IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Pengfei Guan
19 June, 2023

Outline

Executive Summary

- Summary of methodologies

 - Data collection and data wrangling (via API, web scraping)

 - Exploratory Data Analysis (EDA) using SQL, Pandas, and Matplotlib

 - Data Visualization with Folium and dashboard (Plotly Dash)

 - Predictive analysis (classification)

- Summary of all results

 - EDA analysis results

 - Interactive map and dashboard

 - Predictive analysis

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 because SpaceX can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

- Problems you want to find answers

- The features which influence the launch outcome (successful or failure)

- What conditions will make the launches achieve the best outcome (successful landing)

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API
  - Web scraping from Wikipedia
- Perform data wrangling
  - Dealing with missing values
  - Dropping unnecessary columns
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

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

Data Collection – SpaceX API:

1. Import necessary libraries (requests, pandas, numpy, etc.). >>> 2. Define auxiliary functions (getBoosterVersion, getLaunchSite, etc.). >>> 3. use Get request to request and parse the SpaceX launch data. >>> 4. use pandas to construct a data frame for the requested SpaceX launch data. >>> 5. decode the response data as Json (.json()) and turn it to Pandas dataframe (.json_normalize()). >>> 6. filter the data and create a Pandas dataframe with the required columns.

Data Collection – Web Scraping:

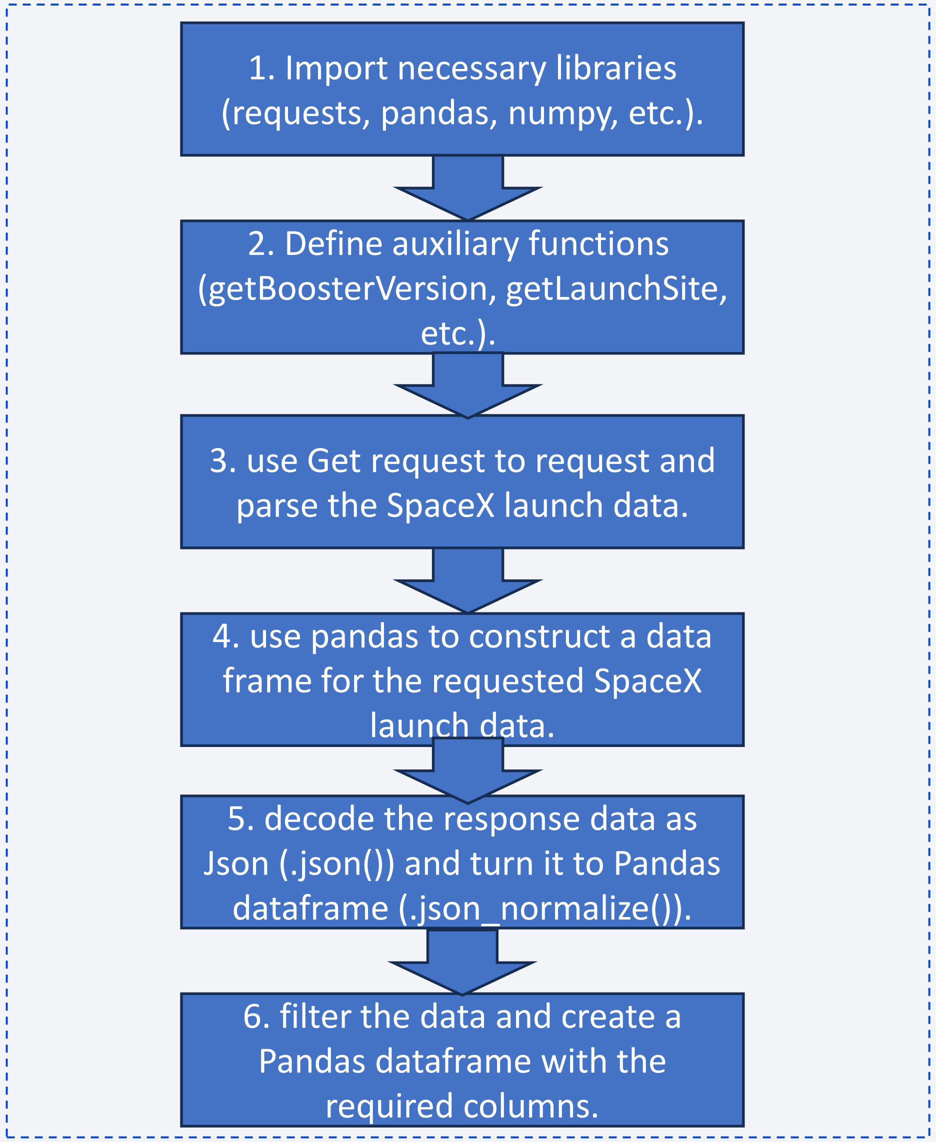
1. Import necessary libraries (requests, pandas, beautifulsoup, etc.). >>> 2. Define auxiliary functions (booster_version, landing_status, etc.). >>> 3. Request the Falcon9 Launch Wiki page from its URL. >>> 4. Extract all column/variable names from the HTML table header. >>> 5. Create a data frame by parsing the launch HTML tables.

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.

- URL: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

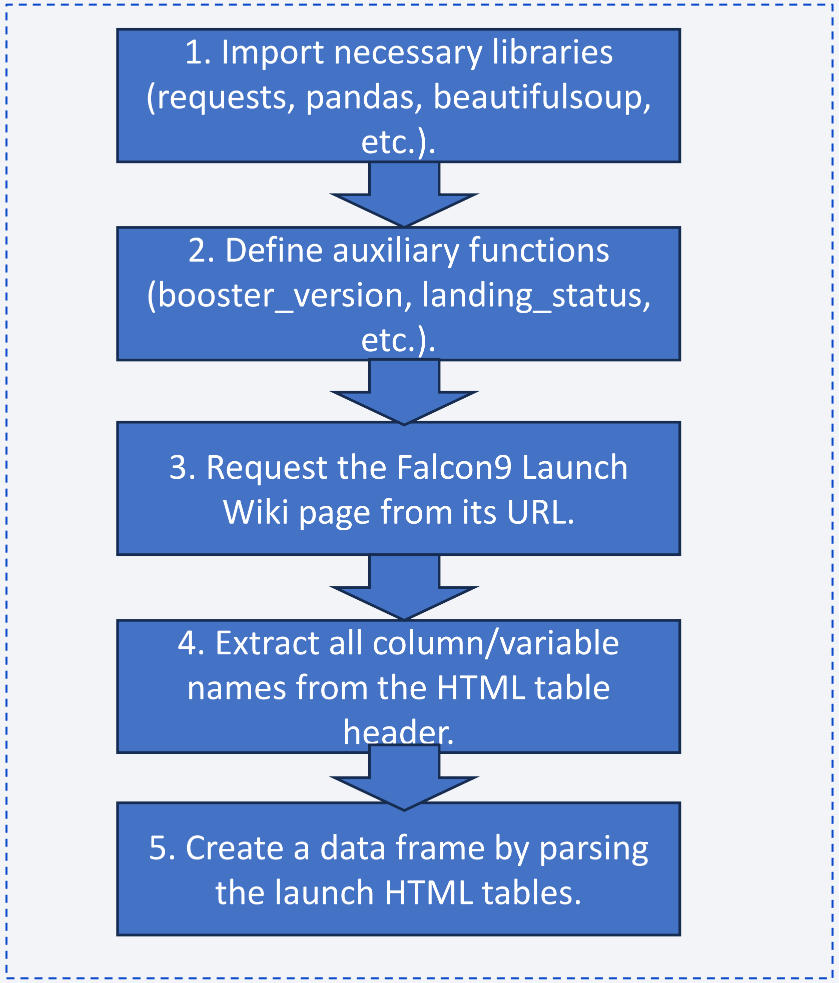

1. Import necessary libraries (requests, pandas, numpy, etc.).
2. Define auxiliary functions (getBoosterVersion, getLaunchSite, etc.).
3. use Get request to request and parse the SpaceX launch data.
4. use pandas to construct a data frame for the requested SpaceX launch data.
5. decode the response data as Json (.json()) and turn it to Pandas dataframe (.json_normalize()).
6. filter the data and create a Pandas dataframe with the required columns.

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

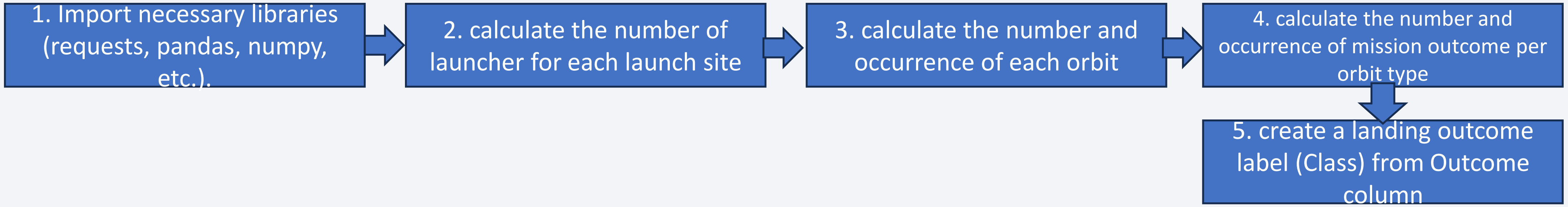
- URL: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/jupyter-labs-webscraping.ipynb


1. Import necessary libraries (requests, pandas, beautifulsoup, etc.).
2. Define auxiliary functions (booster_version, landing_status, etc.).
3. Request the Falcon9 Launch Wiki page from its URL.
4. Extract all column/variable names from the HTML table header.
5. Create a data frame by parsing the launch HTML tables.

Data Wrangling

- Describe how data were processed
- You need to present your data wrangling process using key phrases and flowcharts

Using the data set generated from the previous step, date collection, to proceed with the data wrangling including calculate the number of launcher for each launch site, calculate the number and occurrence of each orbit, calculate the number and occurrence of mission outcome per orbit type, and create a landing outcome label from Outcome column. Creating a landing outcome label (Class) is necessary for the following Exploratory Data Analysis (EDA)


1. Import necessary libraries (requests, pandas, numpy, etc.).
2. calculate the number of launcher for each launch site
3. calculate the number and occurrence of each orbit
4. calculate the number and occurrence of mission outcome per orbit type
5. create a landing outcome label (Class) from Outcome column

- Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose

URL: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts

Catplot chart, scatter chart, bar chart, and line chart are all the plotted charts in the data visualization of the EDA. The catplot and scatter chart help to view the relationship between the launch sites and flight numbers. Besides, scatter chart also presents the relationship between launch sites and pay load mass (kgs) as well as the relationship between orbits and flight numbers (pay load mass). Bar chart provides the visual among different orbits, different orbits have different launch successful rates, the bar chart can compare and contrast the difference. Lastly, the line chart demonstrates the trend of successful launch rate since 2010 till 2020.

- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose

URL: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL – Part 1

- Using bullet point format, summarize the SQL queries you performed

- Task 1: Display the names of the unique launch sites in the space mission
- %sql SELECT DISTINCT Launch_Site FROM SPACEXTBL

- Task 2: Display 5 records where launch sites begin with the string 'CCA'
- %sql SELECT Launch_Site FROM SPACEXTBL WHERE Launch_Site LIKE '%CCA%'

- Task 3: Display the total payload mass carried by boosters launched by NASA (CRS)
- %sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Customer == 'NASA (CRS)'

- Task 4: Display average payload mass carried by booster version F9 v1.1
- %sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version == 'F9 v1.1’

EDA with SQL – Part 2

- Using bullet point format, summarize the SQL queries you performed

- Task 5: List the date when the first succesful landing outcome in ground pad was acheived.
- %sql SELECT MIN(Date) FROM SPACEXTBL WHERE Landing_Outcome == 'Success (ground pad)'

- 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 Booster_Version FROM SPACEXTBL WHERE Landing_Outcome == 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000

- Task 7: List the total number of successful and failure mission outcomes
- %sql SELECT COUNT(Mission_Outcome) FROM SPACEXTBL WHERE Mission_Outcome == 'Success' OR Mission_Outcome == 'Failure'

EDA with SQL – Part 3

- Using bullet point format, summarize the SQL queries you performed

- Task 8: List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- %sql SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ == (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)

- 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, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year. Example: Extract a substring from a string (start at position 4, extract 3 characters): SELECT SUBSTR("SQL Tutorial", 5, 3) AS ExtractString)
- %sql SELECT substr(Date, 4, 2) as 'month', Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTBL WHERE substr(Date, 7, 4) = '2015' AND Landing_Outcome = 'Failure (drone ship)'

EDA with SQL – Part 4

- Using bullet point format, summarize the SQL queries you performed

  - Task 10: Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order. (Note: Example: Extract a substring from a string (start at position 4, extract 3 characters): SELECT SUBSTR("SQL Tutorial", 5, 3) AS ExtractString)
  - %sql SELECT Date, Landing_Outcome FROM SPACEXTBL WHERE (Landing_Outcome LIKE '%Success%') AND ((substr(Date,1,2) > '04' AND substr(Date,4,2) BETWEEN '06' AND '12' AND substr(Date,7,4)='2010') OR (substr(Date,7,4) BETWEEN '2011' AND '2016') OR (substr(Date,1,2) < '20' AND substr(Date,4,2) BETWEEN '01' AND '03' AND substr(Date,7,4)='2017')) ORDER BY substr(Date,7,4) DESC

- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

URL: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/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

Objects marker, circle, and line are created and added to the folium map.

- Explain why you added those objects

The marker objects show the names of launch sites, the colors of successful/failure launches, and the distance number. The circle object highlights the areas of launch sites. The line object is used to plot a line connecting a selected point (near railway, highway, coastline, etc.) from a launch site.

- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

URL: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/lab_jupyter_launch_site_location-jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard

Dropdown menu, pie chart, range slider, and scatter plot are added to the dashboard.

- Explain why you added those plots and interactions

The dropdown menu can select all launch sites and each launch site. The pie chart can show the success percentage for all launch sites and each launch site. The range slider can slice the wanted range for the pay load (kgs). The scatter plot can show the correlation between payload and success launch for all sites an each site. All these plots/graph above can operate interactively and simultaneously.

- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

URL: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
- You need present your model development process using key phrases and flowchart

Data preparation (load, normalize, and split (train/test) data) >>> Model preparation (ML algorithms selection, set parameters for each algorithm to GridSearchCV, Training GridSearchModel models with training dataset) >>> Model evaluation (Get best hyperparameters for each type of model, Compute accuracy for each model with test dataset, Plot Confusion Matrix) >>> Model comparison (Comparison of models according to their accuracy, The model with the best accuracy will be chosen (see Notebook for result))

- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

URL: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/SpaceX_Machine_Learning_Prediction_Part_5_jupyterlite.ipynb

Results

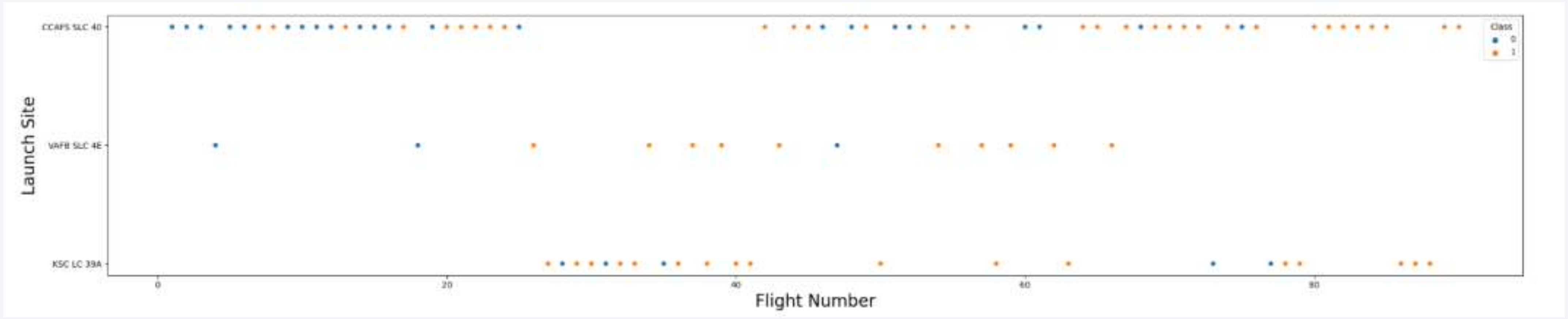
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site
- Show the screenshot of the scatter plot with explanations



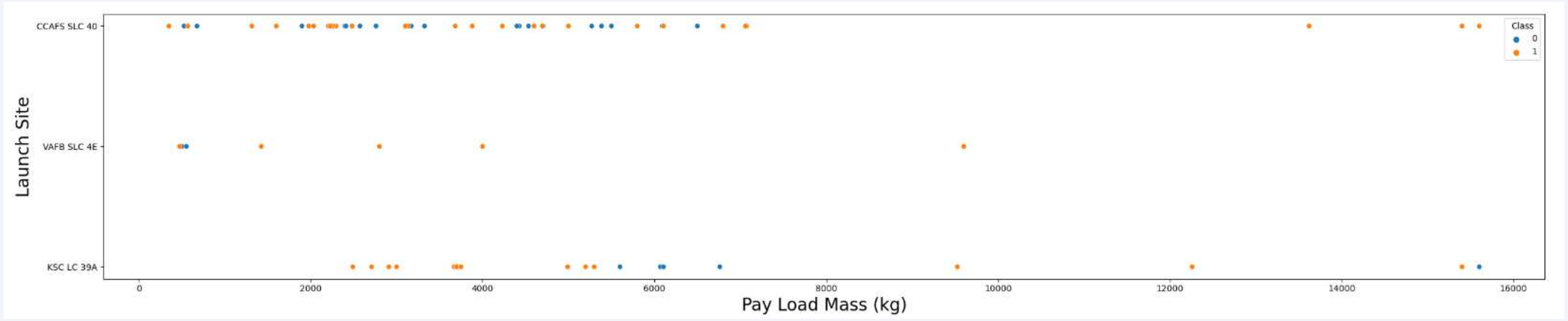
Explanations:

# CCAFS LC-40 launch site takes more launch tasks than KSC LC-39A and VAFB SLC 4E launch sites.

# Different launch sites have different success rates. CCAFS LC-40 has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.

Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site
- Show the screenshot of the scatter plot with explanations



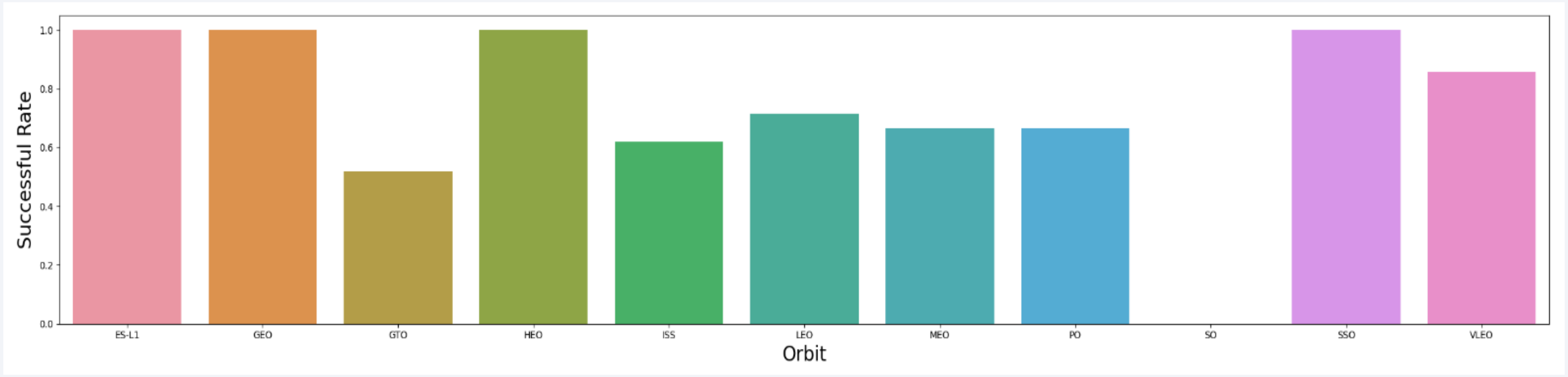
Explanations:

# CCAFS LC-40 launch site takes more launch tasks with the pay load mass less than 8000 kgs.

# There are no rockets launched for heavy payload mass (greater than 10000 kgs) in VAFB-SLC launch site .

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



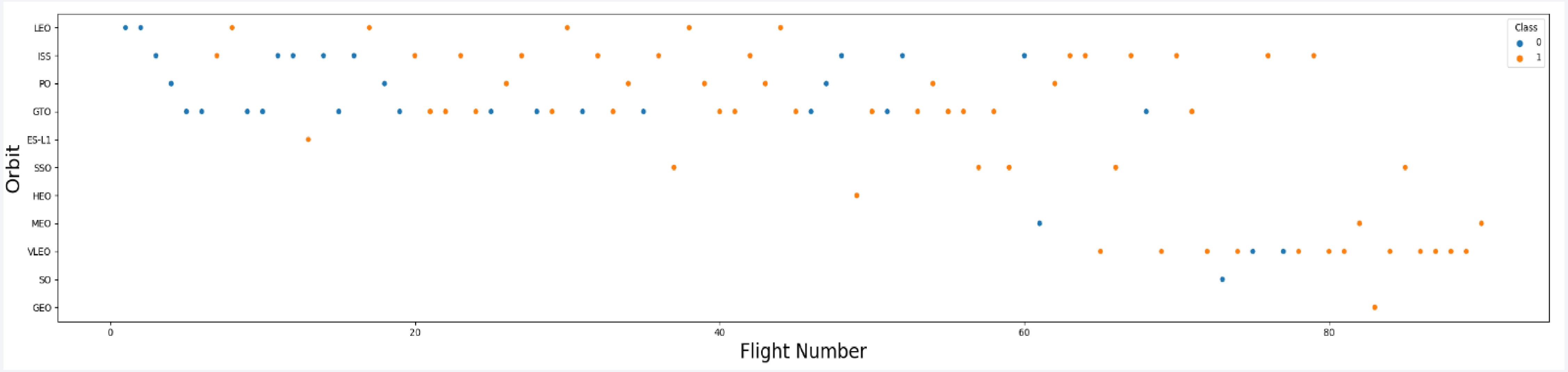
Explanations:

# ES-L1, GEO, HEO, and SSO orbits have the high success rate which is 100%.

# SO orbit has 0 success rate.

Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type
- Show the screenshot of the scatter plot with explanations



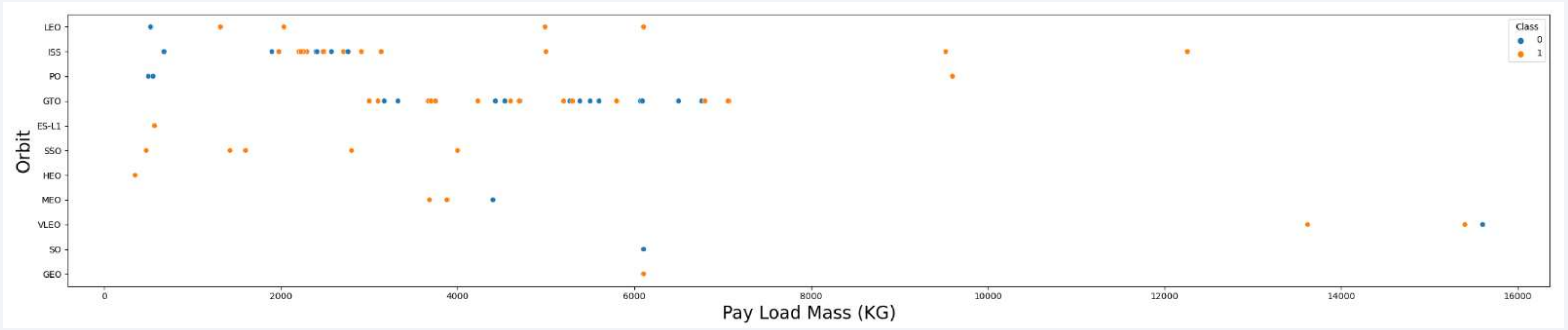
Explanations:

# the Success rate appears related to the number of flights in LEO orbit; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

# there are many launches to VLEO orbit after the 60 flights, and its success rate is high.

Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type
- Show the screenshot of the scatter plot with explanations



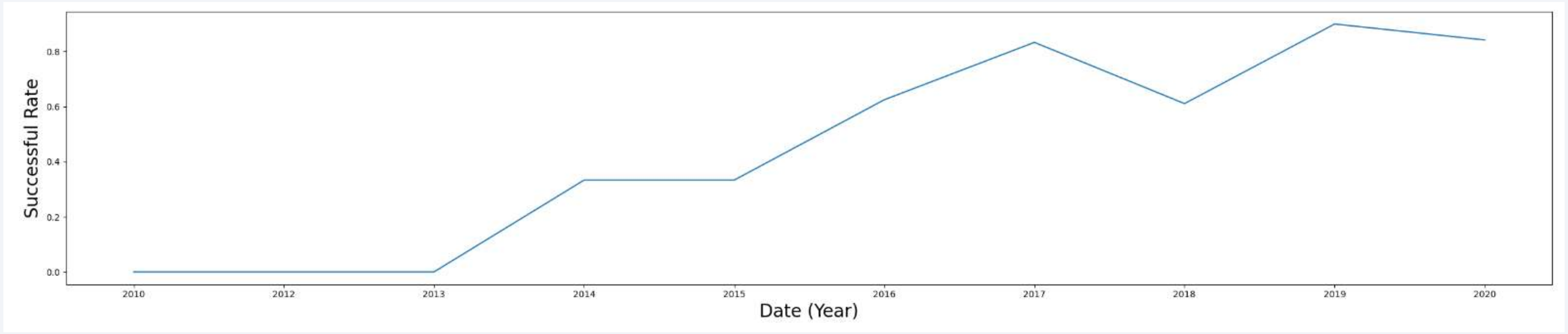
Explanations:

# With heavy pay loads, the successful landing or positive landing rate are more for PO, LEO, and ISS orbits.

# For GTO orbit, it is hard to distinguish the success well as both positive landing rate and negative landing rate are both there.

Launch Success Yearly Trend

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

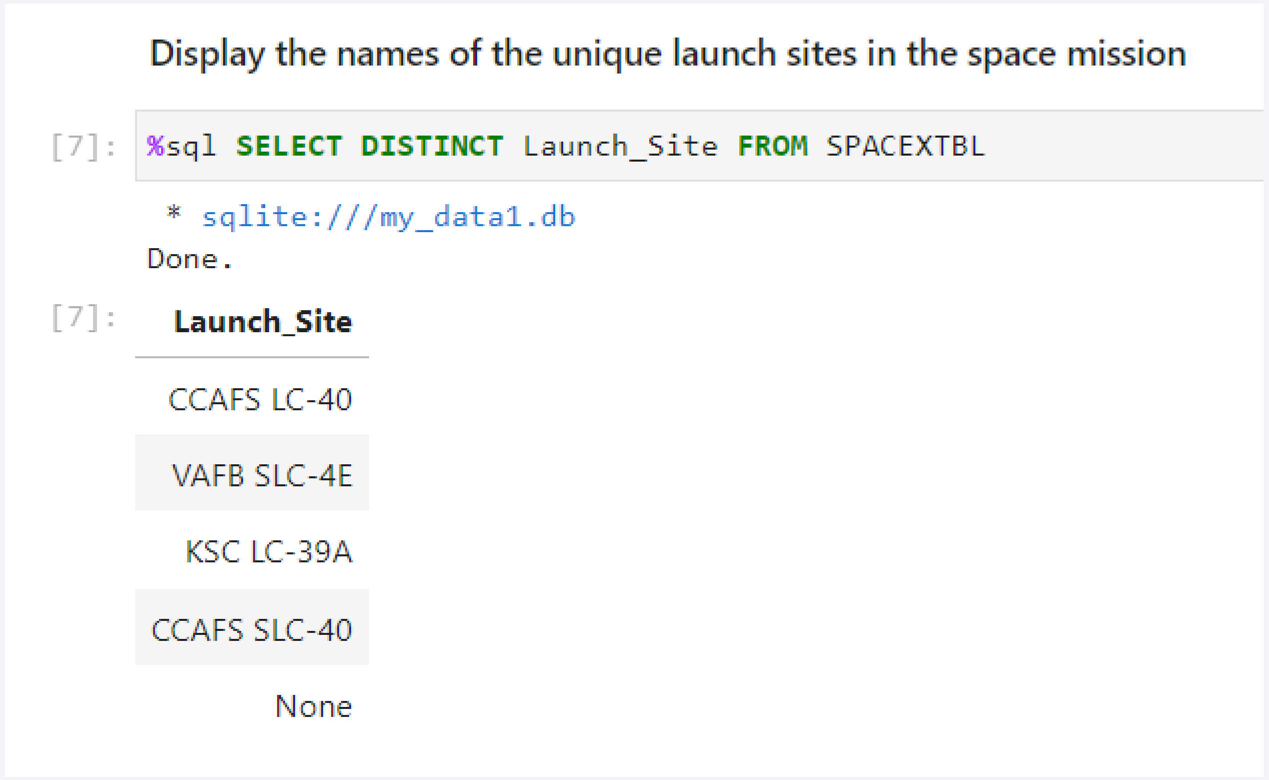


Explanations:

# the successful rate kept increasing since 2013 till 2020.

All Launch Site Names

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



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

```
[7]: %sql SELECT DISTINCT Launch_Site FROM SPACEXTBL
```

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

[7]:

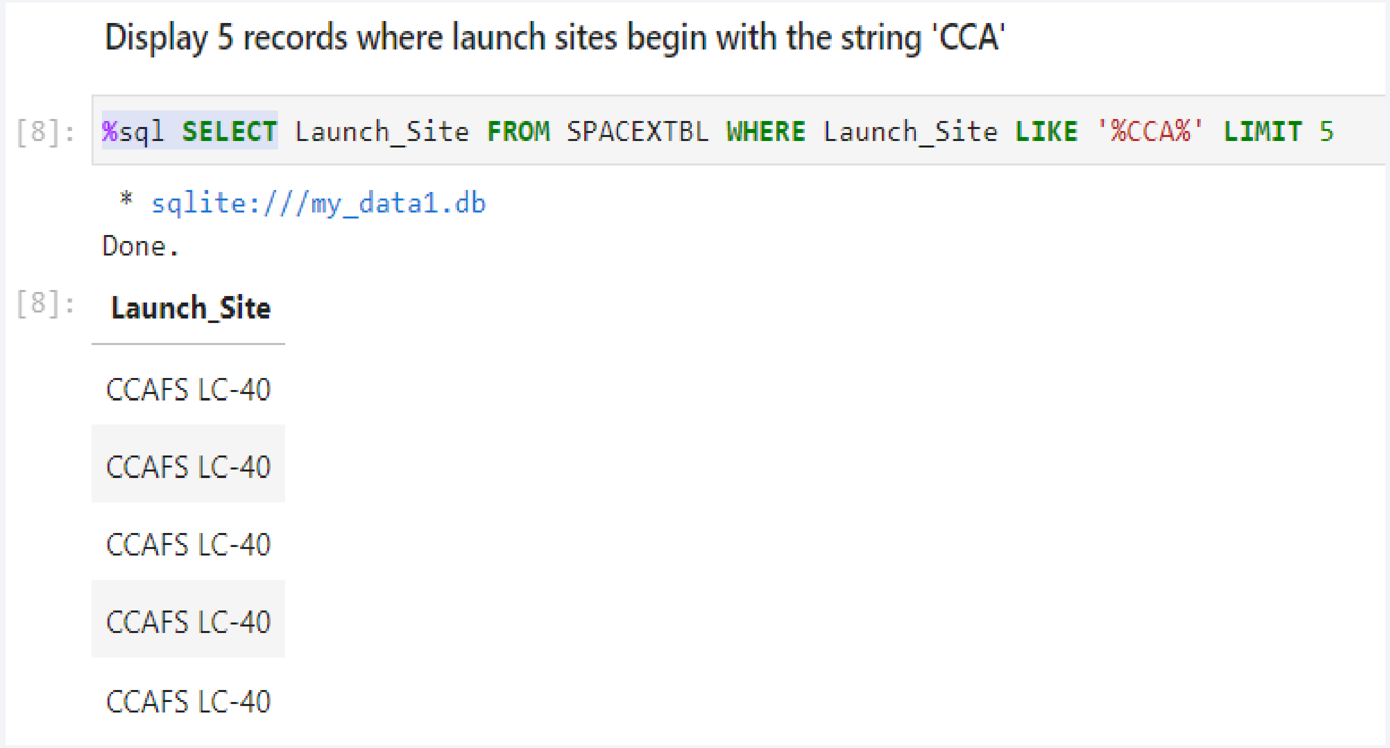
Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
None

Explanation:

# Command DISTINCT is necessary for the query to present unique launch site names.

Launch Site Names Begin with 'CCA'

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



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

```
[8]: %sql SELECT Launch_Site FROM SPACEXTBL WHERE Launch_Site LIKE '%CCA%' LIMIT 5
```

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

[8]:

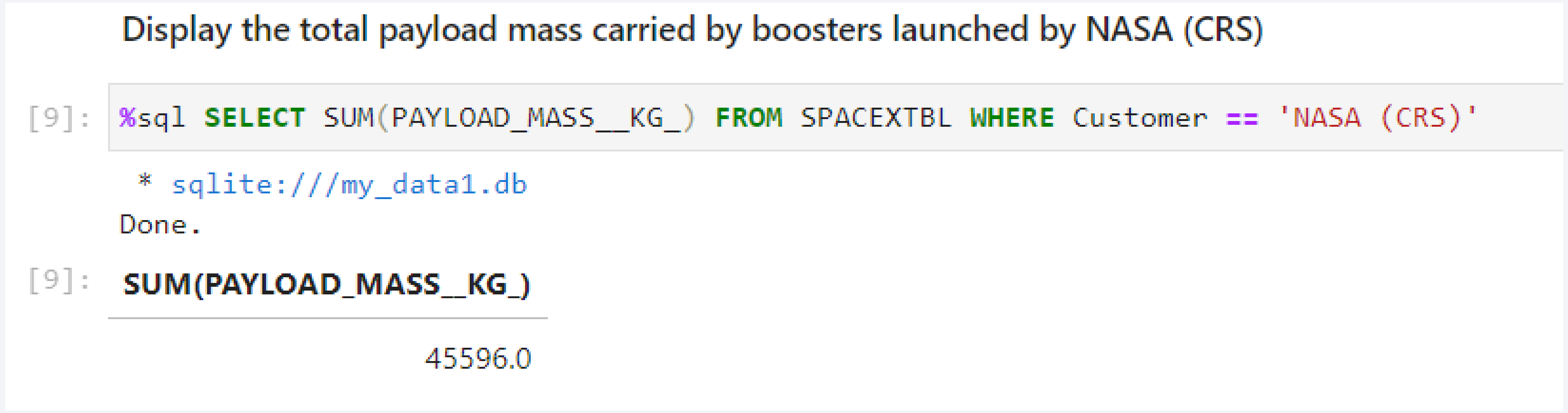
Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Explanation:

# the LIMIT 5 command and LIKE command with ‘%%’ signs are necessary for the query to present the launch sites beginning with ‘CCA’.

Total Payload Mass

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


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

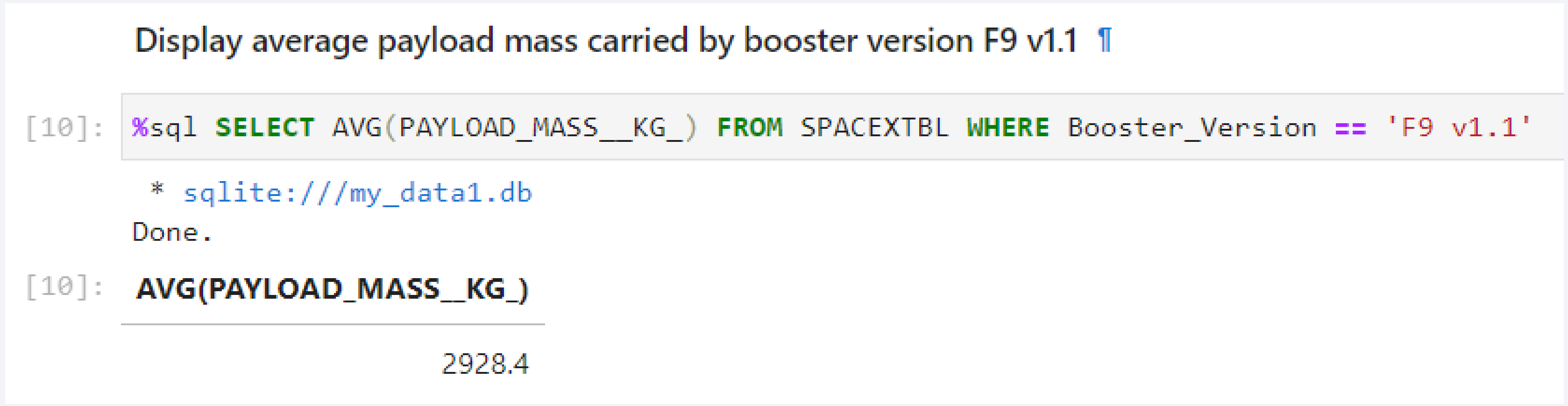
```
[9]: %sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Customer == 'NASA (CRS)'

 * sqlite:///my_data1.db
Done.

[9]: SUM(PAYLOAD_MASS__KG_)
                   45596.0
```

Explanation:

# Command WHERE Customer == ‘NASA (CRS)’ filters the boosters from NASA, then the SUM () command calculates the total pay load.

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



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

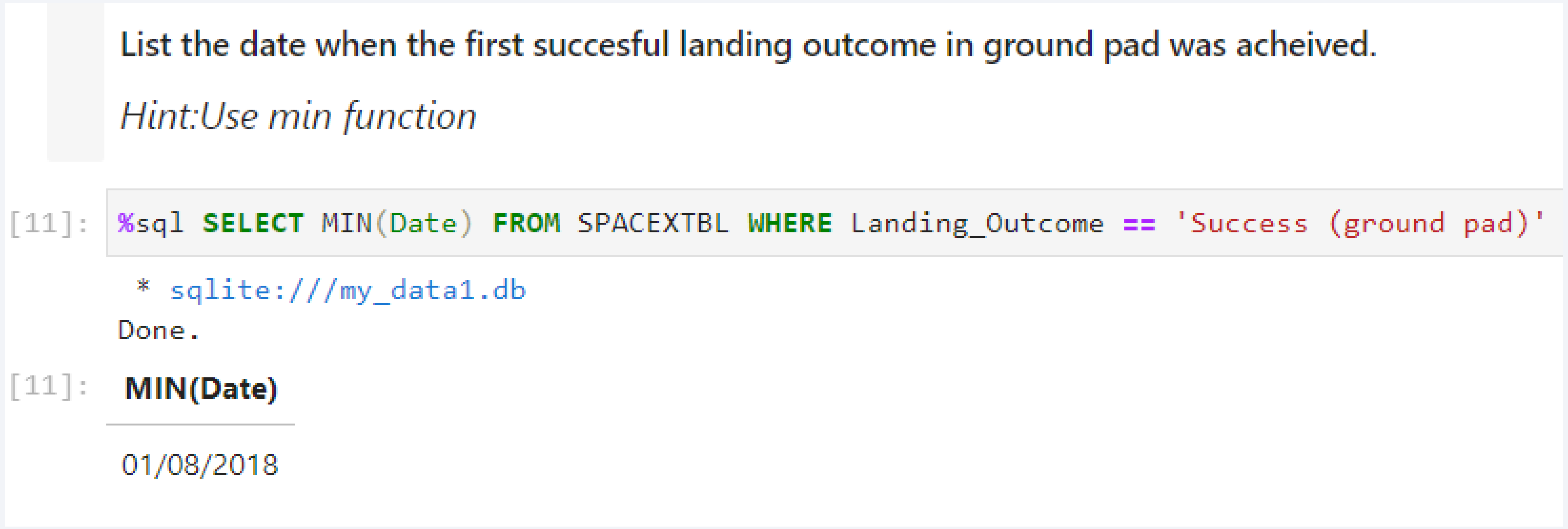
```
[10]: %sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version == 'F9 v1.1'

 * sqlite:///my_data1.db
Done.

[10]: AVG(PAYLOAD_MASS__KG_)
                      2928.4
```

Explanation:

# Command WHERE Booster_Version == 'F9 v1.1’ filters the launches with booster version F9 V1.1, then the command AVG () calculate the average pay load mass.

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


List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
[11]: %sql SELECT MIN(Date) FROM SPACEXTBL WHERE Landing_Outcome == 'Success (ground pad)'

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

[11]:

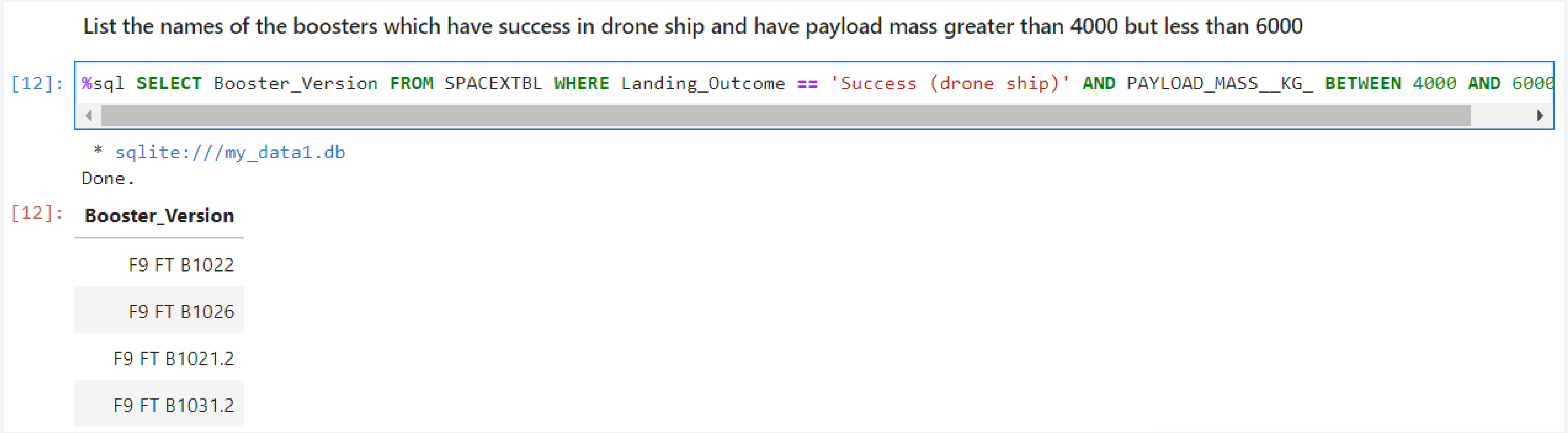
MIN(Date)
01/08/2018

Explanation:

# Command WHERE Landing_Outcome == 'Success (ground pad)' filters the launches with successful outcome on ground pad, then the command MIN () gives the first date.

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



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

```
[12]: %sql SELECT Booster_Version FROM SPACEXTBL WHERE Landing_Outcome == 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

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

[12]:

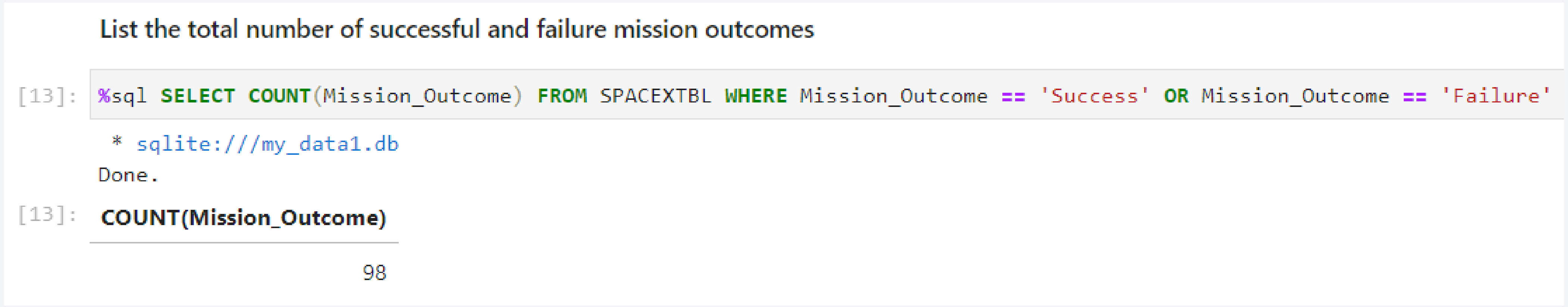
Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Explanation:

# Command WHERE Landing_Outcome == 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000 filter the launches based the requirements.

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

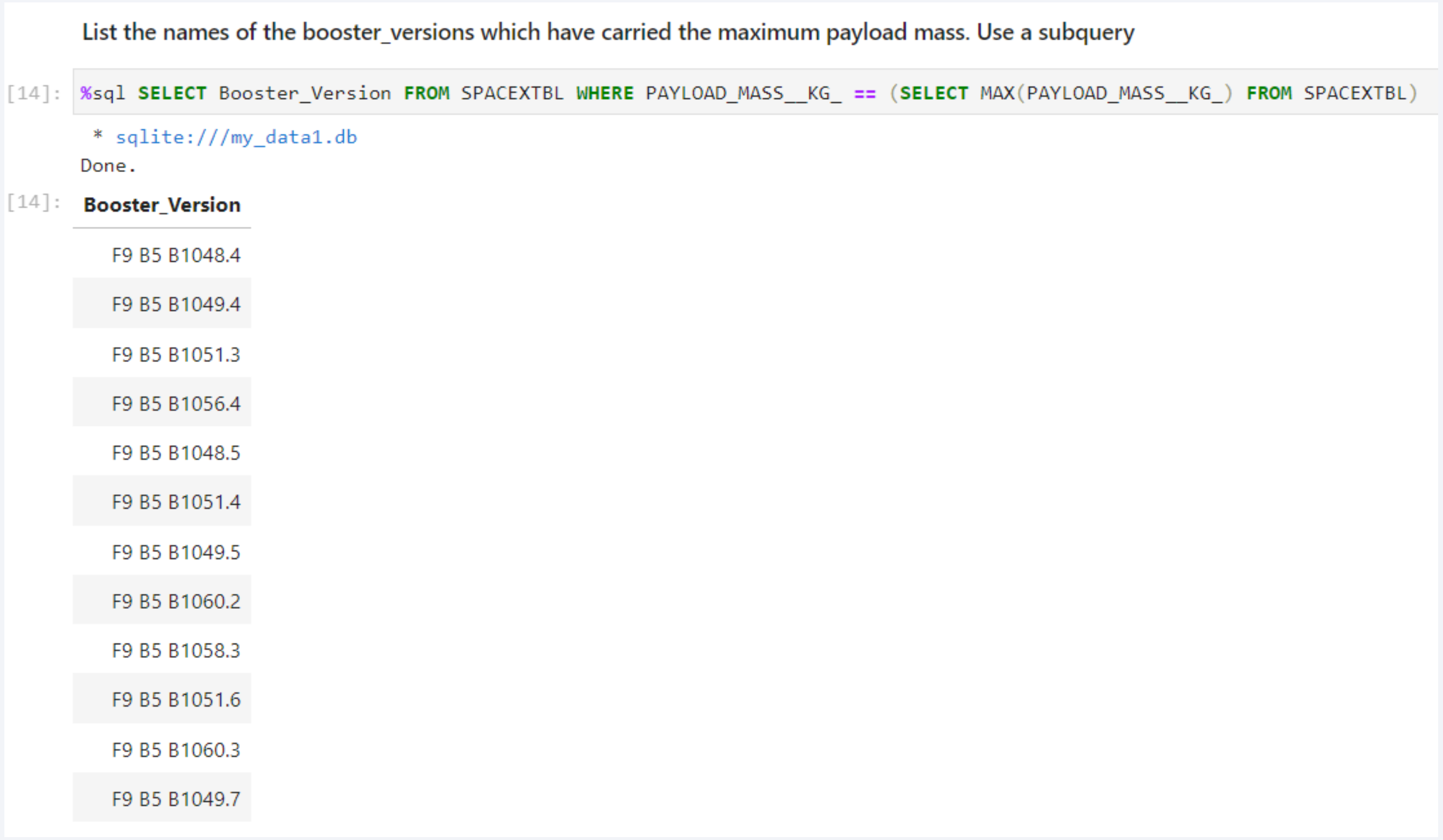
```
[13]: %sql SELECT COUNT(Mission_Outcome) FROM SPACEXTBL WHERE Mission_Outcome == 'Success' OR Mission_Outcome == 'Failure'

 * sqlite:///my_data1.db
Done.

[13]: COUNT(Mission_Outcome)
                          98
```

Explanation:

# Command WHERE Mission_Outcome == 'Success' OR Mission_Outcome == 'Failure' filters the launches with successful and failure mission outcomes. Then, the command COUNT() counts the total number.

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

```
[14]: %sql SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ == (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
 * sqlite:///my_data1.db
Done.
```

[14]:

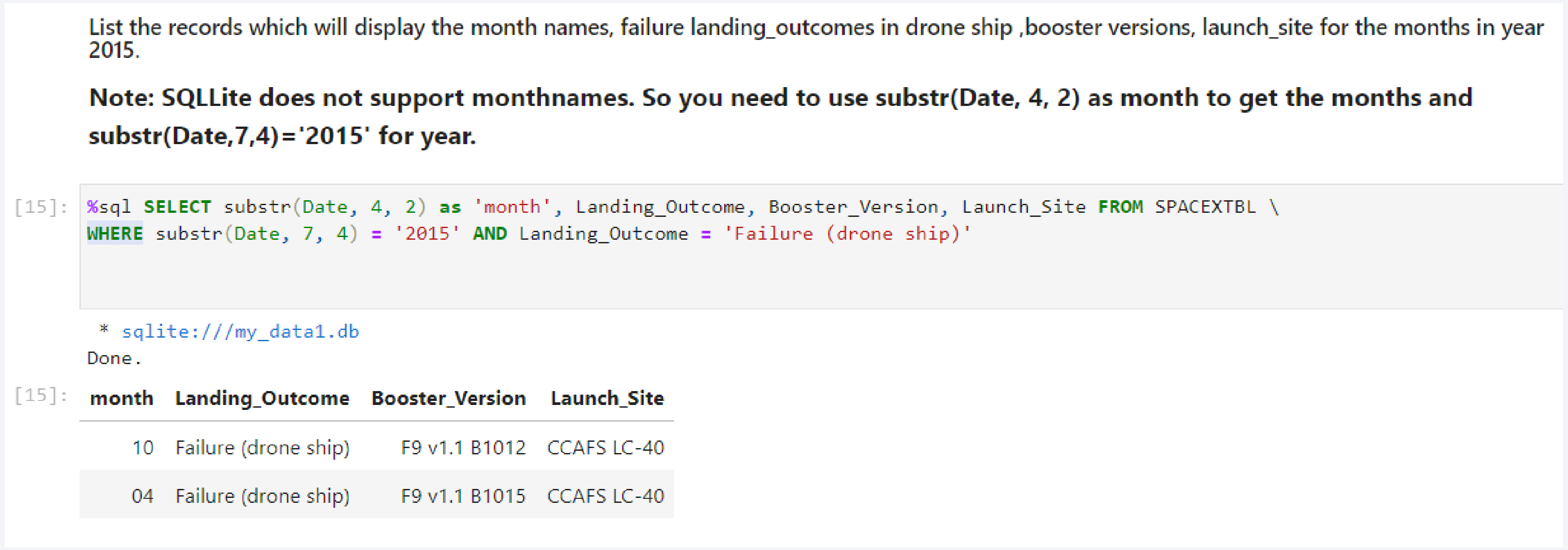
Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

Explanation:

# subquery “SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL” provides the maximum pay load mass, it is included in the WHERE command to filter the boosters carrying the maximum pay load mass.

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



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, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

```
[15]: %sql SELECT substr(Date, 4, 2) as 'month', Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTBL \
      WHERE substr(Date, 7, 4) = '2015' AND Landing_Outcome = 'Failure (drone ship)'
```

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

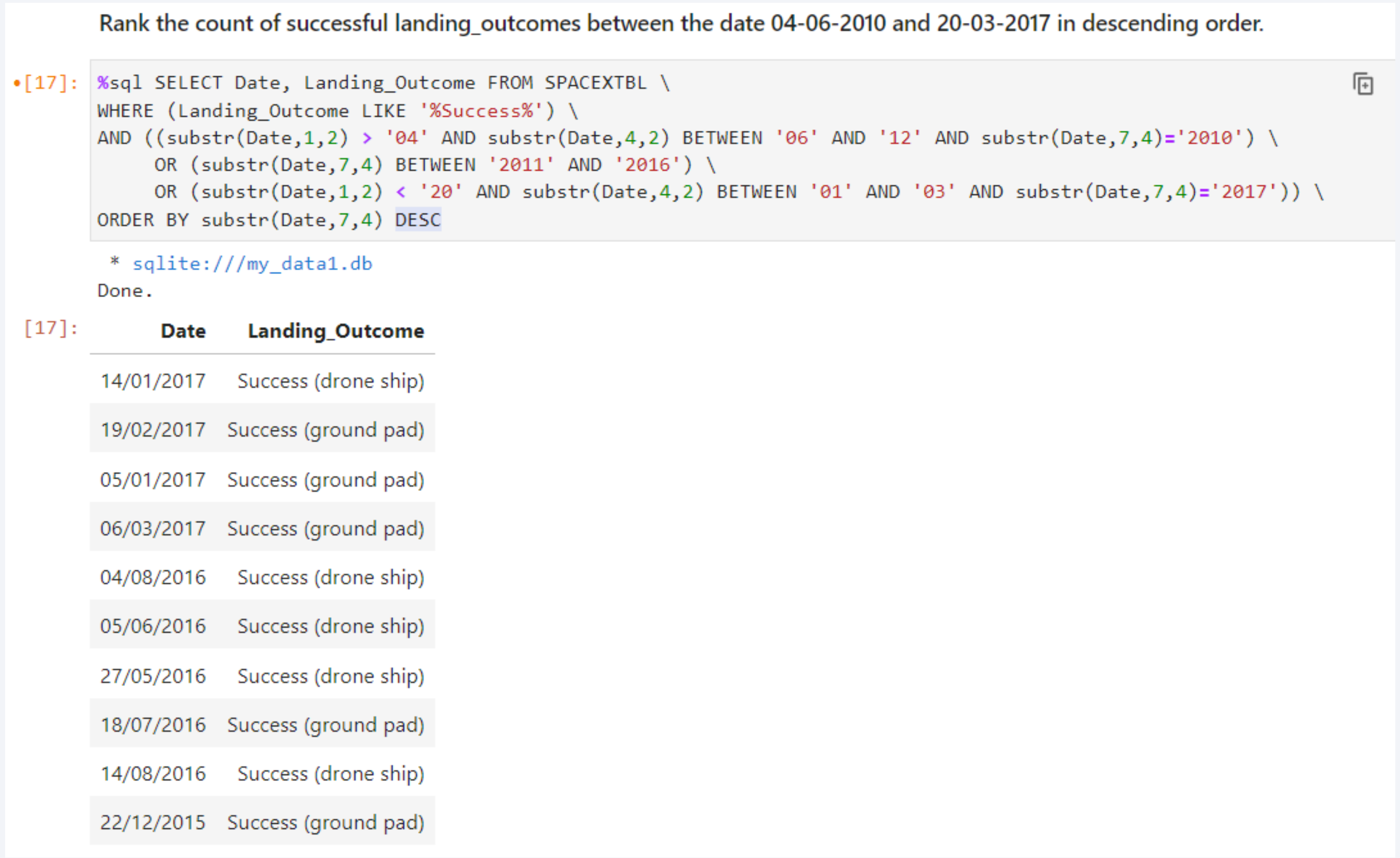
[15]:

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

Explanation: # the WHERE command filters the data based on the requirements. As the date format is like 20/06/2023, we need to use substr() to select the month and required year (2015).

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 successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
[17]: %sql SELECT Date, Landing_Outcome FROM SPACEXTBL \
WHERE (Landing_Outcome LIKE '%Success%') \
AND ((substr(Date,1,2) > '04' AND substr(Date,4,2) BETWEEN '06' AND '12' AND substr(Date,7,4)='2010') \
     OR (substr(Date,7,4) BETWEEN '2011' AND '2016') \
     OR (substr(Date,1,2) < '20' AND substr(Date,4,2) BETWEEN '01' AND '03' AND substr(Date,7,4)='2017')) \
ORDER BY substr(Date,7,4) DESC
```

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

[17]:

Date	Landing_Outcome
14/01/2017	Success (drone ship)
19/02/2017	Success (ground pad)
05/01/2017	Success (ground pad)
06/03/2017	Success (ground pad)
04/08/2016	Success (drone ship)
05/06/2016	Success (drone ship)
27/05/2016	Success (drone ship)
18/07/2016	Success (ground pad)
14/08/2016	Success (drone ship)
22/12/2015	Success (ground pad)

Explanation:

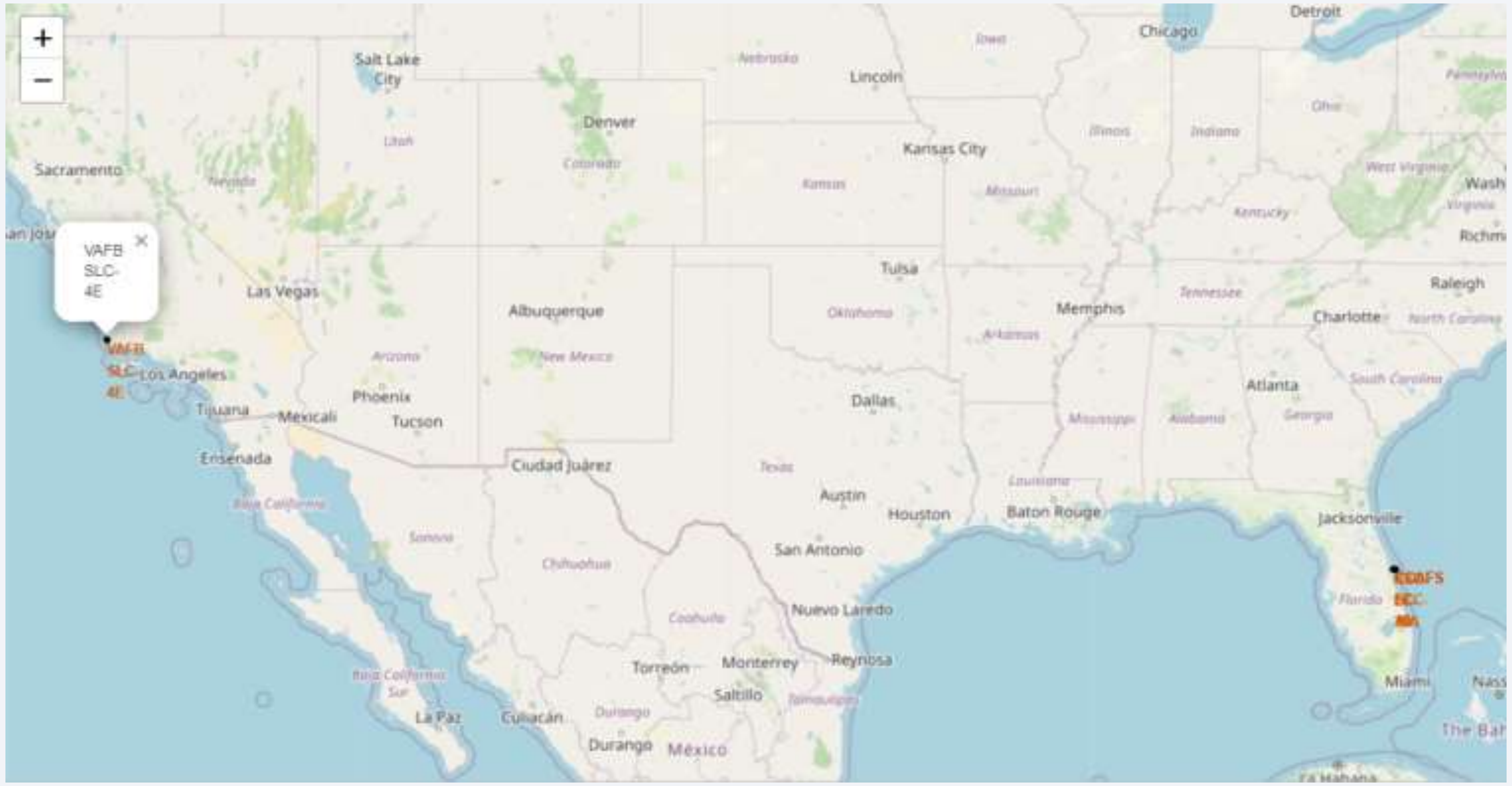
# the substr() is necessary to select the date between 2010-06-04 and 2017-03-20.

Section 3

Launch Sites Proximities Analysis

Mark all launch sites on a map

- 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



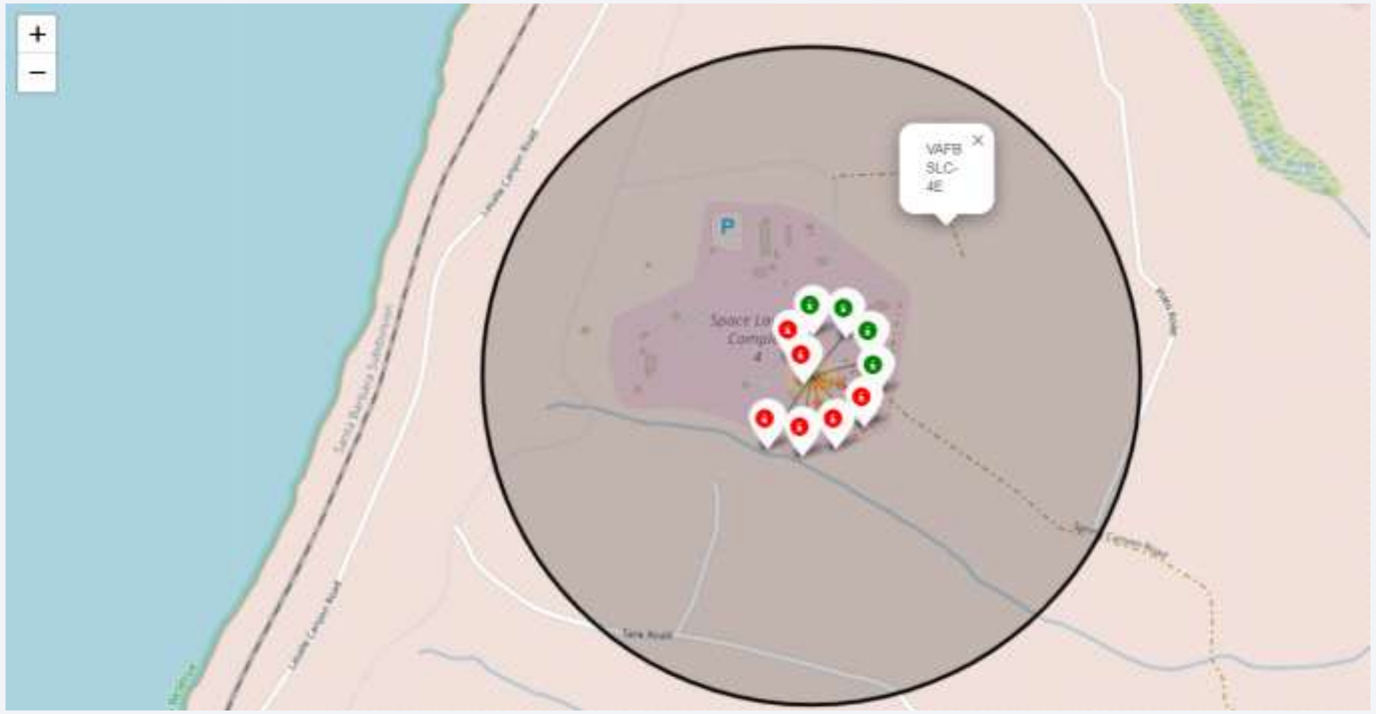
Explanation:

# create circle and marker variables using .Circle() and .Marker() functions. Add the circle and maker to the map using .add_child() function.

# all launch sites are in close proximity to the coast.

Mark the success/failed launches for each site on the map

- 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



# using the CheckClassValue() function to make a list of colors for successful launches (green) and failed launches (red). Insert this list to spacex_df data frame.

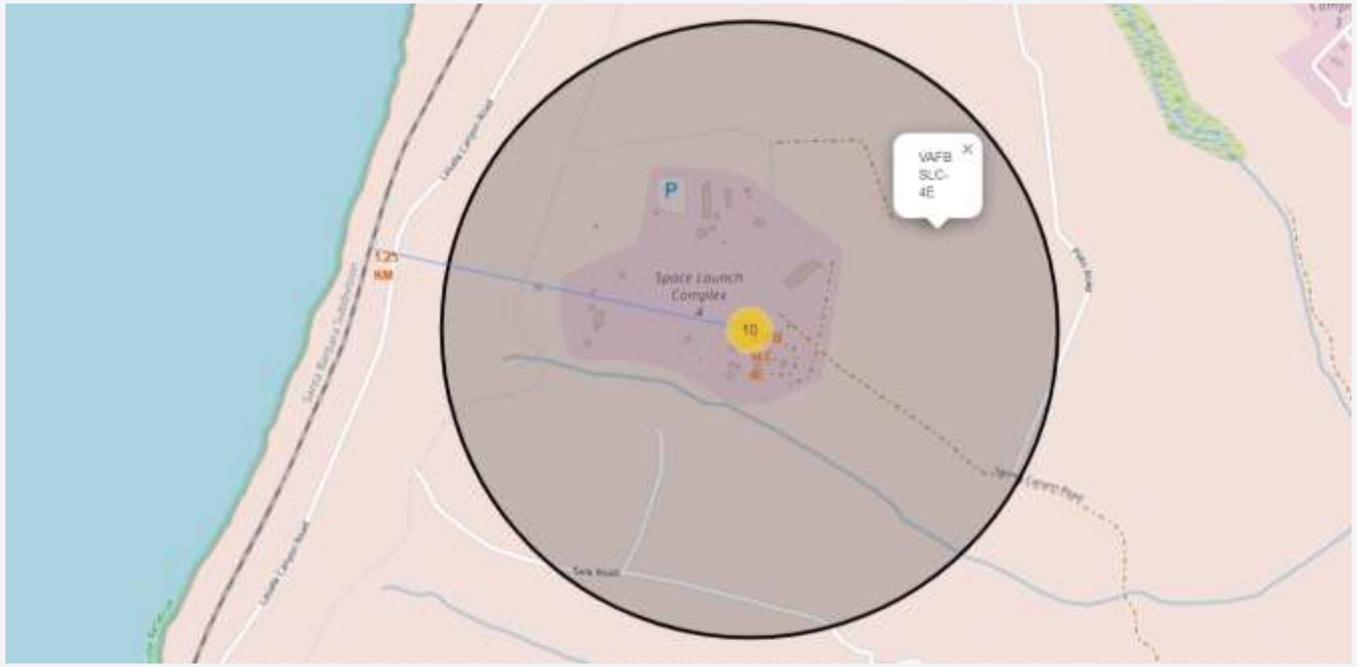
# iterate the rows of spacex_df based on the data in Lat, Long, and marker_color columns to generate a marker variable. Add the marker variable to marker_cluster variable as well as the map.

# each launch site (one location) may have many launches with different outcomes, the cluster can show the different outcomes conveniently.

Calculate the distances between a launch site to its proximities

- 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

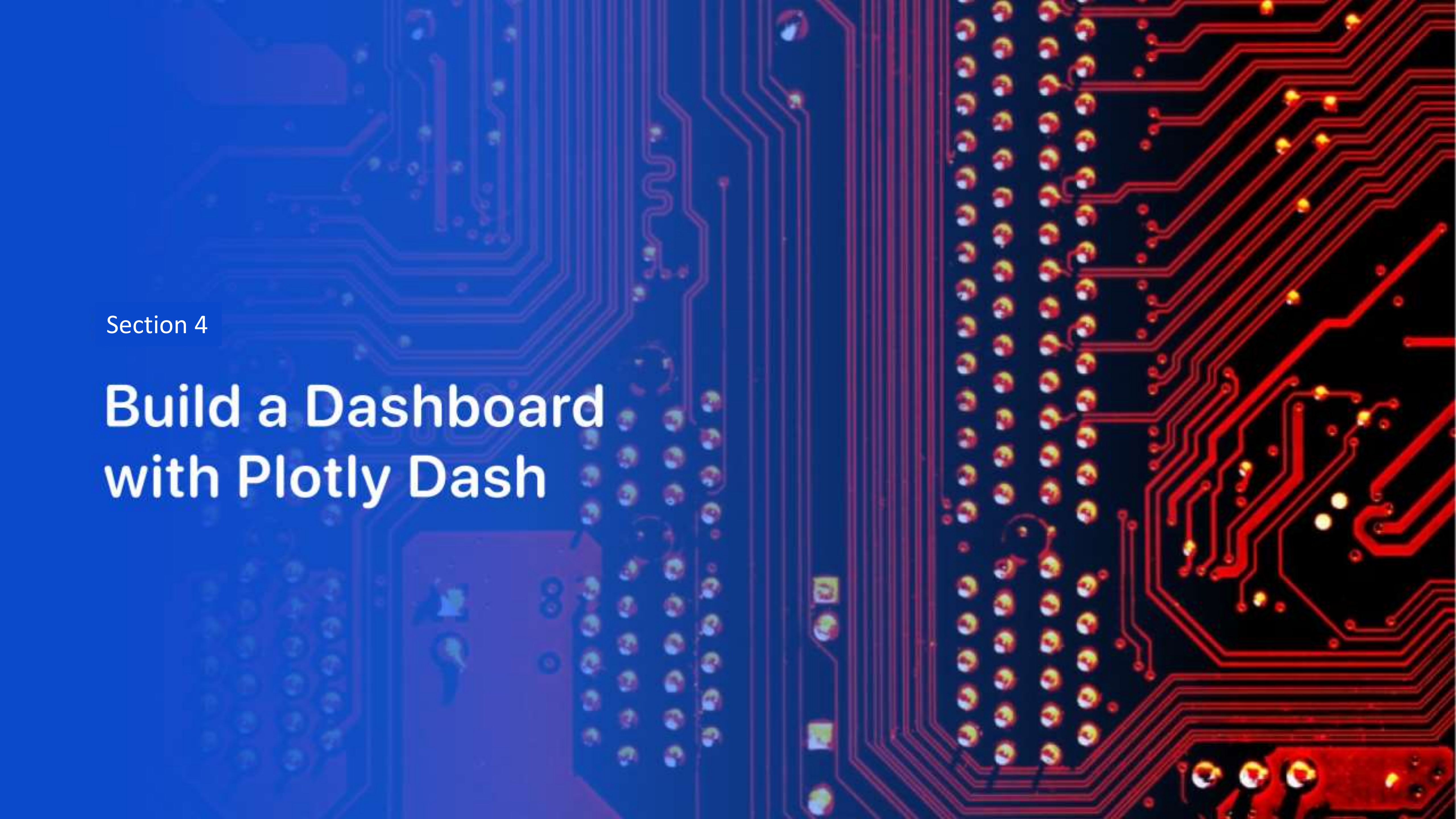
- Explain the important elements and findings on the screenshot



# For example, show the line and distance from Launch site VAFB SLC-4E (34.632834, -120.610745) to the marker point (34.63519, -120.62411) near Santa Barbara Subdivision railway.

# calculate the distance between the two points using calculate_distance(), use the distance to make distance marker, make a line marker, then add the distance marker and line marker to the map.

# It is necessary to have the MousePosition function to get the coordinates (Lat, Long). Distance and line markers can clearly show the distance between a launch site and a proximity (railway, highway, coastline, etc.)

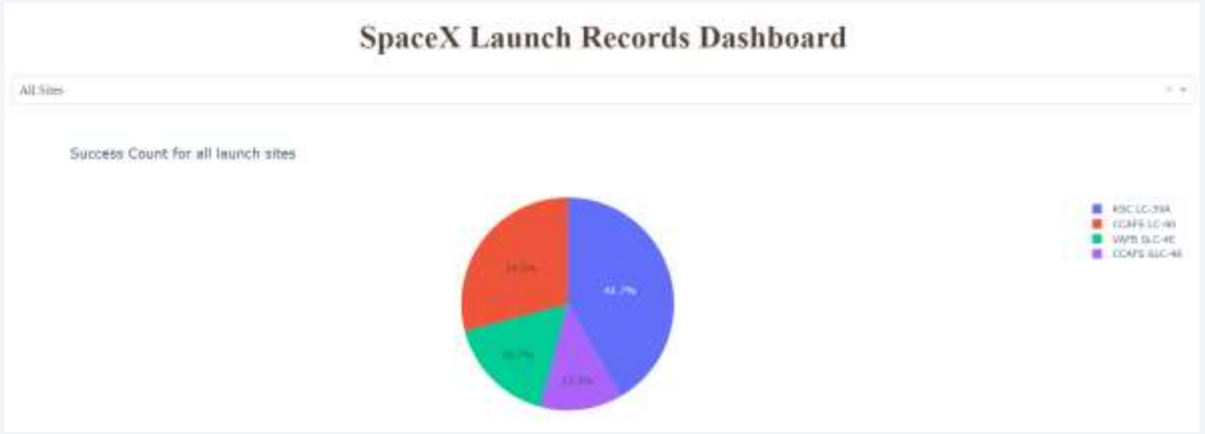


Section 4

Build a Dashboard with Plotly Dash

launch success count for all sites

- Show the screenshot of launch success count for all sites, in a pie chart
- Explain the important elements and findings on the screenshot



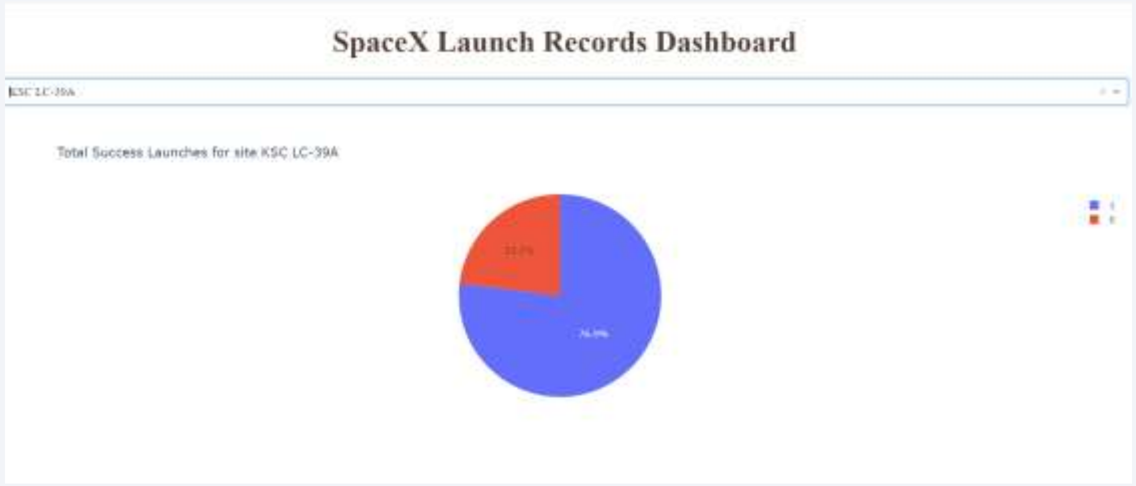
# The launch success count for four launch sites are showed in a pie chart. Different color represents different launch sites. The success rate is shown on the respective segment of pie chart.

# Launch site KSC LC-39A has the highest success rate (41.7%) among all launches.

# Launch site CCAFS SLC-40 has the lowest success rate (12.5%) among all launches.

launch site with highest launch success ratio

- Show the screenshot of the pie chart for the launch site with highest launch success ratio.
- Explain the important elements and findings on the screenshot

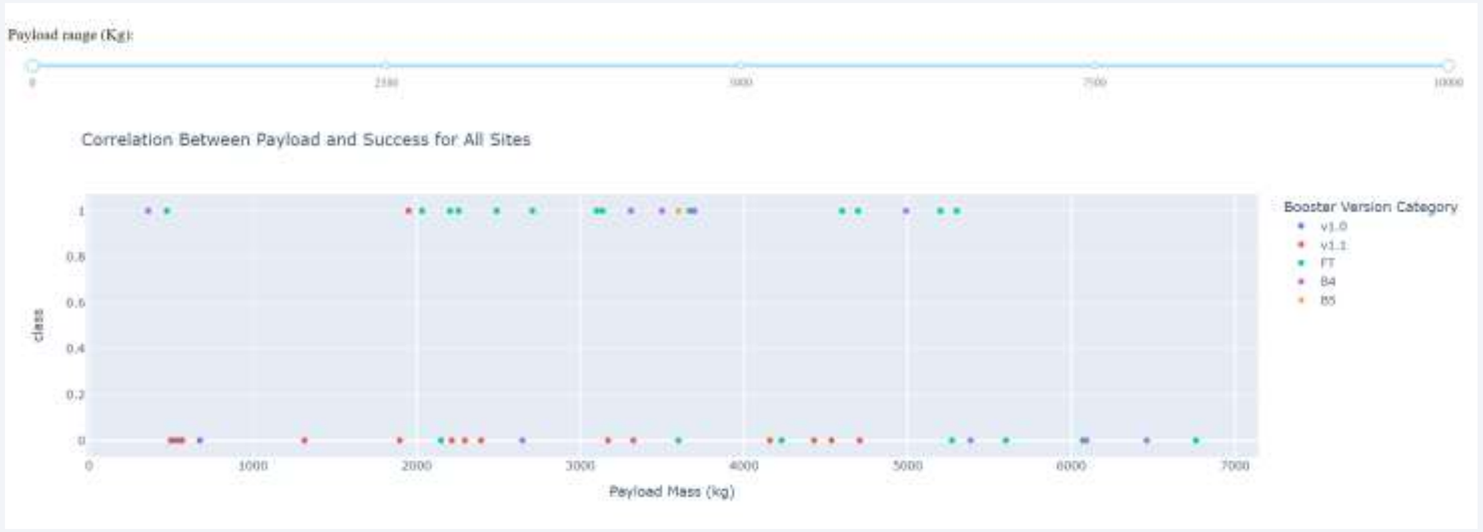

SpaceX Launch Records Dashboard
KSC LC-39A
Total Success Launches for site KSC LC-39A

# Launch site KSC LC-39A has the highest success rate as stated in the previous slide.

# The successful launch ratio is 79.6% whereas the failure ratio is 23.1% in Launch site KSC LC-39A

Payload vs. Launch Outcome scatter plot for all sites

- 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.



# the pay load range between 2000kgs and 4000kgs has the largest success rate as shown in the scatter plot.

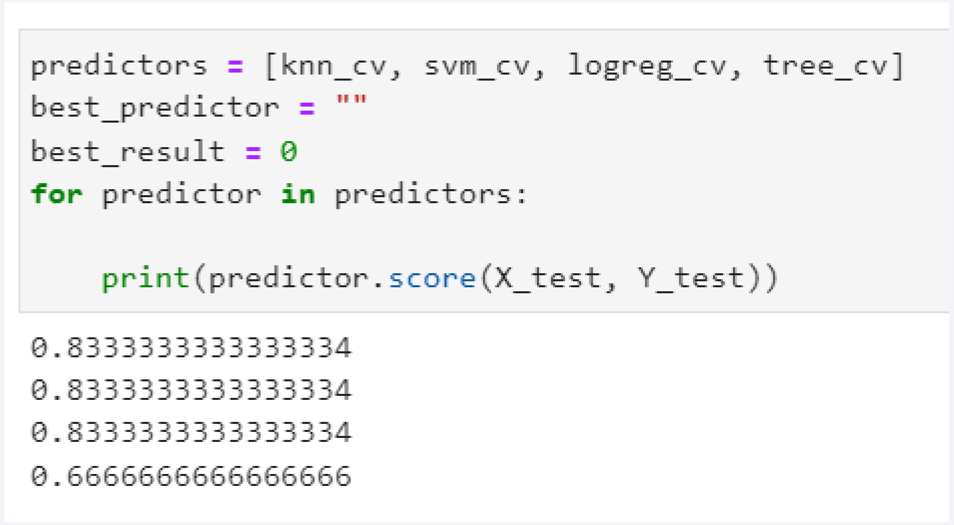
# The booster version FT has the largest success rate among all launches.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Visualize the built model accuracy for all built classification models.



```
predictors = [knn_cv, svm_cv, logreg_cv, tree_cv]
best_predictor = ""
best_result = 0
for predictor in predictors:

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

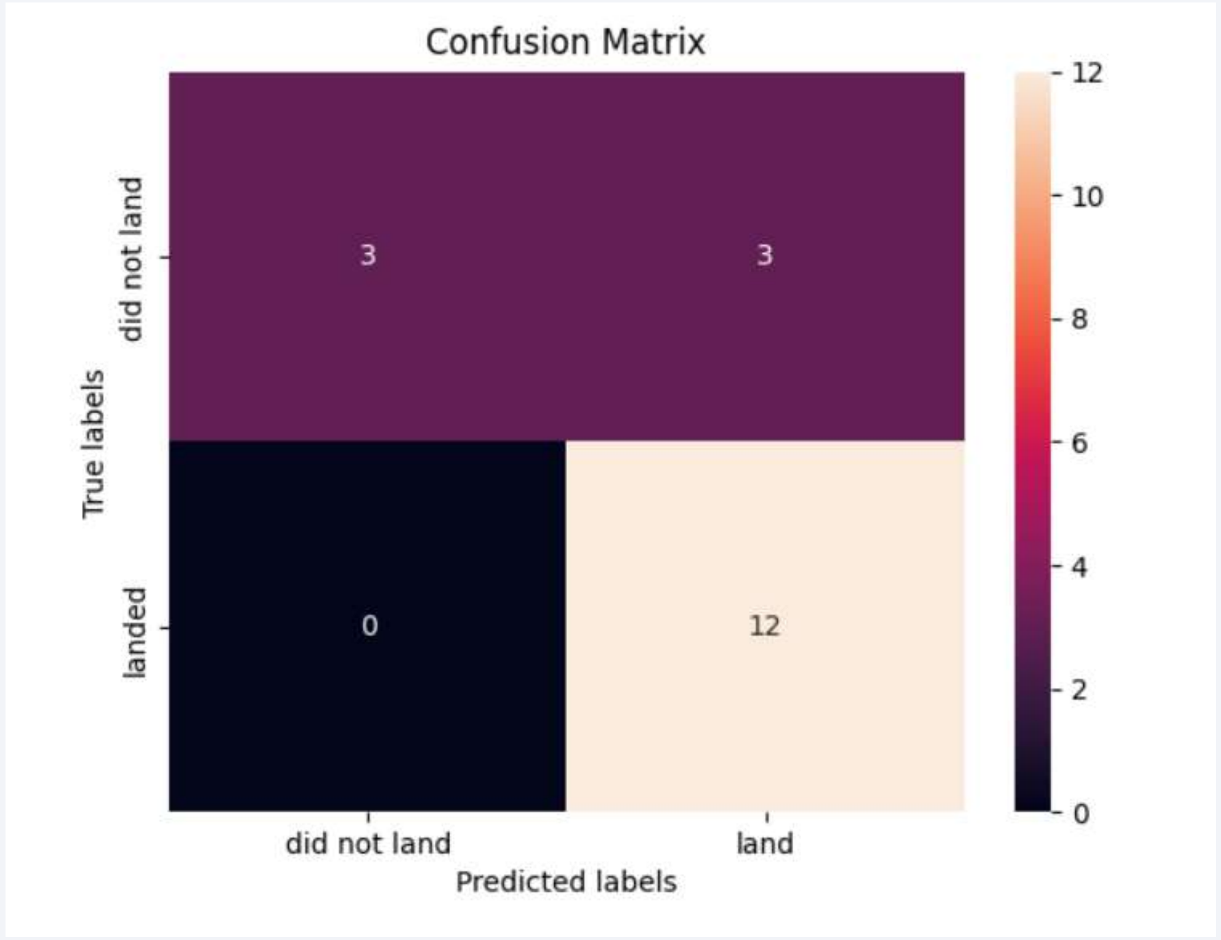
```
0.8333333333333334
0.8333333333333334
0.8333333333333334
0.6666666666666666
```

- Find which model has the highest classification accuracy

KNN, SVM, and Logistic Regression have similar high accuracy, the Decision Tree has lower accuracy.

Confusion Matrix

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



Conclusions

- The features such as launch sites, orbits, flight numbers, pay load mass can influence the successful rate for launches.
- Larger flight numbers have higher successful rate based on the experience from previous launches.
- Orbit ES-L1, GEO, HEO, and SSO have higher successful rate.
- Launch site KSC LC-39A has the highest success rate.
- All launch sites are close to coastline.
- The successful rate kept increasing since 2013 till 2020.
- The pay load range between 2000kgs and 4000kgs has the largest success rate.
- The booster version FT has the largest success rate.

Appendix

- SpaceX API calls notebook: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/jupyter-labs-spacex-data-collection-api.ipynb
- web scraping notebook: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/jupyter-labs-webscraping.ipynb
- data wrangling notebook: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb
- EDA with data visualization notebook: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb
- EDA with SQL notebook: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb
- Interactive map with Folium map: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/lab_jupyter_launch_site_location-jupyterlite.ipynb
- Predictive analysis lab: https://github.com/guanp2023/Applied-Data-Science-Capstone_Final-Presentation/blob/main/SpaceX_Machine_Learning_Prediction_Part_5_jupyterlite.ipynb

Thank you!