IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Damian Mulvena
2025-03-06

Outline

Executive Summary

- Methodologies
  - Public data concerning SpaceX launches, components, reuse and reliability was captured
  - This was examined through graphic and query analysis, for insight into important factors
  - Finally, the data was transformed into Machine Learning predictive models to identify whether rocket reuse could be feasible
- Summary
  - Factors were examined, such as Launch Sites, intended Orbits, Payload mass and Booster version.
  - Progress over time was also examined, whether as “Flight No.” or Yearly-trend, where it could be seen that recovery rates increased over time
- Outcome
  - There is every expectation that SpaceY will be able to reuse the first-stage booster rockets, reducing the launch costs and allowing successful competition with SpaceX

Introduction

- As a data scientist in SpaceY, my objective is to determine how this company can compete with SpaceX in the space transport and delivery market
- One of the key determining factors is whether we can reuse the first stage of the rockets, as this is where most of the work is done in a launch
  - "If we can determine if the first stage will land, we can determine the cost of a launch"
- We need to answer:
  - What is the likelihood of recovering the first stage rocket after a launch?
  - What factors contribute to the success or failure of recovery?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was retrieved from public websites of SpaceX launch data, one an API and the other Wikipedia
- Perform data wrangling
  - Categorical data was selected and converted to binary using “one-hot” encoding
  - A new “class” column was created to reflect successful recovery of the first stage booster
- Perform exploratory data analysis (EDA) using visualization and SQL
  - SQL queries and some plotting were used to identify features that might be modelled to show costs
- Perform interactive visual analytics using Folium and Plotly Dash
  - Geo-plotting of site information and interactive dashboards to explore weight and sites were created
- Perform predictive analysis using classification models
  - GridSearchCV was used to select and tune hyperparameters, to identify the best kind of prediction model

Data Collection Overview

- Primary data was SpaceX API collection, which collates public details of Space-X launches
  - as maintained and documented in https://github.com/r-spacex/SpaceX-API
  - and accessed via http API requests at https://api.spacexdata.com/v4/launches/latest
- Further data on Space-X launches was "scraped" from Wikipedia tables
  - from page https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches
  - which is accessed by Web-Scraping methods, using BeautifulSoup library
- In each case, the data used was in fact a snapshot, selected for course consistency

Data Collection – SpaceX API

- Primary data collection involved making http request "Get" calls, to an API interface at api.spacexdata.com
- Part of the process involved identifying id values within the initially loaded launch data, and using these to lookup additional details
- the GitHub URL of the completed SpaceX API calls notebook is
  - [DSCapstone/jupyter-local-1.1a-spacex-data-collection-api-v2.ipynb at main · damianmulvena/DSCapstone]

Retrieve data
- make API request, to
  - base URL: api.spacexdata.com/v4/
  - API: launches

Reduce
- to selected columns only
- rows up to 2020-01-13

Enrich
- Loop over rows, making API calls on ids as below
  - rockets/id, launchpads/id, payloads/id, cores/id

Wrap up
- combine enrichment arrays to create new dataframe
- drop Falcon-1 rows, and re-label flight numbers according to those rows that remain
- replace missing values in payload column with mean

Data Collection – Scraping

- In web-scraping we retrieve ("request") an html page and parse it to extract relevant details, using the BeautifulSoup library to identify and capture data elements of interest
- the GitHub URL of the completed web-scraping notebook is
  - [DSCapstone/jupyter-local-1.1b-webscraping.ipynb at main · damianmulvena/DSCapstone]()

Get Wiki page
- Retrieve ("Request") wiki page, in this case from a specific archived copy

Identify data
- Find table structures, and select the 3rd one
- Identify table rows having a Flight Number

"Scrape"
- From each row, extract useful data to list

Data object
- combine lists to a dictionary, and then
- convert dictionary to a dataframe

Data Wrangling

- Using dataset_part_1.csv, (nominally) from the API retrieval, we explored some aspects of the data
- Missing values
  - Report missing values as percentage
  - Only LandingPad had missing values, and for now this is expected and valid
- Explored
  - Looked at number of launches by launch site
  - Looked at number of launches for each type of orbit
  - Looked at number of launches with each type of outcome
- Added column to reflect outcomes
  - Created new column “class” to reflect successful or otherwise first stage rocket recovery
- the GitHub URL of the completed data-wrangling notebook is
  - [DSCapstone/jupyter-local-1.2-spacex-data-wrangling-v2.ipynb at main · damianmulvena/DSCapstone]()

EDA with Data Visualization

- Initially, scatterplots were created of Payload, FlightNo and Site combinations, in each case colouring the points by class (success):
  - Payload vs FlightNo (showing fewer recovery failures in later flights and heavier payloads)
  - Site vs FlightNo (showing fewer failures in later flights, but more for CCAFS-SLC-40), and
  - Site vs Payload (which backed up findings from previous 2 plots)
- Next plots looked at Orbit, finding that some orbit destinations had fewer fails:
  - Orbit vs average Class (bar, showing 100% success for 4 orbits)
  - Orbit vs FlightNo (reaffirming later flights as higher success in recovery)
  - Orbit vs Payload (reaffirming heavier payloads as higher success in recovery)
- The last chart (AvgClass by Year) showed a steady rise in success over time
- Finally, discrete-value columns were transformed into categorical columns using one-hot encoding
- the GitHub URL of the completed EDA with data visualisation notebook is
  - DSCapstone/jupyter-local-2.2-eda-dataviz-v2.ipynb at main · damianmulvena/DSCapstone

EDA with SQL

- SQL queries performed in analysis of launch data:
  - Identify unique launch site names from the space missions
  - Show 5 launch_site records, from those with site-names beginning with “CCA” (Cape Canaveral)
  - Calculate total payload mass for boosters launched by “NASA (CRS)”
  - Calculate average payload mass for booster versions with “F9 v1.1”
  - Identify date of the first successful landing to a ground pad
  - List the boosters which landed to a drone ship with payload between 4000 and 6000
  - Give the total mission counts by successful and unsuccessful recovery
  - List the booster_versions which have carried the maximum payload
  - Display month name, landing_outcome, booster version, and launch_site, for launches in 2015 having a failed drone ship landing outcome
  - Rank, in descending order of count, the counts for different landing outcomes, between 2010-06-04 and 2017-03-20
- the GitHub URL of the completed EDA with SQL notebook is
  - DSCapstone/jupyter-local-2.1-eda-sql-coursera_sqllite.ipynb at main · damianmulvena/DSCapstone

Build an Interactive Map with Folium

- Using the launch data and mapping as provided with Folium library, I created:
  - Circle markers, with name popup (i.e. when clicked) to help identify or locate each of the sites
  - Map "DivIcon" Markers which placed a visible site name on the map at coordinates (no specific icon!)
  - A single marker cluster was added to the map, which contained
    - white Icon markers for each launch, with the icon core coloured green or red to denote success or failure
    - and with each icon linked to others of any given site by its latitude and longitude coordinates
  - A mouse pointer object was dropped to the map to help finding coordinates of a point
  - Nearest coastline (railway, road) location markers were created and labelled with distance to nearest site
  - PolyLine markers were then created joining the utility access points to the sites

- Explain why you added those objects
  - Launch site and success/failure markers help us to understand site and success level
  - Access to services is important in determining the cost of running a site, so mapping these is important

- the GitHub URL of the completed Folium mapping notebook is
  - DSCapstone/jupyter-local-3.1-launch-site-location-v2.ipynb at main · damianmulvena/DSCapstone

Build a Dashboard with Plotly Dash

- The graphs and user-interaction controls are as follows:
  - A dropdown allows user to select either ALL sites, or a specific site
  - A RangeSlider control lets user control the payload range of interest in the 2^{nd} graph
  - The first plot is a pie-chart, showing:
    - Total (count) of success, by site, across all sites (key is site-name), or
    - Proportion of success to failure for the selected site (key is success/fail)
  - The lower plot is a scatterplot of success (class) vs Payload Mass (key by Booster Version), whether filtered by sitename, or for all sites
- These plots and controls allow user to more closely examine
  - Success vs Failure for different sites, payload mass ranges, or booster versions
- the GitHub URL of your completed Plotly Dash lab is:
  - DSCapstone/spacex_dash_app.py at main · damianmulvena/DSCapstone

Predictive Analysis (Data Preparation)

- Data for the classification had been prepared in earlier steps
  - The "feature" or X data variables was drawn from dataset_part_3.csv
    - This data used "categorical" (or class) variables which had been converted to binary columns/variables using the "One-hot encoding" method
    - NB. This dataset did not contain the "target" variable "class"
  - The labelled "target" or "classification" (Y) data variable uses the "class" column, as drawn from dataset_part_2.csv - this flag reflects success in recovering the first stage rocket for reuse
  - Finally, the X and Y data sets were split into training and testing (20%) subsets, using the train_test_split function

Predictive Analysis (Classification)

- Different classification models were evaluated using GridSearchCV, a Cross Validation tool which checks a range of parameters for different estimators, using the same data, returning the best parameters and a score
  - For each of 4 estimators – Logistic Regression, SVM (Support Vector Machine), Decision Tree Classifier and KNN (K-Nearest Neighbours) the following steps were taken:
    - The parameters to be compared for the estimator were selected, according to estimator
    - A new estimator object was created
    - GridSearchCV was called, passing it Parameter options, estimator object, scoring method and Cross Validation specification. GridSearchCV processed through multiple parameter options to determine "best"
    - Fit the data (using X_train, Y_train), returning a new predictor_cv object for the results
    - Score the results, using X_test, Y_test
    - Create a prediction yhat using X_test, and plot a confusion matrix of Y_test vs yhat
- the GitHub URL of the completed predictive analysis lab is:
  - DSCapstone/SpaceX-Machine-Learning-Prediction-Part-5-v1-log2.ipynb at main · damianmulvena/DSCapstone

Results

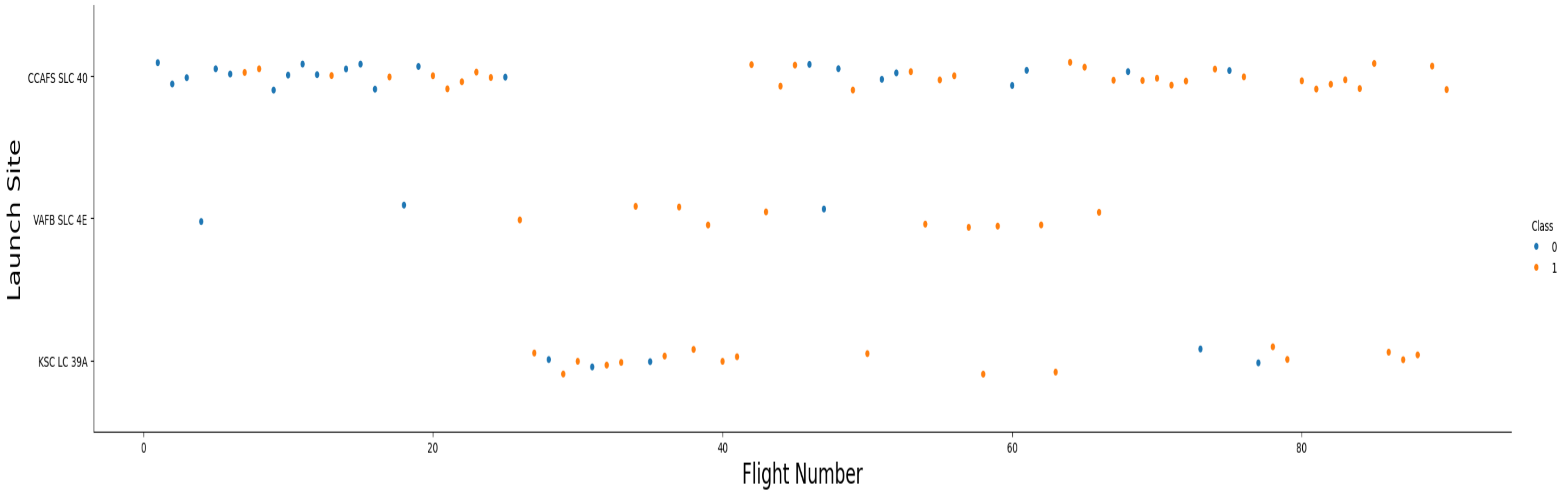
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

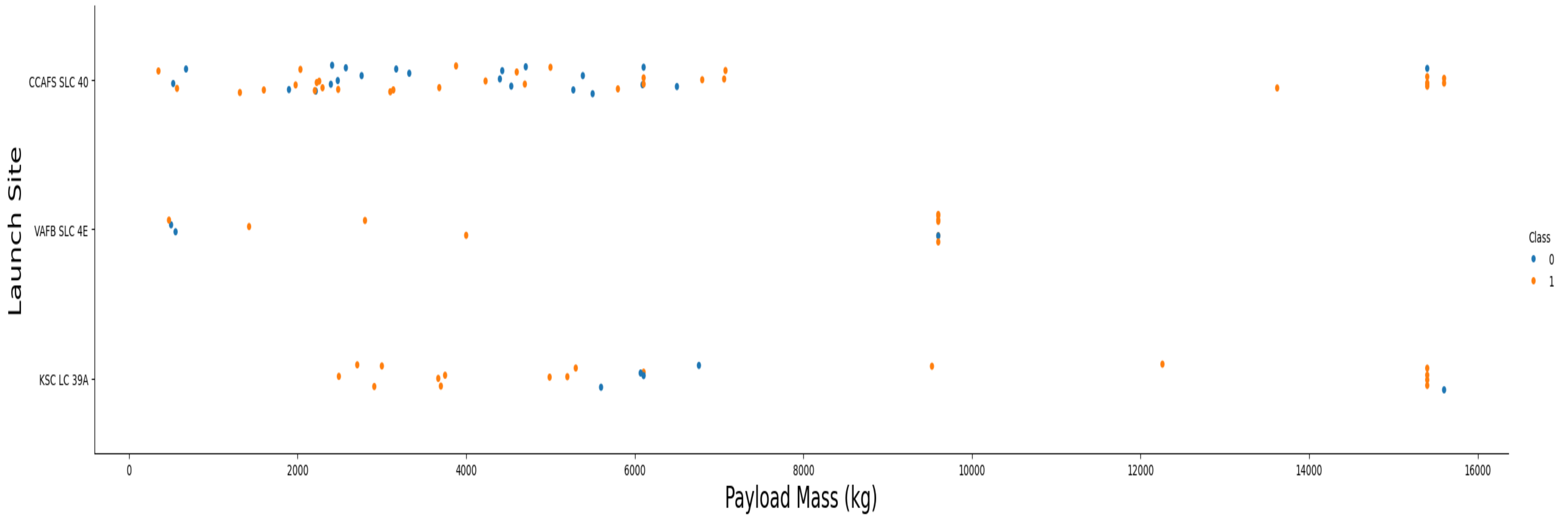
Flight Number vs. Launch Site

- A plot of Flight Number vs. Launch Site, coloured by rocket recovery success or failure, shows that many of the failures occur for earlier flights



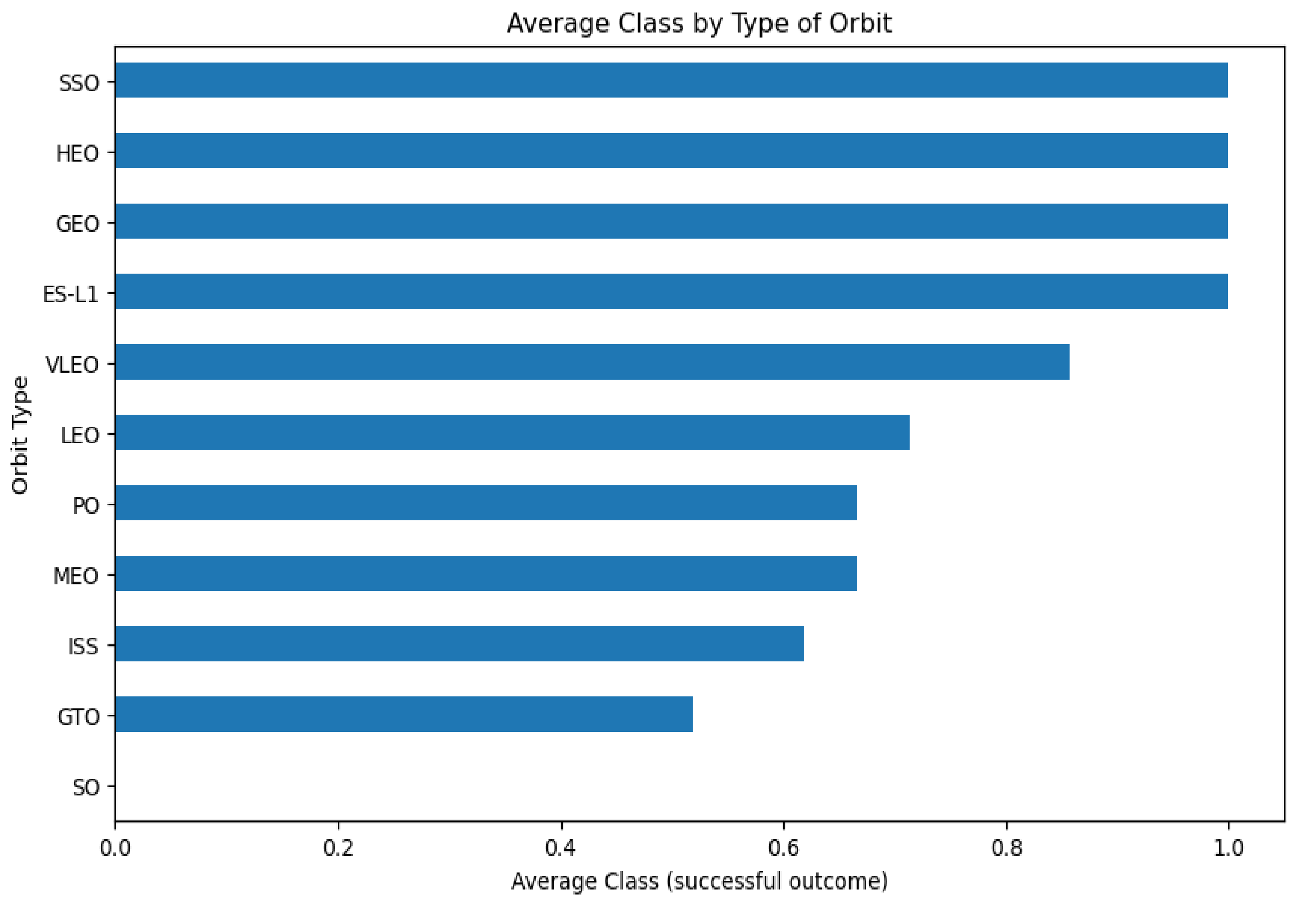
Payload vs. Launch Site

- A plot of Payload vs. Launch Site shows higher recovery for greater payload mass



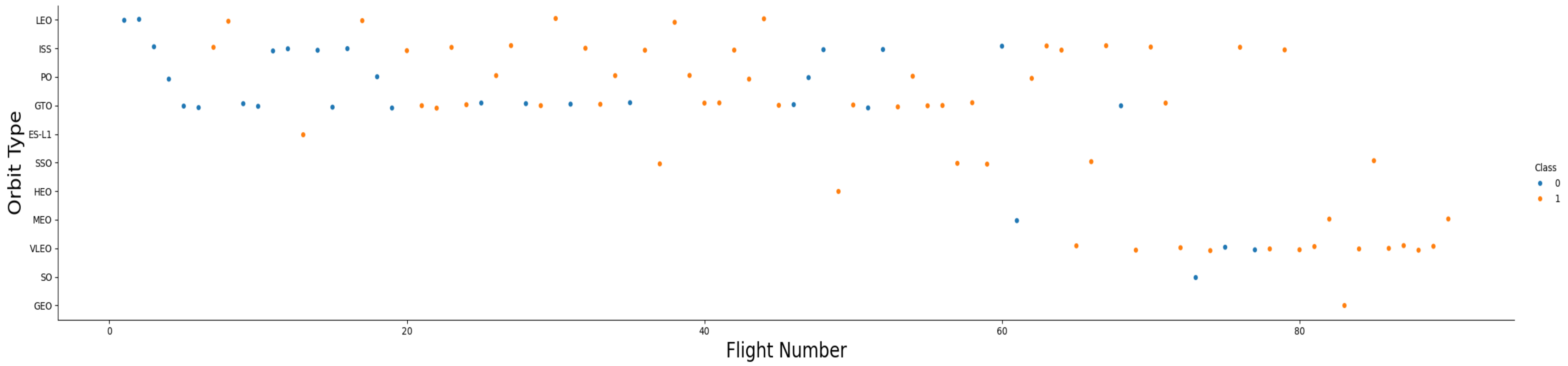
Success Rate vs. Orbit Type

- From a bar chart displaying success rate with each orbit type, it's clear that some orbits have more successful outcomes than others



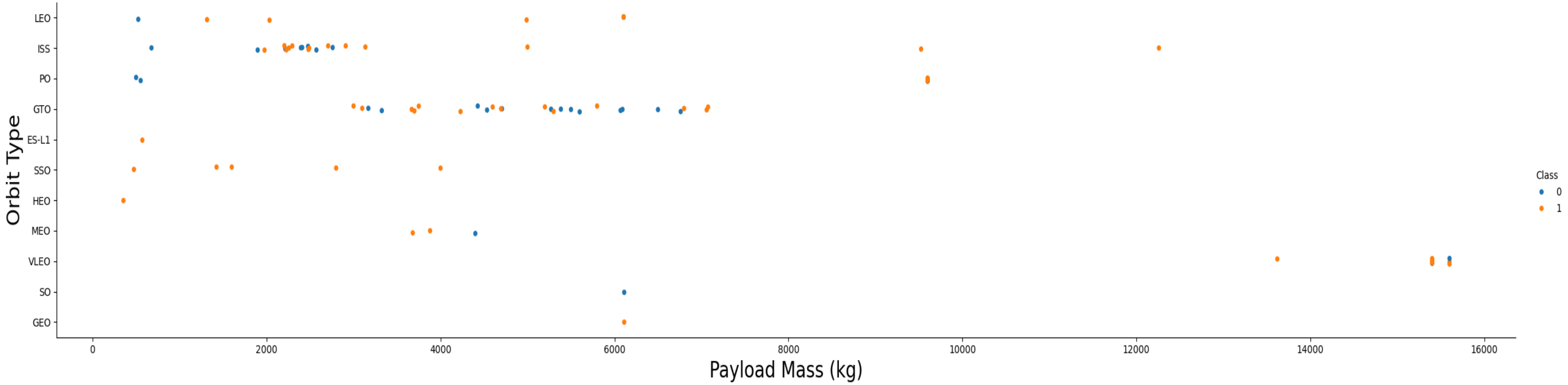
Flight Number vs. Orbit Type

- A scatter point of Flight number vs. Orbit type shows once more that earlier flights have less success in recovering the booster
- It also makes it clear that some “100%” success rates (on the previous plot) are a bit artificial, as some orbits include too few launches for this to be predictive



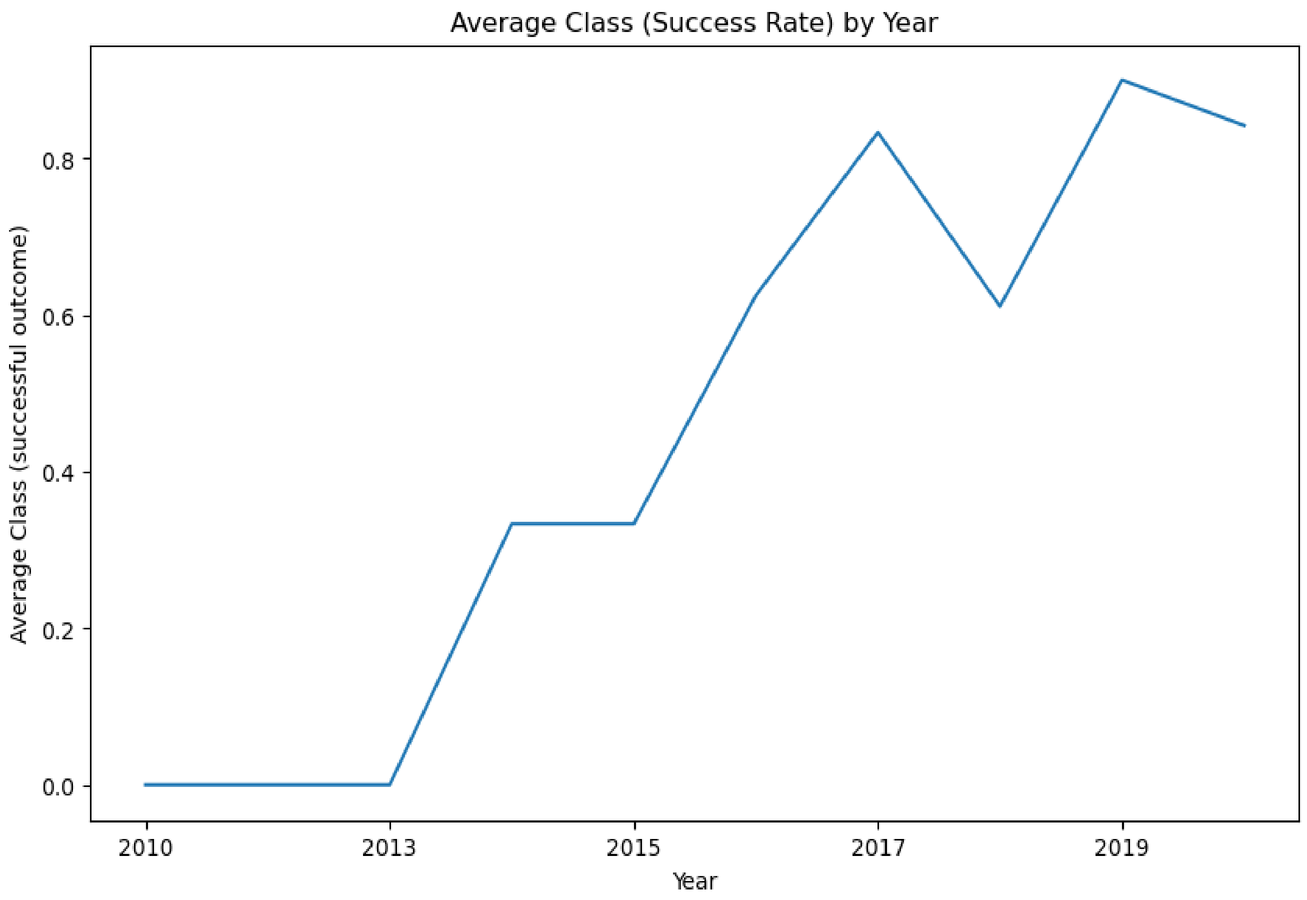
Payload vs. Orbit Type

- A scatter point of Payload vs. Orbit type shows once more that heavier flights are more likely to have successful booster recovery
- It also makes it clear that some "100%" success rates (on earlier plot) are a bit artificial, as some orbits include too few launches for this to be predictive



Launch Success Yearly Trend

- Average Annual Success rate shows a steady trend upward, although with a small dip during 2018
- We might investigate this further, as to what factors might have caused more failures in this period
    - 2018 saw the transition from “Full Thrust” to “Block 5” booster, and some rework was needed before these were able to land successfully
    - Although missions were generally successful, some did not attempt to land the new booster



Launch Site Names

- The launch-site names (unique) found in the data were:

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

Launch Site Names Beginning with 'CCA'

- The first 5 records returned, for launch-site names beginning with ‘CCA’
- It’s apparent that early flights used less powerful (v1) boosters, to Low Earth Orbit, and were not yet able to land successfully, maybe suggesting a degree of “trial running”

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 - for NASA

- The total payload launched where NASA was the customer

total_payload_kg
45596

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 is around 2.5 tonnes

Avg_payload_kg
2534.67

First Successful Ground Landing Date

- The date of the first successful landing outcome on ground pad

Earliest_Ground_Pad
2015-12-22

Drone Ship Landings for Payloads 4000 - 6000

- A list of the boosters used for successful drone ship landings, where payload mass was between 4000 and 6000 kg

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

Mission counts by recovery success and failure

- Counts of the successful and failed (overall) mission outcomes

Mission_outcome	nFlights
Failure	1
Success	100

Boosters Which Carried Maximum Payload

- A list of the boosters which have carried the maximum payload mass

Booster_Version
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

2015 Launch Records

- For drone-ship landing failures in 2015, list the month name, landing_outcome, booster version, and launch site name

monthName	Landing_Outcome	Booster_Version	Launch_Site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Rank Landing Outcomes 2010-06-04 to 2017-03-20

- Rank the counts for each type of landing outcome (such as Failure (drone ship) or Success (ground pad)), in descending order
- Examine records only between 2010-06-04 and 2017-03-20

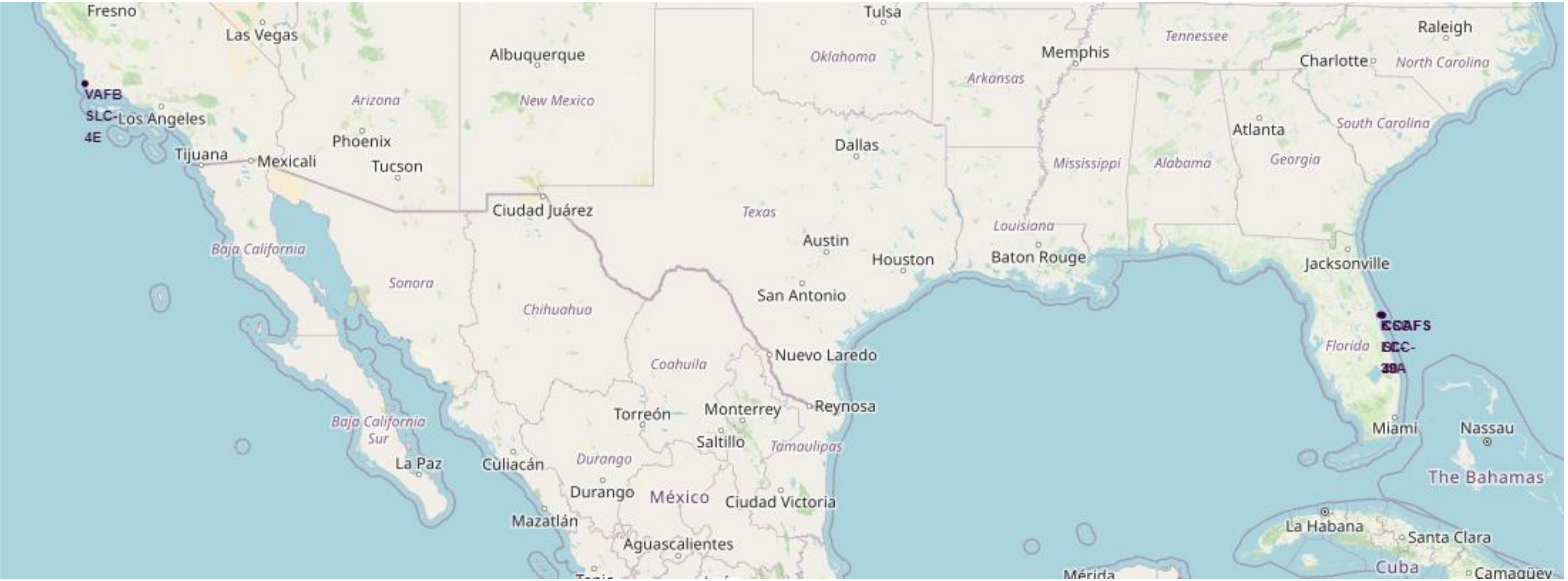
landing_outcome	nOfType	MostFreq
No attempt	10	1
Failure (drone ship)	5	2
Success (drone ship)	5	2
Controlled (ocean)	3	4
Success (ground pad)	3	4
Failure (parachute)	2	6
Uncontrolled (ocean)	2	6
Precluded (drone ship)	1	8

Section 3

Launch Sites Proximities Analysis

Launch Sites Map (zoom level USA) - Folium

- Zoomed out to the lower USA, we can see where the sites are, but with little detail
- DivIcon markers show permanent site labels, but 3 of these overlap in Florida

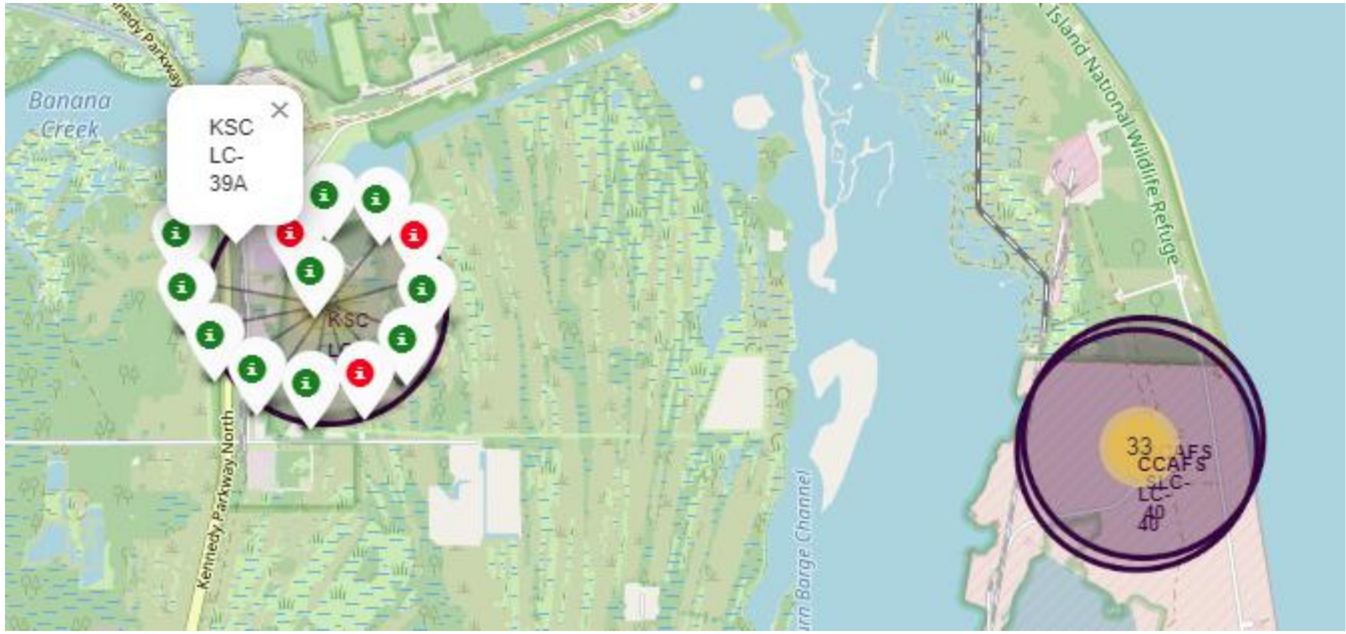


Launch Sites Map (zoom level Florida) - Folium

- Zoomed in to a small area of Florida, we can see three sites (2 are overlapped)
- Circle markers help locate the sites, and one has its popup displayed here
- DivIcon markers provide permanent site label display

Launch Site Success Record - Folium

- A “marker cluster” links groups of icons to a map reference
- Markers with the same coordinates show as a yellow circle giving a count
  - Clicking within the circle zooms in to show individual icons
- Individual (white) icons can be coloured by state – in this case to show recovery



Mapping to Nearby Utilities - Folium

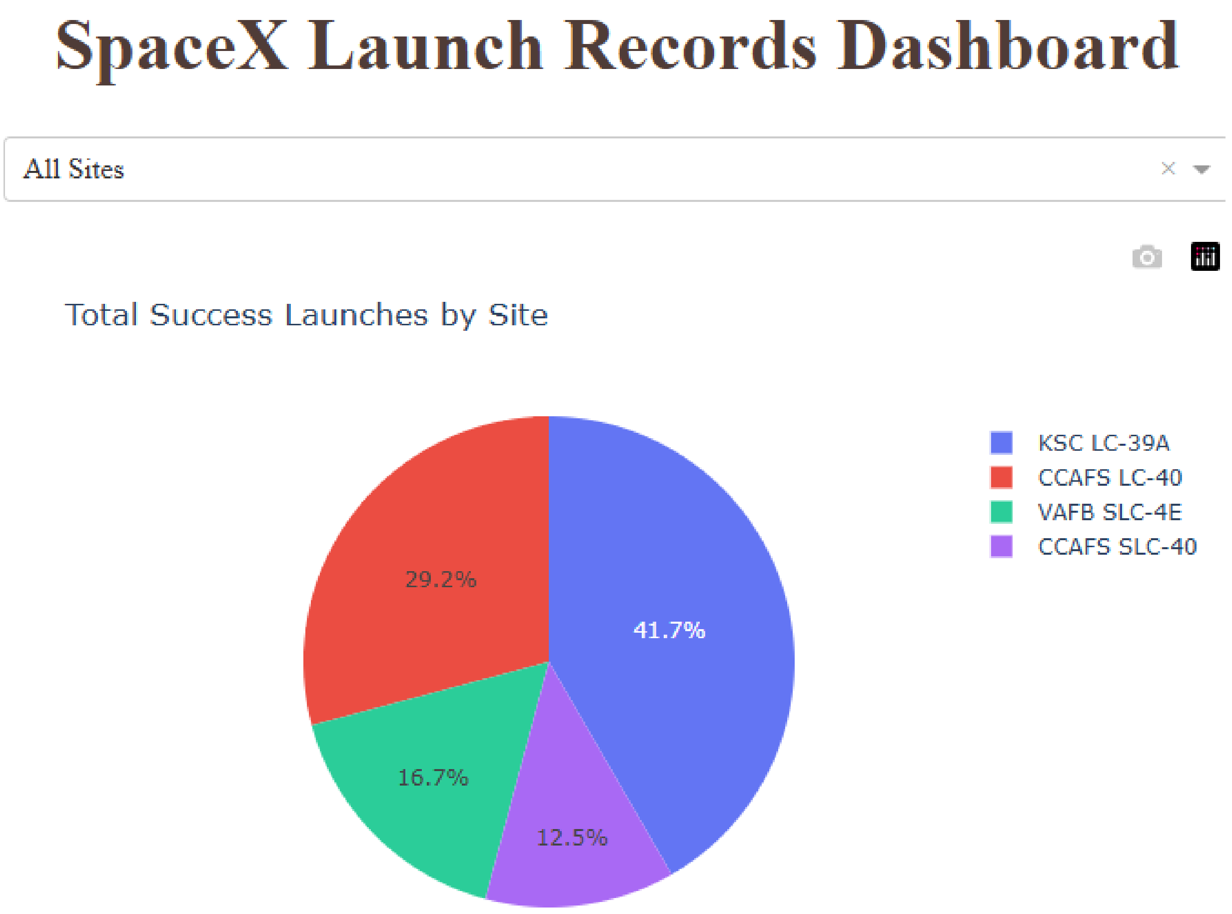
- Maps can be further annotated with
  - Points of interest (such as nearest sea access)
    - Markers can be annotated with distances and not just names
  - Lines (PolyLine) to demonstrate the distances indicated on the marker
- This site, within Vandenberg Air Force Base (VAFB) has few available services, with road and sea access both similar distances away (and NO close rail access)

Section 4

Build a Dashboard with Plotly Dash

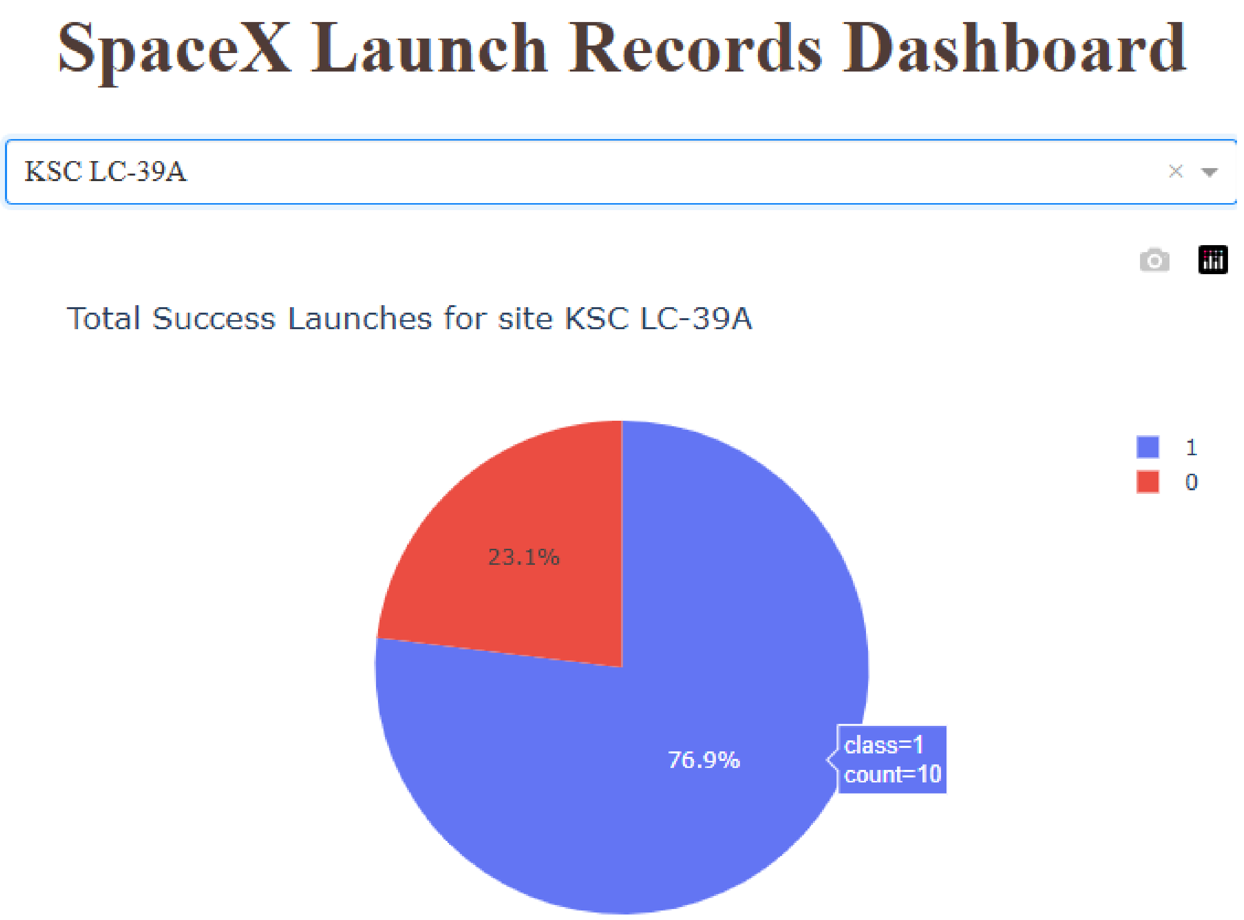
Total Success Launches by Site - Pie

- With “All Sites” selected, the pie chart shows proportion by Site of successful rocket recoveries
- Site KSC has the most successful recoveries, with 10 (as seen when hovering over the segment)
- The maths for “success” works because the class value of 1 shows up, while the value 0 (failure) has no impact



Success vs Failure for Site “KSC” - Pie

- With a single site selected – here KSC, the most successful - the pie chart shows recovery success vs failure
- Hovering over the segments shows 10 successes and 3 failures
- The maths for comparing between categories needs a dataframe group-by step, producing counts for each category that can be plotted



Success vs Payload, by Booster

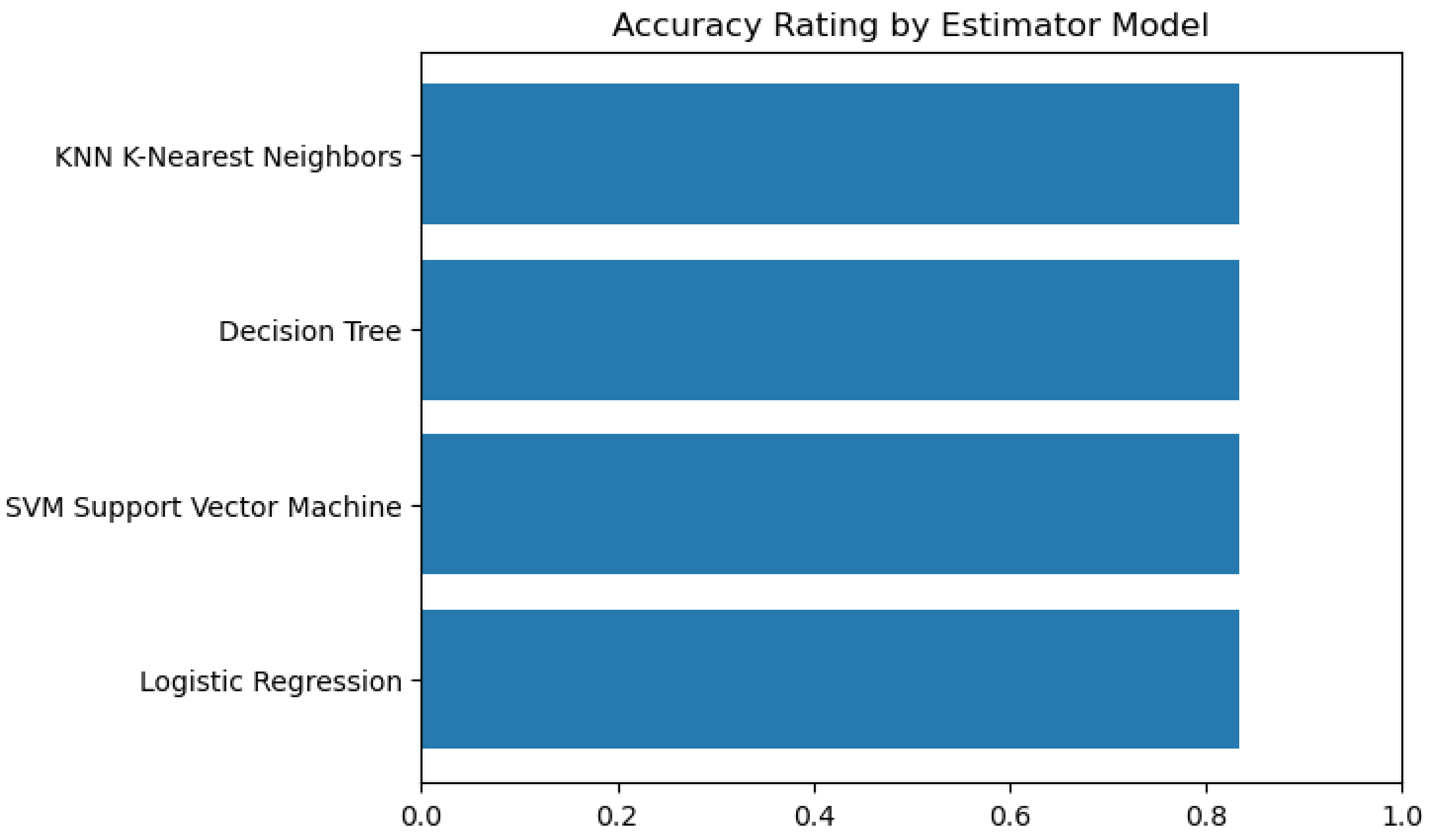
- The chart for Success vs Payload has a key to show which Booster was involved in each launch
  - we can see that FT and B4 are very common in successful launches
- The limits of the Payload axis can be varied with a RangeSlider tool, allowing focus on part of the range
  - I added the range limits to the title as an exercise

Section 5

Predictive Analysis (Classification)

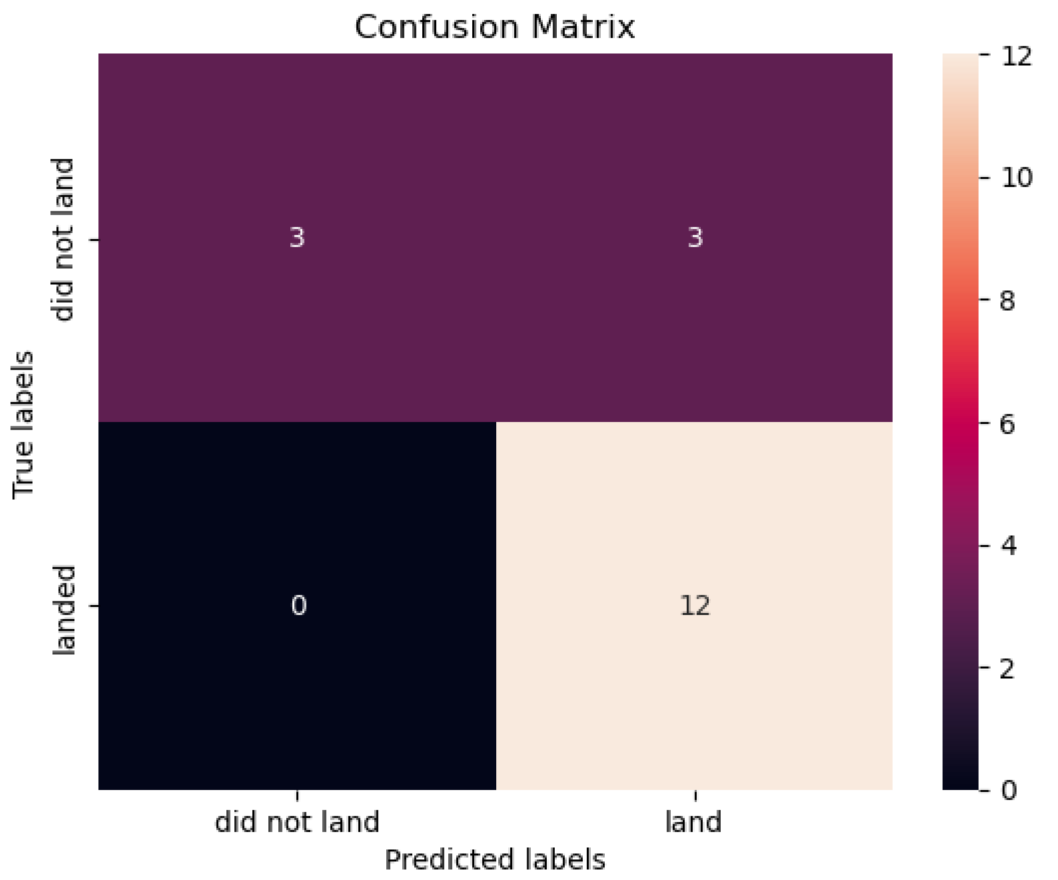
Classification Accuracy

- All estimator models showed the same accuracy. There is nothing to distinguish between them
  - (All = 0.83333)



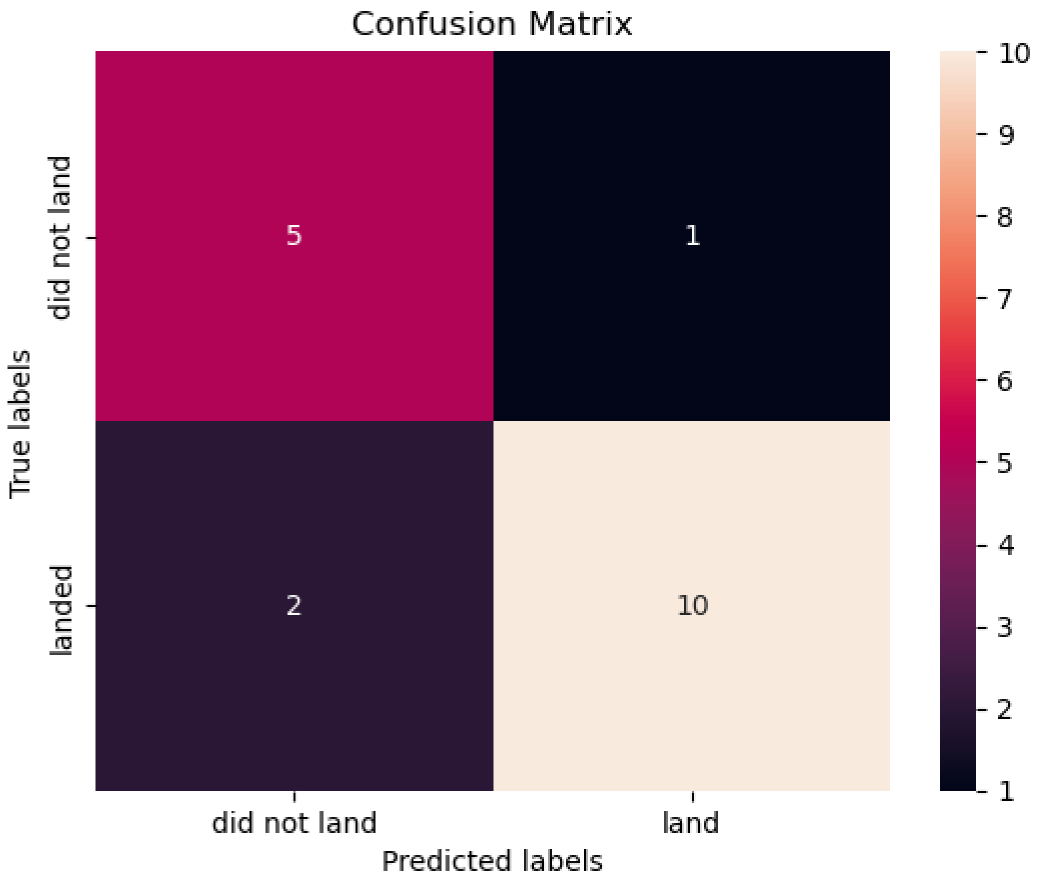
Confusion Matrix – Shared over 3 models

- Three of the 4 models showed this same Confusion Matrix
  - 12 successful landing predictions, and 3 successful predictions of failure-to-land
  - 3 predictions of landing were false, as the true data shows landing failures
    - We are told that SpaceX sometimes “sacrifices” recovery due to other mission parameters
  - The ratio of 15 correct from 18 predictions gives 0.8333 prediction accuracy, under KNN, SVM and Logistic Regression models



Confusion Matrix - Outlier

- One of the 4 models showed this slightly different Confusion Matrix
  - The ratio of 15 correct from 18 predictions is the same, at 0.8333 prediction accuracy, for the Decision Tree model
- While not a requirement, I examined alternate score methods for the model
  - Jaccard Index is 0.625, vs 0.5 for the other models
  - f1 Score is 0.836, vs 0.815 for the other models
- While some scores are a little better for this model, the results are much worse on re-run, so an unstable model here!



Conclusions

- It is clearly very practical to expect to be able to reuse the boosters. Evidence shows that this happens commonly since the relevant technology was introduced
- Even where recovery was not made, there is evidence that this is not always a failure, as given by the outcome None/None which indicates that there was no intention to recover
- The confusion matrices produced in Machine Learning all show a similar level of success in prediction, suggesting there is stability in that result
- The yearly-trend plot shows that successful recovery of boosters is becoming consistently reliable – despite the minor dip in 2018

Appendix

- No other relevant content was created during this project

Thank you!