IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Gabor Kovacs
04/08/2023

Outline

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

Executive Summary

- Summary of methodologies:
  - Data collection from 2 sources: web scraping with beautifulsoup and connection via API
  - Data Wrangling: landing outcome label was generated
  - EDA – Exploratory data analyses utilizing SQL and visualizations
  - Interactive visual analyses by creating dashboard with Folium and Plotly Dash
  - ML Predictive analytics applying classification model, parameter setting and model selection via best score and confusion matrix ranking
- Summary of all results
  - Comparative analytics of the attributes of the 3 SpaceX Launch Sites: with **KSC LC-39A** holding the best success rate
  - Success rate improved steadily from the year 2013 with a hiccup in 2018 then improved again
  - In terms of Booster version **FT** within payload mass range of 2.2-3.2 tons and **B4** within payload mass range of 3.3-5 tons are the most prone to success
  - Predictive analytics showed that 3 out of the 4 model tested are equally good fit: Logistic regression, SVM and K-nearest neighbors, all of these three methods give the best performance, with accuracy of 0.83

Introduction

- Project background and context
  - we are to predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.
- Problems you want to find answers
  - Determine the best conditions of rocket launch to recognize characteristics promote successful landing
  - What are the attributes most important to focus on to improve successful landing rate?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Obtained by both Json Apis and Webscraping of Wikipages with Beautifulsoap package to webscrape HTML tables
- Perform data wrangling
  - Wrangling data using an Api, sampling data and dealing with null values
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected.
- SpaceX rest Api endpoints have been utilized to collect data
- You need to present your data collection process use key phrases and flowcharts
- Key phrases:
  - Url, get response, Json
  - We are going to have a list of Json objects which each represents a launch
  - To converting it to a dataframe: Json_normalize function

Data Collection – SpaceX API

1. Collecting the data via API calls:

```
def getBoosterVersion(data):
    for x in data['rocket']:
        if x:
            response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
            BoosterVersion.append(response['name'])
```

- 2. Convert into dataframe with json

```
# Use json_normalize meethod to convert the json result into a dataframe
jlist = requests.get(static_json_url).json()
df = pd.json_normalize(jlist)
df.head()
```

	static_fire_date_utc	static_fire_date_unix	tbd	net	window	rocket	success	details	crew	ships	capsules	payloads	
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Engine failure at 33 seconds and loss of vehicle	[]	[]	[]	[5eb0e4b5b6c3bb0006eeb1e1]	5e9e4502f50909
1	None	NaN	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Successful first stage burn and transition to second stage, maximum altitude 289 km, Premature	[]	[]	[]	[5eb0e4b6b6c3bb0006eeb1e2]	5e9e4502f50909

Filter for Falcon 9 launches

```
# Hint data['BoosterVersion']!='Falcon 1'
filt =  df['BoosterVersion']!=  'Falcon 1'
data_falcon9 = df.loc[filt]
data_falcon9.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Lon
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.
5	8	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.

Convert the dataframe into csv:

Data Collection - Scraping

1. Create a beautifulsoap instance:

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, 'html.parser')
```

2. Get the column names:

```
column_names = []

# Apply find_all() function with `th` element
# Iterate each th element and apply the provic
# Append the Non-empty column name (`if name i

table = first_launch_table.find_all('th')
for row in table:
    name = extract_column_from_header(row)
    if name is not None and len(name) > 0:
        column_names.append(name)
```

4. Once parsed launched valus filled into the distionary convert it to the final dataframe

```
df=pd.DataFrame(launch_dict)
```

```
df.head()
```

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	[[SpaceX], \n]	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45

3. Identify "th elements" & gather as column names

4. Creating an empty launch dict to host the pandas dataframe:

```
launch_dict= dict.fromkeys(column_nam

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

# Let's initial the launch_dict with
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
```

5. Save to csv

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

1 - Identify and calculate proportion of missing values:

```
In [3]: df.isnull().sum()/df.count()*100

Out[3]: FlightNumber       0.000
        Date               0.000
        BoosterVersion     0.000
        PayloadMass        0.000
        Orbit              0.000
        LaunchSite         0.000
        Outcome            0.000
        Flights            0.000
        GridFins           0.000
        Reused             0.000
        Legs               0.000
        LandingPad        40.625
        Block              0.000
```

2 – Identify column types to get an idea how to deal with them

3 – Generate a set of bad outcomes out of landing outcomes
4- To use it to define landing class indices to generate a new variable
5 – Now we can define a success rate applíing the mean function

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts
  - CatPlot/Scatterplot was used to check association between pair of features
  - BarPlot applied to check orbit vs success rate to grasp which orbit has highest success rate
  - Success rate evolution checked over year: line chart applied after years extracted from dates
  - Feature engineering: get_dummies function applied to decode dataframe into floats enabling predictive analytical methods to exercise on them
- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
  - Simple queries applied to get to unique/distinct launch sites, display records restricted by wildcard strings
  - Utilize aggregate commands to answer queries related to Total and Average, Minimum
  - Specify result by applying restriction via Where command
  - Using subquery to get to Booster Version characterized by the Max payload mass
  - Define a subset of the table by selecting and restricting to specific columns in a particular year
  - Count successful outcomes for a given time frame

- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
  - Markers and circles have been added to Nasa Center and to the 3 launch sites SpaceX uses.
  - Mouse position created in order to be able to capture certain coordinates on the map
  - Draw lines and distance between objects
- Explain why you added those objects
  - In order to form a general idea of where are the launch sites situated
  - For instance we can observe that all launch sites are in close proximity to the coast.
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
  - Pie chart to show the success rate of all launch sites
  - Scatter plot to observe if there is a correlation between payload mass and Booster version
- Explain why you added those plots and interactions
  - Pie chart is an easy way to visualize the proportion of success rate among launch sites as well as on individual launch site level
  - Scatter plot is a good way to check if a feature has explanatory power over an outcome, in other words if the outcome is dependent on the variable
  - We ensured outcome is visible on the scatter plot by color coding label

Predictive Analysis (Classification)

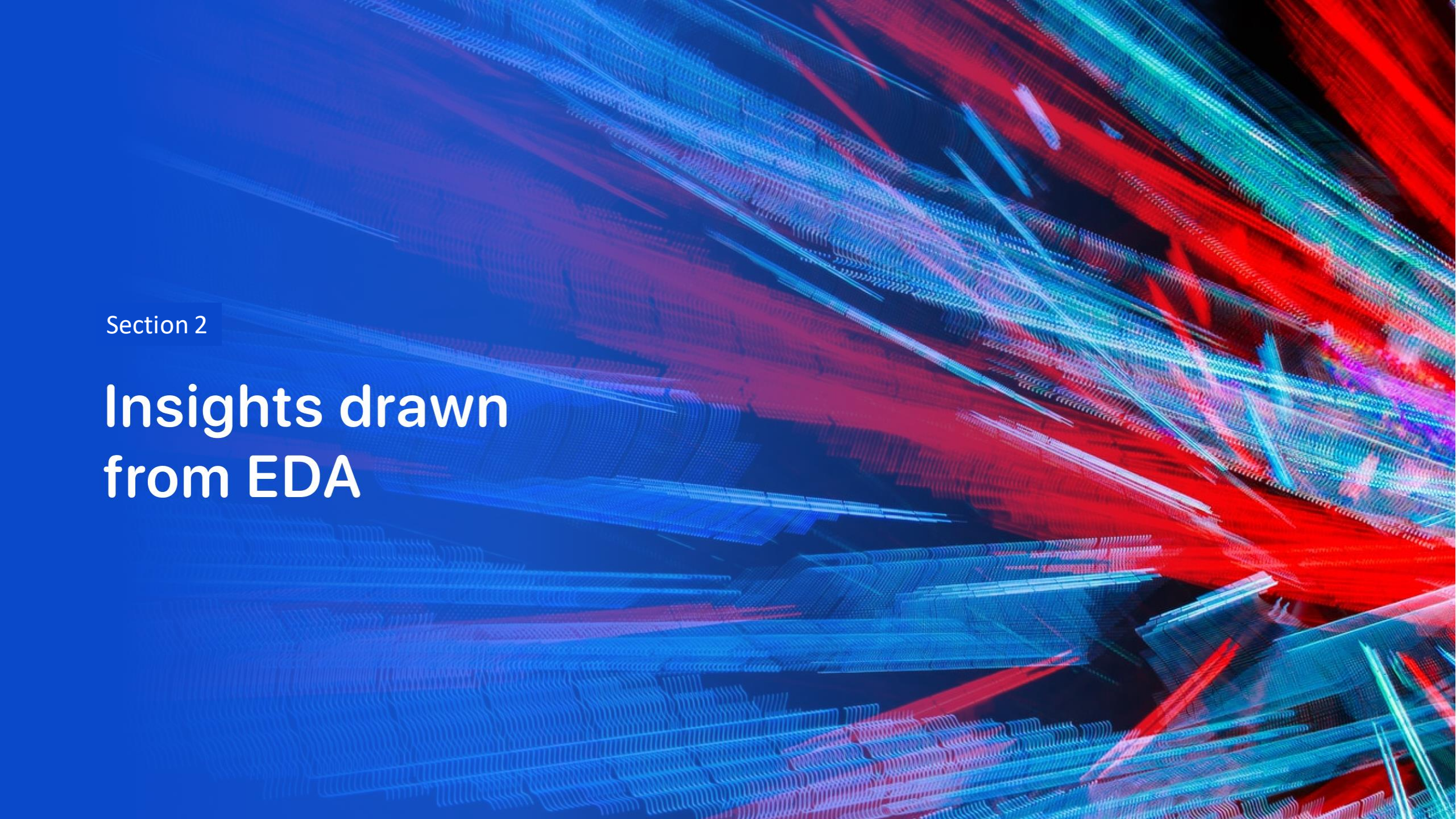
- Summarize how you built, evaluated, improved, and found the best performing classification model
  - PreProcessing and standardizing the data
  - Split the data applying the train-test split method
  - Test 4 different models utilizing grid search for parameter optimization
  - Checking accuracy of each model by
    - Comparing best parameter
    - Visualuze result via confusion matrix
  - Choose the model in line with best accuracy scores

Results

- Exploratory data analysis results
  - There are 4 launch sites used by SpaceX
  - Total Payload Mass (Kg) carried launched by Nasa is 45,596
  - Average payload mass (kg) carried by Booster version F9 v1.1 is 2,535 kg
  - The 1st successful landing by a drone ship were at 04/08/2016
  - We can specifically list the booster version in a given payload mass range which landed on a ground pad
  - We applied aggregate function to see nr of total success vs total failures: 61 vs 40
  - And we restricte result just for a given year to see subtotal of landing outcomes
- Predictive analysis results
  - Logistic regression, SVM and K-nearest neighbors, all of these three methods give the best performance, with accuracy of 0.83

Section 2

Insights drawn from EDA

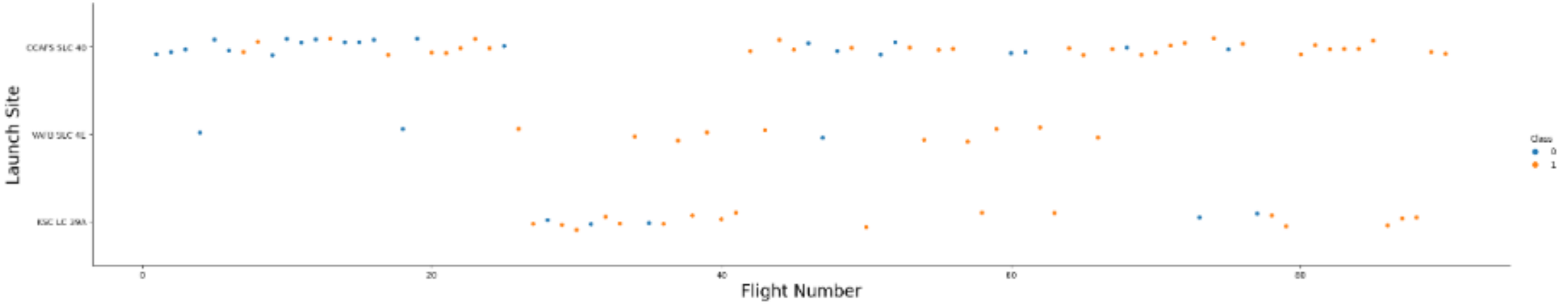


Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site
- Show the screenshot of the scatter plot with explanations

We can see that success rate increases in general tandem with flight number
We can follow how each launch site was utilized as flight numbers increase
We can observ that Vafb SLC 4E has the least failure porpotionately

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



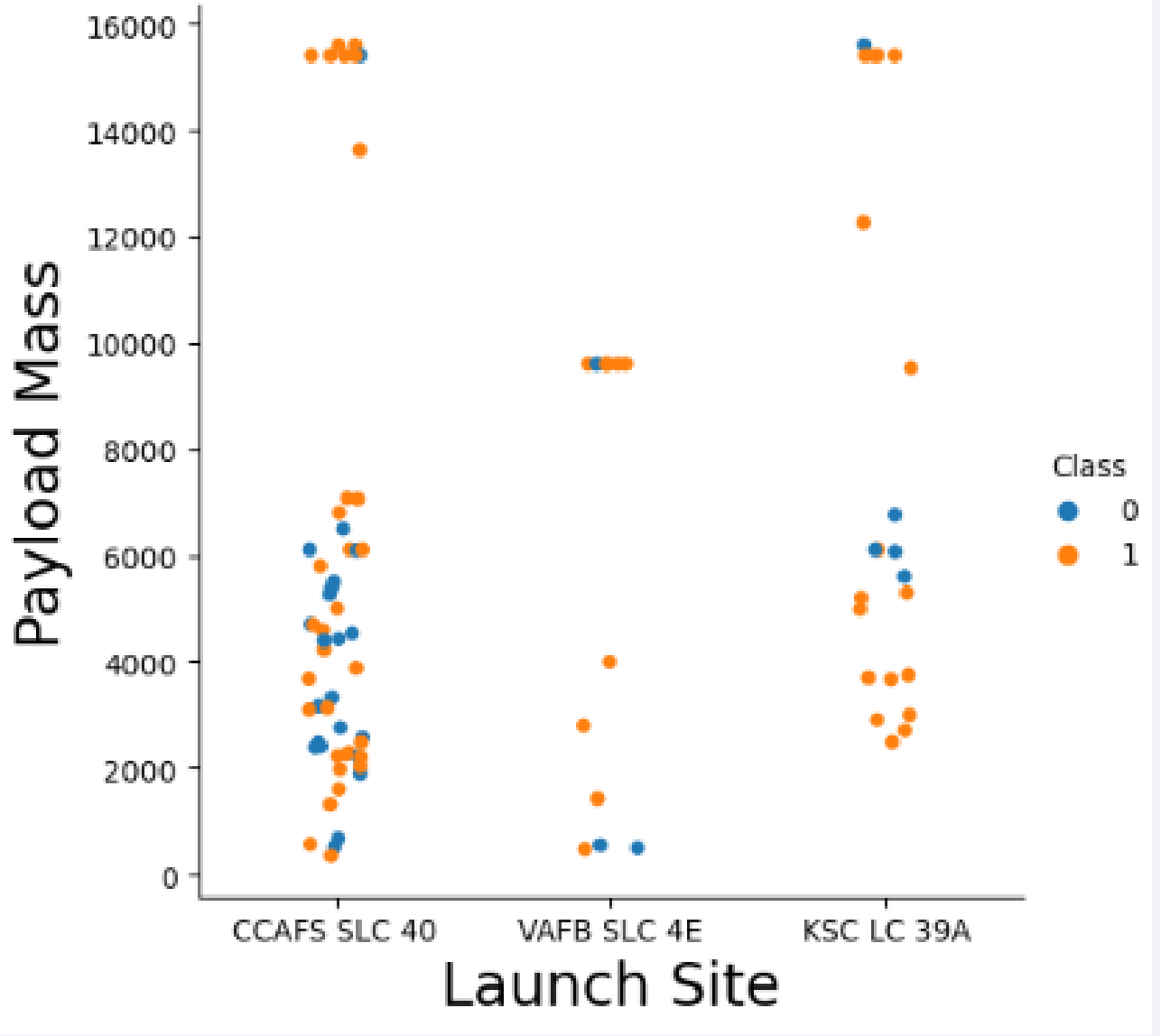
Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site

- Show the screenshot of the scatter plot with explanations

We can detect that as payloadmass increases likelihood of success improves
We can see that payloadmass above 10,000 are forming another cluster
Notice that VAFB SLC 4E is probably a smaller launc site as capping at 10,000
Notice that KSC LC 39A possess a stabile success rate for below 6,000 mass launches

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

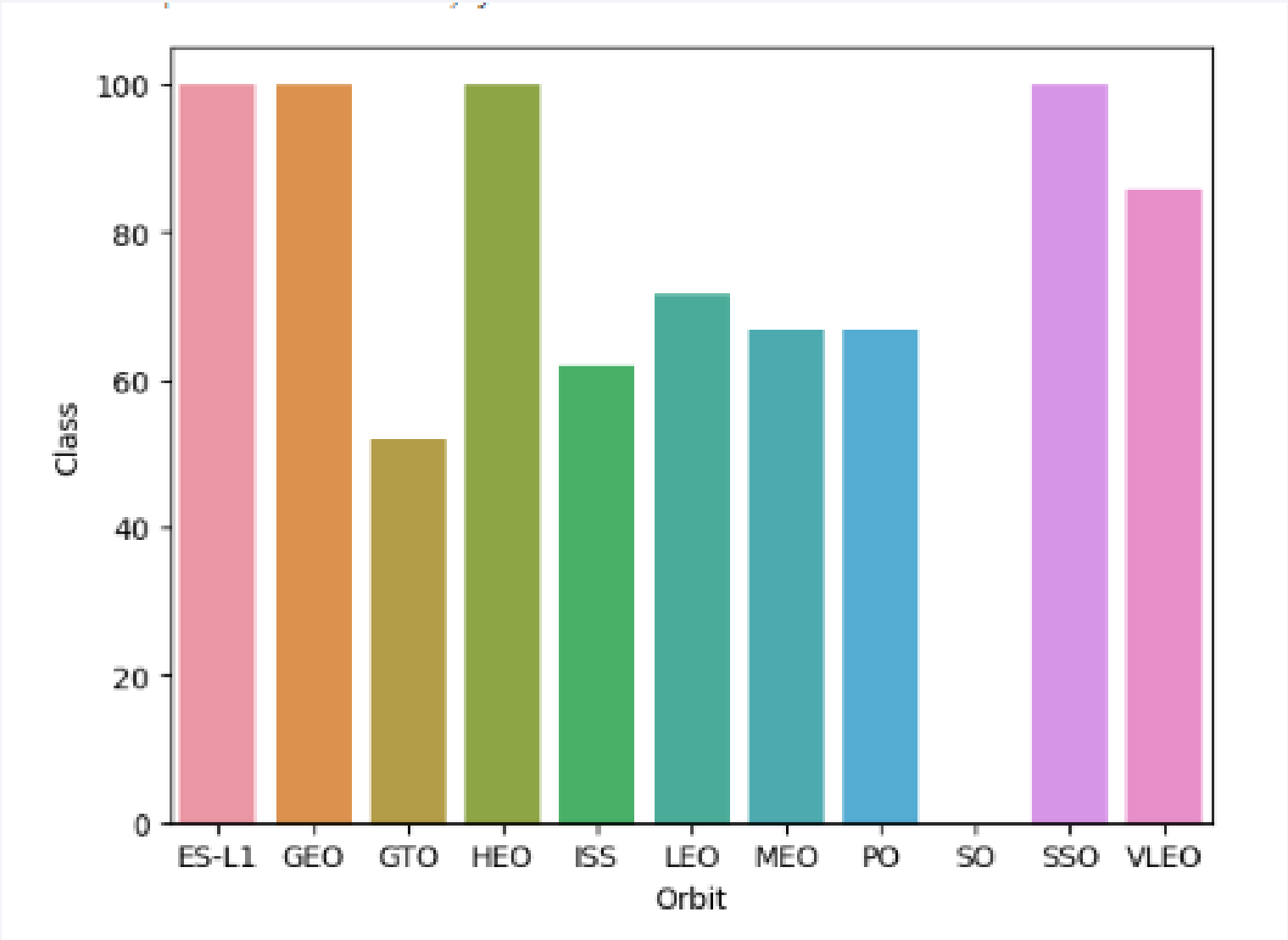


Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type

- Show the screenshot of the scatter plot with explanations

We can observe that the success rate of 4 orbits are outstanding: ES-L1; GEO; HEO; SSO
We can see that only 1 more orbit success rate is acceptable: VLEO
The rest of the orbits need further analyses to determine and validate root cause and escape route out of failure

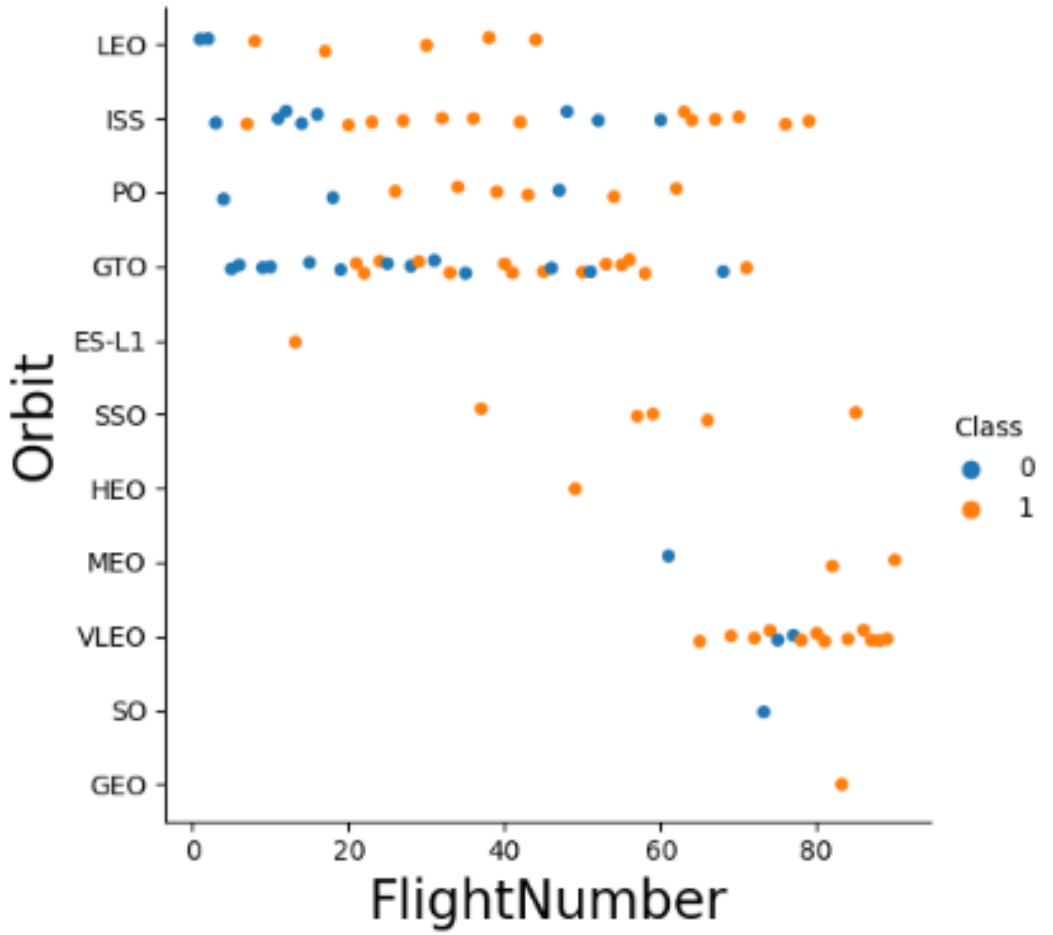


Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type
- Show the screenshot of the scatter plot with explanations

Spot that SSO has an outstanding track record of success rate
Out of the 2 most crowded launches toward ISS and GTO, GTO comes with higher uncertainty rate

```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be t
sns.catplot(x="FlightNumber", y="Orbit", data=df, hue = "Class")
plt.ylabel("Orbit",fontsize=20)
plt.xlabel("FlightNumber",fontsize=20)
plt.show()
```

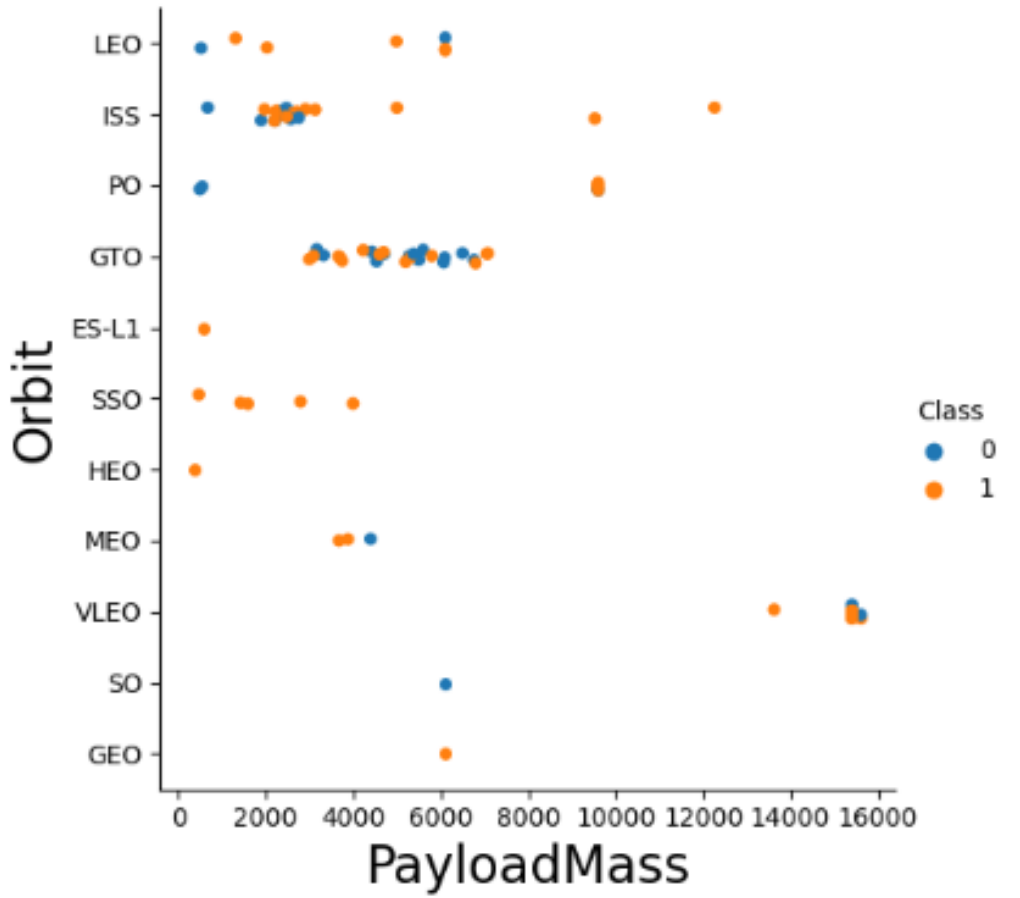


Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type
- Show the screenshot of the scatter plot with explanations

If we change a lens we can study that ISS targeted launches success rate improves as payload mass increases
The same is not true for GTO
Low payloadmass to SSO has the best likelyhood for success (especially if launched from KSC LC 39A)

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



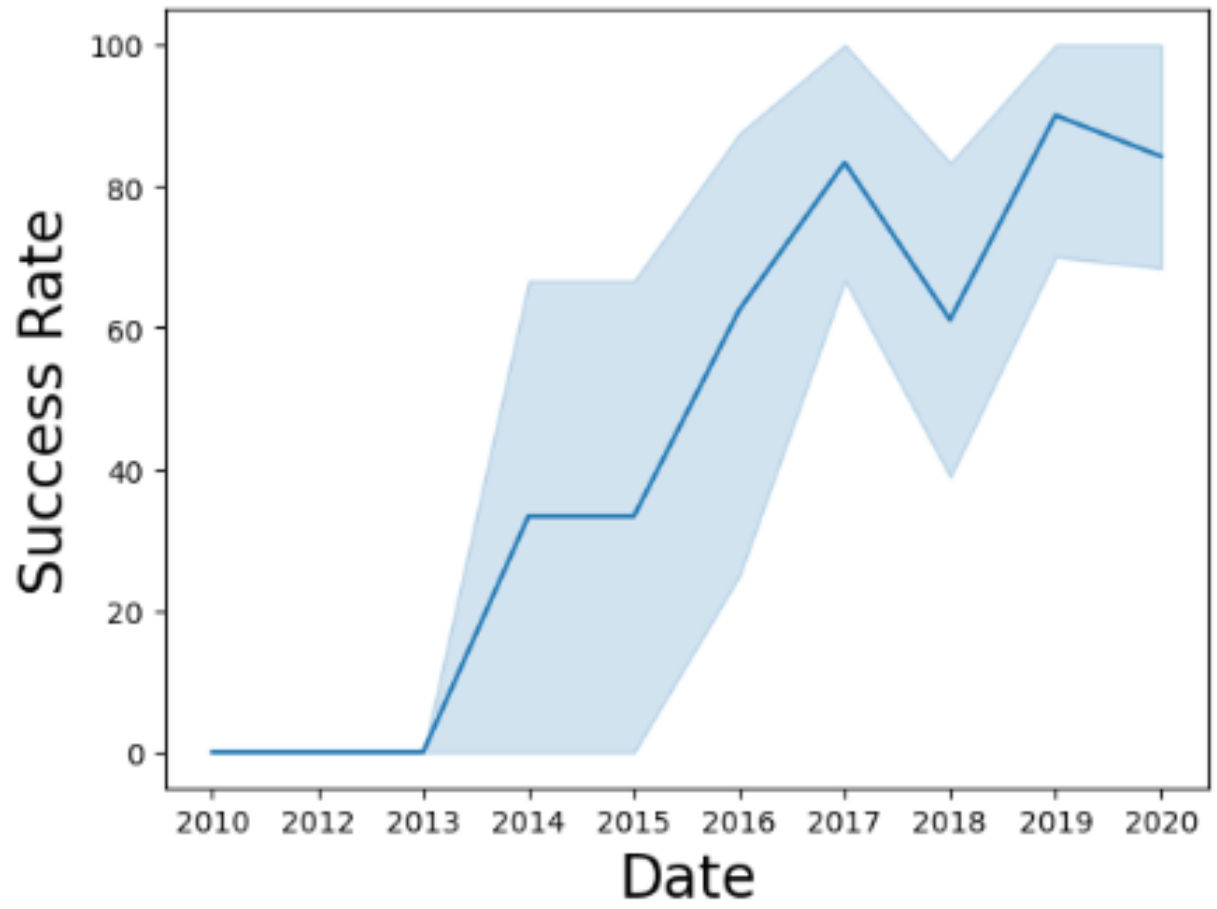
Launch Success Yearly Trend

- Show a line chart of yearly average success rate

- Show the screenshot of the scatter plot with explanations

Launch success clearly improves as time pass through trial and error, experience and innovation accumulated
The hiccup at 2018 deserves a closer look

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
df["Success Rate"] = df["Class"] * 100
sns.lineplot(x="Date", y="Success Rate", data=df)
plt.ylabel("Success Rate",fontsize=20)
plt.xlabel("Date",fontsize=20)
plt.show()
```



All Launch Site Names

- Query aims to list the unique launch sites present in the database

```
%%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

Launch Site Names Begin with 'KSC'

- Query uses wild card to find the launch sites which starts with the text "KSC"

```
%%sql
SELECT * FROM SPACEXTBL WHERE launch_site LIKE 'ksc%' LIMIT 5;
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
19/02/2017	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490.0	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
16/03/2017	6:00:00	F9 FT B1030	KSC LC-39A	EchoStar 23	5600.0	GTO	EchoStar	Success	No attempt
30/03/2017	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300.0	GTO	SES	Success	Success (drone ship)
05/01/2017	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300.0	LEO	NRO	Success	Success (ground pad)
15/05/2017	23:21:00	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070.0	GTO	Inmarsat	Success	No attempt

Total Payload Mass

- Here aggregate function used to get sum of payloadmass, restricted with 'Where' clause to Nasa customers

```
%%sql
select sum(payload_mass__kg_) as Total_paylomass_Nasa from SPACEXTBL where Customer = 'NASA (CRS)';
```

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

Total_paylomass_Nasa
45596.0

Average Payload Mass by F9 v1.1

- Another aggregation function utilized to get average payload mass restricted with 'where' clause to the ones carried by booster version F9 v1.1

```
%%sql
select avg(payload_mass__kg_) as "Avg_pylm_F9_v1.1" from SPACEXTBL where booster_version like 'F9 v1.1%';
```

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

Avg_pylm_F9_v1.1
2534.6666666666665

First Successful Ground Landing Date

- 'Min' function used to reach the earliest date restricted by 'where' clause to specify it is within Success – drone ship

```
%%sql
select min(DATE) as "1st date drone ship landed" from SPACEXTBL WHERE landing_outcome = 'Success (drone ship)';
```

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

1st date drone ship landed
04/08/2016

Successful Drone Ship Landing with Payload between 4000 and 6000

- At this time 2 restriction have been phrased within the where clause, targeted payload mass range specified by the 'between' clause

```
%%sql
select booster_version from SPACEXTBL where landing_outcome = 'Success (ground pad)' and payload_mass__kg_ between 4000 and 6000;
```

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

Booster_Version
F9 FT B1032.1
F9 B4 B1040.1
F9 B4 B1043.1

Total Number of Successful and Failure Mission Outcomes

- We divide landing outcome into 2 brackets via utilization of wild card by applying combination of 'where' with 'Like' clause
- Union All command applied to concatenate results into 1 table

```
%%sql
SELECT 'Success' AS "Outcome", count(*) AS "Count" FROM SPACEXTBL WHERE landing_outcome LIKE 'Success%'
UNION ALL
SELECT 'Failure' AS "Outcome", count(*) AS "Count" FROM SPACEXTBL WHERE landing_outcome NOT LIKE 'Success%'
UNION ALL
SELECT '(All)' AS "Outcome", count(*) AS "Count" FROM SPACEXTBL;
```

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

Outcome	Count
Success	61
Failure	40
(All)	999

Boosters Carried Maximum Payload

- Booster versions with max payload mass listed applying subquery or nested query because aggregations cannot be derived from where clause

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

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

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

2015 Launch Records

- List the records which will display the month names, successful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017
  - No success on generating subquery for data column
- Present your query result with a short explanation here

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

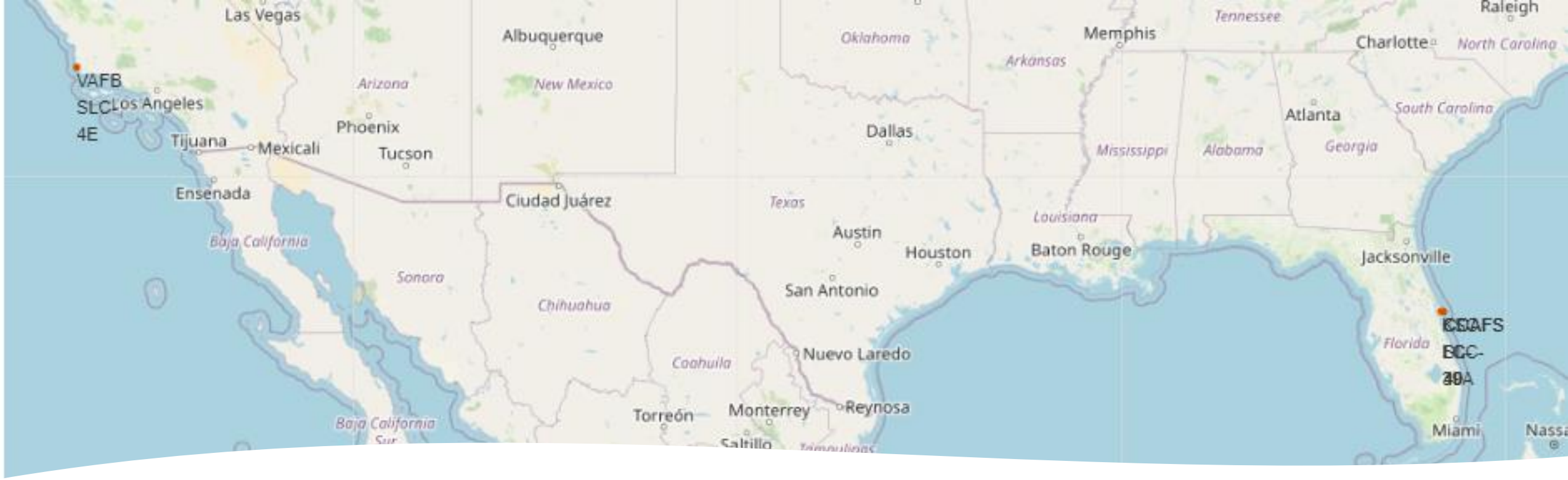
```
%%sql

SELECT landing_outcome, COUNT(*) AS "Count"
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' and '2017-03-20'
GROUP BY landing_outcome
ORDER BY Count DESC
```

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

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

- A summary table has been generated using the groupby function
  - for a given date range using the where clause
  - Ranking ensured using ordered by to descending order


Section 3
Launch Sites
Proximities Analysis



Launch Site positions generated with Folium

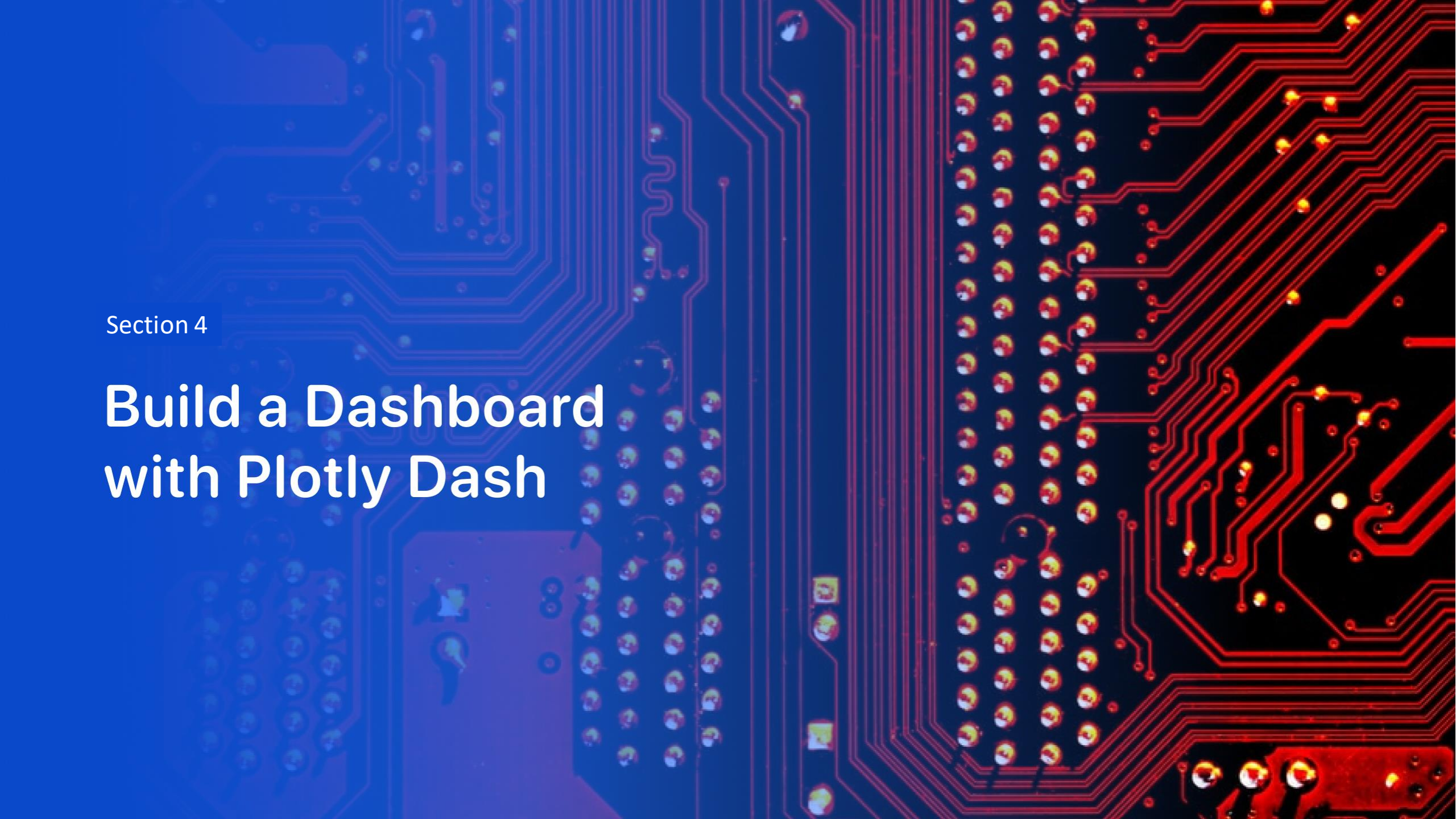
- We can observe that 2 launch sites are relatively close to each other on the east coast while there is 1 launch site on the west coast (WAFB SLC 4E)

Color coding Success vs Unsuccessful launches

- If we observe CCAFS SLC-40 launce site from the right distance there are clear division of 26 unsuccessful and 7 successful launches from that location

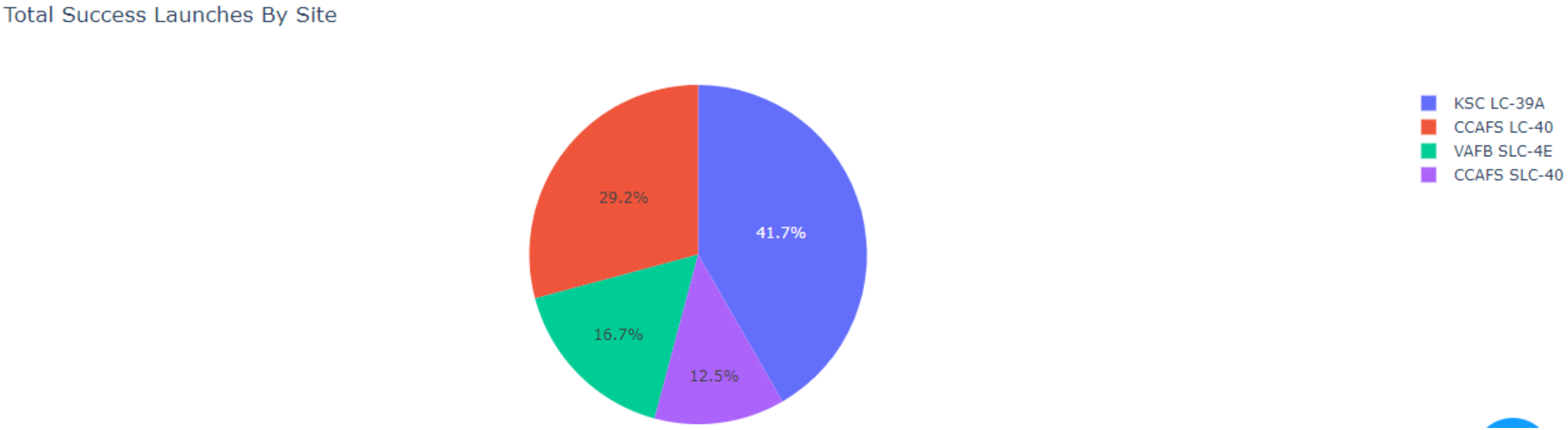
Displaying distance to certain objects on the map with Folium

- Here we can calculate and display the distance between the launch site and the coast with a supporting line.


Section 4
Build a Dashboard with Plotly Dash

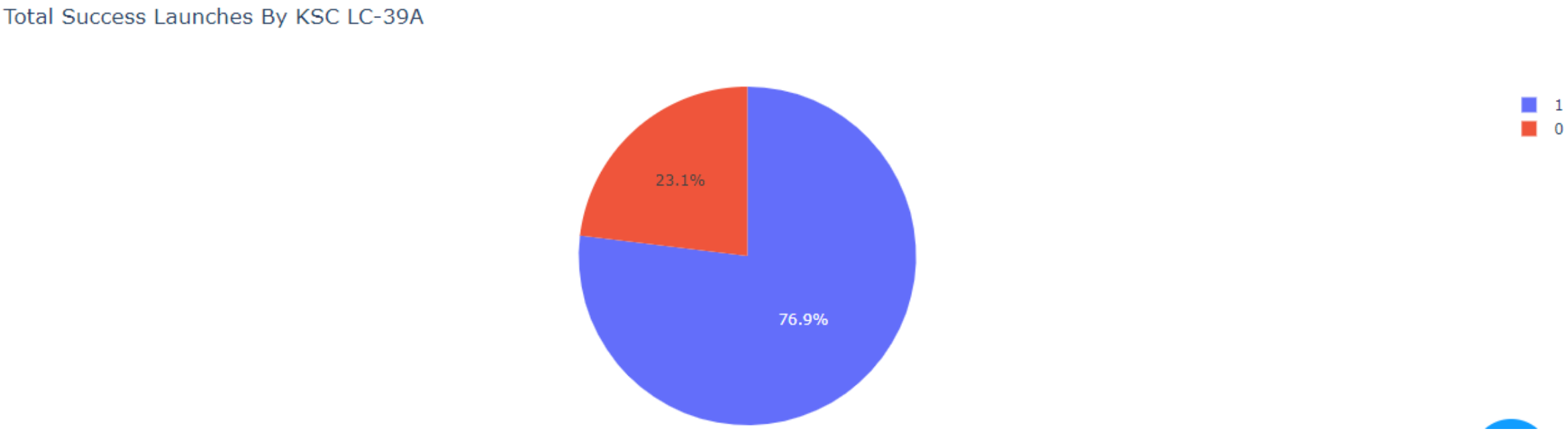
Contribution to Success launches by launch Site

- We can clearly observe that the majority of successful launches are from the site: **KSC LC-39A**
- CCAFS LC40 also possess a good success rate being as successful as the last 2 sites



Success rate within KSC LC-39A

- We can see that the best score hits the ground successfully in three quarters of the cases



Payloadmass and Booster version

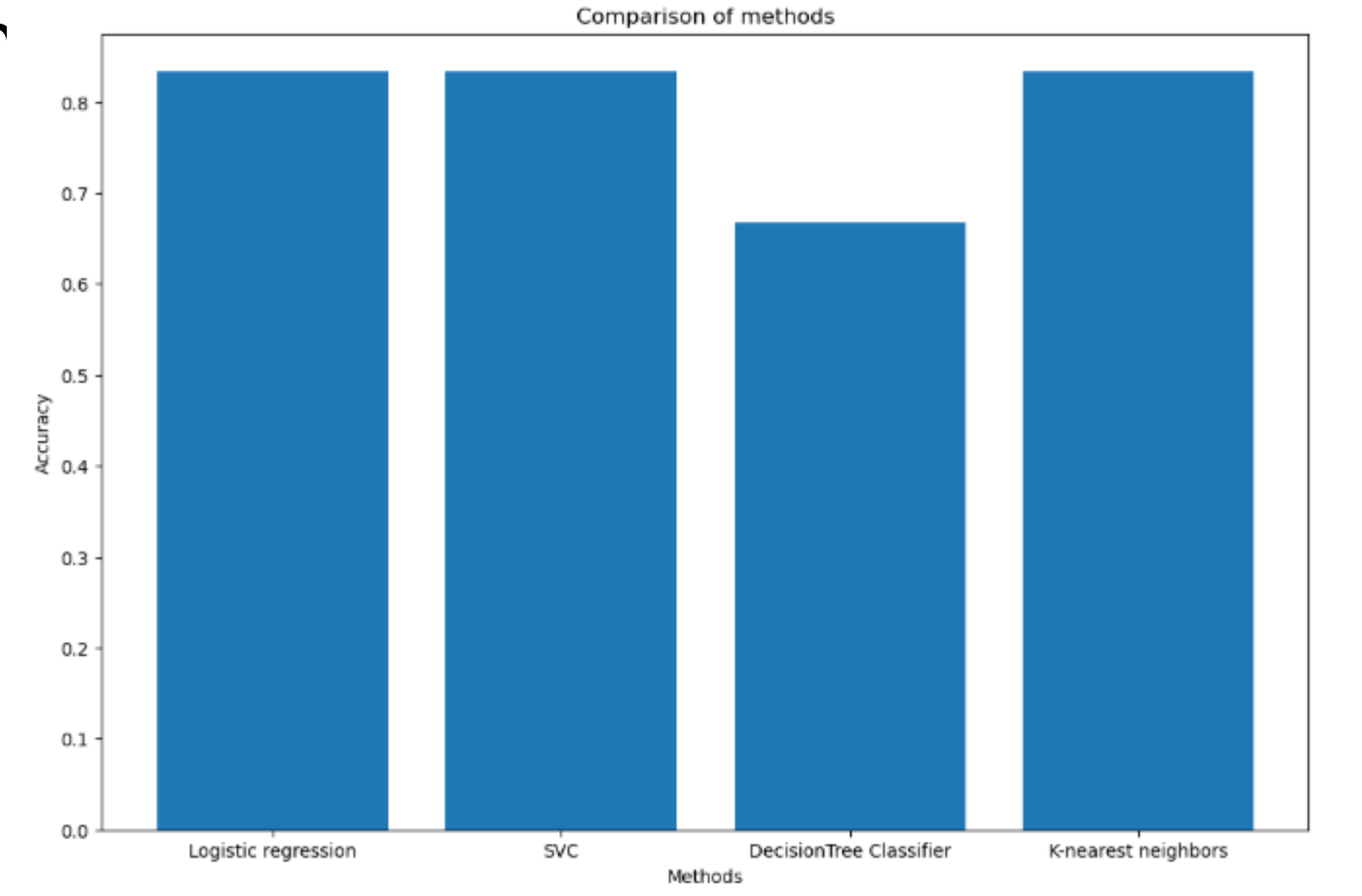
- We can observe that the majority of the successful launches fall within payload mass range of 2-6 tons
- Regarding Booster version FT and B4 are the most prone to success
- Above 5.3 ton mass load success is highly unlikely regardless of booster version
- FT launches have a very stable success ratio between payload mass of 2.2-3.2 tons
- Whereas B4 Booster stabilizes in higher mas range apx 3.3 to 5 tons

Section 5

Predictive Analysis (Classification)

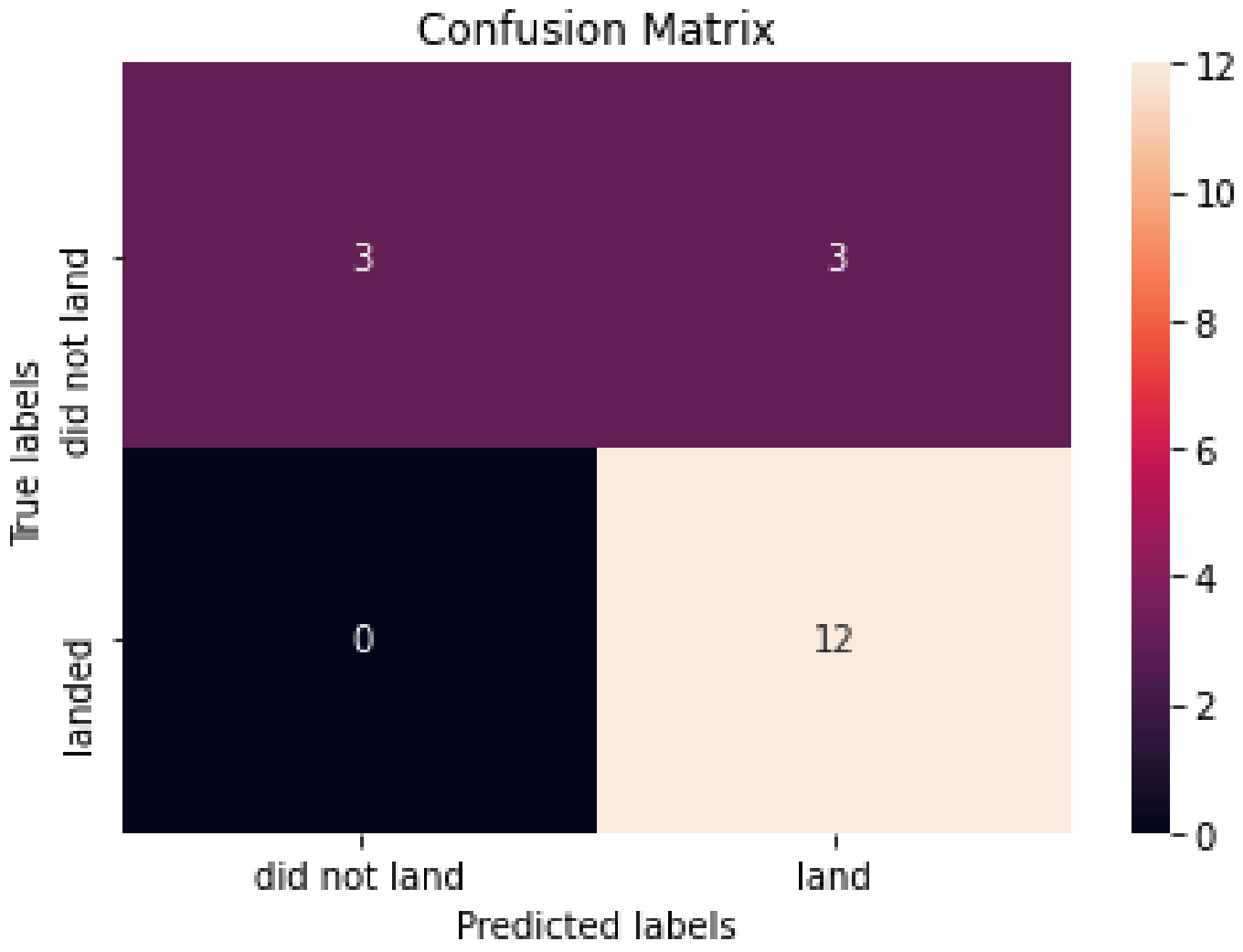
Classificatior Accuracy

- Logistic regression, SVM and K-nearest neighbors, all of these three methods give the best performance, with accuracy of 0.83
- Only exception is the decision tree classifier



Confusion Matrix

- Picked confusion matrix is the logistic regression one, (the other 2 with highest score are basically identical)
- We can see that the model does a very good job predicting the failure, we can take it granted once it predicts unsuccessful landing
- In case the model predicts success it is expected to be failure in one quarter of the cases



Conclusions

- Parameters need to be fine tuned for rockets launched for different purposes (depending on payload mass or target orbit for instance)
- The most secure landing outcome can be characterized by the following attributes:
  - Initial launch: **KSC LC-39A,** Payload Mass in range of: **below 6000 or above 8000,** Booster version: FT or B4
  - Regarding target orbit:
    - For the lower range of payload mass SSO possess and outstanding success rate
    - While on the high payload mass range more data needed to stress the statement that orbits ISS or PO are secure
- Although there are 3 model shows similarly good fit for our problem constant evaluation and test is need going forward as more and more data is available

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!