# Winning Space Race with Data Science

Tasmina Rahman

22nd October 2021


# Outline

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


# Executive Summary

**Methodology Applied**

- Data collection through API and Web Scraping
- Data wrangling
- EDA with SQL and Visualization
- Creating an interactive map with Folium
- Creating a dashboard with Plotly Dash
- Building models using Logistic Regression, Decision Tree and K-Nearest Neighbors

**Summary**

- Logistic regression model was chosen for its popularity and application in the business.
- Successful launches are determined by lighter payload, which orbit the launch is being sent to, the proximity of coast and highway, frequency of launches by launch site (increase in launches leads to higher success rate).

# Introduction

**Project Scope**

- This report has been created as part of the IBM Applied Data Science Capstone Project.
- The Capstone Project is based on the Space industry, it analyses Space X data from its website to determine whether SpaceX Falcon 9 first stage will land successfully. Space X states that the cost of Falcon 9 launch is $62 million dollars compared to $165 million dollars stated by its competitors. Much of the savings are due to its ability to reuse its first stage components.
- In this project Space Y wants to compete with Space X, so uses machine learning techniques to establish whether Falcon 9 rocket launch first stage will land successfully. This information will help Space Y to bid against Space X for rocket launch business.

**Insights to derive from machine learning**

- What factors determine a successful launch?
- What are the key variables in determining success/failure?

Section 1

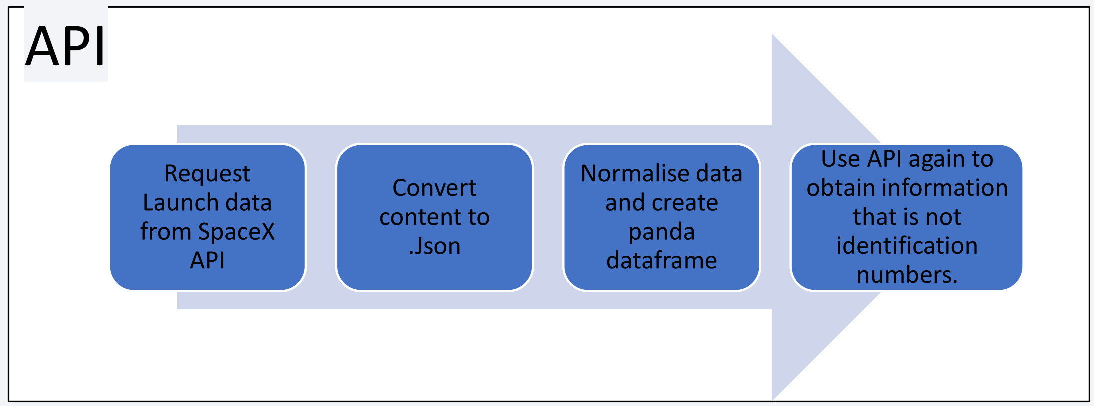
# Methodology

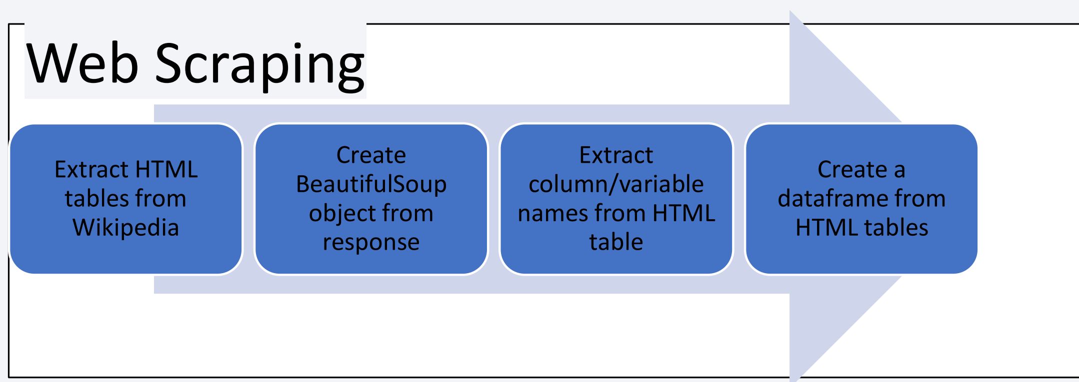
# Methodology

- Data collection methodology: Using SpaceX Rest API and Web Scraping.
- Perform data wrangling : Data was prepared by filling missing data using One Hot Encoding feature, selecting required data and column.
- Perform exploratory data analysis (EDA) using visualization and SQL: Analysis was carried out using SQL and visualization to determine relationship/correlation between variables.
- Perform interactive visual analytics using Folium and Plotly Dash: Explore launch data further by using folium maps and dashboard reporting.
- Perform predictive analysis using classification models: Build, tune, evaluate classification models

# Data Collection

- Data was collected from Wikipedia through web scraping (BeautifulSoup) and REST API to predict success/failure of Falcon 9 First Stage landing.
- This data provided a dataframe with Launch data which included information around Landing outcome, Launch sites, payload mass carried, rocket parts new/used for each  launch.



# Data Collection – SpaceX API

- Use API to request and parse SpaceX launch data
- Use json_normalise method to convert into a dataframe
- Clean data using customised functions
- Combine columns to dictionary
- Create a panda dataframe from the above dictionary

The process flow highlights the steps taken to convert information using API to a dataframe used for analysis.



https://github.com/trahman21-hub21/Applied-Data-Science-Capstone/blob/main/Data%20Collection%20API%20Lab%201.ipynb

# Data Collection – Web Scraping

This highlights process used to create a dataframe through web scrapping

1. Use HTTP GET method to request HTML response
2. Create HTML response using BeautifulSoup Object
3. Collate column names from HTML table header
4. Parse HTML tables to create a dataframe
5. Convert to csv file from dataframe



https://github.com/trahman21-hub21/Applied-Data-Science-Capstone/blob/main/Complete%20the%20Data%20Collection%20with%20Web%20Scraping%20Lab%202.ipynb

# Data Wrangling

- Data wrangling process prepares the data for analysis and provides insights from exploratory analysis to establish the right labels for training supervised models.

- This stage in the analysis simplifies all successful/unsuccessful launch outcomes to a label called ‘Class’ which allows us to establish insight to launch site success rates.

CALCULATE NUMBER OF LAUNCHES ON EACH SITE

CALCULATE THE NUMBER AND OCCURRENCE OF EACH ORBIT

CALCULATE THE NUMBER AND OCCURRENCE OF MISSION OUTCOME PER ORBIT TYPE

CREATE LANDING OUTCOME LABEL FROM OUTCOME COLUMN

DETERMINE SUCCESS RATES



https://github.com/trahman21-hub21/Applied-Data-Science-Capstone/blob/main/Complete%20the%20EDA%20Lab%203.ipynb

# EDA with Data Visualization

## Various visualization charts were utilised to provide insights.

Scatter plots were used to explore relationship between the following variables:

Flight Number VS Launch Site -Payload VS Launch Site
-Success rate VS orbit type - Flight Number VS Payload
-Flight Number VS Orbit -Flight number VS Orbit type

Bar charts are helpful to depict relationships between variables, changes and comparisons of different groups. The following relationships were explored:

-Orbit VS Class

Line graphs are effective in showing changes over short and long-term timeframes.

- Success VS year



https://github.com/trahman21-hub21/Applied-Data-Science-Capstone/blob/main/Spacex_EDA_with_Visualization%20Lab%205.ipynb

# EDA with SQL

SQL queries were performed to gather further insight into the data. The following queries were executed;

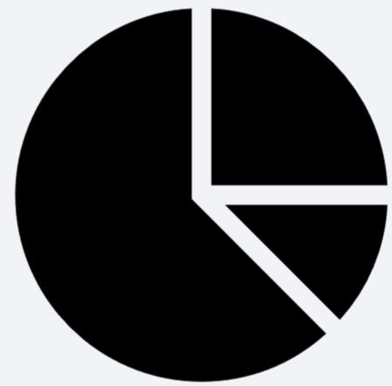
1. ***Display the names of the unique launch sites in the space mission***
2. ***Display 5 records where launch sites begin with the string 'CCA'***
3. ***Display the total payload mass carried by boosters launched by NASA (CRS)***
4. ***Display average payload mass carried by booster version F9 v1.1***
5. ***List the date when the first successful landing outcome in ground pad was achieved.***
6. ***List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000***
7. ***List the total number of successful and failure mission outcomes***
8. ***List the names of the booster versions which have carried the maximum payload mass; use a subquery***
9. ***List the failed landing outcomes in drone ship, booster versions and launch site names in year 2015***
10. ***Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order***



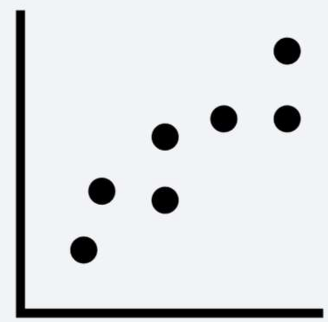
https://github.com/trahman21-hub21/Applied-Data-Science-Capstone/blob/main/Spacex_EDA_SQL%20Lab4.ipynb

# Build a Dashboard with Plotly Dash

## Plotly Dash was used to create an interactive dashboard to analyse/visualise launch data.



**PIE CHART**
Pie chart was created to analyse total launch success rates VS launch sites. The interactive pie chart also allows user to drill down to look at success rates of each specific sites. This was achieved by adding drop down 'Input components' along with 'call back functions' to initialise user choice of dropdown option.



**SCATTER PLOT**
Scatter plot highlights any relationship between 'Payload Mass' and launch success outcome by 'Booster Version Category'. Scatter plots are useful as it allows comparison of large number of datapoints and shows whether variables are positively or negatively correlated.
Scatter plot included 'Range Slider' to assess 'Payload Mass' and 'Call back function' to interactively drill down information by Payload mass.



https://github.com/trahman21-hub21/Applied-Data-Science-Capstone/blob/main/Spacex_dash_app%20Lab%206%20(2).py

# Build an Interactive Map with Folium

## An interactive map with Folium was built to provide Launch Site analysis.

- A circle marker was created using latitude and longitude from the dataset, so we can visually see all the launch sites.
- A color marker of green and red was created to mark launch success and failures in the map.
- Several distance calculations were made against launch site and other markers such as Railway, Coastline, Highway and City to determine their proximity to launch sites and whether that is a determining factor in success/failure of launch sites.



https://github.com/trahman21-hub21/Applied-Data-Science-Capstone/blob/main/Spacex_folium_maps%20Lab%207.ipynb

# Predictive Analysis (Classification)

**Model Build**

1.Load dataframe
2. Standardize data
3. Create training /test datasets
4. Set up parameters
5. Use GridSearchCV function to loop through predefined parameters

**Evaluation**

1. Check model score
2.Analyse model using Confusion Matrix

**Best Fit Model**

1. Assess best model

https://github.com/trahman21-hub21/Applied-Data-Science-Capstone/blob/main/Spcex_Machine%20Learning%20Prediction%20Lab%208.ipynb

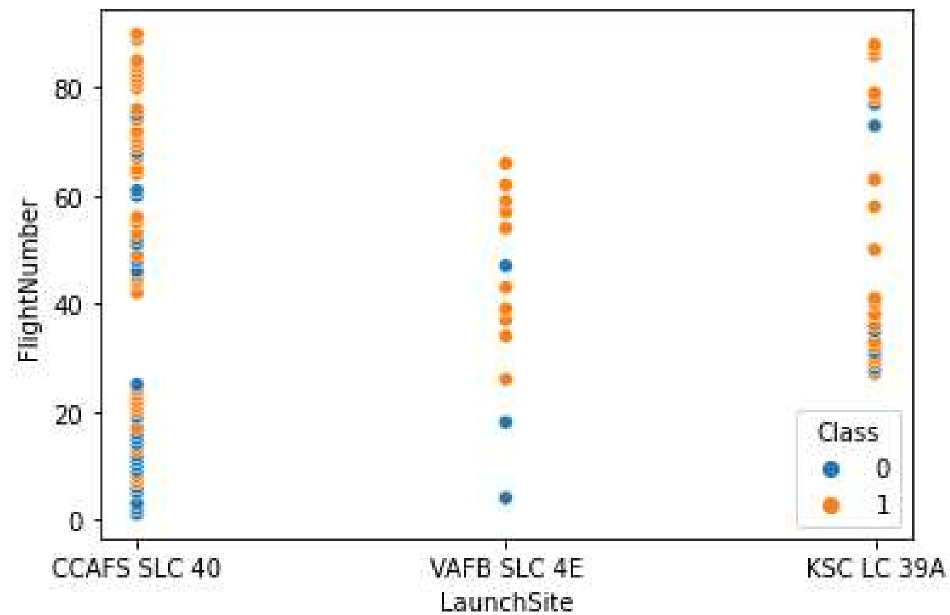
# Results

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

**Insights drawn from EDA**

# Flight Number vs. Launch Site

- The scatter plot shows frequency of flights is correlated with launch success.
- As flight number increases so does the success rate of launch site.

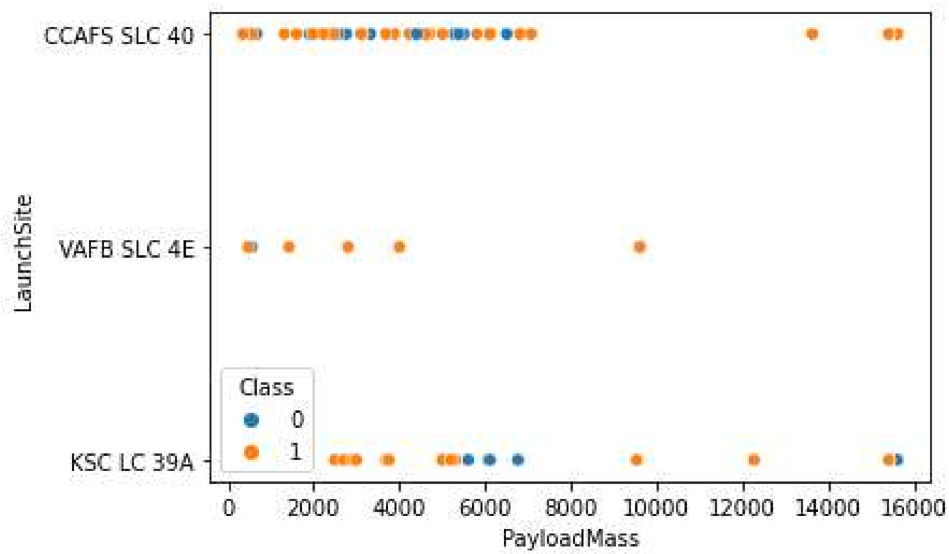


# Launch Site vs Payload

The scatter plot highlights several things;

1. All three sites have launched mostly lighter rockets with successful outcome.

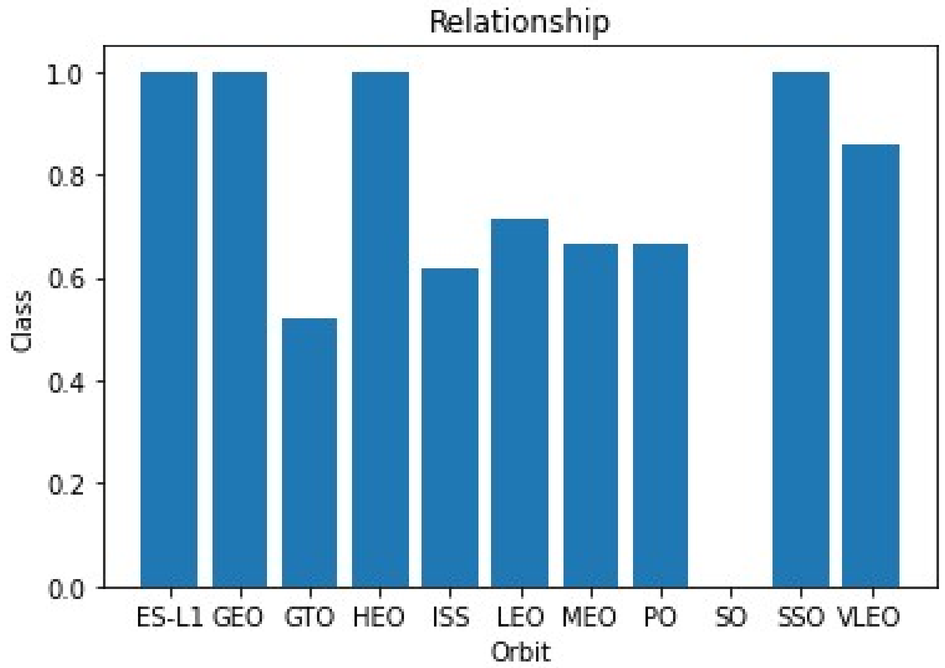
2. CCAFS SLC 40 have launched heavier payload rockets with successful outcomes, the site also launched the most rockets overall. This suggests number of flights leads to more successful outcome with heavier payload rockets. It suggests the more rockets they launches they gain experience in managing heavier payloads.



# Success Rate vs. Orbit Type

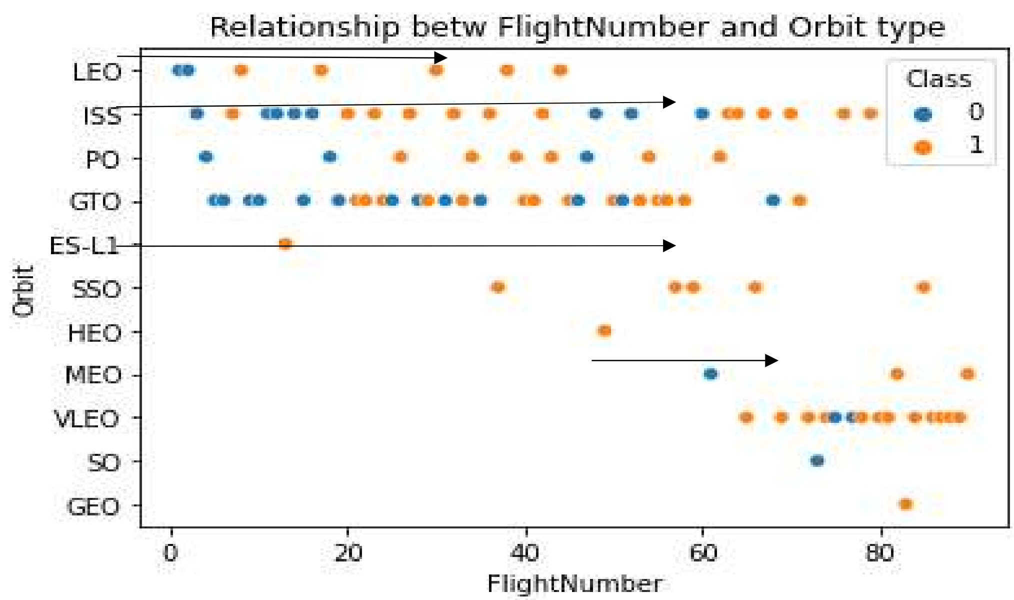
- The bar chart highlights the relationship between Class and Orbit.

- Rocket launched in ES-L1, GEO, HEO and SSO orbit demonstrate the highest success rate.



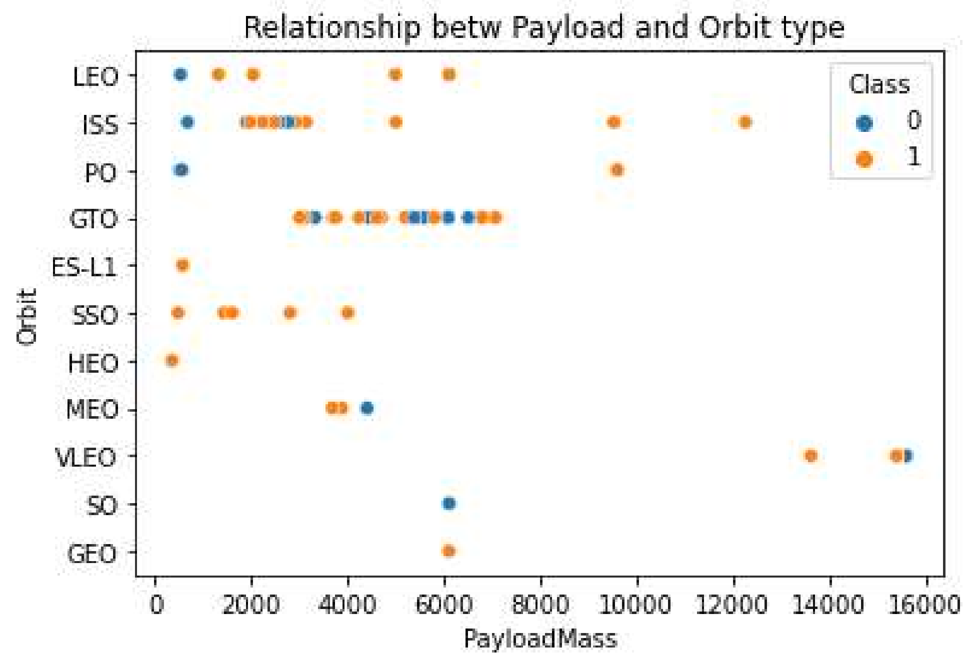
# Flight Number vs. Orbit Type

- The frequency of flights is correlated with launch success. As volumes of flight increases so does the likelihood of successful launches.
- There appears to be no relationship between GTO orbit and flight number.



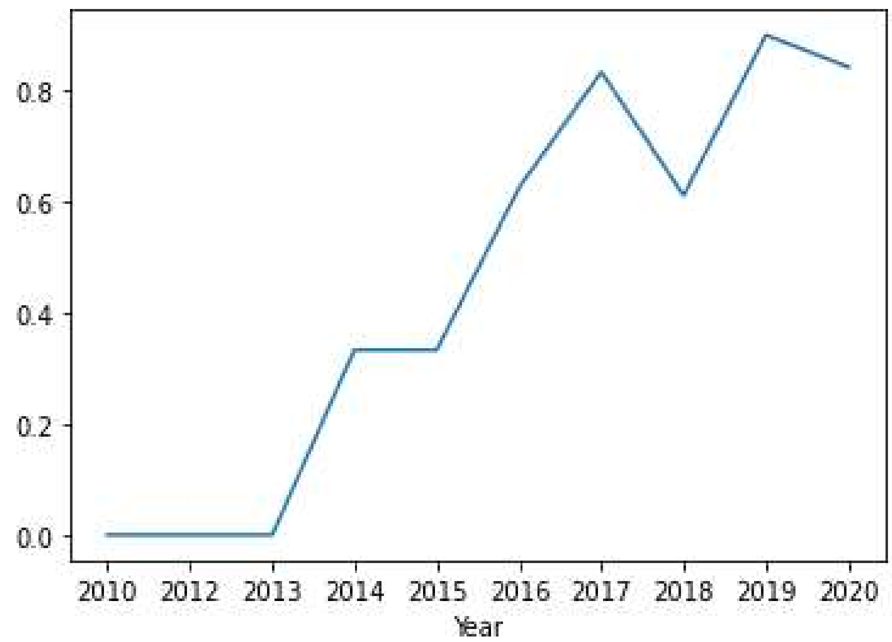
# Payload vs. Orbit Type

- ISS orbit demonstrates most successful rocket launches with heavy payloads, over 8000 kg.
- Heavy payloads have a negative influence on GTO orbits.
- SSO/LEO show multiple successful launches with lighter payloads; under 6000kg payload mass.



# Launch Success Yearly Trend

The line chart demonstrates the increase in successful launches since 2013, which peaked in 2019. The drop in 2020 could be the effects of shut down of operations due to Covid.




mirror_mod.mirror_object
operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
_operation == "MIRROR_Y"
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
_operation == "MIRROR_Z"
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True
OPERATOR CLASSES

# EDA WITH SQL

# All Launch Site Names


%sql SELECT DISTINCT (LAUNCH SITE) FROM SPACEDATASET

| UNIQUE LAUNCH SITES |
| --- |
| CCAFS LC-40 |
| CCAFS SLC-40 |
| KSC LC-39A |
| VAFB SLC-4E |

The query returns only unique (‘distinct’) values from the column ‘Launch site’.

# Launch Site Names Begin with 'CCA'


%sql SELECT * FROM SPACEDATASET WHERE LAUNCH_SITE LIKE ‘CCA%’ LIMIT 5

Asterics (*) returns all values from the table and ‘Like’ is an SQL operator that looks for specified pattern in a column. ‘Limit 5’ retrieves the top 5 rows of the table.

| 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 | 07:44:00 | F9 v1.0 B0005 | CCAFS LC-40 | Dragon demo flight C2 | 525 | LEO (ISS) | NASA (COTS) | Success | No attempt |
| 2012-10-08 | 00:35:00 | 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


```
%sql SELECT SUM(PAYLOAD_MASS_KG_)AS
TOT_PAYLOAD_MASS FROM SPACEDATASET
WHERE CUSTOMER='NASA (CRS)'
```

| TOT_PAYLOAD_MASS |
| --- |
| 45596 |

The 'sum' function returns the 'sum' of all payload mass kg that falls under customer 'NASA (CRS)'

# Average Payload Mass by F9 v1.1


```
%sql SELECT AVG(PAYLOAD_MASS_KG_)AS
AVG_PAYLOAD_MASS FROM SPACEDATASET
WHERE BOOSTER_VERSION = ‘F9 V1.1’
```

| AVG_PAYLOAD_MASS |
| --- |
| 2928 |

‘AVG’ function returns the average of ‘payload mass kg’, the ‘where’ function returns an average when booster version is ‘F9 V1.1’

# First Successful Ground Landing Date


```
%sql SELECT MIN(DATE) AS FIRST_DATE FROM
SPACEDATASET WHERE LANDING_OUTCOME LIKE
'SUCCESS (GROUND PAD)'
```

| FIRST_DATE |
|---|
| 2015-12-22 |

'Min' function returns the earliest date of the successful ground pad launch. The 'where' selects the specified landing outcome and 'Like' operator matches the string and retrieves the value.

# Successful Drone Ship Landing with Payload between 4000 and 6000


%sql SELECT BOOSTER_VERSION FROM SPACEDATASET WHERE PAYLOAD_MASS_KG_ BETWEEN 4000 AND 6000 AND LANDING_OUTCOME LIKE 'SUCCESS (DRONE SHIP)'

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

The query selects Booster version with a 'where' clause. The 'where' clause selects payload between 4000 and 6000 and the 'like' operator retrieves only successful drone ship landing. The 'and' allows multiple conditions to be specified.

# Total Number of Successful and Failure Mission Outcomes


```
%sql SELECT MISSION_OUTCOME,
COUNT(MISSION_OUTCOME) AS TOTAL_OUTCOME FROM
SPACEDATASET GROUP BY MISSION_OUTCOME ORDER BY
MISSION_OUTCOME
```

| TOTAL_OUTCOME | |
|---|---|
| Failure (in flight) | 1 |
| Success | 99 |
| Success (payload status unclear) | 1 |

The query calculates the total number of failure and success as defined in mission_outcome. The ‘count’ function and ‘group by’ aggregates the values in the mission_outcome and ‘order by’ organizes it alphabetically. The default is always ascending.

# Boosters Carried Maximum Payload


%sql SELECT BOOSTER_VERSION, PAYLOAD, PAYLOAD_MASS_KG FROM SPACEDATASET  WHERE PAYLOAD_MASS_KG_ IN (

SELECT MAX(PAYLOAD_MASS_KG_) AS PAYLOAD FROM SPACEDATASET ORDER BY PAYLOAD)

| booster_version | payload | payload_mass__kg_ |
|---|---|---|
| F9 B5 B1048.4 | Starlink 1 v1.0, SpaceX CRS-19 | 15600 |
| F9 B5 B1049.4 | Starlink 2 v1.0, Crew Dragon in-flight abort test | 15600 |
| F9 B5 B1051.3 | Starlink 3 v1.0, Starlink 4 v1.0 | 15600 |
| F9 B5 B1056.4 | Starlink 4 v1.0, SpaceX CRS-20 | 15600 |
| F9 B5 B1048.5 | Starlink 5 v1.0, Starlink 6 v1.0 | 15600 |
| F9 B5 B1051.4 | Starlink 6 v1.0, Crew Dragon Demo-2 | 15600 |
| F9 B5 B1049.5 | Starlink 7 v1.0, Starlink 8 v1.0 | 15600 |
| F9 B5 B1060.2 | Starlink 11 v1.0, Starlink 12 v1.0 | 15600 |
| F9 B5 B1058.3 | Starlink 12 v1.0, Starlink 13 v1.0 | 15600 |
| F9 B5 B1051.6 | Starlink 13 v1.0, Starlink 14 v1.0 | 15600 |
| F9 B5 B1060.3 | Starlink 14 v1.0, GPS III-04 | 15600 |
| F9 B5 B1049.7 | Starlink 15 v1.0, SpaceX CRS-21 | 15600 |

- Selects Booster version, payload and payload mass kg, sub query selects max payload mass, order by organizes the payload in ascending order.

# 2015 Drone Ship Failure Launch Records


```
%sql SELECT LANDING_OUTCOME, BOOSTER_VERSION,
LAUNCH_SITE FROM SPACEXDATASET WHERE
LANDING_OUTCOME LIKE ‘FAILURE (DRONE SHIP)’ AND
YEAR (DATE) = 2015
```

| landing__outcome | booster_version | launch_site |
|---|---|---|
| Failure (drone ship) | F9 v1.1 B1012 | CCAFS LC-40 |
| Failure (drone ship) | F9 v1.1 B1015 | CCAFS LC-40 |

The ‘where’ selects all failed drone ship launches and year (date) function converts the date into year.

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

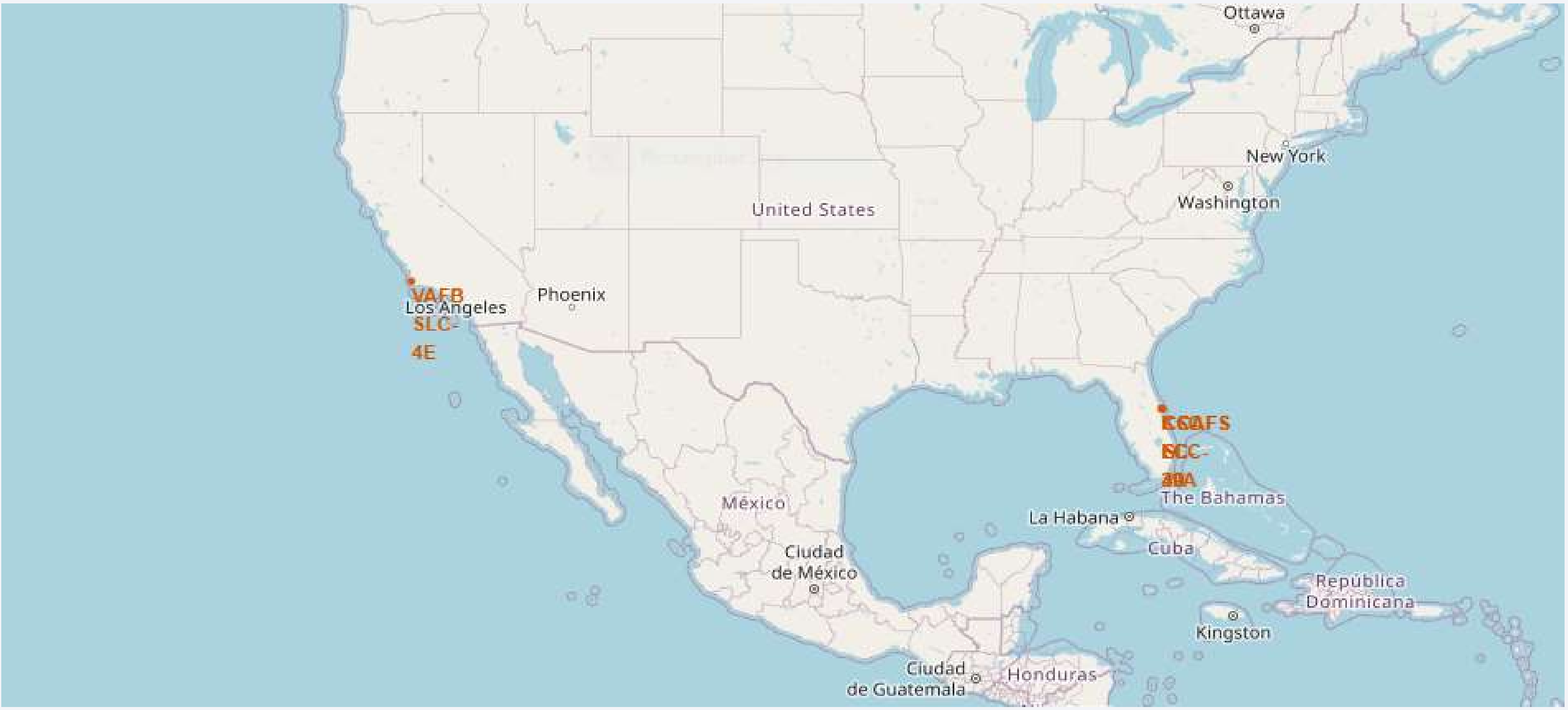

```
%sql COUNT(LANDING_OUTCOME) AS COUNT, LANDING_OUTCOME
FROM SPACEXDATASET WHERE LANDING_OUTCOME IN
(‘SUCCESS(GROUND_PAD’, ‘FAILURE (DRONE SHIP), ‘SUCCESS
(DRONE_SHIP) ’’) AND DATE BETWEEN ‘2010-06-04’ AND ‘2017-03-20’
ORDER BY COUNT(LANDING_OUTCOME) DESC
```

| COUNT | landing__outcome |
|---|---|
| 5 | Failure (drone ship) |
| 5 | Success (drone ship) |
| 3 | Success (ground pad) |

Count function aggregates landing outcome  for the selected ones specified in the ‘where’ condition. ‘Date between’ ensures landing outcome is only aggregated for those that meet the time frame criteria. The ‘and’ allows multiple condition to be specified.

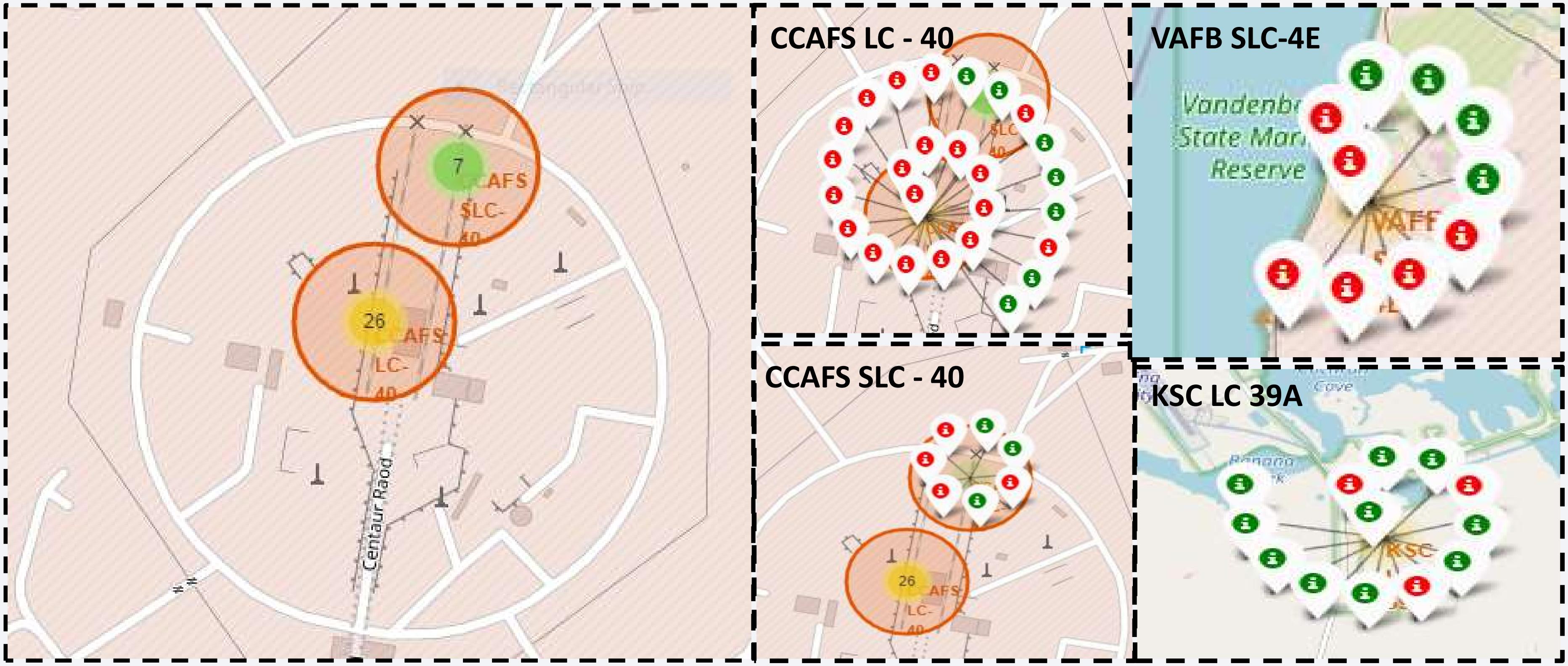

Section 4
Launch Sites
Proximities Analysis

# Space X Launch Sites



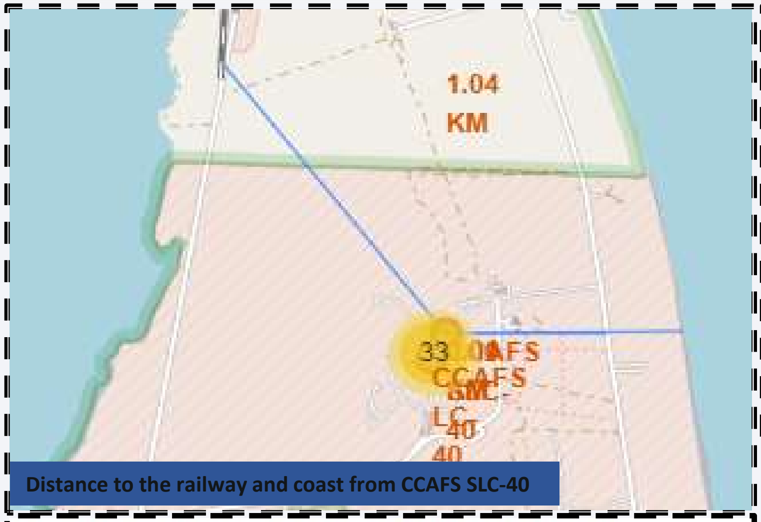

The map highlights Space X launch sites in America, distributed around the coasts of California and Florida.

# Color Markers Displaying Successful/Unsuccessful Launches

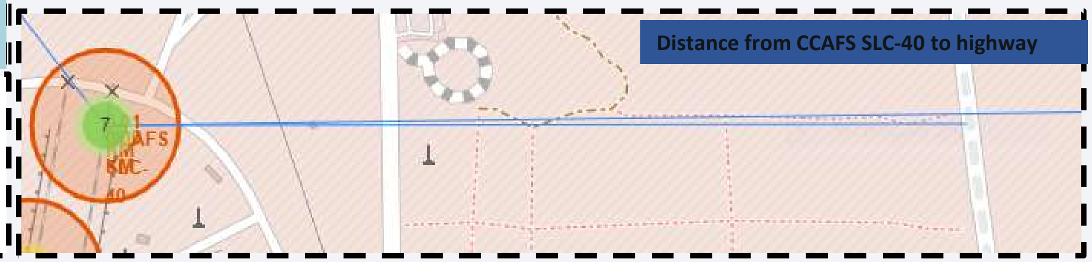



The above map highlights KSC LC-39A launch site as the most successful launch site in Florida.

# Are launch sites located near landmarks?







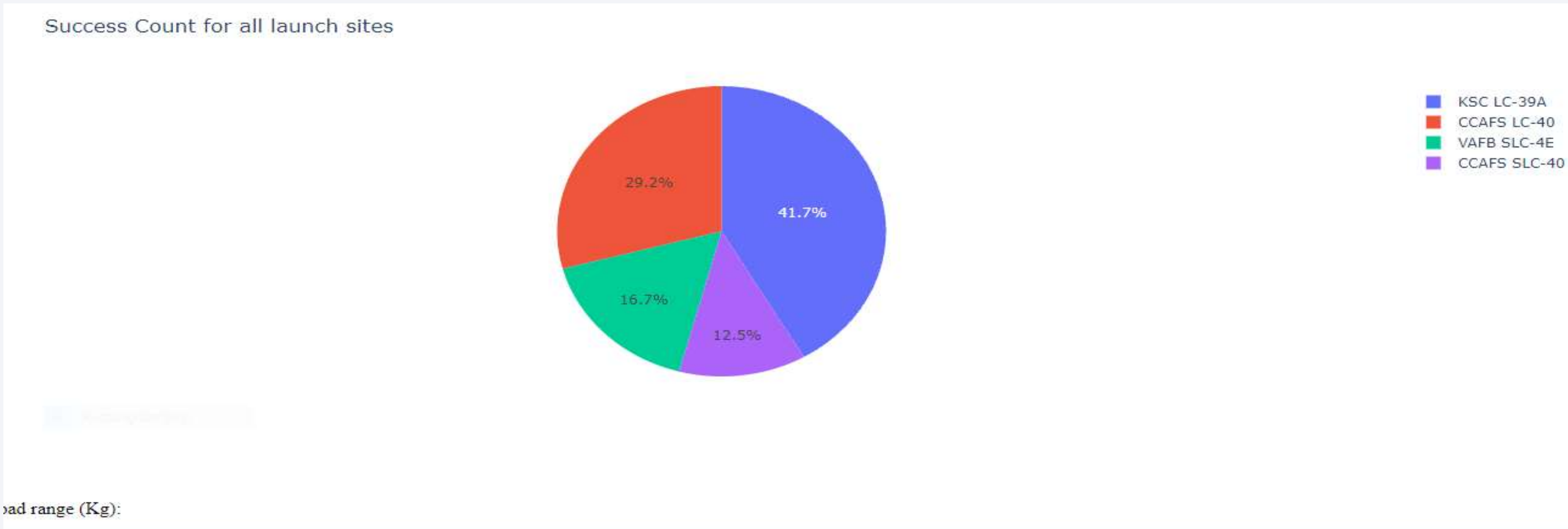

Are launch sites in close proximity to railways?
Are launch sites in close proximity to highways?
Are launch sites in close proximity to coastline?
Do launch sites keep certain distance away from cities?

Launch sites are build in close proximity to coastline and highways as rocket launch is transported to coastline via highways.
Distance is also maintained between railway and launch sites but not with as much distance as cities.
It is also crucial for launch sites not to be in close proximity of cities as shown above.

Section 5

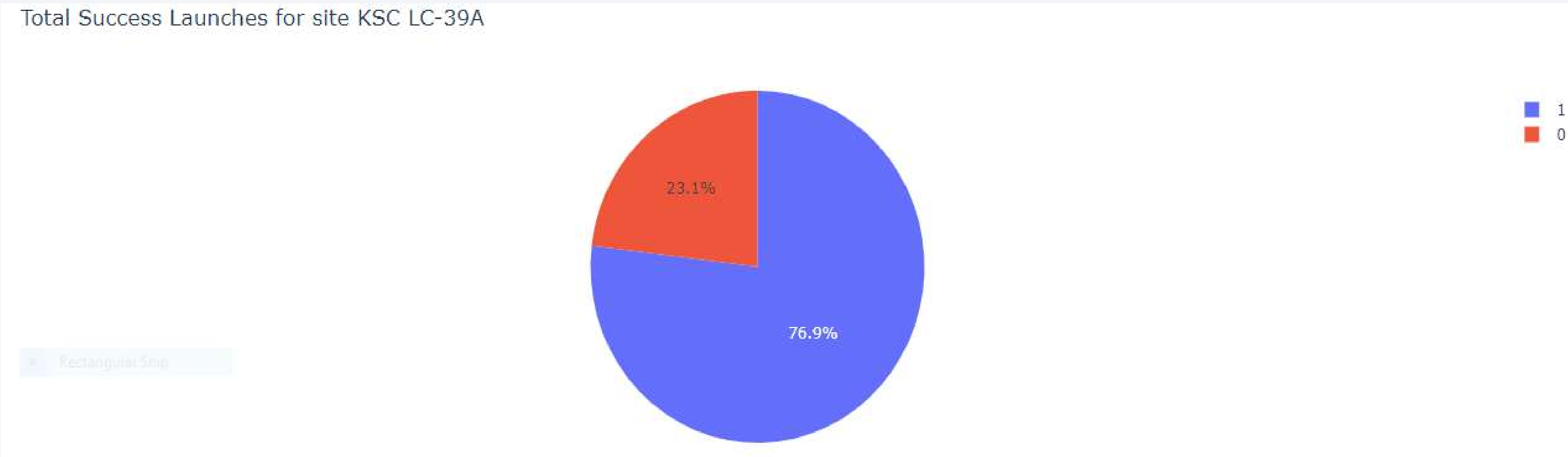
# Build a Dashboard with Plotly Dash

# Launch Site Success



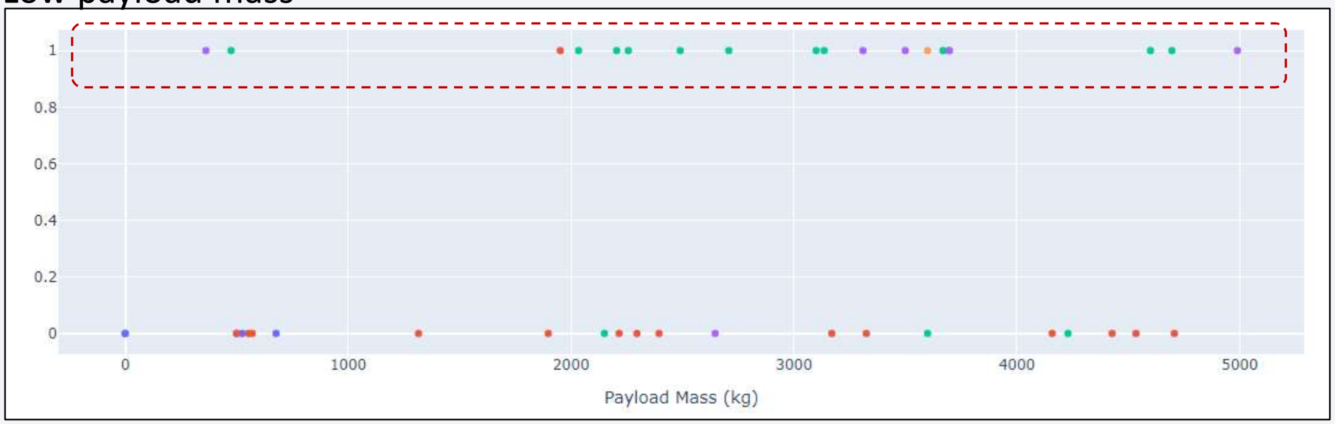

The graph shows KSC LC-39A launch sites have been most successful with its rocket launch.

# 77% of launches have been successful at KSC LC-39A

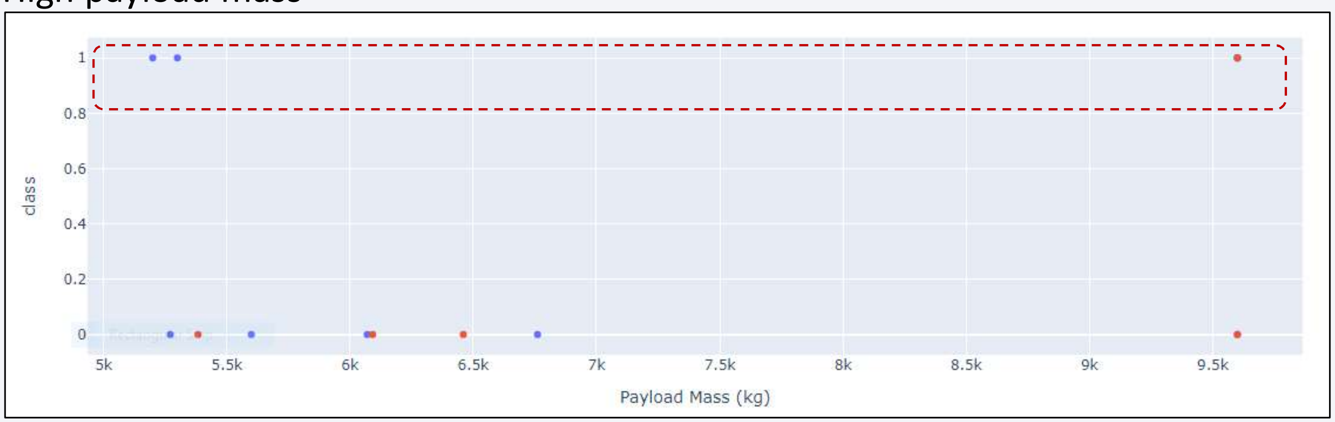


# Payload Range Analysis

Low payload mass




High payload mass




The graphs demonstrate low payload launches are more successful than high load payload mass.

Therefore, launches with low pay load mass are more likely to be successfully launched than heavier payload rocket launches.

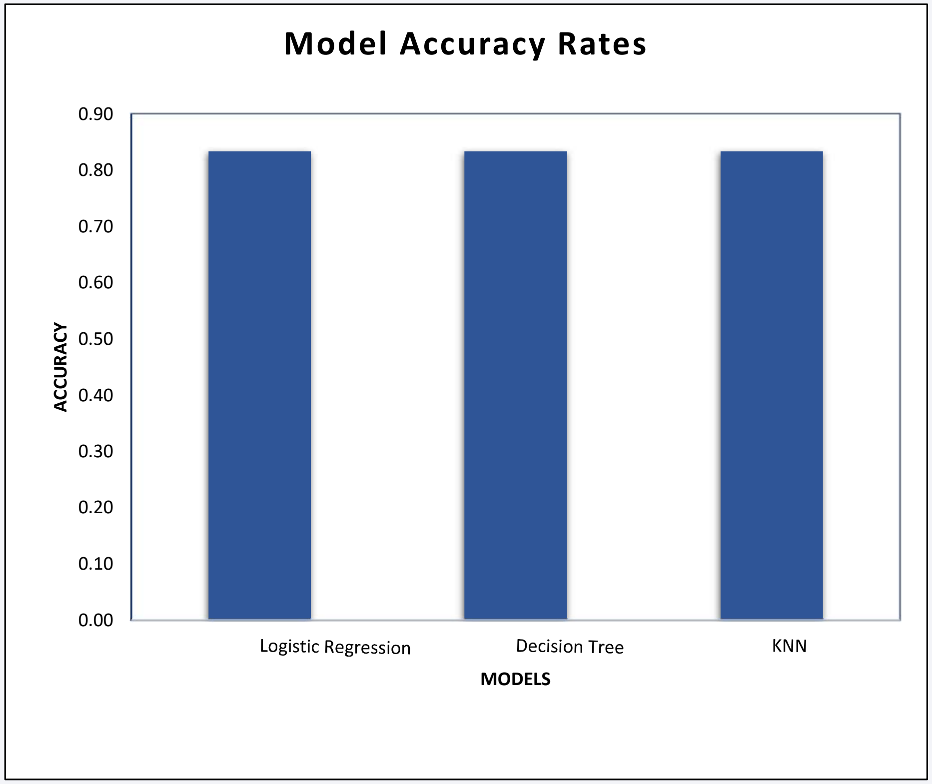
Section 6

# Predictive Analysis (Classification)

# Classification Accuracy

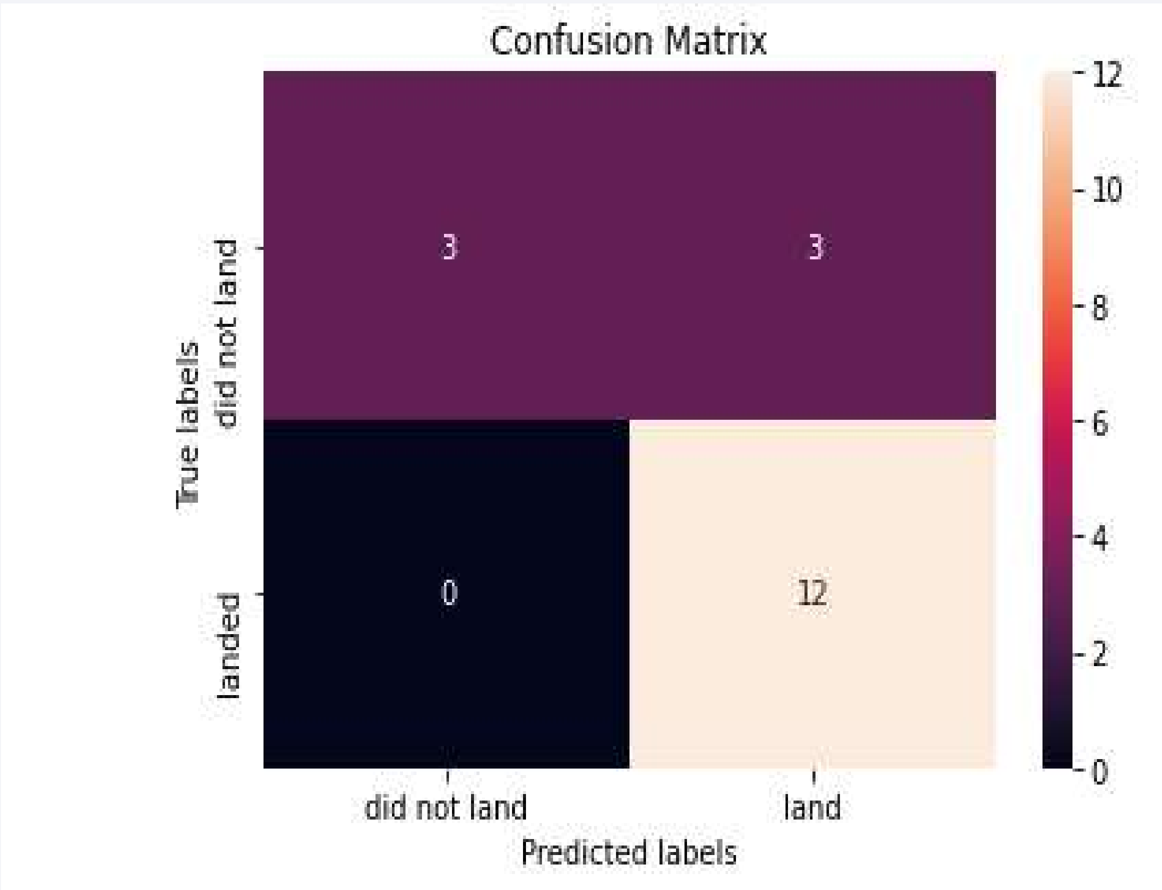
| Models | Accuracy % |
|---|---|
| Logistic Regression | 0.83 |
| Decision Tree | 0.83 |
| KNN | 0.83 |

- All the models shows the same accuracy rate.
- This could be due to small samples.
- The models show 83% accuracy rate using test data.




**Selected input models: FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins,Reused, Legs,LandingPad, Block,ReusedCount,Serial,Longitude, Latitude, Class**

# Confusion Matrix




- The confusion matrix highlight how the model has performed, it breaks down predicted values against actual values.
- We can see there is an issue of False Positive, 3 customers have been misclassified by the model as landed.
- The model has accurately predicted 12 launches that landed successfully as based on the actual data.

# Conclusions

- Logistic regression is the chosen model as its easy to apply and explain across the business as all the models have same accuracy results.
- KSC LC-39A launch site has launched the most successful launches and is near a coast and highway.
- Low weighted payload rocket launches are more successful than heavier payloads.
- There is relationship between flight number and success rate, successful launch increases with number of flights.
- ISS, SSO and LEO orbits have also demonstrated higher success rate with lighter payloads.

# Appendix

- Plotly dash reports

# Plotly Dash report

# Import required libraries import pandas as pdimport dashimport dash_html_components as htmlimport dash_core_components as dccfrom dash.dependencies import Input, Outputimport plotly.express as px#

Read the airline data into pandas dataframespacex_df = pd.read_csv("spacex_launch_dash.csv")max_payload = spacex_df['Payload Mass (kg)'].max()min_payload = spacex_df['Payload Mass (kg)'].min()#

Create a dash applicationapp = dash.Dash(__name__)#

Create an app layoutapp.layout = html.Div(children=[html.H1('SpaceX Launch Records Dashboard', style={'textAlign': 'center', 'color': '#503D36', 'font-size': 40}),

# Add a dropdown list to enable Launch Site selection

# The default select value is for ALL sites

dcc.Dropdown(id='site-dropdown', options=[{'label': 'All Sites', 'value': 'ALL'}, {'label': 'CCAFS LC-40', 'value': 'CCAFS LC-40'}, {'label': 'CCAFS SLC-40', 'value': 'CCAFS SLC-40'}, {'label': 'KSC LC-39A', 'value': 'KSC LC-39A'},{'label': 'VAFB SLC-4E', 'value': 'VAFB SLC-4E'}, ], value= "ALL", placeholder= 'Select a lauch Site here', searchable=True),html.Br(),

# 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,value=[min_payload, max_payload]),

# Add a scatter chart to show the correlation between payload and launch success

html.Div(dcc.Graph(id='success-payload-scatter-chart')), ])

# Add a callback function for `site-dropdown` as input, `success-pie-chart` as output@app.callback(Output(component_id='success-pie-chart',component_property='figure'), Input(component_id='site-dropdown',component_property='value') )def get_pie_chart(site):#filtered_df=spacex_df if site=='ALL': filtered_df=spacex_df fig=px.pie(spacex_df,values='class', names='Launch Site', title=f'Success Count for all launch sites') return fig else: df_site_filtered=spacex_df[spacex_df['Launch Site']== site] filtered_df=spacex_df[spacex_df['Launch Site']==site] df1=filtered_df.groupby(['Launch Site','class']).size().reset_index(name='class count') fig = px.pie(df1, values='class count', names='class', title=f"Total Success Launches for site {site}") return fig# TASK # Add a callback function for `site-dropdown` and `payload-slider` as inputs, `success-payload-scatter-chart` as output@app.callback( Output(component_id='success-payload-scatter-chart', component_property='figure'), [Input(component_id='site-dropdown', component_property='value'), Input(component_id="payload-slider", component_property="value")])def generate_scatter(site,payload): low, high = payload mask = spacex_df[spacex_df['Payload Mass (kg)'].between(low,high)] if site == 'ALL': fig = px.scatter(mask, x='Payload Mass (kg)', y='class', color="Booster Version Category", title=f"Correlation between Payload and Success for all sites") return fig else: mask_filtered = mask[mask['Launch Site'] == site] fig = px.scatter(mask_filtered, x='Payload Mass (kg)', y='class', color="Booster Version Category", title=f"Correlation between Payload and Success for "+site) return fig

# Run the appif __name__ == '__main__': app.run_server()

# Thank you!