IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

David Vaughan
2024

Outline

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

Executive Summary

Summary of methodologies

- Data Collection API Lab
- Data Collection with Web Scraping lab
- Data Wrangeling
- Exploratory Data Analysis with SQL
- Visual analytics with Folium
- Interactive Dasboard with Ploty Dash.
- Machine Learning Predictive Lab

Summary of all results

- Flowcharts of Data collection and Data wrangling
- Screenshots of Data visualization
- Show results of Machine learning

Introduction

- SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars, other providers cost upwards 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.
- The job is to gather data about Space X From publicly available web data, clean and format the data, creates dashboards for visualize the data to gain insights, and create a predictive machine learning model using the data.
- The main question to be answered is if SpaceX will reuse the first stage.
- Try to use different values of the data such as payload size, mission orbit, mission date and other values to make a succesful prediction.

Section 1

Methodology

Methodology Executive Summary

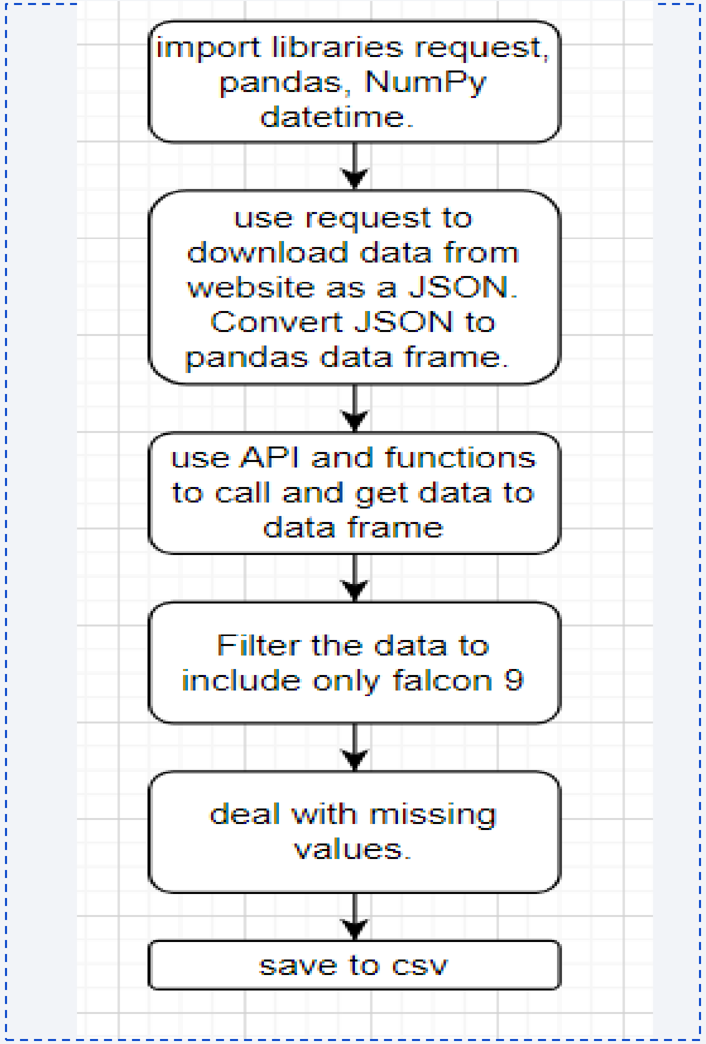
- Data collection methodology:
  - Data was collected by using web scraping form publicly available data and by using the Libraries Request, Beautiful Soup and Pandas.
- Perform data wrangling
  - Data was wrangled using Pandas data frames. Use pandas to manipulate data columns and make new columns based on data.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Use the library Sklearn. Build models using Sklearn, tune them using wrangled data, evaluate the model using built in score functions

Data Collection

- Data was collected using SpaceX API and Scrapping.
- SpaceX API returned data in the JSON format
- Scrapping Used returned Data in a html format.
- https://api.spacexdata.com/v4/launches/past
- https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches

Data Collection – SpaceX API

- Download data into data frame. Data Frame has 17 different columns
- Deal with missing values. For missing values in this Data Frame replace with the mean
- Save data in csv format to be used again
- Data Collection Github Link



import libraries request, pandas, NumPy datetime.

use request to download data from website as a JSON. Convert JSON to pandas data frame.

use API and functions to call and get data to data frame

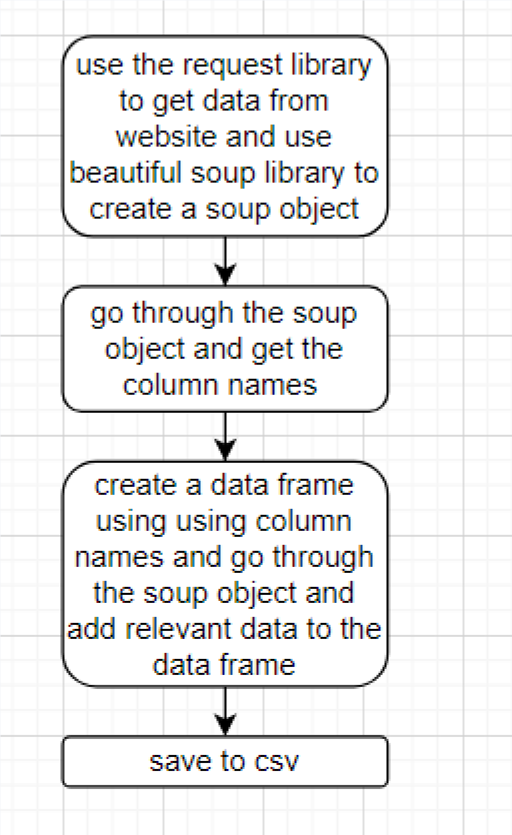
Filter the data to include only falcon 9

deal with missing values.

save to csv

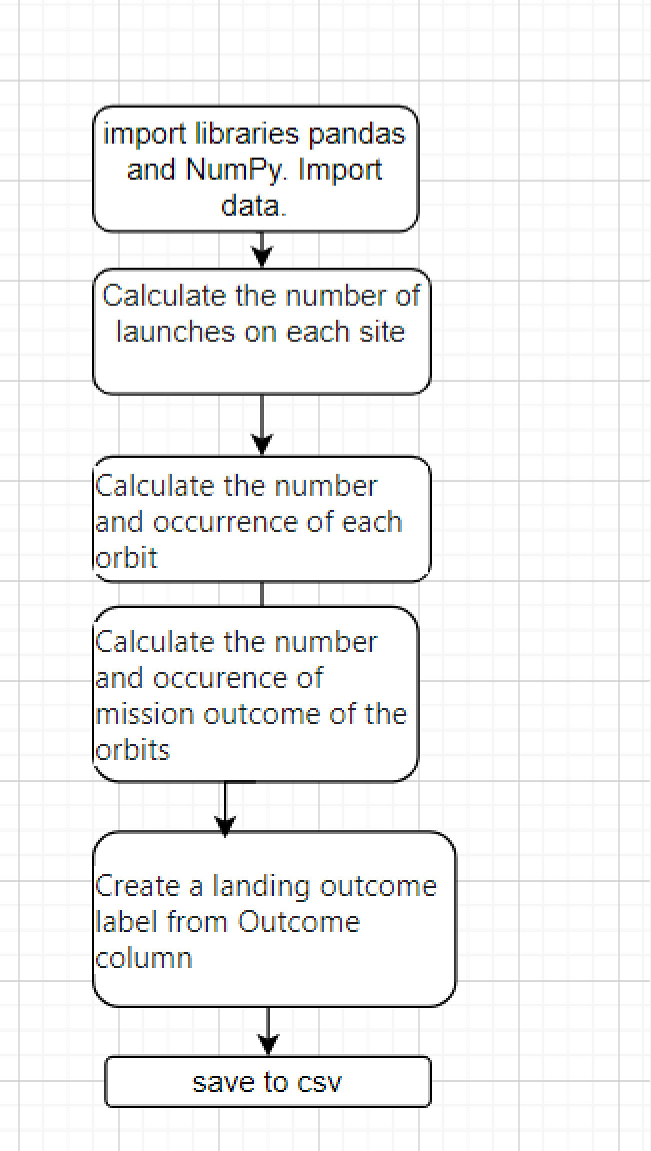
Data Collection - Scraping

- Download key data from a Wikipedia page.
- Handle errors in the data any missing values such as customer name
- Save the data in a csv format for future use
- Data Scraping Github Link


use the request library to get data from website and use beautiful soup library to create a soup object
go through the soup object and get the column names
create a data frame using using column names and go through the soup object and add relevant data to the data frame
save to csv

Data Wrangling

- Use Pandas Data Frames to handle the data.
- Create a new landing outcome column called class that was used as a numerical representation of if a landing was successful or not based on the column outcome.
- Save to csv
- Data wrangling Github link



import libraries pandas and NumPy. Import data.

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

save to csv

EDA with Data Visualization

- Scatter Plots chosen: Flight Number and Launch Site, Payload and Launch Site, FlightNumber and Orbit type, Payload and Orbit type
- Scatter plots allow to show the data in a visual way and show the general correlation between two variables
- Line Chart chosen: launch success yearly trend
- Line chars show how the success changes over time
- Bart Charts chosen: success rate of each orbit type
- Bar charts allow comparisons of the success rate between each orbit type
- EDA Github Link

EDA with SQL

- 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 achieved
- 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.
- SQL Github Link

Build an Interactive Map with Folium

- Added map objects circle to launch points. This allows the user to see where the launch sites are located.
- Add Markers to each individual launch site where green is a successful launch and red is a failed launch. It lets the user see how the record of launch sites success.
- Added lines with distances to a highway, railway, and city. It lets the user see how far away certain locations are from site.
- Folium Github Link

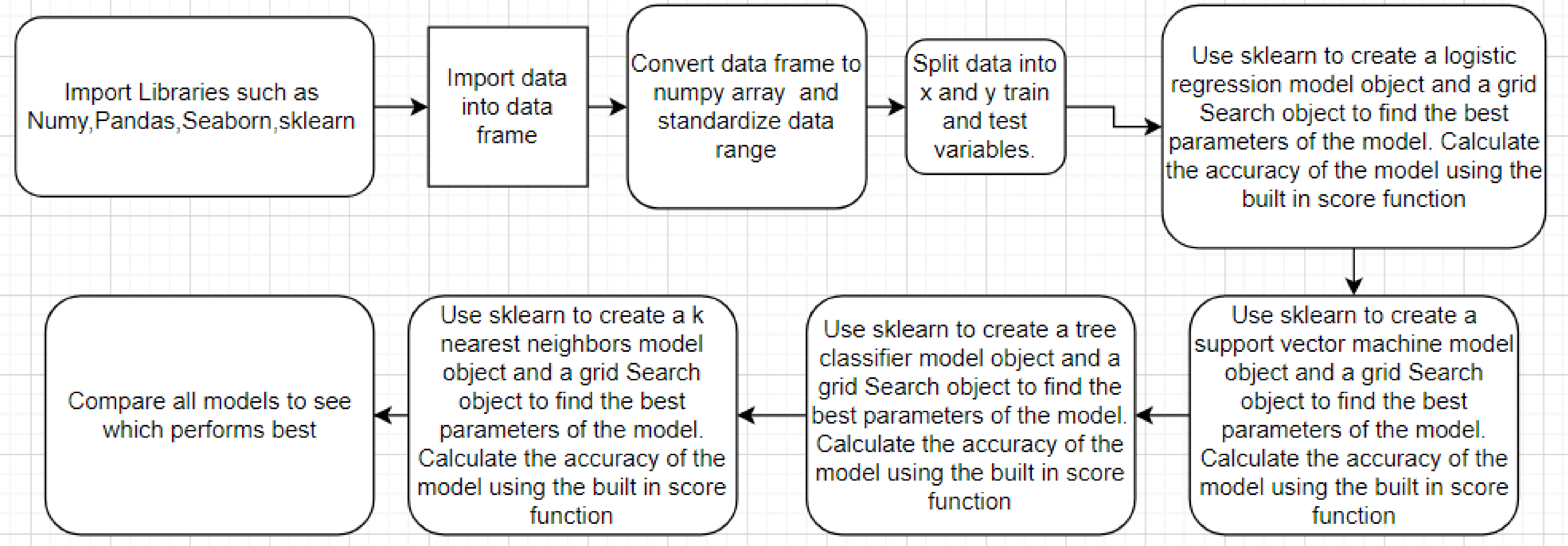
Build a Dashboard with Plotly Dash

- A interaction that was added is one where the user can choose All launch sites or a specific launch site.
- Another interaction added is where the user can the payload weight for a scatter plot.
- The plots added were a pie chart where each slice of the pie is the success at a launch site .This allows you to see where most successful launches come from.
- A pie chart of success vs failure at a specific launch site. This allows you to see if a specific launch site has successful launches.
- A scatter plot of Payload launch and outcome that shows the launches for either all launch sites or the specific chosen launch site. This plot allows the user to see how payload weight effects the success rate of launches.
- Ploty Github link

Predictive Analysis (Classification)

- Models tried were logistic regression, support vector machine, decision tree classification, and k nearest neighbour.
- Built classification models using sklearn.Improved the models by using grid search to find the best parameters for the models Evaluated using built in score function, which finds the amount of correct labels and divides by all labels. Found the best model by comparing scores
- Machine Learning Github Link

Predictive Analysis Flowchart



Import Libraries such as Numy,Pandas,Seaborn,sklearn

Import data into data frame

Convert data frame to numpy array and standardize data range

Split data into x and y train and test variables.

Use sklearn to create a logistic regression model object and a grid Search object to find the best parameters of the model. Calculate the accuracy of the model using the built in score function

Use sklearn to create a support vector machine model object and a grid Search object to find the best parameters of the model. Calculate the accuracy of the model using the built in score function

Use sklearn to create a tree classifier model object and a grid Search object to find the best parameters of the model. Calculate the accuracy of the model using the built in score function

Use sklearn to create a k nearest neighbors model object and a grid Search object to find the best parameters of the model. Calculate the accuracy of the model using the built in score function

Compare all models to see which performs best

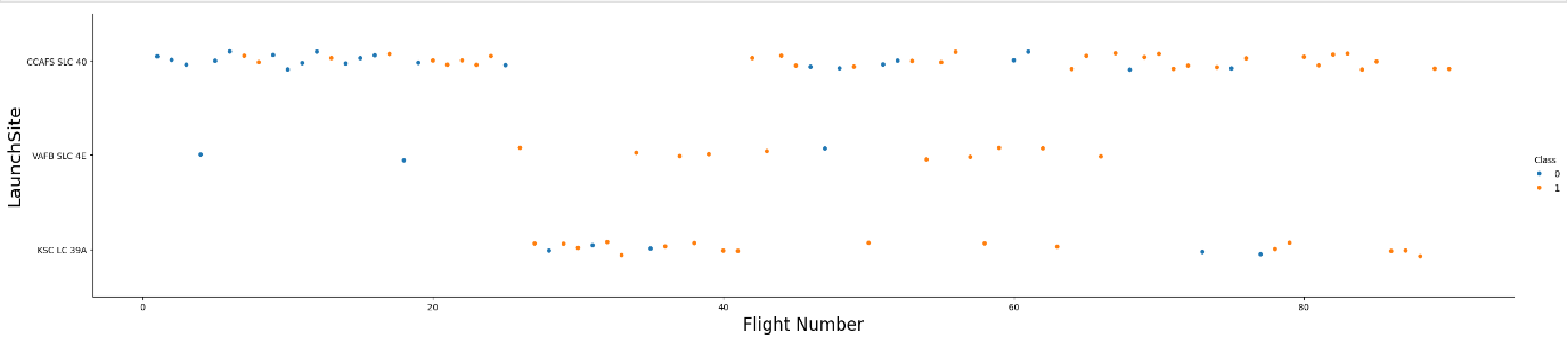
Results to be Shown

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

Section 2

Insights drawn from EDA

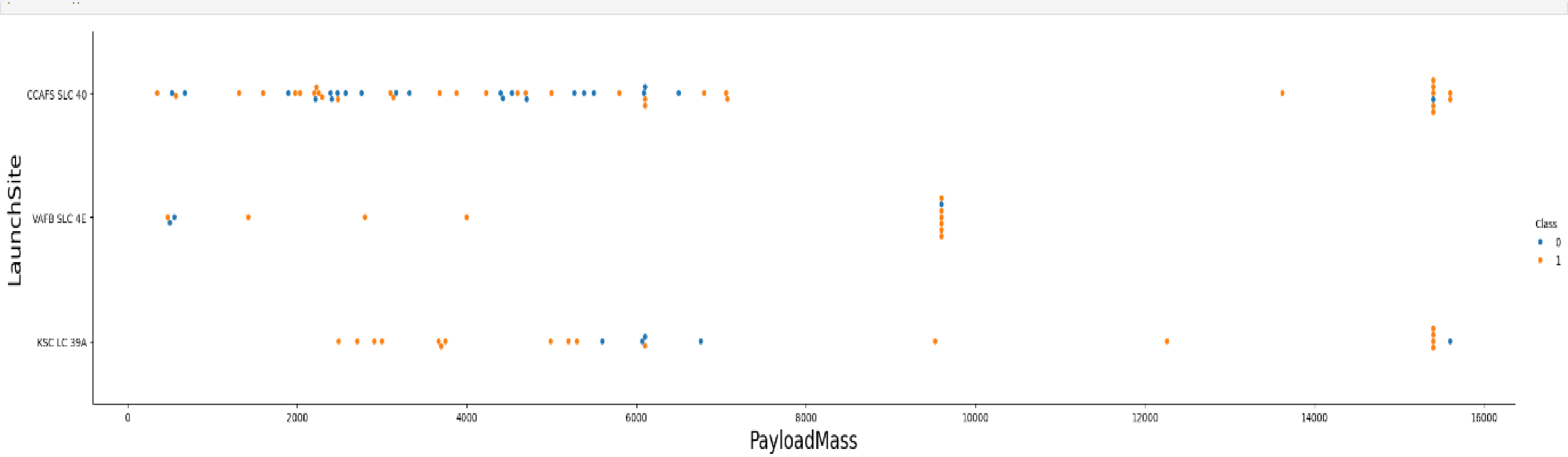
Flight Number vs. Launch Site



- Latter Flights have a high chance of success.
- CCAFS SLC 40 has most launches

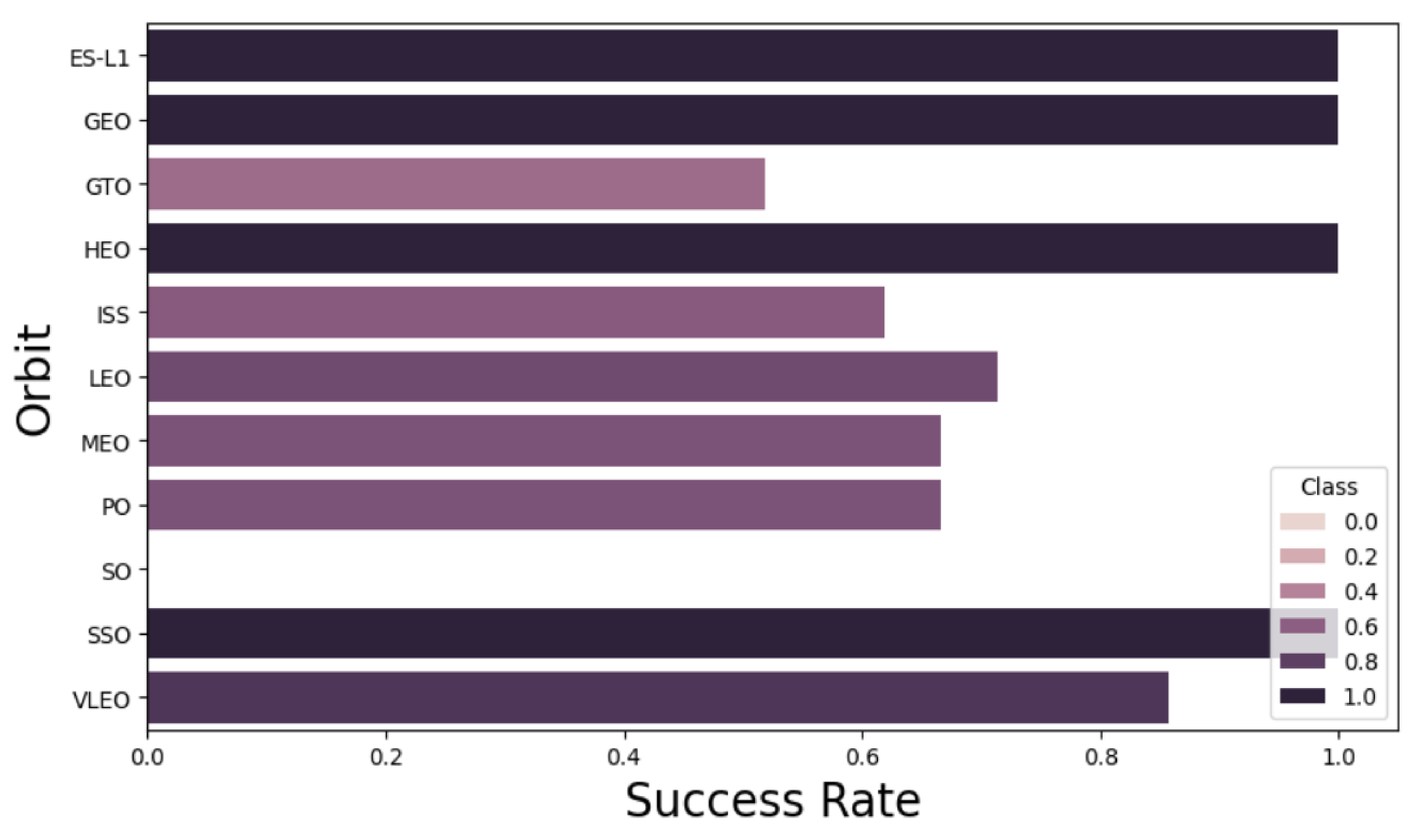
Payload vs. Launch Site

- The 9000 Kg and 15000 Kg range is a popular cluster
- 4000-6000 Kg has lower probability of Success



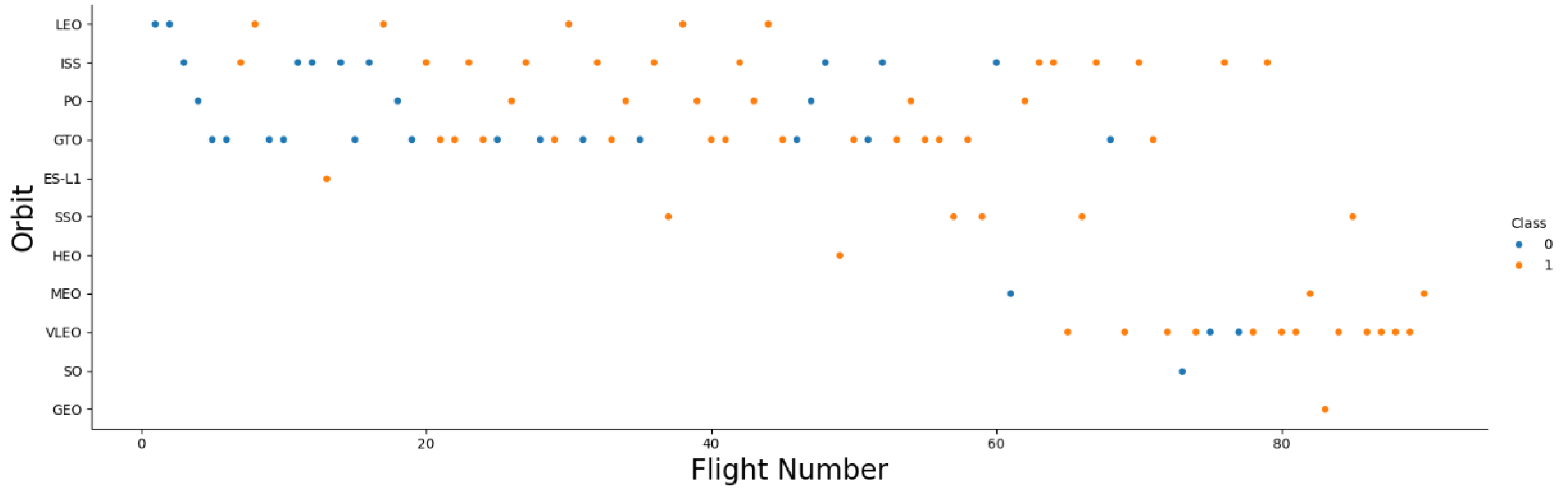
Success Rate vs. Orbit Type

- ES-L1 ,GEO,HEO,SSO have 100% success rate.
- GTO has low success rate
- SO has no successes



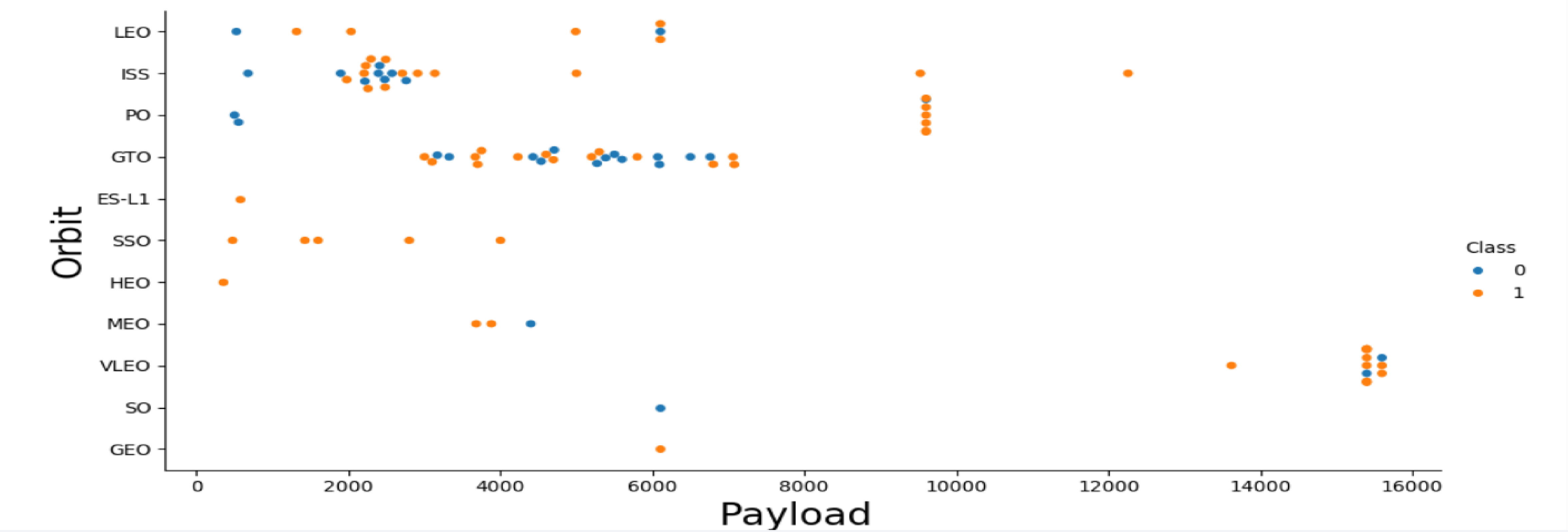
Flight Number vs. Orbit Type

- GEO,SO,HEO each only have 1 launch attempt in them.
- VLEO is a higher orbit that is popular
- Lower Orbits (LEO,ISS,PO,GTO) are more popular



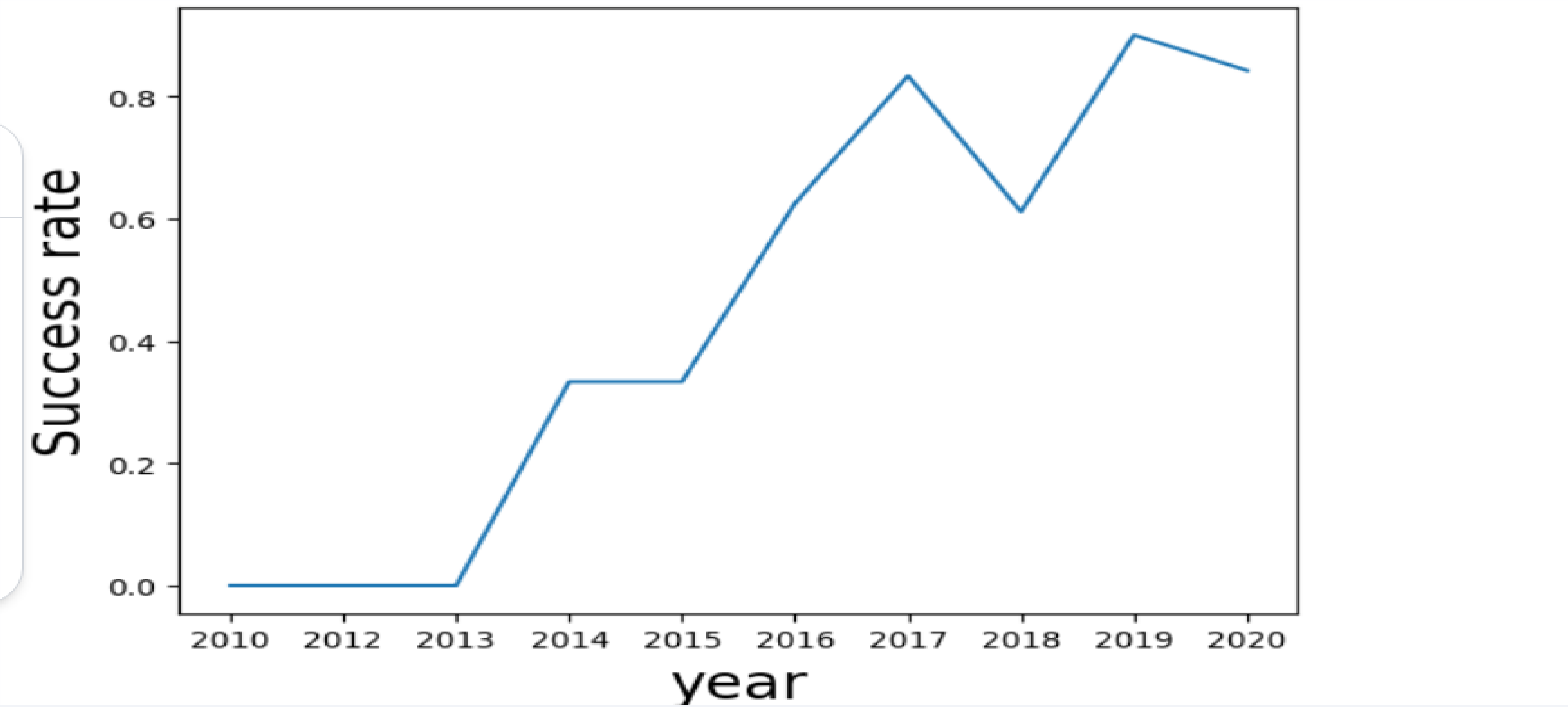
Payload vs. Orbit Type

- VLEO has a high payload and good success rate
- Higher Payloads tend to be more successful.
- Lower orbits tend to have smaller Payloads



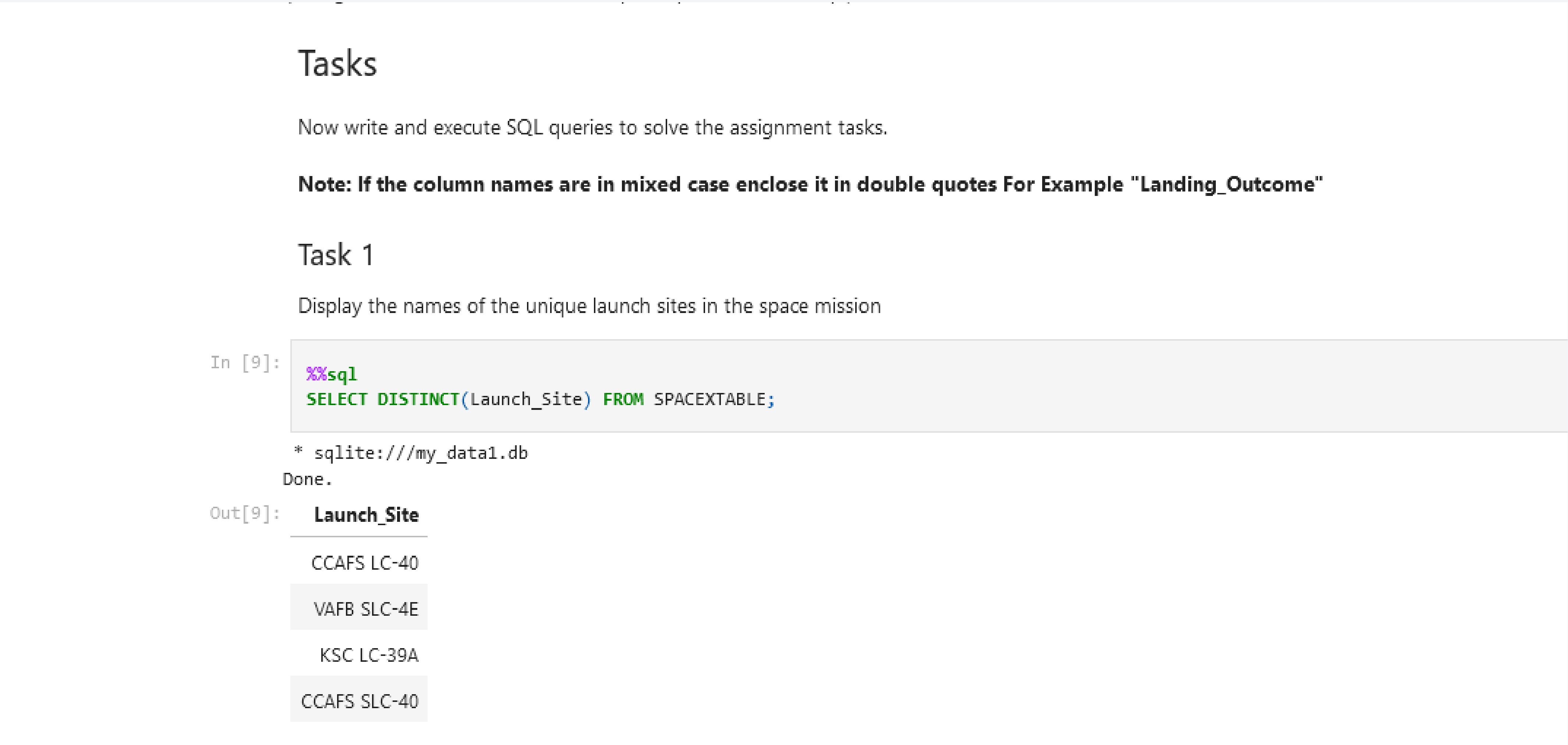
Launch Success Yearly Trend

- The success rate is going up



All Launch Site Names

- To find unique launch sites names select distinct names from the Launch_Site column .



Tasks

Now write and execute SQL queries to solve the assignment tasks.

Note: If the column names are in mixed case enclose it in double quotes For Example "Landing_Outcome"

Task 1

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

```
In [9]:  %%sql
         SELECT DISTINCT(Launch_Site) FROM SPACEXTABLE;
```

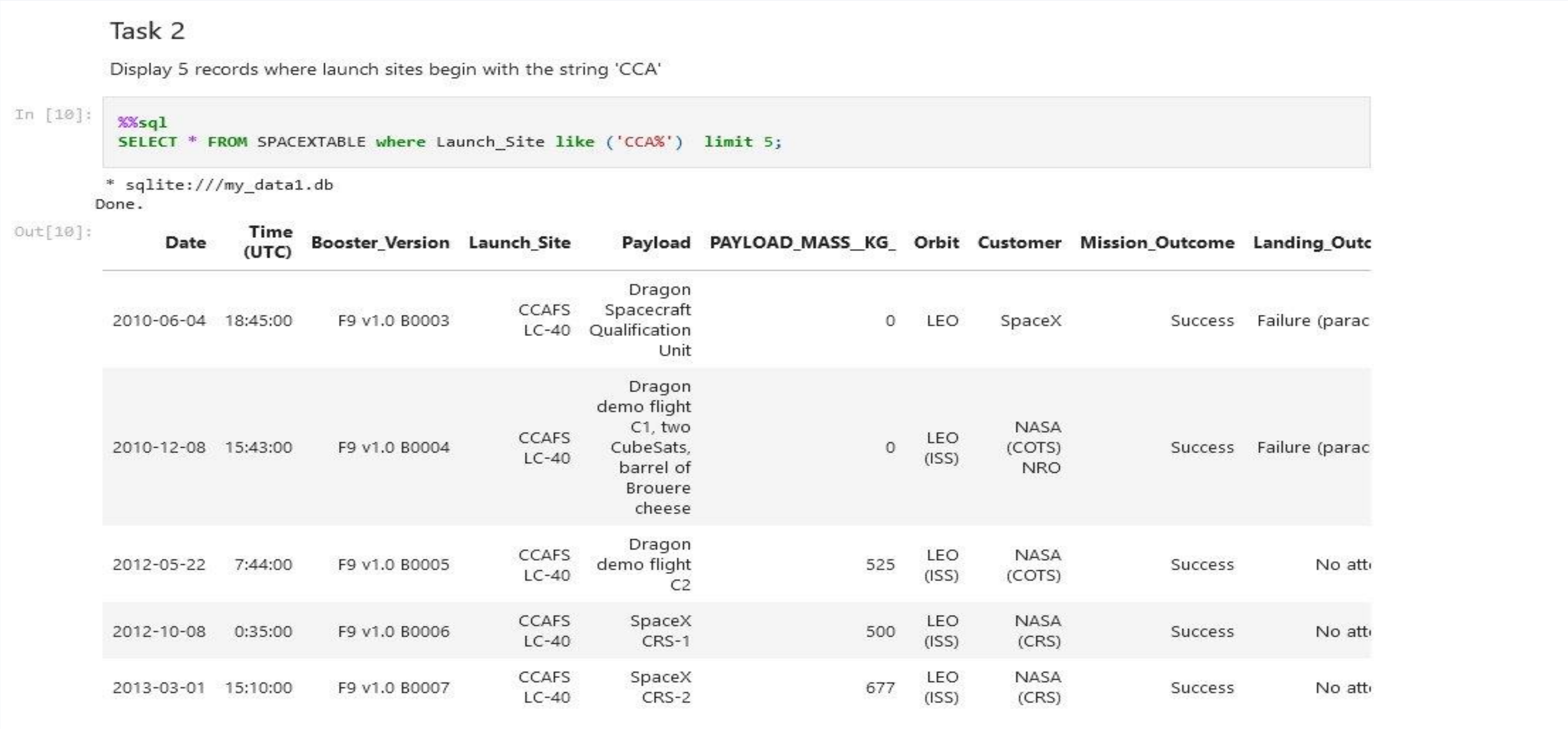
```
 * sqlite:///my_data1.db
Done.
```

Out[9]:

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` by using like function



Task 2

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

In [10]:

```
%%sql
SELECT * FROM SPACEXTABLE where Launch_Site like ('CCA%')  limit 5;
```

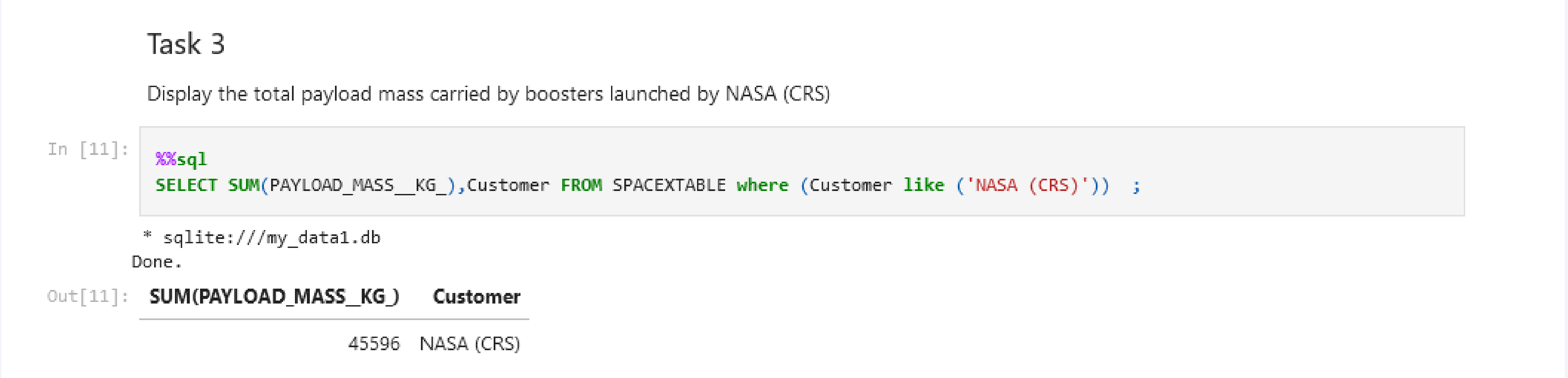
```
 * sqlite:///my_data1.db
Done.
```

Out[10]:

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

Total Payload Mass

- Calculate the total payload carried by boosters from NASA By using SUM and LIKE.



Task 3

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

In [11]:

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_),Customer FROM SPACEXTABLE where (Customer like ('NASA (CRS)'))  ;
```

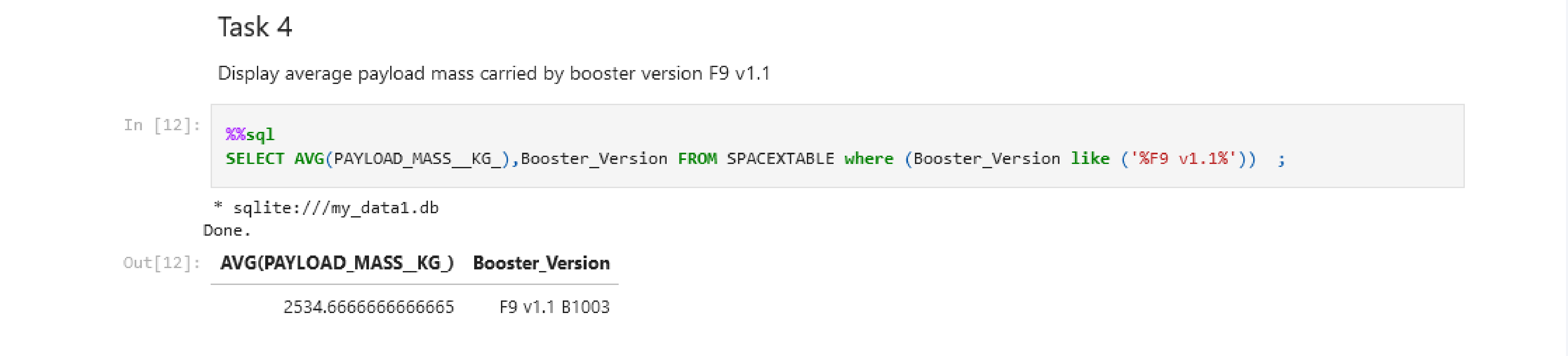
```
 * sqlite:///my_data1.db
Done.
```

Out[11]:

SUM(PAYLOAD_MASS__KG_)	Customer
45596	NASA (CRS)

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1 by using AVG and LIKE.



Task 4

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

In [12]:

```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_),Booster_Version FROM SPACEXTABLE where (Booster_Version like ('%F9 v1.1%'))  ;
```

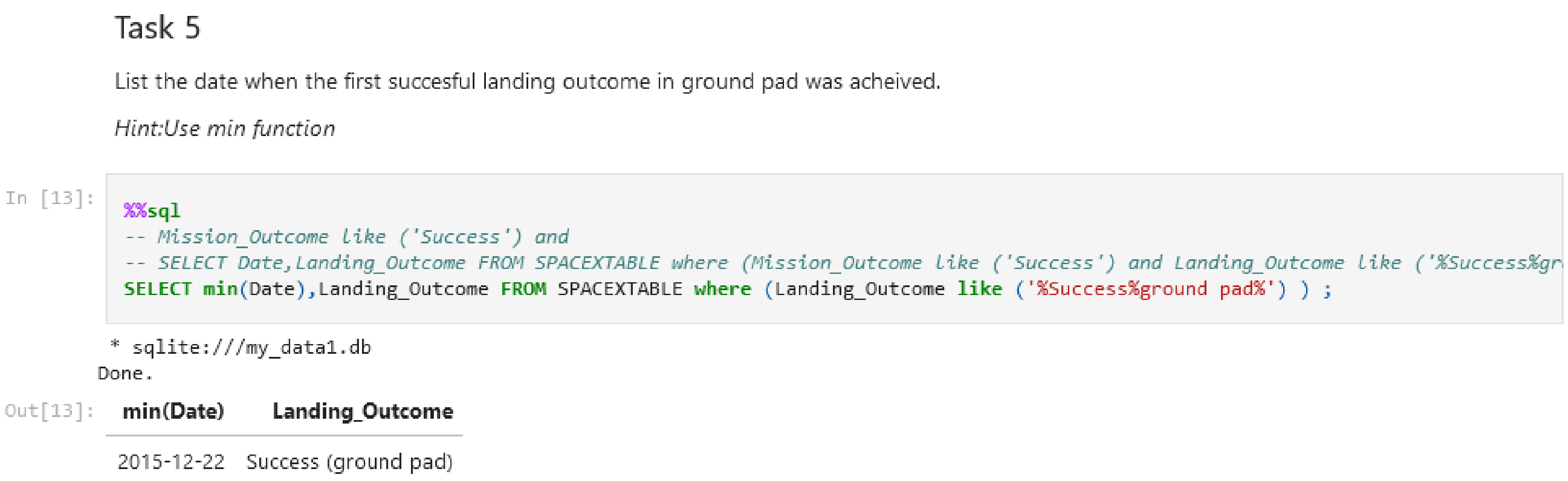
```
 * sqlite:///my_data1.db
Done.
```

Out[12]:

AVG(PAYLOAD_MASS__KG_)	Booster_Version
2534.6666666666665	F9 v1.1 B1003

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad by using min and LIKE



Task 5

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

Hint:Use min function

In [13]:

```
%%sql
-- Mission_Outcome like ('Success') and
-- SELECT Date,Landing_Outcome FROM SPACEXTABLE where (Mission_Outcome like ('Success') and Landing_Outcome like ('%Success%gr
SELECT min(Date),Landing_Outcome FROM SPACEXTABLE where (Landing_Outcome like ('%Success%ground pad%') ) ;
```

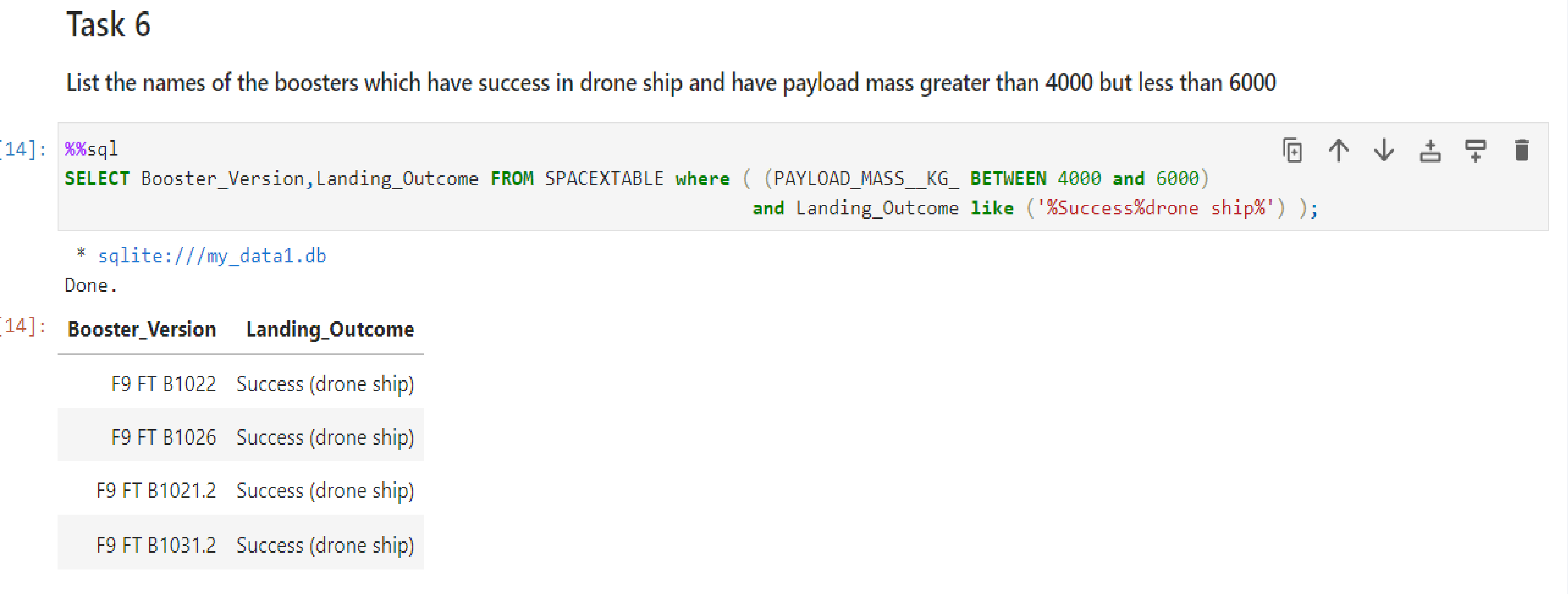
```
 * sqlite:///my_data1.db
Done.
```

Out[13]:

min(Date)	Landing_Outcome
2015-12-22	Success (ground pad)

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 by using BETWEEN and LIKE.



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

[14]:

```
%%sql
SELECT Booster_Version,Landing_Outcome FROM SPACEXTABLE where ( (PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000)
                                                  and Landing_Outcome like ('%Success%drone ship%') );
```

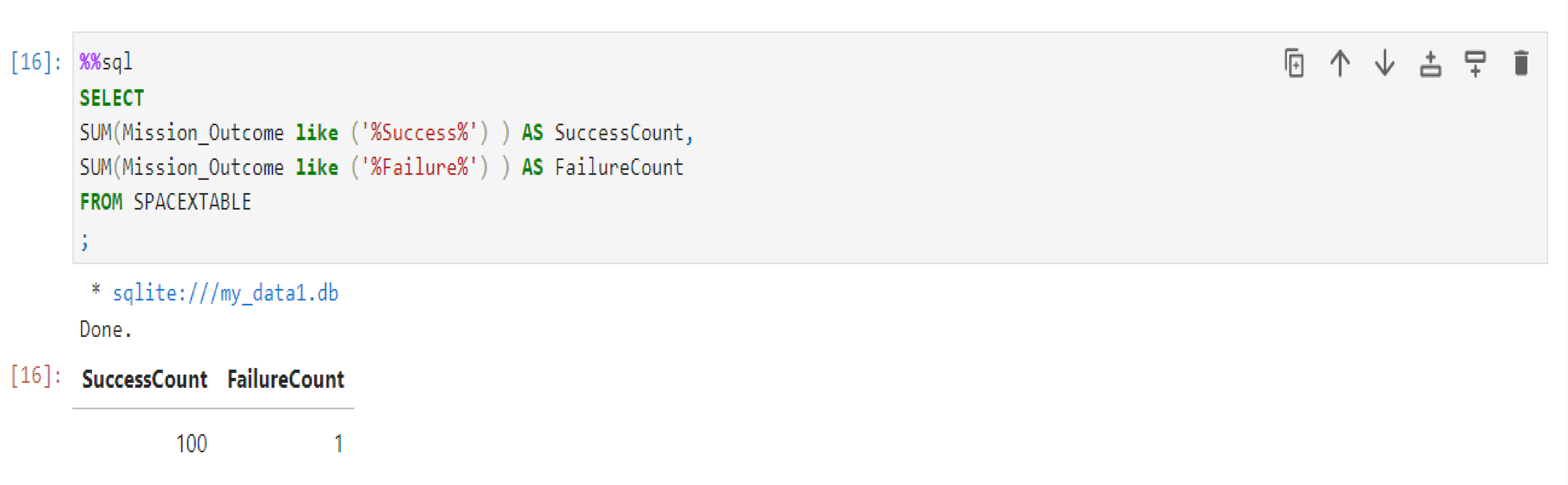
```
 * sqlite:///my_data1.db
Done.
```

[14]:

Booster_Version	Landing_Outcome
F9 FT B1022	Success (drone ship)
F9 FT B1026	Success (drone ship)
F9 FT B1021.2	Success (drone ship)
F9 FT B1031.2	Success (drone ship)

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes by using LIKE and SUM



[16]:

```
%%sql
SELECT
SUM(Mission_Outcome like ('%Success%') ) AS SuccessCount,
SUM(Mission_Outcome like ('%Failure%') ) AS FailureCount
FROM SPACEXTABLE
;
```

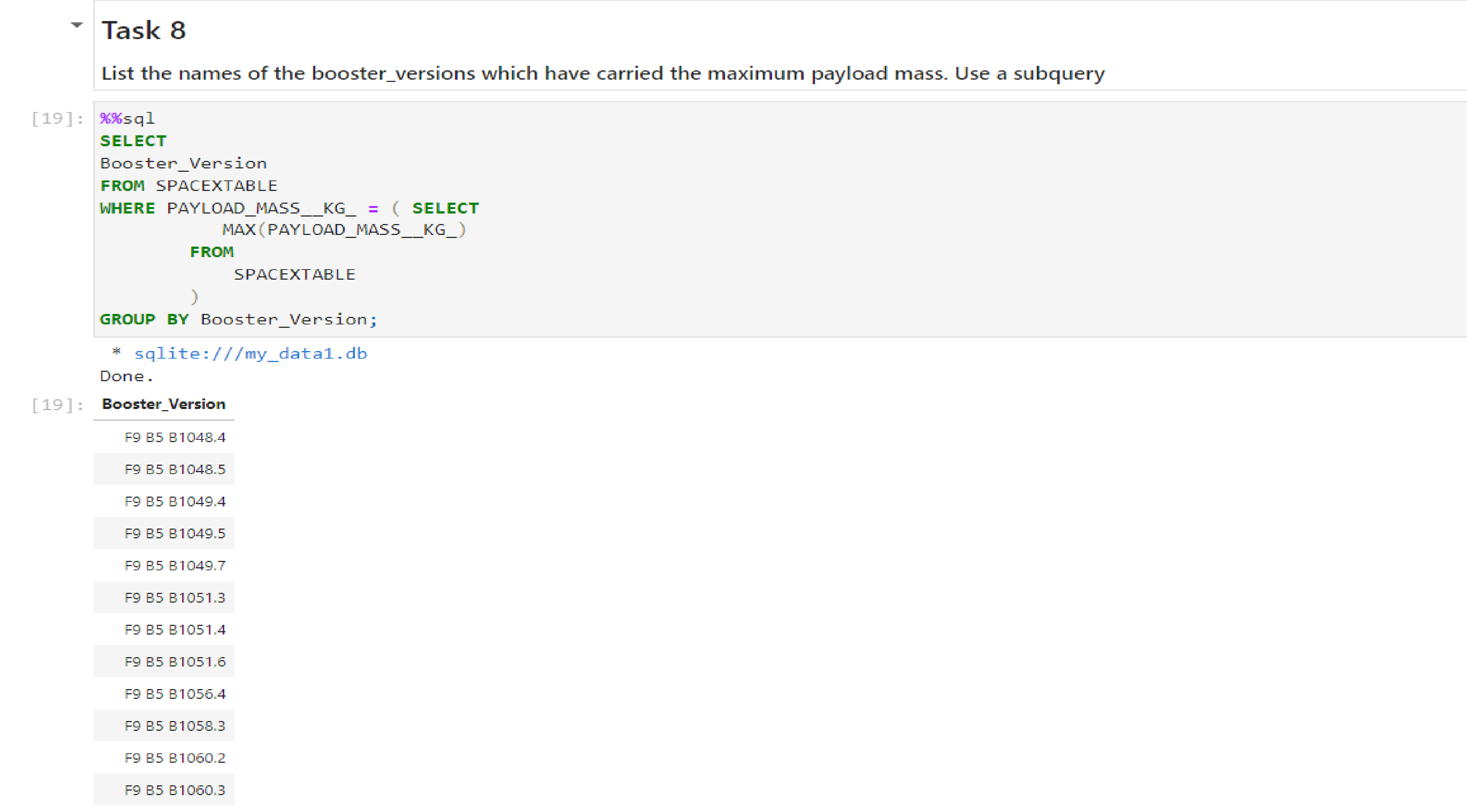
```
 * sqlite:///my_data1.db
Done.
```

[16]:

SuccessCount	FailureCount
100	1

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass by using subquery



Task 8

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

```
[19]: %%sql
SELECT
Booster_Version
FROM SPACEXTABLE
WHERE PAYLOAD_MASS__KG_ = ( SELECT
           MAX(PAYLOAD_MASS__KG_)
        FROM
            SPACEXTABLE
        )
GROUP BY Booster_Version;
```

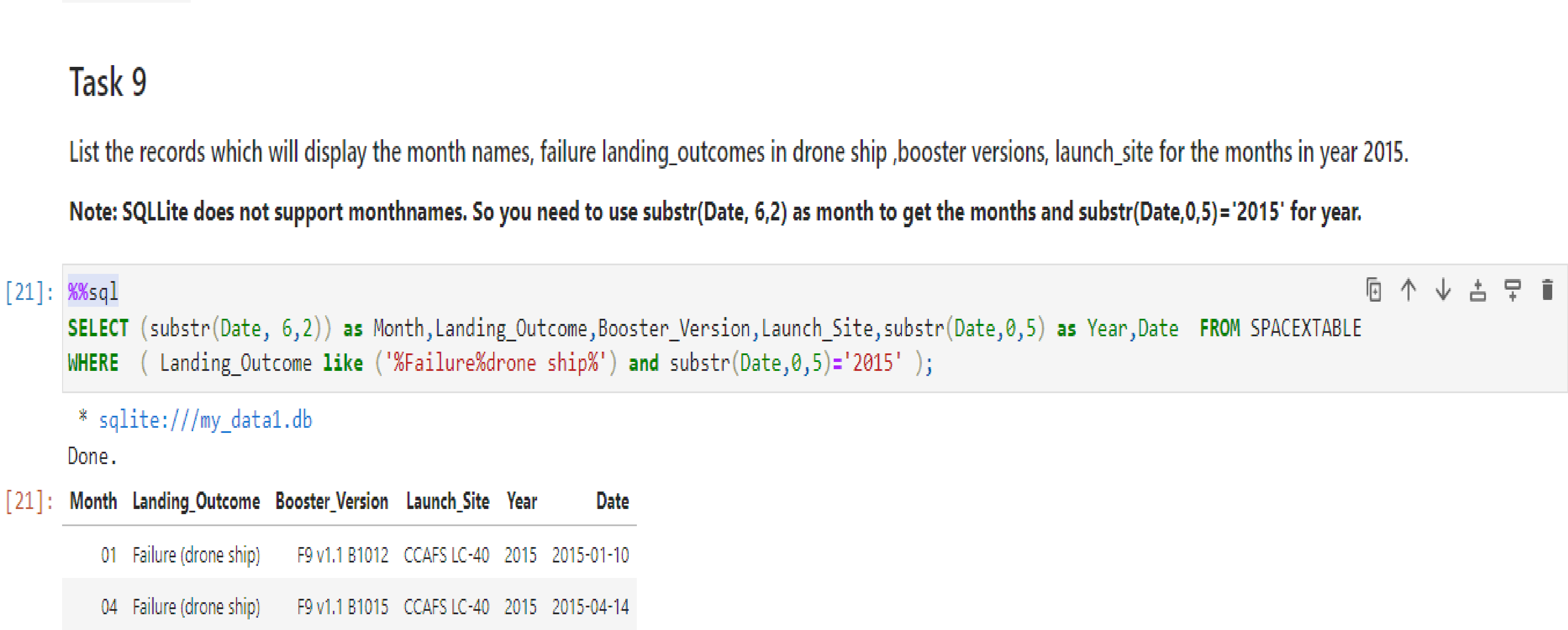
```
 * sqlite:///my_data1.db
Done.
```

[19]:

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

2015 Launch Records

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



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.

```
[21]: %%sql
      SELECT (substr(Date, 6,2)) as Month,Landing_Outcome,Booster_Version,Launch_Site,substr(Date,0,5) as Year,Date  FROM SPACEXTABLE
      WHERE  ( Landing_Outcome like ('%Failure%drone ship%') and substr(Date,0,5)='2015' );
```

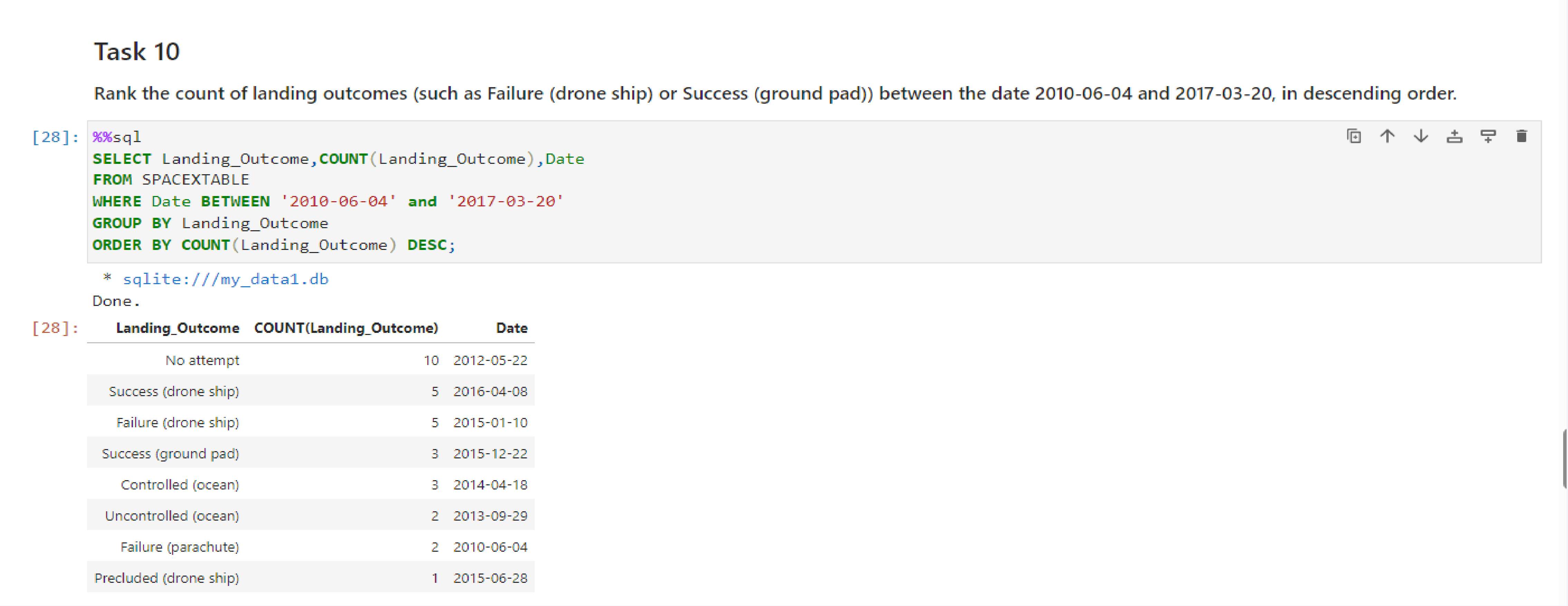
```
 * sqlite:///my_data1.db
Done.
```

[21]:

Month	Landing_Outcome	Booster_Version	Launch_Site	Year	Date
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40	2015	2015-01-10
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40	2015	2015-04-14

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



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.

```
[28]: %%sql
      SELECT Landing_Outcome,COUNT(Landing_Outcome),Date
      FROM SPACEXTABLE
      WHERE Date BETWEEN '2010-06-04' and '2017-03-20'
      GROUP BY Landing_Outcome
      ORDER BY COUNT(Landing_Outcome) DESC;
```

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

[28]:

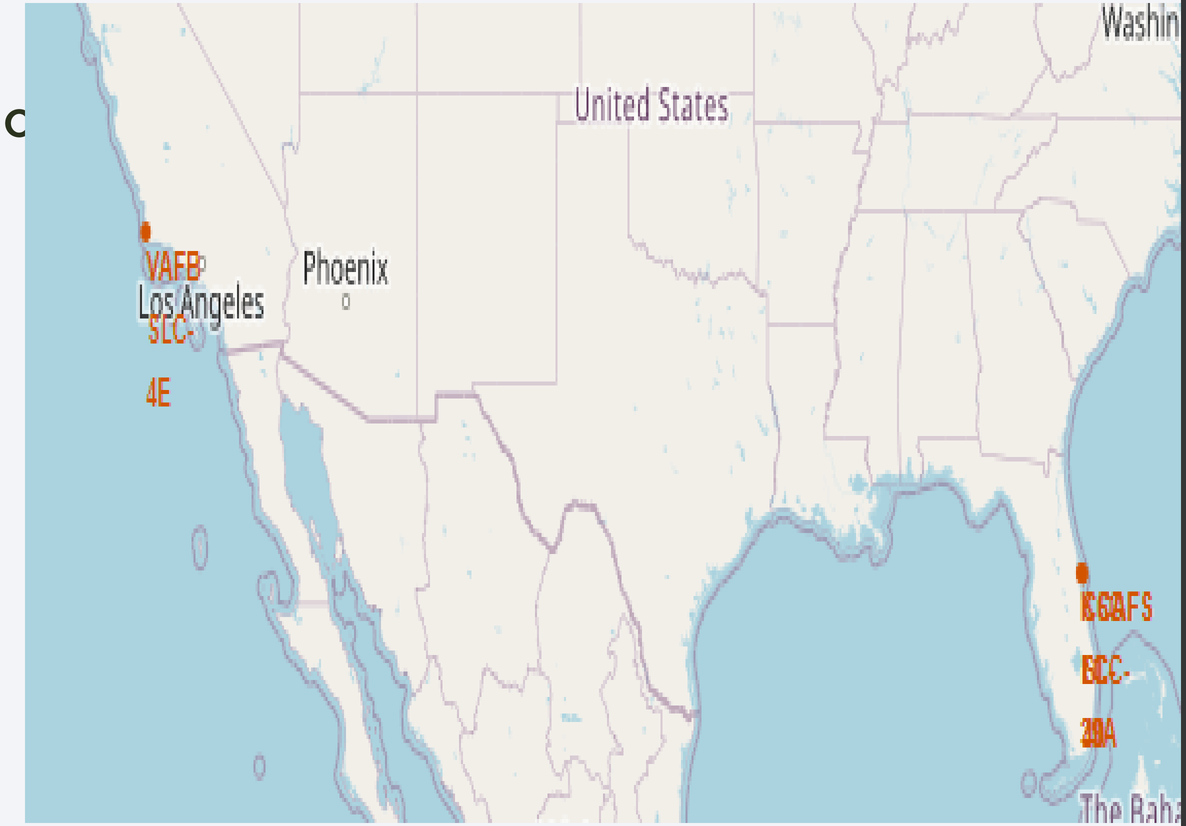
Landing_Outcome	COUNT(Landing_Outcome)	Date
No attempt	10	2012-05-22
Success (drone ship)	5	2016-04-08
Failure (drone ship)	5	2015-01-10
Success (ground pad)	3	2015-12-22
Controlled (ocean)	3	2014-04-18
Uncontrolled (ocean)	2	2013-09-29
Failure (parachute)	2	2010-06-04
Precluded (drone ship)	1	2015-06-28

Section 3

Launch Sites Proximities Analysis

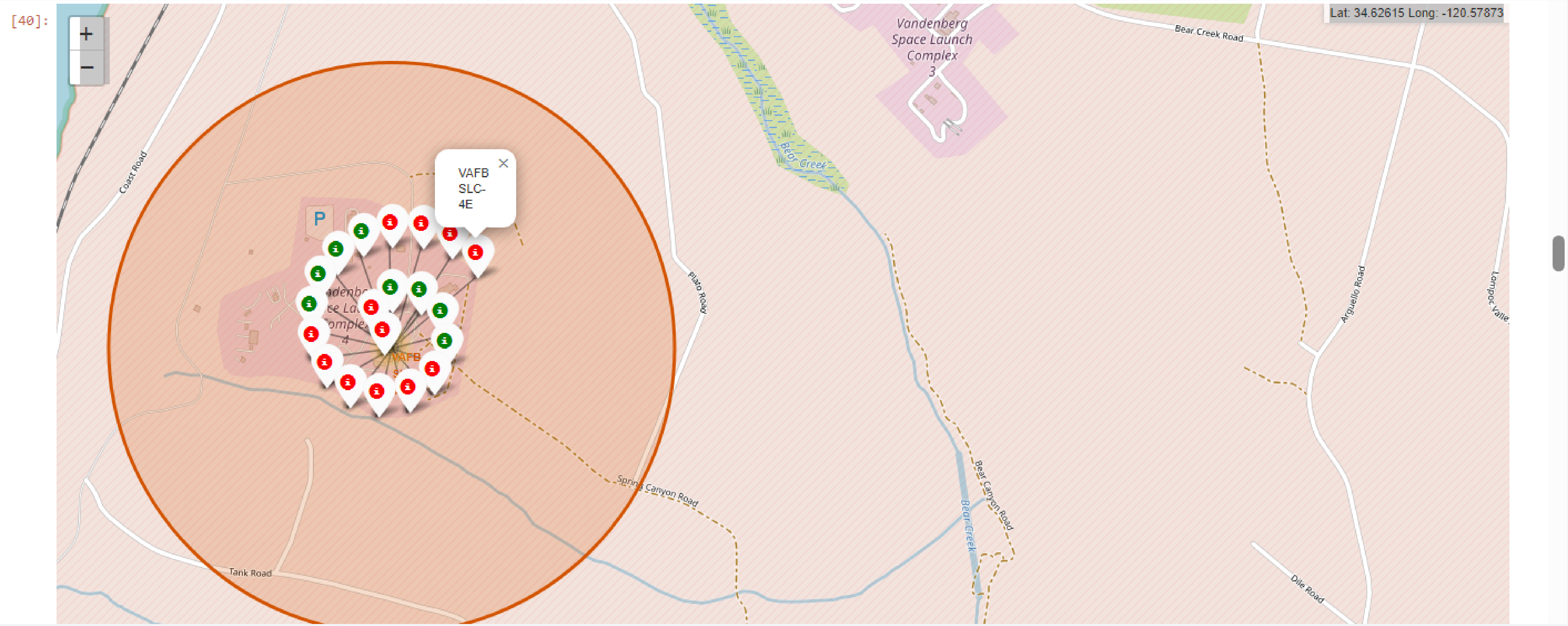
Map With All Launch Sites

- Launch sites are located in the southern half o the U.S and located on the coast.
- This is because being closer to the equator helps with launches and being on the coast allows for missions to be away from people reducing risk.
- They are also located in or near military installations.



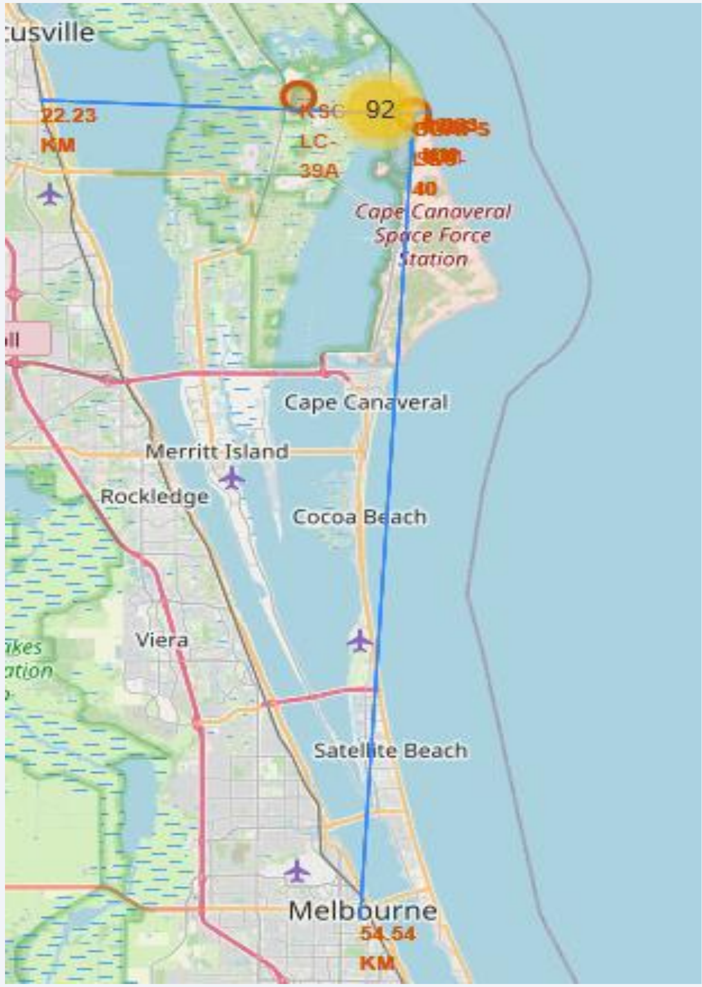
Launch Outcomes For Launch Site VAFB SLC-4E

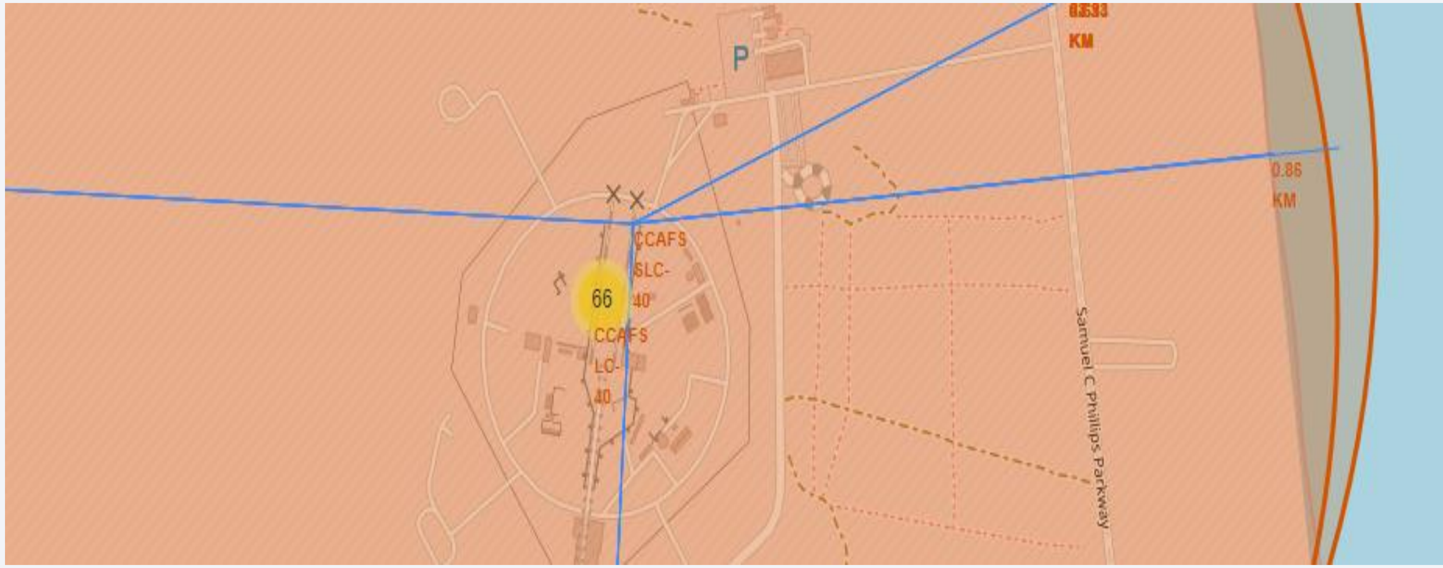
- Launch Successes and Failures happen in chunks for VAFB



Proximity To Landmarks

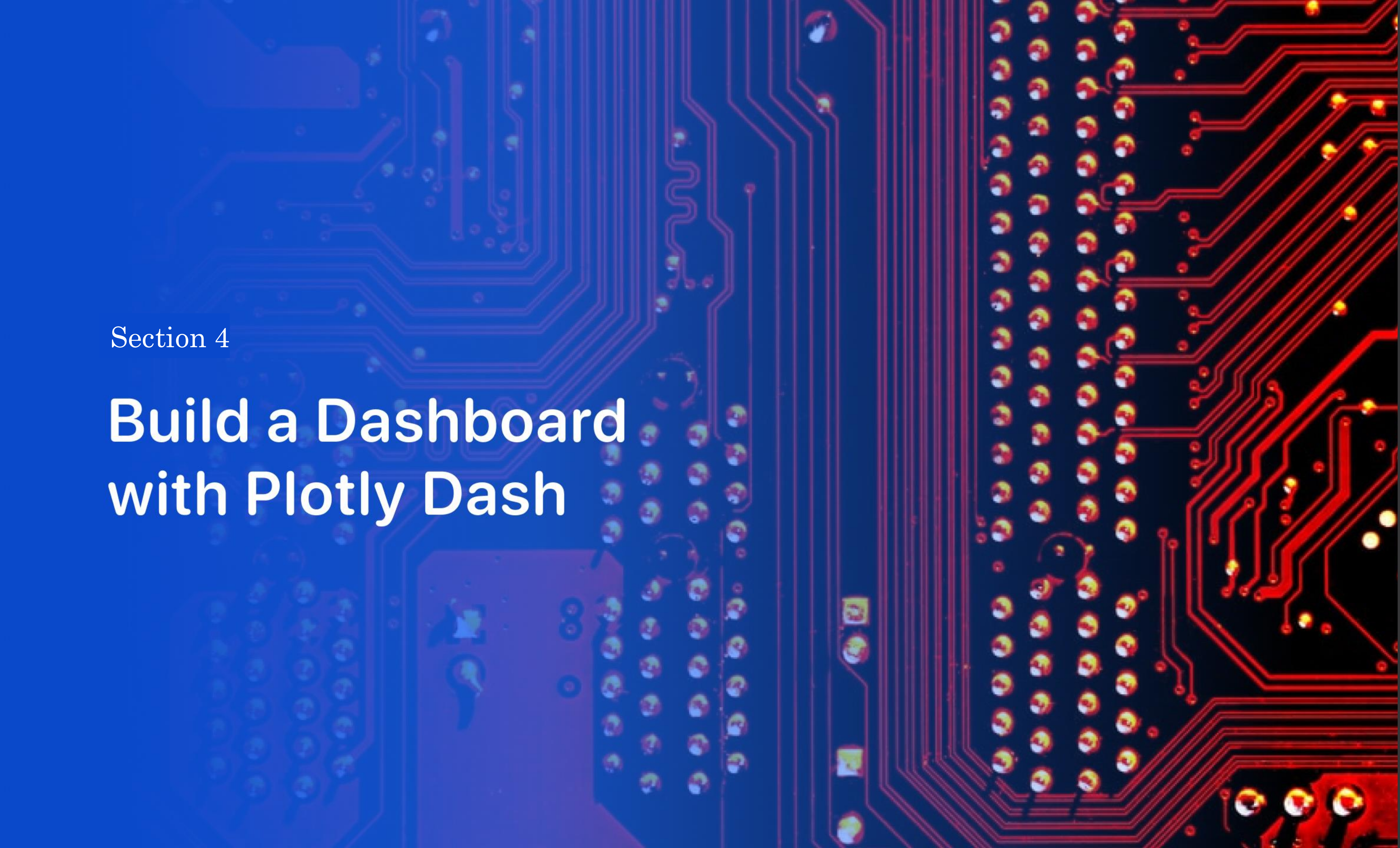
- The important element and finding is that Launch sites are close the sea (.9KM) and roads
- relatively far away from railroads (22.23KM) and city Melbourne (55.54KM) .





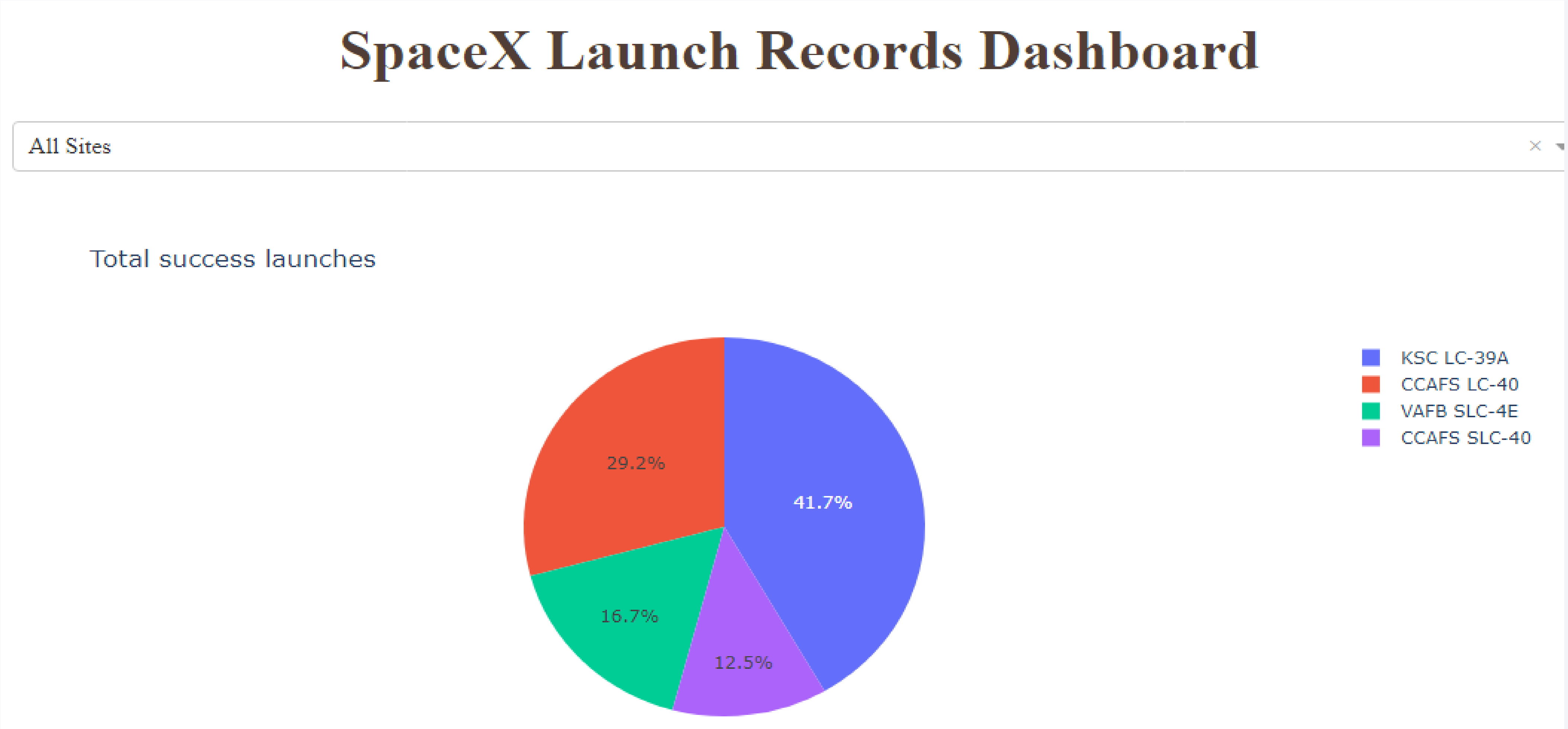
Section 4

Build a Dashboard with Plotly Dash



Launch Success All Sites

- Most successful launches are from KSC LC -39 A or CCAFS LC-40



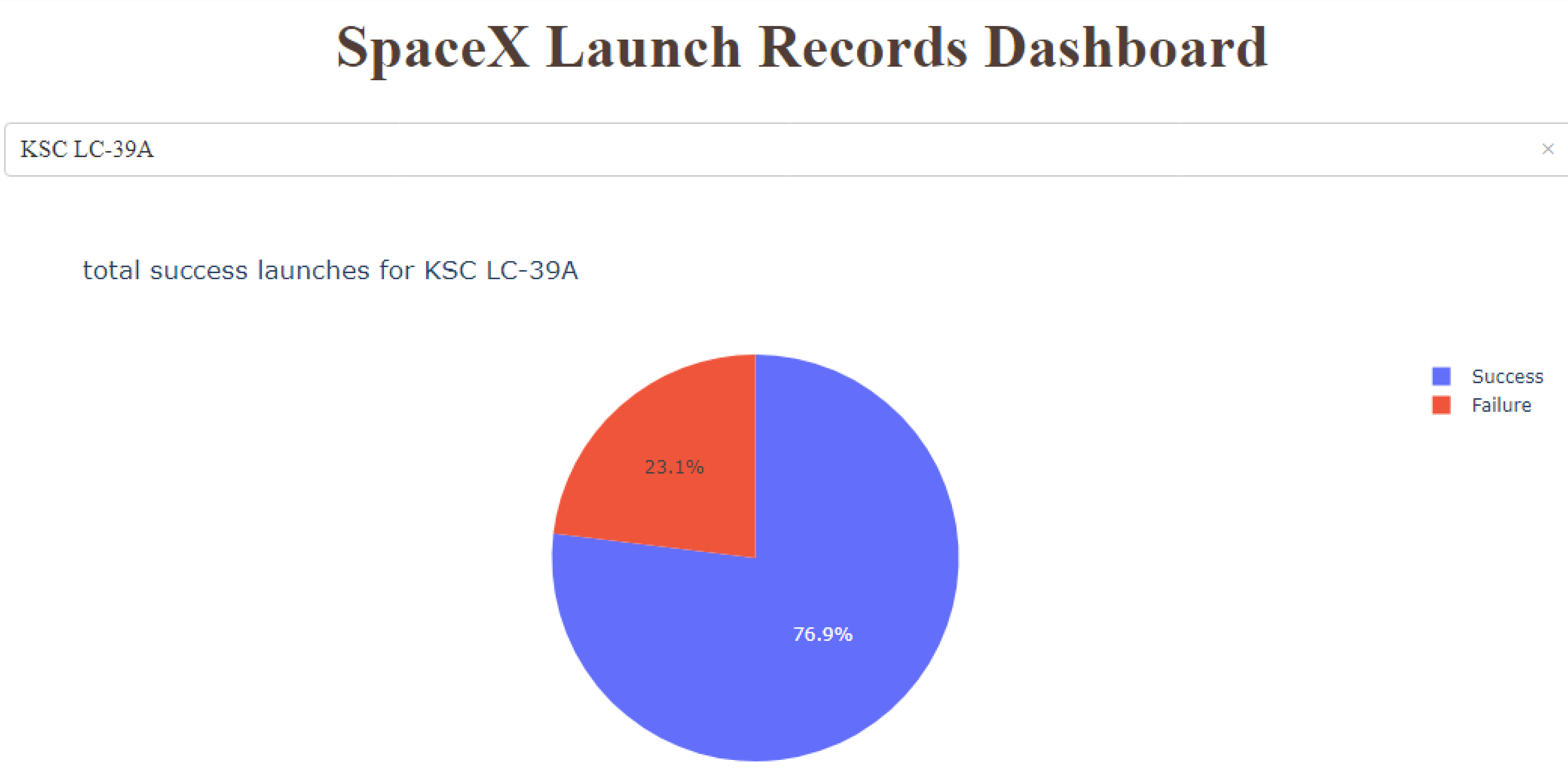
SpaceX Launch Records Dashboard

All Sites

Total success launches

KSC LC-39A Launch Success

- KSC launches have ¾ chance of success



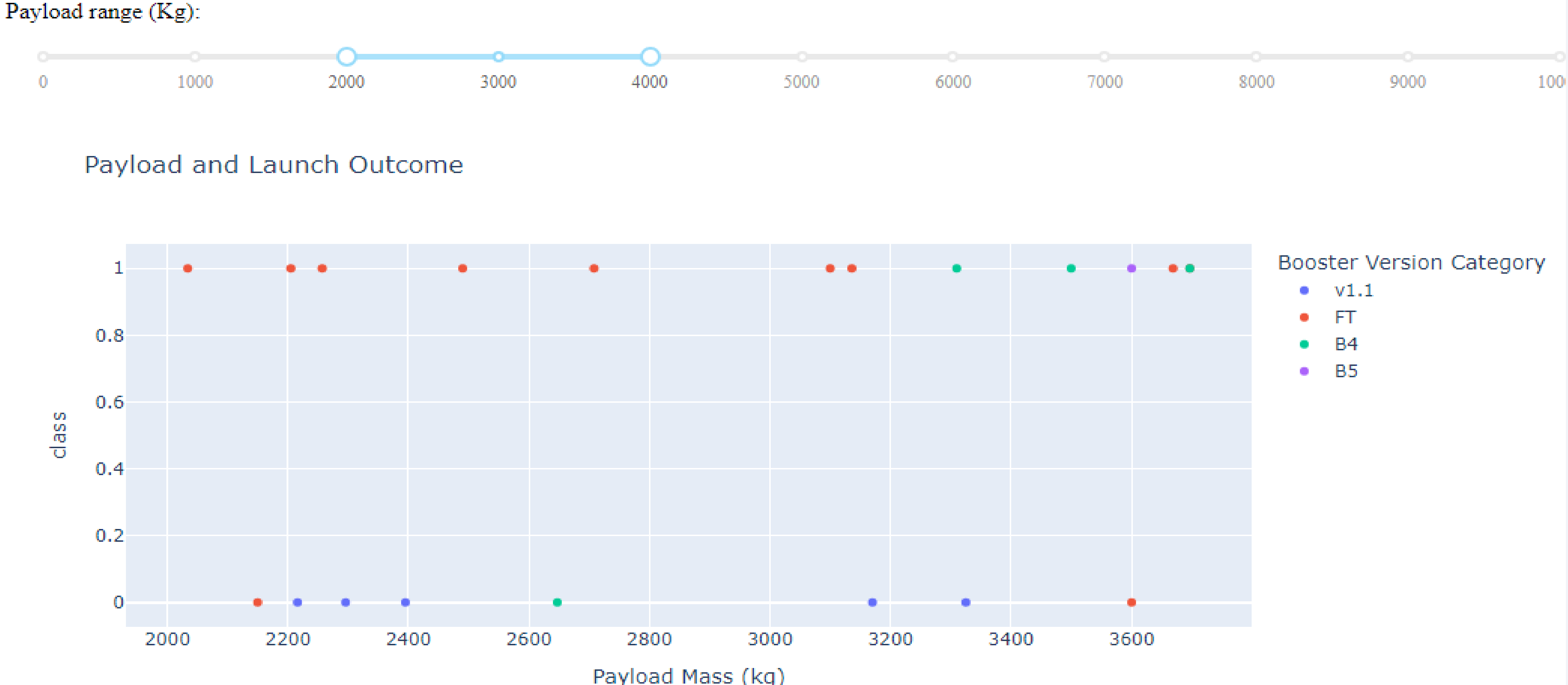
SpaceX Launch Records Dashboard

KSC LC-39A

total success launches for KSC LC-39A

Payload VS. Launch Outcome

- The most successful range is 2000Kg and 4000Kg.
- Booster FT have high success rates.



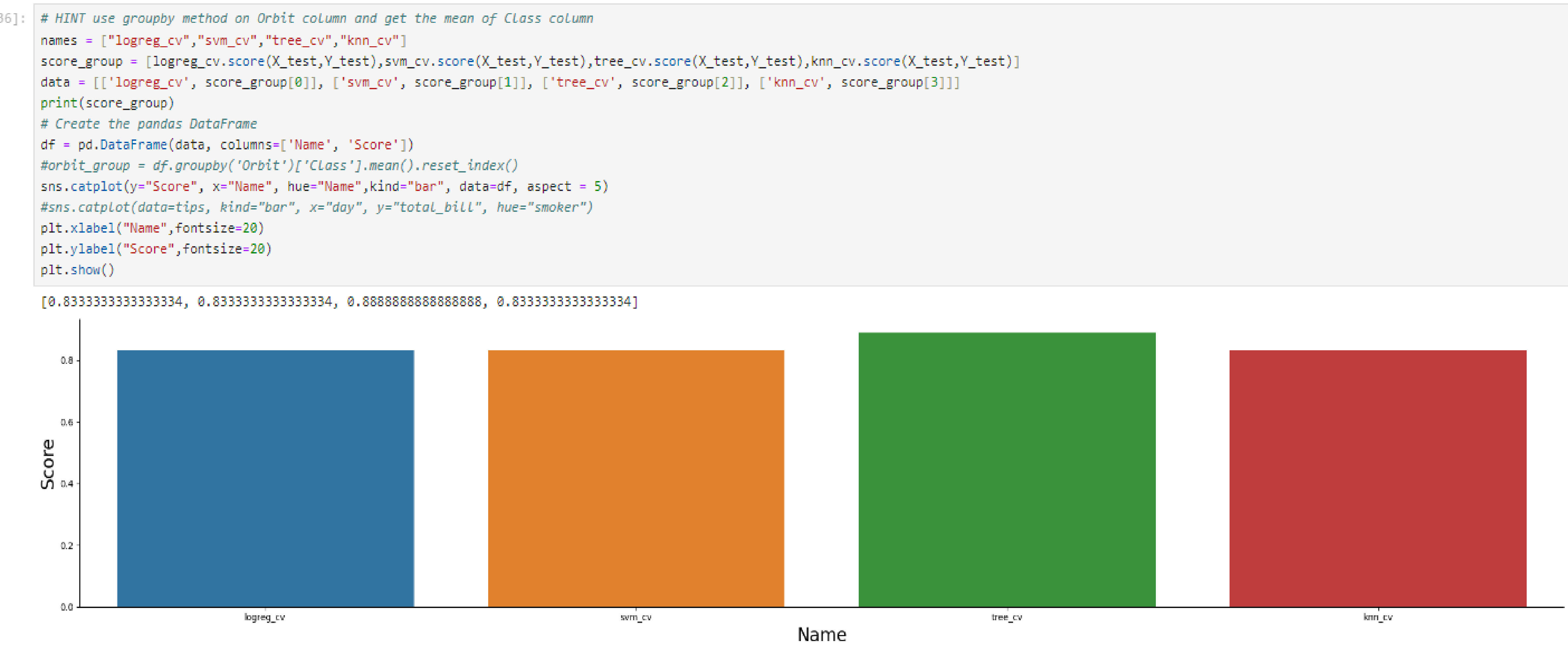
Payload range (Kg):

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Tree has highest score using built in accuracy but models produce different accuracy scores when program is re run

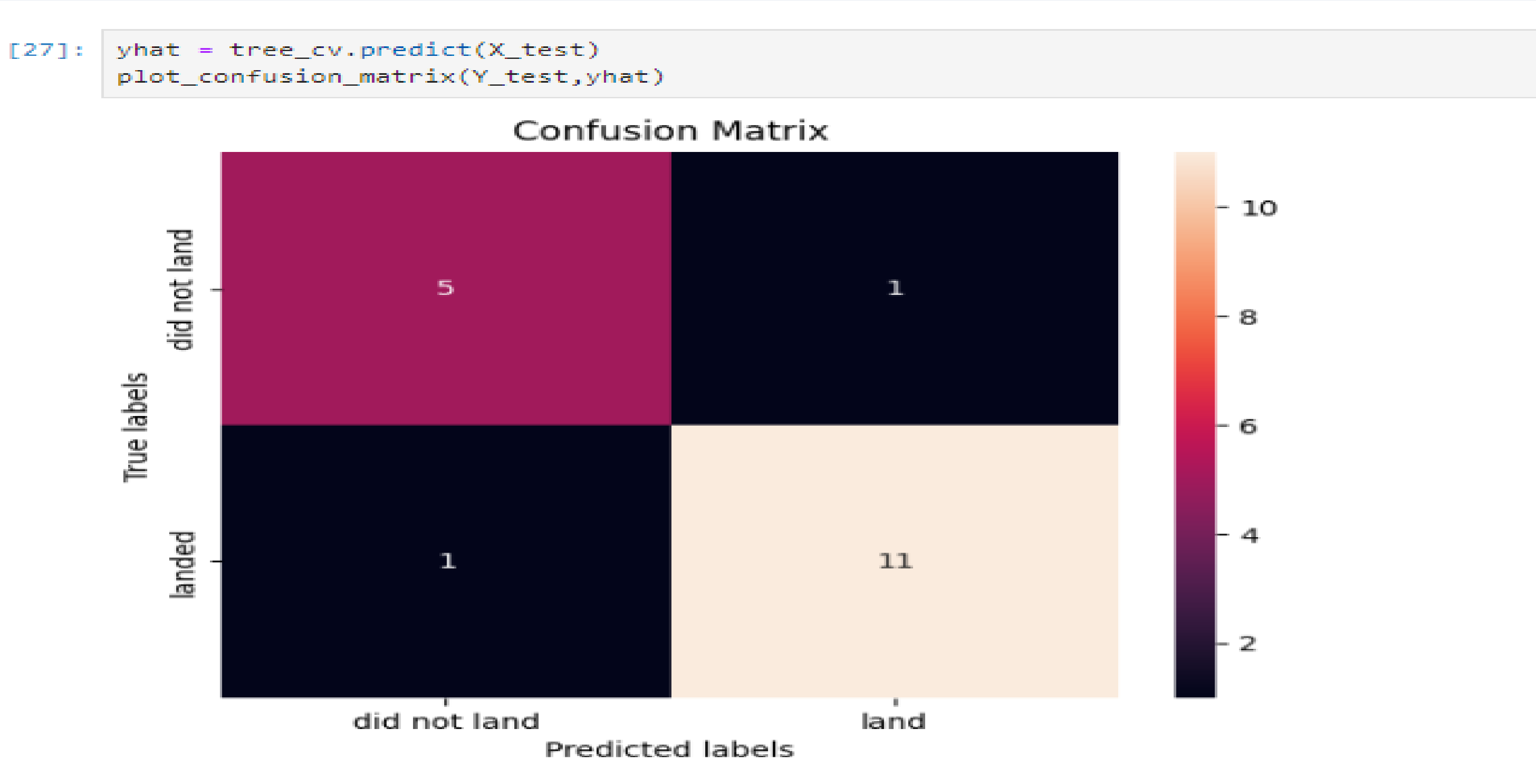


```
# HINT use groupby method on Orbit column and get the mean of Class column
names = ["logreg_cv","svm_cv","tree_cv","knn_cv"]
score_group = [logreg_cv.score(X_test,Y_test),svm_cv.score(X_test,Y_test),tree_cv.score(X_test,Y_test),knn_cv.score(X_test,Y_test)]
data = [['logreg_cv', score_group[0]], ['svm_cv', score_group[1]], ['tree_cv', score_group[2]], ['knn_cv', score_group[3]]]
print(score_group)
# Create the pandas DataFrame
df = pd.DataFrame(data, columns=['Name', 'Score'])
#orbit_group = df.groupby('Orbit')['Class'].mean().reset_index()
sns.catplot(y="Score", x="Name", hue="Name",kind="bar", data=df, aspect = 5)
#sns.catplot(data=tips, kind="bar", x="day", y="total_bill", hue="smoker")
plt.xlabel("Name",fontsize=20)
plt.ylabel("Score",fontsize=20)
plt.show()
```

```
[0.8333333333333334, 0.8333333333333334, 0.8888888888888888, 0.8333333333333334]
```

Confusion Matrix

- The model is good at predicting True Negative
- Model is Average at predicting True Positive



```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

Conclusions

- Lauches landing success have gone up over the years.
- The launch site KSC LC-39A has a high probability of success
- Booster FT has high probability of success
- Launch Sites are Located near the southern U.S coast
- Tree classifier is the best predictive modeler of weather a Lauch will be successful

Appendix

- GitHub - davivaug2/Applied-Data-Science-Capstone-IBM-DV

Thank you!