IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Juan de Souza Ferreira
Out/06/2021

Outline

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

Executive Summary

- Data collection methodology
- Perform data wrangling
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification modelsSummary of all results

Introduction

- In this project, it was collected data from SpaceX API to create a dataset. With this dataset, we can extract some insights from graphic plots. We can also analyze and extract insights from Pandas DataFrames

- The focus of this project is create a Machine Learning model to predict whether a landing will be successful or not for future lauches, based on data collected from the Spacex API

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - The data was collected from SpaceX API, using requests library from Python.
    The Pandas library was used to convert JSON format (API response) to DataFrame
- Perform data wrangling
  - Describe how data was processed
  - To fill null numeric data, we used the mean of the feature in question. It was used a numpy and pandas library to do that.
- Perform exploratory data analysis (EDA) using visualization and SQL
  - We used a matplotlib library to explory and plot data. With SQL we extract some important information

Methodology

Executive Summary

- Perform interactive visual analytics using Folium and Plotly Dash
  - With Folium, we can extract some information about distance and map visualization. Plotly Dash provide us a interactive way to see plots and charts
- Perform predictive analysis using classification models
  - We used GridSearchCV to find the best parameters for Logistic Regression, SVM, Decision Tree, KNN models.

Data Collection

- Requests library was used to get data from SpaceX API.
  The SpaceX API returns a data in JSON format, this data was converted into a DataFrame using a Pandas library.

- BeautifulSoup (BS4) library was used to get data from Wikipedia.
  We parsed the HTML provided from the BS4 to Pandas DataFrame

Data Collection – SpaceX API

In [61]:
```
# Show the head of the dataframe
df.head()
```

Out[61]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Seri
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin1
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin3
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B00(

- Above we can see how data is stored into Pandas DataFrame. In Data Wrangling step, we'll choose some features that'll help us in the forward steps

- GitHub URL of Data Collection step: Applied_Data_Science_Capstone/Data_Collection_API.ipynb at main · Juanszf/Applied_Data_Science_Capstone (github.com)

Data Collection - Scraping

Out[14]:

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA (COTS)\nNRO	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA (COTS)	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA (CRS)	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA (CRS)	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10
...	...	...	...	...	...	...	...	...	...	...	...
116	117	CCSFS	Starlink	15,600 kg	LEO	SpaceX	Success\n	F9 B5B1051.10	Success	9 May 2021	06:42
117	118	KSC	Starlink	~14,000 kg	LEO	SpaceX Capella Space and Tyvak	Success\n	F9 B5B1058.8	Success	15 May 2021	22:56
118	119	CCSFS	Starlink	15,600 kg	LEO	SpaceX	Success\n	F9 B5B1063.2	Success	26 May 2021	18:59
119	120	KSC	SpaceX CRS-22	3,328 kg	LEO	NASA (CRS)	Success\n	F9 B5B1067.1	Success	3 June 2021	17:29
120	121	CCSFS	SXM-8	7,000 kg	GTO	Sirius XM	Success\n	F9 B5	Success	6 June 2021	04:26

121 rows × 11 columns

- Above we can see how data is stored into Pandas DataFrame. In Data Wrangling step, we'll choose some features that'll help us in the forward steps

- GitHub URL of Data Collection step: [Applied_Data_Science_Capstone/Web_Scrapping.ipynb at main · Juanszf/Applied_Data_Science_Capstone (github.com)]

Data Wrangling

```
In [121]: data_falcon9.isnull().sum()

Out[121]: FlightNumber       0
          Date               0
          BoosterVersion     0
          PayloadMass        5
          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
```

- As shown above, Payload Mass column has five null values. One way to solve that is replace the null values for the mean of the Payload Mass column
- We used replace method to replace the np.nan values for 6123.54 (mean of Payload Mass)

Data Wrangling

```
In [10]: for i,outcome in enumerate(landing_outcomes.keys()):
             print(i,outcome)

0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

- As shown above, the column “Landing Outcome” bring to us if the landing is successful or not
- We associated the false results to “Class 0” and the true results to “Class 1”
- The Class column was added to DataFrame, this will help us in the forwards steps
- GitHub URL of Data Collection step: [Applied_Data_Science_Capstone/Data_Wrangling.ipynb at main · Juanszf/Applied_Data_Science_Capstone (github.com)]

EDA with Data Visualization

Summary of charts were plotted in this project:

- Categorical Plot: It was used to visualize the relations of two features
- Bar Plot: It was used to visualize the rate of successfully landing per orbit
- Line Plot: It was used to visualize the increasing of successfully landing over the time

- GitHub URL of Data Collection step: Applied_Data_Science_Capstone/Data_Wrangling.ipynb at main · Juanszf/Applied_Data_Science_Capstone (github.com)

EDA with SQL

Summary of SQL queries we performed

- %%sql SELECT DISTINCT launch_site FROM SPACEXTBL
- %%sql   SELECT * FROM SPACEXTBL WHERE (lower(launch_site) LIKE 'cca%') LIMIT 5
- %%sql   SELECT sum(payload_mass__kg_) AS Total_Payload_Mass_NASA

FROM SPACEXTBL WHERE (customer LIKE 'NASA (CRS)')

- %%sql   SELECT avg(payload_mass__kg_) AS avg_payload_mass_f9_v1_1

FROM SPACEXTBL WHERE (booster_version LIKE 'F9 v1.1')

- %%sql   SELECT min(date) AS first_success FROM SPACEXTBL WHERE (landing__outcome LIKE 'Success (ground pad)')
- %%sql   SELECT booster_version FROM spacextbl WHERE (mission_outcome LIKE 'Success') AND (payload_mass__kg_ BETWEEN 4000 AND 6000)
- %%sql   SELECT mission_outcome, COUNT(*) AS total FROM spacextbl GROUP BY mission_outcome
- %%sql   SELECT booster_version AS booster_with_max_payload_mass FROM spacextbl WHERE payload_mass__kg_ IN (SELECT max(payload_mass__kg_) FROM SPACEXTBL)
- %%sql   SELECT booster_version, launch_site FROM spacextbl WHERE (landing__outcome LIKE 'Failure (drone ship)') AND (year(date) = 2015)
- %%sql   SELECT landing__outcome as landing_outcome, COUNT(*) AS total FROM spacextbl WHERE date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY landing__outcome ORDER BY total DESC

- Github URL: [Applied_Data_Science_Capstone/jupyter-labs-eda-sql-coursera.ipynb at main · Juanszf/Applied_Data_Science_Capstone (github.com)]

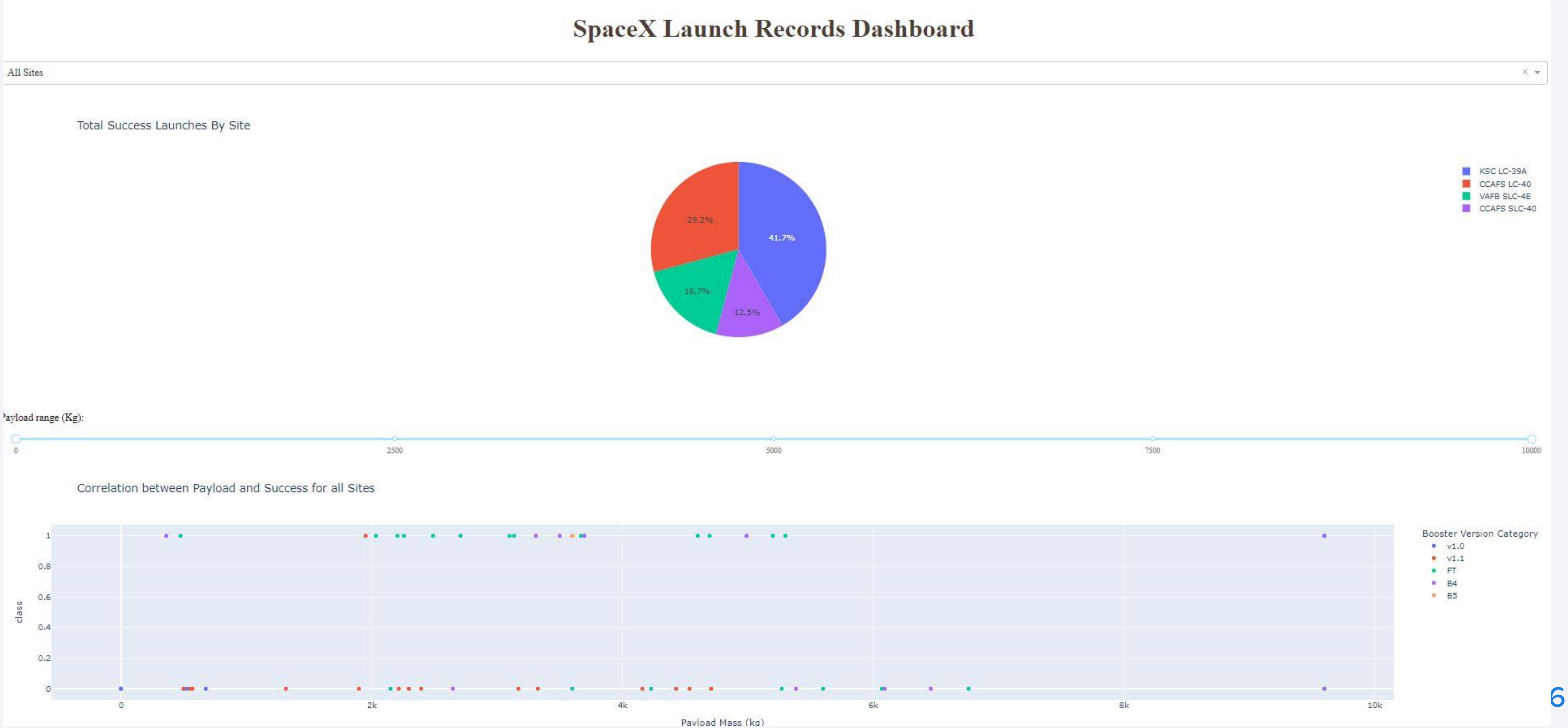
Build an Interactive Map with Folium

Summary of map objects we use in Folium:

- Circles: It was used to indicate the launch locations
- Lines: It was used to indicate the distance between the launch locations and nearest airport

- GitHub URL: Applied_Data_Science_Capstone/Interactive Visual Analytics with Folium lab.ipynb at main · Juanszf/Applied_Data_Science_Capstone (github.com)

Build a Dashboard with Plotly Dash



Predictive Analysis (Classification)

We build four models of machine learning:

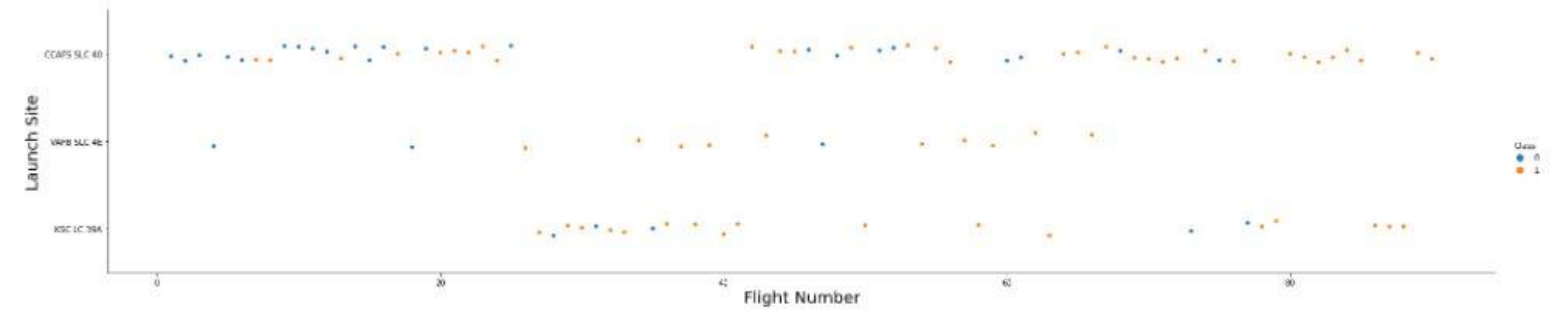
- Logistic Regression
- Support Vector Machine (SVM)
- Decision Tree
- K-nearest neighbors
- GridSearchCV was used to find the best parameters for each machine learning model

- GitHub URL: Applied_Data_Science_Capstone/Machine Learning Prediction.ipynb at main · Juanszf/Applied_Data_Science_Capstone (github.com)

Section 2

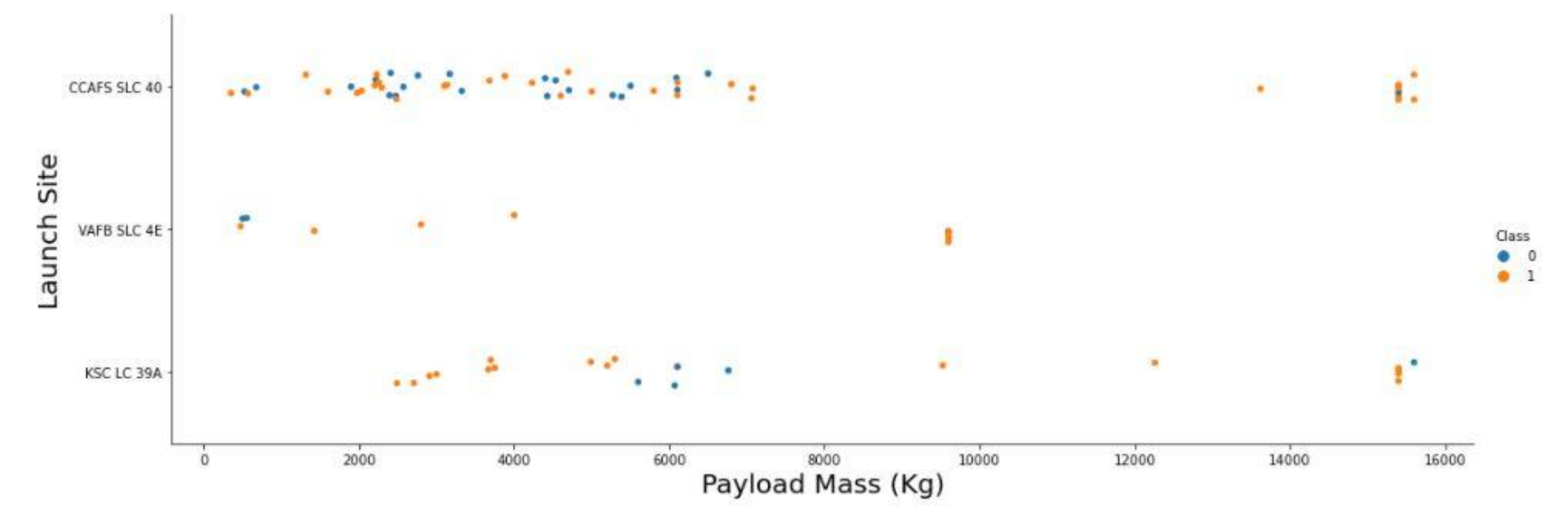
Insights drawn from EDA

Flight Number vs. Launch Site



This plot shows an increasing of successful landing over the flight number

Payload vs. Launch Site



- **CCAFS SLC40 Analysis**

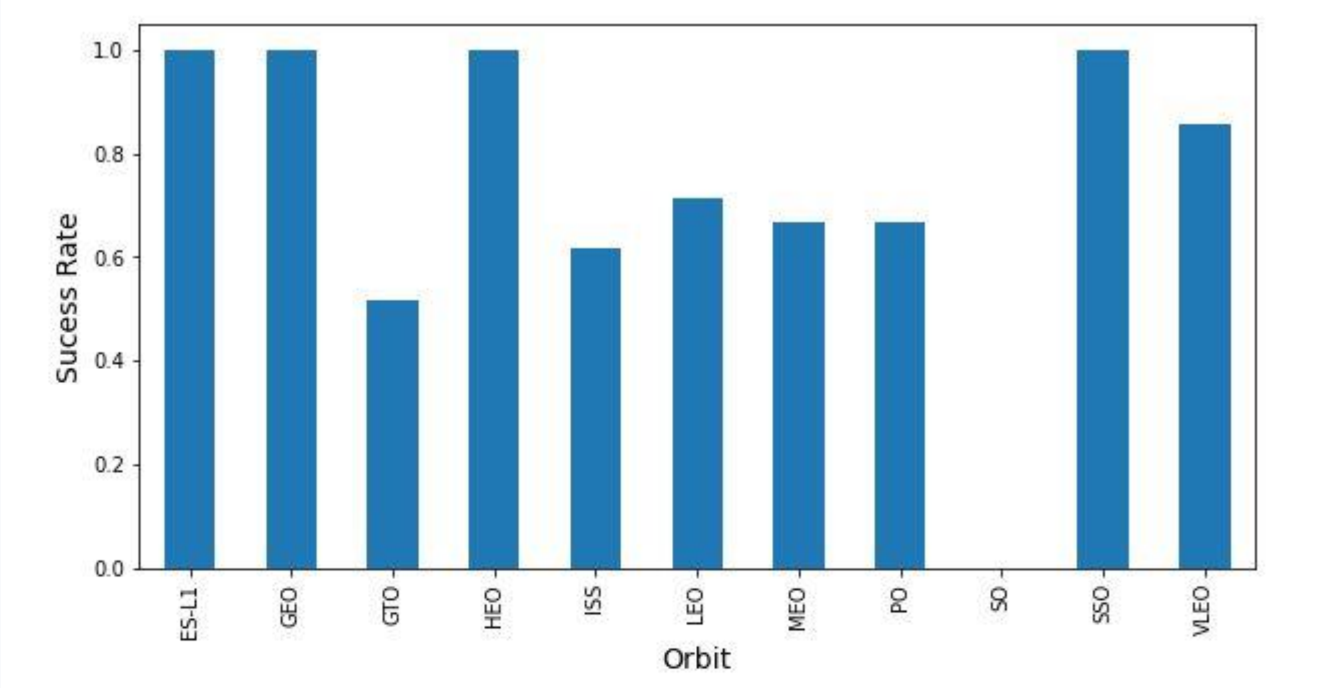
  CCAFS SLC 40 has more Launches then VAFB SLC 4E and KSC LC39A

  Most Flights of CCAFS SLC 40 has the Payload Mass between 0 and 8000 (kg)
- **VAFB SLC 4E**

  Is often used for a specific Payload Mass, due a crowding points near 10000 (kg)
- **KSC LC 39A**

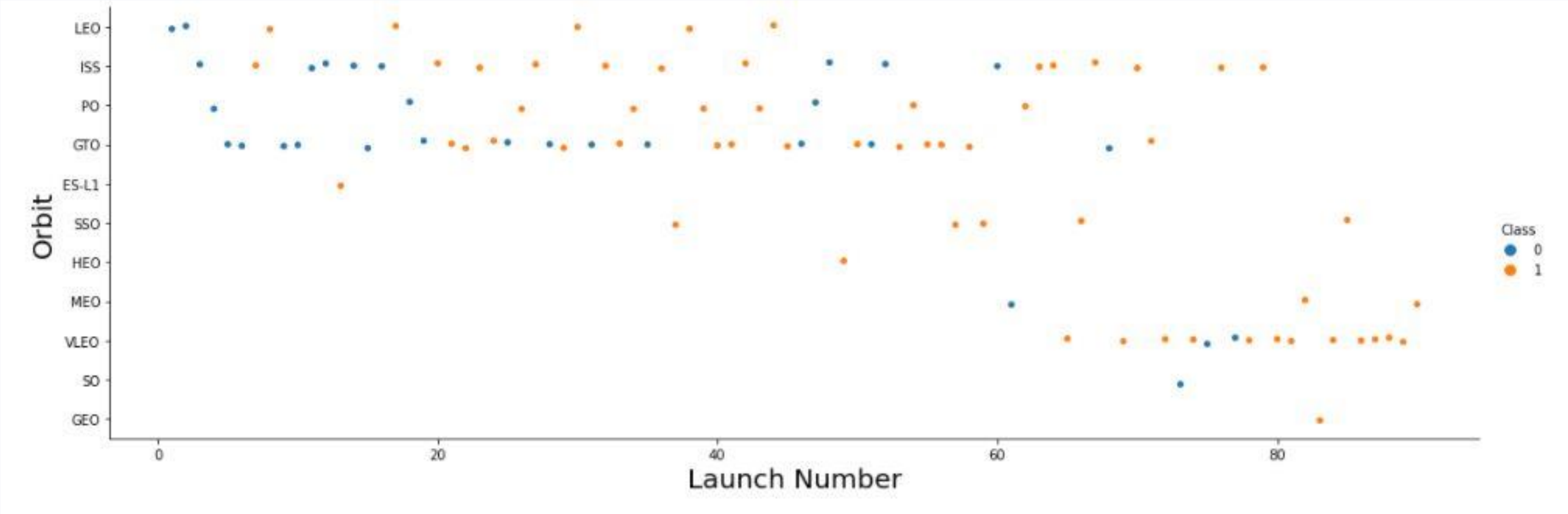
  Is the less used Launch Site, but has a interesting rate of flight success

Success Rate vs. Orbit Type



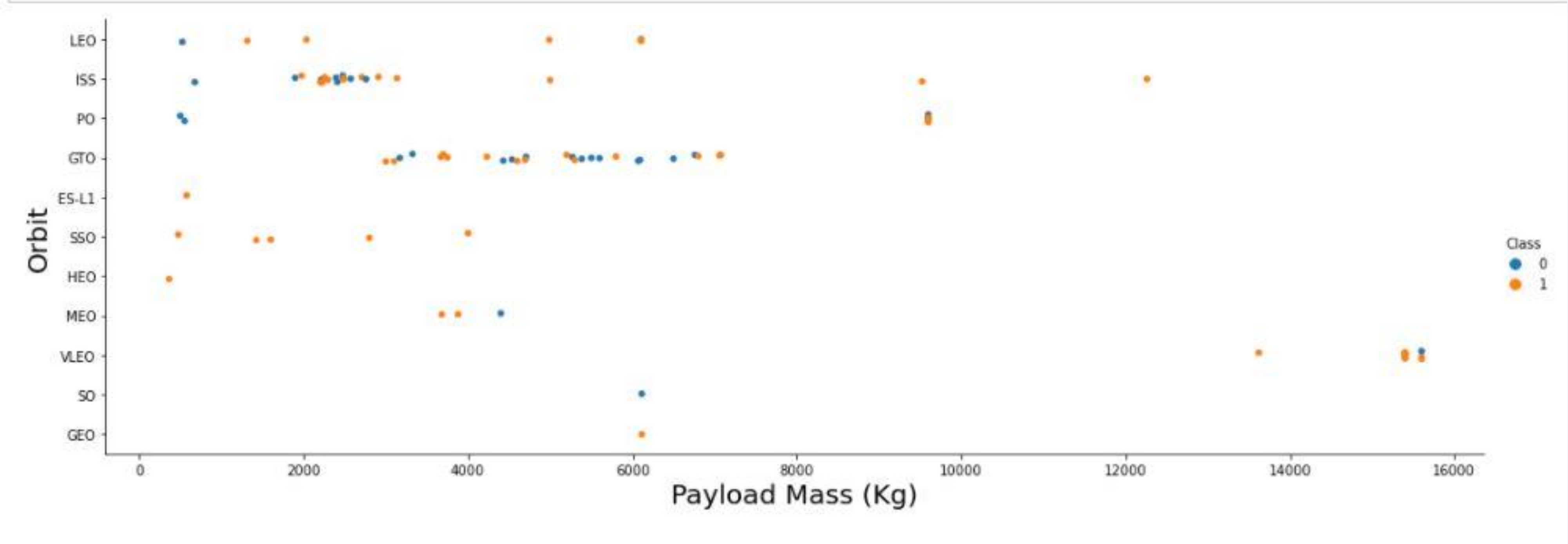
ES-L1, GEO, HEO, SSO have the highest sucess rate

Flight Number vs. Orbit Type

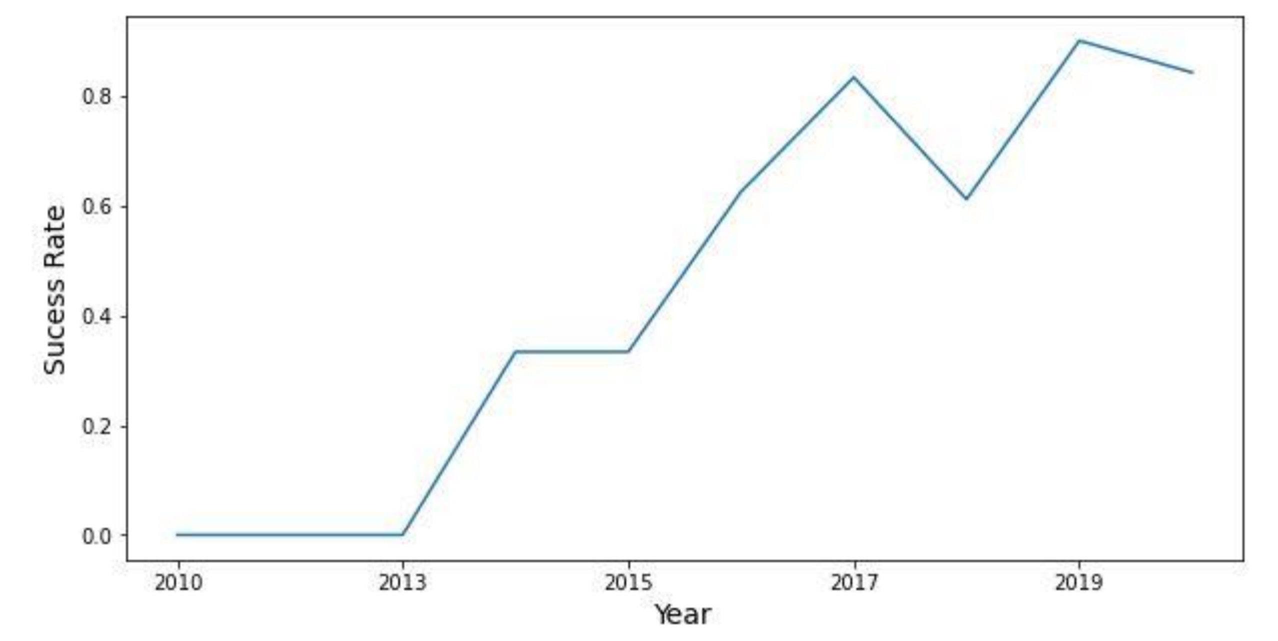


ES-L1, GEO, HEO, SSO have the highest sucess rate, but they have a fewer launches

Payload vs. Orbit Type



Launch Success Yearly Trend



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

All Launch Site Names

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

- %%sql SELECT DISTINCT launch_site FROM SPACEXTBL

With this query we displayed the name of all launch site names

Launch Site Names Begin with 'CCA'

```
In [16]: %%sql   SELECT *
                 FROM SPACEXTBL
                 WHERE (lower(launch_site) LIKE 'cca%')
                 LIMIT 5

 * ibm_db_sa://qjs22631:***@mba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:31321/bludb
Done.
```

Out[16]:

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

```
In [20]: %%sql   SELECT sum(payload_mass__kg_) AS Total_Payload_Mass_NASA
                 FROM SPACEXTBL
                 WHERE (customer LIKE 'NASA (CRS)')

 * ibm_db_sa://qjs22631:***@mba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:31321/bludb
Done.
```

Out[20]:

total_payload_mass_nasa
45596

Average Payload Mass by F9 v1.1

```
In [22]: %%sql   SELECT avg(payload_mass__kg_) AS avg_payload_mass_f9_v1_1
                 FROM SPACEXTBL
                 WHERE (booster_version LIKE 'F9 v1.1')

 * ibm_db_sa://qjs22631:***@mba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:31321/bludb
Done.
```

Out[22]:

avg_payload_mass_f9_v1_1
2928

First Successful Ground Landing Date

```
In [25]: %%sql   SELECT min(date) AS first_success
                 FROM SPACEXTBL
                 WHERE (landing__outcome LIKE 'Success (ground pad)')

 * ibm_db_sa://qjs22631:***@mba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:31321/bludb
Done.

Out[25]:
```

first_success
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

In [28]:
```
%%sql   SELECT booster_version
        FROM spacextbl
        WHERE (mission_outcome LIKE 'Success') AND (payload_mass__kg_ BETWEEN 4000 AND 6000)
```

```
 * ibm_db_sa://qjs22631:***@mba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:31321/bludb
Done.
```

Out[28]:

booster_version
F9 v1.1
F9 v1.1 B1011
F9 v1.1 B1014
F9 v1.1 B1016
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1030
F9 FT B1021.2
F9 FT B1032.1
F9 B4 B1040.1
F9 FT B1031.2
F9 FT B1032.2
F9 B4 B1040.2
F9 B5 B1046.2
F9 B5 B1047.2
F9 B5 B1046.3
F9 B5B1054
F9 B5 B1048.3
F9 B5 B1051.2
F9 B5B1060.1
F9 B5 B1058.2
F9 B5B1062.1

Total Number of Successful and Failure Mission Outcomes

```
In [29]: %%sql   SELECT mission_outcome, COUNT(*) AS total
                 FROM spacextbl
                 GROUP BY mission_outcome
```

```
 * ibm_db_sa://qjs22631:***@mba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:31321/bludb
Done.
```

Out[29]:

mission_outcome	total
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

```
In [30]: %%sql   SELECT booster_version AS booster_with_max_payload_mass
                 FROM spacextbl
                 WHERE payload_mass__kg_ IN (SELECT max(payload_mass__kg_) FROM SPACEXTBL)

 * ibm_db_sa://qjs22631:***@mba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:31321/bludb
Done.
```

Out[30]:

booster_with_max_payload_mass
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

```
In [31]: %%sql   SELECT booster_version, launch_site
                 FROM spacextbl
                 WHERE (landing__outcome LIKE 'Failure (drone ship)') AND (year(date) = 2015)

 * ibm_db_sa://qjs22631:***@mba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:31321/bludb
Done.
```

Out[31]:

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

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

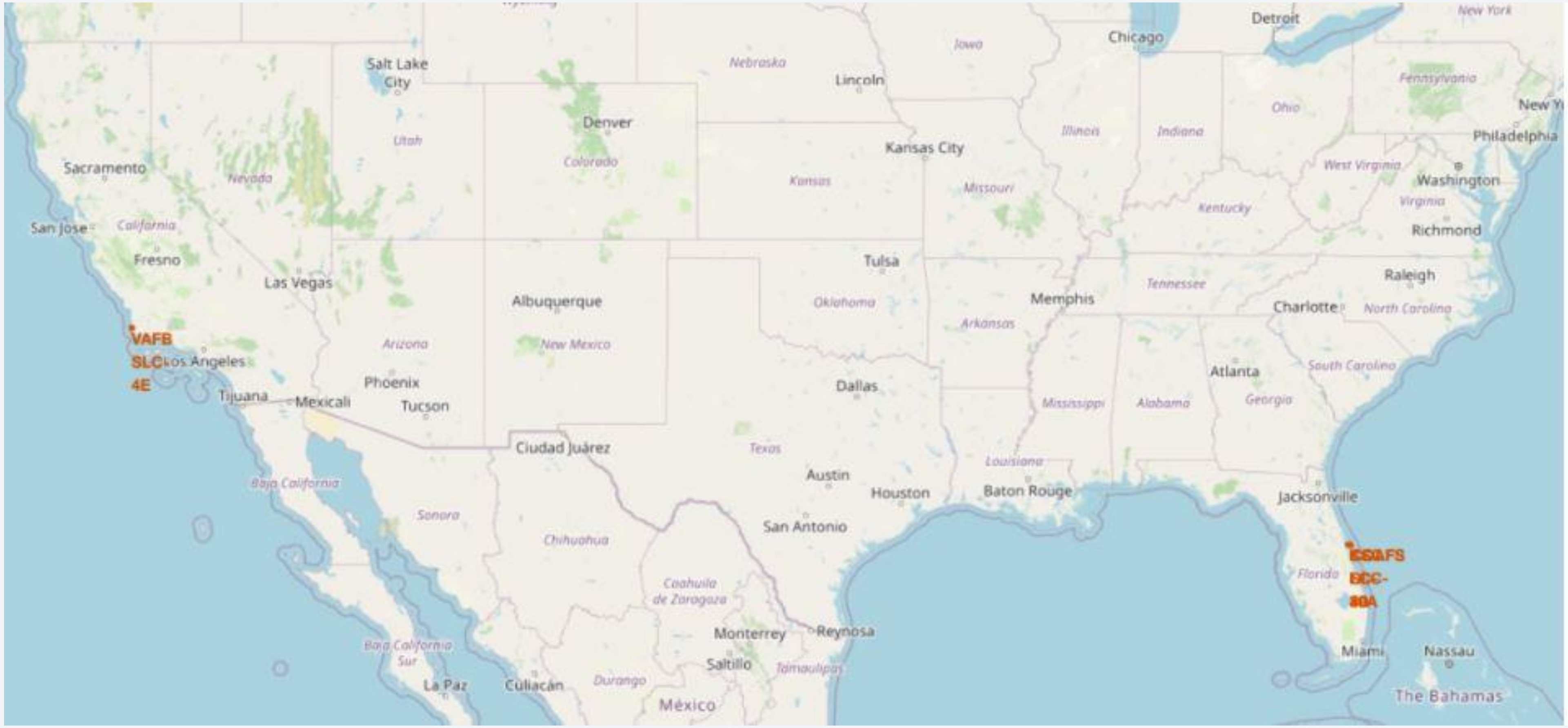
```
In [33]: %%sql   SELECT landing__outcome as landing_outcome, COUNT(*) AS total
                 FROM spacextbl
                 WHERE date BETWEEN '2010-06-04' AND '2017-03-20'
                 GROUP BY landing__outcome
                 ORDER BY total DESC

          * ibm_db_sa://qjs22631:***@mba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:31321/bludb
         Done.
Out[33]:
```

landing_outcome	total
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 4

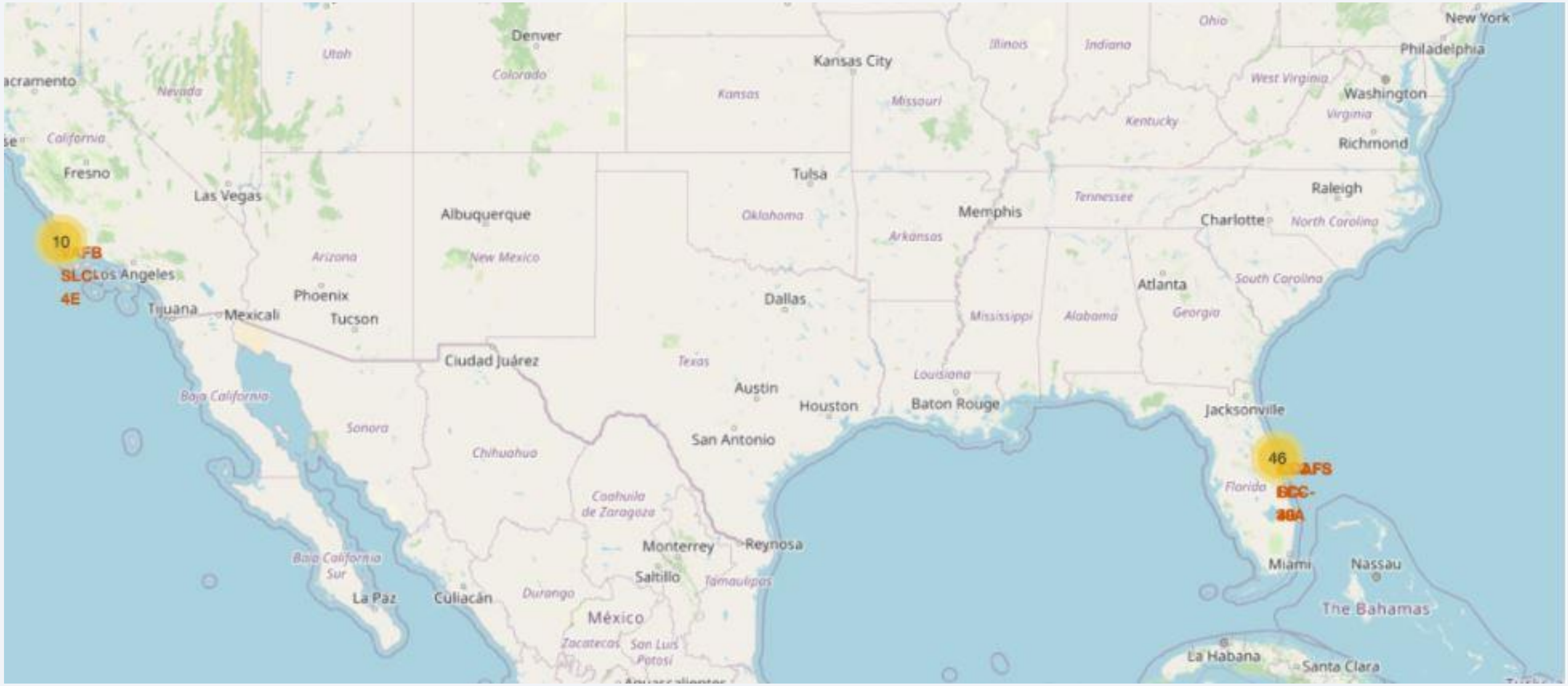
Launch Sites Proximities Analysis

Map view of launch sites

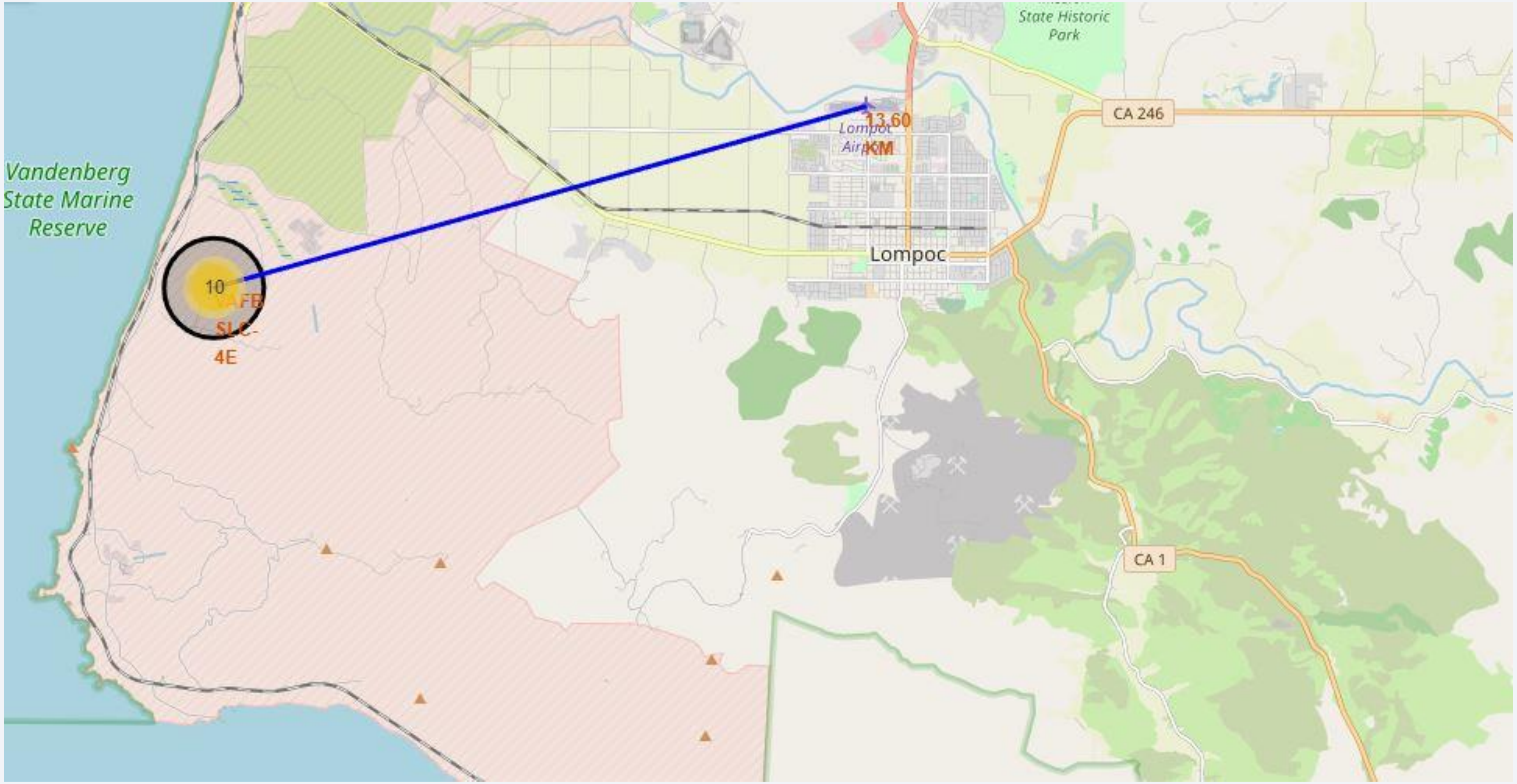


Pins of launch site locations

Map with launch outcomes distribuition



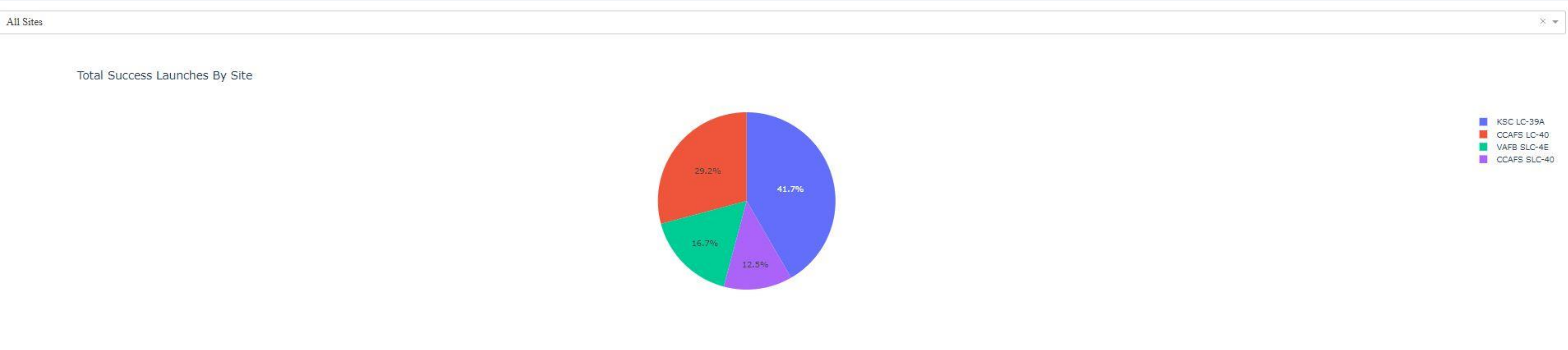
Map with the distance to nearest city airport



Section 5

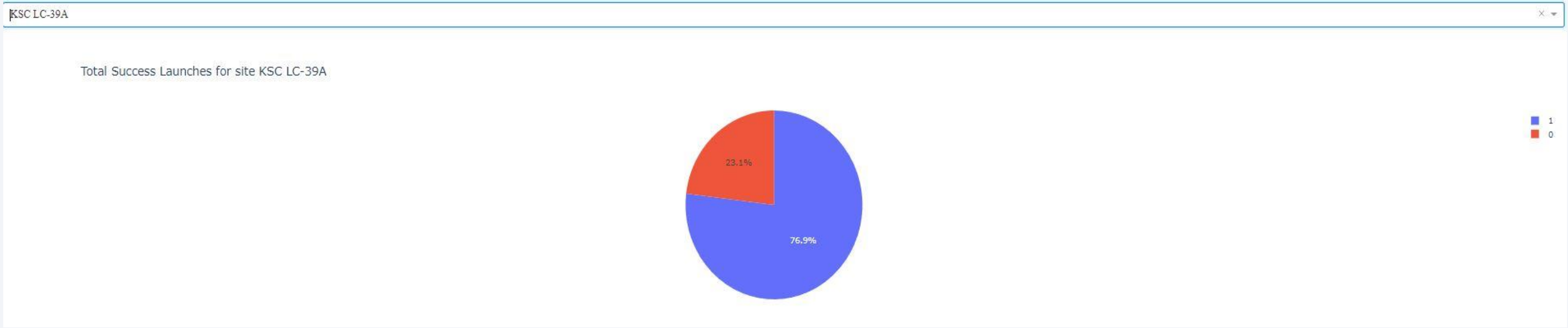
Build a Dashboard with Plotly Dash

Launch Success by Launch Sites

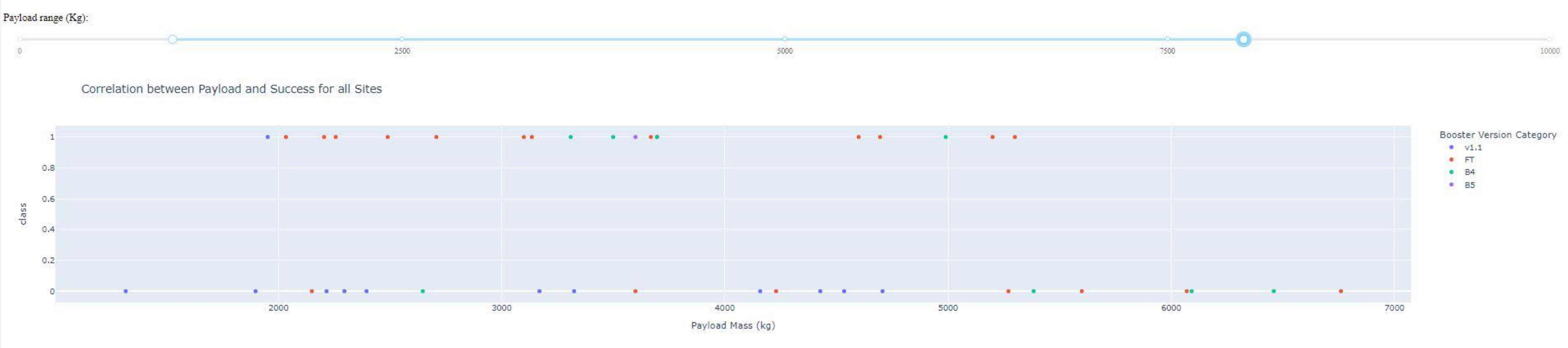


KSC LC-39A have the highest success rate

KSC LC-39A LAUNCHES



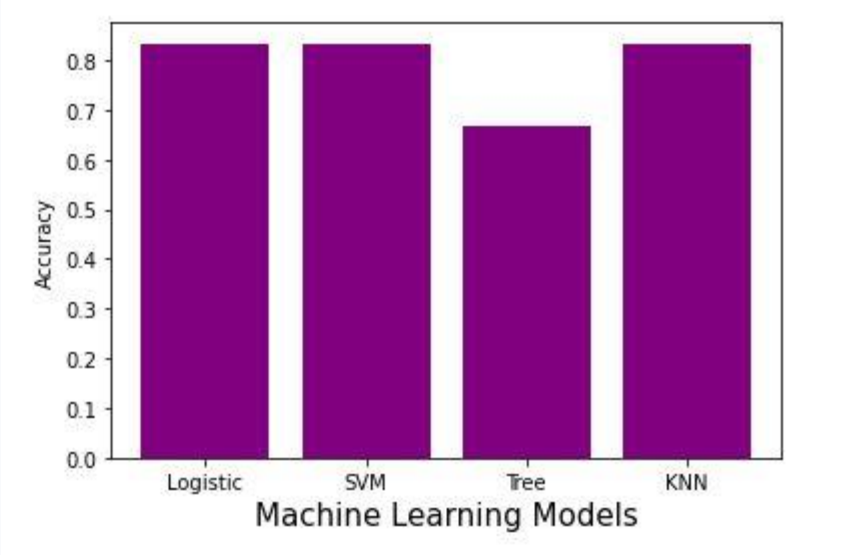
Payload Mass and Success correlation



Section 6

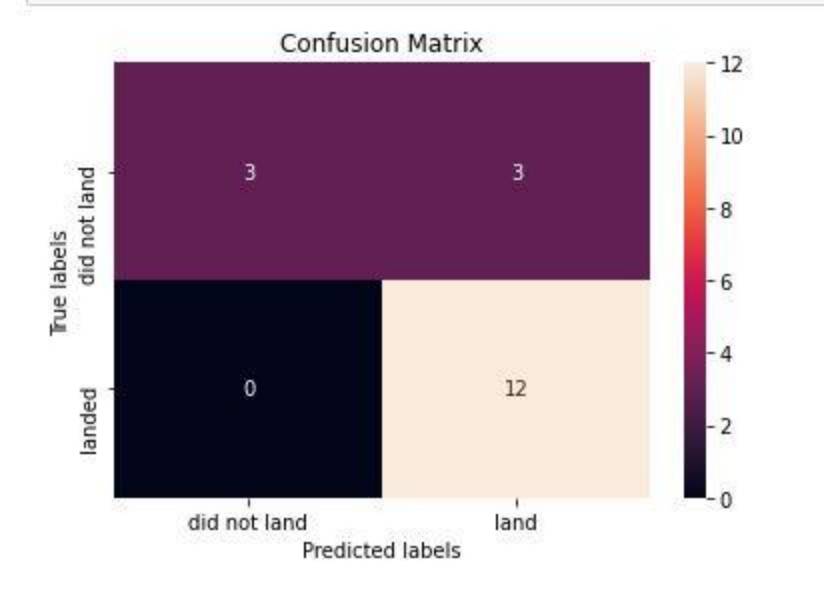
Predictive Analysis (Classification)

Classification Accuracy



- Logistic Regression accuracy: 0,83
- SVM accuracy: 0,83
- Decision Tree: 0,66
- KNN accuracy: 0,83

Confusion Matrix



The confusion matrix shows us that the model predicts successful landings well, but We see that the major problem is false positives.

Conclusions

- All machine learning models have high accuracy, we can use any
- We have an improvement in the successful landing rate over time, which shows an upward trend in the success rate
- The launch locations have a safety distance from cities and roads
- We can predict 80% of successful landing, so, we can determine the coast of 80% of launches

Thank you!