IBM Developer
SKILLS NETWORK

IBM DATA SCIENCE CAPSTONE PROJECT

<Jeampierr Jiménez Chero>
<20/10/2021>

GITHUB URL TO PROJECT

Outline

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

Executive Summary

- Summary of methodologies

Within the methodologies used to carry out my Project, I carried out the following activities:

- Data collection
- Data wrangling
- EDA with data visualization
- EDA with SQL
- Building an interactive map with Folium
- Building a Dashboard with Plotly Dash
- Predictive analysis (Classification)

- Summary of Results

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

Introduction

Figure 2-2: The Falcon Heavy demonstration mission launched from KSC on February 6, 2018

- Project background and context

SpaceX currently has one of the lowest rocket launch costs of any other vendor, worth $ 62 million compared to $ 165 million for its competitors. This figure of 62 million dollars is possible since of the successful landings of the Falcon 9 rockets, the first stage can be taken advantage of.

- Problems you want to find answers

That is why it is essential to be able to determine the success rate of the first stage landings, for this it is necessary to evaluate the various variables that interfere within the model and thus be able to predict the landing success rate.

Section 1

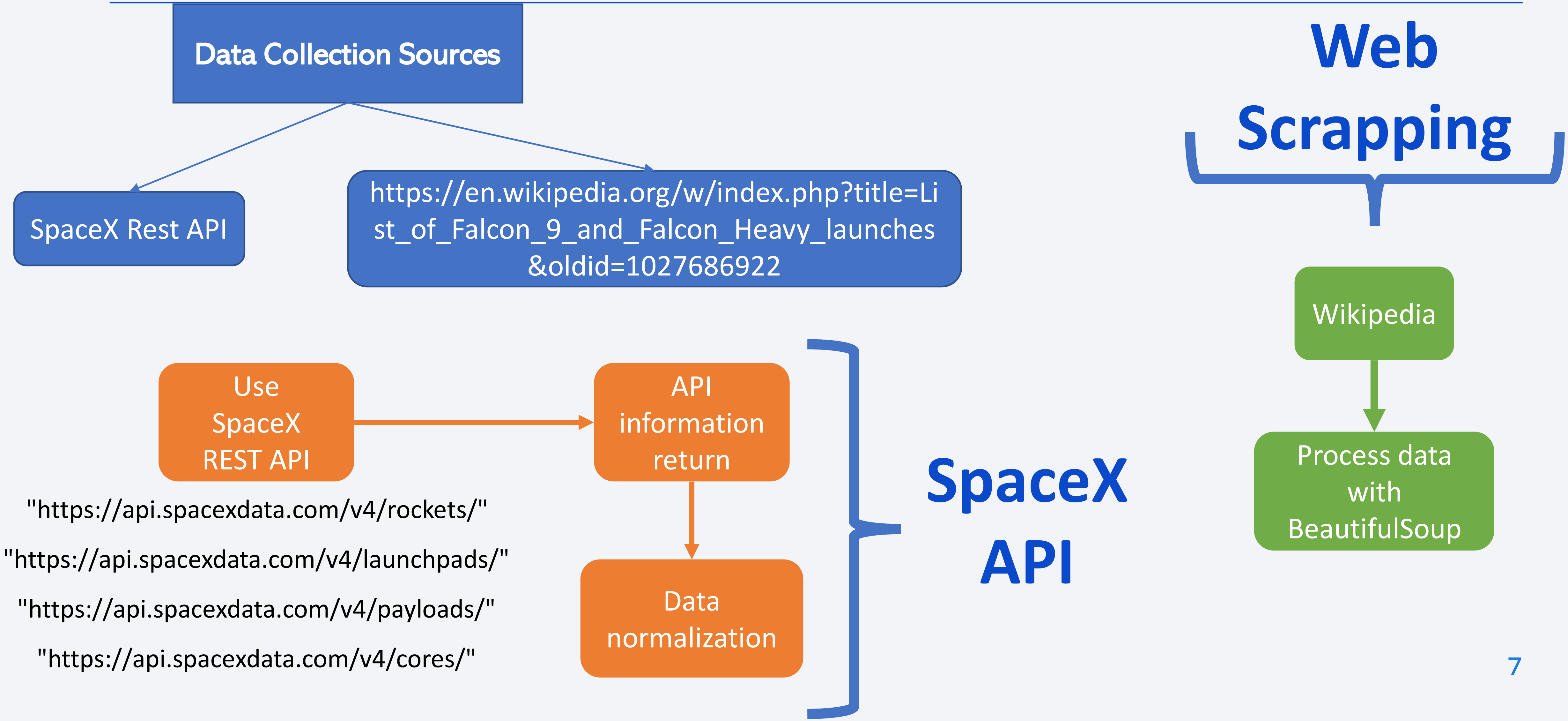
Methodology

Methodology

Executive Summary

- Data collection methodology:
- Perform data wrangling
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection



Data Collection – SpaceX API

Step 1 Capturing the API information

```
In [6]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [7]: response = requests.get(spacex_url)
```

Step 2 Data normalization

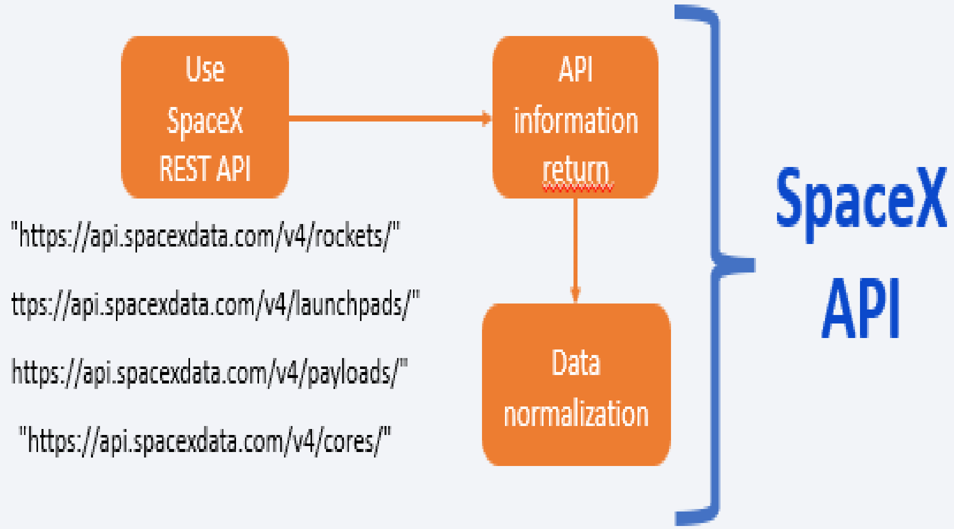
```
In [11]: # Use json_normalize meethod to convert the json result into a dataframe
         df = response.json()
         data=pd.json_normalize(df)
```

Step 3 Obtaining data from stored lists

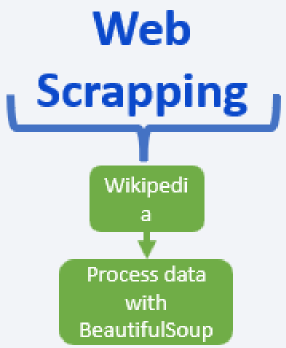
```
In [17]: # Call getBoosterVersion
         getBoosterVersion(data)

In [19]: # Call getLaunchSite
         getLaunchSite(data)

[20]: # Call getPayloadData
      getPayloadData(data)

[21]: # Call getCoreData
      getCoreData(data)
```

Step 4 Convert the file to a CSV format

```
In [32]: data_falcon9.to_csv('dataset_part_1.csv', index=False)
```



[GITHUB URL TO NOTEBOOK]



Data Collection - Scraping

Step 1 Request the Falcon9 Launch Wiki page from its URL

```
In [5]: # use requests.get() method with the provided static_url
        # assign the response to a object

        response = requests.get(static_url).text
        response
```

Step 2 Process and find the table with Beautiful Soup

```
In [6]: # Use BeautifulSoup() to create a Beauti

        soup = BeautifulSoup(response, "lxml")
        soup
```

```
In [8]: # Use the find_all function in the Bea
        # Assign the result to a list called

        html_tables = soup.find_all("table")
        html_tables
```

Step 3 Extract column names

```
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

Step 4 Create the Dictionary

```
In [12]: launch_dict= dict.fromkeys(column_names)

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

         # Let's initial the launch_dict with each
         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']=[]
```

Step 5 Join the data with the keys

```
In [13]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(soup.find_all('tabl
            # get table row
             for rows in table.find_all("tr"):
                 #check to see if first table heading is as numb
                 if rows.th:
                     if rows.th.string:
                         flight_number=rows.th.string.strip()
                         flag=flight_number.isdigit()
```

Step 6 Convert to a DataFrame

```
In [14]: headings = []
         for key,values in dict(launch_dict).items():
             if key not in headings:
                 headings.append(key)
             if values is None:
                 del launch_dict[key]

         def pad_dict_list(dict_list, padel):
             lmax = 0
             for lname in dict_list.keys():
                 lmax = max(lmax, len(dict_list[lname]))
             for lname in dict_list.keys():
                 ll = len(dict_list[lname])
                 if  ll < lmax:
                     dict_list[lname] += [padel] * (lmax - ll)
             return dict_list

         pad_dict_list(launch_dict,0)

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

Step 6 Dataframe to .CSV

```
In [15]: df.to_csv('spacex_web_scraped.csv', index=False)
```

GITHUB URL TO NOTEBOOK

Data Wrangling

1. First, the launch amount for each site was calculated, applying a count of values to the "LaunchSite" column.

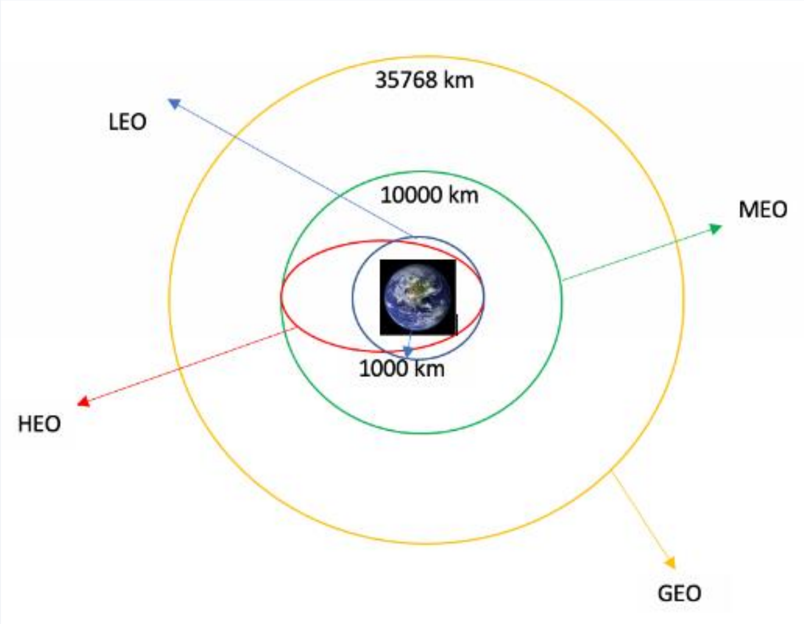
2. The number of launches that occurred per orbit was also calculated by taking the "Orbit" column.

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

4. Assign values of 0 to failed landings and 1 to successful landings and then label it as a new column called "Class", obtaining with it an average success rate of 66%,

5. Save the DataFrame in .CSV format.

Analysis Flow

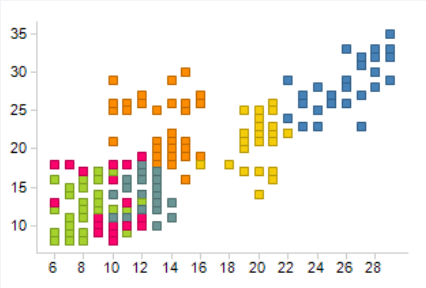


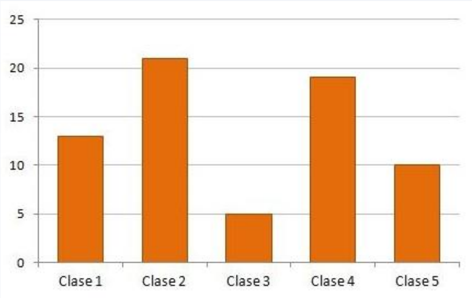
GITHUB URL TO NOTEBOOK

EDA with Data Visualization

Scatter graphs, bar graphs and a trend graph were used for our analysis. These graphs will allow us to explain the relationships between the chosen variables. The graphs used are detailed below:

Comparisons	Chart Type
FlightNumber vs. PayloadMass	Scatter Graphs
FlightNumber vs LaunchSite	Scatter Graphs
PayloadMass vs LaunchSite	Scatter Graphs
FlightNumber vs Orbit	Scatter Graphs
PayloadMass vs. Orbit	Scatter Graphs
Success Rate vs Orbit	Bar Graphs
Success Rate VS. Year	Line Graph





[GITHUB URL TO NOTEBOOK]

EDA with SQL

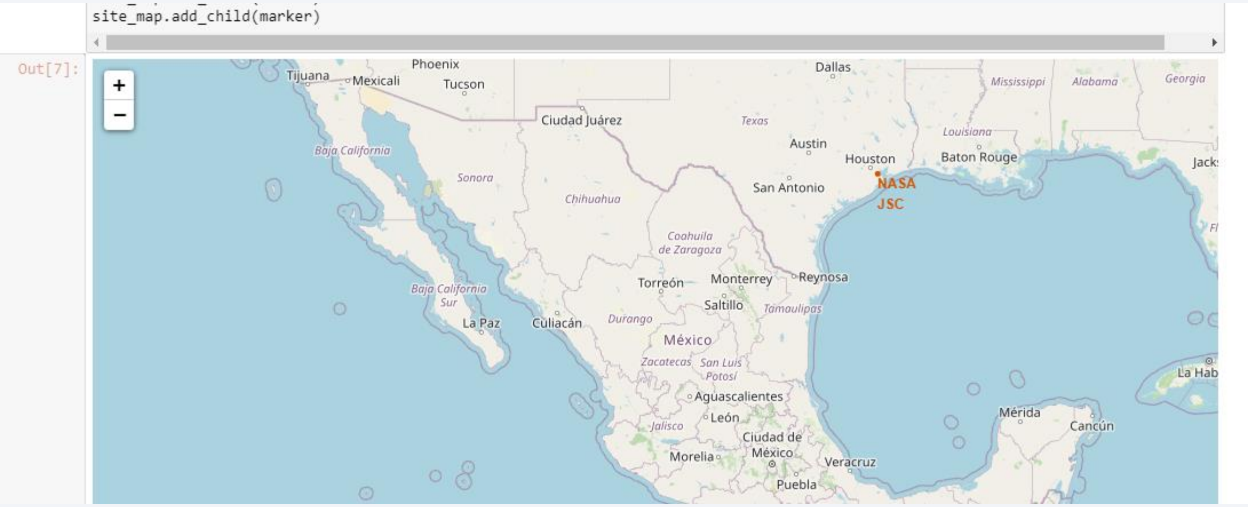
Various queries were made to my database in the IBM DB2 on Cloud cloud, among which we can mention the following:

N°	Queries
1	Display the names of the unique launch sites in the space mission
2	Display 5 records where launch sites begin with the string 'CCA'
3	Display the total payload mass carried by boosters launched by NASA (CRS)
4	Display average payload mass carried by booster version F9 v1.1
5	List the date when the first successful landing outcome in ground pad was achieved.
6	List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7	List the total number of successful and failure mission outcomes
8	List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
9	List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
10	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 URL TO NOTEBOOK

Build an Interactive Map with Folium

- To visualize the launch sites within a map, the latitudes and longitudes of the coordinates were used by adding a marker with the name of the corresponding site. The Folium package was used to carry out this activity.
- Also added red markers for unsuccessful launches (0) and green for successful launches (1)
- A script was also used to calculate the distances by means of the Haversine formula. A script was also used to calculate the distances by means of the Haversine formula.



[GITHUB URL TO NOTEBOOK]

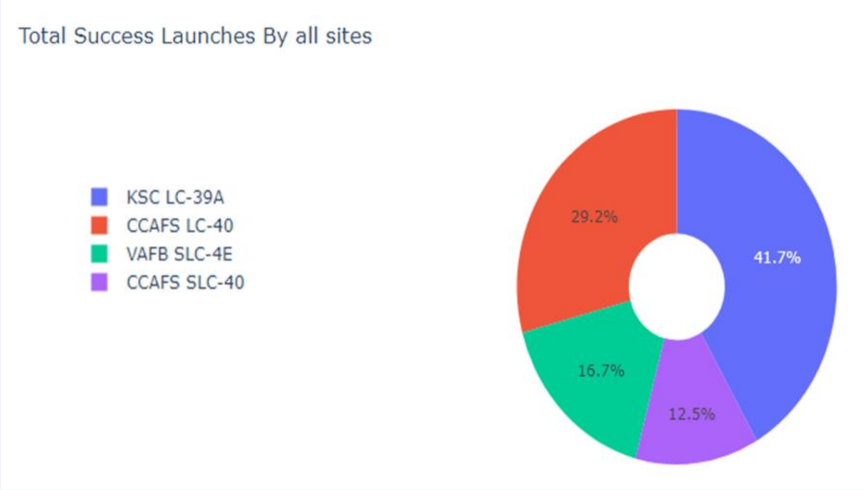
Build a Dashboard with Plotly Dash

- It was made with the help of the Dash and Plotly Packages, an interactive dashboard hosted in the application's server, in which you can view a donut chart in which you can make a selection to verify the success rates in the landings for each launch site.

- A bubble diagram was also created between the variables "Class" and "PayloadMass" (Kg), for each of the versions of the F9 Booster, for this a slider was implemented within the graph in which we can adjust the ranges of kilograms and see which booster versions fall into it.

- Through these interactive graphs we will be able to notice some relationships that exist between the weights, as well as the success rate at each launch site.

[GITHUB URL TO NOTEBOOK]



Predictive Analysis (Classification)

GITHUB URL TO NOTEBOOK

Data acquisition

The data was obtained from previous works with the dataset_part_2.csv and dataset_part_3.csv files

Data processing

The dataset_part_2.csv and dataset_part_3.csv files are a files that was processed in previous jobs, so this activity is discarded since it is a clean data.

Extraction of variables

X : dataset_part_3.csv
Y : Y = data['Class'].to_numpy() from dataset_part_2.csv

Machine Learning Algorithm

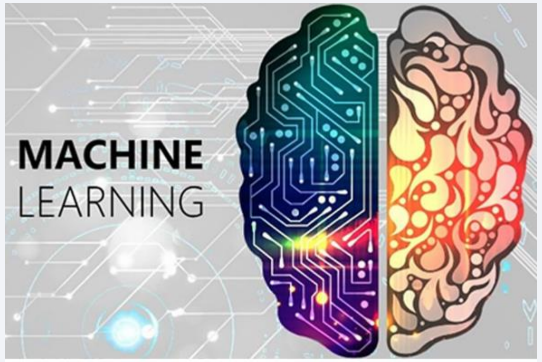
Models:	log_reg	svm	tree	knn

Obtaining the model

Split our data into training and test data sets, Set our parameters and algorithms to GridSearchCV, Fit our datasets into the GridSearchCV objects and train our dataset

Evaluation of results

Check accuracy for each model and plot Confusion Matrix



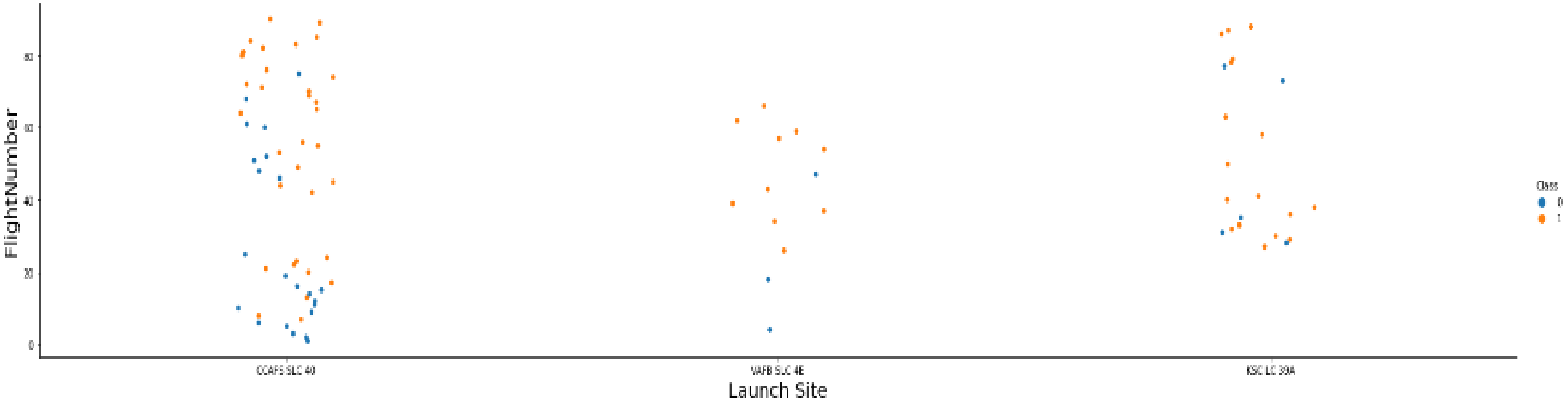
Results

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

Section 2

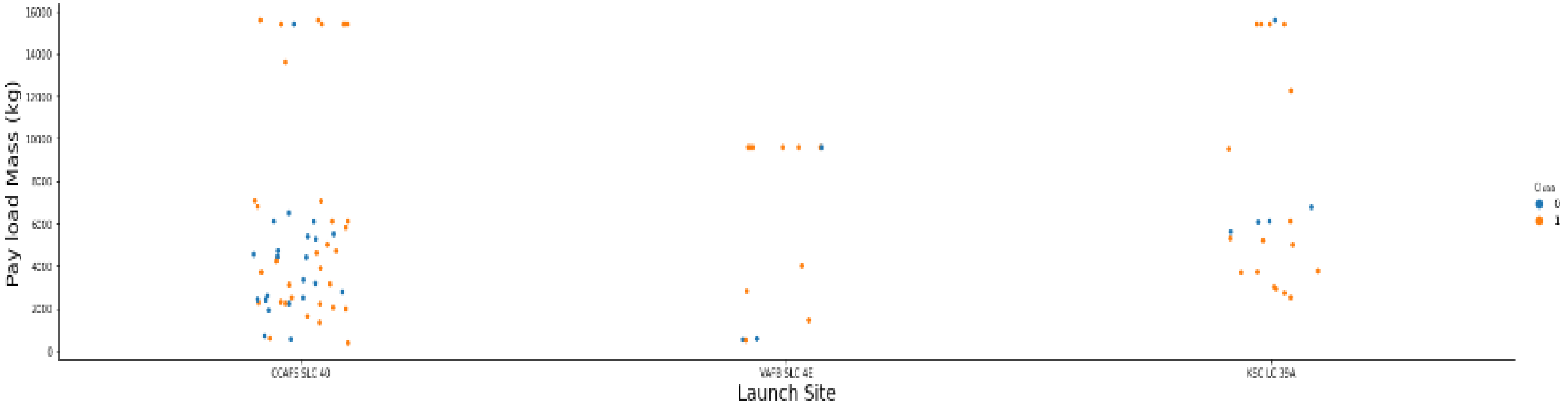
Insights drawn from EDA

Flight Number vs. Launch Site



The most flights with the highest success rate occur at the CCAFS SLC 40 launch site.

Payload vs. Launch Site

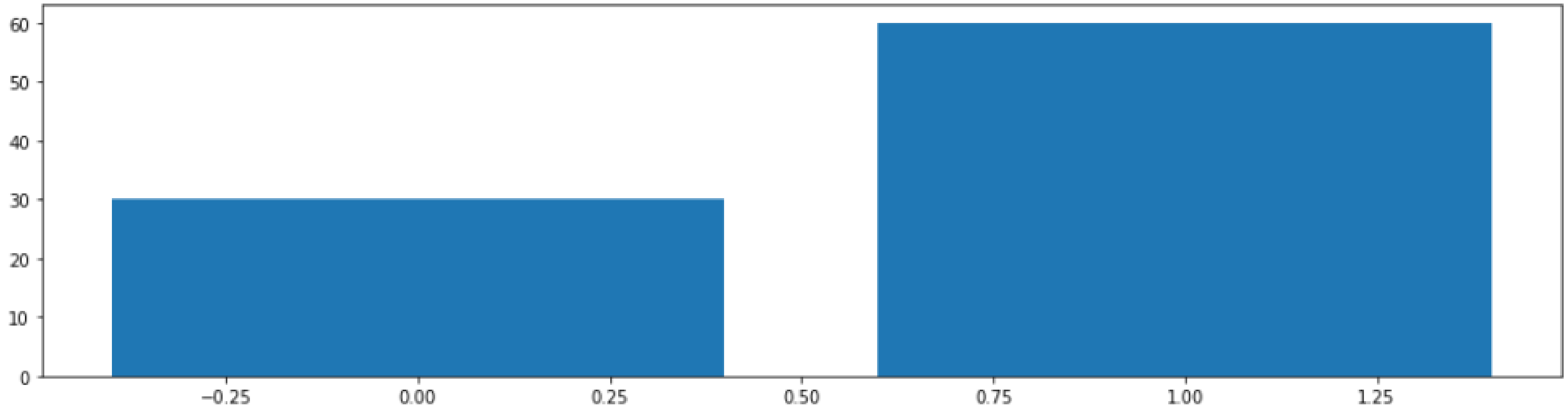


- In the graph we can see that the largest amount of mass is found at the CCAFS SLC 40 launch site, which in turn coincides with the highest success rate.

Overall Success Rate

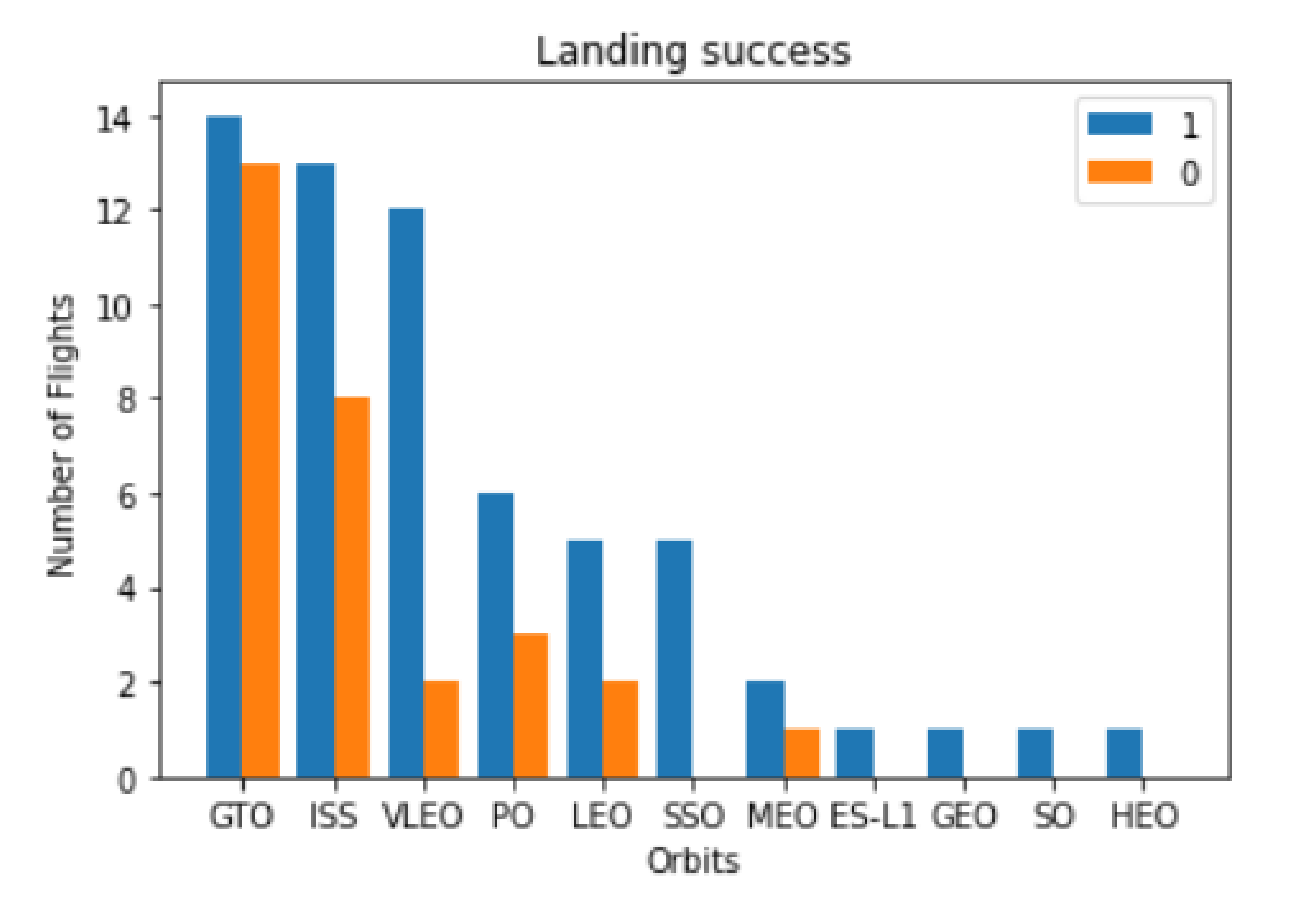
```
ALL ORBIT
Class=0, n=30 (33.333333333333336%)
Class=1, n=60 (66.66666666666666%)
```

Out[11]: <BarContainer object of 2 artists>



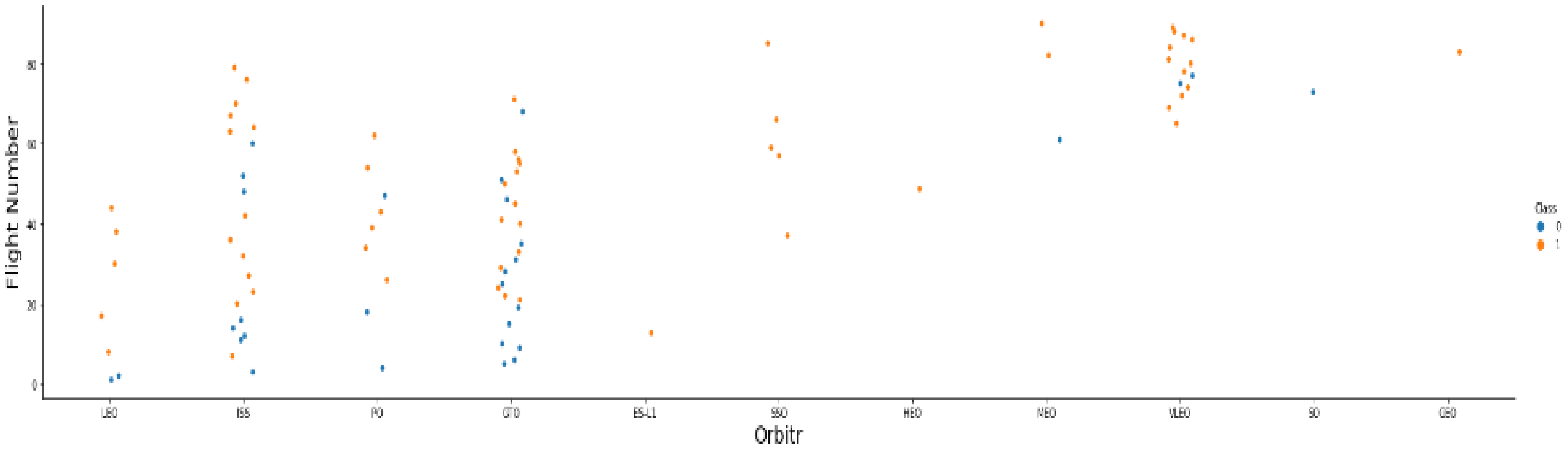
- In this graph we see that among all the orbits you have a success rate of 67%, with 60 flights being counted

Success Rate vs. Orbit Type



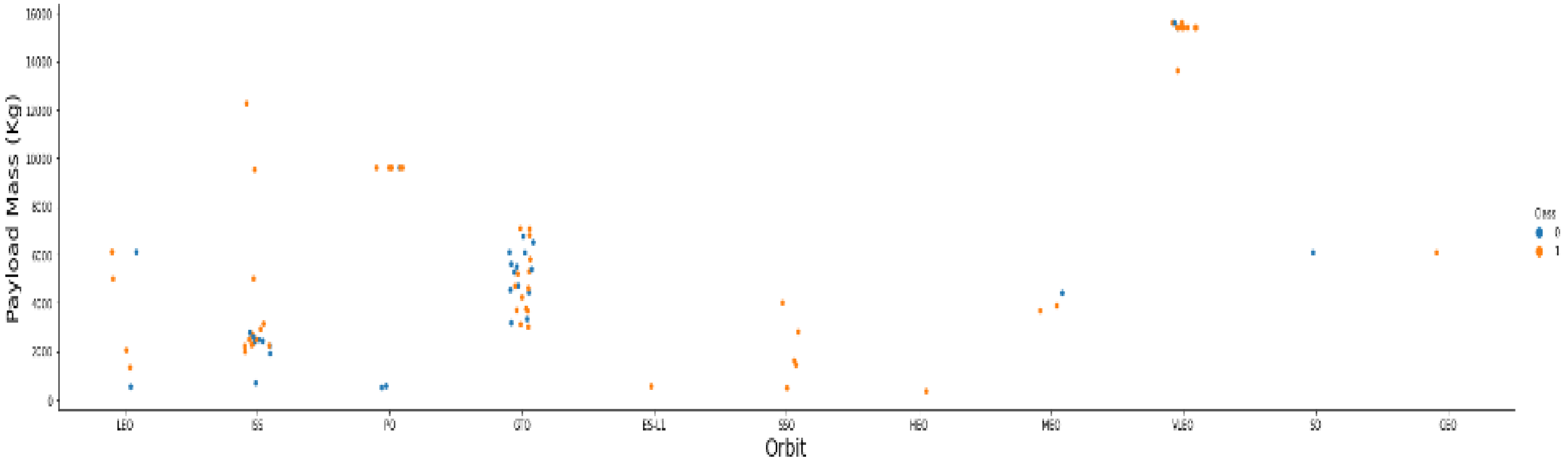
In the bar graph we can see that the GTO orbit presented the highest number of flights, but its success rate is at the value of 51.8%, almost balanced with its failure rate, this gives us an idea that the greater the number of flights. flights there is a probability of equating the success rate with the failure rate, if the launches were made in the other orbits.

Flight Number vs. Orbit Type



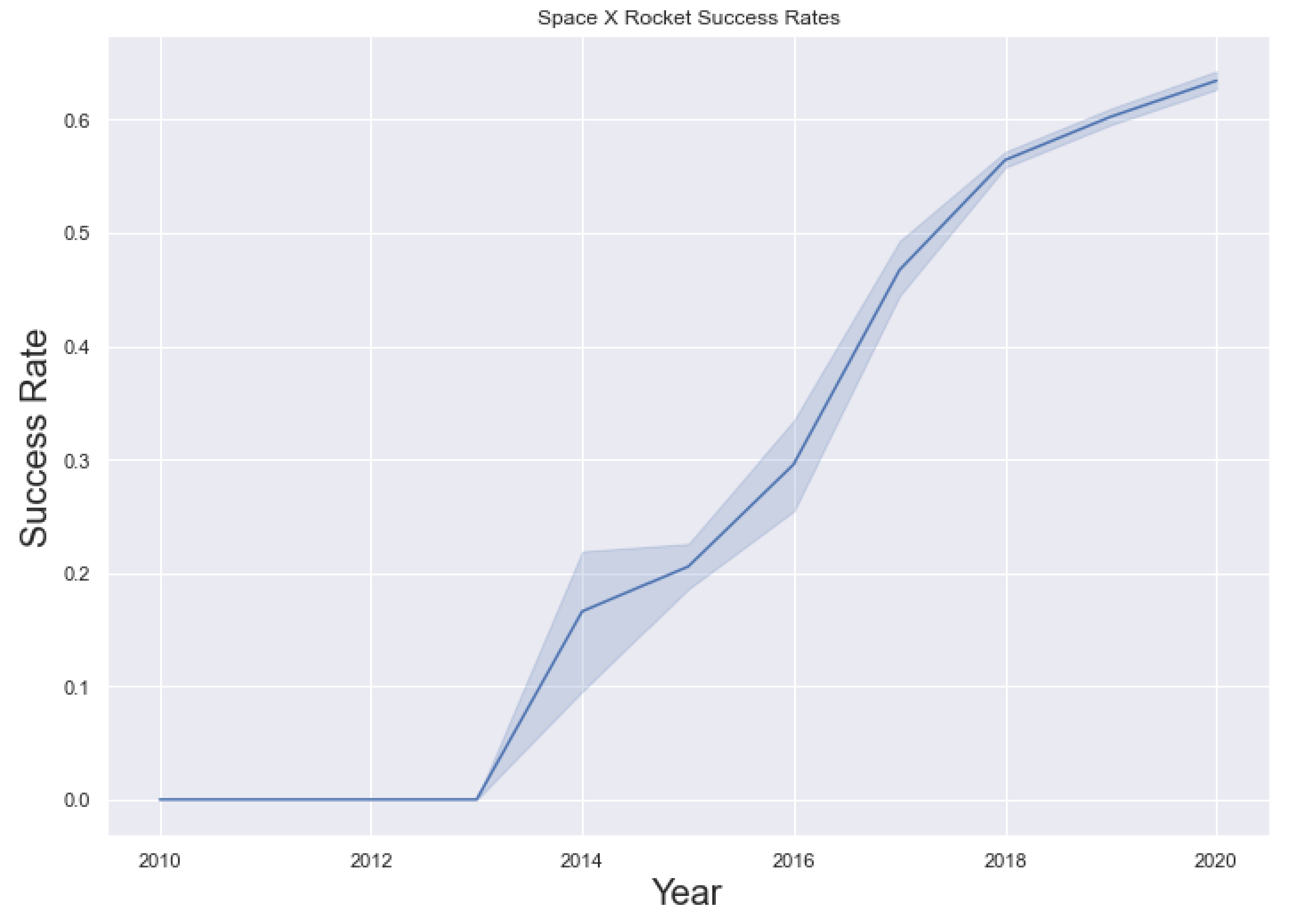
- In the graph we can see that the success rate for the Leo orbit increases as the number of flights increases, but if we transfer this assumption to the GTO orbit, this relationship is not reflected since we can notice that there is an apparent equilibrium between the rate of success and failure rate.

Payload vs. Orbit Type



- In the graph we can see that the VLEO orbit, as the mass increases, will present a higher success rate than the other orbits, but this is not a relationship that we can replicate in the other orbits.

Launch Success Yearly Trend



- In the graph we can see that since 2013 the launch success rate has been increasing, reaching its peak in 2020.

All Launch Site Names

```
In [4]: %sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXBTL

 * ibm_db_sa://btq76615:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.

Out[4]:
```

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

- The “distinct” statement was used on the "launch_sit" column of the SPACEXBTL table of our database

Launch Site Names Begin with 'CCA'

```
In [5]: %sql select*FROM SPACEXBTL where LAUNCH_SITE like 'CCA%' fetch first 5 rows only;

 * ibm_db_sa://btq76615:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

Out[5]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- We select the first 5 records by filtering the values of the "launch_site" column that start with the letters "CCA" in the SPACEXBTL table.

Total Payload Mass

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

```
In [6]: %sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXBTL WHERE CUSTOMER = 'NASA (CRS)';

 * ibm_db_sa://btq76615:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.

Out[6]:
```

1
45596

- The sum function was applied to total the mass found in the column "PAYLOAD_MASS__KG_) only for the client" NASA (CRS) "of the SPACEXBTL table, whose value was 45596 kg.

Average Payload Mass by F9 v1.1

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

```
In [7]: %sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXBTL WHERE BOOSTER_VERSION = 'F9 v1.1';

 * ibm_db_sa://btq76615:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.

Out[7]:
```

1
2928.400000

- The average function was applied to average the mass found in the column "PAYLOAD_MASS__KG_) only for the BOOSTER_VERSION column that has the values of 'F9 v1.1' from the SPACEXBTL table whose value obtained was 2928.4 kg

First Successful Ground Landing Date

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

Hint:Use min function

In [8]:
```
%sql SELECT MIN(DATE) FROM SPACEXBTL WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://btq76615:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

Out[8]:

1
2015-12-22

- The minimum function was applied on the column "DATE" that has as a value in the column "LANDING__OUTCOME" the word "Succes (ground pad)" of the table SPACEXBTL whose value obtained was 2015 - 12 - 22.

Successful Drone Ship Landing with Payload between 4000 and 6000

%sql SELECT BOOSTER_VERSION FROM SPACEXBTL WHERE LANDING__OUTCOME = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000;

```
In [9]: VERSION FROM SPACEXBTL WHERE LANDING__OUTCOME = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000;

 * ibm_db_sa://btq76615:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

Out[9]:

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

- Those values in the "LANDING__OUTCOME" column with the word "Success (drone ship)" and whose weight in the "PAYLOAD_MASS__KG_" window was between 4000 kg and 6000 kg in the SPACEXBTL table were filtered.

Total Number of Successful and Failure Mission Outcomes

%sql SELECT COUNT(MISSION_OUTCOME) FROM SPACEXBTL WHERE MISSION_OUTCOME = 'Success' or MISSION_OUTCOME = 'Success (payload status unclear)' or MISSION_OUTCOME = 'Failure (in flight)';

```
In [11]: %sql SELECT COUNT(MISSION_OUTCOME) FROM SPACEXBTL WHERE MISSION_OUTCOME = 'Success' or MISSION_OUTCOME = 'Success (payload status

 * ibm_db_sa://btq76615:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.

Out[11]:   1
         101
```

- A count was made of the missions that failed and those that were successful, filtering the "MISSION_OUTCOME" column in which these 2 conditions must be met, resulting in 101 flights.

Boosters Carried Maximum Payload

```
In [12]: %sql SELECT BOOSTER_VERSION FROM SPACEXBTL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) from SPACEXBTL);

 * ibm_db_sa://btq76615:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

Out[12]:

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

- A selection was made of the Boosters that had the maximum weight in the column "PAYLOAD_MASS_KG_" of the SPACEXBTL table.

2015 Launch Records

%sql select LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE, DATE from SPACEXBTL where LANDING__OUTCOME = 'Failure (drone ship)' and YEAR(DATE) = 2015 ;

```
In [14]: %sql select LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE, DATE from SPACEXBTL where LANDING__OUTCOME = 'Failure (drone ship)' a

 * ibm_db_sa://btq76615:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.

Out[14]:
```

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

- A list is made with the landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015 obtaining 2 records for this year 2015

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

%sql SELECT * FROM SPACEXBTL WHERE (LANDING__OUTCOME = 'Succes (ground pad)' or LANDING__OUTCOME = 'Failure (drone ship)') and (DATE BETWEEN '2010-06-04' and '2017-03-20') ORDER BY DATE DESC

```
In [15]: %sql SELECT * FROM SPACEXBTL WHERE (LANDING__OUTCOME = 'Succes (ground pad)' or LANDING__OUTCOME = 'Failure (drone ship)') and (D

 * ibm_db_sa://btq76615:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

Out[15]:

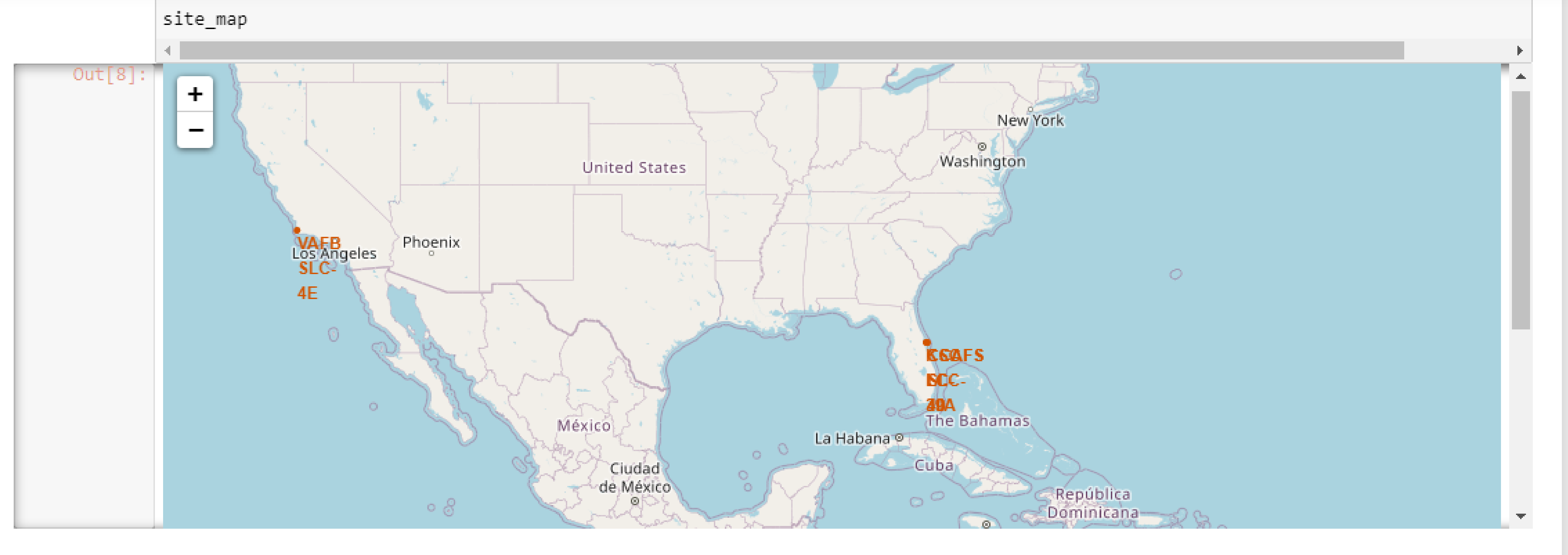
DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2016-06-15	14:29:00	F9 FT B1024	CCAFS LC-40	ABS-2A Eutelsat 117 West B	3600	GTO	ABS Eutelsat	Success	Failure (drone ship)
2016-03-04	23:35:00	F9 FT B1020	CCAFS LC-40	SES-9	5271	GTO	SES	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)
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)
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)

- A count was made on the conditions in the column LANDING__OUTCOME ('Succes (ground pad)' or 'Failure (drone ship)') and in the column "DATE" ('2010-06-04' and '2017-03-20 ') in descending order, obtaining 5 records.

Section 4

Launch Sites Proximities Analysis

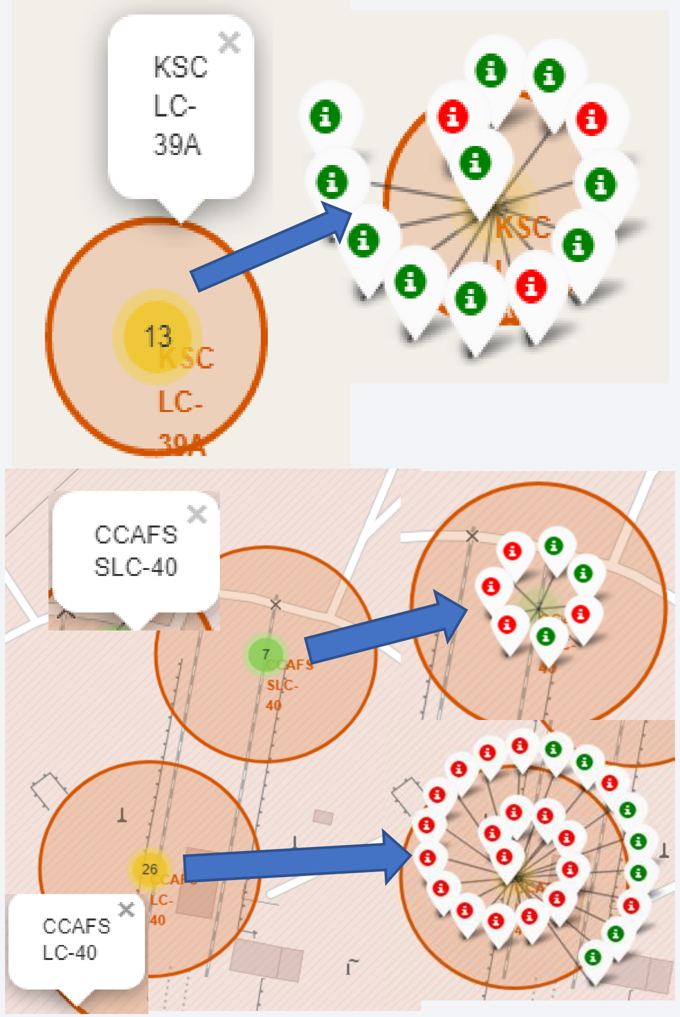
Map of the launches made by SPACEX

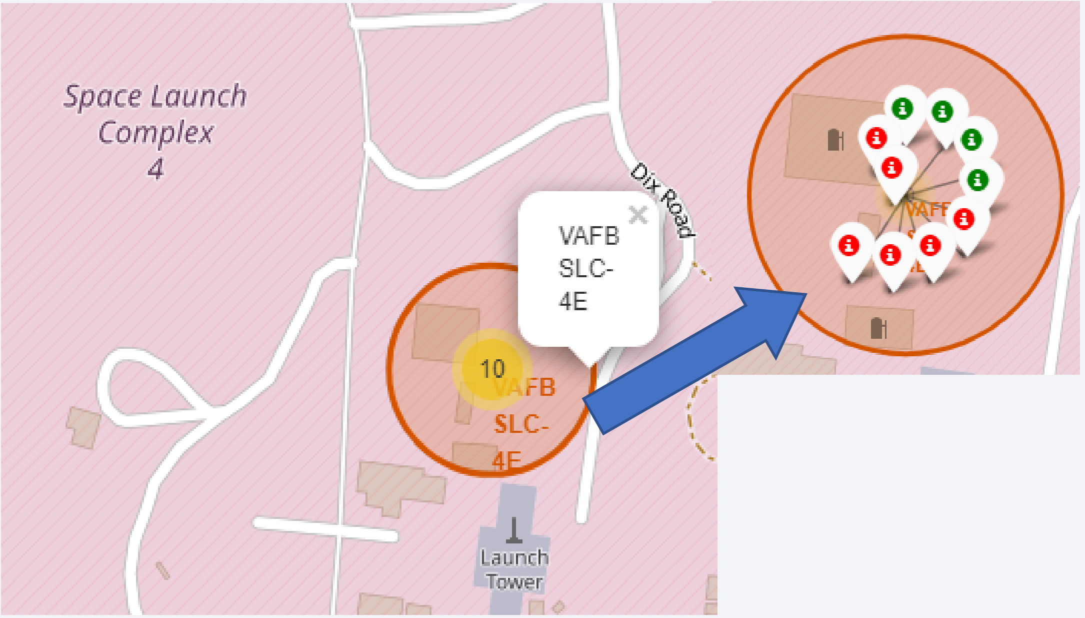


It can be noted that the launches are made from the coasts of the United States of California and Florida

Success rates by launch site

Florida Launch Sites



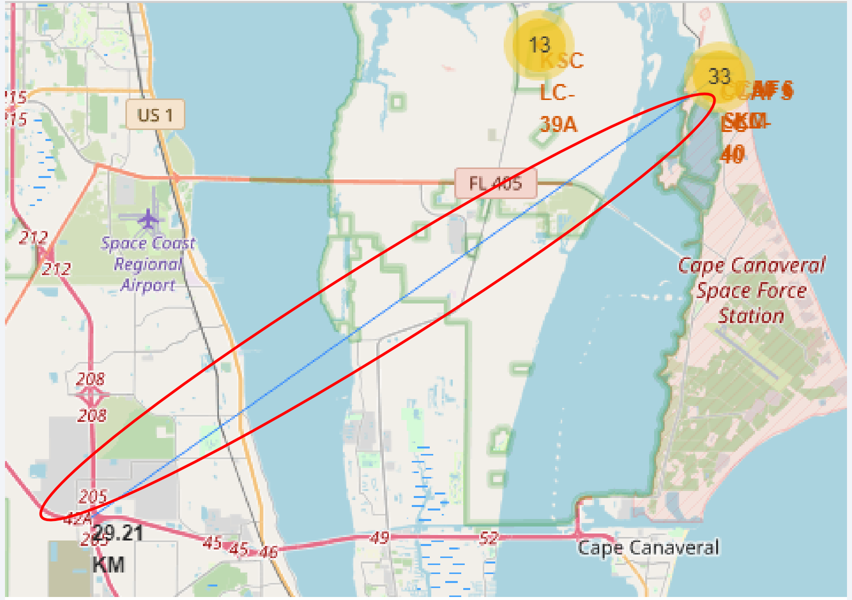


California Launch Site

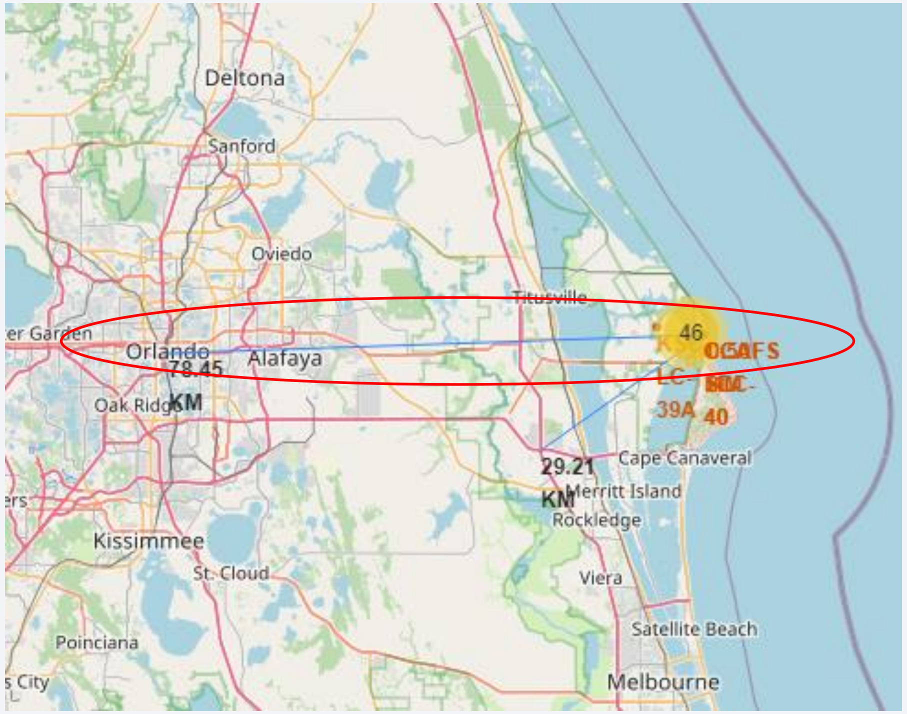
success

No succes

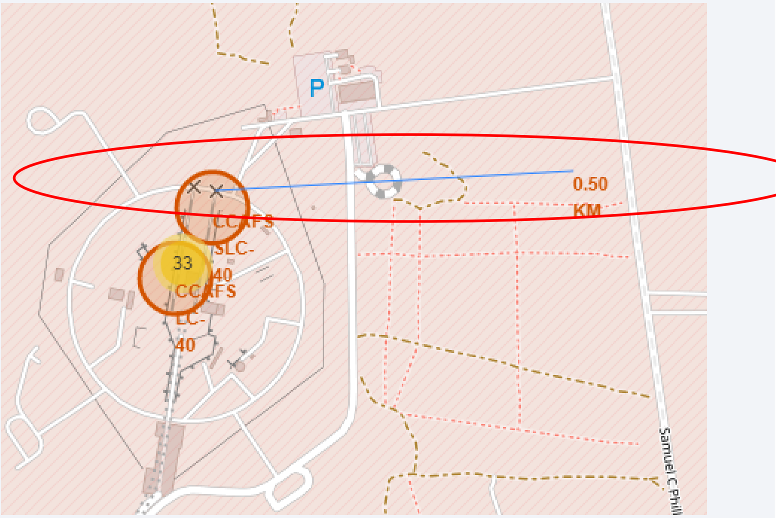
Distances found



Distance to the road 29.21 km



Distance to the Orlando City 78.45 km

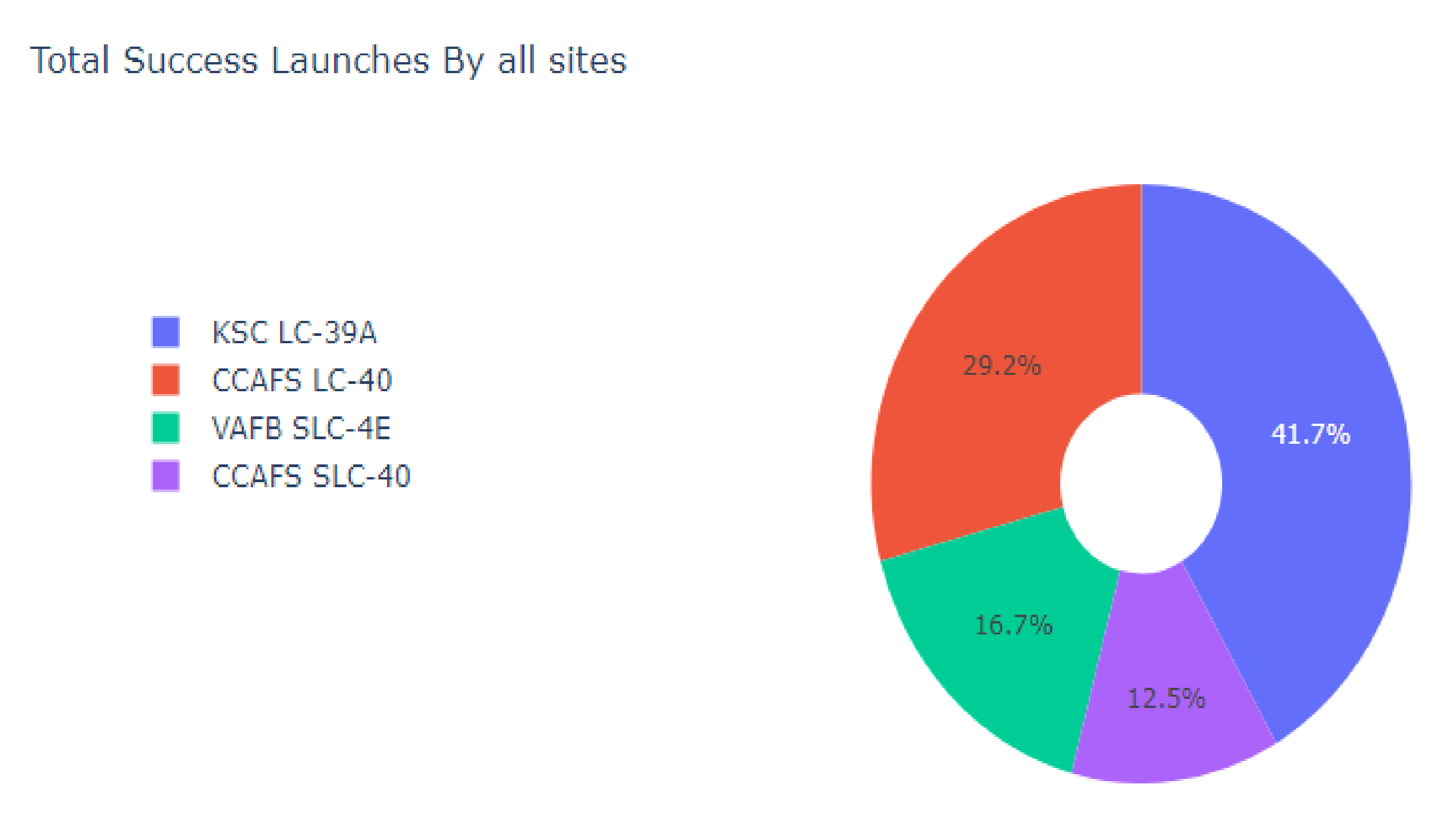


Distance to the coastline 0.50 km

Section 5

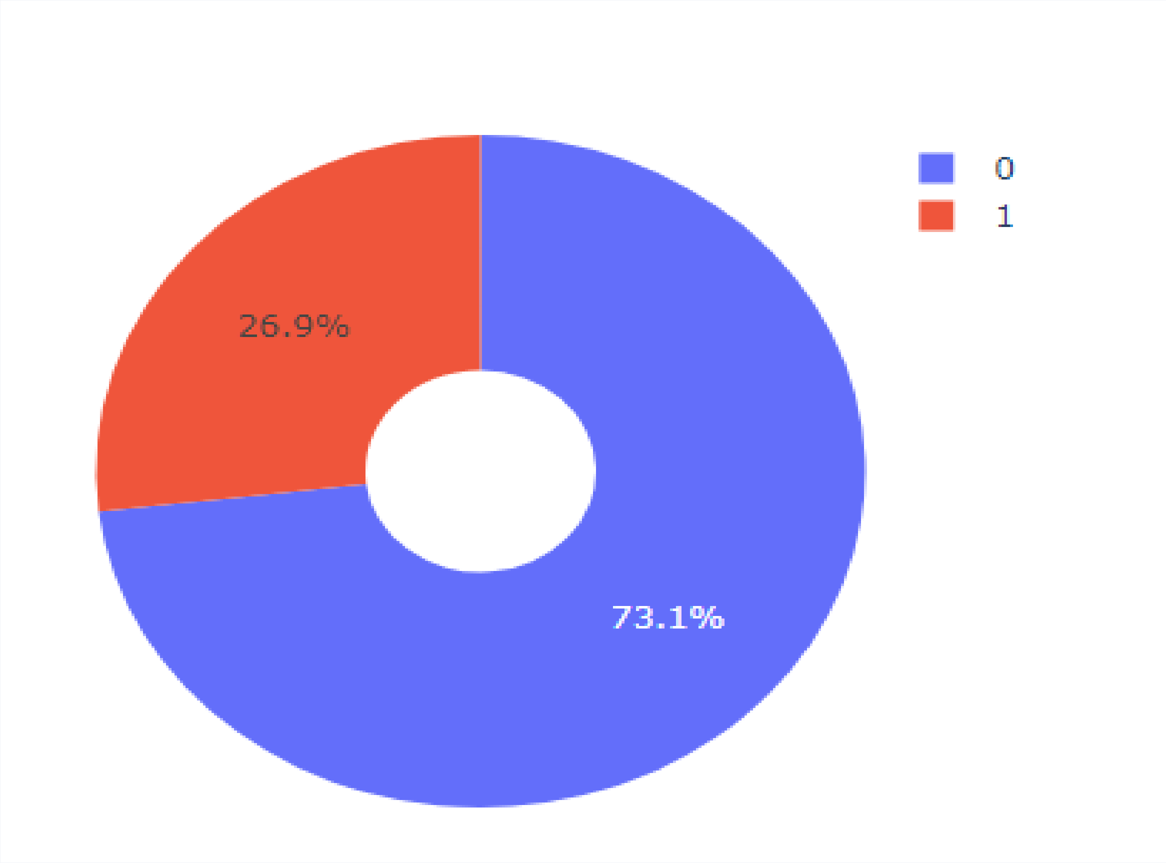
Build a Dashboard with Plotly Dash

SpaceX Launch Records Dashboard



From the graph we can see that the site with the highest success rate is KSC LC-39A with 41.7%, while the site with the lowest success rate is CCAFS SLC-40 with 12.5%.

Total Success Launches for site CCAFS LC-40



From the graph we can see that the success rate of CCAFS SLC-40 is 73.1%.

Payload vs Launch Outcome

- From the graph we can see that there is a higher success rate for those Payload_Mass with the lower weights

Section 6

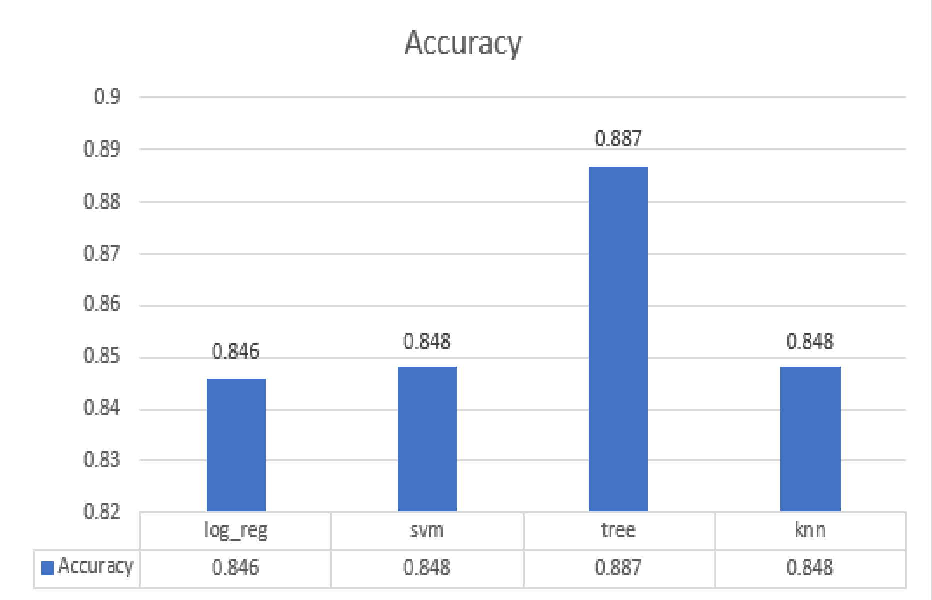
Predictive Analysis (Classification)

Classification Accuracy

```
In [12]: print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
         print("accuracy :",logreg_cv.best_score_)

tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

```
In [16]: print("tuned hpyerparameters :(best parameters) ",svm_cv.best_params_)
         print("accuracy :",svm_cv.best_score_)

tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel':
accuracy : 0.8482142857142856
```

```
In [24]: print("tuned hpyerparameters :(best parameters) ",knn_cv.best_params_)
         print("accuracy :",knn_cv.best_score_)

tuned hpyerparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
accuracy : 0.8482142857142858
```




```
In [20]: print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
         print("accuracy :",tree_cv.best_score_)

tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 12, 'max_features': 'auto', 'min_samples_leaf':
1, 'min_samples_split': 5, 'splitter': 'best'}
accuracy : 0.8875
```

From the bar graph we can see that the model that gives us the best accuracy is the model worked with the decision tree algorithm whose accuracy is 88.7% whose parameters are:

- Best Params is : {'criterion': 'entropy', 'max_depth': 12, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 5, 'splitter': 'best'}

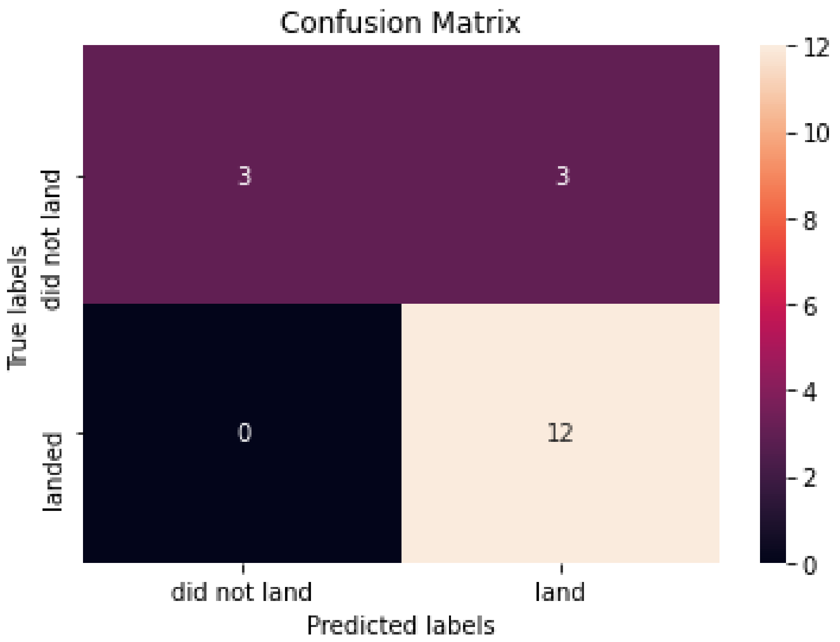
Confusion Matrix

```
In [21]: print("accuracy: ",tree_cv.score(X_test,Y_test))
```

```
accuracy:  0.7222222222222222
```

We can plot the confusion matrix

```
In [22]: yhat = svm_cv.predict(X_test)
         plot_confusion_matrix(Y_test,yhat)
```



From the matrix we can see that the model that gives us the best accuracy is the model worked with the decision tree algorithm whose accuracy is 88.7% whose parameters are:

- Best Params is : {'criterion': 'entropy', 'max_depth': 12, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 5, 'splitter': 'best'}

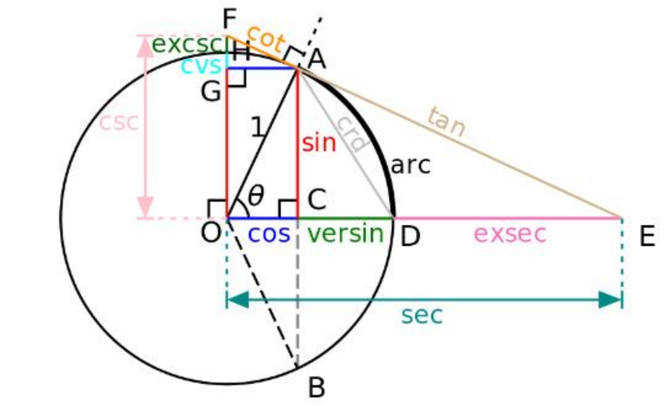
Our model predicted 15 successful flights out of 18, when in reality it was 12 successful flights out of 18, thus having 3 false positive flights

Conclusions

- The best model to predict the success rate is the tree algorithm.
- Given the trend that began in 2013, the success rate can be expected to continue to improve in the following years
- The best success rate given the mass of the Payload_Mass, is found in the launches to the ORBITA Leo, but up to a weight that does not exceed 6000 kg

Figure 8-6: Integrated Falcon 9 on the transporter-erector within the integration hangar and rolling out

Haversine's formula



SOURCE https://community.esri.com/t5/coordinate-reference-systems-blog/distance-on-a-sphere-the-haversine-formula/ba-p/902128

Haversine Formula

The Haversine formula is perhaps the first equation to consider when understanding how to calculate distances on a sphere. The word "Haversine" comes from the function:

> $\text{haversine}(\theta) = \sin^2(\theta/2)$

The following equation where ϕ is latitude, λ is longitude, R is earth's radius (mean radius = 6,371km) is how we translate the above formula to include latitude and longitude coordinates. Note that angles need to be in radians to pass to trig functions:

> $a = \sin^2(\phi B - \phi A/2) + \cos \phi A * \cos \phi B * \sin^2(\lambda B - \lambda A/2)$
>
> $c = 2 * \text{atan2}(\sqrt{a}, \sqrt{(1-a)})$
>
> $d = R \cdot c$

Thank you!