IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

CHAN KAI KHEE
16 OCTOBER 2023

Outline

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

Executive Summary

- Data collection methodology:
  - SpaceX API
- Perform data wrangling
  - Web Scraping
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Logistic regression, random forest, SVM and KNN
  - The highest accuracy is 83.33%.

Introduction

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

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX API
- Perform data wrangling
  - Web Scraping
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Logistic regression, random forest, SVM and KNN

Data Collection – SpaceX API

- Import libraries
- Uses different columns to call the API from the launch data
- Request rocket launch data: https://api.spacexdata.com/v4/launches/past
- Decode the content and turn it into dataframe
- Use API again to get information about the launches – (rocket, payload, launchpad, cores)
- Construct and combine the columns into a dictionary and put into dataframe

Data Collection - Scraping

- Import libraries (beautifulsoup, requests)
- Request the HTML page: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922
- Request the Falcon9 Launch Wiki page from its URL using BeautifulSoup
- Extract all column names from the HTML table header
- Create a data frame by parsing the launch HTML tables

Data Wrangling

Deal with missing values

.mean() and .replace() are used to replace np.nan values in the data

EDA with Data Visualization

- Import the libraries such as **numpy, pandas, and seaborn**
- Use **catplot function** to plot the **scatter chart**
- Plot the **bar chart** and **line chart** to understand the information of the data
- Use the function get_dummies to apply **OneHotEncoder** to convert categorical columns to numerical columns

EDA with SQL

- Connect to the database by loading SQL extension and establish a connection with the database

  %load_ext sql

  %sql sqlite:///my_data1.db

- Write and execute SQL queries to get different information

Build an Interactive Map with Folium

- **Markers, circles, lines** are used to draw folium map
- These objects can highlight the important information on the map to give more clearer explanation.

Build a Dashboard with Plotly Dash

- **Pie chart and scatter chart** are plotted
- The charts show the success count for all launch sites and the success count on payload mass

Predictive Analysis (Classification)

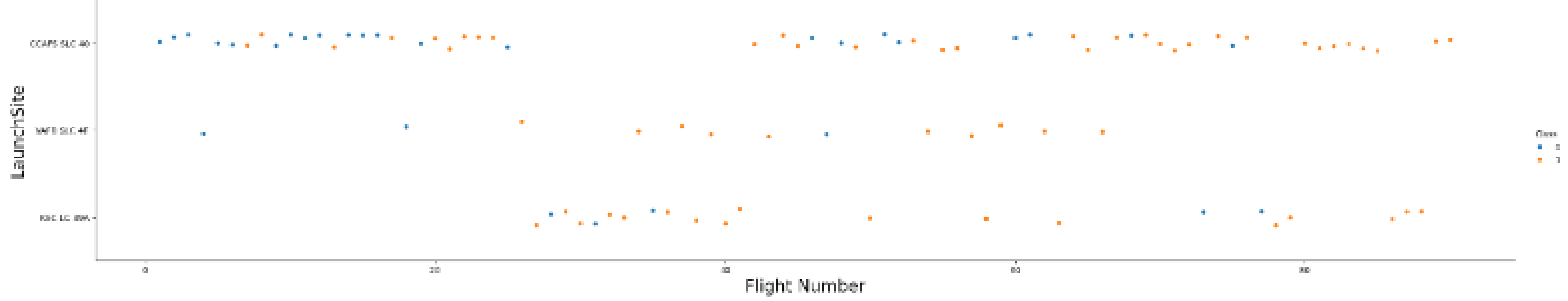
- The machine leaning algorithms such as **logistic regression, random forest, SVM and KNN** are used to perform classification
- The data is split into 80% train dataset and 20% test dataset.
- The models are evaluated using confusion matrix.

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

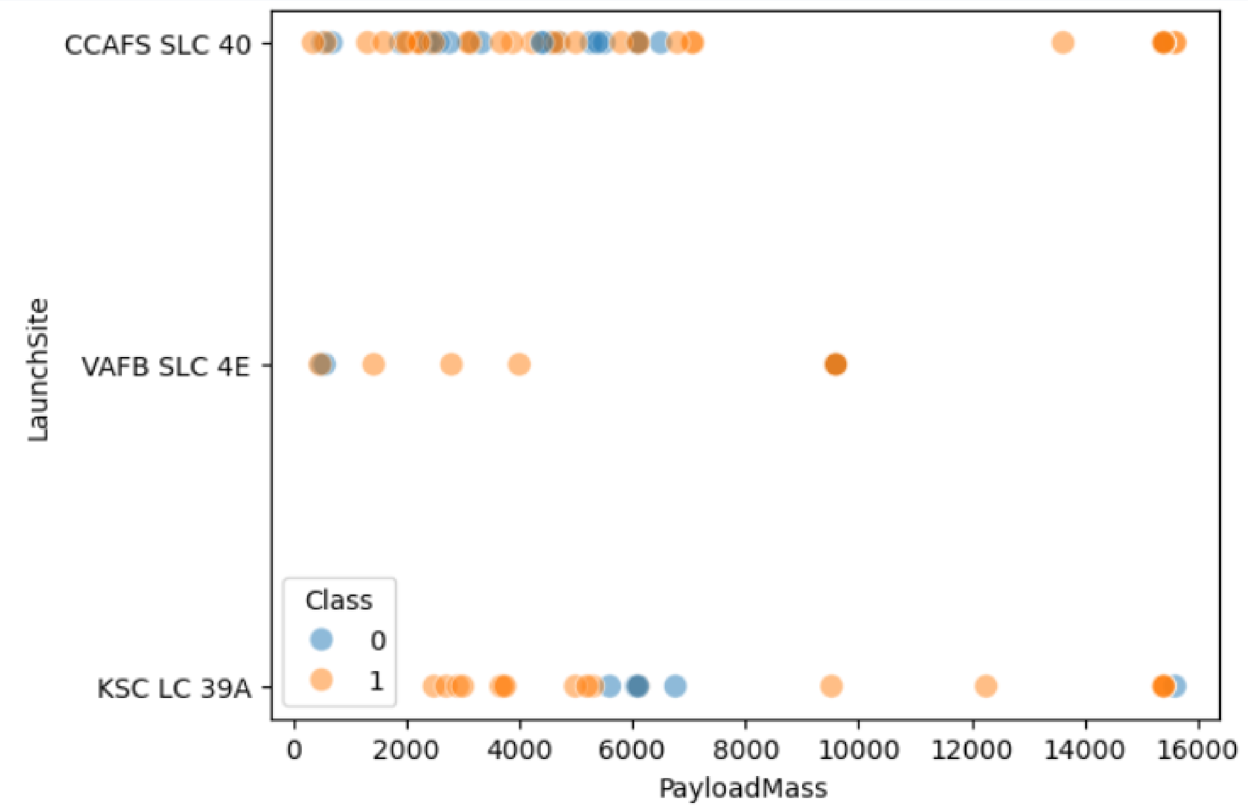
Relationship between flight number and launch site



CCAFS SLC 40 launch site has the highest launch records.

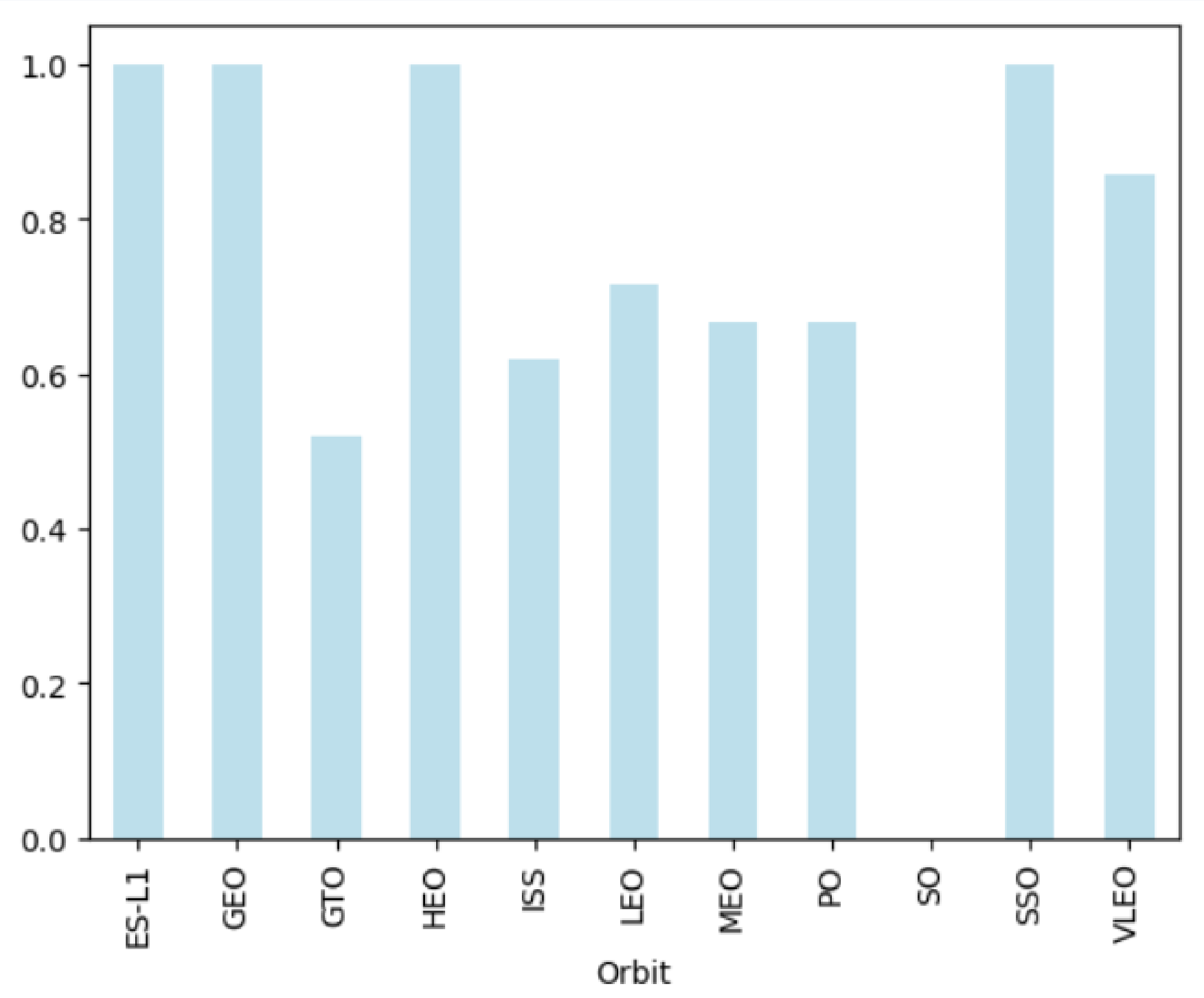
Payload vs. Launch Site

VAFB-SLC launch there are no rockets launched for heavy payload mass(greater than 10000)



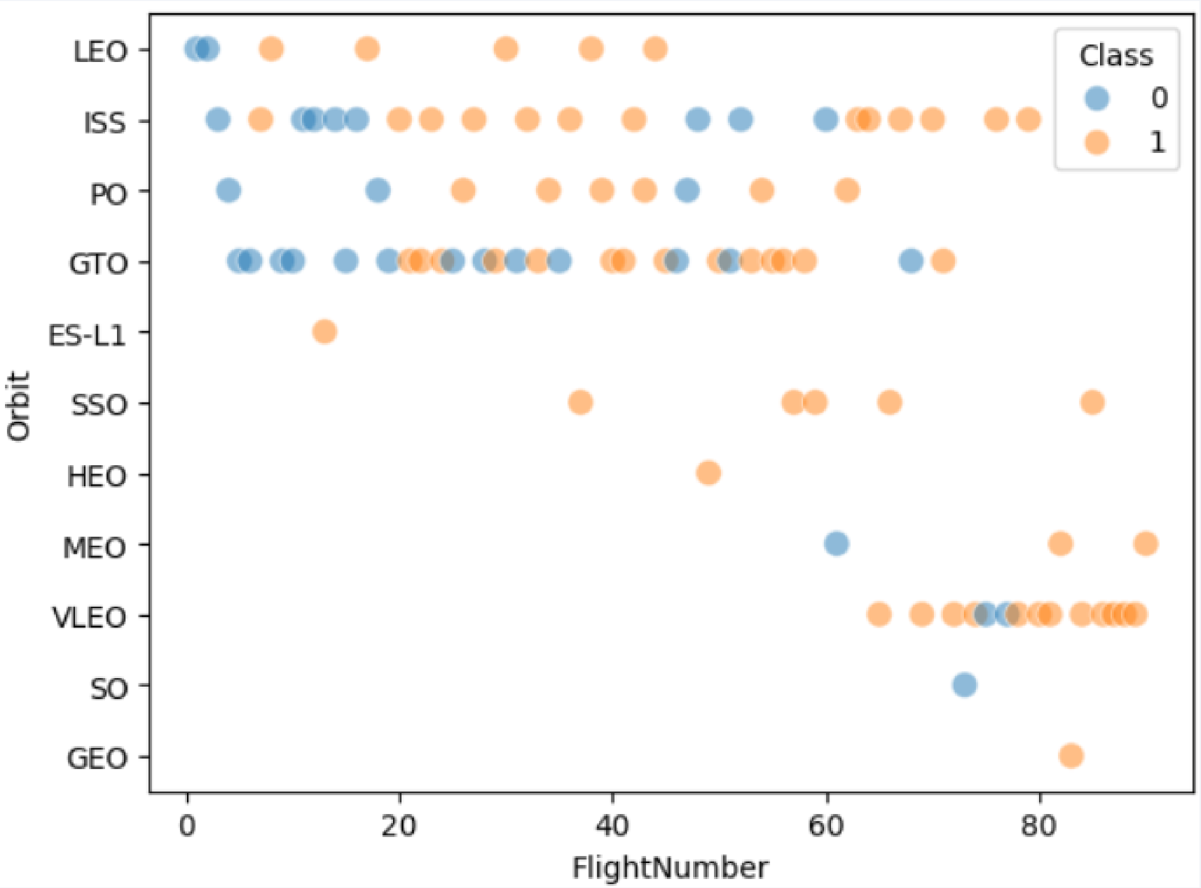
Success Rate vs. Orbit Type

The orbits which have the highest success are **ES-L1, GEO, HEO,** and **SSO.**



Flight Number vs. Orbit Type

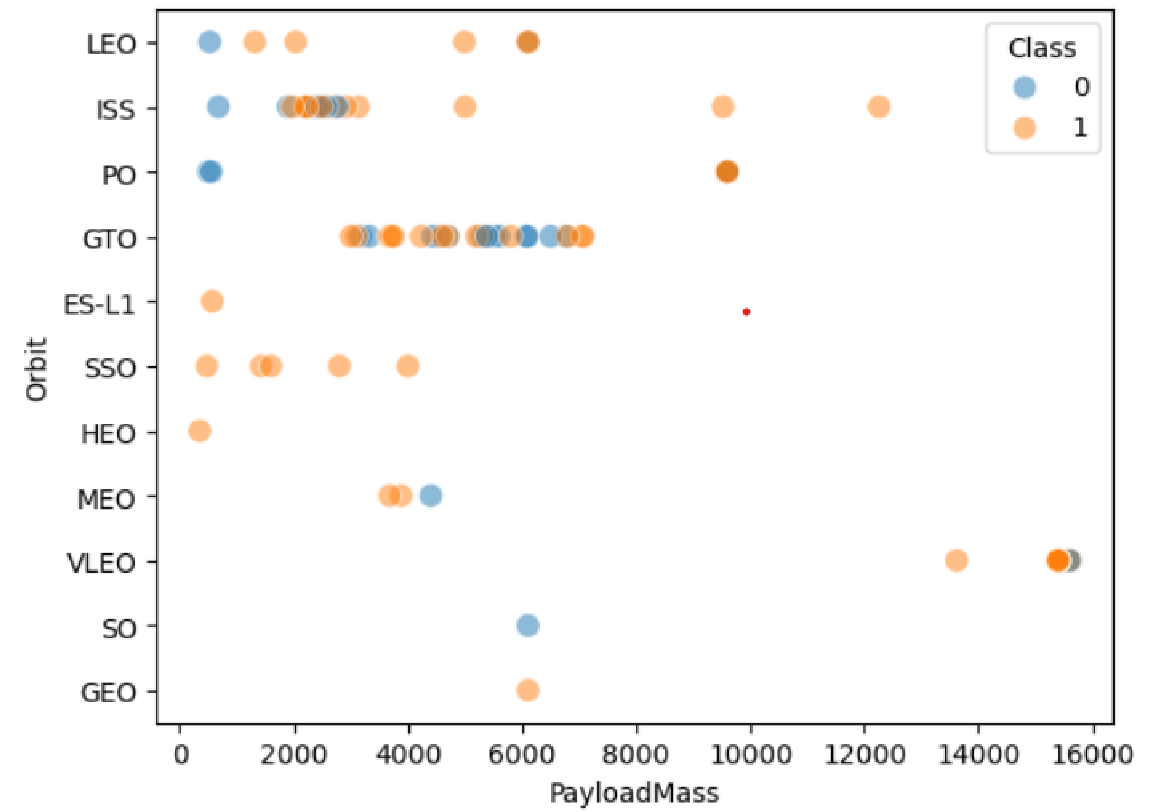
LEO orbit the success appears related to the number of flights; on the other hand, there seems to be no relationship between **flight number when in GTO** orbit.



Payload vs. Orbit Type

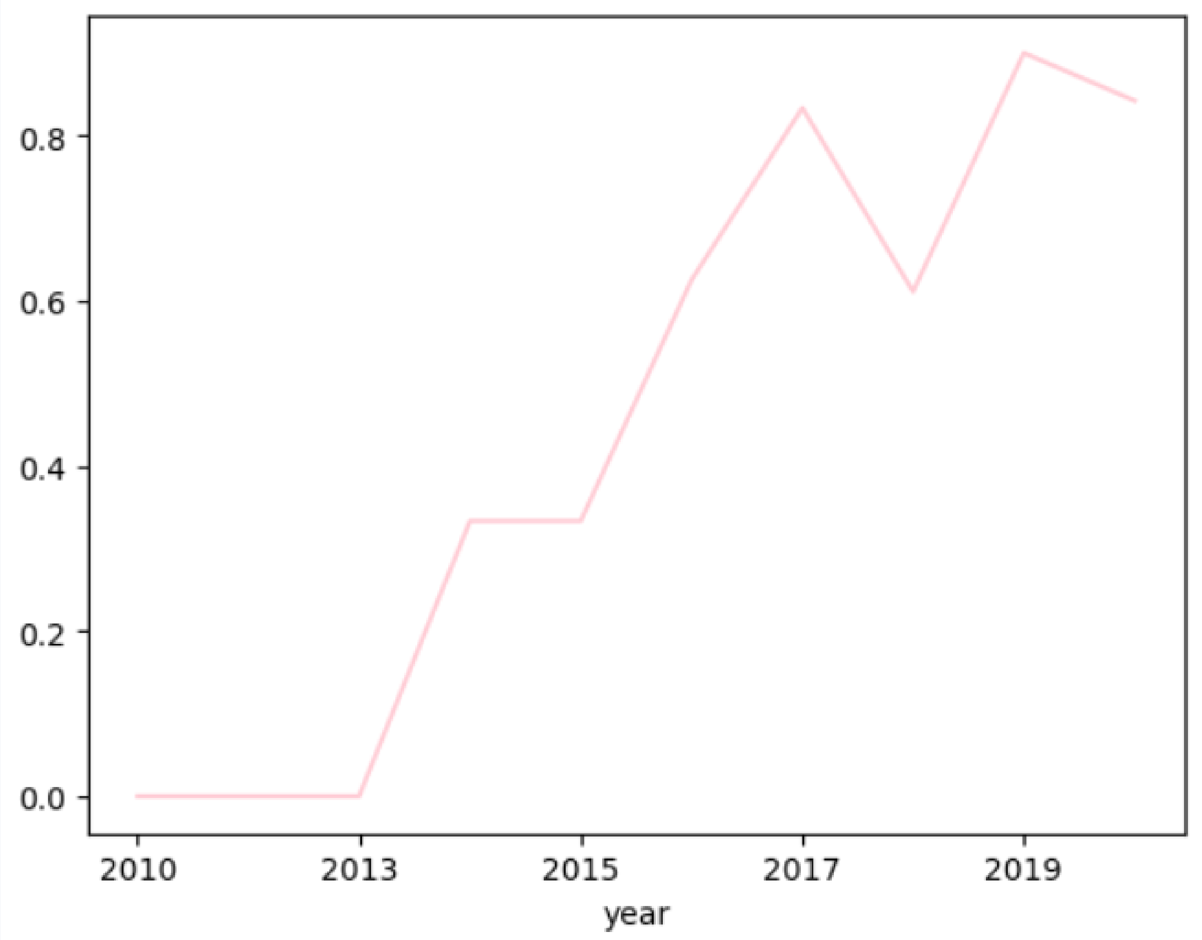
With heavy payloads the successful landing or positive landing rate are more for **Polar,LEO and ISS**.

However for **GTO** we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.



Launch Success Yearly Trend

The success rate is **increasing** from year to year.



All Launch Site Names

Number of launches on each unique launch site

CCAFS SLC 40 → 55

KSC LC 39A → 22

VAFB SLC 4E → 13

The highest number of unique launch sites is from **CCAFS SLC 40**.

Launch Site Names Begin with 'CCA'

The 5 records where launch sites begin with `CCA’

CCAFS LC-40

CCAFS LC-40

CCAFS LC-40

CCAFS LC-40

CCAFS LC-40

Query

```
count=0
for site in df["Launch_Site"]:
    if "CCA" in site and count <5:
        print(site)
        count=count+1
```

Total Payload Mass

The total payload carried by boosters from NASA is **619967KG**.

Query

```
total_mass=df["PAYLOAD_MASS__KG_"].sum()
```

Average Payload Mass by F9 v1.1

The average payload mass carried by booster version F9 v1.1 is **2928.4**

Query

```
df_F9_1_1=df[df["Booster_Version"] == "F9 v1.1"]
df_F9_1_1["PAYLOAD_MASS__KG_"].mean()
```

First Successful Ground Landing Date

The dates of the first successful landing outcome on ground pad

is on **22-12-2015**

Query

```
df_ground_pad=df[df['Landing_Outcome'] == 'Success (ground pad)' ]
df_ground_pad.iloc[0,0]
```

Successful Drone Ship Landing with Payload between 4000 and 6000

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

'F9 FT B1022'

'F9 FT B1026'

'F9 FT B1021.2'

'F9 FT B1031.2'

Query

```
df_success_boosters=df[(df['Landing_Outcome'] == 'Success (drone ship)') & (df['PAYLOAD_MASS__KG_'] < 6000) & (df['PAYLOAD_MASS__KG_'] > 4000)]

booster=df_success_boosters["Booster_Version"].to_list()
```

Total Number of Successful and Failure Mission Outcomes

There were **100 succesful** and **1 unsuccesful** mission

Query

```
success=0
failure=0
for outcome in df['Mission_Outcome']:
    if "Success" in outcome:
        success=success+1
    else:
        failure=failure+1
print(f"There were {success} succesfull and {failure} unsuccesfull mission")
```

Boosters Carried Maximum Payload

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

['F9 B5 B1048.4', 'F9 B5 B1049.4', 'F9 B5 B1051.3', 'F9 B5 B1056.4',

'F9 B5 B1048.5', 'F9 B5 B1051.4', 'F9 B5 B1049.5',

'F9 B5 B1060.2 ', 'F9 B5 B1058.3 ', 'F9 B5 B1051.6',

'F9 B5 B1060.3', 'F9 B5 B1049.7 ‘]

Query

```
max_load=df[df["PAYLOAD_MASS__KG_"] == df['PAYLOAD_MASS__KG_'].max()]
max_load['Booster_Version'].unique()
```

2015 Launch Records

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

['F9 v1.1 B1012', 'F9 v1.1 B1015', 'F9 v1.1 B1017', 'F9 FT B1020', 'F9 FT B1024’]

Query

```
dates=[]
for date in df["Date"]:
    if date[6:10] == "2015":
        dates.append(date)
df_2015=df[df["Date"].isin(dates)]
df_2015=df[(df["Landing_Outcome"]=='Failure (drone ship)')]
failed_boosters=df_2015["Booster_Version"].to_list()
print(f"the boosters \n {failed_boosters} \n have failed to land with drone ships in 2015")
```

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

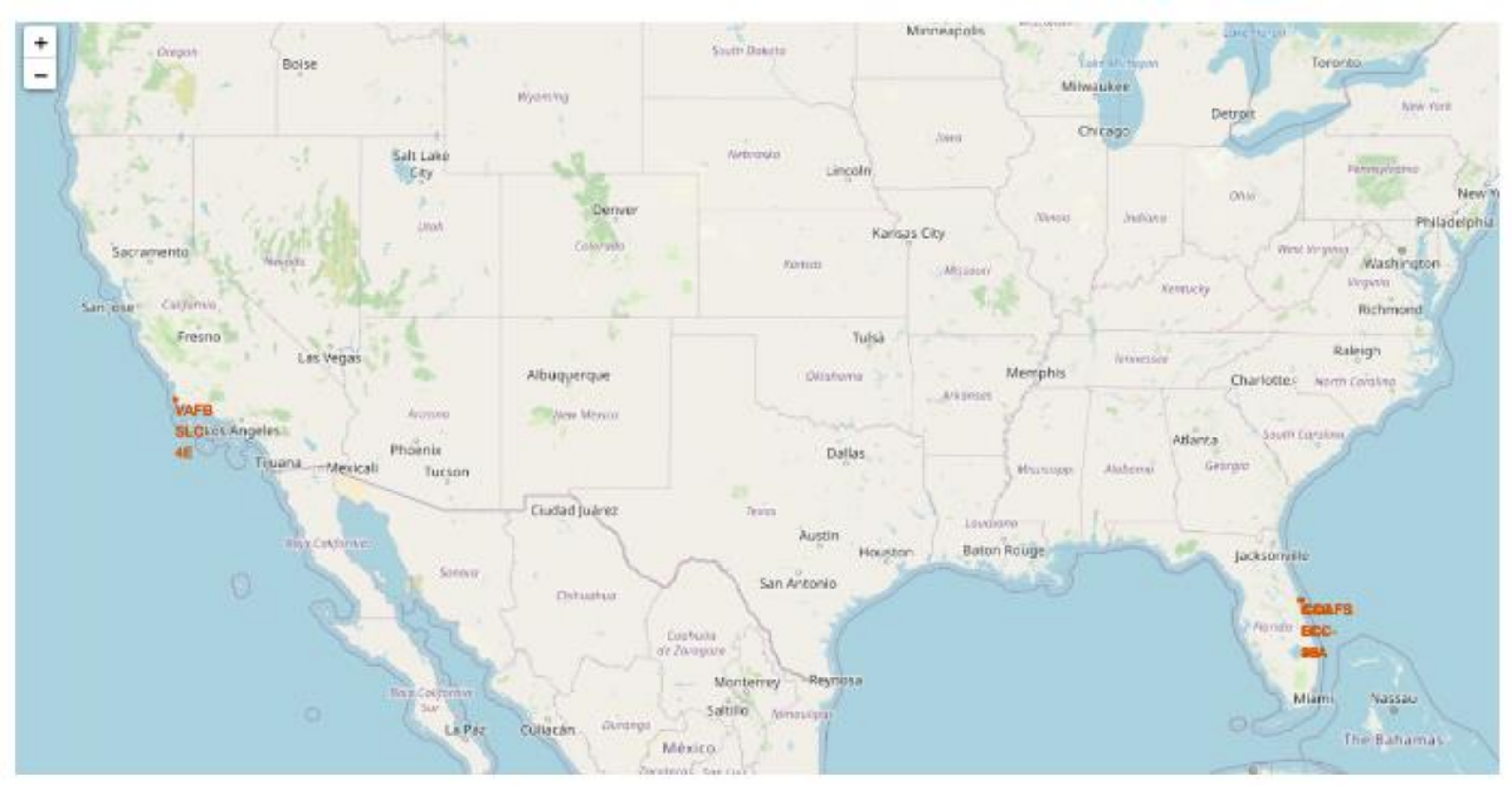
Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

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

Section 3

Launch Sites Proximities Analysis

Location of the launch site



- The map shows that all the launch site are very close to the coast.

Success/failed launches for each site

- The map tells us which launch sites have relatively high success rates.

Distance between a launch site to its proximities

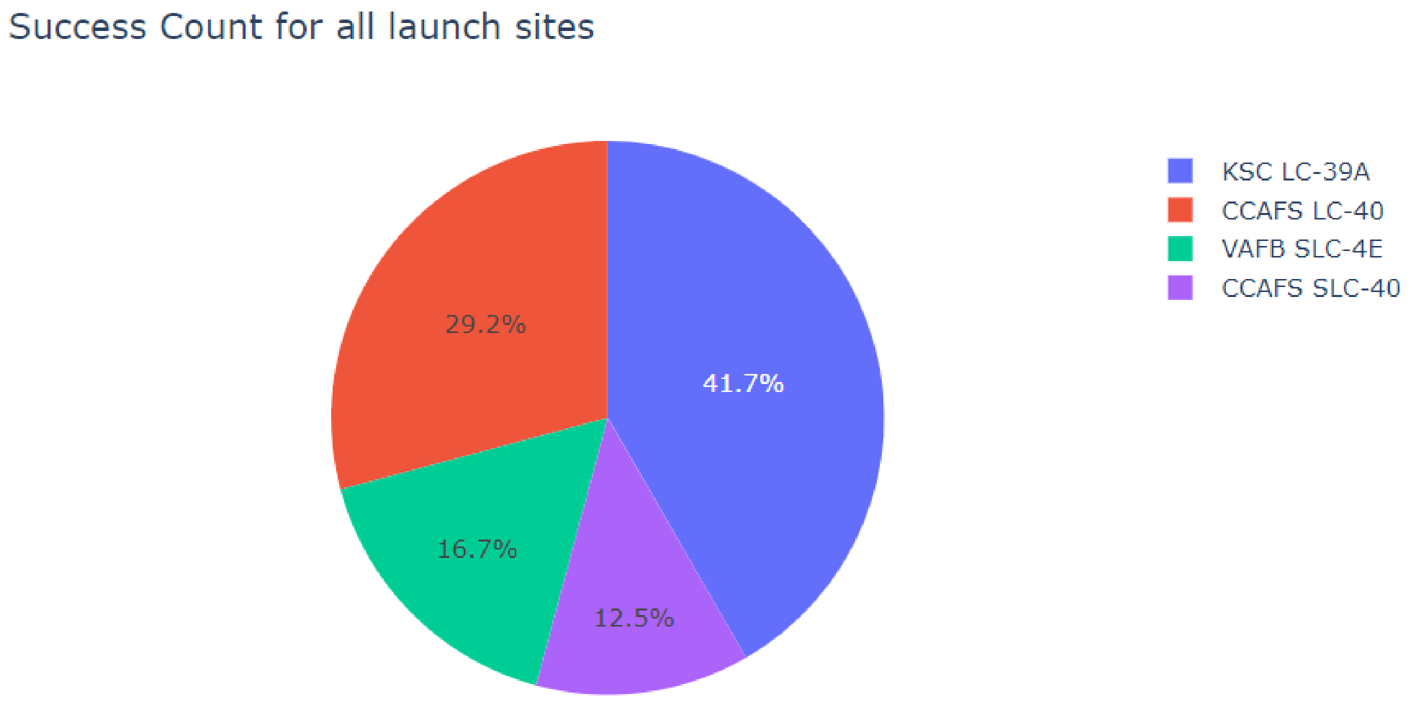
- The maps shows the distance between the launch site and the coast line.

Section 4

Build a Dashboard with Plotly Dash

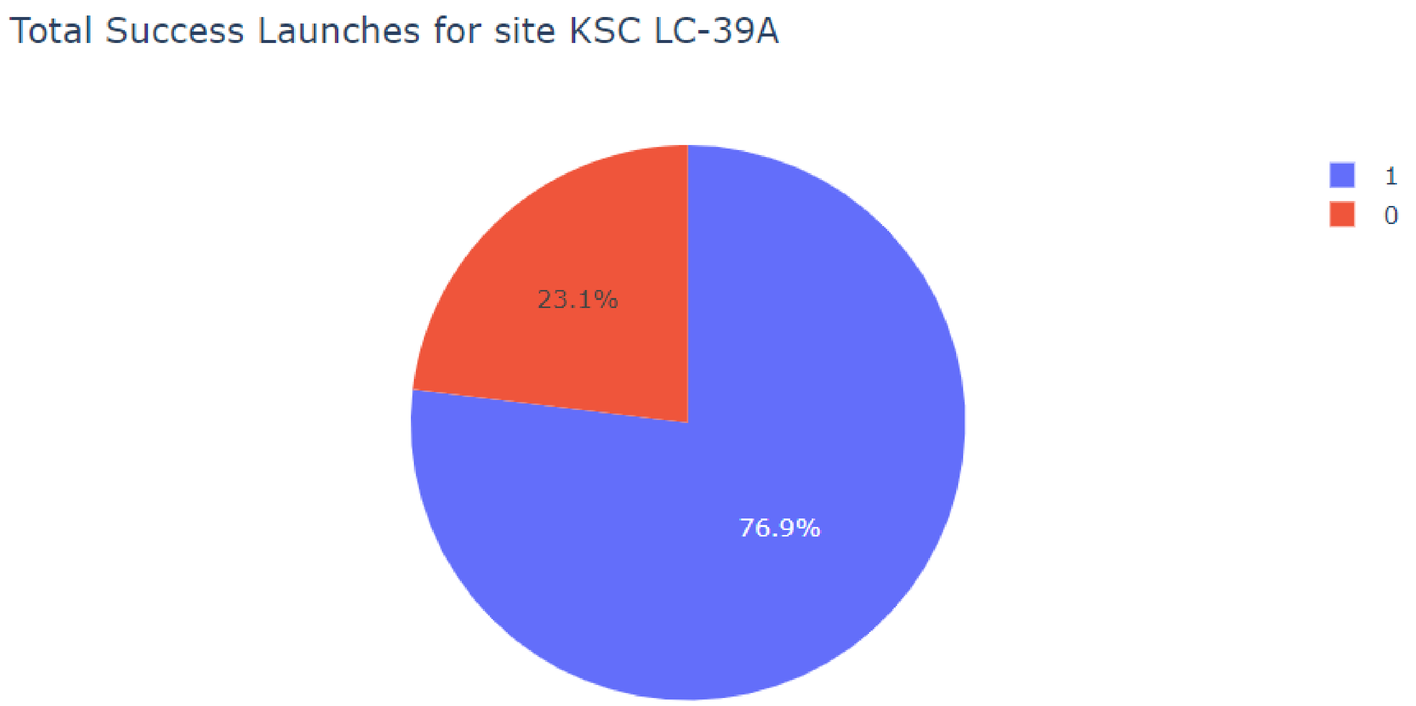
Success Count for all launch sites

- The pie chart shows that **KSC LC-39A** has the highest success count
- Meanwhile, **CCAFS SLC-40** has the lowest success rate.



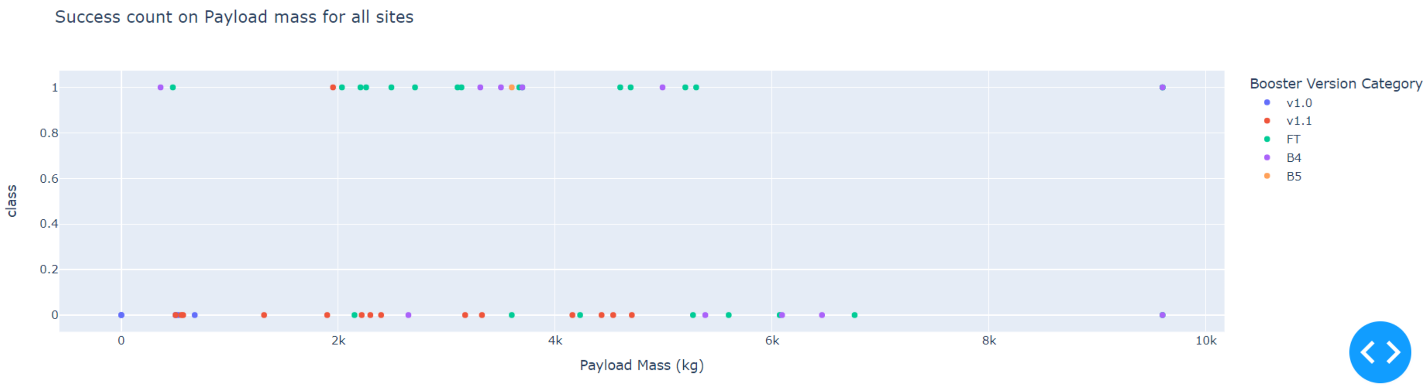
Total Success Launches for site

- The success rate is **76.9%** and unsuccess rate is **23.1%**



Success count on Payload mass

- The chart shows that **v1.1** launch site is most likely to fail no matter the payload mass.
- **FT** launch site is most likely to succeed if the payload mass is below 6k kg.



Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The models have the highest accuracy which is 83.33% are **logistic regression, SVM and KNN.**

Find the method performs best:

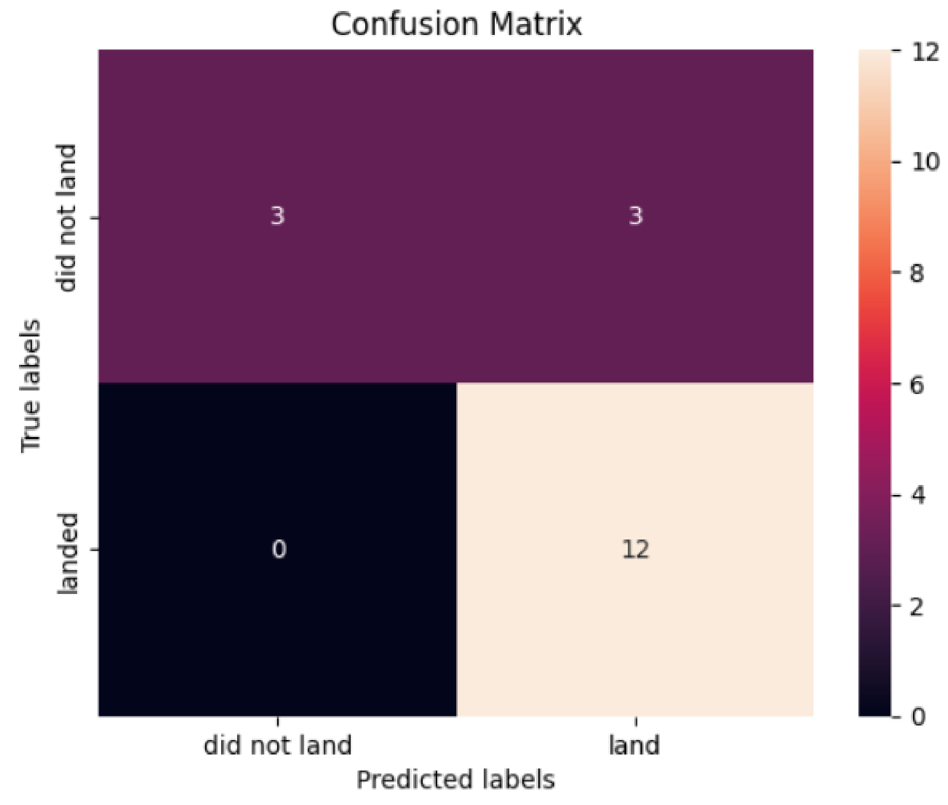
```
predictors = [knn_cv, svm_cv, logreg_cv, tree_cv]
best_predictor = ""
best_result = 0
for predictor in predictors:

    predictor.score(X_test, Y_test)
```

```
0.8333333333333334
```

Confusion Matrix

- The major problem is false positives. There are 12 cases showing that the rocket has landed but actually not.



Conclusions

The model gives an accuracy of **83.33%** to predict the Falcon 9 first stage will land successfully. It is a very good step for SpaceX to reuse the first stage and save much of the cost.

Besides, every step from the beginning to the end is important to extract useful information from the data including data wrangling, EDA, drawing maps using folium, and creating a dashboard. These can ease the process of the prediction and provide ideal results.

Appendix

```python
import dash
from dash import dcc
from dash import html
from dash.dependencies import Input, Output
import pandas as pd
import plotly.graph_objs as go
import plotly.express as px

data=pd.read_csv('https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_dash.csv')

max_payload=data['Payload Mass (kg)'].max()
min_payload=data['Payload Mass (kg)'].min()

app = dash.Dash(__name__)

app.layout=html.Div([
    html.H1("SpaceX Launch Records Dashboard",style={'color': '#503D36', 'font-size': 24}),
    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 here',
                    searchable=True
        ),

        html.Div([
         dcc.RangeSlider(
            id='payload-slider',
            min=0, max=10000,step=1000,
            marks={0: '0',
                   100: '100'},
                value=[min_payload, max_payload]
         )
        ]),

        html.Div(dcc.Graph(id='success-pie-chart')),

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

# 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):
    filtered_df = data
    if entered_site == 'ALL':
        fig = px.pie(data, values='class',
        names='Launch Site',
        title='Success Count for all launch sites')
        return fig
    else:
        filtered_df=data[data['Launch Site']== entered_site]
        filtered_df=filtered_df.groupby(['Launch Site','class']).size().reset_index(name='class count')
        fig=px.pie(filtered_df,values='class count',names='class',title=f"Total Success Launches for site {entered_site}")
        return fig
        # return the outcomes piechart for a selected site

@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 scatter(entered_site, payload):
    filtered_df=data[data['Payload Mass (kg)'].between(payload[0],payload[1])]

    if entered_site=='ALL':
        fig=px.scatter(filtered_df,x='Payload Mass (kg)',y='class',color='Booster Version Category',title='Success count on Payload mass for all sit
        return fig
    else:
        fig=px.scatter(filtered_df[filtered_df['Launch Site']==entered_site],x='Payload Mass (kg)',y='class',color='Booster Version Category',title=
        return fig

if __name__ == '__main__':
    app.run_server(debug=True)
```

Thank you!