IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Isaach Sagel
12/16/2024

Outline

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

Executive Summary

- Prediction of SpaceX's Falcon 9 rockets landing for reusability is crucial for reaching their $62 million budget, compared to $165 million for others. Therefore, it is within the interest of the company the implementation of automated data collection techniques.
- Some of the techniques are python API's, web scraping, Exploratory Data Analysis(EDA), feature engineering and Machine Learning Algorithms.
- The Falcon 9 has a 67 % rate of success according to past registered launches.
- Booster versions F9 FT B1XXX can be deployed to land on drone ships up to 5300 kg payload.
- The LEO orbit(71%) and the ISS orbit(61%) have great opportunity for rockets reaching 2000 kg payload and above.
- The map shows that launch sites, such as CCAFS SLC-40 and KSC LC-39A, are located very close to the coast. Launch sites are located far from cities, such as Titusville, more than 23 km away.
- Machine Learning models were performed to find best hyperparameters and accuracy, Decision Tree have the best score with 0.8625, so it is the best suited for exploratory data analysis for landing of Falcon 9's.

Introduction

- In this project, the main goal is to investigate and predict the results of Falcon 9 rocket landings. To achieve this, it have been used various data collection methods, such as the SpaceX API and web scraping. Leveraging the power of Python libraries, SQL and a number of visualization tools, we have meticulously collected and prepared the necessary data.
- The approach employs advanced machine learning techniques for data validation and model fitting, ensuring accurate predictions and insightful analysis. With this project, the aim is to improve our understanding of the Falcon 9 landings and contribute valuable information to the aerospace community.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Request the data form SpaceX API and Wikipedia, later clean the data.
- Perform data wrangling
  - Collect landing outcomes data from JSON and HTML tables for visualization and analysis.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Standardize the data and find best Hyperparameters

Data Collection

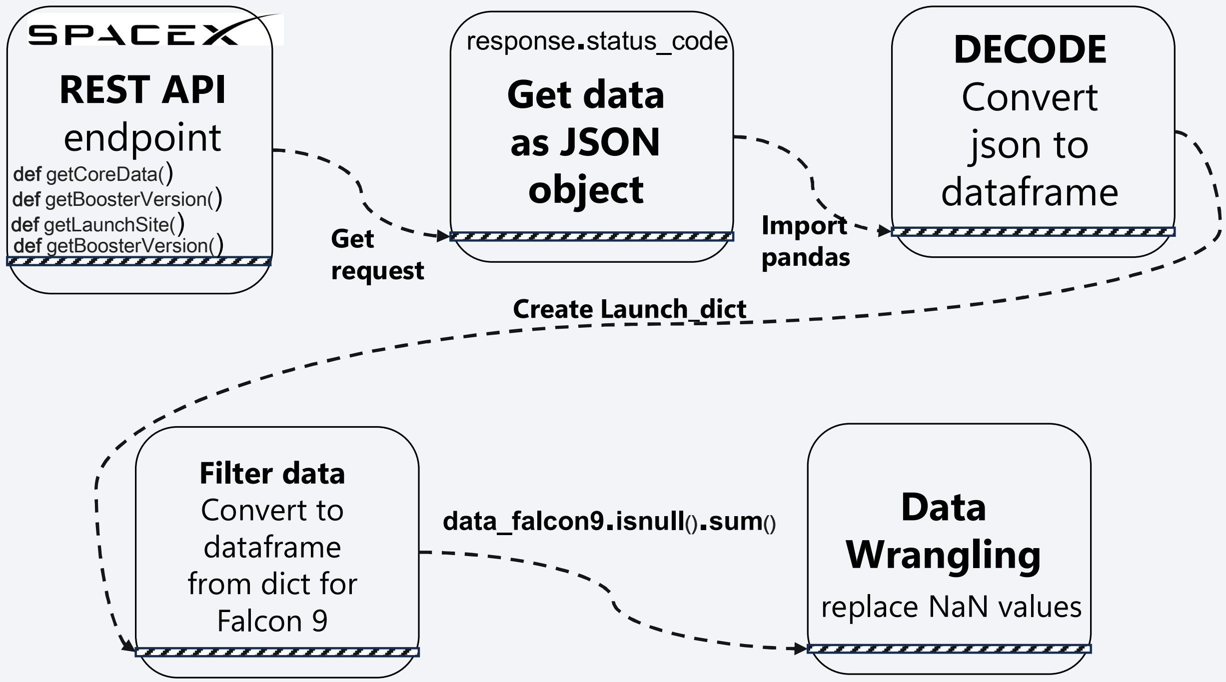
- Data API tehnique was obtained from *https://api.spacexdata.com/v3* , a REST API. The dataset_part_1.csv was created for later data Wrangling.
- The webscraping technique used Wikipedia *https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches* as the main source. The spacex_web_scraped.csv was created for Heavy Launches record.

Data Collection – SpaceX API

- Whole Process

1. Install and import libraries and auxiliary functions
2. Request the data from API in JSON format
3. Convert to dataframe with pandas
4. Filter data using dictionary and then selecting Falcon 9 launches.
5. Deal with missing values replacing them with .mean() by using .replace()

- https://github.com/isaachsgel/IBM-Data-Science-Course/blob/main/jupyter-labs-spacex-data-collection-api-v2.ipynb

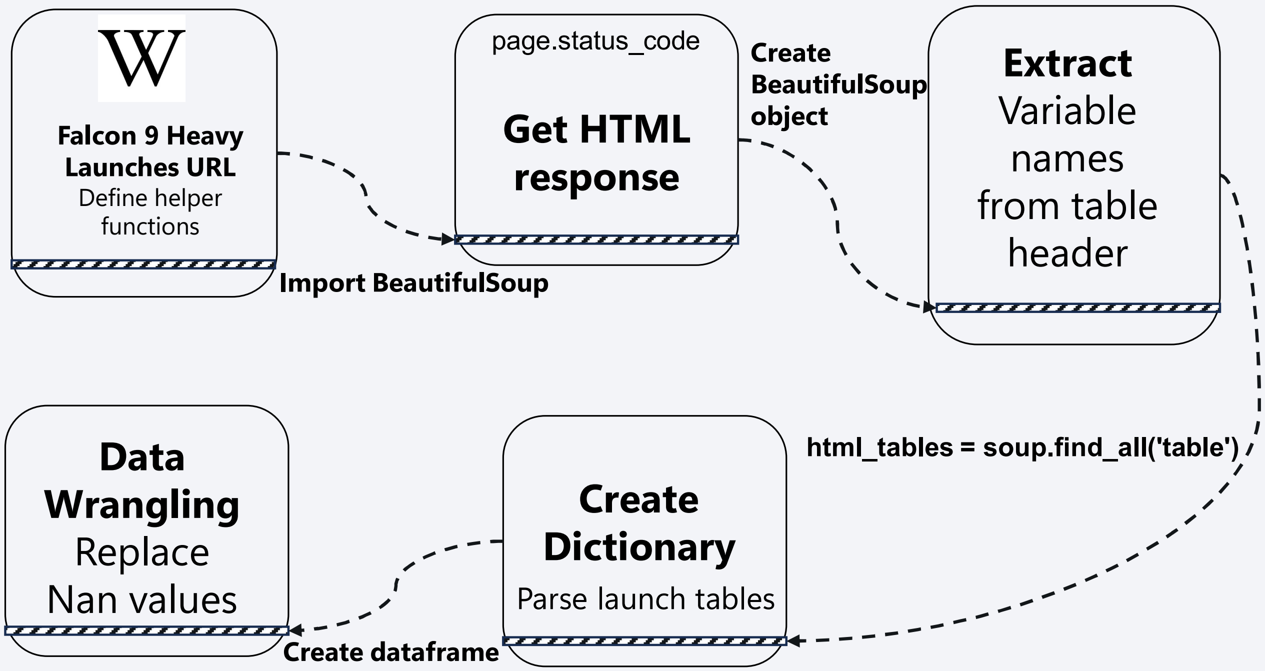


Data Collection - Scraping

- Whole Process

1. Install and import libraries and create helper functions
2. Request the data from Wiki page as HTML response
3. Using BeautifulSoup library, create object using soup = BeautifulSoup(page.text, 'html.parser')
4. Extract columns from header using <th> elements.
5. Create the data frame by parsing the Launch HTML tables from aa empty dictionary
6. Deal with missing values.

- https://github.com/isaachsgel/IBM-Data-Science-Course/blob/main/jupyter-labs-webscraping.ipynb



Data Wrangling

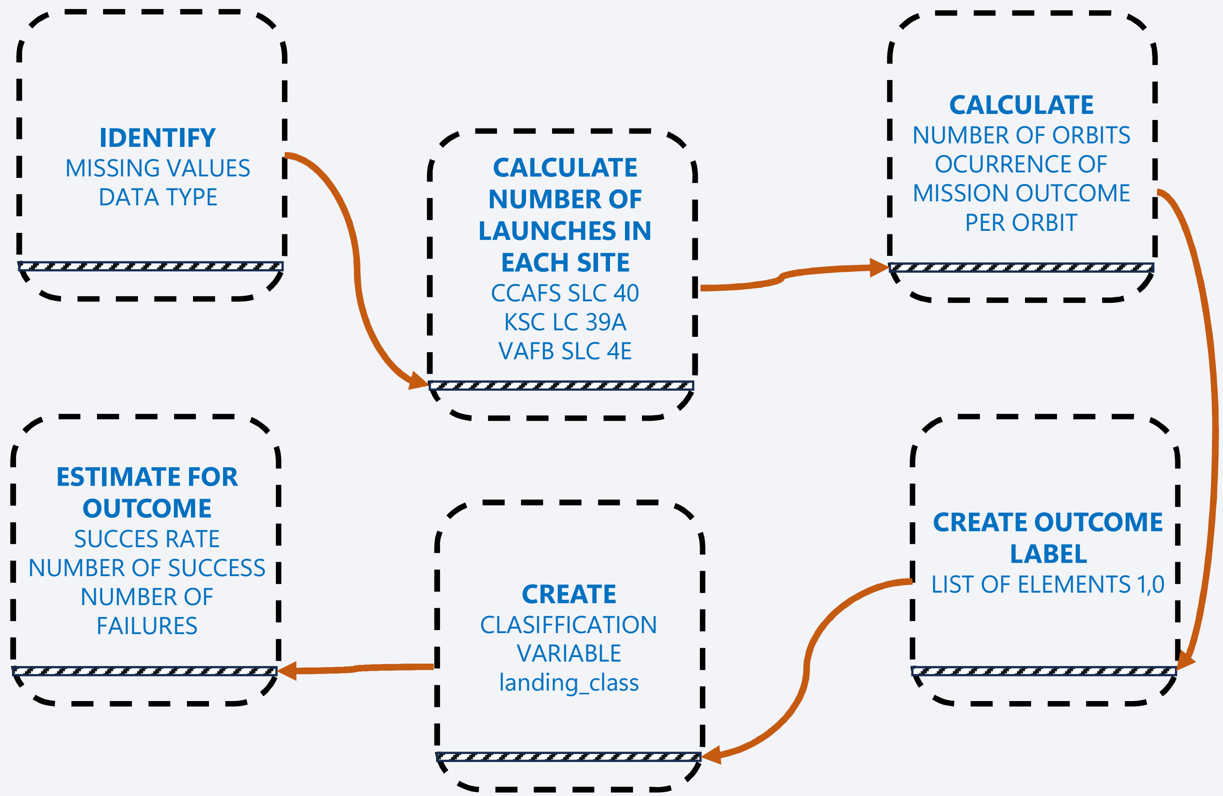
- Exploratory data analysis (EDA) was performed to find some patterns in the data and determine what would be the label to train supervised models.
- First, is crucial to identify the percentage of missing values from the dataset obtained from last section (REST API), then identify which columns are numeric and which are categorical.
- At last, using column Outcome, create a list whose element is 0 if the corresponding row of the result is in the bad_outcome; otherwise, it will be 1. Then assign it to the landing_class variable defined as a label.

Data Wrangling

- Whole Process

1. Determine missing values with .isnull().sum().
2. Identify which columns are numerical and categorical,
3. Use the method value_counts() on the column LaunchSite to determine the number of launches on each site.
4. Use the method value_counts() to determine the number and occurrence of each orbit in the column Orbit.
5. Create a landing outcome label using Outcome column and define a classification variable with a list of elements were success represent 1 and failure 0.

- https://github.com/isaachsgel/IBM-Data-Science-Course/blob/main/labs-jupyter-spacex-Data%20wrangling-v2.ipynb



EDA with Data Visualization

- Summary of plots

  Catplot 1: Flight number vs MassPayload

  Catplot 2: Flight number vs LaunchSite

  Catplot 3: MassPayload vs LaunchSite

  Barchart 1: success rate per orbit type

  Catplot 4: Flight number vs orbit type

  Catplot 5: MassPayload vs orbit type

  Line chart 1: historical success rate

EDA with Data Visualization

- Seaborn and matplotlib used for static charts. The data source comes from historical 2010-2020 launches.
- Key insights: Payload mass affects success rate as in the case of POLAR orbit at heavy payloads versus masses under 1000 kg.
- Higher success rates observed for specific orbits and launch sites such as ISS, GTO and LEO around 70-60%.

- https://github.com/isaachsgel/IBM-Data-Science-Course/blob/main/jupyter-labs-eda-dataviz-v2.ipynb

EDA with SQL

Summary of queries performed(see notebook link for query text code)

- Display the names of the unique launch sites in the space mission.
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first succesful landing outcome in ground pad was acheived.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- 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.
- 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.

- https://github.com/isaachsgel/IBM-Data-Science-Course/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Summary of map objects

  folium.Circle and folium.marker : highlight circle with text label to coordinate in site map.

  markerCluster: adjunct multiple markers to specific coordinate, or simplify circle highlighted area containing large quantity of marker in the same coordinate.

  folium.FeatureGroup: group multiple elements such as markers.

  folium.Polyline: draw distance and show value to roads, railways and cities.

Build an Interactive Map with Folium

- Visualized launch sites with markers, circles, and lines.
- Highlighted key locations such as NASA JSC Space Station.
- Interactive elements show data insights on maps. For example, calculating the distance to towns and highways have a role in safety and access to launch sites.
- The map shows the launch sites, such as CCAFS SLC-40 and KSC LC-39A, are located very close to the coastline.
- The launch sites are situated far from cities, such as Titusville, which is over 23 km away.
- From the map and the symbols shown, the launch sites appear to be relatively close to railways. Railways could be used for transporting heavy payloads and equipment needed for the launches.

https://github.com/isaachsgel/IBM-Data-Science-Course/blob/main/lab-jupyter-launch-site-location-v2.ipynb

Build a Dashboard with Plotly Dash

- This application integrates interactive components designed to provide real-time visual analysis of SpaceX launch data. The following key features enhance its functionality and ease of use:
  - Launch point drop-down menu: a dynamic drop-down menu allows users to select from four different launch points or analyze all points collectively. This feature enables targeted exploration of launch data based on user preferences.
  - Dynamic pie chart display: A callback function seamlessly generates and updates the success graph based on the selected launch location. Leveraging this interaction, users can visually interpret the success rates of specific launch points or compare the overall performance of all locations.
  - Payload Selector: A slider facilitates the selection of payload mass ranges, giving users the flexibility to filter and analyze trends within custom payload ranges. This functionality allows for the identification of underlying patterns in payload success correlations.
- Together, these components foster an intuitive and insightful exploration of SpaceX launch metrics through interactive visualizations.

Build a Dashboard with Plotly Dash

Summary:

- Interactive dashboard with dropdowns and sliders.
- Visualized metrics: payload vs success, orbit vs success.
- The plots and charts provided insights on critical success factors
- Plot success-payload-scatter-chart

https://github.com/isaachsgel/IBM-Data-Science-Course/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

The structured methodology used to predict whether the first stage of a launch will land successfully is described below, taking advantage of the knowledge acquired in previous laboratory exercises:

1. NumPy matrix construction: a specific NumPy matrix was generated from the Class column of the data set to facilitate the numerical operations essential for model development.
2. Data normalization: Input data underwent a rigorous normalization process to ensure consistency and comparability, optimizing model performance.

Predictive Analysis (Classification)

2. Data partitioning with train_test_split: The dataset was methodically partitioned into training and test subsets, separating features (X) and target variables (Y) to enable efficient model evaluation.
3. Hyperparameter optimization: Exhaustive hyperparameter tuning of various classifiers, such as logistic regression, support vector machines (SVM), decision trees and k-nearest neighbors (KNN), was performed. This step aimed to identify the optimal parameter settings for each algorithm.

Predictive Analysis (Classification)

4. Model performance evaluation: The performance of each method was rigorously evaluated with test data to determine the most effective model, ensuring robust and reliable predictions.

This streamlined approach reflects a systematic progression through data preparation, model tuning, and performance validation to achieve accurate and reliable predictions.

Results

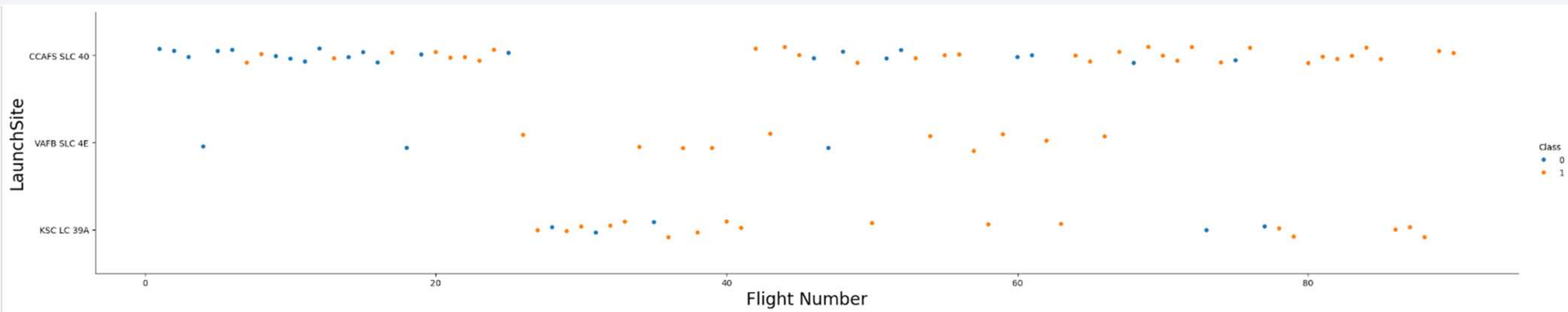
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

https://github.com/isaachsgel/IBM-Data-Science-Course/blob/main/SpaceX-Machine-Learning-Prediction-Part-5-v1.ipynb

Section 2

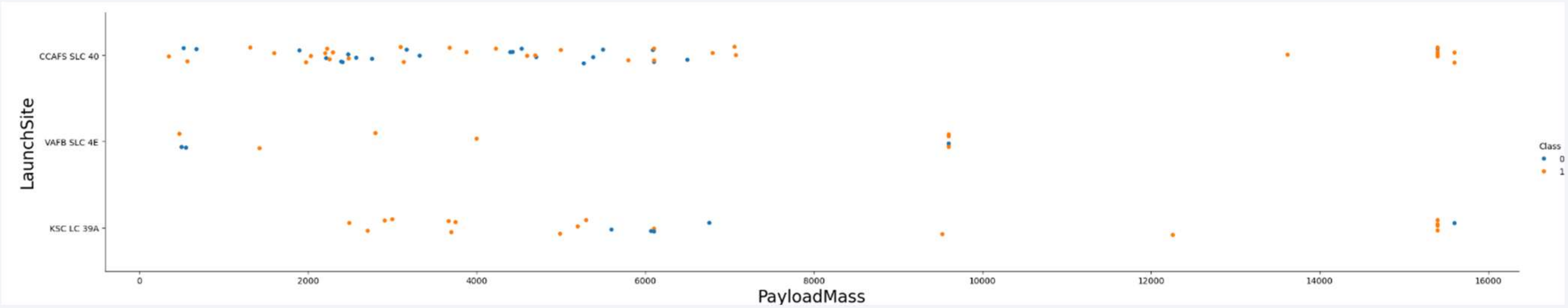
Insights drawn from EDA

Flight Number vs. Launch Site



- CCAFS SLC 40 has the largest number of launches, indicating that it is a high-utilization site for SpaceX missions.
- VAFB SLC 4E infrequent launch pattern may point to its specialized use, possibly for missions requiring unique orbital trajectories.
- KSC LC 39A majority of launches in later flight numbers are successful, demonstrating an upward trajectory in reliability and performance.

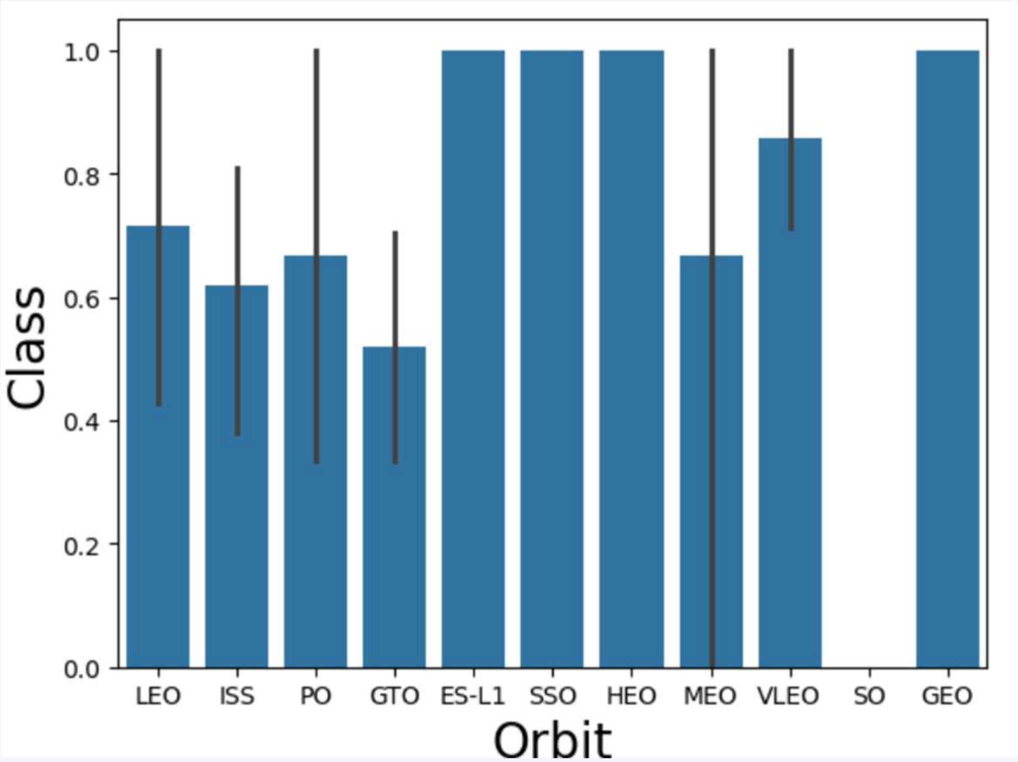
Payload vs. Launch Site



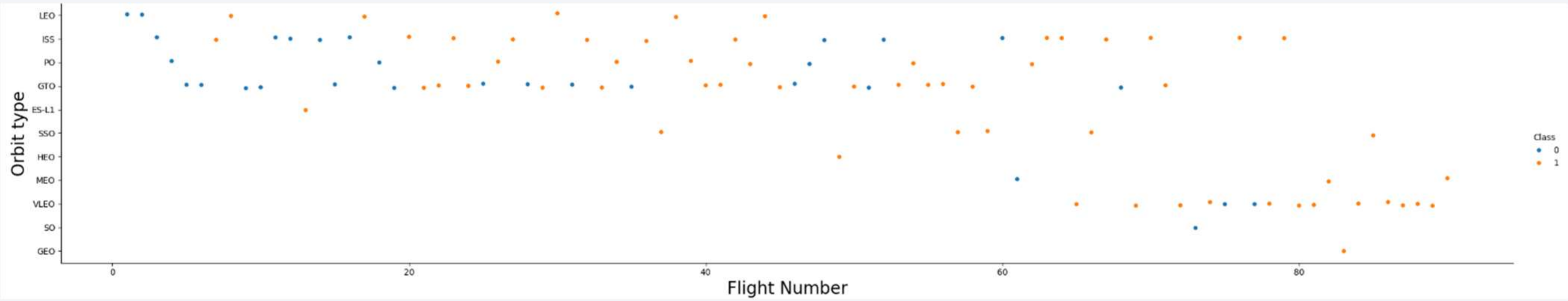
- CCAFS SLC 40 and KSC LC 39A demonstrate exceptional performance for heavy payloads >10,000 kg, achieving near-perfect success rates
- **VAFB SLC 4E** shows mixed results, especially for smaller payloads, possibly highlighting technical constraints or its role in specialized missions.

Success Rate vs. Orbit Type

- The analysis reveals varying levels of success across key orbit types:
- LEO: Performs the best among the analyzed orbits, with approximately 70% success.
- ISS: Shows challenges, likely due to the complexity of ISS docking missions.
- PO: Moderate success rate, critical for earth observation missions.
- GTO: Lowest success rate, reflecting the complexity and precision required for geostationary transfers.

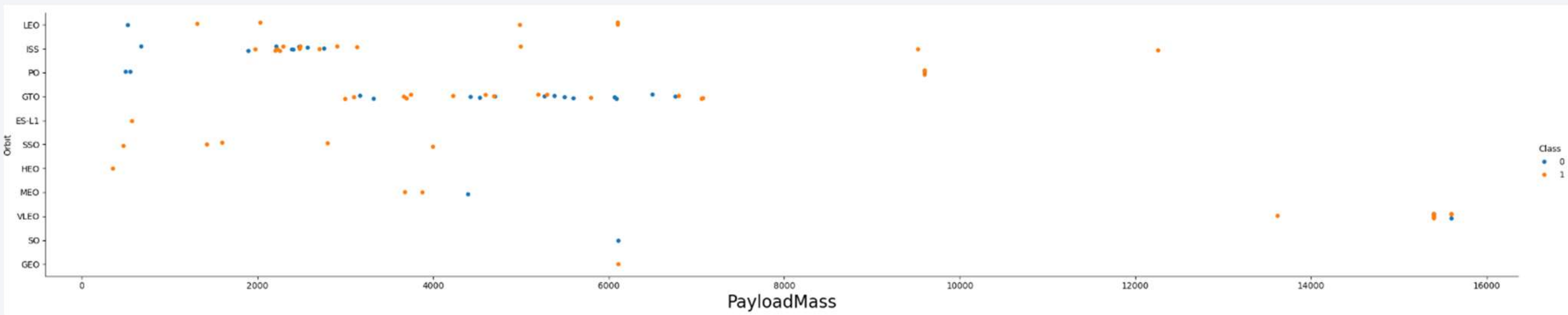


Flight Number vs. Orbit Type



- The scatter plot emphasizes the importance of orbit-specific challenges. While LEO demonstrates reliable success, GTO highlights significant areas for improvement. Focused innovation in launch technology and mission execution will be crucial to enhance success rates across more challenging orbits.

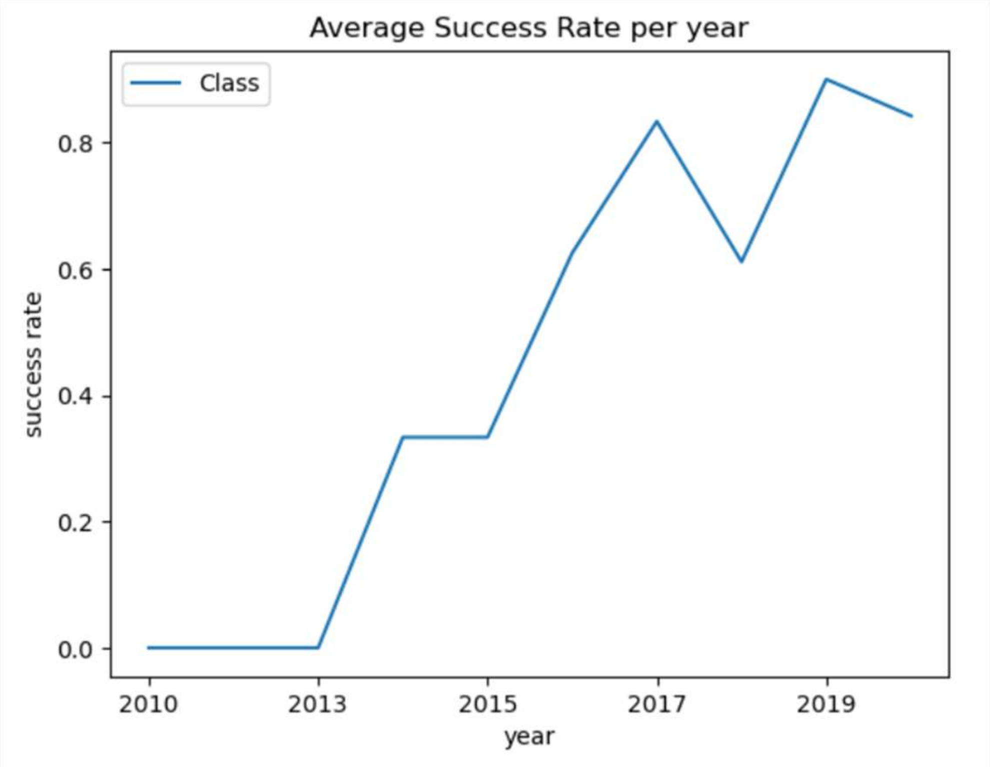
Payload vs. Orbit Type



- GTO missions primarily focus on heavier payloads (5000–16000 kg). There is a notable presence of failures (blue dots), although some successes (orange dots) are achieved.
- PO missions span medium payload ranges (1000–6000 kg), with a balance of successes and failures. A cluster of failures is observed around the 3000–4000 kg range.
- The consistency of payload mass for ISS missions suggests a focused purpose—resupply and crewed missions. The higher success rate highlights improved precision for such missions over time.
- LEO exhibits a high frequency of launches across a wide range of payload masses (from low to high). Most launches are successful (orange dots), especially for payloads below 6000 kg. A few failures (blue dots) occur at lower payloads.

Launch Success Yearly Trend

- It can be observed that the success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.



All Launch Site Names

- There are four sites
- Distinct clause allows finding unique values from database my_data.db

[11]:
```
%%sql
select distinct LAUNCH_SITE
from SPACEXTBL;
```

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

[11]:

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

Launch Site Names Begin with 'CCA'

- WHERE, LIKE and LIMIT allows finding the string ‘CCA’ only up to 5 records.

```
[12]: %%sql
select *
from SPACEXTBL
where LAUNCH_SITE like "CCA%"
limit 5;

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

[12]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- By using SUM function and WHERE clause, the task of calculating total payload mass with querying cab be done properly.

```
[13]: %%sql
select Customer, sum(PAYLOAD_MASS__KG_) as Total_NASA_CRS_mass
from SPACEXTBL
where Customer = "NASA (CRS)";

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

[13]:

Customer	Total_NASA_CRS_mass
NASA (CRS)	45596

Average Payload Mass by F9 v1.1

- Use AS clause and AVG function to find average booster version on PAYLOAD_MASS_KG column from SPACEXTBL

```
[14]: %%sql
select Booster_Version, avg(PAYLOAD_MASS__KG_) as avg_Booster_versionF9_v1_1
from SPACEXTBL
where Booster_Version = "F9 v1.1";

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

[14]:

Booster_Version	avg_Booster_versionF9_v1_1
F9 v1.1	2928.4

First Successful Ground Landing Date

- Use the MIN function to find to find first landing outcome on ground pad date.

```
[15]: %%sql
select  Mission_Outcome, min(Date) as Date_First_Succ_Land
from SPACEXTBL
where Landing_Outcome ='Success (ground pad)';

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

[15]:

Mission_Outcome	Date_First_Succ_Land
Success	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Combine WHERE clause with AND operator to find successful booster in drone ship landing in the range 4000 to 6000 kg payload mass.

```
[16]: %%sql
select Booster_Version,Landing_Outcome, PAYLOAD_MASS__KG_
from SPACEXTBL
where (PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000)
      and Landing_Outcome = 'Success (drone ship)';
```

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

[16]:

Booster_Version	Landing_Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

Total Number of Successful and Failure Mission Outcomes

- Use COUNT function with GROUP BY statement

```
[17]: %%sql
select Mission_Outcome, count(Mission_Outcome) as "Total (Success or failure)"
from SPACEXTBL
GROUP BY MISSION_OUTCOME;

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

[17]:

Mission_Outcome	Total (Success or failure)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- Use subquery whit WHERE clause and IN operator to find the multiple values from the clause.

```
[18]: %%sql
select Booster_Version,Landing_Outcome, PAYLOAD_MASS__KG_
from SPACEXTBL
where PAYLOAD_MASS__KG_ in (select max(PAYLOAD_MASS__KG_)
                            from SPACEXTBL);

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

[18]:

Booster_Version	Landing_Outcome	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Success	15600
F9 B5 B1049.4	Success	15600
F9 B5 B1051.3	Success	15600
F9 B5 B1056.4	Failure	15600
F9 B5 B1048.5	Failure	15600
F9 B5 B1051.4	Success	15600
F9 B5 B1049.5	Success	15600
F9 B5 B1060.2	Success	15600

2015 Launch Records

- Use WHERE clause for failed Landing_Outcome specifically constraining the search with string ‘Failure (drone ship)’, according to their booster versions, and launch site names with the operator AND for year 2015.

[19]:
```
%%sql
SELECT  Date, Booster_Version, Launch_Site, Landing_Outcome
FROM SPACEXTBL
where Landing_Outcome= 'Failure (drone ship)' and Date <= "2015-12-31";
```

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

[19]:

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

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

- USE clause WHERE and Operators OR, AND accordingly to strings in search. Later use GROUP BY statement and ORDER BY, DESC keywords.

```
[20]: %%sql
select Landing_Outcome, count(Landing_Outcome) as "Total Count"
from SPACEXTBL
where Landing_Outcome = "Failure (drone ship)" or Landing_Outcome = "Success (ground pad)" and
Date between "2010-06-04" and "2017-03-20"
GROUP BY Landing_Outcome
order by Landing_Outcome desc;
```

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

[20]:

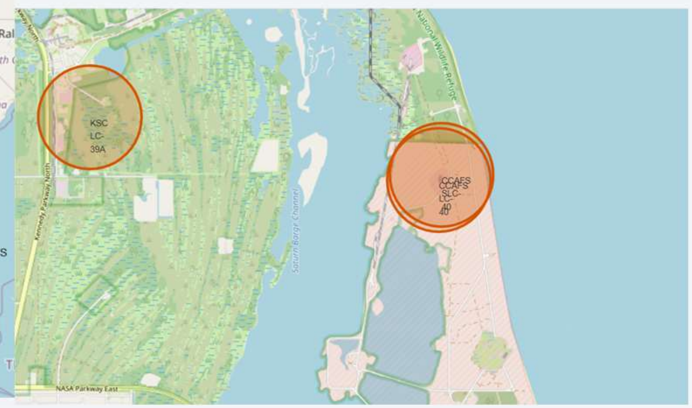
Landing_Outcome	Total Count
Success (ground pad)	3
Failure (drone ship)	5

Section 3

Launch Sites Proximities Analysis

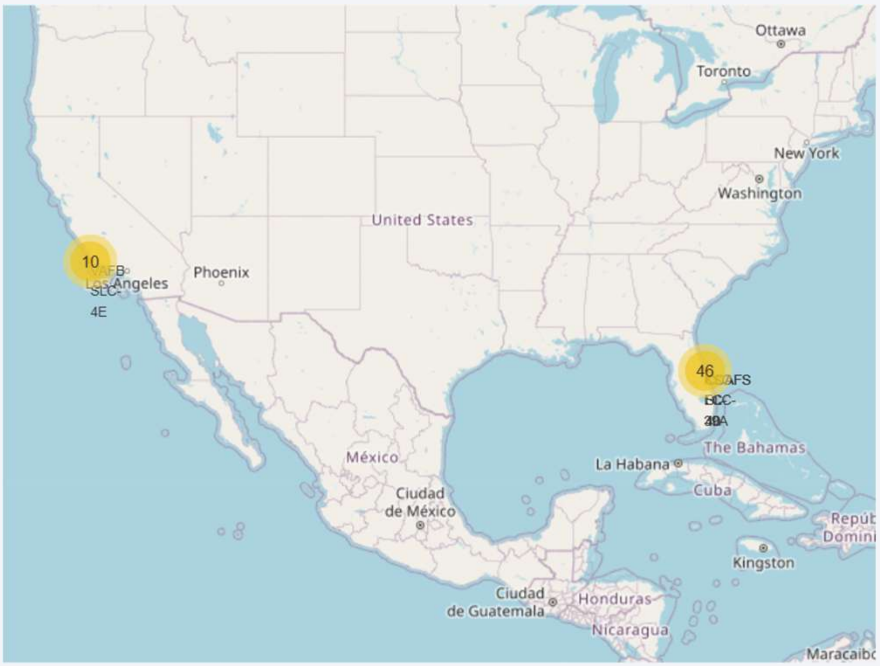
Launch Sites

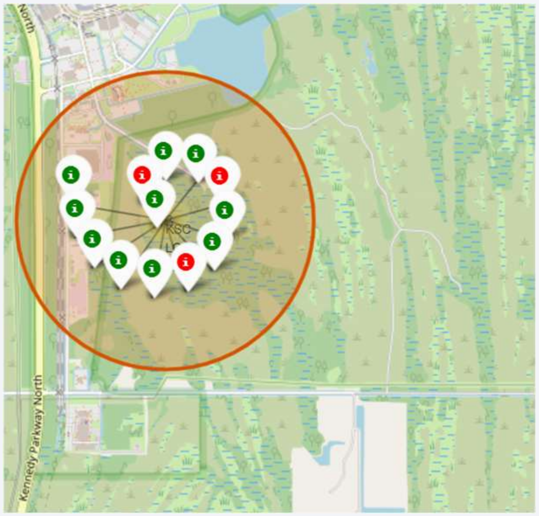
- Launch sites are located near coastlines, relatively close latitudes and similar climate regions.



Success/failure Launch per site

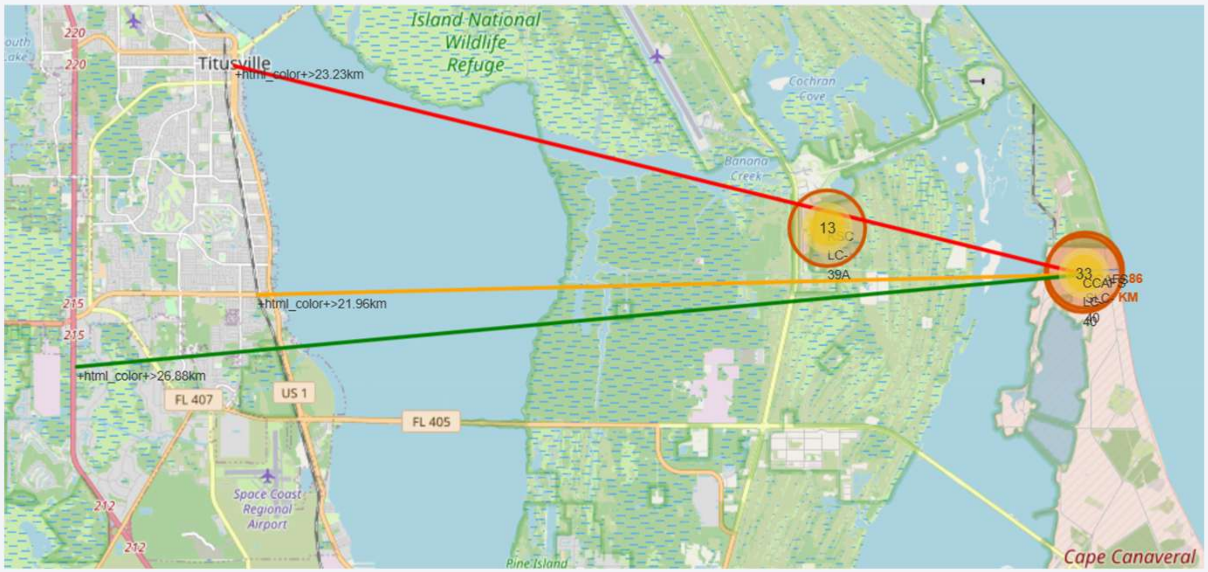
- Using clustering of markers, it can be showcased the successful(green) and failed attempts(red) per site





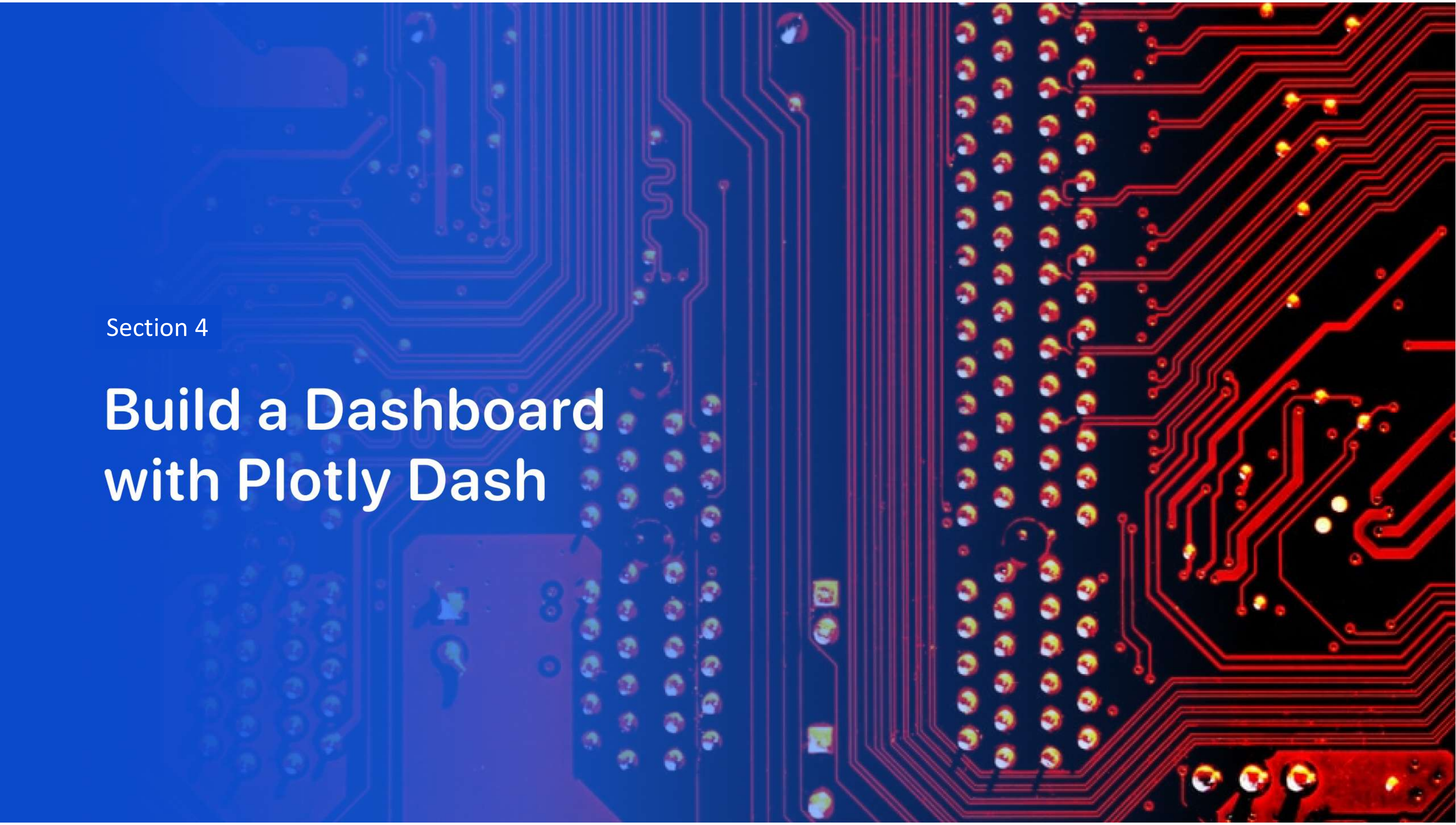
Proximities to the Launch Sites

- Establishing safety distance to cities, proper resupply routes and main access to roads is important in developing an advanced aerospace facility.



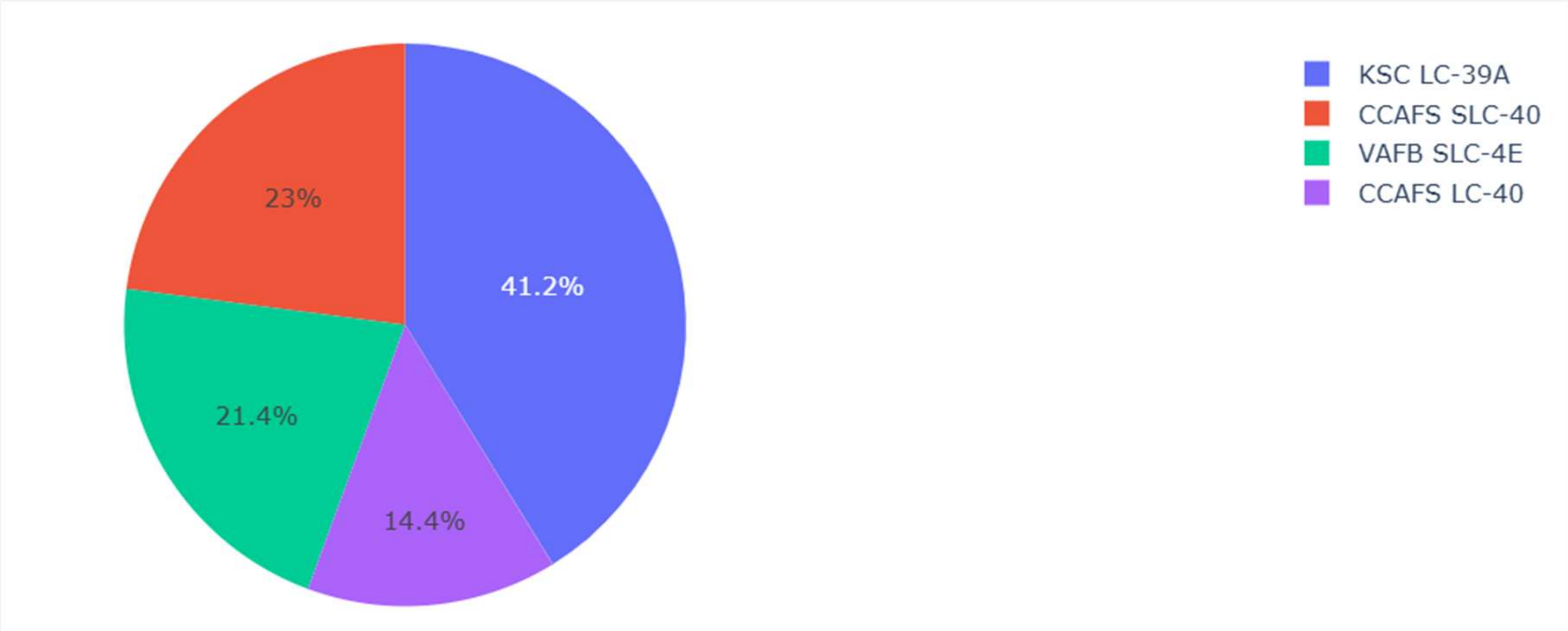
Section 4

Build a Dashboard with Plotly Dash



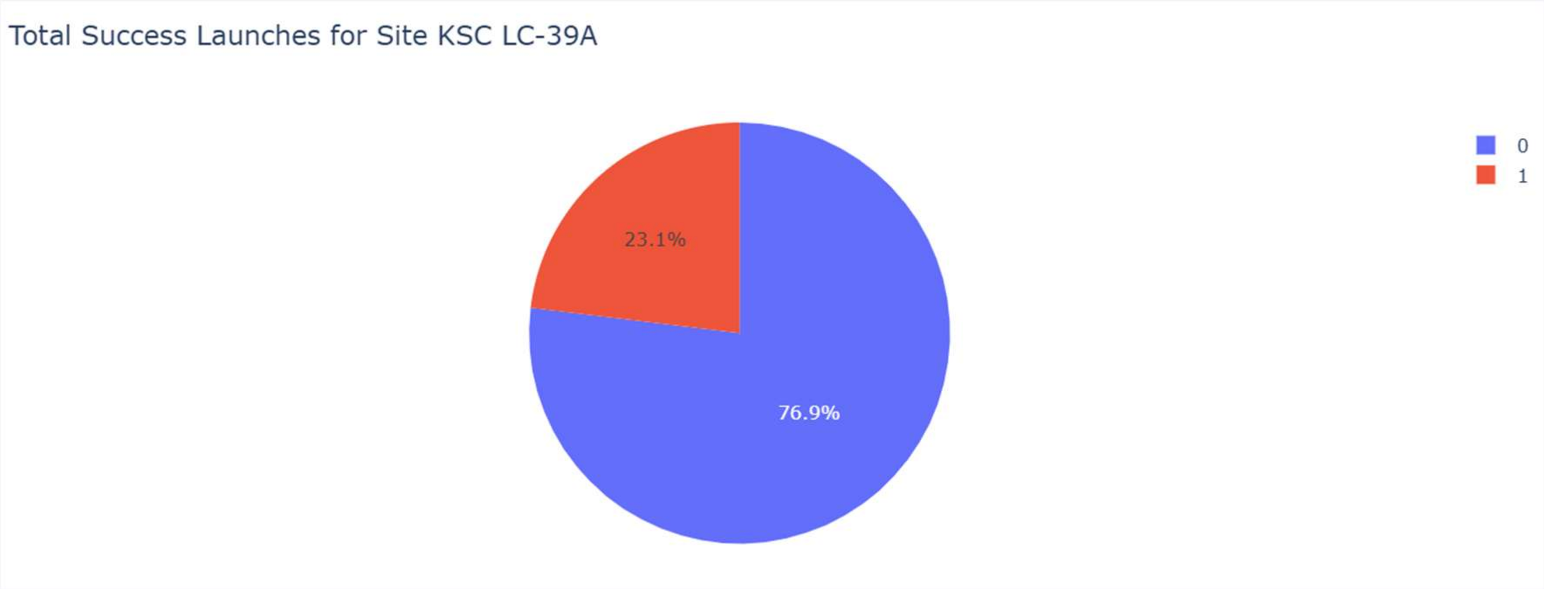
Total success launches by site

- KSC LC-39A proportion in pie chart



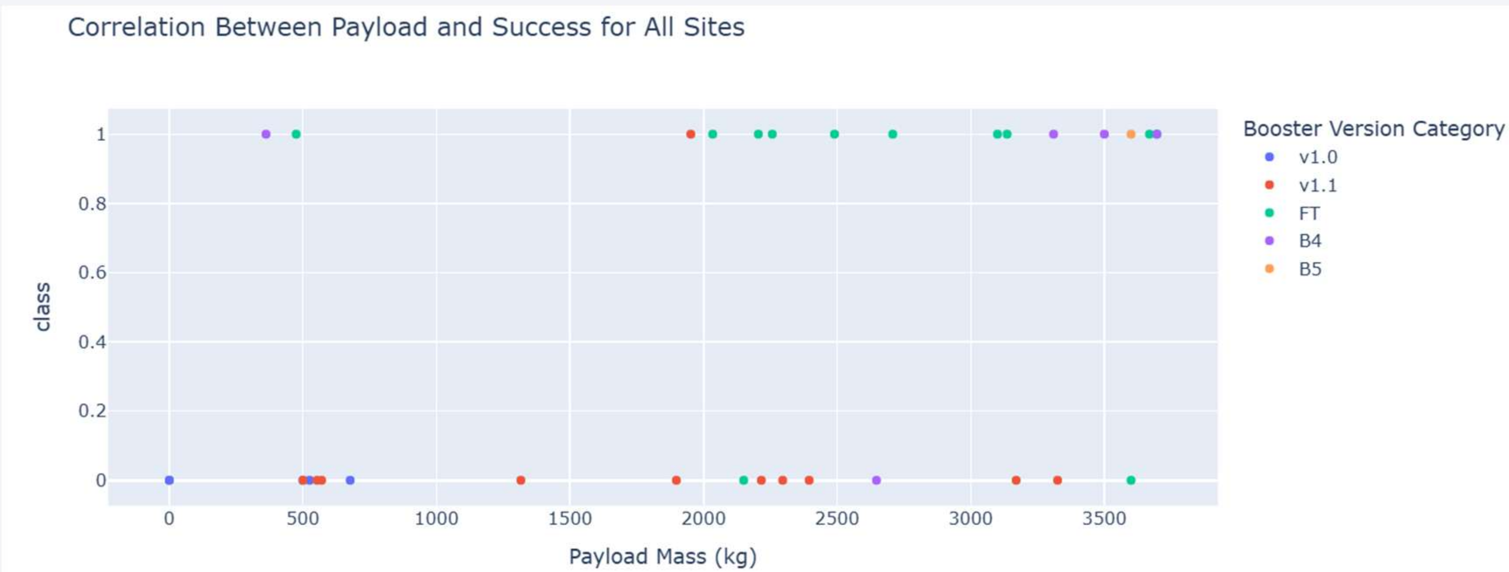
Highest number of success launches site

- Success rate in KSC LC-39A in pie chart



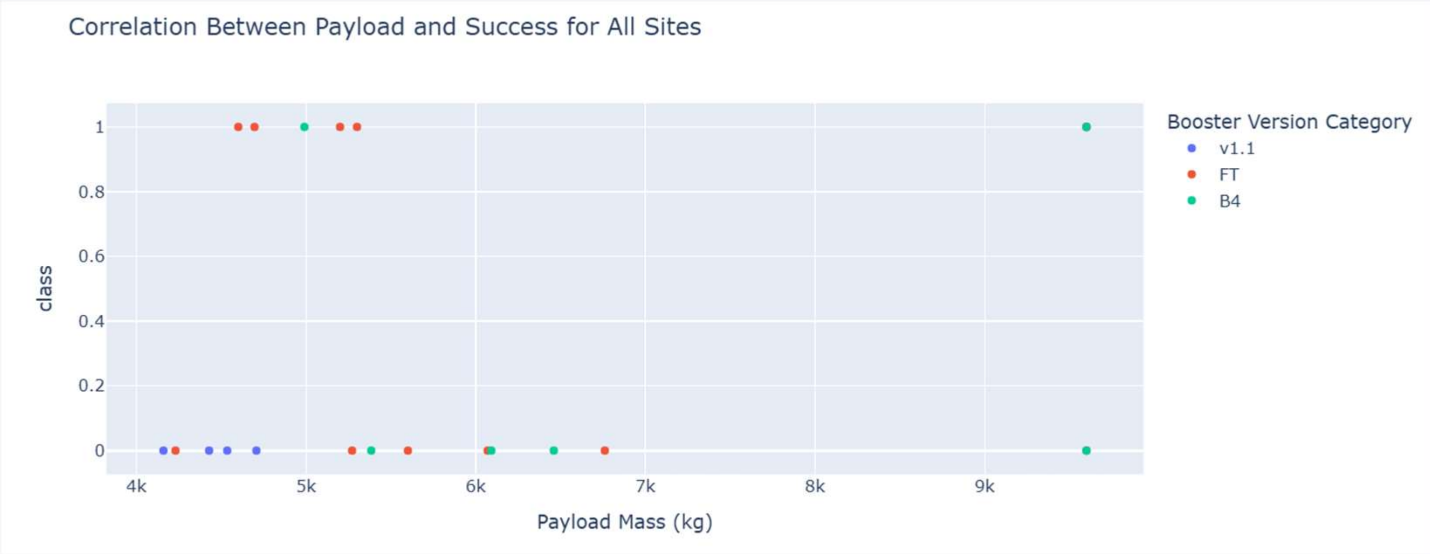
Payload Mass vs Launch Outcome

- FT Booster version are more consistent in the range of 0 to 4000 kg



Payload Mass vs Launch Outcome

- FT and B4 Booster version can perform in the range of 4000 to 10000 kg

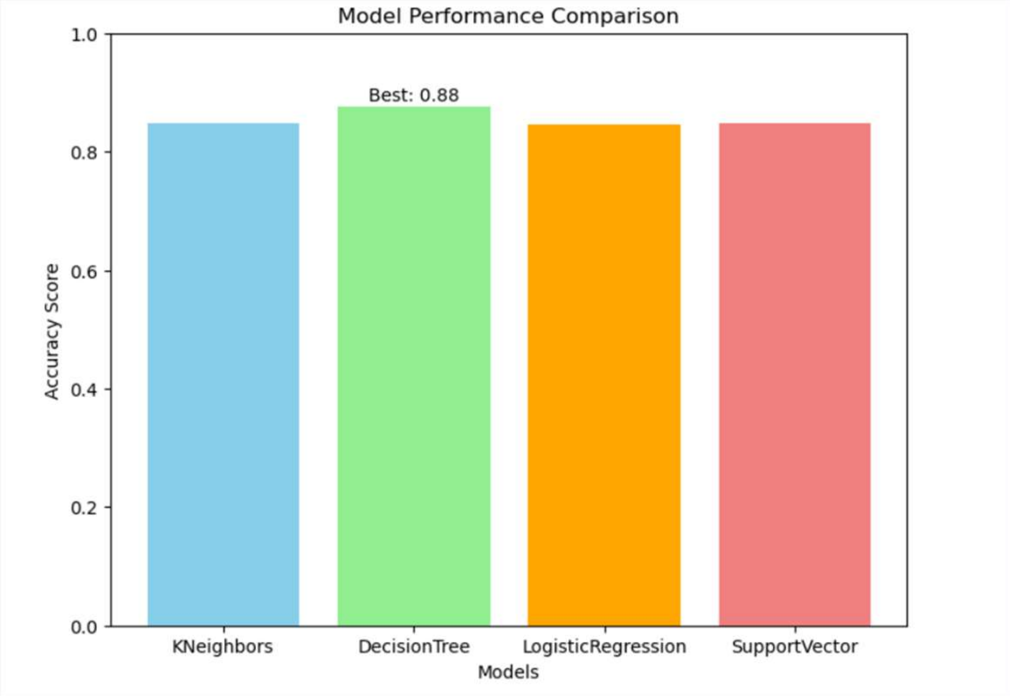


Section 5

Predictive Analysis (Classification)

Classification Accuracy

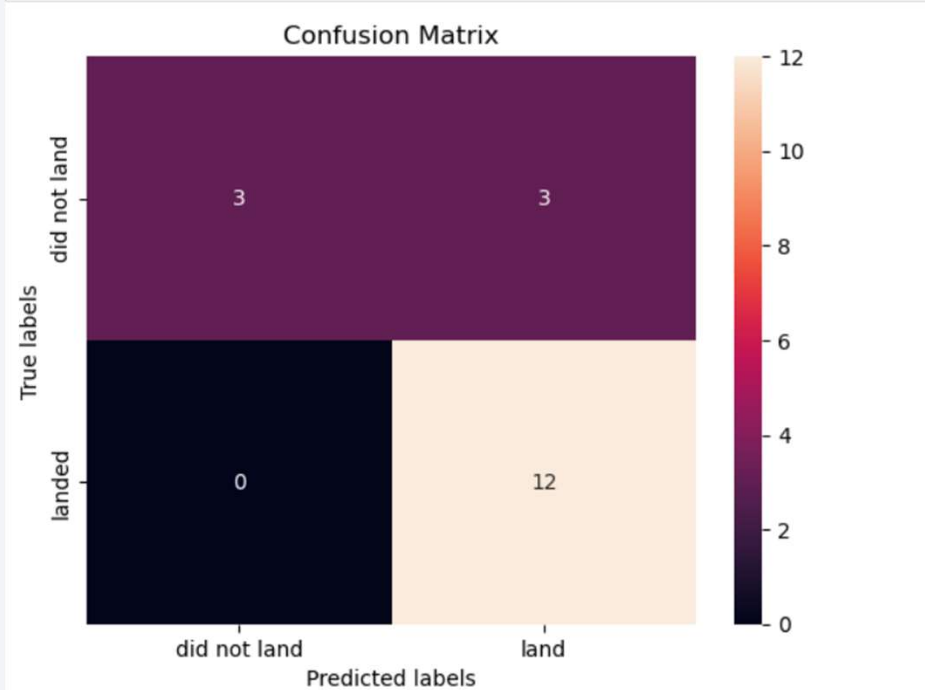
- Models showcase 83.33 % accuracy with score method. With tuned hyperparameters, decision tree performed higher



	LogReg	SVM	Tree	KNN
Jaccard_Score	0.800000	0.800000	0.800000	0.800000
F1_Score	0.888889	0.888889	0.888889	0.888889
Accuracy	0.833333	0.833333	0.833333	0.833333

Confusion Matrix of decision tree model

```
tree_yhat = svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,tree_yhat)
```



•The model demonstrates strong performance in identifying successful landings, as evidenced by 12 accurate predictions with no false negatives (0 cases where the model missed a landing).

•However, the model struggles somewhat with the "did not land" class, producing 3 false positives. These could indicate areas where the model might need refinement to improve precision for unsuccessful outcomes.

Conclusions

- Launch sites are strategically located near railways and highways to facilitate transportation of equipment and staff.
- Proximity to coastlines minimizes risks associated with failed launches.
- Launch sites maintain a safe distance from cities to protect populated areas from accidents.
- The layout of these proximities showcases thoughtful planning, balancing safety, logistics, and operational efficiency for the launch sites.

Conclusions

- The scatterplot tells a compelling story of continuous improvement in launch reliability over time, particularly for high-utilization sites like CCAFS SLC 40 and KSC LC 39A. Early missions across all launch sites exhibited a mix of successes and failures, but as the flight numbers increased, a clear shift toward more successful launches is observed.
- The differences in launch frequency and success patterns also highlight the strategic use of each launch site: a) CCAFS SLC 40 serves as a backbone for frequent missions; b) VAFB SLC 4E caters to specific or less frequent launches and; c) KSC LC 39A emerges as a balanced site with growing reliability.

Conclusions

- LEO has high success rates with a wide range of payload masses, reflecting its versatility and reliability for diverse missions. Meanwhile, ISS with its strong reliability for lighter payloads, highlight precision for space station missions.PO has Moderate reliability, particularly for smaller payloads, but challenges arise with mid-sized payloads. Moreover, GTO consistent technical challenges with heavy payloads, resulting in higher failure rates.
- While LEO emerges as the most reliable target, there is a clear need to address the variability in success rates across ISS, PO, and GTO missions. Focused technological advancements and operational improvements could significantly enhance reliability for these orbits.
- The Decision Tree model performs particularly well on predicting successful landings, making it reliable in recognizing "landed" outcomes. Improvements could be targeted towards reducing false positives (misclassifications of "did not land"), as this might impact the model's precision in real-world scenarios.

Appendix

```
#Create barplot for best prediction model
import matplotlib.pyplot as plt
# Model performance data
models = {'KNeighbors': knn_cv.best_score_,
          'DecisionTree': tree_cv.best_score_,
          'LogisticRegression': logreg_cv.best_score_,
          'SupportVector': svm_cv.best_score_}
# Create the bar plot
plt.figure(figsize=(8, 6))
plt.bar(models.keys(), models.values(), color=['skyblue', 'lightgreen', 'orange', 'lightcoral'])
# Add labels and title
plt.xlabel('Models')
plt.ylabel('Accuracy Score')
plt.title('Model Performance Comparison')
plt.ylim(0, 1)  # Set y-axis range from 0 to 1 for clarity
# Highlight the best model
bestalgorithm = max(models, key=models.get)
plt.text(bestalgorithm, models[bestalgorithm] + 0.01, f'Best: {models[bestalgorithm]:.2f}',
         ha='center', fontsize=10, color='black')

# Display the plot
plt.show()
```

Appendix

- Visit my Github repository https://github.com/isaachsgel/IBM-Data-Science-Course for dataset, database, snippets and the API in Jupyter DASH

Thank you!