IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Or Hostezky

December 2024

Outline

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

Executive Summary

Here we will present an attempt to help **SpaceX** minimize the cost of rocket launches by prediction of landing outcomes.

Methodologies:

- Data Collection via API and web scraping
- Data Wrangling
- Exploratory Data Analysis (EDA) via data visualization and SQL
- Interactive Map using Folium
- Dashboard Building using Plotly Dash
- Predictive Analysis (Classification)

Results:

- Exploratory Data Analysis
- Interactive Visual Analytics
- Predictive Analysis

Introduction

Context:

- **SpaceX** brings an innovative ability to reuse the 1st stage of its Falcon 9 rocket, which lowers launch price by ~70% (~$100M per launch)
- Determining 1st-stage landing outcome enables us to determine launch cost
- Our goal is to implement a workflow to predict 1st-stage landing outcome

Key questions:

- Which factors affect 1st-stage landing outcome and in what way?
- What is the rate of successful landings over time?
- Which learning algorithm performs best in this problem?

Section 1

Methodology

Methodology

Executive Summary

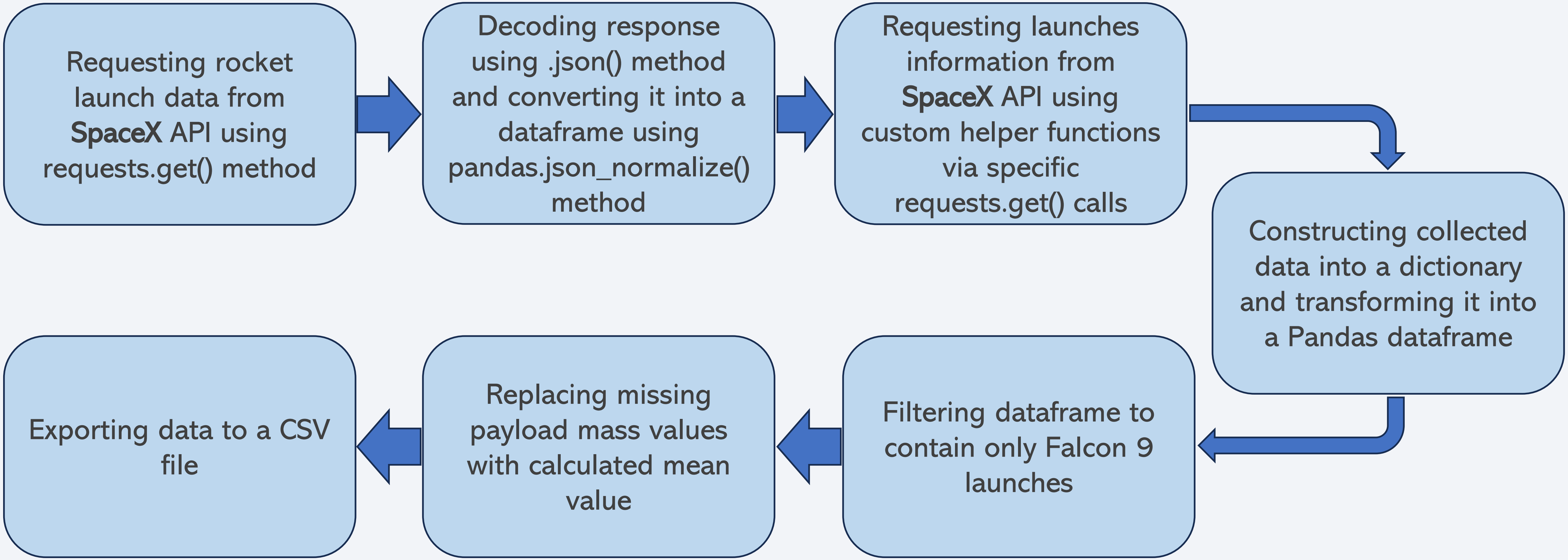
- Data collection
  - Using **SpaceX** REST-API and web scraping from Wikipedia's **SpaceX** entry
- Data wrangling
  - Data filtering, handling missing values, and one-hot encoding of categorical features
- Exploratory data analysis (EDA) using visualization and SQL
- Interactive visual analytics using Folium and Plotly Dash
- Predictive analysis using classification models
  - Fitting different machine learning models (Logistic Regression, SVM, Decision Tree, K Nearest Neighbors), hyperparameter tuning and evaluating each model to find the best performing model

Data Collection

To have a complete set of data about **SpaceX** Falcon 9’s launches for our analysis, we involved two methods of data collection:

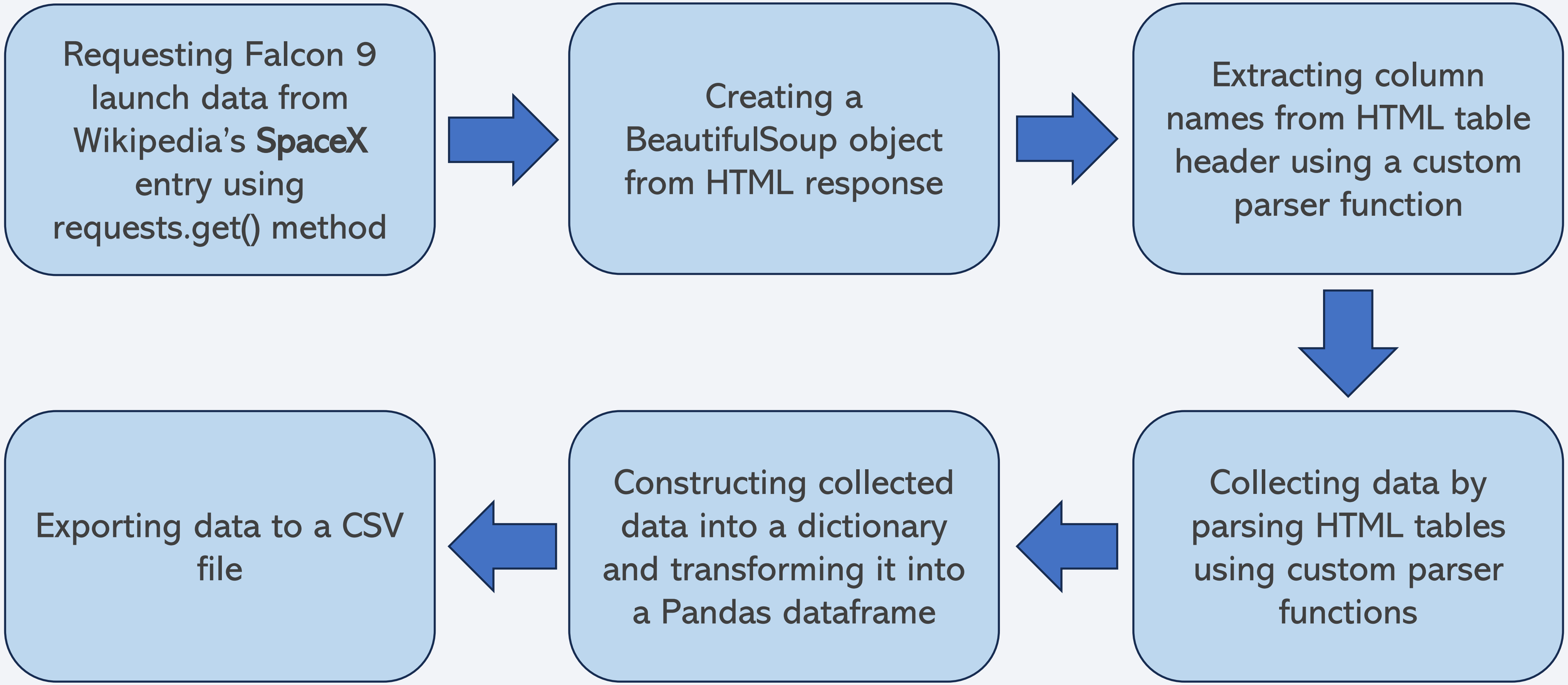
- **API** – We extracted data from **SpaceX** REST API in the form of a JSON using Requests library, and transformed it to a dataframe using Pandas library
- **Web Scraping** – We scraped data from Wikipedia’s **SpaceX** entry using Requests library, and parsed the HTML content using BeatifulSoup library

Data Collection – SpaceX API



[Notebook (GitHub)]

Data Collection - Scraping



[Notebook (GitHub)]

Data Wrangling

We performed some basic EDA to determine and construct training labels:

1. Identifying missing values and column data types
2. Calculating the number of launches on each site using .value_counts() method
3. Calculating the number of occurrences of each orbit type using .value_counts() method
4. Calculating the number and occurrence of mission outcome per orbit type using .value_counts() method
5. Classifying mission landing outcomes into success/fail and constructing a binary landing outcome label
6. Exporting data to a CSV file

[Notebook (GitHub)]

EDA with Data Visualization

To perform EDA, select and engineer (one-hot encode categorical) features, we plotted a few variable relationships using Seaborn library:

- **Categorical scatter plots:** Payload Mass + Launch Site VS Flight Number, Launch Site VS Payload Mass, Orbit Type VS Flight Number + Payload Mass, **all** labeled by Class (=outcome)
  - Categorical scatter plots show relationships between different variables. Such a dependence, if exists, could be used later for machine learning models
- **Bar chart:** Success Rate by Orbit Type
  - Bar charts compare discrete categories of a variable, possibly by groups. They aim to show the relationship between categories and a measured value
- **Line plot:** Success Rate Yearly Trend
  - Line plots show data trends over time (time series)

Notebook (GitHub)

EDA with SQL

To gather more insight about the data, we performed a few SQL queries using SQLite:

- **Displaying** names of unique launch sites in the space mission
- **Displaying** 5 records where launch sites begin with the string 'CCA'
- **Displaying** total payload mass carried by boosters launched by NASA (CRS)
- **Displaying** average payload mass carried by booster version F9 v1.1
- **Listing** date when first successful landing outcome in ground pad was achieved
- **Listing** names of boosters which have success in drone ship and have payload mass between 4000 & 6000
- **Listing** total number of successful and failed mission outcomes
- **Listing** names of booster versions which have carried maximum payload mass
- **Listing** records which will display month names, failure landing outcomes in drone ship, booster versions, and launch sites for months in 2015
- **Ranking** landing outcomes count (Failure (drone ship) / Success (ground pad)) between 2010-06-04 & 2017-03-20, in descending order

Notebook (GitHub)

Build an Interactive Map with Folium

To perform geospatial analysis, we incorporated the following features to a map using Folium library:

- **Circled markers with text labels** (Circle, Marker, and Popup objects) to NASA Johnson Space Center (as example) and to each of the launch sites (demonstrating proximity to coast and equator), using latitude and longitude coordinates
- **Colored markers** (Marker objects) for each launch to show outcomes (success / failure), **clustered by launch sites** (MarkerCluster objects), to identify sites with high success rates
- **Lines** (PolyLine objects) and **distance markers** (Marker objects) between the CCAFS SLC-40 launch site (as example) and its proximities (railway, highway, coastline) and closest city (Titusville, FL), demonstrating location considerations

Notebook (GitHub)

Build a Dashboard with Plotly Dash

We built a dashboard for interactive visual analytics using Plotly Dash, including the following features:

- **Launch-site dropdown list**, enabling the user to either select **(1)** all sites or **(2)** a specific site.
- **Launch-outcome pie chart**, showing:
  - **(1)** Fractions of successful launches by site, out of all successful launches
  - **(2)** Success & failure fractions for the selected site
- **Success Rate VS Payload Mass scatter plot, labeled by Booster Version**, showing the relationship between the two variables. A **Payload Mass slider** is added to enable selection of a wanted payload mass range.

Script (GitHub)

Predictive Analysis (Classification)

We trained and evaluated a few machine learning models on our data using Scikit-learn library and tried to identify the best performing model for this problem:

Creating a NumPy array of the target label 'Class'

→

Standardizing data using preprocessing. StandardScaler(). fit_transform()

→

Splitting data into training and test sets using model_selection.train _test_split() method

→

Cross validating following models:

1. Logistic Regression
2. Support Vector Machine
3. Decision Tree
4. K Nearest Neighbors

on training set using model_selection. GridSearchCV().fit()

→

Calculating accuracy scores for all models using .score() method

→

Visualizing and analyzing confusion matrices for all models using a custom helper function

→

Determining best performing model using accuracy, F1, and Jaccard scores

[Notebook (GitHub)]

Results

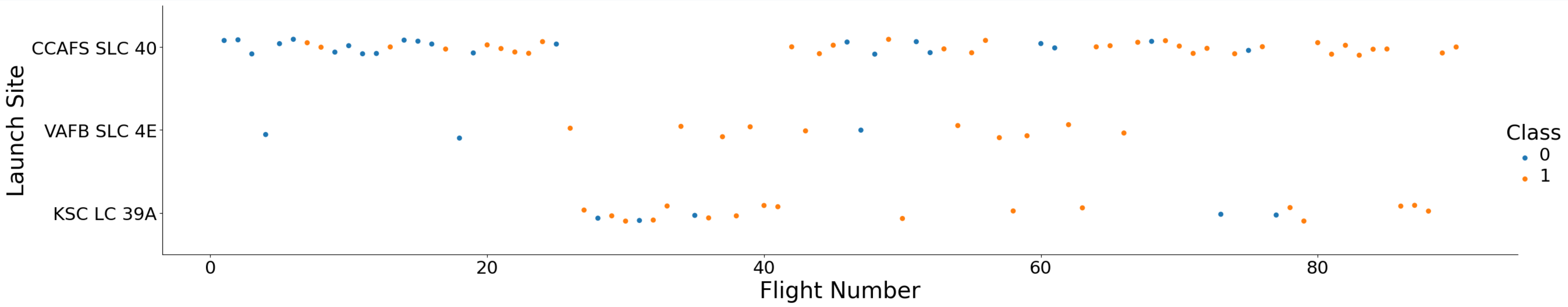
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Launch Site vs. Flight Number

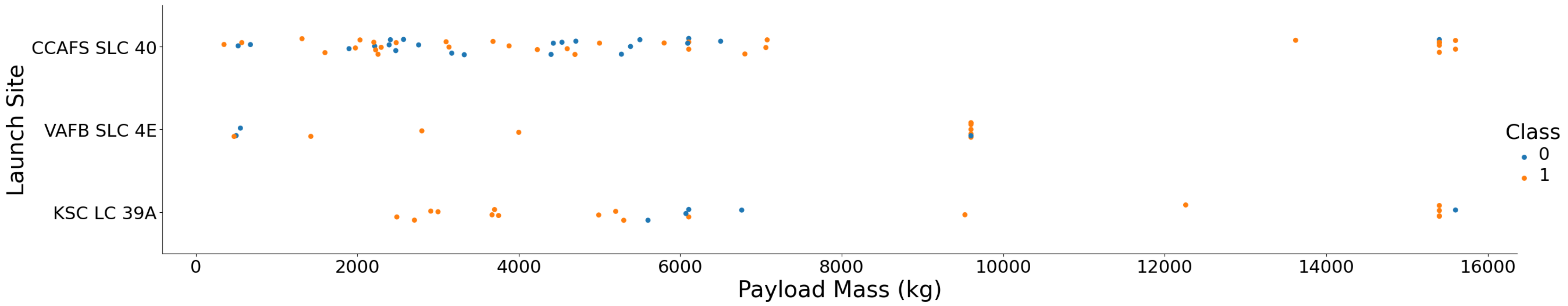
Scatter-point chart, labeled by outcome (Class: 0 – Failure, 1 – Success)



- **Successful launches** become **more common over time**. Therefore, we can assume that a **new launch** will have on average a **higher chance for success** than its formers
- The **CCAFS SLC-40** launch site hosts **significantly more launches** than the other two sites over the time period, except for a time window where it hosts no launches
- From this data it seems that **VAFB SLC-4E** and **KSC LC-39A** have **higher success** rates

Launch Site vs. Payload Mass

Scatter-point chart, labeled by outcome (Class: 0 – Failure, 1 – Success)

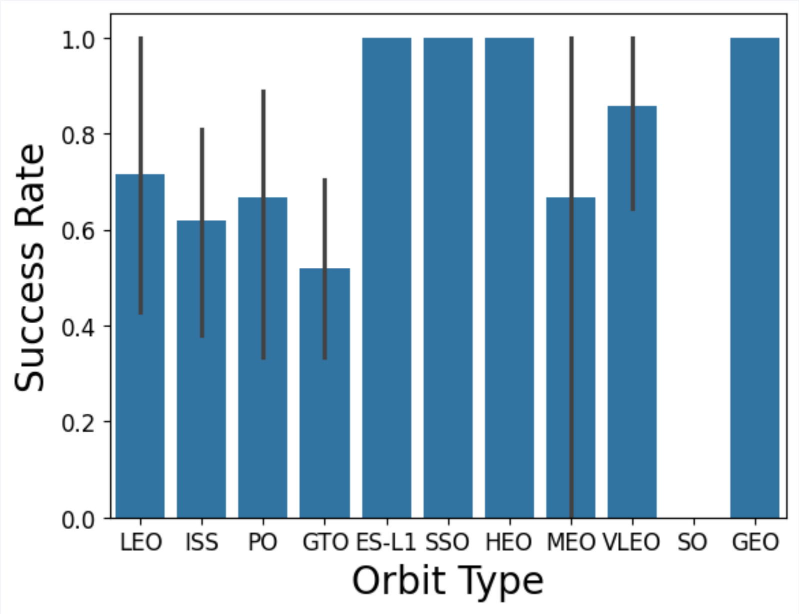


- **Most launches** with a payload **< ~7,000kg**. However, **larger payloads** generally generate **higher success rates**
- **100% success** rate from site **KSC LC-39A** for **small** payload (< ~5,500kg)
- **No flights** launched from site:
  - **CCAFS SLC-40** with **medium-large** payloads (~7,500kg - ~14,000kg)
  - **VAFB SLC-4E** with **large** payloads (> ~10,000kg)
  - **KSC LC-39A** with **small** payloads (< ~2,500kg)

Success Rate vs. Orbit Type

Categorical bar chart

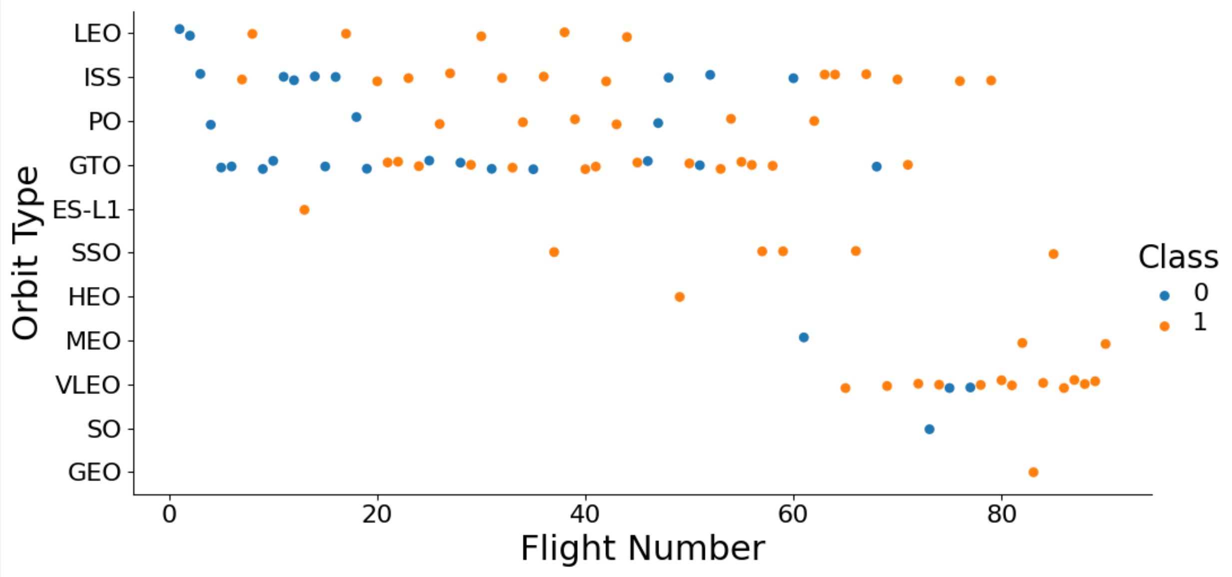
- Whiskers represent 95% confidence intervals



- **ES-L1**, **SSO**, **HEO**, **GEO**, and **VLEO** orbits all have **very high success** rates (all but **VLEO** have 100% success)
- Among **other orbits**, all but **SO** (0% success) have **medium success** rates (~50% - 70%)

Orbit Type vs. Flight Number

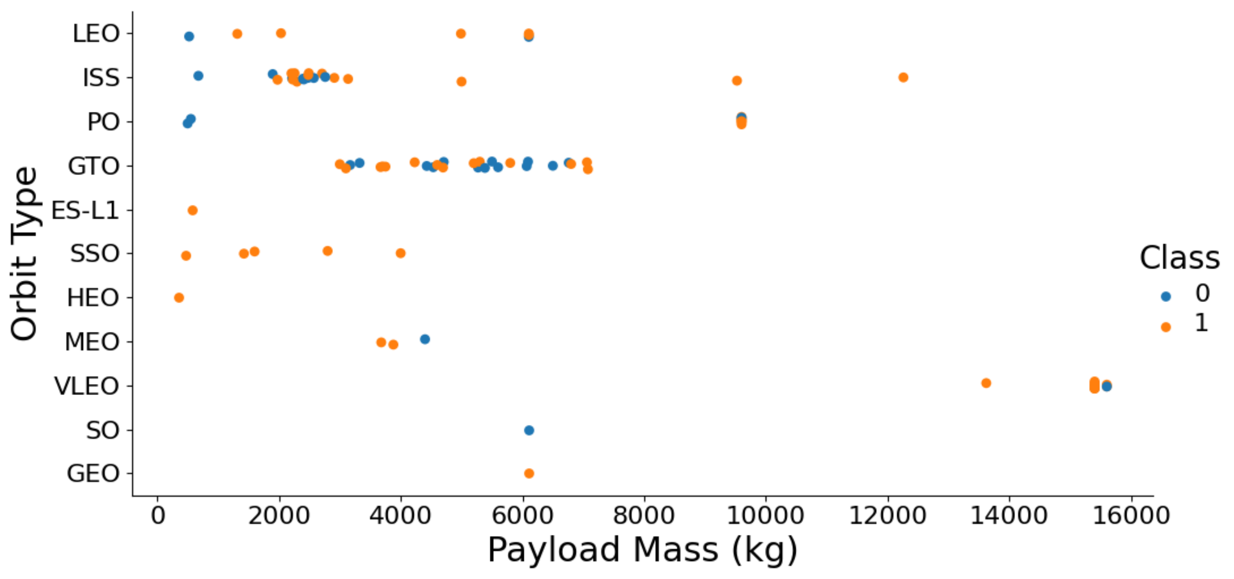
Scatter-point chart, labeled by outcome (Class: 0 – Failure, 1 – Success)



- Here, we can again see **improvement trend** over time **across different orbits,** but **not individually** (**LEO** might be an exception)
- In the **first ~60 flights**, most launches are to **LEO**, **ISS**, **PO**, and **GTO** orbits, whereas **later**, most are to **ISS** (initially) and **VLEO**
- **LEO** and **VLEO** (**ISS** and **GTO**) have distinctively **high (low) success** rates

Orbit Type vs. Payload Mass

Scatter-point chart, labeled by outcome (Class: 0 – Failure, 1 – Success)

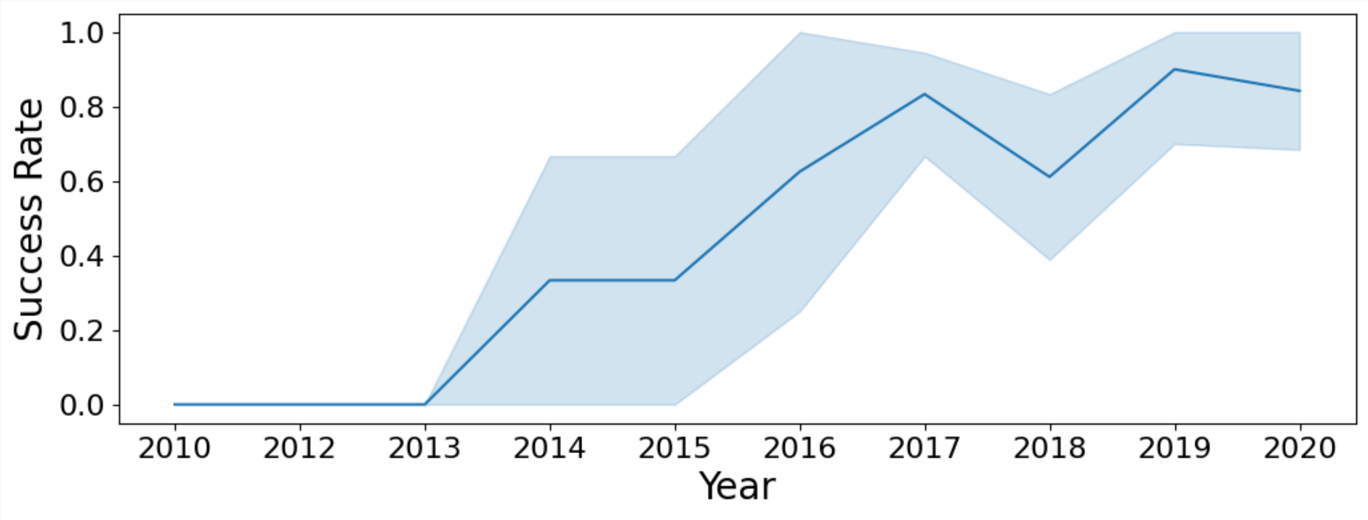


- **Higher success rate** for **large payloads** is evident at **ISS**, **PO** and **VLEO** orbits
- **SSO** has **100% success** rate, albeit a small sample
- **GTO** has a continuous **payload size range at medium payloads** for all its launches

Launch Success Yearly Trend

Line plot

- Colored range represents 95% confidence interval



- Here, the **improvement trend** is most clearly visible, **starting from 2013** after a 3-year plateau
- **Untypical drop** in **2018**, and a **smaller** one in **2020**

All Launch Site Names

Names of the 4 launch sites for Falcon-9

- **DISTINCT** clause on launch-site field

In [10]:
```
%sql SELECT DISTINCT("Launch_Site") FROM SPACEXTABLE
```

```
 * sqlite:///my_data1.db
Done.
```

Out[10]:

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

Launch Site Names Begin with 'CCA'

First 5 records where launch-site name begins with `CCA`

- **WHERE** clause which involves **LIKE** operator to choose relevant sites, and **LIMIT** clause to include only 5 records

In [12]:
```
%sql SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5
```

```
 * sqlite:///my_data1.db
Done.
```

Out[12]:

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

Total payload mass carried by NASA boosters

- **WHERE** clause to choose only NASA boosters, and summing with **SUM** aggregate function

```
In [20]: %%sql
         SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass
         FROM SPACEXTABLE
         WHERE Customer = 'NASA (CRS)'

 * sqlite:///my_data1.db
Done.
```

Out[20]:

Total_Payload_Mass
45596

Average Payload Mass by F9 v1.1

Average payload mass carried by booster version F9 v1.1

- **WHERE** clause to choose version, and averaging with **AVG** aggregate function

In [21]:

```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_Mass
FROM SPACEXTABLE
WHERE Booster_Version = 'F9 v1.1'
```

```
 * sqlite:///my_data1.db
Done.
```

Out[21]:

Average_Payload_Mass
2928.4

First Successful Ground Landing Date

Date of first successful landing outcome on ground pad

- **WHERE** clause to take only relevant launches, and choosing first with **MIN** aggregate function

```
In [22]: %%sql
         SELECT MIN(Date) AS First_Date
         FROM SPACEXTABLE
         WHERE Landing_Outcome = 'Success (ground pad)'

 * sqlite:///my_data1.db
Done.
```

Out[22]:

First_Date
2015-12-22

Successful Drone Ship Landing with Payload between 4000kg and 6000kg

Booster names of successful drone ship landings with payload mass between 4000kg and 6000kg

- **WHERE** clause to take only relevant launches, and **DISTINCT** clause on booster-name field

In [23]:

```
%%sql
SELECT DISTINCT(Booster_Version) AS Booster_Name
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

```
 * sqlite:///my_data1.db
Done.
```

Out[23]:

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

Total Number of Successful and Failure Mission Outcomes

Total number of missions by outcome

- **GROUP BY** clause to group by outcome, and aggregating with **COUNT** function

In [26]:

```
%%sql
SELECT Mission_Outcome, COUNT(*) AS Total_Number
FROM SPACEXTABLE
GROUP BY Mission_Outcome
```

```
 * sqlite:///my_data1.db
Done.
```

Out[26]:

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

Boosters Carried Maximum Payload

Booster names of maximum-payload launches

- **WHERE** clause which involves **sub-query** to find maximum payload mass using **MAX** aggregate function

In [27]:
```
%%sql
SELECT Booster_Version
FROM SPACEXTABLE
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)
```

```
 * sqlite:///my_data1.db
Done.
```

Out[27]:

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 landings in drone ship in 2015, launch months, their booster versions, and launch site names

- **WHERE** clause to choose relevant missions, **SUBSTR** function to extract month/year from date field

In [28]:
```
%%sql
SELECT SUBSTR(Date, 6, 2) AS Month, Landing_Outcome, Booster_Version, Launch_Site
FROM SPACEXTABLE
WHERE SUBSTR(Date, 0, 5) = '2015' AND Landing_Outcome = 'Failure (drone ship)'
```

```
 * sqlite:///my_data1.db
Done.
```

Out[28]:

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

Mission-outcome counts (descending) between 2010-06-04 and 2017-03-20

- **WHERE** clause to take only relevant dates using **BETWEEN-END** operator, **GROUP BY** clause with **COUNT** aggregate function to get frequencies, **ORDER BY** clause with **DESC** command

In [31]:

```
%%sql
SELECT Landing_Outcome, COUNT(1) AS Count
FROM SPACEXTABLE
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY Count DESC
```

```
 * sqlite:///my_data1.db
Done.
```

Out[31]:

Landing_Outcome	Count
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

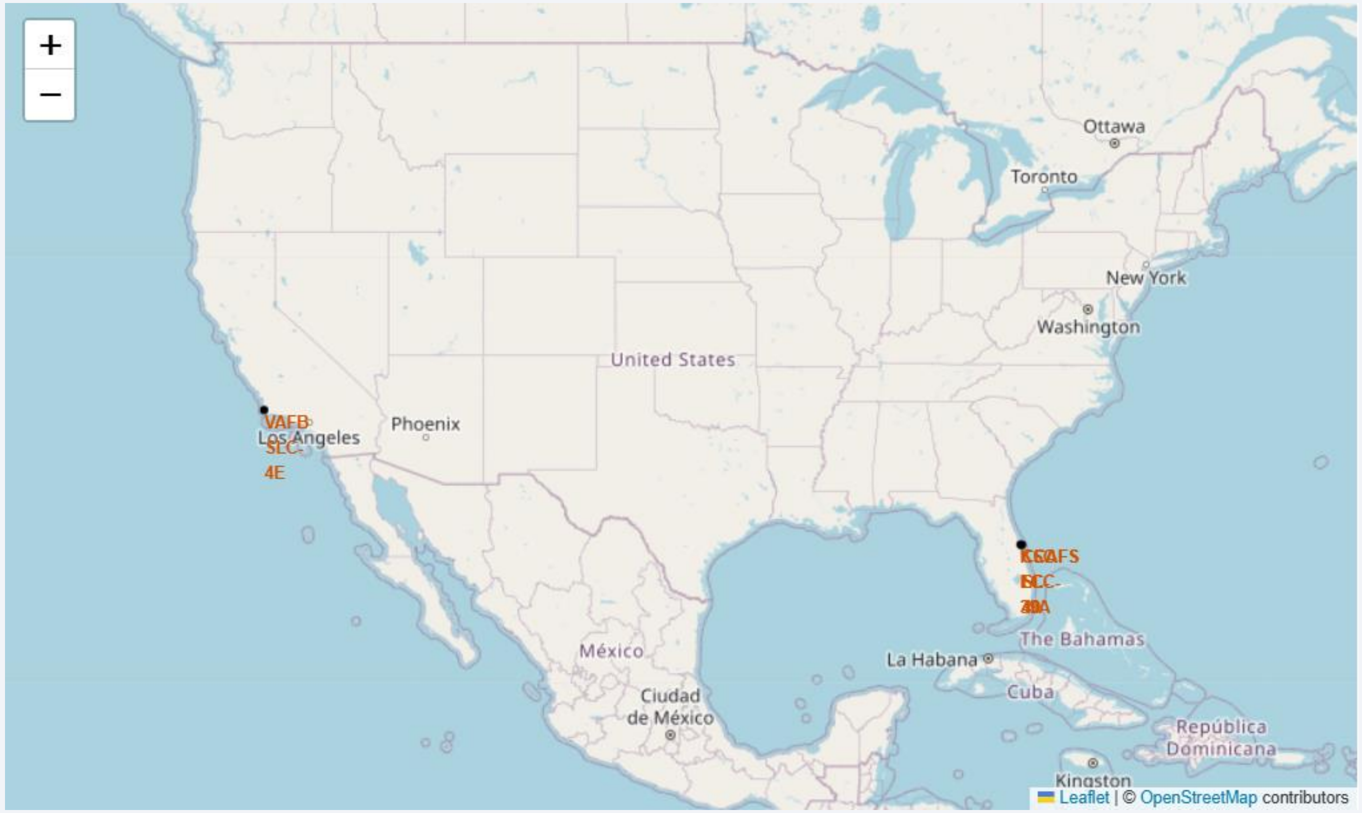
Launch Sites Proximities Analysis

All Launch Sites

- Circled markers with text labels

Findings:

- **Relative proximity to equator** line to minimize fuel consumption and boosters, using Earth's eastward spin to help spaceships get into orbit
- **Close proximity to the coast** for 2 safety reasons:
  - Option to abort and attempt water landing
  - Minimizing risk to people and property from falling debris

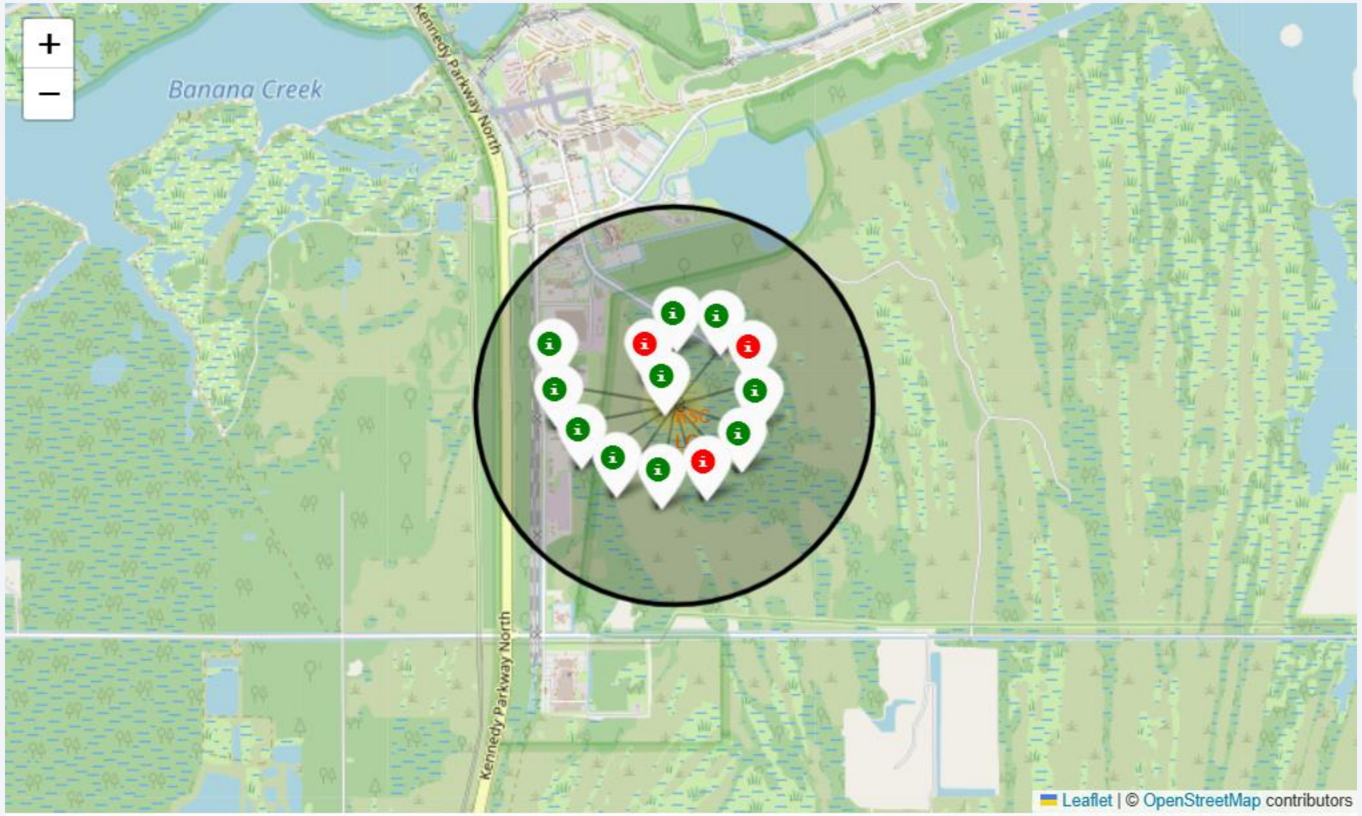


Launch Outcome Labels

- Colored markers (success / failure) in site clusters

Findings:

- **Success rates** for each launch site can be **easily identified**:
  - **KSC LC-39A** has distinctively **high** success
  - **CCAFS SLC-40** and **VAFB SLC-4E** have **medium** success, slightly below half
  - **CCAFS LC-40** has **low** success

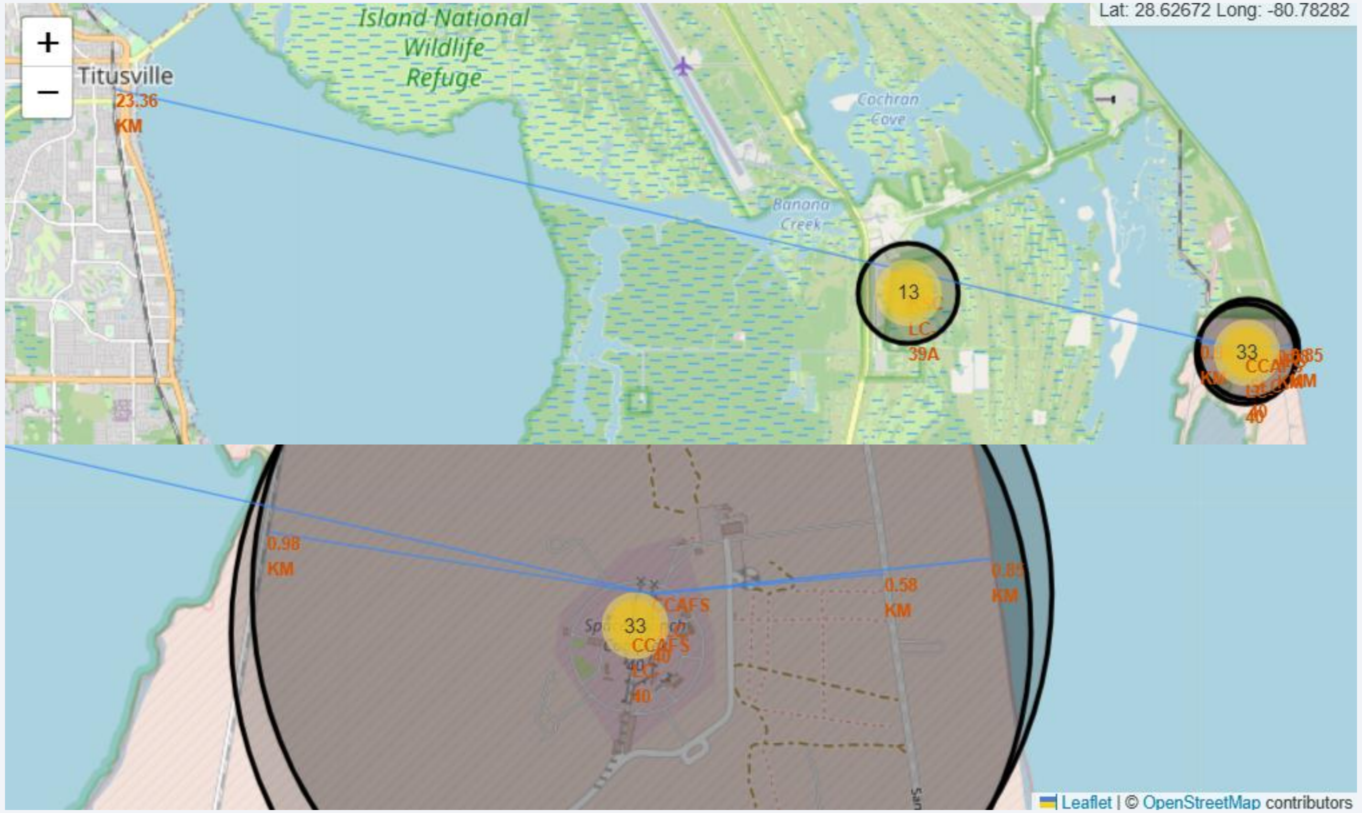


Launch site distances to landmarks

- Distance lines and labels for site **CCAFS SLC-40**

Findings:

- **Close proximity to railways** allows transportation of heavy cargos to site
- **Close proximity to highways** allows easy transportation of people and property to site
- **Close proximity to coast** for previously mentioned reasons
- **Safe distance from cities** to minimize danger to densely-populated areas



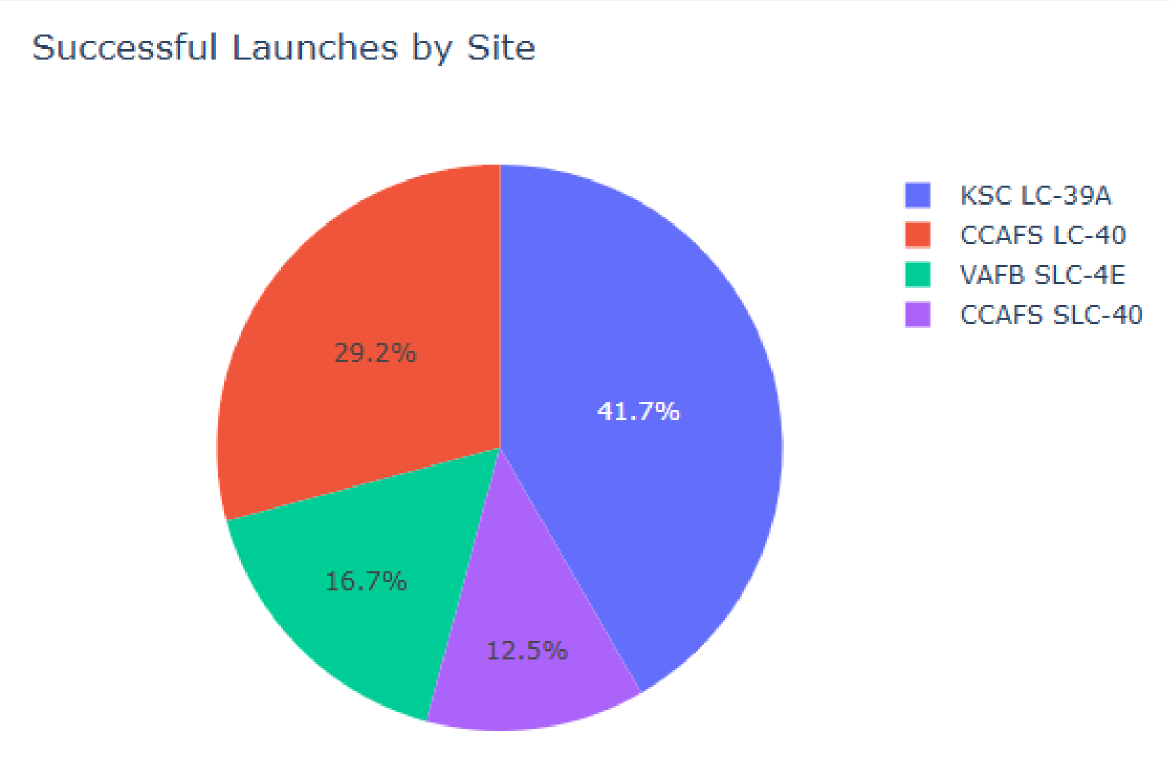
Section 4

Build a Dashboard with Plotly Dash

Success Distribution Between All Sites

Pie chart showing fractions of successful launches for each site out of all successful launches

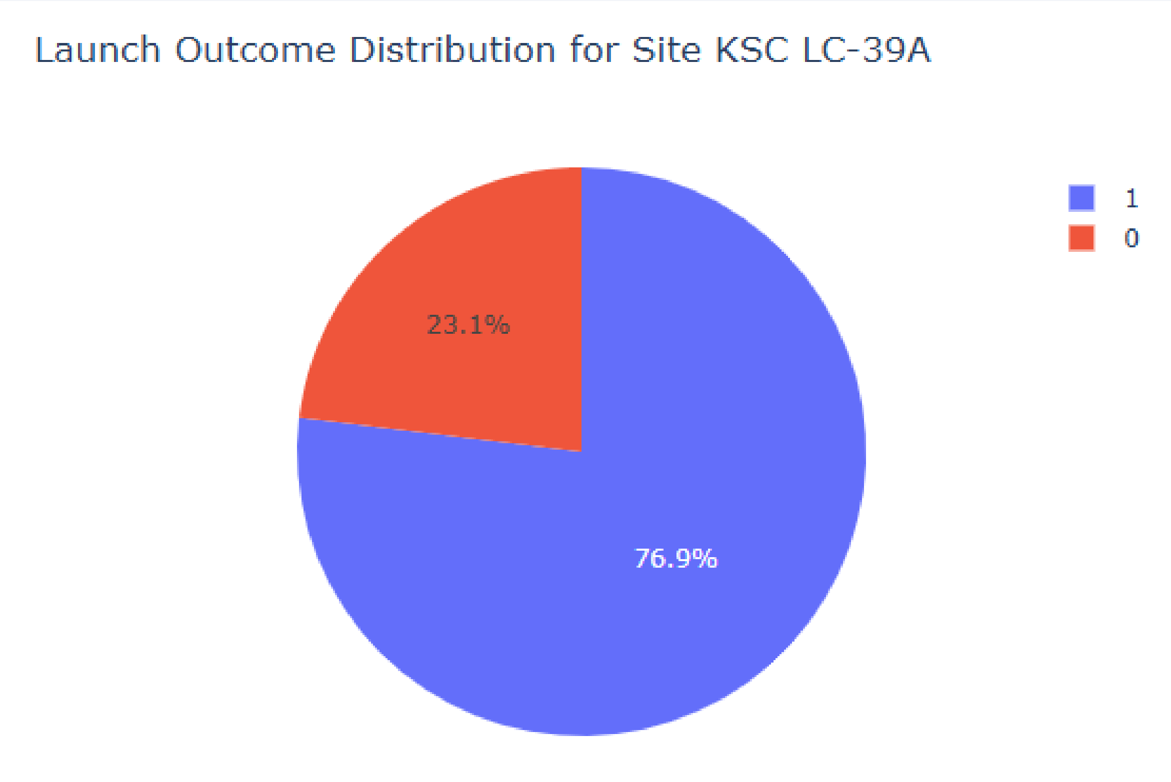
- **KSC LC-39A** product the most successful launches out of all sites (41.7% of all successful launches)



Launch Site Outcome Distribution

Pie chart showing launch-outcome distribution for site with most successful launches (**KSC LC-39A**)
(0 – Failure, 1 – Success)

- **76.9%** of launches from this site are **successful** (10 out of 13)



Launch Outcome vs. Payload Mass

Scatter-point chart showing launch outcomes (0 – **Failure**, 1 – **Success**) as a function of the payload mass for all sites, labeled by booster version, including a slider to choose mass range (<5000kg and >5000kg ranges are shown)

- Almost all **successful launces** are at the **low range**
- **FT boosters** are most successful, whereas **v1.1 boosters** are least

Section 5

Predictive Analysis (Classification)

Model Performance

Model performance was measured using a few evaluation metrics:

	Logistic Regression	Support Vector Machine	Decision Tree	K Nearest Neighbors
Accuracy Score	0.833333	0.833333	0.833333	0.833333
F1 Score	0.888889	0.888889	0.888889	0.888889
Jaccard Score	0.800000	0.800000	0.800000	0.800000
CV Score*	0.846429	0.848214	0.889286	0.848214

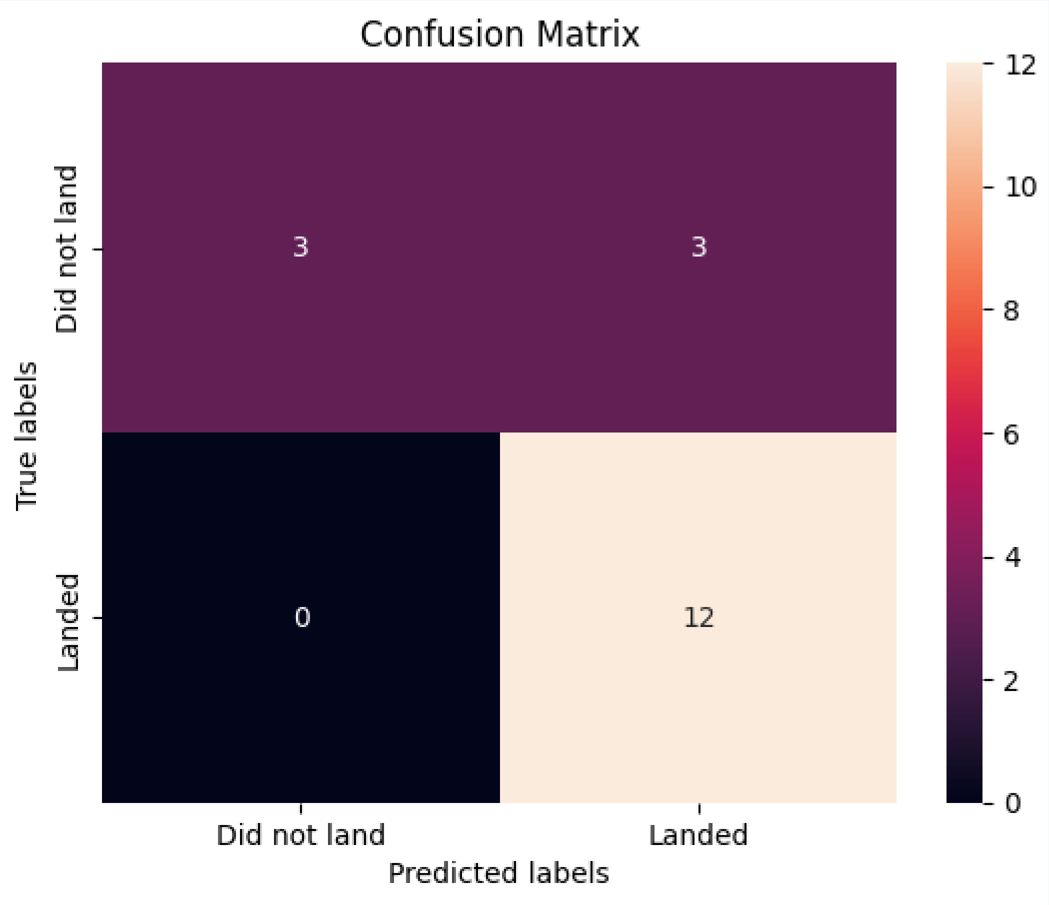
* **CV (cross validation) score** - average model accuracy on all folds using optimal hyper-parameter combination found by grid-search CV

- **Scores based on test set** are all **equal** for all models
- **Cross-validation score** is distinctively higher for the **Decision Tree model**

Confusion Matrix

Confusion matrix is a useful construct to visualize performance of classification models, with a distinction between Type 1 (False Positive) and Type 2 (False Negative) errors

- **All models** produced the **same results**
- Apparent tendency to **Type-1 errors** (and no Type 2)



Model Selection

- From both scores and matrices, it seems **test-set predictions are identical** for all models. Performance can **most likely be resolved if more data is collected** and added to the small (18 cases) test set.
- **Distinctive advantage of the Decision Tree model** in cross-validation scores implies slightly more reliability and generalizability for this problem, yet **additional testing on unseen data is required** to conclude with confidence. However, given no additional information, we should go with this model.
- **Additional factors** should be taken into account when selecting a model, that are **domain and setting** specific. It is also worth considering factors like model **complexity**, **efficiency**, and **interpretability**.

Thank you!

Conclusions

- **Not all data is relevant** for the problem – only some features affect success rate
- Launches with **large payloads** generally have **higher success** rates
- **ES-L1**, **SSO**, **HEO**, **GEO**, and **VLEO** orbits all have **very high success** rates
- General **success** rate shows a clear trend of **increase over time**
- **KSC LC-39A** launch site has the **highest success** rate
- **Launch sites** are located **in proximity** to the **coast** and **equator**
- All models performed equally well, yet the **Decision Tree model** was slightly more **generalizable** for this problem

Conclusions

Limitations and Future Work:

- **Collection of more data** is needed for model-performance evaluation of generalizability on unseen data
- Additional **feature engineering** may improve our model efficiency and performance
- **Ensemble methods** like Random Forest and boosting were not used, yet it is highly likely they can be wielded to improve model performance