IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Shefali Uttam
10th Jan, 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection, cleaning and wrangling
  - Exploratory data analysis with data visualization, SQL, Folium maps and Plotly
  - Machine learning classification to predict if the first stage will land
- Summary of all results
  - Exploratory data analysis results
  - Interactive maps and dashboards
  - Predictive analysis

Introduction

- SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars. Whereas other providers cost upward of 165 million dollars for each launch. Much of the savings for SpaceX rocket launches is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch.

- Since SpaceY wants to compete with SpaceX, and hence we need to determine the price of each launch. This will be achieved by gathering information about all the landings of the Falcon 9 rocket's first stage.

- We will also determine if SpaceX will reuse the first stage, which will help them to provide a less cost for rocket launch.

Section 1

Methodology

Methodology

Executive Summary

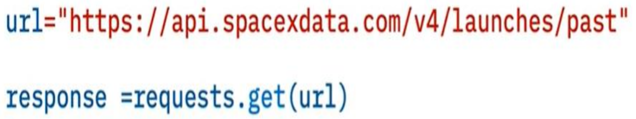
- Data collection methodology:
  - SpaceX REST API and Wikipedia pages.
- Perform data wrangling
  - Some columns like “LaunchSite”, “Orbit”, and “Outcome” are used to form a classification variable Y to predict if the launch was successful or not.
  - Dealt with missing data in the dataset and replaced them with the means
- 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 Collection

- The SpaceX launch data is gathered from the SpaceX REST API. This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
- These data are available on various SpaceX REST API endpoints, which starts with api.spacexdata.com/v4/. We use the different end points: /rockets, /launchpads, /payloads, /cores and /past.
- The Falcon 9 Launch data is also obtained by web scraping related Wiki pages. We have used the Python BeautifulSoup package to web scrape some HTML tables that contain Falcon 9 launch records.
- We have parsed the data from those tables and converted them into a Pandas data frame for visualization and analysis.

Data Collection – SpaceX API

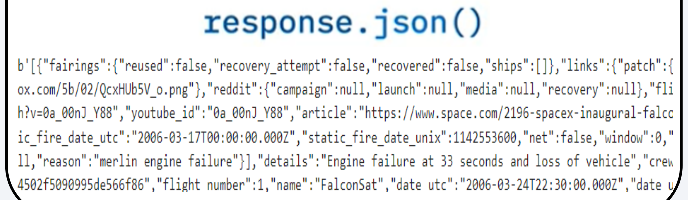
Obtain data from the API



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

response =requests.get(url)
```

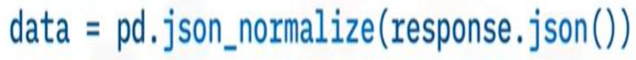
Response is in JSON format



```
response.json()
```

```
b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{
ox.com/5b/02/QcxHUb5V_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"fli
h?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.space.com/2196-spacex-inaugural-falcc
ic_fire_date_utc":"2006-03-17T00:00:00.000Z","static_fire_date_unix":1142553600,"net":false,"window":0,"
ll,"reason":"merlin engine failure"}],"details":"Engine failure at 33 seconds and loss of vehicle","crew
4502f5090995de566f86","flight number":1,"name":"FalconSat","date utc":"2006-03-24T22:30:00.000Z","date u
```

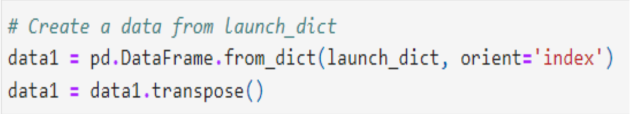
Convert JSON into data frame



```
data = pd.json_normalize(response.json())
```

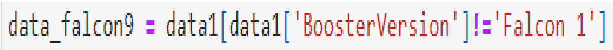
Use the API again to get information about the launches using the IDs given for each launch

Create a Pandas data frame from the dictionary launch_dict



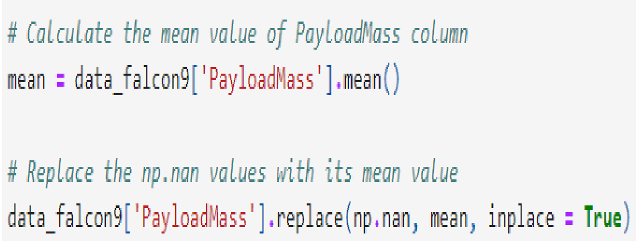
```
# Create a data from launch_dict
data1 = pd.DataFrame.from_dict(launch_dict, orient='index')
data1 = data1.transpose()
```

Filter the data frame to only include Falcon9 launches



```
data_falcon9 = data1[data1['BoosterVersion']!='Falcon 1']
```

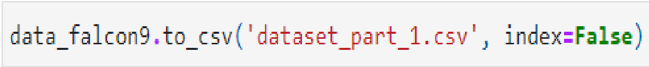
Dealing with missing values in the data frame



```
# Calculate the mean value of PayloadMass column
mean = data_falcon9['PayloadMass'].mean()

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

Exporting data in to a CSV file

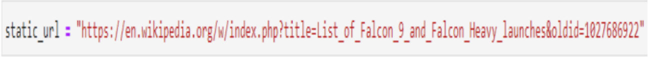


```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

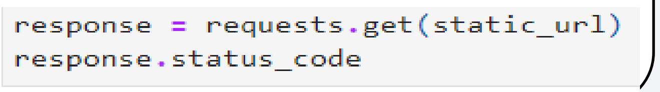
Link to URL

Data Collection - Scraping

Request the Falcon 9 launch Wiki page

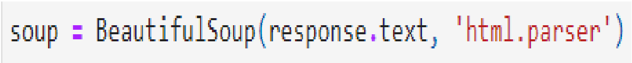


```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```



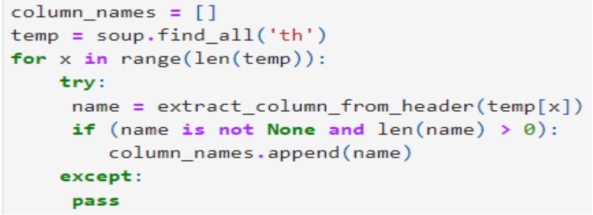
```
response = requests.get(static_url)
response.status_code
```

Create a BeautifulSoup object



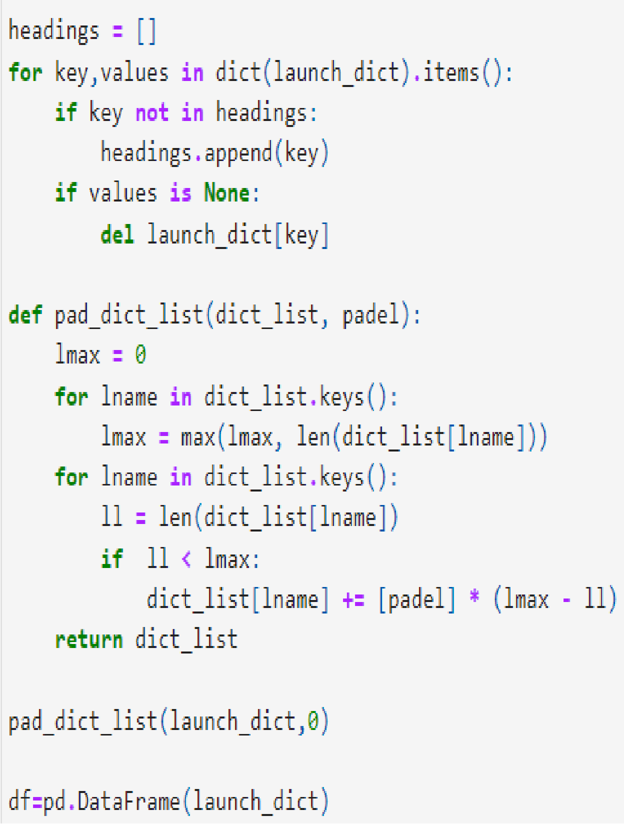
```
soup = BeautifulSoup(response.text, 'html.parser')
```

Extract all variables from the HTML tables



```
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
```

Create a data frame by parsing the launch HTML tables



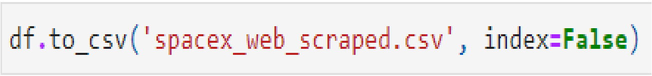
```
headings = []
for key,values in dict(launch_dict).items():
    if key not in headings:
        headings.append(key)
    if values is None:
        del launch_dict[key]

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(launch_dict)
```

Exporting data in to a CSV file

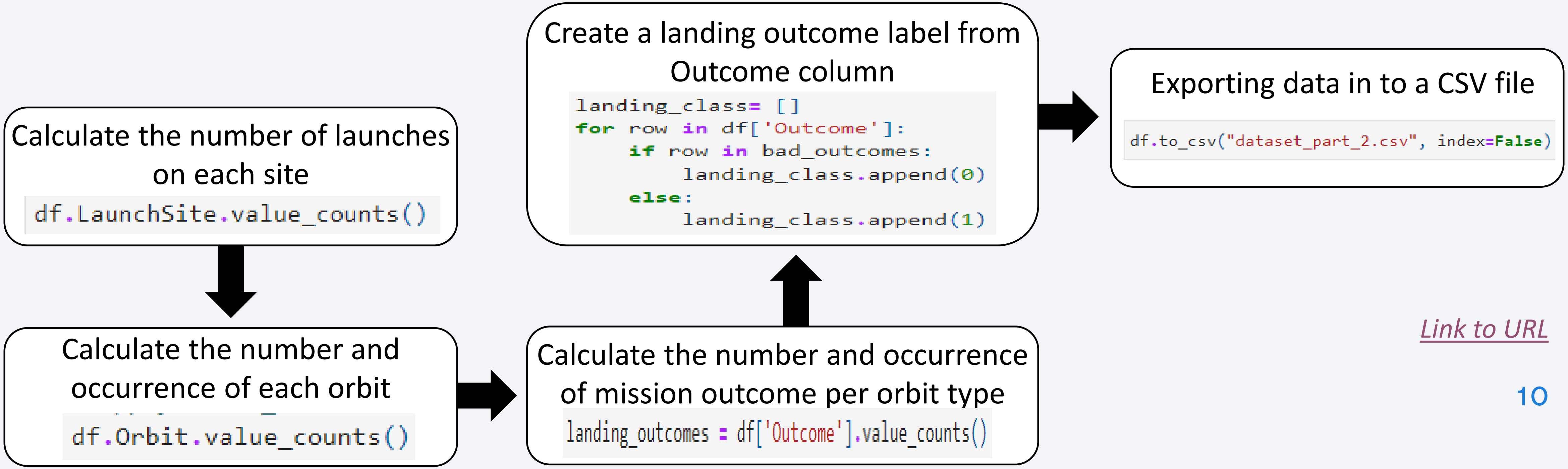


```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Link to URL

Data Wrangling

- The dataset for launches has information about different launch sites, launch orbit and mission outcome where:
  - True Ocean, True RTLS and True ASDS means successful landing of the booster.
  - False Ocean, False RTLS and False ASDS means unsuccessful landing of the booster.
- We will use these information to convert the outcomes into class label where 1 means the booster successfully landed and 0 means it was unsuccessful.



Calculate the number of launches on each site

```
df.LaunchSite.value_counts()
```

Calculate the number and occurrence of each orbit

```
df.Orbit.value_counts()
```

Calculate the number and occurrence of mission outcome per orbit type

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

Create a landing outcome label from Outcome column

```
landing_class= []
for row in df['Outcome']:
    if row in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

Exporting data in to a CSV file

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

Link to URL

EDA with Data Visualization

- Scatter Plot (To study the variation of one variable when another variable is changed)
  - Flight number vs. Payload mass
  - Flight number vs. Launch site
  - Payload mass vs. Launch site
  - Flight number vs. Orbit
  - Payload mass vs. Orbit

- Bar Plot (To study the relationship between a numeric and a categoric variable)
  - Success rate of each orbit

- Line Plot (To track changes over different periods of time)
  - Year vs. Average success rate

Link to URL

EDA with SQL

The list of SQL queries performed on the dataset to find:

- Names of the unique launch sites in the mission
- Show 5 records where the names of launch sites begin with 'CCA’
- Total payload mass carried by boosters launched by NASA (CRS)
- Average payload mass carried by booster version F9 v1.1
- Date for the first successful landing outcome in ground pad
- Names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- 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
- 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

Link to URL

Build an Interactive Map with Folium

- Folium map object centered at NASA Johnson Space Center at Houston, Texas
  - Yellow circle at NASA Johnson Space Center's location with label showing its name (folium.map.Marker, folium.Circle, folium.features.DivIcon)
  - Red circle at each launch site location with label showing its name (folium.map.Marker, folium.Circle, folium.features.DivIcon)
  - Grouping launch records for each location having the same coordinates based on their launch outcomes (folium.MarkerCluster)
  - Markers for all launch sites, with successful launches marked as green and unsuccessful launches marked as red (folium.map.Marker, folium.Icon)
  - Mouse pointer on map to get coordinates at any point (folium.plugins.MousePosition)
  - Line showing distance between a launch site to its proximities, like closest coastline, highway, railroad, city, etc. (folium.map.Marker, folium.PolyLine, folium.features.DivIcon)
- These objects will help us understand the geographical patterns about launch sites and the number of successful and unsuccessful landings

Link to URL

Build a Dashboard with Plotly Dash

- Different graphs and interactions added to the dashboard:
  - A drop-down menu to select the launch site or all launch site (dcc.Dropdown)
  - A pie chart visualizing launch success counts for selected launch site option (@app.callback, px.pie)
  - A range slider to select payload mass to easily select different payload range and see if there is a correlation with mission outcome (dcc.RangeSlider)
  - A payload vs. launch outcome scatter plot to visually observe how payload mass may be correlated with mission outcomes for selected site(s) (@app.callback, px.scatter)

Link to URL

Predictive Analysis (Classification)

- Preparing the data:
  - Load the data and Create a NumPy array from the column “class”
  - Standardize the data and reassign it to the variable using the transform
  - Splitting the data into test and train datasets
- Preparing the model:
  - Selecting the models for evaluation
  - Create a logistic regression object then create a GridSearchCV object
  - Fit the object using training data

- Evaluation of the model:
  - Find the best parameters for the model
  - Calculate the accuracy on the test data
  - Plot the confusion matrix for each model
- Comparison of the models:
  - Comparing the selected models on the basis of the calculated accuracy to find the method that performs best

Link to URL

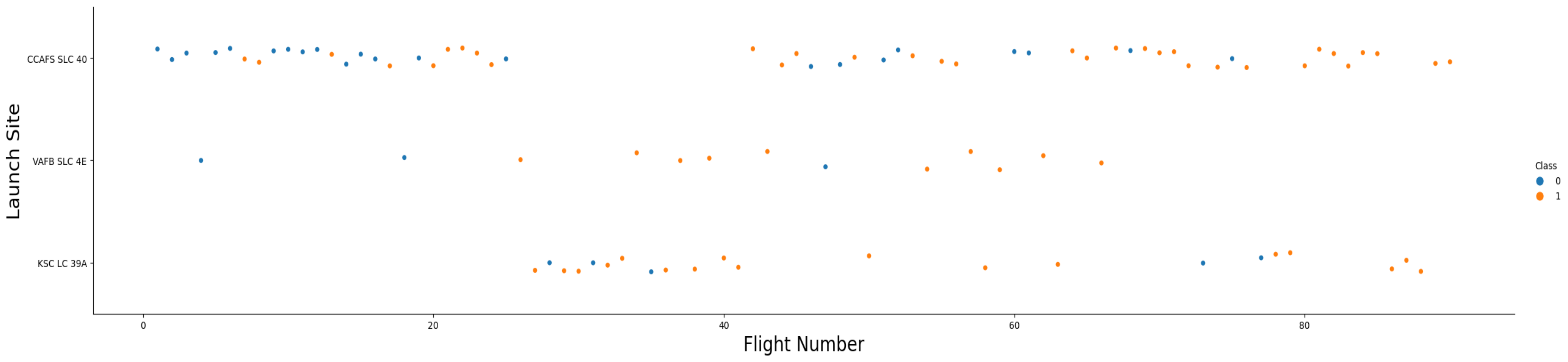
Results

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

Section 2

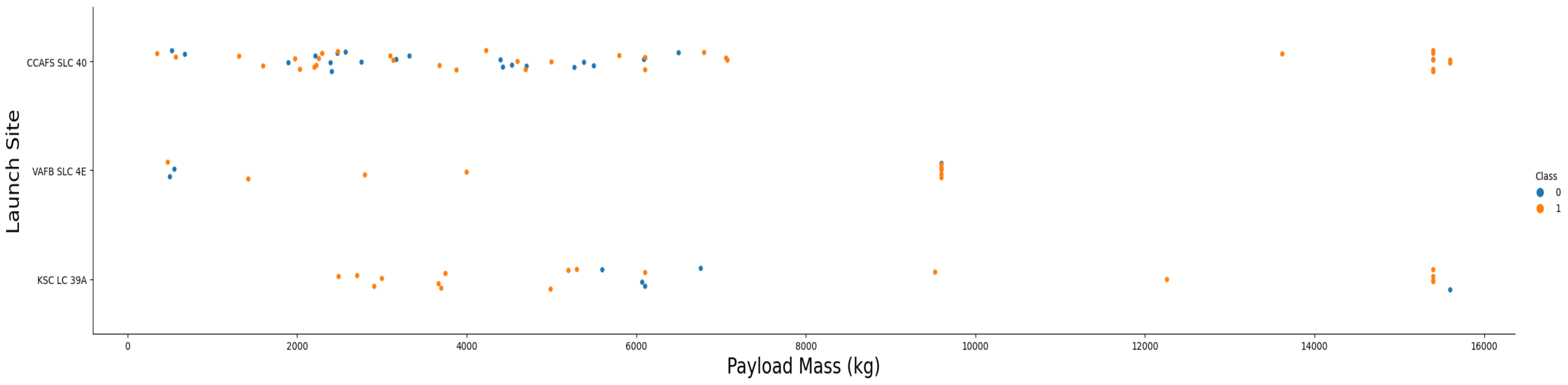
Insights drawn from EDA

Flight Number vs. Launch Site



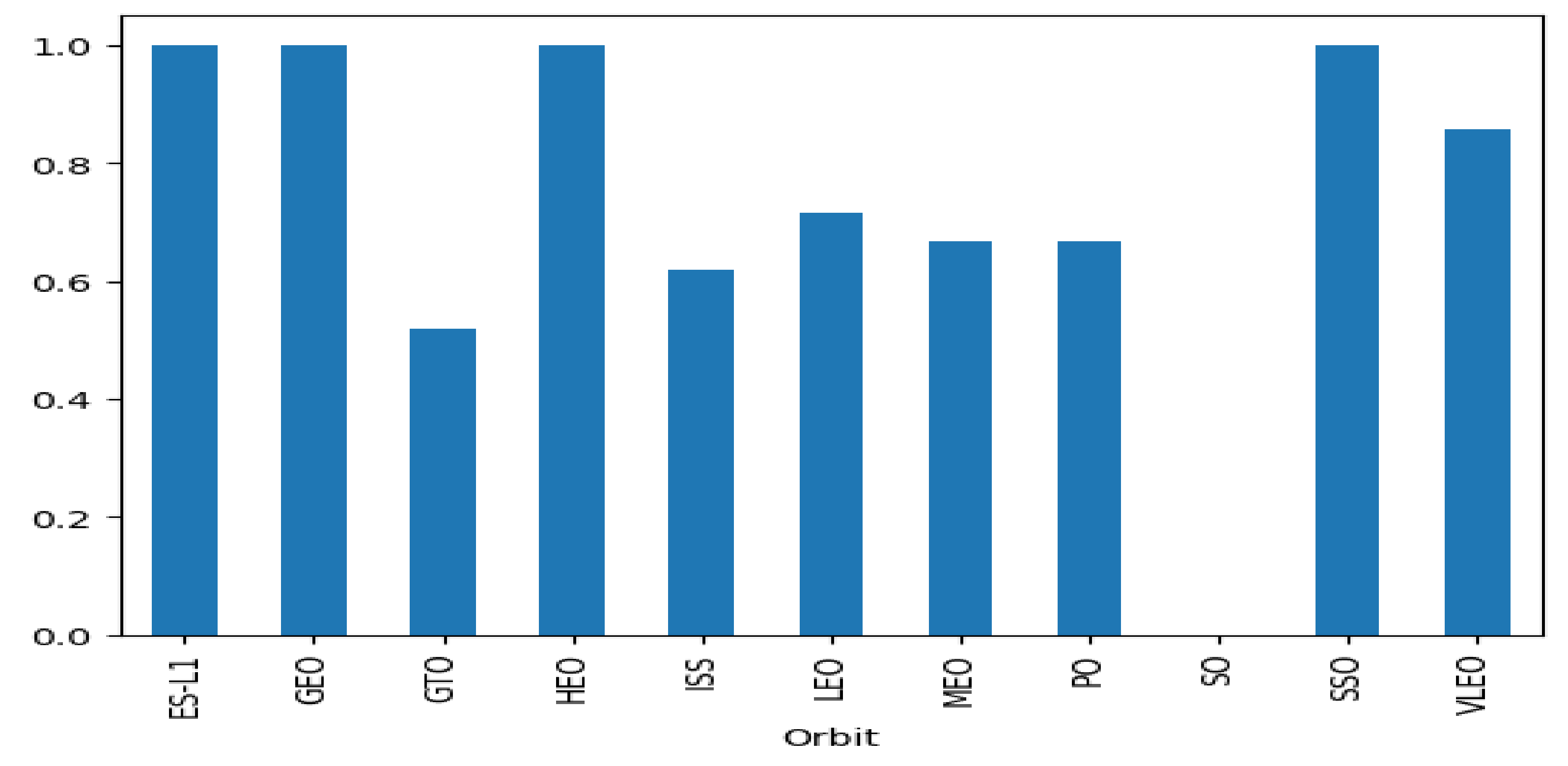
The successful landings for each launch site is increasing with increasing flight number.

Payload vs. Launch Site



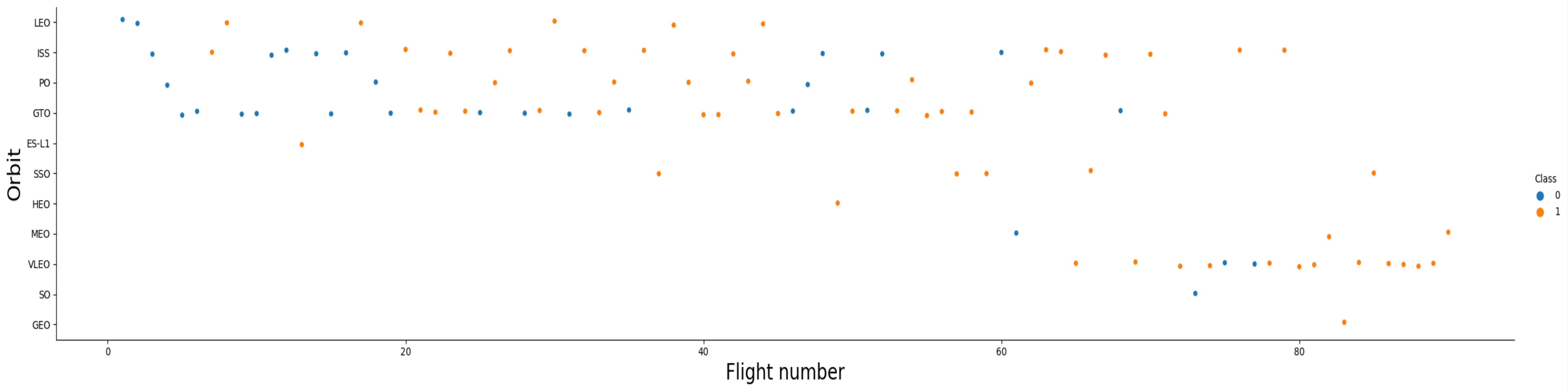
There is an increase in successful landings for each launch site with increasing payload mass. But a too heavy payload can have an unsuccessful landing.

Success Rate vs. Orbit Type



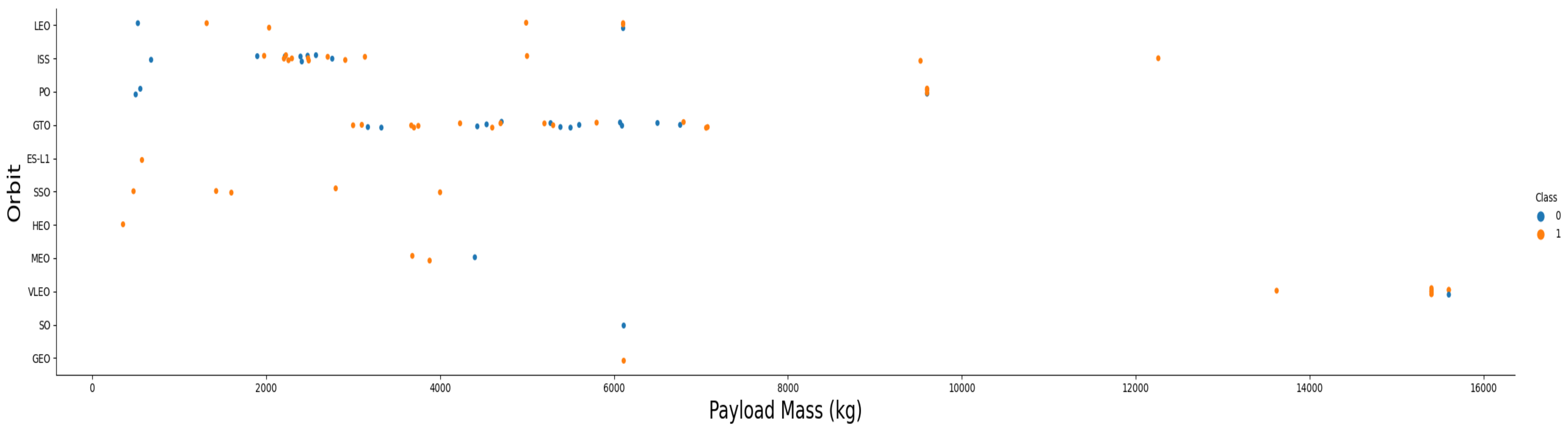
- The orbit types having best success rate: ES-L1, GEO, HEO, VLEO and SSO.
  - The orbit types having average success rate: ISS, LEO, MEO and PO.
    - The orbit types having bad success rate: GTO and SO.

Flight Number vs. Orbit Type



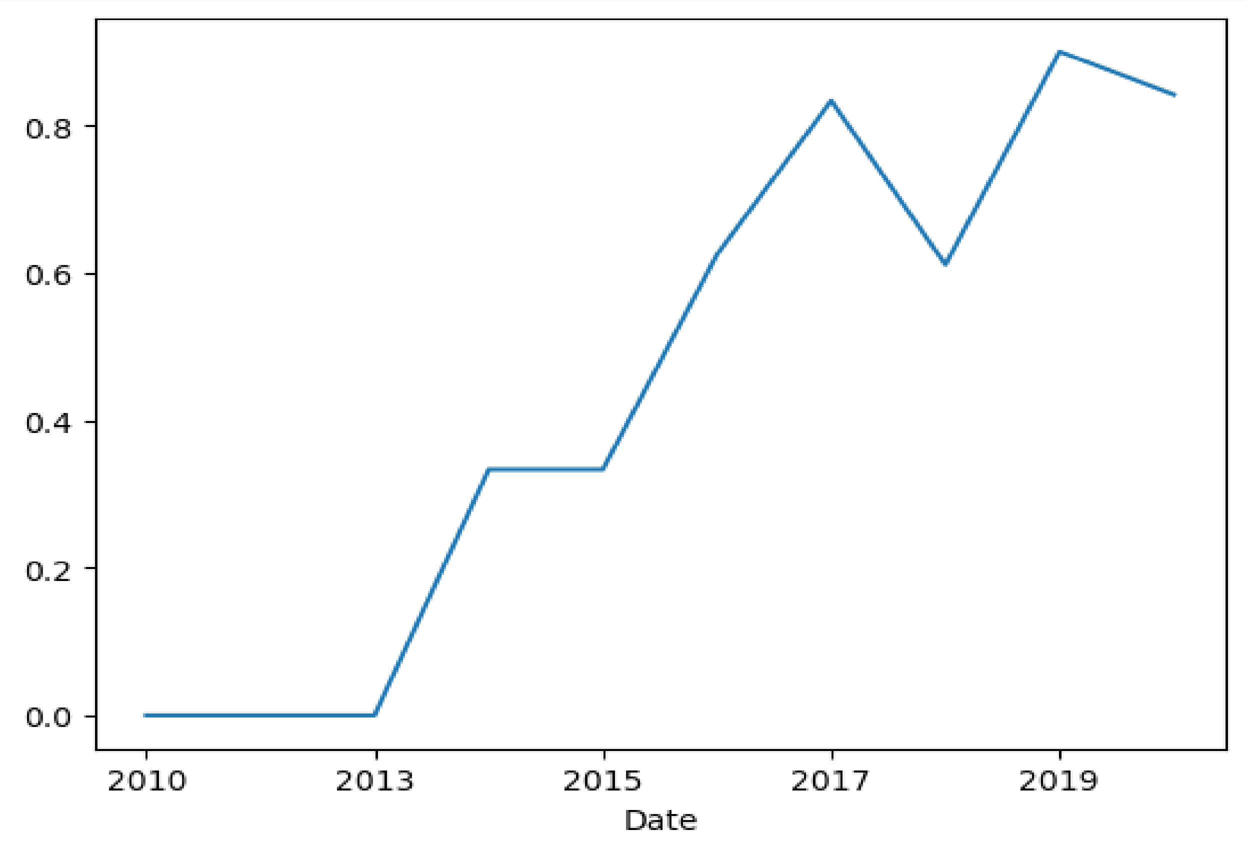
- On an average the successful landings increases with increasing flight number for different orbits.
- For the LEO orbit, the success rate appears related to the number of flights; on the other hand, there seems to be no relationship between flight number and success rate when in the GTO orbit.
- There are few orbits like ES-L1, SSO, HEO, VLEO and GEO which shows a high success rate as compared to other orbits.

Payload vs. Orbit Type



- With heavy payloads the successful landing are more for the PO, LEO and ISS orbits.
- For the GTO orbit we cannot distinguish the relation between payload mass and successful landing as both positive and negative landing rates are distributed throughout the payload mass range.

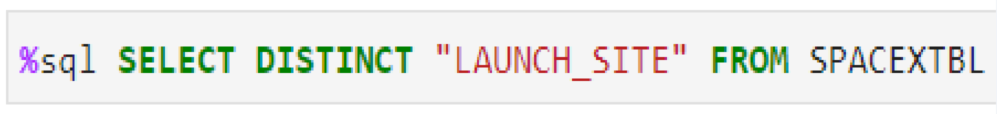
Launch Success Yearly Trend



The yearly average success rate is increasing with passing year.

All Launch Site Names

- SQL query:

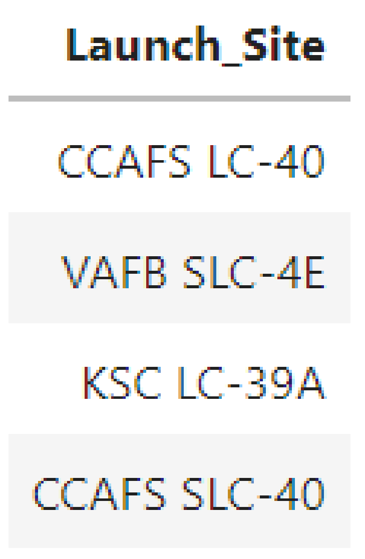


```
%sql SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL
```

- Explanation:

This query is used to display the names of the unique launch sites in the space mission. The command “Distinct” helps to remove any duplicate entry to the list of launch sites.

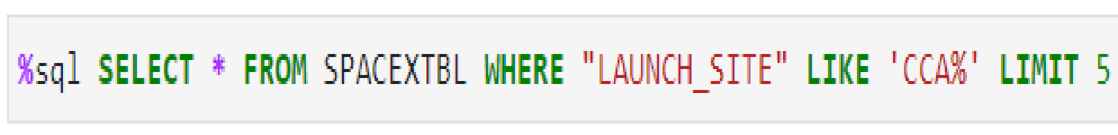
- Names of the unique launch sites:



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

Launch Site Names Begin with 'CCA'

- SQL query:

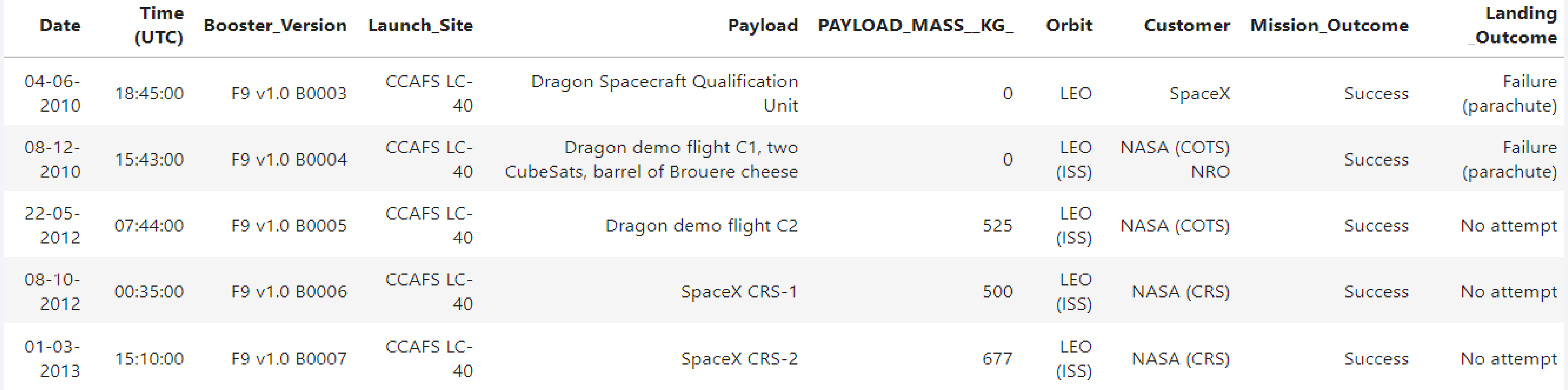


```
%sql SELECT * FROM SPACEXTBL WHERE "LAUNCH_SITE" LIKE 'CCA%' LIMIT 5
```

- 5 records where launch sites begin with “CCA”:

- Explanation:

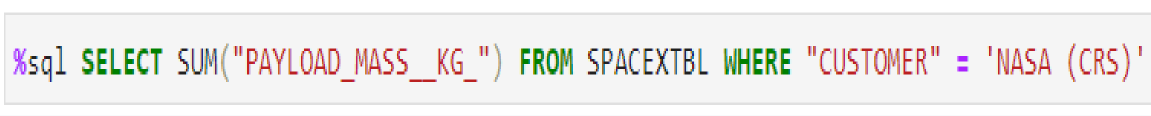
The “LIKE” command helps to choose the names beginning with the substring CCA and “LIMIT 5” command is used to specify the number of records to return.



Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- SQL query:

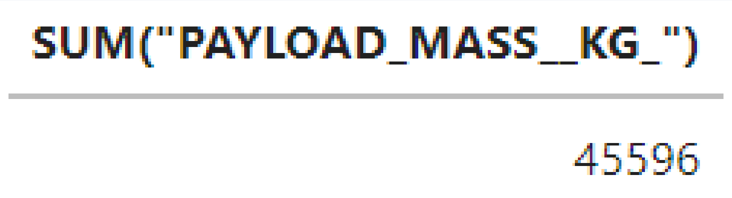


```
%sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "CUSTOMER" = 'NASA (CRS)'
```

- Explanation:

  The command “SUM” in the query is used to find the total payload mass carried by boosters launched by NASA(CRS) in kg.

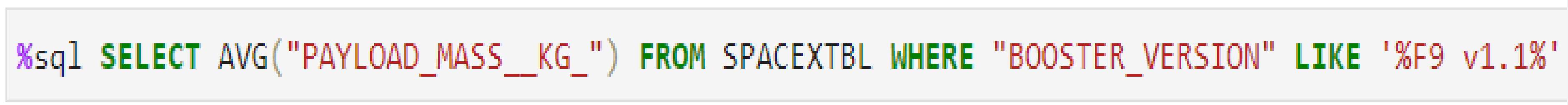
- Total payload carried by boosters from NASA:



SUM("PAYLOAD_MASS__KG_")
45596

Average Payload Mass by F9 v1.1

- SQL query:

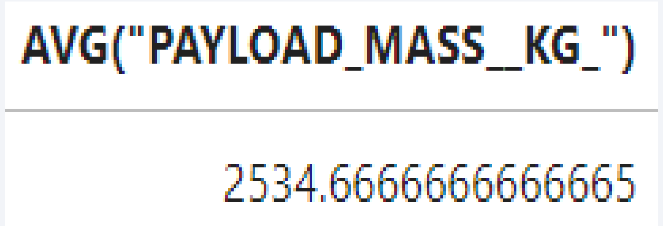


```
%sql SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "BOOSTER_VERSION" LIKE '%F9 v1.1%'
```

- Explanation:

The command "AVG" in the query is used to find the average of the payload mass carried by boosters having version F9 v1.1.

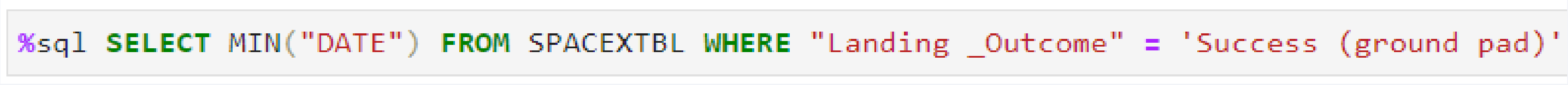
- Average payload mass carried by booster version F9 v1.1:



AVG("PAYLOAD_MASS__KG_")
2534.6666666666665

First Successful Ground Landing Date

- SQL query:

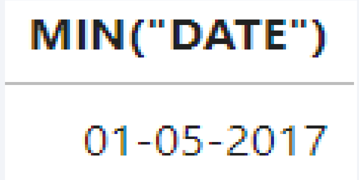


```
%sql SELECT MIN("DATE") FROM SPACEXTBL WHERE "Landing _Outcome" = 'Success (ground pad)'
```

- Explanation:

  The command “min” selects the first date from the list when the landing outcome was a success for the ground pad.

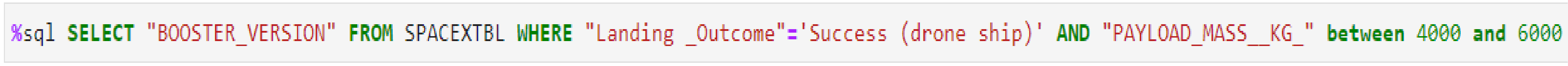
- Dates of the first successful landing outcome on ground pad:



MIN("DATE")
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

- SQL query:

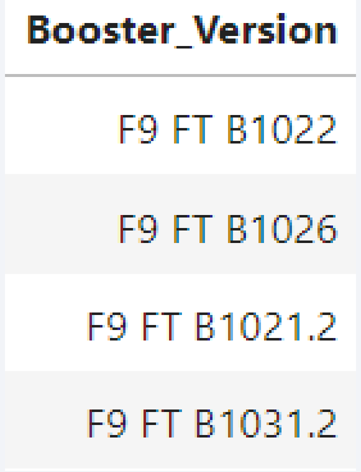


```
%sql SELECT "BOOSTER_VERSION" FROM SPACEXTBL WHERE "Landing _Outcome"='Success (drone ship)' AND "PAYLOAD_MASS__KG_" between 4000 and 6000
```

- Explanation:

  The query selects the names of booster version from the dataset whose landing outcome was a success on drone ship “AND” also have payload mass varying between 4000 to 6000 kg.

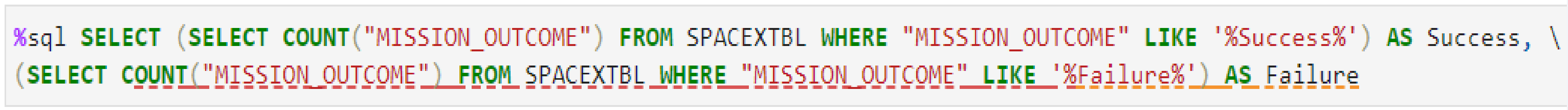
- Names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000:



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

Total Number of Successful and Failure Mission Outcomes

- SQL query:

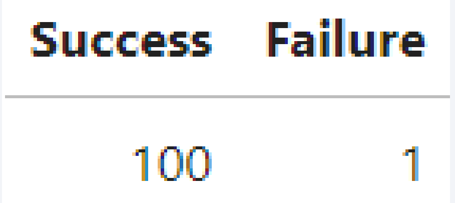


```
%sql SELECT (SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Success%') AS Success, \
(SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Failure%') AS Failure
```

- Explanation:

  The command "COUNT" returns the number of rows that matches the specified criteria where the mission outcome are success or failure.

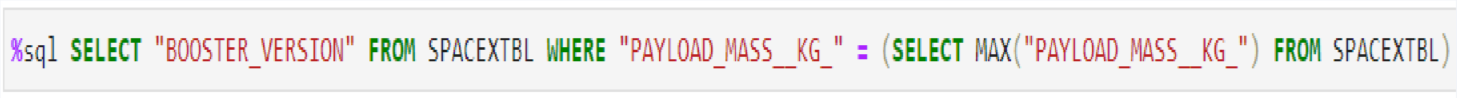
- Total number of successful and failure mission outcomes:



Success	Failure
100	1

Boosters Carried Maximum Payload

- SQL query:

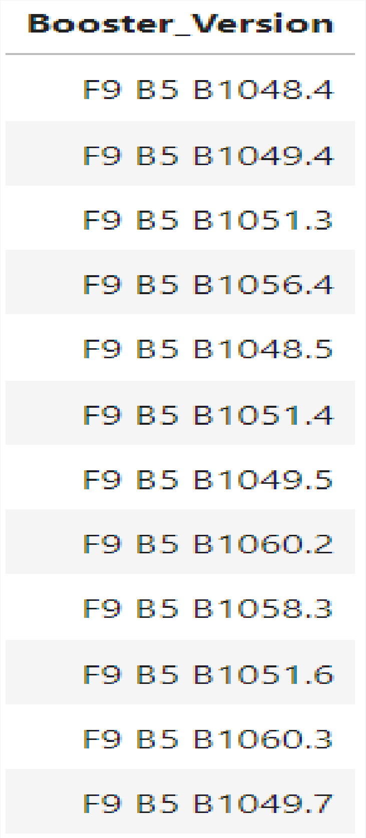


```
%sql SELECT "BOOSTER_VERSION" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL)
```

- Explanation:

  The subquery helps to choose the maximum payload mass from the table and then select the booster versions related to it.

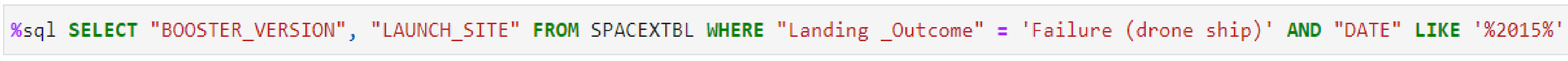
- Names of the booster which have carried the maximum payload mass:



Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

2015 Launch Records

- SQL query:

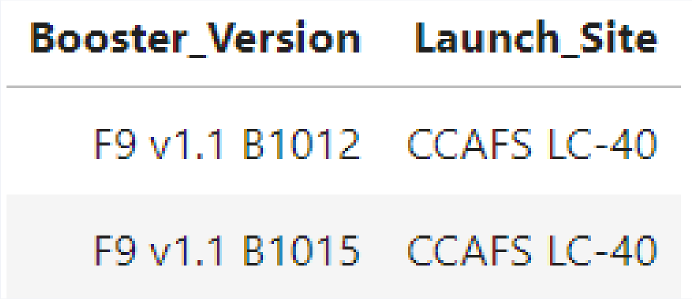


```
%sql SELECT "BOOSTER_VERSION", "LAUNCH_SITE" FROM SPACEXTBL WHERE "Landing _Outcome" = 'Failure (drone ship)' AND "DATE" LIKE '%2015%'
```

- Explanation:

  The query selects the booster version and launch site for failed landing in drone ship for the year 2015. The “LIKE” command helps to choose the date which has the year 2015 in them.

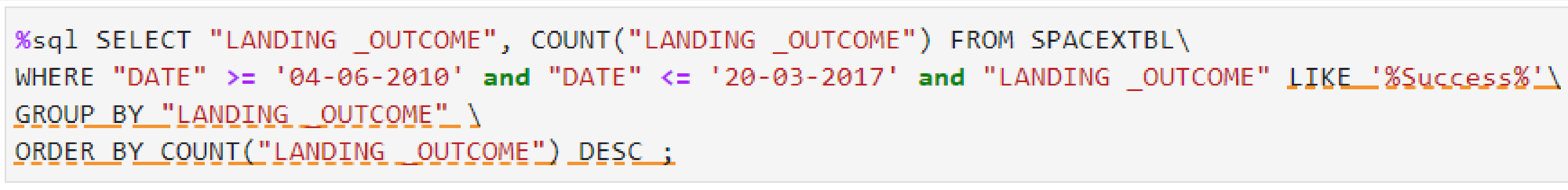
- Failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015:



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

- SQL query:

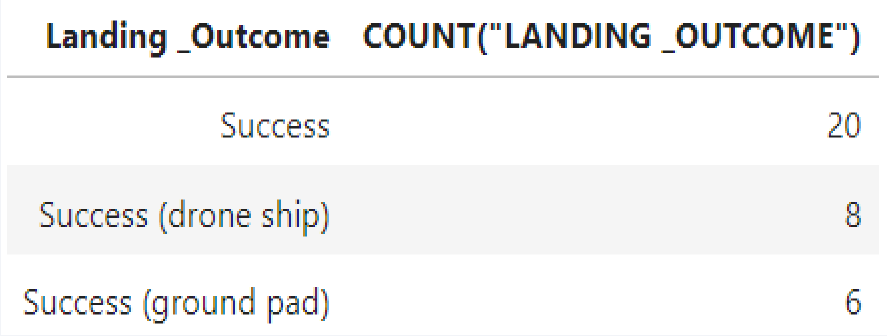


```
%sql SELECT "LANDING _OUTCOME", COUNT("LANDING _OUTCOME") FROM SPACEXTBL\
WHERE "DATE" >= '04-06-2010' and "DATE" <= '20-03-2017' and "LANDING _OUTCOME" LIKE '%Success%'\
GROUP BY "LANDING _OUTCOME" \
ORDER BY COUNT("LANDING _OUTCOME") DESC ;
```

- Explanation:

The query selects and counts the successful landing outcomes between the specified dates. It uses the command "GROUP BY" to group the successful landings for each outcome and finally uses the commands "ORDER BY COUNT" and "DESC" to arrange the rank in descending order.

- Rank the count of successful landing outcomes between the date 2010-06-04 and 2017-03-20 in descending order:

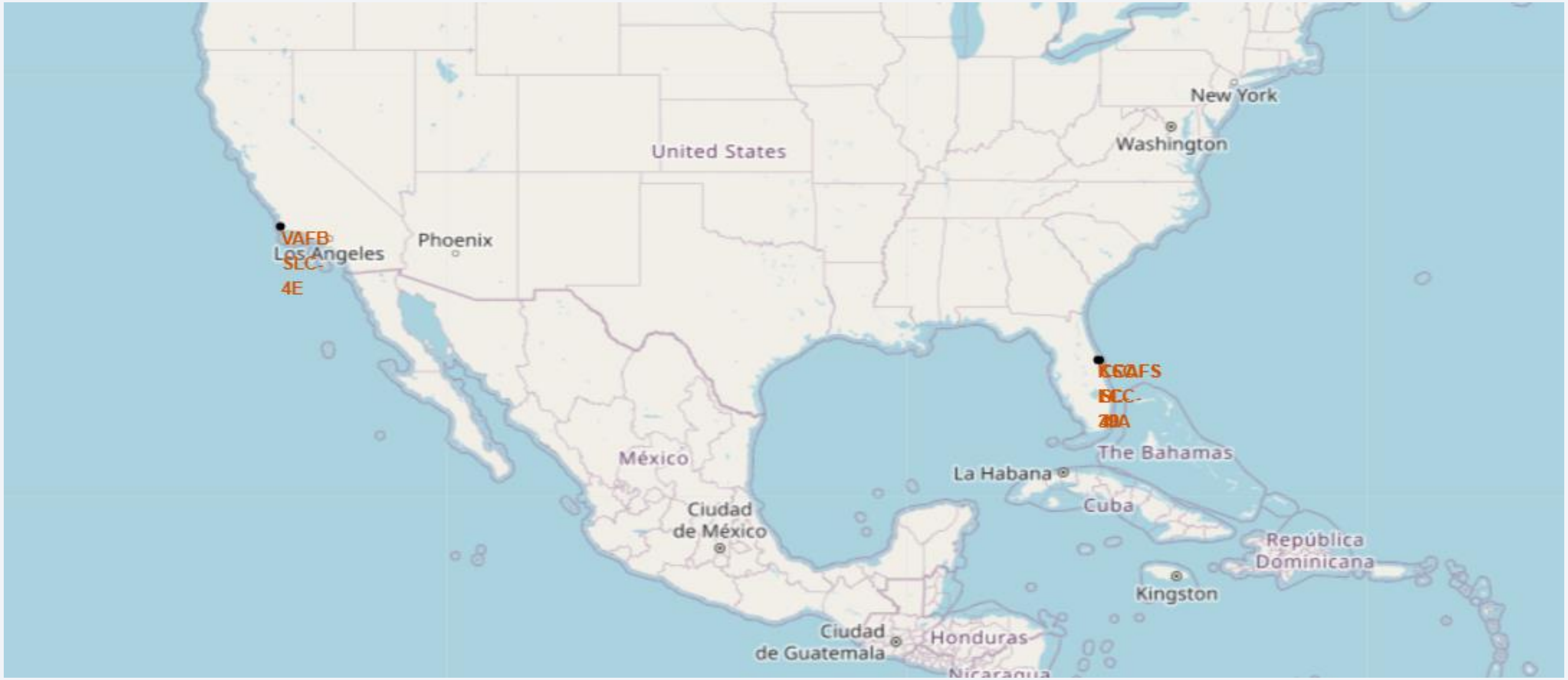


Landing _Outcome	COUNT("LANDING _OUTCOME")
Success	20
Success (drone ship)	8
Success (ground pad)	6

Section 3

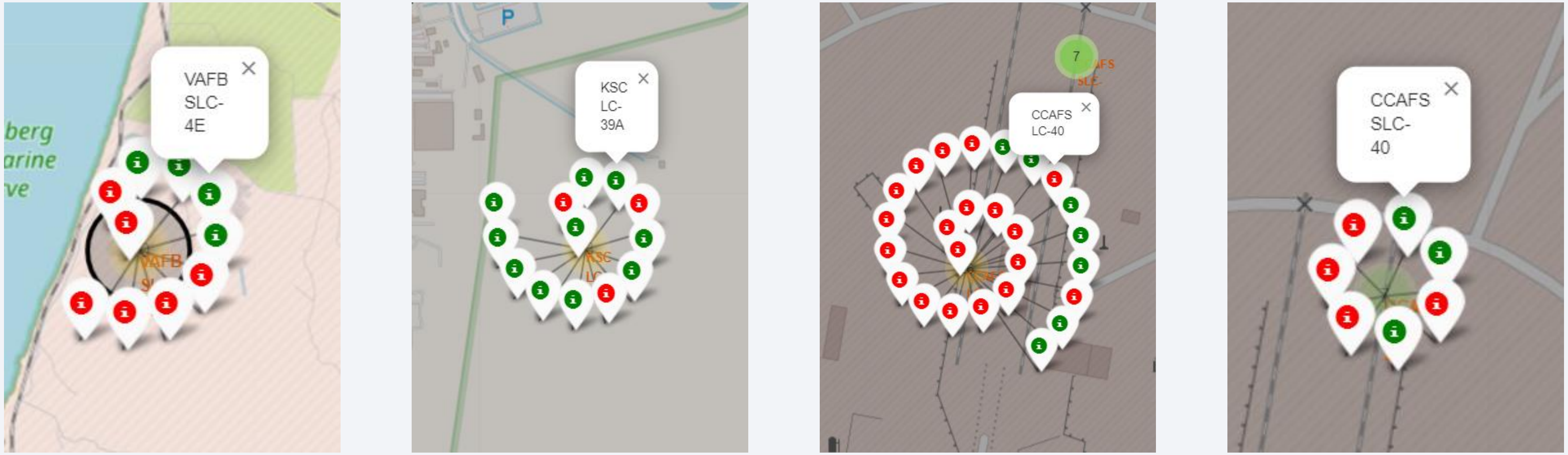
Launch Sites Proximities Analysis

Folium Map – All launch sites



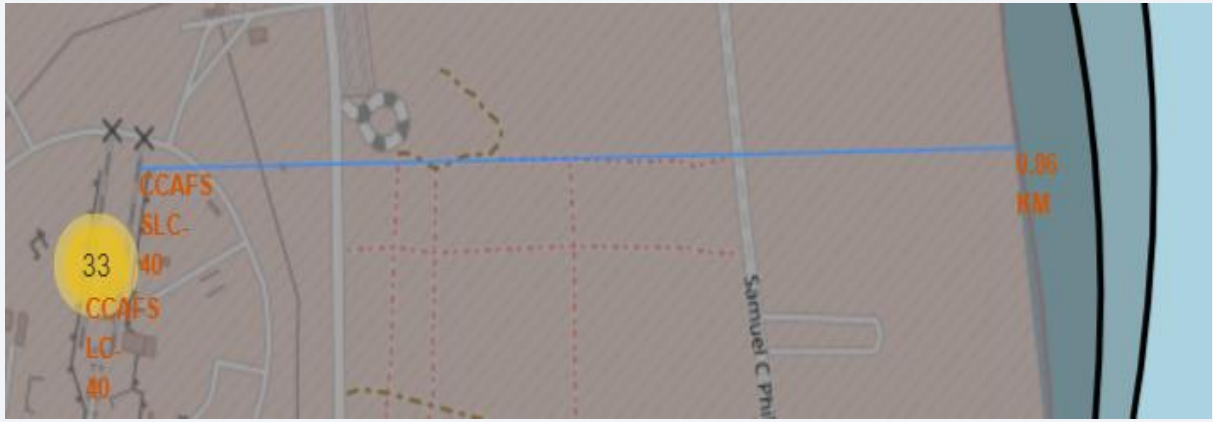
All launch sites are near the coastline of the United States of America.

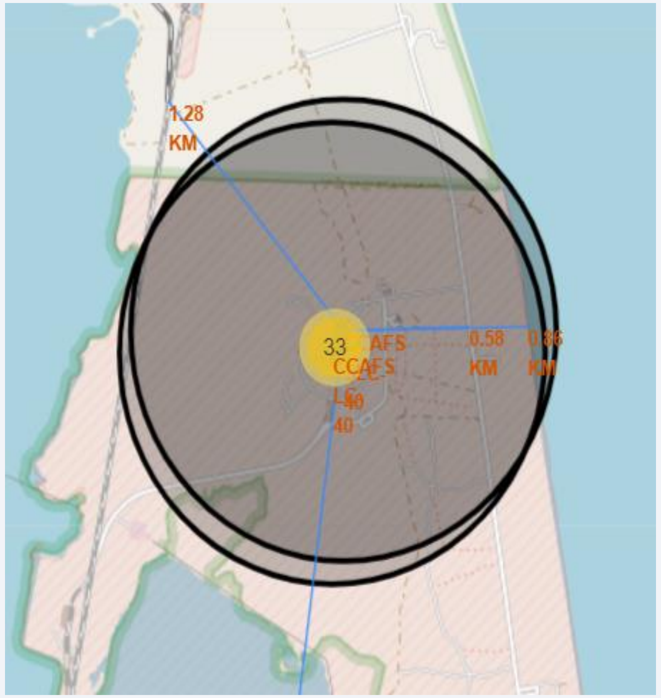
Folium Map – Color labeled launch outcomes

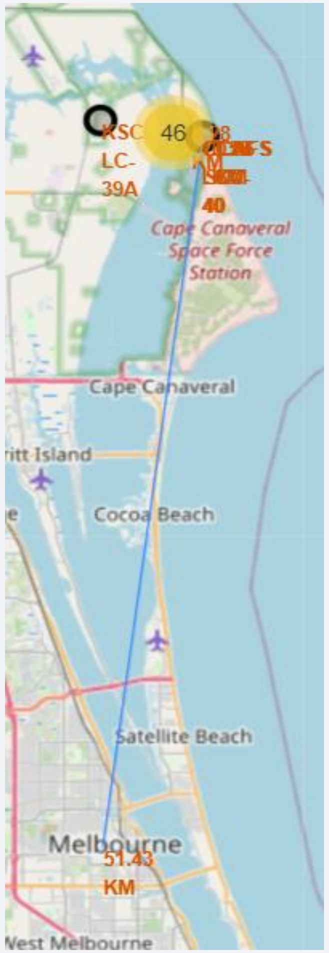


The green markers show successful launches whereas the red markers show unsuccessful launches at each site. We can note that launch site KSC LC-39 A has higher success rate than any other site.

Folium Map - Launch site to its proximities





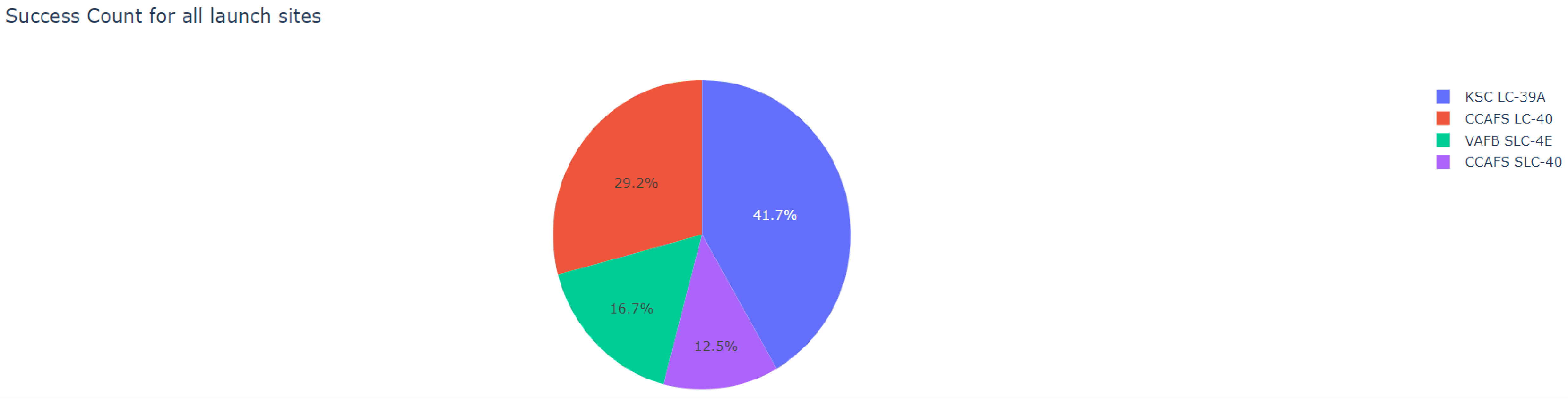


- Is CCAFS SLC-40 in close proximity to railways?- Yes (~ 1.3 km)
- Is CCAFS SLC-40 in close proximity to highways?- Yes (~ 0.6 km)
- Is CCAFS SLC-40 in close proximity to coastline?- Yes (~ 0.9 km)
- Do CCAFS SLC-40 keep certain distance away from closest city?- Yes (Melbourne is ~ 51.4 km apart)

Section 4

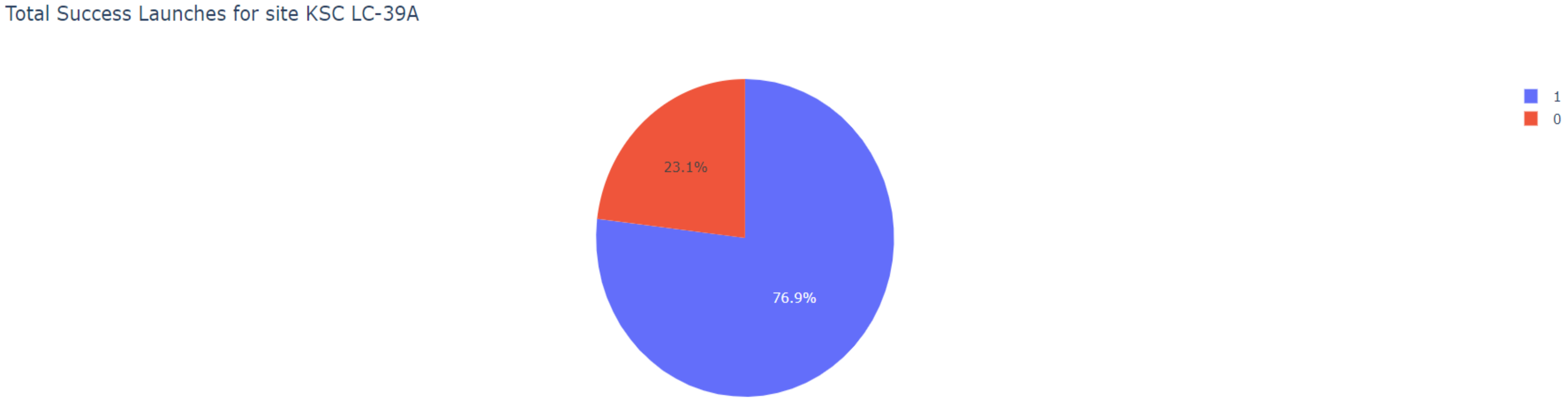
Build a Dashboard with Plotly Dash

Dashboard - Success count for all sites



The highest launch success count is for the KSC LC-39A site.

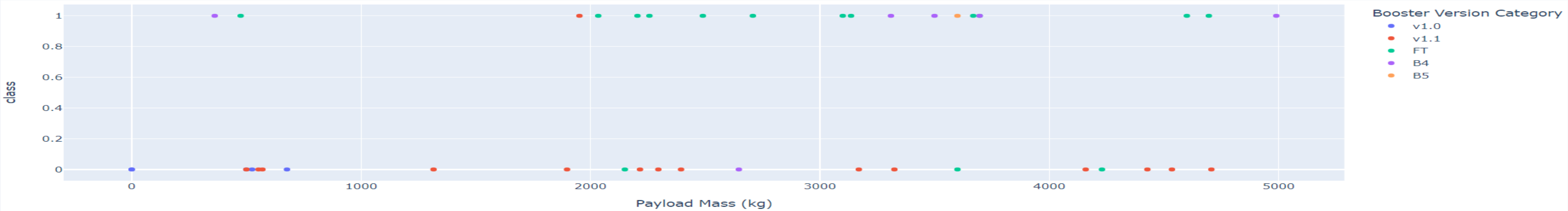
Dashboard – Total success launches for KSC LC-39A site



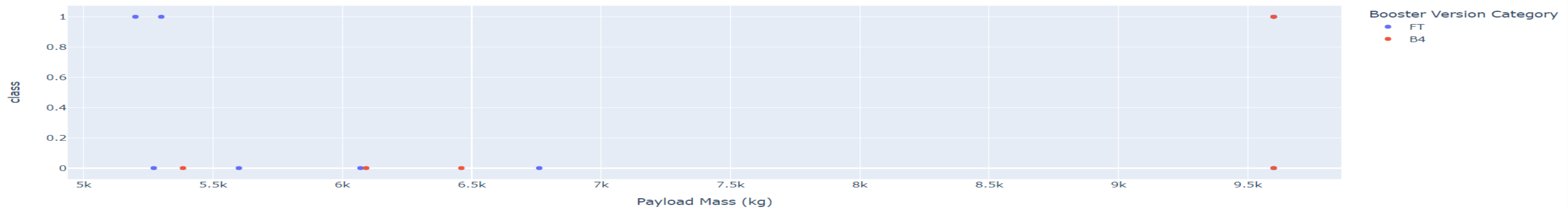
We can see that the percentage of successful launches for this site is 76.9%, whereas the percentage of unsuccessful launches is 23.1%.

Dashboard – Success count on payload mass for all sites

Success counts for low weight payload (0 to 5000 kg)



Success counts for high weight payload (5000 to 10000 kg)

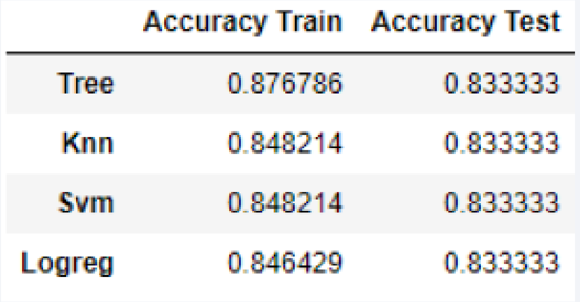


- On comparison, the low weight payloads perform better than the high weight payloads.
- For both low and high weight payloads, the booster FT performs best.

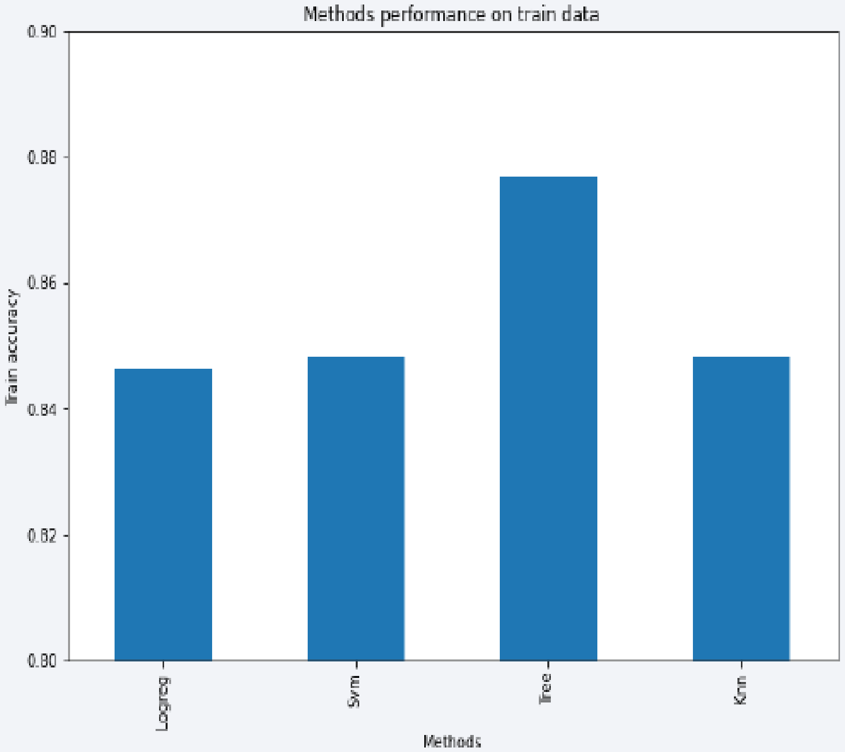
Section 5

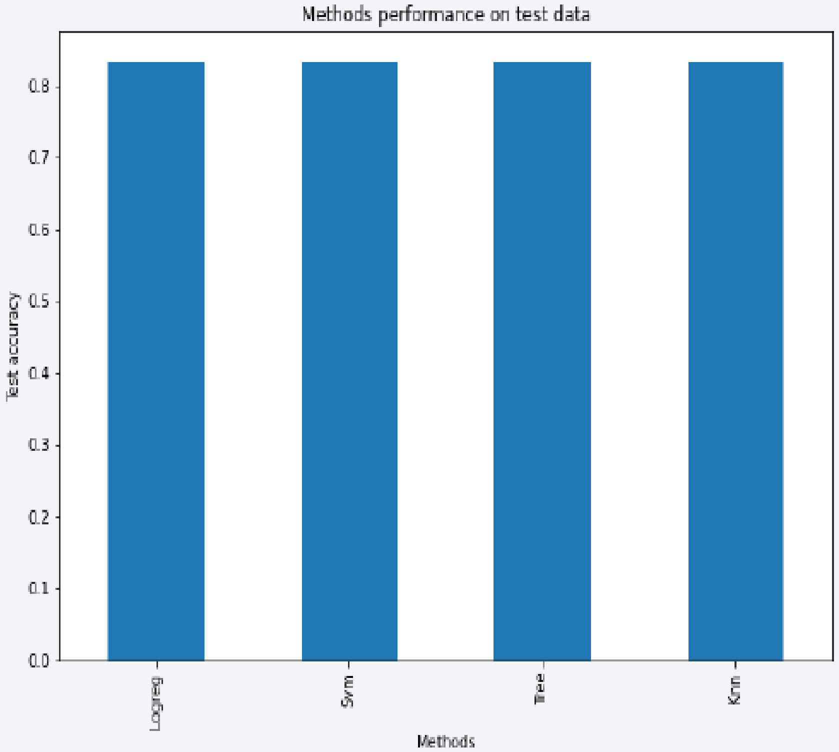
Predictive Analysis (Classification)

Classification Accuracy



	Accuracy Train	Accuracy Test
Tree	0.876786	0.833333
Knn	0.848214	0.833333
Svm	0.848214	0.833333
Logreg	0.846429	0.833333

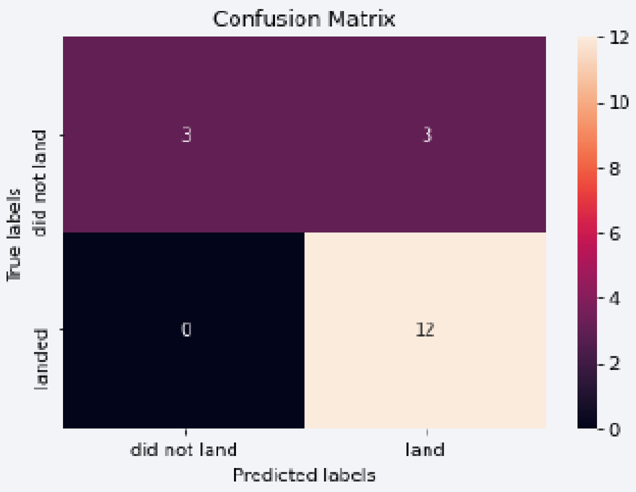




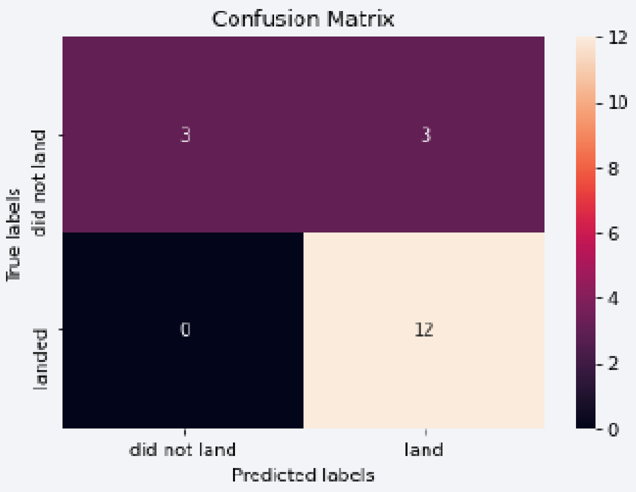
- For accuracy test all models performed equally.
- Out of the four, we choose the best performing model to be the decision tree.

Confusion Matrix

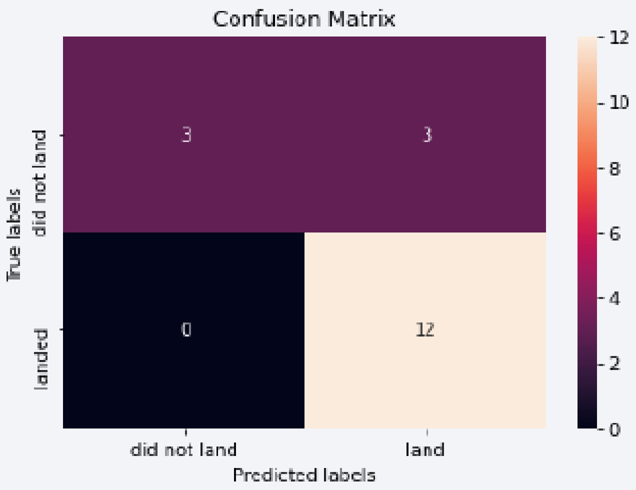
SVM



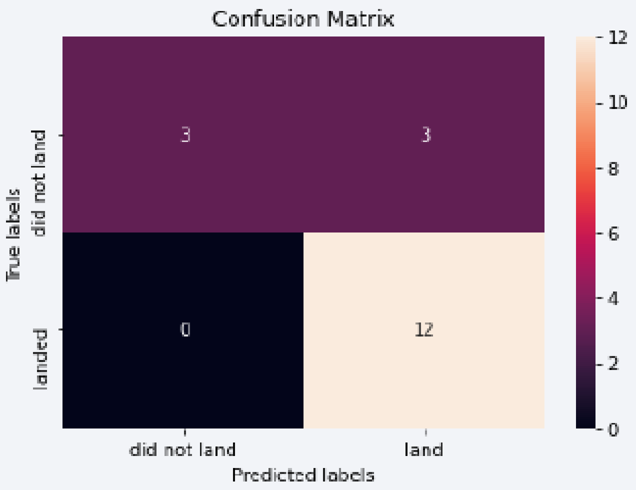
Decision Tree



KNN



LogReg



- As the test accuracy are all equal, the confusion matrices are also identical.
- The main problem of these models are false positives.

Conclusions

- There is an increase in the landing success rate with an increasing flight number for different orbits.
- There is an increase in the landing success rate with payload mass, but a too heavy payload can have an unsuccessful landing. Hence, the low weight payloads perform better than the high weight payloads. With heavy payloads the successful landing are more for the PO, LEO and ISS orbits. For the GTO orbit we cannot distinguish the relation between payload mass and successful landing.
- There are few orbits like ES-L1, SSO, HEO, VLEO and GEO which shows a high success rate as compared to other orbits. The yearly average success rate is increasing with passing year.
- We observe that the launch site KSC LC-39 A has higher success rate than any other site. All launch sites are in close proximity to the coastline. In addition, the launch site CCAFS SLC-40 is in close proximity to railway line and highway but is distant from the nearest city.
- The test accuracy for all models are all equal, so we choose the best model based on the accuracy achieved by training dataset. The model which performs the best for the current dataset is the Decision Tree Algorithm.

Thank you!