IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Haochen Miao
July 9th, 2023

Outline

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

Executive Summary

- Objective: The project aims to develop a prediction model to assess the likelihood of successful landings of SpaceX's first-stage rocket boosters, a crucial determinant of the total cost of launches.
- Methodology: We utilized a comprehensive set of SpaceX launch data that included variables such as payload weight and intended orbit. We then applied several advanced machine learning algorithms to this data.
- Results: The models achieved an accuracy level of 83.3% in predicting first-stage landing success. This level of accuracy can provide significant predictive value in operational and strategic planning.
- Implications: With the ability to predict first-stage landing success, SpaceY can formulate more informed bids against SpaceX, thereby gaining a competitive edge in the commercial space industry.
- Next Steps: Future work will focus on refining the models and incorporating additional parameters to improve the prediction accuracy. Also, we will work on developing strategies for practical application of these models in the business context.
- Conclusion: This project has demonstrated the potential of data-driven decision making in the space industry. Predictive analytics can play a vital role in strategic planning and operational efficiency.

Introduction

- SpaceY is an innovative commercial space travel tech startup that is rapidly ascending in the industry, harboring ambitions to compete directly with SpaceX in the race to Mars.
- SpaceX currently offers launch services starting at $62 million. This pricing includes some fuel reserves for landing the 1st stage rocket booster, facilitating its reuse. Public statements from SpaceX suggest that constructing a 1st stage Falcon 9 booster costs over $15 million, exclusive of recouping research and development costs or profit margins.
- Considering mission-specific parameters such as payload mass and target orbit, the models developed in this study have been able to predict successful landings of the first stage rocket booster with an impressive accuracy level of 83.3%.
- With these predictive capabilities, SpaceY can make more strategic bids against SpaceX. The accuracy of the 1st stage landing predictions provides a reliable indicator of the potential cost of a launch, allowing SpaceY to make more informed decisions and strengthen its competitive positioning in the commercial space race.

Introduction: Business Problem

- SpaceX markets its Falcon 9 rocket launches at a cost of $62 million, a pricing model predicated on the ability to reuse the first stage of their rockets. The construction of the first stage is believed to exceed $15 million, a figure that doesn't take into account costs related to research and development or profit margin considerations.
- In certain scenarios, SpaceX might choose to forego the first stage reuse due to mission-specific parameters such as the payload weight, intended orbit, or customer requirements.
- Given this context, the primary objective of this report is to develop a model that can accurately predict the probability of the first stage rocket successfully landing. This prediction serves as an important proxy for the total cost of a launch, thus enabling more strategic planning and decision-making in launch operations.

Section 1

Methodology

Methodology

1. Data collection
2. Data wrangling
3. Exploratory data analysis
4. Data visualization
5. Model development
6. Reporting results to stakeholders

Data Collection

Start	•Start with Python: The entire data collection process is automated using Python, a high-level, general-purpose programming language.
Use	•Use REST API for SpaceX: REST API (Representational State Transfer Application Programming Interface) allows interaction with the SpaceX data.
Send	•Send 'GET' request: A GET request is sent to retrieve data from the SpaceX API.
Filter	•Filter data for Falcon 9 launches: The received data is filtered to include only Falcon 9 launches, which is the focus of our study.
Handle	•Handle Missing Values: Any missing values in the data are replaced with the mean value of the respective column to maintain data integrity.
Web	•Web Scraping from Wikipedia: In addition to the API data, we also gather information by scraping a list of SpaceX launches from Wikipedia.
Send	•Send Request to Wikipedia: A request is sent to Wikipedia to get the list of SpaceX launches.
Extract	•Extract Data Using HTML Table Header: The data from Wikipedia is structured in an HTML table. We parse this table and extract the data using the table headers.
Parse	•Parse HTML into a Pandas Dataframe: The extracted data is then parsed and transformed into a pandas dataframe for further processing and analysis.
End	•End: The data collection process ends here. The next steps involve data preprocessing, exploratory data analysis, and model building.

Data Collection – SpaceX API

- API
  - Used Open-Source REST API For SpaceX for web scrapping
  - “Get” request and parsed SpaceX launch data
  - Filtered pandas' data frame to include only the Falcon-9 launches
  - Replaced missing values in payload mass with mean

[hide] Flight No.	Date and time (UTC)	Version, booster[b]	Launch site	Payload[c]	Payload mass	Orbit	Customer	Launch outcome	Booster landing
78	7 January 2020 02:19:21[13]	F9 B5 ♺ B1049.4	CCSFS, SLC-40	Starlink 2 v1.0 (60 satellites)	15,600 kg (34,400 lb)[14]	LEO	SpaceX	Success	Success (drone ship)
	Third large batch and second operational flight of Starlink constellation. One of the 60 satellites included a test coating to make the satellite less reflective, and thus less likely to interfere with ground-based astronomical observations.[15]								
79	19 January 2020 15:30[16]	F9 B5 ♺ B1046.4	KSC, LC-39A	Crew Dragon in-flight abort test[17] (Dragon C205.1)	12,050 kg (26,570 lb)	Sub-orbital[18]	NASA (CTS)[19]	Success	No attempt
	An atmospheric test of the Dragon 2 abort system after Max Q. The capsule fired its SuperDraco engines, reached an apogee of 40 km (25 mi), deployed parachutes, and splashed down in the ocean 31 km (19 mi) downrange from the launch site. The test was previously slated to be accomplished with the Crew Dragon Demo-1 capsule;[20] but that test article exploded during a ground test of SuperDraco engines on 20 April 2019.[21] The abort test used the capsule originally intended for the first crewed flight.[22] As expected, the booster was destroyed by aerodynamic forces after the capsule aborted.[23] First flight of a Falcon 9 with only one functional stage — the second stage had a mass simulator in place of its engine.								
80	29 January 2020 14:07[24]	F9 B5 ♺ B1051.3	CCSFS, SLC-40	Starlink 3 v1.0 (60 satellites)	15,600 kg (34,400 lb)[14]	LEO	SpaceX	Success	Success (drone ship)
	Third operational and fourth large batch of Starlink satellites, deployed in a circular 290 km (180 mi) orbit. One of the fairing halves was caught, while the other was fished out of the ocean.[25]								
81	17 February 2020 15:05[26]	F9 B5 ♺ B1056.4	CCSFS, SLC-40	Starlink 4 v1.0 (60 satellites)	15,600 kg (34,400 lb)[14]	LEO	SpaceX	Success	Failure (drone ship)
	Fourth operational and fifth large batch of Starlink satellites. Used a new flight profile which deployed into a 212 km × 386 km (132 mi × 240 mi) elliptical orbit instead of launching into a circular orbit and firing the second stage engine twice. The first stage booster failed to land on the drone ship[27] due to incorrect wind data.[28] This was the first time a flight proven booster failed to land.								
82	7 March 2020 04:50[29]	F9 B5 ♺ B1059.2	CCSFS, SLC-40	SpaceX CRS-20 (Dragon C112.3 ♺)	1,977 kg (4,359 lb)[30] (excl. Dragon mass)	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
	Last launch of phase 1 of the CRS contract. Carries *Bartolomeo*, an ESA platform for hosting external payloads onto ISS.[31] Originally scheduled to launch on 2 March 2020, the launch date was pushed back due to a second stage engine failure. SpaceX decided to swap out the second stage instead of replacing the faulty part.[32] It was SpaceX's third flight of the Dragon C112 and the last launch of the cargo Dragon spacecraft.								

This is the Wikipedia page for web scrapping

Data Collection - Scraping

Data Acquisition

- Gathered historical launch data from the Wikipedia page titled "List of Falcon 9 and Falcon Heavy Launches".

Page Request

- Accessed the Falcon 9 Launch Wiki page using the specified Wikipedia URL.

Table Header Extraction

- Identified all column or variable names from the HTML table header present on the page.

Data Parsing and Conversion

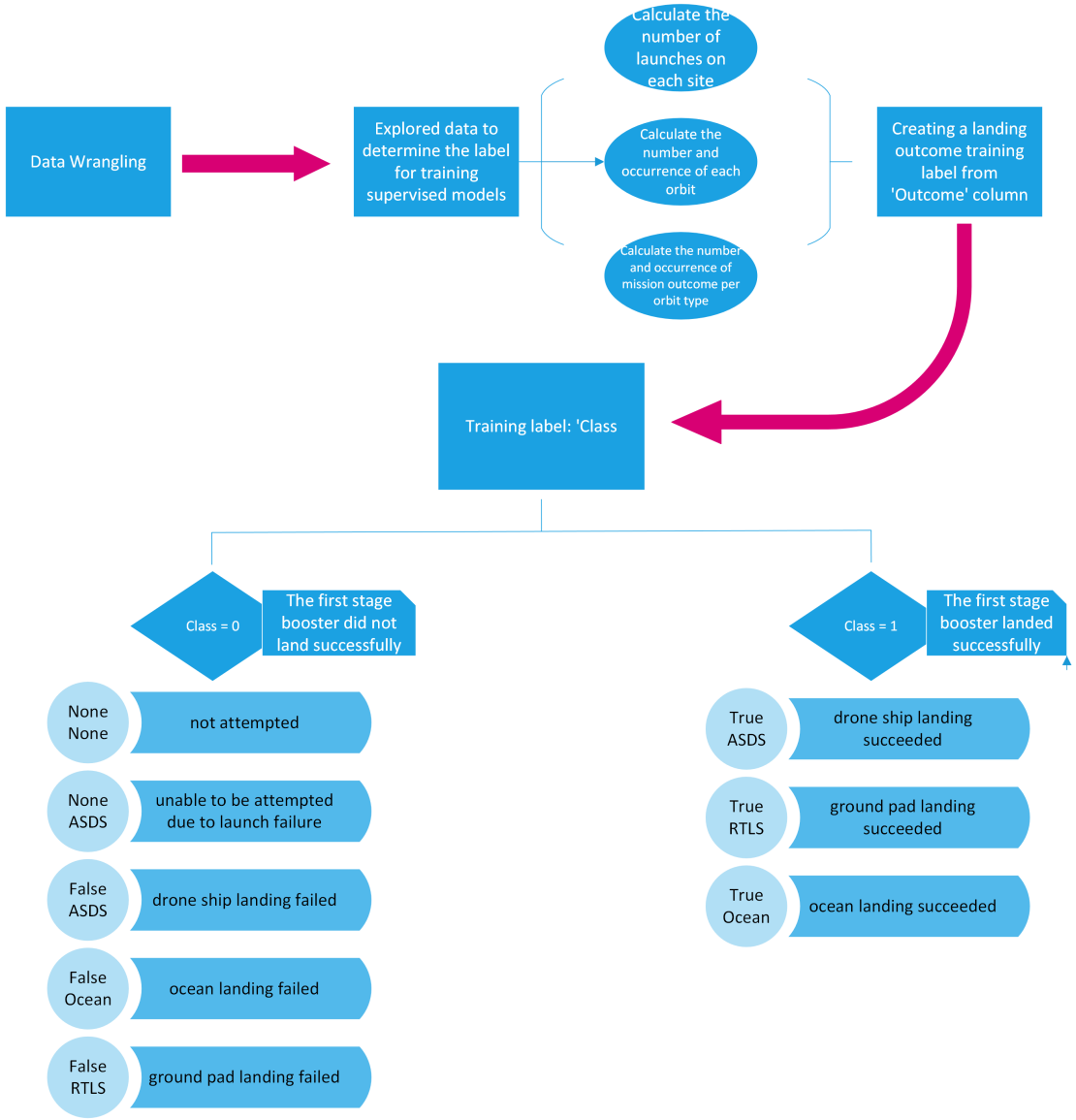
- Extracted the table content, parsed the HTML structure, and subsequently transformed the data into a manageable Pandas DataFrame.

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.561857
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007	-80.577366	28.561857
7	4	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0	B1003	-120.610829	34.632093
8	5	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B1004	-80.577366	28.561857
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
89	86	2020-09-03	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	2	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	12	B1060	-80.603956	28.608058
90	87	2020-10-06	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	3	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	13	B1058	-80.603956	28.608058
91	88	2020-10-18	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	6	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	12	B1051	-80.603956	28.608058
92	89	2020-10-24	Falcon 9	15600.0	VLEO	CCSFS SLC 40	True ASDS	3	True	True	True	5e9e3033383ecbb9e534e7cc	5.0	12	B1060	-80.577366	28.561857
93	90	2020-11-05	Falcon 9	3681.0	MEO	CCSFS SLC 40	True ASDS	1	True	False	True	5e9e3032383ecb6bb234e7ca	5.0	8	B1062	-80.577366	28.561857

90 rows x 17 columns

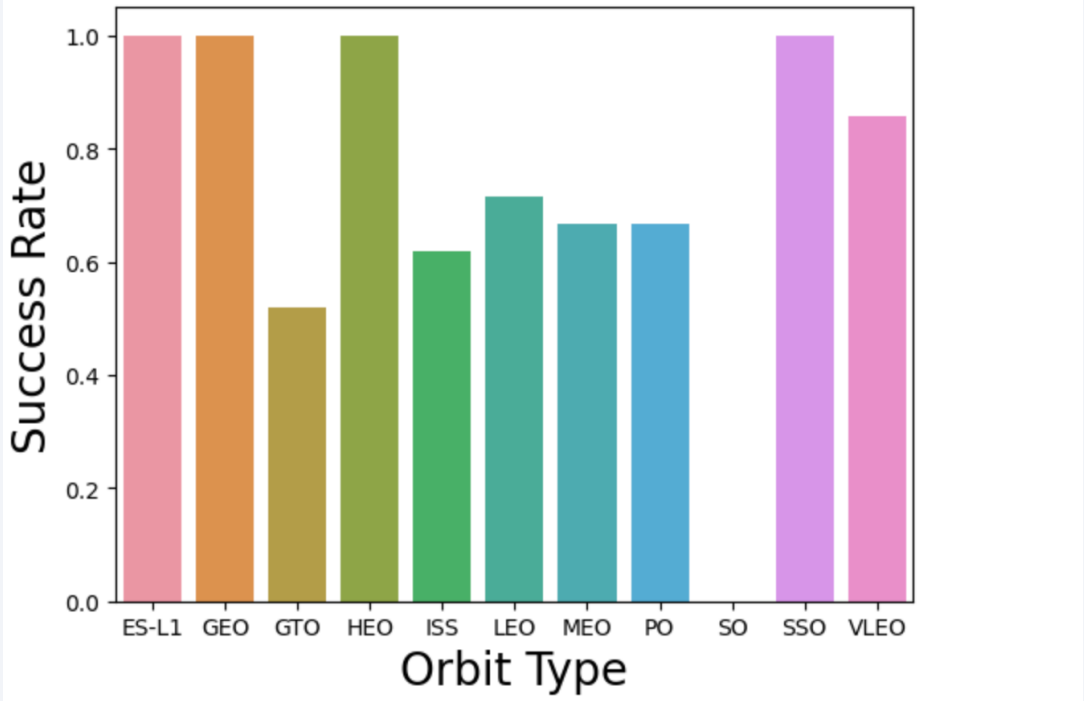
Data Wrangling

[GitHub Page]



EDA with Data Visualization

```
### TASK  3: Visualize the relationship between success rate of each orbit type
df_orbit = df.groupby(df['Orbit'], as_index=False).agg({"Class": "mean"})
#df_orbit
sns.barplot(y="Class", x="Orbit", data=df_orbit)
plt.xlabel("Orbit Type", fontsize=20)
plt.ylabel("Success Rate", fontsize=20)
plt.show()
```



- **EDA with visualization**
  - Read the dataset into a Pandas dataframe
  - Used Matplotlib and Seaborn visualization libraries to plot
    - FlightNumber x PayloadMass †
    - FlightNumber x LaunchSite †
    - Payload x LaunchSite †
    - Orbit type x Success rate
    - FlightNumber x Orbit type †
    - Payload x Orbit type †
    - Year x Success rate

† = with Class overlayed (1st stage booster landing outcome)

EDA with SQL

```
%sql SELECT substr(Date, 4, 2) as Month, MISSION_OUTCOME, Booster_Version, Launch_Site FROM SPACEXTBL WHERE subst
```

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

Month	Mission_Outcome	Booster_Version	Launch_Site
10	Success	F9 v1.1 B1012	CCAFS LC-40
11	Success	F9 v1.1 B1013	CCAFS LC-40
02	Success	F9 v1.1 B1014	CCAFS LC-40
04	Success	F9 v1.1 B1015	CCAFS LC-40
04	Success	F9 v1.1 B1016	CCAFS LC-40
06	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	CCAFS LC-40

- **EDA with SQL**
  - Loaded data into an IBM DB2 instance
  - Ran SQL queries to display and list information about
    - Launch sites
    - Payload masses
    - Booster versions
    - Mission outcomes
    - Booster landings

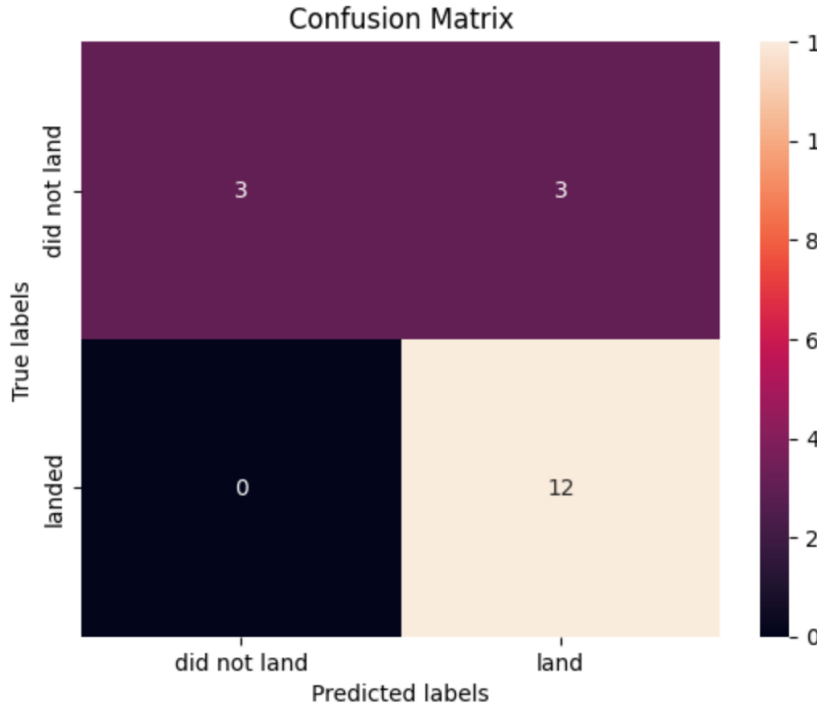
Build an Interactive Map with Folium

- **Launch Sites Location Analysis:**
  - Utilized the Python library Folium to create an interactive map, marking all the launch sites for our project.
  - These markers provide clear visibility of the launch sites' geographic location, aiding in visual analysis of location-specific trends or issues.
  - Added different color codes to distinguish successful launches from the failed ones. This visual cue allows easy identification of site performance at a glance.
- **Proximity Analysis:**
  - To evaluate the strategic location of each launch site, we calculated and marked the distances to important nearby infrastructures including railways, highways, coastlines, and cities.
  - This analysis assists in understanding the factors that might influence the choice of launch site location and its success rate.
- **Use of Interactive Elements:**
  - Interactive markers, circles, and lines were added to represent different data points and their relationships.
  - These interactive features enable users to delve deeper into the data, gain more specific insights, and interact with the data in real-time.
- **Launch Records Dashboard:**
  - Leveraging Plotly Dash, a Python interactive dashboarding library, we created an interactive interface that allows stakeholders to manipulate and explore data in real-time.
  - The dashboard features a pie chart to display the success rate color-coded by the launch site, a scatter chart correlating payload mass with landing outcomes, a range slider for payload limit adjustments, and a drop-down menu for selecting specific or all launch sites.
  - Dashboard

Predictive Analysis (Classification)

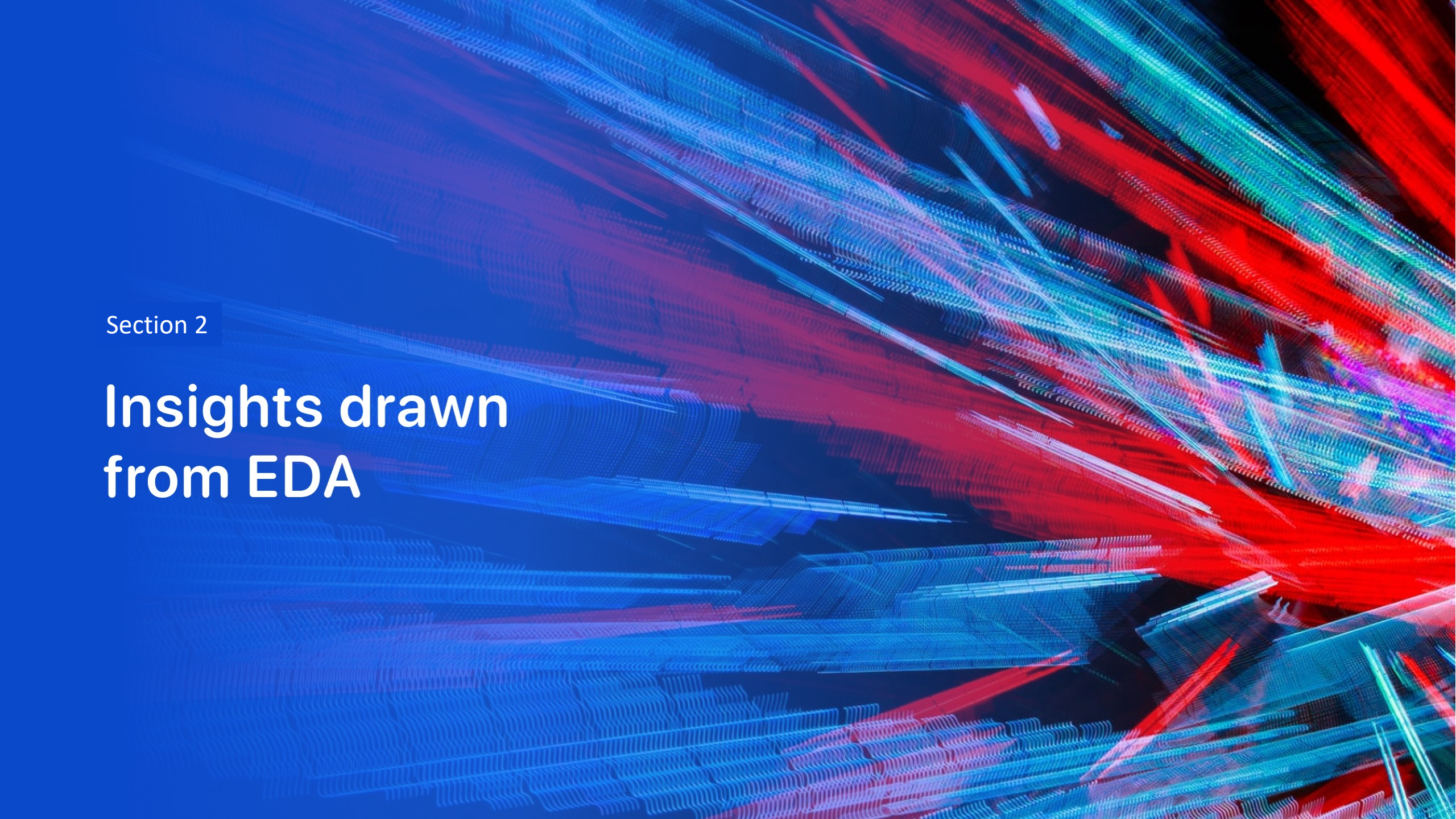
- **Library Imports and Function Definitions:**
  - Utilized essential Python libraries such as Pandas, NumPy, Matplotlib, Seaborn, and Scikit-learn for data manipulation, analysis, and visualization.
  - Defined a function to create a confusion matrix, a crucial tool for evaluating the performance of classification models.
- **Data Preparation:**
  - Loaded the cleaned and processed DataFrame created during data collection.
  - Created a 'Class' column to use as our training label, based on the outcome data obtained during data wrangling.
  - Standardized the data to ensure all features contribute equally to the model.
- **Data Splitting:**
  - Split the data into a training set and a test set to evaluate the model's performance on unseen data.
- **Model Fitting:**
  - Trained several types of models on our data, including Logistic Regression, Support Vector Machine, Decision Tree Classifier, and K Nearest Neighbors Classifier.
- **Hyperparameter Tuning:**
  - Performed a cross-validated grid-search over a range of hyperparameters for each model, utilizing the GridSearchCV function from the Scikit-learn library.
  - This enabled us to select the most effective hyperparameters for each model type.
- **Model Evaluation:**
  - The accuracy of each model was evaluated on the test data.
  - This evaluation process aided in selecting the best-performing model for predicting the successful landing of the first stage rocket booster.

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



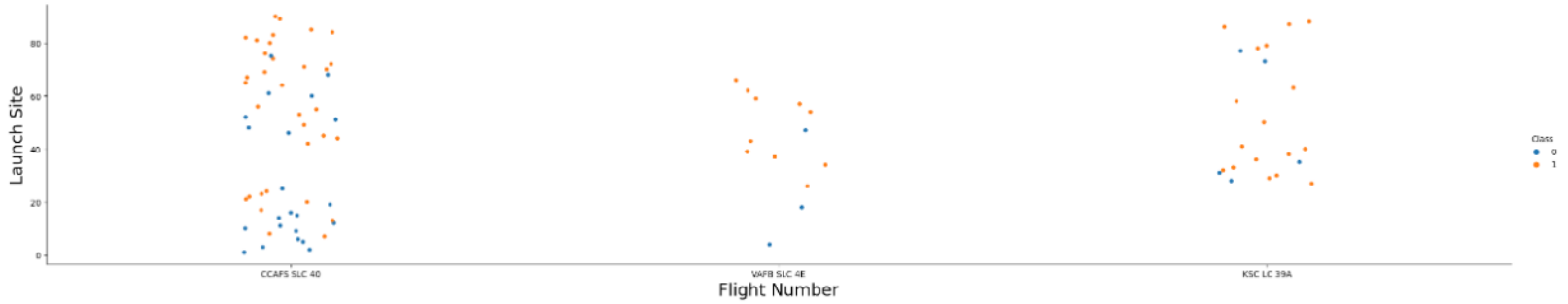
Results

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


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

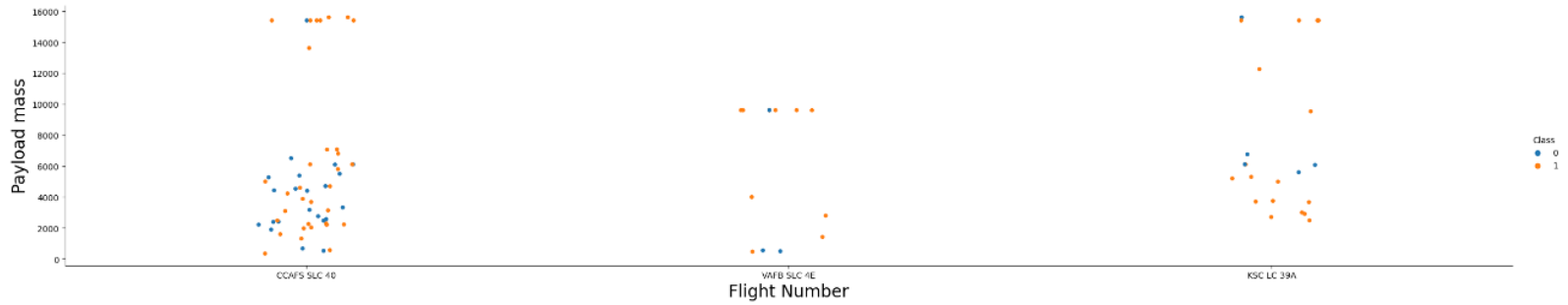
```
### TASK 1: Visualize the relationship between Flight Number and Launch Site
sns.catplot(y='FlightNumber', x='LaunchSite', hue='Class', data=df, aspect= 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Visualization of the Relationship between Payload and Launch Site

Payload vs. Launch Site

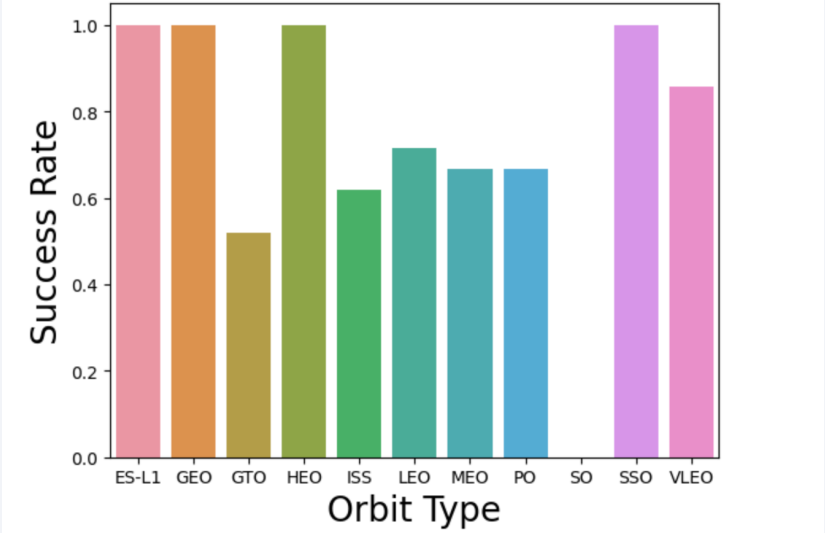
```
### TASK 2: Visualize the relationship between Payload and Launch Site
sns.catplot(y='PayloadMass', x='LaunchSite', hue='Class', data=df, aspect= 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Payload mass",fontsize=20)
plt.show()
```



Visualization of the Relationship between Payload and Launch Site

Success Rate vs. Orbit Type

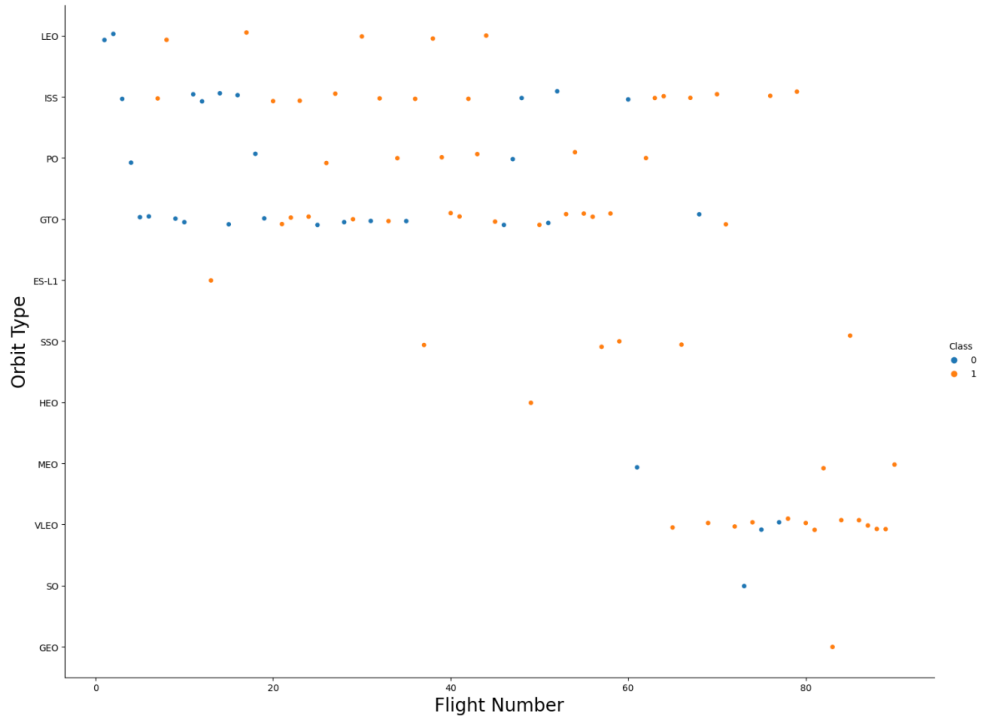
```
### TASK  3: Visualize the relationship between success rate of each orbit type
df_orbit = df.groupby(df['Orbit'], as_index=False).agg({"Class": "mean"})
#df_orbit
sns.barplot(y="Class", x="Orbit", data=df_orbit)
plt.xlabel("Orbit Type", fontsize=20)
plt.ylabel("Success Rate", fontsize=20)
plt.show()
```



Visualization of the Relationship between Success Rate and Orbit Type

Flight Number vs. Orbit Type

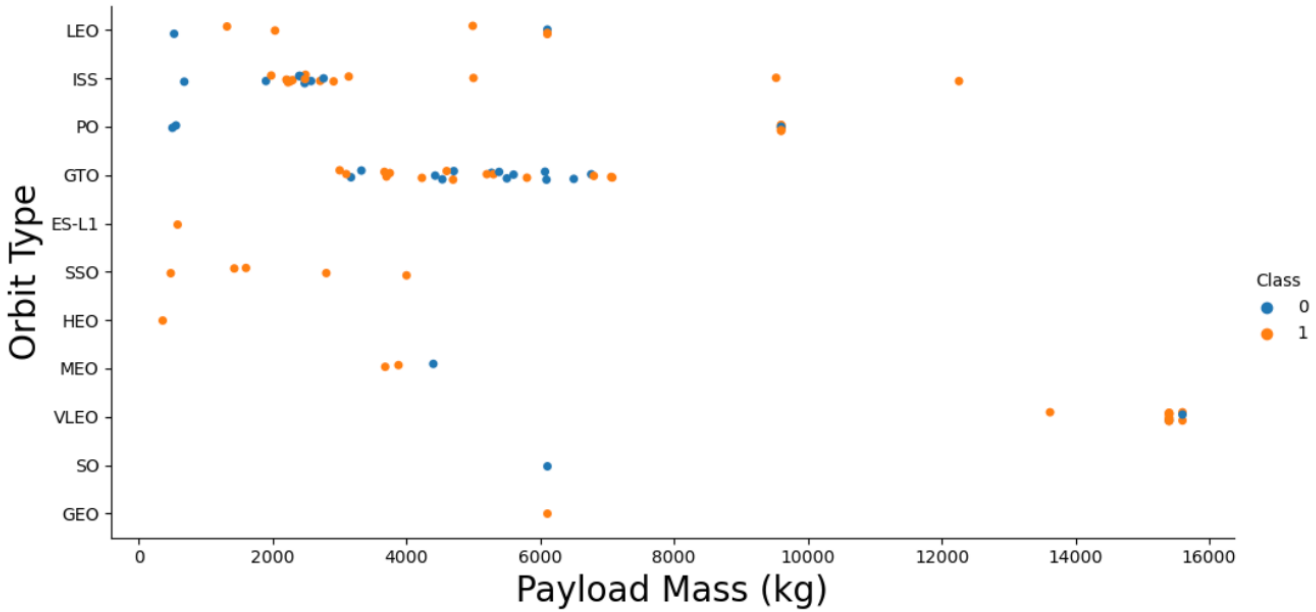
```
### TASK  4: Visualize the relationship between FlightNumber and Orbit type
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect=1.3, height=
plt.xlabel("Flight Number", fontsize=20)
plt.ylabel("Orbit Type", fontsize=20)
plt.show()
```



Visualization of the Relationship between Flight Number and Orbit Type

Payload vs. Orbit Type

```
### TASK  5: Visualize the relationship between Payload and Orbit type
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect=2)
plt.xlabel("Payload Mass (kg)", fontsize=20)
plt.ylabel("Orbit Type", fontsize=20)
plt.show()
```

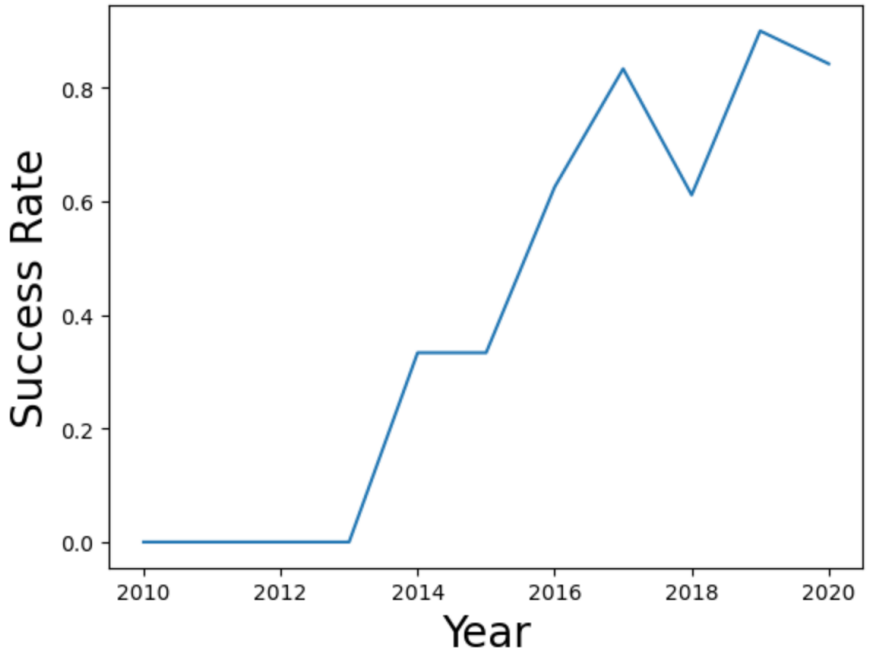


Visualization of the Relationship between Payload and Orbit Type

Launch Success Yearly Trend

```
### TASK  6: Visualize the launch success yearly trend
df["Year"] = pd.DatetimeIndex(df["Date"]).year.astype(int)

df_year = df.groupby(df['Year'], as_index=False).agg({"Class": "mean"})
#df_orbit
sns.lineplot(y="Class", x="Year", data=df_year)
plt.xlabel("Year", fontsize=20)
plt.ylabel("Success Rate", fontsize=20)
plt.show()
```



Visualization of the Relationship between Success Rate and Year Trend

All Launch Site Names

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
None

Names of Unique launch sites in the space mission

Launch Site Names Begin with 'CCA'

Task 2

Display 5 records where launch sites begin with the string 'CCA'

```
%sql SELECT LAUNCH_SITE from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Launch Site Names Started with 'CCA'

Total Payload Mass

Task 3

Display the total payload mass carried by boosters launched by NASA (CRS)

```
%sql SELECT sum(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;
```

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

payloadmass
619967.0

Total Payload Mass

Average Payload Mass by F9 v1.1

Task 4

Display average payload mass carried by booster version F9 v1.1

```
%sql SELECT avg(PAYLOAD_MASS__KG_) AS avgpayload from SPACEXTBL;
```

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

avgpayload
6138.287128712871

Average Payload Mass by F9 v1.1

First Successful Ground Landing Date

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
%sql select min(DATE) from SPACEXTBL
```

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

min(DATE)
01/06/2014

Date of the first successful landing outcome

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
%sql select Booster_Version from SPACEXTBL where Landing_Outcome='Success (drone shi
```

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

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

names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

```
%sql select count(MISSION_OUTCOME) as missionoutcomes from SPACEXTBL GROUP BY MISSION
```

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

missionoutcomes
0
1
98
1
1

total number of successful and failure mission outcomes

Boosters Carried Maximum Payload

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
%sql select BOOSTER_VERSION as boosterversion from SPACEXTBL where PAYLOAD_MASS__KG_=
```

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

boosterversion
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

names of the booster_versions which have carried the maximum payload mass

2015 Launch Records

Task 9

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.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

```
%sql SELECT substr(Date, 4, 2) as Month, MISSION_OUTCOME, Booster_Version, Launch_Sit
```

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

Month	Mission_Outcome	Booster_Version	Launch_Site
10	Success	F9 v1.1 B1012	CCAFS LC-40
11	Success	F9 v1.1 B1013	CCAFS LC-40
02	Success	F9 v1.1 B1014	CCAFS LC-40
04	Success	F9 v1.1 B1015	CCAFS LC-40
04	Success	F9 v1.1 B1016	CCAFS LC-40
06	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	CCAFS LC-40

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

Task 10

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.

```
lower_date=df[df["Date"]=="04-06-2010"]
higher_date=df[df["Date"]=="16-03-2017"]
df_timed=df.iloc[0:31,]
df_timed['Landing_Outcome'].value_counts(ascending=False)
```

```
No attempt                10
Failure (drone ship)       5
Success (drone ship)       5
Controlled (ocean)         3
Success (ground pad)       3
Failure (parachute)        2
Uncontrolled (ocean)       2
Precluded (drone ship)     1
Name: Landing_Outcome, dtype: int64
```


Section 3
Launch Sites
Proximities Analysis

All Launch Sites Location

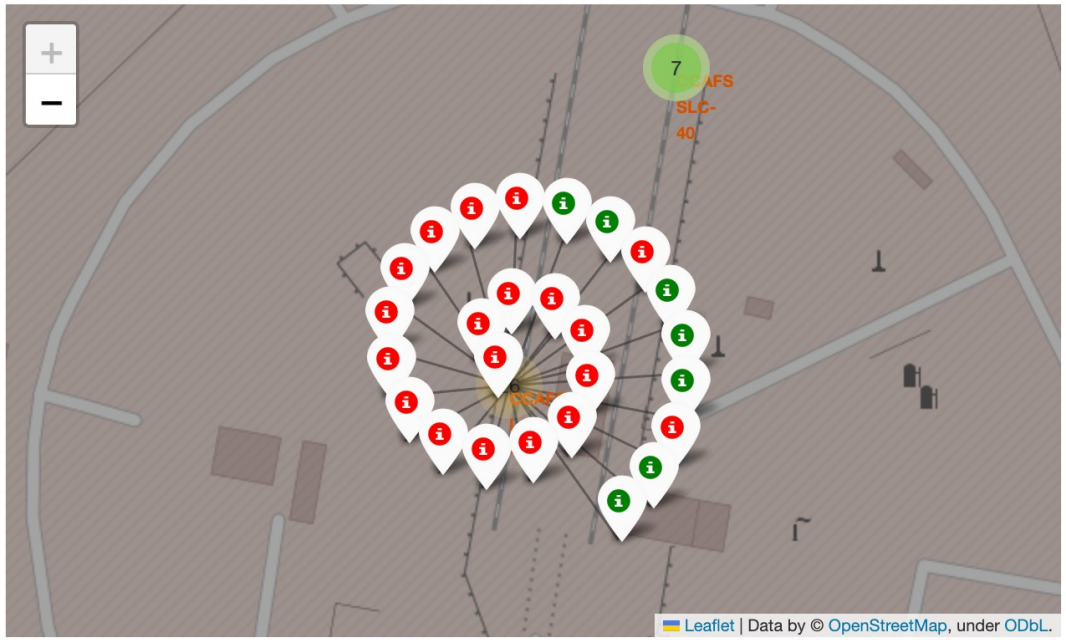
```
# Initialize the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=4)
# For each launch site, add a Circle object based on its coordinate (Lat, Long) va
for index, row in launch_sites_df.iterrows():
    coordinate = [row['Lat'], row['Long']]
    folium.Circle(coordinate, radius=1000, color='#000000', fill=True).add_child(f
    folium.map.Marker(coordinate, icon=DivIcon(icon_size=(20,20),icon_anchor=(0,0),
site_map
```

Visualizing the launch sites on a map highlights the importance of launch site proximity to the coast and equator:

Launch Site Location(zoomed in)

```
# Add marker_cluster to current site_map
site_map.add_child(marker_cluster)

# for each row in spacex_df data frame
# create a Marker object with its coordinate
# and customize the Marker's icon property to indicate if this launch was successed
# e.g., icon=folium.Icon(color='white', icon_color=row['marker_color']
for index, row in spacex_df.iterrows():
    # create and add a Marker cluster to the site map
    coordinate = [row['Lat'], row['Long']]
    folium.map.Marker(coordinate, icon=folium.Icon(color='white',icon_color=row['ma
site_map
```

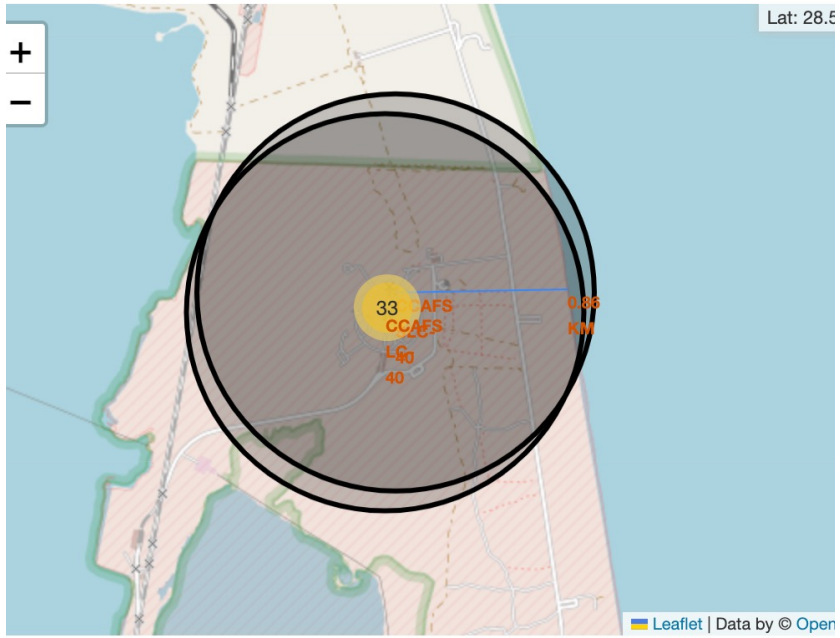


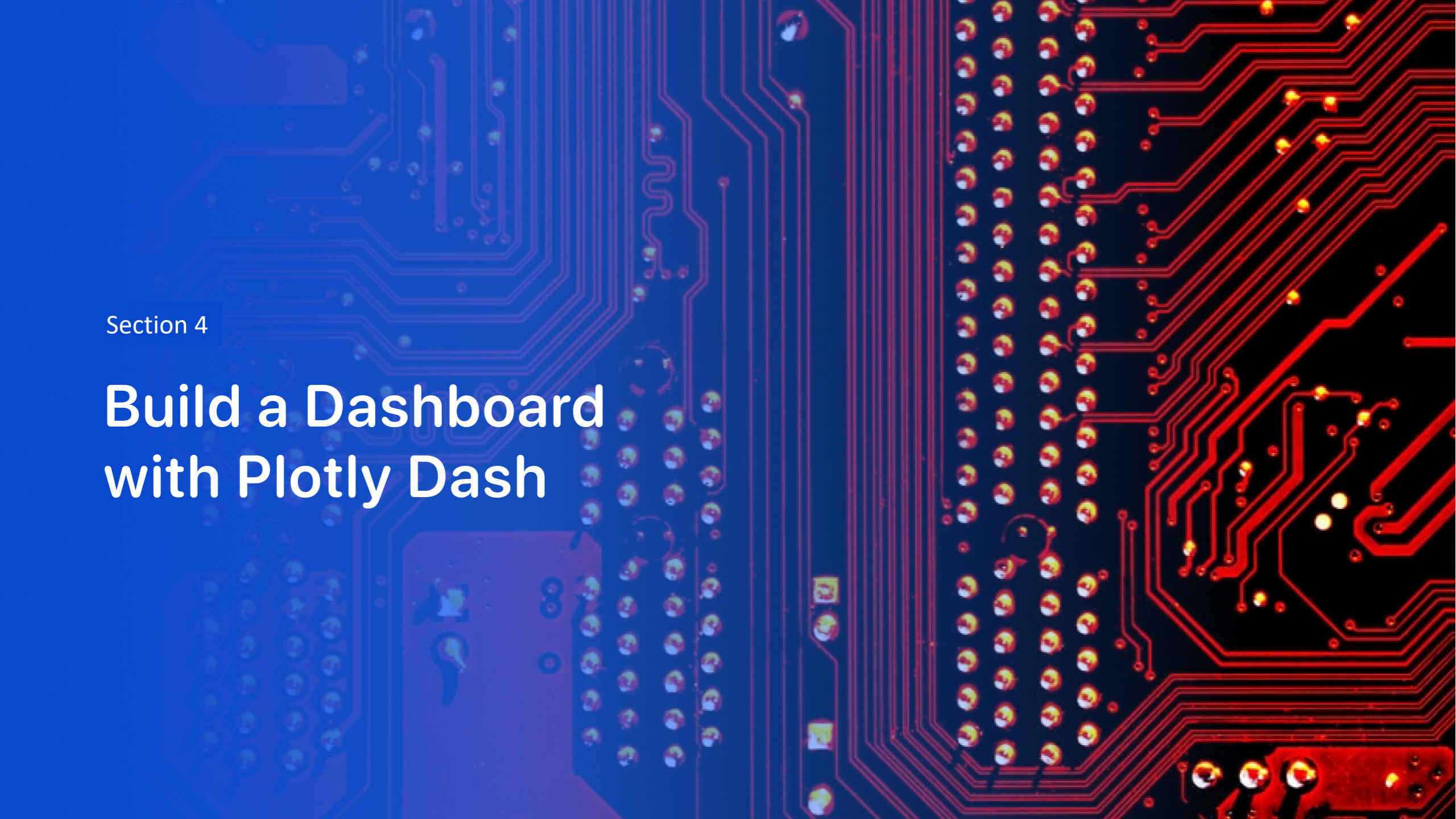
Visualizing the booster landing outcomes for each launch site highlights which launch sites have relatively high success rates, namely KSC LC39A

Launch Site Location(with coast line)

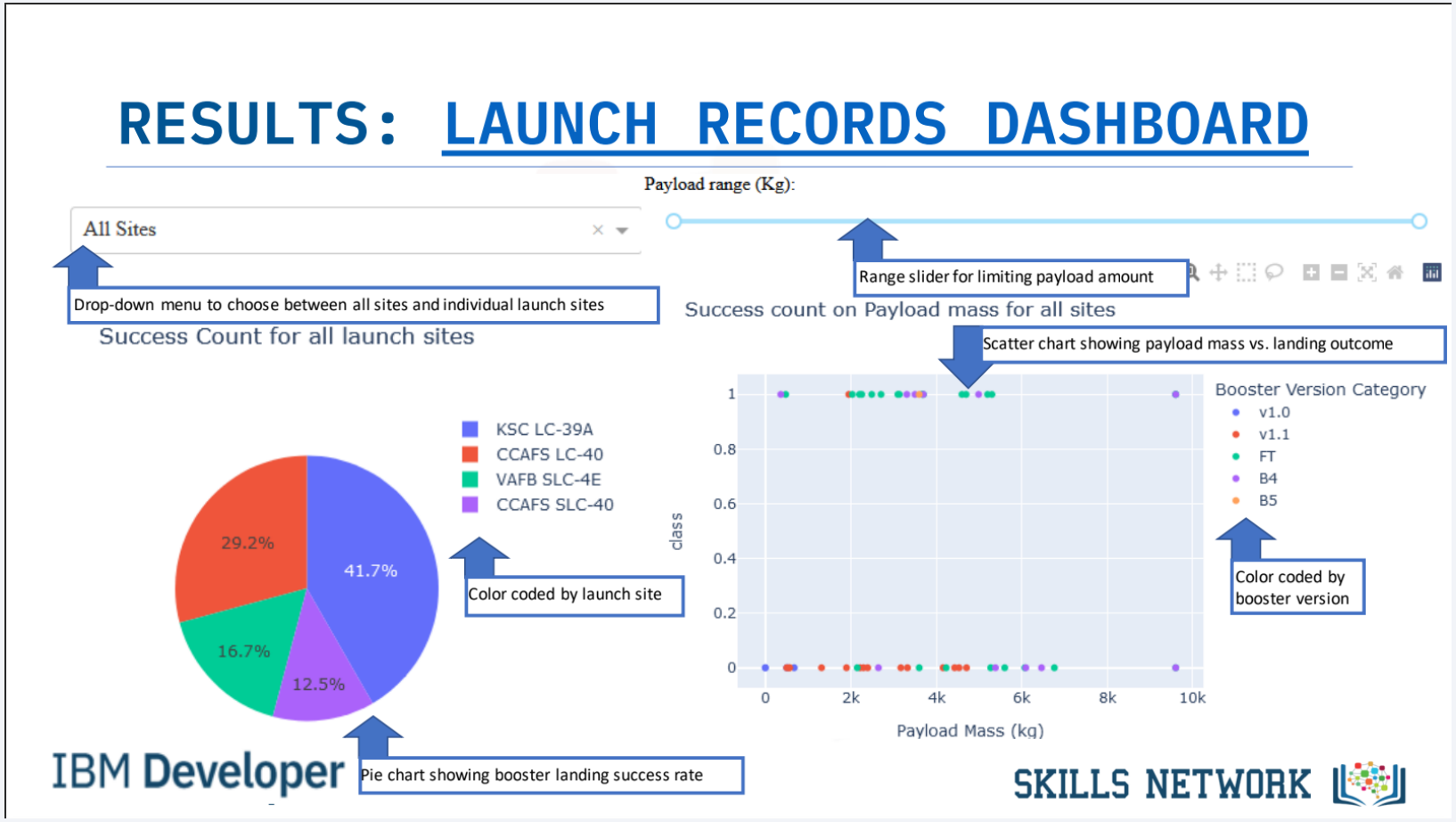
- Visualizing the railway, highway, coastline, and city proximities for each launch site allows us to
- see how close each is, for example:
    - Proximities for CCAFS SLC-40:
        - railway: 1.28 km
            - transporting heavy cargo
        - highway: 0.58 km
            - transporting personel and equipment
        - coastline: 0.86 km
            - optionality to abort launch and attempt water landing
            - minimizing risk from falling debris
        - city: 51.43 km
            - minimizing danger to population dense areas.

```
Create a `folium.PolyLine` object using the coastline coordinat
lines=folium.PolyLine(locations=coordinates, weight=1)
ordinates = [[launch_site_lat,launch_site_lon],[coastline_lat,c
nes=folium.PolyLine(locations=coordinates, weight=1)
te_map.add_child(lines)
```

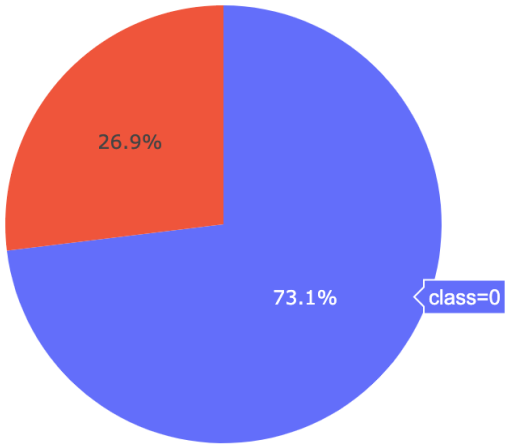



Section 4
Build a Dashboard with Plotly Dash

Dashboard Screenshot



<Dashboard Screenshot 3>



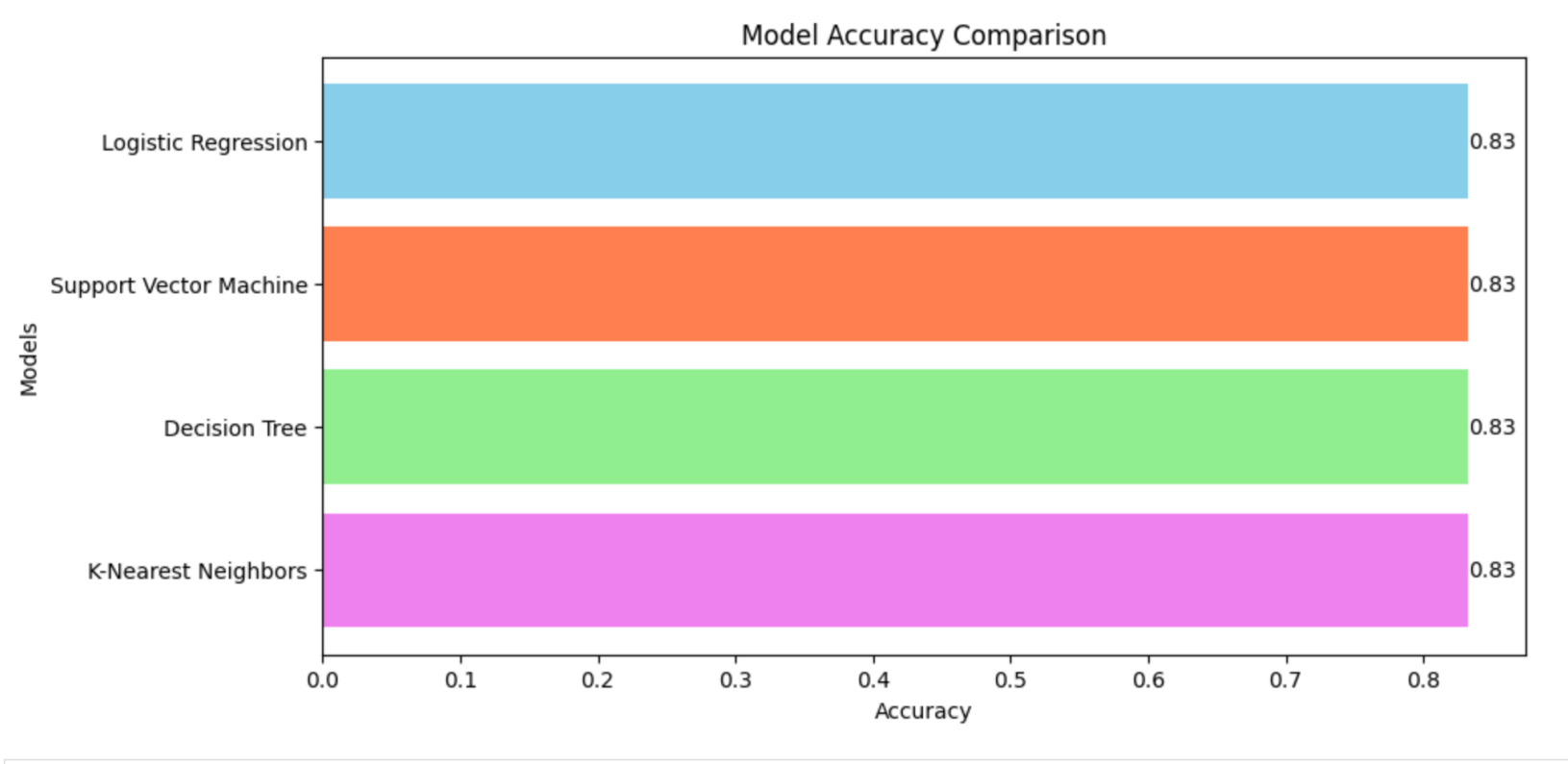
- • Dashboard observations:
- • FAFB SLC-4E had the heaviest successful
- booster landing success
- • KSC LC-39A has the highest booster landing
- success rate
- • Payloads < 5,300 kg had the highest booster
- landing success rate
- • Payloads > 5,300 kg had the lowest booster
- landing success rate

Example dashboard view:
Booster landing success rate for CCAFS LC-40

Section 5

Predictive Analysis (Classification)

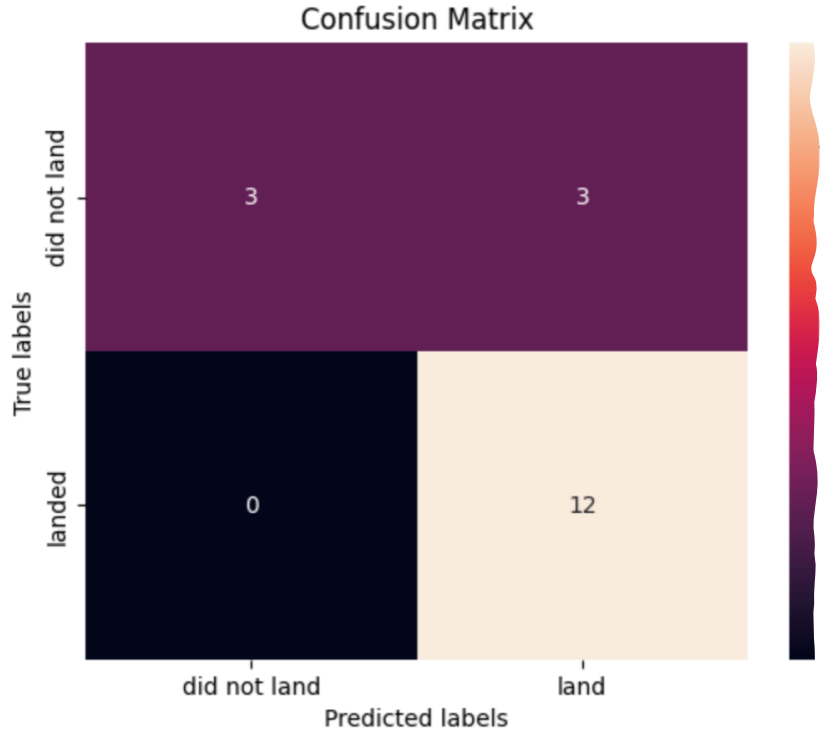
Classification Accuracy



- Each of the four models built came back with the same accuracy score, 83.33%

Confusion Matrix

```
yhat = knn_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



- The confusion matrices of the best performing models (4-way-tie) are the same
- The major problem is false positives as evidenced by the models incorrectly predicting
- the 1st stage booster to land in 3 out of 18 samples in the test set

Conclusions

Content:

- This project's predictive models can help SpaceY anticipate with approximately 83.3% accuracy whether SpaceX will successfully land the 1st stage booster.
- Given the substantial manufacturing cost of the 1st stage booster, around $15 million, this predictive capacity provides SpaceY with a valuable edge when competing with SpaceX for bids.
- This model informs SpaceY's bidding strategies by hinting when SpaceX's bid may include the cost of a sacrificed 1st stage booster, potentially pushing the SpaceX bid from $62 million to over $77 million.

Future Opportunities:

- Optimize the current model by freezing the best combination of hyperparameters and re-fitting it with the complete dataset, rather than just the training data. This approach could yield even more accurate predictions, although it would not allow for subsequent accuracy measurement.
- Enhance the model's performance by incorporating new launch data as it becomes available.
- Refine our approach by dividing our current model into two separate models: one that predicts if SpaceX will attempt to land the 1st stage, and another predicting if SpaceX will succeed in their attempt.
- Develop a related model to predict if SpaceX will launch using a previously flown 1st stage booster, enabling SpaceY to anticipate when SpaceX's bid may include a discount.

Appendix

- https://github.com/haochenmiao/Portfolio/blob/main/Data%20Scientist%20Portfolio/SpaceX-Mission-Analysis-and-Prediction/jupyter-labs-spacex-data-collection-api.ipynb
- https://github.com/haochenmiao/Portfolio/blob/main/Data%20Scientist%20Portfolio/SpaceX-Mission-Analysis-and-Prediction/jupyter-labs-webscraping.ipynb
- https://github.com/haochenmiao/Portfolio/blob/main/Data%20Scientist%20Portfolio/SpaceX-Mission-Analysis-and-Prediction/IBM-DS0321EN-SkillsNetwork_labs_module_1_L3_labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb
- https://github.com/haochenmiao/Portfolio/blob/main/Data%20Scientist%20Portfolio/SpaceX-Mission-Analysis-and-Prediction/jupyter-labs-eda-sql-coursera_sqllite.ipynb
- https://github.com/haochenmiao/Portfolio/blob/main/Data%20Scientist%20Portfolio/SpaceX-Mission-Analysis-and-Prediction/IBM-DS0321EN-SkillsNetwork_labs_module_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb
- https://github.com/haochenmiao/Portfolio/blob/main/Data%20Scientist%20Portfolio/SpaceX-Mission-Analysis-and-Prediction/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb
- https://github.com/haochenmiao/Portfolio/blob/main/Data%20Scientist%20Portfolio/SpaceX-Mission-Analysis-and-Prediction/spacex_dash_app.py
- https://github.com/haochenmiao/Portfolio/blob/main/Data%20Scientist%20Portfolio/SpaceX-Mission-Analysis-and-Prediction/IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb
- https://github.com/haochenmiao/Portfolio/tree/main/Data%20Scientist%20Portfolio/SpaceX-Mission-Analysis-and-Prediction

Thank you!