IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Cüneyt Erem
22.5.2023

# Outline

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

# Executive Summary

- We have collected data from both spacex api and wikipedia webpage. We used folium and dash graphical visualizations. We examined data with sql, then created four different machine learning models and compared accuracy scores.
- Logistic regression, SVM, decision tree and KNN models have similar acuracy score above 80%, however most accurate score is 87% from decision tree.

# Introduction

- SpaceY is a company try to examine SpaceX company rocket landings and examine successful missions with exploring data science methodologies.
- We want to apply four different machine learning models and compare accuracy scores, then plot graphs from data columns to dive into deep

Section 1

# Methodology

# Methodology

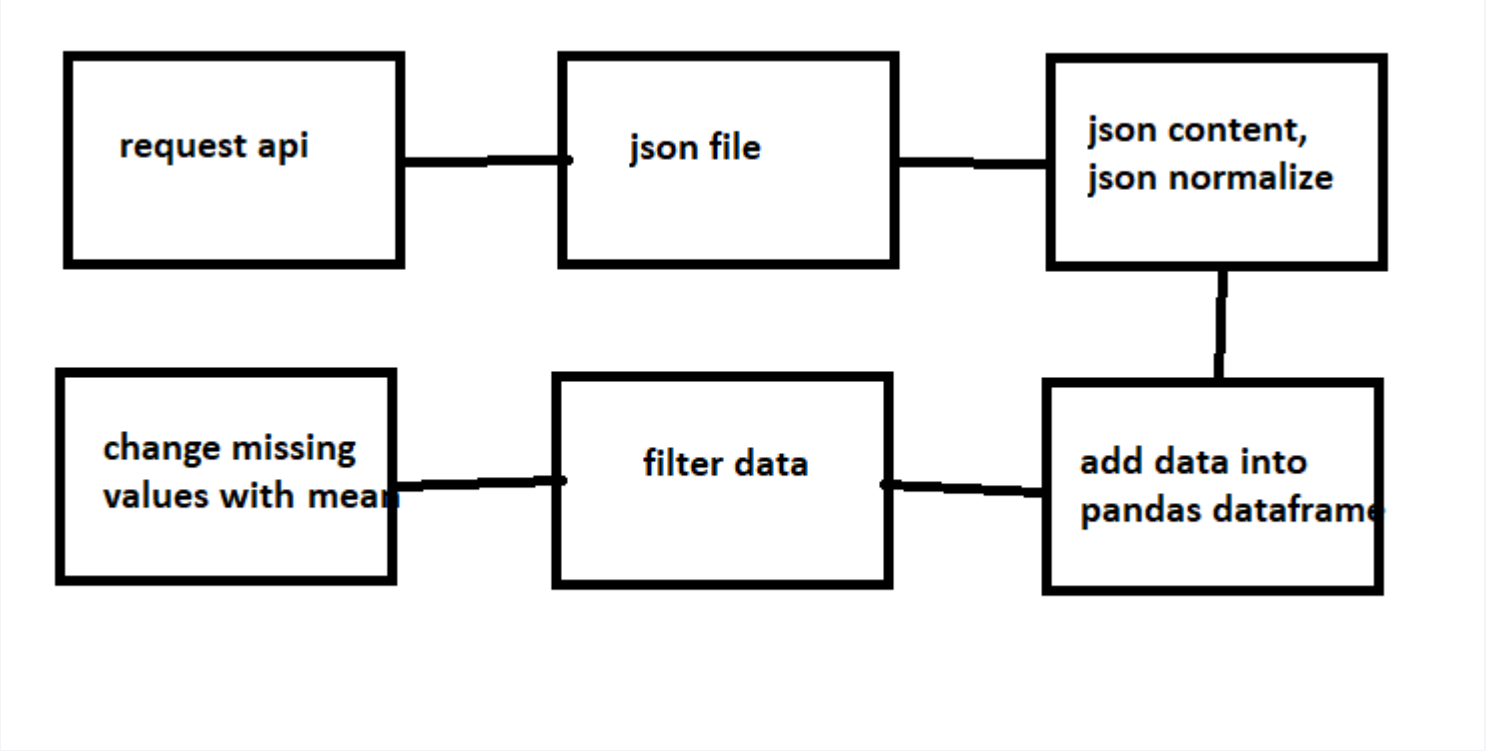
## Executive Summary

- Data collection methodology:
  - Use spacex api and webscraping sikipedia page data
- Perform data wrangling
  - Examine data with rocket launch specifications
- 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 with logistic regression, svm, decision tree and knn algorithms

# Data Collection

- 2 types of data collection behavior is applied such as retrieving from api and webscraping from Wikipedia page.
- By retrieving spacex api data, we can get response data in JSON format. We collected data features like rocket, payloads, launchpad, flightnumber etc.
- By using beautifulsoup package, we can websraping spacex wikipedia page and use find_all method to find specific table tabs. Then we can collect data features like rocket, payloads, launchpad, flightnumber etc. into dataframe.

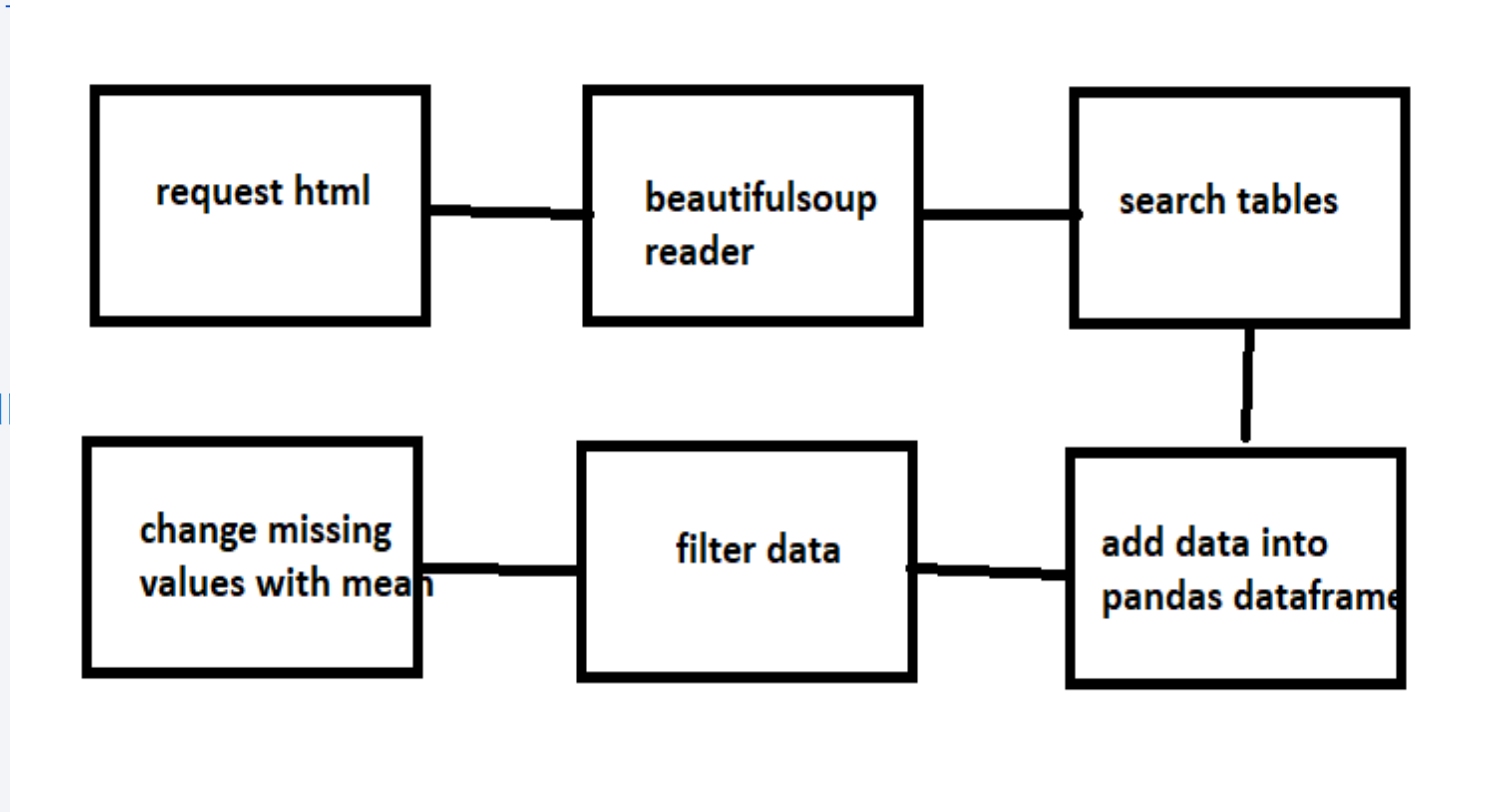
# Data Collection – SpaceX API




- https://github.com/cuneyterem8/certificate_projects/blob/main/ibm_data_science/11_applied_ds_capstone_project/1_jupyter-labs-spacex-data-collection-api.ipynb

# Data Collection - Scraping

Place you




- https://github.com/cuneyterem8/certificate_projects/blob/main/ibm_data_science/11_applied_ds_capstone_project/2_jupyter-labs-webscraping.ipynb

# Data Wrangling

- After reading data with pandas dataframe, we examine data with counting number of null values for each columns from flightnumber to latitude.
- We count number of launchsites, orbits, occurences of outcome, bad_outcomes and mean of class.

| Flight Number | Date | BoosterVersion | PayloadMass | Orbit | LaunchSite | Outcome | Flights | GridFins | Reused | Legs | LandingPad | Block | ReusedCount | Serial | Longitude | Latitude | Class |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2010-06-04 | Falcon 9 | 6104.959412 | LEO | CCAFS SLC 40 | None None | 1 | False | False | False | NaN | 1.0 | 0 | B0003 | -80.577366 | 28.561857 | 0 |

- https://github.com/cuneyterem8/certificate_projects/blob/main/ibm_data_science/11_applied_ds_capstone_project/3_labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

# EDA with Data Visualization

- To see outcome of the launch, we can catplot flightnumber vs payloadmass
- Ksc-lca-39a and vafb-slc-4e have 77% success rate so when flight number incerases, landing successful rate is also increases
- Catplot for flightnumber vs launchsite is also applicable, if flight number > 40, it is more likely to land successfully
- Catplot for launchsite vs payloadmass, if ccafs-slc-40  have more payload and ksc-lc-39a have less payload, then they are more likely to laúnch succesfully
- Best orbits are es-l1, sso, heo and geo
- Success rate increses over time and have highest point at 2020

- https://github.com/cuneyterem8/certificate_projects/blob/main/ibm_data_science/11_applied_ds_capstone_project/5_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

# EDA with SQL

- We find distinct launch sites, first 5 cca launch sites (ccafs-lc-40, vafb-slc-4e etc.), total payloadmass (45596 kg) by nasa and average payloadmass by f9-v1.1 (2534.66).
- In 2014, first landing launch is become successful
- There are four different booster versions which are successful and mass is betwen 4000-6000; f9-ft-b-1022, 1026, 1021.2 and 1031.2
- Total succesful mission is 99 + 1 (unclear) and failure is 1
- In 4 and 10th months, drone ship failures happened
- Between 2010 and 2017, there are many count numbers for success, no attempt etc
- https://github.com/cuneyterem8/certificate_projects/blob/main/ibm_data_science/11_applied_ds_capstone_project/4_jupyter-labs-eda-sql-coursera_sqllite.ipynb

# Build an Interactive Map with Folium

- After finding launch sites with longitudes and latitudes, we added circles and folium marken to the map on nasa coordinates.
- We added yellow circle onto houston with circle (radius=1000) and popup
- Then we make cluster with red and green marker colors and add onto map
- Then we calculated distances between launch sites and closest coastlines by giving coordinates

- https://github.com/cuneyterem8/certificate_projects/blob/main/ibm_data_science/11_applied_ds_capstone_project/6_lab_jupyter_launch_site_location.jupyterlite.ipynb

# Build a Dashboard with Plotly Dash

- We added drop-down interface with five launchsite names by option menu on top of the screen
- We modified success-pie-chart with total success rate for each class and with selected cites option
- Rangeslider is added with three options from 0 to 10,000 kg
- Scatter chart is created with payloadmass vs class columns for min/max payload capacity

- https://github.com/cuneyterem8/certificate_projects/blob/main/ibm_data_science/11_applied_ds_capstone_project/7_spacex_dash_app.py

# Predictive Analysis (Classification)

- The independent data (column: class) is assigned to Y and dependent variables are assigned to X
- We used standard scaler and transformed each X variable, then splitted train and test data with 0.2 test size and random state 2
- We used four different models with different parameters; these are logistic regression, SVM, decision tree and KNN classifiers. We created confusion matrix for each of them and applied gridsearch to find best parameters for each model.

- https://github.com/cuneyterem8/certificate_projects/blob/main/ibm_data_science/11_applied_ds_capstone_project/8_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

# Results

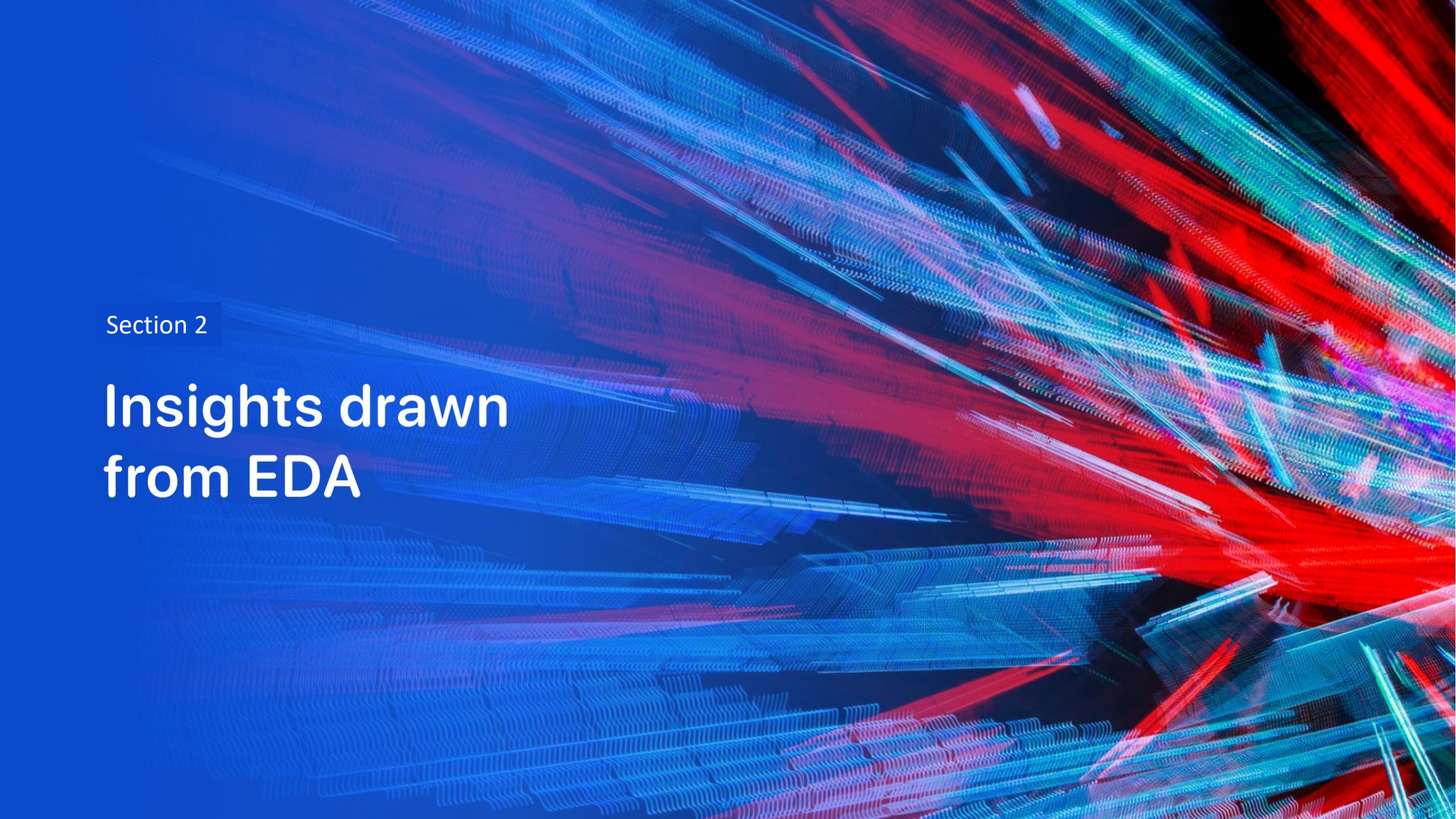
- The best result is decision tree with 87% accuracy score, details can be seen here;

```
performances= pd.DataFrame({"logreg_cv": [logreg_cv.best_params_, logreg_cv.best_score_],
                            "svm_cv": [svm_cv.best_params_, svm_cv.best_score_],
                            "tree_cv": [tree_cv.best_params_, tree_cv.best_score_],
                            "knn_cv": [knn_cv.best_params_, knn_cv.best_score_]}, index= ["parameters", "best accuracy"])
performances
```

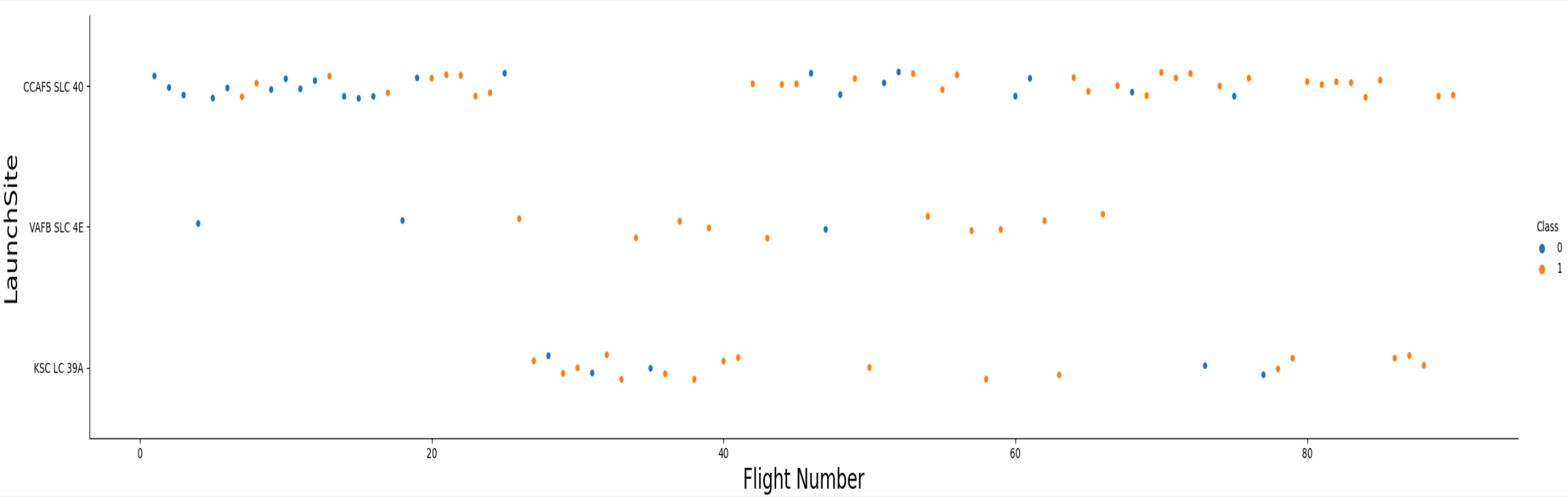
| | logreg_cv | svm_cv | tree_cv | knn_cv |
|---|---|---|---|---|
| **parameters** | {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'} | {'C': 1.0, 'gamma': 0.03162277660168379, 'kern... | {'criterion': 'entropy', 'max_depth': 10, 'max... | {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1} |
| **best accuracy** | 0.846429 | 0.848214 | 0.876786 | 0.848214 |

```
[tree_cv.best_params_, tree_cv.best_score_]
```

```
[{'criterion': 'entropy',
  'max_depth': 10,
  'max_features': 'sqrt',
  'min_samples_leaf': 4,
  'min_samples_split': 2,
  'splitter': 'random'},
 0.8767857142857143]
```

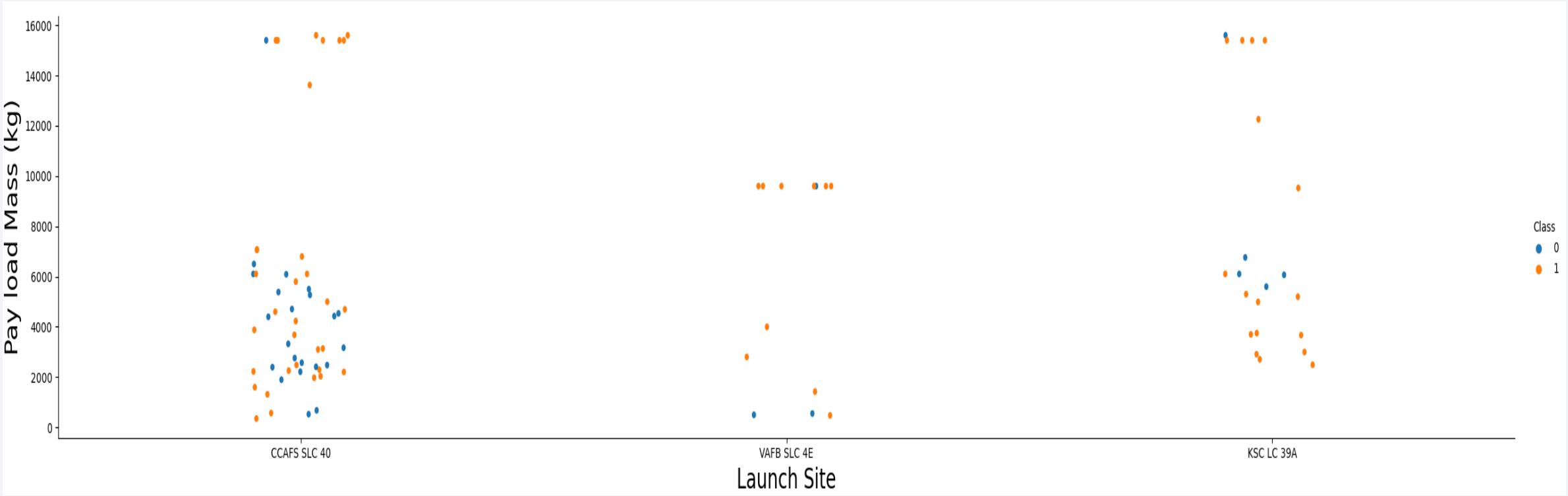

Section 2
Insights drawn from EDA

# Flight Number vs. Launch Site



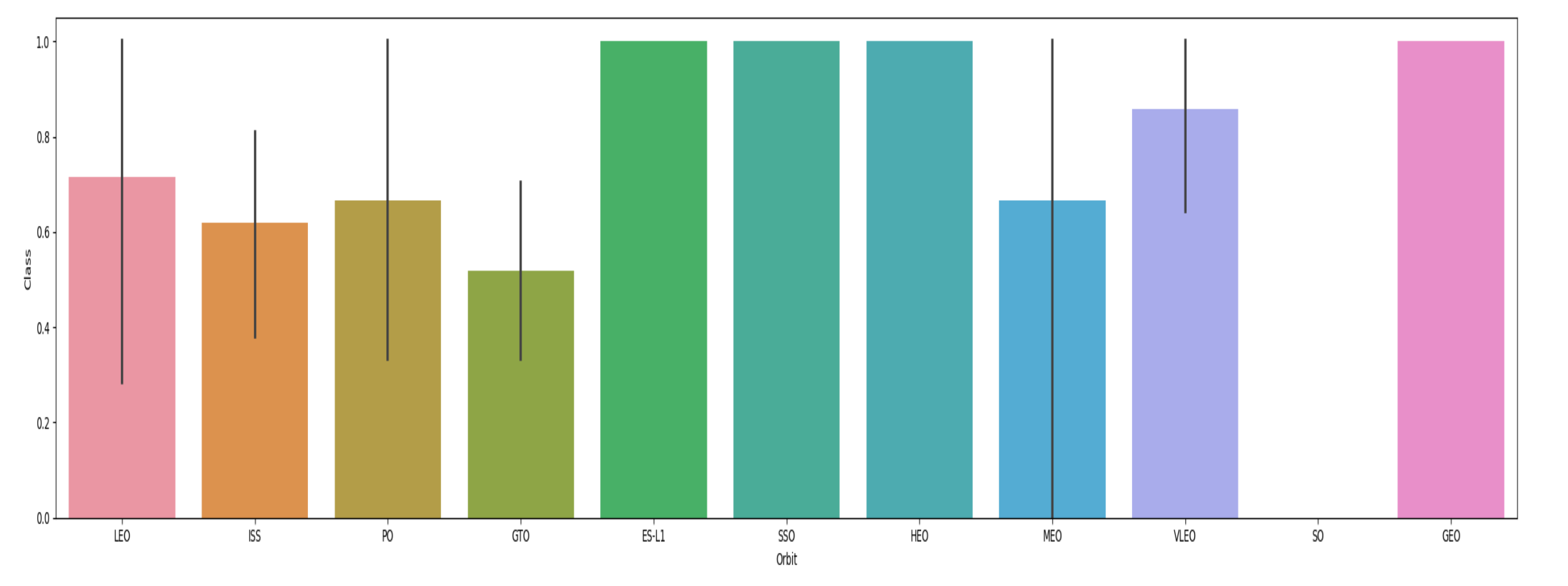

- If flight number increases, success rate is also increases for 3 launchsites

# Payload vs. Launch Site



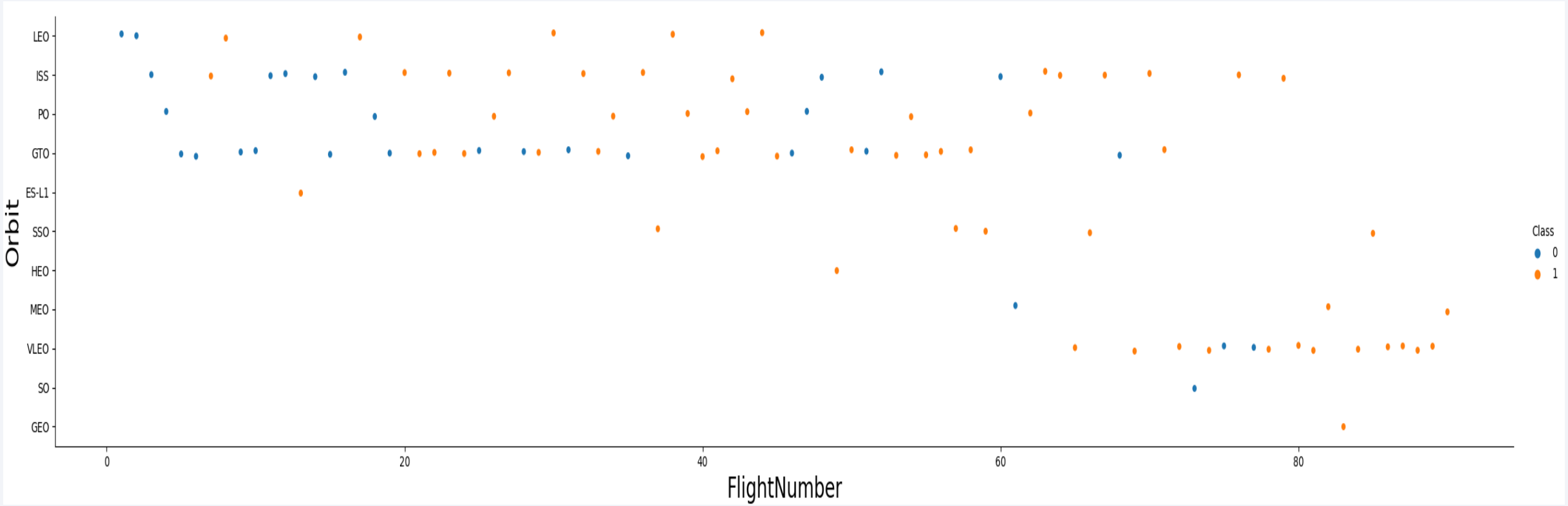

- If launch site is ksc-lc-39a and payload mass is high, then success increases

# Success Rate vs. Orbit Type



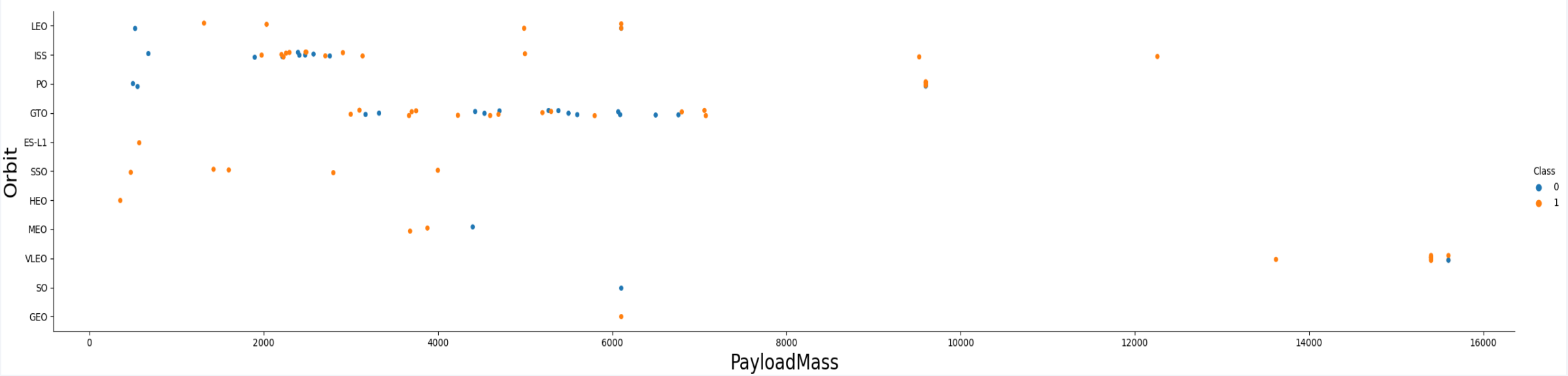

- Most successful orbits are as-l1, ssc, heo and geo with their types

# Flight Number vs. Orbit Type



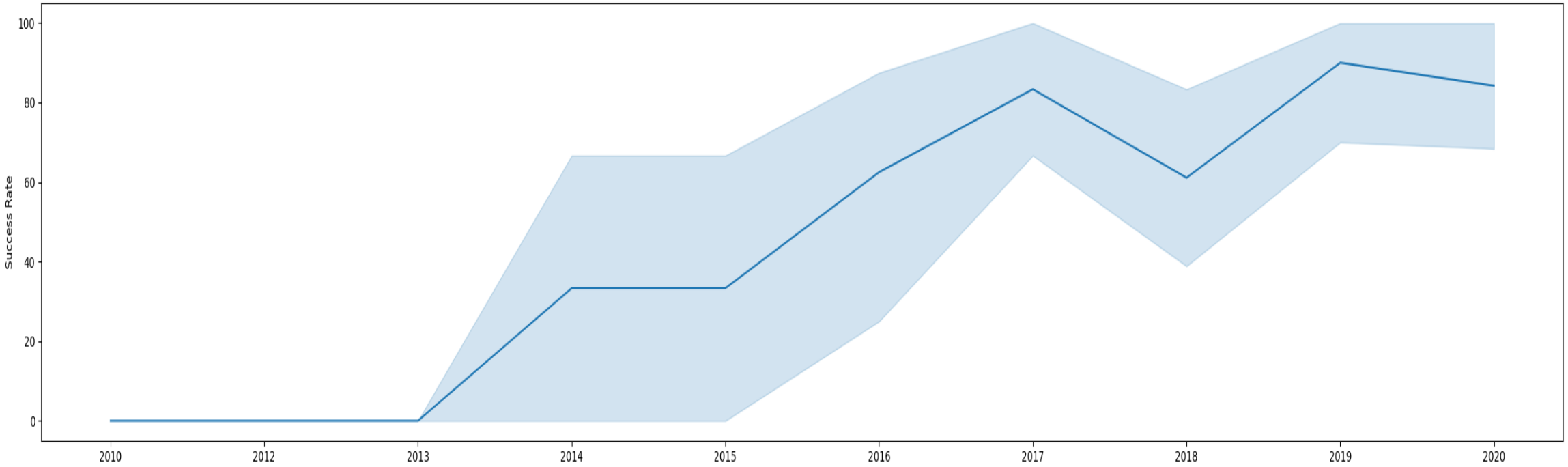

- Overall, if flight number increases, success rate increases for different orbits

# Payload vs. Orbit Type




- Geo: it does not changefor payload
- Sso, heo, leo, iss: mostly successful for each payload
- Meo, so: if payload increases, success rate decreases

# Launch Success Yearly Trend




- It can be clearly seen that success rate increases over time, only exception is year 2018 and small decrease in 2020

# All Launch Site Names

```
%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 |

- There are four distinct launch names as written above

# Launch Site Names Begin with 'CCA'

- CCA like launchsites have different orbits and customers as can be seen right side

```
%sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
```

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

| Date | Time (UTC) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Orbit | Customer | Mission_Outcome | Landing_Outc |
|---|---|---|---|---|---|---|---|---|---|
| 06/04/2010 | 18:45:00 | F9 v1.0 B0003 | CCAFS LC-40 | Dragon Spacecraft Qualification Unit | 0.0 | LEO | SpaceX | Success | Failure (paracl |
| 12/08/2010 | 15:43:00 | F9 v1.0 B0004 | CCAFS LC-40 | Dragon demo flight C1, two CubeSats, barrel of Brouere cheese | 0.0 | LEO (ISS) | NASA (COTS) NRO | Success | Failure (paracl |
| 22/05/2012 | 7:44:00 | F9 v1.0 B0005 | CCAFS LC-40 | Dragon demo flight C2 | 525.0 | LEO (ISS) | NASA (COTS) | Success | No atte |
| 10/08/2012 | 0:35:00 | F9 v1.0 B0006 | CCAFS LC-40 | SpaceX CRS-1 | 500.0 | LEO (ISS) | NASA (CRS) | Success | No atte |
| 03/01/2013 | 15:10:00 | F9 v1.0 B0007 | CCAFS LC-40 | SpaceX CRS-2 | 677.0 | LEO (ISS) | NASA (CRS) | Success | No atte |

# Total Payload Mass

```
%sql select sum(PAYLOAD_MASS__KG_) as sum_payload_mass_kg from SPACEXTBL where Customer like 'NASA (CRS)'
```

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

| sum_payload_mass_kg |
|---|
| 45596.0 |

- Nasa used total payload of 45596 kg

# Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) as avg_payload_mass_kg from SPACEXTBL where Booster_Version like 'F9 v1.1%'
 * sqlite:///my_data1.db
Done.
 avg_payload_mass_kg
 2534.6666666666665
```

- Average payload is 2534.66 for F9-v1.1

# First Successful Ground Landing Date

```
%sql select min(date) as first_landing_date from SPACEXTBL where Mission_Outcome like 'Success'
```

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

| first_landing_date |
| --- |
| 01/06/2014 |

- First successful landing is happened in 2014

# Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select Booster_Version from SPACEXTBL where (Mission_Outcome like 'Success') and (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 60
```

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

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

- There are four different booster version which were successful

# Total Number of Successful and Failure Mission Outcomes

**List the total number of successful and failure mission outcomes**

```
%sql SELECT Mission_Outcome, count(Mission_Outcome) as Count_total_success from SPACEXTBL group by Mission_Outcome
```

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

| Mission_Outcome | Count_total_success |
|---|---|
| None | 0 |
| Failure (in flight) | 1 |
| Success | 98 |
| Success | 1 |
| Success (payload status unclear) | 1 |

- There are 100 success and 1 failure mission outcome

# Boosters Carried Maximum Payload

```
%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 |

- In total, there are 12 booster versions with maximum payloads

# 2015 Launch Records

```
%sql select substr(DATE, 4, 2) as month_name, Landing_Outcome, Booster_Version, Launch_Site  from SPACEXTBL where substr(Dat
```

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

| month_name | Landing_Outcome | Booster_Version | Launch_Site |
|---|---|---|---|
| 10 | Failure (drone ship) | F9 v1.1 B1012 | CCAFS LC-40 |
| 04 | Failure (drone ship) | F9 v1.1 B1015 | CCAFS LC-40 |

- Drone failures are at 4 –th and 10-th months for f9-v1.1

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

```
%sql select Landing_Outcome, count(Landing_Outcome) as count_landing_outcome from SPACEXTBL where Date <= '2017-03-20' GROUP
```

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

| Landing_Outcome | count_landing_outcome |
|---|---|
| Success | 25 |
| No attempt | 14 |
| Success (ground pad) | 8 |
| Success (drone ship) | 8 |
| Failure (drone ship) | 5 |
| Failure | 3 |
| Controlled (ocean) | 3 |
| Failure (parachute) | 2 |
| No attempt | 1 |

- Success is 25, no attempt is 14 and there are other success and failure as above

Section 3

# Launch Sites Proximities Analysis

# <Folium Map Screenshot 1>

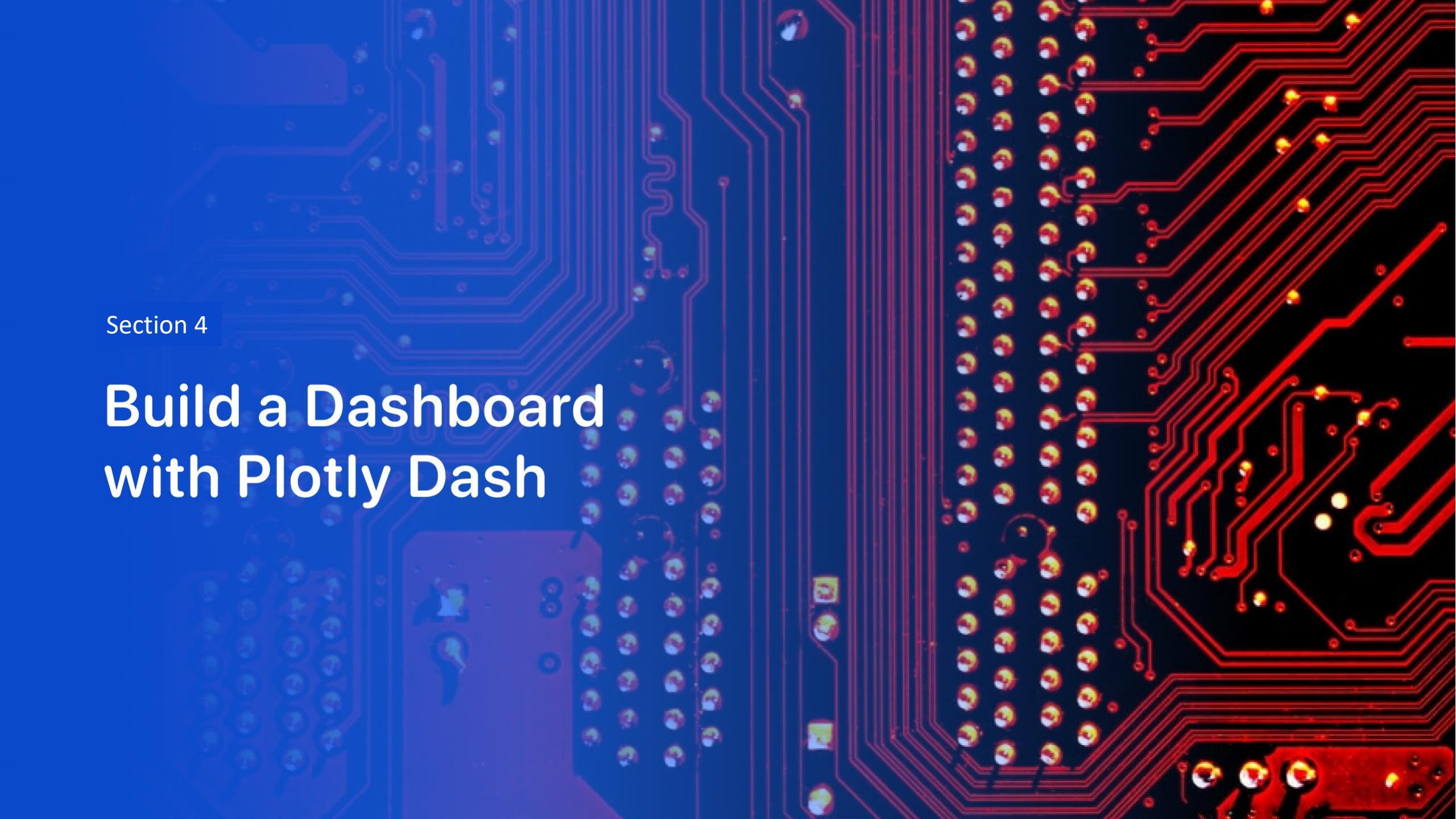

- By adding circles and markers, we can identify 2 launch sites on the map

# <Folium Map Screenshot 2>

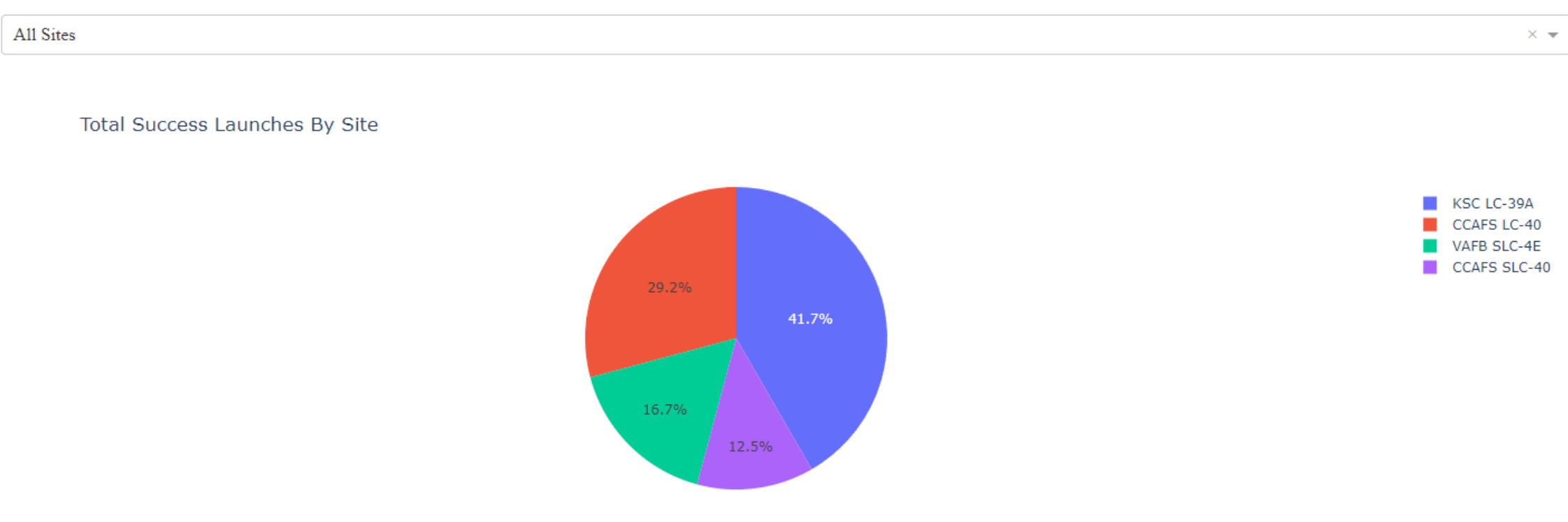

- We can identify classes red or green if successful or not

# <Folium Map Screenshot 3>


- We can calculate distance from center to nearest coastline

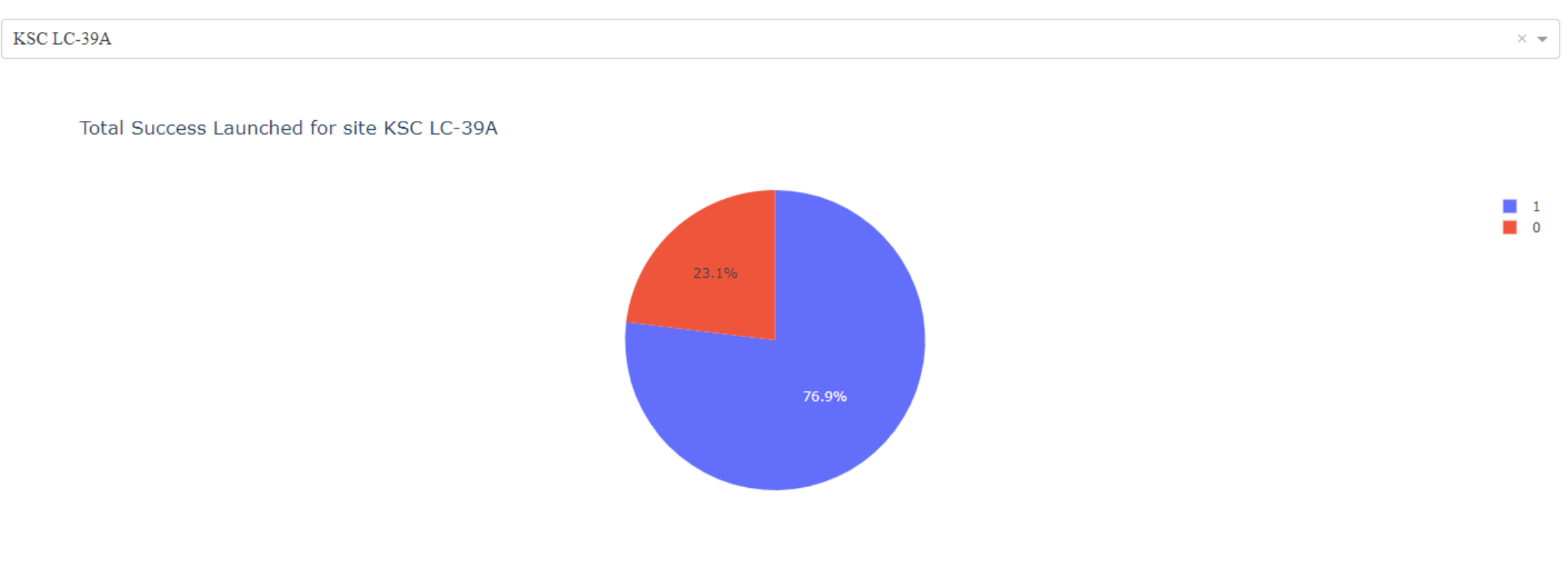

Section 4
Build a Dashboard with Plotly Dash

# <Dashboard Screenshot 1>



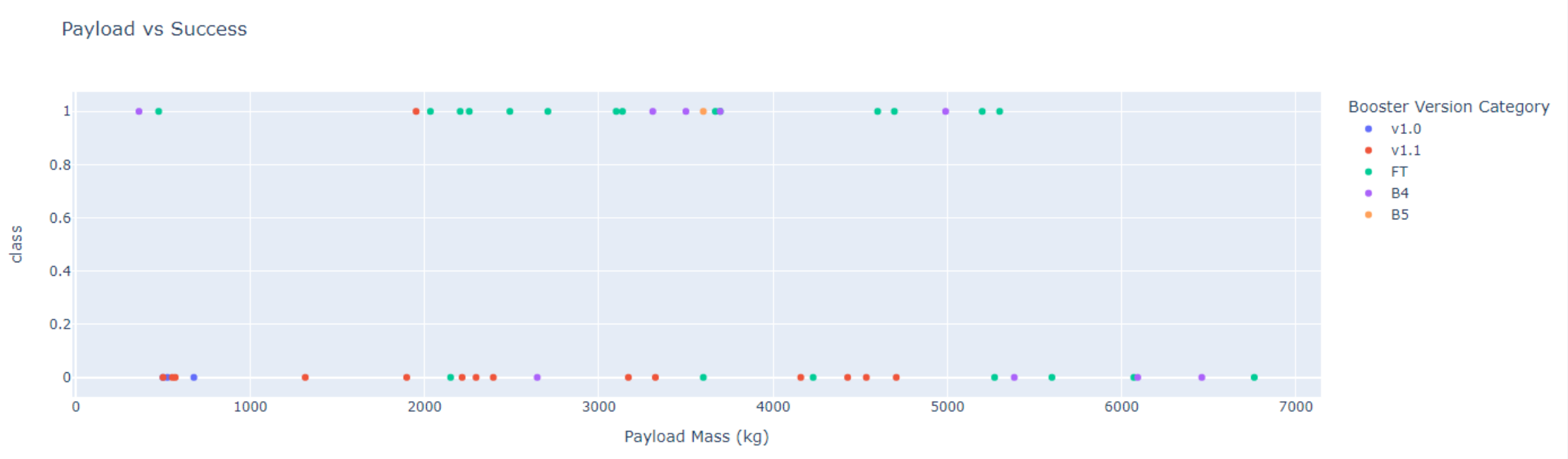

- The four launches from different sites can be seen as above.

# <Dashboard Screenshot 2>




- Highest rate is kcl-lc-39a as 76,9%

# <Dashboard Screenshot 3>




- Payload vs success rate can bwe seen as above, v1.0 has highest success rate

Section 5

# Predictive Analysis (Classification)

# Classification Accuracy

```
performances= pd.DataFrame({"logreg_cv": [logreg_cv.best_params_, logreg_cv.best_score_],
                            "svm_cv": [svm_cv.best_params_, svm_cv.best_score_],
                            "tree_cv": [tree_cv.best_params_, tree_cv.best_score_],
                            "knn_cv": [knn_cv.best_params_, knn_cv.best_score_]}, index= ["parameters", "best accuracy"])
performances
```

| | logreg_cv | svm_cv | tree_cv | knn_cv |
|---|---|---|---|---|
| **parameters** | {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'} | {'C': 1.0, 'gamma': 0.03162277660168379, 'kern... | {'criterion': 'entropy', 'max_depth': 10, 'max... | {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1} |
| **best accuracy** | 0.846429 | 0.848214 | 0.876786 | 0.848214 |

```
[tree_cv.best_params_, tree_cv.best_score_]
```

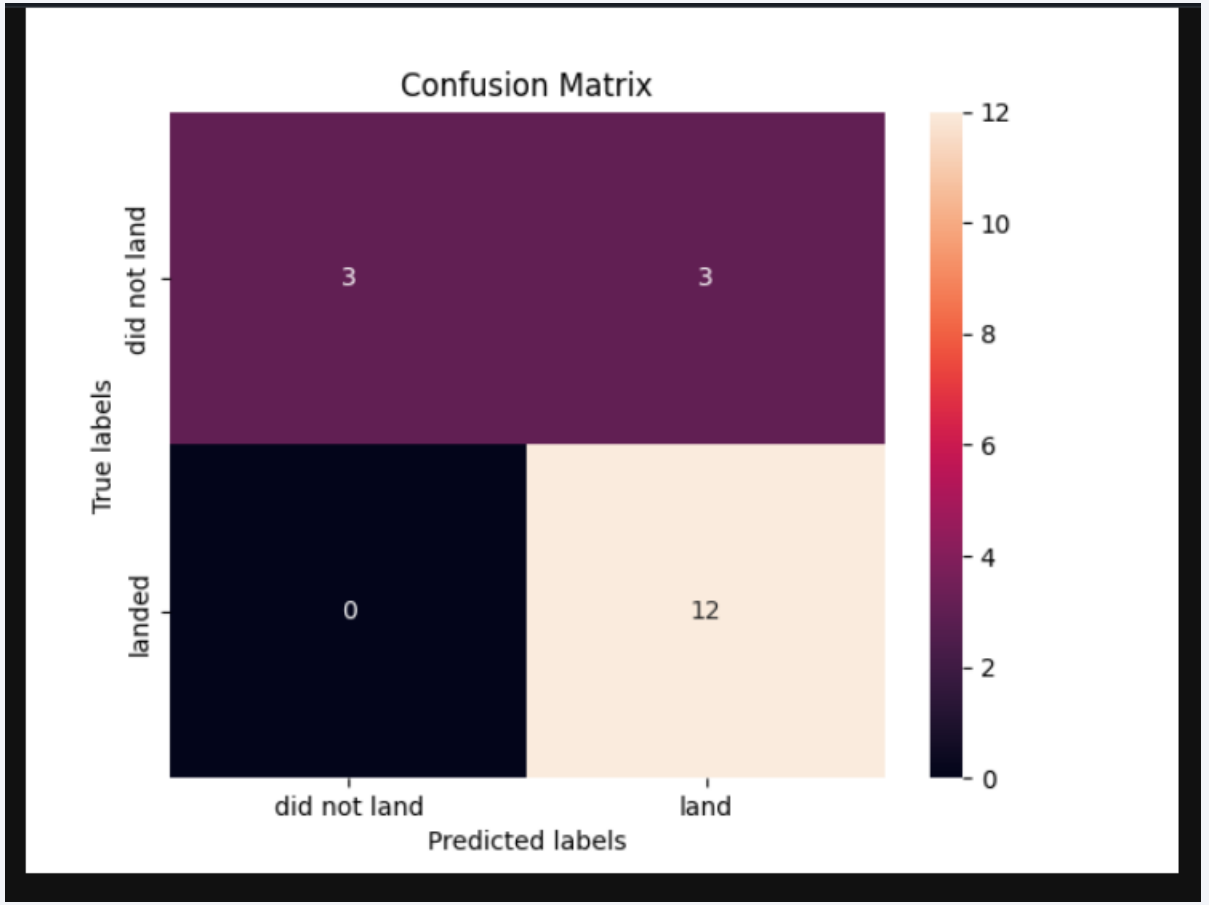
```
[{'criterion': 'entropy',
  'max_depth': 10,
  'max_features': 'sqrt',
  'min_samples_leaf': 4,
  'min_samples_split': 2,
  'splitter': 'random'},
 0.8767857142857143]
```

- Four model accuracy scores are defined as above

# Confusion Matrix

- TP: 3, TN: 12, FP: 3, FN: 0

- So, the accuracy rate for overall performenca is 83.3%



# Conclusions

- We have gathered information from both SpaceX API and a Wikipedia webpage. Our visualizations utilized Folium and Dash for graphical representation. The data was analyzed using SQL, and subsequently, we developed four distinct machine learning models to assess their accuracy scores.
- The logistic regression, SVM, decision tree, and KNN models all demonstrated comparable accuracy scores exceeding 80%. Among them, the decision tree model achieved the highest accuracy score of 87%, making it the most precise.

# Appendix

- [1]https://github.com/cuneyterem8/certificate_projects/tree/main/ibm_data_science/11_applied_ds_capstone_project
- [2]https://www.coursera.org/learn/applied-data-science-capstone/home/week/5

# Thank you!