
# Winning Space Race with Data Science

Scott Kennedy
12 January 2024

# Outline

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

# Executive Summary

## Summary of methodologies

- This project looks to identify insights and analysis that will support prediction of successful 1st stage rocket landing. This was approached with the below methods:
  - **Collect** data using the SpaceX REST API and web scraping scripting
  - **Wrangle** data to clean and create success / fail outcome value per launch
  - **Explore** wrangled data via data visualization, focusing on characteristics : Payload mass, Launch site, number of flights, outcome trend over time
  - **Analyze** this data via SQL queries, calculating several statistics such as: total payload, payload range for launches, total successful and failed launch outcomes
  - **Determine** launch site success rates and proximity to geographical points f interest
  - **Visualize** launch site success rate and their performance in relation to payload mass
  - **Build** machine learning models to predict landing outcomes using logistic regression, support vectors machine (SVM), decision tree, and K-nearest neighbor (KNN) algorithms

## Summary of results

- Launch success has improved over time
- KSC LC-39A has the highest success rate among landing sites
- Orbit types ES -L1, GEO, HEO, and SSO had a 100% success rate, although with limited launches
- Launch sites reviewed are near the equator, and all are close to the coast
- All models performed similarly on the test set. The decision tree model slightly outperformed its peers

# Introduction

- Project Background

SpaceX, a leader in the space industry, strives to make space travel affordable for everyone. Its accomplishments include sending spacecraft to the international space station, launching a satellite constellation that provides internet access and sending manned missions to space.

SpaceX can do this because the rocket launches are relatively inexpensive ($62 million per launch) due to its novel reuse of the first stage of its Falcon 9 rocket. Other providers, which are not able to reuse the first stage, cost upwards of $165 million each.

By determining if the first stage will land, we can determine the price of the launch. To do this, we can use public data and machine learning models to predict whether SpaceX – or a competing company – can reuse the first stage.

- Problems to Explore
  - How payload mass, launch site, number of flights, and orbits affect first-stage landing success
  - Rate of successful landings over time
  - Best predictive model for successful landing (binary classification)

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

# Data Collection – SpaceX API

Utilize a public API to retrieve the required data and clean as necessary

- **Request Historical Data**:
  Send a GET message request to SpaceX API (rocket launch data)
- **Decode Response**: Decode using command ,json() and convert this into a Pandas dataframe via .json_normalize()
- **Manipulate data structure**: Create dictionaries and dataframes to then filter only on Falcon 9 launches
- **Request Additional Data:**
  Utilize additional API calls to retrieve launch outcomes, boosters, launch sites, and payload data for the filtered launches
- **Clean Data**: Replace missing values (Naan's / NULLs) found in Payload Mass with the calculated average via .mean()
- **Export Data** to CSV file use in next step in methodology of this project

# Data Collection – SpaceX API continued

- Supporting GitHub hosted notebook: [Coursera_Capstone/01_SpaceX_Data_Collection_API-SK.ipynb at main · skennedybda/Coursera_Capstone (github.com)]

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

response = requests.get(spacex_url)
```

```
# Takes the dataset and uses the rocket column to call the API and append the data to the list
def getBoosterVersion(data):
    for x in data['rocket']:
        if x:
            response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
            BoosterVersion.append(response['name'])
```

From the `launchpad` we would like to know the name of the launch site being used, the logitude, and the lati

```
# Takes the dataset and uses the launchpad column to call the API and append the data to the list
def getLaunchSite(data):
    for x in data['launchpad']:
        if x:
            response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
            Longitude.append(response['longitude'])
            Latitude.append(response['latitude'])
            LaunchSite.append(response['name'])
```

From the `payload` we would like to learn the mass of the payload and the orbit that it is going to.

```
# Takes the dataset and uses the payloads column to call the API and append the data to the lists
def getPayloadData(data):
    for load in data['payloads']:
        if load:
            response = requests.get("https://api.spacexdata.com/v4/payloads/"+load).json()
            PayloadMass.append(response['mass_kg'])
            Orbit.append(response['orbit'])
```

# Data Collection - Scraping

- Request Data from target Wikipedia webpage
- Create a BeautifulSoup object from the response
- Create column names into a list by extracting the response's HTML Table headers
- Collect data from parsing HTML tables in response
- Create dictionary from parsed data
- Create dataframe from this dictionary
- Export Data to CSV file

- Supporting GitHub hosted notebook:
  Coursera_Capstone/02_SpaceX_Web_Scraping-SK.ipynb at main · skennedybda/Coursera_Capstone (github.com)

# Data Wrangling

- Explore data and determine data labels
- Calculate:
  - Number of launches per site
  - Number and occurrences of orbit
  - Number and occurrence of mission outcomes for each orbit types
- Create landing outcome column by classifying bad outcomes (binary, success = 1, fail = 0)

- Supporting GitHub hosted notebook: Coursera_Capstone/03_SpaceX_Data_Wrangling-SK.ipynb at main · skennedybda/Coursera_Capstone (github.com)

# EDA with Data Visualization

- Created charts to explore insights for the below relationships in the data analyzed:
- Flight Number vs. Payload
- Flight Number vs. Launch Site
- Payload Mass vs. Launch Site
- Payload Mass vs. Orbit Type
- Types of charts used included scatter plots and line charts for trend/correlation analysis. Also utilized bar charts to show relationships between categorized attributes and measured values

- Supporting GitHub hosted notebook: Coursera_Capstone/05_SpaceX_EDA_Data_Visualization-SK.ipynb at main · skennedybda/Coursera_Capstone (github.com)

# EDA with SQL

## Queries

- Return list of launch sites named (unique, no duplicates)
- Return five records where their launch site begins with 'CCS'
- Return the total sum of payload mass carried by boosters launched by NASA (CRS)
- Return the average payload mass carried by booster version F9 v1.1
- Return the date of the first successful landing on a ground pad
- Return the list of boosters that had success landing with a drone ship and had a payload mass between 4,000 – 6,000 kg
- Return the total number of launches by mission outcome
- Return list of Booster versions that carried the largest payload mass
- Return list of failed landing outcomes via drone ship, including their booster version, launch site, month and year of event
- Return Count of landing outcomes between 04 June 2010 and 20 March 2020 in descending order

- Supporting GitHub hosted notebook:
  Coursera_Capstone/04_SpaceX_EDA_SQL-SK.ipynb at main · skennedybda/Coursera_Capstone (github.com)

# Build an Interactive Map with Folium

- Added markers to indicate the locations of the launch sites. Utilized popup labels, red circles to identify target launch sites
- Added colored markers for successful (green) and unsuccessful (red) launches for each site. This assisted in visualizing the success rate at each launch site reviewed
- Also added in colored lines to visualize distance between launch site CCAFS SLC-40 and nearest highway, railway, city, and coastline proximities

- Supporting GitHub hosted notebook: Coursera_Capstone/06_SpaceX_Interactive_Visual_Analytics_Folium-SK.ipynb at main · skennedybda/Coursera_Capstone (github.com)

# Build a Dashboard with Plotly Dash

- Created an interactive Pie and Scatter Plot dashboard.
- Visuals via the Pie chart can be seen for all sites and individually from a drop-down selection.
- Visuals via the Scatter Plot chart can be seen for all payload sizes or customized for a range via a slider scale.
- These visuals show the success rate of launch outcomes for sites, among peers and individually. They also show the launch outcomes per site based on payload size.

- Supporting GitHub hosted Python file:
  Coursera_Capstone/07_SpaceX_Interactive_Visual_Analytics_Plotly-SK.py at main · skennedybda/Coursera_Capstone (github.com)

# Predictive Analysis (Classification)

- Created a NumPy array from the dataset's Class column
- Standardized the dataset with StandardScaler to fit and transform the data
- Split the data to create a Training data set
- Created and Applied a GridSearchCV object on four different classification methods.
  - Logistic regression, Support Vect Machine (SVC), Decision Tree, and K-Nearest Neighbor
- Calculated the accuracy of each model on using the Training data
- Accessed the confusion matric of all models
- Identified the best model by analyzing the calculated Jaccard score, F1 score, and Accuracy statistics

- Supporting GitHub hosted notebook:
  Coursera_Capstone/08_SpaceX_Predictive_Analytics.ipynb-SK.ipynb at main · skennedybda/Coursera_Capstone (github.com)

# Results

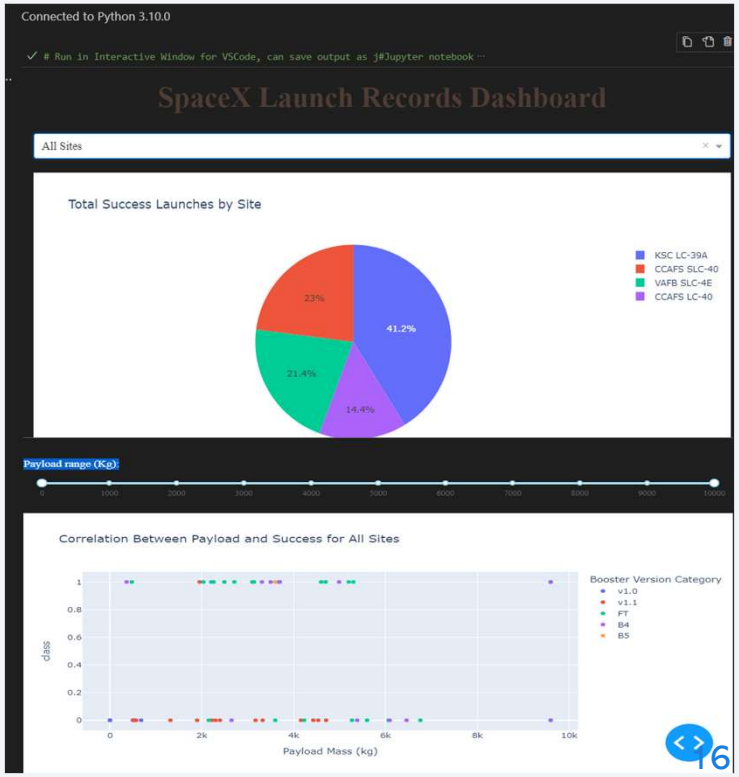
### Exploratory data analysis

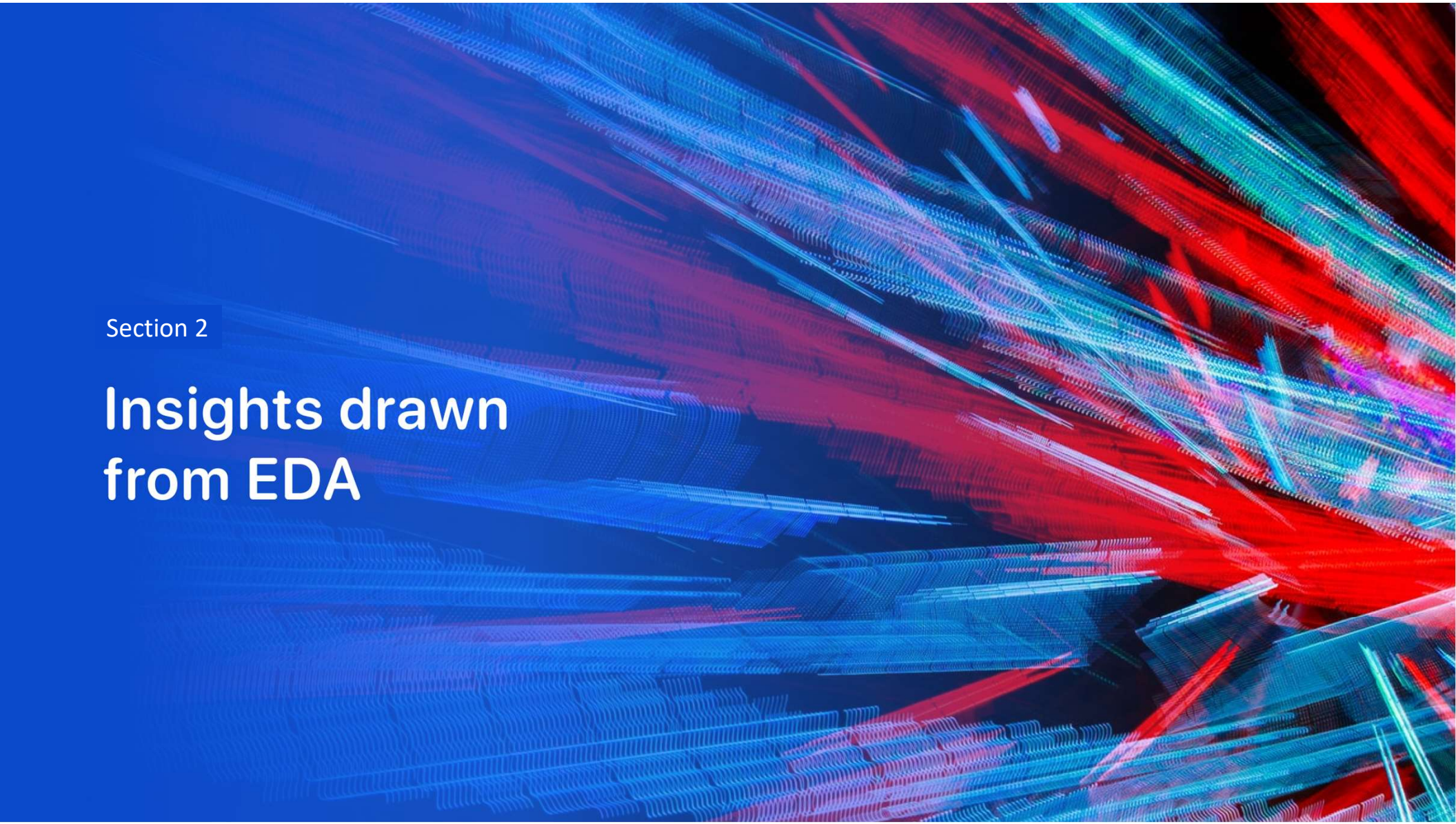
- Success rate of rocket launches improved with more attempts, even with higher payloads
- Launch site KSC LC-39A had the highest success rate among its peers
- Orbit types ES-LI, GEO, HEO, and SSO most successful with 100% success rate.
    - But these require additional launch data and analysis due to Orbit types ISS and GTO being the most widely used in launch attempts limiting launches for the above mentioned

### Predictive analysis results

- Decision Tree model was the best predictive model for the dataset utilized
- All predictive models analyzed returned the same level of accuracy, positively at 83.33%

### Interactive analytics demo in screenshots





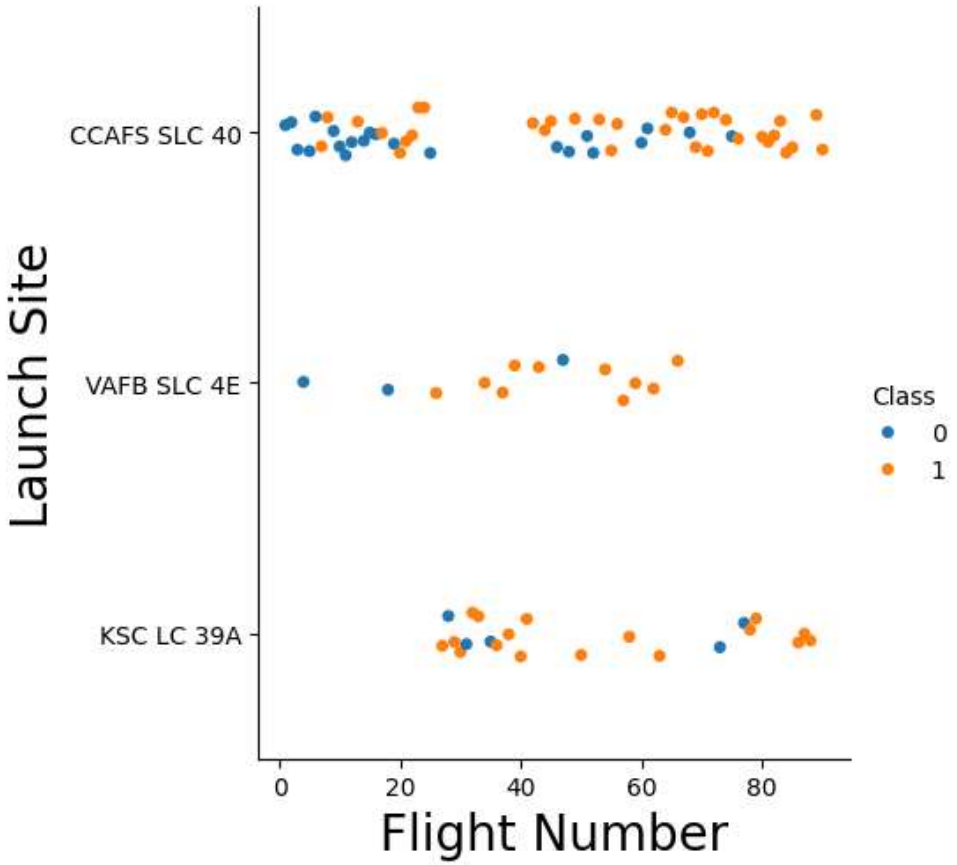
Section 2

# Insights drawn from EDA

# Flight Number vs. Launch Site

- It was found that generally more successful launches were seen when launch sites had higher numbers of flights
- Additionally, it is observed that a higher concentration of successful launches occurred during the later end of launch attempts per site.

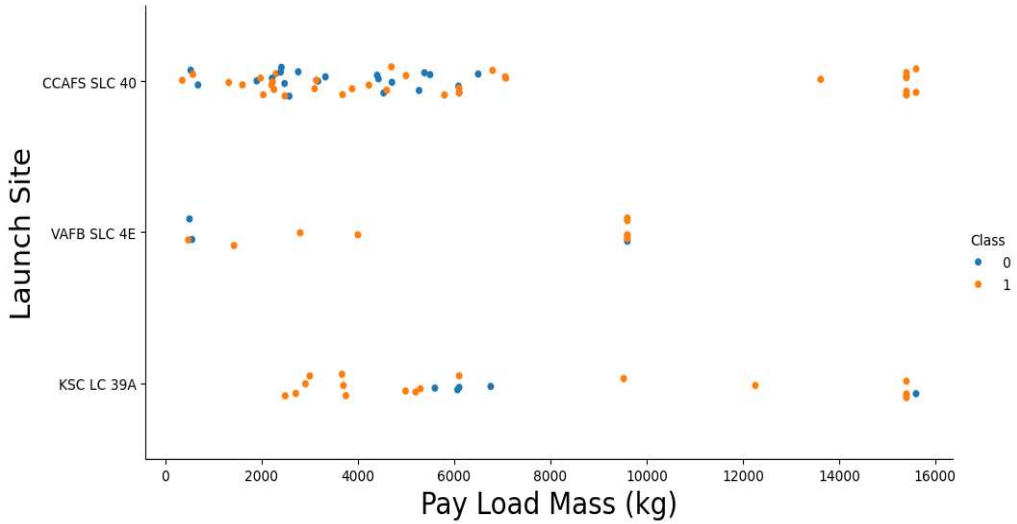
- In this chart Success = 1 via Class



# Payload vs. Launch Site

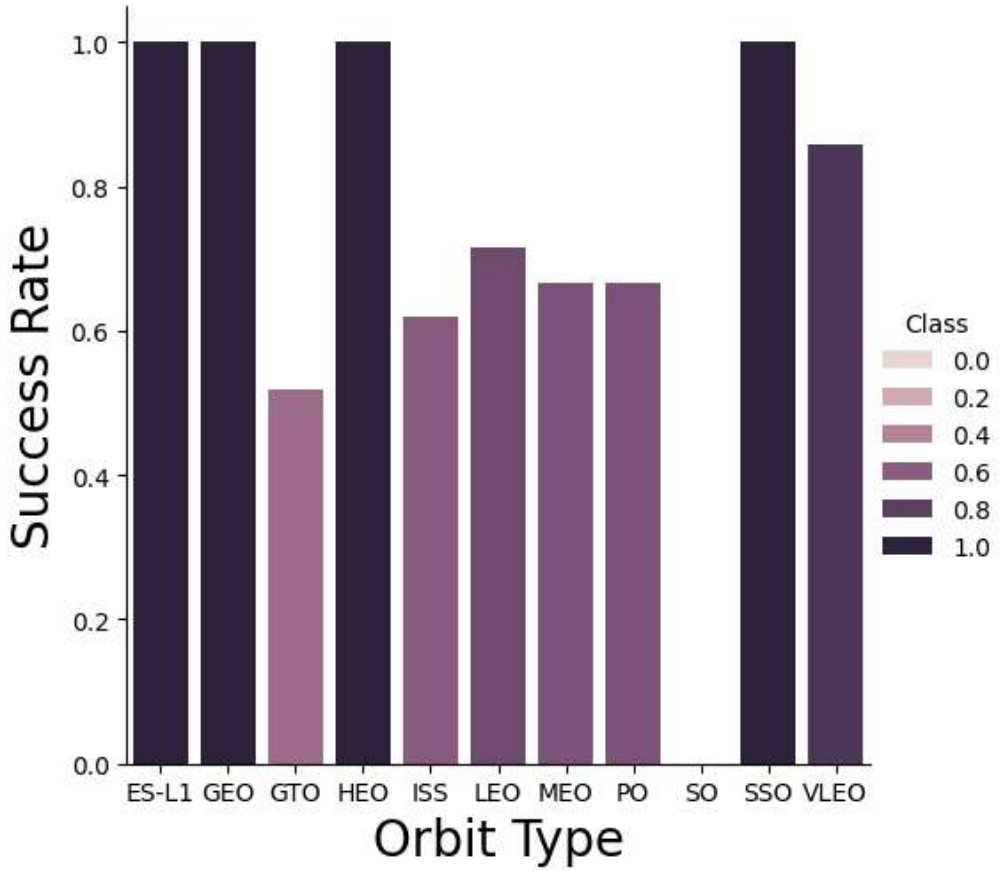
- On observation, higher payload launches resulted in a higher success rate.
- A majority of launches greater than 9,000 kg were successful.
- Launch site VAFB SLC 4E did not launch a payload greater than 10,000 kg, suggesting a capacity limit for the site

- In this chart Success = 1 via Class



# Success Rate vs. Orbit Type

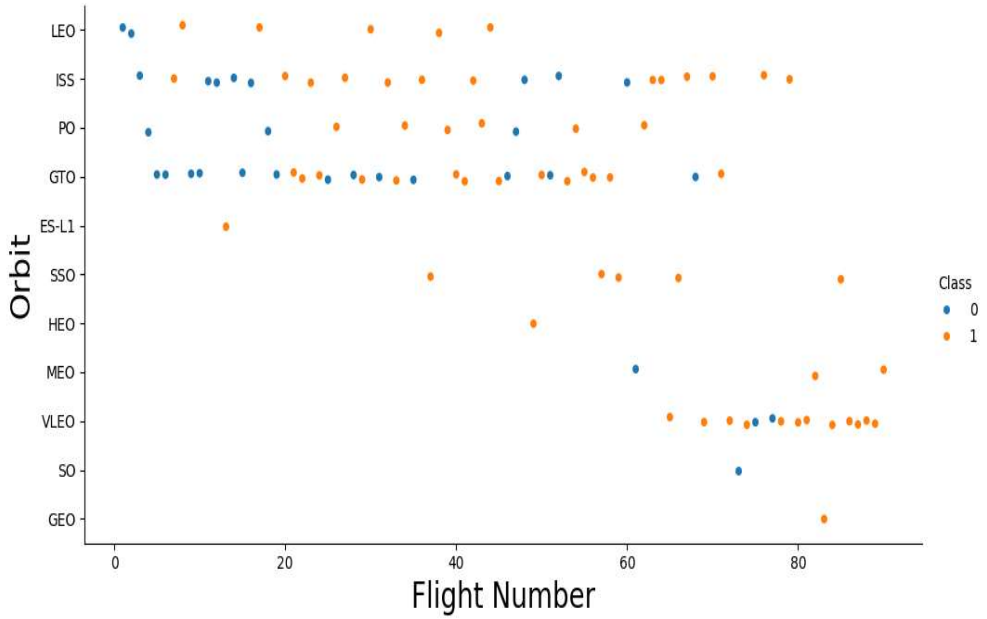
- Success Rate of 100% included Orbit Types of ES-L1, GEO, HEO, SSO
- Success Rate of 50 – 90% included GTO, ISS, LEO, MEO, PO
- Orbit Type of SO did not have a successful launch



# Flight Number vs. Orbit Type

- Viewing orbit type launch outcome by flight number shows that some orbit types were not tested as thoroughly as others.
- This is particularly seen in orbit types GEO and SO where each only had one launch attempt.

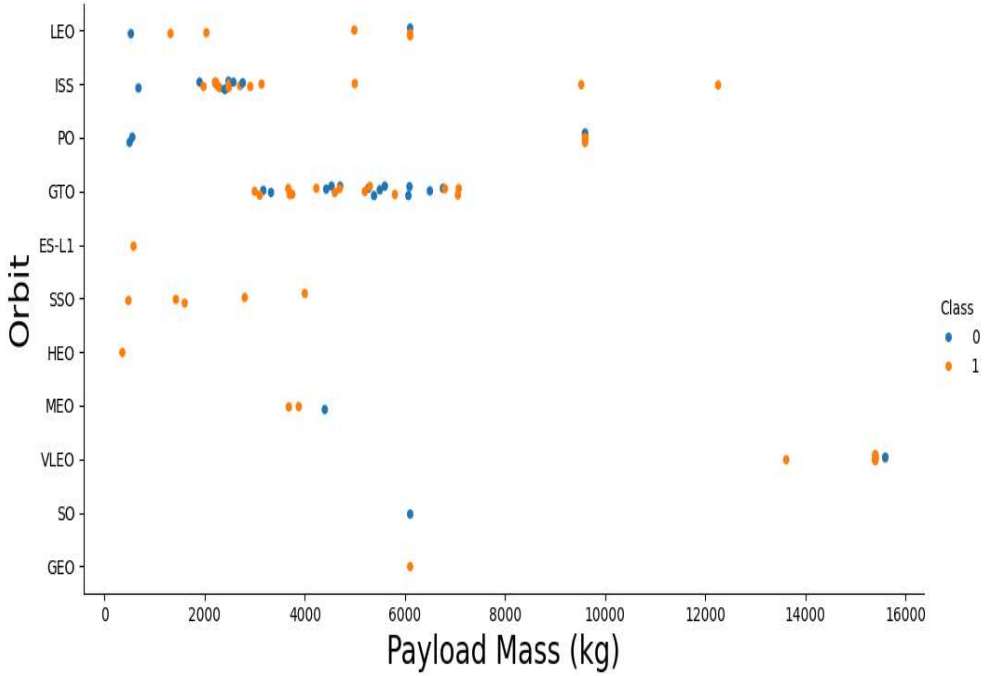
  This leads belief that there is not enough evidence to consider either orbit types as successful / inferior to other orbit types respectfully.
- Majority of launches utilized orbit types IS or GTO
- In this chart Success = 1 via Class



# Payload vs. Orbit Type

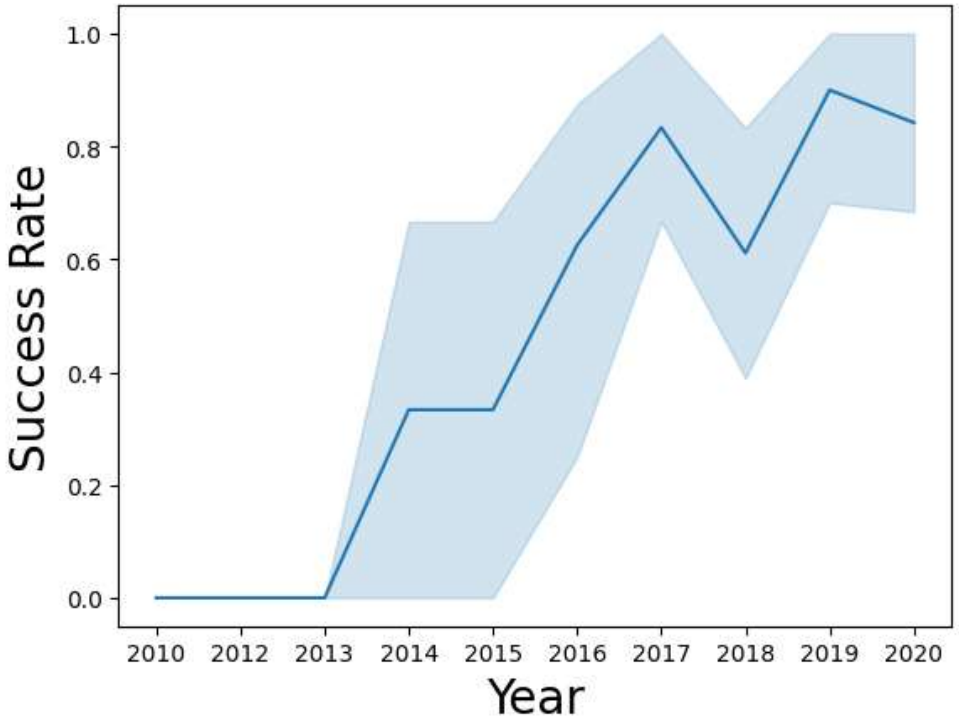
- Orbit Type VLEO was exclusively used for the highest payload launch attempts
  (above 13,000 kg)
- Majority of launch attempts utilized ISS or GTO, with observance that launch attempts concentrated within the 2,000 – 7,000 kg payload range.

  This supports the popularity, or prior establishment, of orbit types of ISS and GTO.
- It is interesting that while these orbit types had failed / no launch attempts at lower payload under 2,000 kg, orbit types with success at that level ES-L1, SSO, HEO were not utilized in higher payload launch attempts
- In this chart Success = 1 via Class



# Launch Success Yearly Trend

- The success rate of launch attempts is positive overall from year 2013 to 2020
- There was a sharp decline in performance between 2017 to 2018 but this was followed with an equal recovery in the following year



# All Launch Site Names

- Query:
  %sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTABLE

- Logic:
  Return a unique list of the different launch sites from the data set

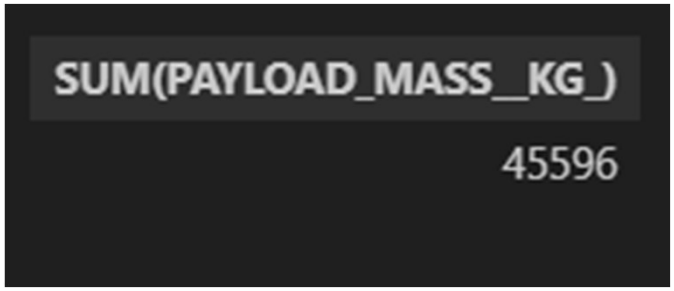
- Alternatively, this result can also be achieved with the below query:
- %sql SELECT LAUNCH_SITE FROM SPACEXTABLE GROUP BY LAUNCH_SITE

# Launch Site Names Begin with 'CCA'

- Query:
  %sql SELECT * FROM SPACEXTABLE WHERE LAUNCH_SITE LIKE ('CCA%') LIMIT 5
- Logic:
  Show up to five transactions that are have launch site name that begins with ‘CCA’

| Date | Time (UTC) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Orbit | Customer | Mission_Outcome | Landing_Outcome |
|---|---|---|---|---|---|---|---|---|---|
| 2010-06-04 | 18:45:00 | F9 v1.0 B0003 | CCAFS LC-40 | Dragon Spacecraft Qualification Unit | 0 | LEO | SpaceX | Success | Failure (parachute) |
| 2010-12-08 | 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 | 7:44:00 | F9 v1.0 B0005 | CCAFS LC-40 | Dragon demo flight C2 | 525 | LEO (ISS) | NASA (COTS) | Success | No attempt |
| 2012-10-08 | 0:35:00 | F9 v1.0 B0006 | CCAFS LC-40 | SpaceX CRS-1 | 500 | LEO (ISS) | NASA (CRS) | Success | No attempt |
| 2013-03-01 | 15:10:00 | F9 v1.0 B0007 | CCAFS LC-40 | SpaceX CRS-2 | 677 | LEO (ISS) | NASA (CRS) | Success | No attempt |

# Total Payload Mass




- Query:
  %sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE CUSTOMER = 'NASA (CRS)' AND Booster_Version IS NOT NULL AND Booster_Version != ‘’

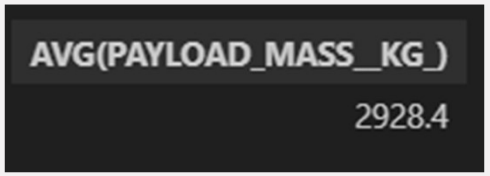
- Logic:
  Show the total amount of payload mass carried by launches using boosters from NASA (CRS).

  The additional where parameters on Booster_Version ensures that there is a booster assigned to the launches included in the summed total.

# Average Payload Mass by F9 v1.1

- Query:
  %sql SELECT
  AVG(PAYLOAD_MASS__KG_) FROM
  SPACEXTABLE WHERE
  BOOSTER_VERSION = 'F9 v1.1’

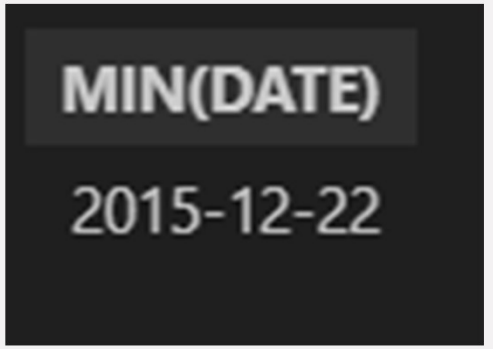
- Logic:
  Show the average payload mass for launches that used booster version ‘F9 v1.1’



# First Successful Ground Landing Date

- Query:
  %sql SELECT MIN(DATE) FROM SPACEXTABLE WHERE MISSION_OUTCOME = 'Success' AND LANDING_OUTCOME LIKE '%ground pad%’

- Logic:
  Show the first date of a successful mission outcome where the landing outcome was a Ground pad type landing

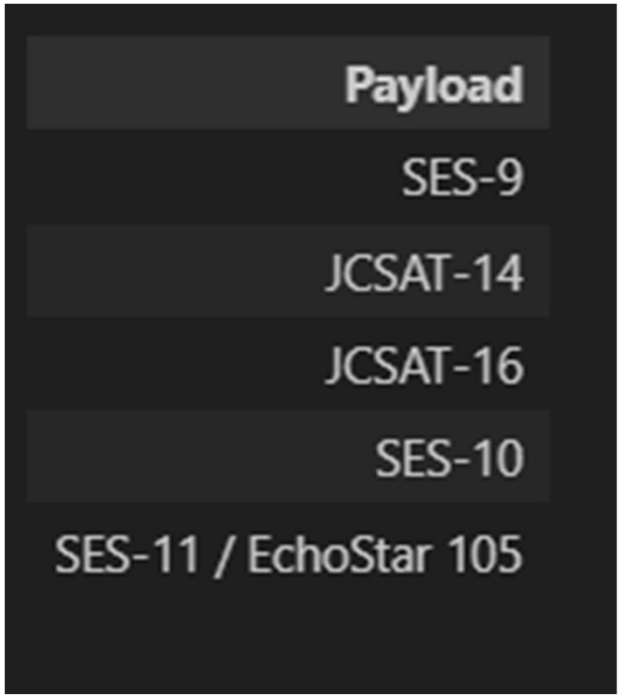


# Successful Drone Ship Landing with Payload between 4000 and 6000

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

- Present your query result with a short explanation here

- %sql SELECT PAYLOAD FROM SPACEXTABLE WHERE MISSION_OUTCOME = 'Success' AND LANDING_OUTCOME LIKE '%drone ship%' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000



# Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here
- %sql SELECT MISSION_OUTCOME, COUNT(*) as 'total number' FROM SPACEXTABLE GROUP BY MISSION_OUTCOME

| Mission Outcome | total number |
|---|---|
| Failure (in flight) | 1 |
| Success | 99 |
| Success (payload status unclear) | 1 |

# Boosters that Carried Maximum Payload

- Query:
  %sql SELECT BOOSTER_VERSION FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)

- Logic:
  Show the booster versions that were used to carry the largest payload mass recorded in the supporting dataset

| 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

- Query:
  %sql SELECT SUBSTR(Date,6,2) as MONTH, DATE, BOOSTER_VERSION, LAUNCH_SITE, LANDING_OUTCOME FROM SPACEXTABLE WHERE LANDING_OUTCOME = 'Failure (drone ship)' and SUBSTR(Date,0,5)='2015’

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

- Logic:
  Output the launches where the landing outcome was a failure by drone ship. In the results presented, show the month, full date, booster version used, launch site and landing outcome type

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

- Query:
  %sql SELECT LANDING_OUTCOME, COUNT(*) as count_outcomes FROM SPACEXTABLE WHERE DATE BETWEEN '2010-06-04' and '2017-03-20' GROUP BY LANDING_OUTCOME ORDER BY count_outcomes DESC

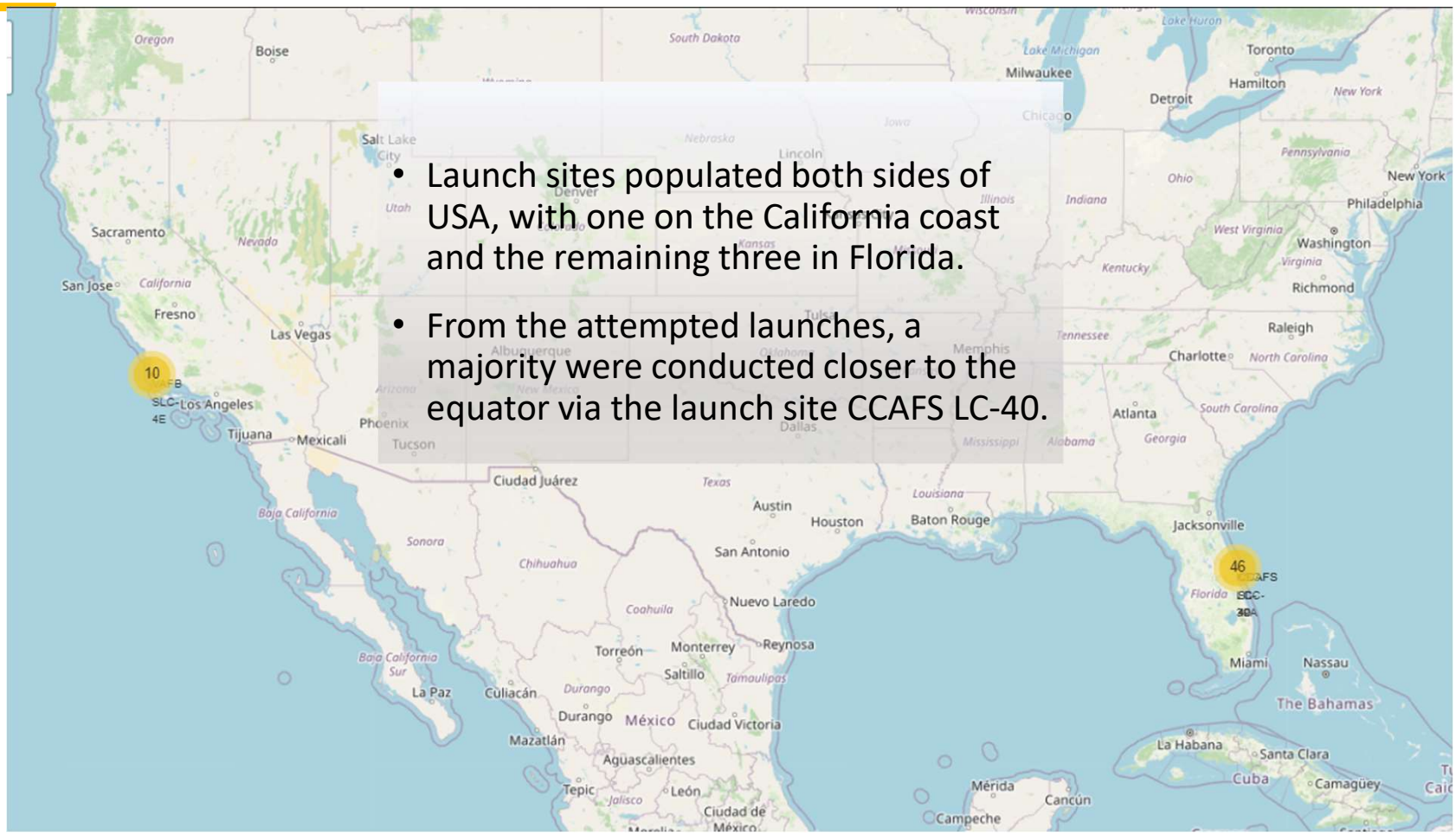
- Logic:
  Output the amount of launch attempts by each type of landing outcome, between the dates of 04 June 2010 and 20 March 2017, ordering / sorting the result to view the landing outcomes by descending order in count value.

| Landing_Outcome | count_outcomes |
|---|---|
| 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 |

Section 3

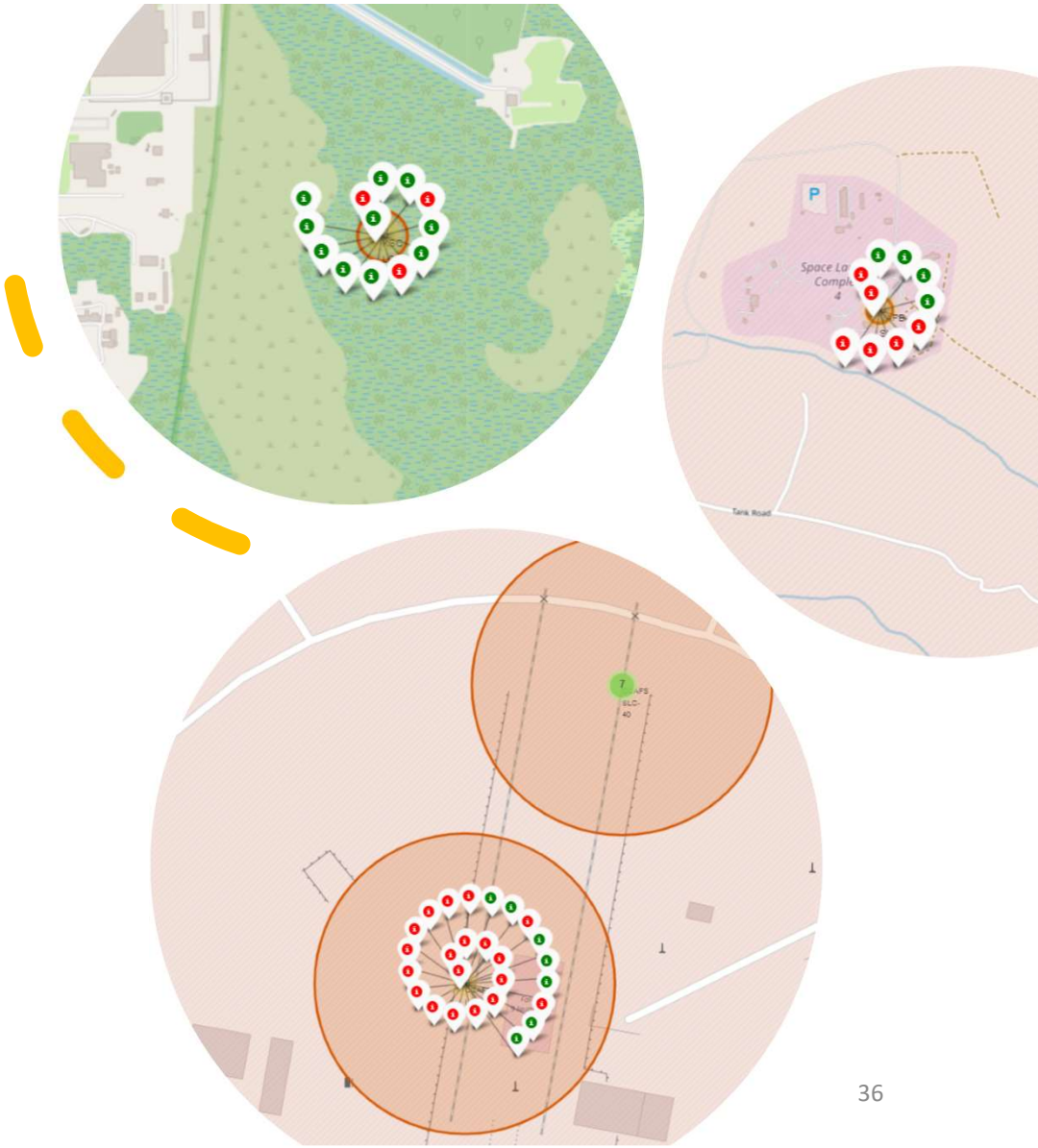
# Launch Sites Proximities Analysis

# Launch Sites - Locations



# Launch Outcomes

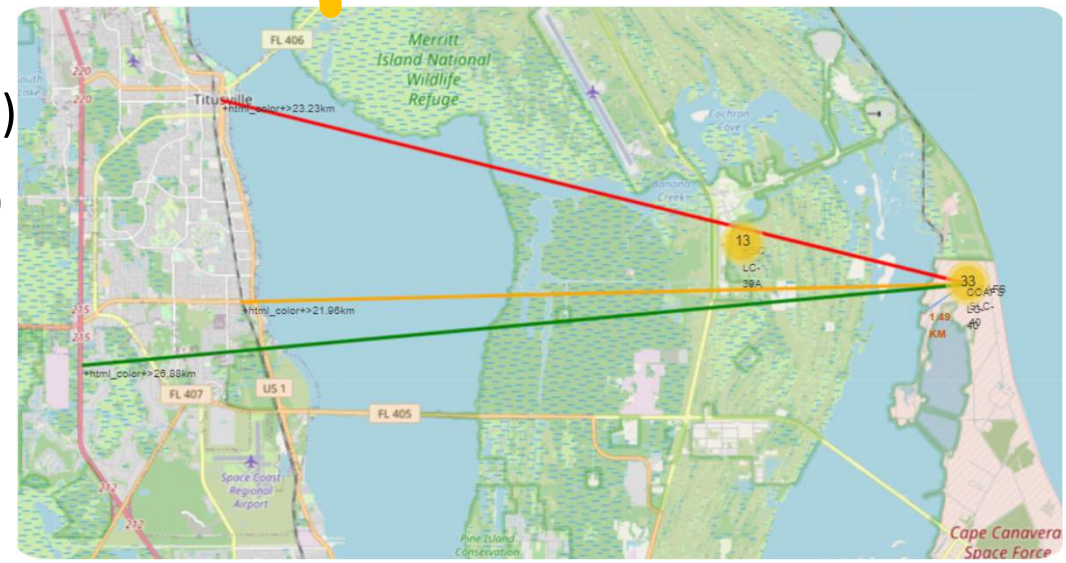
- On the map per site, launch sites are identified by red map circles.
- Launch outcomes are labelled green = successful, red = failed

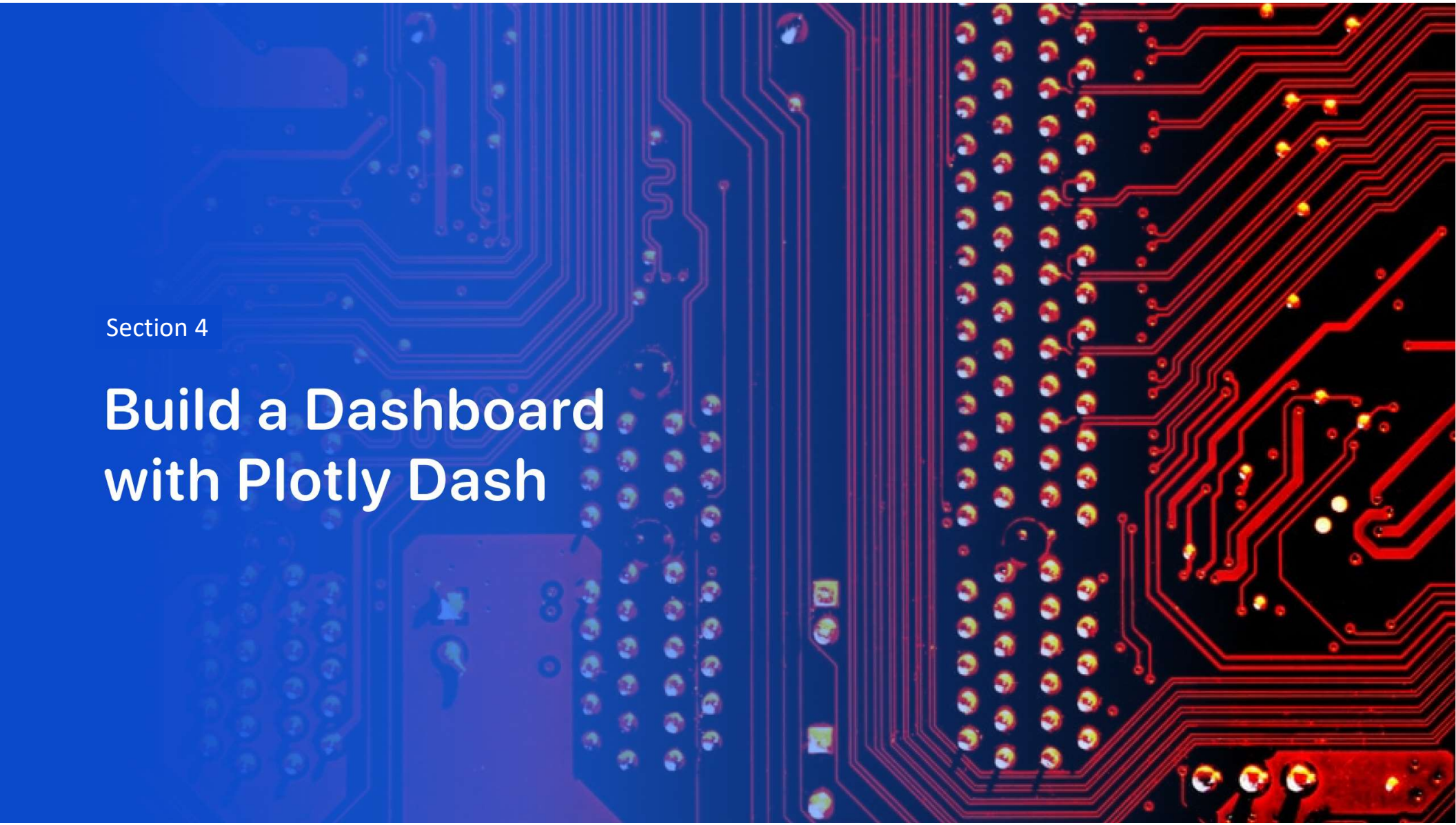


# Launch Site
# Distance to Proximities

CCAFS SLC-40

- 1.49 km from nearest coastline (blue)
- 21.96 km from nearest railway (gold)
- 23.23 km from nearest city (red)
- 26.88 km from nearest highway (green)



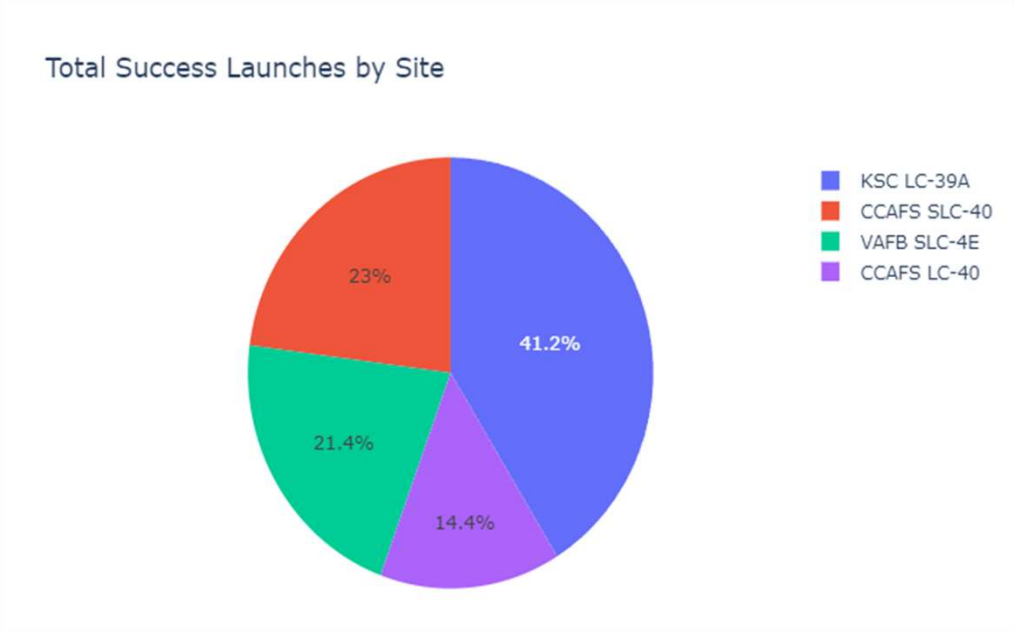


Section 4

# Build a Dashboard with Plotly Dash

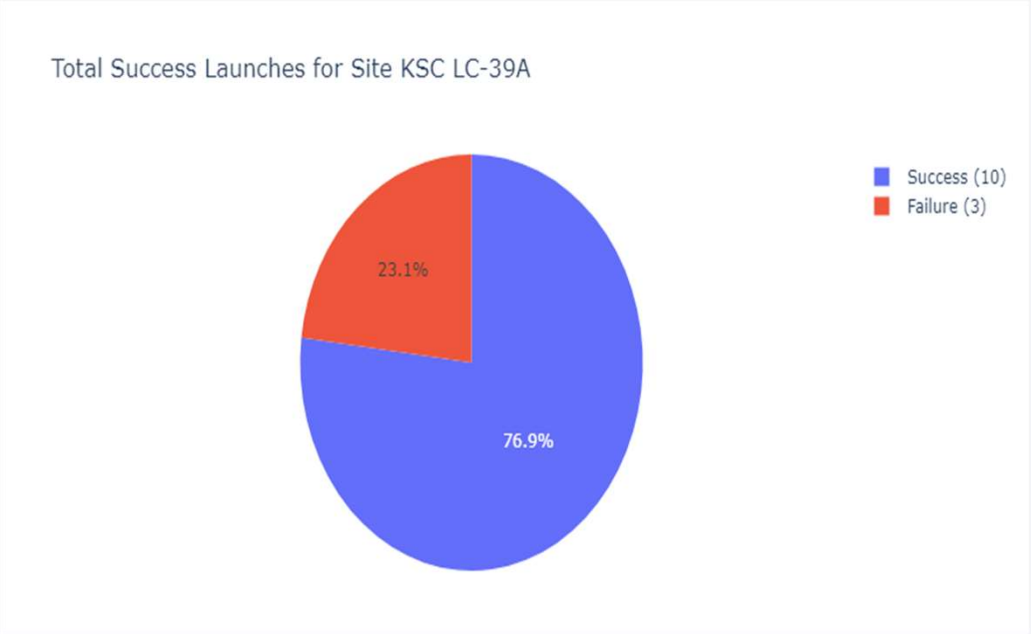
# Breakdown of Launch Success per Site

- **KSC LC-39A** had the **highest proportion of successful launches** among its peers
- The site recorded **41.2%** of successful launches across all launch sites



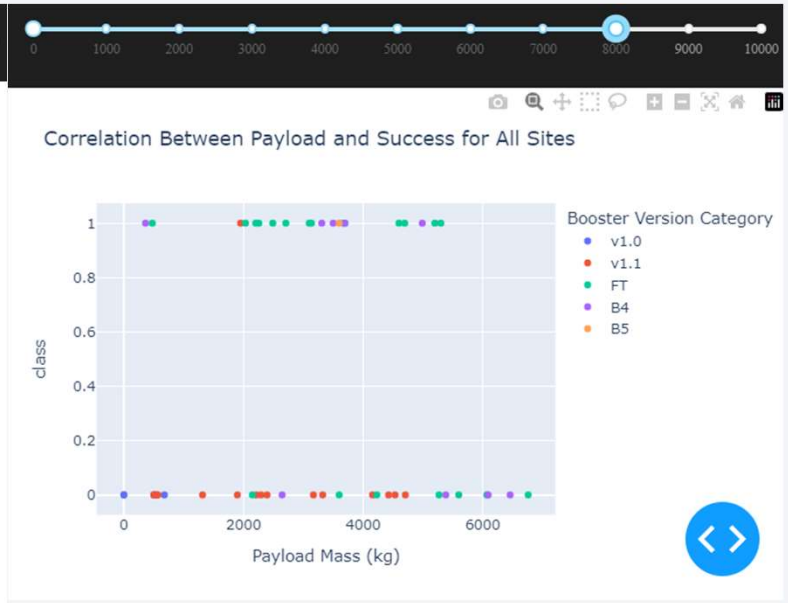
# Success Ratio of Launch Site KSC LC-39A

- This chart displays the success ratio of site KSC LC-39A over all its launch attempts.
- With the legend presenting Successful launches totaling 10 out of the total 13 launches executed.



# Effect of Payloads on Launch Outcomes

- Looking deeper into the scale of payload and its affect on launch outcomes, we can see an observance that smaller payloads have more successful launches
- In these screenshots, successful launches are represented by class = 1

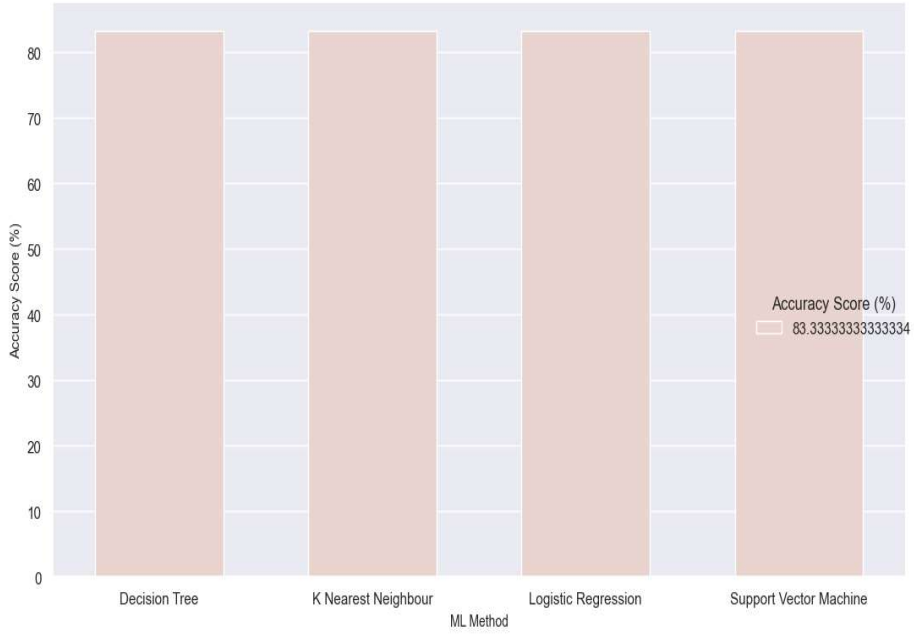



Section 5

# Predictive Analysis (Classification)

# Classification Accuracy

- It was found that each classification model calculated with the same accuracy score of 83.33%
- This is likely due to the limited size of the data set used for the models
- The best model identified was the Decision Tree model




| | LogReg | SVM | Tree | KNN |
|---|---|---|---|---|
| Jaccard_Score | 0.800000 | 0.800000 | 0.800000 | 0.800000 |
| F1_Score | 0.888889 | 0.888889 | 0.888889 | 0.888889 |
| Accuracy | 0.833333 | 0.833333 | 0.833333 | 0.833333 |

```
models = {'KNeighbors':knn_cv.best_score_,
          'DecisionTree':tree_cv.best_score_,
          'LogisticRegression':logreg_cv.best_score_,
          'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

[55] ✓ 0.0s

```
Best model is DecisionTree with a score of 0.8767857142857143
Best params is : {'criterion': 'gini', 'max_depth': 8, 'max_features': 'sqrt', 'mir
```

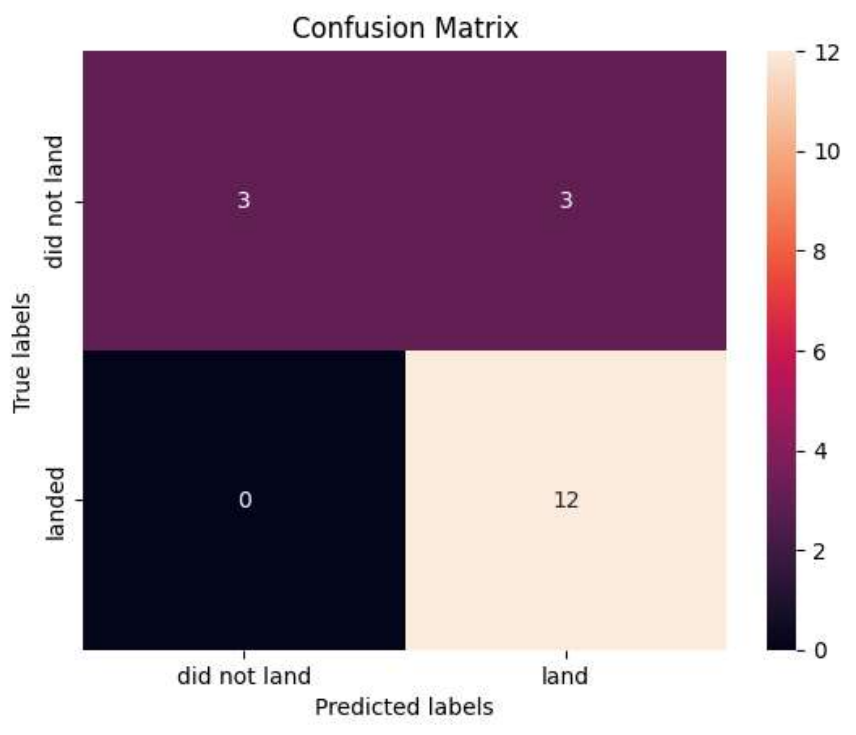
# Confusion Matrix

Best performing model was the Decision Tree algorithm

- A confusion matrix summarizes the performance of a classification algorithm's output, where correct and incorrect (false) predictions are identified.
- There is are false positives, which is not ideal.
  However, the accuracy of this model matches the other models tested.

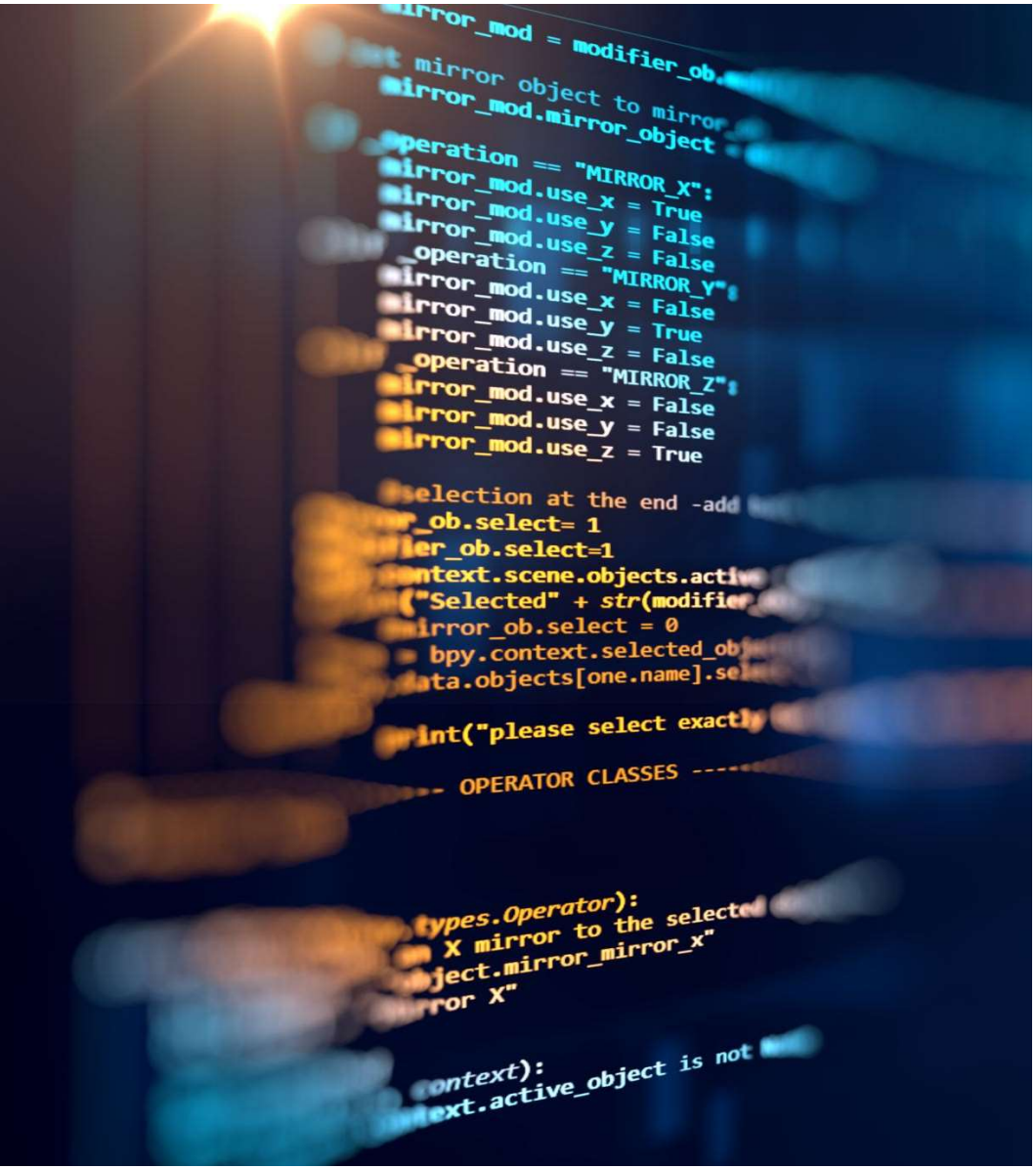
**Confusion Matrix Outputs**:

- 9 True positive, 5 True negative, 1 False positive, 3 False Negatives
- Precision = TP / (TP + FP) = 12 / (12 + 3) = 80%
- Recall = TP / (TP + FN) = 12 / (12 + 0) = 100%
- F1 Score = (2 * Precision * Recall) / (Precision + Recall)
  (2 * 0.80 * 1.00) / (0.80 + 1.00) = 88.88%
- Accuracy = (TP + TN) / (TP + TN + FP + FN) =
  (12 + 3) / (12 + 3 + 3 + 0) = 83.33%



# Conclusions

- This project provides interesting initial insights into pursuit of predicting launch outcomes
- This insight should take the below factors into consideration:
- Data set was limited; a larger dataset would allow there to be more differentiation in accuracy performance for the models' prediction training.
- There are likely additional factors and characteristics that influenced most flights to utilize the same launch site and two orbit types.
- A qualitative exploration of this could be warranted to expose any bias that could affect the data being accepted for appropriate training of the models.




# Appendix

- For reference, I made effort to utilize a local environment for the coding files utilized in this Capstone project. Using VS Code as the IDE, I set up the latest Python interpreter version (3.10) and installed the latest package versions and their dependencies
- This disrupted some import code lines for a few of the labs, requiring some research to rebuild the code and confirm that the labs' directional integrity were not disrupted by my output after committing to such changes
- Special thanks to the authors, supporting professionals and peers that have provided feedback and improvements to this course
- Thank you!

# Thank you!