IBM Data Science Professional Certification

Capstone Project

Predictive Analysis of SpaceX Launches

Hannah Ostoja

October 2023

h.ostcode@outlook.com

Table of Contents

Executive Summary

To better understand the determinants of SpaceX's rocket launch success, a comprehensive data analysis journey of Falcon 9's historical geospatial, mechanical, and time series data was taken.

The analysis reveals that payload specifications, as supported by Sforza (2016), significantly influence launch outcomes, with heavier payloads tending towards certain landing sites, and an optimal payload range of 4000-6000 Kg. Furthermore, geo-meteorological variables, such as coastal and equatorial proximity, as indicated by Romanova, et al. (2013), play a pivotal role in determining optimal launch locales, with an upward trend towards success for sites located on Eastern coastlines closer to the equator under optimal weather conditions. Finally, predictive analysis revealed key features can predict launch success with 83% accuracy across varied models, suggesting the benefit of SpaceX's iterative approach of testing different variables and outcomes.

These findings underscore the criticality of intertwining mechanical designs with geographical insights to optimize launch success rates. As SpaceX continues its endeavors, as reflected in its Falcon User's Guide (2021), integrating these data-driven insights could further elevate its success trajectory, while providing insurmountable information for SpaceY to glean from.

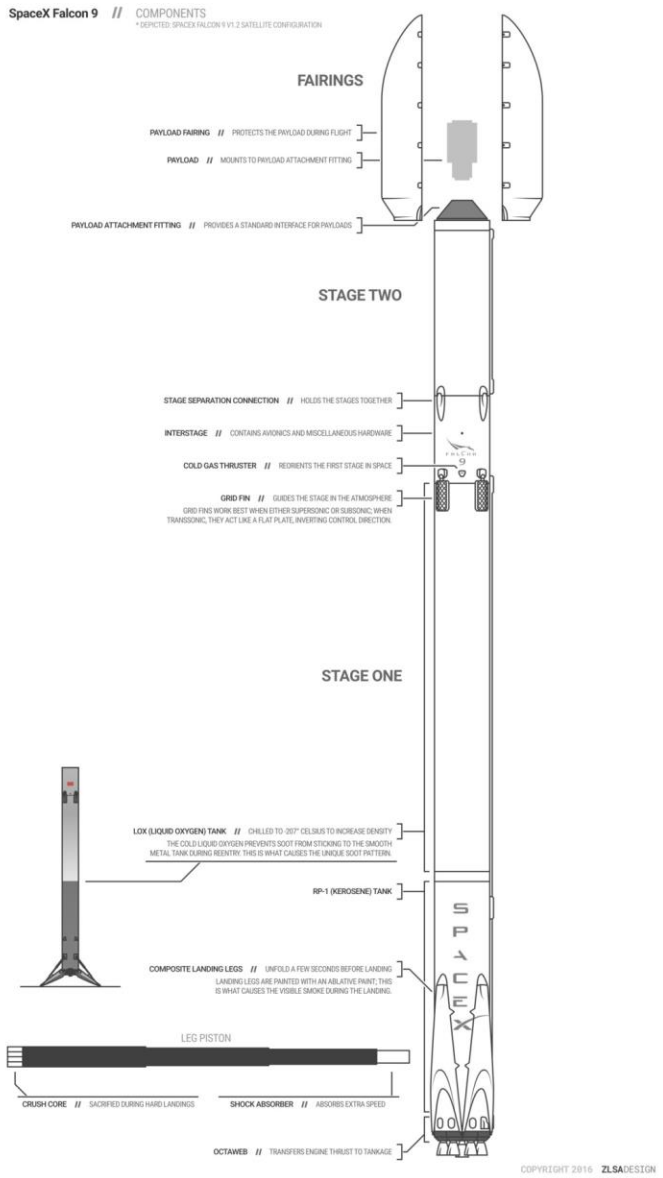
In order to refine and expand on this knowledge, a deeper dive into specific geo-spatial and geo-meteorological conditions, combined with real-time payload adjustments is recommended. Further feature engineering, fine tuning, and other model adjustments could improve predictive accuracy, thus enhancing budget predictions and launch strategies.

Introduction

Our emerging rocket ship startup, SpaceY, seeks to gain valuable insights from competitive industry leaders within the commercial space sector. In this analysis, historical launch data from SpaceX, a prominent pioneer in space exploration and technology, is thoroughly dissected to determine key factors to their business' success and their ability to predict launch outcomes. Specifically, data from the SpaceX Falcon 9 rocket, uniquely comprised of 2 stages, allowing the first stage to be reused if successful and significantly reducing overhead, is explored to provide insight to the most cost effective solution currently in market.

Multiple objectives are present to provide a complete overview of SpaceX's methods. Key variables such as payload dynamics, launch site location, date of launch, and so on, are considered to discern the weight of their impact on launch outcome. Predictive classification models are employed and analyzed to determine if this historical data can reliably predict successes. These findings can then be used to draw correlations between successful launches and operational overhead to pave a path for SpaceY's fiscal strategy.

Prior literature serves as a foundation for this analysis. Notably, Romanova, et al. (2013) employed statistical geospatial analyses to determine the optimal conditions and locales for rocket launches, showcasing the significance of geophysical and geometeorological variables on outcome success. Furthermore, Sforza (2016) quantified the influence of mechanical dynamics on launch outcomes, emphasizing the criticality of payload specifications. Moreover, the SpaceX Falcon 9 User's Guide (2009, 2021) highlighted the best performing mechanical specifications at the time(s) of release, demonstrating the robustness of iterative testing cycles, persistent R&D endeavors and their role achieving sustainable launch success rates.



Methodology

Data Collection

- SpaceX API
- Webscrape SpaceX Wiki

Data Wrangling

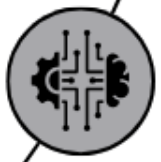
- Explore Dataset
- Create Training Labels

Exploratory Data Analysis

- SQL
- Data Visualizations

Interactive Visualizations

- Folium Map
- Plotly Dashboard



Predictive Modeling

- Logistic Regression
- Support Vector Machine
- Decision Tree
- K-Nearest Neighbors

Data Collection Methodology - API

To decipher SpaceX's historical launch data and harness it for predictive analytics, the following approach was used with the SpaceX API:

- **Data Collection**:
  - Sourced data from the SpaceX REST API (https://api.spacexdata.com/v4/launches/past), which serves as an authoritative reservoir of SpaceX's past launches. Via the API, requisitioned rocket launch data and stored in a response object.
    - **NOTE:** For consistency across student projects, a static response object was utilized. (https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json)
  - Employed **json_normalize()** method to convert the JSON results into a DataFrame to structure format for adept parsing.

```
spacex_url="https://api.spacexdata.com/v4/launches/past"

response = requests.get(spacex_url)
```

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

- **Data Subsetting & Refinement**:
  - Selected a focused subset of features from the comprehensive dataset, emphasizing on pivotal features rocket, payloads, launchpad, cores, flight_number, and date_utc.
  - Pruned rows showcasing multiple payloads or an excess of two rocket boosters. Transmuted **date_utc** column to a datetime format, accentuating solely on the date, and restricted all data to launches prior to November 13, 2020.

```
#Global variables
BoosterVersion = []
PayloadMass = []
Orbit = []
LaunchSite = []
Outcome = []
Flights = []
GridFins = []
Reused = []
Legs = []
LandingPad = []
Block = []
ReusedCount = []
Serial = []
Longitude = []
Latitude = []
```

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

- **Data Transformation**:
  - Cached API data into global variable lists through the custom-defined **get_Data()** functions.
  - Combined various columns into a dictionary which was subsequently used to create a new DataFrame.
  - Post-transformation, filtered data to spotlight only Falcon 9 launches. Additionally, reset the FlightNumber column.

- **Data Wrangling**:
  - Calculated and addressed NaN values. Replaced absent PayloadMass values by the column's average, leaving the only column brandishing NaN values was 'LandingPad' with a total of 26 entries.
  - Exported the data to a CSV format, primed for ensuing analytical efforts (**dataset_part_1a.csv**)

```
# Calculate the mean value of PayloadMass column
pm_mean = data_falcon9.PayloadMass.mean()
# Replace the np.nan values with its mean value
data_falcon9.loc[:,'PayloadMass'] = data_falcon9.PayloadMass.replace(np.nan, pm_mean)
```

Data Collection Methodology - Webscraping

To supplement the API-driven dataset, web scraping was utilized to extract key launch data from SpaceX's dedicated Wikipedia page as below:

- **Source Initialization**:
  - Earmarked the SpaceX wiki page, specifically the snapshot of the "List of Falcon 9 and Falcon Heavy launches" from Wikipedia for extraction. Loaded the designated static_url was with the aforementioned snapshot. Using the **.get()** method, the static_url was invoked, securing an HTTP response object.

- **Data Parsing Initiation**:
  - The HTTP response object was the base ingredient for crafting a BeautifulSoup object, a powerful tool for web data extraction. The focus of extraction was zeroed in on the third table of the page. Leveraging **.find_all('table')**, mapped the content from this table (index 2) to **first_launch_table**.

- **Header Extraction**:
  - Pinpoint the headers through an iteration over **<th>** elements, using the **.find_all('th')** function. Employed the custom **extract_column_from_header()** function to distill all non-empty column names, culminating in the list **column_names**.

- **Dictionary Formation**:
  - Using **dict.fromkeys(column_names)**, an empty dictionary, created **launch_dict**. This dictionary was then cleansed of the 'Date and time ( )' column and initialized with each key corresponding to an empty list.

- **Data Extraction**:
  - Perused all the launch tables on the wiki page to populate the **launch_dict** values with pertinent launch records extracted from the associated table rows.

- **DataFrame Creation & Export**:
  - Converted the populated **launch_dict** to a DataFrame. Post-conversion, saved the data in a CSV format for further analysis (**part_1b_webscraping.csv**).

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

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

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

```
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])

            # Date value
            # TODO: Append the date into launch_dict with key `Date`
            date = datatimelist[0].strip(',')
            launch_dict['Date'].append(date)
            #print(date)

            # Time value
            # TODO: Append the time into launch_dict with key `Time`
            time = datatimelist[1]
            launch_dict['Time'].append(time)
            #print(time)

            # Booster version
            # TODO: Append the bv into launch_dict with key `Version Booster`
            bv=booster_version(row[1])
            if not(bv):
                bv=row[1].a.string
            launch_dict['Version Booster'].append(bv)
            #print(bv)
```

Data Wrangling Methodology

For optimal data-driven decision-making, refining the raw data into a structured and meaningful dataset is crucial. With meticulous attention to detail, the data wrangling process was executed as explained below:

- **Data Initialization**:
  - Launched data from the previous API session and converted into a dataframe. Conducted an initial sweep was on the dataset, focusing on assessing the percentage of null values and identifying the variable datatypes to determine any values or features that may need conversion.

- **Value Analysis**:
  - Employed the **.value_counts()** method, to analyse key columns namely: LaunchSite, Orbit, and Outcome, helping decipher the frequency distribution of these variables.

- **Outcome Segregation**:
  - Based on the landing outcomes, eight distinct categories were identified including True ASDS, None None, and True RTLS, among others. Created a specific set, **bad_outcomes**, to encapsulate unsuccessful landing outcomes 'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', and 'None None'.

- **Outcome Classification**:
  - Delving into the **outcome** column, a binary list was generated. Elements received a value of '0' if the outcome was listed within the **bad_outcome** set, otherwise, they were tagged as '1'. This list, named **landing_class**, was designed to serve as the outcome variable for our impending classification algorithms. A new column, 'Class', was added into the dataframe in order to categorically distinguish between unsuccessful (0) and successful (1) first stage landings.

- **Success Metrics**:
  - Harnessing the 'Class' column, the success rate was calculated and found to be 67%.

- **Data Export**:
  - With the data now structured, enriched, and curated, it was transposed into a CSV format for next steps (**part_2_spacex_data_wrangling**).

```
df.isnull().sum()/df.shape[0]*100
```

```
FlightNumber      0.000000
Date              0.000000
BoosterVersion    0.000000
PayloadMass       0.000000
Orbit             0.000000
LaunchSite        0.000000
Outcome           0.000000
Flights           0.000000
GridFins          0.000000
Reused            0.000000
Legs              0.000000
LandingPad       28.888889
Block             0.000000
ReusedCount       0.000000
Serial            0.000000
Longitude         0.000000
Latitude          0.000000
dtype: float64
```

```
df.dtypes
```

```
FlightNumber      int64
Date             object
BoosterVersion   object
PayloadMass     float64
Orbit            object
LaunchSite       object
Outcome          object
Flights           int64
GridFins           bool
Reused             bool
Legs               bool
LandingPad       object
Block           float64
ReusedCount       int64
Serial           object
Longitude       float64
Latitude        float64
dtype: object
```

```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

```
df['Class']=landing_class
df[['Class']].head(8)
```

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = [0 if outcome in bad_outcomes else 1 for outcome in df['Outcome']]
```

EDA Methodology - SQL

SQL played an instrumental role in extracting intricate insights from the SpaceX dataset through the use of specified queries:

- **Database Initialization**:
  - Established a secure connection to the SQLite database. Integrated the SpaceX CSV data file into the database for further querying.

- **Key Queries Executed**:
  - **Unique Launch Sites**: Retrieved the distinct launch sites engaged in the space missions.
  - **Launch Sites Prefix 'CCA'**: Pinpointed five records of launch sites that prominently started with 'CCA'.
  - **NASA's Payload Mass**: Calculated the cumulative payload mass dispatched by boosters under NASA's CRS initiative.
  - **Average Payload of F9 v1.1**: Derived the mean payload mass for the booster version F9 v1.1.
  - **Milestone Landing**: Identified the date marking the first successful landing outcome on a ground pad.
  - **Booster Specifications**: Listed boosters that had successful drone ship landings and carried a payload mass ranging between 4000 to 6000.
  - **Mission Outcome Statistics**: Aggregated the count of both successful and failed mission outcomes.
  - **Maximum Payload Boosters**: Employing a subquery, spotlighted booster versions that bore the heaviest payload masses.
  - **2015 Analysis**: Filtered records to depict month names, drone ship landing outcomes, booster versions, and launch sites for the year 2015.
  - **Landing Outcome Rank**: Ranked landing outcomes, such as "Failure (drone ship)" and "Success (ground pad)", between the dates 2010-06-04 and 2017-03-20, ensuring a descending order for a prioritized view.

```
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5
```

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD_MASS_KG FROM SPACEXTABLE WHERE "Booster_Version" LIKE "F9 v1.1%"
```

```
%sql SELECT MIN(Date) AS FIRST_LANDING_SUCCESS FROM SPACEXTABLE WHERE "Landing_Outcome" = "Success (ground pad)"
```

```
%sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

```
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE) GROUP BY Booster_Version
```

EDA Methodology – Data Visualization

To gain a deeper understanding of the underlying trends and patterns within the SpaceX dataset, the libraries Seaborn and Plotly were used to create a series of visualizations to shed light on the relationship between various factors and their influence on landing success:

- **Data Preparation**:
  - Imported the previously wrangled SpaceX data into a pandas dataframe, facilitating a seamless data manipulation and visualization workflow.

- **Visualization Generation**:
  - **FlightNumber vs. PayloadMass**: A scatterplot visualizing the relationship between flight number and payload mass, further color-coded by launch outcome.
  - **FlightNumber vs. LaunchSite**: A scatterplot showcasing how different launch sites performed across various flight numbers, differentiated by outcome.
  - **LaunchSite vs. PayloadMass**: The interrelation between the launch sites and the carried payload masses, with an overlay of launch outcomes in scatterplot form.
  - **Landing Success by Orbit**: A bar chart illustrating the success rates of landings across different orbits.
  - **FlightNumber vs. Orbit**: A scatterplot of flights across different orbits and their outcomes.
  - **PayloadMass vs. Orbit**: Highlighted how varying payload masses fared across different orbits, with a focus on their landing success in a scatterplot.
  - **Yearly Success Trend**: A line graph mapping the average success rate over the years, providing a temporal view of SpaceX's performance trajectory.

- **Feature Engineering**:
  - Performed feature selection of vital columns 'FlightNumber', 'PayloadMass', 'Orbit', 'LaunchSite', 'Flights', 'GridFins', 'Reused', 'Legs', 'LandingPad', 'Block', 'ReusedCount', and 'Serial' to serve as features for subsequent analytical steps. Utilized **pd.get_dummies** to perform OneHotEncoding on categorical features Orbits, LaunchSite, LandingPad, and Serial. These encoded features were then amalgamated with the other aforementioned features. Data homogeneity was ensured by converting all values within the dataframe to the float64 data type, after which the data was exported to a CSV for future usage (**dataset_part_3b.csv**).

```
sns.catplot(x="FlightNumber", y="PayloadMass", hue="Class", data=df, aspect = 5)
plt.title("Successful Landings per Payload Mass by Flight Number", fontsize=25)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```

```
orbit_success_rate = df.groupby('Orbit')['Class'].mean()
sns.barplot(x=orbit_success_rate.index, y=orbit_success_rate.values)
plt.title("Success Rate by Orbit", fontsize=20)
plt.xlabel("Orbit", fontsize=15)
plt.ylabel("Success Rate", fontsize=15)
plt.show()
```

```
features_one_hot = pd.get_dummies(features[['Orbit', 'LaunchSite', 'LandingPad','Serial']])
features = features.drop(['Orbit', 'LaunchSite', 'LandingPad', 'Serial'], axis=1)
features_one_hot = features_one_hot.join(features)
```

Interactive Visual Analytics Methodology – Folium Map

To discern potential geographical patterns tied to launch sites and the impact of their locations on successful launches, Folium was employed, a robust Python library that facilitates interactive mapping:

- **Data Ingestion**:
  - Loaded the SpaceX CSV data into a pandas dataframe, ensuring readiness for mapping tasks. Pinpointed the geographical coordinates of each distinct launch site from dataframe, a prerequisite for plotting in Folium.
- **Base Map Creation**:
  - Generated a Folium map with its focal point centered on NASA, laying the groundwork for subsequent overlays and visualizations.
- **Visual Depictions on Map**:
  - Utilized **`folium.Circle`** to fashion a circular representation for the Johnson Space Center and appended a popup with its moniker. A distinct icon, courtesy of **`folium.map.Marker`**, showcasing its abbreviated name was added to the map.
  - Iterated through the **`launch_sites_df`** dataframe. For each site, a circle was plotted, supplemented with a popup showcasing the site's designation. This was achieved using **`folium.Circle`** and **`folium.map.Marker`**.
  - Added markers symbolizing individual launch records. Color-coding was implemented – green markers signifying successful launches, and red indicating filed. These were organized using **`MarkerCluster()`** for better visualization.
- **Advanced Mapping Features**:
  - Incorporated the **`MousePosition`** feature to facilitate real-time retrieval of coordinates for any given point on the map. This proved instrumental in ascertaining coordinates for nearby geographical entities, including coastlines, highways, railroads, and cities in proximity to launch sites.
  - Leveraged **`folium.Marker`** to position markers at the previously identified coordinates. Furthermore, **`folium.PolyLine`** was employed to delineate lines connecting the launch sites to these geographical points. These lines also denoted the distance in kilometers between the entities to assess any present patterns between launch site location and proximity to imperative landmarks.

```
# 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 site name as a popup label
for i in range(len(launch_sites_df)):
    circle = folium.Circle([launch_sites_df.loc[i,'Lat'], launch_sites_df.loc[i,'Long']], radius=1000, color='#d35400', fill=True).add_child(folium.
    marker = folium.map.Marker([launch_sites_df.loc[i,'Lat'], launch_sites_df.loc[i,'Long']], icon=DivIcon(icon_size=(20,20),icon_anchor=(0,0), html
    site_map.add_child(circle)
    site_map.add_child(marker)
site_map
```

```
distance_marker = folium.Marker(
    [34.6372, -120.6232],
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % "{:10.2f} KM".format(distance_railway)
        )
    ).add_to(site_map)

distance_marker = folium.Marker(
    [34.6885, -120.4559],
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % "{:10.2f} KM".format(distance_highway)
        )
    ).add_to(site_map)

distance_marker = folium.Marker(
    [34.6386, -120.4581],
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % "{:10.2f} KM".format(distance_city), #
        )
    ).add_to(site_map)
```

```
# Add Mouse Position to get the coordinate (Lat, Long) for a mouse over on the map
formatter = "function(num) {return L.Util.formatNum(num, 5);};"
mouse_position = MousePosition(
    position='topright',
    separator=' Long: ',
    empty_string='NaN',
    lng_first=False,
    num_digits=20,
    prefix='Lat:',
    lat_formatter=formatter,
    lng_formatter=formatter,
)

site_map.add_child(mouse_position)
site_map
```

```
coordinates1 = [[34.6328, -120.6108], [34.6372, -120.6232]]
lines1=folium.PolyLine(locations=coordinates1, weight=1)
site_map.add_child(lines1)

coordinates2 = [[34.6328, -120.6108], [34.6885, -120.4559]]
lines2=folium.PolyLine(locations=coordinates2, weight=1)
site_map.add_child(lines2)

coordinates3 = [[34.6328, -120.6108], [34.6386, -120.4581]]
lines3=folium.PolyLine(locations=coordinates3, weight=1)
site_map.add_child(lines3)
```

Interactive Visual Analytics Methodology – Plotly Dash

To facilitate an immersive, visual analysis of the SpaceX data, Plotly Dash, a Python framework recognized for crafting rich, web-based applications, was employed as follows:

- **Data Preparation**:
  - Loaded the wrangled SpaceX CSV file into a pandas dataframe to have a structured format for visualization purposes.
- **Dashboard Layout Configuration**:
  - Established the overarching theme of the application by setting the title as 'SpaceX Launch Records Dashboard'. Implemented a dropdown menu, empowering users to effortlessly choose among different launch site analytics (all sites or individual site).
- **Visual Elements**:
  - **Pie Chart Depiction:** Situated beneath the dropdown menu, the pie chart dynamically represents:
    - The cumulative successful launch counts across all sites (when 'all' is selected).
    - A juxtaposition of successful versus unsuccessful launches specific to an individual site selected from the dropdown menu.
  - **Payload Range Selector**: Incorporated **`dcc.RangeSlider()`** to allow users to stipulate a payload mass range in kilograms.
  - **Scatter Chart Projection**: Showcased the correlation between payload mass and launch success. This visual is adaptive, changing its representation contingent on both the dropdown menu's launch site selection and the defined payload range.
- **Interactivity through Callbacks**:
  - **Pie Chart Callback**: A callback function was established to glean input from the dropdown menu, which in turn influences the pie chart's output. The **`get_pie_chart`** function was designed to generate the pie chart corresponding to the site or sites pinpointed via the dropdown.
  - **Scatter Plot Callback**: This callback function hones in on the dropdown menu and the payload mass slider, dynamically altering the scatterplot's output. The **`get_scatter_plot`** function is then invoked to mold the scatter chart, contingent on the chosen launch site(s) and payload mass.
- **Dashboard Activation**:
  - Activated the application, culminating in the unveiling of an insightful and interactive analytical dashboard.

```
# Create an app layout
app.layout = html.Div(children=[html.H1('SpaceX Launch Records Dashboard',
                                        style={'textAlign': 'center', 'color': '#503D36',
                                               'font-size': 40}),
                                # TASK 1: Add a dropdown list to enable Launch Site selection
                                # The default select value is for ALL sites
                                # dcc.Dropdown(id='site-dropdown',...)
                                html.Div(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',
                                    searchable=True,
                                    ),
                                    style={'width':'50%', 'margin': '0 auto', 'padding':'3px', 'font-size': '20px','textAlign': 'center'}
                                ),
                                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', 2500:'2500', 5000:'5000',7500:'7500', 10000:'10000'},
                value=[min_payload, max_payload]),

# TASK 4: Add a scatter chart to show the correlation between payload and launch success
html.Div(dcc.Graph(id='success-payload-scatter-chart')),
])
```

```
# TASK 2:
# Add a callback function for `site-dropdown` as input, `success-pie-chart` as output
# Function decorator to specify function input and output
@app.callback(Output(component_id='success-pie-chart', component_property='figure'),
              Input(component_id='site-dropdown', component_property='value'))

def get_pie_chart(entered_site):
    if entered_site == 'ALL':
        fig = px.pie(spacex_df, values='Class',
        names='Launch Site',
        title='Total Success Launches by Site')
        fig.update_traces(textfont_color='white')
        fig.update_layout(hoverlabel=dict(font_color='white', bordercolor='black'))
        return fig
    else:
        # return the outcomes piechart for a selected site
        filtered_df = spacex_df[spacex_df['Launch Site'] == entered_site].groupby('Class').size().reset_index(name='Counts')
        color_map = {0: '#eb553f', 1: '#636efa'}
        fig = px.pie(filtered_df, values='Counts',
        names=['Failed' if i == 0 else 'Successful' for i in filtered_df.groupby(['Class']).size().index],
        title='Total Success Launches for Site {}'.format(entered_site),
        color='Class',
        color_discrete_map=color_map)
        fig.update_traces(textfont_color='white')
        fig.update_layout(hoverlabel=dict(font_color='white', bordercolor='black'))
        return fig
```

Predictive Analytics Methodology

Through the power of machine learning, utilize machine learning models to predict whether the initial stage of a SpaceX rocket launch will achieve a successful landing, factoring in various determinants:

Data Initialization:

- Imported the data from part 2 (unencoded features and target column 'Class') into a dataframe **data**. Concurrently, the data from part 3, housing the OneHotEncoded features intended for predictions, was loaded into a dataframe **X**.

•Data Transformation:

- Leveraged **.to_numpy()** to mold the 'Class' column from **data** into a NumPy array, assigning it to target variable **Y**. Adopted **preprocessing.StandardScaler()** to normalize the data within **X**. The transformed dataset was reassigned to **X** using **.fit_transform()**.

•Data Partitioning:

- Invoked **train_test_split** on **X** and **Y** with a test set constituting 20% of the data (**test_size=0.2**) and a consistent seed (**random_state=2**). The outcome was apportioned into **X_train**, **X_test**, **Y_train**, and **Y_test**.

•Machine Learning Modeling:

- Embarked on training the data using multiple classification algorithms. Each model employed **GridSearchCV** to discern the optimal parameters, sourcing these from a dedicated dictionary named **parameters** (unique to each algorithm). The selected algorithms encompass:
  - **Logistic Regression**
  - **Support Vector Machines (SVM)**
  - **Decision Tree Classifier**
  - **K-Nearest Neighbors (KNN)**

•Model Evaluation:

- Upon determining the most favorable parameters for each model, gauged the model's prowess by computing accuracy via the **.score()** method. Illustrated the confusion matrix for each algorithm to evaluate the true positive, true negative, false positive, and false negative values.

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

```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}# l1 lasso l2 ridge
lr=LogisticRegression()
logreg_cv = GridSearchCV(lr, parameters, cv=10)
logreg_cv.fit(X_train, Y_train)
```

```
parameters = {'kernel':('linear', 'rbf','poly', 'sigmoid'),
              'C': np.logspace(-3, 3, 5),
              'gamma':np.logspace(-3, 3, 5)}
svm = SVC()
```

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

```
tree = DecisionTreeClassifier()
```

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

```
parameters = {'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
              'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
              'p': [1,2]}

KNN = KNeighborsClassifier()
```

```
knn_cv = GridSearchCV(KNN, parameters, cv=10)
knn_cv.fit(X_train, Y_train)
```

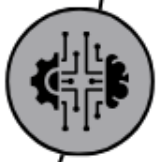
Results

Exploratory Data Analysis

- SQL
- Data Visualizations

Interactive Visualizations

- Folium Map
- Plotly Dash Dashboard



Predictive Analysis

- Logistic Regression
- Support Vector Machine
- Decision Tree
- K-Nearest Neighbors

Exploratory Data Analysis Results

The EDA aimed to identify patterns, relationships, anomalies, and trends within the SpaceX data to understand factors influencing launch success. Data was analyzed using SQL queries to closely inspect important feature values and summary statistics. The data was further explored with visualizations to understand any relationships between relevant features and their impact on success rate prior to feature engineering and predictive modeling.

EDA with SQL Results

Initial data exploration with SQL revealed some interesting findings. Four launch sites are present: CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, and CCAFS SLC-40. A closer inspection reveals details of launch records from sites starting with "CCA". We can see the total payload mass carried by NASA (CRS) boosters is 45596 kg, while the average payload mass carried by Falcon 9 Booster v1.1 is only 2534.67 kg. This is a significant difference that likely impacts the reusability and thus the cost of Falcon 9 launches. Additionally, the first successful landing outcome in ground pad was found to be December 22, 2015, marking a significant turning point in the data.

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

Unique Launch Sites

TOTAL_PAYLOAD_MASS_KG
45596

Total Payload Carried by Boosters Launched by NASA (CRS)

AVG_PAYLOAD_MASS_KG
2534.6666666666665

Average Payload Mass Carried by Booster F9 v1.1

FIRST_LANDING_SUCCESS
2015-12-22

First Successful Landing Outcome

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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 (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
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Five Launches from Site Names with CCA

EDA with SQL results

Here, we further analyze the effect of booster version, payload mass, and successful outcomes. We can see that 4 boosters, all FT class, had successful drone ship landings with a payload range between 4000 and 6000 kg. Additionally, 12 different boosters have carried the maximum payload mass of 15600kg, all within the B5 category. Finally, we analyze the landing outcomes by assessing the total failed and successful outcomes within the dataset, demonstrating a high overall success at 61 versus 10 failed outcomes. A more granular view of April 2010 to March 2017 shows that most landings, 10, had no attempt, 10 had success, and others had either a controlled or unexpected failure. This indicates that many of the successful outcomes occurred after April 2017.

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

Boosters with Success in Drone Ship and Payload Mass of 4000-6000Kg

Successful	Failed
61	10

Total Successful and Failed Mission Outcomes

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

Booster Versions that have Carried Maximum Payload Mass

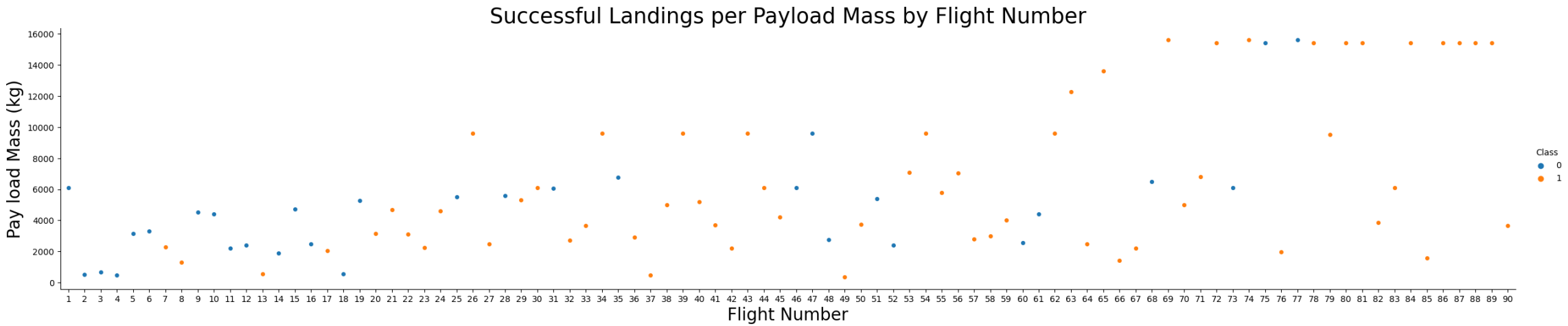
Landing_Outcome	Count
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

Landing Outcome Counts in Descending Order, 2010-06-04 to 2017-03-20

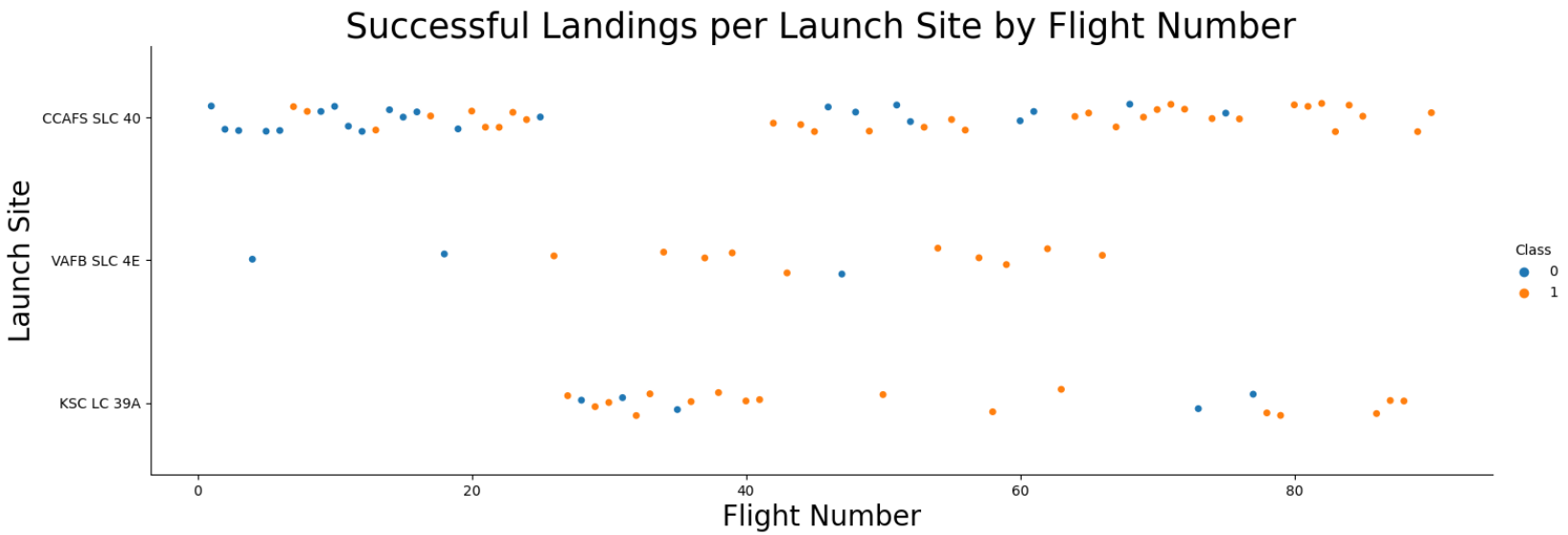
Month	Landing_Outcome	Booster_Version	Launch_Site
10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Month, Booster Version, and Launch Site for Failed Outcome in Drone Ship 2015

EDA with Visualization Results

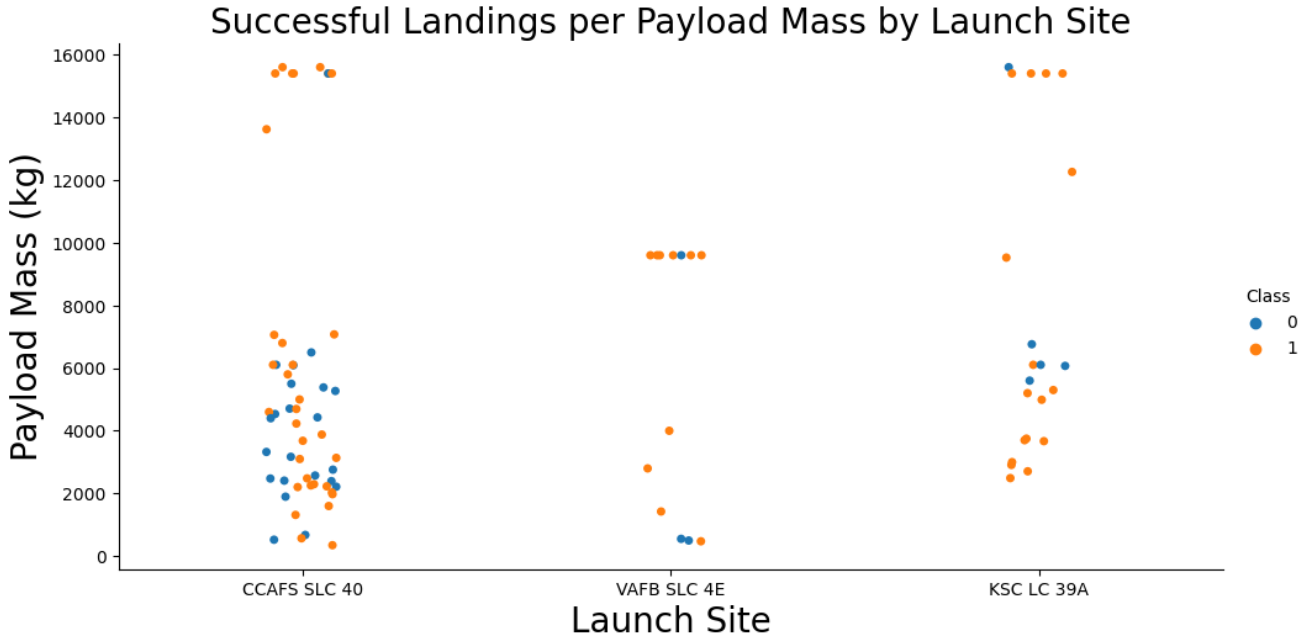


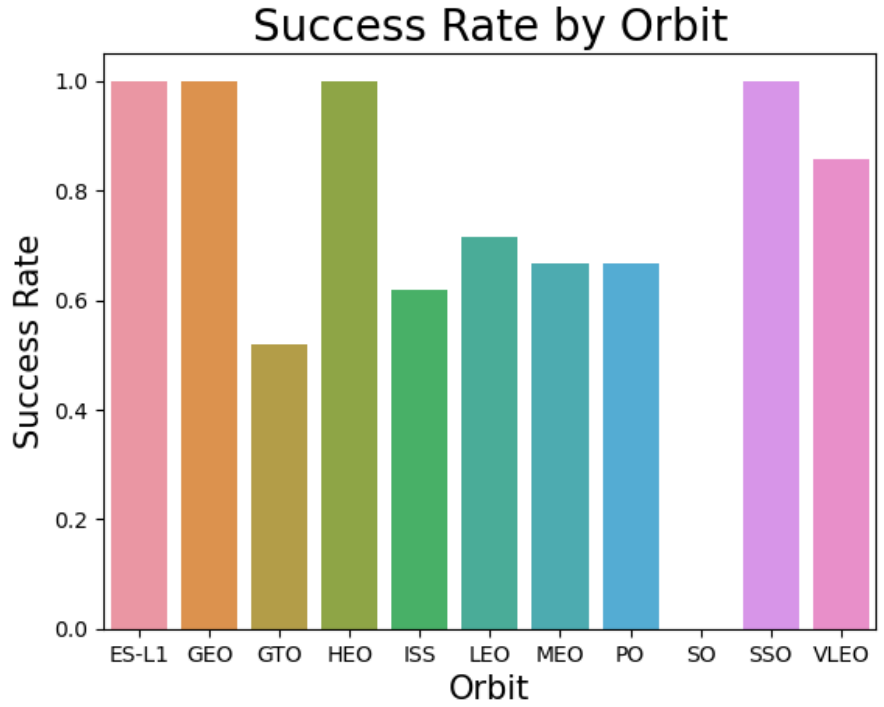
FlightNumber vs. PayloadMass: Success rates vary greatly across launch sites. CCAFS LC-40 is 60% while KSC LC-39A and VAFB SLC 4E are 77%. Overall, success rate appears to increase as both flight number and payload mass increase, suggesting a positive linear correlation between these variables.



FlightNumber vs. LaunchSite: Overall, the success rates at each launch site increased as flight number increased. This is most apparent at launch site CCAFS SLC-40, which hosted the majority of the early flights that landed unsuccessfully, and remains the most utilized launch site to throughout the dataset. There was a brief pause between flight numbers 25-40 where launch site KSC LC-39A was used for most flights, during which there was a strong shift from unsuccessful landings to successful landings before returning to CCAFS SLC-40. This demonstrates an upwards trend toward successful launches as the overall number of rockets launched increases. This is likely attributed to the iterative learning process and application utilized by SpaceX to yield more successful landings (SpaceX Falcon User's Guide (2021)).

EDA with Visualization Results

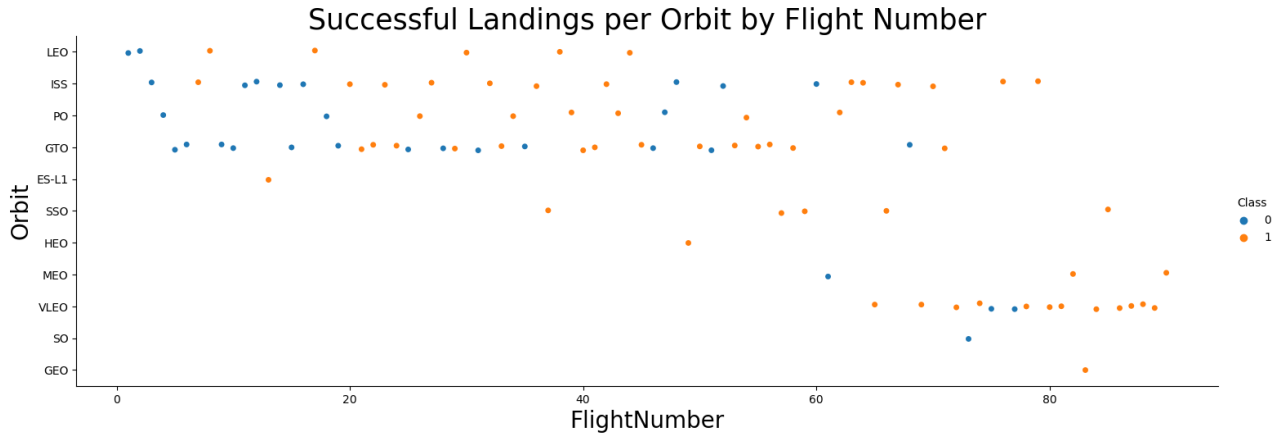




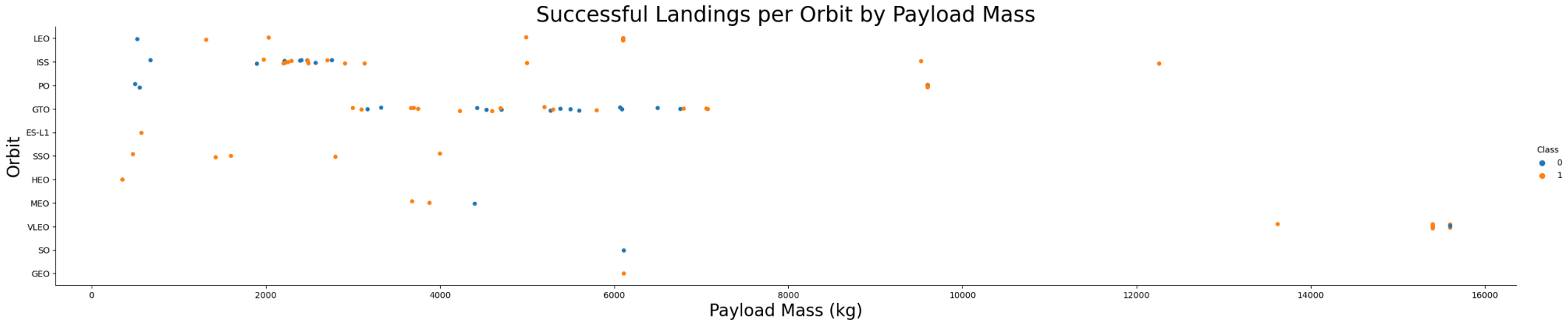
LaunchSite vs. PayloadMass: While the distribution of payload mass by launch site varies greatly, no rockets were launched for heavy payloads (>10000kg) at VAFB-SLC launch site. Similarly, the CCAFS SLC 40 site is not used for mid-sized payloads between 8000-13000kg. However, this site was used most frequently for smaller payloads of 100-7500kg, resulting in a greater percentage of failed launches overall compared to other sites. This suggests that payload mass plays an integral role in launch outcome, as mentioned by Sforza (2016) previously in the literature review.

Success Rate by Orbit: Orbits ES-L1, GEO, HEO, and SSO have the highest average success rate at 100%. Orbit VLEO is the next highest at roughly 85%. Most other orbits result in about 50% success on average. However, orbit SO has a 0% success rate. This extreme outlying value will need to be considered with other variables to determine the impact of orbit on success in this case. Otherwise, orbit appears to be a potentially important factor in success rate.

EDA with Visualization Results

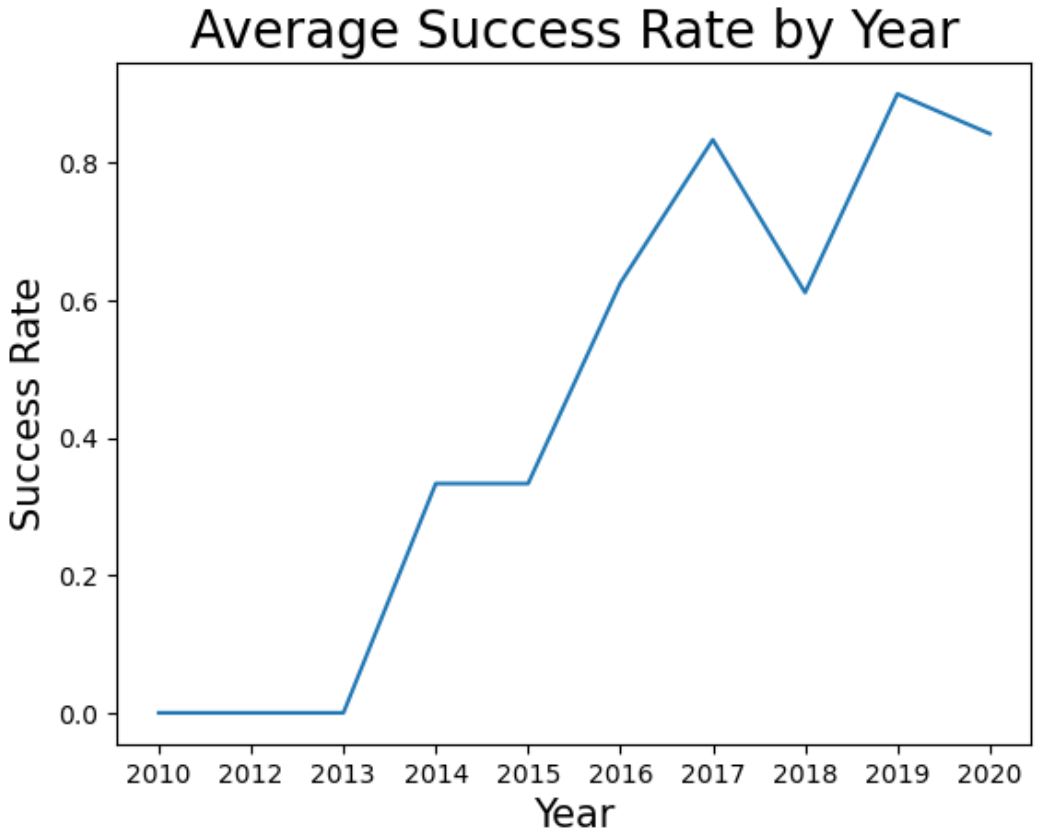


FlightNumber vs. Orbit: In the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no significant relationship between flight number when in GTO, ISS, or PO orbit. Most of the launches are concentrated within the aforementioned orbits, seeing as they have been active since the first flights. The other orbits have hosted drastically less flights overall, the majority of which are from flight number 60 and on. As such, the majority of these launches were successful. Orbit VLEO has been used the most for the newer launches.



PayloadMass vs. Orbit: With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. However for GTO we cannot distinguish this well as both positive landing rate and negative landing (unsuccessful mission) are both there here. There is not a large variance of payload mass for other orbits, therefore it is inconclusive if there is any specific relationship between the mass and success rates for these orbits.

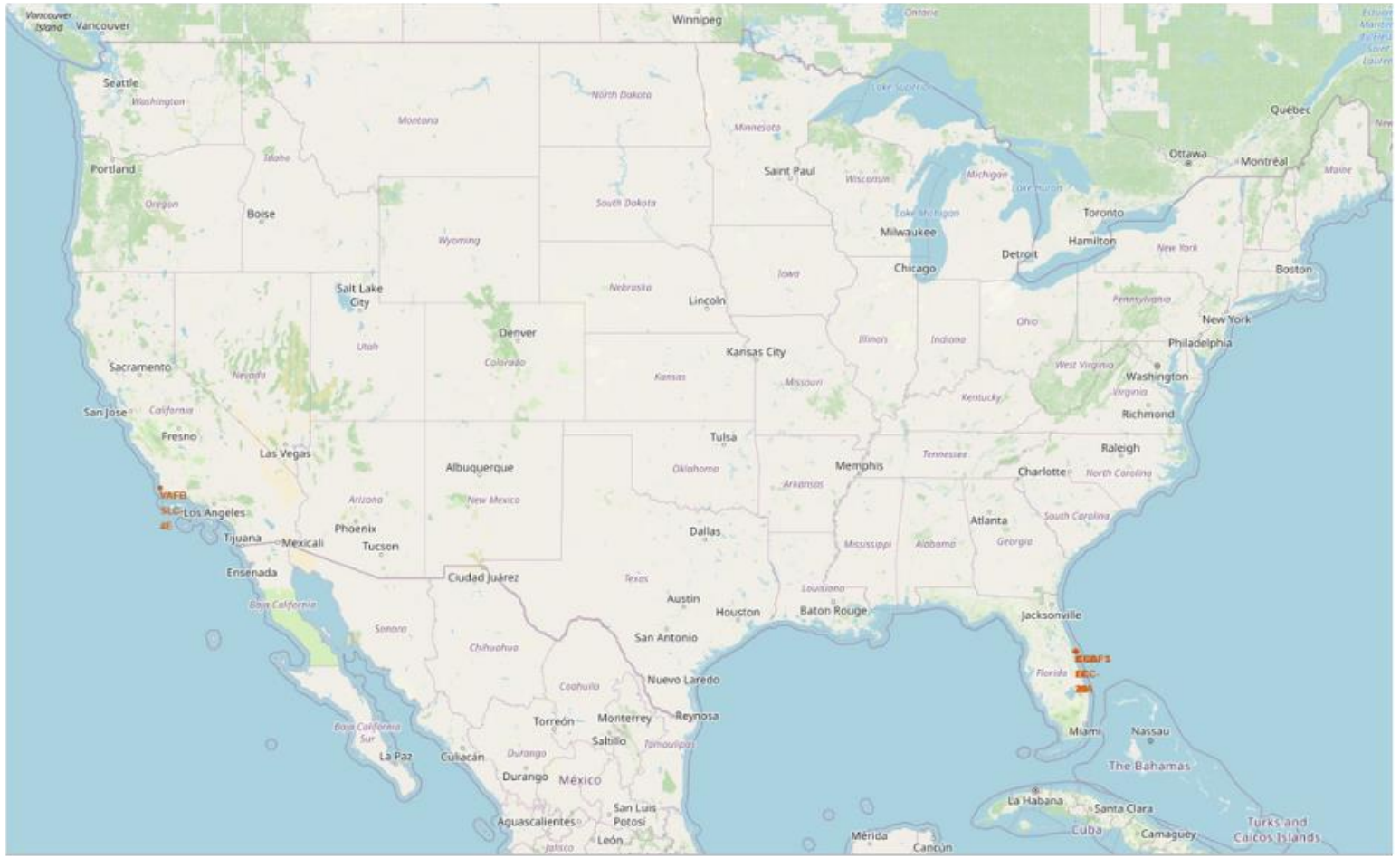
EDA with Visualization Results



Yearly Success Trend: The average success rate started to increase in 2013 and has done so steadily ever since, with drops seen in 2018 and 2020 (from the data currently evaluated.) This strongly suggests SpaceX has been able to glean useful insights from prior launches to increase the success rates over time.

Interactive Map with Folium Results

Map with Markers for SpaceX Launch Sites

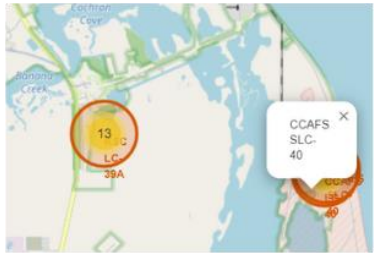


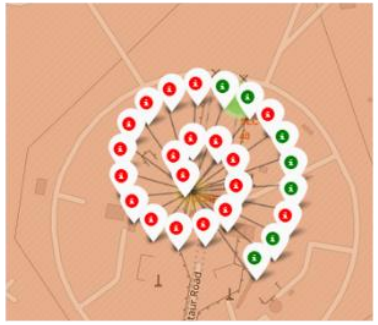
After mapping the coordinates of the 4 SpaceX launch sites, notice the sites are all located on the coastline: one in Southern California and the other 3 within close proximity to one another in Florida. Seeing as NASA's Johnson Space Center is located near the coast as well in Houston, it can be inferred that a coastal location is imperative for launch. Not only that, but since each location is specifically on the coastline in the Southern United States, closer to the equator, it can be implied that has an impact on launches as well.

This is further supported by the evidence in our literature review, Romanova et al. (2013) which indicates an advantage to eastern coastal launch sites near the equator due to the significant increase to propulsion during take-off.

Interactive Map with Folium Results

Map with MarkerCluster to Depict Launch Outcomes Per Site



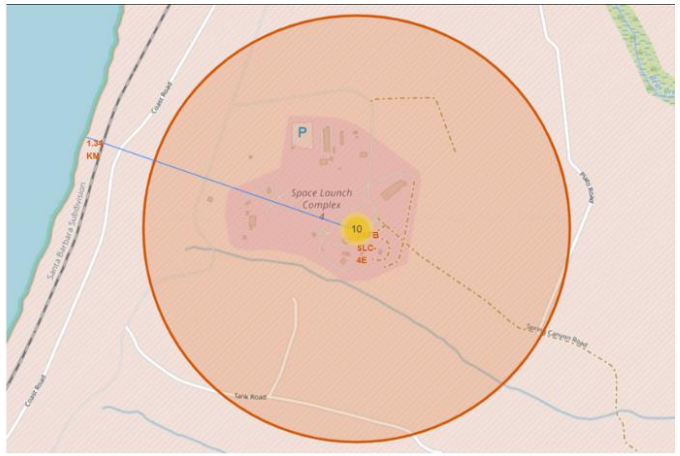


At first glance, it can be observed that the launch site VAFB SLC-4E in California has hosted significantly less launches than Florida, with only 10 launches recorded. Of those, 40% were successful. Florida is home to 3 launch sites within close proximity to one another: CCAFS LC-40, CCAFS SLC-40, and KSC LC-39A. 46 total launches departed from Florida, with 13 at KSC LC-39A and a 77% success rate, 26 at CCAFS LC-40 with a success rate of 27%, and 7 at CCAFS SLC-40 with 43% success. Based on these findings, we can see that the site with the highest success rate is KSC LC-39A and the lowest is CCAFS LC-40.

Interactive Map with Folium Results

Mouse position object to display latitude and longitude. This will help record relevant coordinates.

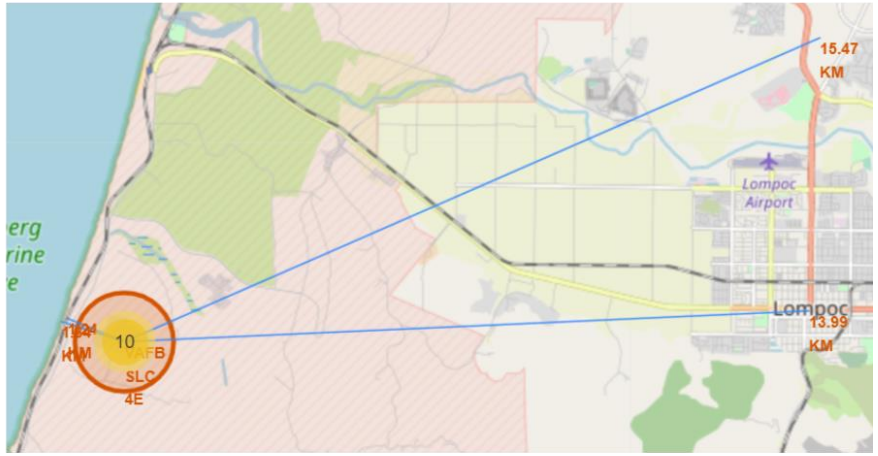
A MousePosition Object was added to the map to retrieve the coordinates of mouse over positions. This was then utilized to record the latitude and longitude of significant landmarks near launch sites. These landmarks include coastline, railroad, highway, and city. By finding the proximity of major civilian transit and settlements, we can establish trends in the best locations for launch site.



Distance Marker to Closest coastline with Polyline to Launchsite VAFB SLC-4E

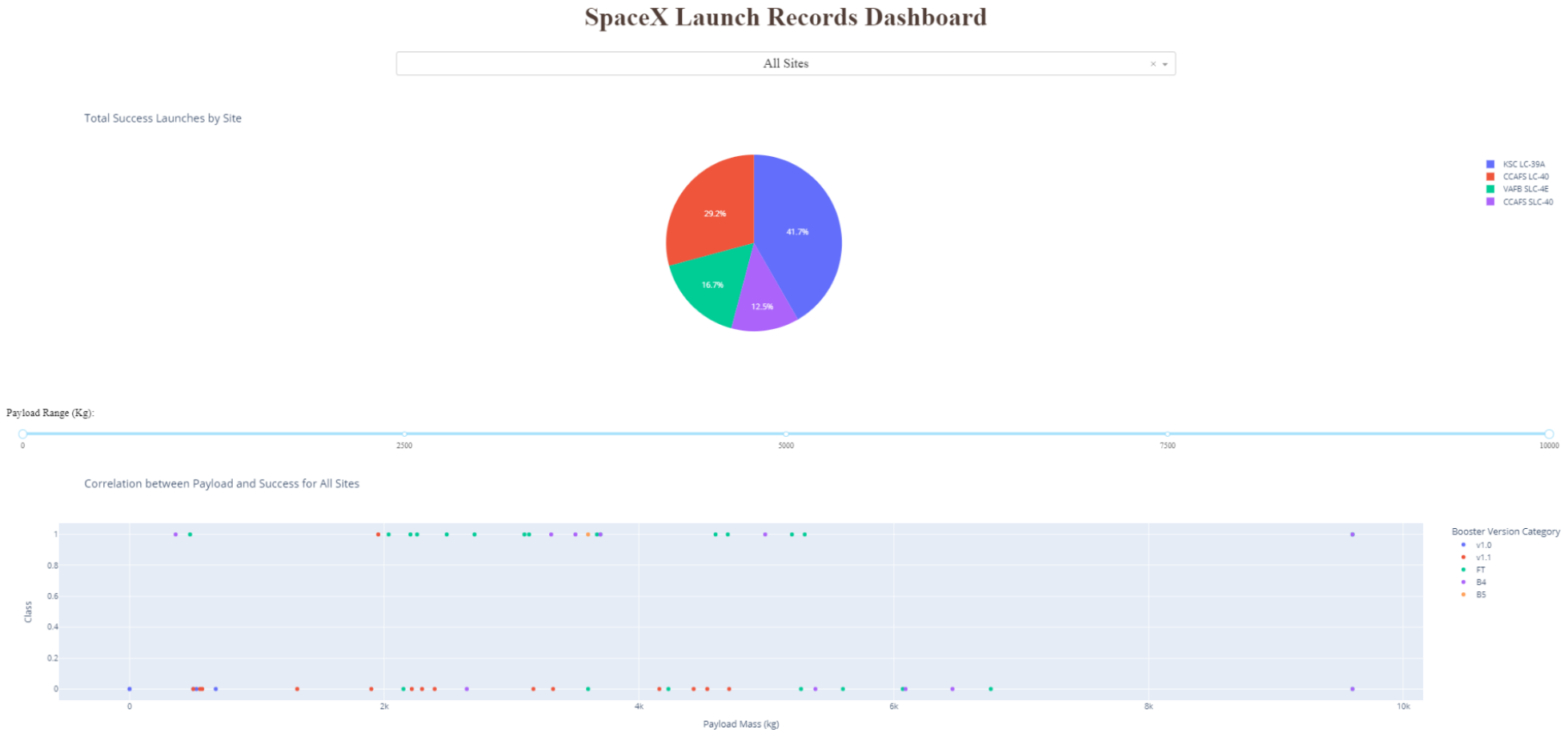
After finding the distance to pertinent locations, it can be observed that launch sites have closer proximity to some locations than others. The launch site is seen to be very close to the coastline, at 1.34 km, as well as the railroad at 1.24 km. It appears beneficial based on site patterns to be close to the coastline, likely to minimize risk of debris fall in civilian areas. Close proximity to the railway system is also beneficial from an infrastructure standpoint to transport parts and fuel for the rockets efficiently.

On the other hand, it can be seen that major highways and cities are further from launch sites in comparison. The nearest highway, Cabrillo Highway, is 15.47 Km away, and the nearest city, Lompoc, is 13.99 km away. There could be several reasons for this, but again most likely due to safety concerns for the general population.



Distance Marker to Closest Highway, City, and Railroad with Polyline to Launchsite VAFB SLC-4E

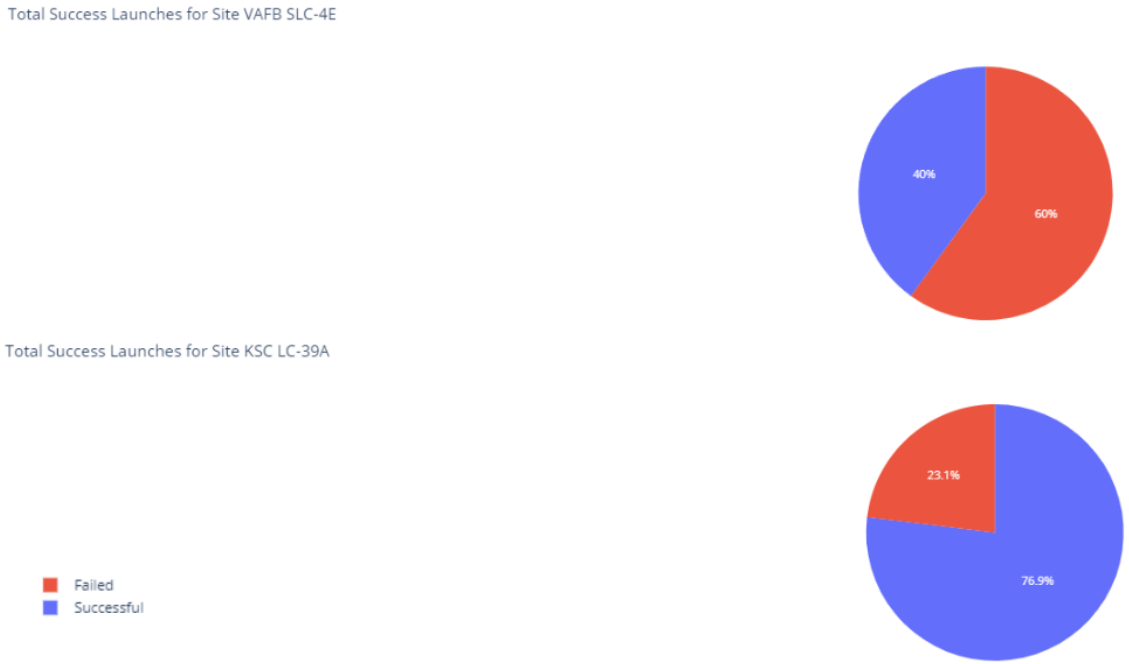
Plotly Dash Dashboard Results



A visual analytical dashboard was built using plotly Dash as seen above. This allows us to view the distribution of success by launch site, payload mass, and booster version in a live interactive format.

As shown in the pie chart above with all sites selected, the SpaceX's total successful launches vary by launch site. In this case, site KSC LC-39A has the highest successful launches at 41.7%, while CCAFS SLC-40 has the lowest at 12.5%.

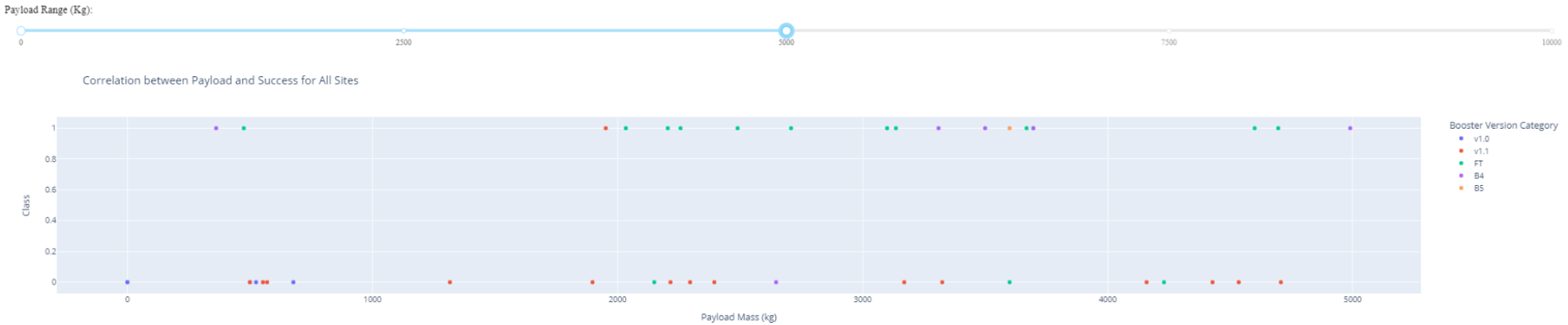
Plotly Dash Dashboard Results



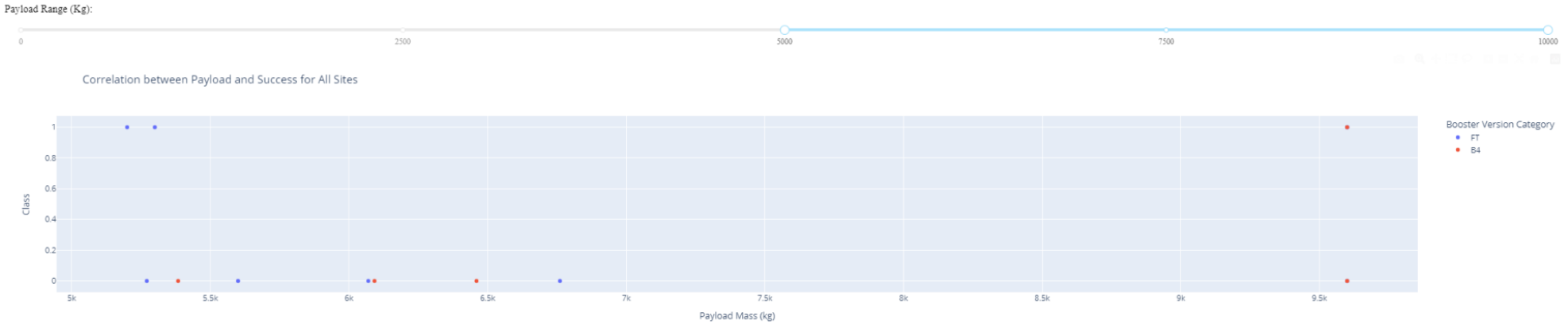
The dashboard can be filtered with the dropdown menu, which allows the selection of a specific site as shown in the pie charts above. These charts display the success rate at each site.

Based on these results, KSC LC-39A has the highest launch success rate of 76.9%. The lowest is CCAFS LC-40 with 26.9% success.

Plotly Dash Dashboard Results



Payload Range 0-5000 Kg

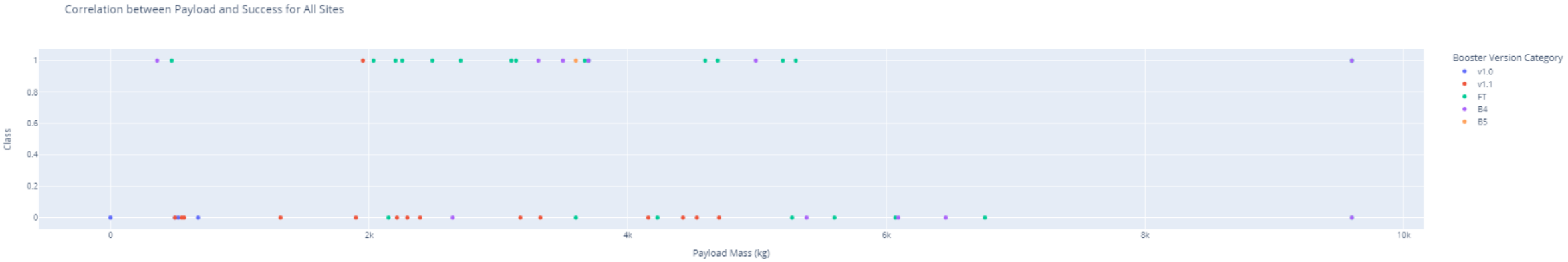


Payload Range 5000-10000 Kg

The lower portion of the dashboard displays a scatterplot of success by payload range and booster category. This can be filtered by launch site and to select a specific payload range.

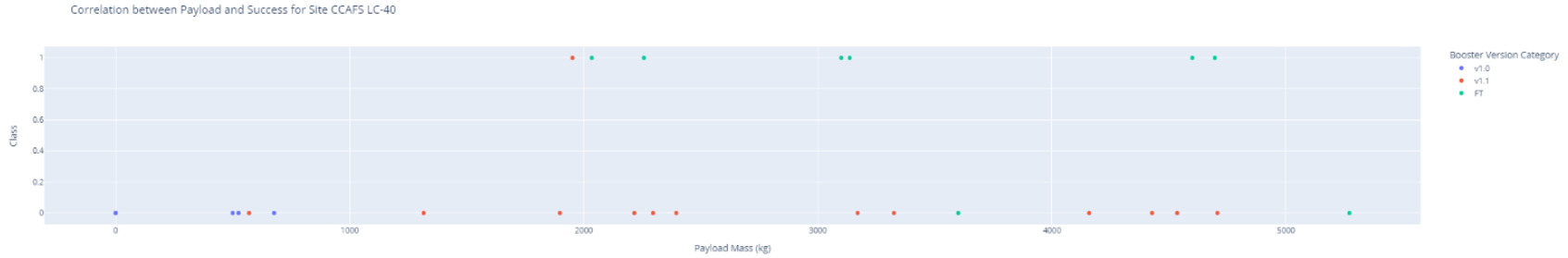
This scatterplot reveals that the payload range of 2000-4000 Kg has the highest rate of successful launches with 60% success. Payload range 6000-10000 has the lowest overall success at only 20%, however only 5 launches used payloads over 6000. In comparison, the range of 0-2000 kg has a success rate of 27% and 11 launches. This seems to show that both incredibly small and large payloads are not ideal for successful outcomes.

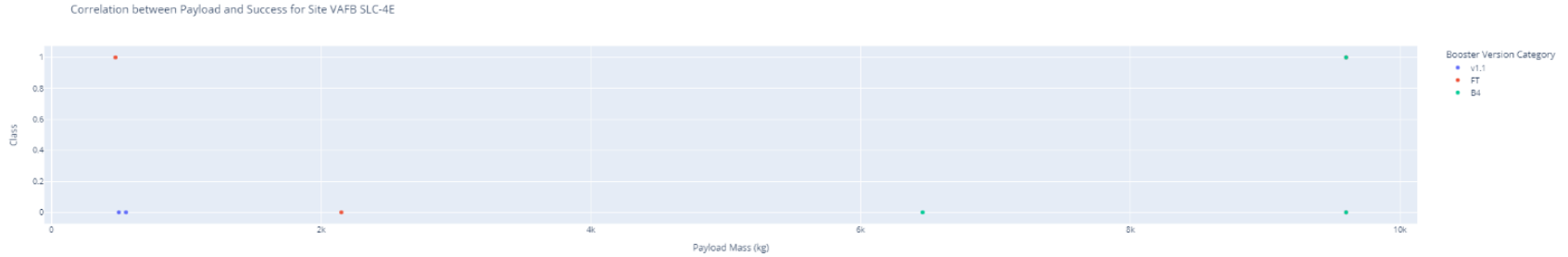
Plotly Dash Dashboard Results

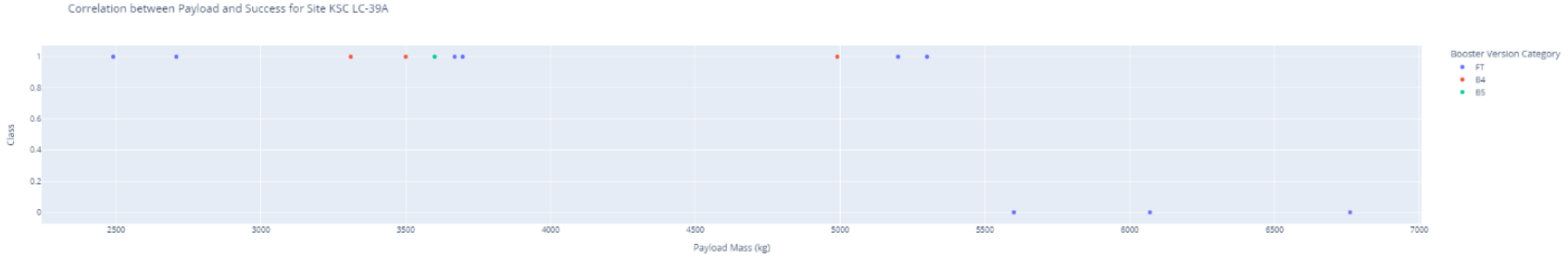


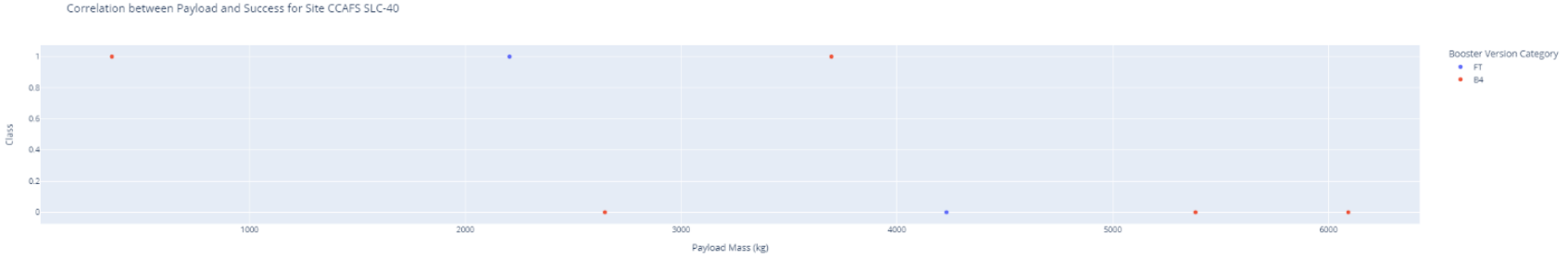
This view can evaluate for a clear view on success rates by booster version. Of the 5 booster versions, B5 has the highest success rate at 100%, but has only been used for one launch. In comparison, B4 has the next highest overall success rate at 83% from 6 launches, followed by FT with 14 launches and a 79% success rate. The least successful booster version is v1.1 with 20 launches and only 11% success.

Plotly Dash Dashboard Results









Here, we can see the booster version and success by payload for each launch site. This offers further insight to the specific payload ranges and boosters that are utilized at each site, and if this has any impact on the site's success rate. Certain sites may also be better equipped to handle specific boosters or payloads based on location, site layout, etc.

From the charts, site CCAFS LC-40 has the most failed launches overall and there does not appear to be a pattern amongst payload mass or booster version as many variation have been utilized. Site KSC LC-39A has the most successful launches, and this appears to be attributed to most launches having a payload range within 2500-5500 kg.

Predictive Analysis Results

Our aim was to predict the likelihood of a successful landing of the SpaceX rocket's first stage based on a variety of features. Four distinct algorithms were employed in this analysis: logistic regression, SVM, decision trees, and k-nearest neighbors. GridSearchCV was implemented to find the best hyperparameters for each model before testing. The resulting best parameters and train data accuracy scores are shown below. As we can see, while the accuracy scores are fairly close, the Decision Tree model scores highest on the train set with 87.5%. Now that the best parameters have been found for each model, they are applied to the test data to determine each model's predictive power.

GridSearchCV Results (Best Parameters)

```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

Logistic Regression

```
tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
accuracy : 0.8482142857142856
```

Support Vector Machine

```
tuned hpyerparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
accuracy : 0.8482142857142858
```

K-Nearest Neighbors

```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'splitter': 'random'}
accuracy : 0.875
```

Decision Tree

Predictive Analysis Results

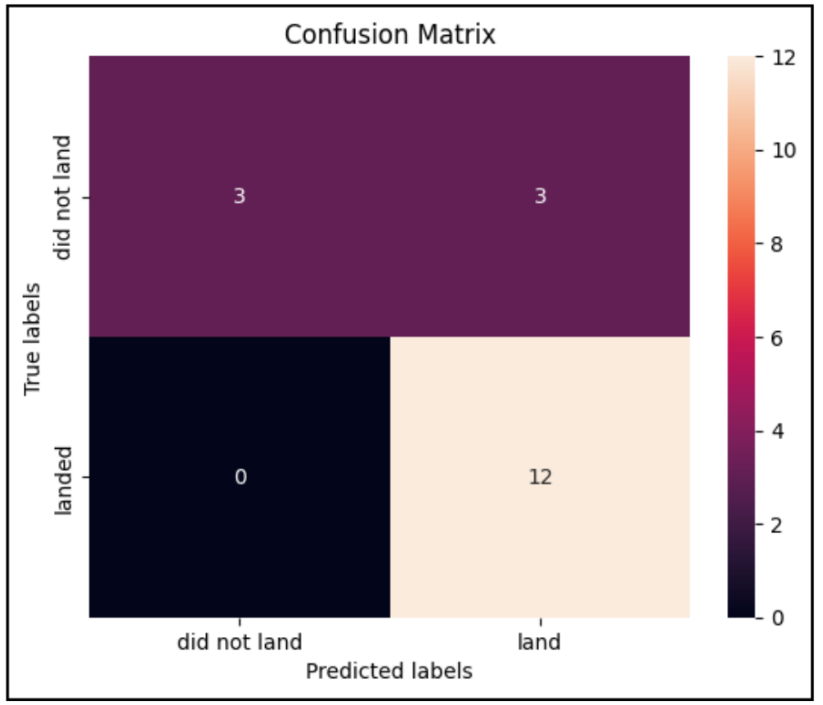
Remarkably, each of our employed models achieved an identical accuracy score of 83.3% on the test data. Similarly, the confusion matrices highlighting the distribution of True Positives and False Positives are identical, indicating the major issue across all models is false positives.

Based on these scores, there is no clear best method. Further exploration would be needed to determine what model, if any, is best to predict space ship landing outcomes. This includes ranking features by importance, which impacts some models, such as decision trees, greater. Scaling methods, as a StandardScaler() transformation, will impact various models differently as well and should cater the data best to the specific model used. Model simplicity and real-time scalability should also be considered. Overall, to achieve better accuracy, further fine tuning, feature engineering, and potentially ensemble methods should be explored.

Test Data Results

Test Score (with .score() method)

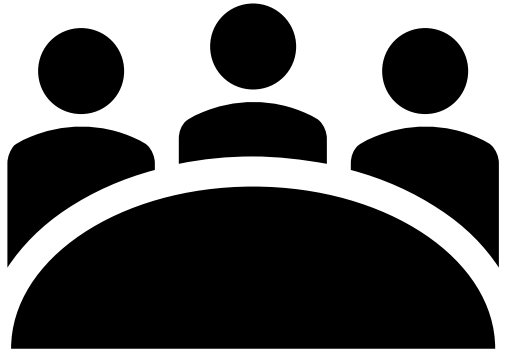
```
Test accuracy : 0.8333333333333334
```



Discussion

The original objective was to discover the key ingredients to SpaceX's Falcon 9 rocket landing success, and how this can be utilized to uncover the cost of a successful launch. By determining the most important factors in launch outcome, we aimed to use supervised machine learning techniques to predict a successful launch. With support of the literature review, we were able to use firm background knowledge to guide and further support the chosen features for analysis, model training, and execution.

Through the use of industry standard data science tools and techniques, many crucial insights were uncovered. Multiple data sources offered a complete view of historical launch data and offered key features for analysis and prediction. Expert data cleaning, wrangling, encoding, and feature engineering allowed us to transform the raw data into meaningful, easily parsable formats for visualizations, interactive dashboards, and classification models.



Exploration of the data unveiled a few primary revelations. As suggested by our literature review, Payload Mass and Flight Number are pivotal in the success of SpaceX's launches, which is supported by our scatterplots and Plotly Dashboard. The consistency in scores across all models suggest that while success can be predicted at some scale, there are still other unpredictable variables present that have an impact on the launches' outcome.

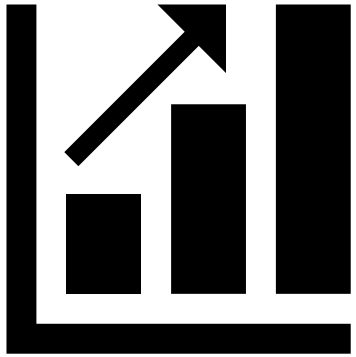
The models, while adept, couldn't provide a complete deciphering of the SpaceX enigma. Variability in external conditions, minor technical nuances, and sheer chance mean that while we can predict and prepare, certainties in space remain elusive.

Our journey through the data did bridge many knowledge gaps. We now have a firmer grasp on launch site efficiencies, the significance of payload masses, and the general reliability of rockets landing successfully. Just like the furthest reaches of space, there remains much more to be explored to paint the full picture.

Overall Findings and Implications

Findings

- **Insight into PayloadMass**: Our analysis spotlighted the significant role that PayloadMass plays in the successful landing of rockets.
- **FlightNumber's Significance**: Flight numbers, indicative of SpaceX's experience curve, showed a marked influence on success rates.
- **Model Consistency**: Across all predictive models, an 83.3% accuracy emerged, underscoring the consistent factors impacting landing success.
- **Launch Sites Insights**: EDA revealed discernible patterns regarding the efficacy of various launch sites, their geographical location and proximity, with certain sites emerging as more favorable than others.



Implications

- **Strategic Decisions**: The importance of PayloadMass could influence SpaceX's decisions on cargo and equipment for future missions.
- **Launch Site Optimizations**: The insights into launch site performances could lead SpaceX to optimize certain locations or even consider new sites.
- **Mission Preparedness**: Knowing the predictors of success can help in refining pre-launch checklists and protocols.
- **Continuous Learning**: The consistent accuracy of our models suggests that SpaceX's learnings over missions are methodical and can be predicted, but there's always room for further refinement.
- **Industry Benchmarking**: These findings can serve as a benchmark for other aerospace endeavors, guiding best practices.
- **Public Perception**: Greater predictability and insights into successes can bolster public confidence in space missions.
- **Investment Decisions**: Investors can be apprised of the systematic determinants of SpaceX's success, aiding future funding decisions.

Conclusion

Our journey into the SpaceX data embarked with clear intent: to unravel the key factors of SpaceX rocket launch success. Through exploratory data analysis, SQL insights, visual narratives, and predictive modeling, we've illuminated pivotal facets influencing these endeavors. Notably, PayloadMass, FlightNumber, and launch site efficacy emerged as significant indicators of outcome.

The implications of these findings are profound. They pave the way for operational improvements, offering guidance for mission preparation and payload decisions, as well as invaluable strategic insights. This analysis, with an accuracy of 83.3%, has highlighted potential avenues for optimizing launch sites and refining various pre-launch protocols. Nevertheless, there is room for further refinement as newer data streams in, and avenues for deeper dives into uncharted variables.

As SpaceY grows and contemplates future endeavors, collaborative efforts and expanded data resources could broaden understanding of the aerospace landscape even further. In the vast expanse of space exploration, it's often the granular nuances of data that help chart our course. This exploration with SpaceX data stands as testament, a reminder that when it comes to understanding the cosmos, the sky isn't the limit; it's just the beginning.

References

- **Romanova, N., Crosby, N., & Pilepenko, V.** (2013). *Relationship of Worldwide Rocket Launch Crashes with Geophysical Parameters*
  - **Journal:** International Journal of Geophysics
  - Link
- **Space Exploration Technologies Corp** (2009, 2021). *Falcon 9 User's Guide*
  - 2009 Link, 2021 Link
- **Sforza, P.** (2016). *Manned Spacecraft Design Principles, Chapter 7 – Launch Mechanics*

Appendix

Dataframes before data wrangling and feature encoding

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10
...	...	...	...	...	...	...	...	...	...	...	...
116	117	CCSFS	Starlink	15,600 kg	LEO	SpaceX	Success\n	F9 B5B1051.10	Success	9 May 2021	06:42
117	118	KSC	Starlink	~14,000 kg	LEO	SpaceX	Success\n	F9 B5B1058.8	Success	15 May 2021	22:56
118	119	CCSFS	Starlink	15,600 kg	LEO	SpaceX	Success\n	F9 B5B1063.2	Success	26 May 2021	18:59
119	120	KSC	SpaceX CRS-22	3,328 kg	LEO	NASA	Success\n	F9 B5B1067.1	Success	3 June 2021	17:29
120	121	CCSFS	SXM-8	7,000 kg	GTO	Sirius XM	Success\n	F9 B5	Success	6 June 2021	04:26

121 rows × 11 columns

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

Launch Sites and Coordinates

	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610745

Dataframes used for predictive models

Target Dataframe

```
data.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

Encoded Feature Dataframe

```
X.head(100)
```

	FlightNumber	PayloadMass	Flights	Block	ReusedCount	Orbit_ES-L1	Orbit_GEO	Orbit_GTO	Orbit_HEO	Orbit_ISS	...	Serial_B1058	Serial_B1059	Serial_B1060	Serial_B1062	GridFins_False	GridFins_True	Reused_False	Reused_True	Legs_False	Legs_True
0	1.0	6104.959412	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0
1	2.0	525.000000	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0
2	3.0	677.000000	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0
3	4.0	500.000000	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0
4	5.0	3170.000000	1.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
85	86.0	15400.000000	2.0	5.0	2.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	1.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0
86	87.0	15400.000000	3.0	5.0	2.0	0.0	0.0	0.0	0.0	0.0	...	1.0	0.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0
87	88.0	15400.000000	6.0	5.0	5.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0
88	89.0	15400.000000	3.0	5.0	2.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	1.0	0.0	0.0	1.0	0.0	1.0	0.0	1.0
89	90.0	3681.000000	1.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	1.0	0.0	1.0	1.0	0.0	0.0	1.0

90 rows × 83 columns