# SpaceX: Rocket Launch Analysis

Kevin Song

September 20, 2023

IBM Developer

SKILLS NETWORK

# Contents

IBM Developer

SKILLS NETWORK

# EXECUTIVE SUMMARY

## Methodologies

- Data Collection through API and Web Scraping techniques
- Data Wrangling
- Explore data with data visualization techniques
- Analyze data with SQL,
- Interactive Visual Analytics with Folium
- Plotly Dash Dashboard
- Machine Learning to predict landing outcomes

## Findings

- Launch success rate improved over the years
- KSC LC-39A has the highest success rate among all landing sites
- The success rate for Orbits ES-L1, GEO, HEO and SSO are100%
- The decision tree model is the best on test datasets, outperformed other models

# INTRODUCTION

## Background

SpaceX, a leading company in space industry, by offering a rocket launches (Falcon 9) as low as 62 million dollars, while other launch providers cost upward of 165 million dollar each. Much of the savings is because SpaceX can reuse the first stage. As a data scientist of a startup rivaling SpaceX, is to analyze data with machine learning techniques to predict the landing outcome of the first stage in the future. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This project is crucial in identifying the right price to compete with SpaceX for a rocket launch.

## Explore

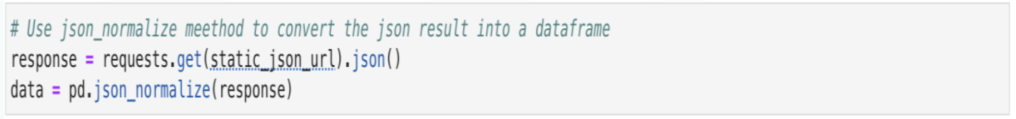
- How payload mass, launch site, number of flights, and orbits affect first-stage landing success
- Rate of successful landings over time
- Best predictive model for successful landing

# METHODOLOGY

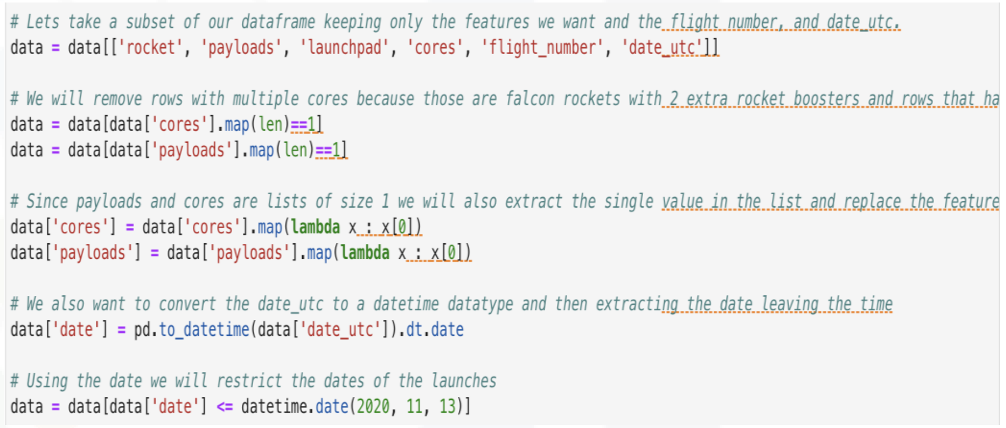
- Data collection methodology:
  - Data was collected using SpaceX REST API and web scrapping from Wikipedia
- Perform data wrangling
  - Data was processed using one-hot encoding for categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

## METHODOLOGY: Data Collection –API

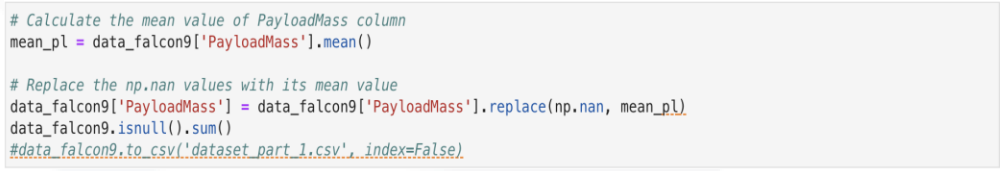
- Request data from SpaceX API (rocket launch data)
- Decode response using .json() and convert to a dataframe using .json_normalize()
- Request information about the launches from SpaceX API using custom functions
- Create dictionary from the data
- Create dataframe from the dictionary
- Filter dataframe to contain only Falcon 9 launches
- Replace missing values of Payload Mass with calculated .mean()
- Export data to csv file



```python
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```



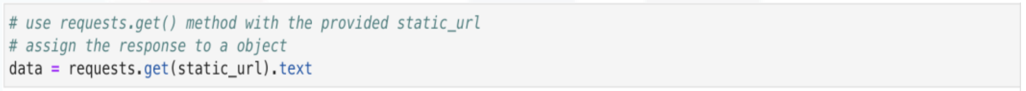
```python
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that ha
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```



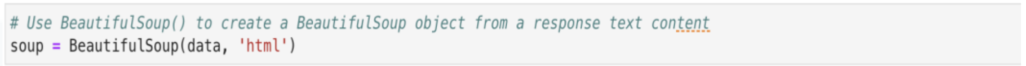
```python
# Calculate the mean value of PayloadMass column
mean_pl = data_falcon9['PayloadMass'].mean()

# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].replace(np.nan, mean_pl)
data_falcon9.isnull().sum()
#data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

## METHODOLOGY: Data Collection –Web Scraping

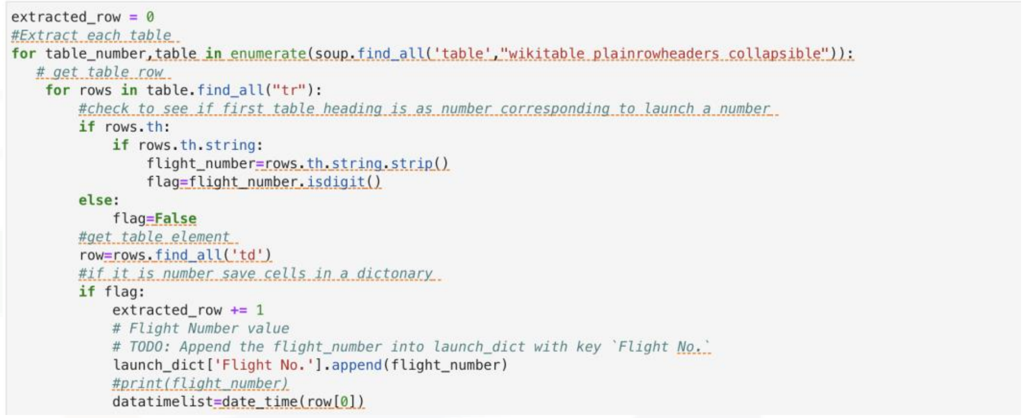
- Request data(Falcon 9 launch data) from Wikipedia
- Create BeautifulSoupobject
- Finding tables
- Extract column names
- Collect data from parsing HTML tables
- Create dictionary from the data
- Create dataframe from the dictionary
- Export data to csv file



```
# use requests.get() method with the provided static_url
# assign the response to a object
data = requests.get(static_url).text
```



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



```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            launch_dict['Flight No.'].append(flight_number)
            #print(flight_number)
            datatimelist=date_time(row[0])
```

IBM Developer

SKILLS NETWORK

# METHODOLOGY: Data Wrangling

Perform EDA and determine data labels

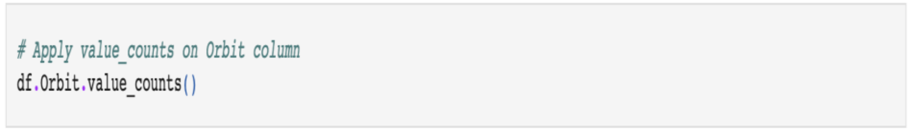
Calculate:

- # of launches for each site
- # and occurrence of orbit
- # and occurrence of mission outcome per orbit type

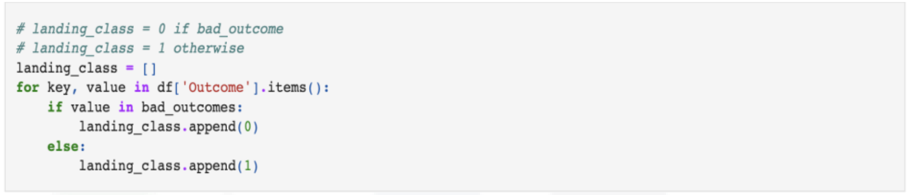
Create binary landing outcome column

Outcomes converted into 1 for a successful landing and 0 for an unsuccessful landing

Export to CSV file



```
# Apply value_counts on Orbit column
df.Orbit.value_counts()
```



```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for key, value in df['Outcome'].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

# METHODOLOGY: EDA with Visualization

Different type of charts

Flight Number vs. Payload

Flight Number vs. Launch Site

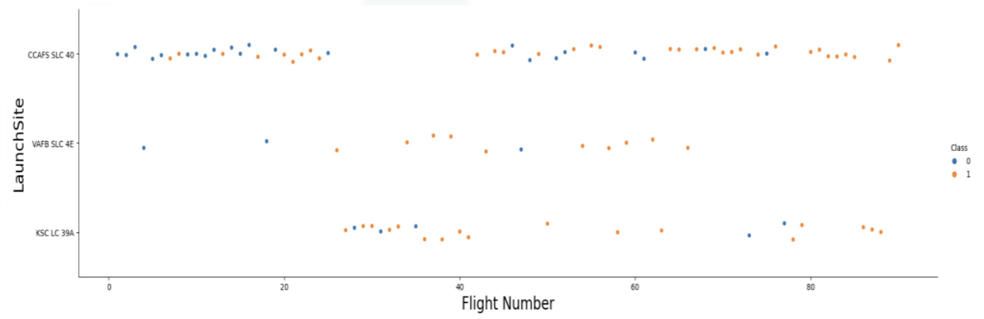
Payload Mass (kg) vs. Launch Site

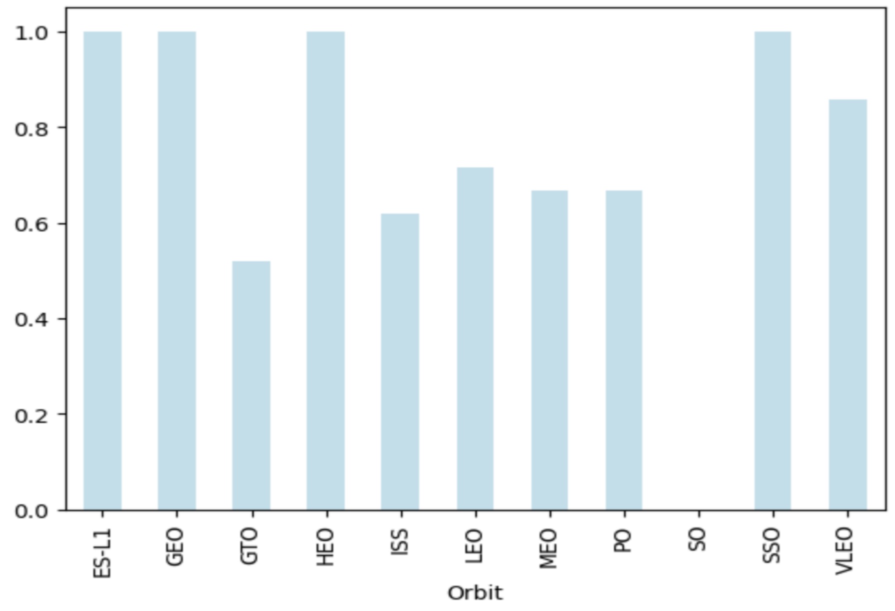
Payload Mass (kg) vs. Orbit type

Analysis

View relationship by using scatter plots. The variables could be useful for machine learning if a relationship exists

Show comparisons among discrete categories with bar charts. Bar charts show the relationships among the categories and measured value.



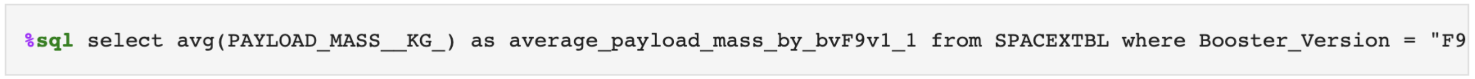


# METHODOLOGY: EDA with SQL

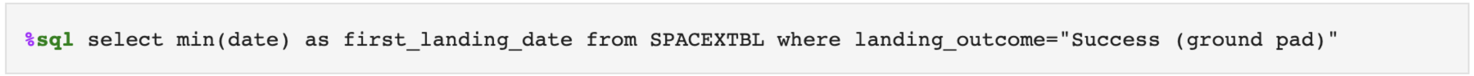
Using SQL queries to get better understanding of the dataset

**Display and List:**

- Names of unique launch sites
- 5 records where launch site begins with 'CCA'
- Total payload mass carried by boosters launched by NASA (CRS)
- Average payload mass carried by booster version F9 v1.1.
- Date of first successful landing on ground pad
- Names of boosters which had success landing on drone ship and have payload mass greater than 4,000 but less than 6,000
- Total number of successful and failed missions
- Names of booster versions which have carried the max payload
- Failed landing outcomes on drone ship, their booster version and launch site for the months in the year 2015
- Count of landing outcomes between 2010-06-04 and 2017-03-20 (desc)



```
%sql select avg(PAYLOAD_MASS__KG_) as average_payload_mass_by_bvF9v1_1 from SPACEXTBL where Booster_Version = "F9
```



```
%sql select min(date) as first_landing_date from SPACEXTBL where landing_outcome="Success (ground pad)"
```

# METHODOLOGY: Map with Folium

Markers Indicating Launch Sites

Added bluecircle at NASA Johnson Space Center's coordinate with a popup label showing its name using its latitude and longitude coordinates

Added red circles at all launch sites coordinates with a popup label showing its name using its name using its latitude and longitude coordinates

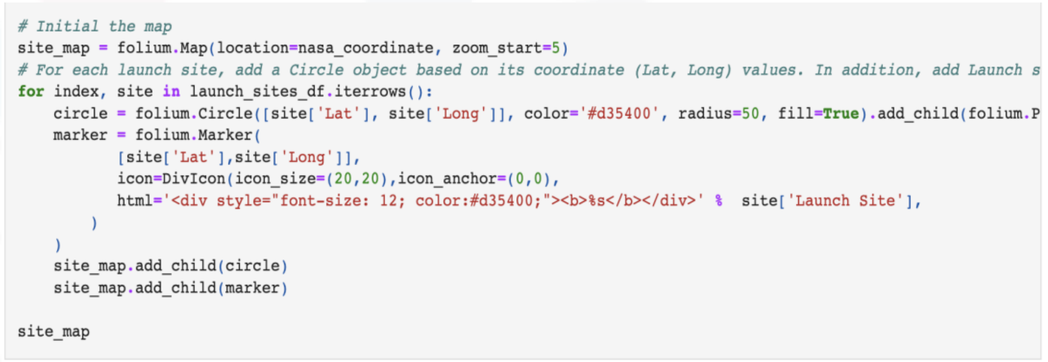
Map with Folium

Colored Markers of Launch Outcomes

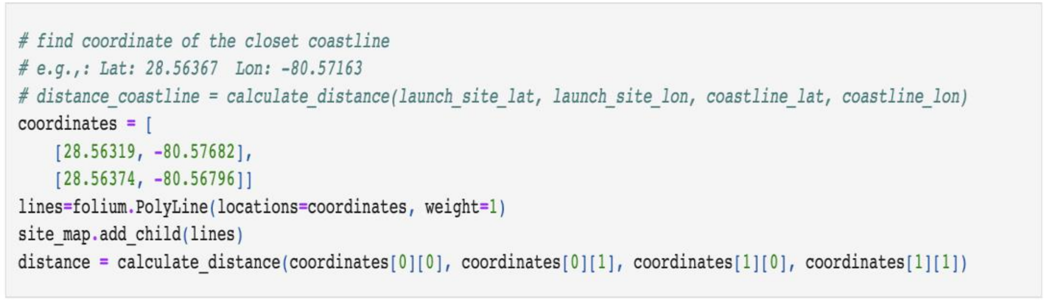
Added colored markers of successful(green) and unsuccessful(red) launches at each launch site to show which launch sites have high success rates

Distances Between a Launch Site to Proximities

Added colored lines to show distance between launch site CCAFS SLC-40 andits proximity to the nearest coastline, railway, highway, and city



```
# Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
# For each launch site, add a Circle object based on its coordinate (Lat, Long) values. In addition, add Launch s
for index, site in launch_sites_df.iterrows():
    circle = folium.Circle([site['Lat'], site['Long']], color='#d35400', radius=50, fill=True).add_child(folium.P
    marker = folium.Marker(
            [site['Lat'],site['Long']],
            icon=DivIcon(icon_size=(20,20),icon_anchor=(0,0),
            html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' %  site['Launch Site'],
        )
    )
    site_map.add_child(circle)
    site_map.add_child(marker)

site_map
```



```
# find coordinate of the closet coastline
# e.g.,: Lat: 28.56367  Lon: -80.57163
# distance_coastline = calculate_distance(launch_site_lat, launch_site_lon, coastline_lat, coastline_lon)
coordinates = [
    [28.56319, -80.57682],
    [28.56374, -80.56796]]
lines=folium.PolyLine(locations=coordinates, weight=1)
site_map.add_child(lines)
distance = calculate_distance(coordinates[0][0], coordinates[0][1], coordinates[1][0], coordinates[1][1])
```

## METHODOLOGY: Dashboard with Plotly Dash

Dropdown List with Launch Sites

- Allow user to select all launch sites or a certain launch site

Slider of Payload Mass Range

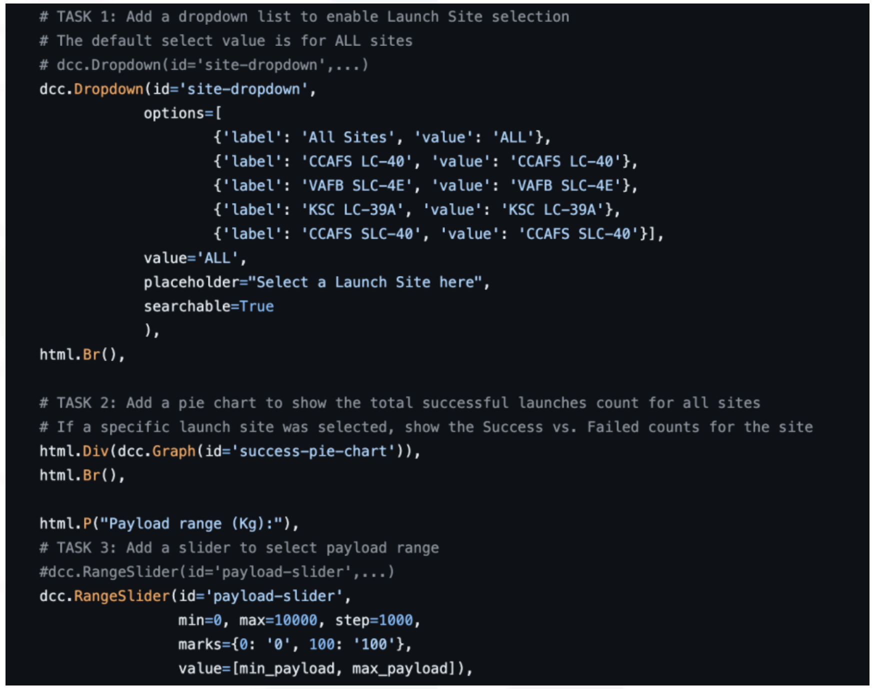
- Allow user to select payload mass range

Pie Chart Showing Successful Launches

- Allow user to see successful and unsuccessful launches as a percent of the total

Scatter Chart Showing Payload Mass vs. Success Rate by Booster Version

- Allow user to see the correlation between Payload and Launch Success



```
# TASK 1: Add a dropdown list to enable Launch Site selection
# The default select value is for ALL sites
# dcc.Dropdown(id='site-dropdown',...)
dcc.Dropdown(id='site-dropdown',
            options=[
                    {'label': 'All Sites', 'value': 'ALL'},
                    {'label': 'CCAFS LC-40', 'value': 'CCAFS LC-40'},
                    {'label': 'VAFB SLC-4E', 'value': 'VAFB SLC-4E'},
                    {'label': 'KSC LC-39A', 'value': 'KSC LC-39A'},
                    {'label': 'CCAFS SLC-40', 'value': 'CCAFS SLC-40'}],
            value='ALL',
            placeholder="Select a Launch Site here",
            searchable=True
            ),
html.Br(),

# TASK 2: Add a pie chart to show the total successful launches count for all sites
# If a specific launch site was selected, show the Success vs. Failed counts for the site
html.Div(dcc.Graph(id='success-pie-chart')),
html.Br(),

html.P("Payload range (Kg):"),
# TASK 3: Add a slider to select payload range
#dcc.RangeSlider(id='payload-slider',...)
dcc.RangeSlider(id='payload-slider',
                min=0, max=10000, step=1000,
                marks={0: '0', 100: '100'},
                value=[min_payload, max_payload]),
```

# METHODOLOGY: Predictive Analytics

## Building the Model

- •Load the dataset into NumPy and Pandas
- •Transform the data and then split into training and test datasets
- •Decide which type of ML to use
- •set the parameters and algorithms to GridSearchCV and fit it to dataset.

## Evaluating the Model

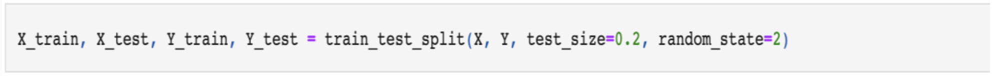
- •Check the accuracy for each model
- •Get tuned hyperparameters for each type of algorithms.
- •plot the confusion matrix.

## Improving the Model

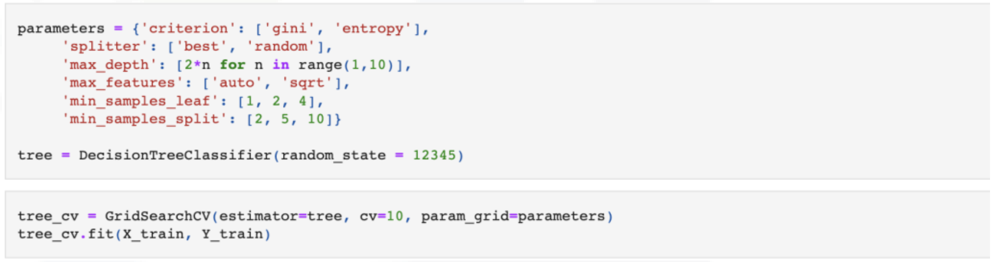
- •Use Feature Engineering and Algorithm Tuning

## Find the Best Model

- •The model with the best accuracy score will be the best performing model.



```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
```



```
parameters = {'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]}

tree = DecisionTreeClassifier(random_state = 12345)
```

```
tree_cv = GridSearchCV(estimator=tree, cv=10, param_grid=parameters)
tree_cv.fit(X_train, Y_train)
```

## RESULTS

IBM Developer

SKILLS NETWORK

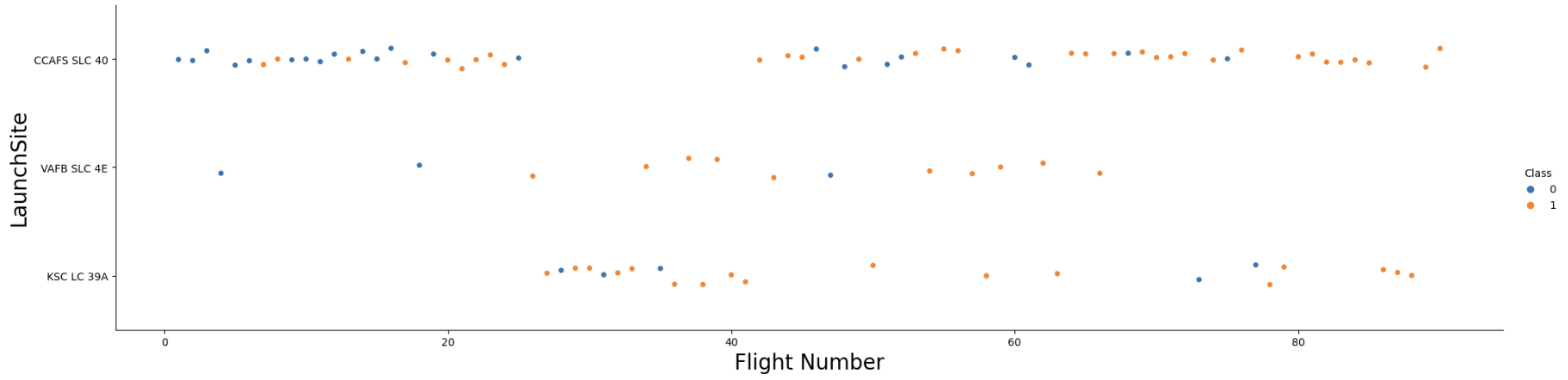
## RESULTS: Flight Number vs. Launch Site

Later flights had a higher success rate

CCAFS SLC 40 has more launches and VAFB SLC 4E has least launches

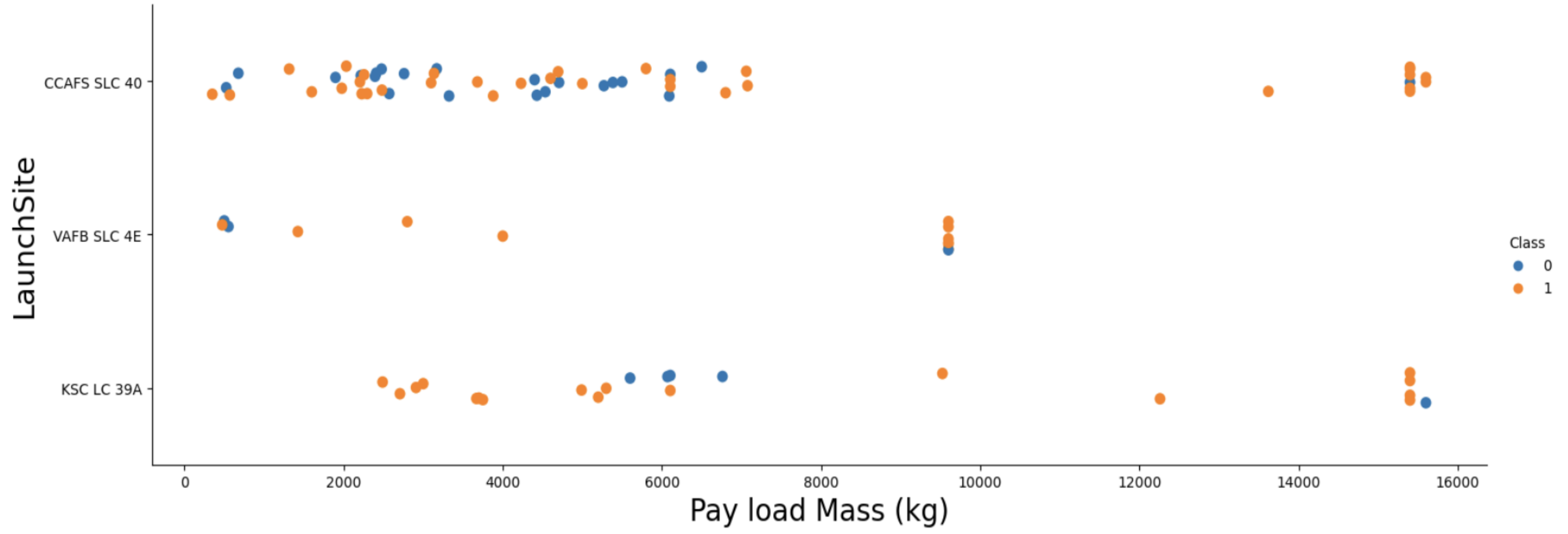
KSC LC 39A have higher success rates

Newer launches have a higher success rate



## RESULTS: Payload vs. Launch Site

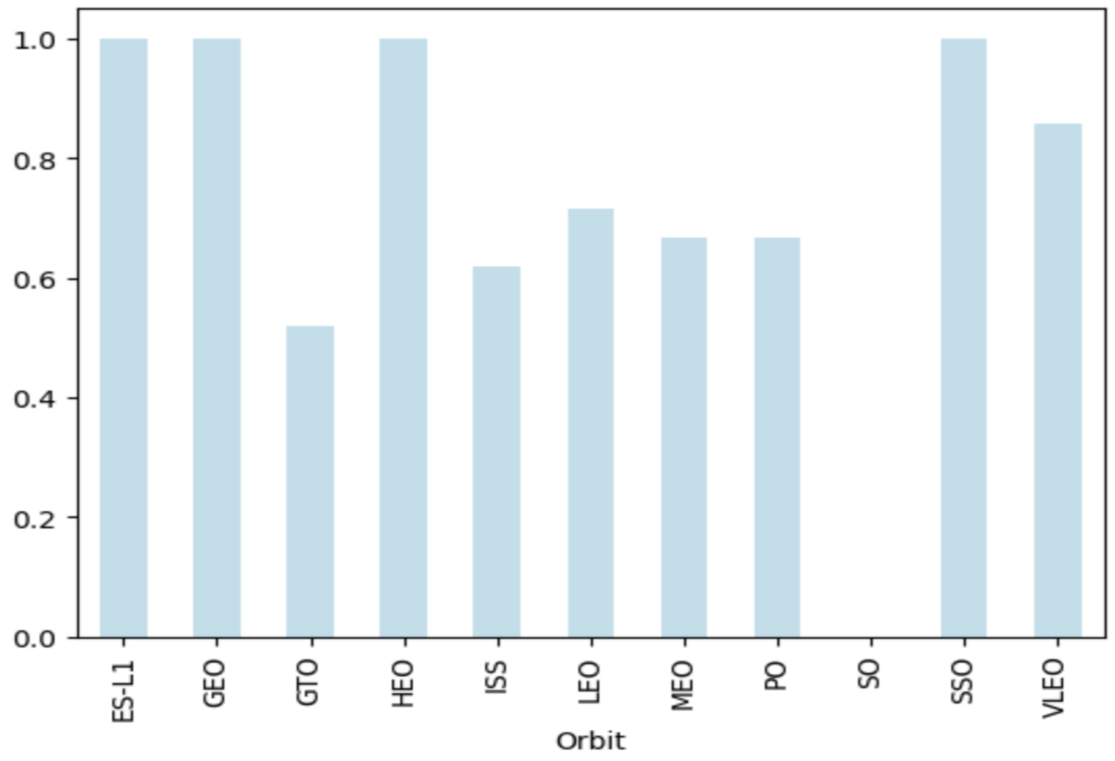
There is no clear pattern to show the launch site success is dependent on pay load mass



## RESULTS: Success Rate by Orbit

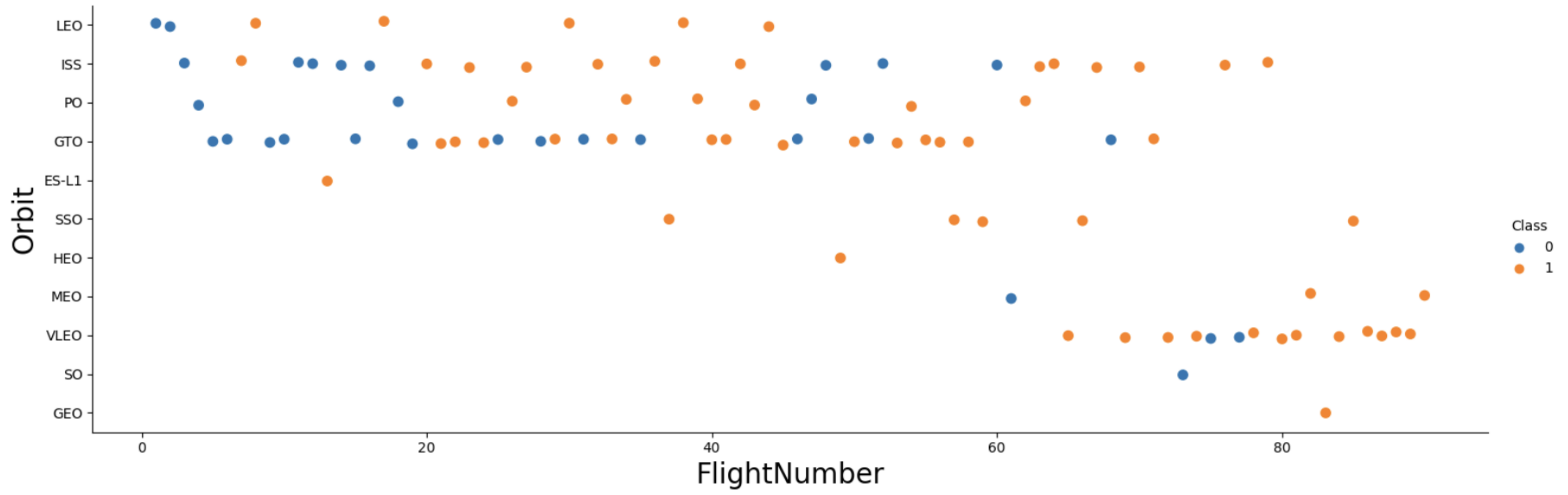
This Bar chart depicted the orbits to influences the landing outcomes.

Some orbits have 100% success rate such as SSO, HEO, GEO AND ES-L1 while SO has 0% rate of success.



## RESULTS: Flight Number vs. Orbit

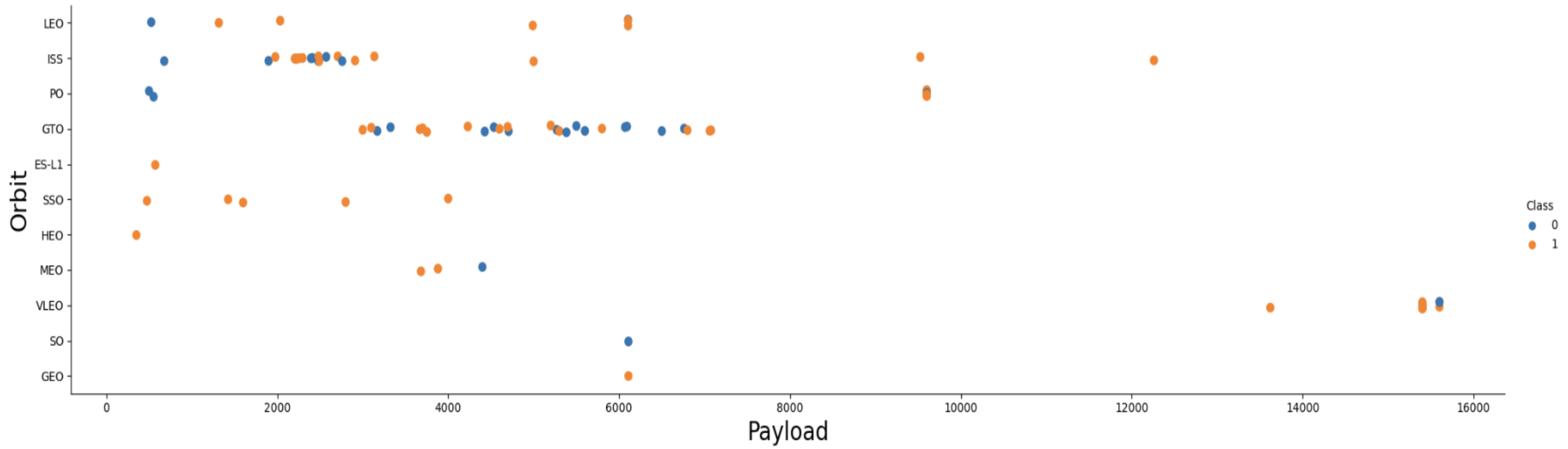
The scatter plot indicates that in general the higher the flight number on each orbits, the greater the success rate, except for GTO which depicts no relationship between these two attributes.



# RESULTS: Payload vs. Orbit

Heavy payloads with LEO, ISS and PO orbits have higher success rate

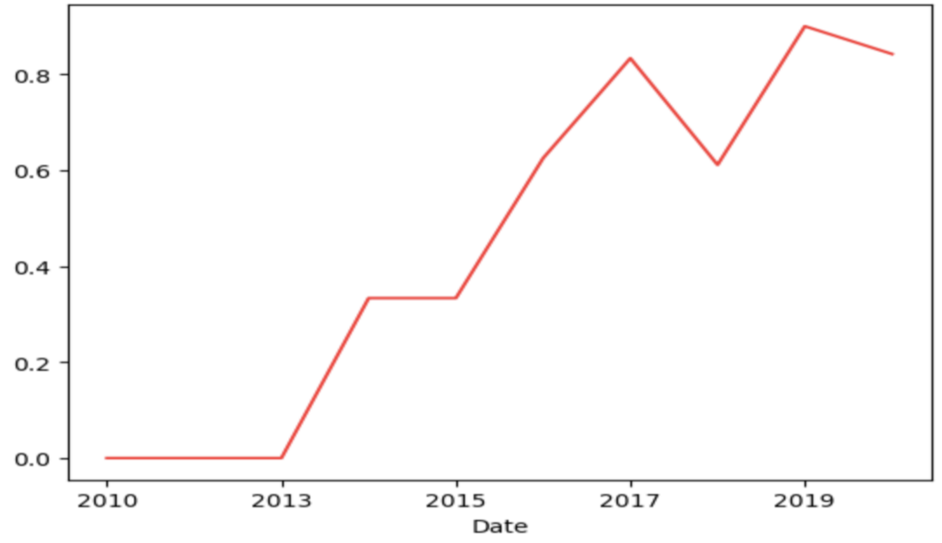
The GTO orbit has mixed success with heavier payloads



## RESULTS: Launch Success over Time

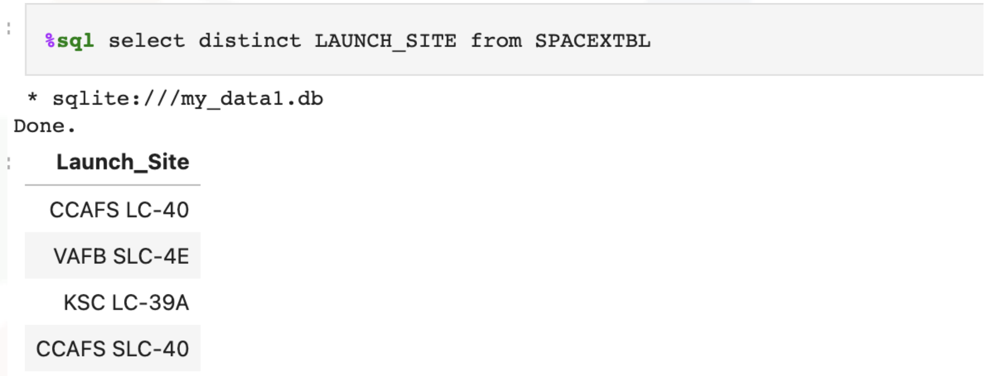
Overall, the success rate has improved since 2013

*This line chart depicted that* success rate improved from 2013 to 2017 and 2018 to 2019



# RESULTS: Launch Site Information

Use key word DISTINCT to show only unique launch site names from the SpaceX data.

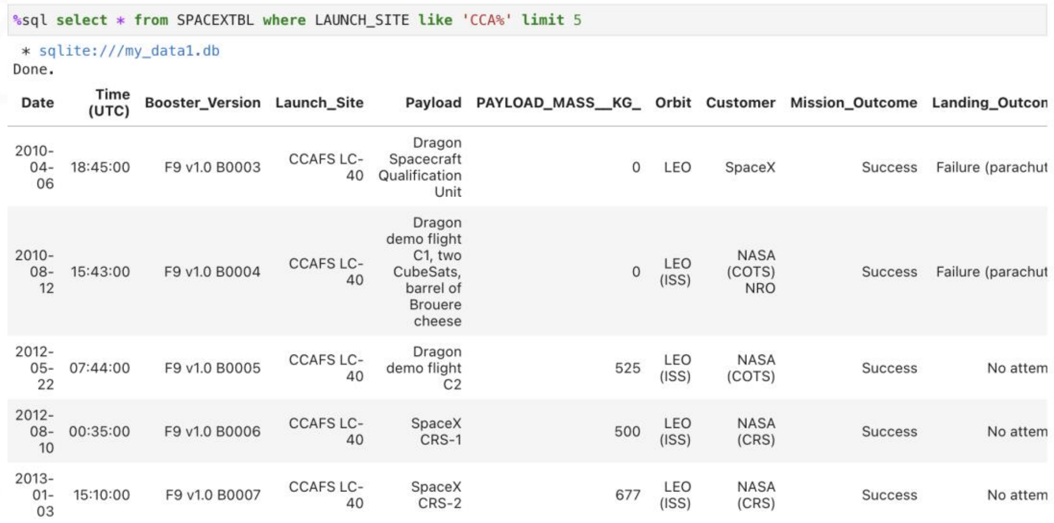


```
%sql select distinct LAUNCH_SITE from SPACEXTBL
```

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

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

Records with Launch Site begins with CCA, displaying 5 records



```
%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_Outcon |
|---|---|---|---|---|---|---|---|---|---|
| 2010-04-06 | 18:45:00 | F9 v1.0 B0003 | CCAFS LC-40 | Dragon Spacecraft Qualification Unit | 0 | LEO | SpaceX | Success | Failure (parachut |
| 2010-08-12 | 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 (parachut |
| 2012-05-22 | 07:44:00 | F9 v1.0 B0005 | CCAFS LC-40 | Dragon demo flight C2 | 525 | LEO (ISS) | NASA (COTS) | Success | No attem |
| 2012-08-10 | 00:35:00 | F9 v1.0 B0006 | CCAFS LC-40 | SpaceX CRS-1 | 500 | LEO (ISS) | NASA (CRS) | Success | No attem |
| 2013-01-03 | 15:10:00 | F9 v1.0 B0007 | CCAFS LC-40 | SpaceX CRS-2 | 677 | LEO (ISS) | NASA (CRS) | Success | No attem |

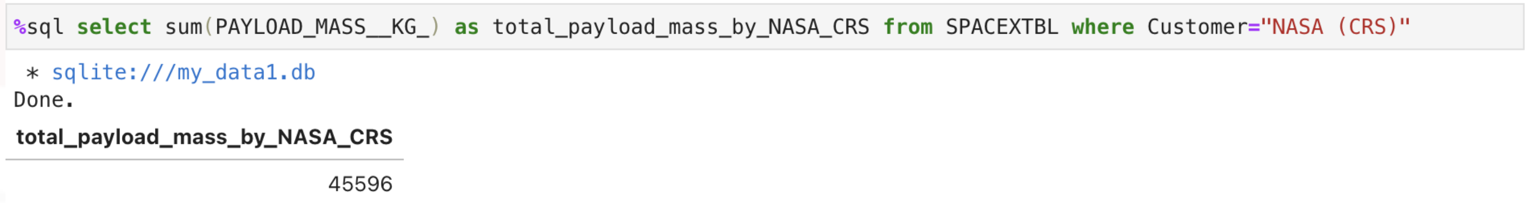
# RESULTS: Payload Mass

Total Payload Mass

45,596 kg carried by boosters launched by NASA (CRS)

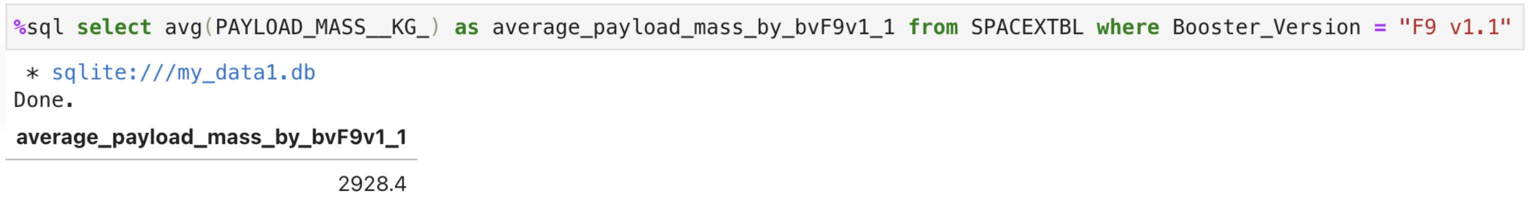
Average Payload Mass

2,928 kg (average) carried by booster version F9 v1.1



```
%sql select sum(PAYLOAD_MASS__KG_) as total_payload_mass_by_NASA_CRS from SPACEXTBL where Customer="NASA (CRS)"
 * sqlite:///my_data1.db
Done.
```

| total_payload_mass_by_NASA_CRS |
|---|
| 45596 |



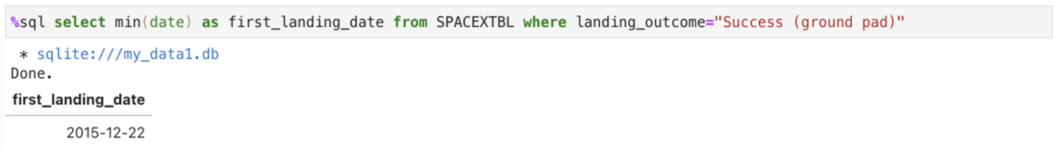
```
%sql select avg(PAYLOAD_MASS__KG_) as average_payload_mass_by_bvF9v1_1 from SPACEXTBL where Booster_Version = "F9 v1.1"
 * sqlite:///my_data1.db
Done.
```

| average_payload_mass_by_bvF9v1_1 |
|---|
| 2928.4 |

## RESULTS: Landing & Mission Info

First successful Landing on Ground Pad

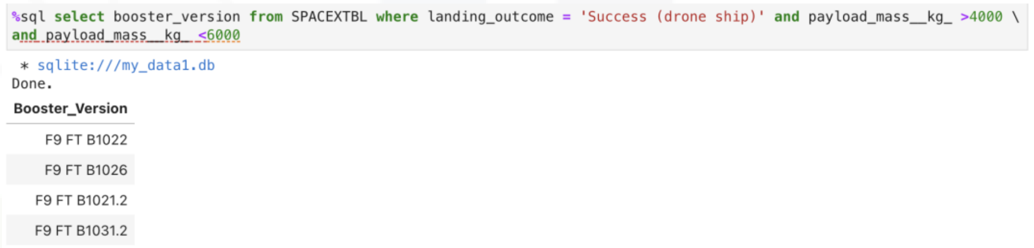
2015-12-22



```
%sql select min(date) as first_landing_date from SPACEXTBL where landing_outcome="Success (ground pad)"

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

| first_landing_date |
|---|
| 2015-12-22 |

Drone Ship Landing with Booster mass greater than 4,000 but less than 6,000



```
%sql select booster_version from SPACEXTBL where landing_outcome = 'Success (drone ship)' and payload_mass__kg_ >4000 \
and payload_mass__kg_ <6000

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

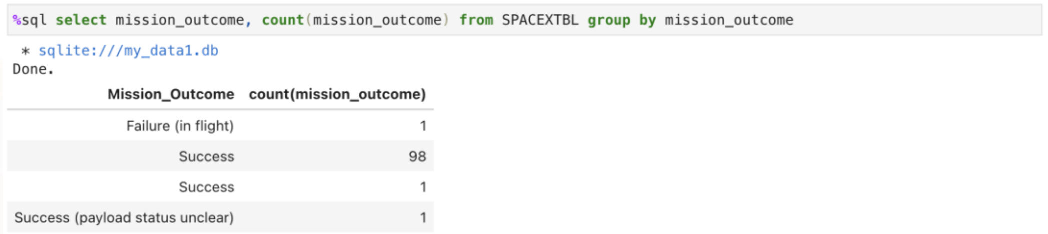
| Booster_Version |
|---|
| F9 FT B1022 |
| F9 FT B1026 |
| F9 FT B1021.2 |
| F9 FT B1031.2 |

Total Number of Successful and Failed Mission Outcomes

1 Failure in Flight

99 Success

1 Success (payload status unclear)

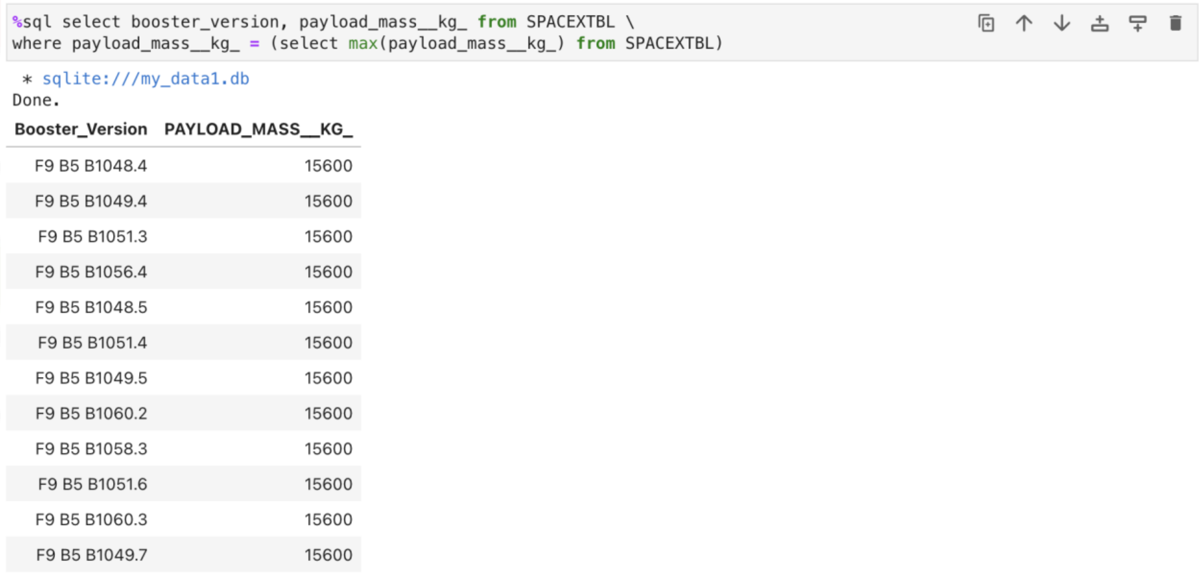


```
%sql select mission_outcome, count(mission_outcome) from SPACEXTBL group by mission_outcome

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

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

## RESULTS: Boosters

Boosters carrying Max Payload



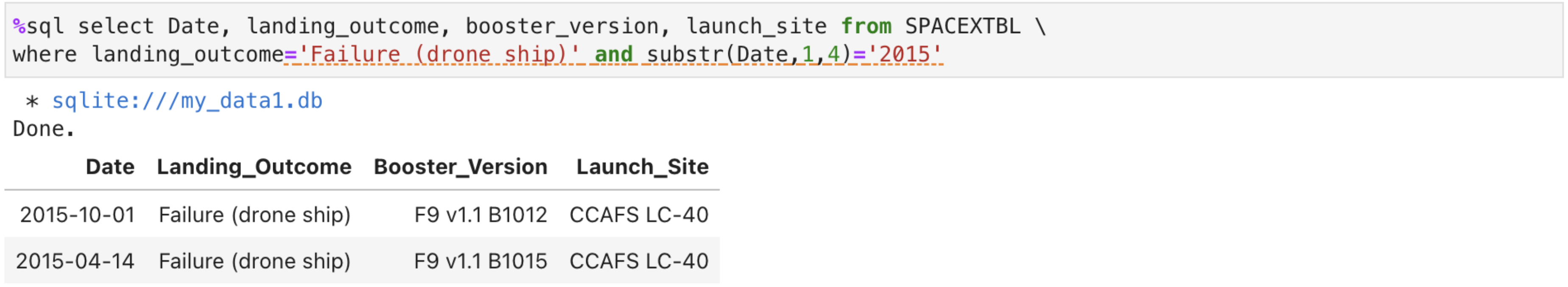
```
%sql select booster_version, payload_mass__kg_ from SPACEXTBL \
where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXTBL)
```

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

| Booster_Version | PAYLOAD_MASS__KG_ |
|---|---|
| F9 B5 B1048.4 | 15600 |
| F9 B5 B1049.4 | 15600 |
| F9 B5 B1051.3 | 15600 |
| F9 B5 B1056.4 | 15600 |
| F9 B5 B1048.5 | 15600 |
| F9 B5 B1051.4 | 15600 |
| F9 B5 B1049.5 | 15600 |
| F9 B5 B1060.2 | 15600 |
| F9 B5 B1058.3 | 15600 |
| F9 B5 B1051.6 | 15600 |
| F9 B5 B1060.3 | 15600 |
| F9 B5 B1049.7 | 15600 |

## RESULTS: Failed Landings on Drone Ship

Filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015



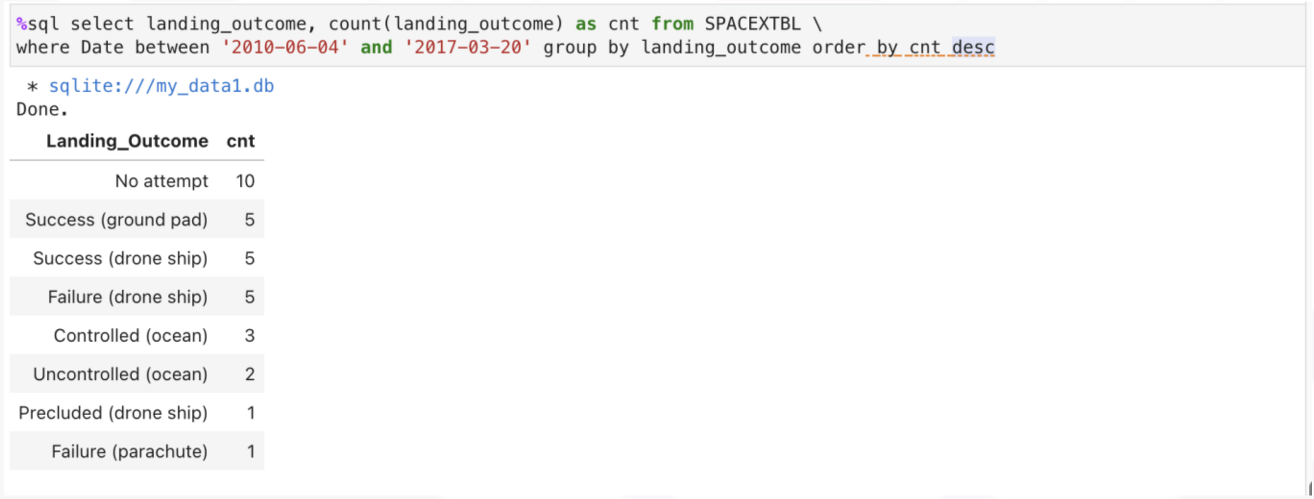
```
%sql select Date, landing_outcome, booster_version, launch_site from SPACEXTBL \
where landing_outcome='Failure (drone ship)' and substr(Date,1,4)='2015'
```

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

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

## RESULTS: Count of Successful Landings

List count of landing outcomes between 2010-06-04 and 2017-03-20 in descending order



```
%sql select landing_outcome, count(landing_outcome) as cnt from SPACEXTBL \
where Date between '2010-06-04' and '2017-03-20' group by landing_outcome order by cnt desc
```

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

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

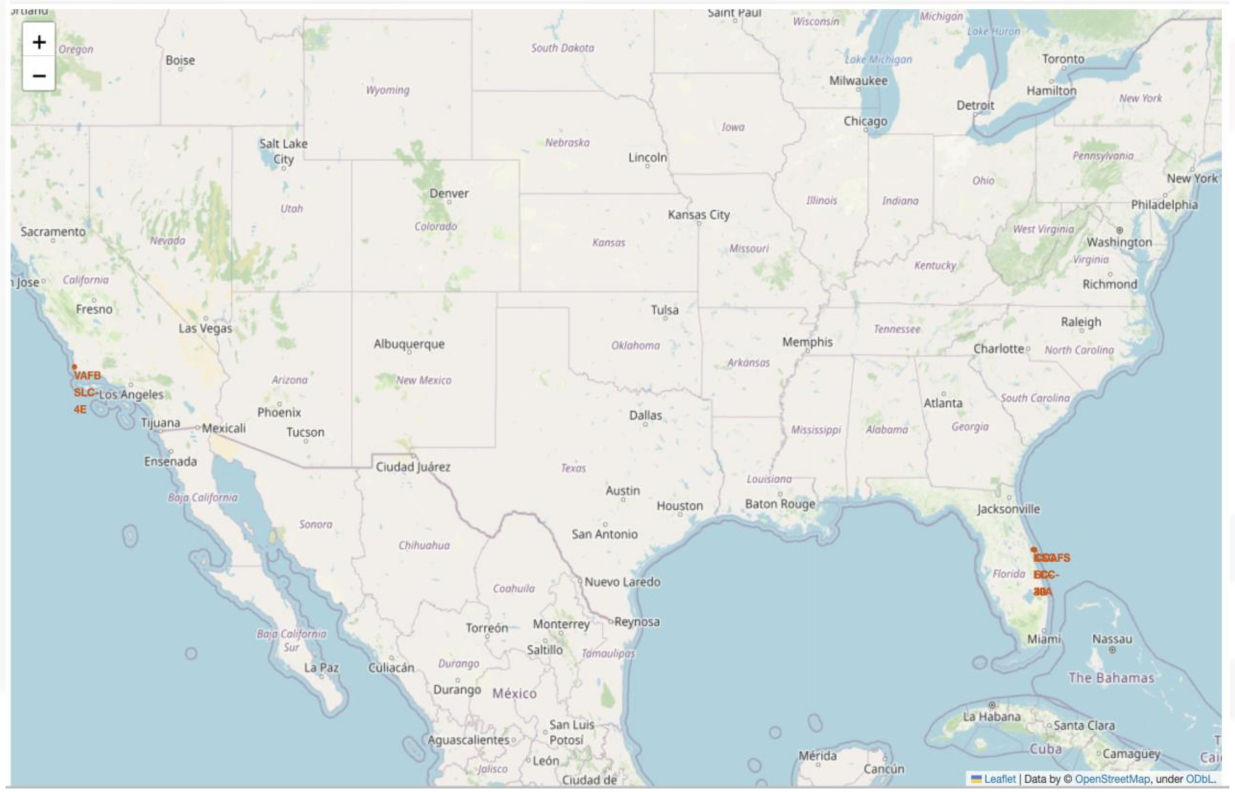
## RESULTS: Launch Site Analysis

IBM Developer

SKILLS NETWORK

## RESULTS: Launch Site Analysis - Sites

All the SpaceX launch sites are located in California and Florida costal line in US



IBM Developer

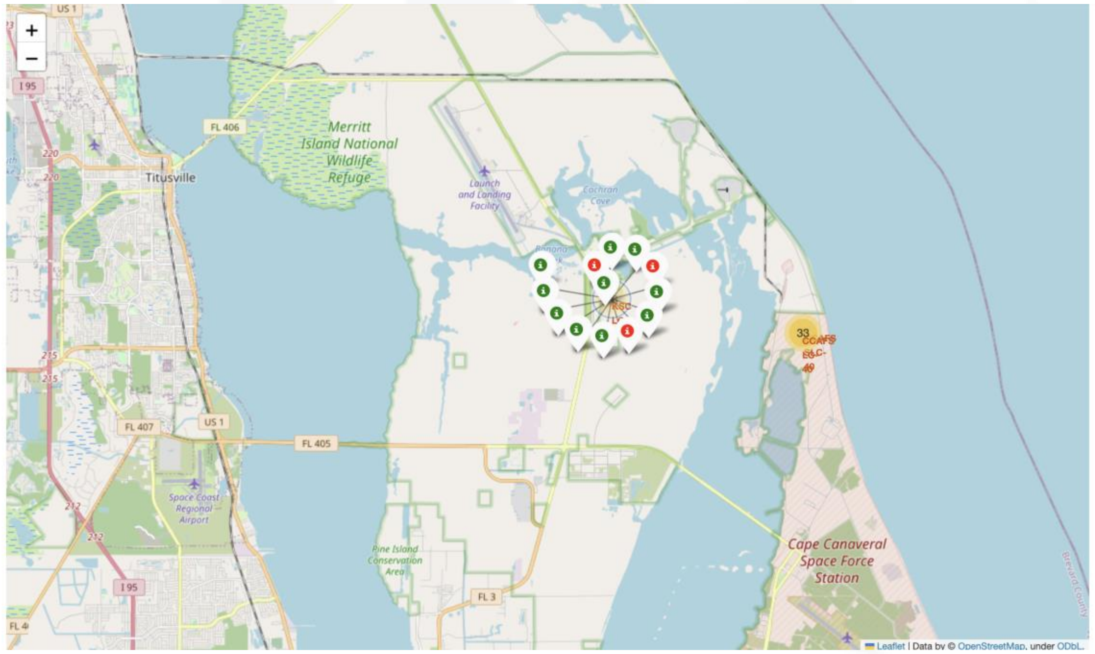
SKILLS NETWORK

## RESULTS: Launch Site Analysis - Color Labeled Launch Outcomes

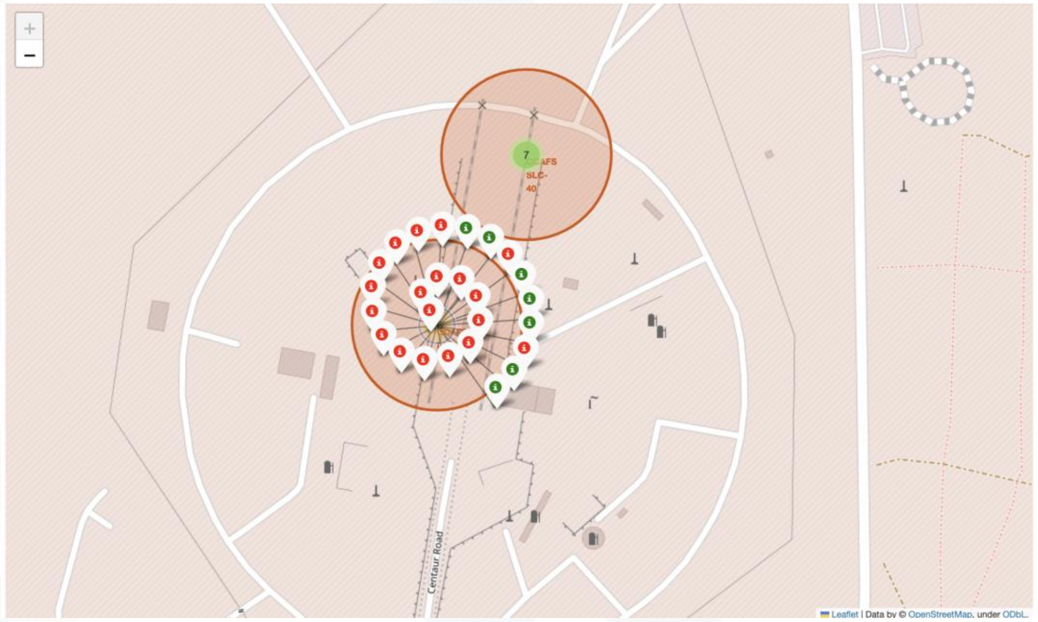
Outcomes at Each Launch Site:

Green markers for successful launches

Red markers for failed launches



Merritt Island National Wildlife Refuge
Titusville
Launch and Landing Facility
Cochran Cove
Cape Canaveral Space Force Station
Space Coast Regional Airport
Pine Island Conservation Area
33
CCAFS SLC-40
Leaflet | Data by © OpenStreetMap, under ODbL.



## RESULTS: Launch Site Analysis - Distance to Proximities

CCAFS SLC-40

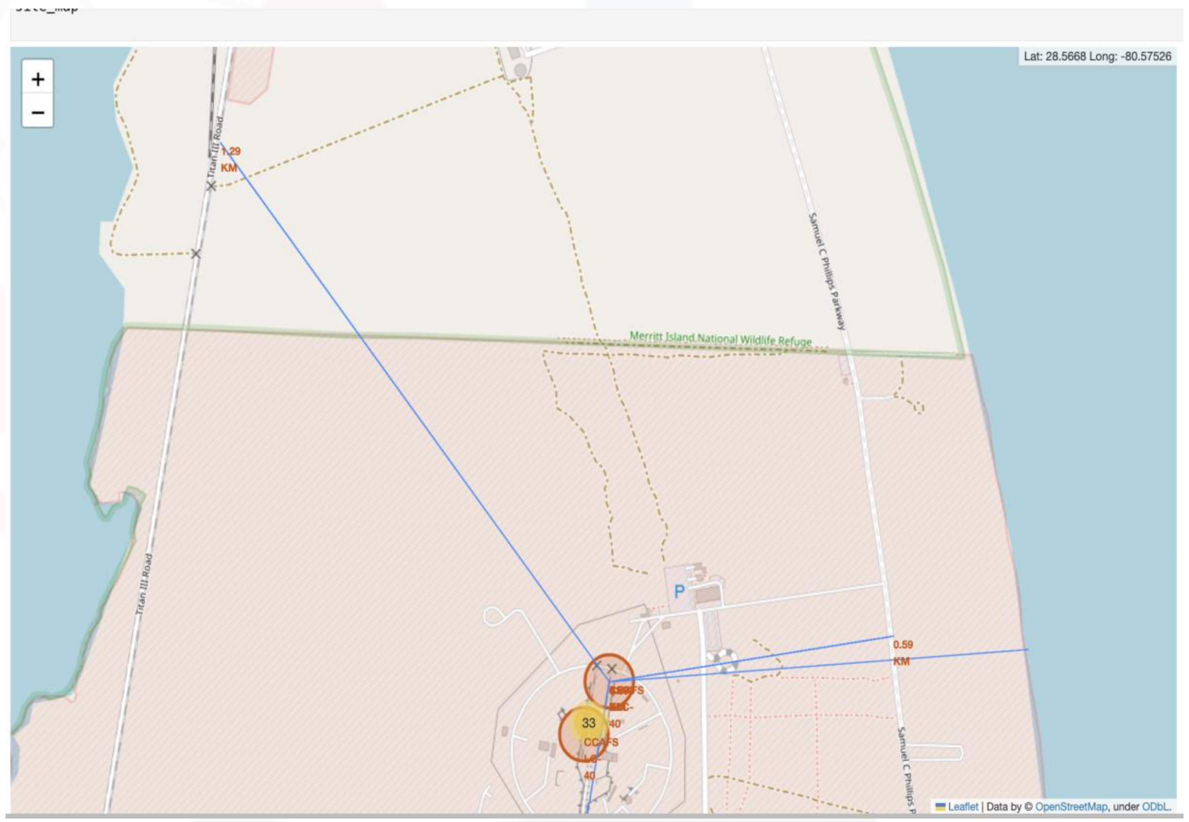
1.29 km from nearest railway

51.93 km from nearest city Melbourne

0.59 km from nearest highway

- Are launch sites in close proximity to railways? Yes,
- Are launch sites in close proximity to highways? Yes,
- Are launch sites in close proximity to coastline? Yes,
- Do launch sites keep certain distance away from cities? Yes,

Launch site need to be away from anything a failed launch can damage, but still close enough to highway and railroad to be able to transport people and equipment to or from it in support of launch activities.

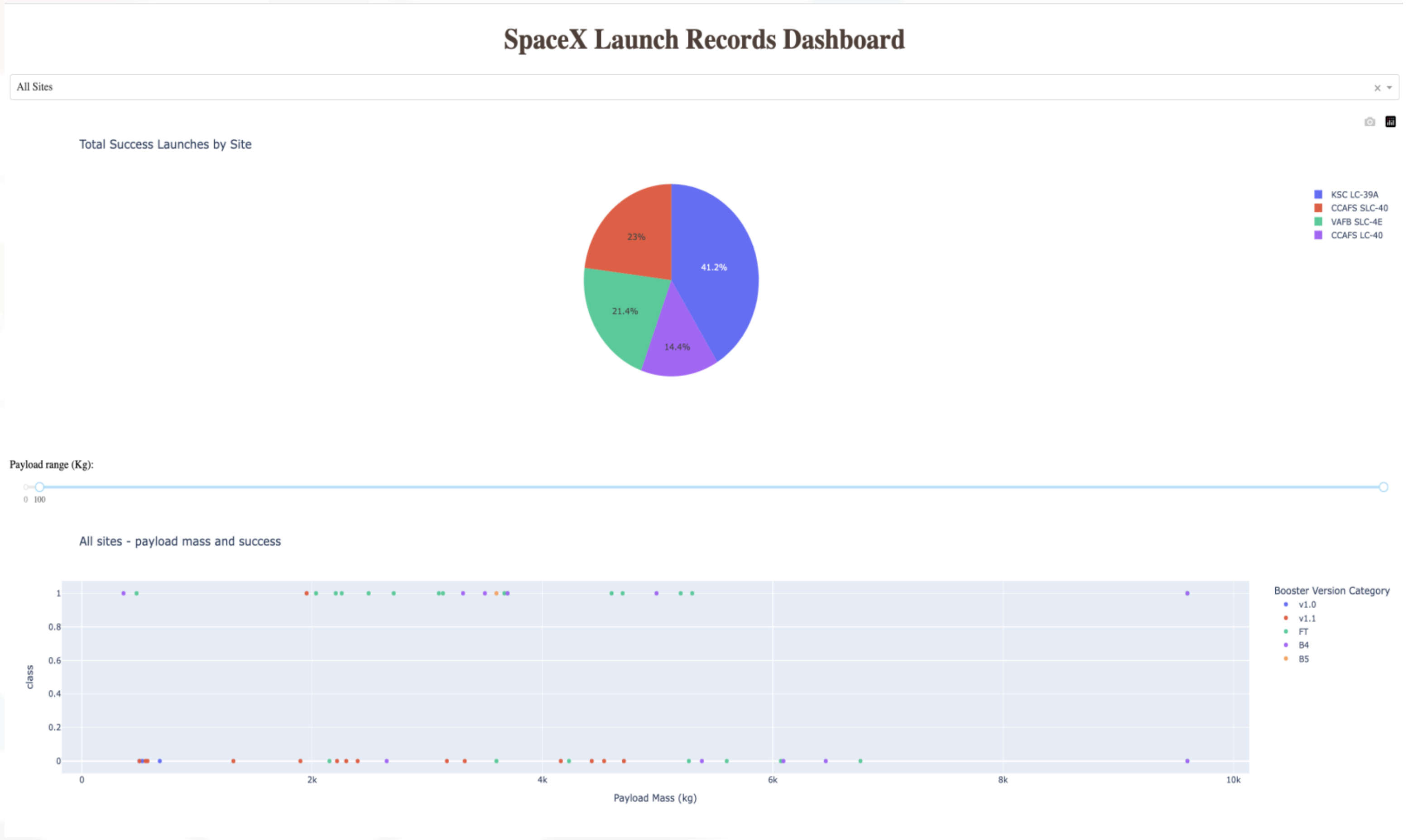


Lat: 28.5668 Long: -80.57526
1.29 KM
0.59 KM
Titan III Road
Samuel C Phillips Parkway
Merritt Island National Wildlife Refuge
33
CCAFS
Leaflet | Data by © OpenStreetMap, under ODbL.

## RESULTS: Dashboard with Plotly

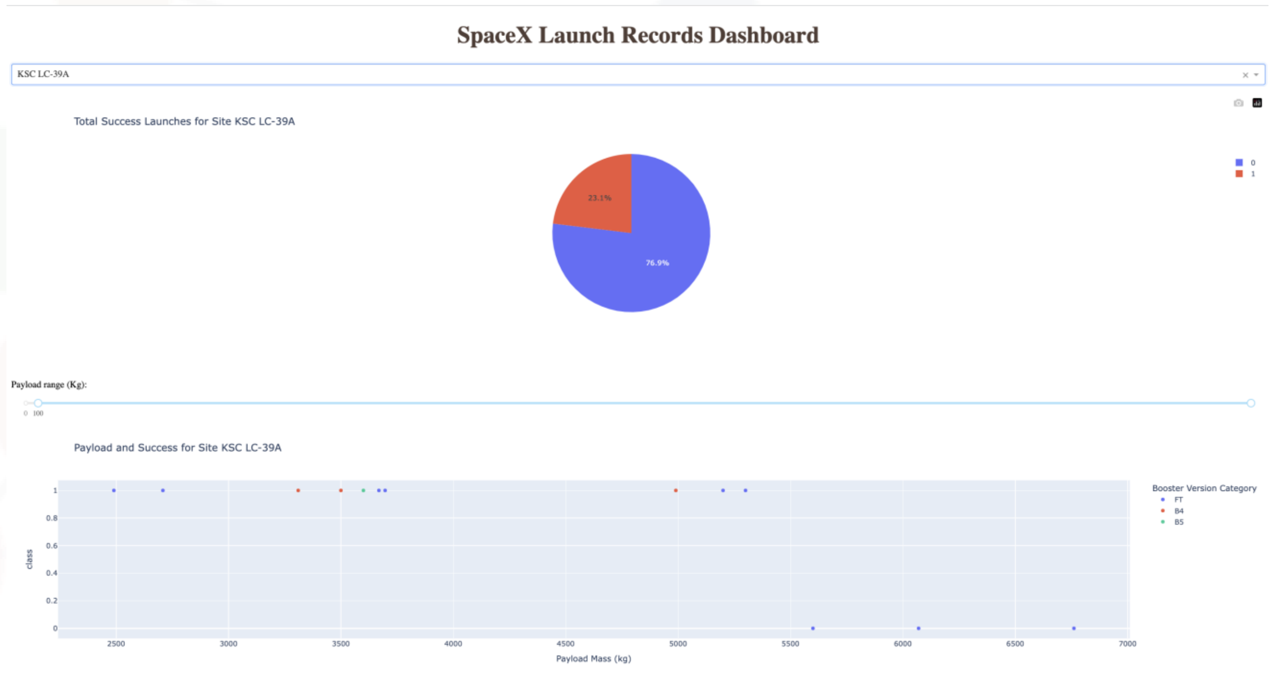
## RESULTS: Dashboard with Plotly - Launch Success

- Which site has the largest successful launches? KSC LC-39A
- Which site has the highest launch success rate? KSC LC-39A
- Which payload range(s) has the highest launch success rate? 2000kg - 4000kg
- Which payload range(s) has the lowest launch success rate? 6000kg - 8000kg
- Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest launch success rate? FT



## RESULTS: Dashboard with Plotly - Launch Site KSC LC-39A

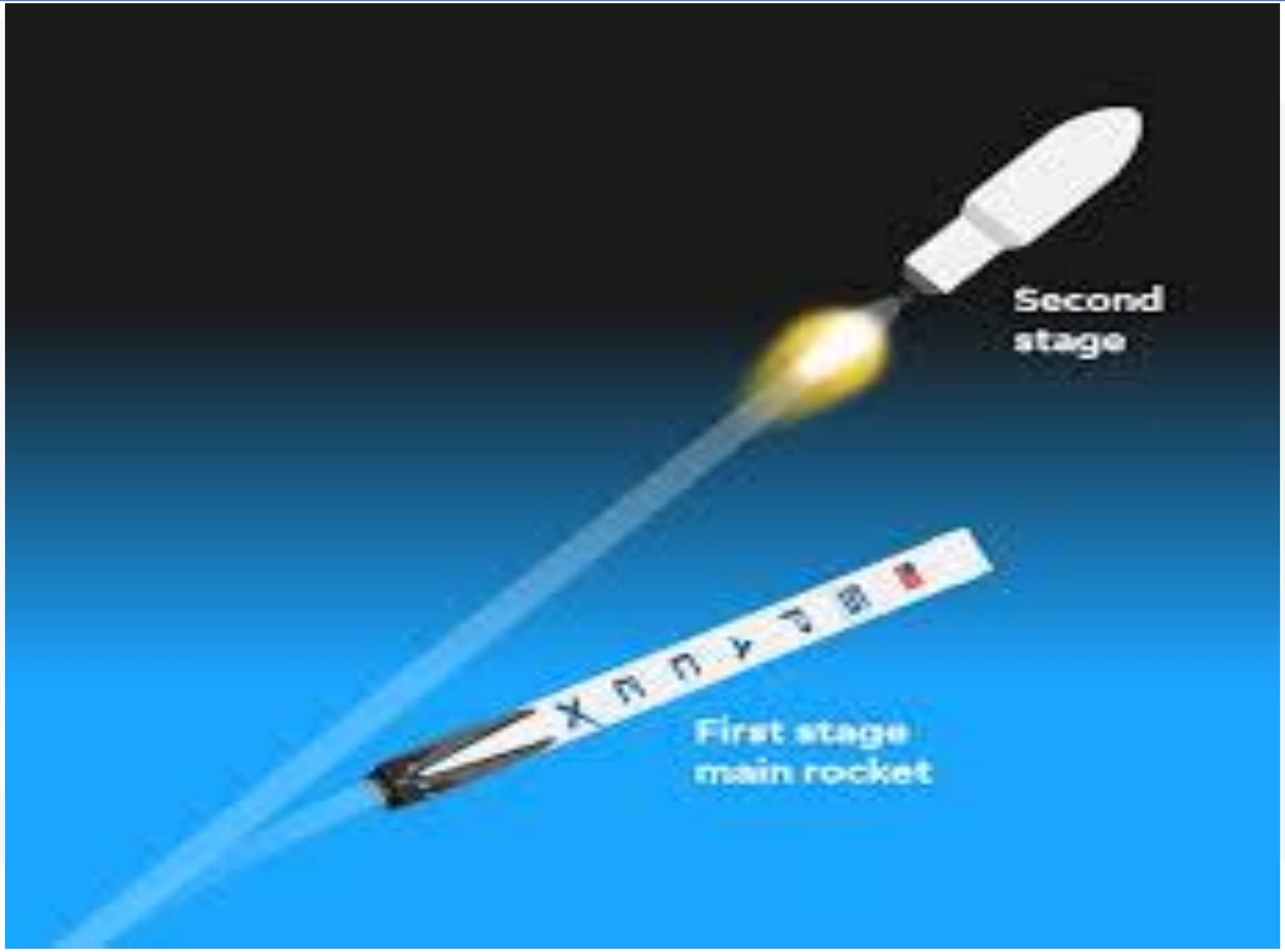
**KSC LC-39A** has achieved a 76.9% success rate, the highest success rate amongst launch sites


SpaceX Launch Records Dashboard
KSC LC-39A
Total Success Launches for Site KSC LC-39A
23.1%
76.9%
0
1
Payload range (Kg):
0
100
Payload and Success for Site KSC LC-39A
class
Payload Mass (kg)
Booster Version Category
FT
B4
B5

IBM Developer

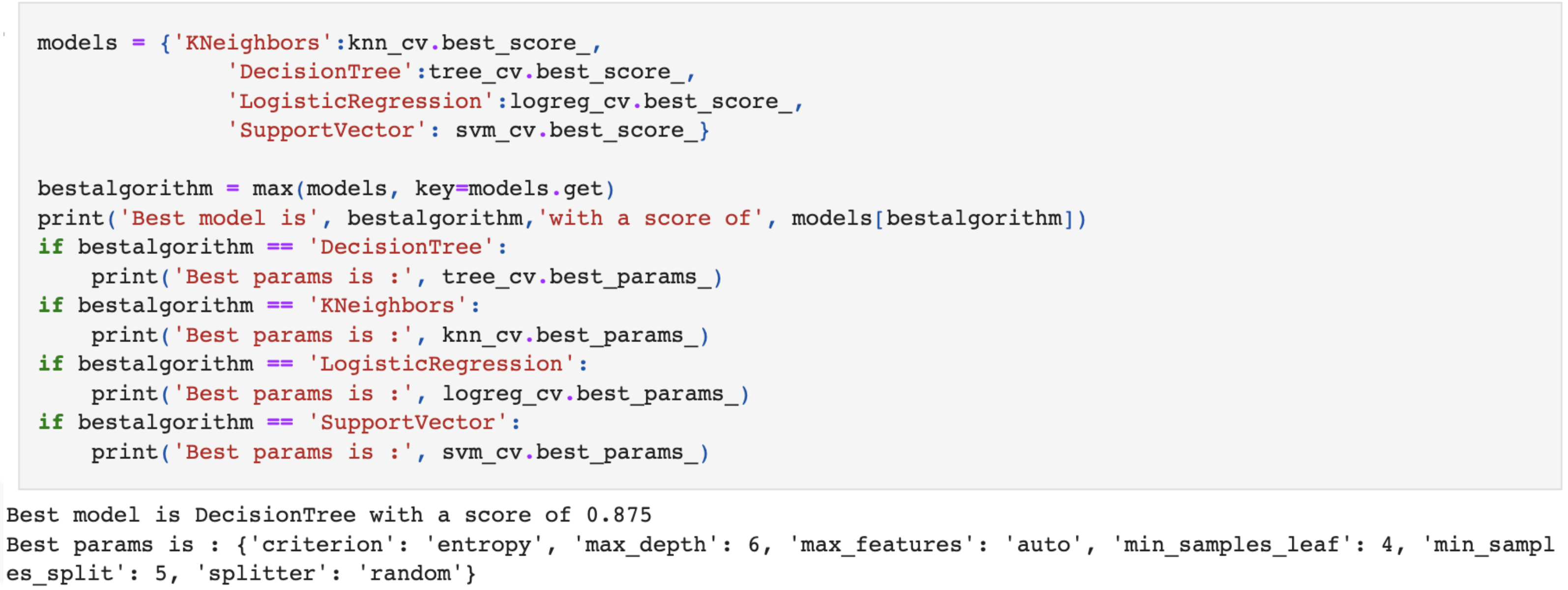
SKILLS NETWORK

## RESULTS: Predictive Analytics



## RESULTS: Predictive Analytics - Classification

All four models performed at about the similar level. The Decision Tree model has the highest classification accuracy



```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.875
Best params is : {'criterion': 'entropy', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_sampl
es_split': 5, 'splitter': 'random'}
```

## RESULTS: Predictive Analytics - Confusion Matrices

All the four models confusion matrices are identical

Confusion Matrix Outputs:

12 True positive

3 True negative

3 False positive

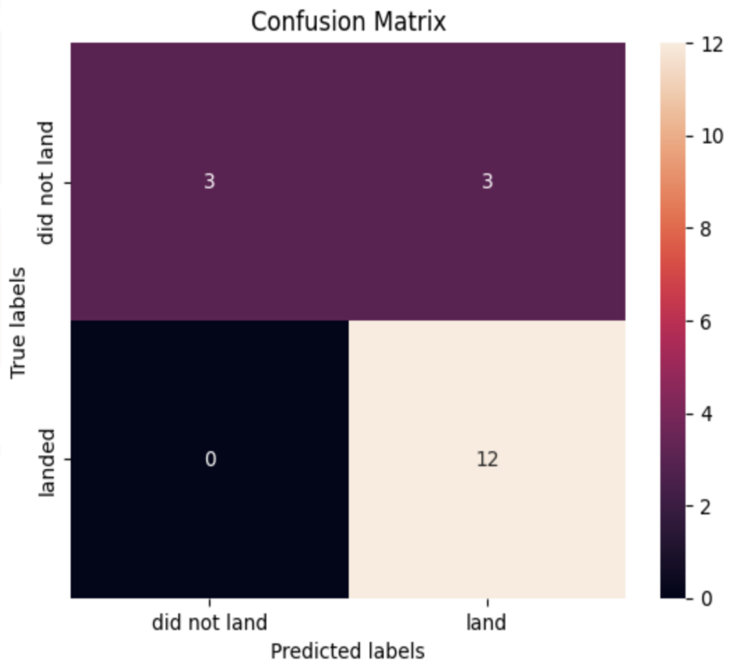
0 False Negative

Accuracy= (TP + TN) / (TP + TN + FP + FN) = 0.833

Precision= TP / (TP + FP)12 / 15 = 0.80

Recall= TP / (TP + FN)12 / 12 = 1

```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



# Conclusion

Through the analysis we concluded that:

- The models performed similarly on the test set with the decision tree model outperform other models
- The payloads range between 2000k to 4000k performed better than the heavy weighted payloads.
- Starting from the year 2013, launch success has increased over time.
- ES-L1, GEO, HEO, and SSO have a 100% success rate
- KSC LC-39A has the highest launch success rate among launch sites
- All the launch sites are close to the coastal line

## APPENDIX: Github URL

https://github.com/kevin-tdc/Applied-data-science-capstone/tree/main

jupyter-labs-spacex-data-collection-api.ipynb

jupyter-labs-webscraping.ipynb

labs-jupyter-spacex-data_wrangling_jupyterlite.ipynb

jupyter-labs-eda-dataviz.jupyterlite.ipynb

jupyter-labs-eda-sql-coursera_sqllite.ipynb

lab_jupyter_launch_site_location.jupyterlite.ipynb

spacex_dash_app

SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Data Science Final Assignment