IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Tarik Horani
11.10.2021

Outline

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

Executive Summary

- **Summary of methodologies**
  - Data collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building interactive map via Folium
  - Building API with Plotly Dash
  - Predictive analysis
- **Summary of all results**
  - Exploratory data analysis report
  - Interactive analytics demo (screenshots)
  - Predictive analysis results

Introduction

- Project background and context
  - We want to predict the successfully landing of Falcon9 first stage rocket, Space X 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 Space X can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch.

- The problems we are trying to solve :
  - Inputs that's influences the landing successful class
  - The relations among all the rocket variable that effect the landing class (successful or failure)
  - Achieving the best conditing to spacex Falcon9 to land successfully

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API
  - Webscrapping (Wikipedia)
- Perform data wrangling
  - One hot encoding data fields for machine learning and dropping irrelative columns
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Visualization using Plotly : Scatter Graphs , Bar graphs showing the relation between the variables
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- **The Data sets were collecting as follows :**
  - Gathering Launching Data from spaceX REST API
  - The gathered data includes information about :the used rocket , launching site , payload mass , landing condition , launching condition
  - The data will be used to predict if we will get a successful or failure landing
  - The REST API URLs start with api.spacexdata.com/v
  - We also applied the web scrapping on Wikipedia tables to get the data via BeautifulSoup

Data Collection – SpaceX API

1- getting the API response

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

2-coverting the response to JSON file and normalize json file to be dataframe :

```
data = pd.json_normalize(response.json())
```

3- applying functions to clean the data :

```
getBoosterVersion(data)  getLaunchSite(data)
getCoreData(data)  getPayloadData(data)
```

4- assign list to dictionary then dataframe:

```
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}
```

```
df= pd.DataFrame(launch_dict)
```

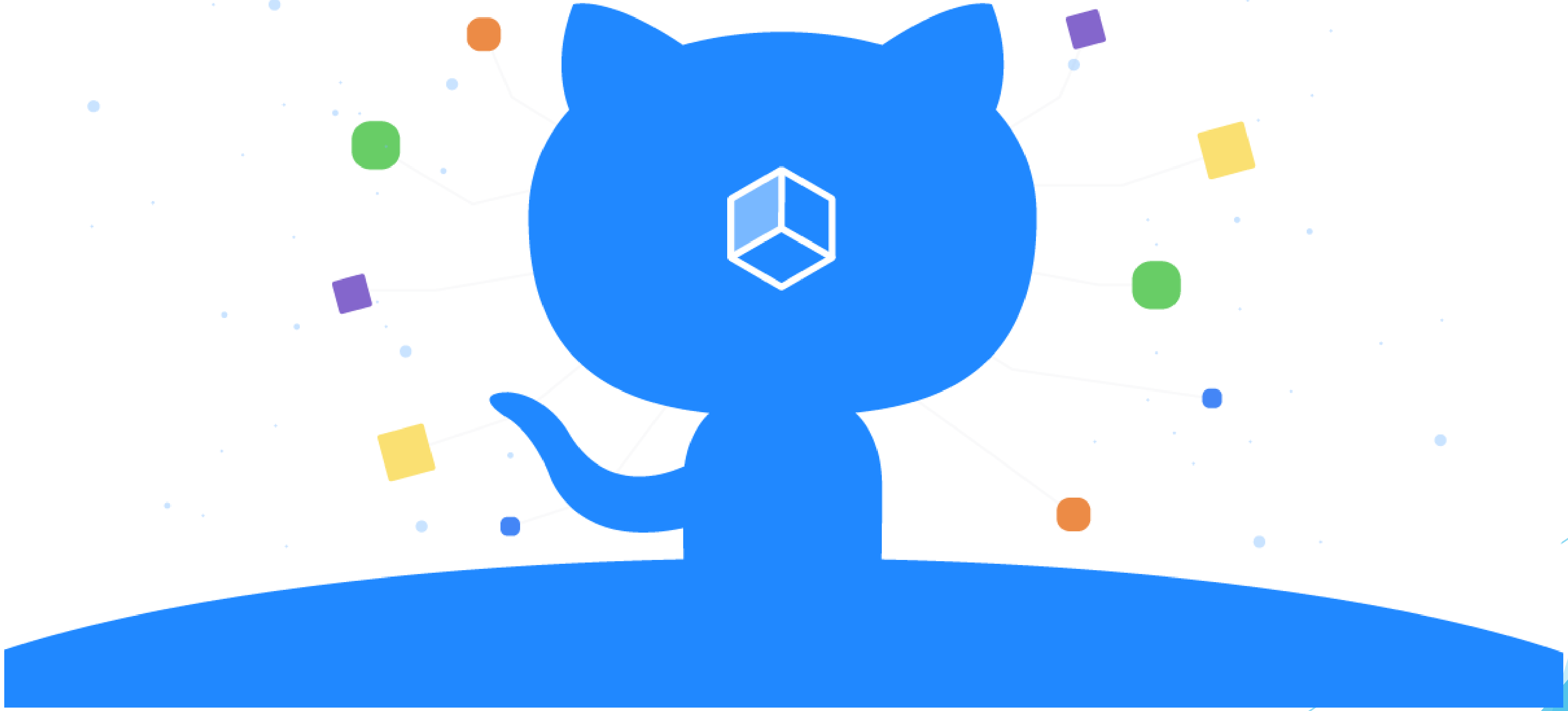
5- filt

```
data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].fillna(mean)
```

```
data_falcon9.to_csv('dataset_part\_1.csv', index=False)
```

GitHub URL

https://github.com/Thorani/capstone2/blob/master/jupyter-labs-spacex-data-collection-api%20.ipynb



Data Collection - Scraping

1- getting the response from the website :

```
response= requests.get(static_url)
response.status_code
```

2- creating BeautifulSoup object :

```
soup= BeautifulSoup(response.content , "html.parser")
```

3-finding all tables:

```
html_tables = soup.find_all("table")
```

4-getting columns names :

```
column_names = []

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
```

5- Create a data frame by parsing the launch HTML tables

```
launch_dict= dict.fromkeys(column_names)
del launch_dict['Date and time ( )']
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']=[]
```

6-appending data to keys:

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
```

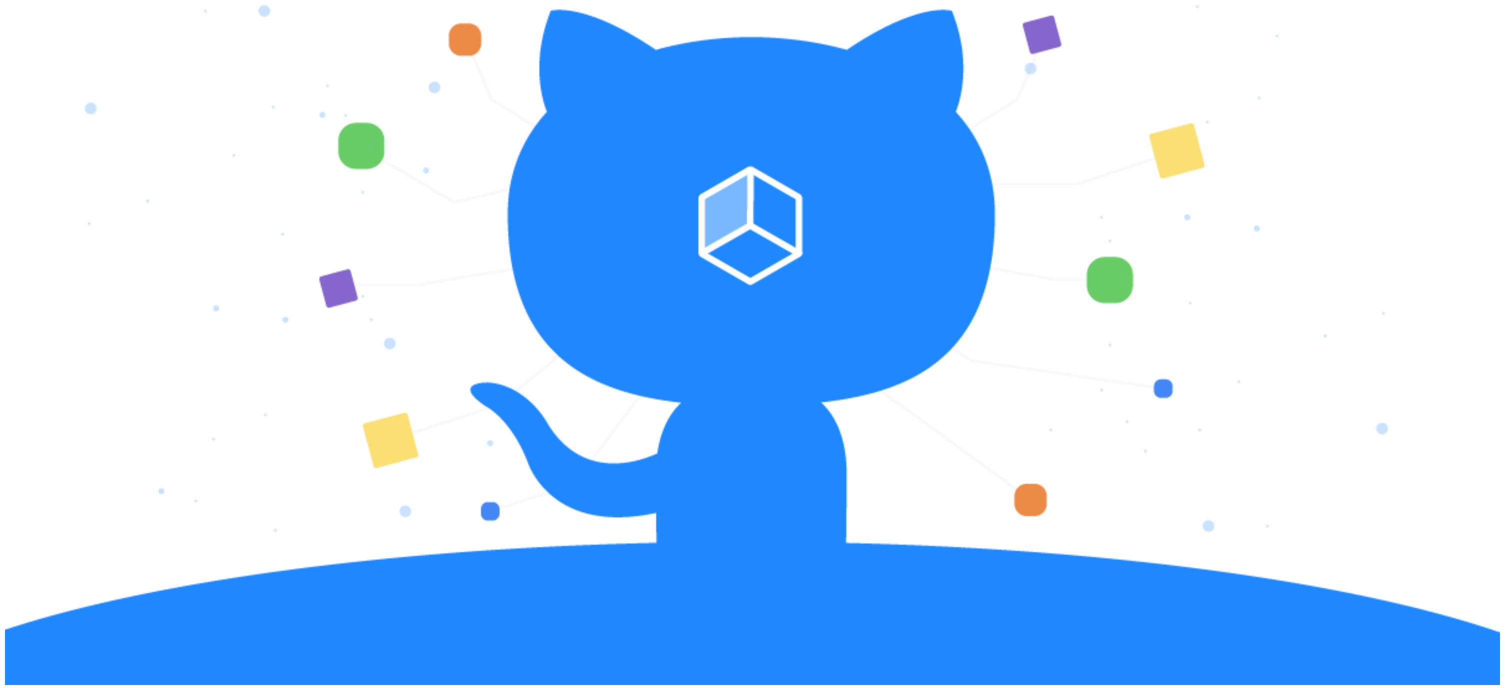
7-converting dictionary to DF and export to csv

```
df = pd.DataFrame.from_dict(launch_dict)
```

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

GitHub URL

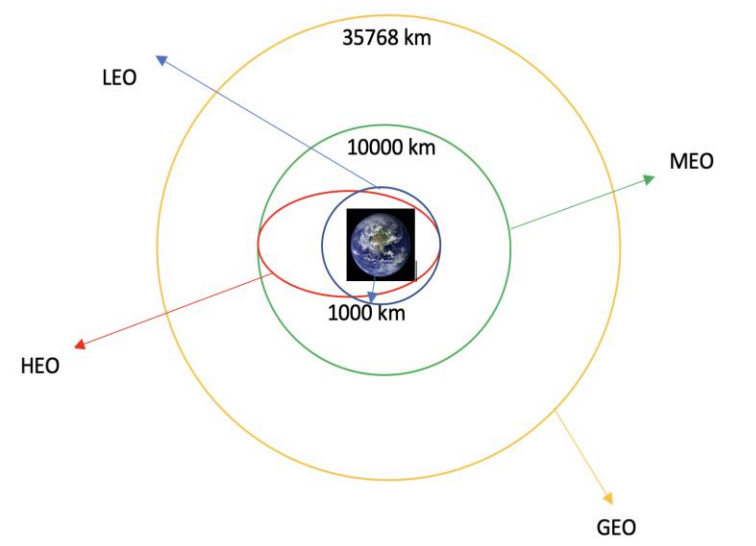
https://github.com/Thorani/capstone2/blob/master/jupyter-labs-webscraping%20(1).ipynb



Data Wrangling

- In the data set, there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean. True RTLS means the mission outcome was successfully landed to a ground pad False RTLS means the mission outcome was unsuccessfully landed to a ground pad.True ASDS means the mission outcome was successfully landed on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship.

the orbits mentioned in data set:



Work flow :

Perform Exploratory data analysis on dataset

Calculate the number of launches on each site

Calculate the number and occurrence of each orbit

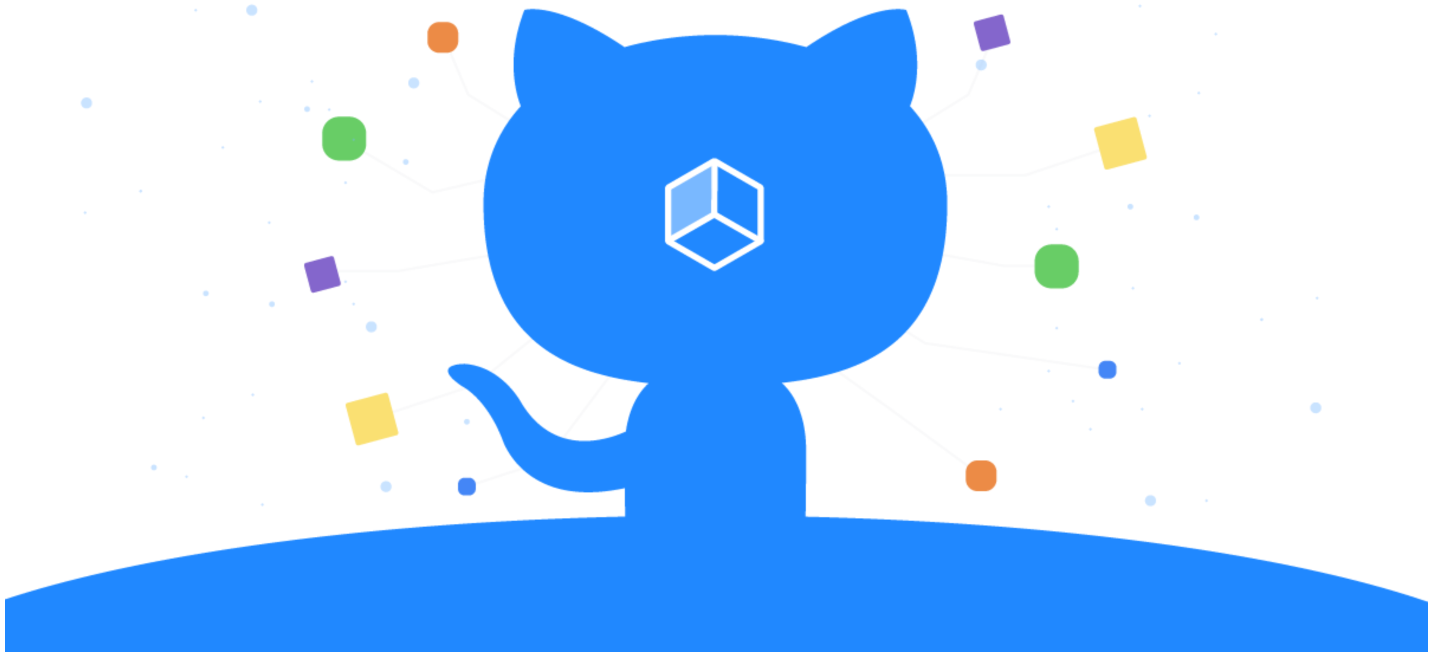
Calculate the number and occurrence of mission outcome per orbit type

Create a landing outcome label from Outcome column

determine the success rate
And export data to CSV file

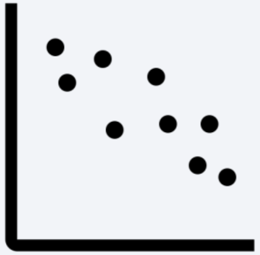
GitHub URL

https://github.com/Thorani/capstone2/blob/master/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

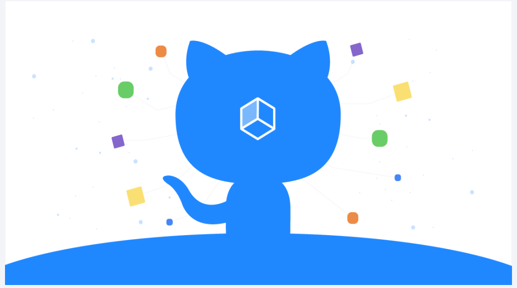
- Scatter plot :
  - Flight number vs launch site
  - Payload mass vs launch site
  - Flght number vs orbit
  - Payload mass vs orbit



Scatter polts show the effect of one variable to another

The relation between variables called correlated

It is used when we have a large data



https://github.com/Thorani/capstone2/blob/master/jupyter-labs-eda-dataviz.ipynb

Bar graph :

Mean vs Orbit :

bar graphs are used to Analyze the ploted bar chart try to find which orbits have high success rate.

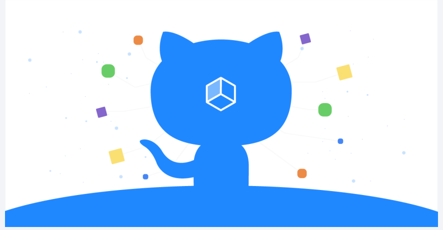
- Line Graph :
  - Success rate vs year

The line graphs are used when we have a linear relationship between the variables that makes it easy to predict the next result

EDA with SQL

- **Using SQL queries to gather and wrangling the dataset as follows:**

  - 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 achieved.
  - 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/Thorani/capstone2/blob/master/jupyter-labs-eda-sql-coursera.ipynb

Build an Interactive Map with Folium

- In order to visualize the launching sites on map : we used the latitude and the longitude coordinates for each site and displaied it with a circle and a pop up with the name of the site and also we used mouse markers with 2 colors RED for the failure launch (class 0) and GREEN for successful launch (class1) in the MarkCluster()
- Also we calculated the distance between the launch sites and the coast ,highways , trails , and Florida .
- We used lines for displaying the mentioned distances

.by the use of the math library with the follows formula :

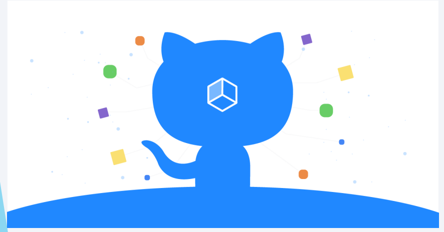
```
from math import sin, cos, sqrt, atan2, radians

def calculate_distance(lat1, lon1, lat2, lon2):
    # approximate radius of earth in km
    R = 6373.0

    lat1 = radians(lat1)
    lon1 = radians(lon1)
    lat2 = radians(lat2)
    lon2 = radians(lon2)

    dlon = lon2 - lon1
    dlat = lat2 - lat1

    a = sin(dlat / 2)**2 + cos(lat1) * cos(lat2) * sin(dlon / 2)**2
    c = 2 * atan2(sqrt(a), sqrt(1 - a))

    distance = R * c
    return distance
```

https://github.com/Thorani/capstone2/blob/master/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- Using plotly dash to build an interactive dashboard so we can
- change the variables to get new results without coding new lines each time

- The Graphs used in the Dashboard :
  - Pie chart : to display the percentage of the successful launches of each site
    - It can illustrate the numerical proportions in data
  - Scatter plot : shows the relationship between the outcome class and the payload mas
    - It can be used to show the non linear relation between variables
    - Easy to clustering and classification the date



https://github.com/Thorani/capstone2/blob/master/spacex_dash_app.py

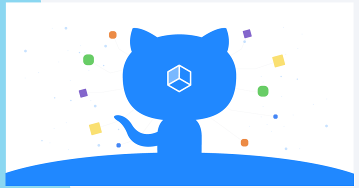
Predictive Analysis (Classification)

Building the Model
- Loading data in numpy and pandas
- Transform data
- Train test split
- Fit data set into GridSearchCV

Evaluating the model
- Check accuracy for each model
- Drawing the confusion Matrix

Improving the model
- Feature engineering
- Algorithm tuning

Select the best Model with the highest accuracy



https://github.com/Thorani/capstone2/blob/master/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

- Exploratory data analysis results

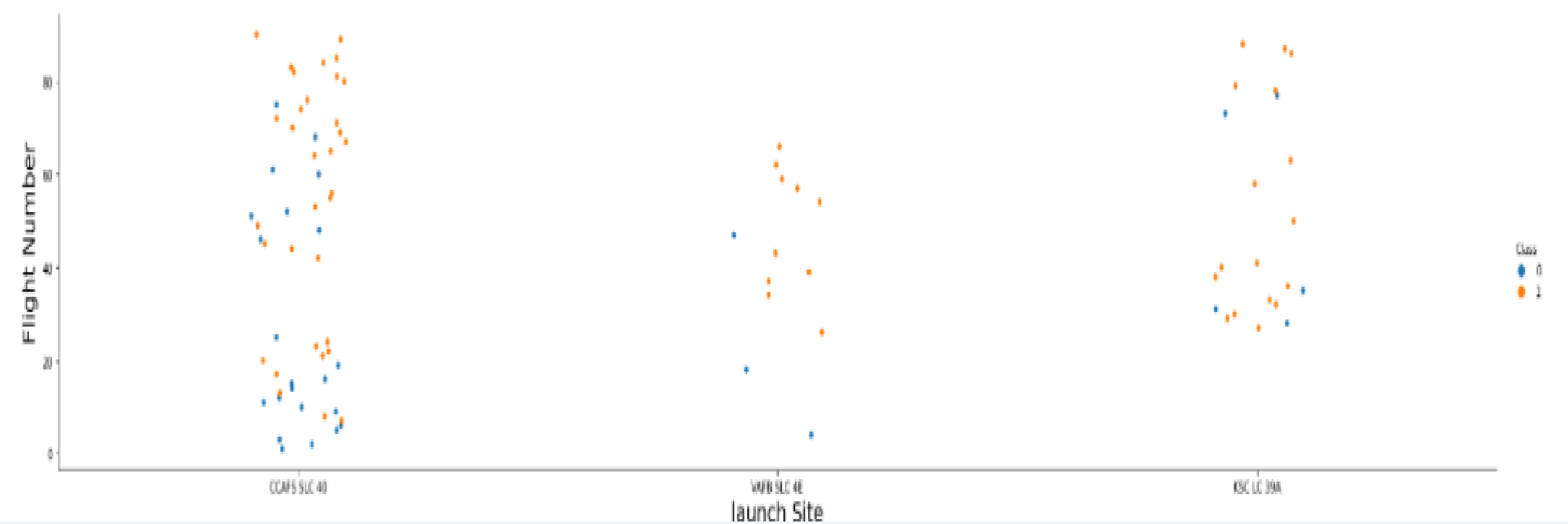
- Interactive analytics demo in
- screenshots

- Predictive analysis results

Section 2

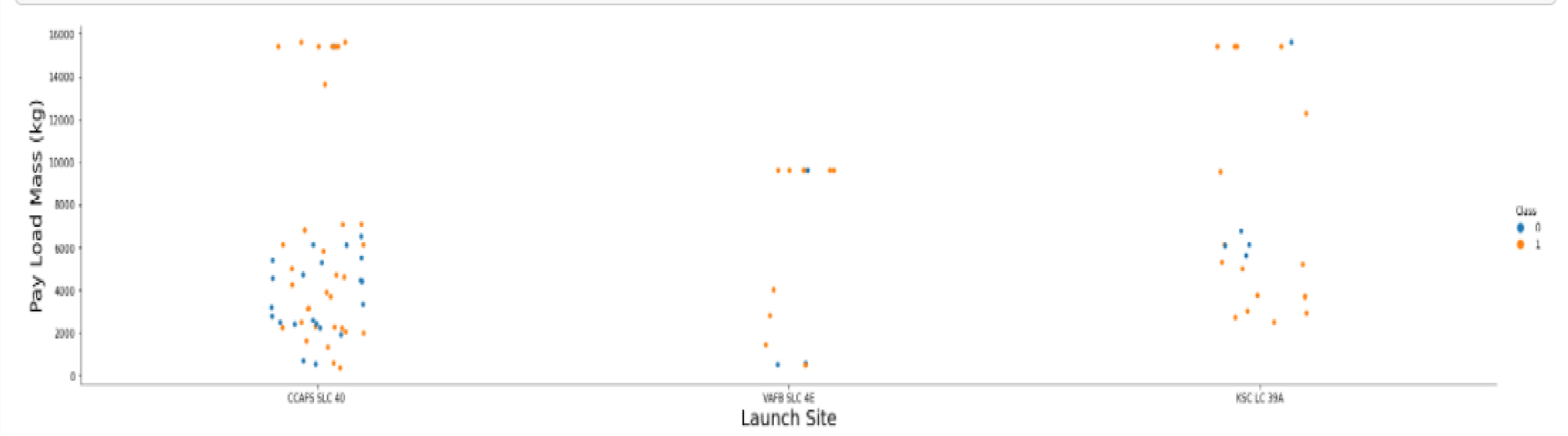
Insights drawn from EDA

Flight Number vs. Launch Site



The success rate increases with the increasing of the flights in the same launch site

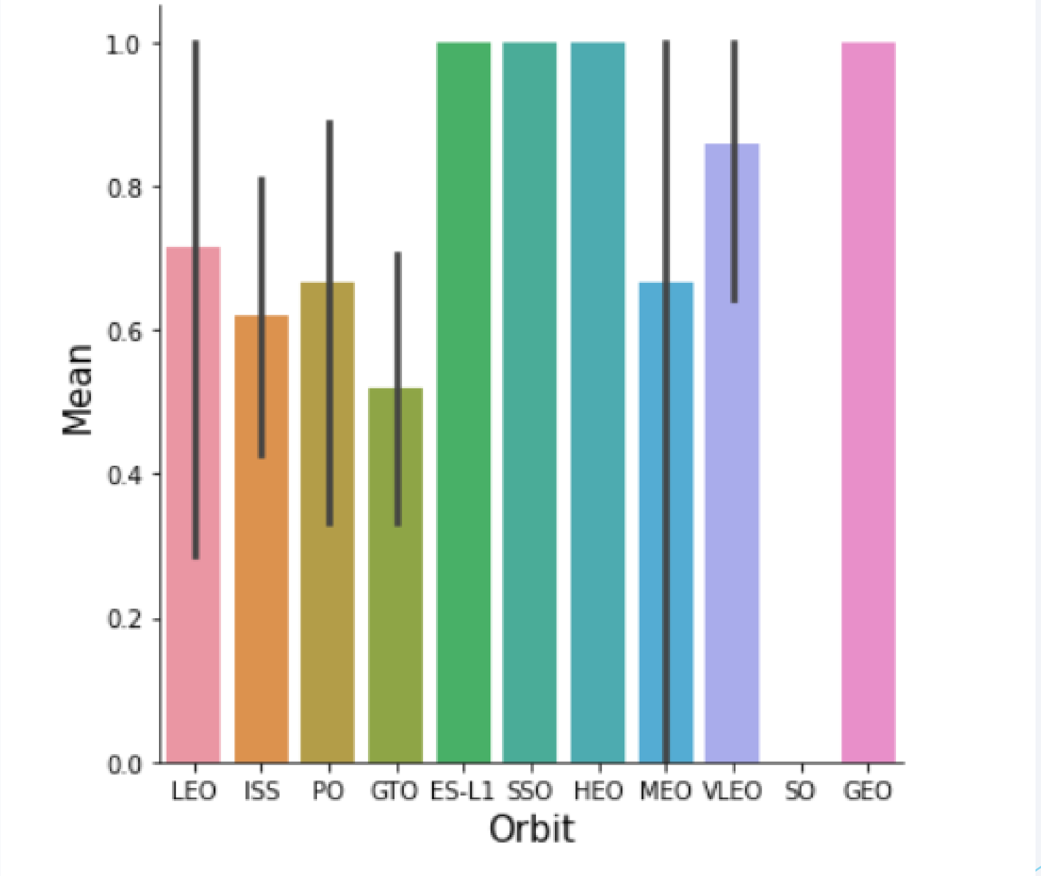
Payload vs. Launch Site



the higher payloads for the site CCAFS SLC 40 have the higher sucess rate for the same site
the medium payloads for the site KSC LC 39A have lower success rate

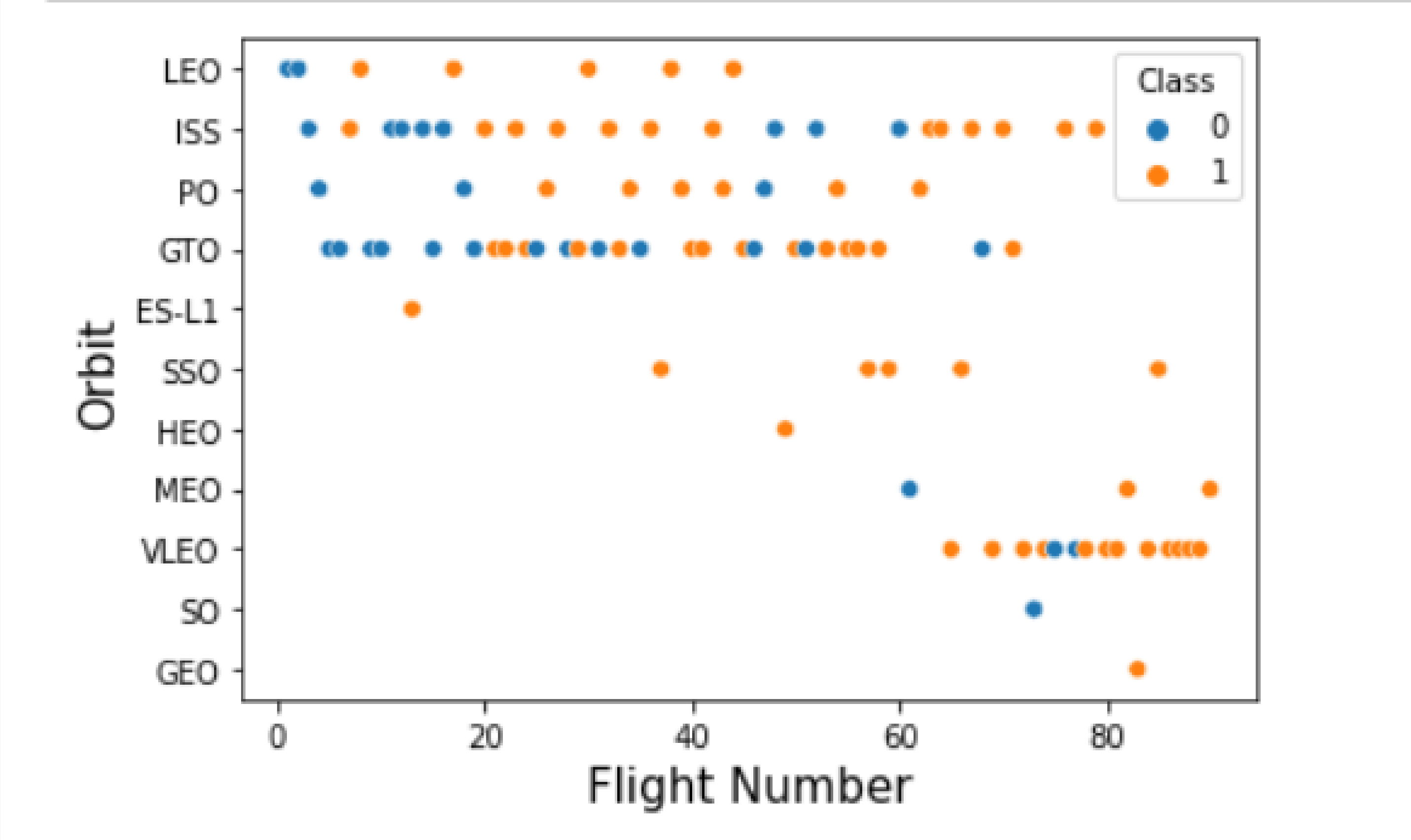
Success Rate vs. Orbit Type

The orbits GEO , ES-L1 , SSO and HEO have the heights successful rate



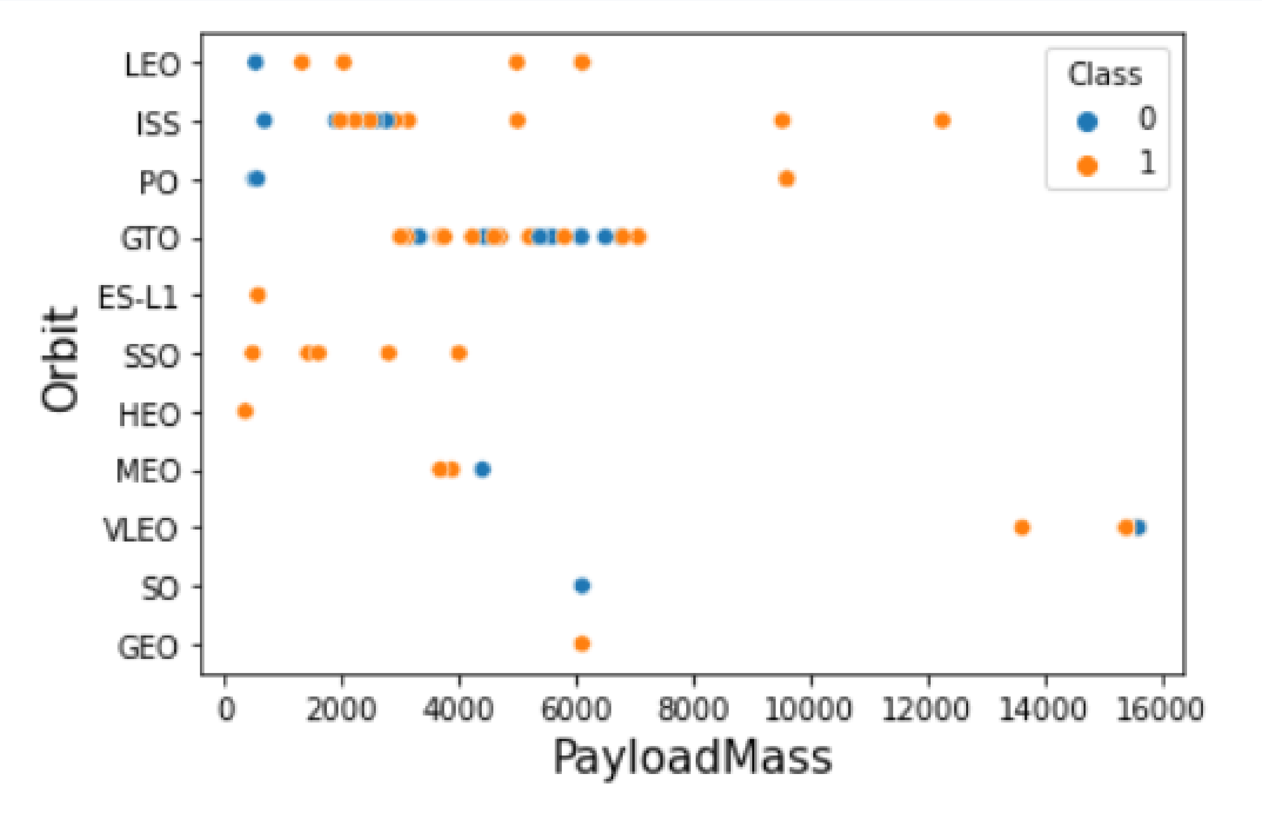
Flight Number vs. Orbit Type

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



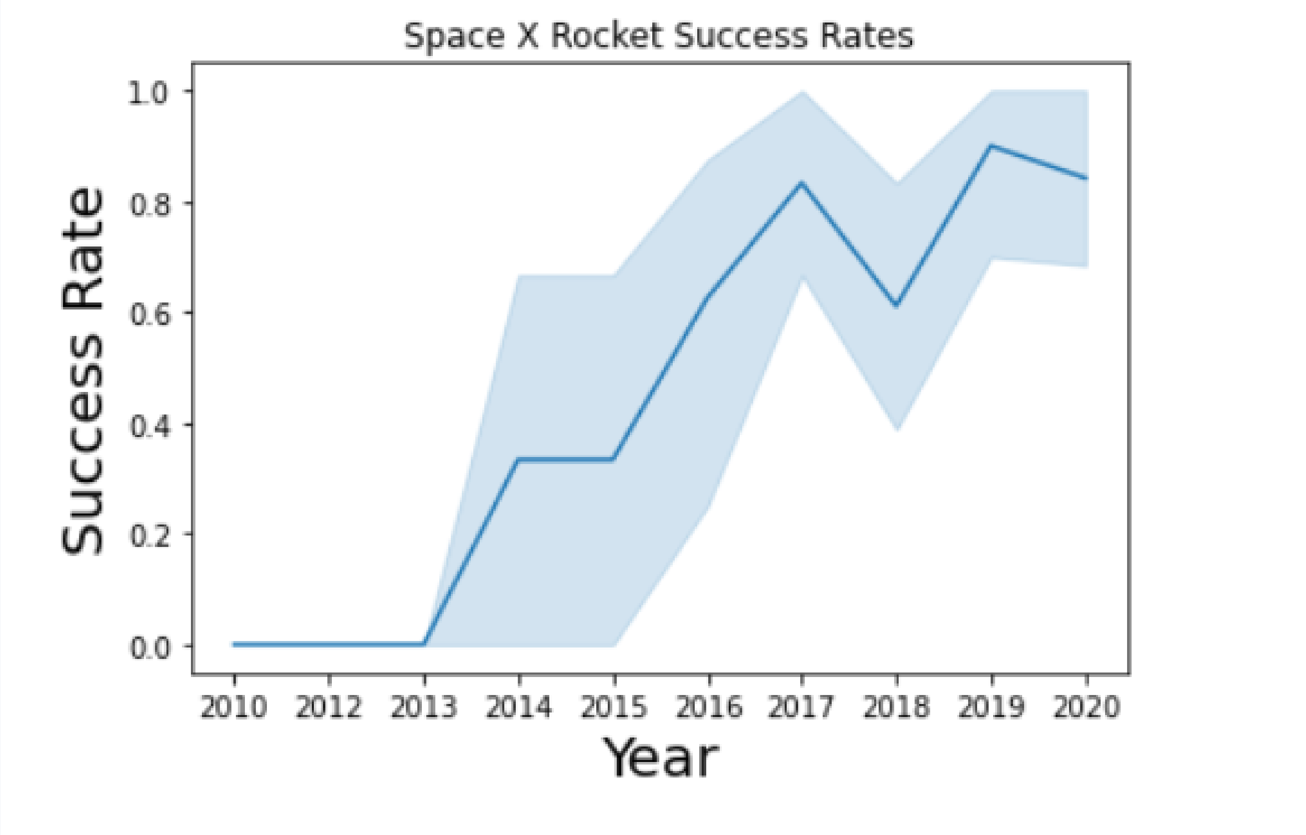
Payload vs. Orbit Type

We can observe that Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits



Launch Success Yearly Trend

we can observe that the success rate since 2013 kept increasing till 2020



All Launch Site Names

We use the DISTINCT command in order to unify the launch sites names

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

```
%%sql
select DISTINCT  Launch_Site from spacex
```

Launch Site Names Begin with 'CCA'

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

```
%%sql
select * from spacex where  Launch_Site like '%CCA%' limit 5
```

Using the limit command we can display the first desired number of records

Total Payload Mass

Total Payload mass

1
111268

```
%%sql
select sum(PAYLOAD_MASS__KG_)  from spacex where Payload like '%CRS%'
```

We use the function sum to summate the total in the columns Payload mass

Average Payload Mass by F9 v1.1

Avg function is used to calculate the average value of the column payload mass

average payload mass carried by booster version F9 v1.1

1
2928

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

First Successful Ground Landing Date

the date when the first successful landing outcome in ground pad was achieved

1
2010-06-04

```
%%sql
select min(Date) from spacex where Mission_Outcome = 'Success'
```

Using min function on the date column to with the condition “success “ to display the first successful landing

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
By selecting only booster_version column and apply the conditions :
Mission_outcome =“success” and
Payload_mas_KG between 4000and 6000

booster_version
F9 B4 B1040.2
F9 B4 B1040.1
F9 B5 B1046.2
F9 B5 B1046.3
F9 B5 B1047.2
F9 B5 B1048.3
F9 B5 B1051.2

```
%%sql
Select distinct Booster_Version from spacex where Mission_Outcome = 'Success' and PAYLOAD_MASS__KG_ between 4000 and 6000
```

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes
By counting the Mission outcome and unique the results

status
1
99
1

```
%%sql
select  count(Mission_Outcome) as status  from spacex group by Mission_Outcome
```

Boosters Carried Maximum Payload

To display the results we used the sub query method on the Payload mass column and the booster version column and the max function on the payload column to display the biggest load

booster	maxload
F9 B4 B1039.2	15600
F9 B4 B1040.2	15600
F9 B4 B1041.2	15600
F9 B4 B1043.2	15600
F9 B4 B1039.1	15600
F9 B4 B1040.1	15600
F9 B4 B1041.1	15600
F9 B4 B1042.1	15600
F9 B4 B1043.1	15600
F9 B4 B1044	15600
F9 B4 B1045.1	15600
F9 B4 B1045.2	15600
F9 B5 B1046.1	15600
F9 B5 B1046.2	15600

```
%%sql
select distinct Booster_Version as booster , (select max(PAYLOAD_MASS__KG_) from spacex ) as maxload from spacex
```

2015 Launch Records

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2015-06-28	14:21:00	F9 v1.1 B1018	CCAFS LC-40	SpaceX CRS-7	1952	LEO (ISS)	NASA (CRS)	Failure (in flight)	Precluded (drone ship)

```
%%sql
select * from spacex where Mission_Outcome like 'Failure%'
```

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

```
%%sql
SELECT COUNT(landing__outcome) FROM spacex WHERE (landing__outcome LIKE '%Success%') AND (DATE >'04-06-2010') AND (DATE < '20-03-2017')
```

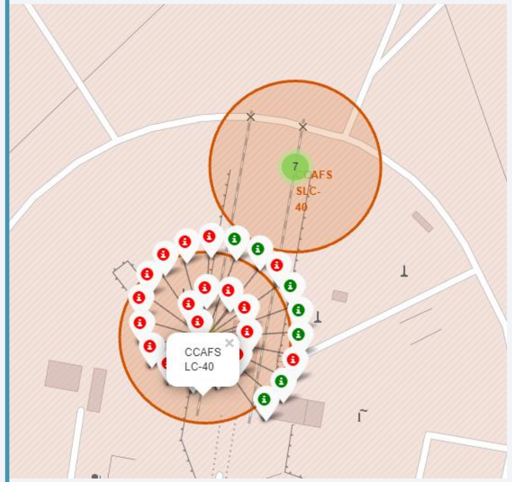
Section 4

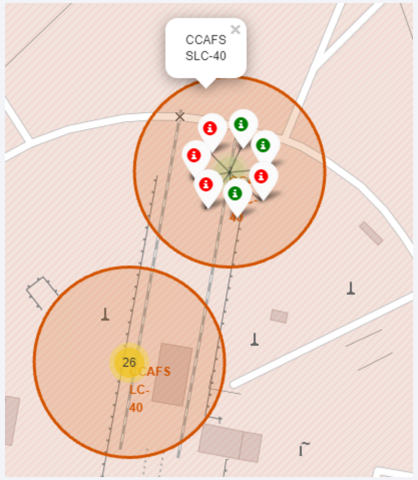
Launch Sites Proximities Analysis

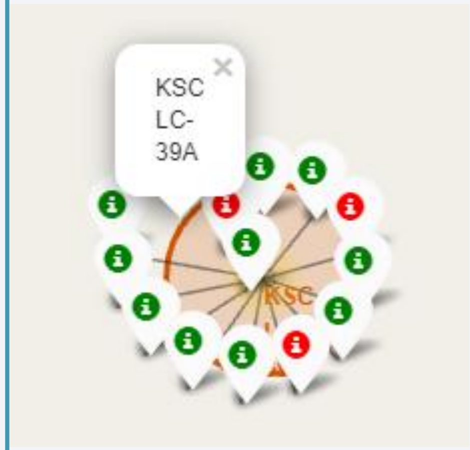
All launching sites map

As noticed , all the launches sites are in the west and east coast of the USA

Color labelled markers





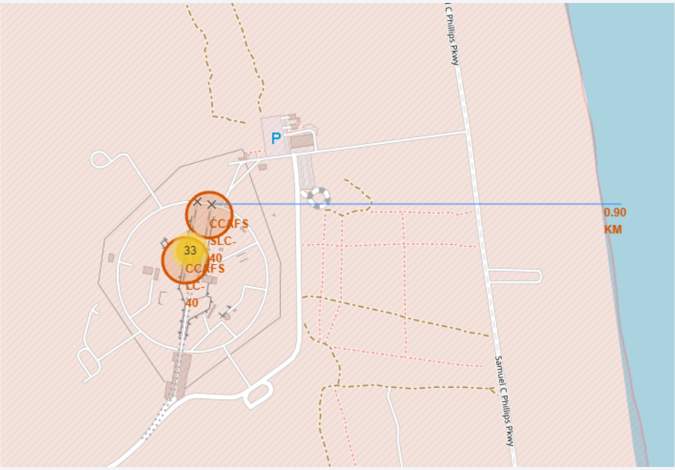


Florida launch sites

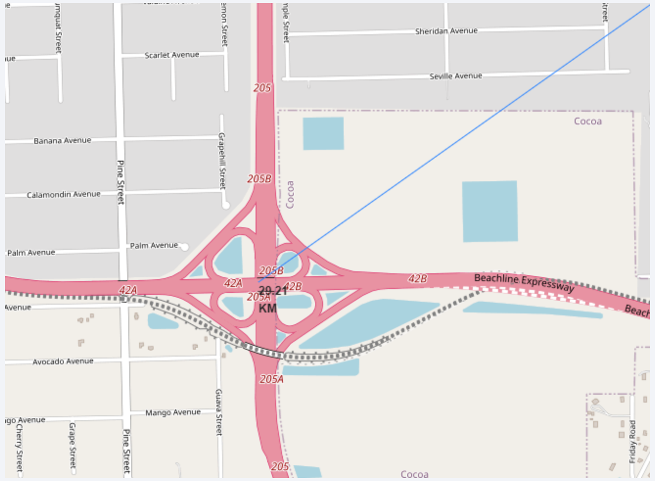
California launch sites

Green marker shows successful launches and Red marker shows failure ones

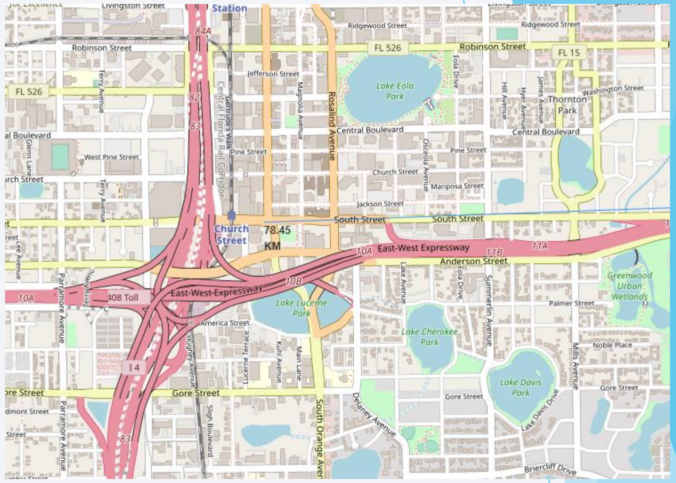
Working out Launches sites distances to landmarks



Distance to coastline



Distance to closest highway

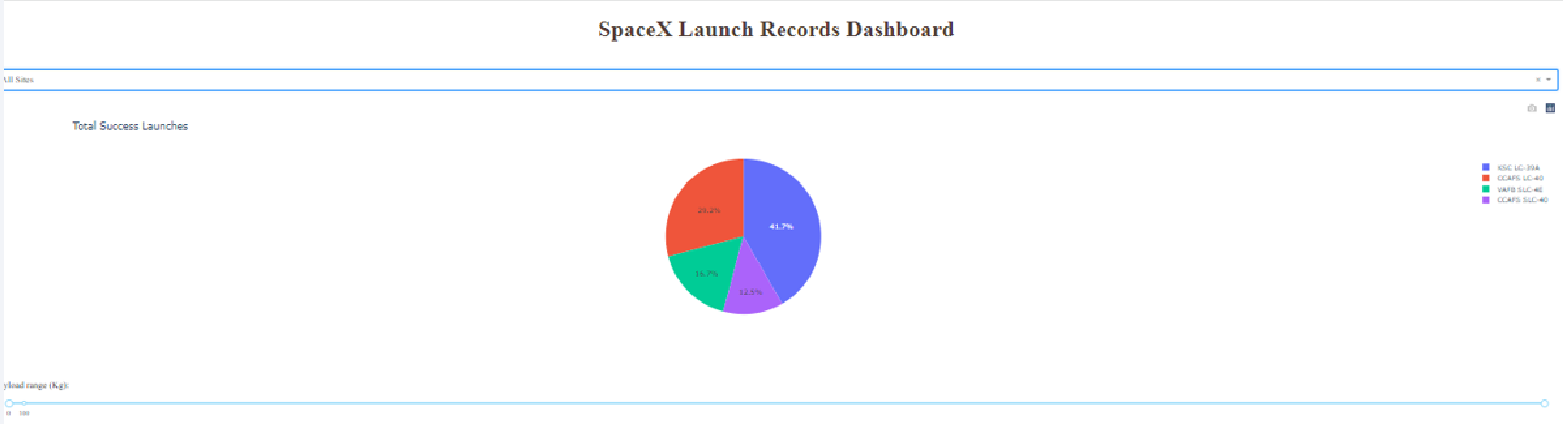


Distance to city

Section 5

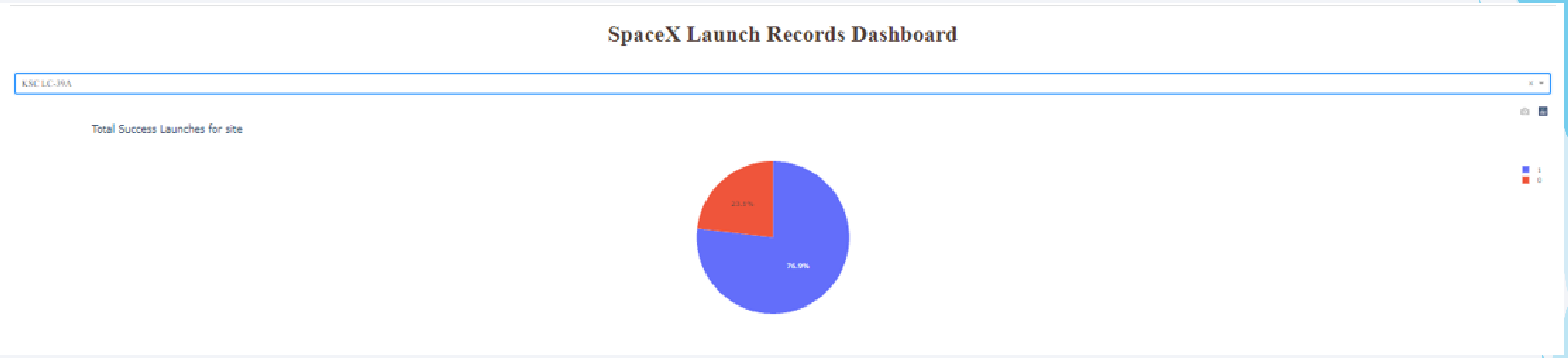
Build a Dashboard with Plotly Dash

PIE-Chart of launch success count for all sites



As shown in the Pie chart :
KSC-LC 39A site has the highest successful rate

Pie-chart for the launch site with highest launch success ratio

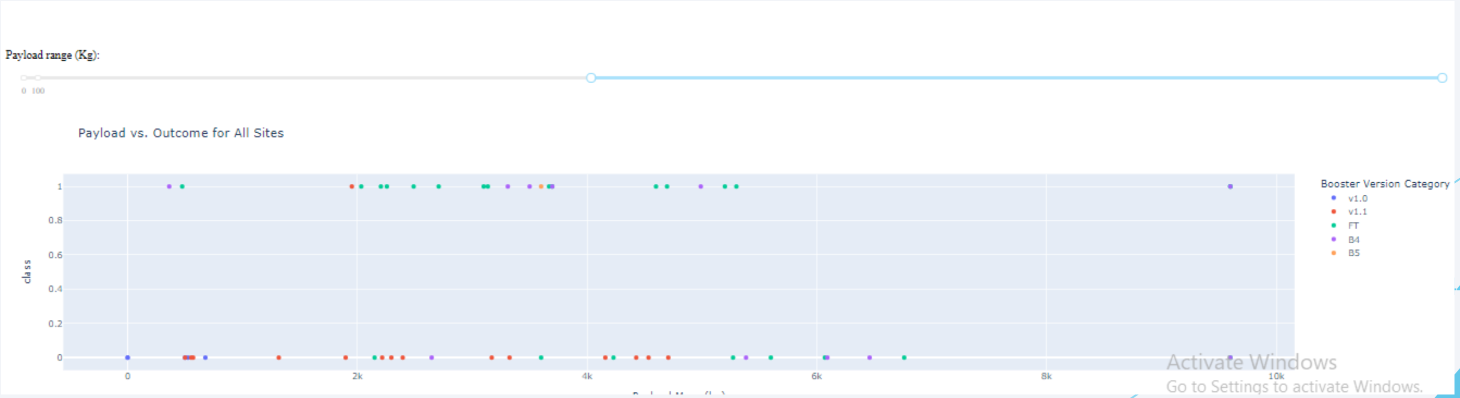


KSC-LC39A achieved 76.9 successful rate and 23,1 failure rate

Scatter plot Payload vs. Launch Outcome scatter plot for all sites

Payload mass 0-4000 kg

Payload mass 4000-10000 kg



Section 6

Predictive Analysis (Classification)

Classification Accuracy

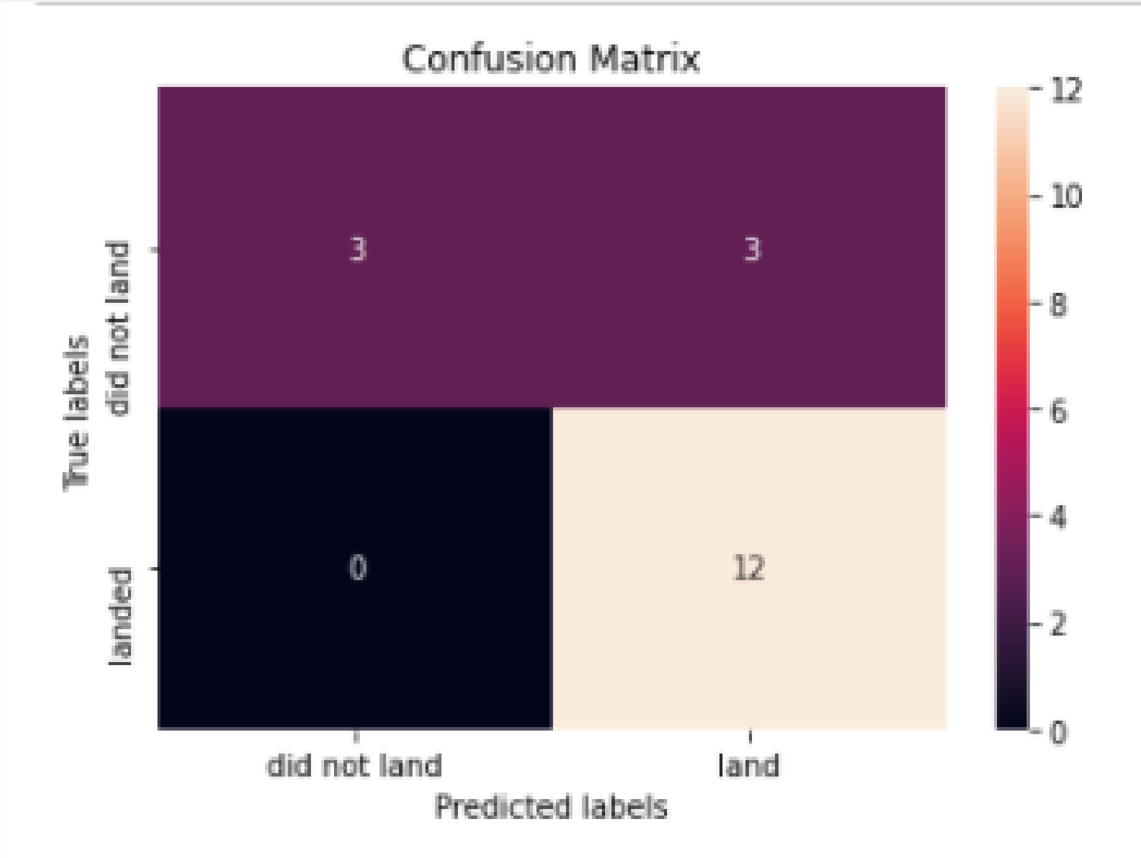
knn	0.840
Tree	0.880
LogesticRegression	0.833

```
methods = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestM = max(methods, key=methods.get)
print('Best Algorithm is',bestM,'with a score of',methods[bestM])
if bestM == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestM == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestM == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.8785714285714284
Best Params is : {'criterion': 'gini', 'max_depth': 14, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 5, 'splitte
r': 'best'}
```

The best algorithm is the Tree with the highest accuracy

Confusion Matrix of Tree Algorithm



Conclusions

- Tree Algorithm is the best ML algorithm for our data set
- Orbits GEO ,HEO , SSO ,ES-L1 have the best successful rate
- KSC LC-39A site has the most successful launching rate
- The successful rate is increasing eventually with the years , we predict to get a better results in the future .
- …

Appendix

Thank you!