Winning Space Race Through Data Science

By Arturo Franco
October 11, 2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API/Web Scraping, Data Wrangling, Exploratory Data Analysis with SQL/ Data Vis, Interactive Visuals with Folium, and ML Predictions
- Summary of all results
  - Exploratory Data Analysis Results
  - Interactive Visuals Results
  - Predictive Analytical Results

Introduction

- Project background and context

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars. While other companies cost up to 165 million dollars each, so much of the savings is because SpaceX can reuse the first stage on launches. Therefore, if we can determine if the first stage will land, we will be able predict the cost of a SpaceX launch. This information can be used if an opposing company wants to go against SpaceX for other reasons. The goal of the project was to create a machine learning pipeline to predict if the first stage will land successfully to back up the company’s reputation in its money saving goal.

- Problems we will want to find answers to

-What factors determine if the rocket will land successfully?
- The features that determine the success rate of a successful landing.
- Operating conditions needs, that must to be in place to ensure a successful landing program.

Section 1: Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from the links related to the matter.
- Perform data wrangling
  - For categorical features, One-hot encoding was applied
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and PlotlyDash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

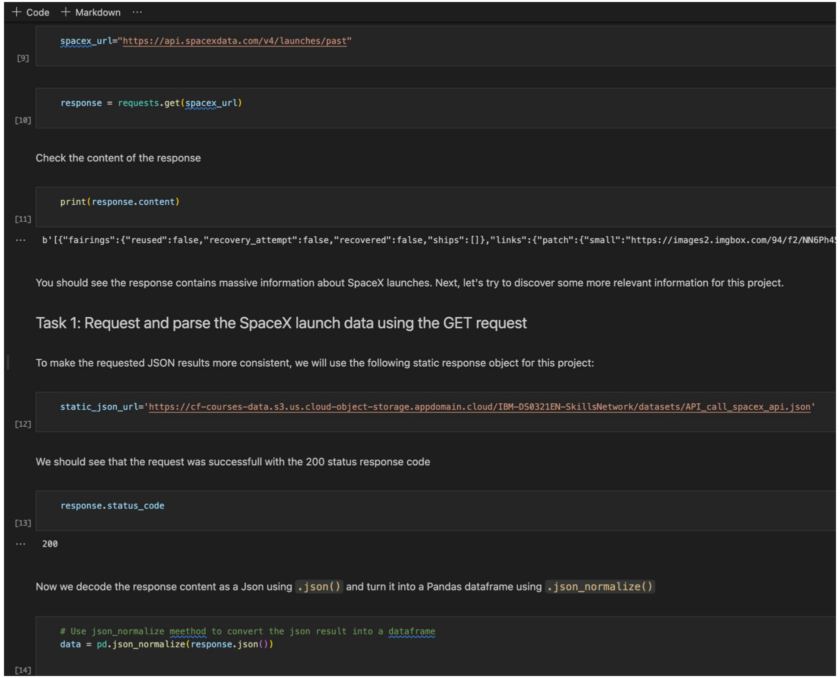
Data Collection

The data was collected using various methods

- Data collection was done using get request to the SpaceX API (getter syntax).
- We then decoded the response content as a Json (.json() function call) and turned it into a pandas dataframe (with .json_normalize()).
- Then cleaned the data, checked for missing values and fill in missing values where the means of the corresponding columns.
- We then performed web scraping from certain links for Falcon 9 launch records with BeautifulSoup. To find the launch records as a HTML table, to then parse the table and convert it to a pandas df for further analysis.

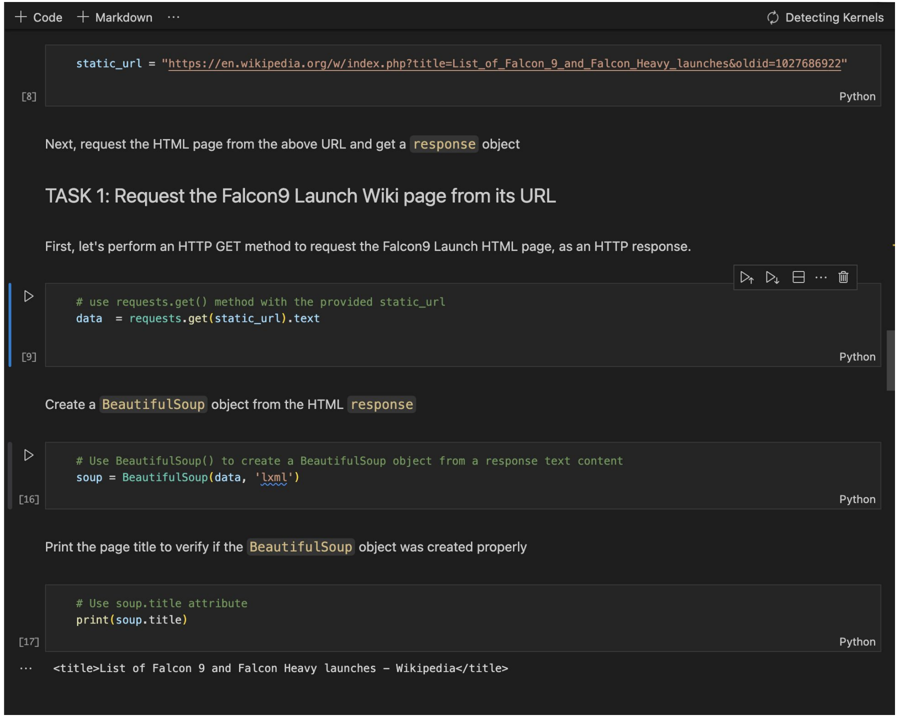
Data Collection - SpaceX API

- We used the get request to the SpaceX API to collect data, then we cleaned the requested data and data formatting.
- The link to the github file is: https://github.com/arturoathome/Final-Project-IBM-Data-Scientest-Certifciate/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



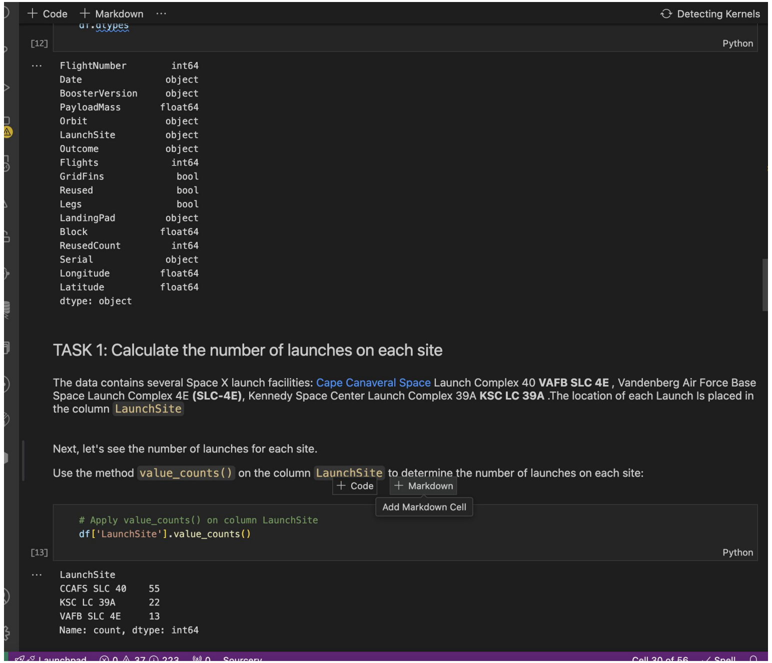
Data Collection - Scraping

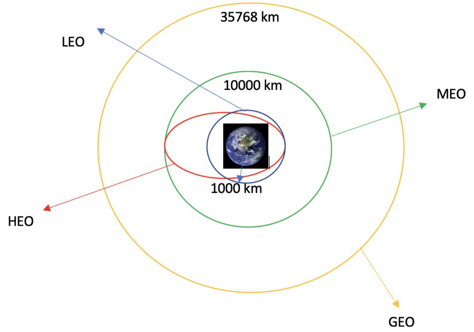
- We applied web scraping to webscrape Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.
- The link to the github file is: https://github.com/arturoathome/Final-Project-IBM-Data-Scientest-Certifciate/blob/main/jupyter-labs-webscraping.ipynb



Data Wrangling

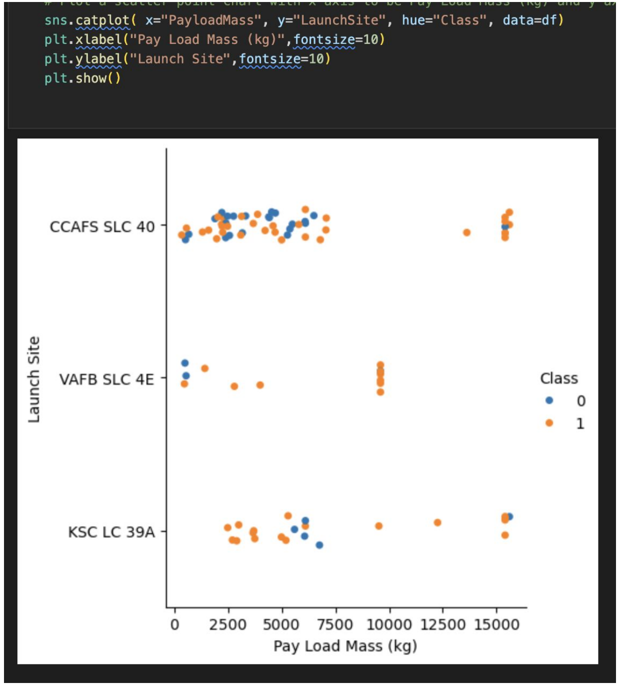
- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the github file is: https://github.com/arturoathome/Final-Project-IBM-Data-Scientest-Certifciate/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb





EDA with Data Visualization

- We explored the data by visualizing factors like flight number and payload mass, flight number and launch site, success rate of each orbit type, flight number and orbit type, and many more.
- Link to the github: https://github.com/arturoathome/Final-Project-IBM-Data-Scientest-Certifciate/blob/main/edadataviz.ipynb

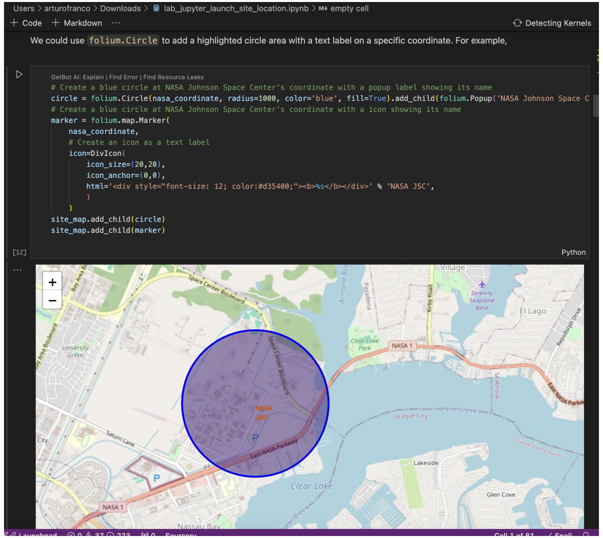


EDA with SQL

- We loaded the SpaceX dataset into a SQL database without leaving the jupyter notebook (using SQL Magic).
- We then used EDA with SQL to get insight from the data. Wrote queries to find out the following:
    - The names of unique launch sites in the space mission
    - The total payload mass carried by boosters launched by NASA (CRS)
    - The average payload mass carried by booster version F9 v1.1
    - The total number of successful and failure mission outcomes
    - The list of names of booster version that can carry max load
- The link to the github is: https://github.com/arturoathome/Final-Project-IBM-Data-Scientest-Certifciate/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

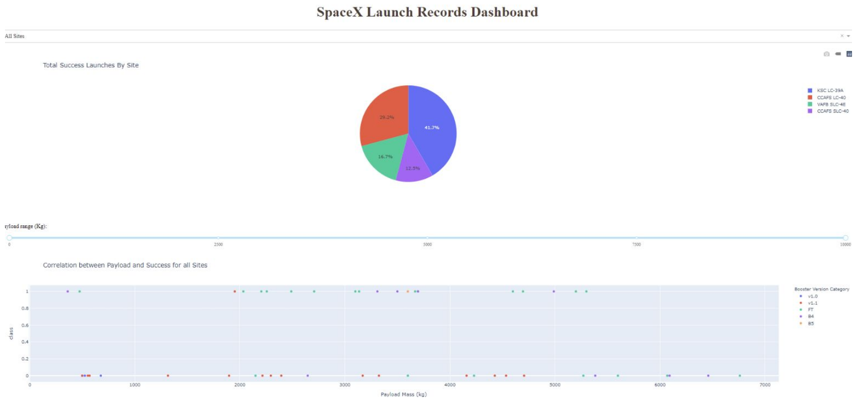
Build An Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- Then assigned the launch outcomes to class 0 for failure and 1 for success.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
    - Are launch sites near railways, highways and coastlines?
    - Do launch sites keep certain distance away from cities?
- Link to github is: https://github.com/arturoathome/Final-Project-IBM-Data-Scientest-Certifciate/blob/main/lab_jupyter_launch_site_location.ipynb



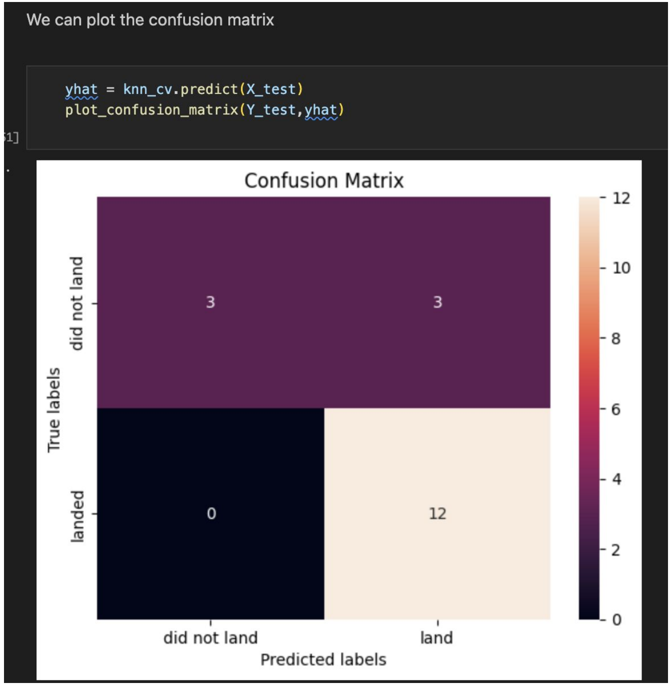
Build A Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly Dash.
- We plotted pie charts showing the total launches by a certain sites.
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for booster version.
- The link to github is: https://github.com/arturoathome/Final-Project-IBM-Data-Scientest-Certifciate



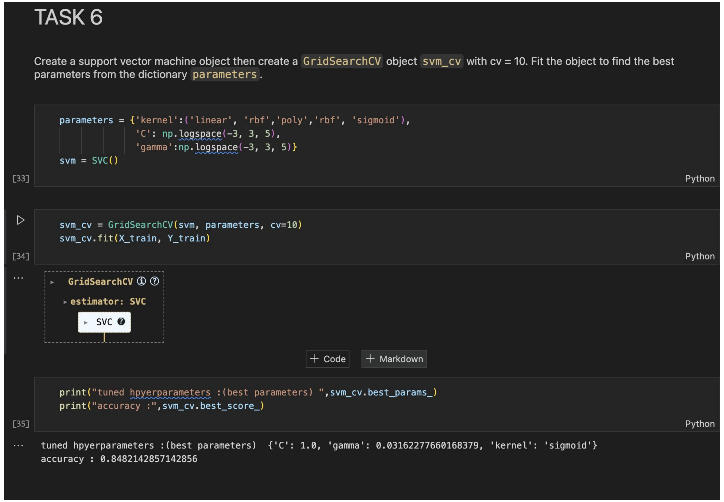
Predictive Analysis (Classification)

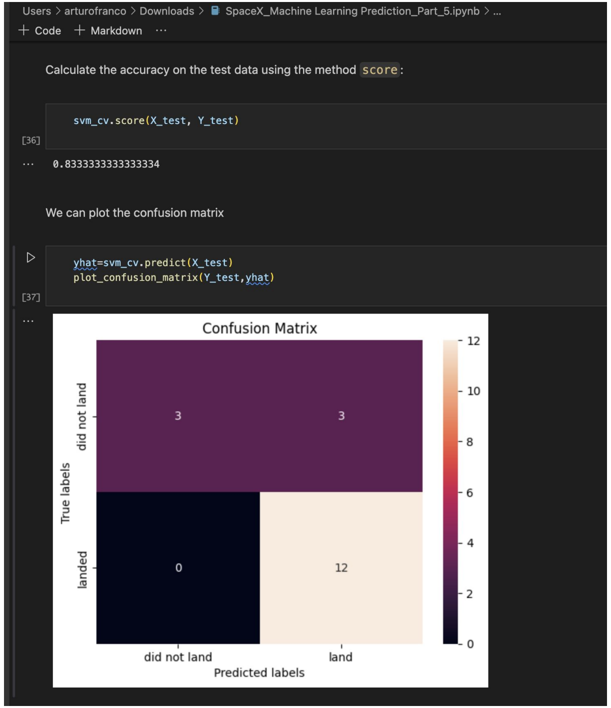
- Loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- Then built different machine learning models and tune different hyperparameters using GridSearchCV.
- Used accuracy of the tree as the metric for our model, improved the model by adjusting the parameters.
- The link to the github is: https://github.com/arturoathome/Final-Project-IBM-Data-Scientest-Certifciate/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb



Results

- Exploratory data analysis results from the predictive models when training and testing variables.
- Predictive analysis results are shown in the photos on the right





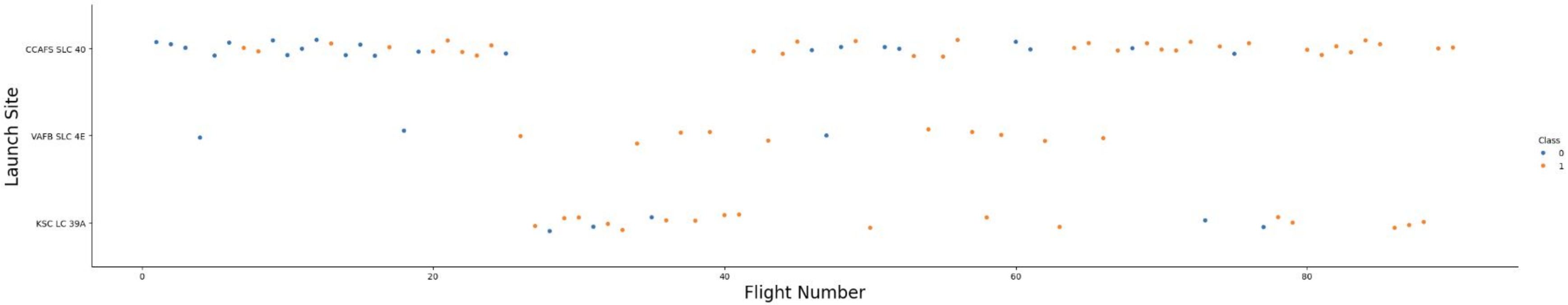
Section 2: Insights Drawn from EDA

Flight Number vs. Launch Site

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

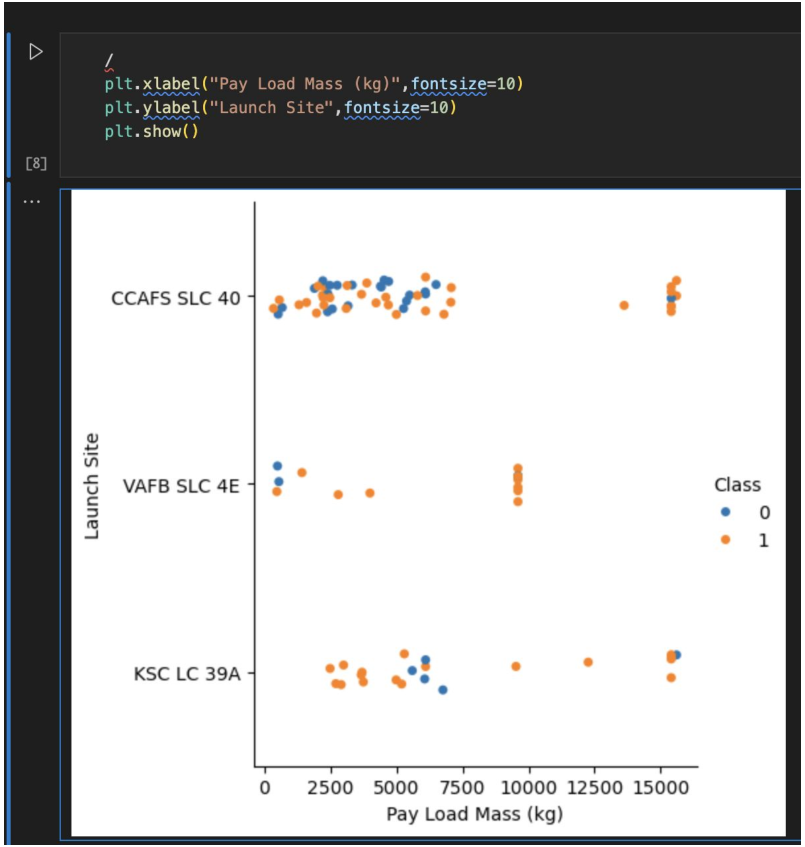
```python
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class va
sns.catplot( x="FlightNumber", y="LaunchSite", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

Pytho



Payload vs. Launch Site

- It was found that the greater the payload for launch, the higher the success rate for launch at the sites CCADS SLC 40 & VAFB SLC 4E

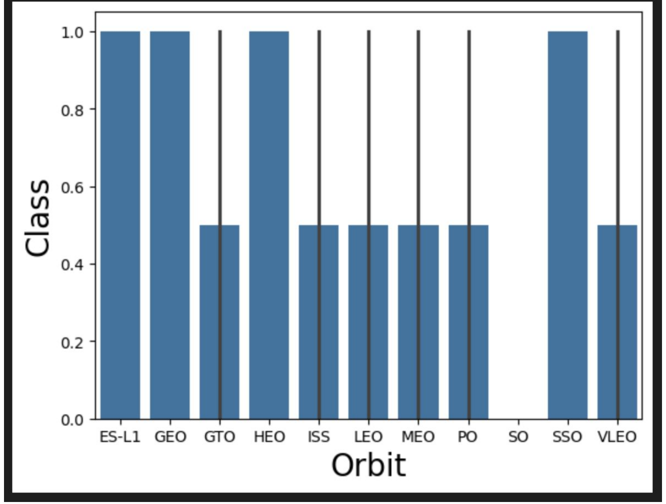


Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, and SSO had the most success rate.
- The rest of them were not as successful or not successful at all, like SO.

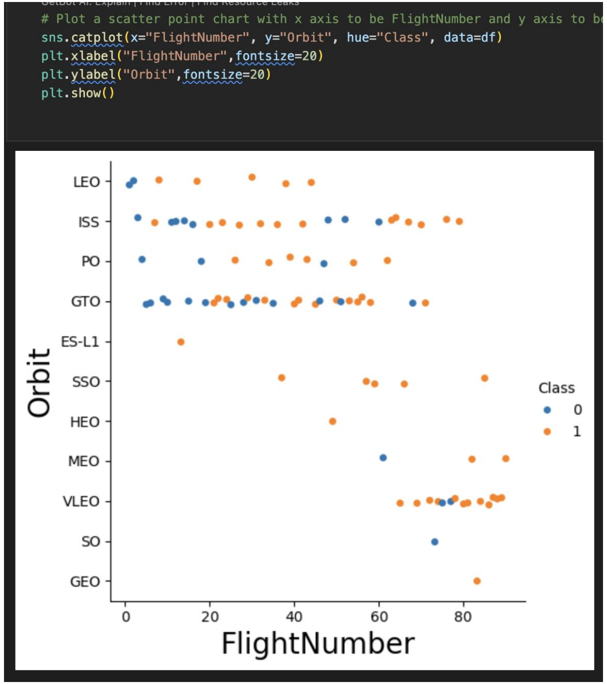
GetBot AI: Explain | Find Error | Find Resource Leaks

```
# HINT use groupby method on Orbit column and get the mean of Class column
t = df.groupby(['Orbit', 'Class'])['Class'].agg(['mean']).reset_index()
sns.barplot(x="Orbit", y="Class", data=t)
plt.xlabel("Orbit",fontsize=20)
plt.ylabel("Class",fontsize=20)
plt.show()
```



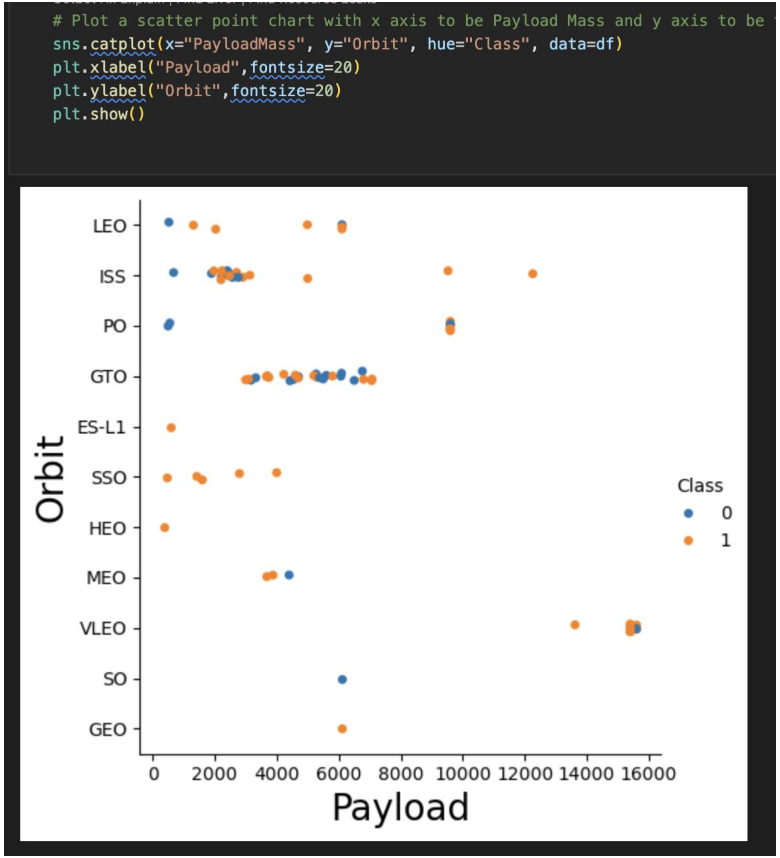
Flight Number vs. Orbit Type

- The plot below shows that in the VLEO orbit success is related to the higher number of flights
- While on the GTO orbit, there is no relationship between flight number and the orbit since there is a split with successful and unsuccessful launches.



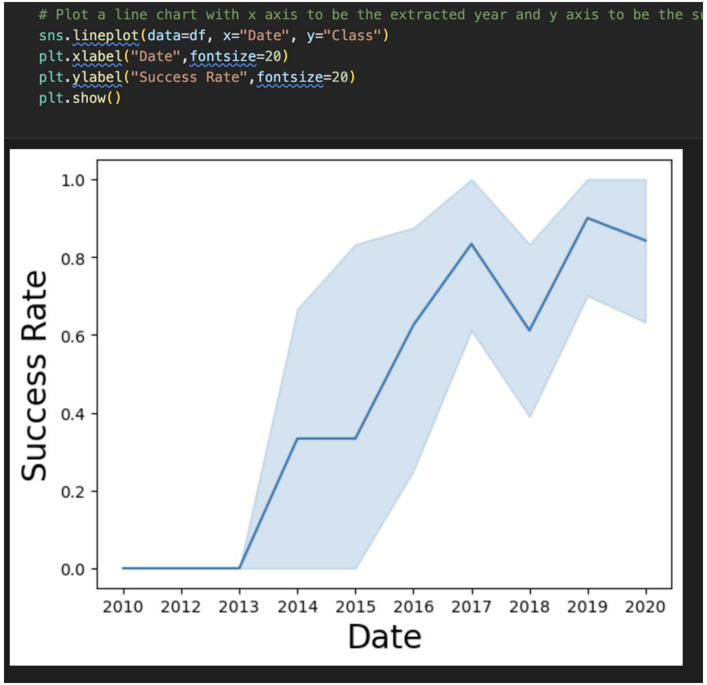
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for VLEO, PO, LEO, and ISS orbits.
- More failures are found in less heavy payloads, specifically under 8,000 usually.



Launch Yearly Success Trend

- From the plot, we can observe that success rate since 2013 kept on increasing till 2017 with a slight dip.
- But picked back up for 2018 to go towards a positive upward manner once again.



All Launch Site Names

- We used the key function DISTINCT to show only unique launch sites from the SpaceX data. So we can know the total number of sites present in the database.

```
%%sql
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL;
```

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

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

Launch Site Names that begin with `CAA’

- We used the query on the right to display the first 5 records where launch sites begin with `CCA`

```
%%sql SELECT LAUNCH_SITE
    FROM SPACEXTBL
    WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- The total payload carried by boosters from NASA was calculated as 45596 using the query on the right.

```
%%sql SELECT SUM(PAYLOAD_MASS__KG_)
      FROM SPACEXTBL
      WHERE Customer = 'NASA (CRS)';
```

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- It was calculated that the average payload mass carried by booster version F9 v1.1 as 2534.7

```
%%sql Select AVG(PAYLOAD_MASS__KG_)
      FROM SPACEXTBL
      WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

AVG(PAYLOAD_MASS__KG_)
2534.6666666666665

First Successful Ground Landing Date

- It was found that the date of the first successful landing outcome on ground pad was December 22, 2015

```
%%sql SELECT MIN(Date)
      FROM SPACEXTBL
      WHERE Landing_Outcome = 'Success (ground pad)';
```

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

MIN(Date)
2015-12-22

Successful Drone Landing with Payload between 4,000 and 6,000

- A specific WHERE clause was used to filter for boosters which have successfully landed on drone ship and used the AND condition to find successful landings with payload mass greater than 4000 but less than 6000

```
%%sql SELECT Booster_Version
      FROM SPACEXTBL
      WHERE Landing_Outcome = 'Success (drone ship)' AND 4000 < PAYLOAD_MASS__KG_ < 6000;
```

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

Booster_Version
F9 FT B1021.1
F9 FT B1022
F9 FT B1023.1
F9 FT B1026
F9 FT B1029.1
F9 FT B1021.2
F9 FT B1029.2
F9 FT B1036.1
F9 FT B1038.1
F9 B4 B1041.1
F9 FT B1031.2
F9 B4 B1042.1
F9 B4 B1045.1
F9 B5 B1046.1

Total Number of Successful and Failure Missions Outcomes

- As you can see we used the COUNT function to count each kind of outcome in the database.
- There were 99 cases of success and 1 failure found

```
%%sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS TOTAL
      FROM SPACEXTBL
      GROUP BY MISSION_OUTCOME;
```

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

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

Boosters Carried Maximum Payload

- It was determined the list of boosters that have carried the maximum payload using a subquery in the WHERE clause and the MAX() function to find such list

```
%%sql SELECT DISTINCT BOOSTER_VERSION
      FROM SPACEXTBL
      WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

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

- Combinations of the WHERE clause, AND, and SUBSTR conditions were used to filter for failed landing outcomes in drone ships, their booster versions, and launch site names for year 2015

```
%%sql SELECT SUBSTR(DATE,6,2) AS MONTH, LANDING_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE
      FROM SPACEXTBL
      WHERE Landing_Outcome = 'Failure (drone ship)' AND SUBSTR(DATE,0,5) = '2015';
```

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

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

Ranking Ocean Landing Outcomes between 2010-06-04 and 2017-03-20

- We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 to 2010-03-20.
- We applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcome in descending order.

GetBot AI: Explain | Find Error | Find Resource Leaks

```
%%sql SELECT LANDING_OUTCOME, COUNT(LANDING_OUTCOME) AS TOTAL
      FROM SPACEXTBL
      WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
      GROUP BY LANDING_OUTCOME
      ORDER BY TOTAL DESC
```

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

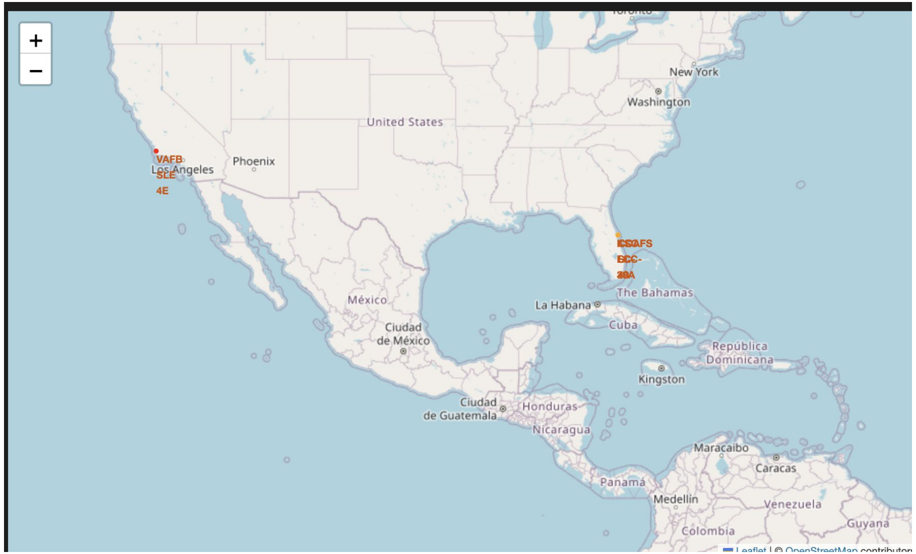
Landing_Outcome	TOTAL
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3: Launch Sites Proximities Analysis

All Launch Site Locations

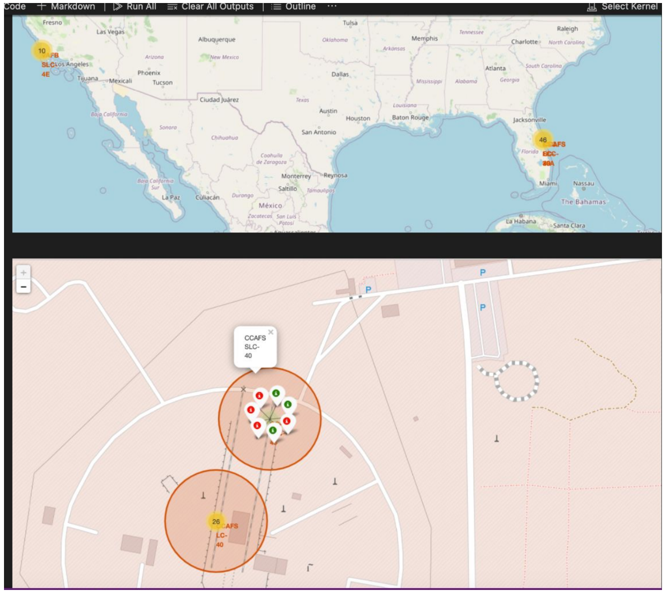
```python
vafb_coordinates = [34.750144,-120.521294]
ksc_coordinates = [28.573255,-80.646895]
ccafs_coordinates = [28.562302,-80.577356]
ccafs_slc_coordinates = [28.563197,-80.576820]

vafb_circle = folium.Circle(vafb_coordinates, radius=1000, color='red', fill=True).add_child(folium.Popup('Vandenberg Spac
vafb_marker = folium.map.Marker(vafb_coordinates, icon=DivIcon(icon_size=(20,20),icon_anchor=(0,0),html='<div style="font-
site_map.add_child(vafb_circle)
site_map.add_child(vafb_marker)

ksc_circle = folium.Circle(ksc_coordinates, radius=1000, color='#green', fill=True).add_child(folium.Popup('Kennedy Space
ksc_marker = folium.map.Marker(ksc_coordinates, icon=DivIcon(icon_size=(20,20),icon_anchor=(0,0),html='<div style="font-si
site_map.add_child(ksc_circle)
site_map.add_child(ksc_marker)

ccafs_circle = folium.Circle(ccafs_coordinates, radius=1000, color='yellow', fill=True).add_child(folium.Popup('Cape Canav
ccafs_marker = folium.map.Marker(ccafs_coordinates, icon=DivIcon(icon_size=(20,20),icon_anchor=(0,0),html='<div style="fon
site_map.add_child(ccafs_circle)
site_map.add_child(ccafs_marker)

ccafs_slc_circle = folium.Circle(ccafs_slc_coordinates, radius=1000, color='orange', fill=True).add_child(folium.Popup('Ca
ccafs_slc_marker = folium.map.Marker(ccafs_slc_coordinates, icon=DivIcon(icon_size=(20,20),icon_anchor=(0,0),html='<div st
site_map.add_child(ccafs_slc_circle)
site_map.add_child(ccafs_slc_marker)
```

Python



Shows Sites At Stations

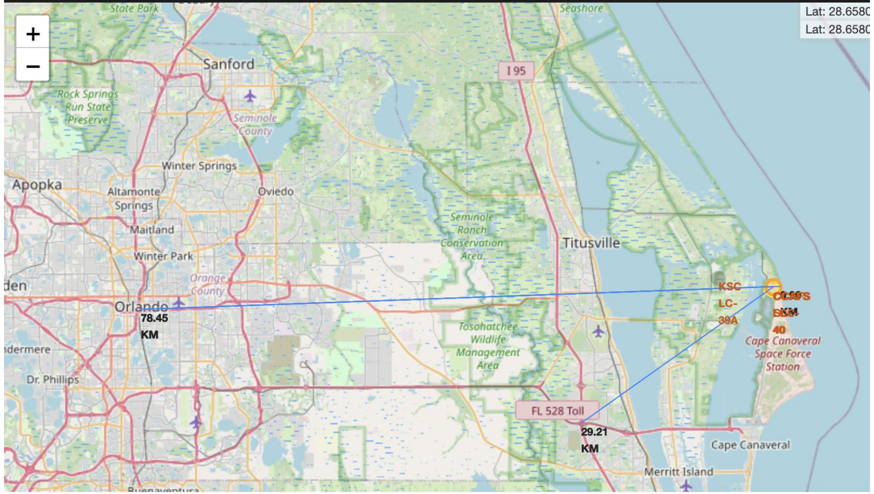
- Here are the launch sites present at Cape Canaveral and shows the count of the others in the West Coast.



Distance of Certain Locations From Site

- Here you can see under the location the distance in KM it is from the Site with a line connected the two.

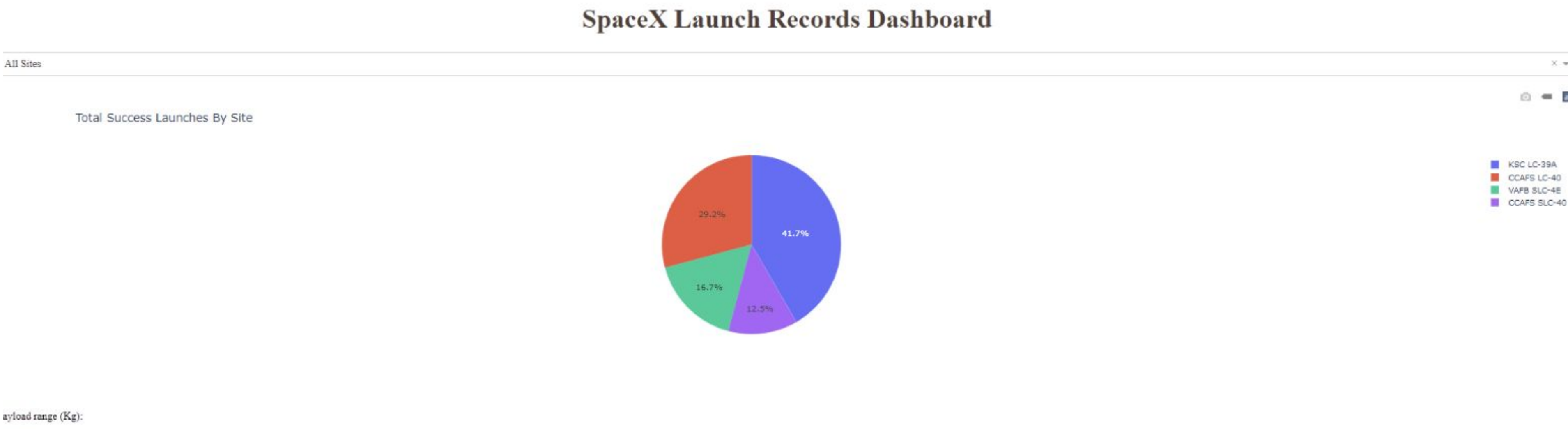
```
coordinates = [
    [28.56342, -80.57674],
    [28.5383, -81.3792]]

lines=folium.PolyLine(locations=coordinates, weight=1)
site_map.add_child(lines)
distance = calculate_distance(coordinates[0][0], coordinates[0][1], coordinates[1][0], coordinates[1][1])
distance_circle = folium.Marker(
    [28.5383, -81.3792],
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#252526;"><b>%s</b></div>' % "{:10.2f} KM".format(distance),
        )
    )
site_map.add_child(distance_circle)
site_map
```



Section 4: Build a Dashboard within Plotly

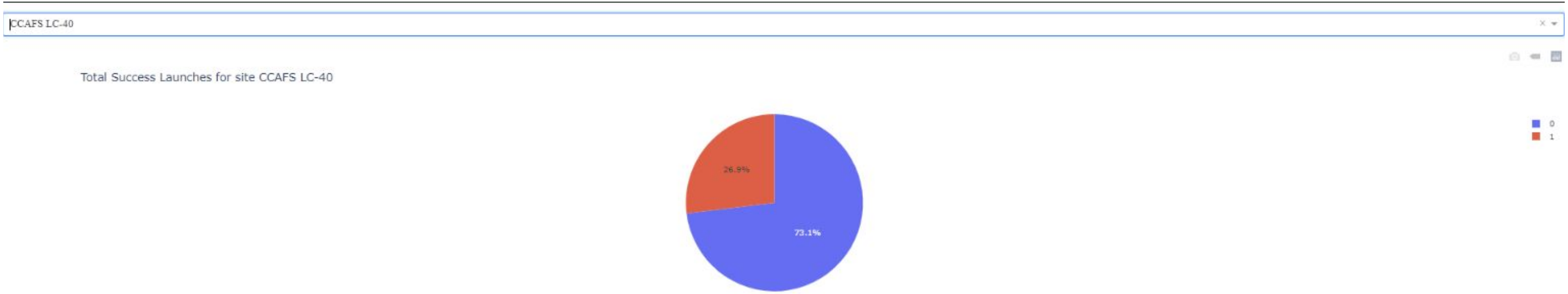
Total Success Launches for Sites

- This pie chart shows success rate via percentages for each launch site so its more visually impactful rather than looking at a table.



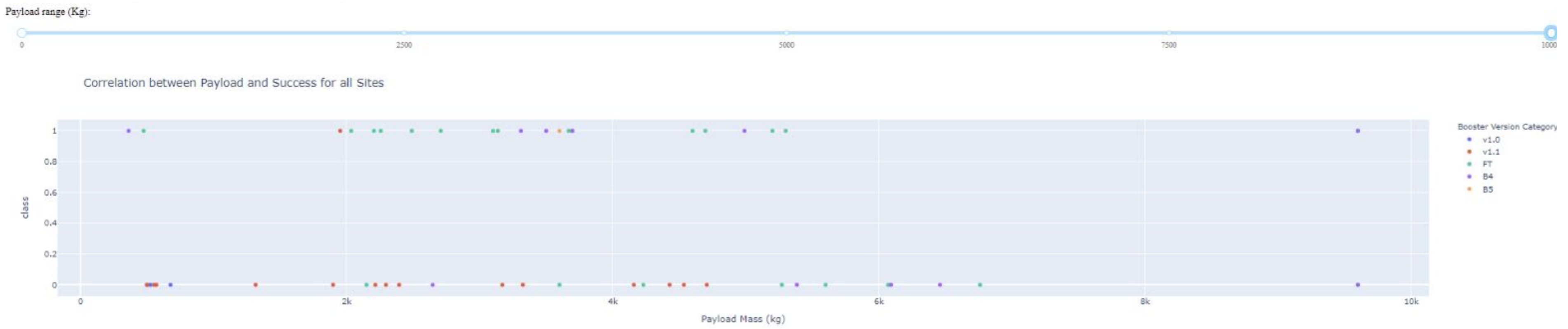
Highest Launch Success Ratio for a Site

- This is a pie chart to show the success rate of CCAFS LC-40 site and the purple is successful landing and red is failed ones.



Payload vs. Launch Outcome with Plot

- You can see here through a scatter plot how the different booster version handle different payload mass or weight at the sites.



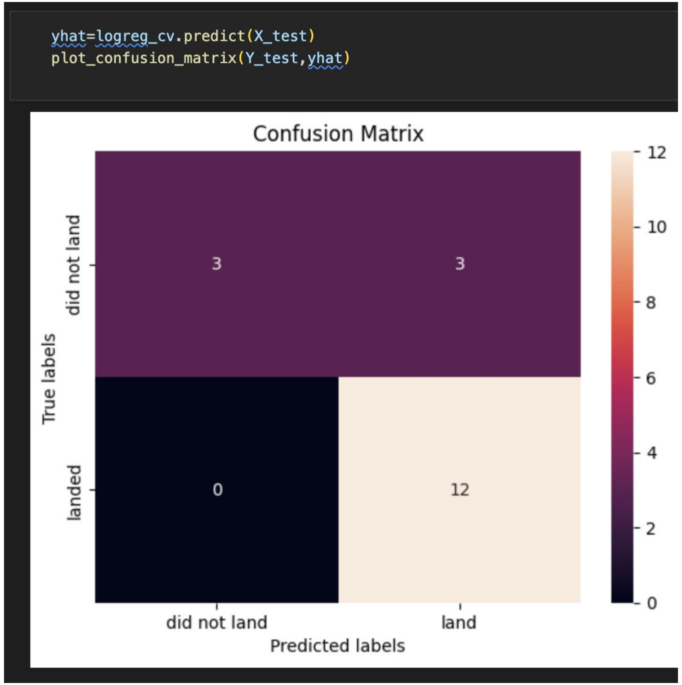
Section 5: Predictive Analysis (Classification)

Classification Accuracy

- The bar chart shows how they all have roughly the same performance when testing the accuracy so the bar chart is not that drastic in difference when seen on the side.
- The tree model had the least score while the others all had the same being .83 out of one so those were the results shown.

Confusion Matrix

- Every model did the same (0.83) except the tree model being 0.78 so therefore there were three models like the one beside this.



Conclusions

- It was concluded that:
  - The larger the flight amount (payload mass) at the site, the greater the success rate for that launch.
  - Launch success rate increased drastically in 2013 till present.
  - It was seen that ES-L1, GEO, HEO, and SSO had the most success rate.
  - The Decision tree classifier is the least best machine learning algorithm for this task since it got the least score.
  - The other models got the same higher score (0.83)

Appendix

- Everything used in this presentation was done and taken from my project conducted from the IBM Data Scientist Certification.
- The corresponding github repository for all the work is here: https://github.com/arturoathome/Final-Project-IBM-Data-Scientest-Certifciate

Thank You! For Reading my Presentation