IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Carlos Gaitán Gil
21/12/2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection, using an API and with Web Scraping.
  - Data wrangling or tranformation.
  - Exploratory Data Analysis with Visualizations and SQL.
  - Interactive Visual Analytics with Map and a Dashboard.
  - Predictive Analysis using different classification models.
- Summary of all results
- Conclussions

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

We will try to predict if the Falcon 9 first stage will land successfully.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

We collected data that includes information about the rocket used, payload delivered, launch and landing specifications and landing outcomes both through SpaceX REST API and Web Scraping.

- Perform data wrangling

We transformed our collected data so the column with the landing outcomes, which initially included 8 types of results, now tells us whether the case was successful (class 1) or not (class 0).

- Perform exploratory data analysis (EDA) using visualization and SQL

- Perform interactive visual analytics using Folium and Plotly Dash

- Perform predictive analysis using classification models

We built, tuned and tested different classification models in order to discover the most accurate algorithm.

Data Collection – SpaceX API

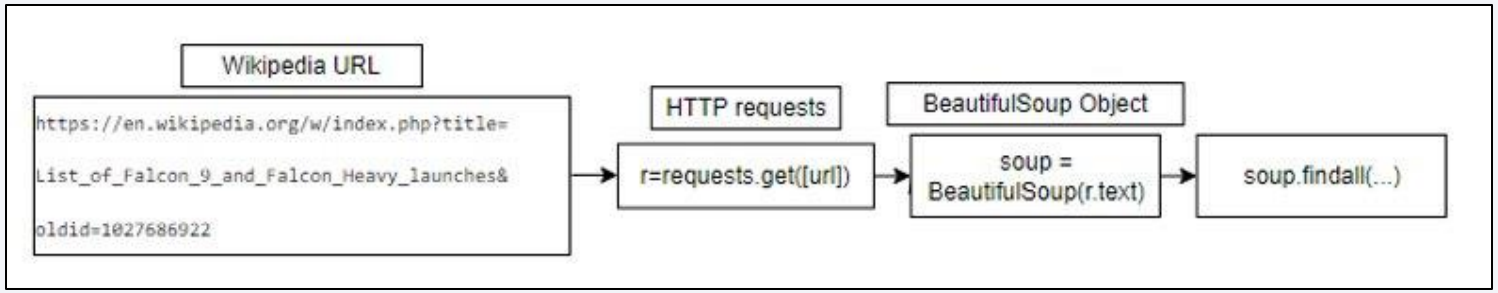
- Data was collected using a specific endpoint in SpaceX REST API, requesting the data from the url of this point, decoding the response as json format and, ultimately, turning it to into a pandas DataFrame.

- GitHub URL:

https://github.com/cgaitangil/testrepo/blob/main/W1.1_DataCollection_API_json.ipynb

Data Collection - Scraping

- Data was collected using a technique called Web Scraping, getting the data from the desired url, Wikipedia in this case, requesting the HTML page with an HTTP GET method and creating a BeatifulSoup object from the response, which allows us to extract information within the page using HTML tags.



- GitHub URL:

https://github.com/cgaitangil/testrepo/blob/main/W1.1_DataCollection_WebScraping.ipynb

Data Wrangling

- During this data transformation process, we selected the column 'Outcome' from our collected data, a categorical variable whose values were True Ocean (the mission outcome was successfully landed to a specific region of the ocean), False Ocean (outcome was unsuccessfully landed to a specific region of the ocean), True RTLS (outcome was successfully landed to a ground pad), False RTLS (outcome was unsuccessfully landed to a ground pad), True ASDS (outcome was successfully landed to a drone ship), False ASDS (outcome was unsuccessfully landed to a drone ship), and None ASDS and None None (both represent a failure to land).
- We mainly converted those outcomes into Training Labels with 1 meaning the booster successfully landed, and 0 meaning it was unsuccessful. We saved the transformed data into a new column called 'Class', which was our target or dependent variable.

- GitHub URL:

https://github.com/cgaitangil/testrepo/blob/main/W1.2_DataWrangling.ipynb

EDA with Data Visualization

- Summarization of charts plotted and reasons/goals.
  - A scatter plot to visualize the relationship between Flight Number and Launch Site.
  - A scatter plot to visualize the relationship between Payload and Launch Site.
  - A bar chart to visualize the relationship between success rate of each orbit type.
  - A scatter plot to visualize the relationship between FlightNumber and Orbit type.
  - A scatter plot to visualize the relationship between Payload and Orbit type.
  - A line plot to visualize the launch success yearly trend.
- GitHub URL:

https://github.com/cgaitangil/testrepo/blob/main/W2.2_EDA_Visualization.ipynb

EDA with SQL

- Unique launch sites in the space mission
  - **SELECT DISTINCT**(Launch_Site) **FROM** SPACEXTABLE
- Display 5 records where launch sites begin with the string 'CCA'
  - **SELECT** * **FROM** SPACEXTABLE **WHERE** Launch_Site **LIKE** 'CCA%' **LIMIT** 5
- Display the total payload mass carried by boosters launched by NASA (CRS)
  - **SELECT SUM**(PAYLOAD_MASS__KG_) **AS** TOTAL **FROM** SPACEXTABLE **WHERE** Customer = 'NASA (CRS)'
- Display average payload mass carried by booster version F9 v1.1
  - **SELECT AVG**(PAYLOAD_MASS__KG_) **AS** 'F9 v1.1 AVG. Payload' **FROM** SPACEXTABLE **WHERE** Booster_Version = 'F9 v1.1'
- List the date when the first succesful landing outcome in ground pad was acheived.
  - **SELECT MIN**(Date) **FROM** SPACEXTABLE **WHERE** Mission_Outcome = 'Success'
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - **SELECT** Booster_Version **FROM** SPACEXTABLE **WHERE** Landing_Outcome = 'Success (drone ship)' **AND** PAYLOAD_MASS__KG_ **BETWEEN** 4000 **AND** 6000

EDA with SQL

- List the total number of successful and failure mission outcomes
  - **SELECT** Mission_Outcome, **COUNT**(*) **FROM** SPACEXTABLE **GROUP BY** Mission_Outcome
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
  - **SELECT** * **FROM** (**SELECT** Booster_Version, PAYLOAD_MASS__KG_ **FROM** SPACEXTABLE **ORDER BY** PAYLOAD_MASS__KG_ **DESC**) **LIMIT** 5
- List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
  - **SELECT** substr(Date, 6, 2), Landing_Outcome, Booster_Version, Launch_Site **FROM** SPACEXTABLE
  - **WHERE** Landing_Outcome = 'Failure (drone ship)' **AND** substr(Date, 0, 5) = '2015'
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.
  - **SELECT** Landing_Outcome, **COUNT**(*) **AS** Total **FROM** SPACEXTABLE **GROUP BY** Landing_Outcome HAVING Date **BETWEEN** '2010-06-04' **AND** '2017-03-20' **ORDER BY** Total **DESC**

- GitHub URL:

https://github.com/cgaitangil/testrepo/blob/main/W2.1_ExploratoryDataAnalysisW_SQL_sqllite.ipynb

Build an Interactive Map with Folium

- Here we created a map using folium Python library. Besides, we put a marker on each of the launch sites (unique/distinct values in the respective column), as well as a 1km circle around those points, using the columns 'Lat' and 'Long' for each location.
- Then we updated our map and, for each land result, we added a marker (green or red, depending on the landing result, columns 'Class') to our Marker Cluster object, previously created and added to our map.
- Finally, for a specific result or sample, we draw a line to the nearest coast and to the city of Los Angeles, in order to discover if the landing sites are far enough from cities and close enough to coasts.

- GitHub URL:

  https://github.com/cgaitangil/testrepo/blob/main/W3.1_IVA_folium_LocationMarksAndDistances.ipynb

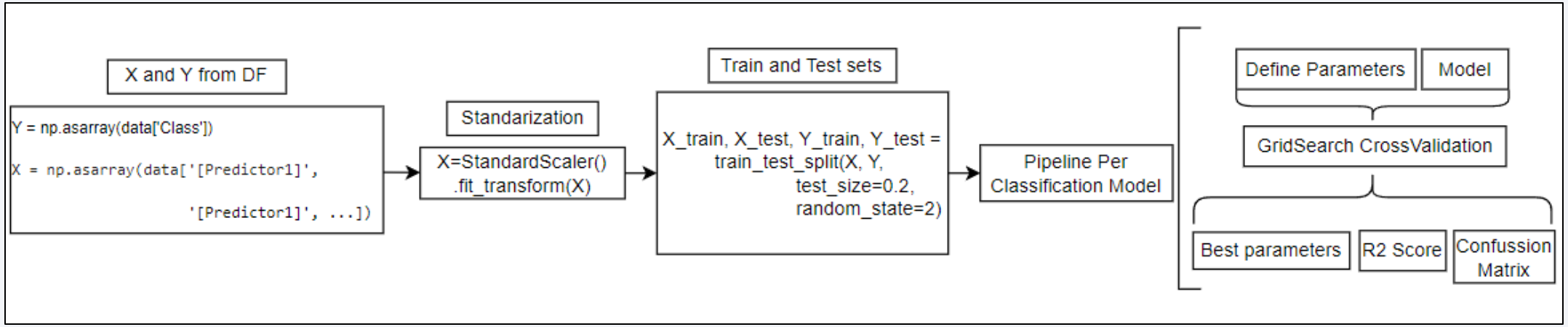
  (I can't see the created maps in GitHub, I don't know why. Please see them in section 3.)

Build a Dashboard with Plotly Dash

- In this section we created:
  - A Dropdown element to select a Launch Site or leave it as ALL, in which case the output will be different.
  - A zone to render (using a callback function) a pie chart visualizing launch success counts in the selected launch site or overall success rate, including all sites, if none was selected in dropdown.
  - A Range Slider to select Payload, which allows us to define a payload range that will affect to the rendered chart.
  - Another zone to render (using a different callback function) a scatter plot visualizing the relationship between payload (x axis) and launch outcome (y axis), considering the Booster Version as the color of those represented points, again conseidering if any launch site was selected or we leave the dropdown value as ALL.
- GitHub URL:

https://github.com/cgaitangil/testrepo/blob/main/W3.2_InteractiveVisualAnalyticsW_dash.py

Predictive Analysis (Classification)



- First, we selected our predictors or independent variables and our target or dependent variable (X and Y) from the initial dataframe.
- Then, we normalize the array or matrix with the predictors, X, to decrease the influence of the variables with high values, setting the mean to 0 and variance from minus one to one.
- After that, we divide X and Y in order to select our train and test data, needed for both in-sample and out-of-sample accuracies.

Predictive Analysis (Classification)

- Now that we have out train and test sets, we apply a pipeline or algorithm to different classifications models, which are Logistic Regression, Support Vector Machine, Decision Tree and K-Nearest Neighbors.
- In this pipeline we use a GridSearch model with cross validation, which allows us to find the best values for different parameters, in order to run each model again with the best parameters for them and compare the results (R2 Score out-of-sample) to select, again and finally, the best classification model.

- GitHub URL:

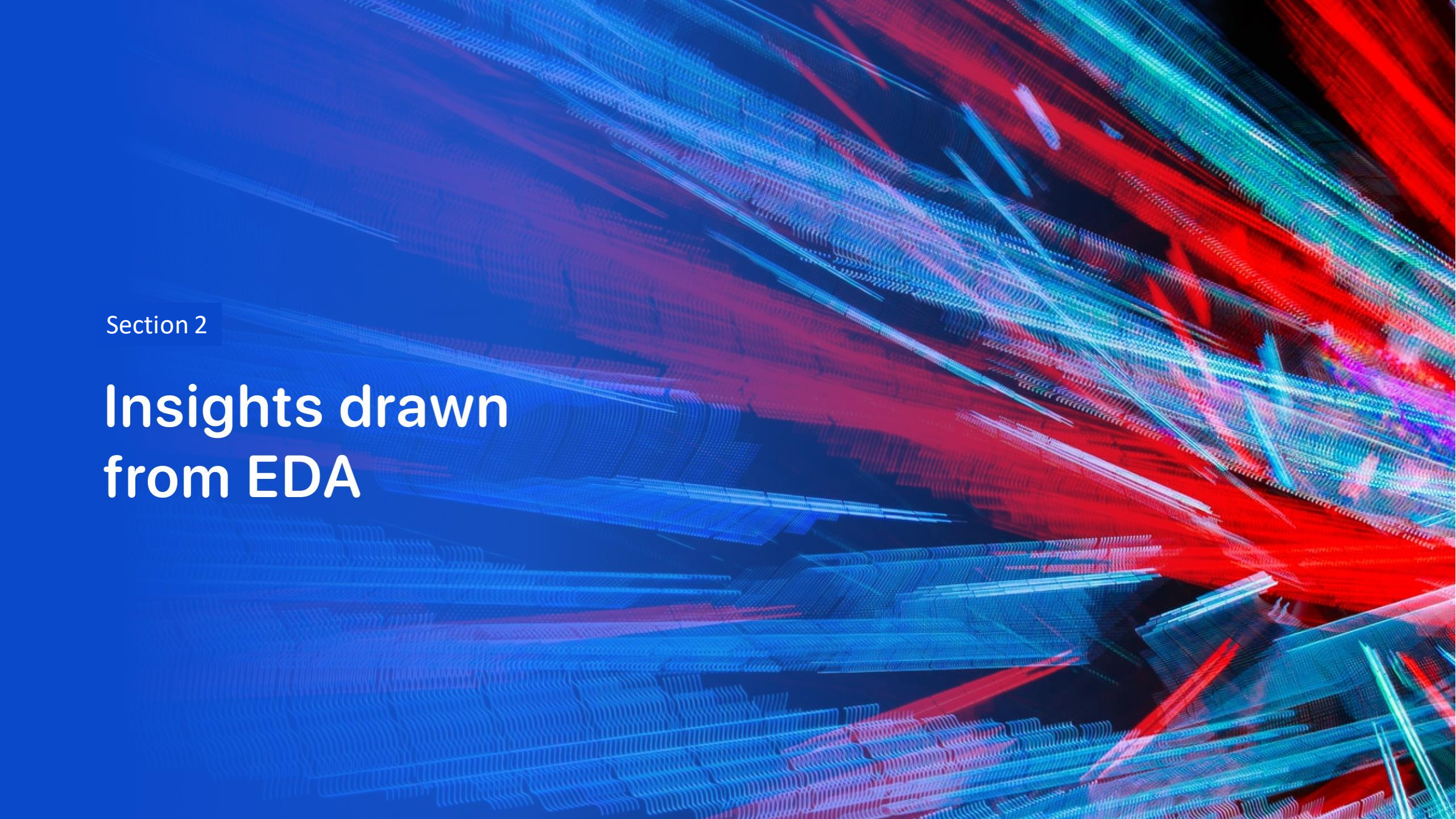
https://github.com/cgaitangil/testrepo/blob/main/W4_PredictiveAnalysis_Classification.ipynb

Results

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

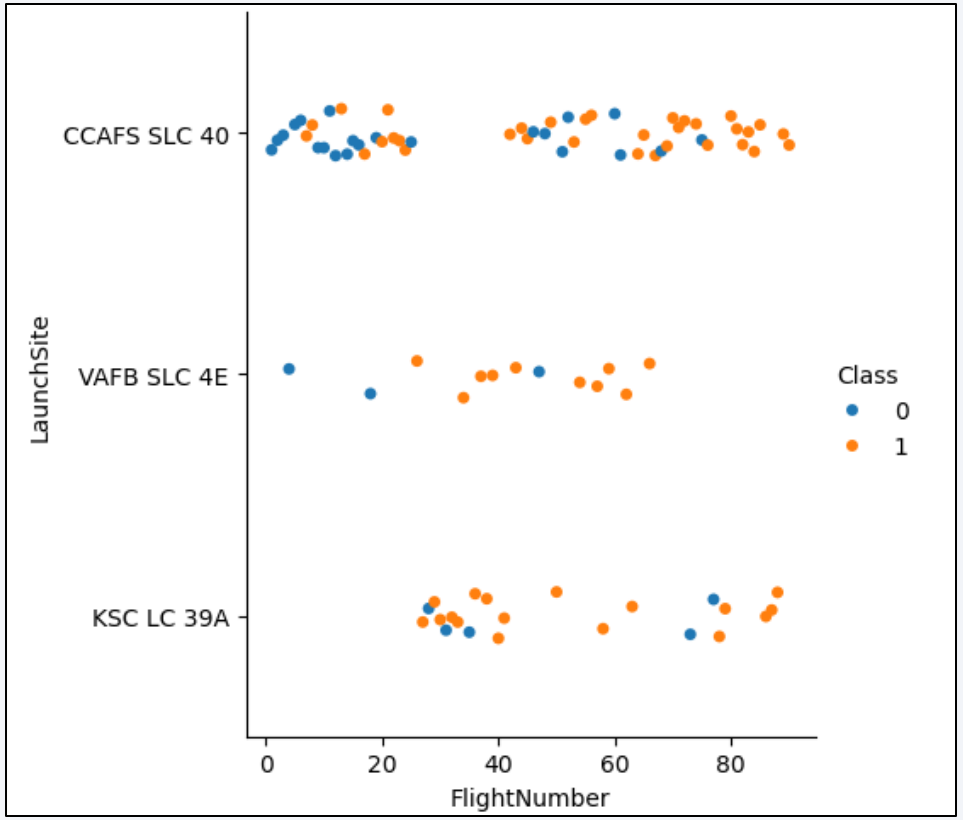
Section 2

Insights drawn from EDA



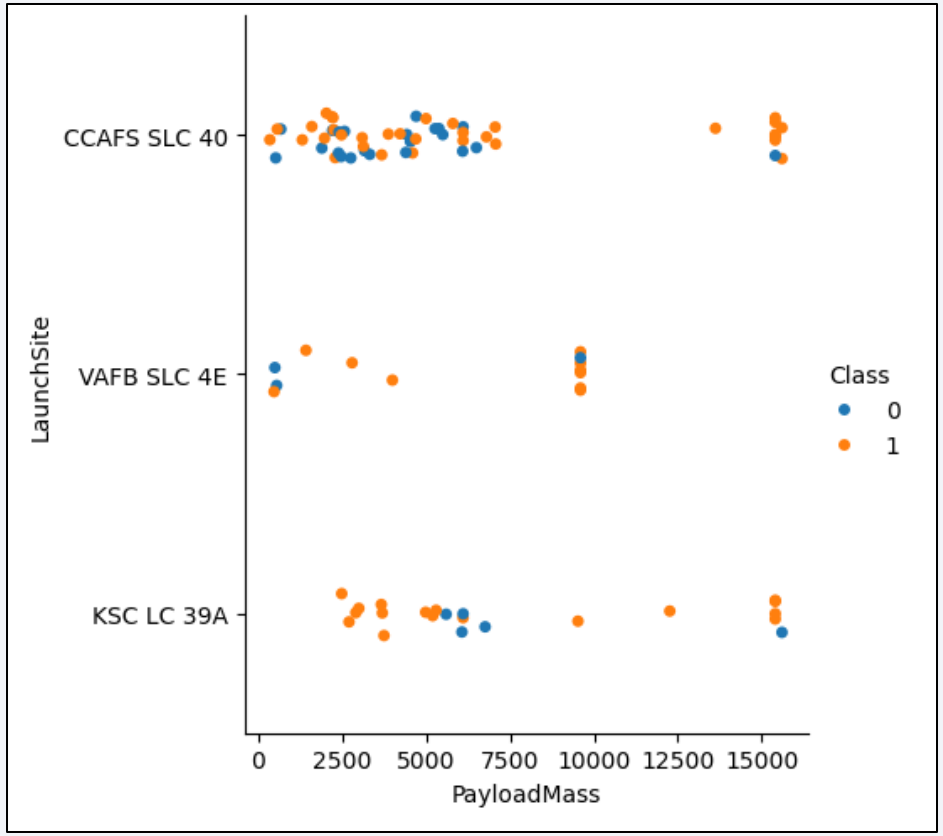
Flight Number vs. Launch Site

- We can see, in any of the launch sites, as the number of flights increases, our mission is more likely to land successfully.



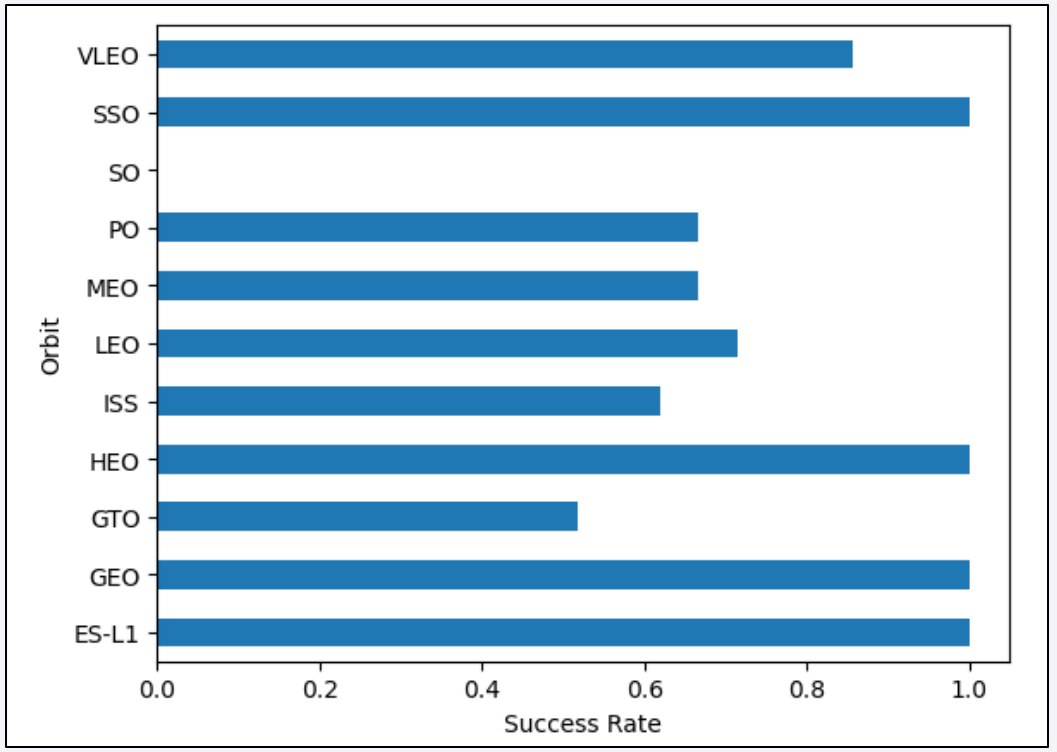
Payload vs. Launch Site

- As we see, the higher the payload, it's most likely to success.
- Also, we can see that there are no rocket launched for heavypayload mass (more than 10k)



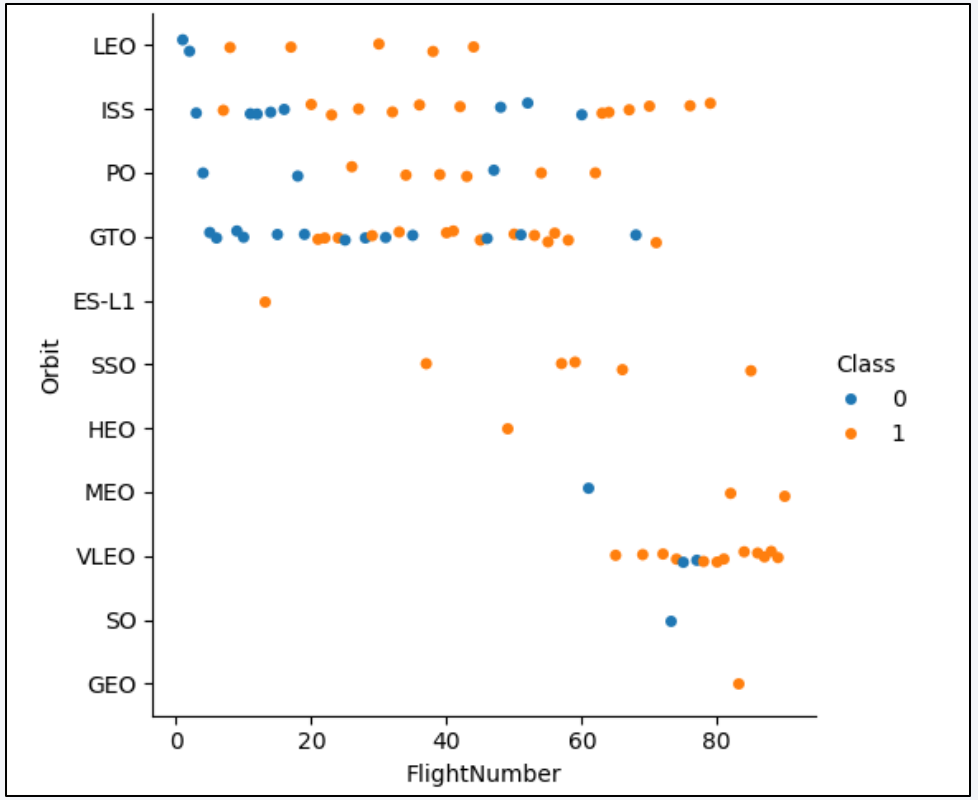
Success Rate vs. Orbit Type

- We can see that the orbits with highest success rate are SSO, HEO, GEO and ES-L1.



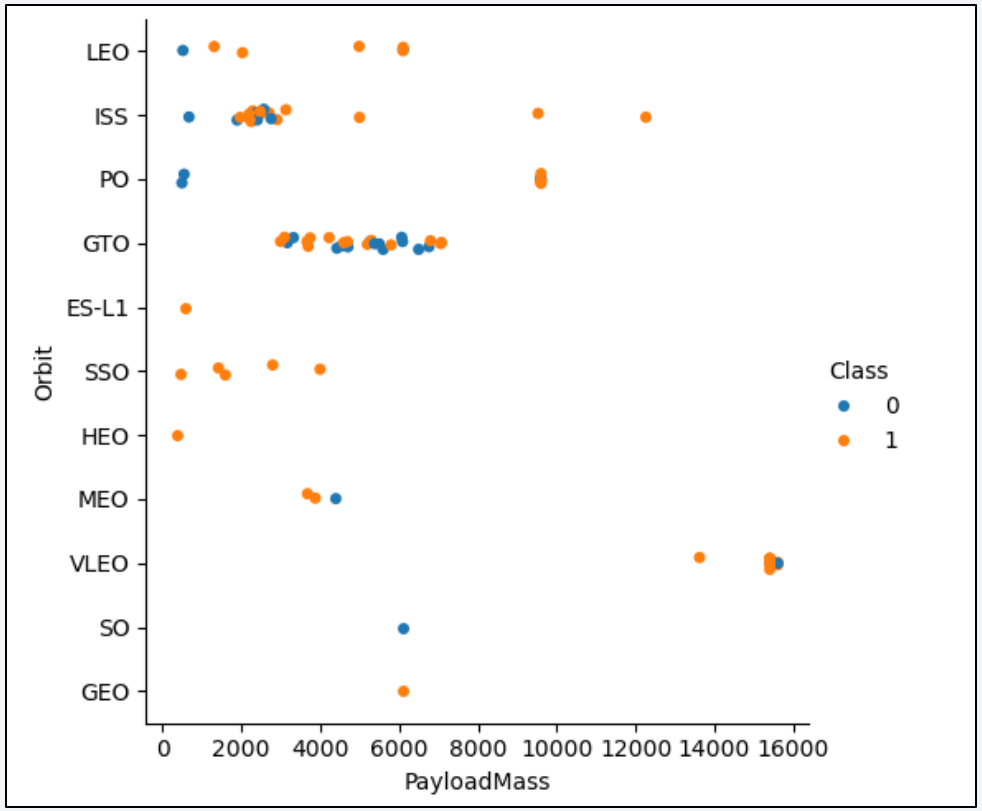
Flight Number vs. Orbit Type

- It shows that as we increase the number of flights, we have more success outcomes.
- As we saw in the previous slide SSO has only successes and SO only failures.



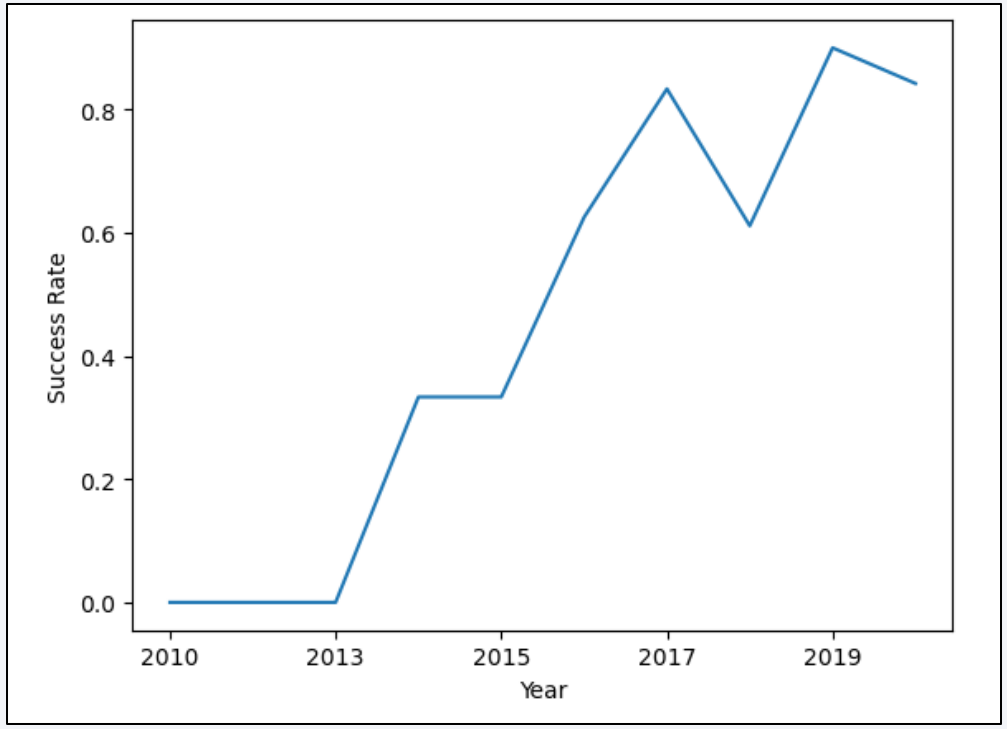
Payload vs. Orbit Type

- We can see that there's no relationship between GTO and payload mass.
- Also, as the payload mass increases, the number successes increases for orbits like LEO, ISS and PO.



Launch Success Yearly Trend

- We can see that success rate has been increasing since 2013, having its peak in 2019.



All Launch Site Names

- Names of the unique launch sites:

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

- Applying the 'Distinct' function to the column 'Launch_Site' in select statement we get the unique values that it includes:
  - Cape Canaveral Air Force Station Launch Complex
  - Vandenberg AFB Space Launch Complex
  - Kennedy Space Center Launch Complex
  - Cape Canaveral Air Force Station Space Launch Complex

Launch Site Names Begin with 'CCA'

- 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	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

- Applying a like condition to Launch_Site column or variable and limiting the result to 5, we get the records whose launch sites begin with `CCA`.

Total Payload Mass

- Total payload carried by boosters from NASA:

TOTAL
45596

- Filtering the column Customer and calculating the sum() of the Payload_Mass__Kg_ column we get that total, i.e. 45,596 kilograms.

Average Payload Mass by F9 v1.1

- Average payload mass carried by booster version F9 v1.:

F9 v1.1 AVG. Payload
2928.4

- By filtering the Booster_version column and applying the function avg() to the Payload column in the select statement we get that 2,928.4 was the average payload mass carried (in kilograms).

First Successful Ground Landing Date

- Date of the first successful landing outcome on ground pad:

MIN(Date)
2010-06-04

- Filtering the column Outcome with the successes and getting the minimum date of the results we get that the first landing with a successful outcome was in 2010/06.

Successful Drone Ship Landing with Payload between 4000 and 6000

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

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

- Filtering both the Outcome and Payload columns we have these Booster versions that matches these restrictions.

Total Number of Successful and Failure Mission Outcomes

- Total number of successful and failure mission outcomes:

Mission_Outcome	COUNT(*)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

- Applying the count(*) function after grouping the entries or samples by the Mission_Outcome column we get all the unique values and their total cases.

Boosters Carried Maximum Payload

- Names of the booster which have carried the maximum payload mass:

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

- By limiting the results of a subquery in which we ordered the Payload_Mass__KG_ in a descending way column we get the Booster Versions with the maximum payload.

2015 Launch Records

- Failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015:

Landing_Outcome	Booster_Version	Launch_Site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- By filtering the Landing_Outcome and Date columns, and selecting the desired ones in the select statement we get these cases, in which the landing failed.

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

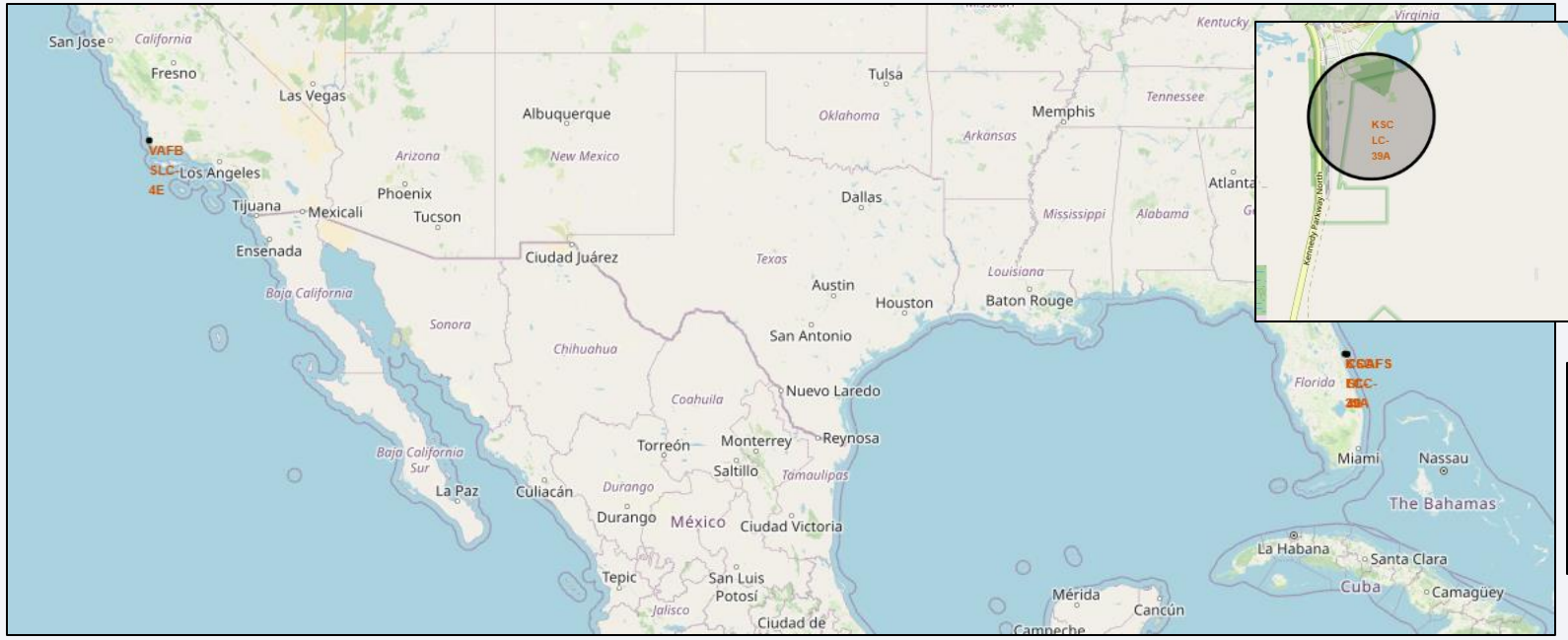
- Ranking of the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order:

Landing_Outcome	Total
No attempt	21
Success (drone ship)	14
Success (ground pad)	9
Failure (drone ship)	5
Controlled (ocean)	5
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

- Grouping by Landing_Outcome column with a filter using a having clause for Date column, applying the count() function to these grouped categories and ordering the results in a descending way.


Section 3
Launch Sites
Proximities Analysis

Landing Sites Location - Markers



	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

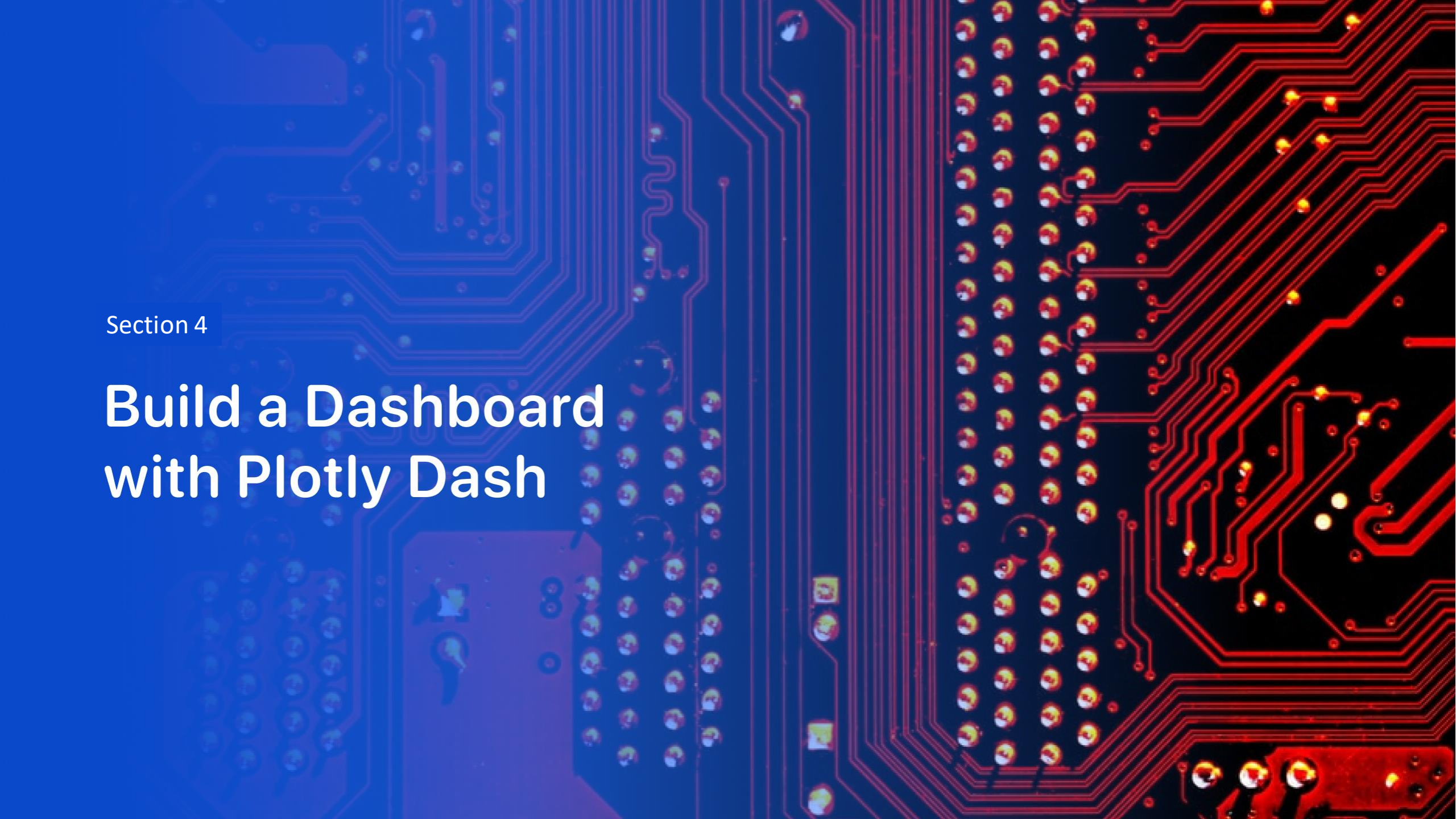
- As we can see in these screenshots, our 4 landing sites are located near to the coast and far enough from cities, which is essential for the well-being of the population if landing result wasn't successful.

Color-labeled launch outcomes

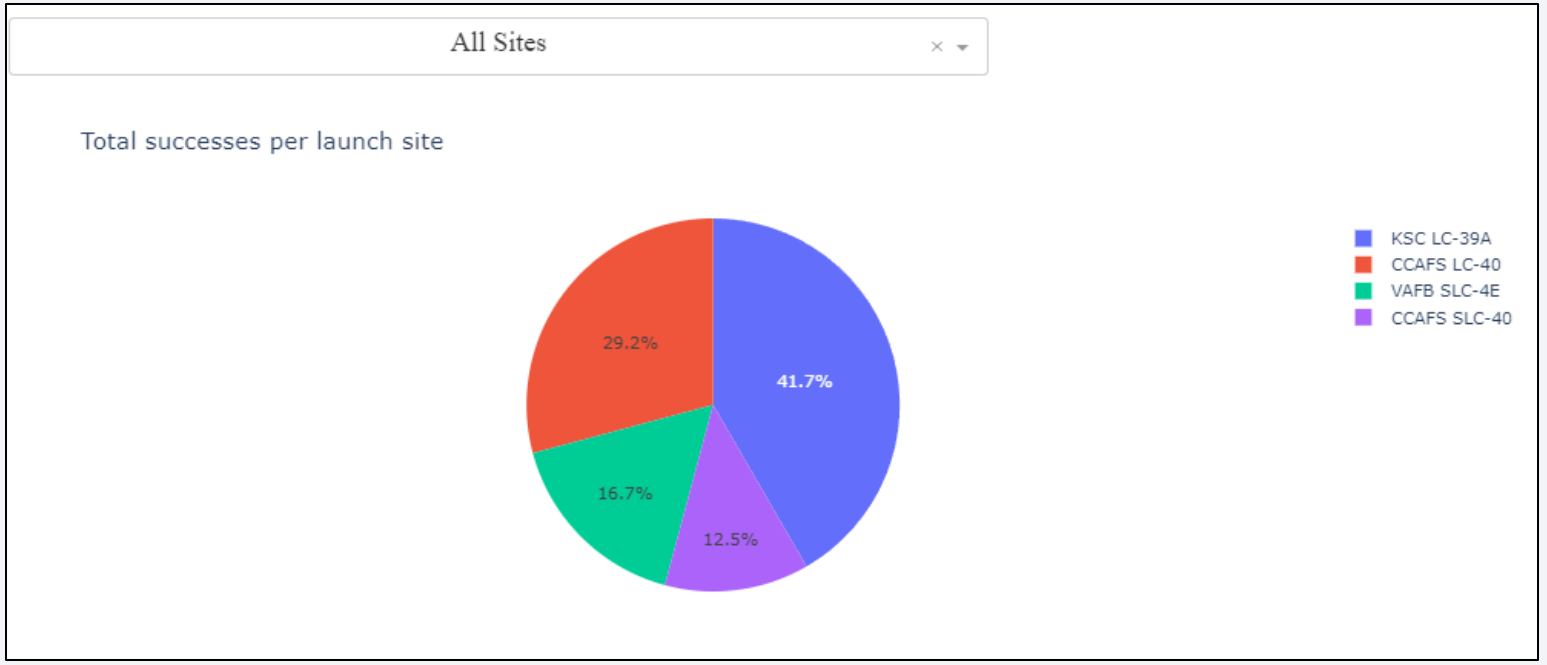
- We can see, on each landing site, the cases in which the launch was either a success or a failure by the color of the marker.

<Folium Map Screenshot 3>

- As we can see and as we mentioned before, these 10 samples in this case (launch tries in VAFB SLC-4E site), are far from the closest cities and close enough the coast.

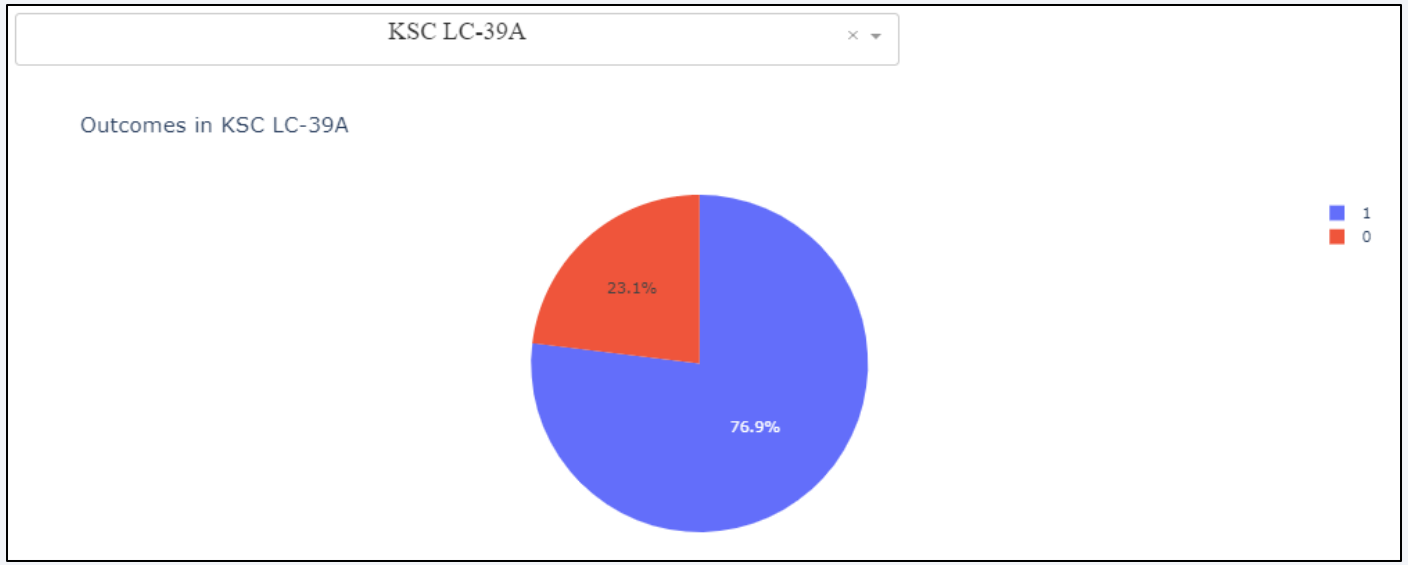

Section 4
Build a Dashboard with Plotly Dash

Total success launches



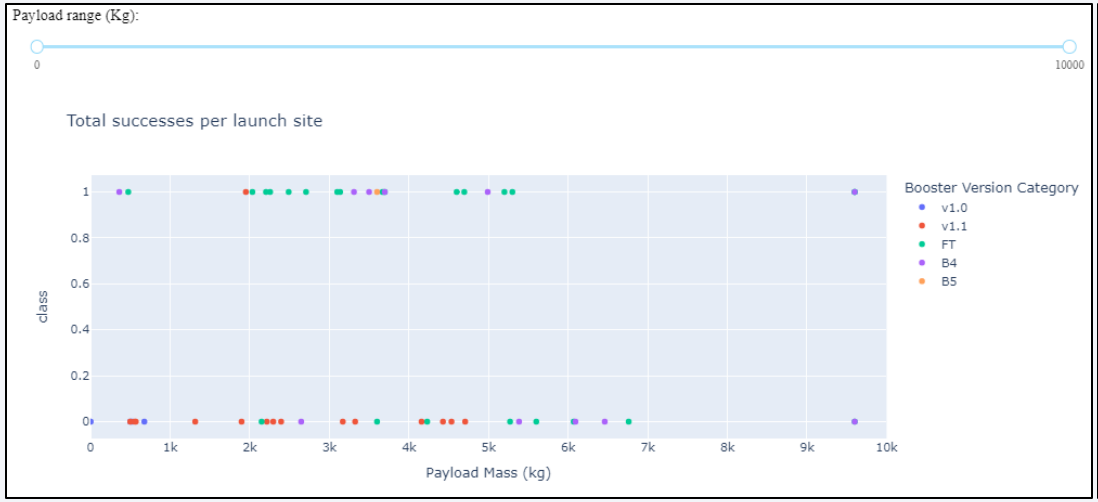
- What we can see in the pie chart is that KSC LC-39A has the highest amount of successes with 41.7 percent, followed by CCAFS LC-40 with a 29.2 percent of the total.

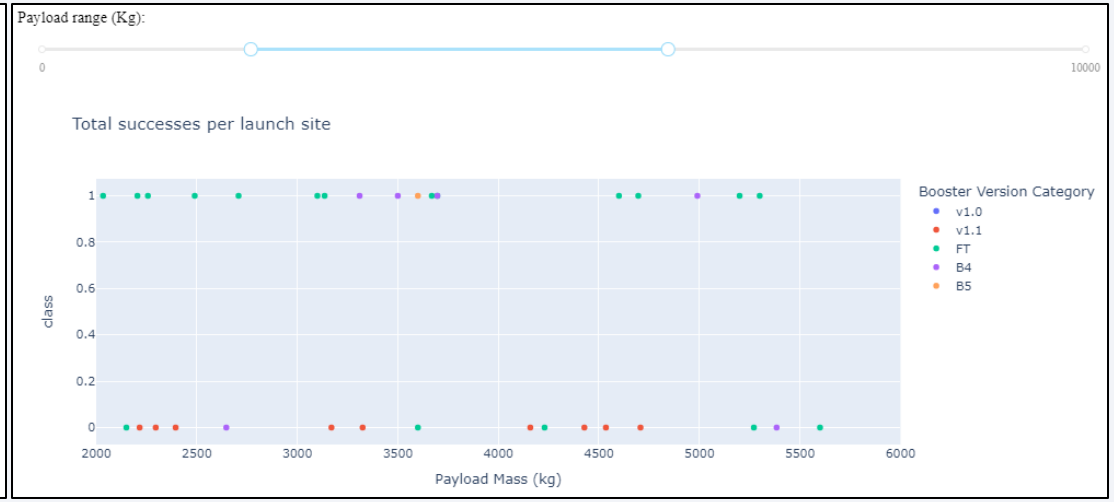
Launch success ratio in the site with most successes



- Being KSC LC-39A the site with most successes, as we saw in the previous slide, we can select it in our dropdown element to see the success ratio in this specific site, which has 76.9 percent of success rate.

Payload vs. Launch Outcome for all sites





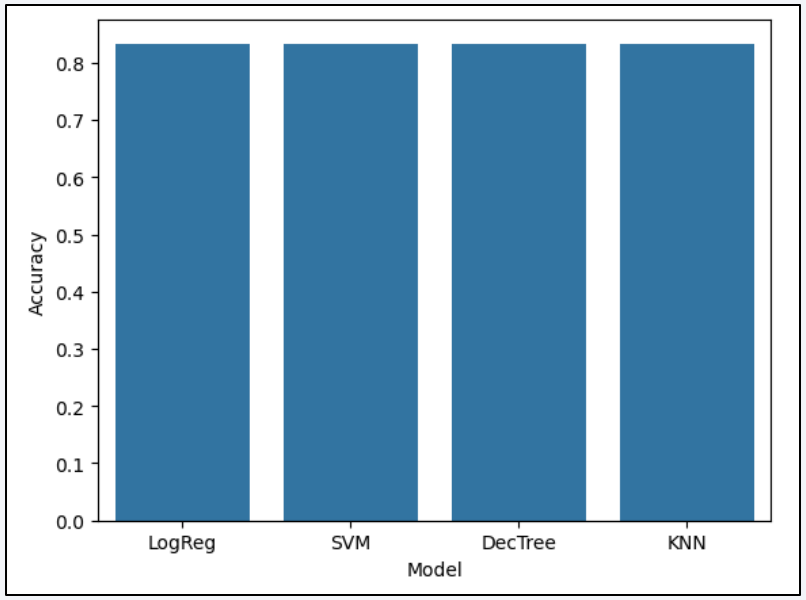
- Having selected ALL in the dropdown element of our dashboard, we then chose diferent ranges for both plots, 0-10k and 2-6k respectively. We also can see the different booster versions as the color of the points.
- We can see that there's few missions landed with more than 7k payload mass.
- We also can see how as we go down in payload mass, the sample or case is more likely to fail.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

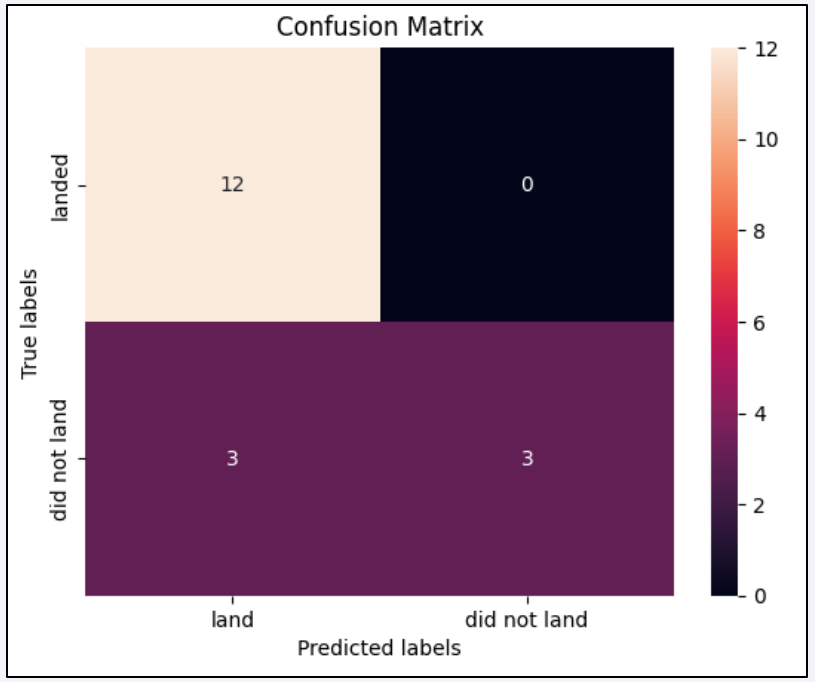
- Built model accuracy for all built classification models, in a bar chart:



- We can see that all the classification models used had the same out-of-sample accuracy.

Confusion Matrix

- Confusion matrix of the best performing model with an explanation:



- As we see, this model predict a success outcome when its true value is also success 100% of the times (12 true positives).
- We also can observe that never predicts fail when the case is a success (0 falses negatives).
- Finally, this model has 3 true negatives and 3 false negatives.

Conclusions

- We were able to predict whether the launch is going to success or fail, in order to get the cost.
- We discovered that every classification method used in this process is has the same accuracy, so we could use any of them, as we wish.
- So, if we wanted to compete with SpaceX, it would be possible thanks to the high accuracy of our model.

Appendix

- Layout of the dashboard, function callbacks and functions that manage it.

```
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': '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'}
                                                ),

                                # 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=10000, 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')),
                                ])

# TASK 2:
```

```
@app.callback(Output(component_id='success-pie-chart', component_property='figure'),
              Input(component_id='site-dropdown', component_property='value'))
def get_pie_chart(entered_site):
    if entered_site == 'ALL':
        total_sites_successes = spacex_df[spacex_df['class']==1].reset_index()
        fig = px.pie(total_sites_successes, values='class',
                     names='Launch Site',
                     title='Total successes per launch site')
        return fig
    else:
        total_outcomes_per_class = spacex_df[spacex_df['Launch Site']==entered_site].groupby('class', as_index=False)['class'].count().reset_index()
        total_outcomes_per_class.columns=['class', 'total']
        fig = px.pie(total_outcomes_per_class,
                     values='total',
                     names='class',
                     title='Outcomes in {}'.format(entered_site))
        return fig

# TASK 4:
# Add a callback function for `site-dropdown` and `payload-slider` as inputs, `success-payload-scatter-chart` as output
@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 get_scatter_chart(entered_site, entered_payload_range):
    if entered_site == 'ALL':
        fig = px.scatter(spacex_df,
                         x='Payload Mass (kg)',
                         y='class',
                         color='Booster Version Category',
                         range_x=entered_payload_range,
                         title='Total successes per launch site')
        return fig
    else:
        filtered_df = spacex_df[spacex_df['Launch Site']==entered_site]
        fig = px.scatter(filtered_df,
                         x='Payload Mass (kg)',
                         y='class',
                         color='Booster Version Category',
                         range_x=entered_payload_range,
                         title='Total successes per launch site')
        return fig
```

Thank you!