IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Qi Wang>
<14/03/2022>

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with Data Visualization
  - Exploratory Data Analysis with SQL
  - Interactive Visual Analytics with Folium
  - Predictive Analysis with Machine Learning
- Summary of all results
  - Exploratory Data Analysis
  - Interactive Analytics in Screenshots
  - Predictive Analytics Results

Introduction

- Project background and context

Space X 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 Space X 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 space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers

- What factors determine if the rocket will land successfully?

- The interaction amongst various features that determine the success rate of a successful landing.

- What operating conditions needs to be fullfilled to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- The data was collected using various methods

- Data collection was done using GET request to the SpaceX API.

- Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().

- We then cleaned the data, checked for missing values and fill in missing values where necessary.

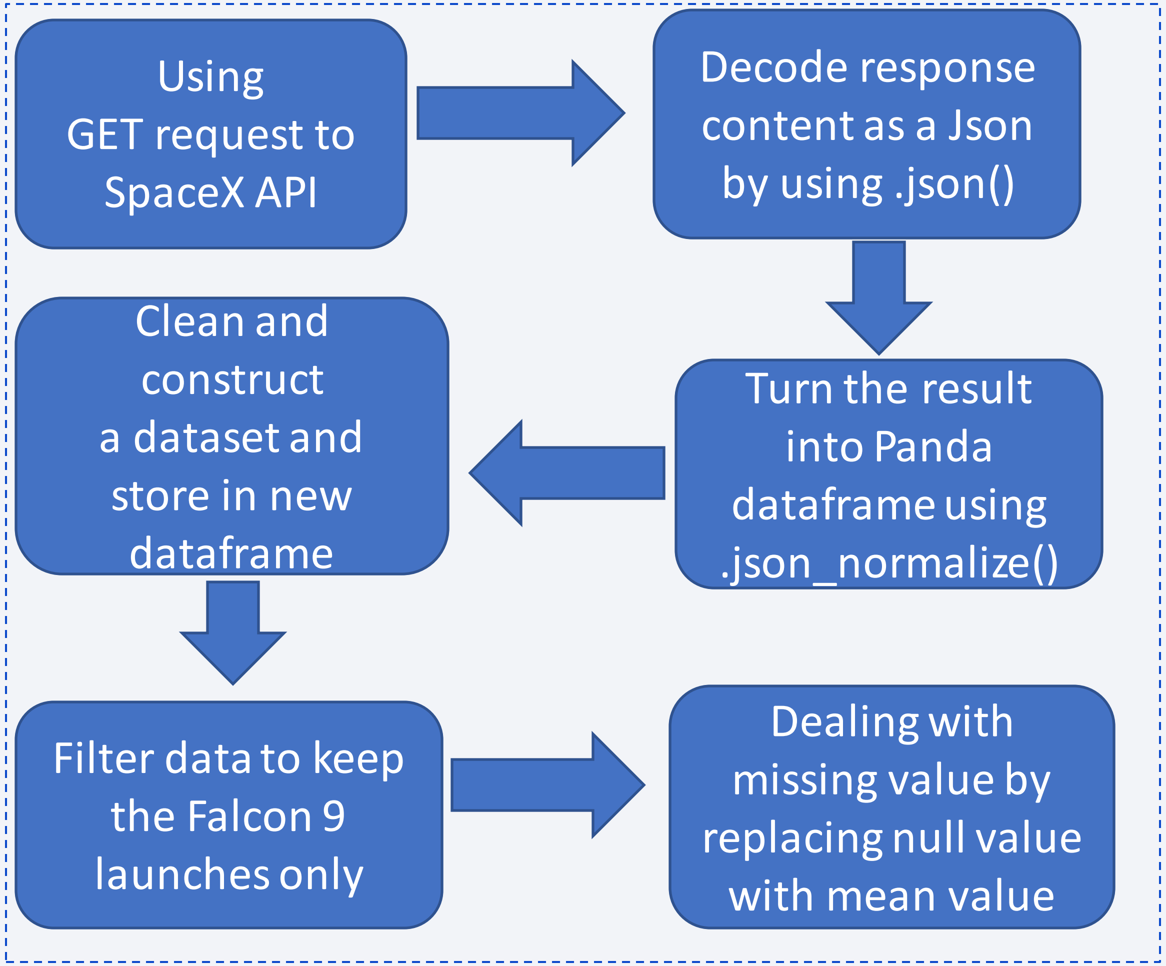
- In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.

- The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection – SpaceX API

- We used the GET request to the SpaceX API to collect data, clean the requested data and did some data wrangling and formatting, finally store it in a data frame.

- The link to the notebook is: https://github.com/says4yeah/finkont/blob/master/Data%20Collection%20with%20Web%20Scraping.ipynb



Using GET request to SpaceX API

Decode response content as a Json by using .json()

Turn the result into Panda dataframe using .json_normalize()

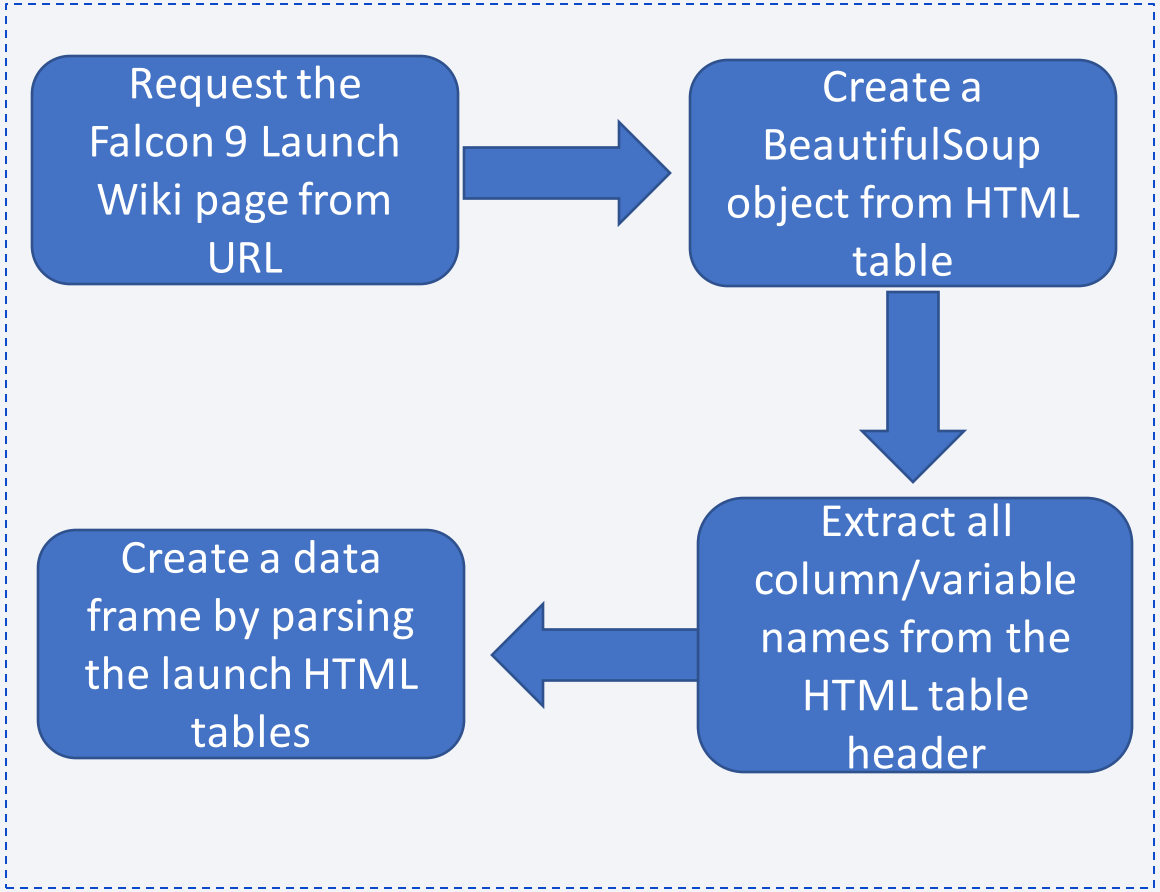
Clean and construct a dataset and store in new dataframe

Filter data to keep the Falcon 9 launches only

Dealing with missing value by replacing null value with mean value

Data Collection - Scraping

- We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.
- The link to the notebook is: https://github.com/says4yeah/finkont/blob/master/Data%20Collection%20with%20Web%20Scraping.ipynb


Request the Falcon 9 Launch Wiki page from URL
Create a BeautifulSoup object from HTML table
Extract all column/variable names from the HTML table header
Create a data frame by parsing the launch HTML tables

Data Wrangling

- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits.
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is: https://github.com/says4yeah/finkont/blob/master/Exploratory%20Data%20Analysis%20for%20Data%20Visualization.ipynb

EDA with Data Visualization

- We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.
- The link to the notebook is: https://github.com/says4yeah/finkont/blob/master/Exploratory%20Data%20Analysis%20for%20Data%20Visualization.ipynb

EDA with SQL

- We loaded the SpaceX dataset into a PstgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:

  - The name of unique launch sites in the space mission.

  - The total payload mass carried by boosters launched by NASA (CRS)

  - The average payload mass carried by booster version F9 V1.1

  - The total number of successful and failure mission outcomes

  - The failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook is:

https://github.com/says4yeah/finkont/blob/master/jupyter-labs-eda-sql-coursera.ipynb

Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1.

  i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rates.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:

  - Are launch sites near railways, highways and coastlines.

  - Do launch sites keep certain distance away from cities.

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- The link to the notebook is: https://github.com/says4yeah/finkont/blob/master/spacex_dash_app.py

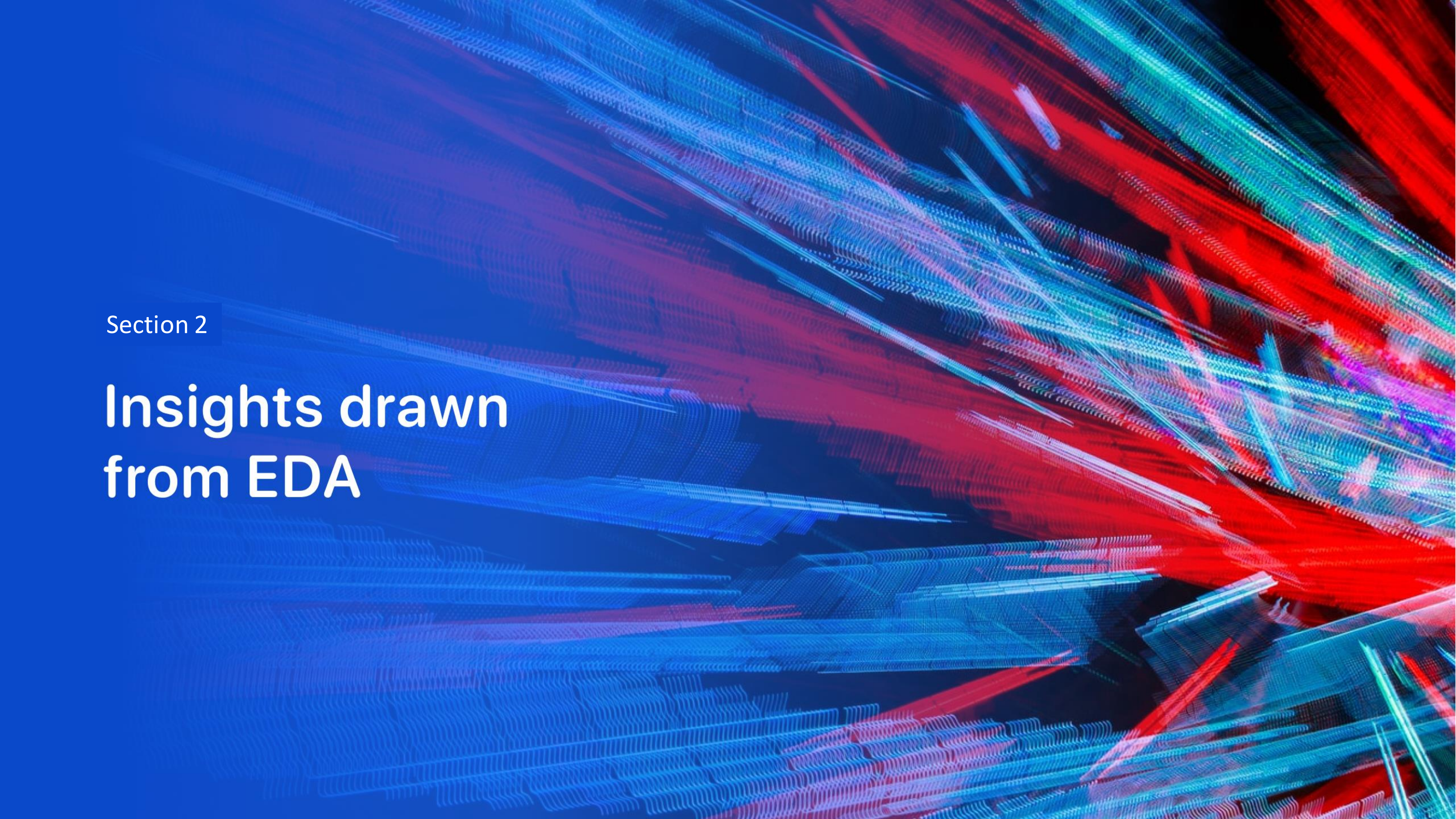
Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuacy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- The link to the notebook is:

https://github.com/says4yeah/finkont/blob/master/Machine%20Learning%20Prediction.ipynb

Results

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

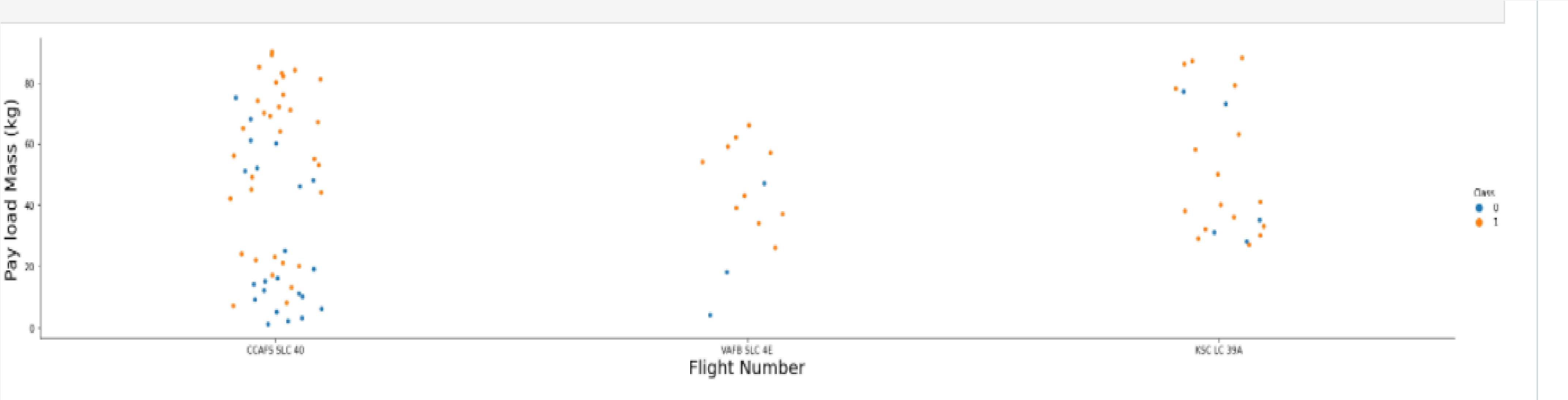


Section 2

Insights drawn from EDA

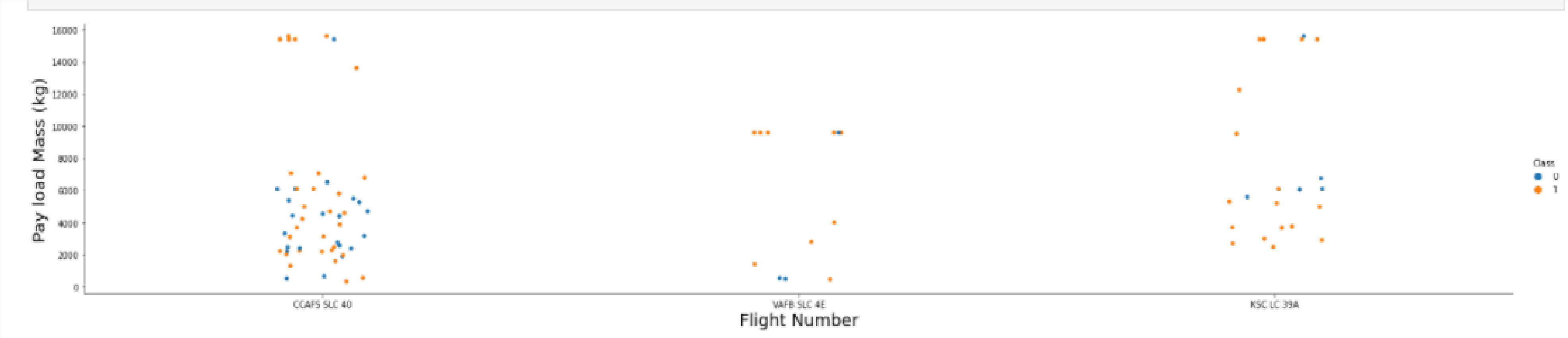
Flight Number vs. Launch Site

- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.



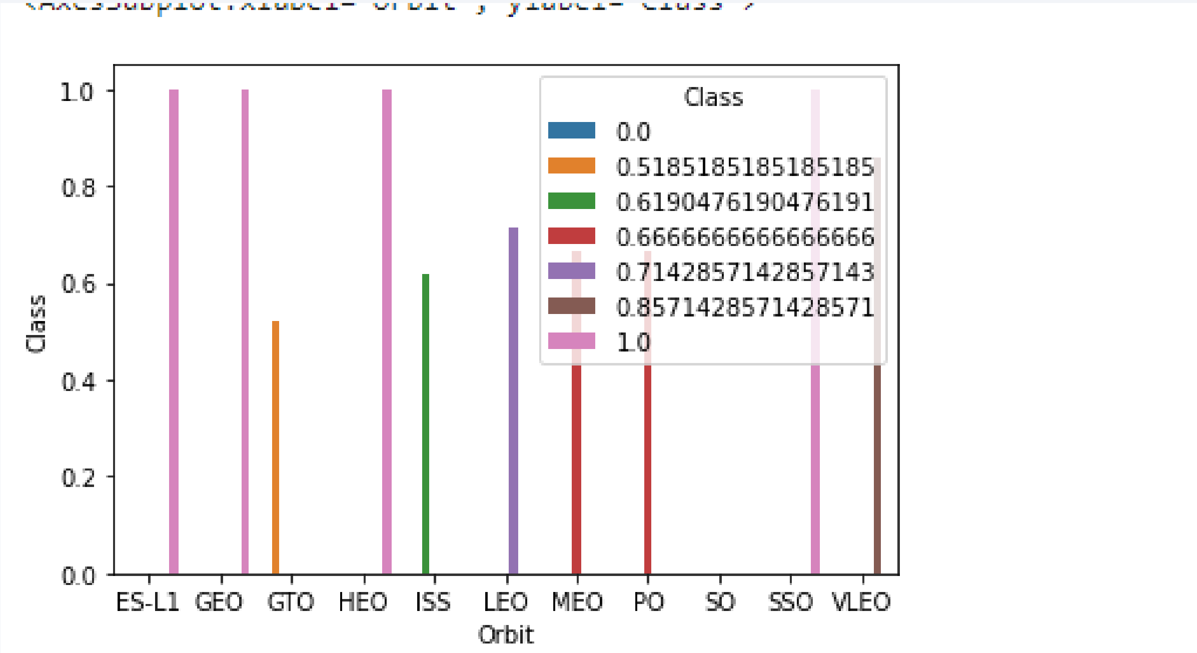
Payload vs. Launch Site

- The greater the payload mass for launch site CCAFS SLC 40 the higher the success rate for the rocket.



Success Rate vs. Orbit Type

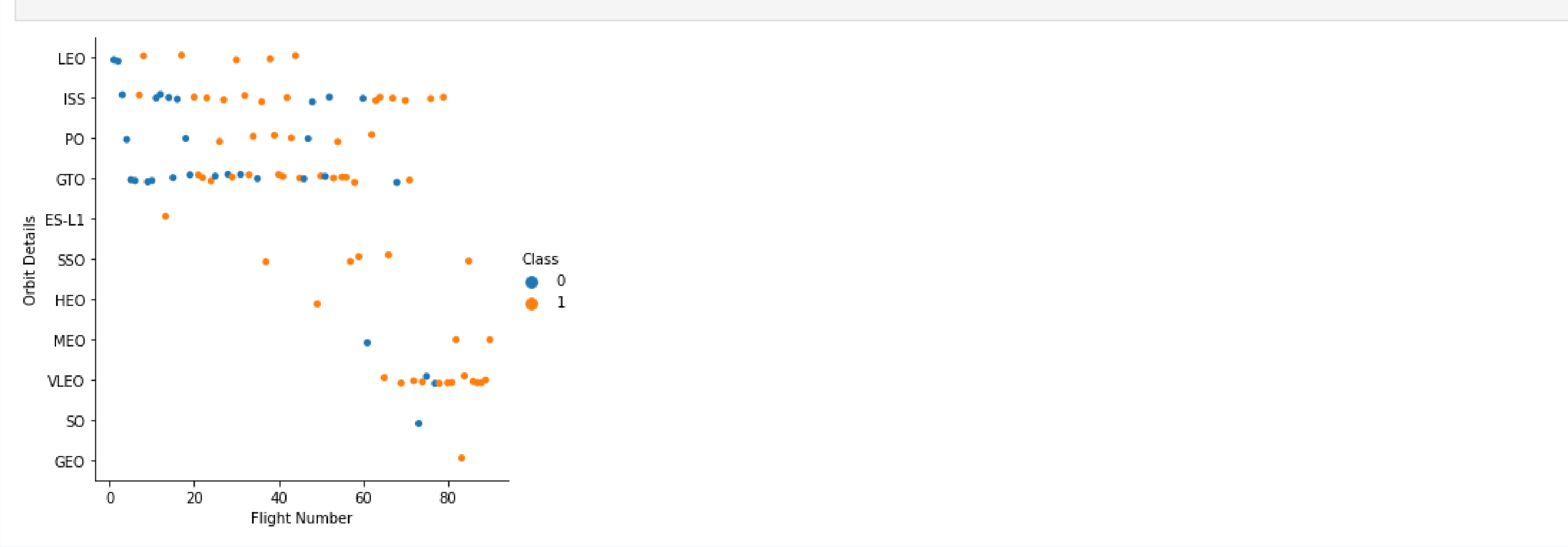
- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



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

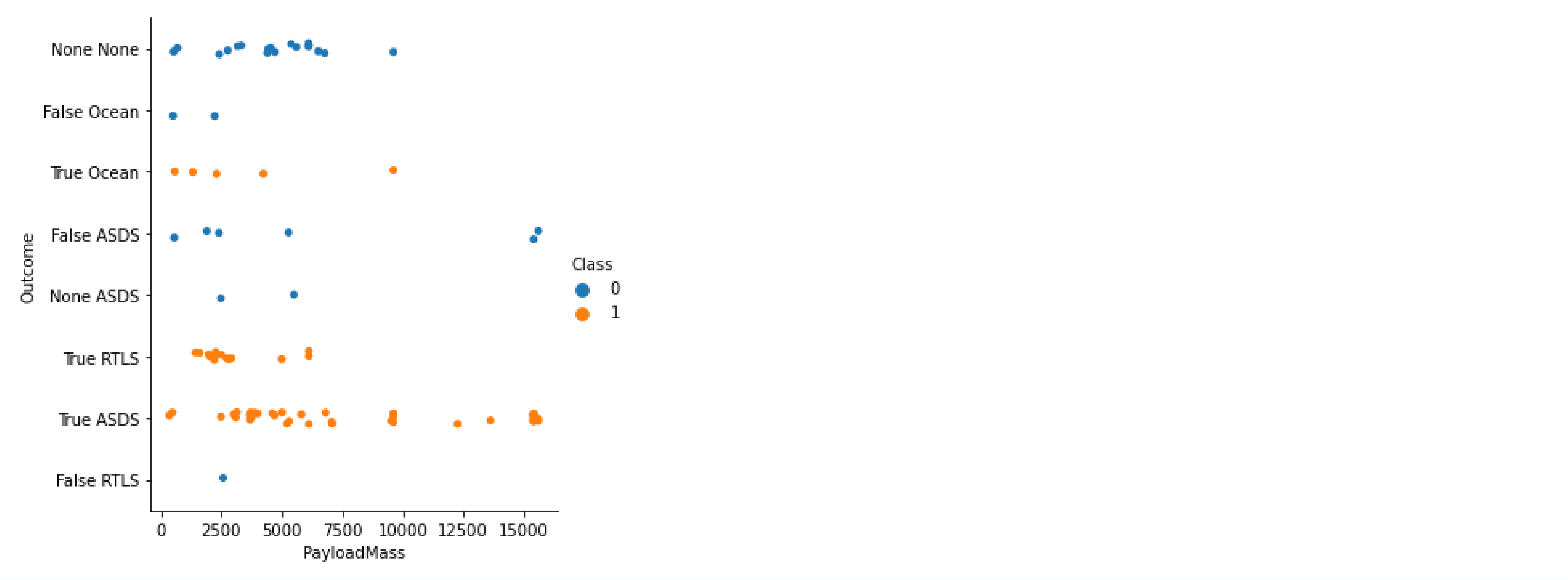
Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



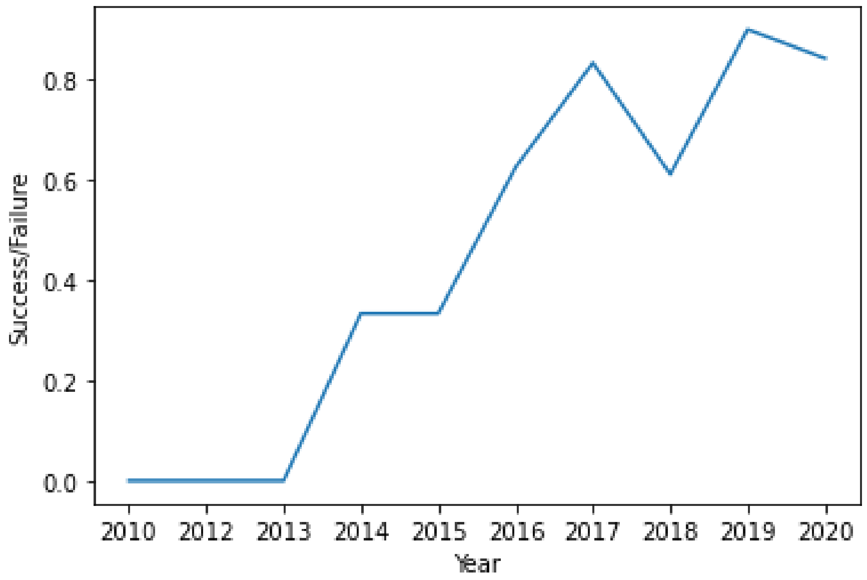
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



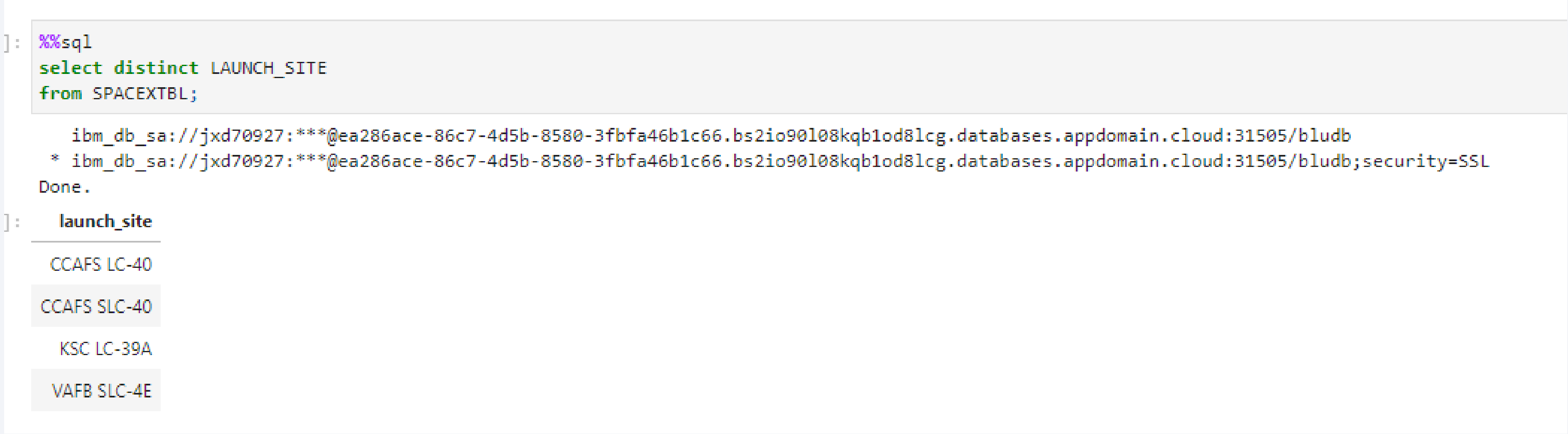
Launch Success Yearly Trend

- From the plot, we can observe that success rate since 2013 kept on increasing till 2020.



All Launch Site Names

- We used the key word DISTINCT to show only unique launch sites from the SpaceX data.



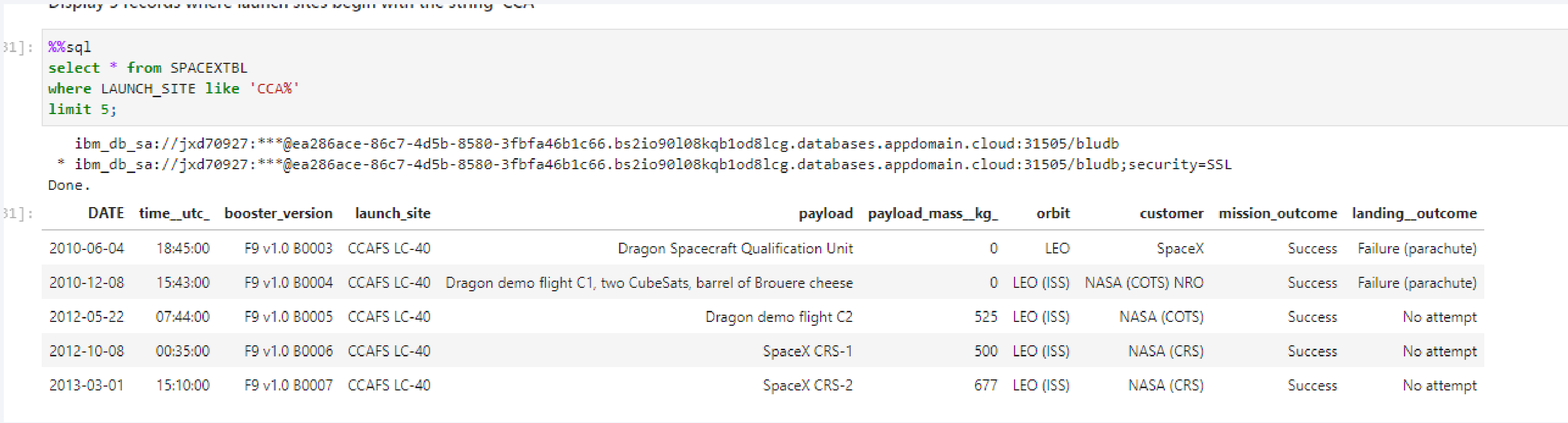
```
%%sql
select distinct LAUNCH_SITE
from SPACEXTBL;
```

```
   ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
 * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb;security=SSL
Done.
```

launch_site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

Launch Site Names Begin with 'CCA'

- We used the query above to display 5 records where launch sites begin with 'CCA'



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

```
31]: %%sql
     select * from SPACEXTBL
     where LAUNCH_SITE like 'CCA%'
     limit 5;
```

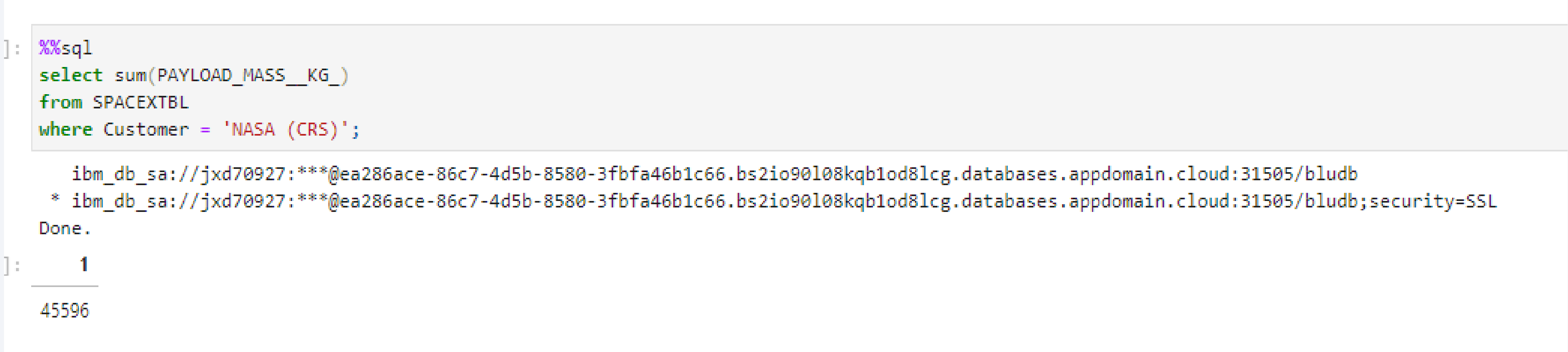
```
   ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
 * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb;security=SSL
Done.
```

31]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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 (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- We calculated the total payload carried by boosters from NASA as 45596 using the query below



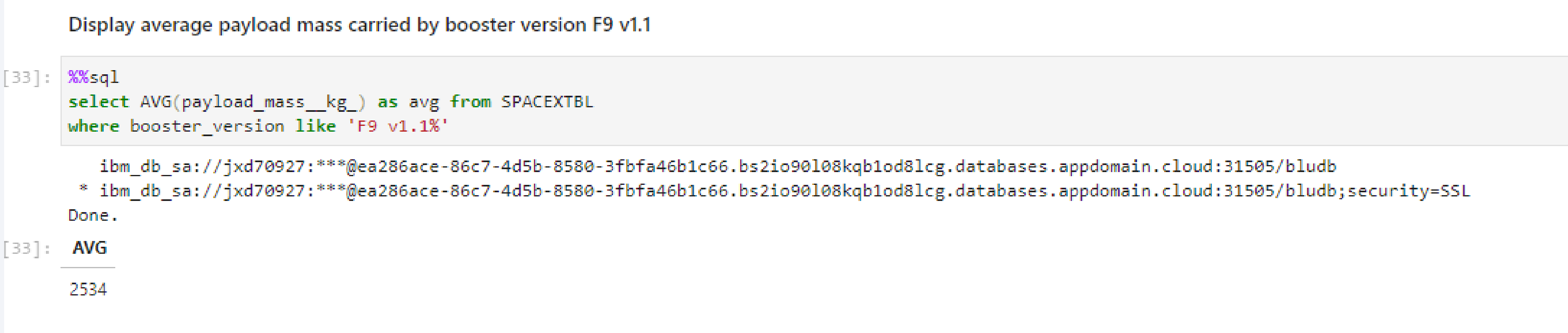
```
]: %%sql
   select sum(PAYLOAD_MASS__KG_)
   from SPACEXTBL
   where Customer = 'NASA (CRS)';
```

```
   ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
 * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb;security=SSL
Done.
```

```
]:    1
   -----
   45596
```

Average Payload Mass by F9 v1.1

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



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

```
[33]: %%sql
      select AVG(payload_mass__kg_) as avg from SPACEXTBL
      where booster_version like 'F9 v1.1%'
```

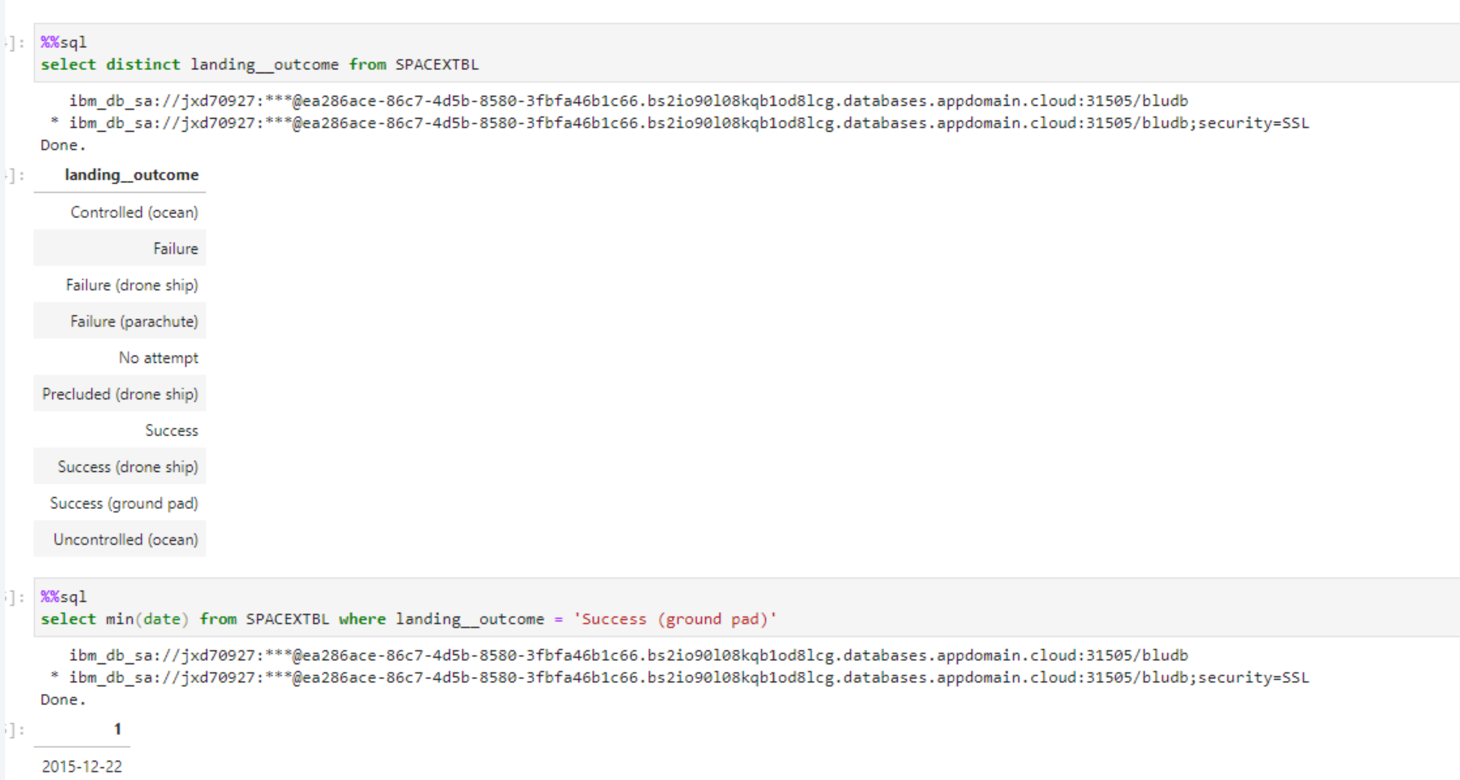
```
   ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
 * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb;security=SSL
Done.
```

[33]:

AVG
2534

First Successful Ground Landing Date

- We used DISTINCT to find the right value representing successful ground landing and then used MIN-function found the dates of the first successful landig outcome on ground pad



```
%%sql
select distinct landing__outcome from SPACEXTBL
```

```
   ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
 * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb;security=SSL
Done.
```

landing__outcome
Controlled (ocean)
Failure
Failure (drone ship)
Failure (parachute)
No attempt
Precluded (drone ship)
Success
Success (drone ship)
Success (ground pad)
Uncontrolled (ocean)

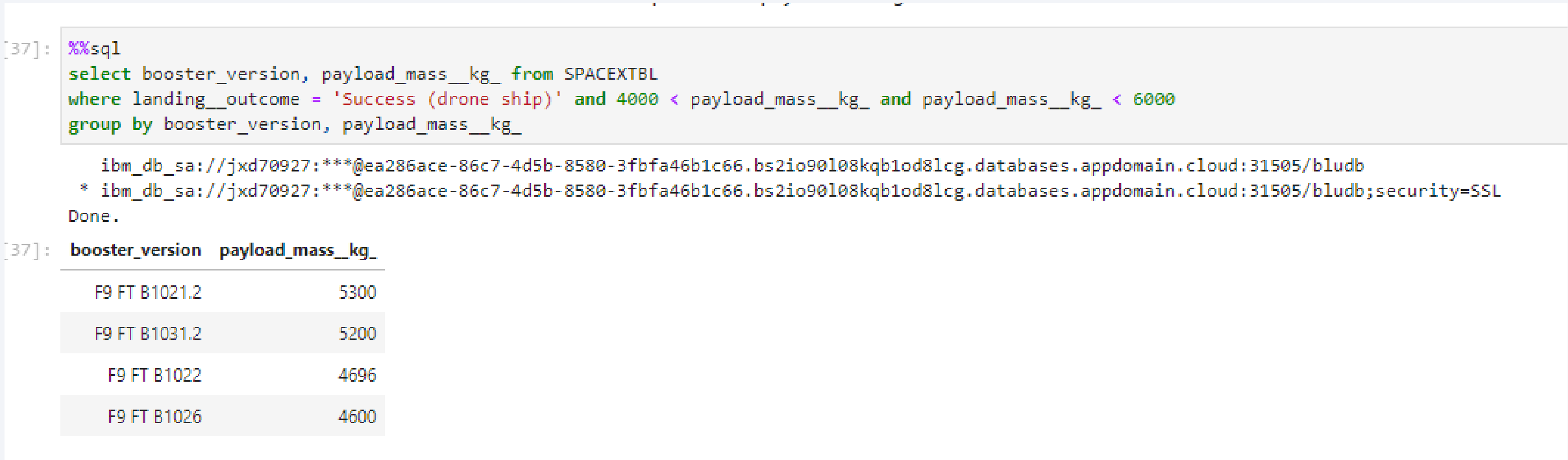
```
%%sql
select min(date) from SPACEXTBL where landing__outcome = 'Success (ground pad)'
```

```
   ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
 * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb;security=SSL
Done.
```

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the WHERE clause to filter for boosters which have successfully landed on drone ship and applied AND condition to determine successful landing with payload mass greater than 4000 but less than 6000



[37]:
```
%%sql
select booster_version, payload_mass__kg_ from SPACEXTBL
where landing__outcome = 'Success (drone ship)' and 4000 < payload_mass__kg_ and payload_mass__kg_ < 6000
group by booster_version, payload_mass__kg_
```

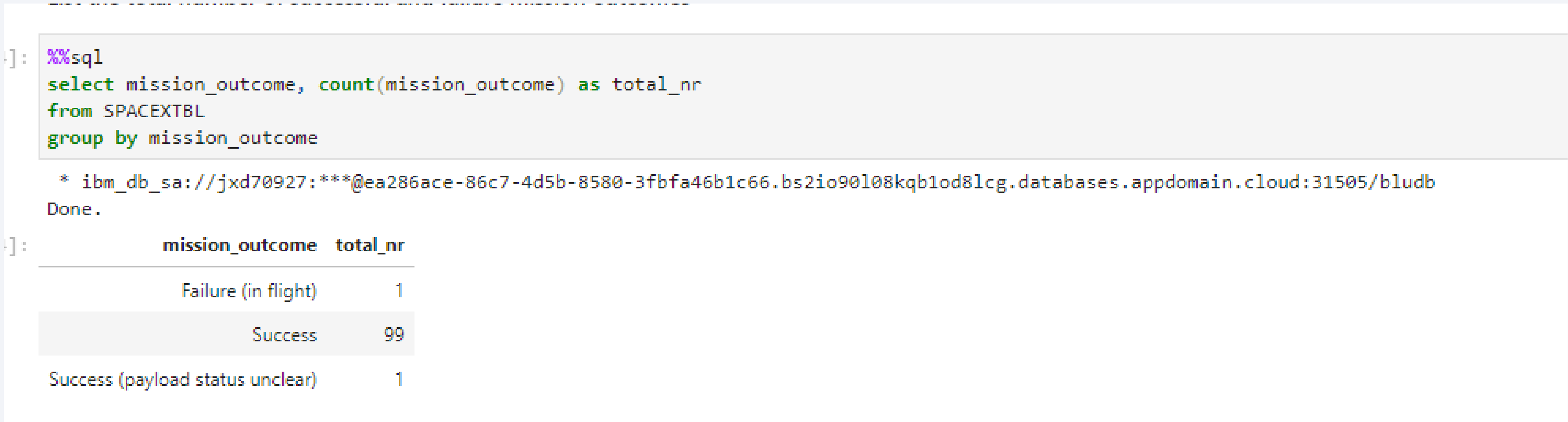
```
   ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
 * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb;security=SSL
Done.
```

[37]:

booster_version	payload_mass__kg_
F9 FT B1021.2	5300
F9 FT B1031.2	5200
F9 FT B1022	4696
F9 FT B1026	4600

Total Number of Successful and Failure Mission Outcomes

- We used wildcard like '%' to filter for WHER Mission Outcome was a success or a failure.



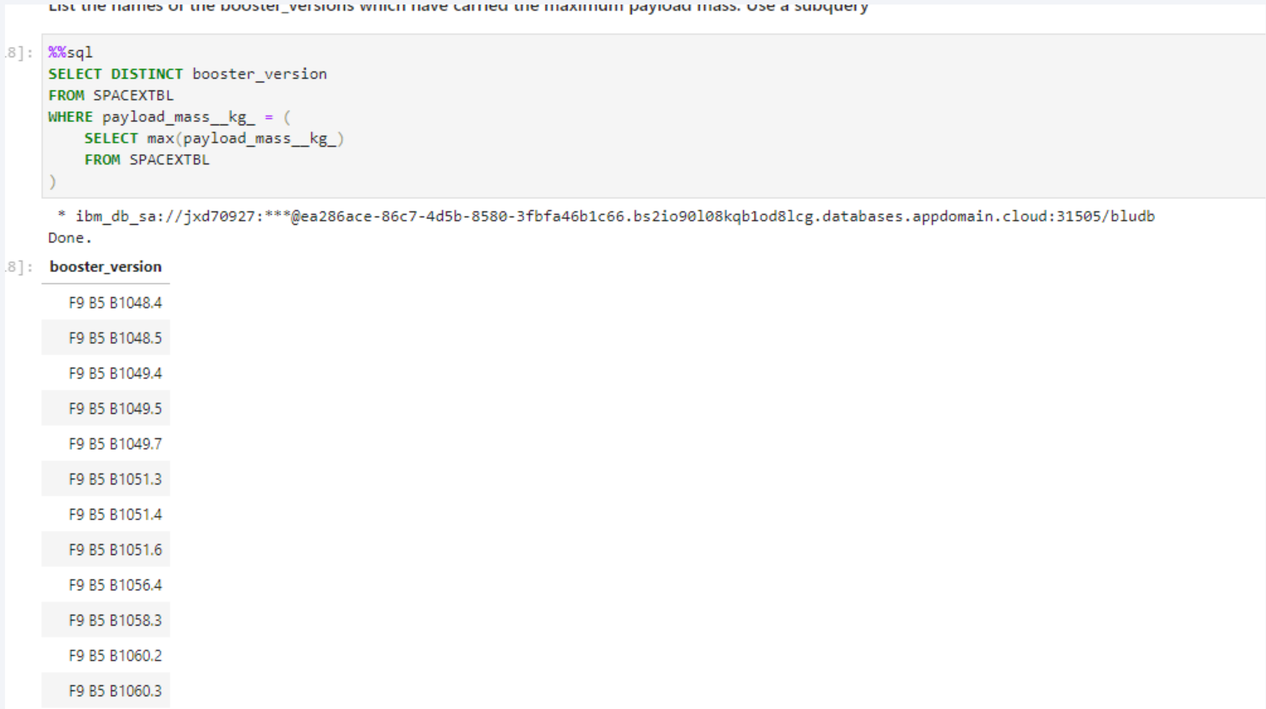
```
%%sql
select mission_outcome, count(mission_outcome) as total_nr
from SPACEXTBL
group by mission_outcome
```

```
 * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

mission_outcome	total_nr
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- We determined the booster that have carried the maximum payload using a subquery in the WHERE clause and the MAX() function.



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

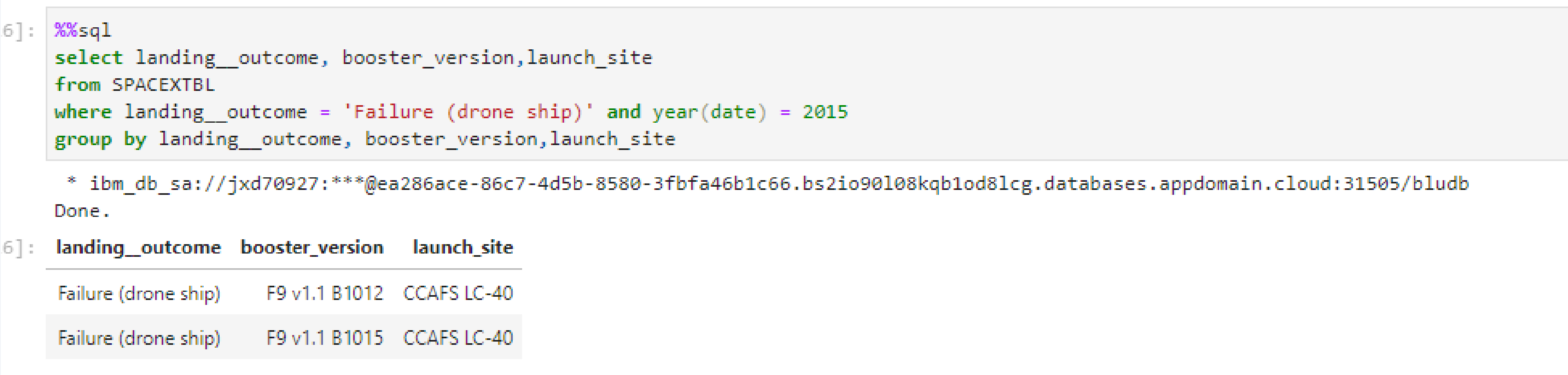
```
%%sql
SELECT DISTINCT booster_version
FROM SPACEXTBL
WHERE payload_mass__kg_ = (
    SELECT max(payload_mass__kg_)
    FROM SPACEXTBL
)
```

```
 * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

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

- We used a combinations of the WHERE clause, LIKE, AND, and BETWEEN conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015



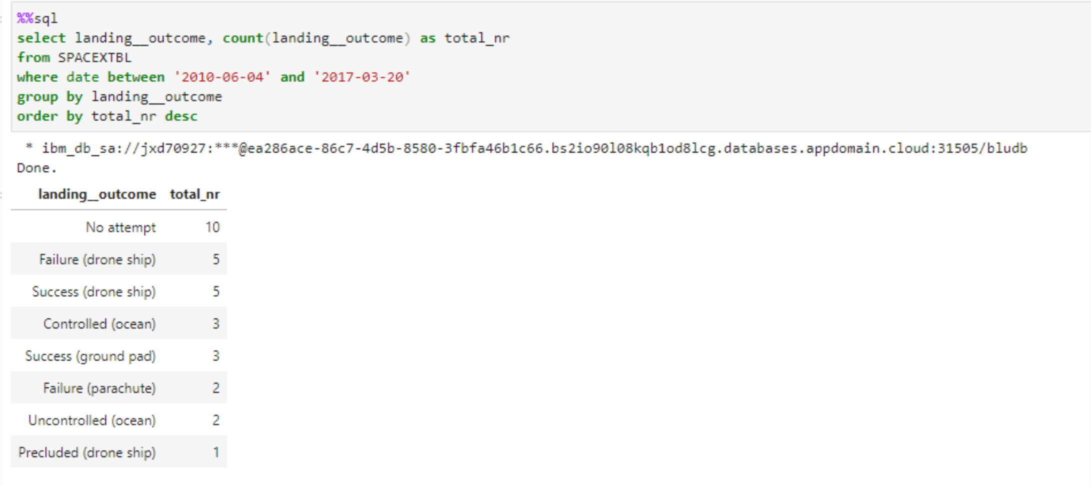
```
6]: %%sql
    select landing__outcome, booster_version,launch_site
    from SPACEXTBL
    where landing__outcome = 'Failure (drone ship)' and year(date) = 2015
    group by landing__outcome, booster_version,launch_site

     * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
    Done.
```

6]:

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

- We used GROUP BY and ORDER BY to rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between 2016-06-04 to 2010-03-20 in a descending order.



```
%%sql
select landing__outcome, count(landing__outcome) as total_nr
from SPACEXTBL
where date between '2010-06-04' and '2017-03-20'
group by landing__outcome
order by total_nr desc
```

```
 * ibm_db_sa://jxd70927:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

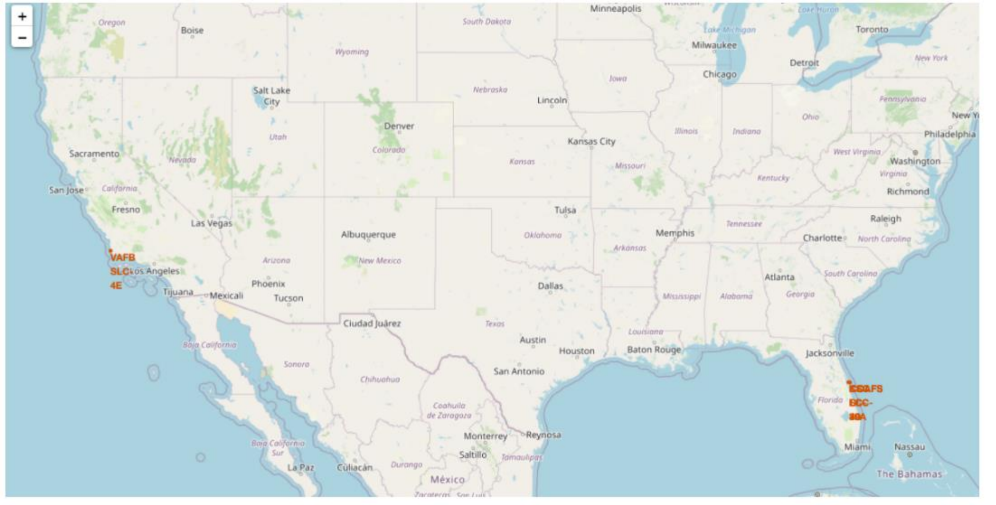
landing__outcome	total_nr
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

- 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

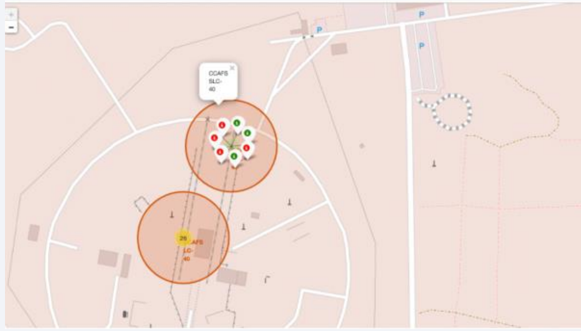


- Explain the important elements and findings on the screenshot

All SpaceX launch sites are in the US coasts, Florida and California

<Folium Map Screenshot 2>

- 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

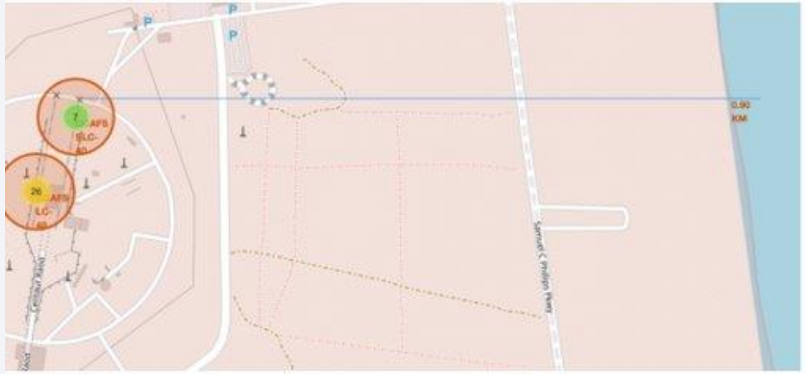


- Explain the important elements and findings on the screenshot

Green Marker: successful launches, Red Marker: failures

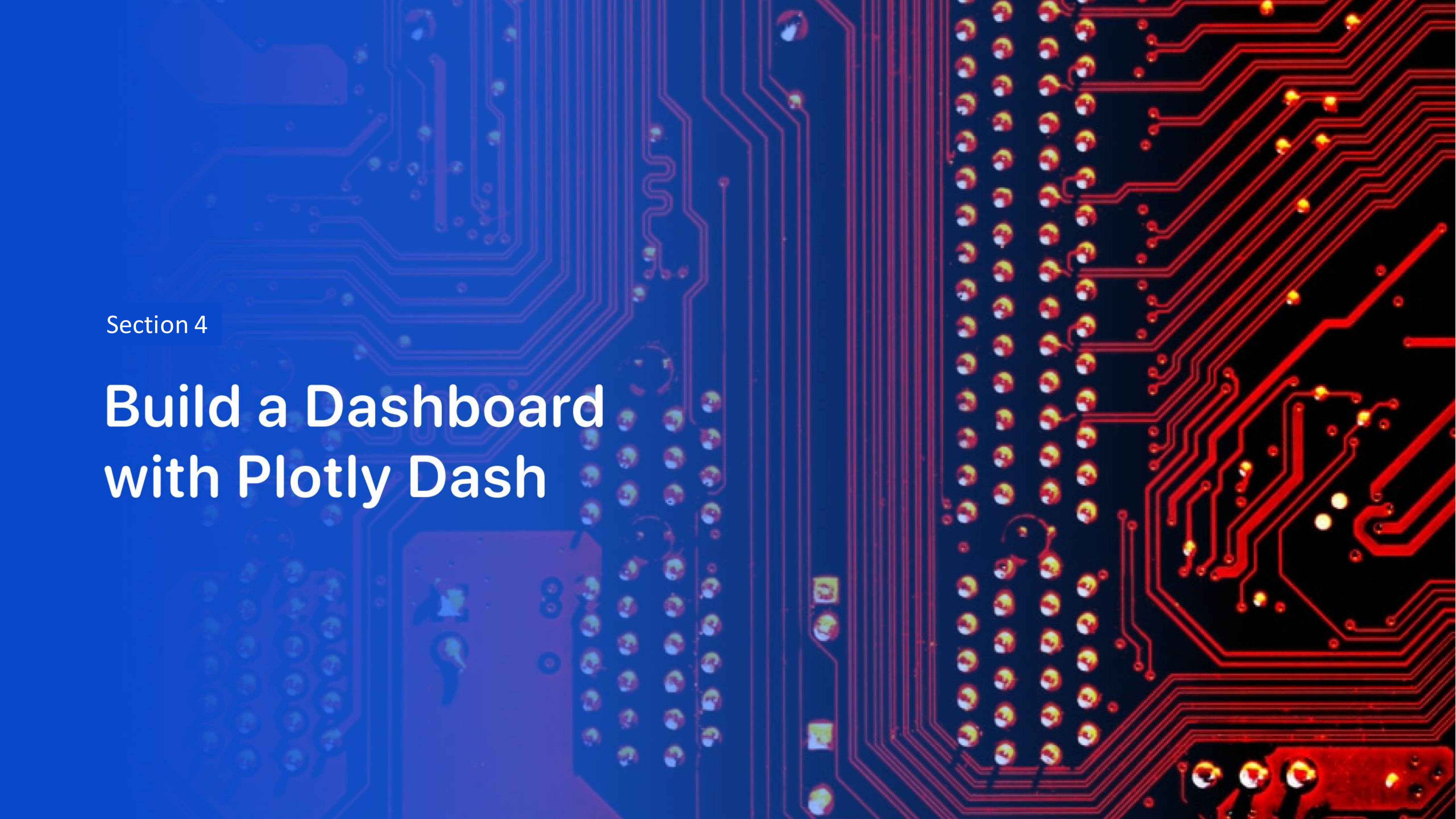
<Folium Map Screenshot 3>

- 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

Launch site are relatively close to railway and highway for transport reasons.

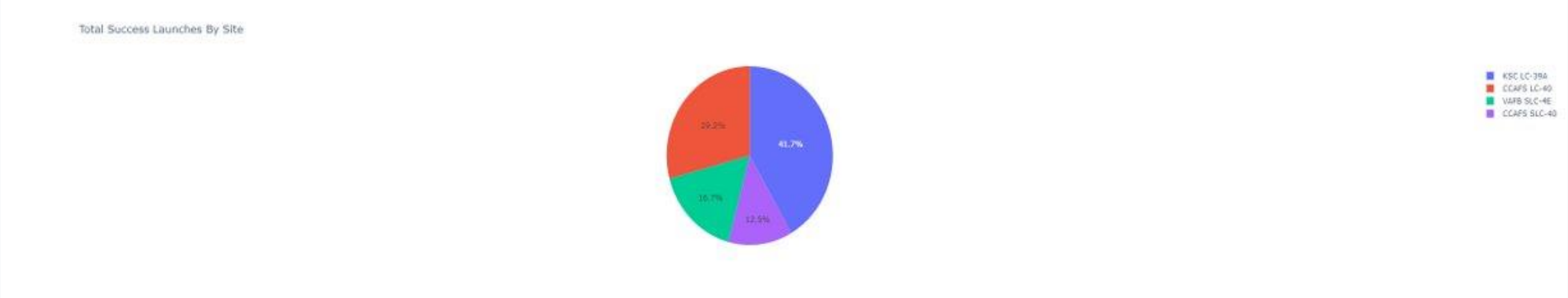


Section 4

Build a Dashboard with Plotly Dash

Pie chart showing the Launch site with the highest launch success ratio

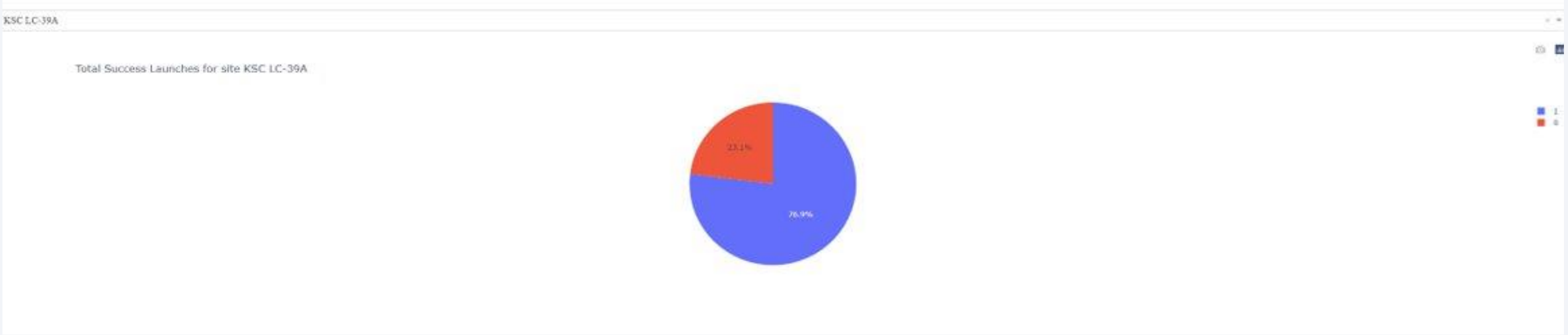
- Replace <Dashboard screenshot 1> title with an appropriate title
- Show the screenshot of launch success count for all sites, in a piechart



- Explain the important elements and findings on the screenshot

Pie chart showing the Launch site with the highest launch success ratio

- 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

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider

- 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

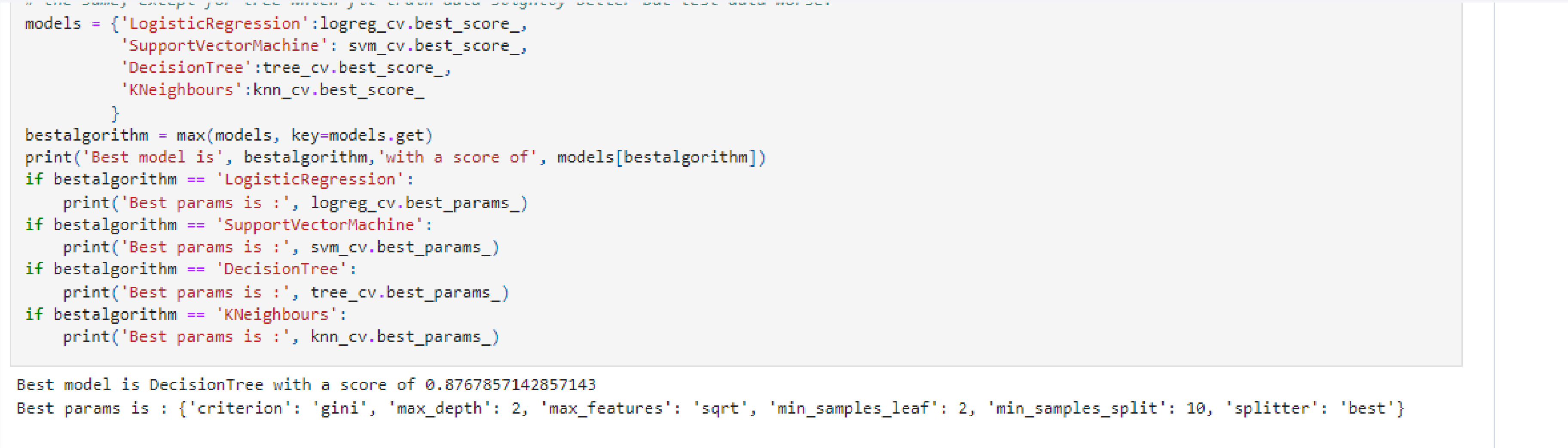
- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The decision tree classifier is the model with the highest classification accuracy

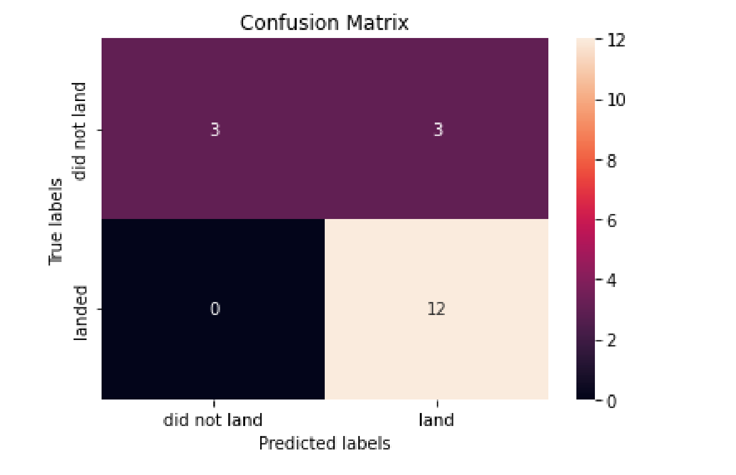


```
# the same, except for tree which fit train data slightly better but test data worse.
models = {'LogisticRegression':logreg_cv.best_score_,
          'SupportVectorMachine': svm_cv.best_score_,
          'DecisionTree':tree_cv.best_score_,
          'KNeighbours':knn_cv.best_score_
         }
bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVectorMachine':
    print('Best params is :', svm_cv.best_params_)
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbours':
    print('Best params is :', knn_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8767857142857143
Best params is : {'criterion': 'gini', 'max_depth': 2, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'best'}
```

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives. i.e., unsuccessful landing marked as successful landing by the classifier.



Conclusions

We can conclude that:

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

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!