IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Egon Rosa Pereira
27th January 2024

Outline

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

Executive Summary

Data Science has become a competitive differentiator for space agencies in achieving ambitious goals.

Essentially, the importance of data analysis in optimizing space missions is explored, from collecting critical information to making agile decisions.

Rocket launches by SpaceX are relatively cheaper than its competitors due to the reuse of the first stage of its rockets. The cost of launching the Falcon 9 rocket is $62 million while other suppliers cost more than $165 million.

The present work focuses on analyzing the success in reusing the first stage of rockets launched into space in order to arrive at a predictive model that determines the success rate of this launch strategy.

Introduction

The first stage of a rocket does most of the work in its launch and is much larger than the second stage.

Therefore, the first stage is very large and expensive. The first stage reuse strategy clearly brings savings in new launches.

However, launching rockets using this strategy does not always occur successfully.
Sometimes the first stage doesn't arrive. Sometimes some type of failure occurs during landing just before or after touching the ground.

There are several factors that influence the success rate of this type of launch, such as payload, orbit type, landing location, among others.
All these variables that influence the success rate of reusing the first stage of a rocket need to be taken into account when creating a predictive machine learning model.

The next slides detail all the steps taken to create a first version of a predictive model that evaluates the success rate of reuse in the first stage.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

Data for training and evaluating the predictive model was collected from two main sources:

- SpaceX API
- Wikipedia

The SpaceX API returns responses in JSON format and was used as the main data source to carry out this work.

While a Wikipedia page titled "List of Falcon 9 and Falcon Heavy launches" was used as an additional data source through the use of web scraping technique (the html page is downloaded and then the relevant data from the various tables and elements on the page are extracted and cleaned).

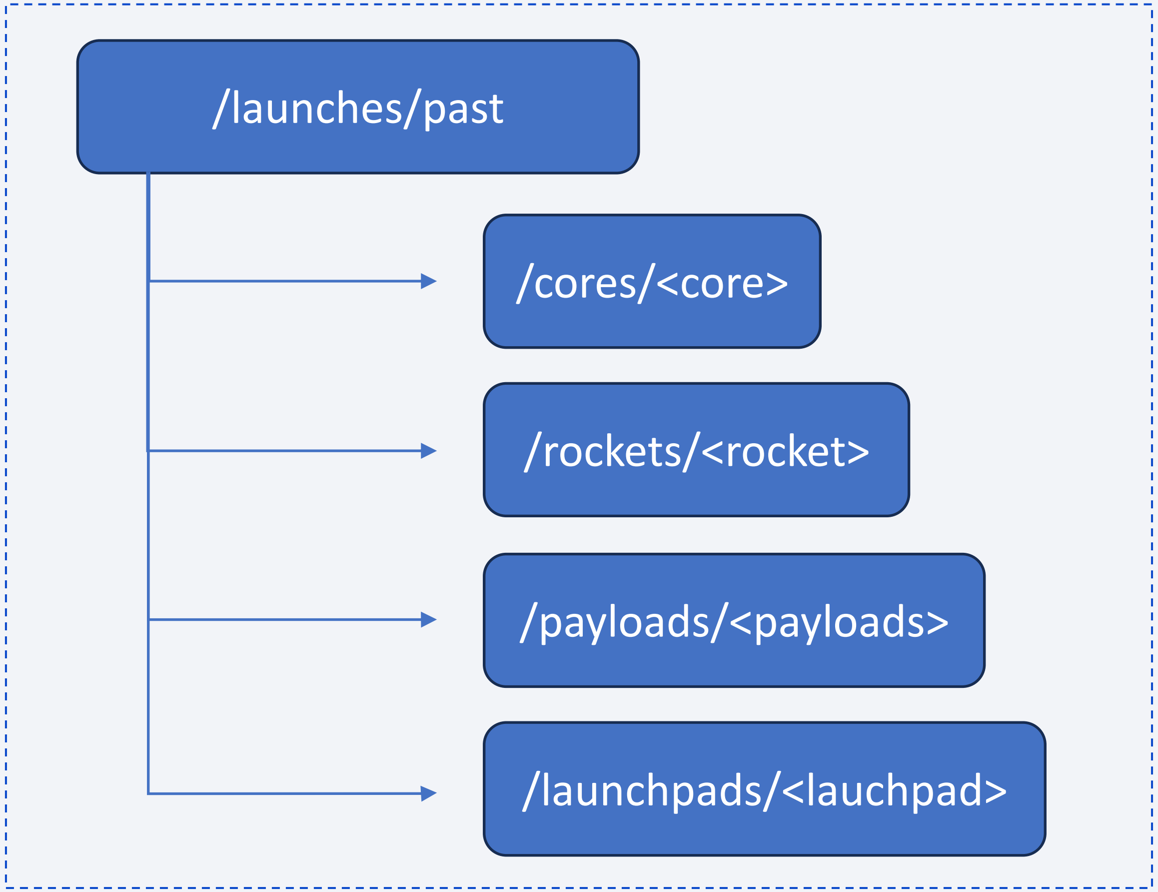
Data Collection – SpaceX API

- Many requests are made to the SpaceX API until the chosen features are retrieved.
- The basic SpaceX API URL is as follows:

https://api.spacexdata.com/v4

Please open the SpaceX API calls notebook for better understanding:

https://github.com/egonrp/certificacaodsfase10/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



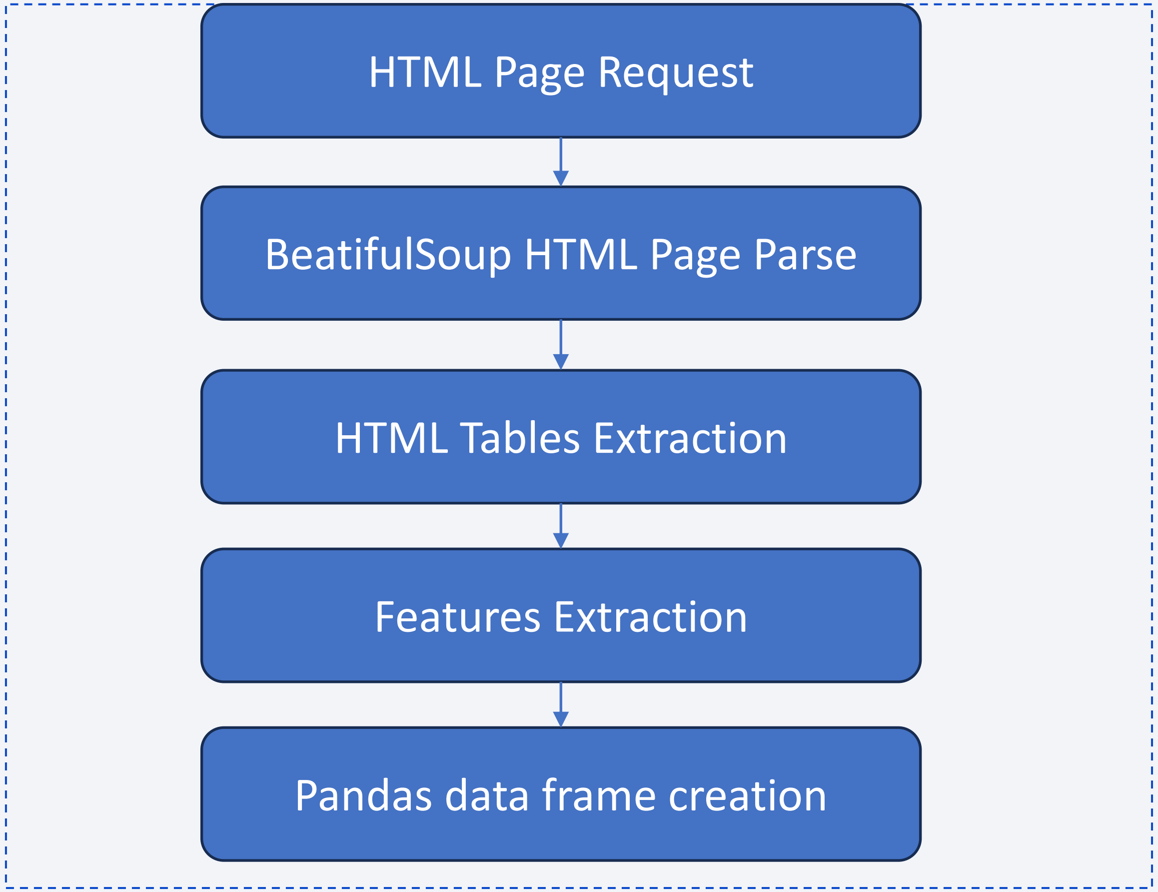
Data Collection - Scraping

- The web scrap of Falcon 9 launch records stored in a HTML table are extracted and converted to a Pandas data frame.
- The Wiki page URL is as follow:

https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

Please open the web scraping notebook for better understanding:

https://github.com/egonrp/certificacaodsfase10/blob/main/jupyter-labs-webscraping.ipynb



Data Wrangling

- First it was necessary to identify the fields with missing values.
- Then identify the numeric and categorical columns of the dataset.
- The values of the relevant features are analyzed for normalization.
- And finally, it was necessary to identify values in the Outcome field that represent success and failure and then transform them into a binary class (1 or 0).

Please, open the data wrangling notebook for better understanding:

https://github.com/egonrp/certificacaodsfase10/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Line charts were used to verify numerical data with progression history as success rate of first stage rocket reuse.
- Bar charts were used to understand the magnitude of some categorical data as success rate by orbit type.
- Scatter point charts were used to help identify patterns and outliers in the analyzed features as relationships between flight numbers and launch sites, payload mass and orbit type, and other patterns.

Please, open the EDA with data visualization notebook for better understanding:

https://github.com/egonrp/certificacaodsfase10/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

Several SQL queries were used to better understand the data:

- Display of the names of the unique launch sites in the space mission
- Display of the total payload mass carried by boosters launched by NASA (CRS)
- Display of the average payload mass carried by booster version F9 v1.1
- List of the date when the first successful landing outcome in ground pad was achieved
- List of the names of the boosters which have success in drone ship and respective payload
- List of the total number of successful and failure mission outcomes, etc

Please, open the EDA with SQL notebook for better understanding:

https://github.com/egonrp/certificacaodsfase10/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- The rocket launch sites present in the dataset were added to the Folium interactive maps. Furthermore, NASA Johnson Space Center location.
- Red and green icons have also been added to the rocket launch sites markers to indicate the number of successes and failures in first stage of rocket reuse missions.
- In addition, straight lines were added to measure the distance between different points of interest from the launch site, such as coast lines, railways, highways and cities
- The circles, markers, lines and icons were added to the folium.plugins.MarkerCluster object, which works as an intermediate layer, or directly to the folium.Map object.

Please, open the interactive map notebook for better understanding*:

https://github.com/egonrp/certificacaodsfase10/blob/main/lab_jupyter_launch_site_location.jupyterlite.ipynb

* GitHub does not display the generated maps, it is necessary to download the notebook and open it locally.

Build a Dashboard with Plotly Dash

The dashboard created with Plotly Dash consists of displaying two interactive charts:

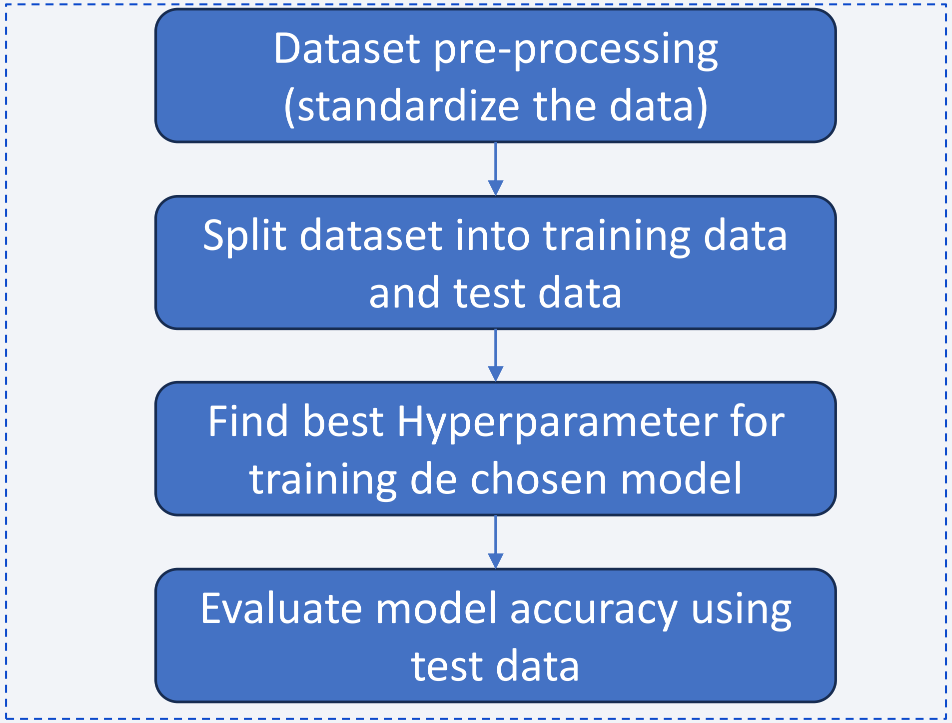
- A pie chart to show the total successful launches count for all sites. If a specific launch site was selected, show the Success vs. Failed counts for the site.
- A scatter chart to show the correlation between payload and launch success. If a specific launch site was selected the data will be filtered. Additionally, a slider component is present to filter mass payload data range.

Please, open Plotly Dash lab source code for better understanding:

https://github.com/egonrp/certificacaodsfase10/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- After Exploratory Data Analysis (EDA) and determination of the training labels, the stage of training the predictive model for the success rate of reusing the first stage of a rocket finally arrived.
- Four machine learning models were chosen to be validated: KNN, Decision Tree, SVM and Logistic Regression. The models were built, evaluated and improved until finding the classification model with the best performance for the test dataset (see the diagram alongside).



Please, open the predictive analysis lab notebook for better understanding:

https://github.com/egonrp/certificacaodsfase10/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

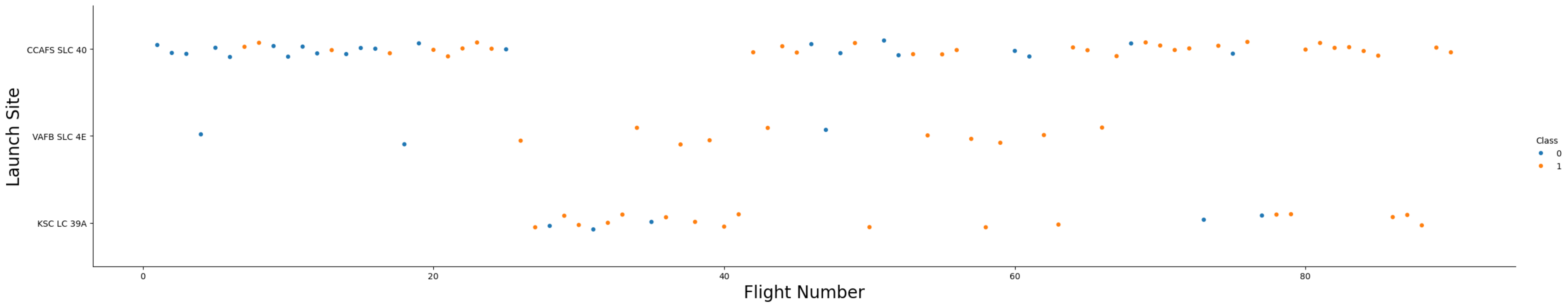
Results

- During the Exploratory Data Analysis (EDA) we discovered a strong relationship between the success in reusing the first stage of rockets and some variables, they are: Flight Numbers, Orbit type, Payload Mass, Booster Version and Launch Site.
- Predictive analysis results
- The Machine Learning algorithms K-Nearest Neighbors (KNN), Decision Tree, Logistic Regression and Support Vector Machine (SVM) were chosen to create the predictive model. Therefore, predictive analysis was carried out based on these models.
- The resulting accuracy of the models with the test data was very close, with a technical tie most of the time during the evaluation of the best hyper-parameters, close to 83%. The Decision Tree model on some occasions reached 88% accuracy, beating the others.
- Follow the results of the experiments and studies carried out in the next slides. ;)

Section 2

Insights drawn from EDA

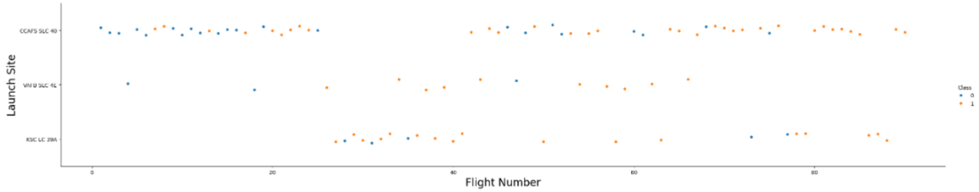
Flight Number vs. Launch Site



- We see that as the flight number increases, the first stage is more likely to land successfully.

In [13]:

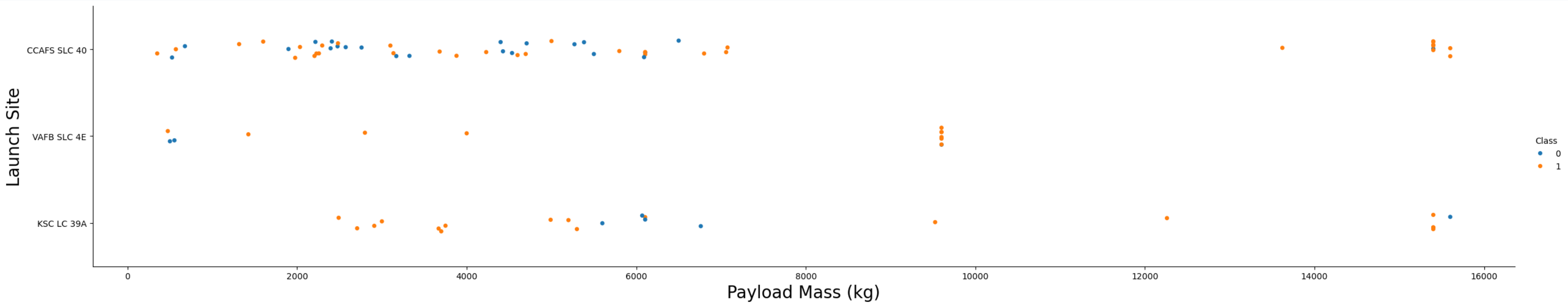
```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and h

sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()

print("* Answer: We see that as the flight number increases, the first stage is more likely to land
```




```
* Answer: We see that as the flight number increases, the first stage is more likely to land successfu
lly.
```

Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

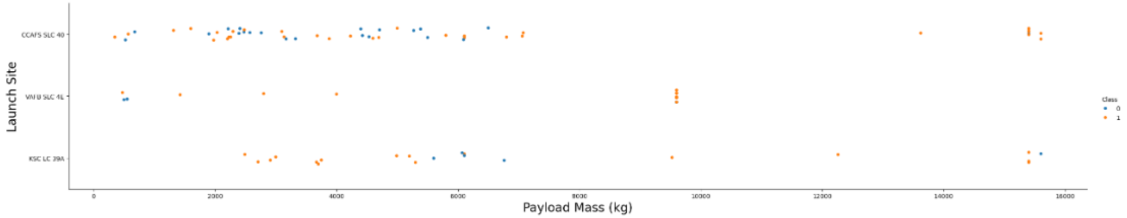
Payload vs. Launch Site



- Note that VAFB-SLC launch site there are no rockets launched for heavy payload mass (greater than 10000).

In [9]:

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site,

sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Payload Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



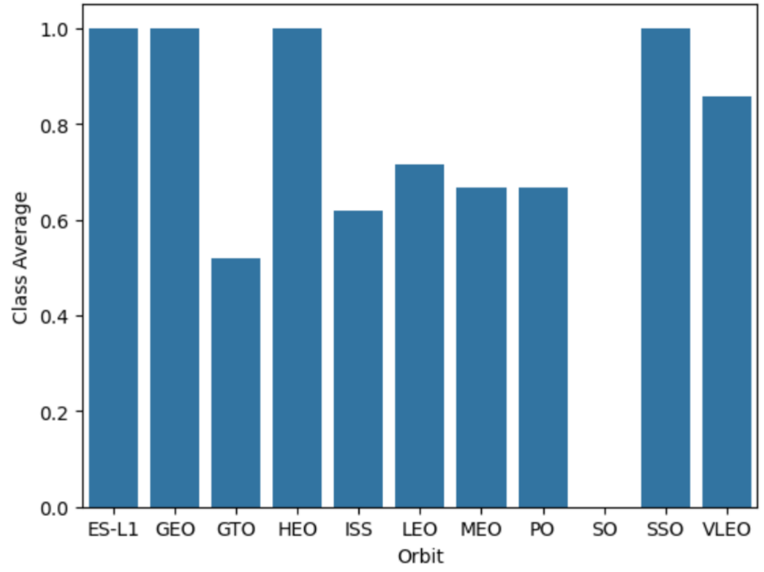
Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

Success Rate vs. Orbit Type

- We see that ES-L1, GEO, HEO and SSO orbits has 100% of success and SO orbit never was successfully.

In [17]:

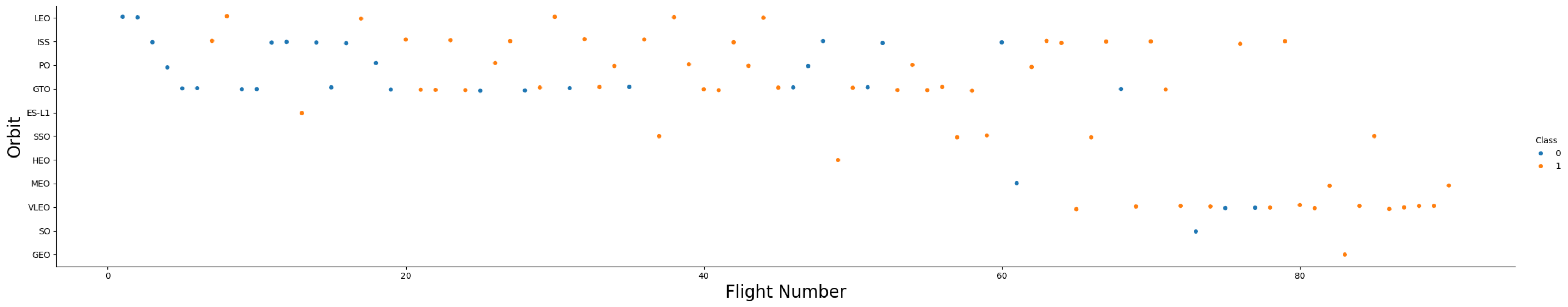
```
# HINT use groupby method on Orbit column and get the mean of Class column

df_grp = df[["Orbit", "Class"]].groupby(["Orbit"], as_index=False).mean()

sns.barplot(x='Orbit',y='Class', data=df_grp)
plt.xlabel("Orbit")
plt.ylabel("Class Average")
plt.show()

print("* Answer: We see that ES-L1, GEO, HEO and SSO orbits has 100% of success and SO orbit never w
```




```
* Answer: We see that ES-L1, GEO, HEO and SSO orbits has 100% of success and SO orbit never was succes
sfully.
```

Analyze the ploted bar chart try to find which orbits have high sucess rate.

Flight Number vs. Orbit Type



- You should 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.

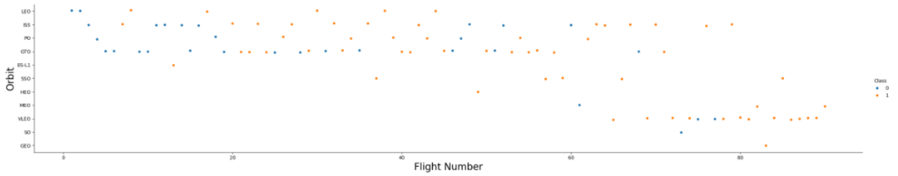
In [18]:

```
### TASK  4: Visualize the relationship between FlightNumber and Orbit type
```

For each orbit, we want to see if there is any relationship between FlightNumber and Orbit type.

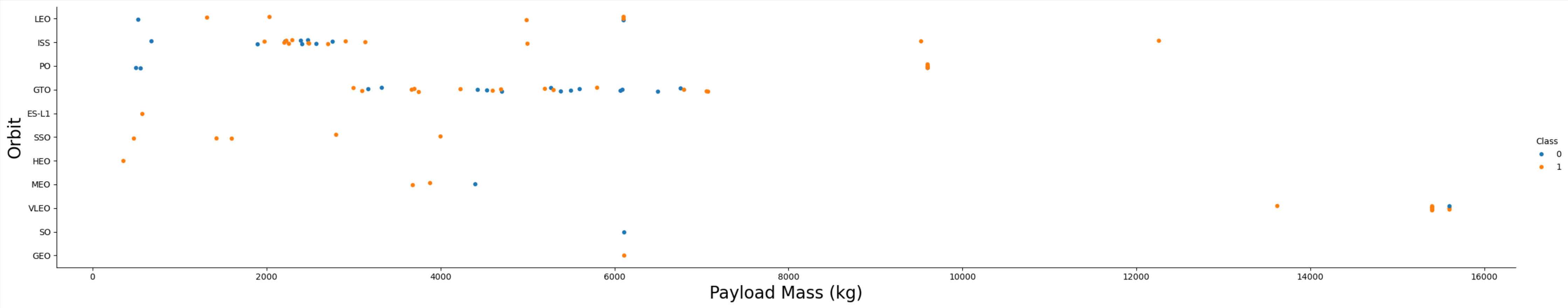
In [19]:

```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to b

sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



You should 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.

Payload vs. Orbit Type

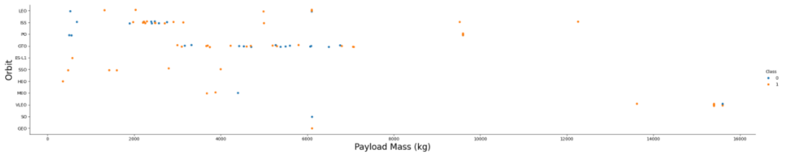


- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.
- However for GTO we cannot distinguish this well as both positive landing rate and negative landing (unsuccessful mission) are both there here.

In [20]:
```
### TASK  5: Visualize the relationship between Payload and Orbit type
```

Similarly, we can plot the Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type

In [21]:
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the

sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Payload Mass (kg)",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

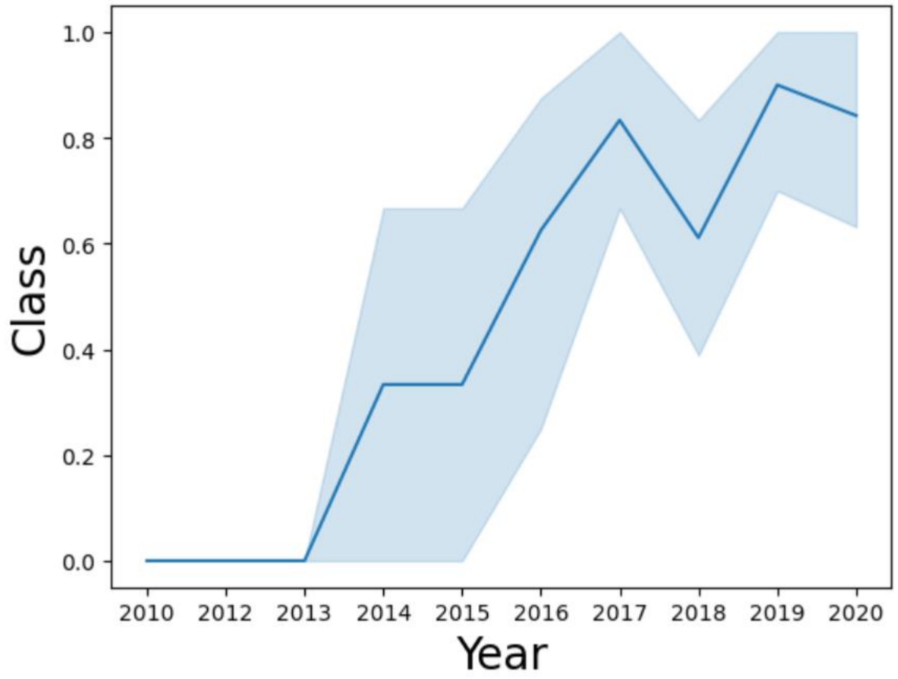
However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend

- You can observe that the success rate since 2013 kept increasing till 2020.

In [24]:

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate

sns.lineplot(y="Class", x="Date", data=df)
plt.xlabel("Year",fontsize=20)
plt.ylabel("Class",fontsize=20)
plt.show()
```



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

All Launch Site Names

- Unique launch sites

Task 1

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

In [8]:
```
%sql select distinct "Launch_Site" from SPACEXTABLE
```

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

Out[8]:

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

Launch Site Names Begin with 'CCA'

- Top 5 records where launch sites begin with `CCA`.

Task 2

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

In [9]:
```
%sql select * from SPACEXTABLE where "Launch_Site" like 'CCA%' limit 5
```

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

Out[9]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Ou
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	S
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	S
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	S
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	S
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	S

Total Payload Mass

- Total payload carried by boosters from NASA.

Task 3

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

In [13]:

```
%sql select SUM("PAYLOAD_MASS__KG_") as 'TotalPayloadMass' from SPACEXTABLE \
where "Customer" = 'NASA (CRS)'
```

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

Out[13]:

TotalPayloadMass
45596

Average Payload Mass by F9 v1.1

- Average payload mass carried by booster version F9 v1.1.

Task 4

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

In [15]:
```
%sql select AVG("PAYLOAD_MASS__KG_") as 'AveragePayloadMass' from SPACEXTABLE \
where "Booster_Version" like 'F9 v1.1%'
```

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

Out[15]:

AveragePayloadMass
2534.6666666666665

First Successful Ground Landing Date

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

Task 5

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

Hint:Use min function

In [16]:

```
%sql select min("Date") as 'FirstSuccesfulLandingOutcome' from SPACEXTABLE where \
"Landing_Outcome" = 'Success (ground pad)'
```

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

Out[16]:

FirstSuccesfulLandingOutcome
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- List of the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000.

Task 6

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

In [17]:

```
%sql select distinct "Booster_Version" from SPACEXTABLE \
where "Landing_Outcome" = 'Success (drone ship)' and \
"PAYLOAD_MASS__KG_" > 4000 and "PAYLOAD_MASS__KG_" < 6000
```

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

Out[17]:

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

Total Number of Successful and Failure Mission Outcomes

- Calculated total number of successful and failure mission outcomes.

Task 7

List the total number of successful and failure mission outcomes

In [19]:

```
%sql select "Mission_Outcome", COUNT(*) as TotalNumberMissionOutcomes from SPACEXTABLE group by \
SUBSTR(TRIM("Mission_Outcome"), 1, 7)
```

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

Out[19]:

Mission_Outcome	TotalNumberMissionOutcomes
Failure (in flight)	1
Success	100

Boosters Carried Maximum Payload

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

Task 8

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

In [21]:
```
%sql select distinct "Booster_Version", "PAYLOAD_MASS__KG_" from SPACEXTABLE \
where "PAYLOAD_MASS__KG_" = (select MAX("PAYLOAD_MASS__KG_") from SPACEXTABLE)
```

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

Out[21]:

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 of the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015.

Task 9

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.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

In [23]:

```
%sql select substr(Date, 6,2) as Month, "Booster_Version", "Launch_Site", "Landing_Outcome" \
from SPACEXTABLE where substr(Date,0,5)='2015' and "Landing_Outcome" = 'Failure (drone ship)'
```

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

Out[23]:

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

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

- Ranked 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.

Task 10

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 [24]:

```
%sql select TRIM("Landing_Outcome") as Landing_Outcome, count(*) as Count from SPACEXTABLE \
where "Date" between '2010-06-04' and '2017-03-20' \
group by TRIM("Landing_Outcome") \
order by count(*) desc
```

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

Out[24]:

Landing_Outcome	Count
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

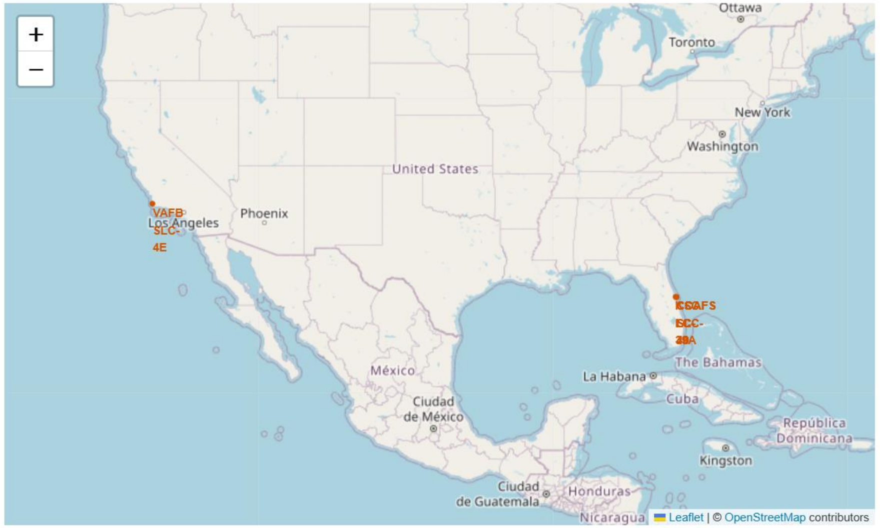
Launch Sites Proximities Analysis

folium_map_all_launch_sites.png

- The current map show markers for all Launch Sites location.
- Looking at the map it is clear how close the Launch Sites are to the coastline and far from cities and main roads.

[54]:
```python
# 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 Lau
for lat, lng, label in zip(launch_sites_df['Lat'].values.tolist(), launch_sites_df['Long'].values.tolist(),
    circle = folium.Circle([lat, lng], radius=1000, color='#d35400', fill=True).add_child(folium.Popup(label
    # Create a blue circle with a icon showing its name
    marker = folium.map.Marker(
        [lat, lng],
        # Create an icon as a text label
        icon=DivIcon(
            icon_size=(20,20),
            icon_anchor=(0,0),
            html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % label,
            )
        )
    site_map.add_child(circle)
    site_map.add_child(marker)

site_map
```

[54]:



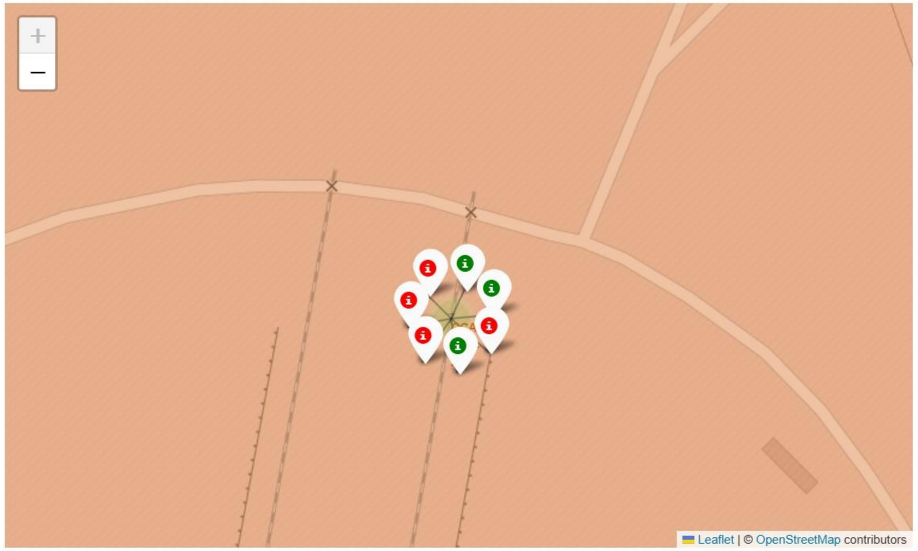
folium_map_reg_green_result.png

- We can see custom markers on the map with green icons indicating successful outcome and red icons indicating failure for a chosen Launch Site.

[59]:

```python
# Add marker_cluster to current site_map
site_map.add_child(marker_cluster)

# for each row in spacex_df data frame
# create a Marker object with its coordinate
# and customize the Marker's icon property to indicate if this launch was successed or failed,
# e.g., icon=folium.Icon(color='white', icon_color=row['marker_color']
for index, record in spacex_df.iterrows():
    # TODO: Create and add a Marker cluster to the site map
    # marker = folium.Marker(...)
    marker = folium.map.Marker(
        [record['Lat'], record['Long']],
        # Create an icon as a text label
        icon=folium.Icon(color='white', icon_color=record['marker_color'])
        )
    marker_cluster.add_child(marker)

site_map
```

[59]:



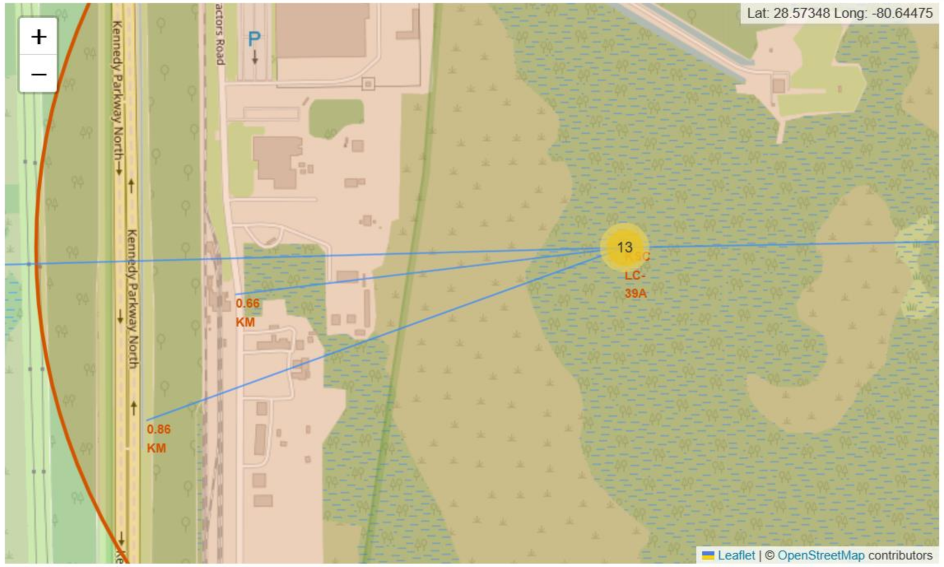
folium_map_distances.png

- The screenshot show the distances of the selected launch site to the closer railway, highway and coastline, with calculated distance.
- The objects folium.PolyLine and folium.Marker were used, in addition to the calculate_distance() function, which calculates the distances involved.

[68]:

```
highway_lat = 28.57062
highway_lon = -80.65521
distance_highway = calculate_distance(launch_site_lat, launch_site_lon, highway_lat, highway_lon)

highway_distance_marker = folium.Marker(
   [highway_lat, highway_lon],
   icon=DivIcon(
      icon_size=(20,20),
      icon_anchor=(0,0),
      html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % "{:10.2f} KM".format(distance_high
      )
   )
marker_cluster.add_child(highway_distance_marker)
highway_lines = folium.PolyLine(locations=[[launch_site_lat, launch_site_lon], [highway_lat, highway_lon]],
marker_cluster.add_child(highway_lines)
site_map
```

[68]:

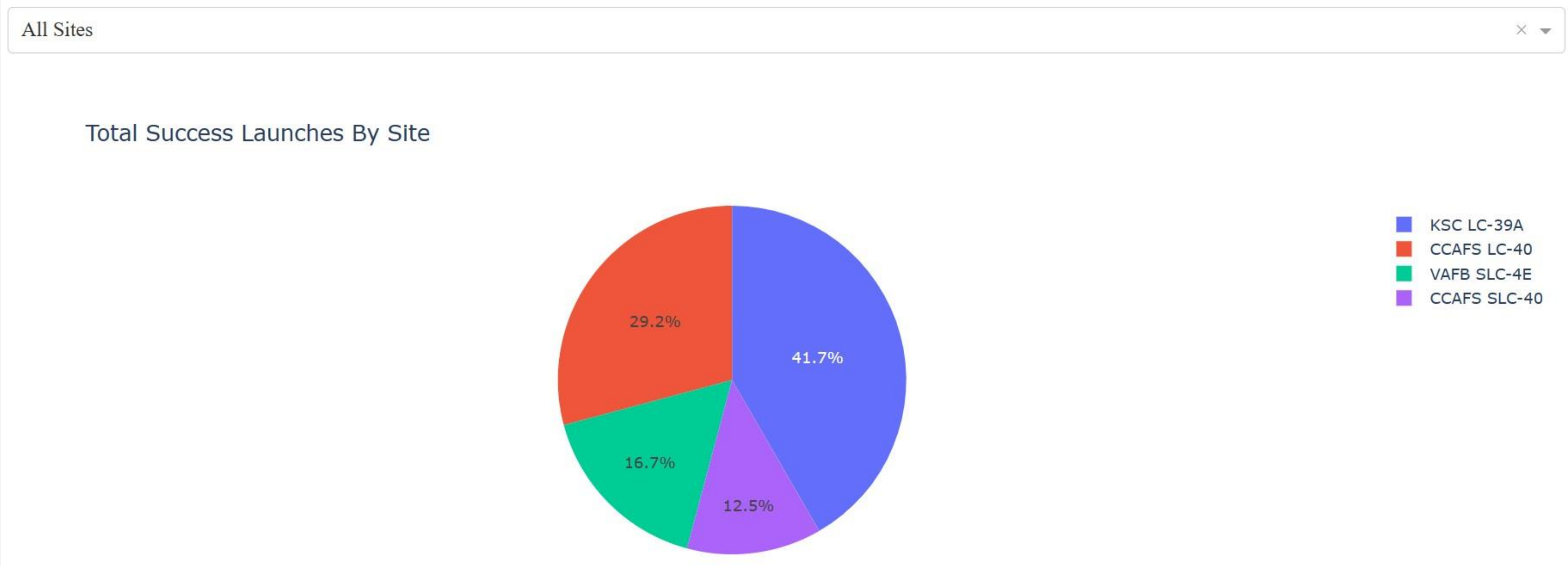


Section 4

Build a Dashboard with Plotly Dash

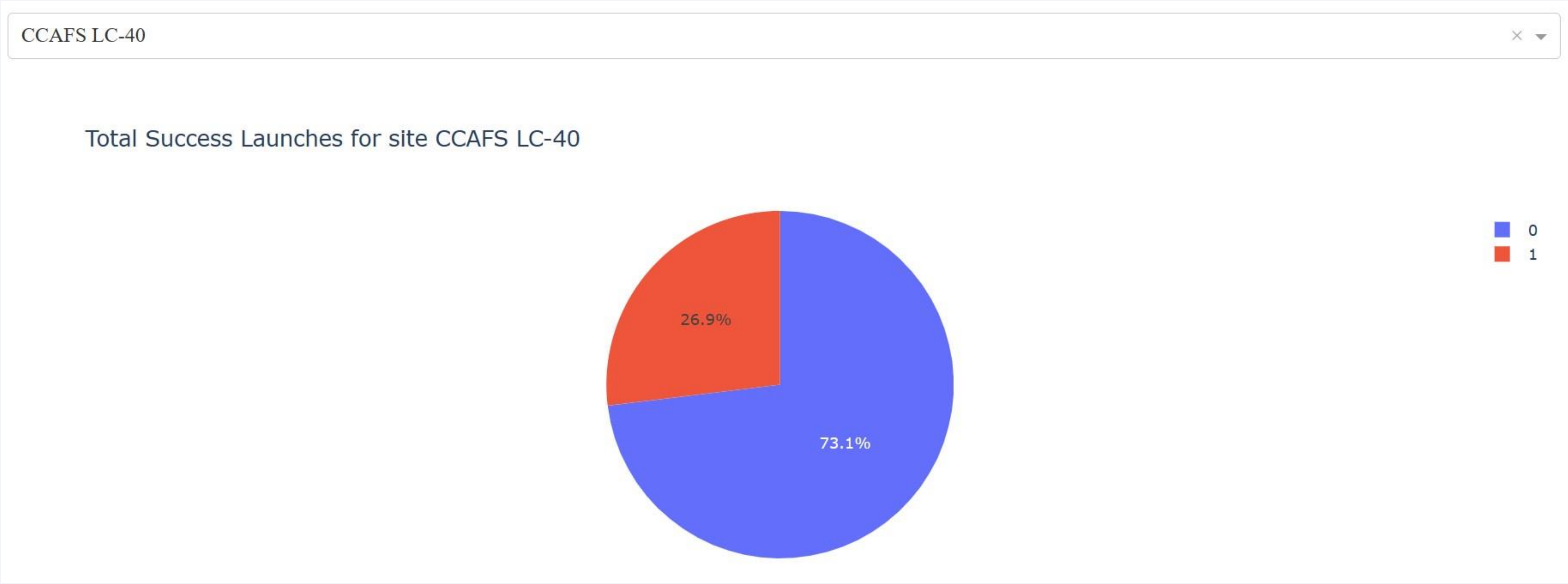
dash10_task2_chart_all.png

- The dashboard has dropdown list to enable Launch Site selection. The default select value is for ALL sites. The KSC LC-39A have the best success ratio in general.



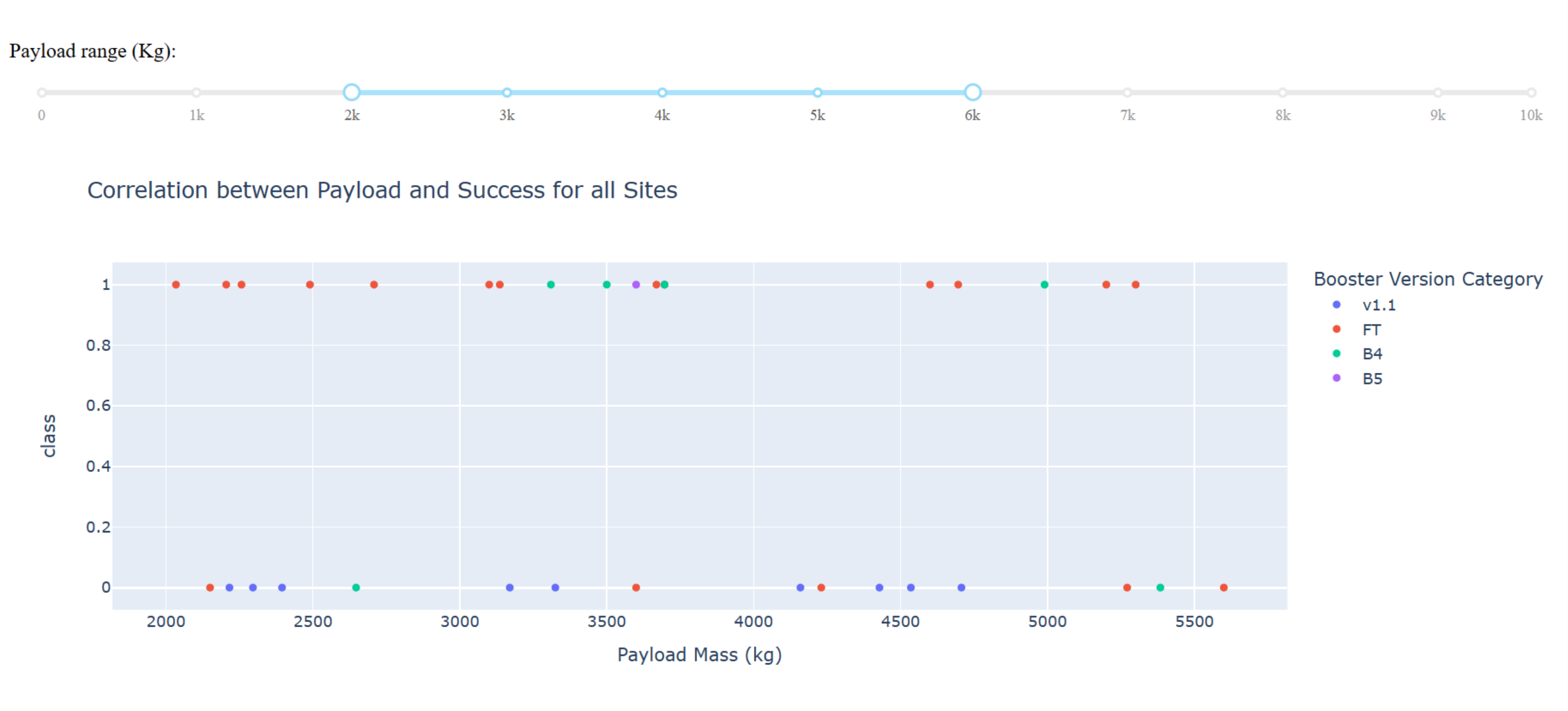
dash10_task2_chart_selected.png

- The CCAFS LC-40 Launch Site have the highest launch success ratio (73.1%).



dash10_task4_range_payload_other.png

- For payload selected range (2k..6k) the Booster Version FT have the largest success rate and the Booster Version v1.1 have the lower success rate.



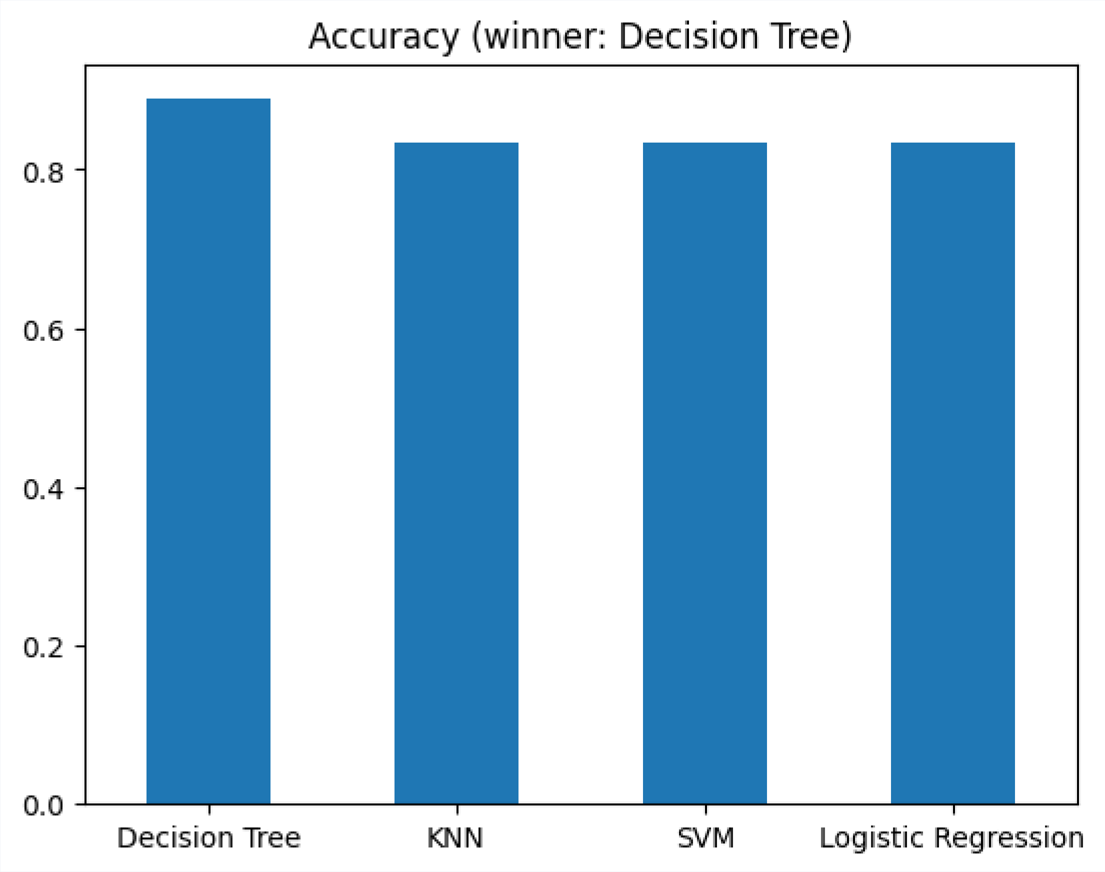
Section 5

Predictive Analysis (Classification)

Classification Accuracy

The “Decision Tree” model presented greater accuracy in relation to the tie that occurred with the other models (88.8% vs 83.3%).

Model	Accuracy
Decision Tree	0.8888888888888884
KNN	0.8333333333333337
SVM	0.8333333333333337
Logistic Regression	0.8333333333333337

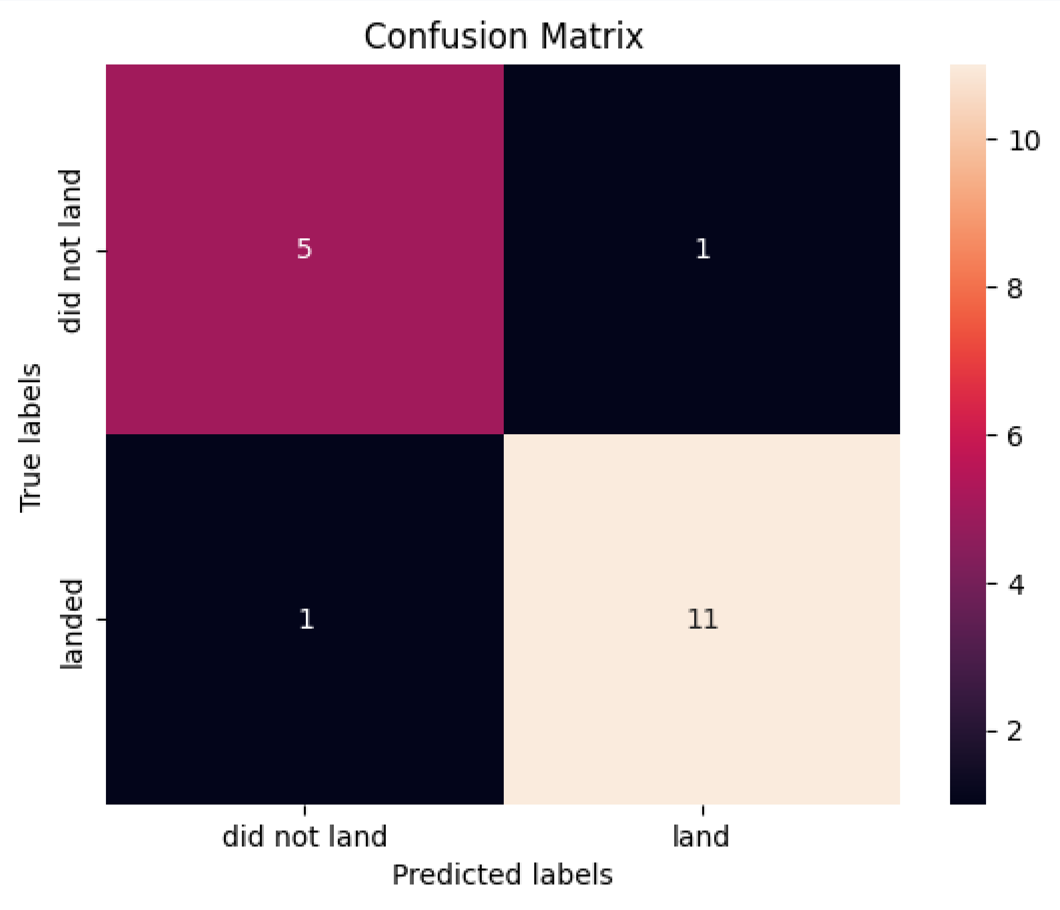


Confusion Matrix

Examining the confusion matrix, we see that “Decision Tree” model can distinguish better between the different classes.

This model performed a little better than the others in the “true negative” rate*.

* 5/18 negative matches from Decision Tree vs 3/18 negative matches from other models.



Conclusions

A good predictive model of the success rate of reusing the first stage of rockets can benefit several areas of business, such as:

- Rocket insurance companies through better assessment of the risks involved in each type of launch.
- Rocket factories can learn from the problems identified for each type of launch and modify them in order to solve each one of them. Thus making launches safer and more reliable, as well as more economical over time.
- Researchers can use the current work as a basis to create predictive models that are more accurate than the current one.

Future work:

- During the training of the different models chosen for this work, a certain randomness was observed in the accuracy results with test data for the "Decision Tree" model, which sometimes tied with the others with 83% and sometimes won with 88%. More data to train and evaluate the models and more adjustments to model parameters can improve this in the future.

Appendix

- Project GitHub repository with all python codes, python notebooks, screenshots and datasets created during this project :

https://github.com/egonrp/certificacaodsfase10

The PDF file of the current presentation can be accessed in the repository via the direct link:

https://github.com/egonrp/certificacaodsfase10/blob/main/ds-capstone-template-coursera.pdf

Thank you!