IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Amith Naik
26-05-2024>

Outline

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

Executive Summary

- Summary of methodologies

  Data is collected and cleaning

  Data analysis

  Interactive data visualization reports

  Predictive analysis

- Summary of all results

Introduction

- 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

Section 1

Methodology

Methodology

Executive Summary

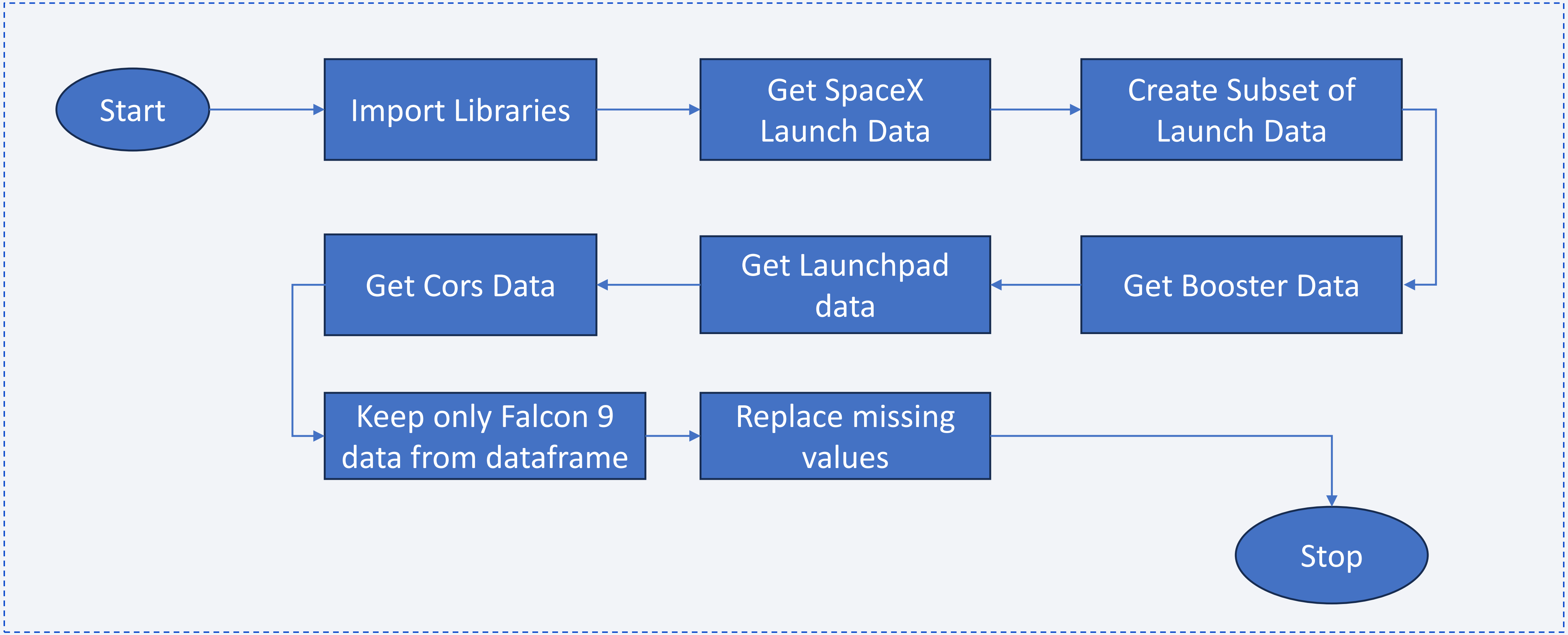
- Data collection methodology:
  - SpaceX launch data is gathered from an API, specifically the SpaceX REST API
  - Data is converted into data frame and added a new column ‘class’ to identify landing results
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Logistic Regression
  - Support Vector Scalar Regression
  - Decision Tree Regression

Data Collection – SpaceX API

- SpaceX data collection URLs:
  - Launch data URL - https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json
  - Get Booster data by rocket id - https://api.spacexdata.com/v4/rockets/{rocketId}
  - Get Launchpad data by launchpad id - https://api.spacexdata.com/v4/launchpads/{launchpadId}
  - Get Payload data by payload id - https://api.spacexdata.com/v4/payloads/{payloadId}
  - Get Cores data by core id - https://api.spacexdata.com/v4/cores/{coreId}

- GitHub link for the notebook- SpaceXDataCollection

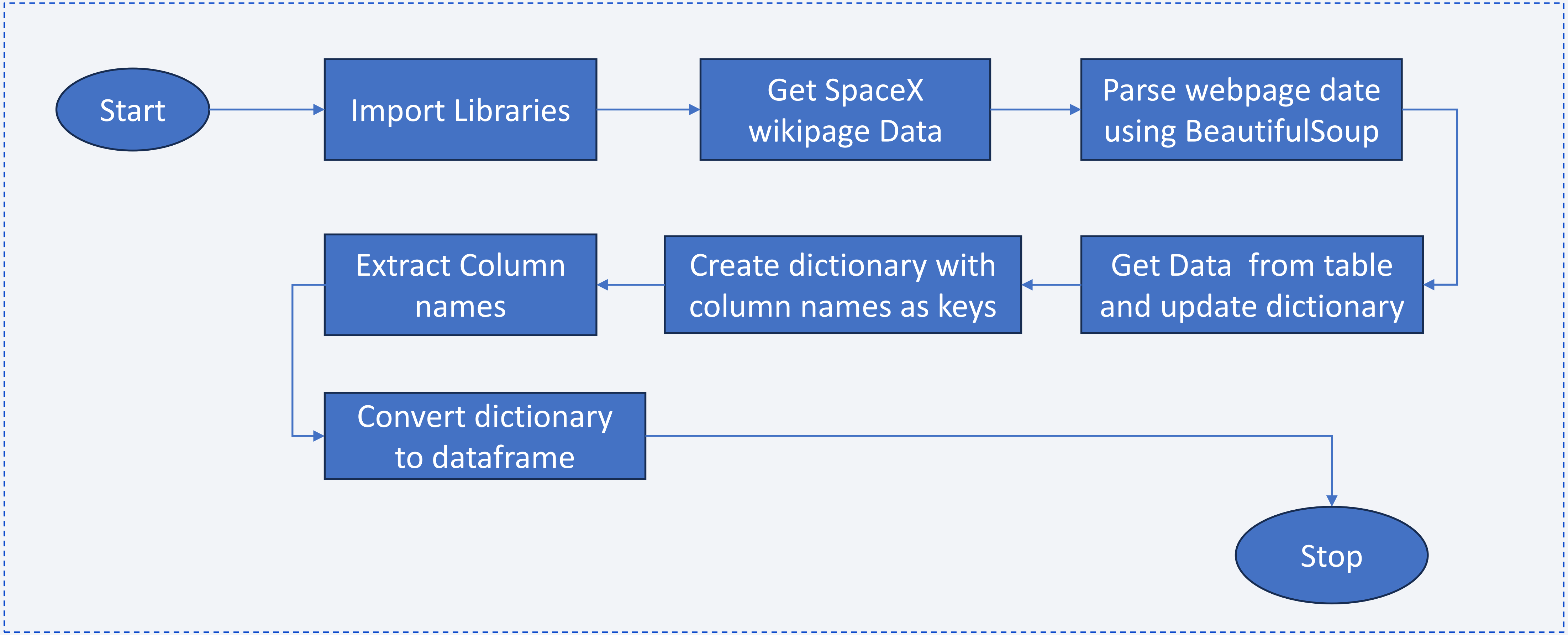
Data Collection Flowchart– SpaceX API



Data Collection - Scraping

- Steps to read and convert web data into a Dataframe
  - Read web data using request url - https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922
  - Create a BeautifulSoup object from a response text content [spacex_soup = BeautifulSoup(html_data,'html.parser')]
  - Extract column names by reading table header(th) using . find_all() method
  - Create an empty dictionary with keys from the extracted column names.
  - Iterate over table rows to read data and update dictionary.
  - Create a data frame by parsing the launch HTML tables.
- GitHub link for the notebook : [SpaceXDataWebScraping]()

Data Collection - Scraping


Start
Import Libraries
Get SpaceX wikipage Data
Parse webpage date using BeautifulSoup
Get Data from table and update dictionary
Create dictionary with column names as keys
Extract Column names
Convert dictionary to dataframe
Stop

Data Wrangling

- Identify the columns with null values using isnull() method.
- Identify occurrences of LaunchSite and Orbit using value_counts() method.
- Identity occurrences of mission outcome and understand success and failure values
- Create new column Class for storing landing outcome in Boolean format
- Determine success rate using mean() method on column Class

GitHub link for Notebook - SapceXDataWrangling

EDA with Data Visualization

- Scatter Plot - Flight Number vs Payload Mass and Class as hue
- Scatter Plot - Flight Number vs Launch site, and Class as hue
- Scatter Plot – Payload mass vs Launch site, and Class as hue
- Bar Plot - For the Success rate of each orbit
- Line Plot -Year vs Success rate
- Use OneHotEncoder to change object type column to float

- GitHub link for Notebook - SpaceXDataVisualization

EDA with SQL

- Display the names of the unique launch sites in the space mission [Select Distinct "Launch_Site" from SPACEXTABLE].
- Display 5 records where launch sites begin with the string 'CCA' [Select * from SPACEXTABLE where "Launch_Site" like 'CCA%' limit 5]
- Display the total payload mass carried by boosters launched by NASA (CRS) Sselect SUM("PAYLOAD_MASS__KG_") from SPACEXTABLE where "Customer" = "NASA (CRS)“]
- Display average payload mass carried by booster version F9 v1.1 [Select AVG("PAYLOAD_MASS__KG_") from SPACEXTABLE where "Booster_Version" like "F9 v1.1“]
- List the date when the first succesful landing outcome in ground pad was achieved [Select min(Date) from SPACEXTABLE where "Landing_Outcome" = "Success (ground pad)“]

- GitHub link for Notebook - SpaceXEDAwithSQL

Build an Interactive Map with Folium

- We have used folium Circle and Marker object to add a highlighted circle area with a text label on a specific coordinate.
- Marker Cluster is used to provide insight on Success and Failure data on each Launch Sites.
- We have used Mouse Position on the map to get coordinate for a mouse over a point on the map. As such, while you are exploring the map, we can easily find the coordinates of any points of interests.
- We have used PolyLine to draw a line between a launch site to the selected coastline point.

- Coordinates are just plain numbers that can not give you any intuitive insights about where are those launch sites. Visualizing those locations by pinning them on a map helps in better understanding.

- GitHub link for Notebook - [SpaceXMapswithFolium]()

Build a Dashboard with Plotly Dash

- We have used dash Dropdown for Launch Sites. User can select individual or All launch sites to get Success rates.
- We have used Range Slider to see success rates as per Payload mass.
- We have used call back functions for dropdown and range slider value change event to update the reports as per user activity.
- User can visualize success rate by Launch Site and Payload mass.

- GitHub link for Python File - DashboardWithPlotlyDash

Predictive Analysis (Classification)

- Load Dataframe using APIs
- Standardize X object using StandardScalar() object.
- Split X and Y for Training and Testing purpose using train_test_split() method.
- Evaluate the model using GridSearchCV by creating LogisticRegression Model. Calculate the Accuracy of the model.
- Evaluate the model using GridSearchCV by creating Support Vector Machine Model. Calculate the Accuracy of the model.
- Evaluate the model using GridSearchCV by creating DecissionTree Model. Calculate the Accuracy of the model.
- Evaluate the model using GridSearchCV by creating K Nearest Neighbor Model. Calculate the Accuracy of the model.
- GitHub Link for the Notebook - SpaceXPredictiveAnalysis

Results

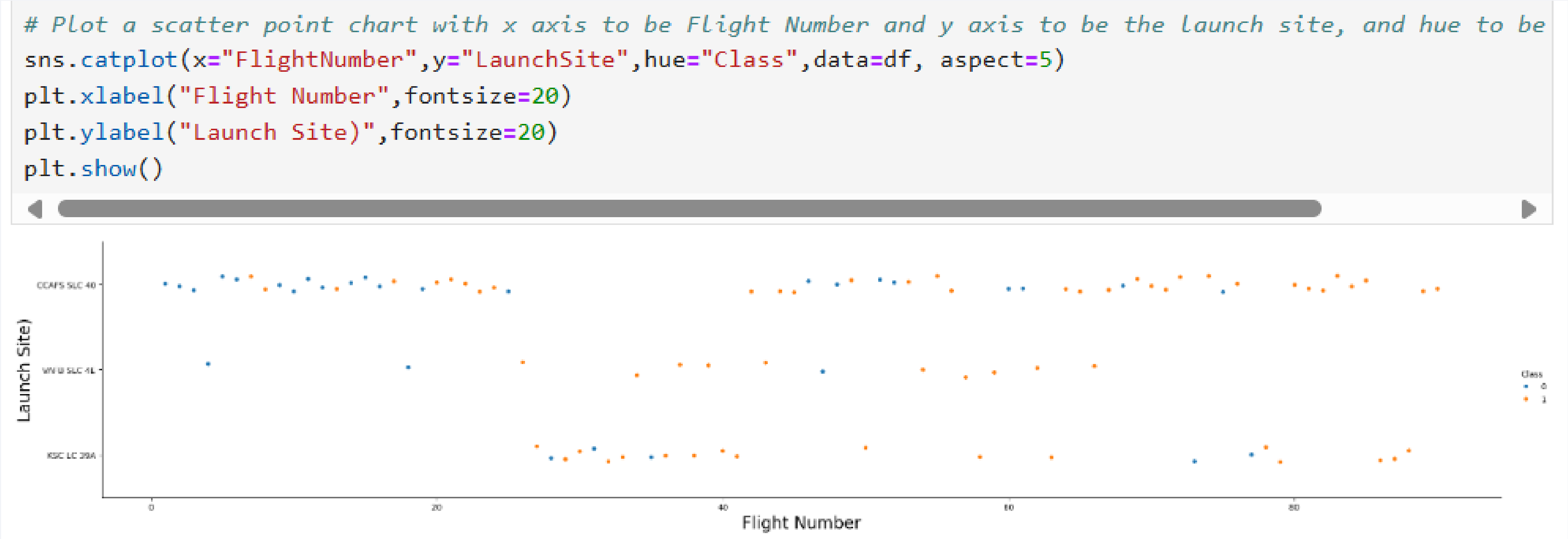
- Logistic Regression Accuracy – 0.822
- Support Vector Machine – Best type is Sigmoid with accuracy 0.822
- Decision Tree Accuracy – 0.877
- K- Nearest Neighbor Accuracy – 0.844

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

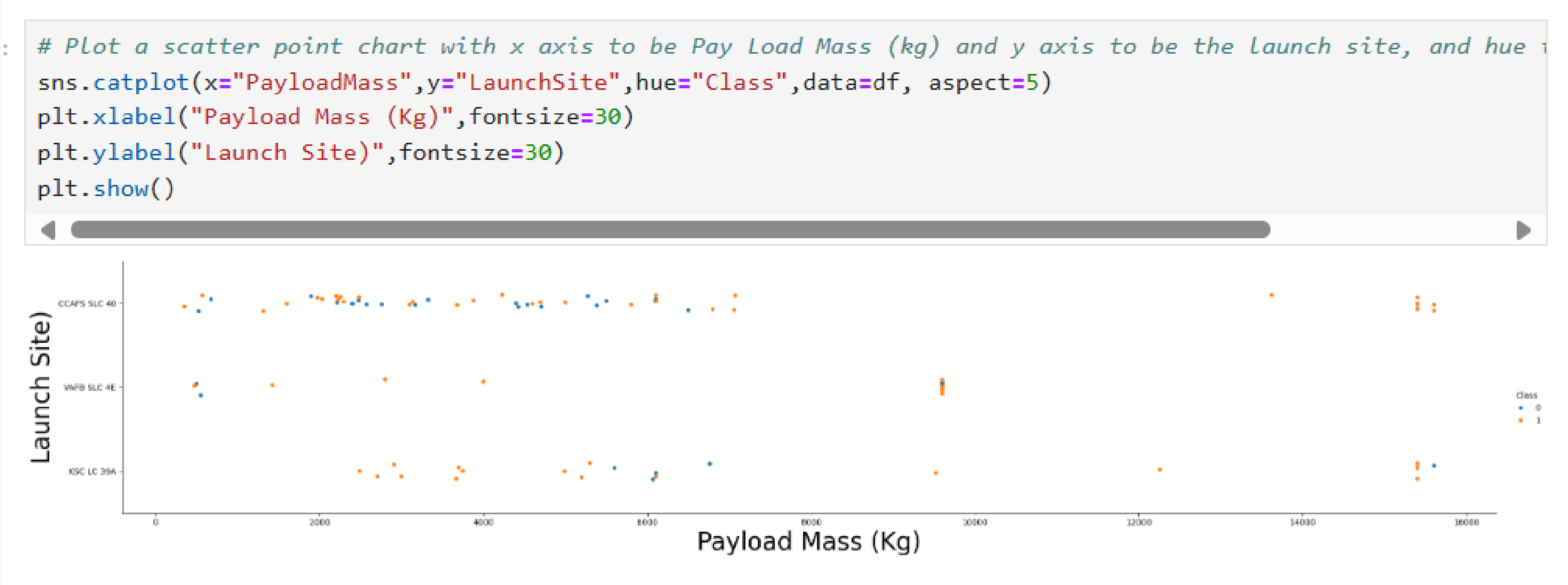
- Show a scatter plot of Flight Number vs. Launch Site



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

Payload vs. Launch Site

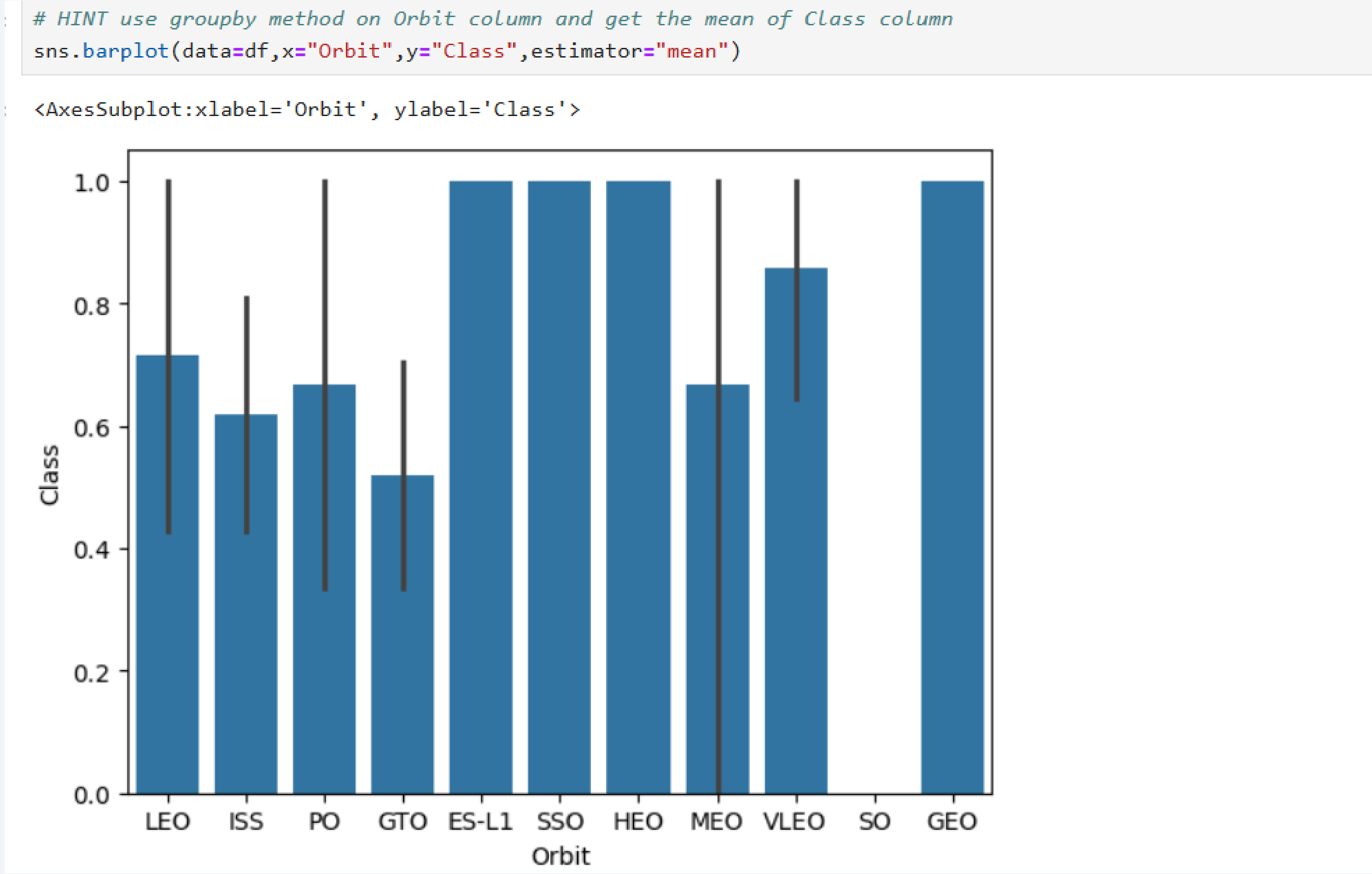
- Show a scatter plot of Payload vs. Launch Site



```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue
sns.catplot(x="PayloadMass",y="LaunchSite",hue="Class",data=df, aspect=5)
plt.xlabel("Payload Mass (Kg)",fontsize=30)
plt.ylabel("Launch Site)",fontsize=30)
plt.show()
```

Success Rate vs. Orbit Type

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

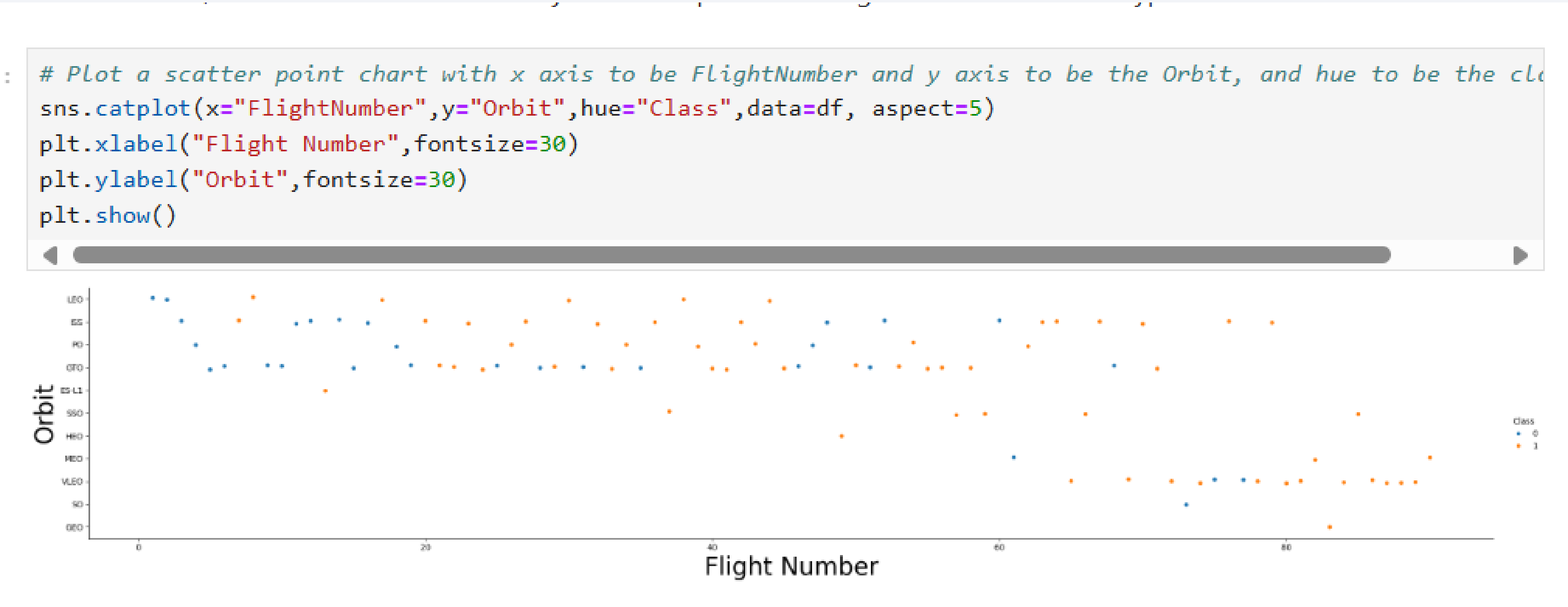


```
# HINT use groupby method on Orbit column and get the mean of Class column
sns.barplot(data=df,x="Orbit",y="Class",estimator="mean")
```

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

Flight Number vs. Orbit Type

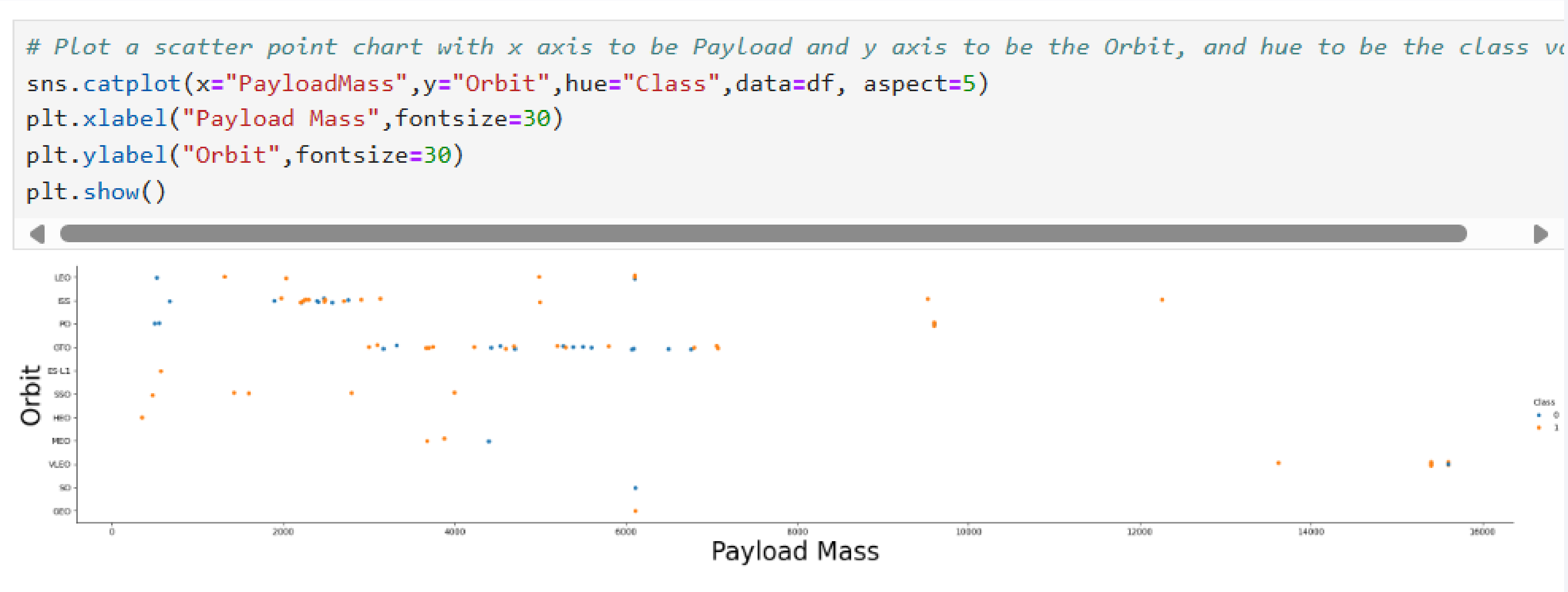
- Show a scatter point of Flight number vs. Orbit type



```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the cl
sns.catplot(x="FlightNumber",y="Orbit",hue="Class",data=df, aspect=5)
plt.xlabel("Flight Number",fontsize=30)
plt.ylabel("Orbit",fontsize=30)
plt.show()
```

Payload vs. Orbit Type

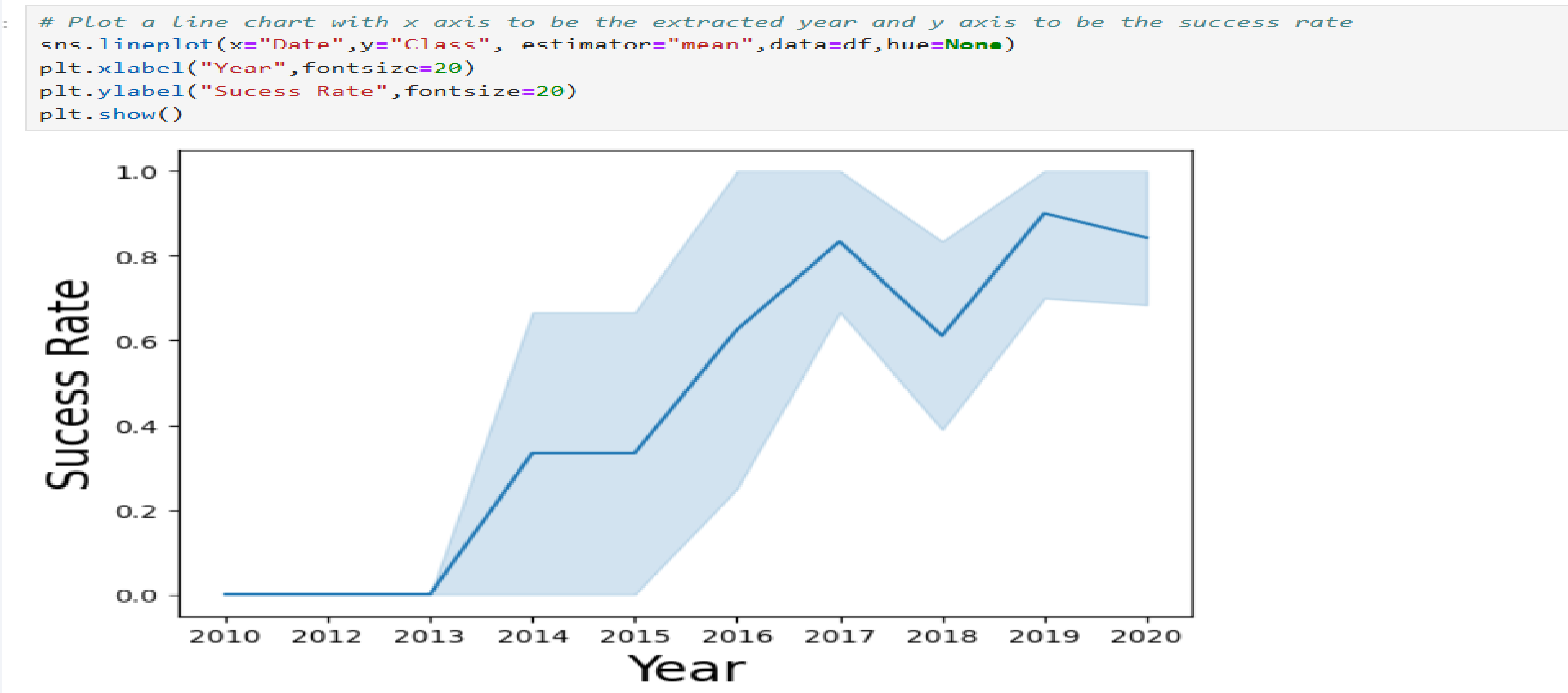
- Show a scatter point of payload vs. orbit type



```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class v
sns.catplot(x="PayloadMass",y="Orbit",hue="Class",data=df, aspect=5)
plt.xlabel("Payload Mass",fontsize=30)
plt.ylabel("Orbit",fontsize=30)
plt.show()
```

Launch Success Yearly Trend

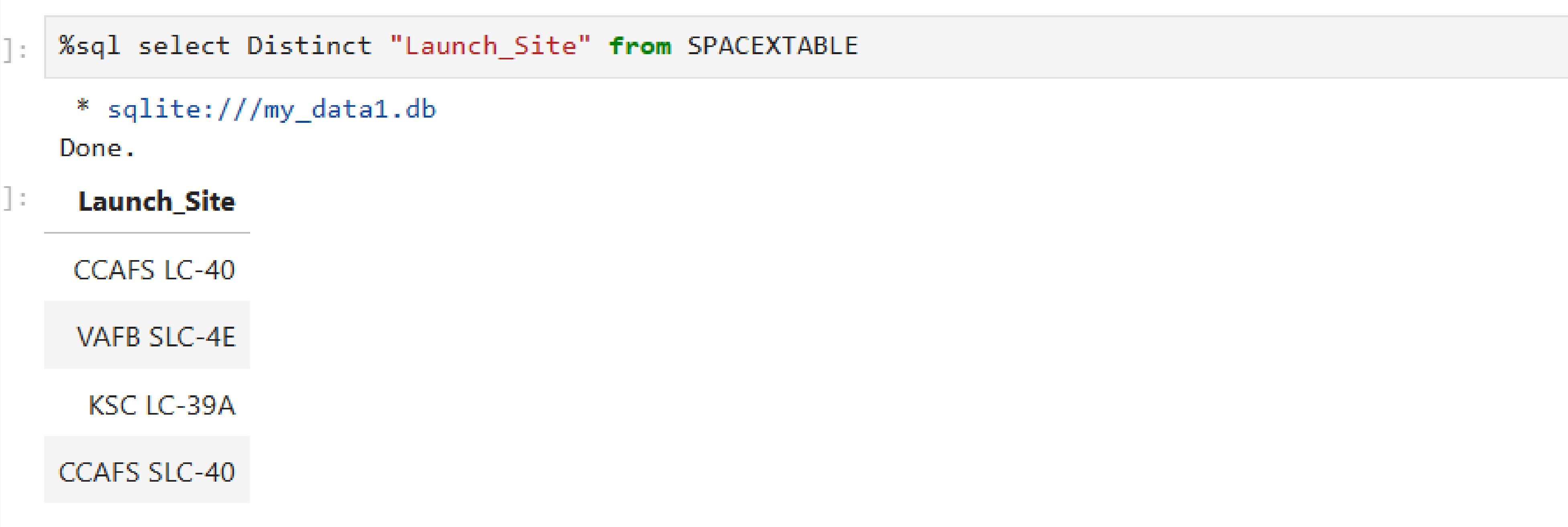
- Show a line chart of yearly average success rate



```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
sns.lineplot(x="Date",y="Class", estimator="mean",data=df,hue=None)
plt.xlabel("Year",fontsize=20)
plt.ylabel("Sucess Rate",fontsize=20)
plt.show()
```

All Launch Site Names

- Find the names of the unique launch sites



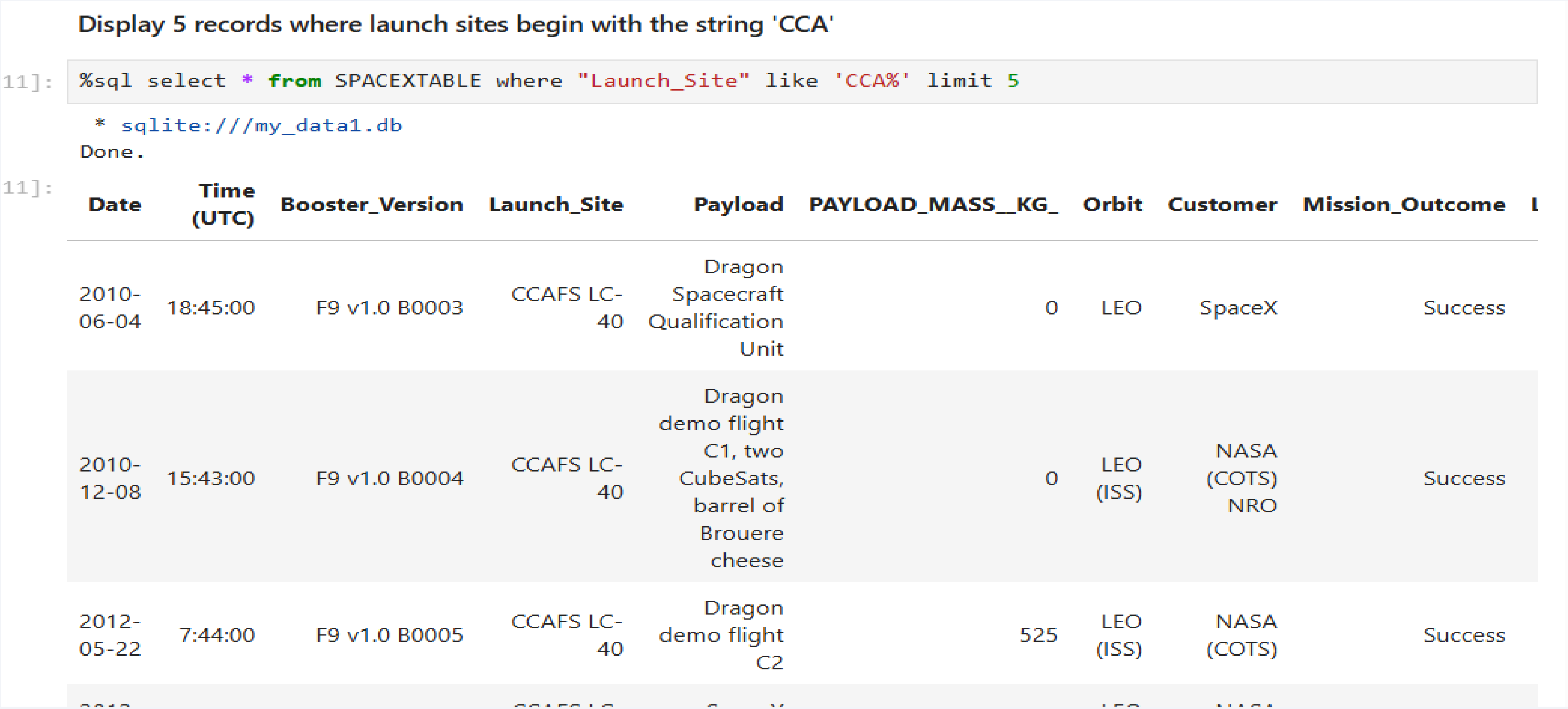
```
]: %sql select Distinct "Launch_Site" from SPACEXTABLE
```

```
 * 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`



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

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

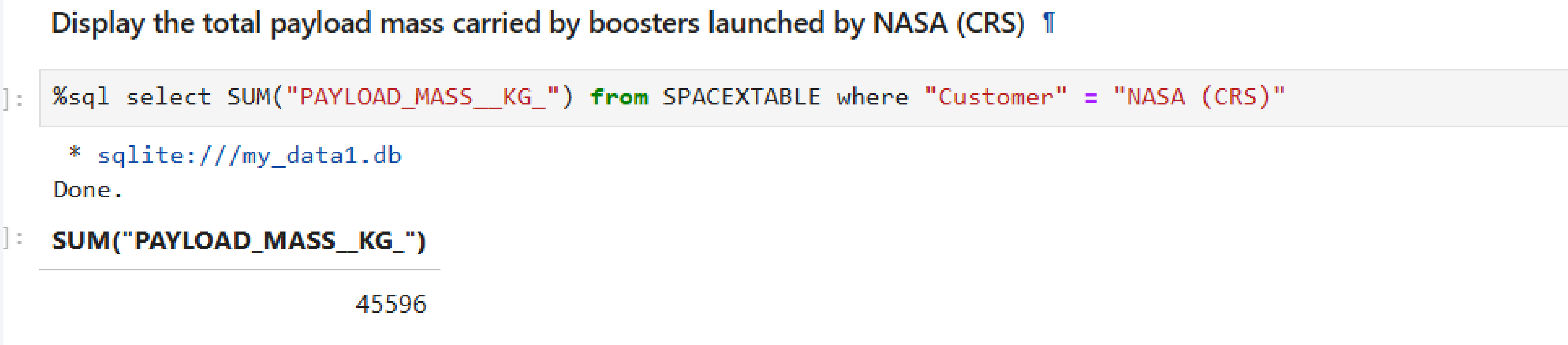
```
 * sqlite:///my_data1.db
Done.
```

11]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	L
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	
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	
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	

Total Payload Mass

- Calculate the total payload carried by boosters from NASA



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

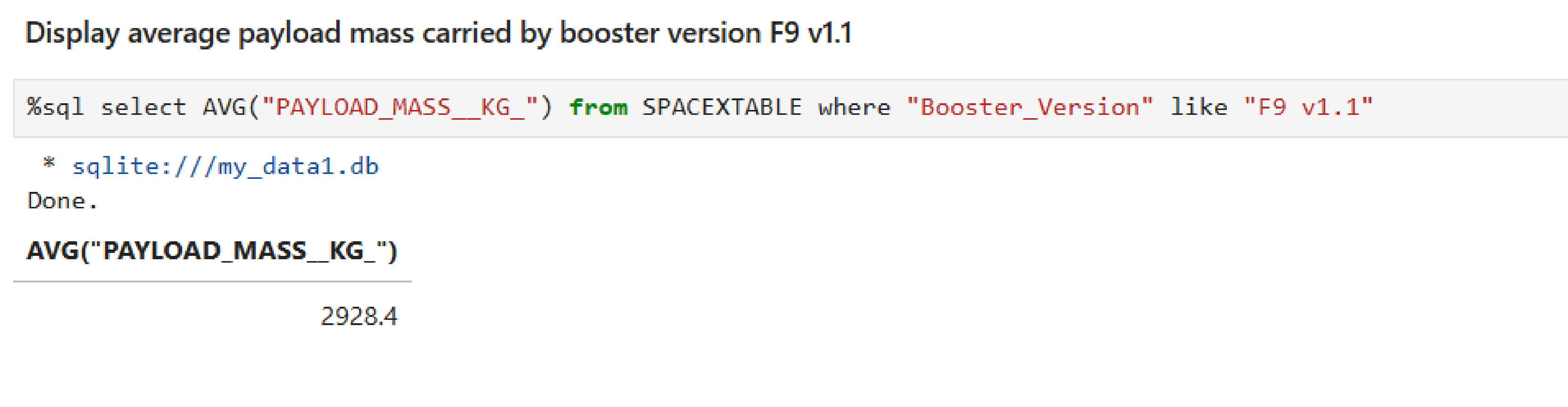
```
]: %sql select SUM("PAYLOAD_MASS__KG_") from SPACEXTABLE where "Customer" = "NASA (CRS)"
```

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

SUM("PAYLOAD_MASS__KG_")
45596

Average Payload Mass by F9 v1.1

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



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

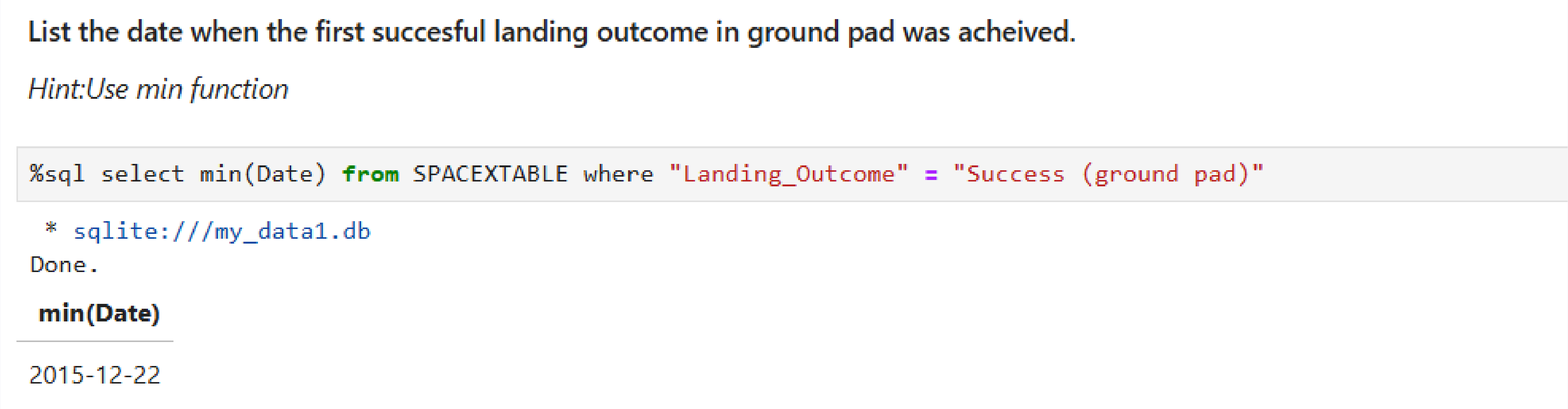
```
%sql select AVG("PAYLOAD_MASS__KG_") from SPACEXTABLE where "Booster_Version" like "F9 v1.1"
```

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

AVG("PAYLOAD_MASS__KG_")
2928.4

First Successful Ground Landing Date

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



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

Hint:Use min function

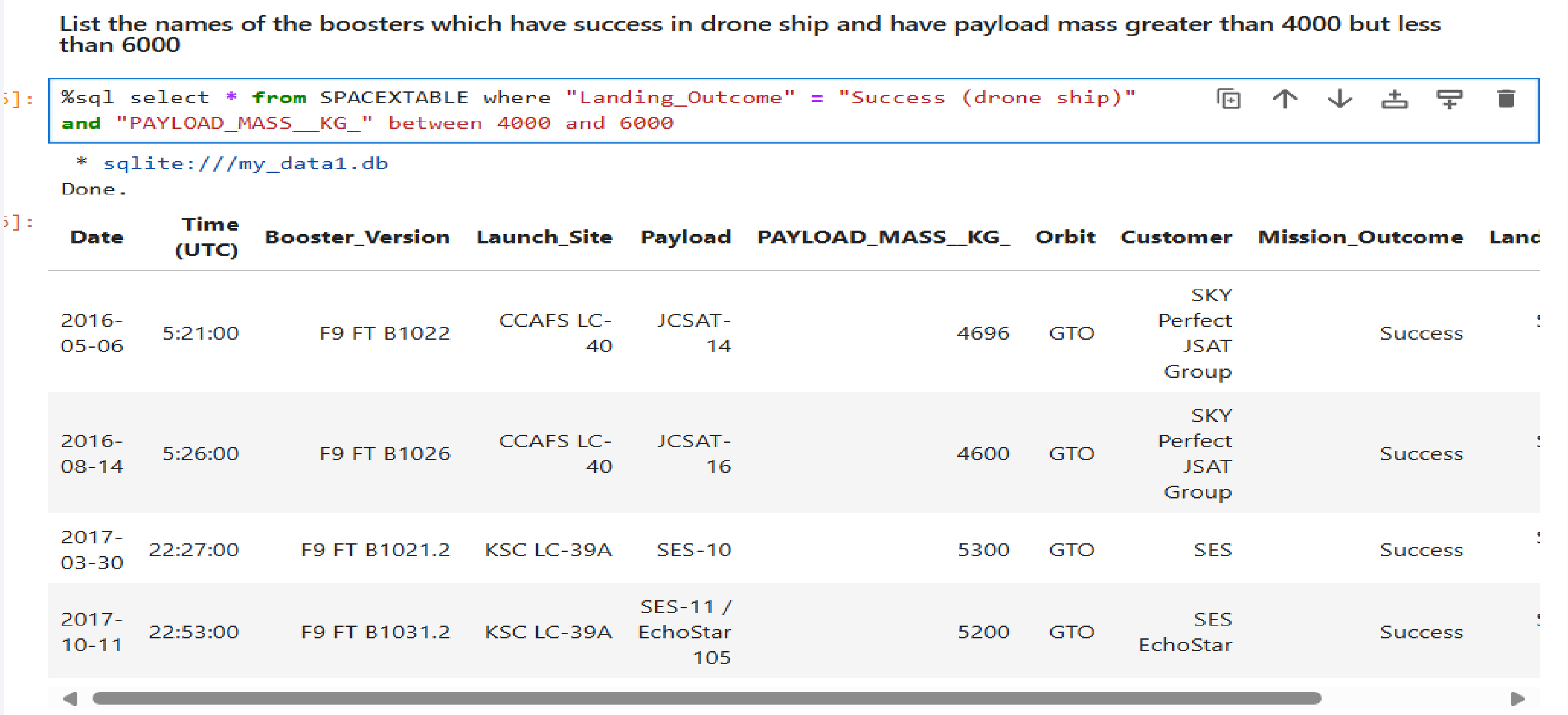
```
%sql select min(Date) from SPACEXTABLE where "Landing_Outcome" = "Success (ground pad)"
```

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

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



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

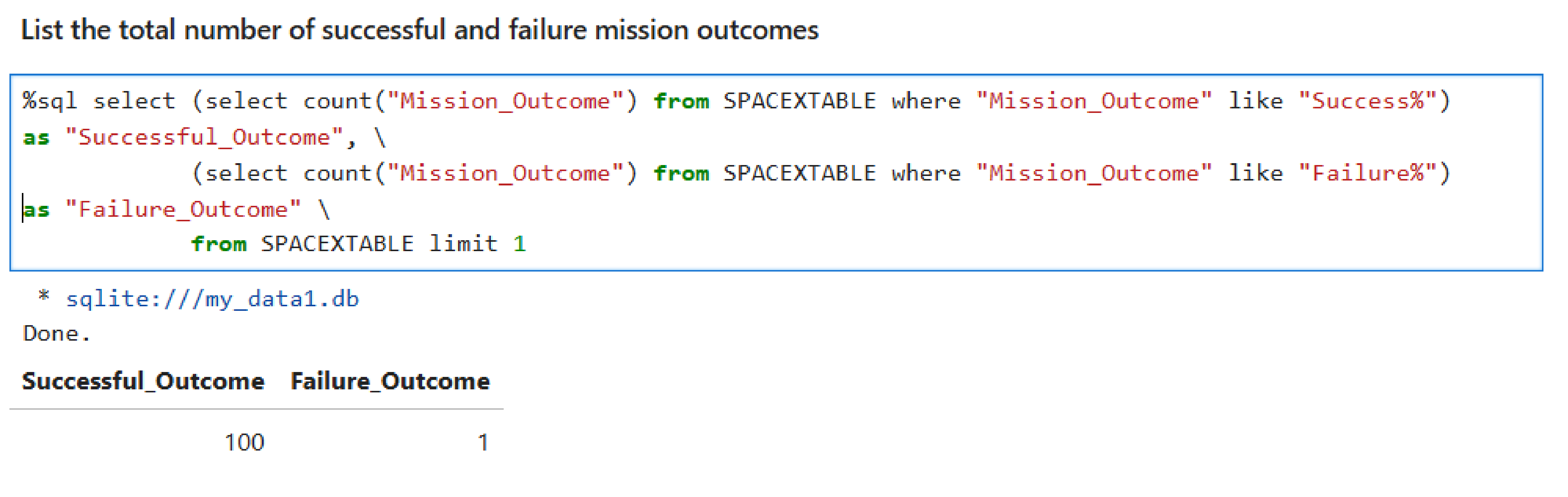
```
%sql select * from SPACEXTABLE where "Landing_Outcome" = "Success (drone ship)"
and "PAYLOAD_MASS__KG_" between 4000 and 6000
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Land
2016-05-06	5:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	
2016-08-14	5:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	
2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	
2017-10-11	22:53:00	F9 FT B1031.2	KSC LC-39A	SES-11 / EchoStar 105	5200	GTO	SES EchoStar	Success	

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes



List the total number of successful and failure mission outcomes

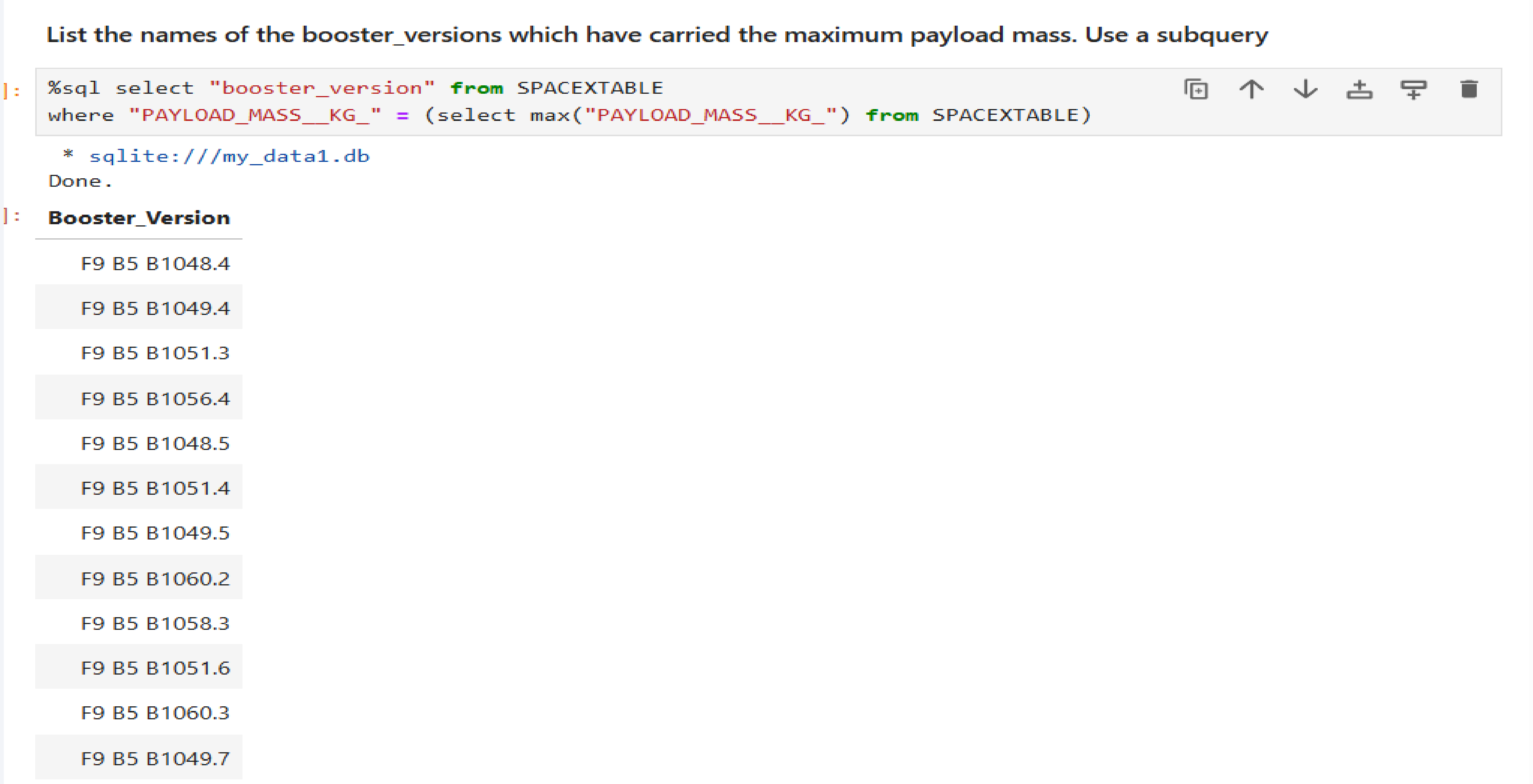
```
%sql select (select count("Mission_Outcome") from SPACEXTABLE where "Mission_Outcome" like "Success%")
as "Successful_Outcome", \
            (select count("Mission_Outcome") from SPACEXTABLE where "Mission_Outcome" like "Failure%")
as "Failure_Outcome" \
            from SPACEXTABLE limit 1
```

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

Successful_Outcome	Failure_Outcome
100	1

Boosters Carried Maximum Payload

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



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

```
]: %sql select "booster_version" from SPACEXTABLE
   where "PAYLOAD_MASS__KG_" = (select max("PAYLOAD_MASS__KG_") from SPACEXTABLE)
```

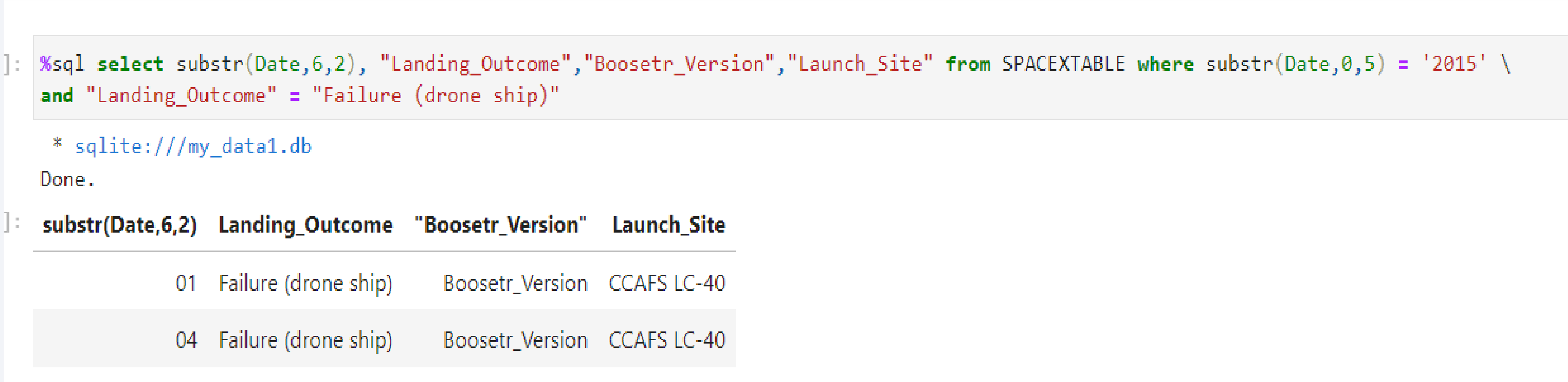
```
 * sqlite:///my_data1.db
Done.
```

]:

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



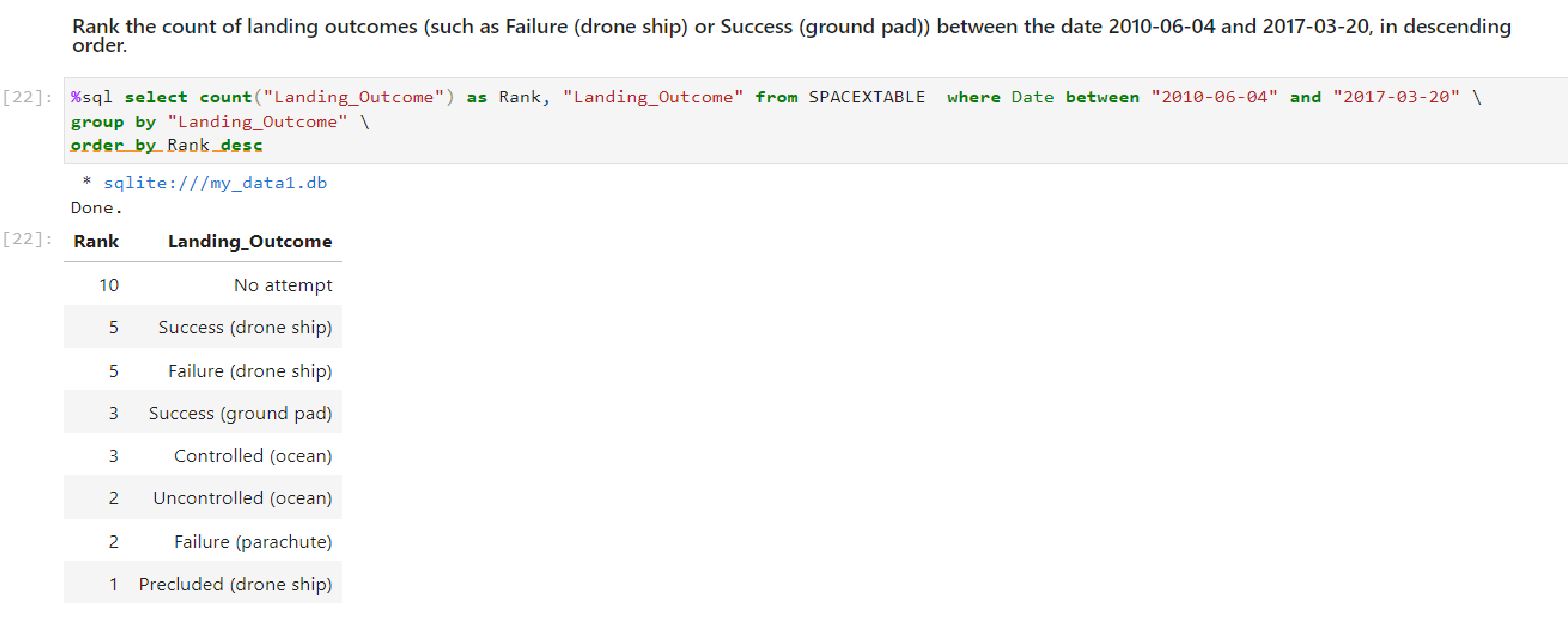
```
%sql select substr(Date,6,2), "Landing_Outcome","Boosetr_Version","Launch_Site" from SPACEXTABLE where substr(Date,0,5) = '2015' \
and "Landing_Outcome" = "Failure (drone ship)"
```

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

substr(Date,6,2)	Landing_Outcome	"Boosetr_Version"	Launch_Site
01	Failure (drone ship)	Boosetr_Version	CCAFS LC-40
04	Failure (drone ship)	Boosetr_Version	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



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.

```
[22]: %sql select count("Landing_Outcome") as Rank, "Landing_Outcome" from SPACEXTABLE  where Date between "2010-06-04" and "2017-03-20" \
      group by "Landing_Outcome" \
      order by Rank desc
```

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

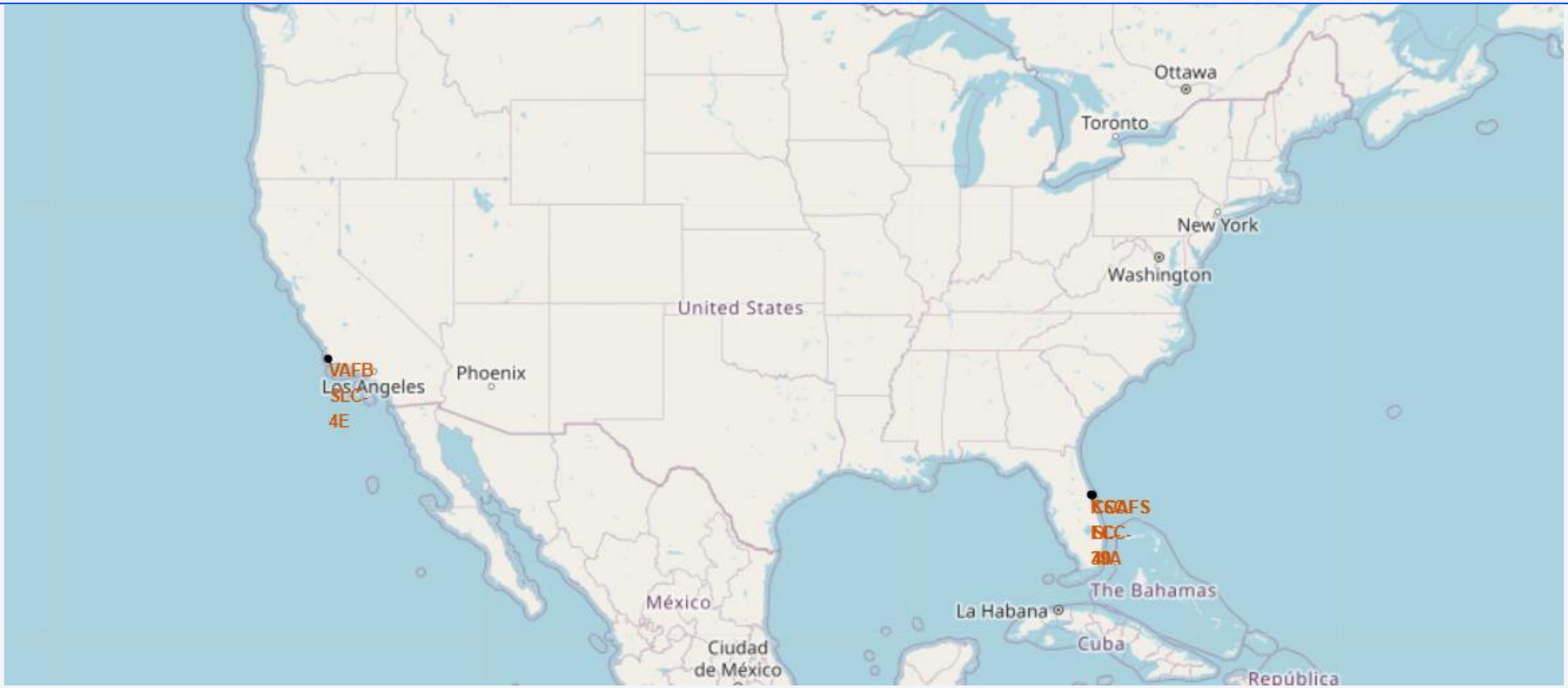
[22]:

Rank	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

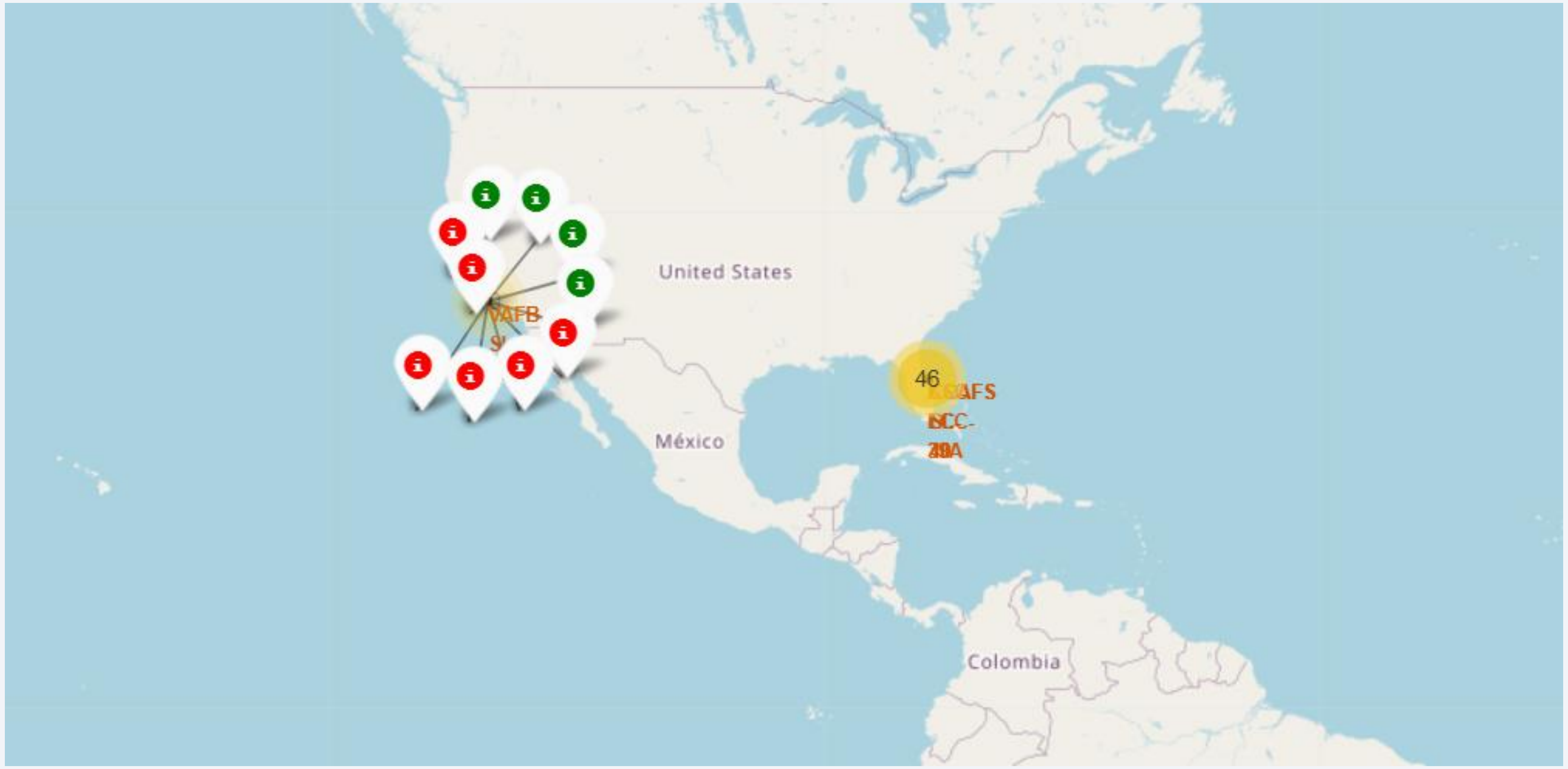
SpaceX Launch site Locations on Map



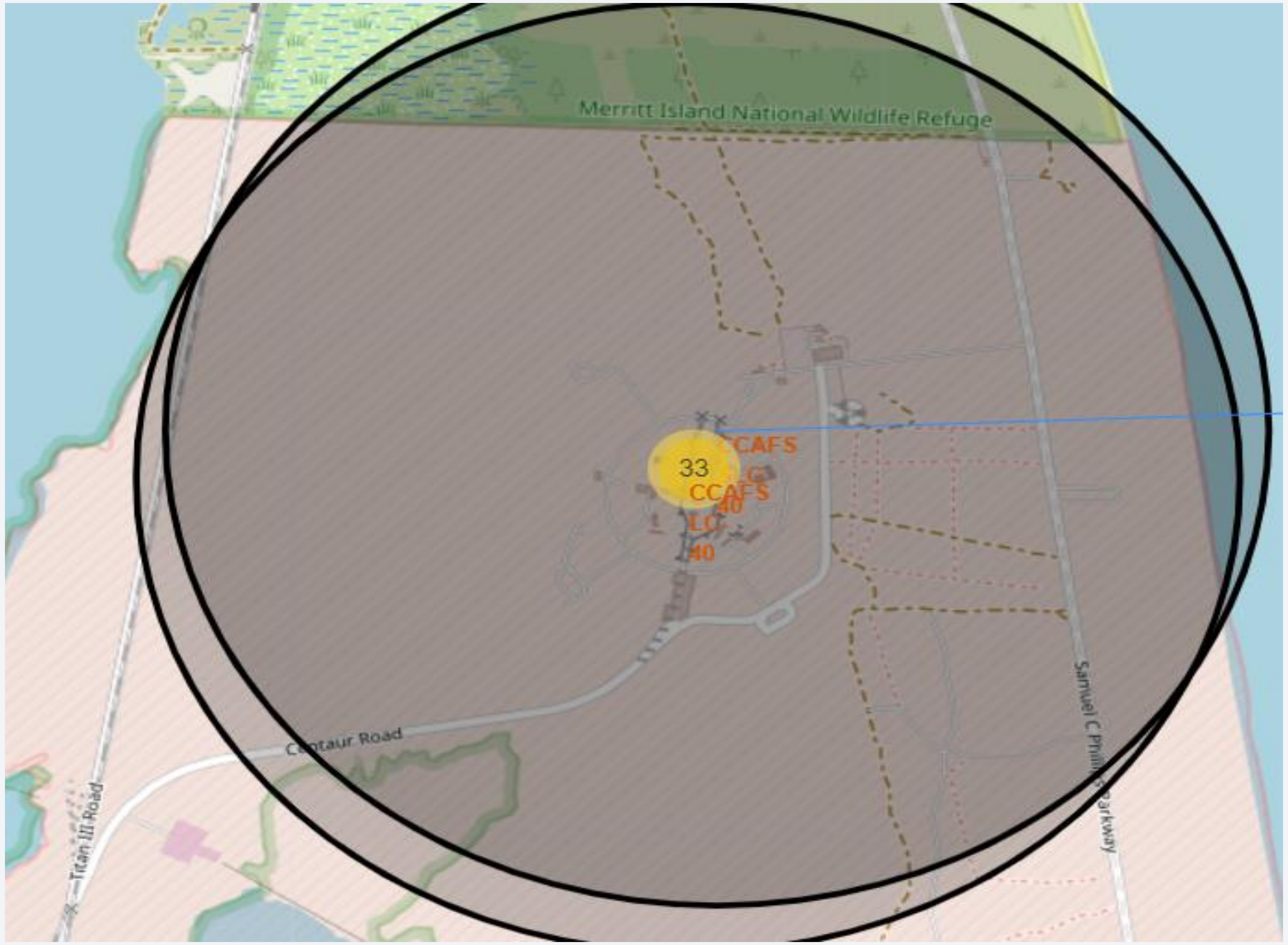
All launch sites in very close proximity to the coast

Launch Outcome of Launching Site

We can identify which Launch site has high success rate by clicking on the marker

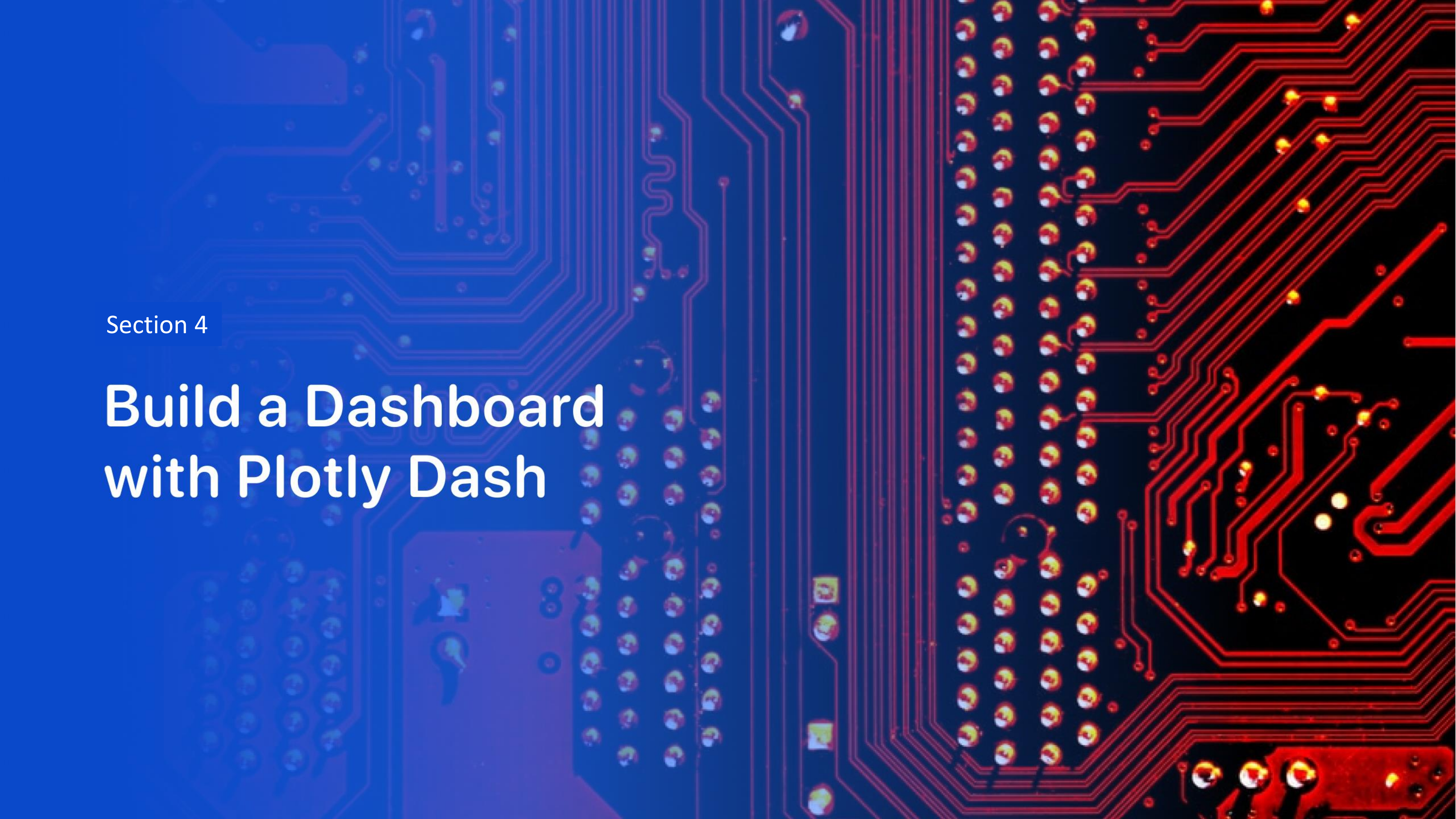


Distance From Coastline

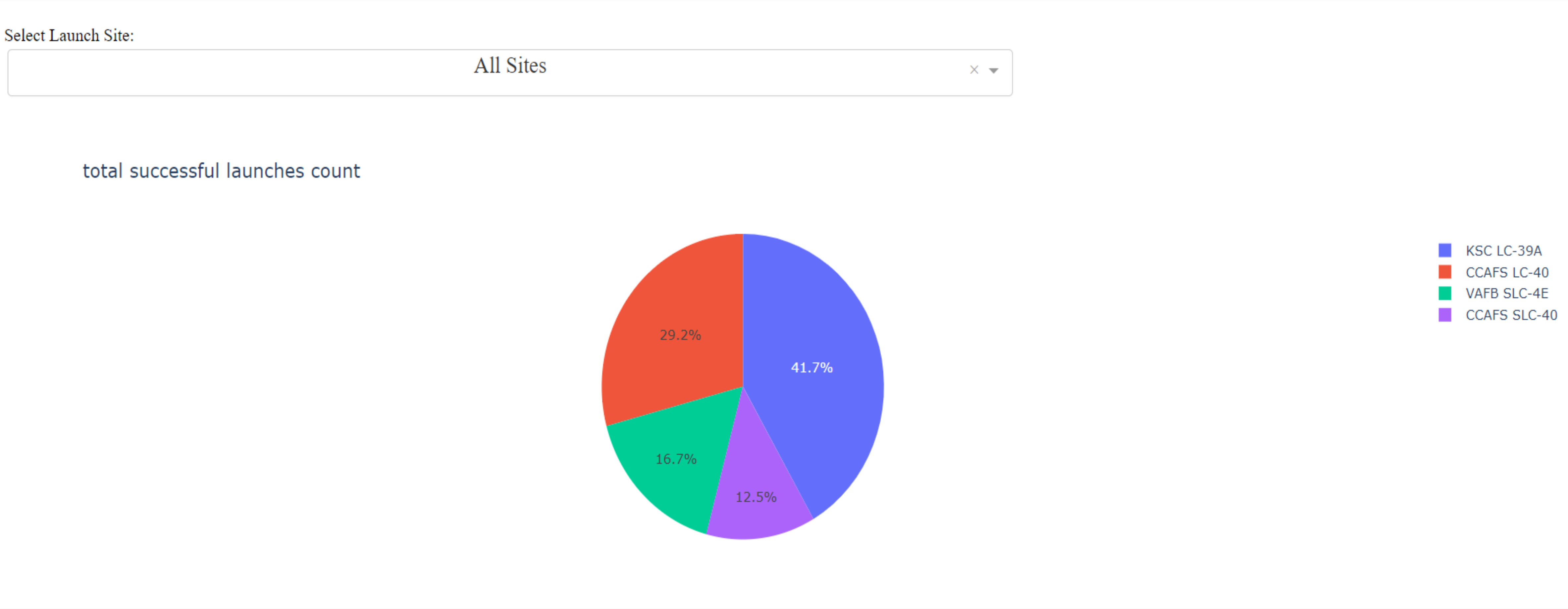

Merritt Island National Wildlife Refuge
33
CCAFS
CCAFS
LC
40
LC
40
Centaur Road
Titan III Road
Samuel C Phillips Parkway

Section 4

Build a Dashboard with Plotly Dash



Success Rate of all Launch Site

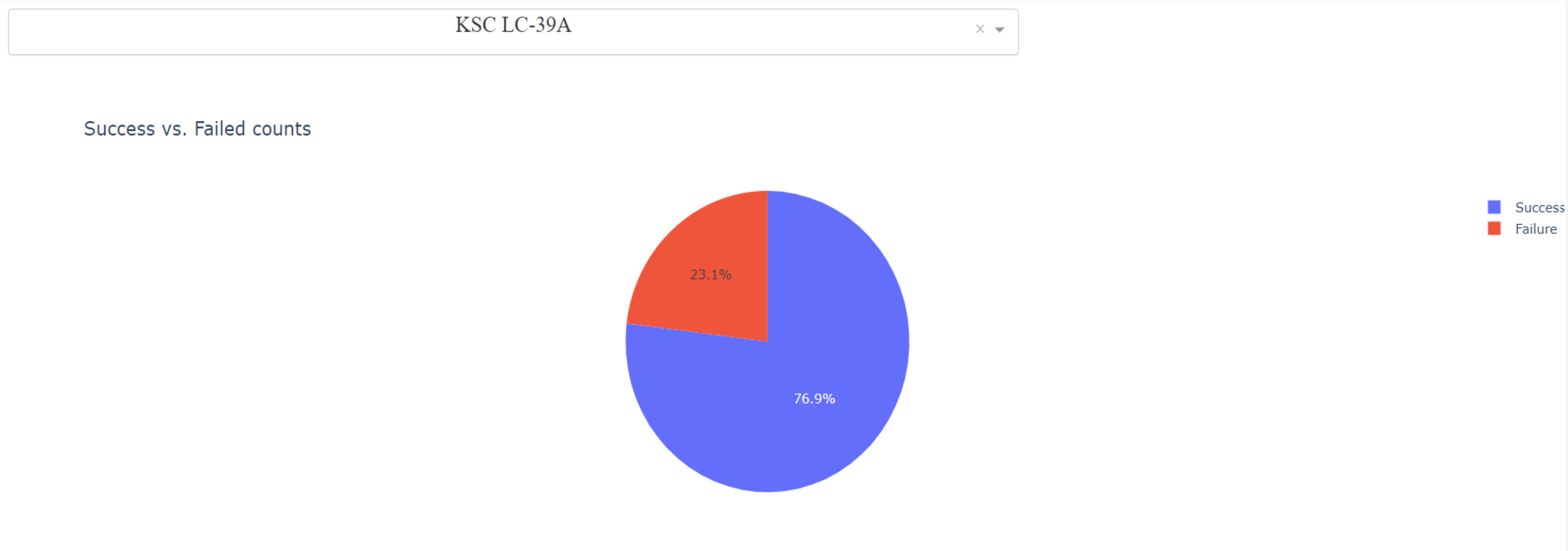


Select Launch Site:

All Sites

KSC LC – 39A has the highest success rate and CCAFS SLC-40 has the least success rate of Landing

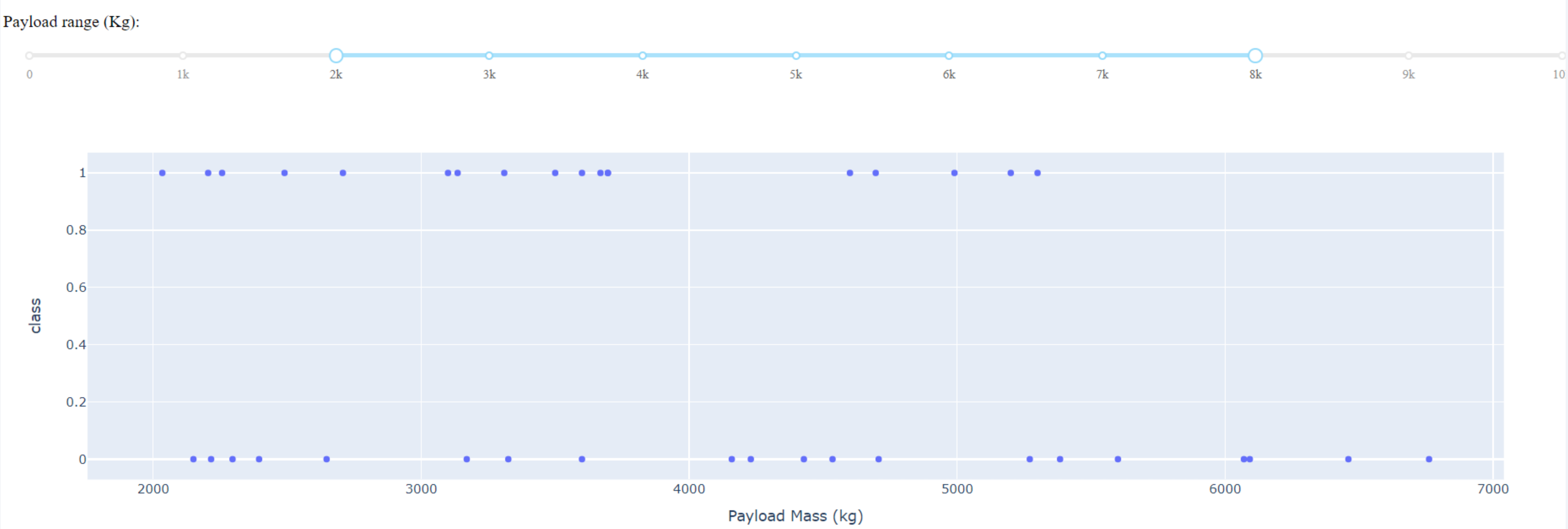
Landing outcome for KSC LC – 39A



KSC LC-39A × ▾

- KSC LC has 76.9% of success rate

Payload vs. Launch Outcome scatter plot

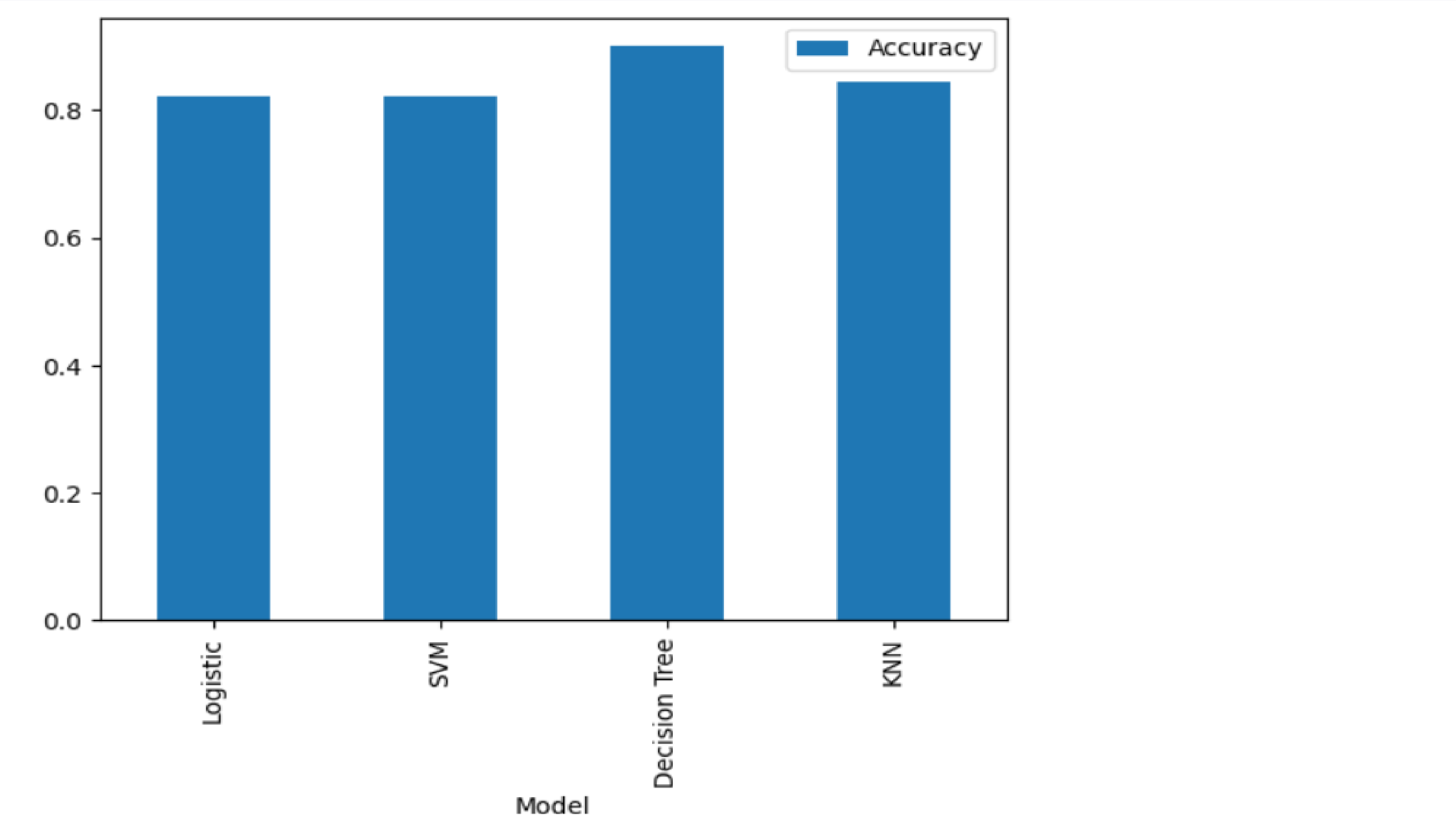


Payload range (Kg):

Section 5

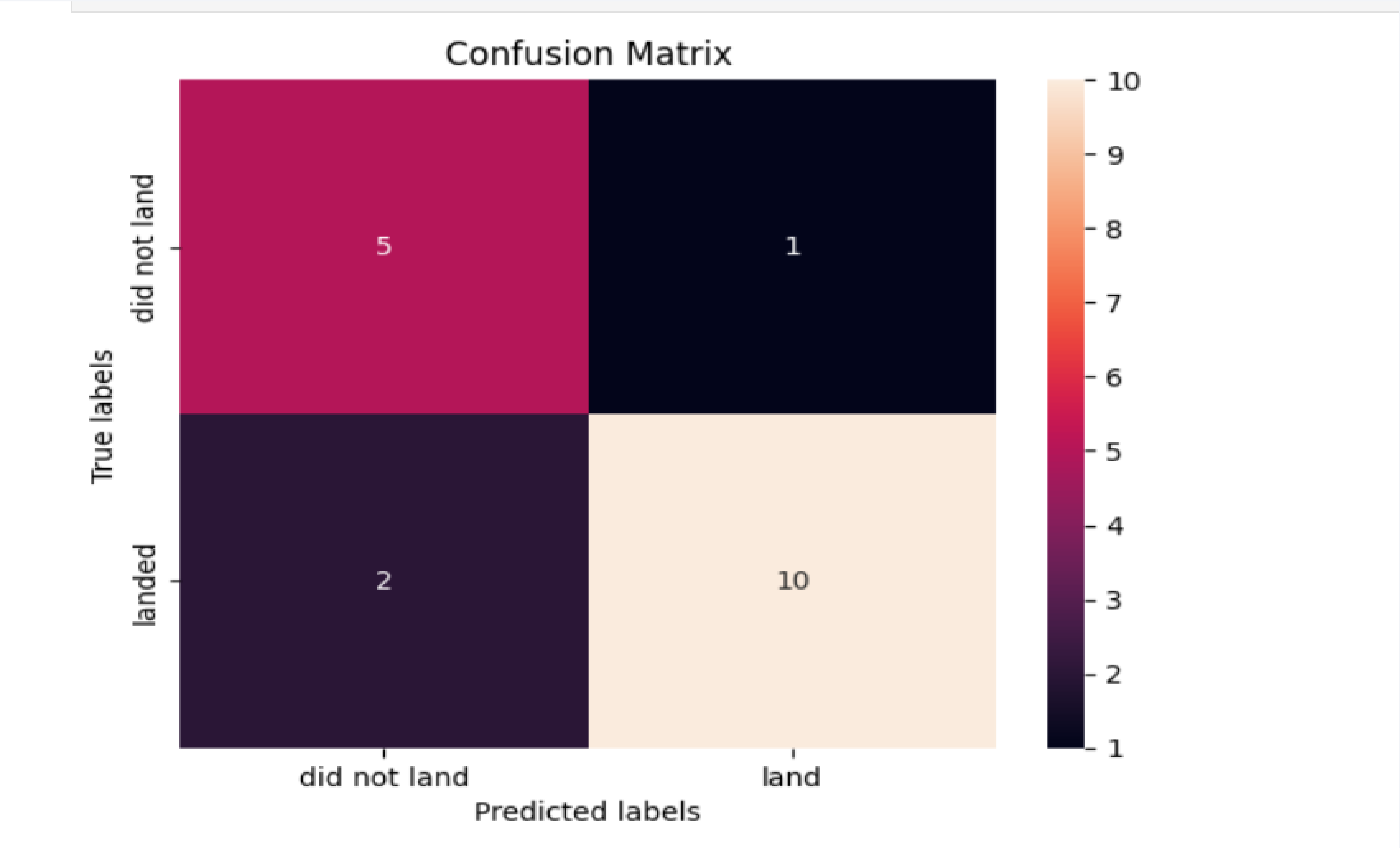
Predictive Analysis (Classification)

Classification Accuracy



- Decision Tree has highest Accuracy among others

Confusion Matrix



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!