IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Raúl García Calleja
9th of March 2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection adapting formats with an API
  - Data Collection with web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis results
  - Interactive Dashboard
  - Predictive Analysis models and results.

Introduction

- Project background and context

  The commercial space age is here, companies are making space travel affordable for everyone. 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 and based on that information to be more competitive.

- Problems you want to find answers
  - Which parameters and factors determine if the first stage rocket will land successfully?
  - Which are the effects between these parameters and factors affecting final result?
  - Which is the best machine learning model to predict successful landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected from Wikipedia, cleaned and adapted with SpaceX API
- Perform data wrangling
- 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

SpaceX API

Web Scrapping

Data Collection – SpaceX API

1.-Request and parse the SpaceX launch data using the GET request & data normalization

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

```
JsonList=requests.get(static_json_url).json()
data=pd.json_normalize(JsonList)
```

2.- Call API to get information about the launches using the IDs given for each launch

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that have multiple payloads in a single rocket.
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

3.- Construct data frame

```
# Create a data from launch_dict
data_falcon9 = pd.DataFrame(launch_dict)
```

You can see full details in following GitHub URL Link → https://github.com/raugaca/Data-Science-Capstone---SpaceX/blob/master/1.1.-Data%20Collection%20API%20Lab.ipynb

Data Collection - Scraping

1.-Request data from html page and create BeautifulSoup object

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

→

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup2 = BeautifulSoup(html_data.text, 'html.parser')
```

2.- Extracting table and column names

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

→

```
ths = first_launch_table.find_all('th')
for th in ths:
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0:
        column_names.append(name)
```

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

```
# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

→

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup2.find_all('table',"wikitable plainrowhea
  # get table row
  for rows in table.find_all("tr"):
    #check to see if first table heading is as number corresponding to launch a n
    if rows.th:
      if rows.th.string:
        flight_number=rows.th.string.strip()
        flag=flight_number.isdigit()
    else:
      flag=False
    #get table element
```

→

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

You can see full details in following GitHub URL Link →

https://github.com/raugaca/Data-Science-Capstone---SpaceX/blob/master/1.2.-%20Data%20Collection%20with%20Web%20Scraping.ipynb

Data Wrangling

1.-Calculate the number of launches on each site

```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

2.- Calculate the number and occurrence of each orbit

```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

3.- Calculate the number and occurence of mission outcome per orbit type

```
# landing_outcomes = values on Outcome column
landing_outcomes=df['Outcome'].value_counts()

for i,outcome in enumerate(landing_outcomes.keys()):
  print(i,outcome)
```

→

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

4.- Create a landing outcome label from Outcome column

```
landing_class=[]
Otcms=df['Outcome']

for i,Otcm in enumerate(Otcms):
  if Otcm in bad_outcomes:
    landing_class.append(0)
  else:
    landing_class.append(1)
```

→

```
df['Class']=landing_class
df[['Class']].head(8)
```

You can see full details in following GitHub URL Link → https://github.com/raugaca/Data-Science-Capstone---SpaceX/blob/master/1.3.-%20Data%20Wrangling.ipynb

EDA with Data Visualization

Scatter Plots

Scatter plots are showing there relationship between different parameters that are linked to launch success. The ones drawn in this study are as follows:

- Flight Number vs. Payload Mass
- Flight Number vs. Launch
- Payload and Launch Site
- Flight Number and Orbit type
- Payload and Orbit type

Bar Chart

Bar chart was used to determine the **Success rate vs. Orbit location**, showing which orbits were having full success rates.

Line Plot

Lines Plot was used to determine the relation between success rates and launch year. As a conclusion, we can see that from 2013 an uptrend success.

You can see full details in following GitHub URL Link → https://github.com/raugaca/Data-Science-Capstone---SpaceX/blob/master/2.2.1%20Exploratory%20Data%20Analysis%20Data%20Visualization.ipynb

EDA with SQL

SQL queries performed during this lab were as follows:

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

You can see full details in following GitHub URL Link → https://github.com/raugaca/Data-Science-Capstone---SpaceX/blob/master/2.1.-%20Exploratory%20Data%20Analysis.ipynb

Build an Interactive Map with Folium

We include some map objects map objects such as markers, circles, lines, etc.

- Generate a map marking launching sites.
- Generate a map showing launching site clusters
- Generate a map showing all launching labelling in green if succeed and in red if failed.
- Generate a map with the distances between a launch site to its proximities

These maps were useful to check following information:

- Launching sites geographical location
- Launching sites proximity to coast and/or Equator line
- Have a realistic overview of successful rates of launching sites.
- Calculate the distances between a launch site to its proximities such as Highways, Railways and cities.

You can see full details in following GitHub URL Link → https://github.com/raugaca/Data-Science-Capstone---SpaceX/blob/master/3.1.-%20Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

Build a Dashboard with Plotly Dash

We built an interactive Dashboard with Plotly Dash, that easily shows launch site success rates and relation between Payload, launch location and success rate.

- Pie chart: In the same figure we can filter through a dropdown menu the different launching locations and see the proportion between success and fail. By selecting "All Sites" we can see success % for each launching site vs total.
- Scatter Plot: it shows the relation between launch location and success rates, but additionally you can filter different Payload mass ranges in order to see impact on the other two parameters.

You can see full details in following GitHub URL Link → https://github.com/raugaca/Data-Science-Capstone---SpaceX/tree/master/Dashboard

Predictive Analysis (Classification)

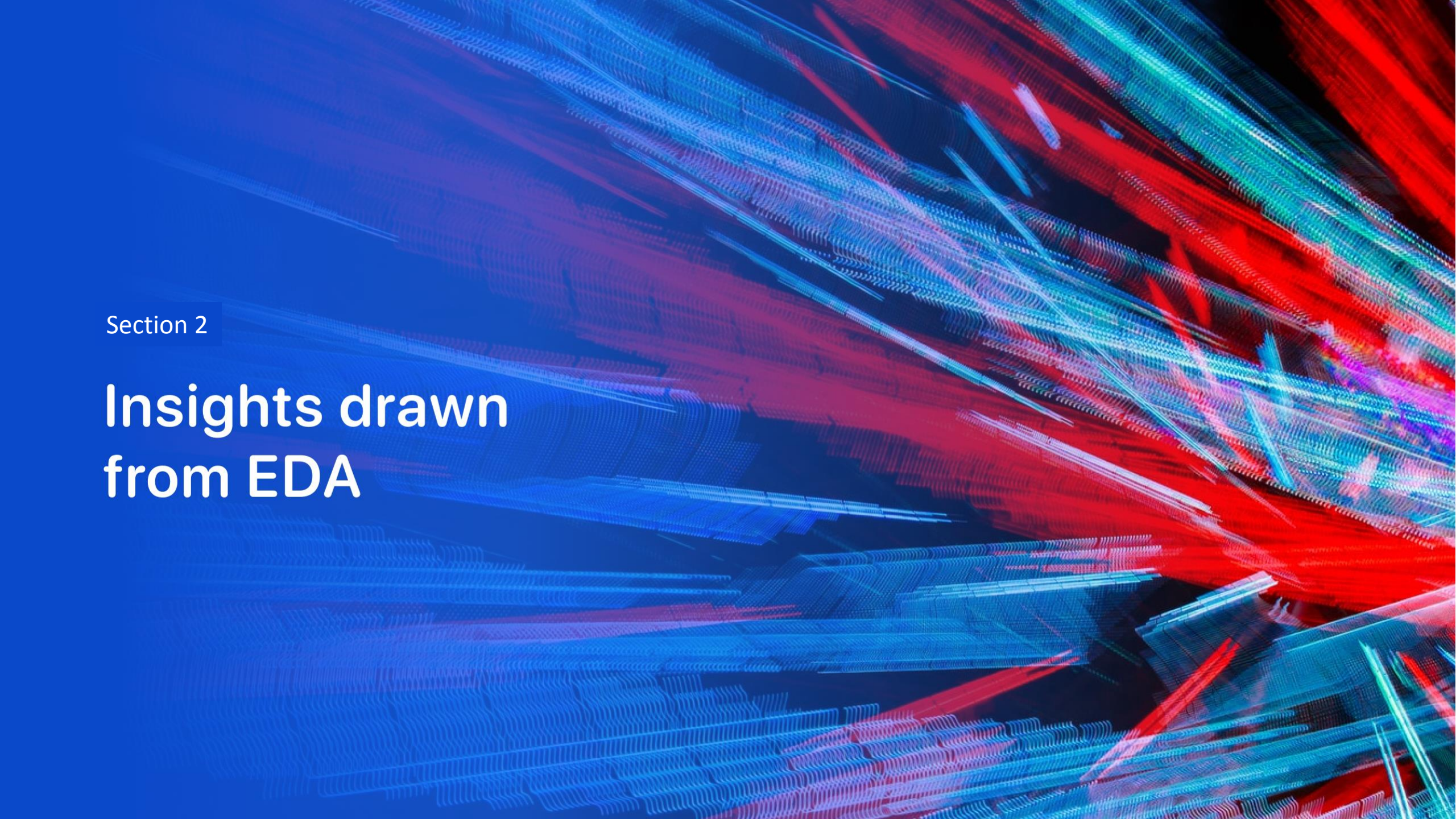
Predictive Analysis process follows this schematic process:

- Load the dataframe
- Create a column for the class which shows if launch was successful(1) or failed(0)
- Standardize the data
- Split into training data and test data
- Find best Hyperparameter for SVM, Classification Trees, Logistic Regression and KNN
- Find the method performs best using Confusion Matrix and Testing the data

You can see full details in following GitHub URL Link → https://github.com/raugaca/Data-Science-Capstone---SpaceX/blob/master/4.1.-%20Machine%20Learning%20Prediction.ipynb

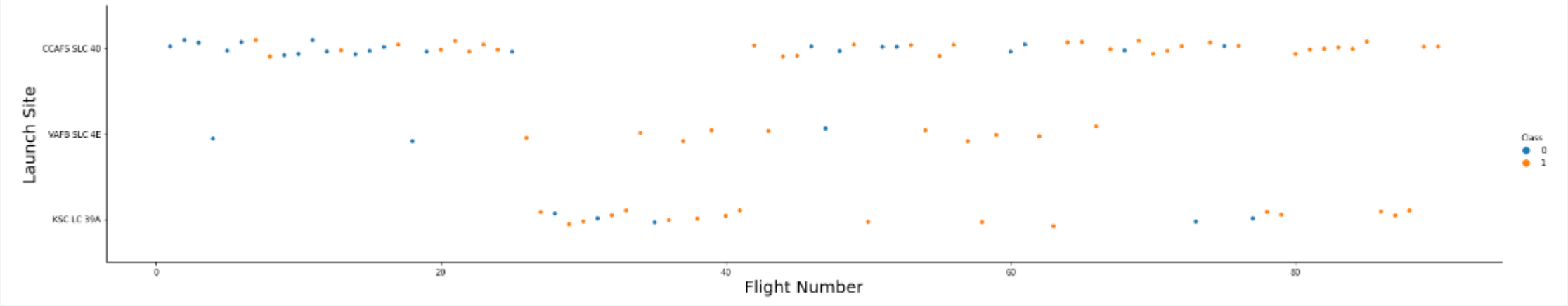
Results

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


Section 2
Insights drawn from EDA

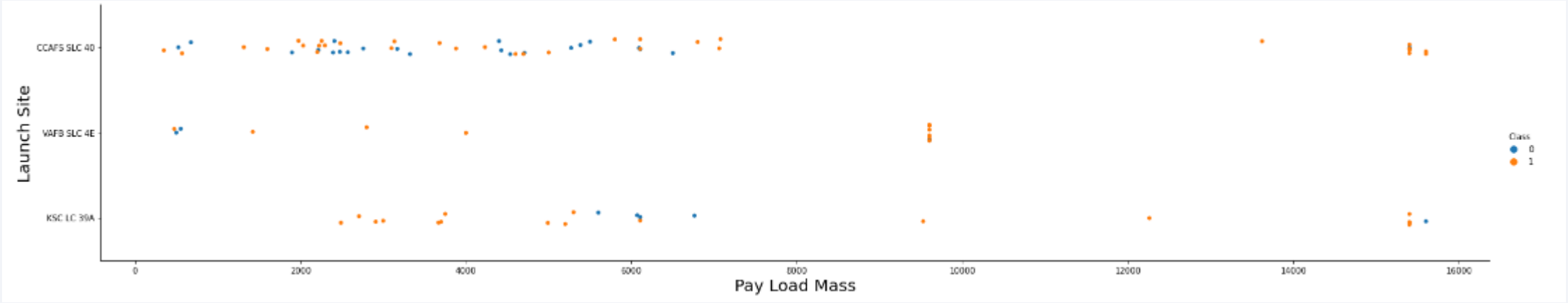
Flight Number vs. Launch Site

- The basic conclution that we get from this plot is that landing success increases as flight number increase for all different sites.



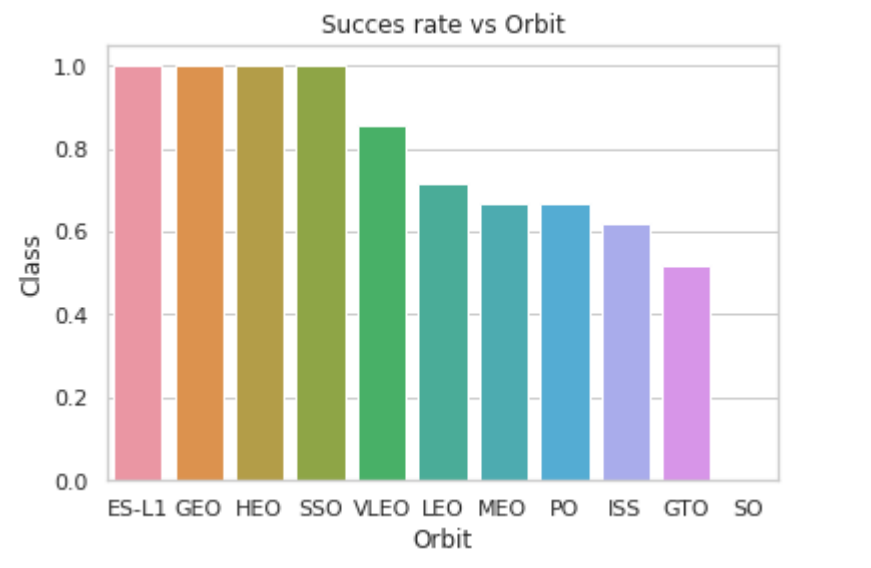
Payload vs. Launch Site

- Almost all filghts succed when heavyload mass is greater tan 10000 kg
- For VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000 kg)



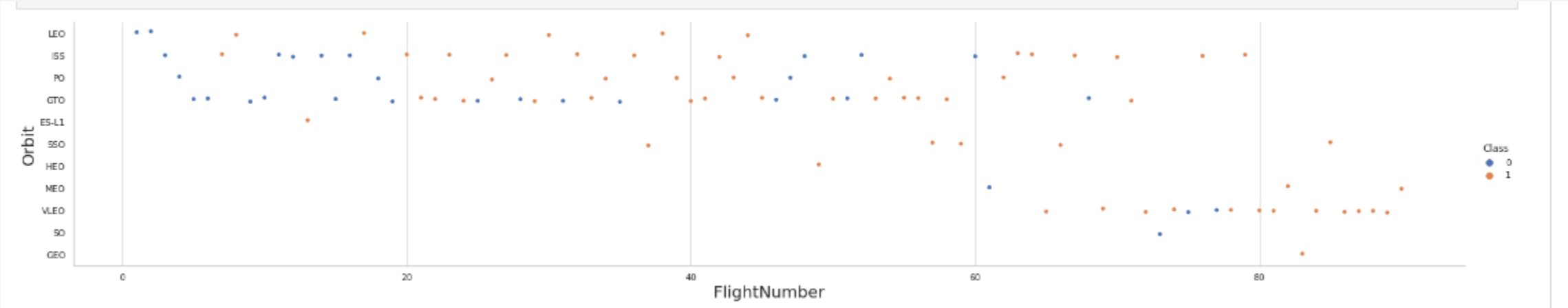
Success Rate vs. Orbit Type

- Orbits: ES-L1, GEO, HEO and SSO have 100% success rate
- SO orbirt has 0% success rate



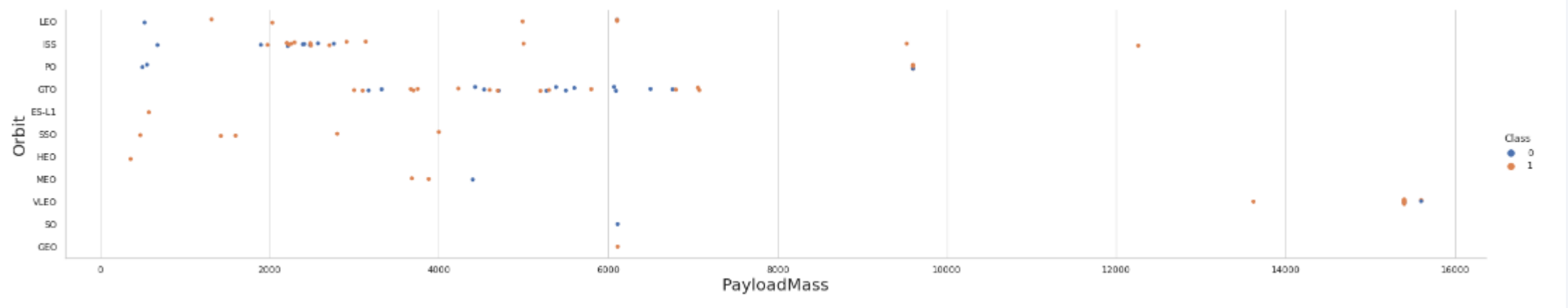
Flight Number vs. Orbit Type

- Initial flights used orbits with lower success rates and last flights are using differnt orbit types with higher success rates.



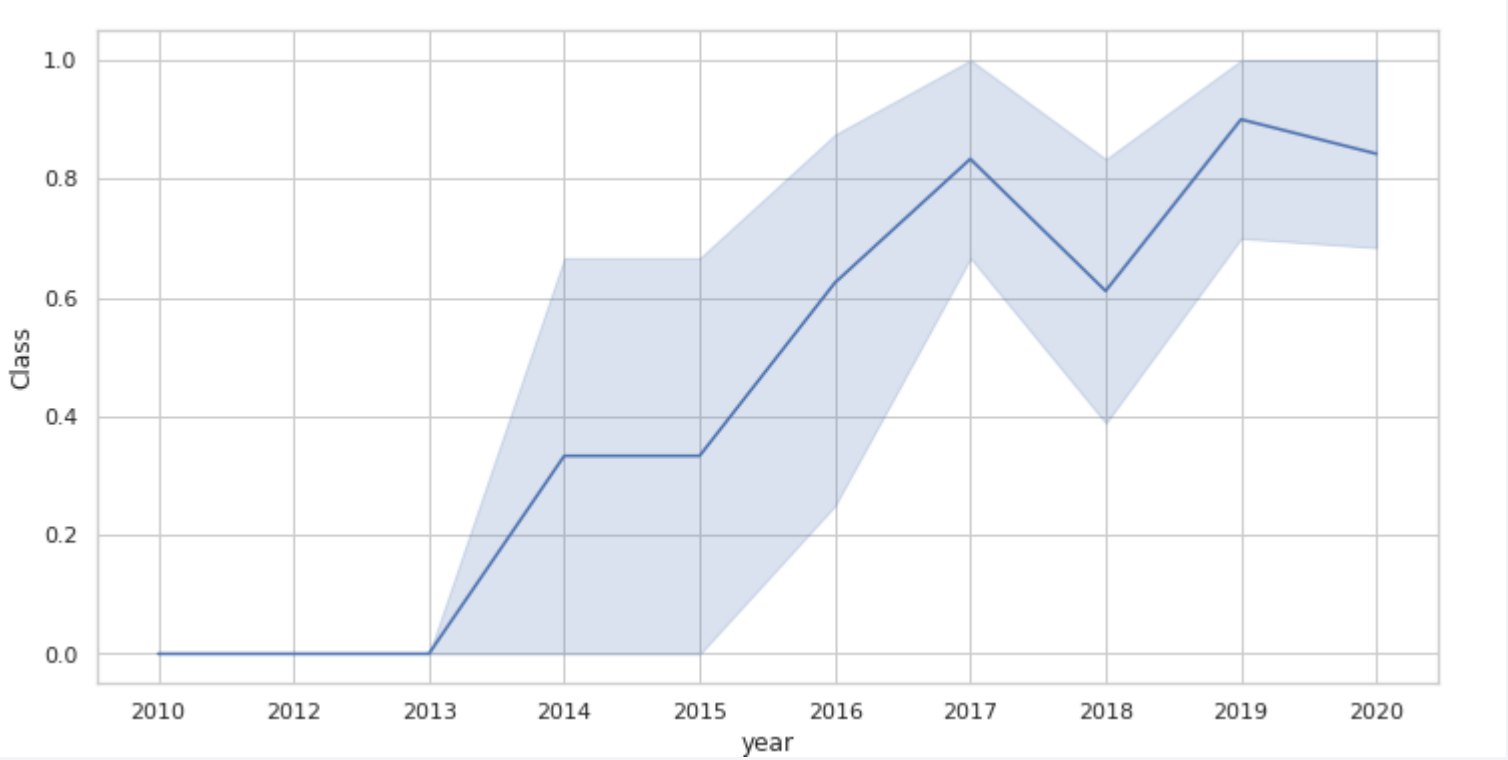
Payload vs. Orbit Type

- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS
- However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.



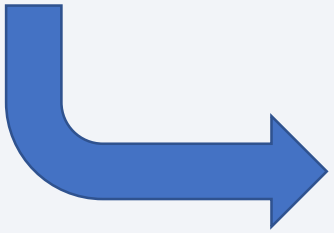
Launch Success Yearly Trend

- We can see in this grafh that since 2013 success reate was increasing (except a puntual decrease in 2018)



All Launch Site Names

- We get all launch site names with SQL command DISTINCT from the SpaceX dataframe.



```
%sql SELECT distinct(launch_site) FROM SPACEXTBL
```

```
 * ibm_db_sa://jxm11796:***@ea286ace-86c7-4d5b-8580-3fbfa
Done.
```

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

Launch Site Names Begin with 'CCA'

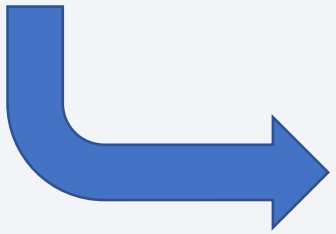
- Key instructions in SQL are "LIKE 'CCA%'" to get all launch site names beginning with 'CCA' and "LIMIT 5" to show only the first 5 records.

```
%%sql
select *
FROM SPACEXTBL
WHERE launch_site
LIKE 'CCA%' LIMIT 5
```

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 (NASA only)

We get all launch site names with SQL command SUM applied to SpaceX dataframe customer NASA (CRS)



```
%%sql
select SUM(payload_mass__kg_)
FROM SPACEXTBL
WHERE customer = 'NASA (CRS)'
```

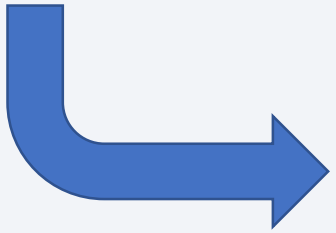
```
 * ibm_db_sa://jxm11796:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90108kqb1od8lcg.databases.appdomain.cloud:31505/BLUDB
Done.
```

5]:

1
45596

Average Payload Mass by F9 v1.1

We Average Payload Mass of F9 v1.1 flight with SQL command **AVG** applied to SpaceX dataframe booster verision F9 v1.1



```
%%sql
SELECT AVG(payload_mass__kg_) as Average_Payload_Carried
FROM SPACEXTBL
WHERE booster_version='F9 v1.1'
```

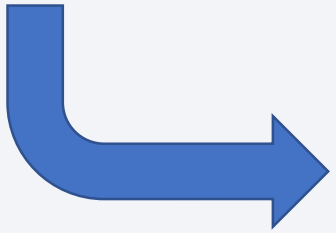
```
 * ibm_db_sa://jxm11796:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90108kqb1od8lcg.databases.appdomain.cloud:31505/BLUDB
Done.
```

9]:

average_payload_carried
2928

First Successful Ground Landing Date

Key commands for First success ful landing date are **LIKE 'Success'** and **ORDER BY date ASC LIMIT1**, to make sure that we get first result date.
v1.1



```
%%sql
SELECT date as First_successful_landing_outcome
FROM SPACEXTBL
WHERE mission_outcome LIKE 'Success'  ORDER BY date ASC LIMIT 1
```

```
 * ibm_db_sa://jxm11796:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90108kqb1od8lcg.databases.appdomain.cloud:31505/BLUDB
Done.
```

]:

first_successful_landing_outcome
2010-06-04

Successful Drone Ship Landing with Payload between 4000 and 6000

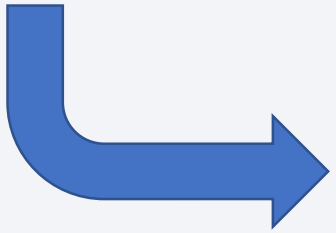
- Key instructions in SQL to ensure that we are filtering Success missions and the Payload is between 4000 kg and 6000 kg

```
%%sql
SELECT  distinct(booster_version)
FROM SPACEXTBL
WHERE mission_outcome LIKE 'Success'  AND payload_mass__kg_>4000 AND payload_mass__kg_<6000
```

booster_version
F9 B4 B1040.2
F9 B4 B1040.1
F9 B5 B1046.2
F9 B5 B1047.2
F9 B5 B1048.3
F9 B5 B1051.2
F9 B5 B1058.2
F9 B5B1054
F9 B5B1060.1
F9 B5B1062.1
F9 FT B1021.2
F9 FT B1031.2
F9 FT B1032.2
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1030
F9 FT B1032.1
F9 v1.1
F9 v1.1 B1011
F9 v1.1 B1014
F9 v1.1 B1016

Total Number of Successful and Failure Mission Outcomes

Key SQL commands are **COUNT** and **GROUP BY** to get totals for different mission outcome categories



```
%%sql
SELECT  mission_outcome, COUNT(mission_outcome)
FROM spacextbl
GROUP BY mission_outcome
```

```
 * ibm_db_sa://jxm11796:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/BLUDB
Done.
```

1]:

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

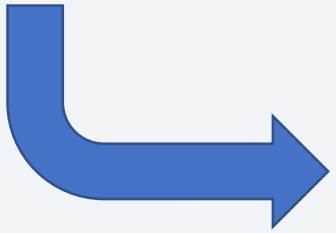
- Key instructions in SQL to include a subquery to get maximum Payload and then filter the Booster versions

```
%%sql
SELECT  booster_version, payload_mass__kg_
FROM SPACEXTBL
WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL)
```

booster_version
F9 B4 B1040.2
F9 B4 B1040.1
F9 B5 B1046.2
F9 B5 B1047.2
F9 B5 B1048.3
F9 B5 B1051.2
F9 B5 B1058.2
F9 B5B1054
F9 B5B1060.1
F9 B5B1062.1
F9 FT B1021.2
F9 FT B1031.2
F9 FT B1032.2
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1030
F9 FT B1032.1
F9 v1.1
F9 v1.1 B1011
F9 v1.1 B1014
F9 v1.1 B1016

2015 Launch Records (failed landing in drone)

Key commands are for filtering required information: YEAR(date)=2015 to get required year and **landing outcome = 'Failure (drone ship)'**



```
%%sql
SELECT landing__outcome, booster_version, launch_site, date
FROM SPACEXTBL
WHERE (YEAR(date)=2015 AND landing__outcome = 'Failure (drone ship)')
```

```
 * ibm_db_sa://jxm11796:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90108kqb1od8lcg.databases.appdomain.cloud:31505/BLUDB
Done.
```

]:

landing__outcome	booster_version	launch_site	DATE
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40	2015-01-10
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40	2015-04-14

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

- We 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 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

```
%%sql
SELECT landing__outcome, COUNT(landing__outcome)
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY landing__outcome
ORDER BY COUNT(landing__outcome) DESC
```

landing__outcome	2
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

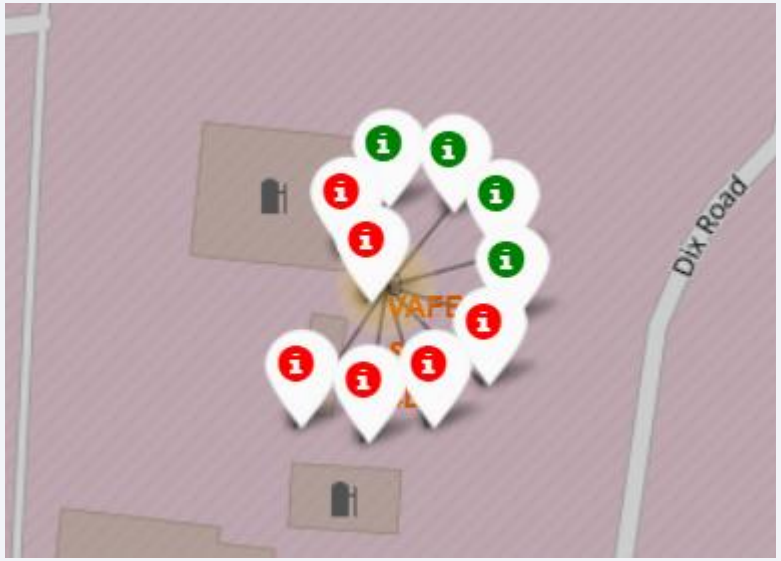
Launch sites location map

- Launch sites are concentrated in two areas: California (1 location) and Florida (3 locations)

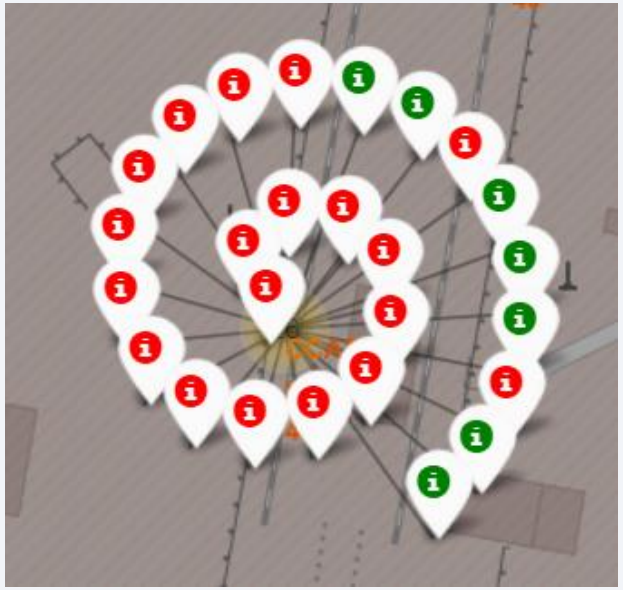
Launching summary details

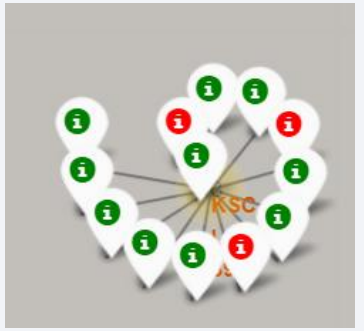
- Green labels represent success launch and read labels represent failed launches.

California



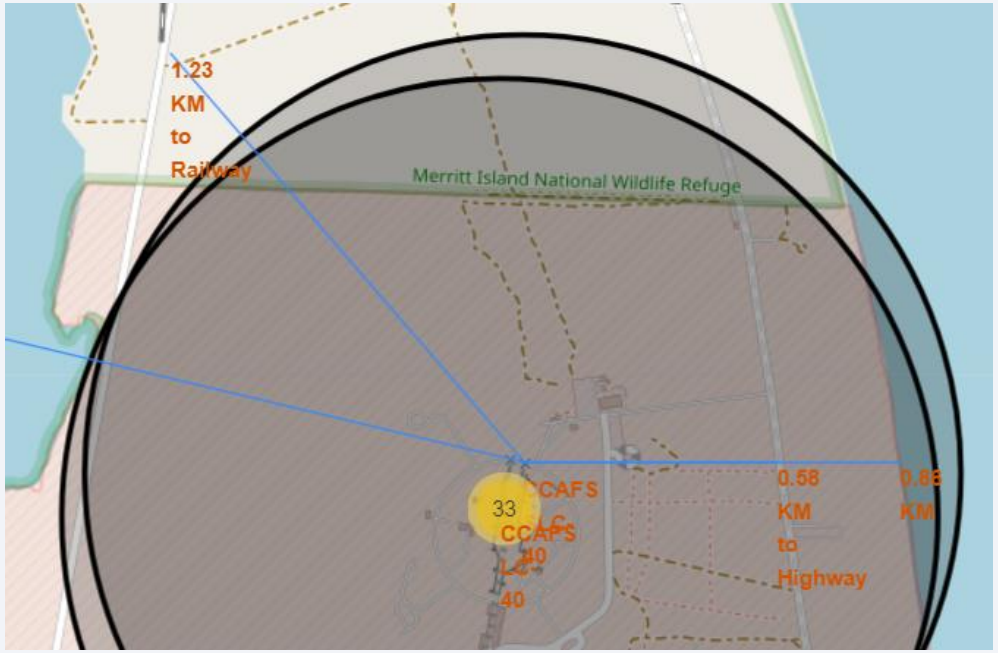
Florida

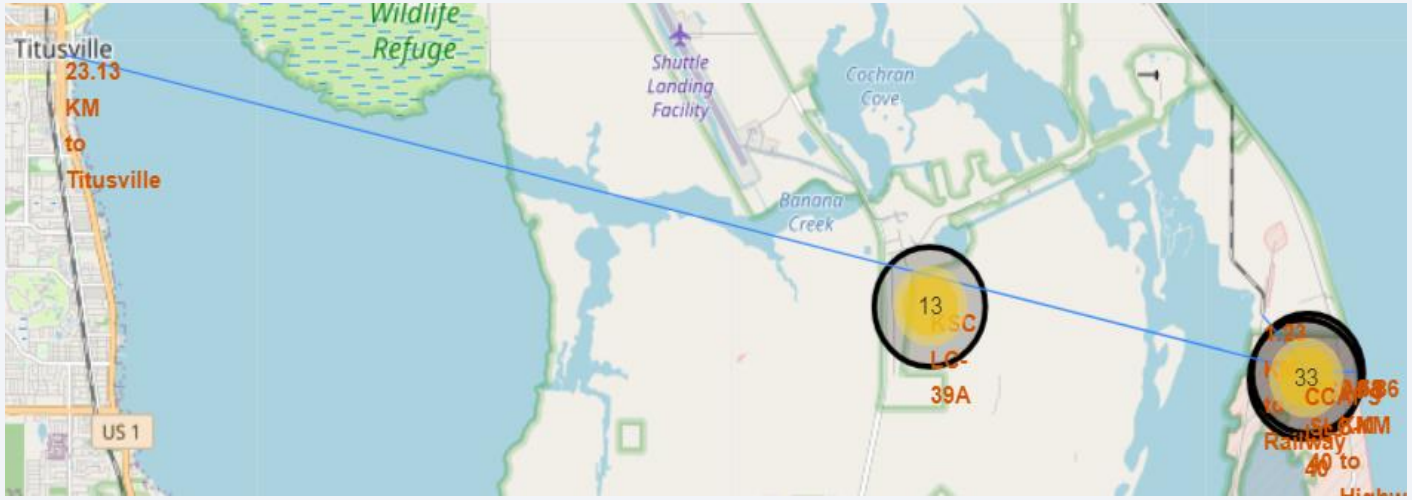


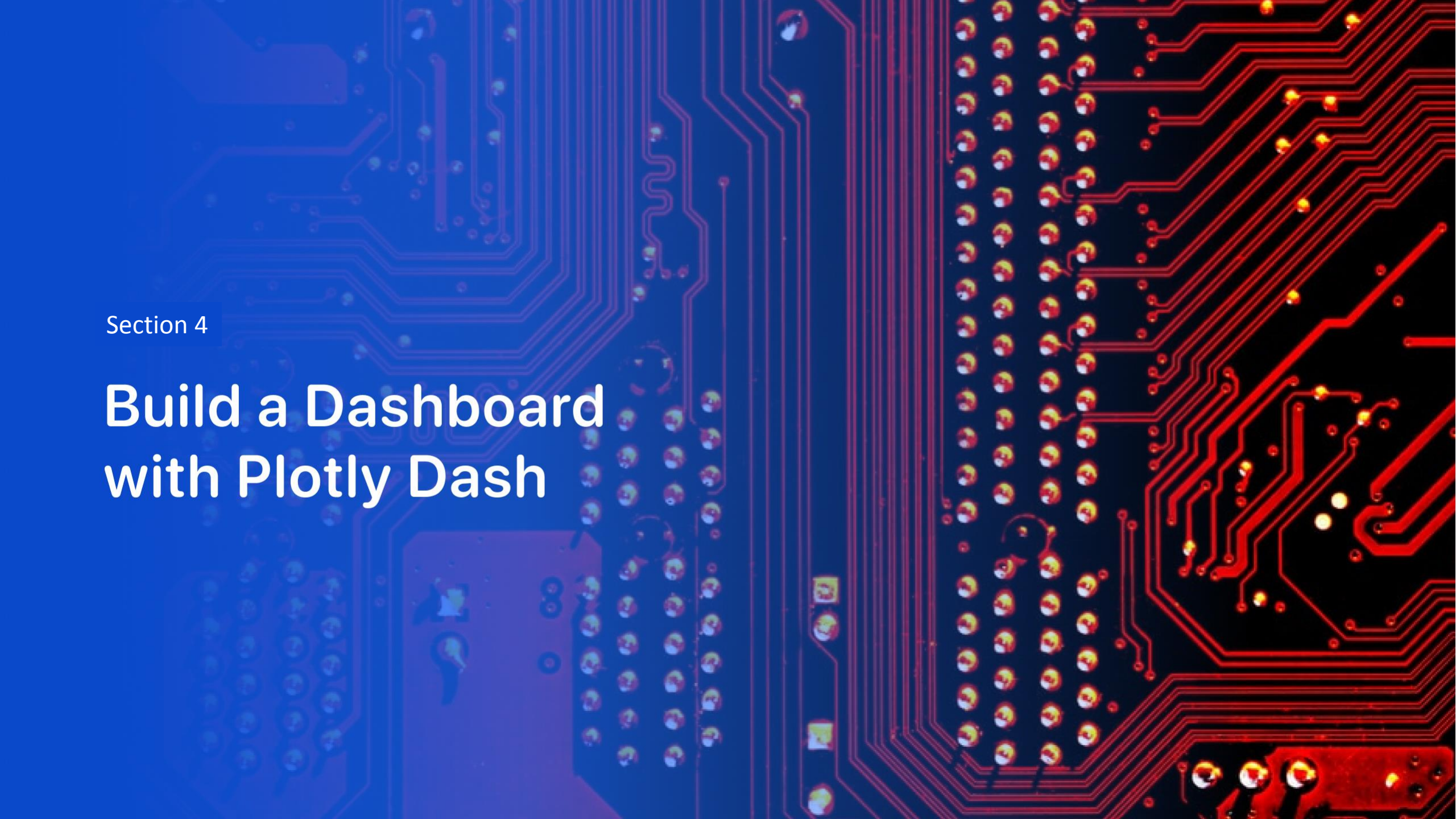


CCAPS SLC launch location key distances

- Distance between CCAPS SLC-40 and closest coastline, Highway, Railway (less than 1.5 Km) and Titusville city (more than 23 km)

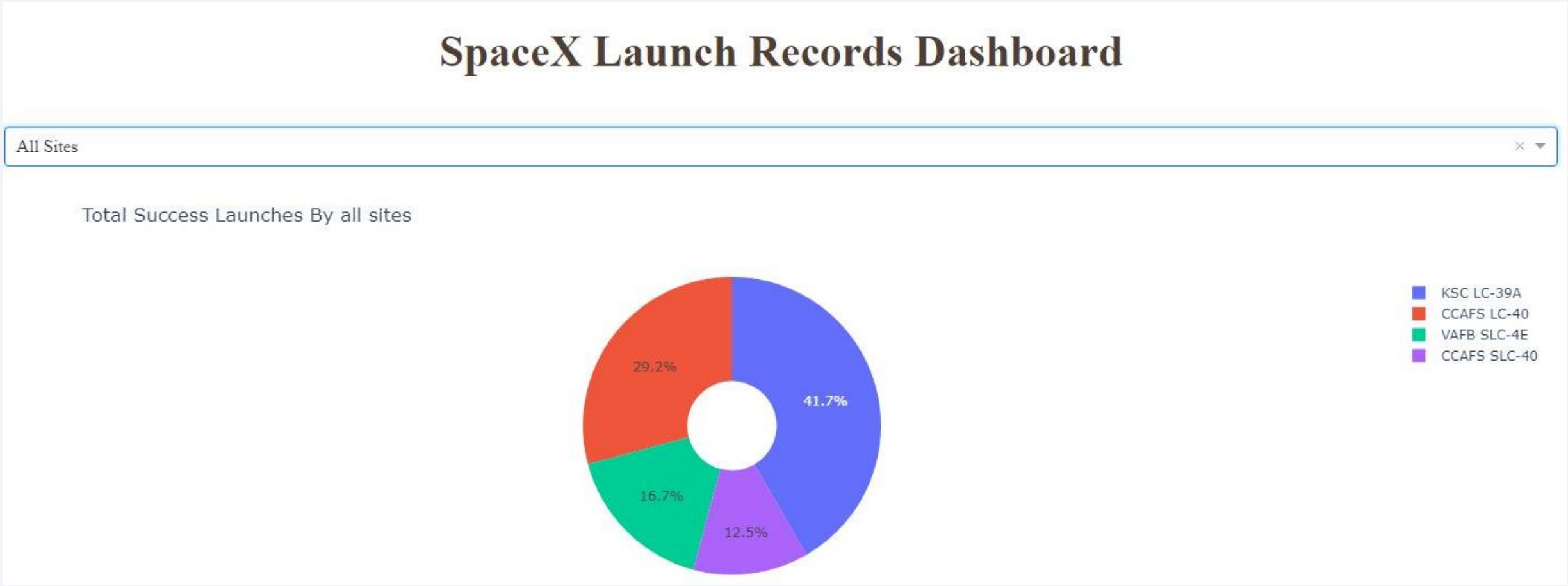





Section 4
Build a Dashboard with Plotly Dash

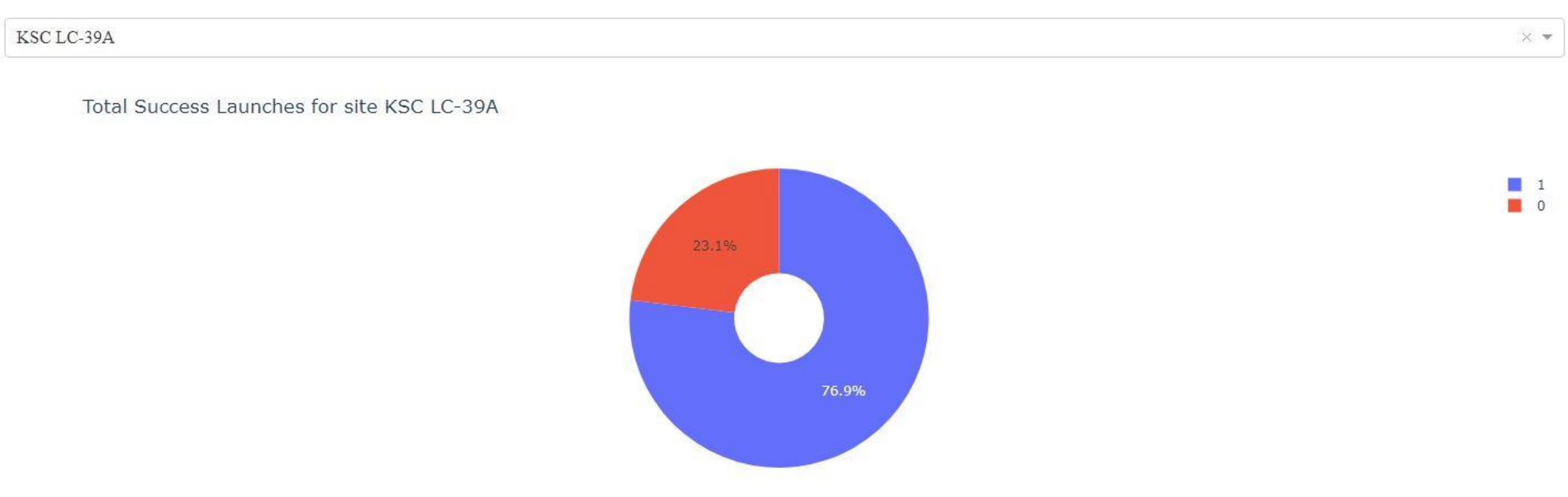
Launch success piechart for all sites

- Dashboard is showing that KSC LC-39A is the launch site with most success from all sites



Launch site with the highest success rate

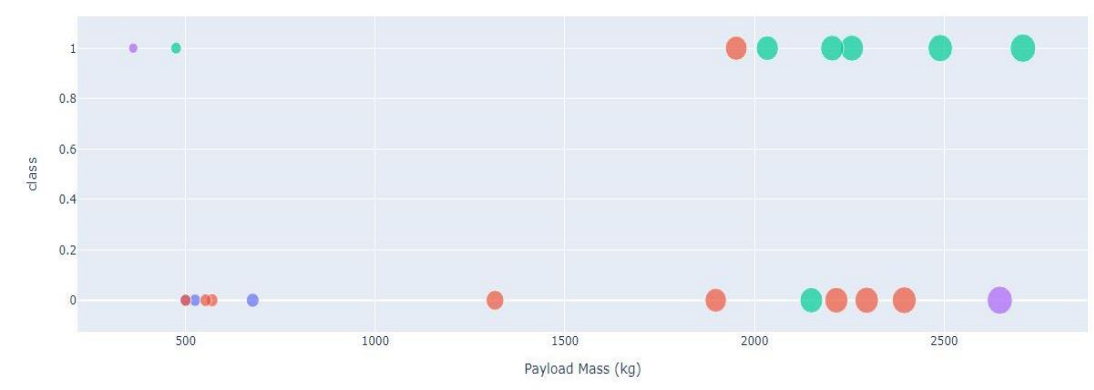
- KSC LC-39A is the launch site with the highest success rate with 76,9%



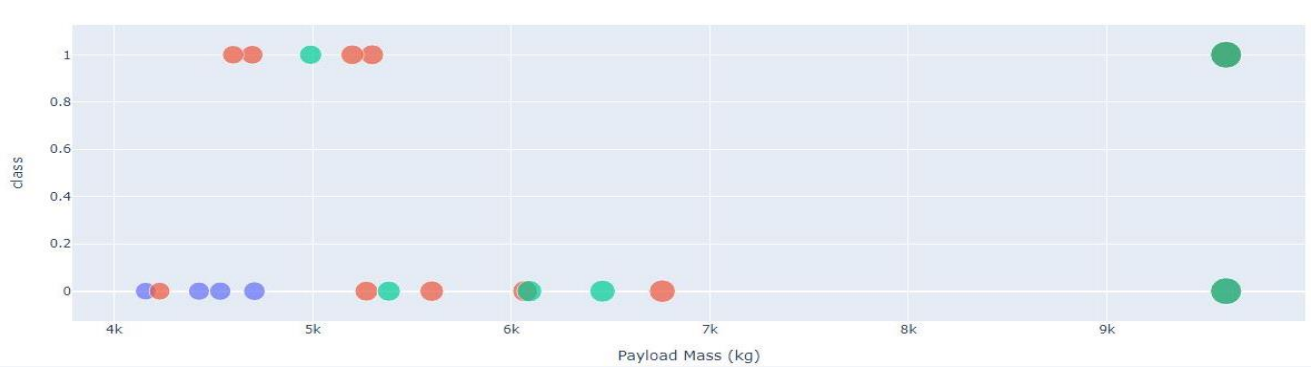
Payload vs. Launch Outcome scatter plot

- Low Payload range has higher success rate than high Payload rate

Payload 0-3000Kg



Payload 4000-10000Kg



Section 5

Predictive Analysis (Classification)

Classification Accuracy

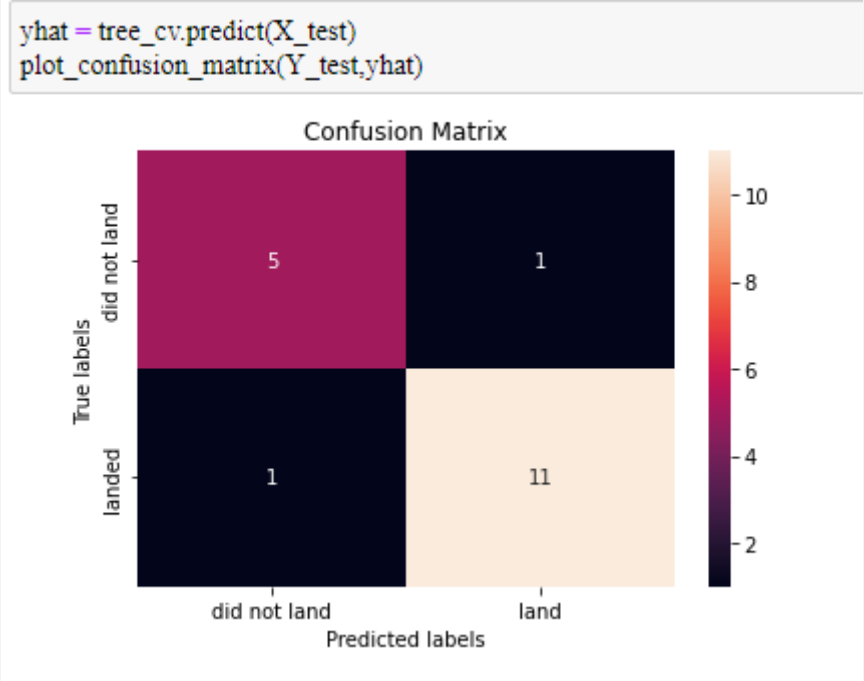
- Decision Tree has the highest accuracy for Training model and Test results

Best Accuracy Logistic Regression: 0.8464 --- Test Accuracy Logistic Regression: 0.8333
Best Accuracy SVM: 0.8482 --- Test Accuracy SVM: 0.8333
Best Accuracy Tree: 0.8857 --- Test Accuracy Tree: 0.8889
Best Accuracy KNN: 0.8482 --- Test Accuracy KNN: 0.8333

Confusion Matrix – Decision Tree

- Confussion matrix is showing that the model is having the highest score for successful landing.

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



Conclusions

Main conclusions of this study are as follows:

- The Decision tree classifier is the best algorithm to predict future launch trials results.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO and SSO had the best success rate.
- KSC LC-39A had the most successful launches rates.
- Low weight payloads perform better than heavy payloads-

Thank you!