IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Jianfeng LIU

2024-01-26

Outline

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

Executive Summary

- Develop Python code to manipulate data, Utilize data science methodologies to define and formulate a real-world business problem. Using Python to create plots or charts to visualize the collected data.
- To make a better view, build an interactive dashboard by Python. e.g., generate interactive maps, plot coordinates and mark clusters.
- Using machine learning to determine if the first stage of Falcon 9 will land successfully. Mainly used ’sklearn’ package to train different classification models.

- After selecting the best hyperpa83.33% accuracyrameters for the decision tree classifier using the collected and validated data has been achieved.

Introduction

We will predict if the Falcon 9 first stage will land successfully. 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. In this lab, you will collect and make sure the data is in the correct format from an API. The following is an example of a successful and launch.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - With using *request* and *beautifulsoup* to get the '.json' data, afterwards is converted a Python Pandas data frame.
- Perform data wrangling
  - Perform some Exploratory data analysis (EDA) to determine what would be the label for training models.
- Perform exploratory data analysis using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection and Data Wrangling

- JSON file: `https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json`

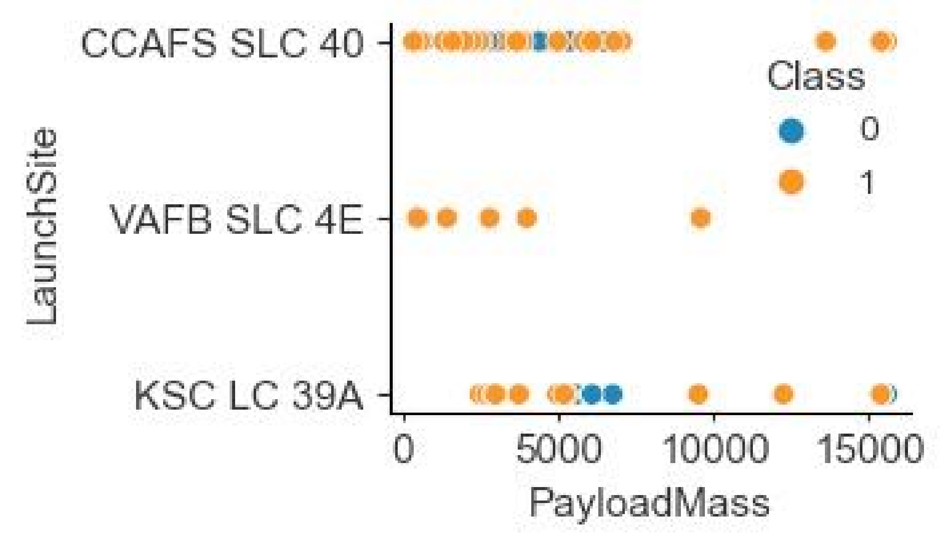
"requests.get(url)" " BeautifulSoup(content)" " pandas.DataFrame( )"

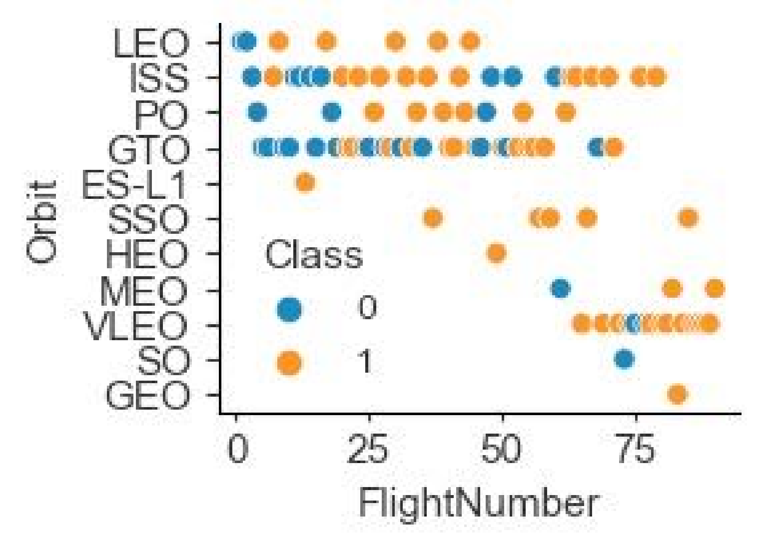
Data Online → Request Data → Data Extraction → Data Convert

- https://github.com/andyandhope/Apllied-data-science-capstone/blob/main/01jupyter-labs-spacex-data-collection-api.ipynb

EDA with Data Visualization

- Data is a Pandas dataframe
- With using Matplotlib and Seaborn, which are popular used in data visualization
  - sns.catplot( )
  - sns. scatterplot( )
  - ....





https://github.com/andyandhope/Apllied-data-science-capstone/blob/main/04jupyter-labs-eda-dataviz.ipynb

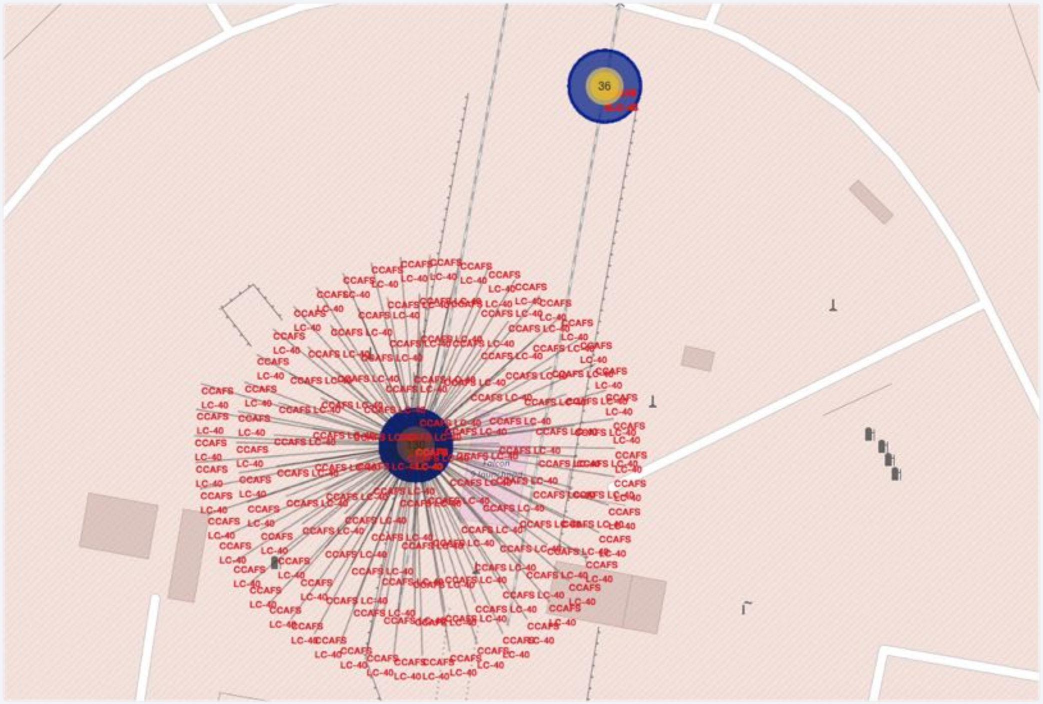
EDA with SQL

- create a new sql database via ”sqlite3”: con=sqlite3.connect(’new_data.db’)
- create cursor to select data
  - cur=con.cursor( )
  - cur.execute(”SELECT” data_interests FROM data_table)...
- get ’records’ via ”table.fetchall()”

https://github.com/andyandhope/Apllied-data-science-capstone/blob/main/03jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

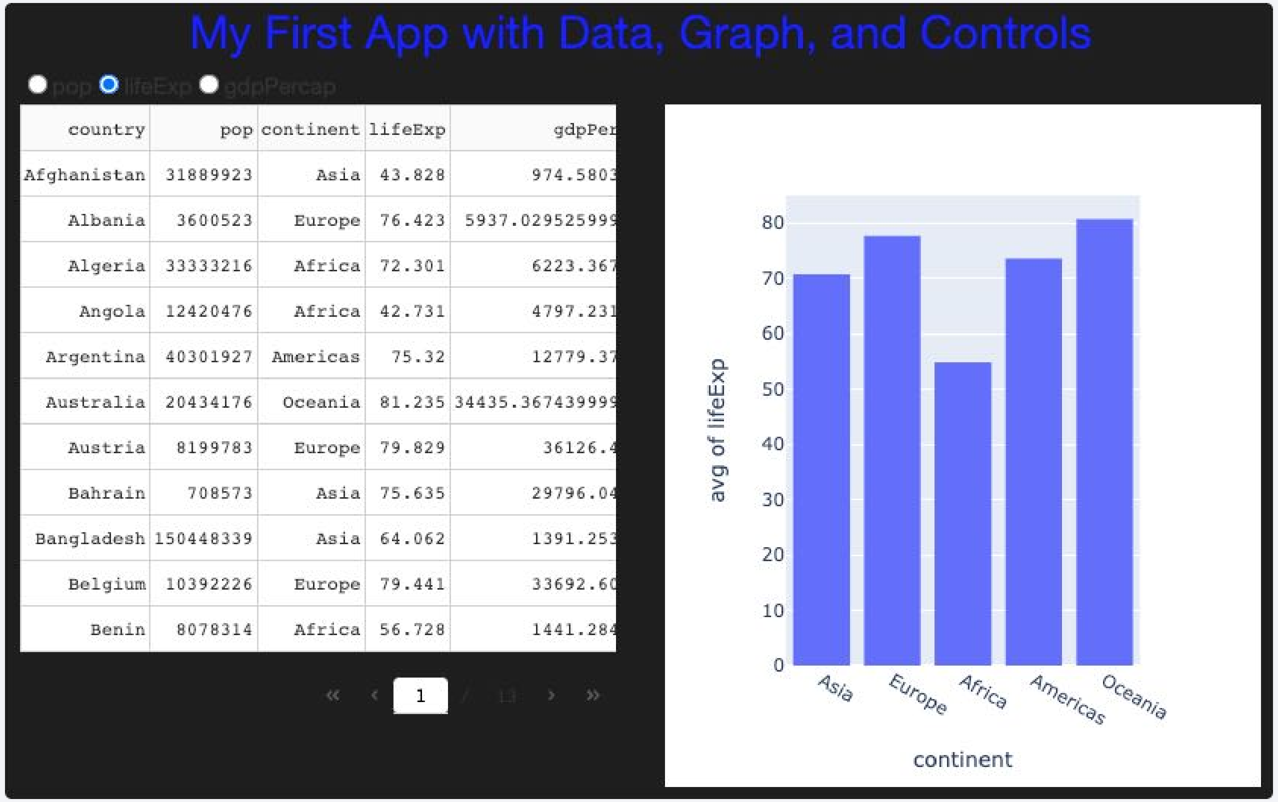
- folium.Map(location, zoom_start=14, tiles="stamenwatercolor", attr=”label")
- Folium makes it easy to visualize data that's been manipulated in Python on an interactive Leaflet map.



https://github.com/andyandhope/Apllied-data-science-capstone/blob/main/05lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

```
# Import packages
from dash import Dash, html, dash_table
import pandas as pd

# Incorporate data
df = pd.read_csv( data_to_be_used  )

# Initialize the app
app = Dash(__name__)

# App layout
app.layout = html.Div(
[
html.Div(children="My First App with Data"),
dash_table.DataTable(data=df.to_dict("records"), page_size=10),
]
)

# Run the app
if __name__ == "__main__":
app.run(debug=True)
```



https://github.com/andyandhope/Apllied-data-science-capstone/blob/main/06_learn_dash_20min.ipynb

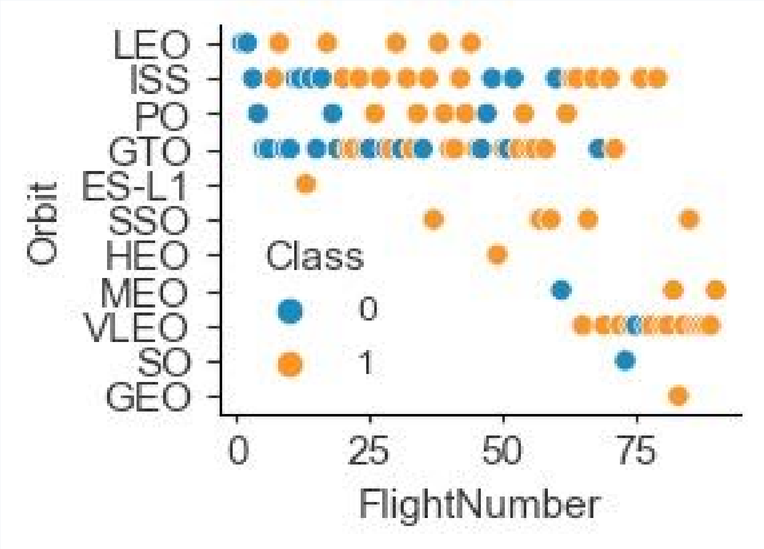
Predictive Analysis (Classification)

- using sklearn package to train the validated data, and find the best Hyperparameter for SVM, Classification Tree and Logistic Regression, here only tree model is shown:

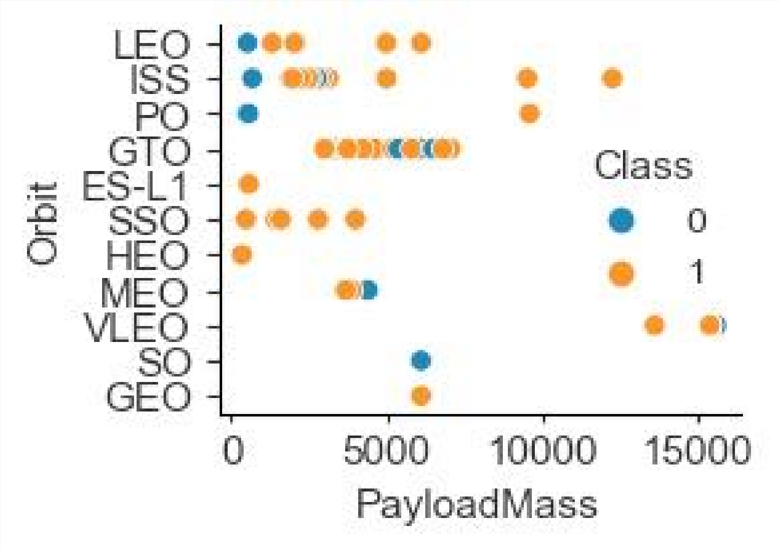
```
parameters = {
"criterion": ["gini", "entropy"],
"splitter": ["best", "random"],
"max_depth": [2 * n for n in range(1, 10)],
"max_features": ["auto", "sqrt"],
"min_samples_leaf": [1, 2, 4],
"min_samples_split": [2, 5, 10],
}
from sklearn import tree

# Decision Tree model
# tree = DecisionTreeClassifier()
dt_model = tree.DecisionTreeClassifier()
tree_cv = GridSearchCV(dt_model, parameters, cv=10, scoring="accuracy")
tree_cv.fit(X_train, Y_train)
```

https://github.com/andyandhope/Apllied-data-science-capstone/blob/main/07SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

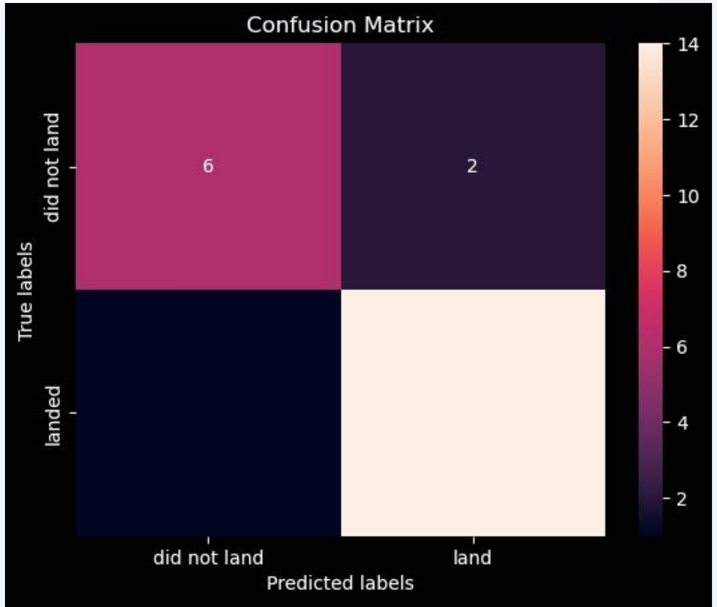
Results



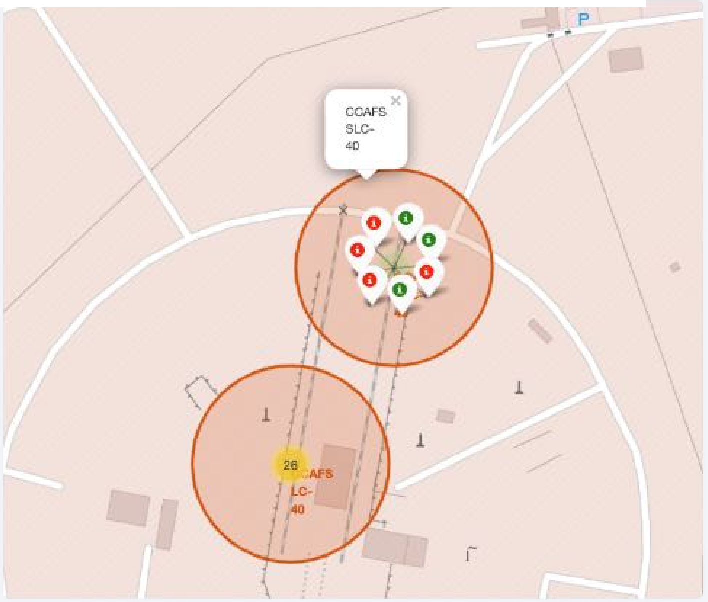
LEO
ISS
PO
GTO
ES-L1
SSO
HEO
MEO
VLEO
SO
GEO
Orbit
Class
0
1
0
25
50
75
FlightNumber



LEO
ISS
PO
GTO
ES-L1
SSO
HEO
MEO
VLEO
SO
GEO
Orbit
Class
0
1
0
5000
10000
15000
PayloadMass



Confusion Matrix
6
2
did not land
landed
True labels
did not land
land
Predicted labels
14
12
10
8
6
4
2

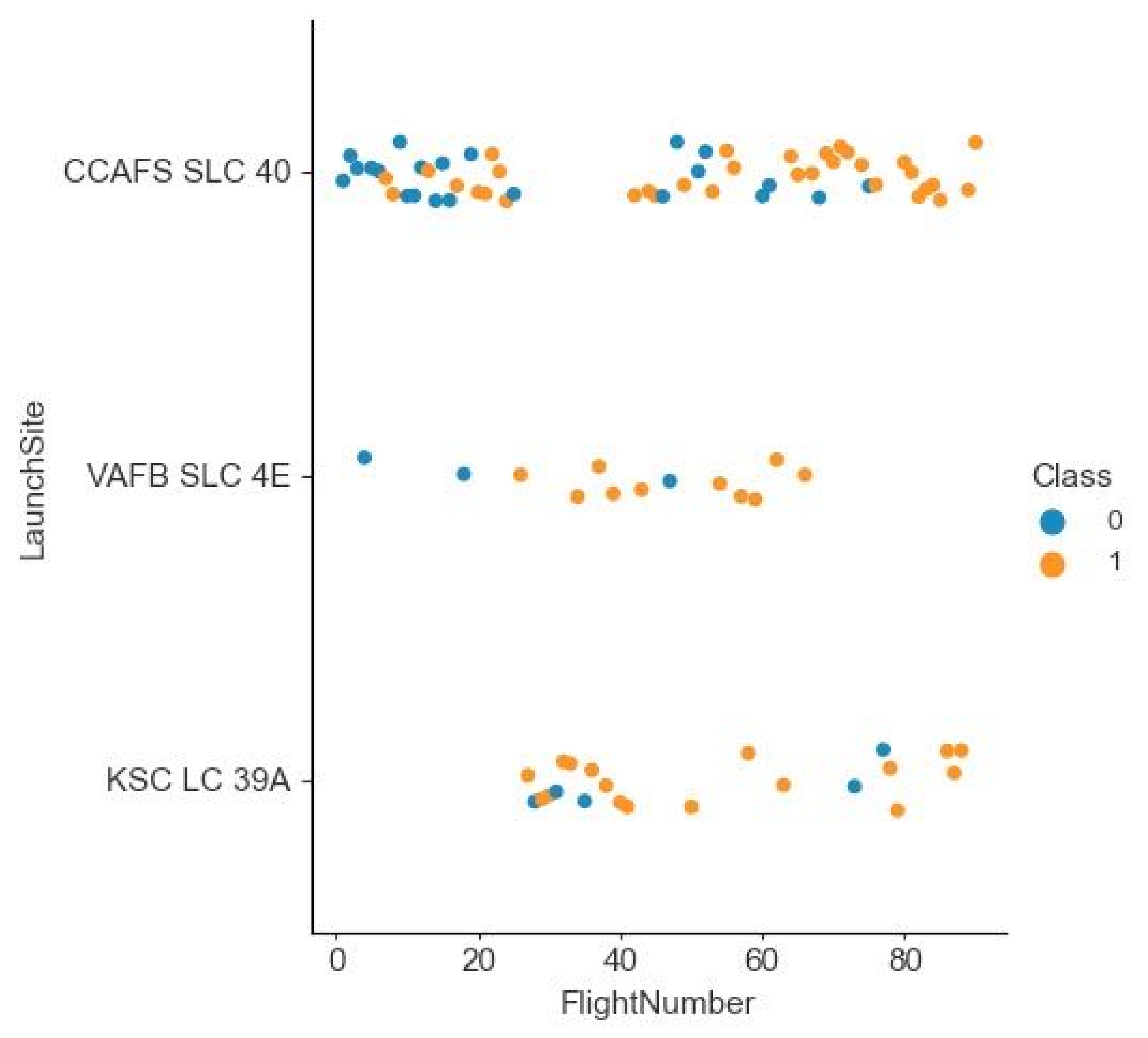


CCAFS
SLC-
40
26

Section 2

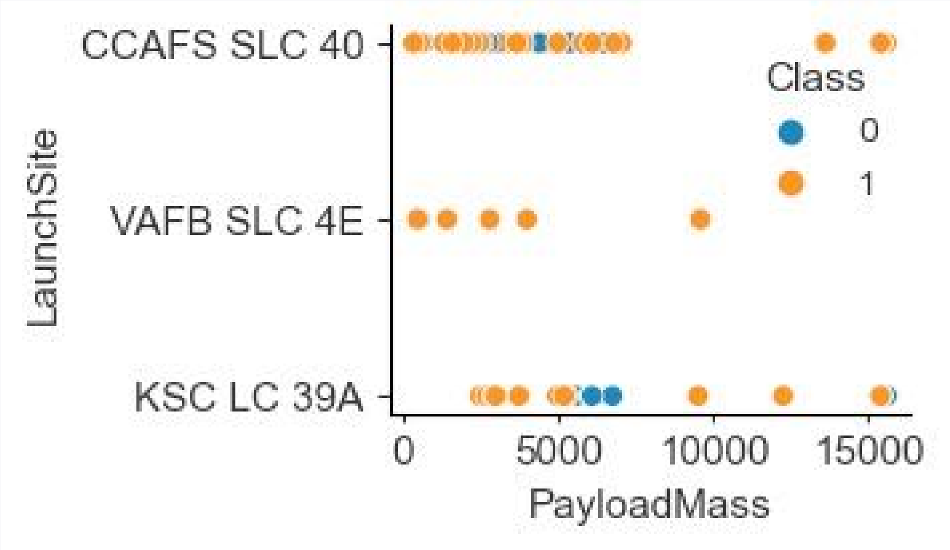
Insights drawn from EDA

Flight Number vs. Launch Site



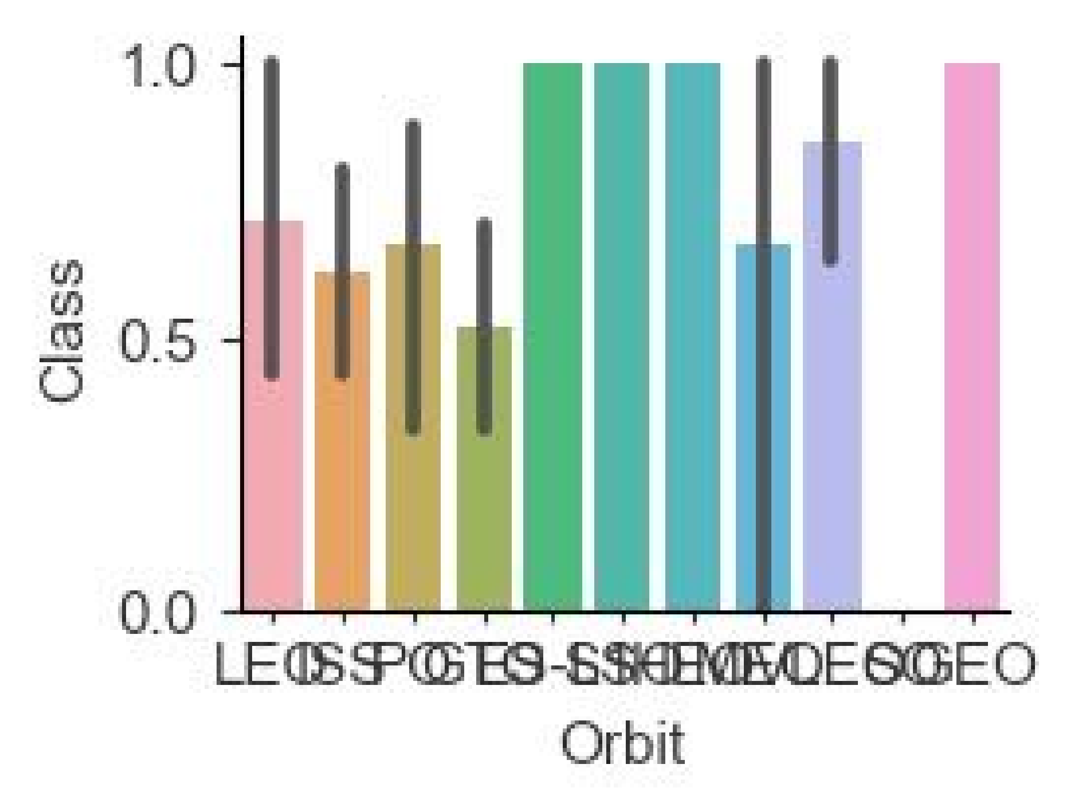

```
sns.catplot(
        data=df,
        x="FlightNumber",
        y="LaunchSite",
        hue="Class”
        )
```

Payload vs. Launch Site



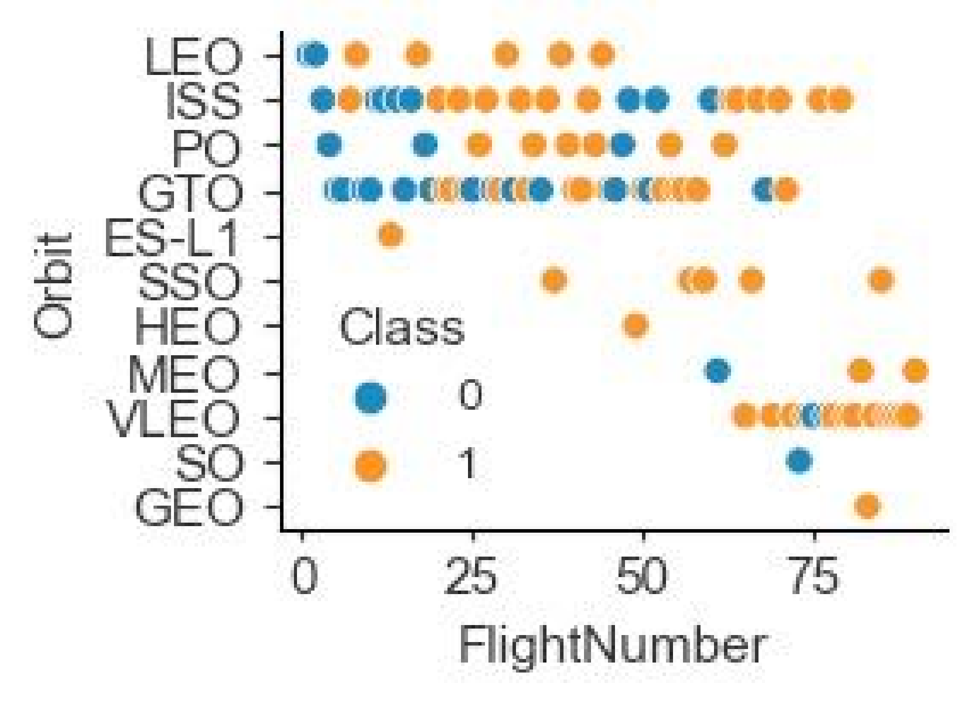

```
sns.scatterplot(data=df, x="PayloadMass", y="LaunchSite", hue="Class")
```

Success Rate vs. Orbit Type



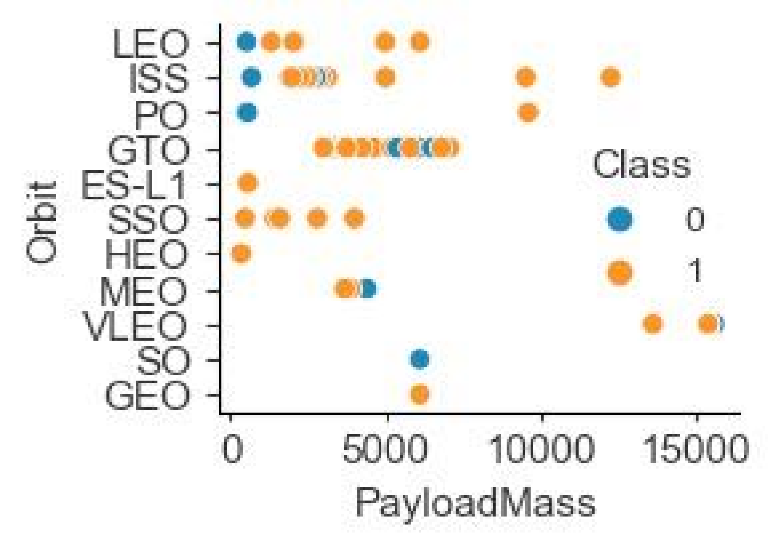

```
sns.barplot(data=df, y="Class", x="Orbit")
```

Flight Number vs. Orbit Type



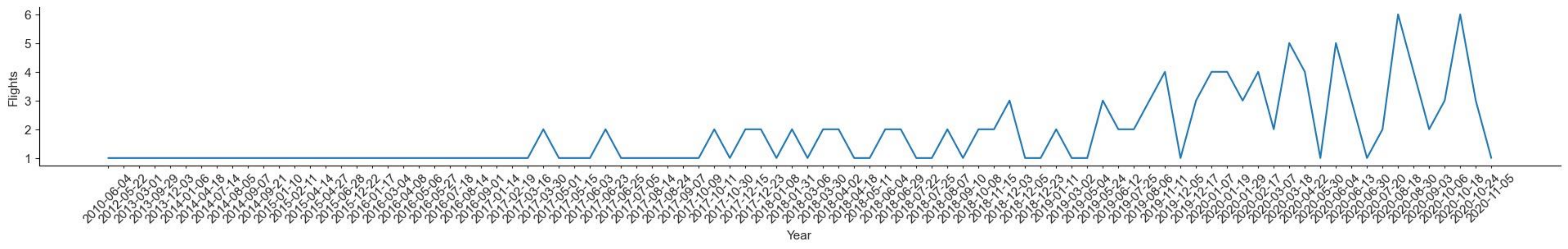

```
sns.scatterplot(data=df, x="FlightNumber", y="Orbit", hue="Class")
```

Payload vs. Orbit Type




```
sns.scatterplot(data=df, x="PayloadMass", y="Orbit", hue="Class")
```

Launch Success Yearly Trend




```
plt.figure(figsize=(24, 2.5))
my_plot = sns.lineplot(data=df, x="Year",
y="Flights")
my_plot.set_xticklabels(my_plot.get_xtickl
abels(), rotation=45)
```

All Launch Site Names

```
tab_SPACEXTABLE = cur.execute("SELECT * FROM SPACEXTABLE")
records = tab_SPACEXTABLE.fetchall()
launchsites = []
for row in records:
        if row[3] not in launchsites:
        launchsites.append(row[3])
launchsites
```

```
['CCAFS LC-40', 'VAFB SLC-4E', 'KSC LC-
39A', 'CCAFS SLC-40']
```

Launch Site Names Begin with 'CCA'

```
tab_SPACEXTABLE = cur.execute("SELECT * FROM SPACEXTABLE LIMIT 5")
records = tab_SPACEXTABLE.fetchall()
CCA_sites = []
for row in records:
        if row[3] not in CCA_sites:
        CCA_sites.append(row[3])
CCA_sites
```

Total Payload Mass

```
for row in records:
      if row[7] == "NASA (CRS)":
      print("Playload:", row[4], "mass(kg):", row[5])
```

```
Playload: SpaceX CRS-1 mass(kg): 500
Playload: SpaceX CRS-2 mass(kg): 677
```

Average Payload Mass by F9 v1.1

```
tab_SPACEXTABLE = cur.execute(
        "SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version LIKE 'F9 v1.1‘ "
)

records = tab_SPACEXTABLE.fetchall()
print(records, a)
```

First Successful Ground Landing Date

```
tab_SPACEXTABLE = cur.execute(
"SELECT Date FROM SPACEXTABLE WHERE Landing_Outcome = 'Success' LIMIT 1"
)

records = tab_SPACEXTABLE.fetchall()
print(records)
```

Successful Drone Ship Landing with Payload between 4000 and 6000

```
tab_SPACEXTABLE = cur.execute(
"SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTABLE"
)
records = tab_SPACEXTABLE.fetchall()
# print(records)
res = []
for row in records:
        if 4000 < row[1] < 6000:
                res.append(row[0])
                print("\n", row[0])
res
```

Total Number of Successful and Failure Mission Outcomes

```
tab_SPACEXTABLE_no_succ = cur.execute(
"SELECT Mission_Outcome FROM SPACEXTABLE WHERE Mission_Outcome != 'Success' "
)
records_no_succ = tab_SPACEXTABLE_no_succ.fetchall()
print(len(records_no_succ))


tab_SPACEXTABLE_succ = cur.execute(
"SELECT Mission_Outcome FROM SPACEXTABLE WHERE Mission_Outcome = 'Success' "
)
records_succ = tab_SPACEXTABLE_succ.fetchall()
print(len(records_succ))
```

Boosters Carried Maximum Payload

```
tab_SPACEXTABLE = cur.execute(
"SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTABLE"
)
records = tab_SPACEXTABLE.fetchall()

# !pip install operator
from operator import itemgetter

max(records, key=itemgetter(1))
```

```
('F9 B5 B1048.4', 15600)
```

2015 Launch Records

```
for i in df["Date"]:
    if i[:4] == "2015":
        # print(i)
        print(calendar.month_name[int(i[5:7])])
```

```
January
February
March
April
April
June
```

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

```python

import seaborn as sns
from ips import *
%matplotlib inline
set_pub()
df.sort_values(by='Date',ascending=True)
df.head(4)
# plt.figure(figsize=(26,3))
# f1=sns.lineplot(data=df, y="PAYLOAD_MASS__KG_",x = 'Date')
# f1.set_xticklabels(f1.get_xticklabels(), rotation=45)
```

	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
0	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-12-08	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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt

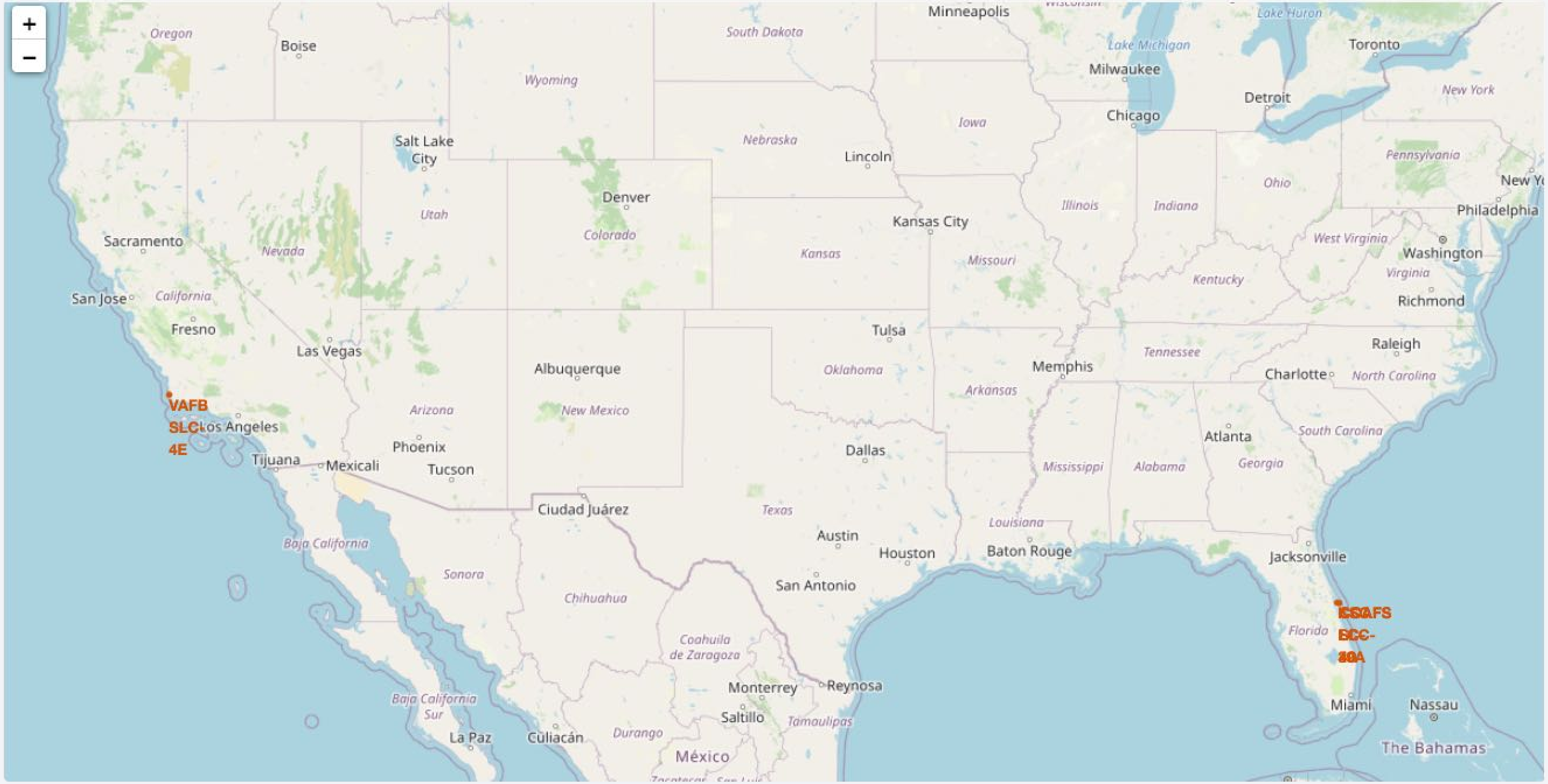
+ Code + Markdown

Section 3

Launch Sites Proximities Analysis

launch sites at NASA

```
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
```



<Folium Map Screenshot 2>

- Replace <Folium map screenshot 2> title with an appropriate title
- Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map
- Explain the important elements and findings on the screenshot

<Folium Map Screenshot 3>

- Replace <Folium map screenshot 3> title with an appropriate title

- Explore the generated folium map and show the screenshot of a selected launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed

- Explain the important elements and findings on the screenshot

Section 4

Build a Dashboard with Plotly Dash

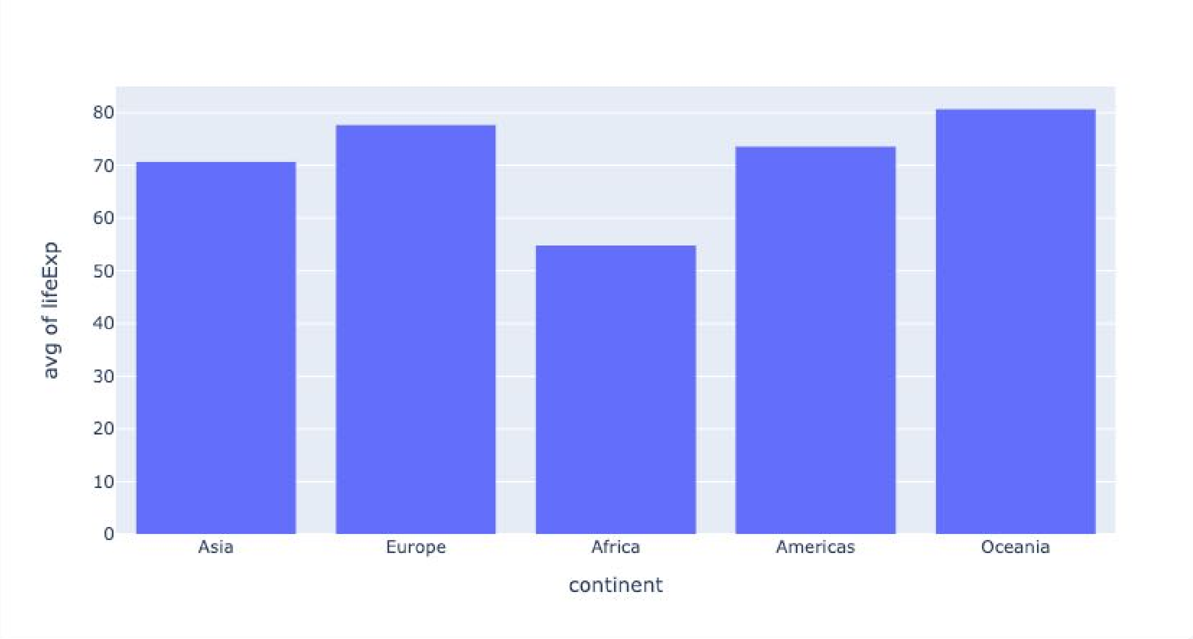
My First App with Data and a Graph

My First App with Data, Graph, and Controls

○ pop ◉ lifeExp ○ gdpPercap

country	pop	continent	lifeExp	gdpPercap
Afghanistan	31889923	Asia	43.828	974.5803384
Albania	3600523	Europe	76.423	5937.029525999999
Algeria	33333216	Africa	72.301	6223.367465
Angola	12420476	Africa	42.731	4797.231267
Argentina	40301927	Americas	75.32	12779.37964
Australia	20434176	Oceania	81.235	34435.367439999995
Austria	8199783	Europe	79.829	36126.4927
Bahrain	708573	Asia	75.635	29796.04834
Bangladesh	150448339	Asia	64.062	1391.253792
Belgium	10392226	Europe	79.441	33692.60508
Benin	8078314	Africa	56.728	1441.284873

« ‹ 1 / 13 › »



Section 5

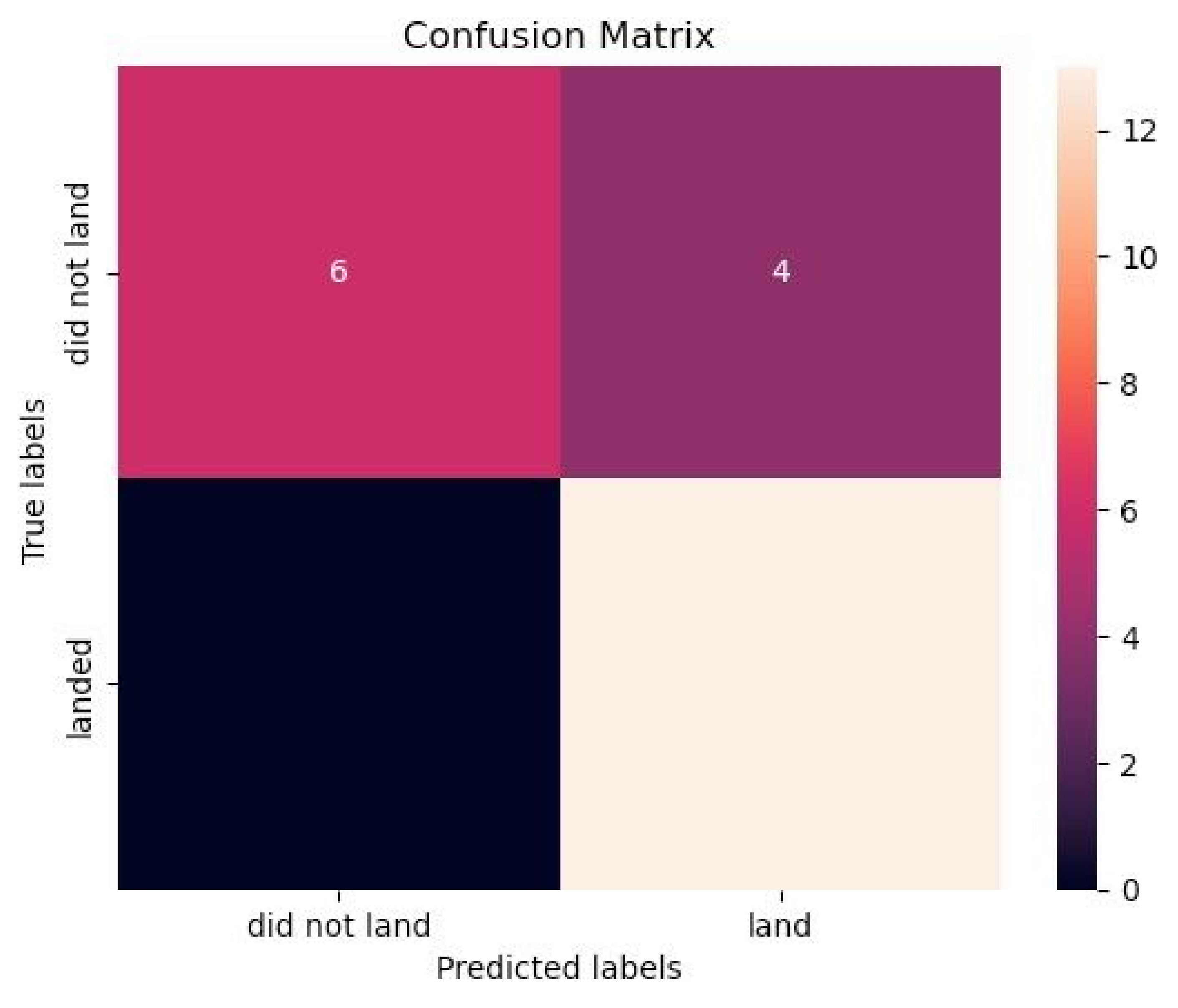
Predictive Analysis (Classification)

Classification Accuracy

- Visualize the built model accuracy for all built classification models, in a bar chart

- Find which model has the highest classification accuracy

Confusion Matrix



Conclusions (what I learnt from this project)

Via employing machine learning techniques to predict the success of the first stage landing of the Falcon 9 rocket. In general, what we need to with doing few steps below:

1. **Data Splitting:**
   - The dataset is divided into training and test sets, a crucial step in machine learning model development.
2. **Model Selection:**
   - Three different machine learning algorithms are considered: Support Vector Machine (SVM), Classification Trees, and Logistic Regression.
3. **Hyperparameter Tuning:**
   - Hyperparameter tuning is performed using the training data to identify the optimal configuration for each algorithm. This process is carried out through GridSearchCV.
4. **Model Evaluation:**
   - The performance of each model is evaluated using the test data.
   - The chosen metrics for evaluation may include accuracy, precision, recall, or F1-score, depending on the specific requirements.
5. **Selection of Best Performing Model:**
   - The model exhibiting the highest predictive accuracy on the test set is selected as the optimal solution for predicting Falcon 9 first stage landing success.

Appendix

- dataset_part_1.csv
- dataset_part_2.csv
- dataset_part_3.csv
- my_data1.db
- spacex_launch_dash.csv
- spacex_launch_geo.csv

Thank you!