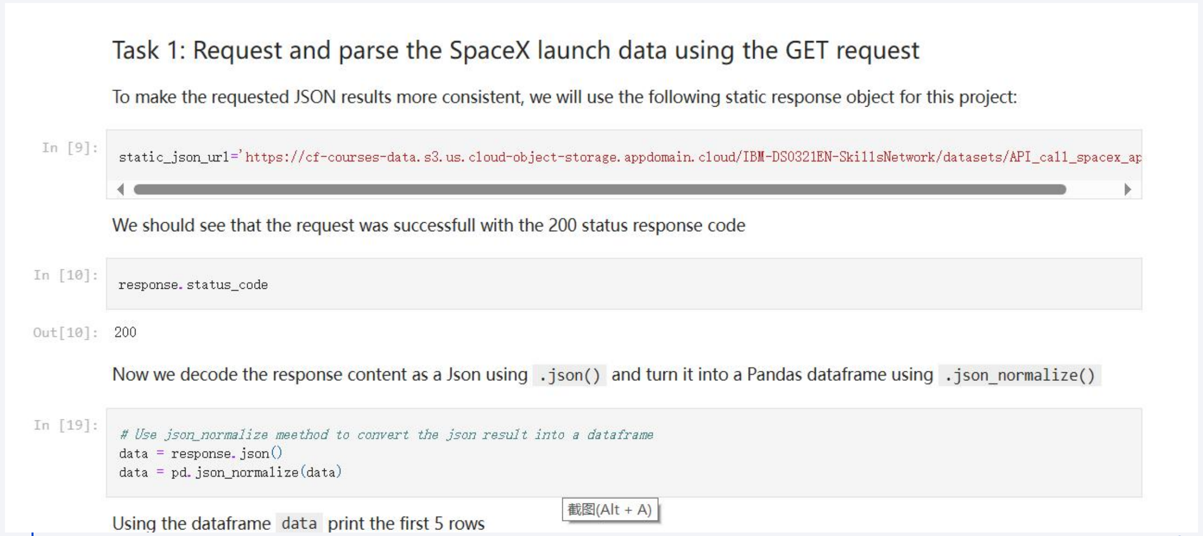
```
In [39]: # Calculate the mean value of PayloadMass column
         a = data_falcon9['PayloadMass'].mean()
         # Replace the np.nan values with its mean value
         data_falcon9['PayloadMass'].replace(np.nan, a)

Out[39]: 4       6123.547647
         5        525.000000
         6        677.000000
         7        500.000000
         8       3170.000000
                    ...
         89     15600.000000
         90     15600.000000
         91     15600.000000
         92     15600.000000
         93      3681.000000
         Name: PayloadMass, Length: 90, dtype: float64
```

You should see the number of missing values of the `PayLoadMass` change to zero.

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose



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_call_spacex_ap
```

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

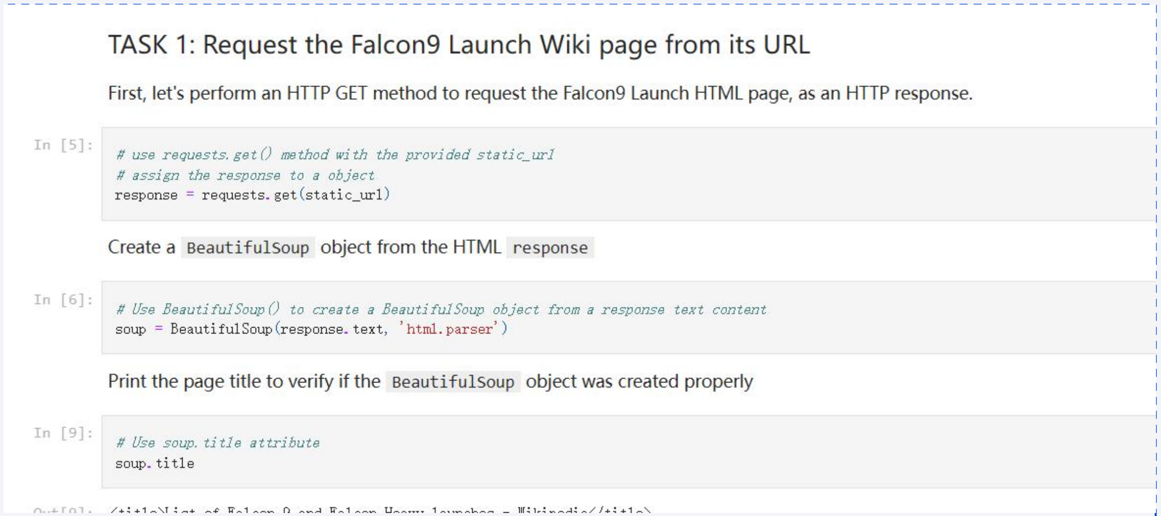
Using the dataframe `data` print the first 5 rows

https://github.com/MaYangle/IBM-Data-Science/blob/main/Applied%20Data%20Science%20Capstone/Collecting%20the%20Data/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection – Scraping

- Present your web scraping process using key phrases and flowcharts

- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose



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.text, 'html.parser')
```

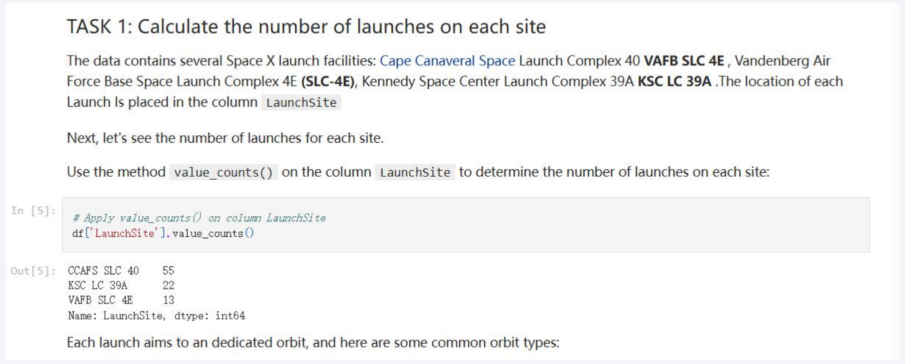
Print the page title to verify if the `BeautifulSoup` object was created properly

```
In [9]: # Use soup.title attribute
        soup.title

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

https://github.com/MaYangle/IBM-Data-Science/blob/main/Applied%20Data%20Science%20Capstone/Collecting%20the%20Data/jupyter-labs-webscraping.ipynb

Data Wrangling

- Describe how data were processed
  - convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.
- You need to present your data wrangling process using key phrases and flowcharts
- Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose



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** , Vandenberg Air Force Base Space Launch Complex 4E **(SLC-4E)**, Kennedy Space 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 each site:

```
In [5]:  # Apply value_counts() on column LaunchSite
         df['LaunchSite'].value_counts()
```

```
Out[5]:  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:

https://github.com/MaYangle/IBM-Data-Science/blob/main/Applied%20Data%20Science%20Capstone/Collecting%20the%20Data/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts
  - Scatter plot charts for Flight number and Launch Site,Payload mass and Launch Site Flight number and Orbit type,Payload mass and Orbit type to see their relationships.
  - Bar chart for success rate and Orbit type to see their relationship.
  - Line chart to see the trend of success rate with years.
- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose

https://github.com/MaYangle/IBM-Data-Science/blob/main/Applied%20Data%20Science%20Capstone/Exploratory%20Data%20Analysis(EDA)/edadataviz.ipynb

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
  - Display the names of the unique launch sites in the space mission
  - Display 5 records where launch sites begin with the string 'CCA'
  - Display the total payload mass carried by boosters launched by NASA (CRS)
  - Display average payload mass carried by booster version F9 v1.1
  - List the date when the first succesful landing outcome in ground pad was acheived.
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - List the total number of successful and failure mission outcomes
  - List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
  - 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.
  - 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.
- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

https://github.com/MaYangle/IBM-Data-Science/blob/main/Applied%20Data%20Science%20Capstone/Exploratory%20Data%20Analysis(EDA)/jupyter-labs-eda-sql-coursera_sqllite.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
  - Create and add folium.Circle and folium.Marker for each launch site on the site map
  - For each launch result in spacex_df data frame, add a folium.Marker to marker_cluster
  - Mark down the point and add a coasting line
- Explain why you added those objects
  - To see whether there exists the similarities in their locations.
  - To distinguish every launch sites
  - To see their distances between the local sites.
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

https://github.com/MaYangle/IBM-Data-Science/blob/main/Applied%20Data%20Science%20Capstone/Interactive%20Visual%20Analytics%20and%20Dashboard/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
  - Add a dropdown list to enable Launch Site selectio
  - Add a pie chart to show the total successful launches count for all sites
  - Add a slider to select payload range
  - Add a scatter chart to show the correlation between payload and launch success
- Explain why you added those plots and interactions
  - In this way, the audience can easily figure out the success rate for different launch sites and also easy to see the correlation between payload and success. It can be concluded how to improve or decrease the success rate.
- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

https://github.com/MaYangle/IBM-Data-Science/blob/main/Applied%20Data%20Science%20Capstone/Interactive%20Visual%20Analytics%20and%20Dashboard/spacex_dash_app.py

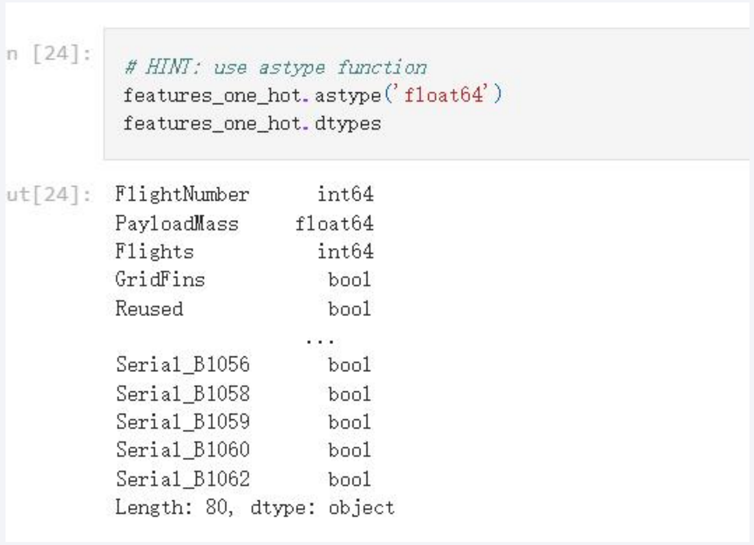
Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
    - Use the models, evaluated by the score functions,improve by the GridSearchCV function to find the best parameters and see the final scores of each models to select the best one.
- You need present your model development process using key phrases and flowchart
- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

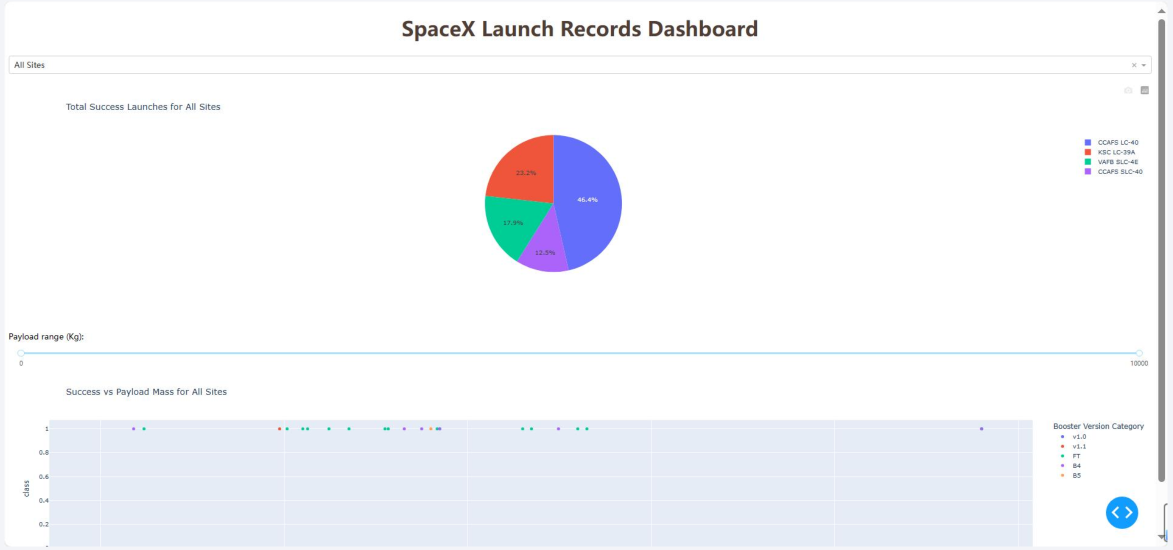
https://github.com/MaYangle/IBM-Data-Science/blob/main/Applied%20Data%20Science%20Capstone/Predictive%20Analysis%EF%BC%88Classification%EF%BC%89/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

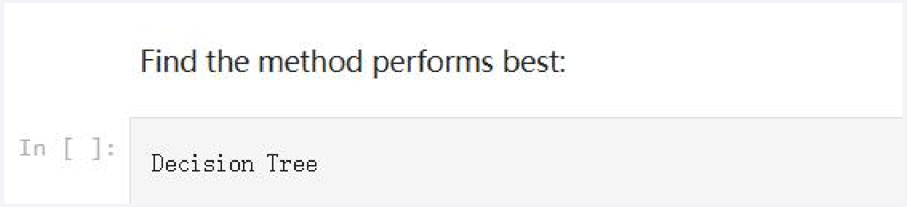
Results

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



```
n [24]:  # HINT: use astype function
         features_one_hot.astype('float64')
         features_one_hot.dtypes

ut[24]:  FlightNumber      int64
         PayloadMass     float64
         Flights           int64
         GridFins           bool
         Reused             bool
                            ...
         Serial_B1056       bool
         Serial_B1058       bool
         Serial_B1059       bool
         Serial_B1060       bool
         Serial_B1062       bool
         Length: 80, dtype: object
```





Find the method performs best:

```
In [ ]:  Decision Tree
```

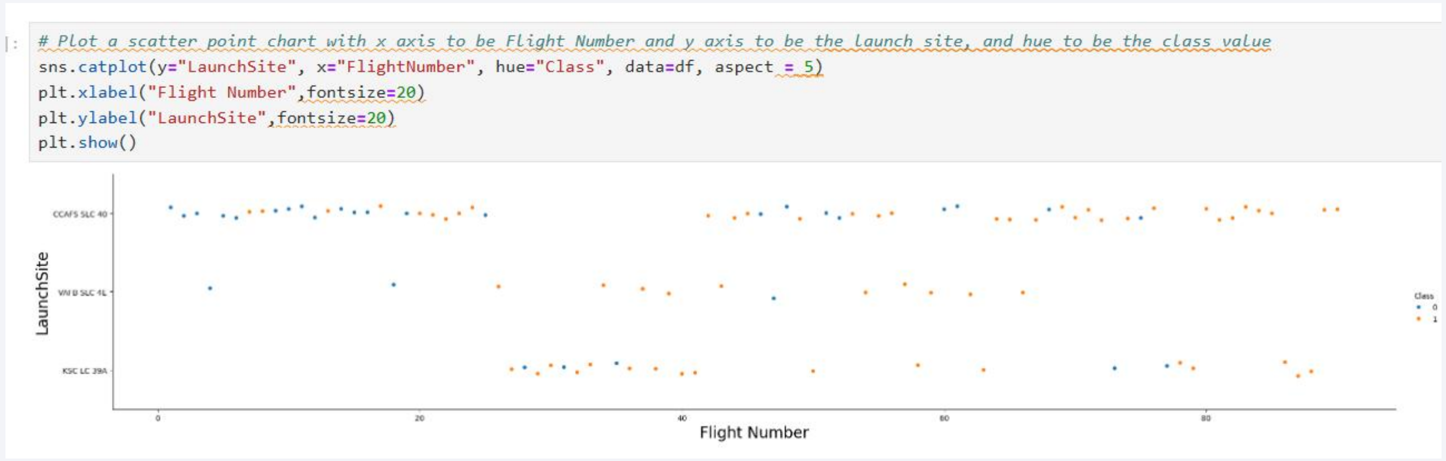
Section 2

Insights drawn from EDA

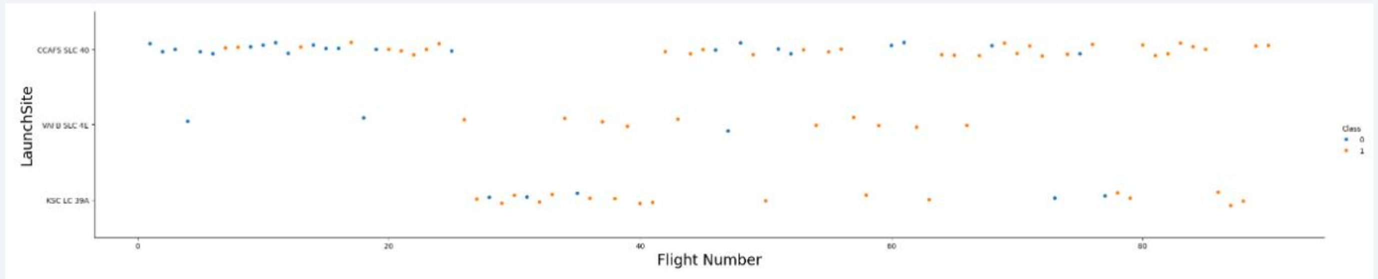
Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site

- Show the screenshot of the scatter plot with explanations



```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```

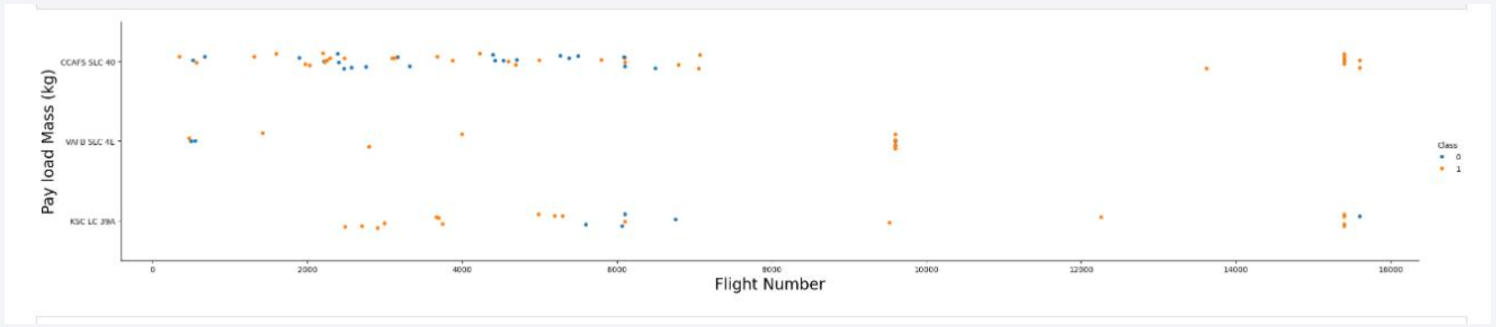


Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

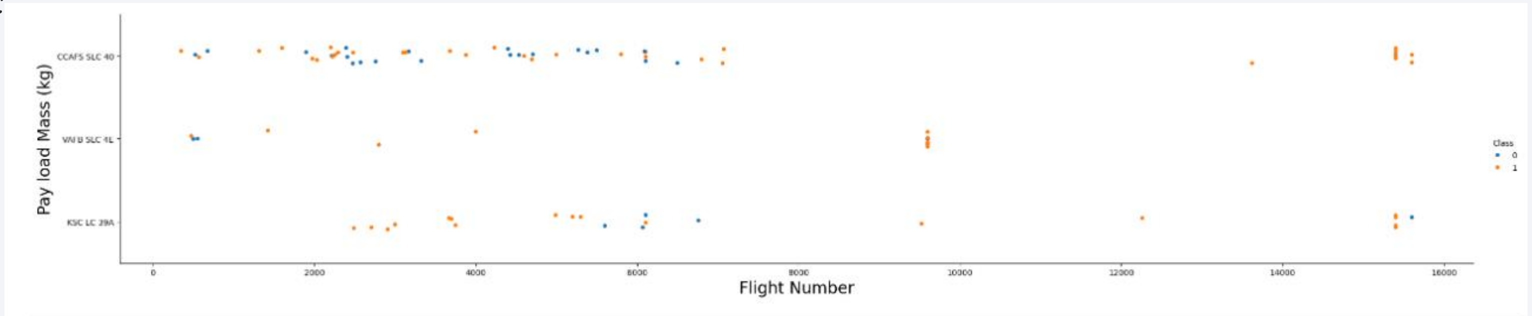
With the increase of the Filght number,the success rate for the both LauchSite become higher.

Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site



- Show the screenshot of the scatter plot with explanations

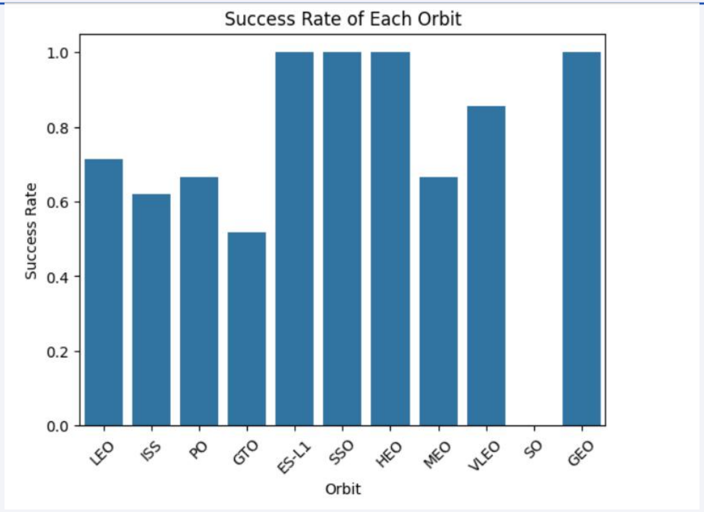


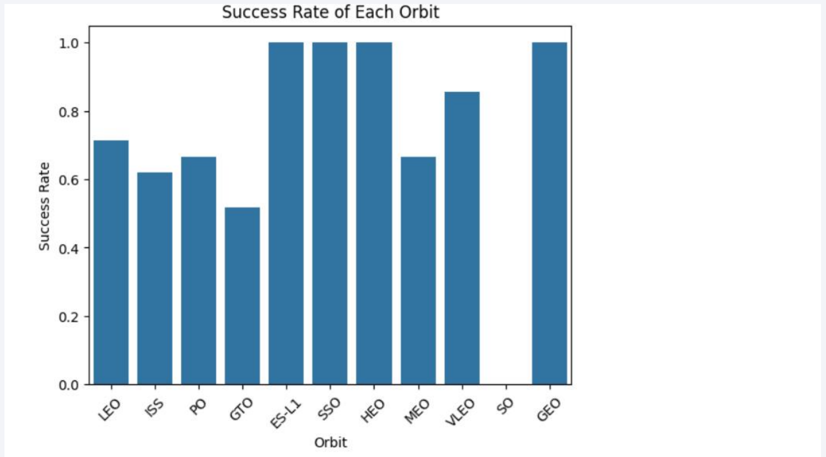
Now if you observe Payload Mass Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type

- Show the screenshot of the scatter plot with explanations

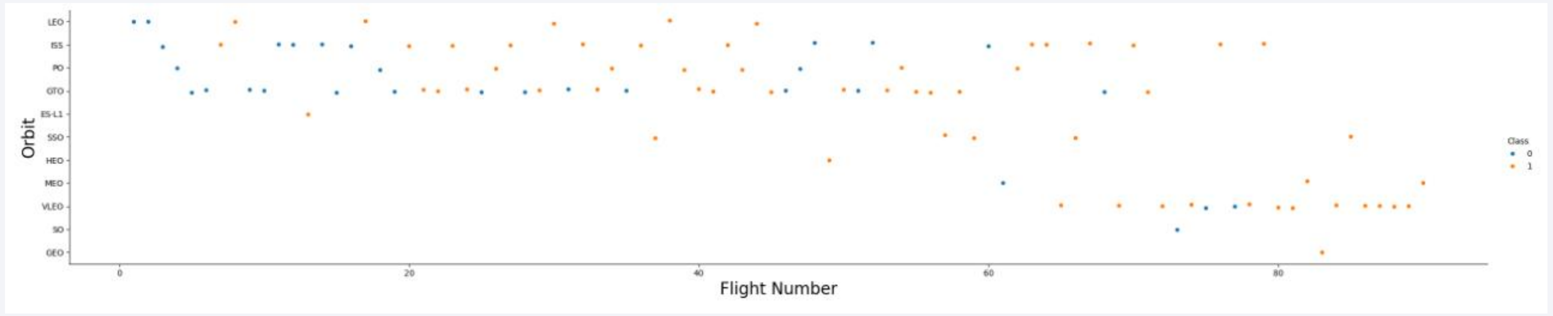




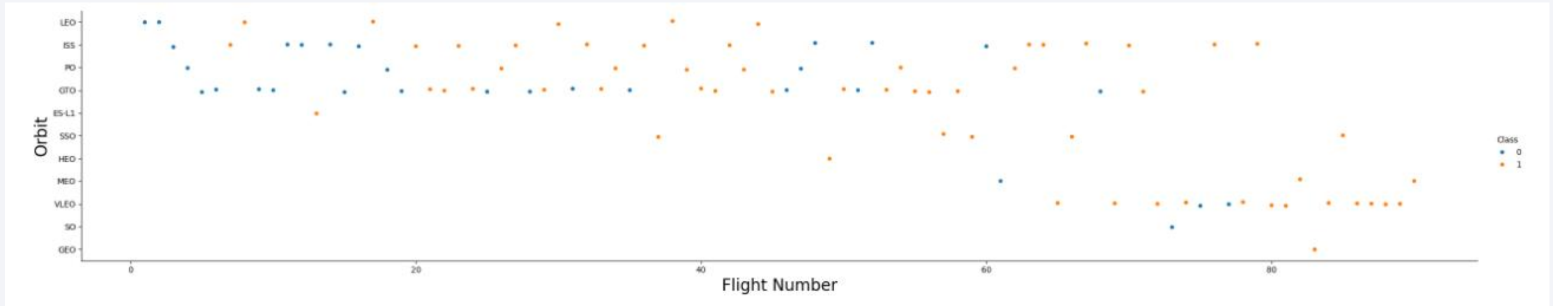
It can easily seen that SO Orbit has the lowest success rate and there has four orbit always succeed.

Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type



- Show the screenshot of the scatter plot with explanations

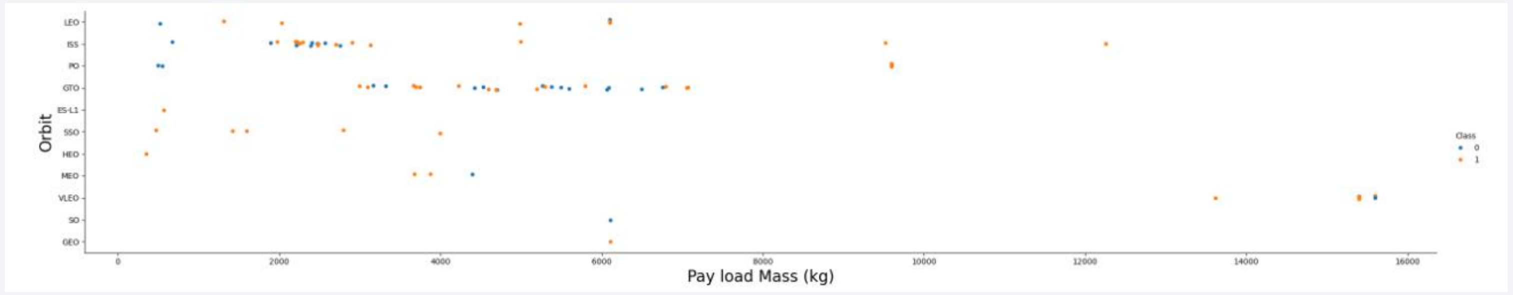


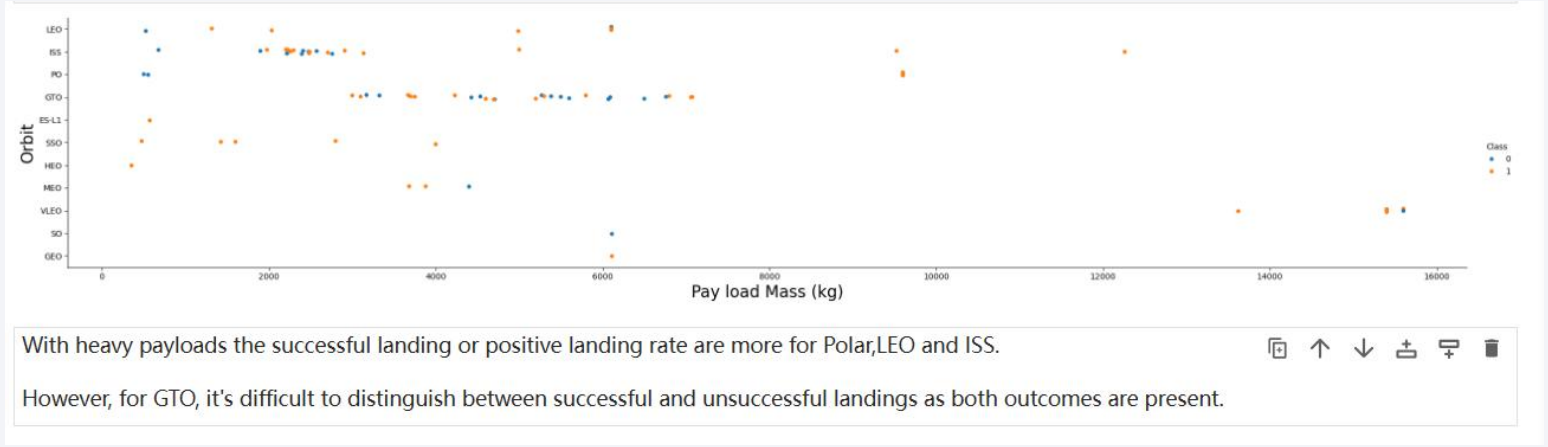
You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.

Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type

- Show the screenshot of the scatter plot with explanations



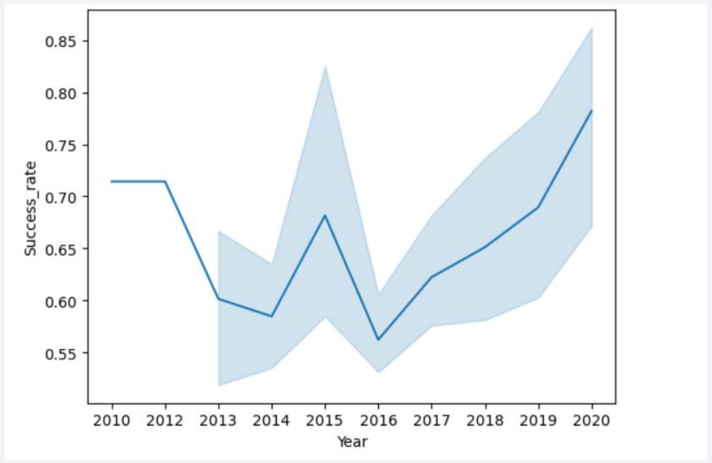


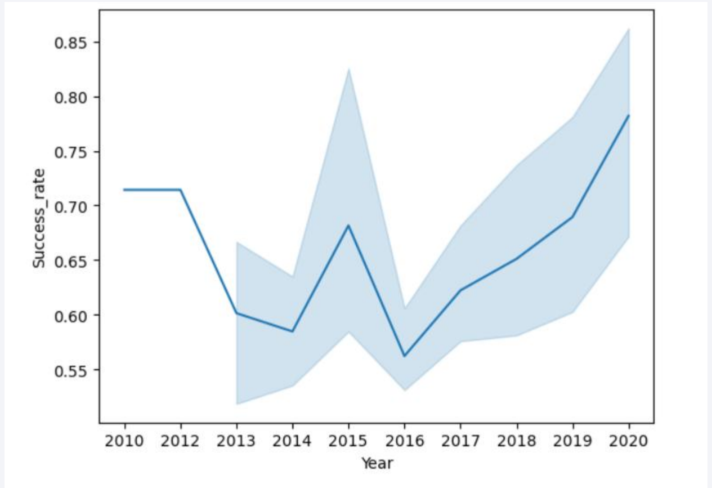
With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.

Launch Success Yearly Trend

- Show a line chart of yearly average success rate
- Show the screenshot of the scatter plot with explanations

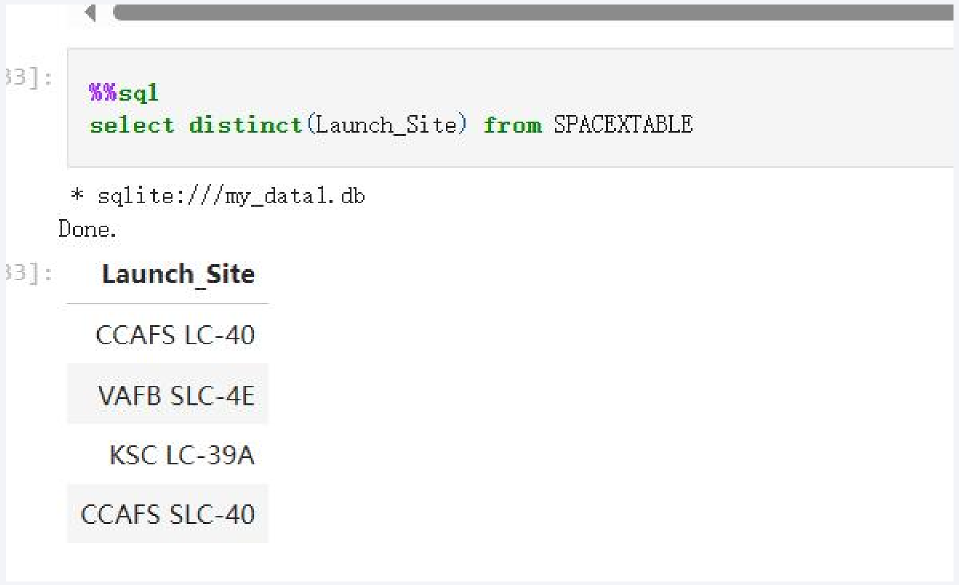




you can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

- Find the names of the unique launch sites
- Present your query result with a short explanation here



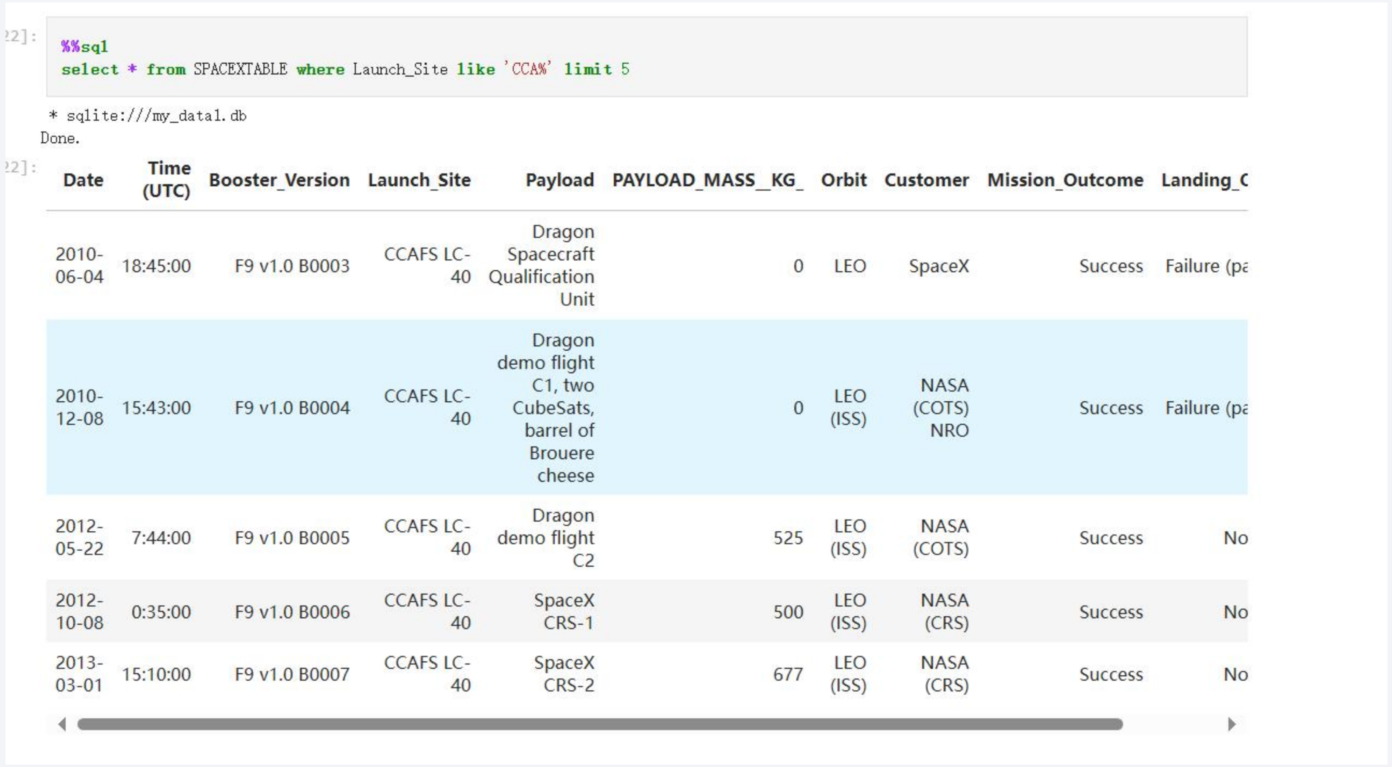
```
%%sql
select distinct(Launch_Site) from SPACEXTABLE
```

```
 * 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'

- Find 5 records where launch sites begin with `CCA`
- Present your query result with a short explanation here



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

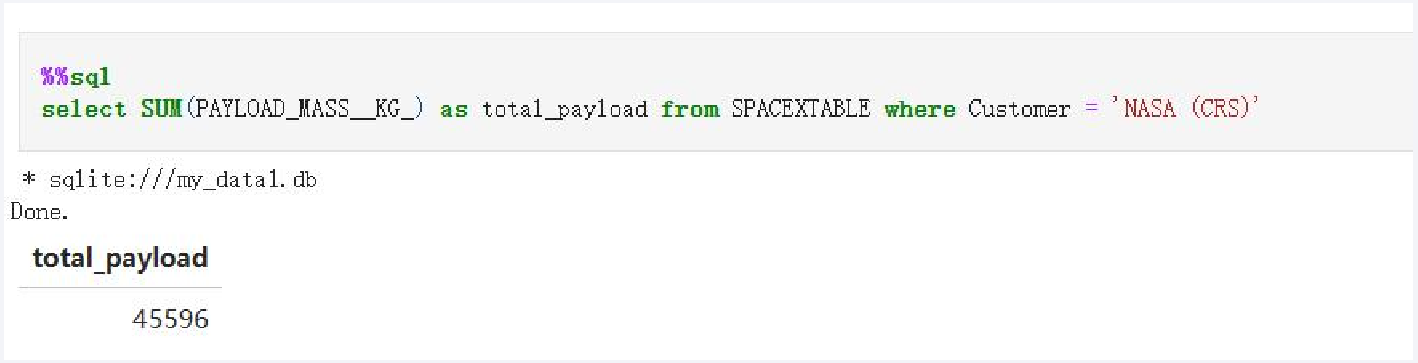
```
 * sqlite:///my_data1.db
Done.
```

[22]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_C
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (pa
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 (pa
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- Present your query result with a short explanation here



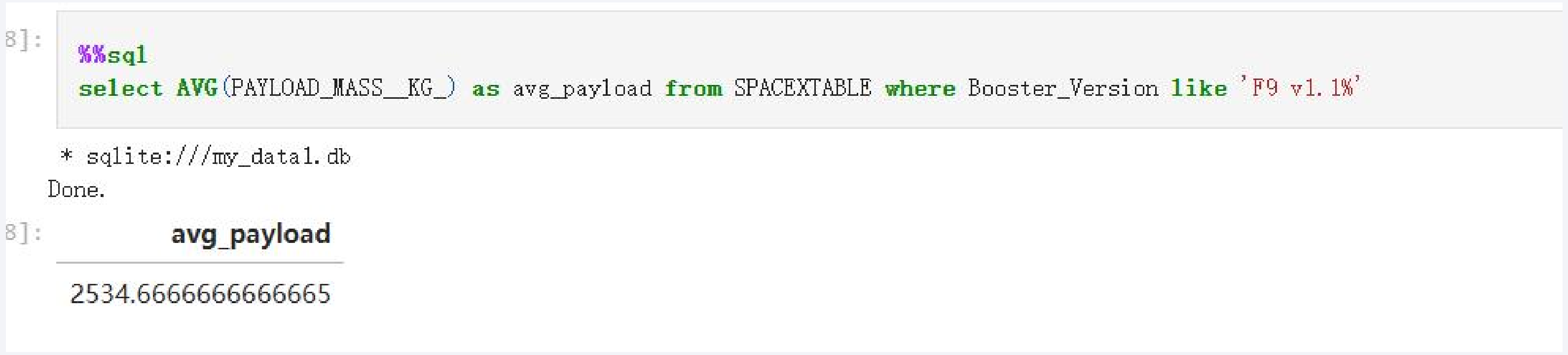
```
%%sql
select SUM(PAYLOAD_MASS__KG_) as total_payload from SPACEXTABLE where Customer = 'NASA (CRS)'
```

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

total_payload
45596

Average Payload Mass by F9 v1.1

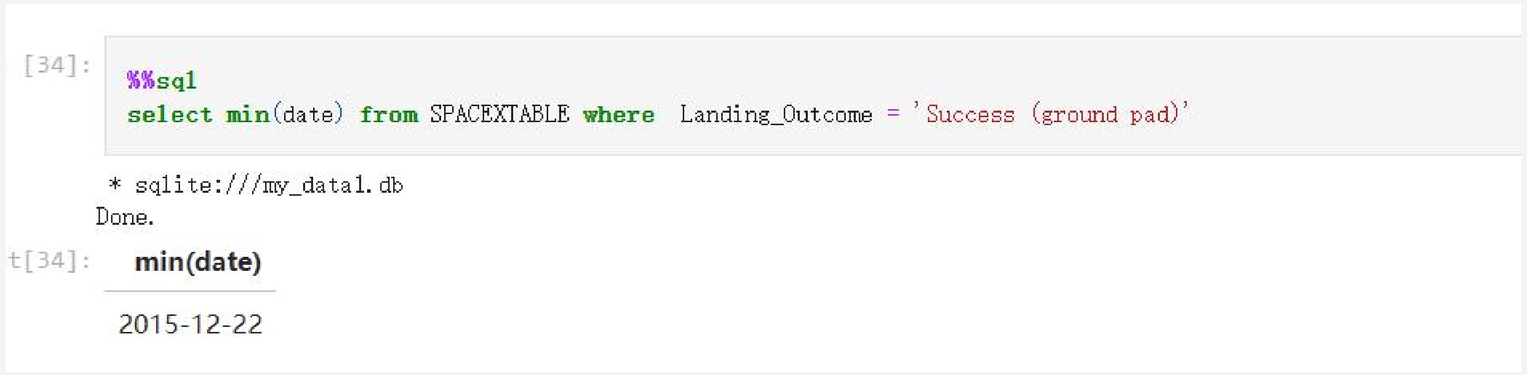
- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here



```
8]:  %%sql
     select AVG(PAYLOAD_MASS__KG_) as avg_payload from SPACEXTABLE where Booster_Version like 'F9 v1.1%'

      * sqlite:///my_data1.db
     Done.
8]:          avg_payload
       2534.6666666666665
```

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Present your query result with a short explanation here

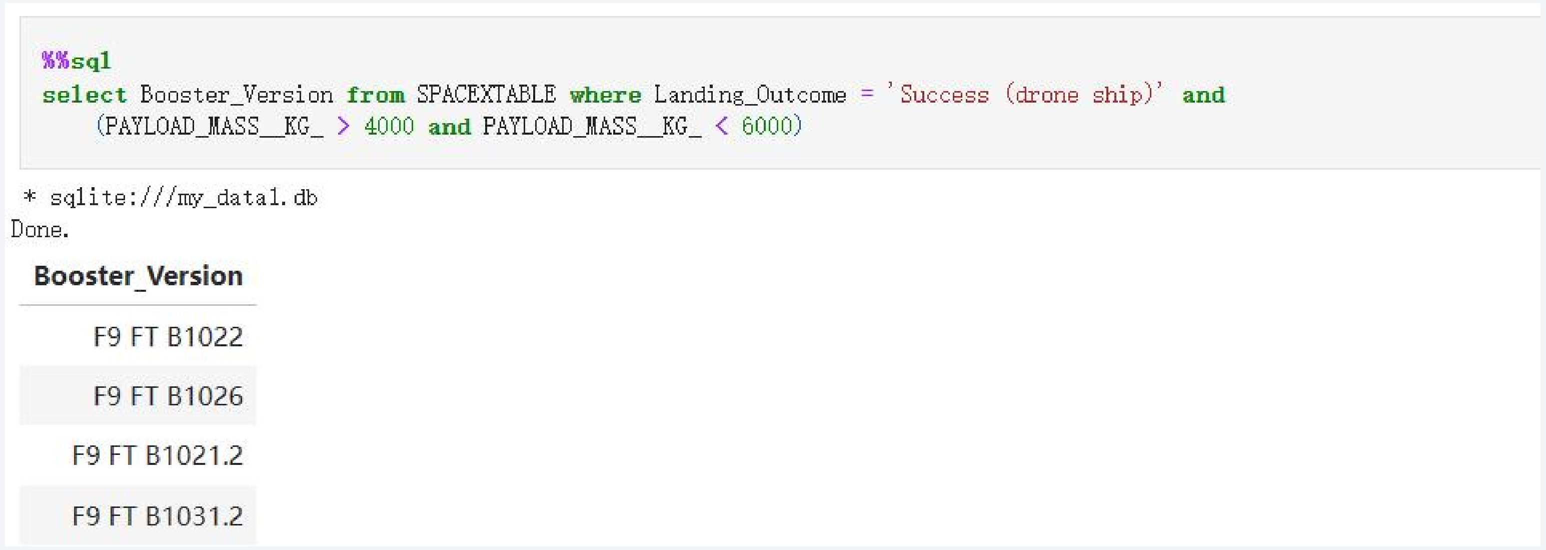


```
[34]: %%sql
      select min(date) from SPACEXTABLE where  Landing_Outcome = 'Success (ground pad)'

       * sqlite:///my_data1.db
      Done.
t[34]:  min(date)
        2015-12-22
```

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

- Present your query result with a short explanation here



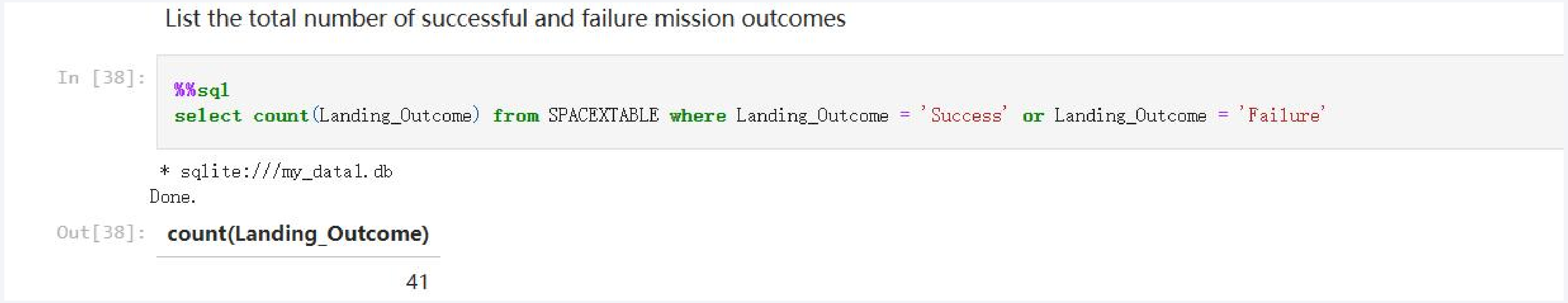
```
%%sql
select Booster_Version from SPACEXTABLE where Landing_Outcome = 'Success (drone ship)' and
    (PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000)
```

```
 * 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

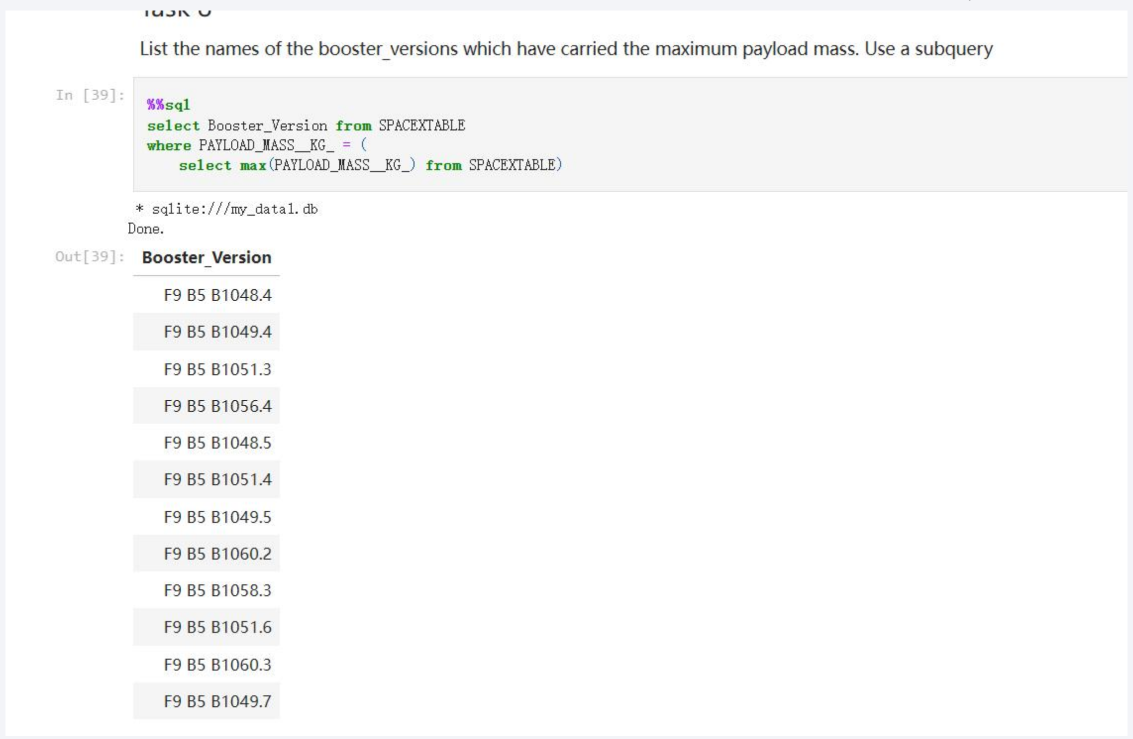
- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here


List the total number of successful and failure mission outcomes

```
In [38]: %%sql
         select count(Landing_Outcome) from SPACEXTABLE where Landing_Outcome = 'Success' or Landing_Outcome = 'Failure'

 * sqlite:///my_data1.db
Done.

Out[38]: count(Landing_Outcome)
                             41
```

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Present your query result with a short explanation here



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

In [39]:
```
%%sql
select Booster_Version from SPACEXTABLE
where PAYLOAD_MASS__KG_ = (
    select max(PAYLOAD_MASS__KG_) from SPACEXTABLE)
```

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

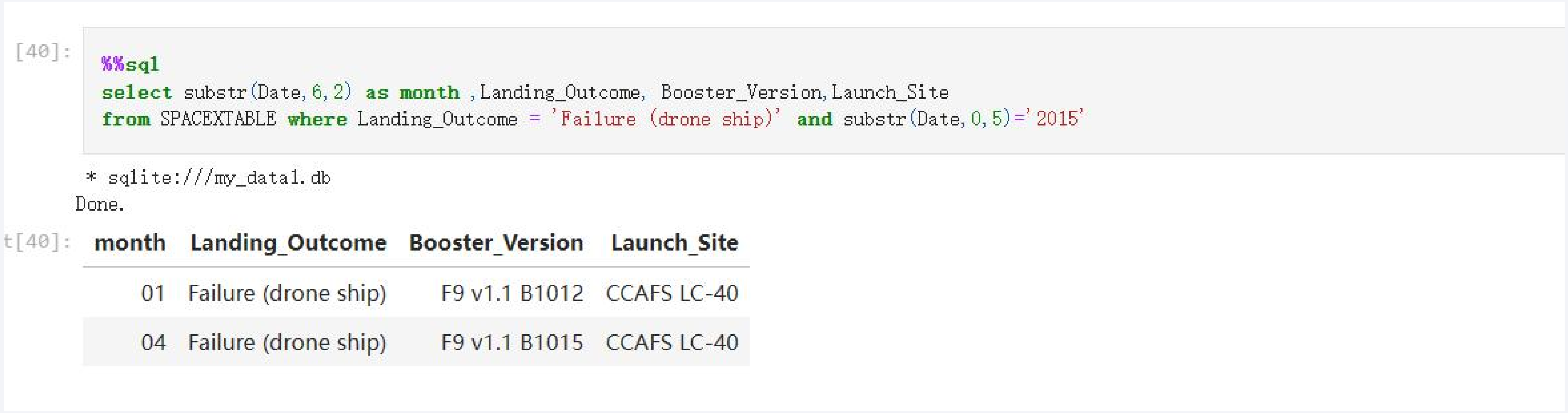
Out[39]:

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

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

- Present your query result with a short explanation here



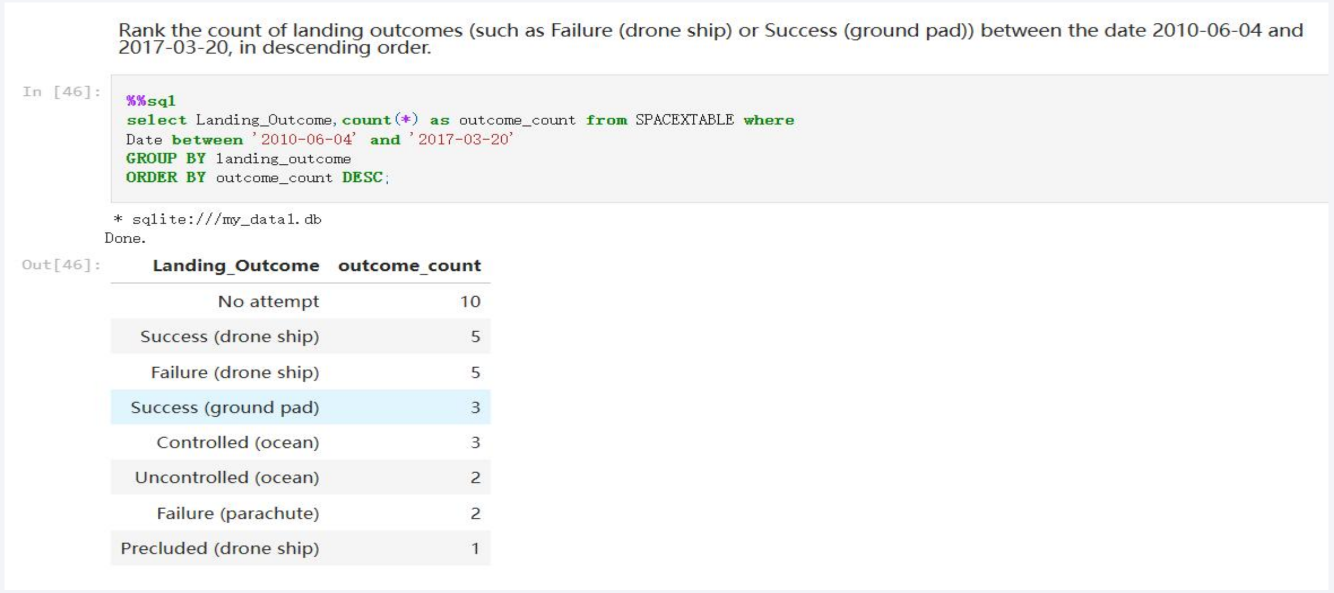
```
[40]: %%sql
      select substr(Date,6,2) as month ,Landing_Outcome, Booster_Version,Launch_Site
      from SPACEXTABLE where Landing_Outcome = 'Failure (drone ship)' and substr(Date,0,5)='2015'

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

t[40]:

month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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
- Present your query result with a short explanation here



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 [46]:
```
%%sql
select Landing_Outcome, count(*) as outcome_count from SPACEXTABLE where
Date between '2010-06-04' and '2017-03-20'
GROUP BY landing_outcome
ORDER BY outcome_count DESC;
```

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

Out[46]:

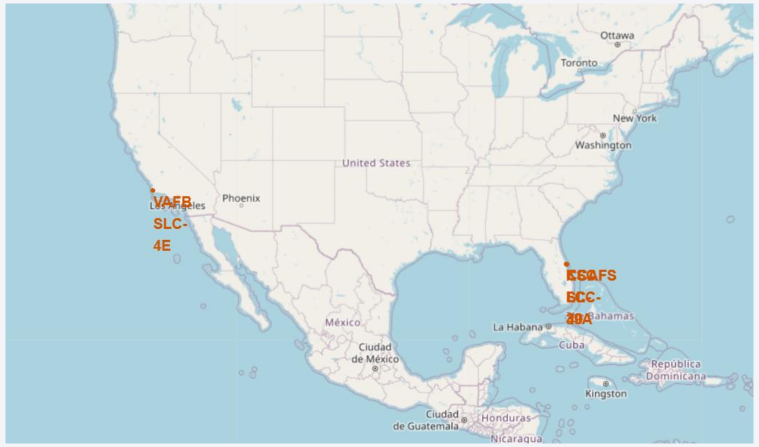
Landing_Outcome	outcome_count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

<Each launch site on the site map>

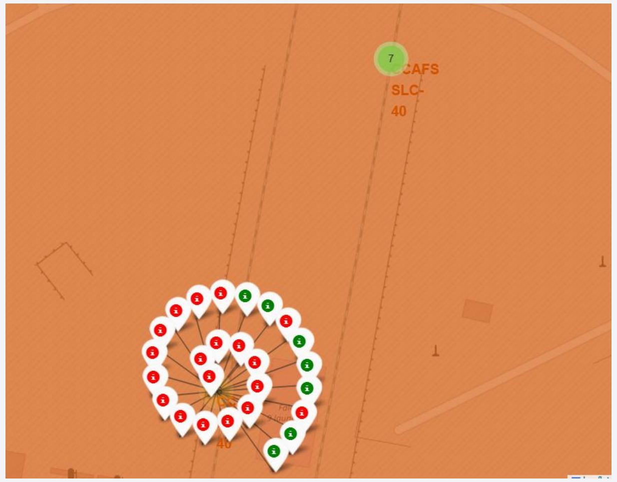
- Replace <Folium map screenshot 1> title with an appropriate title
- Explore the generated folium map and make a proper screenshot to include all launch sites' location markers on a global map



- Explain the important elements and findings on the screenshot
  - I found that the launch sites are both near the coast lines.

<Color for every Launch Sites>

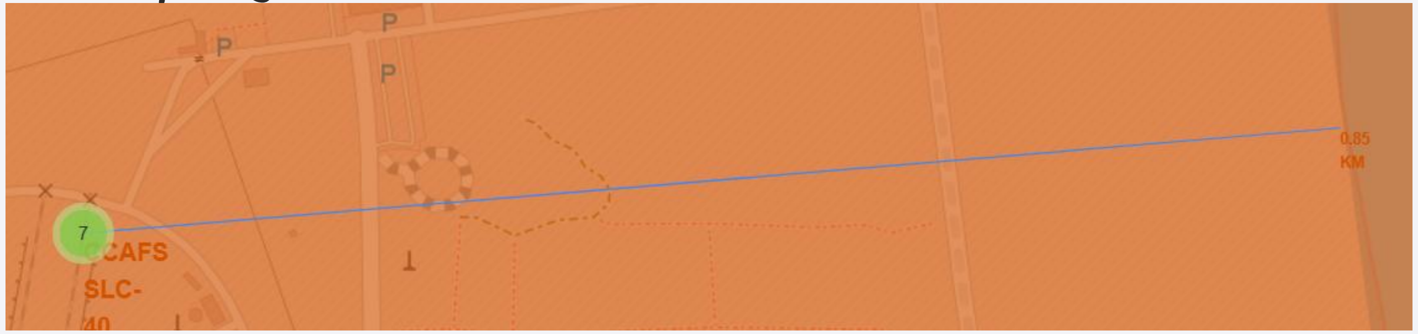
- Replace <Folium map screenshot 2> title with an appropriate title
- Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map



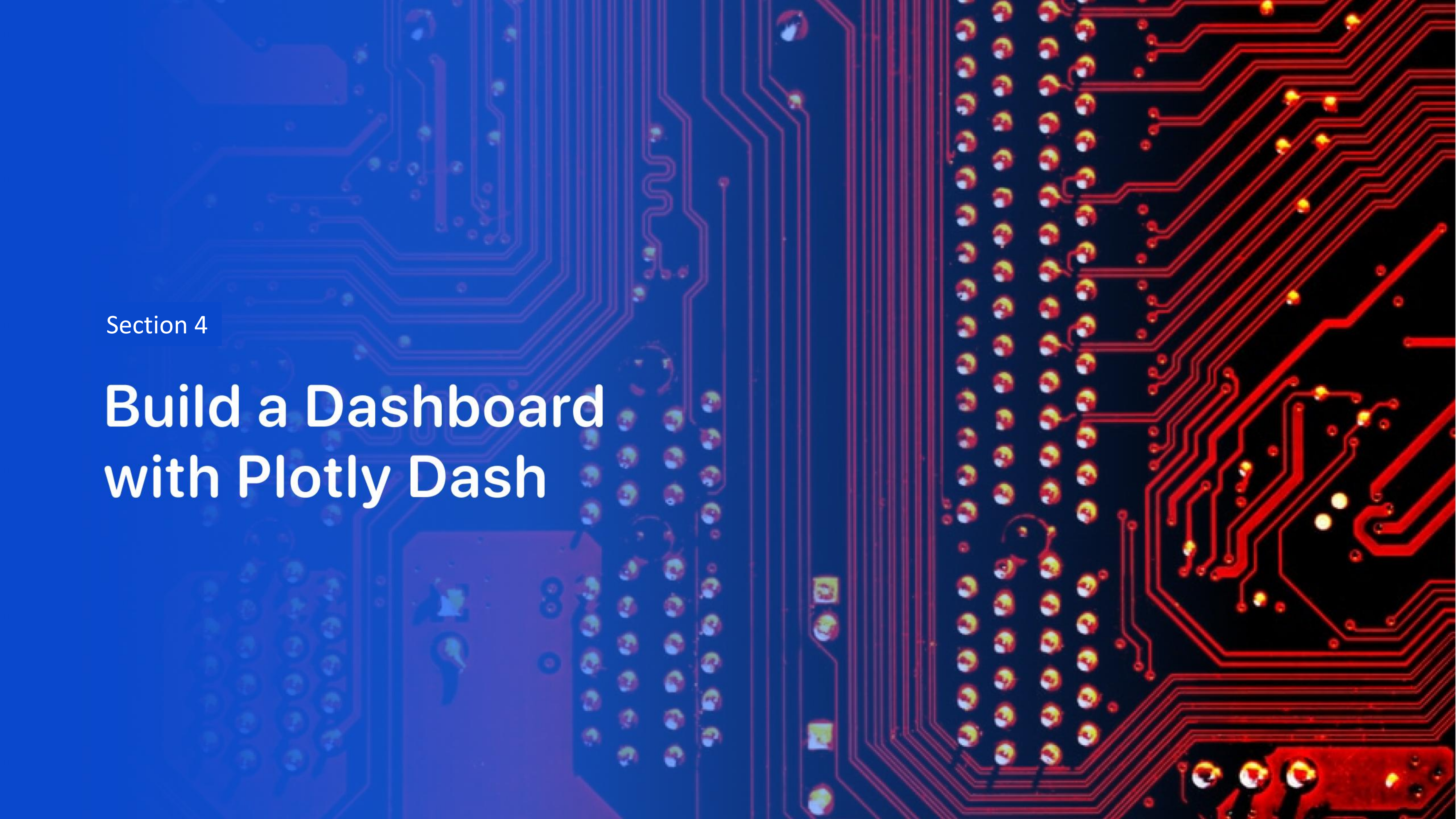
- Explain the important elements and findings on the screenshot
  - KsC LC-39A sites has higher success rate than others

< Distances between a launch site to its proximities>

- Replace <Folium map screenshot 3> title with an appropriate title
- Explore the generated folium map and show the screenshot of a selected launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed



- Explain the important elements and findings on the screenshot
  - Using this Display function, it can easily decide the comprehensive situations for a launch site.

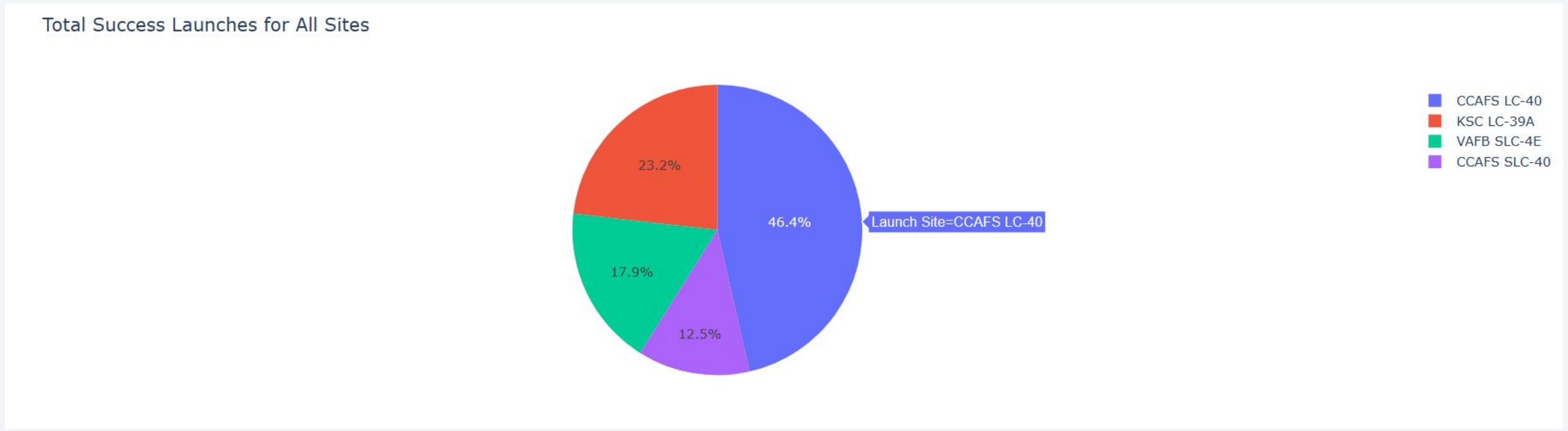


Section 4

Build a Dashboard with Plotly Dash

<Pie chart of launch success count for all sites>

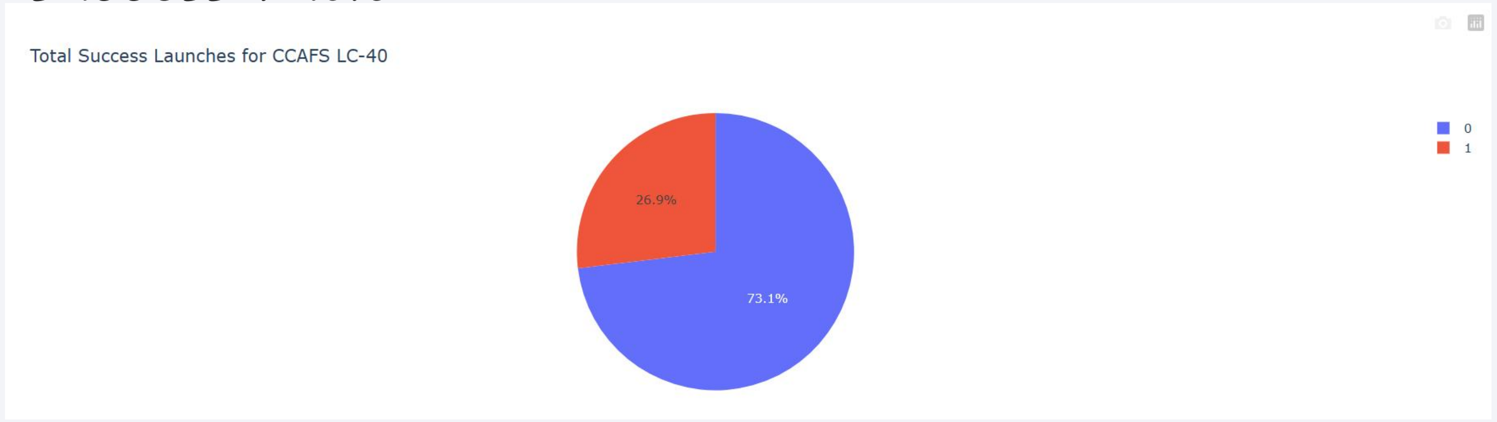
- Replace <Dashboard screenshot 1> title with an appropriate title
- Show the screenshot of launch success count for all sites, in a piechart



- Explain the important elements and findings on the screenshot
  - CCAFS LC-40 has the highest success rate, and CCAFS SLC-40 has the lowest success rate

< Launch site with highest launch success rate>

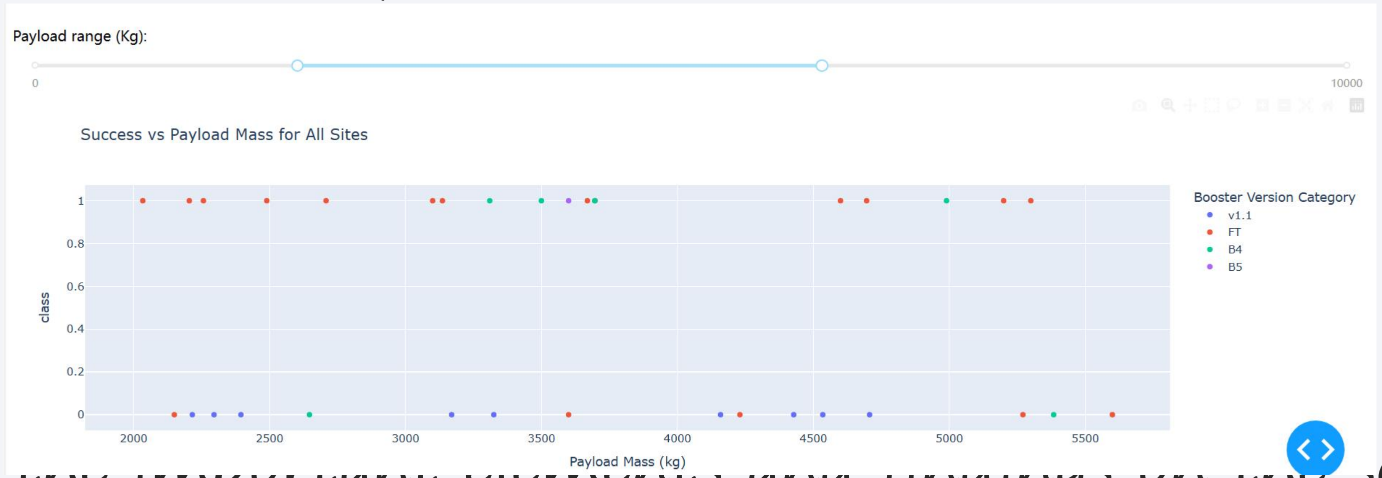
- *Replace <Dashboard screenshot 2> title with an appropriate title*
- *Show the screenshot of the piechart for the launch site with highest launch success ratio*



- *Explain the important elements and findings on the screenshot*
  - Almost three quarters of the Launches for CCAFS LC-40 was successful.

<Scatter plot for all sites>

- 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.
    - FT performs good between 2k-4k,B4 seems no exact changes during different payloads.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

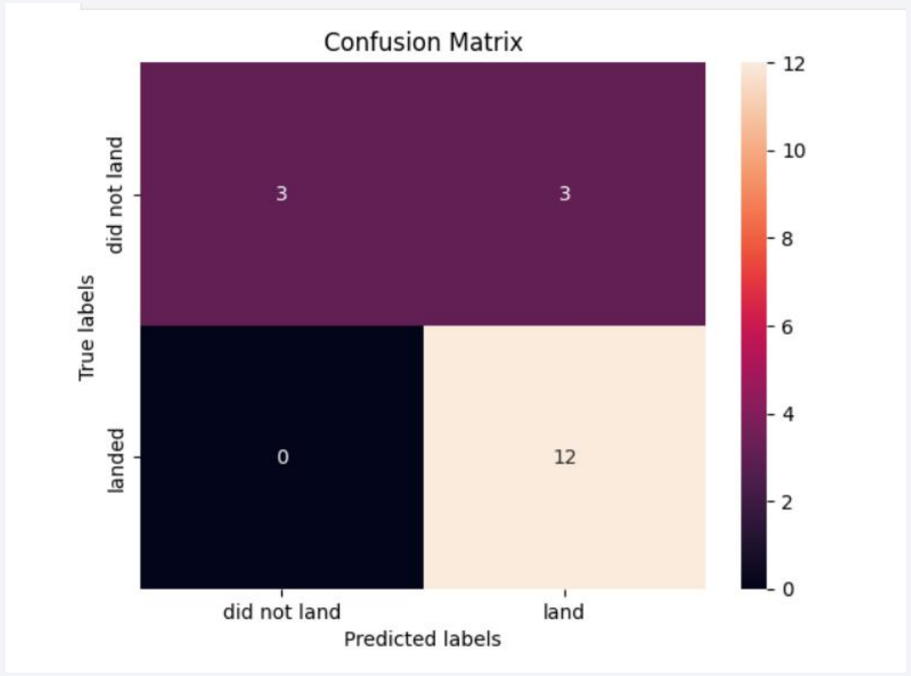
- Visualize the built model accuracy for all built classification models, in a bar chart

- Find which model has the highest classification accuracy
  - Decision Tree

Too tire to get a bar chart, just forgive me for this little shortcoming.

Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation



It perform best in train set and all models perform the same on test sets.

we dont have enough test data i guess.

Conclusions

- Point 1 This projects inclueded all the basic skills for a Data scientist
- Point 2 Using webscraping and requets function, we get the initial data and wash the format for the further processing.
- Point 3 Using the different charts and Dashboards, making the results clear, we can easily see the significants of different features.
- Point 4 By acheving different methods and using GridSearchCV to find the best parameters, we get the best model for the final results.

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!