IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Alexander Sheldrick

09.10.2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through APIs
  - Data Collection through Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis through Data Visualization
  - Interactive Visual Analytics with Folium
  - Making a prediction with Machine Learning Methods
- Summary of all results
  - Exploratory Data Analysis Results
  - Interactive Analytics Results and Screenshots
  - Predictive Analytics Results

Introduction

- Project background and context

  The goal of this project is to estimate costs of Falcon 9 rocket launches. This is primarily determined from whether the first stage will land successfully. 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. In this module, you will be provided with an overview of the problem and the tools you need to complete the course.

- Problems you want to find answers
  - What data can we find to extract patterns from that will be helpful to predict rocket launch and recovery success rates
  - What are the primary features that determine mission outcome
  - How can we best model the outcome for future launches

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected by web scraping Wikipedia pages and from the SpaceX API
- Perform data wrangling
  - Data was cleaned and the mission outcomes were 1-hot encoded for recovery success
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Data was analyzed with regression analysis and machine learning methods

Data Collection

- We Develop Python code to handle json responses from the SpaceX API
- The Data are loaded into and manipulated via a Pandas data frame
- Data is checked for missing or NAN values and replaced or removed appropriately
- Further launch records are scraped from Wikipedia SpaceX entries, organized with BeautifulSoup and analyzed with Pandas

Data Collection – SpaceX API

- First call the get method on the API endpoint, then normalize and clean the data for future use.

- API Notebook link:

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

1. Get request for rocket launch using SpaceX launch data API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

3. Data filtering, formatting and cleaning

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that h
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Data Collection - Scraping

- Scrape Falcon9 Wikipedia launch records
- Parse, convert and filter response with BeautifulSoup
- GitHub URL of the completed web scraping notebook:

https://github.com/AlexSheldrick/DataScienceCert/blob/main/jupyter-labs-webscraping.ipynb

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

Create a `BeautifulSoup` object from the HTML `response`

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text)
```

2. Extract relevant data from soup object:

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

for table_number,table in enumerate(first_launch_table.find_all('th')):
    # get table row
    name = extract_column_from_header(table)
    print(name)
    if name is not None and len(name) > 0:
        column_names.append(name)
```

Data Wrangling

- Missing data is removed or replaced by column average in previous step. Class labels are one-hot encoded.
- GitHub URL completed data wrangling:

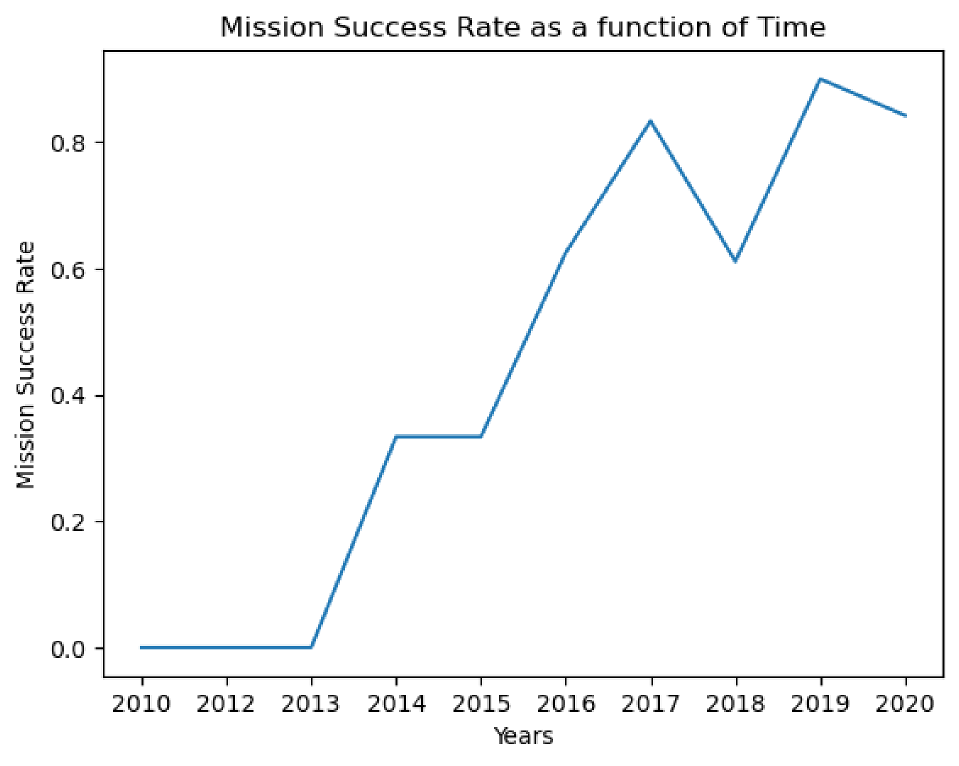
https://github.com/AlexSheldrick/DataScienceCert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_1_L3_labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

EDA with Data Visualization

We used categorical and scatter plots to find relationships between features and mission success rate. Finally, we found a strong relationship between launch Number (i.e. recency) and mission success with a line-plot, implying that SpaceX is increasing mission success rates with successive launches.

GitHub URL completed EDA:

https://github.com/AlexSheldrick/DataScienceCert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb



EDA with SQL

SQL queries:

- Unique launch sites, launches from particular sites, payload launched by particular client or in particular range.
- Date of first successful launch
- Total number of successful and failure mission outcomes
- Finding the booster version and launch site names for failed drone ship missions
- Etc.

GitHub URL of completed EDA with SQL

https://github.com/AlexSheldrick/DataScienceCert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- The lauch sites were marked and the launches from those sites were added as cluster objects, with a color indicating launch success or failure
- Further we marked the closest city, highway, ocean and railroad to a specific launch site, as they are strategically chosen to be a convenient but low risk launch facility close to the equator
- GitHub URL of completed interactive map with Folium map

https://github.com/AlexSheldrick/DataScienceCert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash where the user can interactively explore the success rates of various launch sites as pie graphs
- The relationship of mission outcome vs payload mass (Kg) was plotted as a scatter graph
- GitHub URL Plotly Dash app

https://github.com/AlexSheldrick/DataScienceCert/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- Data is extracted from Pandas into Numpy arrays and then split into train and test batches
- We initialize and train various machine learning models on the training data and cross validate the hyper parameter settings
- Models are then finally evaluated on the test set for accuracy and other relevant metrics
- GitHub URL of completed predictive analysis

https://github.com/AlexSheldrick/DataScienceCert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

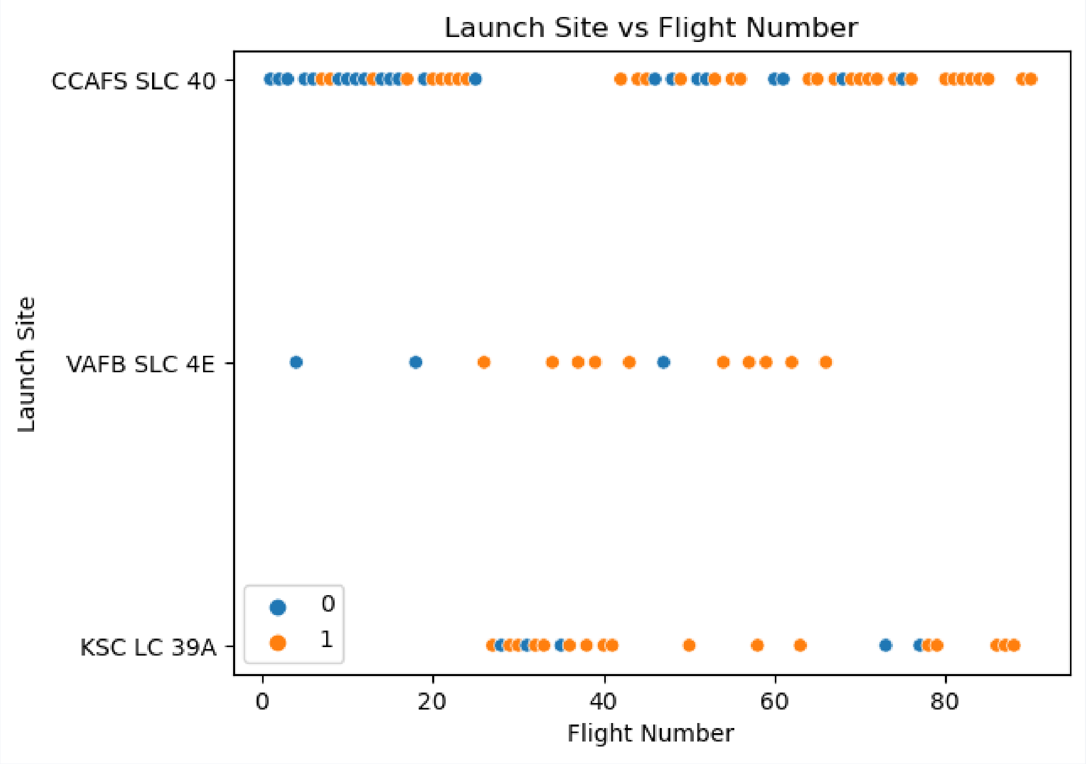
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

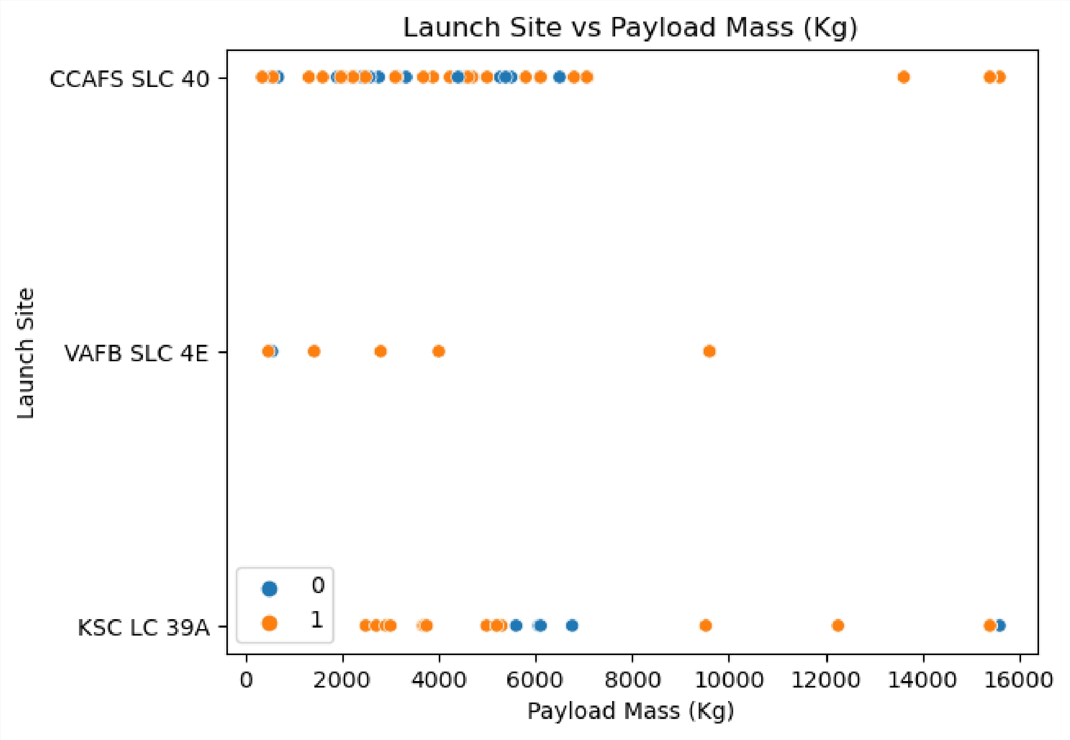
Flight Number vs. Launch Site

- SpaceX mostly launched from CCAFS SLC 40 at the start, with low mission success rate
- With every launch, SpaceX has increased the running average of its mission success rate
- Most launches are from CCAFS SLC 40



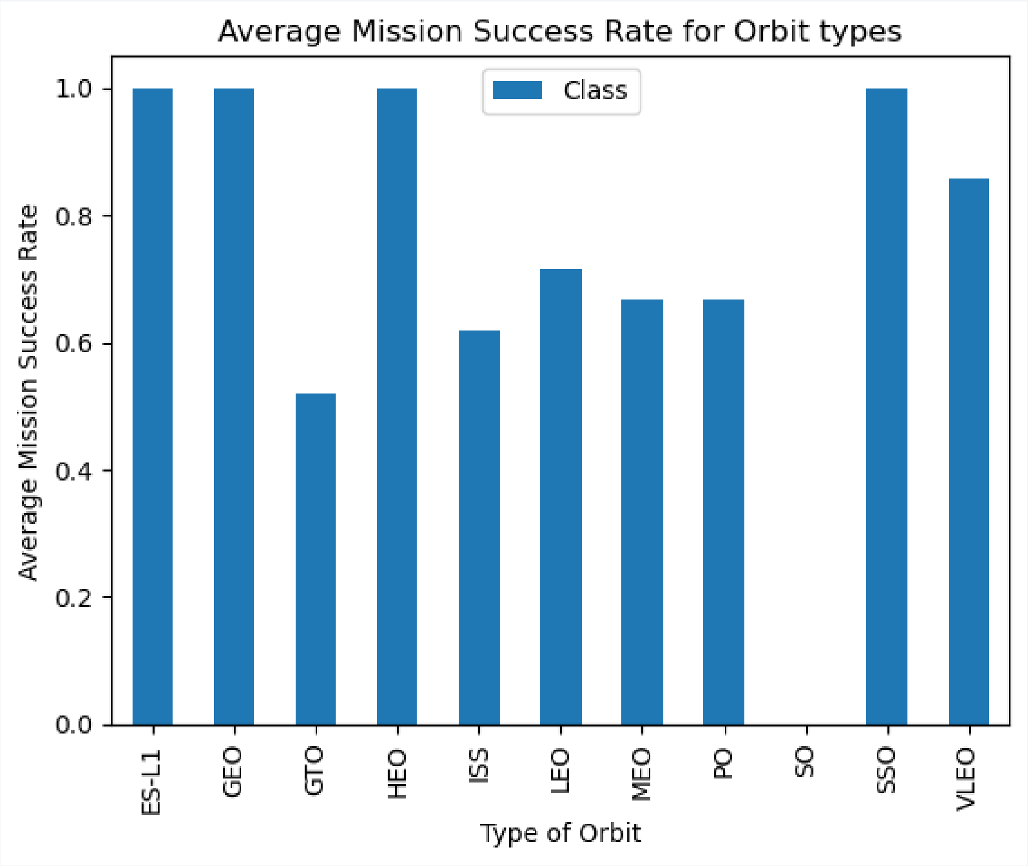
Payload vs. Launch Site

- Launch site vs Payload Mass does not seem very indicative of mission success rate
- VAFB SLC 4E launches lower Payload Mass on average



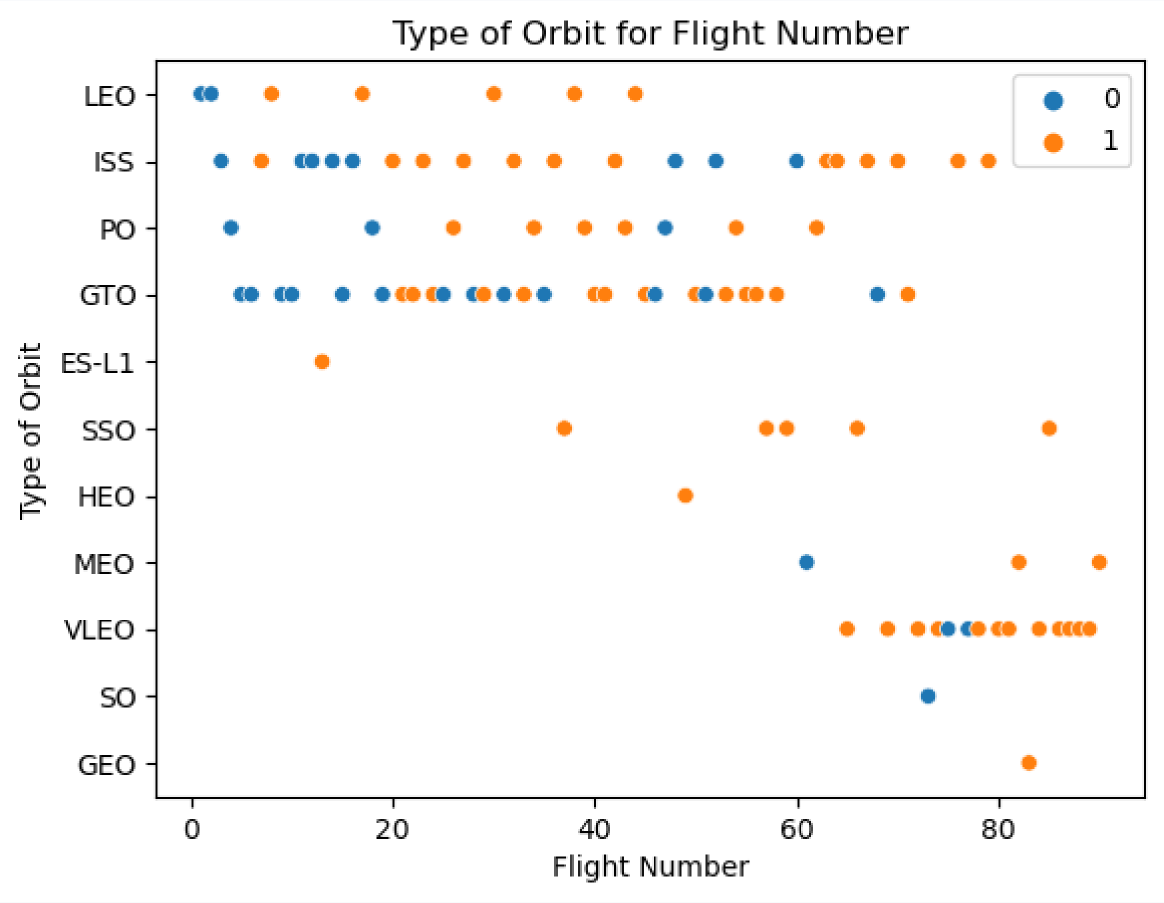
Success Rate vs. Orbit Type

- For four different Orbits the mission success rate has been 100% (ES-L1, GEO, HEO, SSO)



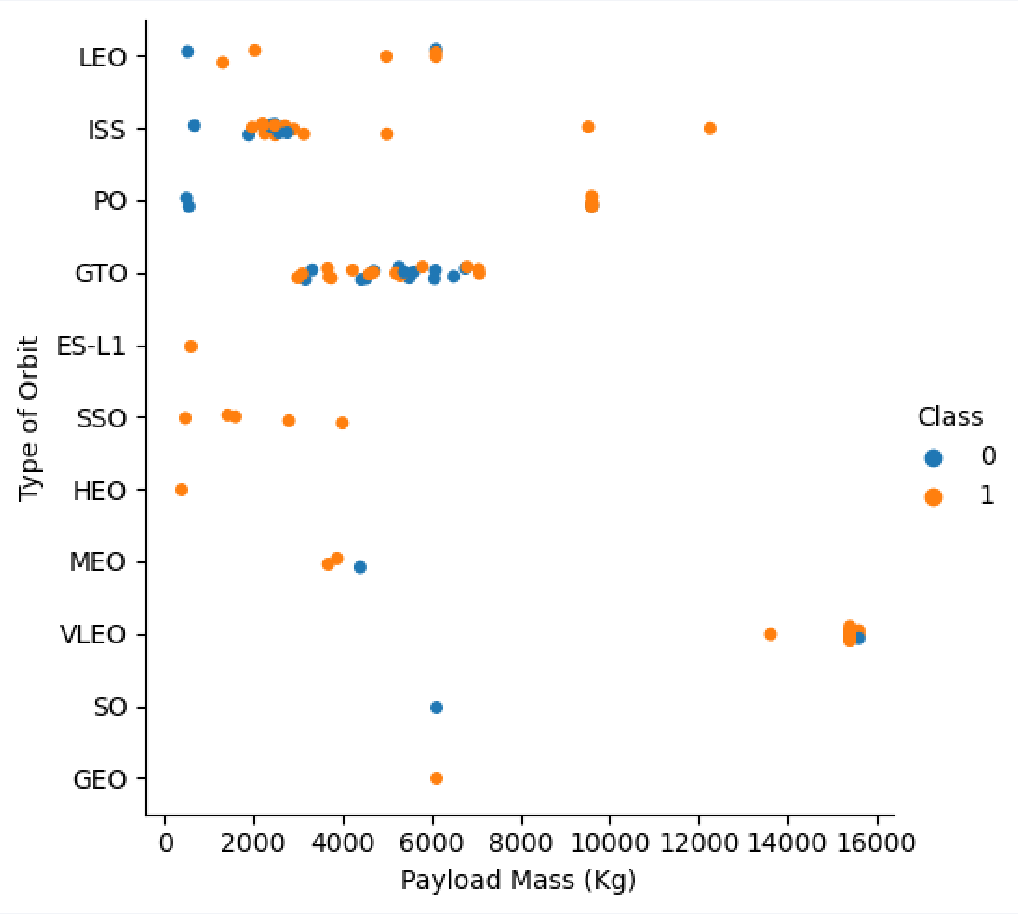
Flight Number vs. Orbit Type

- With greater flight number, the success rates increases for most types of Orbits
- The type of mission that SpaceX flies (i.e. target Orbit) has been changing from LEO to VLEO with flight number



Payload vs. Orbit Type

- The highest payloads are all sent to VLEO
- There is a strong correleation between high payload mass and mission success



Launch Success Yearly Trend

- From 2010-13 SpaceX was unsuccessful in their missions
- Since 2017, the mission success rate has been climbing to around 90%

All Launch Site Names

- The DISTINCT command gives us all the unique launch sites

Task 1

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

```
%sql SELECT DISTINCT("Launch_Site") FROM SPACEXTABLE
```

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

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

Launch Site Names Begin with 'CCA'

- The wildcard operator % with keyword LIKE lets us find all sites beginning with CCA

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

```
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE "CCA%" LIMIT 50;
```

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

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- We can filter the Costumer with WHERE and then sum the total payload with SUM

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

```
%sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTABLE WHERE "Customer" == "NASA (CRS)" GROUP BY "Customer";
```

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

SUM("PAYLOAD_MASS__KG_")
45596

Average Payload Mass by F9 v1.1

- We filter with WHERE and average the resulting table

Task 4

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

```
%sql SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTABLE WHERE "Booster_Version" LIKE "%F9 v1.1%";
```

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

AVG("PAYLOAD_MASS__KG_")
2534.6666666666665

First Successful Ground Landing Date

- The minimum of the filtered outcomes gives us the first day that SpaceX managed to land their rocket on the launch pad

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

Hint:Use min function

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

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

MIN("Date")
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Between allows us to filter the table to a range of numeric values

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

```
%%sql
SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" == "Success (drone ship)"
AND "PAYLOAD_MASS__KG_" BETWEEN 4000 AND 6000 LIMIT 2;
```

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

Booster_Version
F9 FT B1022
F9 FT B1026

Total Number of Successful and Failure Mission Outcomes

- We find four different mission outcomes and do some extra work to verify our findings

List the total number of successful and failure mission outcomes

```
%sql SELECT DISTINCT("Mission_Outcome") FROM SPACEXTABLE;
```

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

Mission_Outcome
Success
Failure (in flight)
Success (payload status unclear)
Success

```
%sql SELECT COUNT("Mission_Outcome") FROM SPACEXTABLE;
```

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

COUNT(Mission_Outcome)
101

```
%%sql SELECT
        (SELECT COUNT("Mission_Outcome") FROM SPACEXTABLE WHERE "Mission_Outcome" in
         ("Success", "Success (payload status unclear)", "Success ")) as successes,
        (SELECT COUNT("Mission_Outcome") FROM SPACEXTABLE WHERE "Mission_Outcome" in
         ("Failure (in flight)")) as failures
    FROM SPACEXTABLE LIMIT 1;
```

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

successes	failures
100	1

Boosters Carried Maximum Payload

- We filter the booster versions with a subquery and show only those, that have equal payload mass to the maximum

Task 8

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

```
%%sql SELECT "Booster_Version", "PAYLOAD_MASS__KG_" FROM SPACEXTABLE WHERE
"PAYLOAD_MASS__KG_" == (select max("PAYLOAD_MASS__KG_") from SPACEXTABLE);
```

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

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

- We filter the results with multiple conditions

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.

```
%%sql
SELECT
    strftime('%m', "Date") AS Month,
    "Booster_Version",
    "Launch_Site",
    "Landing_Outcome",
    "Date"
FROM
    "SPACEXTABLE"
WHERE
    strftime('%Y', "Date") == "2015" AND
    "Landing_Outcome" == 'Failure (drone ship)'
ORDER BY
    Month;
```

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

Month	Booster_Version	Launch_Site	Landing_Outcome	Date
04	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)	2015-04-14
10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)	2015-10-01

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

- We filter outcomes by date and display outcome and frequency of outcome in a new table

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.

```
%%sql
SELECT
    "Landing_Outcome",
    COUNT(*) as Outcome_Count
FROM
    "SPACEXTABLE"
WHERE
    "Date" BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY
    "Landing_Outcome"
ORDER BY
    Outcome_Count DESC;
```

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

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

Section 3

Launch Sites Proximities Analysis

All launch sites' location on a global map

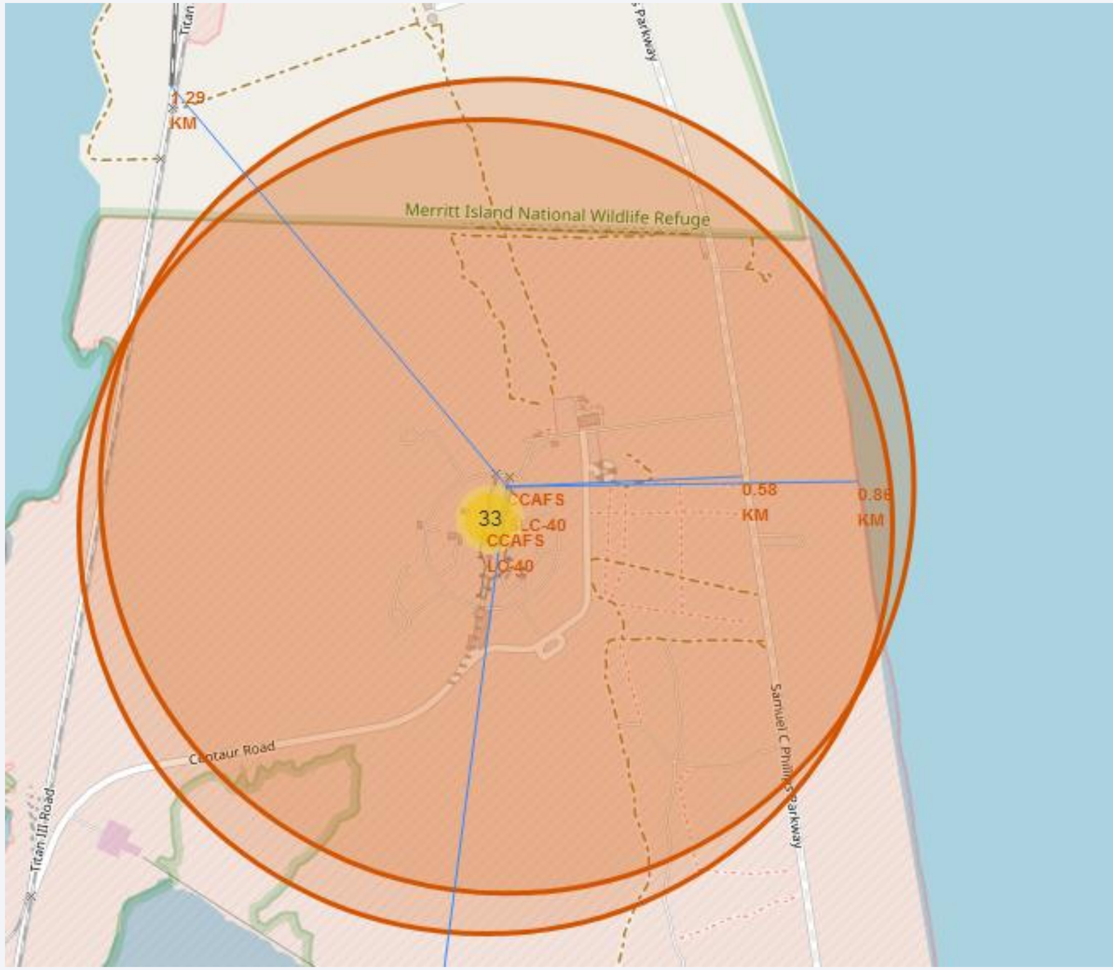
SpaceX bases are all located in the US, close to the ocean and close to the equator

Cluster maps labeled by mission success

- Launch outcomes per base, color labeled with green (success) and red (failure)

Distance of Space base to highway, railway and ocean

- Important infrastructure are nearby (within 1 mile or under 1.6km) and the coastline is under 1km away.

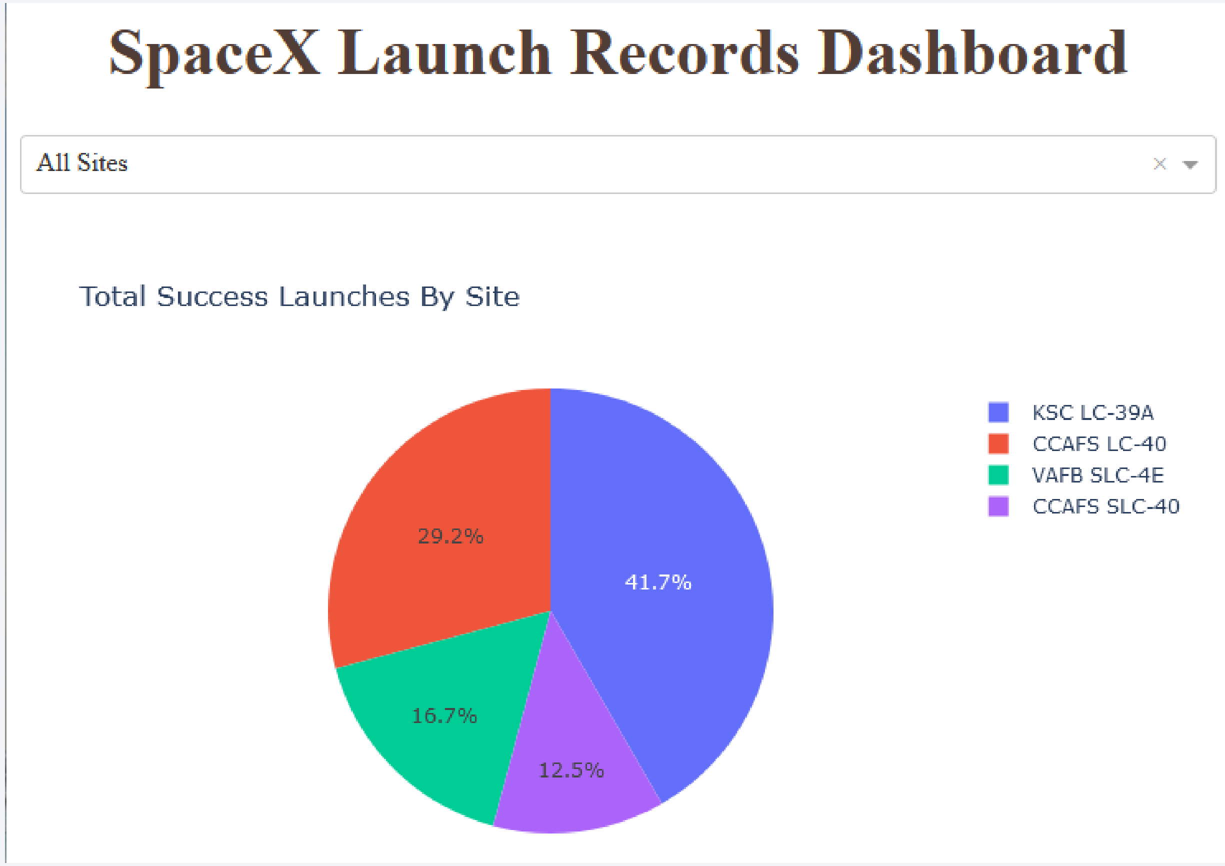


Section 4

Build a Dashboard with Plotly Dash

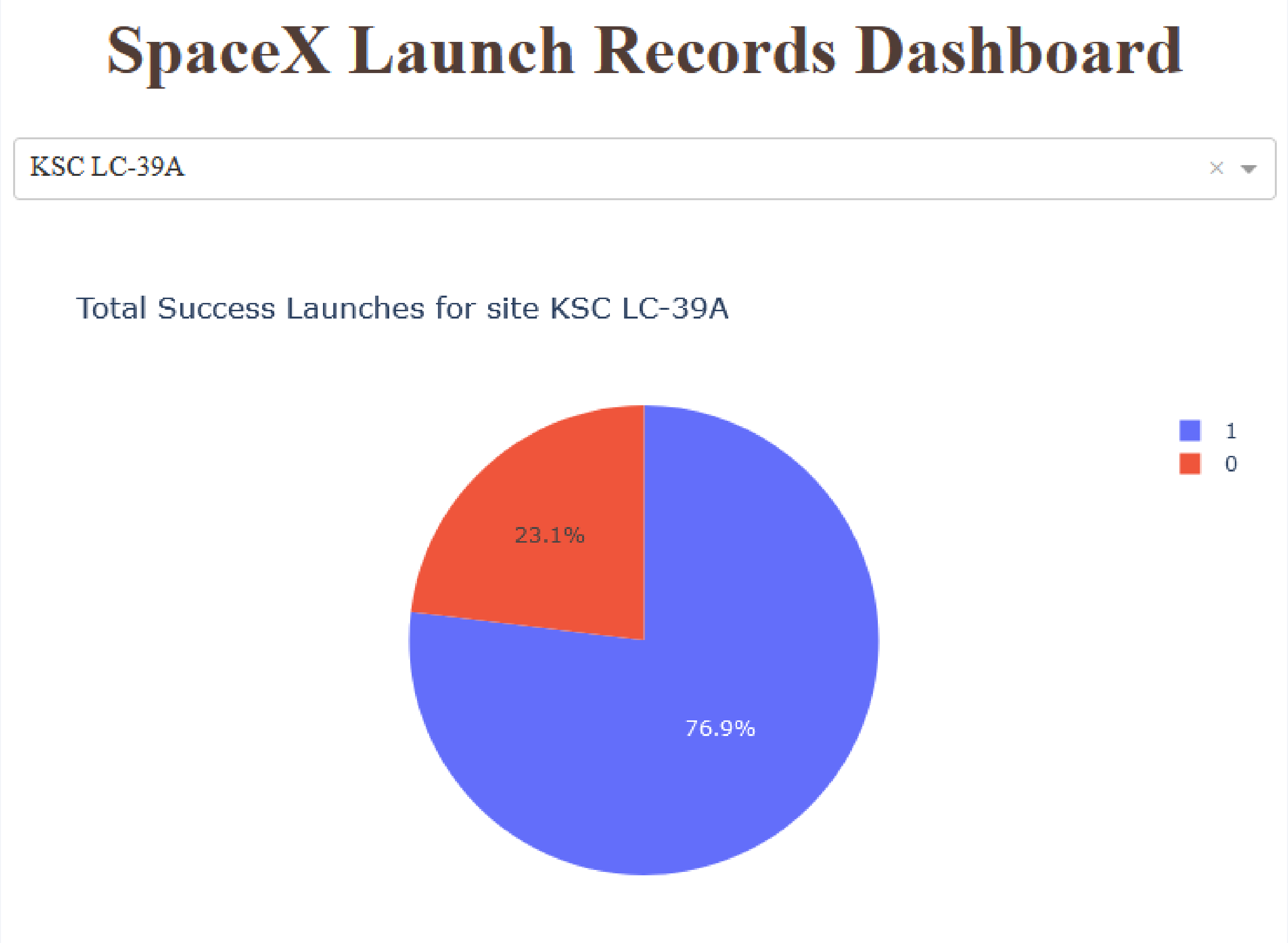
SpaceX launch record for all bases

- SpaceX has the highest number of successful launches from KSC LC-39A, and the least from CCAFS SLC-40



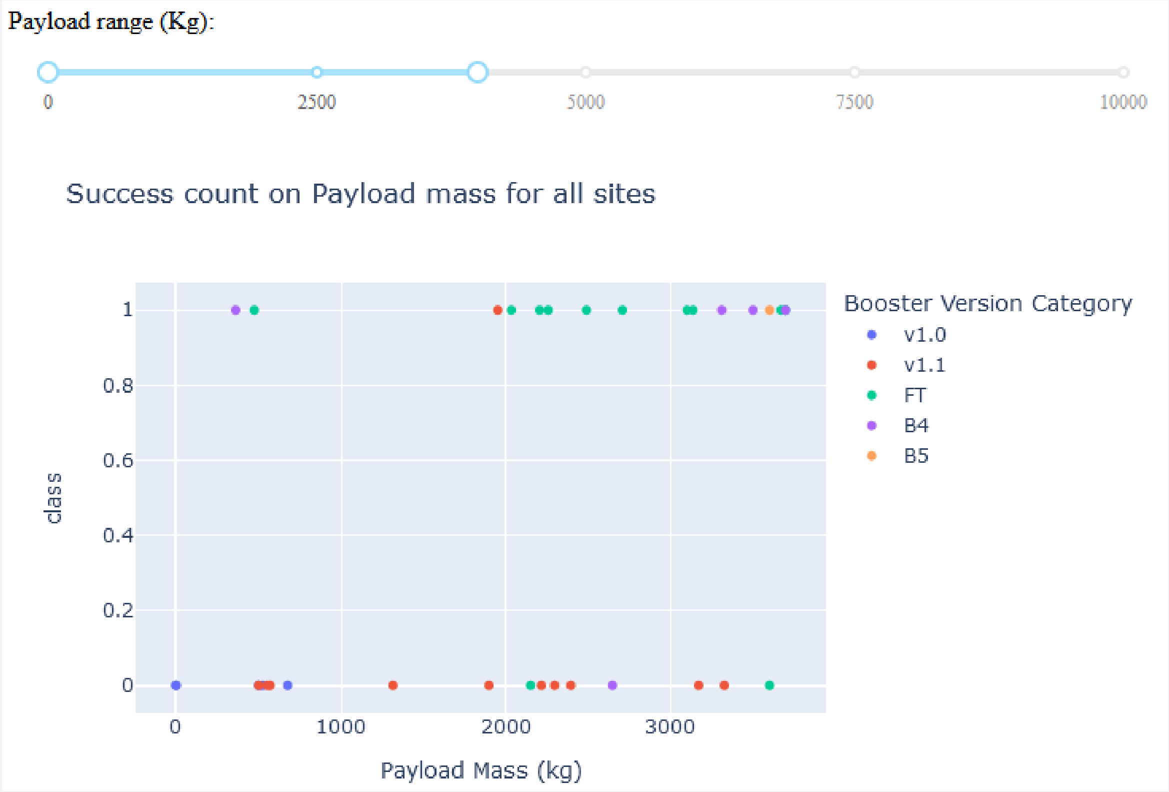
<Dashboard Screenshot 2>

- Indeed, also by relative number, the site KSC LC-39A is the most successful launch site for spaceX



<Dashboard Screenshot 3>

- For launches at low payload mass, the ratio of successful launches to failed launches is close to 50-50



Section 5

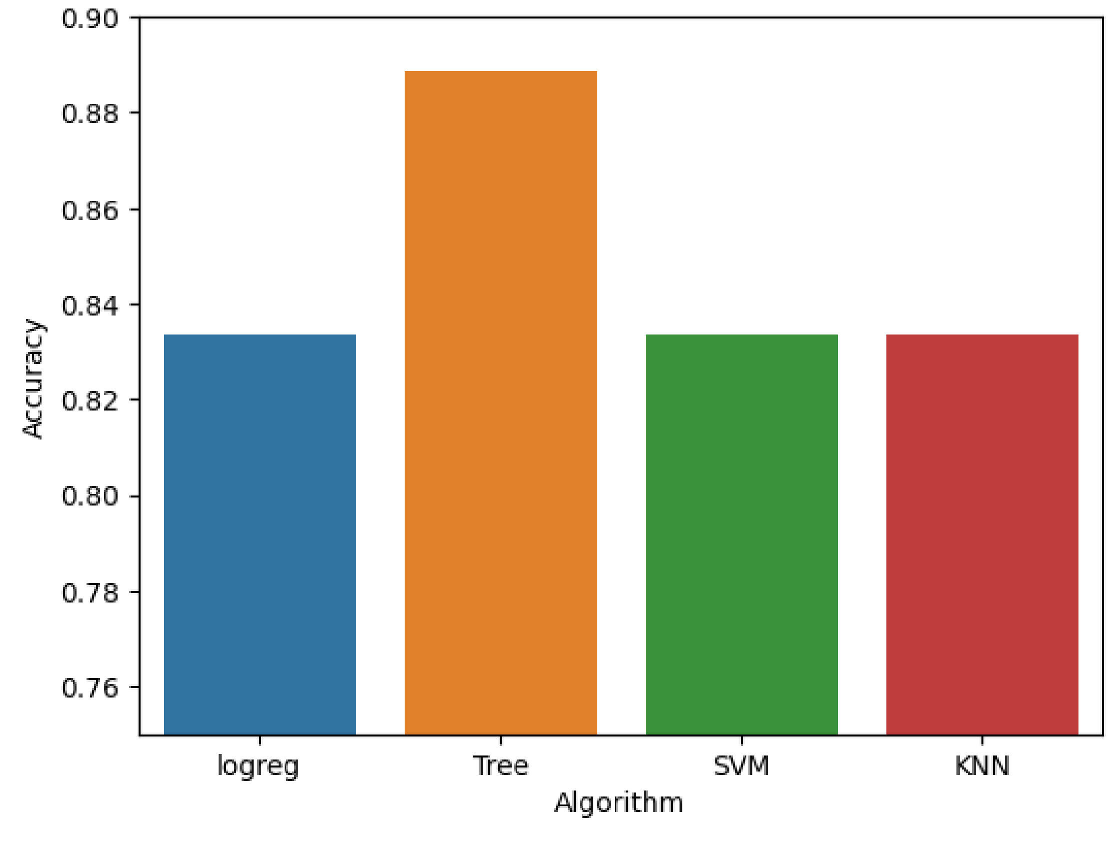
Predictive Analysis (Classification)

Classification Accuracy

- Decision trees have the best accuracy

```
x = ['logreg', 'Tree', 'SVM', 'KNN']
y = [logreg_cv.score(X_test, Y_test), tree_cv.score(X_test, Y_test),
     svm_cv.score(X_test, Y_test), knn_cv.score(X_test, Y_test)]
print(logreg_cv.score(X_test, Y_test), tree_cv.score(X_test, Y_test),
     svm_cv.score(X_test, Y_test), knn_cv.score(X_test, Y_test))
fig = sns.barplot(x = x, y=y)
fig.set_xlabel('Algorithm')
fig.set_ylabel('Accuracy')
fig.set_ylim([0.75, 0.9])
```

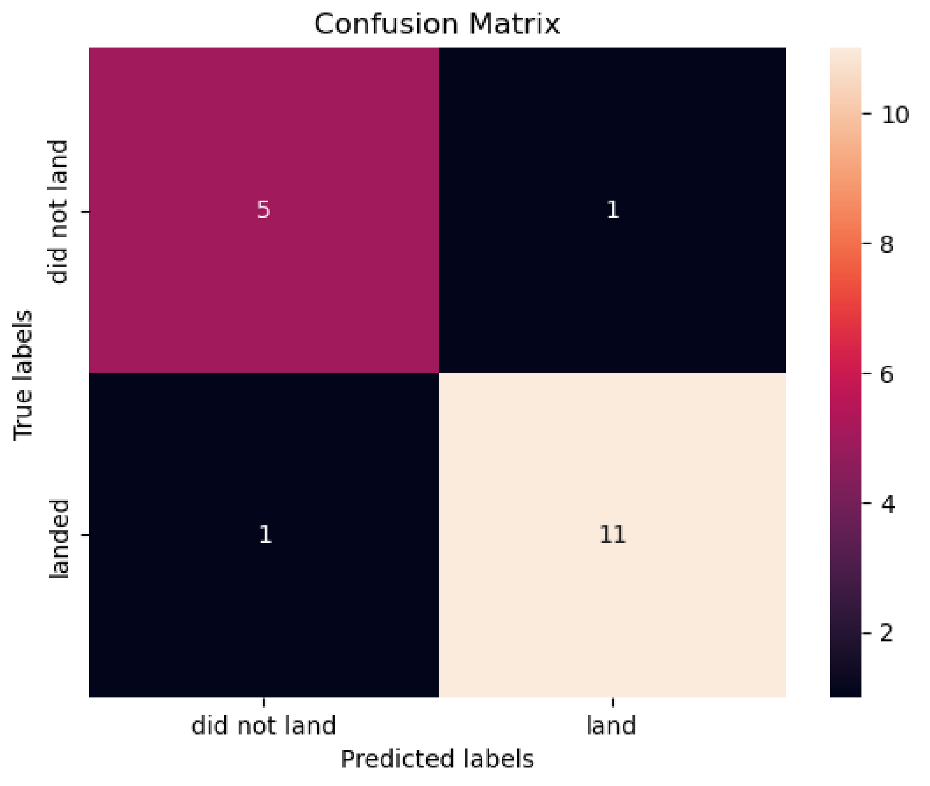
```
0.8333333333333334 0.8888888888888888 0.8333333333333334 0.8333333333333334
```



Confusion Matrix

- Decision trees work the best by far and achieve an accuracy of 16/18 (close to 90%).
- They predicted one false positive (top right) and one false negative (bottom left)

```
In [111]:  yhat = tree_cv.predict(X_test)
           plot_confusion_matrix(Y_test,yhat)
```



Conclusions

- SpaceX is getting progressively better at their missions, e.g. the higher the mission number the higher the mission success chance
- Launch success rate has been increasing steadily from 2013-2020
- KSC LC-39A is their most successful launch site
- The decision tree classifier is the best machine learning algorithm for this task

Thank you!