IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Jose Cabrera Torres
22/09/2024

Outline

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

Executive Summary

- Summary of methodologies
  - API Data collection
  - Web Scrapping
  - Data Wrangling
  - Data analysis with SQL & Data Visualization
  - Machine Learning
- Summary of all results
  - Data analysis results
  - Interactive analytics
  - Predictive analytics results

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. In this lab, you will create a machine learning pipeline to predict if the first stage will land given the data from the preceding labs.
- Problems you want to find answers
  - Factors to determine if the rocket will land successfully
  - Success rate of landing
  - Operating conditions to ensure successful landing program

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Web scraping from Wikipedia and using SpaceX API
- Perform data wrangling
  - Enconding 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

- Data was collected using various methods
  - Data collection using json api responses
  - Data collection using web scrapping
- You need to present your data collection process use key phrases and flowcharts
  - Parsing api responses with requests and json functions (like normalize)
  - Collecting data using bs4 from wikipeadia after a briev code review
  - Data cleaning, filling in missing values where necessary

Data Collection – SpaceX API

- Data from spaceX API was done using requests
- [Full lab is here: xfer0rz/IBM--Data-Collection-API-Lab]()

```
In [17]:
# Takes the dataset and uses the rocket column to call the API and append the data to the list
def getBoosterVersion(data):
    for x in data['rocket']:
        response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
        BoosterVersion.append(response['name'])

In [18]:
# Takes the dataset and uses the launchpad column to call the API and append the data to the list
def getLaunchSite(data):
    for x in data['launchpad']:
        response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
        Longitude.append(response['longitude'])
        Latitude.append(response['latitude'])
        LaunchSite.append(response['name'])

In [19]:
# Takes the dataset and uses the payloads column to call the API and append the data to the lists
def getPayloadData(data):
    for load in data['payloads']:
        response = requests.get("https://api.spacexdata.com/v4/payloads/"+load).json()
        PayloadMass.append(response['mass_kg'])
        Orbit.append(response['orbit'])

In [20]:
# Takes the dataset and uses the cores column to call the API and append the data to the lists
def getCoreData(data):
    for core in data['cores']:
            if core['core'] != None:
                response = requests.get("https://api.spacexdata.com/v4/cores/"+core['core']).json()
                Block.append(response['block'])
                ReusedCount.append(response['reuse_count'])
                Serial.append(response['serial'])
            else:
                Block.append(None)
                ReusedCount.append(None)
                Serial.append(None)
            Outcome.append(str(core['landing_success'])+' '+str(core['landing_type']))
            Flights.append(core['flight'])
            GridFins.append(core['gridfins'])
            Reused.append(core['reused'])
            Legs.append(core['legs'])
            LandingPad.append(core['landpad'])

In [26]:
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = response.json()
# apply json_normalize
data = pd.json_normalize(static_json_df)

print(data.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 187 entries, 0 to 186
Data columns (total 43 columns):
 #   Column                     Non-Null Count  Dtype
```

Data Collection - Scraping

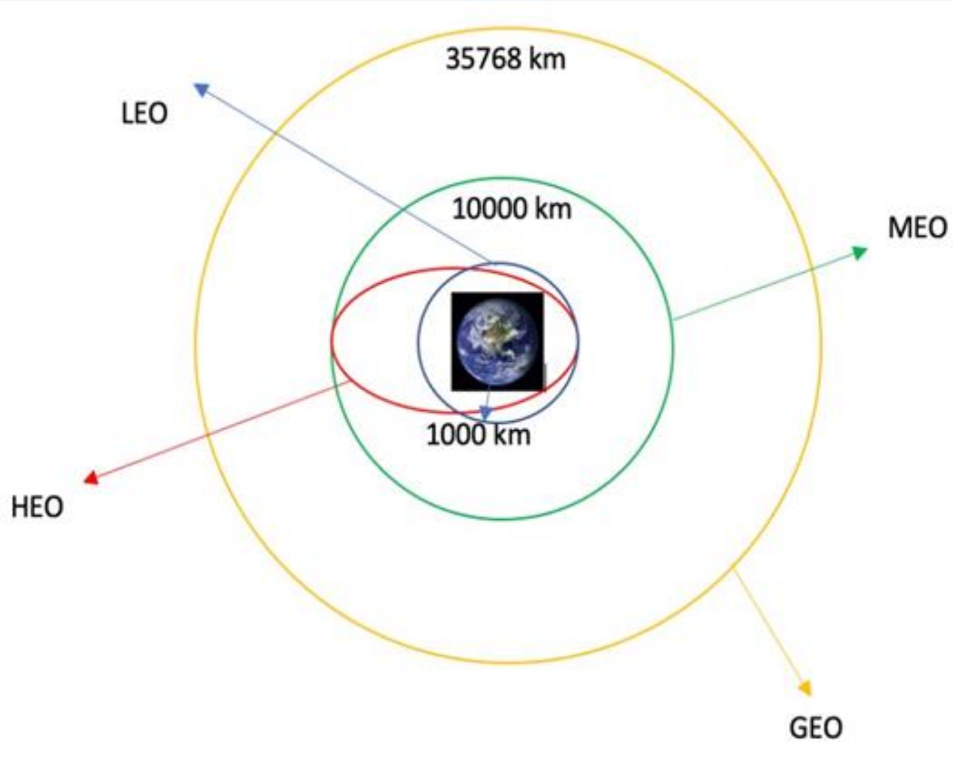
- Data from wikipeadia was collected and parsed using bs4

- Link to the lab on page 8.

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

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>

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)

         <table class="wikitable plainrowheaders collapsible" style="width: 100%;">
         <tbody><tr>
         <th scope="col">Flight No.
         </th>
         <th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>)
         </th>
         <th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">Versio
         n,<br/>Booster</a> <sup class="reference" id="cite_ref-booster_11-0"><a href="#cite_note-booster-11"><span class="cite-bra
         cket">[</span>b<span class="cite-bracket">]</span></a></sup>
         </th>
         <th scope="col">Launch site
         </th>
         <th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-0"><a href="#cite_note-Dragon-12"><span class="cite-b
         racket">[</span>c<span class="cite-bracket">]</span></a></sup>
         </th>
         <th scope="col">Payload mass
         </th>
         <th scope="col">Orbit
         </th>
```

Data Wrangling

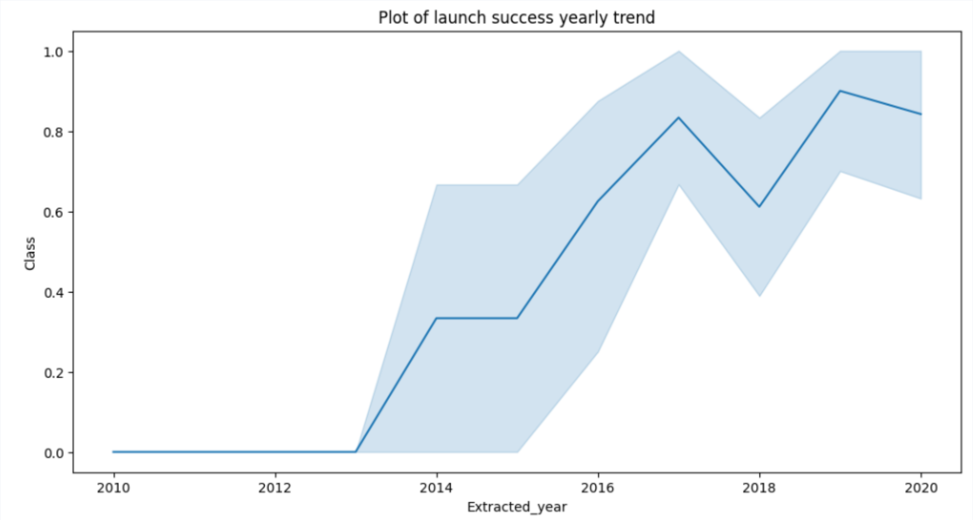
- Exploratory data analysis
- Determined Training labels
- Calculated number of launches
- Created landing outcome
- Exported results to csv

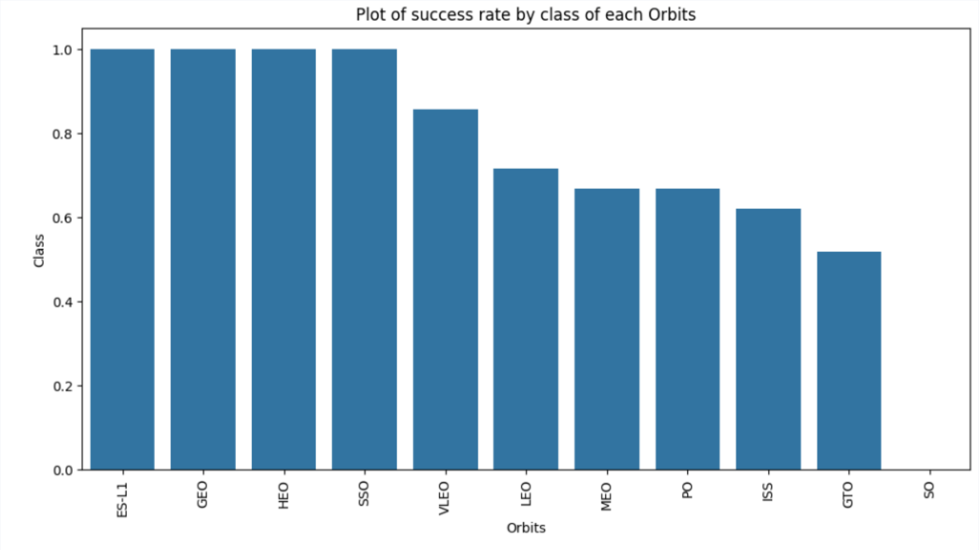


- [Link to the lab on page 8.]

EDA with Data Visualization

- Data was exported due to the relationship between flight number and launch site, payload and launch site, success rate for each orbit, flight number and orbit, at the end, success yearly trend.





- [Link to the lab on page 8.]

EDA with SQL

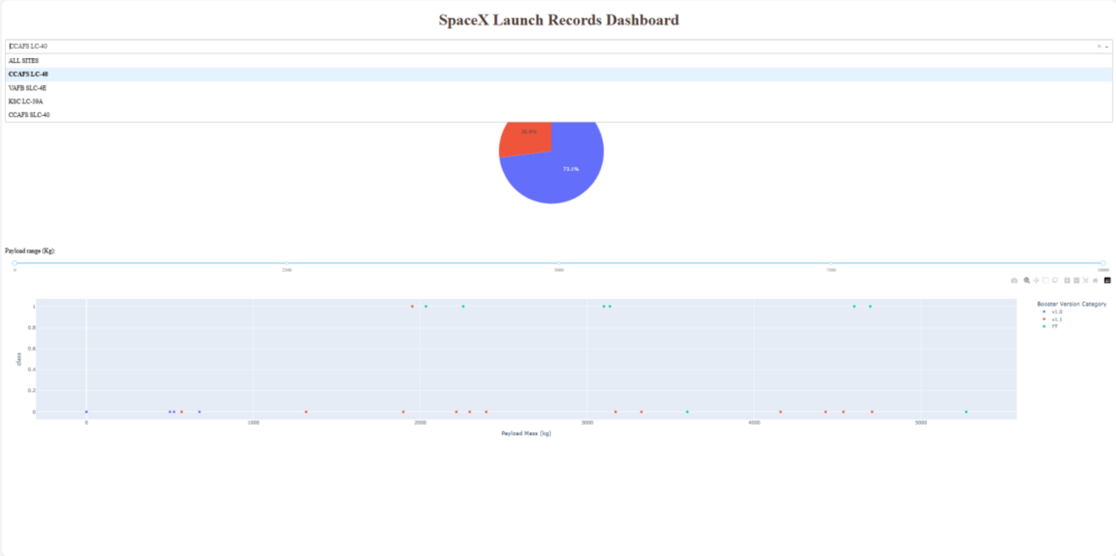
- We filled a sqlite db from csv data
- Then queried sung sql sintax like:
  - Name of unique launches
  - Total payload mass carried by boosters
  - Average payload mass carried by boosters
  - Total number of success & failure mission outcomes
  - Failed landing outcomes in drone ship, booster version and launch site names
- Link to the lab on page 8.

Build an Interactive Map with Folium

- Marked all launch sites, added map objects, like markers, circles, lines, marked success and failures of launches for each site
- Assigned feature launch outcomes to class 0 and 1 (1 for success)
- Using colour marker clusters, identified which launch sites have relatively high success rate
- Calculated distances between a launch site to its proximities and answered a few questions about distances.
- Link to the lab on page 8.

Build a Dashboard with Plotly Dash

- Built an interactive dashboard with plotly dash
- Plotted pie chart showing total launches
- Plotted scatter graph showing relationship with outcome and payload mass for boosters versions
- [Link to the lab on page 8.]



Predictive Analysis (Classification)

- Loaded data using numpy and pandas, transformed, splitted into training and testing datasets
- Built different ML models and tune different parameters using GridSearchCV
- Used accuracy as metric, improved the model using feature eng. And algorithm tuning.
- Found the best performing classification model.
- [Link to the lab on page 8.]

TASK 12 Find the method performs best:

In [18]:

```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.8625
Best Params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
```

Results

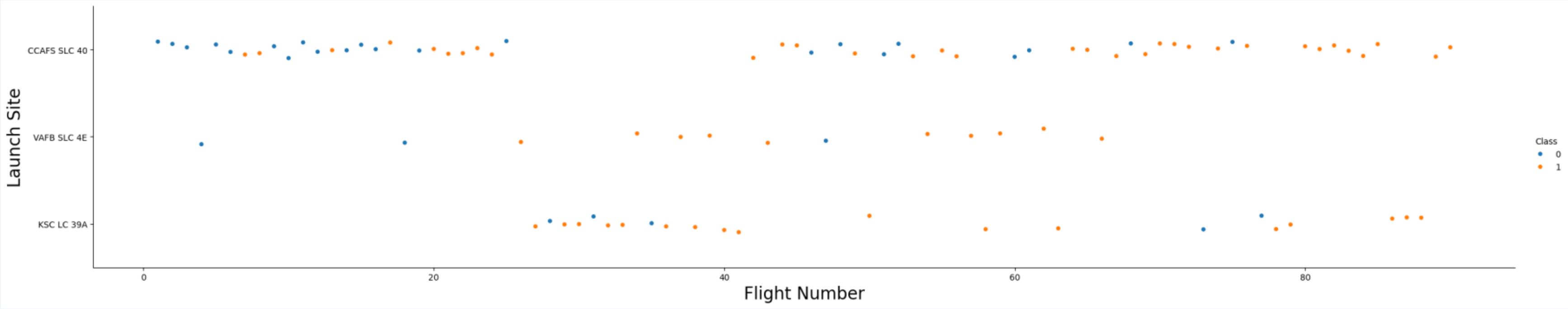
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

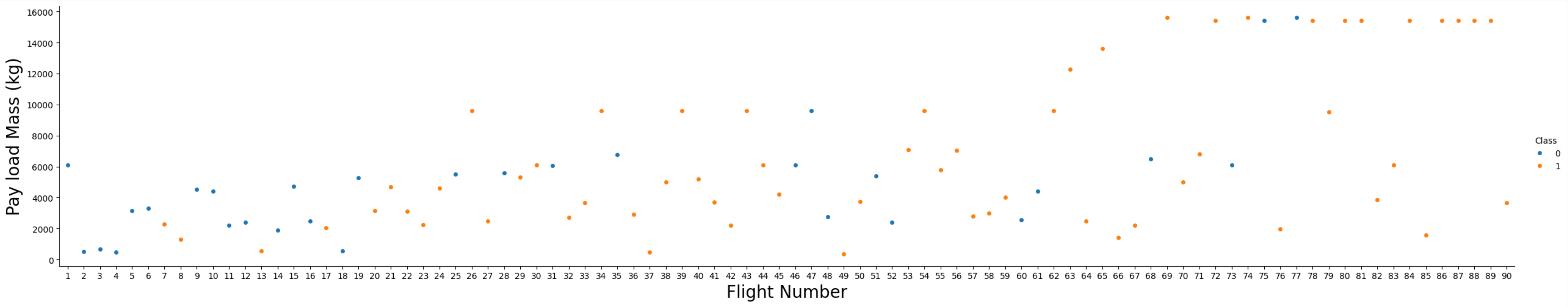
- Show a scatter plot of Flight Number vs. Launch Site



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

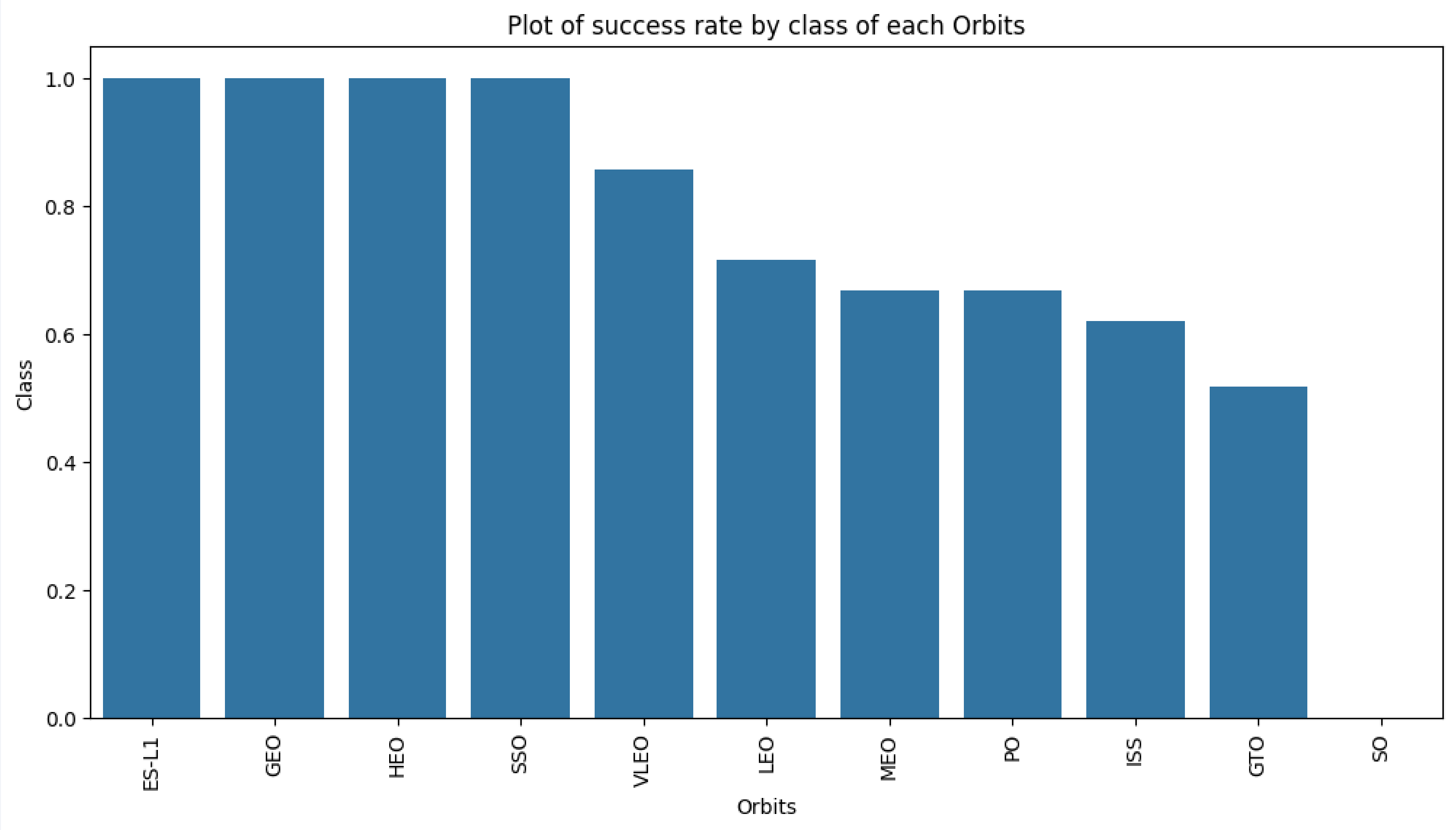
Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site



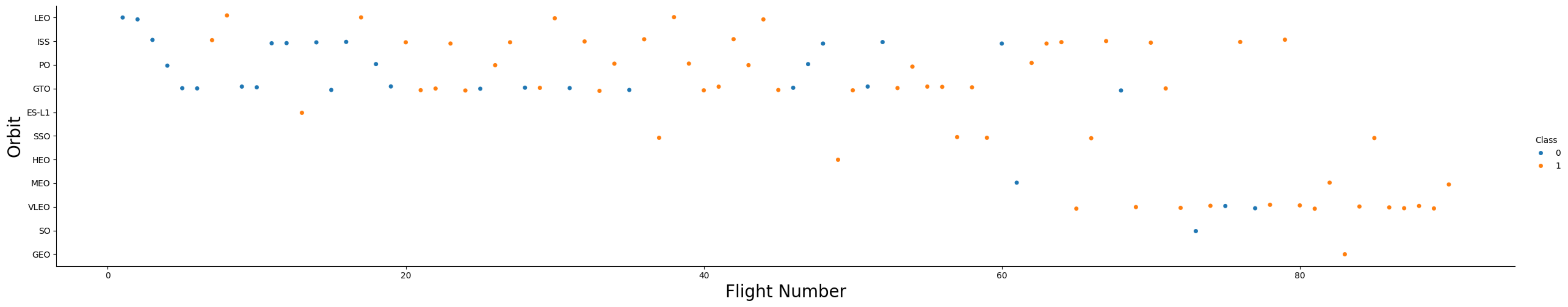
- The greater the payload mass for launch site, higher success rate for the rocket

Success Rate vs. Orbit Type

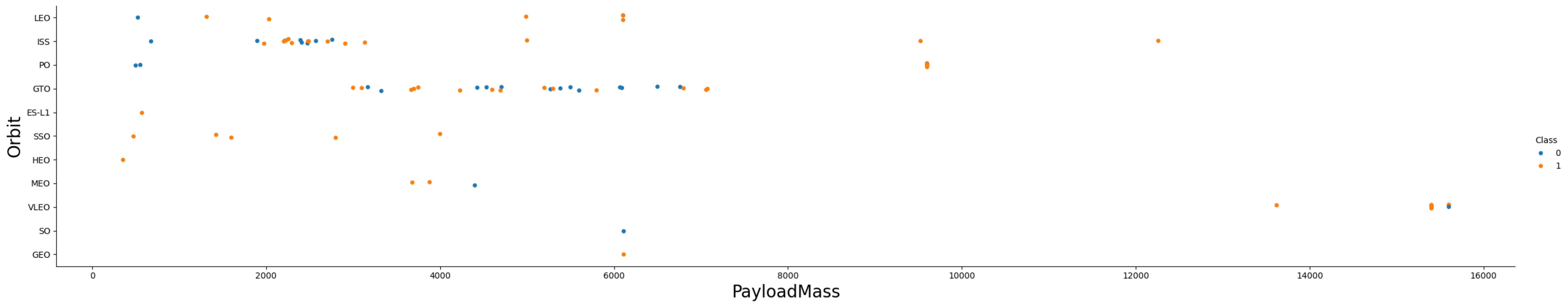


- From the plot, ESL-L1, GEO, HEO, SSO, VLEO had the most success rate

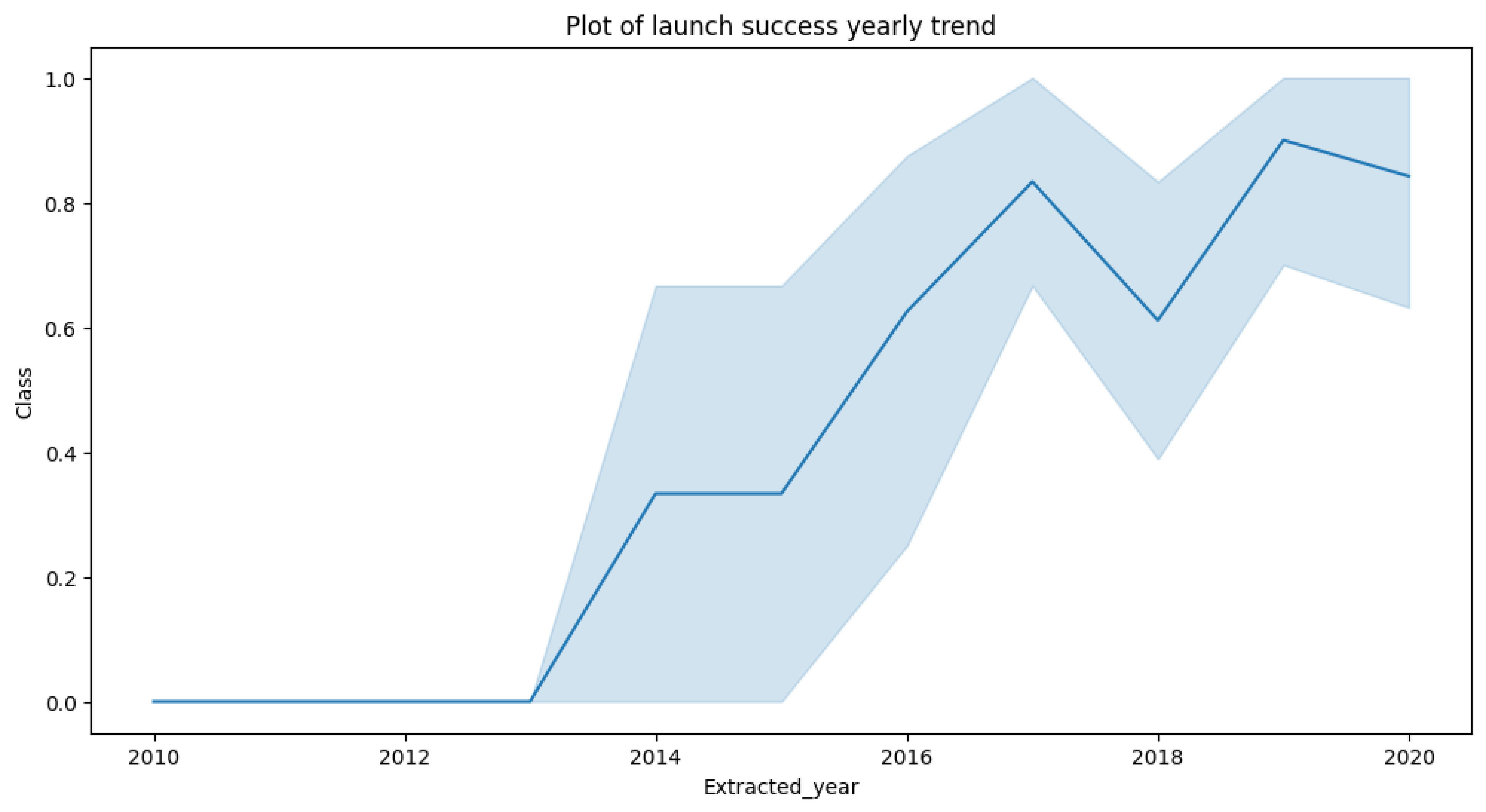
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



- Sh
  av
- Sh
  sc

All Launch Site Names

- Find the names of the unique launch sites
- Present your query result with a short explanation here

In [25]:
```
%sql select distinct Launch_Site from SPACEXTBL
```

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

Out[25]:

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

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- Present your query result with a short explanation here

In [26]:
```
%sql select * from SPACEXTBL where Launch_Site like 'CCA%' limit 5
```

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

Out[26]:

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

- Calculate the total payload carried by boosters from NASA
- Present your query result with a short explanation here

Task 3 Display the total payload mass carried by boosters launched by NASA (CRS)

```
In [29]: %sql select sum(payload_mass__kg_) from SPACEXTBL WHERE customer = 'NASA (CRS)'

 * sqlite:///my_data1.db
Done.

Out[29]:
```

sum(payload_mass__kg_)
45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here

Task 4 Display average payload mass carried by booster version F9 v1.1

```
In [30]: %sql select avg(payload_mass__kg_) from SPACEXTBL WHERE booster_version = 'F9 v1.1'

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

Out[30]:

avg(payload_mass__kg_)
2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Present your query result with a short explanation here

Task 5 List the date when the first successful landing outcome in ground pad was acheived. Hint:Use min function

In [32]:
```
%sql select min(DATE) from SPACEXTBL WHERE landing_outcome = 'Success (ground pad)'
```

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

Out[32]:

min(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

- Present your query resul

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

In [34]:
```
%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.
```

Out[34]:

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

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here

```
In [35]: %sql select mission_outcome, count(mission_outcome) from SPACEXTBL GROUP BY mission_outcome

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

Out[35]:

Mission_Outcome	count(mission_outcome)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Present your query result w

Task 8 List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

In [36]:
```
%sql select booster_version, payload_mass__kg_ from SPACEXTBL\
where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXTBL)
```

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

Out[36]:

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

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

- Present your query re

Task 9 List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [41]:
```
ion, launch_site from SPACEXTBL where landing_outcome = 'Failure (drone ship)' and Date BETWEEN '2015-01-01' AND '2015-12-31'
```

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

Out[41]:

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

- 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

- Present your query r

Task 10 Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

In [40]:
```
%sql select count(landing_outcome), landing_outcome from SPACEXTBL \
where DATE between '2010-06-04' and '2017-03-20' group by landing_outcome\
order by count(landing_outcome) desc
```

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

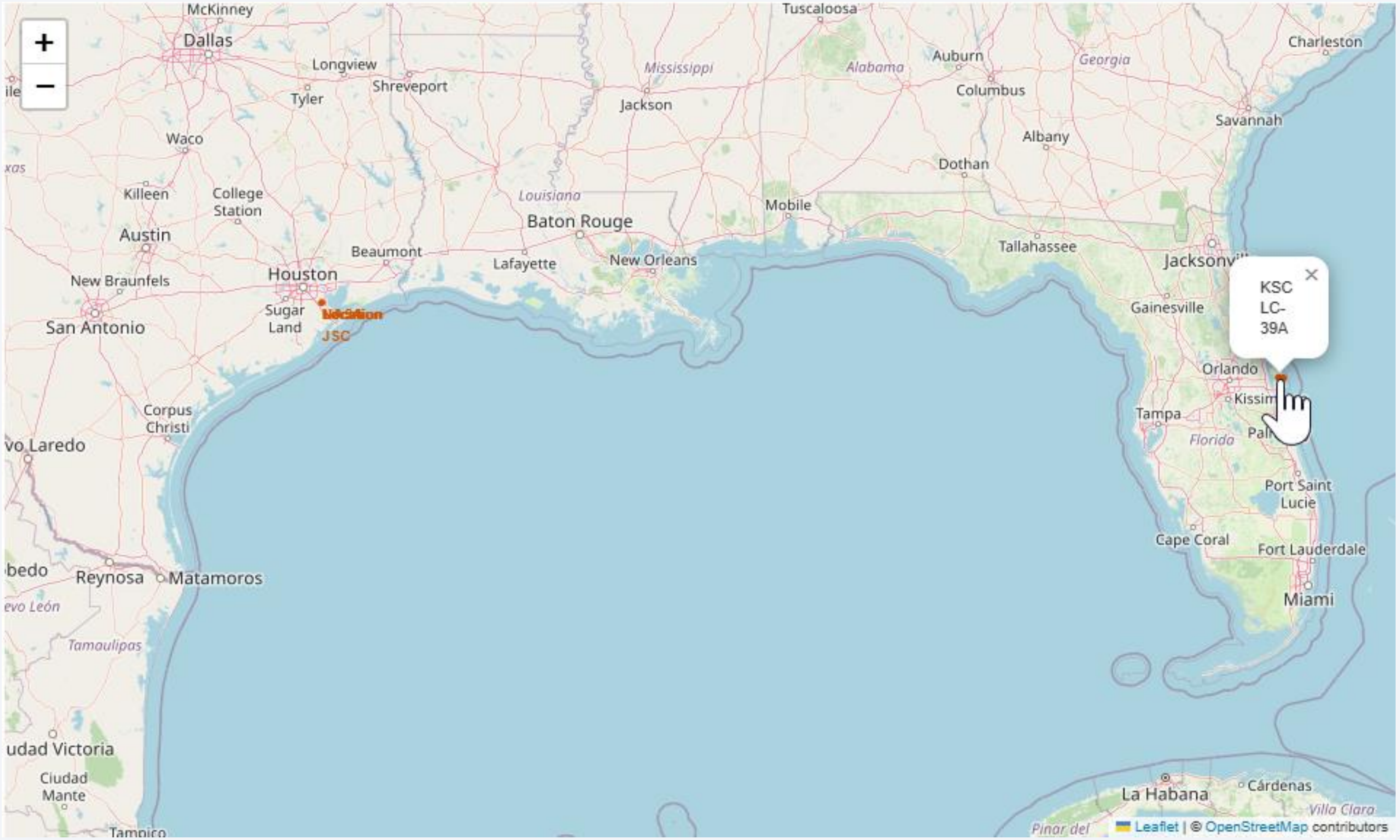
Out[40]:

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

Section 3

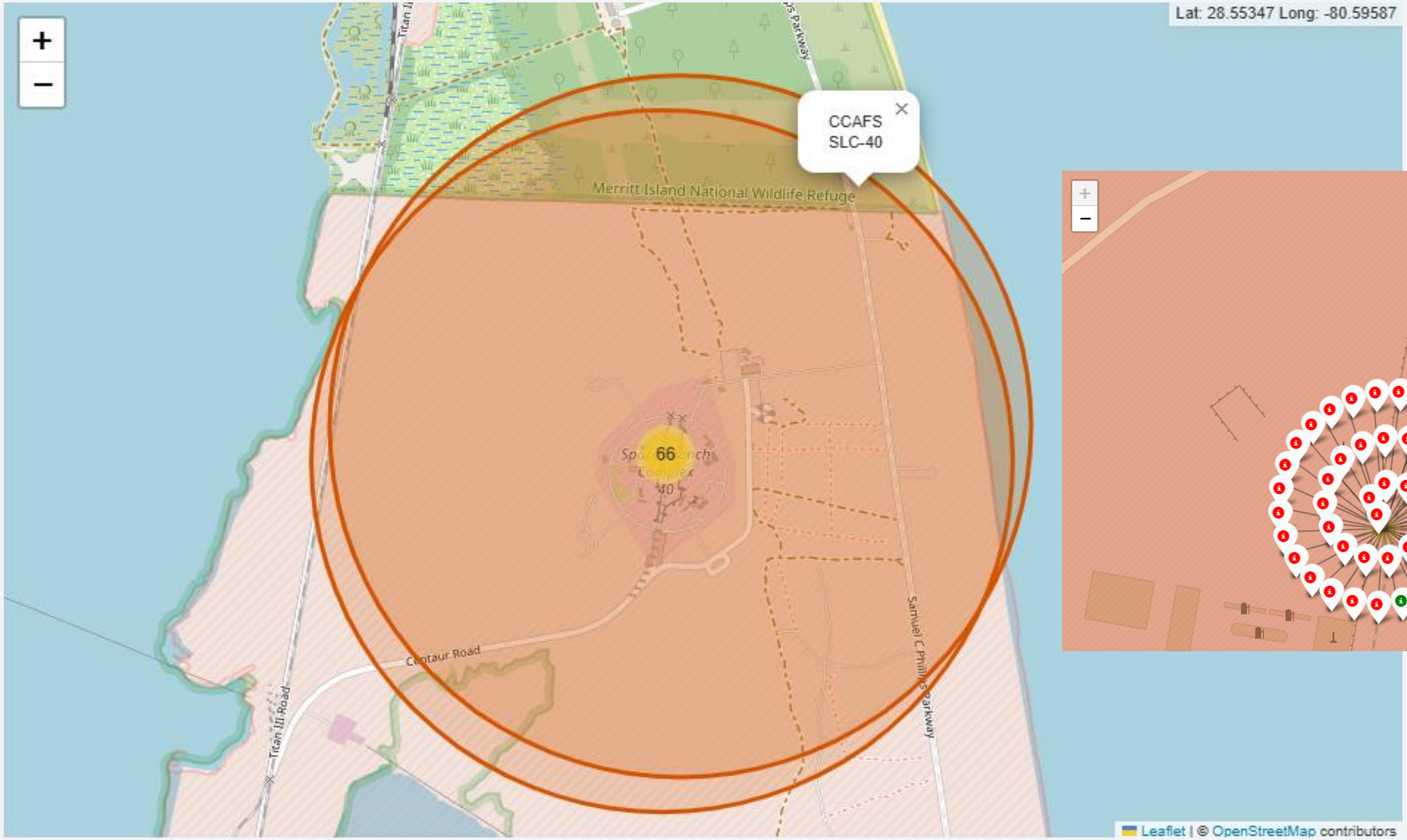
Launch Sites Proximities Analysis

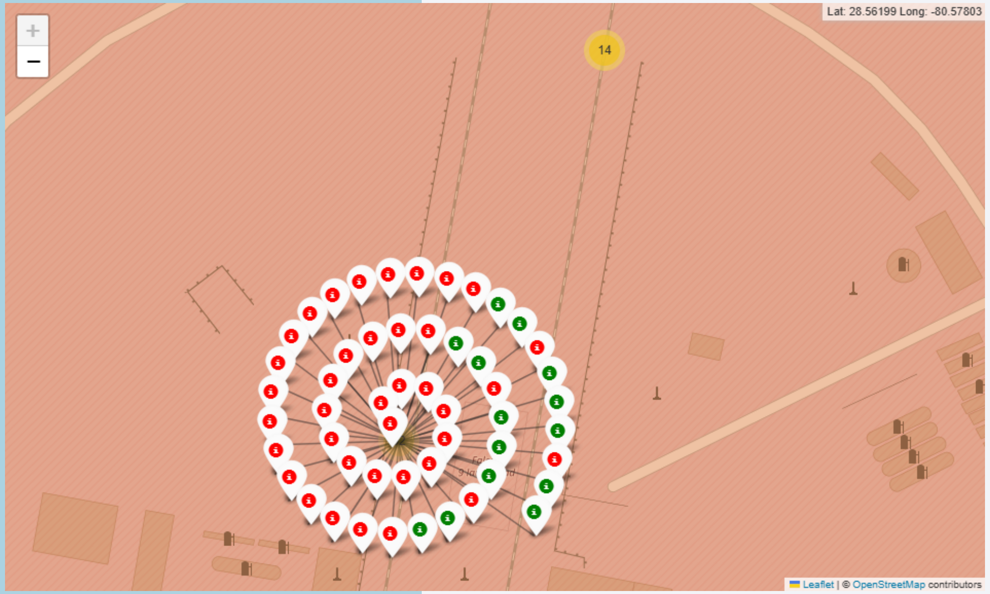
All launch sites global markers



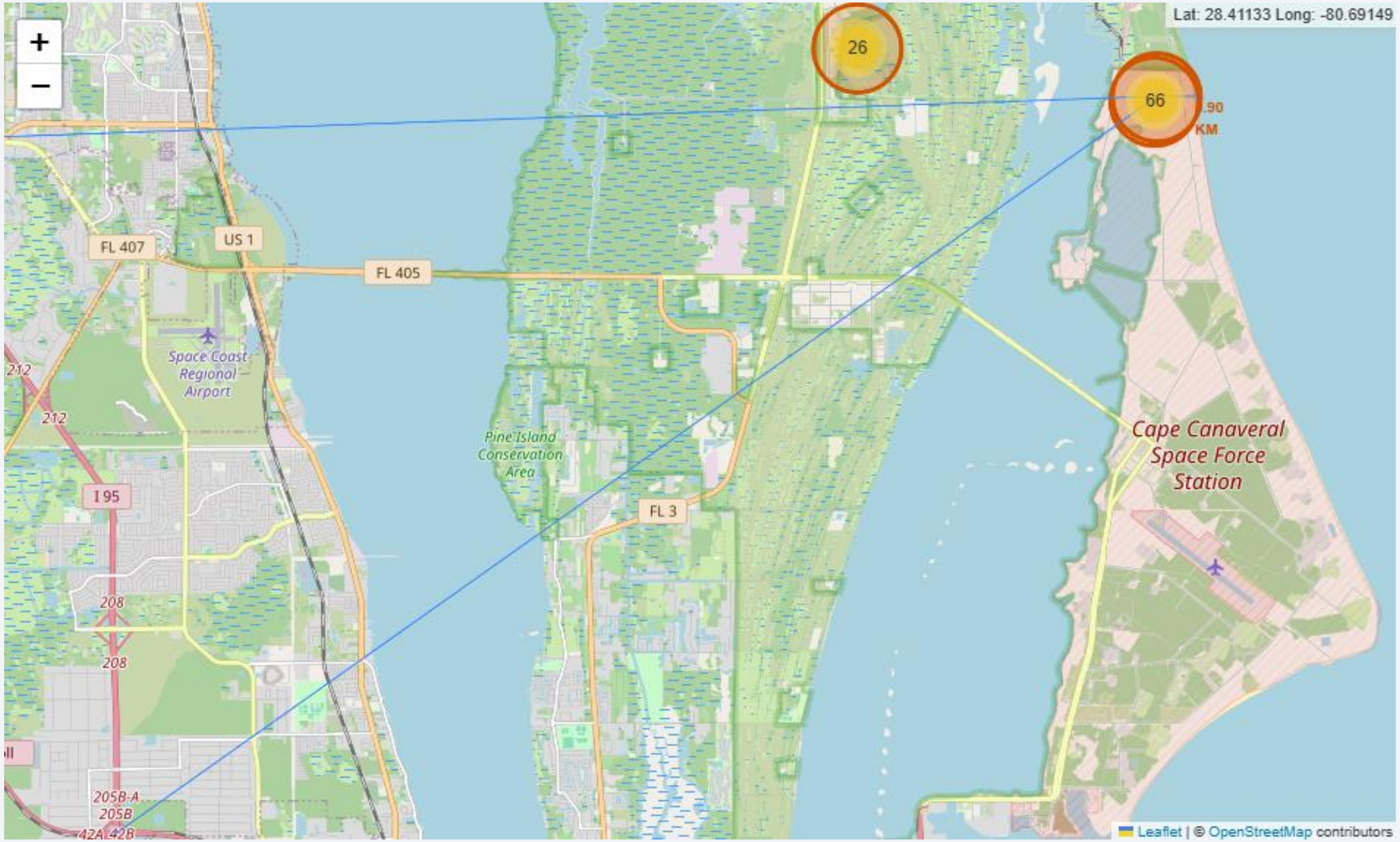
- SpaceX launch sites are in EEUU coasts, like Florida and California

Markers showing launch sites with colour labels





Launch site distance to marks

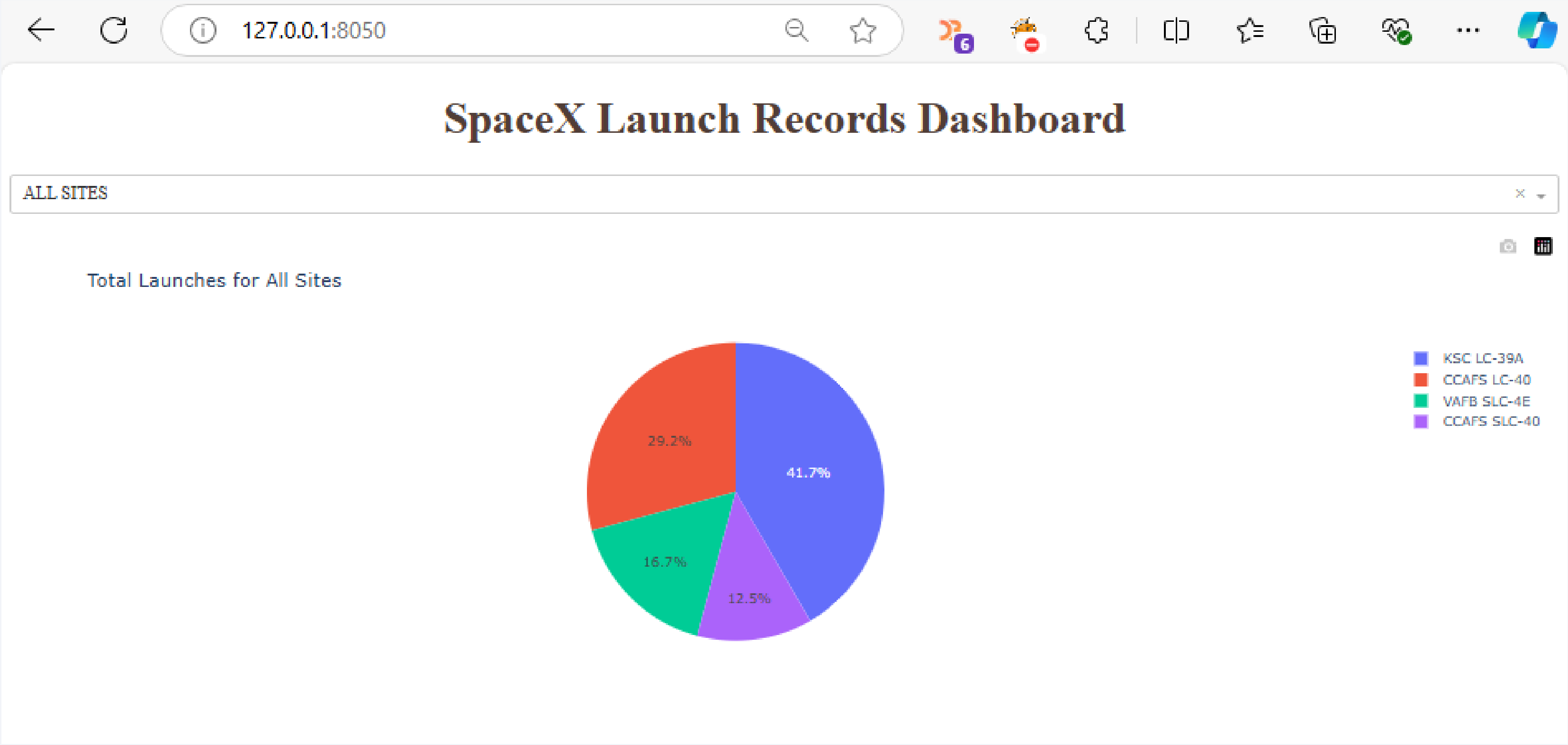


Section 4

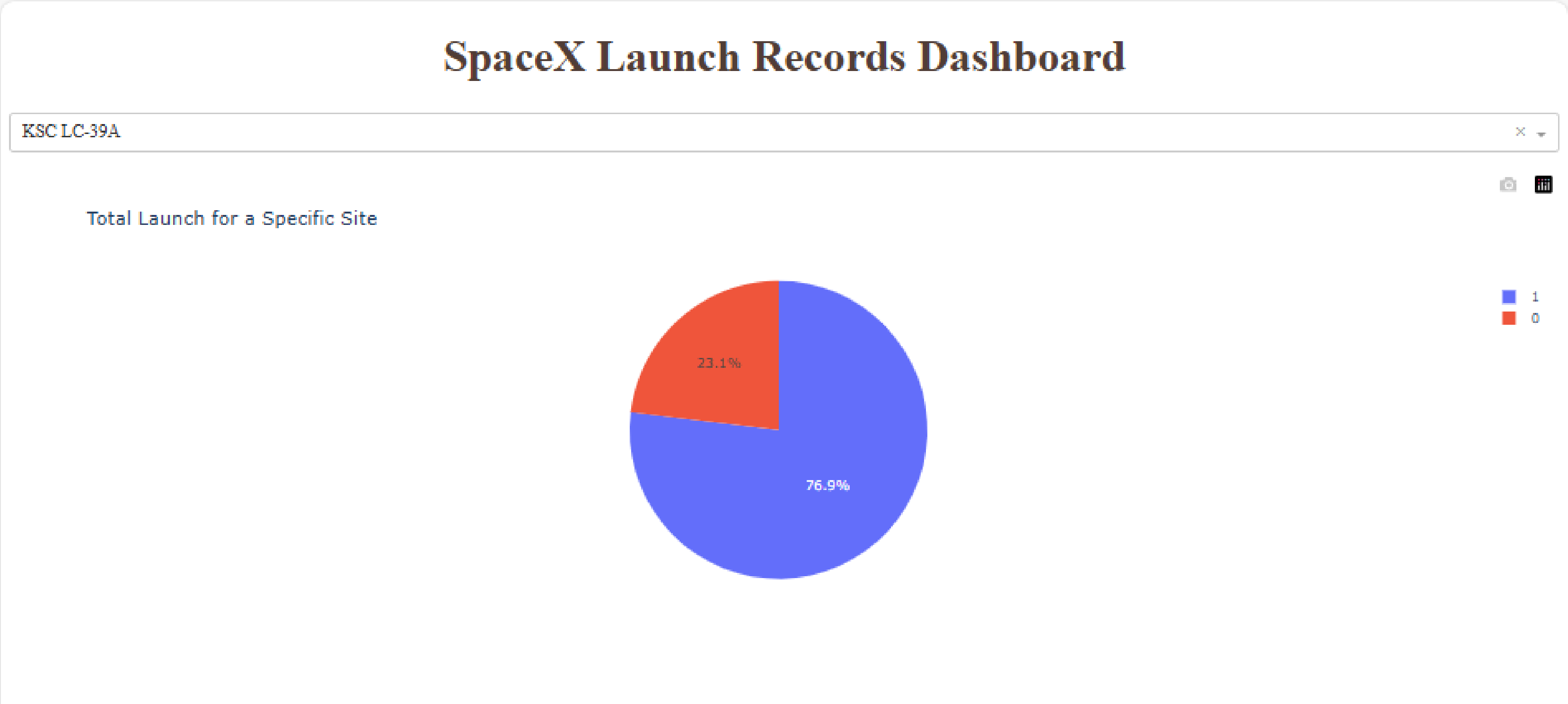
Build a Dashboard with Plotly Dash

Pie chart showing global success percentage

- KSC LC-39A had the most successfull launches from all sites overall

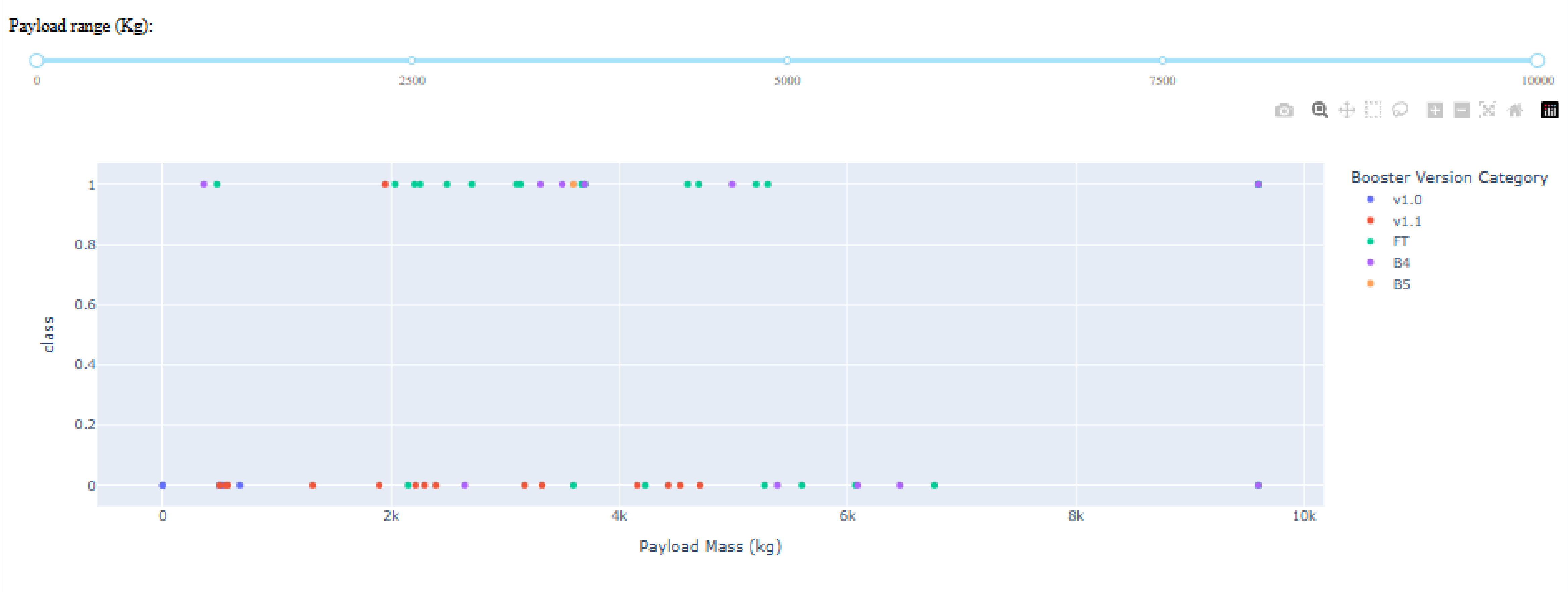


Pie chart showing the launch site with highest success rate



- Achieved almost 80% of success rate

Scatter plot of payload vs launch outcome



Section 5

Predictive Analysis (Classification)

Classification Accuracy

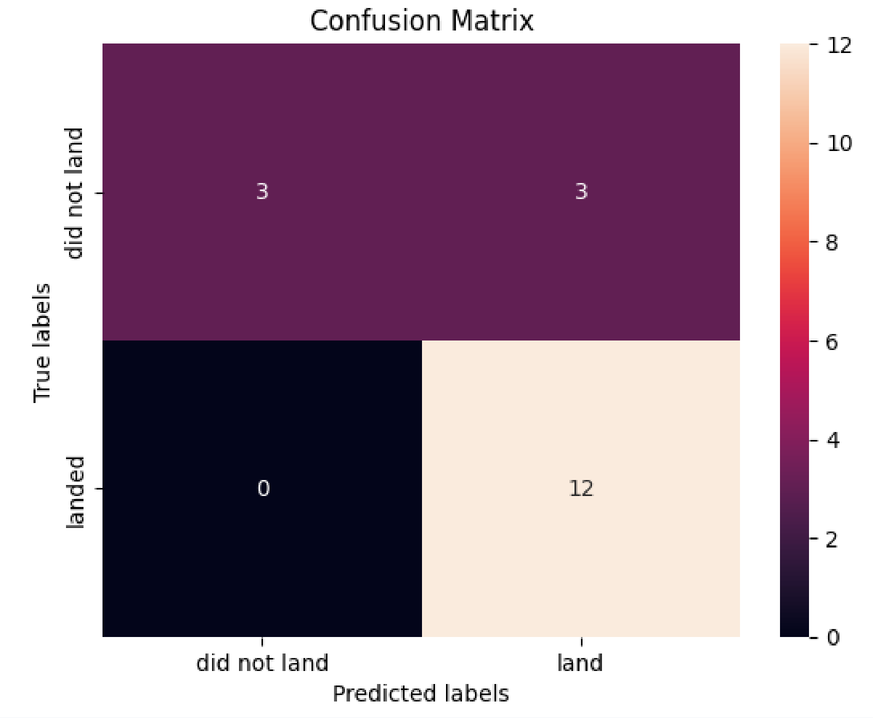
TASK 12 Find the method performs best:

```
In [18]: algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
         bestalgorithm = max(algorithms, key=algorithms.get)
         print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
         if bestalgorithm == 'Tree':
             print('Best Params is :',tree_cv.best_params_)
         if bestalgorithm == 'KNN':
             print('Best Params is :',knn_cv.best_params_)
         if bestalgorithm == 'LogisticRegression':
             print('Best Params is :',logreg_cv.best_params_)

         Best Algorithm is Tree with a score of 0.8625
         Best Params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split':
         2, 'splitter': 'best'}

In [ ]:
```

- Decision tree classifier is the model with the highest classification accuracy

Confusion Matrix

- Confusion matrix can distinguish between the different classes. Major problem is false positives like unsuccessful landing marked as successful by the classifier.



Conclusions

- The larger the flight amount at a launch site, greater success rate.
- Launch success rate started to ingres in 2013.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39ª had the most successful launches of any sites.
- Decission tree clasifier is the best ML algorithm in order to predict the outcome of the landing program.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!