IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Venkatesan M
22/11/2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection through API
  - Data collection through Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result

Introduction

- Project background and context

Due to reusability of rocket first stage, SpaceX advertise Falcon9 rocket launches on its website with a cost of 62million dollars, as compared to other providers cost of 165million dollars each. The launch cost of SpaceX Falcon9 is depends on the first stage return to ground and reuse capability of next launch, so that we can compare alternate company wants to bid against SpaceX for a rocket launch. The prime goal of the project is to create machine learning pipeline to predict whether the first stage will land successfully to the ground.

- Problems you want to find answers

What factors determine if the rocket will land successfully?

What factors determine the rate of a successful landing of first stage?

What external operating conditions need 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
  - Data was processed using One-hot coding with 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
  - To build, tune, evaluate classification models based on the processed data.

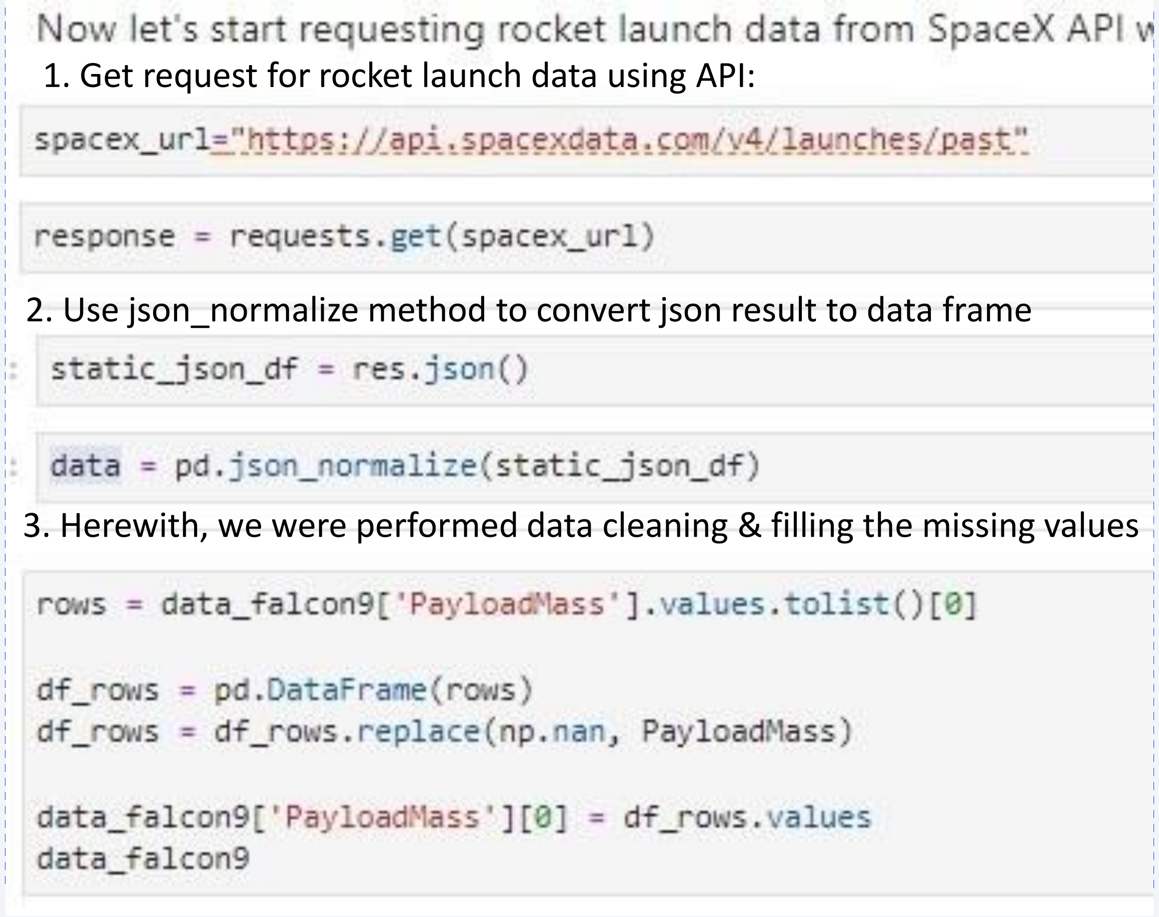
Data Collection

- Before begin with data collection, we need to import libraries, define auxiliary function and requesting rocket launch data from SpaceX API.
- Start with request and parse the SpaceX launch data using the GET request and specify the column name ‘rocket’, ‘payloads’, ‘launchpads’, ‘cores’.
- Since, there are more falcon rocket launch data available, we need to fix the data frame specific to Falcon9 launches.

- In addition, we have performed web scraping from Wikipedia url for Falcon 9 launch records with BeautifulSoup.
- The objective was to extract the launch records as followed by HTML table, parse the table and convert it to a pandas dataframe for further analysis.

Data Collection – SpaceX API

- Data Collection starts with requesting launch data from SpaceX API using request.get(spacex_url)
- The data received from url is .json, we need to convert the data into .json_normalize
- After the normalized, the data have to be deal with missing values, in order to ensure we don’t have no missing values throughout the data frame.
- Finally, the cleaned data can be export and saved as .csv file using the .to_csv function.
- The Link to the notebook is,
- https://github.com/venkataizan/IBM-Data-Science-Professional-Certificate-Course-Work/blob/main/10.%20Applied%20Data%20Science%20Capstone/jupyter-labs-spacex-data-collection-api%20(2).ipynb



Now let's start requesting rocket launch data from SpaceX API w

1. Get request for rocket launch data using API:

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

2. Use json_normalize method to convert json result to data frame

```
static_json_df = res.json()
```

```
data = pd.json_normalize(static_json_df)
```

3. Herewith, we were performed data cleaning & filling the missing values

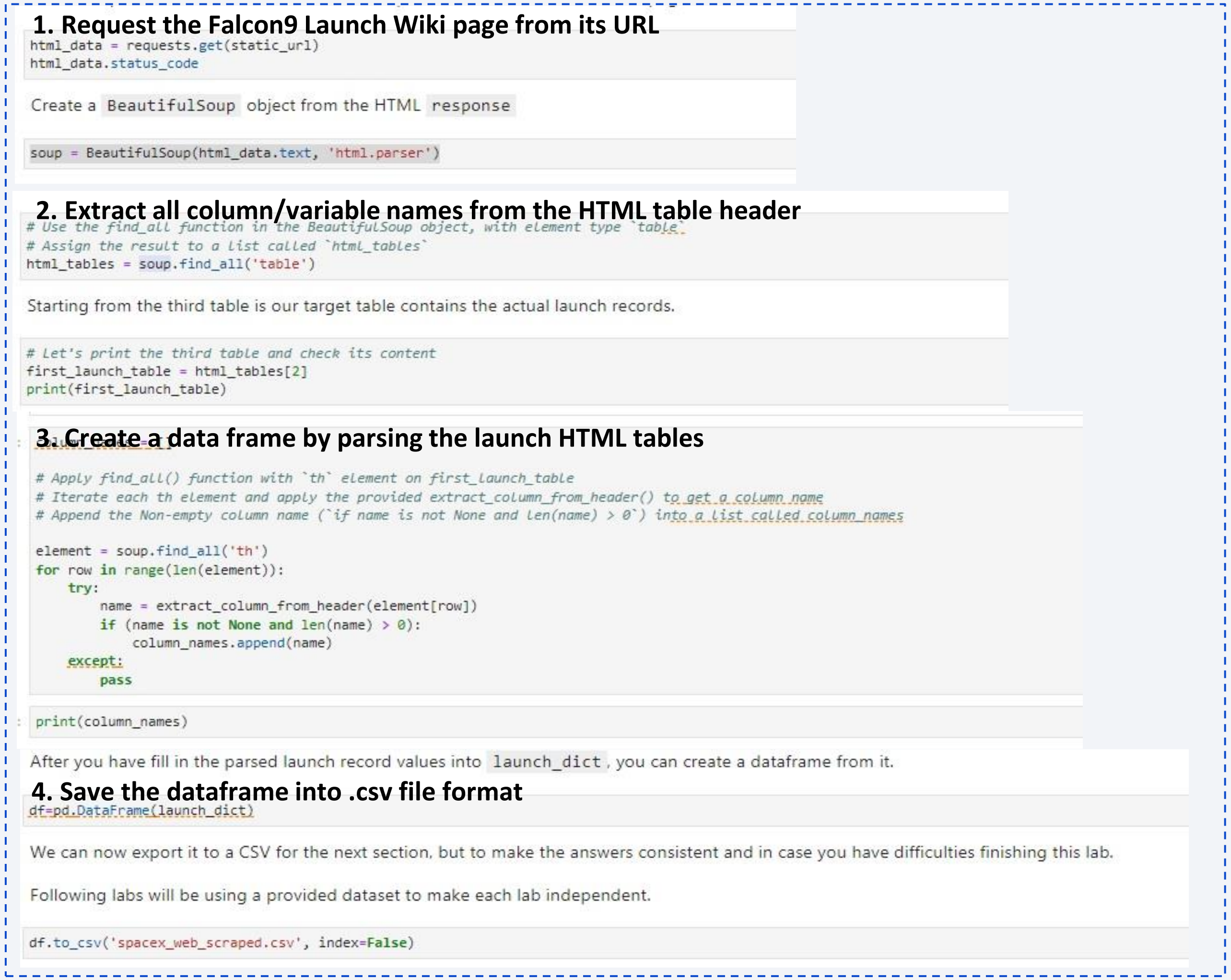
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

- Request the Falcon9 Launch Wiki page from its URL using BeautifulSoup object from HTML response.
- Next extracting all variable names from the HTML table header.
- Prepared a data frame by parsing the launch HTML tables.
- The Link to the notebook is,
- https://github.com/venkataizan/IBM-Data-Science-Professional-Certificate-Course-Work/blob/main/10.%20Applied%20Data%20Science%20Capstone/jupyter-labs-webscraping%20(1).ipynb



1. Request the Falcon9 Launch Wiki page from its URL

```
html_data = requests.get(static_url)
html_data.status_code
```

Create a `BeautifulSoup` object from the HTML `response`

```
soup = BeautifulSoup(html_data.text, 'html.parser')
```

2. Extract all column/variable names from the HTML table header

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

Starting from the third table is our target table contains the actual launch records.

```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

3. Create a data frame by parsing the launch HTML tables

```
# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

```
print(column_names)
```

After you have fill in the parsed launch record values into `launch_dict`, you can create a dataframe from it.

4. Save the dataframe into .csv file format

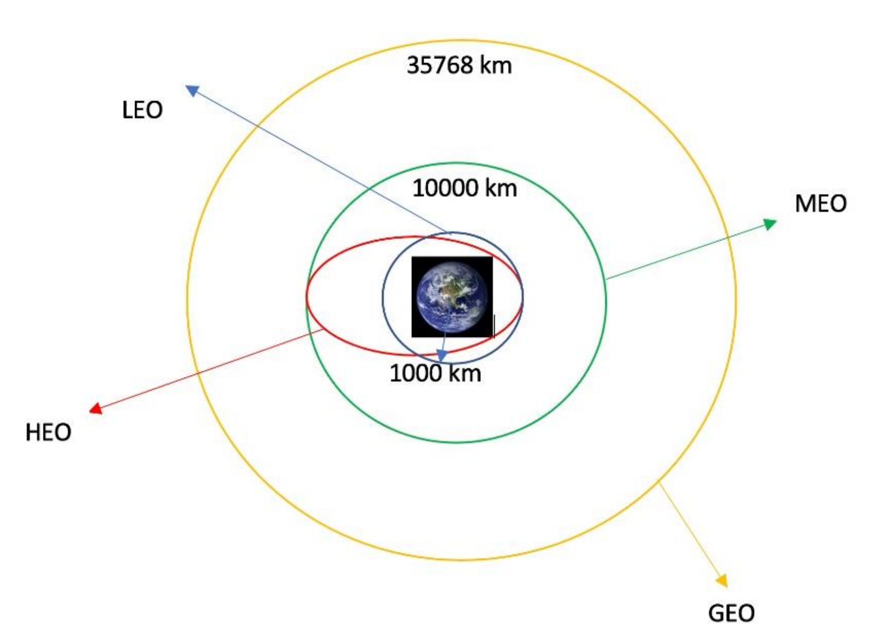
```
df=pd.DataFrame(launch_dict)
```

We can now export it to a CSV for the next section, but to make the answers consistent and in case you have difficulties finishing this lab.

Following labs will be using a provided dataset to make each lab independent.

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

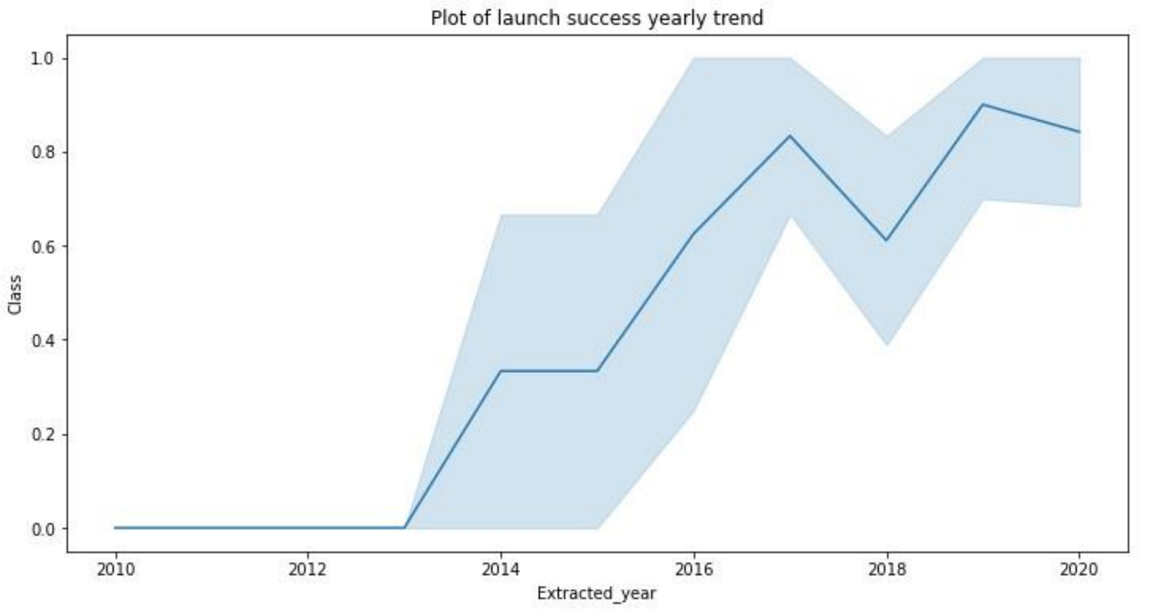


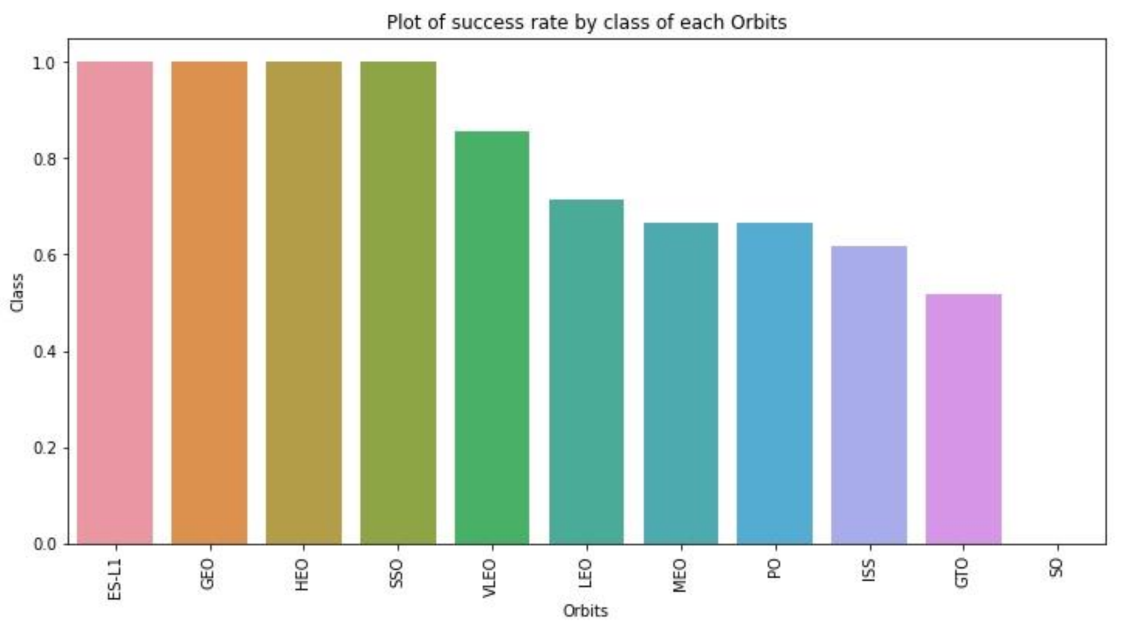
- Calculated the number of launches on each site, occurrence of each orbit using method .value_counts() to determine 'Orbit', 'landing_outcomes'
- Create a landing outcome label from Outcome column using 'landing_class'.
- Finally, exported the results to csv using 'to_csv'
- The Link to the notebook is,

https://github.com/venkataizan/IBM-Data-Science-Professional-Certificate-Course-Work/blob/main/10.%20Applied%20Data%20Science%20Capstone/IBM-DS0321EN-SkillsNetwork_labs_module_1_L3_labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

EDA with Data Visualization

- We have explored the visualization of data between,
  - Flight number and Launch site
  - Payload and Launch site
  - Flight number and orbit type
  - success rate of each orbit type
  - the launch success yearly trend





The Link to the notebook is,

https://github.com/venkataizan/IBM-Data-Science-Professional-Certificate-Course-Work/blob/main/10.%20Applied%20Data%20Science%20Capstone/IBM-DS0321EN-SkillsNetwork_labs_module_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

- We loaded the SpaceX dataset into MySQL database without leaving the jupyter notebook. We applied EDA with SQL to get insight from the data.
  - Collecting insights from the data begins with connecting database with jupyter notebook.
  - Collecting the names of the unique launch sites in the space mission & Displayed launch sites begin with string ‘CCA’.
  - The total payload mass carried by booster launched by NASA (CRS) & the average payload mass carrier by booster version F9 v1.1. The date when the first successful landing outcomes in ground pad was achieved and failure mission outcomes
  - Listed out the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000 & the total number of successful and failure mission outcomes
  - The names of the booster versions which have carried the maximum payload mass using subquery & the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order.
  - https://github.com/venkataizan/IBM-Data-Science-Professional-Certificate-Course-Work/blob/main/10.%20Applied%20Data%20Science%20Capstone/jupyter-labs-eda-sql-coursera_sqllite-Copy1.ipynb

Build an Interactive Map with Folium

- We marked all launch sites and added map objects like folium.markers, folium.lines, and folium.circle to show the success or failure of launches for each sites on the folium maps.
- We have created the binary class for launch sites as ‘0’ for failure and ‘1’for success with color-labeled marker clusters as ‘red’ and ‘green’.
- We have calculated the distance between a launch sites to its proximities.
- We have answered some proximities question as follow,
  - Are launch sites in close proximity to railways, highways, coastline?
  - Do launch sites keep certain distance away from cities?
  - Link: https://github.com/venkataizan/IBM-Data-Science-Professional-Certificate-Course-Work/blob/main/10.%20Applied%20Data%20Science%20Capstone/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite%20(1).ipynb

Build a Dashboard with Plotly Dash

- We have built an interactive dashboard with plotly dash
- We have plotted a pie charts showing the total launch in each sites.
- We rendered scatter graph showing relationship with Outcomes and Payload mass(Kg) for different booster category.
- Finally, we get some meaningful insights visually from the plots and answered some useful questions.

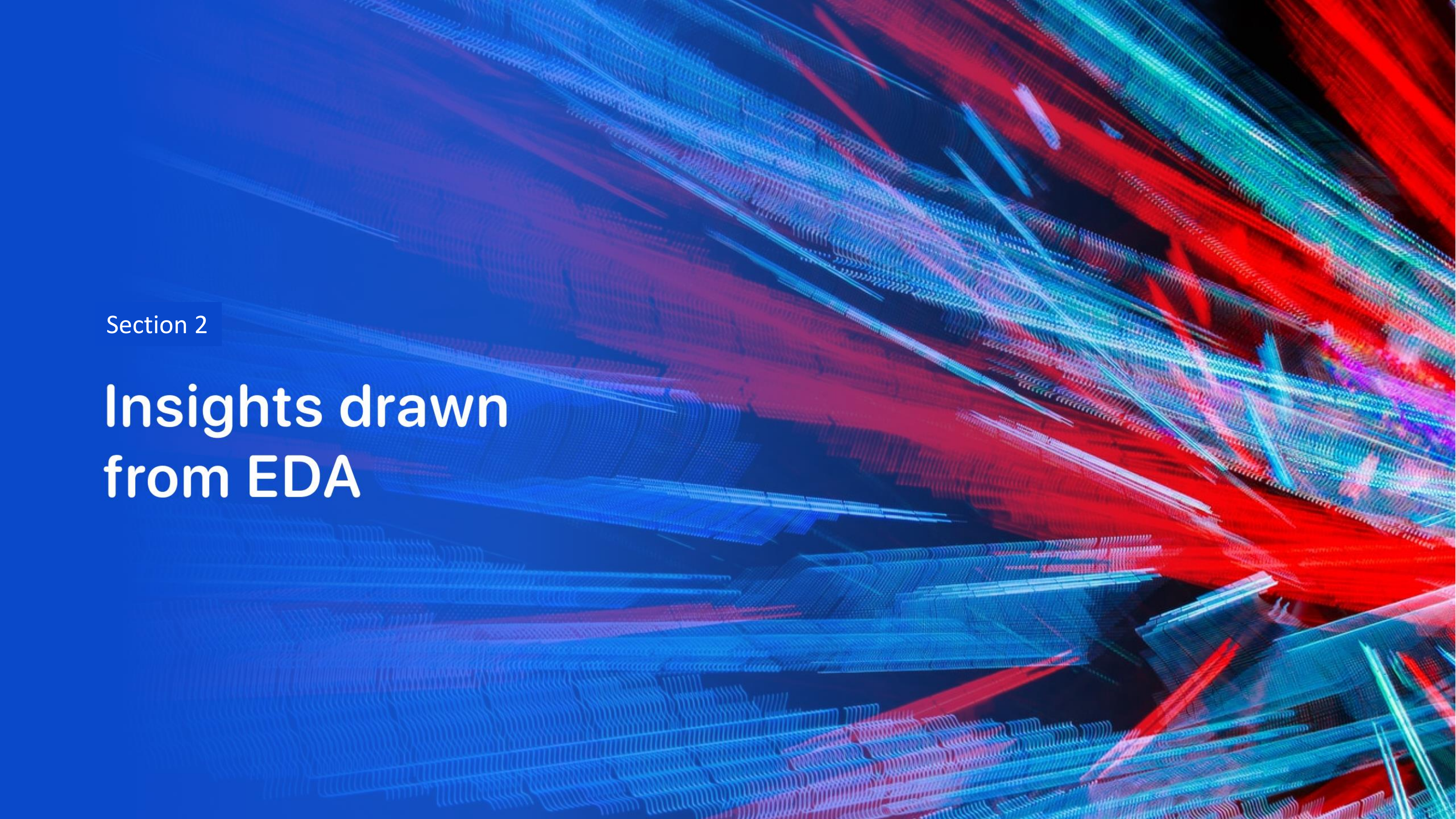
Predictive Analysis (Classification)

- We have loaded the data using numpy and pandas, then transformed the data, split our data into training and testing data using train_test_split.
- Then, the models are trained and hypterparameters are selected using the function GridsearchCV object logreg_cv. We have calculated accuracy of test data using method score.
- We have iterated the models logistic regression, support vector machine, decision tree classifier, k-nearest neighbors and plotted the confusion matrix for each machine learning models.
- Finally, we got better accuracy in decision tree classifier as compared to others.
- The link to the notebook is,

https://github.com/venkataizan/IBM-Data-Science-Professional-Certificate-Course-Work/blob/main/10.%20Applied%20Data%20Science%20Capstone/IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

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

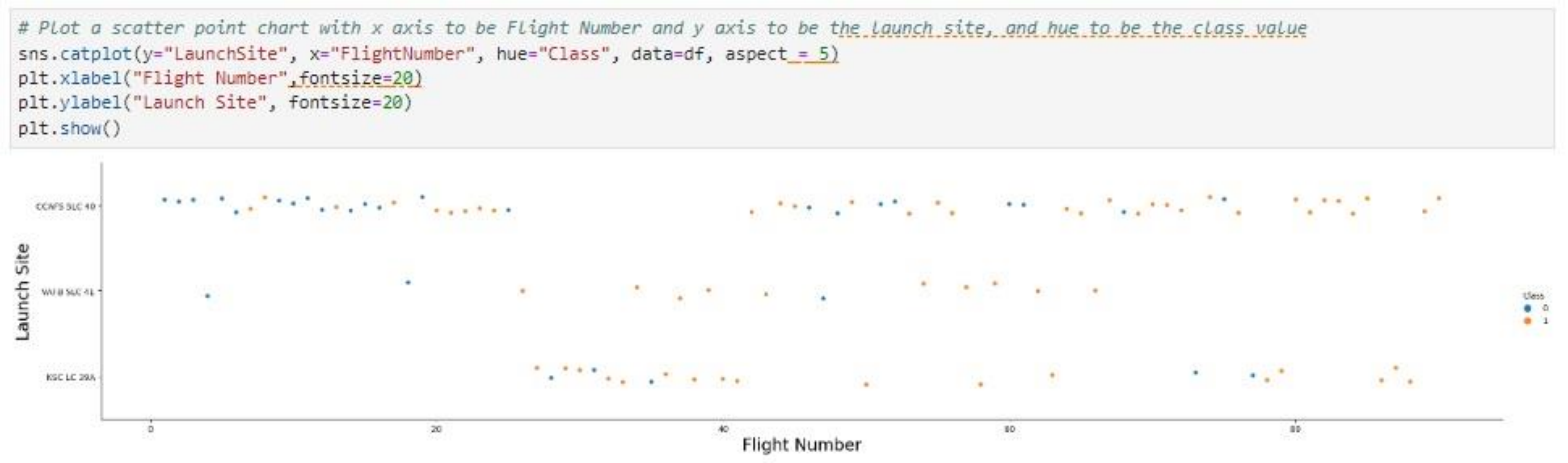


Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

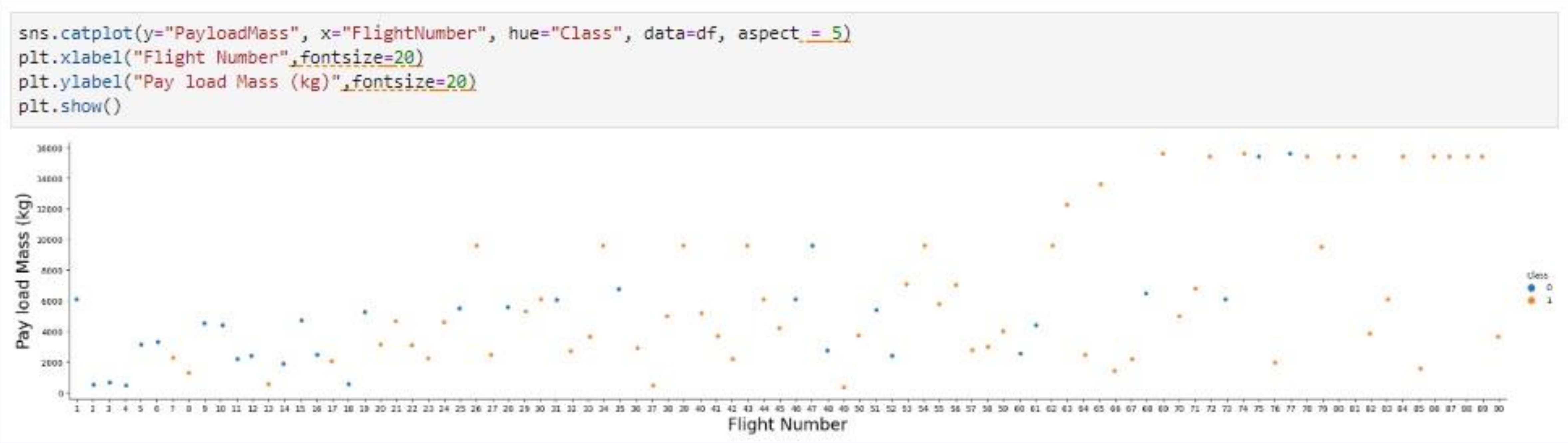
- From the graph, it’s found that larger the flight number at launch site, the greater the success rate at a launch site.



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

Payload vs. Launch Site

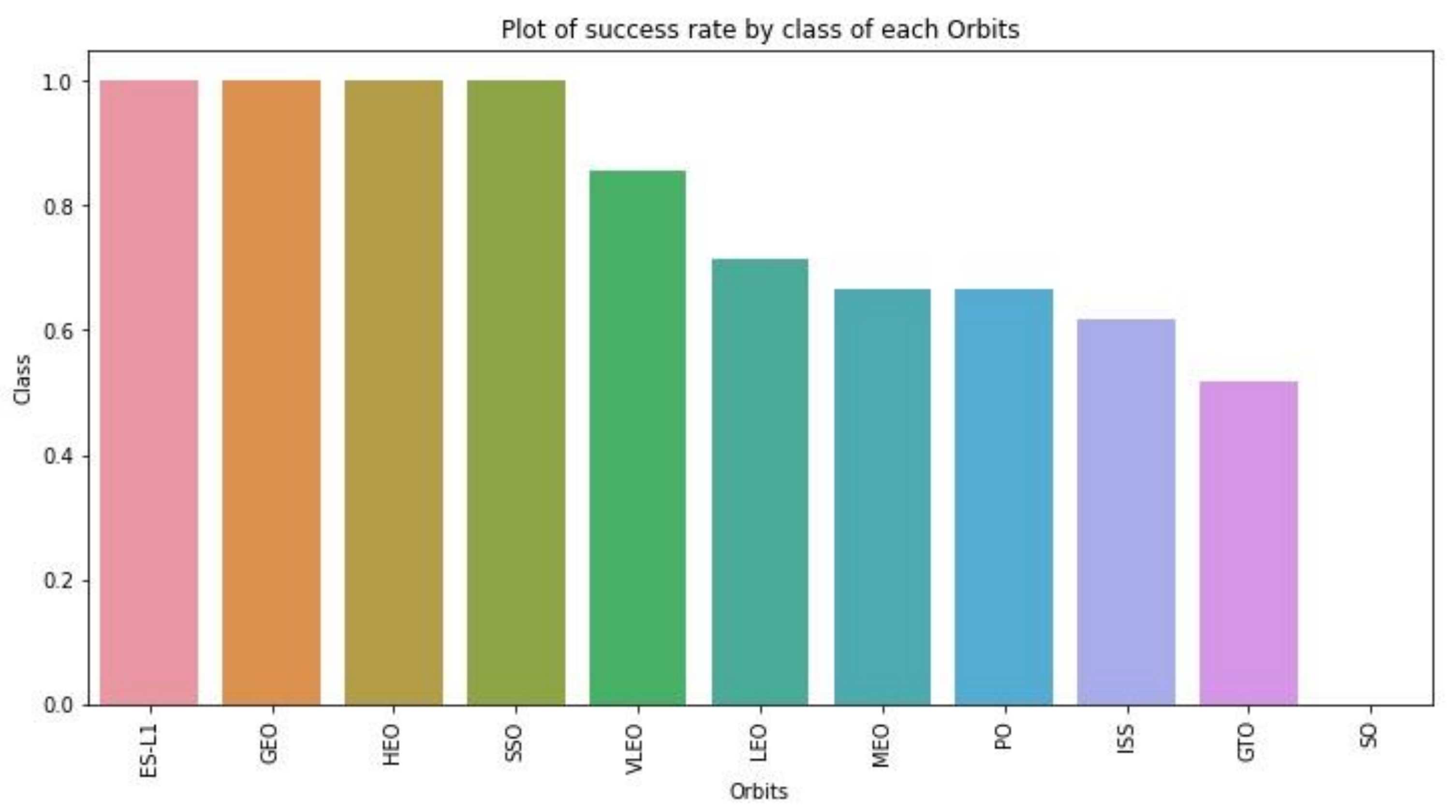
- From the graph, it's proven that the low payload mass would decrease the success rate. In case of high payload mass, the rate of success is high (orange dot)



```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```

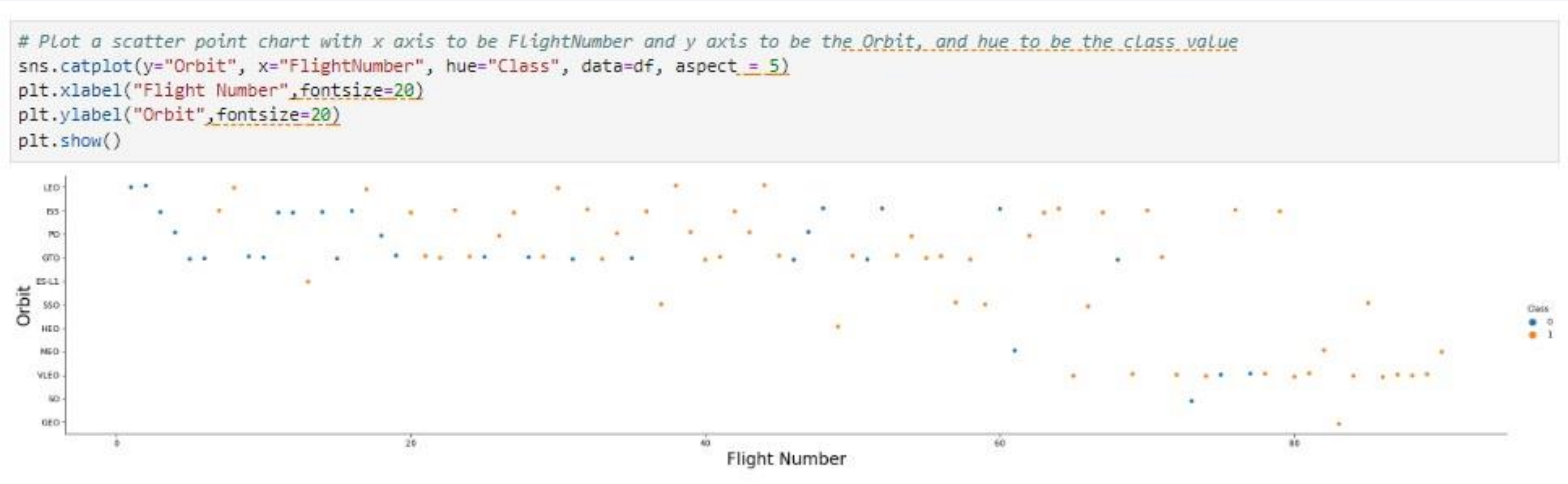
Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO are the most successful rate by class of each orbits.



Flight Number vs. Orbit Type

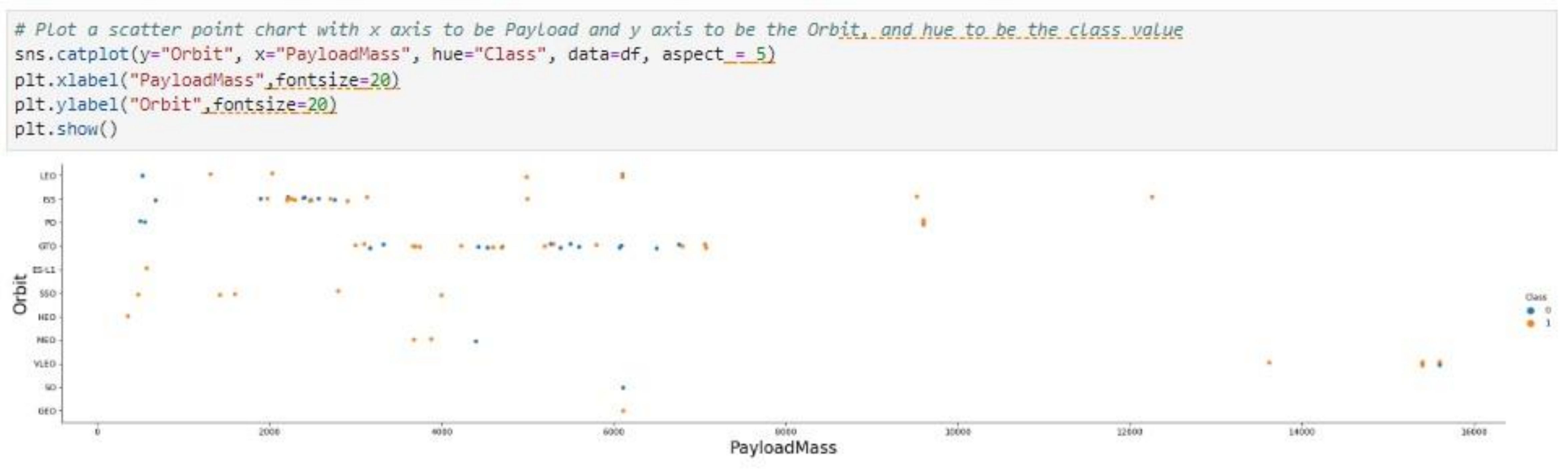
- From the below plot, we can see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.



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

Payload vs. Orbit Type

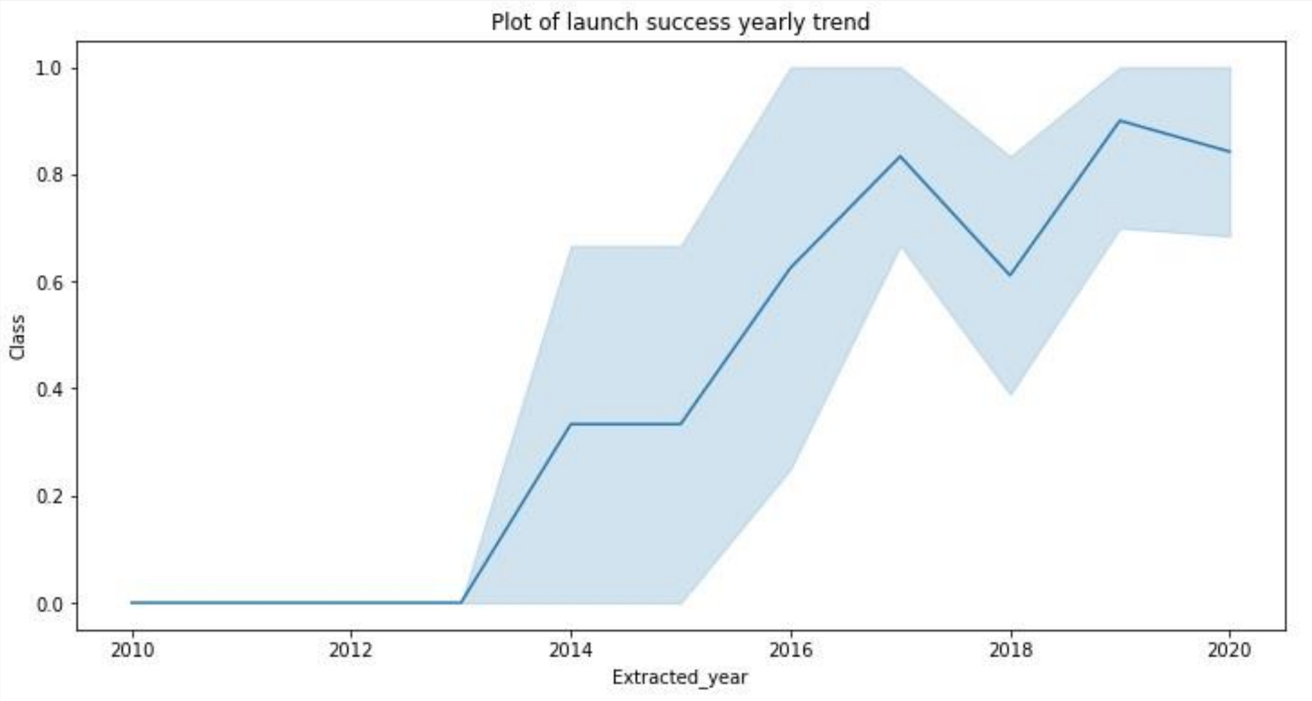
- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.



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

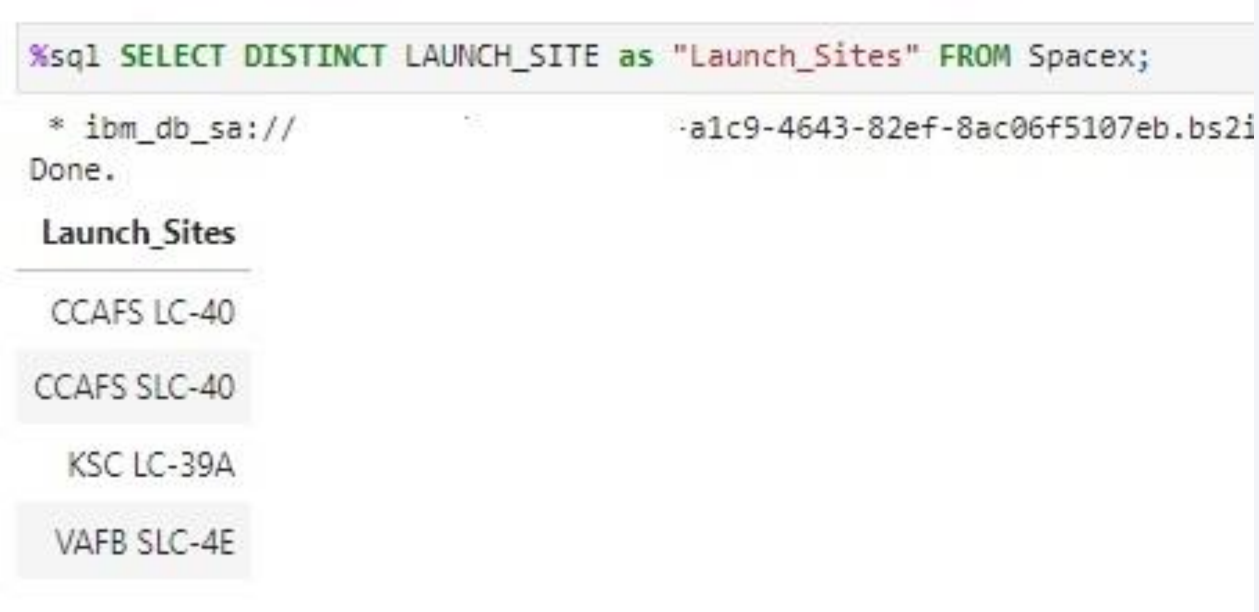
Launch Success Yearly Trend

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



All Launch Site Names

- We have used DISTINCT keyword to show only unique launch sites from the SPACEX data.
- There are four launch sites used to launch rocket vehicles.



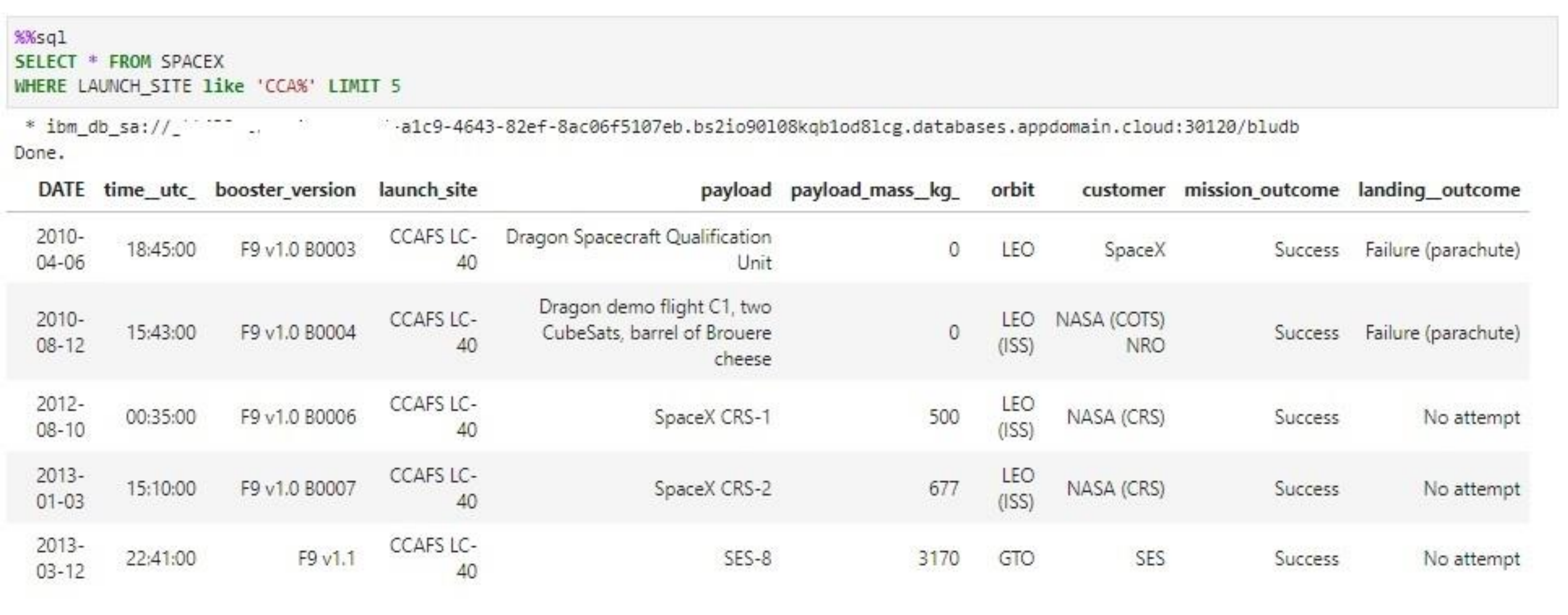
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM Spacex;
```

```
 * ibm_db_sa://                                 -a1c9-4643-82ef-8ac06f5107eb.bs2i
Done.
```

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

Launch Site Names Begin with 'CCA'

- We have used query below to display 5 records where launch sites begin with \`CCA\`



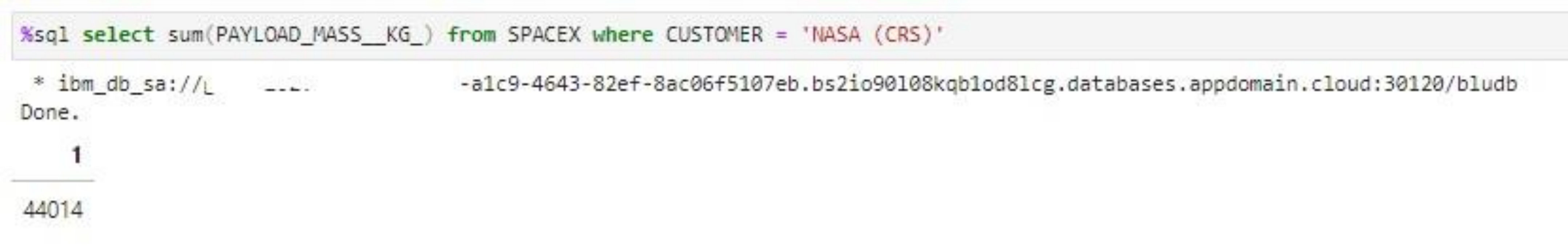
```
%%sql
SELECT * FROM SPACEX
WHERE LAUNCH_SITE like 'CCA%' LIMIT 5
```

```
 * ibm_db_sa://[redacted]-a1c9-4643-82ef-8ac06f5107eb.bs2io90108kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

DATE	time_utc_	booster_version	launch_site	payload	payload_mass_kg_	orbit	customer	mission_outcome	landing_outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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)
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-12	22:41:00	F9 v1.1	CCAFS LC-40	SES-8	3170	GTO	SES	Success	No attempt

Total Payload Mass

- We were calculated the total payload carried by boosters from NASA as 44014 using the query below.



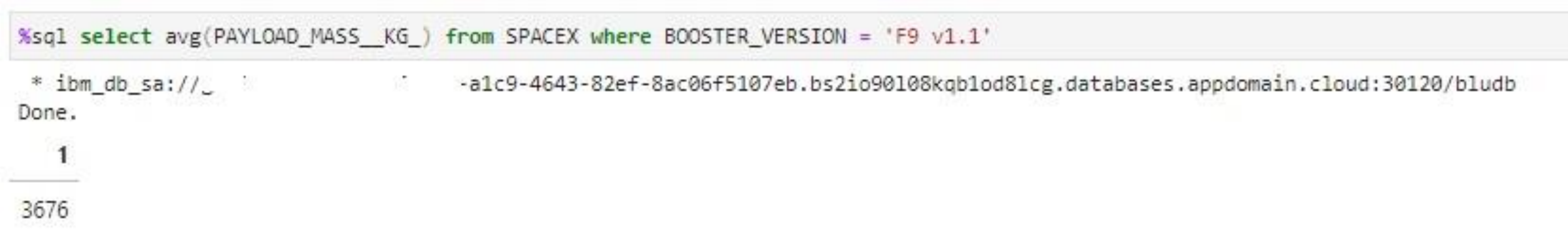
```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEX where CUSTOMER = 'NASA (CRS)'
```

```
 * ibm_db_sa://[redacted]-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

1
44014

Average Payload Mass by F9 v1.1

- We calculated the average payload mass carried by booster version F9 v1.1 as 3676.



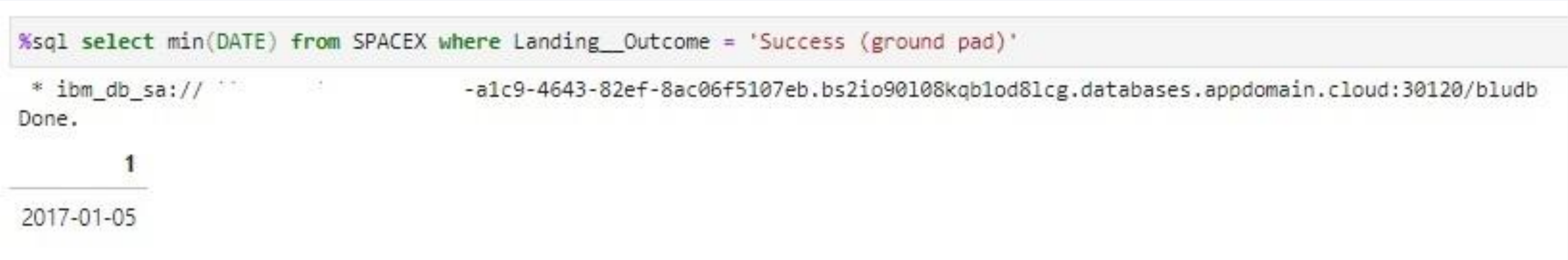
```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEX where BOOSTER_VERSION = 'F9 v1.1'
```

```
 * ibm_db_sa://_                     -a1c9-4643-82ef-8ac06f5107eb.bs2io90108kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

```
   1
-----
3676
```

First Successful Ground Landing Date

- We observed the dates of the first successful landing outcome on ground pad was 2017-01-05

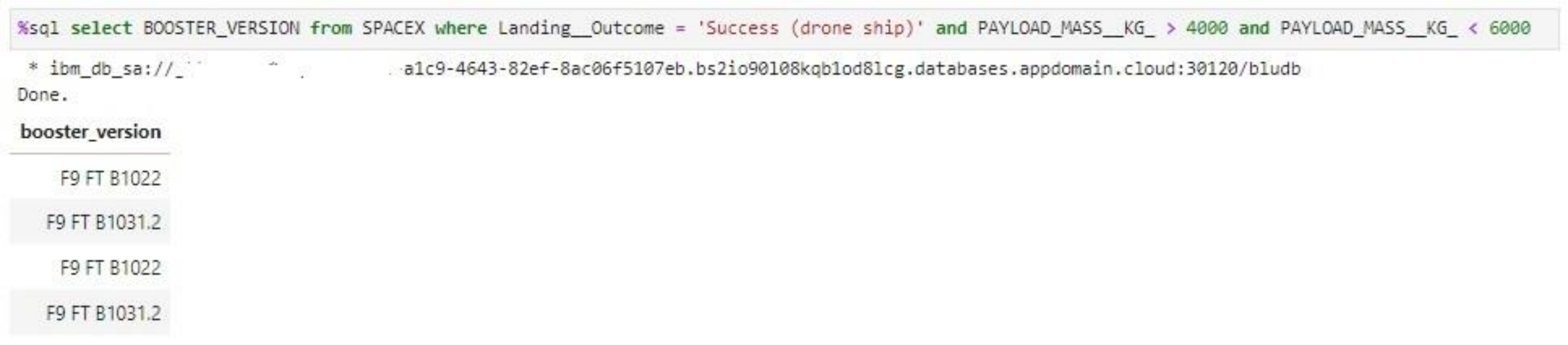


```
%sql select min(DATE) from SPACEX where Landing__Outcome = 'Success (ground pad)'
```

```
 * ibm_db_sa://                    -a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
         1
2017-01-05
```

Successful Drone Ship Landing with Payload between 4000 and 6000

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



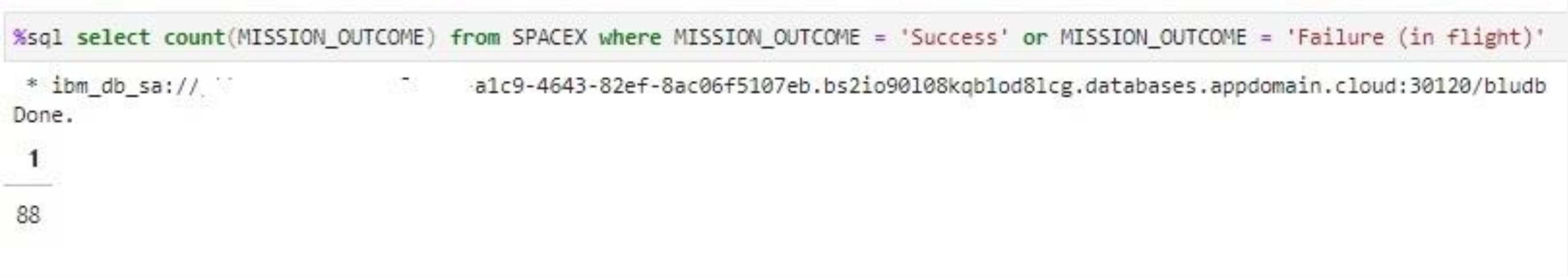
```
%sql select BOOSTER_VERSION from SPACEX where Landing__Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
```

```
 * ibm_db_sa://_                    a1c9-4643-82ef-8ac06f5107eb.bs2io90108kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

booster_version
F9 FT B1022
F9 FT B1031.2
F9 FT B1022
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

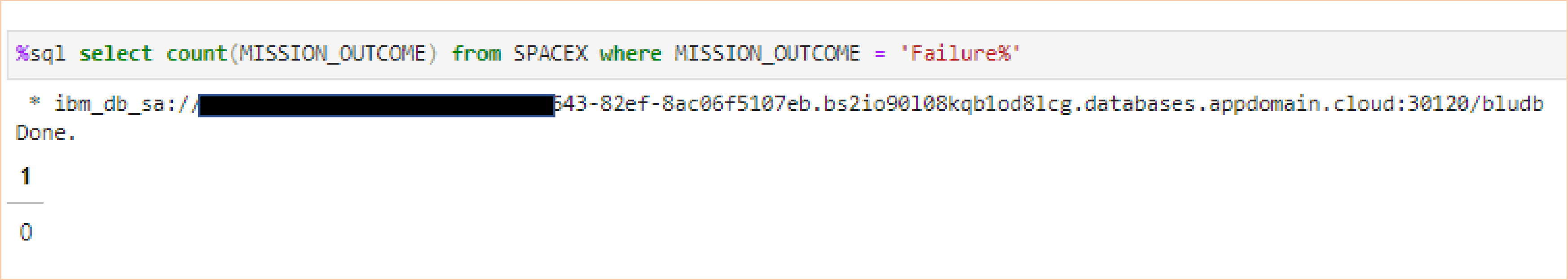
- We used wildcard symbol '%' to filter for WHERE MissionOutcome was success or failure
- The total successful Mission Outcome is '88' and failure is '0'



```
%sql select count(MISSION_OUTCOME) from SPACEX where MISSION_OUTCOME = 'Success' or MISSION_OUTCOME = 'Failure (in flight)'

 * ibm_db_sa://                    a1c9-4643-82ef-8ac06f5107eb.bs2io90108kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.

 1
88
```



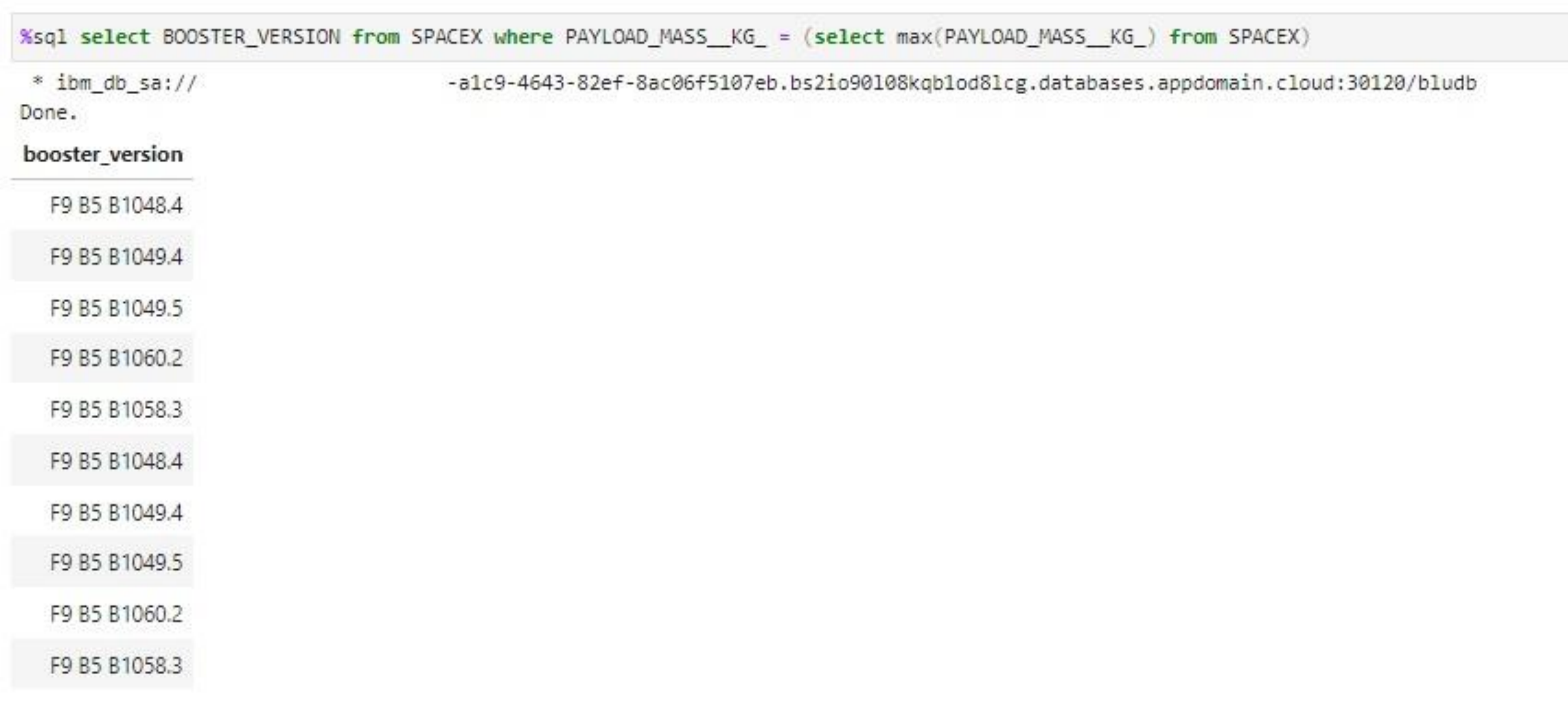
```
%sql select count(MISSION_OUTCOME) from SPACEX where MISSION_OUTCOME = 'Failure%'

 * ibm_db_sa://[redacted]43-82ef-8ac06f5107eb.bs2io90108kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.

1
0
```

Boosters Carried Maximum Payload

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



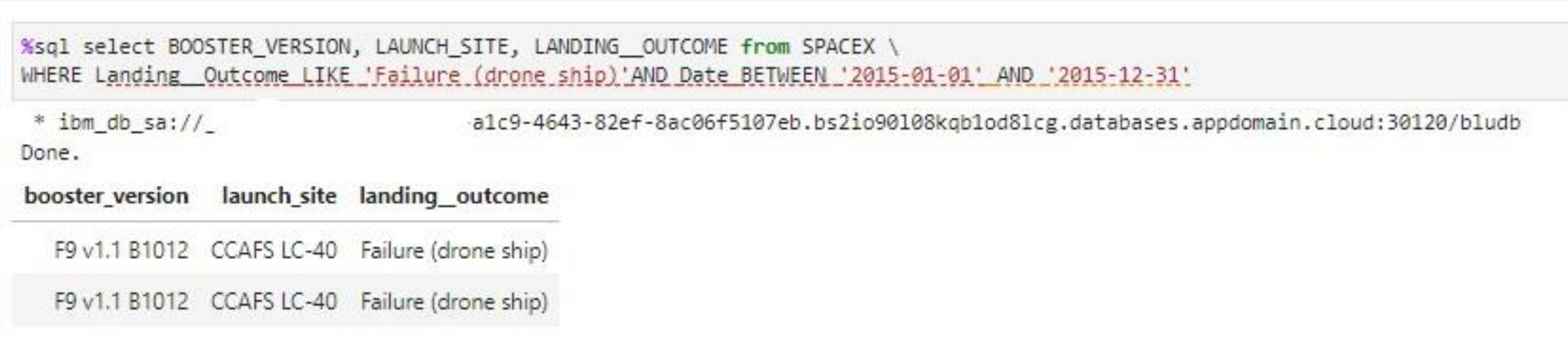
```
%sql select BOOSTER_VERSION from SPACEX where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEX)
```

```
 * ibm_db_sa://                    -a1c9-4643-82ef-8ac06f5107eb.bs2io90108kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

```
booster_version
  F9 B5 B1048.4
  F9 B5 B1049.4
  F9 B5 B1049.5
  F9 B5 B1060.2
  F9 B5 B1058.3
  F9 B5 B1048.4
  F9 B5 B1049.4
  F9 B5 B1049.5
  F9 B5 B1060.2
  F9 B5 B1058.3
```

2015 Launch Records

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



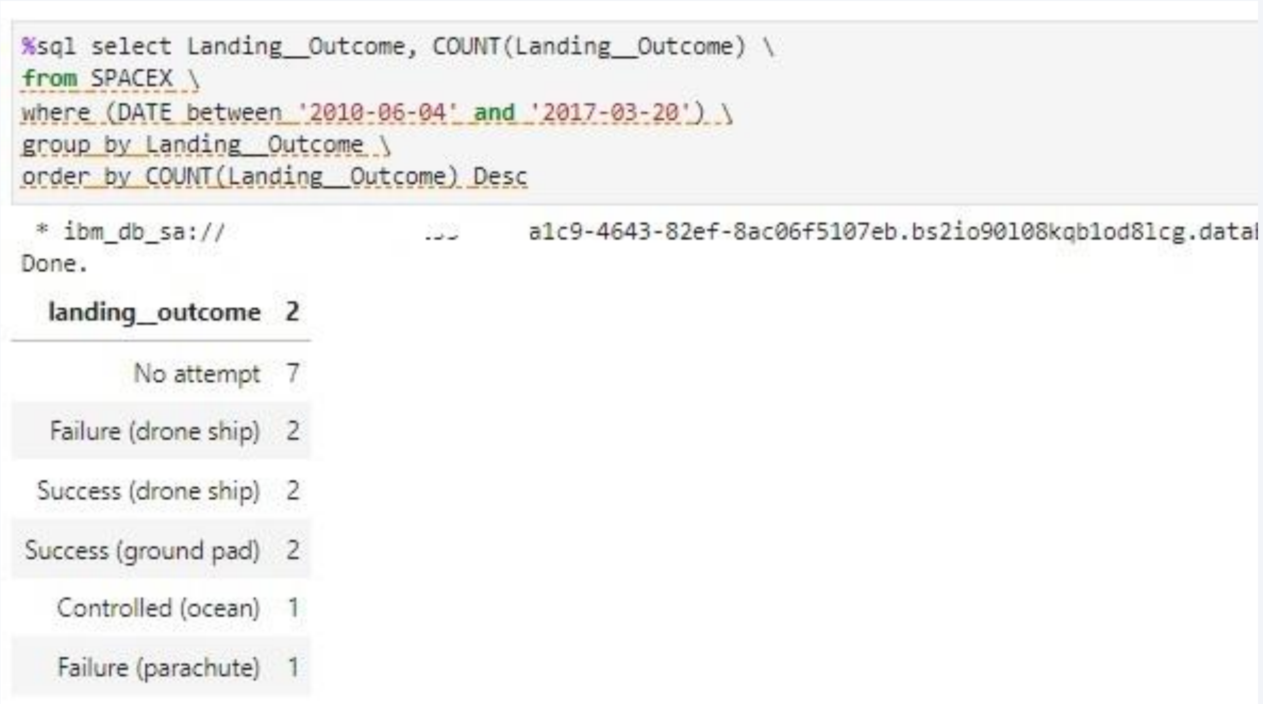
```
%sql select BOOSTER_VERSION, LAUNCH_SITE, LANDING__OUTCOME from SPACEX \
WHERE Landing__Outcome LIKE 'Failure (drone ship)'AND Date BETWEEN '2015-01-01' AND '2015-12-31'
```

```
 * ibm_db_sa://_                         a1c9-4643-82ef-8ac06f5107eb.bs2io90108kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

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

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

- We have selected Landing Outcomes and the **COUNT** of Landing Outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2017-03-20.
- We have applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcomes in descending order.



```
%sql select Landing__Outcome, COUNT(Landing__Outcome) \
from SPACEX \
where (DATE between '2010-06-04' and '2017-03-20') \
group by Landing__Outcome \
order by COUNT(Landing__Outcome) Desc
```

```
 * ibm_db_sa://                          a1c9-4643-82ef-8ac06f5107eb.bs2io90108kqblod8lcg.datal
Done.
```

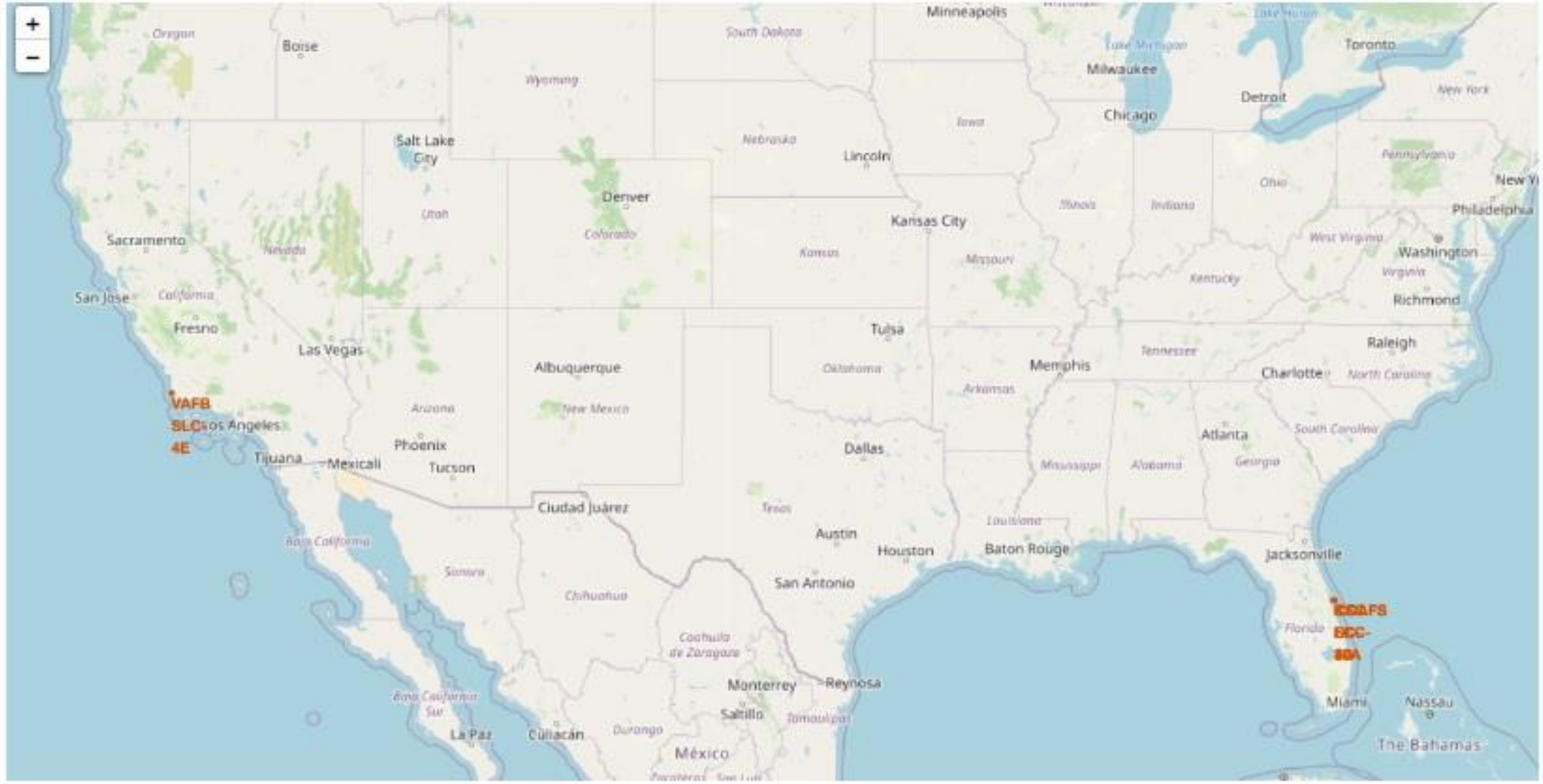
landing__outcome	2
No attempt	7
Failure (drone ship)	2
Success (drone ship)	2
Success (ground pad)	2
Controlled (ocean)	1
Failure (parachute)	1

Section 3

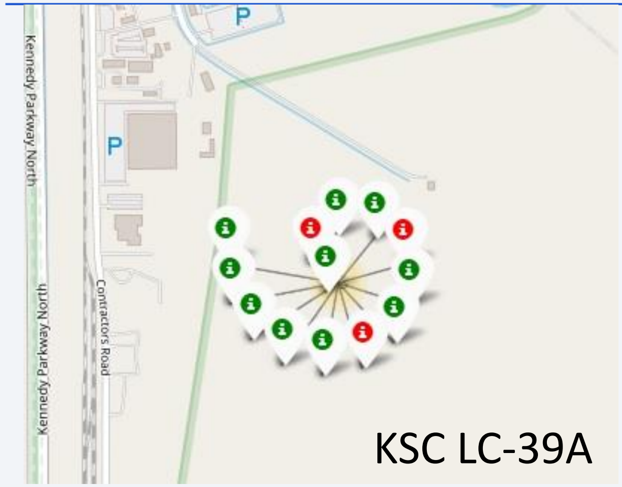
Launch Sites Proximities Analysis

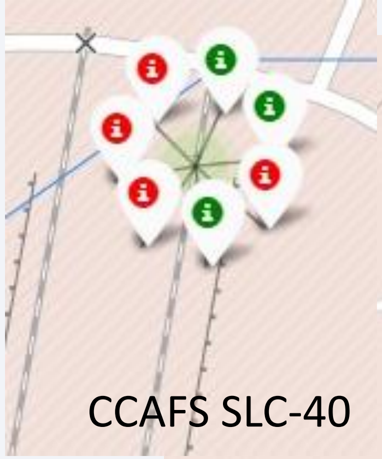
All launch sites US map makers

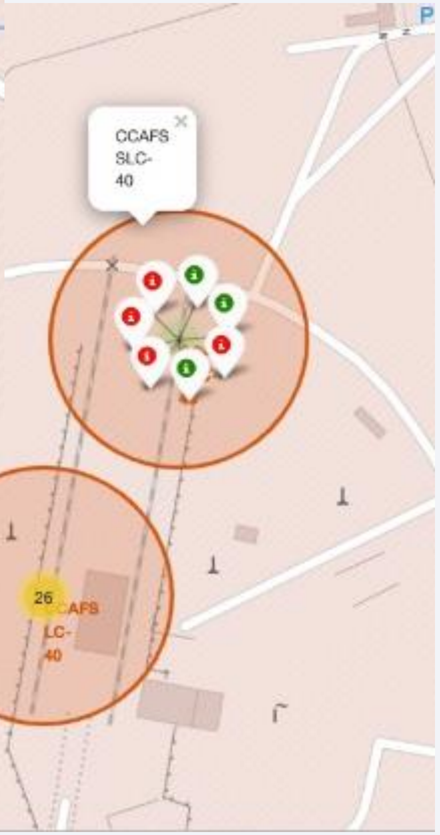
- We can able to see that the SpaceX launch site are in the United States of America coasts Florida and California

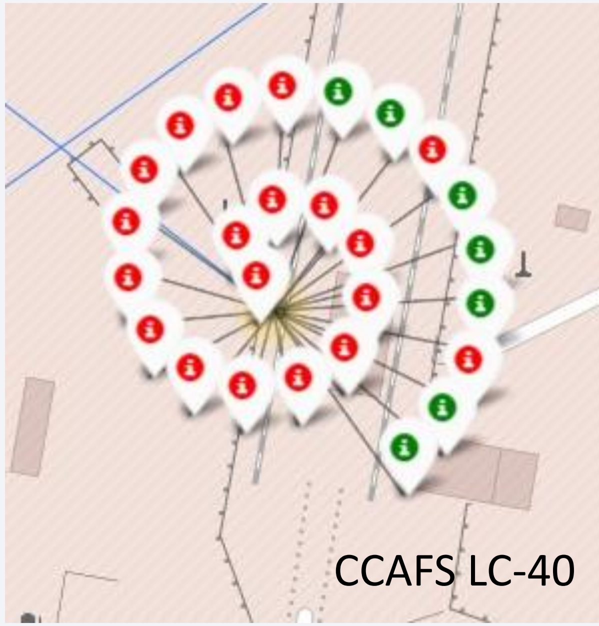


Markers showing launch sites with color labels



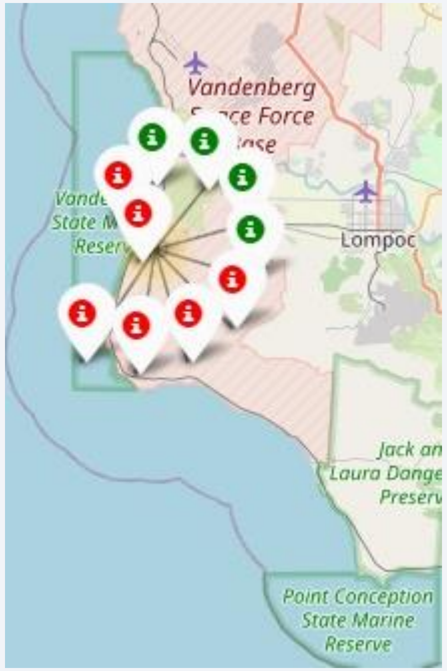






Florida Launch sites

Green marker shows successful Launches and Red marker shows failure.

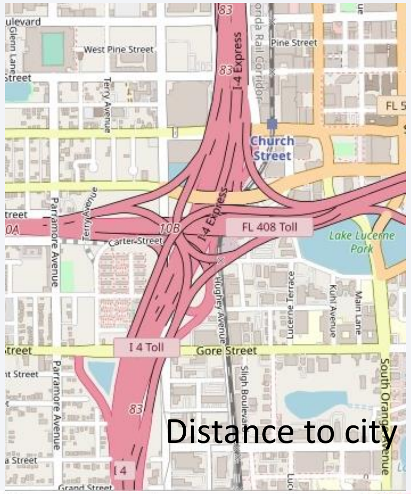


California Launch sites

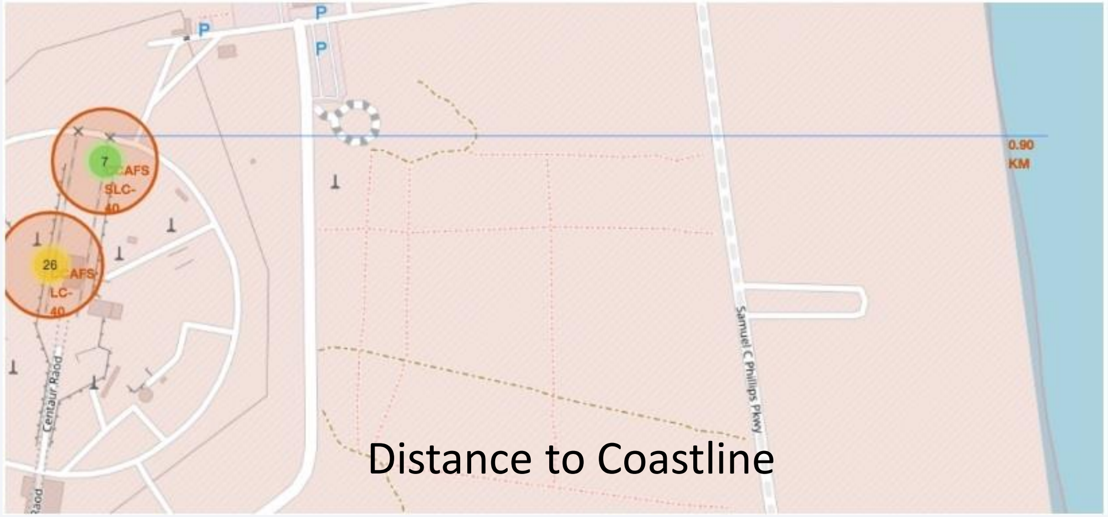
Launch Site distance to landmarks

Amelia Street
Garland Avenue
Lymmo
Lynx offices
A
B
C
D
E
F
G
H
J
K
L
M
N
P
Q
R
S
T
U
V
X
Y
Z
Lynx Central Station
Orlando Urban Trail
Central Florida Rail Corridor
Lymmo

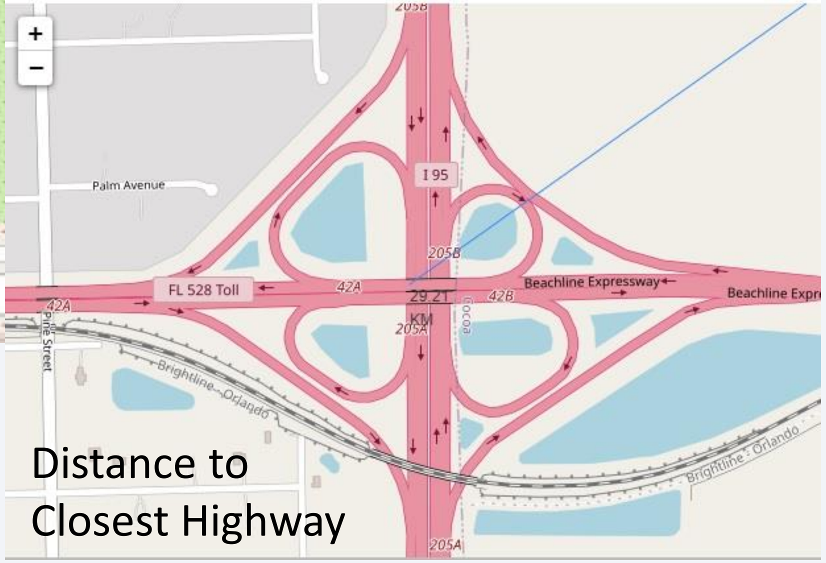
Distance to Railway Station



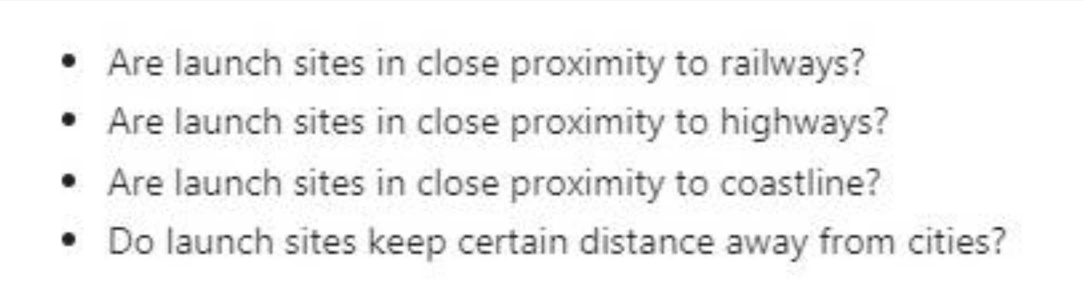
Distance to city



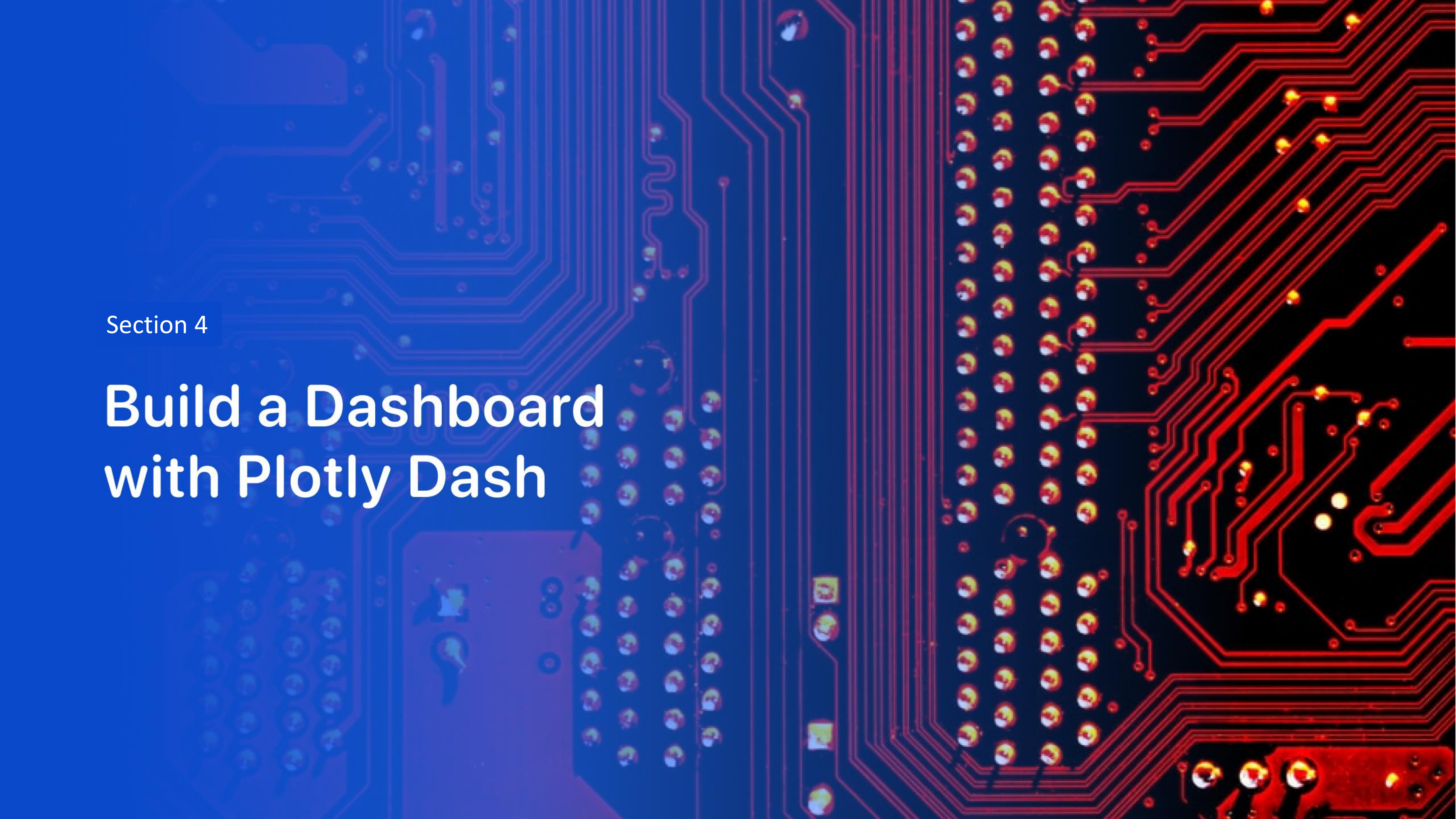
Distance to Coastline



Distance to Closest Highway



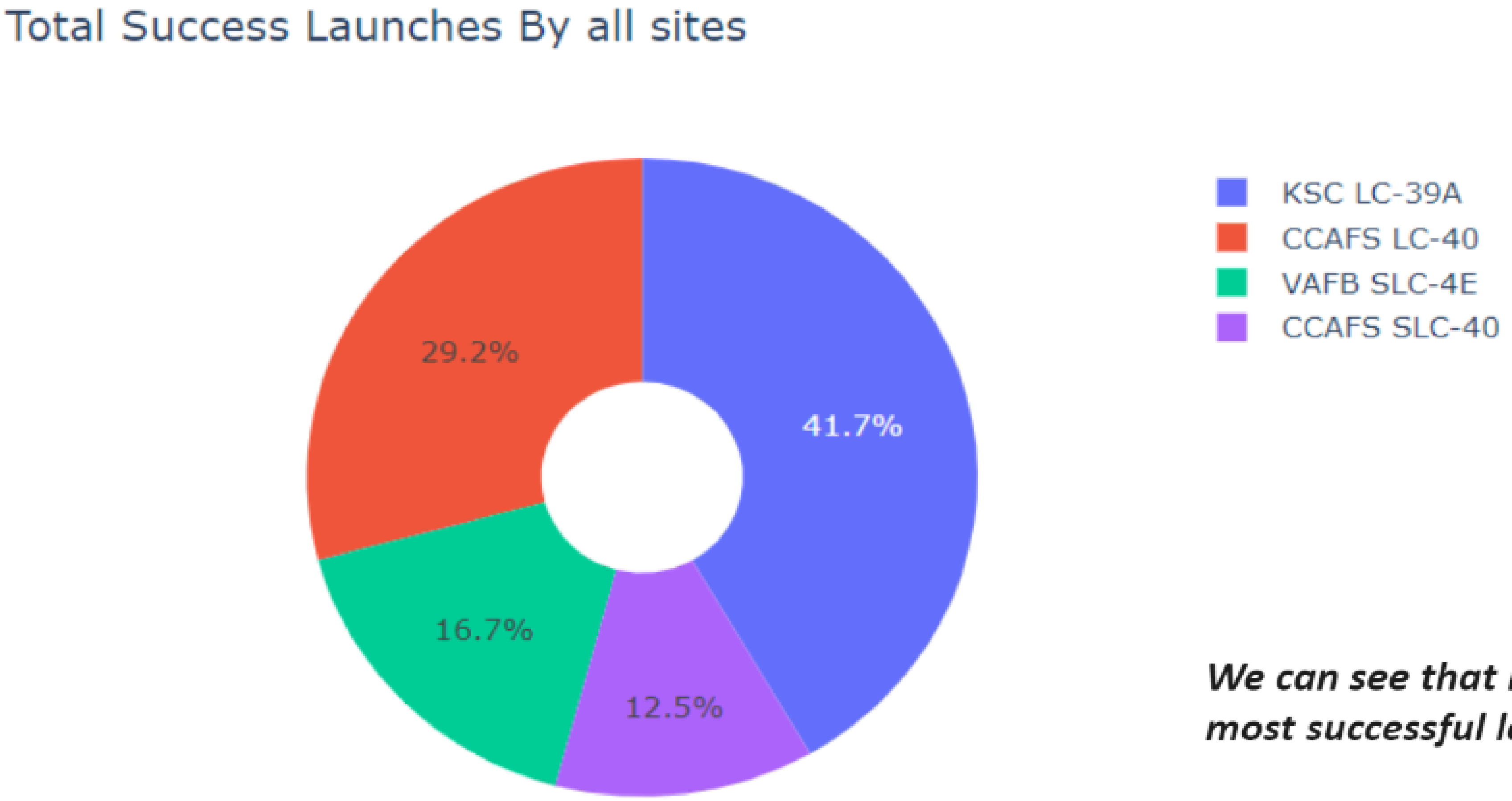
- Are launch sites in close proximity to railways?
- Are launch sites in close proximity to highways?
- Are launch sites in close proximity to coastline?
- Do launch sites keep certain distance away from cities?



Section 4

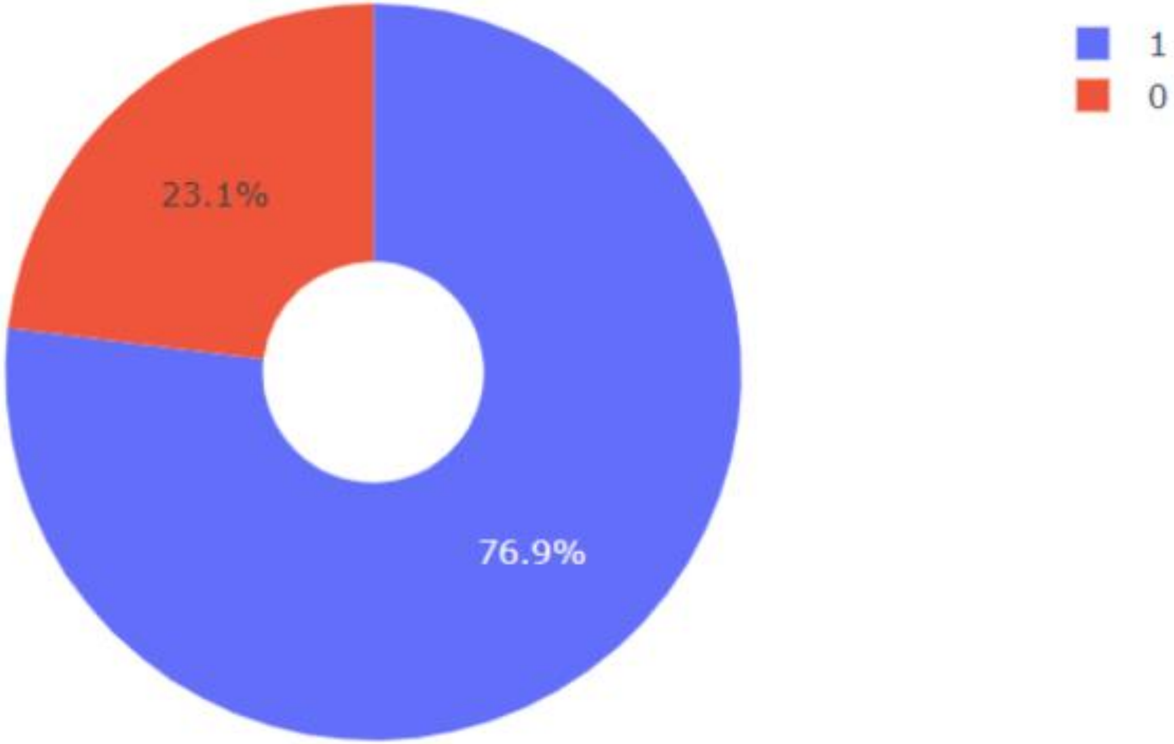
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



We can see that KSC LC-39A had the most successful launches from all the sites

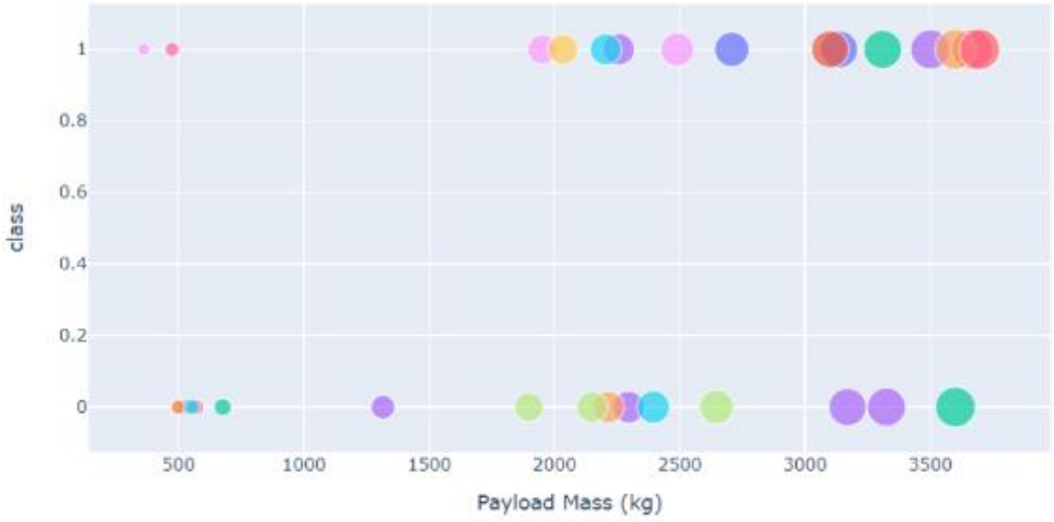
Pie chart showing the Launch site with the highest launch success ratio



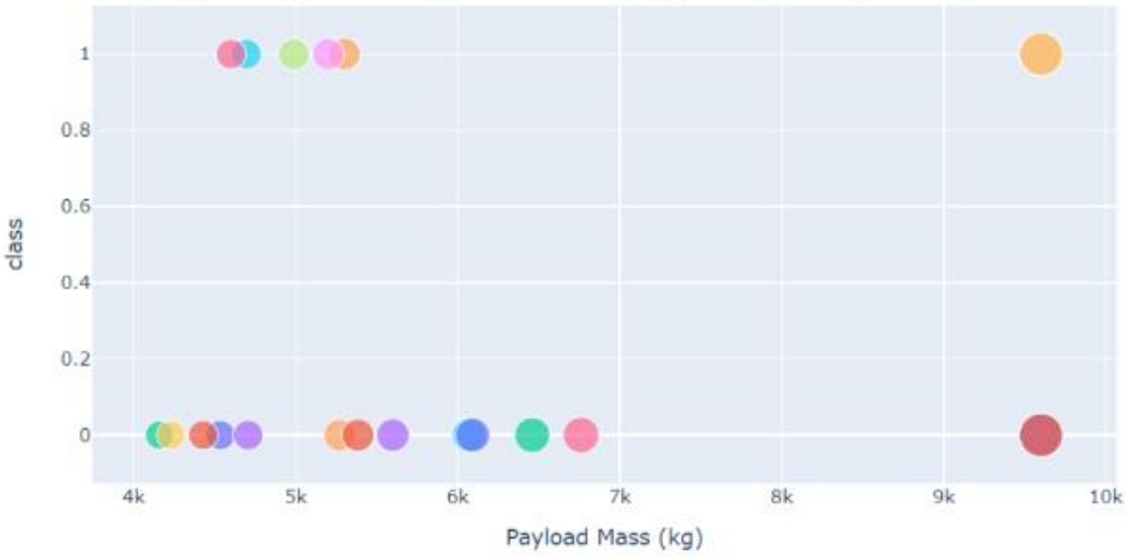
KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

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

Low Weighted Payload 0kg – 4000kg



Heavy Weighted Payload 4000kg – 10000kg



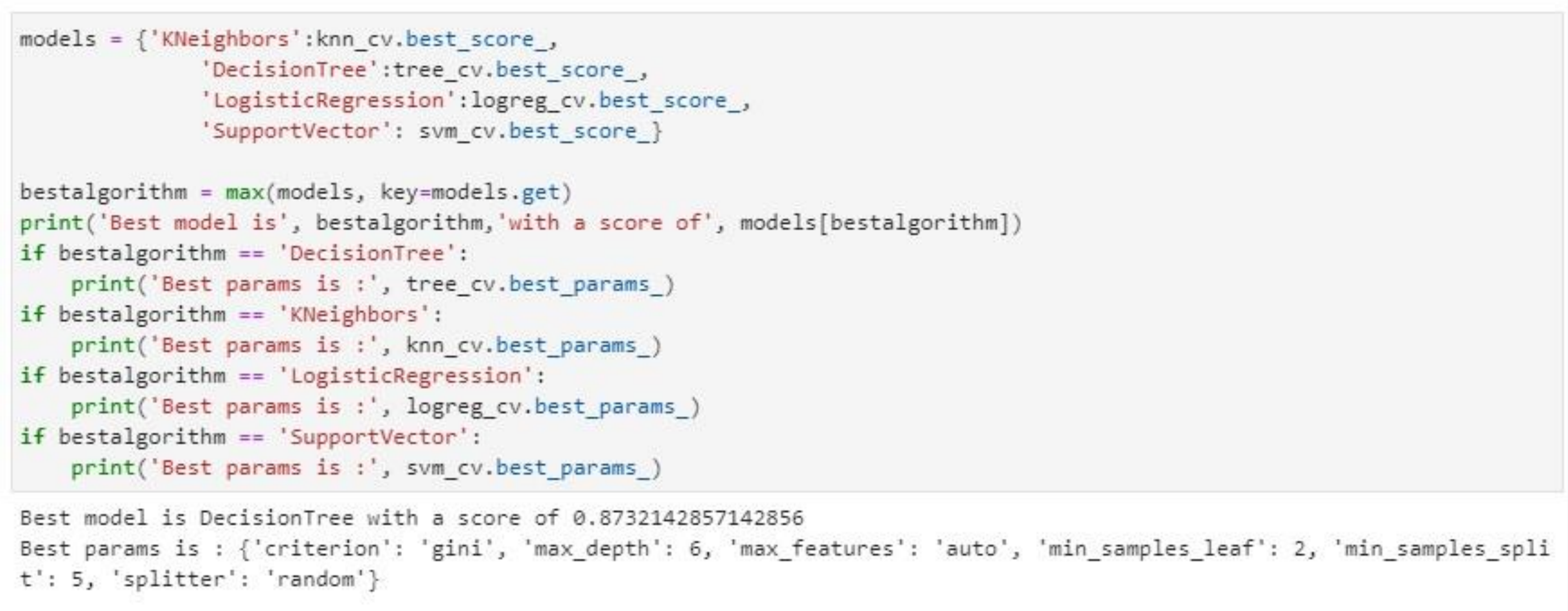
We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

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



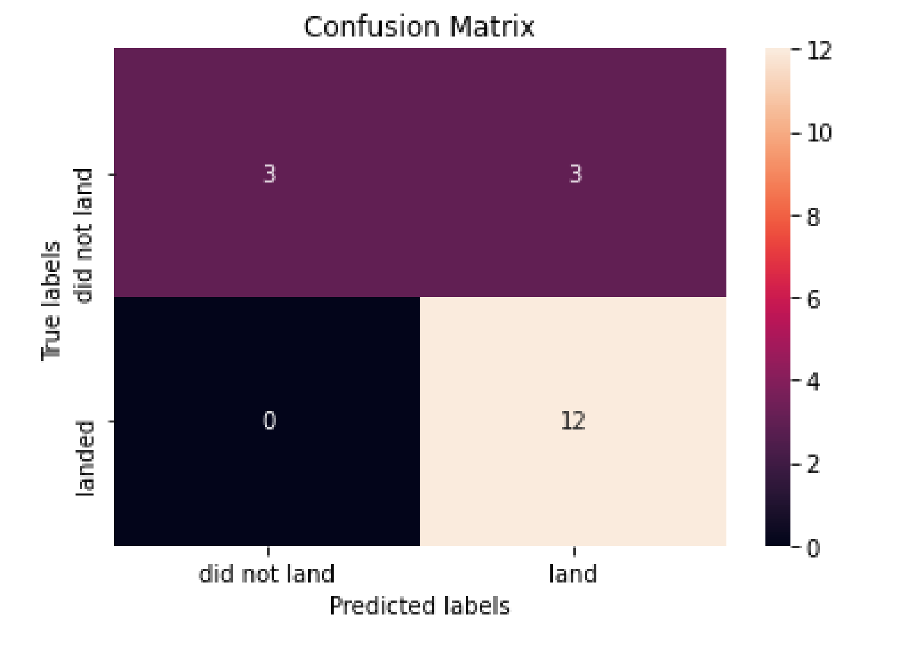
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_spli
t': 5, 'splitter': 'random'}
```

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.

Thank you!