IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ho Thi Quynh Van
09/06/2024

Outline

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

Executive Summary

- Methodologies
  - Data collection and data wrangling
  - EDA with Data visualization and with SQL
  - Building interactive map with Folium and building Dashboard
  - Predictive Analysis
- Summary of results
  - EDA
  - Interactive analysis
  - Predictive analysis

Introduction

- ## Context

SpaceX offers Falcon 9 rocket launches on its website for 62 million dollars; other companies charge up to 165 million dollars apiece; much of the savings is due to SpaceX's ability to reuse the first stage. If we can predict whether the first stage will land, we can estimate the cost of a launch. This data can be utilized if another firm wishes to compete with SpaceX for a rocket launch.

- ## Problems

Using classification algorithms to predict whether the first stage will land successfully or not

Section 1

Methodology

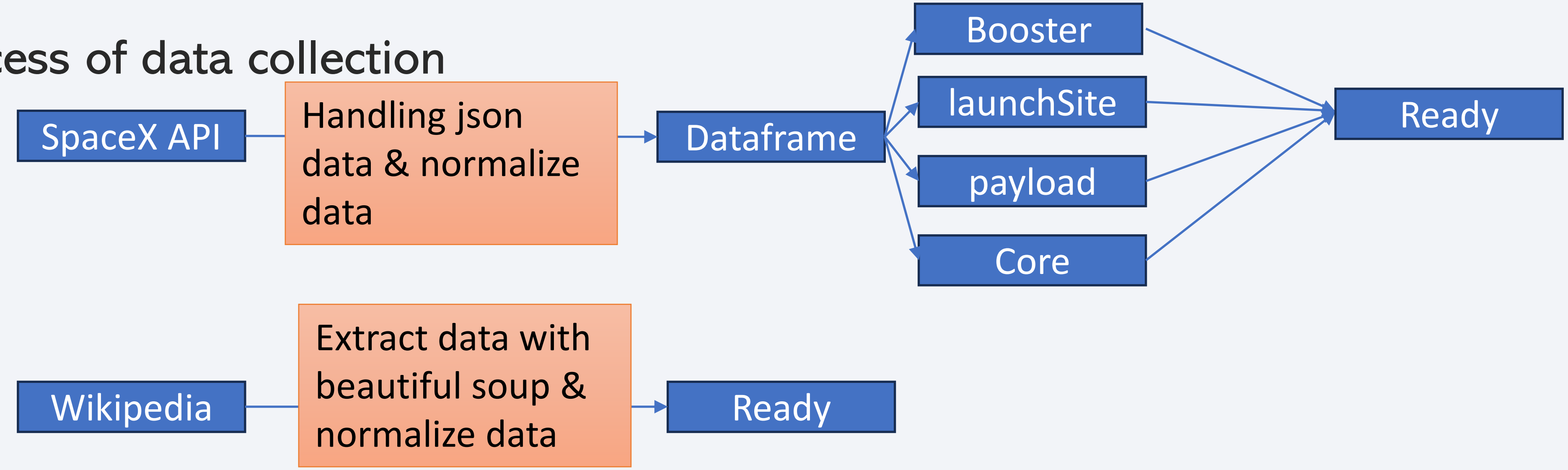
Methodology

Executive Summary

- Data collection methodology:
    - SpaceX API
    - Scraping from Wikipedia
- Perform data wrangling
    - One hot encoding features, handling null values and unused columns
- 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, KNN, SVM, Decision tree

Data Collection

- Data is collected from two sources:
  - SpaceX API: data of launches including rocket, payload, launch specification, landing specification and landing result
  - Wikipedia: Other data
- Process of data collection



Data Collection – SpaceX API

- https://github.com/quynhvan16/Winning-space-race-with-Data-Science-/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

```
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())
```

```
BoosterVersion
```

```
[]
```

Now, let's apply `getBoosterVersi`

```
# Call getBoosterVersion
getBoosterVersion(data)
```

the list has now been update

```
BoosterVersion[0:5]
```

```
['Falcon 1', 'Falcon 1', 'Falc
```

we can apply the rest of the functions

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

```
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}
```

Then, we need to create a Pandas data frame from the dictionary launch_di

```
# Create a data from launch_dict
data = pd.DataFrame(launch_dict)
```

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data[data.BoosterVersion == "Falcon 9"]
```

Data Collection - Scraping

- https://github.com/quynhvan16/Winning-space-race-with-Data-Science-/blob/main/jupyter-labs-webscraping.ipynb

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

```python
# 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")
print(html_tables)
```

```python
column_names = []

# Apply find_all() function with `th` element on
# Iterate each th element and apply the provided
# Append the Non-empty column name (`if name is
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

```python
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

```python
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
```

Data Wrangling

- https://github.com/quynhvan16/Winning-space-race-with-Data-Science-/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

1.

```
# 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
```

2.

```
# 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
```

3.

```
# landing_outcomes = values on Outcome column
landing_outcomes = df.Outcome.value_counts()
landing_outcomes
```

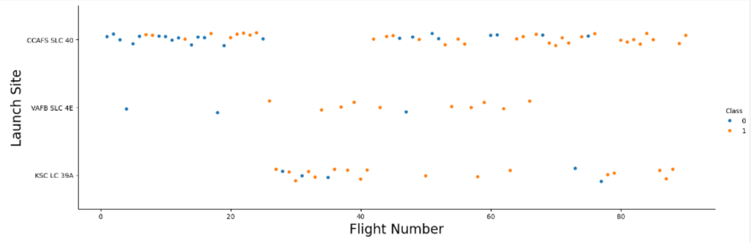
```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

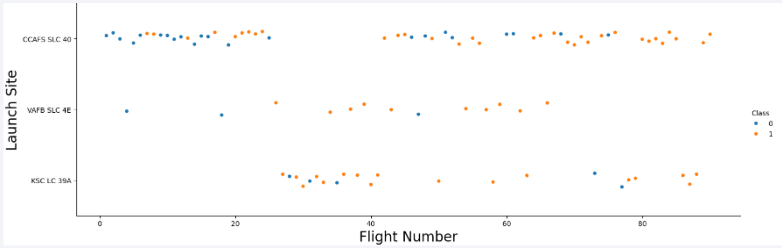
4.

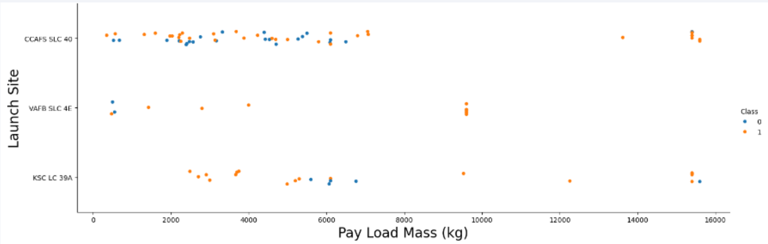
```
# landing_class = 0 if bad_outcome
landing_class = [0 if x in bad_outcomes else 1 for x in df['Outcome']]
# landing_class
df['Class']=landing_class
print(df[['Class']].head(8))
print(df["Class"].mean())  # probability of positive outcome 2/3
print(df.head(5))
# landing_class = 1 otherwise
```

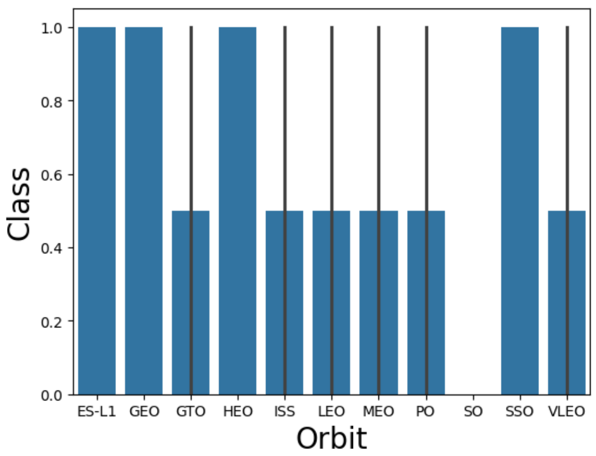
EDA with Data Visualization

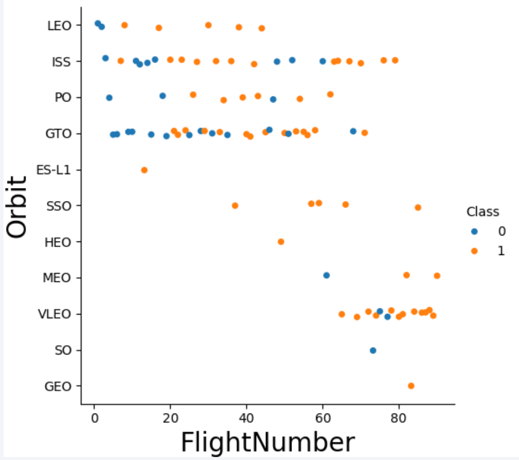
https://github.com/quynhvan16/Winning-space-race-with-Data-Science-/blob/main/edadataviz.ipynb

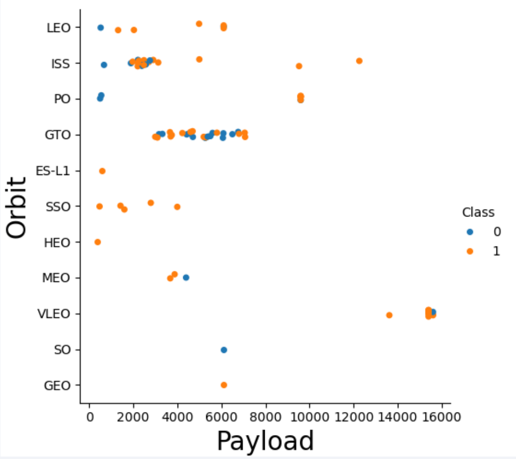


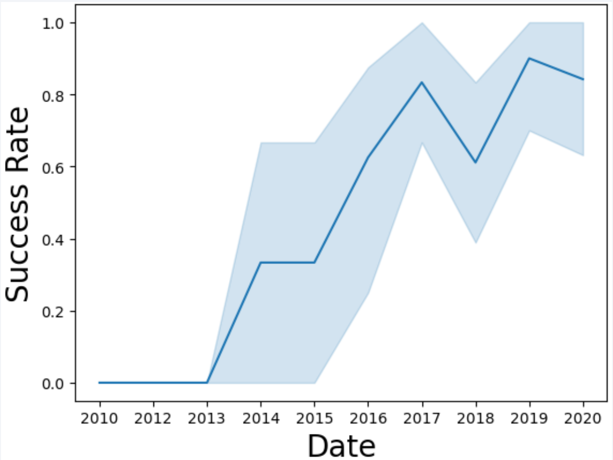












EDA with SQL

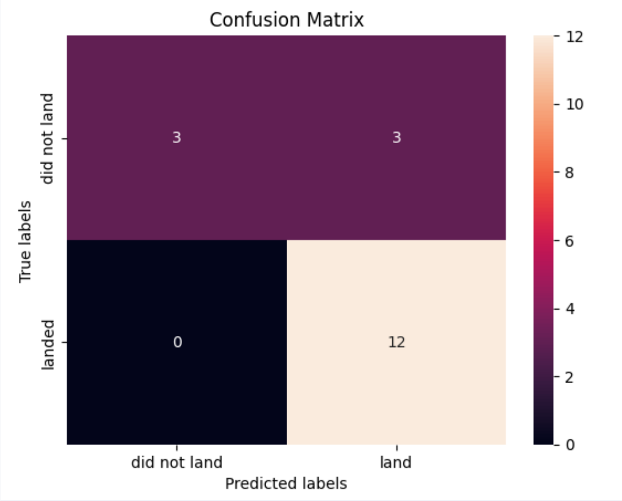
- SQL queries
  - %sql select distinct LAUNCH_SITE from SPACEXTBL
  - %sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
  - %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'
  - %sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
  - %sql select min(DATE) from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'%sql select Booster_Version from SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
  - %sql select count(Mission_Outcome) from SPACEXTBL WHERE Mission_Outcome = 'Success' or Mission_Outcome = 'Failure (in flight)'
  - %sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
  - %sql SELECT SUBSTR(Date,6,2) AS Month, Booster_Version, Launch_site FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Failure%drone%' AND SUBSTR(Date,0,5) = '2015'
  - %sql SELECT Landing_Outcome, COUNT(*) AS Numbers FROM SPACEXTBL WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY Numbers DESC;
- https://github.com/quynhvan16/Winning-space-race-with-Data-Science-/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

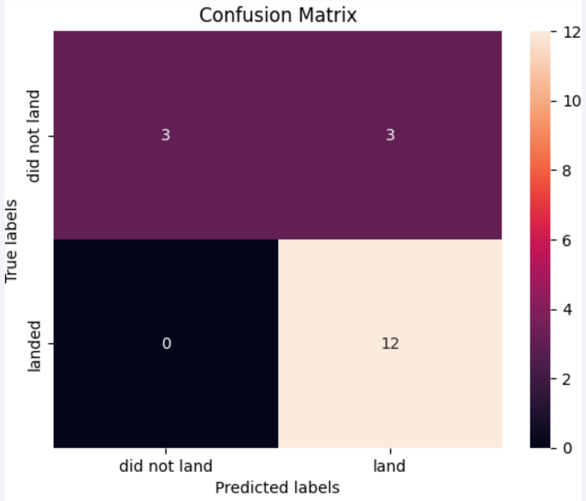
- https://github.com/quynhvan16/Winning-space-race-with-Data-Science-/blob/main/lab_jupyter_launch_site_location.ipynb

Predictive Analysis (Classification)

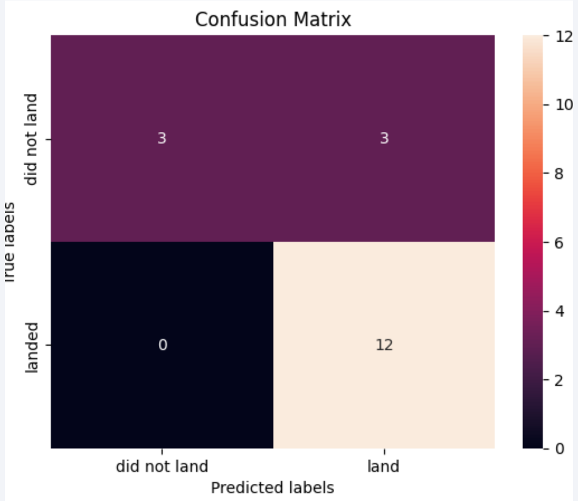
- The model with best out of sample accuracy were KNN, logistic regression and SVM with accuracy around 0.83
- https://github.com/quynhvan16/Winning-space-race-with-Data-Science-/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb



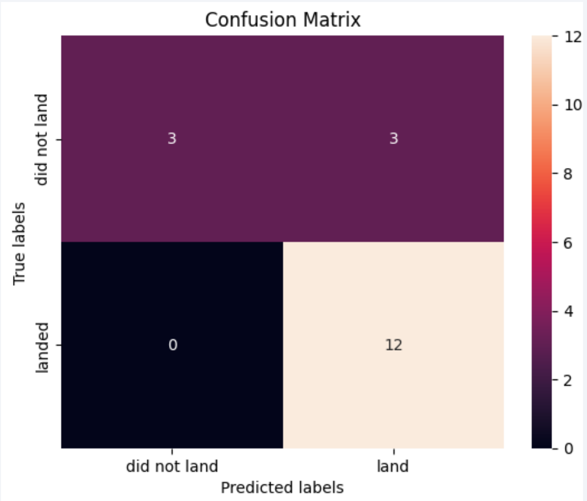
Logistic regression



SVM



Tree



KNN

Results

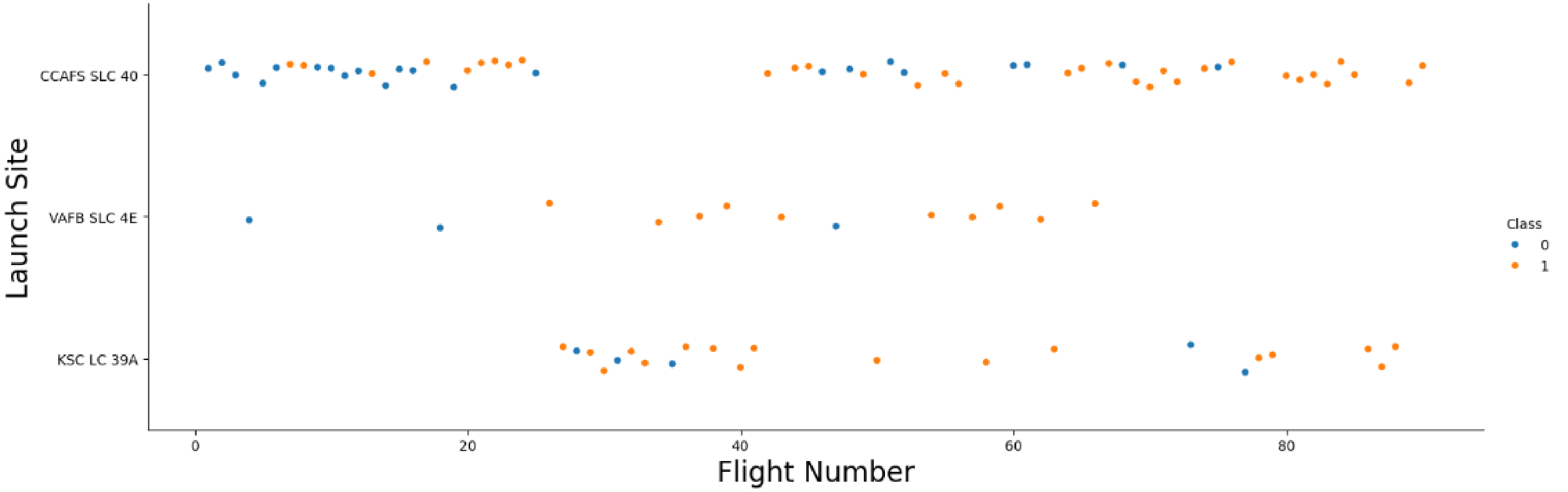
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

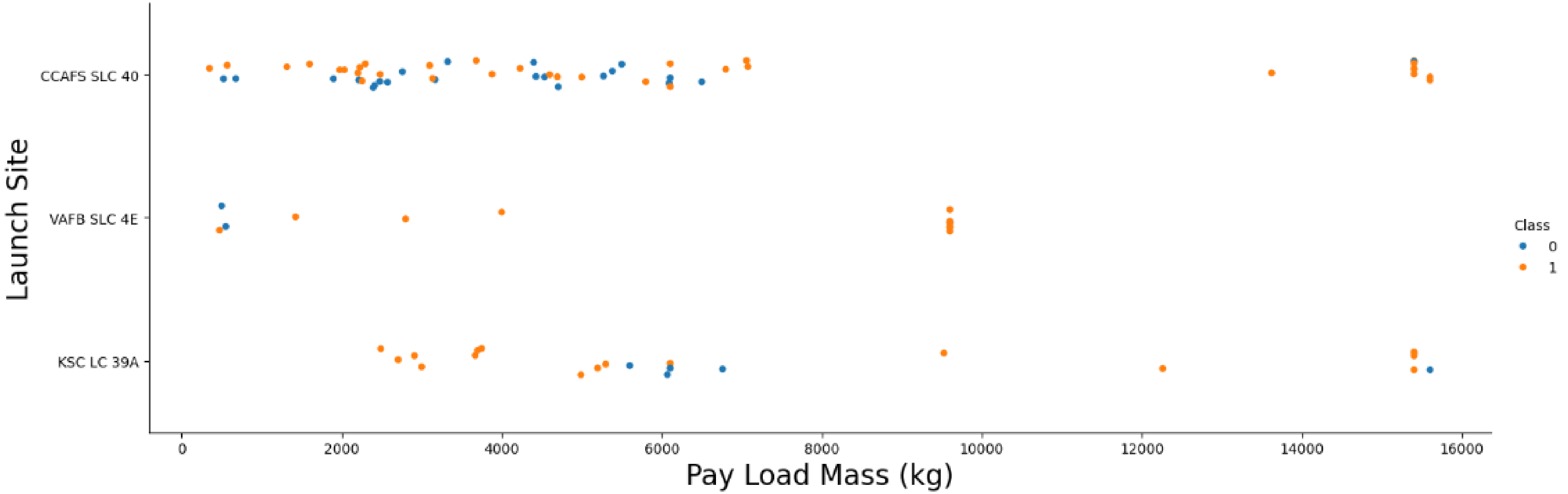
```
### TASK 1: Visualize the relationship between Flight Number and Launch Site
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 3)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



The launch site CCAFS SLC 40 had a higher opportunity of success, when the payload mass was lower.

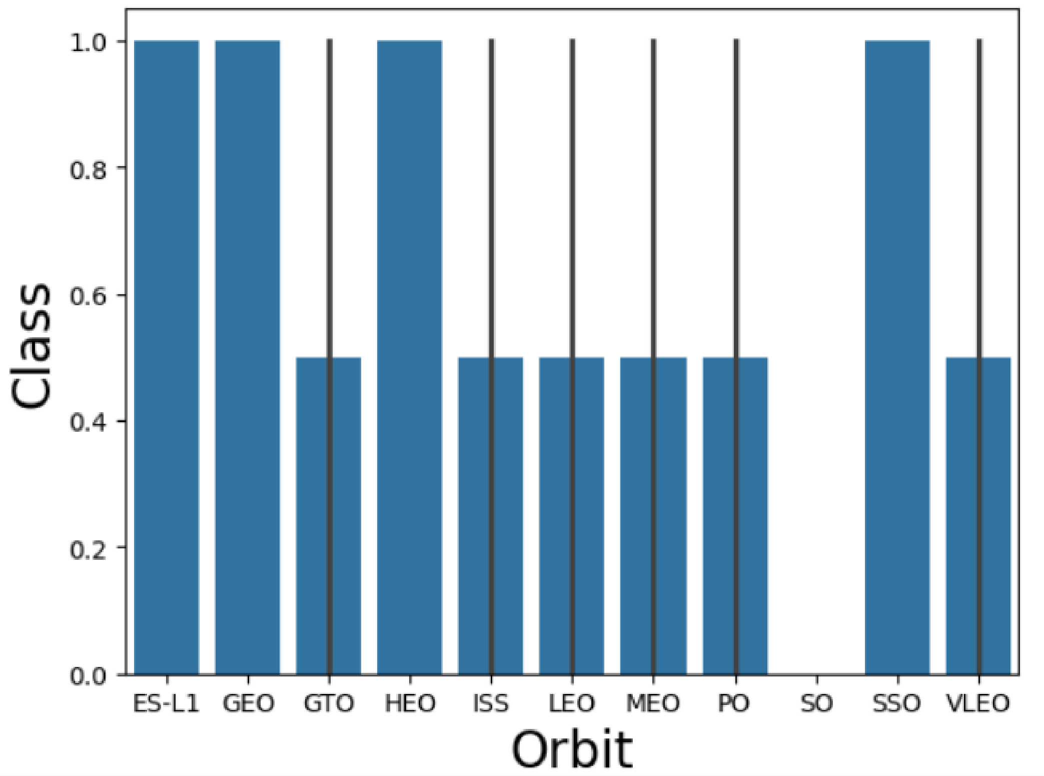
Payload vs. Launch Site

```
### TASK 2: Visualize the relationship between Payload and Launch Site
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 3)
plt.xlabel("Pay Load Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Success Rate vs. Orbit Type

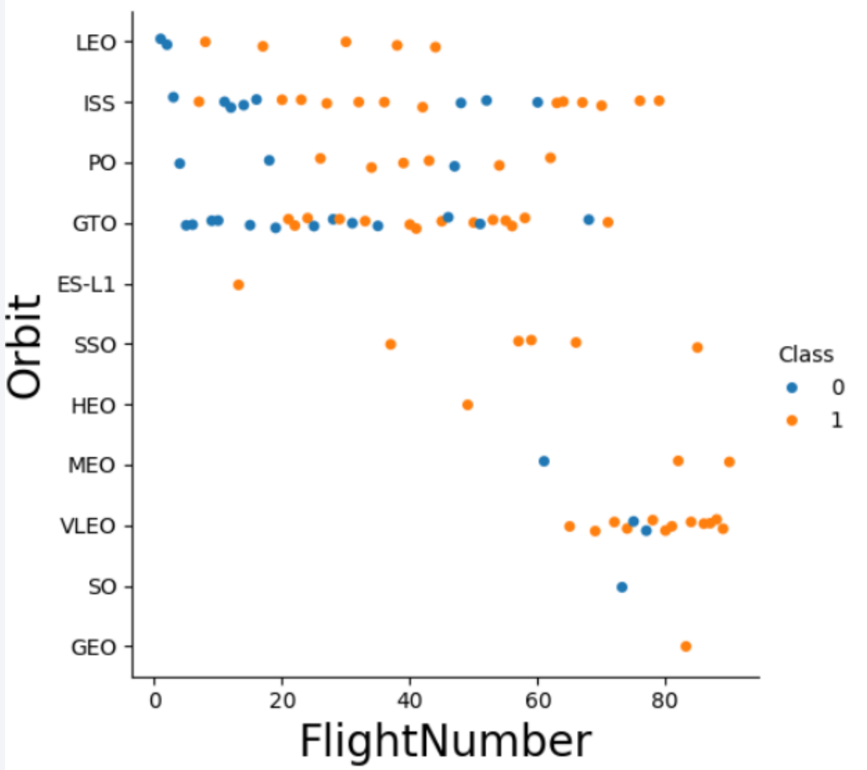
```
# HINT use groupby method on Orbit column and get the mean of Class column
t = df.groupby(['Orbit', 'Class'])['Class'].agg(['mean']).reset_index()
sns.barplot(y="Class", x="Orbit", data=t)

plt.xlabel("Orbit",fontsize=20)
plt.ylabel("Class",fontsize=20)
plt.show()
```



SSO, HEO, ES-L1 and GEO orbits have a higher success possibility

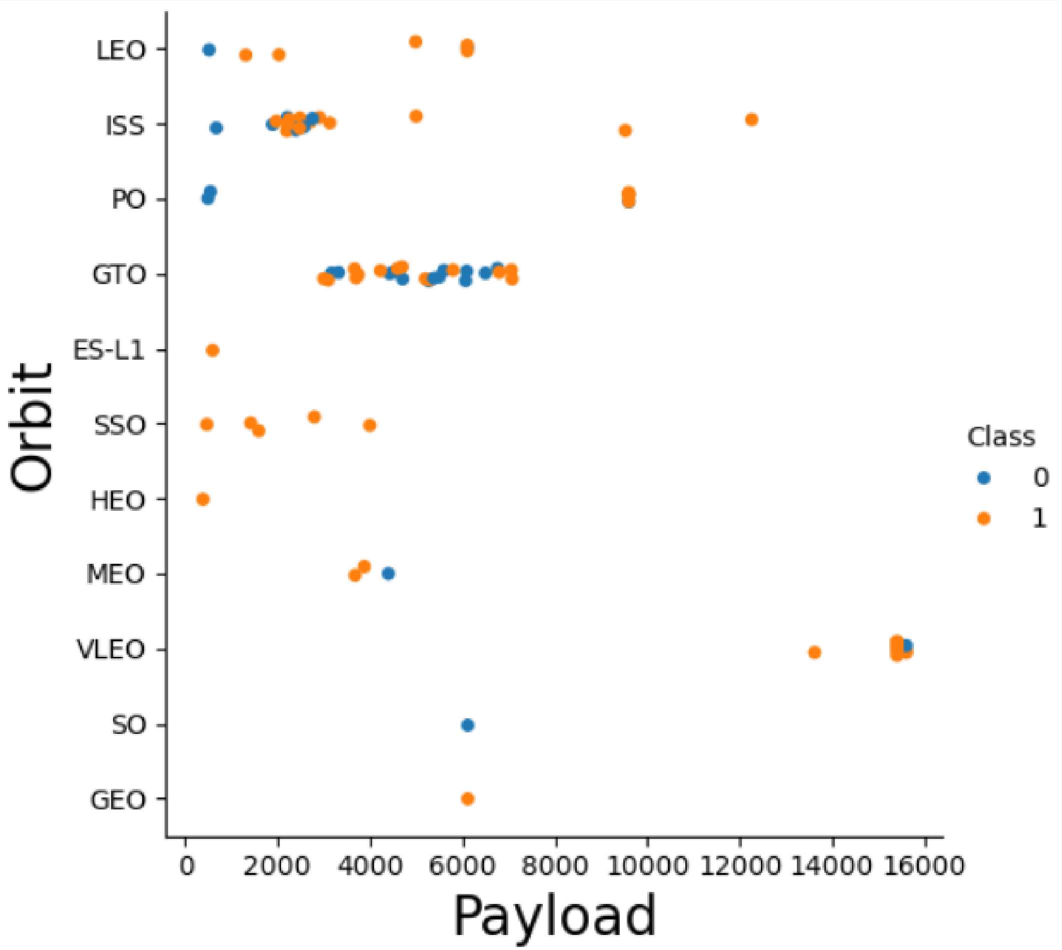
Flight Number vs. Orbit Type

```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df)
plt.xlabel("FlightNumber",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



For the LEO orbit, the higher the flight number, the higher chance of success, whereas there is no relationships between the flight number and success rate for other orbits

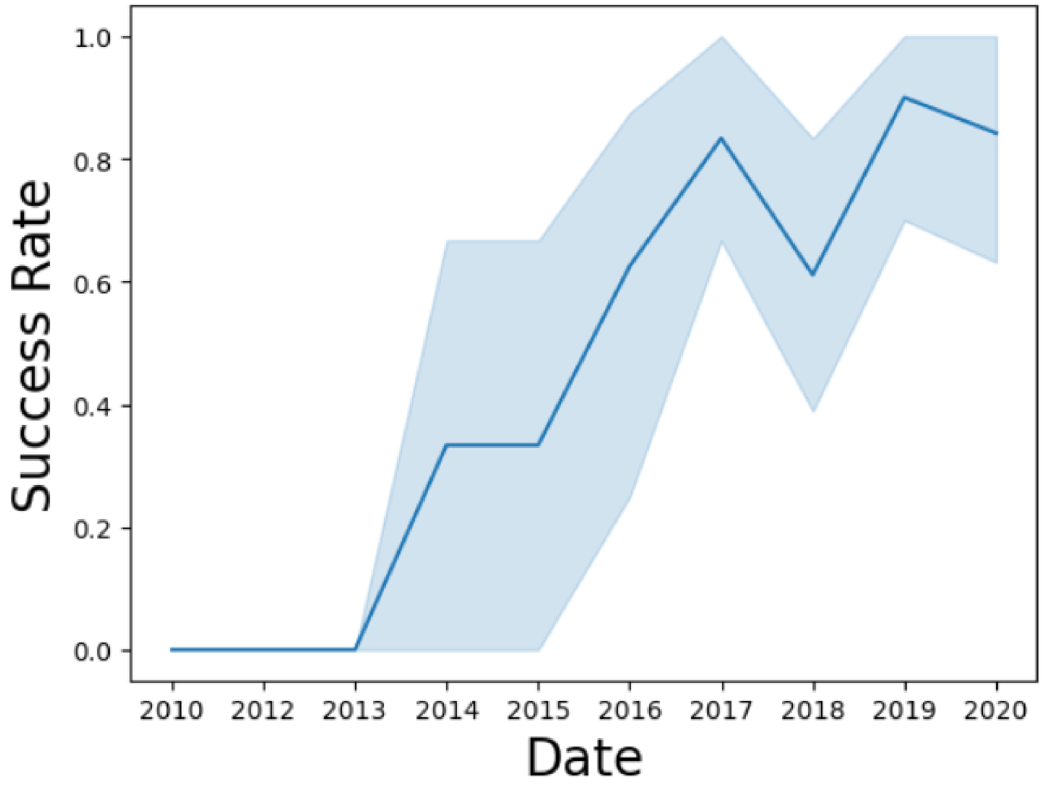
Payload vs. Orbit Type



Heavy payloads have a impact negatively on GTO orbits and positively impact on ISS and LEO orbits.

Launch Success Yearly Trend

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
sns.lineplot(data=df, x="Date", y="Class")
plt.xlabel("Date",fontsize=20)
plt.ylabel("Success Rate",fontsize=20)
plt.show()
```



The sucess rate kept growing from 2013 to 2020

All Launch Site Names

In [13]:
```
%sql select distinct LAUNCH_SITE from SPACEXTBL
```

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

Out[13]:

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

Launch Site Names Begin with 'CCA'

```
%sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
```

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

[No Title]

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'
```

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

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
```

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

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

```
%sql select min(DATE) from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
```

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

min(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

In [19]:
```
%sql select Booster_Version from SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)'
and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
```

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

Out[19]:

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

Total Number of Successful and Failure Mission Outcomes

```
%sql select count(Mission_Outcome) from SPACEXTBL WHERE Mission_Outcome = 'Success' or Mission_Outcome = 'Failure (in flight)'
```

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

count(Mission_Outcome)
99

Boosters Carried Maximum Payload

```
%sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

Booster_Version
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

2015 Launch Records

```
%sql SELECT SUBSTR(Date,6,2) AS Month, Booster_Version, Launch_site FROM SPACEXTBL
WHERE Landing_Outcome LIKE 'Failure%drone%' AND SUBSTR(Date,0,5) = '2015'
```

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

Month	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

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

```
%sql SELECT Landing_Outcome, COUNT(*) AS Numbers FROM SPACEXTBL WHERE
Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY Numbers DESC;
```

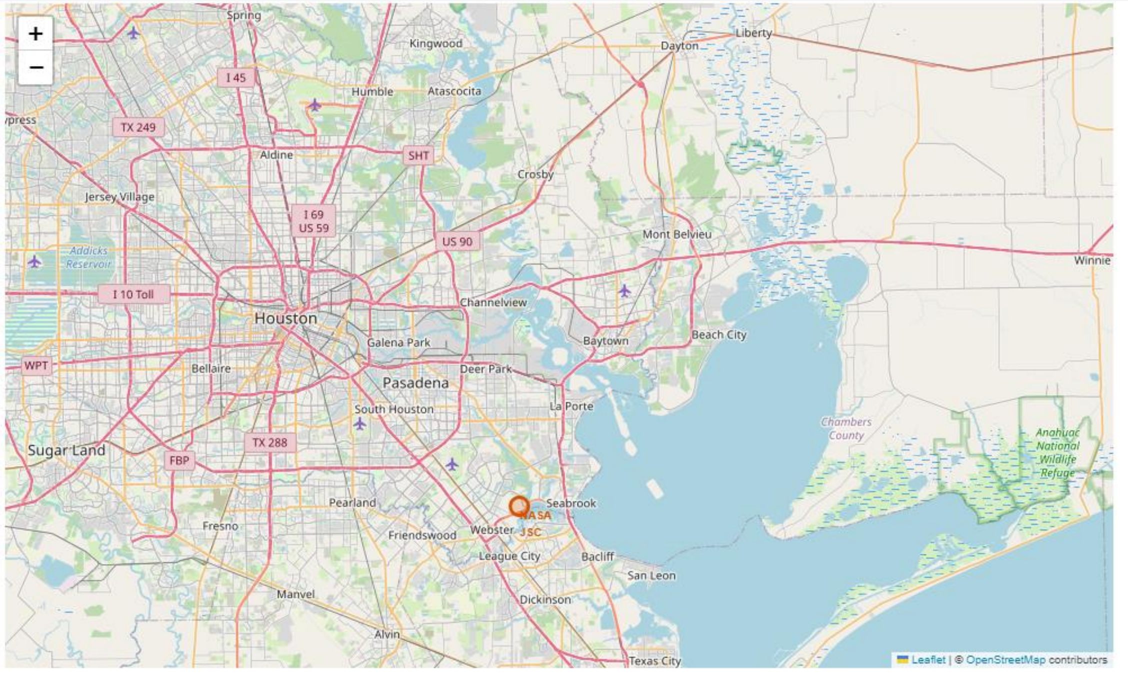
```
 * sqlite:///my_data1.db
Done.
```

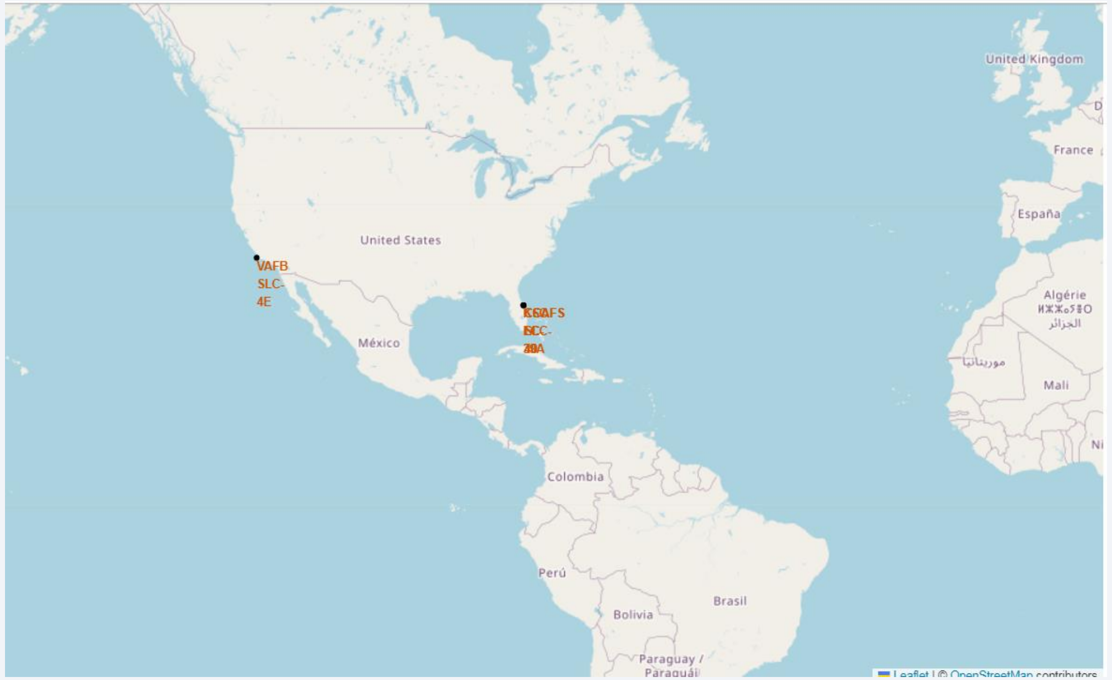
Landing_Outcome	Numbers
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

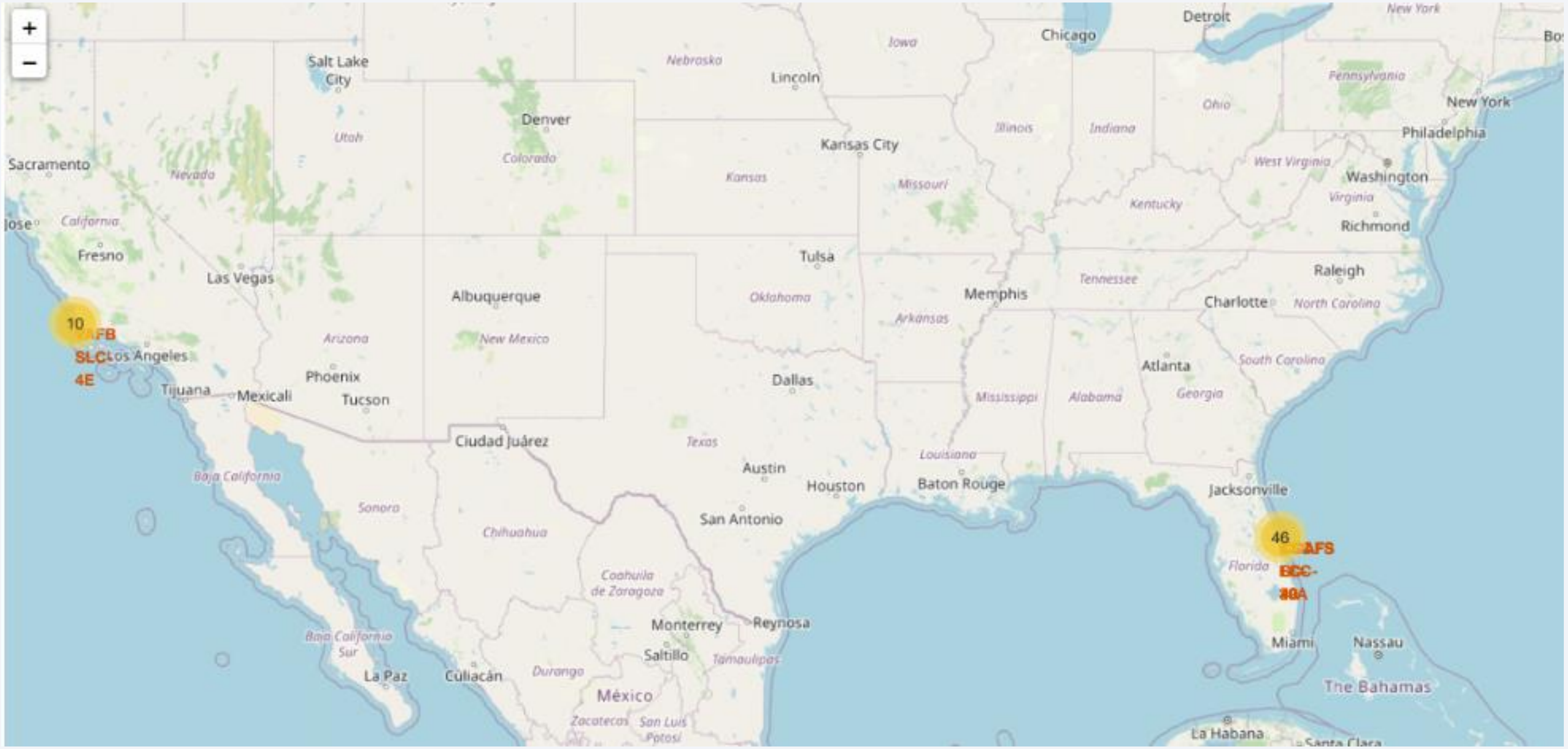
Launch Sites Proximities Analysis

<Folium Map Screenshot 1>





<Folium Map Screenshot 2>

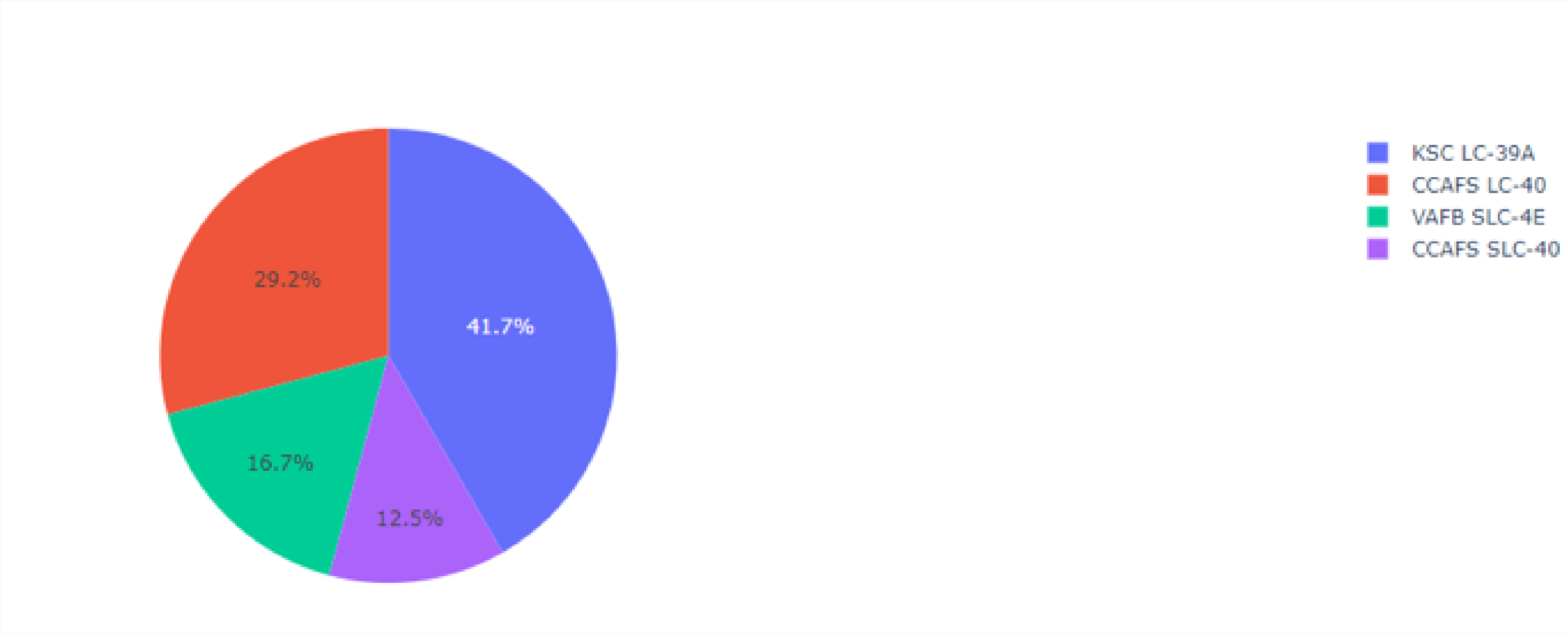


<Folium Map Screenshot 3>

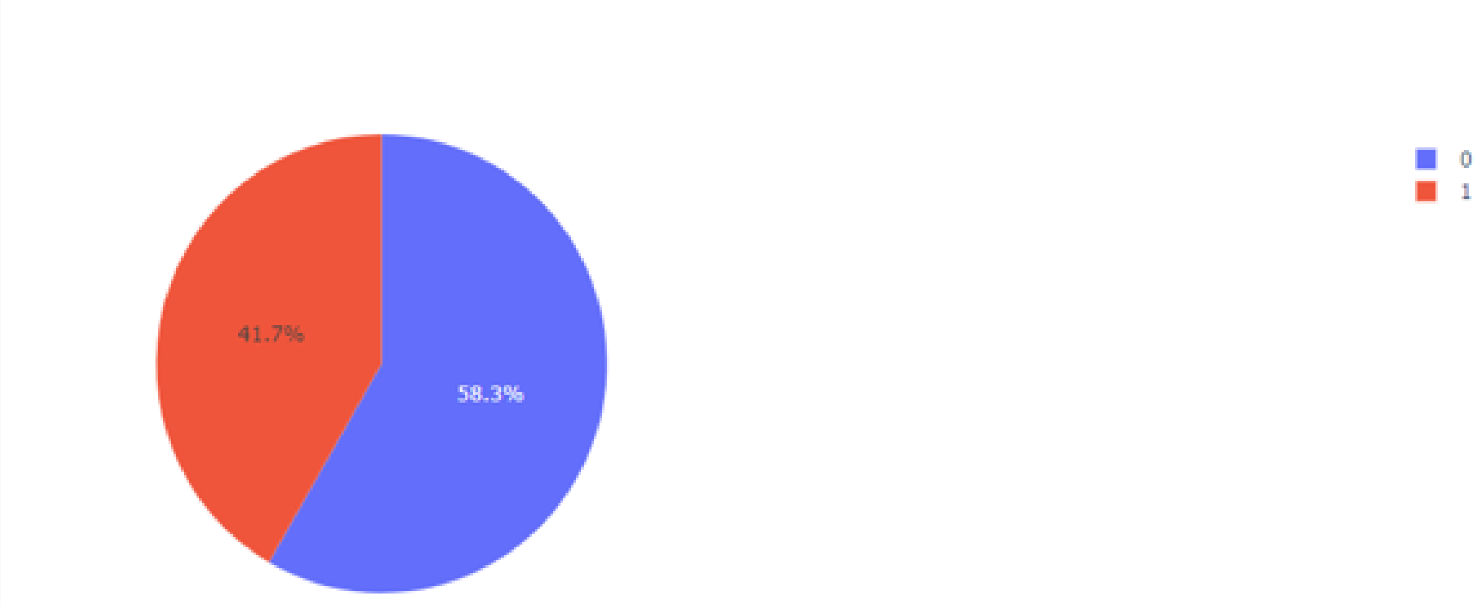
Section 4

Build a Dashboard with Plotly Dash

<Dashboard Screenshot 1>



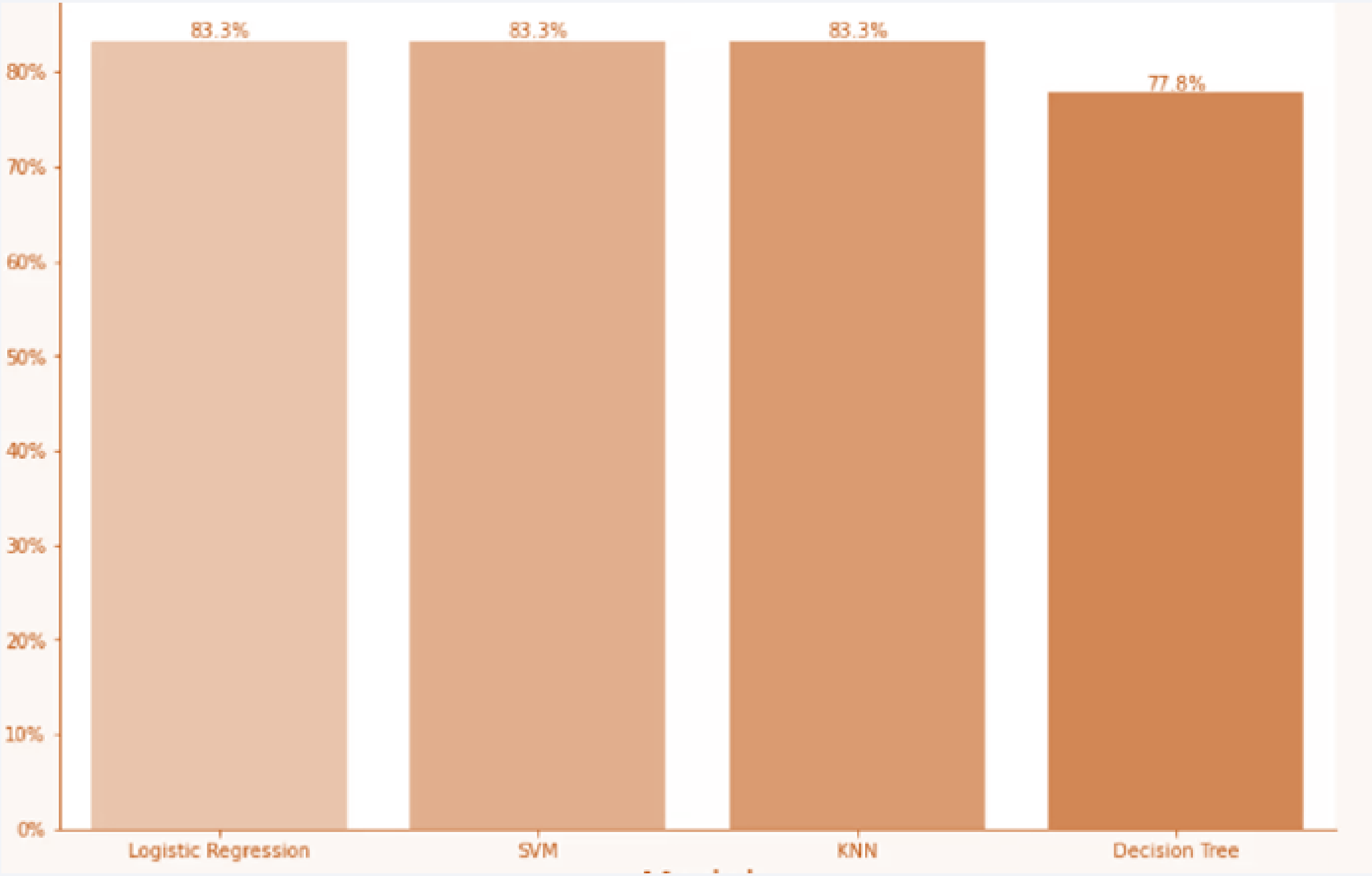
<Dashboard Screenshot 2>



Section 5

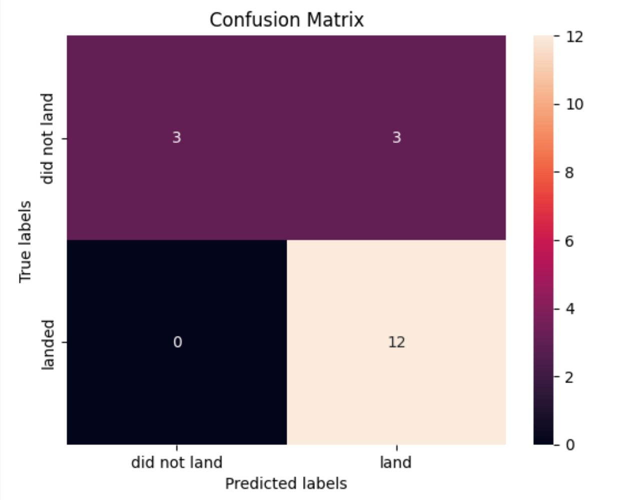
Predictive Analysis (Classification)

Classification Accuracy

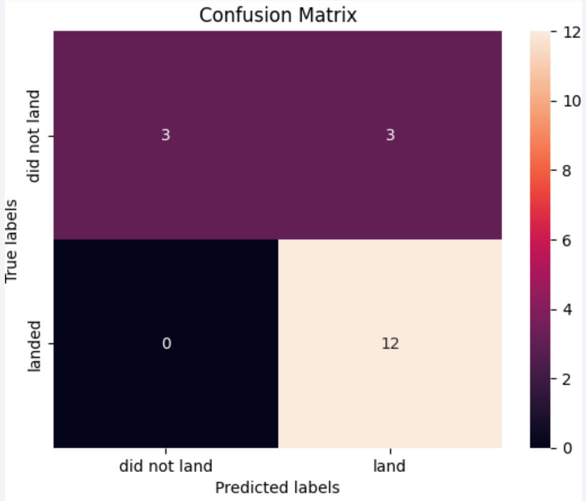


Confusion Matrix

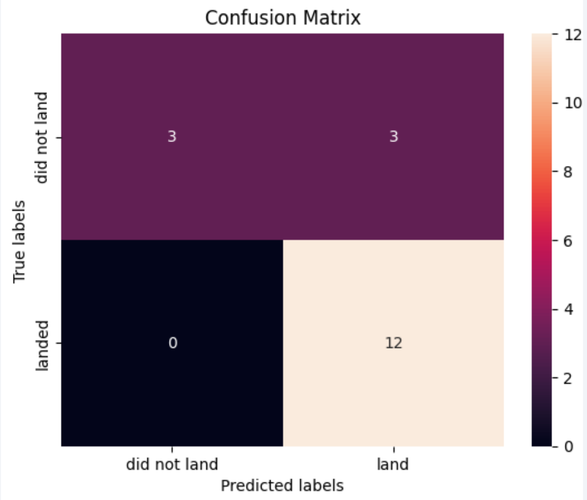
The best performing models are logistic regression, KNN and SVM. These models delivered an accuracy of 83% on test data



Logistic regression



SVM



KNN

Conclusions

- The quantity of successful launches grows yearly
- KSC LC-39A had the highest success rate among all sites
- The success rate was relied on the orbit and payload mass, ISS and VLEO orbits had a good success rate.
- SVM, logistic regression and KNN are best models to predict if the stage one would land or not, it had an accuracy of 83%
- Space x is leading the space race

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!