IBM Data Science Capstone Project

Space X Falcon 9 Landing Analysis

Sham Kin Tat Kinder

Github: IBM_Data_Science_Course

Nov 28, 2022

Outline

- Executive Summary
- Introduction
- Methodology
- Exploratory Data Analysis
- Results
- Conclusion
- Appendix

Executive Summary

Summary of methodologies

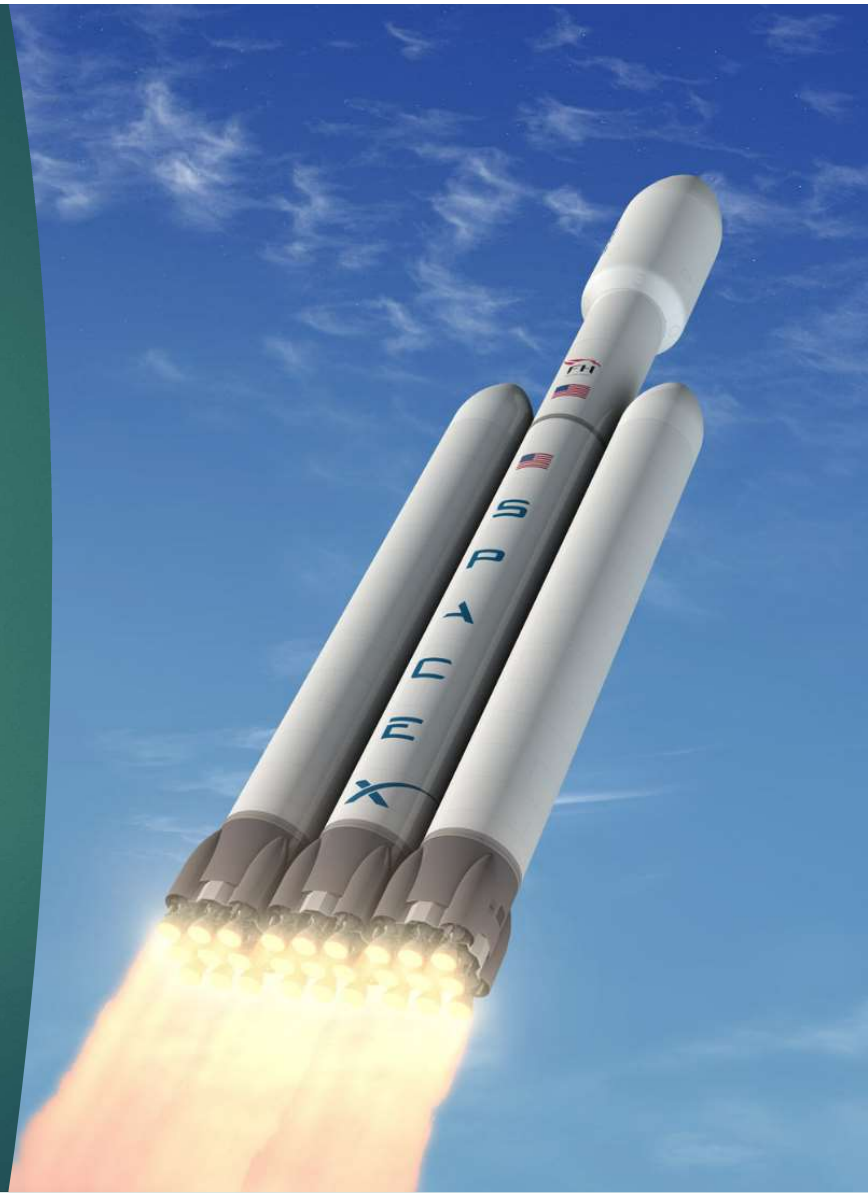
This project is following the *Data Science Process* which including:

1. Data Collection
2. Data Wrangling
3. Exploratory Data Analysis
4. Interactive Visual Analytics
5. Predictive Analysis (Classification)

Summary of all results

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

Project Background

- SpaceX spent about $62 million to launch the Falcon 9 rocket. The cost is much cheaper than other suppliers, more than $165 million, largely because SpaceX can reuse the first stage, which is a saving, so the rocket can be re-launched.
- If we can predict whether the first stage can land success or not, we can determine the costs and use that information to evaluate whether alternative companies should bid on SpaceX for the rocket launch.
- This project will predict the Space X Falcon 9 first stage can land successfully or not.

Problem Statement

- What affects whether a rocket can land successfully?
- Which variables will affect the successful landing of the rocket?
- What needs to be done by SpaceX to ensure the best rocket landing success rate.

Section 1

Methodology

Methodology

- Data collection methodology:
  - Combined data from SpaceX public API and web scaping SpaceX Wikipedia page
- Perform data wrangling
  - Classifying true landings as successful and unsuccessful otherwise
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Using GridSearchCV to Tuned the machine learning models

Data Collection

- Data collection process involved a combination of API requests from Space X public API and web scraping data from a table in Space X's Wikipedia entry.
- The next slide will show the flowchart of data collection from API and the one after will show the flowchart of data collection from webscraping.

Data Collection – SpaceX API

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

Github: jupyter-labs-spacex-data-collection-api.ipynb

1. Getting Response from API
2. Converting Response to a .json file
3. Apply custom functions to clean data
4. Assign list to dictionary then dataframe
5. Filter dataframe and export to flat file (.csv)

1

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

2

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

3

```
getBoosterVersion(data)
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
```

4

```
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}
```

5

```
data_falcon9 = df[df['BoosterVersion']!= 'Falcon 1']
data_falcon9.to_csv('spacex-data-collection-api.csv', index=False)
```

Data Collection – Scraping

A web scrape of historical Falcon 9 launch records was collected from the Wikipedia page titled "List of Falcon 9 and Falcon Heavy launches".

Github: jupyter-labs-webscraping.ipynb

1. Getting Response from HTML
2. Creating BeautifulSoup Object
3. Finding the HTML tables
4. Getting column names
5. Creation of dictionary
6. Appending data to dictionary
7. Converting dictionary to dataframe
8. Dataframe to .CSV

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)
```

2
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data, 'html5lib')
```

3
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

4
```
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 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)
```

5
```
launch_dict= dict.fromkeys(column_names)

# 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']=[]
```

6
```
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')
```

7
```
df=pd.DataFrame(launch_dict)
```

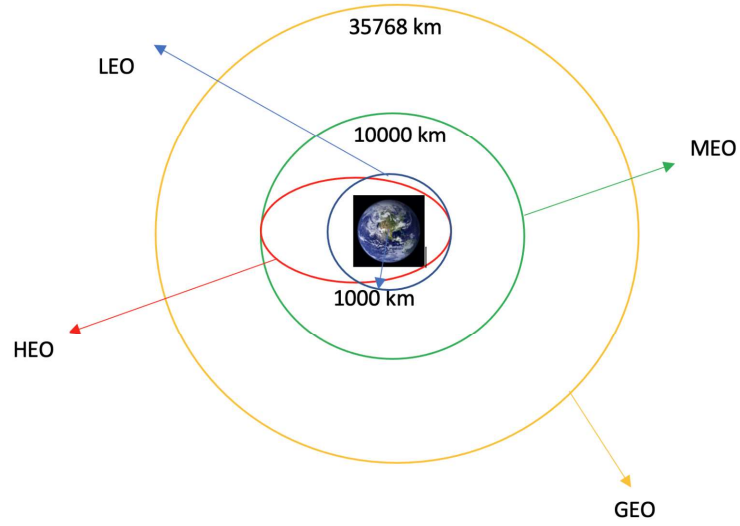
8
```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

- Create a label with landing outcomes where successful = 1 & failure = 0.
- Outcome column has two components: ‘Mission Outcome’ ‘Landing Location’
- New training label column ‘class’ with a value of 1 if ‘Mission Outcome’ is True and 0 otherwise. Value Mapping:
- True ASDS, True RTLS, & True Ocean – set to -> 1
- None None, False ASDS, None ASDS, False Ocean, False RTLS – set to -> 0

- Github: data-wrangling.ipynb

Diagram showing common orbit types SpaceX uses



Exploratory Data Analysis (EDA) Visualization

SCATTER GRAPHS

Scatter graphs were produced to visualize the relationships between:

1. Flight Number and Launch Site
2. Payload and Launch Site
3. Orbit Type and Flight Number
4. Payload and Orbit Type

Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation . Scatter plots usually consist of a large body of data.

BAR GRAPHS

A bar chart was produced to visualize the relationship between:

1. Success Rate and Orbit Type

A bar diagram makes it easy to compare sets of data between different groups at a glance. The graph represents categories on one axis and a discrete value in the other. The goal is to show the relationship between the two axes. Bar charts can also show big changes in data over time.

LINE GRAPHS

Line charts were produced to visualize the relationships between:

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

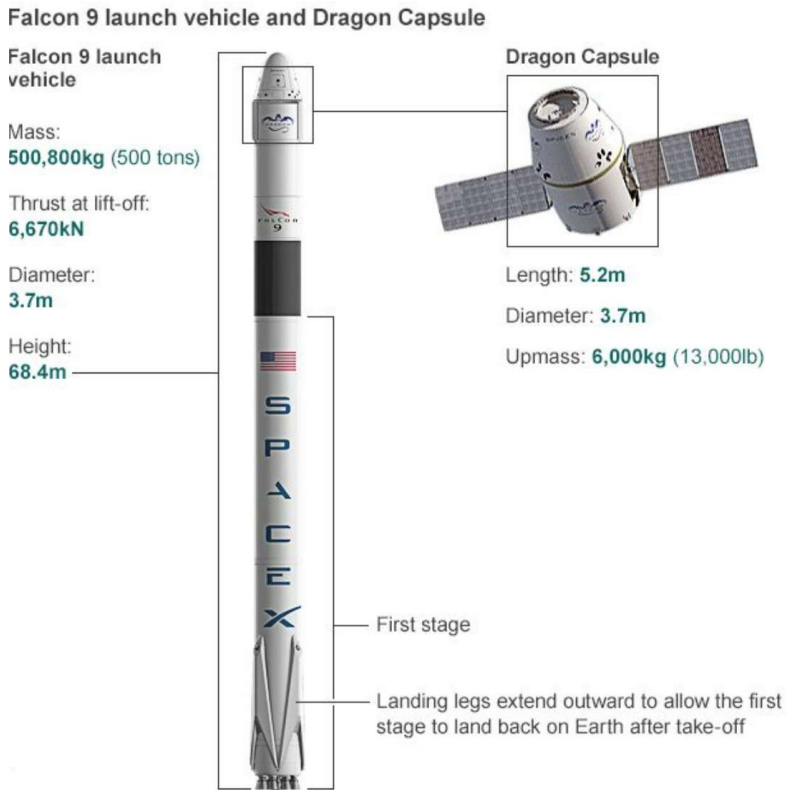
Line graphs are useful in that they show data variables and trends very clearly and can help to make predictions about the results of data not yet recorded

Github: eda-visualization.ipynb

Exploratory Data Analysis (EDA) SQL

Performed SQL queries to gather information about the dataset. For example, of some questions we were asked about the data we needed information about. Which we are using SQL queries to get the answers in the dataset :

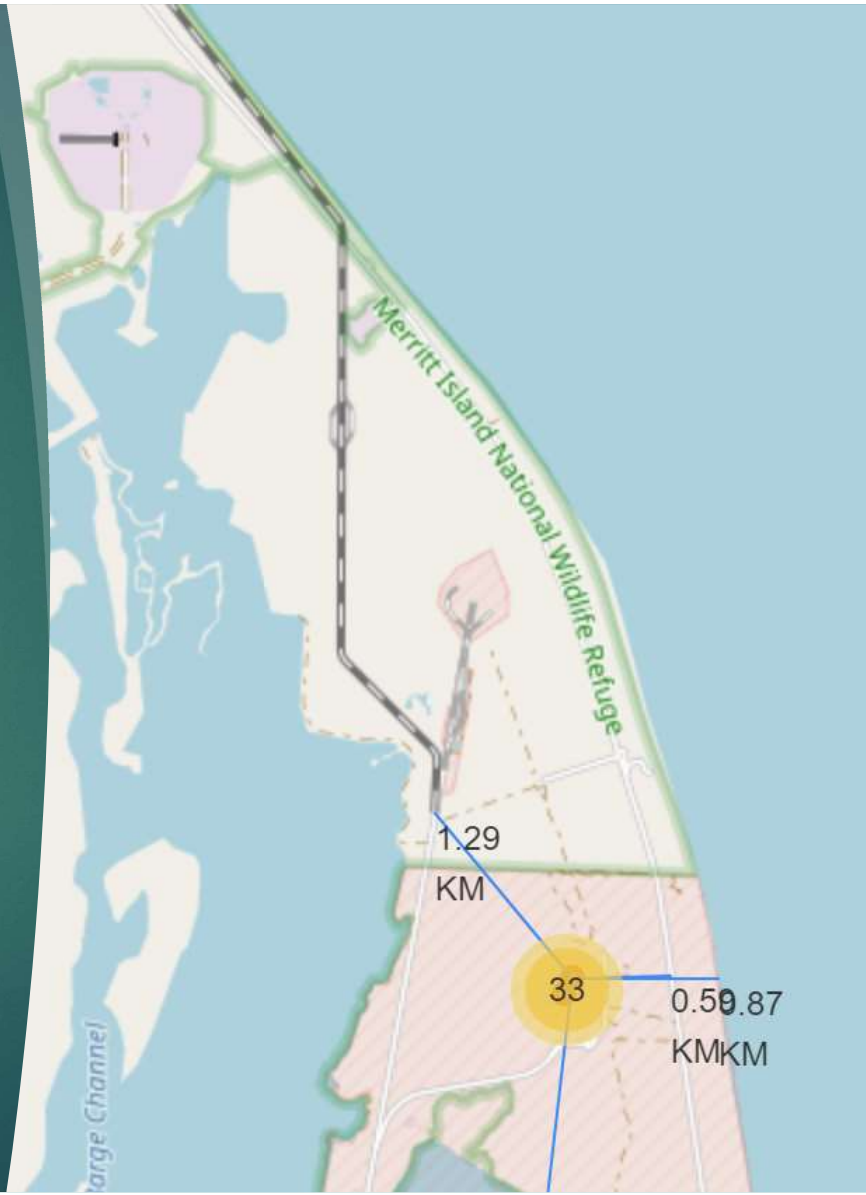
- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display the average payload mass carried by booster version F9 v1.1
- List the date when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



Building an interactive map with Folium

- To visualize the Launch Data into an interactive map.
  - Initialise the map using a Folium Map object
  - Add a folium.Circle and folium.Marker for each launch site on the launch map
- We assigned the dataframe launch_outcomes(failures, successes) to classes 0 and 1.
  - 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.
- Using Haversine's formula we calculated the distance
  - 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.

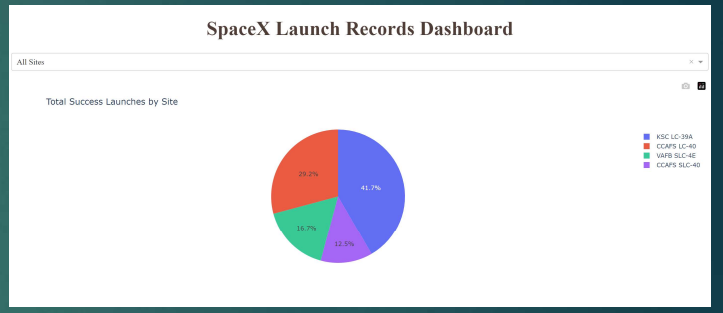
▸Github: launch-site-location.ipynb

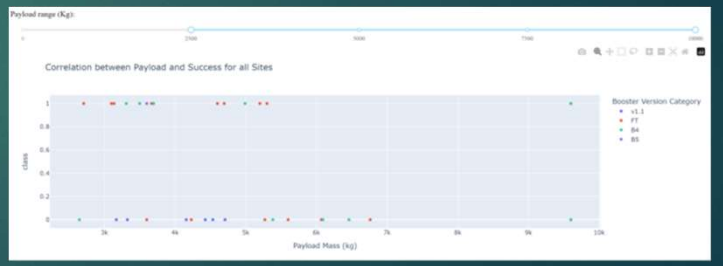


Build a Dashboard with Plotly Dash

Dashboard includes a pie chart and a scatter plot.

- Pie chart (px.pie()) showing the total successful launches per site
  1. This makes it clear to see which sites are most successful
  2. The chart could also be filtered (using a dcc.Dropdown() object) to see the success/failure ratio for an individual site
- Scatter graph (px.scatter()) to show the correlation between outcome (success or not) and payload mass (kg)
  1. This could be filtered (using a RangeSlider() object) by ranges of payload masses
  2. It could also be filtered by booster version





Github: spacex_dash_app.py

Predictive Analysis (Classification)

1. BUILDING MODEL

- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our dataset.

2. EVALUATING MODEL

- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion Matrix

3. IMPROVING MODEL

- Feature Engineering
- Algorithm Tuning

4. FINDING THE BEST PERFORMING CLASSIFICATION MODEL

- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

Github: SpaceX Machine Learning Prediction

Results

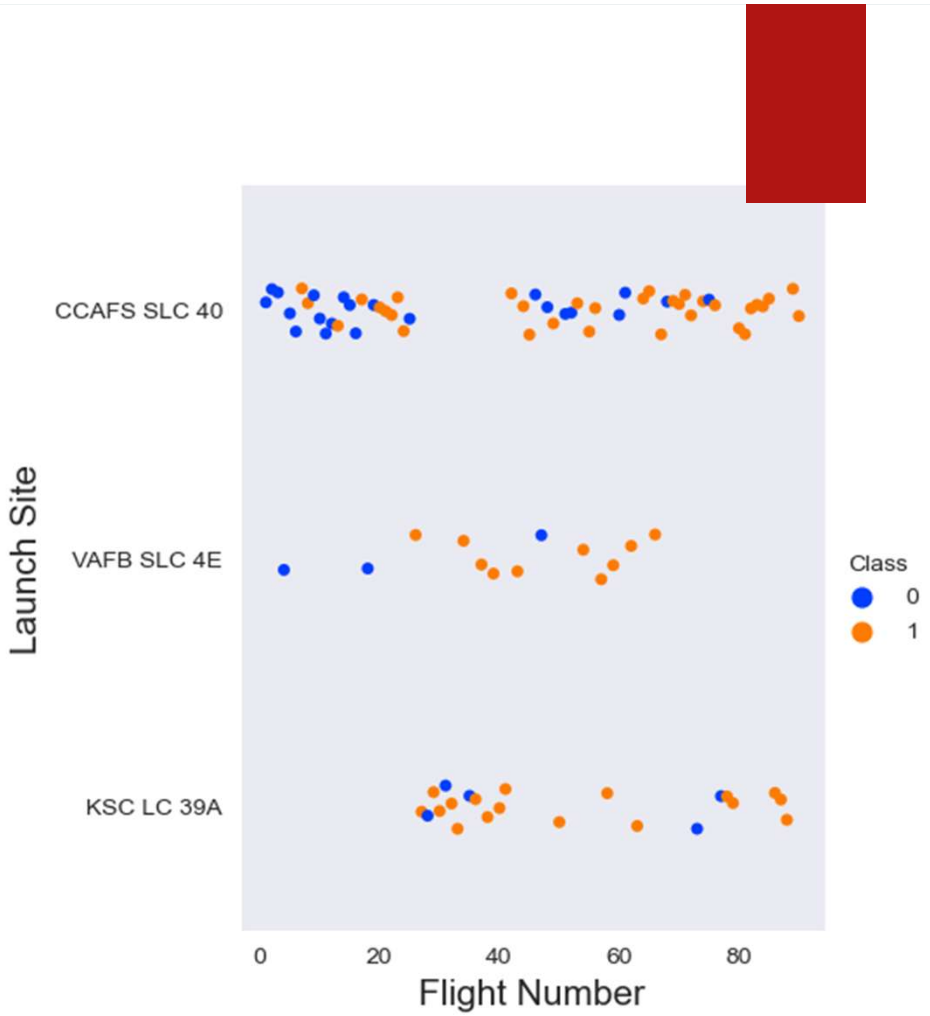
- Exploratory Data Analysis
- Interactive Analytics
- Predictive Analysis

Section 2
EDA with Visualization

Flight Number vs Launch Site

The scatter plot of Launch Site vs Flight Number shows that:

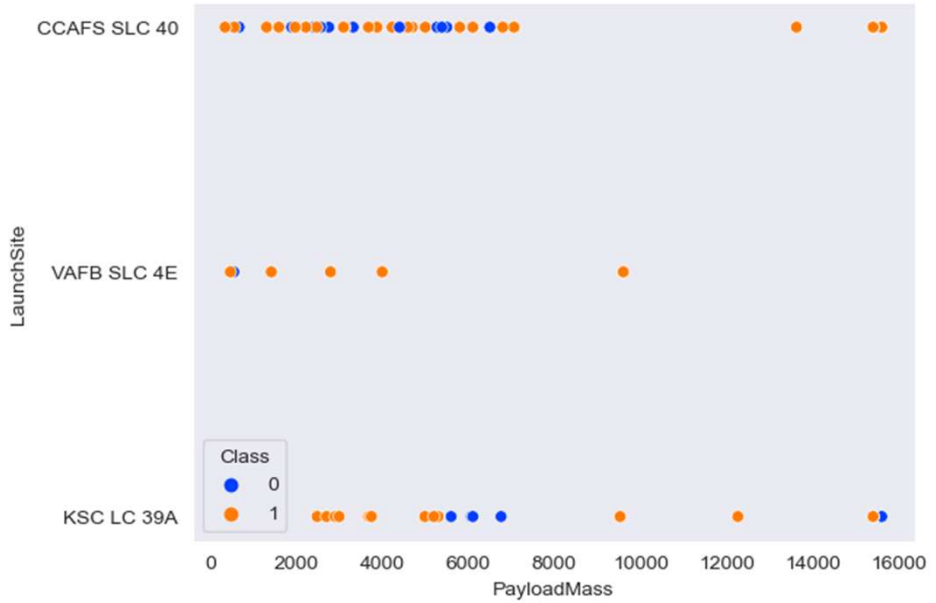
- The more flights at the launch site, the higher the success rate of the launch site
- Early flights (flight numbers < 30) were launched from CCAFS SLC 40 but were usually unsuccessful.
- While flights from VAFB SLC 4E also showed this trend, the earlier flights were less successful.
- KSC LC 39A did not launch earlier flights, so launches from this site were more successful.
- Flights number more than 30, there are much more successful landings (level = 1).



Payload vs Launch Site

The scatter plot of Launch Site vs Payload shows that:

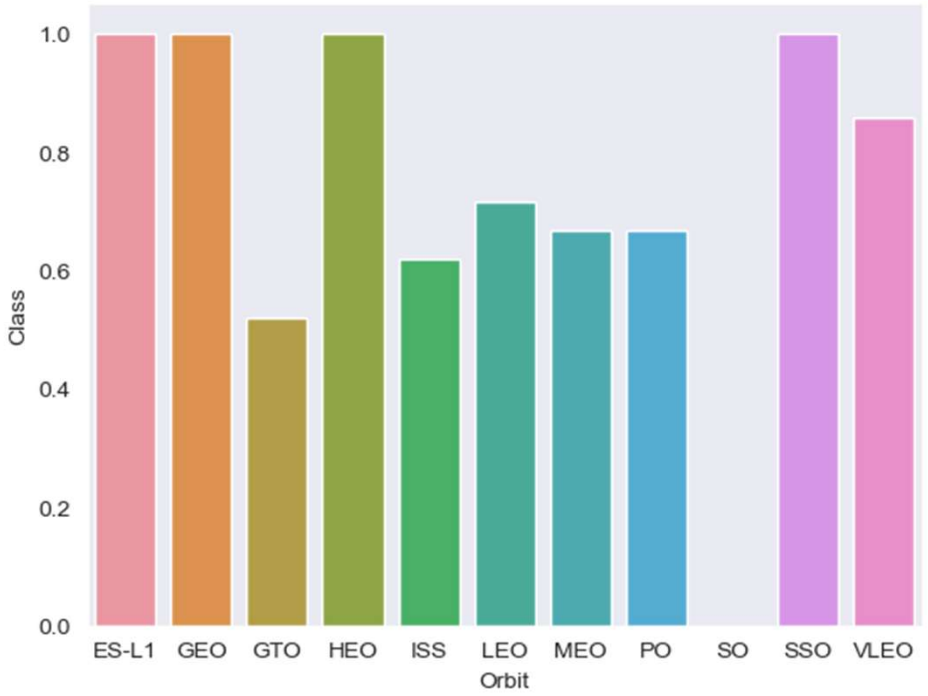
- Green indicates a successful launch; purple indicates a failed launch.
- Payload mass seems to fall mostly between 0-6000kg. Very few successful landings above a payload mass of about 7,000kg, but there is also much less data on these heavier launches.
- No clear correlation between payload mass and success rate for a given launch site.
- Different launch sites also appear to use different payload masses. All sites launched a variety of payload masses, with most launches from CCAFS SLC 40 being relatively light payloads (with some outliers).



Success Rate vs Orbit Type

A bar graph of success rate versus orbit type shows that the following tracks have the highest (100%) success rate:

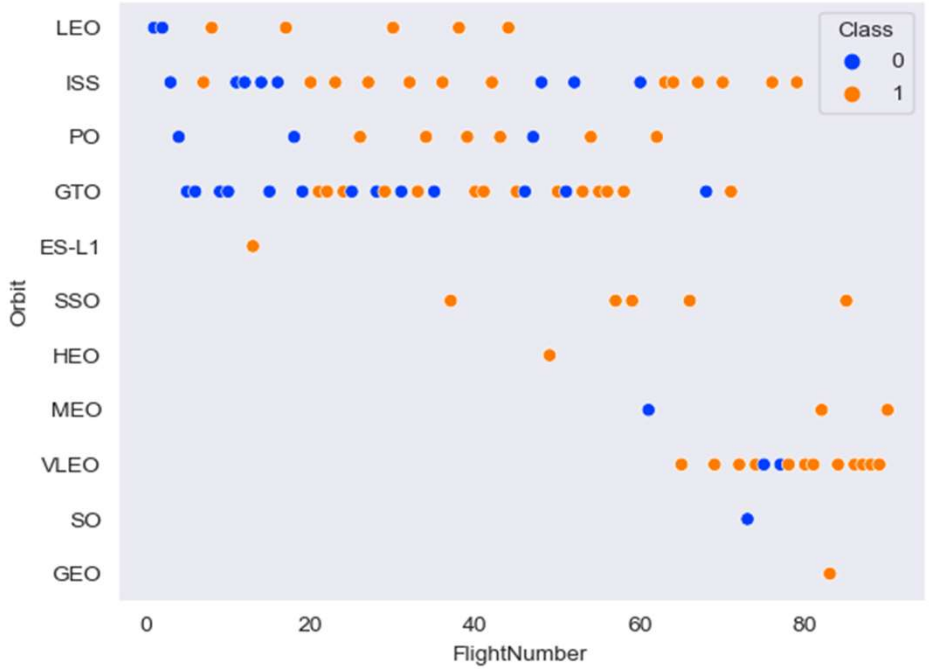
- ES-L1 (1), GEO (1), HEO (1) have 100% success rate (sample sizes in parenthesis) SSO (5) has 100% success rate
- VLEO (14) has decent success rate and attempts
- GTO (27) has the around 50% success rate but largest sample
- SO (1) has 0% success rate



Flight Number vs Orbit type

The scatterplot of orbit types versus flight numbers shows some useful things that the previous plots did not, such as:

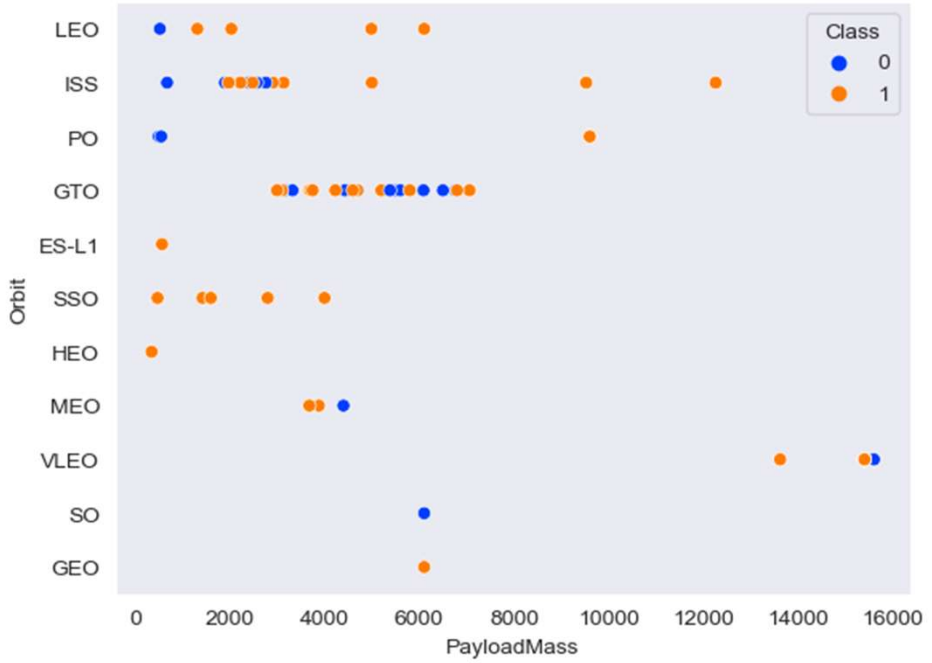
- Launch Orbit preference changed to Flight Number.
- Launch Outcome seems to correlate with this preference.
- SpaceX started in LEO orbit, had some success in LEO, and returned to VLEO for recent launches
- SpaceX seems to do better in lower or sun-synchronous orbits



Payload vs Orbit type

This scatter plot of Orbit Type vs Payload Mass shows that:

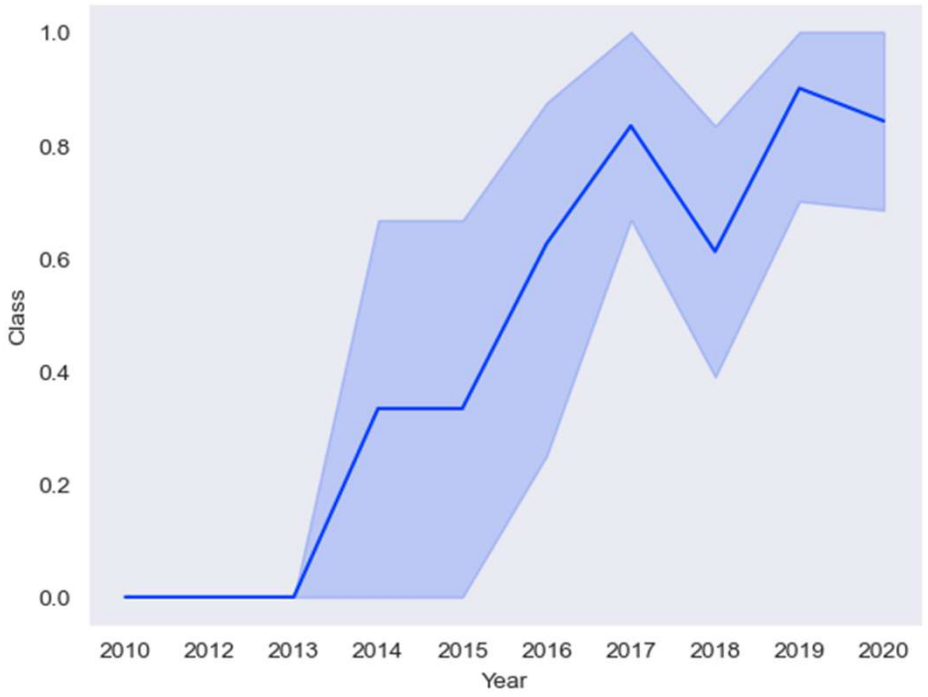
- Payload mass appears to be orbit-dependent
- PO, International Space Station and Leo track are more successful with heavy loads
- For GTO, the relationship between payload mass and success rate is unclear.
- Another most successful orbital VLEO only has payload mass values at the high end of the range



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).
- Success generally increases over time since 2013 with a slight dip in 2018
- Success in recent years at around 80%



Section 3 EDA with SQL

All Launch Site Names

Find the names of the unique launch sites.

- Using the word DISTINCT in the query means that it will only
- Show Unique values in the Launch_Site column from tblSpaceX

Task 1

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

```
In [11]: %sql select DISTINCT LAUNCH_SITE from SPACEX
```

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

Out[11]:

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

Launch Site Names Beginning with `CCA`

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

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

Task 2

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

In [12]:
```
%sql select * from SPACEX where launch_site like 'CCA%' limit 5
```

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

Out[12]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS_KG	Orbit	Customer	Mission_Outcome	Landing_Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

Calculate the total payload carried by boosters from NASA.

- This query sums the total payload mass in kg where NASA was the customer.
- CRS stands for Commercial Resupply Services which indicates that these payloads were sent to the International Space Station (ISS).

Task 3

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

In [13]:
```
%sql select sum(payload_mass_kg) as sum from SPACEX where customer like 'NASA (CRS)'
```

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

Out[13]:

sum
45596

Average Payload Mass by F9 v1.1

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

- This query calculates the average payload mass or launches which used booster version F9 v1.1
- Average payload mass of F9 1.1 is on the low end of our payload mass range

Task 4

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

```
In [14]: %sql select avg(payload_mass_kg) as Average from SPACEX where booster_version like 'F9 v1.1%'
 * sqlite:///spacex.db
Done.
```

Out[14]:

Average
2534.6666666666665

First Successful Ground Landing Date

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

- This query returns the first successful ground pad landing date.
- First ground pad landing wasn't until the end of 2015.
- Successful landings in general appear starting 2014.

Task 5

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

Hint:Use min function

```
In [19]: %sql select min(date) as Date from SPACEX where mission_outcome like 'Success'
 * sqlite:///spacex.db
Done.
```

Out[19]:

Date
01-03-2013

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.

- This query returns the four booster versions that had successful drone ship landings and a payload mass between 4000 and 6000 noninclusively.

Task 6

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

In [20]:
```
%sql select booster_version from SPACEX where (mission_outcome like 'Success') AND (payload_mass_kg between 4000 AND 6000) AND (landing_out
```

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

Out[20]:

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

Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcome.

- This query returns a count of each mission outcome.
- SpaceX appears to achieve its mission outcome nearly 99% of the time.
- This means that most of the landing failures are intended.
- Interestingly, one launch has an unclear payload status and unfortunately one failed in flight.

Task 7

List the total number of successful and failure mission outcomes

```
In [21]: %sql SELECT mission_outcome, count(*) as Count FROM SPACEX GROUP by mission_outcome ORDER BY mission_outcome
 * sqlite:///spacex.db
Done.
```

Out[21]:

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

Boosters Carried Maximum Payload

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

- This query returns the booster versions that carried the highest payload mass of 15600 kg.
- These booster versions are very similar and all are of the F9 B5 B10xx.x variety.
- This likely indicates payload mass correlates with the booster version that is used.

Task 8

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

In [23]:
```
maxm = %sql select max(payload_mass_kg) from SPACEX
maxv = maxm[0][0]

%sql select booster_version from SPACEX where payload_mass_kg=(select max(payload_mass_kg) from SPACEX)
```

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

Out[23]:

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

2015 Launch Records

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

- This query returns the Month, Landing Outcome, Booster Version, Payload Mass (kg), and Launch site of 2015 launches where stage 1 failed to land on a drone ship.
- There were two such occurrences.

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, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

In [27]:
```
%sql select substr(Date, 4, 2) as Month, landing_outcome, booster_version, launch_site from SPACEX where substr(Date,7,4) = '2015' AND land
```

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

Out[27]:

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

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.

- This query returns a list of successful landings and between 2010-06-04 and 2017-03-20 inclusively.
- There are two types of successful landing outcomes: drone ship and ground pad landings.
- There were 8 successful landings in total during this time period

Task 10

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order. ¶

```
In [29]: %sql select landing_Outcome, count(*) as count from SPACEX  where Date >= '04-06-2010' AND Date <= '20-03-2017' GROUP by landing_outcome O
```

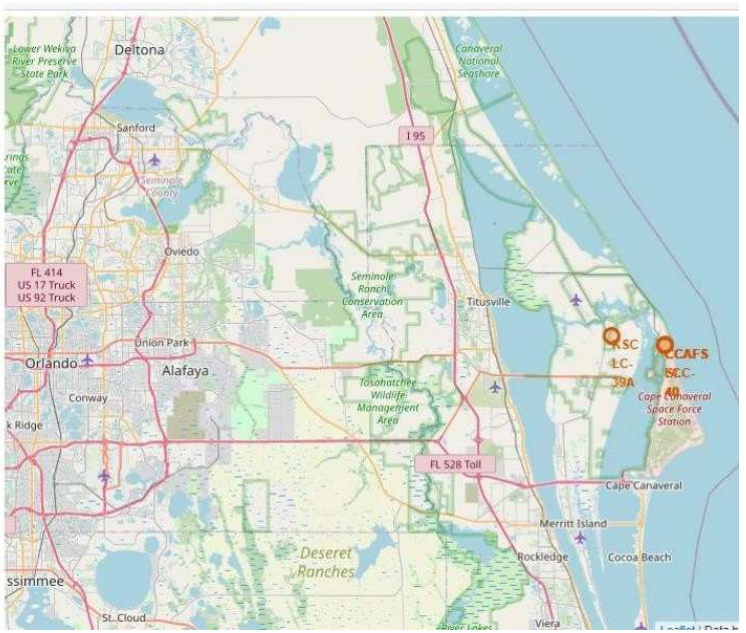
```
 * sqlite:///spacex.db
Done.
```

Out[29]:

Landing_Outcome	count
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

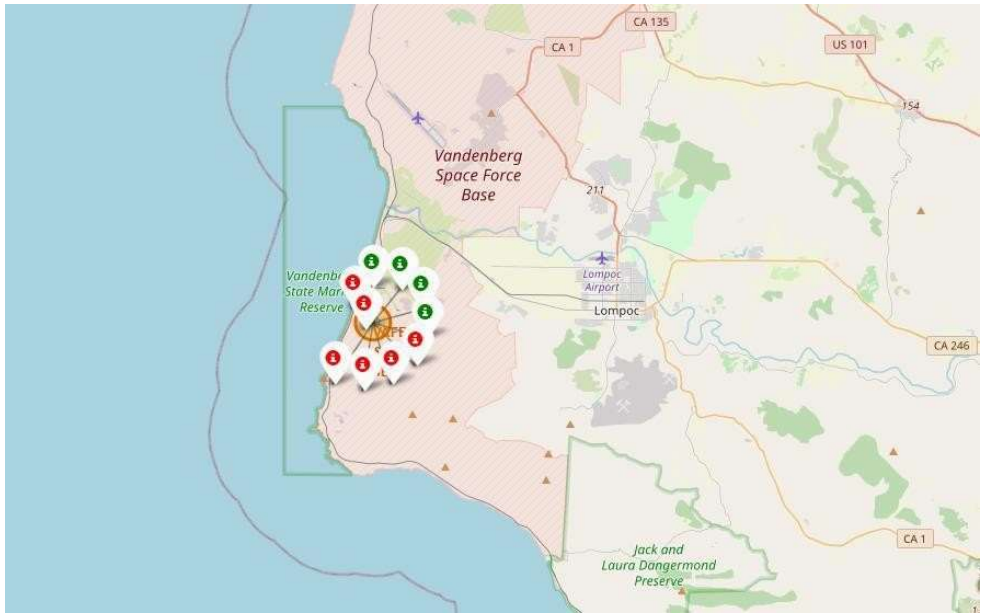
Section 4 Interactive Map

Launch Site Locations



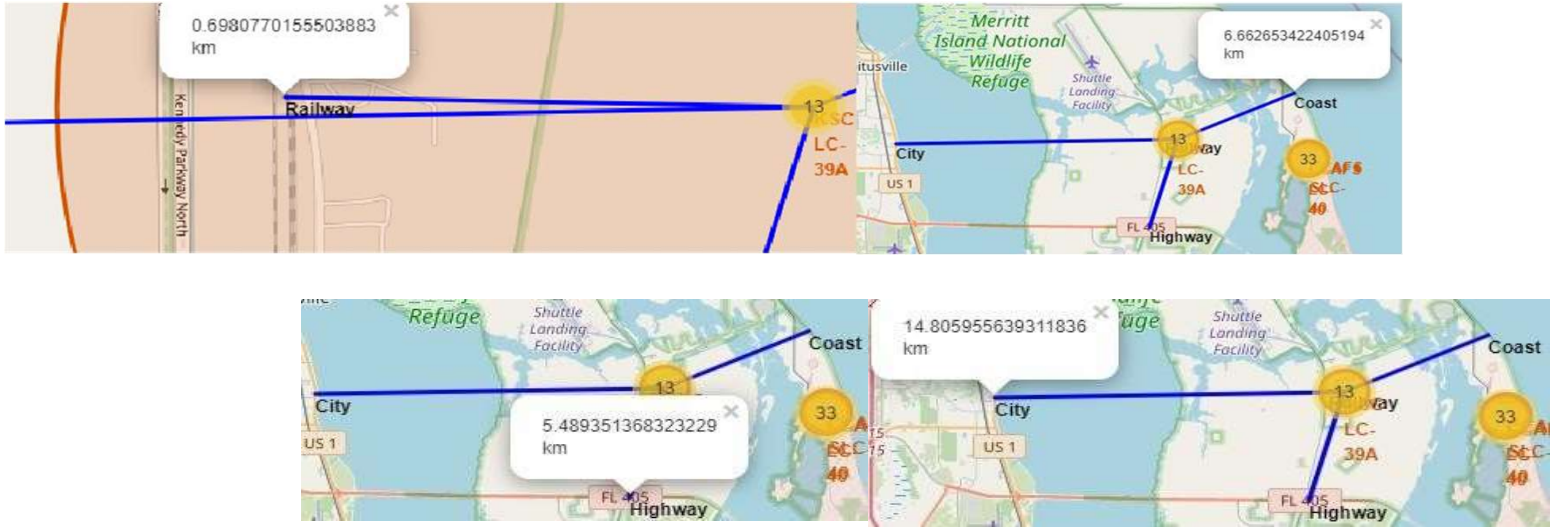
The left map shows all launch sites relative US map. The right map shows the two Florida launch sites since they are very close to each other. All launch sites are near the ocean.

Color-Coded Launch Markers



Clusters on Folium map can be clicked on to display each successful landing (green icon) and failed landing (red icon). In this example VAFB SLC-4E shows 4 successful landings and 6 failed landings.

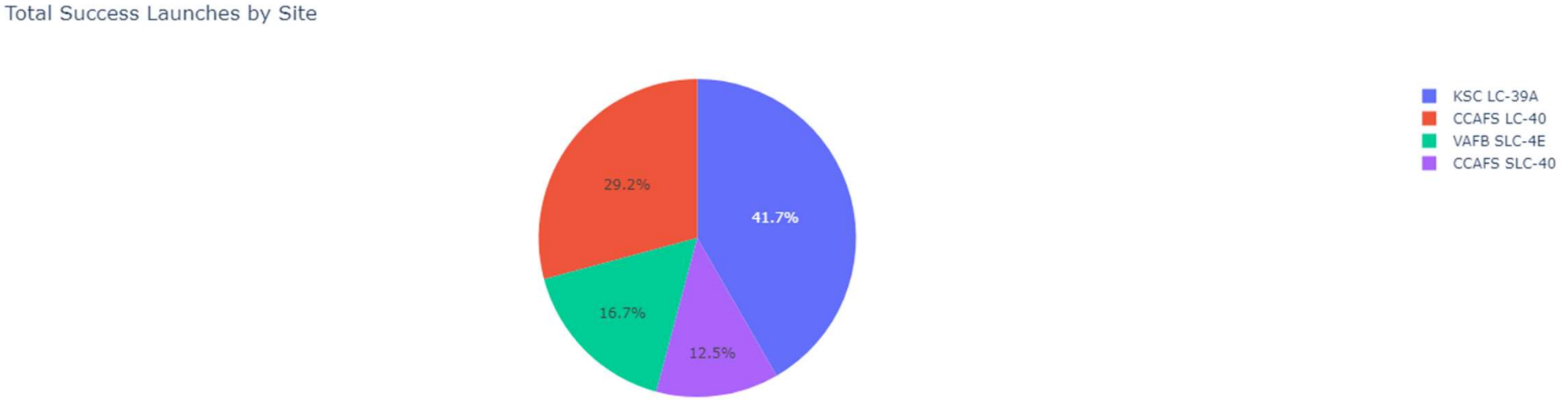
Key Location Proximities



Using KSC LC-39A as an example, launch sites are very close to railways for large part and supply transportation. Launch sites are close to highways for human and supply transport. Launch sites are also close to coasts and relatively far from cities so that launch failures can land in the sea to avoid rockets falling on densely populated areas.

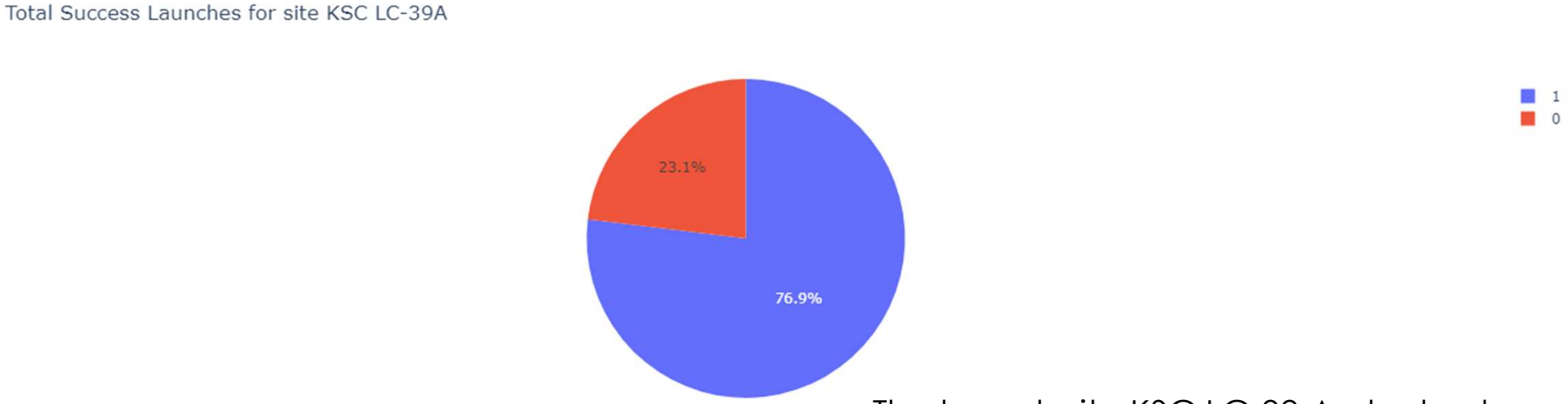
Section 5 Dashboard

Successful Launches Across Launch Sites



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

Successful Launches Across Launch Sites



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

Successful Launches Across Launch Sites

1

2

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

Accuracy Score and Best Score for each classification algorithm produces the following result:

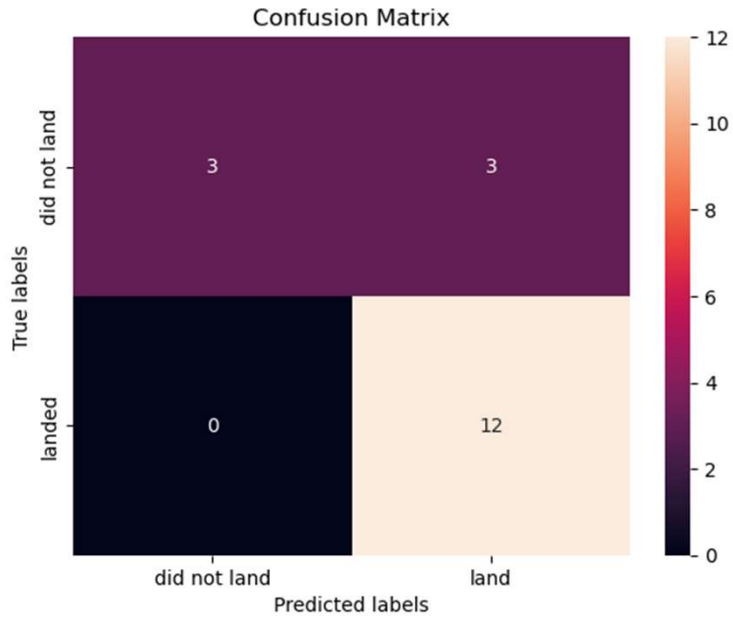
The KNN model has the highest classification accuracy:

- The Accuracy Score is 83.33%
- 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	0.666667	0.889286
3	K Nearest Neighbours	0.833333	0.903571

Confusion Matrix

- As shown previously, best performing classification model is the KNNmodel, with an accuracy of 83.33%.
- Correct predictions are on a diagonal from top left to bottom right.



Conclusions

- Our mission: Develop a machine learning model for Space Y, which wants to bid against SpaceX
- The goal of the model is to predict when the first stage will land successfully to save about $100 million
- Data used comes from public SpaceX API and web scraping SpaceX Wikipedia page
- Create data tags and store data into SQLite database
- Created a dashboard for the visualization
- We created a machine learning model with 83% accuracy
- SpaceY's Allon Mask can use this model to more accurately predict whether a launch will successfully land on Stage 1 before launch, so as to decide whether to launch
- If possible, more data should be collected to better identify the best machine learning model and improve accuracy

Appendix

Appendix

GitHub: IBM_Data_Science_Course

Thank You!