IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

TUSHAR AGGARWAL
FEB 2023

Outline

Executive Summary

- Summary of methodologies
- Data Collection
- Data Wrangling
- EDA with data visualization
- EDA with SQL
- Building an Interactive map with Folium
- Building a Plotly Dash Dashboard
- Predictive Analysis (Classification)

- Summary of all results
- EDA result
- Interactive Analysis
- Predictive Analysis

Introduction

- Project background and context

➢ SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.

- Problems you want to find answers

➢ In this project task is to predicting if the first stage of the SpaceX Falcon 9 Rocket will land successfully

Section 1

Methodology

Methodology

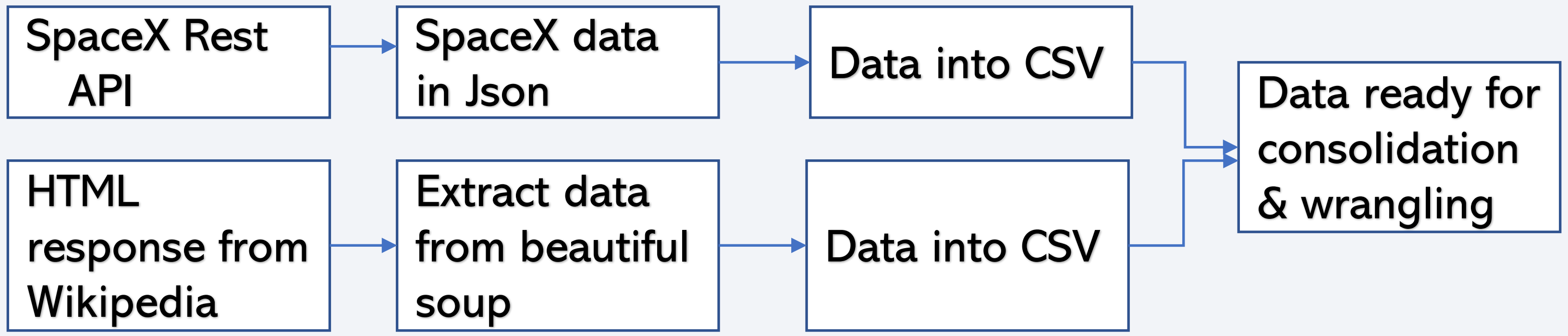
Executive Summary

- Data collection methodology:
- We will perform a get request using the requests library to obtain the launch data, which we will use to get the data from the API. This result can be viewed by calling the .json() method.
- Perform data wrangling
- The data is reviewed based on the attributes (example: Payload Mass, Orbit, Launch Site, etc)
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
- SVM, KNN, DT models have been built and evaluated for the best classifier

Data Collection

- Describe how data sets were collected.
- SpaceX launch data that is gathered from an API, specifically the SpaceX REST API.
- This API will give us data about launches, including information about the rocket used,
- payload delivered, launch specifications, landing specifications, and landing outcome.
- The SpaceX REST API endpoints, or URL, starts with api.spacexdata.com/v4/.

- You need to present your data collection process use key phrases and flowcharts


SpaceX Rest API
SpaceX data in Json
Data into CSV
HTML response from Wikipedia
Extract data from beautiful soup
Data into CSV
Data ready for consolidation & wrangling

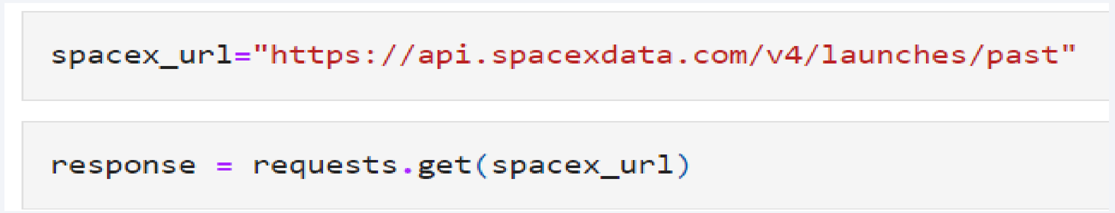
Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose.

- https://github.com/tushar2704/IBM_DS_Capstone/blob/main/api_1.ipynb

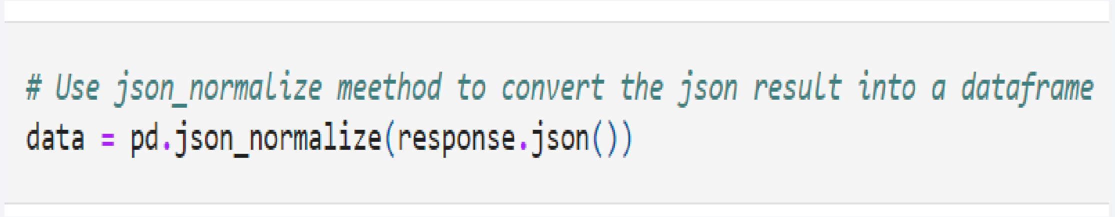
1. Getting Response from the SpaceX REST endpoints



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

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

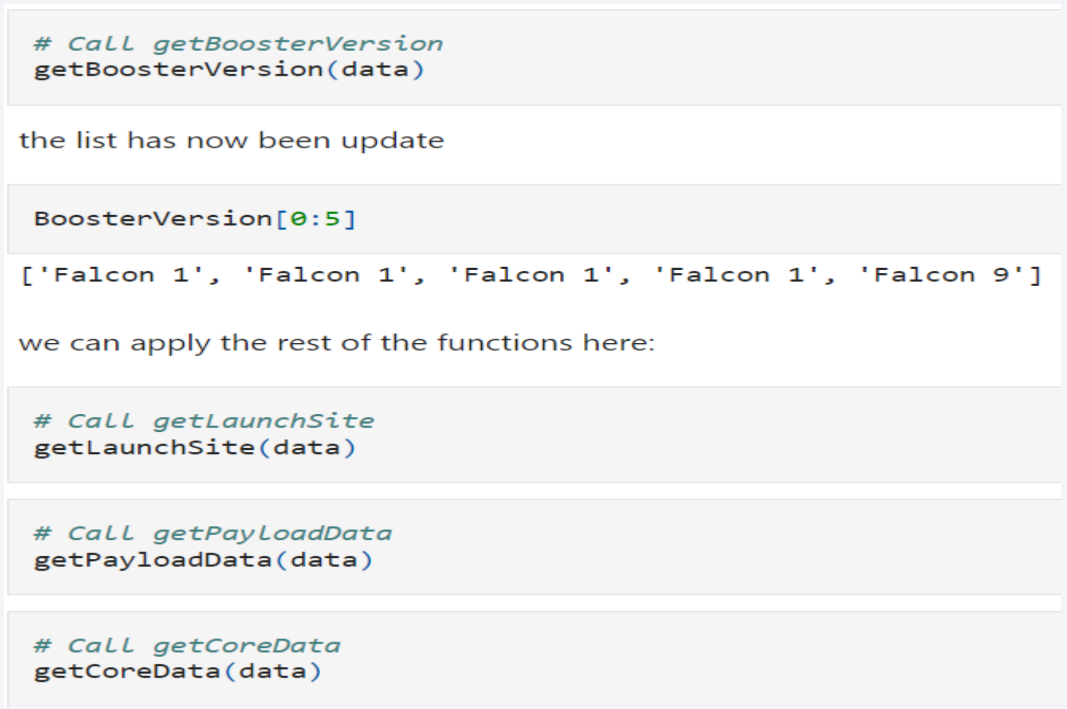
2. Convert 'response' to JSON



```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Data Collection (Continue) – SpaceX API

3. Apply Custom Functions



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

the list has now been update

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

```
['Falcon 1', 'Falcon 1', 'Falcon 1', 'Falcon 1', 'Falcon 9']
```

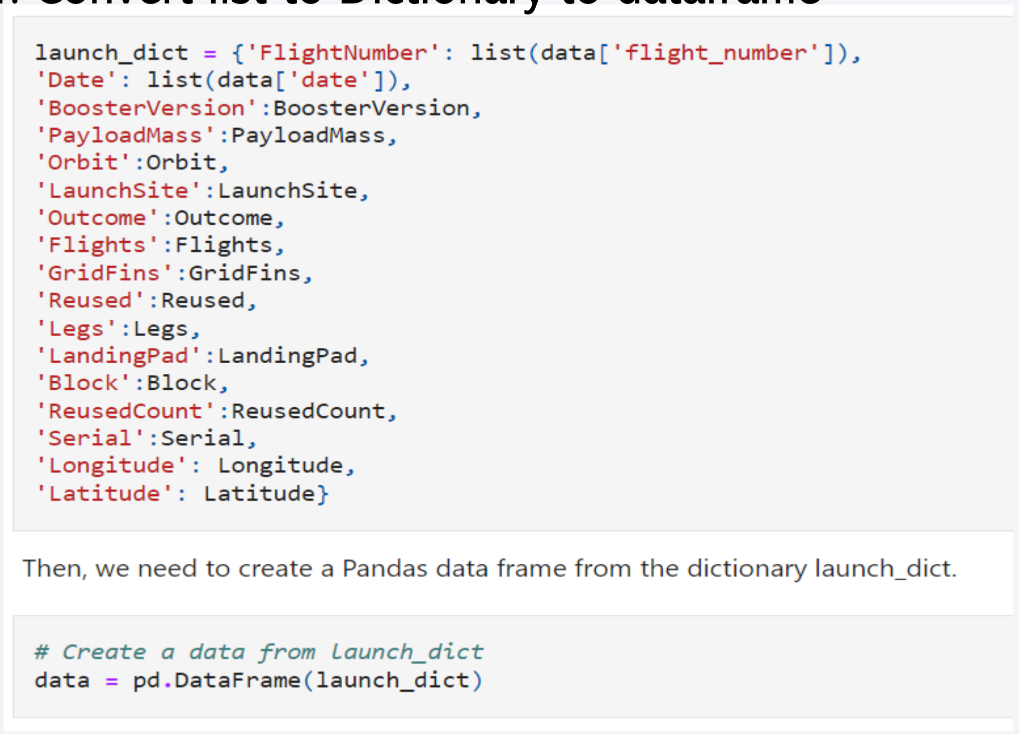
we can apply the rest of the functions here:

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

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

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

4. Convert list to Dictionary to 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}
```

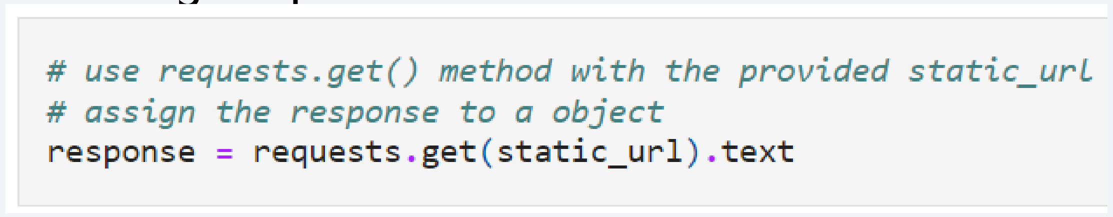
Then, we need to create a Pandas data frame from the dictionary launch_dict.

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

Data Collection - Scraping

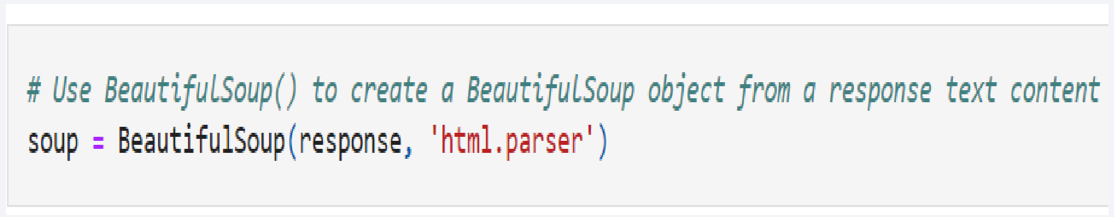
- Present your web scraping process using key phrases and flowcharts
- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose
- https://github.com/Komathi1977/testpro/blob/Master/jupyter-labs-webscraping.ipynb

1. Getting Response from HTML



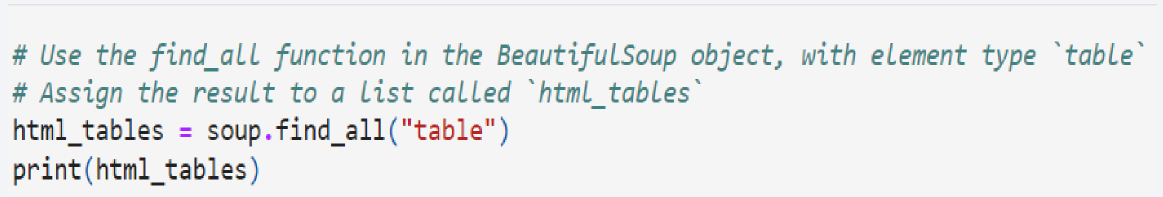
```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url).text
```

2. Creating BeautifulSoup Object



```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response, 'html.parser')
```

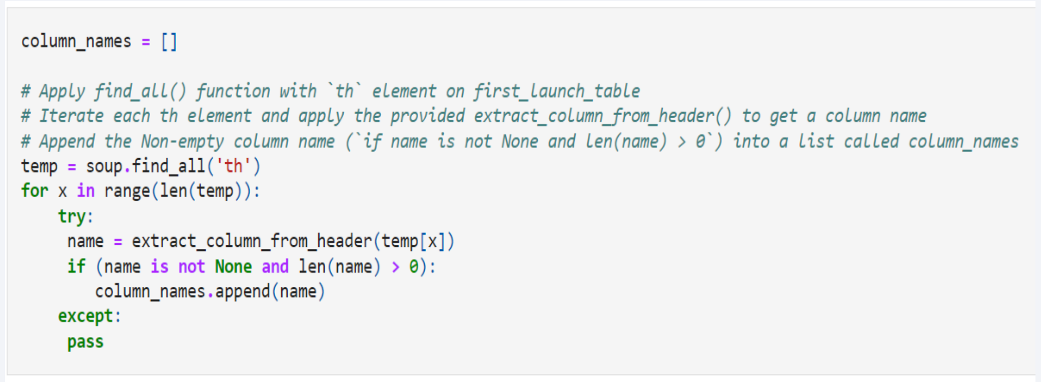
3. Finding tables



```
# 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)
```

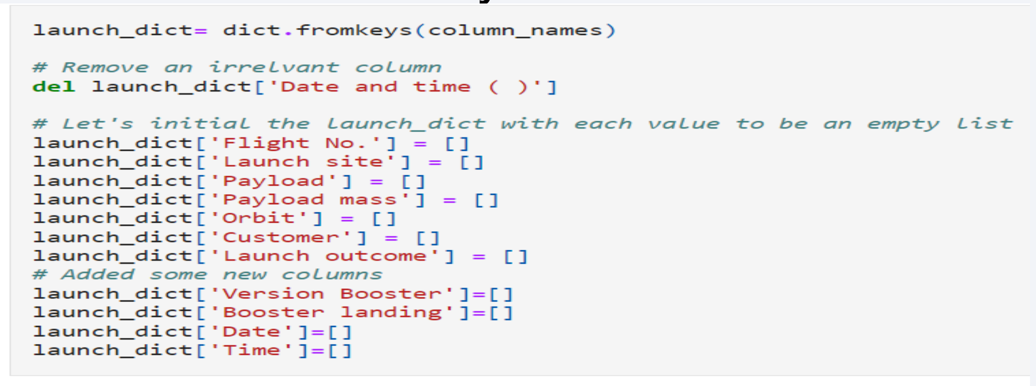
Data Collection (Continue) - Scraping

4. Getting Column names



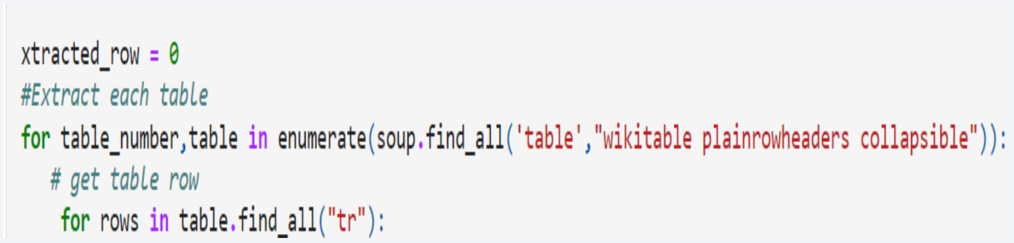
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called 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. Creation of dictionary



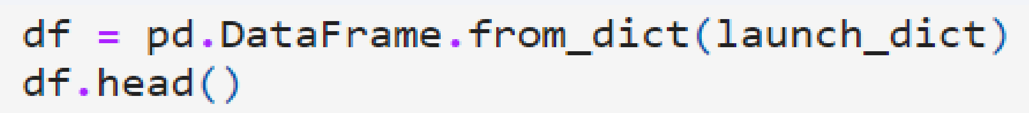
```
launch_dict= dict.fromkeys(column_names)

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

# Let's initial the launch_dict with each value to be an empty list
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



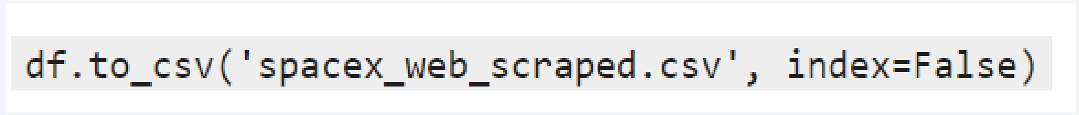
```
xtracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
```

7. Converting dictionary to dataframe



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

8. Dataframe to .CSV



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

Data Wrangling

- Describe how data were processed
- You need to present your data wrangling process using key phrases and flowcharts
- Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose
- https://github.com/tushar2704/IBM_DS_Capstone/blob/main/TA_webscraping.ipynb

Data Wrangling

1. Identify the columns of missing values and the data types from the dataframe

```
df.isnull().sum()/df.count()*100
```

```
df.dtypes
```

2. Identify the launch site

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

3. Identify the specific Orbit

```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

4. Identify the occurrence of mission outcome [er orbit type

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

5. Create "Outcome" column in the dataframe with 1 or 0 to indicate success or failed

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

landing_class = df['Outcome'].replace({'False Ocean': 0, 'False ASDS': 0, 'None None': 0, 'None ASDS': 0, 'False RTLS': 0, 'True ASDS': 1, 'True RTL
df['Outcome'] = df['Outcome'].astype(int)
df.info()
```

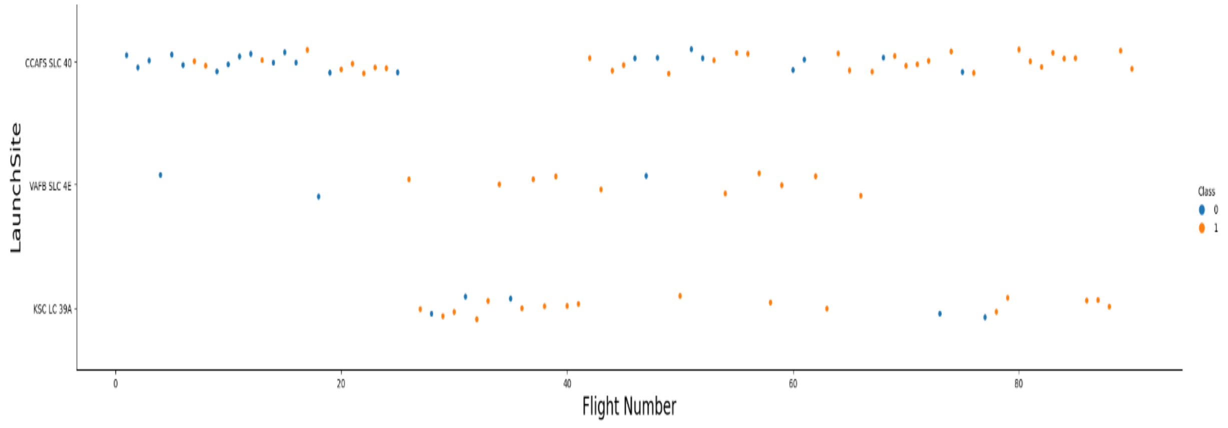
6. Dataframe to .CSV

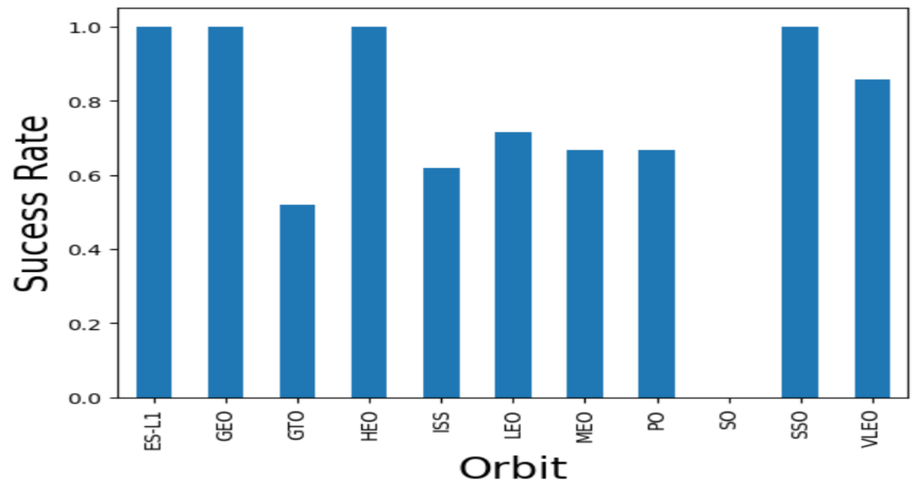
```
df.to_csv("dataset_part\_2.csv", index=False)
```

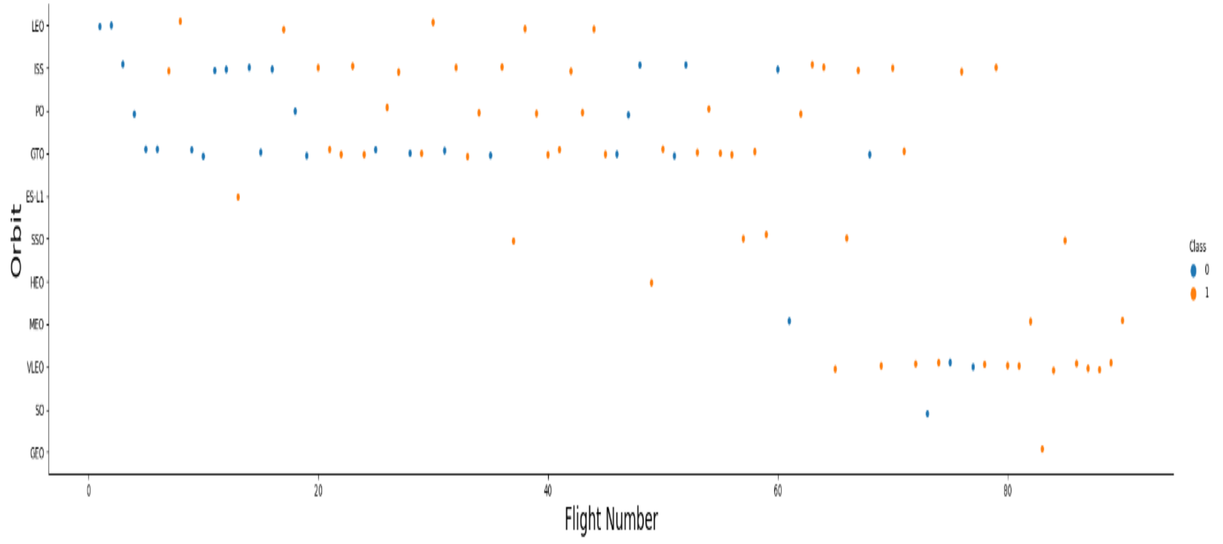
EDA with Data Visualization

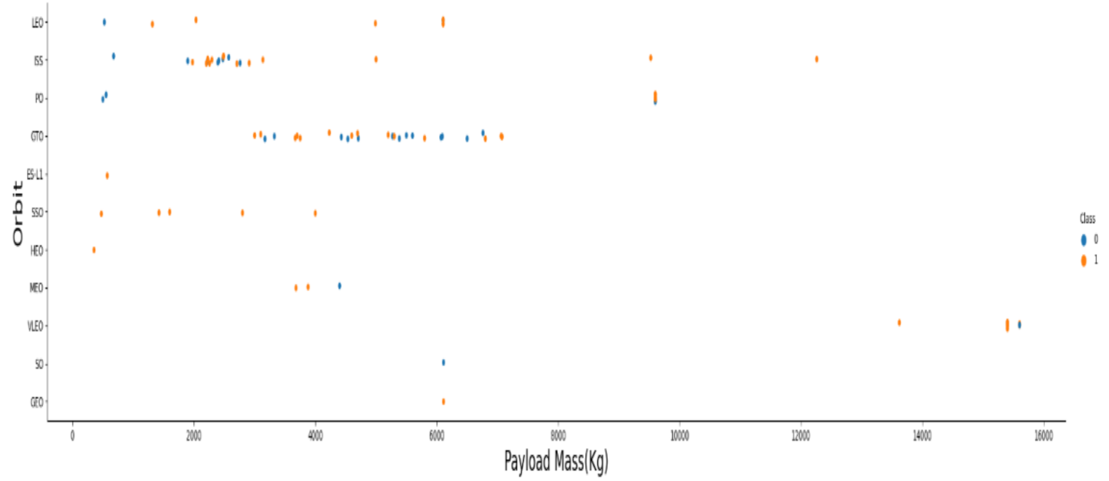
- Summarize what charts were plotted and why you used those charts
- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose
- https://github.com/tushar2704/IBM_DS_Capstone/blob/main/TA_eda.ipynb

EDA with Data Visualization

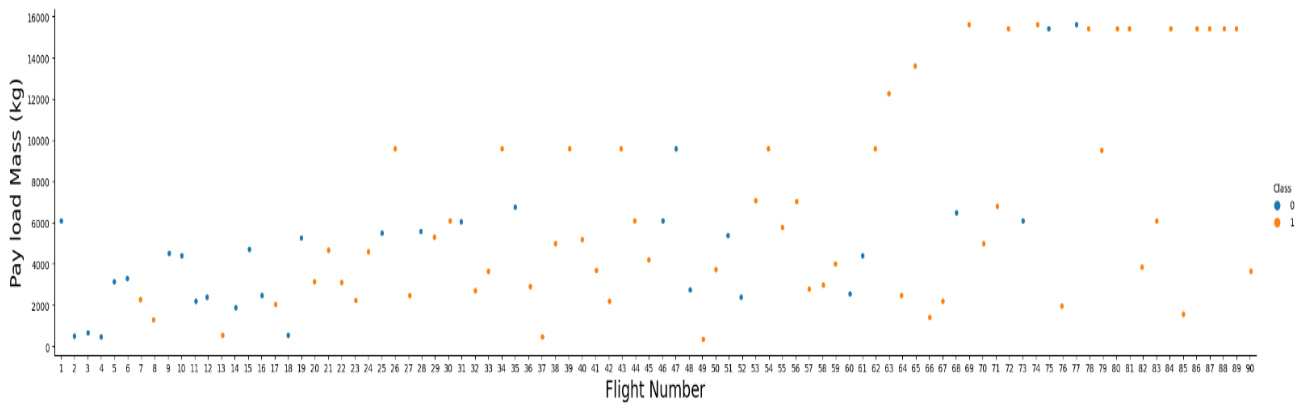




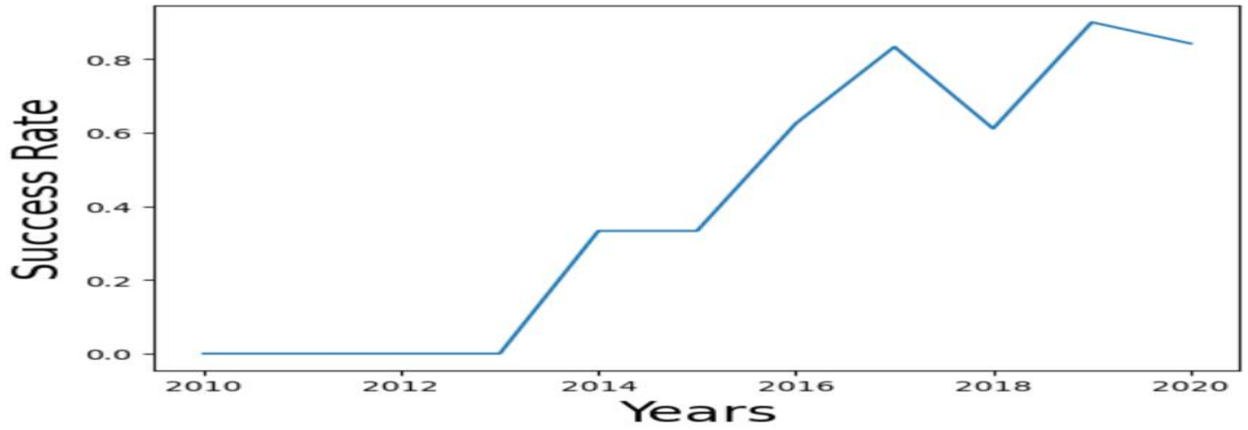




EDA with Data Visualization (continue)

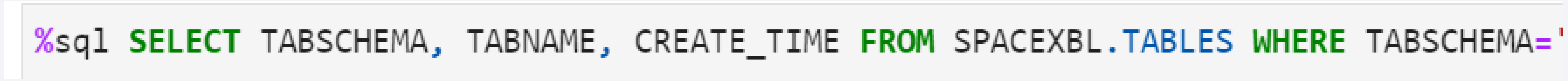


We see that different launch sites have different success rates. `CCAFS LC-40` has a success rate of 60 %, while `KSC LC-39A` and `VAFB SLC 4E` has a success rate of



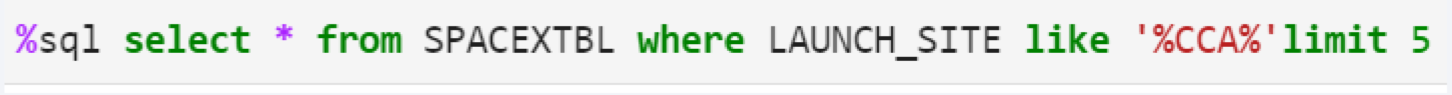
EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
- Display the names of the unique launch sites in the space mission



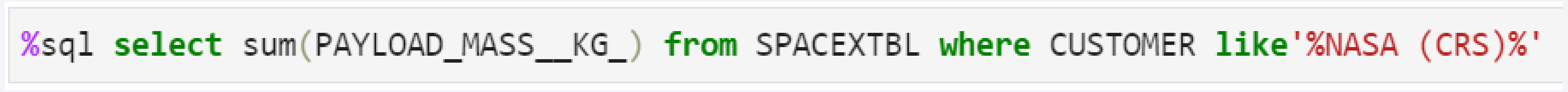
```
%sql SELECT TABSCHEMA, TABNAME, CREATE_TIME FROM SPACEXBL.TABLES WHERE TABSCHEMA='
```

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



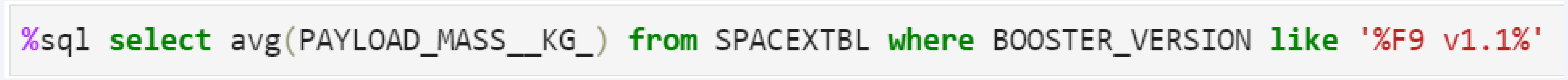
```
%sql select * from SPACEXTBL where LAUNCH_SITE like '%CCA%'limit 5
```

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



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

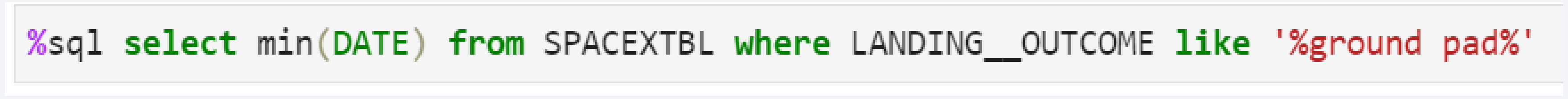
- Display average payload mass carried by booster version F9 v1.1



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

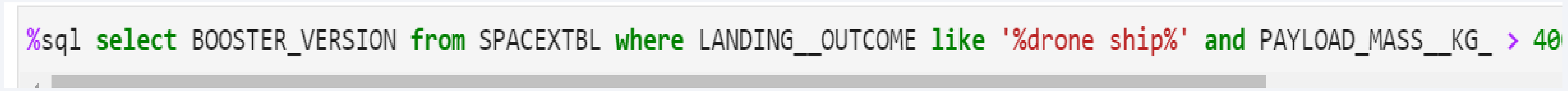
EDA with SQL (continue)

- Using bullet point format, summarize the SQL queries you performed
- List the date when the first successful landing outcome in ground pad was achieved.



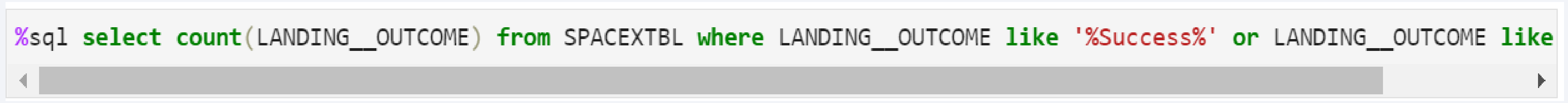
```
%sql select min(DATE) from SPACEXTBL where LANDING__OUTCOME like '%ground pad%'
```

- 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 like '%drone ship%' and PAYLOAD_MASS__KG_ > 40
```

- List the total number of successful and failure mission outcomes

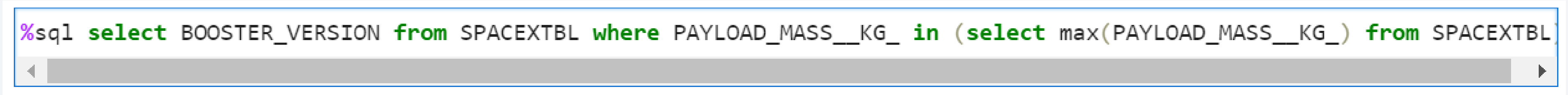


```
%sql select count(LANDING__OUTCOME) from SPACEXTBL where LANDING__OUTCOME like '%Success%' or LANDING__OUTCOME like
```

EDA with SQL (continue)

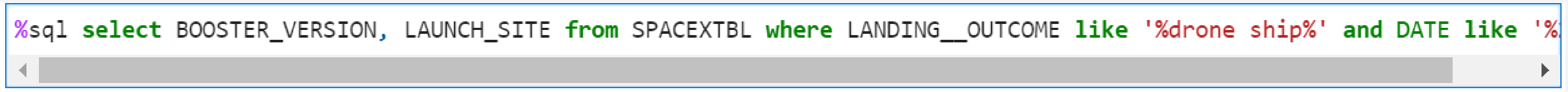
- Using bullet point format, summarize the SQL queries you performed

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



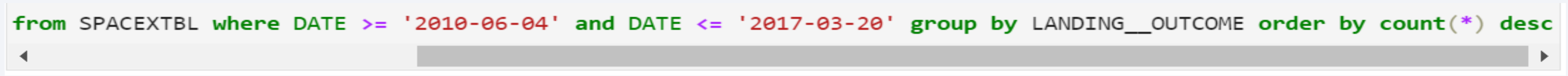
```
%sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ in (select max(PAYLOAD_MASS__KG_) from SPACEXTBL
```

- 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.



```
%sql select BOOSTER_VERSION, LAUNCH_SITE from SPACEXTBL where LANDING__OUTCOME like '%drone ship%' and DATE like '%
```

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



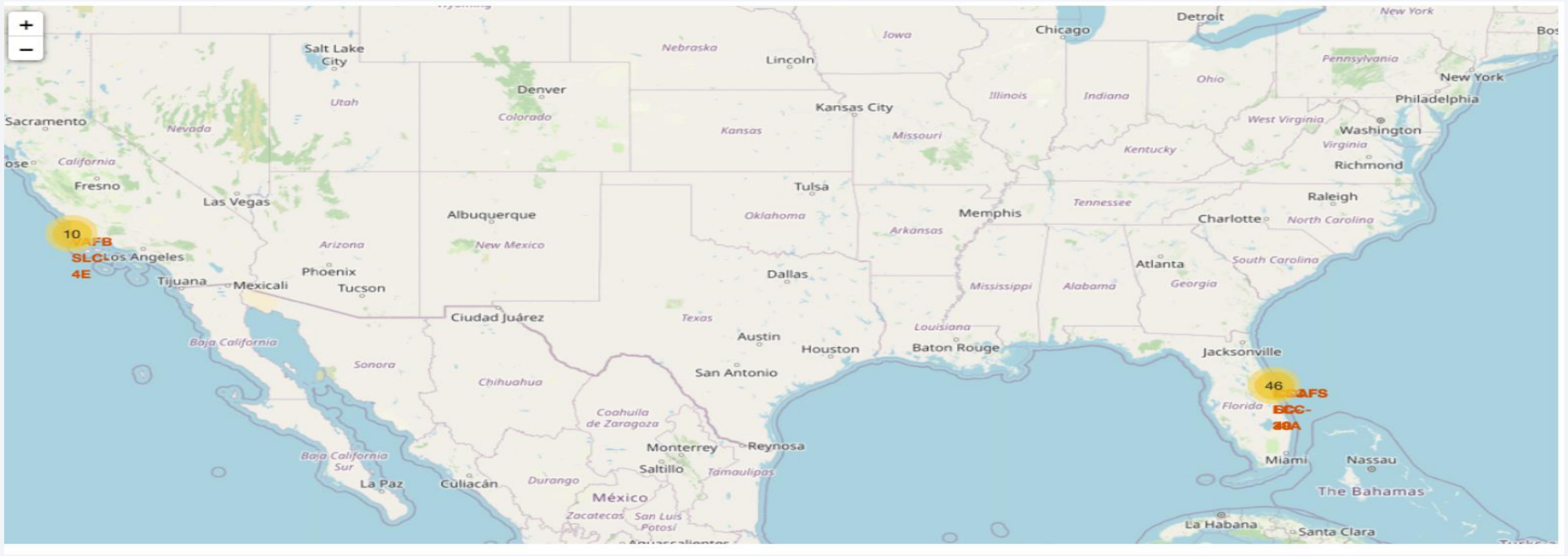
```
from SPACEXTBL where DATE >= '2010-06-04' and DATE <= '2017-03-20' group by LANDING__OUTCOME order by count(*) desc
```

- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

- https://github.com/tushar2704/IBM_DS_Capstone/blob/main/TA_SQL.ipynb

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map

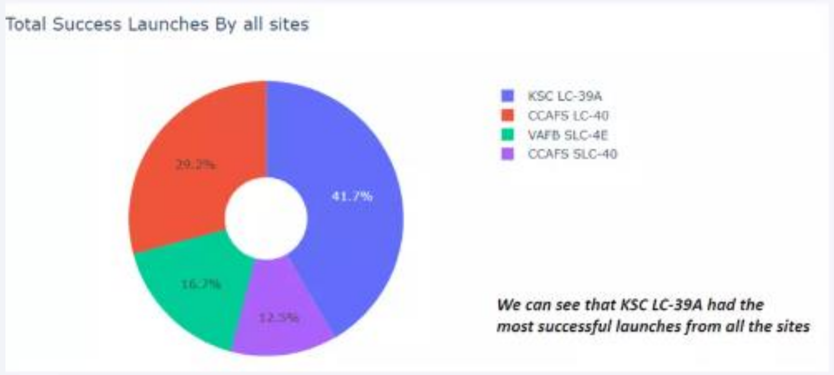


Build an Interactive Map with Folium (continue)

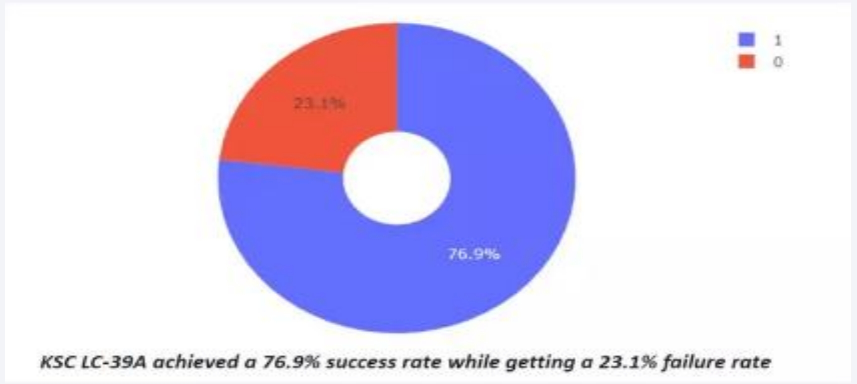
- Explain why you added those objects
- To enhance the map by adding the launch outcomes for each site, and see which sites have high success rates.
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose
- https://github.com/tushar2704/IBM_DS_Capstone/blob/main/TA_location.ipynb

Build a Dashboard with Plotly Dash

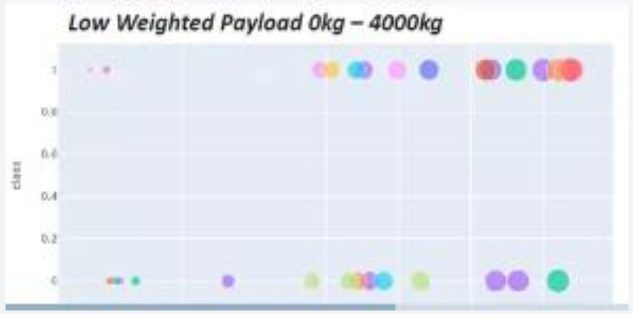
- Summarize what plots/graphs and interactions you have added to a dashboard

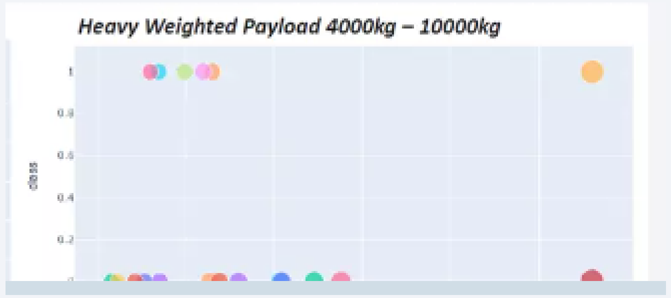


We can see that KSC LC-39A had the most successful launches from all the sites



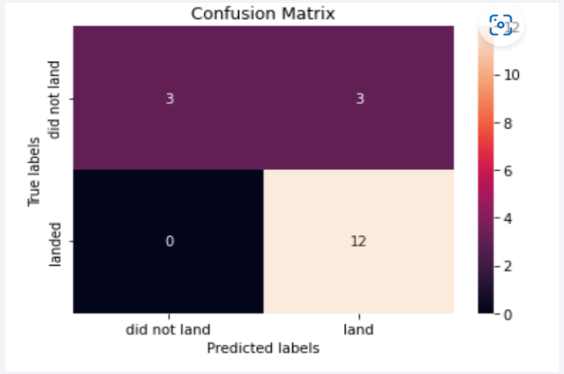
KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

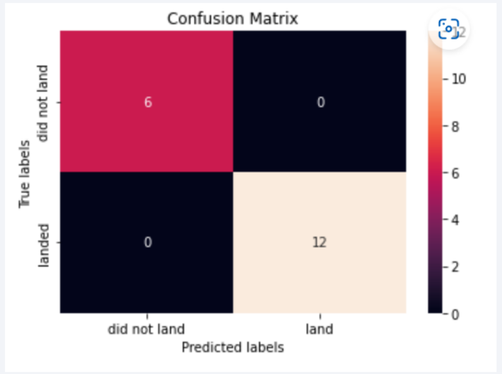


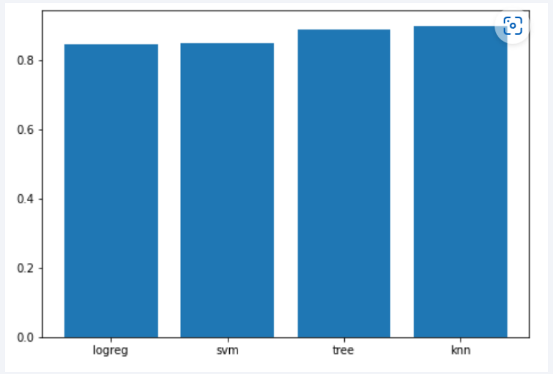


Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
- SVM, KNN and Logistic Regression Model used in this Analysis
- You need present your model development process using key phrases and flowchart

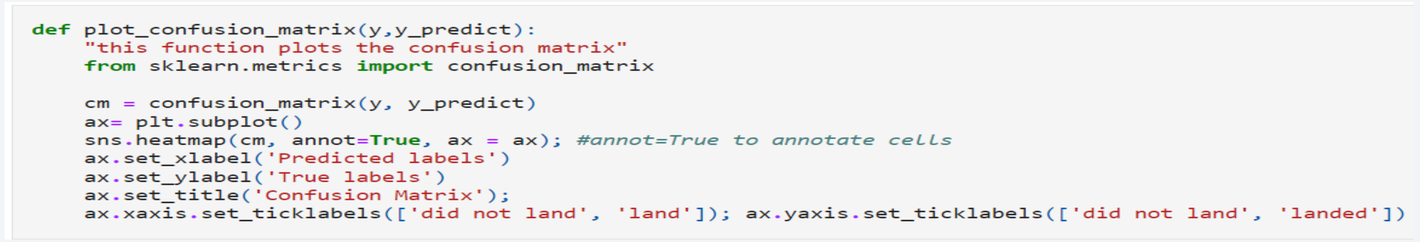






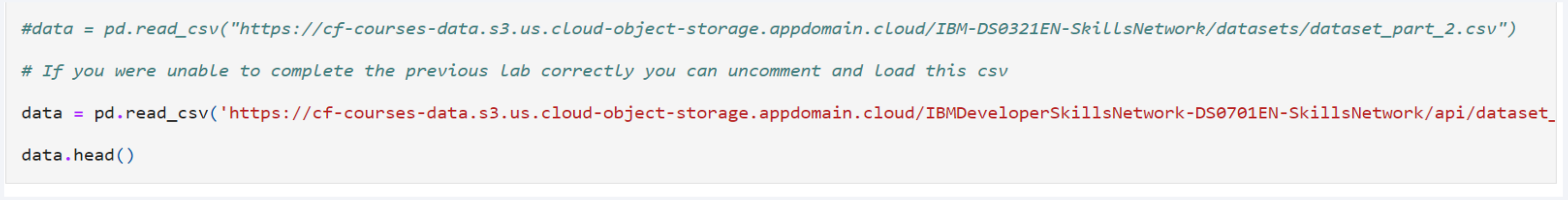
Predictive Analysis (Classification) – (continue)

1. Function plot is used to plot the confusion matrix



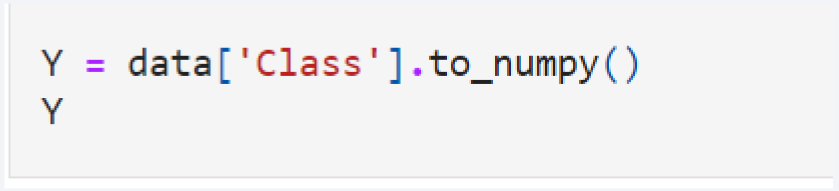
```python
def plot_confusion_matrix(y,y_predict):
    "this function plots the confusion matrix"
    from sklearn.metrics import confusion_matrix

    cm = confusion_matrix(y, y_predict)
    ax= plt.subplot()
    sns.heatmap(cm, annot=True, ax = ax); #annot=True to annotate cells
    ax.set_xlabel('Predicted labels')
    ax.set_ylabel('True labels')
    ax.set_title('Confusion Matrix');
    ax.xaxis.set_ticklabels(['did not land', 'land']); ax.yaxis.set_ticklabels(['did not land', 'landed'])
```

- 2. Dataframe loaded



```python
#data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_2.csv")

# If you were unable to complete the previous lab correctly you can uncomment and load this csv

data = pd.read_csv('https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DS0701EN-SkillsNetwork/api/dataset_

data.head()
```

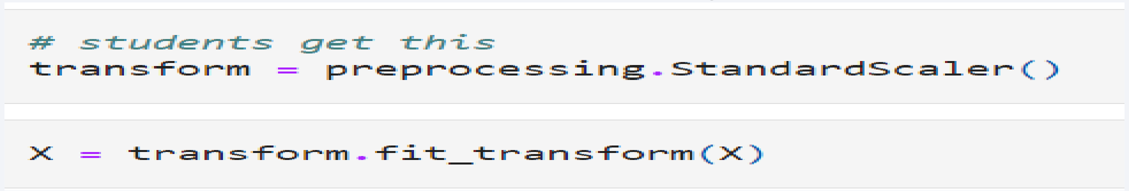
- 3. Create Numpy Array from the column Class in “data”



```python
Y = data['Class'].to_numpy()
Y
```

Predictive Analysis (Classification) – (continue)

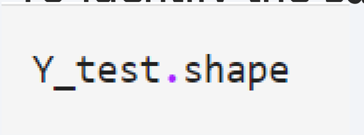
4. Standard the data in X and reassign it to the variable



```
# students get this
transform = preprocessing.StandardScaler()
```

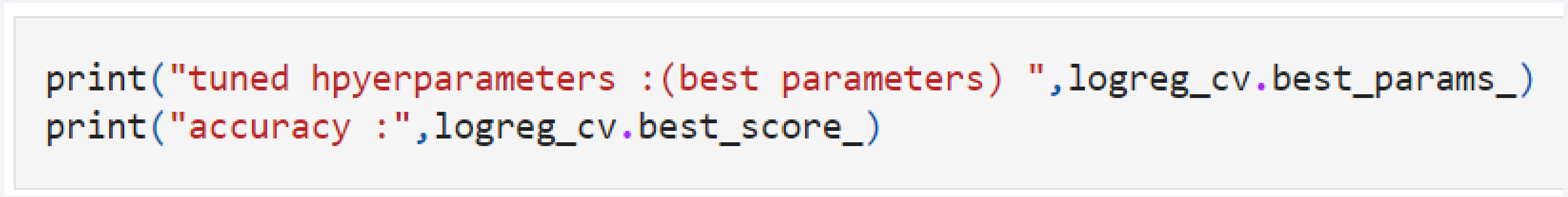
```
X = transform.fit_transform(X)
```

5. To identify the sample text



```
Y_test.shape
```

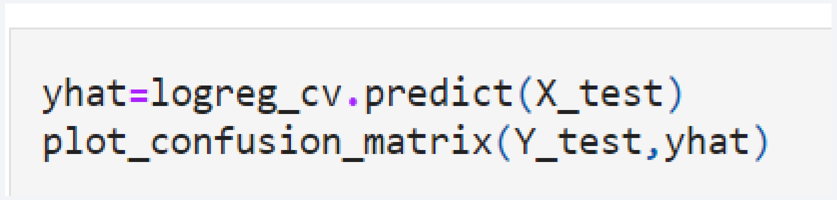
6. To display the parameter using data attributes



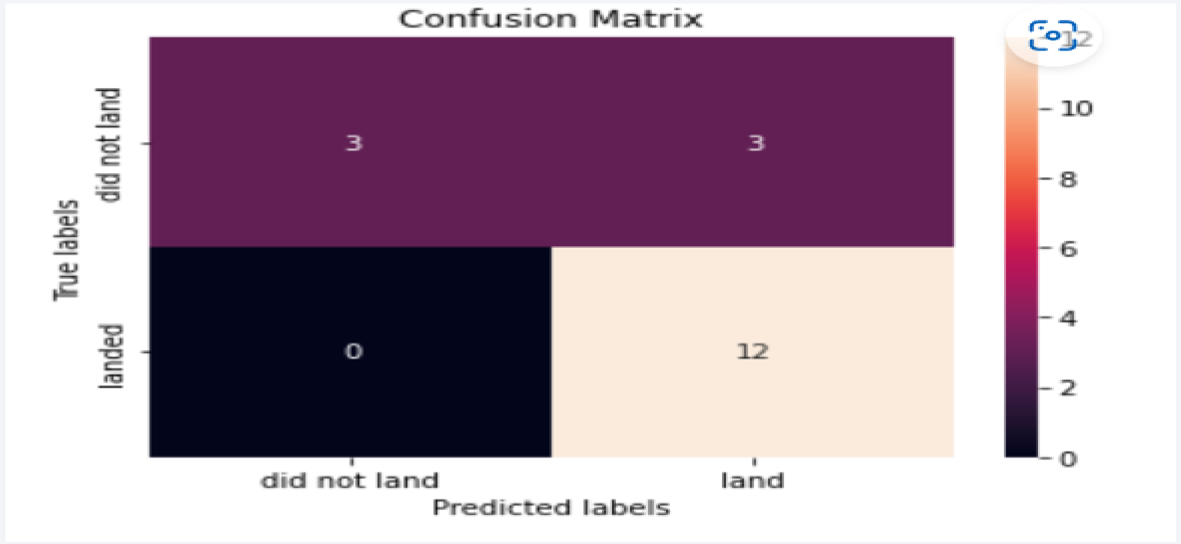
```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

Predictive Analysis (Classification) – (continue)

7. Calculate the accuracy on the confusion matrix using score method

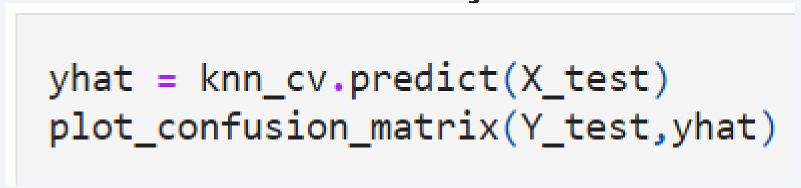


```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

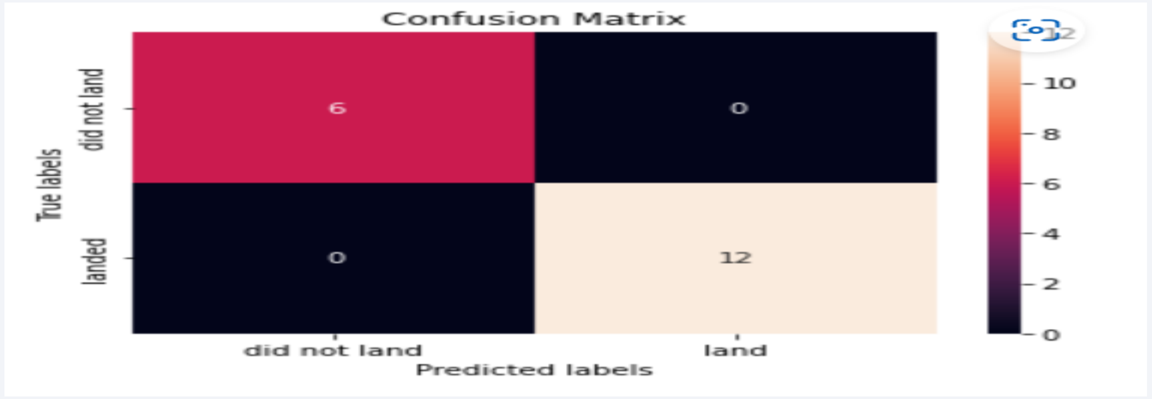


Predictive Analysis (Classification) – (continue)

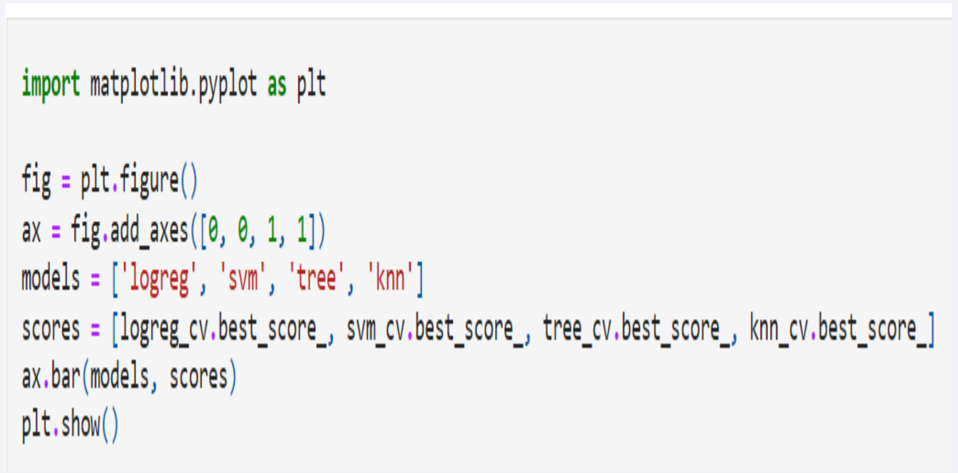
8. Calculate the accuracy of tree_cv using score method



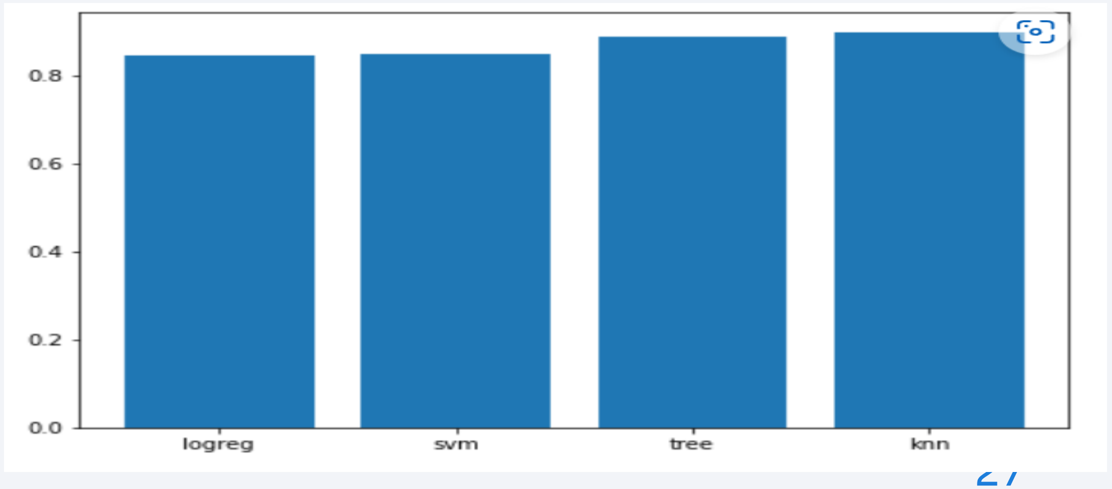
```
yhat = knn_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



- 9. To find best performance



```
import matplotlib.pyplot as plt

fig = plt.figure()
ax = fig.add_axes([0, 0, 1, 1])
models = ['logreg', 'svm', 'tree', 'knn']
scores = [logreg_cv.best_score_, svm_cv.best_score_, tree_cv.best_score_, knn_cv.best_score_]
ax.bar(models, scores)
plt.show()
```



Results

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

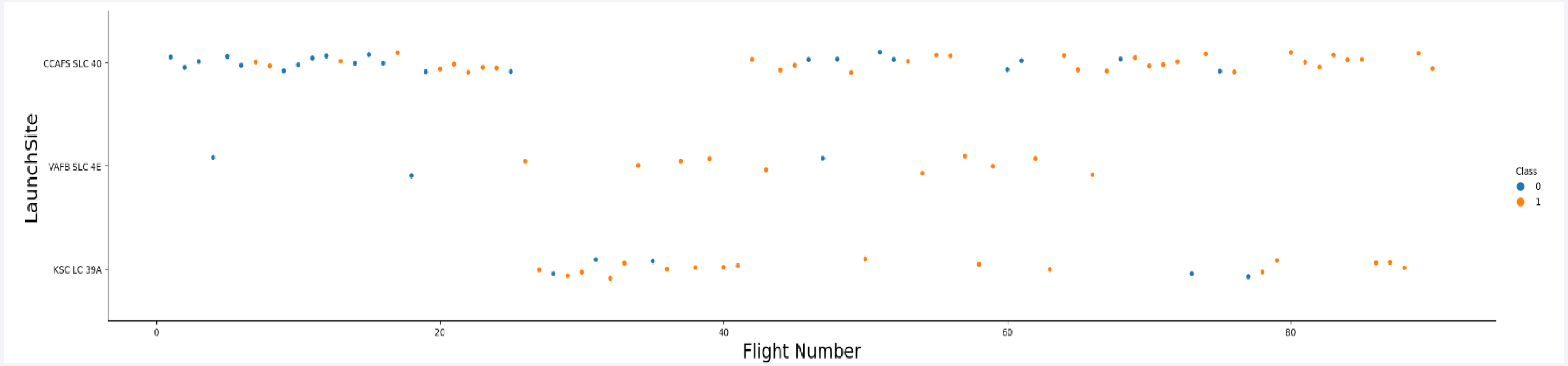
- SVM, KNN and Logistic Regression Model are the most accurate and best in prediction accuracy for this dataset

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

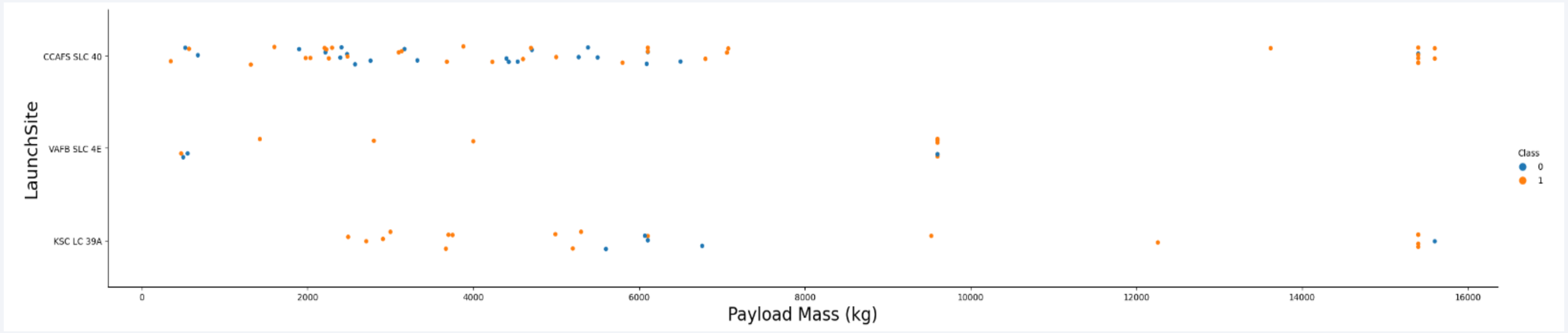
- Show a scatter plot of Flight Number vs. Launch Site



- CCAFS SLC 40 has higher launches than from other sites

Payload vs. Launch Site

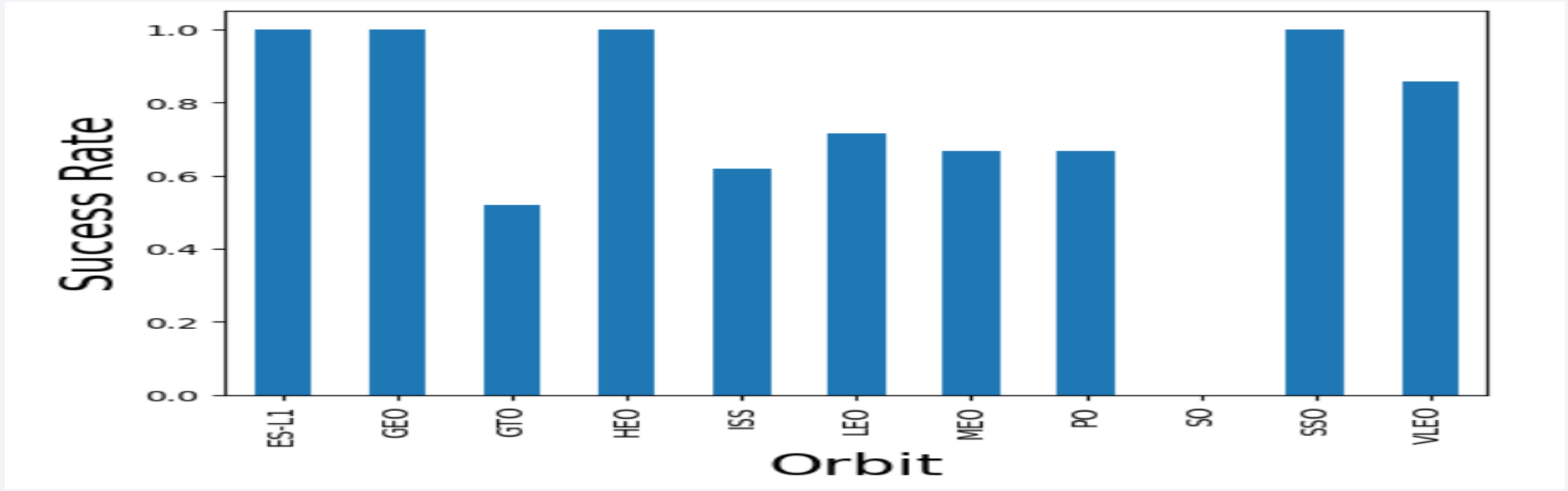
- Show a scatter plot of Payload vs. Launch Site



- From the scatter plot, three sites has payload mass <8000kg, whereas CCAFS SLC 40 has a wider range of payload mass.

Success Rate vs. Orbit Type

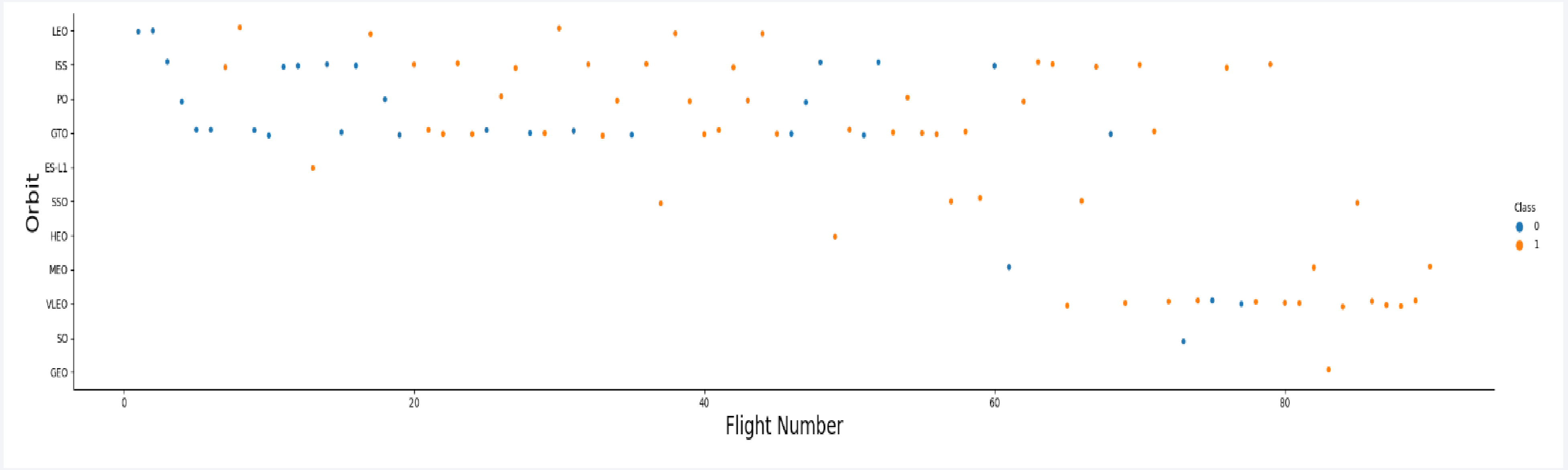
- Show a bar chart for the success rate of each orbit type



- The orbit type ES-11, GEO, HEO, SSO are the highest success rate.

Flight Number vs. Orbit Type

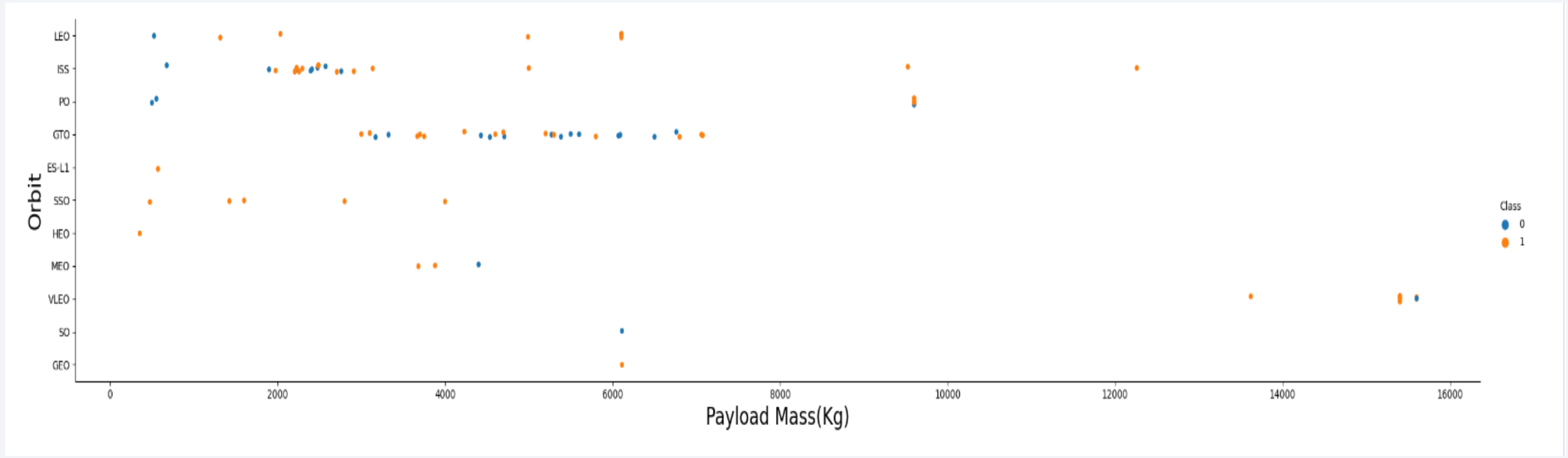
- Show a scatter point of Flight number vs. Orbit type



- From the scatter plot, at the beginning, LEO, ISS, PO and GTO dominates the launches whereas the rest has no launches or just a couple launches however it also noted that VLEO has launches recent years.

Payload vs. Orbit Type

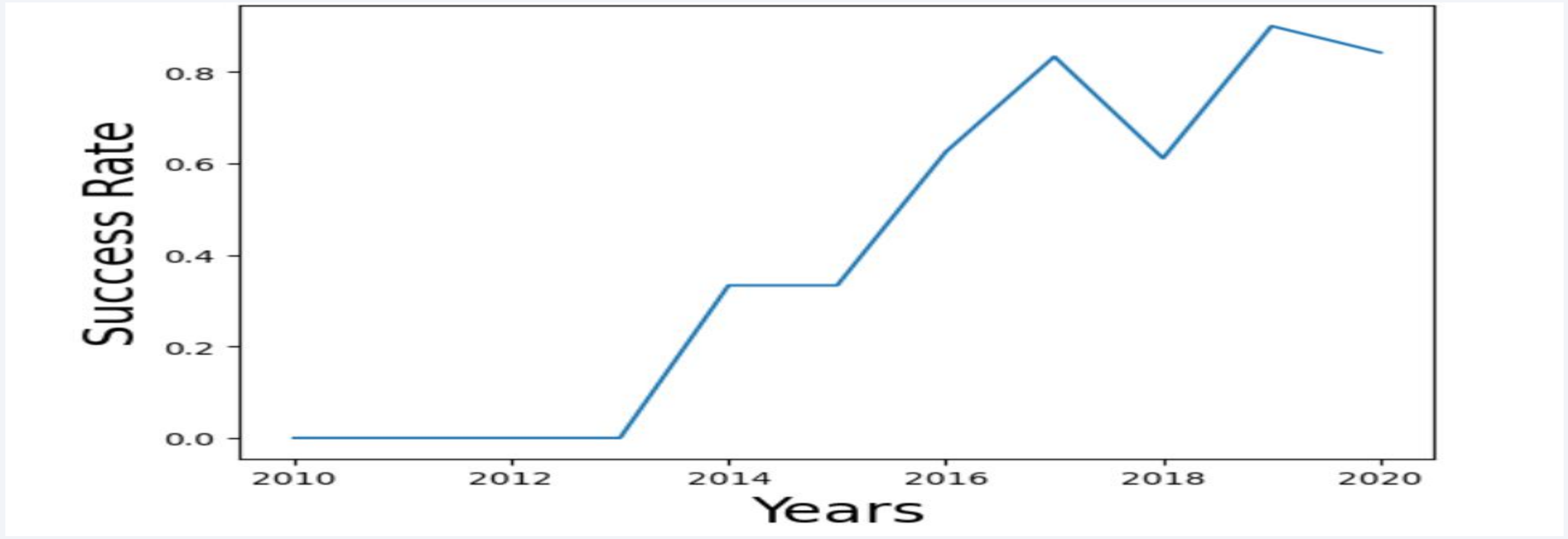
- Show a scatter point of payload vs. orbit type



- There are a strong correlation between ISS and Payload at the range around 2000, as well as between GTO and the range of 4000-8000

Launch Success Yearly Trend

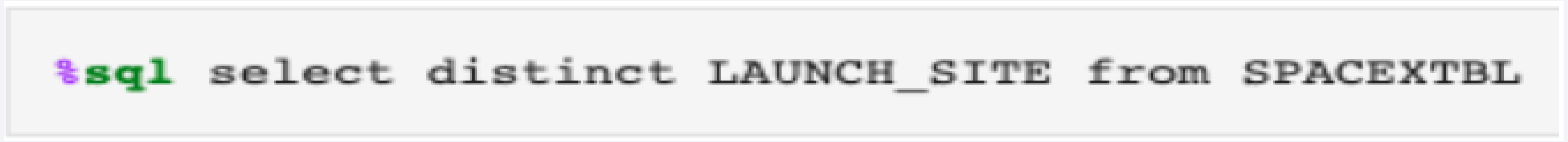
- Show a line chart of yearly average success rate



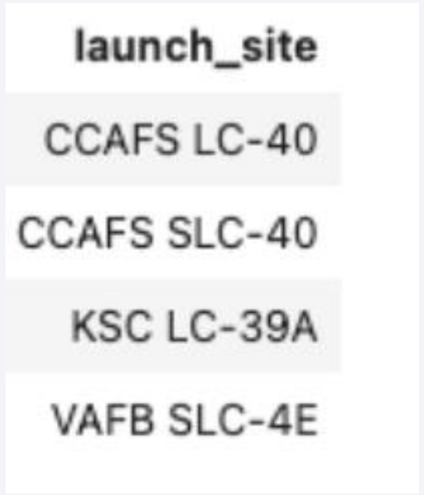
- The success rate has been increased since 2013 to 2017 and fall at 2018 but it's continue to increase in 2019

All Launch Site Names

- Find the names of the unique launch sites



```
%sql select distinct LAUNCH_SITE from SPACEXTBL
```



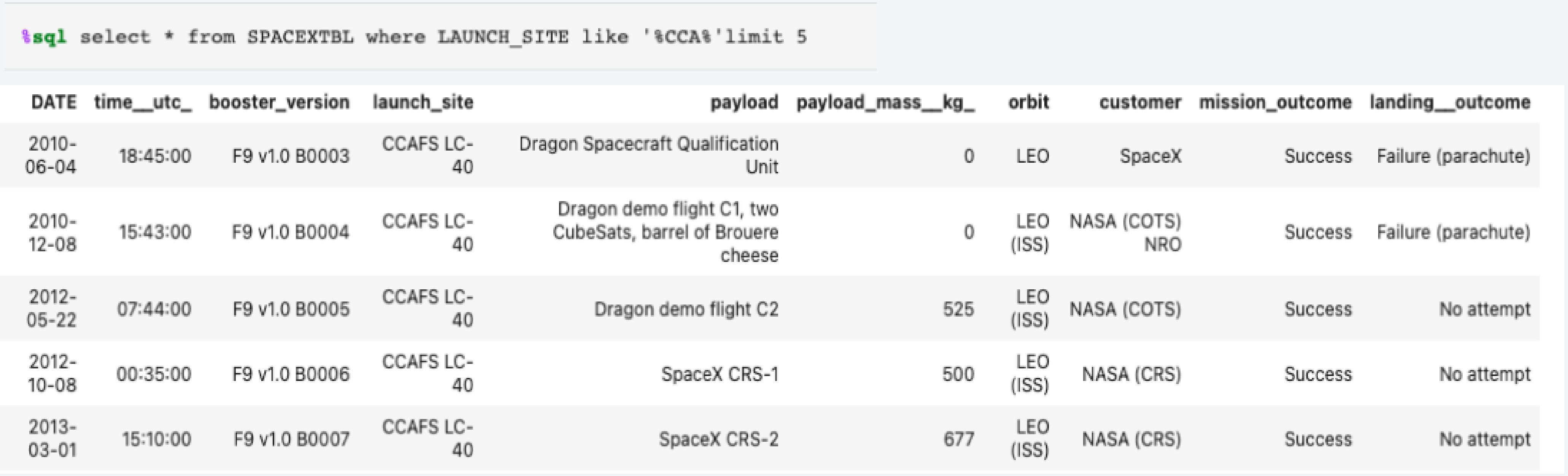
launch_site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

From the dataset there are 4 unique launch sites for all the launches.

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

- Present your query result with a short explanation here

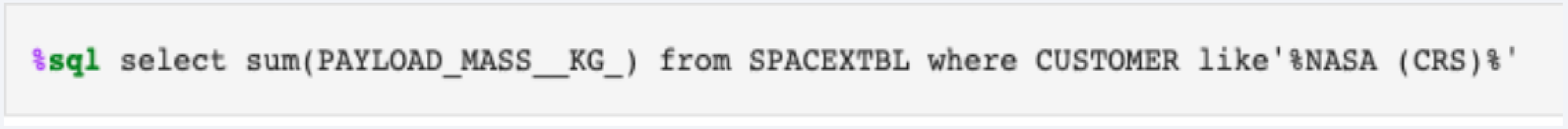


```
%sql select * from SPACEXTBL where LAUNCH_SITE like '%CCA%'limit 5
```

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

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- Present your query result with a short explanation here

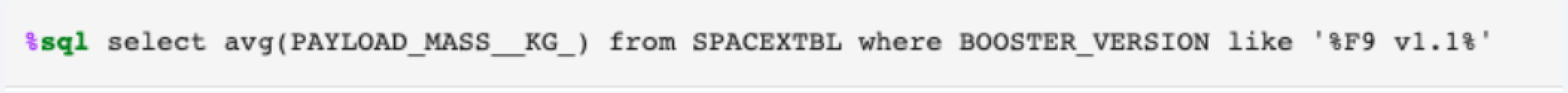


```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER like'%NASA (CRS)%'
```

```
48213
```

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here

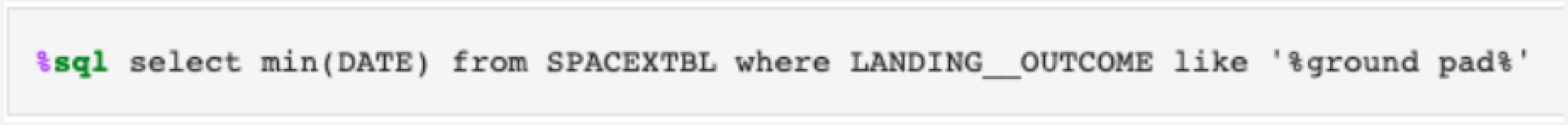


```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION like '%F9 v1.1%'
```

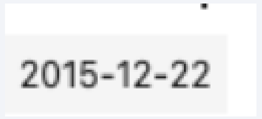
```
2534
```

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Present your query result with a short explanation here



```
%sql select min(DATE) from SPACEXTBL where LANDING__OUTCOME like '%ground pad%'
```

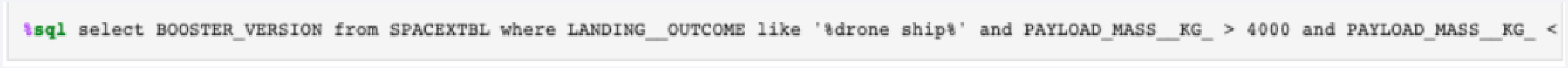


```
2015-12-22
```

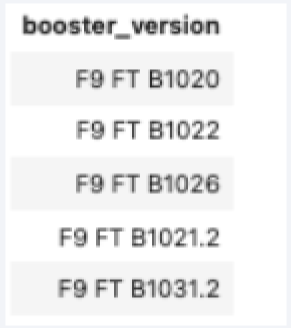
Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

- Present your query result with a short explanation here



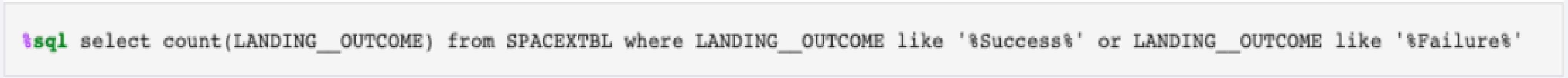
```
%sql select BOOSTER_VERSION from SPACEXTBL where LANDING__OUTCOME like '%drone ship%' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ <
```



booster_version
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here

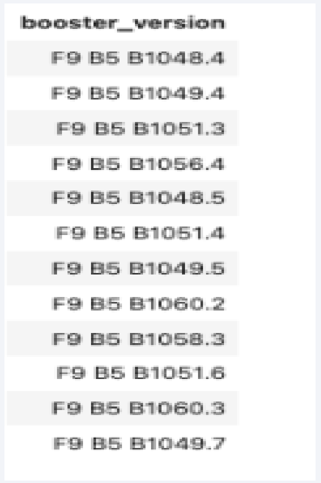


```
%sql select count(LANDING__OUTCOME) from SPACEXTBL where LANDING__OUTCOME like '%Success%' or LANDING__OUTCOME like '%Failure%'
```

73

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Present your query result with a short explanation here
- BOOSTER_VERSION from the data then use a subquery to select the maximum value from the 'PAYLOAD_MASS__KG_' column, then select the booster version that has the maximum 'PAYLOAD_MASS__KG_'.

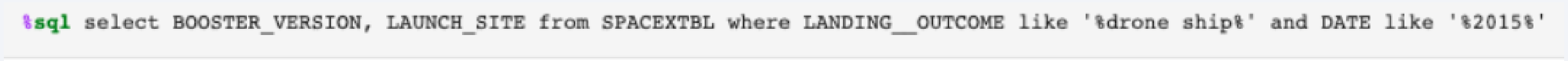


booster_version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

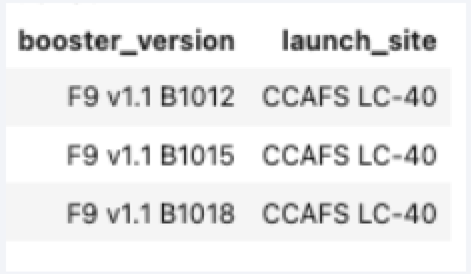
2015 Launch Records

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

- Present your query result with a short explanation here



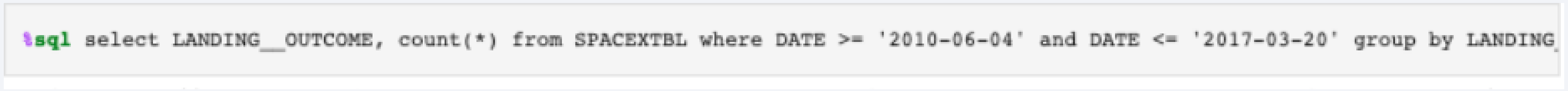
```
%sql select BOOSTER_VERSION, LAUNCH_SITE from SPACEXTBL where LANDING__OUTCOME like '%drone ship%' and DATE like '%2015%'
```



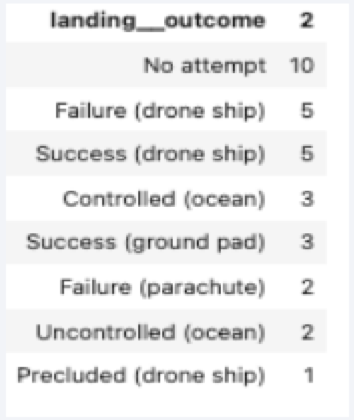
booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40
F9 v1.1 B1018	CCAFS LC-40

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
- Present your query result with a short explanation here



```
%sql select LANDING__OUTCOME, count(*) from SPACEXTBL where DATE >= '2010-06-04' and DATE <= '2017-03-20' group by LANDING_
```

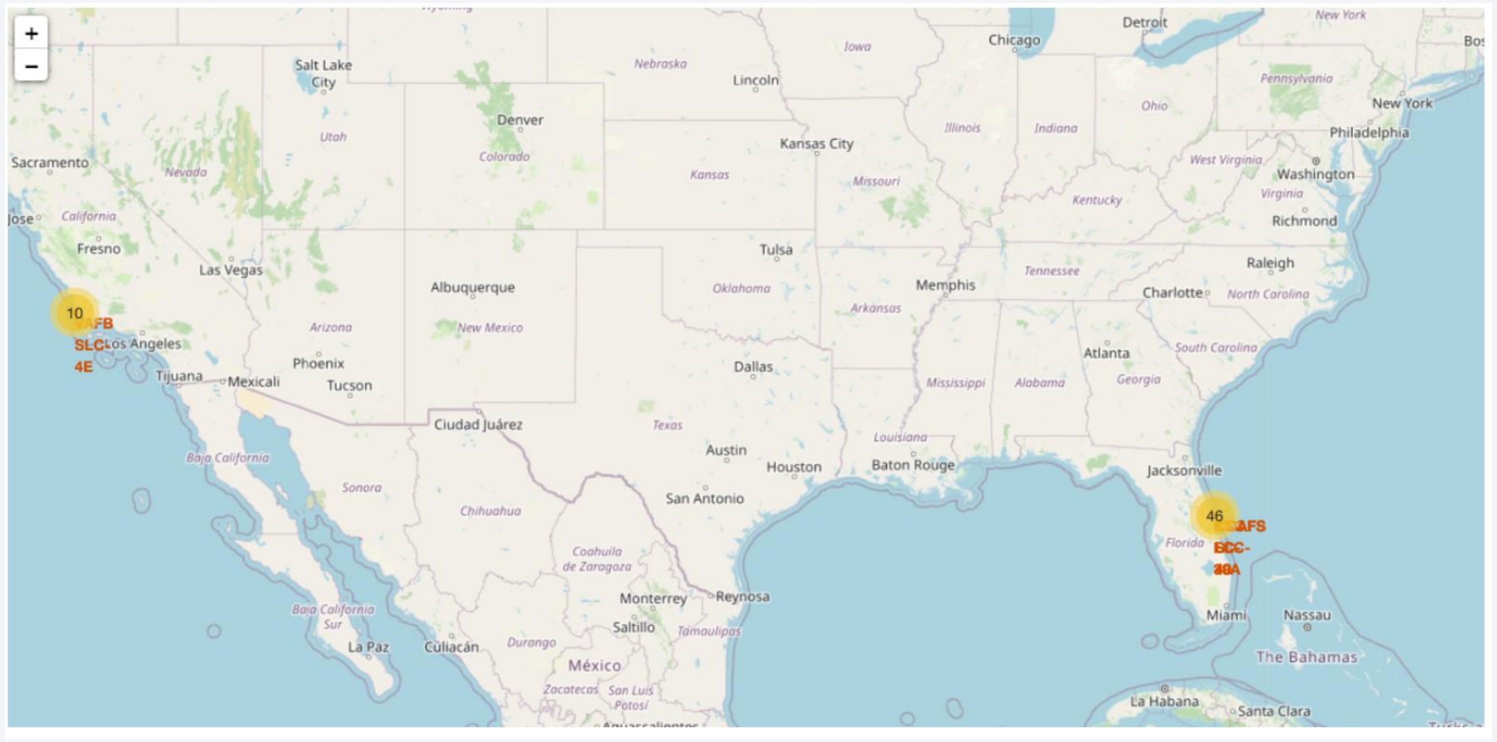


landing__outcome	2
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 3

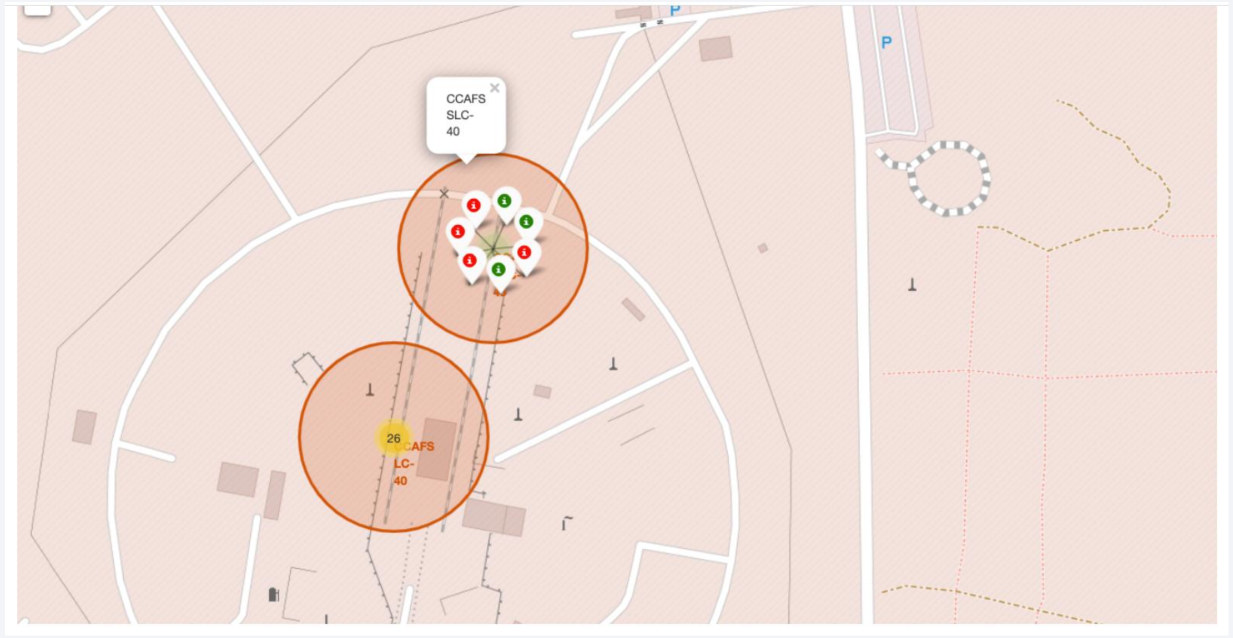
Launch Sites Proximities Analysis

Folium Map – All Launch Sites



- The three launch sites (KSC LC39A, CCAFS SLC-40, and CCAFS LC-40) are located very closed to each other, but VAFB SLC-4E is far away from these 3 sites

Folium Map – CCAFS SLC- 40 with Launch Outcomes



- Color-labeled markers in marker clusters, you should be able to easily identify which launch sites have relatively high success rates.

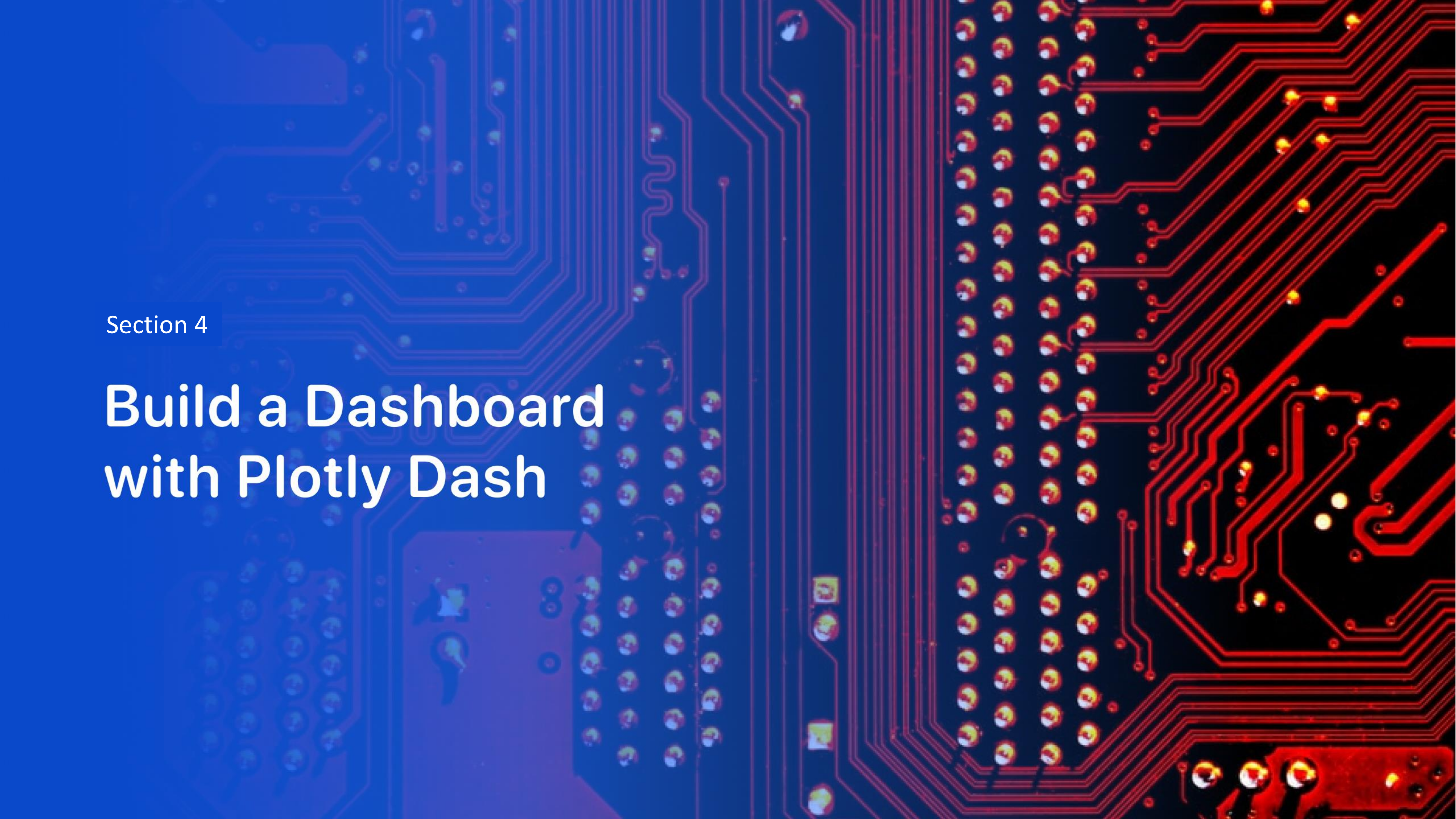
Folium Map - symbol

A railway map symbol may look like this:

A highway map symbol may look like this:

A city map symbol may look like this:

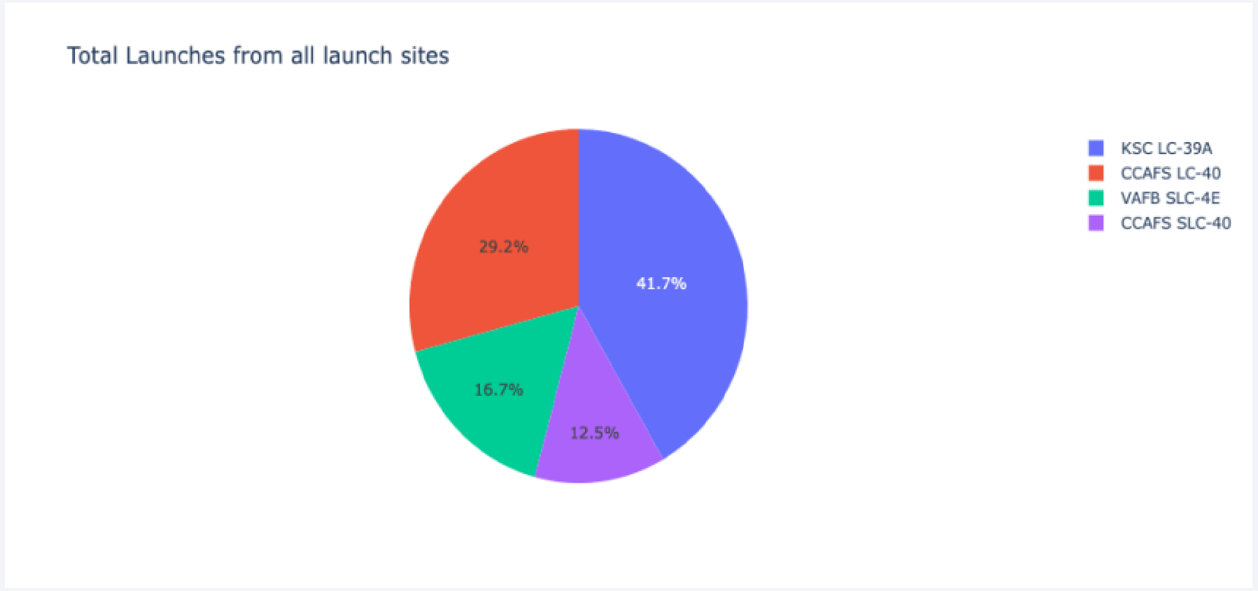
- The line between a launch site to its closest city, railway, highway, etc



Section 4

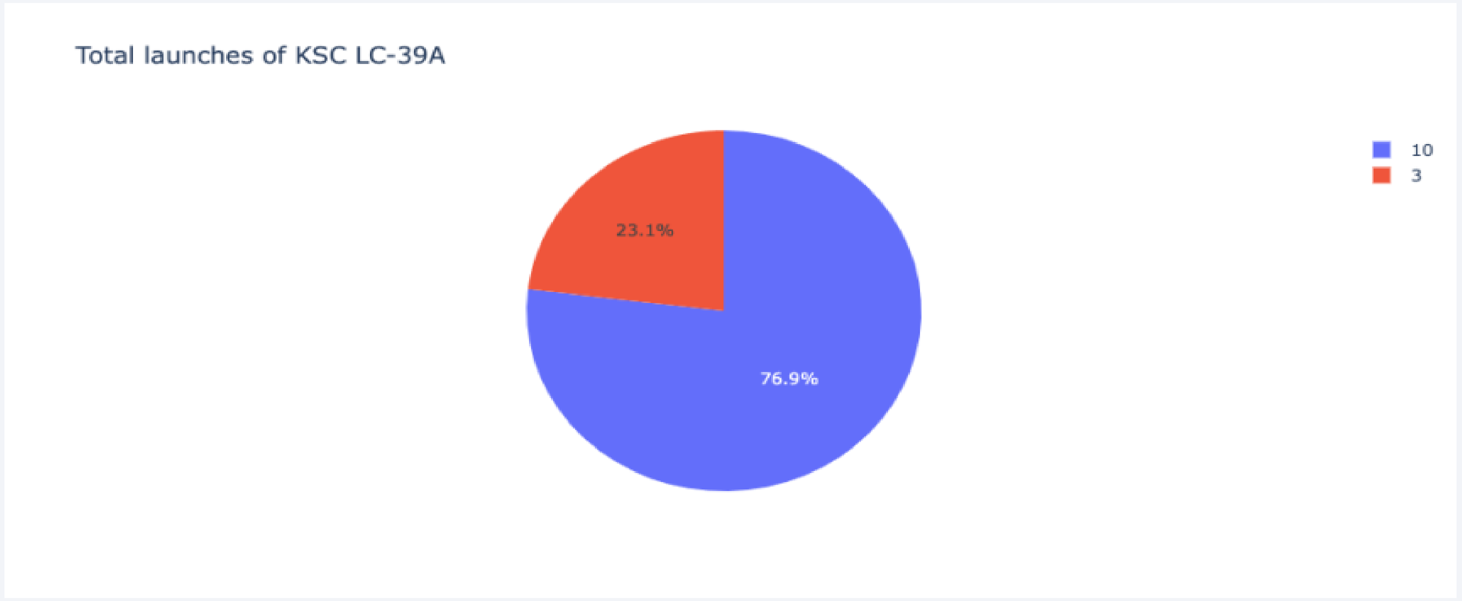
Build a Dashboard with Plotly Dash

Pie chart - Total launches (%) from all launch sites



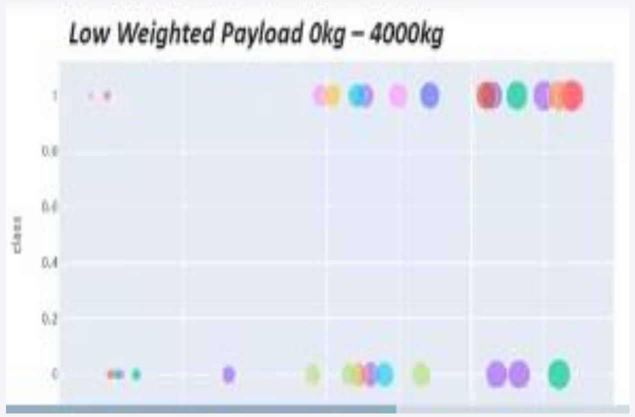
- From this pie chart, we observed that the percentage calculates as per launch sites. KSC LC-39A has the most highest percentage whereas VAFB SLC-4E has the lowest percentage from the launch site.

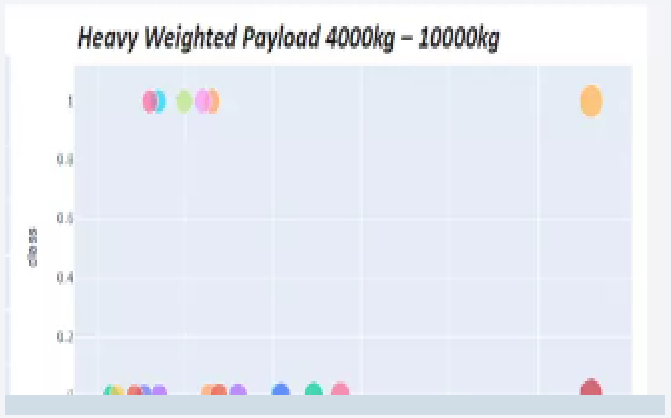
Pie Chart – Success (%) and Failure (%) launch at KSC LC-39A



- From the pie chart, KSC LC-39A has the highest success rate in percentage because of its location.

Payload vs Launch outcome





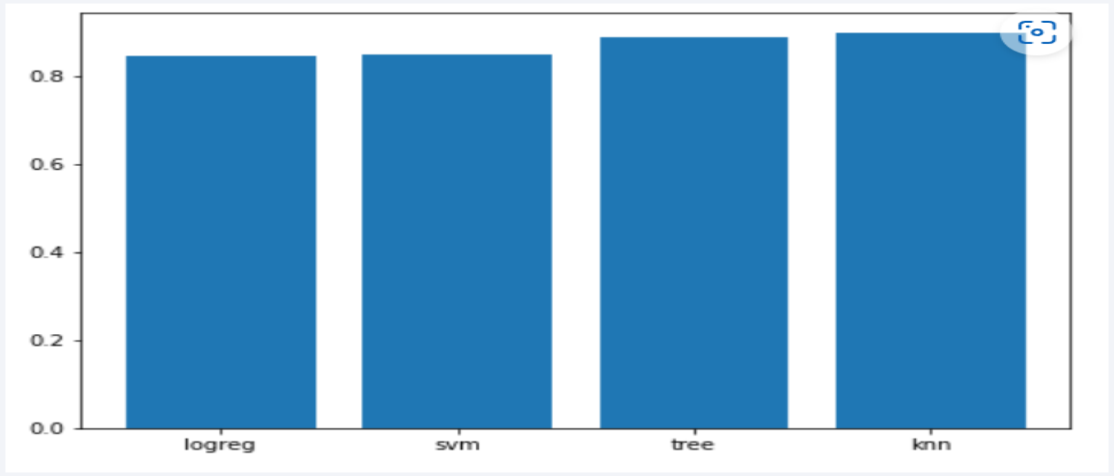
- From this outcome, the success rate for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

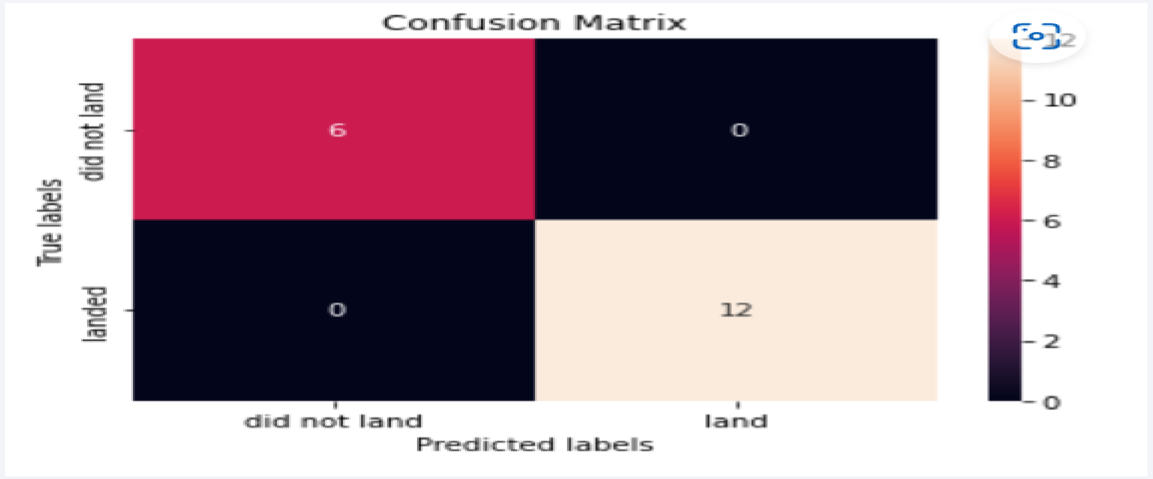
Classification Accuracy

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



- Find which model has the highest classification accuracy
- K-nearest neighbor has the highest accuracy.

Confusion Matrix



- This confusion Matrix is from KNN model.

Conclusions

- SVM, KNN and Logistic Regression Model are the most accurate and best in prediction accuracy for this dataset.
- The success rate for SpaceX launches directly collared in years where they eventually perfect the launches.
- The best success rate for Orbit GEO, HEO, SSO, ES L1.
- KSC LC 39A had the most successful launches compare from all sites

Appendix

- All the notebooks and datasets are present in github repo.
- https://github.com/tushar2704/IBM_DS_Capstone

Thank you!