



IBM Developer  
SKILLS NETWORK

# Winning Space Race with Data Science

Gregor Baguhin  
January 1, 2023



# Outline

---

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

# Executive Summary

---

- Summary of methodologies
  - Data Collection Methodologies
    - SpaceX Rest API will be used to collect SpaceX launch data
    - Data Wrangling will use JSON objects and Python Beautiful Soup package to web scrape HTML tables for Falcon 9 launch data
  - Exploratory data analysis (EDA) will be performed using visualization with Pandas and Matplotlib
  - Exploratory data analysis (EDA) will be performed using SQL queries on a DB2 database
  - Perform interactive visual analytics using Folium and Plotly Dash
  - Perform predictive analysis four using classification models: Logistic Regression, Support Vector Machines, Decision Tree Classifier and K-Nearest Neighbors

# Executive Summary

---

- Summary of all results:
  - As the number of flights increases, the safe landing rates of first stage also increase
  - The higher the payload mass, the lower the success rate for landing is
  - Landing success was attained 67% of the time
  - Success rates have steadily climbed from 2013 to 2020
  - Orbits with the highest success rates are ES-L1, GEO, HEO, & SSO
  - Launch site CCAFS SLC 40 had the highest number and rate of successful landings
  - All the prediction classification algorithms used yielded the same accuracy of 0.833333

# Introduction

---

- Project background and context
  - Commercial space companies like Virgin Galactic, Rocket Lab, Blue Origin and SpaceX have proliferated in recent years making space travel more affordable
  - SpaceX is perhaps the most successful having sent spacecraft to the International Space Station, launched Starlink internet satellites and sending manned missions to space
  - Much of the reason for SpaceX's success is its Falcon 9 rocket which can reuse its first-stage rockets if they land successfully, making it much cheaper than the others
- Problems you want to find answers
  - Our job is determining the price of each launch of Falcon 9 rockets for a new rocket company SpaceY
  - Using machine learning model on public information, we can make predictions for a successful first stage landing that allows its reuse in order to determine the price of each launch.



Section 1

# Methodology

# Methodology

---

## Executive Summary

- Data collection methodology:
  - SpaceX launch data will be gathered from the SpaceX REST API. We will be working with the endpoint `api.spacexdata.com/v4/launches/past`
  - We will perform a get request using the requests library to obtain the launch data, which we will use to get the data from the API.
  - Our response will be in the form of a JSON, specifically a list of JSON objects.
- Perform data wrangling
  - We have a list of JSON objects which each represent a launch
  - To convert this JSON to a data frame, we can use the `json_normalize` function which will allow us to “normalize” the structured json data into a flat table

# Methodology

---

## Executive Summary

- Perform data wrangling (cont'd)
  - We will be using the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records
  - We need to parse the data from those tables and convert them into a Pandas data frame for further visualization and analysis
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Perform EDA using Pandas and Matplotlib for visualization
  - Perform EDA using SQL queries on a DB2 database
- Perform interactive visual analytics using Folium and Plotly Dash
  - We will use Folium and Plotly Dash to build an interactive map and dashboard to perform interactive visual analytics



# Methodology

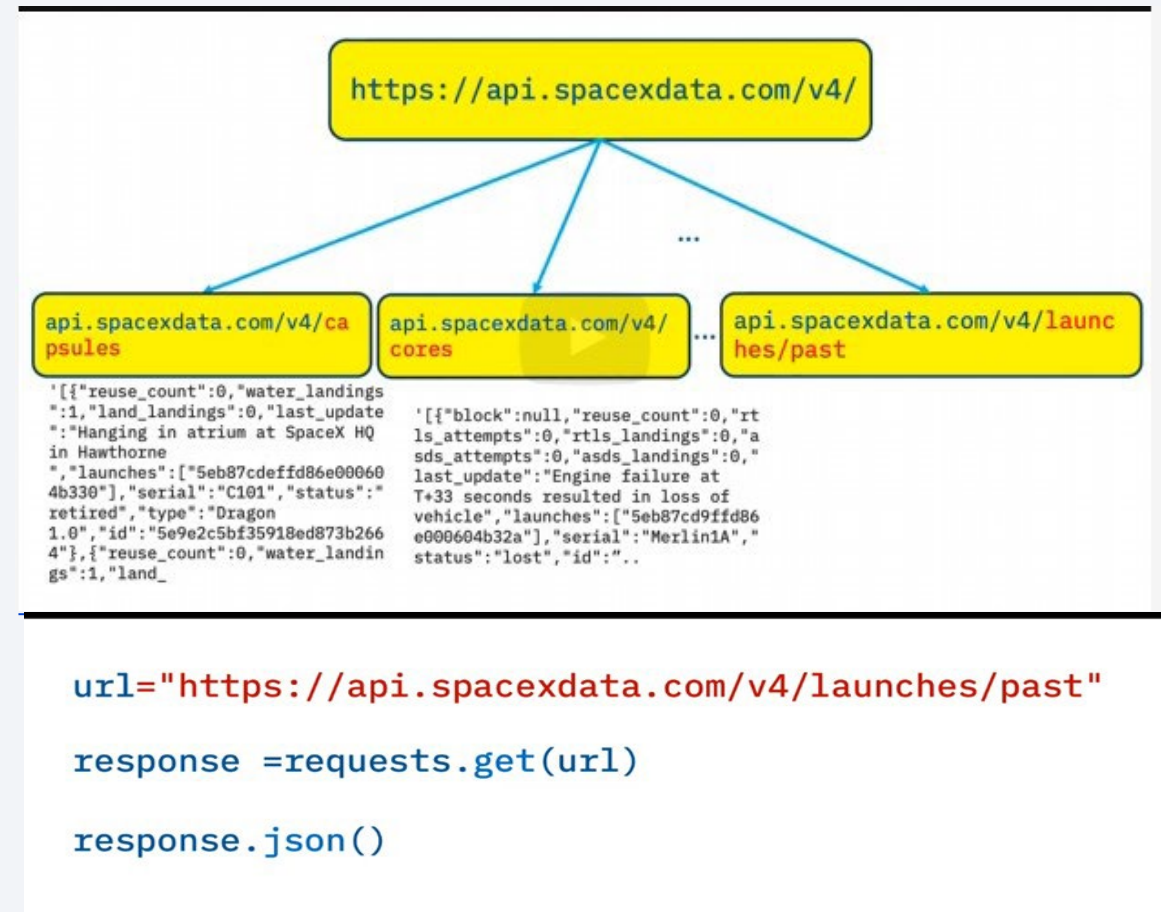
---

## Executive Summary

- Perform predictive analysis using classification models
  - We will build a machine learning pipeline to predict if the first stage of the Falcon 9 lands successfully by doing the following:
    - Preprocessing, allowing us to standardize our data
    - Train\_test\_split, allowing us to split our data into training and testing data
    - We will train the model and perform Grid Search
    - We will determine the model with the best accuracy using the training data
    - We will test Logistic Regression, Support Vector machines, Decision Tree Classifier, and K-nearest neighbors
    - We will output the Confusion Matrix

# Data Collection – SpaceX API

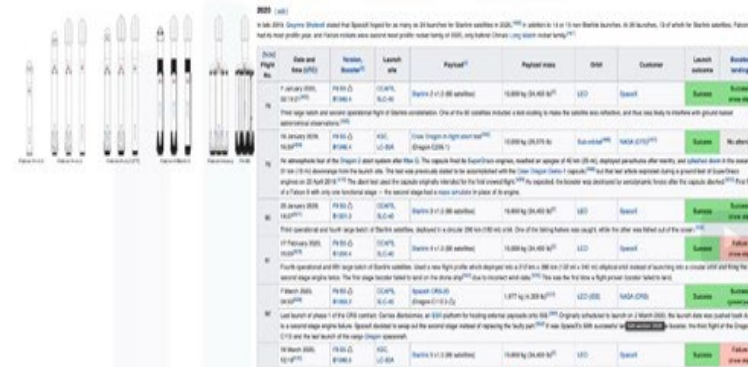
- Launch data collection with SpaceX REST calls to different API endpoints named capsules, cores, and launches/past
- Using the Request object's get method, data is viewed as a list of JSON objects through the Response object's json() method
- <https://github.com/gbaguhin/python-for-datascience/blob/main/jupyter-labs-spacex-data-collection-api.ipynb>



# Data Collection - Scraping

- Using the Python BeautifulSoup package to web-scrape HTML tables that contain Falcon 9 launch data
- Data is parsed and converted into Pandas dataframes for further visualization and analysis (see figure on the right)
- [https://github.com/gbaguhin/python\\_for\\_data-science/blob/main/jupyter-labs-webscraping.ipynb](https://github.com/gbaguhin/python_for_data-science/blob/main/jupyter-labs-webscraping.ipynb)

## Web scraping Falcon 9 Launch records



Flight No.	Date and Time (UTC)	Version	Booster	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Booster status
1	2006-03-24	Falcon 1	1.0	LEO	Merlin1A	100 kg	LEO	SpaceX	Success	Reused
2	2007-03-21	Falcon 1	1.0	LEO	Merlin2A	100 kg	LEO	SpaceX	Success	Reused
3	2008-09-28	Falcon 1	1.0	LEO	Merlin2C	100 kg	LEO	SpaceX	Success	Reused
4	2009-07-13	Falcon 1	1.0	LEO	Merlin3C	100 kg	LEO	SpaceX	Success	Reused
5	2010-06-04	Falcon 9	1.0	LEO	Merlin3C	100 kg	LEO	SpaceX	Success	Reused

Web scraping with BeautifulSoup

FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude		
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None	None	1	False	False	False	None	NaN	0	Merlin1A	167.743129	9.047721
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None	None	1	False	False	False	None	NaN	0	Merlin2A	167.743129	9.047721
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None	None	1	False	False	False	None	NaN	0	Merlin2C	167.743129	9.047721
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None	None	1	False	False	False	None	NaN	0	Merlin3C	167.743129	9.047721
4	6	2010-06-04	Falcon 9	NaN	LEO	CCAFS SLC 40	None	None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857

# Data Wrangling

- Data is wrangled using an API call and returns in a list of JSON objects
- To convert JSON lists into a Pandas data frame, we use `json_normalize()` method. This will render data into a flat table as shown on the figure on the right
- We will use the API again to collect specific data for each ID number on the table, Several functions are available to use to collect booster, launchpad, payload and core data also shown on the right.
- [https://github.com/gbaguhin/python\\_for\\_data\\_science/blob/main/IBM-DS0321EN-SkillsNetwork\\_labs\\_module\\_1\\_L3\\_labs-jupyter-spacex-data\\_wrangling\\_jupyterlite.jupyterlite.ipynb](https://github.com/gbaguhin/python_for_data_science/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_1_L3_labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb)

## Wrangling Data using an API

```
data = pd.json_normalize(response.json())
```

[illegible]

Function	Targets	Endpoint
getBoosterVersion	Rockets	URL: <a href="https://api.spacexdata.com/v4/rockets">https://api.spacexdata.com/v4/rockets</a>
getLaunchSite	Launchpads	URL: <a href="https://api.spacexdata.com/v4/launchpads">https://api.spacexdata.com/v4/launchpads</a>
getPayloadData	Payloads	URL: <a href="https://api.spacexdata.com/v4/payloads">https://api.spacexdata.com/v4/payloads</a>
getCoreData	getCoreData	URL: <a href="https://api.spacexdata.com/v4/cores">https://api.spacexdata.com/v4/cores</a>

# EDA with Data Visualization

---

- Charts plotted and why I used those charts:
  - Flight Number vs Payload Mass – scatter point chart generated to find correlation between number of flights and mass of payloads
  - Flight Number vs Launch Site – scatter point chart to find the relationship between the number of continuous flights and launch sites
  - Payload and Launch Site – scatter point chart to visualize the relationship between payload mass and launch site
  - Success Rate vs Orbit Type – bar chart to find which orbits have high success rates
  - Flight Number vs Orbit Type – scatter point chart to visualize relationship between the number of continuous flights and orbit type
  - Payload Mass vs Orbit Type – scatter point chart to reveal the relationship between payload mass and orbit type
  - Year vs Average Success Rate – line chart to visualize the average launch success trend
- [https://github.com/gbaguhin/python\\_for\\_datascience/blob/main/EDA%20with%20Data%20Visualization.ipynb](https://github.com/gbaguhin/python_for_datascience/blob/main/EDA%20with%20Data%20Visualization.ipynb)



# EDA with SQL

---

- Summary of SQL queries performed:
  - `%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXDATASET` - Displays the names of the unique launch sites in the space mission
  - `%sql SELECT * FROM SPACEXDATASET where LAUNCH_SITE LIKE 'KSC%' FETCH FIRST 5 ROWS ONLY;` - Displays 5 records where launch sites begin with the string 'KSC'
  - `%%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXDATASET where CUSTOMER = 'NASA (CRS)';` - Displays the total payload mass carried by boosters launched by NASA (CRS)
  - `%%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXDATASET where Booster_Version = 'F9 v1.1';` - Displays average payload mass carried by booster version F9 v1.1
  - `%sql SELECT MIN(date) FROM SPACEXDATASET where "Landing _Outcome" = 'Success (drone ship)';` - Lists the date where the first successful landing outcome in drone ship was achieved.
  - `%%sql SELECT Booster_Version FROM SPACEXDATASET where "Landing _Outcome" = 'Success (ground pad)' and (PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000);` - Lists the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000

# EDA with SQL

---

- Summary of SQL queries performed:
  - `%%sql SELECT Mission_Outcome, count(Mission_Outcome) as Total FROM SPACEXDATASET Group By Mission_Outcome ;` - Lists the total number of successful and failure mission outcomes
  - `%%sql select Booster_Version from SPACEXDATASET where PAYLOAD_MASS__KG_ = (SELECT max(PAYLOAD_MASS__KG_) FROM SPACEXDATASET);` - Lists the names of the booster\_versions which have carried the maximum payload mass. Use a subquery
  - `%%sql Select Monthname(date) as Month, "Landing _Outcome", Booster_Version, Launch_Site from SPACEXDATASET where (year(date)= '2017' and "Landing _Outcome" = 'Success (ground pad)');` - Lists the records which will display the month names, successful landing\_outcomes in ground pad ,booster versions, launch\_site for the months in year 2017
  - `%%sql select "Landing _Outcome", count ("Landing _Outcome") as "Count of Outcome" from SPACEXDATASET WHERE "Landing _Outcome" like 'Success%' and (date >= '2010-06-04' and date <= '2017-03-20' )group by "Landing _Outcome" order by "Count of Outcome" desc;` - Ranks the count of successful landing\_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.
- [https://github.com/gbaguhin/python\\_for\\_datascience/blob/main/EDA%20with%20SQL.ipynb](https://github.com/gbaguhin/python_for_datascience/blob/main/EDA%20with%20SQL.ipynb)

# Build an Interactive Map with Folium

---

- Map objects created and added to a folium map
  - `folium.Circle`: to add a highlighted circle area with text label to quickly locate and identify a launch site on the map
  - `folium.Marker`: to add Leaflet marker on the map with optional popup text
  - `folium.Popup`: added to folium map to add text description in a popup form
  - `folium.Polyline`: draws lines on the map between given coordinates
- [https://github.com/gbaguhin/python\\_for\\_datascience/blob/main/Interactive%20Visual%20Analytics%20with%20Folium.ipynb](https://github.com/gbaguhin/python_for_datascience/blob/main/Interactive%20Visual%20Analytics%20with%20Folium.ipynb)

# Build a Dashboard with Plotly Dash

---

- Plots/graphs and interactions added to a dashboard
  - Pie Chart: provides a quick visualization of relationships between parameters as they change in value
  - Dropdown List: provides parameter filters that control the behavior of the charts
  - Scatterplot Chart: provides a more granular relationship pattern as parameter scenarios change
  - Range Slider: provides another parameter filter to visually show their effects as they are adjusted, i.e., payload mass effect on launch success or failure
- [https://github.com/gbaguhin/python\\_for\\_datascience/blob/main/spacex\\_launch\\_dash.py](https://github.com/gbaguhin/python_for_datascience/blob/main/spacex_launch_dash.py)

# Predictive Analysis (Classification)

---

- How I built, evaluated, improved, and found the best performing classification model
  - Prepping data
    - Load the data
    - Create a Numpy array from column “Class”
    - Standardize data using preprocessing object
    - Use function `train_test_split` to split the data X and Y into training and test data
  - Logistic Regression
    - Create a logistic regression object then create a `GridSearchCV` object `logreg_cv` with `cv = 10`
    - Fit the object to find the best parameters from the dictionary parameters
    - We display the best parameters using the data and the accuracy on the validation data
    - Calculate the accuracy on the test data using the method `score`
    - Plot the Confusion Matrix



# Predictive Analysis (Classification)

---

- How I built, evaluated, improved, and found the best performing classification model (cont'd)
  - Support Vector Machine
    - Create a support vector machine object then create a GridSearchCV object `svm_cv` with `cv - 10`
    - Fit the object to find the best parameters from the dictionary parameters.
    - Display the best parameters using the data and the accuracy on the validation data
    - Calculate the accuracy on the test data using the method score
    - Plot the Confusion Matrix
  - Decision Tree Classifier
    - Create a decision tree classifier object then create a GridSearchCV object `tree_cv` with `cv = 10`
    - Fit the object to find the best parameters from the dictionary parameters.
    - Display the best parameters using the data and the accuracy on the validation data
    - Calculate the accuracy on the test data using the method score
    - Plot the Confusion Matrix

# Predictive Analysis (Classification)

---

- How I built, evaluated, improved, and found the best performing classification model (cont'd)
  - K Nearest Neighbors
    - Create a k nearest neighbors object then create a GridSearchCV object `knn_cv` with `cv = 10`.
    - Fit the object to find the best parameters from the dictionary parameters.
    - Display the best parameters using the data and the accuracy on the validation data
    - Calculate the accuracy on the test data using the method score
    - Plot the Confusion Matrix

# Predictive Analysis (Classification)

## Model Development Process

Prepping Data	Classification Algorithm	Fit Object	Train Model	Test Accuracy
Create Numpy array for 'Class'	Support Vector Machine			
Standardize data	Decision Tree Classifier	Create GridSearchCV object	Find best parameters & accuracy of training data	Calculate accuracy on test data
train_test_split	K Nearest Neighbors			
	Logistic Regression			
				Plot Confusion Matrix

```
graph LR; A[Standardize data] --> B[Decision Tree Classifier]; C[train_test_split] --> D[K Nearest Neighbors]; B --> E[Create GridSearchCV object]; D --> E; E --> F[Find best parameters & accuracy of training data]; F --> G[Calculate accuracy on test data]; G --> H[Plot Confusion Matrix];
```

# Results

---

- Exploratory data analysis results
  - As the number of flights increases, the safe landing rates of first stage also increase
  - The higher the payload mass, the lower the success rate for landing is
  - Different launch sites have different success rates
  - In all three launch sites, the higher the number of flights the higher the success rate is
  - In all three launch sites, the higher the payload mass, the lower the success rate is
  - Orbits with the highest success rates are ES-L1, GEO, HEO, & SSO
  - In the LEO orbit, success appears related to the number of flights
  - No relationship between success and number of flights in GEO orbit
  - With heavy payloads the successful landing or positive landing rate are more for Polar, LEO, and ISS.

# Results

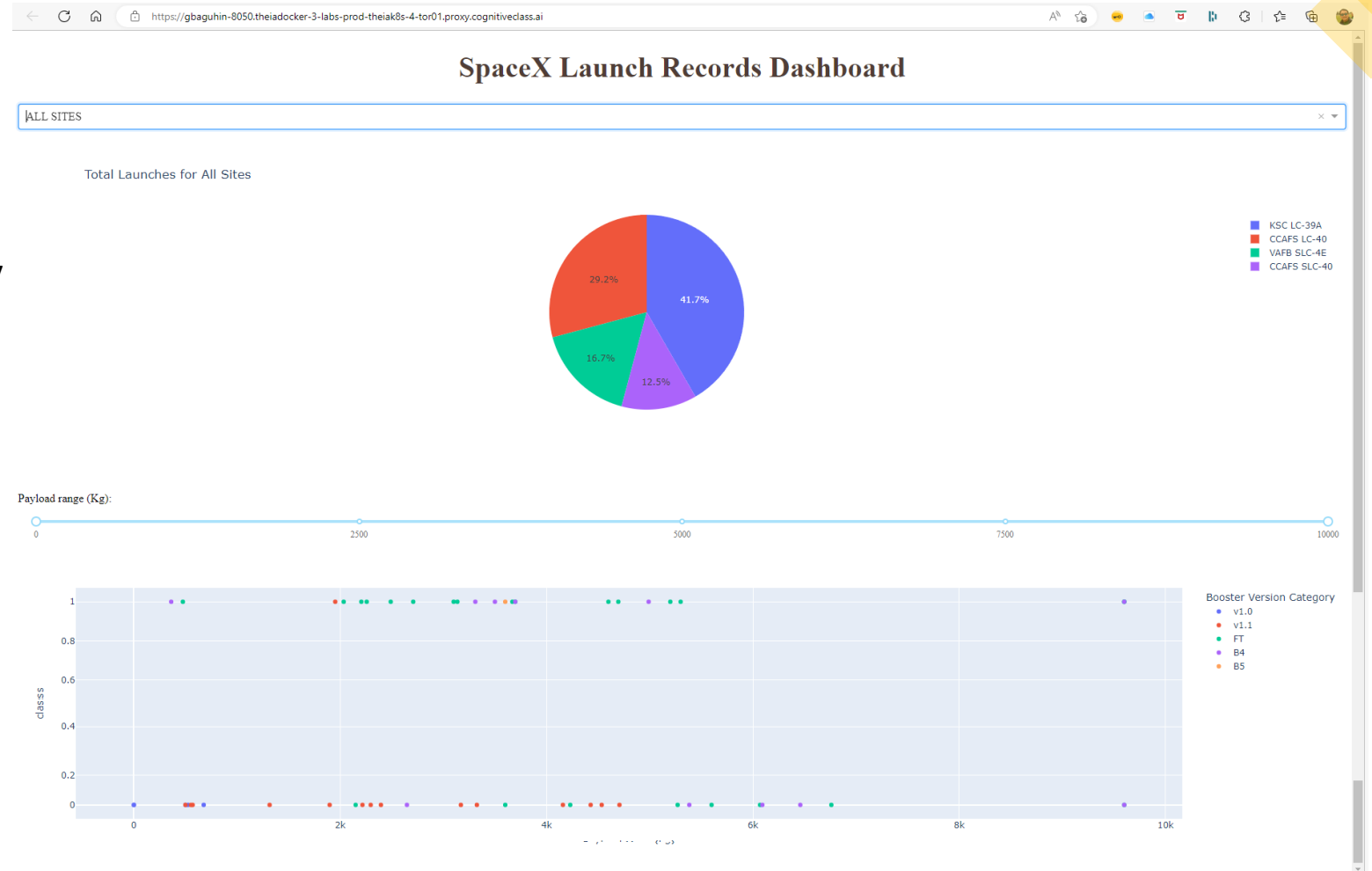
---

- Exploratory data analysis results (cont'd)
  - For GTO we cannot distinguish this well as both positive landing rate and negative landing (unsuccessful mission) are both there.
  - Success rates have climbed from 2013 to 2020



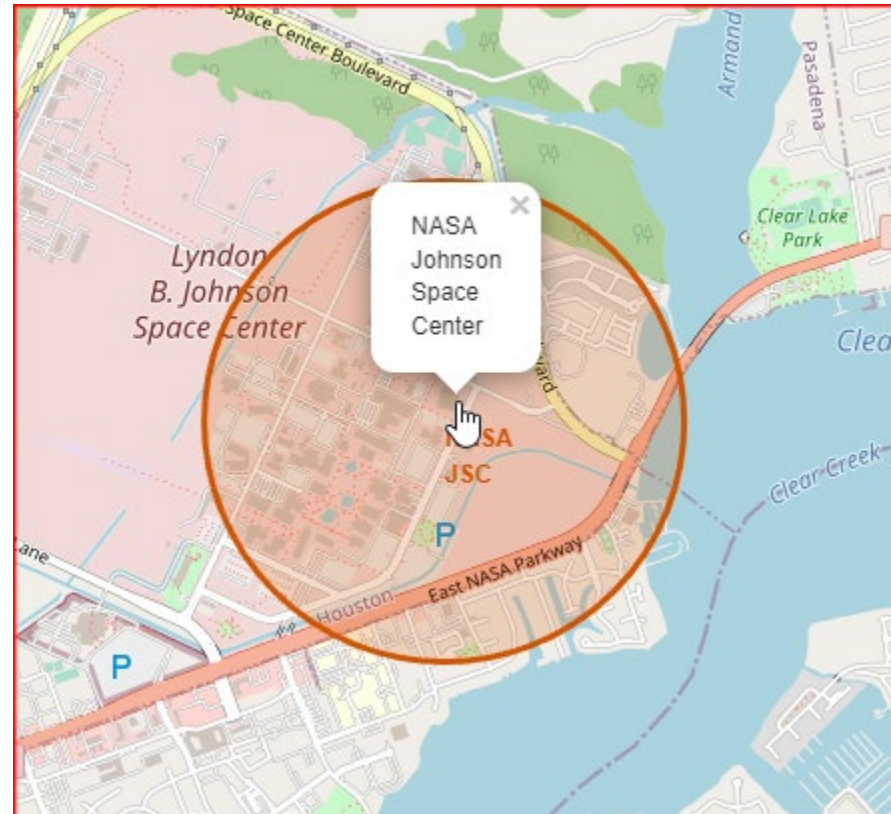
# Results

- Interactive analytics demo in screenshots
  - Interactive Dashboard with Plotly Dash



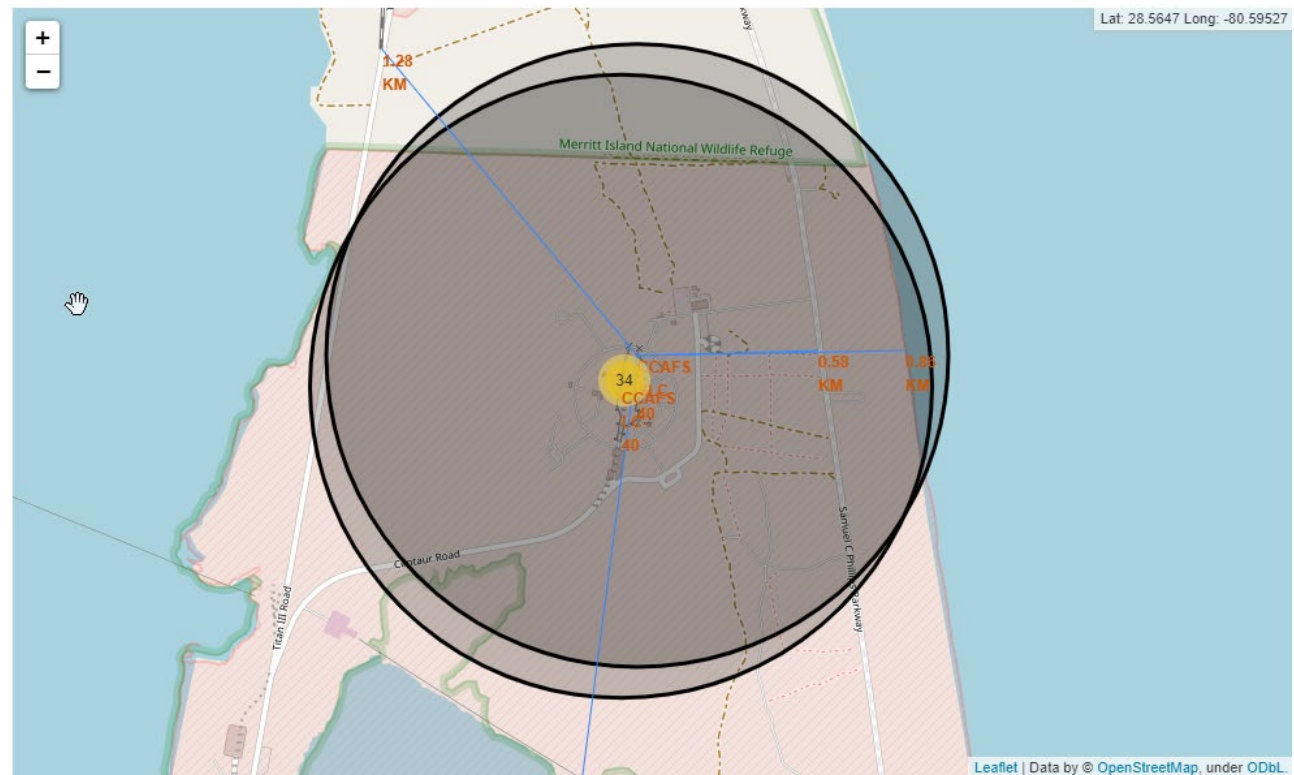
# Results

- Interactive analytics demo in screenshots
  - Folium Map of the Johnson Space Center in Houston, Texas
  - Folium Circle
  - Folium Marker



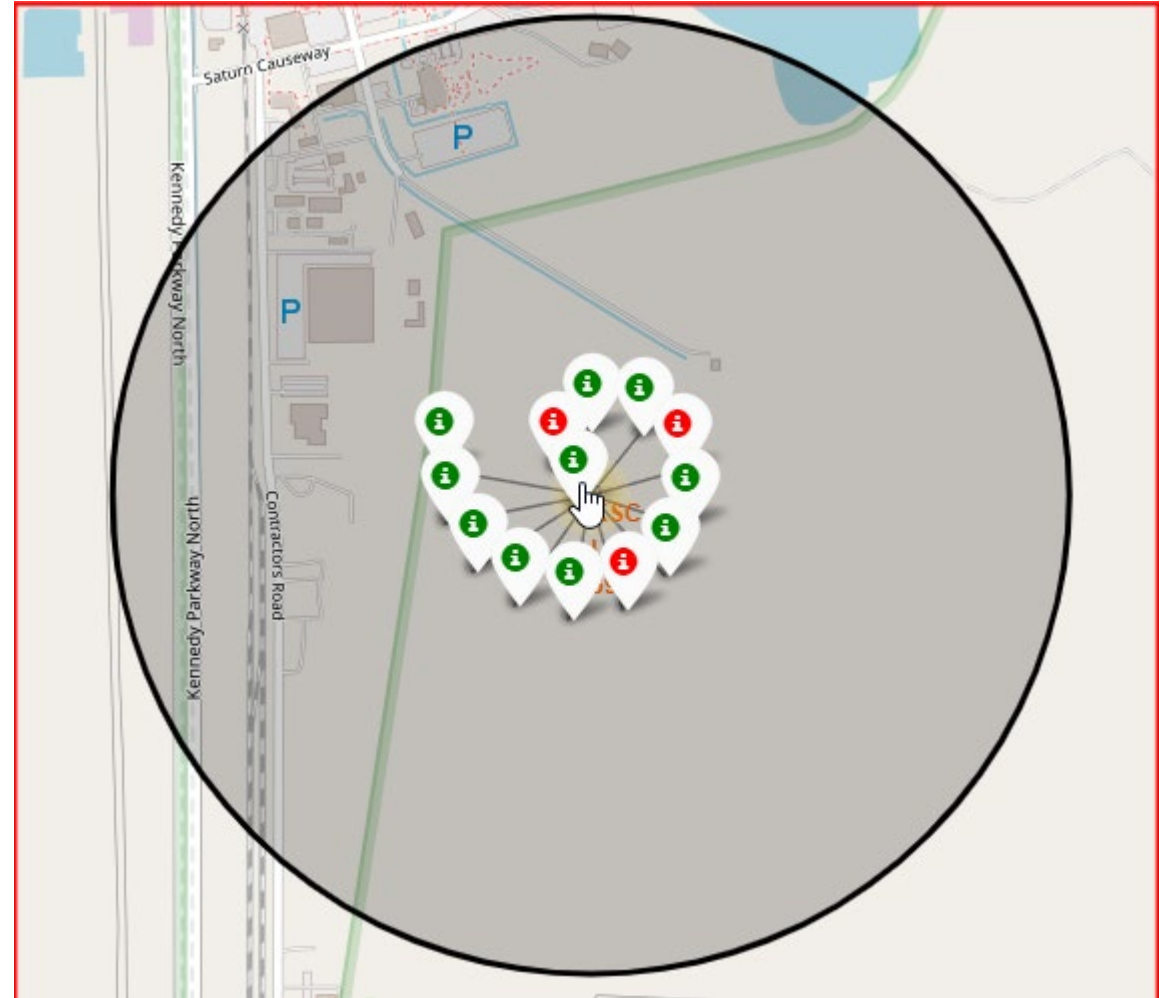
# Results

- Interactive analytics demo in screenshots
  - Folium Map
  - Folium Circle
  - Folium Polyline



# Results

- Interactive analytics demo in screenshots
  - Folium Map
  - Folium Circle
  - Folium Marker Cluster



# Results

- Predictive analysis results

- Logistic regression:

- `yhat=logreg_cv.predict(X_test)`

- `logreg_cv.score(X_test, Y_test)`

- 0.8333333333333334

- Vector Machine Object:

- `svm_cv.score(X_test, Y_test)`

- 0.8333333333333334

- Decision Tree Classifier:

- `tree_cv.score(X_test,Y_test)`

- 0.8333333333333334



# Results

- Predictive analysis results

- K Nearest Neighbors:

```
knn_cv.score(X_test,Y_test)
```

```
0.8333333333333334
```

All these algorithms give the same result



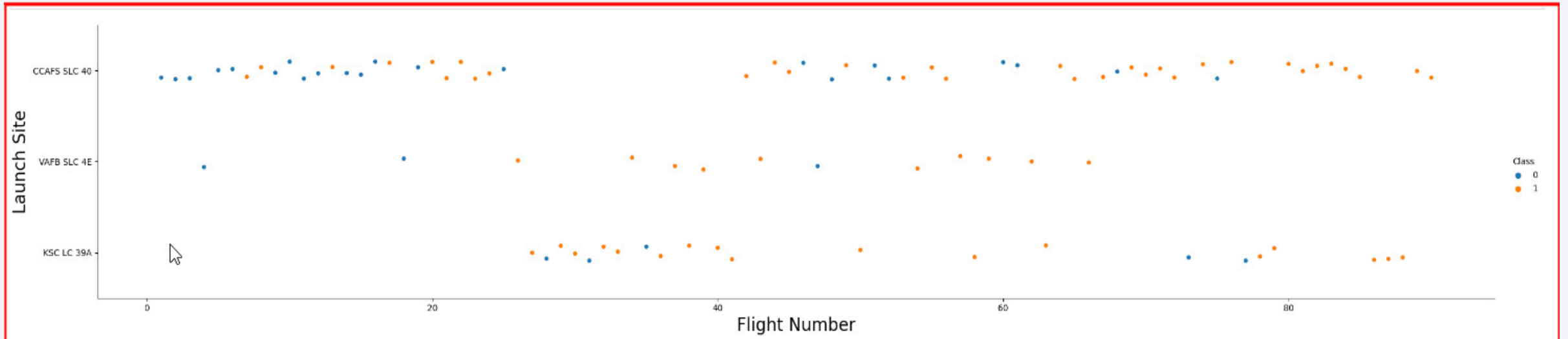
The background of the slide is an abstract composition. It features a dark blue base color. Overlaid on this are numerous diagonal streaks in shades of red and cyan. A faint, light blue grid pattern is also visible, particularly in the lower half of the image. The overall effect is dynamic and technological.

Section 2

# Insights drawn from EDA

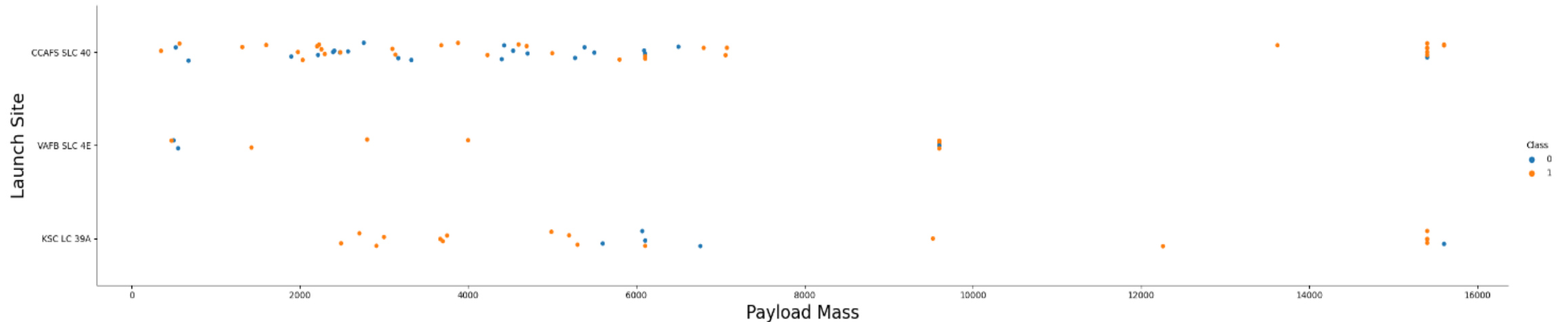


# Flight Number vs. Launch Site



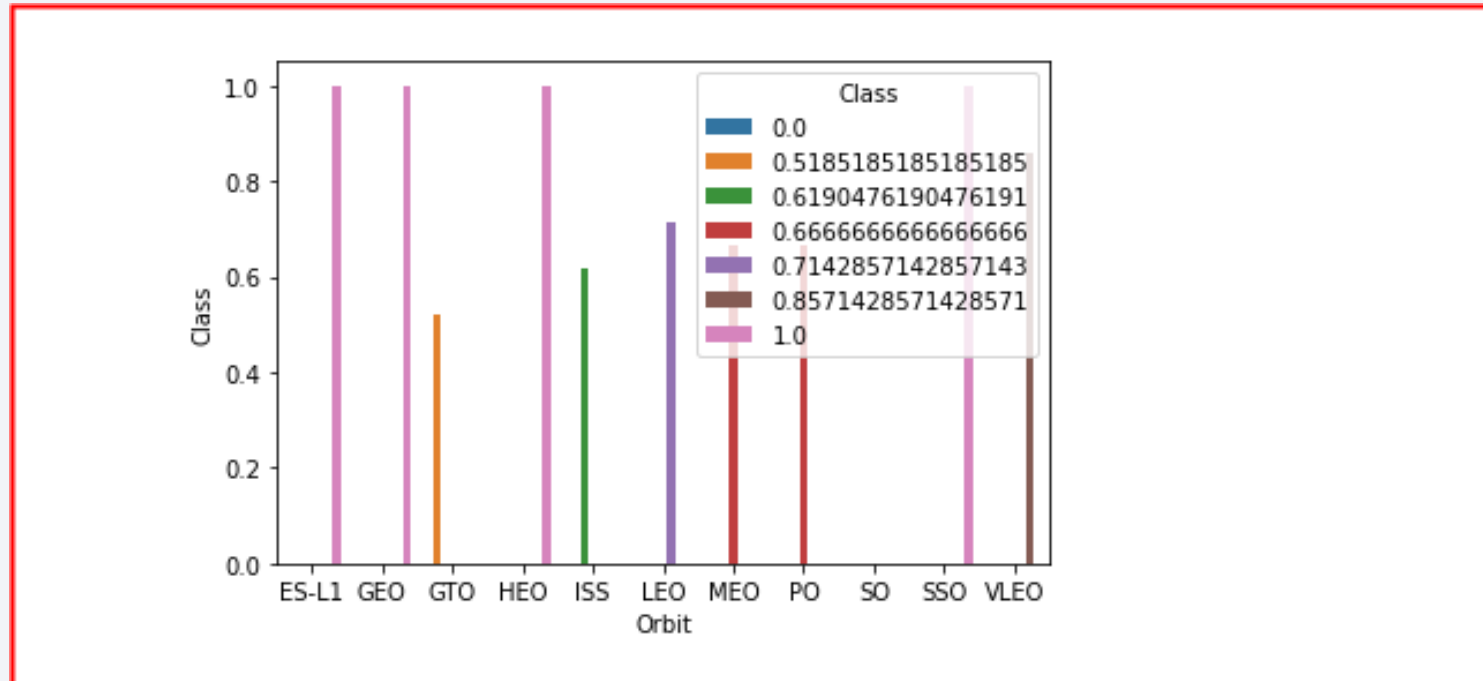
The scatter plot shows launch site CCAFS SLC 40 had the highest number of successful landings followed by KSC LC 39A and VAFB SLC 4E in descending order. It also shows more successful landings were made in the later flights apparently due to more landing practice and experience.

# Payload vs. Launch Site



The scatter plot of Payload vs. Launch Site above shows launch site KSC LC 39A had the highest landing attempts and successes. Site CCAFS SLC 40, follows and Site VAFB SLC 4E, which did not have any landing payload mass above 10000, came in last.

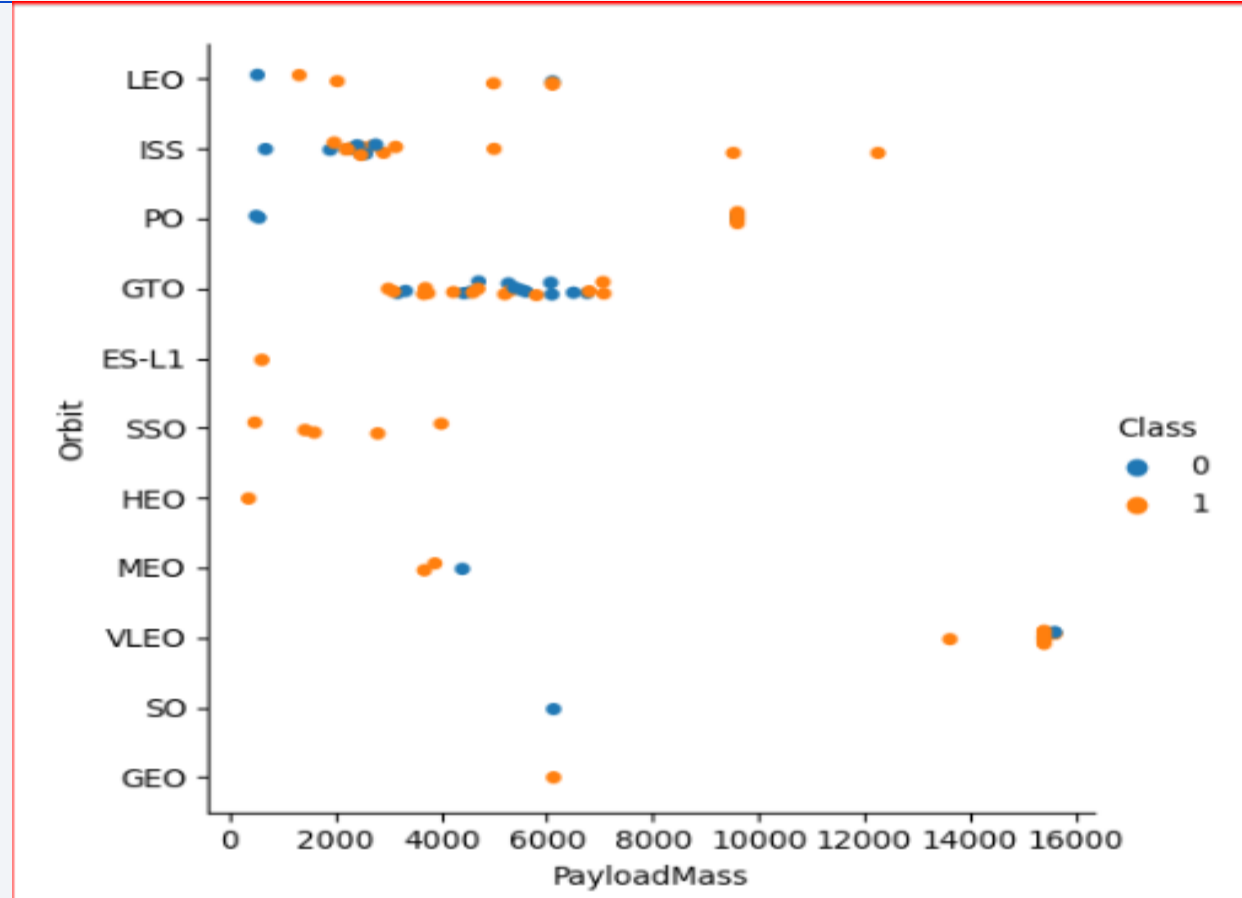
# Success Rate vs. Orbit Type



The bar graph shows the orbits which have 100% success rate, i.e., ES-L1, GEO, HEO, & SSO orbit types but doesn't show that GTO, ISS, & VLEO orbits have the highest number of landing attempts.



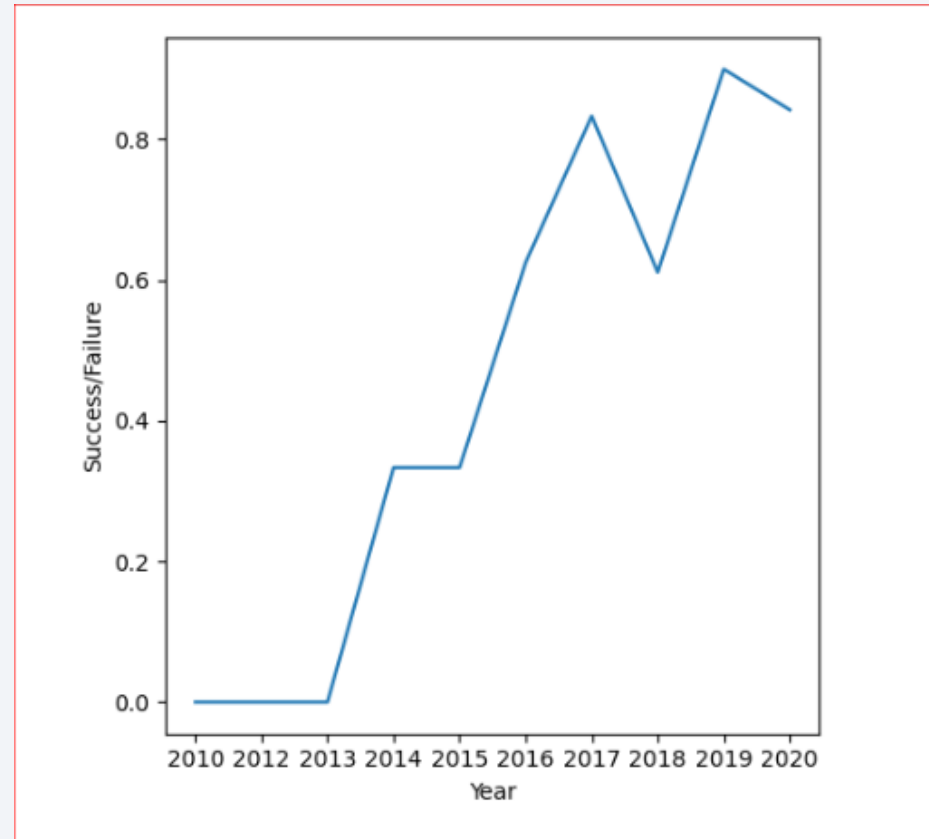
# Payload Mass vs. Orbit Type



The scatter plot above shows most launches are with payload masses under 8000 kgs. However, higher success rates were found for launches with payload masses over 6000 notably in LEO, ISS and Polar orbits.

# Launch Success Yearly Trend

---



The line graph shows an annual rising success trend since 2013 dipping slightly around 2018 but bouncing right back in 2019 and 2020.



# All Launch Site Names

---

Names of the unique launch sites:

```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXDATASET
```

Query result: There are 4 launch sites for SpaceX flights. Three are in Florida and one in California.

```
In [13]: %sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXDATASET
```

```
* ibm_db_sa://szd02040:***@b70af05b-76e4-4bca-a1f5-2  
Done.
```

```
Out[13]:
```

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

# Launch Site Names Begin with 'KSC'

Top 5 records where launch sites' names start with `KSC`:

```
%sql SELECT * FROM SPACEXDATASET where LAUNCH_SITE LIKE 'KSC%'FETCH FIRST 5 ROWS ONLY;
```

Query result: Shows first 5 records out of 25 launches from the Kennedy Space Center in Florida

```
In [14]: %sql SELECT * FROM SPACEXDATASET where LAUNCH_SITE LIKE 'KSC%'FETCH FIRST 5 ROWS ONLY;
```

```
* ibm_db_sa://szd02040:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/bludb
Done.
```

```
Out[14]:
```

DATE	Time (UTC)	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	Landing_Outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-03-16	06:00:00	F9 FT B1030	KSC LC-39A	EchoStar 23	5600	GTO	EchoStar	Success	No attempt
2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
2017-05-01	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
2017-05-15	23:21:00	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070	GTO	Inmarsat	Success	No attempt

# Total Payload Mass

---

Total payload carried by boosters from NASA

```
%%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXDATASET  
where CUSTOMER = 'NASA (CRS)';
```

Query result: Payload mass total for NASA boosters totaled 45596 kgs distributed over 20 launches.

```
In [15]: %%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXDATASET  
         where CUSTOMER = 'NASA (CRS)';  
  
         * ibm_db_sa://szd02040:***@b70af05b-76e4-4bca-a1f5-23db  
         Done.  
  
Out[15]: 1  
         45596
```

# Average Payload Mass by F9 v1.1

---

Average payload mass carried by booster version F9 v1.1

```
%%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXDATASET
```

```
where Booster_Version = 'F9 v1.1';
```

Query result: Booster version F9 v1.1 was used in 5 launches with an average payload of 2928 kgs.

```
In [16]: %%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXDATASET
         where Booster_Version = 'F9 v1.1';

         * ibm_db_sa://szd02040:***@b70af05b-76e4-4bca-a1f5-23db
         Done.

Out[16]: 1
         2928
```

# First Successful Ground Landing Date

---

Dates of the first successful landing outcome on ground pad

```
%sql SELECT MIN(date) FROM SPACEXDATASET where "Landing _Outcome" = 'Success (ground pad)';
```

Query result: First successful landing on a ground pad was on May 1, 2017, at launch site KSC LC-39A carrying a NROL-76 payload for NRO.

Date ▼	Launch_Site ▼	Payload ▼	Customer ▼	Booster_Ve ▼
01-05-2017	KSC LC-39A	NROL-76	NRO	F9 FT B1032.1

## Successful Drone Ship Landing with Payload between 4000 and 6000

---

Names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000:

```
%%sql SELECT Booster_Version FROM SPACEXDATASET  
where "Landing _Outcome" = 'Success (drone ship)'  
and (PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000);
```

Query result:

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

# Total Number of Successful and Failure Mission Outcomes

---

Total number of successful and failure mission outcomes:

```
%%sql SELECT Mission_Outcome, count(Mission_Outcome) as Total FROM SPACEXDATASET  
Group By Mission_Outcome ;
```

Query result: The result shows an almost 100% successful mission outcomes. In comparison, the rate for landing the first stage safely that has an almost a 2 to 1 success-to-failure ratio.

```
%%sql SELECT Mission_Outcome, count(Mission_Outcome) as Total FROM SPACEXDATASET  
Group By Mission_Outcome ;
```

```
* ibm_db_sa://szd02040:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu01q  
Done.
```

mission_outcome	total
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

# Boosters Carried Maximum Payload

---

Names of the booster which have carried the maximum payload mass:

```
%%sql select Booster_Version from SPACEXDATASET  
where PAYLOAD_MASS__KG_ = (SELECT max(PAYLOAD_MASS__KG_) FROM SPACEXDATASET);
```

Query results: Results show 12 boosters which have carried the maximum payload mass.

```
%%sql select Booster_Version from SPACEXDATASET  
where PAYLOAD_MASS__KG_ = (SELECT max(PAYLOAD_MASS__KG_) FROM SPACEXDATASET);
```

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



# 2017 Launch Records

Records which will display the month names, successful landing\_outcomes in ground pad, booster versions, launch\_site for the months in year 2017:

```
%%sql Select Monthname(date) as Month, "Landing _Outcome", Booster_Version, Launch_Site from  
SPACEXDATASET where (year(date)= '2017' and "Landing _Outcome" = 'Success (ground pad)');
```

Query results: Results show six successful landing outcomes in 2017. Five were launched from Kennedy Space Center and one was launched from Cape Canaveral.

```
%%sql Select Monthname(date) as Month, "Landing _Outcome", Booster_Version, Launch_Site from SPACEXDATASET  
where (year(date)= '2017' and "Landing _Outcome" = 'Success (ground pad)');
```

MONTH	Landing _Outcome	booster_version	launch_site
February	Success (ground pad)	F9 FT B1031.1	KSC LC-39A
May	Success (ground pad)	F9 FT B1032.1	KSC LC-39A
June	Success (ground pad)	F9 FT B1035.1	KSC LC-39A
August	Success (ground pad)	F9 B4 B1039.1	KSC LC-39A
September	Success (ground pad)	F9 B4 B1040.1	KSC LC-39A
December	Success (ground pad)	F9 FT B1035.2	CCAFS SLC-40

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

---

Rank of the count of successful landing\_outcomes between the date 2010-06-04 and 2017-03-20 in descending order:

```
%%sql select "Landing _Outcome", count ("Landing _Outcome") as "Count of Outcome" from  
SPACEXDATASET WHERE "Landing _Outcome" like 'Success%'  
and (date >= '2010-06-04' and date <= '2017-03-20' )  
group by "Landing _Outcome" order by "Count of Outcome" desc;
```

Query results: Results show that drone ship landing had 5 successful outcomes while ground pad landing had 3.

```
%%sql select "Landing _Outcome", count ("Landing _Outcome") as "Count of Outcome" from SPACEXDATASET  
WHERE "Landing _Outcome" like 'Success%'  
and (date >= '2010-06-04' and date <= '2017-03-20' )  
group by "Landing _Outcome"  
order by "Count of Outcome" desc;
```

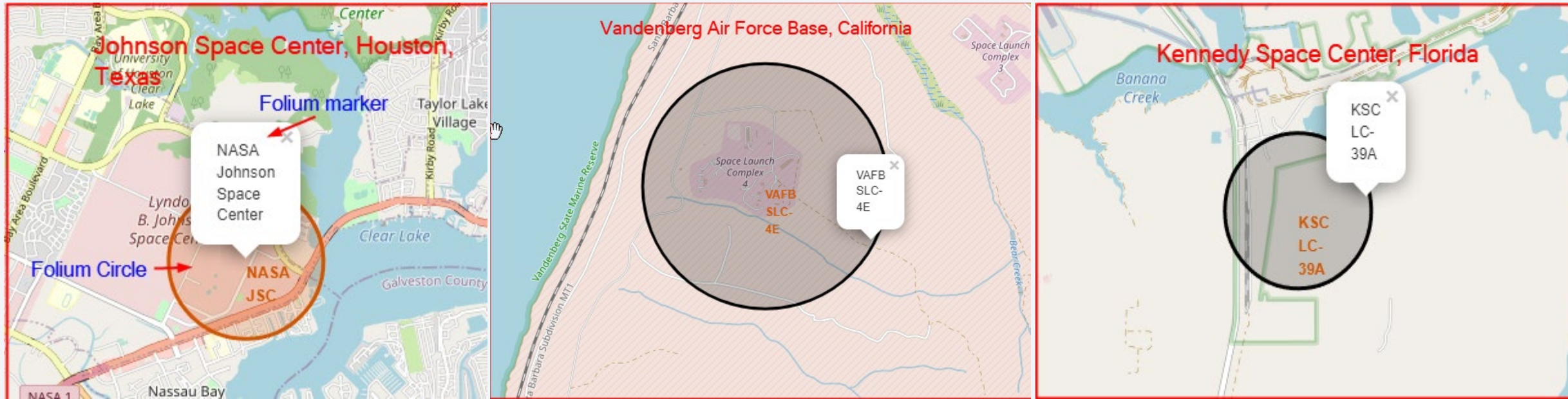
Landing _Outcome	Count of Outcome
Success (drone ship)	5
Success (ground pad)	3

A satellite view of Earth from space, showing the curvature of the planet and the glow of city lights at night. The background is a deep blue, and the Earth's surface is a mix of dark blue and bright yellow/orange lights.

Section 3

# Launch Sites Proximities Analysis

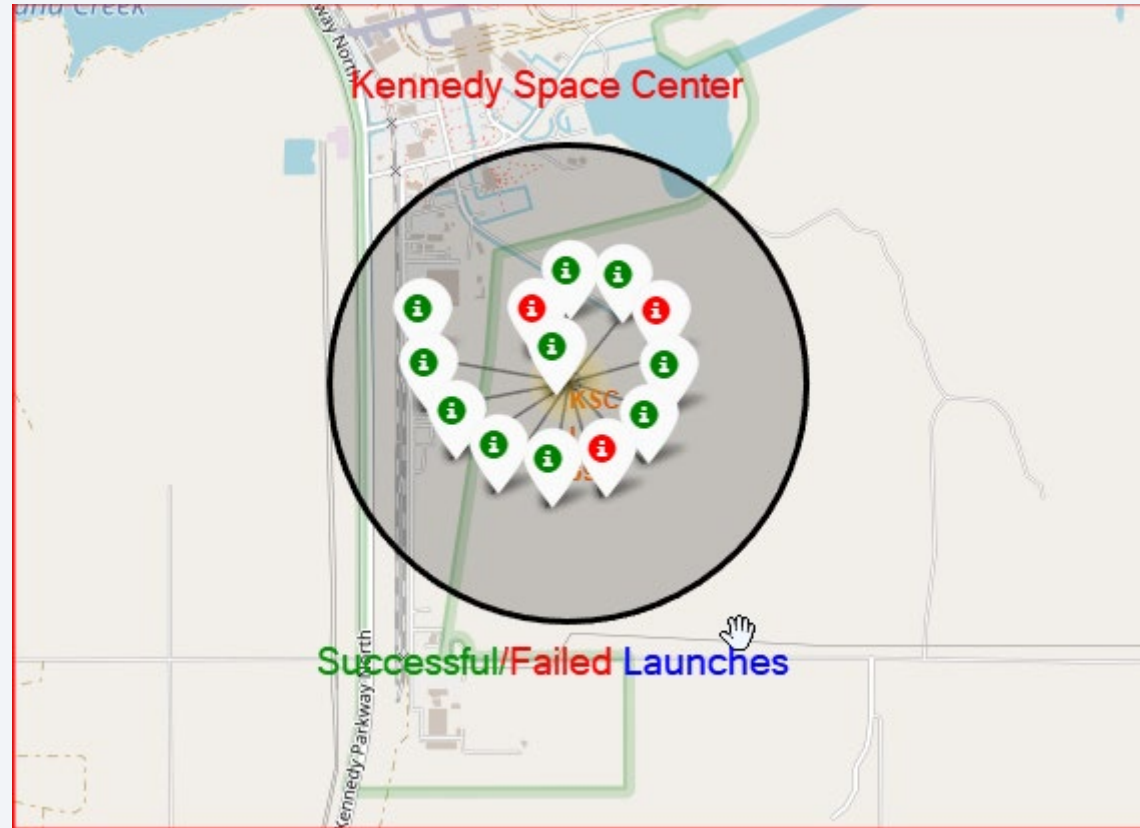
# Marking Launch Sites on a Map using Folium



Launch sites in Texas, California and Florida are shown on a folium map enhanced by a highlighted circle created by folium Circle function and a text label created by a folium Marker function.

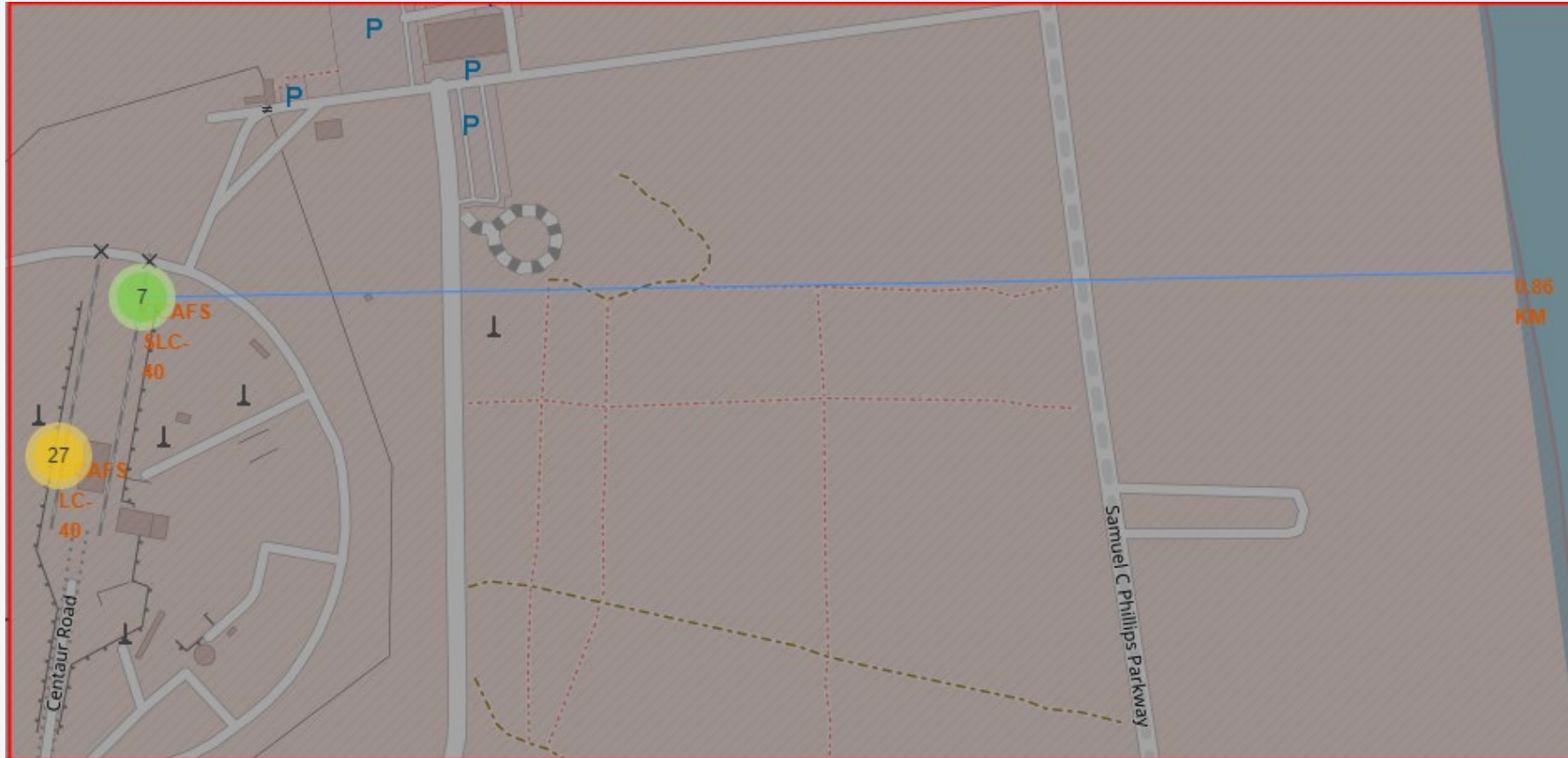


# Marking Successful/Failed Launches by Color



Shown above is a folium Map of the Kennedy Space Center launch site further enhanced by a color-coded launch outcomes using the folium MarkerCluster object. In the example above, green represents the successful launches and red are the failed launches.

# Calculate distance between a launch site and nearby landmarks



Using Folium Maps MousePosition function and coordinates, one can calculate distances between launch sites and proximities like coastlines, railway tracks, highways, etc. In the screenshot above launch site CCAFS SLC 40 is about 0.90 km from the coastline.



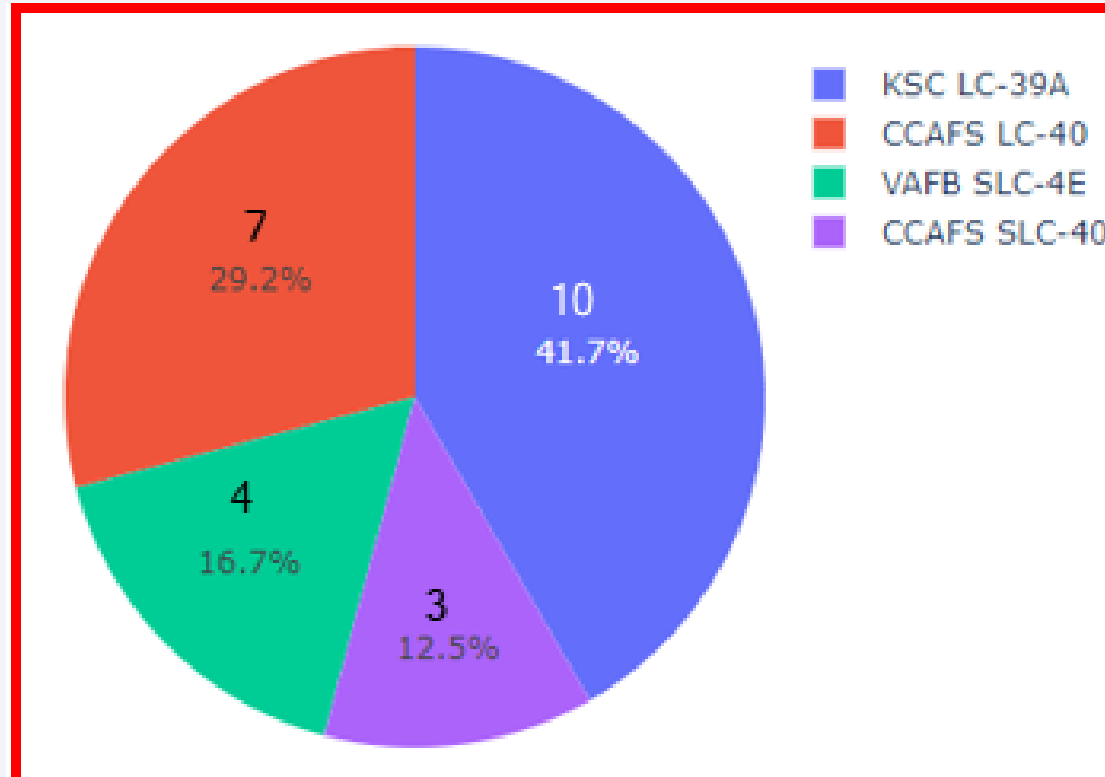
Section 4

# Build a Dashboard with Plotly Dash



# Launch Success Counts & Percentage for All Sites

---

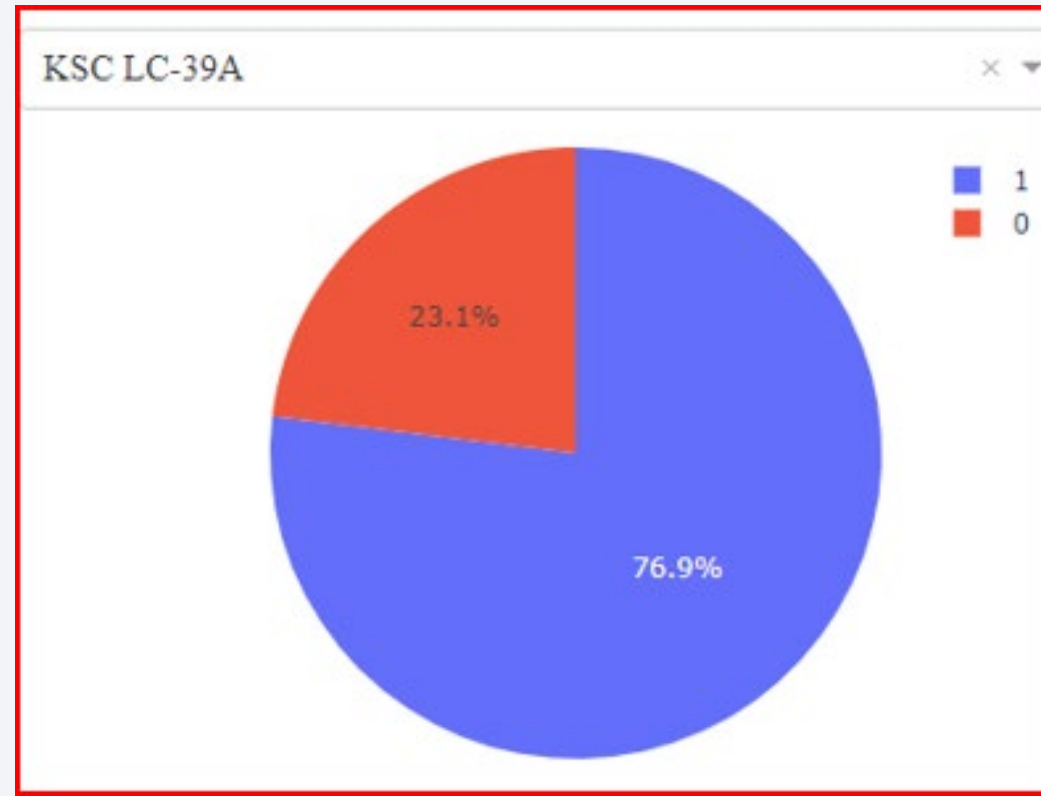


This pie chart shows each site's count and per cent contribution to total successful launches (24) for all sites. KSC LC-39A topped the list with 41.7% while CCAFS SLC-40 is last with 12.5%.



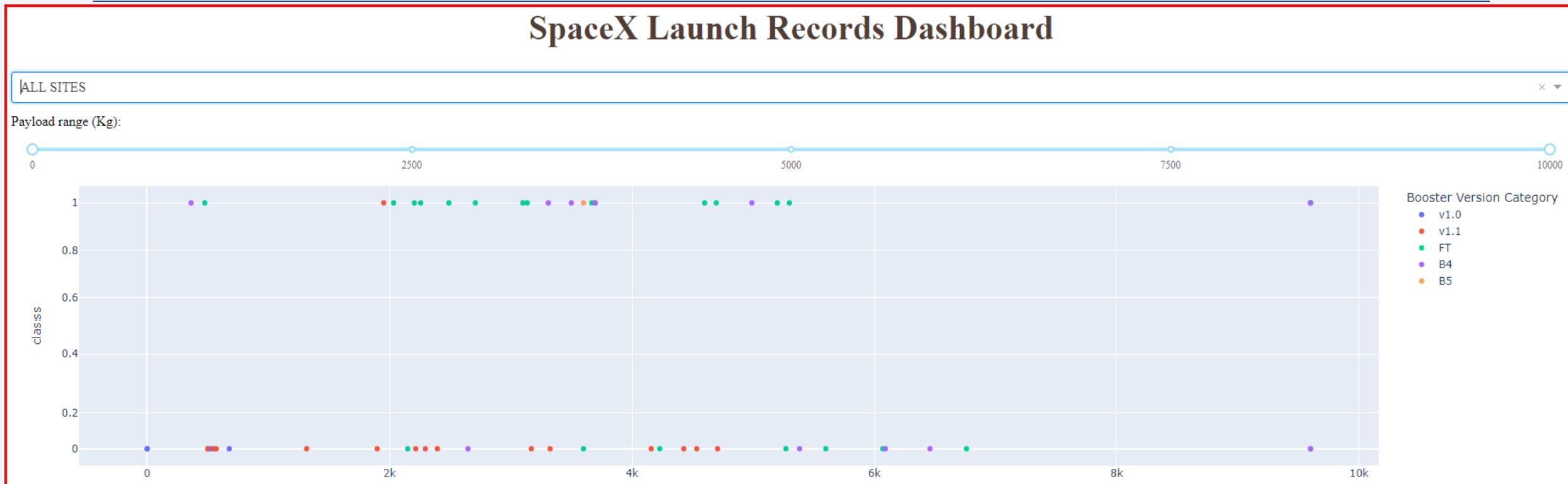
# Launching Site with Highest Launch Success Ratio

---



The pie chart above shows the launch site KSC LC-39A which had the highest launch success ratio of 76.9%. It also had the highest number of successful launches of the four launch sites as shown on the previous slide.

# Payload vs Launch Outcome Scatter Plot



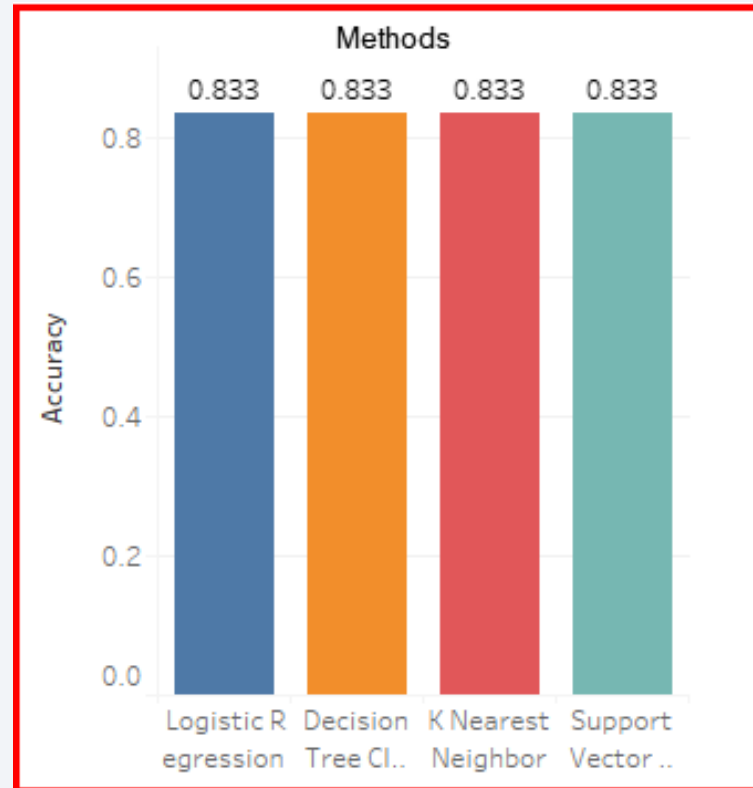
The above screenshot shows Payload vs. Launch Outcome scatter plot for all sites, with a range slider. In the live range slider, it can be visually shown that payload masses lesser than 5000 kgs resulted in more launch success than those with heavier payloads.



Section 5

# Predictive Analysis (Classification)

# Classification Accuracy



The bar graph shows the same accuracies for the four different machine learning classification algorithms. This could be attributed to the relatively small test sample size of 18. Once the sample size increases, then the accuracies will begin to show variations and predictions become more reliable.

# Confusion Matrix



Just like their accuracies, the Confusion Matrix is also the same for all prediction methods. All show that the major problem is false positives. Three are shown on this confusion matrix.

# Conclusions

---

- As the number of flights increases, the safe landing rates of first stage also increase
- The higher the payload mass, the lower the success rate for landing is
- Success rates have steadily climbed from 2013 to 2020
- Orbits with the highest success rates are ES-L1, GEO, HEO, & SSO
- Launch site CCAFS SLC 40 had the highest number and rate of successful landings
- All the prediction classification algorithms used yielded the same accuracy of 0.833333
- The resulting Confusion Matrix was identical to all four classification algorithms employed in the machine learning modeling and testing
- A larger test sample size is needed to give more reliable prediction accuracies.
- Landing success was attained 67% of the time for SpaceX. This means that we can train our data from SpaceY to achieve similar or better landing outcomes thus making the new company as feasible and as profitable as SpaceX.

# Appendix

---

- [https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset\\_part\\_1.csv](https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_1.csv)
- [https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset\\_part\\_2.csv](https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_2.csv)
- [https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex\\_launch\\_geo.csv](https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_geo.csv)
- [https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex\\_launch\\_dash.csv](https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_dash.csv)
- ```
SELECT Dataset_part_2.Class, Count(Dataset_part_2.Class) AS Count Of Class
FROM Dataset_part_2
GROUP BY Dataset_part_2.Class;
```



Thank you!

