IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Stephen Spivey
August 12, 2024

Outline

Executive Summary

Methodologies	Results
Logistic Regression	**Best parameters (Train):** {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'} **Accuracy (Train):** 0.846429 **Accuracy (Test):** 0.833333
Support Vector Machine	**Best parameters (Train):** {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'} **Accuracy (Train):** 0.848214 **Accuracy (Test):** 0.833333
Decision Tree	**Best parameters (Train):** {'criterion': 'entropy', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'splitter': 'random'} **Accuracy (Train):** 0.875 **Accuracy (Test):** 0.611111
K-Nearest Neighbor	**Best parameters (Train):** {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1} **Accuracy (Train):** 0.848214 **Accuracy (Test):** 0.833333

- The Logistic Regression, Support Vector Machine and K-Nearest Neighbor models performed equally well, as measured by the Accuracy score on Test data.
  - The best parameters for each model are shown as Best parameters (Train), based on hyperparameter tuning using the GridSearchCV function.
- The Decision Tree model did not perform as well as the other three, as measured by the Accuracy score on Test data.

Introduction

Project Background	Objectives
• SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars • Other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage	• Develop a machine learning classification model to predict if the first stage of future launches will land, allowing us to determine their cost • This information can be used if an alternate company wants to bid against SpaceX for a rocket launch

Section 1

Methodology

Methodology

Methodologies	Results
Data Collection	• SpaceX launch data downloaded from API • Launch records scraped from Wikipedia using BeautifulSoup
Data Wrangling	• Missing values replaced • Number/occurrences of launches, orbits, mission outcomes calculated • Landing outcome feature created
Exploratory Data Analysis	• SQL queries displaying launch site data, average/total payload mass, dates, booster versions, mission/landing outcome statistics • Cat plots, bar plots, scatter plots, line plots visualize relationship between independent variables and landing outcome • Folium map identifying launch sites with related outcome statistics and points of interest
Modeling/Selection	• Data preprocessed using StandardScaler • Train/test datasets formed • Best hyper-parameters, accuracy score and Confusion Matrix for four classification algorithms

Data Collection

dataset_part_1

SpaceX API

- Decode the response content as a Json using .json() and turn it into a Pandas data frame using .json_normalize().
- Only keep features: 'rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc'.
- Define global variables, apply custom functions, create new data frame.
- Filter new data frame to show only Falcon 9 launches.
- Handle missing values, reset index (data wrangling).

dataset_part_2

SpaceX API

- Handle missing values, reset index (data wrangling)

dataset_part_3

SpaceX API

- Feature engineering
- Encode categorical variables

spacex_web_scraped

Wikipedia

- Request the Falcon9 Launch Wiki page from its URL.
- Extract all column/variable names from the HTML table header.
- Create a data frame by parsing the launch HTML tables.
- Export to .csv.

my_data1.db

SQL

- Create table SPACEXTBL from SQL database for EDA.

Data Collection – SpaceX API

(1) Request rocket launch data from SpaceX API at the specified URL.
(2) Make the API request.
(3) Check the content of the response.
(4) Decode the response content as a Json using .json().
(5) Use json_normalize method to convert the json result into a data frame.

Use the API again to get information about the launches using the IDs given for each launch. Specifically use columns rocket, payloads, launchpad, and cores.

(6) Take a subset of our data frame keeping only the features we want. Run custom functions to extract data and store in new variables. Add these variables to a dictionary, then create a Pandas data frame with it.
(7) Filter the data frame to only include Falcon 9 launches.
(8) Handle missing values (data wrangling).

(1)
```
spacex_url="https://
api.spacexdata.com/v4/
```

(2)
```
response =
requests.get(spacex_url)
```

(3)
```
print(response.content)
```

(4)
```
json_data = response.json()
```

(5)
```
df = pd.json_normalize(json_data)
```

(6)
```
df_subset = df[['rocket', 'payloads', 'launchpad',
'cores', 'flight_number', 'date_utc']]
```

(7)
```
data_falcon9 = df_launch[df_launch['BoosterVersion']
== 'Falcon 9']
```

(8)
```
data_falcon9.isnull().sum()
```

GitHub URL

https://github.com/stephen-spivey/ibm-spacex-capstone/blob/main/jupyter-labs-spacex-data-collection-

Data Collection - Scraping

(1) Assign to variable the website from which data is collected.

(2) Use requests.get() method with the provided static_url.

(3) Assign the response to a object.

(4) Create a BeautifulSoup object from the HTML response.

(5) Extract all column/variable names from the HTML table header.

(6) Create an empty dictionary with keys from the extracted column names

(7) Fill up the launch_dict with records extracted from table rows.

(8) Create a data frame from launch_dict.

(1) static_url = "https://en.wikipedia.org/w/

(2) response = requests.get(static_url)

(3) falcon9_data = response.text

(4) soup = BeautifulSoup(falcon9_data,

(5) html_tables = soup.find_all('table')

(6) launch_dict= dict.fromkeys(column_names)

(7) for table_number,table in enumerate(soup.find_all('table',"wikitable

(8) df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })

GitHub URL

https://github.com/stephen-spivey/ibm-spacex-capstone/blob/main/jupyter-labs-webscraping.ipynb

Data Wrangling

(1) Load Space X dataset.
(2) Identify and calculate the percentage of the missing values in each attribute.
(3) Identify which columns are numerical and categorical:
(4) Calculate the number of launches on each site.
(5) Calculate the number and occurrence of each orbit.
(6) Calculate the number and occurrence of mission outcome of the orbits.
(7) Create a landing outcome label from Outcome column.

(1) df=pd.read_csv ("dataset_part_1.csv")

(2) df.isnull().sum()/len(df)*100

(3) df.dtypes

(4) df['LaunchSite'].value_counts()

(5) df['Orbit'].value_counts()

(6) landing_outcomes = df['Outcome'].value_counts()

(7) landing_class = [0 if outcome in bad_outcomes else 1 for outcome in

GitHub URL

https://github.com/stephen-spivey/ibm-spacex-capstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

Charts Plotted	Explanation
Catplot visualizing the relationship between Flight Number and Launch Site	Determine if launch success is correlated with earlier or later flights at each site
Catplot visualizing the relationship between Payload Mass and Launch Site	Determine if launch success is correlated with size of payload at each site
Barplot visualizing the relationship between success rate of each orbit type	Identify which orbits have the highest launch success rates
Scatterplot visualizing the relationship between FlightNumber and Orbit type	Determine if specific orbits had greater success with earlier or later launches
Scatterplot visualizing the relationship between Payload Mass and Orbit type	Determine if certain payload sizes had greater success with earlier or later launches
Line chart visualize the launch success yearly trend	Recognize if overall launch success has increased over time

GitHub URL

https://github.com/stephen-spivey/ibm-spacex-capstone/blob/main/edadataviz.ipynb

EDA with SQL

- **SELECT DISTINCT Launch_Site FROM SPACEXTBL;** Display the names of the unique launch sites in the space mission
- **SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5;** Display 5 records where launch sites begin with the string 'CCA'
- **SELECT SUM(PAYLOAD_MASS__KG_) AS TotalPayloadMass FROM SPACEXTBL WHERE Customer = 'NASA (CRS)'** Display the total payload mass carried by boosters launched by NASA (CRS)
- **SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version = 'F9 v1.1'** Display average payload mass carried by booster version F9 v1.1
- **SELECT MIN(Date) AS EarliestDate FROM SPACEXTBL WHERE Landing_Outcome LIKE '%(ground pad)' AND Mission_Outcome = 'Success';** List the date when the first succesful landing outcome in ground pad was achieved
- **SELECT Booster_Version FROM SPACEXTBL WHERE Landing_Outcome LIKE '%(drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;** List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- **SELECT CASE WHEN Mission_Outcome LIKE '%success%' THEN 'Successful' ELSE 'Failures' END AS OutcomeCategory, COUNT(*) AS TotalCount FROM SPACEXTBL GROUP BY OutcomeCategory;** List the total number of successful and failure mission outcomes
- **SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);** List the names of the booster_versions which have carried the maximum payload mass
- **SELECT substr(Date, 6,2), Mission_Outcome, Booster_Version, Launch_Site FROM SPACEXTBL WHERE substr(Date,0,5)='2015'** List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015
- **SELECT Landing_Outcome, COUNT(*) AS OutcomeCount FROM SPACEXTBL WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY OutcomeCount DESC;** Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

GitHub URL

https://github.com/stephen-spivey/ibm-spacex-capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite%20(1).ipynb

Build an Interactive Map with Folium

Objects Added	Explanation
Circle added to mark all launch sites	Supports analysis of launch site data
Success/failed launches for each site	Determine which sites have high success rates
Distance label between a launch site to its proximities	Identify geographic features of existing launch sites
Line between a launch site and the coastline	Analyze characteristics of successful launch sites
Line between a launch site to its closest city	Analyze characteristics of successful launch sites

GitHub URL

https://github.com/stephen-spivey/ibm-spacex-capstone/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

Objects Added	Explanation
Dropdown list for Launch Site selection	Launch Site selection
Pie chart to show the total successful launches count	Show the count of total successful launches
Slider to select payload range	Payload Range selection
Scatter chart to show correlation between payload and launch success	Show the correlation between payload and launch success
Callbacks (2)	Update pie chart; update scatter chart

GitHub URL

https://github.com/stephen-spivey/ibm-spacex-capstone/blob/main/spacex_dash_app-2.py

Predictive Analysis (Classification)

(1) Load data.

(2) Create a NumPy array from the column Class in data, by applying the method to_numpy() then assign it to the variable Y.

(3) Standardize the data in X then reassign it to the variable X.

(4) Split the data X and Y into training and test data.

(5) Create logistic regression, GridSearchCV objects. Fit objects to find the best parameters, accuracy scores.

(6) Create support vector machine, GridSearchCV objects. Fit objects to find the best parameters, accuracy scores.

(7) Create decision tree, GridSearchCV objects. Fit objects to find the best parameters, accuracy scores.

(8) Create K-nearest neighbors, GridSearchCV objects. Fit objects to find the best parameters, accuracy scores.

(1) **df = pd.read_csv('dataset_part_2.csv')**

(2) **Y = pd.Series(df['Class'] .to_numpy())**

(3) **X = transform.fit_ transform(df)**

(4) **X_train, X_test, Y_train, Y_test = train_test_split(X, Y,**

(5) **logreg_cv.fit(X_train,Y_train)**

(6) **svm_cv.fit(X_train,Y_train)**

(7) **tree_cv.fit(X_train,Y_train)**

(8) **knn_cv.fit(X_train,Y_train)**

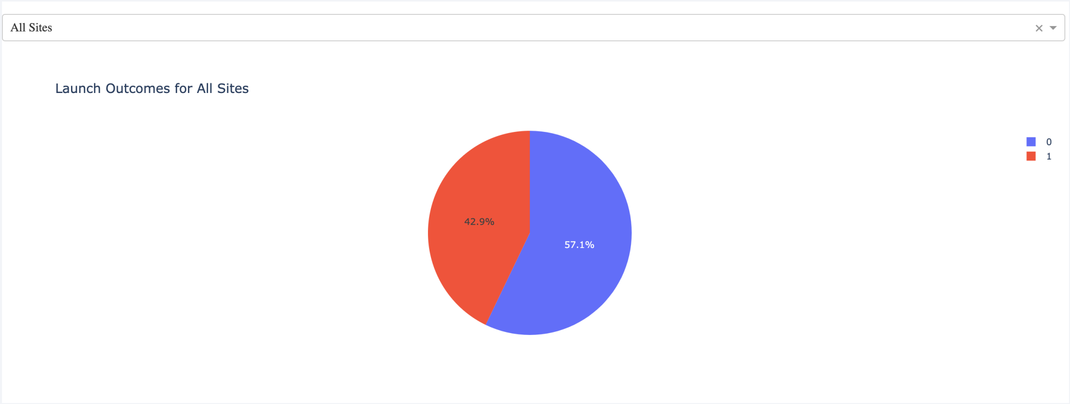
GitHub URL

https://github.com/stephen-spivey/ibm-spacex-capstone/blob/main/

Results

Exploratory data analysis results

- Success rates for Falcon 9 launches have increased steadily over time.
- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS. Interestingly, Polar, LEO and ISS also have the highest failure rates for light payloads.



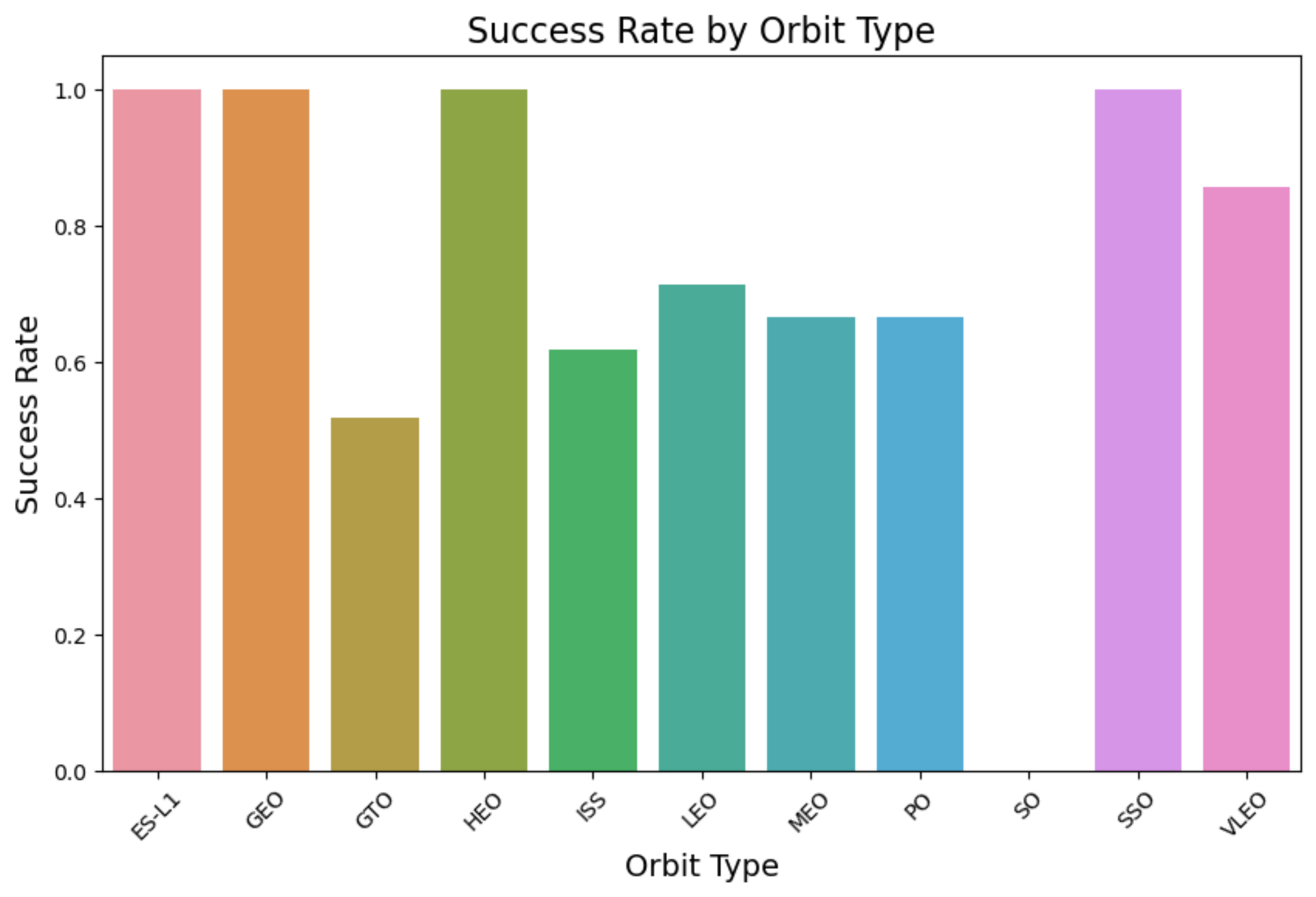
Predictive analysis results

- Logistic Regression, support vector machine and K-nearest neighbor models perform the same on the accuracy score (0.83)
- Decision tree perform significantly lower when it comes to accuracy.
- accuracy : 0.833

Section 2

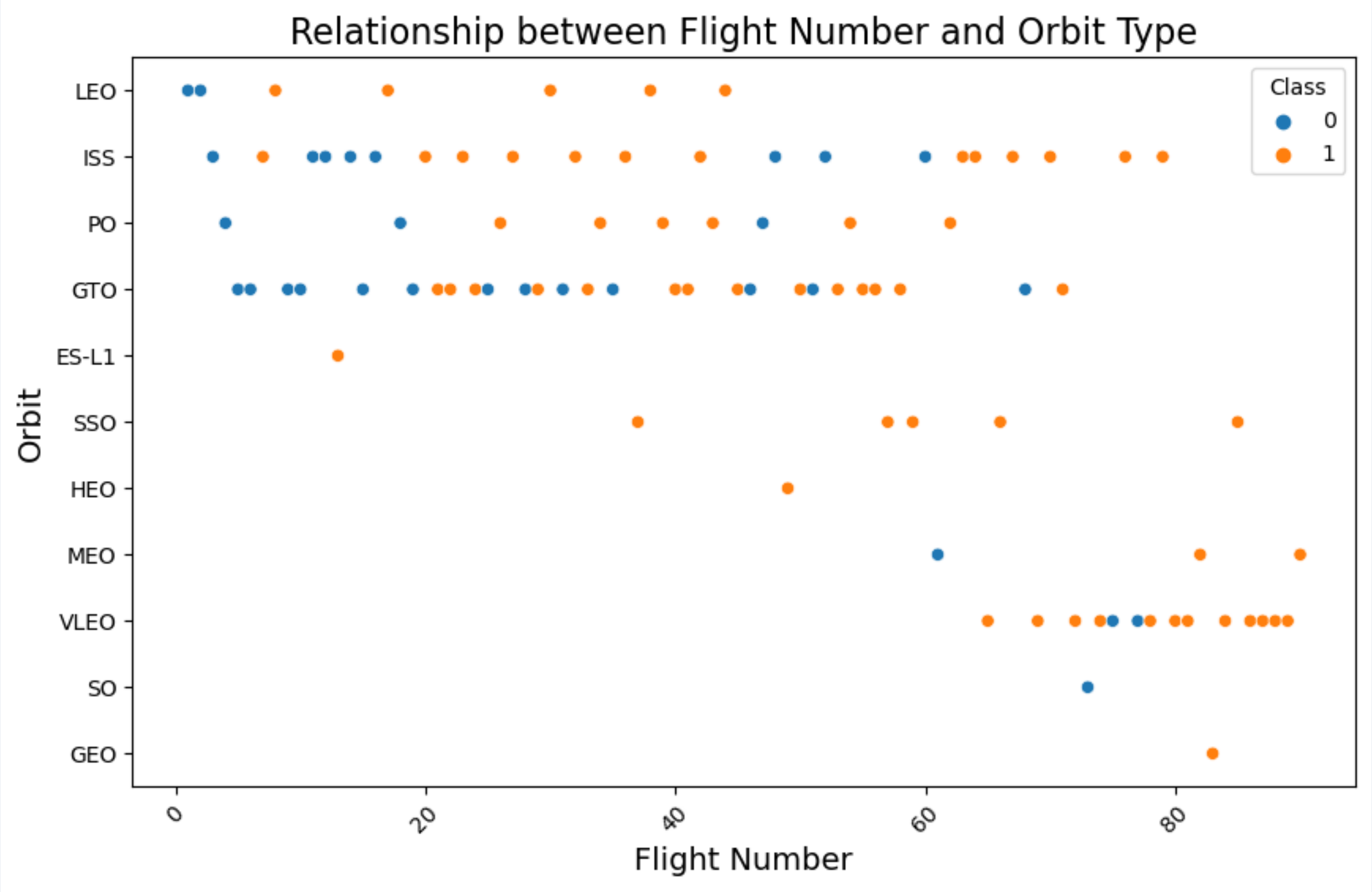
Insights drawn from EDA

Success Rate vs. Orbit Type



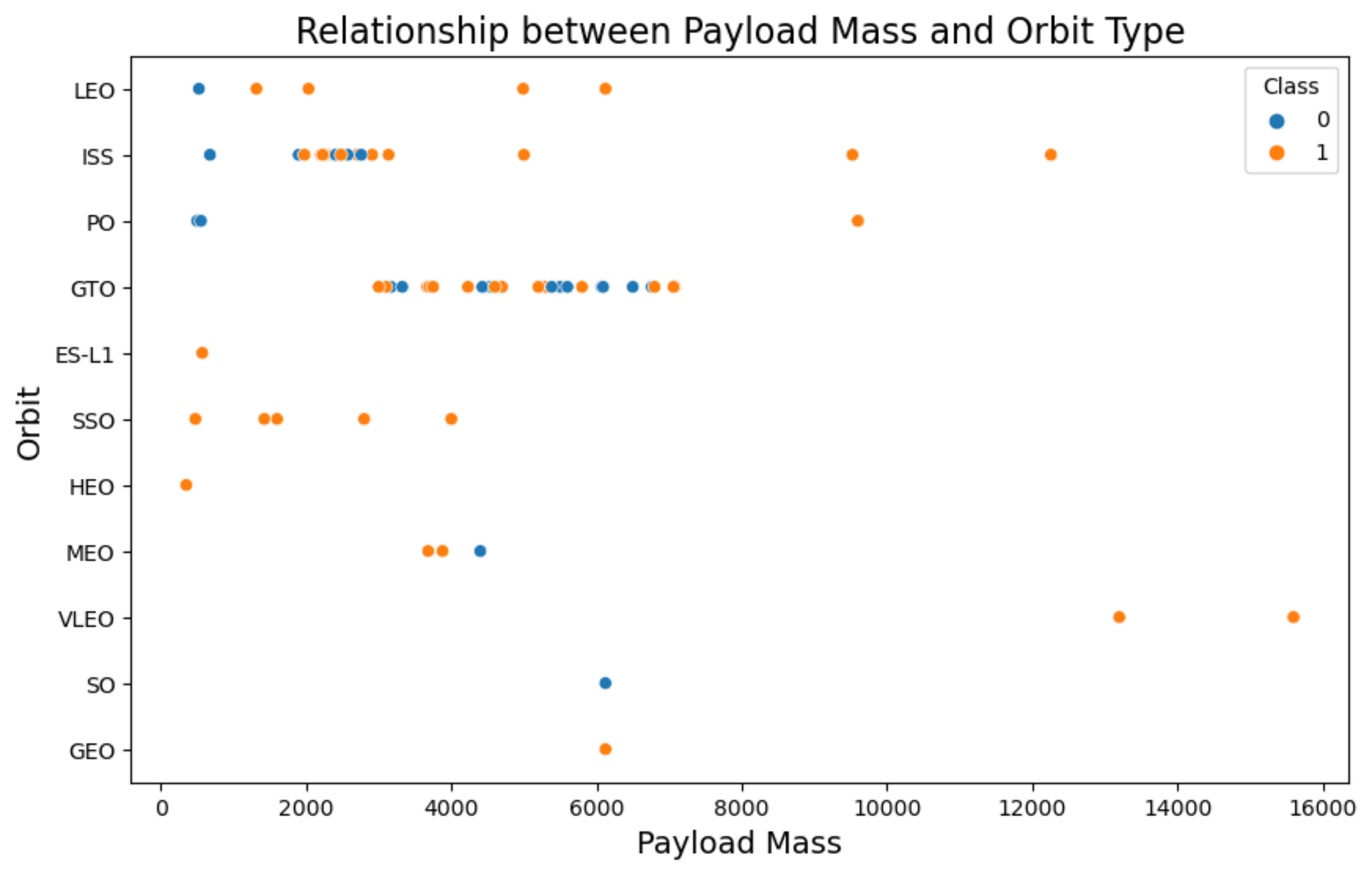
- ES-L1, GEO, GTO AND SSO orbits have the highest success rates.
- 100% of these orbits have landed successfully.
- GTO has the lowest success rate.
- Only about half of launches into this orbit have landed successfully.

Flight Number vs. Orbit Type



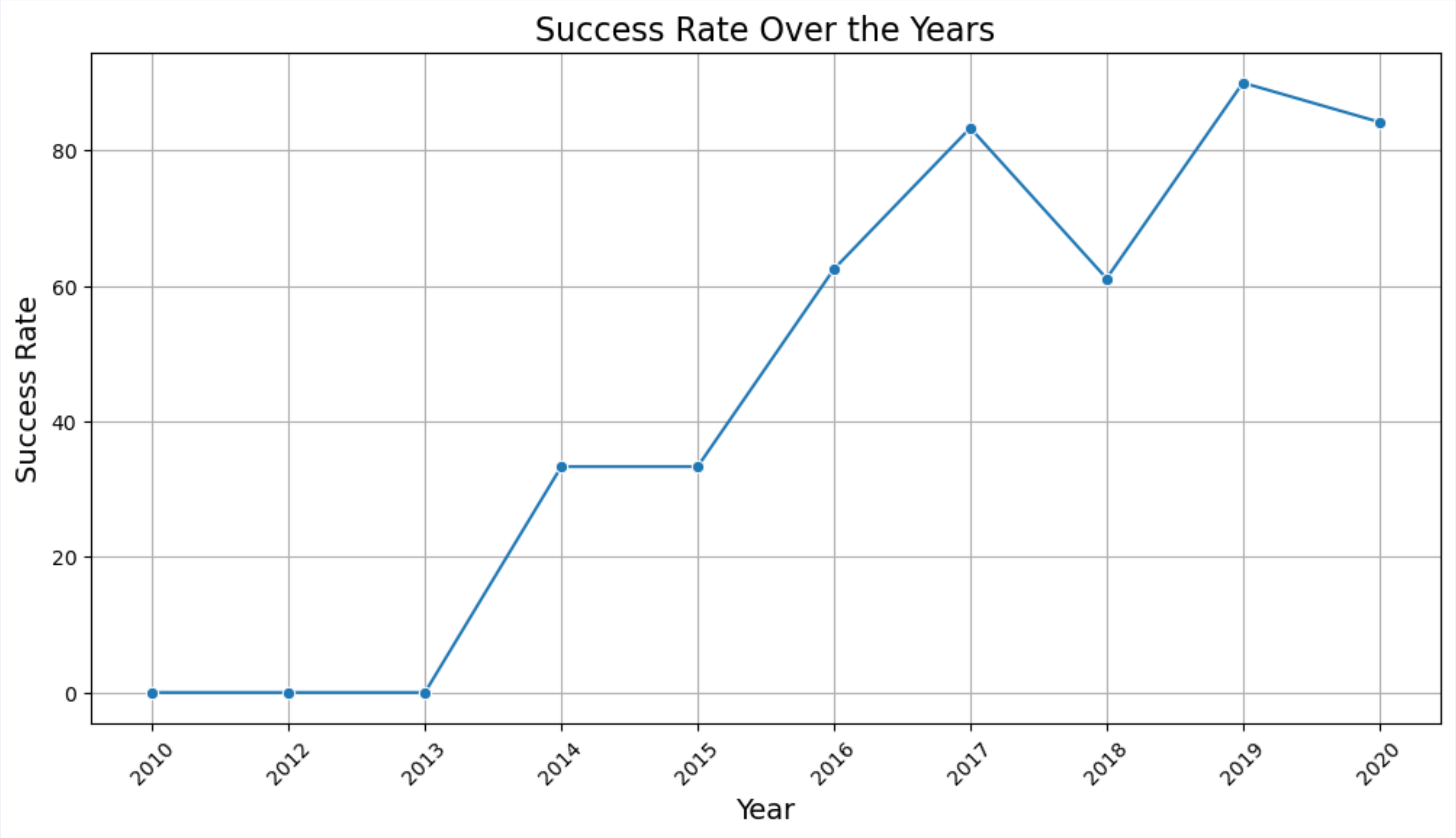
- In most orbits (LEO, VLEO, SSO, MEO, GTO), successful landings (Class 1) increased with later launches.
- LEO, ISS and GTO orbits have had the most launches.
- Several orbits (ES-L1, HEO, SO, GEO) have only had one launch.

Payload vs. Orbit Type



- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.
- Interestingly, Polar, LEO and ISS also have the highest failure rates for light payloads.

Launch Success Yearly Trend



- Success rates have increased steadily over the 2013-2020 period.
- The exception was in 2018, when success rates fell from above 80% to around 60%.
- Success rates also dipped slightly in 2020.

All Launch Site Names

- **Find the names of the unique launch sites**
- %sql SELECT DISTINCT Launch_Site FROM SPACEXTBL;
- Explanation:
  - SELECT DISTINCT Launch_Site: Retrieves unique values from the Launch_Site column.
  - FROM SPACEXTBL: Specifies the table from which to retrieve the data.

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

Launch Site Names Begin with 'CCA'

- **Find 5 records where launch sites begin with `CCA`**
- %sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
- Explanation:
  - SELECT *: Retrieves all columns from the table.
  - WHERE Launch_Site LIKE 'CCA%': Filters the records to include only those where Launch_Site starts with 'CCA'. The %wildcard allows for any sequence of characters following ‘CCA'.
  - LIMIT 5: Limits the output to the first 5 records that meet the criteria.

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

- **Calculate the total payload carried by boosters from NASA**
- %sql SELECT SUM(PAYLOAD_MASS__KG_) AS TotalPayloadMass FROM SPACEXTBL WHERE Customer = 'NASA (CRS)'
- Explanation:
  - SUM(PAYLOAD_MASS__KG_): Calculates the sum of the PAYLOAD_MASS__KG_ column.
  - WHERE Customer = 'NASA (CRS)': Filters the rows to include only those where the Customer column is 'NASA (CRS)'.

TotalPayloadMass
45596

Average Payload Mass by F9 v1.1

- **Calculate the average payload mass carried by booster version F9 v1.1**
- %sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version = 'F9 v1.1'
- Explanation:
  - AVG(PAYLOAD_MASS__KG_): Computes the average value of the PAYLOAD_MASS__KG_ column.
  - WHERE Booster_Version = 'F9 v1.1': Filters the rows to include only those where the Booster_Version is 'F9 v1.1'.

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- **Find the dates of the first successful landing outcome on ground pad**
- %sql SELECT MIN(Date) AS EarliestDate FROM SPACEXTBL WHERE Landing_Outcome LIKE '%(ground pad)' AND Mission_Outcome = 'Success';
- Explanation:
  - MIN(Date): Finds the earliest date from the Date column.
  - WHERE Landing_Outcome LIKE '%(ground pad)': Filters the rows to include only those where Landing_Outcomecontains '(ground pad)'.
  - AND Mission_Outcome = 'Success': Further filters to include only records where Mission_Outcome is 'Success'

EarliestDate
2015-12-22

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**
- %sql SELECT Booster_Version FROM SPACEXTBL WHERE Landing_Outcome LIKE '%(drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
- Key Points:
  - LIKE '%(drone ship)': Filters for records where Landing_Outcome contains the phrase (drone ship).
  - PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000: Ensures the payload mass is within the specified range.

Booster_Version
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- **Calculate the total number of successful and failure mission outcomes**
- %%sql SELECT CASE WHEN Mission_Outcome LIKE '%success%' THEN ‘Successful' ELSE ‘Failures' END AS OutcomeCategory,COUNT(*) AS TotalCount FROM SPACEXTBL GROUP BY OutcomeCategory;
- What This Query Does:
  - CASE statement: Checks if Mission_Outcome contains the substring 'success'.
  - OutcomeCategory: Labels records as 'Successful' or 'Failures'.
  - COUNT(*): Counts the total number of records in each category.
  - GROUP BY: Groups the results by OutcomeCategory.

OutcomeCategory	TotalCount
Failures	1
Successful	100

Boosters Carried Maximum Payload

- **List the names of the booster which have carried the maximum payload mass**
- %%sql SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
- Explanation:
  - SELECT Booster_Version: Retrieves the Booster_Version for the maximum payload mass.
  - WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL): Filters the rows to include only those where the PAYLOAD_MASS__KG_ is equal to the maximum payload mass found in the SPACEXTBL table.

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

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- %%sql SELECT substr(Date, 6,2), Mission_Outcome, Booster_Version, Launch_Site FROM SPACEXTBL WHERE substr(Date,0,5)='2015'
- Explanation:
  - substr(Date, 6, 2) AS Month: Extracts the month from the Date column. Here, 6 is the starting position, and 2 is the length of the substring.
  - Mission_Outcome, Booster_Version, Launch_Site: Retrieves these columns from the table.
  - WHERE substr(Date, 1, 4) = '2015': Filters records to include only those where the year part of the Date is '2015'. Note that the starting position is 1 and length is 4 to correctly extract the year

substr(Date, 6,2)	Mission_Outcome	Booster_Version	Launch_Site
1	Success	F9 v1.1 B1012	CCAFS LC-40
2	Success	F9 v1.1 B1013	CCAFS LC-40
3	Success	F9 v1.1 B1014	CCAFS LC-40
4	Success	F9 v1.1 B1015	CCAFS LC-40
4	Success	F9 v1.1 B1016	CCAFS LC-40
6	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	CCAFS LC-40

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

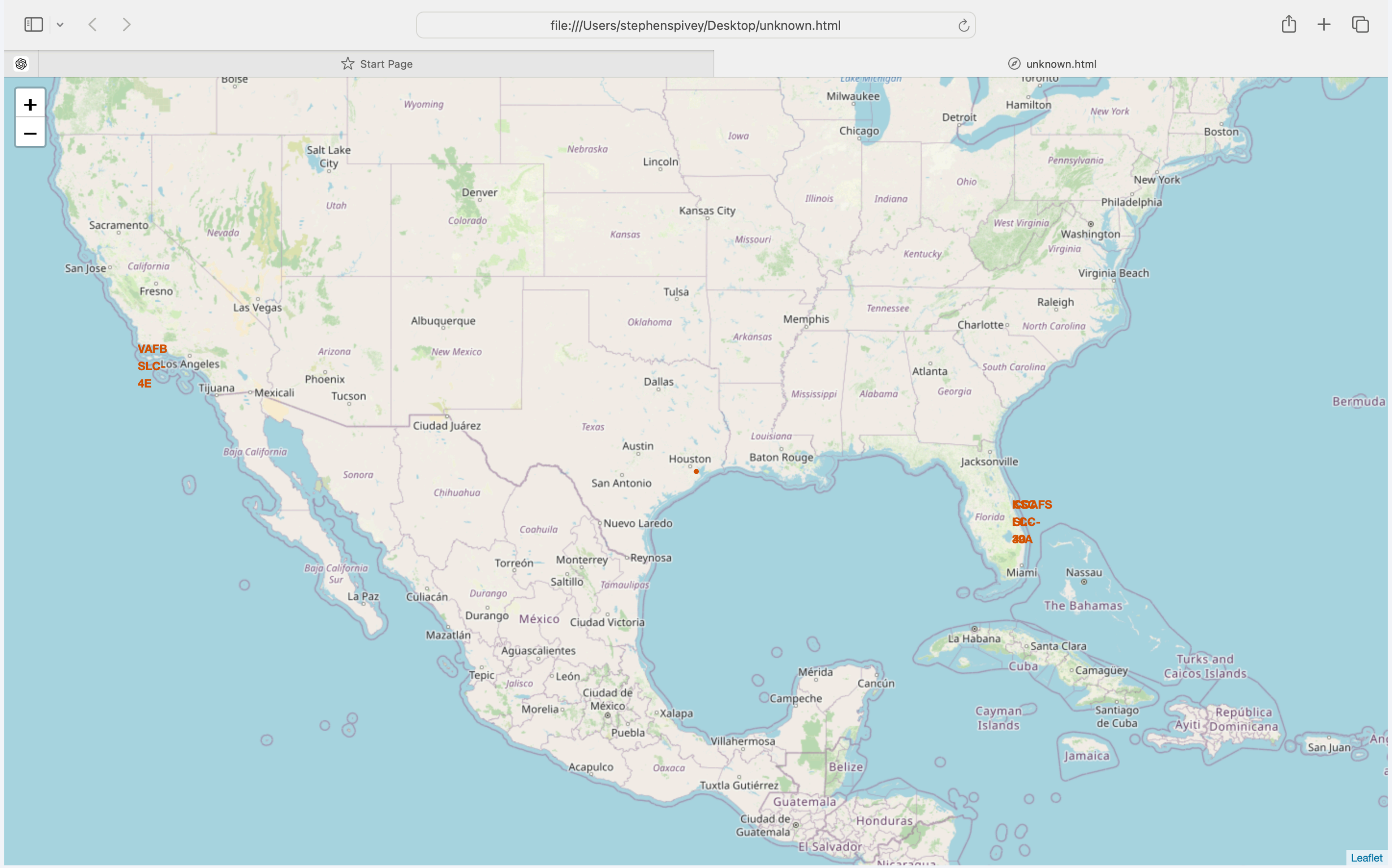
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order
- %%sql SELECT Landing_Outcome, COUNT(*) AS OutcomeCount FROM SPACEXTBL WHERE Date BETWEEN '2010-06-04' AND ‘2017-03-20' GROUP BY Landing_Outcome ORDER BY OutcomeCount DESC;
- Explanation:
  - SELECT Landing_Outcome, COUNT(*) AS OutcomeCount: Selects the Landing_Outcome and counts the number of occurrences, labeling it as OutcomeCount.
  - FROM SPACEXTBL: Specifies the table to query from.
  - WHERE Date BETWEEN '2010-06-04' AND '2017-03-20': Filters the records to include only those where the Datefalls within the specified range.
  - GROUP BY Landing_Outcome: Groups the results by the Landing_Outcome to aggregate the counts for each unique landing outcome.
  - ORDER BY OutcomeCount DESC: Orders the results by the count in descending order, so the most frequent landing outcomes appear first.

Landing_Outcome	OutcomeCount
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

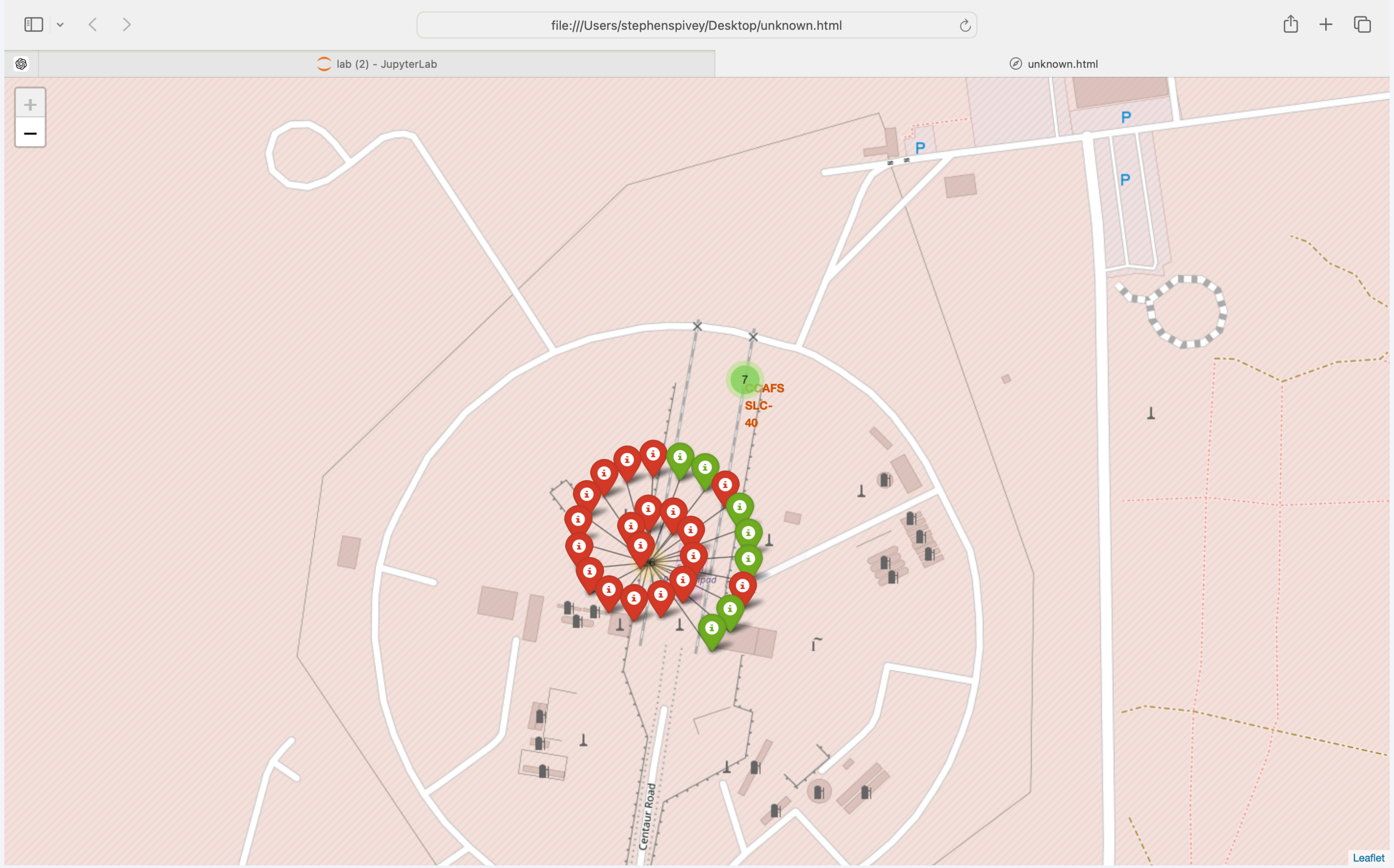
Launch Sites Proximities Analysis

All Launch Sites



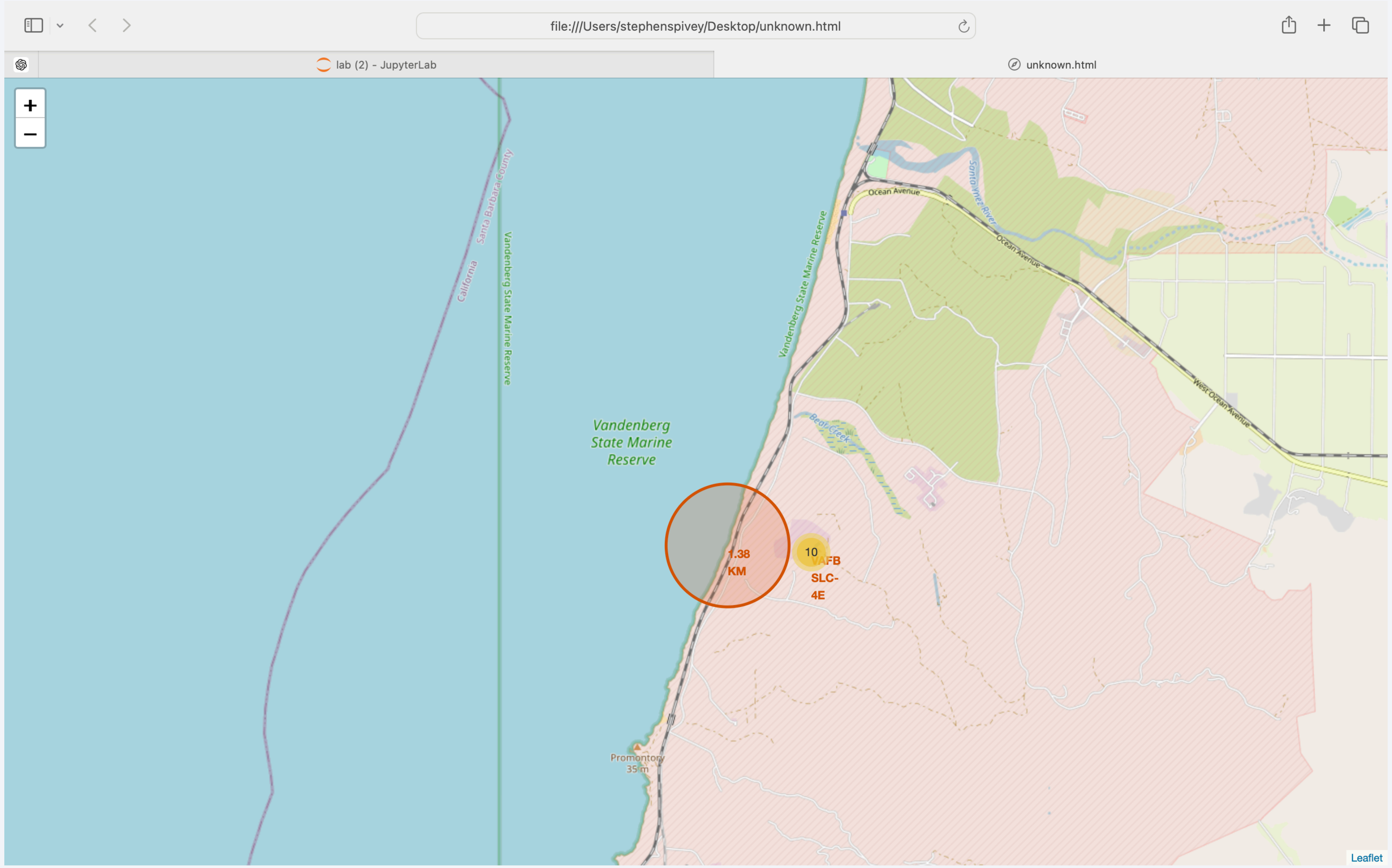
- Cape Canaveral and Kennedy Space Center are located next to each other on the Central Florida coast.
- Vandenberg Space Force Base (VAFB SLC-4E) located in Southern California
- Johnson Space Center in Houston, TX is marked with a small red dot, with no label for the relevant launch platform.

Color-labeled Launch Outcomes



- Closeup view of launch outcomes at one of two Cape Canaveral launchpads.
- Successful launches are shown in green, while failures/bad outcomes are in red.

Launch Site Distance to Proximities

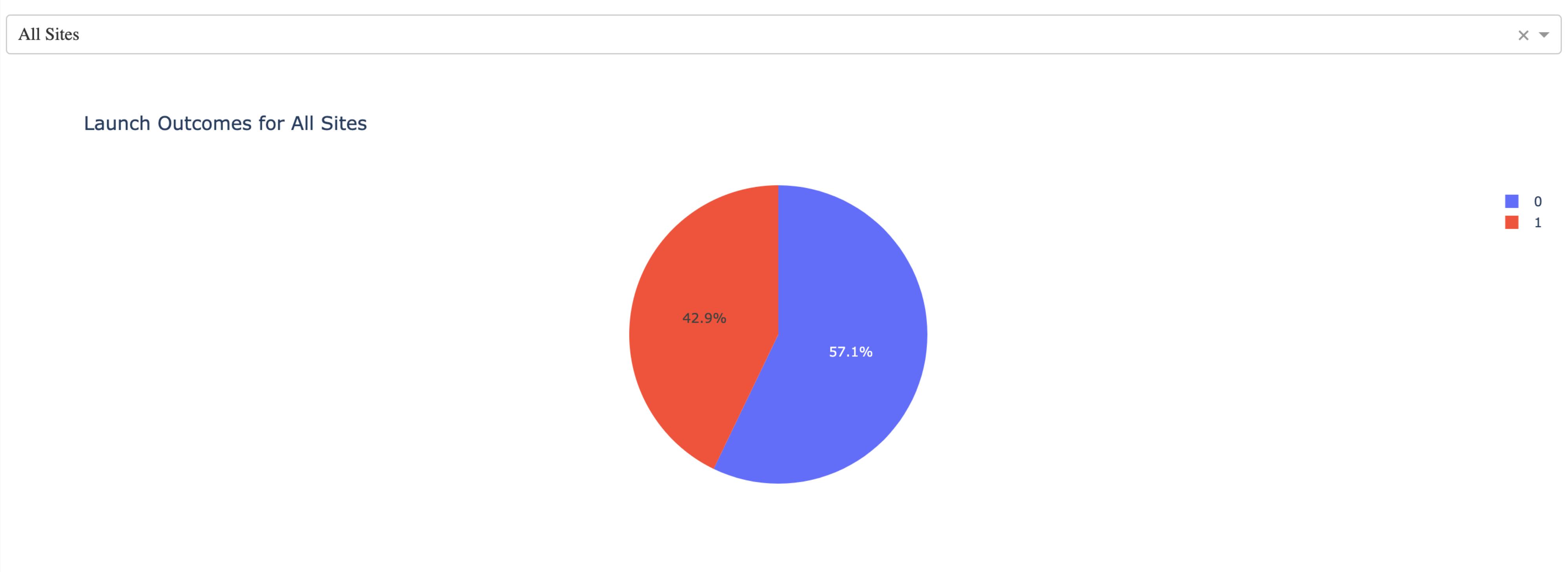


- Proximity of Vandenberg Space Force Base (VAFB SLC-4E) to coastline shown.
- Distance shown as 1.38 km and labeled accordingly.

Section 4

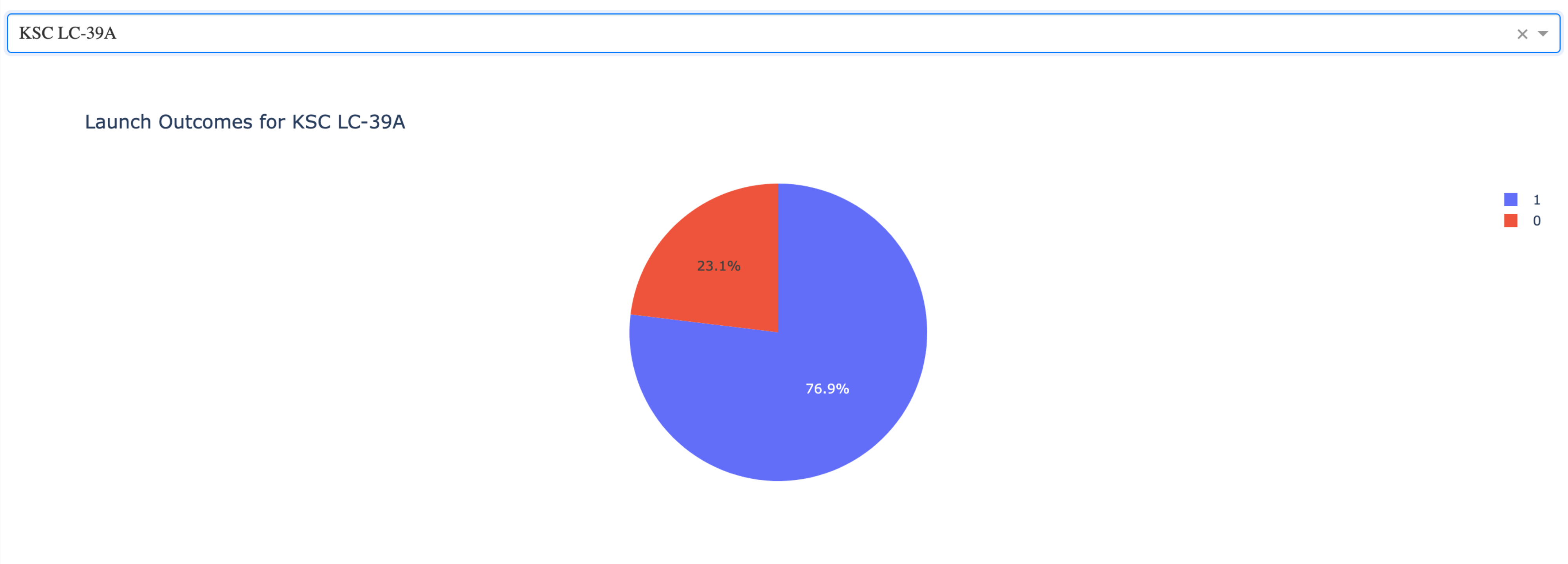
Build a Dashboard with Plotly Dash

Launch Outcomes for All Sites



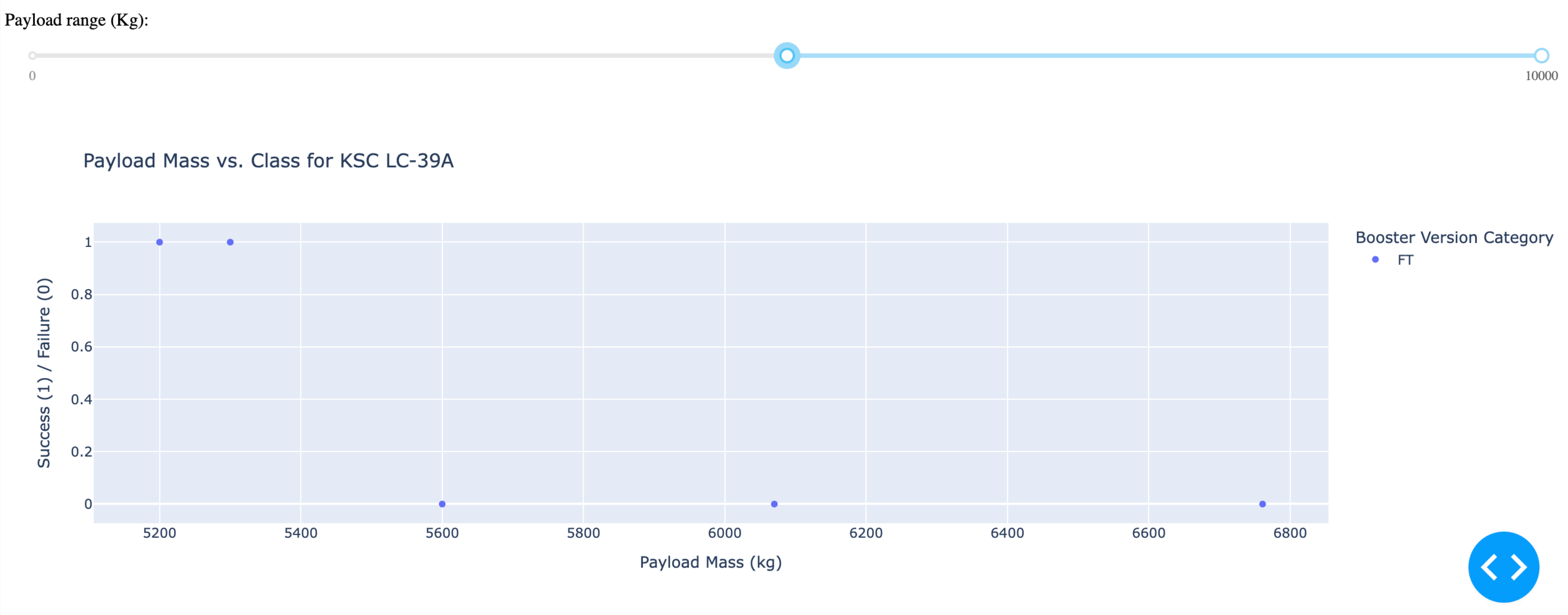
- All four launchpads that host the Falcon 9 rocket have a combined success rate of 57.1%.
- Two sites have a significantly higher success rate (above 70%), which is offset by increased failures at the other launchpads.

Highest Launch Success Ratio



- KSC LC-39E launchpad (Kennedy Space Center) has the highest success ratio for Falcon 9 missions (76.9%).
- This compares to a 57.1% success rate for the four Falcon 9 launchpads combined.

Payload vs. Launch Outcome

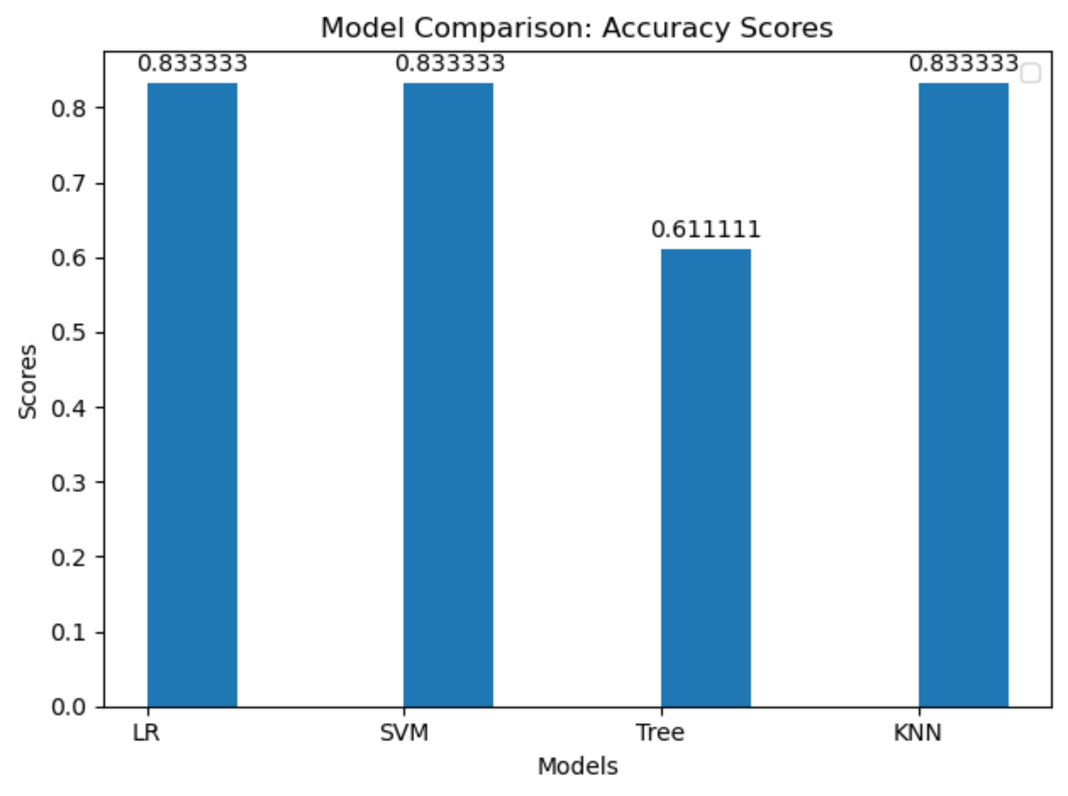


- As it relates to launchpad KSC LC-39A, successful launch rates are higher at the lower range of payloads between 5,200 and 6,800 kg.
- This view can be modified by adjusting the slider shown in the image to highlight different payload ranges.
- The dropdown menu at the top of the dashboard (not shown) allows the view to show different launch pads as well as all sites combined.

Section 5

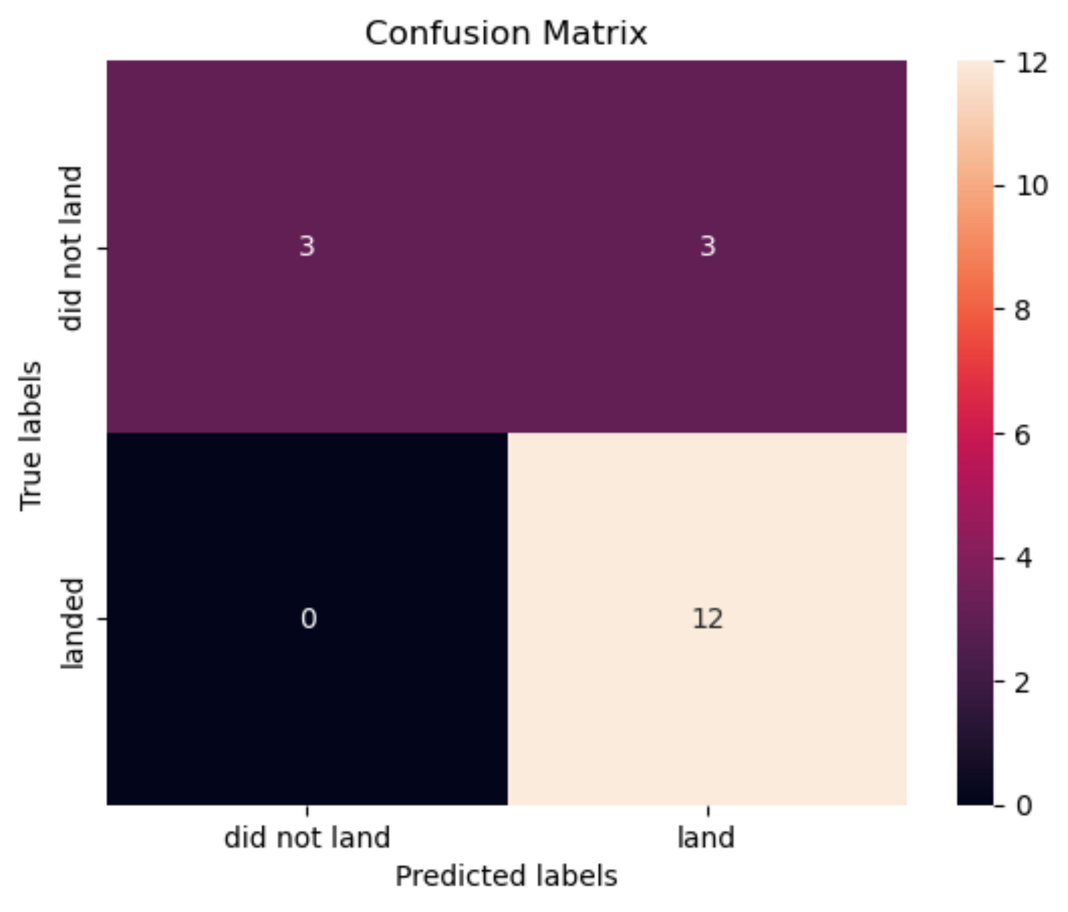
Predictive Analysis (Classification)

Classification Accuracy



- Logistic Regression, support vector machine and K-nearest neighbor models perform the same on the accuracy score (0.83)
- Decision tree perform significantly lower when it comes to accuracy.

Confusion Matrix



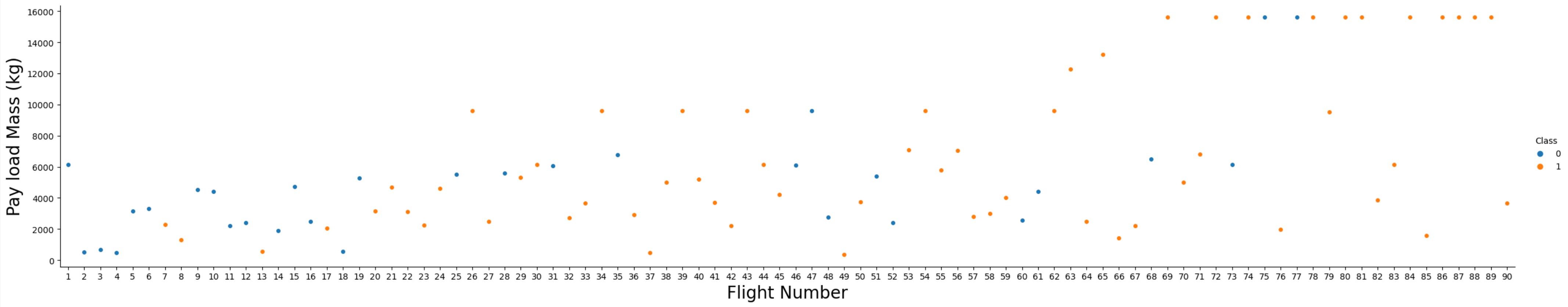
- Confusion matrix for logistic regression, support vector machine and K-Nearest neighbor classification models:
- Accuracy score: 0.8333
- True Positives (TP): 12 cases where the model correctly predicted "landed."
- True Negatives (TN): 3 cases where the model correctly predicted "did not land."
- False Positives (FP): 3 cases where the model incorrectly predicted "landed" when it actually "did not land."
- False Negatives (FN): 0 cases where the model incorrectly predicted "did not land" when it actually "landed."

Conclusions

- Success rates for Falcon 9 launches have increased steadily over time.
- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS. Interestingly, Polar, LEO and ISS also have the highest failure rates for light payloads.
- Logistic regression, support vector machine and K-nearest neighbor models with have higher accuracy than decision tree:
  - accuracy : 0.833 vs. 0.611

Appendix

Relationship between Flight Number and Payload Mass (Kg)



NOTE: Class 0 highlights failed launches while Class 1 marks successful ones.

Thank you!