IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Eva Zhang
31/05/2024

Outline

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

Executive Summary

- The SpaceX Capstone project aims to predict the successful and unsuccessful launches of the Falcon 9 rockets. We will conduct Exploratory Data Analisis (EDA) and predictive Machine Learning to help us solve this issue.
- Among the various classification algorithms used in this project, the decision tree algorithm stands out and offers the best result when comparing the different accuracies.

Introduction

- 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. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.
- We will predict if the Falcon 9 first stage will land successfully.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Use of SpaceX APIs and Web scrapping
- Perform data wrangling
  - We used the "outcome" column in order to create the binary "Class" column to use it as the target label.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Splitting of data (training/testing sets), find the best Hyperparameter for SVM, Classification Trees, and Logistic Regression using SearchGridCV, evaluate classification models on accuracy

Data Collection

- The SpaceX Api URL: https://api.spacexdata.com/v4/
- The endpoints :
  - rockets/[rocket]: booster name (in order to filter on Falcon 9)
  - launchpads/[launchpad]: longitude, latitude, launch site
  - payloads/[payloads]: payload mass (kg), orbit
  - cores/[cores]: outcome of the landing, the type of the landing, number of flights with that core, whether gridfins were used, whether the core is reused, whether legs were used, the landing pad used, the block of the core which is a number used to seperate version of cores, the number of times this specific core has been reused, and the serial of the core.
  - launches/past: rocket launch data, flight number, date utc
- Web Scraping URL: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

Data Collection – SpaceX API

- The endpoint used to get the launches data is https://api.spacexdata.com/v4/launches/past (see image)

- https://github.com/evazhangeva/applied-data-science-capstone/blob/main/1-data-collection-API.ipynb

In [6]:
```python
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```python
response = requests.get(spacex_url)
```

Check the content of the response

In [8]:
```python
print(response.content)
```


```
b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"li
tps://images2.imgbox.com/94/f2/NN6Ph45r_o.png","large":"https://images2.imgbox.com/5b/02/
{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"origin
ebcast":"https://www.youtube.com/watch?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article
196-spacex-inaugural-falcon-1-rocket-lost-launch.html","wikipedia":"https://en.wikipedia.
_fire_date_utc":"2006-03-17T00:00:00.000Z","static_fire_date_unix":1142553600,"net":false
0d95eda69955f709d1eb","success":false,"failures":[{"time":33,"altitude":null,"reason":"me
tails":"Engine failure at 33 seconds and loss of vehicle","crew":[],"ships":[],"capsules"
6c3bb0006eeb1e1"],"launchpad":"5e9e4502f5090995de566f86","flight_number":1,"name":"Falcon
4T22:30:00.000Z","date_unix":1143239400,"date_local":"2006-03-25T10:30:00+12:00","date_pr
g":false,"cores":[{"core":"5e9e289df35918033d3b2623","flight":1,"gridfins":false,"legs":f
ng_attempt":false,"landing_success":null,"landing_type":null,"landpad":null}],"auto_updat
h_library_id":null,"id":"5eb87cd9ffd86e000604b32a"},{"fairings":{"reused":false,"recovery
d":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.com/f9/4a/ZboXReNb
mages2.imgbox.com/80/a2/bkWotCIS_o.png"},"reddit":{"campaign":null,"launch":null,"media":
ckr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?
d":"Lk4zQ2wP-Nc","article":"https://www.space.com/3590-spacex-falcon-1-rocket-fails-reach
a":"https://en.wikipedia.org/wiki/DemoSat"},"static_fire_date_utc":null,"static_fire_date
indow":0,"rocket":"5e9d0d95eda69955f709d1eb","success":false,"failures":[{"time":301,"alt
nic oscillation leading to premature engine shutdown"}],"details":"Successful first stage
```

Data Collection - Scraping

- Web Scraping using this URL: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

- https://github.com/evazhangeva/applied-data-science-capstone/blob/main/2-data-collection-webscraping.ipynb

In [15]:

```python
# Use the find_all function in the BeautifulSoup object, with element type `
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

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

In [16]:

```python
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time"
C</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="I
s">Version,<br/>Booster</a> <sup class="reference" id="cite_ref-booster_11-0">
a></sup>
</th>
<th scope="col">Launch site
</th>
<th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-0"><a hre
>
</th>
<th scope="col">Payload mass
</th>
<th scope="col">Orbit
```

Data Wrangling

- We used the "outcome" column in order to create the binary "Class" column to use it as the target label

- https://github.com/evazhangeva/applied-data-science-capstone/blob/main/3-data-wrangling.ipynb

In [18]:
```
df.head(5)
```

Out[18]:

LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0

EDA with Data Visualization

- Scatter plot for payload mass against flight number categorized by class
- Scatter plot for launch site against flight number categorized by class
- Scatter plot for launch site against payload mass categorized by class
- Bar chart for the success rate of each orbit
- Scatter plot for orbit against flight number categorized by class
- Scatter plot for orbit against payload mass categorized by class
- Linear plot of the launch success yearly trend

- https://github.com/evazhangeva/applied-data-science-capstone/blob/main/5-EDA-data-vizualisation.ipynb

EDA with SQL

- Use of Db2 database and sqlite3 library, table SPACEXTABLE
- Use of SELECT, DISTINCT, AS, FROM, WHERE, LIMIT, LIKE, SUM(), AVG(), MIN(), BETWEEN, COUNT(), and YEAR()
- Queries about launch sites names, total payload mass, average payload mass
- Queries about successful and unsuccessful landing outcome
- Queries to rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))
- https://github.com/evazhangeva/applied-data-science-capstone/blob/main/4-EDA-SQL.ipynb

Build an Interactive Map with Folium

- Use of map objects such as markers and circles showing different launch sites and launch outcomes for each launch sites
- Use of line map object to draw distances between launch sites and cities, railways, highways and coastlines
- https://github.com/evazhangeva/applied-data-science-capstone/blob/main/6-data-vizualisation-folium.ipynb

Build a Dashboard with Plotly Dash

- Creation of pie chart to show total successful launches count for all sites or show the Success vs. Failed counts for a specific launch site selected via a dropdown list
- Creation of an interactive scatter plot to show the correlation between payload and launch success, filter by launch site and payload range
- Use of callback funtions and plotly.express library (px.pie() and px.scatter())
- https://github.com/evazhangeva/applied-data-science-capstone/blob/main/7-dash_app.py

Predictive Analysis (Classification)

- Perform exploratory Data Analysis and determine Training Labels
  - create a column for the class (0 Failed, 1 Success)
  - Standardize the data using a Standard Scaler
  - Split into training data and test data: 20% of test data
- Find best Hyperparameter for SVM, Classification Trees and Logistic Regression using SearchGridCV with cv=10
- Compare confusion matrix and accuracies on both training and testing data

- https://github.com/evazhangeva/applied-data-science-capstone/blob/main/8-machine-learning-prediction.ipynb

	Accuracy	Score
Logistic Regression	0.846429	0.833333
SVM	0.848214	0.833333
Decision tree	0.889286	0.833333
KNN	0.848214	0.833333

Results

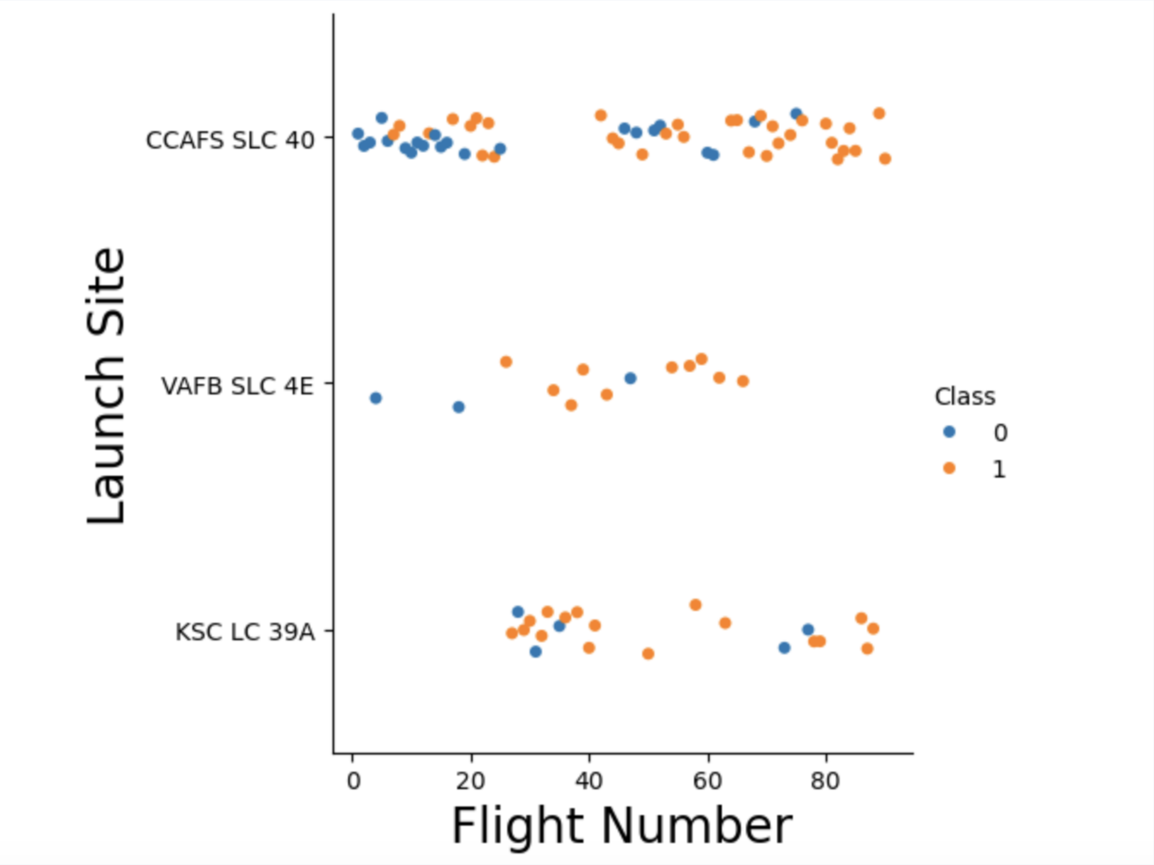
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

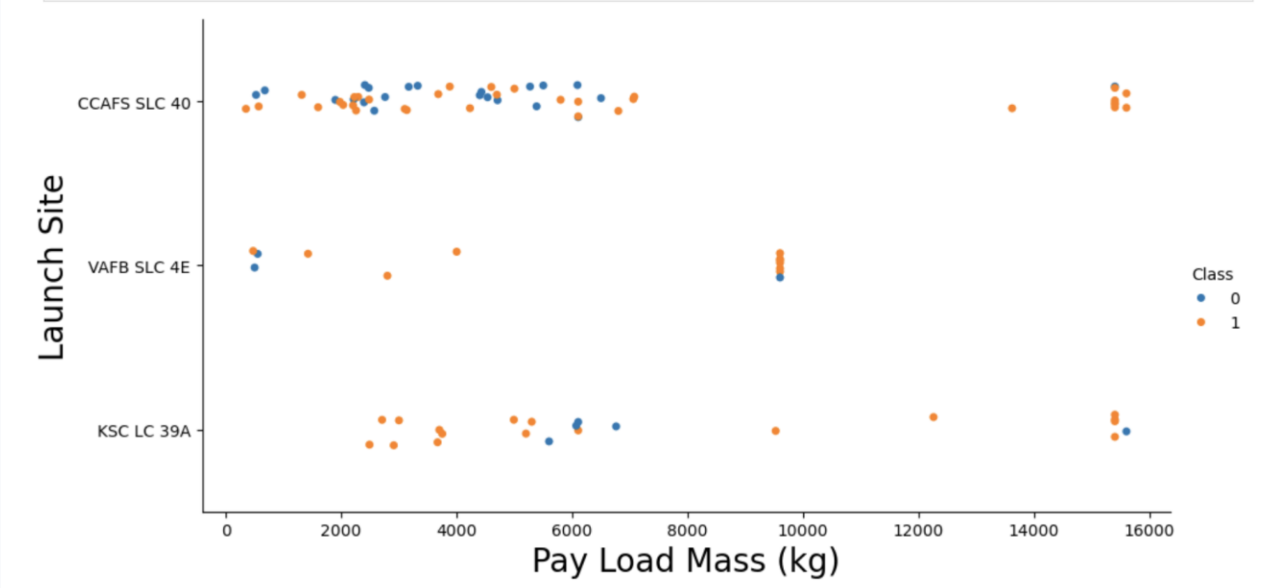
Flight Number vs. Launch Site

- The success rate increases after Flight Number 40



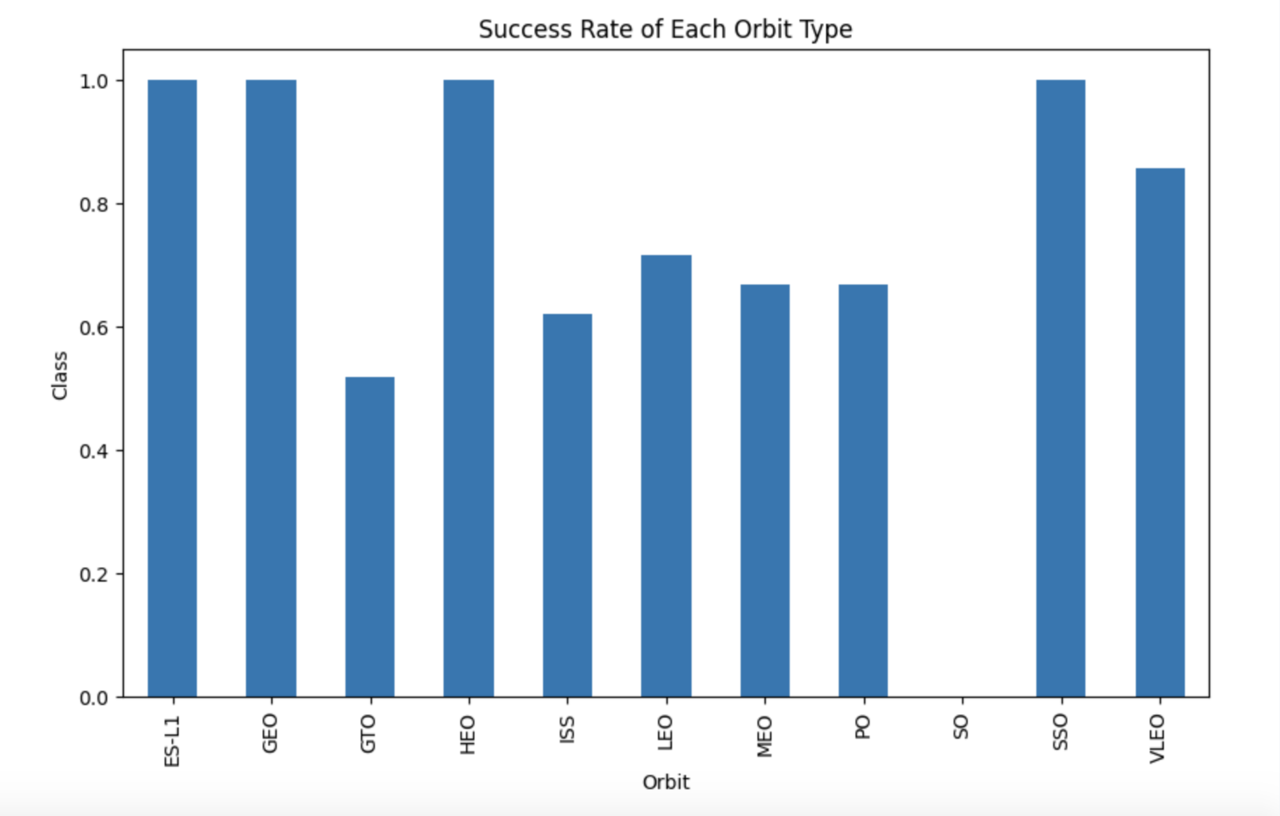
Payload vs. Launch Site

- The success rate increases as the Pay Load Mass increases



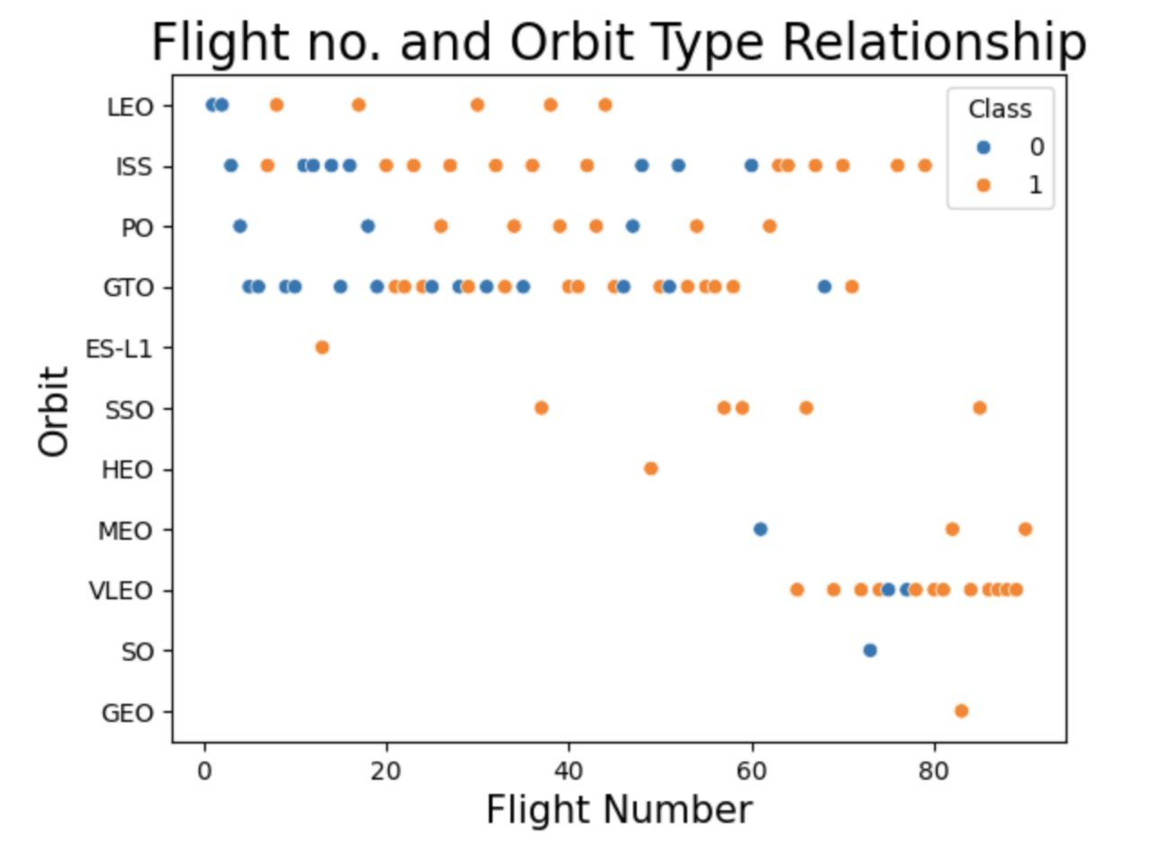
Success Rate vs. Orbit Type

- ESL1, GEO, HEO and SSO have a success rate of 100%



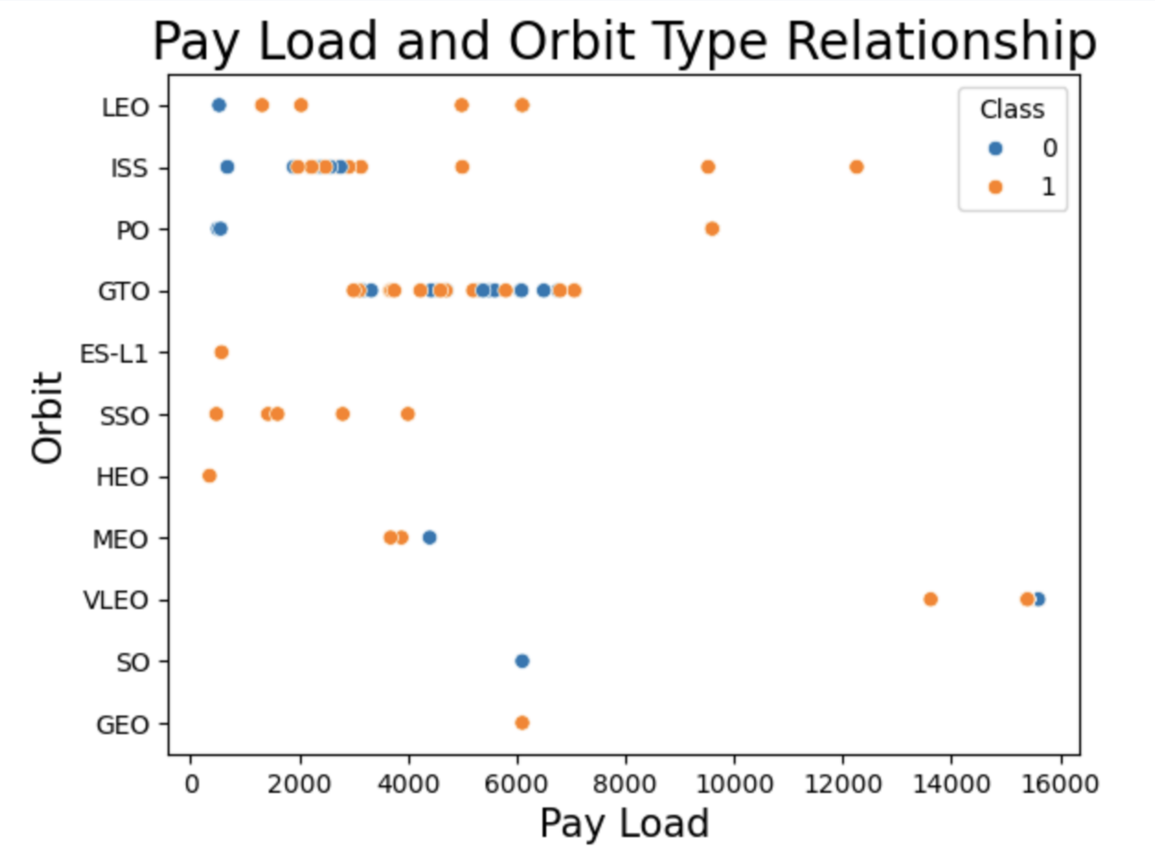
Flight Number vs. Orbit Type

- 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



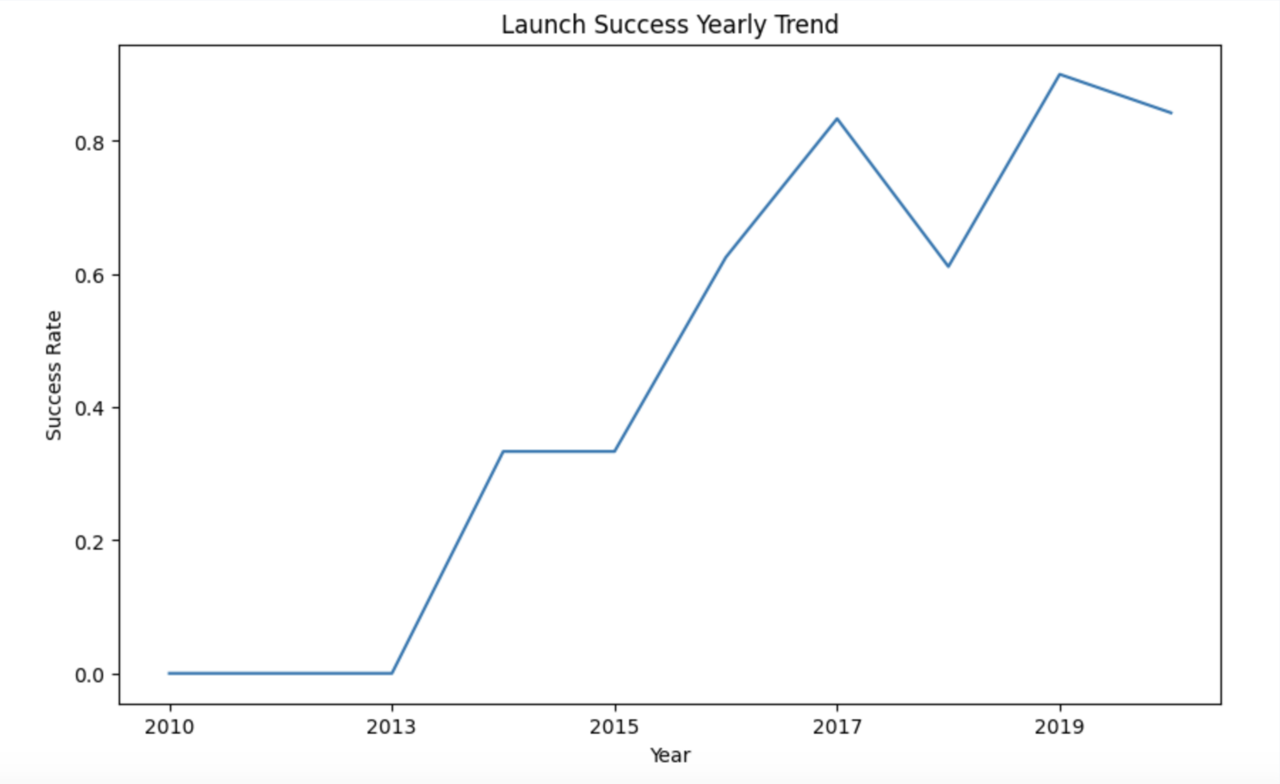
Payload vs. Orbit Type

- 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.



Launch Success Yearly Trend

- The success rate since 2013 kept increasing until 2020



All Launch Site Names

- SELECT DISTINCT Launch_Site from SPACEXTABLE
- DISTINCT is to have unique value in the query result

Launch Site Names Begin with 'CCA'

- SELECT * from SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
- Like is to filter on 'CCA%', % is a wildcard character and LIMIT is to return the first 5 rows

Total Payload Mass

- select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer = 'NASA (CRS)'
- Sum() to calculate the sum of all payload mass, where clause to filter on NASA (CRS)

Average Payload Mass by F9 v1.1

- select avg(PAYLOAD_MASS__KG_) from SPACEXTABLE where Booster_Version LIKE 'F9 v1.1';
- Avg() function to calculate the average, Where … like … to filter on F9 v1.1

First Successful Ground Landing Date

- select min(Date) as min_date from SPACEXTABLE where Landing_Outcome = 'Success (ground pad)';
- min date to have the earlier date that has a successful outcome

Successful Drone Ship Landing with Payload between 4000 and 6000

- **select** Booster_Version **from** SPACEXTABLE
- **where** (PAYLOAD_MASS__KG_ > 4000 **and** PAYLOAD_MASS__KG_ < 6000)
- **and** (Landing_Outcome = 'Success (drone ship)');

- Use of Where clause and AND operator to filter on payload mass and landing outcome

Total Number of Successful and Failure Mission Outcomes

- **select** Mission_Outcome, **count**(Mission_Outcome) **as** counts **from** SPACEXTABLE **group by** Mission_Outcome;
- Use of Group by clause and count() agregate to count the number of launches for each mission outcome

Boosters Carried Maximum Payload

- **select** Booster_Version, PAYLOAD_MASS__KG_ **from** SPACEXTABLE
- **where** PAYLOAD_MASS__KG_ = (**select max**(PAYLOAD_MASS__KG_) **from** SPACEXTABLE);
- We use a sub query to match to payload mass with the max payload mass in the table

2015 Launch Records

- **select** substr(Date, 6,2) **as Month**, Landing_Outcome, Booster_Version, Launch_Site **from** SPACEXTABLE
- **where** Landing_Outcome = 'Failure (drone ship)' **and** substr(Date,0,5)='2015'

- **Use of substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year**

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

- **select** Landing_Outcome, **count(*) as** LandingCounts **from** SPACEXTABLE
- **where** Date **between** '2010-06-04' **and** '2017-03-20'
- **group by** Landing_Outcome
- **order by count(*) desc**;

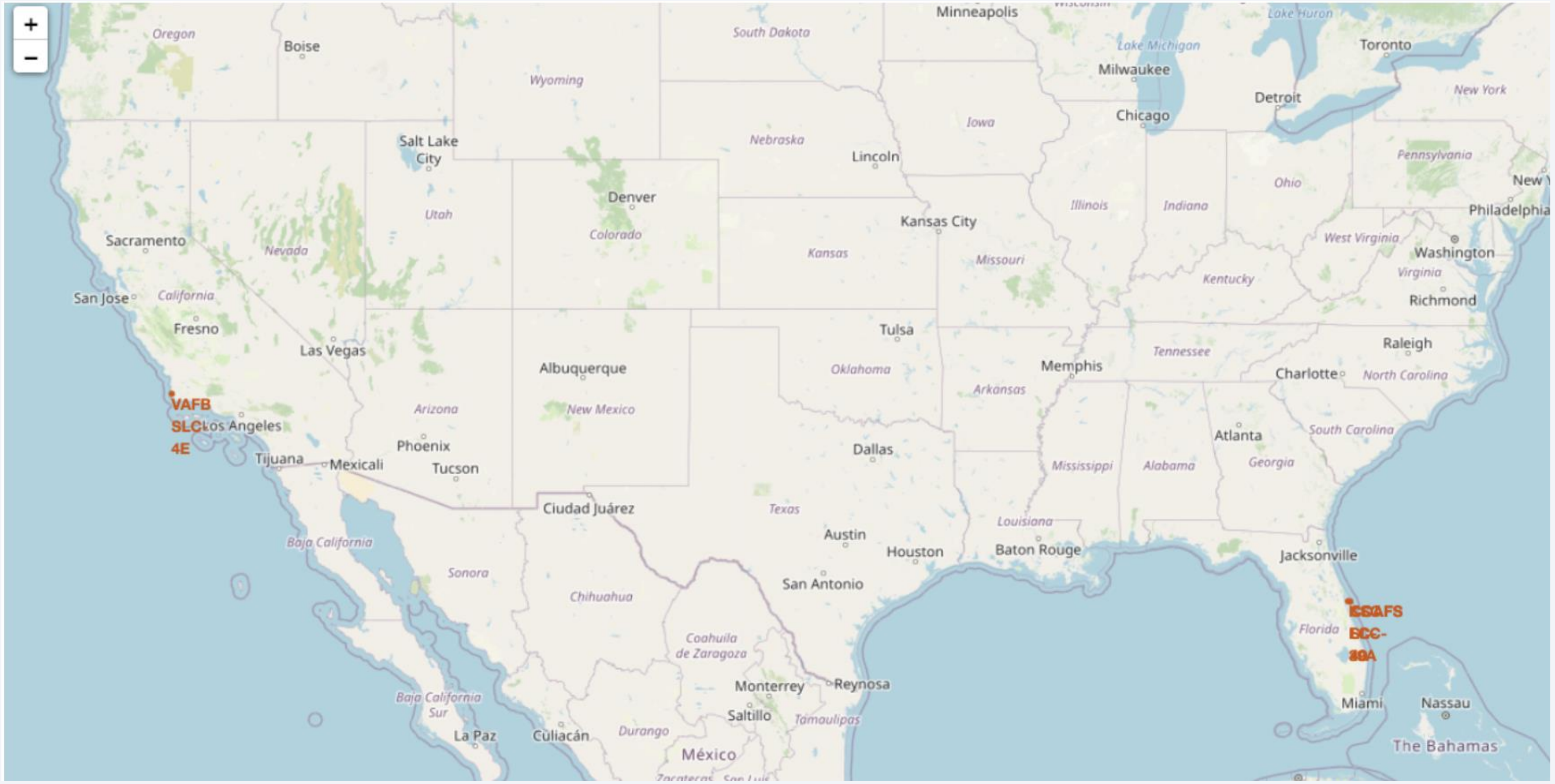
- Where clause to filter on the date
- Group by and count agregate to count the number of launches for each landing outcome
- Order by the number of launches, in descending order (desc)

Section 3

Launch Sites Proximities Analysis

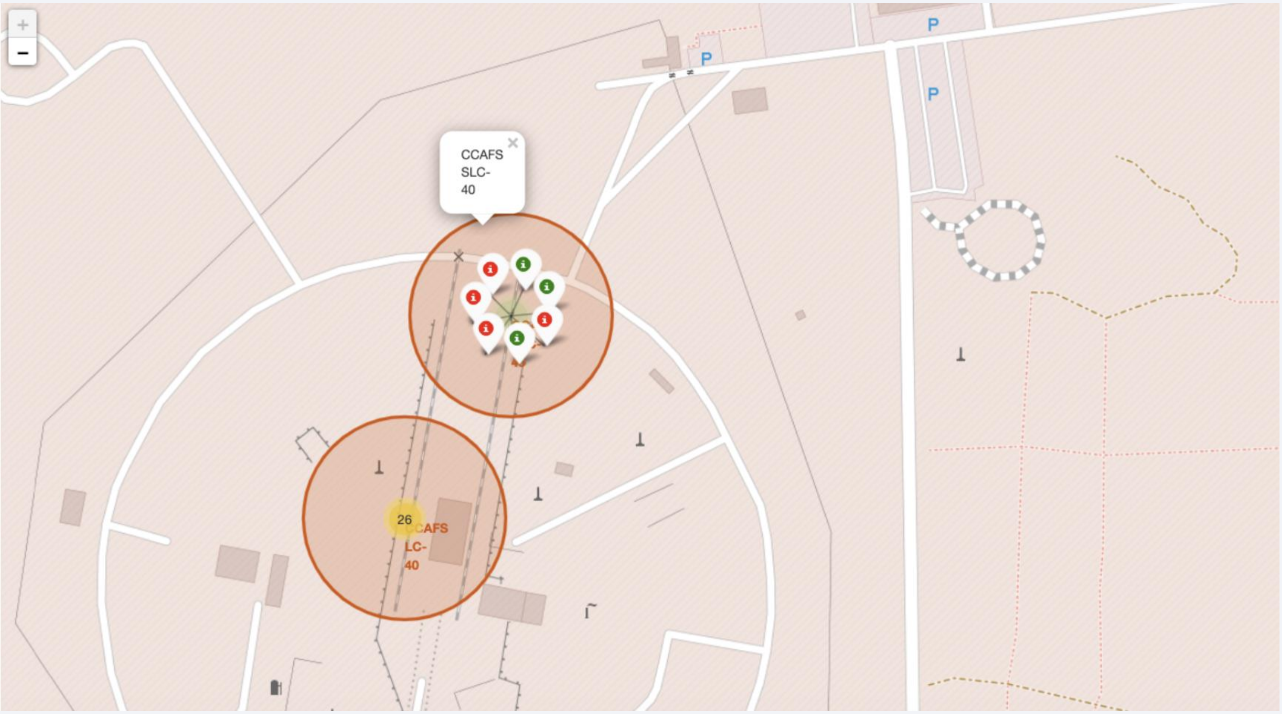
Launch Sites' Location marked on a folium map

- We marked CCAFS LC-40, CCAFS SLC-40, KSC LC-39A and VAFB SLC-4E launch site, they are all very close to coastlines



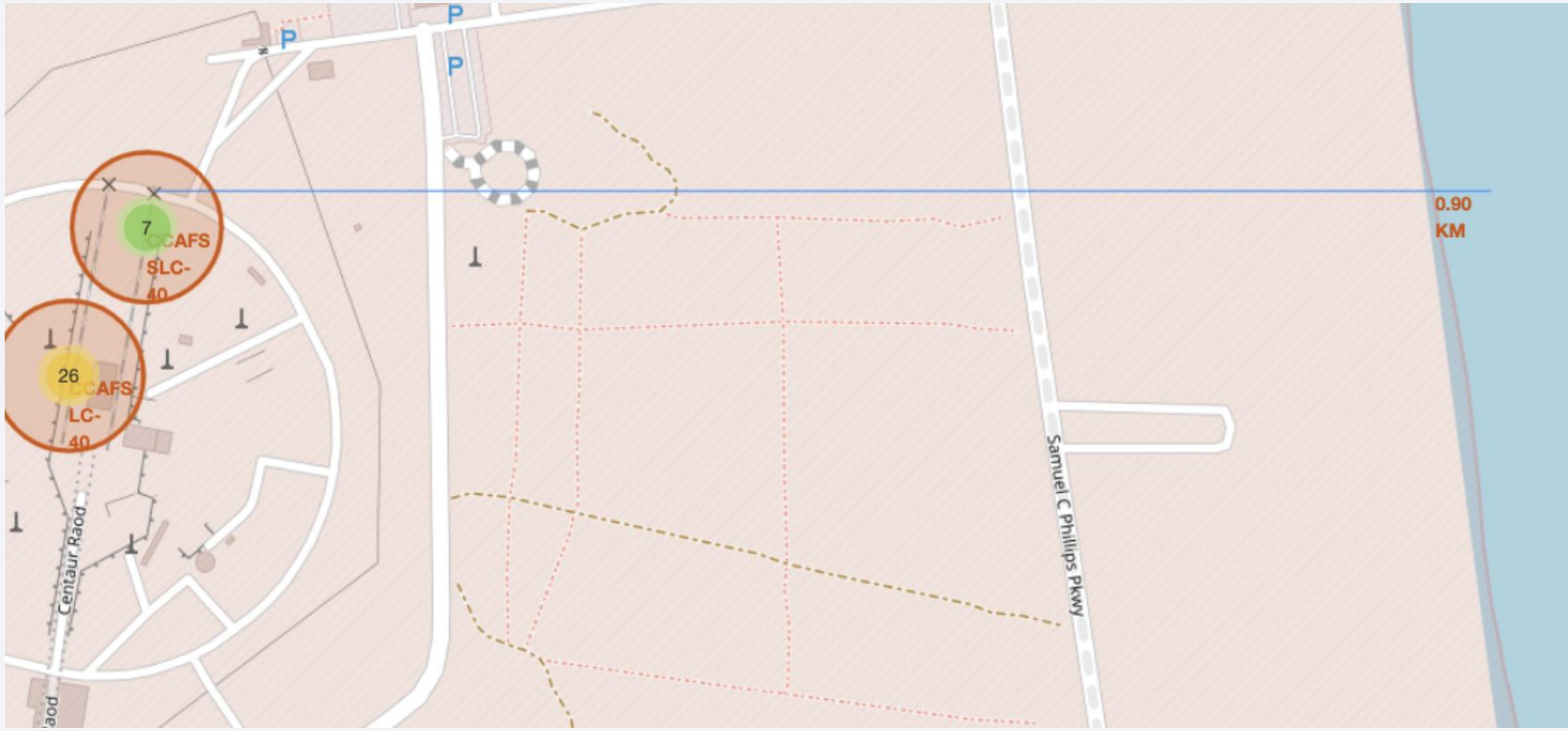
Color labeled launch outcomes of CCAFS SLC-40

- We use Marker cluster to help organize the markers. We have green marker for a successful outcome and a red marker for a failed outcome



Distance to coastline of CCAFS SLC-40

- The launch sites are near to coastlines but far from railways, highways, cities

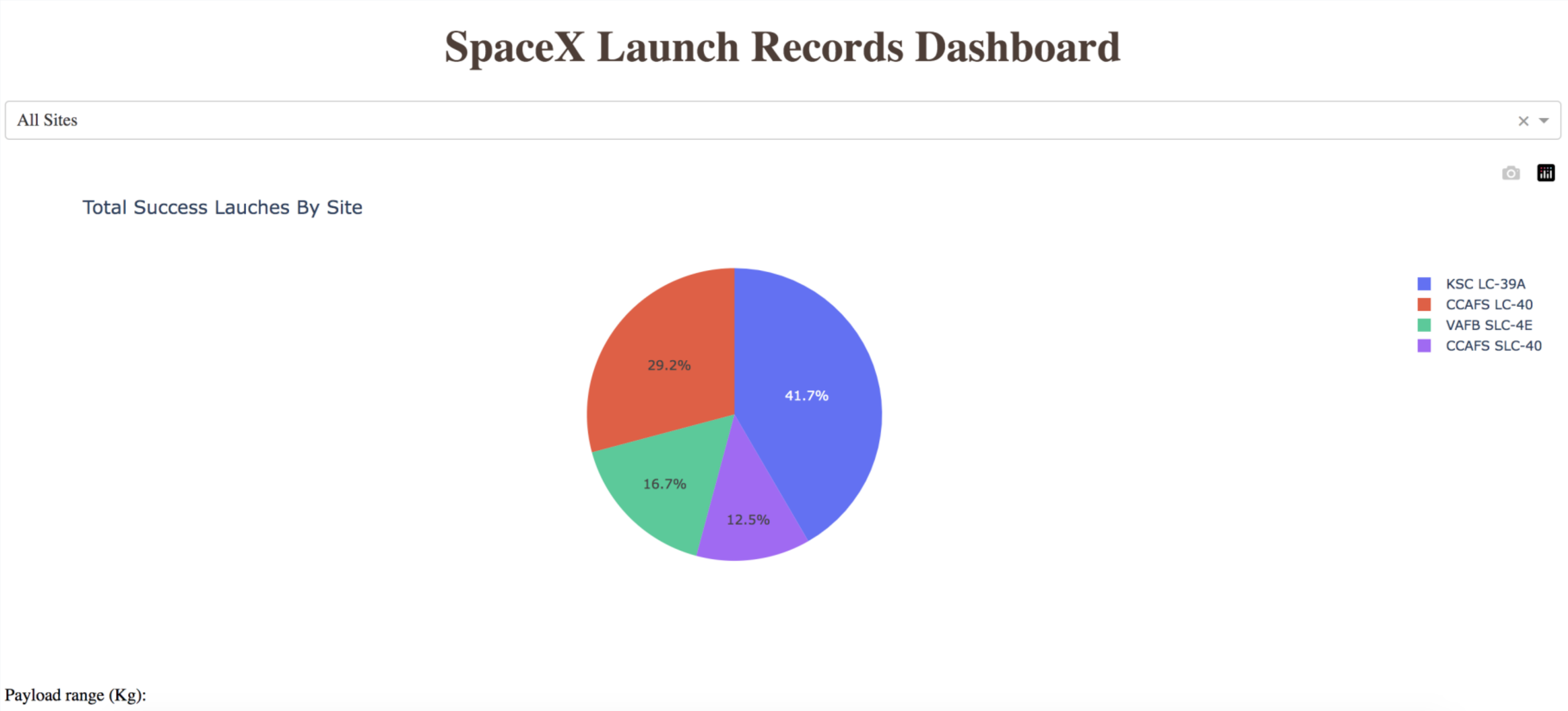


Section 4

Build a Dashboard with Plotly Dash

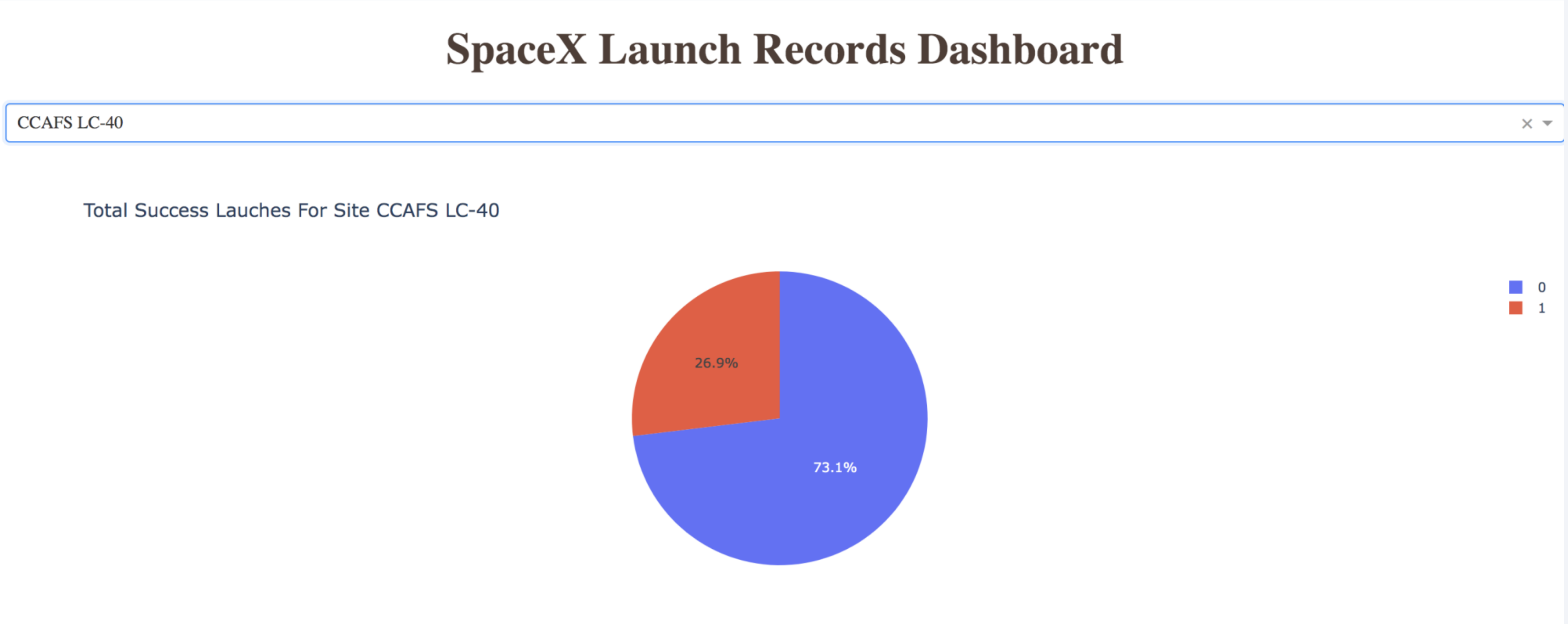
Total Success launches by site

- KSC LC-39A has the largest successful launches



Launch site with highest launch success ratio

- CCAFS LS-40 has a launch success ratio of 26.9%



Payload VS. Launch Outcome scatter plot

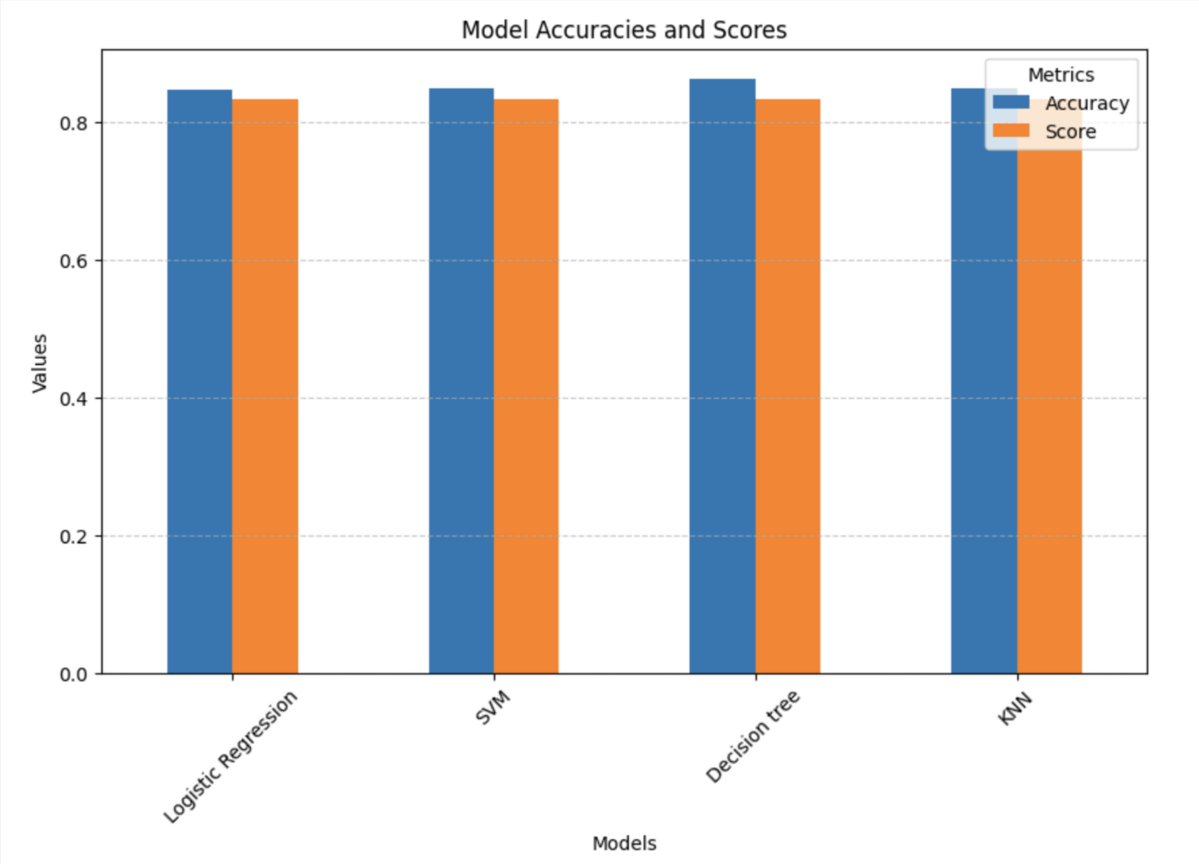
- The 2000-4000 payload range has the highest launch success rate
- The 6000-8000 payload range has the lowest launch success rate
- FT F9 Booster version has the highest launch success rate

Section 5

Predictive Analysis (Classification)

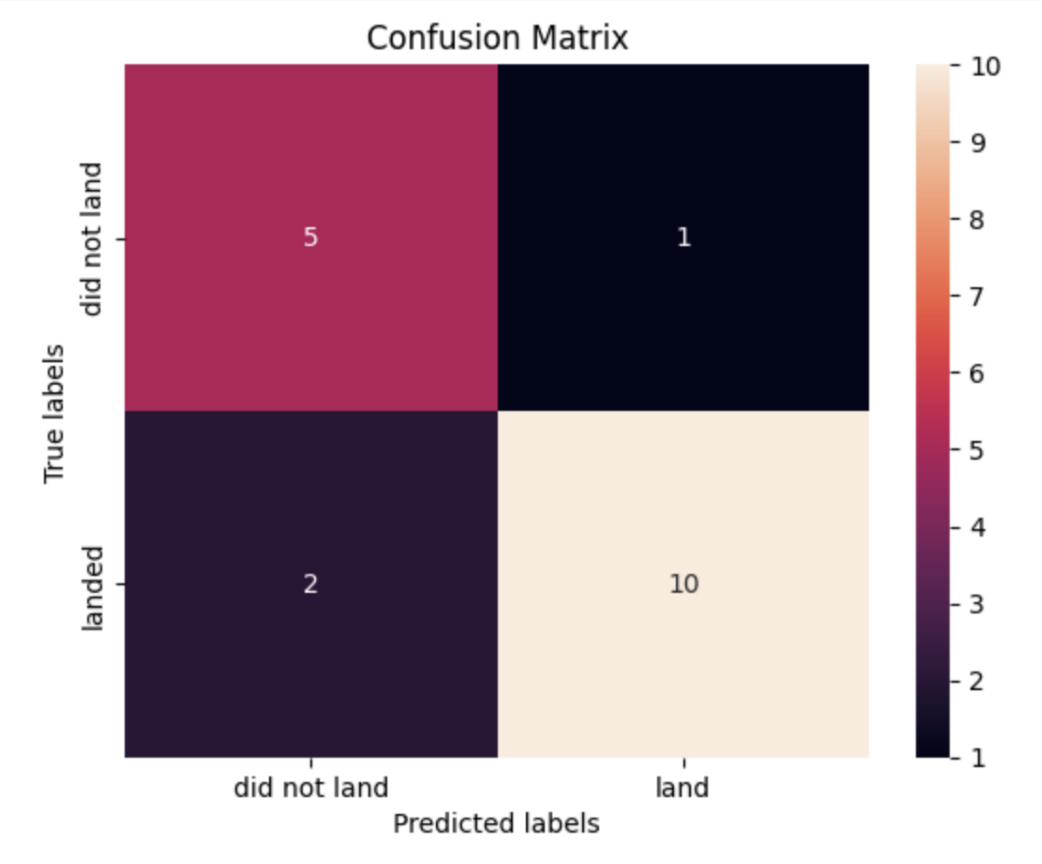
Classification Accuracy

- The decision tree has the highest accuracy



Confusion Matrix

- The confusion matrix of the best performing model beacause it has the lowest number of false negative



Conclusions

- According to the table and bar chart, the decision tree might be the best fit. Also, according to the confusion matrix, it is the matrix with fewer false positives.

	Accuracy	Score
Logistic Regression	0.846429	0.833333
SVM	0.848214	0.833333
Decision tree	0.889286	0.833333
KNN	0.848214	0.833333

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!