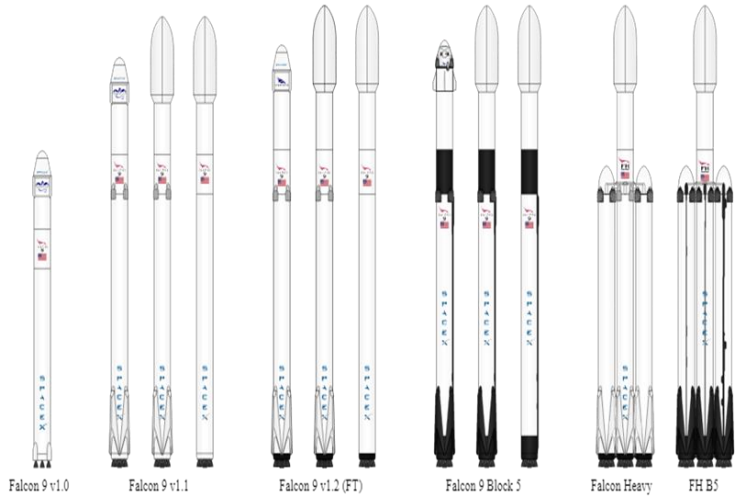
Data Science Capstone project

RAKESH PRABHU

28/08/2021

Outline



- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

- The business person who want to determine if the first stage will land or not, it determine the cost of a launch. The Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.
- Therefore we can determine the first stage will land or not that can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch. In this lab, we will collect and make sure the data is in the correct format from an API.

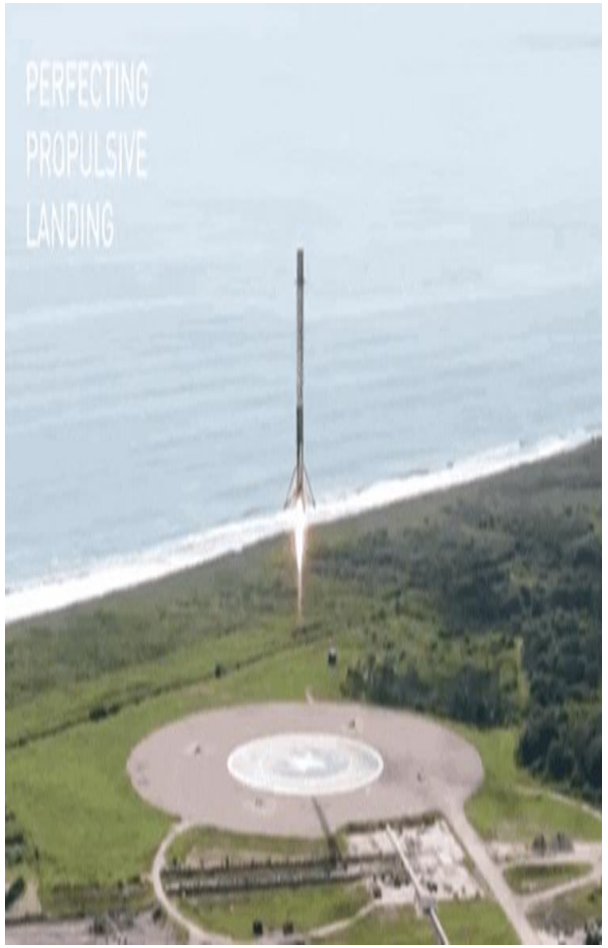
Introduction

In this capstone project, is used to predict if the Falcon 9 first stage will land successfully or Not. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.

Therefore this project tries to determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

In this lab, you will collect and make sure the data is in the correct format from an API.

Methodology



- Data collection methodology:
  - In this lab, you will make a get request to the SpaceX API.
  - Request to the SpaceX API
  - Clean the requested data
- Perform data wrangling
  - The raw data has been collected and we need to improve the quality by performing data wrangling.
- Perform exploratory data analysis (EDA) using visualization and SQL
  - SQL skills are use to address query the data and gather insights.
- Perform interactive visual analytics using Folium and Plotly Dash
  - The basic statistical analysis and data visualization, will help to see directly how variables might be related to each other.
- Perform predictive analysis using classification models
  - The model is developed, evaluate, and refine predictive models for discovering more exciting insights.

Methodology

Data collection

- The 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. Our goal is to use this data to predict whether SpaceX will attempt to land a rocket or not. The SpaceX REST API endpoints, or URL, starts with api.spacexdata.com/v4/. The different end points, for example: /capsules and /cores in our study will be working with the endpoint api.spacexdata.com/v4/launches/past.

- We want to transform this raw data into a clean dataset which provides meaningful data on the situation we are trying to address:
  - Wrangling Data using an API,
  - Sampling Data, and
  - Dealing with Nulls.

Data collection – SpaceX API

```
[ ]  # Create a data from launch_dict
     import pandas as pd
     data=pd.DataFrame(launch_dict)
```

Show the summary of the dataframe

```
[ ]  # Show the head of the dataframe
     data.head()
```

FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude
1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin1A	167.743129
2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2A	167.743129
4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2C	167.743129
5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin3C	167.743129
6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366

Data collection – Web scraping

Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.

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

```
landing_outcomes = True ASDS     41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
None ASDS       2
False Ocean     2
False RTLS      1
Name: Outcome, dtype: int64
```

`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 to a drone ship `False ASDS` means the mission outcome was unsuccessfully landed to a drone ship. `None ASDS` and `None None` these represent a failure to land.

```
[ ] for i,outcome in enumerate(landing_outcomes.keys()):
        print(i,outcome)
```

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.
- In this lab we will mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

- Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose

Data wrangling

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`; otherwise, it's one.
Then assign it to the variable `landing_class`:

```
[ ]  # landing_class = 0 if bad_outcome
     bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
     landing_class = 0
     bad_outcomes = 0

     # landing_class = 1 otherwise
     good_outcomes=set(landing_outcomes.keys()[[0,2,4]])
     landing_class = 1
     good_outcomes=1
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

```
[ ]  df['Class']=landing_class
     df[['Class']].head(8)
```



```
[ ]  df.head(5)
```

ghtNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latit
1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561
2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561
3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561
4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632
5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561

EDA with data visualization

- Instead of presenting your findings in static graphs, interactive data visualization, or dashboarding, can always tell a more appealing story. In this module, we will be using Folium and Plotly Dash to build an interactive map and dashboard to perform interactive visual analytics.

- The first part of this module will be focused on analysing launch site geo and proximities with Folium. We will first mark the launch site locations and their close proximities on an interactive map.

- Then, we can explore the map with those markers and try to discover any patterns from them. Finally, we will be building a dashboard application with the Python Plotly Dash package.

Build an interactive Visual Analytics

- The Interactive Visual Analytics and Dashboard module. Will be used to build a Dashboard for stakeholders.
- Interactive visual analytics enables users to explore and manipulate data in an interactive and real-time way.
- Common interactions including zoom-in and zoom-out, pan, filter, search, and link. With interactive visual analytics, users could find visual patterns faster and more effectively.

Build a Dashboard with Plotly Dash

- we can explore the map with those markers and try to discover any patterns from them.
- Finally, we will be building a dashboard application with the Python Plotly Dash package.
- This dashboard application contains input components such as a dropdown list and a range slider to interact with a pie chart and a scatter point chart. Can be use it to find more insights from the SpaceX dataset more easily than with static graphs.

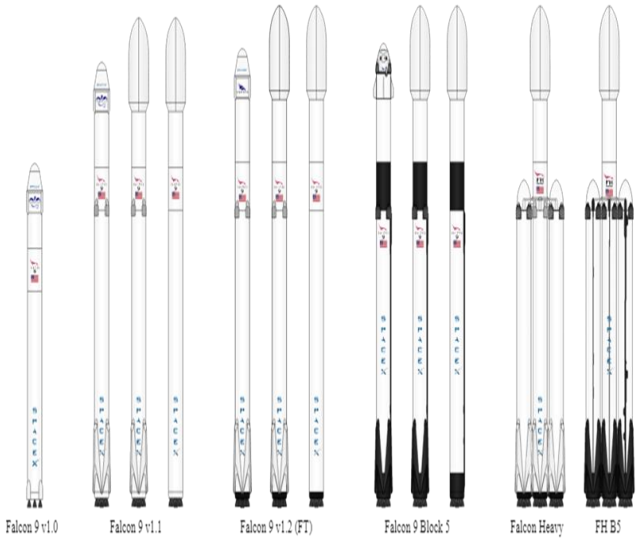
Predictive analysis

- The Predictive Analysis will build a machine learning pipeline to predict if the first stage of the Falcon 9 lands successfully.

- This will include: Pre-processing, allowing us to standardize our data, and Train_test_split, allowing us to split our data into training and testing_data, then train the model and perform Grid Search, allowing us to find the hyper parameters that allow a given algorithm to perform best.

- Using the best hyper parameter values, we will determine the model with the best accuracy using the training data and test Logistic Regression, Support Vector machines, Decision Tree Classifier, and K-nearest neighbors. Finally, we will output the confusion matrix and result are shown in the figure.

Results



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

EDA with Visualization

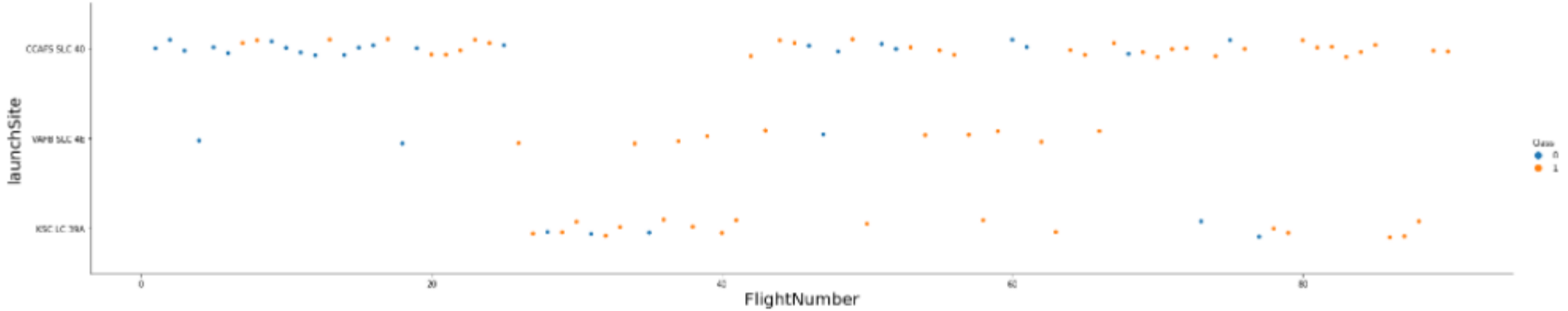
In this work the prediction is done if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is due to the fact that SpaceX can reuse the first stage. In this work Exploratory Data Analysis and Feature Engineering helps to find Falcon 9 first stage will land successfully

Flight Number vs. Launch Site

TASK 1: Visualize the relationship between Flight Number and Launch Site

Use the function `catplot` to plot `FlightNumber` vs `LaunchSite`, set the parameter `x` parameter to `FlightNumber`,set the `y` to `Launch Site` and set the parameter `hue` to `'class'`

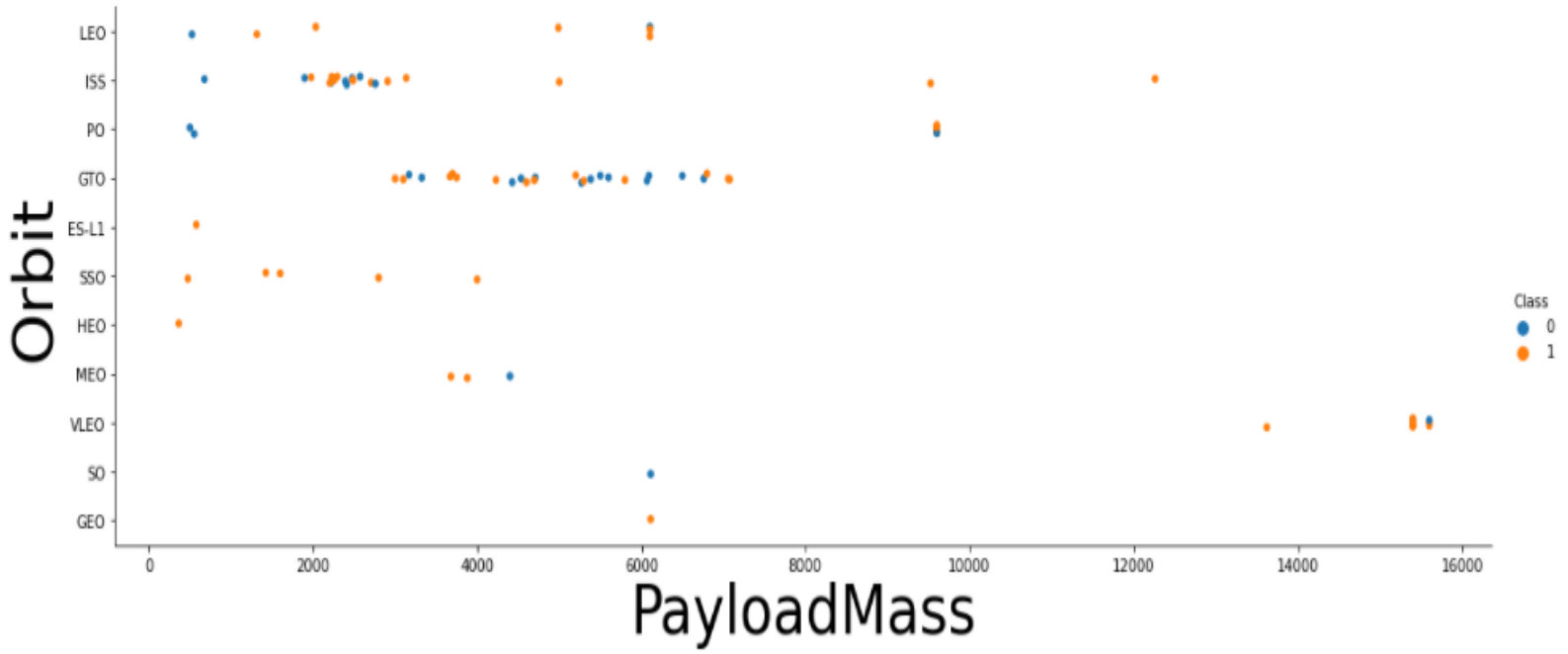
In [6]:
```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("FlightNumber",fontsize=20)
plt.ylabel("launchSite",fontsize=20)
plt.show()
```



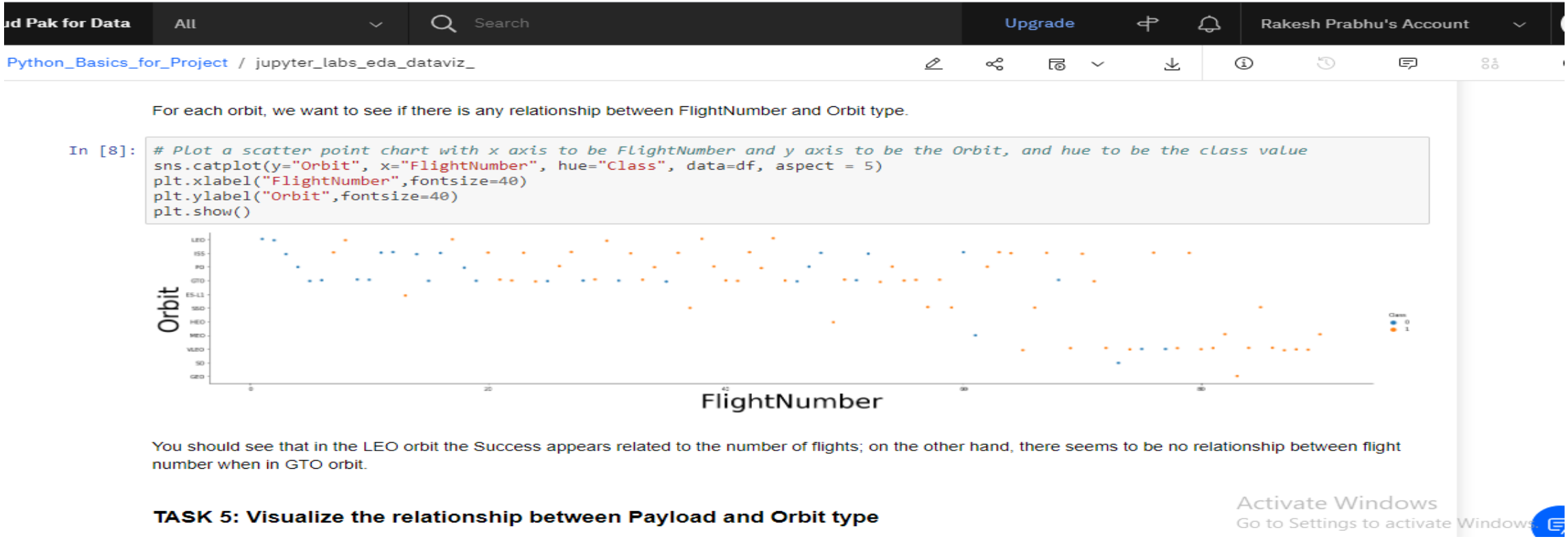
Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

Payload vs. Launch Site

In [9]:
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 3)
plt.xlabel("PayloadMass",fontsize=40)
plt.ylabel("Orbit",fontsize=40)
plt.show()
```



Flight Number vs. Orbit type



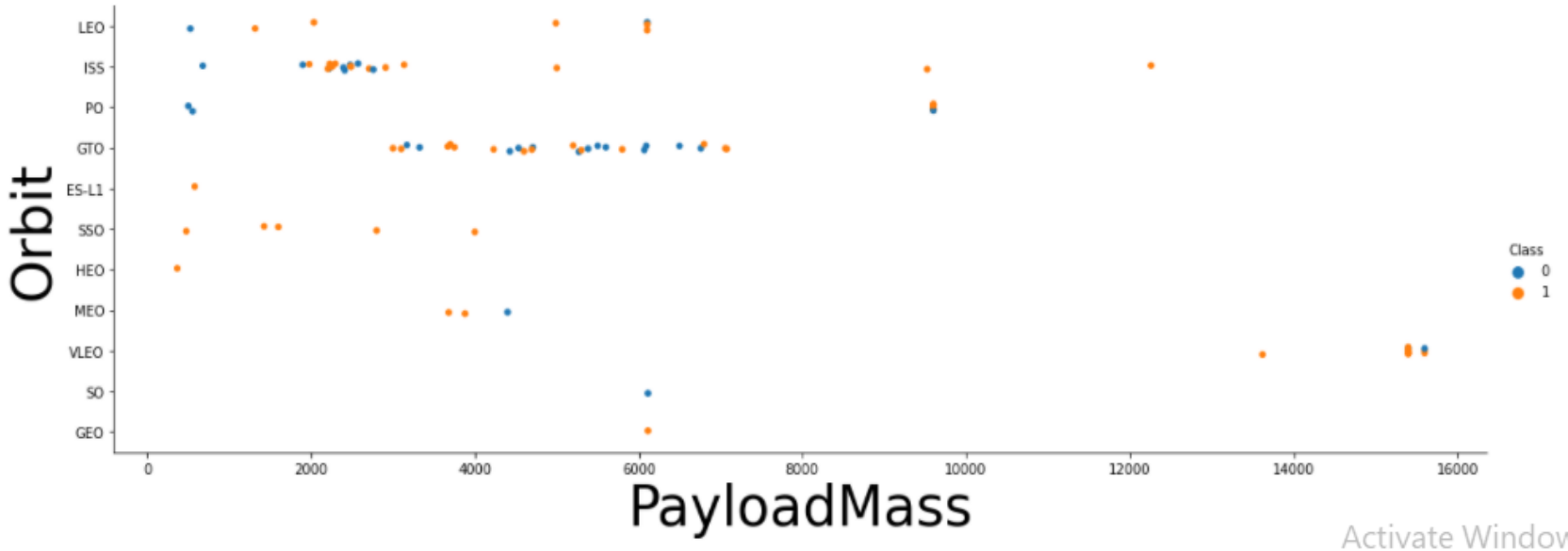
Payload vs. Orbit type

IBM Cloud Pak for Data | All | Search | Upgrade | Rakesh Prabhu's Account | RP

Projects / Python_Basics_for_Project / jupyter_labs_eda_dataviz_

Similarly, we can plot the Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type

In [9]:
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 3)
plt.xlabel("PayloadMass",fontsize=40)
plt.ylabel("Orbit",fontsize=40)
plt.show()
```



Launch success yearly trend

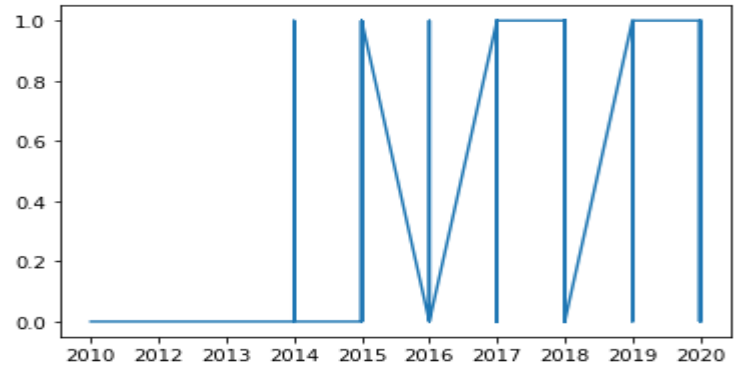
ud Pak for Data | All | Search | Upgrade | Rakesh Prabhu's Account

Python_Basics_for_Project / jupyter_labs_eda_dataviz_

```
89   2020

[90 rows x 1 columns]>
```

In [13]:
```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate

Y=df['Class']
plt.plot(X, Y)
```

Out[13]:
```
[<matplotlib.lines.Line2D at 0x7f201fec6e90>]
```



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

EDA with SQL

SQl helps to determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch. This dataset includes a record for each payload carried during a SpaceX mission into outer space.

All launch site names

Tasks

Now write and execute SQL queries to solve the assignment tasks.

Task 1

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

```
%sql SELECT DISTINCT("LAUNCH_SITE") as " unique launch sites"  FROM SPACEX
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

unique launch sites
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

Launch site names begin with `CCA`

[ ] KSC LC-39A
VAFB SLC-4E

Task 2

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

```
[ ] %sql select * from SPACEX WHERE "LAUNCH_SITE" = 'CCAFS SLC-40' LIMIT 5;
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2017-12-15	15:36:00	F9 FT B1035.2	CCAFS SLC-40	SpaceX CRS-13	2205	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2018-01-08	01:00:00	F9 B4 B1043.1	CCAFS SLC-40	Zuma	5000	LEO	Northrop Grumman	Success (payload status unclear)	Success (ground pad)
2018-01-31	21:25:00	F9 FT B1032.2	CCAFS SLC-40	GovSat-1 / SES-16	4230	GTO	SES	Success	Controlled (ocean)
2018-03-06	05:33:00	F9 B4 B1044	CCAFS SLC-40	Hispasat 30W-6 PODSat	6092	GTO	Hispasat NovaWurks	Success	No attempt
2018-04-02	20:30:00	F9 B4 B1039.2	CCAFS SLC-40	SpaceX CRS-14	2647	LEO (ISS)	NASA (CRS)	Success	No attempt

Task 3

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

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 carried by boosters launched by NASA (CRS)" FROM SPACEX  where "CUSTOMER"='NASA (CRS)'
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

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

45596

Task 4

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

```
[ ] %sql SELECT AVG("PAYLOAD_MASS__KG_") as "average payload mass carried by booster version F9 v1.1" FROM SPACEX  where "BOOSTER_VERSION"='F9 v1.1'
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

average payload mass carried by booster version F9 v1.1

2928

Average payload mass by F9 v1.1

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

```
%sql SELECT SUM("PAYLOAD_MASS__KG_") as "total payload mass carried by boosters launched by NASA (CRS)" FROM SPACEX  where "CUSTOMER"='NASA (CRS)'
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

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

45596

Task 4

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

```
[ ] %sql SELECT AVG("PAYLOAD_MASS__KG_") as "average payload mass carried by booster version F9 v1.1" FROM SPACEX  where "BOOSTER_VERSION"='F9 v1.1'
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

average payload mass carried by booster version F9 v1.1

2928

First successful ground landing date

Task 5

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

Hint:Use min function

```
[ ] %sql SELECT * FROM SPACEX  where  "LANDING_OUTCOME"= 'Success';
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

DATE	time_utc_	booster_version	launch_site	payload	payload_mass_kg_	orbit	customer	mission_outcome	landing_outcome
2018-07-22	05:50:00	F9 B5B1047.1	CCAFS SLC-40	Telstar 19V	7075	GTO	Telesat	Success	Success
2018-07-25	11:39:00	F9 B5B1048.1	VAFB SLC-4E	Iridium NEXT-7	9600	Polar LEO	Iridium Communications	Success	Success
2018-08-07	05:18:00	F9 B5 B1046.2	CCAFS SLC-40	Merah Putih	5800	GTO	Telkom Indonesia	Success	Success
2018-09-10	04:45:00	F9 B5B1049.1	CCAFS SLC-40	Telstar 18V / Apstar-5C	7060	GTO	Telesat	Success	Success
2018-10-08	02:22:00	F9 B5 B1048.2	VAFB SLC-4E	SAOCOM 1A	3000	SSO	CONAE	Success	Success

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

```
%sql SELECT COUNT("MISSION_OUTCOME") FROM SPACEX  where  "MISSION_OUTCOME"= 'Success';
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
 1
99
```

```
%sql SELECT COUNT("MISSION_OUTCOME") FROM SPACEX  where  "MISSION_OUTCOME"= 'Failure ';
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
 1
0
```

Task 9

Boosters carried maximum payload

Task 9

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

```
[ ] %sql SELECT * From SPACEX where "LANDING__OUTCOME"= 'Failure'
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2018-12-05	18:16:00	F9 B5B1050	CCAFS SLC-40	SpaceX CRS-16	2500	LEO (ISS)	NASA (CRS)	Success	Failure
2020-02-17	15:05:00	F9 B5 B1056.4	CCAFS SLC-40	Starlink 4 v1.0, SpaceX CRS-20	15600	LEO	SpaceX	Success	Failure
2020-03-18	12:16:00	F9 B5 B1048.5	KSC LC-39A	Starlink 5 v1.0, Starlink 6 v1.0	15600	LEO	SpaceX	Success	Failure

Task 10

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

```
[ ] %sql SELECT COUNT("LANDING__OUTCOME") FROM SPACEX  where  ("LANDING__OUTCOME"= 'Failure(drone ship)')
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

2015 and recent launch records

Task 9

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

```
[ ]  %sql SELECT * From SPACEX where "LANDING__OUTCOME"= 'Failure'
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2018-12-05	18:16:00	F9 B5B1050	CCAFS SLC-40	SpaceX CRS-16	2500	LEO (ISS)	NASA (CRS)	Success	Failure
2020-02-17	15:05:00	F9 B5 B1056.4	CCAFS SLC-40	Starlink 4 v1.0, SpaceX CRS-20	15600	LEO	SpaceX	Success	Failure
2020-03-18	12:16:00	F9 B5 B1048.5	KSC LC-39A	Starlink 5 v1.0, Starlink 6 v1.0	15600	LEO	SpaceX	Success	Failure

Task 10

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

```
[ ]  %sql SELECT COUNT("LANDING__OUTCOME") FROM SPACEX  where  ("LANDING__OUTCOME"= 'Failure(drone ship)')
```

```
 * ibm_db_sa://zhr81900:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Interactive map with Folium

Interactive map with Folium

This will help to get this task as follows

1: Mark all launch sites on a map

2: Mark the success/failed launches for each site on the map

3: Calculate the distances between a launch site to its proximities

<Folium map screenshot 1>

Now, you can take a look at what are the coordinates for each site.

```
# Select relevant sub-columns: `Launch Site`, `Lat(Latitude)`, `Long(Longitude)`, `class`
spacex_df = spacex_df[['Launch Site', 'Lat', 'Long', 'class']]
launch_sites_df = spacex_df.groupby(['Launch Site'], as_index=False).first()
launch_sites_df = launch_sites_df[['Launch Site', 'Lat', 'Long']]
launch_sites_df
```

	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610746

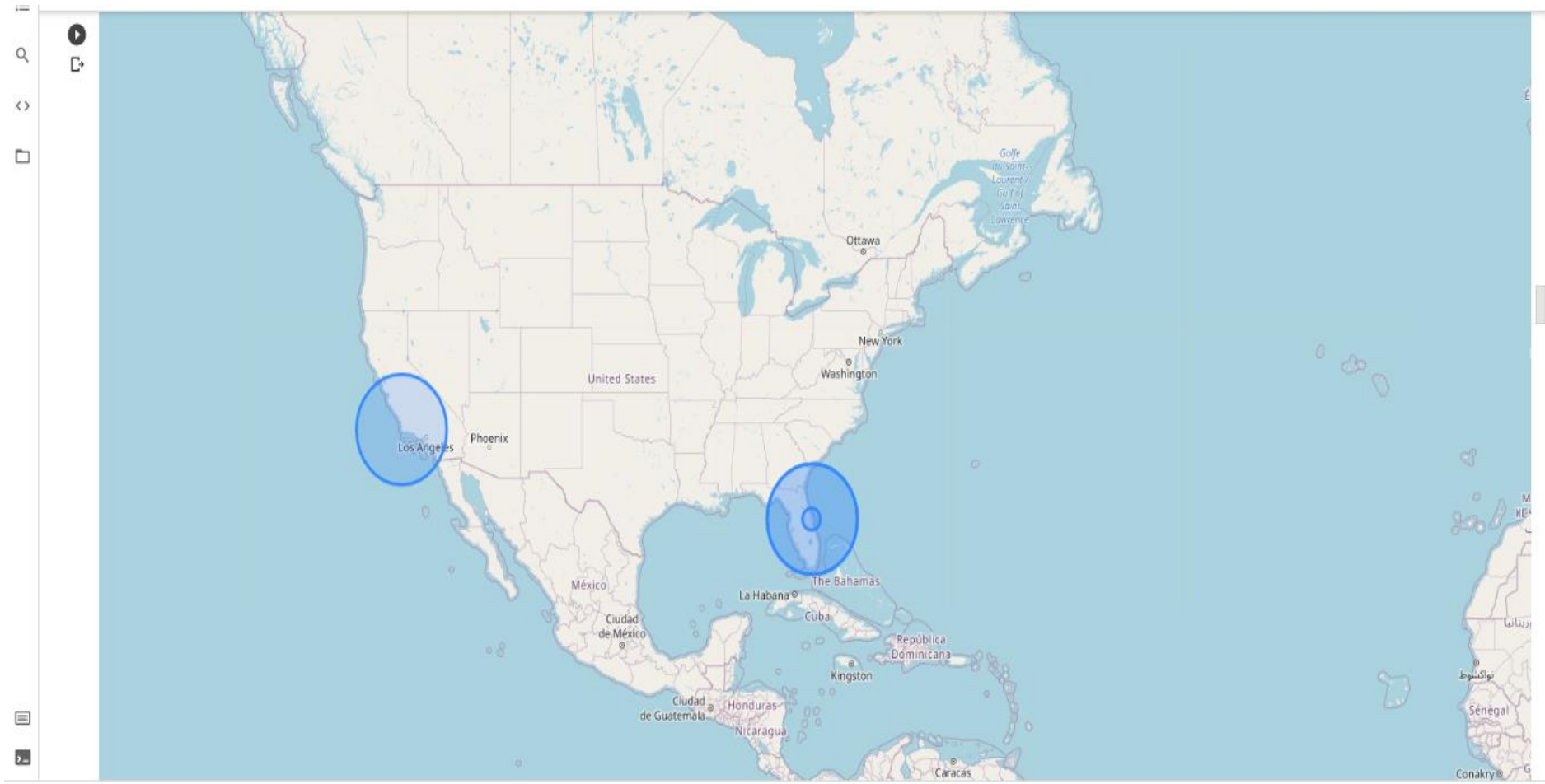
Above coordinates are just plain numbers that can not give you any intuitive insights about where are those launch sites. If you are very good at geography, you can interpret those numbers directly in your mind. If not, that's fine too. Let's visualize those locations by pinning them on a map.

We first need to create a folium `Map` object, with an initial center location to be NASA Johnson Space Center at Houston, Texas.

```
[ ] # Start location is NASA Johnson Space Center
    nasa_coordinate = [29.559684888503615, -95.0830971930759]
    site_map = folium.Map(location=nasa_coordinate, zoom_start=10)
```

We could use `folium.Circle` to add a highlighted circle area with a text label on a specific coordinate. For example,

<Folium map screenshot 2>



<Folium map screenshot 3>

Your updated map with distance line should look like the following screenshot:

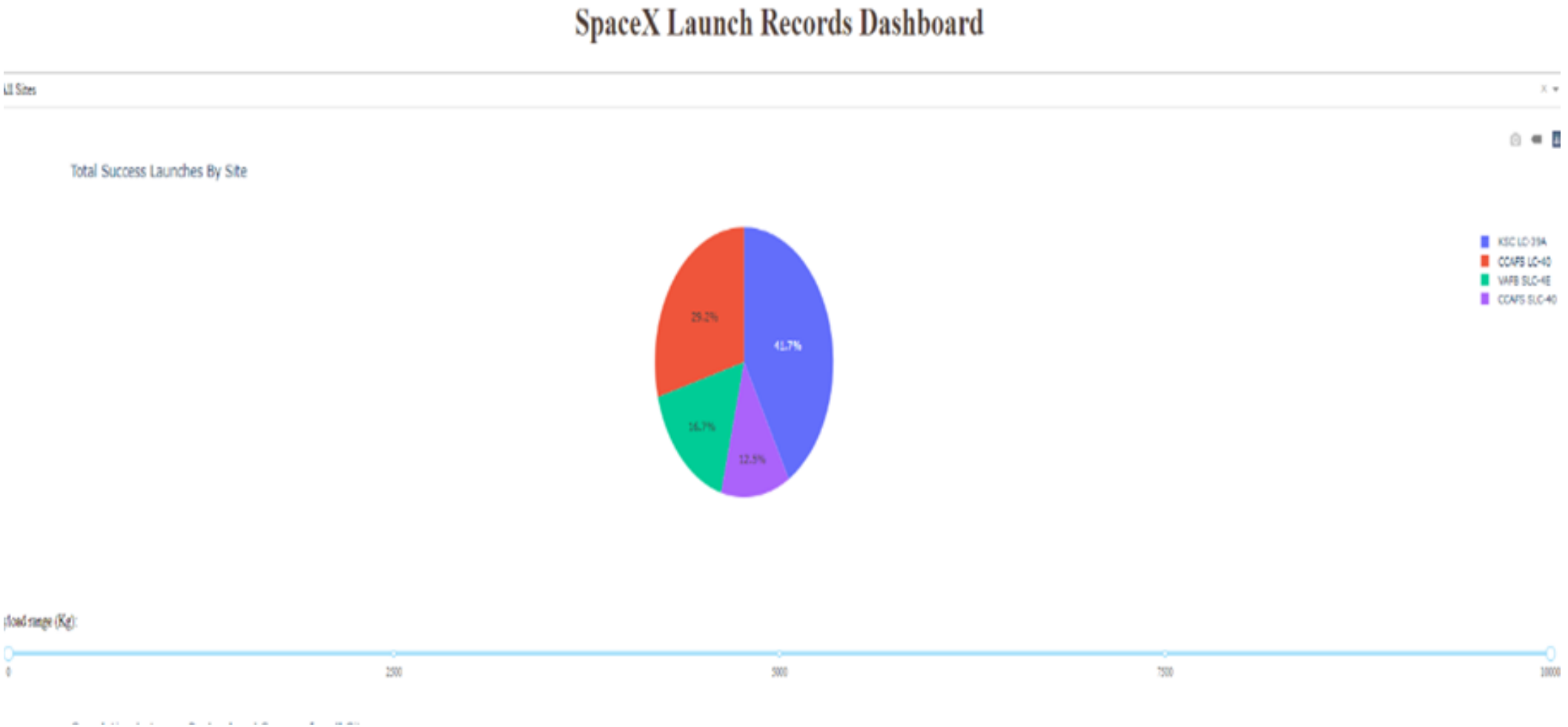
<Folium map screenshot 4>

```
# create and add a folium.Marker on your selected closest raiwaly point on the map
# show the distance to the launch site using the icon property
print(distance_railway)

m1=folium.Map(location=VAFBSL)

folium.CircleMarker(
    location=[30.25726,-81.6032],
    radius=50,
    popup="P1",
    fill=True,
    icon=folium.Icon(icon="cloud"),
).add_to(m1)

folium.CircleMarker(
    location=[28.563197,-80.576820],
    radius=50,
    popup="p2",
    fill=True,
    icon=folium.Icon(color="green"),
).add_to(m1)
```

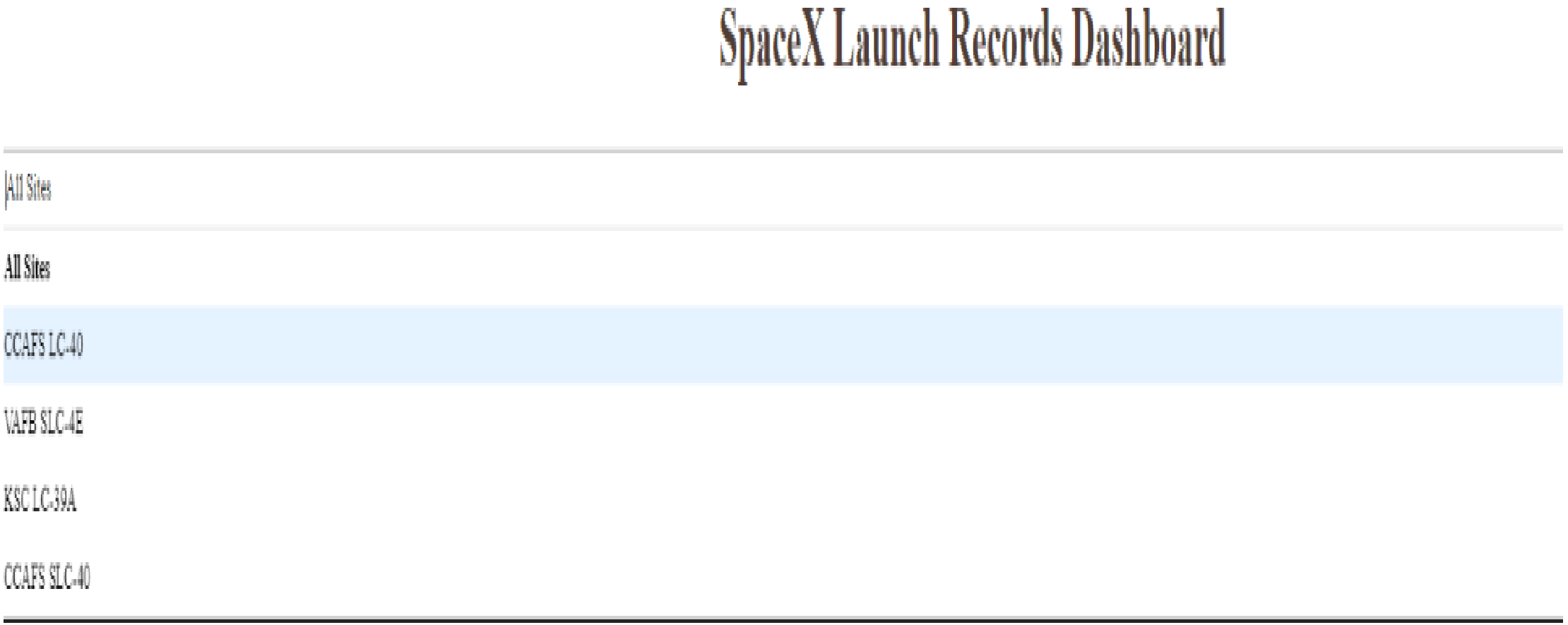
```
213.06122918703852
<folium.vector_layers.CircleMarker at 0x7f3cb56cb490>
```

Build a Dashboard with Plotly Dash

<Dashboard screenshot 1>



<Dashboard screenshot 2>



Predictive analysis (Classification)

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.

This information can be used if an alternate company wants to bid against space X for a rocket launch. This work create a machine learning pipeline to predict if the first stage will land given the data from the preceding labs.

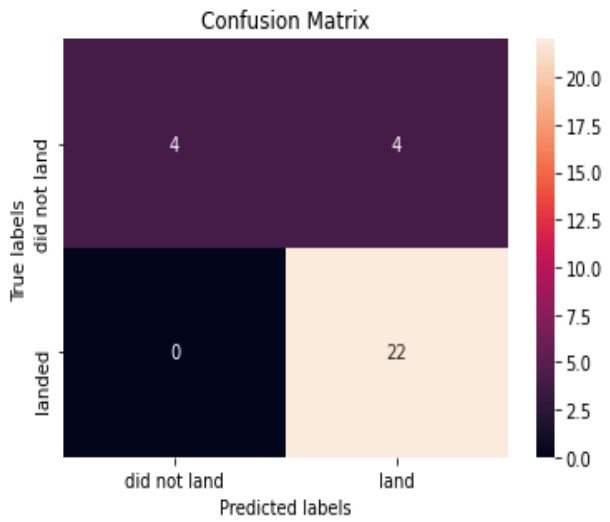
Confusion Matrix of KNN Model

```
[ ] print(knn_cv.score(X_test, y_test))
```

```
0.8666666666666667
```

We can plot the confusion matrix

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

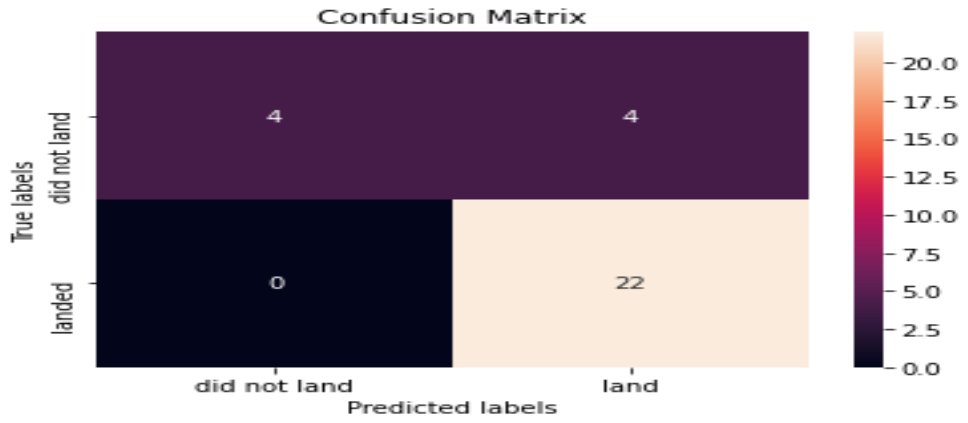


Confusion Matrix of Logistic Regression Model

```
[ ]  print(logreg_cv.score(X_test,y_test))

     0.8666666666666667
```

Lets look at the confusion matrix:

```
[ ]  yhat=logreg_cv.predict(X_test)
     plot_confusion_matrix(y_test,yhat)
```



CONCLUSION

- Performing Request to the SpaceX API and Clean the requested data from collected data.
- Performing exploratory Data Analysis and determine using Data wrangling
- Dataset includes a record for each payload carried during a SpaceX mission into outer space. his information can be used if an alternate company wants to bid against SpaceX for a rocket launch, SQL helps to make analysis
- performing more interactive visual analytics using Folium help to get detailed picture
- Performing exploratory Data Analysis and Feature Engineering using Pandas and Matplotlib has performed
- machine learning pipeline to predict if the first stage will land given the data from the preceding labs. KNN and Logistic Regression Model shows better result


Thanks!