Failure to Launch: A Data Science Approach to Predictive Analytics

Brendan Shea, PhD,
brendanpshea@gmail.com

Date: February 2022

Outline

Submitted in partial fulfillment of the requirements for IBM's "Professional Certificate in Data Science.

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

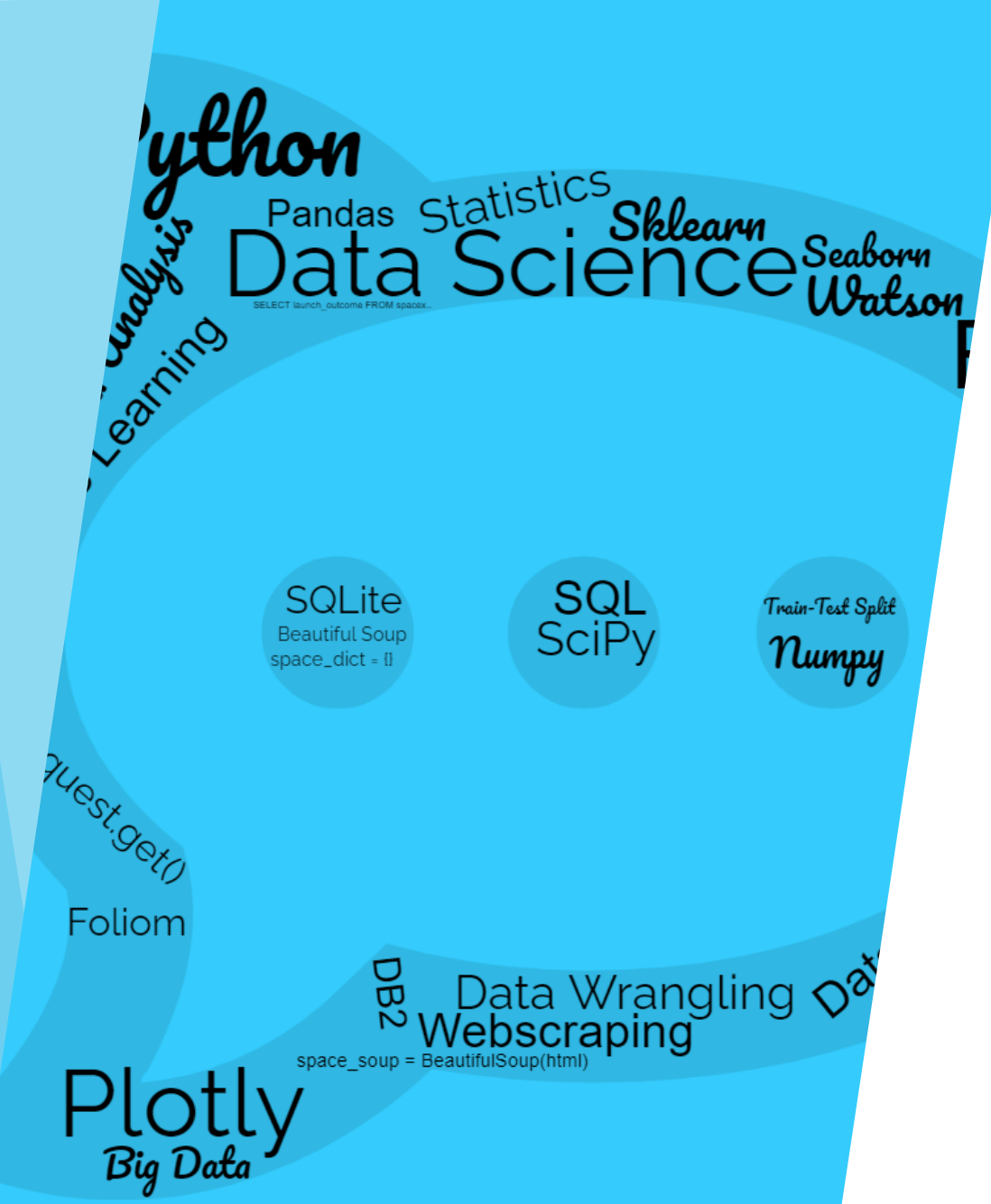
Github: https://github.com/brendanpshea/ibm-data-science

Executive Summary

- In this capstone project for IBM's "Professional Certificate in Data Science", I analyzed publicly available data concerning launches of SpaceX's Falcon9 rocket using Python-based **Jupyter Notebooks.**
- I scraped, "wrangled" and normalized data using a variety of methods.
- I performed exploratory data analysis using **SQL, Pandas,** and **Numpy,** produced static graphics using **Seaborn** and **Matplotlib,** and dynamic maps and dashboards using **Folium** and **Plotly Dash.**
- Finally, I cleaned the data, split it into train/test sets, used a number of machine learning methods to predict mission success/failure.
- The machine learning models I produced have a high level of accuracy (over 80%, though on small data set) in predicting landing outcomes. This level of predictive accuracy would be valuable for many tasks.

Introduction

- This project, which I completed as part of IBM's "Professional Certificate in Data Science" concerns a simplified version of a real-world problem: that of predicting the failure or success of a process (such as the launching of a spaceship, and the attempt to recover part of it for re-use). Data science methods are ideally suited to the answering of these sorts of questions.
- The major problem address was that of predicting mission success/failure for SpaceX launches. Along the way, however, I ended up exploring the relationships between a number of key variables including things such as the location of the launch, the year, the payload, and quantitative and qualitative properties of the data.
- The completed project demonstrates my capacities to (1) gather data, (2) perform exploratory data analysis, (3) construct visualizations and dashboards, and (4) select and implement predictive machine learning methods.



Section 1: Methodology

In this section, I outline the processes of data collection and cleaning, the design of visualizations, and the choice of predictive of machine learning methods.

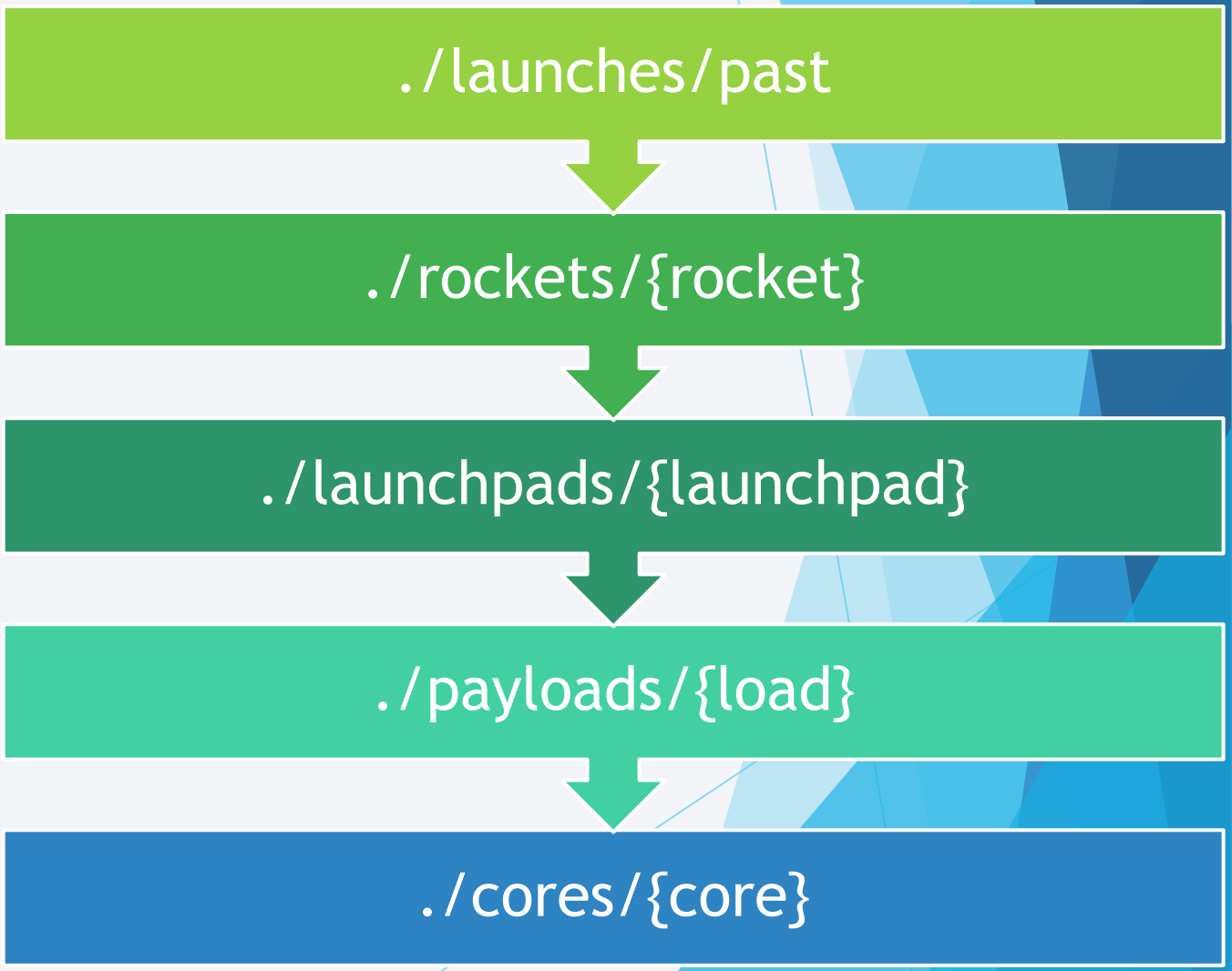
Data Collection

- As with real-world projects, The data for this project were collected using a number of publicly available resources, including SpaceX's RESTful APIs and various Wikipedia pages containing details on the launches.
- Collecting these data required the use of a number of different API calls and web-scraping methods, as the ability to parse what was returned, and store in the appropriate types of data structures that helped capture the *relationships* of these various sources of data.

Data Collection – SpaceX API

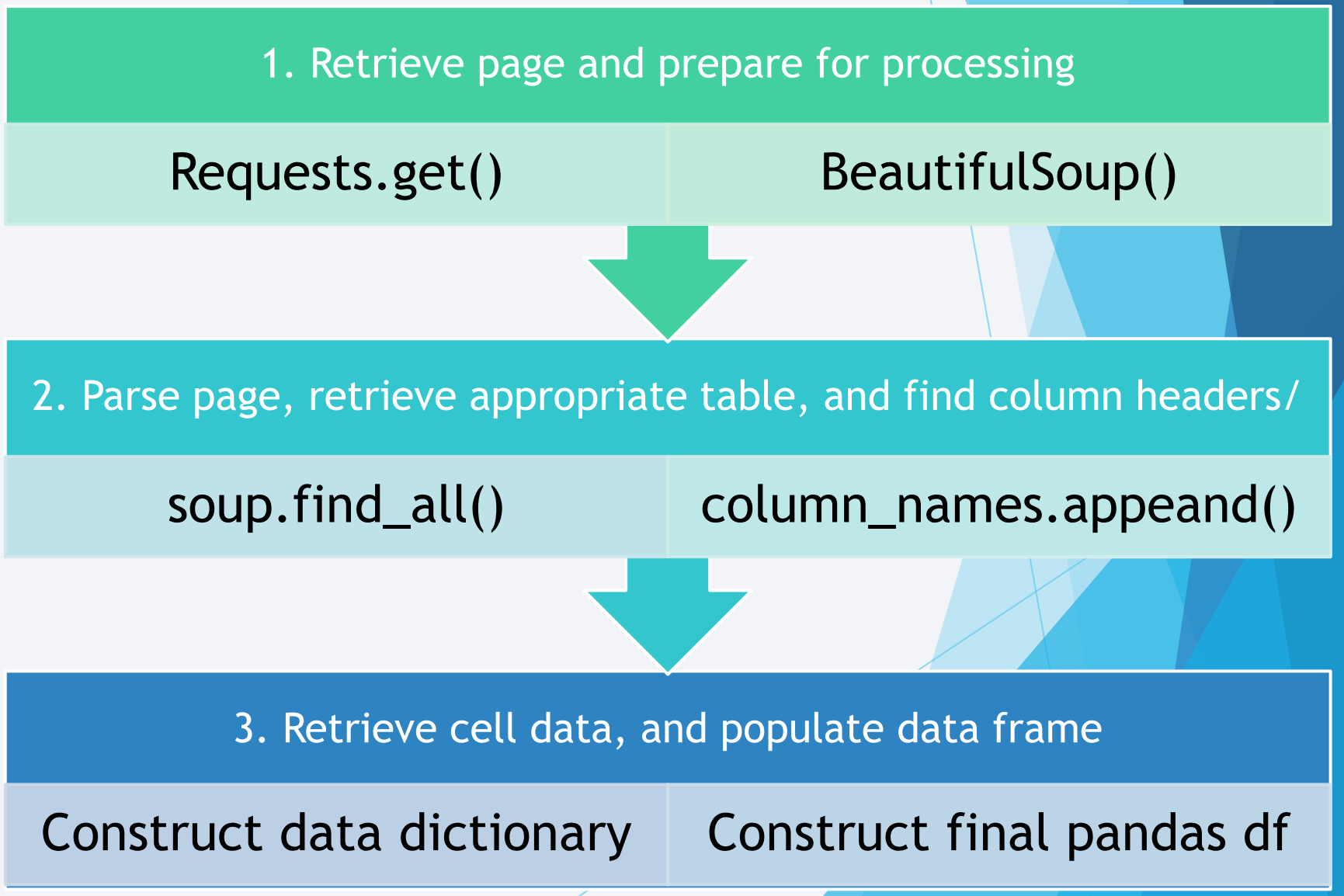
- I started the data collection process with a series of SpaceX API calls at https://api.spacexdata.com/v4/ and loaded the results into a Pandas dataframe.
- The accompanying figure shows the sequence of these calls to various API endpoints.

Github: https://github.com/brendanpshea/ibm-data-science/blob/main/jupyter_labs_spacex_data_collection_api.ipynb



Data Collection - Scraping

- The next phase of data collection involved retrieving and parsing data from Wikipedia pages. Again, data was loaded into a Pandas dataframe.
- Github: https://github.com/brendanpshea/ibm-data-science/blob/main/jupyter_labs_webscraping.ipynb



Data Wrangling

Data Wrangling refers to process of cleaning a preparing data for further analysis. For SpaceX data, I did the following:

1. Identified missing values and determined how to deal with them.
2. Determined the shape of the final dataframe, including number and type of columns, and number of rows.
3. Calculated (a) the number of launches at each sites, (b) the count of each orbit type, and (c) the count of each “mission” outcome by orbit type
4. Created a standardized “mission outcome” column success vs. failure.

Github: https://github.com/brendanpshea/ibm-data-science/blob/main/labs_jupyter_spacex_Data_wrangling.ipynb

EDA with Data Visualization

In my **exploratory data analysis (EDA),** I constructed several static visualizations including:

- Numerous scatterplots exploring the relationship between (a) pairs of quantitative variables and (b) one quantitative and one qualitative variable.
- Line charts showing success rate by year.
- A chart tracking success rate by different types of orbits.
- Visualizations of simple linear regression models exploring the above.

Github: https://github.com/brendanpshea/ibm-data-science/blob/main/jupyter_labs_eda_dataviz.ipynb

EDA with SQL

I also used **Structured Query Language (SQL)** as part of my EDA. Samples queries include:

- Retrieving the names of DISTINCT launch sites, as well as those LIKE certain string patterns.
- Determining total payload mass GROUPED BY client and booster type.
- Determining the date of the first successful ground pad landing.
- Finding the booster names associated launches with a payload mass BETWEEN certain values, as well as those that carried the max payload.
- Ranking outcomes by date.
- Many others

Github: https://github.com/brendanpshea/ibm-data-science/blob/main/jupyter_labs_eda_sql_coursera.ipynb

Interactive Maps with Folium

In addition to static visualizations, I created a series of interactive **Folium maps** that would allow users to explore spatial related to SpaceX launches. These included:

1. Adding markers for each launch site.
2. Coding these launch sites with different colors for success and failures.
3. Calculating the distances between the launch site and notable geographical features, such as coastlines.

Github: https://github.com/brendanpshea/ibm-data-science/blob/main/lab_jupyter_launch_site_location.ipynb

Data Dashboarding with Plotly Dash

Next, I constructed a webapp using Plotly Dash, which provides end-users with an *interactive* visualization. The following functionalities and graphics were included:

1. A dropdown menu allowed users to select which launch site they would like to see data for.
2. Piecharts tracked success rates by launch site.
3. A range selector elements allowed users to further filter data by payload mass (in kg).
4. A scatterplot presented success rates by selected payload range and launch site.

Github: https://github.com/brendanpshea/ibm-data-science/blob/main/Build_a_Dashboard_Application_with_Plotly_Dash.ipynb

Predictive Analysis (Classification)

The final part of the project involved utilizing a number of different machine methods that might be used for predictive analytics.

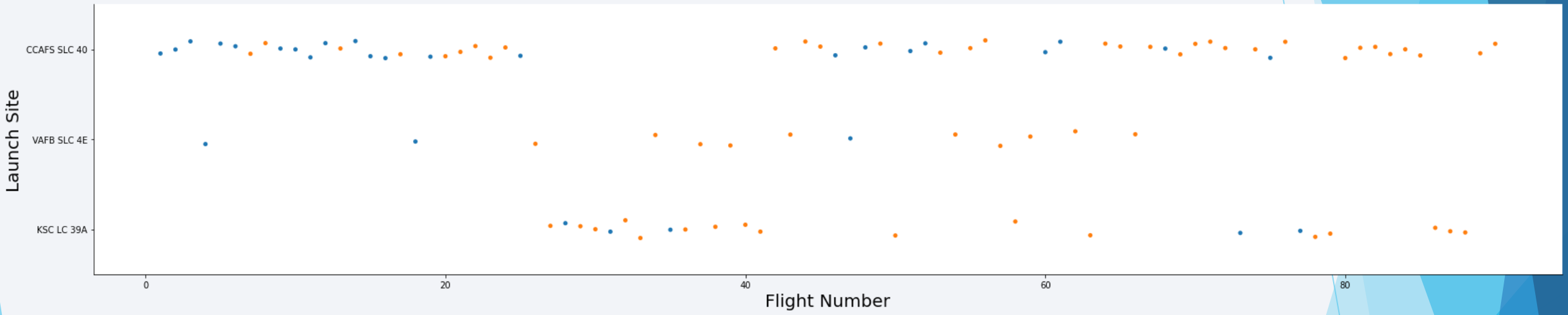
1. Methods included **logistic regression, k-nearest neighbors, support vector machines, decision trees,** and **k-nearest neighbors.**
2. I split the data into “training” and “testing” subsets to improve out-of-sample accuracy.
3. I used **GridSearchCV** to find the optimal parameters for each method.
4. I determined best methods with both **confusion matrices** and quantitative scoring.

Github: https://github.com/brendanpshea/ibm-data-science/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.ipynb

Section 2: Results of Exploratory Data Analysis

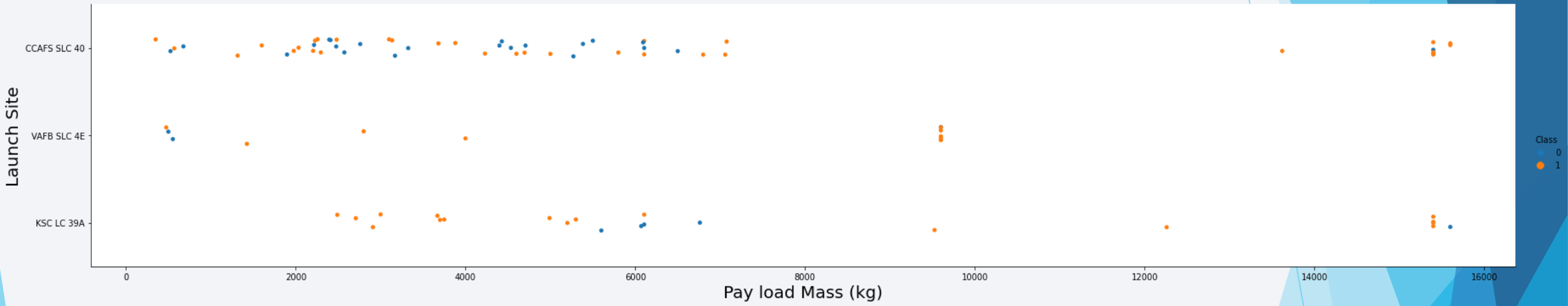
The section presents the results of the initial EDA, including sample visualizations.

Flight Number vs. Launch Site



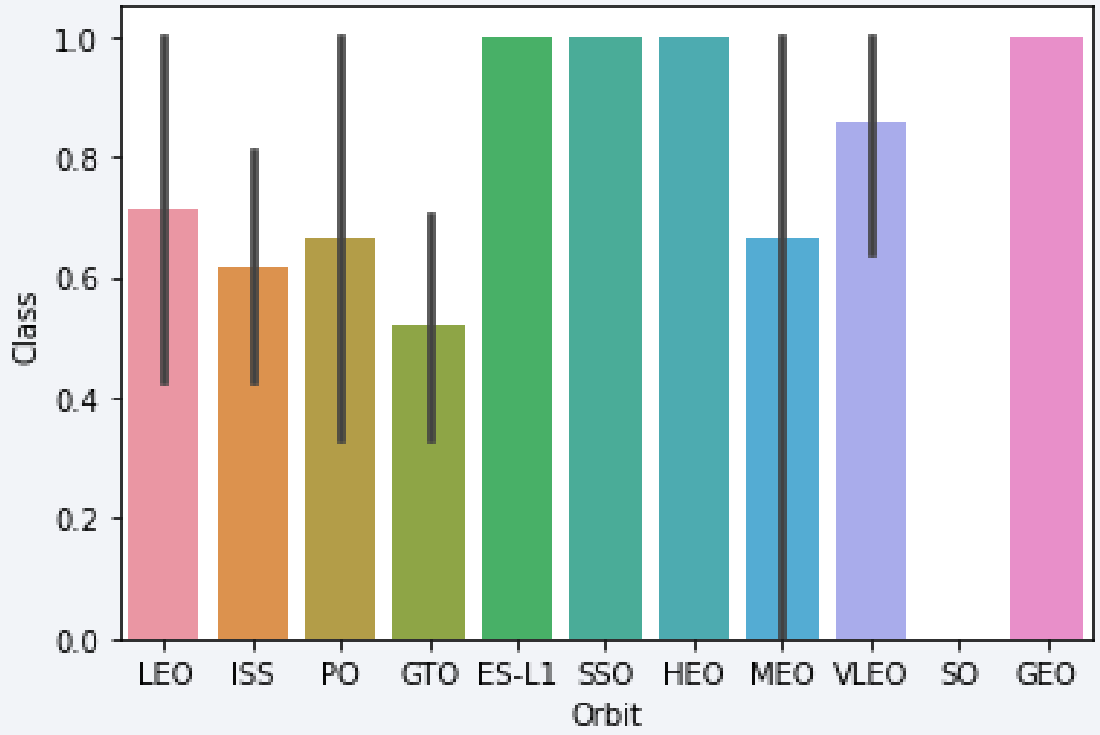
This scatter plot suggests that successful launches have increased with flight number at least 2 of 3 launch sites.

Payload vs. Launch Site



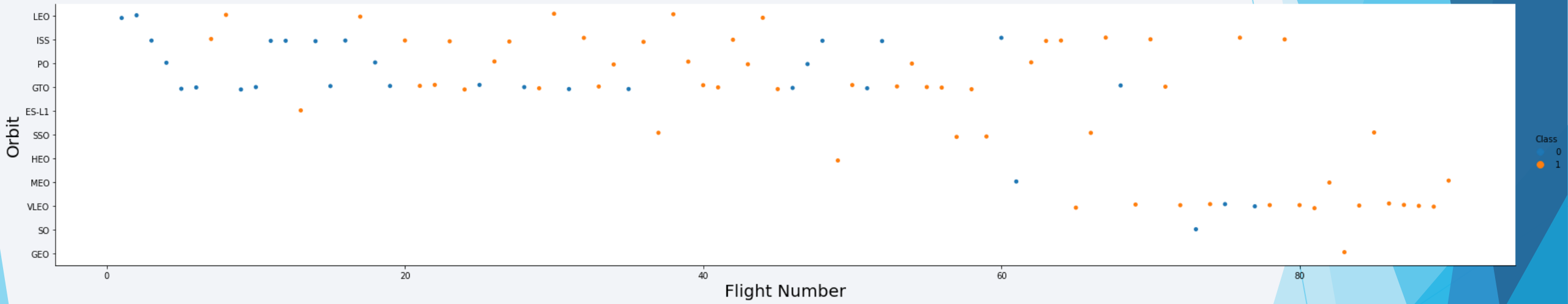
Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavy payload mass(greater than 10000). Success rates do not appear to vary significantly by mass.

Success Rate vs. Orbit Type



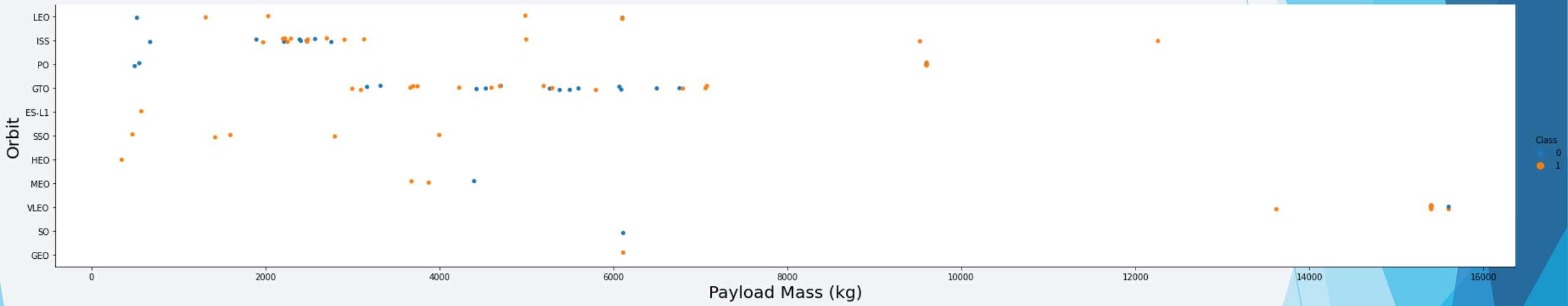
The orbits with the highest success rates appear to be ES-L1, GEO, HEO, and SSO.

Flight Number vs. Orbit Type



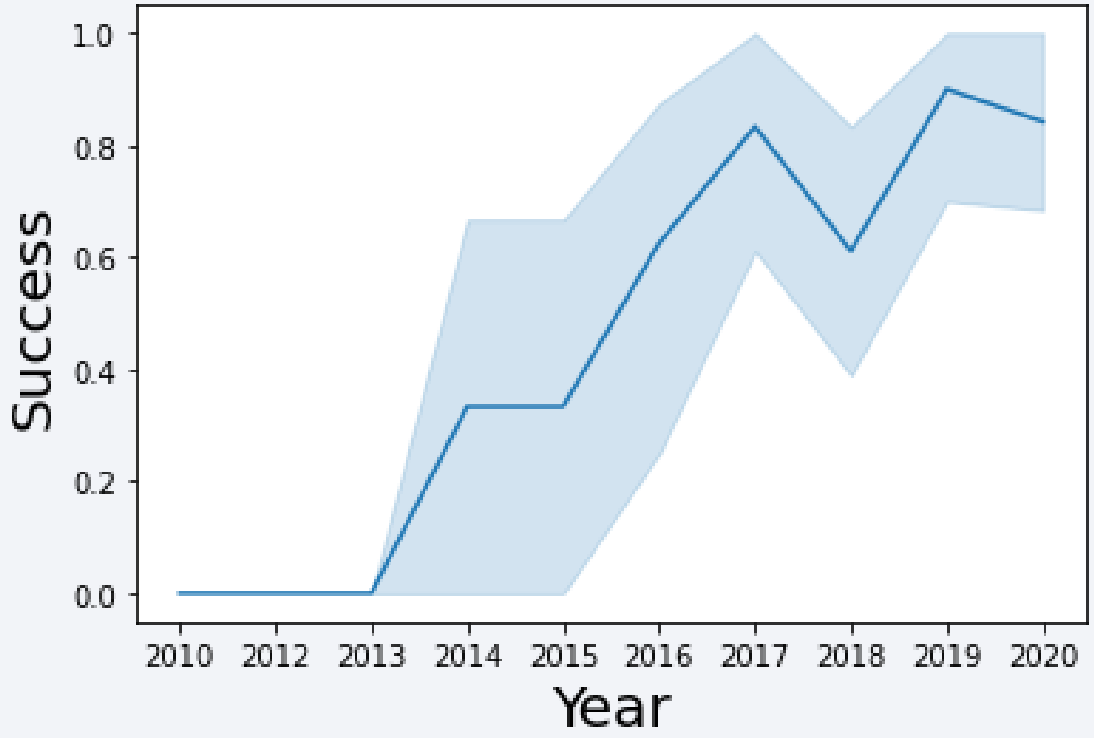
In the LEO orbit, there is a strong correlation between success correlates with the number of flights. In the GTO orbit, there is no such correlation.

Payload vs. Orbit Type



LEO, ISS, and Polar show higher success rates with higher payloads.
Again, for GTO, there does not appear to be an evident correlation.

Launch Success Yearly Trend



Success rates have increased steadily since 2013.

All Launch Site Names

Unique launch sites for 2013-20 were located with a SQL query.

```
%sql select distinct launch_site from spacex;
```

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

Launch Site Names Begin with 'CCA'

SQL was used to select just those launch sites beginning with certain strings, such as ‘CCA’.

```
%sql select * from spacex where launch_site
 like 'CCA%' limit 5;
```

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		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 by Client

I further analyzed such as “Total Payload Mass” by filtering by client (in this case, NASA).

```
%sql select sum(PAYLOAD_MASS__KG_) as "Total Payload Mass" from
 spacex where customer = 'NASA (CRS)'
```

Total Payload Mass
45596

Average Payload Mass by F9 v1.1

I also used SQL to do things such as calculate averages filtered by booster type.

```
%sql select avg(PAYLOAD_MASS__KG_) as "Average Payload Ma
ss" from spacex where Booster_Version = 'F9 v1.1'
```

Average Payload Mass
2928.4

First Successful Ground Landing Date

I also utilized SQL and Pandas in combination to analyze data. Here, I find the data of the first successful ground landing date.

```
my_dates = %sql select date from spacex where landing_outcome =
'Success (ground pad)'

date_df = pd.DataFrame(my_dates)
date_df.columns = ['Dates']
pd.to_datetime(date_df['Dates']).min()
```

```
Timestamp('2015-12-22 00:00:00')
```

Successful Drone Ship Landing with Payload between 4000 and 6000

SQL queries were also used to select data that fell within a specified range, and further filter these by other requirements. Here, I found instances of successful drone ship landings with payloads of between 4,000 and 6,000.

```
%%sql
select booster_version from spacex
  where landing_outcome = 'Success (drone ship)' and
  PAYLOAD_MASS__KG_ between 4000 and 6000
```

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

Total Number of Successful and Failure Mission Outcomes

I used SQL to calculate total success and failure outcomes.

```
select (select count(landing_outcome) from spacex where landing
_outcome like 'Success%') as "Success", (select count(landing_o
utcome) from spacex where landing_outcome like 'Failure%') as "
Failure"
```

Success	Failure
61	10

Boosters Carried Maximum Payload

SQL subqueries were used to find boosters that have carried the maximum payload.

```
%%sql

select booster_version from spacex whe
re PAYLOAD_MASS__KG_ =
  (select max(PAYLOAD_MASS__KG_) from
spacex)
```

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

Here, SQL is used to find FAILED landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015 .

```
select Booster_Version, launch_site from spacex
  where landing_outcome = 'Failure (drone ship)' and
  date like '%2015'
```

Booster_Version	Launch_Site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

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

Finally, I produced an ordered 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 involved SQL, Pandas, and (more advanced parts of_ SQL Alchemy.

```
all_data = conn.execute("select * from spacex")
df = pd.DataFrame(all_data)
conn.execute("drop table if exists q10Table")

query = "select * from spacex"
df2 = pd.read_sql(query, conn)
df2['Date'] = pd.to_datetime(df2['Date'])
q10_list = df2[(df2['Date'] > pd.to_datetime('2010-06-
04')) & (df2['Date'] < pd.to_datetime('2017-03-20'))]
q10_list.to_sql('q10Table', engine)

results = conn.execute("select landing_outcome, count(land
ing_outcome) from q10Table group by landing_outcome order
by count(landing_outcome)")
df = pd.DataFrame(results)
df.columns =['Landing Outcome', "Count"]
df
```

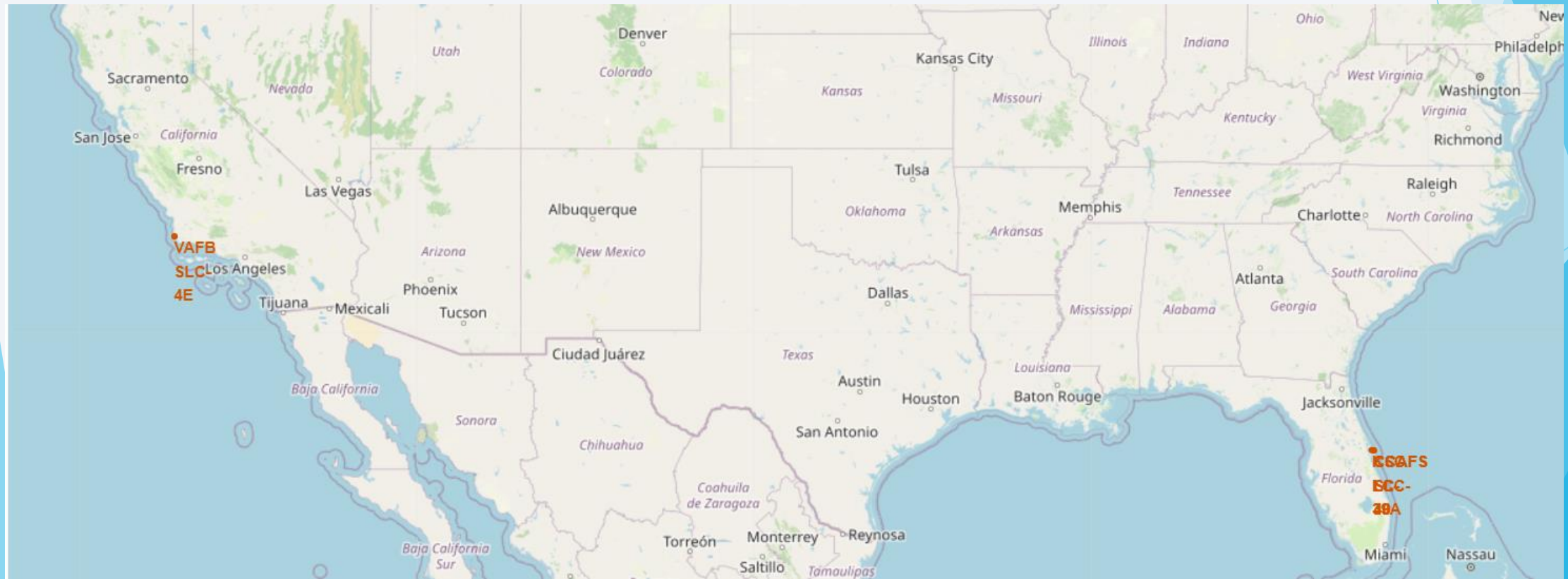
0	Failure (parachute)	1
1	Precluded (drone ship)	1
2	Uncontrolled (ocean)	2
3	Controlled (ocean)	3
4	Failure (drone ship)	5
5	Success (drone ship)	5
6	Success (ground pad)	5
7	No attempt	10

Section 3: Launch Site Proximity Analysis

Here, I use Folium maps to provide insight into the influence of geographical features on the locations of launch sites and launch outcomes.

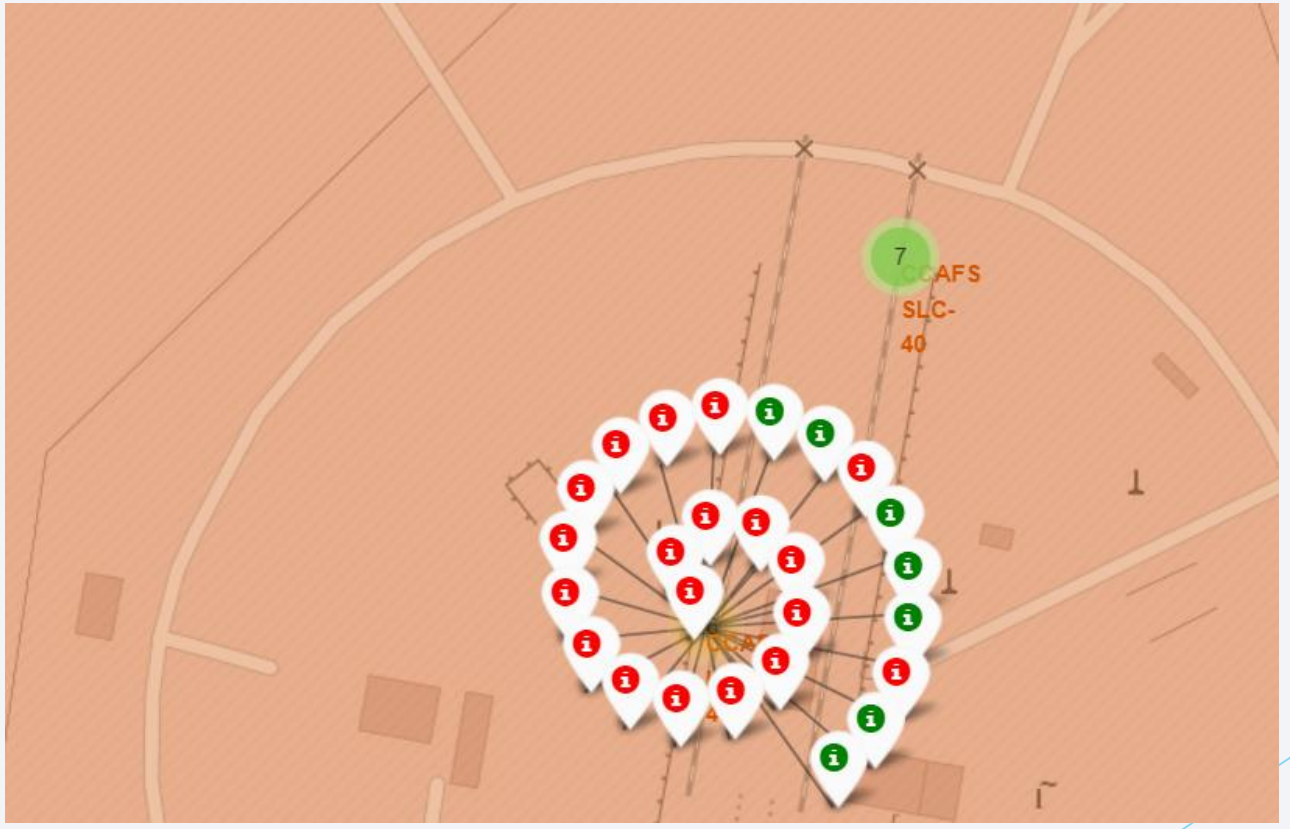
Folium Map With Launch Sites

- The following figure is an interactive Folium map with all SpaceX launch sites indicated by different markers.
- Users can zoom in/out, navigate to different areas of the map, etc. in order to explore different geographical features.



Folium Map With Launch Outcomes

- I also created a Folium map that allowed users to track the geographic location of successful vs. unsuccessful launches.



Folium Map With Distances

Folium maps are also used to track and display distances to key geographical features, such as coastlines (for example, all launch sites are located close to coastlines).

Flask

web development,
one drop at a time

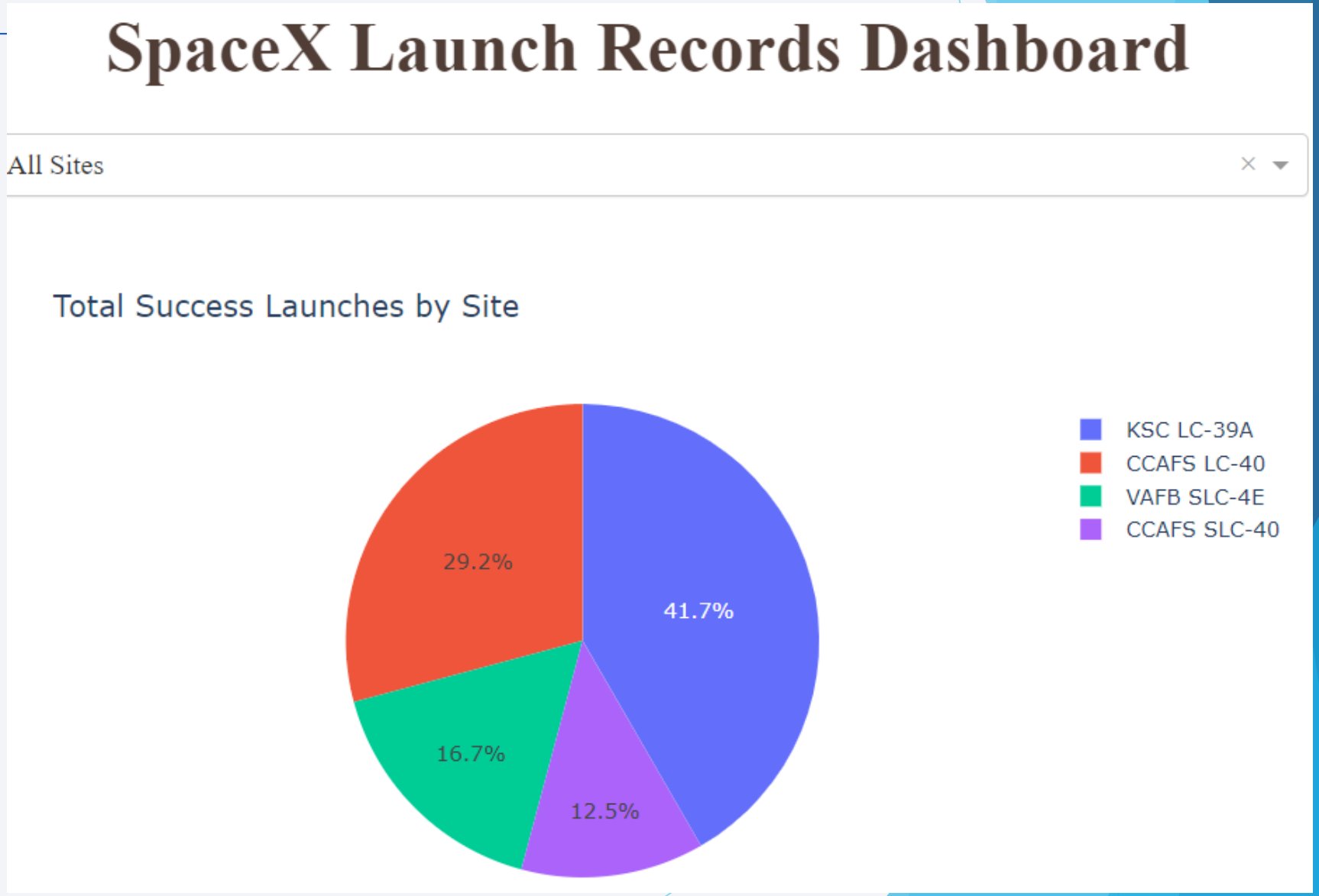
Section 4: Building a Dashboard With Plotly Dash

Here, I create a web application that allows users to interact with data visually.

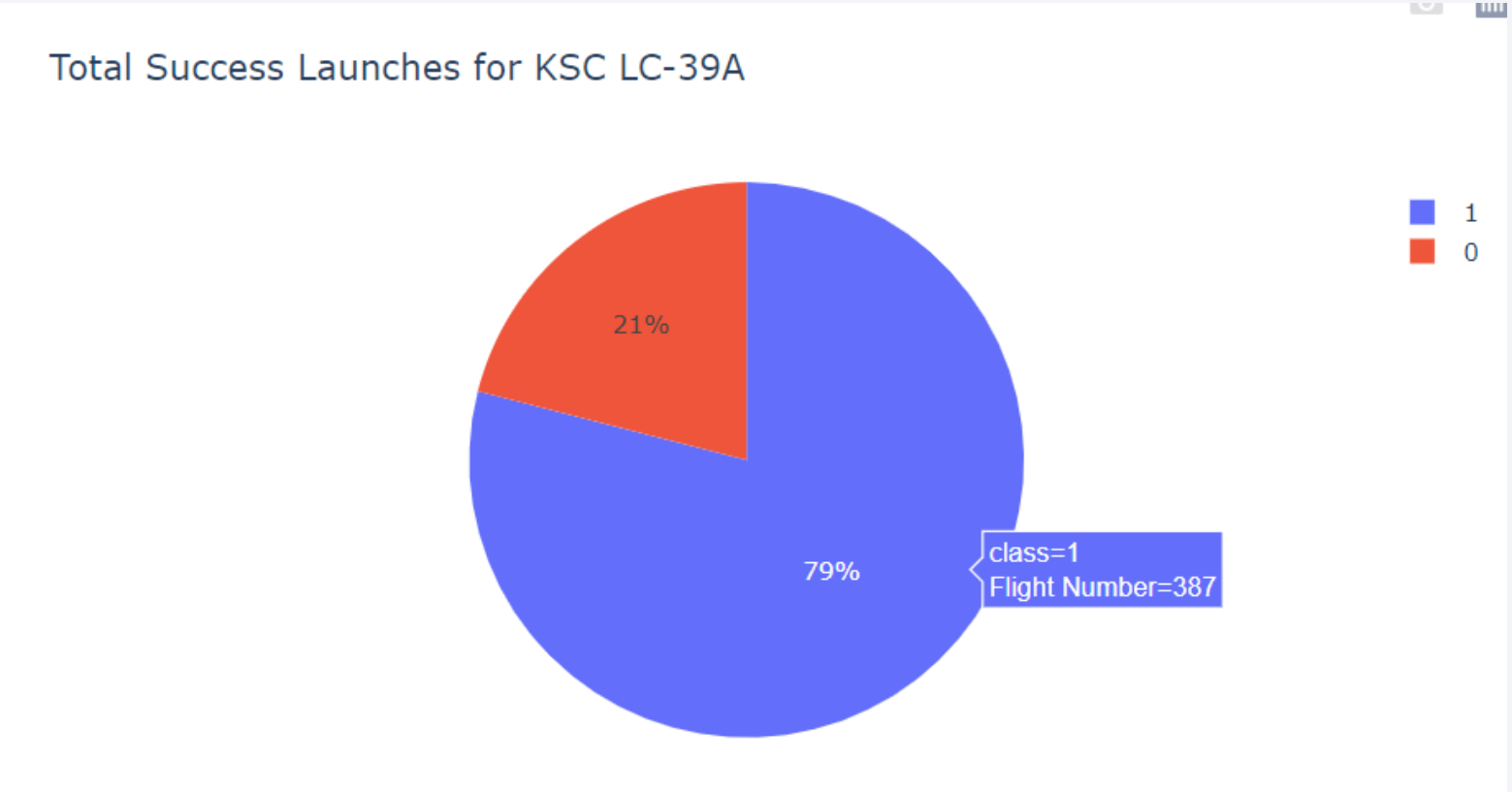
Dashboard: Successful Launches by Site

The Plotly Dashboard I developed for this project allows endusers to interact with data visually.

Here, we can see that a plurality of launches have occurred at Kenney Space Center (KSC).

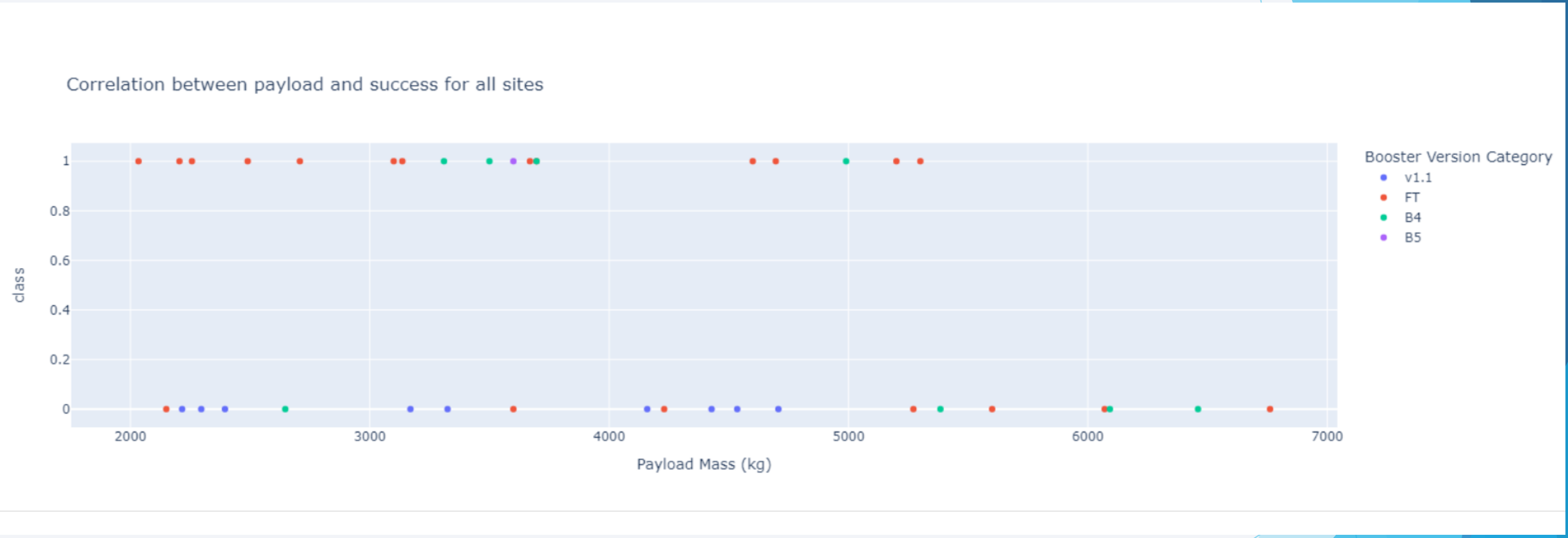


Dashboard: Launch Site With Highest Success Rate



- The Plotly Dashboard I've designed also allows users to filter results by launch site. Here, we see the success/failure ratio for Kennedy Space Center (the site with the highest success rate).

Dashboard: Payload vs. Launch Outcome



The dashboard also provides end-users with the capacity to zero on payload ranges that interest them. Here, we see a scatter plot showing the relationship between payload and launch outcome for all sites.

Dashboard: Try it Yourself

If you'd like to explore the data dashboard and see its capacities for yourself, you can launch it from Google Colab, using the following Github link:

https://github.com/brendanpshea/ibm-data-science/blob/main/Build_a_Dashboard_Application_with_Plotly_Dash.ipynb

SciPy

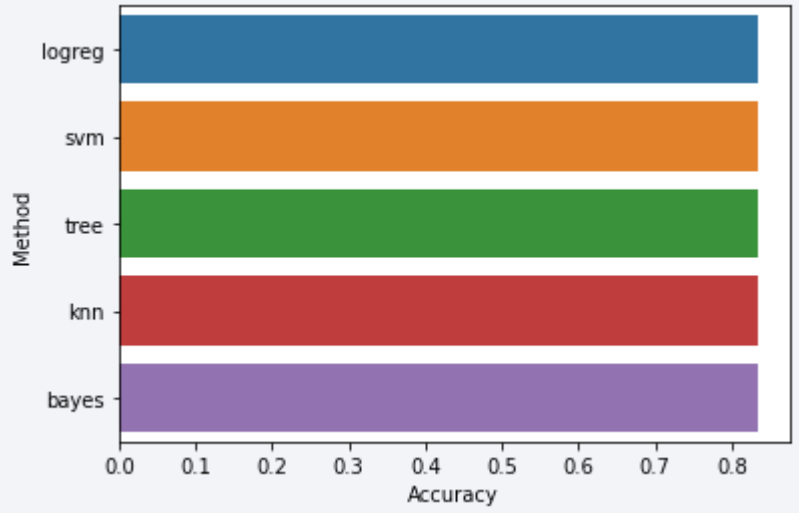
Section 4: Predictive Analysis and Classification

This section provides an overview of the machine methods that were used, and the results of these methods.

Classification Accuracy

The following bar chart shows the accuracy of the machine learning methods used on the "Test" set of data. On this small data set, the choice of method didn't make a difference.

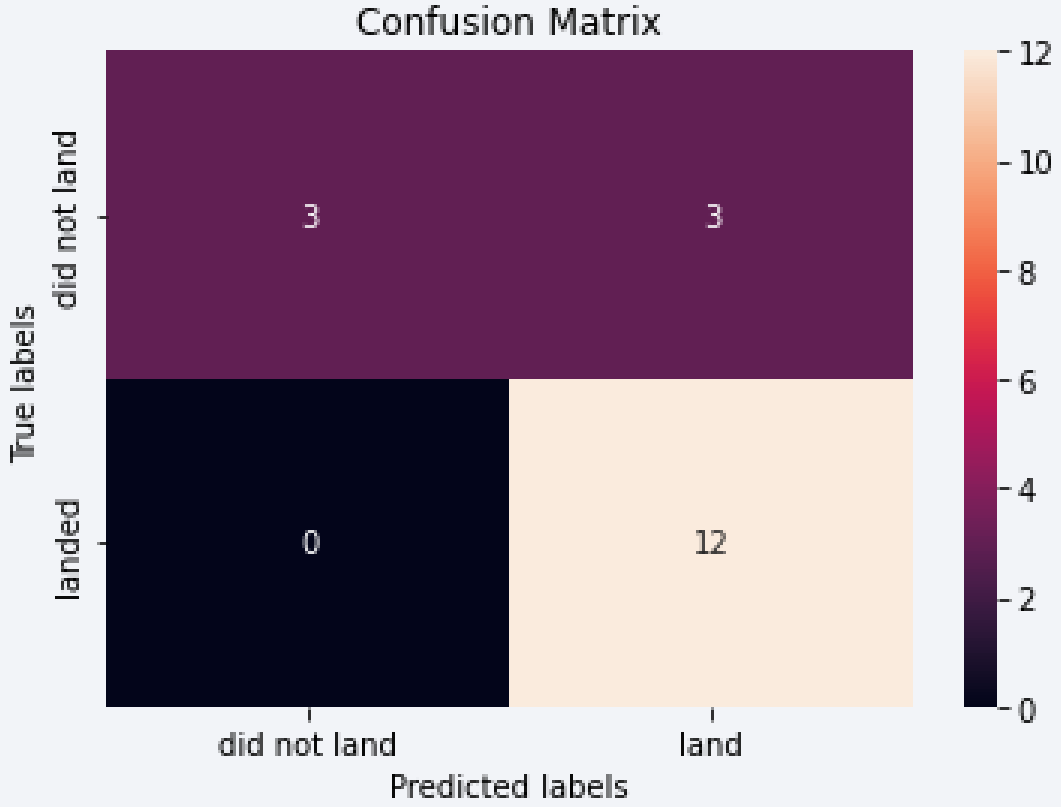
Github: https://github.com/brendanpshea/ibm-data-science/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.ipynb



	Method	Accuracy
0	logreg	0.833333
1	svm	0.833333
2	tree	0.833333
3	knn	0.833333
4	bayes	0.833333

Confusion Matrix: Logistic Regression

The confusion matrix for logistic regression shows all errors are “false positives” where success was predicted and failure was actually the case.



Conclusions

- Data science methods are ideally suited to the sort of predictive analytics questions addressed here, such as “Will a launch be successful, given what we know of factors X1, X2, etc.?”
- Traditional methods of data analysis (statistics, SQL) can and should be incorporated with new “machine learning” methods if we want to produce informative results.
- In order to be meaningful *for users,* the design of static visualizations and interactive dashboards is crucial.
- Sample size matters! The machine learning methods yielded little insight over what could be learning from something like multiple regression or naïve bayes. In larger samples, though, this could change.

Appendix and Acknowledgements

All code for this project is available at:
https://github.com/brendanpshea/ibm-data-science

Thanks so much to the IBM Team that put together this excellent set of courses! In particular, I'd like to thank Alex Aklson, Aije Egwaikhide, Svetlana Levitan, Azim Hirjani, Hima Vasudevan, Joseph Santarcangelo, Rav Ahuja, Romeo Kienzler, Saishruthi Swaminathan, Saeed Aghabozorgi, and Yan Luo.

Thanks also to my fellow students—the discussion board discussions were a great resource for working through tricky problems.

Thank You!