IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ognjen Kalajdžić
2023/03/01

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
  - SpaceX 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 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. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.
  - This goal of the project is to predict if the first stage will land successfully.
- Problems you want to find answers
  - What factors are most important for the success of a rocket landing?
  - Correlation between factors that influence the success of a rocket landing.
  - What conditions must be met for the rocket to land successfully?

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
    - Building, tuning, evaluating classification models

Data Collection

- How the data sets were collected.
  - 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

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.

- The link to the notebook:

https://github.com/ognjen104/Final-DS-course/blob/master/Data%20collection%20API.ipynb

Let's request the data from SPaceX API with the following URL:

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

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

In [8]:
```
#You can print the response content
#print(response.content)
```

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

In [10]:
```
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = response.json()
```

In [11]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

In [12]:
```
# Let's check
data.head(5)
```

Data Collection - Scraping

- We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup. Then, we parsed the table and converted it into a pandas dataframe.

- The link to the notebook:

https://github.com/ognjen104/Final-DS-course/blob/master/Data%20collection%20with%20Web%20Scraping.ipynb

In [4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Let's 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
html_data = requests.get(static_url)
html_data.status_code
```

Out[5]:
```
200
```

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text, 'html.parser')
```

In [7]:
```
# Use soup.title attribute
soup.title
```

Out[7]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

Let's extract all column/variable names from the HTML table header

In [8]:
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

In [9]:
```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

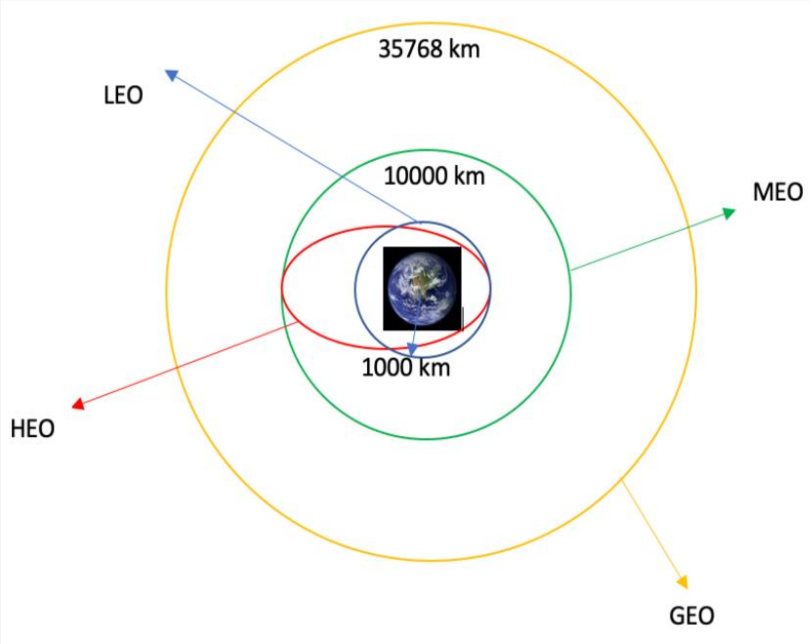
In [10]:
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

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

Data Wrangling

- We made the data more accessible and easier to analyze through the three steps:
  - 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.

- The link to the notebook:

https://github.com/ognjen104/Final-DS-course/blob/master/Data%20wrangling.ipynb

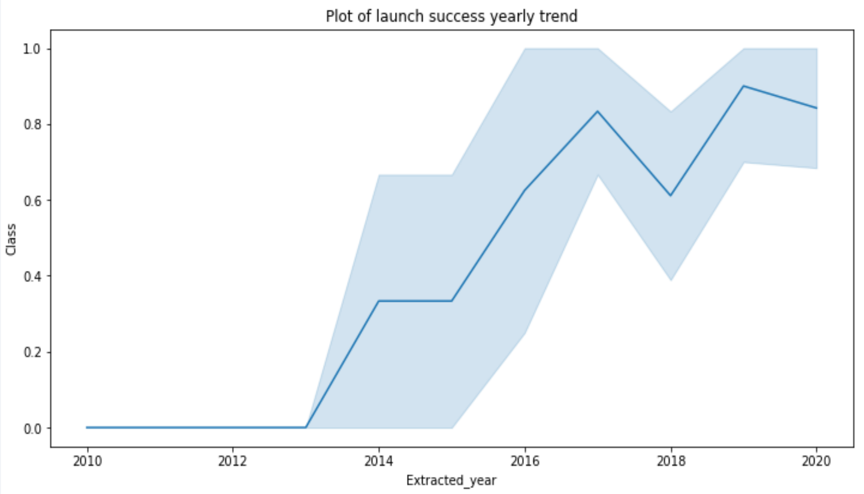


EDA with Data Visualization

- We have visualized 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.

- The link to the notebook:

https://github.com/ognjen104/Final-DS-course/blob/master/EDA%20with%20Data%20Visualization.ipynb



EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the Jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook:

https://github.com/ognjen104/Final-DS-course/blob/master/EDA%20with%20SQL.ipynb

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.
  - 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.

- The link to the notebook:

  https://github.com/ognjen104/Final-DS-course/blob/master/Interactive%20Visual%20Analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

- The link to the notebook:

  https://github.com/ognjen104/Final-DS-course/blob/master/spacex_dash_app.py

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.

- The link to the notebook:

  https://github.com/ognjen104/Final-DS-course/blob/master/Machine%20Learning%20Prediction.ipynb

Results

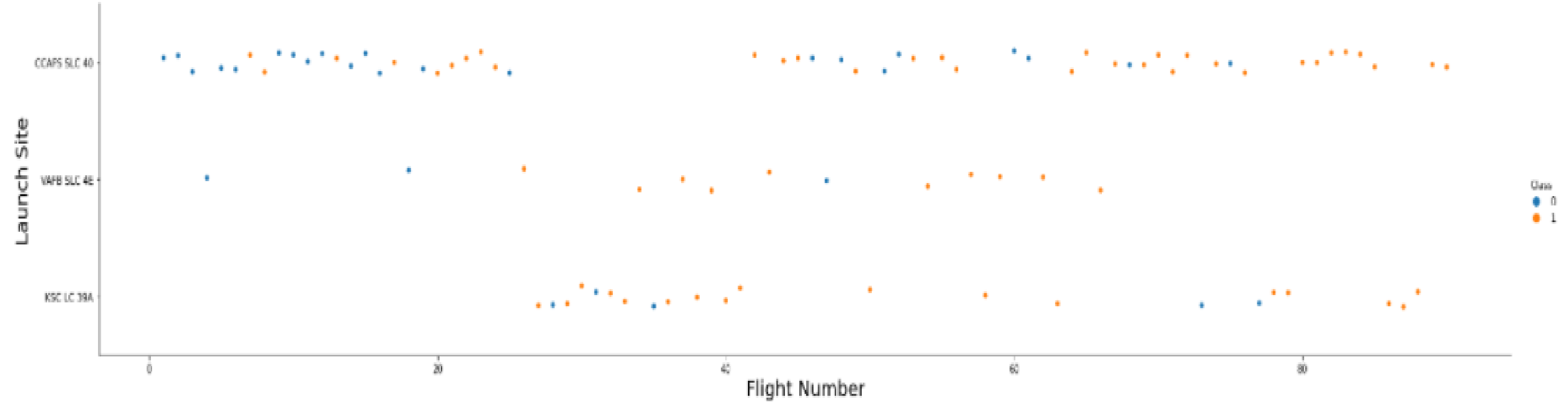
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

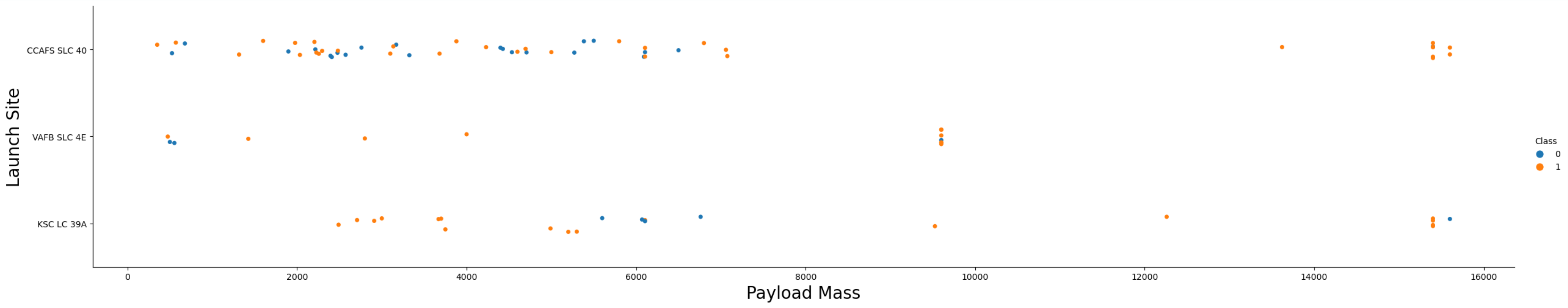
- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.



Payload vs. Launch Site

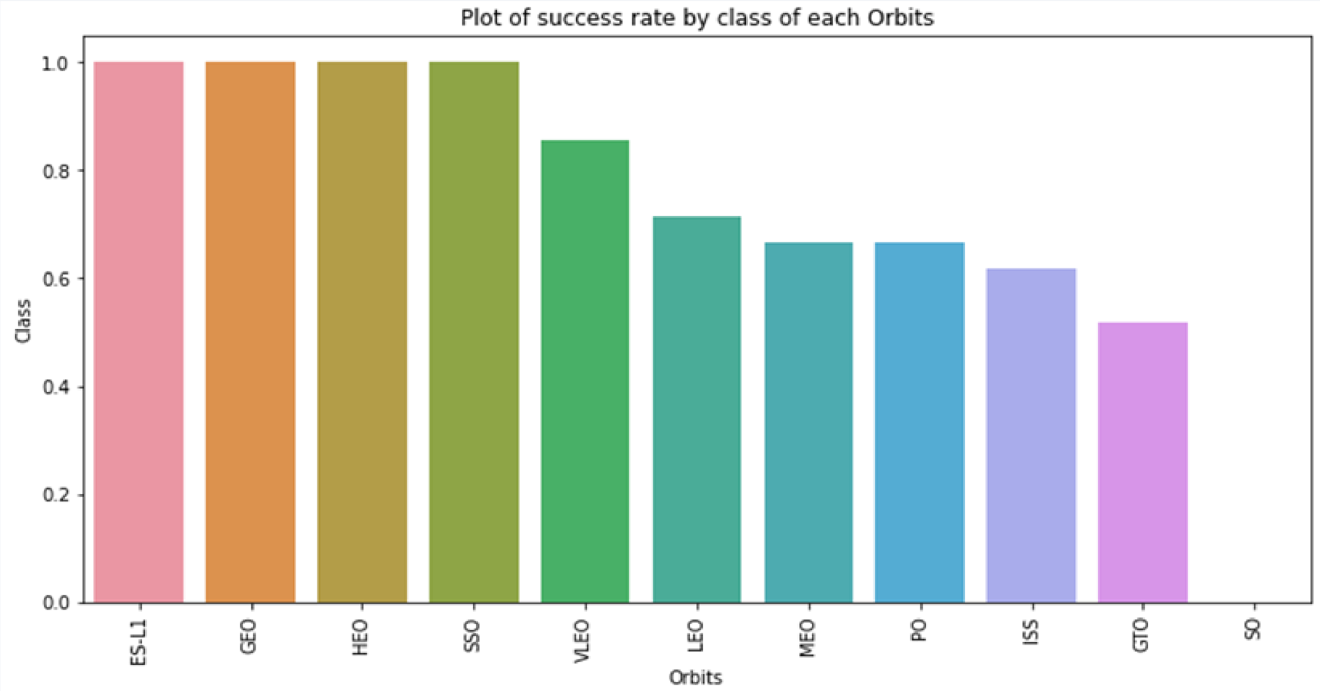
- Show a scatter plot of Payload vs. Launch Site

- Show the screenshot of the scatter plot with explanations



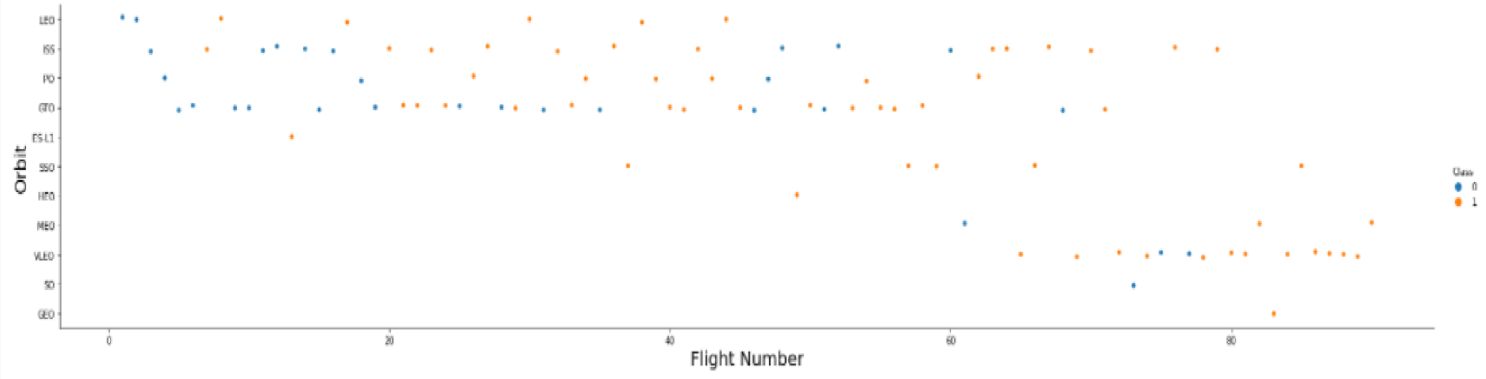
Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



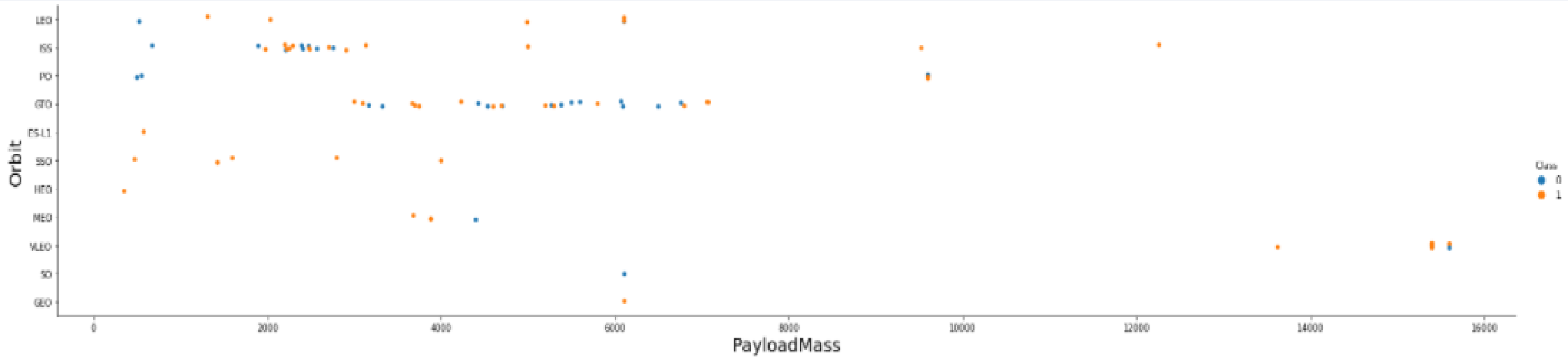
Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



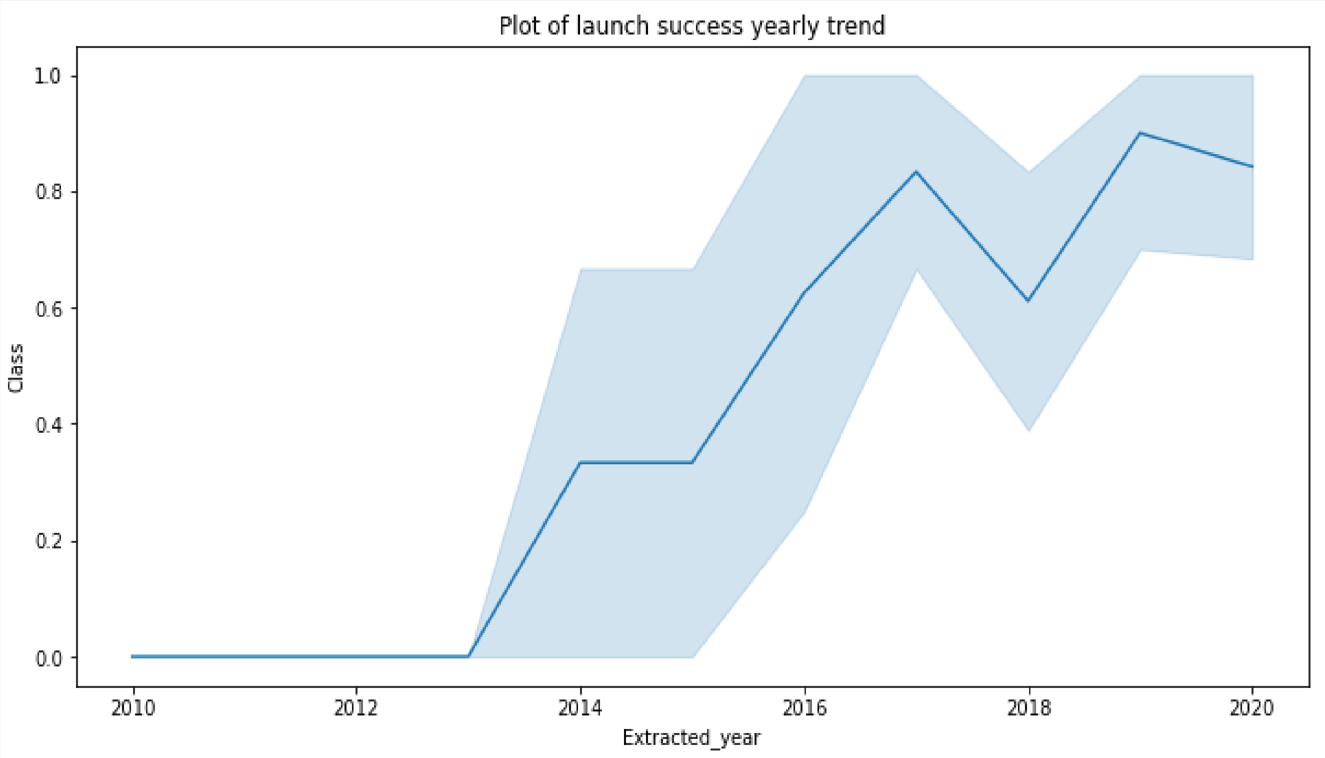
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits



Launch Success Yearly Trend

- From the plot, we can observe that success rate since 2013 kept on increasing till 2020



All Launch Site Names

- We used the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

```
In [4]: %sql select distinct launch_site from SPACEXDATASET;

 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
Out[4]:
```

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

Launch Site Names Begin with 'CCA'

- We used the query above to display 5 records where launch sites begin with `CCA`

```
In [5]: %sql select * from SPACEXDATASET where launch_site like 'CCA%' limit 5;

 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/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

Total Payload Mass

- We calculated the total payload carried by boosters from NASA as 45596 using the query below.

```
In [6]: %sql select sum(payload_mass__kg_) as total_payload_mass from SPACEXDATASET where customer = 'NASA (CRS)';

 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
Out[6]:
```

total_payload_mass
45596

Average Payload Mass by F9 v1.1

- We calculated the average payload mass carried by booster version F9 v1.1 as 2928.4

```
In [7]: %sql select avg(payload_mass__kg_) as average_payload_mass from SPACEXDATASET where booster_version like '%F9 v1.1%';
 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
Out[7]:
```

average_payload_mass
2534

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

```
In [8]: %sql select min(date) as first_successful_landing from SPACEXDATASET where landing__outcome = 'Success (ground pad)';

 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90108kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
Out[8]:
```

first_successful_landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

```
In [9]: %sql select booster_version from SPACEXDATASET where landing__outcome = 'Success (drone ship)' and payload_mass__kg_ between 4
000 and 6000;

 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
Out[9]:
```

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

Total Number of Successful and Failure Mission Outcomes

- We used GROUP BY and COUNT to check how many MissionOutcome was a success or a failure

```
In [10]: %sql select mission_outcome, count(*) as total_number from SPACEXDATASET group by mission_outcome;

 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
Out[10]:
```

mission_outcome	total_number
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

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

```
In [11]: %sql select booster_version from SPACEXDATASET where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXDATASET);

 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
Out[11]:
```

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

- We used a combinations of the **WHERE** clause and **YEAR ()** function to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

```
In [12]: %%sql select monthname(date) as month, date, booster_version, launch_site, landing__outcome from SPACEXDATASET
             where landing__outcome = 'Failure (drone ship)' and year(date)=2015;

 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[12]:

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

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

- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

```
In [13]: %%sql select landing__outcome, count(*) as count_outcomes from SPACEXDATASET
                 where date between '2010-06-04' and '2017-03-20'
                 group by landing__outcome
                 order by count_outcomes desc;

          * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
         Done.
Out[13]:
```

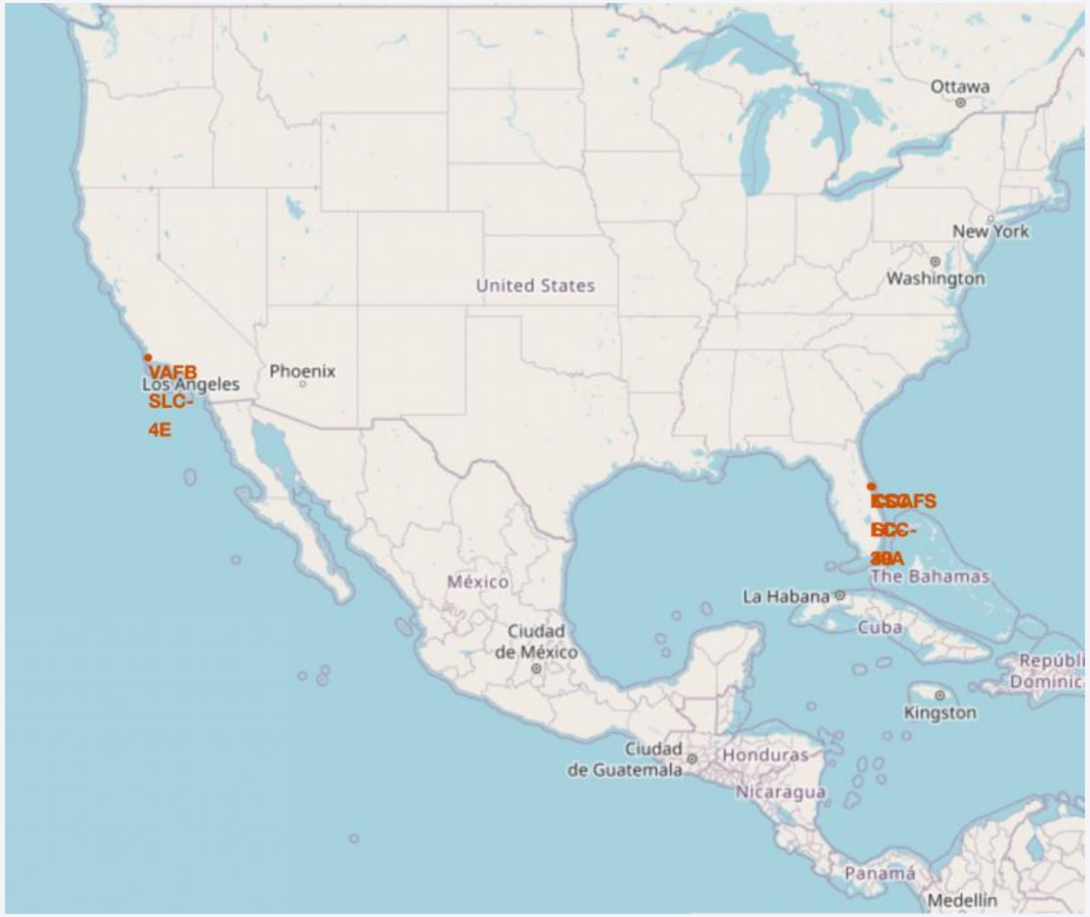
landing__outcome	count_outcomes
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

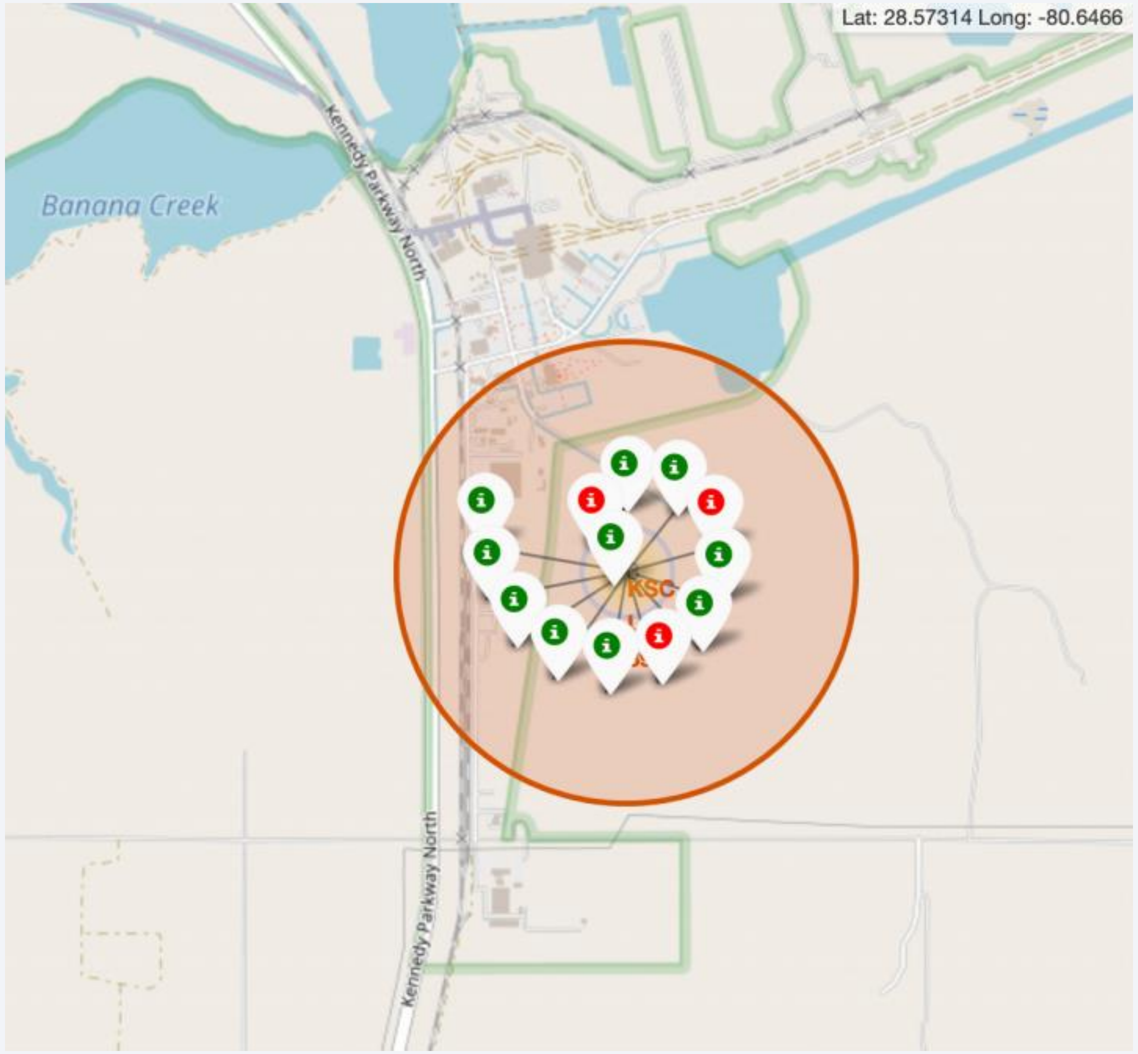
All launch sites global map markers

- We can see that the SpaceX launch sites are in USA coasts Florida and California



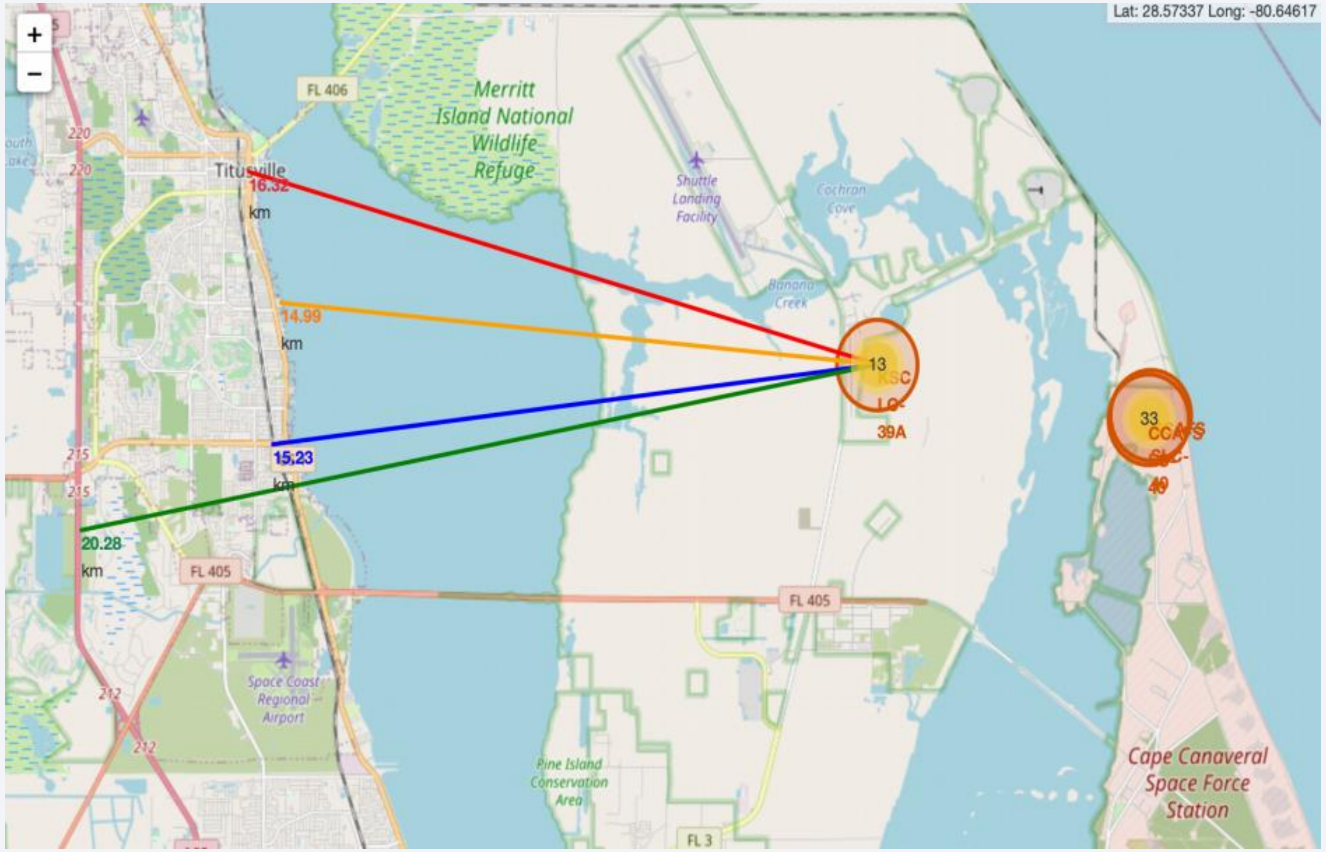
Markers showing launch sites with color labels

- From the colour-labeled markers we should be able to easily identify which launch sites have relatively high success rates.
  - Green Marker = Successful Launch
  - Red Marker = Failed Launch
- Launch Site KSC LC-39A has a very high Success Rate.



Launch Site distance to landmarks

From the visual analysis of the launch site KSC LC-39A we can clearly see that it is:
- relative close to railway (15.23 km)
- relative close to highway (20.28 km)
- relative close to coastline (14.99 km)
• Also the launch site KSC LC-39A is relative close to its closest city Titusville (16.32 km).
• Failed rocket with its high speed can cover distances like 15-20 km in few seconds. It could be potentially dangerous to populated areas.

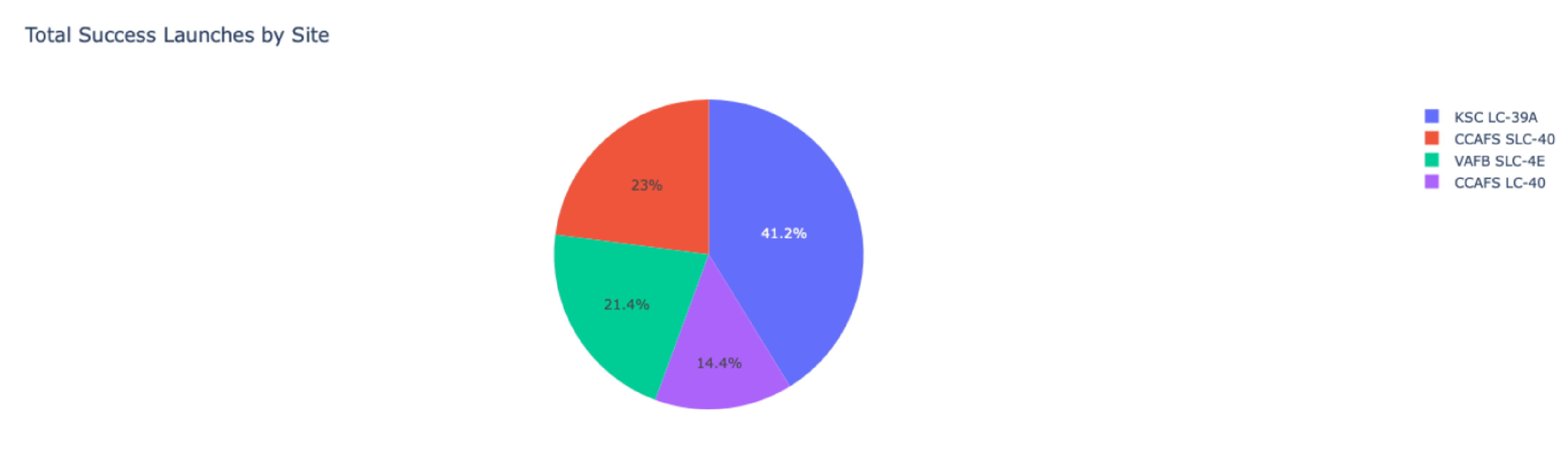


Section 4

Build a Dashboard with Plotly Dash

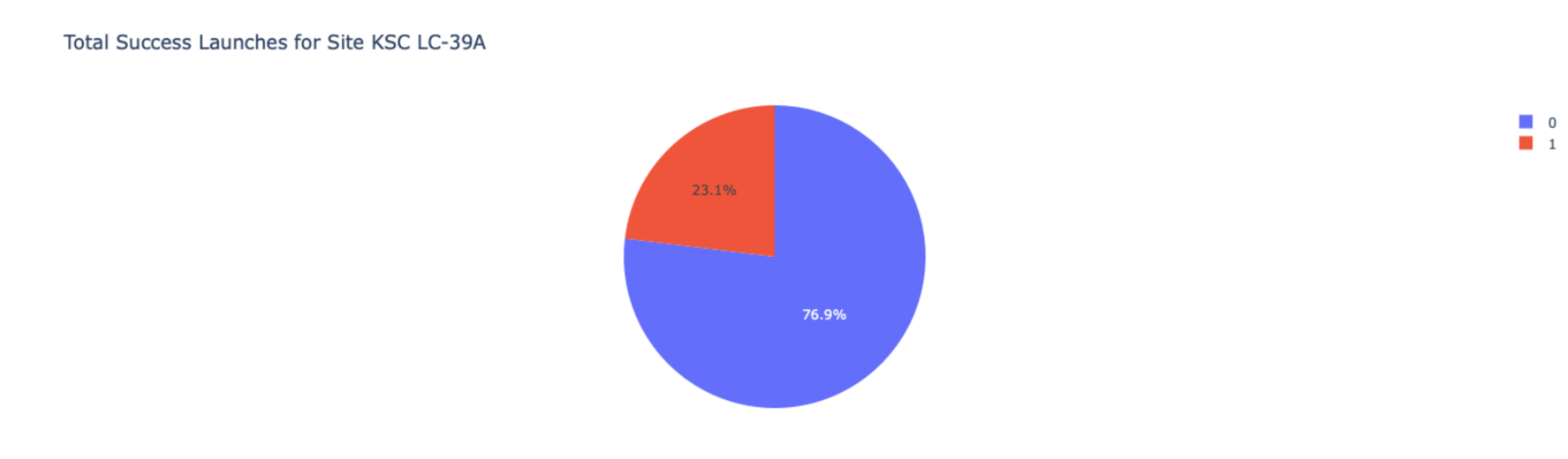
Pie chart showing the success percentage achieved by each launch site

- The chart clearly shows that from all the sites, KSC LC-39A has the most successful launches.



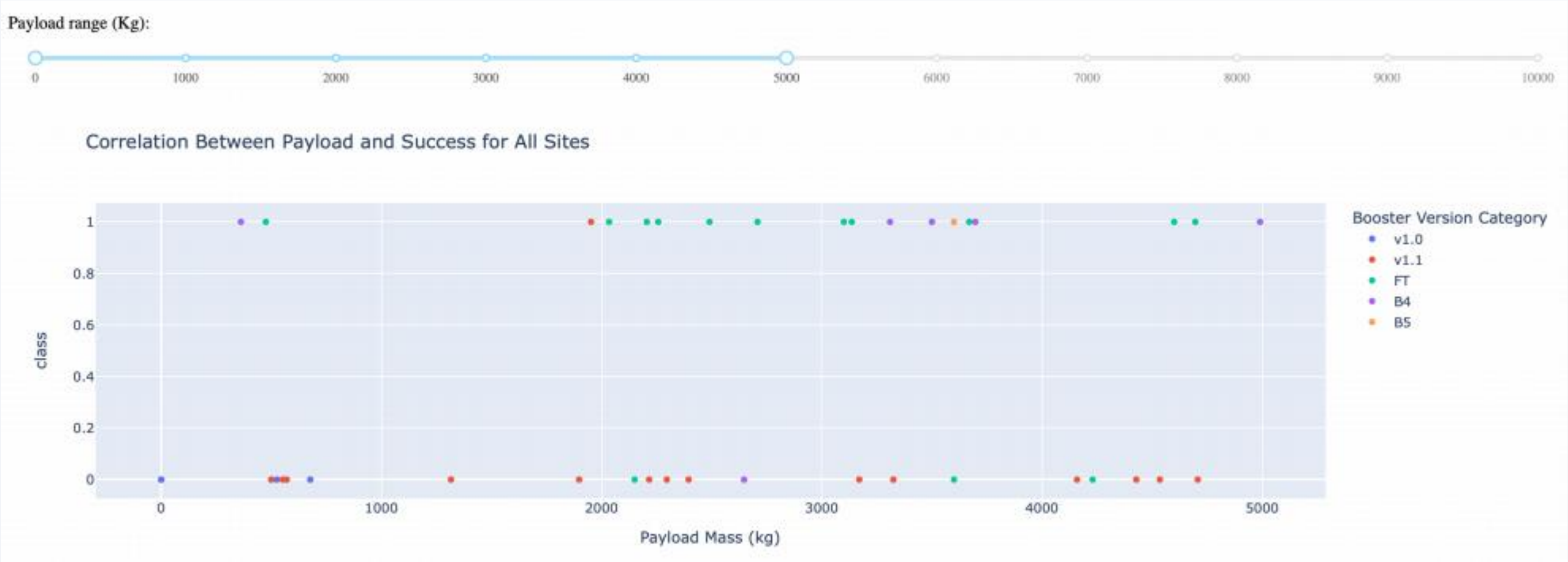
Pie chart showing the Launch site with the highest launch success ratio

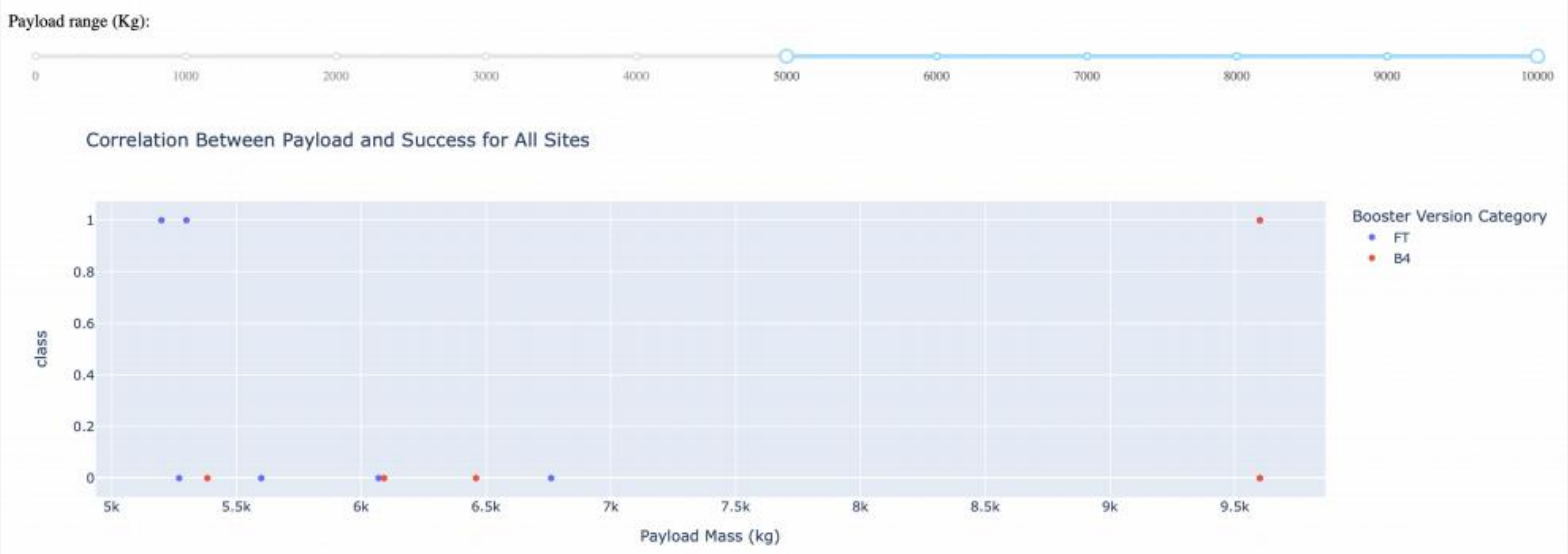
- KSC LC-39A has the highest launch success rate (76.9%) with 10 successful and only 3 failed landings.



Scatter plot of Payload vs Launch Outcome for all sites, with different payload in the range slider

- The charts show that payloads between 2000 and 5500 kg have the highest success rate.





Section 5

Predictive Analysis (Classification)

Classification Accuracy

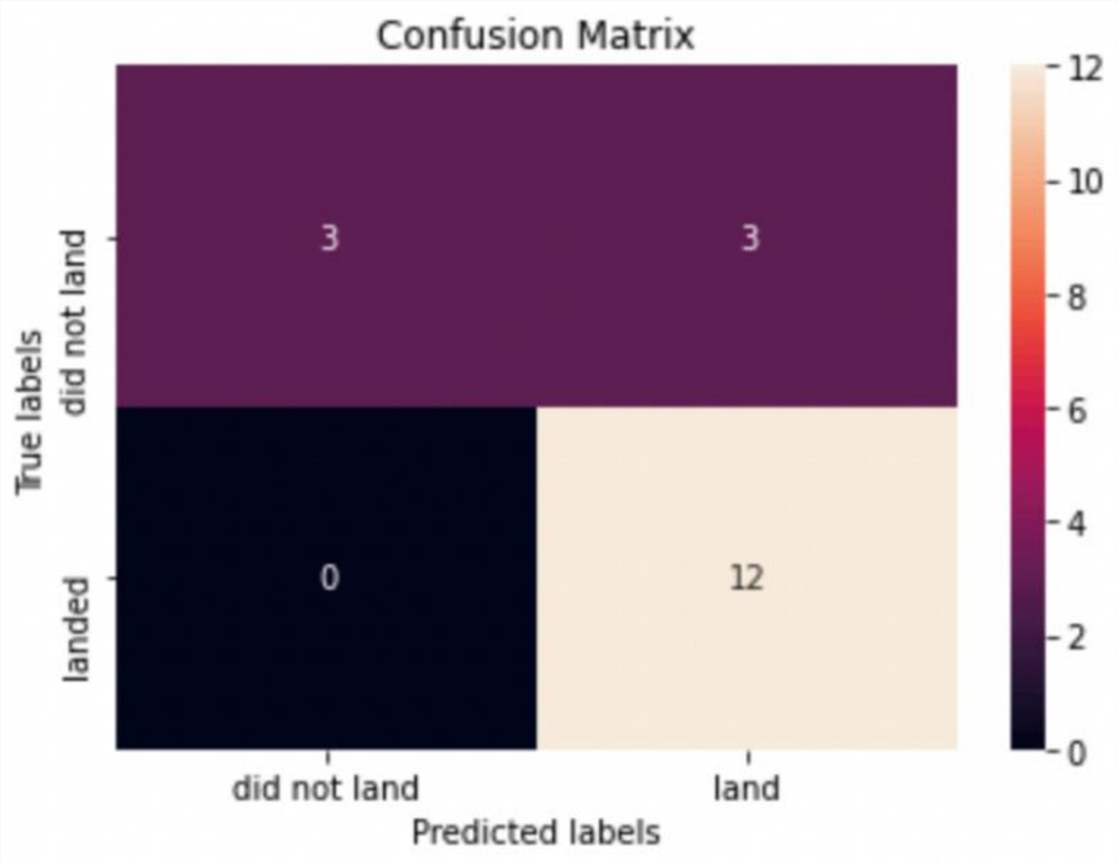
- Based on the scores of the Test Set, we can not confirm which method performs best.
- Same Test Set scores may be due to the small test sample size (18 samples). Therefore, we tested all methods based on the whole Dataset.
- The scores of the whole Dataset confirm that the best model is the Decision Tree Model. This model has not only higher scores, but also the highest accuracy.

	LogReg	SVM	Tree	KNN
Jaccard_Score	0.800000	0.800000	0.800000	0.800000
F1_Score	0.888889	0.888889	0.888889	0.888889
Accuracy	0.833333	0.833333	0.833333	0.833333

	LogReg	SVM	Tree	KNN
Jaccard_Score	0.833333	0.845070	0.882353	0.819444
F1_Score	0.909091	0.916031	0.937500	0.900763
Accuracy	0.866667	0.877778	0.911111	0.855556

Confusion Matrix

- Examining the confusion matrix, we see that logistic regression can distinguish between the different classes. We see that the major problem is false positives.



Conclusions

We can conclude that:

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Appendix

- Special Thanks to:
  Instructors
  Coursera
  IBM

Thank you!