IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Giancarlo Malugani
May 17, 2022

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- Project background and context
  - We will assume the role of a Data Scientist working for a startup intending to compete with SpaceX, and in the process follow the Data Science methodology involving data collection, data wrangling, exploratory data analysis, data visualization, model development, model evaluation, and reporting your results to stakeholders.
  - We will predict if the first stage of the SpaceX Falcon 9 rocket 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.
- Problems you want to find answers
  - If we can accurately predict the likelihood of the first stage rocket landing successfully, we can determine the cost of a launch. With the help of our Data Science findings and models, the competing startup we have been hired by can make more informed bids against SpaceX for a rocket launch.
- Project Repository
  - https://github.com/monkodev/DataScience_Final

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - The data was collected using SpaceX API (link shown in subsequent slides) & using webscraping technique from de web of Wikipedia
- Perform data wrangling
  - With data collected, we process of cleaning and unifying messy and complex data sets for easy access and analysis
- 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

- Data sets were collected using two of the most used methods in data science: Webscraping of web pages and connections through SpaceX API REST to obtain a JSON file.
- WebScraping
  - We collect data from Wikipedia URL
  - Extract a Falcon 9 launch records HTML table using BeautifulSoup
  - Parse the table and convert it into a Pandas data frame
- API Request
  - We will make a get request to the SpaceX API (link)
  - Clean the requested data. Check & Fill missing values

Data Collection – SpaceX API

- Calls to SpaceX API REST to collect data.
- Access Github Repo here: (Note: This notebook are in Spanish LANG): https://github.com/monkodev/DataScience_Final/blob/main/Jupyter%20Lab%20Api%20SpaceX.ipynb
- View my Jupiter Notebook from IBM Watson (click here)

Ahora comencemos a solicitar datos de lanzamiento de cohetes de la API de SpaceX con la siguiente URL:

```
In [7]: spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
In [8]: response = requests.get(spacex_url)
```

Revisamos el contenido de response para asegurarnos que contiene datos!!!!

```
In [9]: print(response.content)
```

```
In [10]: response.status_code
Out[10]: 200
```

Decodificamos el contenido del Json usando `.json()` y lo convertimos en un dataframe manipulable y operable `.json_normalize()`

```
In [13]: # Con json_normalize convertimos el resultado en un dataframe
datos = pd.json_normalize(response.json())
```

Usaremos el dataframe `datos` para mostrar las primeras 5 filas

Data Collection - Scraping

- Web Scrap Falcon 9 launch records with BeautifulSoup
  - Extract a Falcon9 launch records HTML table from WIKI
  - Parse the table & convert it into a Pandas DataFrame
- Github Repo: https://github.com/monkodev/DataScience_Final/blob/main/TP%20Final%20con%20Web%20Scrapping%20.ipynb

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

Next, request the HTML page from the above URL and get a `response` object

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

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

Create a `BeautifulSoup` object from the HTML `response`

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(datos, 'html5lib')
```

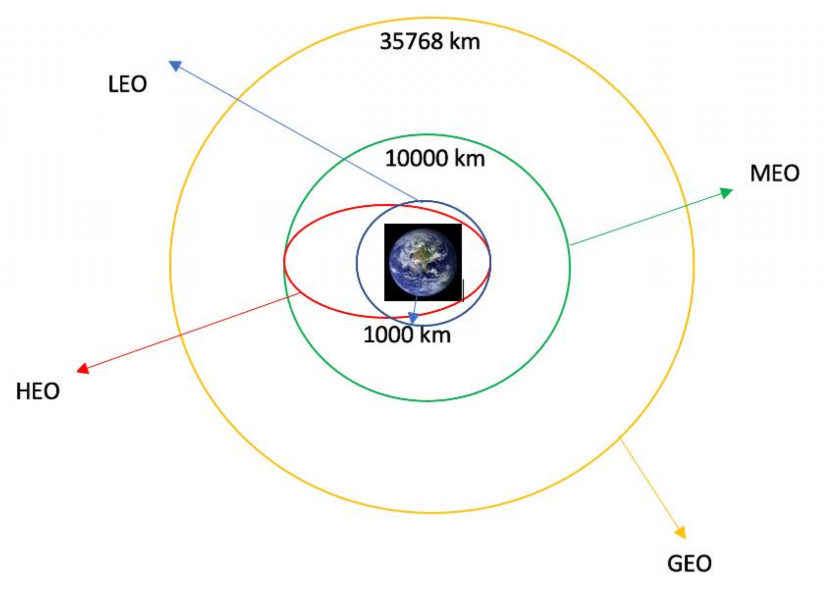
Print the page title to verify if the `BeautifulSoup` object was created properly

Data Wrangling

- Basically, the purpose of this work we had convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful
- We calculated the number of launches at each site and the number and occurrence of each orbits and other info. Finally we exported the results to csv file & save them in Github project Repo (view this file)
- View complete notebook on Github: https://github.com/monkodev/DataScience_Final/blob/main/SpaceX%20Data%20Wrangling.ipynb

```
In [7]:  # Apply value_counts() on column LaunchSite
         df['LaunchSite'].value_counts()

Out[7]:  CCAFS SLC 40    55
         KSC LC 39A      22
         VAFB SLC 4E     13
         Name: LaunchSite, dtype: int64
```

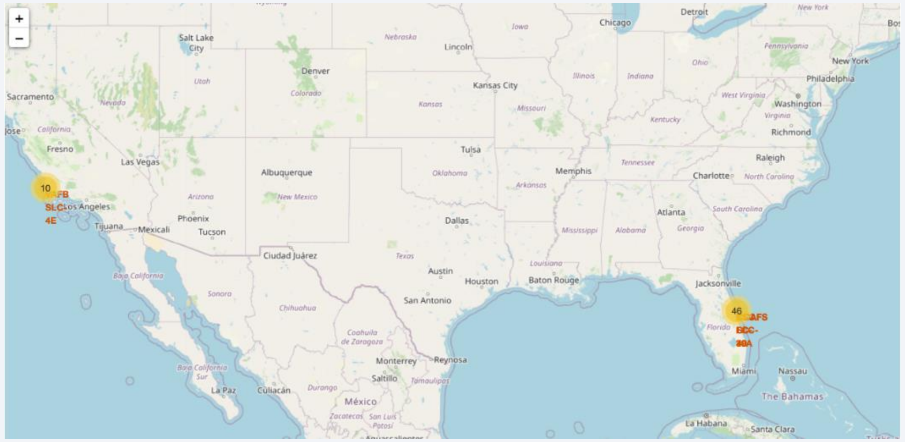


EDA with SQL

- In this notebook the work is to understand the SpaceX dataset and then load it into a table in Db2 database
- When data is loaded into database we execute some queries to answer and present important information. For example:
  - Display names of the unique launch sites in the space mission
  - Display 5 records where launch sites begin with the string 'CCA'
  - Total payload mass carried by boosters launched by NASA (CRS)
  - Average payload mass carried by booster version F9 v1.1
  - List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
  - ……. **+ 5 others SQL queries with important information**
- Link to the notebook is: https://github.com/monkodev/DataScience_Final/blob/main/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

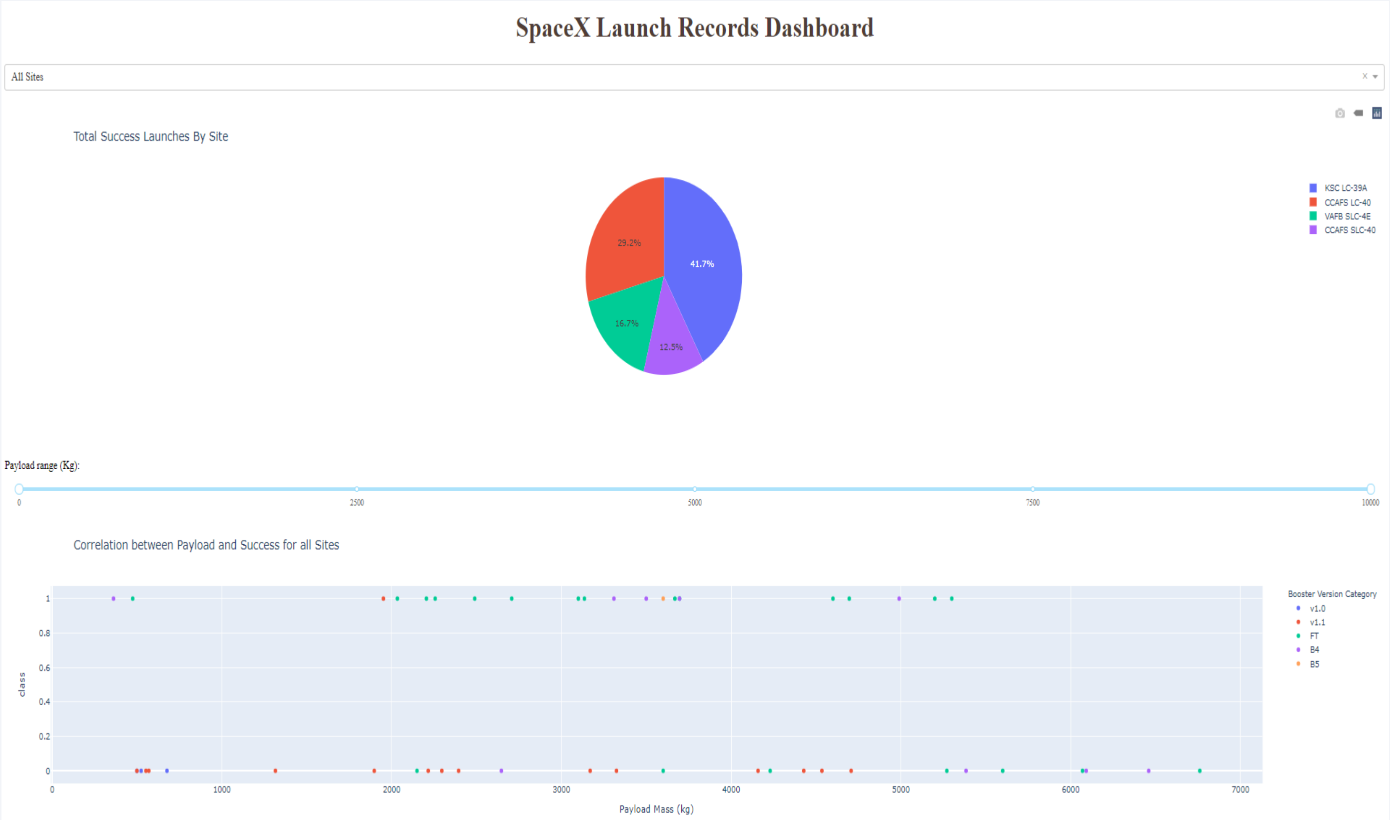
- With Folium library we added a map into a notebook and mark some facts (with circles, markers or lines) to discover important data such as:
  - All launch sites on a map
  - The success/failed launches for each site on the map
  - Calculate the distances between a launch site to its proximities
- Link to GitHub Repo: https://github.com/monkodev/DataScience_Final/blob/main/Analisis%20interactivo%20con%20Folium.ipynb



Build a Dashboard with Plotly Dash

- We Built a real-time dashboard for users to perform and analyze launch records interactively with Plotly Dash.

  - Pie chart showing the total launches by a certain sites
  - Scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version

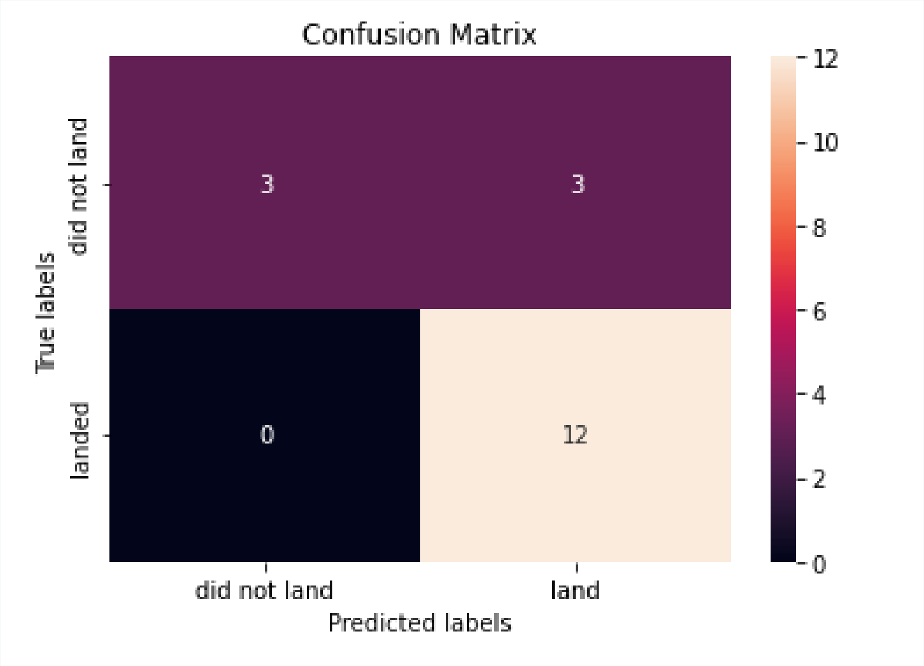


- View python code: https://github.com/monkodev/DataScience_Final/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- Perform exploratory Data Analysis and determine Training Labels
  - create a column for the class
  - Standardize the data
  - Split into training data and test data
- Find best Hyperparameter for SVM, Classification Trees and Logistic Regression
  - Find the method performs best using test data

Link to notebook:
https://github.com/monkodev/DataScience_Final/blob/main/ML%20Lab%20Malugani.ipynb



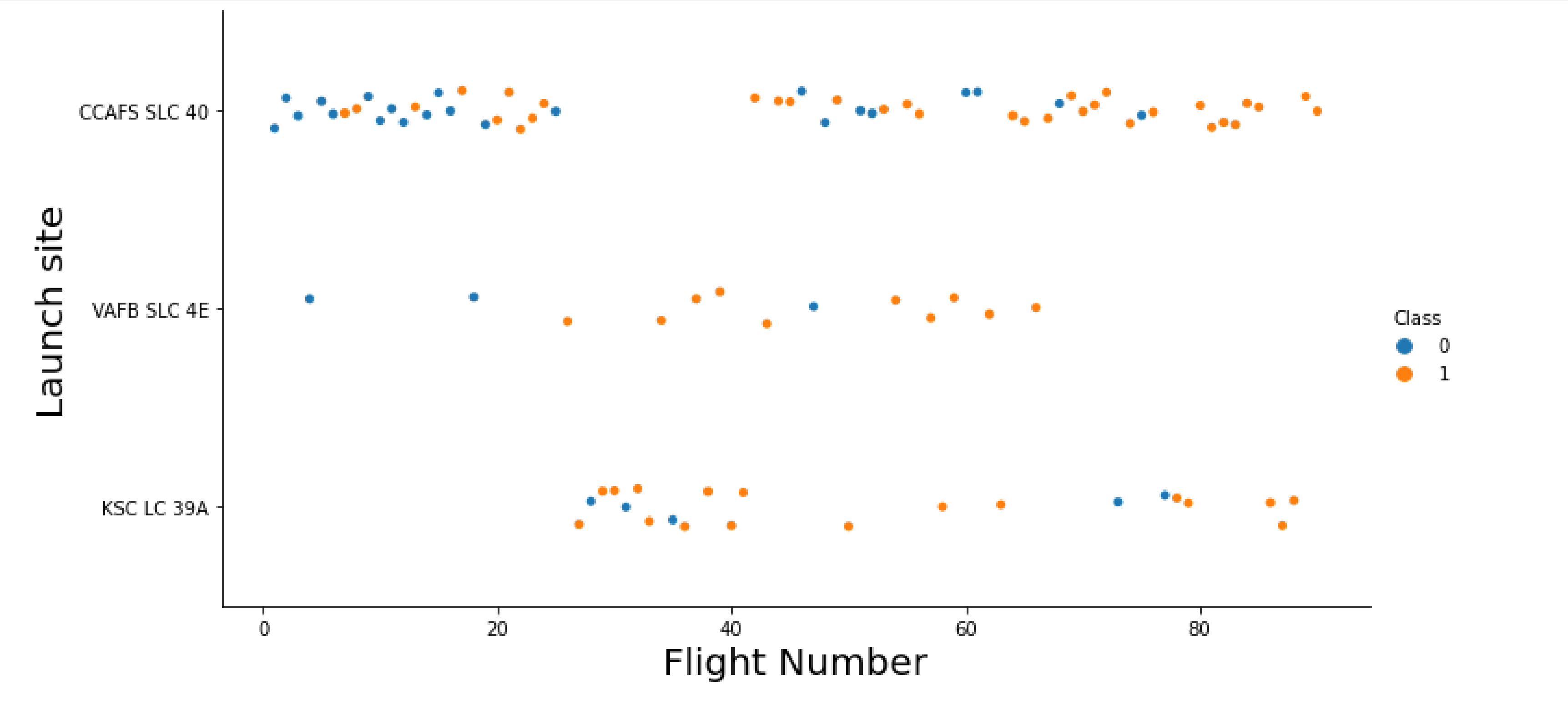
Results

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

Section 2

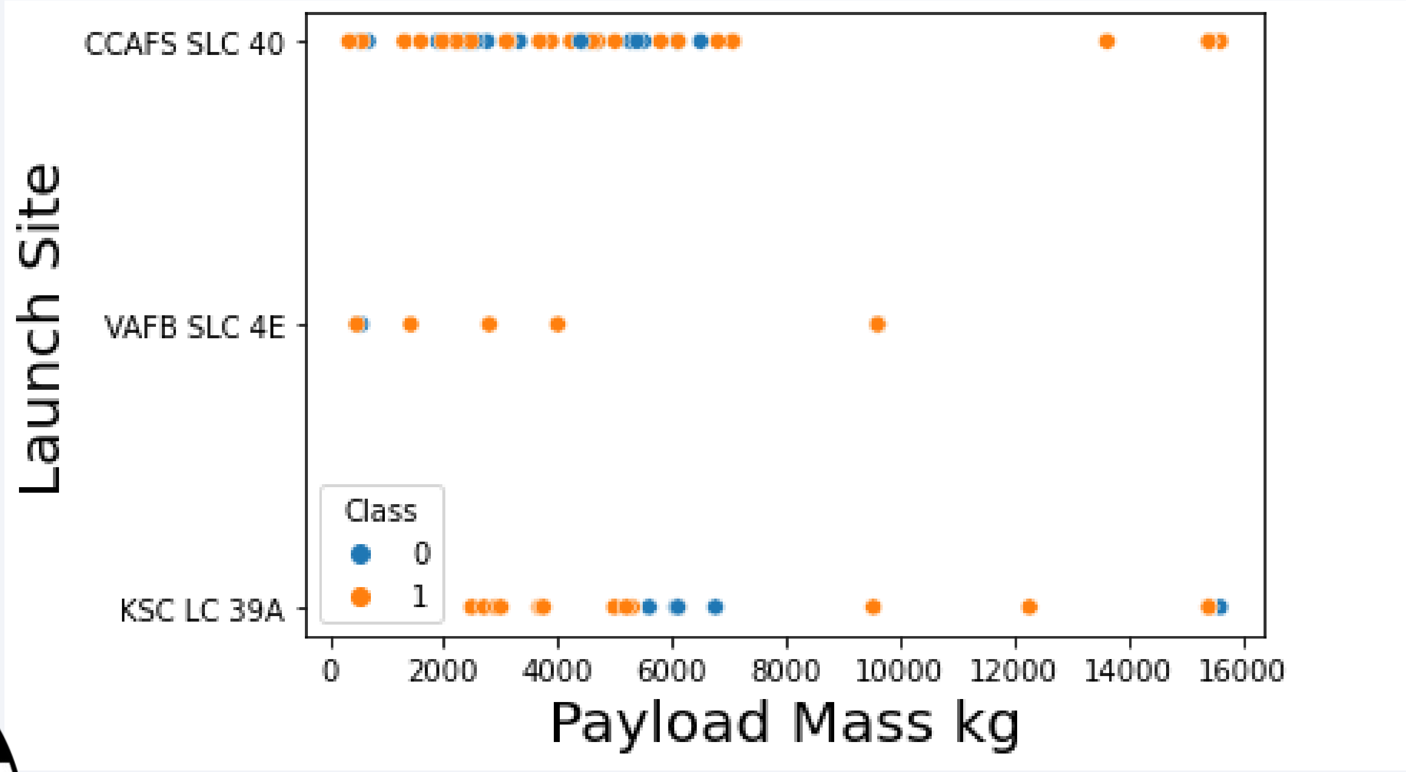
Insights drawn from EDA

Flight Number vs. Launch Site



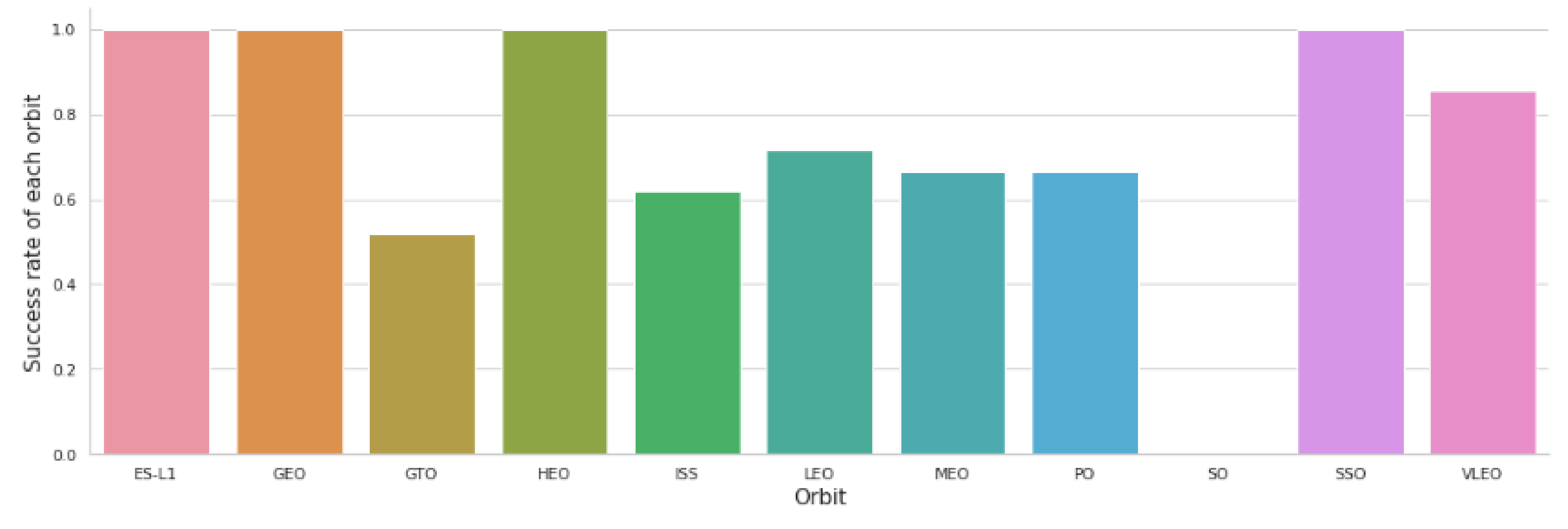
- With the graph that visualize we can conclude that the greater the amount of flight in a launch site; the higher the success rate of that site

Payload vs. Launch Site



- We find the VAFB-SLC launchsite there are no rockets launched for heavy payload mass(greater than 10000)
- In CCAFS rocket for greater mass; the greater the probability of success in the launch

Success Rate vs. Orbit Type

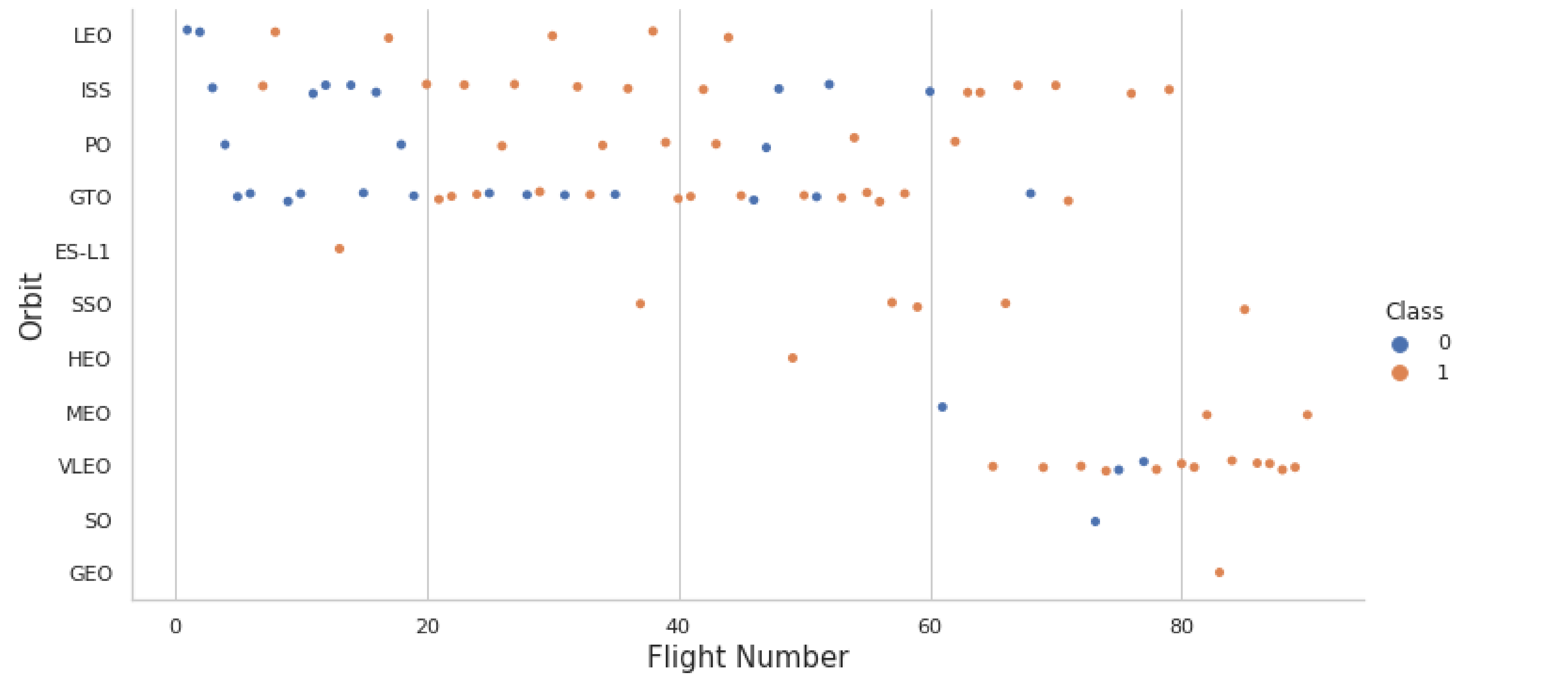



```
In [16]: bar1=df.groupby("Orbit")["Class"].mean()
         bar1

Out[16]: Orbit
         ES-L1    1.000000
         GEO      1.000000
         GTO      0.518519
         HEO      1.000000
         ISS      0.619048
         LEO      0.714286
         MEO      0.666667
         PO       0.666667
         SO       0.000000
         SSO      1.000000
         VLEO     0.857143
         Name: Class, dtype: float64
```

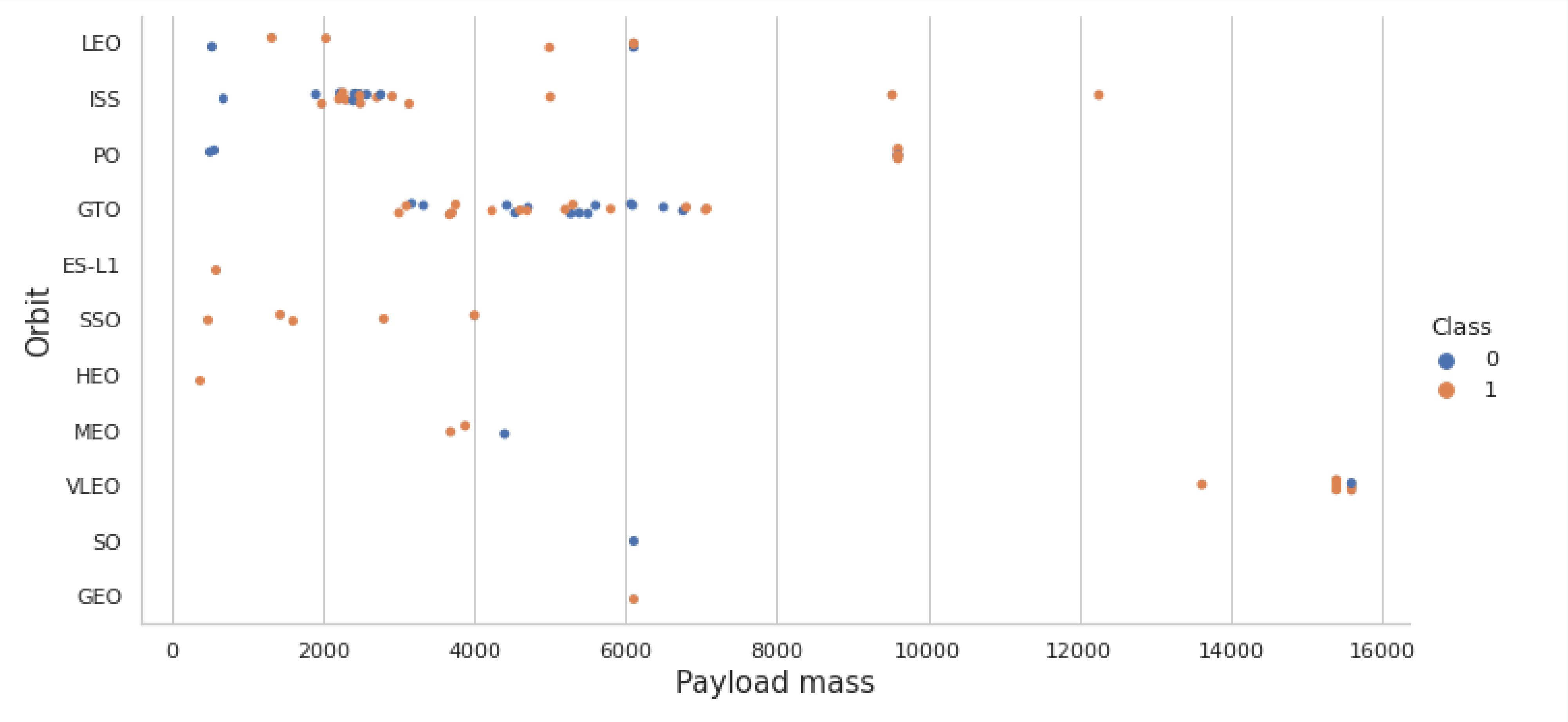
As you can see in the graph in the code that we show you, the orbits ES-L1, GEO, HEO, SSO, VLEO are the ones with the highest success rates in SpaceX space missions

Flight Number vs. Orbit Type



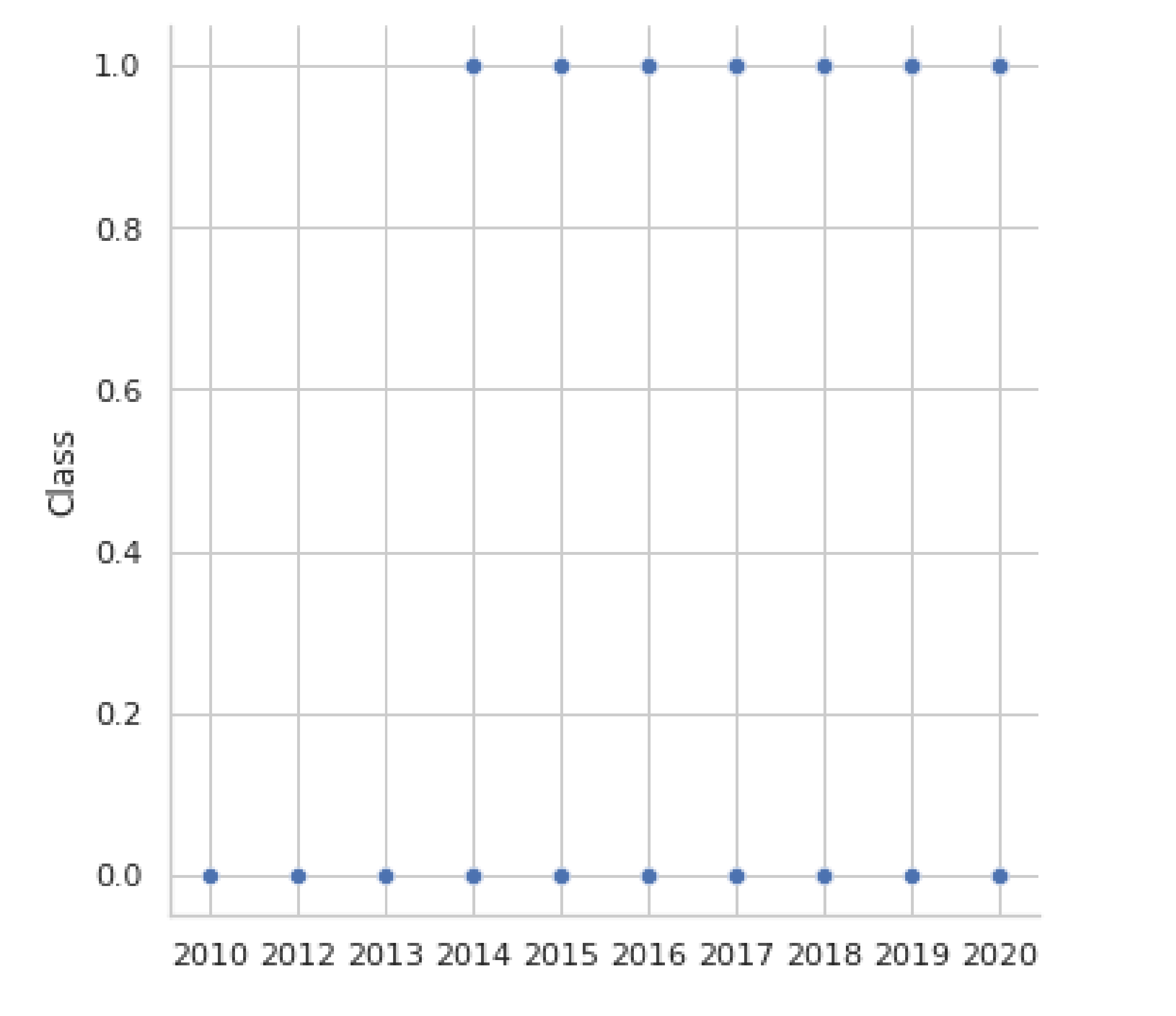
- We can see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit

Payload vs. Orbit Type



- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.

Launch Success Yearly Trend



- We can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

In [13]:

```
%sql SELECT distinct(launch_site) FROM spacex;
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[13]:

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

In [14]:

```
%sql SELECT unique(launch_site) FROM spacex;
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[14]:

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

We can use unique statement as well as distinct to show the different launch site names

Launch Site Names Begin with 'CCA'

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

In [16]:
```
%sql SELECT launch_site from SPACEX where launch_site LIKE 'CCA%' limit 5;
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[16]:

launch_site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

- We show the 5 launch sites with the limit statement. To tell sql to look for the string 'CCA' we do it with like statement

Total Payload Mass

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

In [18]:
```
%sql select sum (payload_mass__kg_) as payloadmass from spacex;
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[18]:

payloadmass
256163

result

- The total payload carried by boosters from NASA are 156.163 kg
- We use sum statement to add all the rows of the fields "payload_mass__kg_"

Average Payload Mass by F9 v1.1

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

In [19]:
```
%sql select avg(payload_mass__kg_) as payloadmass from SPACEX;
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[19]:

payloadmass
5692

- The AVG() function returns the average value of an expression. In this case display the average value of all records of payload_mass

First Successful Ground Landing Date

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

Hint:Use min function

In [20]:
```
%sql select min(date) from spacex
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[20]:

1
2010-04-06

We validate the result of the sql query searching in google!

https://spaceflightnow.com/falcon9/004/falcon9.html

https://www.britannica.com › Falcon-launch-vehicle

Falcon | Definition, History, & Facts - Encyclopedia Britannica

Buscar: When was SpaceX first successful landing?

When was Falcon 9 first successful launch?

June 4, 2010

Falcon 9 saw its first successful launch on **June 4, 2010**, and its second on December 8, 2010. Quick Facts: Made in America: All structures, engines, avionics, and ground systems designed, manufactured and tested in the United States by SpaceX. Falcon 9 with a Dragon spacecraft is 48.1 meters (157 feet) tall.

https://spaceflightnow.com › falcon9 › falcon9

SpaceX Falcon 9 rocket facts - Spaceflight Now

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

In [21]:
```
%sql select booster_version from spacex where landing__outcome='Success (drone ship)' and payload_mass__kg_ BETWEEN 4000 and 6000;
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[21]:

booster_version
F9 FT B1022
F9 FT B1031.2

- We used the where clause to filter for boosters which have successfully landed on drone ship and applied the and condition to determine successful landing with payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

In [22]:
```
%sql select count(mission_outcome) as missionoutcomes from spacex GROUP BY mission_outcome;
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[22]:

missionoutcomes
44
1

We count the number of failed records with the count() clause.
We must necessarily use the group by to indicate the index field

Boosters Carried Maximum Payload

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

In [23]:
```
%sql select booster_version as boosterversion from spacex where payload_mass__kg_=(select max(payload_mass__kg_) from spacex);
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[23]:

boosterversion
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3

- Using subqueries we show the versions of the boosters that had the maximum weight recorded. We use max() clause.

2015 Launch Records

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

In [24]:

```
%sql SELECT MONTH(DATE),MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE FROM SPACEX where EXTRACT(YEAR FROM DATE)='2015';
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[24]:

1	mission_outcome	booster_version	launch_site
10	Success	F9 v1.1 B1012	CCAFS LC-40
11	Success	F9 v1.1 B1013	CCAFS LC-40
2	Success	F9 v1.1 B1014	CCAFS LC-40

The month() function returns the month of a date datetime() .
In the same way "extract (year from date)" informs us of the year to compare it with our searched year = 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

In [25]:
```
%sql SELECT LANDING__OUTCOME FROM SPACEX WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY DATE DESC;
```

```
 * ibm_db_sa://kxh79332:***@55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31929/bludb
Done.
```

Out[25]:

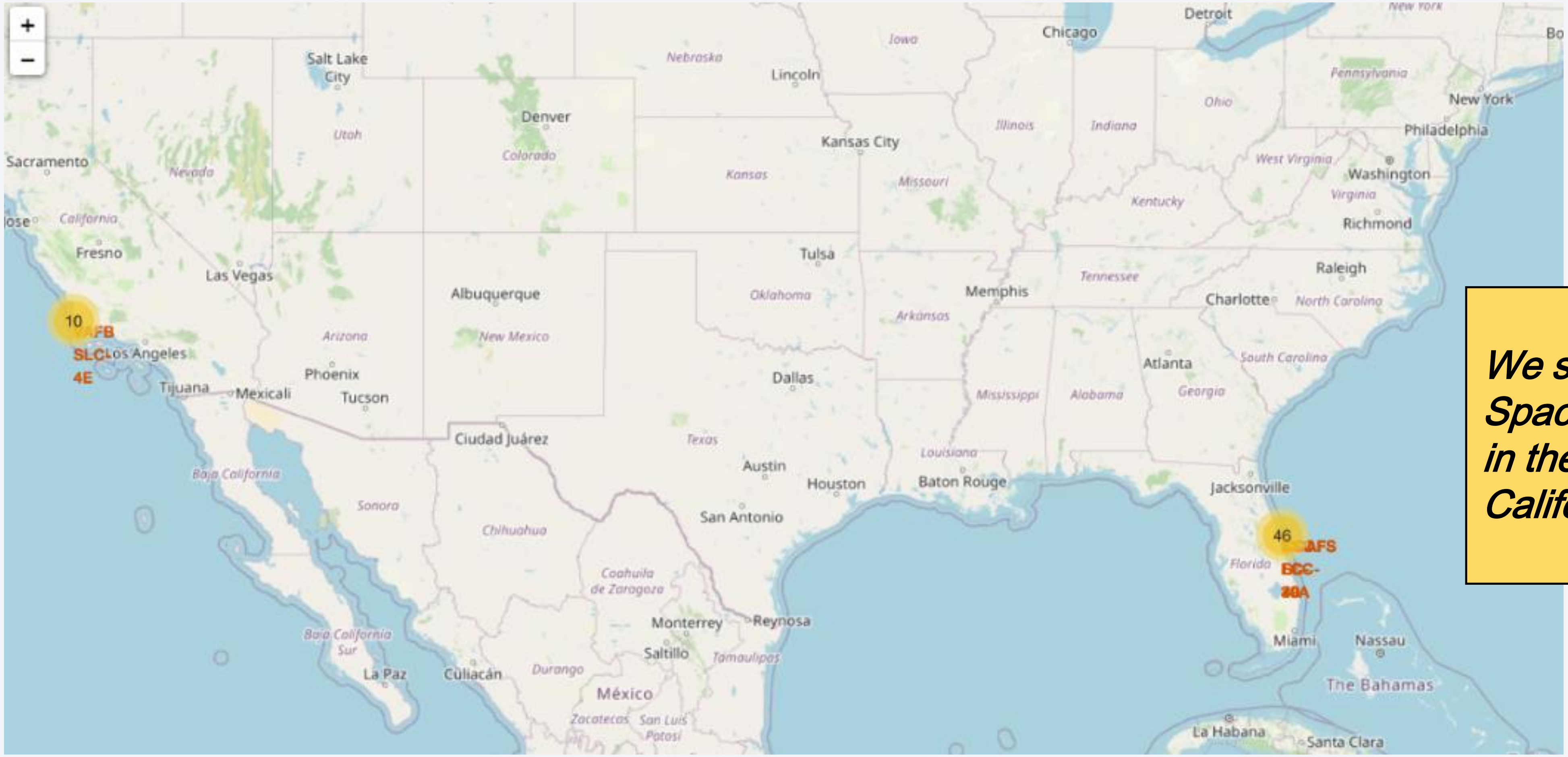
landing__outcome
Success (ground pad)
Success (ground pad)
Success (drone ship)
Success (drone ship)
Failure (drone ship)
Controlled (ocean)
Failure (drone ship)
No attempt
No attempt
No attempt
No attempt
No attempt
No attempt
No attempt
Failure (parachute)

- Select the landing oucomes and we use the where clause to filter dates.
- Using between clause we can select records into two dates range.

Section 3

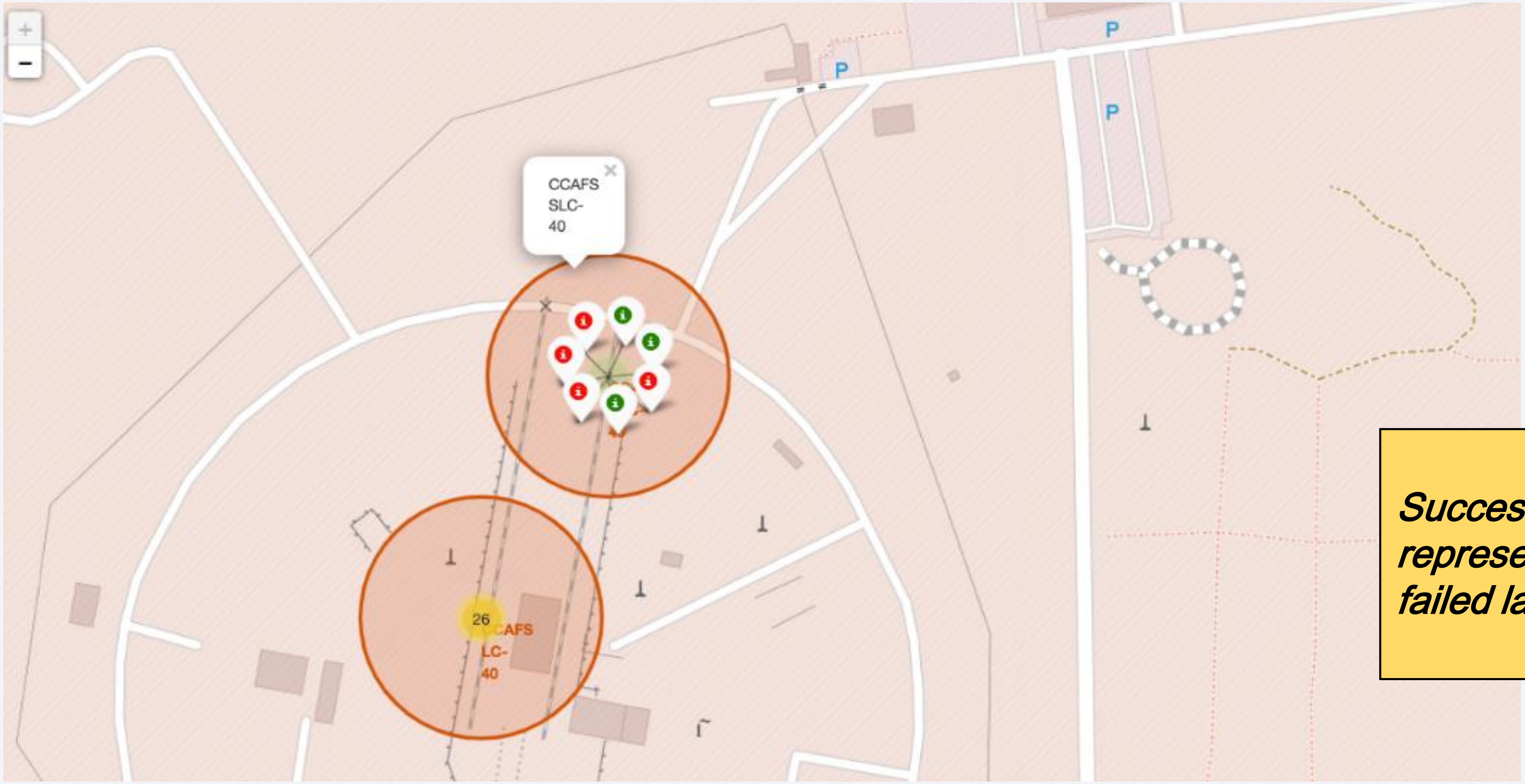
Launch Sites Proximities Analysis

North America Launch Map



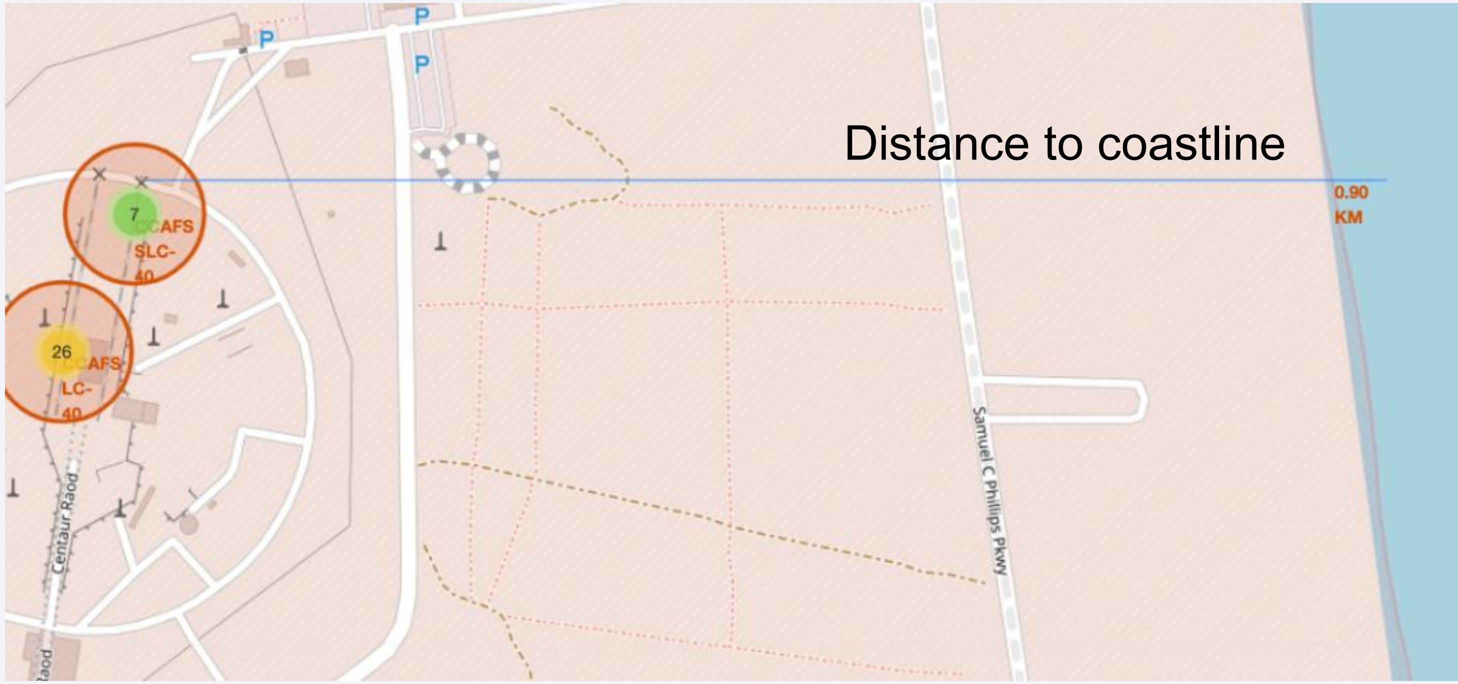
We see on the map that SpaceX launches have been in the states of Florida and California

Florida Launch Site



Successful launches are represented in green and failed launches in red.

Proximities to launch sites



- 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

- We calculate distance to Melbourne - Australia

In [20]:

```
# Create a marker with distance to a closest city, railway, highway, etc.
# Draw a line between the marker to the launch site
lat_melbourne = 28.070061
long_melbourne = -80.561104
distance_melbourne = calculate_distance(launch_sites_df.iloc[0]["Lat"], launch_sites_df.iloc[0]["Long"], lat_melbourne, long_melbourne)
distance_melbourne
```

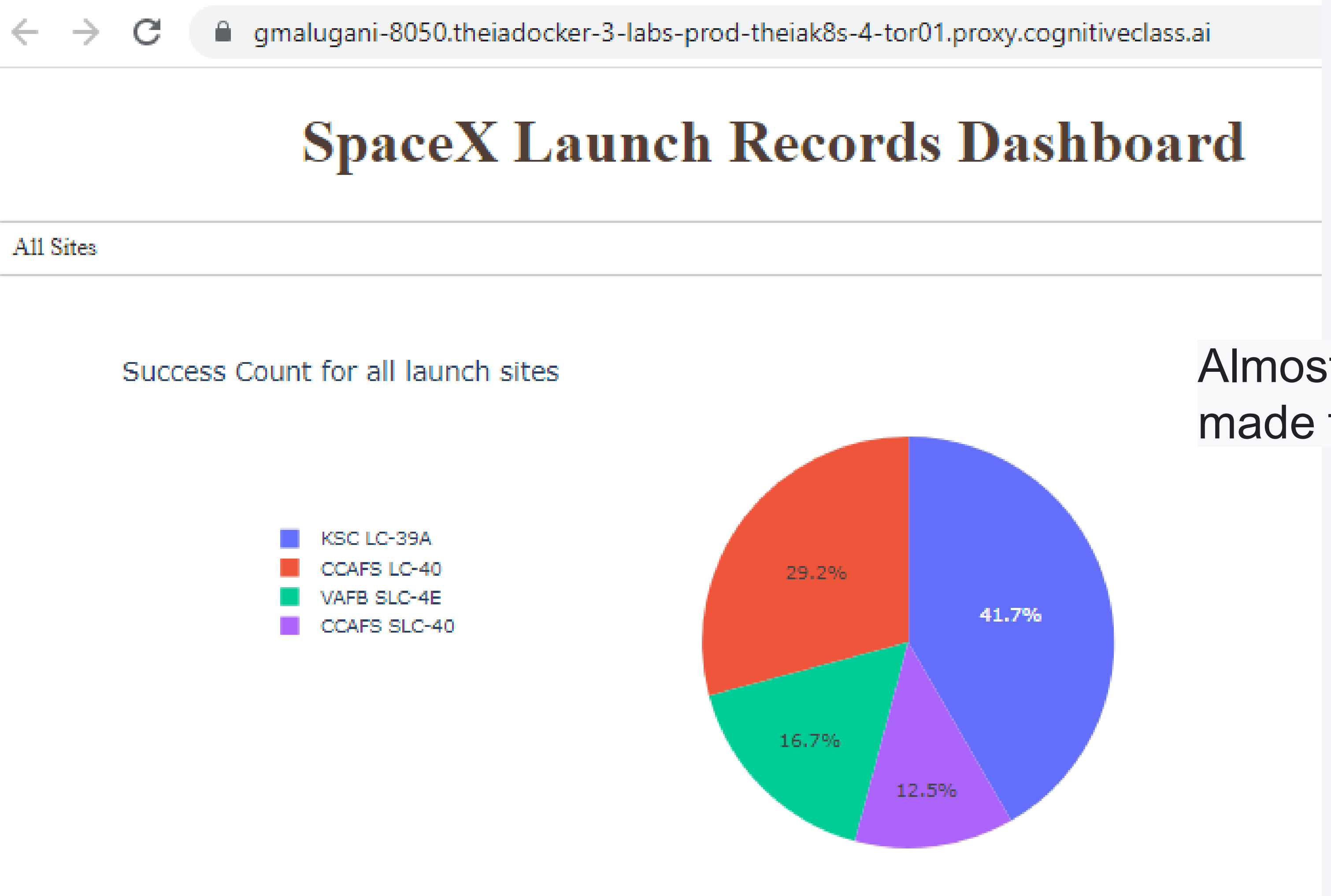
Out[20]:

```
54.77500486370945
```

Section 4

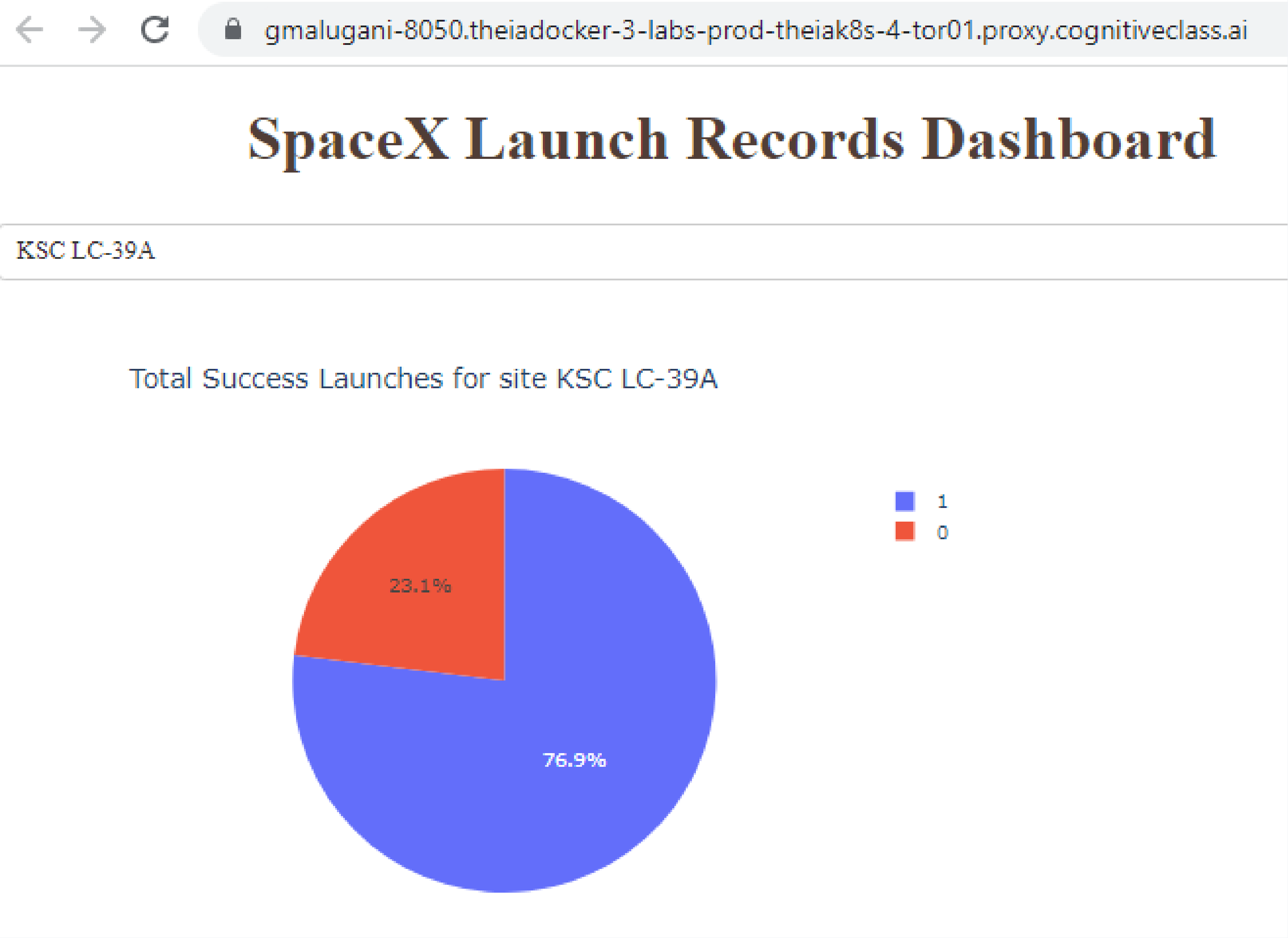
Build a Dashboard with Plotly Dash

Total Success Launches by all sites



Almost half of the launches were made from KSCLC (42%)

Site with highest probability of success

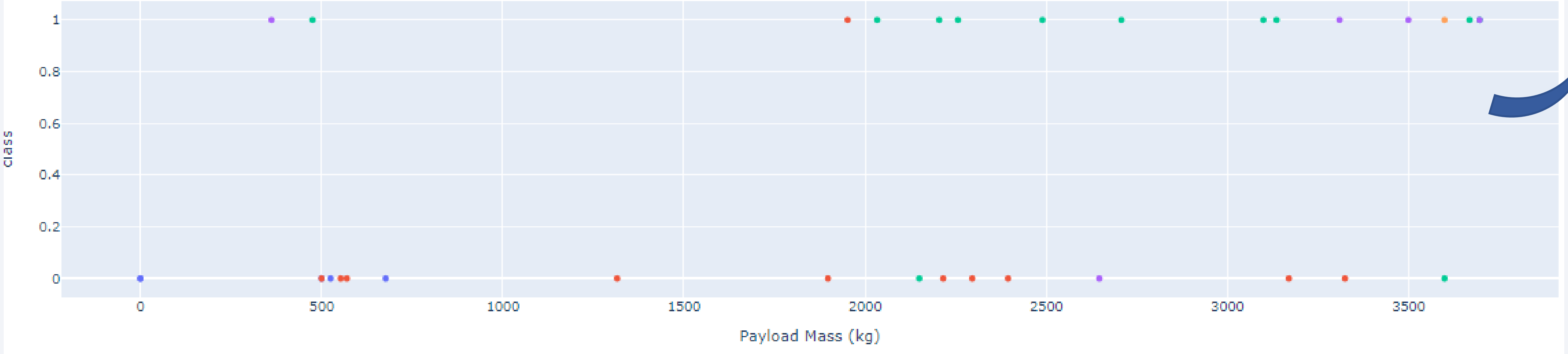


Site KSC LC-39 A have 77% of success in each launch

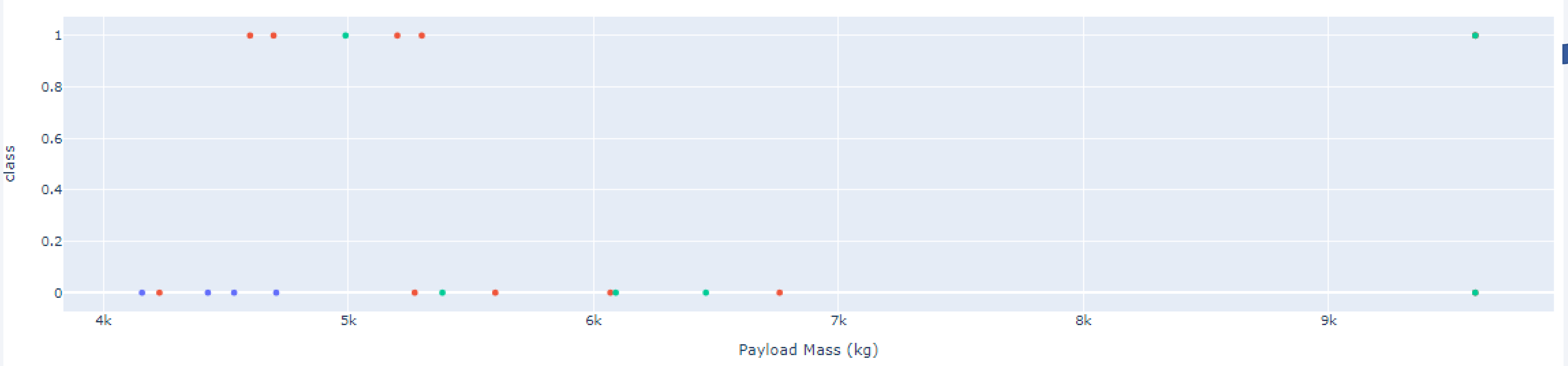
Comparision of loads

With lighter loads there is a greater probability of success

Low Loads (range: 0-4000 kg)



High Loads (range: 4000-10000 kg)



Section 5

Predictive Analysis (Classification)

Classification Accuracy

In [31]:
```
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]}

tree = DecisionTreeClassifier()
```

In [32]:
```
grid_search = GridSearchCV(tree, parameters, cv=10)
tree_cv = grid_search.fit(X_train, Y_train)
```

In [33]:
```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 8, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
accuracy : 0.8732142857142857
```

In [35]:
```
print('Accuracy for Logistics Regression method:', logreg_cv.score(X_test, Y_test))
print( 'Accuracy for Support Vector Machine method:', svm_cv.score(X_test, Y_test))
print('Accuracy for Decision tree method:', tree_cv.score(X_test, Y_test))
print('Accuracy for K nearsdt neighbors method:', knn_cv.score(X_test, Y_test))
```

```
Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.6111111111111112
Accuracy for K nearsdt neighbors method: 0.8333333333333334
```

The decision tree classifier is the model with the highest classification accuracy

Confusion Matrix

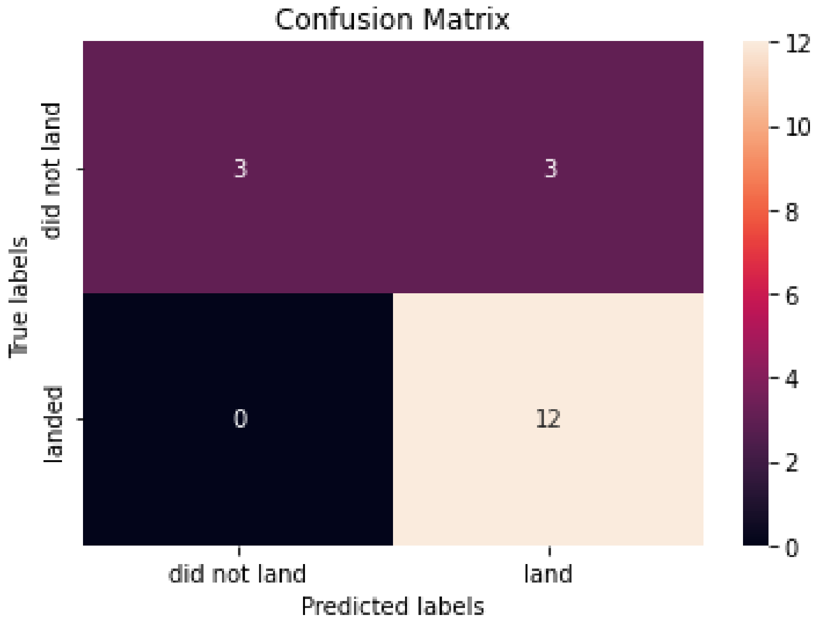
The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different clases.

In [34]:
```
tree_cv.score(X_test, Y_test)
```

Out[34]:
```
0.6111111111111112
```

We can plot the confusion matrix

In [24]:
```
yhat = svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Conclusions

- The higher the number of launches on a launch site, the higher the success rate on a launch site
- The launch success rate started to increase from 2014
- ES-L1, GEO, HEO, SSO, VLEO orbits had the highest success rate
- KSC LC-39A had the most successful launches of all sites
- Decision tree classifier is the best machine learning algorithm for this research

Appendix

- To solve all the labs it was extremely necessary to consult the IT bible : Stackoverflow (www.stackoverflow.com)
- Learn here Pandas Tutorial (https://www.w3schools.com/python/pandas/default.asp)
- https://spaceflightnow.com/falcon9/004/falcon9.html
- https://docs.spacexdata.com/

Thank you!