IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<EUN JEONG JI>
<08/15/2022>

Outline

Executive Summary

- Summary of methodologies
- - Data collection
- - Data wrangling
- - Exploratory data analysis (EDA) using visualization and SQL
- - Building interactive visual analytics using Folium and Plotly Dash
- - predictive analysis using classification models
- Summary of all results
- - EDA results
- - The interactive analytics results
- - The insight and prediction results

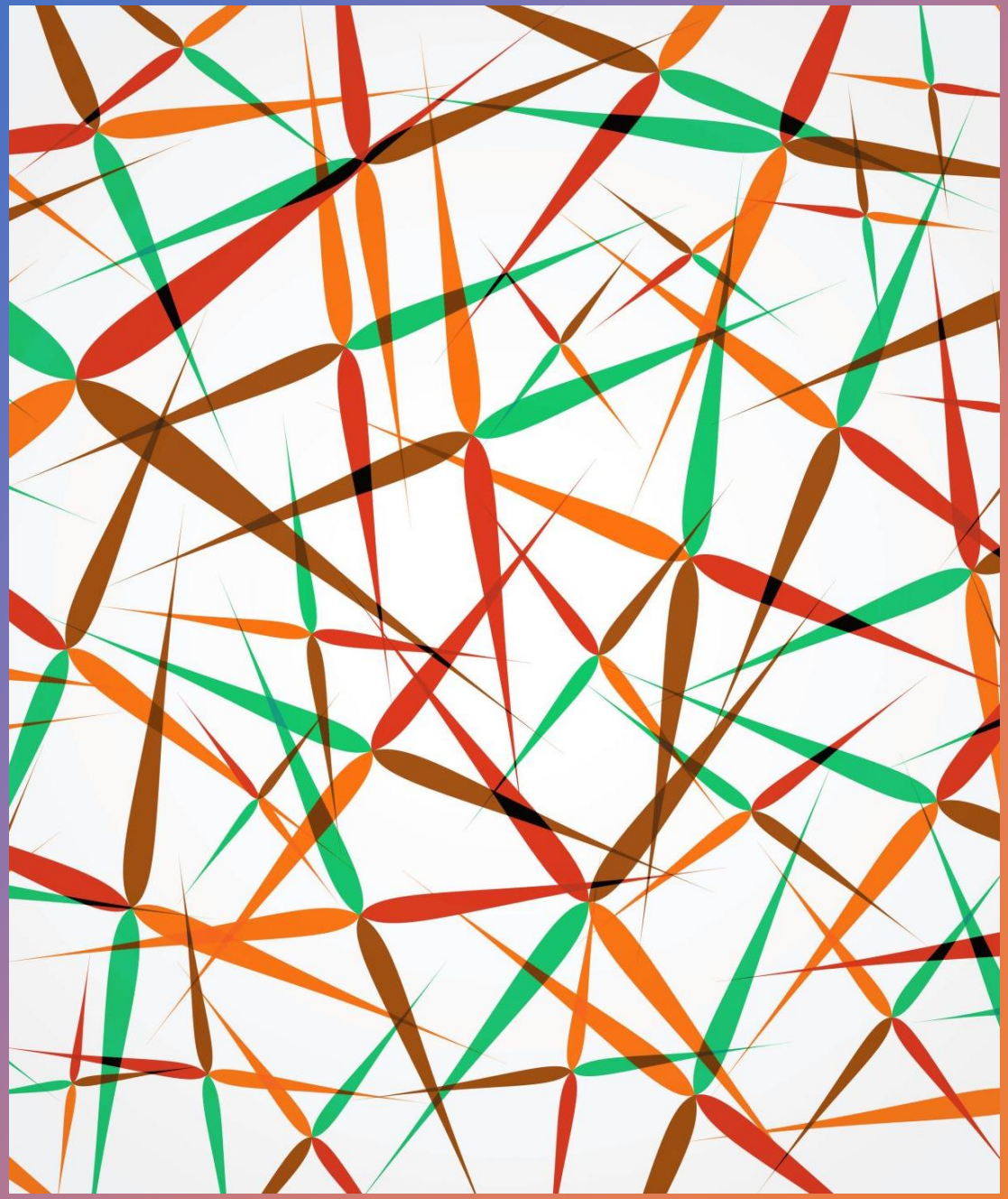
Introduction

- Project background and context
- Space X was able to reduce the cost a lot compare with other companies by reusing the rocket.
- Whether a rocket can be reused depends on the succeed of the land in the first stage.

- Problems you want to find answers
- 1. What factors can impact the successful landing of the Falcon 9
- 2. Whether the rocket land successfully?
- 3. what is the accuracy of successful landing?

Section 1

Methodology



Methodology

- Executive Summary
- Data collection methodology:
  - With Rest API and Web Scrapping
- Perform data wrangling
  - Data were transformed and one hot encoded to be deployed on the Machine Learning
  - Model and the unrelated data was removed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Various classification machine learning models like KNN or SVM were built to find out best classifier

Data Collection

- Describe how data sets were collected.

- Data sets were collected using the API call from several websites, I collected the core data from https://api.spacexdata.com/v4 website.
- The API gave us data about launches, including information about the rocket used, payload delivered, landing specification and landing outcome.
- Another popular data source for obtaining Falcon 9 Launch data is web scraping Wikipedia using Beautifulsoup.
-

Data Collection – SpaceX API

1. Collecting the data with API call

Now let's start requesting rocket launch data from SpaceX API with the following URL:

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

[15]: response = requests.get(spacex_url)
```

Check the content of the response

2. Converting to data frame with help of JSON

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

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

We should see that the request was successfull with the 200 status response code

```
[18]: response.status_code
[18]: 200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
[22]: # Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

```
[23]: # Get the head of the dataframe
data.head()
```

3. Data preprocessing

Then, we need to create a Pandas data frame from the dictionary launch_dict.

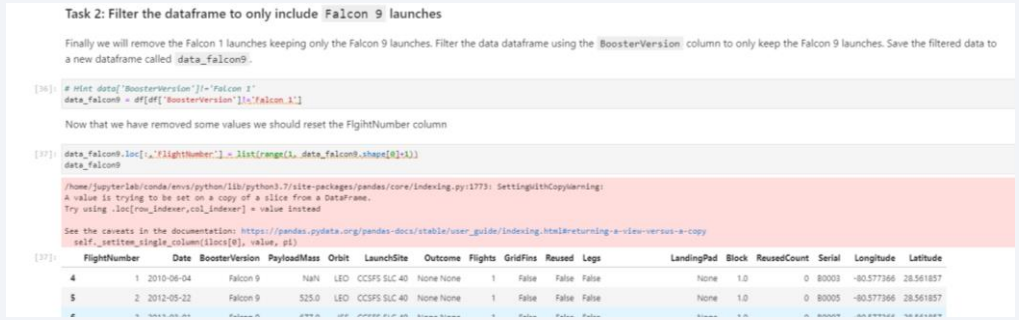
```
[34]: # Create a data from launch_dict
df = pd.DataFrame.from_dict(launch_dict)
```

Show the summary of the dataframe

```
[35]: # Show the head of the dataframe
df.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin1A	167.743129	9.047721
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2A	167.743129	9.047721
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2C	167.743129	9.047721
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin3C	167.743129	9.047721
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857

4. Filtering the data to keep only Falcon 9 launches



Task 2: Filter the dataframe to only include `Falcon 9` launches

Finally we will remove the Falcon 1 launches keeping only the Falcon 9 launches. Filter the data dataframe using the `BoosterVersion` column to only keep the Falcon 9 launches. Save the filtered data to a new dataframe called `data_falcon9`.

```
[36]: # Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = df[df['BoosterVersion']!='Falcon 1']
```

Now that we have removed some values we should reset the FlgihtNumber column

```
[37]: data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

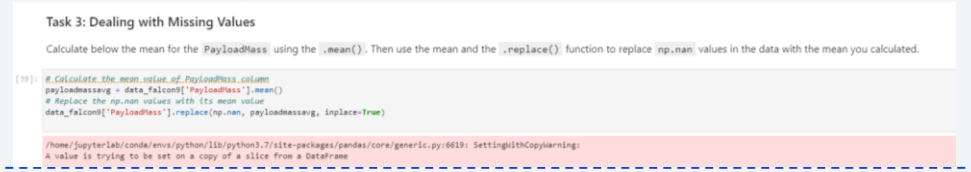
```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages/pandas/core/indexing.py:1773: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  self._setitem_single_column(ilocs[0], value, pi)
```

[37]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.561857

5. Converting the data to CSV file



Task 3: Dealing with Missing Values

Calculate below the mean for the `PayloadMass` using the `.mean()`. Then use the mean and the `.replace()` function to replace `np.nan` values in the data with the mean you calculated.

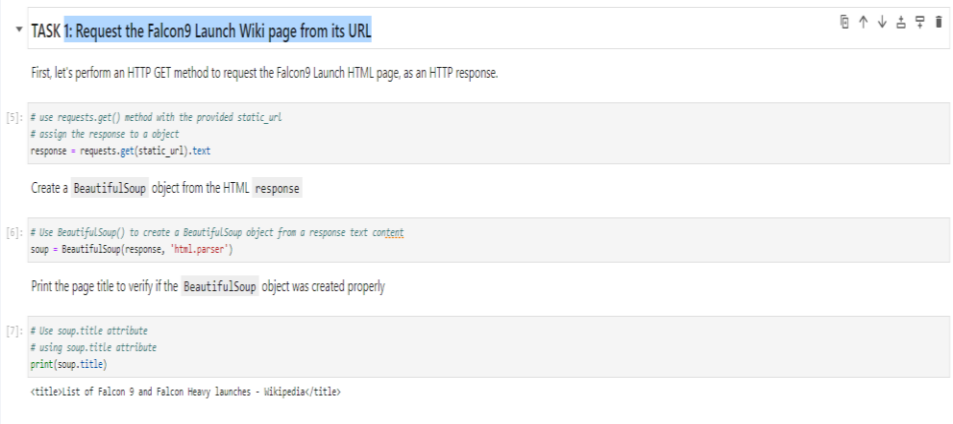
```
[39]: # Calculate the mean value of PayloadMass column
payloadmassavg = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].replace(np.nan, payloadmassavg, inplace=True)
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages/pandas/core/generic.py:6619: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

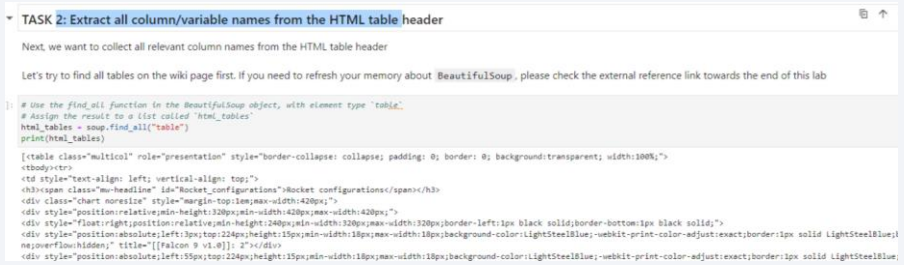
[The-Assignment-of-Cousera-/jupyter-labs-spacex-data-collection-api.ipynb at main · victory8230/The-Assignment-of-Cousera- (github.com)](https://github.com/victory8230/The-Assignment-of-Cousera-/blob/main/jupyter-labs-spacex-data-collection-api.ipynb)

Data Collection - Scraping

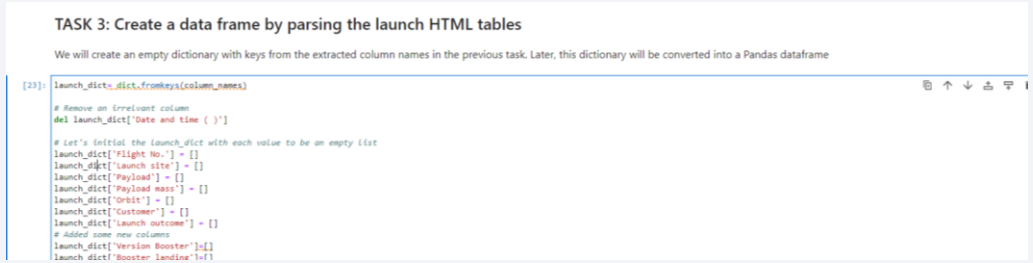
- **1: Request the Falcon9 Launch Wiki page from its URL**

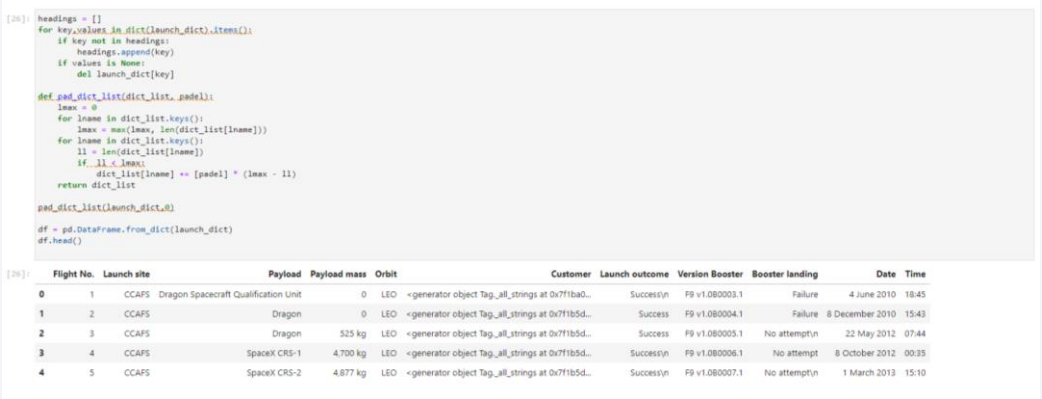


2: Extract all column/variable names from the HTML table



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





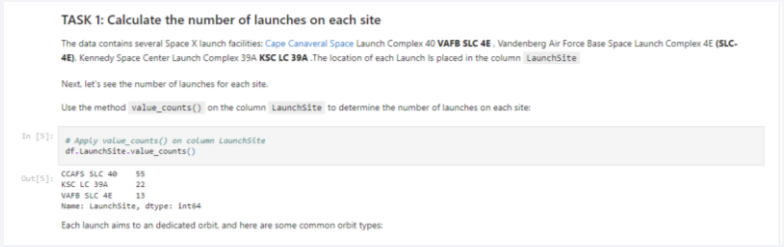
	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	<generator object Tag._all_strings at 0x7f1ba0...	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	<generator object Tag._all_strings at 0x7f1b5d...	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	<generator object Tag._all_strings at 0x7f1b5d...	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	<generator object Tag._all_strings at 0x7f1b5d...	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	<generator object Tag._all_strings at 0x7f1b5d...	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10

[The-Assignment-of-Cousera-/jupyter-labs-webscraping.ipynb at main · victory8230/The-Assignment-of-Cousera- (github.com)](https://github.com/victory8230/The-Assignment-of-Cousera-/blob/main/jupyter-labs-webscraping.ipynb)

Data Wrangling

- Exploratory Data Analysis Load Space X dataset

1: Calculate the number of launches on each site



TASK 1: Calculate the number of launches on each site

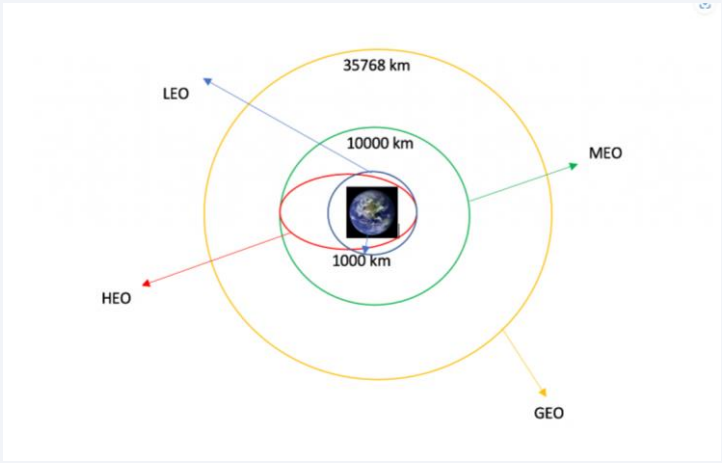
The data contains several Space X launch facilities: Cape Canaveral Space Launch Complex 40 **VAFB SLC 4E** , Vandenberg Air Force Base Space Launch Complex 4E **(SLC-4E)**, Kennedy Space Center Launch Complex 39A **KSC LC 39A** .The location of each Launch Is placed in the column `LaunchSite`

Next, let's see the number of launches for each site.

Use the method `value_counts()` on the column `LlaunchSite` to determine the number of launches on each site:

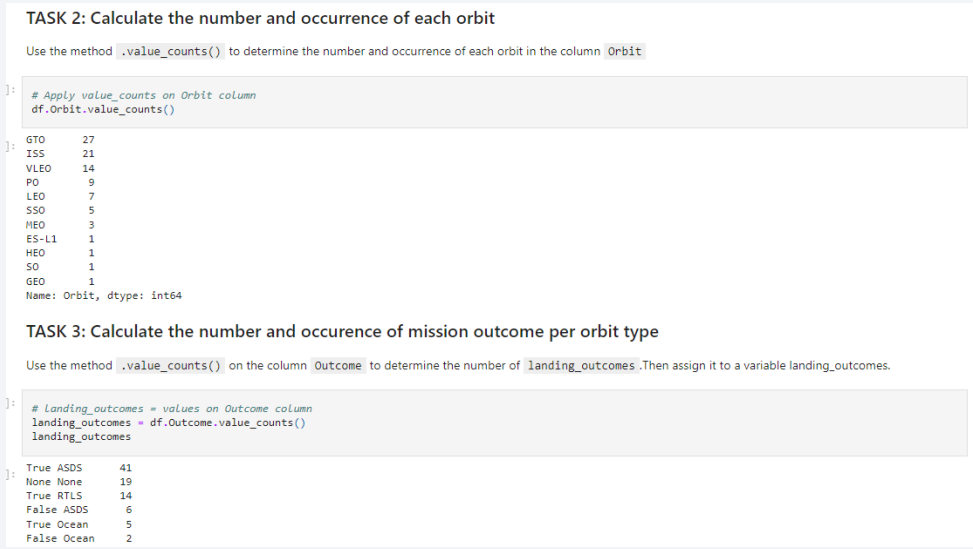
```
In [5]: # Apply value_counts() on column LaunchSite
        df.LaunchSite.value_counts()
Out[5]: CCAFS SLC 40    55
        KSC LC 39A      22
        VAFB SLC 4E     13
        Name: LaunchSite, dtype: int64
```

Each launch aims to an dedicated orbit, and here are some common orbit types:



[The-Assignment-of-Cousera-/labs-jupyter-spacex-Data wrangling (1).ipynb at main · victory8230/The-Assignment-of-Cousera- (github.com)]

2: Calculate the number and occurrence of each orbit and occurrence of mission outcome per orbit type



TASK 2: Calculate the number and occurrence of each orbit

Use the method `.value_counts()` to determine the number and occurrence of each orbit in the column `Orbit`

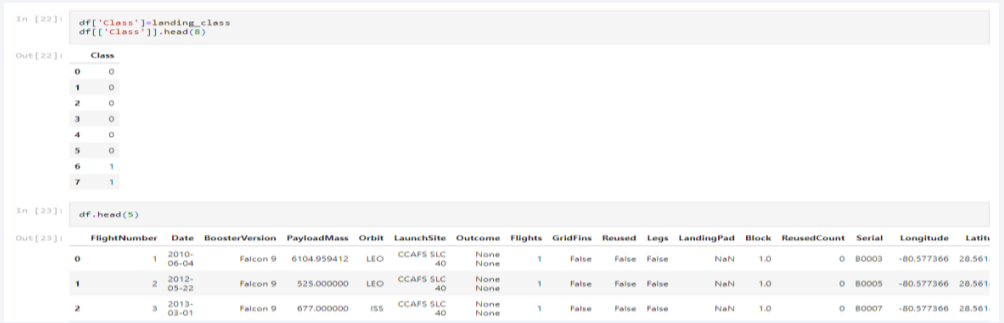
```
]: # Apply value_counts on Orbit column
   df.Orbit.value_counts()
]: GTO     27
   ISS     21
   VLEO    14
   PO       9
   LEO      7
   SSO      5
   MEO      3
   ES-L1    1
   HEO      1
   SO       1
   GEO      1
   Name: Orbit, dtype: int64
```

TASK 3: Calculate the number and occurence of mission outcome per orbit type

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outcomes` .Then assign it to a variable landing_outcomes.

```
]: # landing_outcomes = values on Outcome column
   landing_outcomes = df.Outcome.value_counts()
   landing_outcomes
]: True ASDS      41
   None None      19
   True RTLS      14
   False ASDS      6
   True Ocean      5
   False Ocean     2
```

3: Create a landing outcome label from Outcome column



```
In [22]: df['Class']=landing_class
         df[['Class']].head(8)
Out[22]:
```

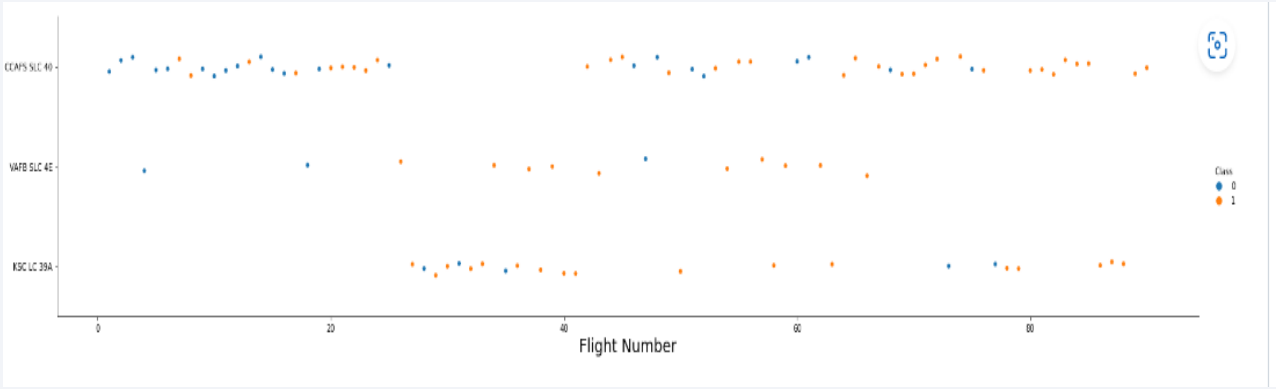
	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

```
In [23]: df.head(5)
Out[23]:
```

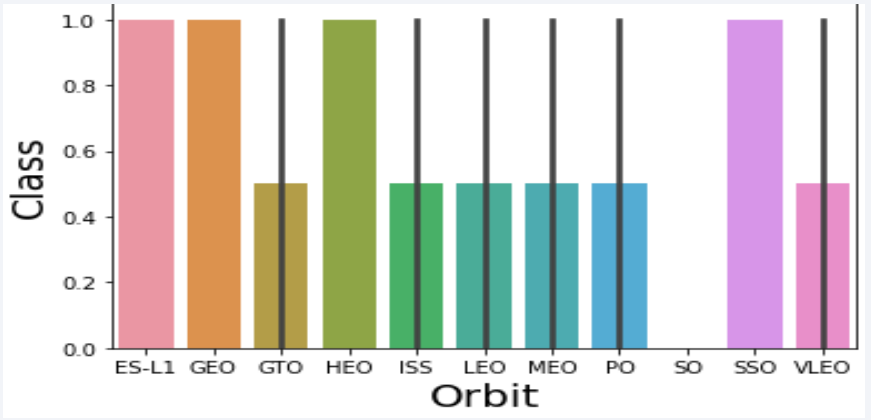
	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitu
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561

EDA with Data Visualization

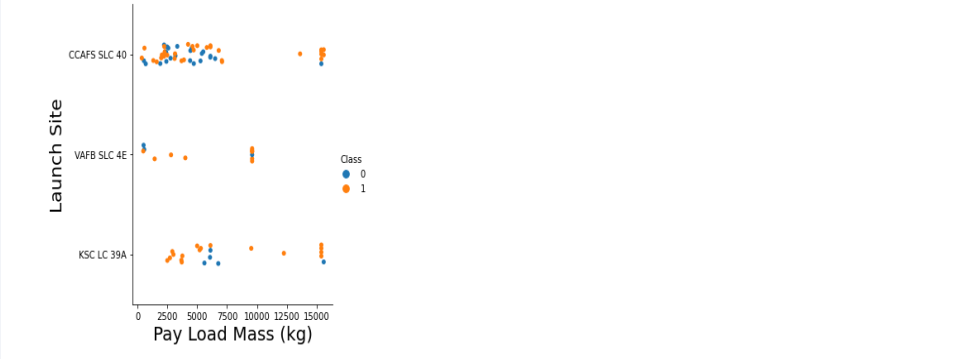
1: The relationship between Flight Number and Launch Site



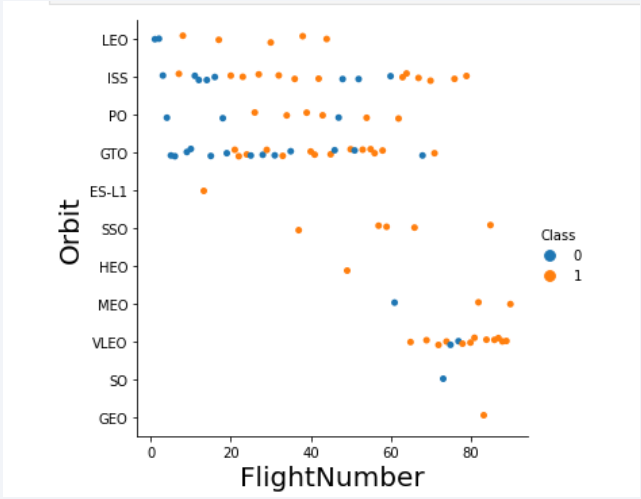
3: The relationship between success rate of each orbit type



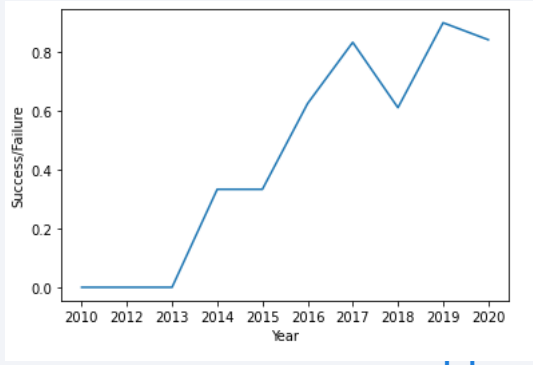
2. The relationship between Payload and Launch Site



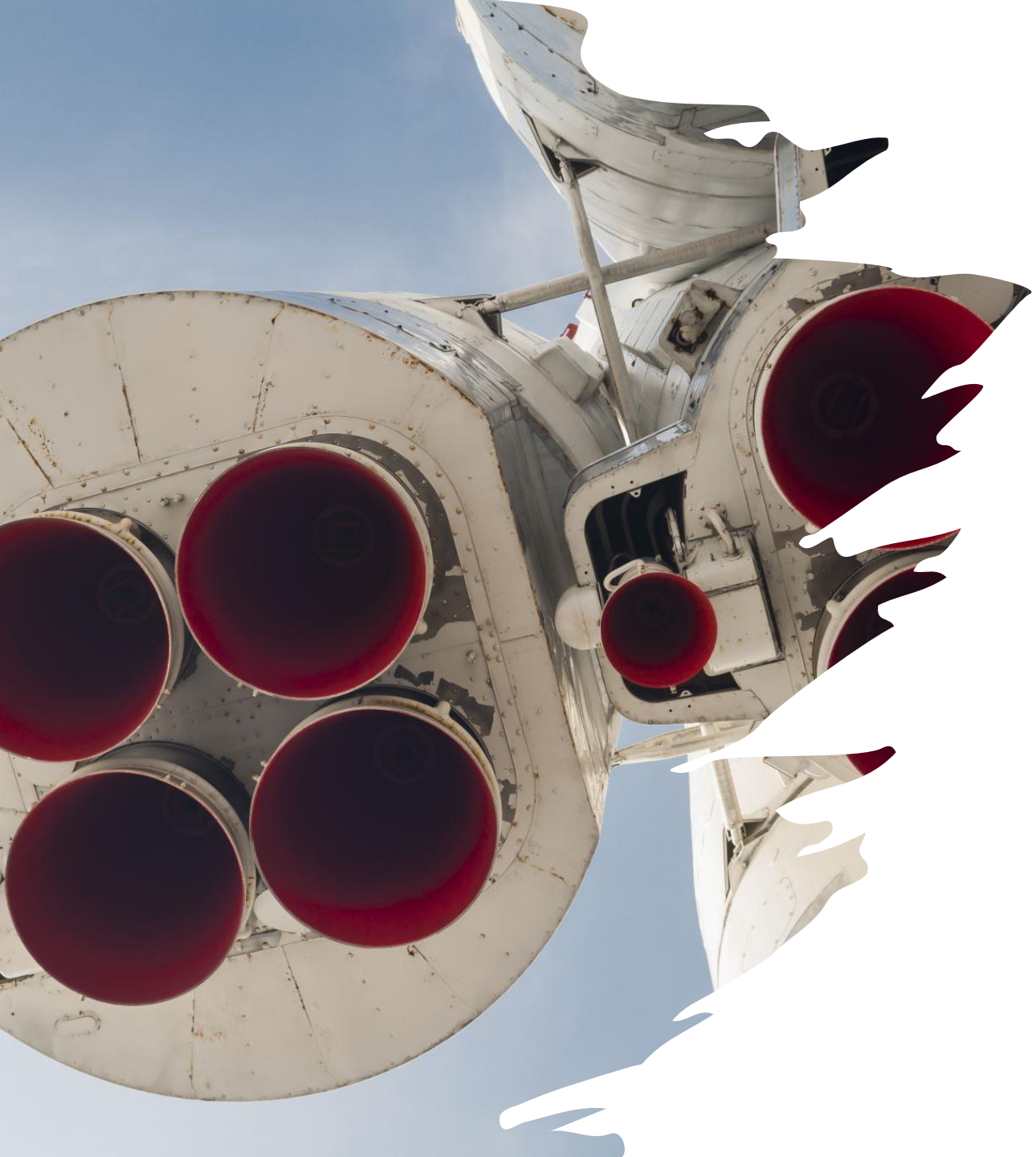
4.The relationship between Flight Number and Orbit type



5. The launch success yearly trend



[The-Assignment-of-Cousera-/jupyter-labs-eda-dataviz (1).ipynb at main · victory8230/The-Assignment-of-Cousera- (github.com)]

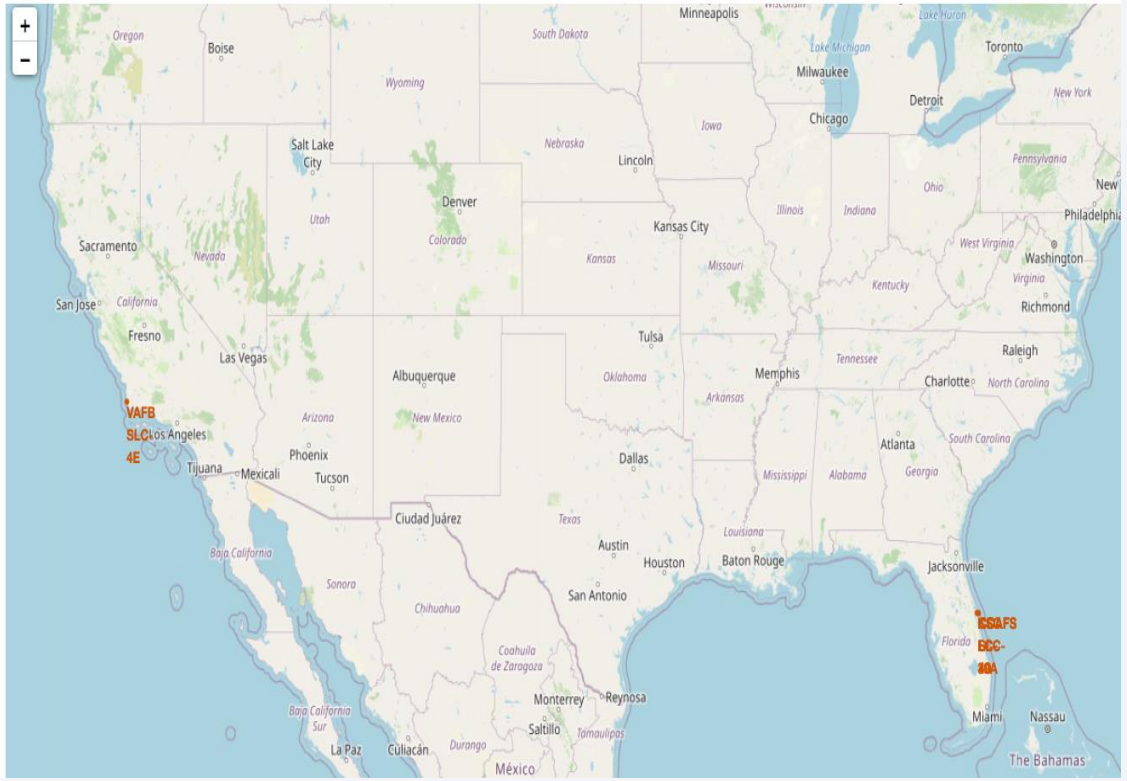


EDA with SQL

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first 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 records which will display the month names, failure landing outcomes in drone ship,
- booster versions, launch site for the months in year 2015.
- Rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in
- descending order.
- The-Assignment-of-Cousera-/jupyter-labs-eda-sql-coursera_sqllite (1).ipynb at main · victory8230/The-Assignment-of-Cousera- (github.com)

Build an Interactive Map with Folium

Map markers have been added to the map to find an optimal location for building a launch site and calculate the distances between a launch site to its proximities

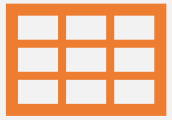


[The-Assignment-of-Cousera-/lab_jupyter_launch_site_location (2).ipynb at main · victory8230/The-Assignment-of-Cousera- (github.com)]

Build a Dashboard with Plotly Dash

- Dash and html components were used as they are the mist important thing and almost everything depends on them, such as graphs, tables, dropdowns
- Pandas was used to simplifying the work by creating dataframe.
- Plotly was used to plot the graphs
- Pie chart and scatter charter were used to for plotting purposes.
- Rangeslider was used for payload mass range selection
- Dropdown was use for launch sites

Predictive Analysis (Classification)



1. Building the model Create column for the class

Standardize the data

Split the data into train and test set Build model and fit the data

2.Evaluate the data Calculate the accuracy Calculate the confusion matrix Pilot the results

3.Finding the model Find the best hyperparameters

Find the test model with highest accuracy

Confirm the optimal model

Results

Exploratory data analysis results

-Collecting the data and wrangling the data by preprocessing data

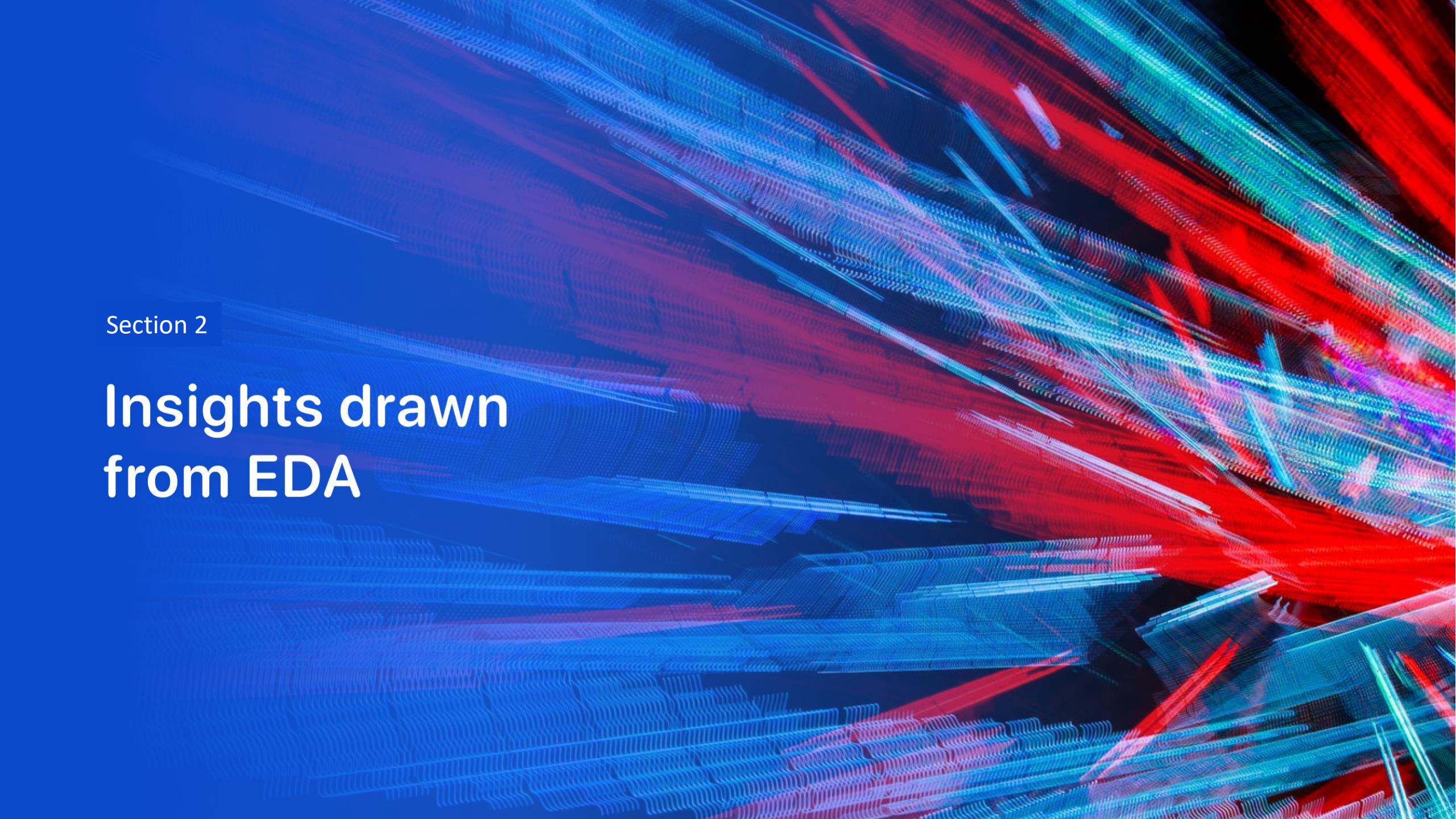
Interactive analytics demo in screenshots

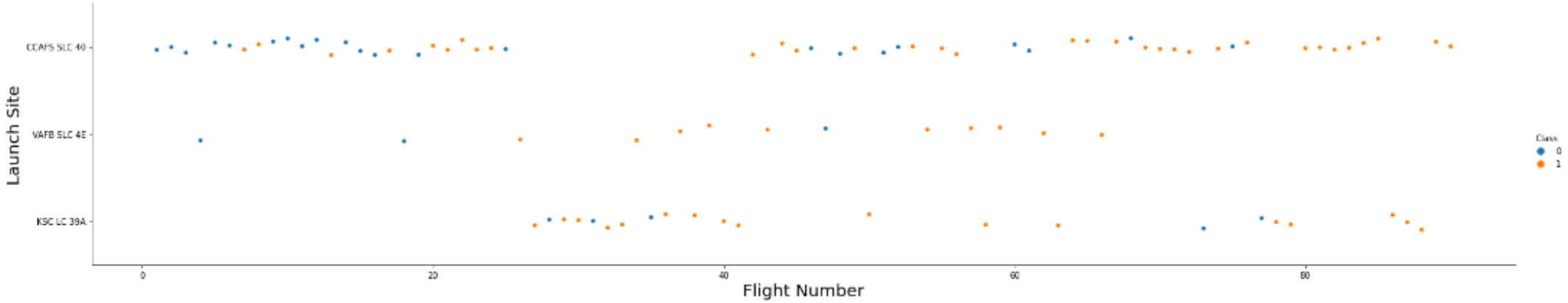
-Get insight from the chart graph and table made by data

Predictive analysis results

-Find the test model with highest accuracy

-Confirm the optimal model


Section 2
Insights drawn from EDA

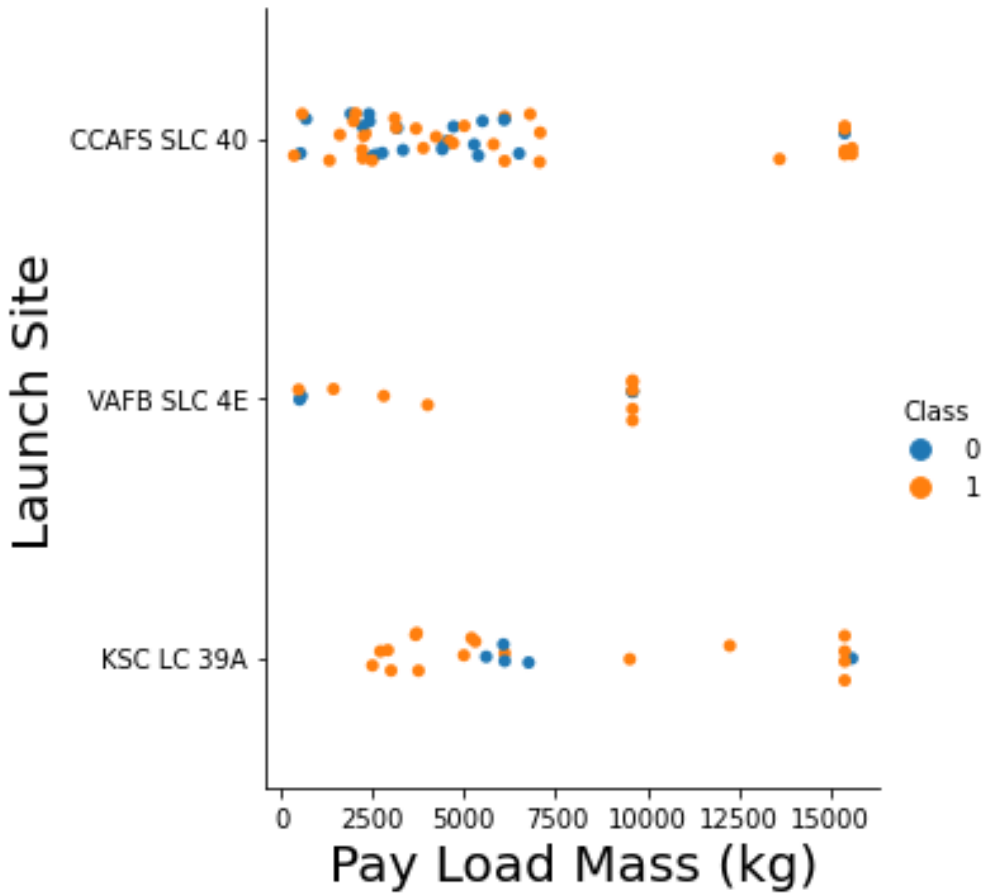


Flight Number vs. Launch Site

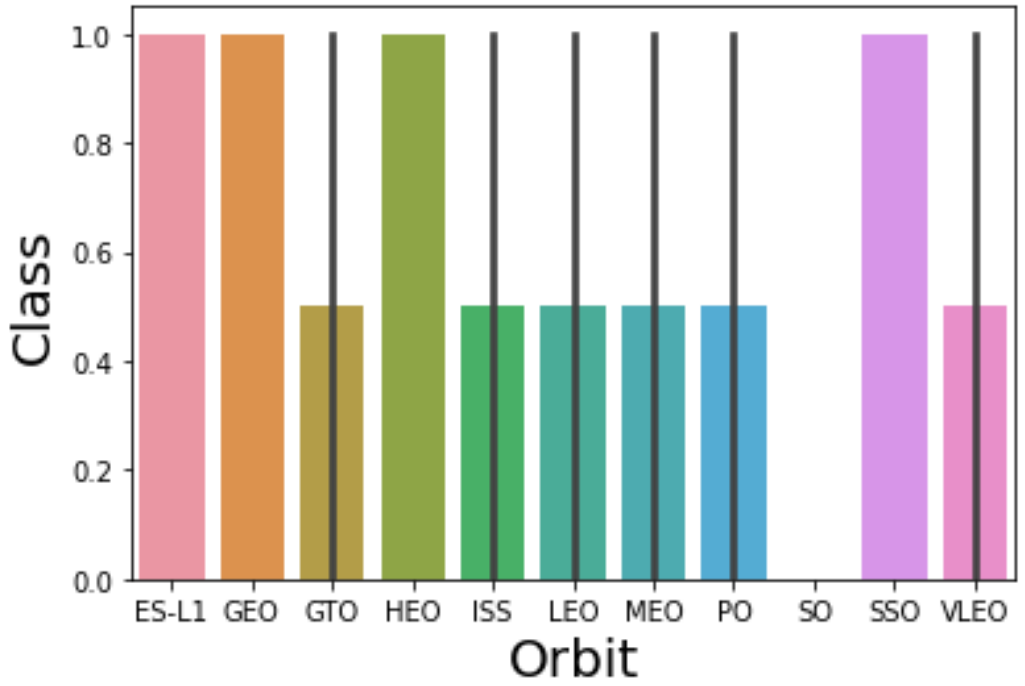
- Launch from the site of CCAFS SLC 40 are significantly higher than launches from the other sites
- With the increase the number of flights, the success rate is increasing in the launch sites

Payload vs. Launch Site

- The range of the rockets pay load mass are between in 0 and 7500 Kg
- Mostly, the lower pay load mass rocket launched The CCAFS SCL 40 launch Site



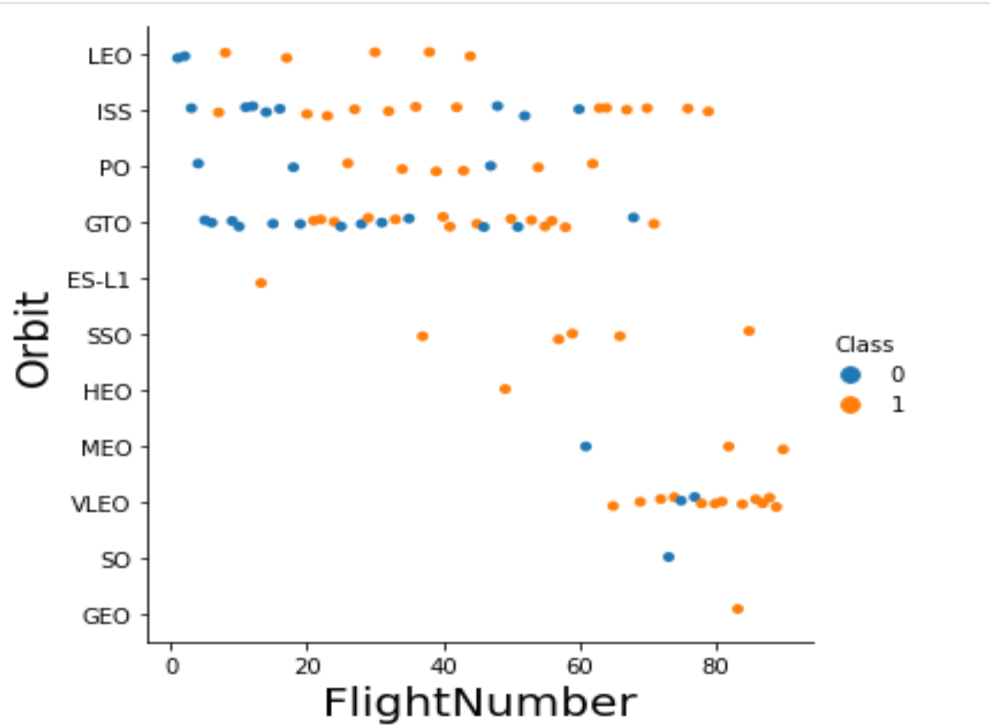
Success Rate vs. Orbit Type

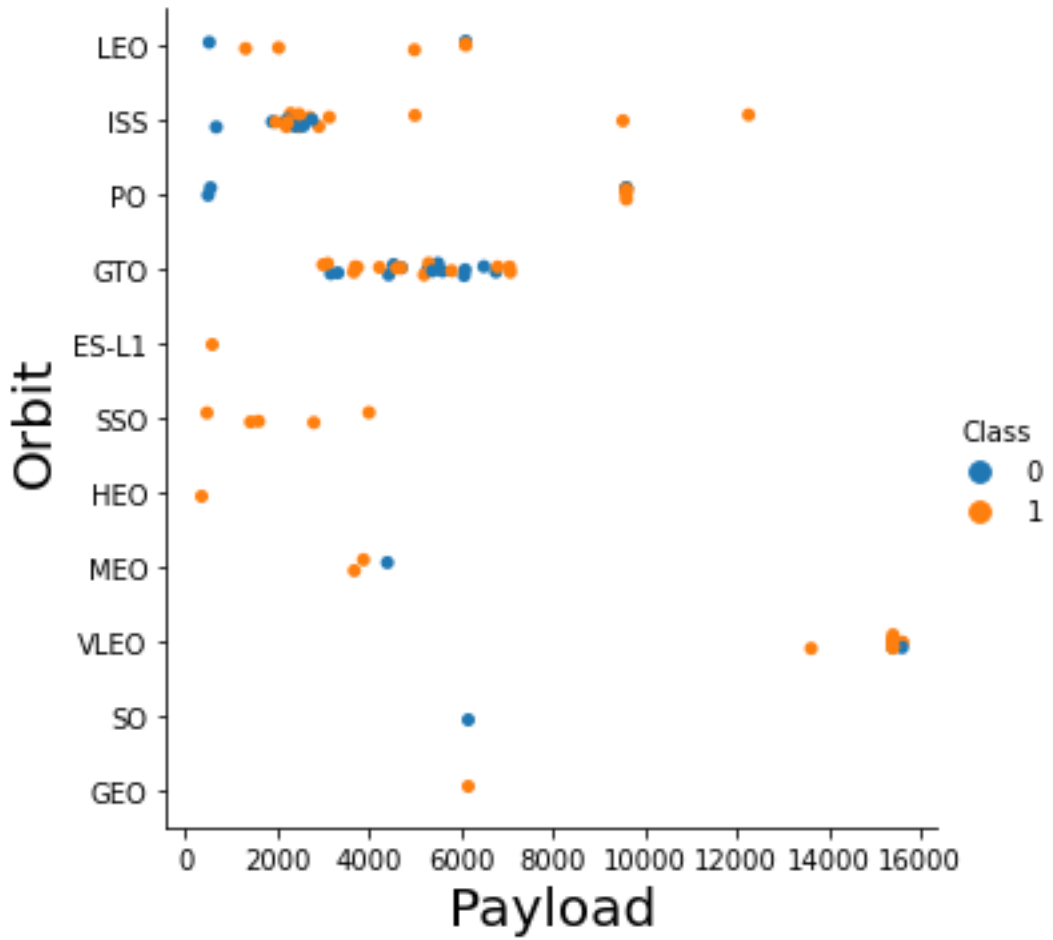


- ES-L 1, GEO, HEO, and SSO have a success rate of 100&
- SO has a success rate of 0%

Flight Number vs. Orbit Type

- 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.

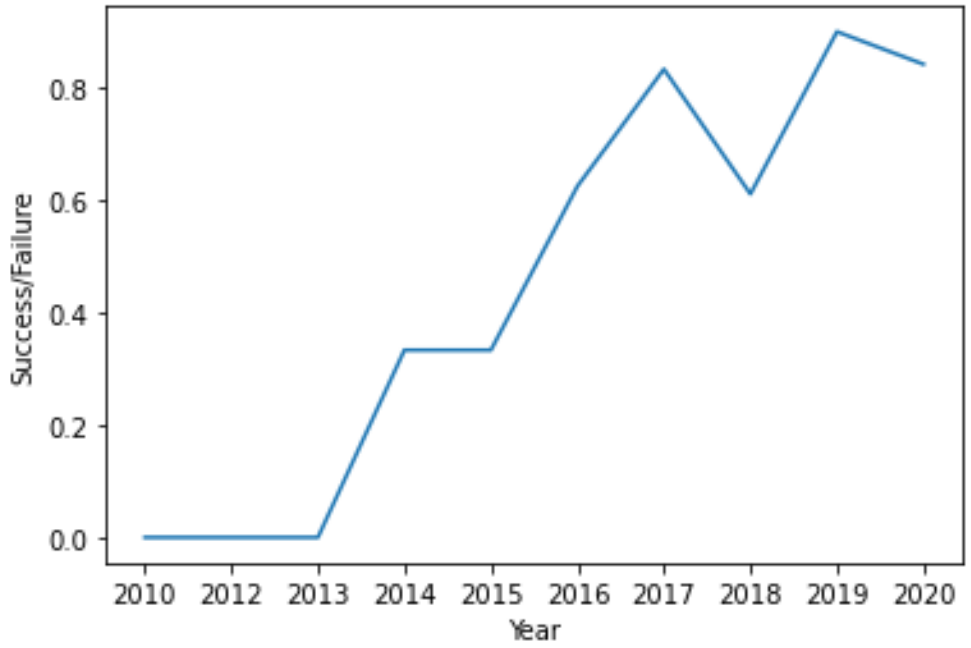




Payload vs. Orbit Type

- 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

Launch Success Yearly Trend



- The success rate since 2013 kept increasing till 2020 except 2018.

All Launch Site Names

- We can get the unique values by using “DISTINCT”

```
[13]: %sql SELECT DISTINCT LAUNCH_SITE as “Launch_Sites” FROM SPACEXTBL;
```

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

[13]:

“Launch_Sites”
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`
- We can get only 5 rows by using LIMIT

In [9]:
```
%sql SELECT LAUNCH_SITE from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

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

Out[9]:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

In [20]:
```
%sql select sum(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;
```

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

Out[20]:

payloadmass
619967

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- We can get the sum of all values by using “SUM”

```
In [21]: %sql select avg(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;
 * sqlite:///my_data1.db
Done.
```

Out[21]:

payloadmass
6138.287128712871

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- We can get the average of all values by using “AUG”

```
[22]: %sql select min(DATE) from SPACEXTBL;
```

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

[22]:

min(DATE)
01-03-2013

First Successful Ground Landing Date

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

- We can get the first successful data by using “MIN”. because first date is same with the minimum date

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
- %sql select BOOSTER_VERSION from SPACEXTBL where Landing_Outcome='Success (drone ship)'And PAYLOAD_MASS__KG_ > 4000 And PAYLOAD_MASS__KG_ <6000;

```
%sql select BOOSTER_VERSION from SPACEXTBL where LANDING__OUTCOME='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEE
```

```
 * ibm_db_sa://ktf76410:***@ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:31321/bludb
Done.
```

booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here

```
[39]: %sql select count(MISSION_OUTCOME) as missionoutcomes from SPACEXTBL GROUP BY MISSION_OUTCOME;
 * sqlite:///my_data1.db
Done.
```

[39]:

missionoutcomes
1
98
1
1

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Present your query result with a short explanation here

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

```
[44]: %sql select BOOSTER_VERSION  from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

[44]:

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

In [16]:

```
%sql SELECT TO_CHAR(TO_DATE(MONTH("DATE"), 'MM'), 'MONTH') AS MONTH_NAME, \
    LANDING__OUTCOME AS LANDING__OUTCOME, \
    BOOSTER_VERSION AS BOOSTER_VERSION, \
    LAUNCH_SITE AS LAUNCH_SITE \
    FROM SPACEXTBL WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND "DATE" LIKE '%2015%'
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

Out[16]:

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

Rank the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending or

```
%sql SELECT "DATE", COUNT(LANDING__OUTCOME) as COUNT FROM SPACEXTBL \
    WHERE "DATE" BETWEEN '2010-06-04' and '2017-03-20' AND LANDING__OUTCOME LIKE '%Success%' \
    GROUP BY "DATE" \
    ORDER BY COUNT(LANDING__OUTCOME) DESC
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

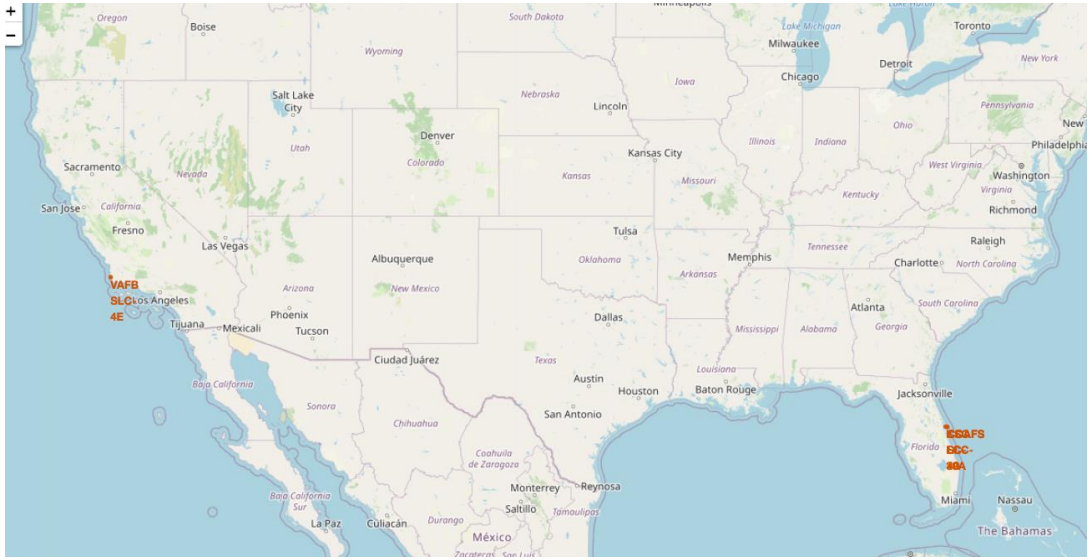
DATE	COUNT
2015-12-22	1
2016-04-08	1
2016-05-06	1
2016-05-27	1
2016-07-18	1
2016-08-14	1
2017-01-14	1
2017-02-19	1

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

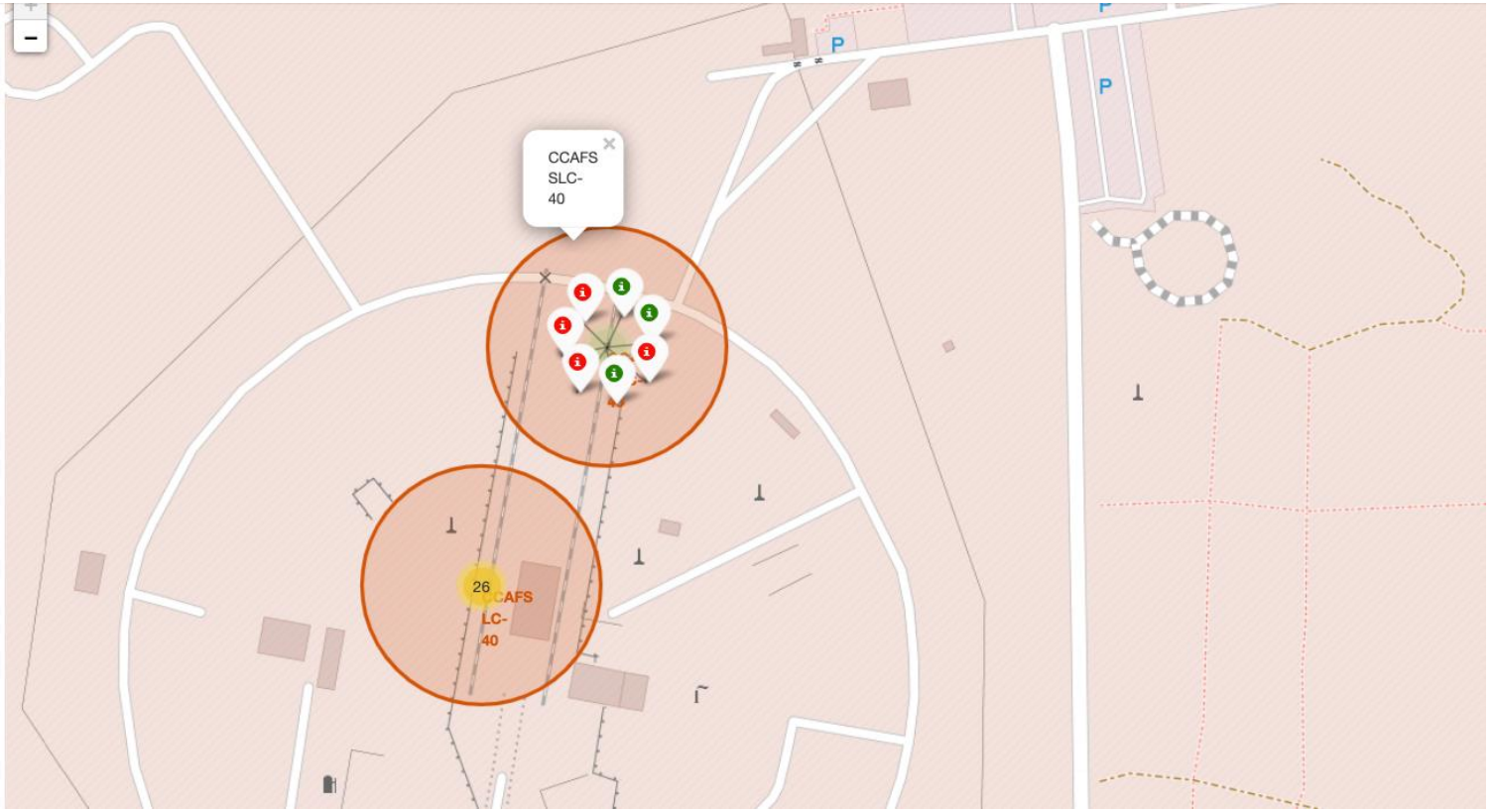
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order


Section 3
Launch Sites
Proximities Analysis

< All launch sites’ location >

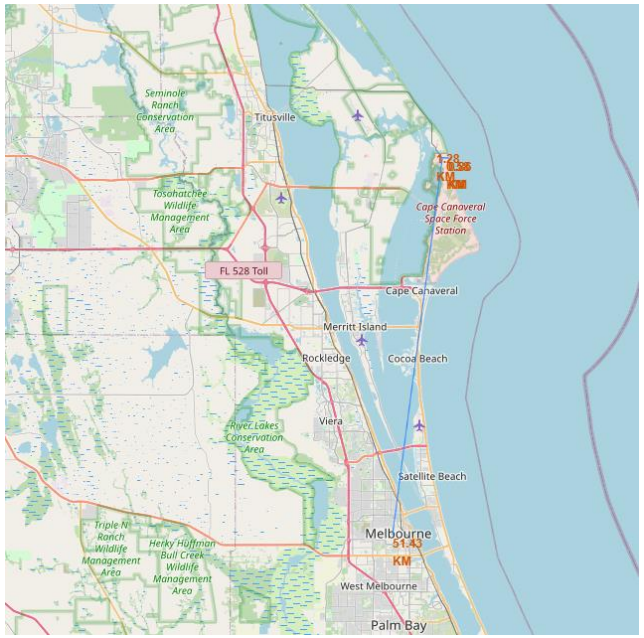


- All launch sites’ location markers on a global map near Florida and California

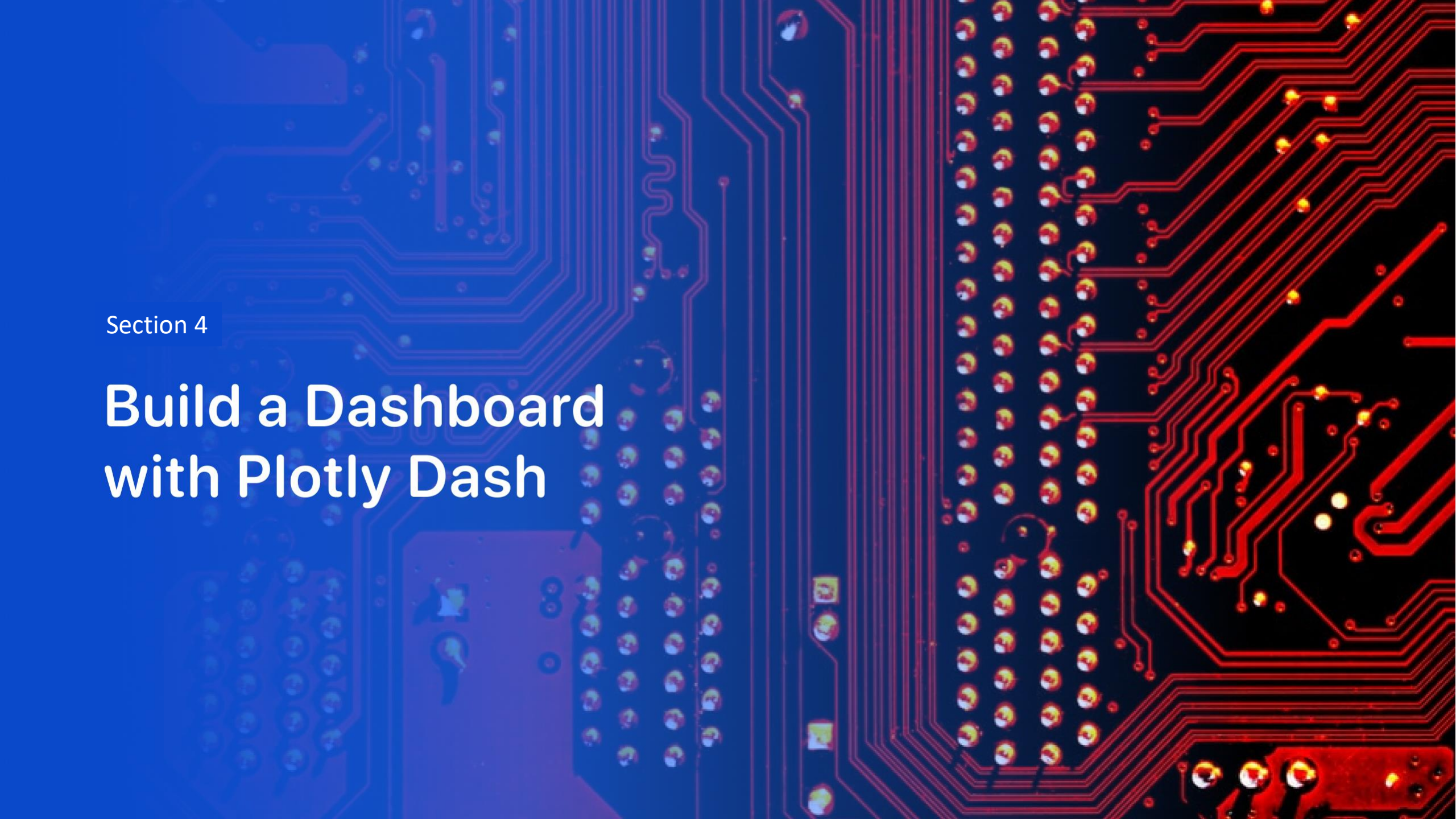


< launch outcomes >

- Green means successful
- Red means failure



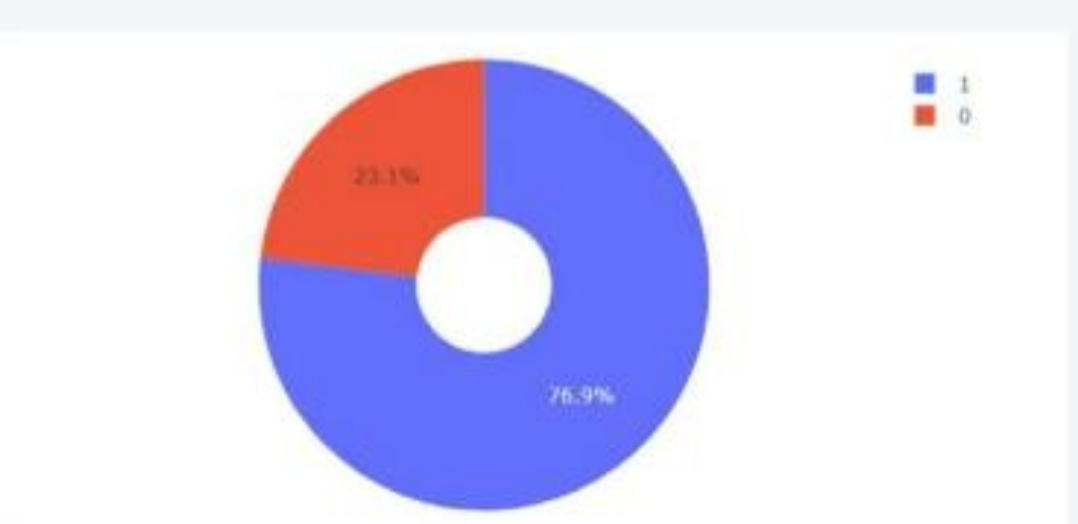
< launch site to its proximities >


Section 4
Build a Dashboard with Plotly Dash

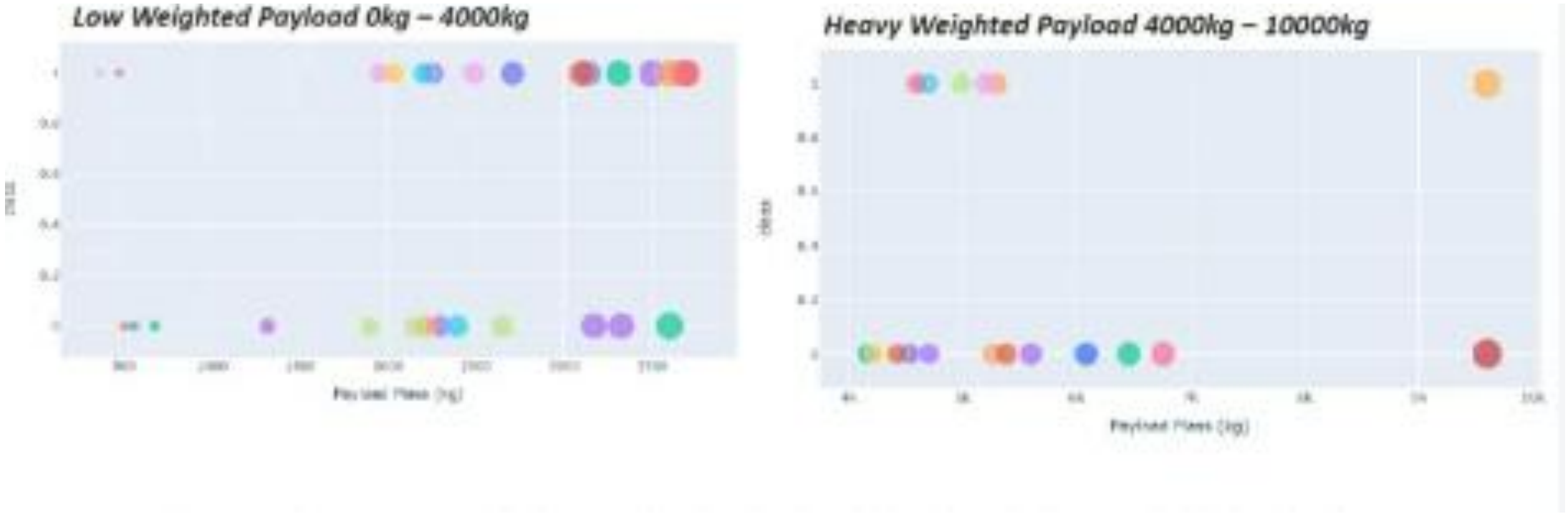
<Dash launch success count for all sites, in a piechart >

- KSC LC-39A is the most successful launches from all the sites

<highest launch success ratio >



- The pie chart for the launch site with highest launch success ratio
- KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate



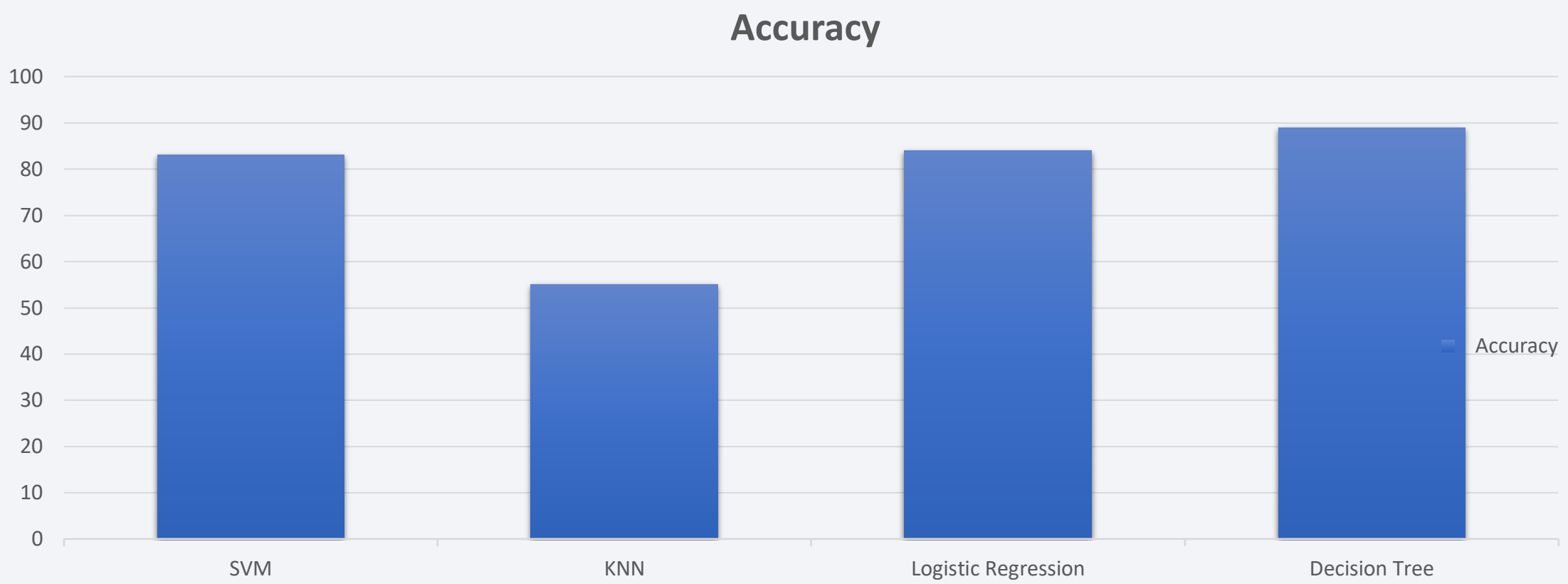
<Payload vs. Launch Outcome scatter plot for all sites >

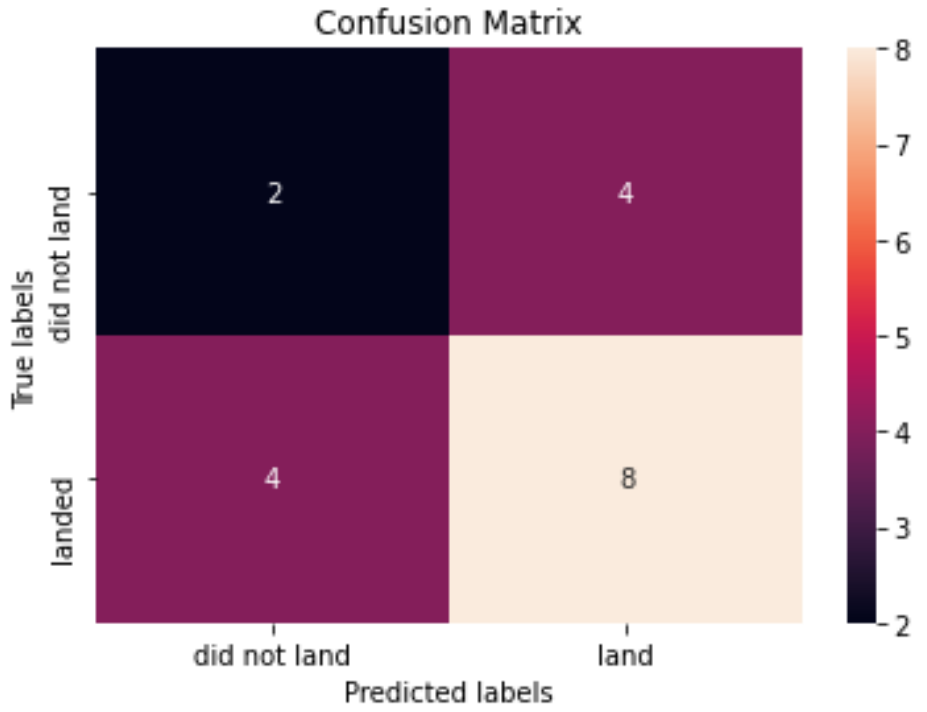
- Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

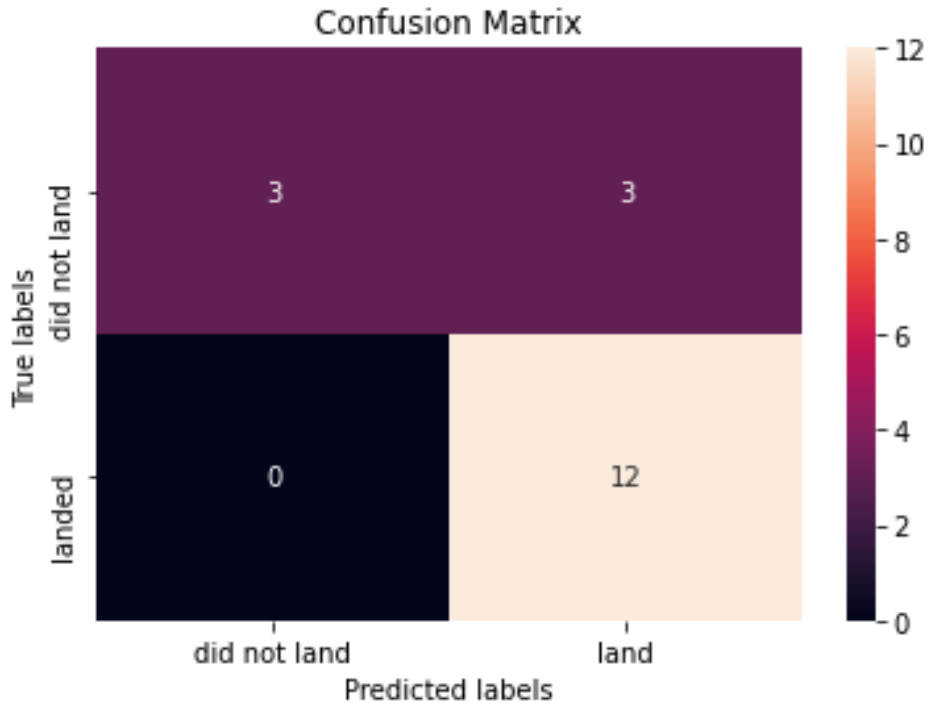
Section 5

Predictive Analysis (Classification)

Classification Accuracy







Confusion Matrix

- We see that the major problem is false positives

Conclusions

- Launch from the site of CCAFS SLC 40 are significantly higher than launches from the other sites
- With the increase the number of flights, the success rate is increasing in the launch sites
- The success rate since 2013 kept increasing till 2020 except 2018.
- KSC LC-39A is the most successful launches from all the sites
- KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate
- The most accuracy model is decision tree model

Thank you!