IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Andrew Hirst>
<11th June 2024>

Outline

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

Executive Summary

- Summary of methodologies

  Data collection via Pandas

  Data collection API keys

  Using Mapping via Folium

    Machine learning for the predictions

- Summary of all results

  Data is illustrated via Tables & charts with some charts showing predictions.

Introduction

- Project background and context

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 if we can determine if the first stage will land through producing was probability models with Python/Machine Learning, 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.

- Problems you want to find answers

Does the launch site effect the probability of the rocket successfuly launching or not.

The interaction amongst various features that determine the success rate of a successful landing.

What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected via a API key then converted into a json file to normalize the data prior to conversion to a dataframe
- Perform data wrangling
  - Using One Hot encoding applied to categorical columns in the data.
- Perform exploratory data analysis (EDA) using visualization and SQL
- 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 was collected using the various methods :-

Using the API from SpaceX to obtain the data.

Loaded into a JSON File, which was then converted into a Pandas dataframe.

Data was then cleaned for missing data using averages to fill the missing gaps.

We then run the data through some Machine Learning models and then compare which one gives the most acurate prodictions.

Data Collection – SpaceX API

- We firstly load the SpaceX Data from a API (http) address then load the response content into a json file to normalize the data before converted into a dataframe.
- 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

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
print(response)
```

```
<Response [200]>
```

```
response = requests.get(spacex_url)
data = response.json()
data = pd.json_normalize(data)
data.head()
```

Data Collection - Scraping

- You will notice that a lot of the data are ID's. For example the rocket column has no information about the rocket just an indentification number,so we will use the API again to get the information.
- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that have multiple payloads in a single rocket
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Data Wrangling

We notice in the table under the PayloadMass column with have some nulls as confirmed ->
We will replace these NAN values with the mean average.Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose

```
data.isnull().sum()
```

```
FlightNumber      0
Date              0
BoosterVersion    0
PayloadMass       6
Orbit             0
LaunchSite        0
Outcome           0
Flights           0
GridFins          0
Reused            0
Legs              0
LandingPad       30
Block             4
ReusedCount       0
Serial            0
Longitude         0
Latitude          0
dtype: int64
```

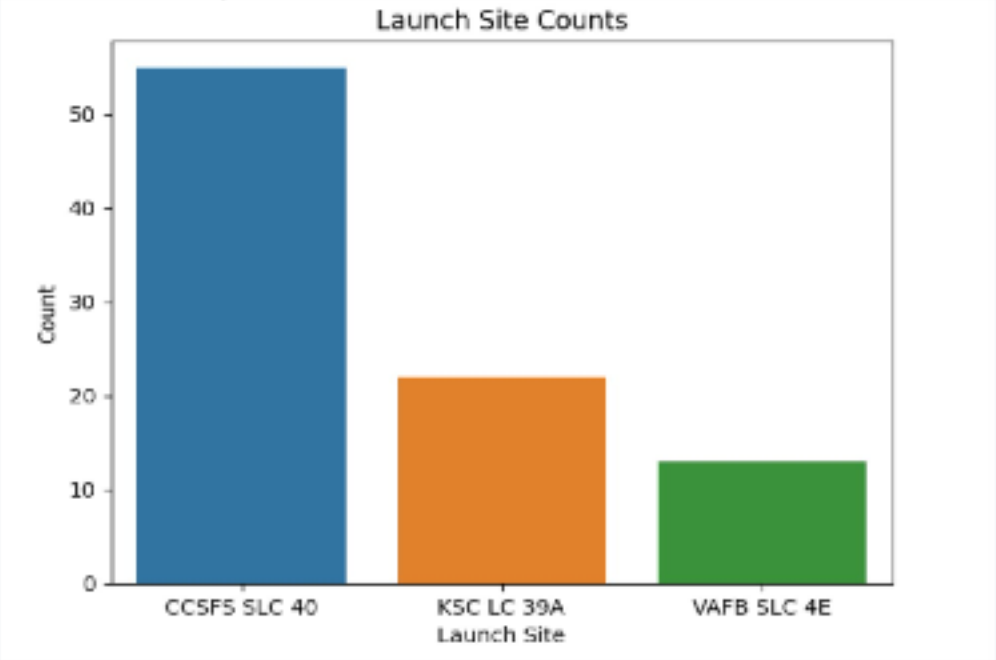
```
# Calculate the mean value of PayloadMass column
mean_calc = data_falcon9.PayloadMass.mean()
# Replace the np.nan values with its mean value
data_falcon9.loc[data_falcon9['PayloadMass'].isnull(),'PayloadMass'] = mean_calc
```

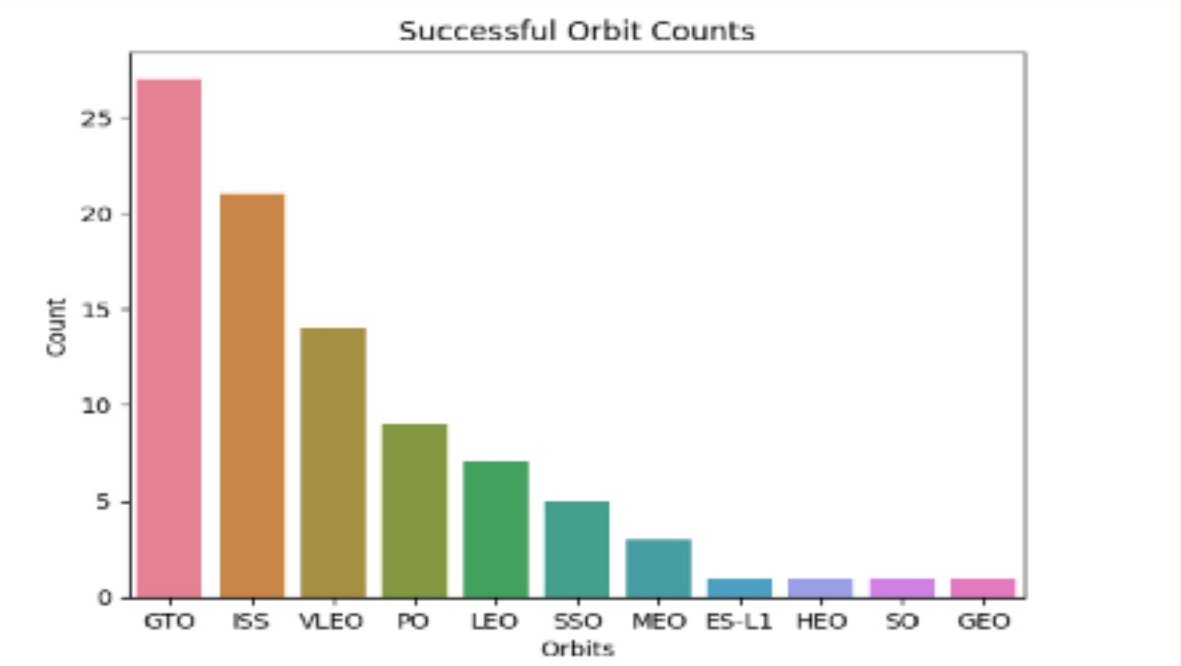
```
data_falcon9.isnull().sum()
```

```
FlightNumber      0
Date              0
BoosterVersion    0
PayloadMass       0
Orbit             0
LaunchSite        0
Outcome           0
Flights           0
GridFins          0
Reused            0
Legs              0
LandingPad       26
Block             0
ReusedCount       0
Serial            0
Longitude         0
Latitude          0
dtype: int64
```

EDA with Data Visualization

We firstly looked at the number of successfull launches between the launch sites and also the number of orbits with Barcharts as per below:-





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

EDA with SQL

Using bullet point format, summarize the SQL queries you performed

List the Unique Launch Sites

List all Launch Sites with names starting with 'CCA'

Total Payload Mass in KG

Total AVG Payload Mass in KG

Successful & Un-Successful Missions

Failed landings on Drone ship between 2015-01-01 & 2015-12-31

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

Build an Interactive Map with Folium

We built a python Folium map showing all the SpaceX launch sites within the USA, also placed circles with the number of successful/un-successfull launches - Green for successful and Red for un-successful.

Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

Build a Dashboard with Plotly Dash

Within the Plotly Dash board we are showing a pie chart which shows the successful launches -v- the un-successful ones in % terms for each spacex launch station

Also showing a scatter graph showing the relationship between the outcome of the different Booster Versions and the Payload mass load.

Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

Predictive Analysis (Classification)

We imported all the relevent Python Libaries for machine learning/prediction modeling, we then went through each one of the learning models and compared the test score for each one to discover which had the best score for predicting the successful landing of the Spacex falcon rocket.

Prior to running the data through the models we performed a Train,Test,Split on the data applying 20% of the data to be test data, then used the method GridSearchCV for each learning model to fit the data(X_train, Y_Train) and then showed the accuracy score for each model.

Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

Results

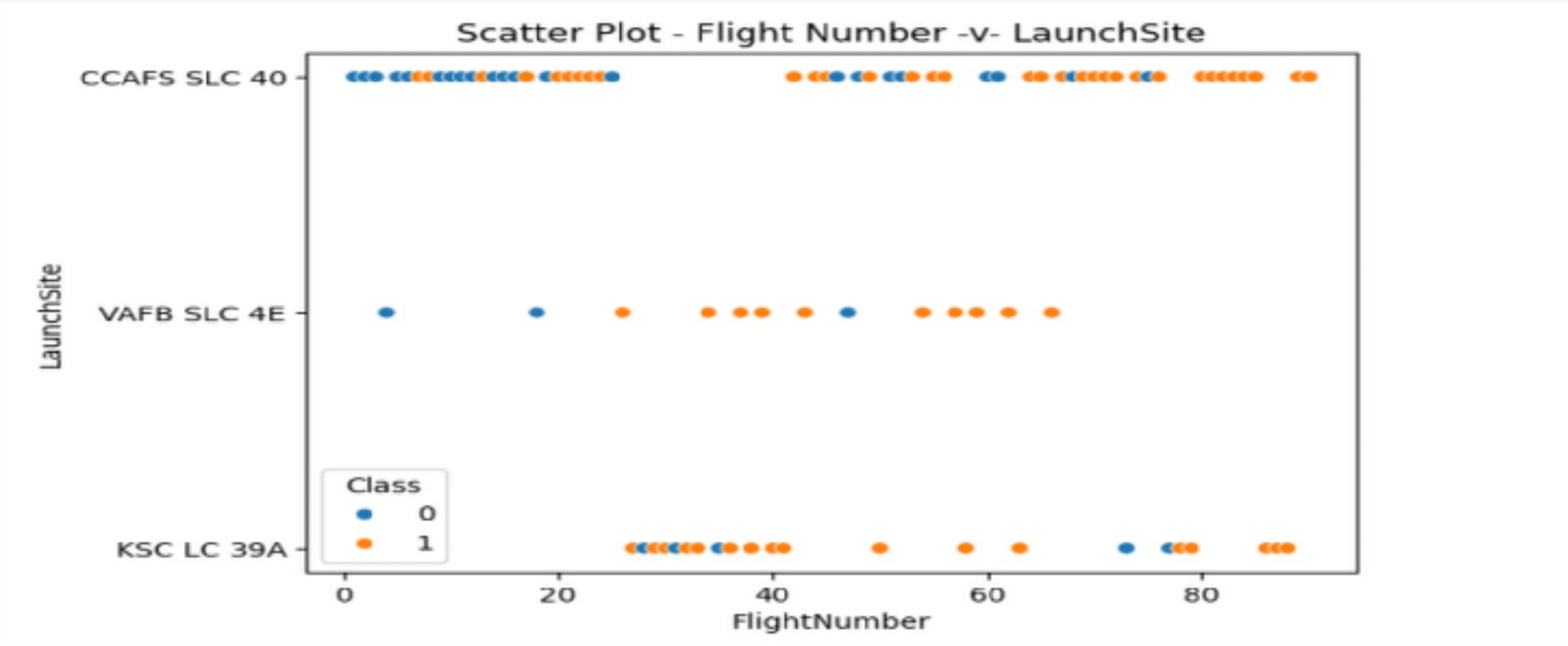
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

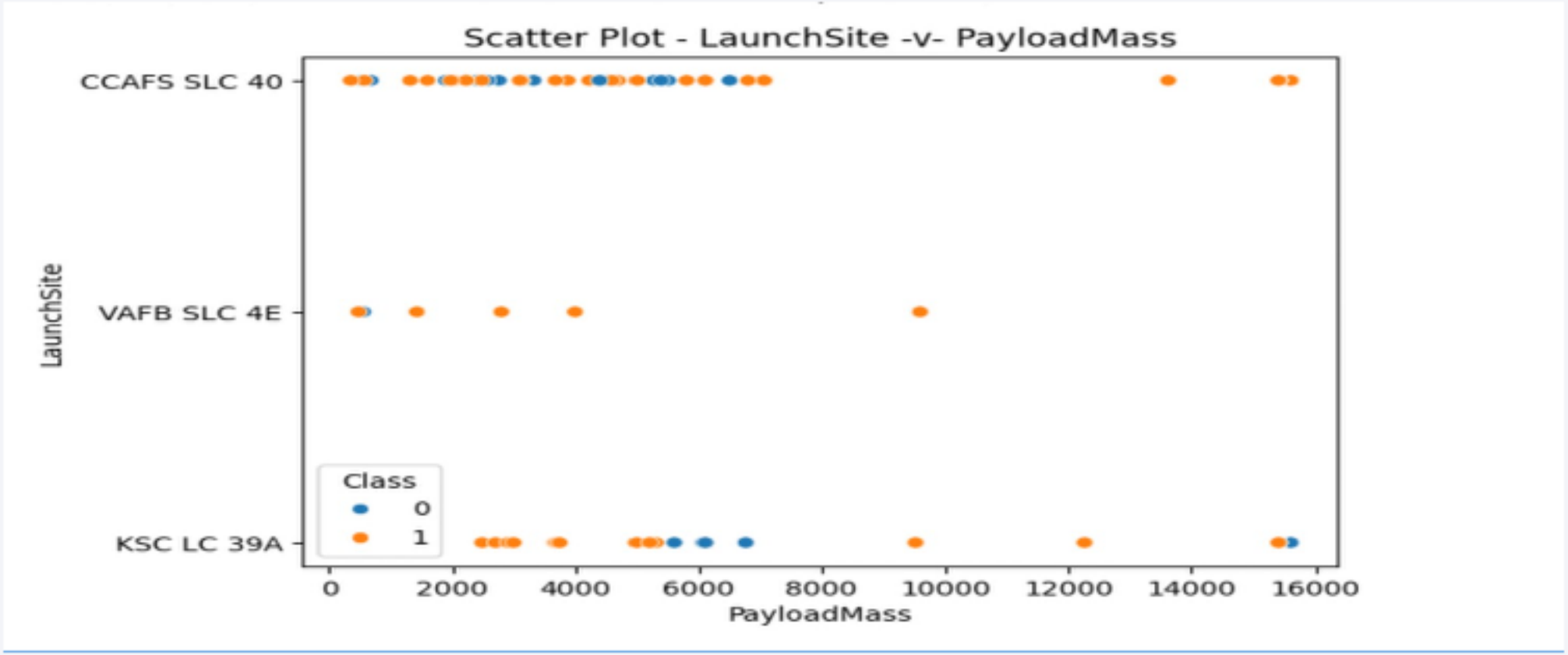
Flight Number vs. Launch Site

- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site, LaunchSite appears to be the site where most of the successful launches have taken place.



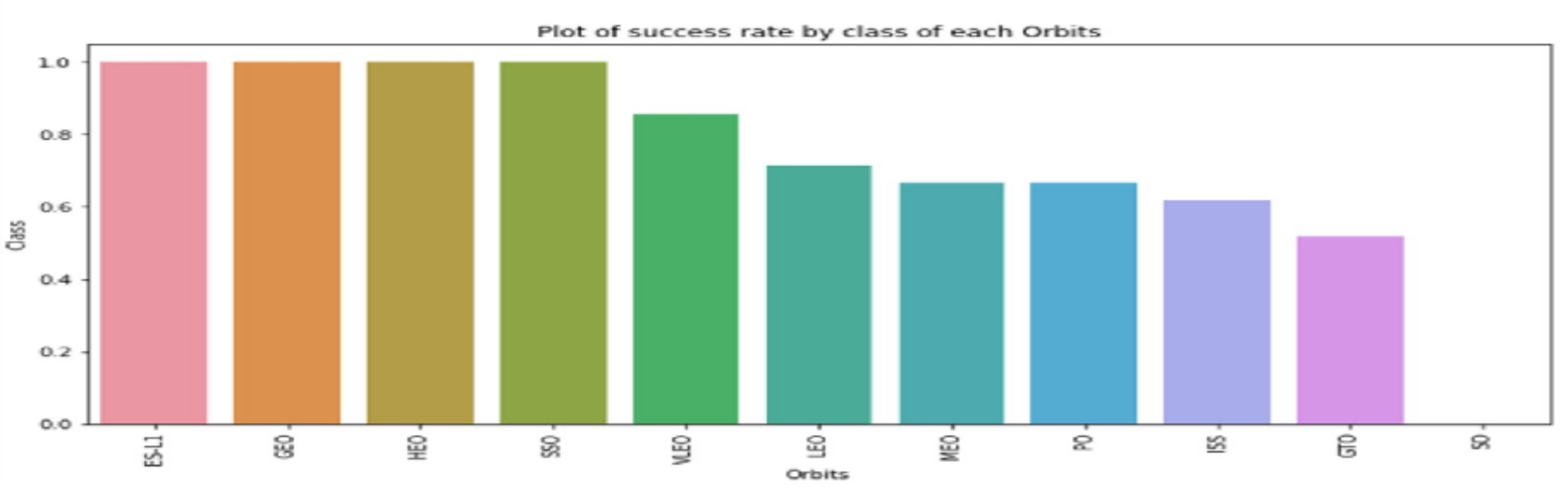
Payload vs. Launch Site

The greatest amount of launches with smaller PayloadMass appears to be at launch site CCAFS SLC 40



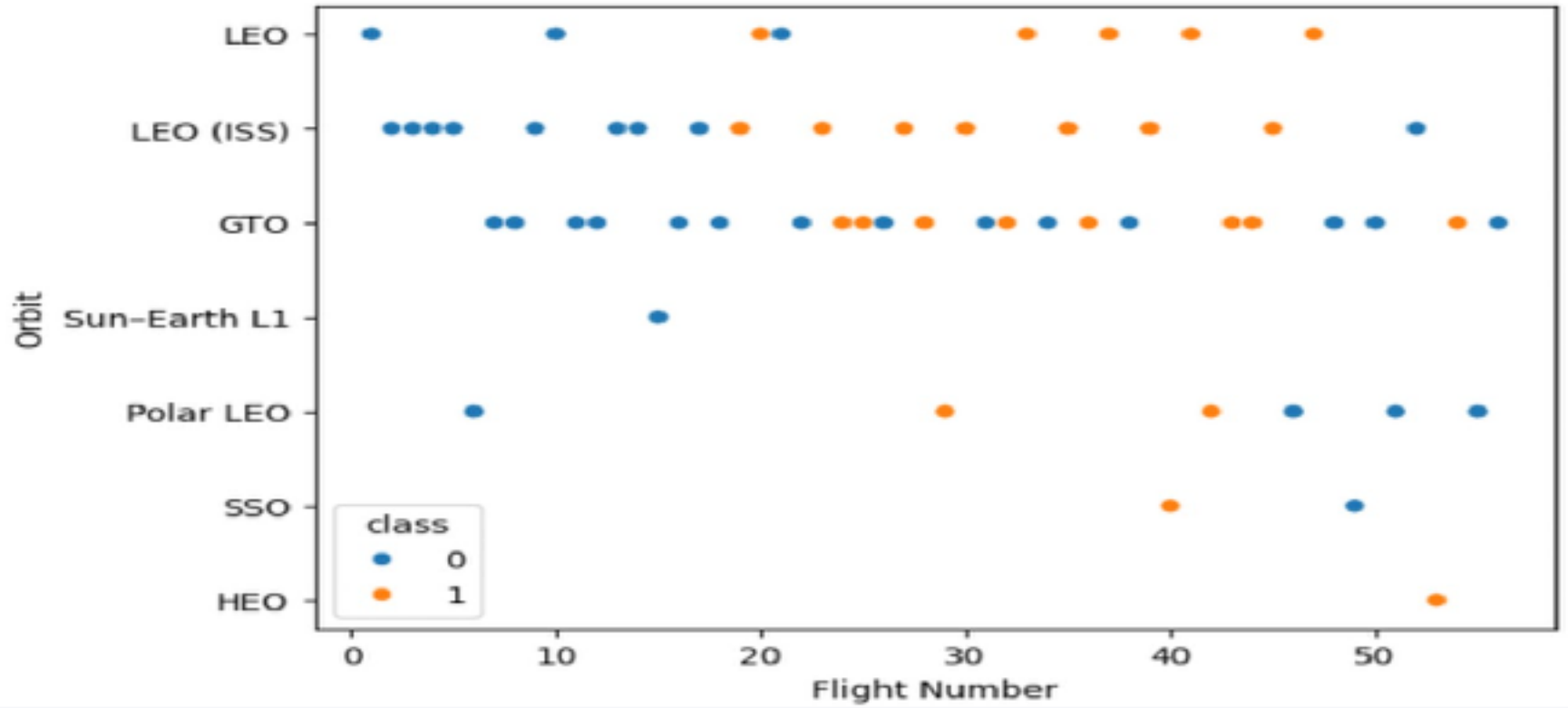
Success Rate vs. Orbit Type

Out of all the orbits ES-L1, GEO, HWO & SSO had the most successful orbits.



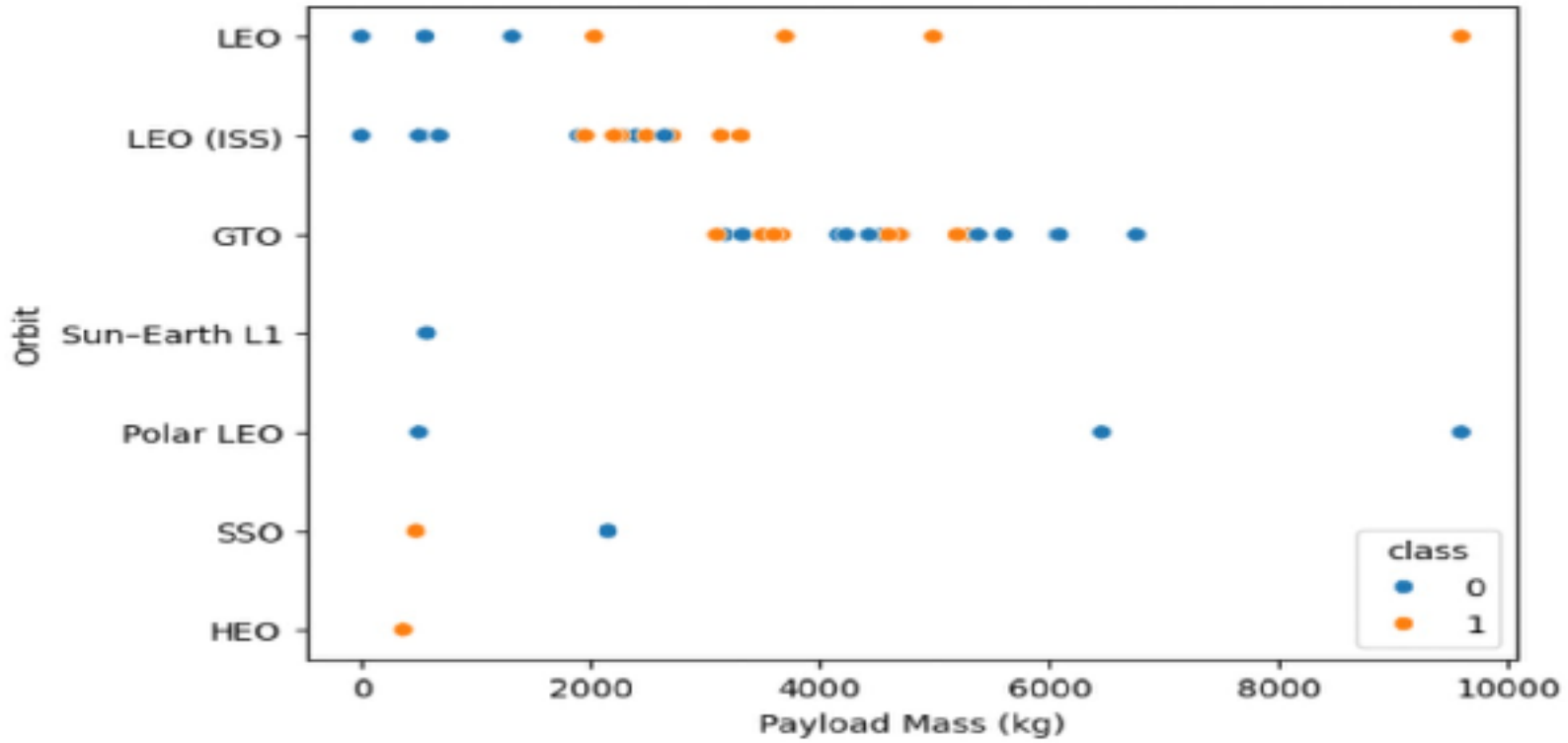
Flight Number vs. Orbit Type

- So we can show that the successful launches have been under the LEO (ISS) orbit type, with regards to the GTO Orbit type the scatterplot shows no corelation at all.



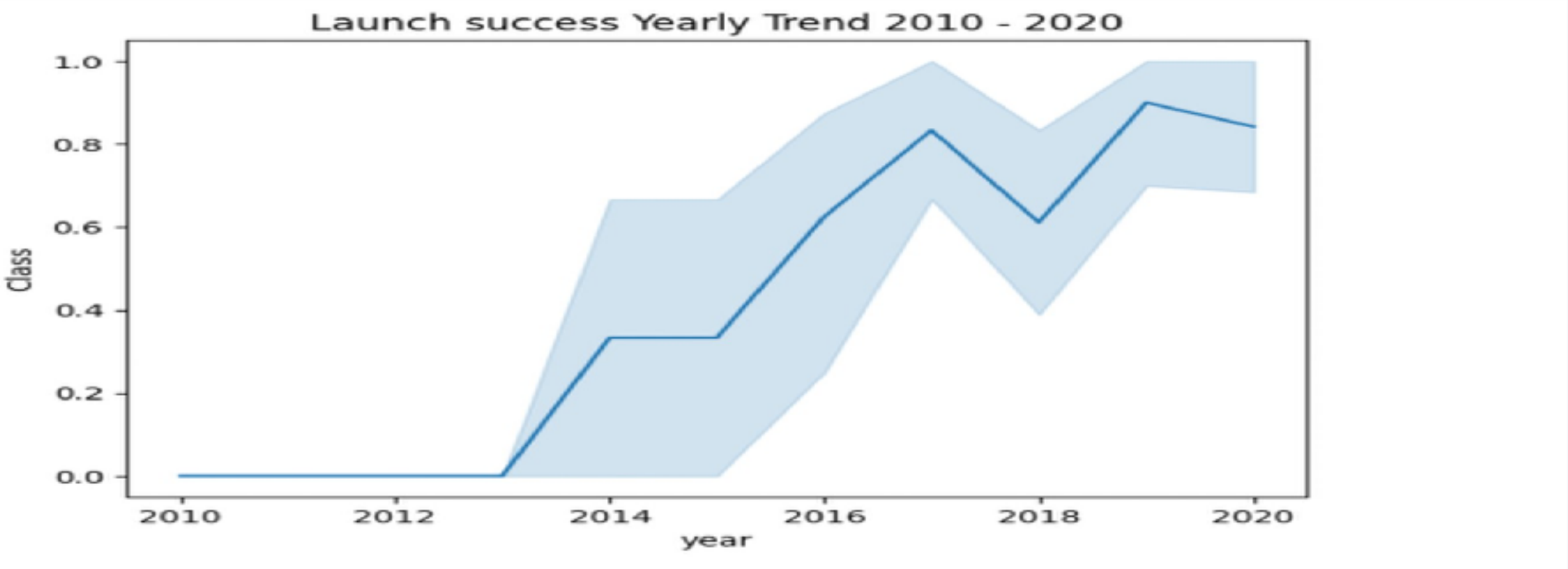
Payload vs. Orbit Type

- We notice looking at this scatter plot that the higher the payload the more successful the orbits are.



Launch Success Yearly Trend

- We can see in the Trend line graph below that the Falcon 9 rocket started mid point of 2012 to launch successfully with a small drop in 2016 and then recovered in 2018.



All Launch Site Names

In order to get the unique launch sites we used some sql code with the 'DISTINCT' sql command to only get the unique sites.

```
job2= '''
          select DISTINCT Launch_Site FROM SPACEXTABLE
'''
table = pd.read_sql_query(job2, con)
table
```

	Launch_Site
0	CCAFS LC-40
1	VAFB SLC-4E
2	KSC LC-39A
3	CCAFS SLC-40

Launch Site Names Begin with 'CCA'

To do the search in SQL to find 5 Launch Sites that start with 'CCA' in the name we used the conditional stmt 'WHERE' for the launch_site column and then applied the 'Like' 'CCA%' command the '%' is a wild card which says match any character after CCA, we then limit the results to 5 using the Limit command.

[25]:
```
job3= '''
        select * FROM SPACEXTABLE
        WHERE Launch_Site LIKE 'CCA%'
        LIMIt 5
'''
table = pd.read_sql_query(job3, con)
table
```

[25]:

	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
0	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	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)
2	2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	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

To work out the Total Payload Mass in SQL we use the SUM command on the column 'Payload_Mass_KG' column within the data making sure that we are only interested in NASA for the customer using the WHERE conditional SQL stmt.

```
job4= '''
        select SUM(PAYLOAD_MASS__KG_) as TotalMass FROM SPACEXTABLE
        WHERE Customer LIKE 'NASA (CRS)'

'''
table = pd.read_sql_query(job4, con)
table
```

	TotalMass
0	45596

Average Payload Mass by F9 v1.1

To get the average figure for the F9 rocket Payload Mass we used the 'AVG' stmt on the 'Payload_Mass__KG column making sure that we filtered on the Booster_Version. The result was 2928.4 for the average mass.

AVG Payload Mass in KG

```
27]: job5= '''
         select AVG(PAYLOAD_MASS__KG_) as AvgMass FROM SPACEXTABLE
         WHERE Booster_Version = 'F9 v1.1'

     '''
     table = pd.read_sql_query(job5, con)
     table
```

27]:

	AvgMass
0	2928.4

First Successful Ground Landing Date

To get the earliest 1st date of a successful ground landing we used the 'Min' stmts on the 'Date' column and filtered for 'Success (ground pad). The 1st Successful date was 2015-12-22.

```
]: job6= '''
          select MIN(Date) as [First successful date] FROM SPACEXTABLE
          WHERE Landing_Outcome Like 'Success (ground pad)'

   '''
   table = pd.read_sql_query(job6, con)
   table
```

]:

	First successful date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Below list the results of the Drone Ships that successfuly landed that carried an original payload mass of between 4,000 and 6,000 KG's.

```
]: job8= '''
         select Mission_Outcome, Booster_Version, Landing_Outcome FROM SPACEXTABLE
         WHERE Landing_Outcome Like 'Success (Drone Ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
'''
table = pd.read_sql_query(job8, con)
table
```

]:

	Mission_Outcome	Booster_Version	Landing_Outcome
0	Success	F9 FT B1022	Success (drone ship)
1	Success	F9 FT B1026	Success (drone ship)
2	Success	F9 FT B1021.2	Success (drone ship)
3	Success	F9 FT B1031.2	Success (drone ship)

Total Number of Successful and Failure Mission Outcomes

Successful & Un-Successful Missions

```
[28]: job6= '''
          select COUNT('Mission_Outcome') as [Successful Missions]  FROM SPACEXTABLE
          WHERE Mission_Outcome Like '%Success'

      '''
      table = pd.read_sql_query(job6, con)
      table

      job7= '''
          select COUNT('Mission_Outcome') as [Un-Successful Missions]  FROM SPACEXTABLE
          WHERE Mission_Outcome Like 'Failure%'

      '''
      table1 = pd.read_sql_query(job7, con)
      table1
```

[28]:

	Un-Successful Missions
0	1

```
[29]: job6= '''
          select COUNT('Mission_Outcome') as [Successful Missions]  FROM SPACEXTABLE
          WHERE Mission_Outcome Like '%Success'

      '''
      table = pd.read_sql_query(job6, con)
      table
```

[29]:

	Successful Missions
0	98

To the left shows 1st the Un-Successful missions of 1 and below that the successful missions of 98, filtering on the Mission outcome column for '%Success' or 'Failure %'.

Boosters Carried Maximum Payload

```
task9= '''
        SELECT Booster_Version, PAYLOAD_MASS__KG_
        FROM SPACEXTABLE

        WHERE PAYLOAD_MASS__KG_ = (
                                    select max( PAYLOAD_MASS__KG_ )
                                    FROM SPACEXTABLE
                                    )
        '''
table = pd.read_sql_query(task9, con)
table
```

	Booster_Version	PAYLOAD_MASS__KG_
0	F9 B5 B1048.4	15600
1	F9 B5 B1049.4	15600
2	F9 B5 B1051.3	15600
3	F9 B5 B1056.4	15600
4	F9 B5 B1048.5	15600
5	F9 B5 B1051.4	15600
6	F9 B5 B1049.5	15600
7	F9 B5 B1060.2	15600
8	F9 B5 B1058.3	15600
9	F9 B5 B1051.6	15600
10	F9 B5 B1060.3	15600
11	F9 B5 B1049.7	15600

This table lists all the booster_Versions with the Max Payload they carried.

2015 Launch Records

Below is the table showing the failed landings in the 2015 with the Booster_Versions and Launch sites indicated.

[41]:
```
job8= '''
        select Mission_Outcome, Launch_Site, Booster_Version, Landing_Outcome FROM SPACEXTABLE
        WHERE Landing_Outcome Like 'Failure (Drone Ship)' AND DATE BETWEEN '2015-01-01' AND '2015-12-31'
'''
table = pd.read_sql_query(job8, con)
table
```

[41]:

	Mission_Outcome	Launch_Site	Booster_Version	Landing_Outcome
0	Success	CCAFS LC-40	F9 v1.1 B1012	Failure (drone ship)
1	Success	CCAFS LC-40	F9 v1.1 B1015	Failure (drone ship)

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

```
job10 = '''
        SELECT Landing_Outcome, COUNT(Landing_Outcome)
        FROM SPACEXTABLE
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY Landing_Outcome
        ORDER BY count(landing_Outcome) DESC

'''
table = pd.read_sql_query(job10,con)
table
```

	Landing_Outcome	COUNT(Landing_Outcome)
0	No attempt	10
1	Success (drone ship)	5
2	Failure (drone ship)	5
3	Success (ground pad)	3
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Failure (parachute)	2
7	Precluded (drone ship)	1

The table (left) shows the number of catagories in landing outcomes between June 2010 and March 2017, we notice that 'No attempt' landing outcome was the highest occurances of 10.

We used the 'group by' stmt on Landing Outcomes to group all the outcomes together then 'ordered by' sql stmt in descending order.

Section 3

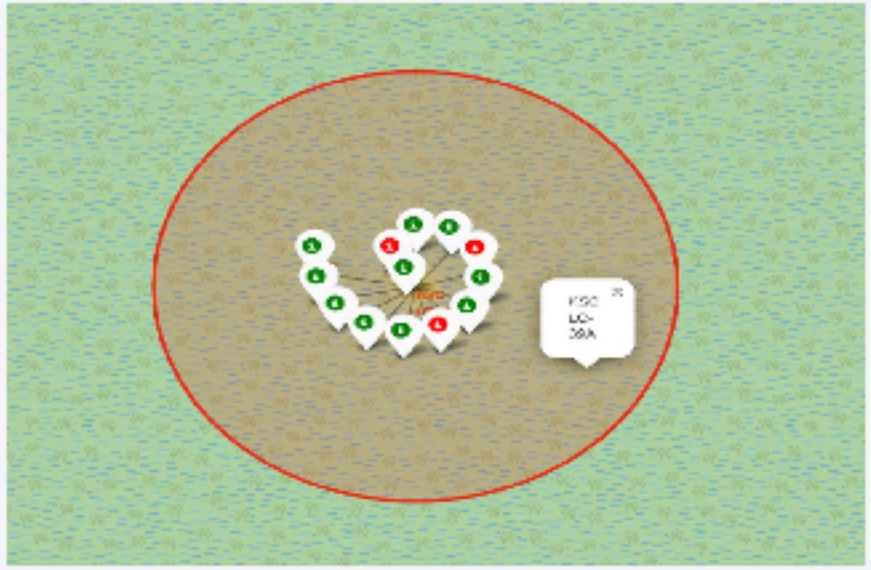
Launch Sites Proximities Analysis

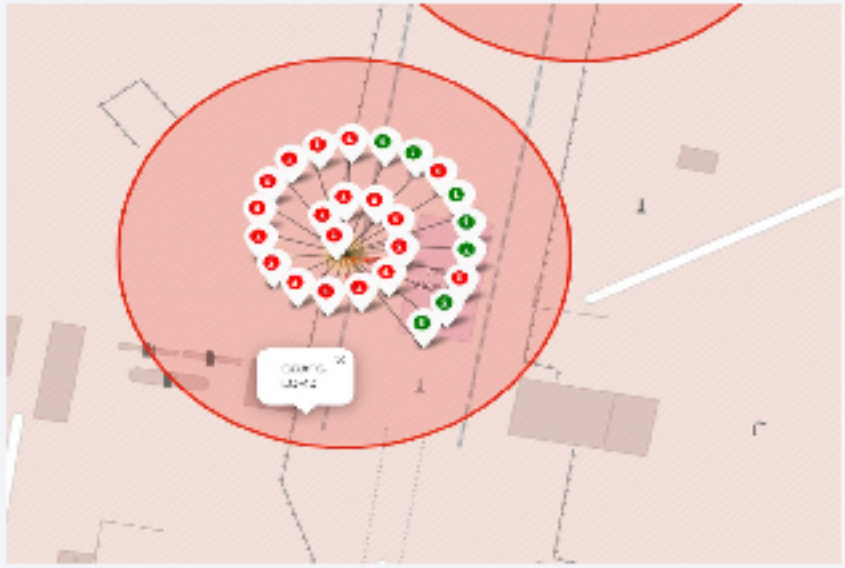
A Map of the SpaceX Launch sites

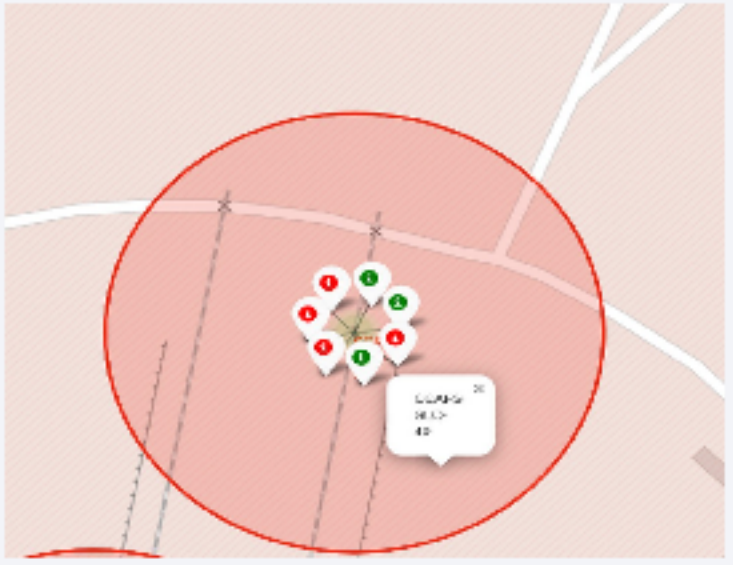
Successful & Un-successful SpaceX launches

The Picture to the left shows in Green the successful launches and Red the un-successful launches at the California site

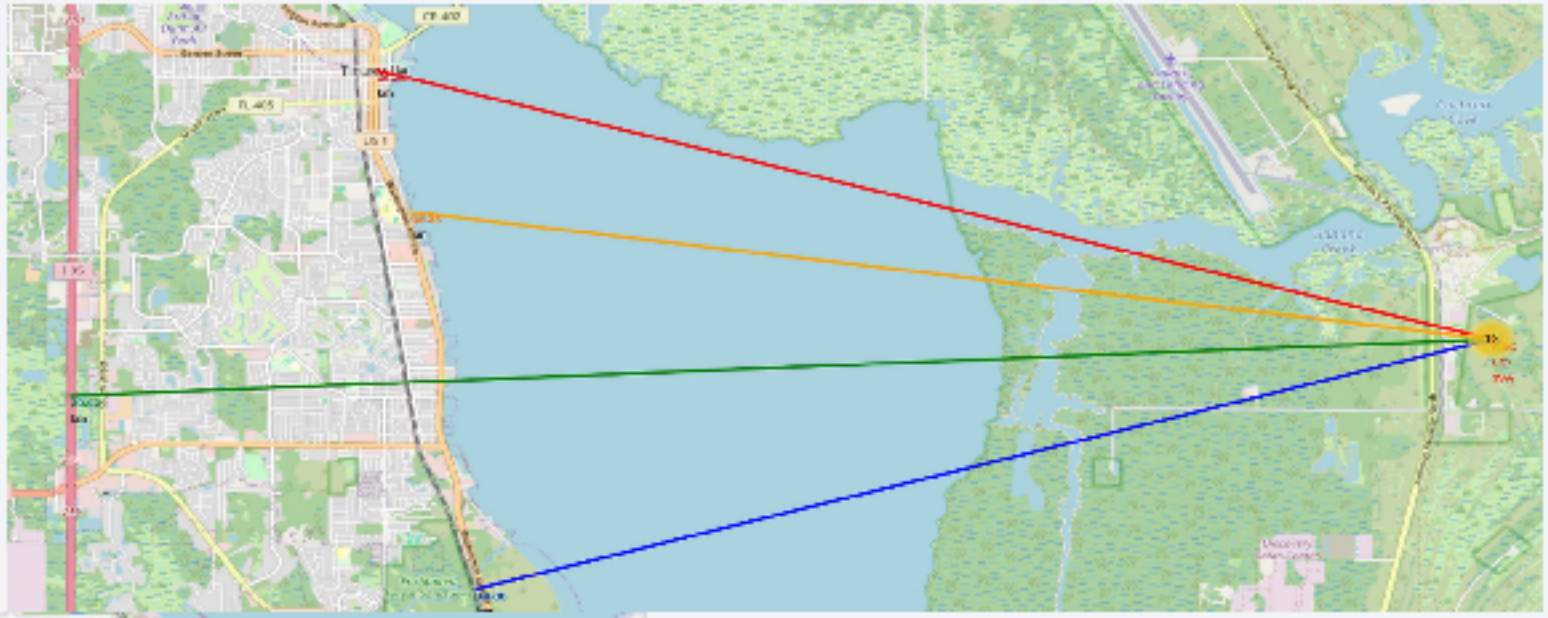
The 3 pictures below shows in Green the successful and Red the Un-Successful launches at the Florida sites.







Distance to Nearest City,Coastline,Highway and RailwayTrack from Florida Launcher



Map to left shows all the important sites in relation to the the florida launch site KSCLC-39A

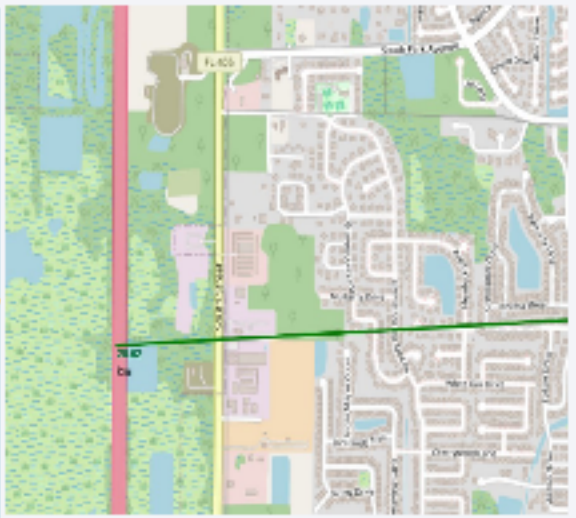
- Red line nearest city Trustville 16.29 Km's away

- Orange line coastline 15.35 Km's away

- Green line main Road 215 20.02 km's away

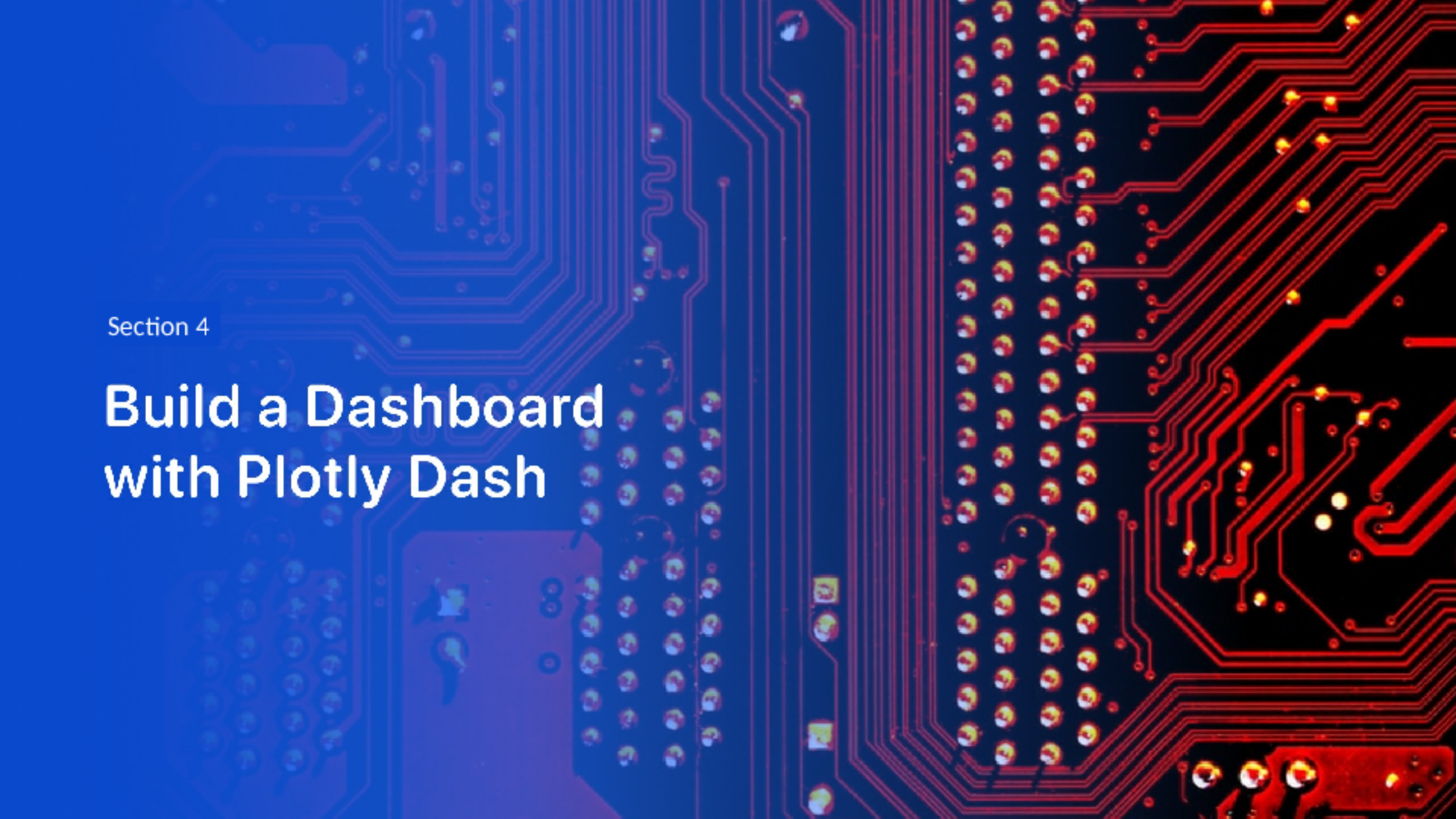
- Blue line Nearest Railway Line 14.98 km's away

Pictures below are closeups of the above.

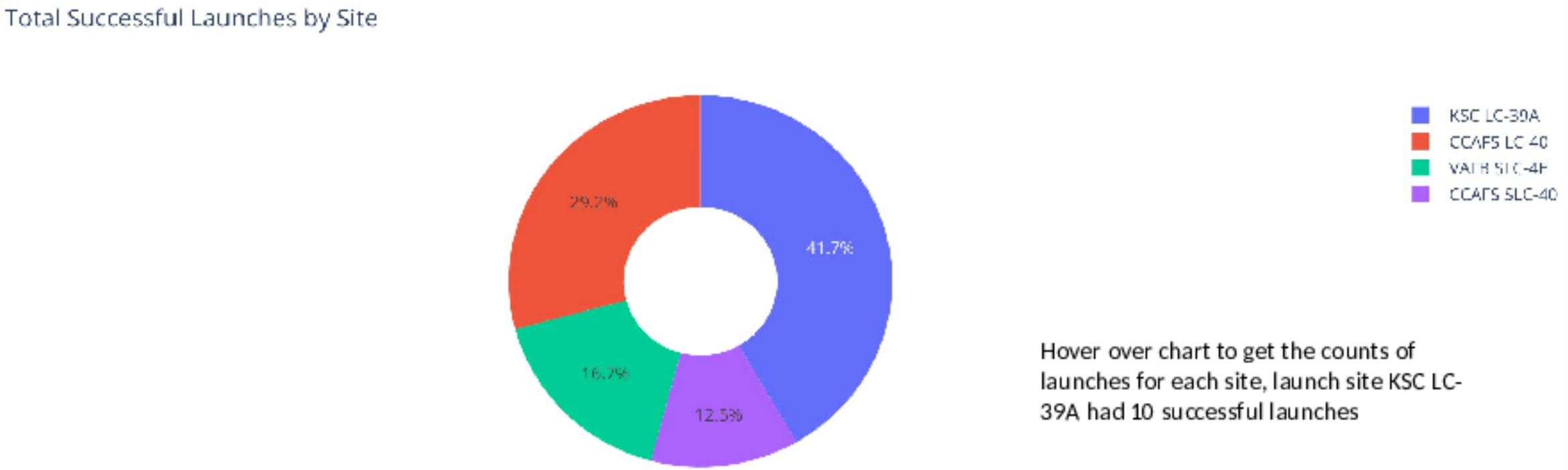


Section 4

Build a Dashboard with Plotly Dash



Pie Chart showing the % of successful Launches by site



We can see that Launch site KSC LC-39A had the highest number of Successful Launches

Pie chart showing the % in Successful -v- Un-successful Launches

SpaceX Launch Records Dashboard

All Sites

Total Success vs Failure Launches

There where slightly more failures then successful launches of the SpaceX rockets.

Scatter Plots showing frequency of successful -v- Un-Successful launches based on Payload

When looking at both the scatterplots the top one shows that we had more successful launches when the payload in the particular rocket is from 4,000kg's in weight and under

The 2nd scatter plot at the bottom shows that we had many more unsuccessful launches when the payload was 4,000kg and above.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

The best learning model turned out to be the DecisionTree model as shown below.

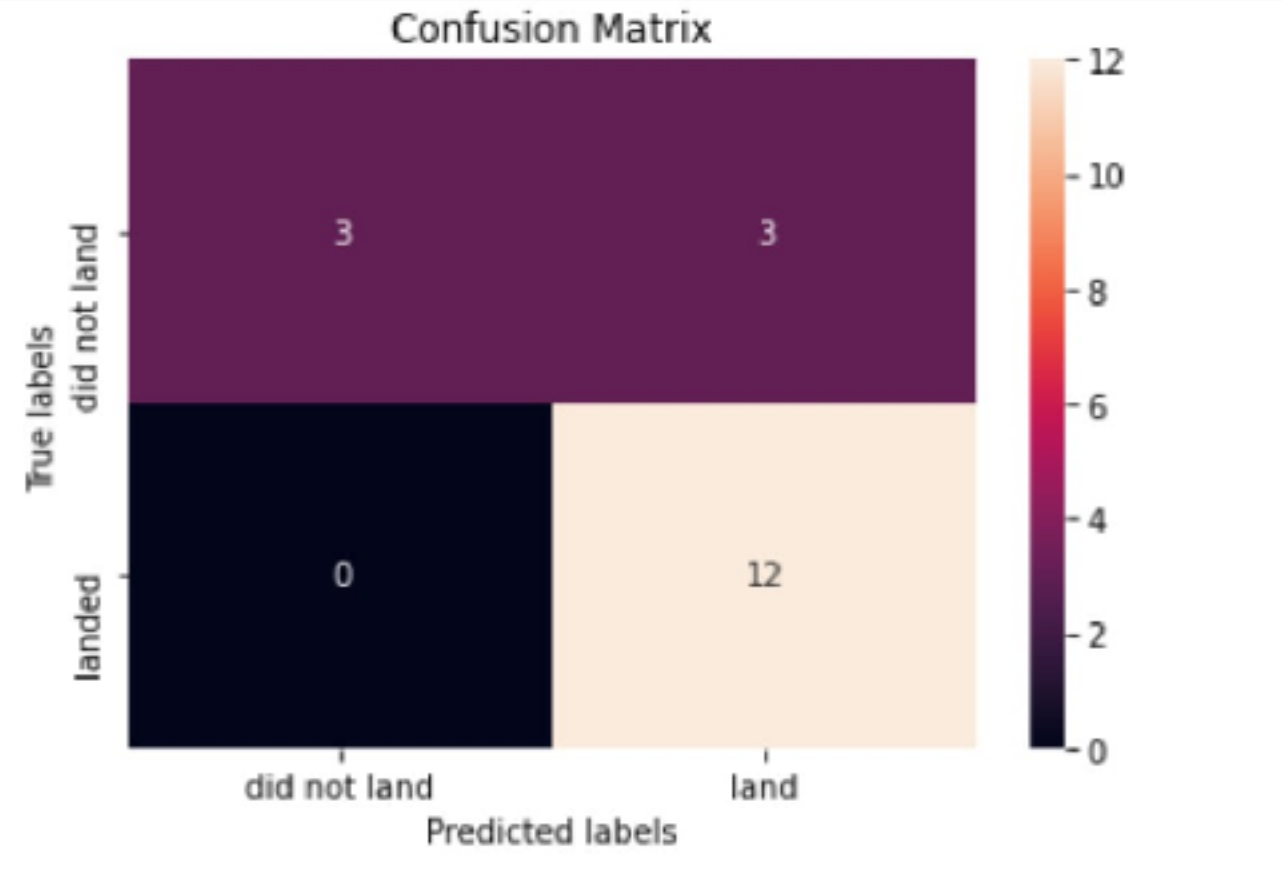
Find the method performs best:

```
models = {'KNeighbors':knn_cv.best_score_,
          'DecisionTree':tree_cv.best_score_,
          'LogisticRegression':logreg_cv.best_score_,
          'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix



We can see from the Confusion Matrix using the Descision Tree model that it can recognise the SpaceX rockets that landed correctly but the problem is with the those rockets that did not land but are recognised as landed of which there are 3 incorrectly identified as landed when they did not.

Conclusions

The 3 Launch Sites with the most successful launches where:- CCSFS KSC VAFB

2013 was the year when SpaceX started to successfully launch it's Falcon 9 rockets

When the payload was 4,000kg's in weight we noticed there where more successful launches then when the payload was 4,000kg's and above.

In Machine Learning the model that gave the best acurate prodictions in successful launches was the Deicision Tree model.

Appendix

Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!