IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Reza Karimpour
Nov 21, 2023

Outline

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

Executive Summary

Summary of methodologies

- Data collection from SpaceX API and web scraping from a Wikipedia page
- Data wrangling and exploratory data analysis using SQL and data visualization for understanding the data correlation and preparing feature engineering
- Using Folium and Plotly Dash application to provide stakeholders with interactive visual analytics on SpaceX launch data in real-time
- Finding best Hyperparameters for SVM, Classification Trees, Logistic Regression, and KNN (k nearest neighbors) algorithms to perform Machine learning predictions

Summary of all results

The performed exploratory data analyses showed significant correlation between the results of first-stage landing attempts and several launch parameters including booster version, payload mass, type of aimed orbit, launch site, etc. Therefore those parameters were used to prepare machine learning (ML) models to predict if the first stage will land successfully. The trained ML models were able to predict the result with high accuracy (>80%).

Introduction

Project background and context

According to SpaceX, lunching a Falcon 9 rocket costs 62 million dollars, a price that is considerably lower than other providers' costs (upward of 165 million dollars). Such a considerable cost difference is due to the attempts of SpaceX for reusing the first stage of the launched rockets. However, SpaceX attempts do not always result is successful first-stage landings which consequently cause a jump in the cost of the preformed launch. Therefore if we can determine whether the first stage will land successfully we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

The Problem

A rocket launch has so many specific parameters that can potentially affect the result of an attempt for landing the first-stage rocket. These parameters differ from one launch to another and include booster version, payload mass, type of aimed orbit, launch site, and so many others. However, to decide which ones are more important and can more significantly affect the final result one has to carefully analyze the launch data to understand the correlation between various parameters and the landing result. Therefore, a thorough analysis of the data is required to determine what parameters can considered as the independent parameters of a machine learning model for predicting the landing results without unnecessarily increasing the dimensions of the problem.

Section 1

Methodology

Methodology

Executive Summary

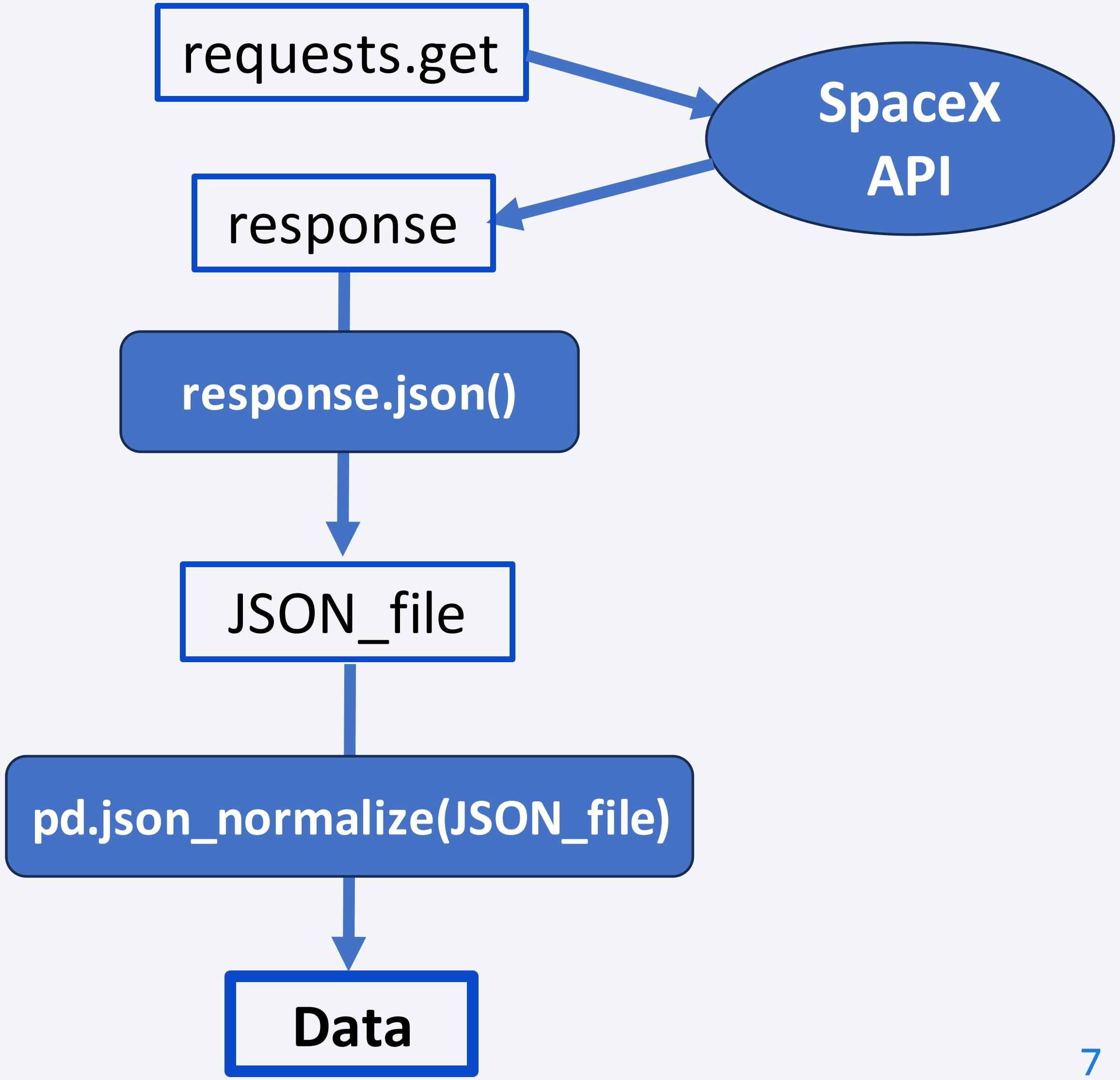
- Data collection methodology:
  - The data was collected from the SpaceX API in the form of a JSON file by using a 'get request'. The requested JSON was then stored as a DataFrame.
- Perform data wrangling
  - The collected data was filtered to keep only the Falcon 9 data. It was then further processed to deal with missing data (of the payload mass of the rockets) by replacing the missing ones by the mean value. Categorical attributes were also converted to numerical data by applying 'get_dummies'.
- 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 – SpaceX API

- The data was requested from the SpaceX API by using a 'get request' call. The response from the API was then decoded in the form of a JSON file and finally stored in a Pandas data frame using .json_normalize().

- GitHub URL of the completed SpaceX API calls notebook:

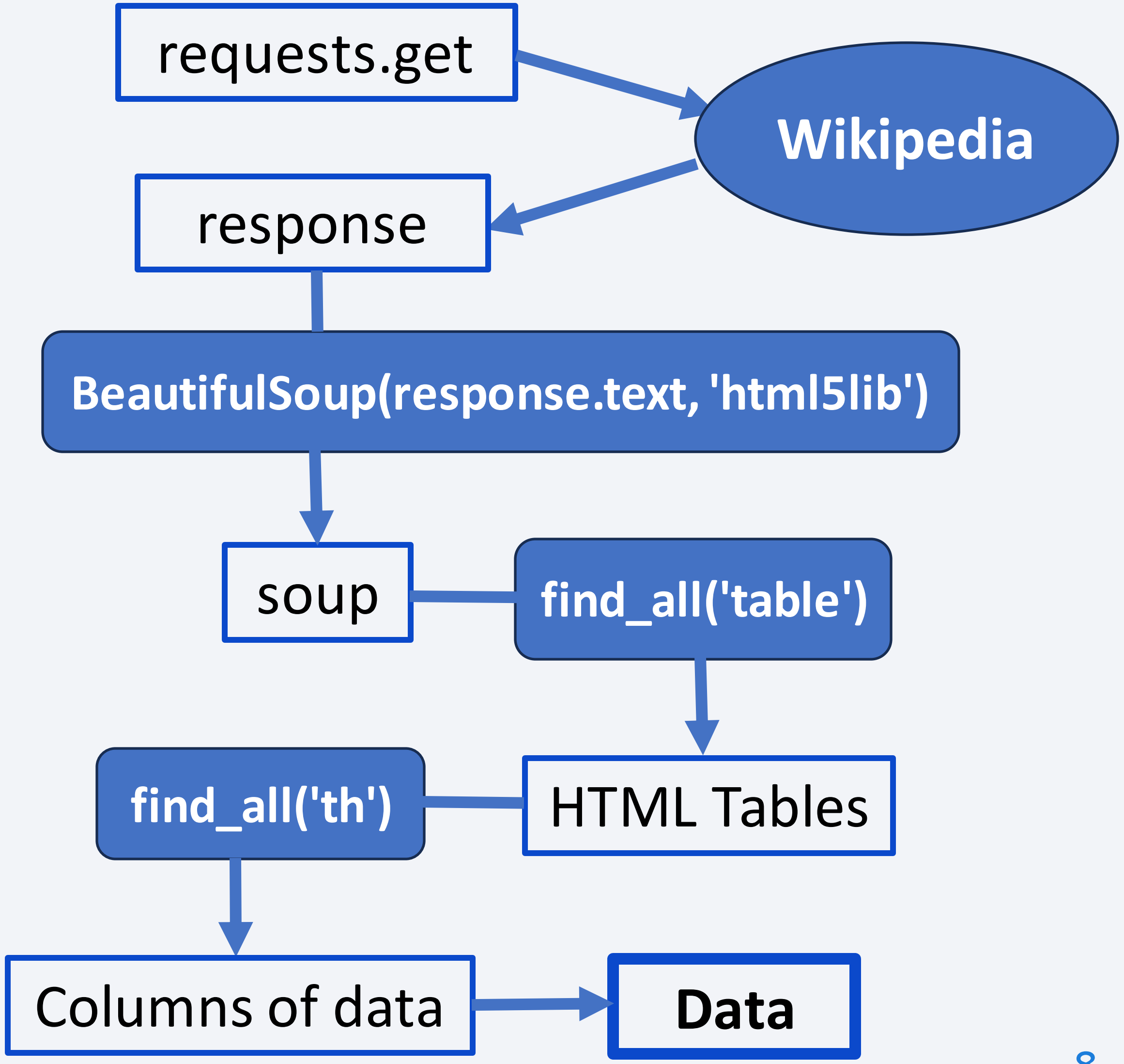
https://github.com/rezakp0/applied-data-science-capstone/blob/main/W1_01_spacex-data-collection-api.ipynb



Data Collection - Scraping

- Falcon 9 historical launch records were also collected from a [Wikipedia page](#) titled "List of Falcon 9 and Falcon Heavy launches" using a 'get requests' call. Then a BeautifulSoup object is created from the html response we got from the call in the first step. Using the find_all('table) we then find all the tables in the soup object. The third table is our target table from which we extract columns of data using find_all('th'). Then we store the columns in a dictionary and finally store it as a Pandas dataframe.
- GitHub URL of the completed web scraping notebook:

https://github.com/rezakp0/applied-data-science-capstone/blob/main/W1_02_webscraping.ipynb



Data Wrangling

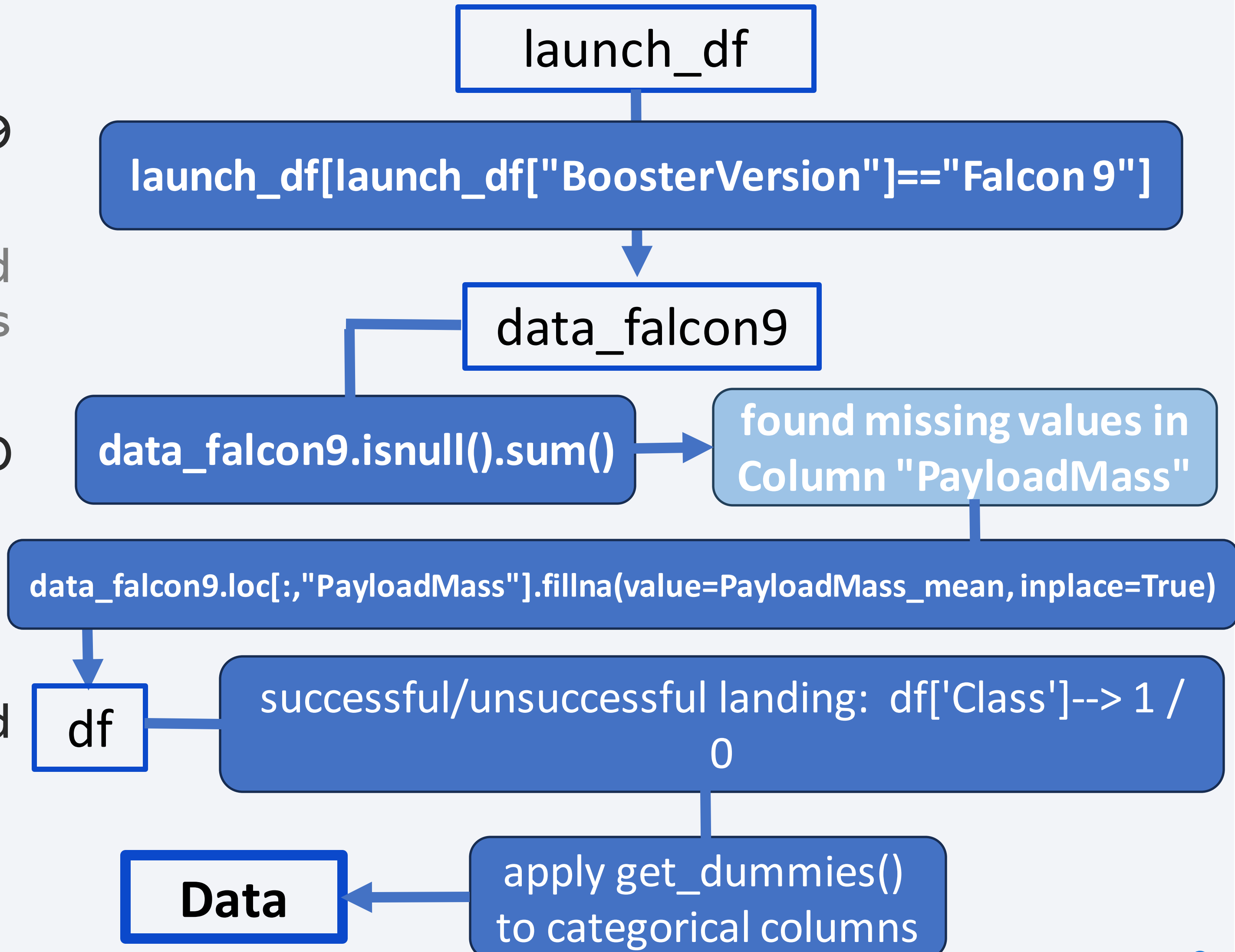
- Convert the raw data into a usable form:

1) Filter the data to only include Falcon 9 records
2) Deal with the missing values: find and replace missing values with mean values in each column
3) Create column "Class" with values 1/0 for successful/unsuccessful landings
4) Convert categorical data into numerical values: use get_dummies()

- GitHub URLs of data wrangling related notebooks:

[W1_03_Data_wrangling.ipynb](), and

[W4_SpaceX_Machine_Learning_Prediction.ipynb]()



EDA with Data Visualization

- Charts that were plotted for EDA purpose:

1) Scatter plots of (**FlightNumber** vs. **PayloadMass**) & (**FlightNumber** vs. **Orbit type**) & (**PayloadMass** vs. **Launch Site**) & (**PayloadMass** vs. **Orbit type**), each with color labels for the class of the landing outcome: to see the correlation between the considered variables and the outcome.

2) Bar chart of Success rate vs Orbit types: to compare success rates for launches to different orbits

3) Line chart of launch success vs year: to see the yearly trend of success rate

- GitHub URL of EDA with data visualization notebook:

https://github.com/rezakp0/applied-data-science-capstone/blob/main/W2_02_eda_dataviz.ipynb

EDA with SQL

- Names of unique launch sites: %sql select distinct "Launch_Site" from SPACEXTABLE;
- 5 records where launch sites begin with 'CCA': %sql select * from SPACEXTABLE where "Launch_Site" like 'CCA%' limit 5;
- Total payload mass carried by boosters launched by NASA (CRS): %sql select sum("PAYLOAD_MASS__KG_") as "Total Payload Mass [NASA (CRS)]" from SPACEXTABLE where "Customer"=="NASA (CRS)"
- Average payload mass carried by booster version F9 v1.1: %sql select avg("PAYLOAD_MASS__KG_") as "AVG Payload Mass Carried by F9 v1.1" from SPACEXTABLE where "Booster_Version"=="F9 v1.1"
- Date of first successful landing on the ground pad: %%sql select min("Date") as "Date of First Succesful Landing on Ground Pad" from SPACEXTABLE where "Landing_Outcome"=="Success (ground pad)"

(Continued in the next slide)

- GitHub URL of completed EDA with SQL notebook:

https://github.com/rezakp0/applied-data-science-capstone/blob/main/W2_01_eda-sql_sqllite.ipynb

EDA with SQL

- Boosters with payload mass between 4000 and 6000, successfully landed on drone ships: %%sql select "Booster_Version" as "Boosters with payload mass between 4000 and 6000, successfully landed on drone ships" from SPACEXTABLE where "Landing_Outcome"=="Success (drone ship)" and "PAYLOAD_MASS__KG_" between 4000 and 6000
- List total number of successful and failure mission outcomes: %%sql select "Mission_Outcome" as "Mission Outcome", count("Mission_Outcome") as "Count" from SPACEXTABLE group by "Mission_Outcome"
- Boosters that have carried maximum payload mass: %%sql select "Booster_Version" as "Boosters which have carried maximum payload mass" from SPACEXTABLE where "PAYLOAD_MASS__KG_"==(select max("PAYLOAD_MASS__KG_") from SPACEXTABLE)
- List records displaying the month names, failure landing_outcomes in drone ship, booster versions, and launch_site for the months in year 2015: %%sql select substr("Date",6,2) as "Month in 2015", "Booster_Version", "Launch_Site", "Landing_Outcome" from SPACEXTABLE where "Landing_Outcome"=="Failure (drone ship)" and substr("Date", 1, 4)=="2015" limit 5
- Rank (desc.) the count of landing outcomes between the dates 2010-06-04 and 2017-03-20: %%sql select "Landing_Outcome" as "Landing Outcome", count(*) as "Count" from SPACEXTABLE where "Date" between "2010-06-04" and "2017-03-20" group by "Landing_Outcome" order by "Count" desc

https://github.com/rezakp0/applied-data-science-capstone/blob/main/W2_01_eda-sql_sqllite.ipynb

Build an Interactive Map with Folium

- Mark each launch site on a map with a circle and the name of the site as a marker, to visually see the locations of the sites on the map.
- Add a green/red marker for each individual successful/failed landing outcome at each site: This shows which site has more successful/failed landing outcomes (first use MarkerCluster to add several markers to the same points on the map).
- Mark down a point on the closest coastline/railway to the site CCAFS SLC-40 and draw a PolyLine connecting the point to the site: such marks and lines show how the site are located in the proximity of railways, coasts, highways but away from cities (first add a MousePosition to the map so one could see the coordinates on the map by hovering the mouse over the map).
- GitHub URL of interactive map with Folium:

https://github.com/rezakp0/applied-data-science-capstone/blob/main/W3_01_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- **Add a launch site drop-down input component:** to let a user select the launch site for which the information are presented on the dashboard
- **Add a callback function to render success-pie-chart based on selected launch:** the Pie chart shows the success/fail rate of landing outcomes in the selected site
- **Add a range slider to select payload mass range:** to let a user select a range for payload mass of the launched rockets
- **Add a callback function to render a success_payload scatter chart:** this scatter plot shows the success rate of launches for the selected range of payload mass and for different boosters in the selected site
- GitHub URL of Plotly Dash lab:

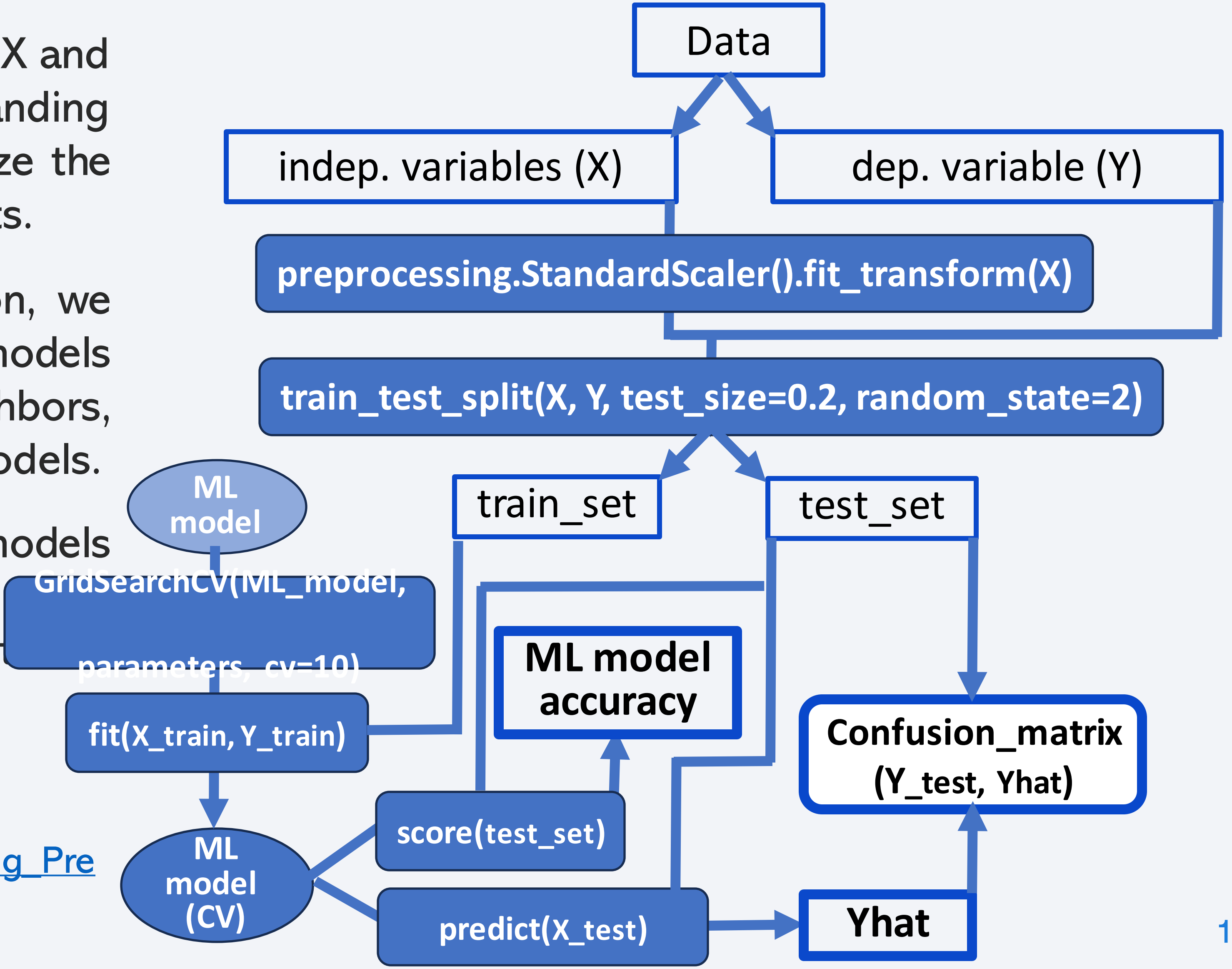
https://github.com/rezakp0/applied-data-science-capstone/blob/main/W3_02_Spacex_Dashboard.ipynb (JupyterLab)

https://github.com/rezakp0/applied-data-science-capstone/blob/main/Spacex_Dashboard.py (python code)

Predictive Analysis (Classification)

- After determining independent variables X and dependent variable Y(*i.e.* class of landing outcomes) of the problem, we standardize the data and split it into training and test sets.
- Then, using Grid Search Cross-Validation, we find best Hyperparameter for the models SVM, Classification Trees, k nearest neighbors, and Logistic Regression, and train the models.
- Then we calculate the accuracy of the models on the test data, get confusion matrix for each and plot it
- GitHub URL of predictive analysis lab:

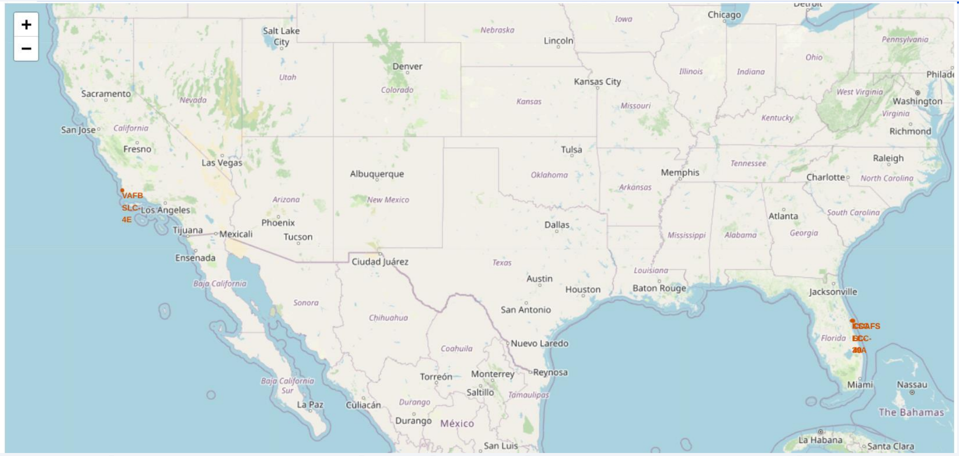
https://github.com/rezakp0/applied-data-science-capstone/blob/main/W4_SpaceX_Machine_Learning_Prediction.ipynb



Results: Exploratory data analysis

- **Success rate vs launch site:** different launch sites have different success rates.
- **Success rate vs flight number at different sites:** successful landings (Class 1) in all launch sites become more frequent as the flight number increases.
- **Payloads at launch sites:** payload of lunches have different ranges at different sites (e.g. at VAFB-SLC there are no rockets launched with heavy payload mass).
- **Success rate vs orbit type:** for ES-L1, GEO, HEO, and SSO is the highest, i.e. 1, whereas for the orbit SO is the lowest, i.e. 0.
- **Success rate vs flight number for different orbits:** for LEO orbit the success rate seems to be related to the number of flights, but no correlation is seen when the aimed orbit is GTO.
- **Yearly trend of success rate:** success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing again.

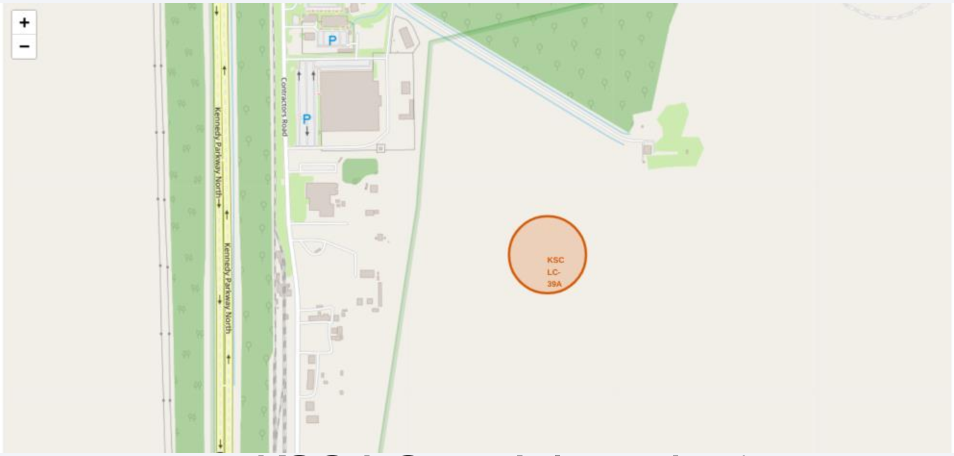
Results: Interactive analytics demo – Folium Maps



launch sites marked with circles and names

CCAFS LC-40 and CCAFS SLC-40 launch sites

VAFB SLC-4E launch site



KSC LC-39A launch site

https://github.com/rezakp0/applied-data-science-capstone/blob/main/W3_01_launch_site_location.ipynb

Results: Interactive analytics demo – Folium Maps

successful/failed landings at CCAFS LC-40

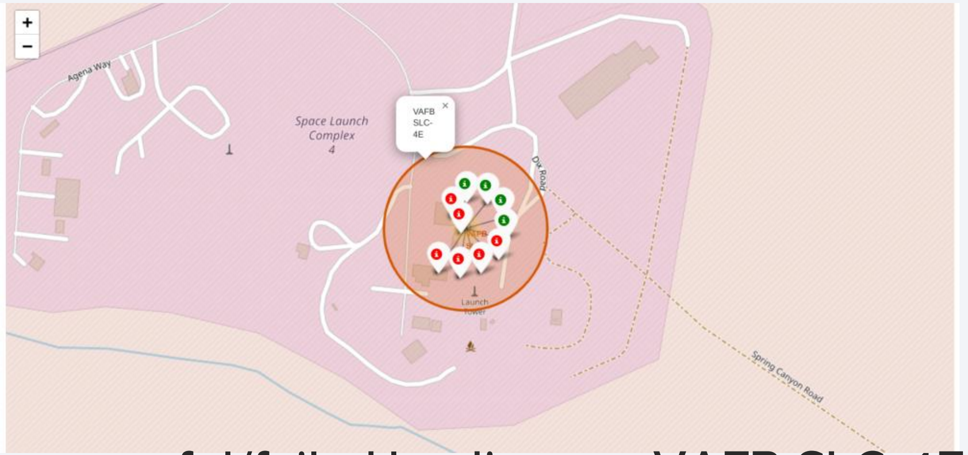
successful/failed landings at CCAFS SLC-40

successful

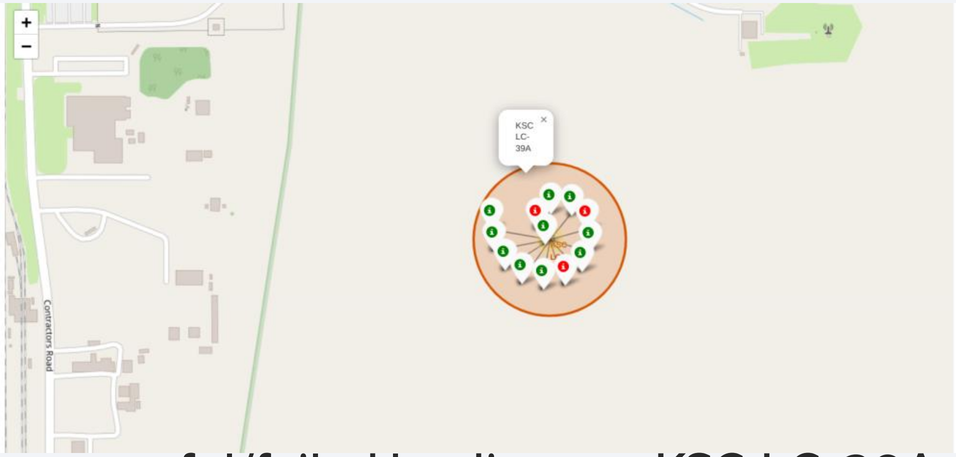
green

failed

red



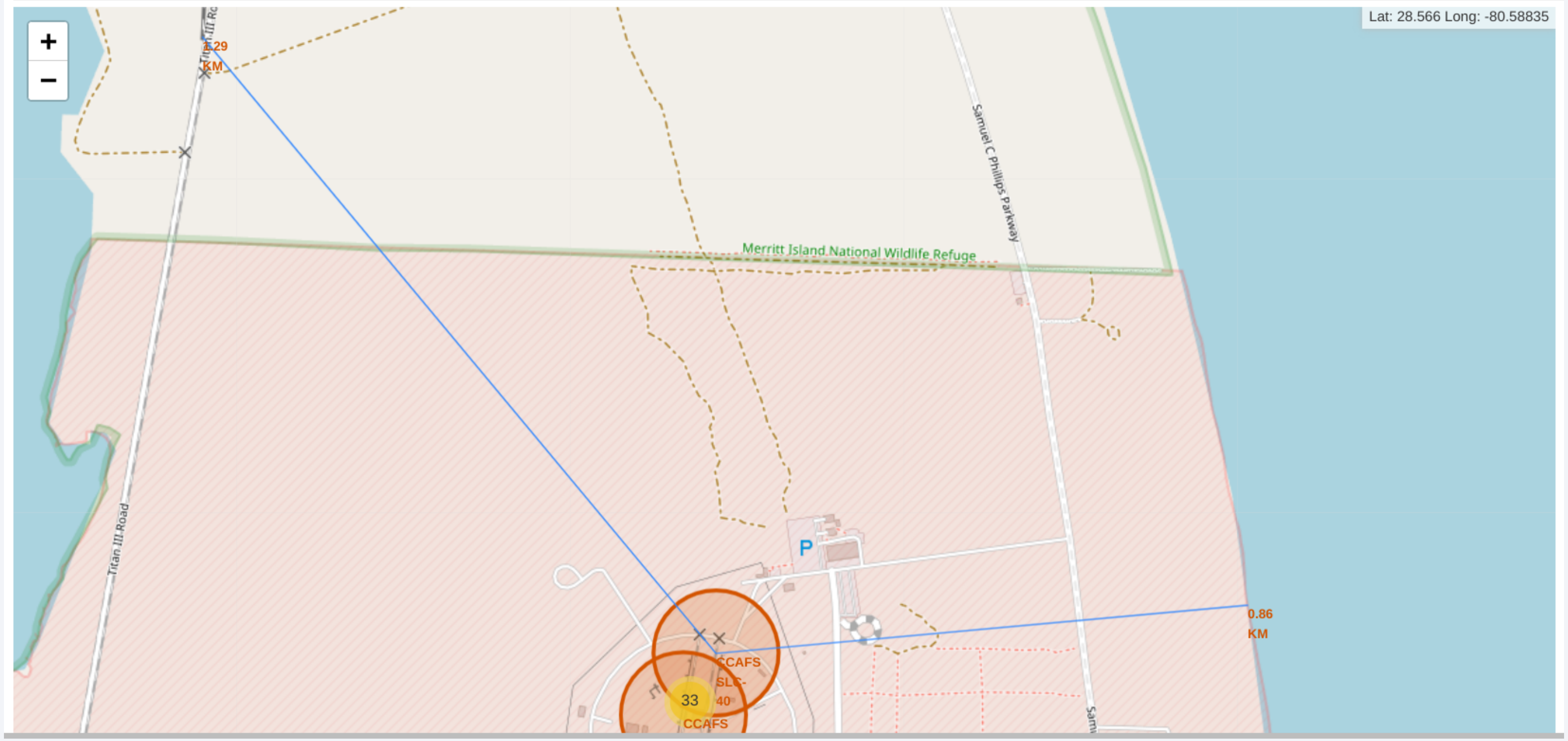
successful/failed landings at VAFB SLC-4E



successful/failed landings at KSC LC-39A

https://github.com/rezakp0/applied-data-science-capstone/blob/main/W3_01_launch_site_location.ipynb

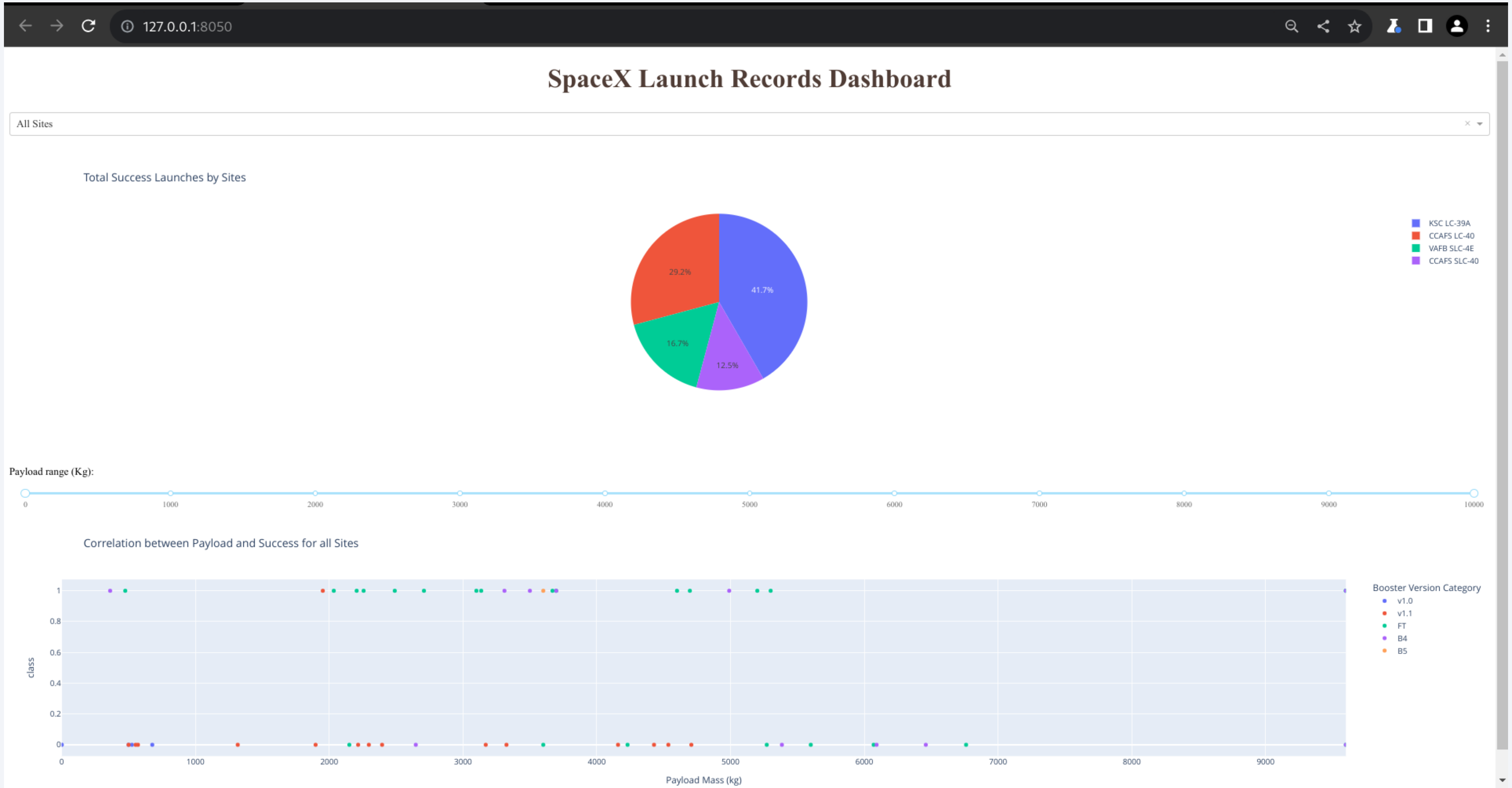
Results: Interactive analytics demo – Folium Maps



Distances between the closest coastline and railway to CCAFS SLC-40 launch site (with polylines in blue)

https://github.com/rezakp0/applied-data-science-capstone/blob/main/W3_01_launch_site_location.ipynb

Results: Interactive analytics demo – Plotly Dash



https://github.com/rezakp0/applied-data-science-capstone/blob/main/W3_02_Spacex_Dashboard.ipynb

https://github.com/rezakp0/applied-data-science-capstone/blob/main/Spacex_Dashboard.py

Results: Predictive analysis

- Among the four ML models **Logistic Regression**, **SVM**, **Decision Tree Classifier**, and **KNN,** the Classification Trees model performs slightly better as shown in the following accuracy table.

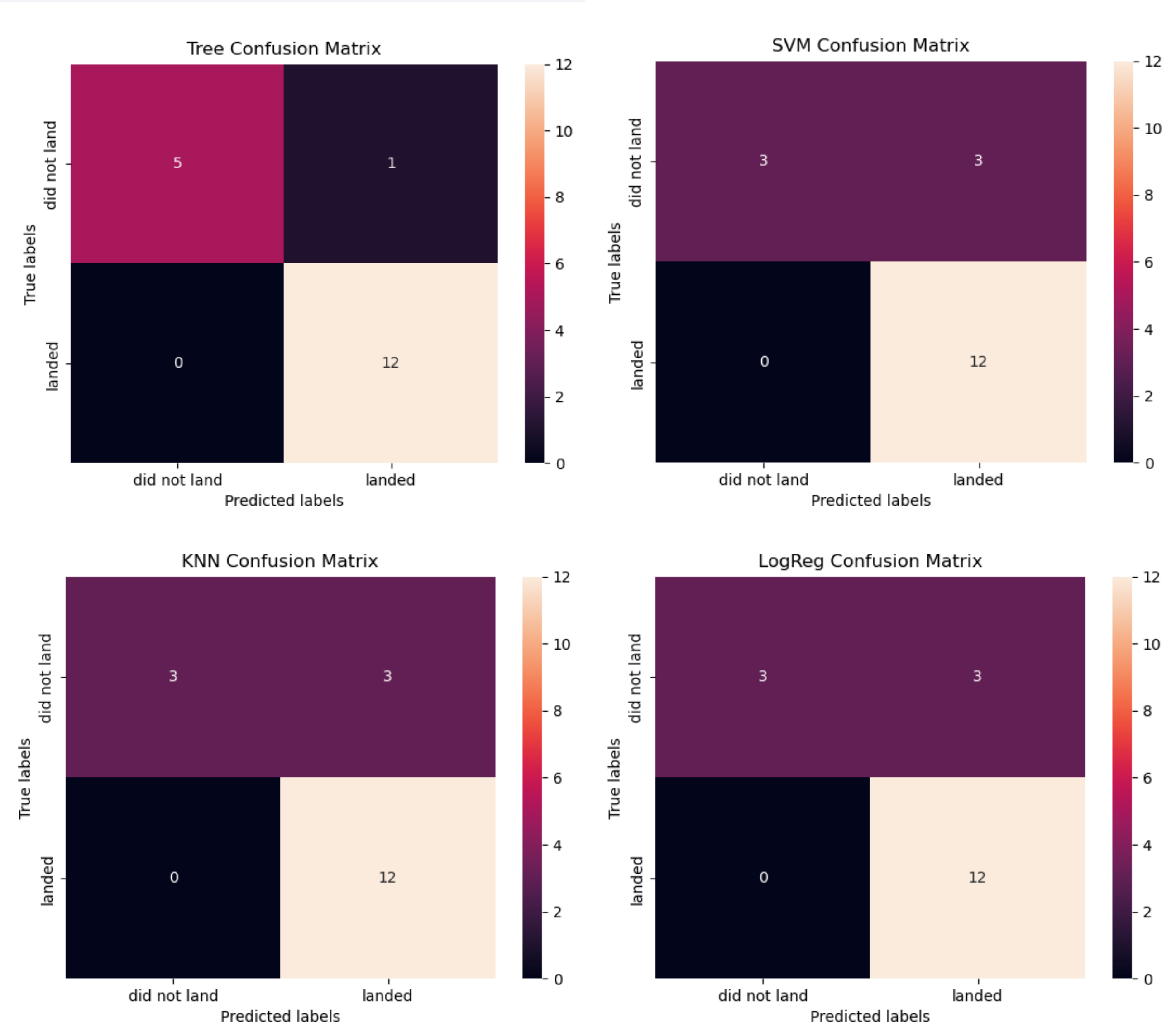
- The Decision Tree Classifier has *higher accuracy scores* (from .score, .jaccard_score, f1_score, and r2_score) and *smaller errors* (mean_absolute_error and mean_squared_error).

Model	Accuracy Score	Jaccard Index	F1 score	MAE	MSE	R2 Score
Logistic Regression	0.833333	0.800000	0.888889	0.166667	0.166667	0.25
SVM	0.833333	0.800000	0.888889	0.166667	0.166667	0.25
Decision Tree Classifier	0.944444	0.923077	0.960000	0.055556	0.055556	0.75
KNN	0.833333	0.800000	0.888889	0.166667	0.166667	0.25

https://github.com/rezakp0/applied-data-science-capstone/blob/main/W4_SpaceX_Machine_Learning_Prediction.ipynb

Results: Predictive analysis

- All the models can distinguish between the different classes of the landing outcomes.
- Decision Tree Classifier predicts less false positives compared to the other models (1 vs 3).
- All the models perfectly did not result any false negative.



https://github.com/rezakp0/applied-data-science-capstone/blob/main/W4_SpaceX_Machine_Learning_Prediction.ipynb

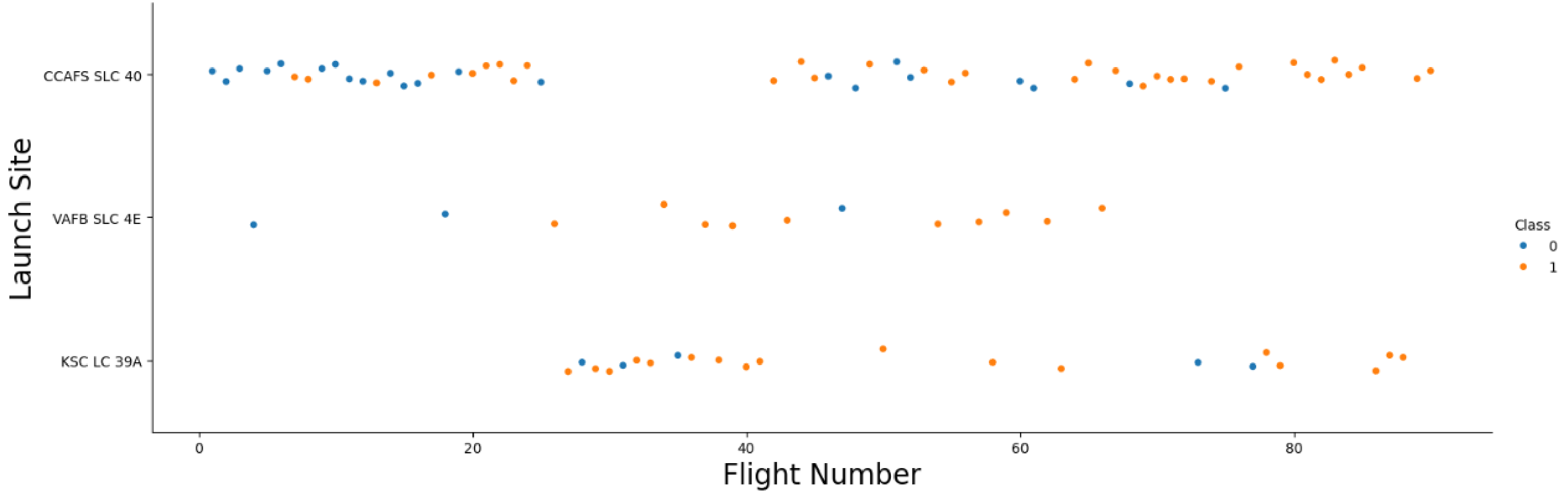
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

[40]:

```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hu
sns.catplot(y="LaunchSite", x="FlightNumber", hue='Class', data=df, aspect=3)
plt.xlabel("Flight Number", fontsize=20)
plt.ylabel("Launch Site", fontsize=20)
plt.show()
```

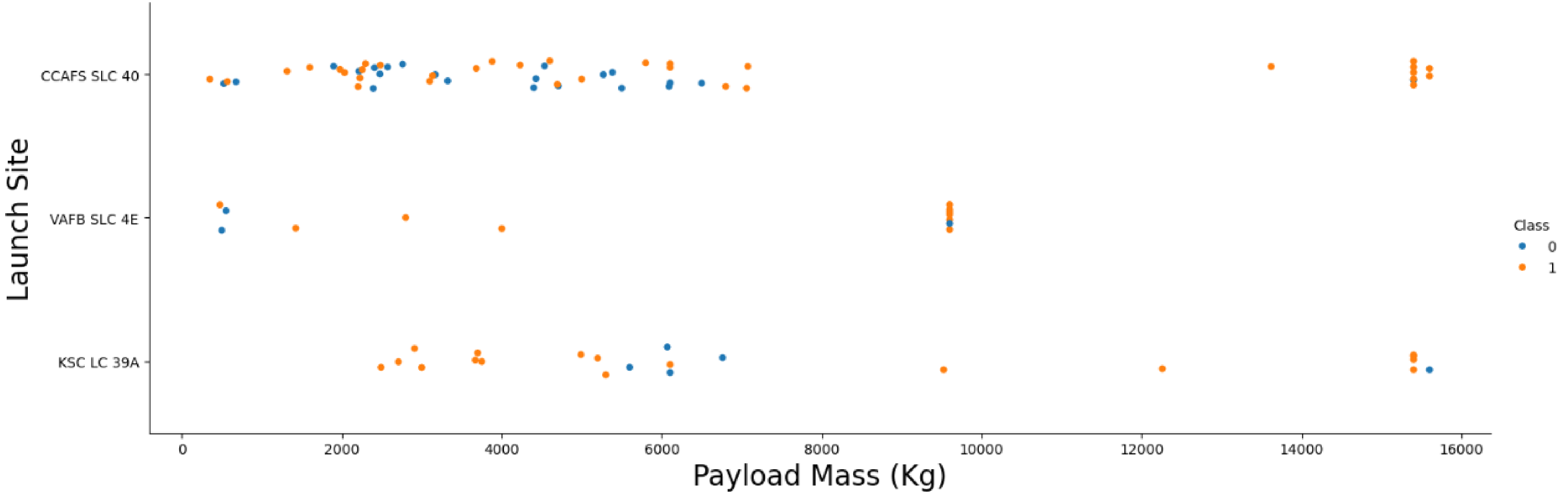


Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

The patterns found in the Flight Number vs. Launch Site scatter point plots show that successful landings (Class 1) in all launch sites become more frequent as the flight number increases.

Payload vs. Launch Site

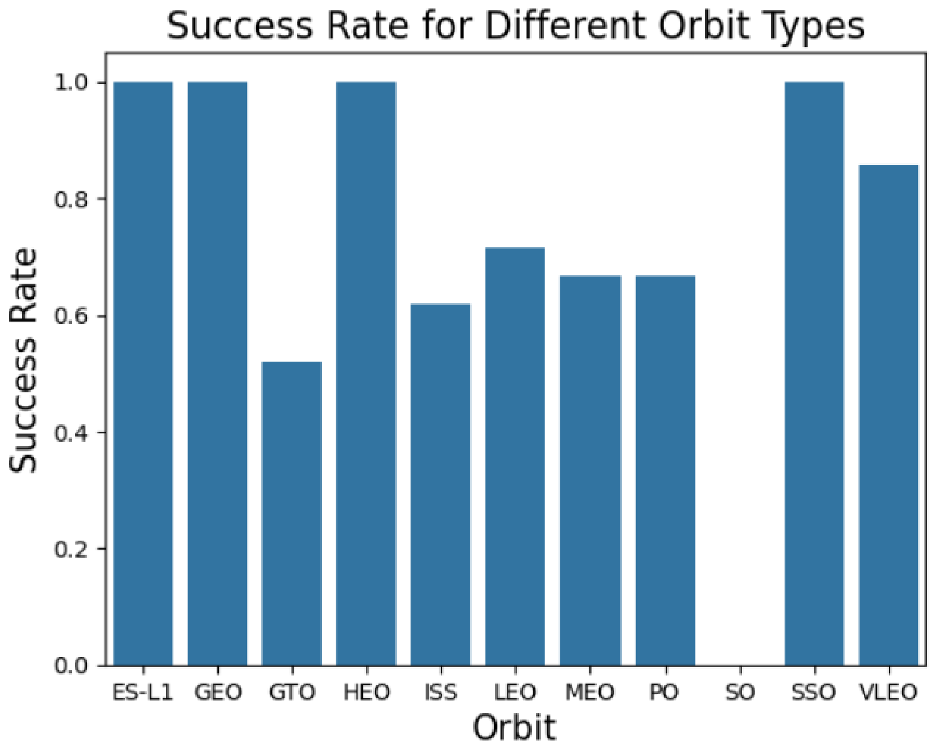
```
[53]: # Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, a
sns.catplot(y="LaunchSite", x="PayloadMass", hue='Class', data=df, aspect=3)
plt.xlabel("Payload Mass (Kg)", fontsize=20)
plt.ylabel("Launch Site", fontsize=20)
plt.show()
```



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

```
[69]: # HINT use groupby method on Orbit column and get the mean of Class column
sns.barplot(df.groupby("Orbit").mean("Class"), x='Orbit', y='Class')
plt.xlabel("Orbit", fontsize=15)
plt.ylabel("Success Rate", fontsize=15)
plt.title("Success Rate for Different Orbit Types", fontsize=16)
plt.show()
```

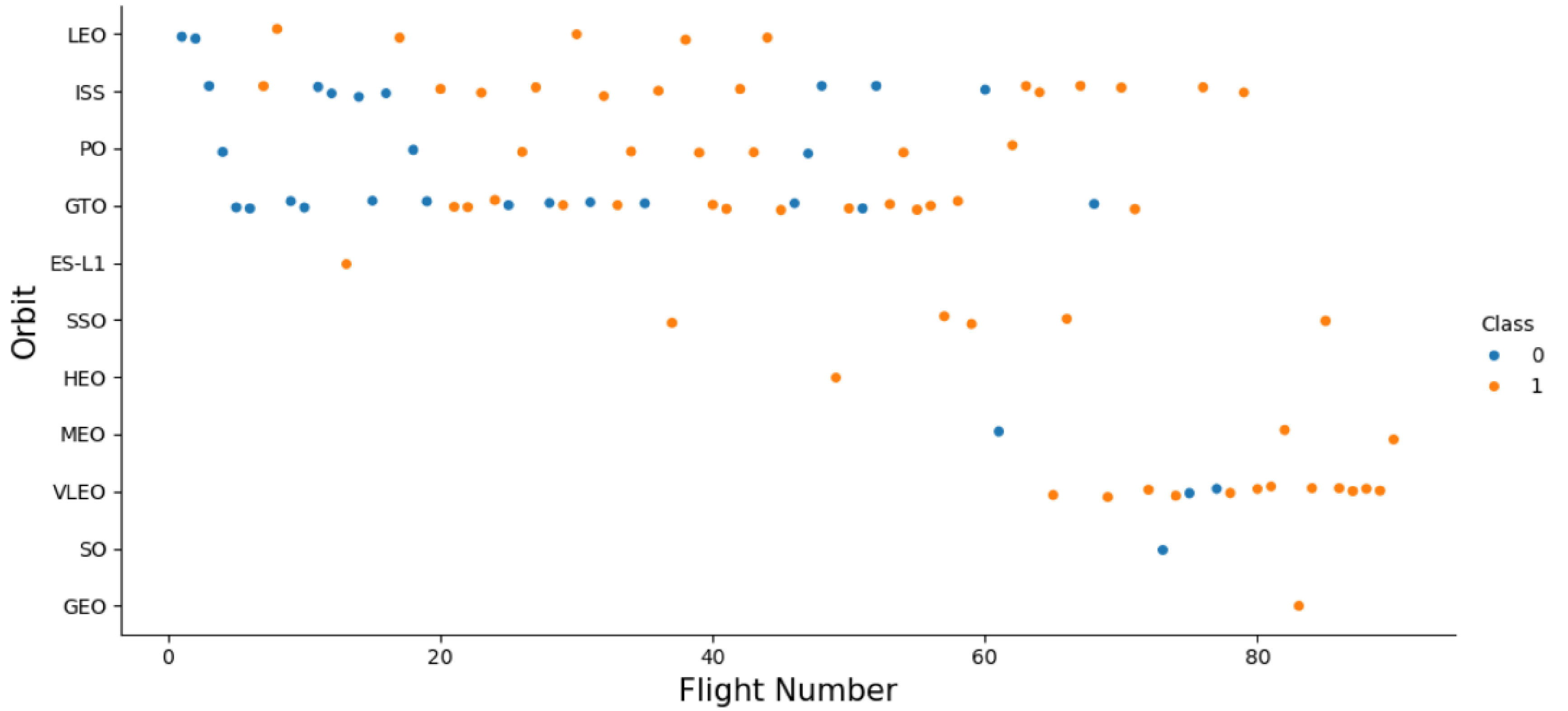


Analyze the ploted bar chart try to find which orbits have high sucess rate.

Success rate for launches to orbits ES-L1, GEO, HEO, and SSO is the highest (=1) whereas the success rate for the orbit SO is the lowest (=0)

Flight Number vs. Orbit Type

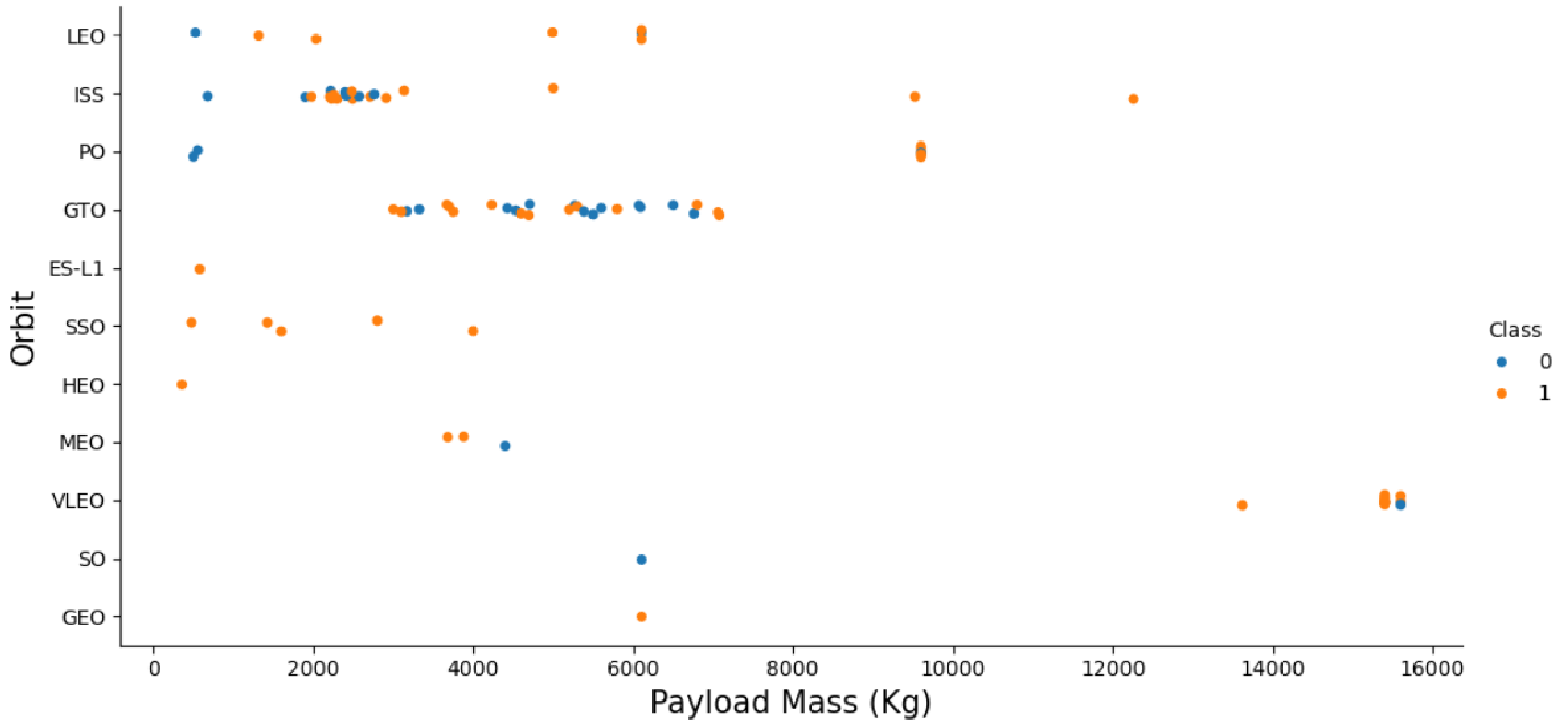
```
[79]: # Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit,
# and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect=2)
plt.xlabel("Flight Number", fontsize=15)
plt.ylabel("Orbit", fontsize=15)
plt.show()
```



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

```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect=2)
plt.xlabel("Payload Mass (Kg)", fontsize=15)
plt.ylabel("Orbit", fontsize=15)
plt.show()
```



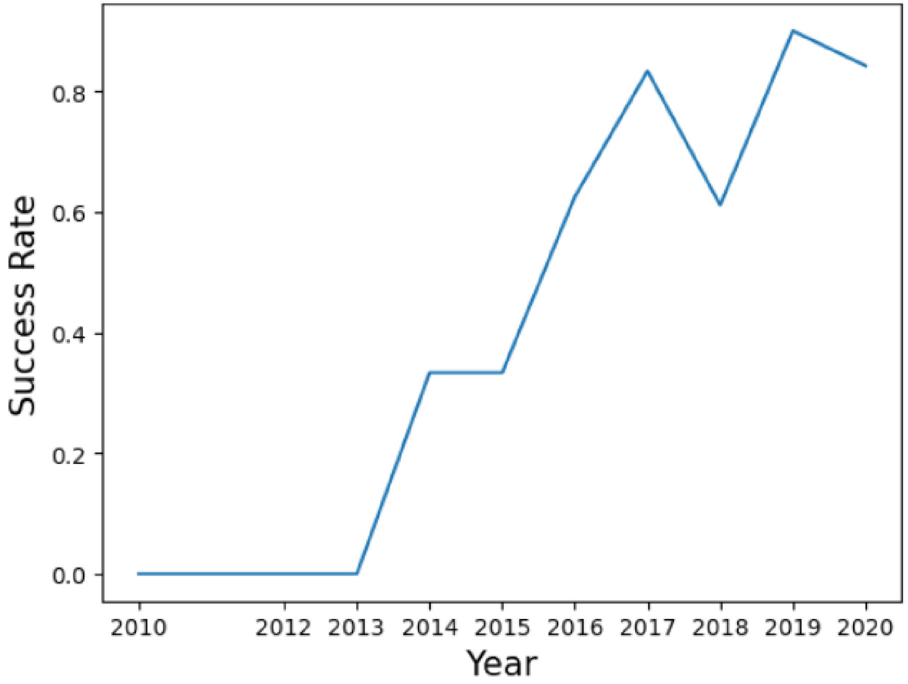
With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

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

```python
# Alternatively, I use pd.to_datetime(df['Date']).dt.year to get the year for each row of the dataframe, then group the
# dataframe by the year, and finally get the average success rate from that as follows:
df['Year'] = pd.to_datetime(df['Date']).dt.year
```

```python
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
sns.lineplot(df.groupby("Year").mean("Class"), x="Year", y="Class")
plt.xlabel("Year", fontsize=15)
plt.ylabel("Success Rate", fontsize=15)
plt.xticks(df['Year'].unique())
plt.show()
```



You can observe that the success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.

All Launch Site Names

To find the names of the unique launch sites one can use a SQL query as shown below:

```
[9]: %sql select distinct "Launch_Site" from SPACEXTABLE;
 * sqlite:///my_data1.db
Done.
```

[9]:

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

or alternatively use the method unique() with the series `spacex_df['Launch Site']` as follows:

```
[13]: print("Names of the unique Launch Sites: ",spacex_df['Launch Site'].unique())
Names of the unique Launch Sites:  ['CCAFS LC-40' 'VAFB SLC-4E' 'KSC LC-39A' 'CCAFS SLC-40']
```

Launch Site Names Begin with 'CCA'

To find 5 records where launch sites begin with `CCA`, we use a SQL query as follows; the where – like clause limits the answer to the rows with launch site starting with `CCA` and the 'limit 5' clause keeps the first 5 rows in the answer:

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

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

[10]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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
2012-08-10	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

To calculate the total payload carried by boosters from NASA, one can use a SQL query with a where clause limiting the answer to those rows that the column "Customer" has a value of "NASA (CRS)", as the following:

```
[11]: %%sql select sum("PAYLOAD_MASS__KG_") as "Total Payload Mass [NASA (CRS)]"
                                           from SPACEXTABLE where "Customer"=="NASA (CRS)"

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

[11]:

Total Payload Mass [NASA (CRS)]
45596

Average Payload Mass by F9 v1.1

To calculate the average payload mass carried by booster version F9 v1.1, one can use a SQL query with a where clause limiting the answer to rows with a value of "F9 v1.1" in the column "Booster_Version":

```
[12]: %%sql select avg("PAYLOAD_MASS__KG_") as "AVG Payload Mass Carried by F9 v1.1"
                                       from SPACEXTABLE where "Booster_Version"=="F9 v1.1"

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

[12]:

AVG Payload Mass Carried by F9 v1.1
2928.4

First Successful Ground Landing Date

To find the date of the first successful landing outcome on ground pad, we use a SQL query with where "Landing_Outcome"=="Success (ground pad)" to limit the query to records successful landings on ground pads, and use the min() function to get the minimum value of on the "Date" column:

```
%%sql select min("Date") as "Date of First Succesful Landing on Ground Pad" from SPACEXTABLE
where "Landing_Outcome"=="Success (ground pad)"
```

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

Date of First Succesful Landing on Ground Pad
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

To list the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000, we use a SQL query that looks in rows with "Landing_Outcome" given as "Success (drone ship)" for "PAYLOAD_MASS_KG" between 4000 and 6000, as the following:

```
%%sql select "Booster_Version"
    as "Boosters with payload mass between 4000 and 6000, successfully landed on drone ships"
    from SPACEXTABLE where "Landing_Outcome"=="Success (drone ship)"
    and "PAYLOAD_MASS__KG_" between 4000 and 6000
```

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

Boosters with payload mass between 4000 and 6000, successfully landed on drone ships
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

To calculate the total number of successful and failure mission outcomes, we group the SPACEXTABLE table by the value of "Mission Outcome" column and use the count function for the "Mission Outcome" column .

```
%%sql select "Mission_Outcome" as "Mission Outcome", count("Mission_Outcome") as "Count"
from SPACEXTABLE group by "Mission_Outcome"
```

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

Mission Outcome	Count
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

To list the names of the booster which have carried the maximum payload mass, we use a SQL query with a subquery for getting rows with maximum value on the PAYLOAD_MASS_KG column:

```
%%sql select "Booster_Version" as "Boosters which have carried maximum payload mass"
    from SPACEXTABLE
    where "PAYLOAD_MASS__KG_"==(select max("PAYLOAD_MASS__KG_") from SPACEXTABLE)
```

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

Boosters which have carried maximum payload mass
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

To list the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015, we use the following SQL query. Since SQLLite does not support MONTHNAME() function, we need to use substr(Date, 6,2) as month to get the months and substr(Date,1,4)='2015' for year.

```
%%sql select substr("Date",6,2) as "Month in 2015", "Booster_Version", "Launch_Site",
                                    "Landing_Outcome"
    from SPACEXTABLE where "Landing_Outcome"=="Failure (drone ship)"
    and substr("Date", 1, 4)=="2015" limit 5
```

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

Month in 2015	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

To rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the dates 2010-06-04 and 2017-03-20, in descending order, we use 'between' clause in the SQL query to specify the starting and ending dates of the time interval.

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

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

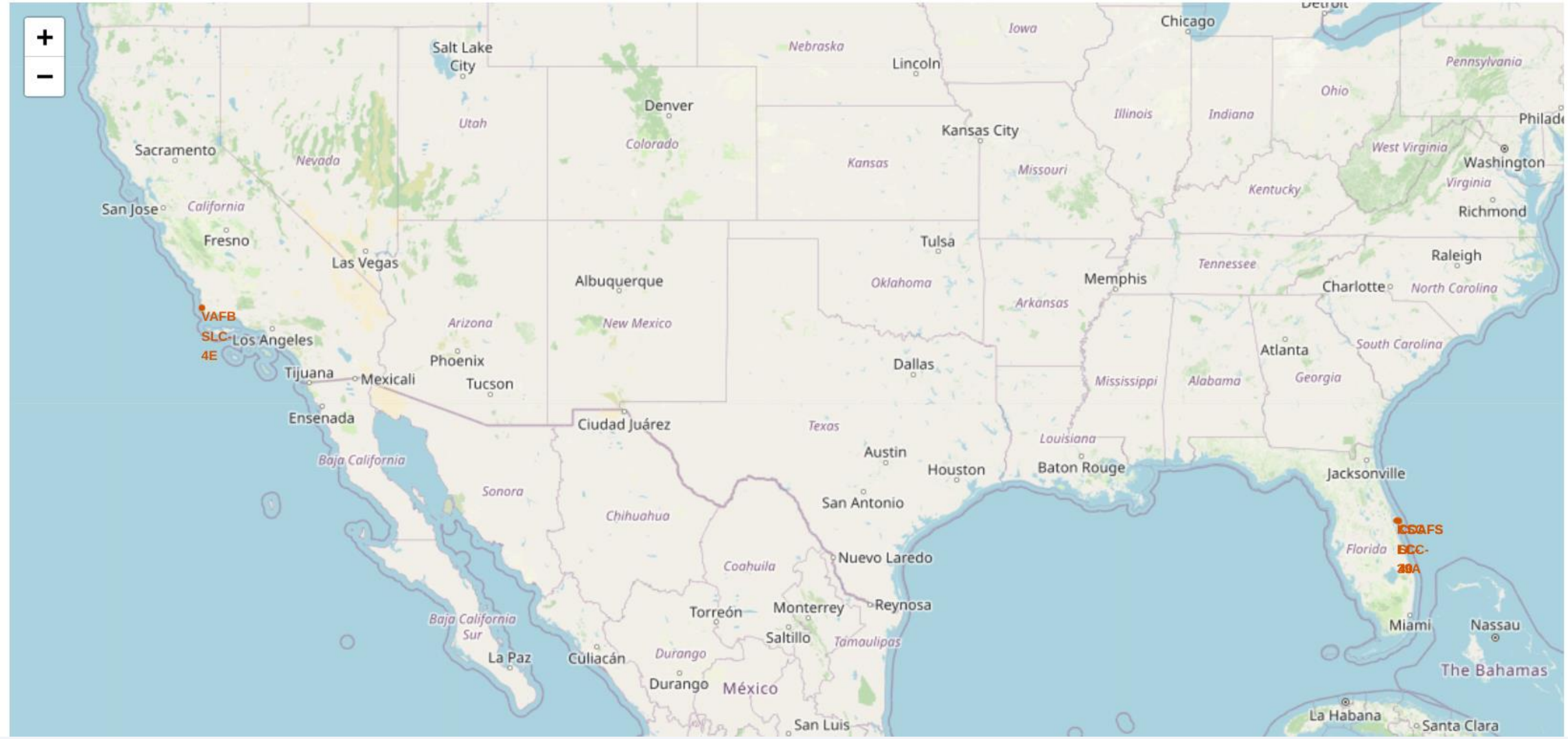
Landing Outcome	Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

launch sites marked with circles and names

AS shown on the map, the launch sites (marked in red) are located near coastlines and away from major cities and crowded areas.

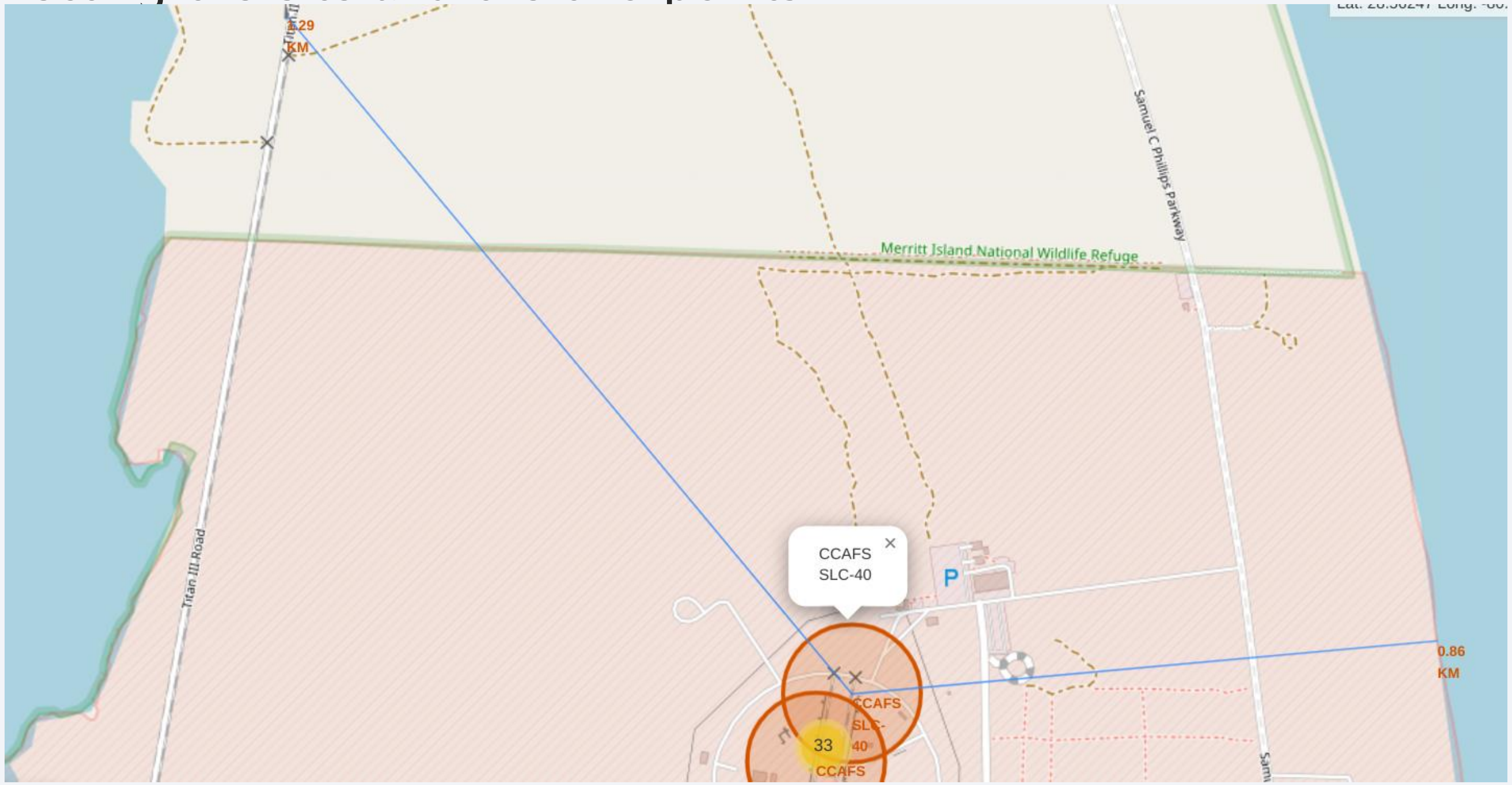


Successes and failures on an interactive map

Successful and failed landing-attempts shown on each site as colored labels; Comparing the green and red markers of different launch sites on the folium maps show that the KSC-39A launch site has a better success rate in general.

CCAFS SLC-40 launch site

The following map shows the proximity of the launch site CCAFS SLC-40 to the closest railway and coastline. This site is away from the closest coastline by a distance of 0.86 Km, whereas its distance to the nearest railway is 1.29 Km, as marked in red at the end of the blue lines connecting the site and the two points.

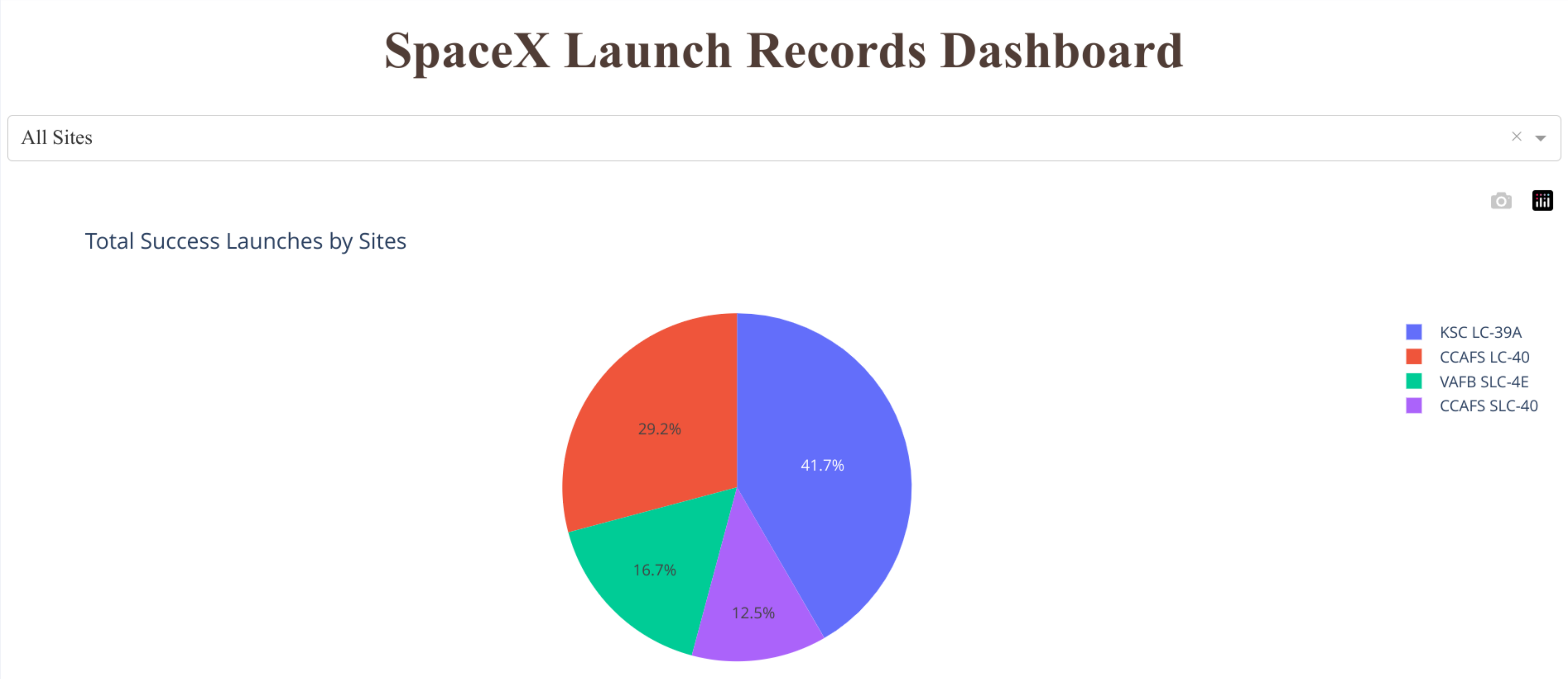


Section 4

Build a Dashboard with Plotly Dash

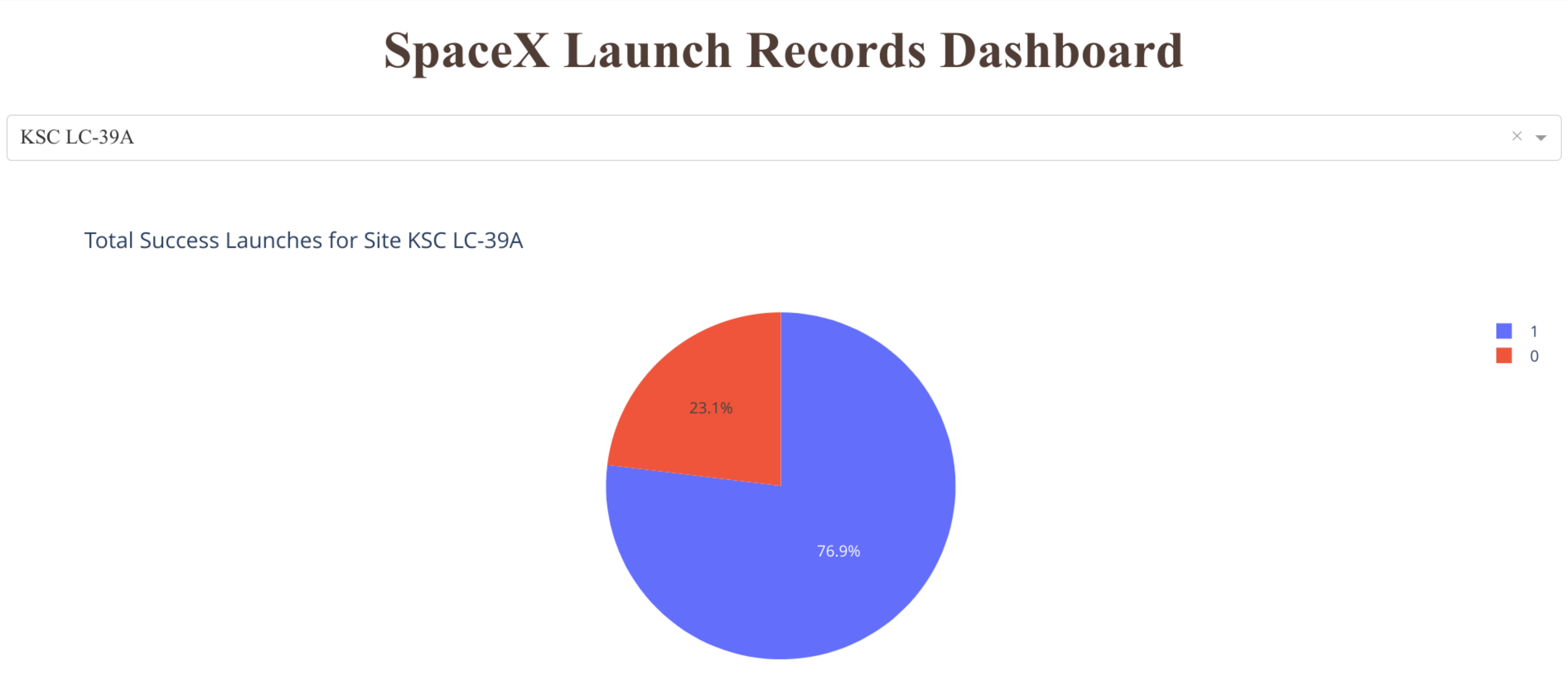
SpaceX launch records dashboard: success counts for all sites

The following pie chart from the SpaceX launch records dashboard shows the successful launches for all sites. It is seen that the site KSC LC-39A has the largest share (41.7%) of successful launches among the four sites.



SpaceX launch records dashboard: success for KSC LC-39A

The following pie chart from the SpaceX launch records dashboard shows that 76.9% of launches were successful at the site KSC LC-39A; 76.9% successful launches.



SpaceX launch records dashboard: payload vs launch outcome

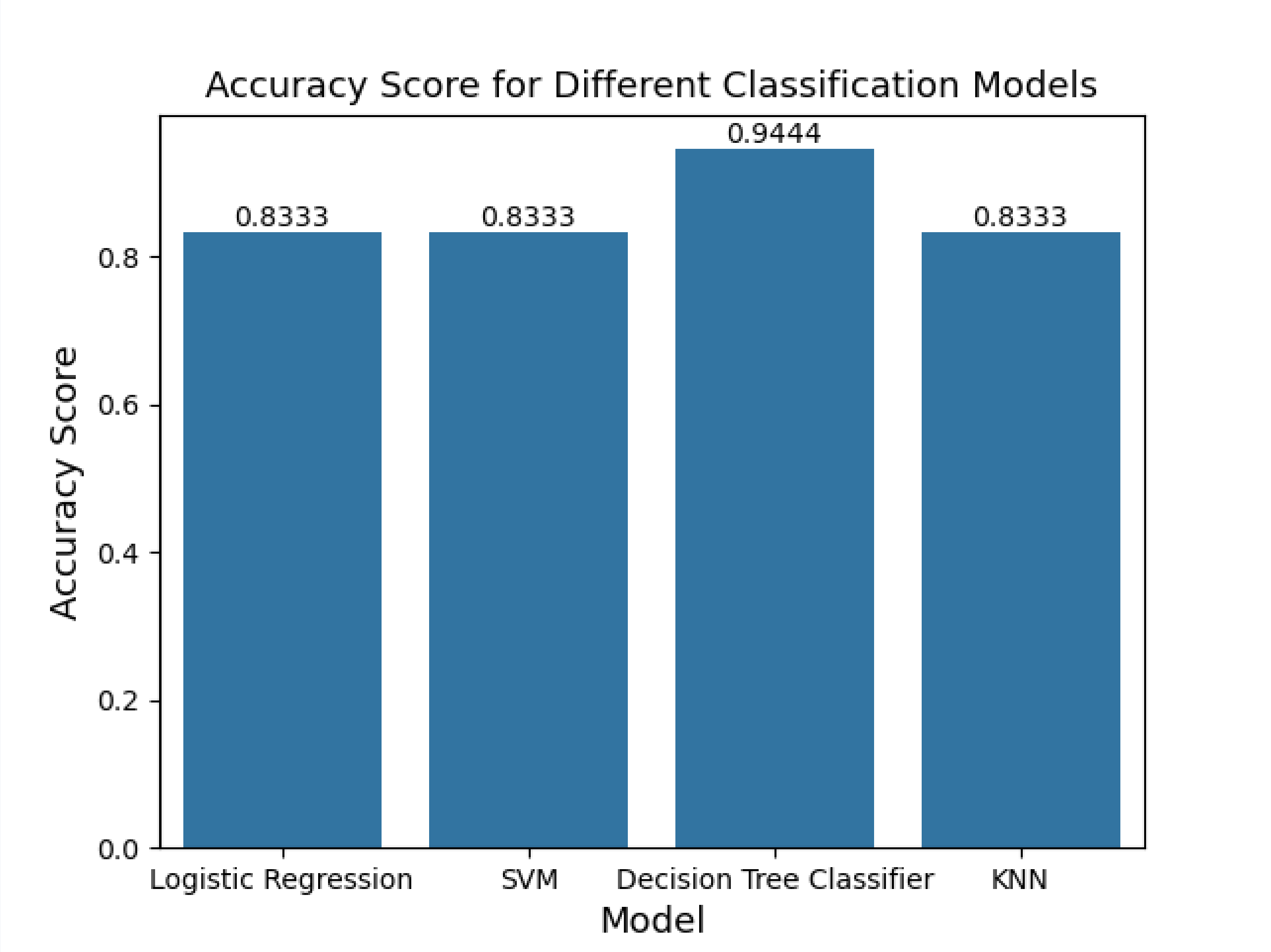
The following scatter plots from the SpaceX launch records dashboard shows launch outcomes for a range of payloads up to 10000 for all sites. **(1)** FT boosters are the most successful and v1.1 boosters are the least successful boosters for payloads up to 5000. **(2)** for payloads more than 5000, v1.0 and FT are the only two boosters used for launching rockets. **(3)** launches are more frequent in the range of 0-5000 compared to 5000-10000 for the payloads. **(4)** B5 and B4 are the least used boosters (used only once and twice, respectively) in the range 0-10000, whereas v1.1 and FT are the most used boosters in this range.

Section 5

Predictive Analysis (Classification)

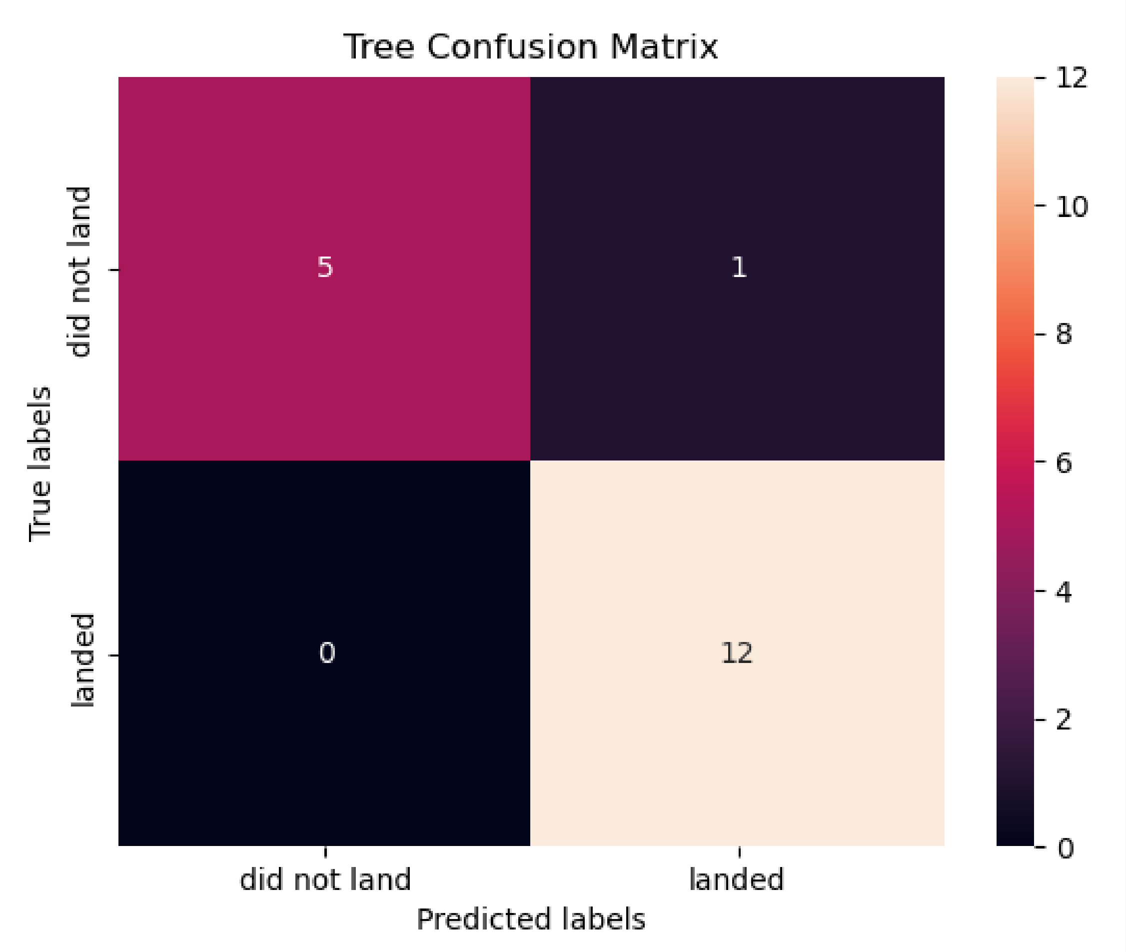
Classification Accuracy

The Decision Tree Classifier model has the highest classification accuracy.



Confusion Matrix

Confusion matrix of the best performing model, i.e. Decision Classification Tree, show that the model do not predict any false negative. The model results only one false positive which is better than the 3 false positive prediction of other three models.



Conclusions

- Significant correlations exist between the results of first-stage landing attempts and several launch parameters.
- Booster version, type of orbit, payload mass, launch site, and flight number show significant correlation with the landing outcome.
- Success rate of landing attempts of SpaceX launches is quickly growing by time.
- A Classification Model has to include all the important independent variables to be able to make accurate predictions for the landing outcomes.

Appendix

- All relevant Python codes, Jupyter Notebooks, and data sets created during this project can be found on a dedicated GitHub repository on the following link:

https://github.com/rezakp0/applied-data-science-capstone

Thank you!