IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<David Orlando Romero Soto>
<30-08-2024>

Outline

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

Executive Summary

- **Summary of methodologies**
  - Data collection via API and Web Scrapping
  - Data Wrangling and EDA
  - Interactive visualizations and dashboards
  - Predictive Analysis

- **Summary of all results**
  - Data analysis with visualization
  - Data modeling for predictive analysis

Introduction

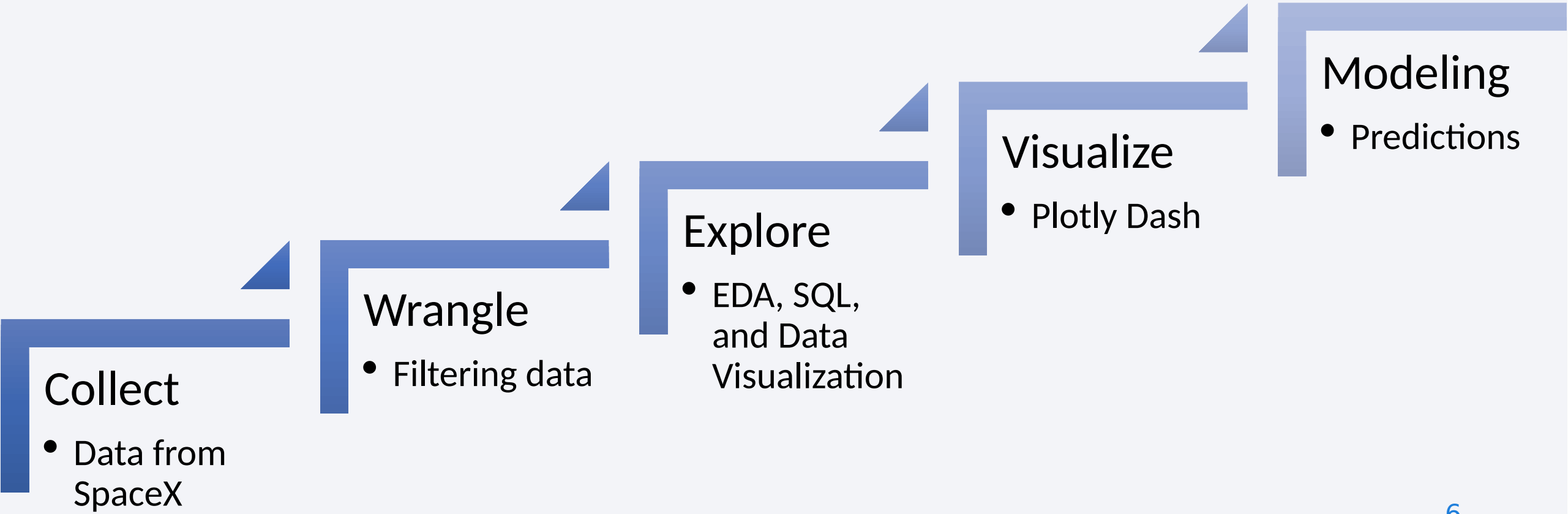
- **Project background and context**
- In this project, it describes the work of predicting the first stage of the Falcon 9 rocket will land successfully. SpaceX advertises the launches of its Falcon 9 rocket on its website at a cost of $62 million, while other suppliers have costs in excess of $165 million each. The significant reduction in cost is due to SpaceX's ability to reuse the first stage of the rocket. Therefore, by determining the probability that the first stage will land, the total cost of a launch can be estimated.

- **Problems you want to find answers**
  - What are the factors that determine a successful landing?
  - What are the rate of successful landings over time?
  - What is the best predictive mode?

Section 1

Methodology

Methodology



Data Collection

The data was collected using a variety of methods:

- Data collection was performed using an API.
- The content of the response was decoded as a Json and converted to a pandas DataFrame.
- The data was cleaned, checking for missing values.
- Was performed a web scrapping in Wikipedia to get the Falcon 9 Launch Records.

Data Collection – SpaceX API

1

Obtain Data from spaceX URL by API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"

response = requests.get(spacex_url)
```

2

Decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

3

Delete missing values

```
# Calculate the mean value of PayloadMass column
#Mean_PayloadMass = data_falcon9.PayloadMass.mean()
Mean_PayloadMass = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
#data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].replace(np.nan, Mean_PayloadMass)
data_falcon9.loc[:, 'PayloadMass'] = data_falcon9['PayloadMass'].fillna(Mean_PayloadMass)
```

Dataframe

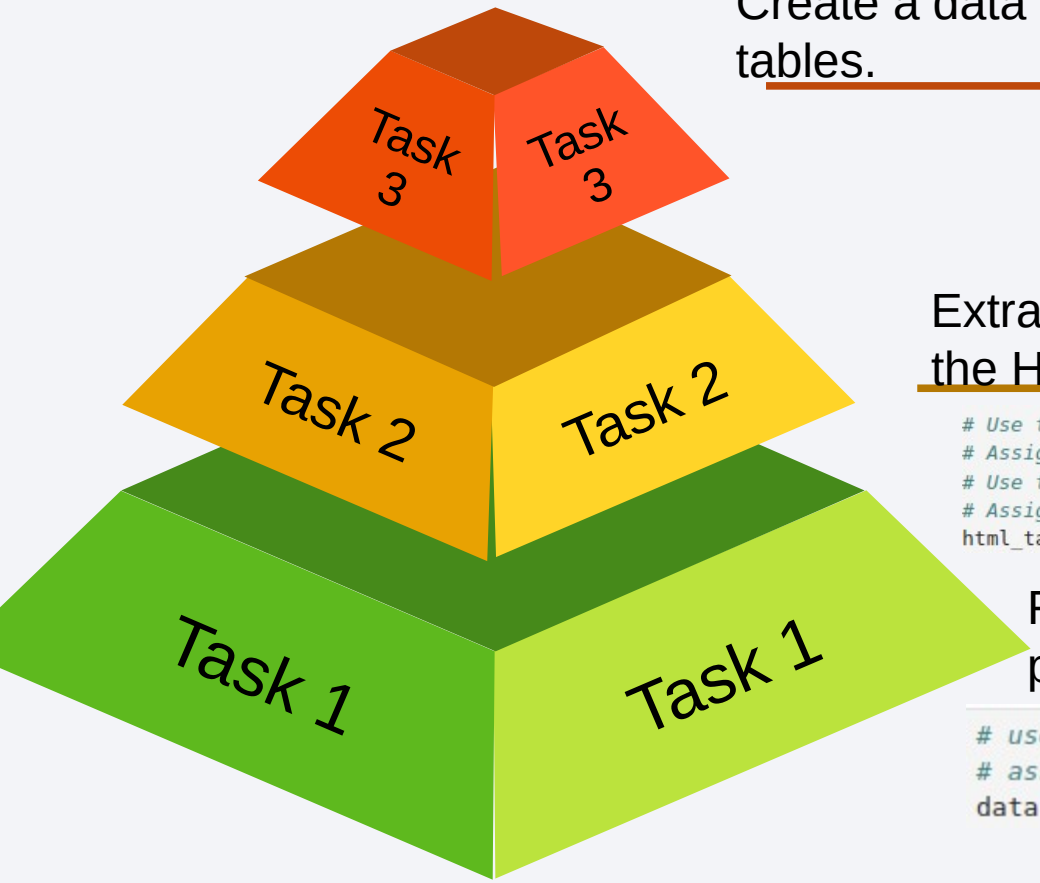
4

[63]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False
7	4	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False
8	5	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False
...	...	...	...	...	...	...	...	...	...

https://github.com/DavOrland/IBMCourse/blob/main/jupyter-labs-spacex-data-collection-api.ipyn

Data Collection - Scraping



Create a data frame by parsing the launch HTML tables.

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

Extract all column/variable names from the HTML table header

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

Request the Falcon9 Launch Wiki page from its URL

```
# use requests.get() method with the provided static_url
# assign the response to a object
data  = requests.get(static_url).text
```

https://github.com/DavOrland/IBMCourse/blob/main/jupyter-labs-webscraping.ipynb

Data Wrangling

TASK 1: Calculate the number of launches on each site

```
# Apply value_counts() on column LaunchSite
df.LaunchSite.value_counts()
```

TASK 2: Calculate the number and occurrence of each orbit

```
# Apply value_counts on Orbit column
df.Orbit.value_counts()
```

TASK 3: Calculate the number and occurrence of mission outcome of the orbits

```
# landing_outcomes = values on Outcome column
landing_outcomes = df.Outcome.value_counts()
landing_outcomes
```

TASK 4: Create a landing outcome label from Outcome column

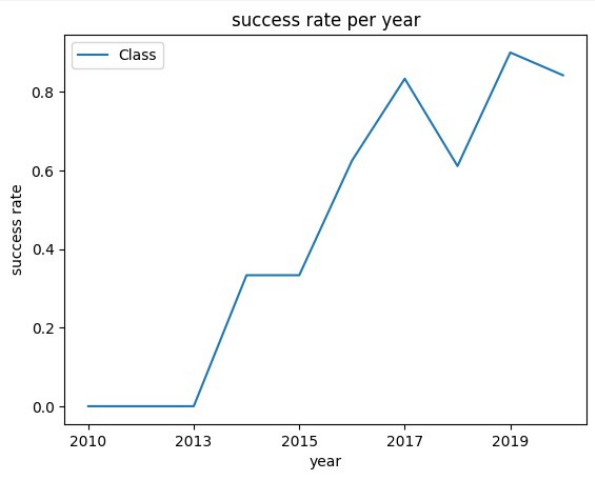
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for key,value in df["Outcome"].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

https://github.com/DavOrland/IBMCourse/blob/main/labs-jupyter-spacex-Data%20wrangling.ipyn

EDA with Data Visualization

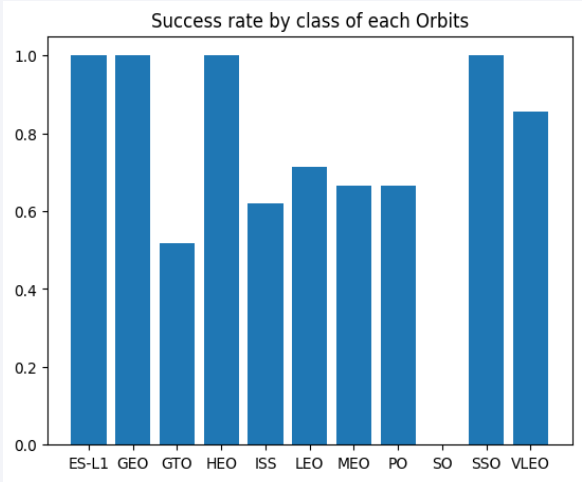
Line Graph

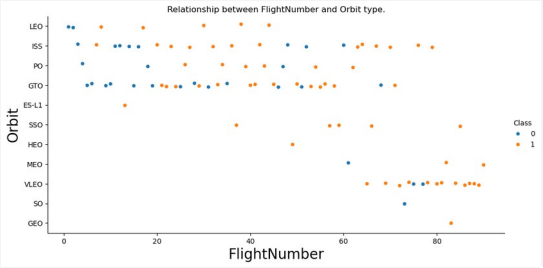
To show trends over time. Useful for representing sequential or temporal data.

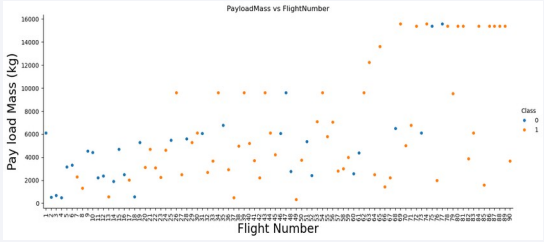


Bar Graph

For comparing different categories or groups. Efficient to show the quantity or frequency of different categories.

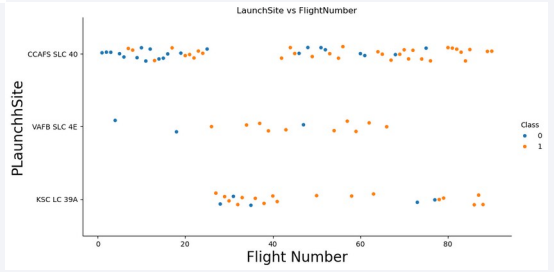


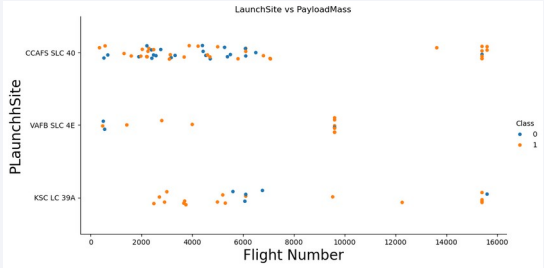


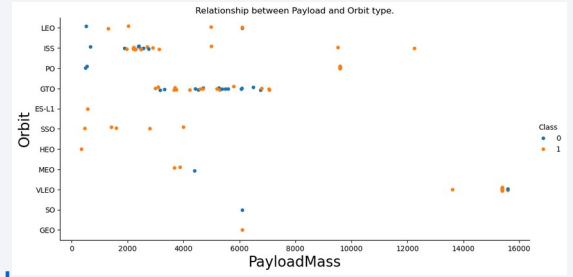


Scatter Graph

Used to see the relationship between two numerical variables. It allows to observe patterns, trends or the correlation between variables.







https://github.com/DavOrland/IBMCourse/blob/main/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

Tasks https://github.com/DavOrland/IBMCourse/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

- Setup SQL in Python
- Load Database
- Display the total payload mass carried by boosters launched by NASA (CRS)
- 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.

Build an Interactive Map with Folium

Launch Sites

- Blue circle at NASA Johnson Space Center's
- Red circles at all launch sites coordinates

Launch Outcomes

- Successful (green)
- Unsuccessful (red)

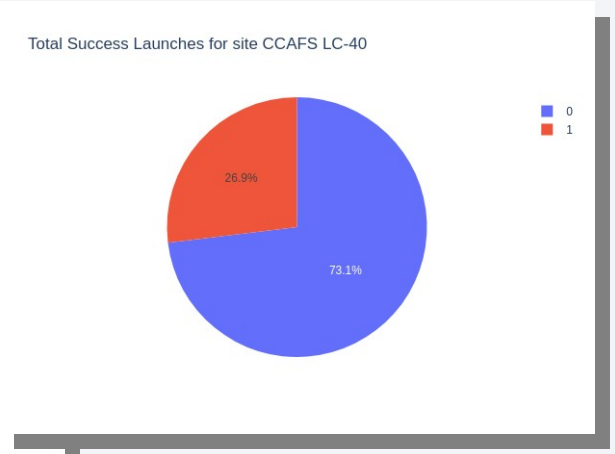
The objects and markers on this Folium map are designed to provide a comprehensive, interactive visualization of space launch sites and their associated data. By representing the locations, outcomes, and proximities visually, users can gain insights into the success rates, geographical challenges, and logistical considerations of various launch sites. The use of different colors and markers enhances the map's readability and effectiveness as a tool for analysis and communication.

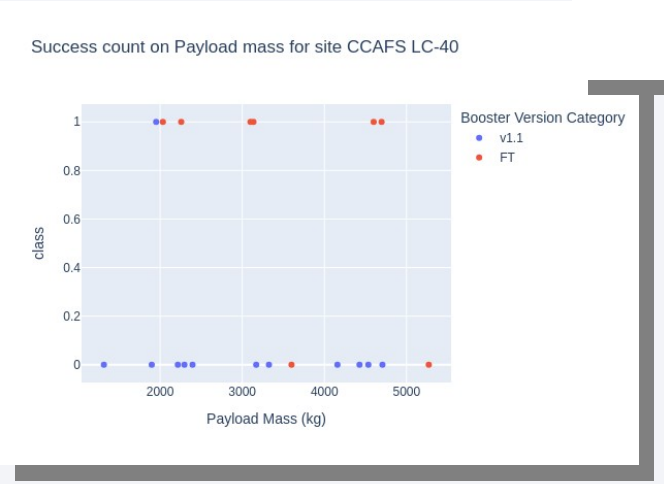
https://github.com/DavOrland/IBMCourse/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

Interactions

- **Dropdown List with Launch Sites**
- **Pie Chart of Successful Launches**
- **Slider of Payload Mass Range**
- **Scatter Chart Showing Payload Mass vs. Success Rate by Booster Version**





The dashboard's plots provide a comprehensive and customizable experience for analyzing space launch data. By filter and visualize data according to specific criteria—such as launch site, payload mass, and booster version—the dashboard facilitates a deeper understanding of the factors influencing launch success. This interactivity can tailor their analysis to specific needs, making the dashboard a powerful tool for both broad and detailed exploration of the data.

https://github.com/DavOrland/IBMCourse/tree/main/DashApp

Predictive Analysis (Classification)

Building the Classification Model

1 - Data Preparation

- Creating NumPy Array from the Class Column
- Standardizing the Data
- Splitting the Data

2 - Model Training and Parameter Optimization

- Creating a GridSearchCV Object
- Applying GridSearchCV to Different Algorithms:
  - Logistic Regression (LogisticRegression()): A linear model used for binary classification.
  - Support Vector Machine (SVC()): A model that finds the optimal hyperplane for separating classes in a high-dimensional space.
  - Decision Tree (DecisionTreeClassifier()): A non-linear model that splits data into subsets based on feature values.
- GridSearchCV was used to find the optimal parameters for each algorithm by evaluating various combinations during cross-validation.

Model Evaluation and Selection

3 - Evaluating Model Performance:

- Calculating Accuracy
- Assessing the Confusion Matrix

4 - Identifying the Best Model

- Accuracy

Improvement and Best Performing Model

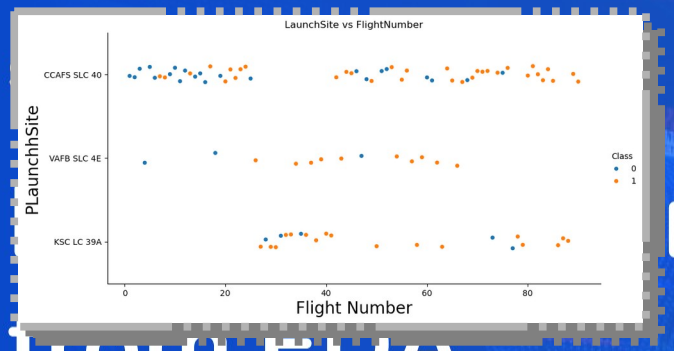
5 - Best Performing Model

After evaluating all models based on Accuracy, the model with the best combination of these metrics was selected as the best-performing classifier.

https://github.com/DavOrland/IBMCourse/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

esults

Exploratory data analysis results



Interactive analytics demo in screenshots

n

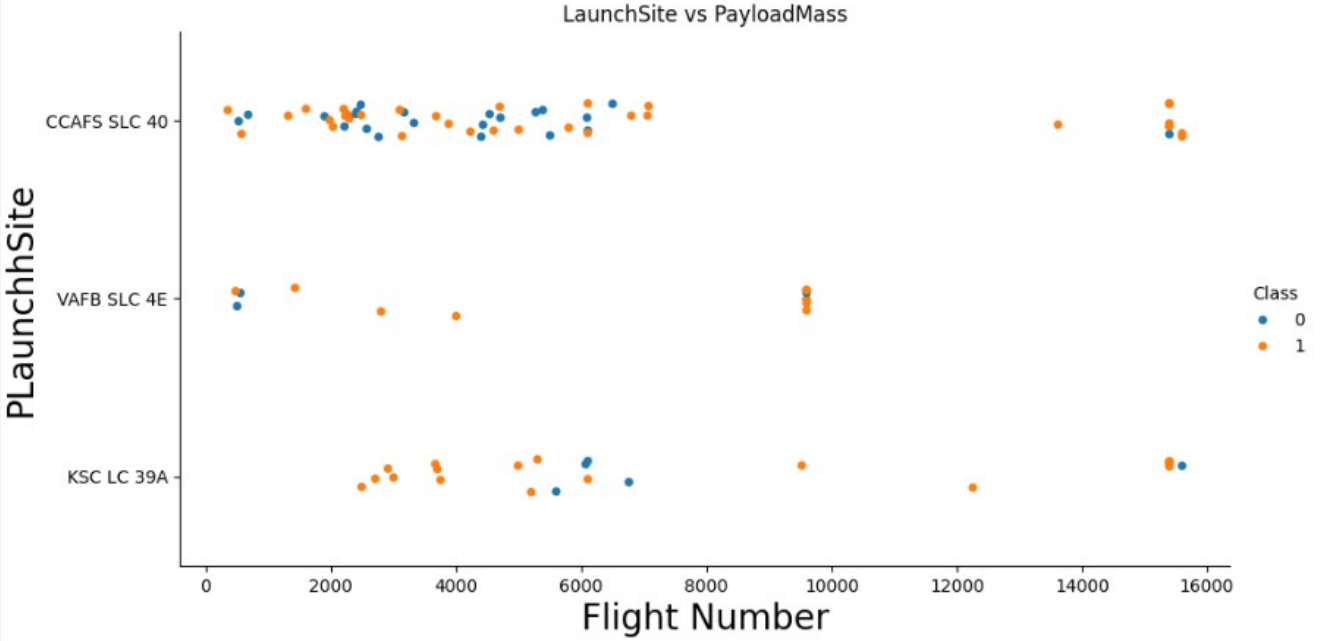
from EDA

Predictive analysis results

Flight Number vs. Launch Site

Flight Success Trends Over Time:
Earlier Flights: Show a higher failure rate (more blue dots).
Later Flights: Display an improved success rate (more orange dots), suggesting effective learning and adaptation over time.
Launch Site Success Rates:
CCAFS SLC 40: Represents about half of the launches with mixed success.
VAFB SLC 4E & KSC LC 39A: These sites show higher success rates, indicating better performance.
Inferred Insight:
Newer Launches: Tend to have higher success rates, likely due to technological and procedural advancements.

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 2)
plt.title('LaunchSite vs PayloadMass')
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("PLaunchhSite",fontsize=20)
plt.show()
```



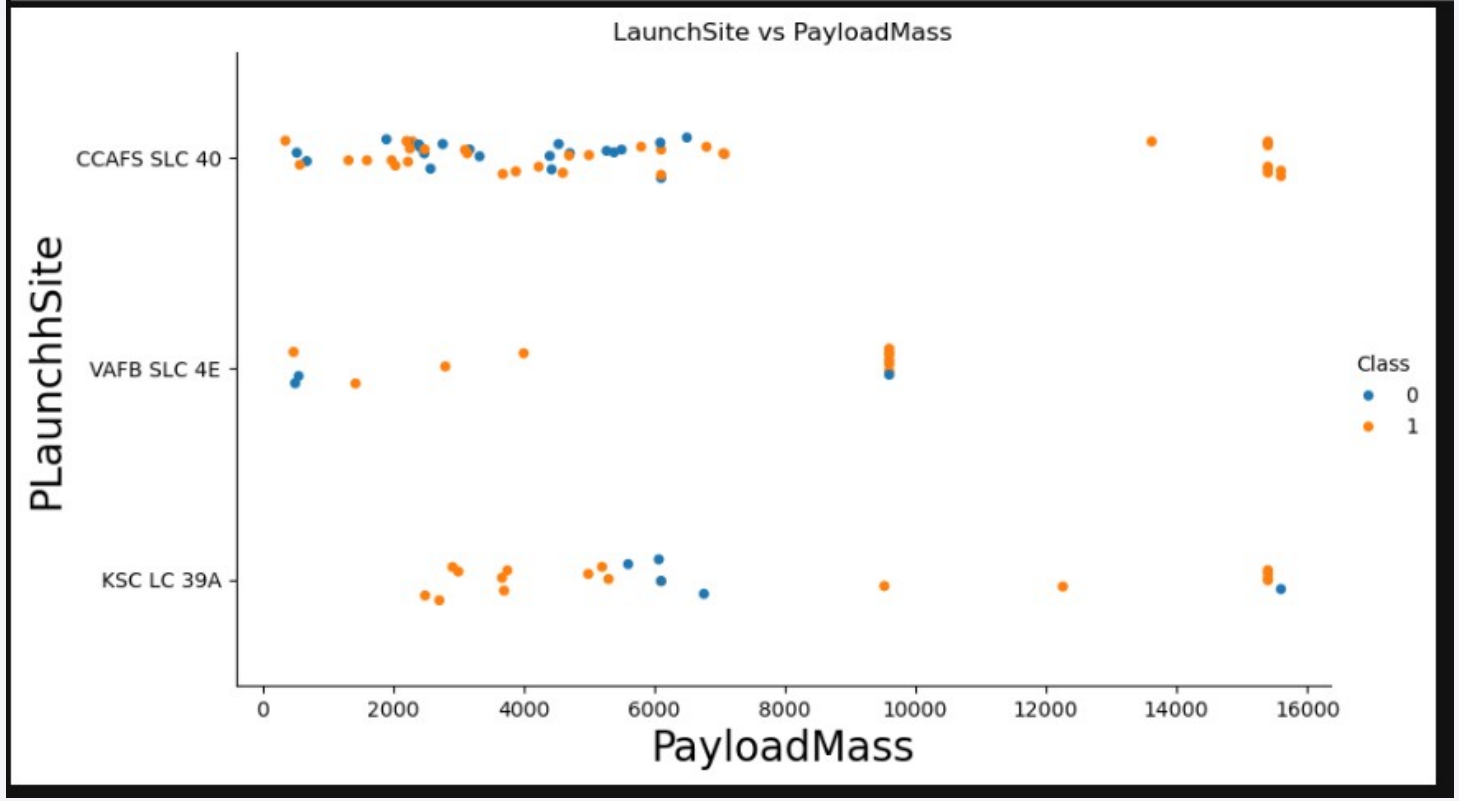
Section 2

Insights drawn from EDA

Payload vs. Launch Site

Payload Mass and Success Rate: Higher payload masses generally correlate with higher success rates, especially for payloads above 7,000 kg.
KSC LC 39A Performance: The KSC LC 39A launch site has a 100% success rate for payloads under 5,500 kg, indicating strong performance for lighter launches.
VAFB SLC 4E Payload Range: VAFB SLC 4E has not launched payloads over 10,000 kg, suggesting either a capacity limitation or specific mission choices.

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 1.8)
plt.title('LaunchSite vs PayloadMass')
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("PLaunchhSite",fontsize=20)
plt.show()
```



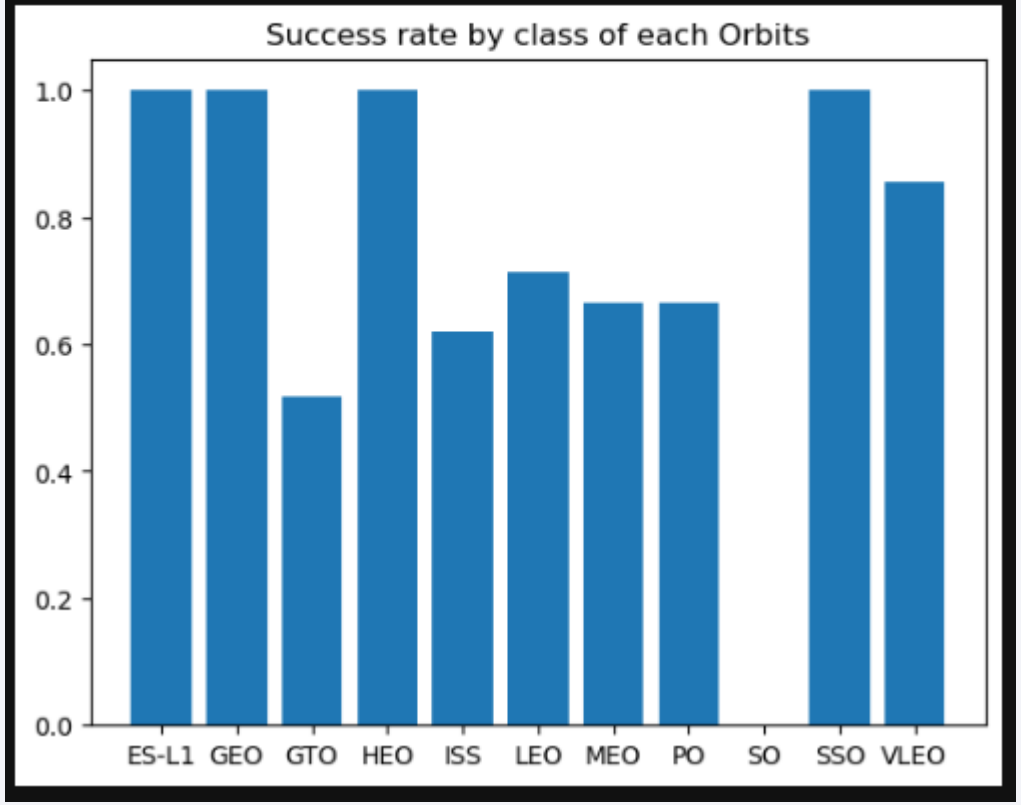
Success Rate vs. Orbit Type

100% Success Rate: The orbits ES-L1, GEO, HEO, and SSO have a perfect success rate, indicating consistent reliability in these orbital classes.

- **50%-80% Success Rate:** Orbits like GTO, ISS, LEO, MEO, and PO have moderate success rates, suggesting variability in mission outcomes depending on the challenges specific to these orbits.

- **0% Success Rate:** The SO orbit has a 0% success rate, indicating significant difficulties or challenges associated with launching missions into this orbit.

```
# HINT use groupby method on Orbit column and get the mean of Class co
orbit = df[['Orbit','Class']].groupby('Orbit').mean()
plt.title('Success rate by class of each Orbits')
plt.bar(orbit.index.values, orbit['Class'])
plt.show()
```

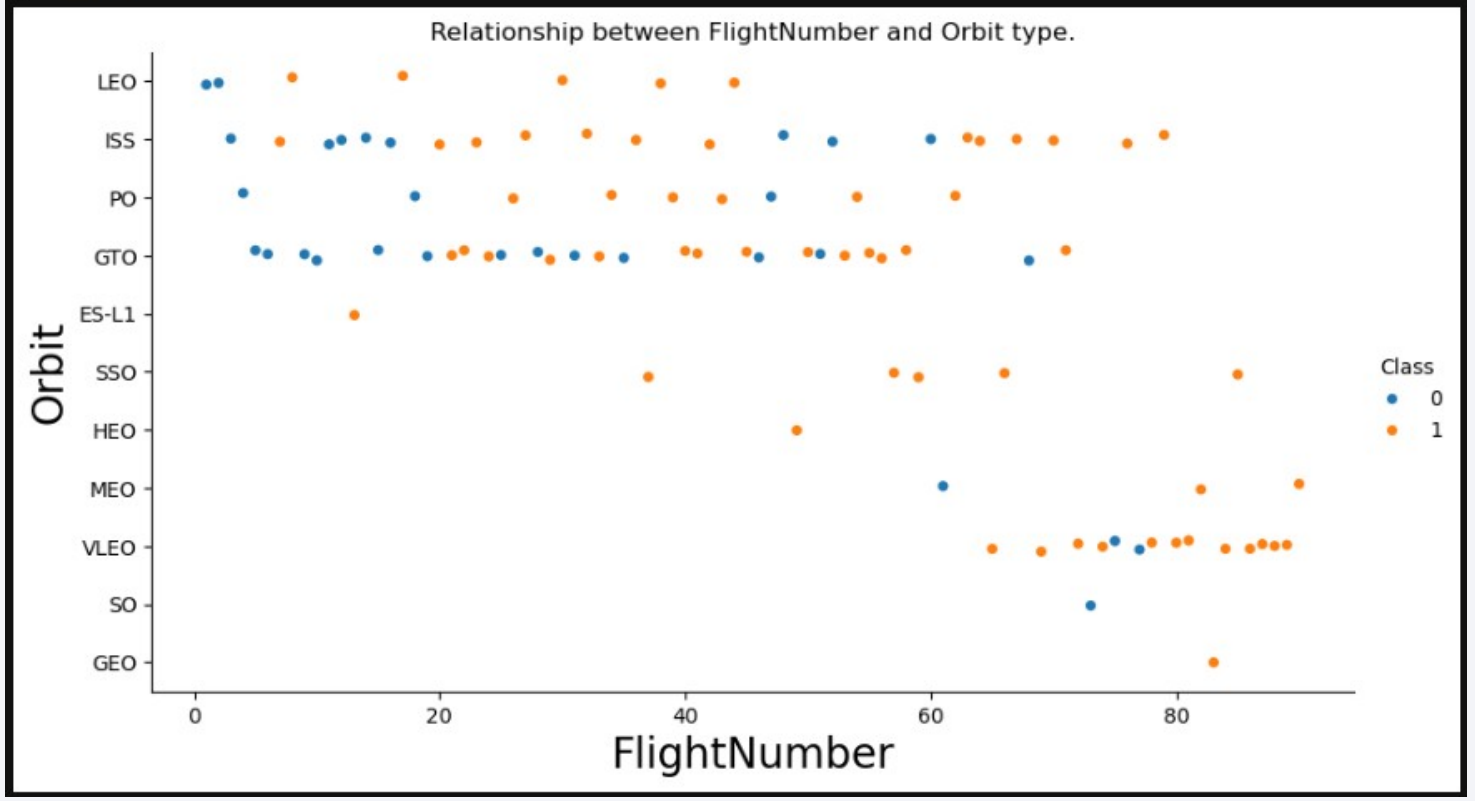


Flight Number vs. Orbit Type

Exploratory Data Analysis

- 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

```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 1.8)
plt.title('Relationship between FlightNumber and Orbit type.')
plt.xlabel("FlightNumber",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```

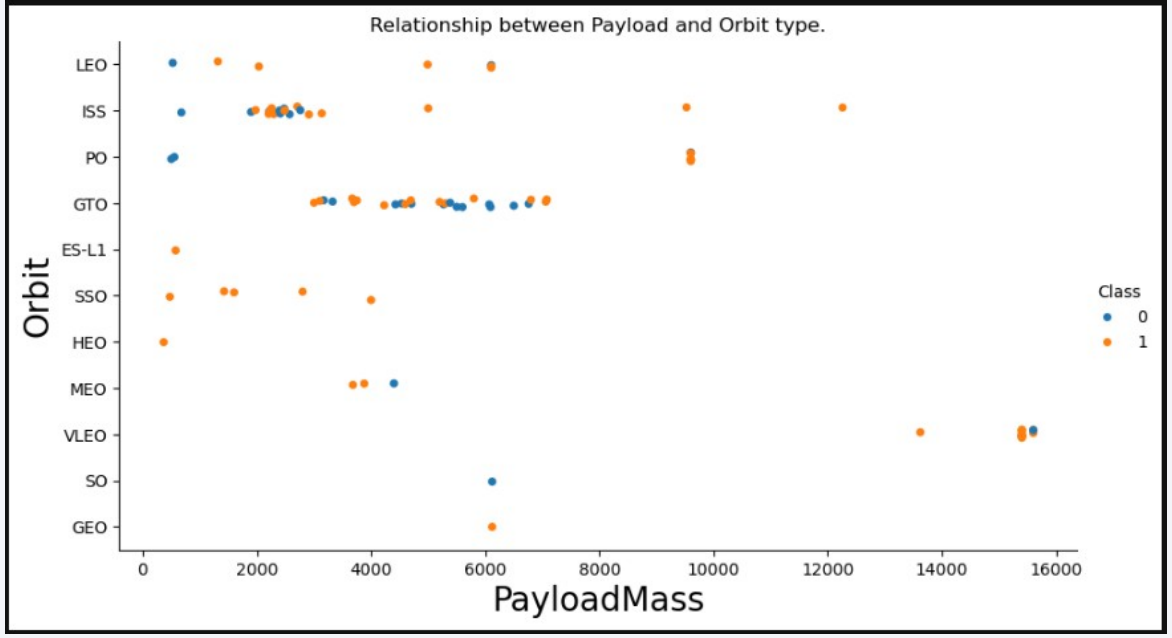


Payload vs. Orbit Type

Heavy Payloads and Preferred Orbits:
Heavier payloads are primarily associated with LEO (Low Earth Orbit), ISS (International Space Station), and PO (Polar Orbit). These orbits seem more successful for missions involving heavy payloads.

GTO Orbit:
The GTO (Geostationary Transfer Orbit) shows mixed success with heavier payloads, indicating variability in outcomes.

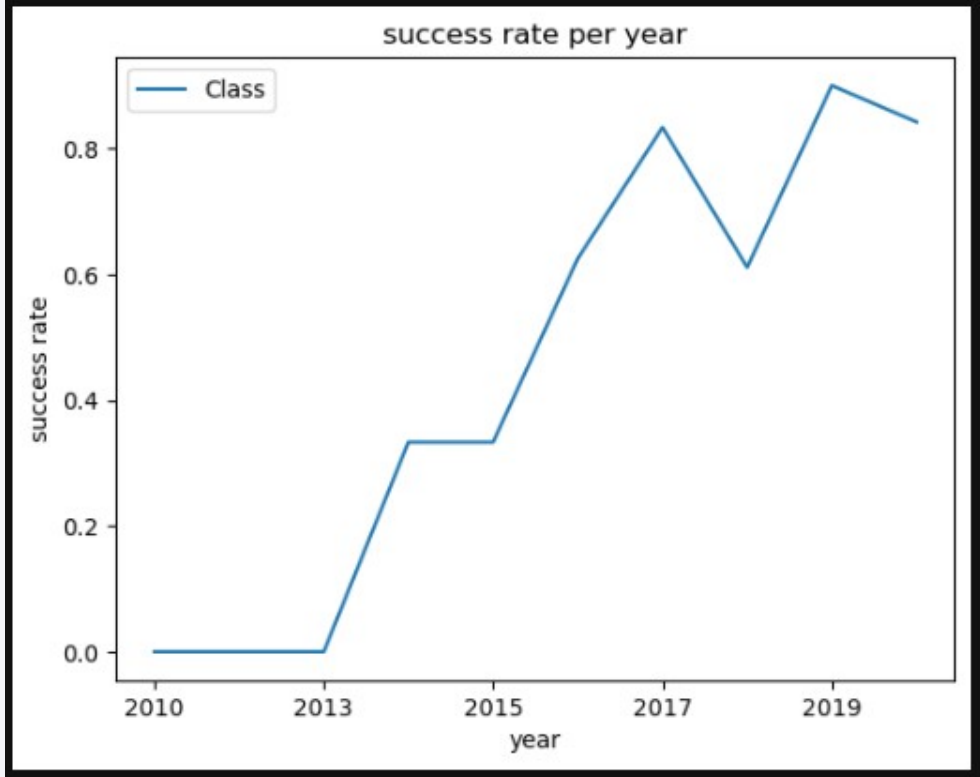
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 1.8)
plt.title('Relationship between Payload and Orbit type.')
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Launch Success Yearly Trend

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
df['year'] = Extract_year('')
df[['Class','year']].groupby('year').mean().plot()
plt.title('success rate per year')
plt.ylabel('success rate')
plt.xlabel('year')
```

```
Text(0.5, 0, 'year')
```



You can observe that the success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.

All Launch Site Names

Distinct function is used to return unique values from a specified column or set of columns. DISTINCT in a query filters out duplicate records from the result set, ensuring that each combination of selected values appears only once… In this case, the Launch Sites

Display the names of the unique launch sites in the space mission

```
%sql select distinct Launch_Site from SPACEXTBL
```

```
 * 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 SPACEXTBL where Launch_Site like 'CCA%' limit 5
```

```
 * 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

This SQL query retrieves the first 5 records from the SPACEXTBL table where the Launch_Site starts with "CCA".

Total Payload Mass

```
%sql select sum(payload_mass__kg_) from SPACEXTBL WHERE customer = 'NASA (CRS)'
 * sqlite:///my_data1.db
Done.
sum(payload_mass__kg_)
45596
```

This query returns the total payload mass (in kilograms) for all launches conducted by NASA under the Commercial Resupply Services (CRS) program.

Average Payload Mass by F9 v1.1

```
%sql select avg(payload_mass__kg_) from SPACEXTBL WHERE booster_version = 'F9 v1.1'
 * sqlite:///my_data1.db
Done.
avg(payload_mass__kg_)
2928.4
```

This SQL query calculates the average (avg) value of the payload_mass__kg_ column from the SPACEXTBL table, but only for rows where the booster_version is 'F9 v1.1'. In other words, it returns the average payload mass (in kilograms) for all launches that used the 'F9 v1.1' booster version.

First Successful Ground Landing Date

```
%sql select min(DATE) from SPACEXTBL WHERE landing_outcome = 'Success (ground pad)'
 * sqlite:///my_data1.db
Done.
min(DATE)
2015-12-22
```

This SQL query finds the earliest date (the minimum value of the DATE column) from the SPACEXTBL table, but only for rows where the landing_outcome is 'Success (ground pad)'. In other words, it returns the date of the first successful landing on a ground pad.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select booster_version from SPACEXTBL where landing_outcome = 'Success (drone ship)'\
and payload_mass__kg_ between 4000 and 6000
```

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

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

This SQL query retrieves the booster_version from the SPACEXTBL table for all records where the landing_outcome is 'Success (drone ship)' and the payload_mass__kg_ is between 4000 and 6000 kilograms. Essentially, it returns the versions of boosters that successfully landed on a drone ship and carried a payload weighing between 4000 and 6000 kg.

Total Number of Successful and Failure Mission Outcomes

```
%sql select mission_outcome, count(mission_outcome) from SPACEXTBL GROUP BY mission_outcome
 * sqlite:///my_data1.db
Done.
```

Mission_Outcome	count(mission_outcome)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

This query provides a count of occurrences for each type of mission outcome, effectively giving you a summary of how many missions resulted in each specific outcome.

Boosters Carried Maximum Payload

```
%sql select booster_version, payload_mass__kg_ from SPACEXTBL\
where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXTBL)
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

The query returns the booster version(s) and corresponding payload mass for the launch(es) that carried the heaviest payload in the dataset.

2015 Launch Records

```
%sql select booster_version, launch_site from SPACEXTBL where landing_outcome = 'Failure (drone ship)' and substr(DATE,0,5) = '2015'
 * sqlite:///my_data1.db
Done.
```

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

The query returns the booster version and launch site for all 2015 launches that attempted to land on a drone ship but failed.

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

```
%sql select count(landing_outcome), landing_outcome from SPACEXTBL \
where DATE between '2010-06-04' and '2017-03-20' group by landing_outcome\
order by count(landing_outcome) desc
```

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

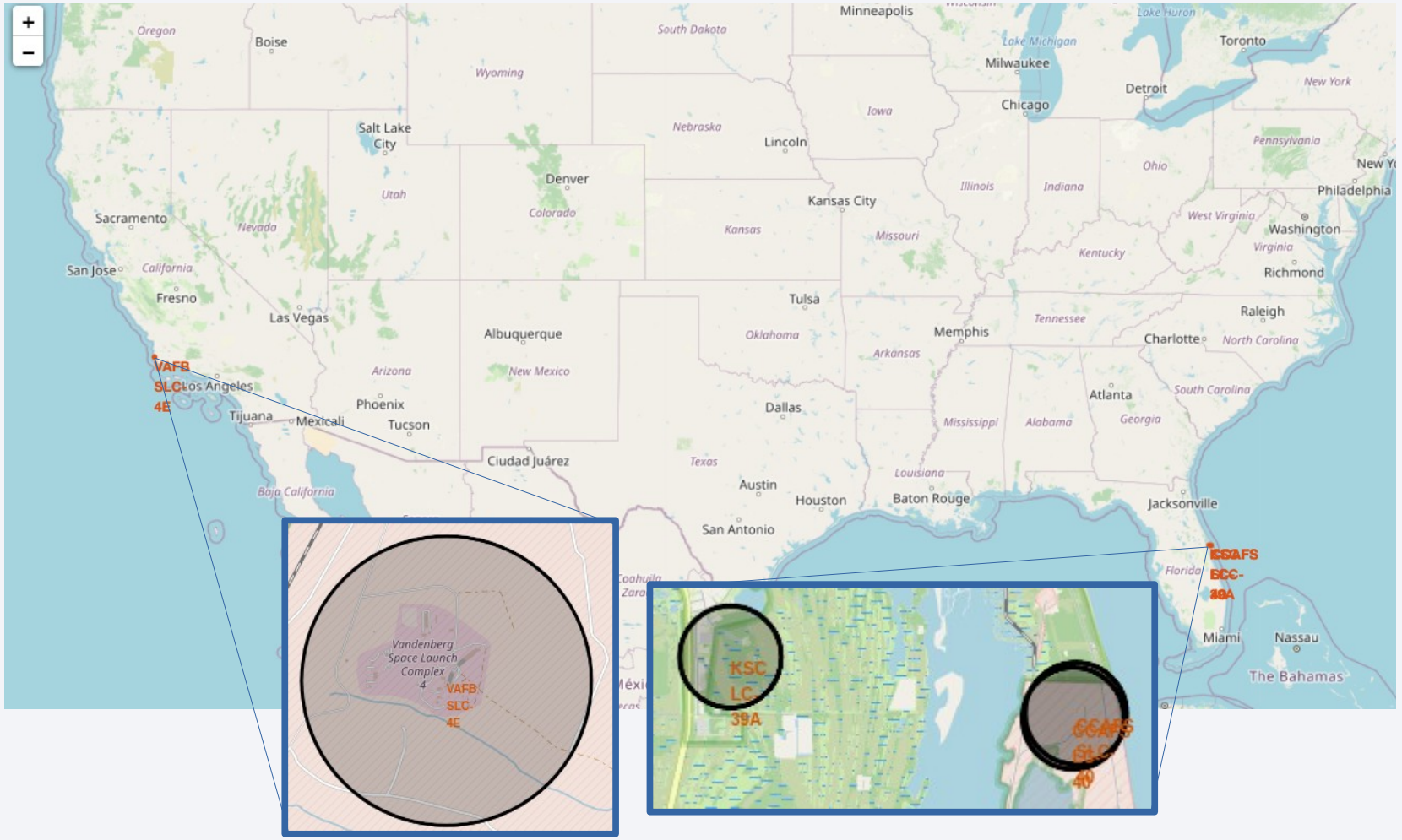
count(landing_outcome)	Landing_Outcome
10	No attempt
5	Success (drone ship)
5	Failure (drone ship)
3	Success (ground pad)
3	Controlled (ocean)
2	Uncontrolled (ocean)
2	Failure (parachute)
1	Precluded (drone ship)

The query returns a list of landing outcomes that occurred between June 4, 2010, and March 20, 2017, along with the count of how many times each outcome occurred. The list is sorted so that the most frequent landing outcome is at the top.

Section 3

Launch Sites Proximities Analysis

Launch Map

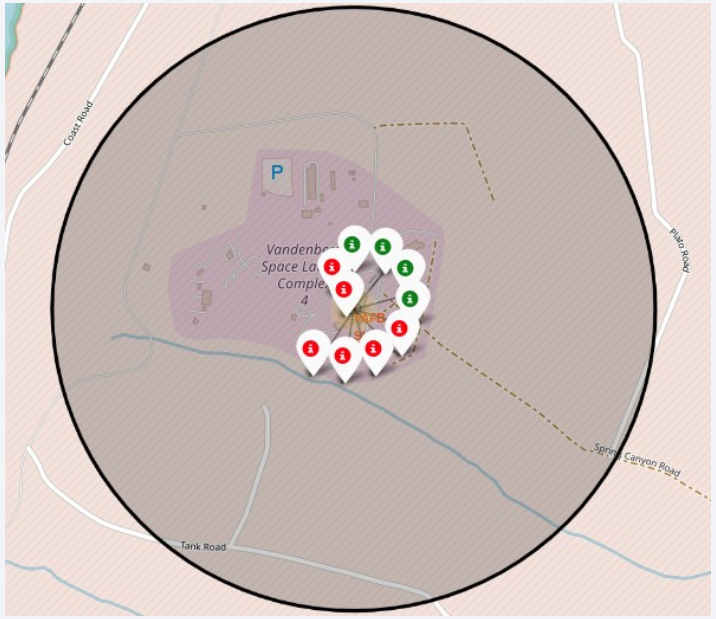


SpaceX Launch sites near Florida and California.

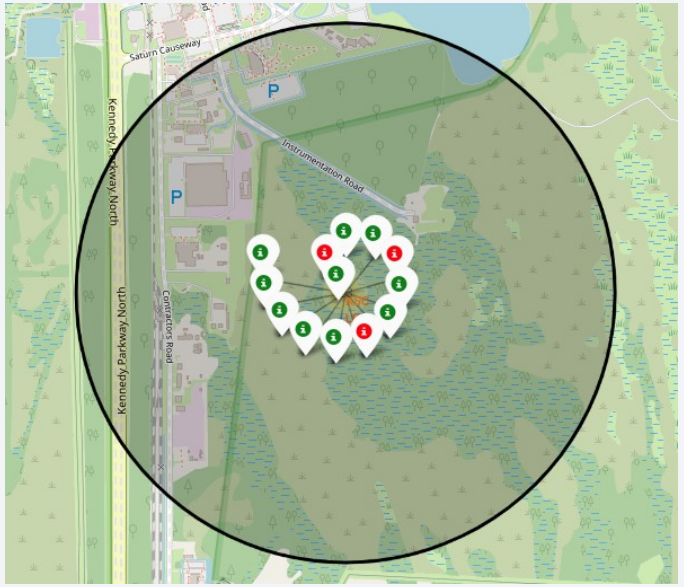
Launch Sites

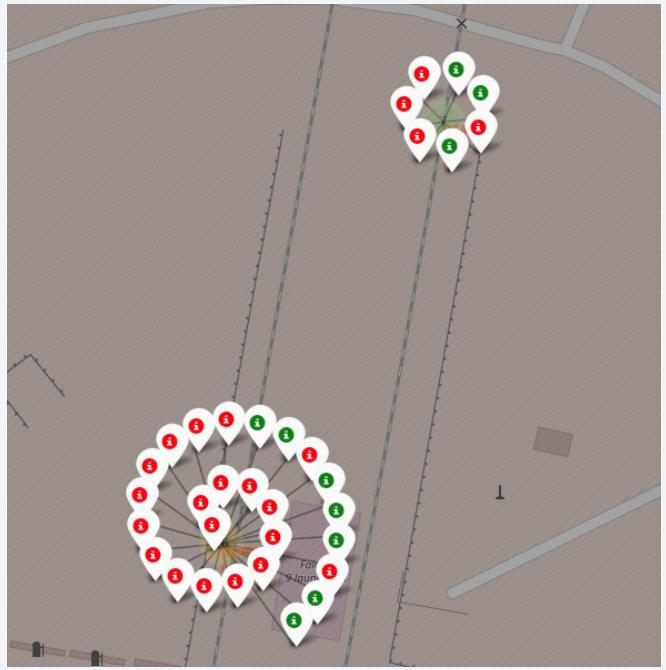
Green marker: Successful Launch

Red marker: Failure



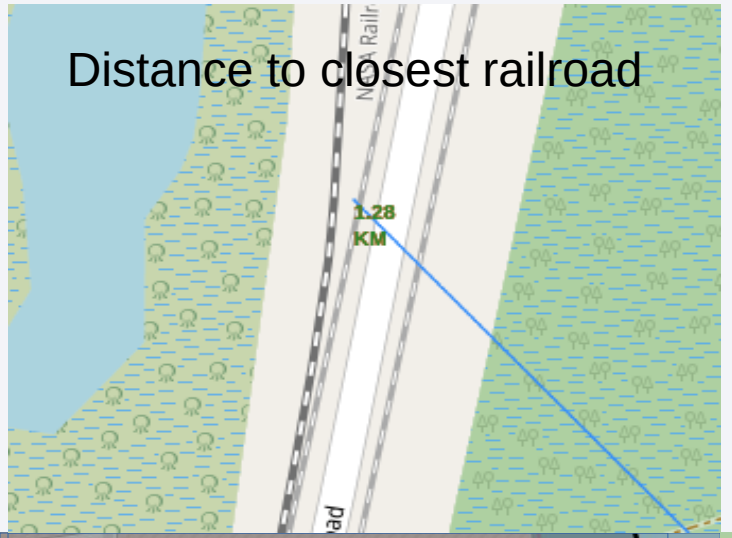
California Site



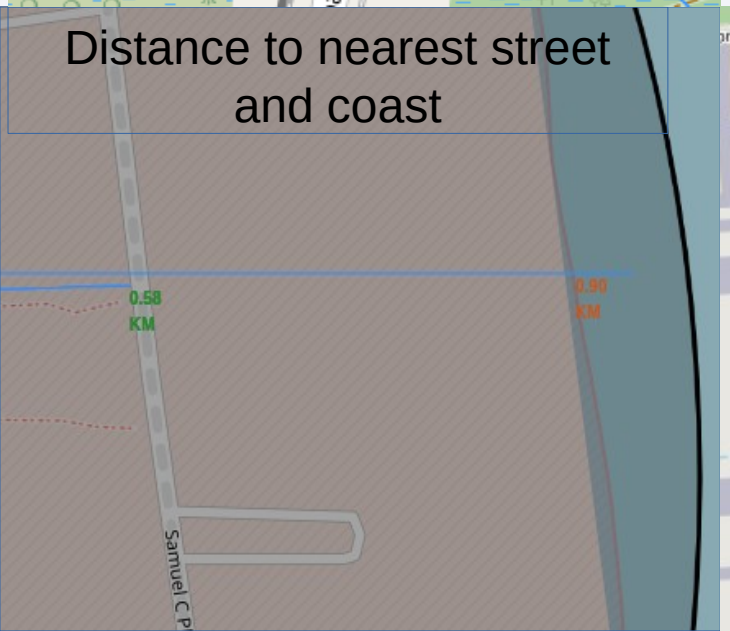


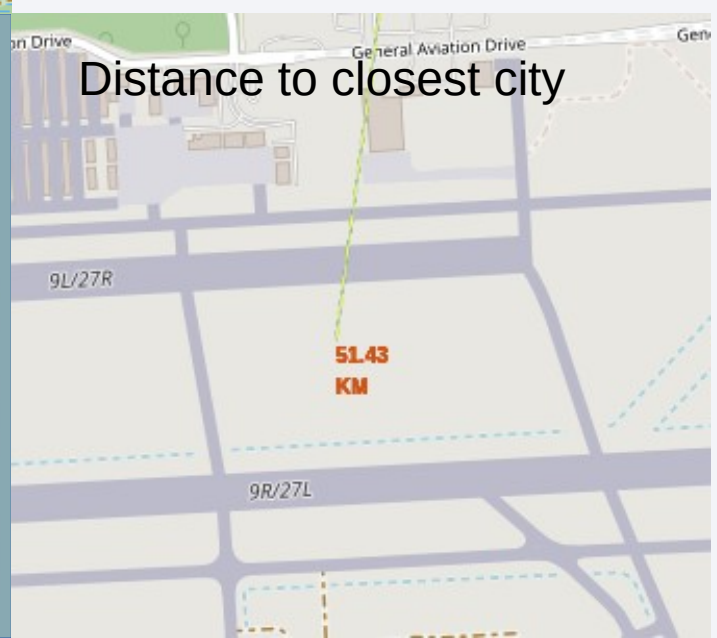
Florida Site

Launch Site Distances

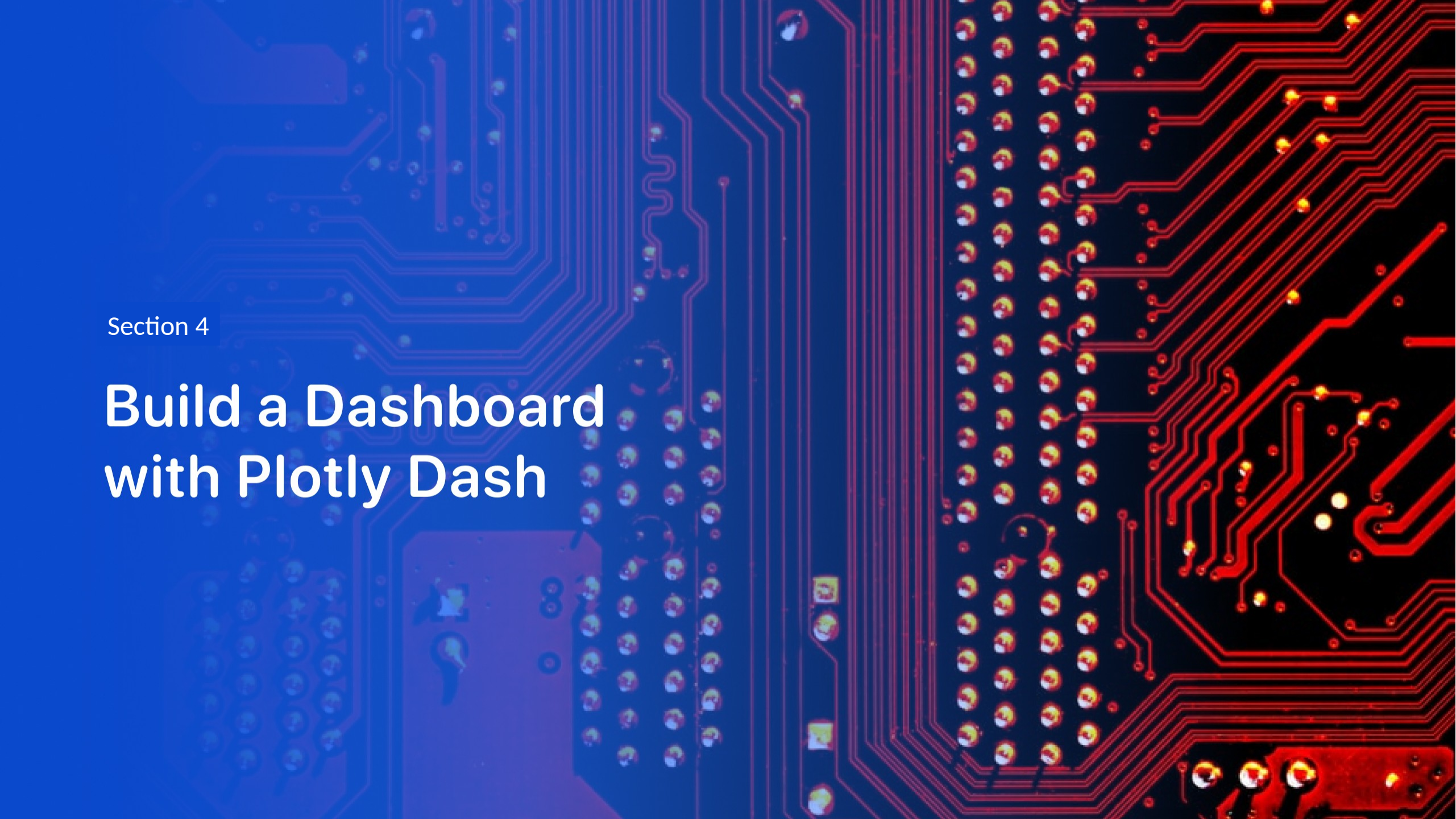


Launch sites are not typically located in close proximity to railways or highways, ensuring minimal disruption and risk from these infrastructures.

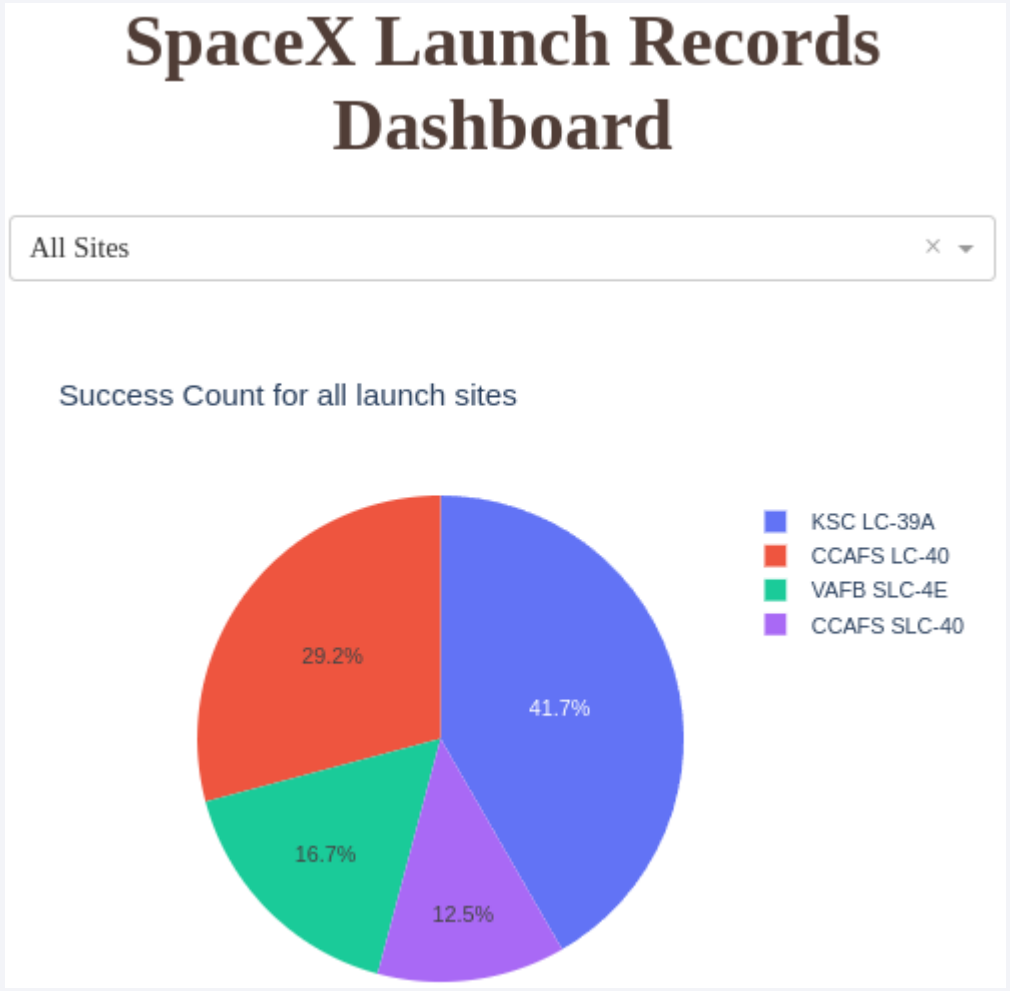


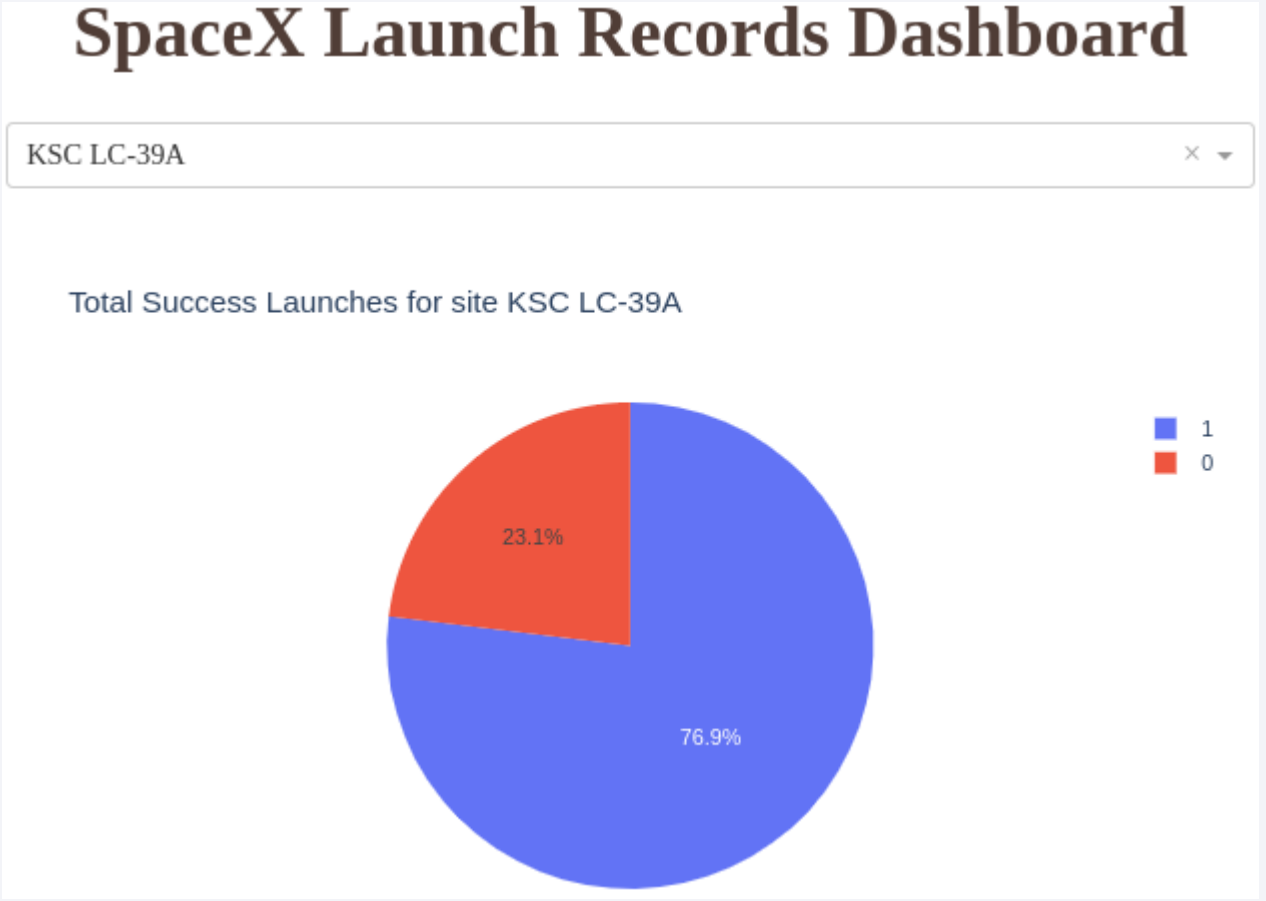


However, they are often situated near coastlines, which provides a clear trajectory for launches over open water, minimizing potential hazards to populated areas. Additionally, launch sites maintain a certain distance away from cities, further ensuring safety and reducing the risk of accidents affecting urban areas.

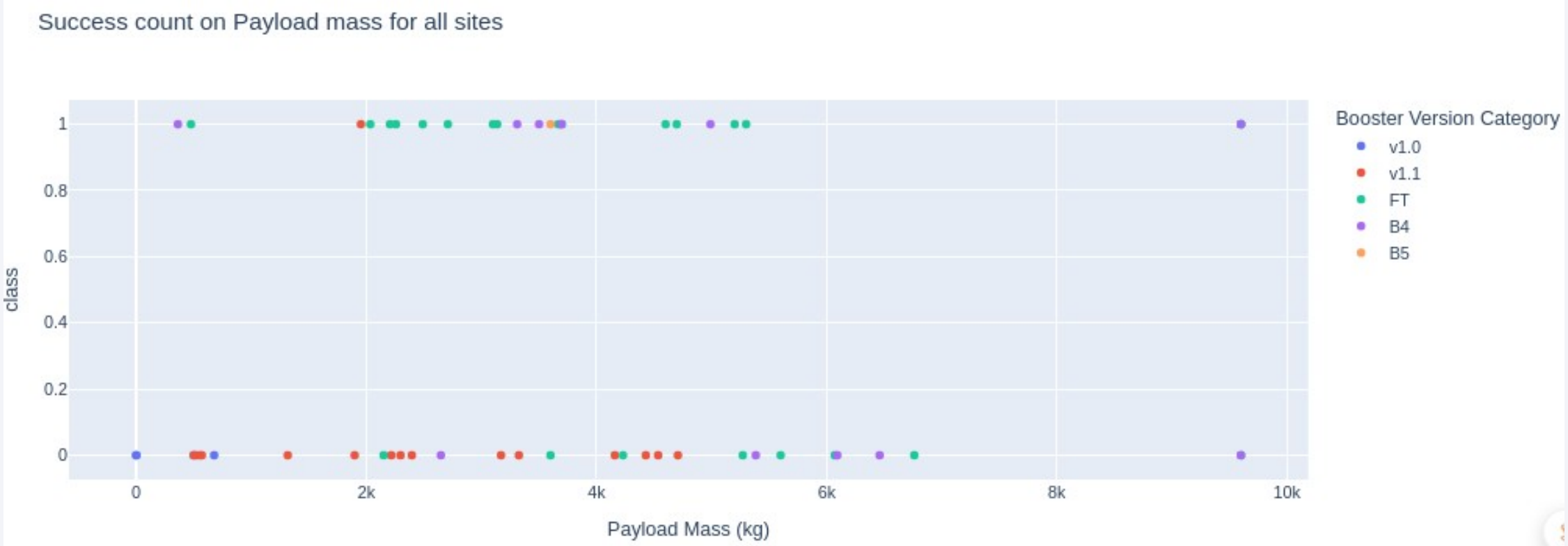

Section 4
Build a Dashboard with Plotly Dash

Launch Success Count





Scatter plot of Payload vs Launch Outcome

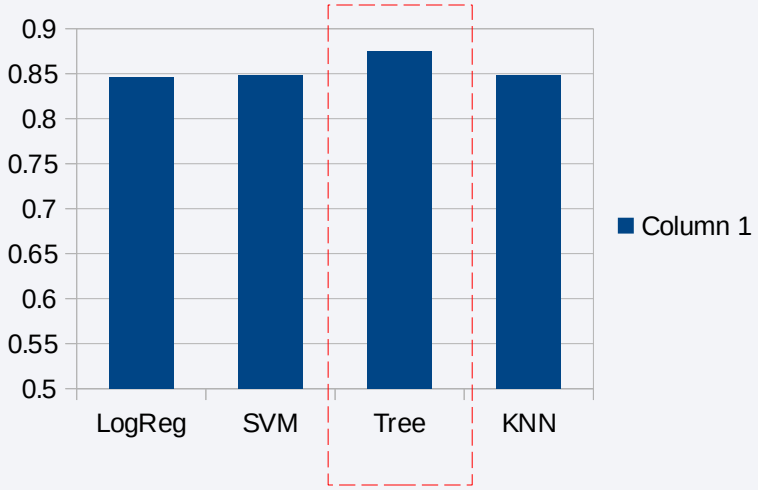


Section 5

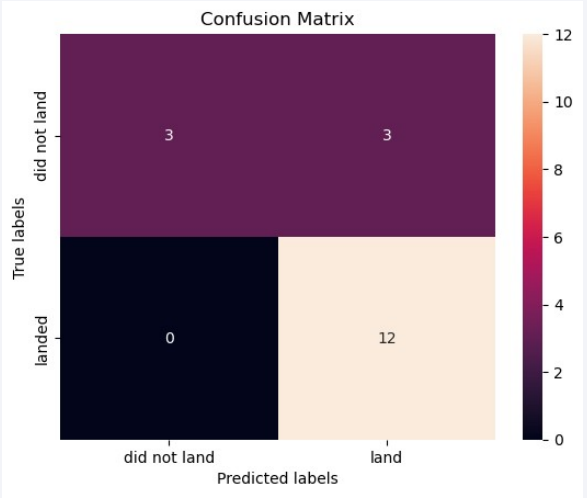
Predictive Analysis (Classification)

Classification Accuracy

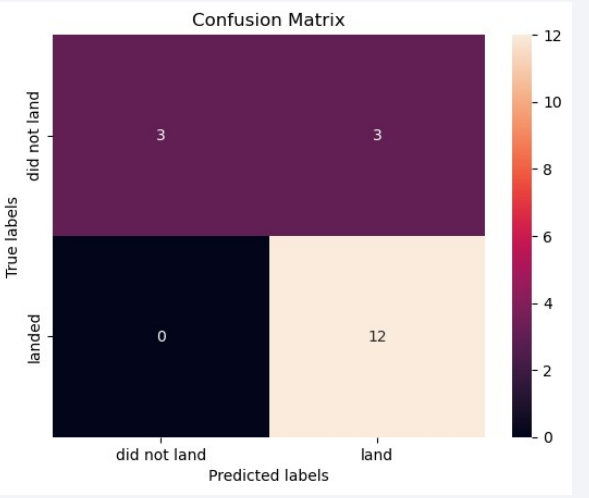
Model	Accuracy
Logistic Regression	0.8464285714285713
SVM	0.8482142857142856
Decision Tree	0.875
KNN	0.8482142857142858



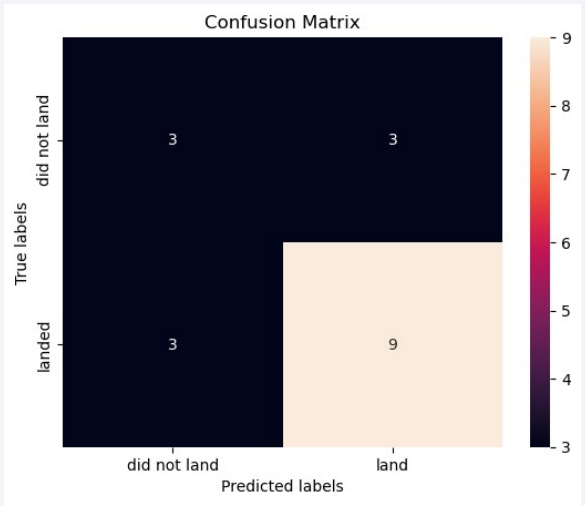
Logistic Regression Model



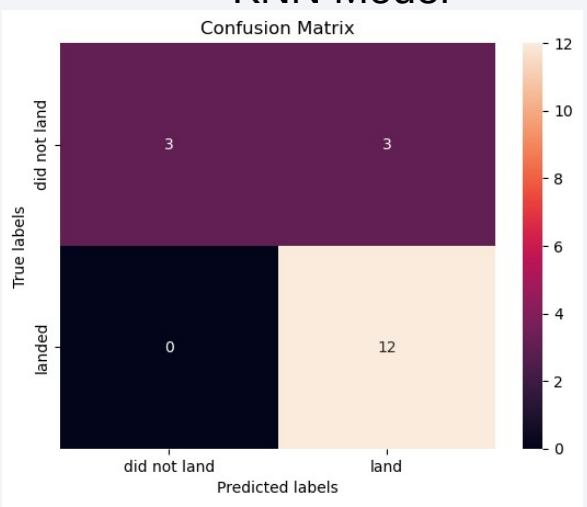
Support vector machine model



Decision Tree Model



KNN Model



Conclusions

- **Correlation between Payload Mass and Success Rate:**
  - A notable trend is observed where the success rate of rocket launches improves as the payload mass increases. This implies that rockets carrying heavier payloads tend to achieve higher reliability, likely due to the extensive engineering and rigorous testing these missions undergo.
- **Historical Progress in Launch Success:**
  - Over time, the overall success rate of rocket launches has steadily improved. This progress can be attributed to advancements in technology, more efficient processes, and accumulated experience in rocket design and operation.
- **Kennedy Space Center Launch Complex 39A (KSC LC-39A):**
  - KSC LC-39A stands out as the most successful launch site, achieving a 100% success rate for payloads under 5,500 kg. This site’s legacy of reliability is a testament to its state-of-the-art facilities and optimal geographical location, which contribute to the precision and safety of launches.
- **Optimal Machine Learning Model for Launch Predictions:**
  - Among the various machine learning models evaluated, the Decision Tree model emerged as the most effective for predicting launch success. Its performance surpasses other models due to its ability to handle complex datasets, make interpretable predictions, and capture non-linear relationships between variables, making it an invaluable tool for analyzing factors that influence launch outcomes.