IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Caspar Yan Hansen

01-06-2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction

- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result

Introduction

- Project background and context

Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers
    - What determines successful landings?
    - Are there any interactions amongst factors determines success?
    - What operating conditional need to be in place to ensure success?

Section 1

Methodology

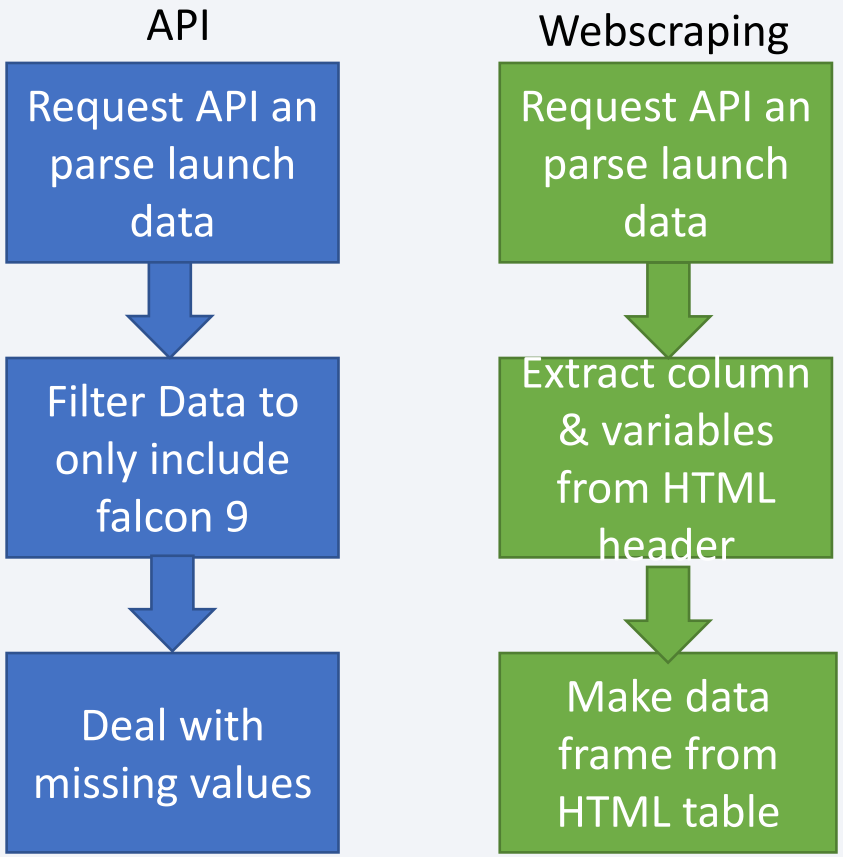
Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- 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 data was collected using various methods
  - Data collection was done using get request to the SpaceX API.
  - Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
  - We then cleaned the data, checked for missing values and fill in missing values where necessary.
  - In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
  - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.



Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose

- Source Code: [capstone/Caspar data collection.ipynb at master · casparyan/capstone · GitHub]()

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_
```

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
# 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 and the flight number, and
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boos
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 single value in the list an
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 extracting the date leaving
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)]
```

```
In [22]: # Create a data from launch_dict
         df = pd.DataFrame.from_dict(launch_dict)
```

```
data_falcon9.isnull().sum()
```

```
FlightNumber       0
Date               0
BoosterVersion     0
PayloadMass        0
Orbit              0
LaunchSite         0
Outcome            0
Flights            0
GridFins           0
Reused             0
Legs               0
LandingPad        26
Block              0
ReusedCount        0
Serial             0
Longitude          0
Latitude           0
dtype: int64
```

You should see the number of missing values of the `PayLoadMass` change to zero.

Now we should have no missing values in our dataset except for in `LandingPad`.

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

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose

- [capstone/Caspar Webscraping.ipynb at master · casparyan/capstone (github.com)]()

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

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

Print the page title to verify if the `BeautifulSoup` object was created properly

```python
# Use soup.title attribute
print(soup.title)
```

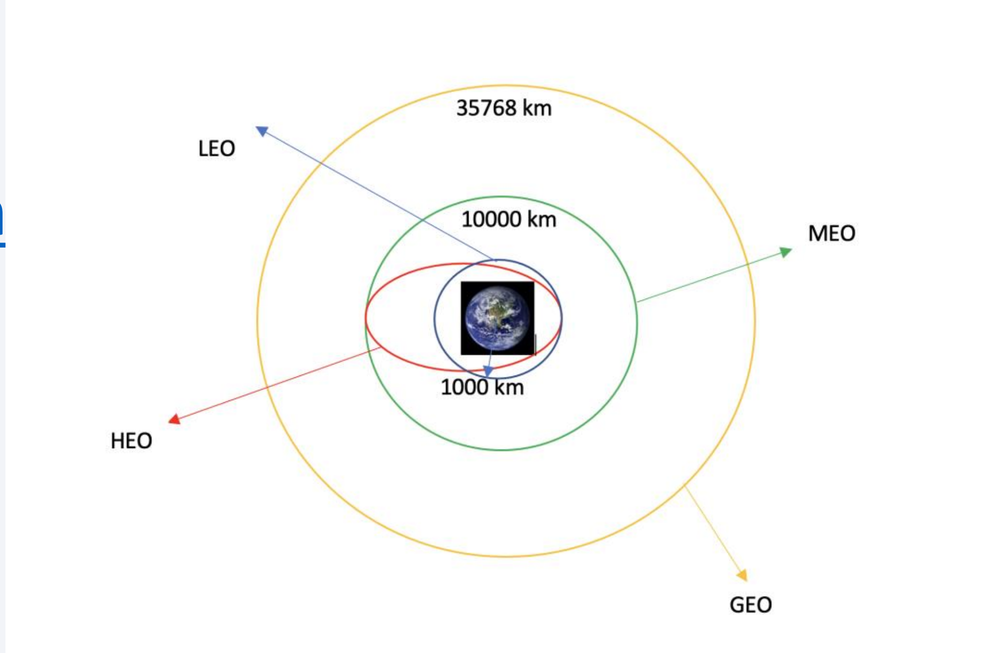
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

```python
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()
```

Data Wrangling

- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.

- [capstone/Caspar Datawrangling.ipyn casparyan/capstone (github.com)]



EDA with Data Visualization

- Scatter plots
    - Used to show much much a variable affected another. Shows correlation between two variables.
- Bar Graphs
    - Compare different groups at quick glance. You can put categorical values on the x axis and continuous on y.
- Line Graphs
    - Very useful for showing trends and predicting output
- capstone/Caspar EDA wiz.ipynb at master · casparyan/capstone (github.com)

EDA with SQL

- Names of unique launch sites
- Top 5 launch sites that begins with CCA
- Total payload carried by NASA
- Average paylod by booster version F9 v1.1
- First successful landing in ground pad
- List of booster names that have success in drone ship and larger payload than 4000 but less than 6000
- List of total number of success and fail missions
- List all the booster versions that have carried max load
- List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
- Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

- capstone/jupyter-labs-eda-sql-coursera_sqllite.ipynb at master · casparyan/capstone (github.com)

Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
  - Are launch sites near railways, highways and coastlines.
  - Do launch sites keep certain distance away from cities.

capstone/Caspar folium (1) (1).ipynb at master · casparyan/capstone (github.com)

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
- Build an interactive dashboard in Plotly
- Added a pie chart to show total launches and part of total launces from certain sites
- Scatter plot to show relationship between outcome, payload and booster version.
- capstone/app.py at master · casparyan/capstone (github.com)

Predictive Analysis (Classification)

- Model Build
  - Loaded dataset in NumPy and pandas
  - Split data into training and test dataset
  - Set parameters via GridSearchCV
  - Train via GridSeachCV objects fitting
- Evaluate
  - Calculate accuracy for each model
  - Tune hyperparameters
  - Make Confusion matrix
- Model improvement
  - Feature engineering
    Tuning
- capstone/Caspar_Machine_Learning_Prediction_Part_5.jupyterl.ipynb at master · casparyan/capstone (github.com)

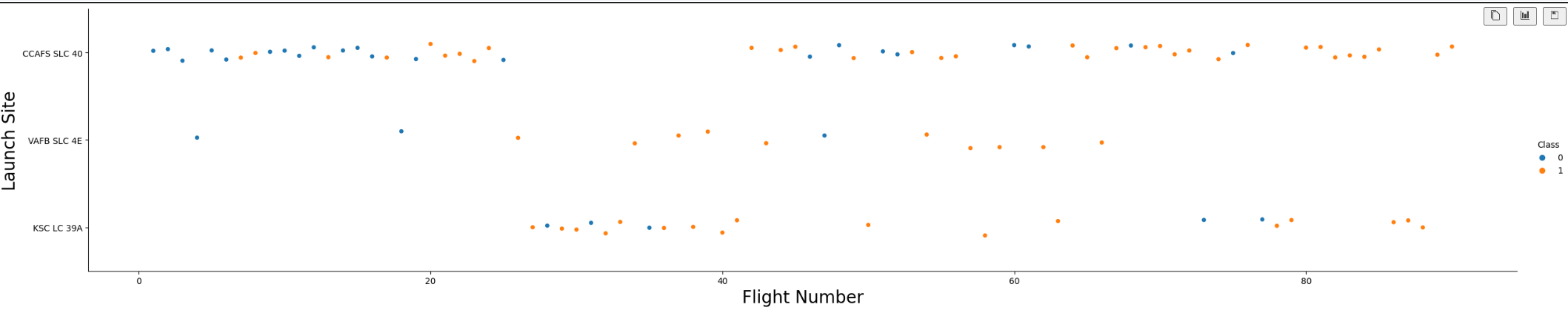
Results

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

Section 2

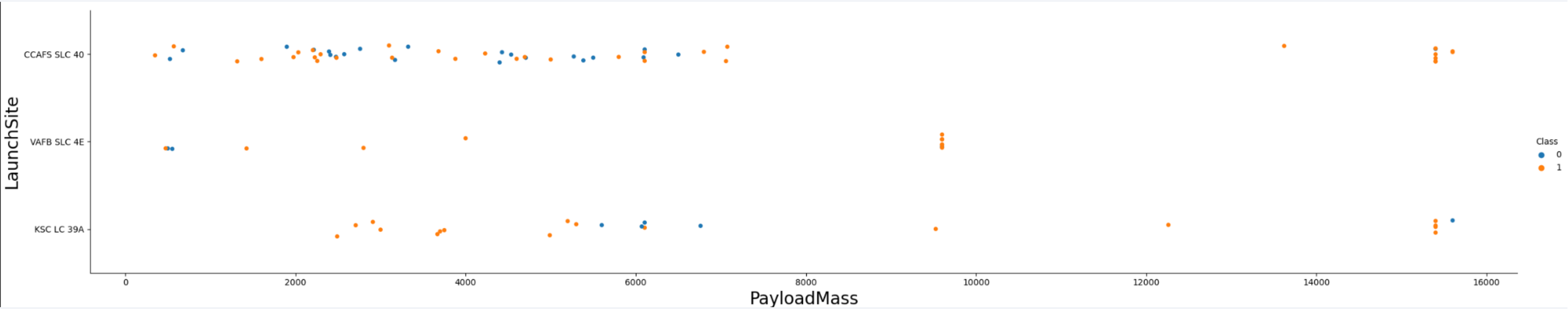
Insights drawn from EDA

Flight Number vs. Launch Site



- The higher the flightnumber from a launch site then you have a higher success rate
- This is also reflected in CCAF early launches that had relative more failures

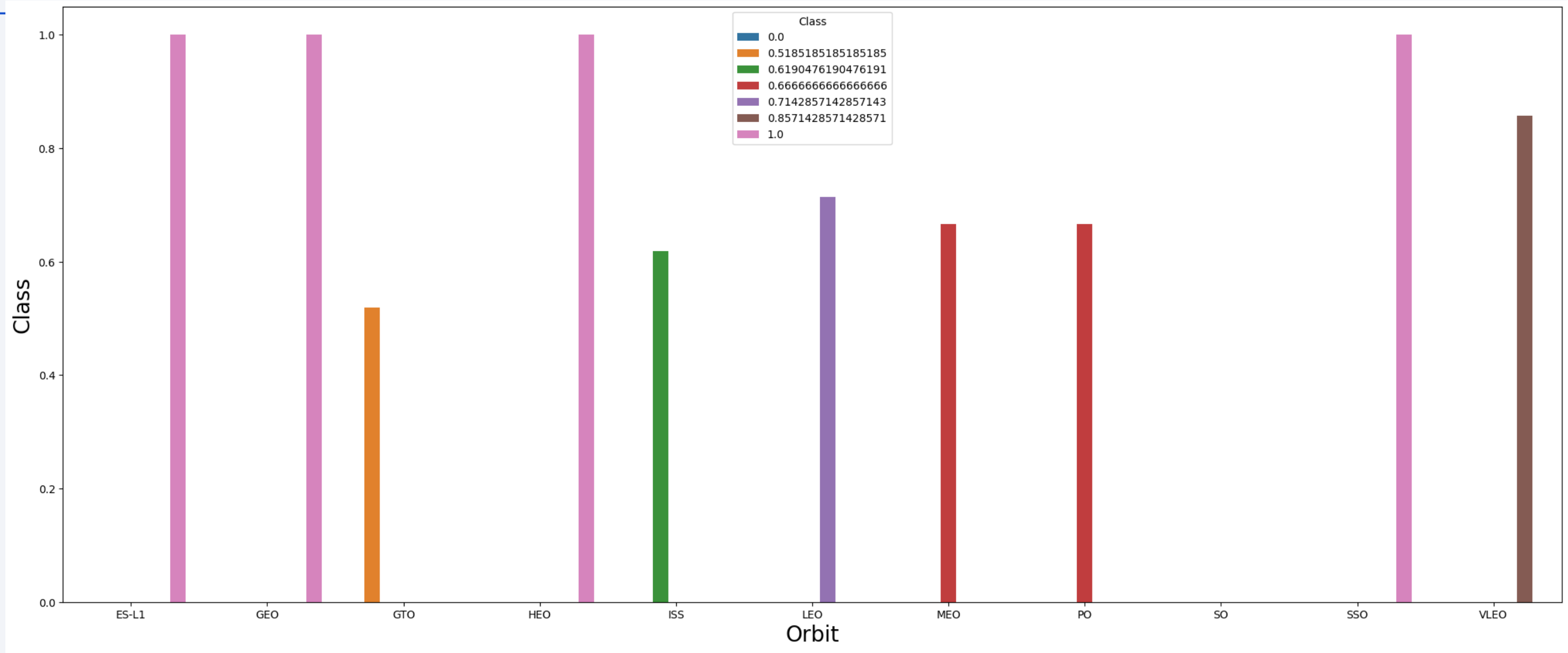
Payload vs. Launch Site



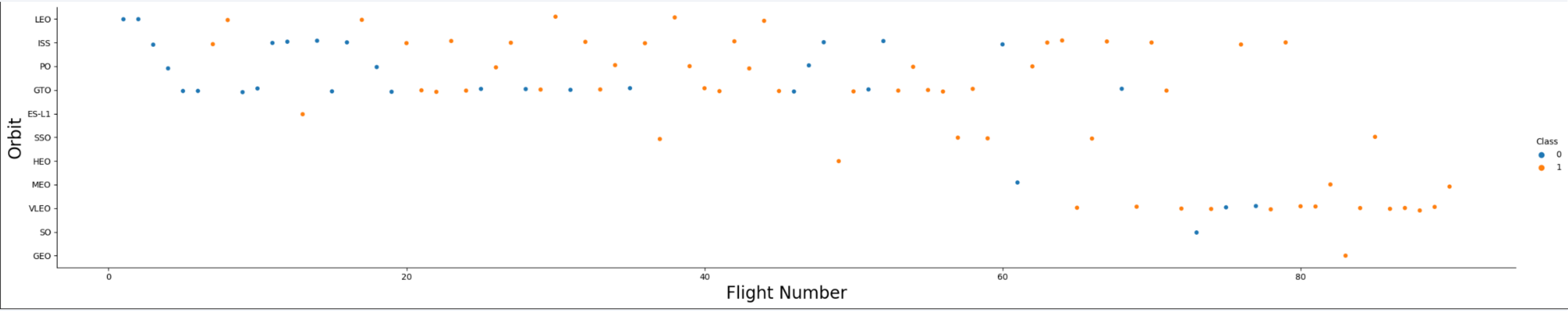
- No higher payload that 10000 from VAFB.
- Succes is seen in clusters

Success Rate vs. Orbit Type

- Orbit ES-LI, GEO, HEO and Sso had the highest succesrate



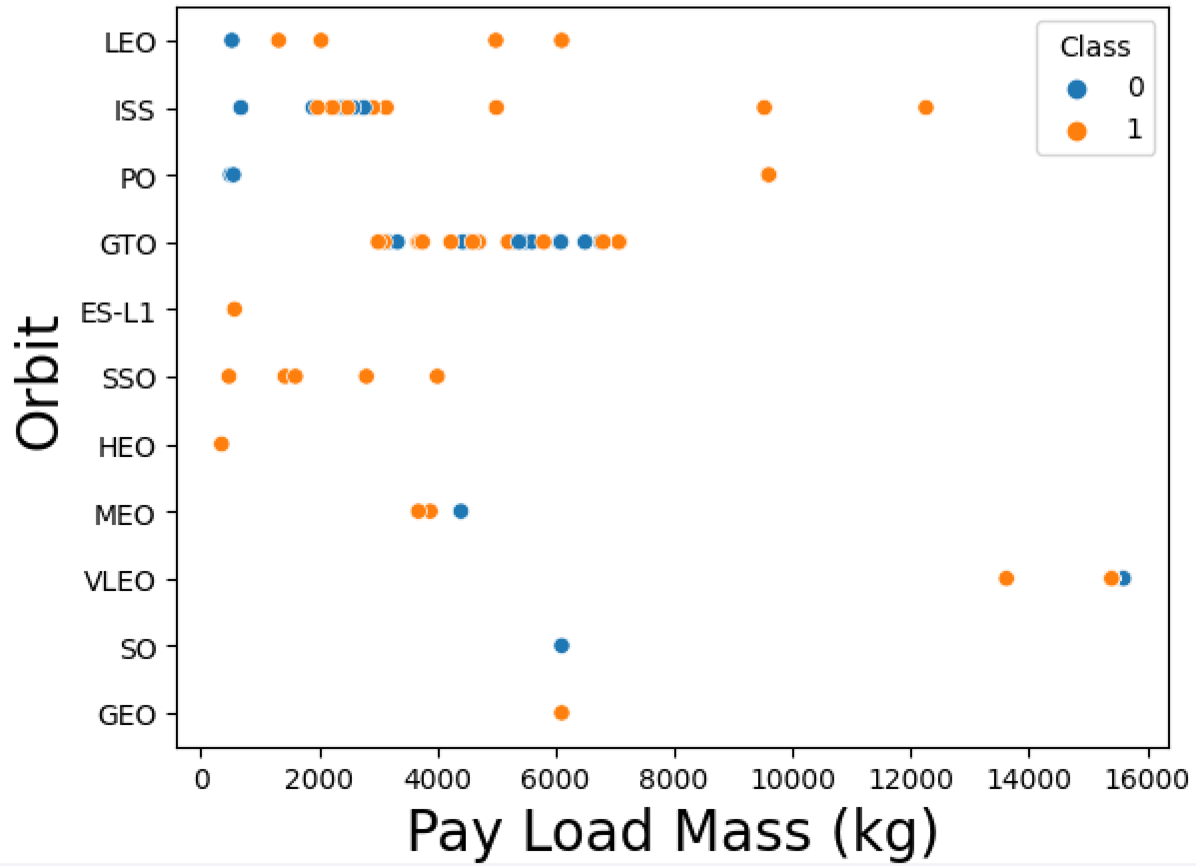
Flight Number vs. Orbit Type



- It's visible that high success rate is not necessary correlated with many flights. HEO and SSO had relative fewer flights

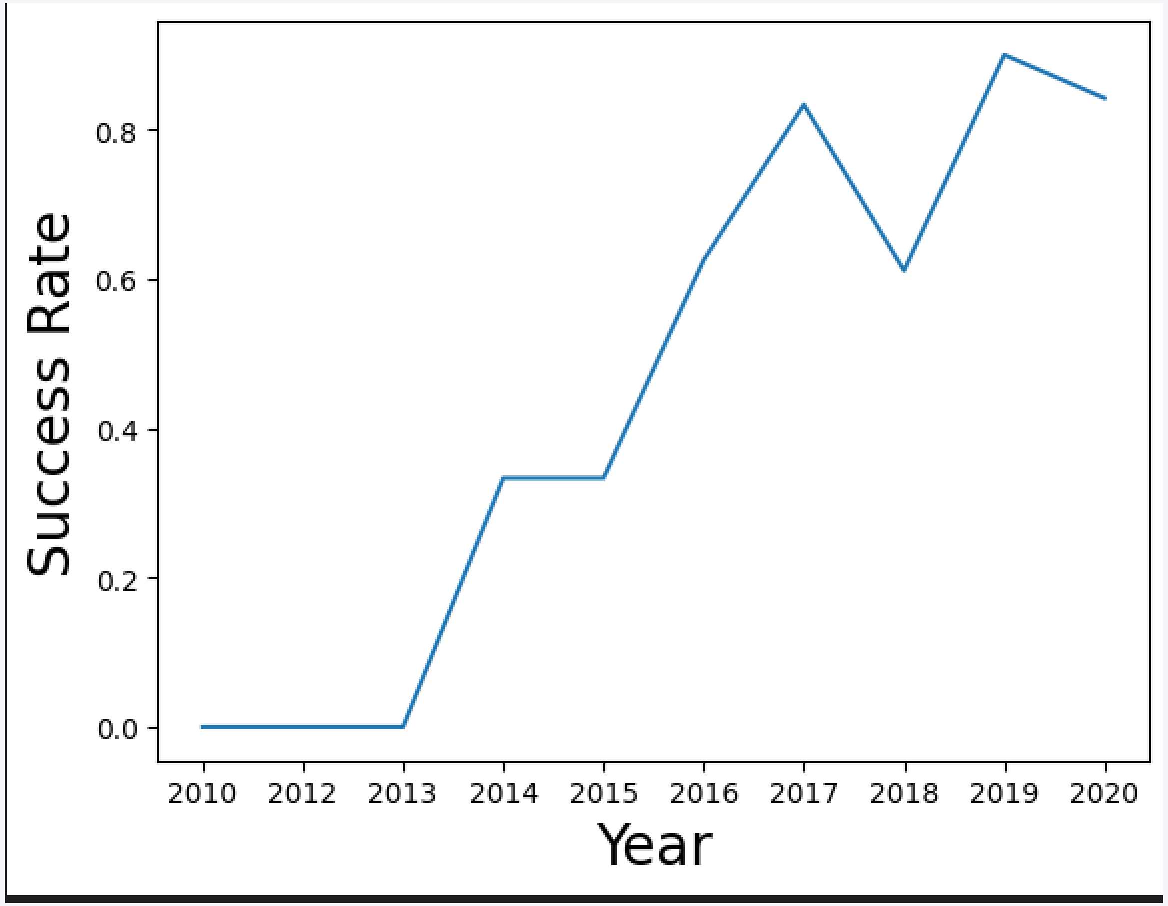
Payload vs. Orbit Type

- GTO has negative influence from higher payload, while LEO, ISS and PO has positive correlation with higher payloads.



Launch Success Yearly Trend

- Succes increases over time



All Launch Site Names

- Used DISTINCT to find unique launch sites

```
%sql SELECT DISTINCT(Launch_Site) from SPACEXTBL
```

```
 * sqlite:///my_data1.db
Done.
```

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

Launch Site Names Begin with 'CCA'

- Using where statement and LIKE to find CCA related instances as well as limit to only show 5.

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

```
%sql SELECT * FROM SPACEXTBL where Launch_Site LIKE 'CCA%' limit 5
```

```
* sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outc
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (paracl
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (paracl
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No att
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No att
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No att

Total Payload Mass

- Using SUM to get the total payload and LIKE to take all NASA flights

Task 3

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

[15]:
```
%sql SELECT sum("PAYLOAD_MASS__KG_") as TotalPayload from SPACEXTBL where Customer LIKE 'NASA%'
```

```
 * sqlite:///my_data1.db
Done.
```

[15]:

TotalPayload
99980.0

Average Payload Mass by F9 v1.1

- Using AVG to find average weight and filter via LIKE

Task 4

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

```
[17]: %sql select avg("PAYLOAD_MASS__KG_") as AvgPayload FROM SPACEXTBL  where Booster_Version LIKE 'F9 v1.1%'
```

```
 * sqlite:///my_data1.db
Done.
```

[17]:

AvgPayload
2534.6666666666665

First Successful Ground Landing Date

- Due to an error in how data is processed then max finds first flight instead of min.

Task 5

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

Hint:Use min function

```
25]: %sql select max(Date) from SPACEXTBL where "Landing_Outcome" = "Success (ground pad)"
```

```
 * sqlite:///my_data1.db
Done.
```

25]:

max(Date)
22/12/2015

```
26]: %sql select min(Date) from SPACEXTBL where "Landing_Outcome" = "Success (ground pad)"
```

```
 * sqlite:///my_data1.db
Done.
```

26]:

min(Date)
01/08/2018

Successful Drone Ship Landing with Payload between 4000 and 6000

- Using between to filter based of payload mass while and statement to filter from success.

Task 6

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

```
%sql select Booster_Version from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000
```

```
 * sqlite:///my_data1.db
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

- Using Count to get the number of each instance from the table, and group by mission_outcome.

Task 7

List the total number of successful and failure mission outcomes

```
%sql select Mission_Outcome, count(Mission_Outcome) as NumberOfMission from SPACEXTBL GROUP BY Mission_Outcome
```

```
 * sqlite:///my_data1.db
Done.
```

Mission_Outcome	NumberOfMission
None	0
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- Listing the Booster versions via distinct to ensure no doubles.
- Subquery to identify max payload mass

Task 8

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

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

```
 * sqlite:///my_data1.db
Done.
```

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

- Listing failed flights for the months in year 2015

```
%sql SELECT substr(Date,4,2) as month, Date , Booster_Version, Launch_Site, Landing_Outcome FROM SPACEXTBL where Landing_Outcome = 'Failure (drone ship)' and substr(Date,7,4)='2015'
```

```
 * sqlite:///my_data1.db
Done.
```

month	Date	Booster_Version	Launch_Site	Landing_Outcome
10	01/10/2015	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	14/04/2015	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

Counted landing outcome and grouped by landing outcome. Ordered it by date

```
%sql SELECT  Date, count(Landing_Outcome), Landing_Outcome from SPACEXTBL WHERE Date BETWEEN '04-06-2010' AND '20-03-2017' group by Landing_Outcome order by Date DESC
```

* sqlite:///my_data1.db
Done.

Date	count(Landing_Outcome)	Landing_Outcome
18/07/2016	7	Success (ground pad)
18/04/2014	2	Controlled (ocean)
14/04/2015	3	Failure (drone ship)
12/05/2018	3	Failure
10/08/2012	10	No attempt
08/07/2018	20	Success
08/06/2019	1	No attempt
06/04/2010	2	Failure (parachute)
04/08/2016	8	Success (drone ship)

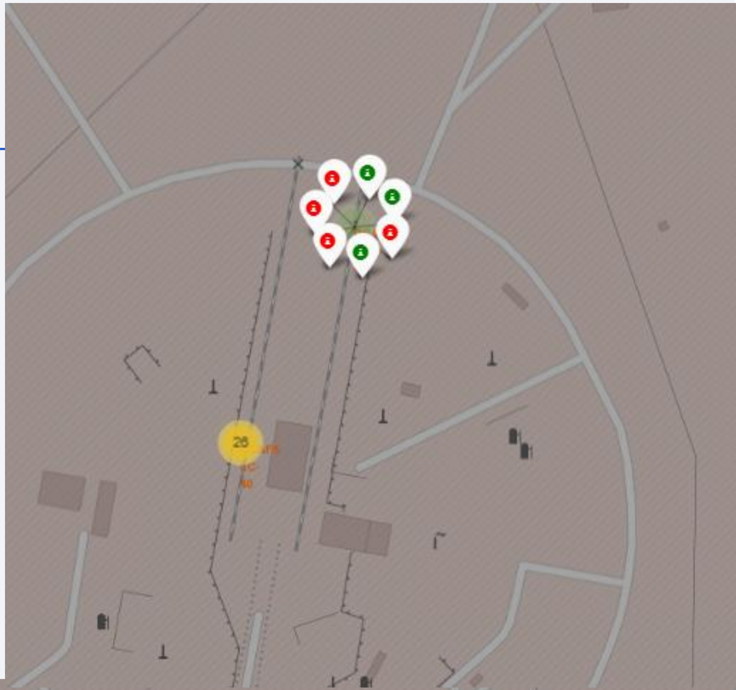
Section 3

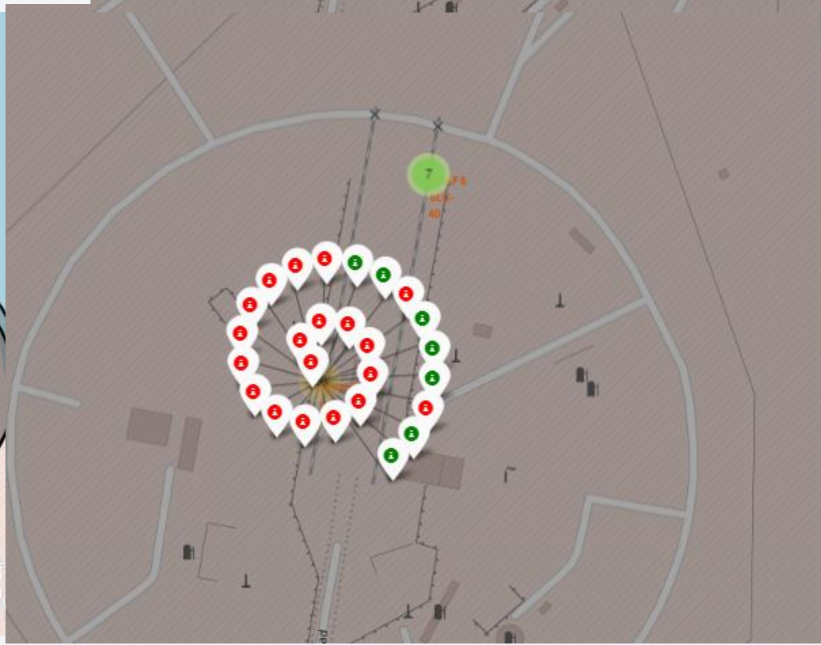
Launch Sites Proximities Analysis

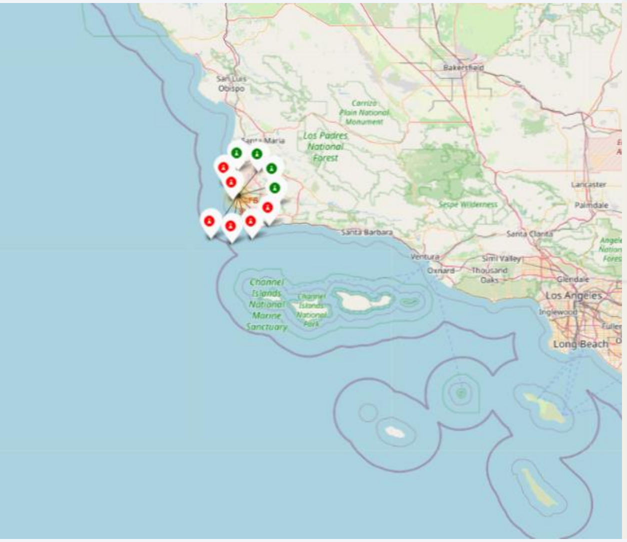
Global launch sites

- All launch sites are located in USA. But on both west and east coast

Succes of launches







Distances

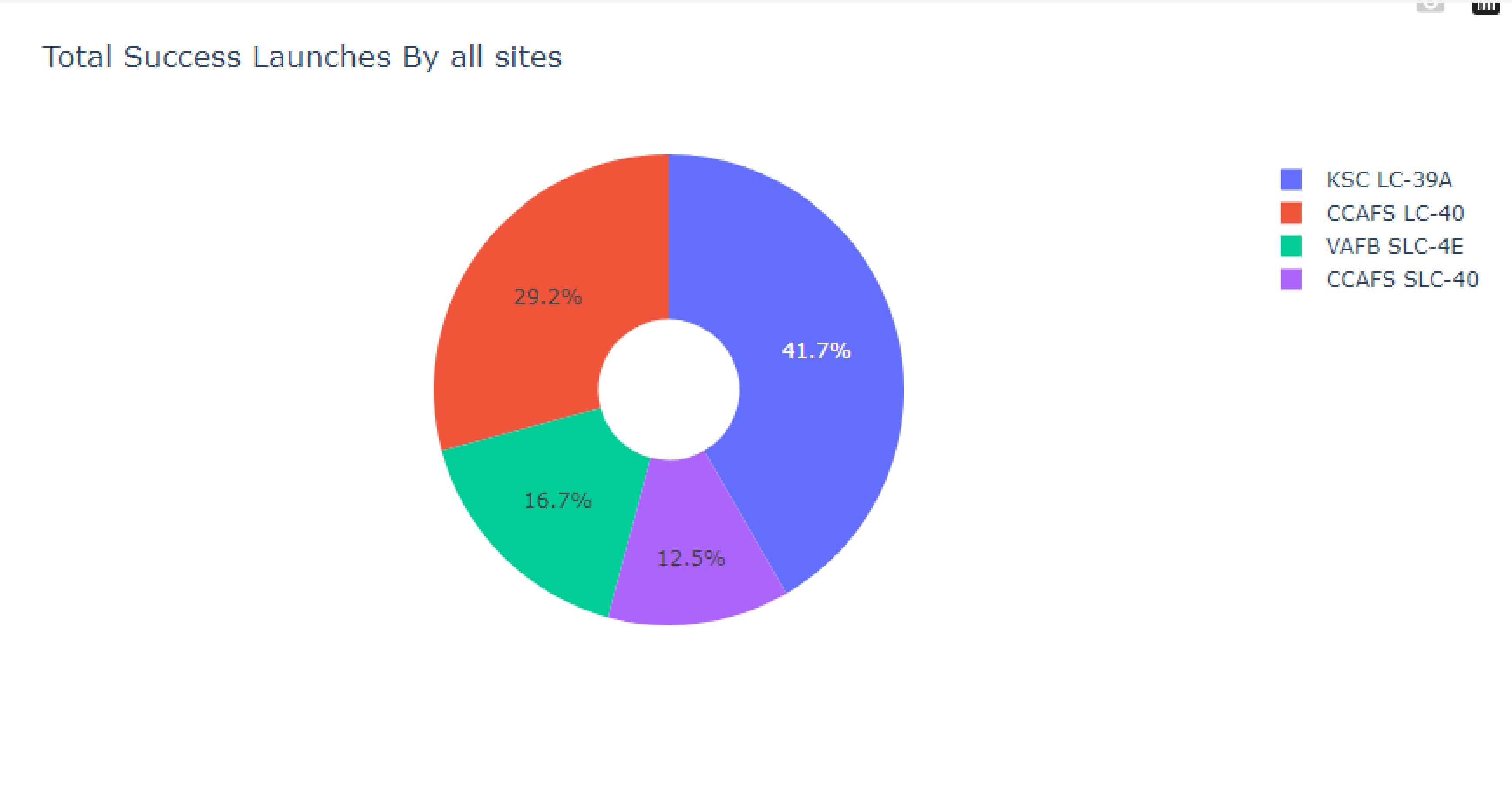
- Close to railways? No
- Close to Highways? No
- Close to coastline? Yes
- Keep certain distance to cities? Yes

Section 4

Build a Dashboard with Plotly Dash

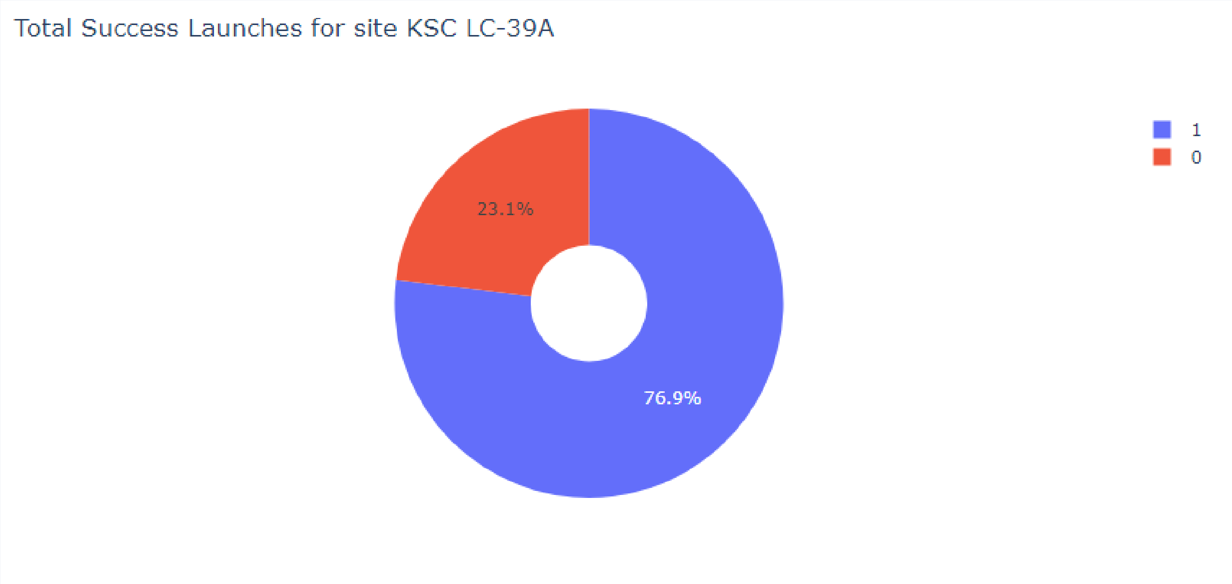
Total Succes

- KSC had the most successful launches from all sites



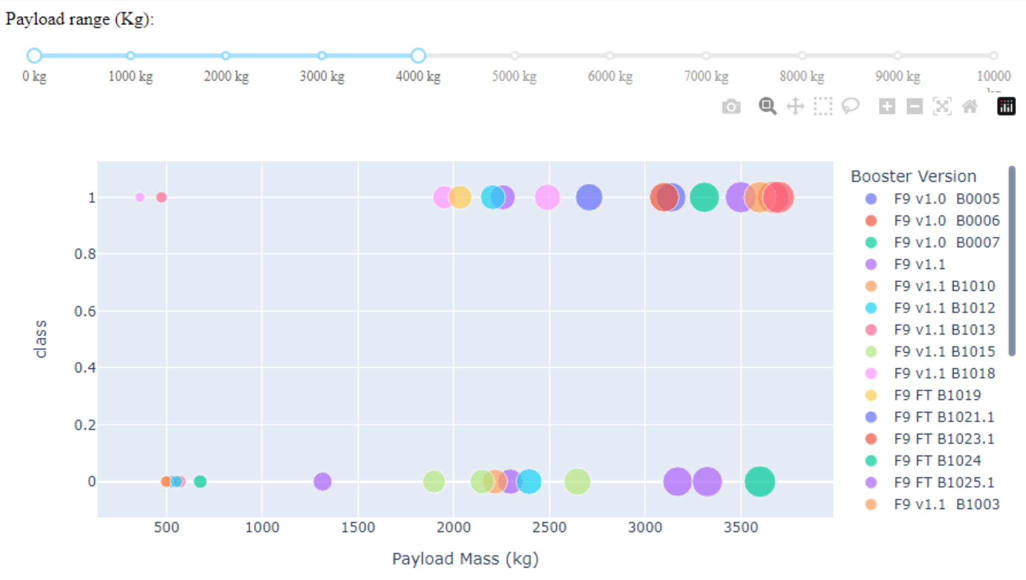
KSC success rate

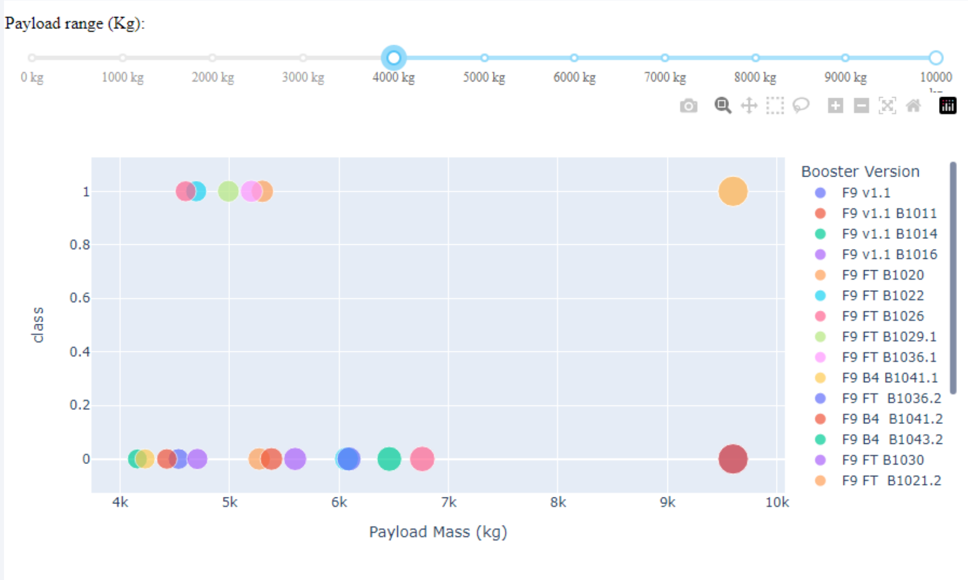
- Its visible that even with most success based from last slide KSC also has highest success rate with 76.9% success rate



Weight matters

- Higher success rate for lower payloads



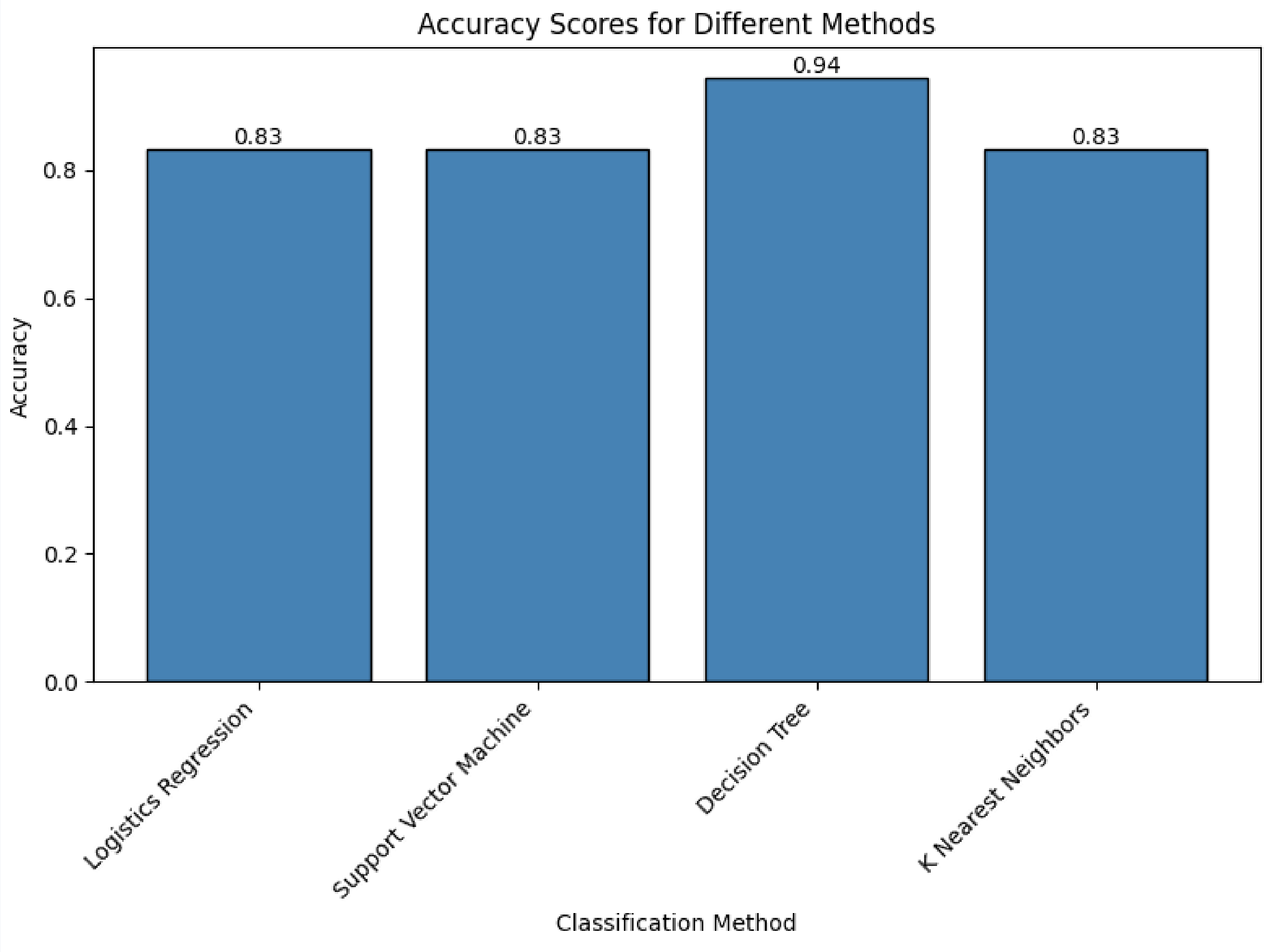


Section 5

Predictive Analysis (Classification)

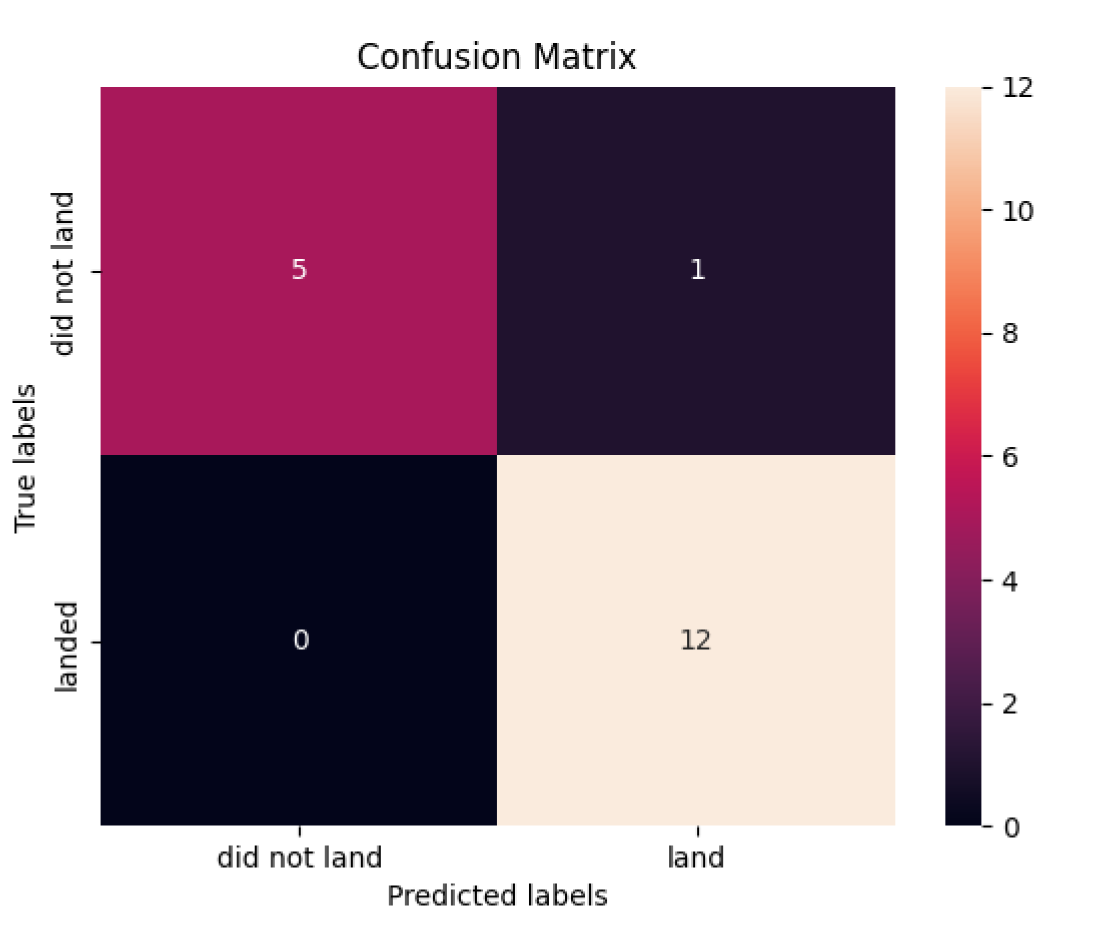
Classification Accuracy

- Decision tree had the highest accuracy on this iteration



Confusion Matrix

- Here we can see the decision tree only has 1 false positive and no false negatives



Conclusions

- The more lauches from a site the higher success rate
- Succes rate increased over time
- Specific orbits had higher success rates namely: ESL1, GEO, HEO, SSO and VLEO
- KSC LC-39A had the most launches as well as highest success rate of any sites
- Lower weight increases success rates
- When trying to model the success then the decision tree is the best classifier in this case for the task.

Thank you!