IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Anirjit Datta Roy
08th October , 2024

# Outline

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

# Executive Summary

## Summary of methodologies

- Data Collection (API and Scrapping )
- Data Wrangling
- EDA (SQL And Data Visualization)
- Interactive Maps with Folium
- Dashboarding using Plotly and Dash
- Model building and hyperparameter tuning
- Finding Best Model

## Summary of all results

- Results of EDA , Folium , Plotly and Dash
- Result of Predictive analysis

# Introduction

## Project background and context

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch. So , here we will predict if the Falcon 9 first stage will land successfully .

## Problems you want to find answers

- The factors which decide a successful landing
- How landing is dependent on individual input features
- What would be the combination of input features for an ideal landing

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - Data Was collected through SpaceX API and Web scrapping from Wikipedia.
- Perform data wrangling
  - Null handling , One Hot encoding for categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

# Data Collection

Data was collected in 2 ways:

- SpaceX API: We sent request through API . We received the response as a JSON using .json() function and further we converted that to a pandas dataframe with the help of .json_normalize()
- Web Scrapping: We used BeatifulSoup Library to scrape data from Wikipedia. We received the HTML response and after parsing the response , we converted that into pandas dataframe.

In the next slides , we will see in details.

# Data Collection – SpaceX API

- We used SpaceX API to collect data , stored it and did some basic pre-processing (cleaning , formatting etc.)

- Notebook: [SpaceX API]

In [25]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [26]:
```
response = requests.get(spacex_url)
```

Check the content of the response

In [27]:
```
#print(response.content)
```

You should see the response contains massive information about SpaceX launches. Next, let's try to discover some more relevant information for this project.

### Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

In [28]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_
```

We should see that the request was successfull with the 200 status response code

In [29]:
```
response.status_code
```

Out[29]: 200

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

In [30]:
```
# Use json_normalize meethod to convert the json result into a dataframe
data=pd.json_normalize(response.json())
data.head()
```

# Data Collection - Scraping

- Using BeatufilSoup Library scraped web data for SpaceX Falcon 9 and further converted the data into a pandas dataframe

- Notebook: [Web Scrapping]()

In [3]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Next, request the HTML page from the above URL and get a `response` object

### TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

In [4]:
```
# use requests.get() method with the provided static_url
response=requests.get(static_url)
# assign the response to a object
content = response.text
```

Create a `BeautifulSoup` object from the HTML `response`

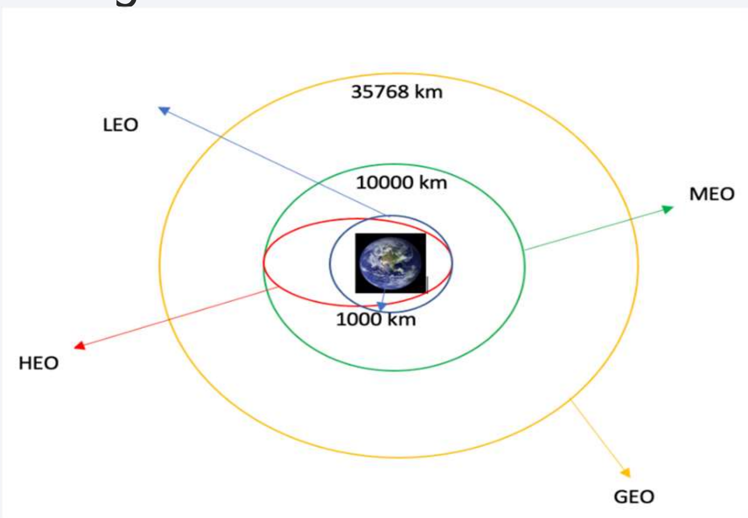
In [5]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup=BeautifulSoup(content,'html.parser')
```

In [7]:
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
html_tables=soup.find_all('table')
# Assign the result to a list called `html_tables`
html_tables
```

In [10]:
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
header_cells=first_launch_table.find_all('th')
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
for cell in header_cells:
    name=extract_column_from_header(cell)
    if name is not None and len(name) > 0:
        column_names.append(name)

# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
```

# Data Wrangling

- In this part , we have analyzed the data we have got , We have got a picture about the characteristics of different features , we have found out some important statistics like : LaunchSite wise Success rate , Orbit wise Success rate , landing outcomes etc.




Notebook: [Data Wrangling]

# EDA with Data Visualization

- We saw relationships among different parameters , like , relationship between Flight number and LaunchSite , Payload Mass and Launch Site,success rate of each orbit type, FlightNumber and Orbit type, Payload Mass and Orbit type, launch success yearly trend etc.

- Notebook: [Data Visualization EDA]

# EDA with SQL

- We applied SQL to perform EDA to get critical insights from our data Such as:
- Name of Unique Launchsites
- Total payload carried by Boosters
- Total number of Successful and failed missions

And so on…

Notebook: [EDA with SQL]

# Build an Interactive Map with Folium

We marked:

- all launch sites
- success and failures for each launch sites
- Distance between the launch sites and coastline / Civilization

Notebook: [Folium]

# Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- Pie charts for showing total launches from each Site
- Scatterplot between Outcome and Payload Mass

- Notebook: Plotly Dash Lab

# Predictive Analysis (Classification)

- We divided the dataset into train and test
- We fit the training data , and implemented GridSearchCV
- We did Hyperparameter tuning to figure out the best fit model for the data

- Notebook: ML Model , Cross Validation , Hyper Parameter tuning and finding best model

# Results

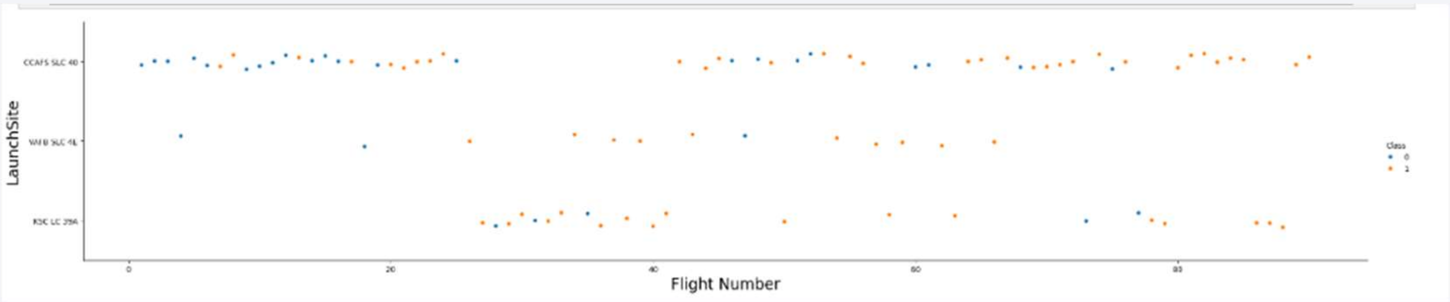
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

# Insights drawn from EDA

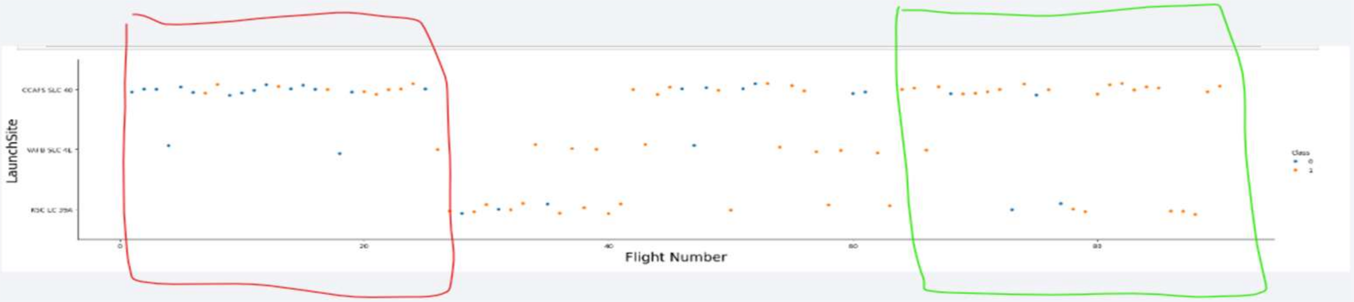
# Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site



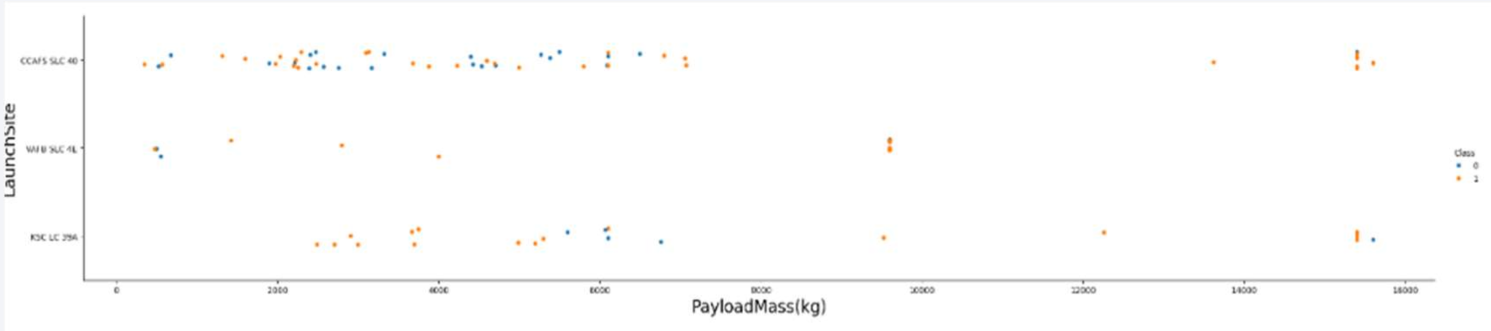

- Show the screenshot of the scatter plot with explanations

We can see larger the no. of launches from a Site , greater the probability of it being a success (Red marked area less success , Green area more success)



# Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site



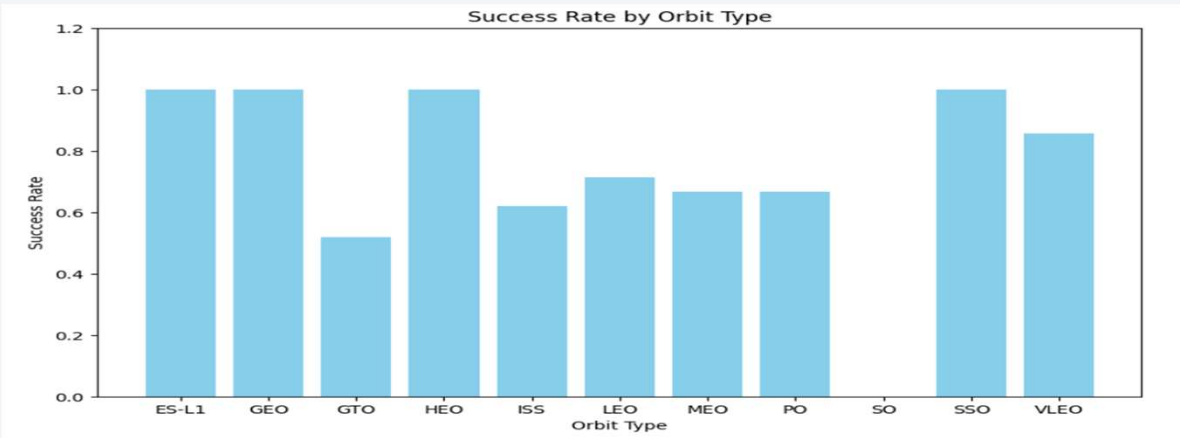

- Show the screenshot of the scatter plot with explanations

We can see: We always try to keep the Payload Mass as low as possible

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

# Success Rate vs. Orbit Type

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

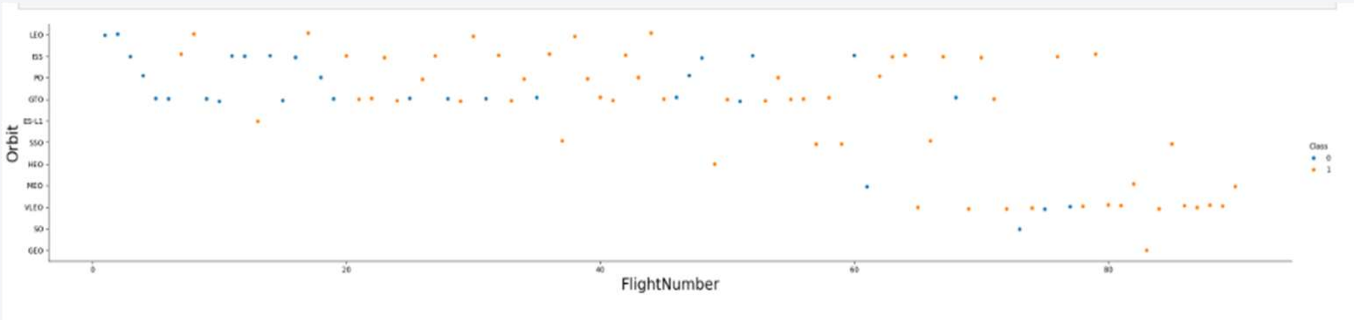



- Show the screenshot of the scatter plot with explanations

ES-L1,GEO,HEO,SSO had the most success rates.

# Flight Number vs. Orbit Type

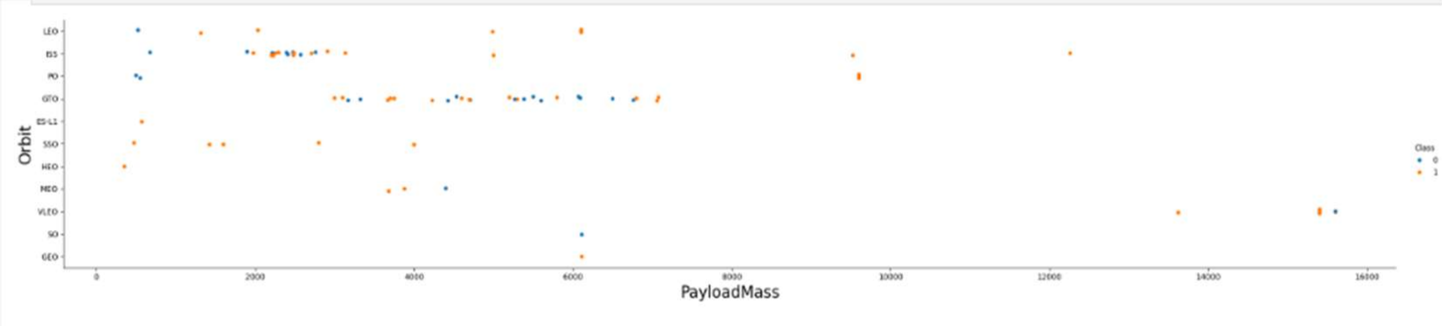
- Show a scatter point of Flight number vs. Orbit type




- Show the screenshot of the scatter plot with explanation
- If Flight < 20 , irrespective of any Orbit , failure dominates success
- We can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success

# Payload vs. Orbit Type

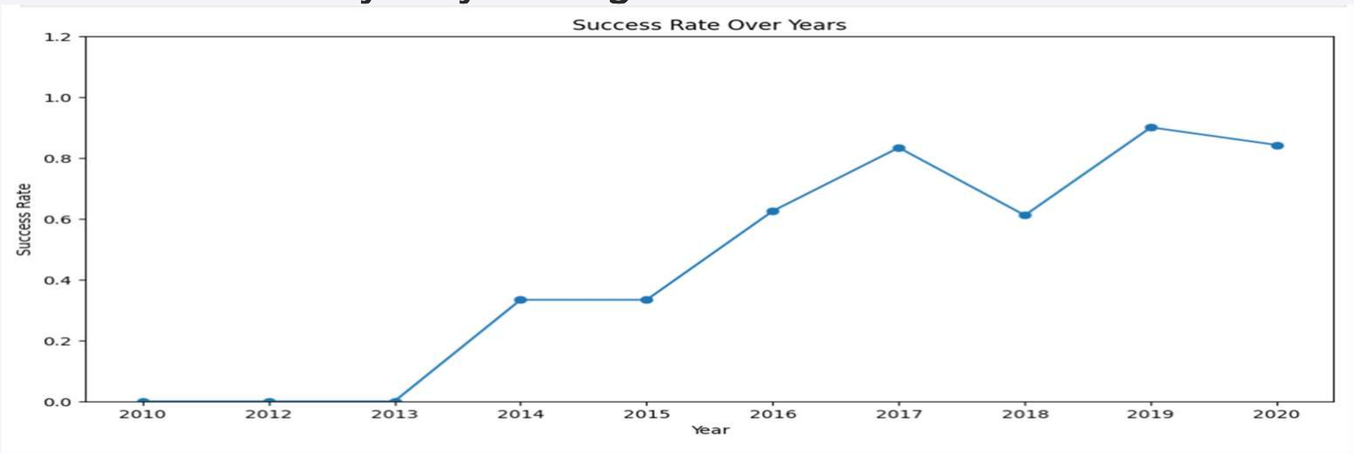
- Show a scatter point of payload vs. orbit type




- Show the screenshot of the scatter plot with explanations
- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.
- for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.

# Launch Success Yearly Trend

- Show a line chart of yearly average success rate




- Show the screenshot of the scatter plot with explanations

After 2013 , there is a sudden spike in Success rate

# All Launch Site Names

- Find the names of the unique launch sites

```
In [10]: %sql select distinct Launch_Site from SPACEXTBL

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

Out[10]:

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

- Present your query result with a short explanation here
- Used DISTINCT Keyword to find the names

# Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

In [11]:
```
%sql select *  from SPACEXTBL where Launch_Site like 'CCA%' LIMIT 5
```

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

Out[11]:

| 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 |

# Total Payload Mass

- Calculate the total payload carried by boosters from NASA

```
In [12]:  %sql select SUM(PAYLOAD_MASS__KG_) as total_payload_mass from SPACEXTBL where customer = 'NASA (CRS)'

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

Out[12]:

| total_payload_mass |
|---|
| 45596 |

- Present your query result with a short explanation here
- Used SUM aggregator function an NASA filter
- Result=45596

# Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

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

```
In [13]: %sql select AVG(PAYLOAD_MASS__KG_) as avergae_payload_mass from SPACEXTBL where Booster_Version like 'F9 v1.1%'

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

Out[13]:

| avergae_payload_mass |
|---|
| 2534.6666666666665 |

- Present your query result with a short explanation here
- Used AVG aggregator and F9 v1.1 Filter
- Result=2534.67

# First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```
In [14]: %sql select min(date) as first_successful_landing from SPACEXTBL where Landing_Outcome = 'Success (ground pad)';

 * sqlite:///my_data1.db
Done.

Out[14]:
```

| first_successful_landing |
|---|
| 2015-12-22 |

- Present your query result with a short explanation here
- Used min function an Success filter
- Result= 2015-12-22

# Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

```
In [15]: ect Booster_Version from SPACEXTBL where landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000

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

Out[15]:

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

- Present your query result with a short explanation here
- All resulting Booster Versions are type F9 FT B-1XXXX

# Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
In [16]: %sql select mission_Outcome,count(*) as total from SPACEXTBL group by mission_Outcome

 * sqlite:///my_data1.db
Done.
Out[16]:
```

| Mission_Outcome | total |
|---|---|
| Failure (in flight) | 1 |
| Success | 98 |
| Success | 1 |
| Success (payload status unclear) | 1 |

- Present your query result with a short explanation here
- Total Success =100
- Total Failure=1

# Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

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

In [17]:
```
%sql select booster_version from SPACEXTBL where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXTBL);
```

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

Out[17]:

| 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 |

- Present your query result with a short explanation here:
- All are type: ‘F9 B5 B10XX.X’

# 2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [25]:
```
%sql select substr(Date,6,2) as month, date, booster_version, launch_site, Landing_Outcome from SPACEXTBL where Landing_Outc
```

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

Out[25]:

| month | Date | Booster_Version | Launch_Site | Landing_Outcome |
|---|---|---|---|---|
| 01 | 2015-01-10 | F9 v1.1 B1012 | CCAFS LC-40 | Failure (drone ship) |
| 04 | 2015-04-14 | F9 v1.1 B1015 | CCAFS LC-40 | Failure (drone ship) |

- Present your query result with a short explanation here
- Both were from Same Launch Site: CCAFS LC-40

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

- 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

```
In [27]: %%sql select Landing_Outcome, count(*) as count_outcomes from SPACEXTBL
               where date between '2010-06-04' and '2017-03-20'
               group by Landing_Outcome
               order by count_outcomes desc;

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

Out[27]:

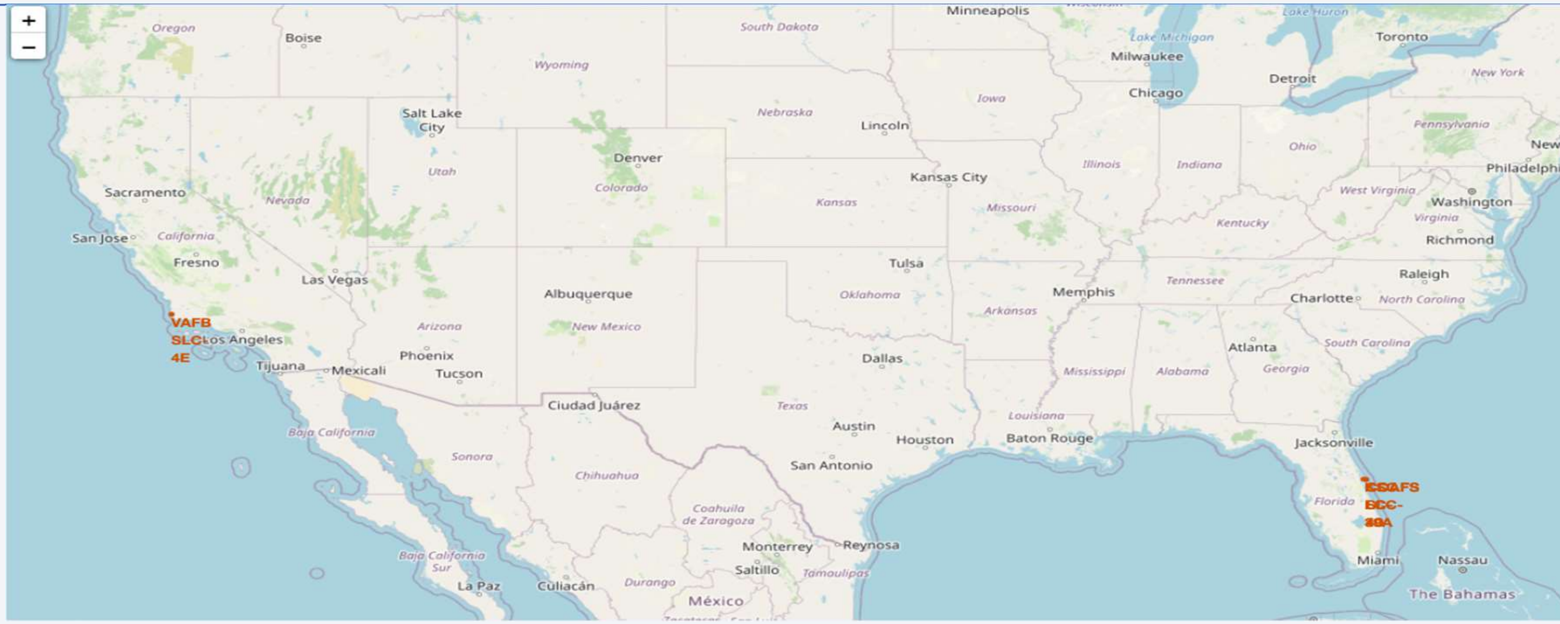
| Landing_Outcome | count_outcomes |
|---|---|
| No attempt | 10 |
| Success (drone ship) | 5 |
| Failure (drone ship) | 5 |
| Success (ground pad) | 3 |
| Controlled (ocean) | 3 |
| Uncontrolled (ocean) | 2 |
| Failure (parachute) | 2 |
| Precluded (drone ship) | 1 |

- Present your query result with a short explanation here
- No attempt with highest count 10 , Precluded (drone ship) with lowest count 1

Section 3

# Launch Sites Proximities Analysis

# Launch Sites

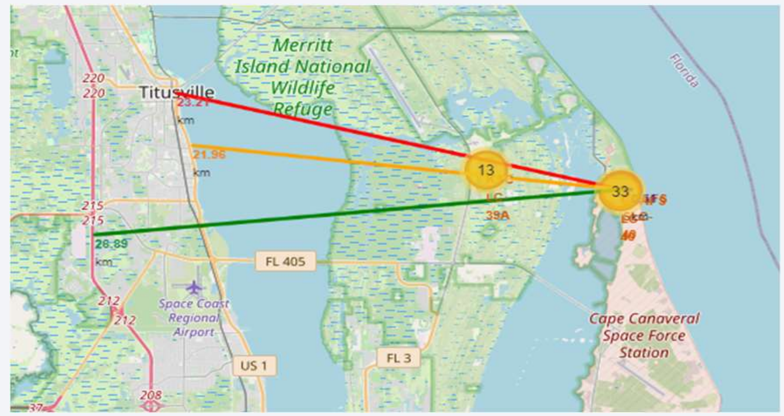



- Explain the important elements and findings on the screenshot
- Launch Sites are y the Coast of LA and Florida

# Successes and Failures


- Explain the important elements and findings on the screenshot
- We can see success/Failures for individual launch sites
- Green Marked are Successes and Reds are Failures

# Distances between a launch site to its proximities

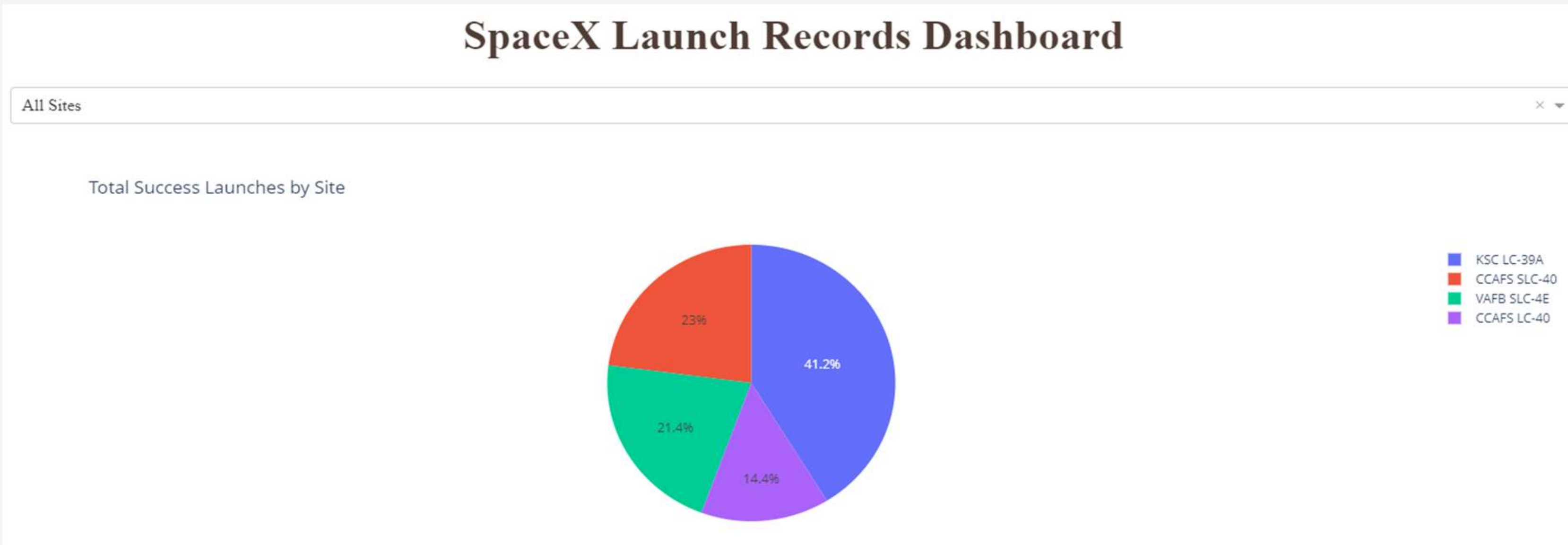




- Explain the important elements and findings on the screenshot
- We can see the distance between Launchsite and Coast is very less (Picture 1: 0.51km)
- Whereas, Distances of any Civilization(Township/Railways/Street) is far more (Pic2: 20Km+)

Section 4

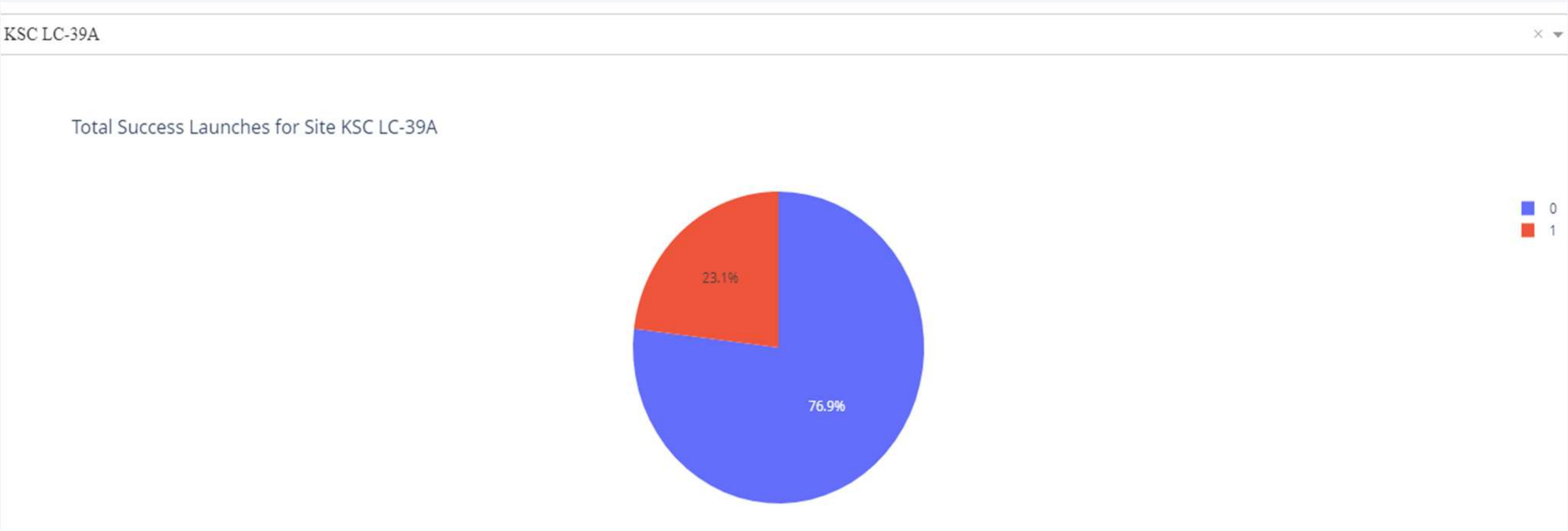
# Build a Dashboard with Plotly Dash

# % of success by each Site




- Explain the important elements and findings on the screenshot:
- KSC LC-39A is having highest % of contribution in overall success (41.2%)
- CCAFS LC-40 is having lowest % of contribution in overall success (14.4%)

# Launchsite having highest Success




- Explain the important elements and findings on the screenshot
- KSC LC-39A has the highest Success (76.9%)

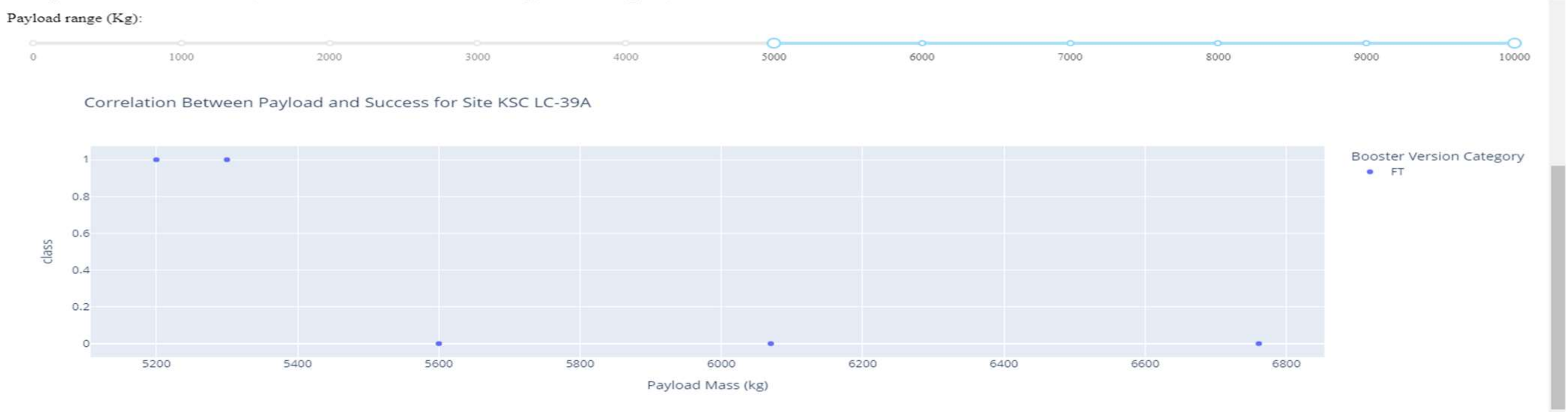
# Payload vs Launch Scatterplot1

Payload Mass(0-5000Kg Range)


- For 0-5K KG Payload Mass range FT Booster has significantly higher Success (class 1) compared to other Boosters

# Payload vs Launch Scatterplot2

Payload Mass(5000-10000Kg Range)




- For 5-10K KG Payload Mass range FT is the only working Booster
- Also we an see above a threshold of mass (5.6K KG) nothing works as expected

Section 5

# Predictive Analysis (Classification)

{

# Classification Accuracy

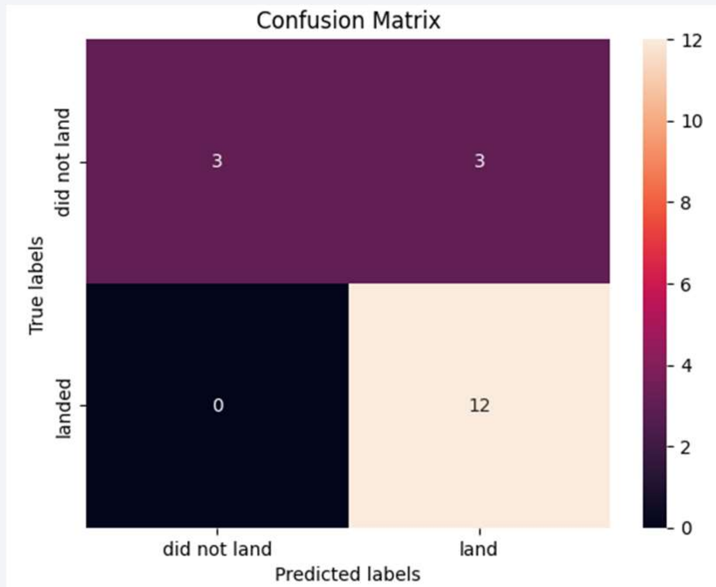
Find the method performs best:

```
[42]: models = {
    'KNNeighbors': knn_cv.best_score_,
    'DecisionTree': tree_cv.best_score_,
    'LogisticRegression': logreg_cv.best_score_,
    'SupportVector': svm_cv.best_score_
}

# Best Model
best_algorithm = max(models, key=models.get)
best_score = models[best_algorithm]

# Best Parameters
best_params = {
    'KNNeighbors': knn_cv.best_params_,
    'DecisionTree': tree_cv.best_params_,
    'LogisticRegression': logreg_cv.best_params_,
    'SupportVector': svm_cv.best_params_
}

print(f'Best model is {best_algorithm} with a score of {best_score}')
print(f'Best params are: {best_params[best_algorithm]}')

Best model is DecisionTree with a score of 0.8857142857142858
Best params are: {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'random'}
```

- Best Model is a Decision Tree 'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'random’}
- Accuracy is: 88.57%

# Confusion Matrix

- True Positive (TP)=12
- False Negative(FN)=0
- True Negative(TN)=3
- False Positive(FP)=3



# Conclusions

- Launchsites having more success rates are preferred over others for a new launch
- Launchsites are closer to the coast and further from Civilization
- Post 2013 there is a boom in Launch successes
- ES-L1,GEO,HEO,SSO,VLEO these orbits have the most successes
- KSC LC-39A has the most successful launches
- Decision Tree classifier is the Best ML Algorithm for the dataset
- We achieved an accuracy of 88.6%

# Thank you!