IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Federico Dignani
24 November 2022

Outline

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

Executive Summary

- Methodologies
  - Data Collection through API
  - Data Collection through web scrapping
  - Data Wrangling
  - Exploratory Data Analysis (EDA) with SQL
  - EDA with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Results
  - Exploratory Data Analysis results
  - Interactive Analytics results
  - Predictive Analysis

Introduction

- Project background and context

  Space X 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 Space X 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 space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Questions we want to answer
  - What factors determine if the rocket will launch successfully?
  - What feature interaction determine the success rate of a successful landing?
  - What conditions are needed to increase the probability of successful landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX public API and web scrapped from Wikipedia
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using machine learning models
  - How to build, tune, evaluate classification models

Data Collection

- Datasets came from two sources:
  - Source #1 was the SpaceX API, publicly available and obtained via get/request method.
  - Next, I decoded the response as a Json and turn it into a dataframe using Panda's .json_normalize() built-in function.
  - Data cleaning was made, looking for missing values and filling them where necessary.
  - Source #2 was web scrapping from Wikipedia for Falcon 9 historical launch records, using BeautifulSoup library.
  - The goal was to extract the records as an HTML table, parse the table, and convert it into a dataframe for data analysis.

Data Collection – SpaceX API

- There's a five-step process when getting the dataset from an API, looks like this:

1. Get response from API

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

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

2. Convert response to a .json file

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

3. Apply custom functions to clean data

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

4. Assign list to dictionary, then dataframe.

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

Then, we need to create a Pandas data frame from the dictionary launch_dict.

```
# Create a data from launch_dict
data2 = pd.DataFrame(launch_dict)
```

5. Filter dataframe and export to a .csv file

```
data_falcon9.to_csv('dataset_part\_1.csv', index=False)
```

Full link to the notebook [here]

Data Collection - Scraping

1. Request the Falcon9 Launch Wiki page from url

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

html_data = requests.get(static_url)
html_data.status_code
```

2. Create a BeautifulSoup object from the HTML response

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text, 'html.parser')
```

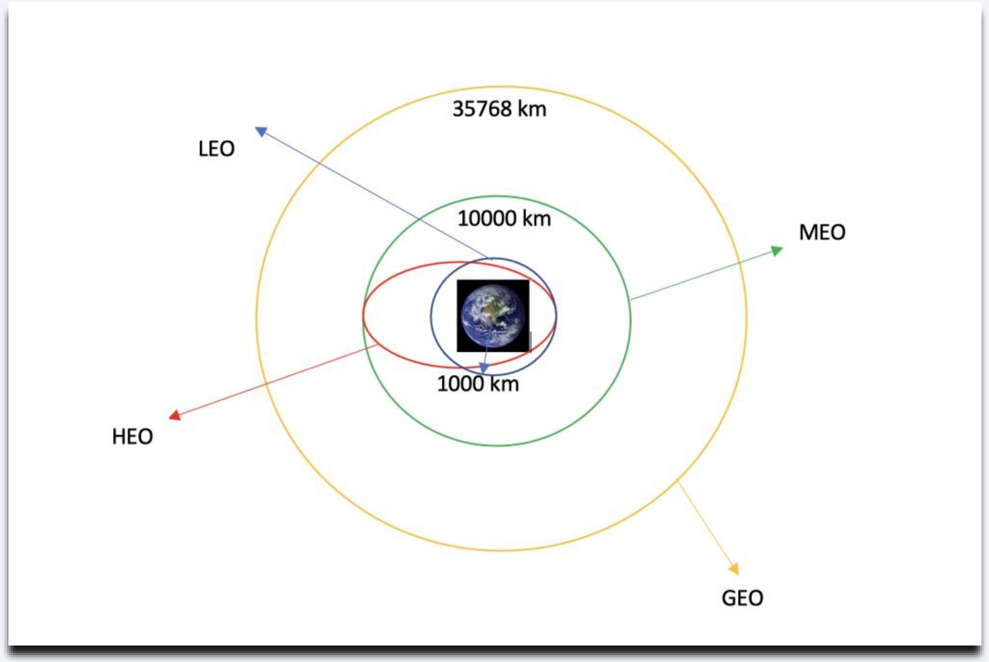
3. Extract all column/variable names from the HTML header

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            launch_dict['Flight No.'].append(flight_number)
            #print(flight_number)
            datatimelist=date_time(row[0])
```

Full link to the notebook here

Data Wrangling

- Number of launches at each site, number and occurrence of each orbit were calculated.
- Landing outcome variable was created from Outcome column and worked out success rate for every landing in dataset.
- Handle missing values.



Full link to the notebook here

EDA with Data Visualization

- Three types of charts were used during EDA:

Scatterplot: show how much a variable is affected by another, useful to find correlation between two variables.

Bar graph: easy to read, bar graphs plot relationships between a categorical and a numerical value. They can also show big changes in data over time.

Line plot: ideal to see change over time periods. Show data variables and trends very clearly.

Full link to the notebook [here]

EDA with SQL

- Using SQL, the following queries were performed:
  - Name of the unique launch sites in the space mission
  - 5 records where launch sites begin with the string ‘CCA’
  - Total payload mass carried by boosters launched by NASA (CRS)
  - Average payload mass carried by booster version F9 v1.1
  - Date where the successful landing outcome in drone ship was achieved.
  - Names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000
  - Total number of successful and failure mission outcomes
  - Names of the booster_versions which have carried the maximum payload mass
  - Records which will display the month names, successful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017
  - Ranking the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

Full link to the notebook [here]()

Build an Interactive Map with Folium

- Objects representing each launch site were created in the map with Latitude and Longitude information present in the dataset. Red and green markers were added for each failed/successful launch outcome.

- Lines were drawn calculating distance between launch sites and railways, highways, coastlines, cities, etc. Questions were asked, is there any reason they are far or near this sites?

Full link to the notebook here

Build a Dashboard with Plotly Dash

- An interactive dashboard was built with Plotly Dash

- Every launch site has a pie chart showing successful/failed launchs

- A scatter plot showing the relationship with Outcome and Payload Mass (Kg) for the different booster version

Full link to the notebook [here]

Predictive Analysis (Classification)

- Data was loaded using Numpy and Pandas, transformed, and splitted into training and testing set for proper evaluation.
- Six different Machine Learning models were tested, using accuracy as the evaluation metric
- Hyperparameter tuning using GridSearchCV was performed to boost the models' performance.

Full link to the notebook here

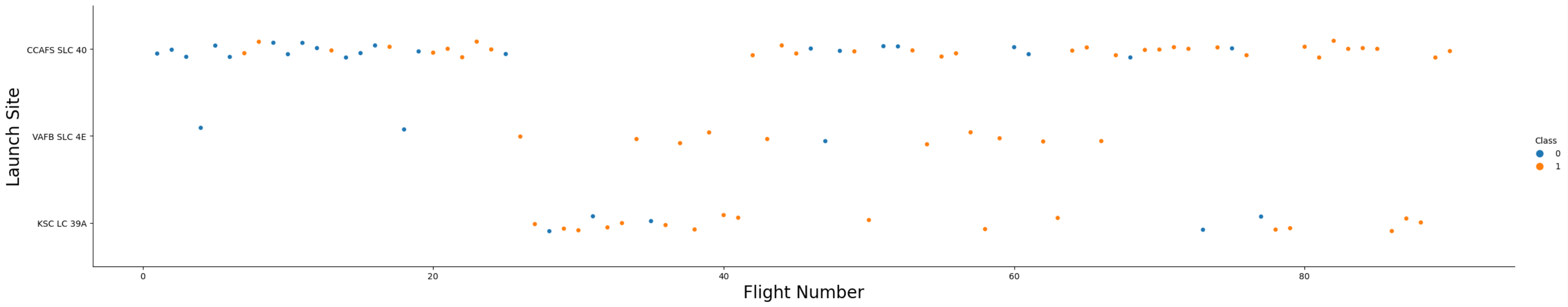
Results

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

Section 2

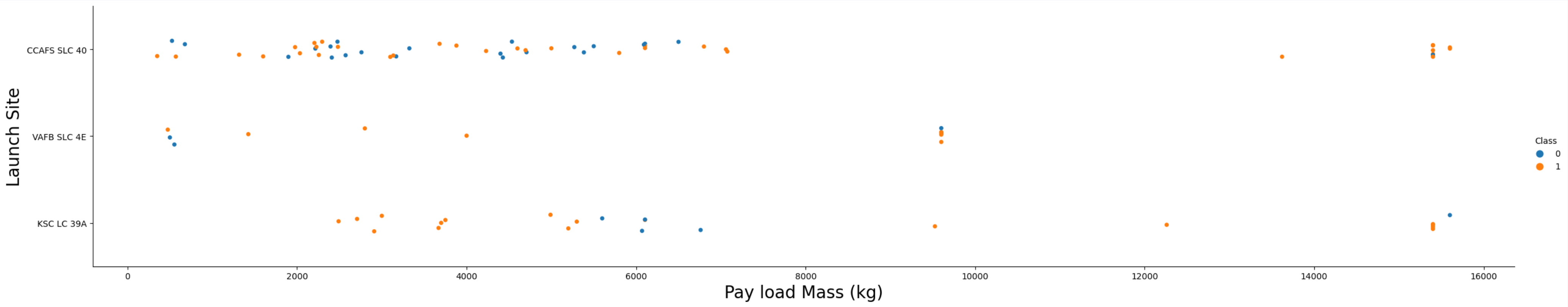
Insights drawn from EDA

Flight Number vs. Launch Site



- First flights were launched mainly from CCAFS LC-40, with a low amount of success. After flight #40 is the predominant launch site and with high success rate.
- VAFB SLC 4E is the less used Launch Site, despite a high success rate.
- From flights 25 till 40, KSC LC-39A took CCAFS LC-40 place, with better overall performance. Onwards, it was a secondary launch site. Probably used when the former is under maintenance.

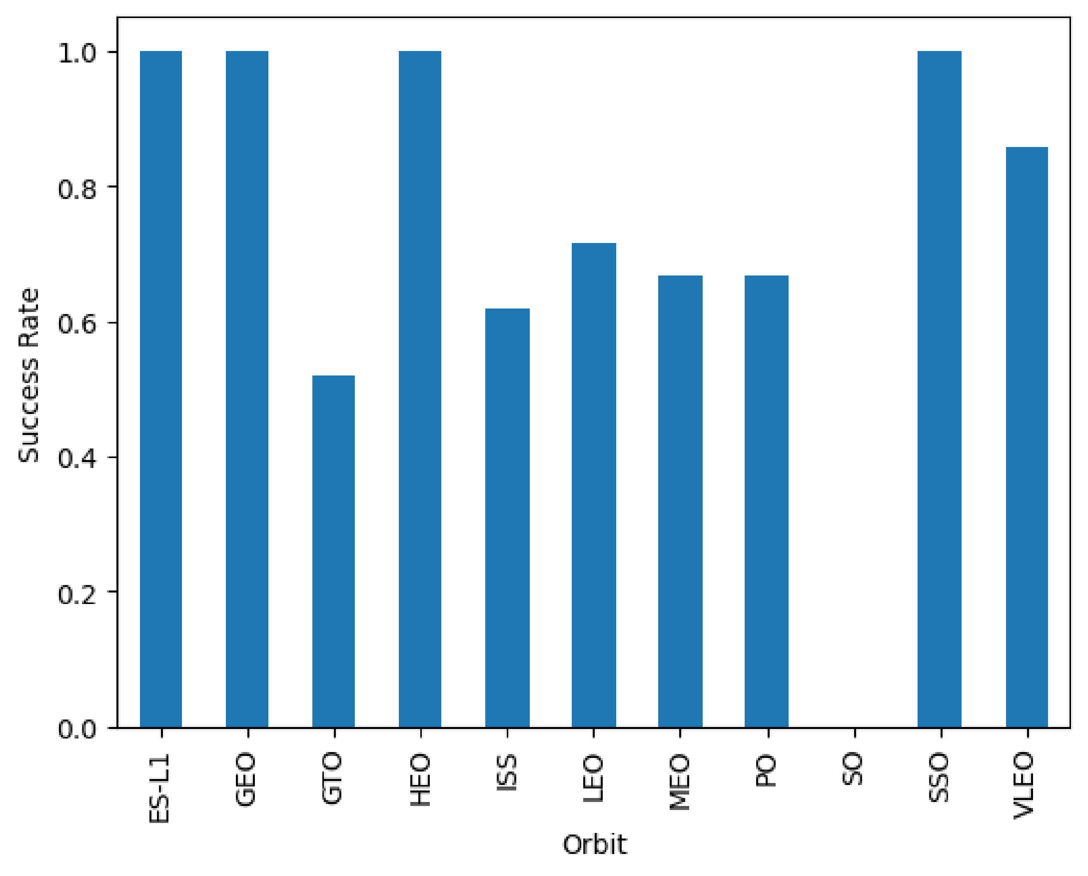
Payload vs. Launch Site



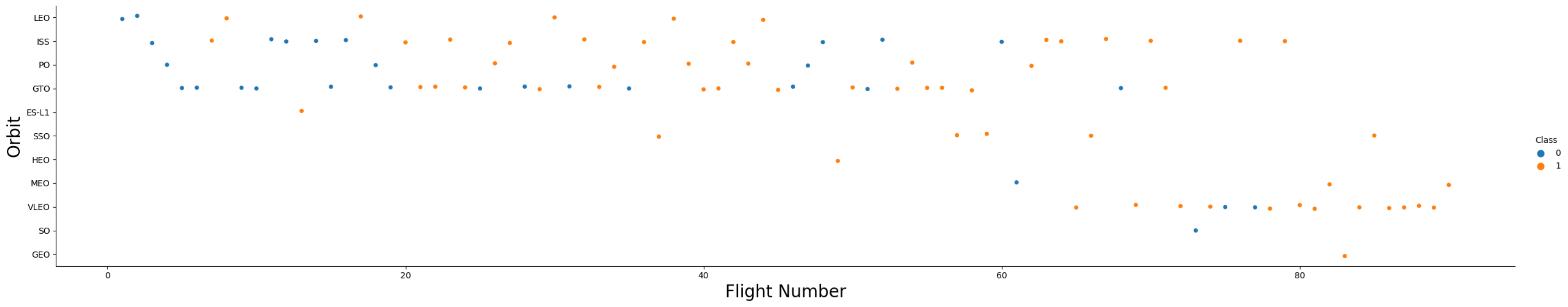
- For the VAFB-SLC Launch Site there are no rockets launched for heavy payload mass (greater than 10000)

Success Rate vs. Orbit Type

- ES-L1, GEO, HEO and SSO orbits have 100% success rate.
- VLEO +80% success rate.

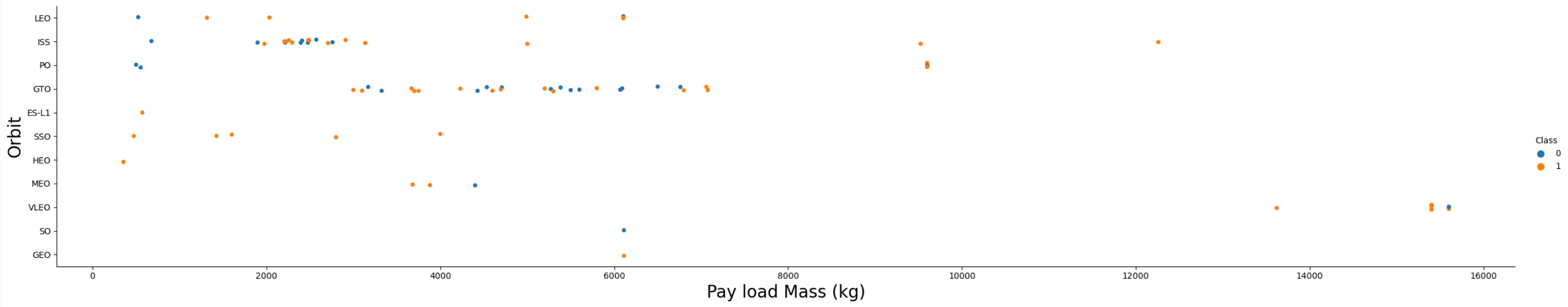


Flight Number vs. Orbit Type



- In the LEO Orbit, Success appears related to the Flight Number.
- There seems to be no relationship between Flight Number when in GTO orbit.

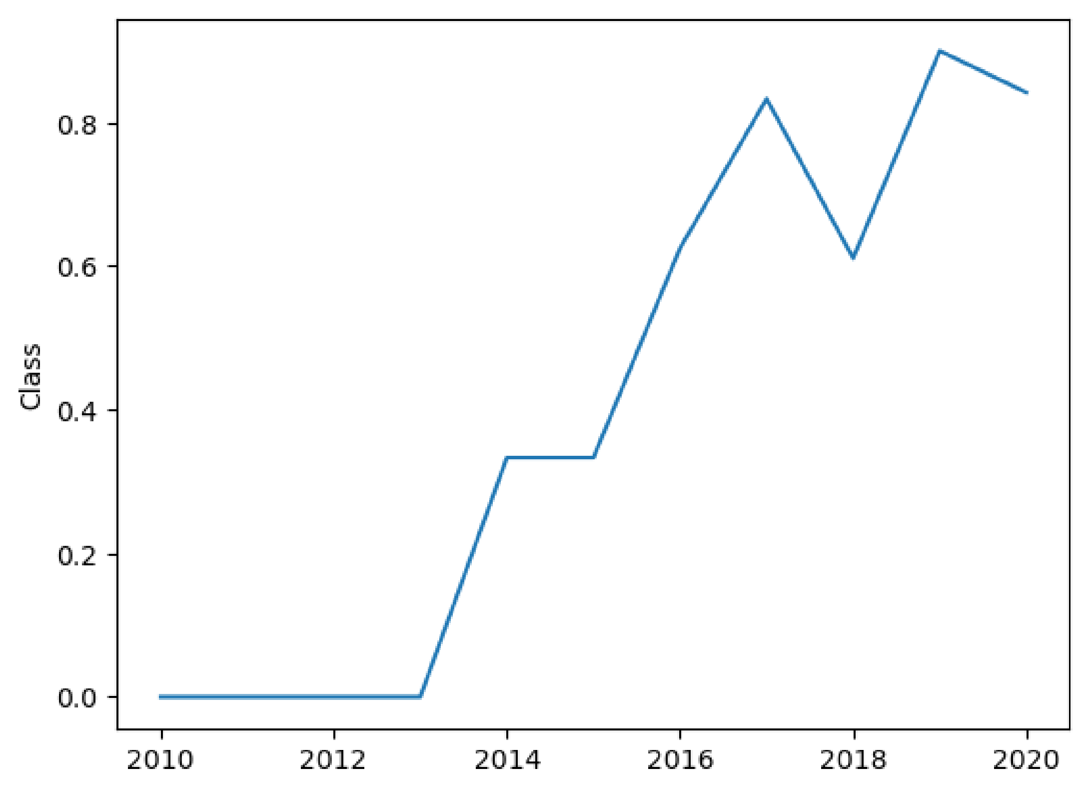
Payload vs. Orbit Type



- With heavy payloads the successful landing rate are higher for Polar, LEO and ISS orbits.
- 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

- Success rate since 2013 kept increasing till 2020
- In 2017 established in the +80% level, with a dropdown in 2018



All Launch Site Names

- We used the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

```
In [4]: %sql select distinct LAUNCH_SITE from SPACEX;

 * ibm_db_sa://cjr01249:***@98538591-7217-4024-b
Done.
```

Out[4]:

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

Launch Site Names Begin with 'CCA'

- Using the expression **LIMIT** 5 will only show top 5 records from SPACEX.
- **LIKE** keyword has a wild card with the letters 'CCA%', the percentage in the end suggests that the LAUNCH_SITE name must start with CCA.

In [5]:

```
%sql select * from SPACEX where LAUNCH_SITE like 'CCA%' limit 5
```

```
 * ibm_db_sa://cjr01249:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98
Done.
```

Out[5]:

DATE	time__utc_	booster_version	launch_site	payload	payl
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	

Total Payload Mass

- Using the function **SUM** aggregates the total in the column PAYLOAD_MASS_KG_

- The **WHERE** clause filters the dataset to only perform calculations on Customer NASA (CRS)

```
In [6]: %sql select sum(payload_mass__kg_) as TPM_NASA_CRS FROM SPACEX where customer = 'NASA (CRS)'

 * ibm_db_sa://cjr01249:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databas
Done.

Out[6]: tpm_nasa_crs
          45596
```

Average Payload Mass by F9 v1.1

- Using the function **AVG** works out the average in the column PAYLOAD_MASS_KG_

- The **WHERE** clause filters the dataset to only perform calculations on booster_version F9 v1.1

```
%sql select avg(payload_mass__kg_) as AVG_PM_F9_v11 FROM SPACEX where booster_version LIKE 'F9 v1.1%'
```

```
 * ibm_db_sa://cjr01249:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdoma
Done.
```

avg_pm_f9_v11
2534

First Successful Ground Landing Date

- Using the function **MIN** calculates the minimum date in the column DATE
- The **WHERE** clause filters the dataset to only perform calculations on Landing_Outcome Success (ground pad)

```
%sql select min(DATE) from SPACEX where landing__outcome = 'Success (ground pad)'

 * ibm_db_sa://cjr01249:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u
Done.

         1
2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

In [9]:
```
%sql select payload as name_of_booster from SPACEX where landing__outcome = 'Success (drone ship)' and payload_mass__kg_ BETWEEN 4000 and 6000
```

```
 * ibm_db_sa://cjr01249:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[9]:

name_of_booster
JCSAT-14
JCSAT-16
SES-10
SES-11 / EchoStar 105

- We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

- **COUNT** function was used to count number of records in the dataset.

- **GROUP BY** clause was added to present the results grouped by the mission outcome (failure and success).

```
%sql select count(*) AS total_number, mission_outcome from SPACEX group by mission_outcome
```

```
 * ibm_db_sa://cjr01249:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databas
Done.
```

total_number	mission_outcome
1	Failure (in flight)
99	Success
1	Success (payload status unclear)

Boosters Carried Maximum Payload

```
%sql select booster_version as boosterversion from SPACEX where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEX);
```

boosterversion
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

- A sub-query was inside the **WHERE** clause, in this case indicating the **MAX** value of PAYLOAD_MASS__KG_

2015 Launch Records

```
%sql select DATE, landing__outcome, booster_version, launch_site from SPACEX where year(DATE) = 2015 and landing__outcome = 'Failure (drone ship)'
```

```
 * ibm_db_sa://cjr01249:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

DATE	landing__outcome	booster_version	launch_site
2015-01-10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015-04-14	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- We used a combinations of the **WHERE** and **AND** clauses to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015.

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

```
%sql SELECT count(landing__outcome), LANDING__OUTCOME from SPACEX WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' group by LANDING__OUTCOME order by
```

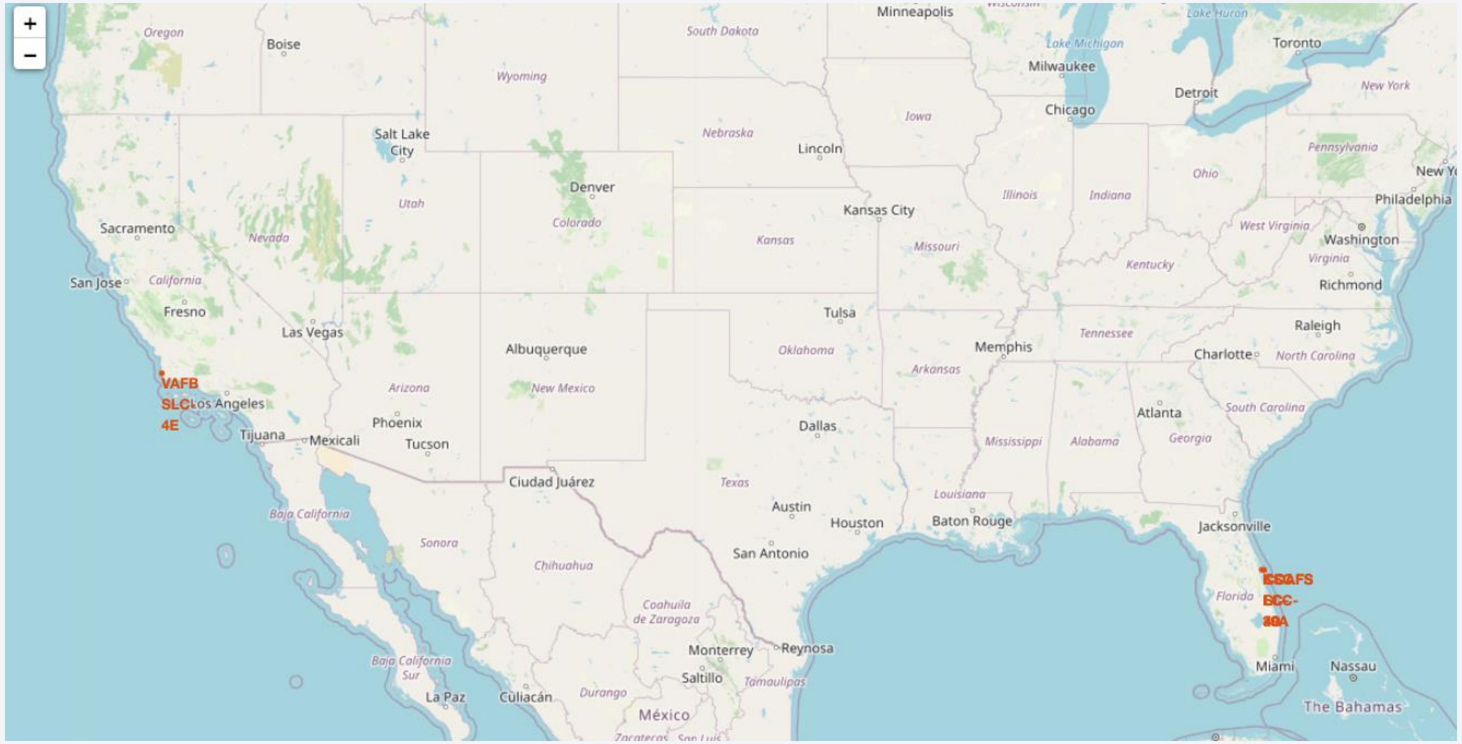
- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing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

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

Section 3

Launch Sites Proximities Analysis

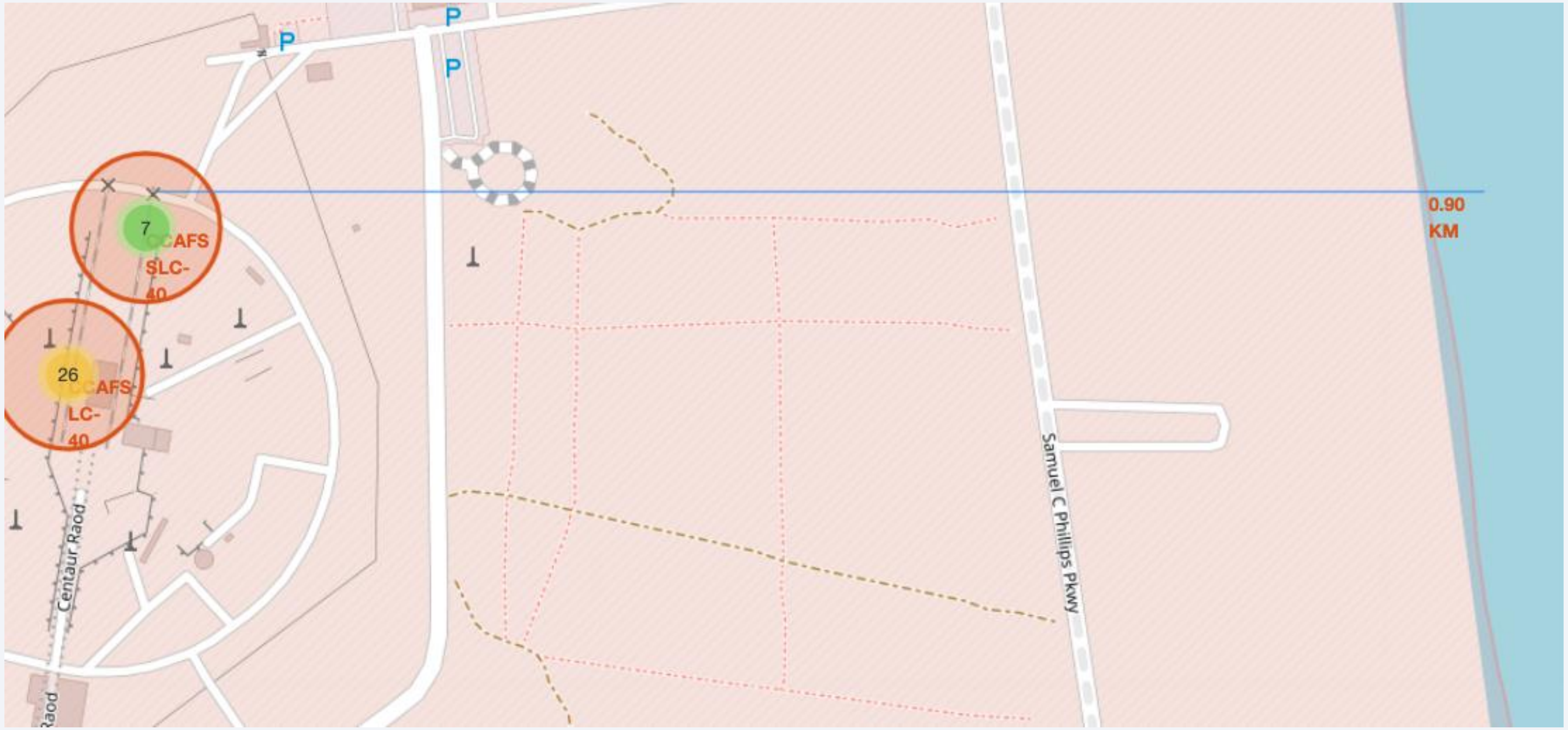
Launch Sites’ Location Markers



SpaceX Launch Sites are located in both US coasts, Florida and California.

Markers with color labels

Launch Site's distance to landmarks

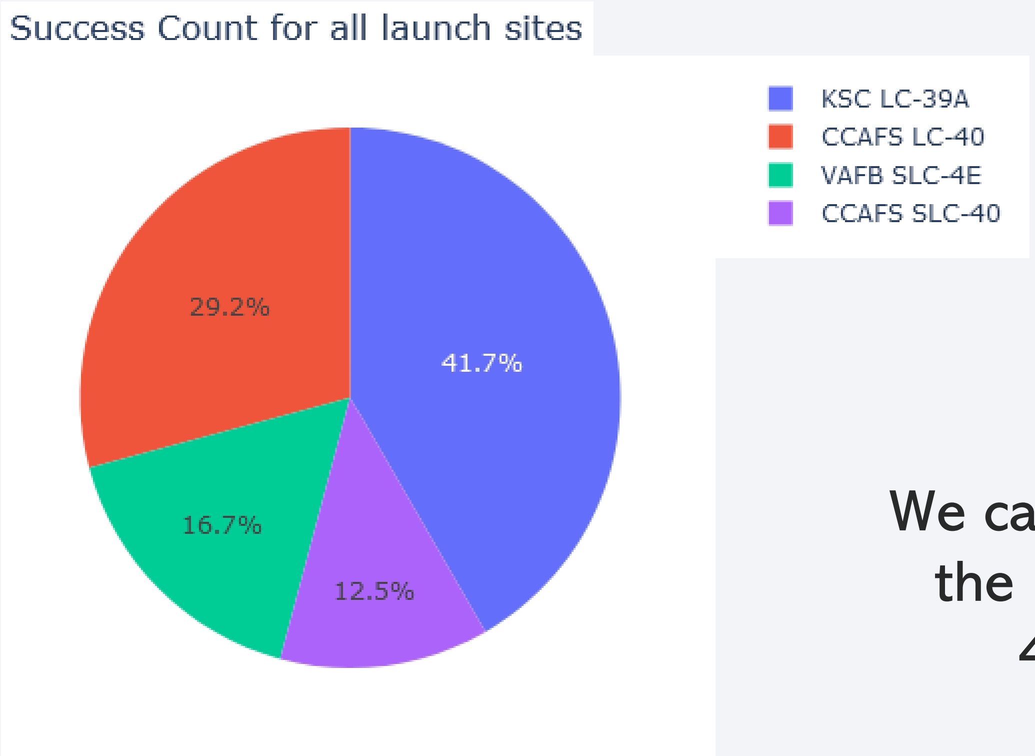


SpaceX Launch Sites are located near the ocean for safety reasons.

Section 4

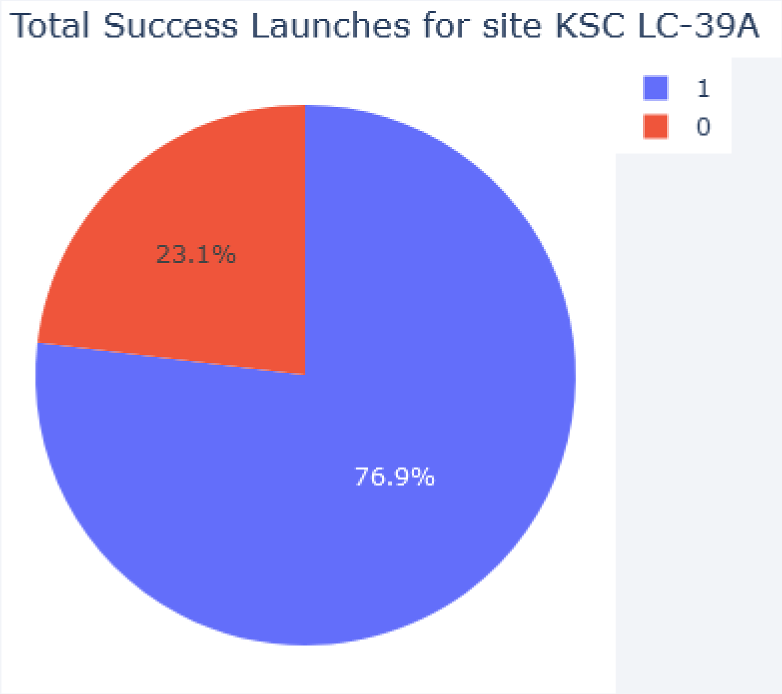
Build a Dashboard with Plotly Dash

Success percentage achieved per Launch Site



We can see clearly that **KSC LC-39A** has the highest success percentage with 41.7% of successful launches

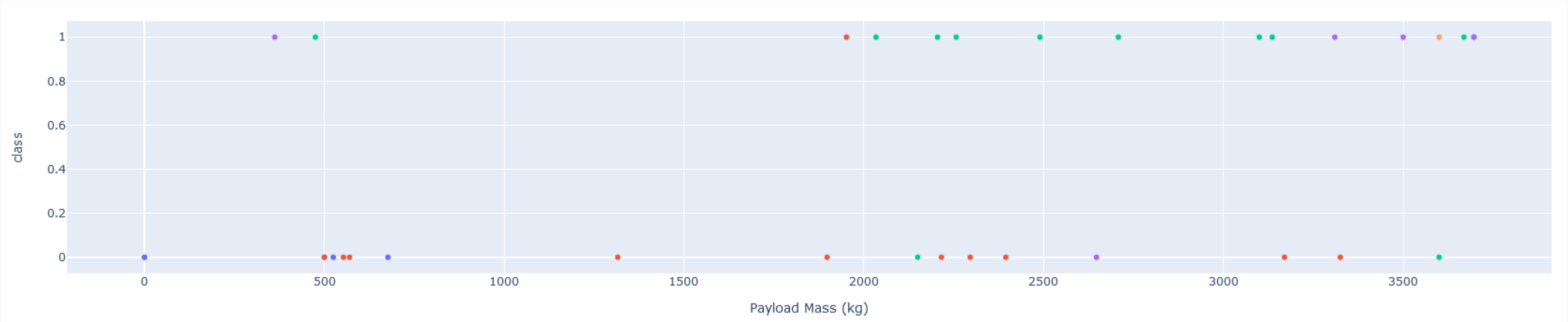
Launch Site with the highest launch success



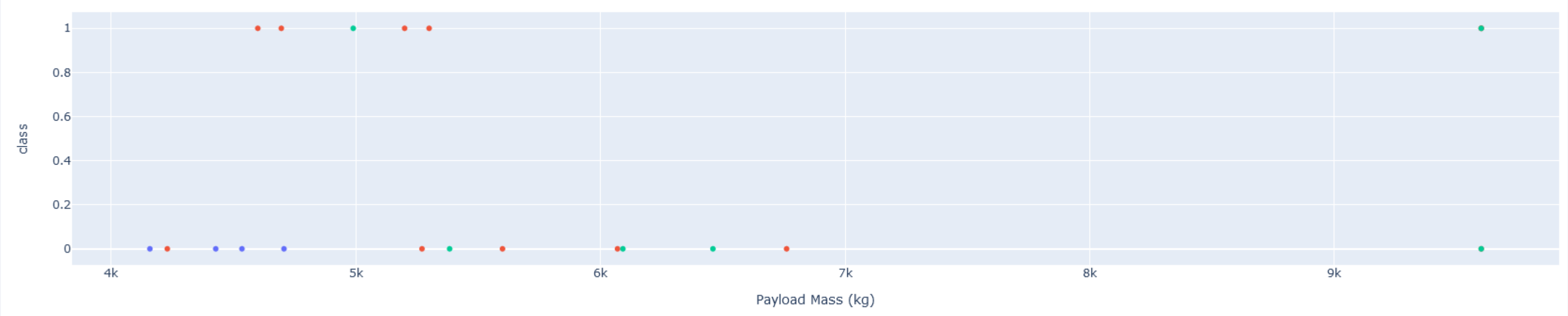
KSC LC-39A launch site shows a 76.9% of successful launches and a 23.1% failure rate

<Dashboard Screenshot 3>

Payload Mass 0-4000kg



Payload Mass 4000-10000kg



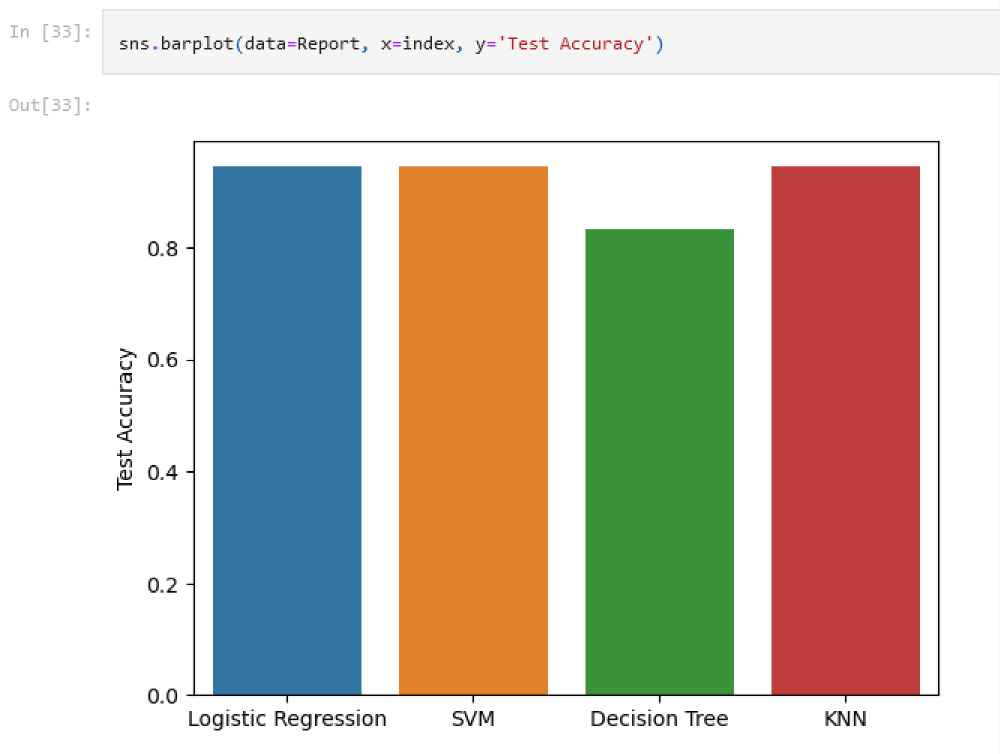
Success rate for low payload mass is significantly higher than high payload mass

Section 5

Predictive Analysis (Classification)

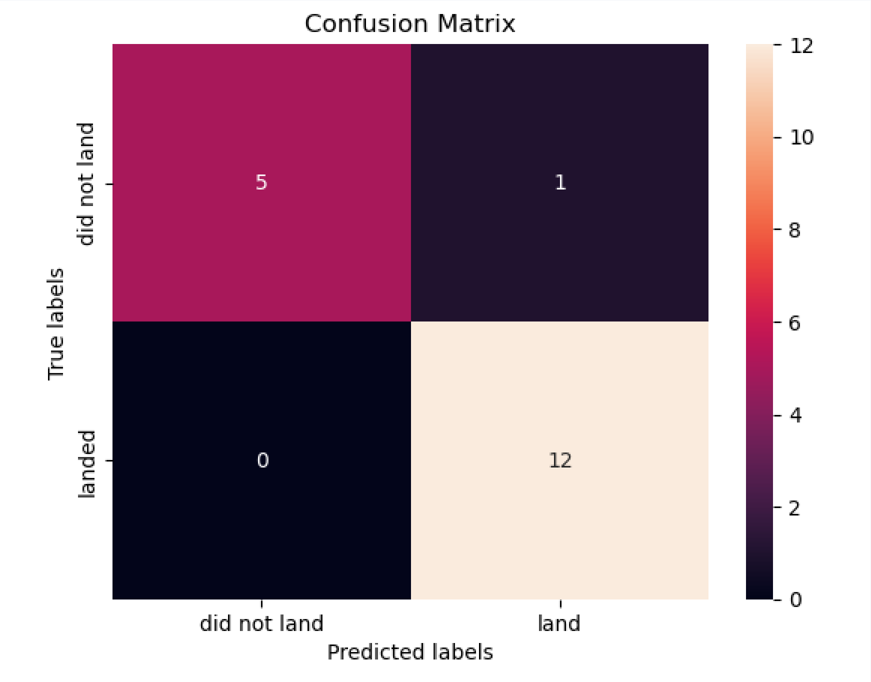
Classification Accuracy

- The final accuracy of the model in the test set, was 94% for 3 of the 4 models. Only Decision Tree showed a different performance of 83%.



Confusion Matrix

- All models show the same confusion matrix, with only one mis prediction, a False Positive. In the upper right we see predicted “landed”, that actually did not land.



Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site. They learn from the past.
- Launch success rate has been increasing since 2013.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the highest success rate.
- KSC LC-39A had the most successful launches of any sites.
- Success rate for low payload mass is significantly higher than high payload mass.
- All Machine Learning Algorithms performed the same, except for Decision Tree Classifier.

Thank you!