Winning Space Race with Data Science

<Jiahui Li>

< 2021-09-27 >

Outline

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

Executive Summary

- Summary of methodologies
- We will use this method,　EDA with Data Visualization、EDA with SQL、Build an Interactive Map with FoliumBuild a 、Dashboard with Plotly Dash

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.
- Problems you want to find answers
  - What we are looking for is the probability of successful launch and return of the rocket, as well as the analysis of the launch and development of the company over the years.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- 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

- Describe how data sets were collected.
  - requesting rocket launch data from SpaceX API with the following URL:

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

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

And will get

```
print(response.content)
```


b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.com/3c/0e/T8iJcSN3_o.png","large":"https://images2.imgbox.com/40/e3/GypSkayF_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.space.com/2196-spacex-inaugural-falcon-1-rocket-lost-launch.html","wikipedia":"https://en.wikipedia.org/wiki/DemoSat"},"static_fire_date_utc":"2006-03-17T00:00:00.000Z","static_fire_date_unix":1142553600,"net":false,"window":0,"rocket":"5e9d0d95eda69955f709d1eb","success":false,"failures":[{"time":33,"altitude":null,"reason":"merlin engine failure"}],"details":"Engine failure at 33 seconds and loss of vehicle","crew":[],"ships":[],"capsules":[],"payloads":["5eb0e4b5b6c3bb0006eeb1e1"],"launchpad":"5e9e4502f5090995de566f86","flight_number":1,"name":"FalconSat","date_utc":"2006-03-24T22:30:00.000Z","date_unix":1143239400,"date_local":"2006-03-25T10:30:00+12:00","date_precision":"hour","upcoming":false,"cores":[{"core":"5e9e289df35918033d3b2623","flight":1,"gridfins":false,"legs":false,"reused":false,"landing_attempt":false,"landing_success":null,"landing_type":null,"landpad":null}],"auto_update":true,"tbd":false,"launch_library_id":null,"id":"5eb87cd9ffd86e000604b32a"},{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.com/4f/e3/I0lkuJ2e_o.png","large":"https://images2.imgbox.com/be/e7/iNqsqVYM_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?v=Lk4zQ2wP-Nc","youtube_id":"Lk4zQ2wP-Nc","article":"https://www.space.com/3590-spacex-falcon-1-rocket-fails-reach-orbit.html","wikipedia":"https://en.wikipedia.org/wiki/DemoSat"},"static_fire_date_utc":null,"static_fire_date_unix":null,"net":false,"window":0,"rocket":"5e9d0d95eda69955f709d1eb","success":false,"failures":[{"time":301,"altitude":289,"reason":"harmonic oscillation leading to premature engine shutdown"}],"details":"Successful first stage burn and transition to second stage, maximum altitude 289 km, Premature engine shutdown at T+7 min 30 s, Failed to reach orbit, Failed to recover first stage","crew":[],"ships":[],"capsules":[],"payloads":["5eb0e4b6b6c3bb0006eeb1e2"],"launchpad":"5e9e4502f5090995de566f86","flight_number":2,"name":"DemoSat","date_utc":"2007-03-21T01:10:00.000Z","date_unix":1174439400,"date_local":"2007-03-21T13:10:00+12:00","date_precision":"hour","upcoming":false,"cores":[{"core":"5e9e289ef35918416a3b2624","flight":1,"gridfins":false,"legs":false,"reused":false,"landing_attempt":false,"landing_success":null,"landing_type":null,"landpad":null}],"auto_update":true,"tbd":false,"launch_library_id":null,"id":"5eb87cdaffd86e00

Normalized data:

```
In [10]: response = requests.get(static_json_url)
```

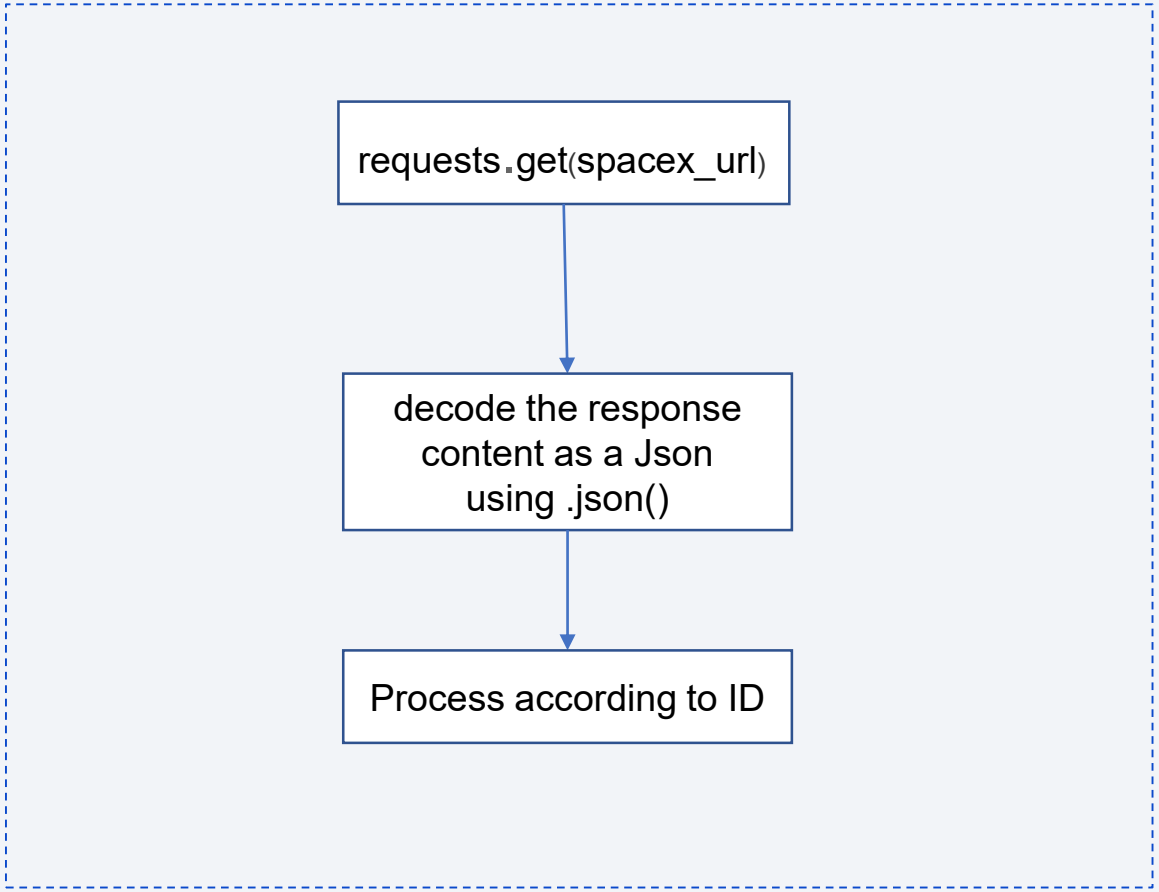
Now we decode the response content as a Json using . json() and turn it into a Pandas dataframe using . json_normalize()

```
In [11]: # Use json_normalize meethod to convert the json result into a dataframe = response
         data = pd.json_normalize(response.json())
         data.head()
```

Data Collection – SpaceX API

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

- GitHub URL:
  https://github.com/Jiahuielva/Applied-Data-Science-Capstone/blob/main/1-Collecting%20the%20data.ipynb



Data Collection - Scraping

- 1. request the HTML page

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

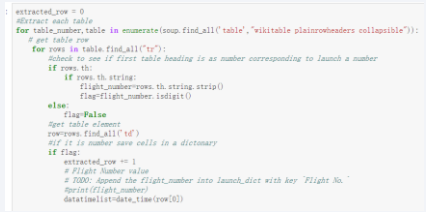
2. Create a BeautifulSoup object

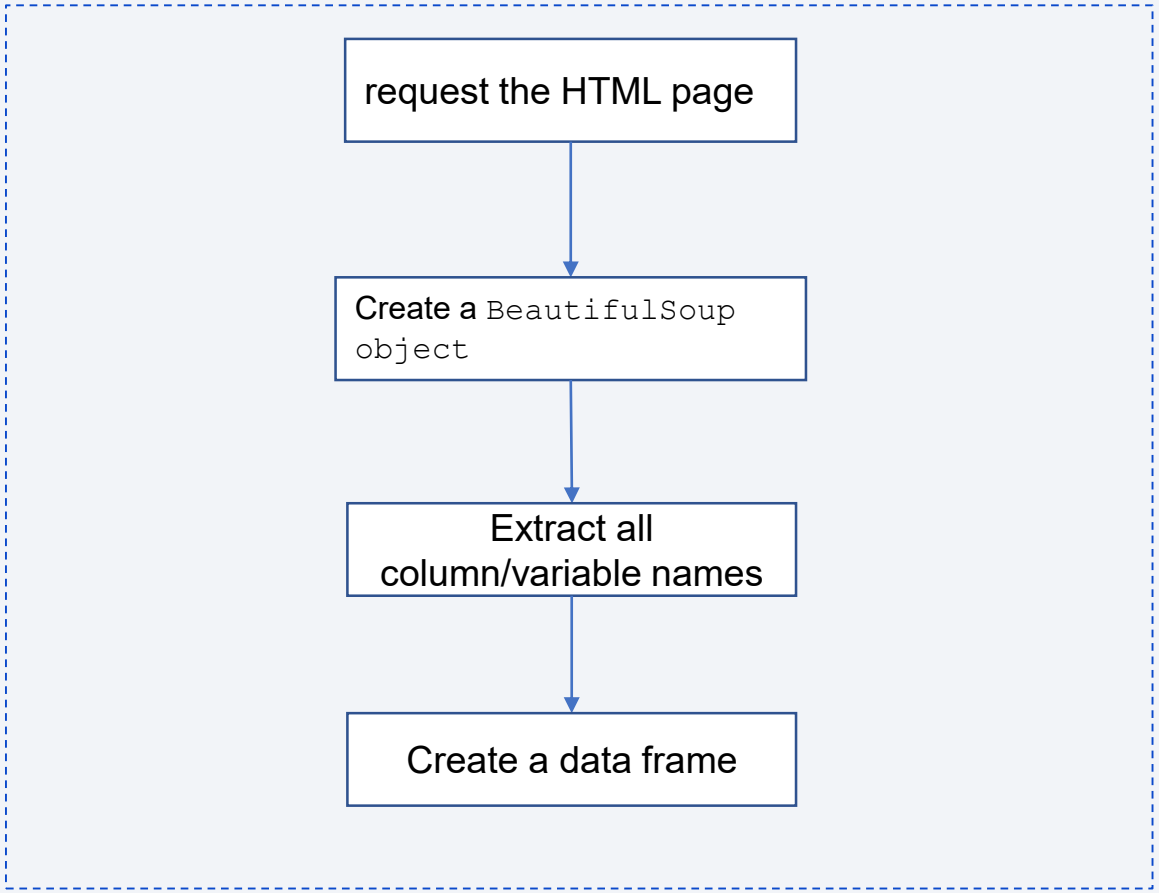
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text)
```

- 3. Extract all column/variable names from the HTML table header

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

- 4. *Create a data frame* by parsing the launch HTML tables

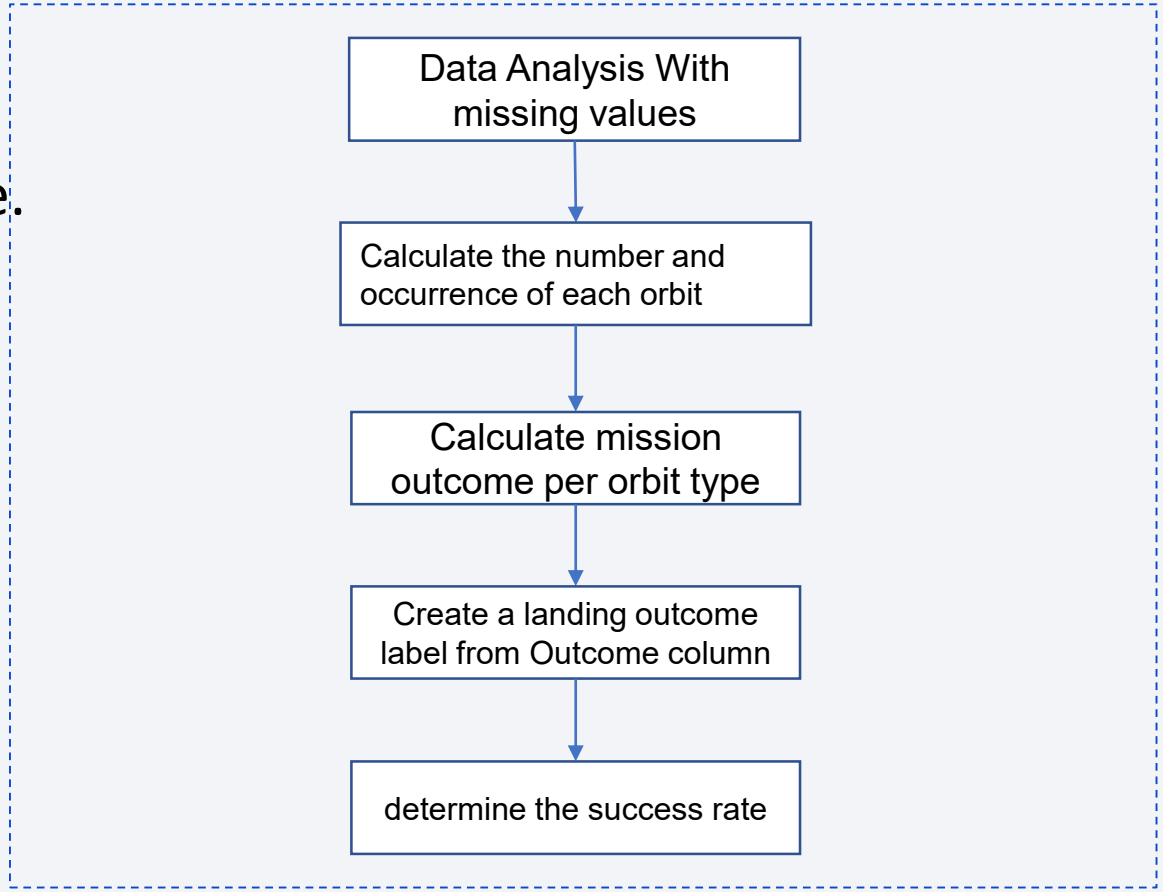




GitHub URL: https://github.com/Jiahuielva/Applied-Data-Science-Capstone/blob/main/2-%20Data%20Collection%20-%20Scraping.ipynb

Data Wrangling

- how data were processed
  - Identify and calculate the percentage of the missing values in each attribute.
  - And found that the landingpad column is seriously missing
  - So, We execute the flow chart on the right to make up for the missing values.



- GitHub URL:https://github.com/Jiahuielva/Applied-Data-Science-Capstone/blob/main/3%20-%20Data%20Wrangling.ipynb

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts

  - We use line chart and scatter plot etc. These plots are a good way to show the relationship.

- GitHub URL: https://github.com/Jiahuielva/Applied-Data-Science-Capstone/blob/main/4%20-%20Exploring%20and%20Preparing%20Data.ipynb

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed

```
%sql SELECT sum(payload_mass__kg_) FROM Spacex WHERE customer  = 'NASA (CRS)'
```

```
 * ibm_db_sa://gjd14388:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

1
45596

- Add the GitHub URL： https://github.com/Jiahuielva/Applied-Data-Science-Capstone/blob/main/5%20-%20eda-sql.ipynb

Build an Interactive Map with Folium

- I create a lot of things like， markers, circles, lines, etc. you created and added to a folium map
- I added these things to help me better locate the launch site.

- Add the GitHub URL: https://github.com/Jiahuielva/Applied-Data-Science-Capstone/blob/main/6-Folium.ipynb

Build a Dashboard with Plotly Dash

- **Build a Dashboard Application with Plotly Dash**
- building a Plotly Dash application for users to perform interactive visual analytics on SpaceX launch data in real-time.

- the GitHub URL: https://github.com/Jiahuielva/Applied-Data-Science-Capstone/blob/main/7-spacex_dash_app.py

Predictive Analysis (Classification)

- will create a machine learning pipeline to predict if the first stage will land given the data from the preceding labs. Find the method performs best using test data

- the GitHub URL: https://github.com/Jiahuielva/Applied-Data-Science-Capstone/blob/main/9-model%20Classification.ipynb

Results

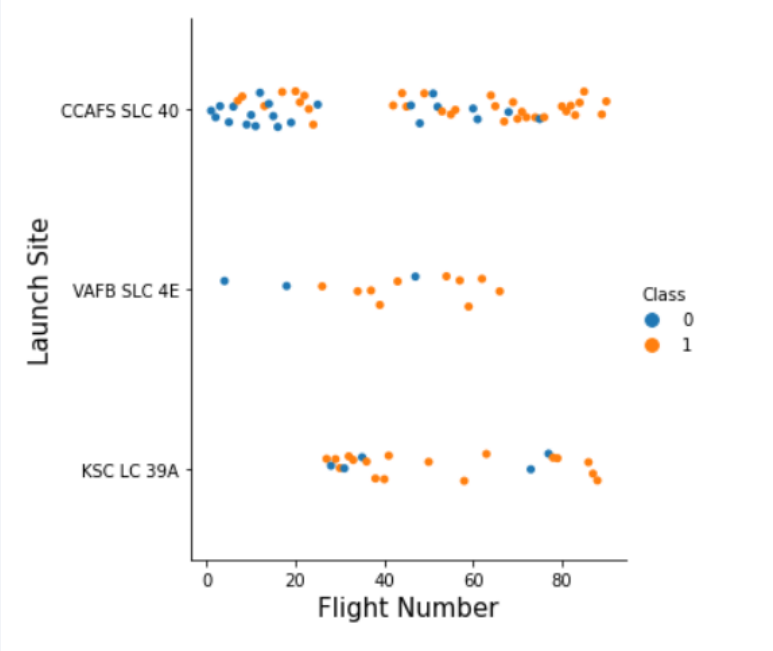
- Exploratory data analysis results
- Find best Hyperparameter for SVM, Classification Trees and Logistic Regression
- Find the method performs best using test data

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- a scatter plot of Flight Number vs. Launch Site

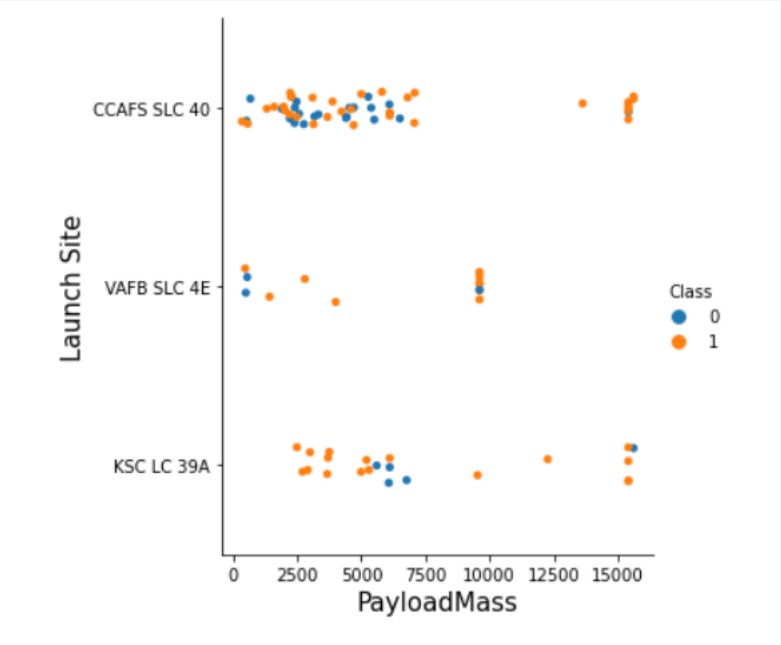


Class: represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

We see that different launch sites have different success rates. `KSC LC-39A` and `VAFB SLC 4E` has a success rate to be higher.

Payload vs. Launch Site

- A scatter plot of Payload vs. Launch Site

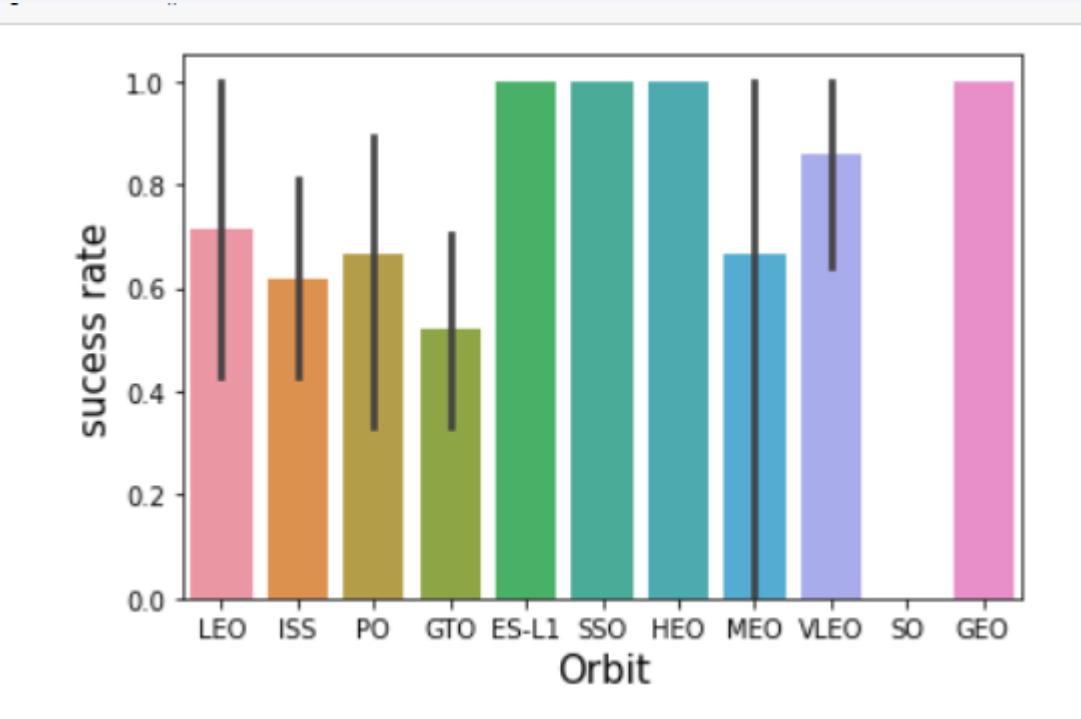


launch sites and their payload mass It doesn't matter.

In VAFB SLC 4E sites, The rocket launched has a smaller load than the other two.

Success Rate vs. Orbit Type

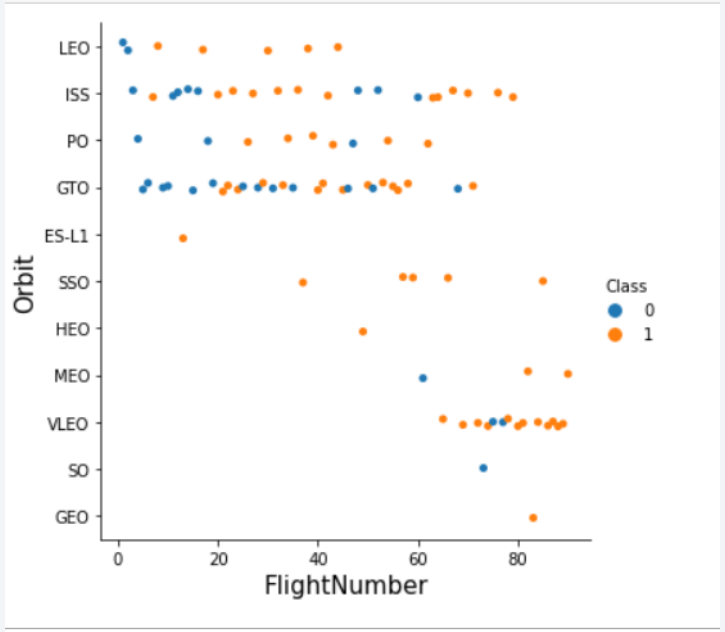
- Show a bar chart for the success rate of each orbit type



We found that 'Po', 'GTO', 'ISS' and' MEO 'had the lowest success rate.And 'ES-L1', 'SSO', 'HEO' and' GEO ' had 100% success rate.

Flight Number vs. Orbit Type

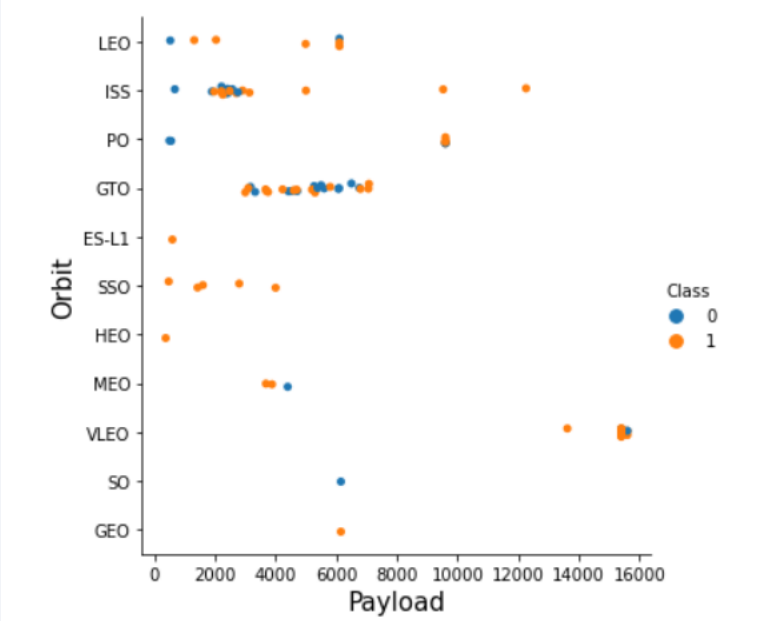
- Show a scatter point of Flight number vs. Orbit type



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

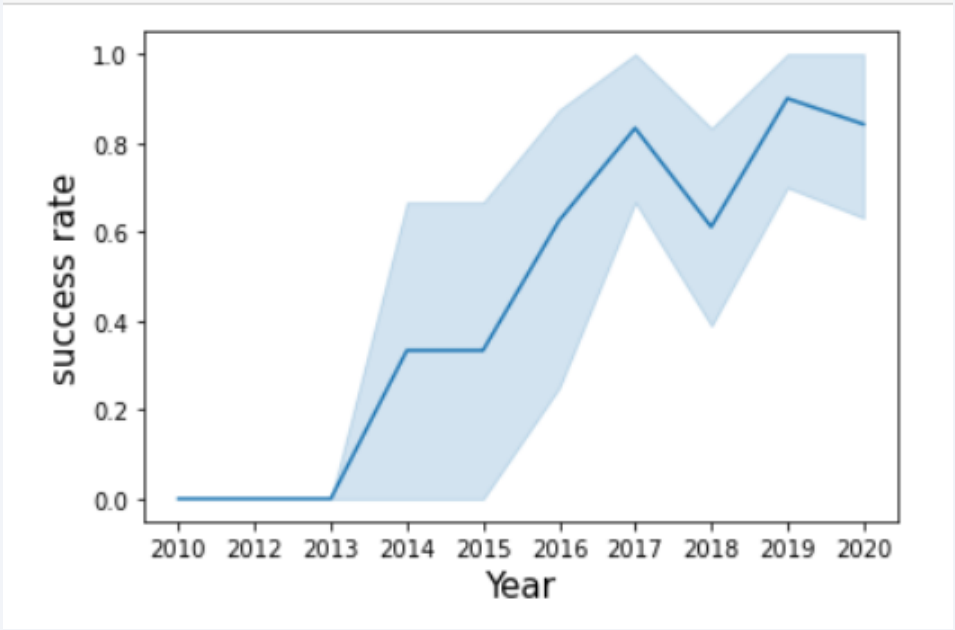
- Show a scatter point of payload vs. orbit type



Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

Launch Success Yearly Trend

- Show a line chart of yearly average success rate



can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

- Find the names of the unique launch sites

Display the names of the unique launch sites in the space mission

```
[4]: %sql select distinct(launch_site) from Spacex
```

```
 * ibm_db_sa://gjd14388:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

[4]:

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

There are only four launch sites,

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

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

- Use sql： SELECT * FROM Spacex WHERE launch_site LIKE 'CCA%' LIMIT 5

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

```
%sql SELECT sum(payload_mass__kg_) FROM Spacex WHERE customer  = 'NASA (CRS)'
```

```
 * ibm_db_sa://gjd14388:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

1
45596

- **total payload mass carried by boosters launched by NASA (CRS) is 45596KG**

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

```
[10]: %sql SELECT avg(payload_mass__kg_) FROM Spacex WHERE booster_version LIKE 'F9 v1.1%'
 * ibm_db_sa://gjd14388:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
[10]:    1
      2534
```

- **average payload mass carried by booster version F9 v1.1 is 2534KG**

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```
%sql SELECT MIN(DATE) FROM Spacext WHERE Landing_Outcome LIKE '%Success%'
```

```
 * ibm_db_sa://gjd14388:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.dat
Done.
```

1
2015-12-22

- the first data is 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

booster_version
F9 B4 B1040.1
F9 B4 B1043.1
F9 B5 B1046.2
F9 B5 B1047.2
F9 B5 B1048.3
F9 B5 B1051.2
F9 B5 B1058.2
F9 B5B1060.1
F9 B5B1062.1
F9 FT B1021.2
F9 FT B1031.2
F9 FT B1022
F9 FT B1026
F9 FT B1032.1

USE SQL：SELECT DISTINCT booster_version FROM Spacext WHERE Landing_Outcome LIKE '%Success%' AND payload_mass__kg_>4000 AND payload_mass__kg_<6000

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
[45]: %sql SELECT mission_outcome,COUNT(*) FROM Spacext GROUP BY mission_outcome
 * ibm_db_sa://gjd14388:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

[45]:

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

- Launch success rate is high

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

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

```
[48]: %sql SELECT booster_version FROM Spacext WHERE payload_mass__kg_ = ( SELECT MAX(payload_mass__kg_) FROM Spacext  )
```

```
 * ibm_db_sa://gjd14388:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

[48]:

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

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

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

```
[63]: %sql SELECT DATE,Landing_Outcome,booster_version,launch_site FROM Spacext WHERE Landing_Outcome LIKE '%drone ship%' AND YEAR(DATE) =2015
```

```
 * ibm_db_sa://gjd14388:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

[63]:

DATE	landing_outcome	booster_version	launch_site
2015-01-10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015-04-14	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40
2015-06-28	Precluded (drone ship)	F9 v1.1 B1018	CCAFS LC-40

THREE rows of data， few and far between

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

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

```
[70]: %sql SELECT Landing_Outcome,count(*) FROM Spacext WHERE DATE>='2010-06-04' AND DATE<='2017-03-20' GROUP BY Landing_Outcome
 * ibm_db_sa://gjd14388:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

[70]:

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

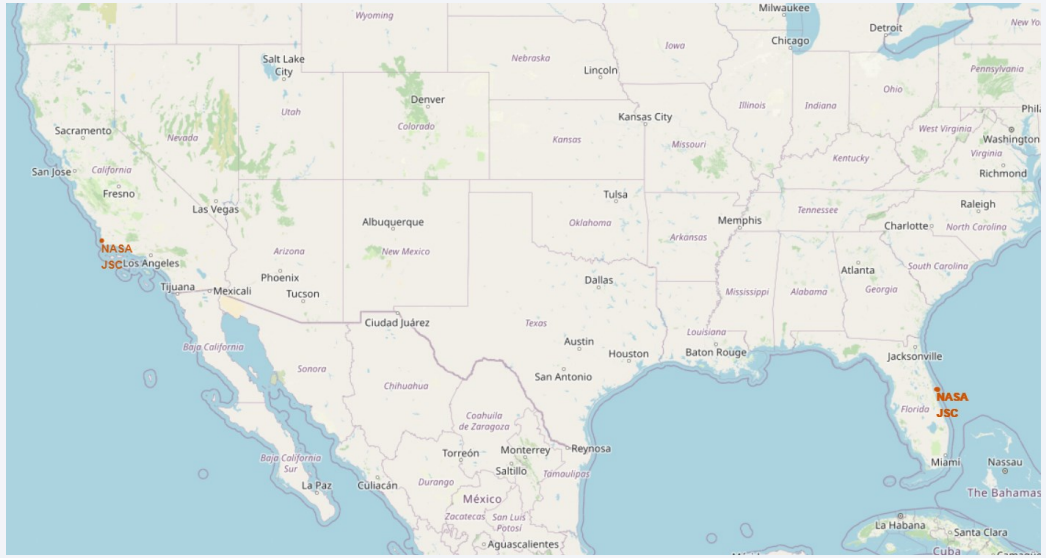
Grouped according to the different types of landings are as follows.

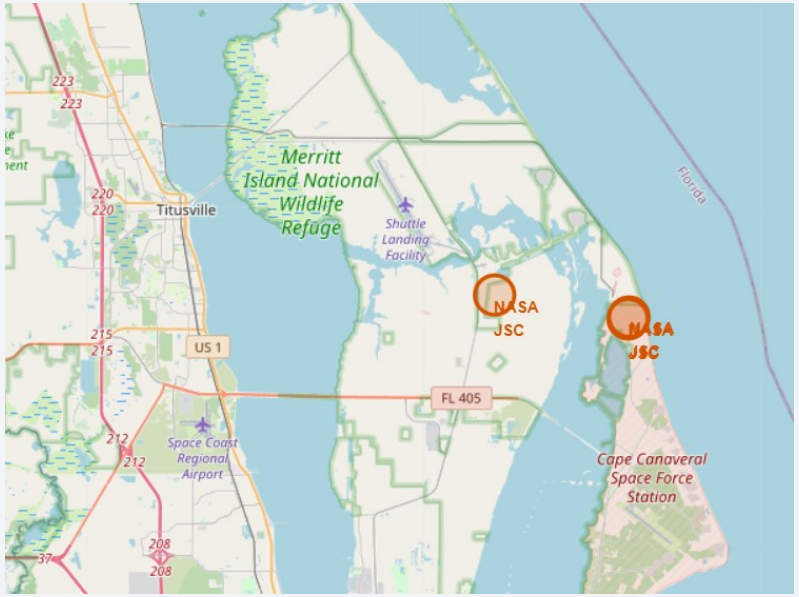
Section 4

Launch Sites Proximities Analysis

Mark all launch sites on a map

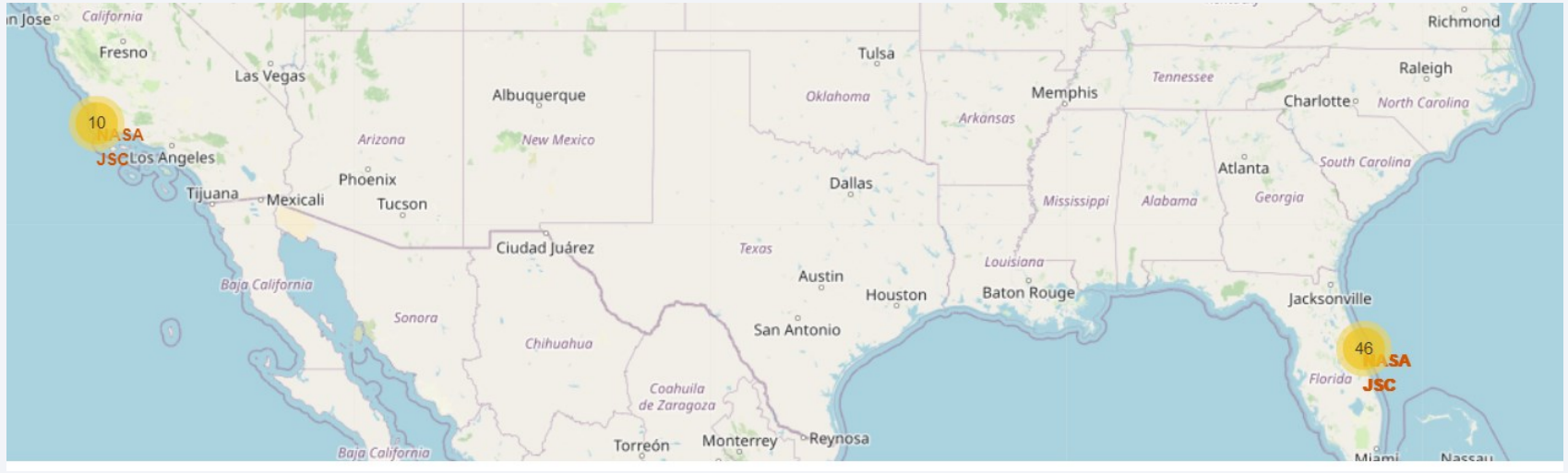
- Explore the generated folium map and make a proper screenshot to include all launch sites' location markers on a global map

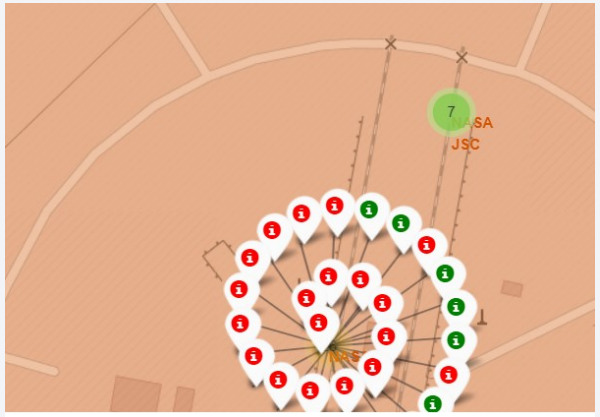




There are four launch sites, as shown in the figure, three of which are located next to each other.

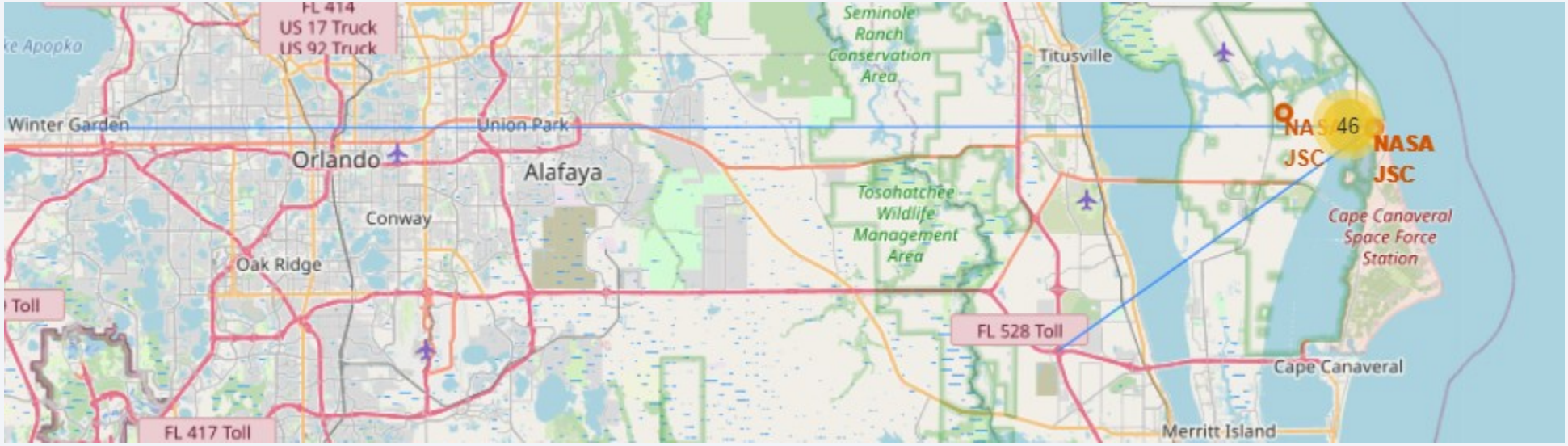
Mark the success/failed launches for each site on the map

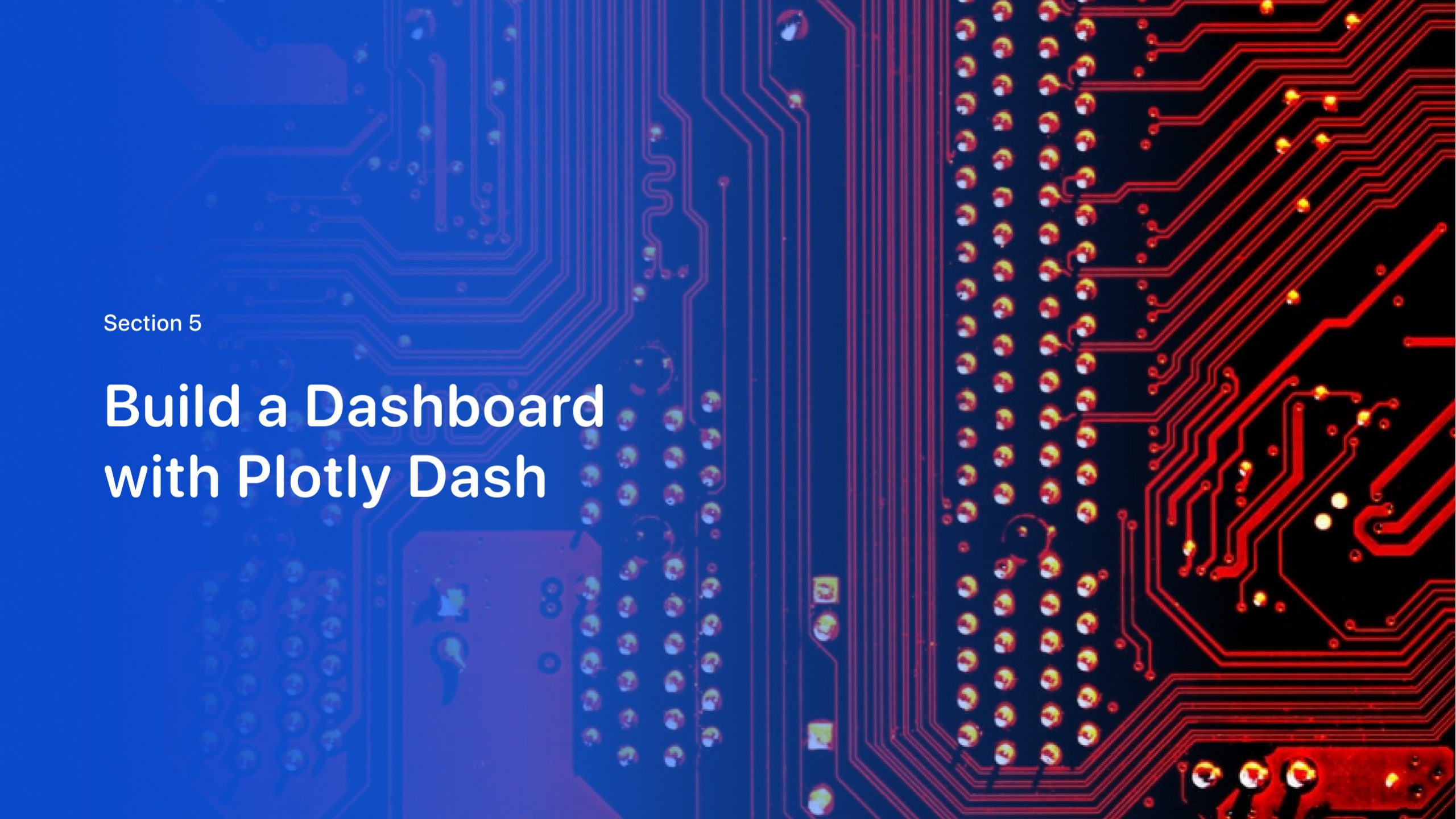




The distribution of each number is shown in the figure

Calculate the distances between a launch site to its proximities



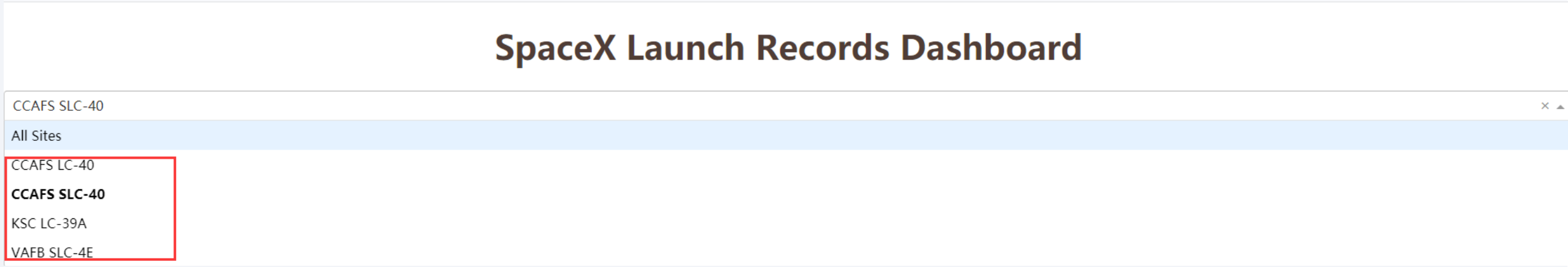


Section 5

Build a Dashboard with Plotly Dash

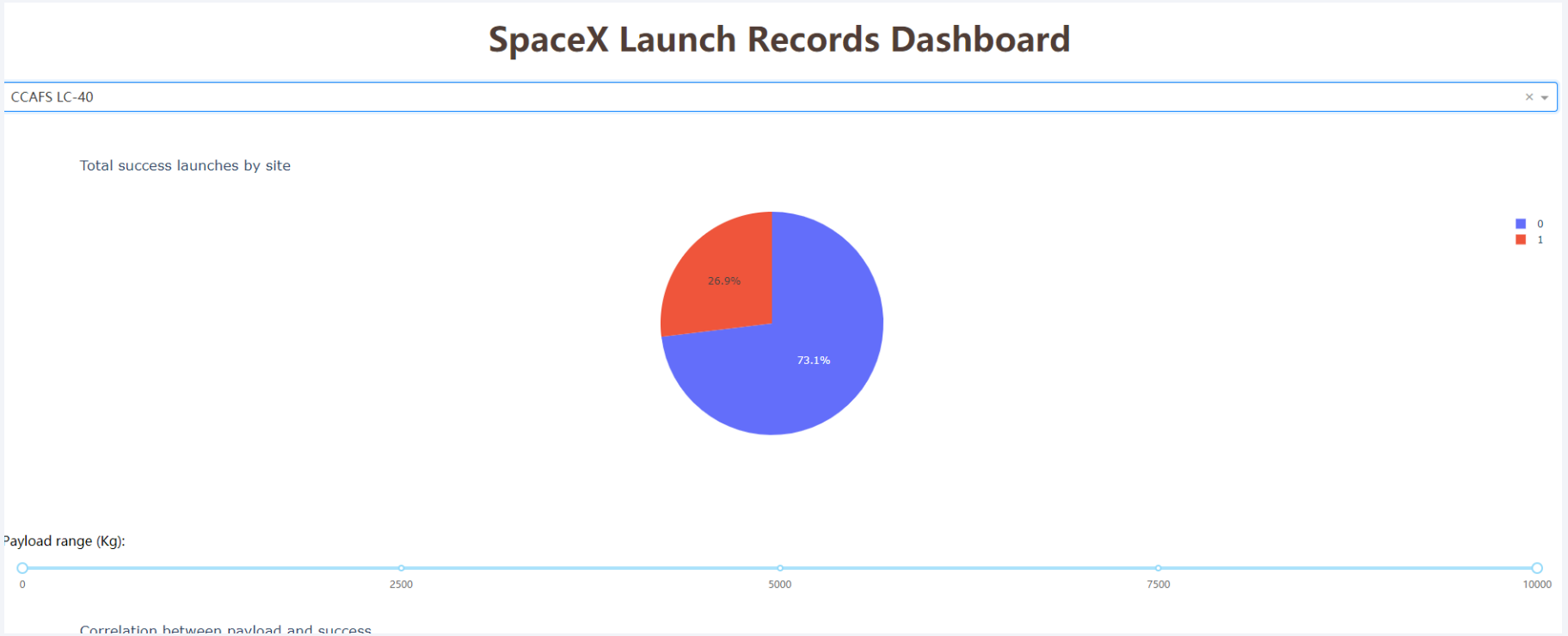
Add a Launch Site Drop-down Input Component

- We have four different launch sites and we would like to first see which one has the largest success count. Then, we would like to select one specific site and check its detailed success rate (class=0 vs. class=1).



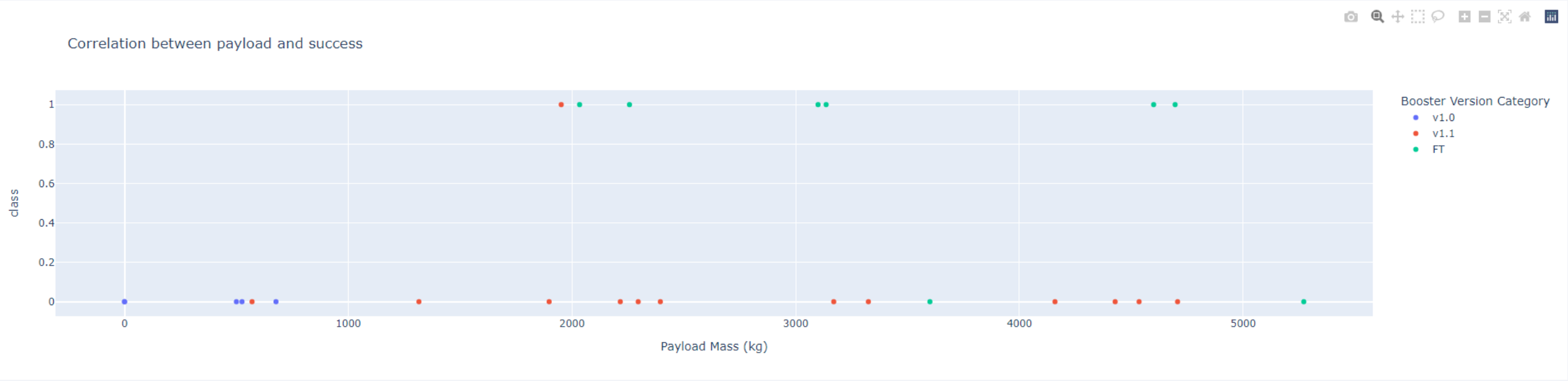
render success-pie-chart based on selected site dropdown

- The general idea of this callback function is to get the selected launch site from site-dropdown and render a pie chart visualizing launch success counts.



render the success-payload-scatter-chart scatter plot

- Next, we want to plot a scatter plot with the x axis to be the payload and the y axis to be the launch outcome (i.e., class column). As such, we can visually observe how payload may be correlated with mission outcomes for selected site(s).

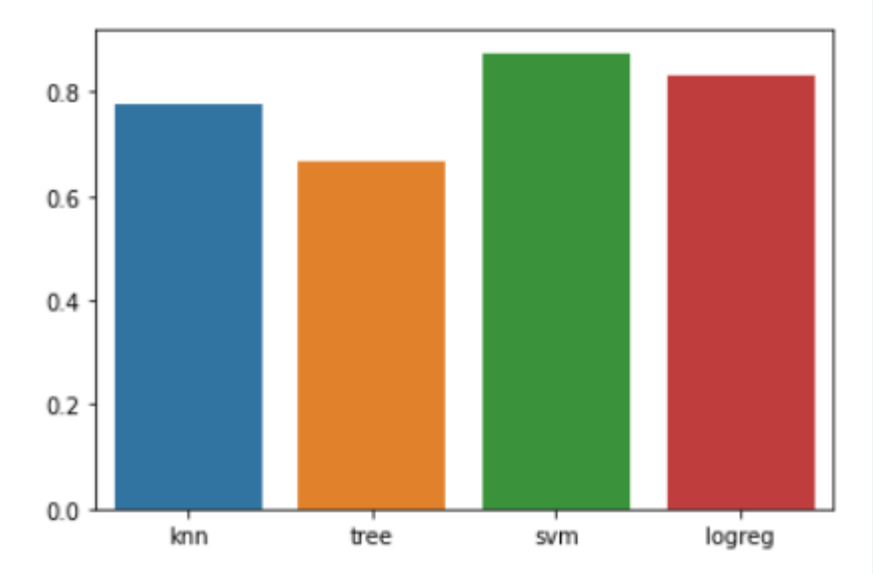


Section 6

Predictive Analysis (Classification)

Classification Accuracy

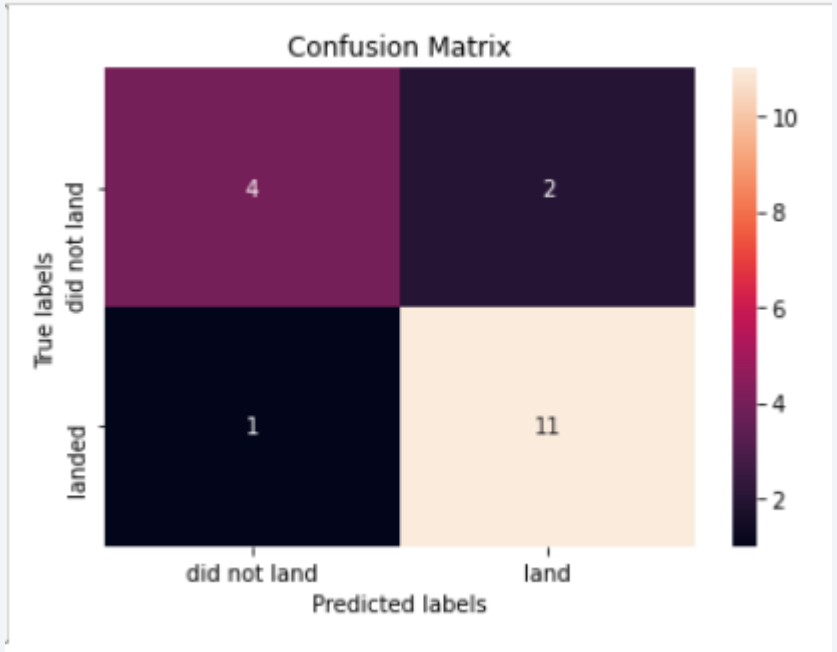
- Visualize the built model accuracy for all built classification models, in a bar chart



SVM has the highest classification accuracy, is 0.833%

Confusion Matrix

- Show the confusion matrix of the best performing model



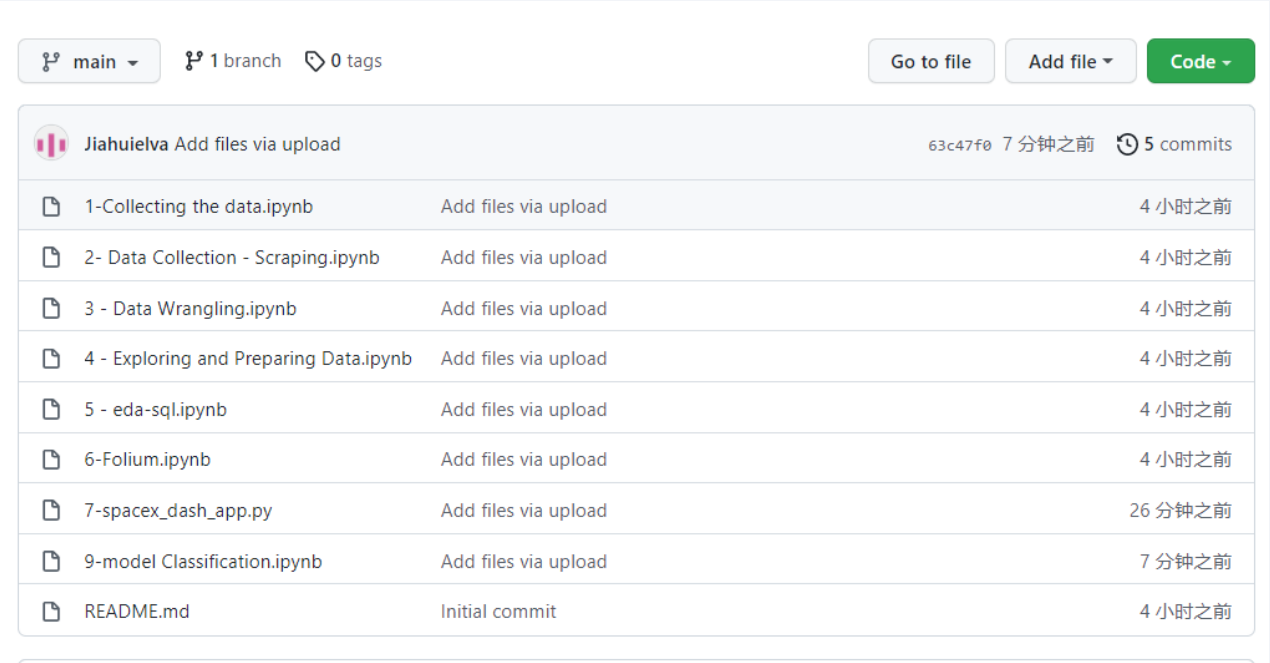
They're all on diagonal lines, so they're pretty accurate.

Conclusions

- SVM is the best classifier for the current task
- That gives us an idea of SPACEX's launch
- Better data analysis allows us to understand the content more intuitively

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



GitHub URL: https://github.com/Jiahuielva/Applied-Data-Science-Capstone

Thank you!