IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Dhananjay Roy
10-11-2021

[GitHub Repository]

Outline

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

Executive Summary

- For this entire exercise, data was collected using SpaceX REST API and web scraping Wikipedia pages.
- In the dataset there were several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident. We converted those non numeric outcomes into training labels with 1 means the booster successfully landed and 0 means it was unsuccessful.
- We then performed, exploratory data analysis using visualization and SQL and interactive visual analysis using Folium and Plotly Dash.
- Different Machine learning models built and tested for accuracy. The model with best accuracy was then selected.

Introduction

- 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.
- By utilizing Data Science tools and Machine Learning, we are trying to predict if the first stage of Falcon 9 will land successfully. If we can determine if the first stage will land, we can determine the cost of a launch.
- We needed the relationships between different rocket variables and the launch outcome explored for us to build a model that performs the best.
- Geography of the different launch sites and its effect on the launch outcome needed exploration.

Section 1

Methodology

Methodology

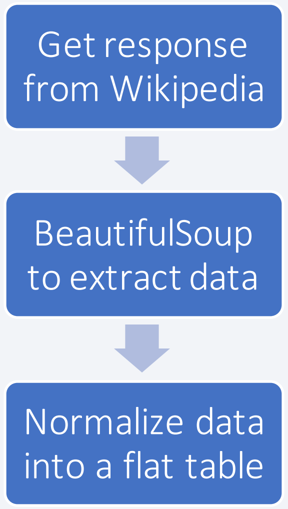
Executive Summary

- Data collection methodology:
  - Data for this exercise was collected using the SpaceX Rest API and Web Scraping.
- Perform data wrangling
  - We performed One Hot Encoding on the required columns and removed some irrelevant columns from the dataset.
- 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

- We collected the required data using the SpaceX REST API and scraping Falcon 9 Wikipedia page.
- The SpaceX API gives us data about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcomes.
- The SpaceX API endpoint starts with api.spacexdata.com/v4/.
- We used the api.spacexdata.com/v4/launches/past.
- Another method used for data collection is Web Scraping SpaceX Falcon 9 Wikipedia page using BeautifulSoup.

Data Collection using API



Data Collection using BeautifulSoup

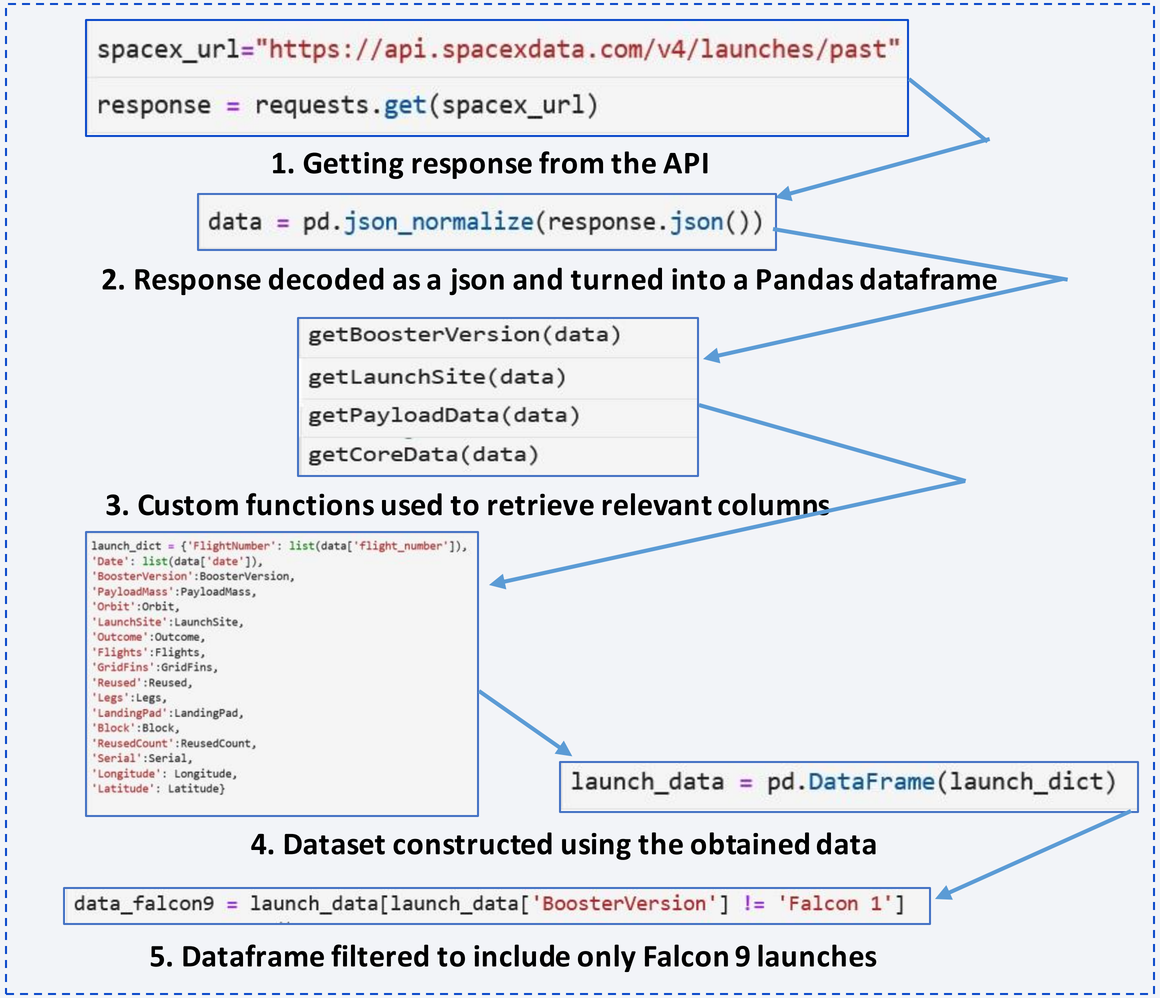
Data Collection – SpaceX API

- We requested rocket launch data from SpaceX REST API with the following URL:

  https://api.spacexdata.com/v4/launches/past

- The json response received was then turned into a Pandas dataframe.
- Relevant columns were retrieved using custom functions.
- We then filtered the dataframe to only include Falcon 9 launches.

[GitHub URL to Notebook]()



Data Collection - Scraping

- We performed an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.
- A BeautifulSoup object was created from the HTML response.
- We collected all relevant column names from the HTML table header.
- Finally, we created a data frame by parsing the launch HTML tables.

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

1. Getting response from the HTML

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

2. Creating a BeautifulSoup object

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

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

3. All relevant column names extracted

```
launch_dict= dict.fromkeys(column_names)

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

# Let's initial the launch_dict with each value to be an empty list
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']=[]
```

4. Dictionary with keys from the extracted column names created

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

5. The dictionary is converted in to Pandas dataframe

GitHub URL to Notebook

Data Wrangling

- We performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.
- In the data set, there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, **`True Ocean`** means the mission outcome was successfully landed to a specific region of the ocean while **`False Ocean`** means the mission outcome was unsuccessfully landed to a specific region of the ocean.
- **`True RTLS`** means the mission outcome was successfully landed to a ground pad **`False RTLS`** means the mission outcome was unsuccessfully landed to a ground pad. **`True ASDS`** means the mission outcome was successfully landed on a drone ship **`False ASDS`** means the mission outcome was unsuccessfully landed on a drone ship.
- We mainly converted those outcomes into Training Labels with `1` means the booster successfully landed `0` means it was unsuccessful.

GitHub URL to Notebook

EDA with Data Visualization

- We used scatter plots to visualize relationships between different variables. Flight Number vs Payload Mass, Flight Number vs Launch Site, Payload vs Launch Site, Flight Number vs Orbit type, and Payload vs Orbit type scatter plots were used to determine their corelation.
- We crerated bar chart to visually check if there are any relationship between Success rate and Orbit type.
- Finally, a line chart was plotted to get the average launch success trend.

[GitHub URL to Notebook]

EDA with SQL

To gain a complete understanding of the dataset, following SQL queries were executed:

- Displayed the names of the unique launch sites in the space mission
- Displayed 5 records where launch sites begin with the string 'CCA'
- Displayed the total payload mass carried by boosters launched by NASA (CRS)
- Displayed average payload mass carried by booster version F9 v1.1
- Listed the date when the first successful landing outcome in ground pad was acheived.
- Listed the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- Listed the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- Listed the 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

GitHub URL to Notebook

Build an Interactive Map with Folium

- The launch success rate may depend on the location and proximities of a launch site, i.e., the initial position of rocket trajectories.
- Finding an optimal location for building a launch site certainly involves many factors. We tried analyzing the existing launch site locations to hopefully discover some of the factors.
- First, we added a folium Circle for each launch site and then added a folium marker for each launch site on the map to visualize these locations.
- We created markers for all launch records. If a launch was successful (class=1), then we use a green marker and if a launch was failed, we use a red marker (class=0).
- To explore a launch site's proximity from railway, highway and coastline etc., we first added a MousePosition on the map to get coordinate for a mouse over a point on the map. Drew a line and put a marker on the map to show its distance.

[GitHub URL to Notebook]()

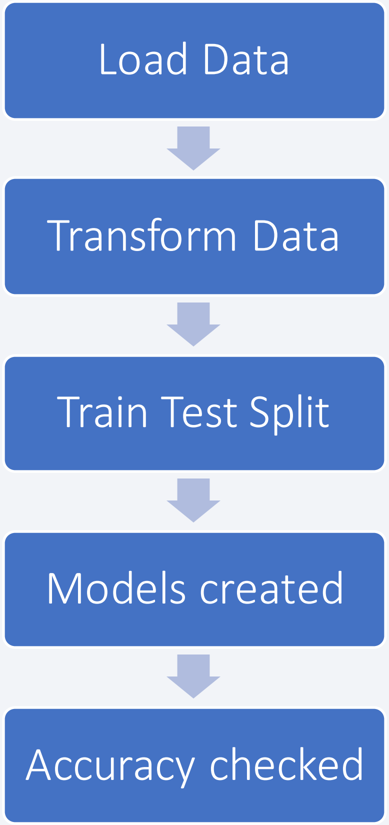
Build a Dashboard with Plotly Dash

- The dashboard application contains input components such as a **dropdown** list and a **range slider** to interact with a **pie chart** and a **scatter plot** chart.
- The **dropdown** menu was added to let us select different launch sites.
- We added a **range slider** as we wanted to be able to easily select different payload range and see if we can identify some visual patterns regarding the correlation between payload and mission outcomes.
- With the help of the pie chart and scatter plot chart we were able to answer questions like which launch site has the largest successful launches,, which site has the highest launch success rate, which payload range has the highest and lowest launch success rates etc.

GitHub URL to Notebook

Predictive Analysis (Classification)

- First the dataframe was loaded and then we created a NumPy array from the column 'Class'.
- Standardized the data using transform.
- We split the data into training and testing data sets.
- Different Machine Learning objects and then a GridSearchCV object created.
- We then fitted the object to find the best parameters.
- Accuracy of the each model calculated using the method **score** and then we plotted the confusion matrix.
- The model with the best accuracy score then selected.



GitHub URL to Notebook

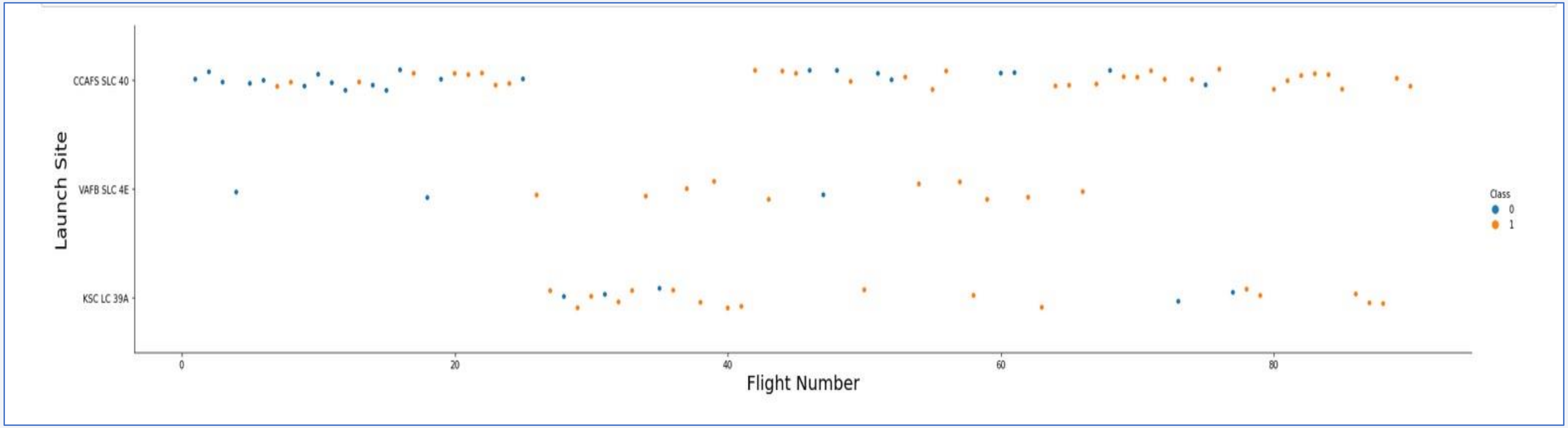
Results

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

Section 2

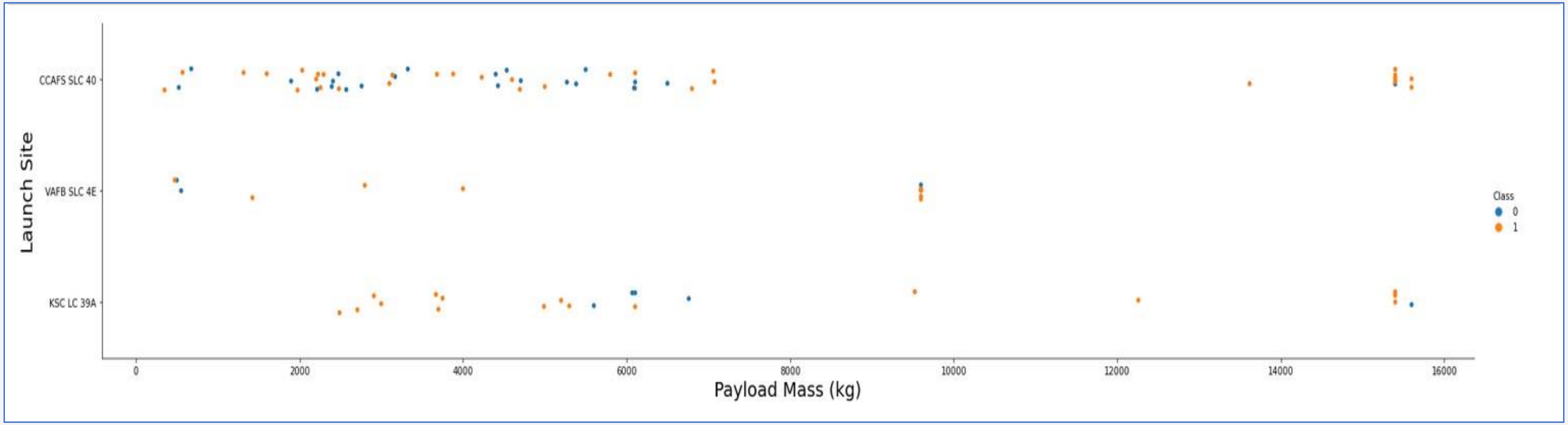
Insights drawn from EDA

Flight Number vs. Launch Site



- From the above scatter plot, it is clear that, as the number of flight increases from a particular launch site, the first stage is more likely to land successfully.

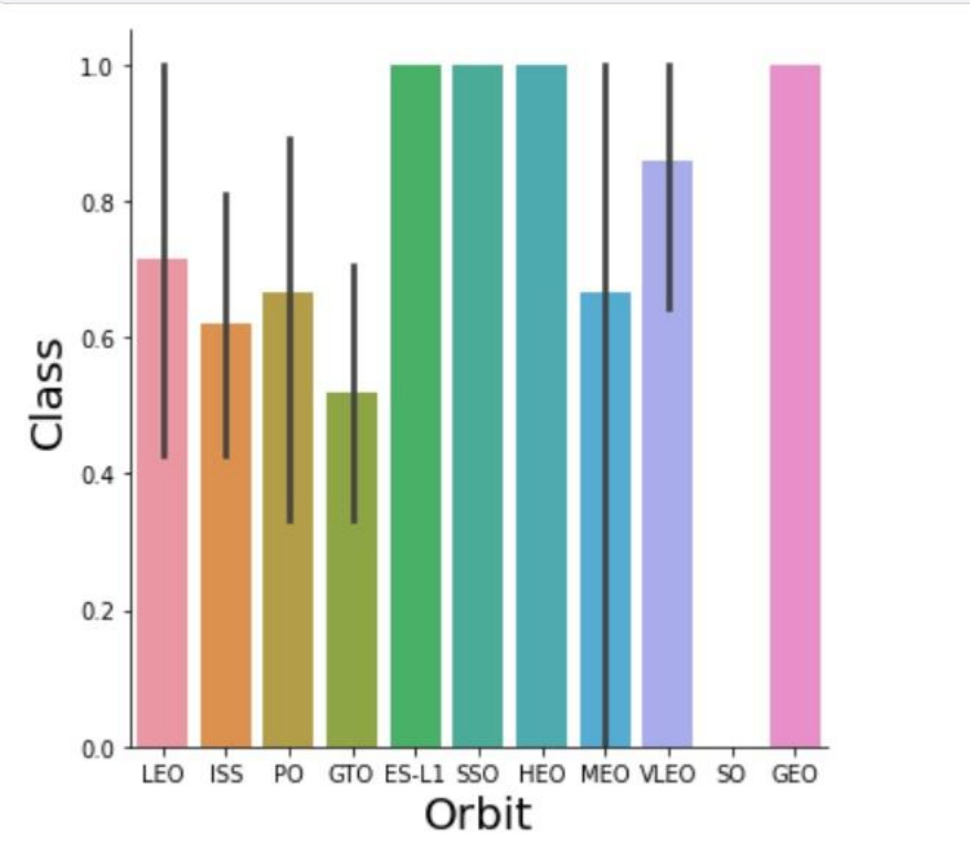
Payload vs. Launch Site



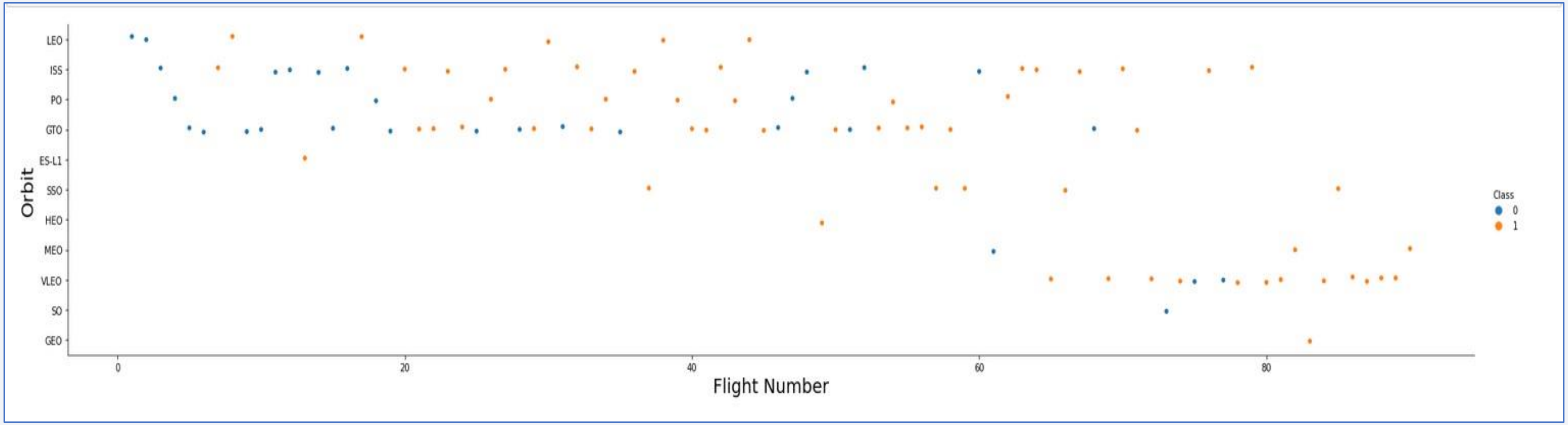
- The more massive the payload, the higher the success rate of a launch site. For the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).

Success Rate vs. Orbit Type

- ES-L1, SSO, HEO and GEO have the best success rate.

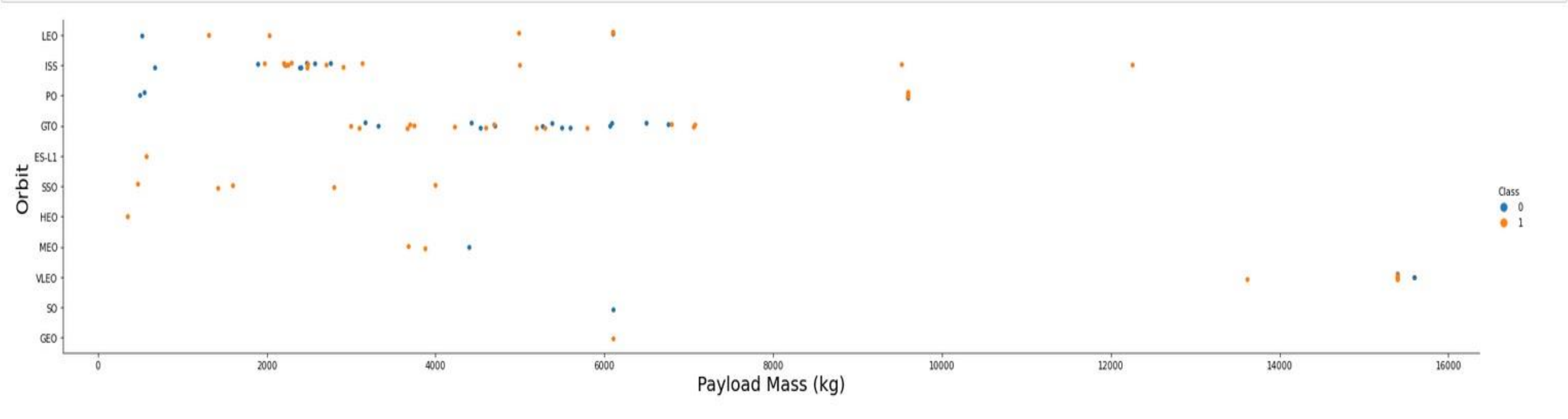


Flight Number vs. Orbit Type



- We see that in the LEO orbit the success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

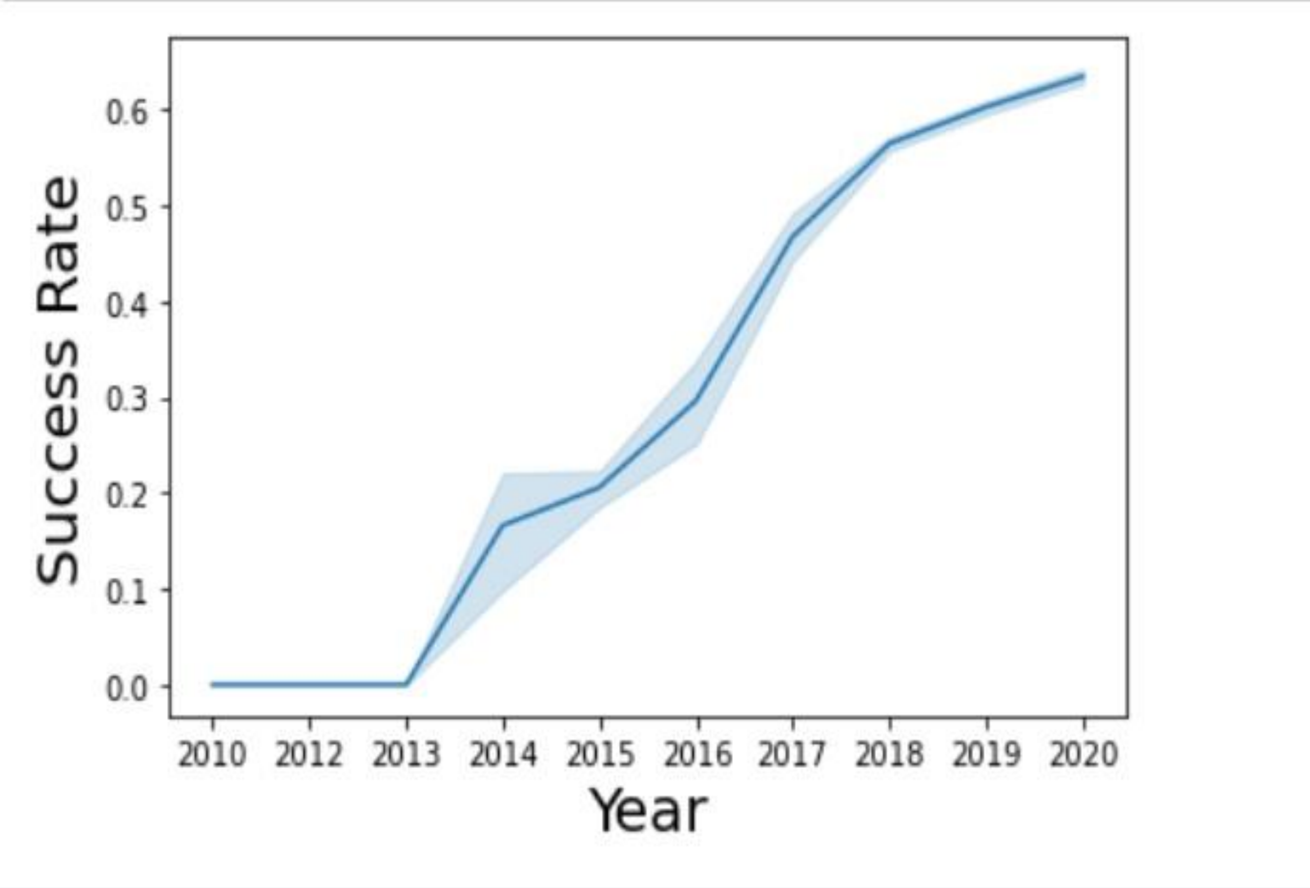
Payload vs. Orbit Type



- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.
- However, for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are there.

Launch Success Yearly Trend

- Here, we can observe that the sucess rate since 2013 kept increasing till 2020.



All Launch Site Names

- The 'DISTINCT' in the query makes sure that only the unique values in the 'LAUNCH_SITE' column of the 'SPACEXDATASET' table are shown.

```
%sql select DISTINCT LAUNCH_SITE from SPACEXDATASET
```

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

Launch Site Names Begin with 'CCA'

- Here, we used the 'like' clause with the word 'CCA' and '%' wildcard operator at the end of it made sure that the launch site name starts with 'CCA'.
- 'limit 5' is there to limit the result to just 5 rows.

```
%sql select * from SPACEXDATASET where LAUNCH_SITE like 'CCA%' limit 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

```
%sql select sum(PAYLOAD_MASS__KG_) as Total_Payload_NASA_CRS from SPACEXDATASET where CUSTOMER = 'NASA (CRS)'
```

total_payload_nasa_crs
45596

- In this query statement, the sum() function returns the total sum of the numeric column 'PAYLOAD_MASS_KG_'.
- 'as' keyword is used to give an alias (Total_payload_NASA_CRS) to the column to increase readability.
- 'where' clause with column name 'CUSTOMER' is used to perform calculations on only NASA (CRS)

Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) as AVG_PAYLOAD_MASS from SPACEXDATASET where BOOSTER_VERSION = 'F9 v1.1'
```

avg_payload_mass
2928

- Avg() function works out the average in the column 'PAYLOAD_MASS_KG_'.
- The 'where' clause filters the dataset and performs calculation on 'BOOSTER_VERSION' 'F9 v1.1'.

First Successful Ground Landing Date

```
%sql select min(DATE) as FIRST_SUCCESSFUL_LANDING_IN_GROUND_PAD from SPACEXDATASET where LANDING__OUTCOME = 'Success (g
round pad)'
```

first_successful_landing_in_ground_pad
2015-12-22

- Min() function is used here to get the lowest value in the 'DATE' column.
- Where clause queries results where the value in 'LANDING_OUTCOME' is 'Success (ground pad)'.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select BOOSTER_VERSION from SPACEXDATASET where LANDING__OUTCOME = 'Success (drone ship)' and PAYLOAD_MASS__KG_ >
4000 \
and PAYLOAD_MASS__KG_ < 6000
```

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

- 'where' clause filters results where 'LANDING_OUTCOME' is 'Success (drone ship)'.
- 'and' clause is used to combine multiple 'where' clauses, in this case, 'PAYLOAD_MASS_KG_ > 4000' and 'PAYLOAD_MASS_KG_ < 6000'.

Total Number of Successful and Failure Mission Outcomes

```
%sql select count(CASE WHEN mission_outcome like 'Success%' then 1 ELSE NULL END ) as Successful_Mission_outcomes, \
COUNT(CASE WHEN mission_outcome like 'Failure%' then 1 ELSE NULL END) as Failure_mission_outcomes from SPACEXDATASET
```

successful_mission_outcomes	failure_mission_outcomes
100	1

- The 'CASE' statement goes through conditions and returns a value when the first condition is met (like an if-then-else statement).
- We used 'count()' function to count the values returned by the 'CASE' statements where 'mission_outcome' are 'Success' and 'Failure'.

Boosters Carried Maximum Payload

```
%sql select booster_version, max(payload_mass__kg_) as Max_Payload_Mass from SPACEXDATASET group by booster_version \
order by Max_Payload_Mass desc
```

booster_version	max_payload_mass
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600
F9 B5 B1051.3	15600
F9 B5 B1051.4	15600
F9 B5 B1051.6	15600
F9 B5 B1056.4	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600
F9 B5 B1060.3	15600

Table continued

- Max() function returns the biggest value in the column 'payload_mass_kg_'.
- 'group by' is used to arrange identical data in 'booster_version' column in groups.
- The keywords 'order by' and 'desc' is used to sort the data in descending order.

2015 Launch Records

```
%sql select booster_version, launch_site, landing__outcome from SPACEXDATASET where landing__outcome = 'Failure (drone ship)' \
and YEAR(date) = 2015
```

booster_version	launch_site	landing__outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

- Values from columns 'booster_version', 'launch_site', and 'landing_outcome' queried.
- 'where' clause is used for column 'landing_outcome' to filter results for only 'Failure (drone ship)' value and 'YEAR(date)' is used to filter results where value is only '2015' in 'date' column.

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

```
%sql select landing__outcome, count(landing__outcome) as Outcomes_Count from SPACEXDATASET group by landing__outcome \
order by Outcomes_Count desc
```

landing__outcome	outcomes_count
Success	38
No attempt	22
Success (drone ship)	14
Success (ground pad)	9
Controlled (ocean)	5
Failure (drone ship)	5
Failure	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

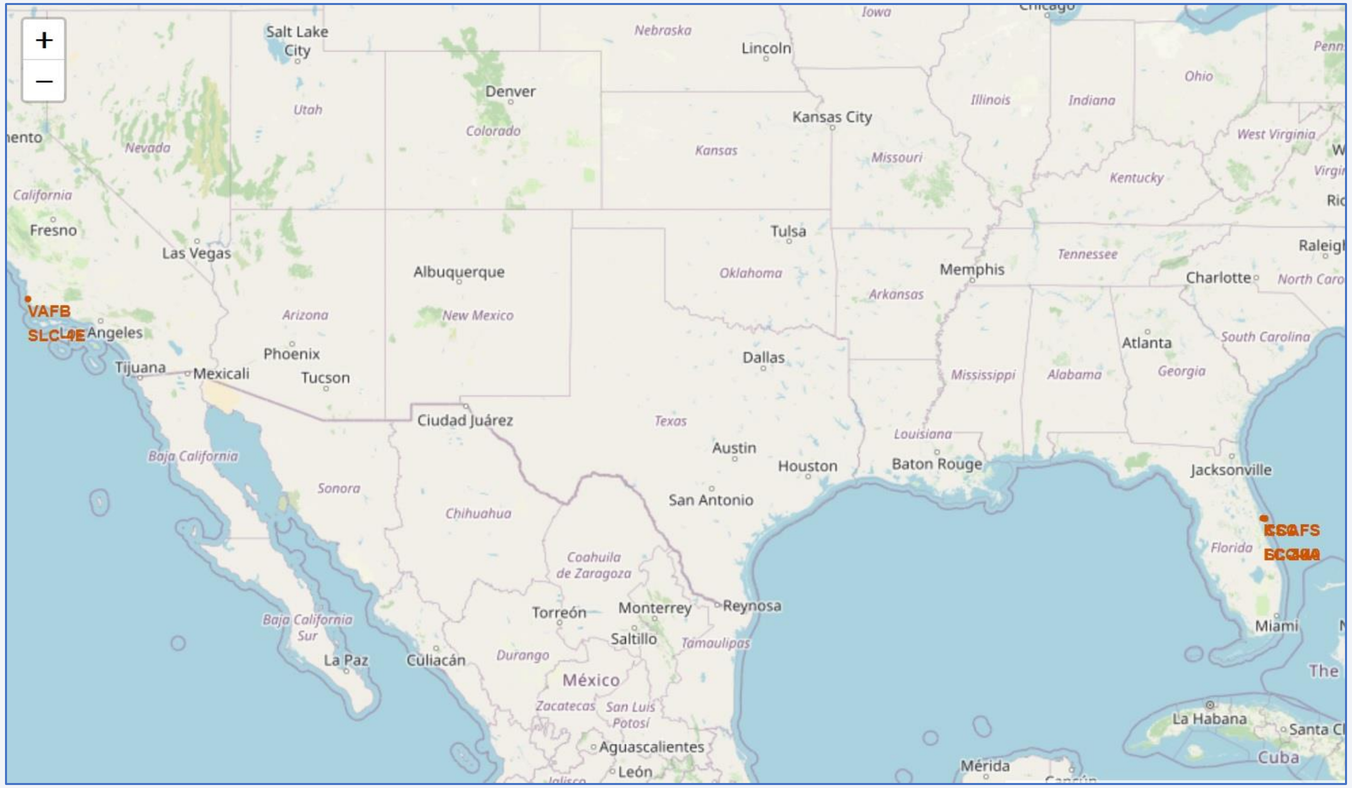
- Count() function is used to count values in 'landing_outcome' under the alias 'Outcomes_count' ('as' keyword used).
- 'group by' used to arrange identical data in 'landing_outcome'.
- Finally, 'order by' and 'desc' keywords used to sort the results in descending order.

Section 4

Launch Sites Proximities Analysis

Launch Sites Marker on Global Map

- Exploring the map reveals that all the launch sites are located very close to coast and are in proximity to the earth's equator line.



Launch Outcomes for each Launch Sites

- Zooming in on a launch site, we can see the launch outcomes for each launch sites.
- Green and red markers represents success and failed outcomes respectively.
- From the color-labeled markers in marker clusters, we are able to easily identify which launch sites have relatively high success rates.

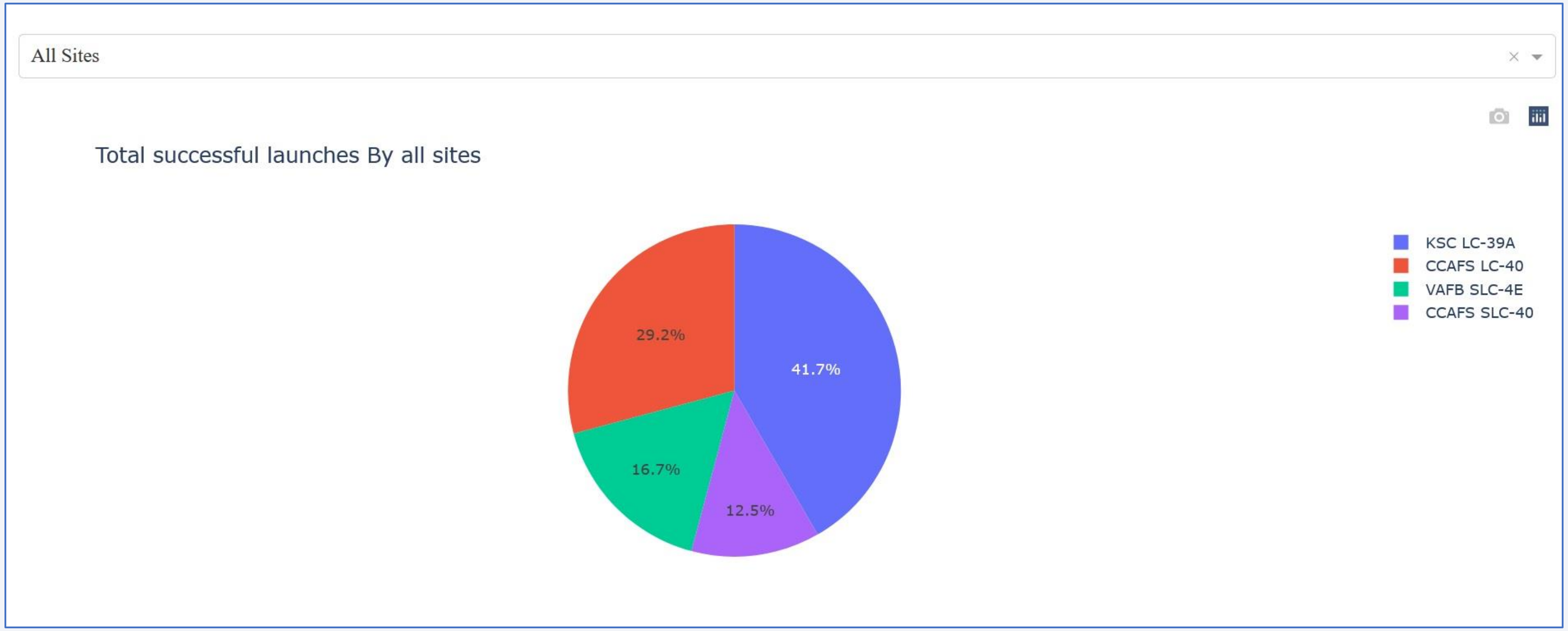
Distance Between a Launch Site to its Proximities

- Launch sites are in close proximity to coastline, railways and highways.
- Launch sites keep certain distance aways from the cities.

Section 5

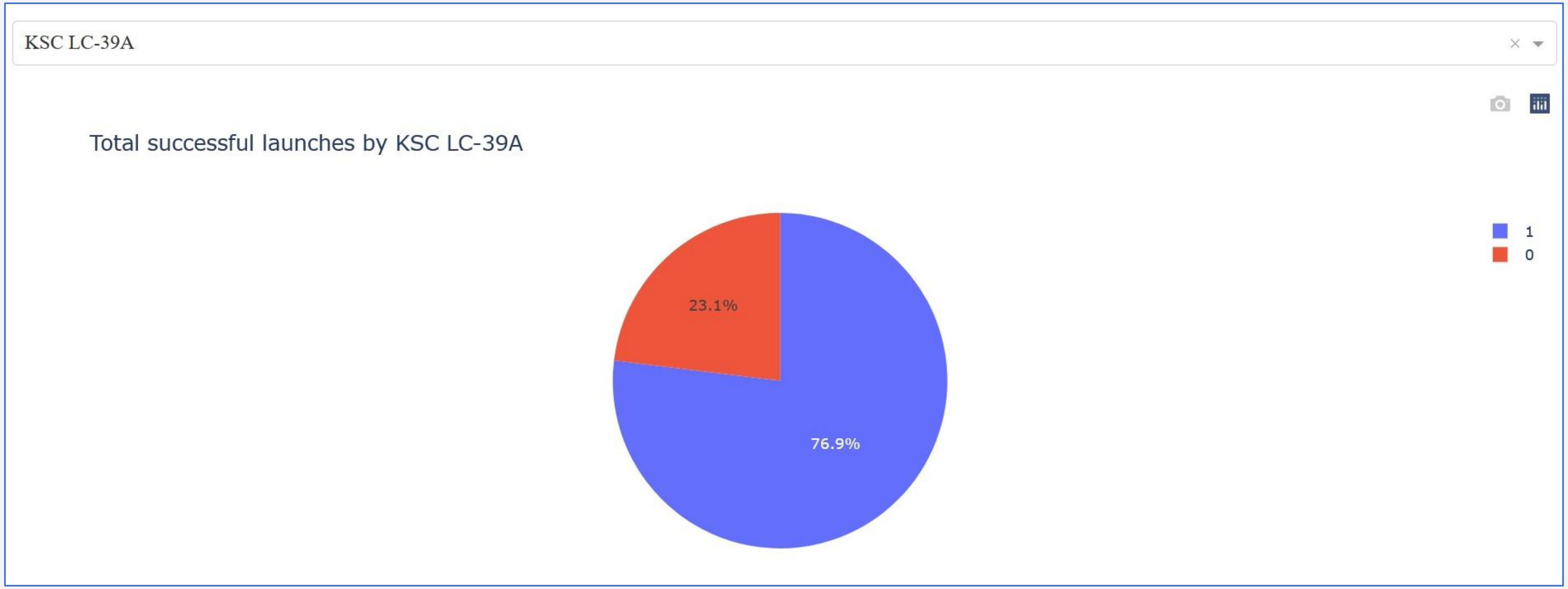
Build a Dashboard with Plotly Dash

Success Rate of Launch Sites



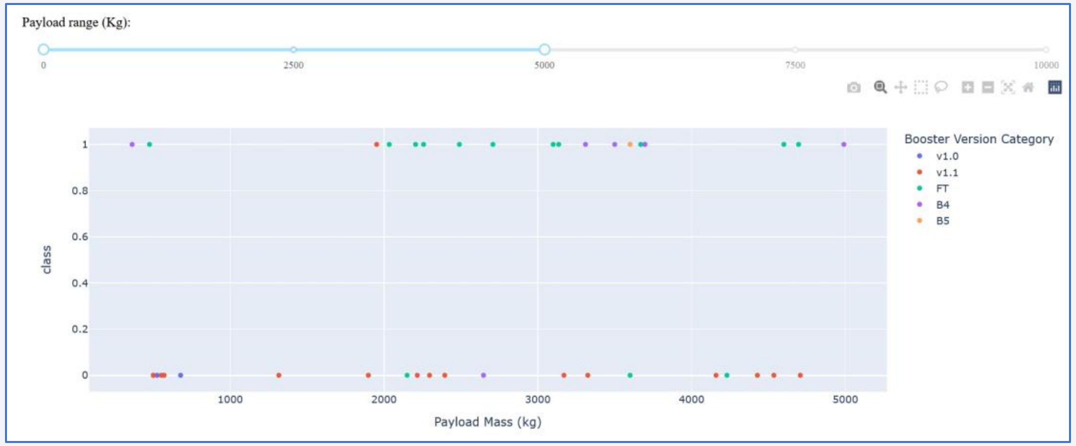
- This pie chart represents the percentage wise success rate of the launch sites.
- KSC LC-39A and CCAFS LC-40 are the two most successful launch sites for spaceX.

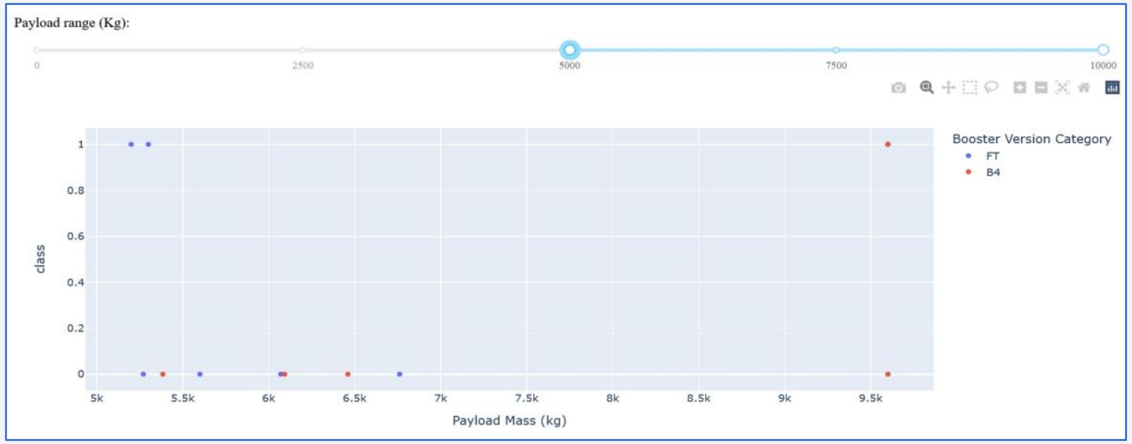
Launch Site With the Highest Success Rate



- With the success rate of 77% KSC LC-39A has the most successful launches.

Relation Between Payload and Launch Outcome





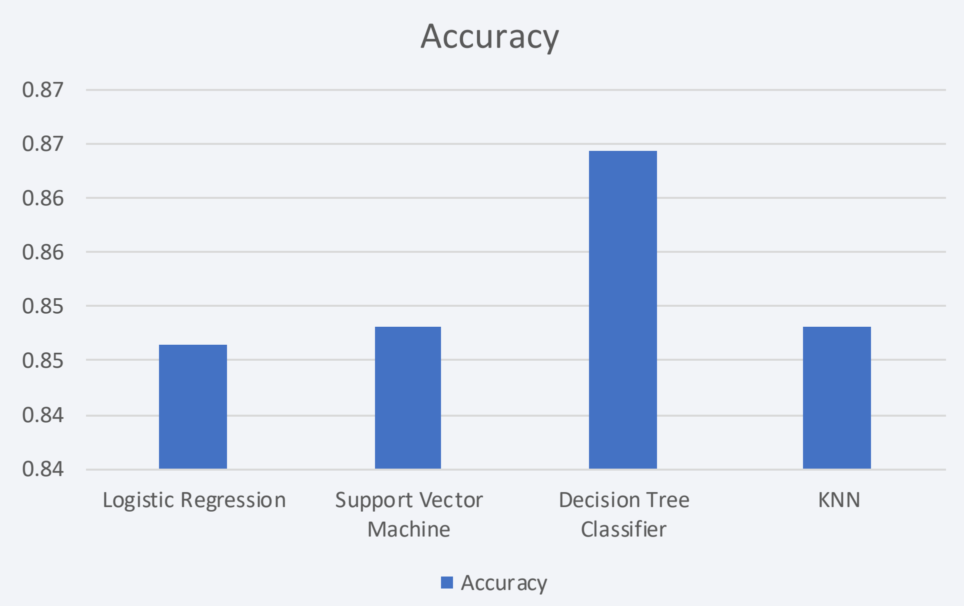
- Payload mass affects the landing outcome inversely.

Section 6

Predictive Analysis (Classification)

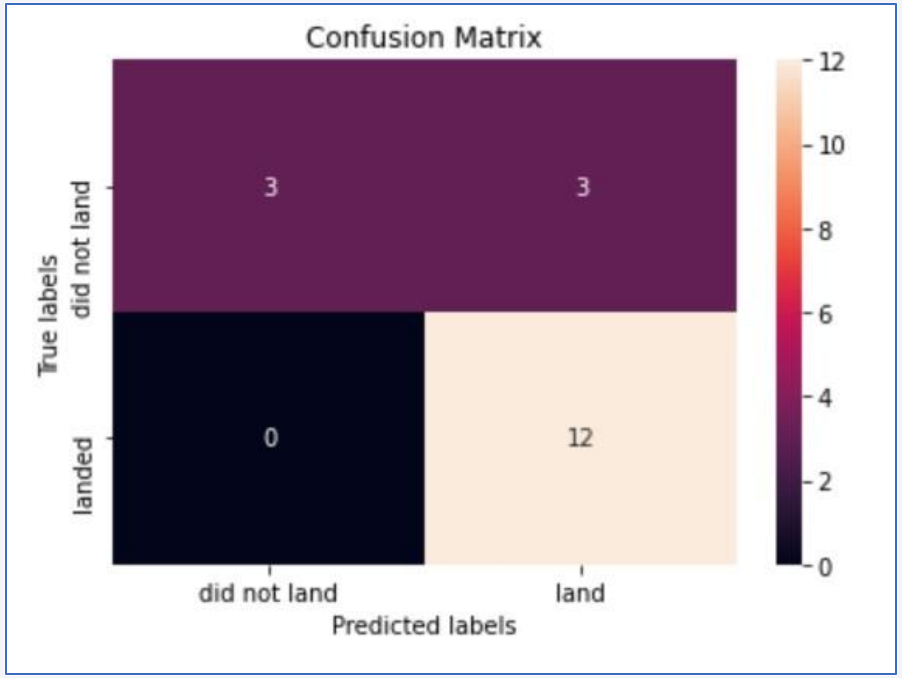
Classification Accuracy

- The Decision Tree Classifier model with an accuracy score of 86% was the best performing model.



Confusion Matrix

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



Confusion Matrix - Decision Tree

Conclusions

- The greater the flight number of a launch site, the greater is its success rate.
- For certain launch sites, success rate increases with an increase in payload mass.
- Missions launched for ES-L1, SSO, HEO and GEO orbits have the highest success rates.
- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.
- Success rate since 2013 kept increasing.
- Launch sites require certain geographical requirements met such close proximity to coastline, railways and highways.
- Launch sites need to be certain distance away from cities.
- Launch site KSC LC-39A has the most successful launches.
- With an accuracy of 86%, the decision tree classifier is the best performing model for our dataset.

Appendix

- Ibm_db_sa, sqlalchemy and ipython-sql libraries for python is used in the notebook.
- Function mentioned in the below screenshot, used to calculate the distance between two points on the map based on their Lat and Long values -

```python
from math import sin, cos, sqrt, atan2, radians

def calculate_distance(lat1, lon1, lat2, lon2):
    # approximate radius of earth in km
    R = 6373.0

    lat1 = radians(lat1)
    lon1 = radians(lon1)
    lat2 = radians(lat2)
    lon2 = radians(lon2)

    dlon = lon2 - lon1
    dlat = lat2 - lat1

    a = sin(dlat / 2)**2 + cos(lat1) * cos(lat2) * sin(dlon / 2)**2
    c = 2 * atan2(sqrt(a), sqrt(1 - a))

    distance = R * c
    return distance
```

Thank you!