IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Tony
2022-06-25

Outline

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

Executive Summary

- **Summary of methodologies:**
- Data Collection through API;
- Data Collection with Web Scraping;
- Exploratory Data Analysis (EDA) with Data Visualization;
- Data Wrangling;
- EDA with SQL & Data Visualization;
- Interactive Visual Analytics & Dashboards ;
- Machine Learning Prediction.

- **Summary of all results:**

- Collecting valuable data from public sources;
- Exploratory Data Analysis Results to identify which features are the best to predict success of launchings;
- Interactive Analytics;
- Predictive Analytics Result.

Introduction

- **Project background and context:**

SpaceX is a revolutionary company who has disrupt the space industry by offering a rocket launches specifically Falcon 9 as low as 62 million dollars; while other providers cost upward of 165 million dollar each. Most of this saving thanks to SpaceX astounding idea to reuse the first stage of the launch by re-land the rocket to be used on the next mission. Repeating this process will make the price down even further. As a data scientist of a startup rivaling SpaceX ,The objective is to evaluate the viability of the new company Space Y to compete with Space X.

In Brief:

- Space X has best pricing ($62 million vs. $165 million USD)
- Largely due to ability to recover part of rocket (Stage 1)
- Space Y wants to compete with Space X

- **The Problems:**

• The main characteristics of a successful or failed landing and Identifying all factors that influence the landing outcome.

• The relationship between each variables on the success or failure of a landing and how it is affecting the outcome.

• The conditions which will allow SpaceX to achieve the best landing success rate and The best condition needed to increase the probability of successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API
  - Web Scrapping from Wikipedia
- Perform data wrangling
  - Dropping unnecessary columns
  - One Hot Encoding for classification models
- 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 that was collected until this step were normalized, divided in training and test data sets and evaluated by four different classification models, being the accuracy of each model evaluated using different combinations of parameters.

Data Collection

- How data sets were collected :

Datasets are collected from Rest SpaceX API and web scrapping Wikipedia

  - The information obtained by the API are rocket, launches, payload information.
  - The Space X REST API URL is api.spacexdata.com/v4/

SpaceX Rest API Call → API returns JSON file → Make Data frame from JSON → Clean Data and export

- Data collection process :

For web scrapping, we will use the Beautiful Soup to extract the launch records as HTML table, parse the table and convert it to a pandas data frame for further analysis.

Get HTML response from Wikipedia → Extract data with Beautiful Soup → Make Dataframe → Export Data

Data Collection – SpaceX API

- Data collection with SpaceX REST calls :

```python
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)]
```

Getting Request For Rocket Lunch data (Using APIs)

Getting .JSON file + Lists(Launch Site, Booster Version, Payload Data)

Using Json_normalize Function to DataFrame data from JSON

Filter data to only include Falcon 9 launches

Impute missing PayloadMass values with mean

GitHub URL:

https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

- We applied web scrapping to web scrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas data frame.

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

```
# 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
```

Request Wikipedia html

BeautifulSoup Html.lib Parser

Find and Extract All Column names from Header

Convert tables into Data Frame

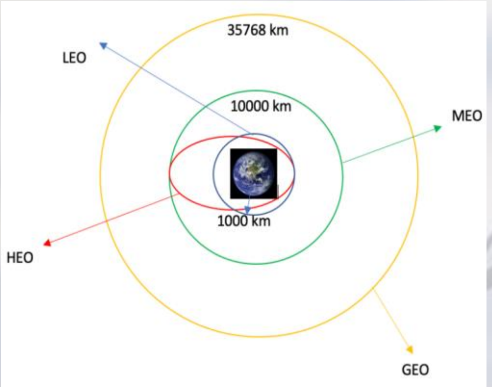
GitHub URL:

https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/jupyter-labs-webscraping.ipynb

Data Wrangling

Some Exploratory Data Analysis (EDA) was performed on the dataset, then we calculated the number of launches at each site, and the number and occurrence of each orbits and last we created landing outcome label from outcome column and exported the results to csv.

1. Calculate launches number for each site
2. Calculate the number and occurrence of each orbit
3. Calculate number and occurrence of mission outcome per orbit type
4. Create landing outcome label from Outcome column
5. Export to file




```
df['LaunchSite'].value_counts()

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

```
df['Orbit'].value_counts()

GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
SO        1
ES-L1     1
HEO       1
GEO       1
Name: Orbit, dtype: int6
```

```
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes

True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
None ASDS       2
False Ocean     2
False RTLS      1
Name: Outcome, dtype: int64
```

```
landing_class = []
for key,value in df["Outcome"].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
df['Class']=landing_class
```

```
df.to_csv("dataset_part_2.csv", index=False)
```

GitHub URL:

https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/Data_Wrangling.ipynb

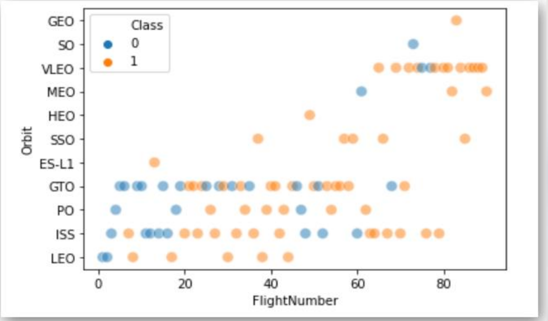
EDA with Data Visualization

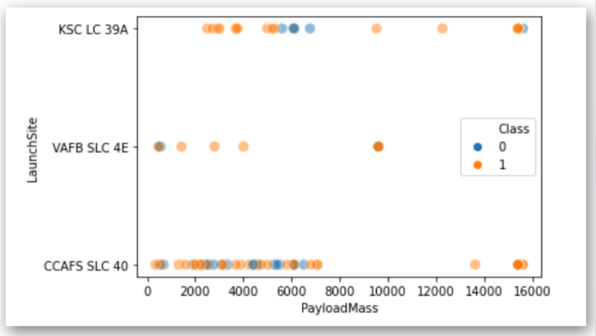
We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.

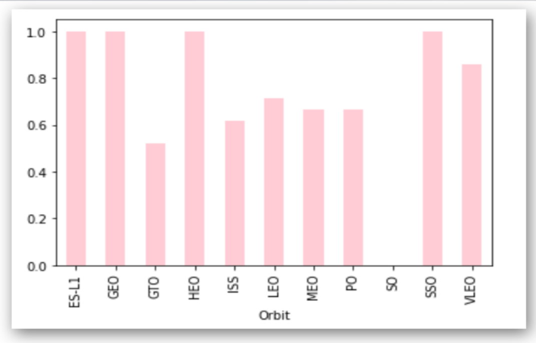
Scatter plots, line charts, and bar plots were used to compare relationships between variables to decide if a relationship exists so that they could be used in training the machine learning model

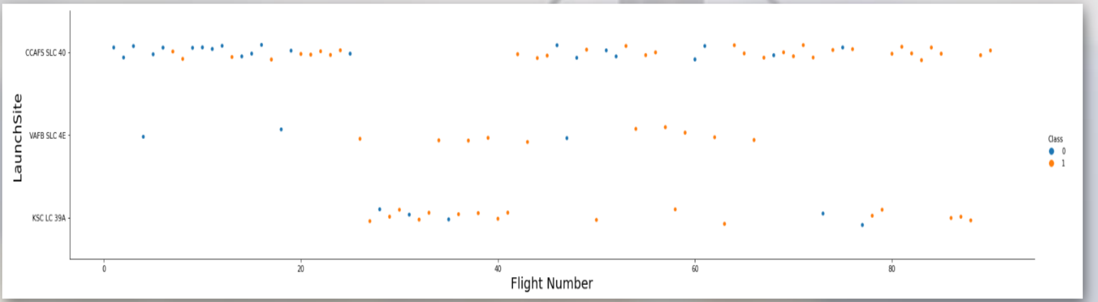
GitHub URL:

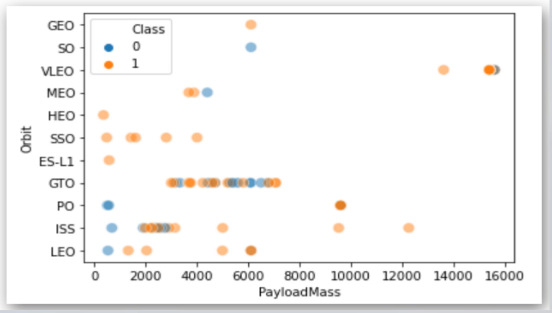
https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/jupyter-labs-eda-dataviz.ipynb











https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

The following SQL queries were performed:

- Names of the unique launch sites in the space mission;
- Top 5 launch sites whose name begin with the string 'CCA’;
- Total payload mass carried by boosters launched by NASA (CRS);
- Average payload mass carried by booster version F9 v1.1;
- Date when the first successful landing outcome in ground pad was achieved;
- Names of the boosters which have success in drone ship and have payload mass between 4000 and 6000 kg;
- Total number of successful and failure mission outcomes;
- Names of the booster versions which have carried the maximum payload mass;
- Failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015; and
- Rank of the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20.

GitHub URL:

https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/jupyter-labs-eda-sql-coursera.ipynb

Build an Interactive Map with Folium

Folium map object is a map centered on NASA Johnson Space Center at Houson, Texas :

• Red circle at NASA Johnson Space Center's coordinate with label showing its name (**folium.Circle, folium.map.Marker**).
• Red circles at each launch site coordinates with label showing launch site name (**folium.Circle**, **folium.map.Marker**, **folium.features.DivIcon**).
• The grouping of points in a cluster to display multiple and different information for the same coordinates (**folium.plugins.MarkerCluster**).
• Markers to show successful and unsuccessful landings. **Green** for successful landing and **Red** for unsuccessful landing.(We assigned the dataframe launch_outcomes(failure, success) to classes 0 and 1 with Red and Green markers on the map in MarkerCluster()) .
• Markers to show distance between launch site to key locations (railway, highway, coastline, city) and plot a line between them.

These objects are created in order to understand better the problem and the data. We can show easily all launch sites, their surroundings and the number of successful and unsuccessful landings.

GitHub URL:

https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/Visual_Analytics_with_Folium.ipynb

Build a Dashboard with Plotly Dash

Dashboard has dropdown, pie chart, slider and scatter plot components

- Built an interactive dashboard with Plotly dash
- Pie charts showing the total launches by a certain sites
- Scatter chart shows the relationship between two variables, in particular Success vs Payload Mass
- Dropdown allows a user to choose the launch site or all launch sites
- Dashboard includes a pie chart and a scatter plot.
- Pie chart can be selected to show distribution of successful landings across all launch sites and can be selected to show individual launch site success rates.

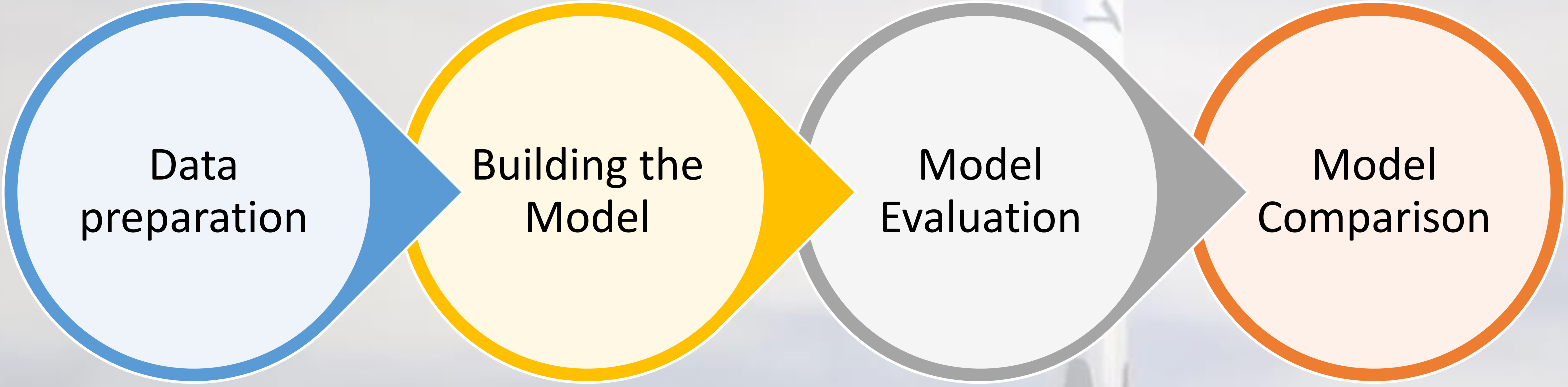
GitHub URL:

https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/Spacex_Dash_App.py

Predictive Analysis (Classification)

- Load dataset
- Normalize data
- Split data into training and test sets.

- Check accuracy for each Model
- Set the best parameters and algorithms to GridSearchCV and fit it to dataset.
- Training GridSearchModel models with training dataset



Find the Best Model.The model with the best accuracy score will be the best performing model.

- Decide which type of ML to use
- set the parameters and algorithms to GridSearchCV and fit it to dataset.

GitHub URL:

https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/Predictive_Analysis_-_Machine_Learning_Lab.ipynb

Results

Exploratory data analysis results

Interactive analytics demo in screenshots

Predictive analysis results

Some Results :

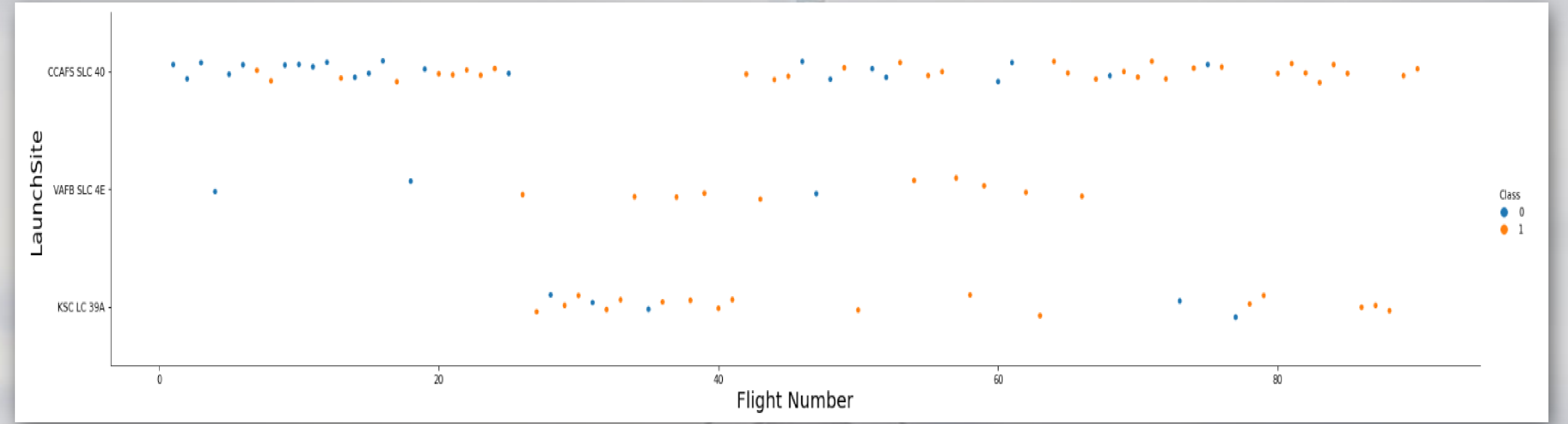
- Space X uses 4 different launch sites;
- The first launches were done to Space X itself and NASA;
- The average payload of F9 v1.1 booster is 2,928 kg;
- The first success landing outcome happened in 2015 fiver year after the first launch;
- Two booster versions failed at landing in drone ships in 2015: F9 v1.1 B1012 and F9 v1.1 B1015;
- The number of landing outcomes became as better as years passed.
- Predictive Analysis showed that Decision Tree Classifier is the best model to predict successful landings, having high accuracy
- …

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

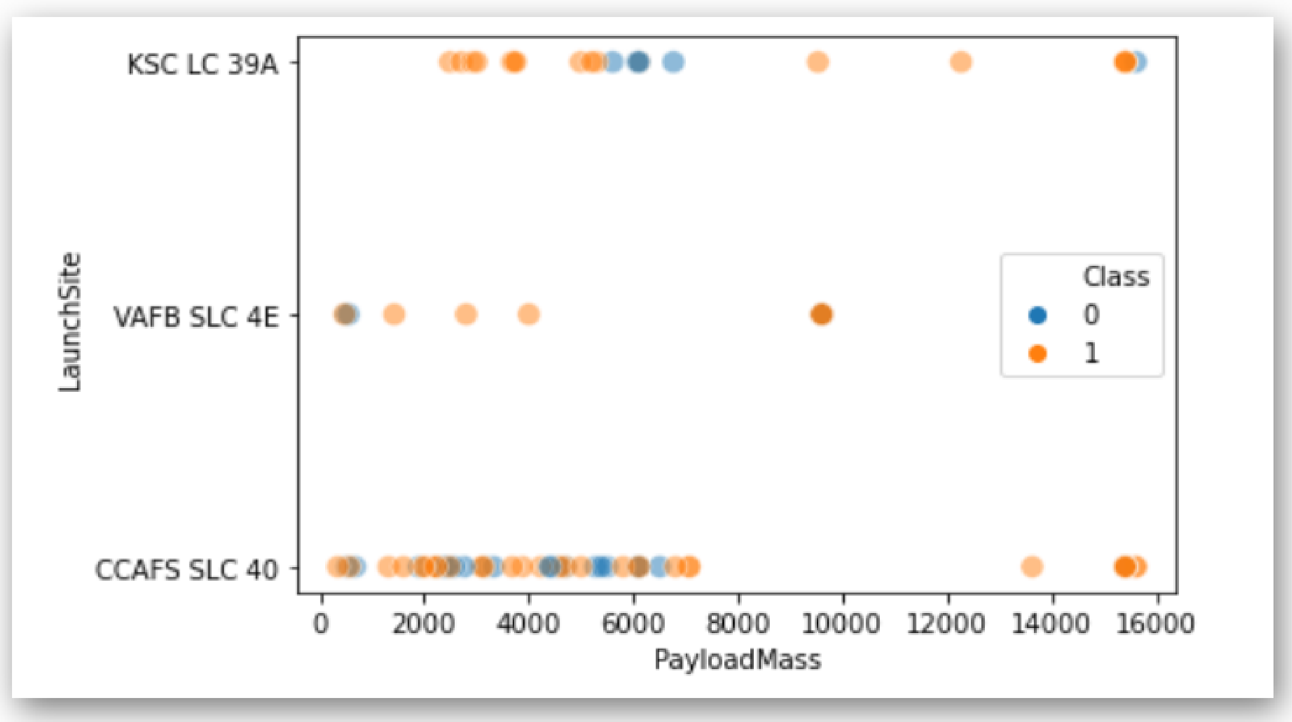
Orange indicates successful launch; Blue indicates unsuccessful launch.



https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/jupyter-labs-eda-dataviz.ipynb

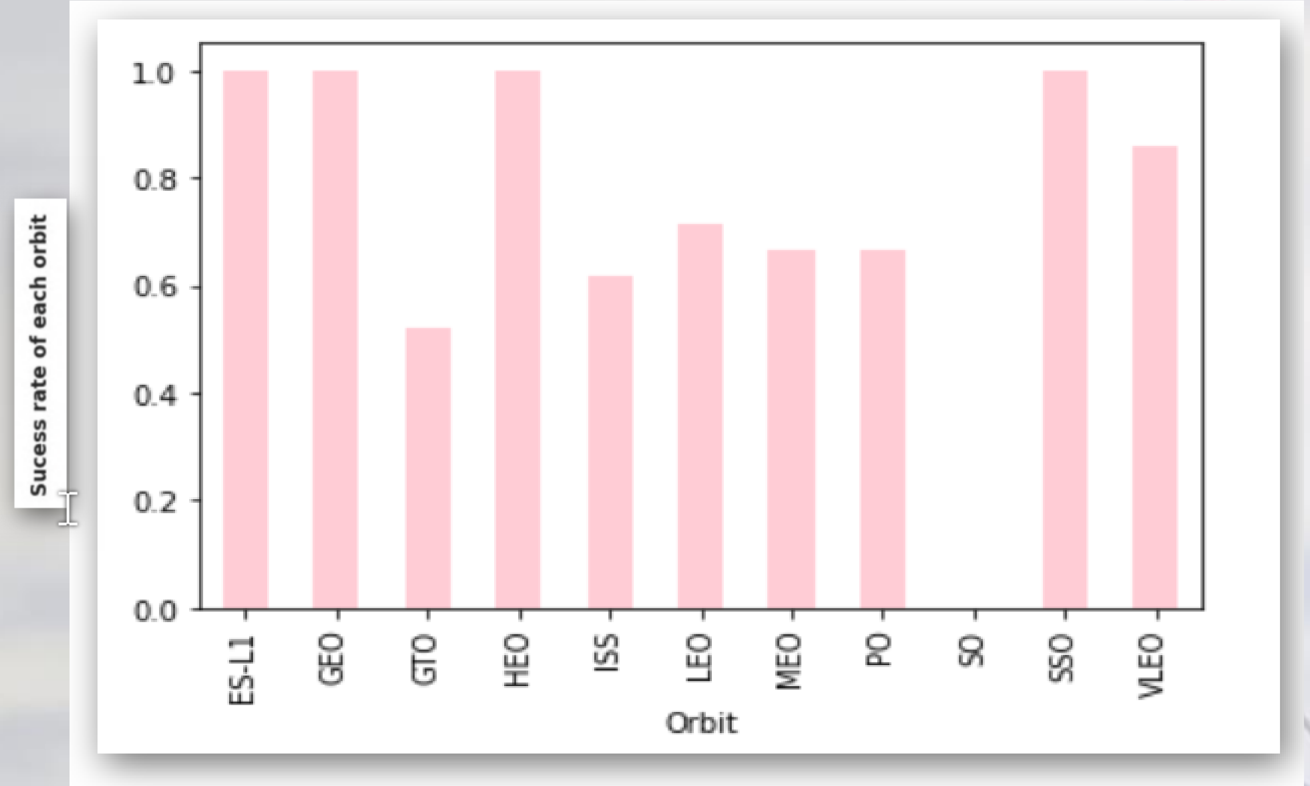
- The plot shows that an increase in success rate over time (indicated in Flight Number).
- According to the plot above, it's possible to verify that the best launch site nowadays is CCAF5 SLC 40, where most of recent launches were successful;

Payload vs. Launch Site



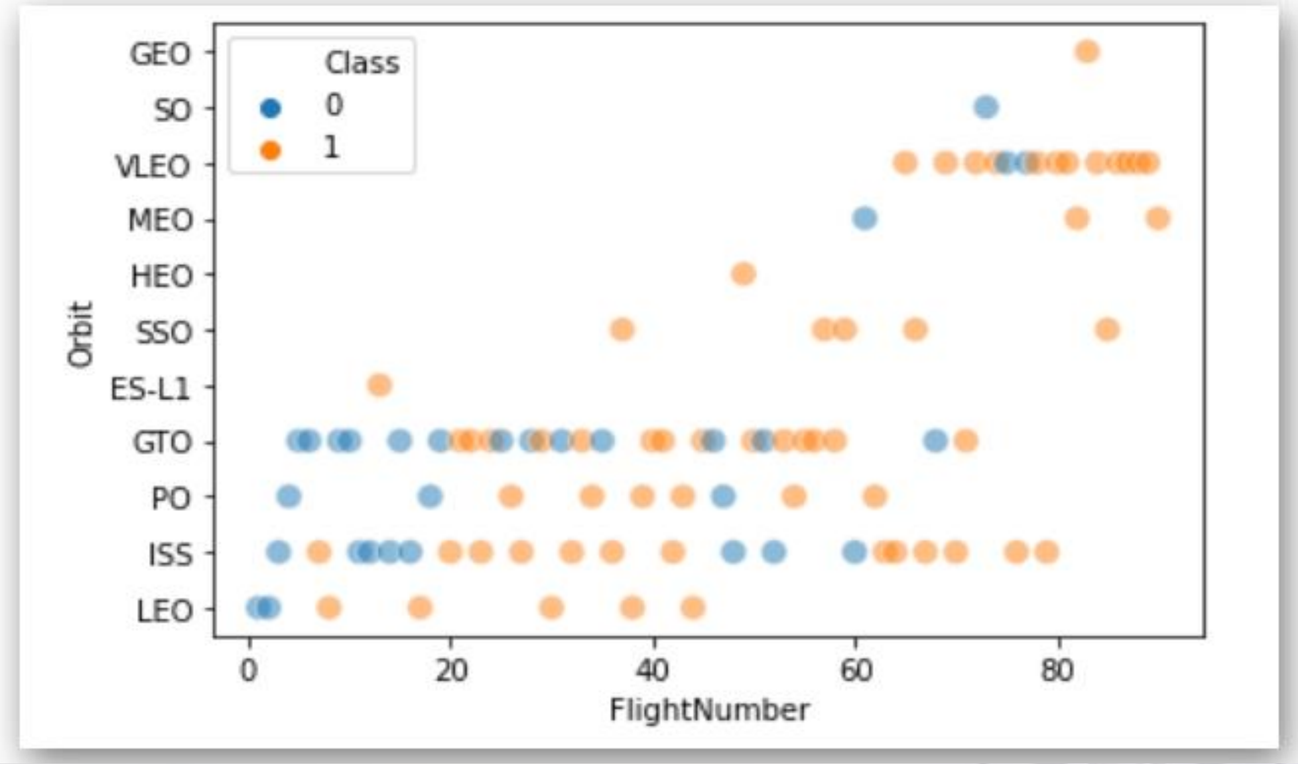
- Payload mass appears to fall mostly between 0-7000 kg
- This plot shows once the pay load mass is greater than 7000kg, the probability of the success rate will be highly increased
- Payloads over 9,000kg have excellent success rate;
- Payloads over 12,000kg seems to be possible only on CCAFS SLC 40 and KSC LC 39A launch sites.
- Depending on the launch site, A heavier payload may be a consideration for a successful landing.

Success Rate vs. Orbit Type



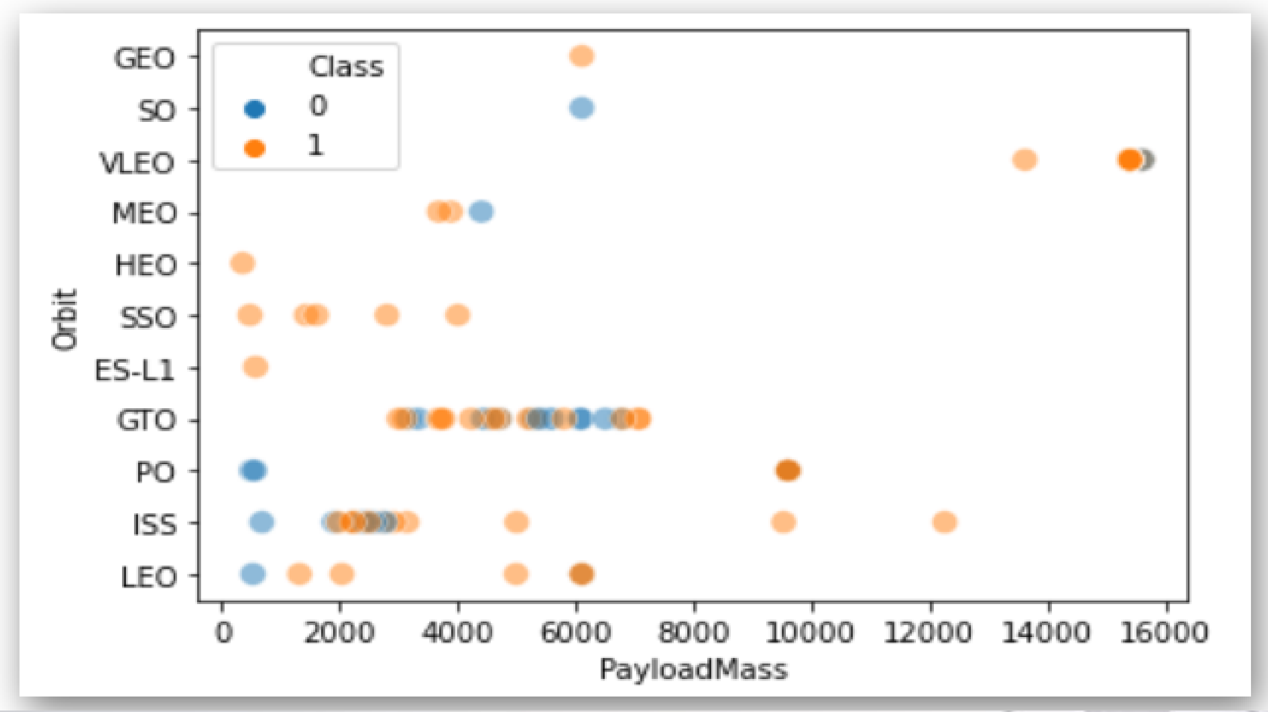
According to Plot we can see success rate for different orbit types.
We note that ES-L1, GEO, HEO, SSO have the best success rate.

Flight Number vs. Orbit Type



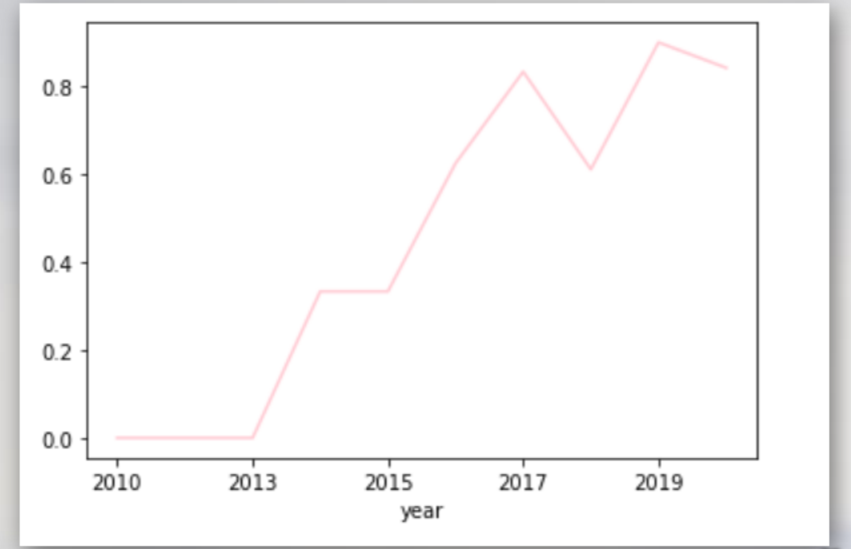
We observe that the success rate increases with the number of flights for the LEO orbit ,so success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.

Payload vs. Orbit Type



We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.
Heavier payload has negative impact on MEO and VLEO orbit.
For SO, GEO and HEO orbit need more dataset to see any pattern or trend.

Launch Success Yearly Trend



https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/jupyter-labs-eda-dataviz.ipynb

The plot shows that success rate since 2013 kept on increasing till 2020.

Success in recent years at around 80%

All Launch Site Names

Using the key word DISTINCT to show only unique launch sites from the SpaceX data.

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

In [18]:
```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL;
```

```
 * ibm_db_sa://nbc67803:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[18]:

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

Launch Site Names Begin with 'CCA'

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

In [19]:
```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

```
 * ibm_db_sa://nbc67803:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[19]:

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

Displaying 5 records where launch sites begin with `CCA`

Total Payload Mass

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

In [20]:
```
%sql SELECT SUM(payload_mass__kg_) FROM SPACEXTBL WHERE customer = 'NASA (CRS)';
```

```
 * ibm_db_sa://nbc67803:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[20]:

1
45596

The query sums the total payload mass in kg where NASA CRS was the customer.

CRS = Commercial Resupply Services
which indicates that these payloads were sent to the International Space Station (ISS).

Average Payload Mass by F9 v1.1

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

In [21]:
```
%sql SELECT AVG(payload_mass__kg_) FROM SPACEXTBL WHERE booster_version = 'F9 v1.1';
```

```
 * ibm_db_sa://nbc67803:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[21]:

1
2928

- The query results the average payload mass carried by booster version F9 v1.1
- Average payload mass of F9 1.1 is on the low end of our payload mass range

First Successful Ground Landing Date

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

Hint:Use min function

In [24]:
```
%sql SELECT MIN(DATE) FROM NBC67803.SPACEXTBL WHERE mission_outcome = 'Success';
```

```
 * ibm_db_sa://nbc67803:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[24]:

1
2010-06-04

Query give us the oldest successful landing. The WHERE clause filters dataset in order to keep only records where landing was successful. With the MIN function, we select the record with the oldest date.

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

In [29]:
```
%sql SELECT booster_version FROM NBC67803.SPACEXTBL WHERE "LANDING__OUTCOME" = 'Success (drone ship)' and payload_mass__kg_ BETWEEN 4000 and 6000;
```

```
 * ibm_db_sa://nbc67803:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[29]:

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

Used the WHERE clause to List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000.

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%';
```

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

Successful Mission
100

Failure Mission
1

By using wildcard like '%' can filter for WHERE Mission_Outcome was a success or a failure.

Boosters Carried Maximum Payload

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

In [33]:
```
%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://nbc67803:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[33]:

booster_version
F9 v1.0 B0003
F9 v1.0 B0004
F9 B4 B1045.1
F9 FT B1038.1
F9 v1.0 B0006
F9 v1.1 B1003
F9 v1.0 B0005
F9 v1.1 B1017
F9 v1.1 B1013
F9 v1.0 B0007

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

2015 Launch Records

In [35]:
```
%sql SELECT booster_version, launch_site FROM (SELECT * FROM NBC67803.SPACEXTBL WHERE "LANDING__OUTCOME" LIKE 'Failure%' and DATE LIKE '2015%');
```

```
 * ibm_db_sa://nbc67803:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[35]:

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

The query returns the Booster Version and Launch site of 2015 launches where stage 1 failed to land on a drone ship.

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

In [38]:
```
%sql SELECT * FROM NBC67803.SPACEXTBL WHERE "LANDING__OUTCOME" = 'Failure (drone ship)' and DATE BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY "LAN
```

```
 * ibm_db_sa://nbc67803:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[38]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2015-01-10	09:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
2016-01-17	18:42:00	F9 v1.1 B1017	VAFB SLC-4E	Jason-3	553	LEO	NASA (LSP) NOAA CNES	Success	Failure (drone ship)
2016-03-04	23:35:00	F9 FT B1020	CCAFS LC-40	SES-9	5271	GTO	SES	Success	Failure (drone ship)
2016-06-15	14:29:00	F9 FT B1024	CCAFS LC-40	ABS-2A Eutelsat 117 West B	3600	GTO	ABS Eutelsat	Success	Failure (drone ship)

SQL Query

```
%sql SELECT "LANDING _OUTCOME", COUNT("LANDING _OUTCOME") FROM SPACEXTBL\
WHERE "DATE" >= '04-06-2010' and "DATE" <= '20-03-2017' and "LANDING _OUTCOME" LIKE '%Success%'\
GROUP BY "LANDING _OUTCOME" \
ORDER BY COUNT("LANDING _OUTCOME") DESC ;
```

Results

Landing _Outcome	COUNT("LANDING _OUTCOME")
Success	20
Success (drone ship)	8
Success (ground pad)	6

This query returns landing outcomes and their count where mission was successful and date is between 04/06/2010 and 20/03/2017. The GROUP BY clause groups results by landing outcome and ORDER BY COUNT DESC shows results in decreasing order.

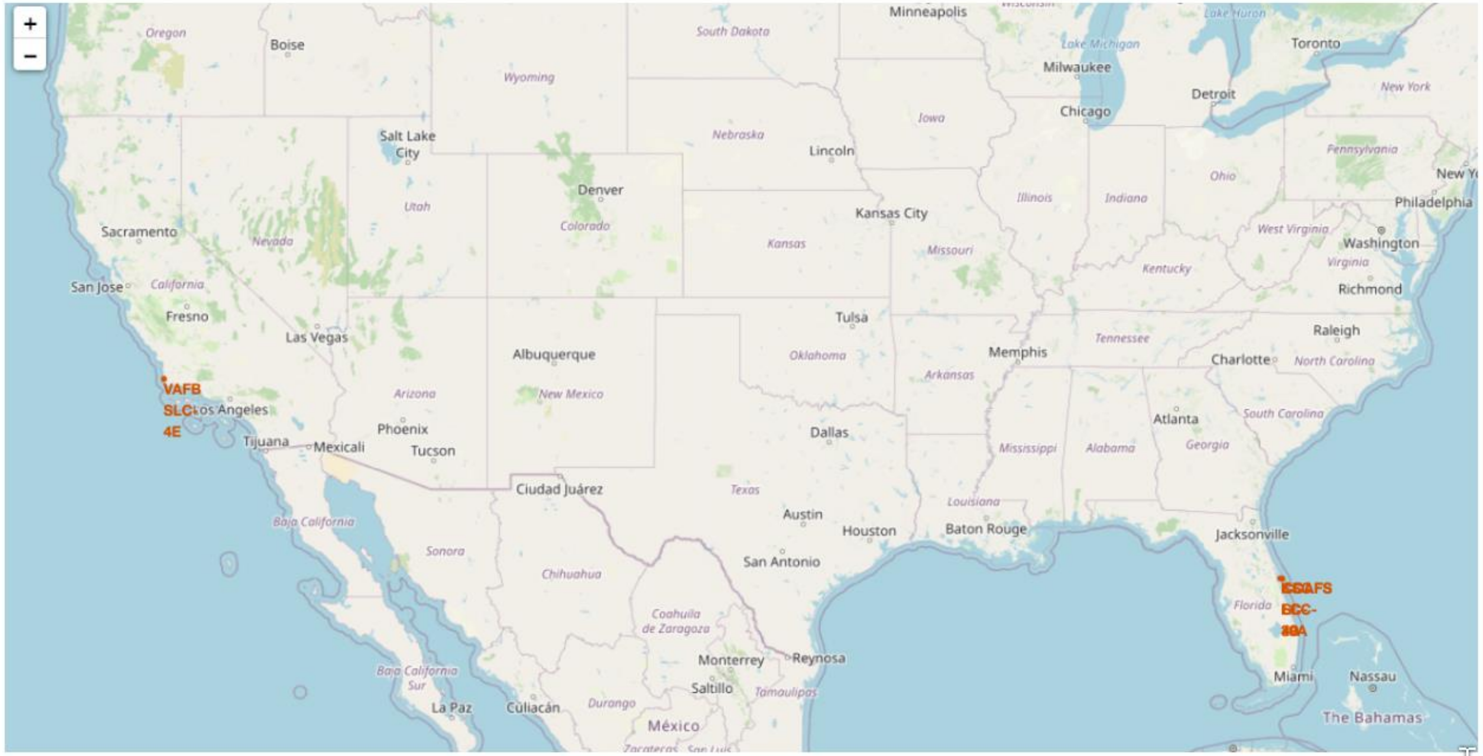
Section 3

Launch Sites Proximities Analysis

Location of all the Launch Sites

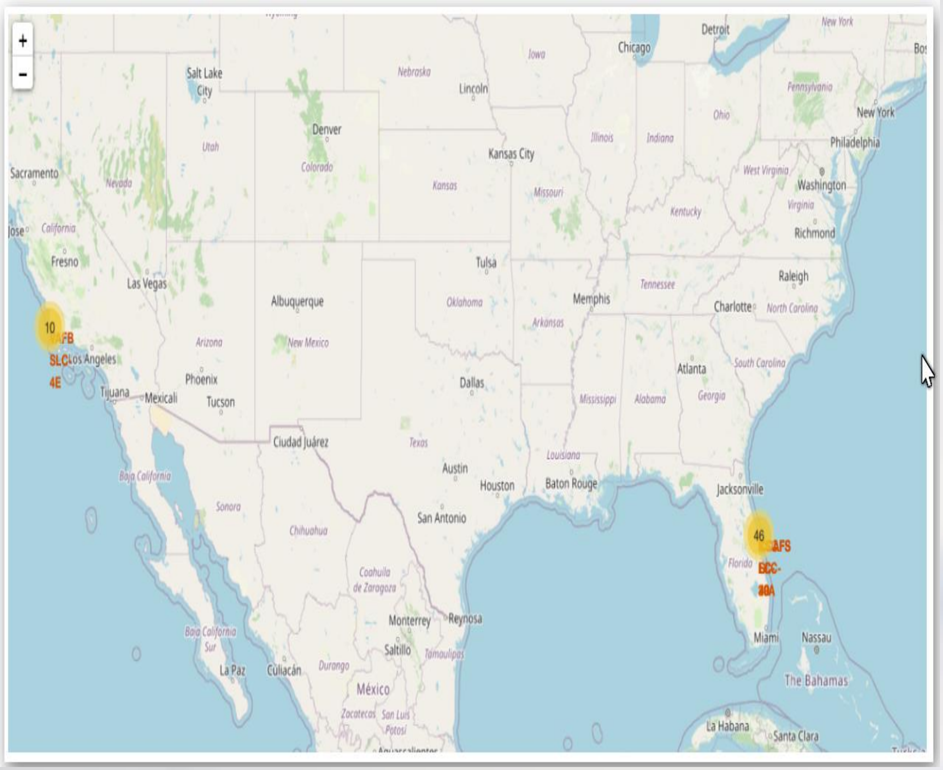
```
# Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
# For each launch site, add a Circle object based on its coordinate (Lat, Long) values. In addition, add Launch site name as a popup label
```

The generated map with marked launch sites should look similar to the following:



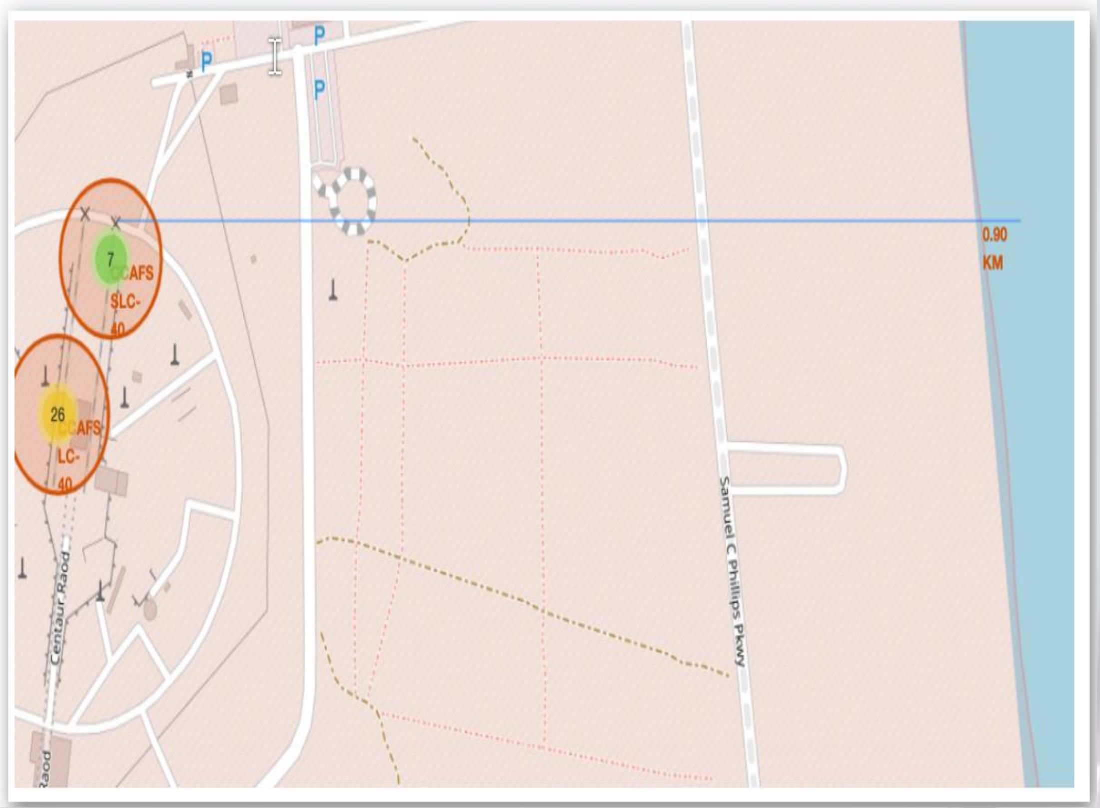
All the SpaceX launch sites are located inside the United States.

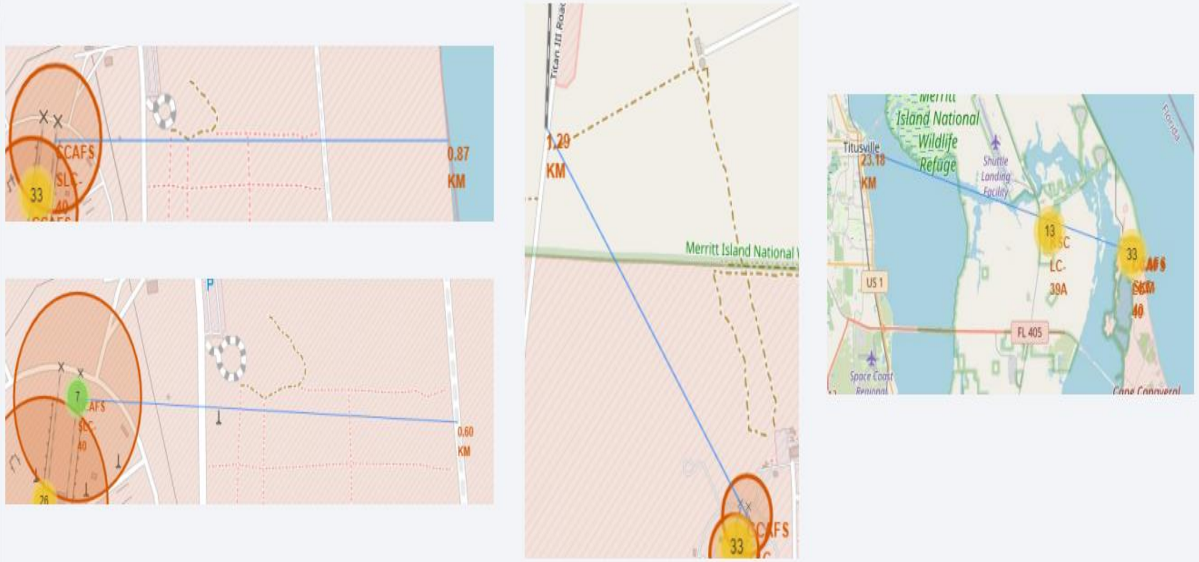
launch sites Markers with color labels



Green marker represents successful launches. **Red** marker represents unsuccessful launches for KSC LC-39A has a higher launch success rate

The distances between launch site CCAFS SLC-40 to its proximities



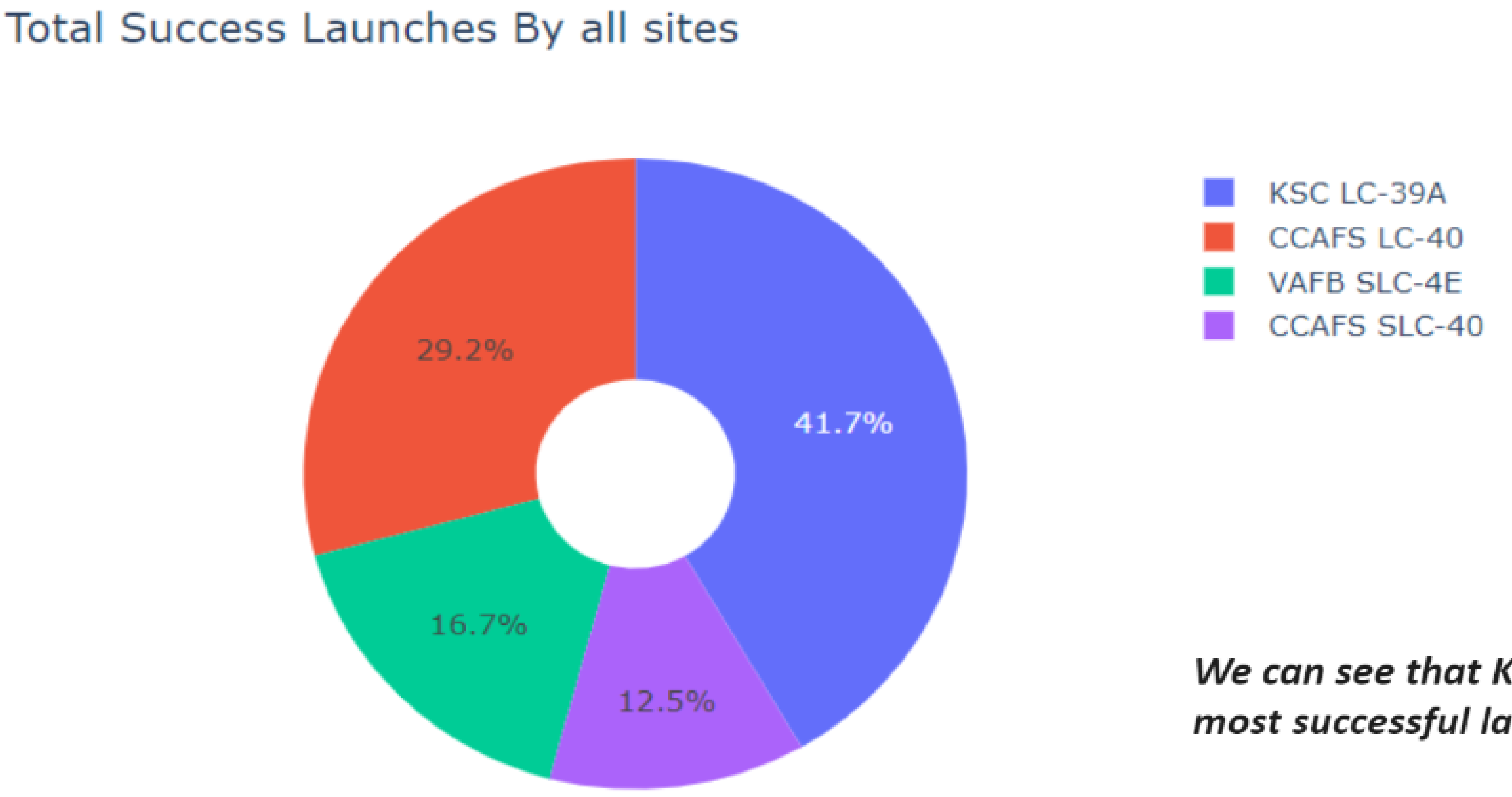


Is CCAFS SLC-40 in close proximity to railways ? Yes
Is CCAFS SLC-40 in close proximity to highways ? Yes
Is CCAFS SLC-40 in close proximity to coastline ? Yes
Do CCAFS SLC-40 keeps certain distance away from cities ? No

Section 4

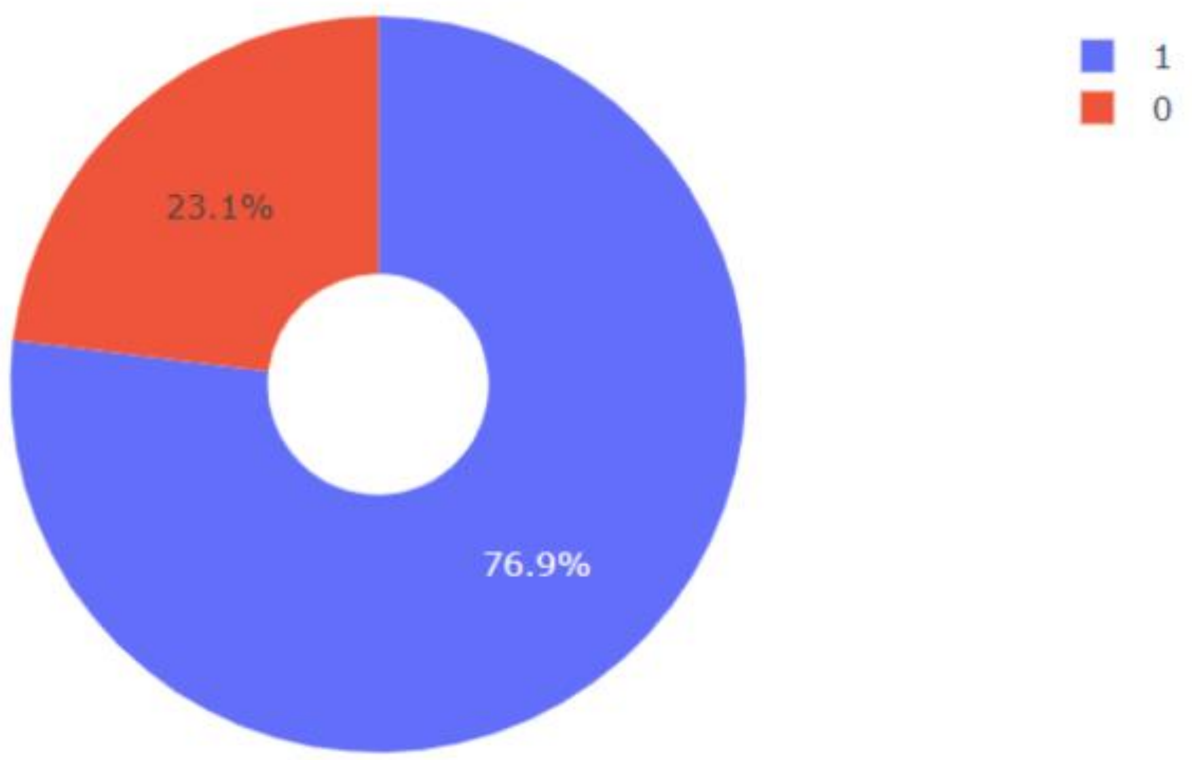
Build a Dashboard with Plotly Dash

The success percentage achieved by each launch site



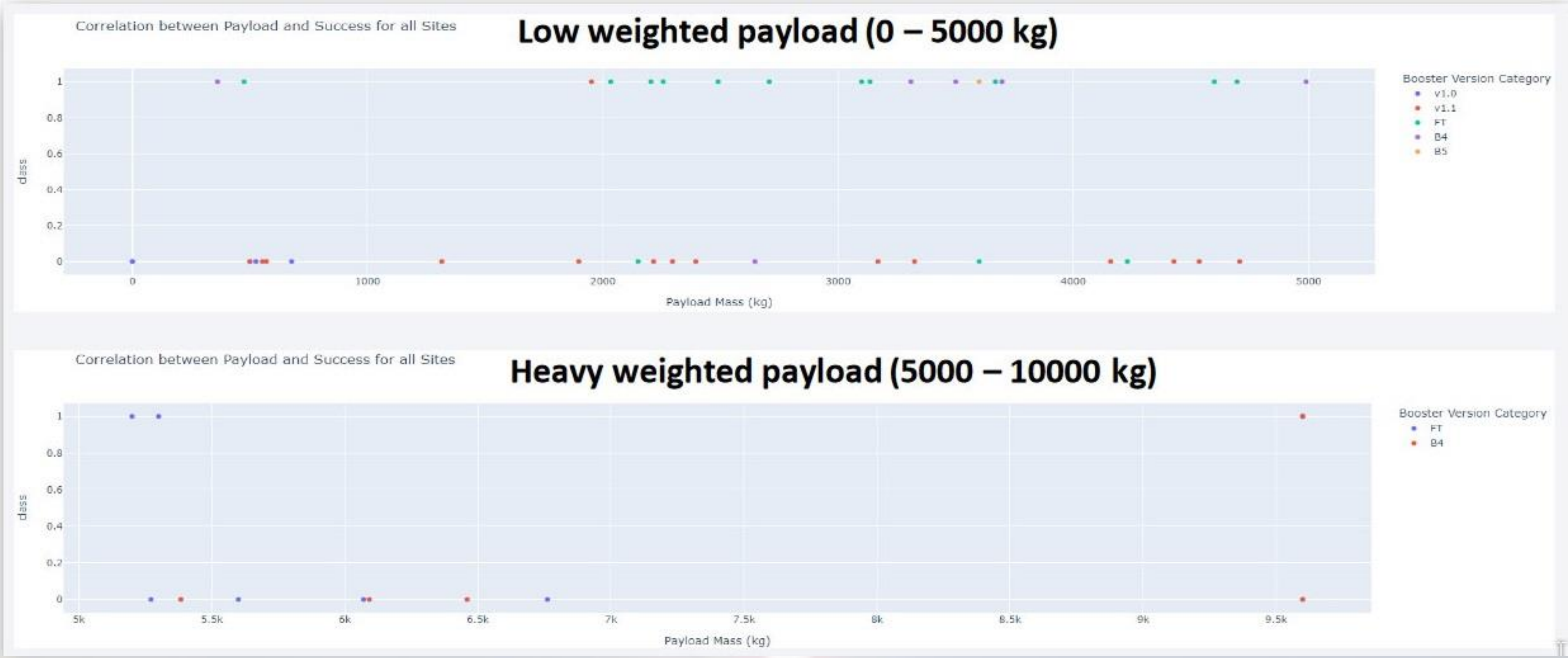
We can see that KSC LC-39A had the most successful launches from all the sites

The highest launch success ratio in the Launch site



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

Payload vs. Launch Outcome



Section 5

Predictive Analysis (Classification)

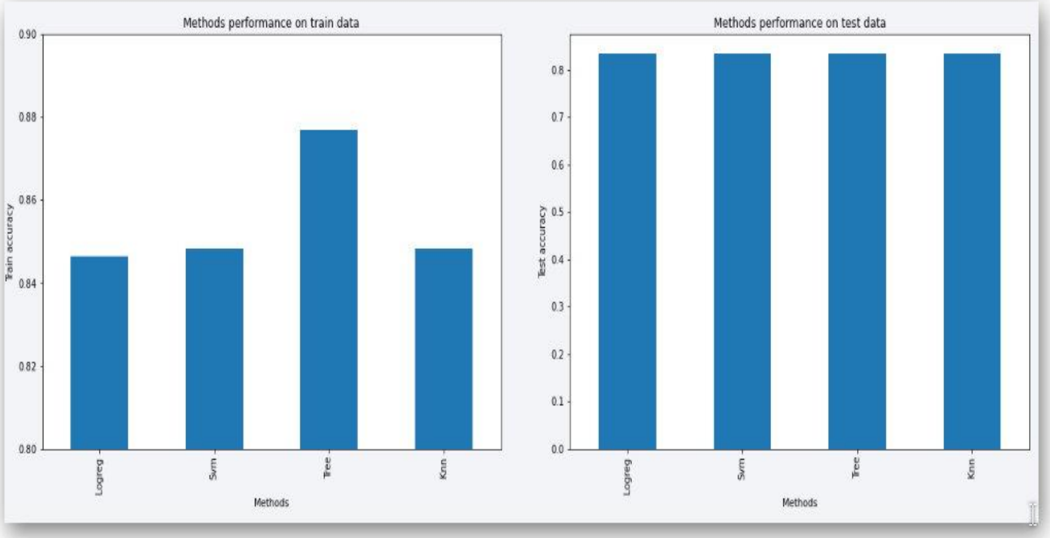
Classification Accuracy

Find the method performs best:

```
In [54]:
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

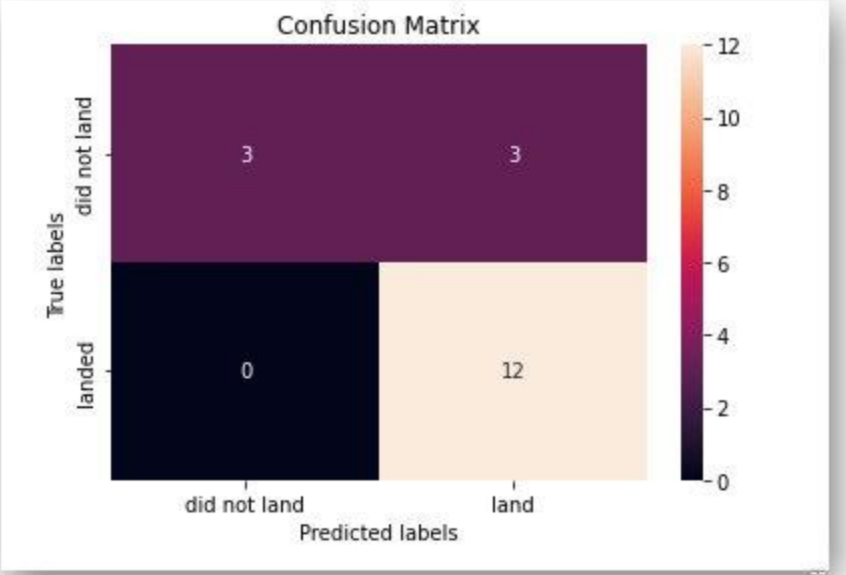
```
Best Algorithm is Tree with a score of 0.9017857142857142
Best Params is : {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'random'}
```

https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/Predictive_Analysis_-_Machine_Learning_Lab.ipynb



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

Confusion Matrix



https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science/blob/main/Predictive_Analysis_-_Machine_Learning_Lab.ipynb

		Actual values	
		1	0
Predicted values	1	TP	FP
	0	FN	TN

The confusion matrices are identical. The main problem of these models are false positives.

Confusion matrix of Decision Tree shows that the accuracy by showing the big numbers of true positive and true negative compared to the false ones.

Conclusions

- The Tree Classifier Algorithm is the best Machine Learning approach for this dataset;
- KSC LC-39A have the most successful launches of any sites;
- Launches above 7,000kg are less risky;
- The best launch site is KSC LC-39A;
- Some orbits require a light or heavy payload mass. But generally low weighted payloads perform better than the heavy weighted payloads;
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- Launch success rate started to increase in 2013 till 2020.

Appendix

GitHub Repository URL:

https://github.com/Morteza-Kabiri/Capstone-Project-Final-IBM-Applied-Data-Science

Special Thanks to All Instructors:

https://www.coursera.org/professional-certificates/ibm-data-science?#instructors

Thank you!