IBM Developer
SKILLS NETWORK

Daniel Herrera

12/09/2021

Outline

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

Executive Summary

The following methodologies were used for this analysis

- Data Collection (SpaceX API and Web Scraping)
- Data Wrangling
- EDA with Data Visualization
- EDA with SQL
- Interactive Map of launch sites
- Dashboard with plotly dash
- Predictive Analysis

Introduction

This project requires predicting the success rate of future Falcon 9 launches. In this way, the company can more accurately estimate the launch costs for each rocket.

Information will be collected from the SpaceX API and using Web Scraping on Wikipedia articles.
Later, all the raw data obtained will be processed in order to build accurate classification models.

Section 1

Methodology

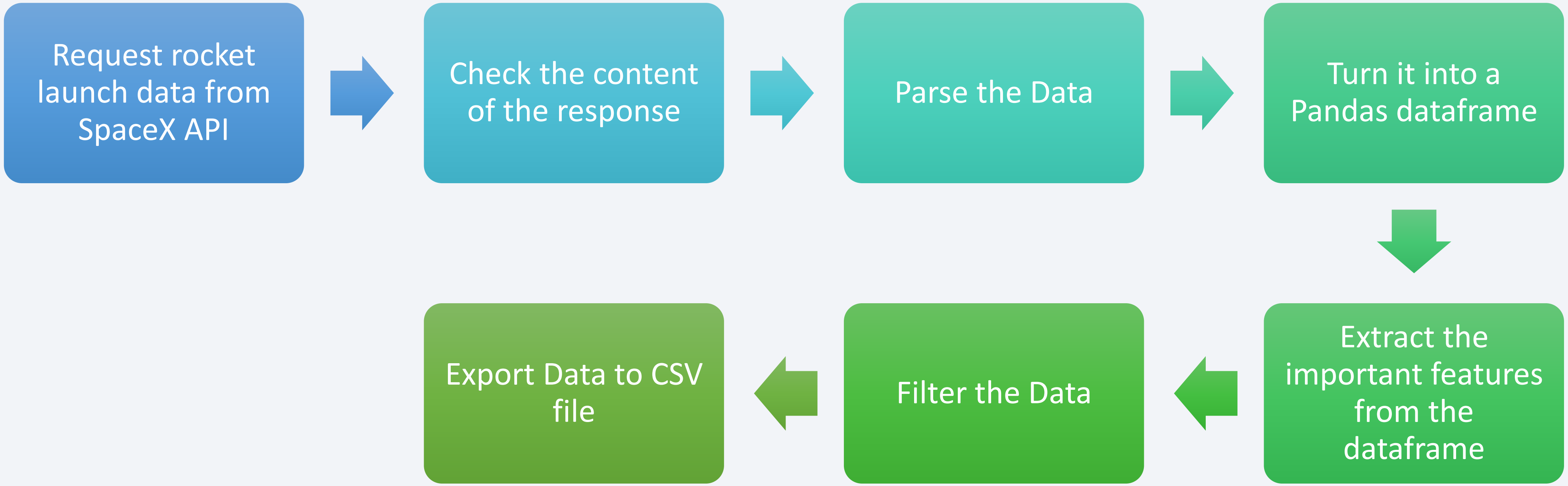
Data Collection

The data was mainly collected from the SpaceX API and using web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy launches

1. Data Collection from API and Web Scraping
2. Raw Data Checking
3. Organizing of Data
4. Data Processing
5. Export Data to CSV file

Data Collection - SpaceX API

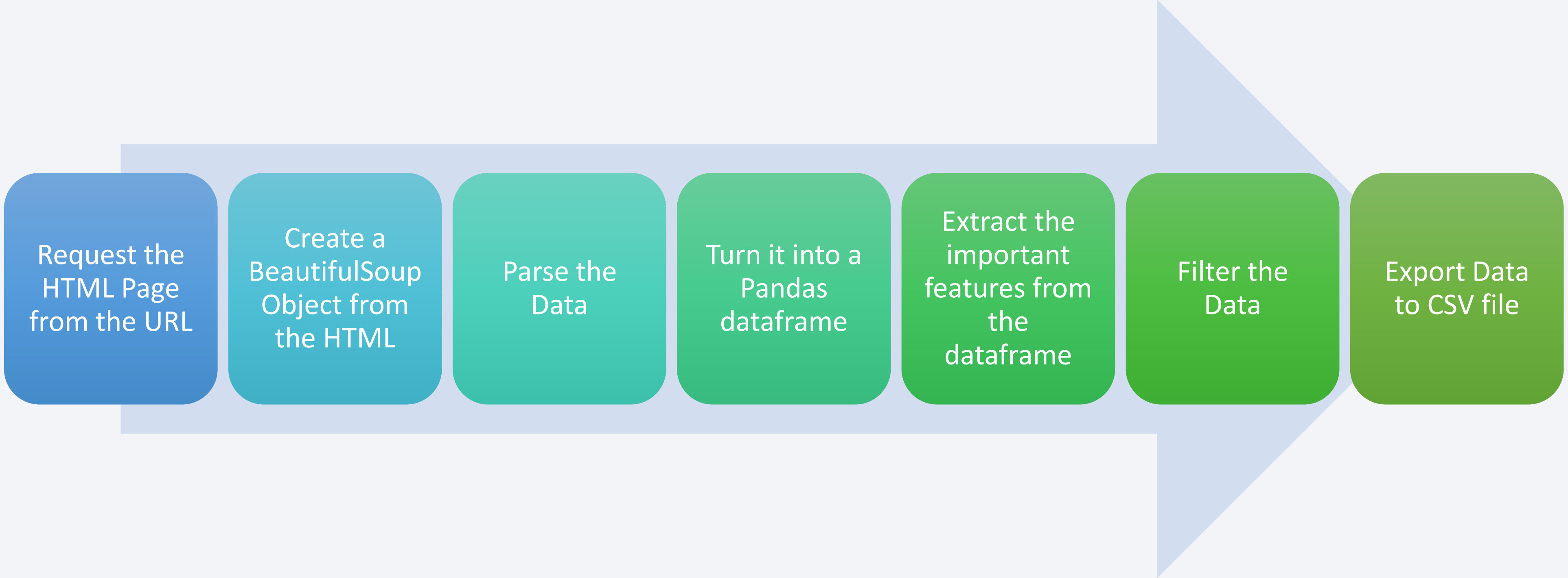
In this stage we will make a get request to the SpaceX API and clean the request data. The steps of the stage it can be summarized in the flowchart below



GitHub URL of the Notebook:
https://github.com/danielh27/Capstone/blob/master/Data%20Collection%20API.ipynb

Data Collection - Scraping

Web scraping was used to collect Falcon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy launches. Data it is extracted from tables in the html code then is parsed and converted into a Pandas data frame.



GitHub URL of the Notebook:
https://github.com/danielh27/Capstone/blob/master/Data%20Collection%20with%20Web%20Scraping.ipynb

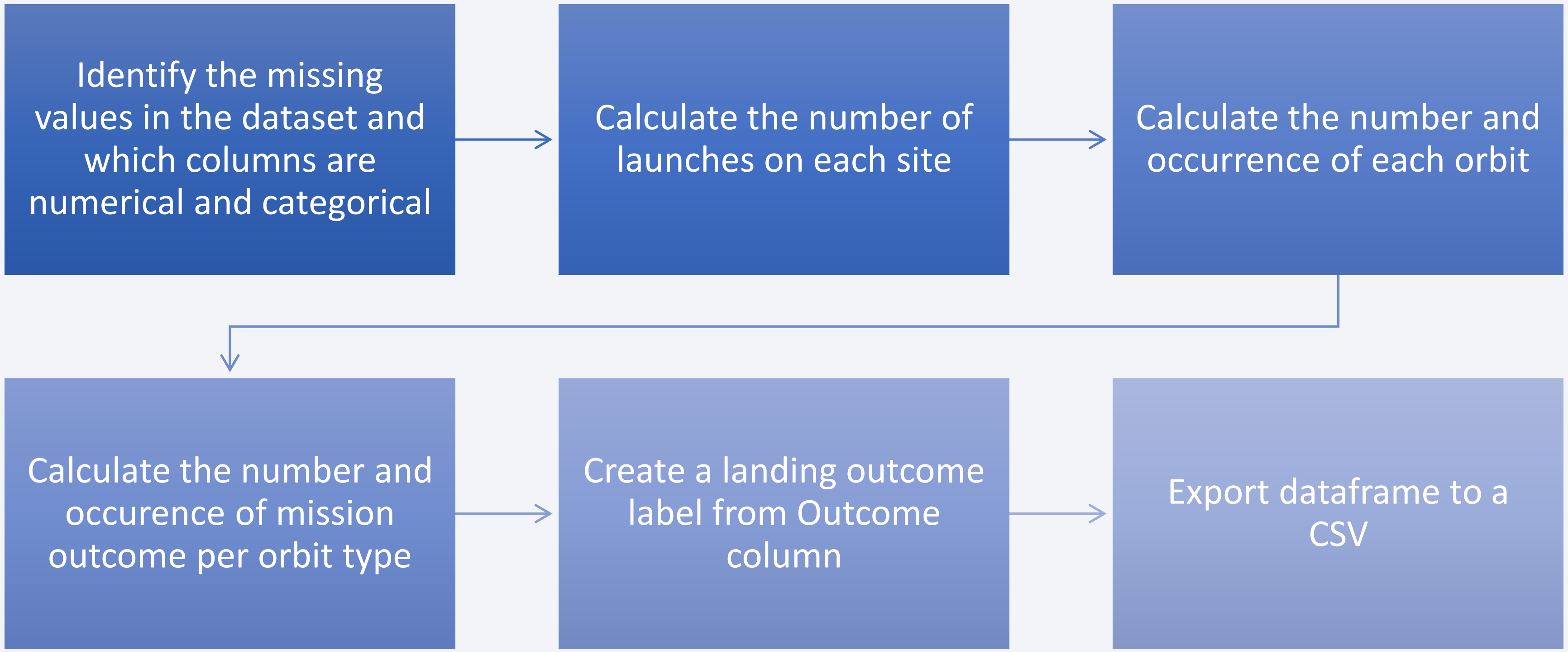
Data Wrangling

In this stage, we performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training 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” means the mission outcome was successfully landed while “False” means the mission outcome was unsuccessfully landed.

We mainly converted those outcomes into Training Labels with 1 means the booster successfully landed and 0 means it was unsuccessful.

Data Wrangling

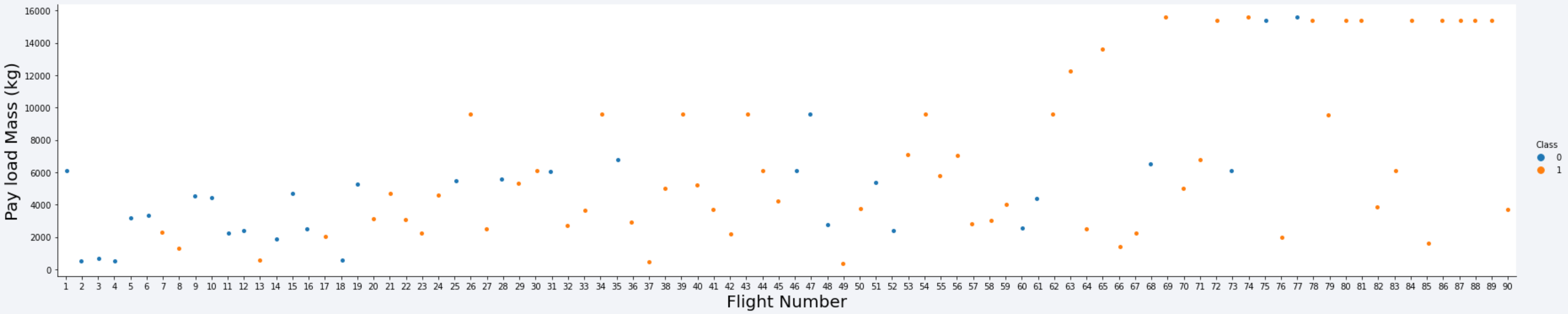
The steps of the stage it can be summarized in the flowchart below



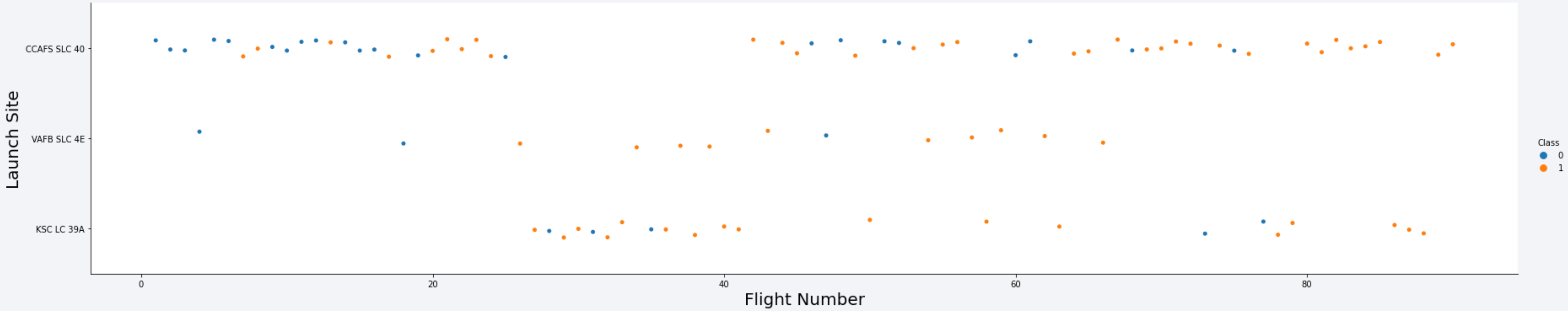
GitHub URL of the Notebook:
https://github.com/danielh27/Capstone/blob/master/EDA.ipynb

EDA with Data Visualization

Payload Mass VS Flight Number

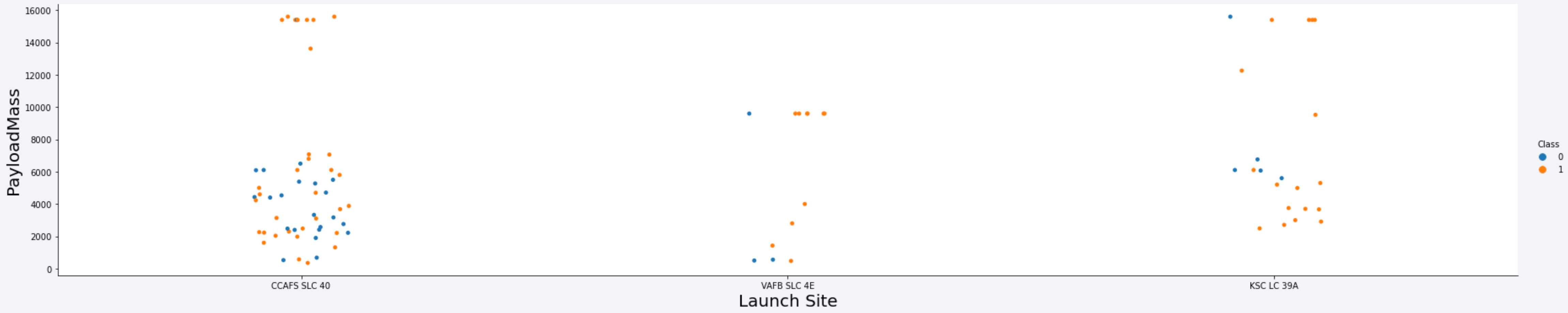


Launch Site VS Flight Number

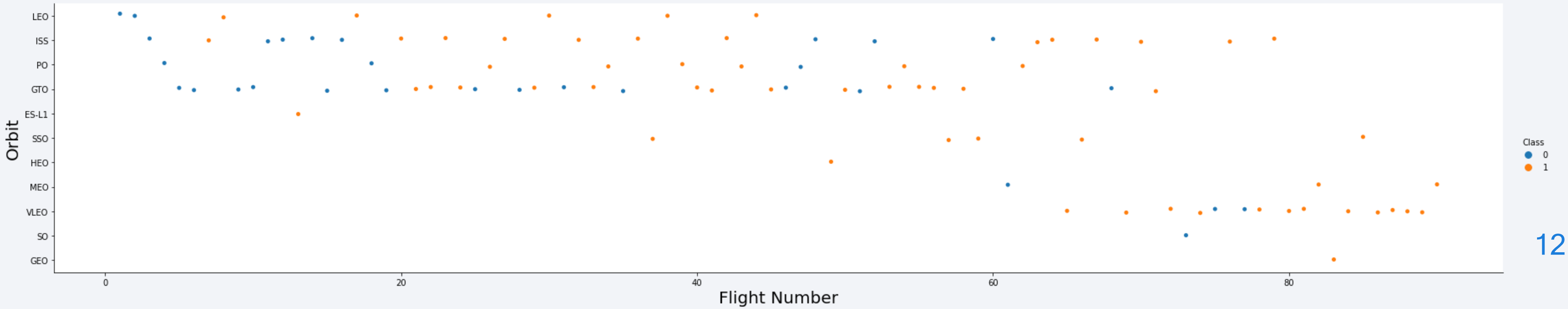


EDA with Data Visualization

Payload Mass VS Launch Site

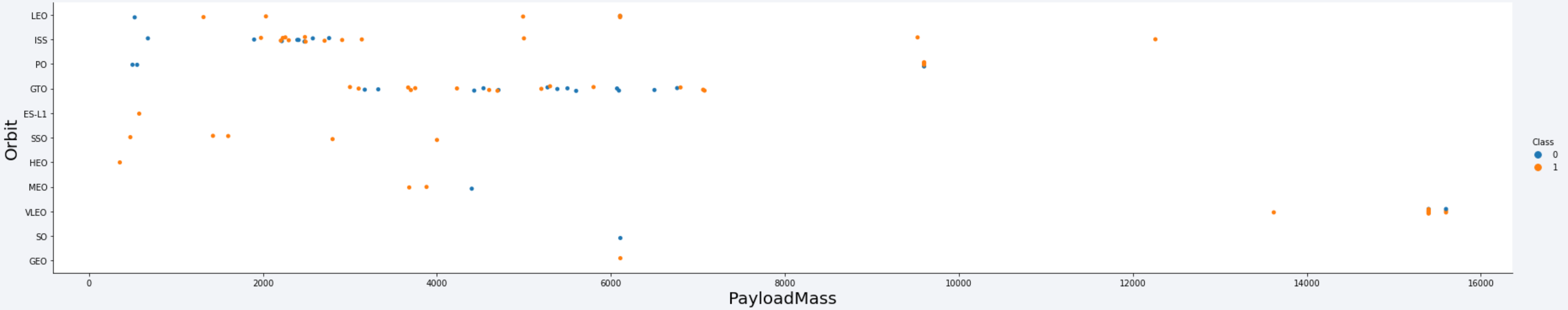


Orbit VS Flight Number



EDA with Data Visualization

Orbit VS Payload Mass



GitHub URL of the Notebook:
https://github.com/danielh27/Capstone/blob/master/EDA%20with%20Data%20Visualization.ipynb

EDA with SQL

- %sql select distinct(LAUNCH_SITE) from SPACEXDATASET
- %sql select * from SPACEXDATASET where LAUNCH_SITE like 'CCA%' limit 5
- %sql select SUM(payload_mass__kg_) from SPACEXDATASET where CUSTOMER = 'NASA (CRS)‘
- %sql select avg(payload_mass__kg_) from SPACEXDATASET where booster_version LIKE 'F9 v1.1%‘
- %sql select min(date) from SPACEXDATASET where landing__outcome = 'Success (ground pad)‘
- %sql select * from SPACEXDATASET where landing__outcome = 'Success (drone ship)' and (payload_mass__kg_ between 4000 and 6000)
- %sql select mission_outcome, COUNT(*) from SPACEXDATASET group by mission_outcome
- %sql select booster_version, payload_mass__kg_ from SPACEXDATASET where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXDATASET)
- %sql select date, booster_version, launch_site, landing__outcome from SPACEXDATASET where landing__outcome = 'Failure (drone ship)' and (date like '2015%')
- %sql select landing__outcome,count(landing__outcome) as landing_outcome_COUNT from SPACEXDATASET where DATE between '2010-06-04' and '2017-03-20' group by landing__outcome

GitHub URL of the Notebook:
https://github.com/danielh27/Capstone/blob/master/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

For this stage a map of the USA was used. Subsequently, markers for the different launch site locations were created and added. Names were also added to the markers to be able to identify them.

A characteristic that can be highlighted is that the places are differentiated by color, in this way if a launch was successful or failed you can know.

On the map you can also see the distance between each launch site and also if there are any railway near those sites.

GitHub URL of the Notebook:
https://github.com/danielh27/Capstone/blob/master/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

Build a Dashboard with Plotly Dash

Two charts were used for the dashboard, the first is a pie chart that shows the number of successful launches depending on the place. The second chart is a scatter chart that shows the relationship between the payload mass and the outcome for the launch site.

At the top of the dashboard there is a drop-down menu to be able to select the launch site that will display the pie chart. Below the pie chart is a slider to select the payload mass range. Both graphs are dynamic and change the information in real time thanks to the callback functions.

GitHub URL of the Python File:
https://github.com/danielh27/Capstone/blob/master/Dashboard%20with%20Plotly%20Dash.py

Predictive Analysis (Classification)

For this stage, the dataset was standardized with all the information, then all the data was separated so that different samples could be used both for training and for tests.

We proceeded to create different types of objects to be able to run the tests in the different methods. The same samples were used to test the different models to ensure reliability in the results.

The models tested were: Logistic Regression, Support Vector Machine, decision tree and k nearest neighbors

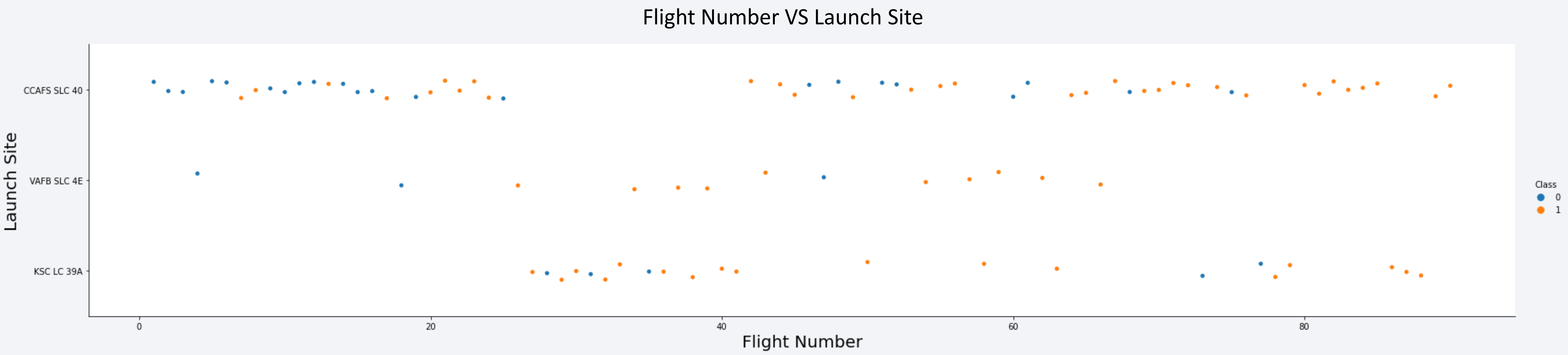
After testing them all it was possible to conclude that none has a great difference with respect to another, practically all these algorithms give the same result

GitHub URL of the Notebook:
https://github.com/danielh27/Capstone/blob/master/Machine%20Learning%20Prediction.ipynb

Section 2

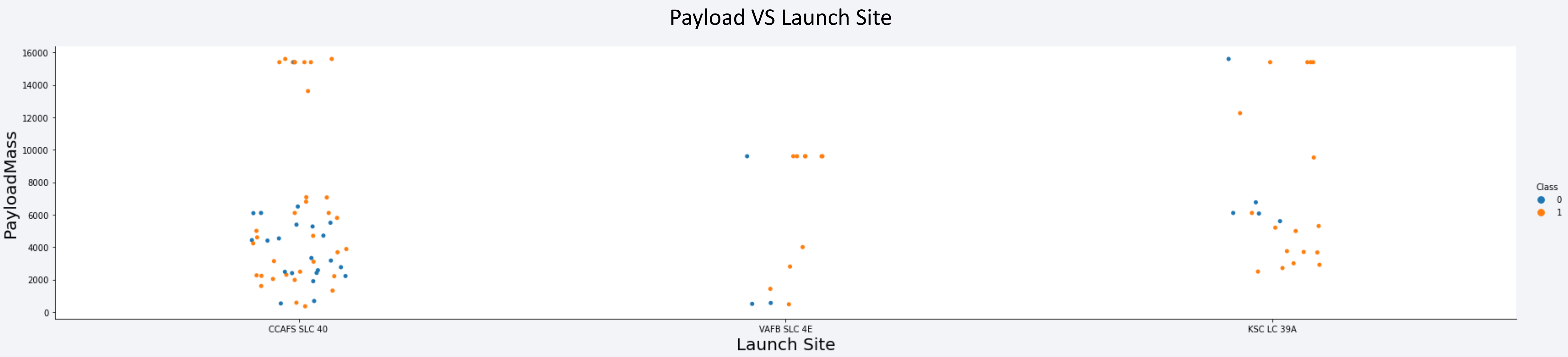
Insights drawn from EDA

Flight Number vs. Launch Site



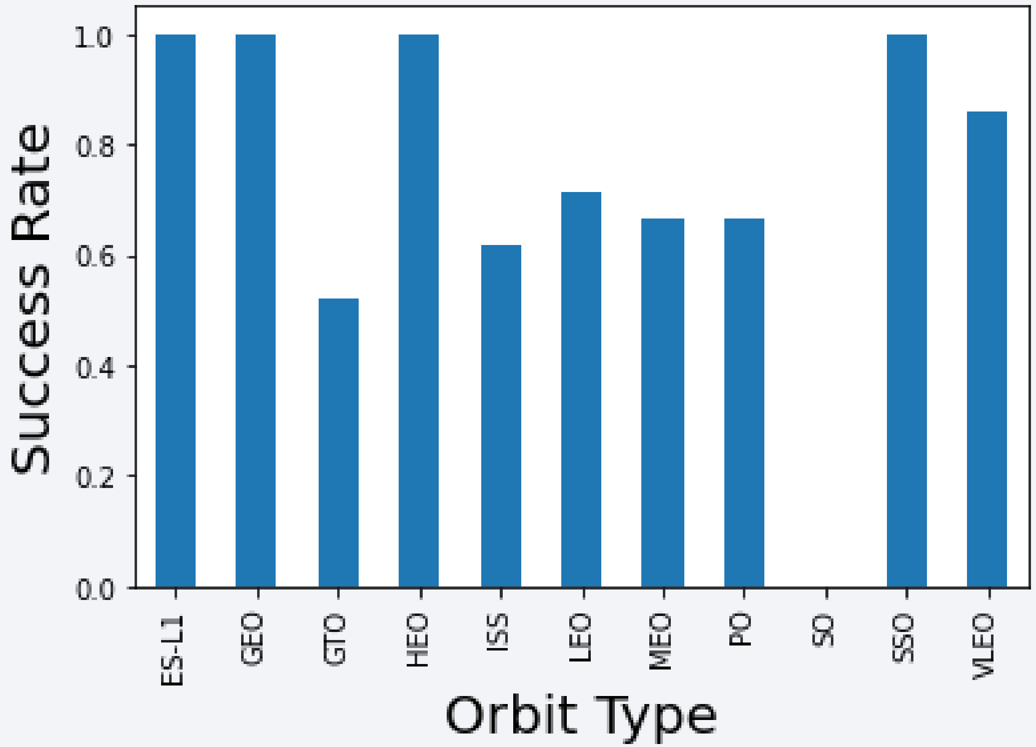
In this plot we can see the relationship between the two features, it can be seen how the success rate increases as the number of flights increases.

Payload vs. Launch Site



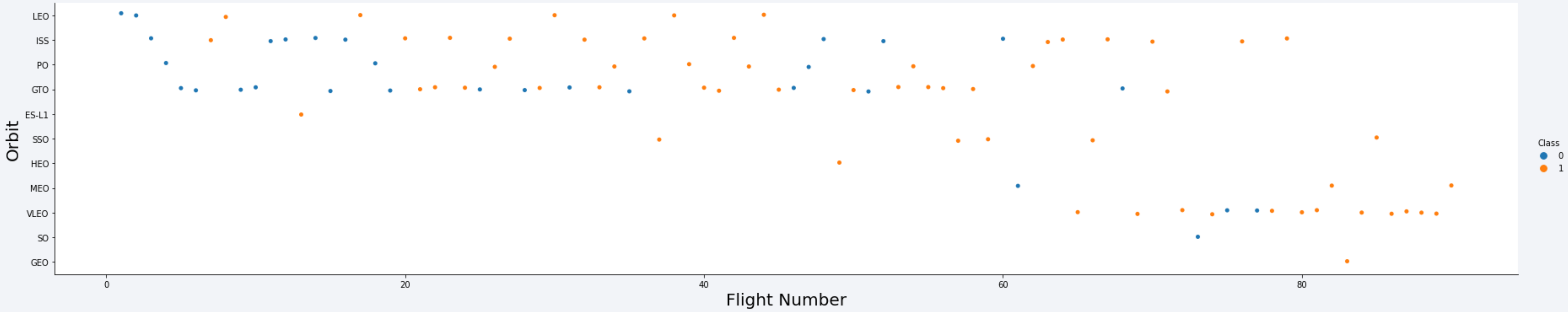
In this plot you can see how the payload is related to the launch success rate.

Success Rate vs. Orbit Type



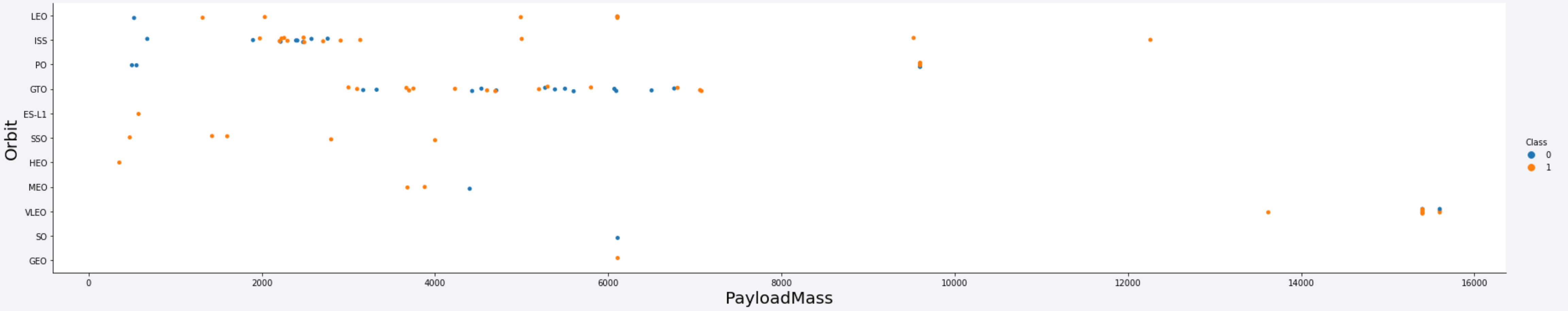
It can be seen that the ES-L1, GEO, HEO and SSO orbits have a higher success rate than the other orbits.

Flight Number vs. Orbit Type



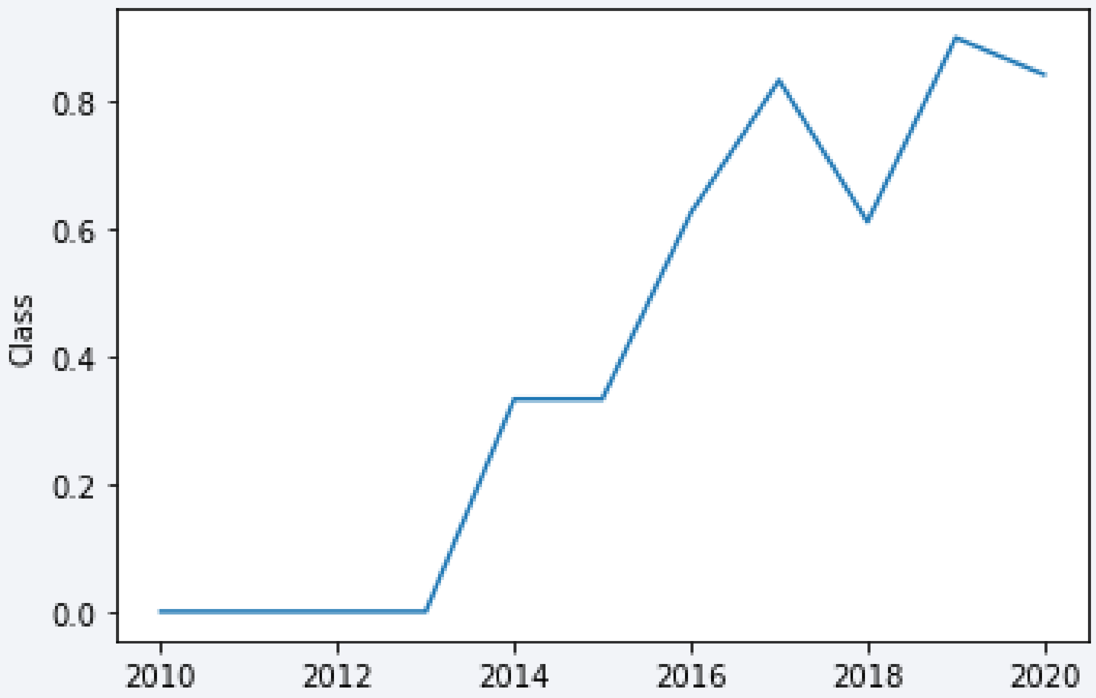
In the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type



It can be seen that Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

Launch Success Yearly Trend



It can be seen that the success rate since 2013 kept increasing till 2020

All Launch Site Names

```
In [4]: %sql select distinct(LAUNCH_SITE) from SPACEXDATASET

 * ibm_db_sa://qmf11403:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.

Out[4]:
```

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

You can see the 5 launch sites

Launch Site Names Begin with 'CCA'

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

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

```
 * ibm_db_sa://qmf11403:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/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

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

In [6]:
```
%%sql select SUM(payload_mass__kg_) from SPACEXDATASET where CUSTOMER = 'NASA (CRS)'
```

```
 * ibm_db_sa://qmf11403:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[6]:

1
45596

Average Payload Mass by F9 v1.1

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

```
In [7]: %%sql select avg(payload_mass__kg_) from SPACEXDATASET where booster_version LIKE 'F9 v1.1%'

 * ibm_db_sa://qmf11403:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.

Out[7]:
```

1
2534

First Successful Ground Landing Date

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

Hint:Use min function

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

 * ibm_db_sa://qmf11403:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[8]:

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

In [9]:
```
%%sql select * from SPACEXDATASET where landing__outcome = 'Success (drone ship)' and (payload_mass__kg_ between 4000 and 6000)
```

```
 * ibm_db_sa://qmf11403:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[9]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2016-05-06	05:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-08-14	05:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
2017-10-11	22:53:00	F9 FT B1031.2	KSC LC-39A	SES-11 / EchoStar 105	5200	GTO	SES EchoStar	Success	Success (drone ship)

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

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

```
 * ibm_db_sa://qmf11403:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[10]:

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

Boosters Carried Maximum Payload

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

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

```
 * ibm_db_sa://qmf11403:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[11]:

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

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

```
 * ibm_db_sa://qmf11403:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[12]:

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

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

In [23]:
```
%sql SELECT LANDING__OUTCOME,count(LANDING__OUTCOME)as LANDING_OUTCOME_COUNT from SPACEXDATASET where DATE between '2010-06-04' and '2017-03-20' group by LANDING__OUTCOME
```

```
 * ibm_db_sa://qmf11403:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

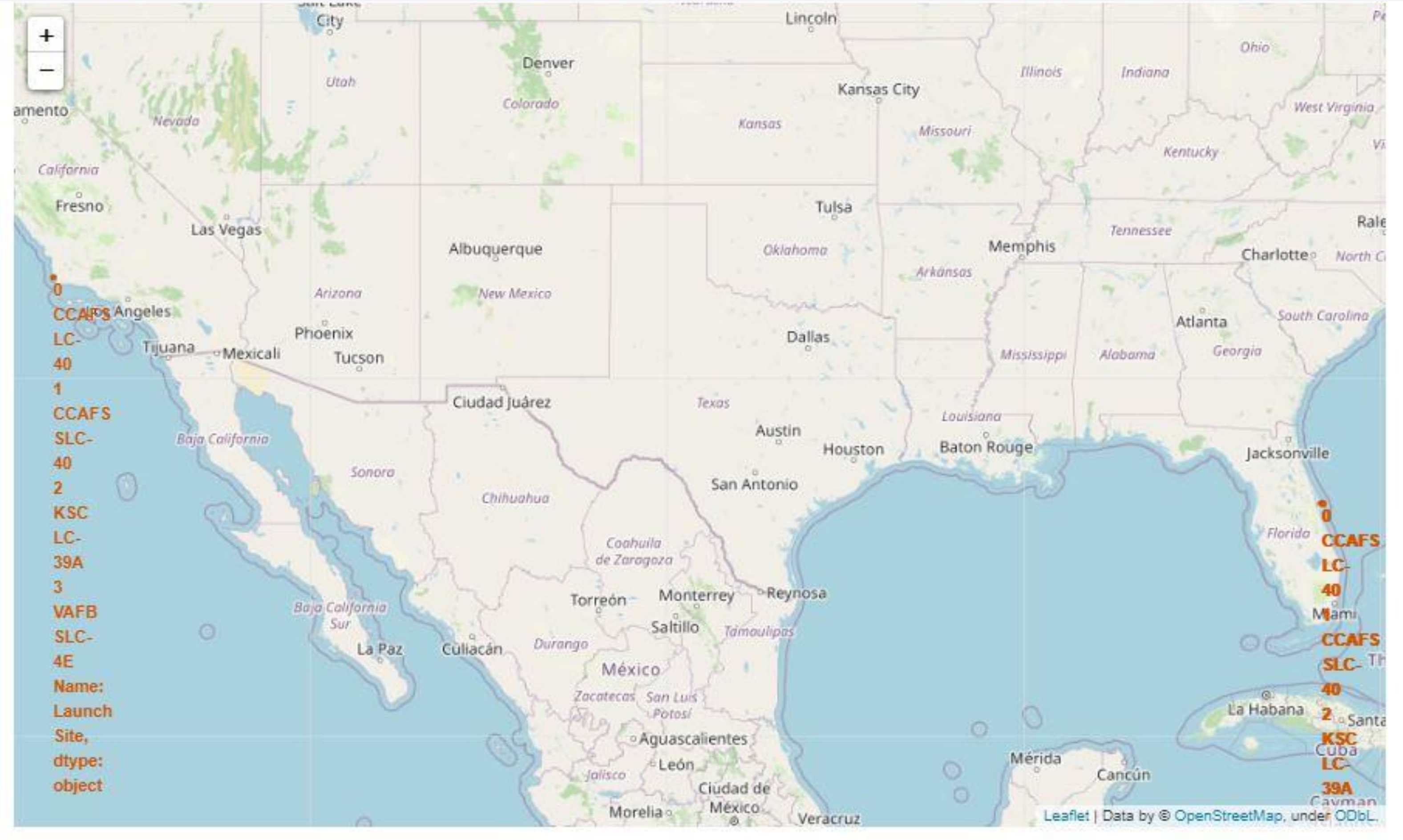
Out[23]:

landing__outcome	landing_outcome_count
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

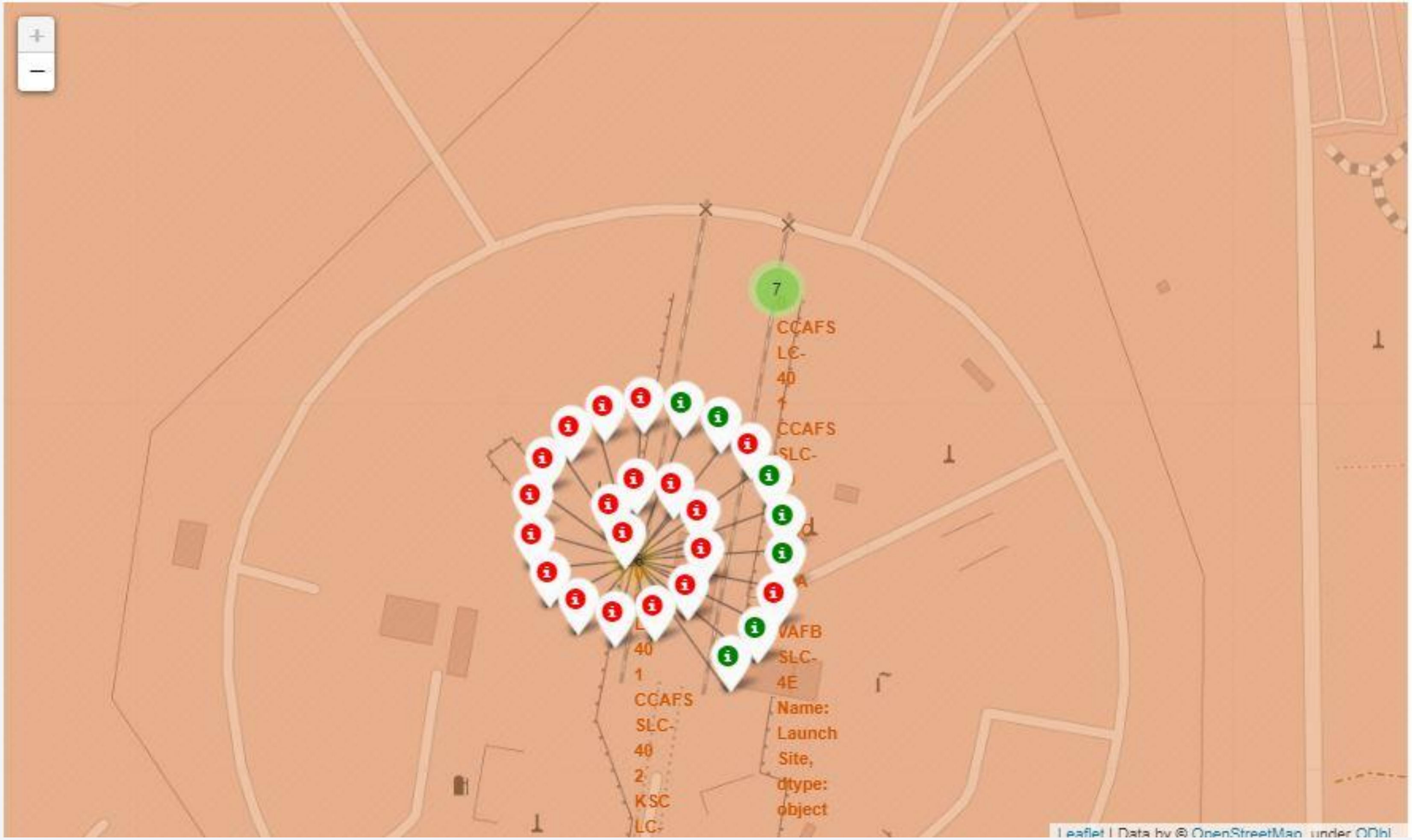
Section 4

Launch Sites Proximities Analysis

Launch Sites Map



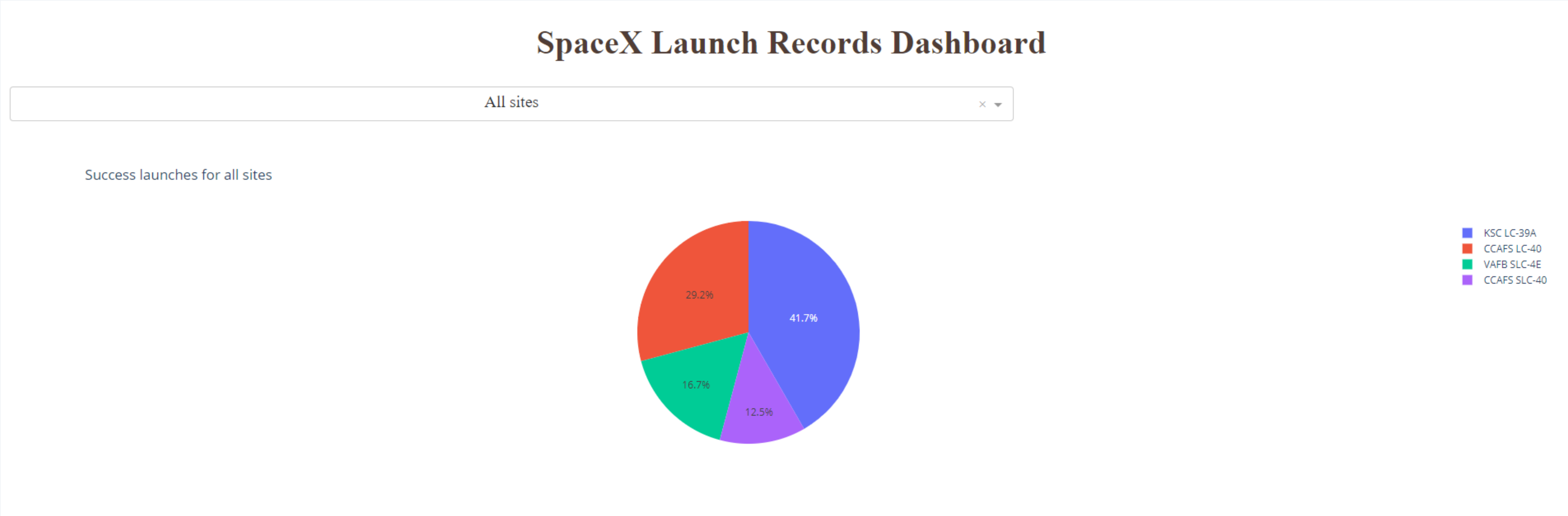
Color-labeled launch outcomes



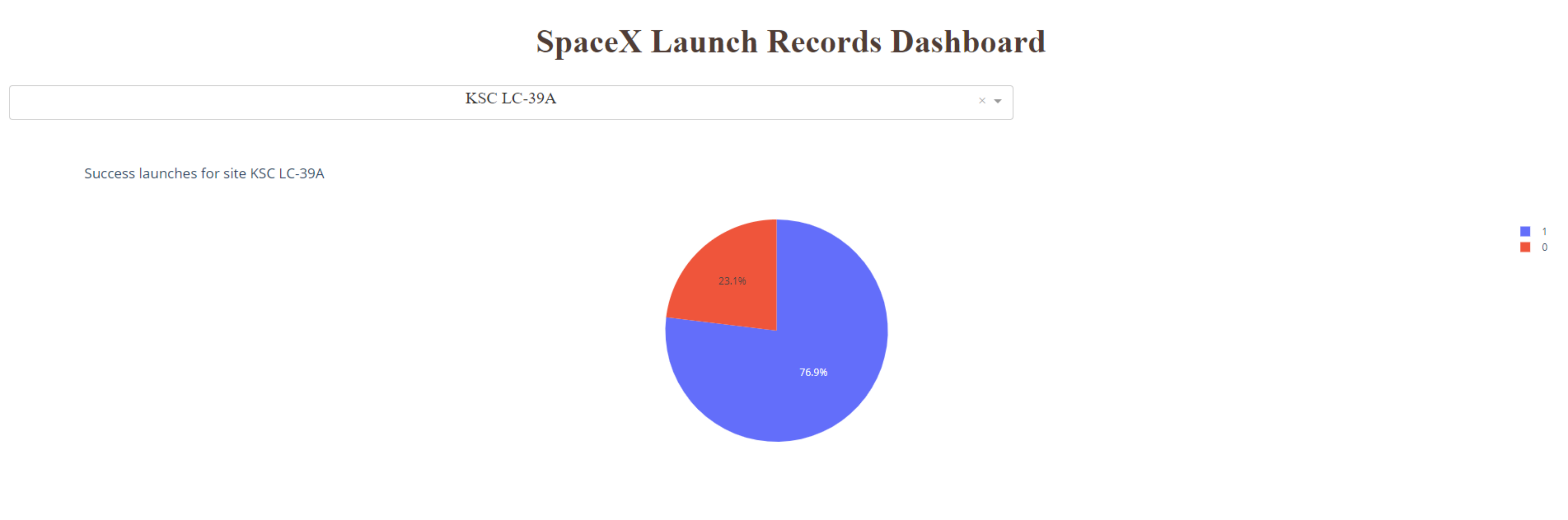
Section 5

Build a Dashboard with Plotly Dash

Success ratio launches for all sites

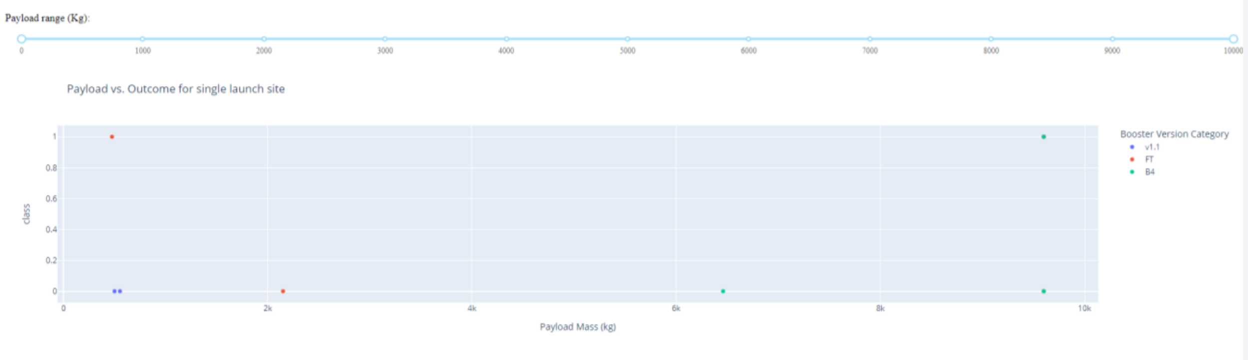


Success ratio launches for KSC LC-39A

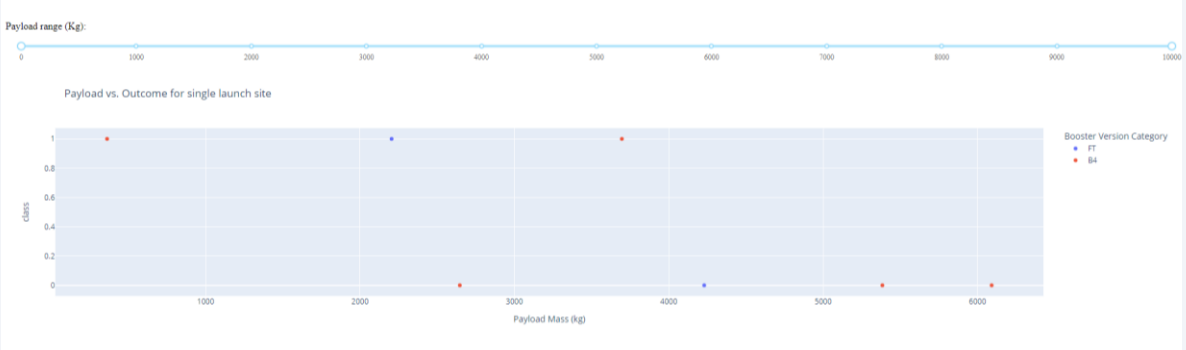


Payload vs. Launch Outcome scatter plot for all sites

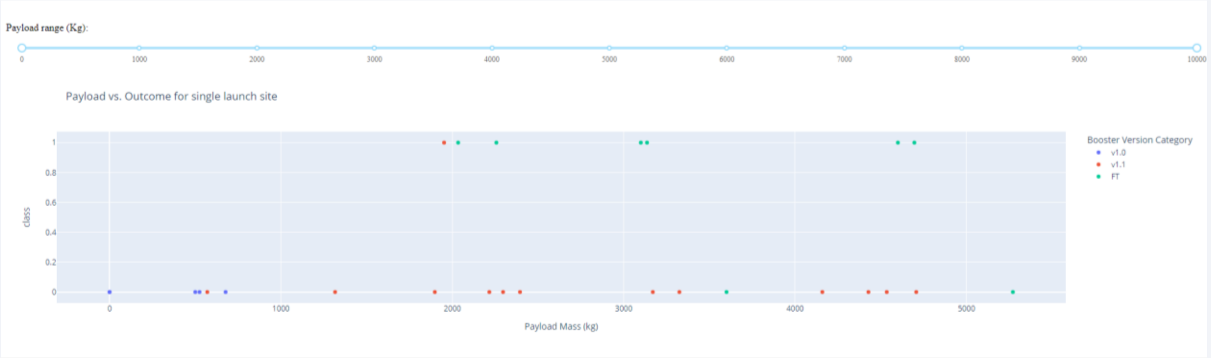
VAFB SLC-4E



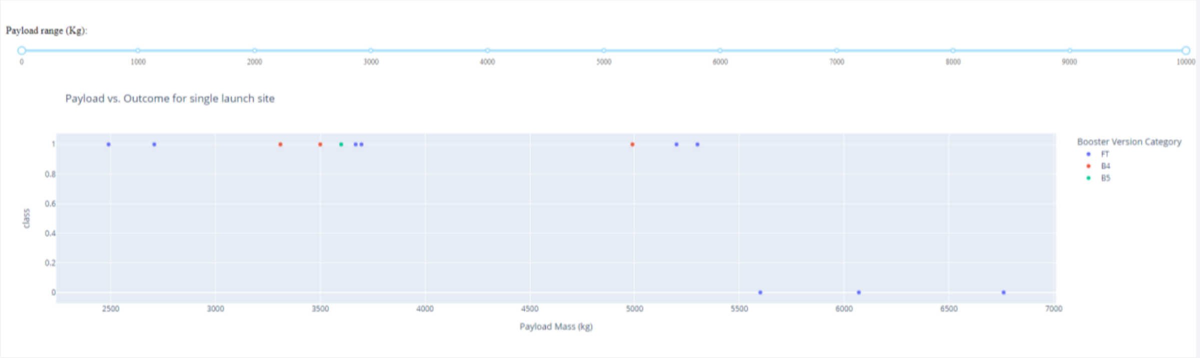
CCAFS SLC-40



CCAFS LC-40



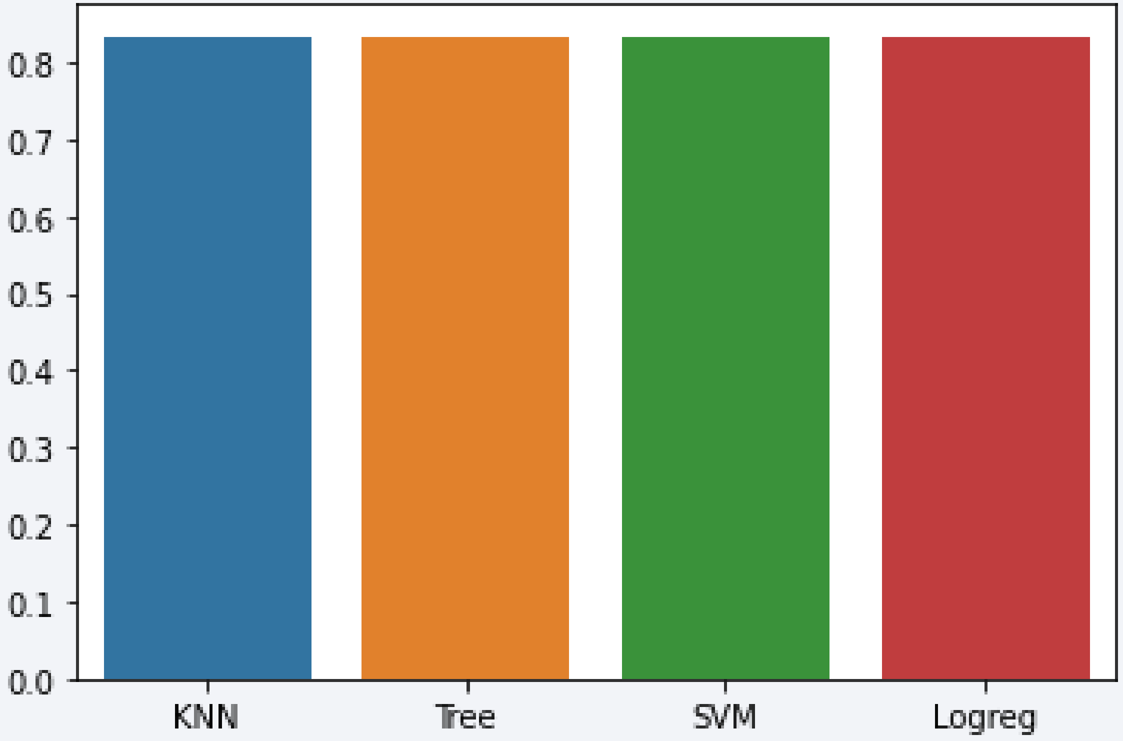
KSC LC-39A



Section 6

Predictive Analysis (Classification)

Classification Accuracy



Practically all these algorithms give the same result

Confusion Matrix

We see that decision tree can distinguish between the different classes. We see that the major problem is false positives.

Conclusions

After carrying out numerous tests with the different samples of the dataset in each of the classification methods, it can be concluded that for this case any method can be used, since any one will give the same result.

The built model can be used to predict the success rate of future launches, having an accuracy rate of 83%

Appendix

```python
# Create an app layout
app.layout = html.Div(children=[html.H1('SpaceX Launch Records Dashboard',
                                        style={'textAlign': 'center', 'color': '#503D36',
                                               'font-size': 40}),
                                # TASK 1: Add a dropdown list to enable Launch Site selection
                                # The default select value is for ALL sites
                                # dcc.Dropdown(id='site-dropdown',...)
                                dcc.Dropdown(id='site-dropdown',
                                             options=[
                                                 {'label': 'All sites', 'value': 'ALL'},
                                                 {'label': 'CCAFS LC-40', 'value': 'CCAFS LC-40'},
                                                 {'label': 'VAFB SLC-4E', 'value': 'VAFB SLC-4E'},
                                                 {'label': 'KSC LC-39A', 'value': 'KSC LC-39A'},
                                                 {'label': 'CCAFS SLC-40', 'value': 'CCAFS SLC-40'}
                                             ],
                                             value='ALL',
                                             placeholder = 'Select a Launch Site here',
                                             searchable = True,
                                             style={'width':'80%', 'padding':'3px', 'font-size':'20px', 'text-align-last':'center'}
                                             ),
```

```python
                                html.P("Payload range (Kg):"),
                                # TASK 3: Add a slider to select payload range
                                dcc.RangeSlider(id='payload-slider',
                                                min = 0,
                                                max = 10000,
                                                step = 1000,
                                                marks = {
                                                    0: '0',
                                                    1000: '1000',
                                                    2000: '2000',
                                                    3000: '3000',
                                                    4000: '4000',
                                                    5000: '5000',
                                                    6000: '6000',
                                                    7000: '7000',
                                                    8000: '8000',
                                                    9000: '9000',
                                                    10000: '10000'
                                                },
                                                value = [min_payload, max_payload]
                                ),

                                # TASK 4: Add a scatter chart to show the correlation between payload and launch success
                                html.Div(dcc.Graph(id='success-payload-scatter-chart')),
                                ])
```

```python
# TASK 2:
# Add a callback function for `site-dropdown` as input, `success-pie-chart` as output
@app.callback(  Output(component_id='success-pie-chart', component_property='figure'),
                Input(component_id='site-dropdown', component_property='value'))

# TASK 4:
# Add a callback function for `site-dropdown` and `payload-slider` as inputs, `success-payload-scatter-chart` as output
def pie_chart(site_dropdown):
    fig = ''
    if site_dropdown == 'ALL':
        title_graph = "Success launches for all sites"
        fig = px.pie(spacex_df, values='class', names='Launch Site', title = title_graph)

    else:
        title_graph = f"Success launches for site {site_dropdown}"
        filtered_DD= spacex_df[spacex_df['Launch Site'] == site_dropdown]
        filtered_LS = filtered_DD.groupby(['Launch Site','class']).size().reset_index(name='class count')
        fig = px.pie(filtered_LS, values='class count', names='class', title=title_graph)
    return fig


@app.callback(  Output(component_id='success-payload-scatter-chart', component_property='figure'),
                Input(component_id='site-dropdown', component_property='value'),
                Input(component_id="payload-slider", component_property="value"))

def payload_chart(site, payload):
    low,high = (payload[0], payload[1])
    df = spacex_df
    filtered_Spa = spacex_df[spacex_df['Payload Mass (kg)'].between(low,high)]

    if site == 'ALL':
        title_sca = 'Payload vs. Outcome for ALL'
        fig = px.scatter(filtered_Spa,x="Payload Mass (kg)", y="class", color="Booster Version Category", title='Payload vs. Outcome for All Sites')

    else:
        title_sca = ('Payload vs. Outcome for single launch site')
        filtered_sit= filtered_Spa[filtered_Spa['Launch Site'] == site]
        fig = px.scatter(filtered_sit,x="Payload Mass (kg)", y="class", color="Booster Version Category", title=title_sca)
    return fig
```

Thank you!