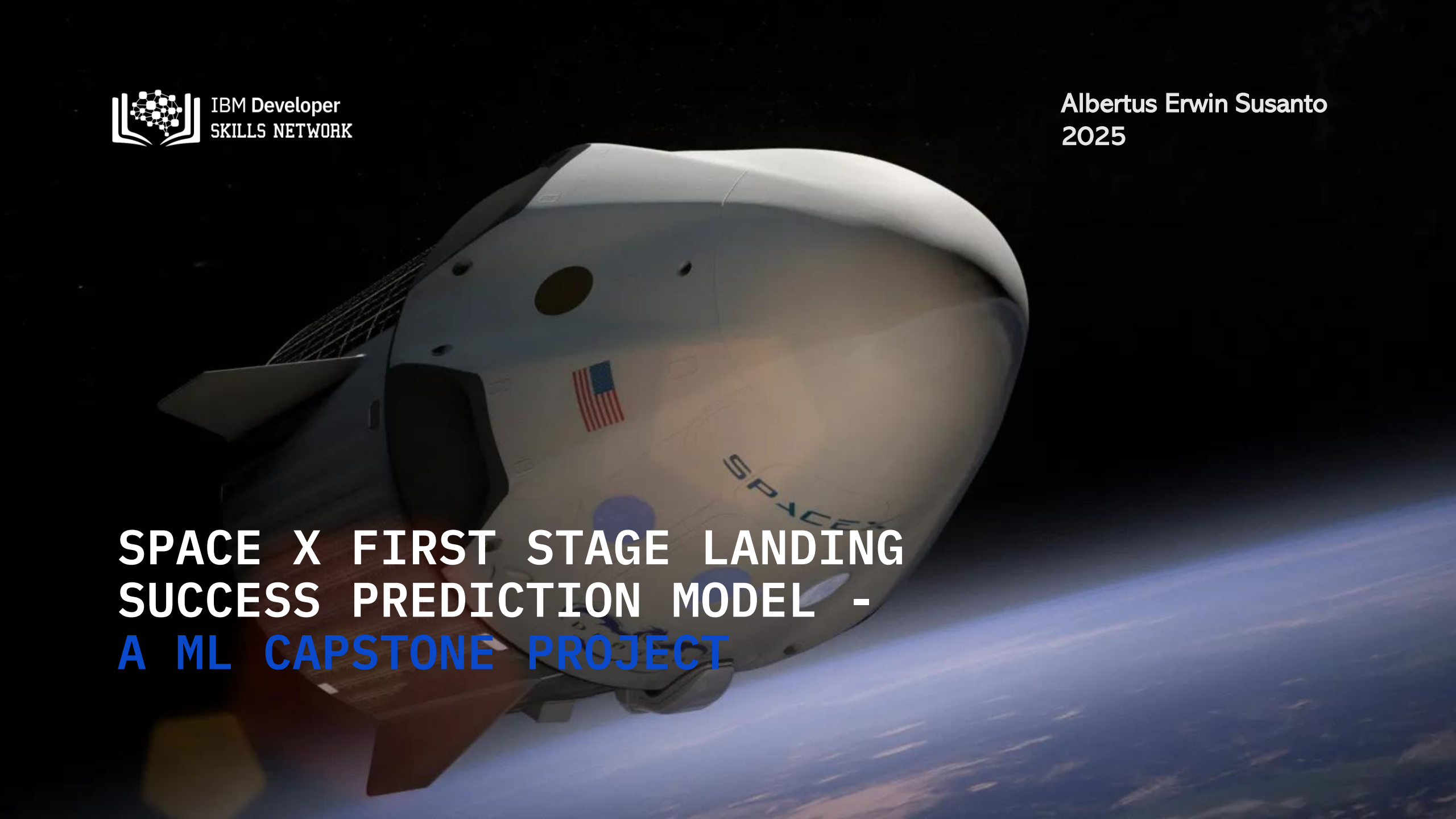


Albertus Erwin Susanto
2025

SPACE X FIRST STAGE LANDING SUCCESS PREDICTION MODEL - A ML CAPSTONE PROJECT

Outline

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

Executive Summary

Summary of Methodologies

The research attempts to identify the factors for a successful rocket landing. Our methodologies include:

- Collect data using SpaceX REST API and web scraping techniques
- Wrangle data to create success/fail outcome variable
- Explore data with data visualization techniques, considering the following factors: payload, launch site, flight number and yearly trend
- Analyze the data with SQL, calculating the following statistics: total payload, payload range for successful launches, and total of successful and failed outcomes
- Explore launch site success rates and proximity to geographical markers
- Visualize the launch sites with the most success and successful payload ranges
- Build Models to predict landing outcomes using logistic regression, support vector machine (SVM), decision tree and K-nearest neighbor (KNN)

Summary of Results

- EDA:
  - Launch success has improved over time
  - KSC LC-39A has the highest success rate among landing sites
  - Orbits ES-L1, GEO, HEO, and SSO have a 100% success rate
- Geographical Analysis (Folium):
  - Most launch sites are near the equator, and all are close to the coast.
- Predictive Analysis:
  - All models performed similarly on the test set, i.e. accuracy of 83%.

Introduction

- The advancements in space technology have significantly transformed the commercial space industry, with companies like SpaceX leading the way in cost-effective space exploration. One of SpaceX's key innovations is the reusability of the Falcon 9 first-stage booster, which dramatically reduces the cost of space missions. A typical Falcon 9 launch costs $62 million, whereas other space launch providers charge over $165 million per launch. **The ability to successfully land and reuse the first-stage booster is a crucial factor in these cost savings.**
- This capstone project focuses on predicting the successful landing of the Falcon 9 first stage. **By developing a predictive model, we aim to assess the probability of a successful landing and explore its impact on launch costs.** This information is particularly valuable for competing aerospace companies seeking to evaluate the feasibility of bidding against SpaceX for commercial satellite launches.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology: through SpaceX API and web scrapping Wikipedia.
- Perform data wrangling
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models: Logistic Regression, SVM, KNN, and Decision Tree.

Data Collection – by SpaceX API

- To build a predictive model for Falcon 9 first-stage landings, we needed historical launch data from SpaceX. **This data was collected using the SpaceX API**, which provides structured information on past launches, including launch details, payload specifications, booster versions, and landing outcomes. The following steps summarize the data retrieval process:

  1. **API Retrieval:** Extracted past launch records from https://api.spacexdata.com/v4/launches/past, including rocket type, launch site, payload mass, orbit, and landing success.
  2. **Feature Extraction:** Queried additional API endpoints to map rocket IDs to booster versions, retrieve launchpad locations, and fetch payload and core details.
  3. **Data Cleaning & Transformation:** Processed missing values, converted categorical variables, and structured the dataset for analysis.
  4. **Storage:** The final dataset was saved as a CSV file for machine learning modeling.

Data Collection – by SpaceX API

- After wrangling the data we get from the API, we collect the features, including **FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude.**

- See the result at: https://github.com/aerwins-yyw/space_x-first_stage_landing/blob/332b637cfa0987aa7842a93306204b0fe8fbbfa8/1%20jupyter-labs-spacex-data-collection-api-v2.ipynb

Data Collection – by Web Scrapping

- We also try to collect the data through an alternative way as part of learning in this project, that is through web scrapping.
- **We perform web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled `List of Falcon 9 and Falcon Heavy launches`.**
- The link: https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches
- For the sake of uniformity, the link we used is the static version: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922
- See the result at: https://github.com/aerwins-yyw/space_x-first_stage_landing/blob/332b637cfa0987aa7842a93306204b0fe8fbbfa8/2%20jupyter-labs-webscraping.ipynb

Data Wrangling

- During the data wrangling phase, I cleaned and prepared the dataset for further analysis. Here's what I did step by step:

1. **Checked for Missing Values and Data Types:** To ensure data integrity, I checked for missing values. I also examined the data types to identify any necessary conversions.
2. **Explored Key Features:** To understand the dataset better, I used to analyze (1) Launch sites to determine which locations had the most launches and (2) Orbit types to identify common orbital destinations.
3. **Processed Landing Outcomes:** Since the dataset contained different landing outcomes, I examined the Outcome column using df["Outcome"].value_counts(). I then categorized the outcomes into two groups (1) Successful landings (reusable boosters) and (2) Unsuccessful landings (e.g., ocean landings, crashes). To label these outcomes, I created a binary classification variable (Class): 1 for successful landings, 0 for failed landings.
4. **Calculated Landing Success Rate:** To measure overall landing performance, I calculated the success rate using df["Class"].mean() and printed the result.
5. **Saved the Processed Dataset:** Finally, I saved the cleaned dataset as dataset_part_2.csv, ensuring it was structured and ready for analysis.

- See the result at: https://github.com/aerwins-yyw/space_x-first_stage_landing/blob/332b637cfa0987aa7842a93306204b0fe8fbbfa8/3%20labs-jupyter-spacex-Data%20wrangling-v2.ipynb

EDA with SQL

- To get the preliminary insights from our data, we make an exploratory data analysis using SQL queries. Here are what I did:

1. **Creating a Cleaned Table (SPACEXTABLE):** I started by creating a new table called SPACEXTABLE based on SPACEXTBL, but I made sure to exclude rows where the Date column was null. This helped me clean the dataset and ensure that all records in my new table contained valid dates.
2. **Exploring the Data:** After creating SPACEXTABLE, I ran a simple SELECT * query to view all the records. This allowed me to verify that the data was correctly transferred and filtered.
3. **Finding Unique Launch Sites:** I wanted to see all the different launch sites used in the dataset, so I used SELECT DISTINCT LAUNCH_SITE to retrieve a list of unique launch sites. This gave me a clearer idea of the different locations from which SpaceX launches its rockets.

EDA with SQL

4. **Filtering Data for a Specific Launch Site (CCAFS LC-40):** To focus on launches from the CCAFS LC-40 site, I wrote a query to filter records where LAUNCH_SITE matched 'CCAFS LC-40' and limited the results to 5 rows. This helped me examine the data specific to that launch site without pulling an overwhelming number of records.
5. **Calculating Total Payload Mass for NASA CRS Missions:** Lastly, I wanted to analyze the total payload mass for NASA's Commercial Resupply Services (CRS) missions. I used SUM(PAYLOAD_MASS__KG_) while filtering for records where the CUSTOMER column contained 'NASA (CRS)'. This helped me understand the total weight of payloads that SpaceX launched for NASA's CRS program.

- To see the complete code and the result, see here: https://github.com/aerwins-yyw/space_x-first_stage_landing/blob/332b637cfa0987aa7842a93306204b0fe8fbbfa8/5%20jupyter-labs-eda-dataviz-v2.ipynb

EDA with Panda and Matplotlib

- Using the data collected, I continue the exploratory data analysis by making plots to see the relationship between features. The plots I made include:
  1. **Scatter plot on the relationship between Flight Number and Launch Site**
  2. **Scatter plot on the relationship between launch sites and their payload mass**
  3. **Horizontal bar chart on the relationship between success rate of each orbit type**
  4. **Scatter plot on the relationship between Flight Number and Orbit type**
  5. **Scatter plot on the relationship between Payload and Orbit type**
  6. **Line chart to see the average launch success trend**

- The features we have include: 'FlightNumber', 'PayloadMass', 'Orbit', 'LaunchSite', 'Flights', 'GridFins', 'Reused', 'Legs', 'LandingPad', 'Block', 'ReusedCount', 'Serial'
- After observing those relationship, we apply OneHotEncoder to the column Orbits, LaunchSite, LandingPad, and Serial.
- To see more details, please check on: https://github.com/aerwins-yyw/space_x-first_stage_landing/blob/2ea031da325b1cc0d1fd776762897ff5cc764e66/5%20jupyter-labs-eda-dataviz-v2.ipynb

Build an Interactive Map with Folium

- Here, I conduct an analysis on the geographical location of the launch sites. The steps include:

    1. **Downloading and Loading the Dataset:** I downloaded the spacex_launch_geo.csv file and loaded it into a pandas DataFrame. This dataset contains information about SpaceX launch sites, including their latitude, longitude, and success classification.

    2. **Selecting Relevant Columns and Organizing Data:** I filtered the dataset to keep only the essential columns:
        a. Launch Site: The name of the launch site
        b. Lat (Latitude): The geographic latitude of the launch site
        c. Long (Longitude): The geographic longitude of the launch site
        d. class: The success classification of the launch

Build an Interactive Map with Folium

3. **Grouping:** I then grouped the data by launch site to get unique site locations.

4. **Creating a Base Map:** I initialized a folium map centered around NASA Johnson Space Center, setting an appropriate zoom level to visualize the launch sites.

5. **Next Steps:** Based on this setup, I went on to:
   a. Add markers for different launch sites on the map.
   b. Cluster them using MarkerCluster for better visualization.
   c. Display additional information such as launch success rates or coordinates when hovering over a site.

- For further details, please check: https://github.com/aerwins-yyw/space_x-first_stage_landing/blob/2271789ba5068fbad731edaa2a6b8bd8d05ad48b/6%20lab-jupyter-launch-site-location-v2.ipynb

Build a Dashboard with Plotly Dash

- Then I built a dashboard with Plotly Dash. I decided to run it locally instead of from the Skills Learn platform provided by IBM to have more control on the dashboard. The steps include:

1. **I began by importing key libraries for building the dashboard:**
    a. dash, dash_html_components, and dash_core_components to create the web-based dashboard.
    b. plotly.express for interactive visualizations.
    c. pandas for data handling.
    d. wget to download the dataset.

2. **Downloading and Loading SpaceX Launch Data:** I downloaded the spacex_launch_dash.csv file and loaded it into a pandas DataFrame. This dataset contains information about SpaceX launches, including: Launch site, Payload mass, Class (successful or failed launches), Booster versions.

3. **Examining the First Few Rows:** I checked the first 10 rows of the dataset using .head(10) to get an overview of the data structure.

Build a Dashboard with Plotly Dash

4. **Downloading a Dash App Skeleton:** I downloaded a template (spacex_dash_app.py), which likely provided a starting point for building the dashboard.

5. **Identifying Min and Max Payload Mass:** I extracted the minimum and maximum payload mass from the dataset, which would be useful for setting range selectors in the dashboard.

6. **Creating a Dash Application:** I initialized a Dash app, which serves as the foundation for building an interactive web application.

- For further details, please check: https://github.com/aerwins-yyw/space_x-first_stage_landing/blob/2271789ba5068fbad731edaa2a6b8bd8d05ad48b/7%20lab-jupyter-dash.ipynb

Predictive Analysis (Classification)

- Here we conduct our predictive analysis by building our classification models. The steps include:
  1. **Importing Libraries for Data Processing and Machine Learning:** I started by importing essential libraries:
     a. pandas and numpy for handling and manipulating data
     b. matplotlib.pyplot and seaborn for visualizations.
     c. sklearn.preprocessing for standardizing features.
     d. sklearn.model_selection for splitting data and performing hyperparameter tuning.

  2. **Defining a Function to Plot the Confusion Matrix:** I wrote a function, plot_confusion_matrix(), to visualize the model's performance. It computes the confusion matrix using sklearn.metrics.confusion_matrix(). Uses seaborn's heatmap to display classification accuracy. Labels axes to distinguish between successful and unsuccessful landings.

  3. **Loading SpaceX Launch Data:** I loaded dataset_part_2.csv, which contains relevant data on Falcon 9 launches. Then I extracted the target variable (Class), which indicates whether the first stage successfully landed (1) or failed (0). Then, I loaded dataset_part_3.csv, which contains the feature set (X) for training my models.

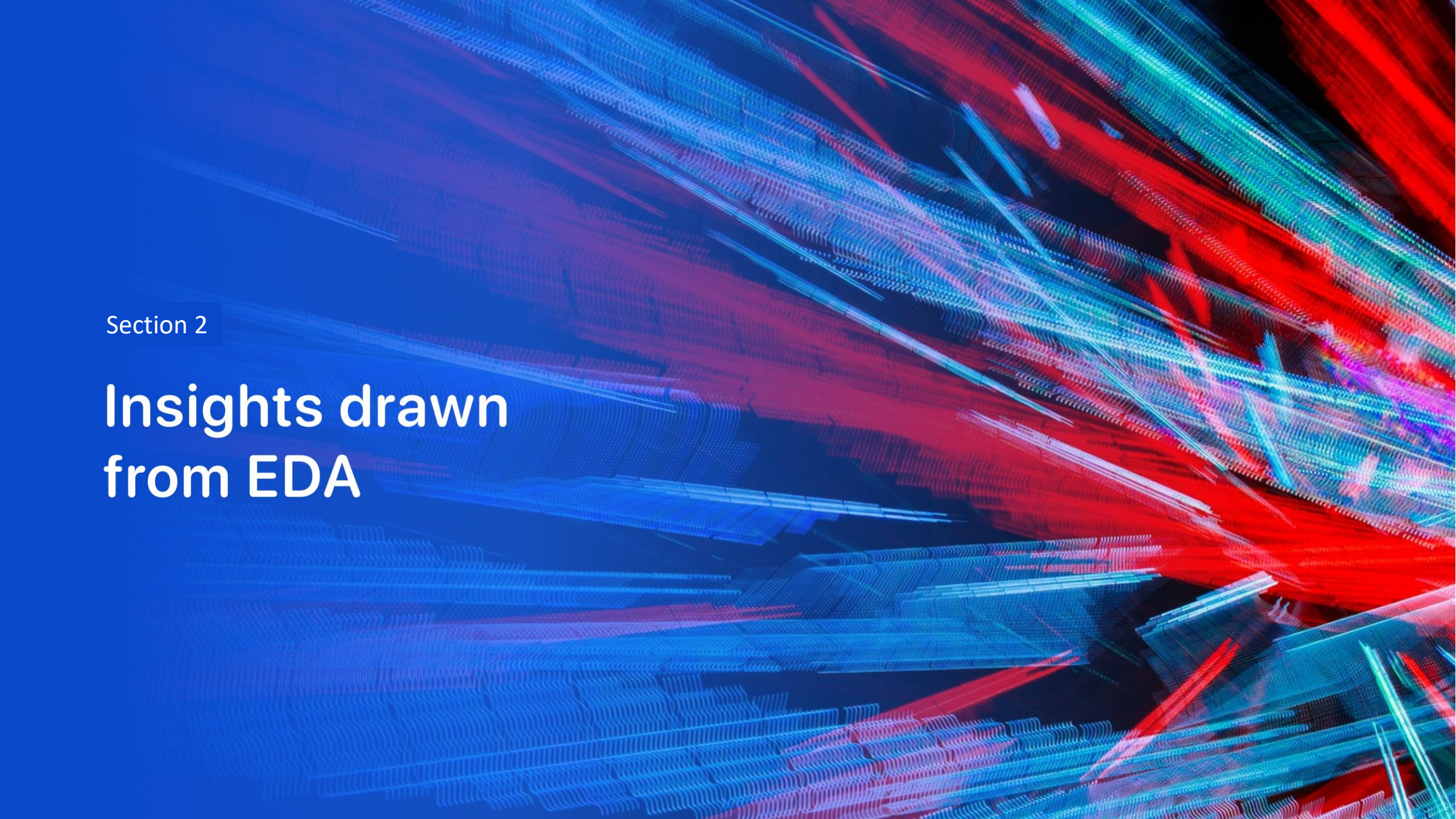
Predictive Analysis (Classification)

4. **Preprocessing the Data:** I standardized the feature set using StandardScaler() from sklearn.preprocessing to ensure that all numerical variables are on the same scale. I split the dataset into training (80%) and testing (20%) subsets using train_test_split(). This helped in evaluating model performance effectively.

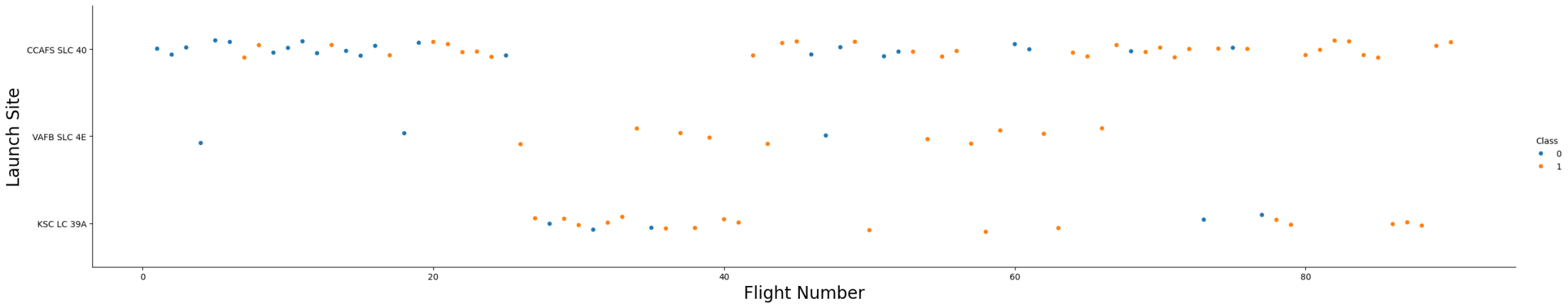
5. **Training Machine Learning Models:** I trained four classification models to predict whether a Falcon 9 first stage would successfully land:
   a. Logistic Regression: A simple linear classifier useful for binary outcomes.
   b. Support Vector Machines (SVM): Helps separate classes with an optimal hyperplane.
   c. Decision Tree: A model that learns from data by splitting features based on decision rules.
   d. K-Nearest Neighbors (KNN): Classifies data based on the majority of neighboring points.

6. **Evaluating Model Performance:** I compared the models based on accuracy and confusion matrices. I used GridSearchCV() for hyperparameter tuning, ensuring each model had the best-performing settings.

- For further details please check: https://github.com/aerwins-yyw/space_x-first_stage_landing/blob/08d5e579ea713dd3b1623eaa68dc70ceb84b88ca/8%20SpaceX-Machine-Learning-Prediction-Part-5-v1.ipynb

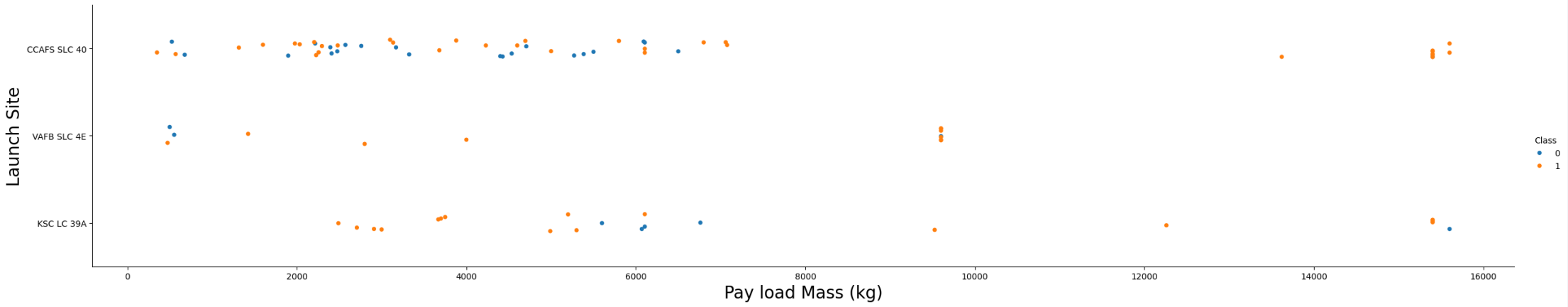

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



- Earlier flights had a lower success rate (blue = fail)
- Later flights had a higher success rate (orange = success)
- Around half of launches were from CCAFS SLC 40 launch site
- VAFB SLC 4E and KSC LC 39A have higher success rates
- We can infer that new launches have a higher success rate

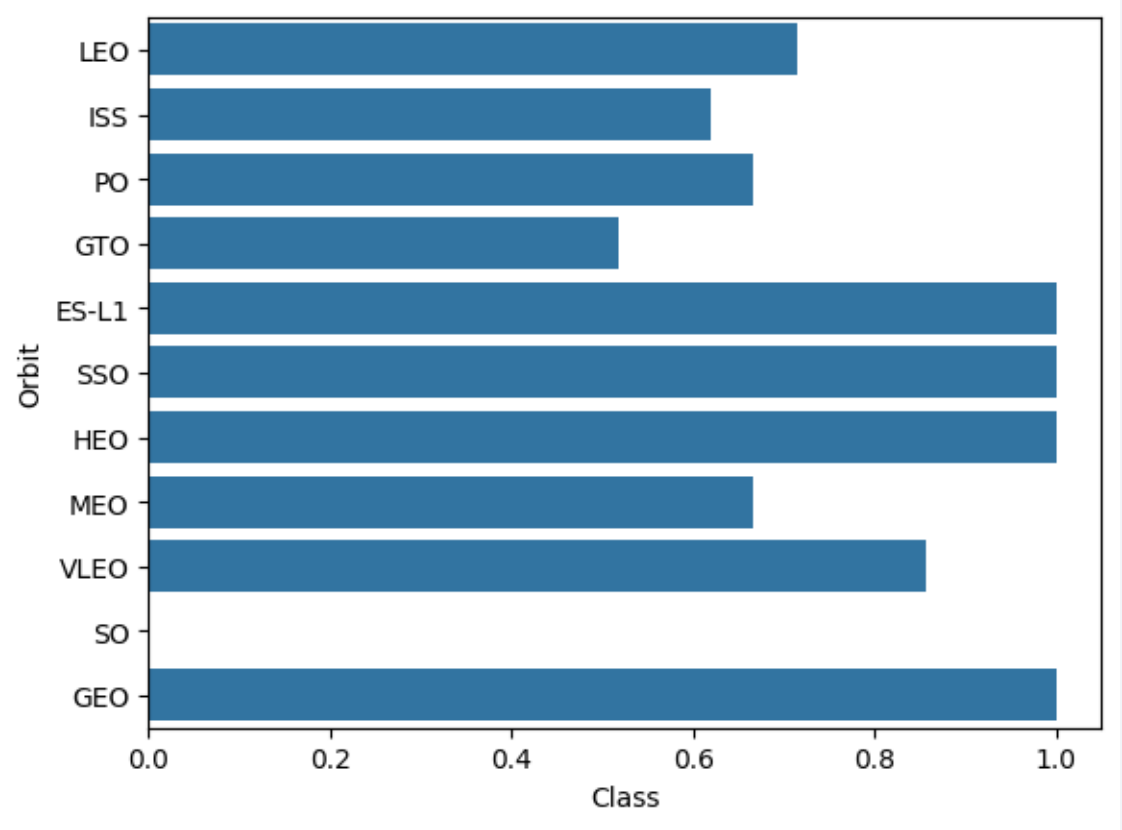
Payload vs. Launch Site



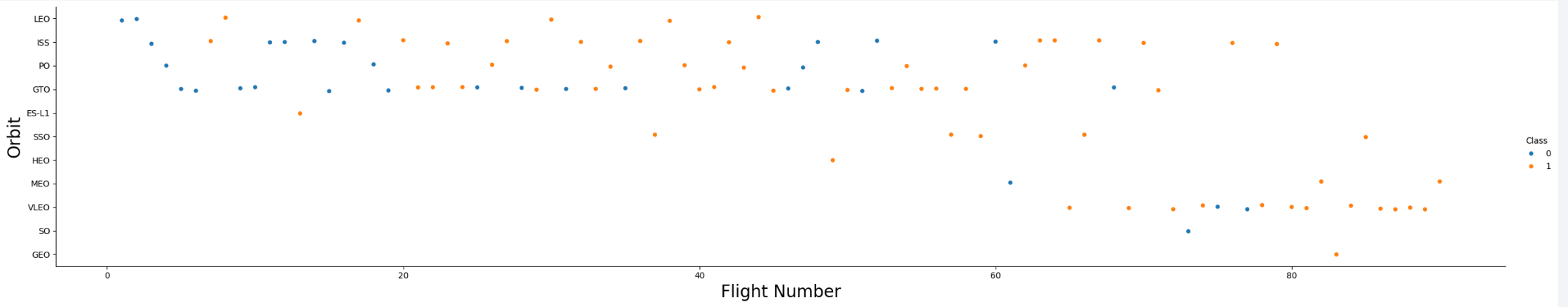
- Typically, the higher the payload mass (kg), the higher the success rate
- Most launces with a payload greater than 7,000 kg were successful
- KSC LC 39A has a 100% success rate for launches less than 5,500 kg
- VAFB SKC 4E has not launched anything greater than ~10,000 kg

Success Rate vs. Orbit Type

- 100% Success Rate:
  ES-L1, GEO, HEO and SSO
- 50%-80% Success Rate:
  GTO, ISS, LEO, MEO, PO
- 0% Success Rate:
  SO

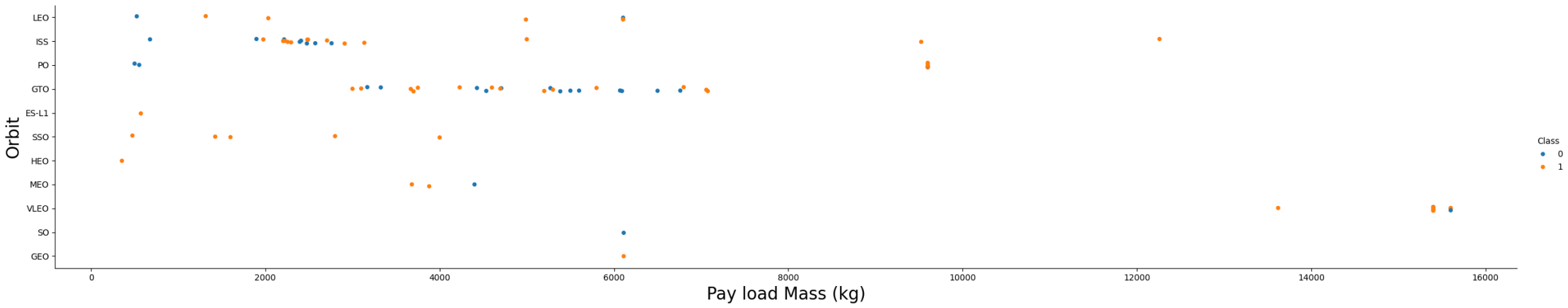


Flight Number vs. Orbit Type



- The success rate typically increases with the number of flights for each orbit
- This relationship is highly apparent for the LEO orbit
- The GTO orbit, however, does not follow this trend

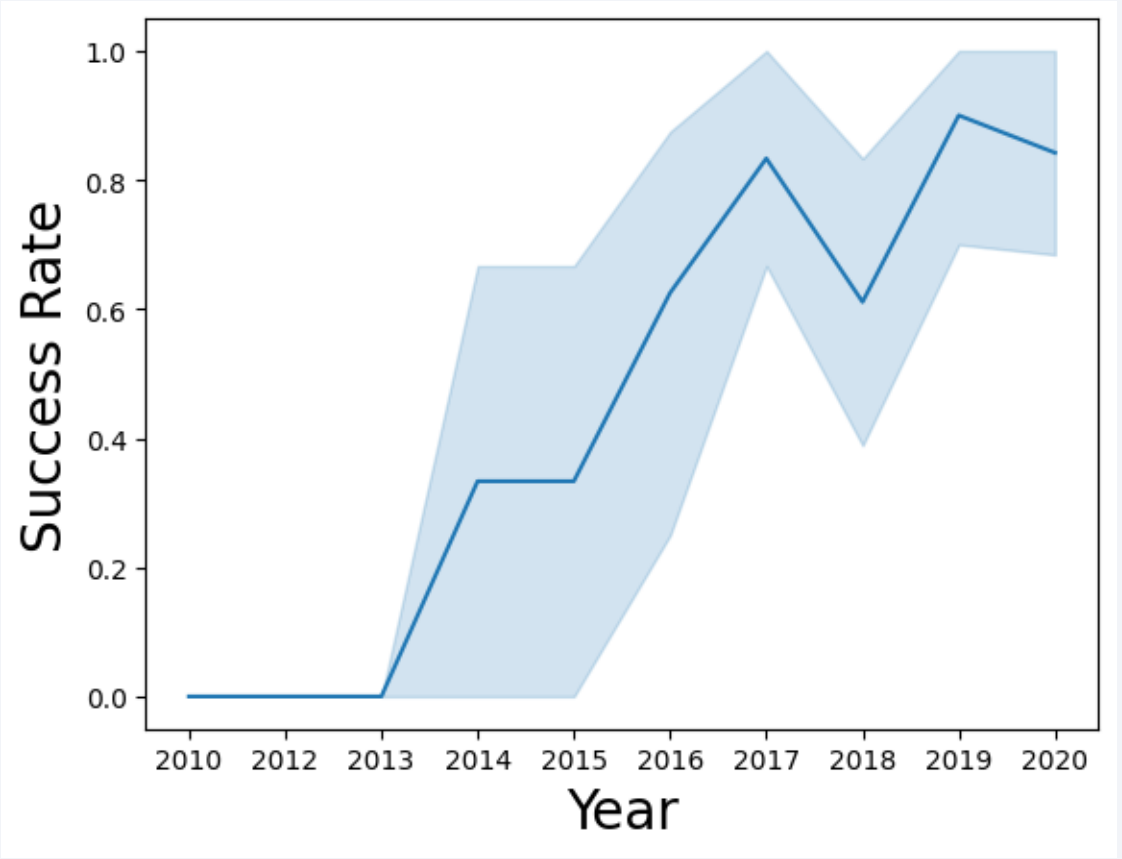
Payload vs. Orbit Type



- Heavy payloads are better with LEO, ISS and PO orbits
- The GTO orbit has mixed success with heavier payloads

Launch Success Yearly Trend

- The success rate improved from 2013-2017 and 2018-2019
- The success rate decreased from 2017-2018 and from 2019-2020
- Overall, the success rate has improved since 2013



Launch Site

	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610745

The launch sites are:
CCAFS LC-40, CCAFS SLC-40, KSC LC-39A, VAFB SLC-4E.

The table below shows the 5 records from the launch site CCAFS LC-40.

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

Payload Mass

- 45,596 kg (total) carried by boosters launched by NASA (CRS)
- 2,928 kg (average) carried by booster version F9 v1.1

Landing Mission Info

- 1st Successful Landing in Ground Pad is on 12/22/2015
- Booster mass greater than 4,000 but less than 6,000: JSCAT-14, JSCAT-16, SES-10, SES-11 / EchoStar 105
- Total Number of Successful and Failed Mission Outcomes:
  - 1 Failure in Flight
  - 99 Success
  - 1 Success (payload status unclear)

Boosters Carried Max Payload

- Boosters that Carried Max Payload:
  - 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

- The failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015:

month	Date	Booster_Version	Launch_Site	Landing _Outcome
01	10-01-2015	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	14-04-2015	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

Landing Outcomes

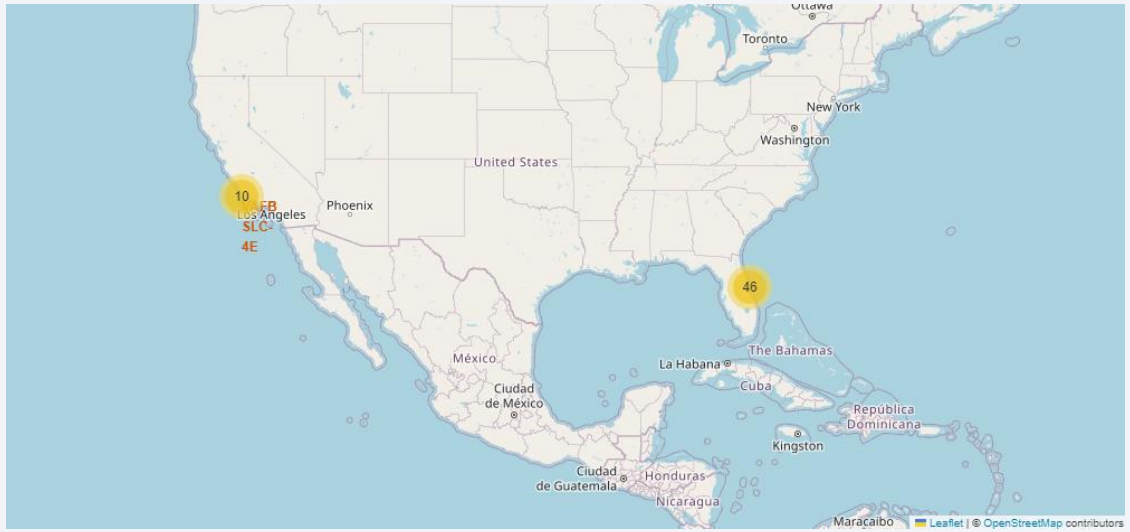
- The count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) **between the date 2010-06-04 and 2017-03-20**, in descending order

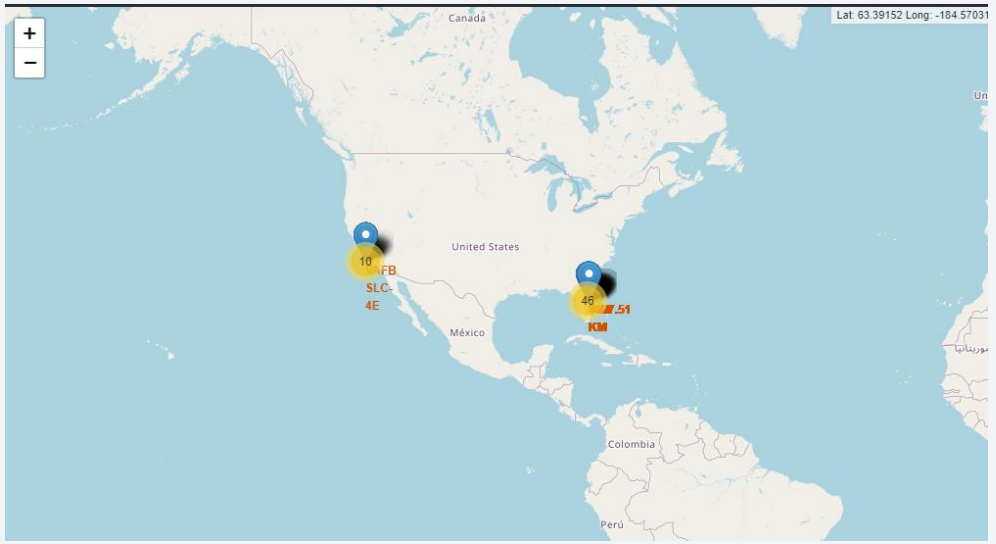
Landing _Outcome	count_outcomes
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

Section 3

Launch Sites Proximities Analysis

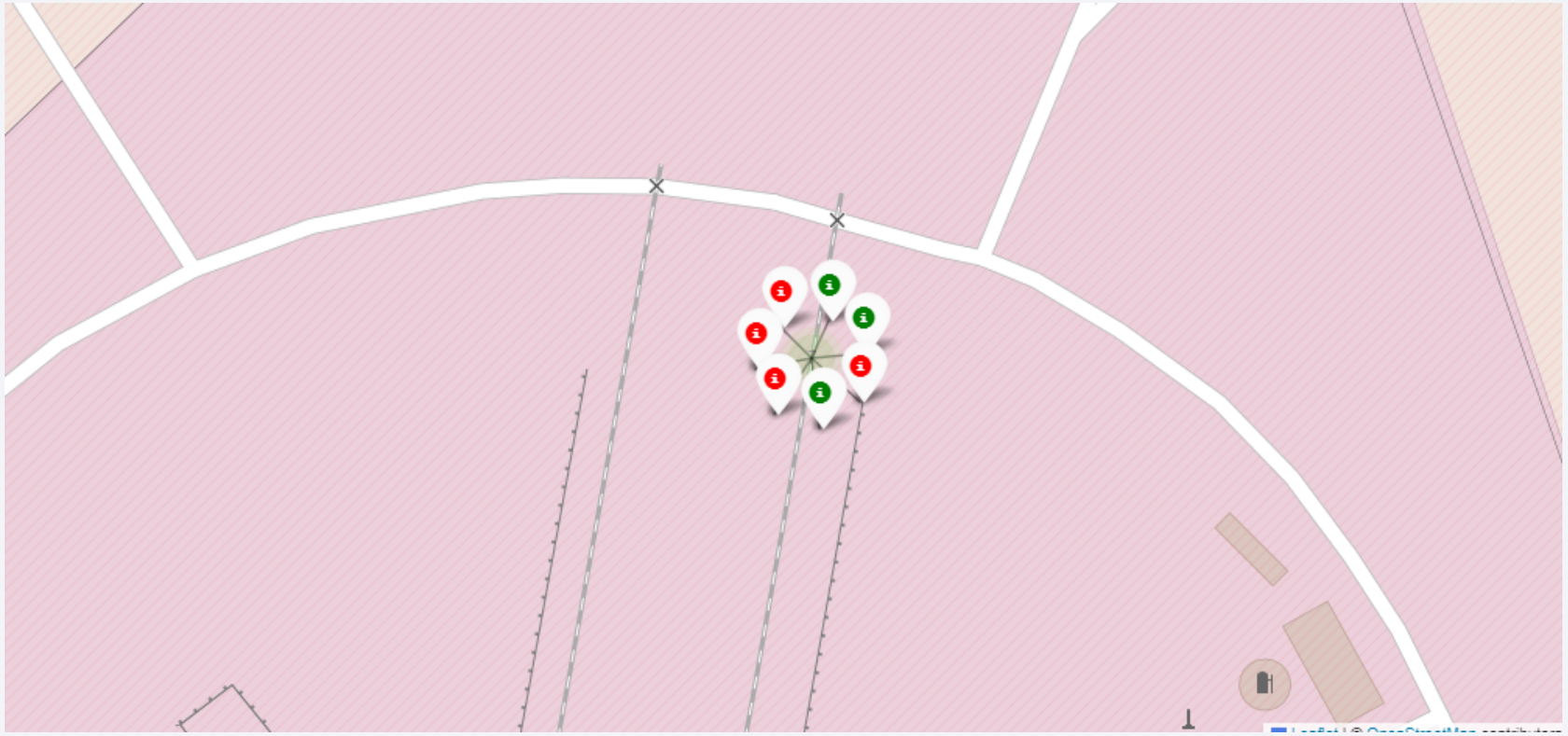
Launch Sites Location





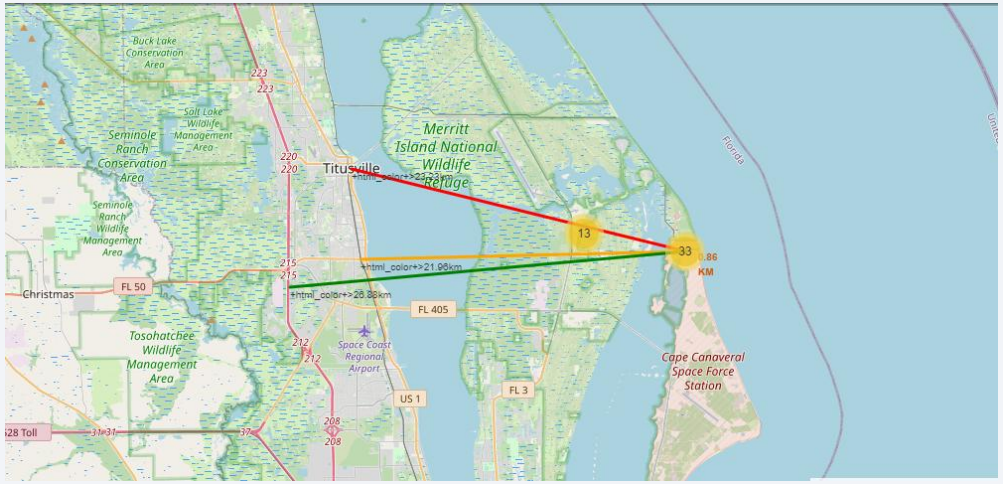
- It appears that the launch sites are close to the equator.
- The closer the launch site to the equator, the easier it is to launch to equatorial orbit, and the more help you get from Earth's rotation for a prograde orbit. Rockets launched from sites near the equator get an additional natural boost - due to the rotational speed of earth - that helps save the cost of putting in extra fuel and boosters.

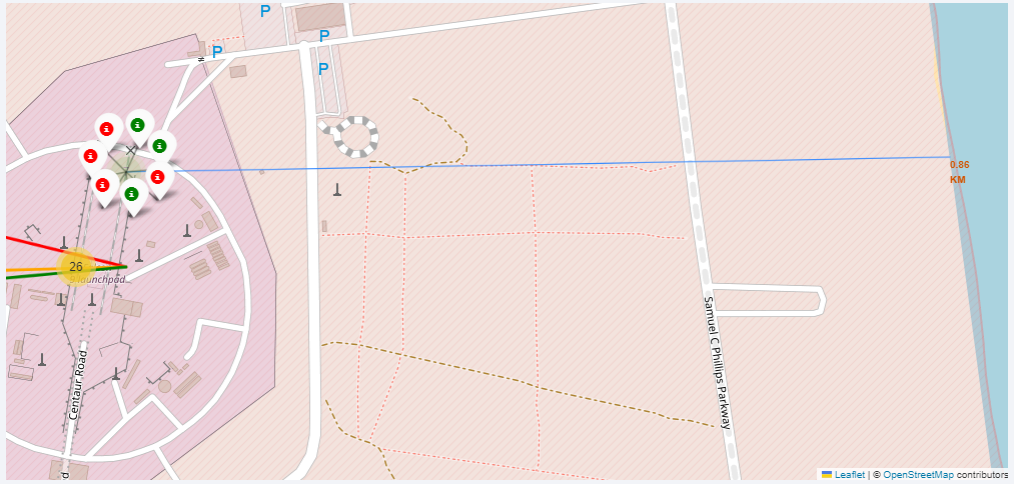
CCAFS SLC-40



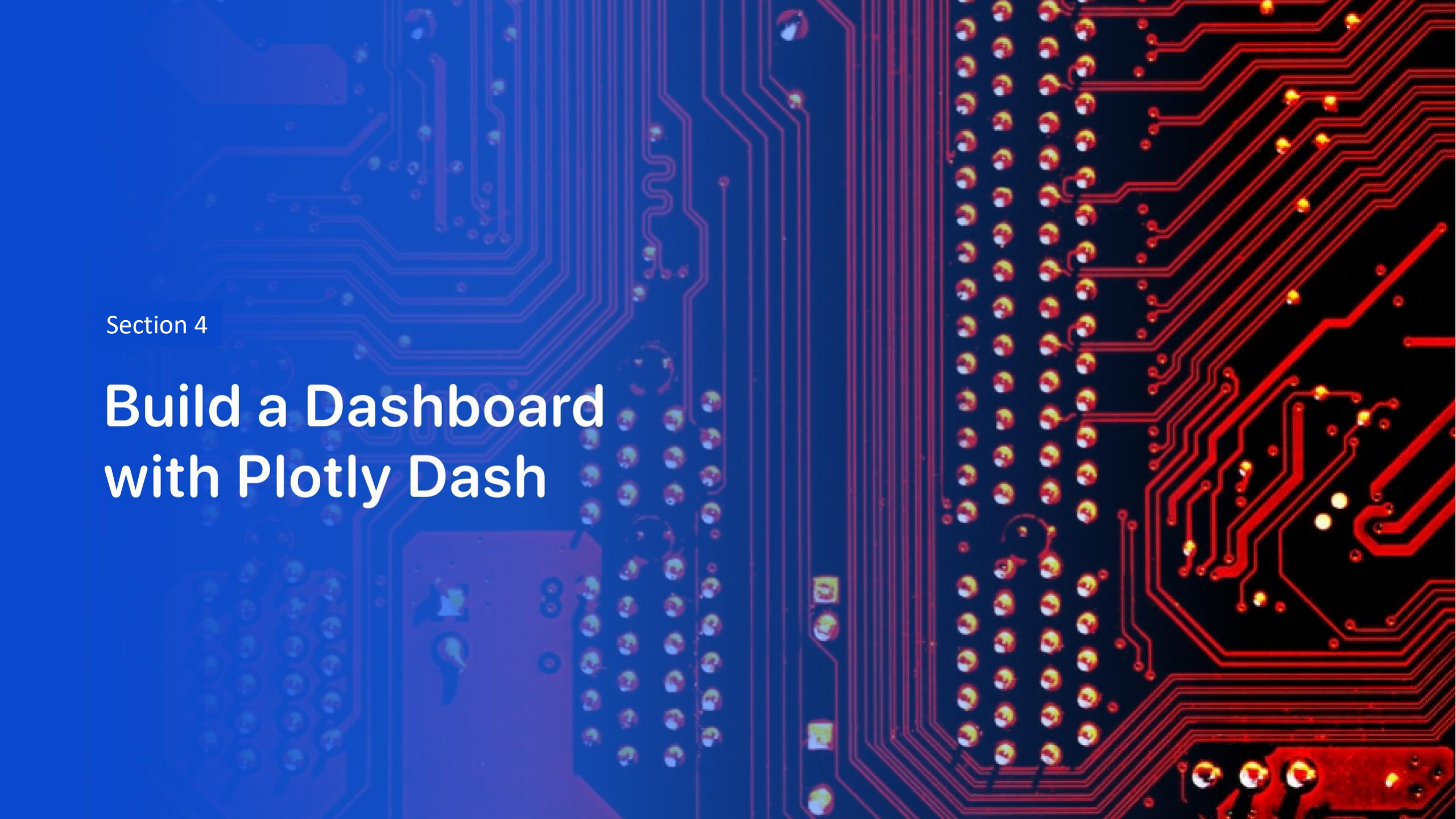
- Launch site CCAFS SLC-40 has a 3/7 success rate (42.9%). The Green ones are successful launches while the red ones are the failed ones.

Proximities of Launch Sites

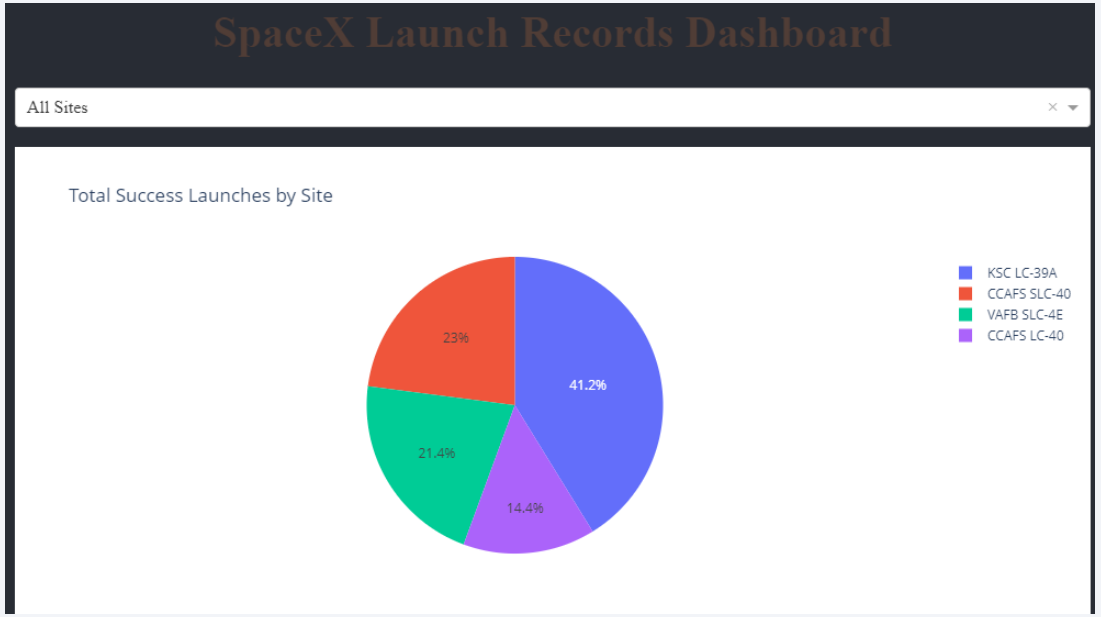




- For the launch site CCAFS SLC-40, it is 0.86 km from nearest coastline, 21.96 km from nearest railway, 23.23 km from nearest city, and 26.88 km from nearest highway.
- Why?
  - Coasts: help ensure that spent stages dropped along the launch path or failed launches don't fall on people or property.
  - Safety / Security: needs to be an exclusion zone around the launch site to keep unauthorized people away and keep people safe.
  - Transportation/Infrastructure and Cities: need to be away from anything a failed launch can damage, but still close enough to roads/rails/docks to be able to bring people and material to or from it in support of launch activities

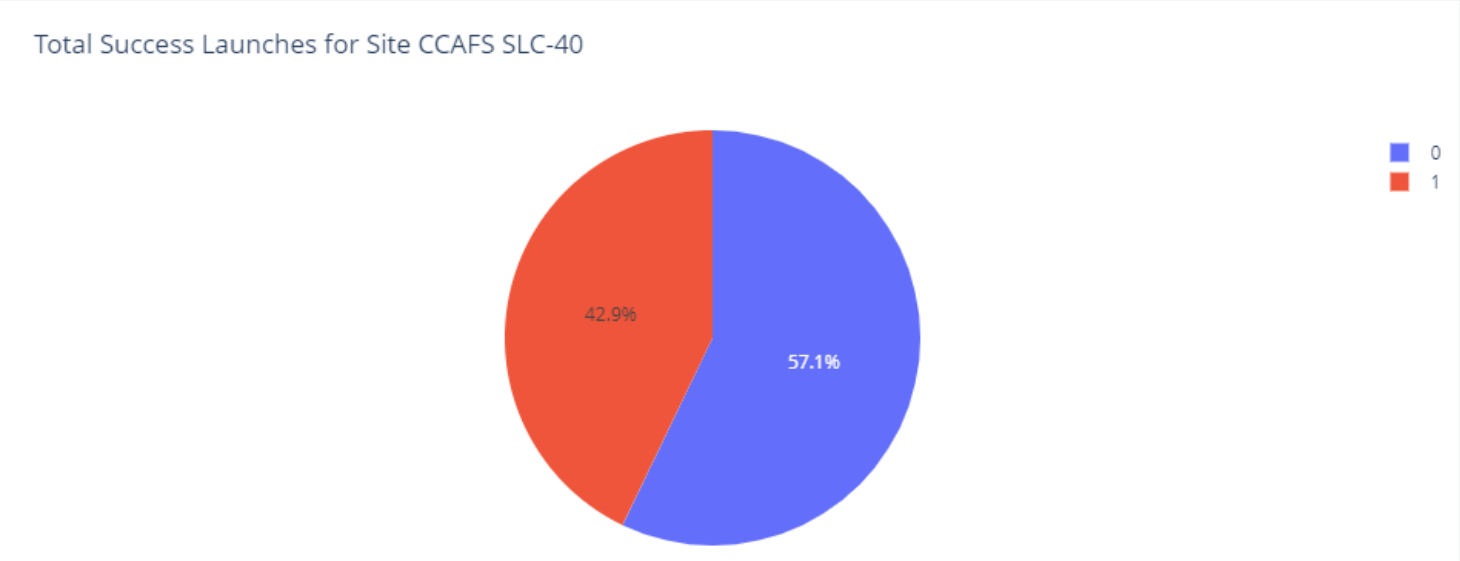

Section 4
Build a Dashboard with Plotly Dash

Total Success Launches by Site



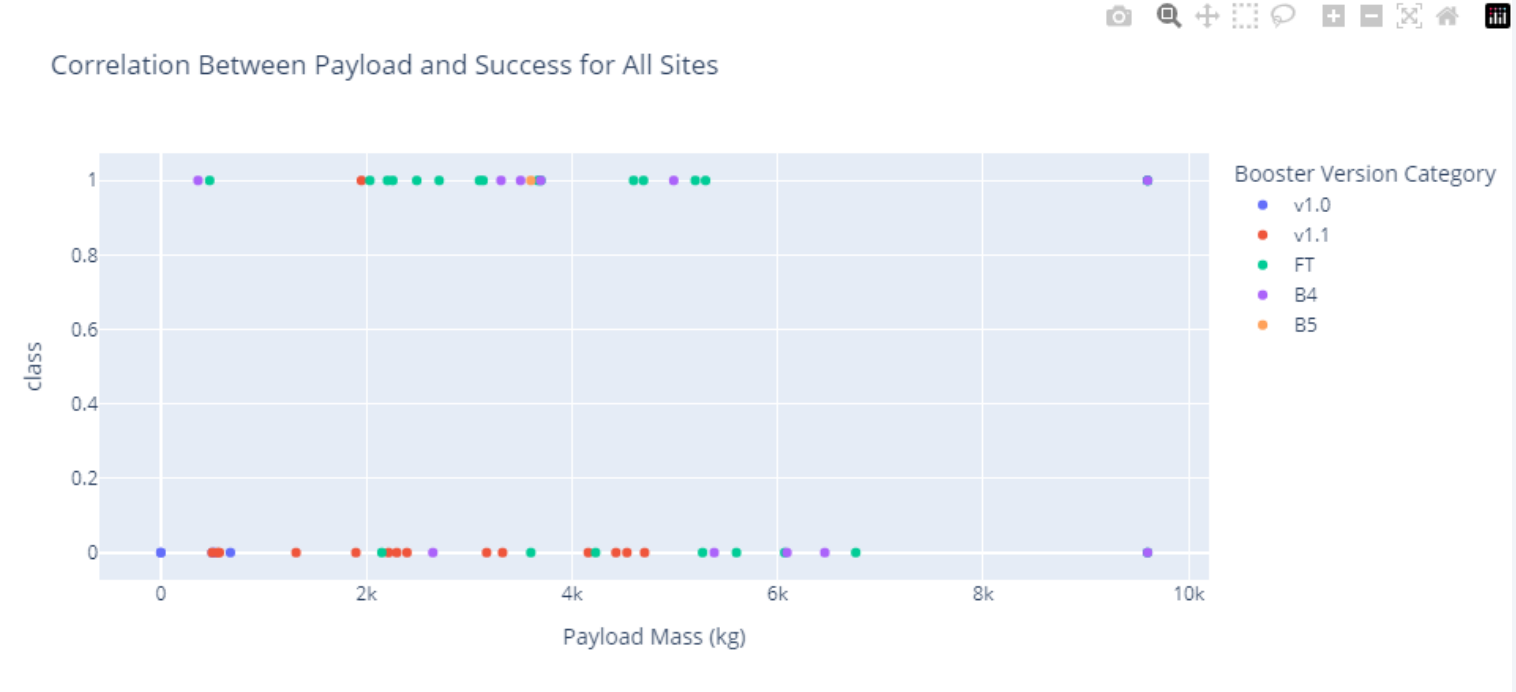
- Through the pie chart that we show using Plotly Dash, we see that KSC LC-39A is the site that has higher percentage of the total success of all sites, accounting to 41.2% of the total.

Total Success Launches for Site CCAFS SLC-40



- Meanwhile, the site that has the highest success rate is CCAFS SLC-40, which made 42% success out of their launches.

Correlation between Payload and Success for All Sites



- Class 1 indicating successful outcome and 0 indicating an unsuccessful outcome
- Payloads between 2,000 kg and 5,000 kg have the highest success rate. To be more specific, it is between 2880 and 3840.

Section 5

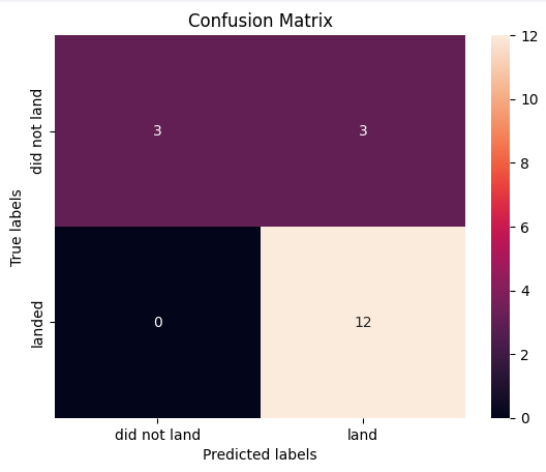
Predictive Analysis (Classification)

Classification Accuracy

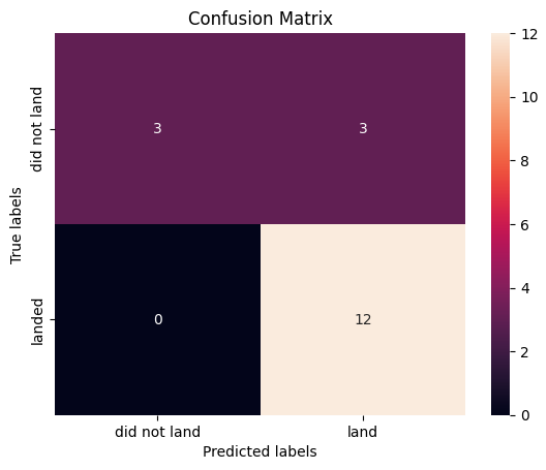
	Model	Train Accuracy Score	Test Accuracy Score
0	Logistic Regression	0.846429	0.833333
1	SVM	0.848214	0.833333
2	Decision Tree	0.875000	0.833333
3	KNN	0.848214	0.833333

- The result shows that all models despite showing different performance on the training data, with Decision Tree producing the highest accuracy, all models' accuracy on the test set are almost identical here, with the score of 0.833.

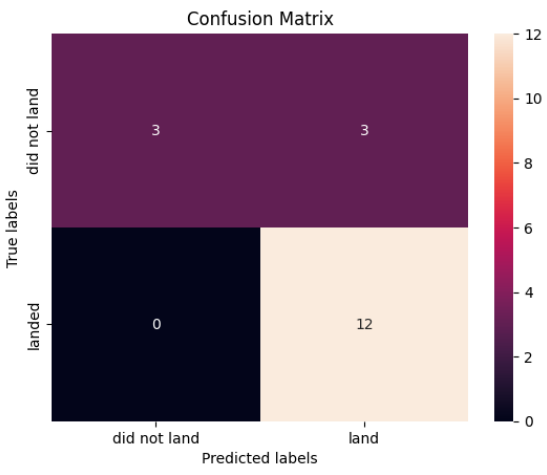
Confusion Matrix



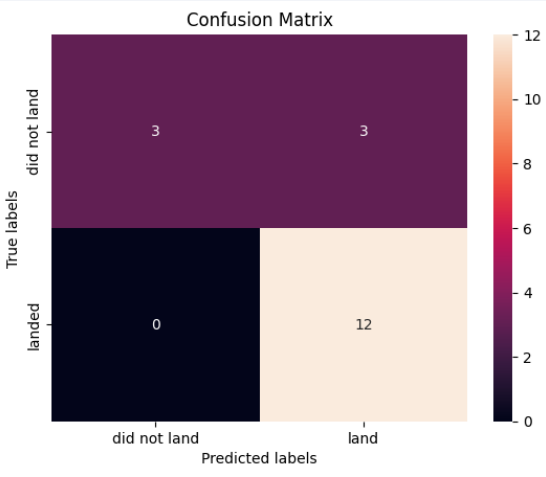
Logistic Regression



SVM



Decision Tree



KNN

- The confusion matrix applied on the test data shows similar result. The performance of all models are identical.

Conclusions

- Any of these models can be used for prediction, but simpler models like Logistic Regression or SVM might be preferred due to their interpretability and robustness.
- Decision Tree has the highest training accuracy but does not generalize better than the others.
- If needed, further hyperparameter tuning or trying ensemble methods (like Random Forest) might improve performance

Thank you!