Applied Data Science Capstone

Space x falcon 9 landing prediction

Diana Camacho R.
28 Jul, 2022

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
  - Visualization – Charts
  - Dashboard
- Discussion
  - Findings & Implications
- Conclusion
- Appendix

EXECUTIVE SUMMARY

Space x advertises falcon 9 rocket launches with a much lower cost than other providers, the main purpose of this capstone project is to analyse the cost of the launches and predict the landings of the first stage of the falcon 9.

The data analyzed was taken from Space x and other web sources focusing on the falcon 9 launches.

Falcon 9 launches cost 62 million dollars, this estimates a cost reduction of 61.72% compared to 162 million in costs for other providers

Key Milestone → Determinate the probability of successful landings, to calculate appropriately the cost of a launch

Main points of the project

- Data collection from Space x
  - API data collection
  - Space x data collection (web scraping)
- EDA Data Wrangling
- EDA with SQL
- EDA with Data visualization
- Visual analytics and dashboards
  - Visualization maps (Folium)
  - Interactive dashboards (Ploty Dash)
- Machine learning predictions
- Results and recommendations

INTRODUCTION

Since the beginning of the space race there has been substantial technological advances in the field of space exploration, many of these advances have improved our daily lives as detailed in the "Space technology hall of fame"[1] from the Space Foundation, but all this came at a very high cost, From its creation in 1958 through 2018, the National Aeronautics and Space Administration (NASA) spent almost one trillion inflation-adjusted dollars.

A percentage of this cost has been invested in rocket launching, but in recent years private space travel companies have achieved goals before reserved only for countries; 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.

Main problem

The principal task is to determinate if the first stage of the Falcon 9 will land successfully.

Questions for analysis

- Understanding the data about the company and determinate the main factors in the launch and landing of Falcon 9
- Analyze the data through EDA, visual analytics and prediction models and determinate the best conditions to ensure a successful landing Wich will ensure cost reductions in future launches.
- Provide information that can be used if an alternate company wants to bid against space X for a rocket launch.

1. https://www.spacefoundation.org/what-we-do/space-technology-hall-of-fame/

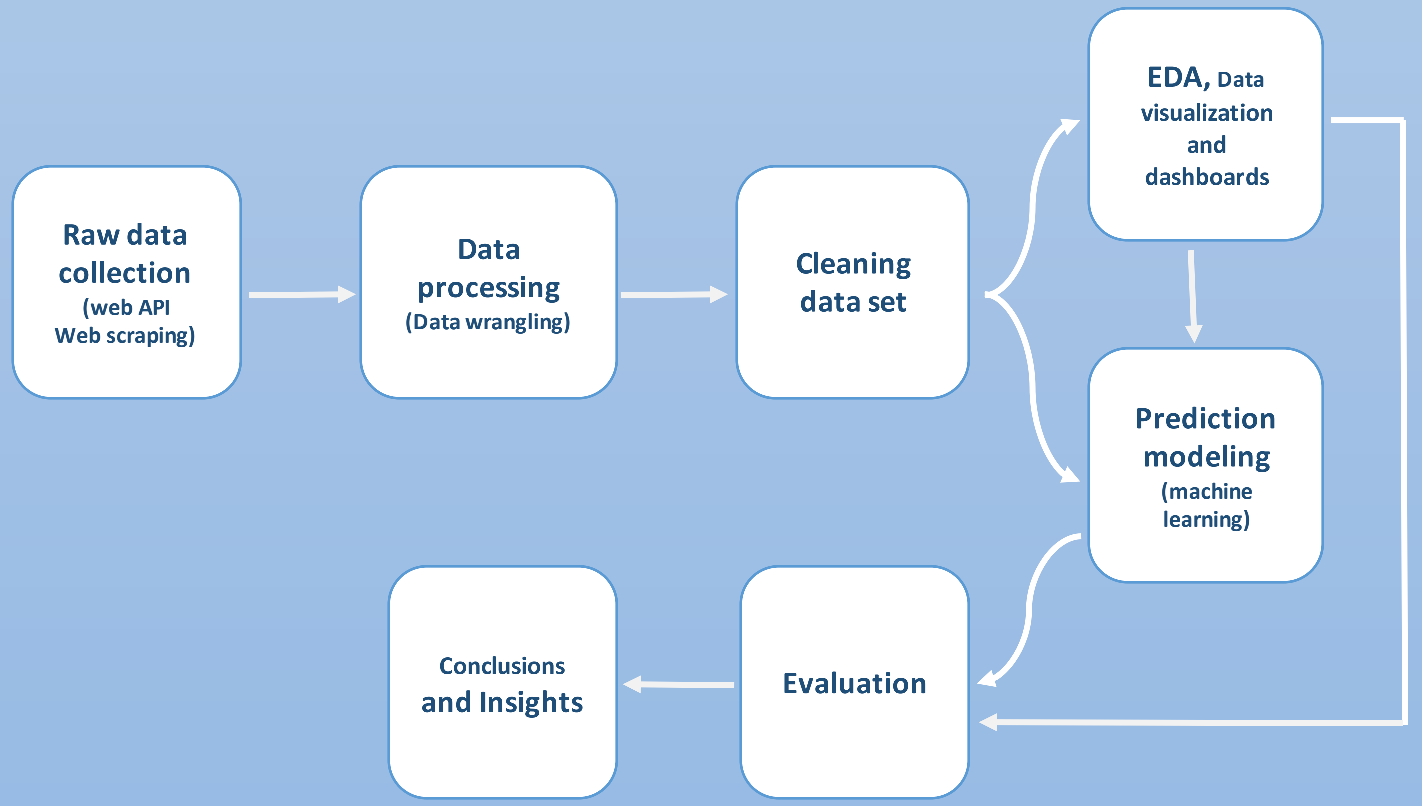
METHODOLOGY

Processing of the data set

The objective of first stages of our methodology process (data collection, web scraping and data wrangling) will help us with the understanding of the business through the understanding of the data processed.

Analysis of the data set

The exploratory data analysis and the prediction models will lead us to reach the resolution of our main problem and to draw conclusions based on this results.



DATA SOURCES

The data used in this project was gathered from different sources:

Company website

- Boosters name info: https://api.spacexdata.com/v4/rockets/
- Launch sites being used: https://api.spacexdata.com/v4/launchpads/
- Payload mass and Orbit in transit: https://api.spacexdata.com/v4/payloads/
- Landing information (outcomes, number, types, core info, hardware used or reutilized, landing pads): https://api.spacexdata.com/v4/cores/
- Rocket launches past data: https://api.spacexdata.com/v4/launches/past

Other sources (Wikipedia)

- Falcon 9 and Falcon Heavy launches: https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches

DATA COLLECTION (SPACE X API REQUESTS)

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data[data['BoosterVersion']!='Falcon 1']
```

Now that we have removed some values we should reset the FlgihtNumber column

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

```
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/pandas/core/indexing.py:1773: Se
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/use
  self._setitem_single_column(ilocs[0], value, pi)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False

```
# Calculate the mean value of PayloadMass column
payloadMass_mean = data_falcon9["PayloadMass"].mean()
print(payloadMass_mean)
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].replace(np.nan, payloadMass_mean, inplace=True)
data_falcon9.head(5)
```

```
6123.547647058824
```

For the collection of the data in the Space X API we used the requests.get() function creating a response object with the raw data, the data was normalized with a Json function and loaded into a new data frame.

Data wrangling was used to take specific columns with relevant information for the project and rows with multiple cores were removed of the data frame. We formatted the data of columns with multiple values and converted the values to readable date and time.

Helper functions were applied to the outcome and the different lists were loaded to a new data frame, finally we dropped the rows with Falcon 1 launches and replace missing values with the column mean.

To see the complete project documentation follow the link to the Jupyter Notebook on my GitHub: https://github.com/Diana-MCR/applied_DS_capstone/blob/main/week1_Data%20_collection_API_lab.ipynb

DATA COLLECTION (WIKIPEDIA WEB SCRAPING)

For the next step in the collection of the data we perform web scraping to collect Falcon 9 historical launch records from the Wikipedia page titled **List of Falcon 9 and Falcon Heavy launches.**

The data from the URL was scraped, the HTML was requested with request.get() and assigned to a response object, a BeatifulSoup object was created from the response afterwards; we search for all the tables and collected all the relevant column names then these were added to an empty dictionary.

We parsed the table and converted the results into a pandas data frame.

To see the complete project documentation follow the link to the Jupyter Notebook on my GitHub: https://github.com/Diana-MCR/applied_DS_capstone/blob/main/week1_Web_Scraping.ipynb

```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

Create a `BeautifulSoup` object from the HTML `response`

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

Print the page title to verify if the `BeautifulSoup` object was created properly

```
# Use soup.title attribute
soup.find("title")
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

```
# 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")
```

Starting from the third table is our target table contains the actual launch records.

```
# 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%;">
```

DATA WRANGLING

```
df.head(8)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
5	6	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
6	7	2014-04-18	Falcon 9	2296.000000	ISS	CCAFS SLC 40	True Ocean	1	False	False	True	NaN	1.0
7	8	2014-07-14	Falcon 9	1316.000000	LEO	CCAFS SLC 40	True Ocean	1	False	False	True	NaN	1.0

We can use the following line of code to determine the success rate:

```
df["Class"].mean()
```

```
0.6666666666666666
```

The purpose of this stage was to find patterns in the data and determinate the label for training models.

For this we use pandas to find the missing values and to identify the type of each column we also find the most relevant patterns in the data

- Calculating the number of launches in each Launch site
- Calculating the number of occurrences in each orbit
- Calculating the number and occurrence of each outcome per orbit

Using the outcome we creating a new "Class" column representing failed and successful landings. With this we were able to determinate the success rate.

To see the complete project documentation follow the link to the Jupyter Notebook on my GitHub: https://github.com/Diana-MCR/applied_DS_capstone/blob/main/Week1_spacex-Data%20wrangling.ipynb

RESULTS

EXPLORATORY DATA ANALYSIS (EDA)

The first method utilized in the exploration of the data was to load the .CSV dataset into a table in a Db2 database then execute queries to answer some questions.

- We identified the distinct launch sites.
- We found the total payload mass carried by boosters launched by NASA (CRS).
- We calculate the average payload mass carried by booster version F9 v1.1.
- We found the date of the first successful landing in ground pad.
- We list the total number of successful and failure mission outcomes.
- We List the names of the booster versions which have carried the maximum payload mass.
- We Rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017.

IBM Db2 on Cloud

Load Data | Load History | Tables | Views | Indexes | Aliases | MQTs | Sequences | Application objects

LHZ39883.SPACEXDATASET

Back

Export to CSV

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

To see the complete project documentation follow the link to the Jupyter Notebook on my GitHub: https://github.com/Diana-MCR/applied_DS_capstone/blob/main/Week2_eda-sql-coursera_sqllite.ipynb

Findings

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

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

With the SQL DISTINCT method we identified the different launch sites.

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

Landing _Outcome	COUNT
Success	20
Success (drone ship)	8
Success (ground pad)	6

With the SQL COUNT method we rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017.

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

With the SQL subqueries we listed the names of the boosters that carried the maximum payload mass

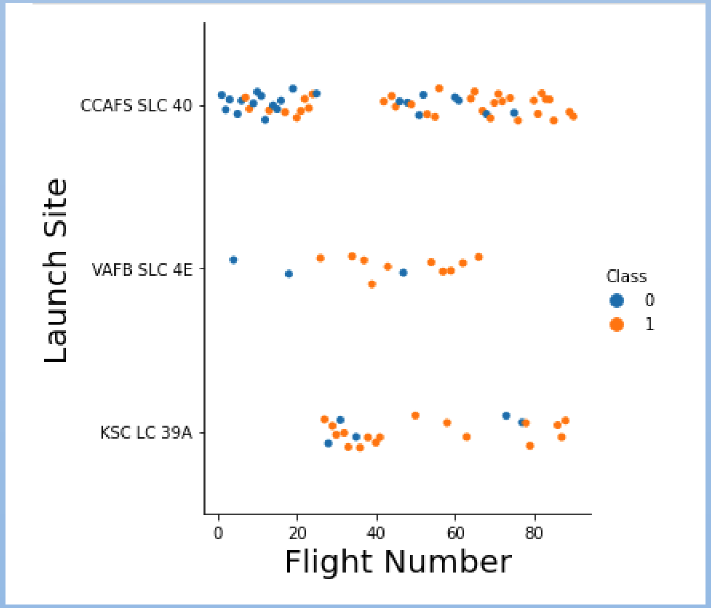
COUNT(MISSION_OUTCOME)
1
98
1
1

With the SQL COUNT and % method we calculated the successful or failed outcomes

EDA DATA VISUALIZATION

Using exploratory analysis and preparing data feature engineering with pandas and matplotlib we predicted if the Falcon 9 will land successfully and we draw insights from the different visualizations created.

Relationship between flight number and launch site

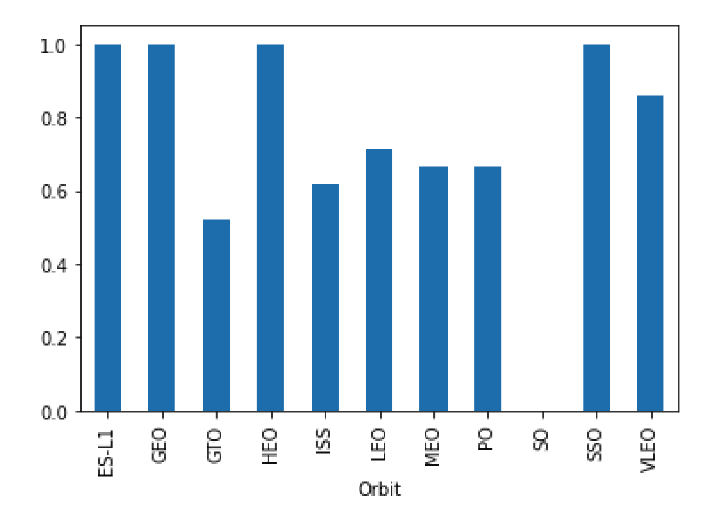


Findings

From the visualization of the relationship between the launch sites and the flight numbers we observe that more flights have been launched at CCAFS SLC 40, especially the first 20 flights, but we noticed the success rate for this launch site is lower than the others; VAFB SLC had less launches but it has the higher success rate; we can also observe than for latest flights the probability of success increased.

To see the complete project documentation follow the link to the Jupyter Notebook on my GitHub: https://github.com/Diana-MCR/applied_DS_capstone/blob/main/week2_labs-eda-dataviz.ipynb

Success rate of each orbit



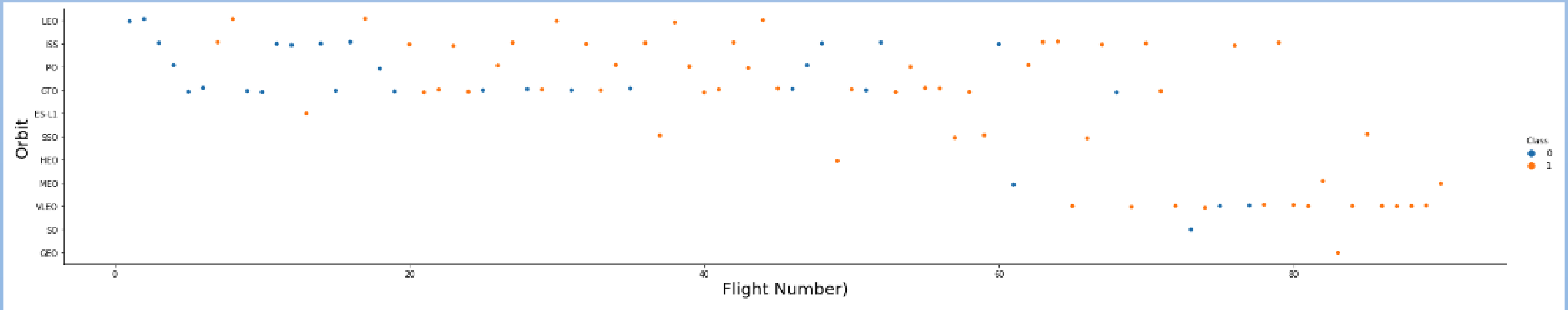
Findings

From the bar plot we can observe that the ES-L1, GEO, HEO and SSO orbits have the highest success rate and the SO orbit has the lowest rate with 0.0

From the scatter plot showing the relationship between the flight number and the orbit we can see that in the LEO orbit the success is strongly related to the numbers of flights, but we cannot observe the same positive relationship for the GTO orbit.

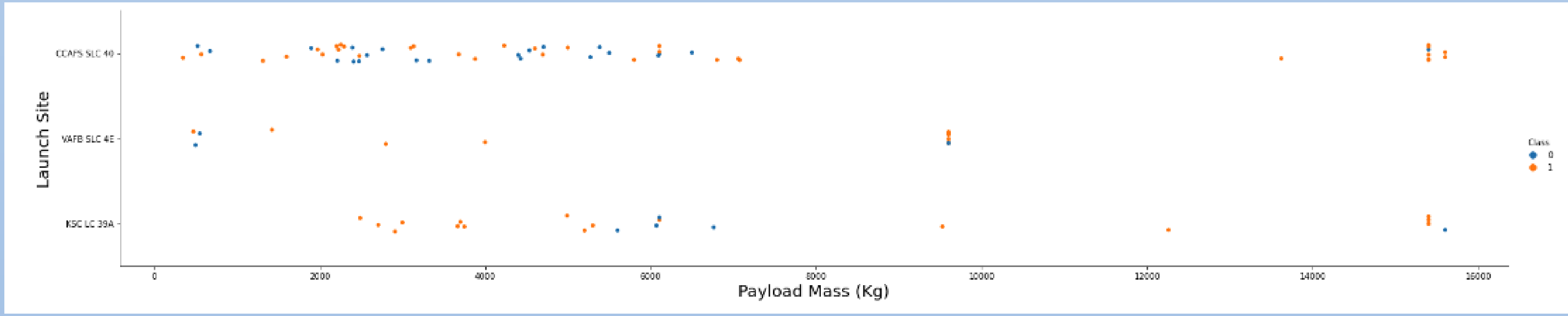
Furthermore when we analyse the two plots together we find that the orbits mentioned before with highest and lowest success rates have significatively less flights numbers (only between 1 to 5 flights max per orbit) per example the SO orbit with the only rate success at 0.0 in the scatter plot we find that this is because the only flight launched to that orbit was failed.

Relationship between flight number and Orbit type



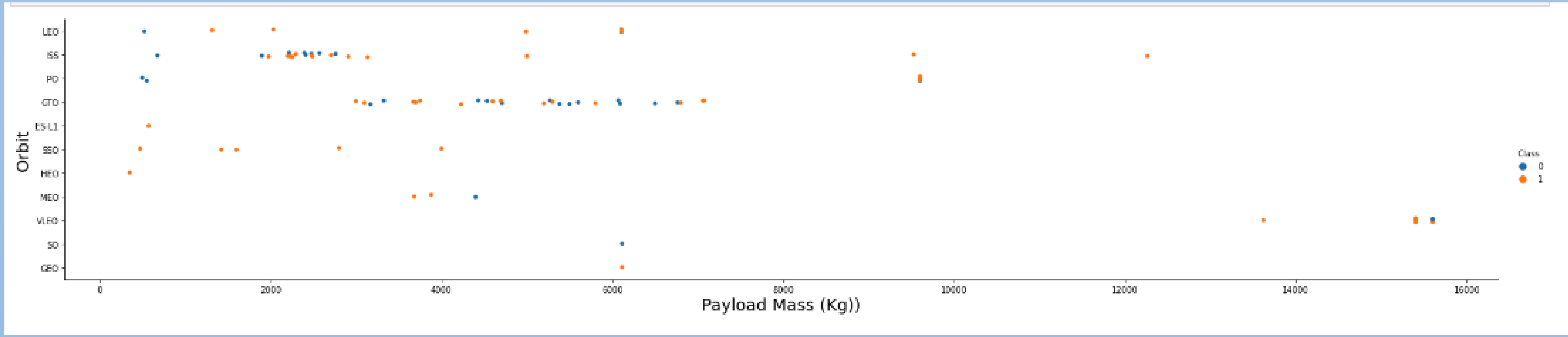
If we observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC Launch site there are no rockets launched for heavy payload mass(greater than 10000).

Relationship between Payload and Launch site



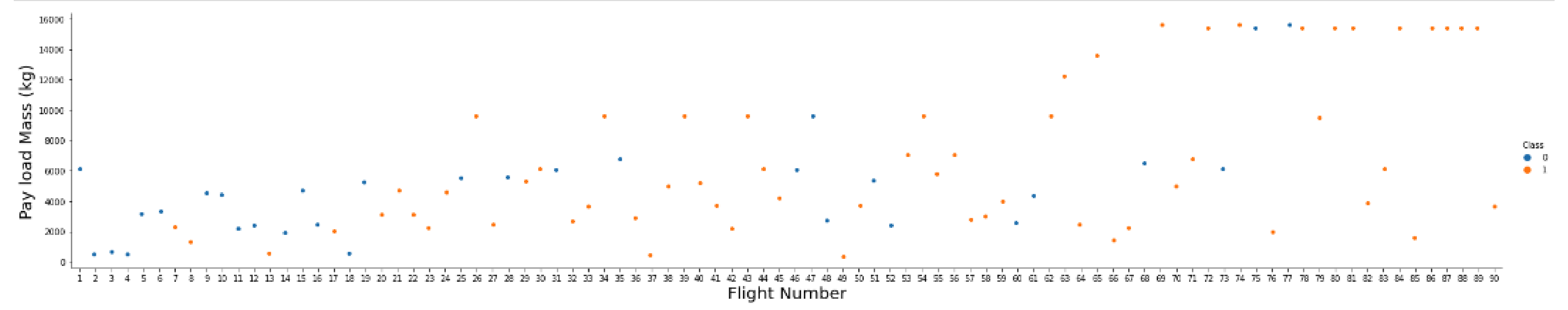
In the second graph we can see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Relationship between Payload and Orbit type



In addition to our previous observations when we compare the two plots we can clearly identify that the weight of the payload affects directly the amounts of flights, it seems the more weight of the payload the fewer flights to less Orbits can be launched but the success rate has increased.

Relationship between Payload and Flight number

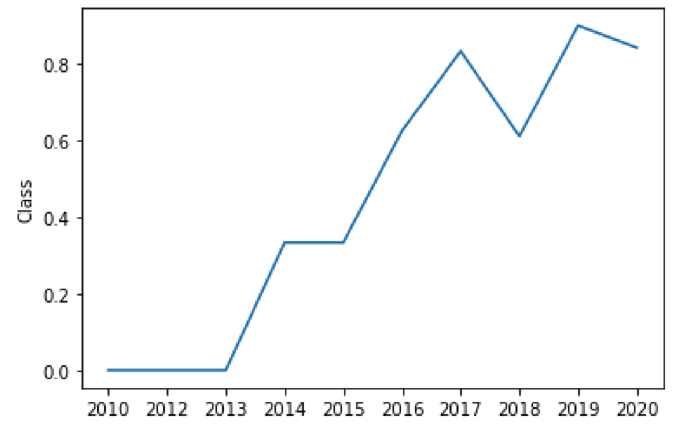


Findings

When we analyse the Flight Number vs. Payload Mass plot and overlay the outcome of the launch. We see that as the flight number increases, the first stage is more likely to land successfully.

Moreover taking into account the last Four graphs we can finally determinate that only the most recent flights can carry the more massive payloads (greater than 10,000kg), they can also transit to the most distant orbits with an increased landing success rate for all the flights.

Launch success yearly trend



VISUAL ANALYTICS (maps with folium)

The launch success rate may depend on many factors such as payload mass, orbit type, and so on. It may also depend on the location and proximities of a launch site, we utilized folium to help us visualize the Space x data set launch sites in maps and find geographical patterns.

To start the geographical visualization we create a map with folium after we add Circle and Marker objects for each launch site on the data set.

Launch sites coordinates dataset

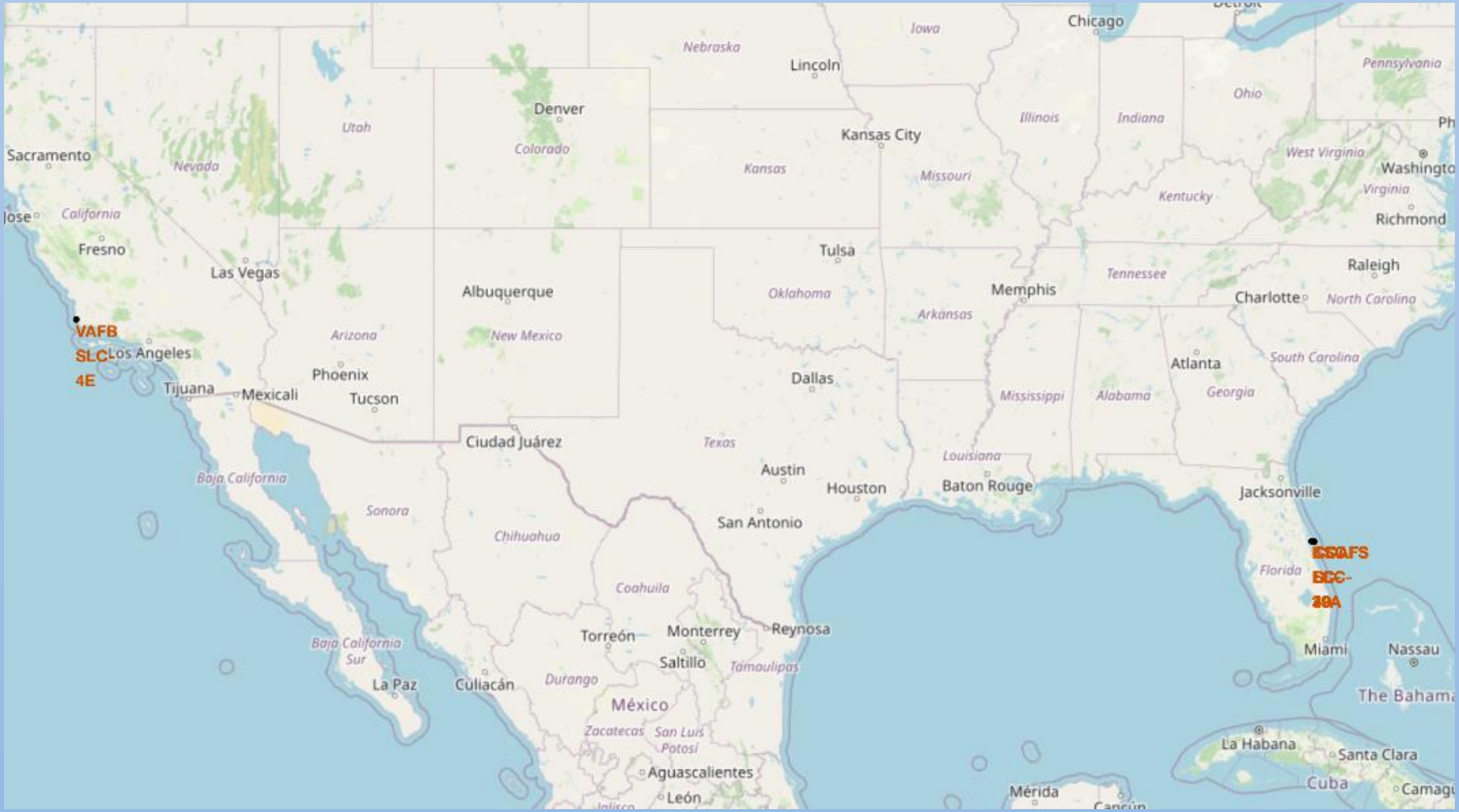
	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610745

```
# Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
# For each launch site, add a Circle object based on its coordinate (Lat, Long) values. In addition, add Launch site name as a popup labe

for index, site in launch_sites_df.iterrows():
    lat = site['Lat']
    long = site['Long']
    coordinate = [lat, long]
    circle = folium.Circle(location=coordinate, radius=5000, color='#000000', fill=True).add_child(folium.Popup(site['Launch Site']))
    marker = folium.map.Marker(coordinate, icon=DivIcon(icon_size=(20, 20), icon_anchor=(0,0), html='<div style="font-size: 12; color:#d3
    site_map.add_child(circle)
    site_map.add_child(marker)

site_map
```

Marked locations in the map



Findings

the launch sites follow certain patterns:
All the sites are located in southern regions of U.S. of America in close proximity to the coast.

To see the complete project documentation follow the link to the Jupyter Notebook on my GitHub: https://github.com/Diana-MCR/applied_DS_capstone/blob/main/week3_launch_site_location.ipynb

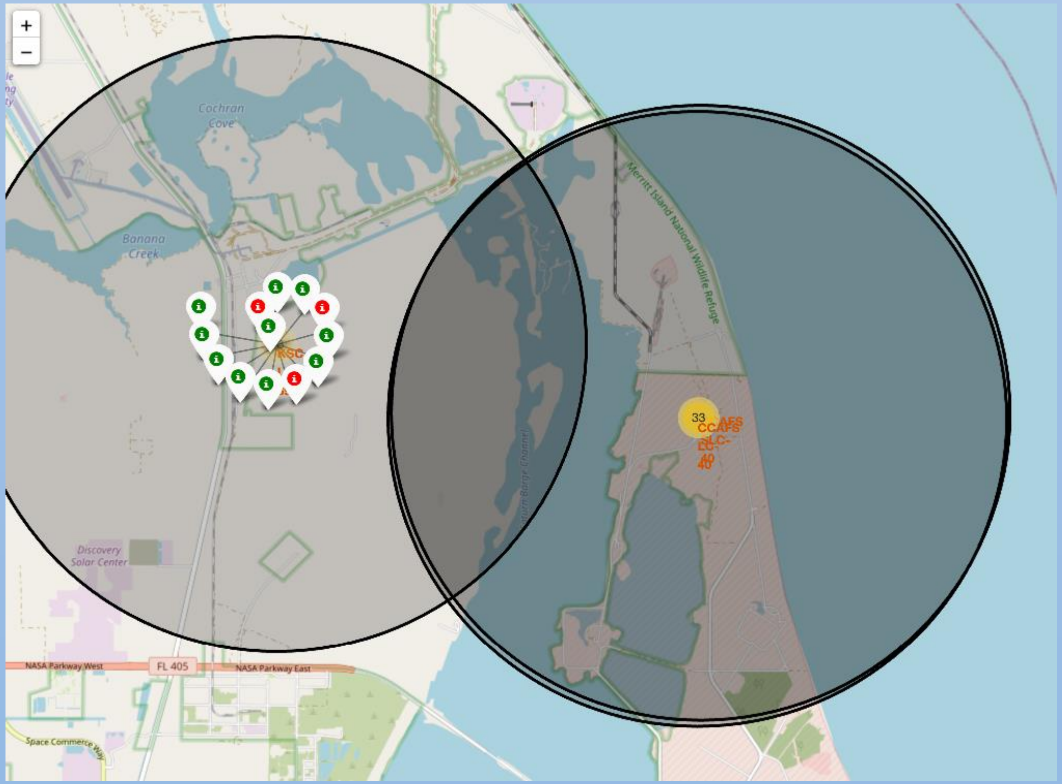
As many launches happen in the same in the same launch coordinates we added a new column in the data frame with the classification by outcome (successful landings were catalogued with green color and failed landings with red color).

Each landing record was then added to the map with the help of a marker_cluster object, thanks to this method we can easily visualize the success rate for each launch site.

Data frame with marker_color classification column

	Launch Site	Lat	Long	class	marker_color
46	KSC LC-39A	28.573255	-80.646895	1	green
47	KSC LC-39A	28.573255	-80.646895	1	green
48	KSC LC-39A	28.573255	-80.646895	1	green
49	CCAFS SLC-40	28.563197	-80.576820	1	green
50	CCAFS SLC-40	28.563197	-80.576820	1	green
51	CCAFS SLC-40	28.563197	-80.576820	0	red
52	CCAFS SLC-40	28.563197	-80.576820	0	red
53	CCAFS SLC-40	28.563197	-80.576820	0	red
54	CCAFS SLC-40	28.563197	-80.576820	1	green
55	CCAFS SLC-40	28.563197	-80.576820	0	red

Marked launches in the map (classification by marker_color)

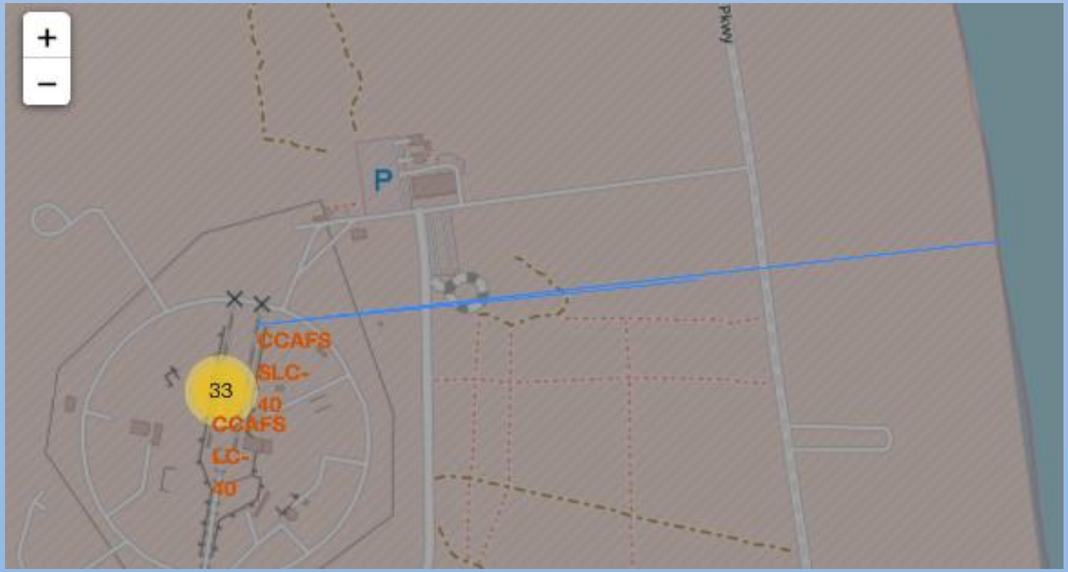


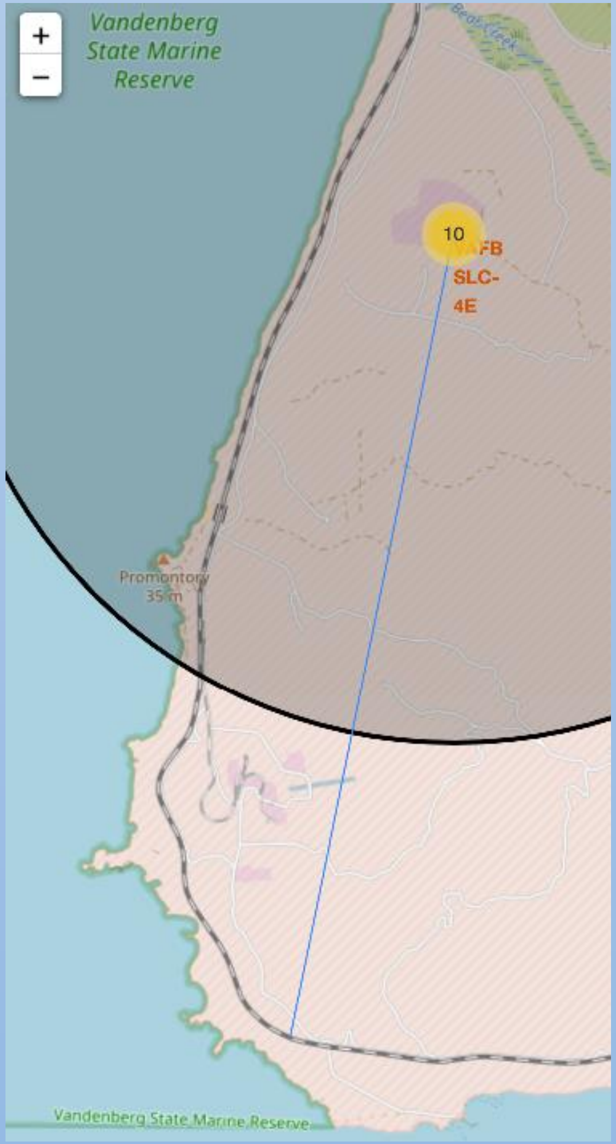
Visualization of distances between launch sites and proximities

By adding a mouse position on the map we were able to explore the map and find the proximity of the launch sites to some points of interest.

For this we help ourselves with defined functions for calculating the distances between coordinates and a Polyline object to mark the distance in the map.

Distance from launch site to the coastline





Distance from launch site to railway

Findings

Thanks to this we can successfully determine that launch sites meet certain conditions:

- They are close to coastlines.
- Far from cities and highways.
- There are railroads close to launch sites.

The recognition of the conditions required for the the locations of the launch sites help us to identify in a map what other coordinates will be ideal to establish other possible future launch sites.

DASHBOARD (INTERACTIVE APP WITH PLOTLY DASH)

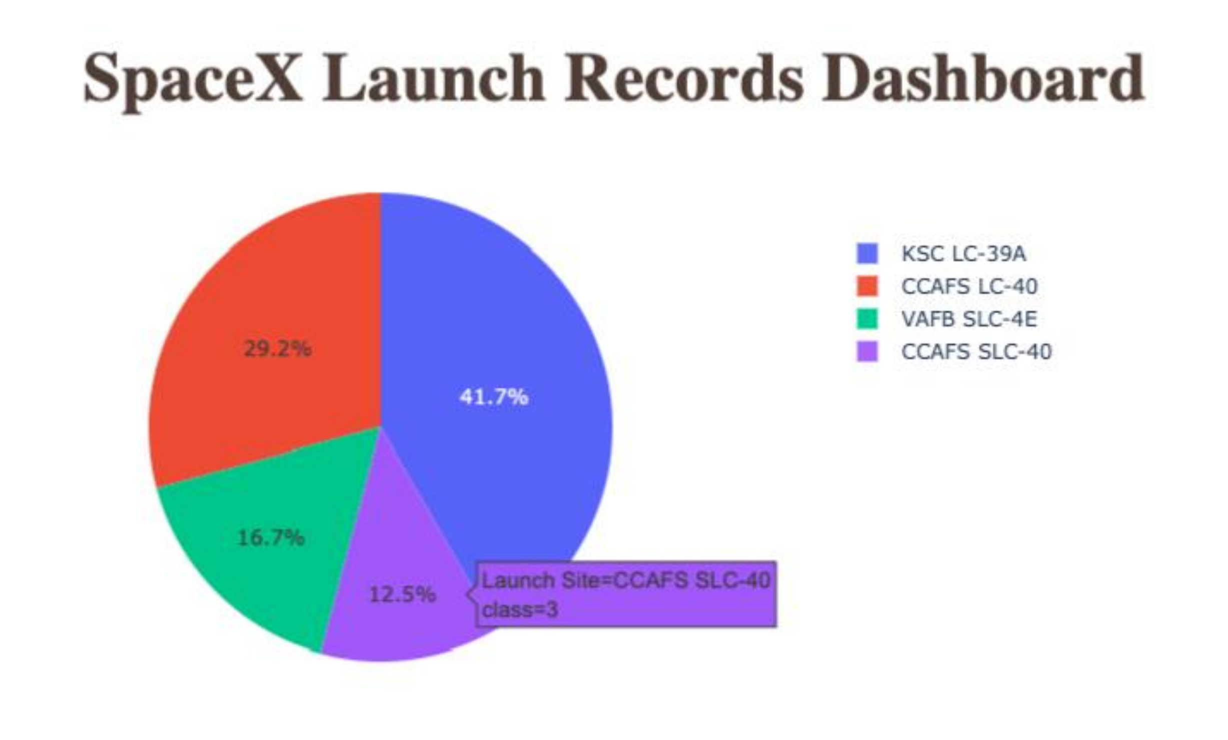
The dashboard allow us to visualize the data in a different way by interacting with all the components.

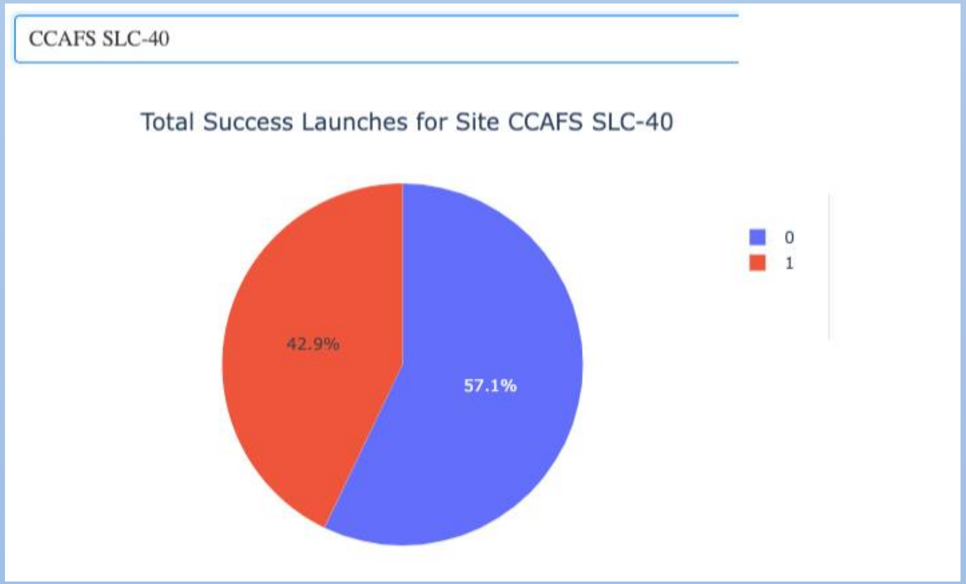
Findings

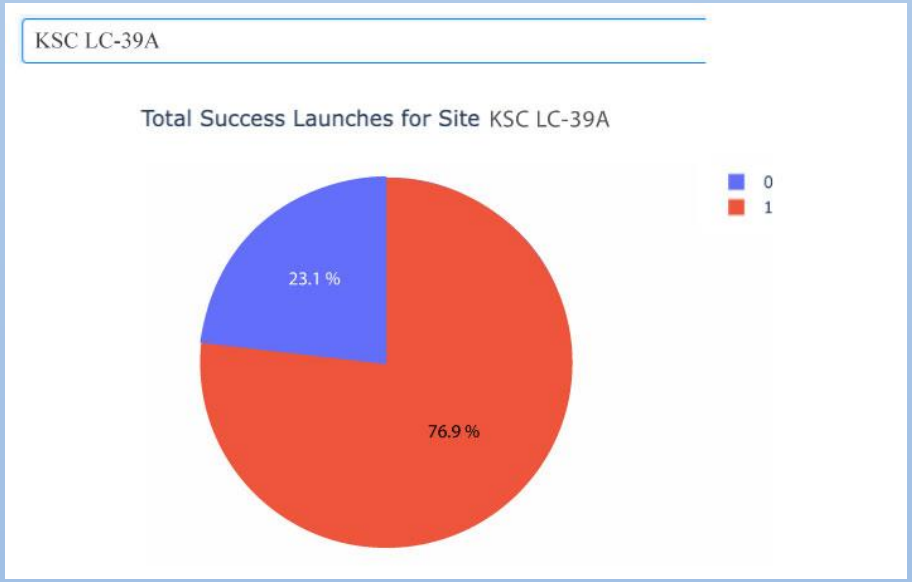
In this pie chart we can observe that KSC LC-39A is the launch site with the most successful rate with 41.7% ; in comparison CCAFS SLC-40 is the launch site with the lower successful rate with 12.5%.

To see the complete project documentation follow the link to the Jupyter Notebook on my GitHub: https://github.com/Diana-MCR/applied_DS_capstone/blob/main/week_3_Dashboard_application_Plotly.py

Total successful launches by all sites



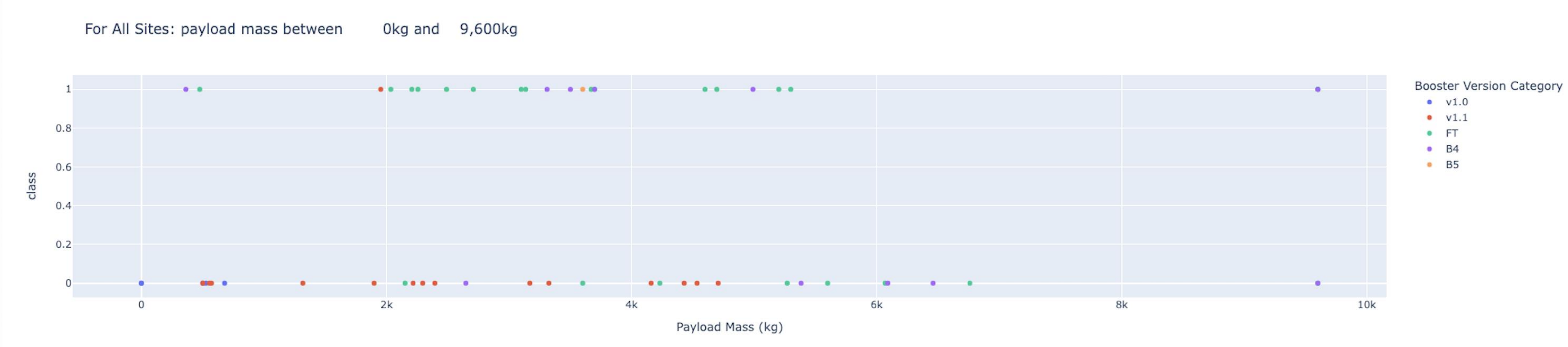




Findings

In this two pie charts we can observe with more detail the launch sites with higher and lower success rates, the successful launches for CCAFS SLC-40 are only 42.9% of all the launches, in comparison the successful launches for KSC LC-39A ascend to 76.9%

Relationship between Payload and Launch Outcome for all Booster versions



Findings

The scatter plot show us that for the booster version FT the success rate is higher with payloads between 0 and 6000kg while for the booster version V1.1 the success rate is significatively low with payloads between 0 to 6000kg.

In addition only the booster version B4 had launches with payloads higher than 9000kg and shows both successful and failed launches.

We can finally conclude that the success rate for low weighted Payloads is higher than for heavy weighted payloads.

IBM Developer SKILLS NETWORK

MACHINE LEARNING PREDICTION

With the machine learning prediction tools we created a machine learning pipeline to predict if the first stage will land given the data from the preceding labs.

The object was to perform exploratory Data Analysis and determine Training Labels so we could:

- create a column for the class
- Standardize the data
- Split into training data and test data
- Find best Hyperparameter for SVM, Classification Trees and Logistic Regression
- Find the method performs best using test data.

```
# students get this
transform = preprocessing.StandardScaler()
```

```
X = transform.fit_transform(X)
X
```

```
array([[-1.71291154e+00, -5.29526321e-17, -6.53912840e-01, ...,
        -8.35531692e-01,  1.93309133e+00, -1.93309133e+00],
       [-1.67441914e+00, -1.19523159e+00, -6.53912840e-01, ...,
        -8.35531692e-01,  1.93309133e+00, -1.93309133e+00],
       [-1.63592675e+00, -1.16267307e+00, -6.53912840e-01, ...,
        -8.35531692e-01,  1.93309133e+00, -1.93309133e+00],
       ...,
       [ 1.63592675e+00,  1.99100483e+00,  3.49060516e+00, ...,
         1.19684269e+00, -5.17306132e-01,  5.17306132e-01],
       [ 1.67441914e+00,  1.99100483e+00,  1.00389436e+00, ...,
         1.19684269e+00, -5.17306132e-01,  5.17306132e-01],
       [ 1.71291154e+00, -5.19213966e-01, -6.53912840e-01, ...,
        -8.35531692e-01, -5.17306132e-01,  5.17306132e-01]])
```

```
X_train, X_test, Y_train, Y_test
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
print("Train set: ", X_train.shape, Y_train.shape)
print("Test set: ", X_test.shape, Y_test.shape)
```

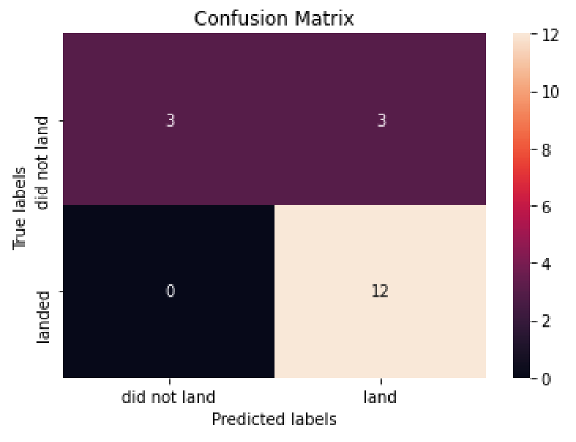
```
Train set:  (72, 83) (72,)
Test set:  (18, 83) (18,)
```

we can see we only have 18 test samples.

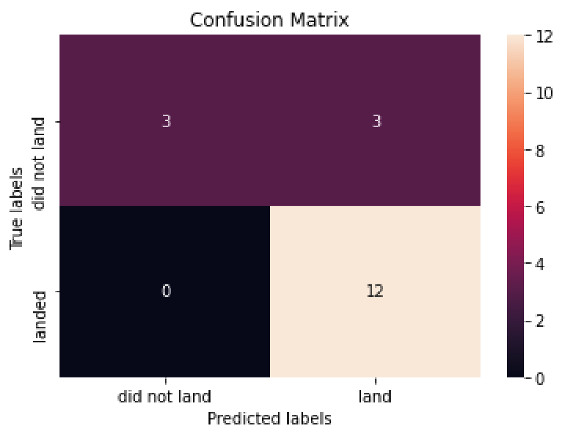
```
Y_test.shape
```

```
(18,)
```

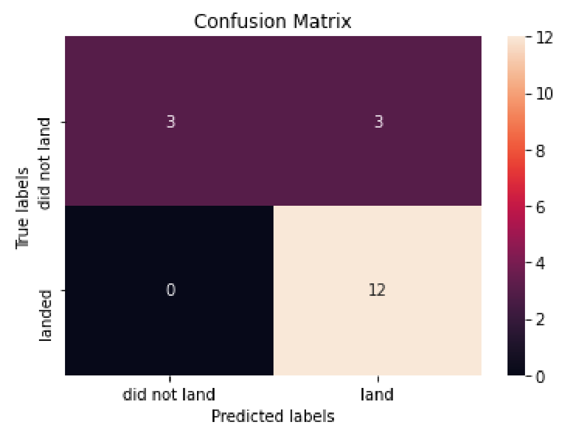
Standarization of the data and creation of Train test split set



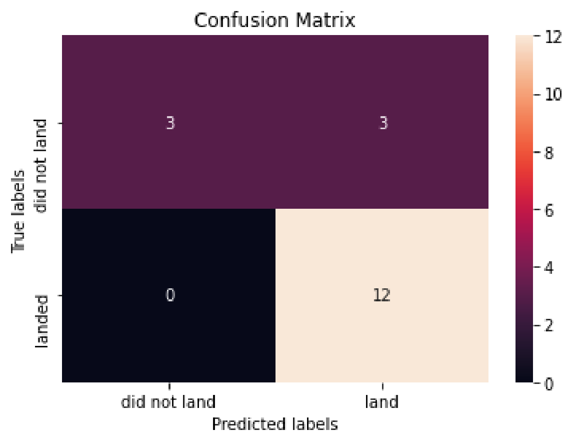
Logistic regression confusion matrix



Support vector machine confusion matrix



Decision tree confusion matrix



K nearest neighbors confusion matrix

Findings

By analysing the confusion matrices for all the models by can observe that all the models present the same results.

The classifiers can distinguish the different classes but all of them still present some false positives.

To see the complete project documentation follow the link to the Jupyter Notebook on my GitHub: https://github.com/Diana-MCR/applied_DS_capstone/blob/main/Week4_Machine%20Learning%20Prediction_Part_5.ipynb

```
knn_cv.score(X_test, Y_test)
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/sit
s are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseV
/home/jupyterlab/conda/envs/python/lib/python3.7/sit
s are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseV
0.8333333333333334
```

```
svm_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```

```
tree_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```

Findings

When we calculated the accuracy scores for the models we find out that all the methods utilized give practically the same result.

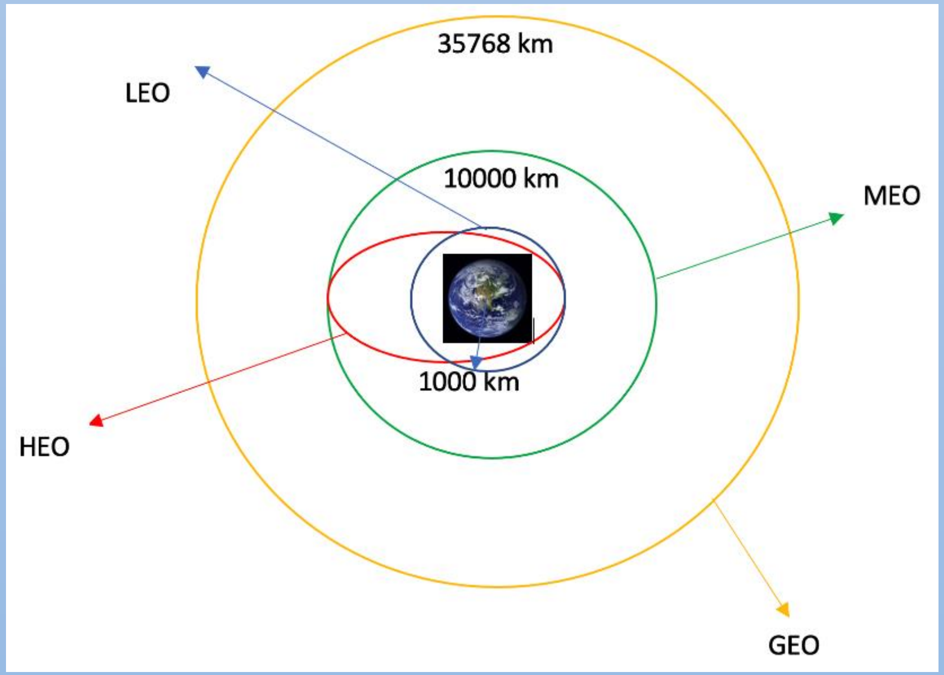
From this we can deduct either that there's not best method for the prediction or any of the methods is appropriate for the prediction of successful launches

CONCLUSION

OVERALL FINDINGS AND IMPLICATIONS

- The orbits ES-L1, GEO, HEO and SSO have the highest success rates.
- Success rates of Space X launches have been increasing considerably from in a ten year period, by 2020 the success rate reached to over 90%, giving a positive perspective for future launches.
- KSC LC 39-A is the launch site with most successful launches.
- The increment of the payload mass has overall negative impact in the success of the launches.
- The accuracy score of the prediction models is 83.3% which implies that the election of an specific prediction model over other won't affect the results.

APENDIX



GRAPHIC REPRESENTATION OF SOME OF THE ORBITS

```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',...)
                                html.Br(),
                                dcc.Dropdown(
                                    id='site-dropdown',
                                    options=[
                                        {'label': 'All Sites', 'value': 'ALL'},
                                        {'label': 'CCAFS LC-40', 'value': 'CCAFS LC-40'},
                                        {'label': 'CCAFS SLC-40', 'value': 'CCAFS SLC-40'},
                                        {'label': 'KSC LC-39A', 'value': 'KSC LC-39A'},
                                        {'label': 'VAFB SLC-4E', 'value': 'VAFB SLC-4E'},
                                    ],
                                    value='ALL',
                                    placeholder='Select a launch site here',
                                    searchable=True
                                ),

                                # TASK 2: Add a pie chart to show the total successful launches count for all sites
                                # If a specific launch site was selected, show the Success vs. Failed counts for the site
                                html.Div(dcc.Graph(id='success-pie-chart')),
                                html.Br(),

                                html.P("Payload range (Kg):"),
                                # TASK 3: Add a slider to select payload range
                                #dcc.RangeSlider(id='payload-slider',...)
                                dcc.RangeSlider(
                                    id='payload-slider',
                                    min=0, max=1000, step=1000,
                                    marks={
                                        0: '0',
                                        100: '100'},
                                    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')),
                                ])
```

DASHBOARD APPLICATION (CREATION WITH PLOTLY)


THANK YOU.
SPACEX