Winning Space Race with Data Science

YU CHEN

2022/09/18

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- Background:

SpaceX found in 2002, by Elon Musk, their goal is to make spaceflight affordable by reducing the space transportation costs, and then to enable the colonization of Mars. Falcon 9, which flown and reused over 100 times, became their most success series.

- Problems

Why Falcon 9 succeeded?

How SpaceX compares to others?

Section 1

Methodology

Methodology

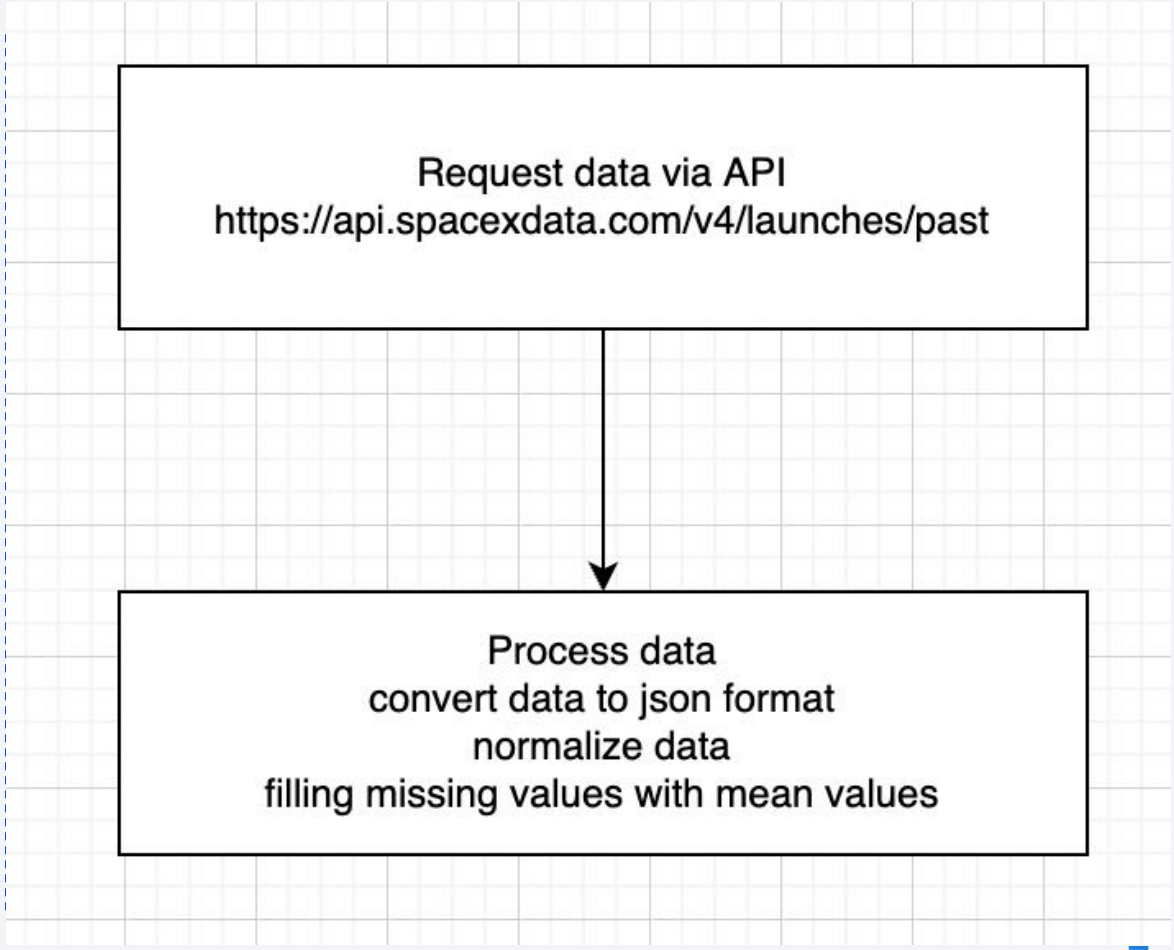
Executive Summary

- Data collection methodology:
- Perform data wrangling
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection – SpaceX API

FlightNumber	0
Date	0
BoosterVersion	0
PayloadMass	0
Orbit	1
LaunchSite	0
Outcome	0
Flights	0
GridFins	0
Reused	0
Legs	0
LandingPad	26

- https://github.com/adm73/ibm-ds/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

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

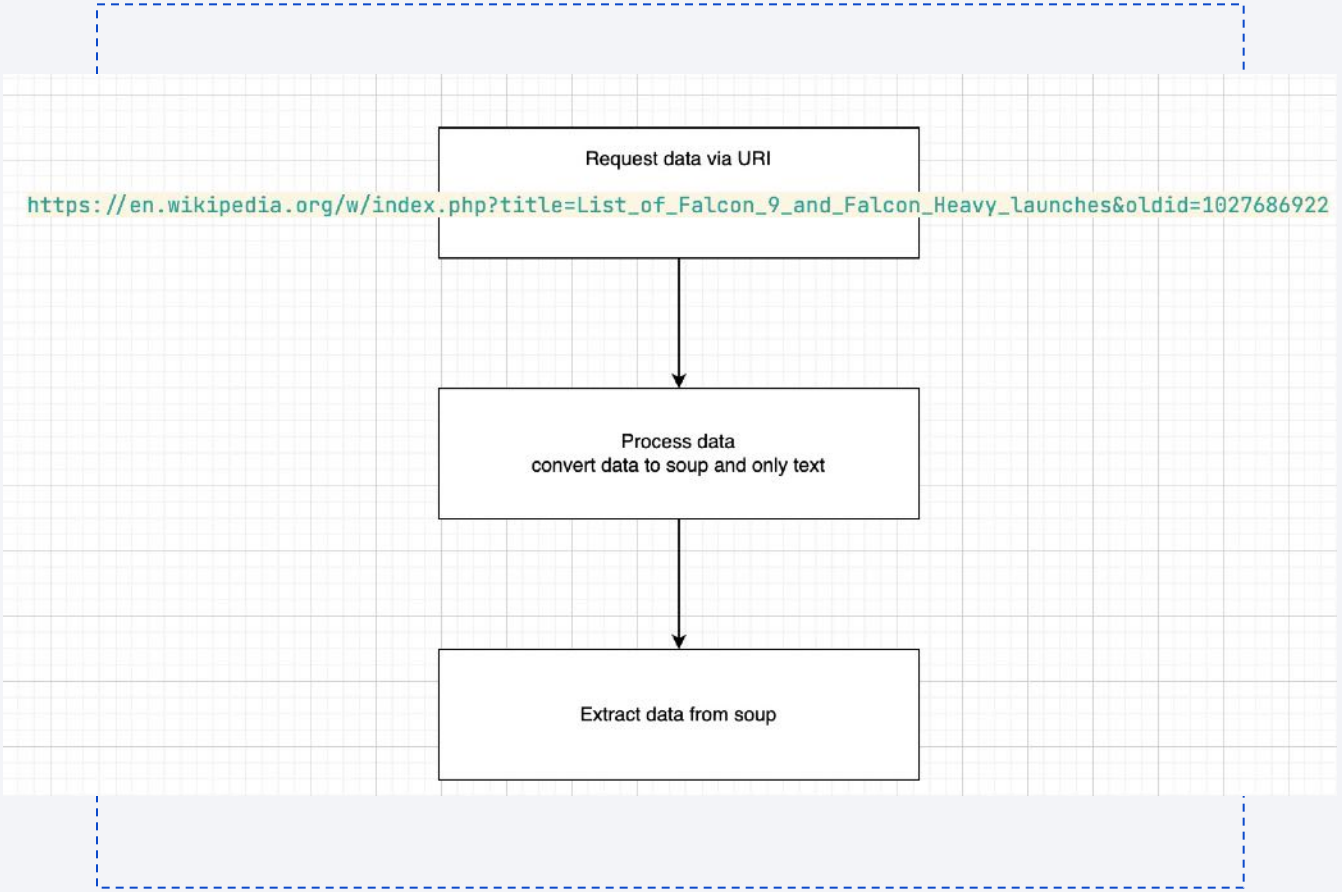
Create a BeautifulSoup object from the HTML response

Add Code Cell Add Markdown Cell

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

soup = BeautifulSoup(getdata, 'html.parser')
```

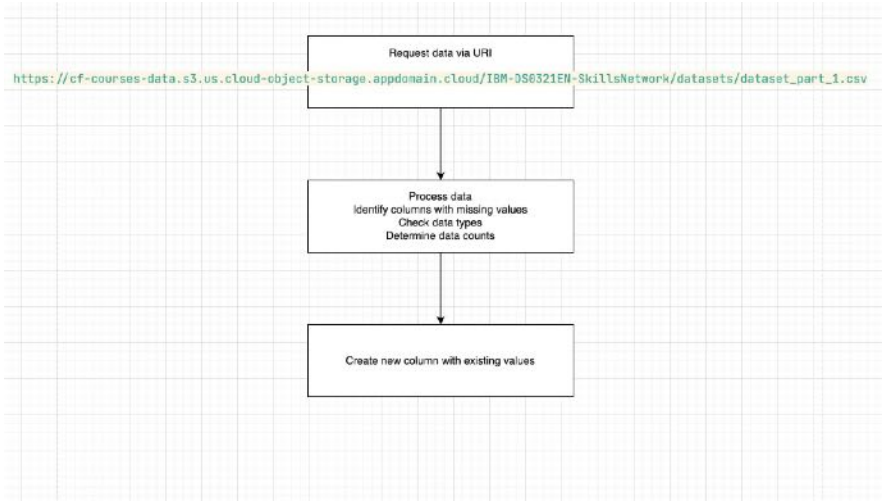
Print the page title to verify if the BeautifulSoup object was created properly

```
# Use soup.title attribute
soup.title
```



- https://github.com/adm73/ibm-ds/blob/main/jupyter-labs-webscraping.ipynb

Data Wrangling



- https://github.com/adm73/ibm-ds/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

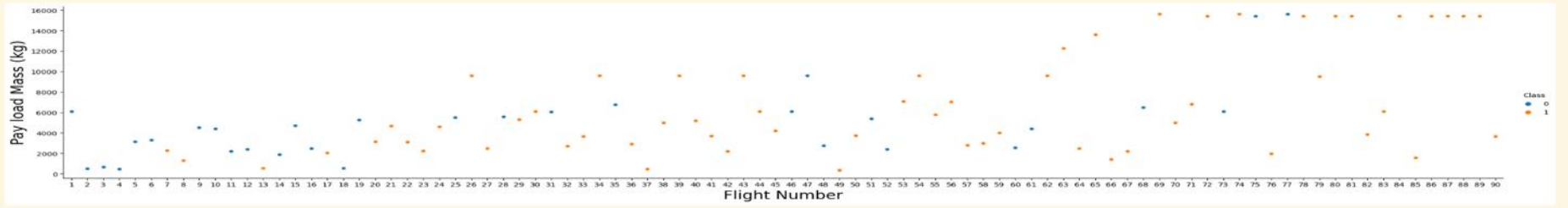
TASK 4: Create a landing outcome label from Outcome column

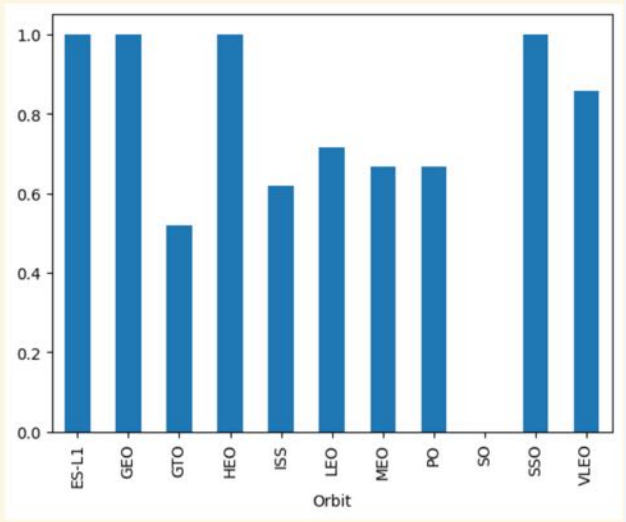
Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwis assign it to the variable `landing_class`:

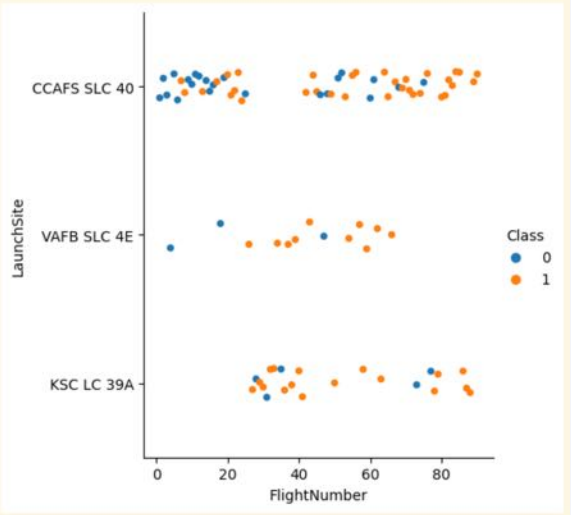
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise

landing_class = []
for i,outcome in df['Outcome'].items():
    # print(i,outcome)
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)

```

EDA with Data Visualization

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







- https://github.com/adm73/ibm-ds/blob/main/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- **Display the names of the unique launch sites in the space mission**
- **Display 5 records where launch sites begin with the string 'CCA'**
- **Display the total payload mass carried by boosters launched by NASA (CRS)**
- **Display average payload mass carried by booster version F9 v1.1**
- **List the date when the first successful landing outcome in ground pad was acheived.**
- **List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000**
- **List the total number of successful and failure mission outcomes**
- **List the names of the booster_versions which have carried the maximum payload mass. Use a subquery**
- **List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015**
- **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**
- https://github.com/adm73/ibm-ds/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- **Mark all launch sites on a map**
- **Mark the success/failed launches for each site on the map**
- **Calculate the distances between a launch site to its proximities**

The launch success rate may depend on many factors such as payload mass, orbit type, and so on. It may also depend on the location and proximities of a launch site, i.e., the initial position of rocket trajectories. Finding an optimal location for building a launch site certainly involves many factors and hopefully we could discover some of the factors by analyzing the existing launch site locations.

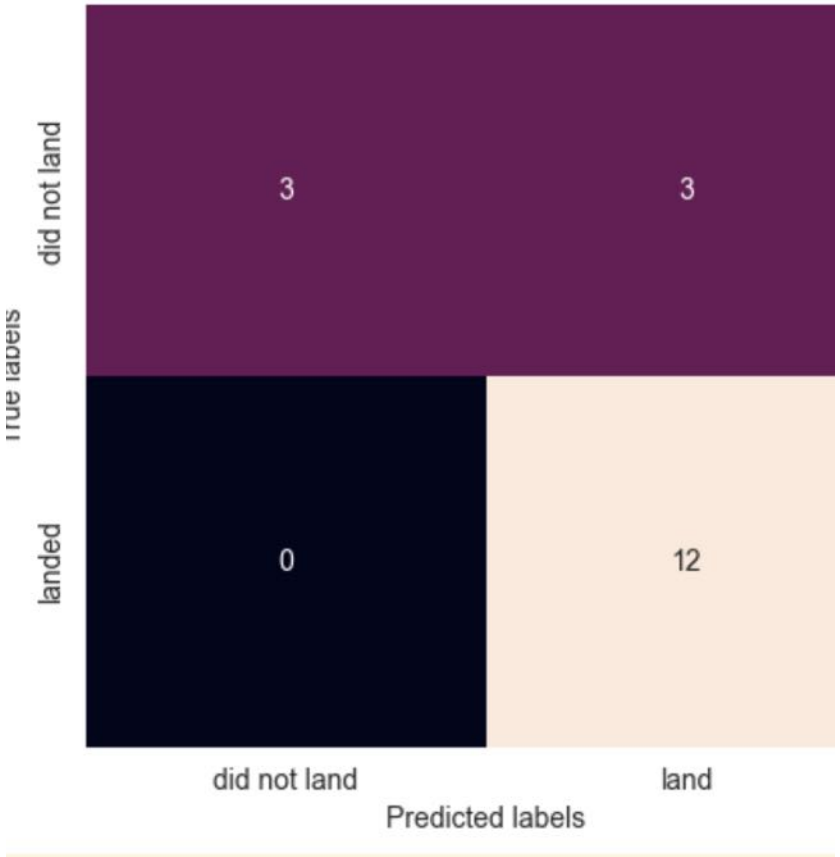
- https://github.com/adm73/ibm-ds/blob/main/lab_jupyter_launch_site_location.ipynb

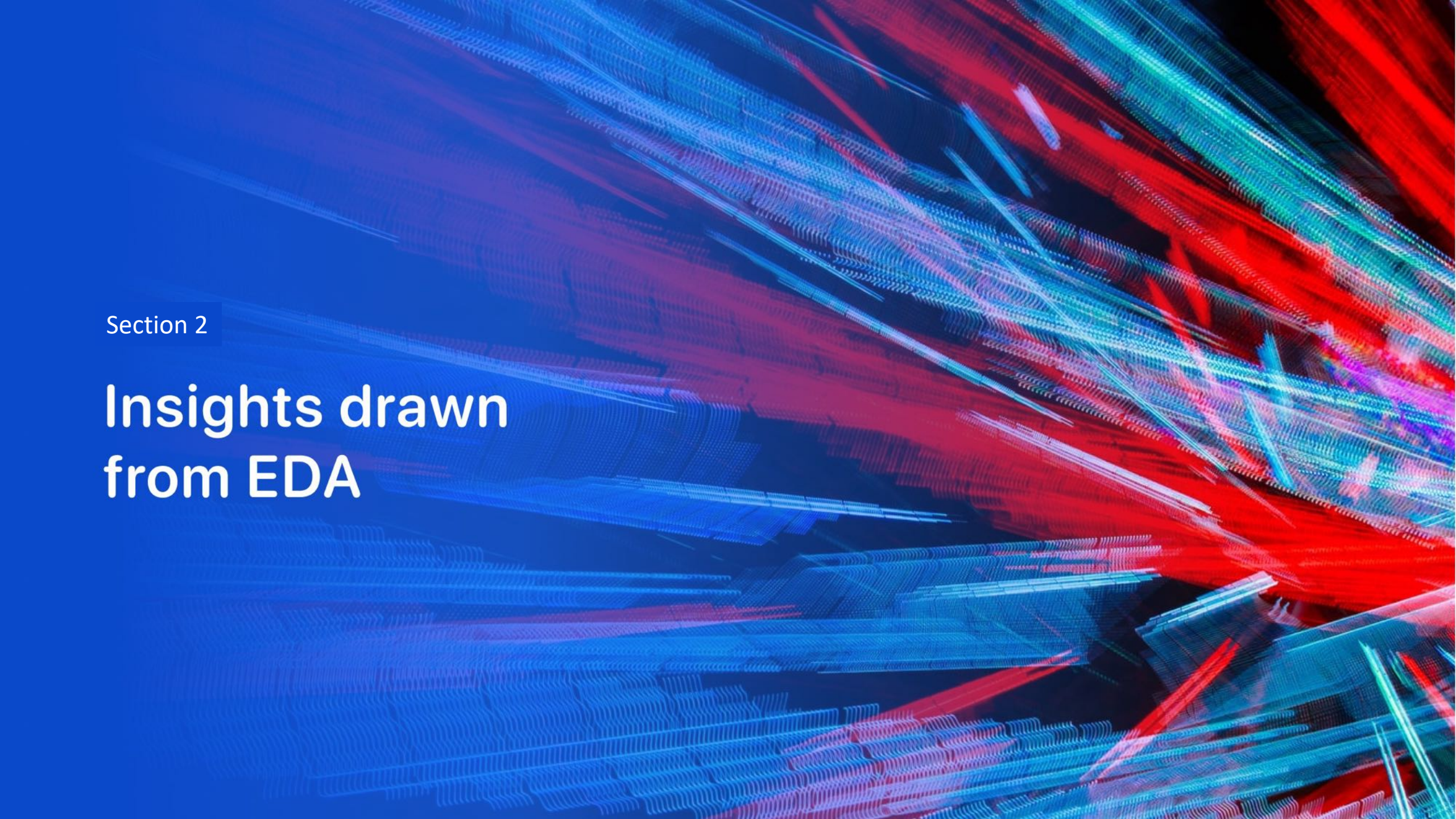
Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
- Explain why you added those plots and interactions
- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

Predictive Analysis (Classification)

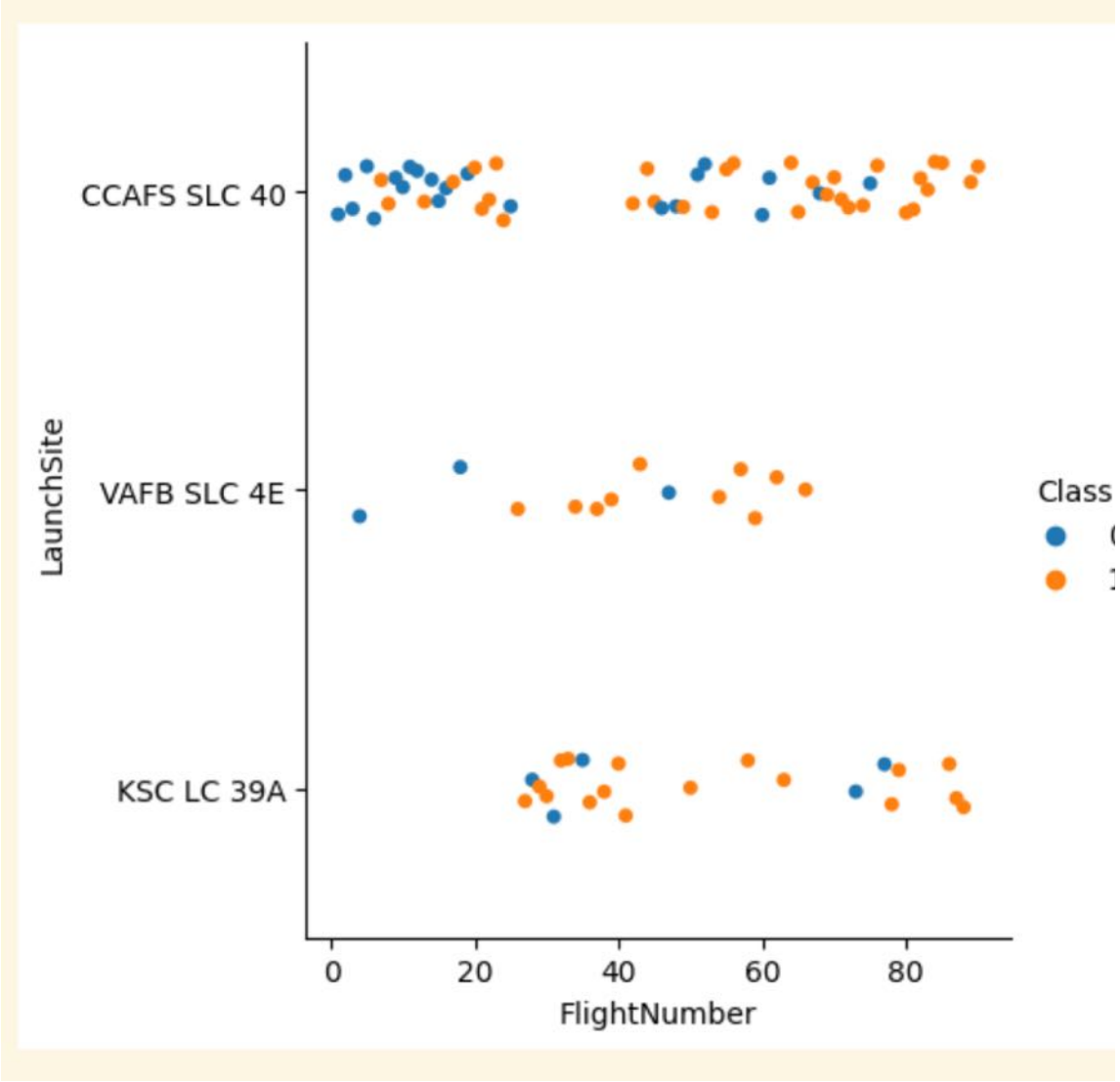
- Create a NumPy array from the column Class in data, by applying the method to_numpy() then assign it to the variable Y,make sure the output is a Pandas series (only one bracket df['name of column']).
- Standardize the data in X then reassign it to the variable X using the transform provided below.
- Use the function train_test_split to split the data X and Y into training and test data. Set the parameter test_size to 0.2 and random_state to 2. The training data and test data should be assigned to the following labels.
- Create a logistic regression object then create a GridSearchCV object logreg_cv with cv = 10. Fit the object to find the best parameters from the dictionary parameters.
- Calculate the accuracy on the test data using the method score:
- Create a support vector machine object then create a GridSearchCV object svm_cv with cv - 10. Fit the object to find the best parameters from the dictionary parameters.
- Calculate the accuracy on the test data using the method score:
- Create a decision tree classifier object then create a GridSearchCV object tree_cv with cv = 10. Fit the object to find the best parameters from the dictionary parameters.
- Calculate the accuracy of tree_cv on the test data using the method score:
- Create a k nearest neighbors object then create a GridSearchCV object knn_cv with cv = 10. Fit the object to find the best parameters from the dictionary parameters.
- Calculate the accuracy of tree_cv on the test data using the method score:
- https://github.com/adm73/ibm-ds/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb




Section 2
Insights drawn from EDA

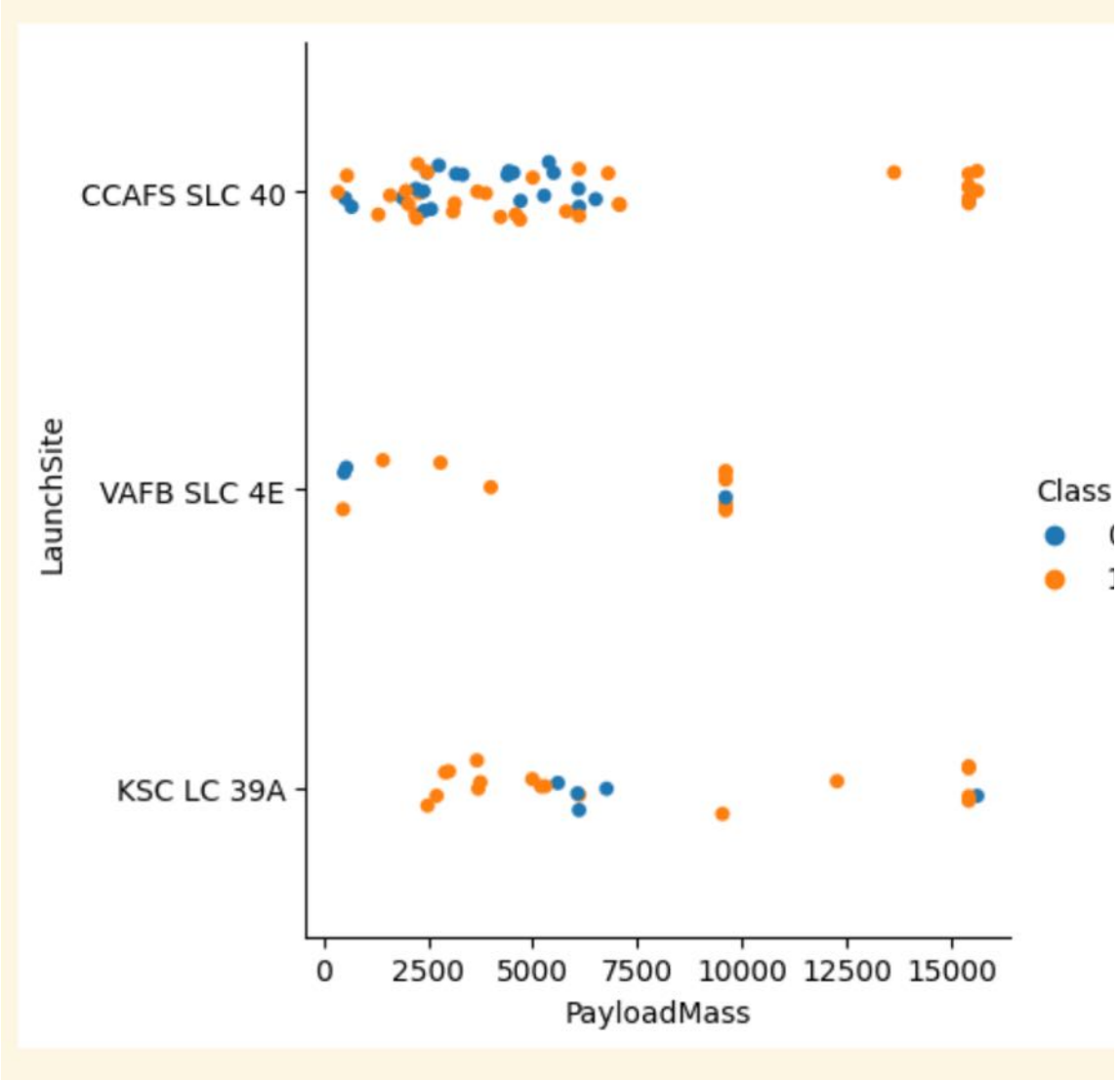
Flight Number vs. Launch Site

- CCAFS has more flights
- VAFB has least flights



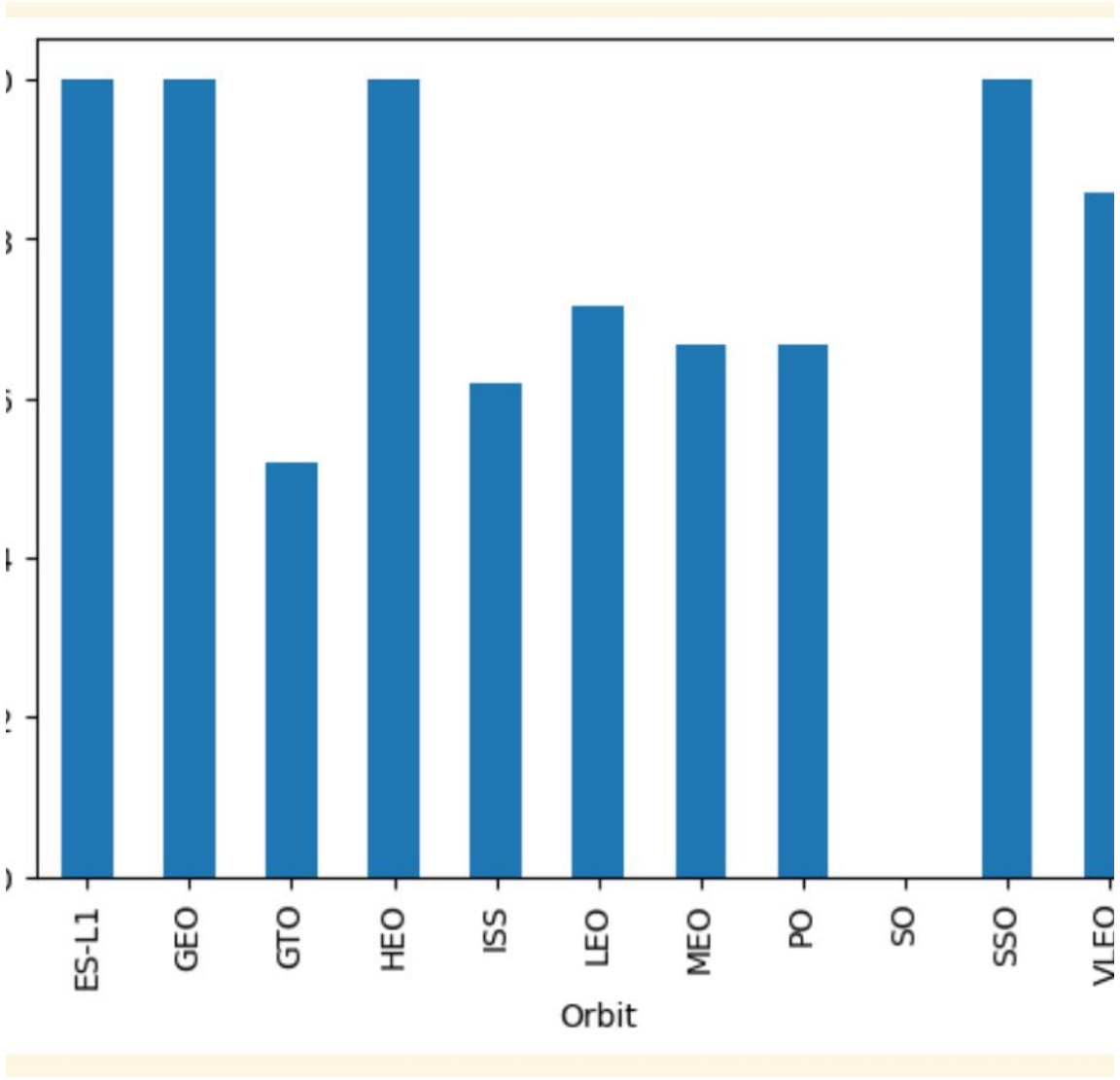
Payload vs. Launch Site

- VAFB has no PayloadMass greater than 10000
- Most of the PayloadMass are less than 7500 at CCAFS site
- Most of the payloadMass at ~10000 launched at VAFB Site



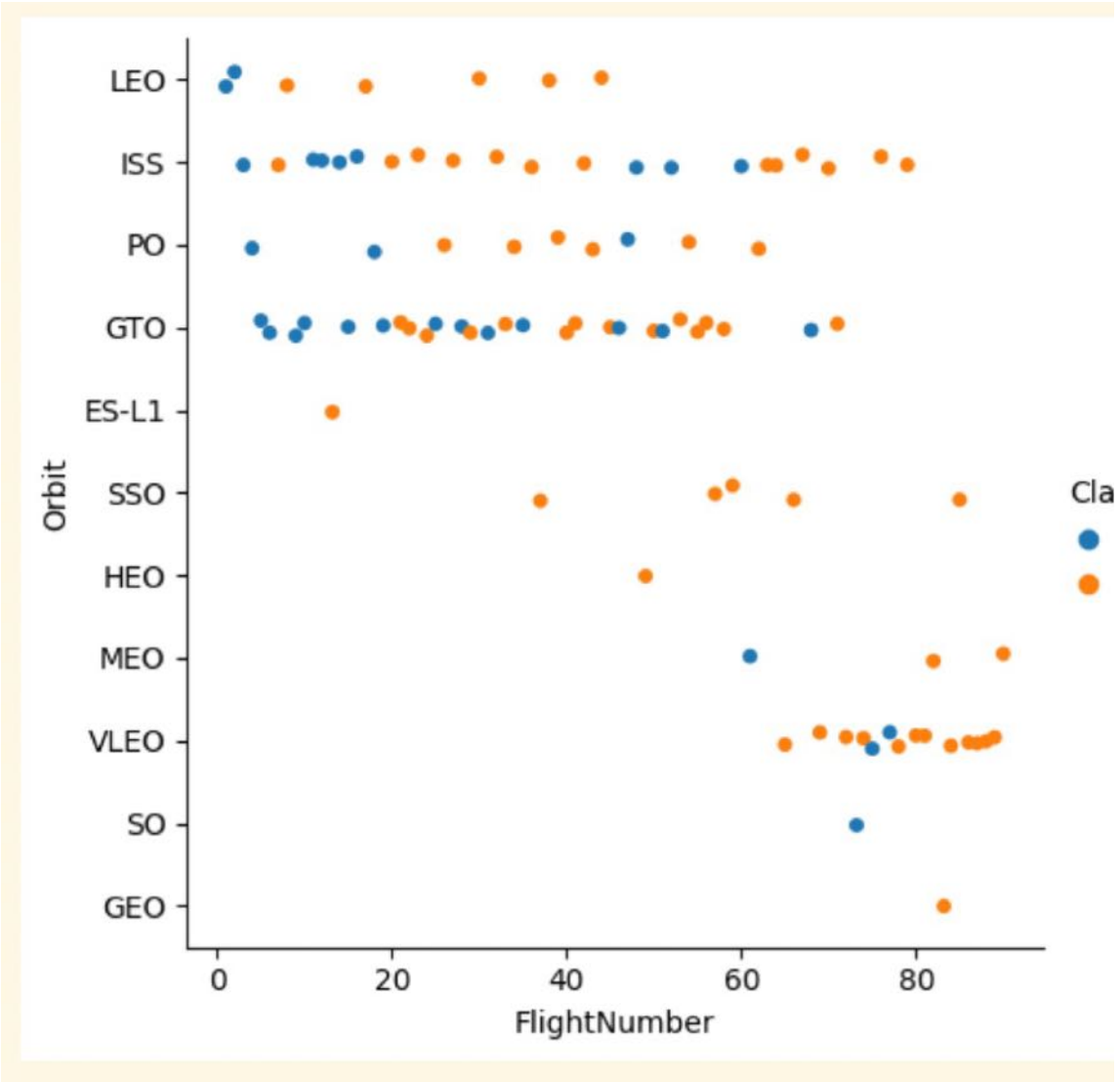
Success Rate vs. Orbit Type

- Orbit Type of SO are all failed
- ES-L1, GEO, HEO, SSO have a 100% success rate



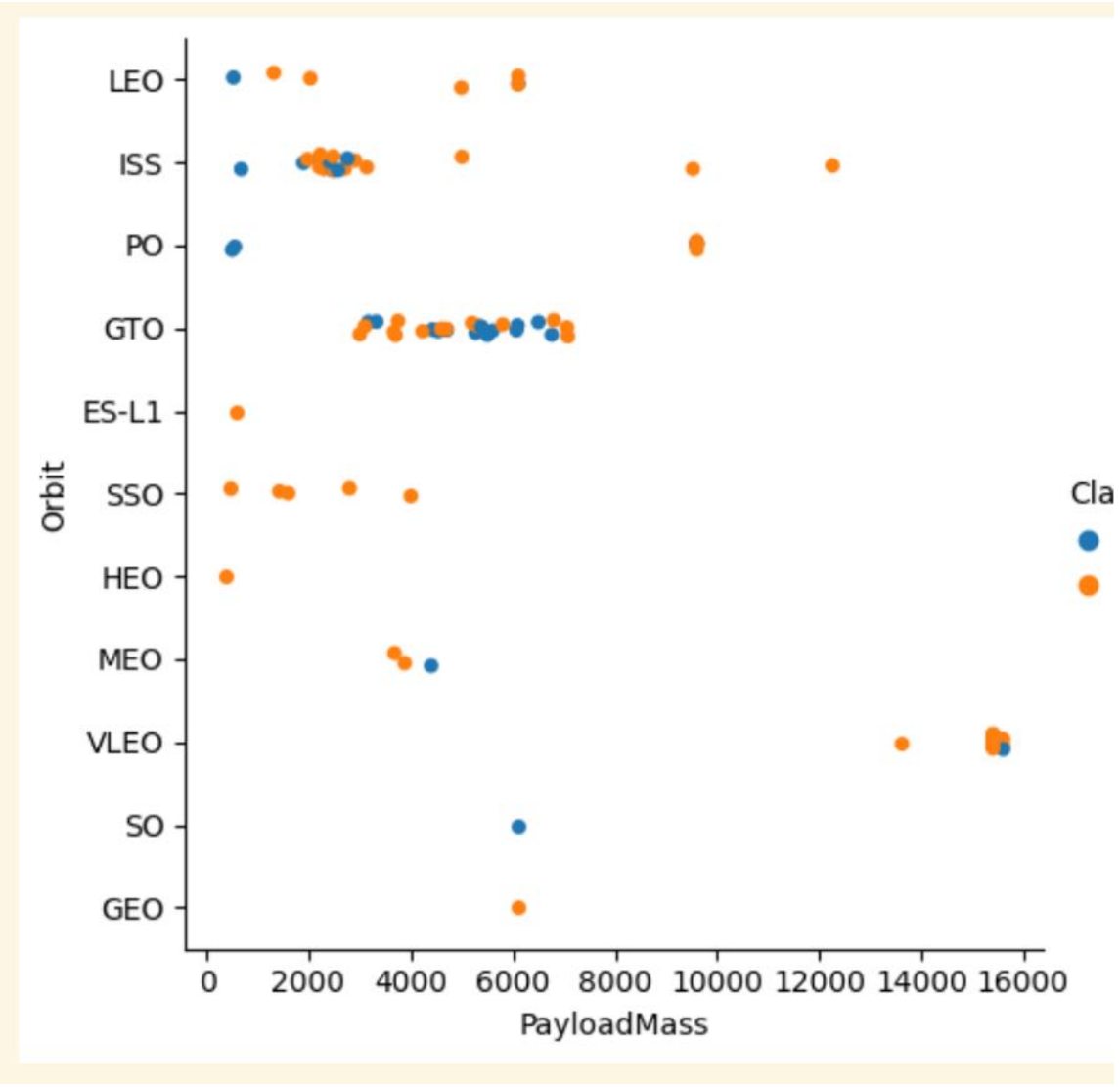
Flight Number vs. Orbit Type

- At early stage, more types of LEO, ISS, PO, and GTO
- At later stage, more VLEO type of orbit



Payload vs. Orbit Type

- VLEO has the highest PayloadMass
- GTO’s range is in 2000 – 8000
- SO and GEO are at 6000



Launch Success Yearly Trend

- From 2013, the success rate increased dramatically
- 2010 – 2012, the success rate was zero

Task 1

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

```
%sql select DISTINCT(Launch_site) from SPACEXTBL;
```

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

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

All Launch Site Names

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

Launch Site Names Begin with 'CCA'

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

```
%sql select * from spacextbl where launch_site like 'CCA%' LIMIT 5
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

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

```
%sql select SUM(PAYLOAD_MASS__kg_) as total_payload_mass from spacextbl where customer = 'SpaceX';
```

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

total_payload_mass
185220

Average Payload Mass by F9 v1.1

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

Task 4

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

```
%sql select AVG(PAYLOAD_MASS__KG_) as avg_payload_mass from spacextbl where Booster_Version = 'F9 v1.1'
```

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

avg_payload_mass
2928.4

First Successful Ground Landing Date

- **List the date when the first succesful landing outcome in ground pad was acheived.**
-

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
%sql select * from spacextbl where 'Landing _Outcome' = "Success (ground pad)";
```

```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: LANDING_OUTCOME
[SQL: select * from spacextbl where LANDING_OUTCOME = "Success (ground pad)" ;]
(Background on this error at: http://sqlalche.me/e/e3q8)
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- **List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000**

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
%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.
```

Booster_Version

Total Number of Successful and Failure Mission Outcomes

- **List the total number of successful and failure mission outcomes**

List the total number of successful and failure mission outcomes

```
%sql select \
      sum(case when Upper(mission_outcome) like '%SUCCESS%' then 1 else 0 end ) AS "success", \
      sum(case when Upper(mission_outcome) like '%FAILTURE%' then 1 else 0 end ) AS "failed" \
from spacextbl;
```

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

success	failed
100	0

Boosters Carried Maximum Payload

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

Task 8

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

```
%sql SELECT DISTINCT BOOSTER_VERSION AS max_payload_mass FROM SPACEXTBL \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

max_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

2015 Launch Records

- **List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.**

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

```
%sql SELECT SUBSTR(Date, 4, 2) as month, booster_version, launch_site \
FROM spacextbl \
      WHERE substr(date, 7, 4) = '2015'
```

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

month	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
02	F9 v1.1 B1013	CCAFS LC-40
03	F9 v1.1 B1014	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40
04	F9 v1.1 B1016	CCAFS LC-40
06	F9 v1.1 B1018	CCAFS LC-40
12	F9 FT B1019	CCAFS LC-40

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

- **Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.**

Task 10

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql select count(landing__outcome) as count_lo from spacextbl where date between "2010-06-04" and "2017-03-20" group by landing_outcome ORDER BY count_lo DESC;
```

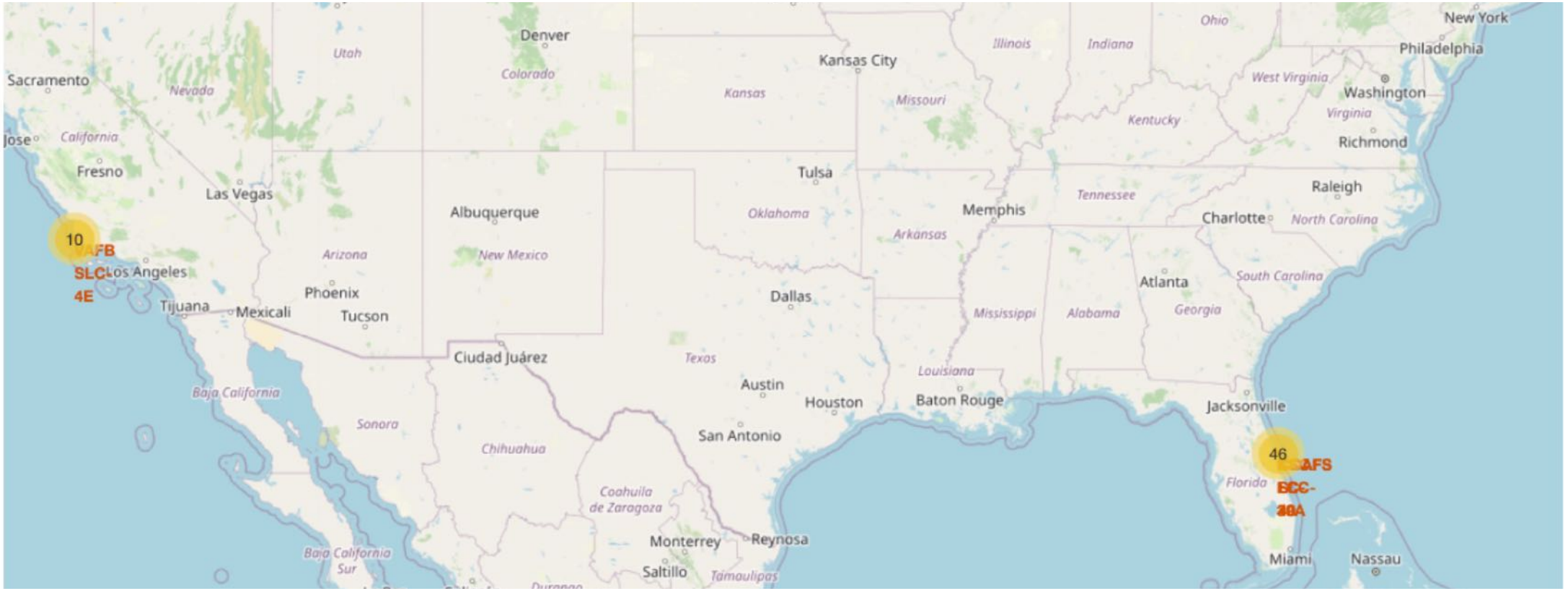
```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: landing__outcome
```

Section 3

Launch Sites Proximities Analysis

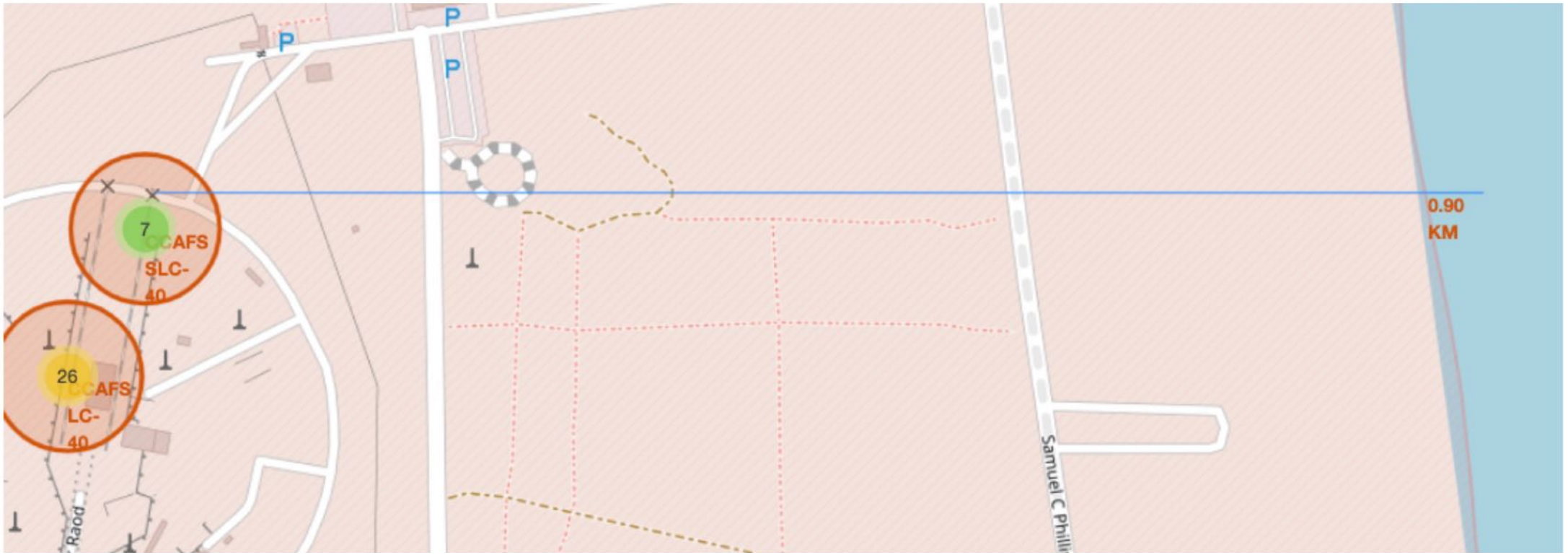
<Folium Map Screenshot 1>

- Explore the generated folium map and make a proper screenshot to include all launch sites' location markers on a global map



<Folium Map Screenshot 2>

- Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map



<Folium Map Screenshot 3>

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

Section 5

Predictive Analysis (Classification)

TASK 12

Find the method performs best:

```
scores = [knn_cv.best_score_,
tree_cv.best_score_,
logreg_cv.best_score_,
svm_cv.best_score_]

print(max(scores)) # tree_cv
```

0.875

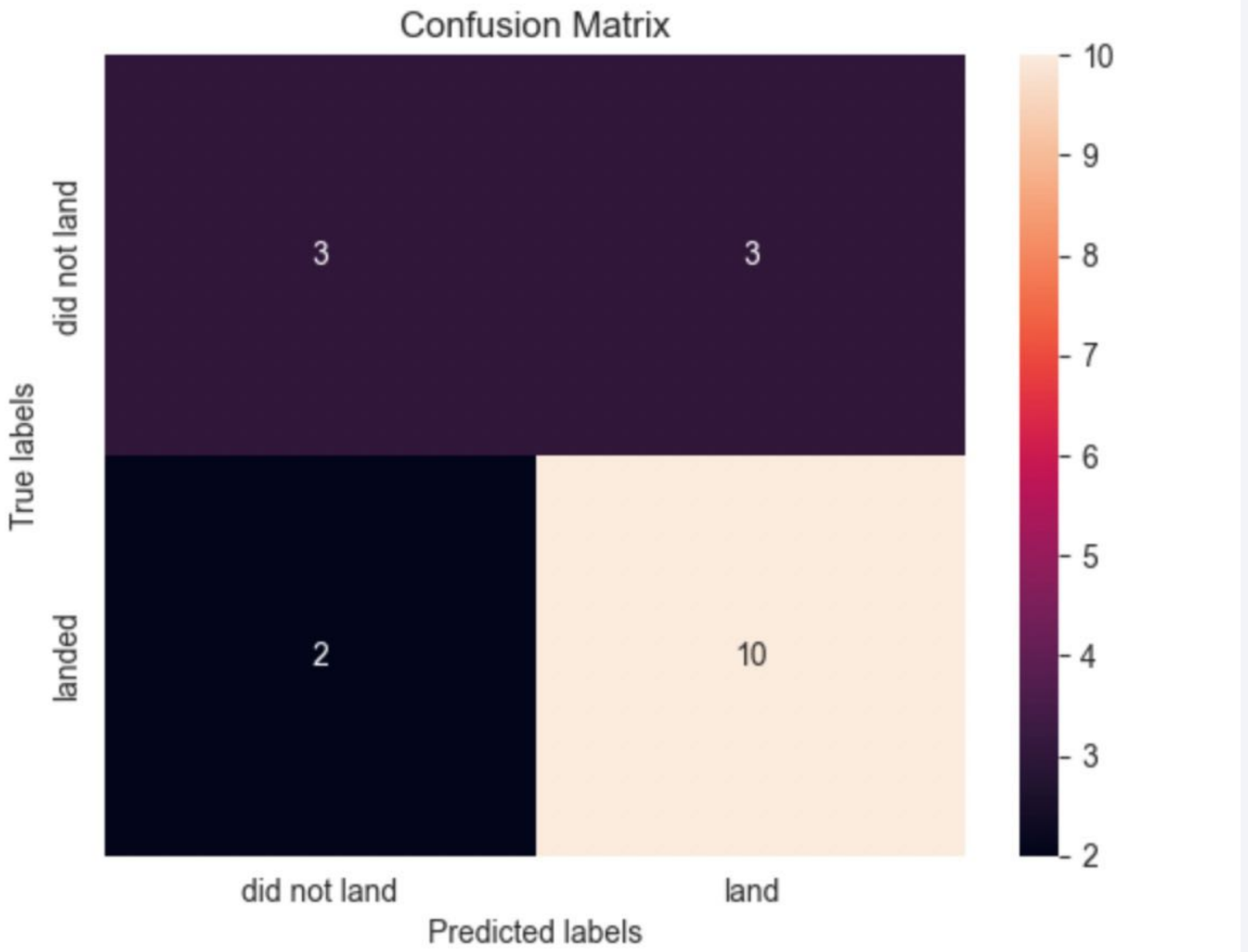
Classification Accuracy

- Desicion tree has the highest accuracy

Confusion Matrix

- Land has the highest value

```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Conclusions

- PayloadMass, and Orbit Type is the key factor
- From 2013, the success rate increased dramatically

Thank you!