IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Prateek Mishra
16/09/2022

Outline

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

Executive Summary

- Methodologies used were data collection, data wrangling, EDA with data visualization and SQL, creating interactive map with folium, using plotly dash to create a dashboard and predictive analysis using classification.
- The results contained include exploratory data analysis results, interactive analytics (including screenshot in this powerpoint), and the results of the predictive analysis

Introduction

- SpaceX is one of the global leaders in space data, and one of the most promising enterprises for the future of space exploration. The project is centered around SpaceX's Falcon 9 rocket launchers, culminating in a prediction of whether or not Falcon 9 will land successfully, as this prediction can effectively predict the cost of a launch.
- We want to find answers to the following problems:
  - Correlations between variables and launch success rate
  - Optimal conditions to achieve best possible landing rate

Section 1

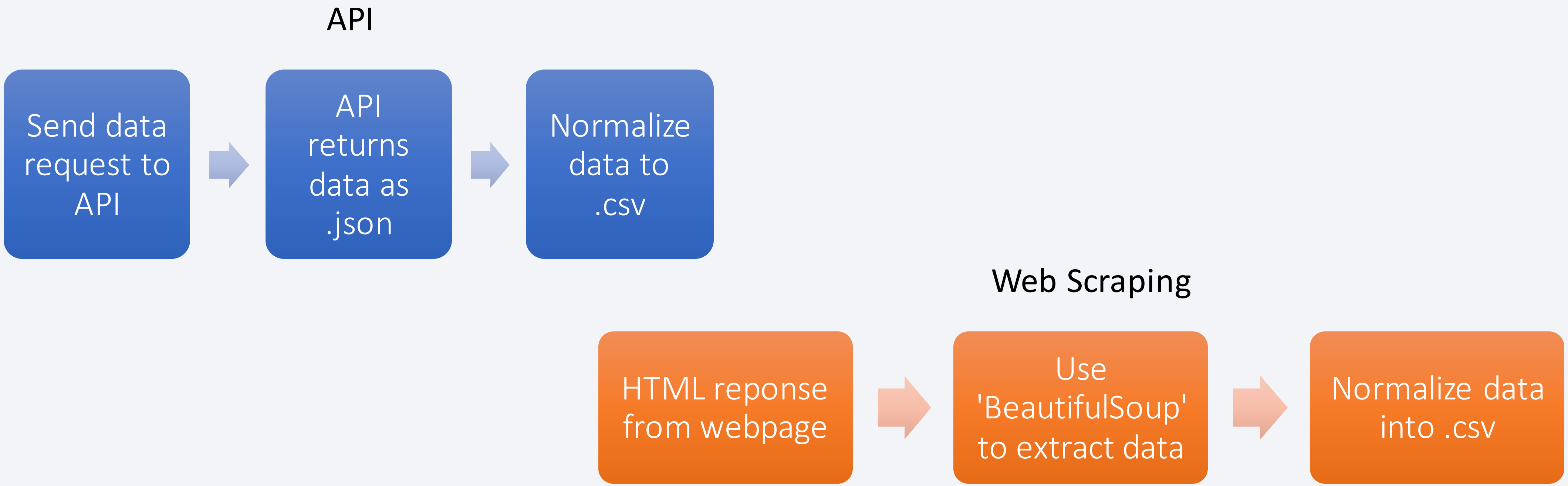
Methodology

Methodology

- Data collection methodology:
  - Data was collected through SpaceX API
  - Web scraping through wikipedia.
- Perform data wrangling
  - The outcomes of the launch were converted into training labels.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Finding the best hyperparameter for the model
  - Classification Trees, Logistic Regression

Data Collection

- Data was collected through both API and web scraping



Data Collection – SpaceX API

1. Get the url
2. Convert to .json
3. Clean data
4. Create dataframe
5. Export to .csv

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

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

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

```
# Lets take a subset of our dataframe keeping only the features we want a
nd 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 rocket
s 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 s
ingle 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 ex
tracting 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)]
```

github

Data Collection - Scraping

1. Get html response
2. Use BeautifulSoup
3. Assign tables to list
4. Extract columns from html header
5. Fill launch dict with records

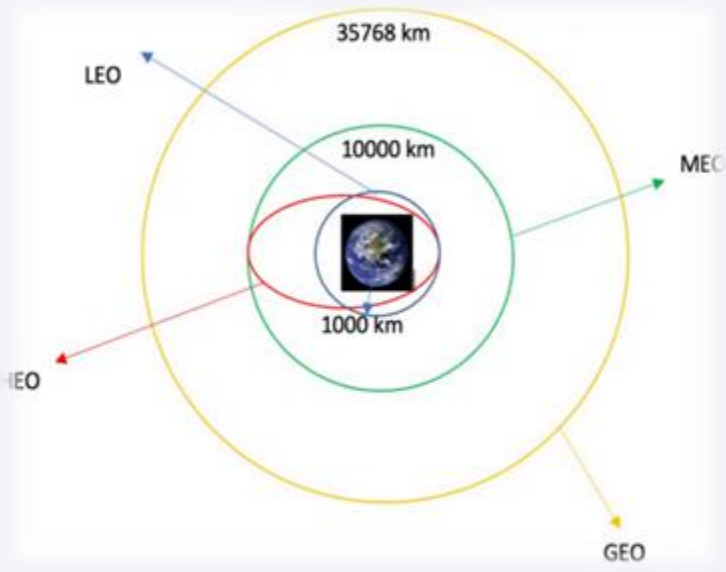
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response te
xt content
soup = BeautifulSoup(data,'html.parser')
```

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plai
nrowheaders collapsible")):
    # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding t
o launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
```

github

Data Wrangling

- Labels were created for the data points, where 1=pass, 0=fail
- Two outcome columns, namely 'Landing Location' and 'Mission Outcome'
- Possibilities were:
  - True Ocean
  - False Ocean
  - True RTLS
  - False RTLS
  - True ASDS
  - False ASDS



github

EDA with Data Visualization

- Scatterplots: They show dependency of two variables (eg Payload and Flight Number, or Payload and Orbit Type). They tell us basic relation between two parameters.
- Bar Graph: Based on the inference from scatterplots, we may formulate bar graphs, which give us a more detailed numerical comparison of data
- Line Graph: This is typically used to show the trend of the data, and is very useful for analysis.

github

EDA with SQL

- Displaying the names of the launch sites.
- Displaying 5 records where launch sites begin with the string ‘CCA’.
- Displaying the total payload mass carried by booster launched by NASA (CRS).
- Displaying the average payload mass carried by booster version F9 v1.1.
- Listing the date when the first successful landing outcome in ground pad was achieved.
- Listing the names of the booster versions which have carried the maximum payload mass.
- Listing the failed landing outcomes in drone ship, their booster versions, and launch sites names for in year 2015.
- Ranking the count of landing outcomes or success between the date 2010-06-04 and 2017-03-20, in descending order

github

Build an Interactive Map with Folium

- We took the data of the coordinates of launch sites and plotted it.
- We then assigned the dataframe launch_outcomes(failure,success) to classes 0 and 1 with Red and Green markers on the map in MarkerCluster().
- Haversine's formula was used to calculate the distance of the launch sites to various landmark to find
    - Proximity of launch sites to railways, highways and coastlines
    - Proximity of launch sites to nearby cities

github

Build a Dashboard with Plotly Dash

- The interactive dashboard was made with plotly, allowing the user to play around and experiment with the data
- The two major components were
  - PIE CHARTS
  - SCATTERPLOTS

[github]

Predictive Analysis (Classification)

- BUILDING : We built the model by performing EDA on existing data
- EVALUATING : Data was standardized and split into training and testing datasets
- IMPROVING : We tried to find the best hyperparameters for SVM Classification Trees and Logistic Regression
- TESTING : Prediction was found to be 83% accurate

github

Results

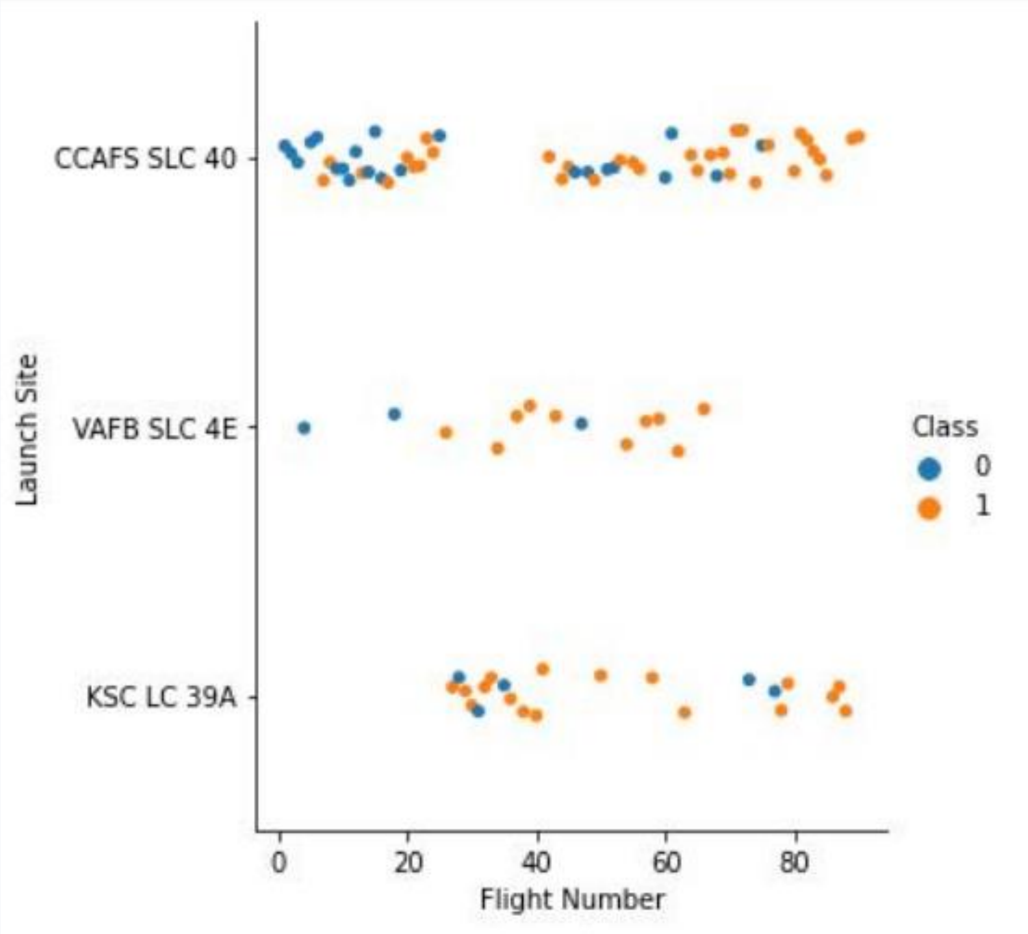
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

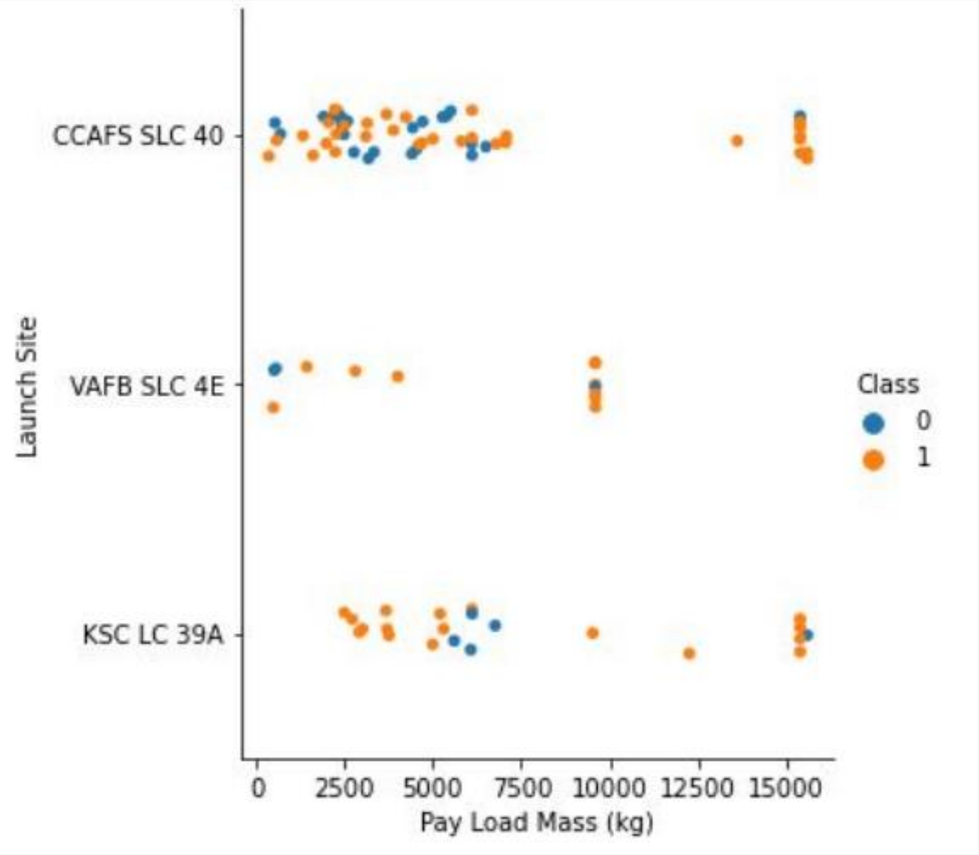
Flight Number vs. Launch Site

- Class 0 (blue) represents unsuccessful launch, and Class 1 (orange) represents successful launch.
- The larger the flights amount of the launch site, the greater the success rate will be.



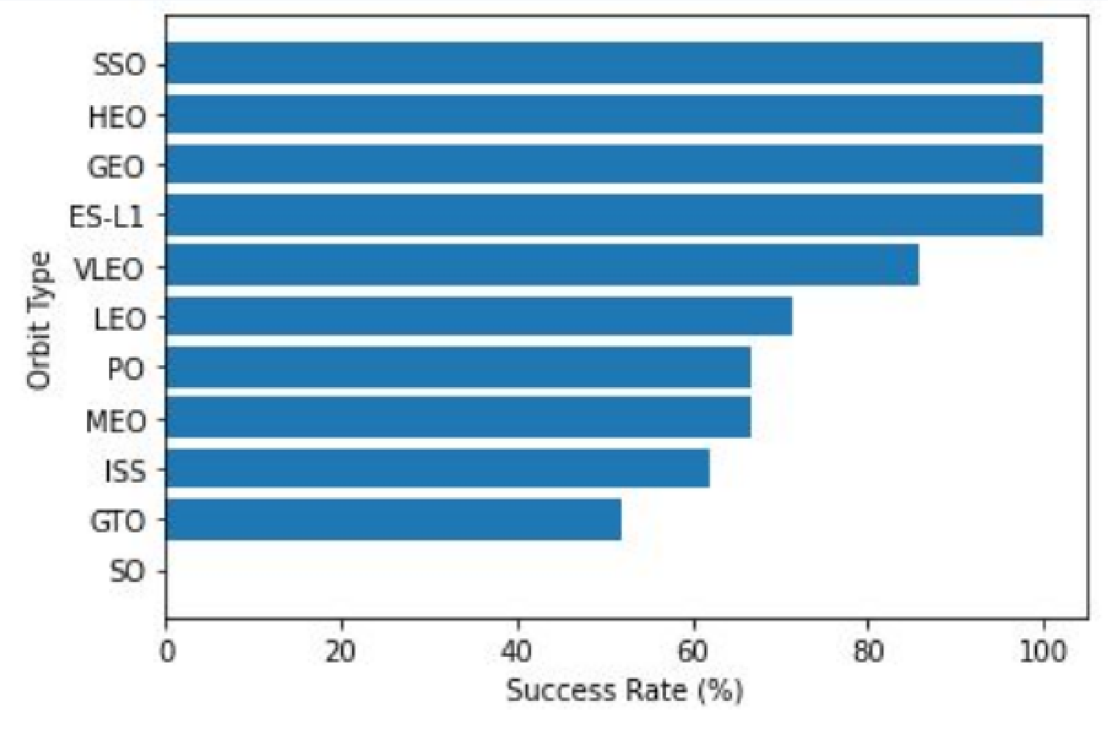
Payload vs. Launch Site

- Class 0 (blue) represents unsuccessful launch, and Class 1 (orange) represents successful launch
- Once the pay load mass is greater than 7000kg, the probability of the success rate will be highly increased.



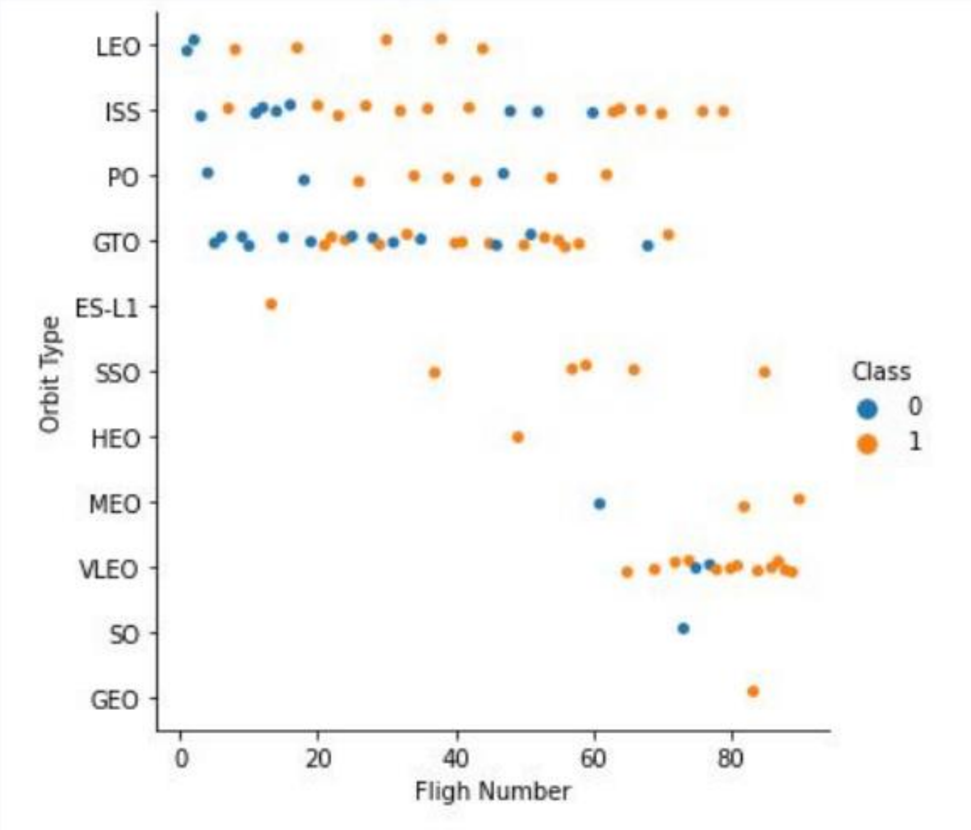
Success Rate vs. Orbit Type

- Orbit types SSO, HEO, GEO, and ES-L1 have the 100% success rates
- GTO has the lowest success rate at 50% among those that had a successful launch
- SO had 0 launches



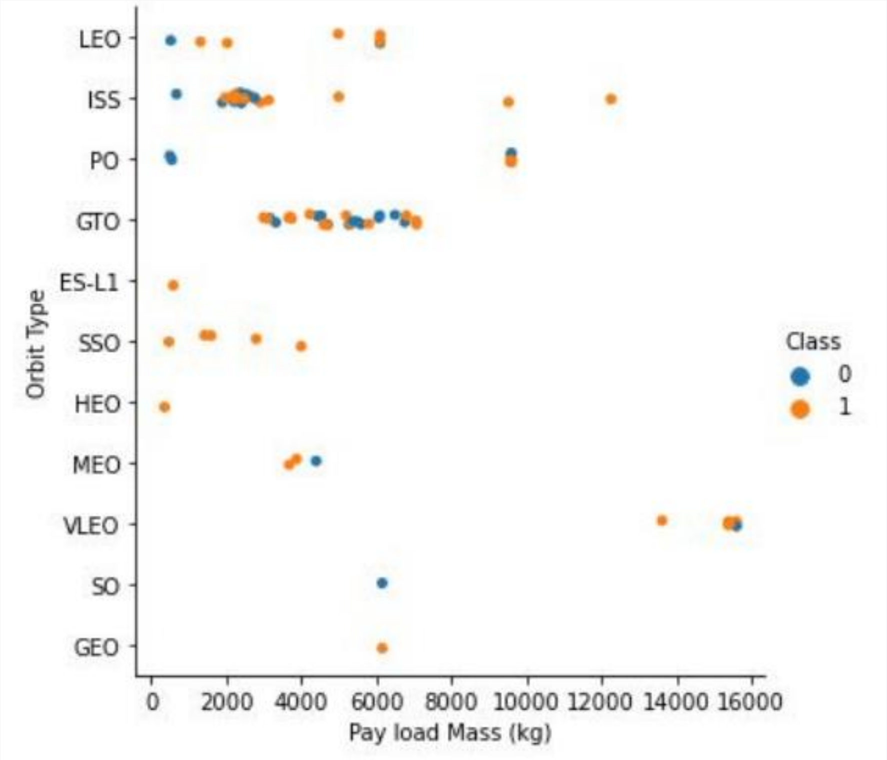
Flight Number vs. Orbit Type

- Class 0 (blue) represents unsuccessful launch, and Class 1 (orange) represents successful launch.
- Launch outcome is generally linked with FLight Number



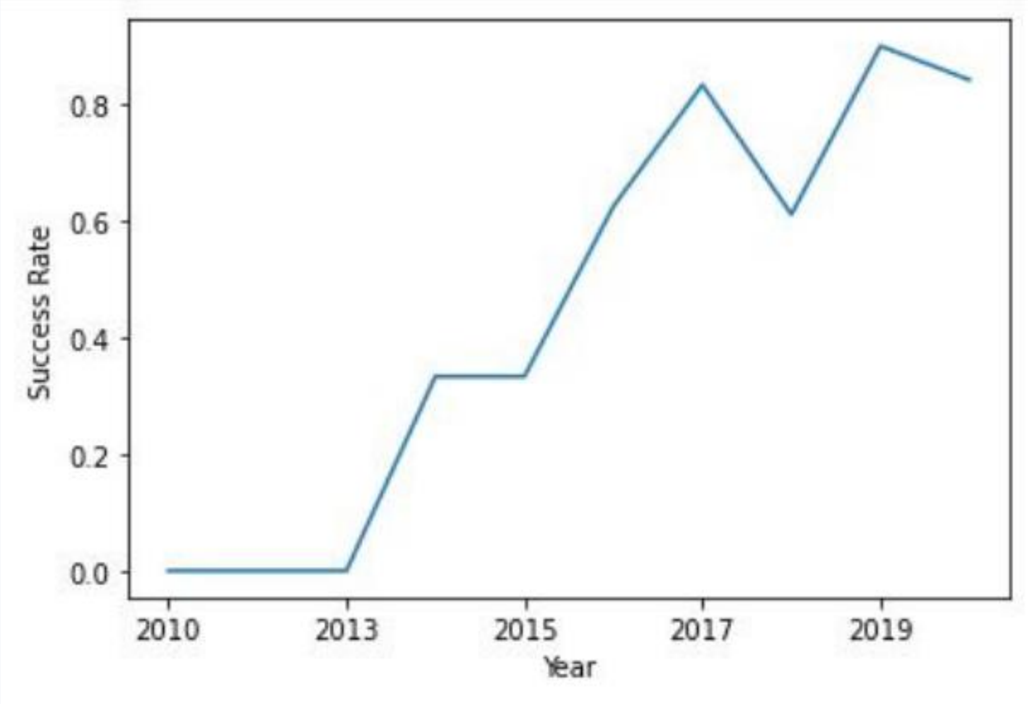
Payload vs. Orbit Type

- Class 0 (blue) represents unsuccessful launch, and Class 1 (orange) represents successful launch.
- With heavy payloads the successful landing or positive landing rate are more for LEO and ISS.



Launch Success Yearly Trend

- Static in 2010-13
- Increased in 2013-17
- Success rate dropped to 60% in 2018
- Post 2019 success rate is approx 80%



All Launch Site Names

SELECT DISTINCT selects all the launch sites once each

In [5]:

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

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Out[5]:

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

Launch Site Names Begin with 'CCA'

- We use the LIKE keyword here to find Sites beginning with CCA

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

In [11]:
```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	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
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- We use the SUM keyword here to return total payload mass

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Total Payload Mass by NASA (CRS)
45596

Average Payload Mass by F9 v1.1

- We have used the AVG keyword to return average payload mass

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu01qde00.databases.appdomain.cloud:32731/bludb
Done.
```

Average Payload Mass by Booster Version F9 v1.1
2928

First Successful Ground Landing Date

- The MIN() function has been used here to find the earliest date

```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

We have used multiple conditions using the AND keyword here

```
%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://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.datab
ases.appdomain.cloud:32731/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

List the total number of successful and failure mission outcomes

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Successful Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Success%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Successful Mission
100

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Failure Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Failure%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.clou
d:32731/bludb
Done.
```

Failure Mission
1

Boosters Carried Maximum Payload

- We have used the MAX function and the WHERE clause for this query

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEX
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX);
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.clou
d:32731/bludb
Done.
```

Booster Versions which carried the Maximum Payload Mass
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

2015 Launch Records

- We have used WHERE clause to 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 DATE LIKE '2015-%' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.
databases.appdomain.cloud:32731/bludb
Done.
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

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

- We have used the ORDER BY and GROUP BY functions to return 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 as "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" FROM SPACEX \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  LANDING__OUTCOME \
ORDER BY COUNT(LANDING__OUTCOME) DESC ;
```

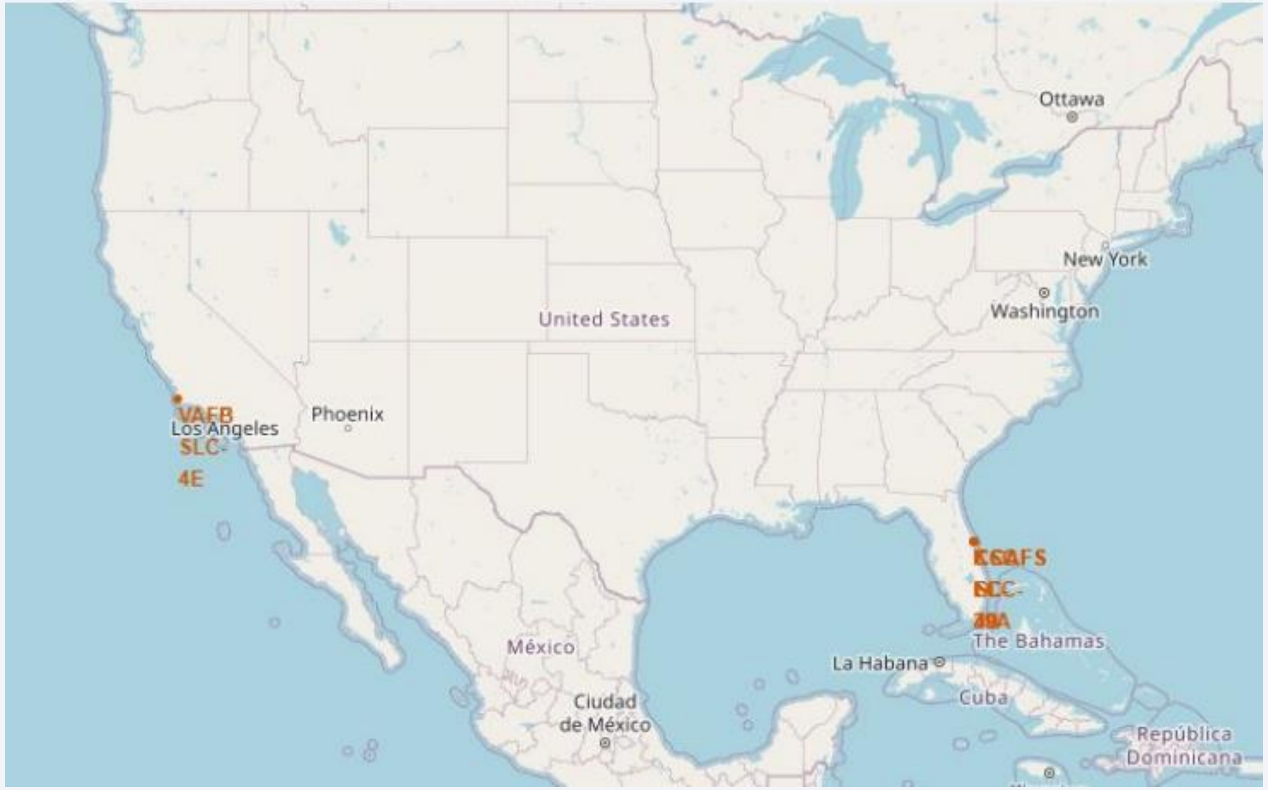
```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.c
loud:32731/bludb
Done.
```

Landing Outcome	Total Count
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

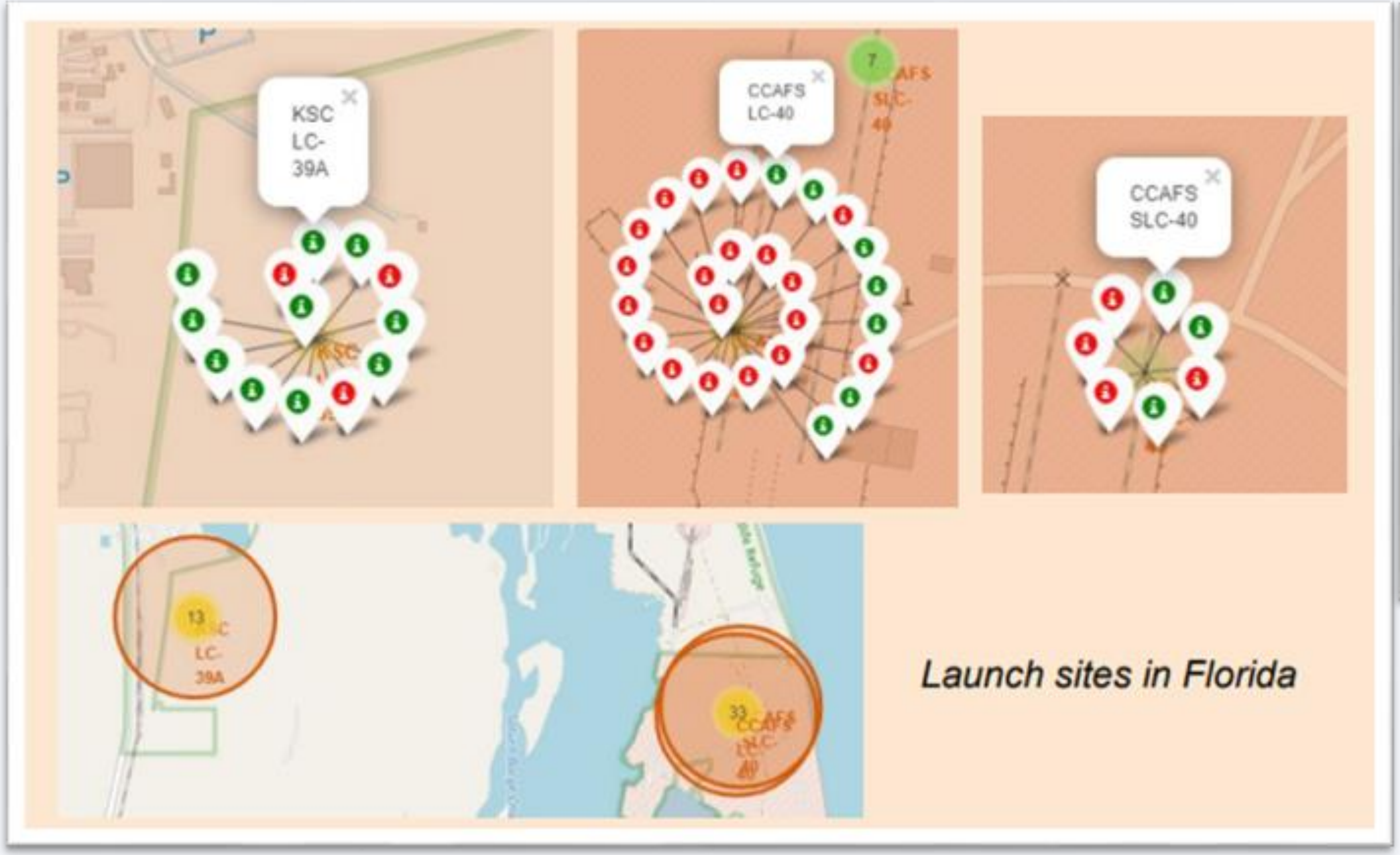
Launch Sites Proximities Analysis

Launch Site Map



This map shows the launch sites. It is evident that all these sites are located on the east or west coast.

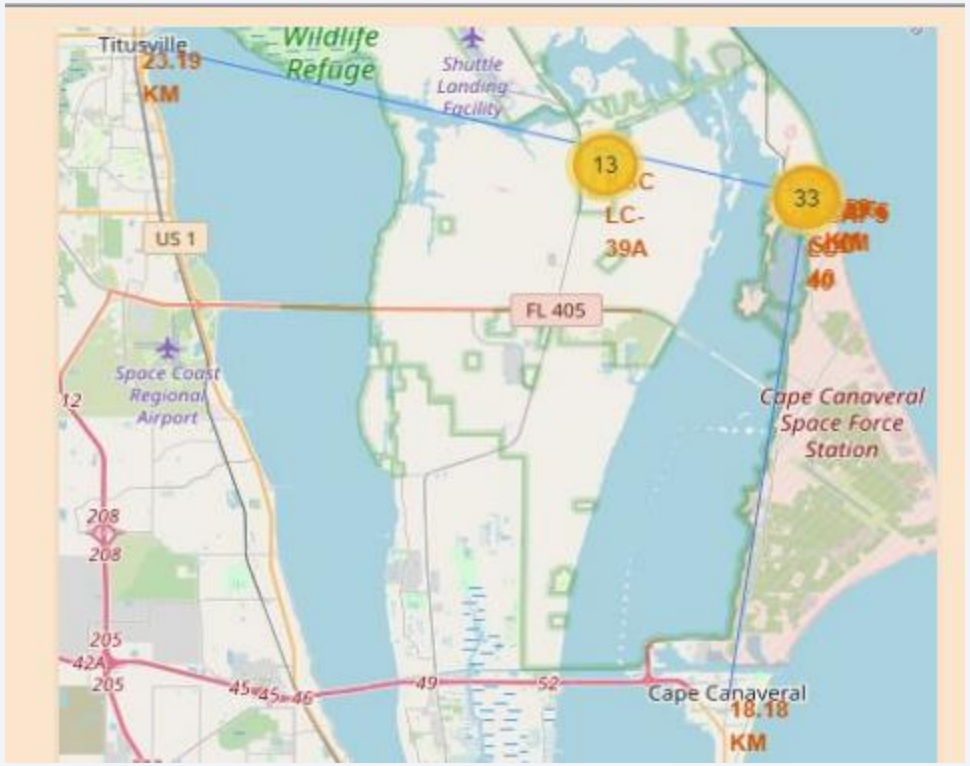
Launch Outcomes



Launch sites in Florida

Launch site in California

Proximity to Public Spaces



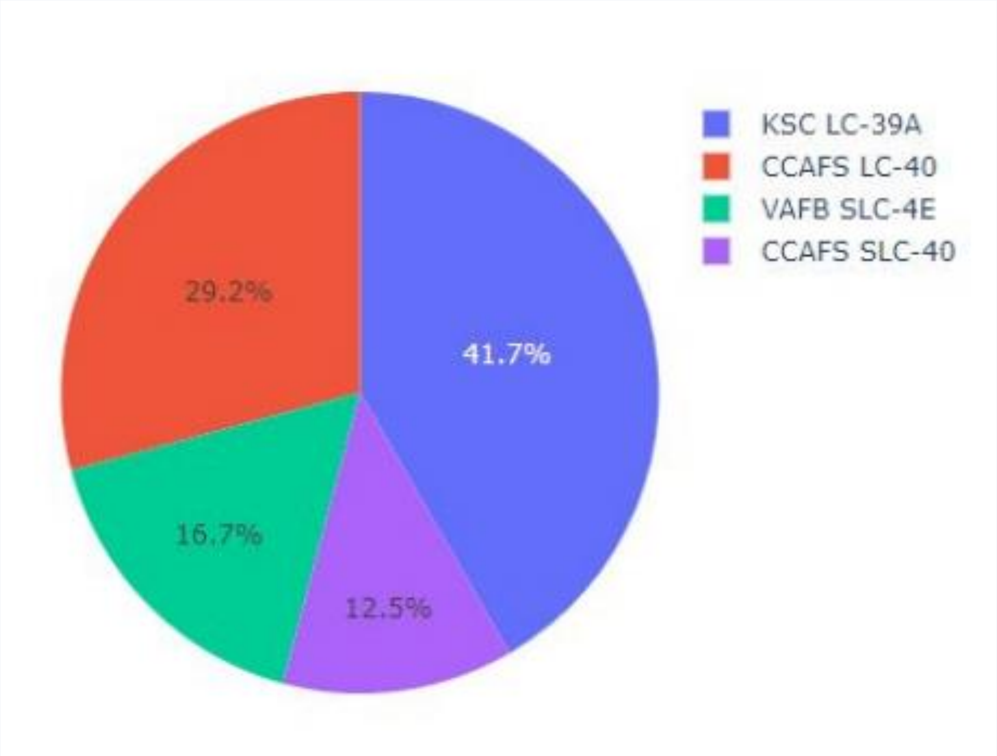
Launch sites are kept away from residential areas, highways and railways
They are kept in proximity to less populated coastlines

Section 4

Build a Dashboard with Plotly Dash

Launch Success Pie Chart

- KSLC-39A records the most launch success among all sites.

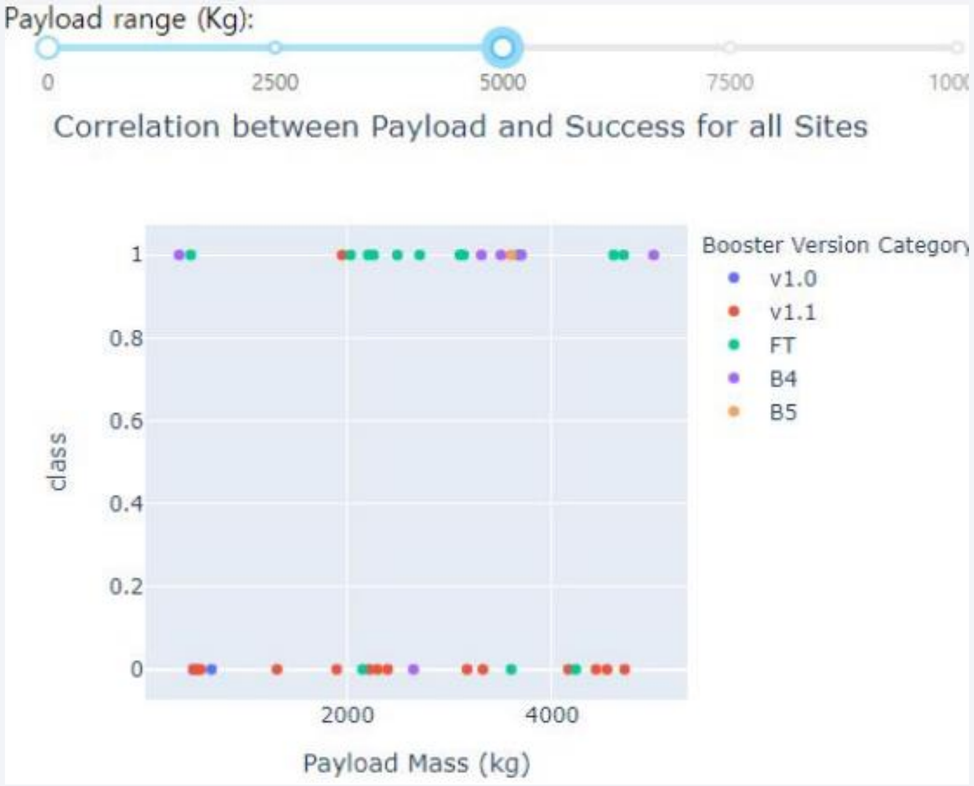


Launch Site Success Ratio

• KSC LC-39A HAS 76.9% success rate

Payload vs Launch Results

- These figures show that the launch success rate (class 1) for low weighted payloads(<5000 kg) is higher than that of heavy weighted payloads (>5000 kg)

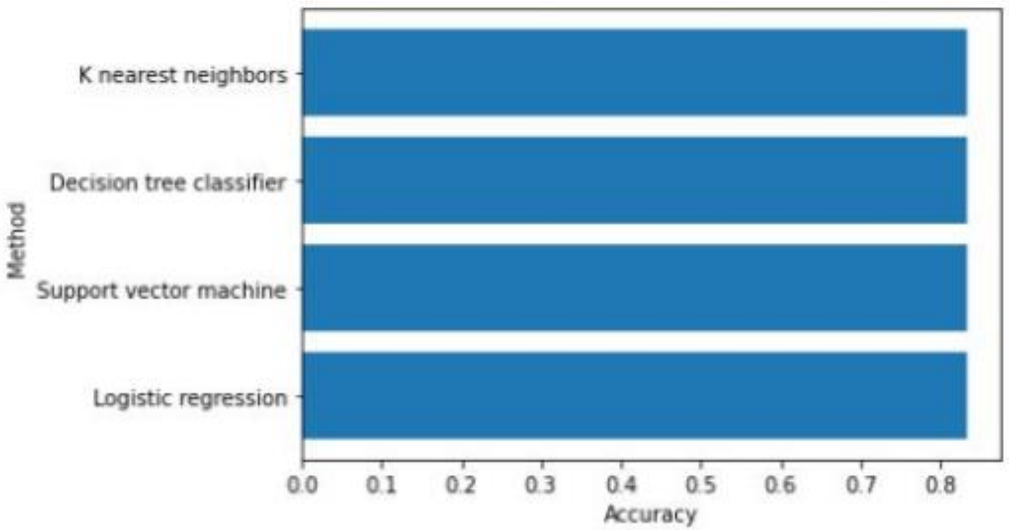


Section 5

Predictive Analysis (Classification)

Classification Accuracy

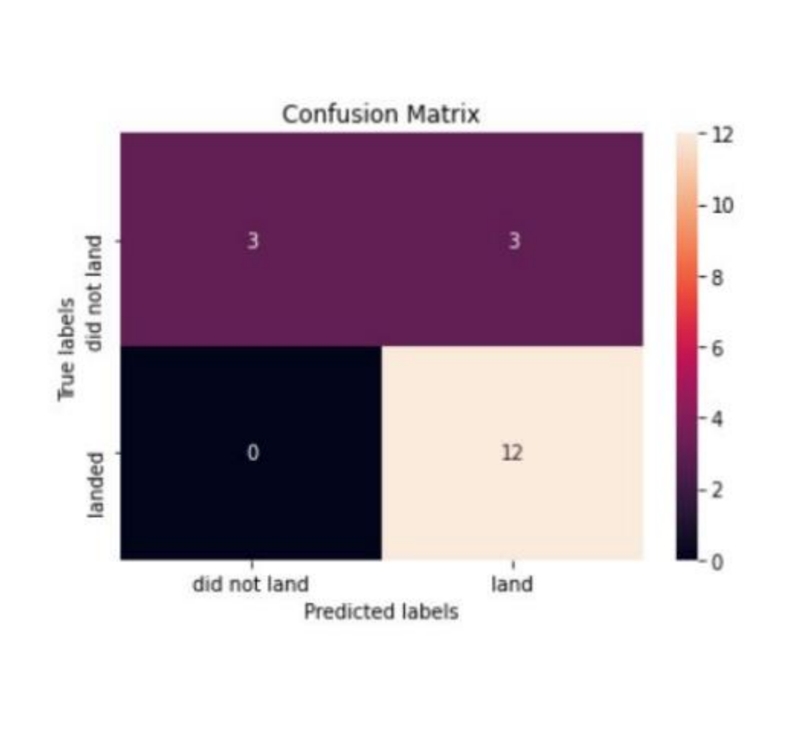
- In the test set, the accuracy of all models was virtually the same at 83.33%.
- However, the dataset was small, meaning that the accuracy can be increased



	Method	Accuracy
0	Logistic regression	0.833333
1	Support vector machine	0.833333
2	Decision tree classifier	0.833333
3	K nearest neighbors	0.833333

Confusion Matrix

- The confusion matrix is the same for all models because all models performed the same for the test set.
- The models are good at predicting successful landings



Conclusions

- As the number of flights increased, the success rate increased, and recently it has exceeded 80%.
- Orbital types SSO, HEO, GEO, and ES-L1 have the highest success rate (100%).
- The launch site is close to railways, highways, and coastline, but far from cities.
- KSLC-39A has the highest number of launch successes and the highest success rate among all sites.
- The launch success rate of low weighted payloads is higher than that of heavy weighted payloads.
- In this dataset, all models have the same accuracy (83.33%), but it seems that more data is needed to determine the optimal model due to the small data size.

Appendix

Github Repository: https://github.com/D4RK-ness/IBM-Applied-Data-Science-Capstone

Thank you!