IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Vinay Mahamunkar
28-05-2024

Outline

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

Executive Summary

This project follows these steps:

- Data Collection
- Data Wrangling
- Exploratory Data Analysis
- Interactive Visual Analytics
- Predictive Analysis (Classification)

This project produced the following outputs and visualizations:

1. Exploratory Data Analysis (EDA) results
2. Geospatial analytics
3. Interactive dashboard
4. Predictive analysis of classification models

Introduction

- SpaceX launches Falcon 9 rockets at a cost of around $62m. This is considerably cheaper than other providers (which usually cost upwards of $165m), and much of the savings are because SpaceX can land, and then re-use the first stage of the rocket.

- If we can make predictions on whether the first stage will land, we can determine the cost of a launch, and use this information to assess whether or not an alternate company should bid and SpaceX for a rocket launch.

- This project will ultimately predict if the Space X Falcon 9 first stage will land successfully.

Section 1

Methodology

Methodology

Executive Summary

- **Data collection methodology:**

  Making GET requests to the SpaceX REST API
  Web Scraping

- **Perform data wrangling:**

  Using the `.fillna()` method to remove NaN values

  Using the `.value_counts()` method to determine the following:

  Number of launches on each site.

  Number and occurrence of each orbit.

  Number and occurrence of mission outcome per orbit type

  Creating a landing outcome label that shows the following:

  0 when the booster did not land successfully

  1 when the booster did land successfully

- **Perform exploratory data analysis (EDA) using visualization and SQL:**

  Using SQL queries to manipulate and evaluate the SpaceX dataset.

  Using Pandas and Matplotlib to visualize relationships between variables, and determine patterns.

Interactive Visual Analytics:

Geospatial analytics using Folium

Creating an interactive dashboard using Plotly Dash

Data Modelling and Evaluation:

Using Scikit-Learn to:

Pre-process (standardize) the data
Split the data into training and testing data using train_test_split
Train different classification models
Find hyperparameters using GridSearchCV

Plotting confusion matrices for each classification model.
Assessing the accuracy of each classification model.

Data Collection – SpaceX API

Using the SpaceX API to retrieve data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.

- Make a GET response to the SpaceX REST API
- Convert the response to a .json file then to a Pandas DataFrame
- Use custom logic to clean the data (see Appendix)
- Define lists for data to be stored in
- Call custom functions (see Appendix) to retrieve data and fill the lists
- Use these lists as values in a dictionary and construct the dataset
- Create a Pandas DataFrame from the constructed dictionary dataset
- Filter the DataFrame to only include Falcon 9 launches
- Reset the FlightNumber column
- Replace missing values of PayloadMass with the mean PayloadMass value

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

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

```
#Global variables
BoosterVersion = []
PayloadMass = []
Orbit = []
LaunchSite = []
Outcome = []
Flights = []
GridFins = []
Reused = []
Legs = []
LandingPad = []
Block = []
ReusedCount = []
Serial = []
Longitude = []
Latitude = []
```

```
# Call getBoosterVersion
getBoosterVersion(data)

the list has now been update

BoosterVersion[0:5]

we can apply the rest of the fu

# Call getLaunchSite
getLaunchSite(data)

# Call getPayloadData
getPayloadData(data)

# Call getCoreData
getCoreData(data)
```

```
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}
```

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

```
data_falcon9.isnull().sum()

data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9

data_falcon9 = data_falcon9.fillna(value={'PayloadMass': data_falcon9['PayloadMass'].mean()})
```


Github Link: https://github.com/vinaym09/coursera-ds/blob/main/Capstone%20Project/Data%20collection/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

- Web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy launches.
- Request the HTML page from the static URL
- Assign the response to an object
- Create a BeautifulSoup object from the HTML response object.
- Find all tables within the HTML page.
- Collect all column header names from the tables found within the HTML page
- Use the column names as keys in a dictionary
- Use custom functions and logic to parse all launch tables (see Appendix) to fill the dictionary values
- Convert the dictionary to a Pandas DataFrame ready for export

1

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

```
response = requests.get(static_url)
# assign the response to a object
data = response.text
```

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

2

```
# Use BeautifulSoup() to create a BeautifulSoup object
soup = BeautifulSoup(data, "html.parser")

html_tables = soup.find_all('table')
```

3

```
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if(name != None and len(name) > 0):
        column_names.append(name)
```

4

```
# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

5

```
df = pd.DataFrame(launch_dict)
```

Github Link:https://github.com/vinaym09/coursera-ds/blob/main/Capstone%20Project/Data%20collection/jupyter-labs-webscraping.ipynb

Data Wrangling

- The SpaceX dataset contains several Space X launch facilities, and each location is in the LaunchSite column.
- Each launch aims to a dedicated orbit, and some of the common orbit types are shown in the figure below. The orbit type is in the Orbit column.

Initial Data Exploration:

- Using the .value_counts() method to determine the following:

1. Number of launches on each site
2. Number and occurrence of each orbit
3. Number and occurrence of landing outcome per orbit type

```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()

CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()

GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

```
# landing_outcomes = values on Outcome column
landing_outcomes= df['Outcome'].value_counts()
landing_outcomes

True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

- The landing outcome is shown in the Outcome column:

True Ocean – the mission outcome was successfully landed to a specific region of the ocean

False Ocean – the mission outcome was unsuccessfully landed to a specific region of the ocean.

True RTLS – the mission outcome was successfully landed to a ground pad

False RTLS – the mission outcome was unsuccessfully landed to a ground pad.

True ASDS – the mission outcome was successfully landed to a drone ship

False ASDS – the mission outcome was unsuccessfully landed to a drone ship.

None ASDS and None None – these represent a failure to land

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes

{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

Data Wrangling

- To determine whether a booster will successfully land, it is best to have a binary column, i.e., where the value is 1 or 0, representing the success of the landing.
- This is done by:

1. Defining a set of unsuccessful (bad) outcomes, bad_outcome
2. Creating a list, landing_class, where the element is 0 if the corresponding row in Outcome is in the set bad_outcome, otherwise, it's 1.
3. Create a Class column that contains the values from the list landing_class
4. Export the DataFrame as a .csv file.

```
landing_class = []

for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
# landing_class = 1 otherwise
```

```
df['Class']=landing_class
df[['Class']].head(8)
```

```
df.to_csv("dataset_part_2.csv", index=False)
```

Github Link:https://github.com/vinaym09/coursera-ds/blob/main/Capstone%20Project/Data%20Wrangling/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

SCATTER CHARTS

Scatter charts are useful to observe relationships, or correlations, between two numeric variables.

Scatter charts were produced to visualize the relationships between:

- Flight Number and Launch Site
- Payload and Launch Site
- Orbit Type and Flight Number
- Payload and Orbit Type

BAR CHART

Bar charts are used to compare a numerical value to a categorical variable. Horizontal or vertical bar charts can be used, depending on the size of the data.

A bar chart was produced to visualize the relationship between:

- Success Rate and Orbit Type

LINE CHARTS

Line charts contain numerical values on both axes and are generally used to show the change of a variable over time.

Line charts were produced to visualize the relationships between:

- Success Rate and Year (i.e. the launch success yearly trend)

Github Link: https://github.com/vinaym09/coursera-ds/blob/main/Capstone%20Project/EDA/edadataviz.ipynb

EDA with SQL

- **The SQL queries performed on the data set were used to:**

1. Display the names of the unique launch sites in the space mission
2. Display 5 records where launch sites begin with the string ‘CCA’
3. Display the total payload mass carried by boosters launched by NASA (CRS)
4. Display the average payload mass carried by booster version F9 v1.1
5. List the date when the first successful landing outcome on a ground pad was achieved
6. List the names of the boosters which had success on a drone ship and a payload mass between 4000 and 6000 kg
7. List the total number of successful and failed mission outcomes
8. List the names of the booster versions which have carried the maximum payload mass
9. List the failed landing outcomes on drone ships, their booster versions, and launch site names for 2015
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

Github Link:https://github.com/vinaym09/coursera-ds/blob/main/Capstone%20Project/EDA/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

1. Mark all launch sites on a map

   Initialise the map using a Folium Map object
   Add a folium.Circle and folium.Marker for each launch site on the launch map

2. Mark the success/failed launches for each site on a map

   As many launches have the same coordinates, it makes sense to cluster them together.
   Before clustering them, assign a marker colour of successful (class = 1) as green, and failed (class = 0) as red.
   To put the launches into clusters, for each launch, add a folium.Marker to the MarkerCluster() object.
   Create an icon as a text label, assigning the icon_color as the marker_colour determined previously.

3. Calculate the distances between a launch site to its proximities

   To explore the proximities of launch sites, calculations of distances between points can be made using the Lat and Long values.
   After marking a point using the Lat and Long values, create a folium.Marker object to show the distance.
   To display the distance line between two points, draw a folium.PolyLine and add this to the map.

Github Link:https://github.com/vinaym09/coursera-ds/blob/main/Capstone%20Project/interactive%20visual%20analysis/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

1. Pie chart (px.pie()) showing the total successful launches per site
    - This makes it clear to see which sites are most successful
    - The chart could also be filtered (using a dcc.Dropdown() object) to see the success/failure ratio for an individual site

2. Scatter graph (px.scatter()) to show the correlation between outcome (success or not) and payload mass (kg)
    - This could be filtered (using a RangeSlider() object) by ranges of payload masses
    - It could also be filtered by booster version

Github Link:https://github.com/vinaym09/coursera-ds/blob/main/Capstone%20Project/interactive%20visual%20analysis/space_launch_ploatly.py

Predictive Analysis

The following steps were taking to develop, evaluate, and find the best performing classification model:

Model Development

- To prepare the dataset for model development:
  - Load dataset
  - Perform necessary data transformations (standardise and pre-process)
  - Split data into training and test data sets, using train_test_split()
  - Decide which type of machine learning algorithms are most appropriate
- For each chosen algorithm:
  - Create a GridSearchCV object and a dictionary of parameters
  - Fit the object to the parameters
  - Use the training data set to train the model

Model Evaluation

- For each chosen algorithm:
  - Using the output GridSearchCV object:
    - Check the tuned hyperparameters (best_params_)
    - Check the accuracy (score and best_score_)
  - Plot and examine the Confusion Matrix

Finding the Best Classification Model

- Review the accuracy scores for all chosen algorithms
- The model with the highest accuracy score is determined as the best performing model

Github Link:https://github.com/vinaym09/coursera-ds/blob/main/Capstone%20Project/Predictive%20analysis/predictive%20analysis.ipynb

Results

Exploratory data analysis

Interactive analytics

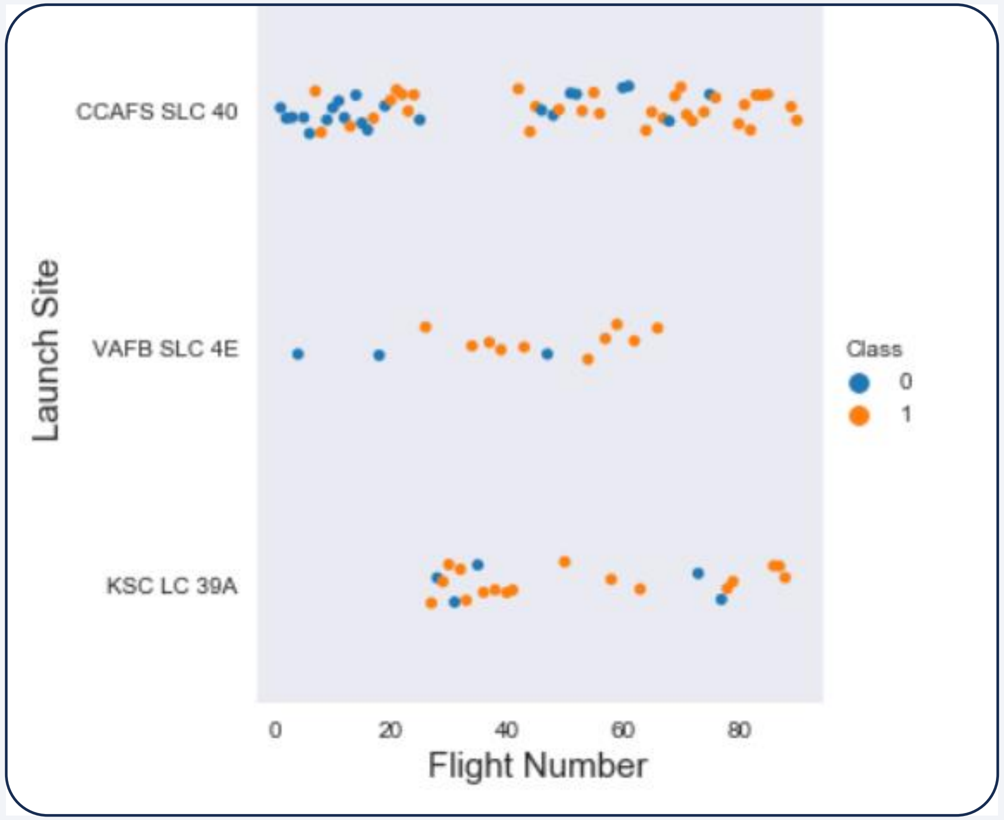
Predictive analysis

Section 2

Insights drawn from EDA

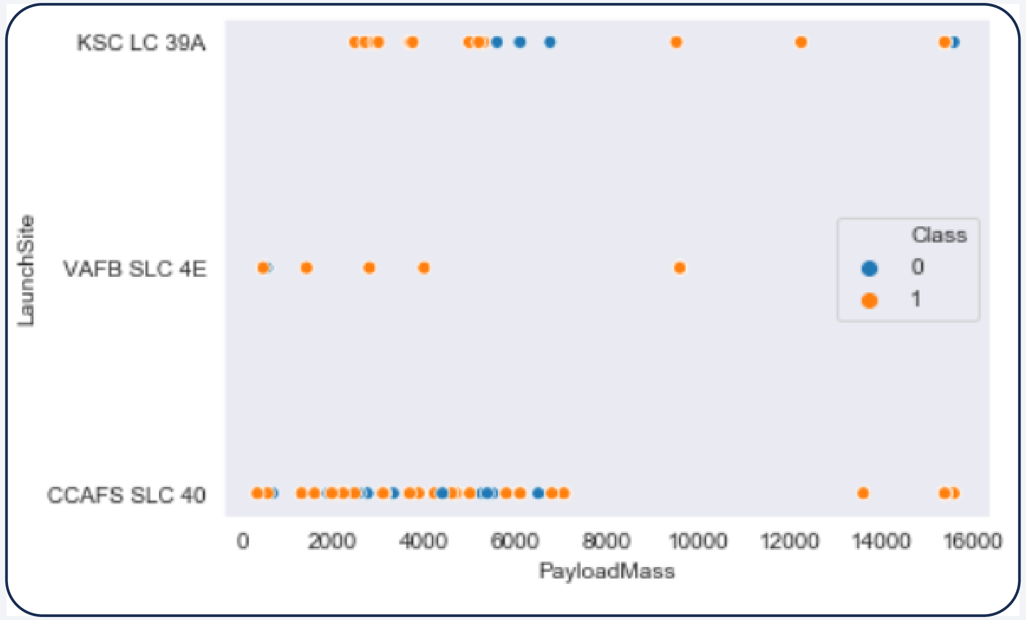
Flight Number vs. Launch Site

- The scatter plot of Launch Site vs. Flight Number shows that:
- As the number of flights increases, the rate of success at a launch site increases.
- Most of the early flights (flight numbers < 30) were launched from CCAFS SLC 40, and were generally unsuccessful.
- The flights from VAFB SLC 4E also show this trend, that earlier flights were less successful.
- No early flights were launched from KSC LC 39A, so the launches from this site are more successful.
- Above a flight number of around 30, there are significantly more successful landings (Class = 1).



Payload vs. Launch Site

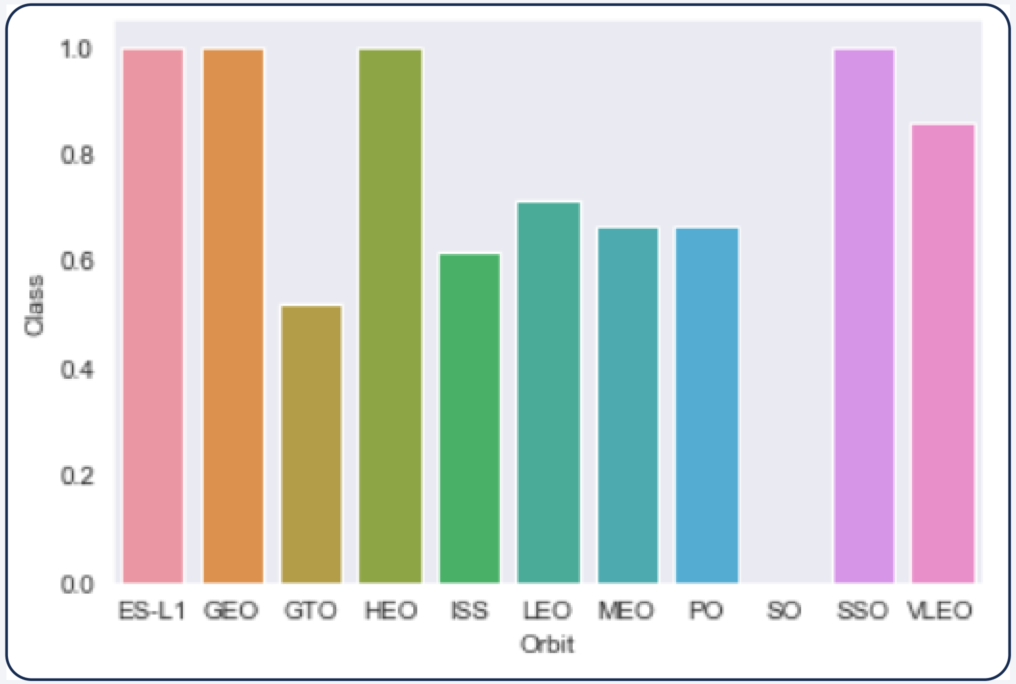
- The scatter plot of Launch Site vs. Payload Mass shows that:
- Above a payload mass of around 7000 kg, there are very few unsuccessful landings, but there is also far less data for these heavier launches.
- There is no clear correlation between payload mass and success rate for a given launch site.
- All sites launched a variety of payload masses, with most of the launches from CCAFS SLC 40 being comparatively lighter payloads (with some outliers).



Success Rate vs. Orbit Type

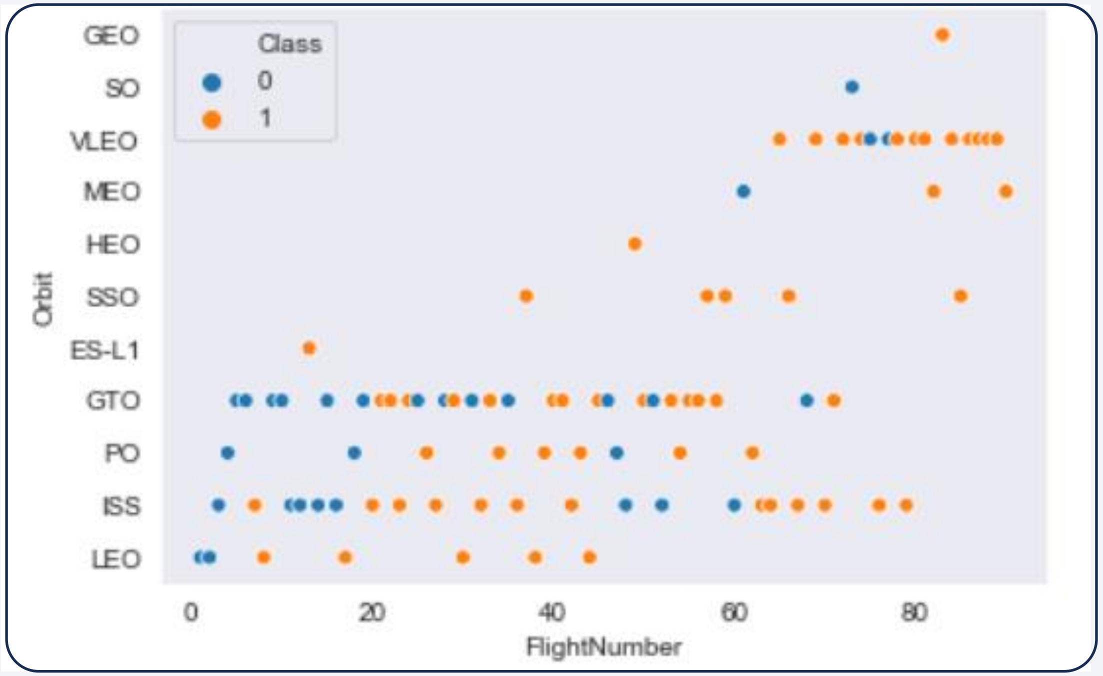
- The bar chart of Success Rate vs. Orbit Type shows that the following orbits have the highest (100%) success rate:
- ES-L1 (Earth-Sun First Lagrangian Point)
- GEO (Geostationary Orbit)
- HEO (High Earth Orbit)
- SSO (Sun-synchronous Orbit)

- The orbit with the lowest (0%) success rate is:
- SO (Heliocentric Orbit)



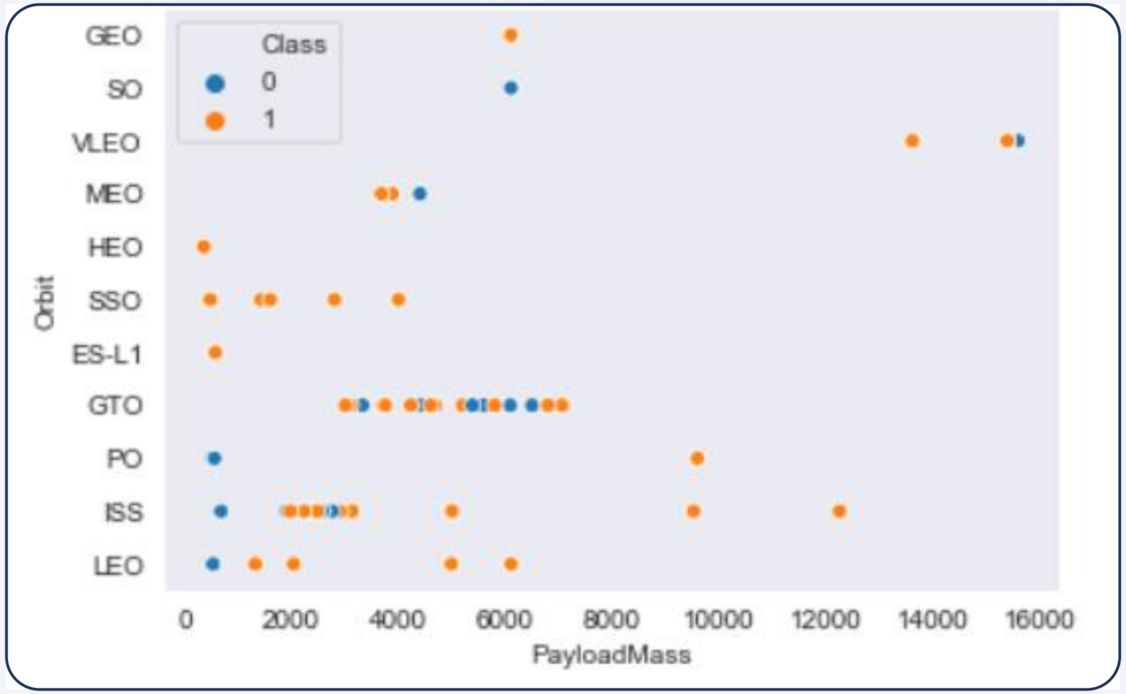
Flight Number vs. Orbit Type

- This scatter plot of Orbit Type vs. Flight number shows a few useful things that the previous plots did not, such as:
- The 100% success rate of GEO, HEO, and ES-L1 orbits can be explained by only having 1 flight into the respective orbits.
- The 100% success rate in SSO is more impressive, with 5 successful flights.
- There is little relationship between Flight Number and Success Rate for GTO.
- Generally, as Flight Number increases, the success rate increases. This is most extreme for LEO, where unsuccessful landings only occurred for the low flight numbers (early launches).



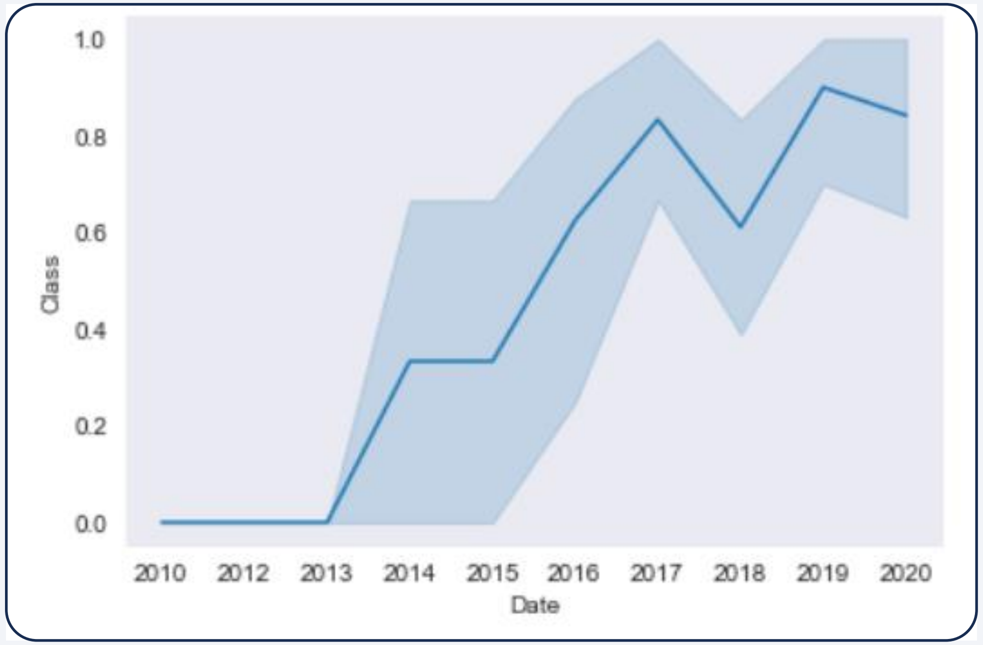
Payload vs. Orbit Type

- This scatter plot of Orbit Type vs. Payload Mass shows that:
- The following orbit types have more success with heavy payloads:
  - PO (although the number of data points is small)
  - ISS
  - LEO
- For GTO, the relationship between payload mass and success rate is unclear.
- VLEO (Very Low Earth Orbit) launches are associated with heavier payloads, which makes intuitive sense.



Launch Success Yearly Trend

- The line chart of yearly average success rate shows that:
- Between 2010 and 2013, all landings were unsuccessful (as the success rate is 0).
- After 2013, the success rate generally increased, despite small dips in 2018 and 2020.
- After 2016, there was always a greater than 50% chance of success.



All Launch Site Names

- Find the names of the unique launch sites.

```
%sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL;
```

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

The word UNIQUE returns only unique values from the LAUNCH_SITE column of the SPACEXTBL table.

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with ‘CCA’.

```
%sql SELECT LAUNCH_SITE FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

- LIMIT 5 fetches only 5 records, and the LIKE keyword is used with the wild card ‘CCA%’ to retrieve string values beginning with ‘CCA’.

Total Payload Mass

- Calculate the total payload carried by boosters from NASA.

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD_MASS FROM SPACEXTBL \
    WHERE CUSTOMER = 'NASA (CRS)';
```

TOTAL_PAYLOAD_MASS
45596

- The SUM keyword is used to calculate the total of the LAUNCH column, and the SUM keyword (and the associated condition) filters the results to only boosters from NASA (CRS).

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1.

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVERAGE_PAYLOAD_MASS FROM SPACEXTBL \
    WHERE BOOSTER_VERSION = 'F9 v1.1'
```

AVERAGE_PAYLOAD_MASS
2928.4

- The AVG keyword is used to calculate the average of the PAYLOAD_MASS__KG_ column, and the WHERE keyword (and the associated condition) filters the results to only the F9 v1.1 booster version.

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad.

```
%sql SELECT MIN(DATE) AS FIRST_SUCCESSFUL_GROUND_LANDING FROM SPACEXTBL \
    WHERE Landing_Outcome = 'Success (ground pad)';
```

FIRST_SUCCESSFUL_GROUND_LANDING
2015-12-22

- The MIN keyword is used to calculate the minimum of the DATE column, i.e. the first date, and the WHERE keyword (and the associated condition) filters the results to only the successful ground pad landings.

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
%sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTBL WHERE Landing_Outcome='Success (drone ship)'\
AND PAYLOAD_MASS__KG_ >4000 AND PAYLOAD_MASS__KG_ <6000
```

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

- The WHERE keyword is used to filter the results to include only those that satisfy both conditions in the brackets (as the AND keyword is also used). The BETWEEN keyword allows for 4000 < x < 6000 values to be selected.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcome.

```
%sql SELECT Mission_Outcome,COUNT(Mission_Outcome) FROM SPACEXTBL GROUP BY Mission_Outcome;
```

Mission_Outcome	COUNT(Mission_Outcome)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

- The COUNT keyword is used to calculate the total number of mission outcomes, and the GROUPBY keyword is also used to group these results by the type of mission outcome.

Boosters Carried Maximum Payload

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

```
%sql SELECT DISTINCT(BOOSTER_VERSION) FROM SPACEXTBL \
    WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

Booster_Version
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 subquery is used here. The SELECT statement within the brackets finds the maximum payload, and this value is used in the WHERE condition. The DISTINCT keyword is then used to retrieve only distinct /unique booster versions.

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015.

```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL \
    WHERE (LANDING__OUTCOME = 'Failure (drone ship)') AND (EXTRACT(YEAR FROM DATE) = '2015');
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

- The WHERE keyword is used to filter the results for only failed landing outcomes, AND only for the year of 2015.

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

- 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(Landing_Outcome) AS TOTAL_NUMBER FROM SPACEXTBL \
    WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
    GROUP BY Landing_Outcome \
    ORDER BY TOTAL_NUMBER DESC;
```

Landing_Outcome	TOTAL_NUMBER
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

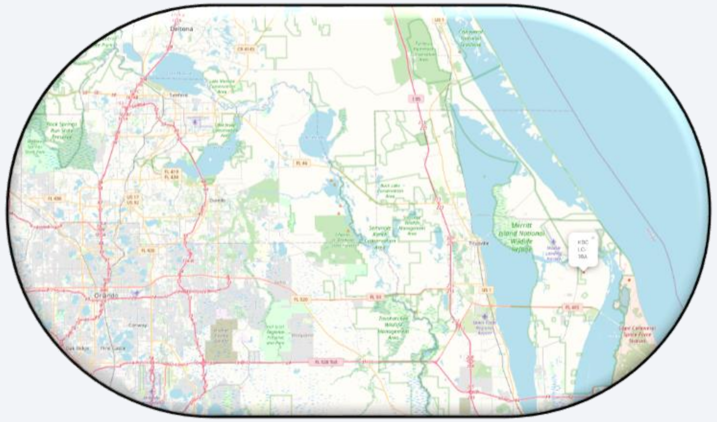
- The WHERE keyword is used with the BETWEEN keyword to filter the results to dates only within those specified. The results are then grouped and ordered, using the keywords GROUP BY and ORDER BY, respectively, where DESC is used to specify the descending order.

Section 3

Launch Sites Proximities Analysis

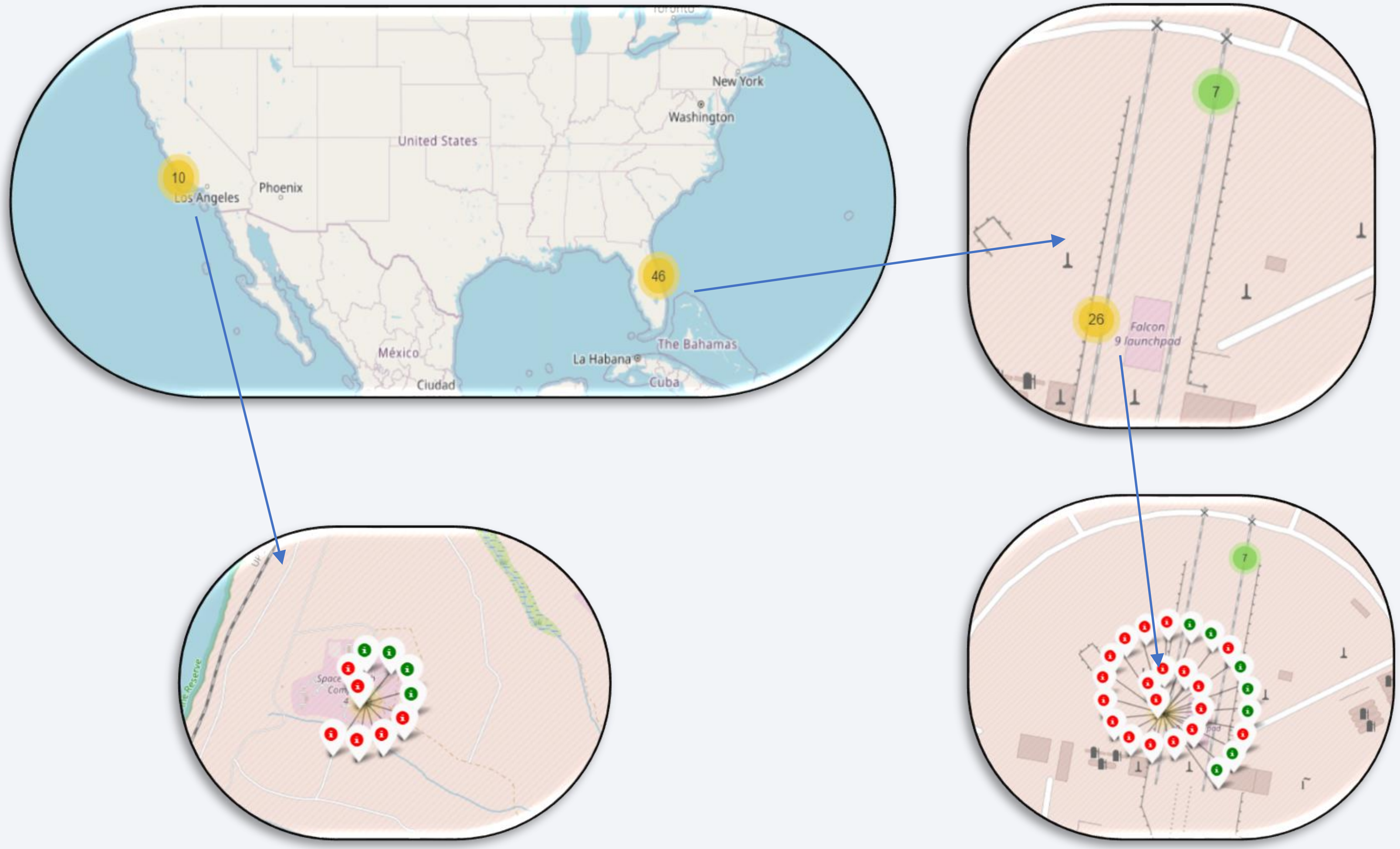
All launch sites map

All SpaceX launch sites are on coasts of the United States of America, specifically Florida and California.



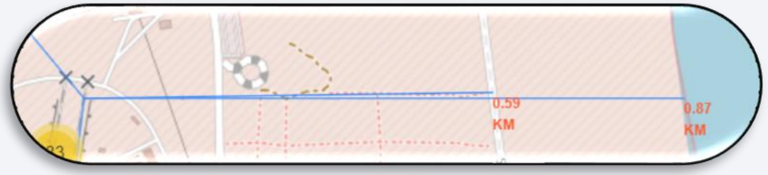
Success/failed launches for each site

Launches have been grouped into clusters, and annotated with green icons for successful launches, and red icons for failed launches.

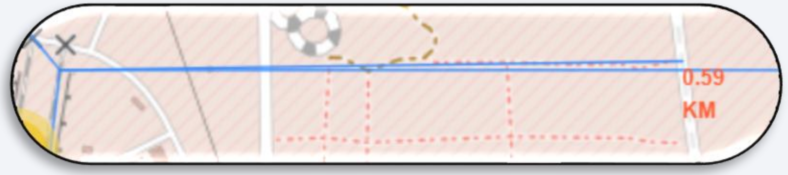


Proximity of launch sites to other points of interest

We can see the closest coastline, railway, highway, city from the launch site



Closest Coastline is 0.87KM away from launch Site.



Closest Highway is 0.59KM away from launch Site.

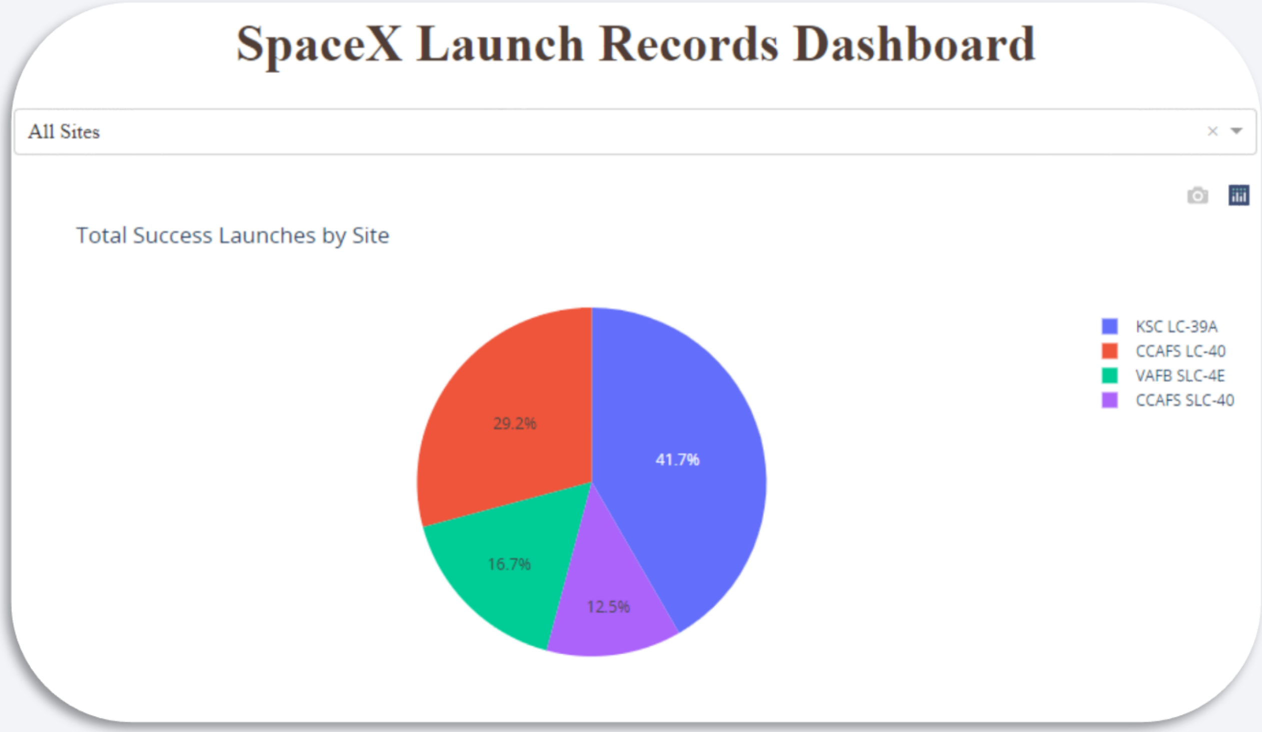
Closest Railway is 1.29KM away from launch Site.

Closest City is 51.74 KM away from launch Site.

Section 4

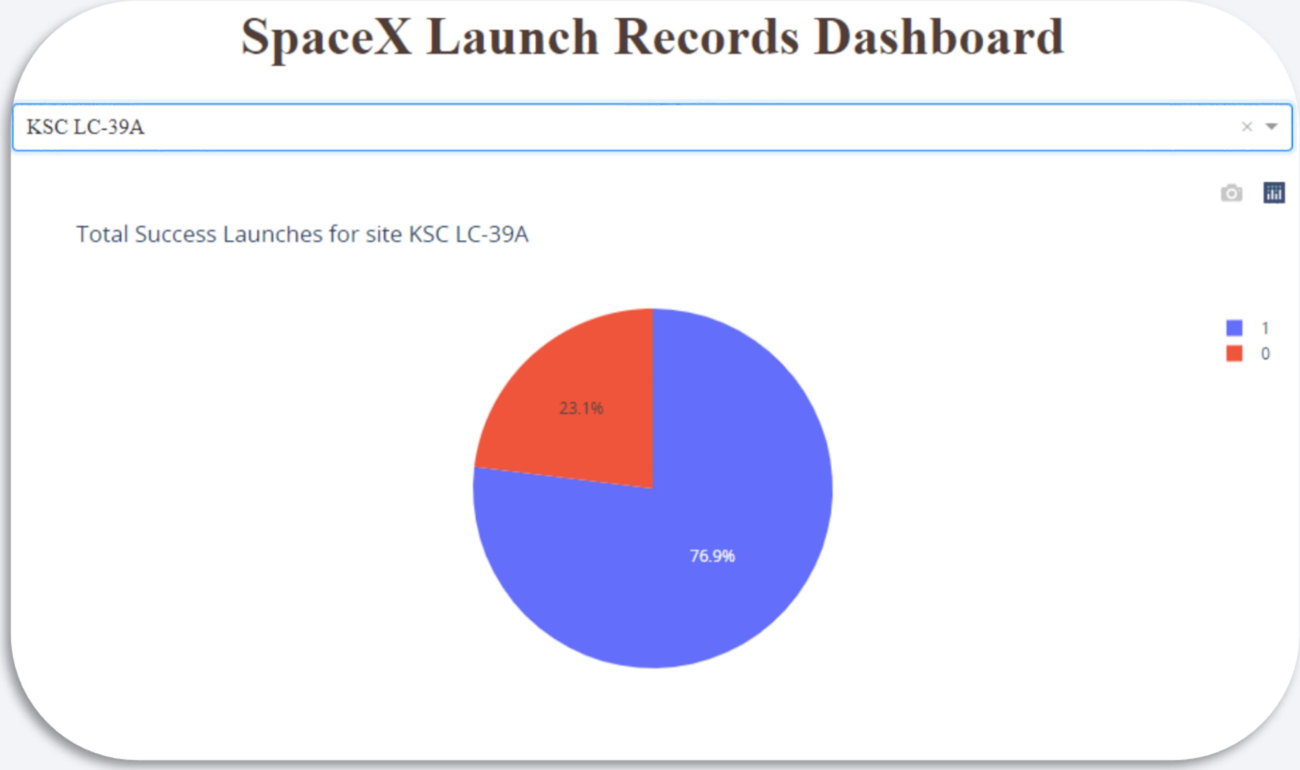
Build a Dashboard with Plotly Dash

launch success count for all sites



- The launch site KSC LC-39 A had the most successful launches, with 41.7% of the total successful launches.

Pie chart for the launch site with highest launch success ratio



The launch site KSC LC-39 A also had the highest rate of successful launches, with a 76.9% success rate.

Launch Outcome VS. Payload scatter plot for all sites

- Plotting the launch outcome vs. payload for all sites shows a gap around 4000 kg, so it makes sense to split the data into 2 ranges:
  - 0 – 4000 kg (low payloads)
  - 4000 – 10000 kg (massive payloads)
- From these 2 plots, it can be shown that the success for massive payloads is lower than that for low payloads.
- It is also worth noting that some booster types (v1.0 and B5) have not been launched with massive payloads.

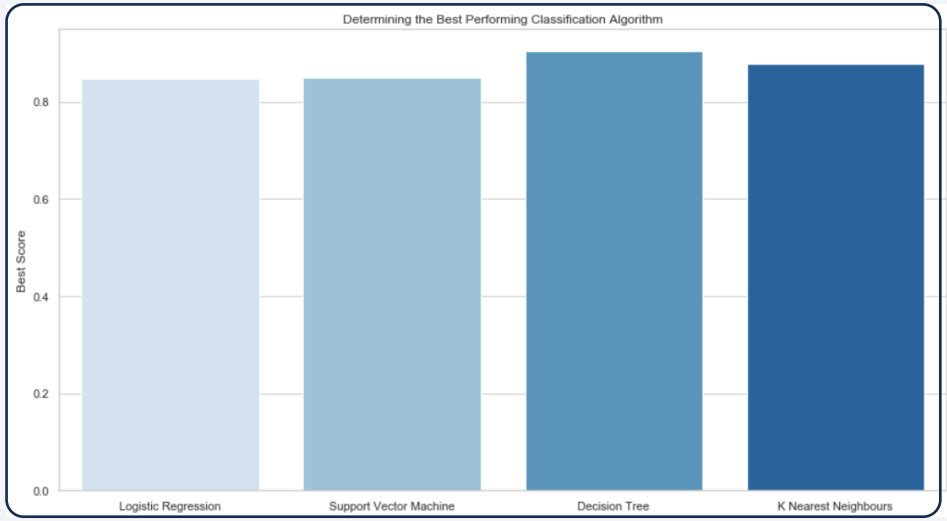
Section 5

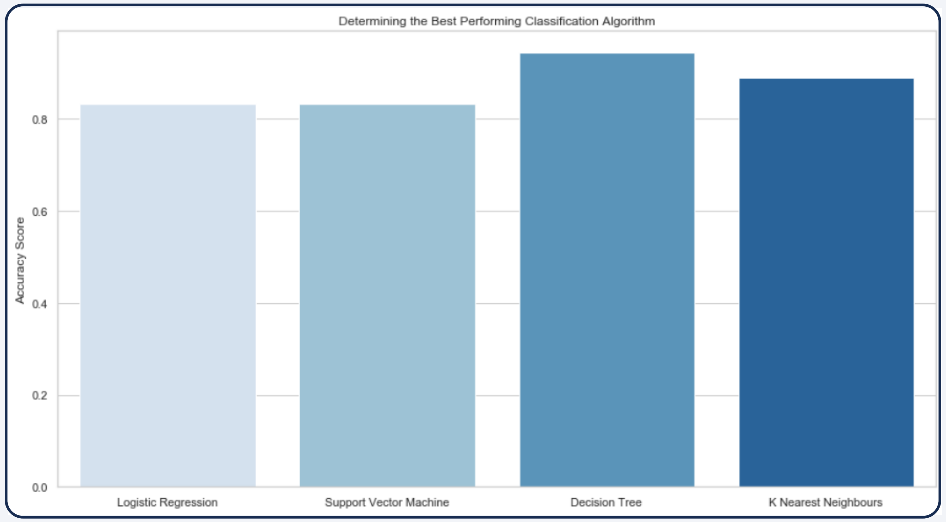
Predictive Analysis (Classification)

Classification Accuracy

- Plotting the Accuracy Score and Best Score for each classification algorithm produces the following result:
- The Decision Tree model has the highest classification accuracy
  - The Accuracy Score is 94.44%
  - The Best Score is 90.36%

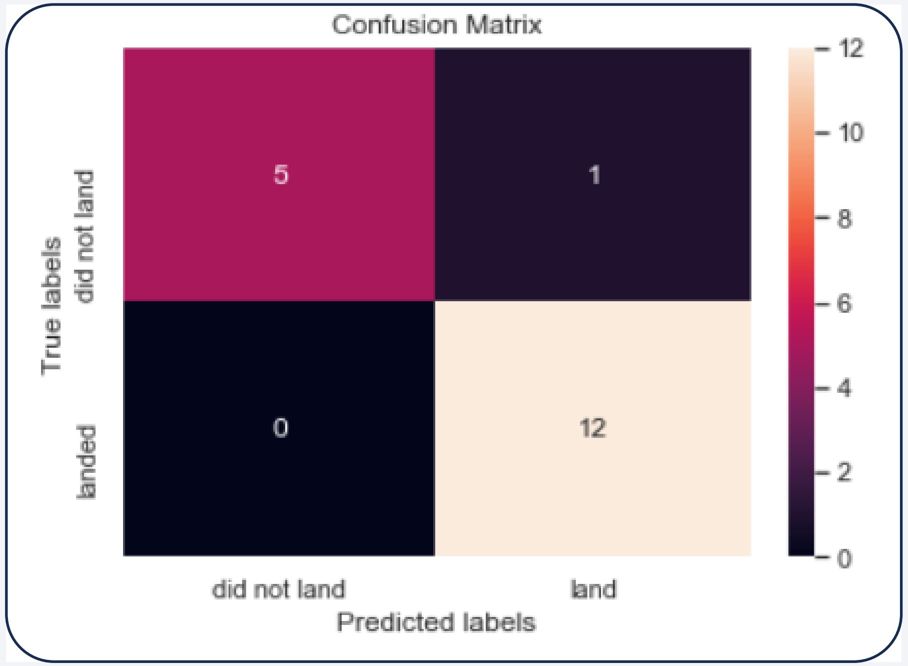
	Algorithm	Accuracy Score	Best Score
0	Logistic Regression	0.833333	0.846429
1	Support Vector Machine	0.833333	0.848214
2	Decision Tree	0.944444	0.903571
3	K Nearest Neighbours	0.888889	0.876786





Confusion Matrix

- As shown previously, best performing classification model is the Decision Tree model, with an accuracy of 94.44%.
- This is explained by the confusion matrix, which shows only 1 out of 18 total results classified incorrectly (a false positive, shown in the top-right corner).
- The other 17 results are correctly classified (5 did not land, 12 did land).



Conclusions

- As the number of flights increases, the rate of success at a launch site increases, with most early flights being unsuccessful. I.e. with more experience, the success rate increases.
  - Between 2010 and 2013, all landings were unsuccessful (as the success rate is 0).
  - After 2013, the success rate generally increased, despite small dips in 2018 and 2020.
  - After 2016, there was always a greater than 50% chance of success.

- Orbit types ES-L1, GEO, HEO, and SSO, have the highest (100%) success rate.
  - The 100% success rate of GEO, HEO, and ES-L1 orbits can be explained by only having 1 flight into the respective orbits.
  - The 100% success rate in SSO is more impressive, with 5 successful flights.
  - The orbit types PO, ISS, and LEO, have more success with heavy payloads:
  - VLEO (Very Low Earth Orbit) launches are associated with heavier payloads, which makes intuitive sense.

- The launch site KSC LC-39 A had the most successful launches, with 41.7% of the total successful launches, and also the highest rate of successful launches, with a 76.9% success rate.

- The success for massive payloads (over 4000kg) is lower than that for low payloads.

- The best performing classification model is the Decision Tree model, with an accuracy of 94.44%.

Thank you!