IBM Developer
SKILLS NETWORK

SpaceX Falcon9 First Stage Landing Prediction

Name: Nalin Malla
Date: 2023-12-03

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

This project aims to understand how SpaceX a market leading aerospace company is able to gain competitive edge by significantly reducing their per flight cost using proprietary technologies which allow the first stage of their Falcon9 rocket to land.

We also choose the best Machine Learning prediction models used to predict the success of Falcon9 launch. This can be useful for SpaceX competitors if they want to bid against SpaceX for a rocket launch.

Throughout the duration of this project I performed various activities such as: Data Collect, Data Wrangling, Exploratory Data Analysis, Feature Engineering, Data Visualization, Dashboard creation, Predictive Analysis and Documentation.

The experience from this project has provided valuable insights into the nuances of data science methodology. The culmination of these efforts has led to the growth in my technical skills.

Introduction

Project background and context:

- This project was done for the fulfillment of "IBM: Applied Data Science Capstone" certification.
- SpaceX is an market leading Aerospace company which has dominated the market due to their proprietary technology which allow the first stage of their Falcon 9 rocket to land.
- This allows the first stage of the rocket to be reused significantly reducing their cost.
- SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards of 165 million dollars each.

Problems you want to find answers:

- How are various variables related to mission success i.e. successful landing of first stage?
- Which factors influence success and there signification?
- Which Machine Learning Prediction model is the best for predicting the success of Falcon 9 rockets?

Section 1

Methodology

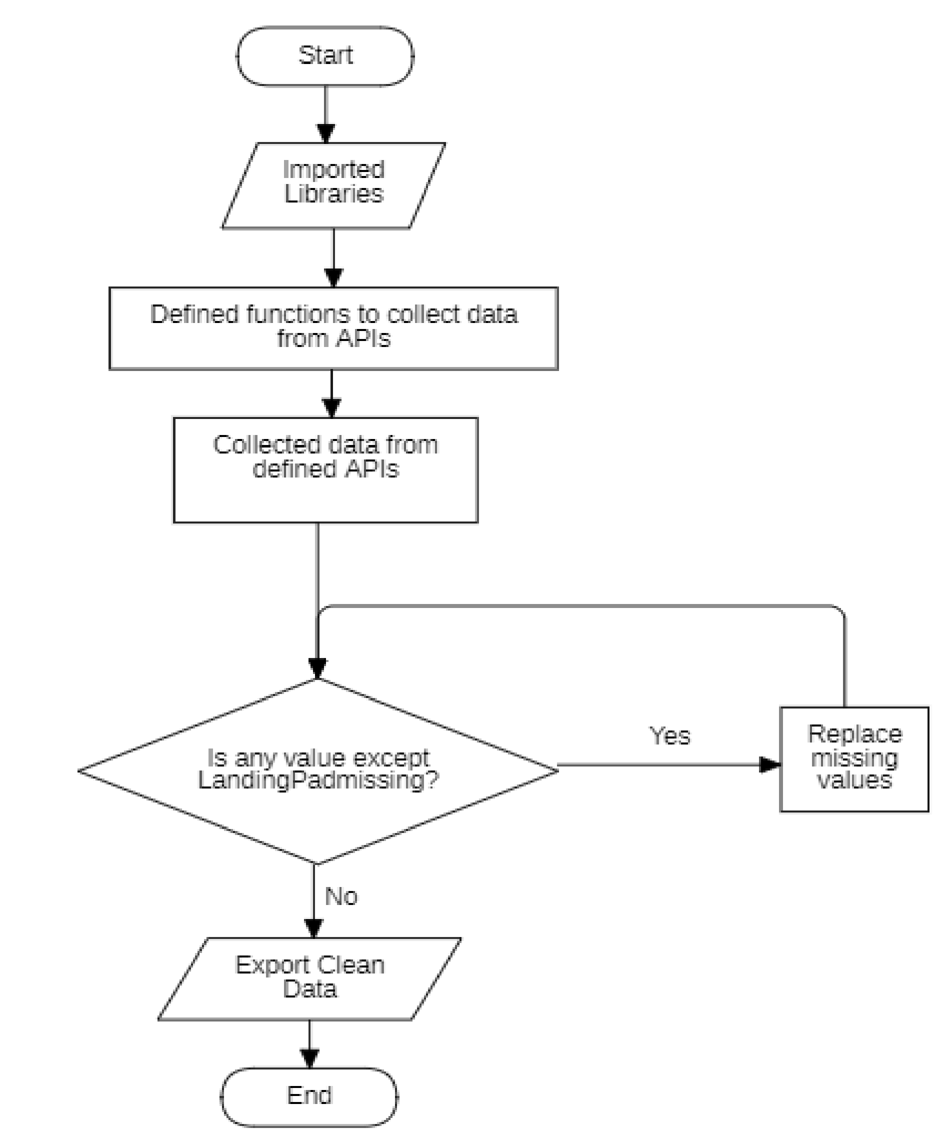
Methodology

During this project I performed the following operations:

- **Data collection:**
    - Data was collected via the SpaceX APIs and webscraping the wikipage of spacex falcone rockekts.
- **Perform data wrangling:**
    - Here, I mainly mapped mission outcomes of various categories like. True ASDS, True RTLS, etc into clear binary value of 0 & 1 for machine learning models and also made data analysis easier.
    - Standardization
    - Feature Engineering using One Hot Encoding

- **Perform exploratory data analysis (EDA) using visualization and SQL:**
  - To better understand the data.
  - To identify patterns and relationship between various variables
- **Perform interactive visual analytics using Folium and Plotly Dash:**
  - To understand relationship between various variables
  - To gain insight on landing sites using Geospatial maps.
- **Perform predictive analysis using classification models:**
  - Tested different classification models with various parameter value using GridSearchCV to find:
    - The best parameters for each models.
    - The best models among them.

Data Collection



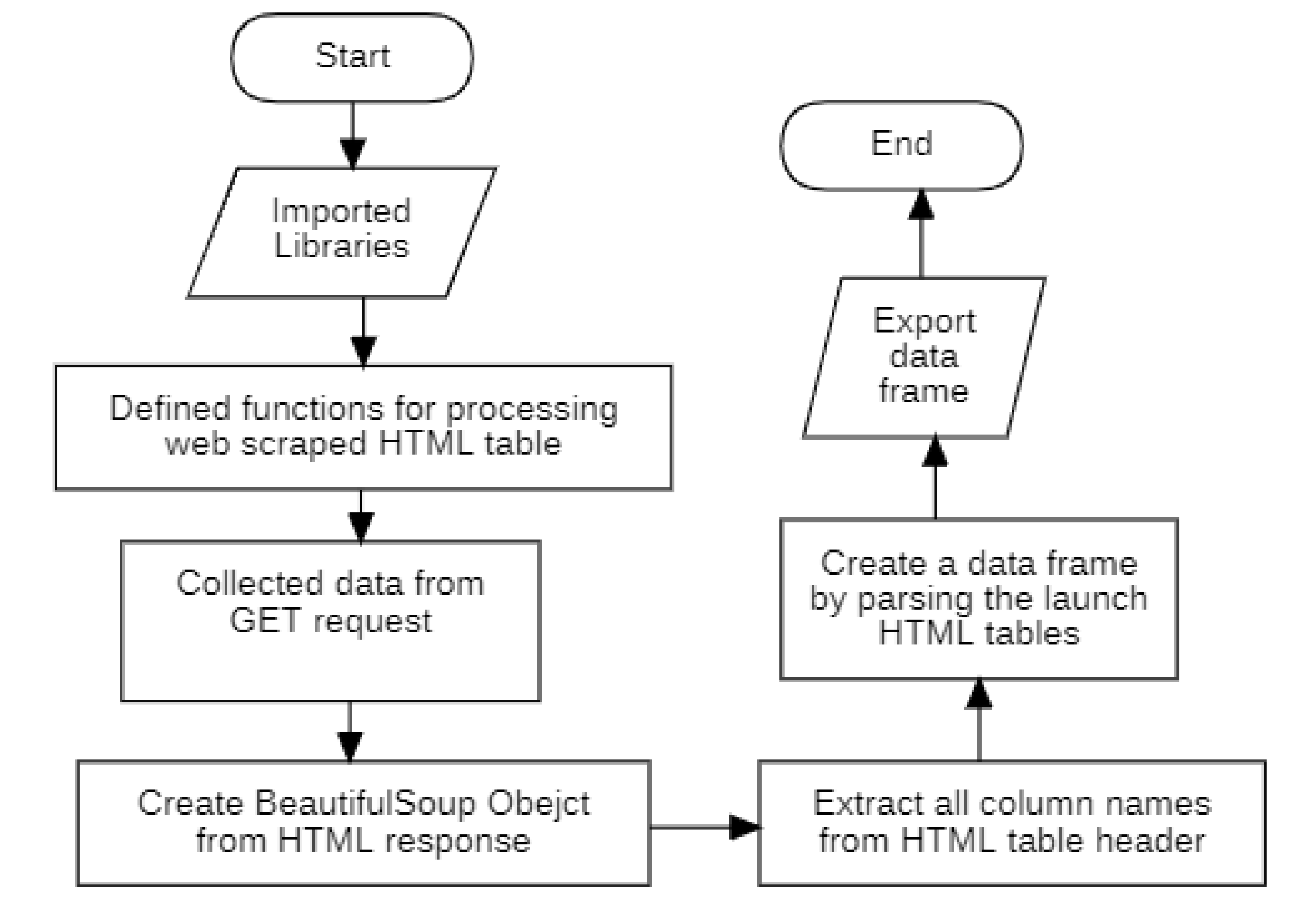
The following REST APIs were used:

- https://api.spacexdata.com/v4/launches/past
- https://api.spacexdata.com/v4/cores/
- https://api.spacexdata.com/v4/payloads/
- https://api.spacexdata.com/v4/launchpads/
- https://api.spacexdata.com/v4/rockets/

The Jupyter notebook used for data collection is:

- https://github.com/NalinMalla/SpaceX-Falcon9-Cost-Predictor/blob/main/1.1.%20Data-collection-api.ipynb

Data Collection - Scraping



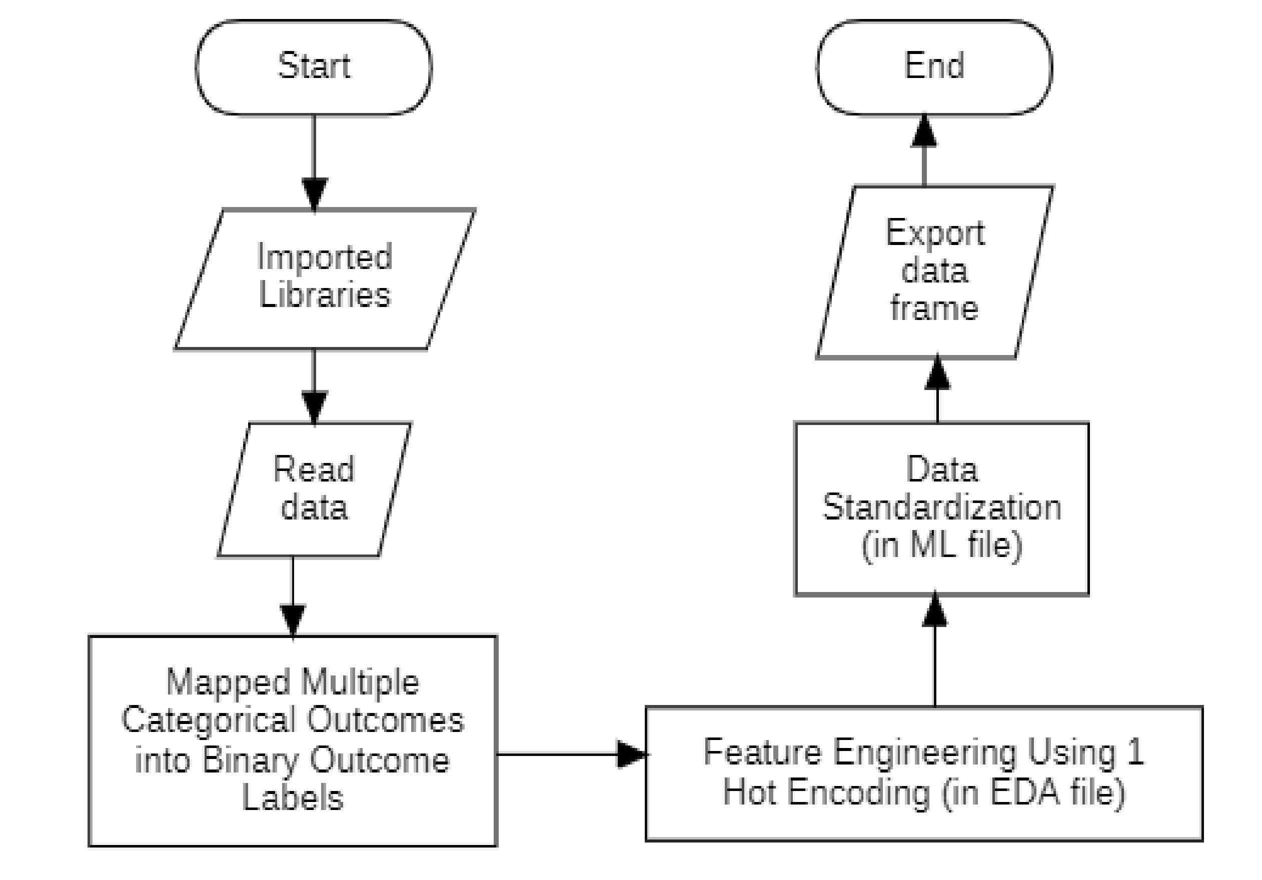
The following webpage was used for web scraping:

- https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

The Jupyter notebook used for Web scraping is:

- https://github.com/NalinMalla/SpaceX-Falcon9-Cost-Predictor/blob/main/1.2.%20Webscraping.ipynb

Data Wrangling



The Jupyter notebooks used for Data Wrangling are:

- https://github.com/NalinMalla/SpaceX-Falcon9-Cost-Predictor/blob/main/1.3.%20Data%20wrangling.ipynb

For Feature Engineering:

- https://github.com/NalinMalla/SpaceX-Falcon9-Cost-Predictor/blob/main/2.2.%20EDA-dataviz.ipynb

For Data Standardization:

- https://github.com/NalinMalla/SpaceX-Falcon9-Cost-Predictor/blob/main/4.%20Machine_Learning_Prediction.ipynb

EDA with Data Visualization

Charts which were Used for EDA with Data Visualization as follows:

1. Scatter & Line Plot: For showing correlation
2. Bar graph: For comparison of magnitude
3. Pie Chart: For representing variable shares of a whole.

The Jupyter notebook used for EDA with Data Visualization is:

- https://github.com/NalinMalla/SpaceX-Falcon9-Cost-Predictor/blob/main/2.2.%20EDA-dataviz.ipynb

EDA with SQL

The following SQL queries were performed:

- Selecting unique values using DISTINCT
- Using aggregate function like SUM, AVG, COUNT to get basic statistics.
- Using LIKE for String comparison
- Subqueries and Grouping
- Limiting Outcomes
- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

The Jupyter notebook used for EDA with SQL is:

- https://github.com/NalinMalla/SpaceX-Falcon9-Cost-Predictor/blob/main/2.1.%20EDA-SQLite.ipynb

Build an Interactive Map with Folium

The following operations were performed:

- Marked all launch sites on a map using:
    - folium.Circle()
    - folium.map.Marker()
- Added colored marks for success/failed launches on each launch site using:
    - folium.plugins.MarkerCluster
- Added polyline between a launch site and coastline near it using:
    - folium.plugins.MousePosition
    - folium.features.DivIcon
    - folium.PolyLine

The Jupyter notebook used for Interactive Map is:

- https://github.com/NalinMalla/SpaceX-Falcon9-Cost-Predictor/blob/main/3.1.%20Interactive-Viz-Geo-Site_location.ipynb

Build a Dashboard with Plotly Dash

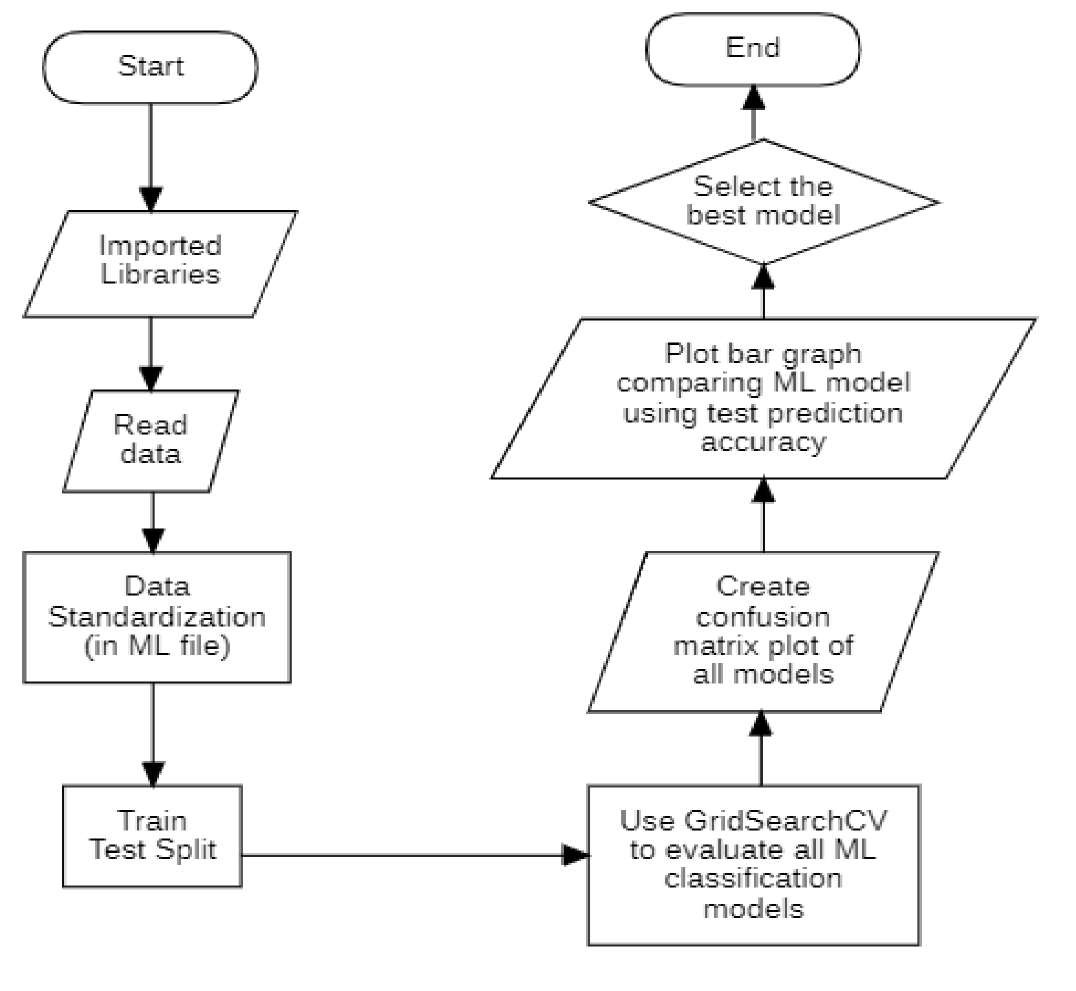
The following operations were performed:

- Created dash app basic layout which included:
  - Title using html.H1()
  - Dropdown Input for launch sites using dcc.Dropdown()
  - Pie Chart to show the successful launch rates of landing sites using dcc.Graph() and px.pie()
  - A range slider input for payload range using dcc.RangeSlider()
  - Scatter Plot to show the correlation between payload and launch success using dcc.Graph() and px.scatter()
- Add a callback function for `site-dropdown` as input, `success-pie-chart` as output.
- Add a callback function for `site-dropdown` and `payload-slider` as inputs, `success-payload-scatter-chart` as output.

The Jupyter notebook used for Dashboard App is:

- https://github.com/NalinMalla/SpaceX-Falcon9-Cost-Predictor/blob/main/3.2.%20Spacex_Dash_App.py

Predictive Analysis (Classification)



The Jupyter notebook used for Predictive Analysis is:

- https://github.com/NalinMalla/SpaceX-Falcon9-Cost-Predictor/blob/main/4.%20Machine_Learning_Prediction.ipynb

Results

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

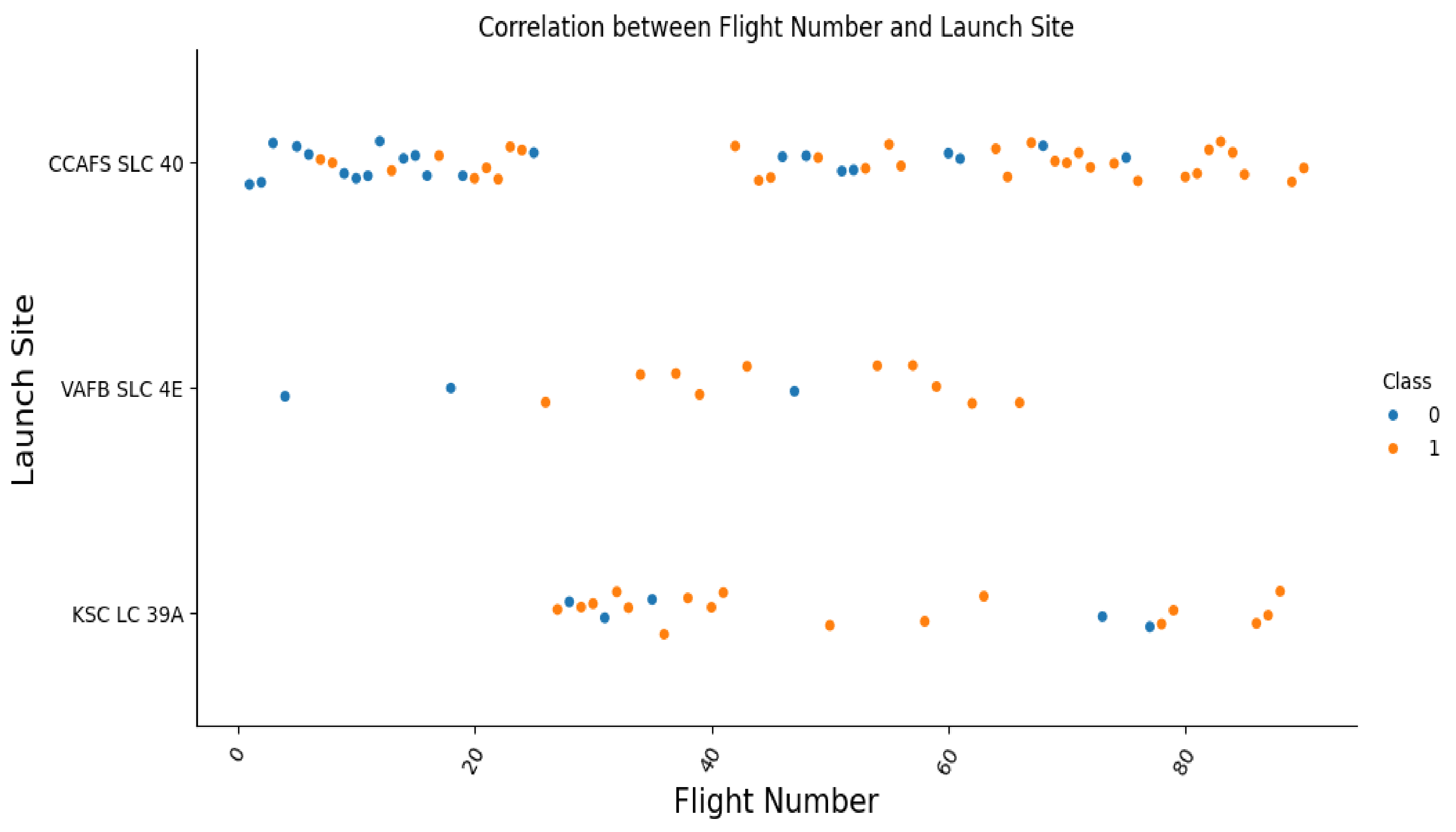
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

Launch Site which are:

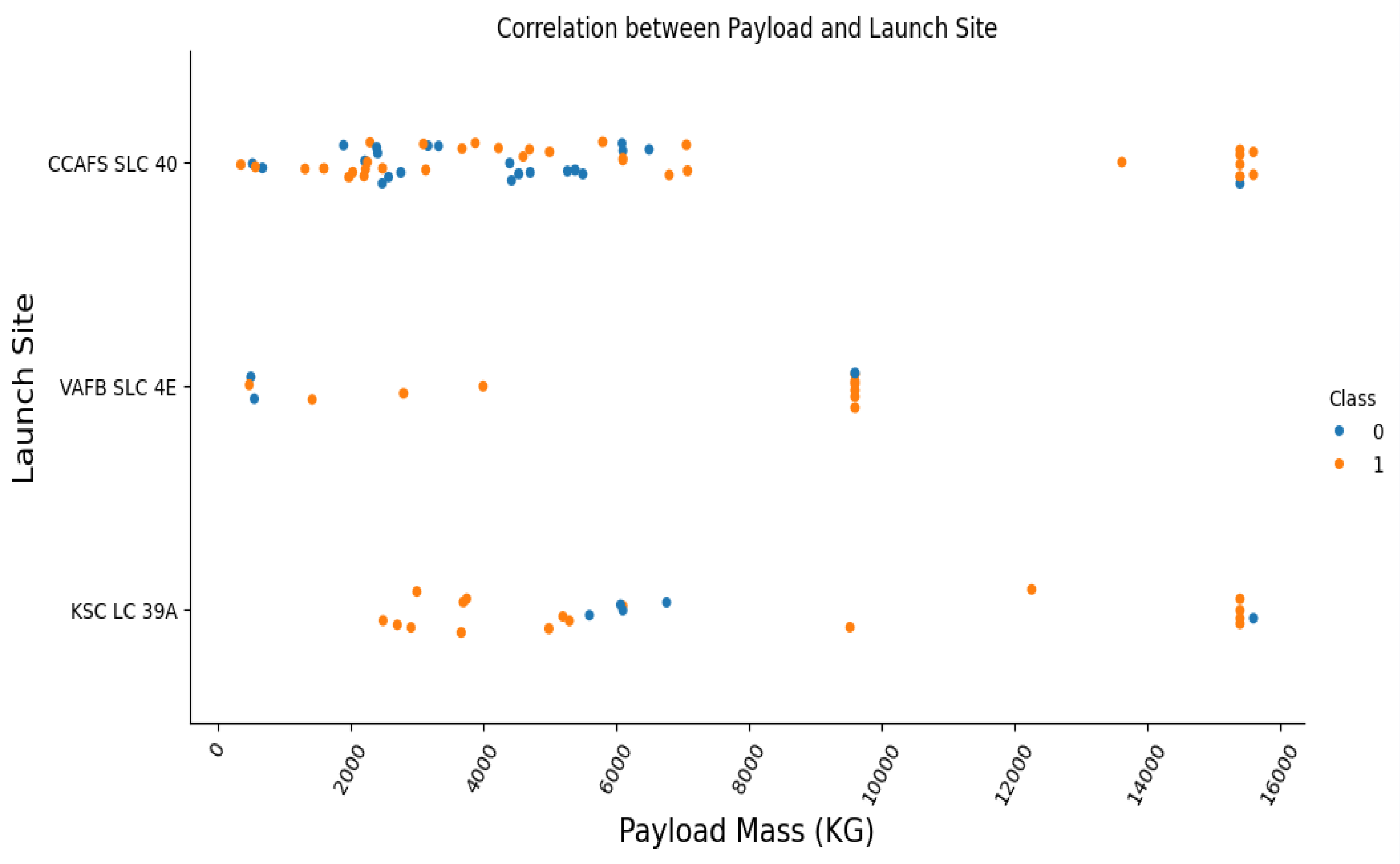
- **Used Most Frequently:** CCAFS SLC 40
- **Used Least Frequently:** VAFB SLC 4E
- **Newly Used:** KSC LC 39A
- **Recently Unused:** VAFB SLC 4E



Payload vs. Launch Site

Launch Site which are used for:

- **Heavy & Light Payloads**
    - CCAFS SLC 40
    - KSC LC 39A
- **Only Light Payloads** (Less than 10000kg):
    - VAFB SLC 4E



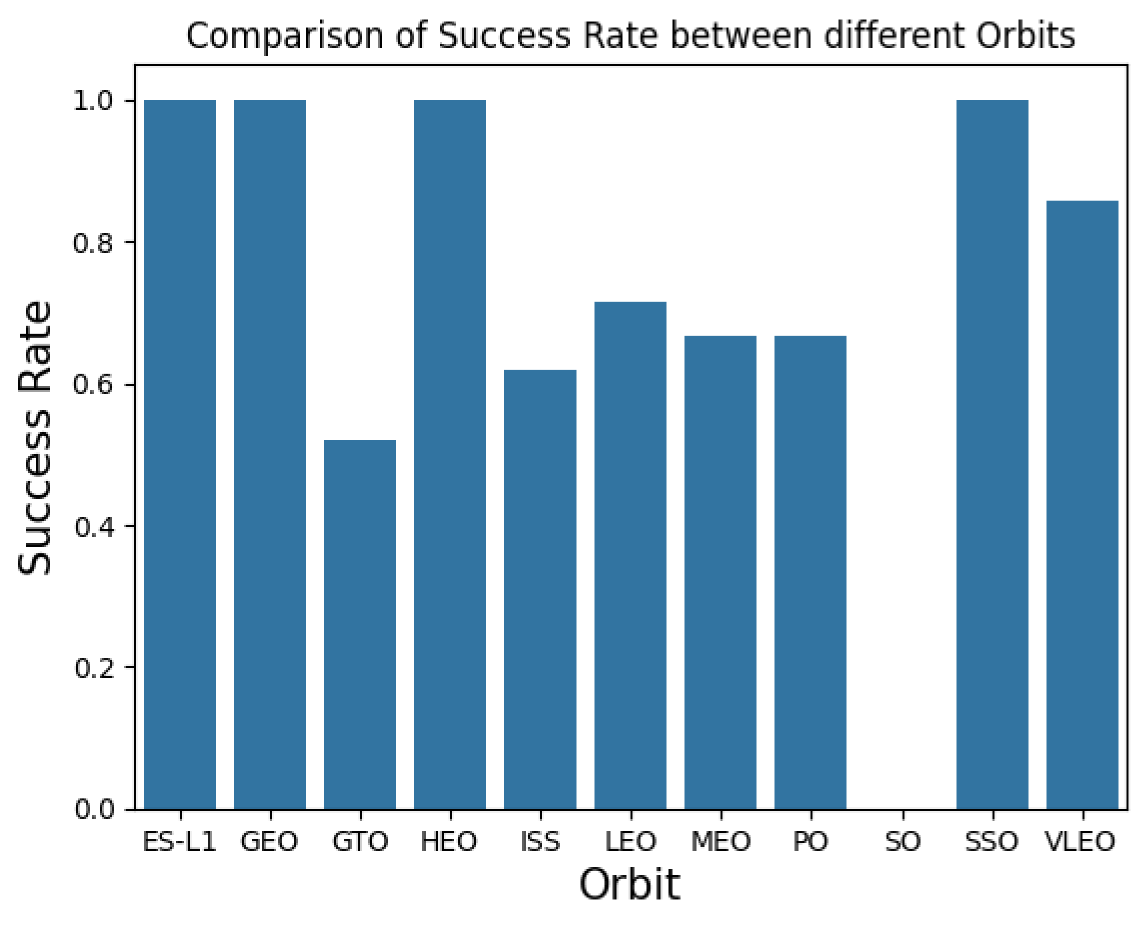
Success Rate vs. Orbit Type

Most Successful Orbit Type: VLEO

- **Success Rate:** 85.7%
- **No. of flights:** 14 (3rd highest)

Unusable Orbit Data (Distorted due to low number of flights):

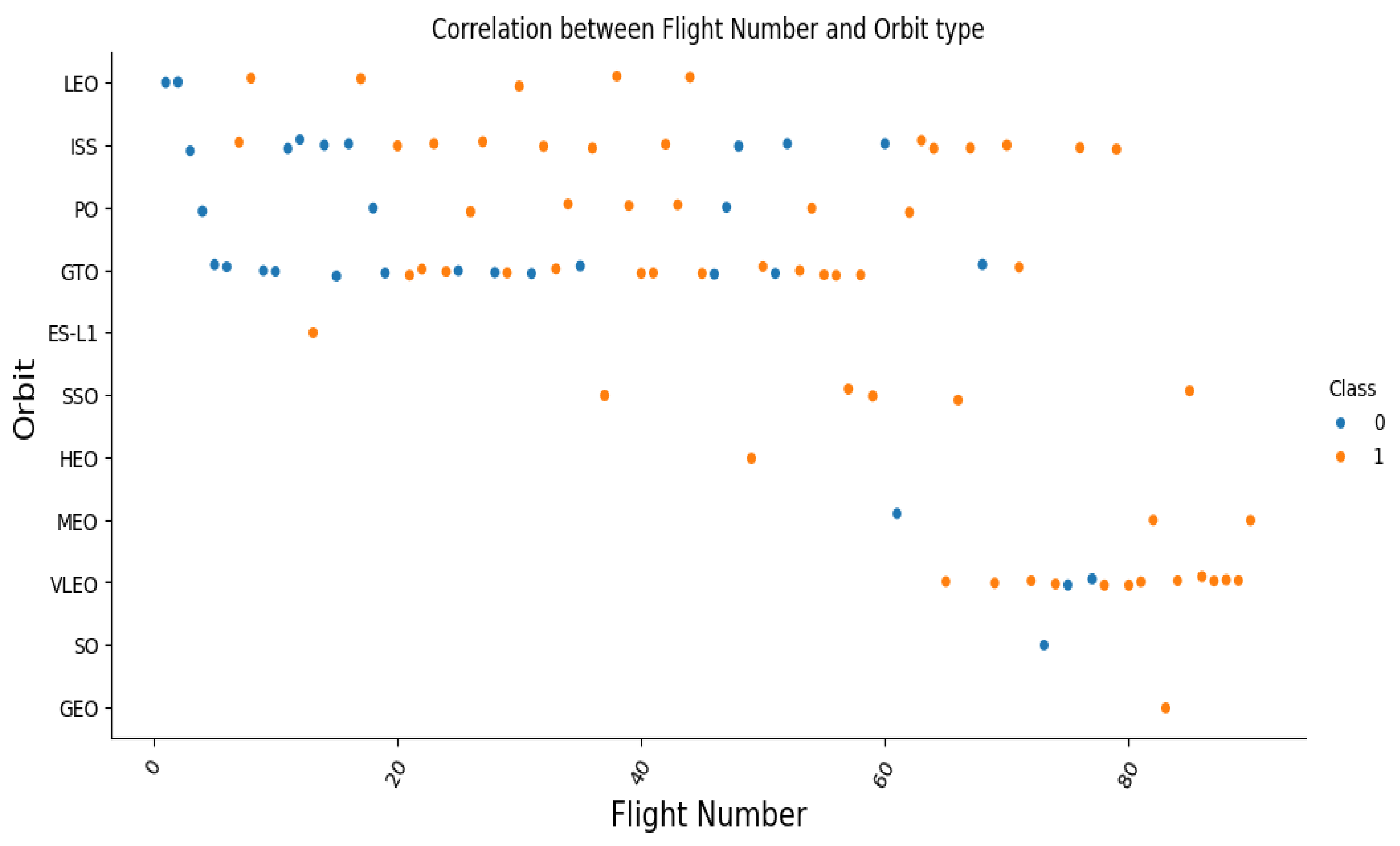
- ES-L1 (1 flight)
- GEO (1 flight)
- HEO (1 flight)
- SO (1 flight)
- SSO (5 flights)



Flight Number vs. Orbit Type

Orbits which are:

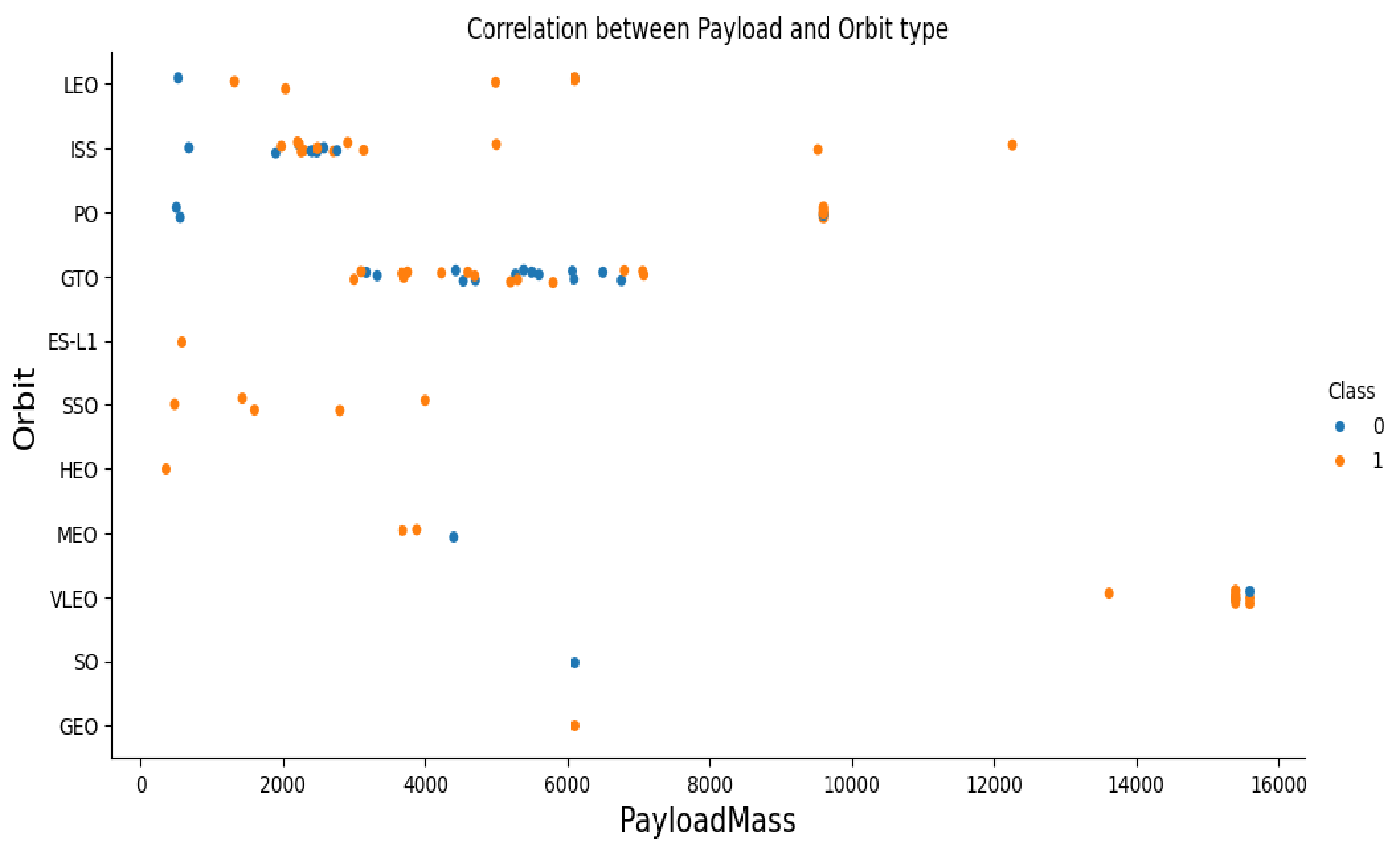
- **Used Most Frequently:**
  - GTO (27 flights)
  - ISS (21 flights)
- **Used Most Frequently in Recent Times:**
  - VLEO (14 flights)
- **Used Only Once:**
  - ES-L1
  - GEO
  - HEO
  - SO



Payload vs. Orbit Type

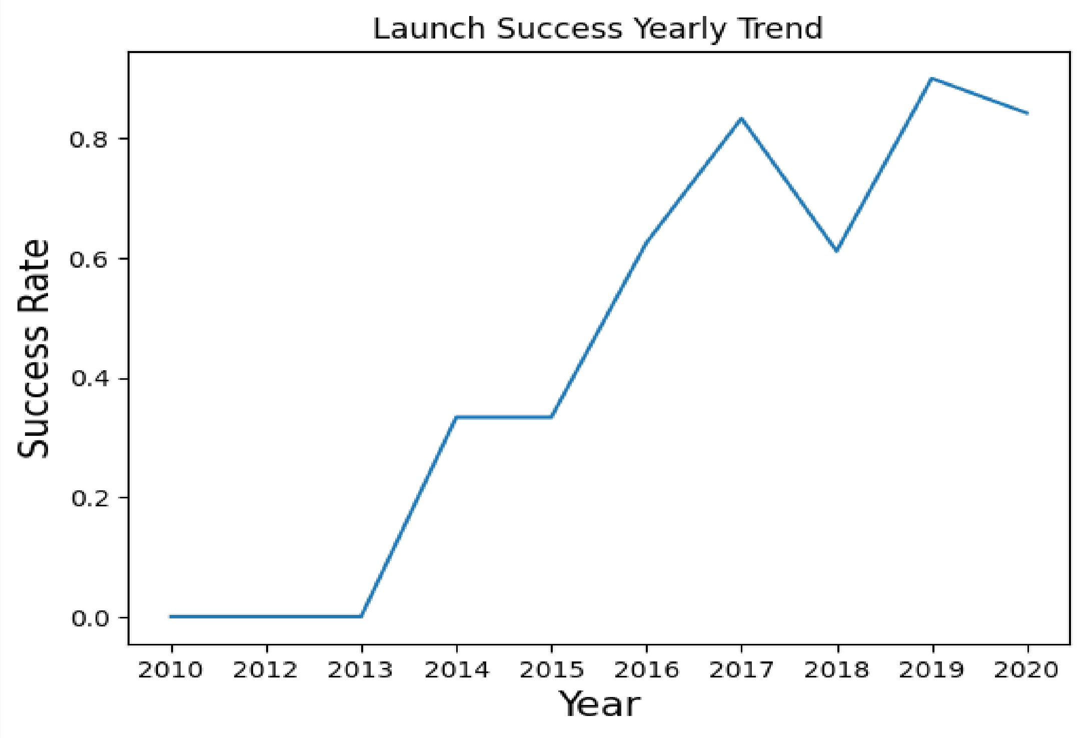
Orbits which are used for:

- **Heavy Payloads** (>10000kg):
  - VLEO
  - ISS
- **Moderate Payloads** (5000kg-10000kg):
  - GTO
  - PO
  - SO
  - GEO
  - LEO
- **Light Payloads** (<5000kg):
  - SSO
  - MEO
  - ES-L1
  - HEO



Launch Success Yearly Trend

- **Year has a highly positive and significant relation with Success Rate.**
  - **Pearson Coefficient:** 0.9338
  - **P-value:** 7.718×10^{-5}
  - But they have a **non-linear relationship.**
- **Most Successful Year:** 2019 (90% success)
- **Most Unsuccessful Years:** 2010-2013




```
pearson_coef, p_value = stats.pearsonr(df_y['Year'], df_y['Class'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
[32]  ✓ 0.0s
... The Pearson Correlation Coefficient is 0.9338768457532939  with a P-value of P = 7.718061284872038e-05
```

All Launch Site Names

We can get the names of all launch site by:

- Querying only the "Launch_Site" column from the database table using SELECT statement.
- Getting unique values of launch site names using DISTINCT keyword to remove duplicate entries from all records.
- In the output both CCAFS SLC-40 and CCAFS LC-40 are specified but they are the same launch site so data need to be cleaned to fix this.

```
%%sql
SELECT DISTINCT Launch_Site FROM spacextable;
```

[10] Python

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

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

Launch Site Names Begin with 'CCA'

```
%%sql
SELECT * FROM spacextable WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

[11] Python

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

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

Query:

```
%%sql
SELECT * FROM spacextable WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

Explanation:

We can get 5 records of site names beginning with “CCA” by:

- Querying entire rows using wildcard * from the database table using SELECT statement.
- Using LIKE keyword to compare string values in “Launch_Site” column starting with “CCA”.
- Filtering the results to have specified LIKE property using WHERE keyword.
- Limiting the output records to 5 using LIMIT keyword.

Total Payload Mass

We can get the total payload mass of boosters launched by “NASA (CRS)” by:

- Querying sum of only the “payload_mass__kg_” column using SUM aggregate function along side the SELECT statement.
- Using LIKE keyword to compare string values in “customer” column having “NASA (CRS)” in it.
- Filtering the results to have specified LIKE property using WHERE keyword.

```
%%sql
SELECT SUM(payload_mass__kg_) AS 'Total Payload Mass'
FROM spacextable
WHERE customer LIKE '%NASA (CRS)%';
```

Python

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

Total Payload Mass
48213

Average Payload Mass by F9 v1.1

We can find the information mentioned in the title by:

- Querying sum of only the “payload_mass__kg_” column using AVG aggregate function along side the SELECT statement.
- Using LIKE keyword to compare string values in “booster_version” column having “F9 v1.1” in it.
- Filtering the results to have specified LIKE property using WHERE keyword.

```sql
%%sql
SELECT AVG(payload_mass__kg_) AS 'Average Payload Mass' FROM spacextable
  WHERE booster_version LIKE '%F9 v1.1%';
```

Python

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

Average Payload Mass
2534.6666666666665

First Successful Ground Landing Date

We can find the information mentioned in the title by:

- Querying only the “date” column using SELECT statement.
- Using LIKE keyword to compare string values in “landing_outcome” column to be “Success (ground pad)”.
- Filtering the results to have specified LIKE property using WHERE keyword.
- Limiting output to 1 using LIMIT.

```
%%sql
SELECT date FROM spacextable
  WHERE landing_outcome LIKE 'Success (ground pad)'
  LIMIT 1;
```

✓ 0.0s Python

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

Date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

We can find the information mentioned in the title by:

- Querying only the “booster_version” column using SELECT statement.
- Using LIKE keyword to compare string values in “landing_outcome” column to be “Success (ground pad)” and “payload_mass__kg_” column values to be between 4000 & 6000.
- Filtering the results to have specified property using WHERE keyword.

```
%%sql
SELECT booster_version FROM spacextable
  WHERE landing_outcome LIKE 'Success (drone ship)'
    AND (payload_mass__kg_ > 4000 AND
    payload_mass__kg_ < 6000);
```

✓ 0.0s Python

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

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

Total Number of Successful and Failure Mission Outcomes

We can find the information mentioned in the title by:

- Querying count of only the "mission_outcome" column using COUNT aggregate.
- Using LIKE keyword to compare string values in "mission_outcome" column to have "Success" & "Failure" in it.
- Filtering the results to have specified LIKE property using WHERE keyword.

```
%%sql
SELECT COUNT(mission_outcome) AS 'No. of Successful Mission' FROM spacextable
    WHERE mission_outcome LIKE '%Success%';
```

0.0s

* sqlite:///my_data1.db
Done.

No. of Successful Mission
100

```
%%sql
SELECT COUNT(mission_outcome) AS 'No. of Unsuccessful Mission' FROM spacextabl
    WHERE mission_outcome LIKE '%Failure%';
```

0.0s

* sqlite:///my_data1.db
Done.

No. of Unsuccessful Mission
1

Boosters Carried Maximum Payload

```
%%sql
SELECT DISTINCT booster_version FROM spacextable
  WHERE payload_mass__kg_ == (SELECT MAX
  (payload_mass__kg_) FROM spacextable);
```

0.0s Python

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

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

We can find the information mentioned in the title by:

- Querying only unique values in the “booster_version” column using DISTINCT keyword.
- Filtering the results using WHERE keyword to have maximum “payload_mass__kg_” using MAX in subquery.

2015 Launch Records

```
%%sql
SELECT SUBSTR(date,6,2) AS 'Month',
booster_version, launch_site, landing_outcome
FROM spacextable
  WHERE landing_outcome LIKE 'Failure (drone ship)
  ' AND SUBSTR(date,0,5) LIKE '2015';
```

✓ 0.0s Python

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

Month	Booster_Version	Launch_Site	Landing_Outcome
01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

Query:

```
%%sql
SELECT SUBSTR(date,6,2) AS 'Month', booster_version, launch_site, landing_outcome FROM spacextable WHERE landing_outcome LIKE 'Failure (drone ship)' AND SUBSTR(date,0,5) LIKE '2015';
```

Explanation:

We can list the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015 by:

- Querying only month part of date using SUBSTR along side other mentioned columns using SELECT.
- Using LIKE keyword to compare string values in “landing_outcome” column be “Failure (drone ship)” and substring of date containing year to be “2015”.
- Filtering the results to have specified property using WHERE keyword.

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

```
%%sql
SELECT landing_outcome, COUNT(landing_outcome)
FROM spacextable GROUP BY landing_outcome
  HAVING date BETWEEN '2010-06-04' AND
  '2017-03-20'
  ORDER BY COUNT(landing_outcome) DESC
```

✓ 0.0s Python

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

Landing_Outcome	COUNT(landing_outcome)
No attempt	21
Success (drone ship)	14
Success (ground pad)	9
Failure (drone ship)	5
Controlled (ocean)	5
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Query:

```
%%sql
SELECT landing_outcome, COUNT(landing_outcome) FROM spacextable GROUP BY landing_outcome
HAVING date BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY COUNT(landing_outcome) DESC
```

Explanation:

We can 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 by:

- Querying only landing_outcome column and its count using COUNT aggregate function.
- Grouping the results according to landing_outcome.
- Filtering the results using WHERE to have date in between specified date using BETWEEN.
- Presenting the filtered result in descending order on the basis of COUNT(landing_outcome) using ORDER BY keyword.

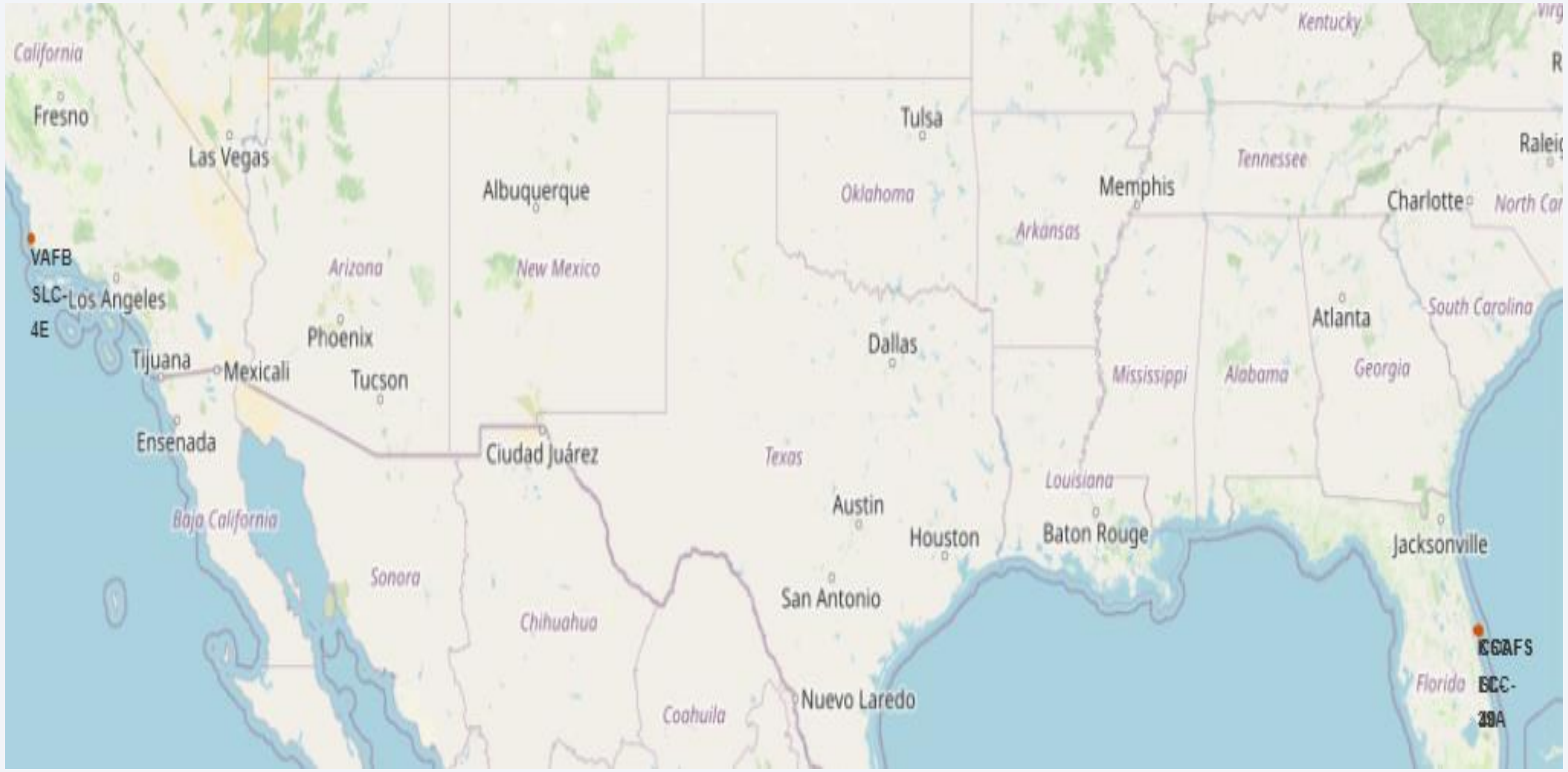
Section 3

Launch Sites Proximities Analysis

Map showing all Landing Sites

In the given figure sites are highlighted in orange and we can see that:

- They are all **located near costal regions of the U.S.** which allows rockets to land on water and reduces damage in case of failure.
- They are all **located near the equator line** because the linear velocity of Earth's surface is greatest towards the equator which is advantageous for launching rockets.



Map marking launch success & failures in each site

In the given figure we can see that:

- CCAFS SLC-40 launch site has more failures that success.
- Thus, the colored markers in marker cluster makes identifying launch sites success rate easier.

Map of a launch site in proximity to coastline

In the given figure we can see that:

- A blue line highlighting a path from CCAFS SLC-40 launch site to nearest coastline.
- The distance between the launch site and nearest coastline. i.e. 1.23km highlighted in orange.

Section 4

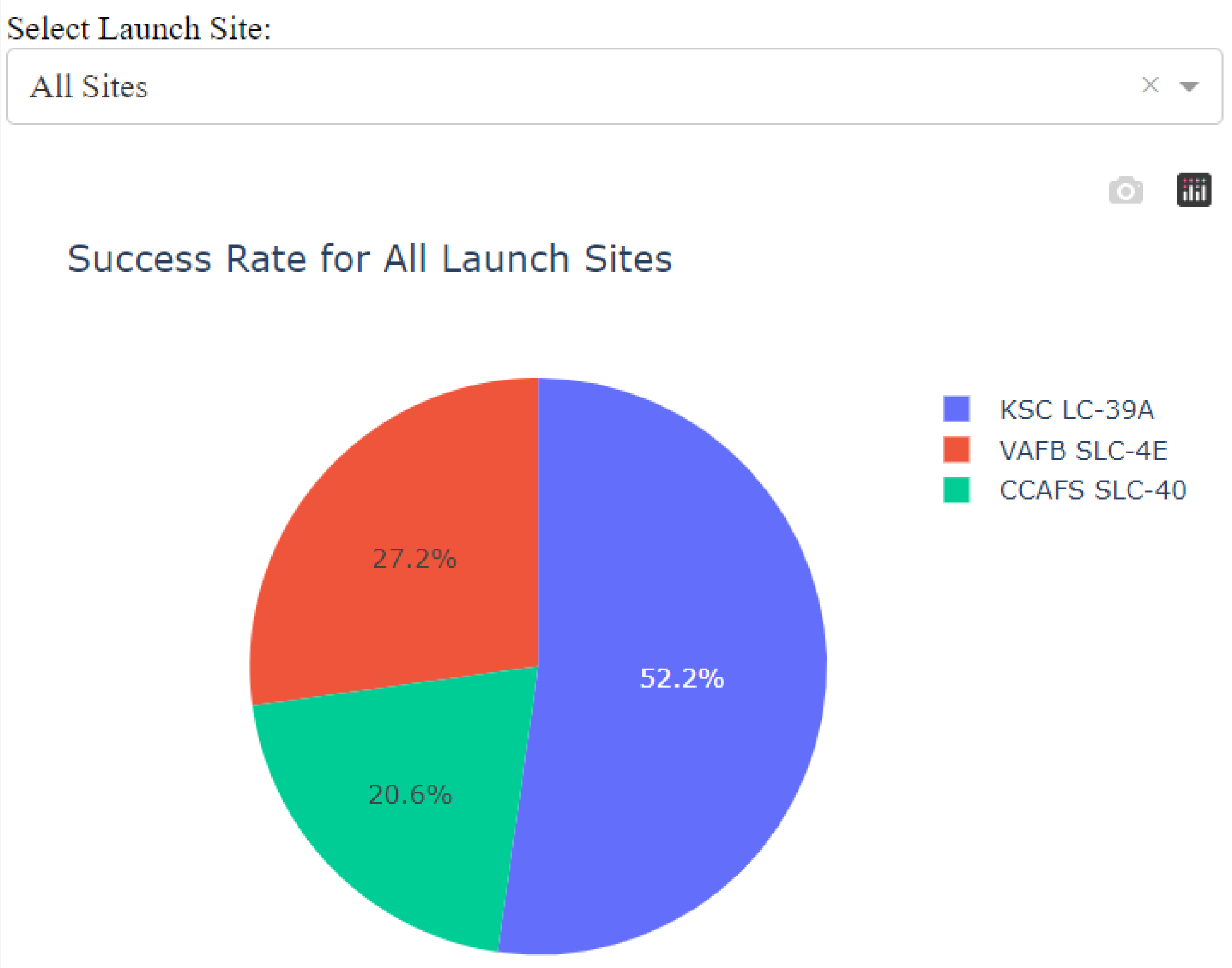
Build a Dashboard with Plotly Dash

Success count for all Launch Sites

The given pie chart shows the contribution of each launch site toward total successful launches:

- KSC LC-39A is the most successful launch site contributing to 52.2% of all successful missions.
- CCAFS SLC-40 is the least successful launch site.
- 72.6% of successful missions were launched from Florida as KSC LC-39A and CCAFS SLC-40 both are within it.

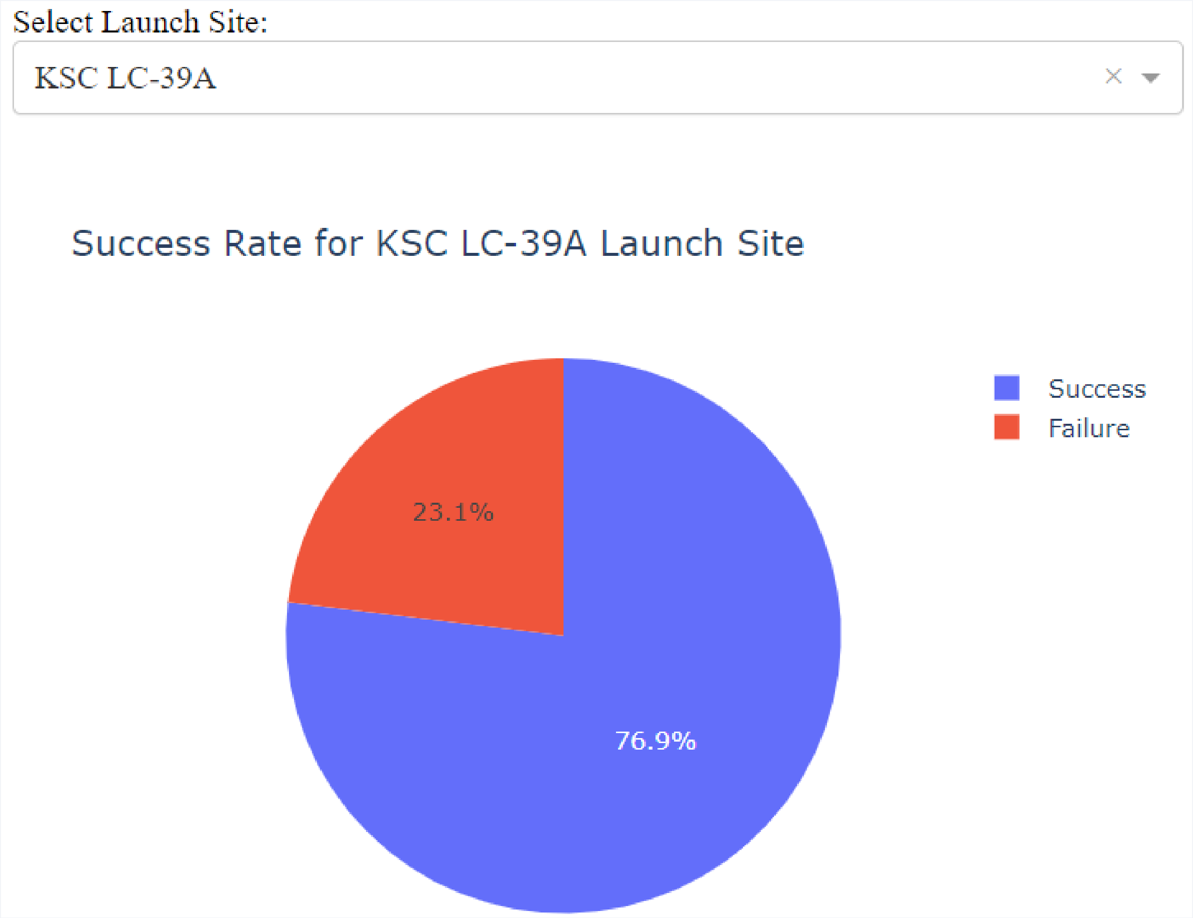
Note:
Data was cleaned to replace CCAFS LC-40 with CCAFS SLC-40 because they both are the same launch site.



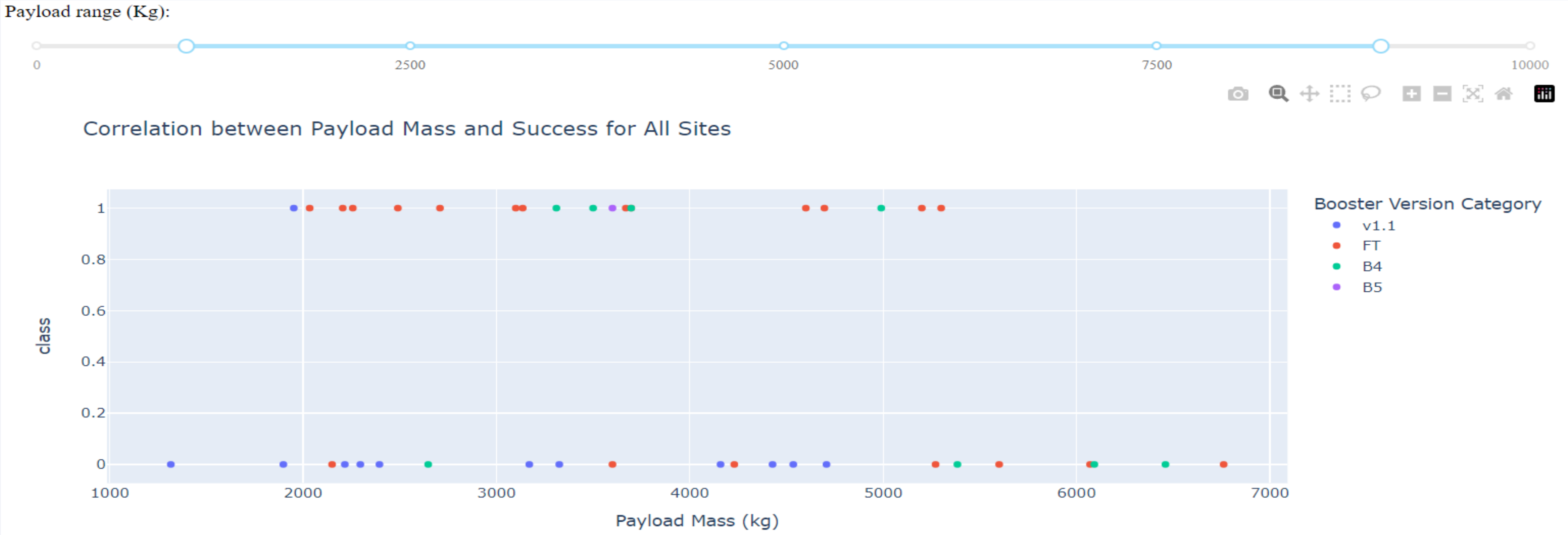
Launch Site with Highest Launch Success Ratio

The given pie chart shows the success rate of KSC LC-39A launch site as:

- KSC LC-39A has the highest launch success rate of 76.9%.
- Consequently, it’s failure rate is 23.1%.



Correlation between Payload Mass & Success for all sites



- **Most Successful Booster Version:** FT
- **Least Successful Booster Version:** v1.1
- **Most Successful Payload Mass Range:** 3100kg – 3700kg (7 success)
- **Least Successful Payload Mass Range:** 4100kg – 4800kg (5 failure)

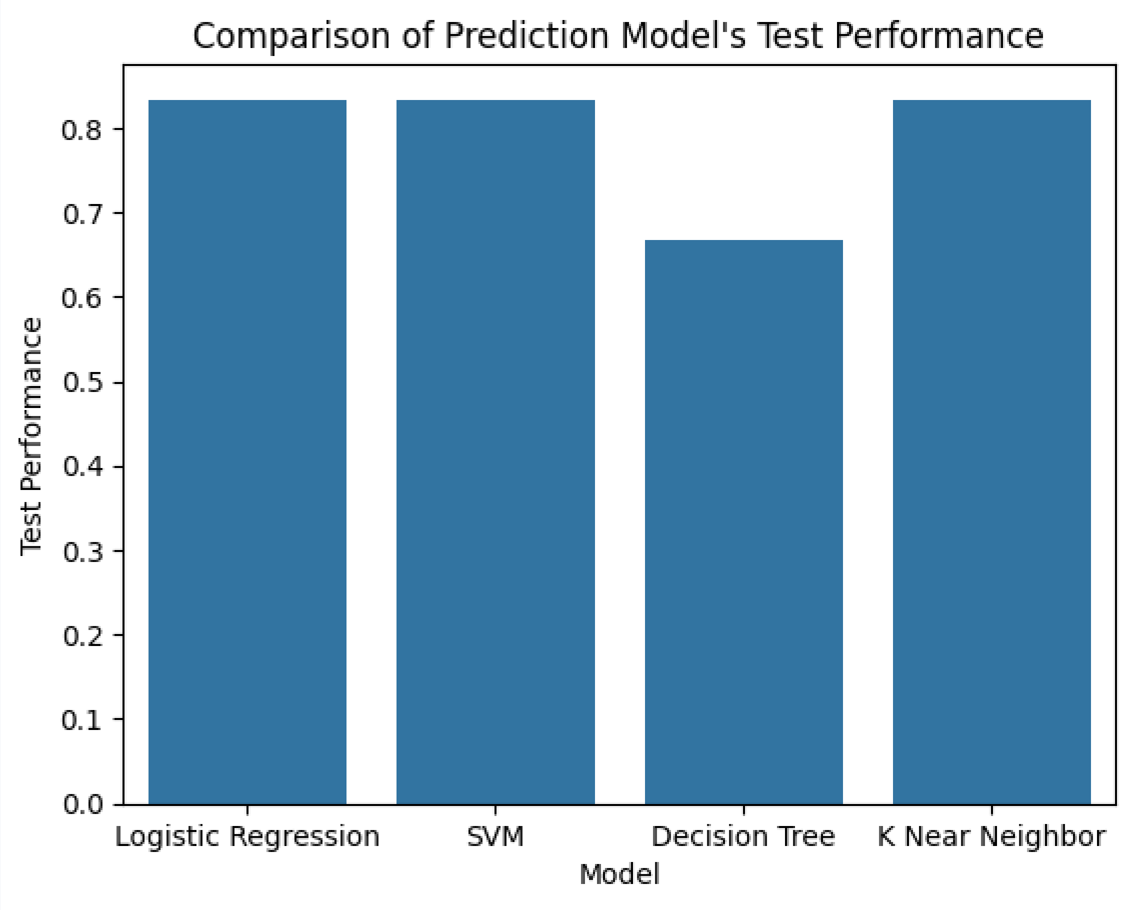
Section 5

Predictive Analysis (Classification)

Classification Accuracy

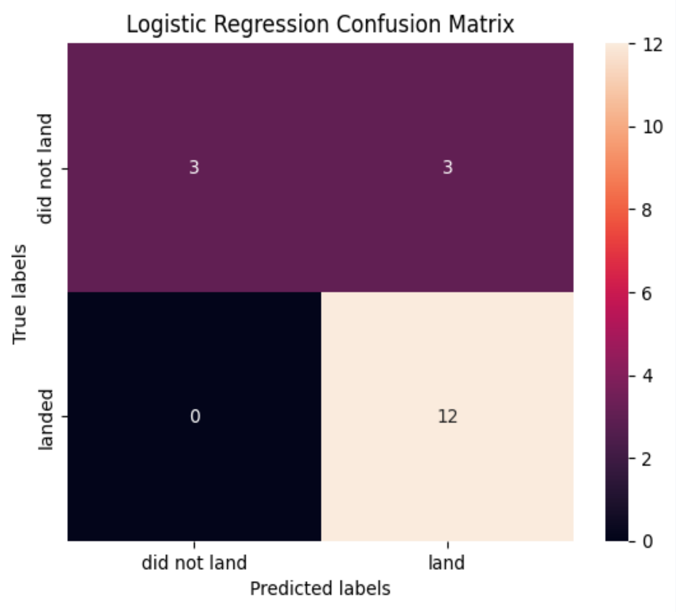
The given bar chart shows the prediction accuracy of various Machine Learning models using new testing data set.

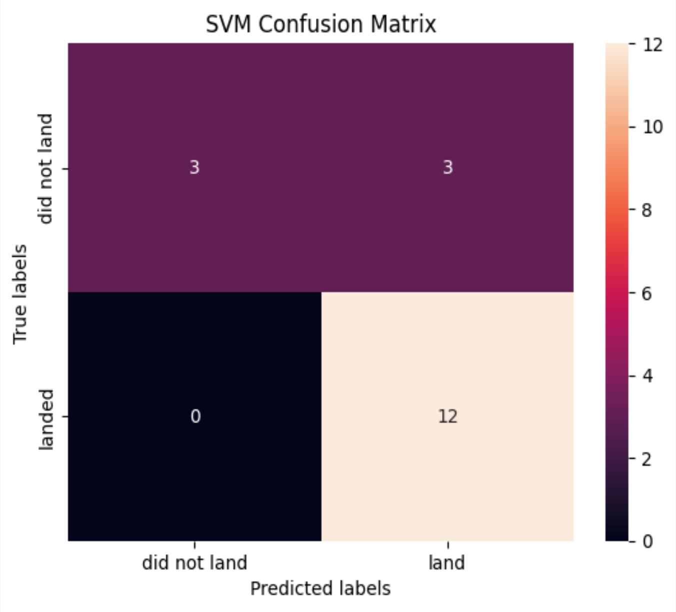
- The Decision Tree classifier scored lowest with accuracy of 66.6% due to its inability to fit all training data.
- All other prediction models performed the same with testing accuracy of 83.3%.

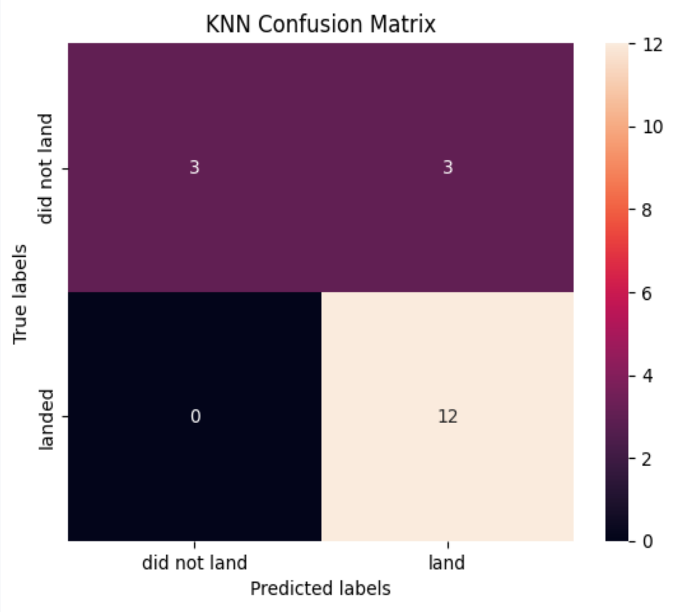


Confusion Matrix

Even the confusion matrices are same for all three best models.







Conclusions

- Year has a highly positive and significant but non-linear relation with Success Rate.
- The increase in success rate which seemed due to increase in flight number and the added experiences that come with it, also seem to be heavily influenced by the use of VLEO orbit and its high success rate.
- The positive relation between success rate and weight might be due to VLEO orbit due to its high no. of flights with high payload weight.
- All landing sites are located near the equator line because the linear velocity of Earth's surface is greatest towards the equator which is advantageous for launching rockets.
- 72.6% of successful missions were launched from Florida as KSC LC-39A and CCAFS SLC-40 both are within it.
- KSC LC-39A has the highest launch success rate of 76.9% and contributes to 52.2% of all successful missions.
- "FT" is the most successful booster version where as "v1.1." is the least successful.
- 3100kg to 3700kg is the most successful payload mass where as 4100kg to 4800kg is the least successful.


Thank you!