Winning Space Race with Data Science

Vigneshwaran Namasivayam

December 23. 2022

Outline

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

Executive Summary

- Summary of methodologies
  - This case study has the follows steps:
  - Data collection
  - Data wrangling
  - Exploratory data analysis via SQL
  - Interactive data visualisation
  - Predictive analytics via machine learning algorithms
- Summary of all results

  The case studies produce the following outputs
  - Exploratory data analysis Analysis results
  - Interactive dashboard
  - Predictive analysis of machine learning models

Introduction

- Project background and context
  - SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars which is economically cheaper than other providers (cost upward of 165 million dollars each). Space X could achieve this because they can land and reuse the first stage of the rocket.
  - Therefore if we can determine if the first stage will land, we can determine the cost of a launch.
- Problems you want to find answers
  - How could we determine the price of each launch?
  - What are the key features responsible for successful and failed landing?
  - Can we ultimately predict if the Space X falcon9 first stage will land successfully?
  - What are the computational approaches can be used to predict if Space X will reuse the first stage?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX data collected by making GET request to REST API
  - Web Scraping
- Perform data wrangling
  - The main steps in data wrangling are as follows

  1. Data discovery 2. Structuring 3. Cleaning 4. Enriching 5. Validating
- Perform exploratory data analysis (EDA) using visualization and SQL
  - To manipulate an evaluate the data SQL was employed
  - For pattern determination and data visualisation Pandas and Matplotlib was used

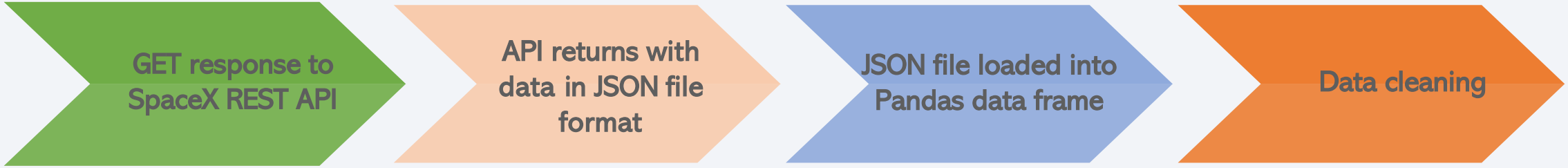
Methodology

Executive Summary (Continued)

- Perform interactive visual analytics using Folium and Plotly Dash
  - Folium was employed to analyze geospatial understanding
  - Plotly Dash was used to create interactive dashboard
- Perform predictive analysis using classification models
  - The main steps in machine learning are as follows
  - A) Data set preparation: 1. Transformation and normalization 2. Splitting the data set into training and testing data 3. feature reduction
  - B) Modeling: 1. Selection of model 2. Training the model 3. Evaluating the model 4. Parameter tuning 5. Prediction 6. Publish

Data Collection

- SpaceX API was used to retrieve the data about launch information including landing outcome, payload, payload mass, mission outcome, booster version



- BeautifulSoup was employed to retrieve the data from Wikipedia

Data Collection SpaceX REST API

1

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

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

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

2

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

3

```
# Call getBoosterVersion
getBoosterVersion(data)
```

the list has now been update

```
BoosterVersion[0:5]
```

```
['Falcon 1', 'Falcon 1', 'Falcon 1', 'Falcon 1', 'Falcon 9']
```

we can apply the rest of the functions here:

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

4

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

5

```
# Create a data from launch_dict
data=pd.DataFrame({key:pd.Series(value) for key, value in launch_dict.items()})
```

6

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

7

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```


Github: https://github.com/vnamasiva/IBM_DS_Capstone_Project/blob/main/Data_Collection.ipynb

Data Collection Web Scrapping

1

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Next, request the HTML page from the above URL and get a `response` object

2

```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

3

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text, "html.parser")
```

4

```
# 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.findAll('table')
```

5

```
column_names = []

# 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

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

6

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# 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']=[]
```

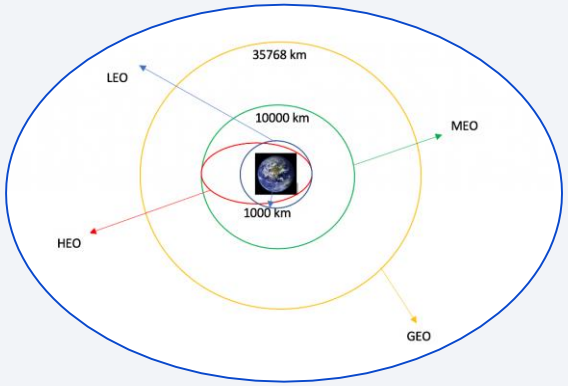
7

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


Github: https://github.com/vnamasiva/IBM_DS_Capstone_Project/blob/main/Webscraping_SpaceX_records_from_Wikipedia.ipynb

Data Wrangling

- SpaceX data set contains several different cases such as either booster land successfully or did not have a successful landing.



- Initial Data mining

  .Value_counts() method help to determine the following:
  - The number of launches on each site
  - The number and occurrence of mission outcome per orbit type
  - Create a landing outcome label from Outcome column

1

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

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

2

```
# 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
```

3

```
# 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
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

4

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class=[]
for keys,value in df['Outcome'].items():
    if  value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

5

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

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

Github: https://github.com/vnamasiva/IBM_DS_Capstone_Project/blob/main/Data%20wrangling.ipynb

EDA with Data Visualization

Scatter graph

Scatter graphs used to visualize the relationship between.

- Flight number vs Launch site
- Payload vs Launch site
- Orbit typer vs Flight number
- Payload vs Orbit typ

Scatter graphs are useful to observe correlations or relationship between the two variables

Bar graph

Bar graph used to visualise the relationship between.

- Success rate vs orbit

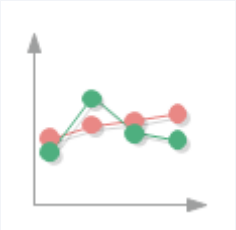
Bar graphs are useful to observe correlations or relationship between the numeric and categorical variables

Line Graph

Line graph used to visualise the relationship between.

- Success rate vs Years

Line graphs are useful to observe data variables and their trends over the course of time



EDA with SQL

- SQL framework was employed to understand the SpaceX dataset
- SQL queries are executed to answer following questions
  - 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
  - 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

Github: https://github.com/vnamasiva/IBM_DS_Capstone_Project/blob/main/EDA-SQL%20Notebook.ipynb

Geospatial interactive visual analytics with folium

The following steps were taken to visualize launch sites locations data on a folium interactive map

- Marking all launch sites on a map
  - Folium map object was used to initialise map
  - Labelling the launch site coordinates using folium.circle and folium, map.Marker
- Marking the success/failed launches for each site on the map
  - Since multiple launches have same coordinate, it is better to group them together
  - It make sense to color code successful launches to green (class=1) and failed launched to red (class=0)
  - folium.Marker and the MarkerCluster() object was used to cluster each launch
- Calculate the distances between a launch site to its proximities
  - To calculate the launch site proximities Last and Long values were used to calculate the distance between the points
  - folium.Marker object was used to show the distance after the points were assigned
  - folium.Polyline was used to show the distance line between two points and add this to map

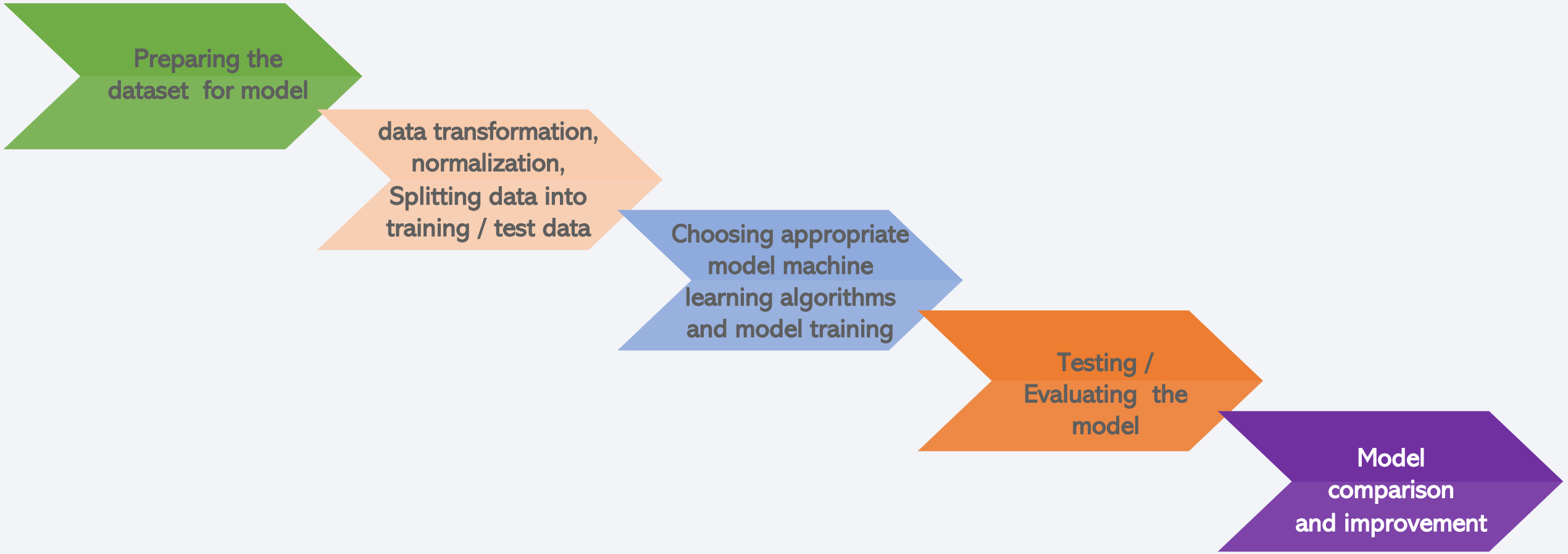
Github: https://github.com/vnamasiva/IBM_DS_Capstone_Project/blob/main/Interactive%20Visual%20Analytics%20with%20Folium.ipynb

Building interactive dashboard with plotly dash

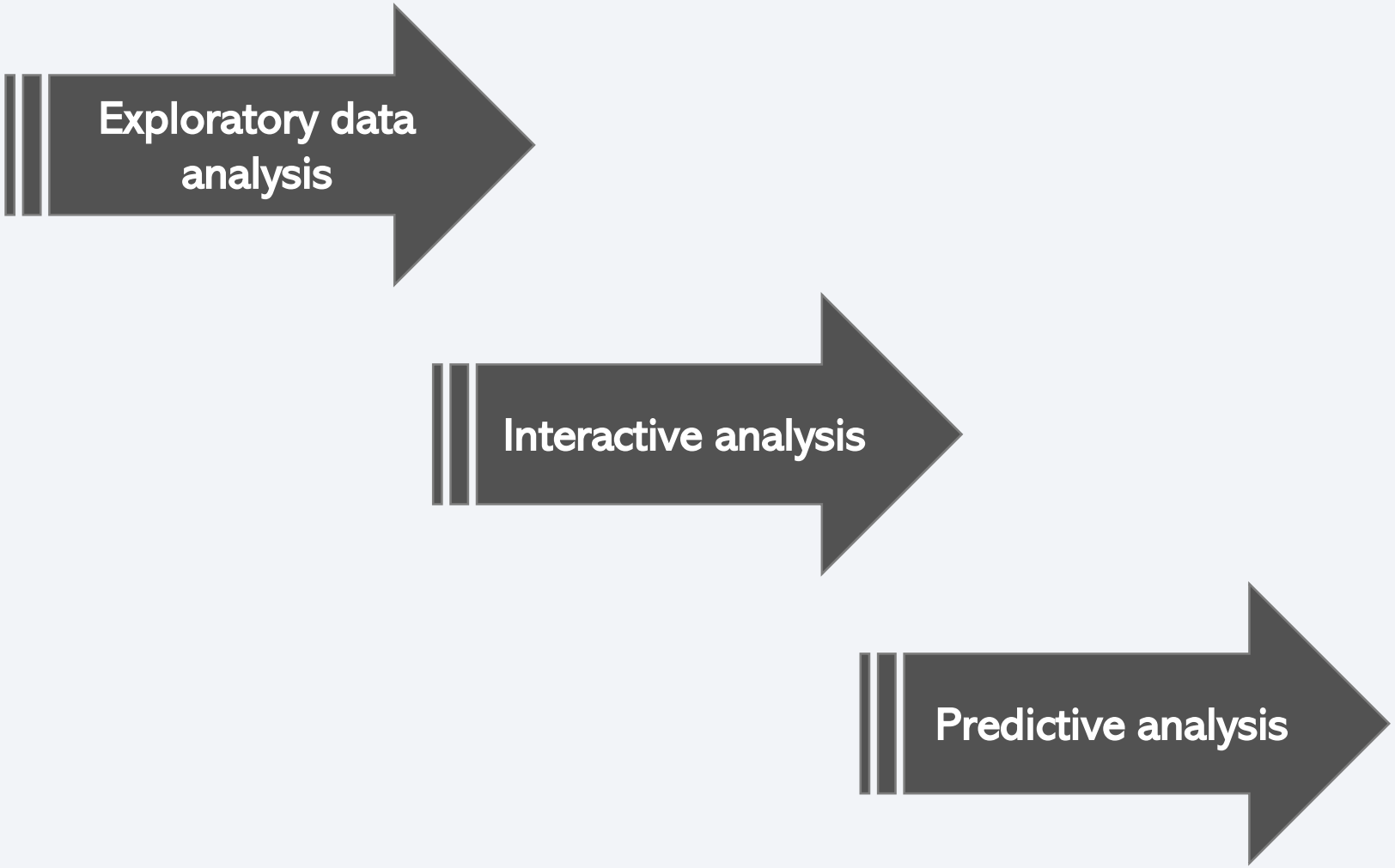
- Dashboard composed of elements with dropdown, pie chart and scatter graph
  - Launch sites / launch site can be accessed via Dropdown object
- Pie graph shows the number of total successful launches per site
  - This allows the use to see which site have most successful launched
  - Individual launch site success and failure ratio can also be determined using this chart
- scatter graph can be used to illustrate the collation between payload mass and the outcome (success or failure)
  - Payload mass can be filtered using a RangeSlider() object.

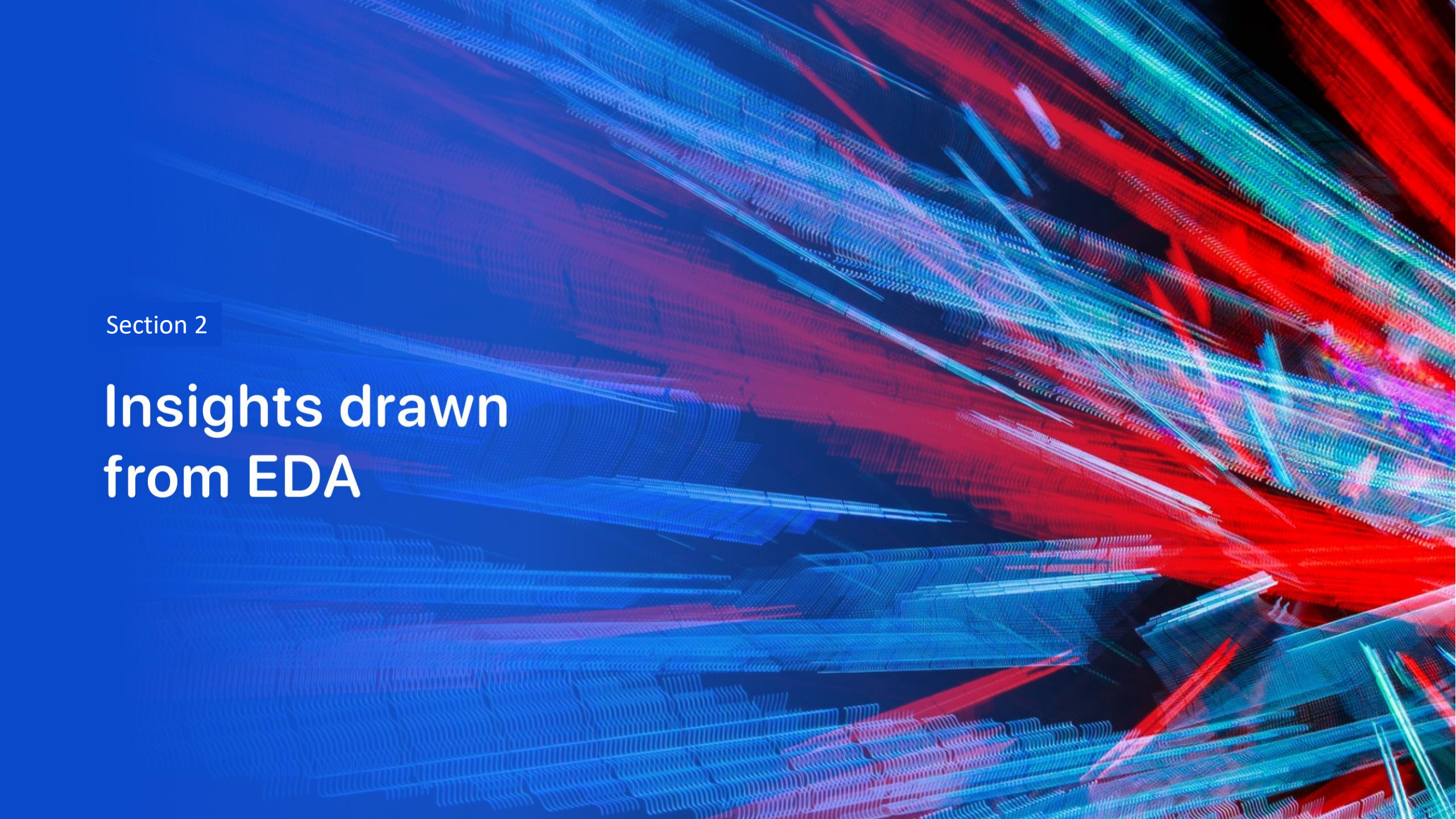
Github: https://github.com/vnamasiva/IBM_DS_Capstone_Project/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

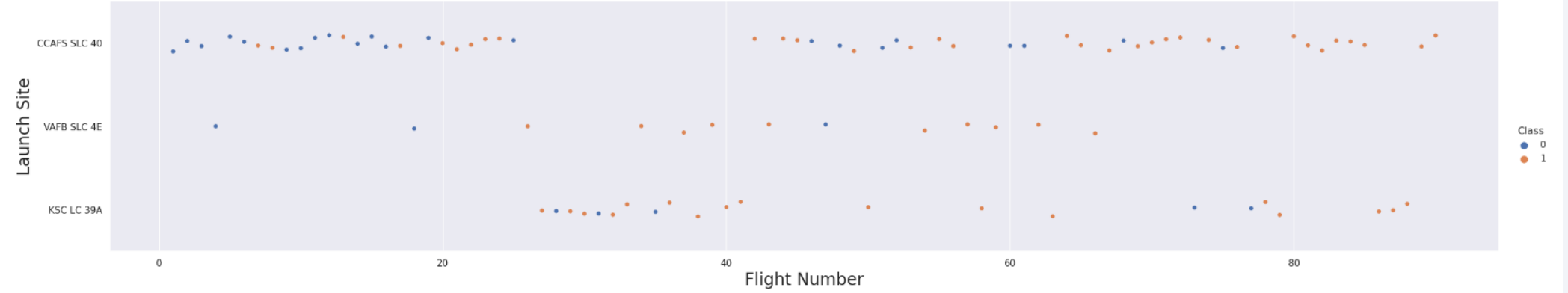


Results


Exploratory data analysis
Interactive analysis
Predictive analysis


Section 2
Insights drawn from EDA

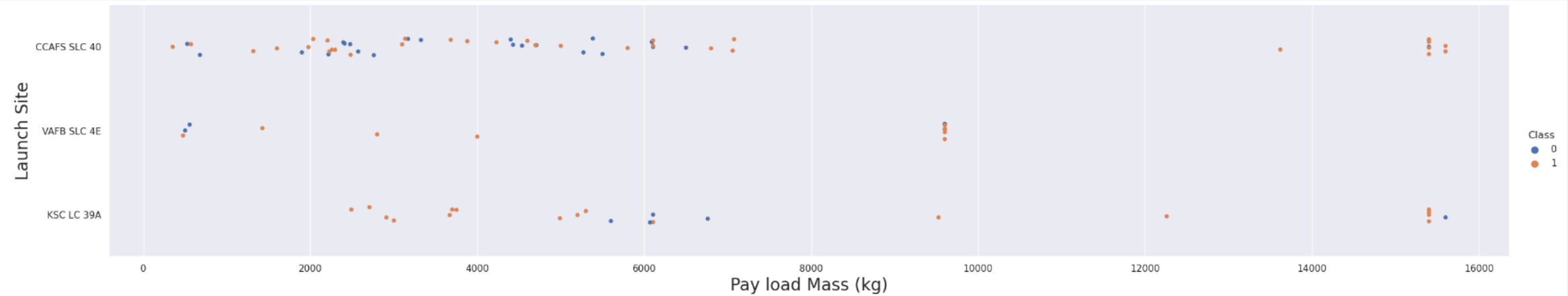
Flight Number vs. Launch Site



The scatter graph of Launch site vs Flight number revealed that

- Success rate increases with the increase in the number of flights
- KSC LC 39A launch site is most successful launches compared to other launch sites
- Many of the early flights were unsuccessful at launch site CCADS SLC 40

Payload vs. Launch Site

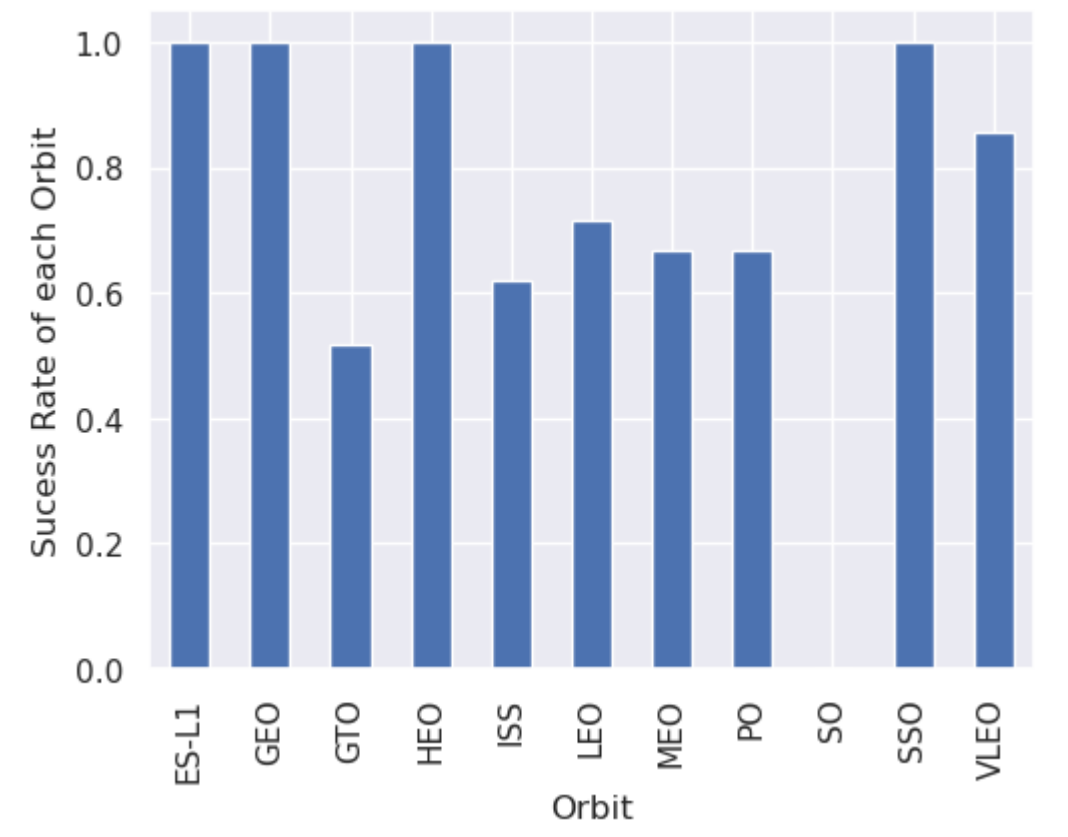


The scatter graph of Launch site vs Payload mass shows that

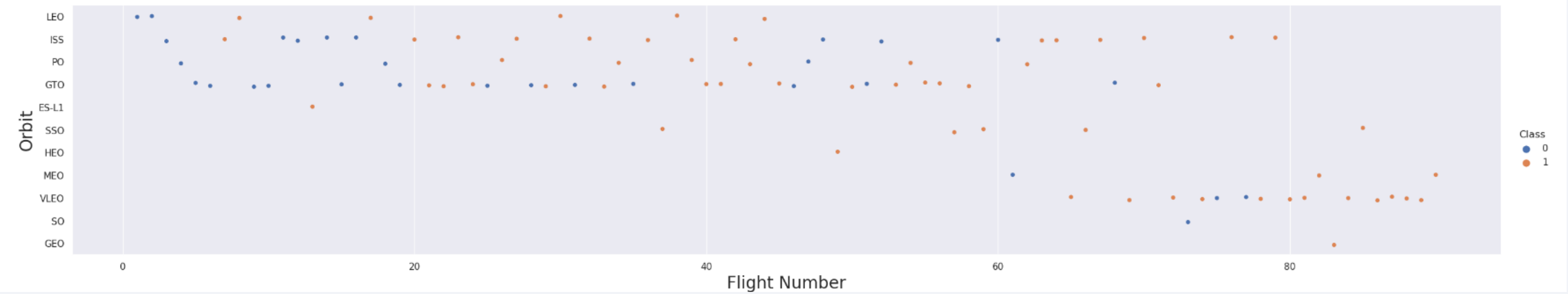
- At a specified launch site, Launch site vs Payload mass does not show clear correlation
- Most successful landing was observed the payload between 1000 to 5000 kg

Success Rate vs. Orbit Type

- This bar chart illustrates that the four of the below mentioned orbit type shows 100% success rate
  - ES-L-1
  - GEO
  - HEO
  - SSO
- The orbit type with the success rate zero is
  - SO
- The VLEO has ~80% and the remaining orbit type are between 40 to 60% success rate



Flight Number vs. Orbit Type



The scatter graph of Flight number vs Orbit type revealed some the useful information which the bar graph did not show

- Highest success rate for GEO, HEO and ES-L1 orbits due to the single flight to the respective orbit type
- There only little correlation between flight number and GTO
- All the five flight were successful for the SSO orbit.

Payload vs. Orbit Type

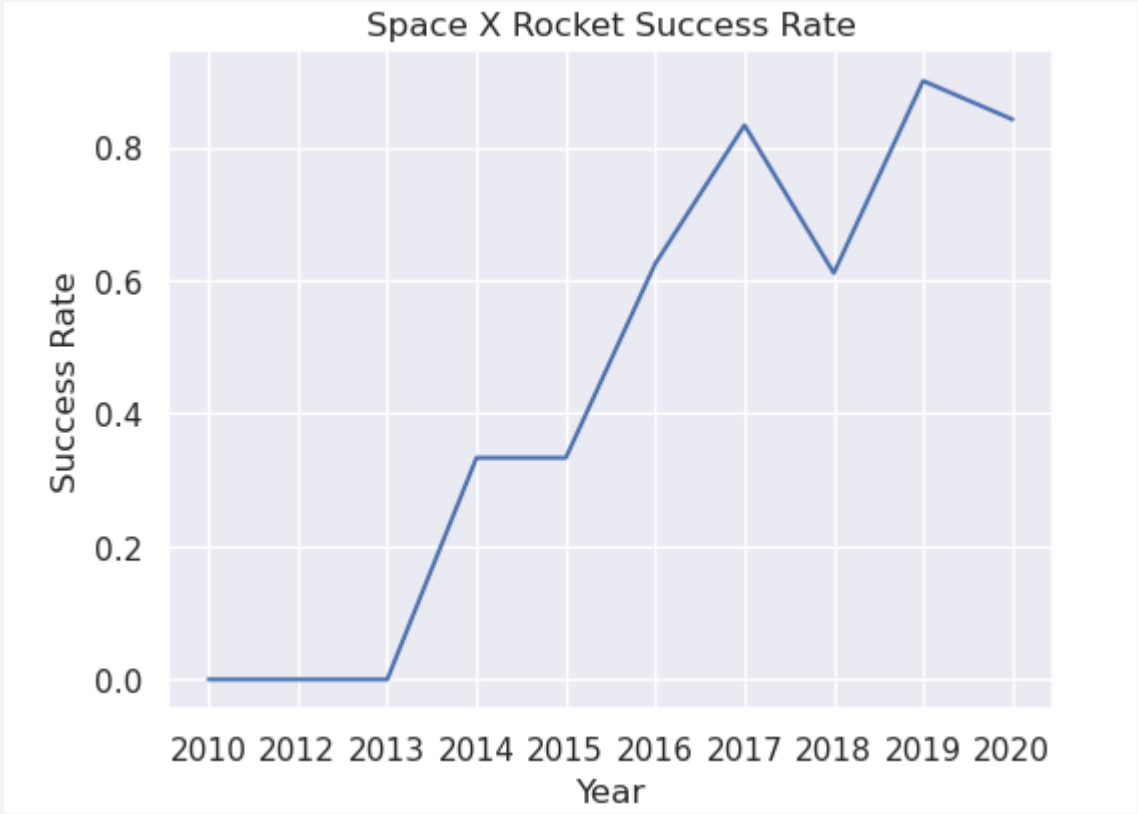
- The scatter graph of Payload vs Flight number showing:

  Among the different orbit types, the three mentioned orbit type have most successful launch with heavy payload

  - PO
  - ISS
  - LEO

Launch Success Yearly Trend

- The line chart clearly illustrate that
  - All the landings between the years between 2010 and 2013, all are unsuccessful
  - After 2013, in general the success rate shows an increased trend
  - In 2019, it reached to a high success rate



All Launch Site Names

Display the names of the unique launch sites in the space mission

```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEX
```

```
 * ibm_db_sa://gyg99847:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

2]:

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

- The duplicates of launch sites are removed by the DISTINCT query and the unique values are displayed from the SPACEX table

Launch Site Names Begin with 'CCA'

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

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

```
 * ibm_db_sa://gyg99847:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/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

- Records where launch sites begin with `CCA` extracted using WHERE and LIKE query
- With the use of LIMIT function the output of 5 records from the filtered file are displayed.

Total Payload Mass

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEX WHERE CUSTOMER = 'NASA (CRS)'
```

```
 * ibm_db_sa://gyg99847:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

4]:

1
22007

- SUM keyword has been used to calculate the total mass with WHERE keyword to extract the total mass with the customer as NASA

Average Payload Mass by F9 v1.1

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEX WHERE BOOSTER_VERSION LIKE '%F9 v1.1%'
```

```
 * ibm_db_sa://gyg99847:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

5]:

1
3226

- The keyword AVG is used to calculate the average payload mass in kg and the WHERE is used to filter the booster version F9 v1.1

First Successful Ground Landing Date

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

Hint:Use min function

```
%sql SELECT MIN(DATE) FROM SPACEX WHERE LANDING__OUTCOME LIKE '%Success%'
```

```
 * ibm_db_sa://gyg99847:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

5]:

1
2016-06-05

- MIN keyword was used to filter the first successful landing outcome om ground pad

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
%sql SELECT BOOSTER_VERSION FROM SPACEX WHERE LANDING__OUTCOME='Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

```
 * ibm_db_sa://gyg99847:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

7]:

booster_version
F9 FT B1022
F9 FT B1031.2

- WHERE keyword is used to list the booster version successful landing on drone ship with payload mass between 4000 and 6000 kg

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
%sql select count(MISSION_OUTCOME) as missionoutcome from SPACEX GROUP BY MISSION_OUTCOME;
```

```
 * ibm_db_sa://gyg99847:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

8]:

missionoutcome
44
1

- COUNT keyword has been used to calculate the total number of mission out comes with GROUPBY keyword to group the successful and unsuccessful mission out come

Boosters Carried Maximum Payload

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

```
%sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEX WHERE PAYLOAD_MASS__KG_ = (SELECT max(PAYLOAD_MASS__KG_) FROM SPACEX);
```

```
 * ibm_db_sa://gyg99847:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

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

- DISTINCT keyword has been used to extract the booster unique booster version with MAX function to retrieve the booster carried the maximum payload

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 FROM SPACEX WHERE LANDING__OUTCOME='Failure (drone ship)' AND DATE LIKE '2015%';
```

```
 * ibm_db_sa://gyg99847:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

0]:

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40

- WHERE keyword has been used to extract the result for the failed landing outcome AND date LIKE for the year 2015

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

```
sql SELECT LANDING__OUTCOME, COUNT(*) AS qty FROM SPACEX WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY LANDING__OUTCOME ORDER BY qty DESC;
```

```
 * ibm_db_sa://gyg99847:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

]:

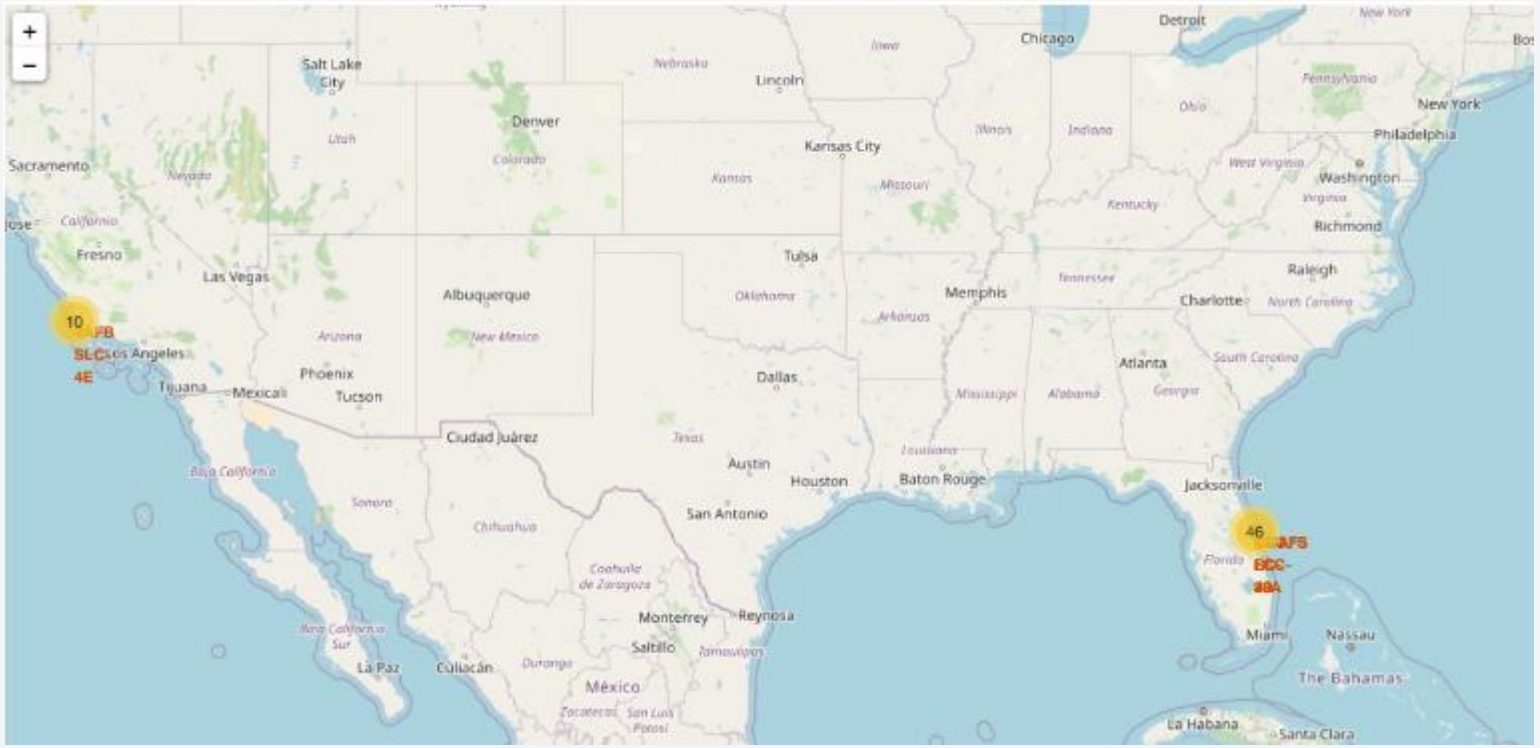
landing__outcome	qty
No attempt	7
Failure (drone ship)	2
Success (drone ship)	2
Success (ground pad)	2
Controlled (ocean)	1
Failure (parachute)	1

- This figure illustrate the 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
- COUNT keyword was used together with WHERE as well as BETWEEN keywords to filter the results to dates with in the specified column
- Further the results are grouped using the keyword GROUP BY with DESC to get the results in a descending order

Section 3

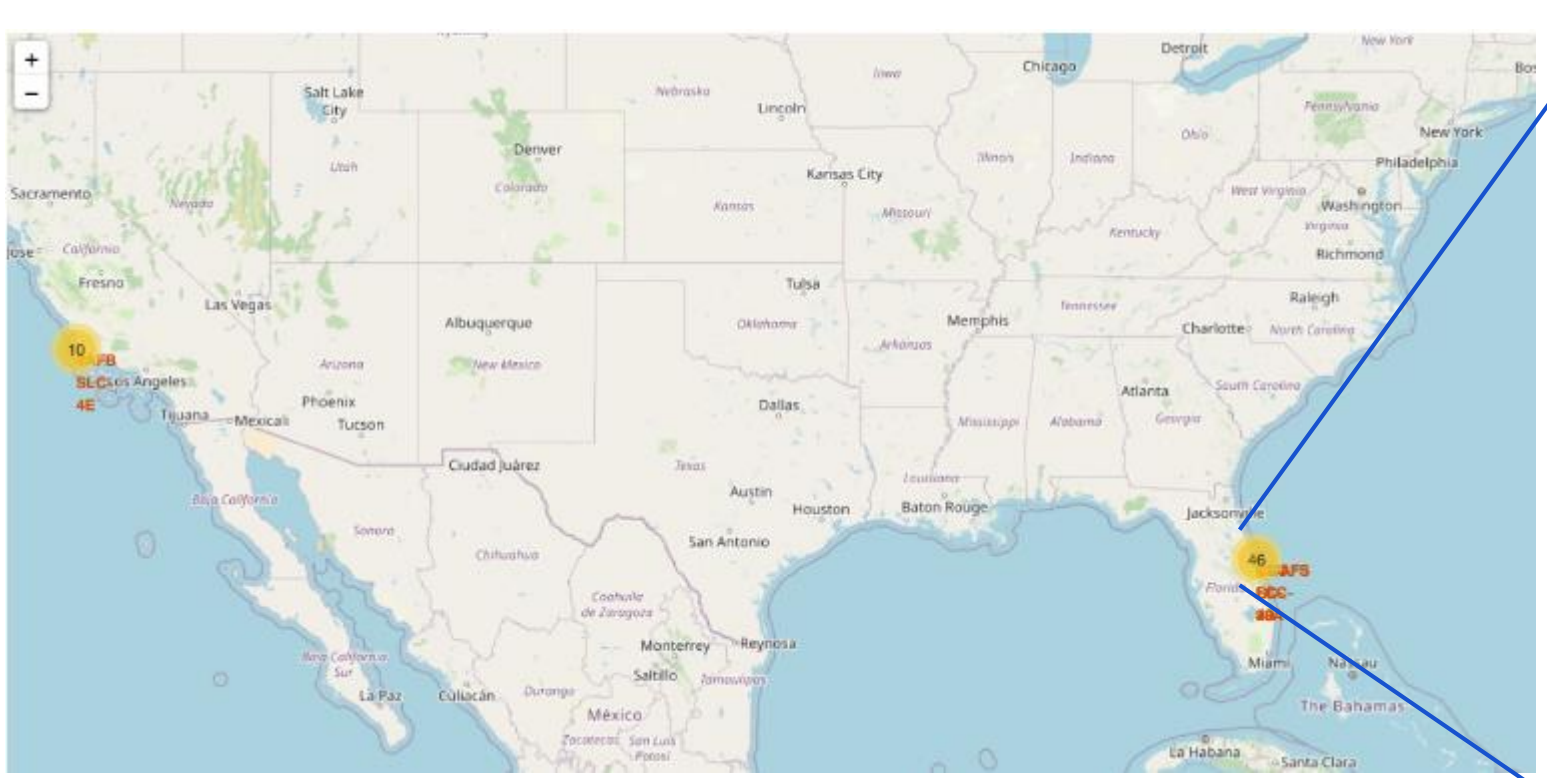
Launch Sites Proximities Analysis

SpaceX launching stations on a folium map



The SpaceX launch site in the coast of USA,
Florida
California

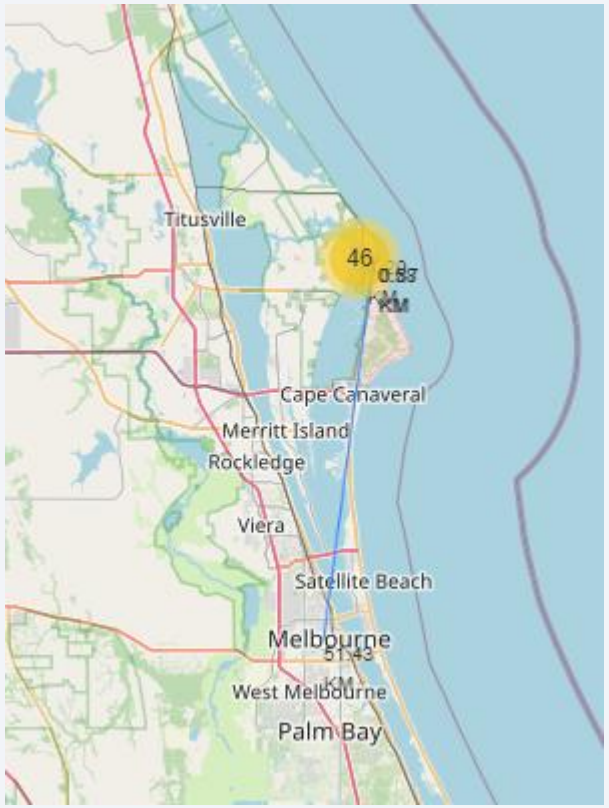
Launch site with markers and color labels

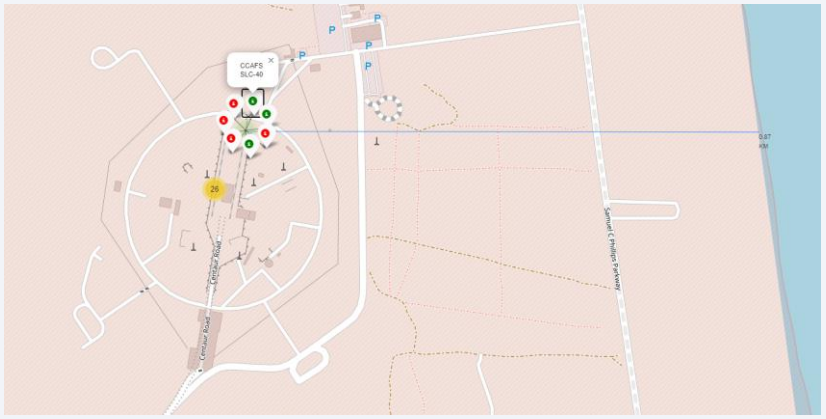


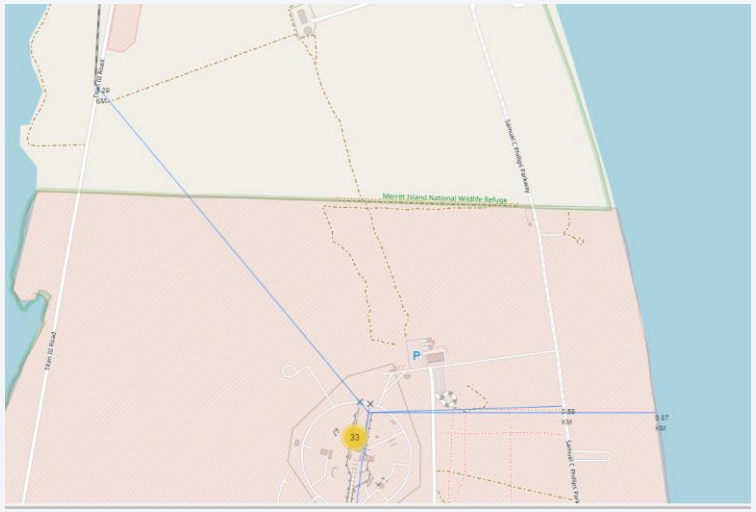
The SpaceX launch site in the coast of USA,
Florida
California

Markers showing launch sites with color labels
Green: Successful
Red: Failure

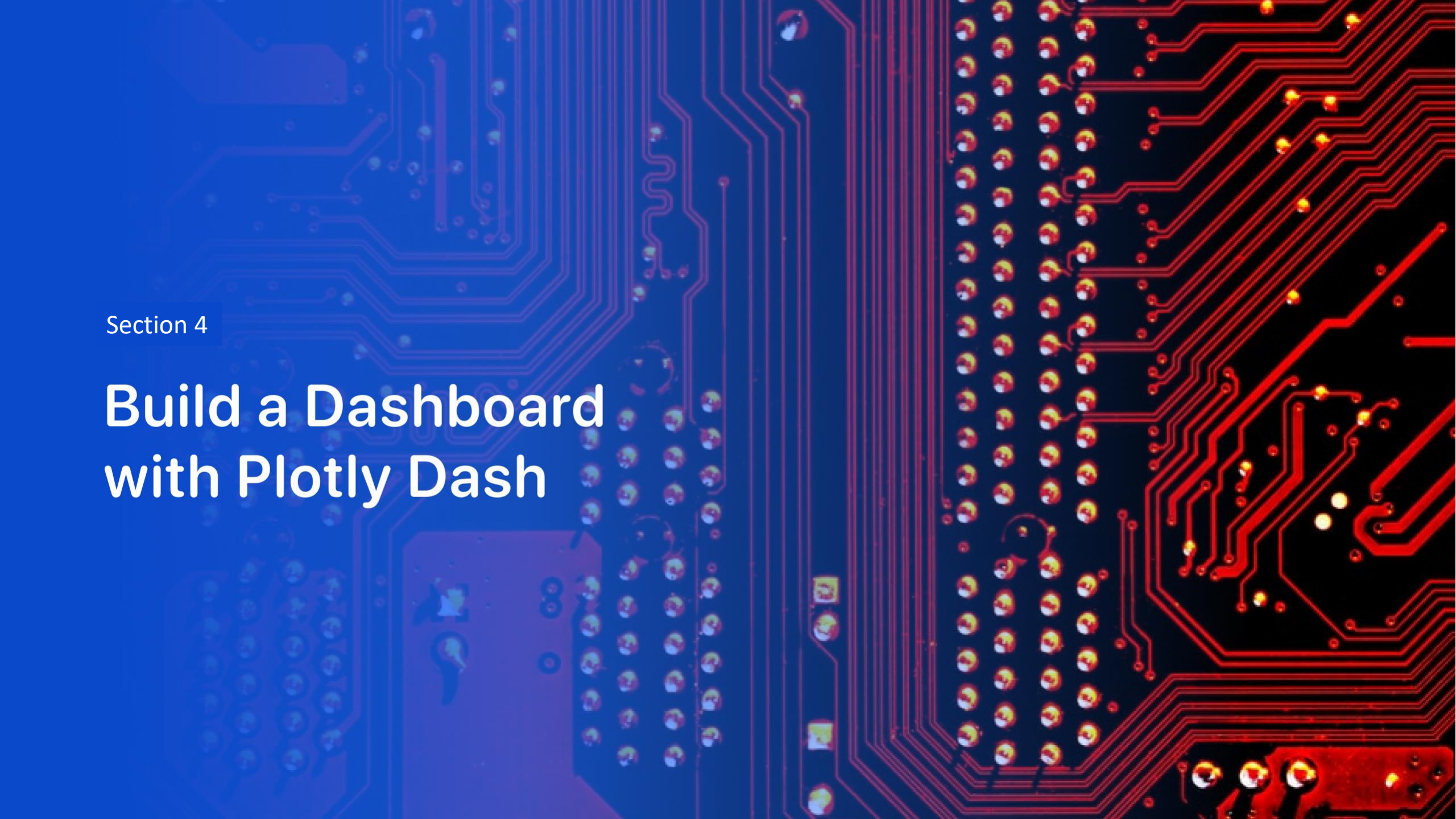
Proximity of launch sites together with area of interest



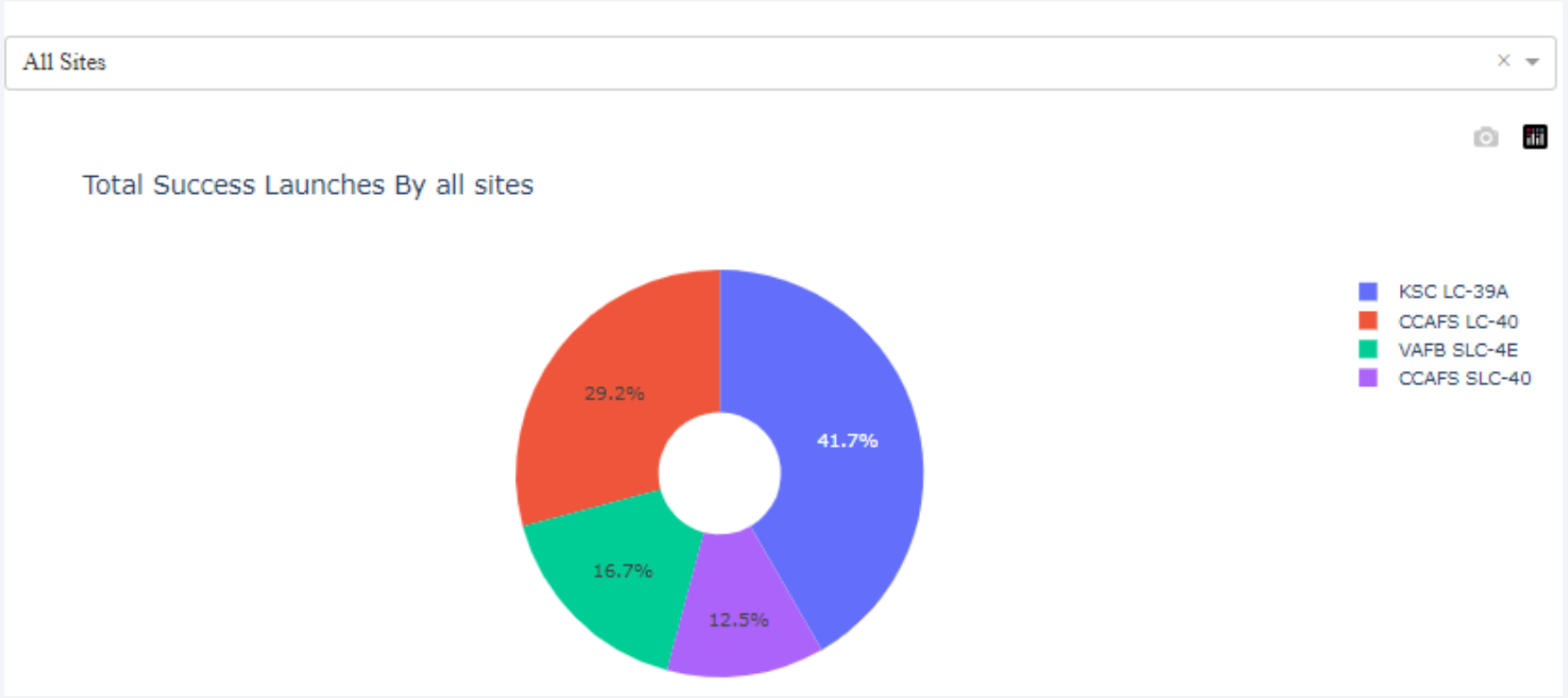




- Using the CCAFS SLC-40 launch site as an example, the distances between launch site and nearest railway station, costal line, city and high way calculated
- Closest railway station lies at 1.28 km
- Closest high way lies at 0.584 km
- Closest city is Melbourne around 51.43 km away from launch site
- Closest coastline lies at 0.87 km to the east from the launching site

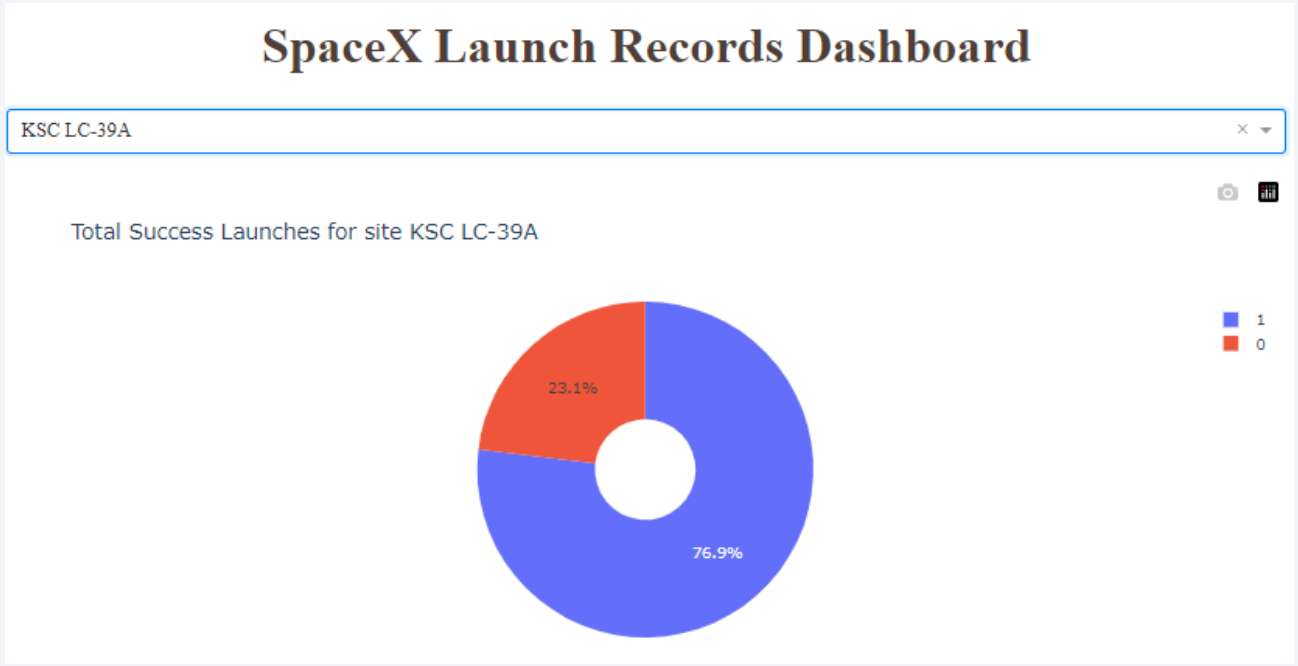

Section 4
Build a Dashboard with Plotly Dash

Launch success count for all sites



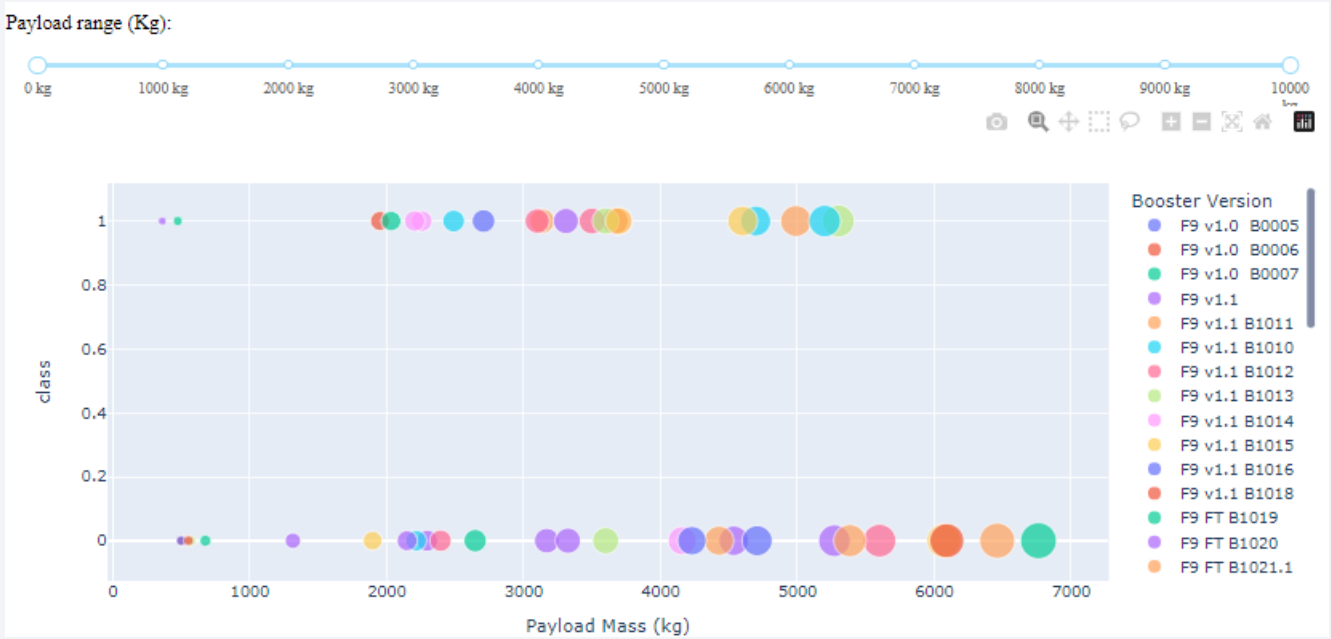
- The launch site KSC LC-39A has the best launch success rate (with 41.7%) compared to the other sites

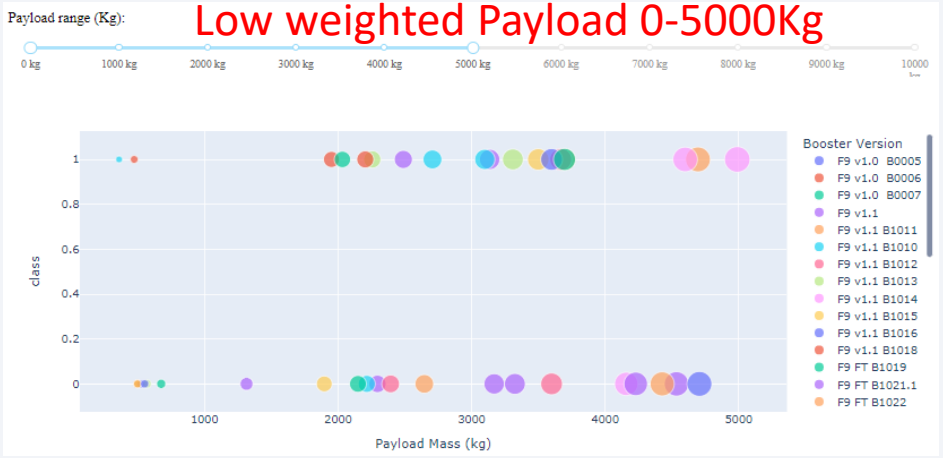
<Dashboard Screenshot 2>

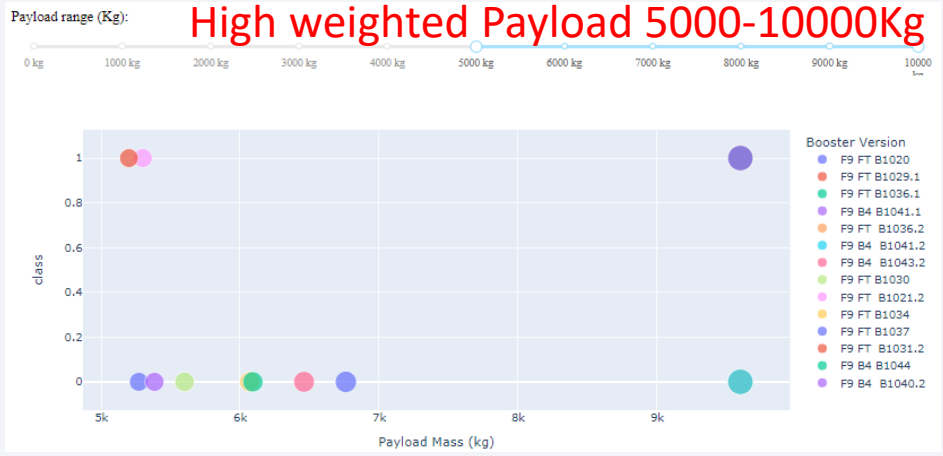


- Pie chart illustrates that KSC LC-39A site is the best successful launches with 76.9%

Low Payloads have higher success rate







- Scatter plot between launch outcome vs payload illustrates low weight payload shows better success rate compared to higher payloads

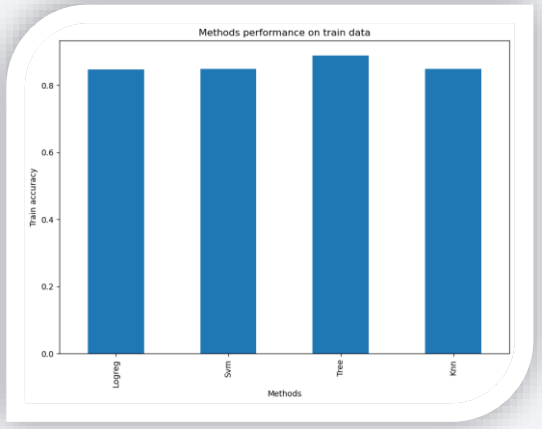
Section 5

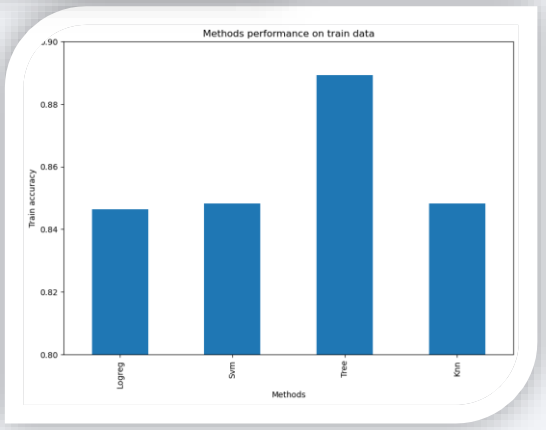
Predictive Analysis (Classification)

Classification Accuracy

- Following results were achieved after plotting the accuracy score and the best score for various classification algorithm
- The decision Tree model shows the highest accuracy
- The accuracy train score is 88.7%
- The accuracy test score is (83.33%) similar to the other model

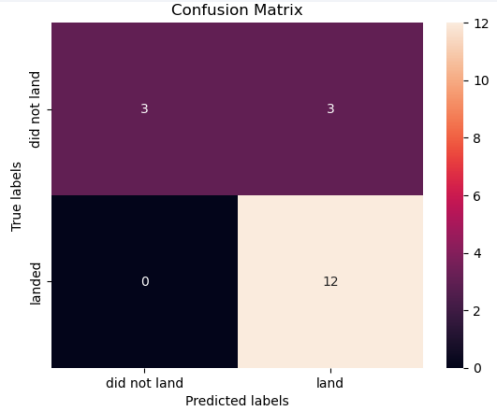
	Accuracy Train	Accuracy Test
Logreg	0.846429	0.833333
Svm	0.848214	0.833333
Tree	0.889286	0.833333
Knn	0.848214	0.833333



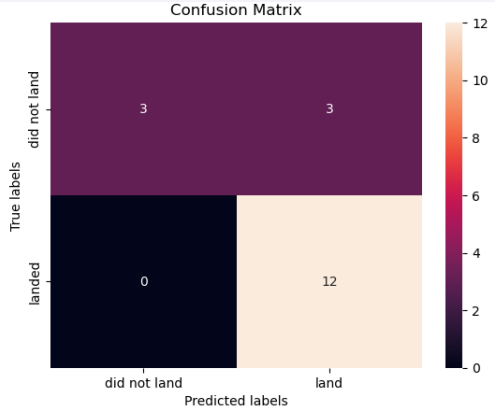


Confusion Matrix

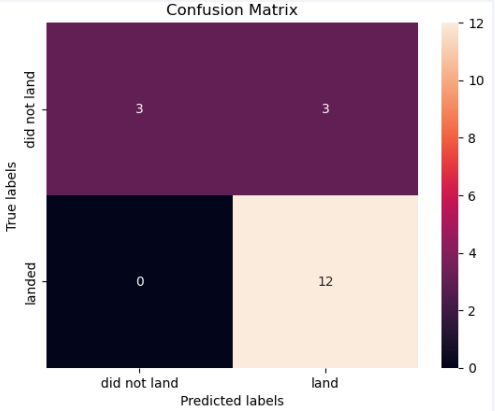
Tree



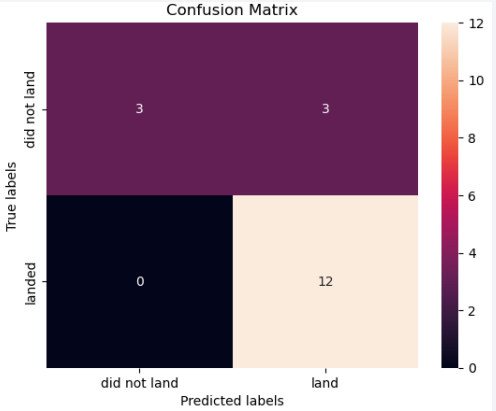
Log-regression



KNN



SVM



- As described previously, the test accuracy for all the model are equal. results
- Results from the confusion matrix are also very identical
- One of the major problem in all of those models are, false positive cases.

Conclusions

- The study analysis clearly demonstrates that as the success at a launch sites increases as the number of flights increases.
- Among the different orbit types, GEO, HEO, SSO, ES-L1 shows the best success rates
- Although GEO, HEO and ES-I1 have 100% success rate but they had only one flight to the respective orbits
- SSO have the 100% success rate with 5 successful flights
- PO, ISS and LEO orbits have most successful launches with heavy payloads.
- Launch site KSC LC-39A have the most successful launches with 41.7% of total success rate with personal success rate of 76.9%
- In general Payload below 5000 kg have higher success rate compared to the heavy payload
- Decision tree model seems to be better compared to the other models

Appendix

- SpaceX data was collected using Space X API
- Data was also collected from Wikipedia using beautifulSoup

Thank you!