IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Mihir Damania
12/04/2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection:
    - Utilized SpaceX API and web scraping to gather rocket launch data and Falcon 9 historical launch records.
    - Standardized data formats and normalized JSON responses into DataFrames for analysis.
  - Data Wrangling:
    - Analyzed launch sites, orbits, and mission outcomes.
    - Classified landing success to evaluate the success of the landing stage.
  - EDA and Data Visualization:
    - Employed various plots to visualize relationships between flight number, payload mass, launch outcomes, and orbit types.
    - Utilized SQL queries to extract insights about launch sites, payload mass, mission outcomes, and landing success.

Executive Summary

- Interactive Map with Folium:
  - Mapped SpaceX launch sites and nearby features to identify optimal launch locations.
  - Visualized launch outcomes and distances to key features using markers, circles, and polylines.
- Dashboard with Plotly Dash:
  - Created an interactive dashboard allowing users to explore launch success and payload mass outcomes.
  - Implemented features such as dropdown menu, pie charts, scatter plots, and payload mass range selector.
- Predictive Analysis (Classification):
  - Standardized features and split data into training and testing sets.
  - Tuned machine learning models using GridSearchCV with 10-fold cross-validation.
  - Evaluated model effectiveness using accuracy and confusion matrix to predict first stage reuse.

Executive Summary

- Summary of all results
  - Flight Number and Launch Success:
    - Initial launches at CCAFS SLC-40 had few successes, indicating early failures common in space missions.
  - Payload Mass and Launch Site Optimization:
    - VAFB-SLC site avoided for payloads >10,000kg, highlighting the importance of dedicated launch sites for different payloads.
  - Orbit Success Rates:
    - ES-L1, GEO, HEO, and SSO orbits show 100% success rates, suggesting focusing on these orbits initially for early successes.
  - Success Rate Trends:
    - Increasing flight experience led to higher success rates, particularly noticeable in LEO orbits, which are easier to achieve success initially.
  - Payload Mass Impact:
    - Higher payload masses correlate with increased success for LEO, ISS, and PO orbits, with most heavy payloads launched to VLEO.

Executive Summary

- Benefits of Multiple Launch Sites:
  - Having multiple launch sites such as CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, and CCAFS SLC-40 allows flexibility for different launch vehicles.
- Payload and Landing Insights:
  - F9 v1.1's average payload mass is 2,534.67kg, with specific boosters successfully landing on a drone ship carrying payloads between 4,000 and 6,000kg.
- Maximum Payload Mass and Location Optimization:
  - F9 B5 B1048.4 carried the maximum payload mass of 15,600kg, highlighting the importance of launch site locations near the equator and coastline.
- Overall Launch Success and Model Accuracy:
  - Total launch success from all sites is 42.9%, with KSC LC-39A showing the highest success ratio at 76.9%.
  - Classification models achieved an accuracy of 83% in predicting launch outcomes, indicating reliable performance in classifying labels.

Introduction

- Commercial space age is booming with companies like Virgin Galactic, Rocket Lab, Blue Origin, and SpaceX making space travel more accessible.
- SpaceX's achievements include sending spacecraft to the International Space Station, launching Starlink, and conducting manned missions to space.
- SpaceX's Falcon 9 rocket launches are relatively inexpensive compared to other providers, primarily due to reusability of the first stage.
- Falcon 9 launch involves the first stage, which does most of the work, and the second stage, which assists in reaching orbit.
- Unlike other providers, SpaceX can recover the first stage, although there are instances where it may not successfully land.
- Objective of this project is to determine the success rate of each launch by analyzing SpaceX's launch history and creating dashboards for your team.
- Additionally, you will use machine learning to predict whether SpaceX will land the first stage successfully, using publicly available data.
- This approach offers an alternative to traditional rocket science methods for predicting first stage landing success.
- The ultimate goal is to provide insights into launch costs and reusability for informed decision-making in the space industry.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Fetch rocket launch data using SpaceX REST API, normalize into DataFrame, and retrieve additional data for analysis.
  - Scrape Wikipedia, extract launch data, and create DataFrame for analysis.
- Perform data wrangling
  - Utilized various pandas methods such as value_counts() to analyze launch sites, orbits, and mission outcomes. Also created landing_class list to classify landing success based on specific outcomes.
- 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 features using StandardScaler library, split data into train and test set, tune models using GridSearchCV and evaluate accuracy using confusion matrix.

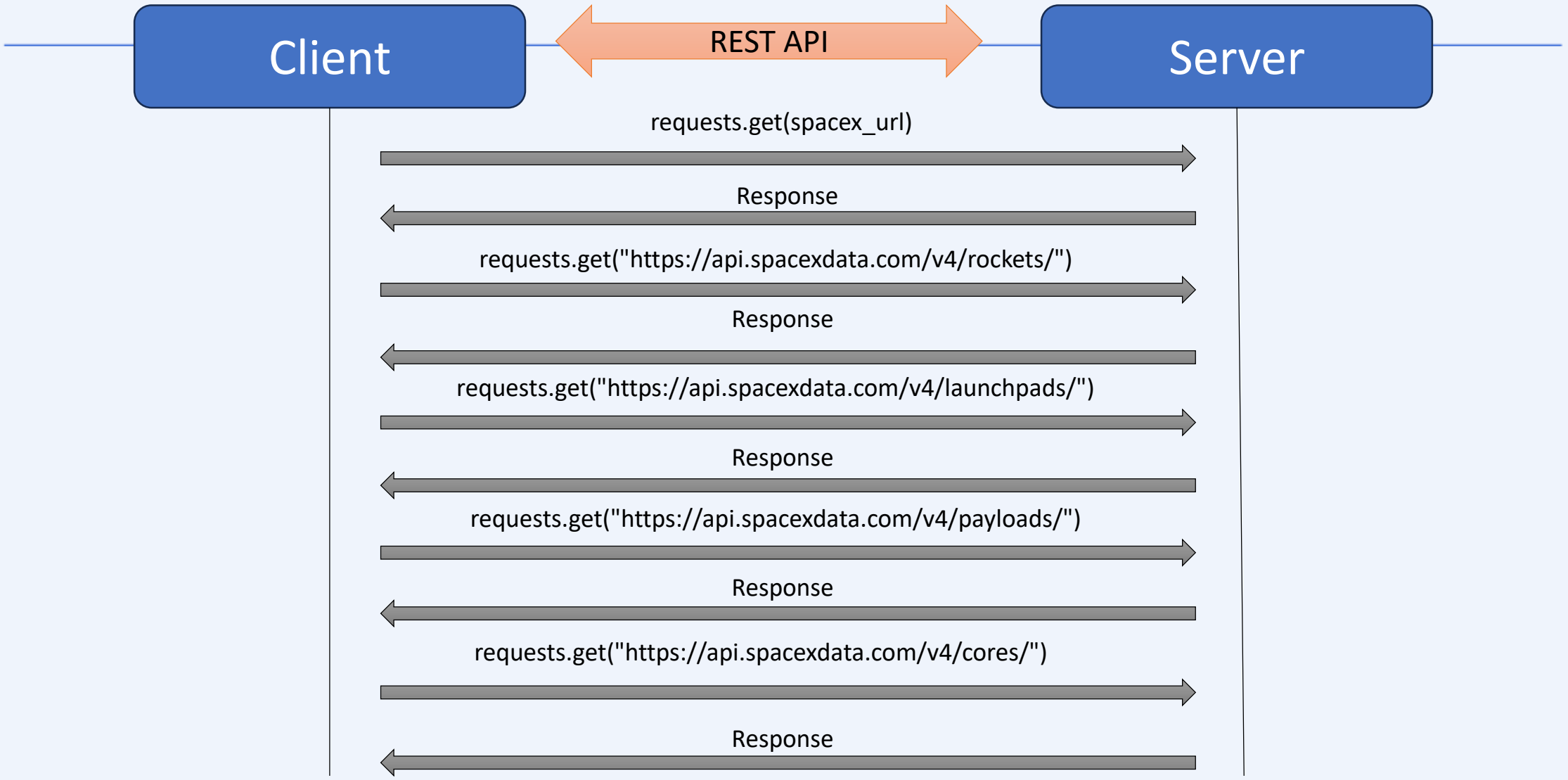
Data Collection

- Data Collection from SpaceX API:
  - Import libraries and define functions for API interaction.
  - Fetch rocket launch data and ensure successful retrieval.
  - Decode JSON response and normalize data into a DataFrame.
  - Select essential columns and convert date columns.
  - Retrieve additional data for each ID and compile into a DataFrame.
- Web Scraping for Falcon 9 Launch Records:
  - Scrape Wikipedia for Falcon 9 launch data.
  - Retrieve HTML content and parse with BeautifulSoup.
  - Identify and select launch table for analysis.
  - Extract and clean column names.
  - Create dictionary with column names and populate with launch records finally convert dictionary into a DataFrame.

Data Collection – SpaceX API

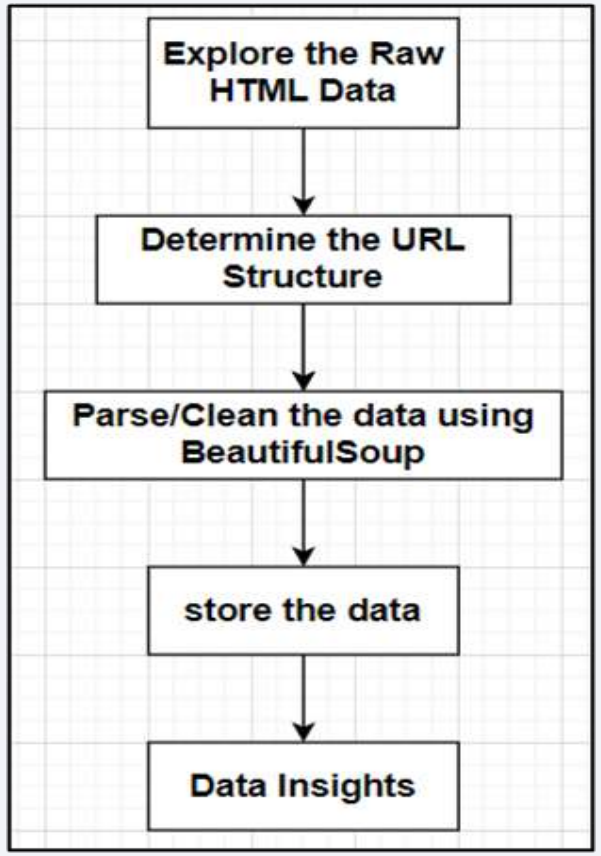
- Importing Request and necessary Libraries & predefining some functions to interact with the SpaceX API using specific identification numbers.
- API Data Retrieval
  - Fetch rocket launch data using requests.get() from the SpaceX API endpoint.
- Check the response status code to ensure successful data retrieval and preview the response content to understand the initial data structure.
- JSON Decoding & Data Normalization
  - Decode the JSON response and use pd.json_normalize() to flatten the JSON structure into a DataFrame.
- Feature Selection - Filter the DataFrame to include only essential columns required for the analysis.
- Convert date columns to datetime type and restrict the dataset to records up until November 2020.
- Data Extraction & Compilation
  - Using predefined functions to retrieve additional data via API for each ID.
  - Append results to a list, create a dictionary, and convert it to a DataFrame for comprehensive analysis.
- Github-Link for Jupyter notebook of data collection from SpaceX api.

SpaceX REST API calls flowchart



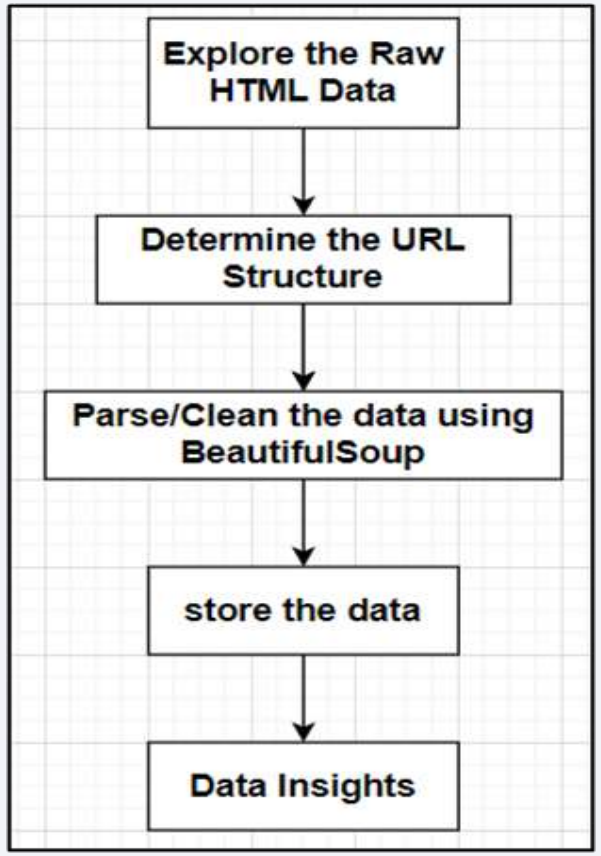
Data Collection - Scraping

- Web scraping a Wikipedia page to collect Falcon 9 historical launch records.
- Request Falcon 9 Launch Page
  - Perform an HTTP GET request to retrieve the Falcon 9 Launch HTML page using requests.get().
  - Load the HTML content into BeautifulSoup for parsing.
- Locate HTML Tables
  - Use BeautifulSoup to find all tables embedded within the webpage.
  - Identify and select the specific launch table required for analysis.



Data Collection - Scraping

- Extract Column Names
  - Iterate through table headers (<th> tags) within the selected launch table.
  - Apply a function to extract and clean the column names from these headers.
- Prepare Data Collection Structure-Create an empty dictionary where keys correspond to the extracted column names.
- Parse Table Data = Extract launch records from table rows and fill the dictionary with corresponding data.
- Create DataFrame= Convert the populated dictionary into a pandas DataFrame for further analysis and visualization.
- GitHub-Link for Jupyter notebook of data collection using webscraping.



Data Wrangling

- Data Analysis on Launch Sites
  - Use df['LaunchSite'].value_counts() to calculate the number of launches at each site.
- Analysis of Orbits
  - Calculate the occurrence of each orbit type using df["Orbit"].value_counts().
- Mission Outcomes by Orbit
  - Determine mission outcomes using landing_outcomes = df['Outcome'].value_counts().
  - Identify bad outcomes by selecting specific indices: bad_outcomes = set(landing_outcomes.keys()[[1,3,5,6,7]]).
- Landing Success Classification
  - Create landing_class list: Assign 0 if outcome is in bad_outcomes, 1 otherwise.
  - This is applied to evaluate the success of the landing stage.
- GitHub-Link for Data Wrangling Jupyter notebook.

EDA with Data Visualization

- Plotted a scatter chart to observe the relationship between FlightNumber and PayloadMass overlaid with launch outcomes.
- Used a categorical plot (Catplot) to visualize how the flight number correlates with different launch sites.
- Categorical scatter plot to analyze the payload mass handled by different launch sites.
- Bar plot visualizing the success rate across different orbit types.
- Scatter plot to explore the relationship between flight number and orbit type.
- Scatter plot assessing how payload mass affects success across different orbits.
- Line chart displaying the average yearly success rate from 2013 to 2020.
- GitHub-Link for Jupyter notebook of EDA with data visualization.

EDA with SQL

- To display the names of the unique launch sites in the space mission
  - %sql SELECT DISTINCT Launch_Site from SPACEXTABLE ;
- For 5 records where launch sites begin with the string 'CCA'
  - %sql select * from SPACEXTABLE WHERE Launch_Site like 'CCA%' limit 5;
- To get the total payload mass carried by boosters launched by NASA (CRS)
  - %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE WHERE Customer like 'NASA (CRS)';
- To display average payload mass carried by booster version F9 v1.1
  - %sql select avg(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version like 'F9 v1.1%';
- Date of first successful landing outcome in ground pad was achieved.
  - %sql select MIN(Date),Landing_Outcome from SPACEXTABLE WHERE Landing_Outcome like 'Success%';
- Listing 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,PAYLOAD_MASS__KG_ from SPACEXTABLE WHERE Landing_Outcome like 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_<6000;

EDA with SQL

- The total number of successful and failure mission outcomes
  - %sql select Mission_Outcome,COUNT(*) as Total_Number from SPACEXTABLE group by Mission_Outcome;
- Using subquery function listing the names of the booster_versions which have carried the maximum payload mass.
  - %sql select Booster_version,MAX(PAYLOAD_MASS__KG_) from SPACEXTABLE;
- List of records which display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
  - %sql SELECT substr(Date, 6, 2) as Month,Landing_Outcome,Booster_version,Launch_site FROM SPACEXTABLE WHERE substr(Date, 0, 5) = '2015' AND Landing_Outcome LIKE 'Failure (drone ship)';
- Ranking the count of landing outcomes between the date 2010-06-04 and 2017-03-20, in descending order.
  - %sql SELECT Landing_Outcome,COUNT(*) AS outcome_count FROM SPACEXTABLE WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY outcome_count DESC;
- GitHub-Link for Jupyter notebook of EDA with SQL.

Build an Interactive Map with Folium

- There are many factors that contribute to a successful launch, one of which is the location of the launch site. Objective is to identify the optimal location for the launch site by mapping SpaceX launch sites and the distances to their proximities.
- Creating a folium Map object, with an initial center location as NASA Johnson Space Center at Houston, Texas.
- Next using `folium.Circle` to add a highlighted circle area with a text label on a specific coordinate.
- Creating and add `folium.Circle` and `folium.Marker` for each launch site on the site map.
- Adding the launch outcomes for each site using marker cluster, and see which sites have high success rates.
- Adding a `MousePosition` function on the map to get coordinate for a mouse over a point on the map.
- Calculating distances between the launch site and nearby important features using coordinates, then add polylines on the map to indicate the nearest coastline, city, railway line, and highway.
- GitHub-Link for Jupyter notebook of interactive map with folium.

Build a Dashboard with Plotly Dash

- Interactive Dropdown Feature
  - Includes a dropdown menu at the top of the interface, allowing users to select a specific launch site or choose to view cumulative results for all sites.
- Launch Success Visualization
  - The first chart is a pie chart displaying the percentage of total successful launches, which can be viewed for all launch sites collectively or for a selected individual site.
- Payload Mass vs. Launch Outcomes
  - The second visualization is a scatter plot that illustrates the relationship between payload mass and launch outcomes. This plot differentiates various booster versions using distinct colors.
- Payload Mass Range Selector
  - Features a slider to adjust the payload mass range, enabling users to explore the relationship between payload mass and launch outcomes across different payload mass intervals.
- GitHub-Link of Jupyter notebook for plotly dashboard.

Predictive Analysis (Classification)

- The target variable was converted into a numpy array and the feature variables were standardized using the StandardScaler to normalize the data, ensuring that the model evaluates each feature uniformly.
- The dataset was divided into a training set and a testing set using the train_test_split function, which helps in assessing the model's performance on unseen data.
- A dictionary named parameters containing potential hyperparameters for tuning was created. A machine learning model (such as Logistic Regression, SVM, Decision Tree, KNN) was instantiated. GridSearchCV was set up to perform exhaustive search over specified parameter values, using 10-fold cross-validation. This process involves splitting the training data into 10 parts, using 9 for training and 1 for validation, iteratively.
- The GridSearchCV object was fitted to the training data to determine the most effective parameters. The best combination of parameters is obtained from model.best_params_, and the best accuracy from cross-validation is recorded through model.best_score_.
- The model's effectiveness is then tested using the reserved test data, employing predictions to compute the model’s accuracy and visualize its performance through a confusion matrix, which displays the correct and incorrect classifications for each class.
- GitHub-Link for the jupyter notebook of predictive analysis.

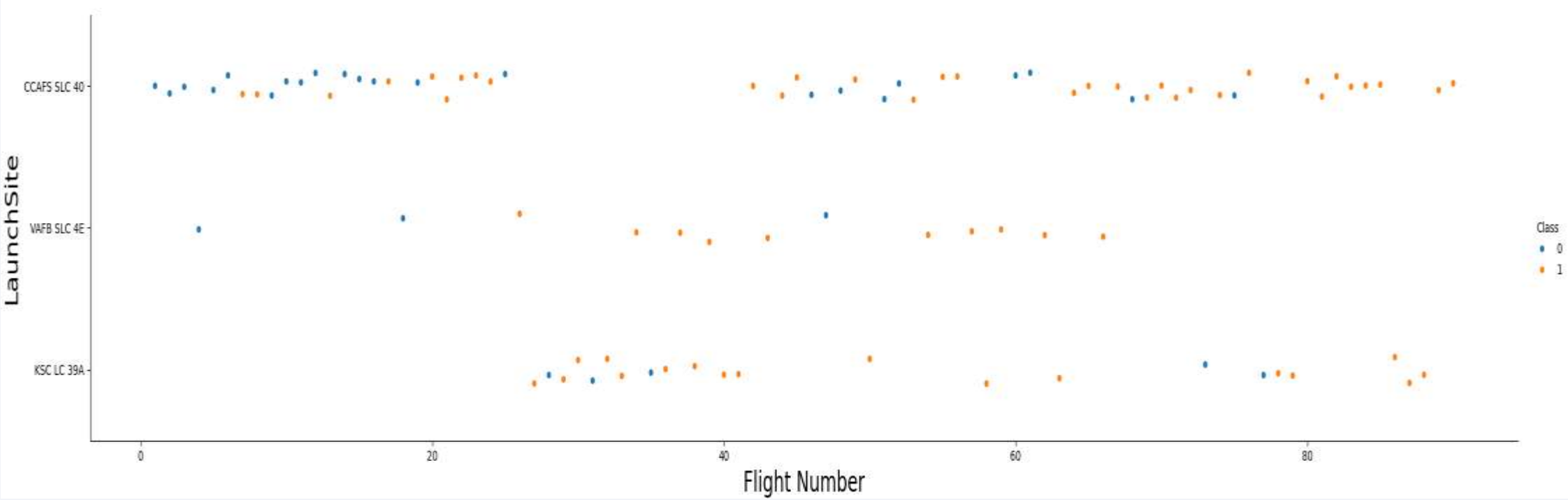
Results

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

Section 2

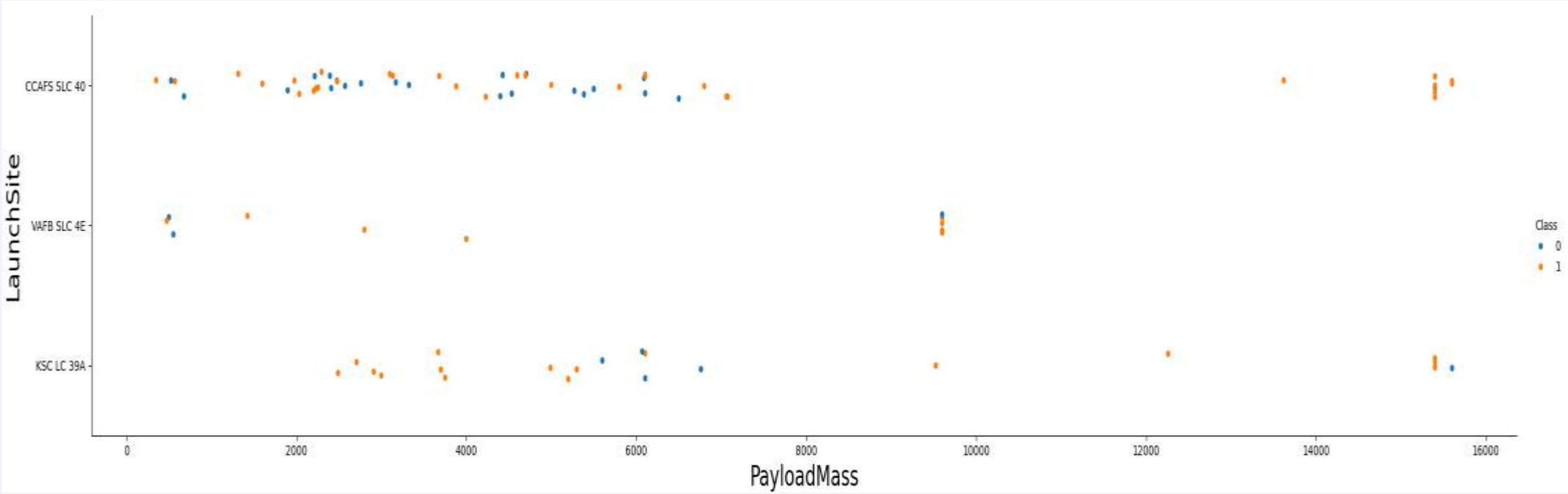
Insights drawn from EDA

Flight Number vs. Launch Site



We can see that most of the earlier launch attempts were made from CCAFS LC-40 in which few were successful. Furthermore as the Flight Number increases the number of successful attempts also increases.

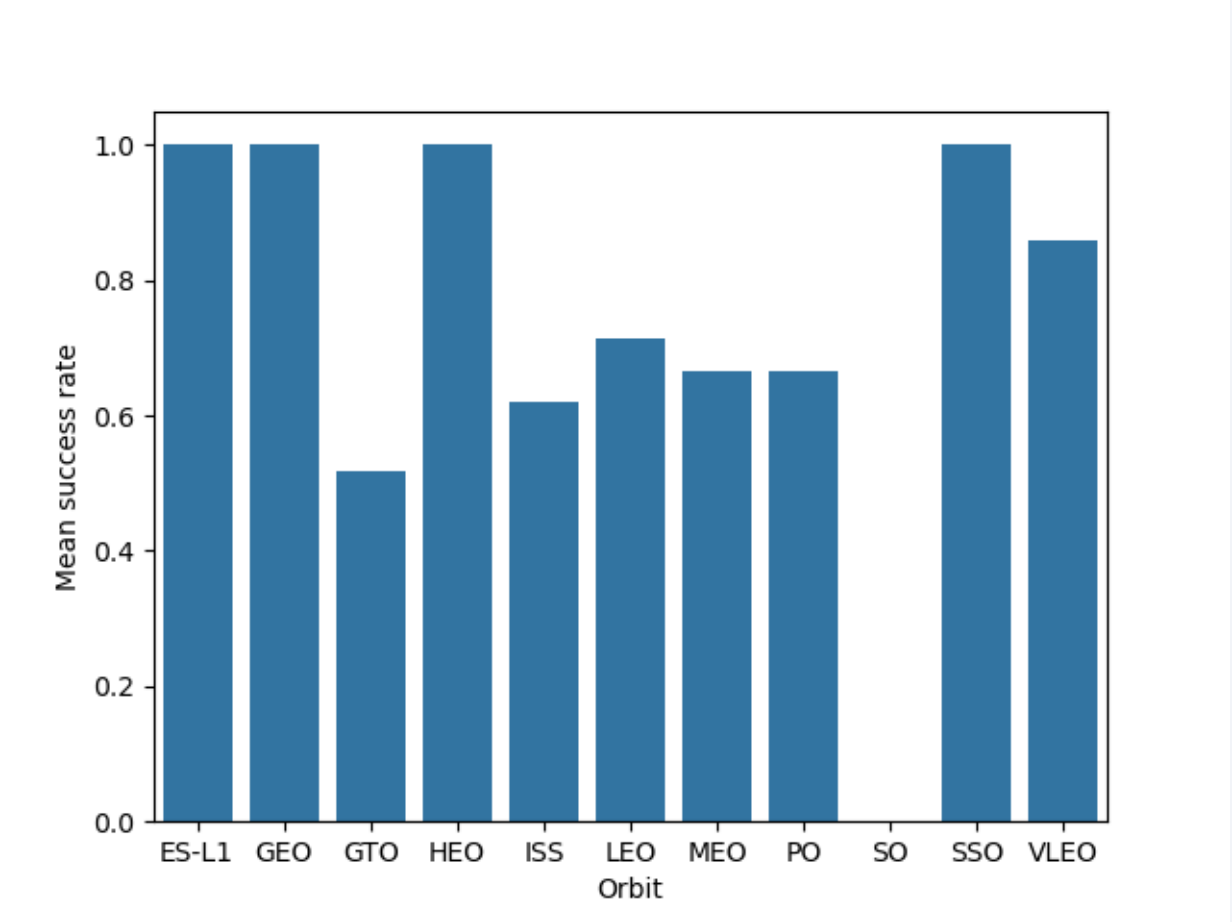
Payload vs. Launch Site



In this scatter plot, for the VAFB-SLC launchsite there are no rockets launched for heavy payload mass(greater than 10000).

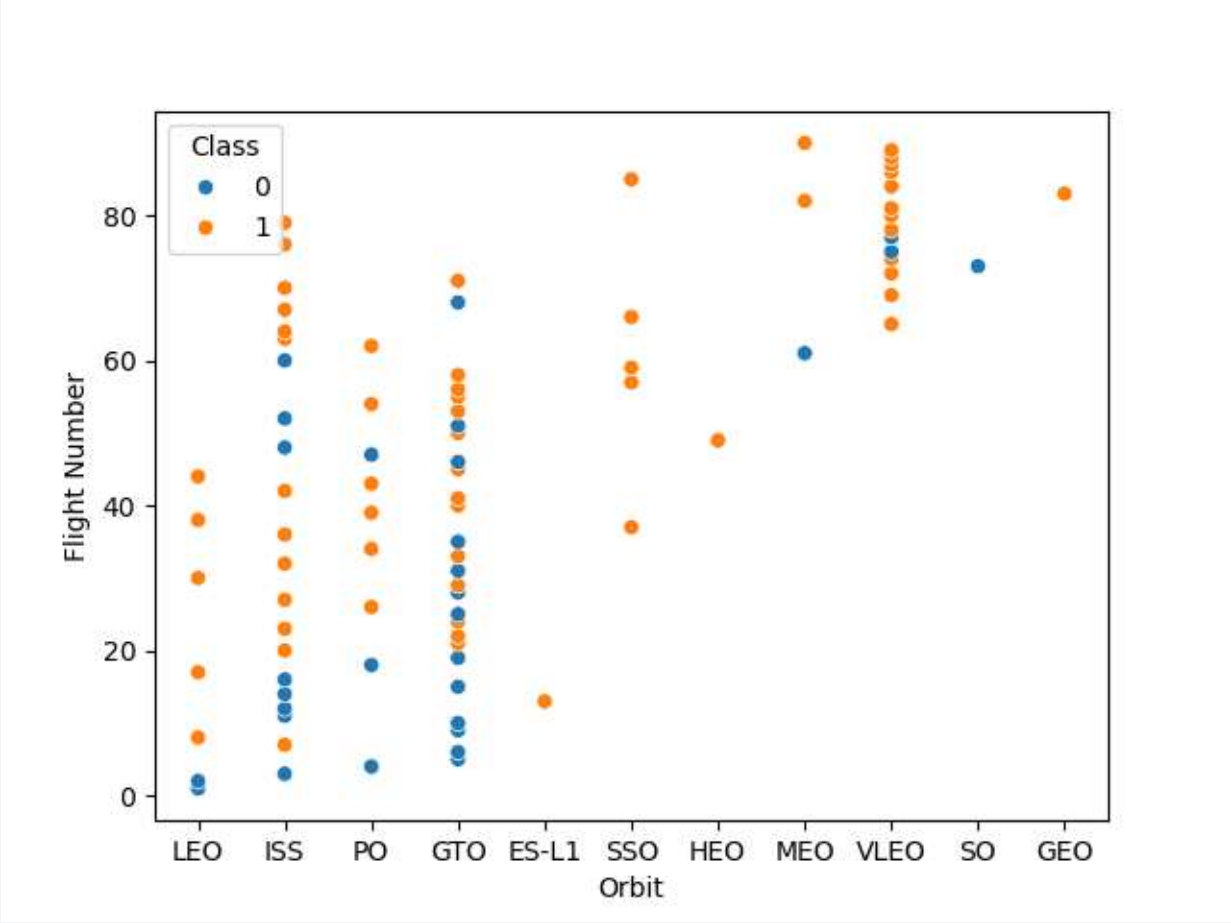
Success Rate vs. Orbit Type

From the bar chart, we can say that ES-L1, GEO, HEO and SSO has the highest success rate compared to other launch to orbits.



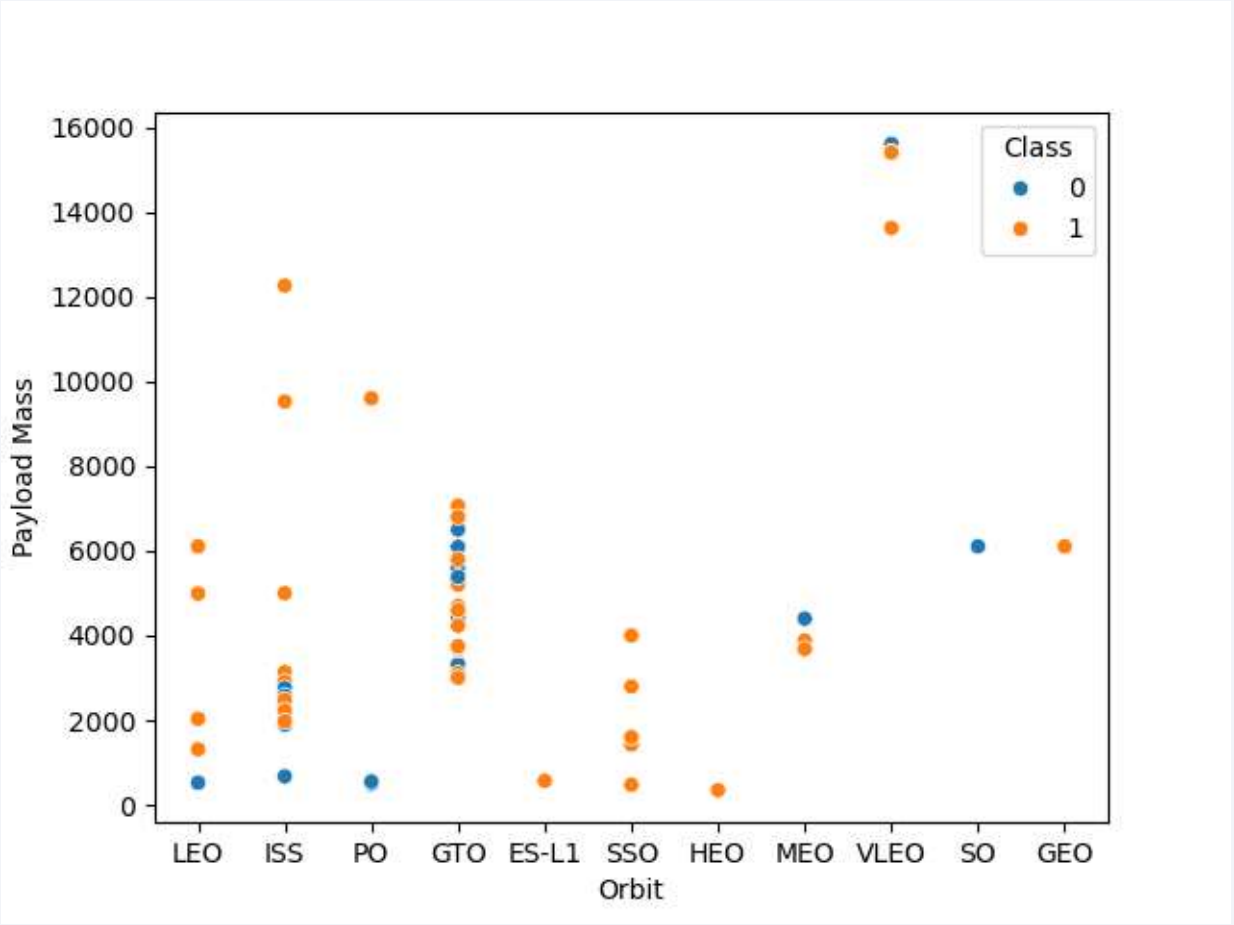
Flight Number vs. Orbit Type

We can see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.



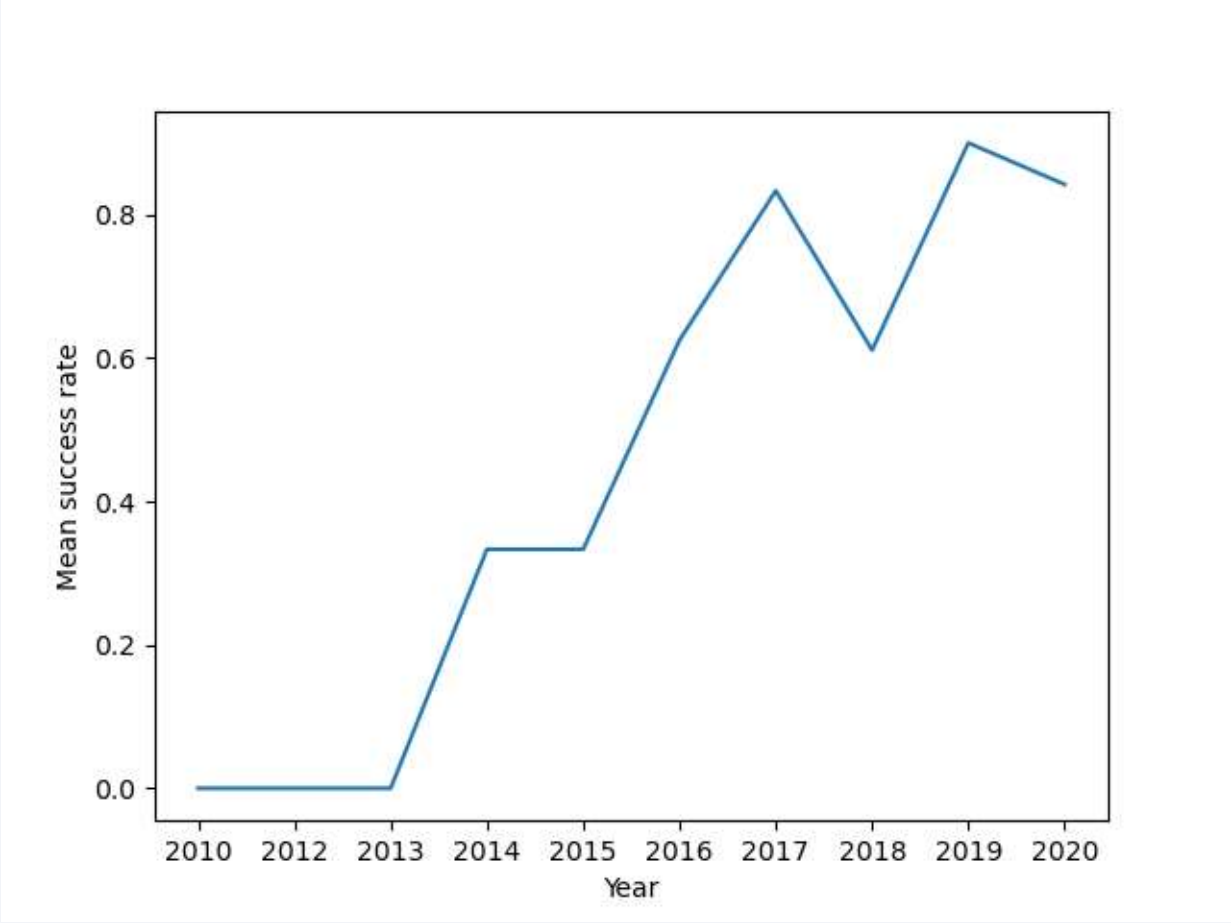
Payload vs. Orbit Type

- With heavy payloads the successful landing rate are more for Polar, LEO and ISS.
- However for GTO we cannot distinguish correctly since successful landing and unsuccessful landing are both present here.
- Furthermore heavier payload mass above 12000 Kg was only launched to VLEO.



Launch Success Yearly Trend

In the line chart we can observe that the success rate since 2013 kept increasing till 2020.



All Launch Site Names

Task 1

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

```
[48]: %sql SELECT DISTINCT Launch_Site from SPACEXTABLE ;
```

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

[48]:

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

- There are four launch site namely CCAFS LC-40, VAFB SLC-4E, KSC LC-39A and CCAFS SLC-40.
- Here we are selecting the distinct item of the column launch site from SPACEXTABLE.

Launch Site Names Begin with 'CCA'

Task 2

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

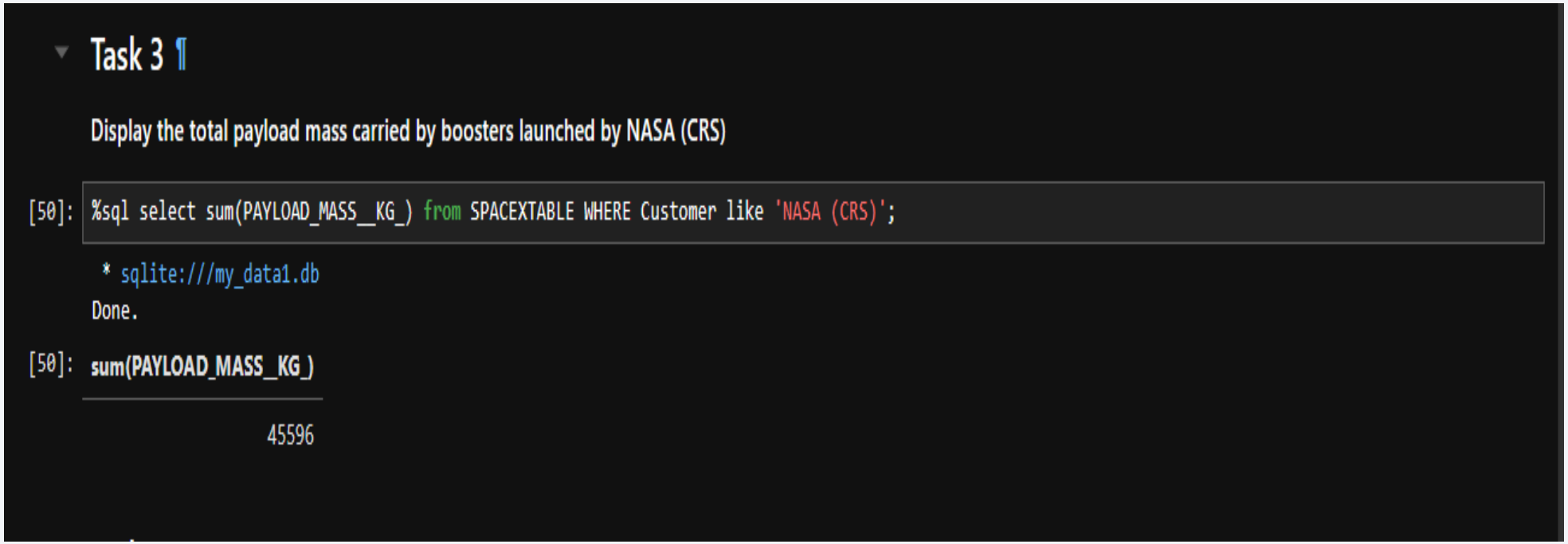
```
[49]: %sql select * from SPACEXTABLE WHERE Launch_Site like 'CCA%' limit 5;
 * sqlite:///my_data1.db
Done.
```

[49]:

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

- Here we are selecting from SPACEXTABLE using WHERE statement launchsite name that starts with ‘CAA’.
- Results shows top 5 observations because of LIMIT statement.

Total Payload Mass



Task 3 ¶

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

```
[50]: %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE WHERE Customer like 'NASA (CRS)';
```

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

[50]:

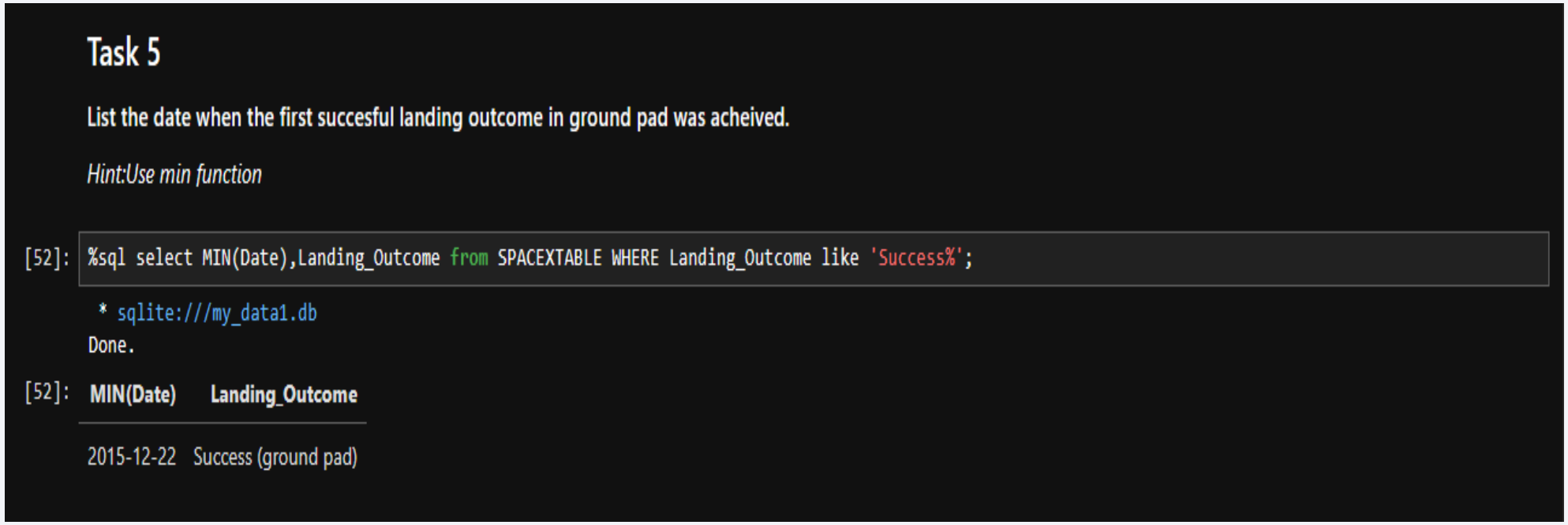
sum(PAYLOAD_MASS_KG_)
45596

- The total payload carried by boosters from NASA is 45596 KG.
- Using the conditional WHERE statement we select the payload mass of the customer named “NASA (CRS)” and summing the payload mass.

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 is 2534.67 KG.
- Using the conditional WHERE statement we select the payload mass of the booster_version named “F9 v1.1 ” and averaging the payload mass.

First Successful Ground Landing Date



- The dates of the first successful landing outcome on ground pad was 22nd December 2015.
- Using the conditional WHERE statement we select the successful launch outcomes and finding the oldest success date using Min function on date.

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

```
[53]: %sql select Booster_version,PAYLOAD_MASS__KG_ from SPACEXTABLE WHERE Landing_Outcome like 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_
```

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

[53]:

Booster_Version	PAYLOAD_MASS__KG_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

- The names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 are F9 FT B1022, F9 FT B1026, F9 FT B1021.2 and F9 FT B1031.2 .
- Condition of payload mentioned using WHERE statement to select booster version and payload mass.

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

```
[54]: %sql select Mission_Outcome,COUNT(*) as Total_Number from SPACEXTABLE group by Mission_Outcome;
```

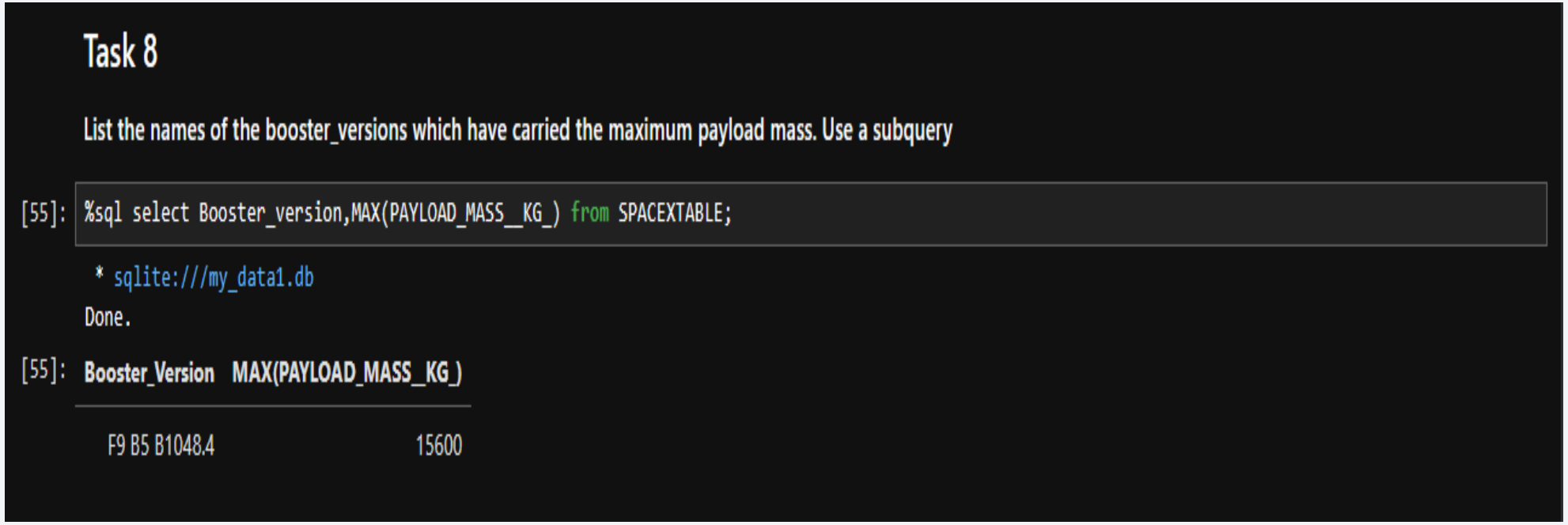
```
 * sqlite:///my_data1.db
Done.
```

[54]:

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

- The total number of successful mission outcomes are 100 while only one mission outcome failed.
- Grouping the SPACEXTABLE by mission outcomes and extracting each mission outcome with the total number using the count function.

Boosters Carried Maximum Payload



- The name of the booster which has carried the maximum payload mass is F9 B5 B1048.4 of about 15600 KG.
- Using the max function on payload mass we select the booster version and payload mass from the SPACEXTABLE.

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, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

[56]:
```
%sql SELECT substr(Date, 6, 2) as Month,Landing_Outcome,Booster_version,Launch_site FROM SPACEXTABLE WHERE substr(Date, 0, 5) = '2015' AND Landing_Outcome
```

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

[56]:

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

- In year 2015, there were two failed landing outcomes in drone ship, their booster versions were F9 v1.1 B1012 and F9 v1.1 B1015, and launch site names for both was CCAFS LC-40.
- Applying condition using WHERE statement on SPACEXTABLE extracting month, landing outcome, booster version and launch site.

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.

```
[57]: %sql SELECT Landing_Outcome,COUNT(*) AS outcome_count FROM SPACEXTABLE WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY
```

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

[57]:

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

- Landing outcome as “no attempt” has highest count of 10 while “precluded (drone ship)” has lowest of 1.
- Here we group by landing outcome and order by outcome count using WHERE observations are from 2010-06-04 and 2017-03-20 and selecting landing outcome and their count.

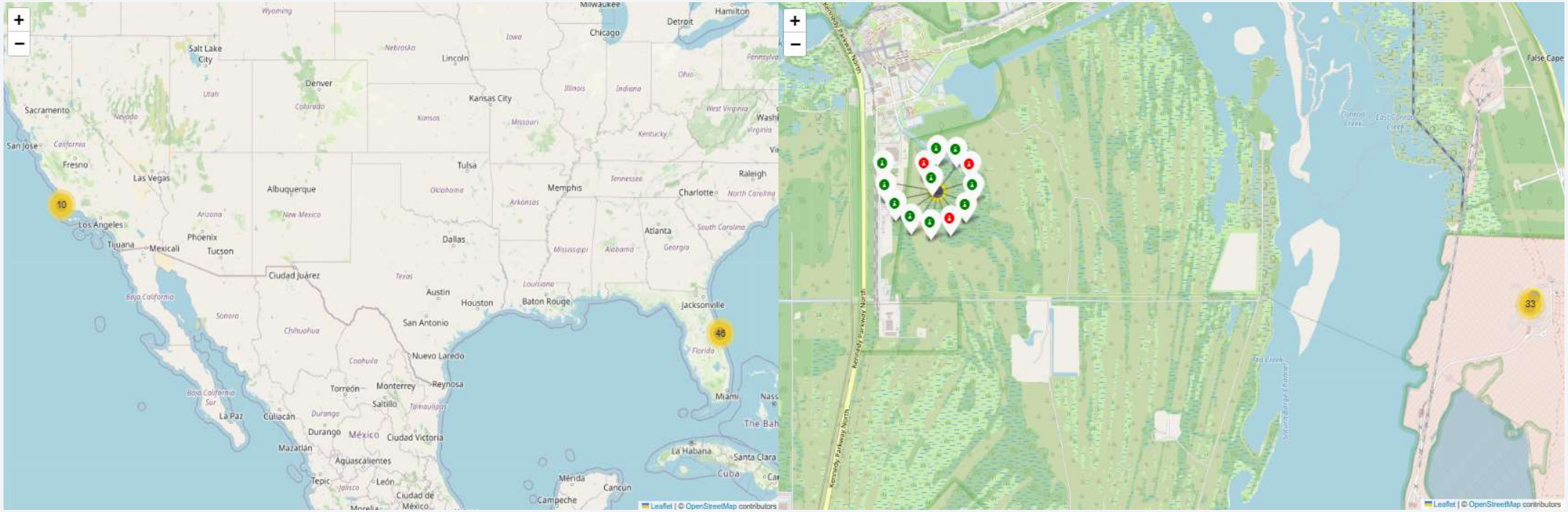
Section 3

Launch Sites Proximities Analysis

All SpaceX Launch Sites Located

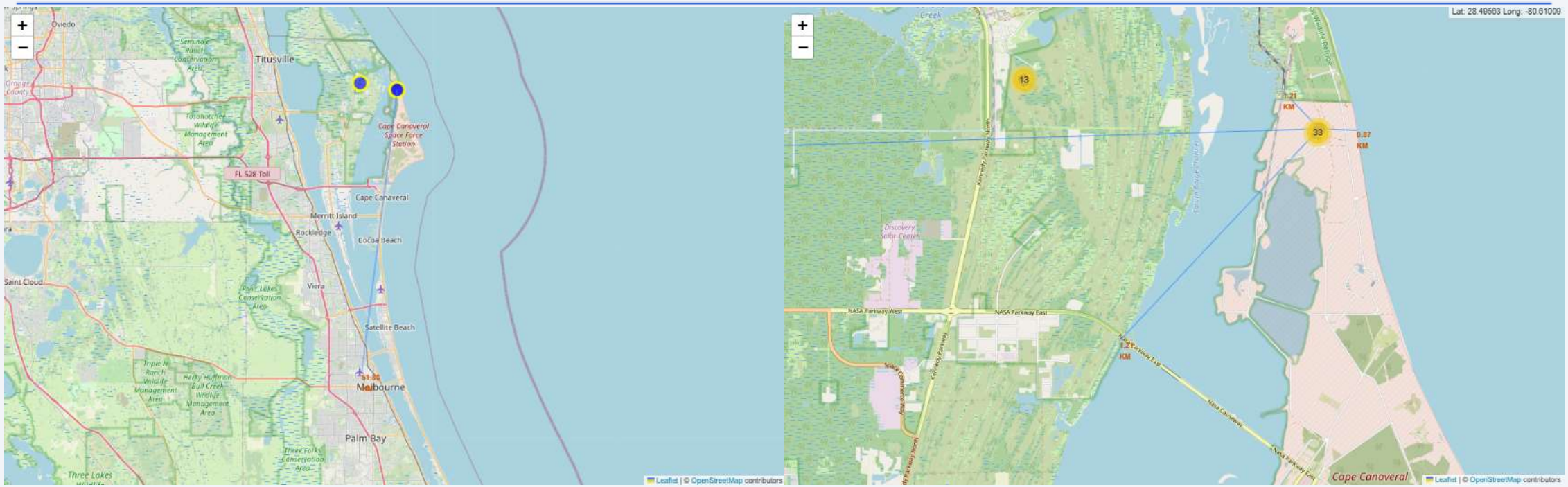
- All the launch site are closer to equator as well as closer to coastline.
- Rocket launches are usually eastward to take advantage of the velocity boost to rocket velocity from the earths rotation so launch site are usually chosen on a coast with water to the east and located closer to the equator than farther north to benefit from the rotational velocity and achieve favorable orbit paths.

Mapping Launch Outcomes to each Launch Site

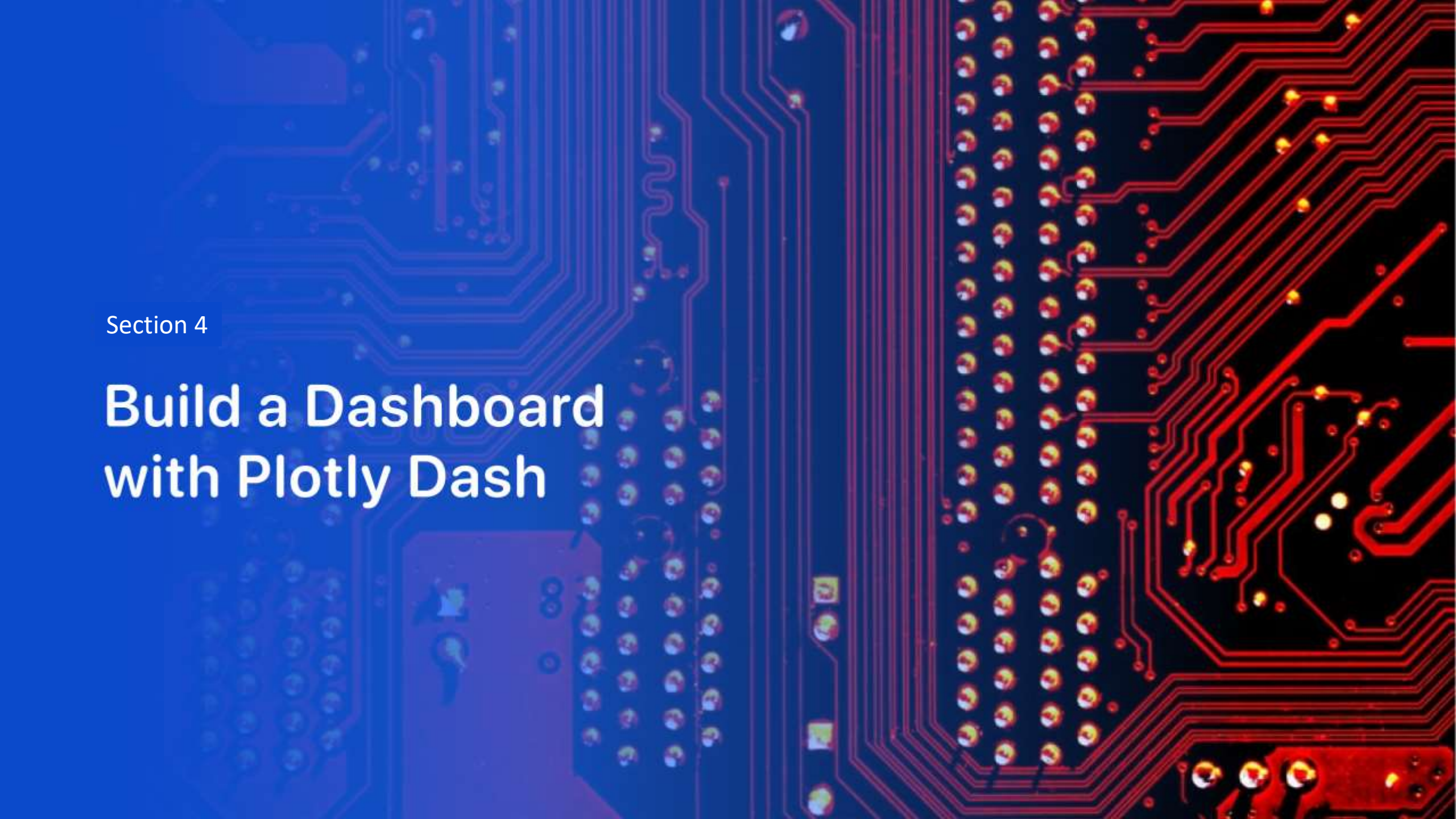


- Mapping the launch outcome to each launch site using the marker cluster function.
- We can see that KSC LC-39A launch site has the highest success rate among the launch site with 10 successful launch outcomes.

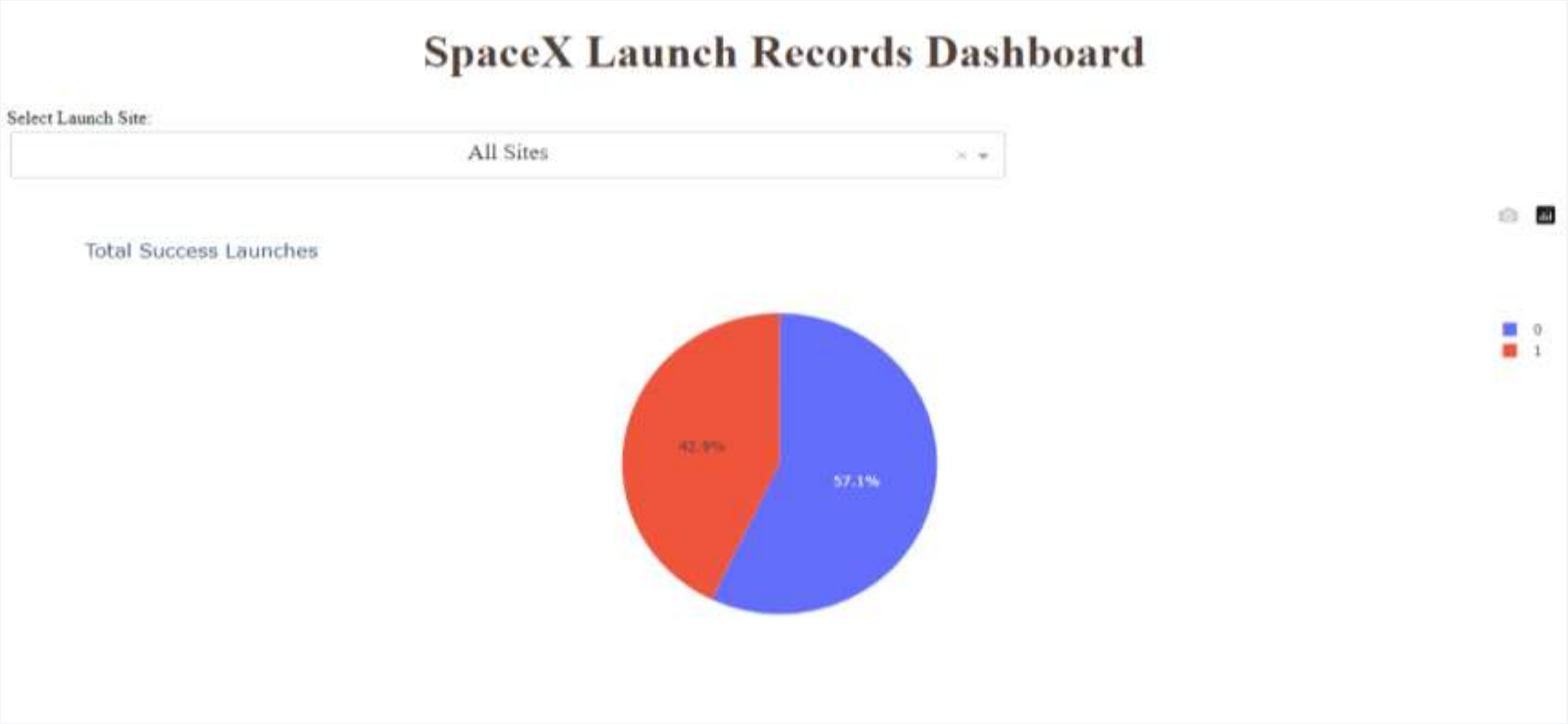
Mapping Distance of Launch Site to Proximities



- All the launch station expect KSC LC-39A are less then a 1 KM away from coastline.
- All launch site have close proximity to railway and highway.
- All the Launch site are at least 50 KM away from the main city.

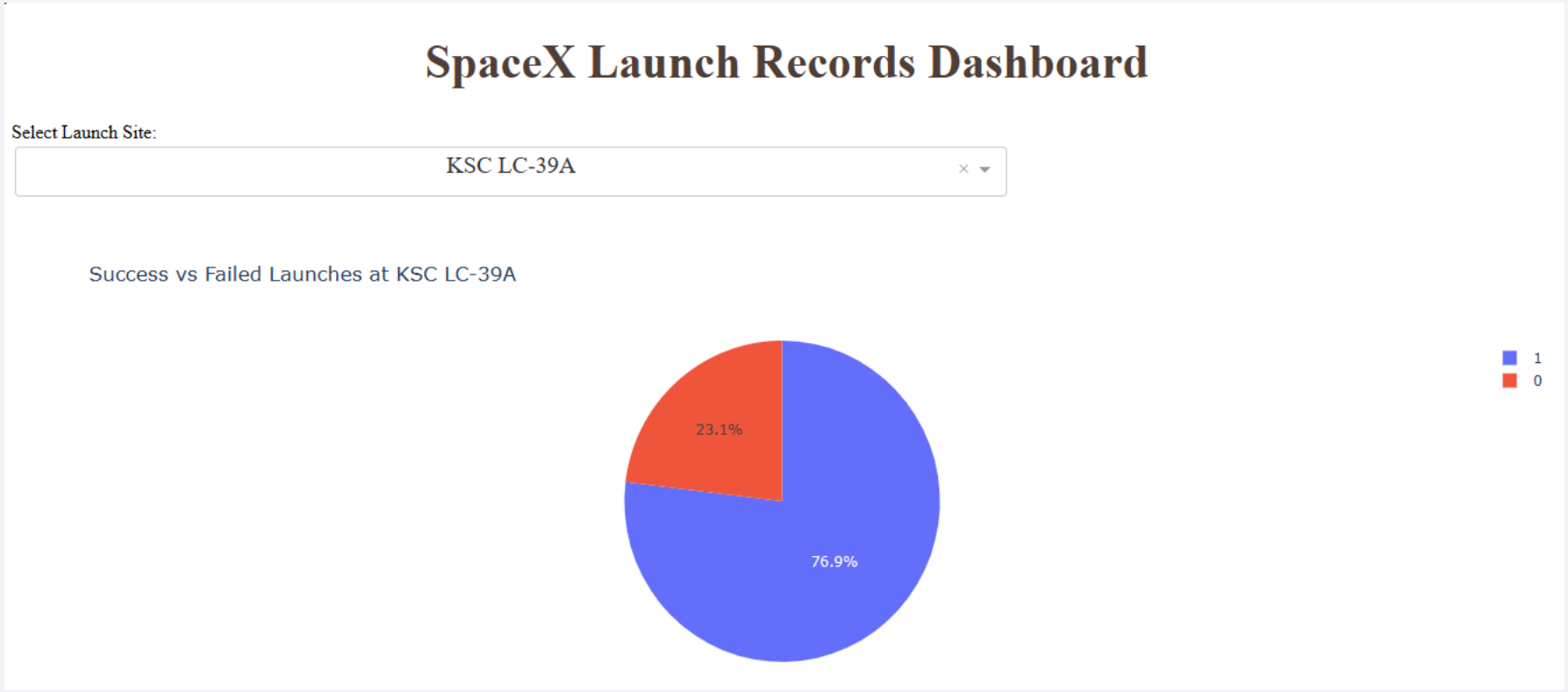

Section 4
Build a Dashboard with Plotly Dash

SpaceX Launch Records Dashboard



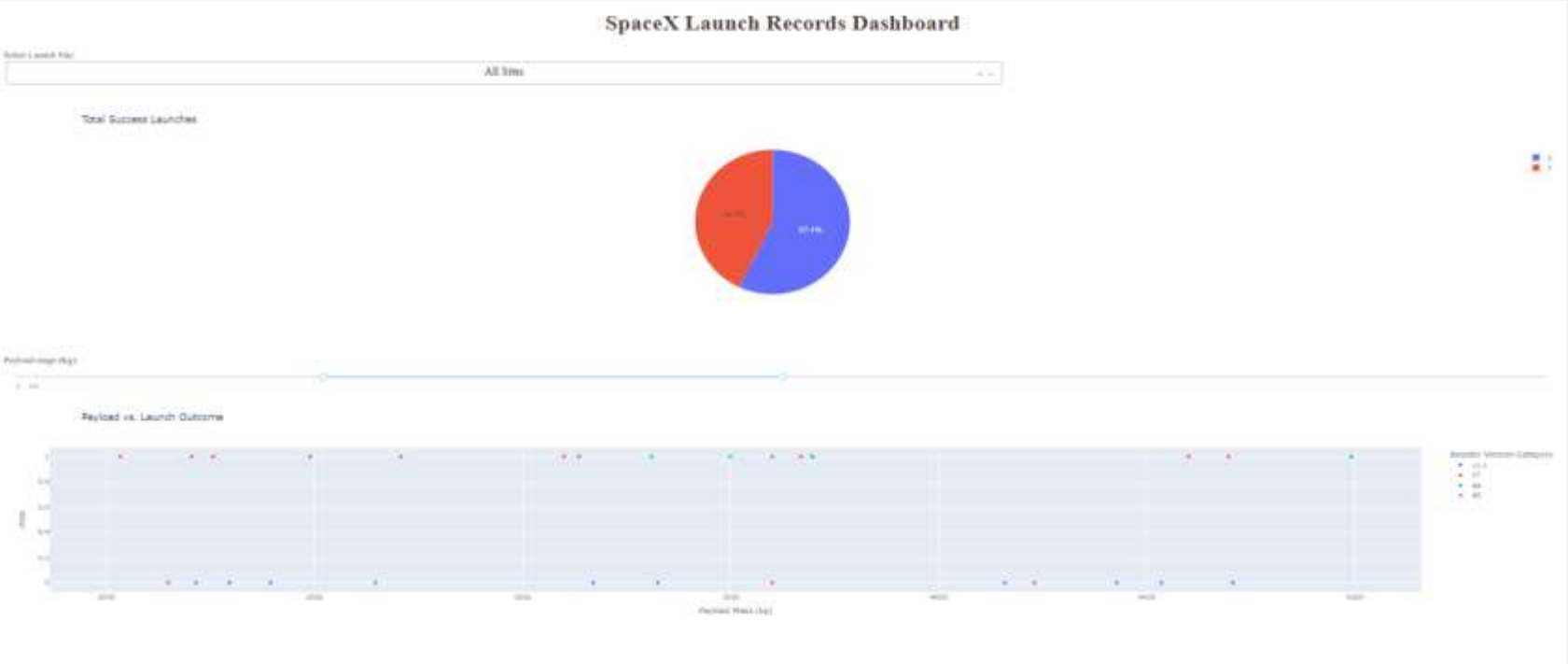
- Total success achieved from all the launch sites is 42.9% which is less than 50%.
- Here we have dropdown component to select launch sites and pie chart plotting the percentage of success.

Launch Site with Highest Launch Success Ratio



- The pie chart results shows that launch site with highest launch success ratio is KSC LC-39A with 76.9%.

Interactive Payload vs. Launch Outcome Scatter Plot



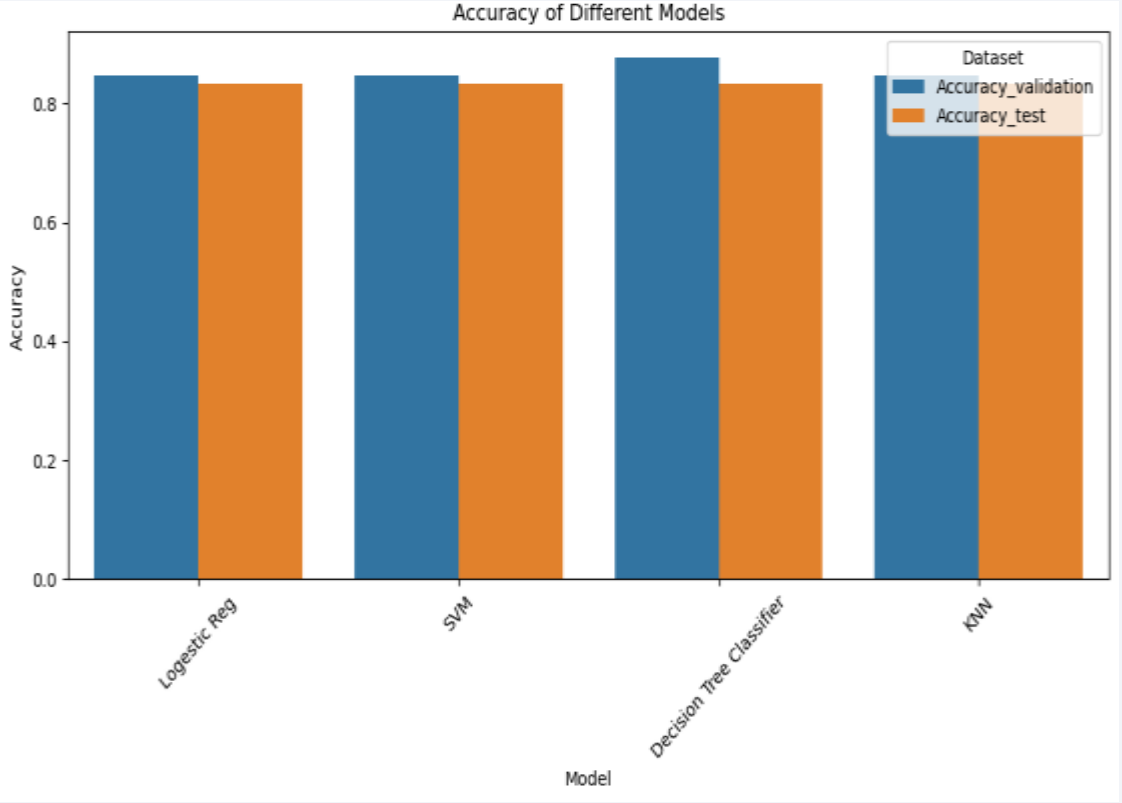
- Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider.
- Launch Outcome with payload between 2000 kg to 6000 kg has the highest success rate.
- Booster version F9 FT had the highest success rate among all the boosters.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

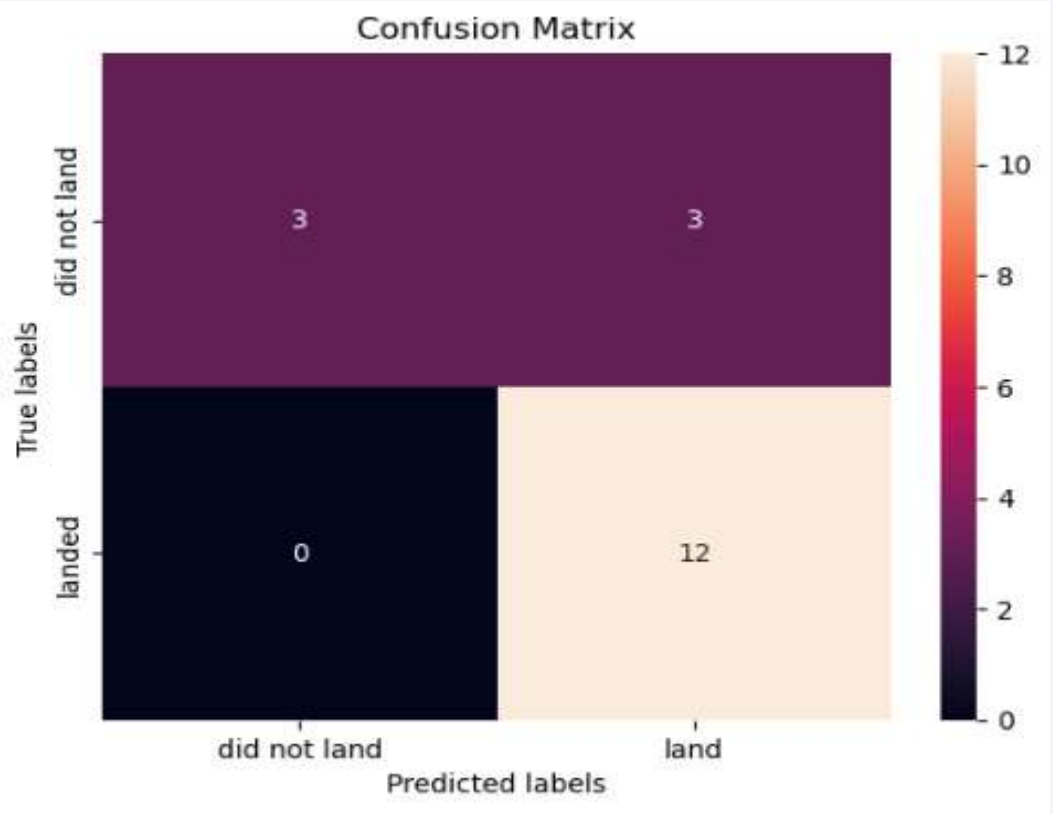
- Visualize the built model accuracy for all built classification models, in a bar chart
- As shown in the bar chart, all classification model performed similarly on the test dataset and showed same accuracy score.
- However there seems overfitting with Decision Tree Classifier model since the accuracy score on test dataset was fluctuating showing huge variation.



Confusion Matrix

- All the models performed similarly with respect to test dataset hence the confusion matrix represents all the classifier models.
- Out of 12 actual positive landing outcomes models classified all of them correctly however out of 6 negative landing outcomes models failed to classify 3 of them.

	Precision	Recall	F1-score
Did not land	1	0.5	0.67
Landed	0.8	1	0.89
Average accuracy (overall performance)			0.78



Conclusions

- There is a clear relationship between the flight number and successful launches. Initially, most launches occurred at CCAFS SLC-40 with very few successful launches. SpaceY's lesson learned is that there will be initial failures.
- The VAFB-SLC launch site was not used for payloads heavier than 10,000kg. Dedicating different launch sites for launches with different payload masses and vehicle types is important.
- ES-L1, GEO, HEO, and SSO orbits have a 100% success rate. Therefore focusing initially on these orbits can lead to early success.
- As the number of flights increased, the success rate of launches for LEO also increased. With experience, success at LEO orbit becomes more probable. LEO is the easiest orbit to achieve a successful launch at initial stages compared to other orbits.
- With an increase in payload mass, the chances of a successful launch also increase for LEO, ISS, and PO orbits. Most heavier payload masses were launched to VLEO.
- The success rate increased after 2013. It took SpaceX 3 years to achieve their first success in launching Falcon 9. Continuous iteration is essential until success, and it will take some years.
- They have four launch sites: CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, and CCAFS SLC-40. Having multiple launch sites is beneficial for adapting to different types of launch vehicles.

Conclusions

- The average payload mass carried by the booster version F9 v1.1 is 2,534.67kg.
- F9 FT B1022, F9 FT B1026, F9 FT B1021.2, and F9 FT B1031.2 successfully landed on a drone ship carrying payloads between 4,000 and 6,000kg.
- F9 B5 B1048.4 carried the maximum payload mass, which was about 15,600kg.
- All launch sites are located near the equator and coastline to take advantage of Earth's rotation and to achieve a favorable orbit path.
- All launch sites are 1km away from the coastline, with the railway line and highway close to 5km, but the nearest main city is at least 50km away.
- The total success achieved from all the launch sites is 42.9%, which is less than 50%.
- The launch site with the highest launch success ratio is KSC LC-39A with 76.9%.
- All classification models show the same accuracy of 83% in classifying labels in the test dataset. This means that 83% of the labels were correctly predicted by the models.

Appendix

- Information about the SpaceX dataset on the right image.
- Launch count of the each launchsite is below.

```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

```
LaunchSite
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: count, dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90 entries, 0 to 89
Data columns (total 18 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   FlightNumber    90 non-null     int64
 1   Date            90 non-null     object
 2   BoosterVersion  90 non-null     object
 3   PayloadMass     90 non-null     float64
 4   Orbit           90 non-null     object
 5   LaunchSite      90 non-null     object
 6   Outcome         90 non-null     object
 7   Flights         90 non-null     int64
 8   GridFins        90 non-null     bool
 9   Reused          90 non-null     bool
 10  Legs            90 non-null     bool
 11  LandingPad      64 non-null     object
 12  Block           90 non-null     float64
 13  ReusedCount     90 non-null     int64
 14  Serial          90 non-null     object
 15  Longitude       90 non-null     float64
 16  Latitude        90 non-null     float64
 17  Class           90 non-null     int64
dtypes: bool(3), float64(4), int64(4), object(7)
memory usage: 10.9+ KB
```

Appendix

- Classification model parameter and accuracy.

```
#Logistic regressoin model parameters and accuracy.
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

```
#SVM model parameters and accuracy.
print("tuned hpyerparameters :(best parameters) ",svm_cv.best_params_)
print("accuracy :",svm_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
accuracy : 0.8482142857142856
```

```
#Decision Tree Classifier model parameters and accuracy.
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 4, 'max_features': 'log2', 'min_samples_leaf': 2, 'min_samples_split': 1
0, 'splitter': 'best'}
accuracy : 0.8767857142857143
```

```
#KNN model parameters and accuracy.
print("tuned hpyerparameters :(best parameters) ",knn_cv.best_params_)
print("accuracy :",knn_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
accuracy : 0.8482142857142858
```

Thank you!