IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Vicente Bustamante
6/10/25

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- Project background and context
- Space X anuncia el lanzamiento del cohete Falcon 9 en su sitio web con un costo de 62 millones de dólares; otros proveedores cuestan más de 165 millones de dólares cada uno. Gran parte del ahorro se debe a que Space X puede reutilizar la primera etapa. Por lo tanto, si podemos determinar si la primera etapa aterrizará, podemos determinar el costo del lanzamiento. Esta información puede utilizarse si otra empresa desea pujar contra Space X por el lanzamiento de un cohete. El objetivo del proyecto es crear un flujo de trabajo de aprendizaje automático para predecir si la primera etapa aterrizará con éxito.
- Problems you want to find answers
- La interacción entre diversas características determina la tasa de éxito de un aterrizaje.¿Qué condiciones operativas deben existir para garantizar el éxito del programa de aterrizaje?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Los datos se recopilaron utilizando la API de SpaceX y web scrapping a Wikipedia
  - Perform data wrangling
  - One-hot encoding fue aplicado a características categóricas
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models:

Cómo construir, ajustar y evaluar modelos de clasificación

Data Collection

- Los datos se recopilaron mediante diversos métodos. La recopilación de datos se realizó mediante una solicitud GET a la API de SpaceX.A continuación, decodificamos el contenido de la respuesta como JSON mediante la función .json() y lo convertimos en un dataframe de Pandas mediante .json_normalize().Después, limpiamos los datos, verificamos si faltaban valores y los completamos cuando fue necesario.Además, realizamos un raspado web de Wikipedia para los registros de lanzamiento del Falcon 9 con BeautifulSoup.El objetivo era extraer los registros de lanzamiento como una tabla HTML, analizarla y convertirla en un dataframe de Pandas para su posterior análisis.

Data Collection – SpaceX API

- Utilize la solicitud GET a la API de SpaceX para recopilar datos, limpiar los datos solicitados e hicimos algunas manipulaciones y formateos básicos de datos.
- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose

1. Get request for rocket launch data using API

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

2. Use json_normalize method to convert json result to dataframe

In [12]:
```
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = res.json()
```

In [13]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

In [30]:
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

- Aplicamos el web scrapping para extraer los registros de lanzamiento de Falcon 9 con BeautifulSoup.Analizamos la tabla y la convertimos en un dataframe de Pandas.
- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose

1. Apply HTTP Get method to request the Falcon 9 rocket launch page

In [4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

In [5]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
html_data = requests.get(static_url)
html_data.status_code
```

Out[5]:
```
200
```

2. Create a BeautifulSoup object from the HTML response

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text, 'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

In [7]:
```
# Use soup.title attribute
soup.title
```

Out[7]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

3. Extract all column names from the HTML table header

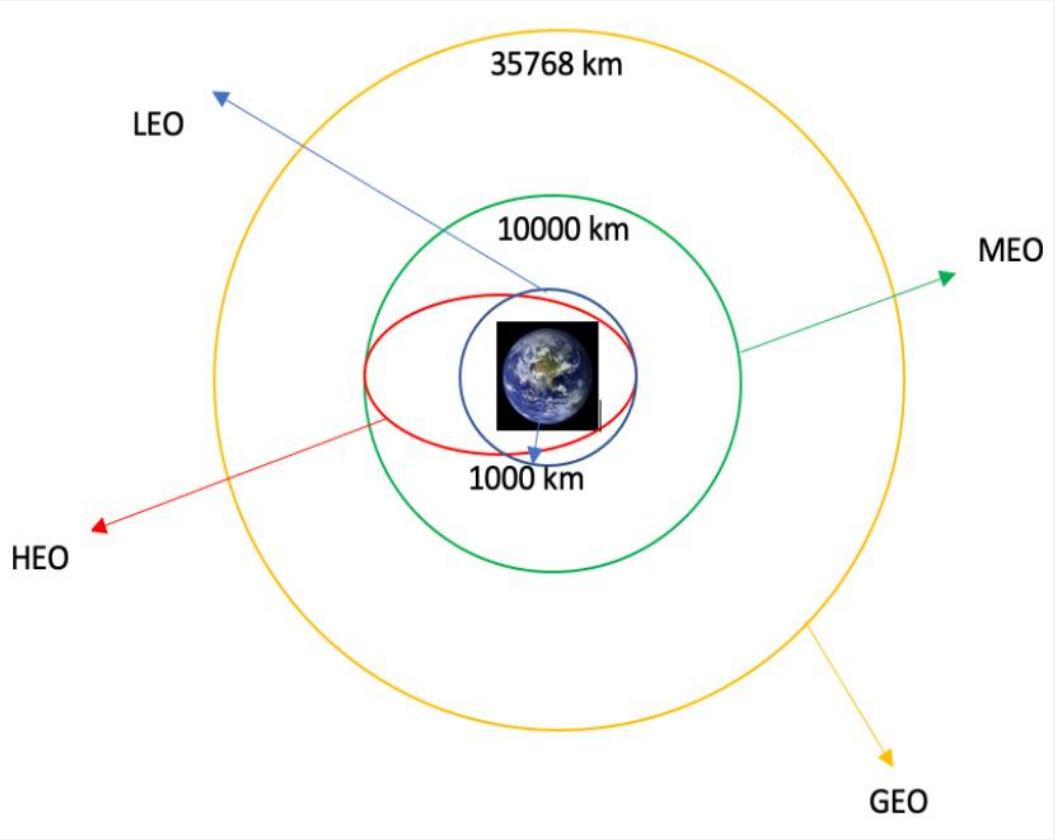
In [10]:
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

4. Create a dataframe by parsing the launch HTML tables
5. Export data to csv

Data Wrangling

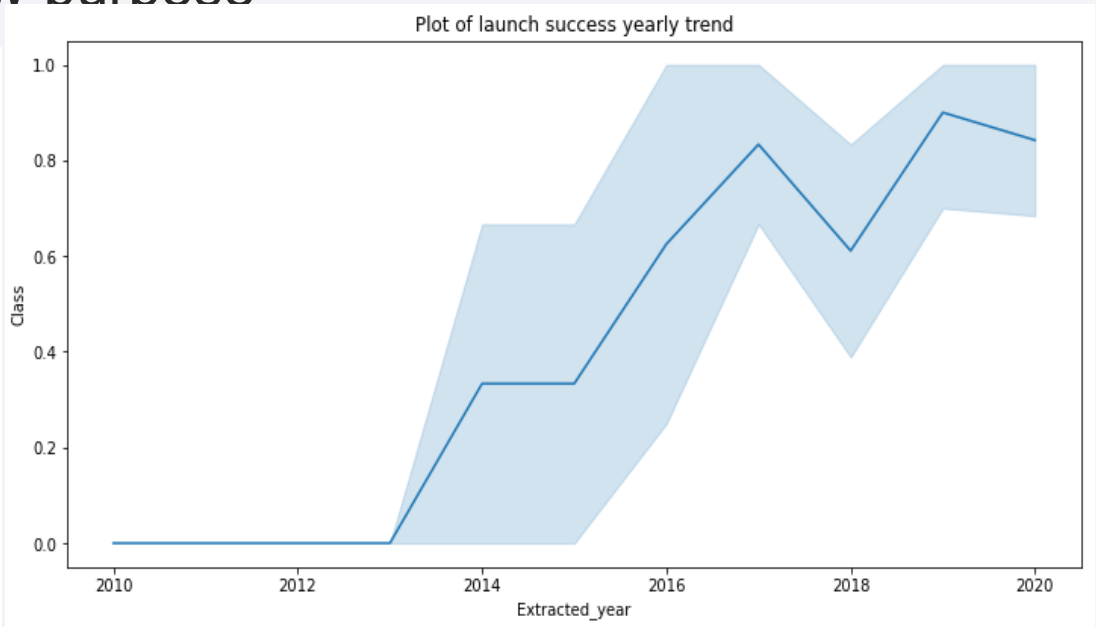
- Realizamos un análisis exploratorio de datos y determinamos las etiquetas de entrenamiento.Calculamos el número de lanzamientos en cada sitio, así como el número y la frecuencia de cada órbita.Creamos la etiqueta de resultado del aterrizaje a partir de la columna de resultados y exportamos los resultados a CSV.

- Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose



EDA with Data Visualization

- Exploramos los datos visualizando la relación entre el número de vuelo y el sitio de lanzamiento, la carga útil y el sitio de lanzamiento, la tasa de éxito de cada tipo de órbita, el número de vuelo y el tipo de órbita y la tendencia anual de éxito del lanzamiento.
- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose



EDA with SQL

- Se cargo el conjunto de datos de SpaceX en una base de datos PostgreSQL sin salir del Jupyter Notebook.Aplicamos EDA con SQL para obtener información de los datos. Creamos consultas para averiguar, por ejemplo:Los nombres de los sitios de lanzamiento únicos en la misión espacial.La masa total de la carga útil transportada por los propulsores lanzados por la NASA (CRS).La masa promedio de la carga útil transportada por la versión F9 v1.1 del propulsor.El número total de misiones exitosas y fallidas.Los aterrizajes fallidos de los drones, su versión de propulsor y los nombres de los sitios de lanzamiento.

- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose

Build an Interactive Map with Folium

- Marcamos todos los sitios de lanzamiento y añadimos objetos como marcadores, círculos y líneas para indicar el éxito o el fracaso de los lanzamientos en cada sitio del mapa de folio.Asignamos los resultados de lanzamiento característicos (éxito o fracaso) a las clases 0 y 1; es decir, 0 para fracaso y 1 para éxito.Utilizando los grupos de marcadores con etiquetas de color, identificamos los sitios de lanzamiento con una tasa de éxito relativamente alta.Calculamos las distancias entre un sitio de lanzamiento y sus proximidades. Permitiendonos responder a algunas preguntascomo lo son :¿Están los sitios de lanzamiento cerca de vías férreas, carreteras y costas?¿Se mantienen los sitios de lanzamiento a cierta distancia de las ciudades?

Build a Dashboard with Plotly Dash

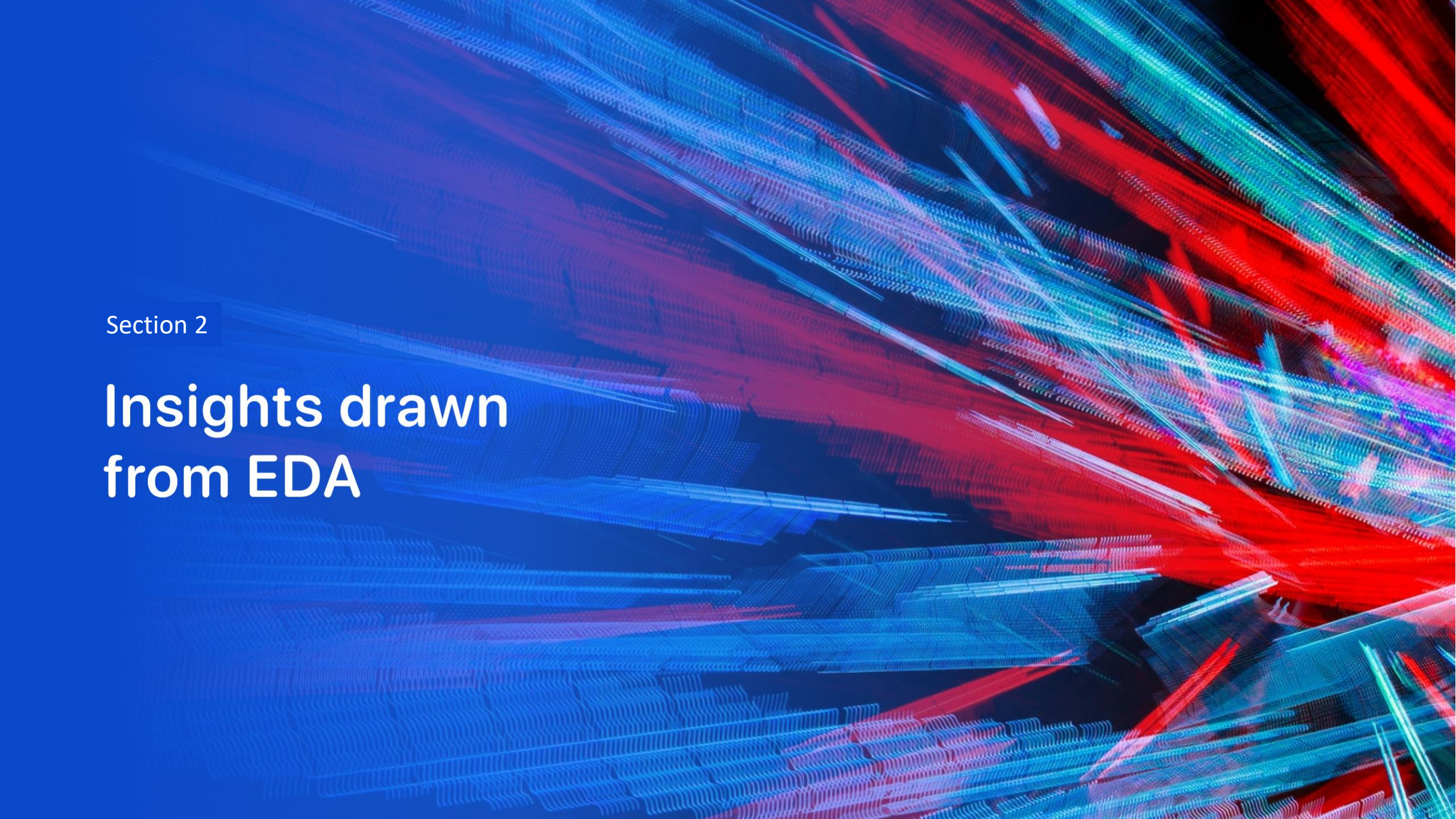
- Creamos un panel interactivo con Plotly dash.Creamos gráficos circulares que muestran el total de lanzamientos en determinados sitios.Creamos un gráfico de dispersión que muestra la relación entre el resultado y la masa de la carga útil (kg) para las diferentes versiones del propulsor.
- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

Predictive Analysis (Classification)

- Cargamos los datos con Numpy y Pandas, los transformamos y los dividimos en entrenamiento y prueba.Creamos diferentes modelos de aprendizaje automático y ajustamos diferentes hiperparámetros con GridSearchCV.Usamos la precisión como métrica para nuestro modelo y lo mejoramos mediante ingeniería de características y ajuste de algoritmos.Encontramos el modelo de clasificación con mejor rendimiento.
  Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

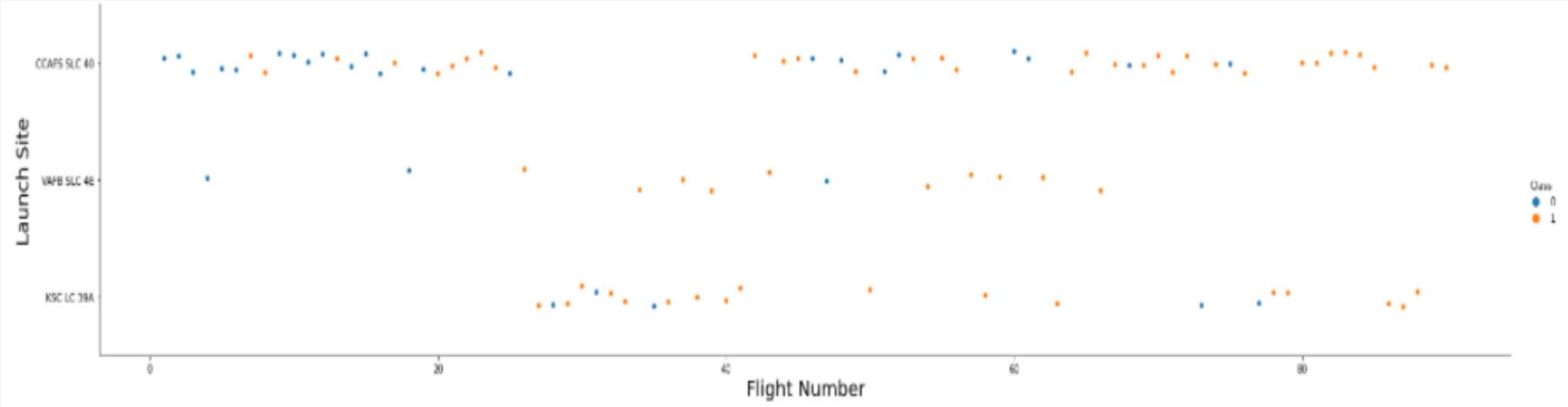
Results

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

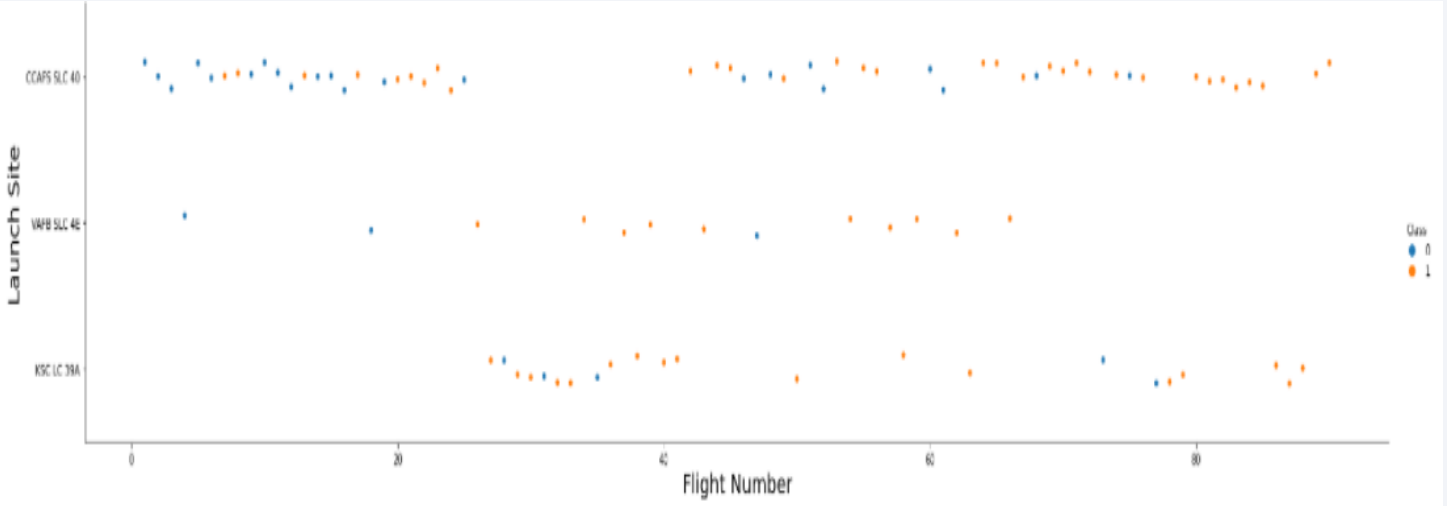

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

- A partir de la gráfica, descubrimos que cuanto mayor sea la cantidad de vuelos en un sitio de lanzamiento, mayor será la tasa de éxito en dicho sitio.

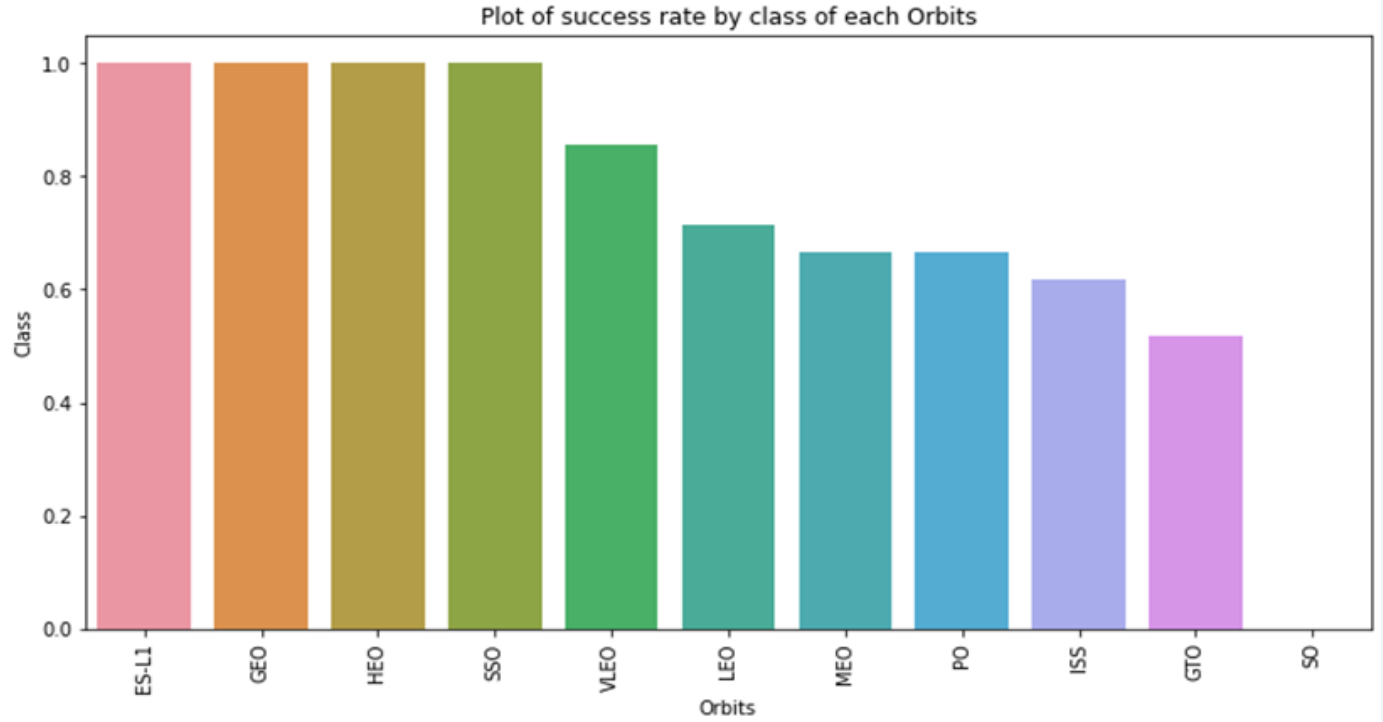


Payload vs. Launch Site



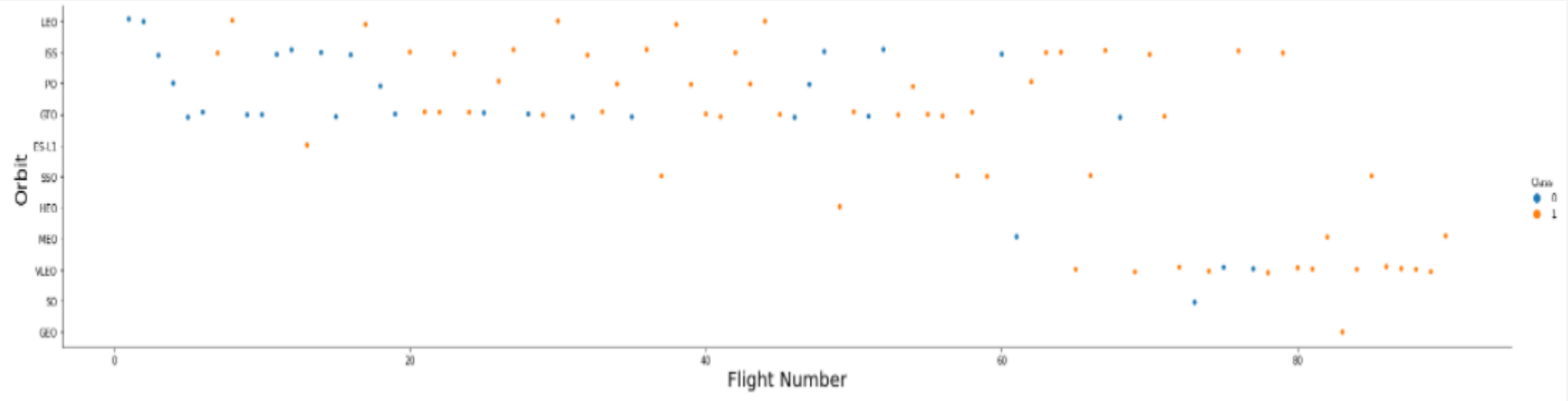
Success Rate vs. Orbit Type

- Del gráfico se desprende que ES-L1, GEO, HEO, SSO y VLEO tuvieron la mayor tasa de éxito.



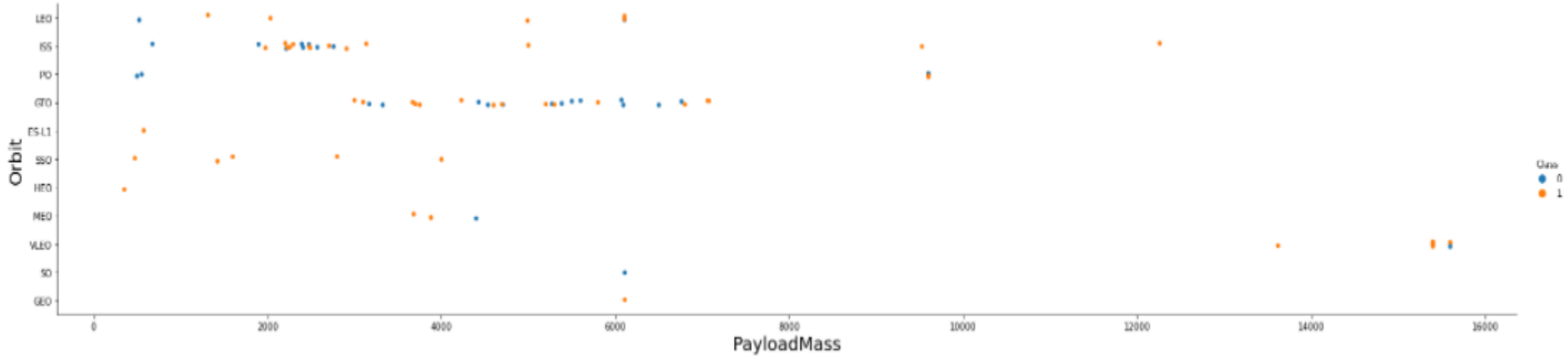
Flight Number vs. Orbit Type

- La gráfica a continuación muestra el número de vuelo en función del tipo de órbita. Observamos que, en la órbita LEO, el éxito está relacionado con el número de vuelos, mientras que en la órbita GTO no existe relación entre el número de vuelo y la órbita.



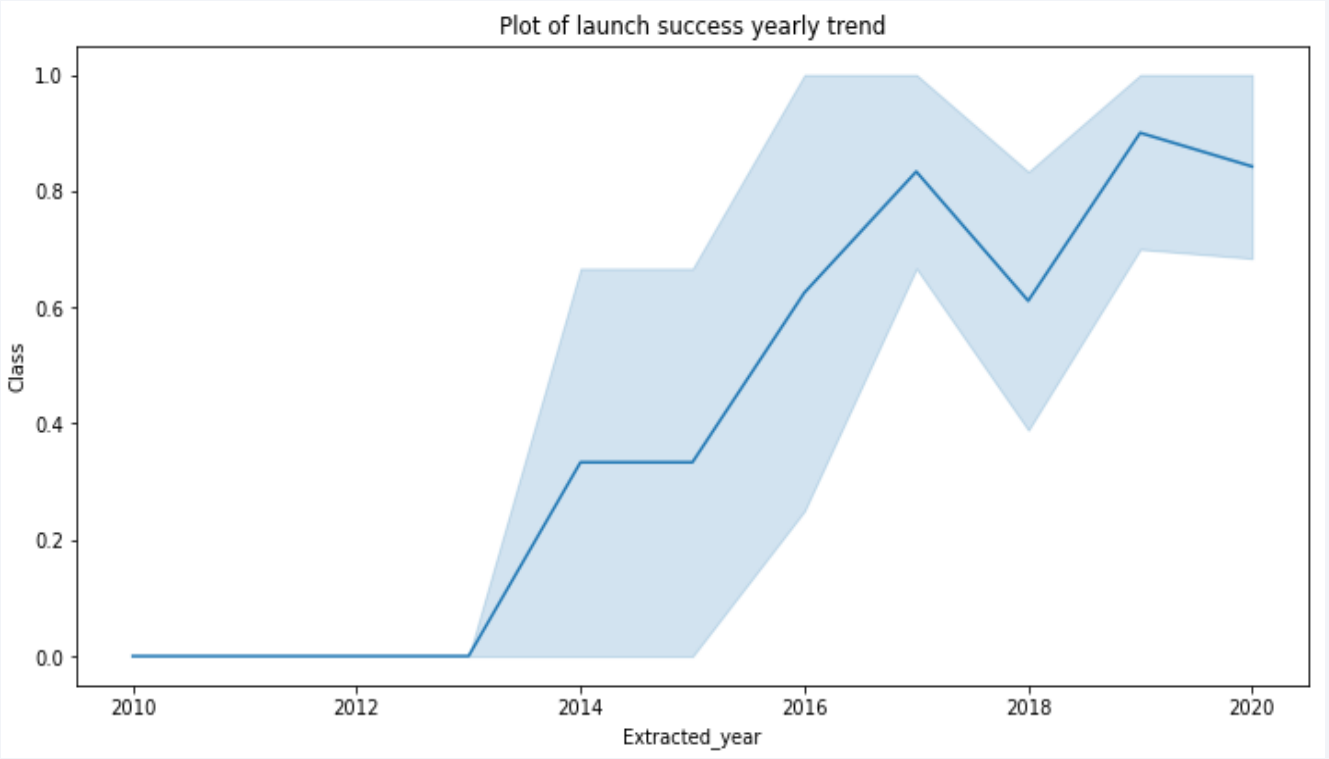
Payload vs. Orbit Type

- Podemos observar que con cargas útiles pesadas, los aterrizajes exitosos son más para órbitas PO, LEO e ISS.



Launch Success Yearly Trend

- Del gráfico se puede observar que la tasa de éxito desde 2013 siguió aumentando hasta 2020.



All Launch Site Names

- Utilizamos la palabra clave DISTINCT para mostrar sólo sitios de lanzamiento únicos de los datos de SpaceX.

Display the names of the unique launch sites in the space mission

In [10]:
```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

- Se Utilizo la consulta anterior para mostrar 5 registros donde los sitios de lanzamiento comienzan con 'CCA'

Display 5 records where launch sites begin with the string 'CCA'

In [11]:
```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	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
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Calculamos la carga útil total transportada por los propulsores de la NASA como 45596 utilizando la consulta a continuación

Display the total payload mass carried by boosters launched by NASA (CRS)

In [12]:
```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

- Se calculo que la masa promedio de carga útil transportada por la versión de refuerzo F9 v1.1 es 2928,4

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

In [13]:
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

- Se observo que la fecha del primer aterrizaje exitoso en la plataforma terrestre fue el 22 de diciembre de 2015.

In [14]:
```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Se utilizo la cláusula WHERE para filtrar los propulsores que aterrizaron exitosamente en el barco no tripulado y aplicamos la condición AND para determinar el aterrizaje exitoso con una masa de carga útil mayor a 4000 pero menor a 6000.

In [15]:
```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- Se uso comodines como '%' para filtrar DÓNDE MissionOutcome fue un éxito o un fracaso

List the total number of successful and failure mission outcomes

In [16]:

```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

Out[16]:

The total number of successful mission outcome is:

	successoutcome
0	100

The total number of failed mission outcome is:

	failureoutcome
0	1

Boosters Carried Maximum Payload

- Se determino el amplificador que transportó la carga útil máxima utilizando una subconsulta en la cláusula WHERE y la función MAX().

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

In [17]:

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

- Se utilizo una combinación de la cláusula WHERE y las condiciones LIKE, AND y BETWEEN para filtrar los resultados de aterrizajes fallidos en naves no tripuladas, sus versiones de refuerzo y los nombres de los sitios de lanzamiento para el año 2015.

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

In [18]:
```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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))

In [19]:
```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

- Se seleccionaron los resultados de aterrizaje y el COUNT de los resultados de aterrizaje de los datos y usamos la cláusula WHERE para filtrar los resultados de aterrizaje BETWEEN del 04/06/2010 al 20/03/2010. Aplicamos la cláusula GROUP B para agrupar los resultados de aterrizaje y la cláusula ORDER BY para ordenarlos en orden descendente.

Section 3

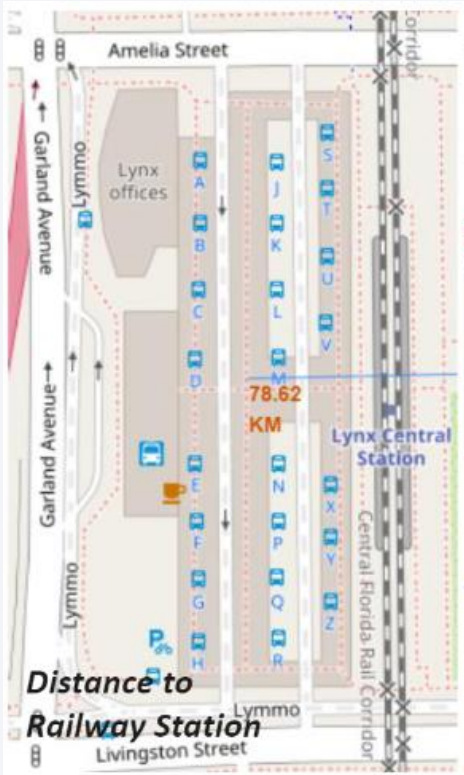
Launch Sites Proximities Analysis

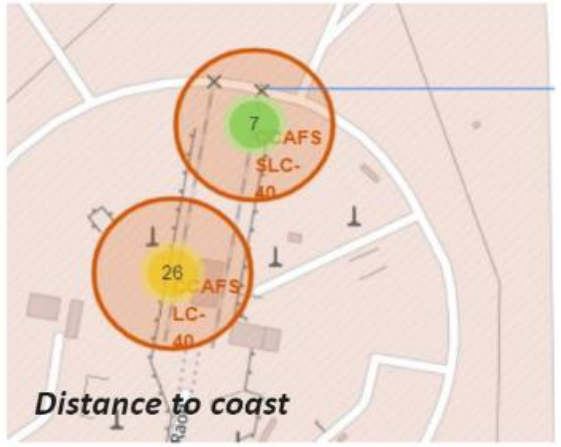
Sitios de lanzamiento

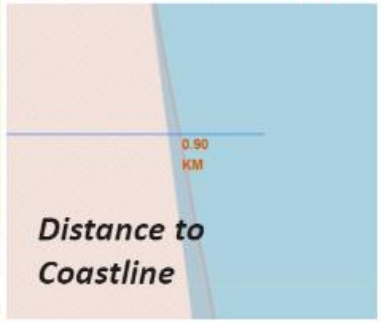
We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

Sitios de lanzamiento Florida

Distancia sitios de lanzamiento







- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

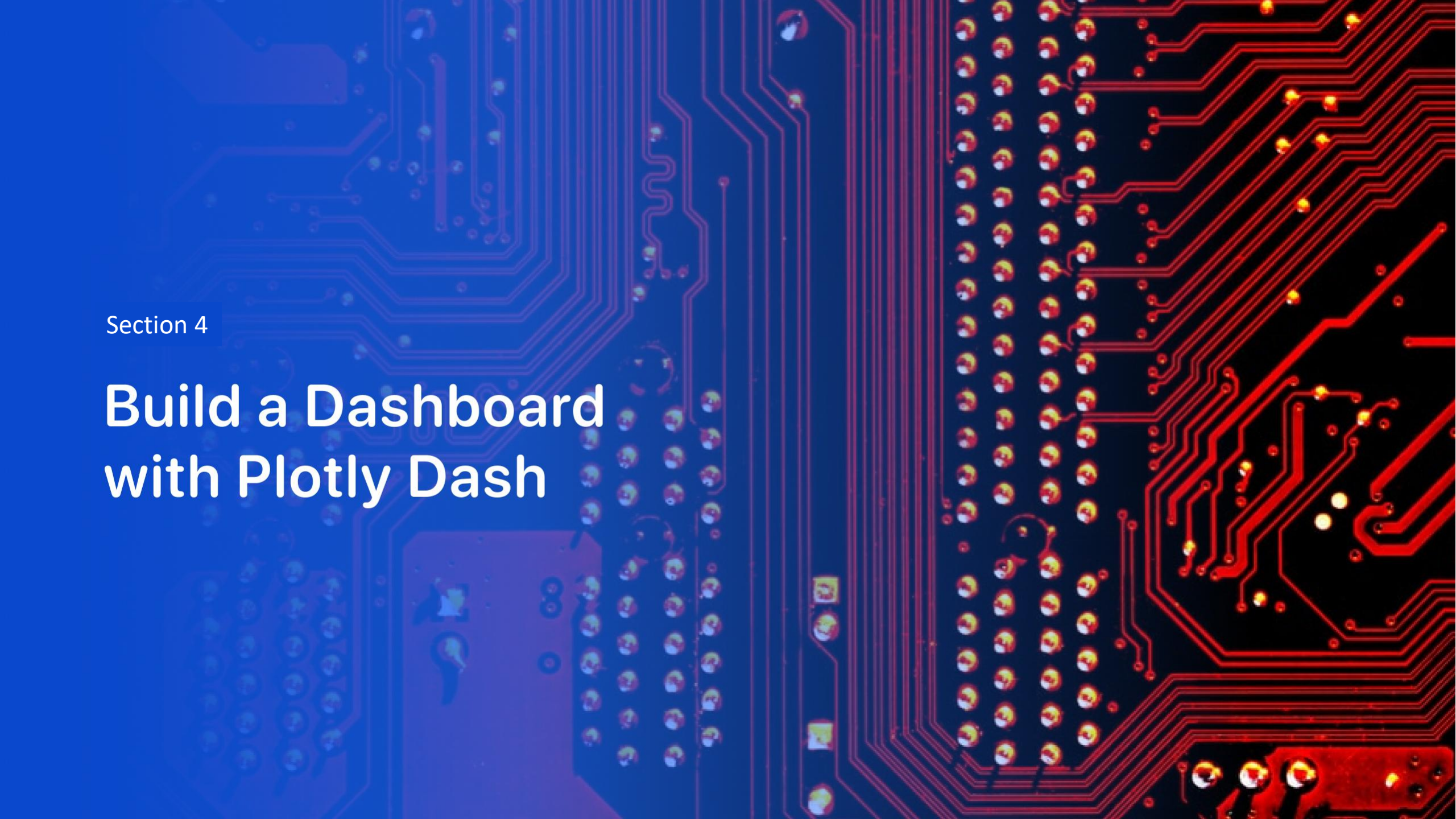
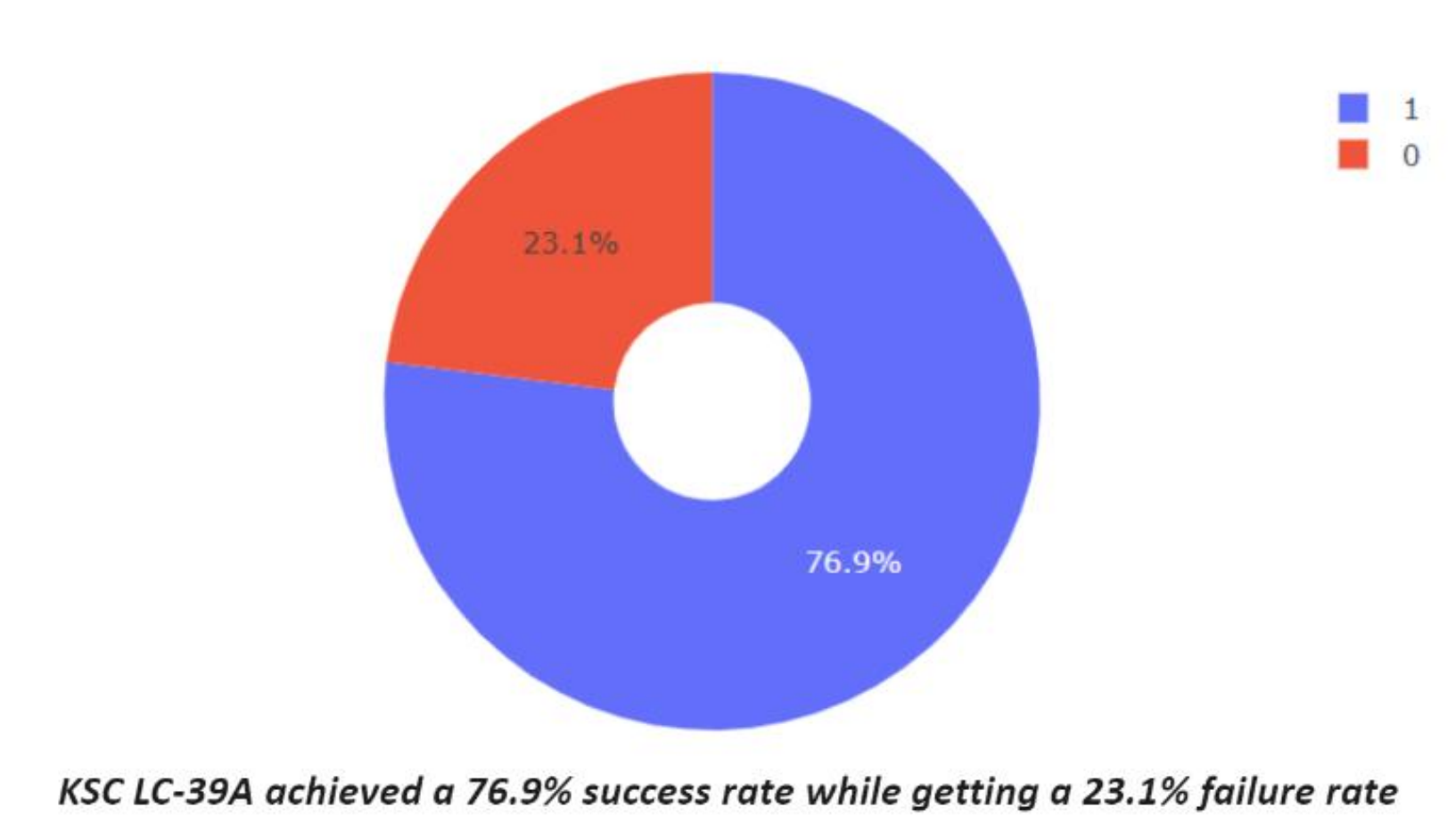

Section 4
Build a Dashboard with Plotly Dash

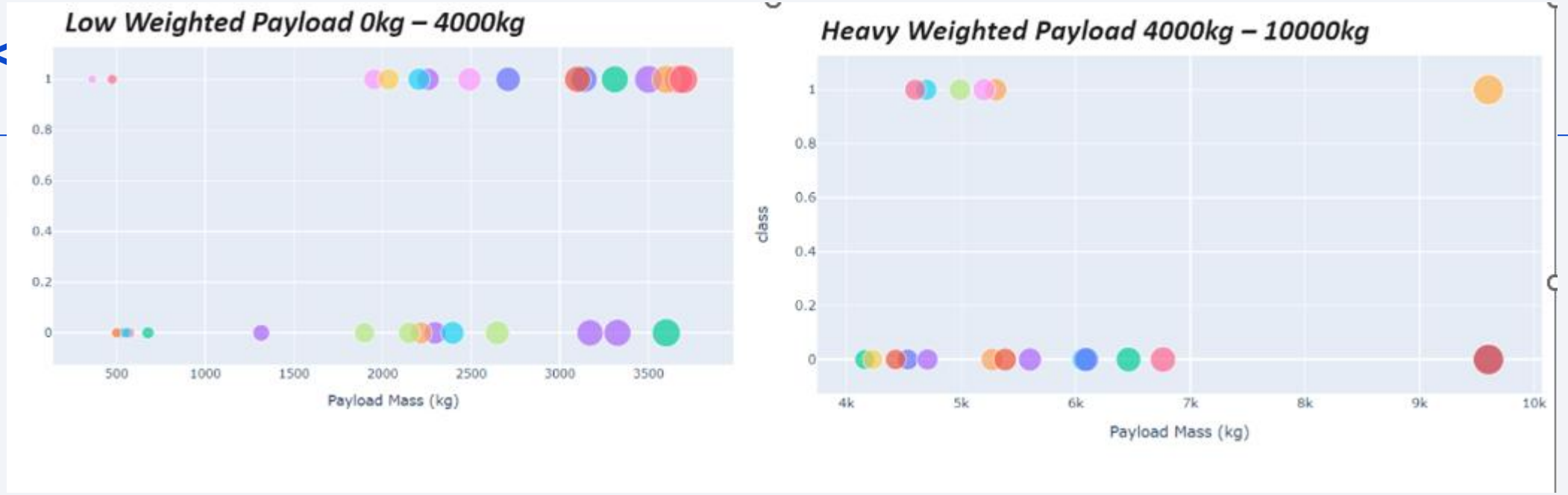
Grafico circular de lanzamientos

- Podemos ver que el KSC LC-39A es el que mas lanzaminetos exitos tiene

Gráfico circular que muestra el sitio de lanzamiento con la mayor tasa de éxito de lanzamiento



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate



- Podemos ver que la tasa de éxito de los payloads de bajo peso es mayor que la de las payloads de alto peso.

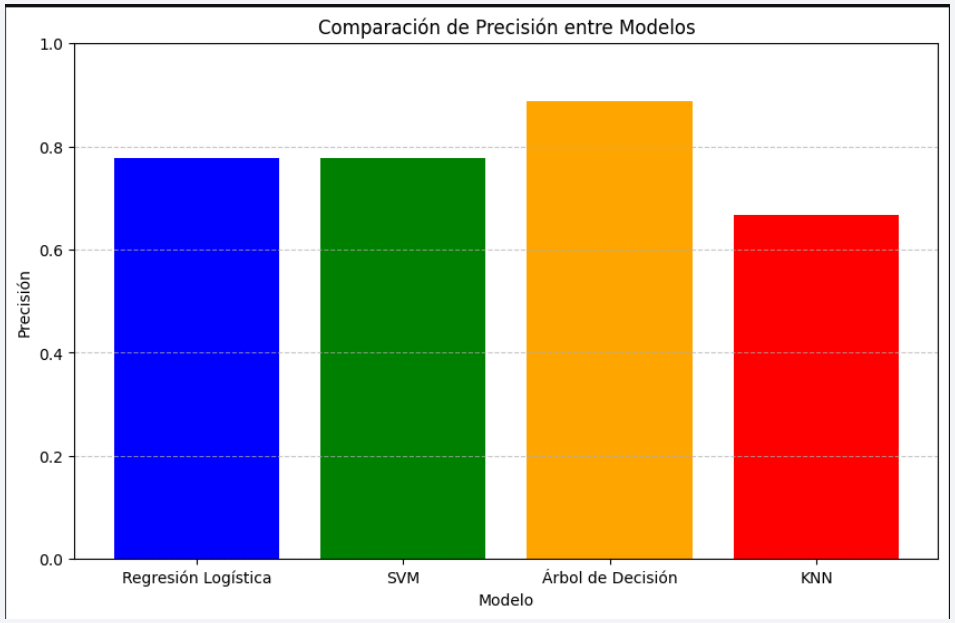
Section 5

Predictive Analysis (Classification)

Classification Accuracy

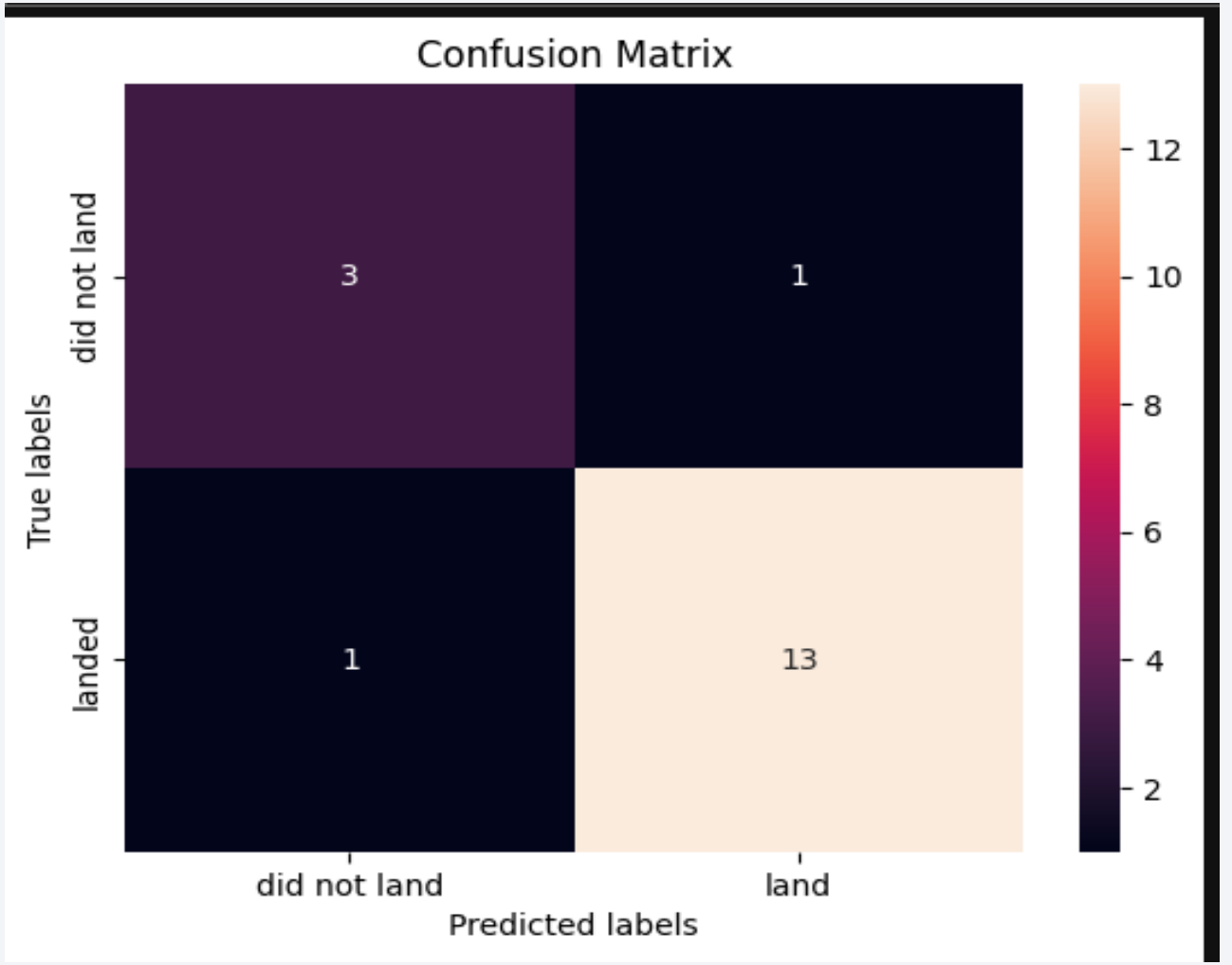
- El modelo de arbol de decision fue el modelo con mayor precision

```
Comparación de precisión de modelos:
Regresión Logística: 0.7778
SVM: 0.7778
Árbol de Decisión: 0.8889
KNN: 0.6667
```



Confusion Matrix

- La matriz de confusión del clasificador de árbol de decisión muestra que este puede distinguir entre las diferentes clases.
- El principal problema reside en los falsos positivos, es decir, los aterrizajes fallidos que el clasificador marca como exitosos.



Conclusions

- Point 1
- Point 2
- Point 3
- Point 4
- …

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!