IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Phong Hao Pham
11/11/2021

Outline

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

Executive Summary

- Summary of methodologies

  Applying the basic in 10 previous courses: Making a main question, Approaching Problems, Collect data, Analysis, Visualization, Modeling, Report

- Summary of all results:

  - ➢Store data on database and query
  - ➢Analyze the dataset
  - ➢Classify the target with many Machine Learning models and choose the best model.
  - ➢Built a basic dashboard to track the Launch Sites.

Introduction

- 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.

- Problems

  ➢ Determine if the first stage will land

  ➢ Determine the cost of a launch

Section 1

Methodology

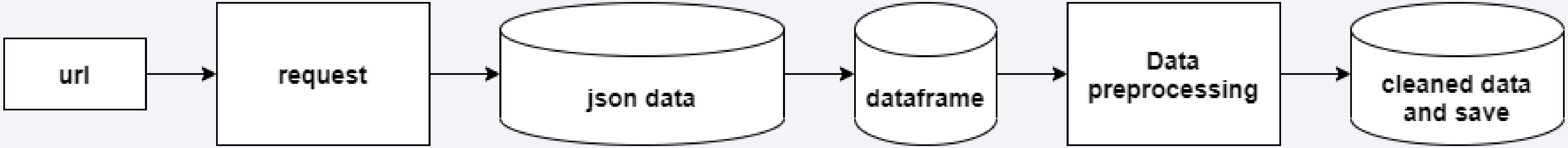
Methodology

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 models

Data Collection

- How data sets were collected
  - ➢Request to the SpaceX API
  - ➢Clean the requested data
- Flowcharts



Data Collection – SpaceX API

- Data collection with SpaceX REST

  - Step 1: Request get url
  - Step 2: Get content of request
  - Step 3: Convert content of request to dict (json format)
  - Step 4: Convert dict to dataframe in Pandas Python.

- Github URL:

https://github.com/WindPham/Cousera/blob/master/IBM_Data_Science/_10_Applied_Data_Science_Capstone/week_01/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

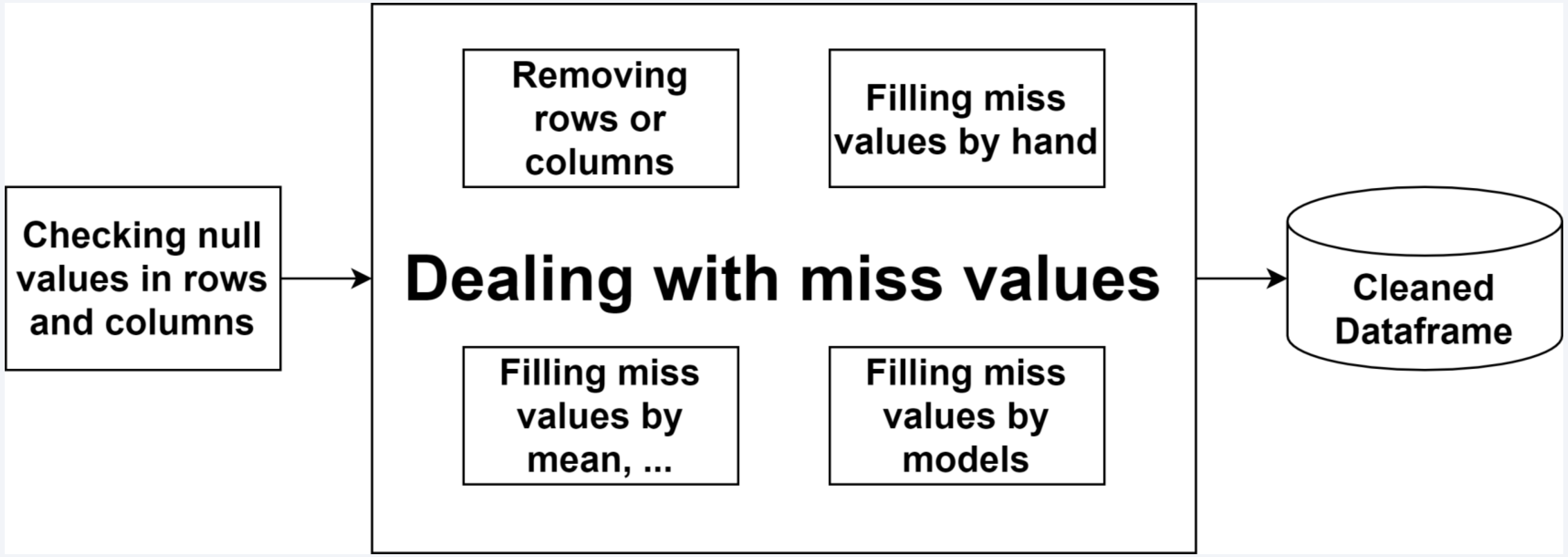
- Flowcharts

- GitHub URL:

https://github.com/WindPham/Cousera/blob/master/IBM_Data_Science/_10_Applied_Data_Science_Capstone/week_01/jupyter-labs-webscraping.ipynb

Data Wrangling

- Flowcharts:



- GitHub URL:

https://github.com/WindPham/Cousera/blob/master/IBM_Data_Science/_10_Applied_Data_Science_Capstone/week_01/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Charts: Scatter chart, bar chart, line chart
- Explanation:

  We need display the correlations of some continuous variables and class, so Scatter plot is suitable because it can illustrate the colors of class and the correlation of variables. Moreover, bar chart supports for performance between value domain of Launch Site and mean of class. Additionally, line chart display the probability of Class during many years.

- GitHub URL:

https://github.com/WindPham/Cousera/blob/master/IBM_Data_Science/_10_Applied_Data_Science_Capstone/week_02/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- Using sqlite3 instead of db2
- Explanation:
  - ➢Sqlite3 is so easy to install, code.
  - ➢Using sqlite3 is more convenient than db2 because it doesn't need account in Watson Studio and read port in code, …
  - ➢Using sqlite3 is quicker than db2 because it can create a virtual database in Jupyter Notebook environment. We can work with this database easily if dataset is small.
- GitHub URL:

https://github.com/WindPham/Cousera/blob/master/IBM_Data_Science/_10_Applied_Data_Science_Capstone/week_02/jupyter-labs-eda-sql-coursera-sqlite3-python.ipynb

Build an Interactive Map with Folium

- Map objects: MarkerCluster, Circle, Marker, MousePosition, PolyLine
- Explanation:
  - ➢ MarkerCluster is used for creating many markers for each discrete value (location) in a columns.
  - ➢ Circle is used for circling/highlighting a location with a big red circle .
  - ➢ PolyLine is used for connect two or more location with calculated weights.
- GitHub URL:

https://github.com/WindPham/Cousera/blob/master/IBM_Data_Science/_10_Applied_Data_Science_Capstone/week_03/lab_jupyter_launch_site_location.ipynb

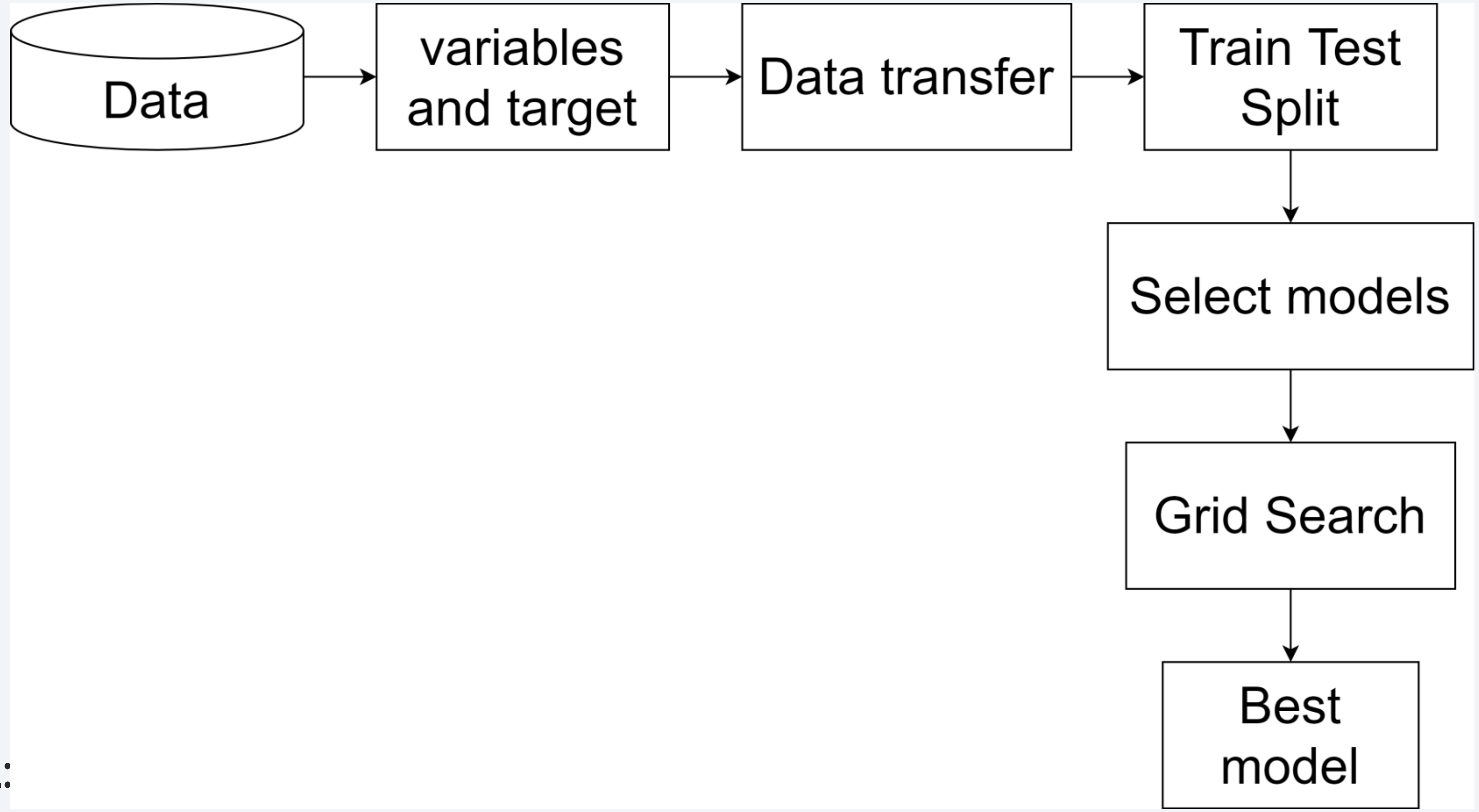
Dashboard with Plotly Dash for SpaceX

- Charts: Pie chart, Slider, Scatter chart
- Explanation:
  - ➢The launch site attribute is category, so pie chart is the most reasonable in this case.
  - ➢The payload mass is continuous value, we need perform it with the binary attribute as "class" on a range set before, so the scatter plot is a suitable selection for this case.
- GitHub URL:

https://github.com/WindPham/Cousera/blob/master/IBM_Data_Science/_10_Applied_Data_Science_Capstone/week_03/dash_interactivity.py

Predictive Analysis (Classification)

- Flowchart



- GitHub URL:

https://github.com/WindPham/Cousera/blob/master/IBM_Data_Science/_10_Applied_Data_Science_Capstone/week_04/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

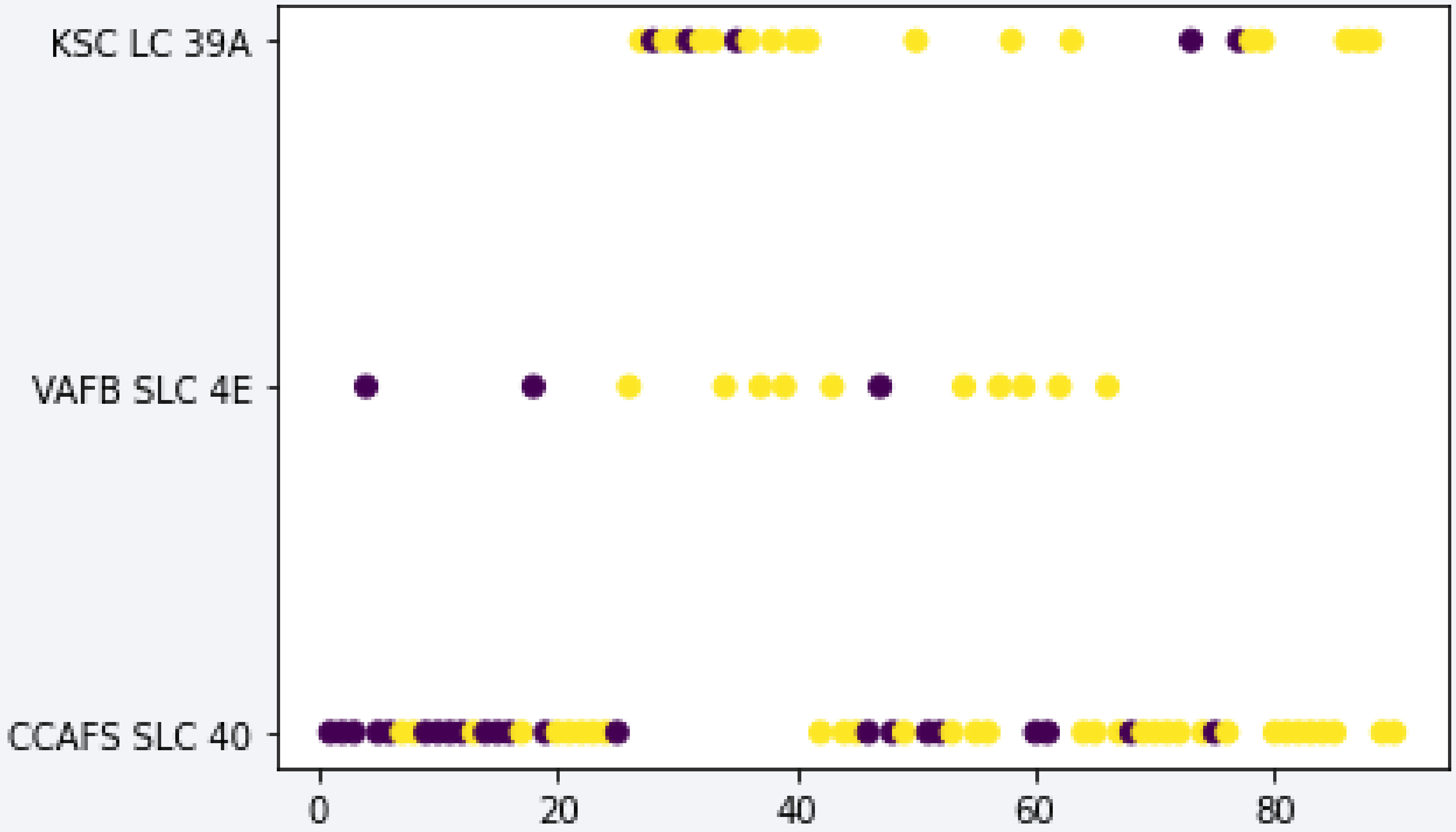
- Exploratory data analysis
- Analytics demo dataset in Dashboard
- Predictive analysis results by modeling

Section 2

Insights drawn from EDA

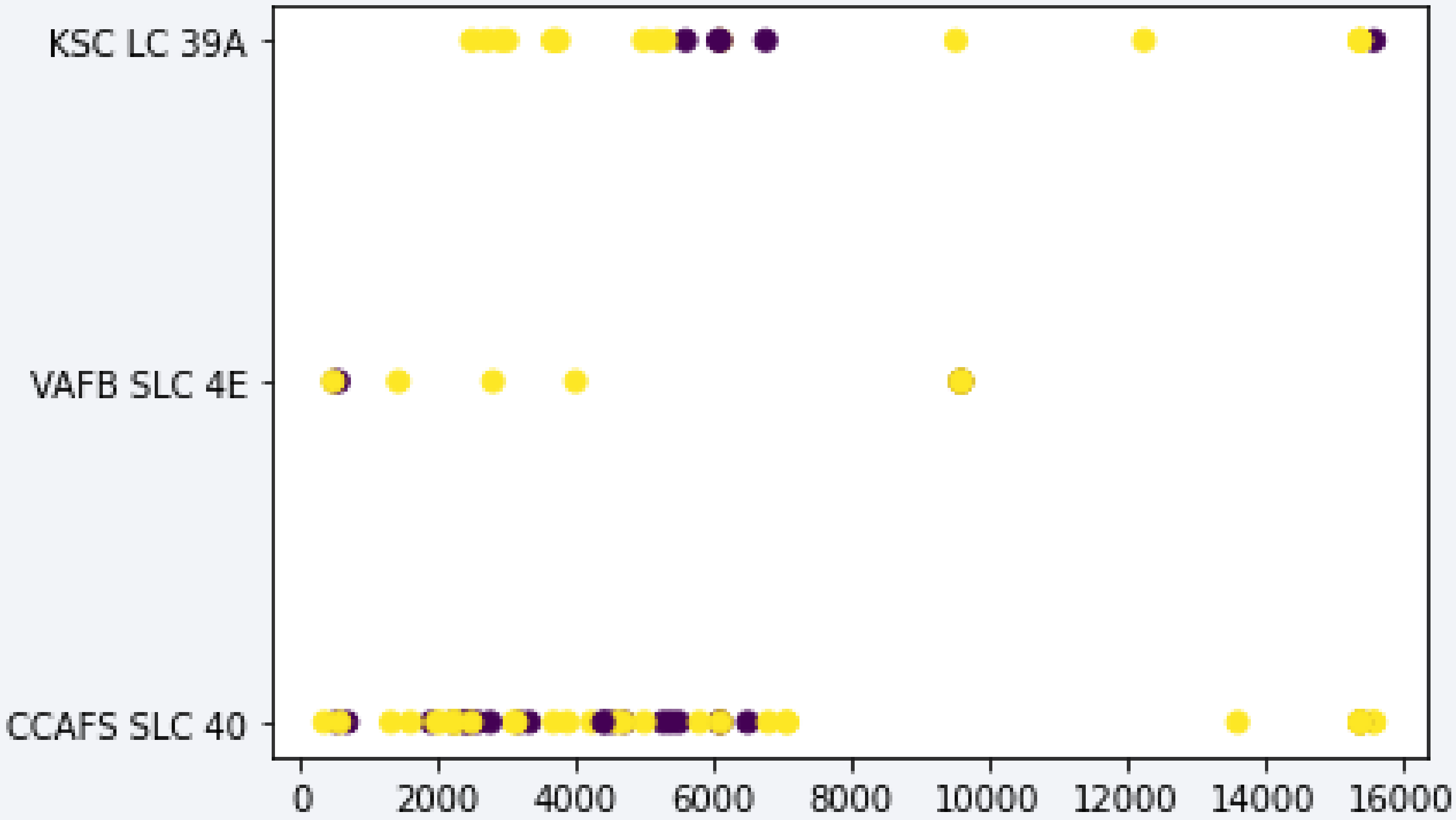
Flight Number vs. Launch Site

- A scatter plot of Flight Number vs. Launch Site



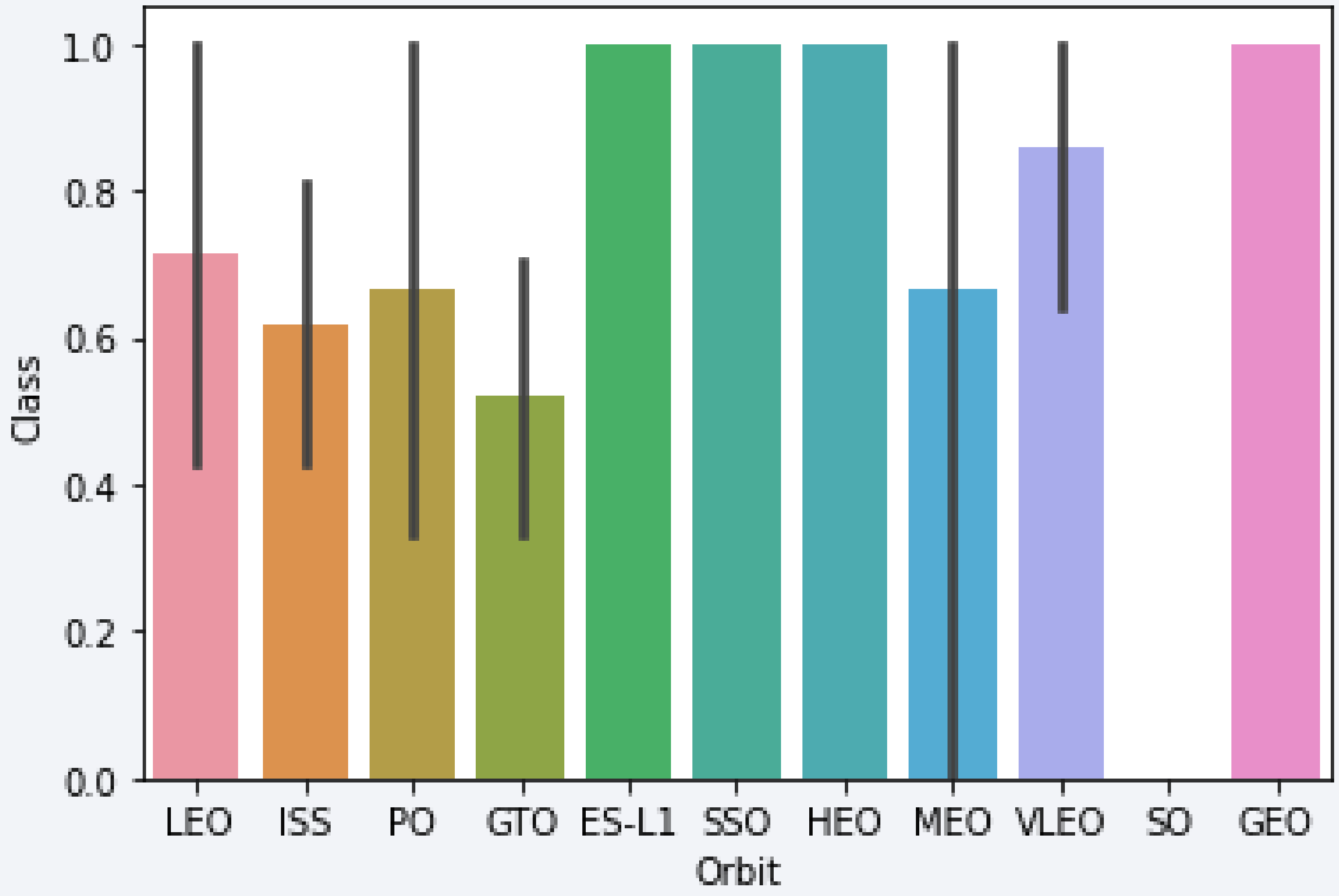
Payload vs. Launch Site

- A scatter plot of Payload vs. Launch Site



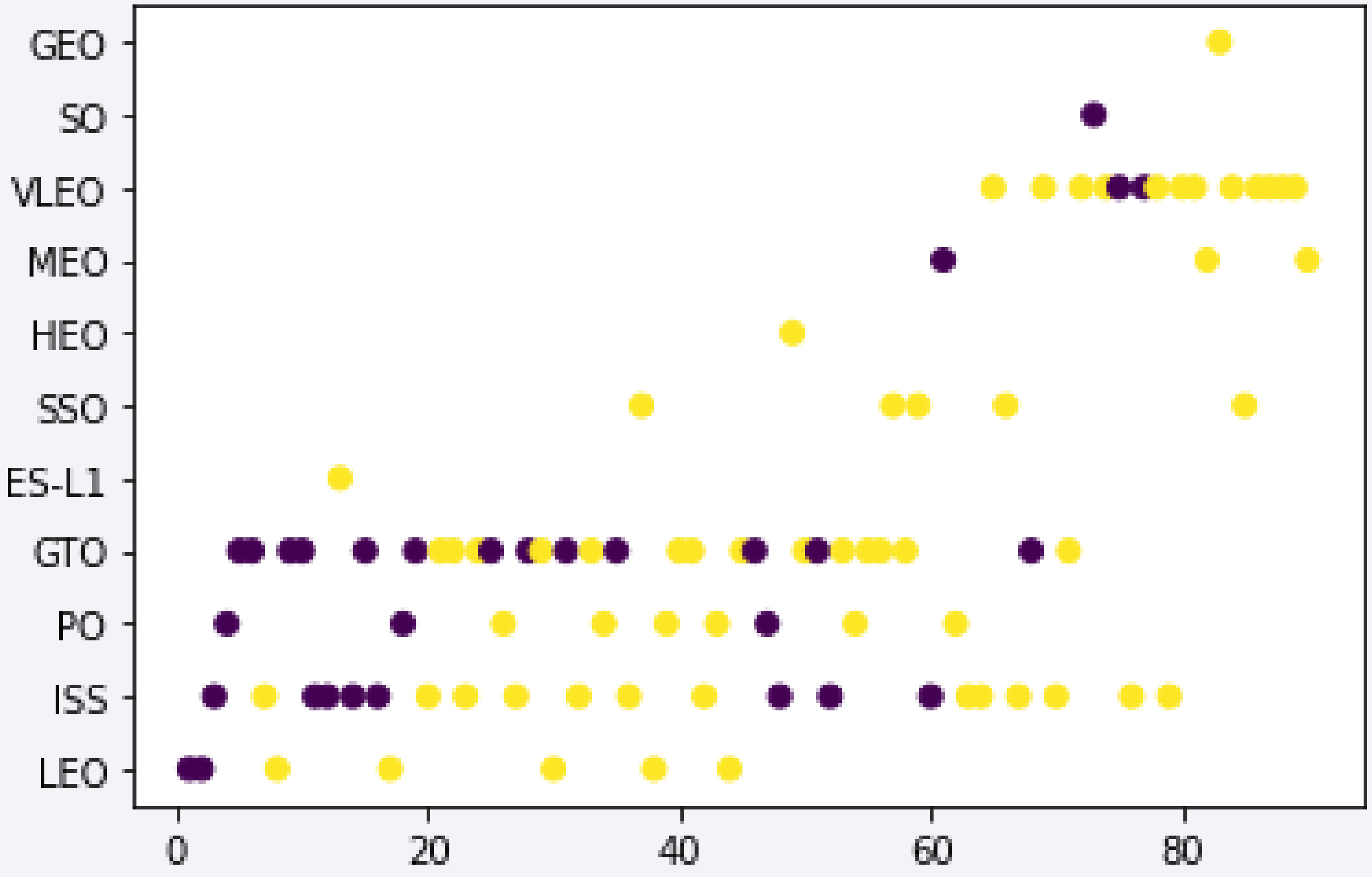
Success Rate vs. Orbit Type

- A bar chart for the success rate of each orbit type



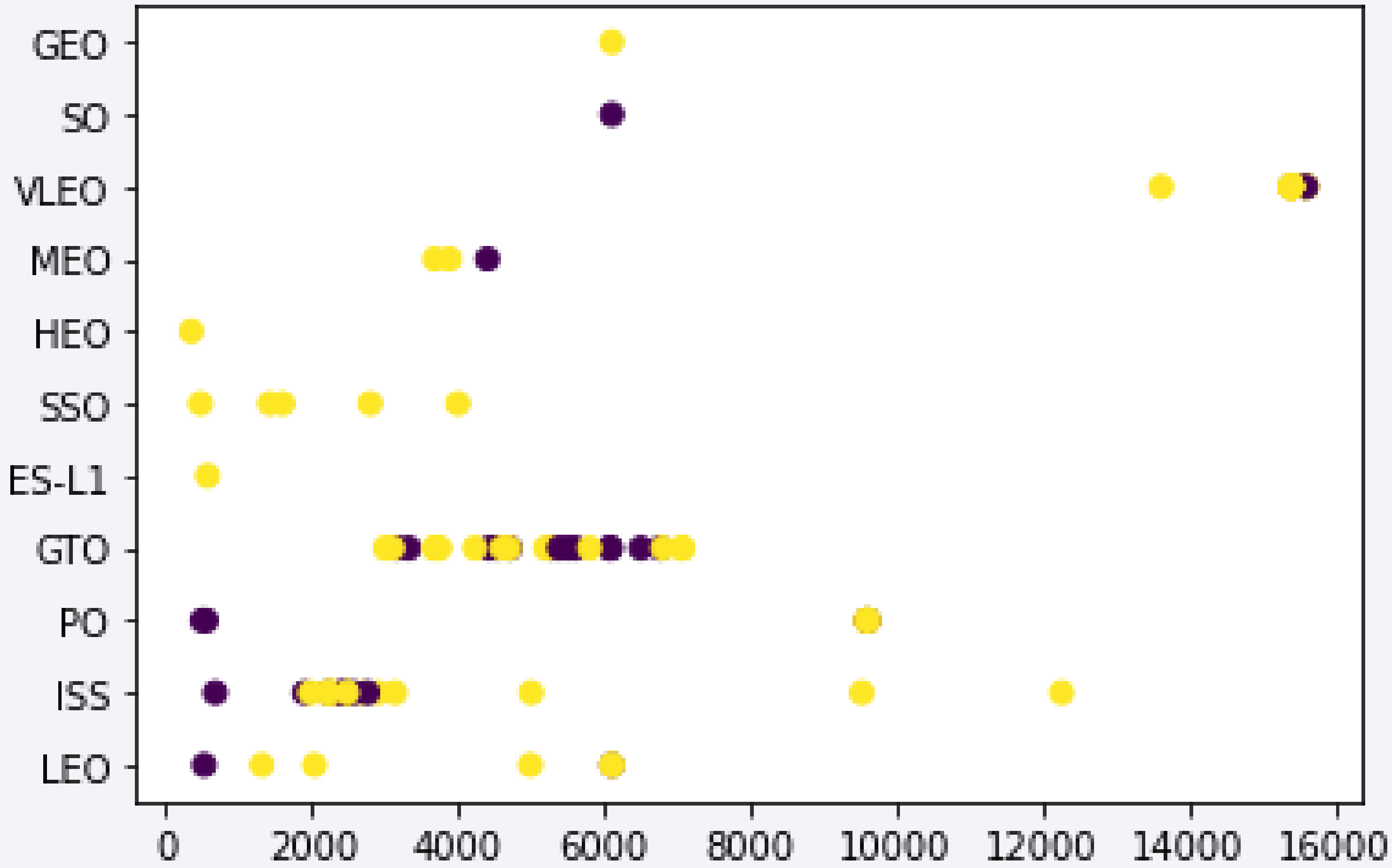
Flight Number vs. Orbit Type

- A scatter point of Flight number vs. Orbit type



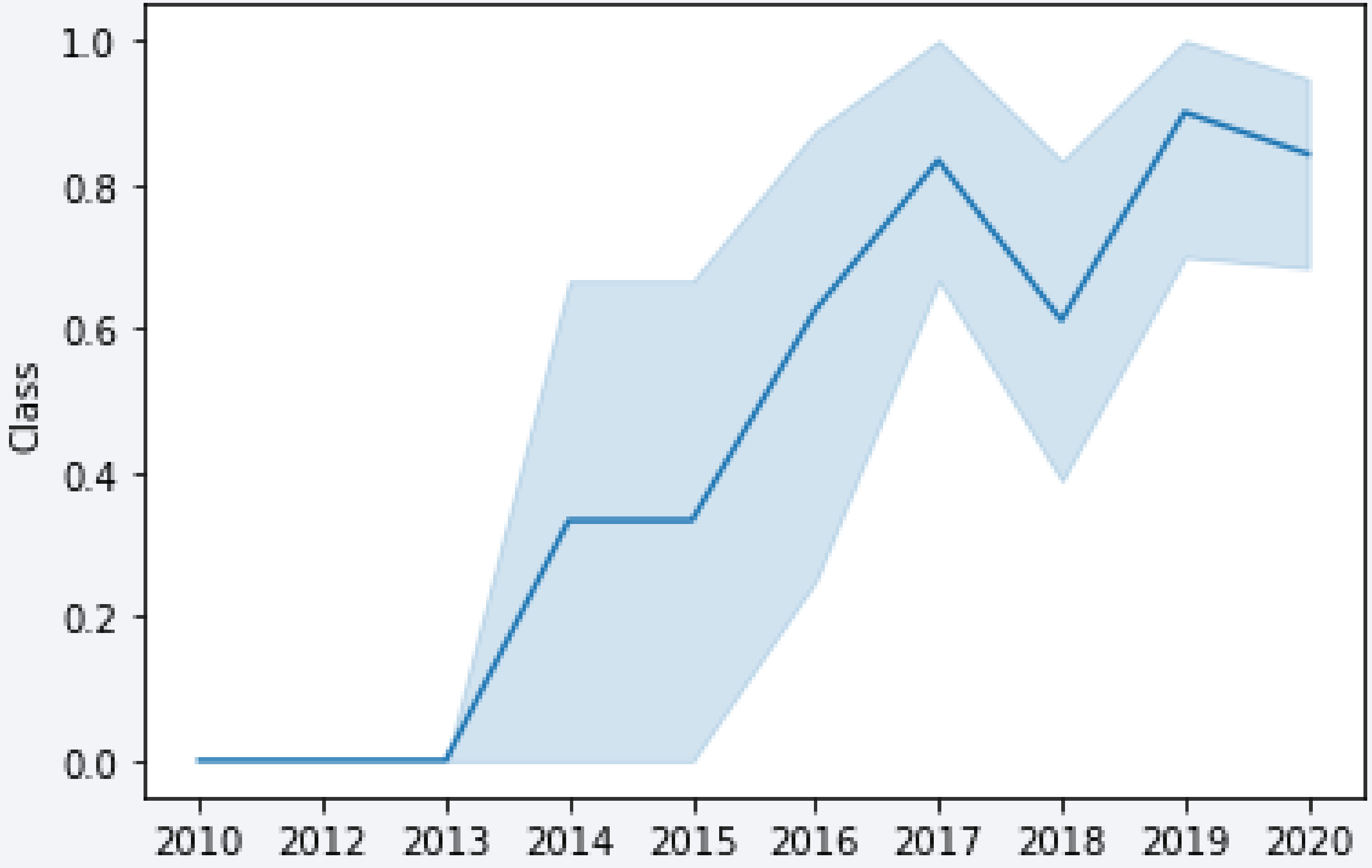
Payload vs. Orbit Type

- A scatter point of payload vs. orbit type



Launch Success Yearly Trend

- A line chart of yearly average success rate



All Launch Site Names

- Query:

```
query = "select distinct(Launch_Site) from SPACEXTBL";
pro1 = conn.execute(query);
df1 = sql_to_df(pro1);
df1
```

- Result:

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

Launch Site Names Begin with 'CCA'

- Query:

```
query = "select * from SPACEXTBL where Launch_Site like 'CCA%' limit 5";
pro2 = conn.execute(query);
df2 = sql_to_df(pro2);
df2
```

- Result:

	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
0	2010-06-04 00:00:00	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-12-08 00:00:00	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2012-05-22 00:00:00	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-10-08 00:00:00	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-03-01 00:00:00	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Query:

```
query = "select sum(PAYLOAD_MASS__KG_) as 'Total_Payload_mass' from SPACEXTBL where Customer == 'NASA (CRS)'";
pro3 = conn.execute(query);
df3 = sql_to_df(pro3);
df3
```

- Result:

	Total_Payload_mass
0	45596

Average Payload Mass by F9 v1.1

- Query:

```
query = "select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where Booster_version like 'F9 v1.1%'";
pro4 = conn.execute(query);
df4 = sql_to_df(pro4);
df4
```

- Result:

	avg(PAYLOAD_MASS__KG_)
0	2534.666667

First Successful Ground Landing Date

- Query:

```
query = "select date from SPACEXTBL where \"Landing _Outcome\" == 'Success (ground pad)' order by date asc limit 1";
pro5 = conn.execute(query);
df5 = sql_to_df(pro5);
df5
```

- Result:

	Date
0	2015-12-22 00:00:00

Successful Drone Ship Landing with Payload between 4000 and 6000

- Query:

```
query = " select booster_version from SPACEXTBL where \"Landing _Outcome\"=='Success (drone ship)' \
          and PAYLOAD_MASS__KG_ between 4000 and 6000";
pro6 = conn.execute(query);
df6 = sql_to_df(pro6);
df6
```

- Result:

	Booster_Version
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- Query:

```
query = "select count(mission_outcome) from SPACEXTBL group by mission_outcome";
pro7 = conn.execute(query);
df7 = sql_to_df(pro7);
df7
```

- Result:

	count(mission_outcome)
0	1
1	98
2	1
3	1

Boosters Carried Maximum Payload

- Query:

```
query = "select booster_version from SPACEXTBL where PAYLOAD_MASS__KG_ == (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)";
pro8 = conn.execute(query);
df8 = sql_to_df(pro8);
df8
```

- Result:

	Booster_Version
0	F9 B5 B1048.4
1	F9 B5 B1049.4
2	F9 B5 B1051.3
3	F9 B5 B1056.4
4	F9 B5 B1048.5
5	F9 B5 B1051.4

2015 Launch Records

- Query:

```
query = " select \"Landing _Outcome\", Booster_version, Launch_Site, date \
          from SPACEXTBL where \"Landing _Outcome\" == 'Failure (drone ship)' and \
          date between '2015-01-01' and '2015-12-31'";
pro9 = conn.execute(query);
df9 = sql_to_df(pro9);
df9
```

- Result:

	Landing _Outcome	Booster_Version	Launch_Site	Date
0	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40	2015-01-10 00:00:00
1	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40	2015-04-14 00:00:00

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

- Query:

```
query = " select date, \"Landing _Outcome\" from SPACEXTBL \
          where date between '2010-06-04' and '2017-03-20' and \
          (\"Landing _Outcome\" == 'Failure (drone ship)' or \"Landing _Outcome\" == 'Success (ground pad)') \
order by date desc";
pro10 = conn.execute(query);
df10 = sql_to_df(pro10);
df10
```

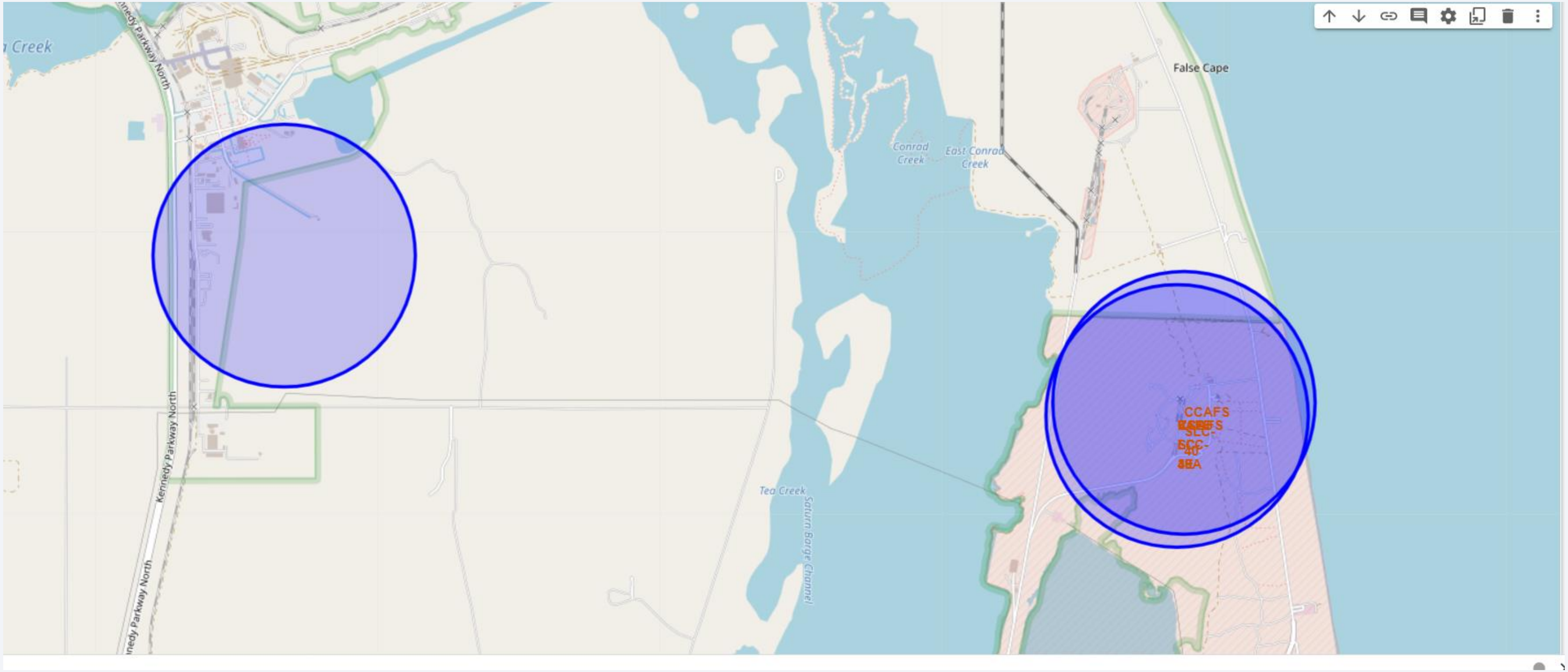
- Result:

	Date	Landing _Outcome
0	2017-02-19 00:00:00	Success (ground pad)
1	2016-07-18 00:00:00	Success (ground pad)
2	2016-06-15 00:00:00	Failure (drone ship)
3	2016-03-04 00:00:00	Failure (drone ship)
4	2016-01-17 00:00:00	Failure (drone ship)
5	2015-12-22 00:00:00	Success (ground pad)
6	2015-04-14 00:00:00	Failure (drone ship)
7	2015-01-10 00:00:00	Failure (drone ship)

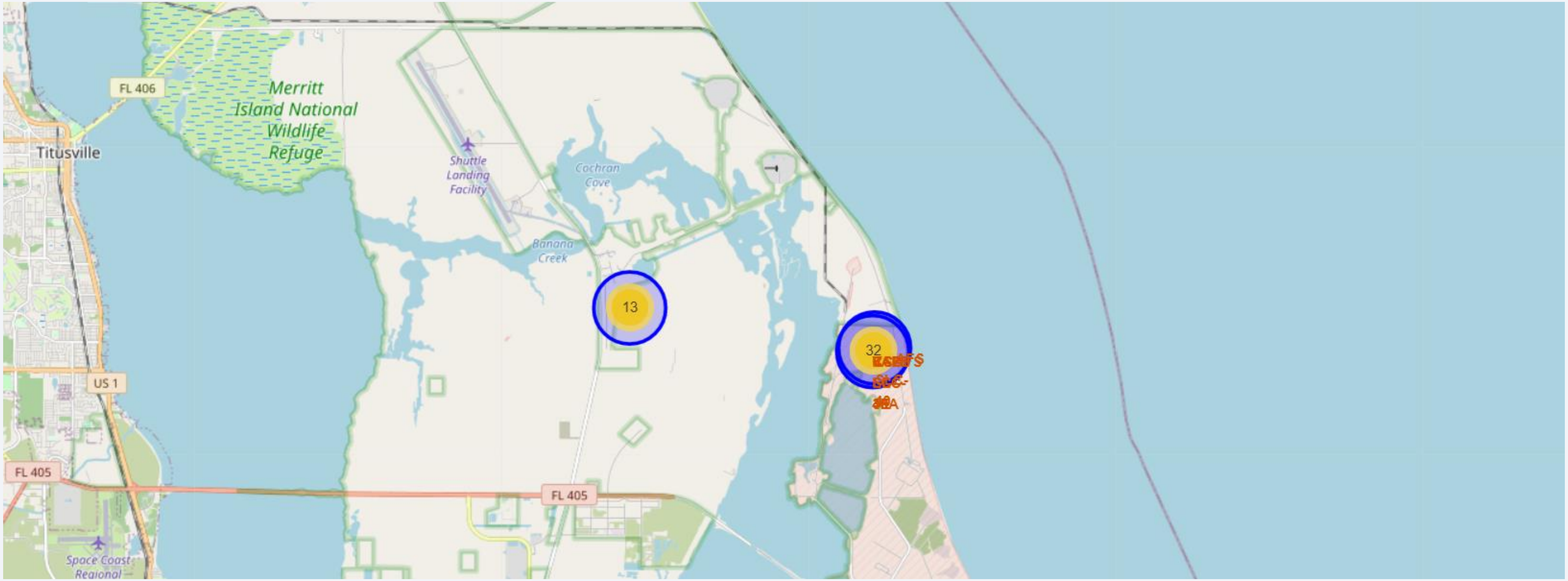
Section 4

Launch Sites Proximities Analysis

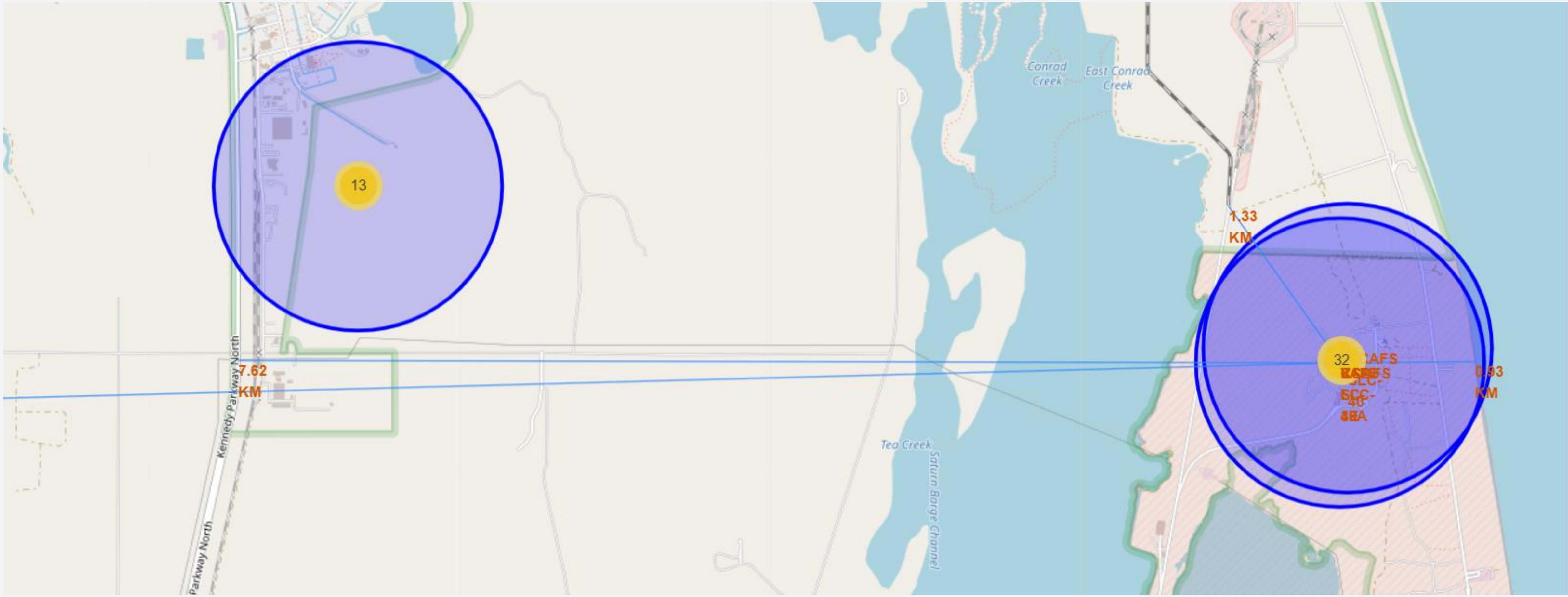
All launch sites' location



Launch outcomes



Railway, highway, coastline

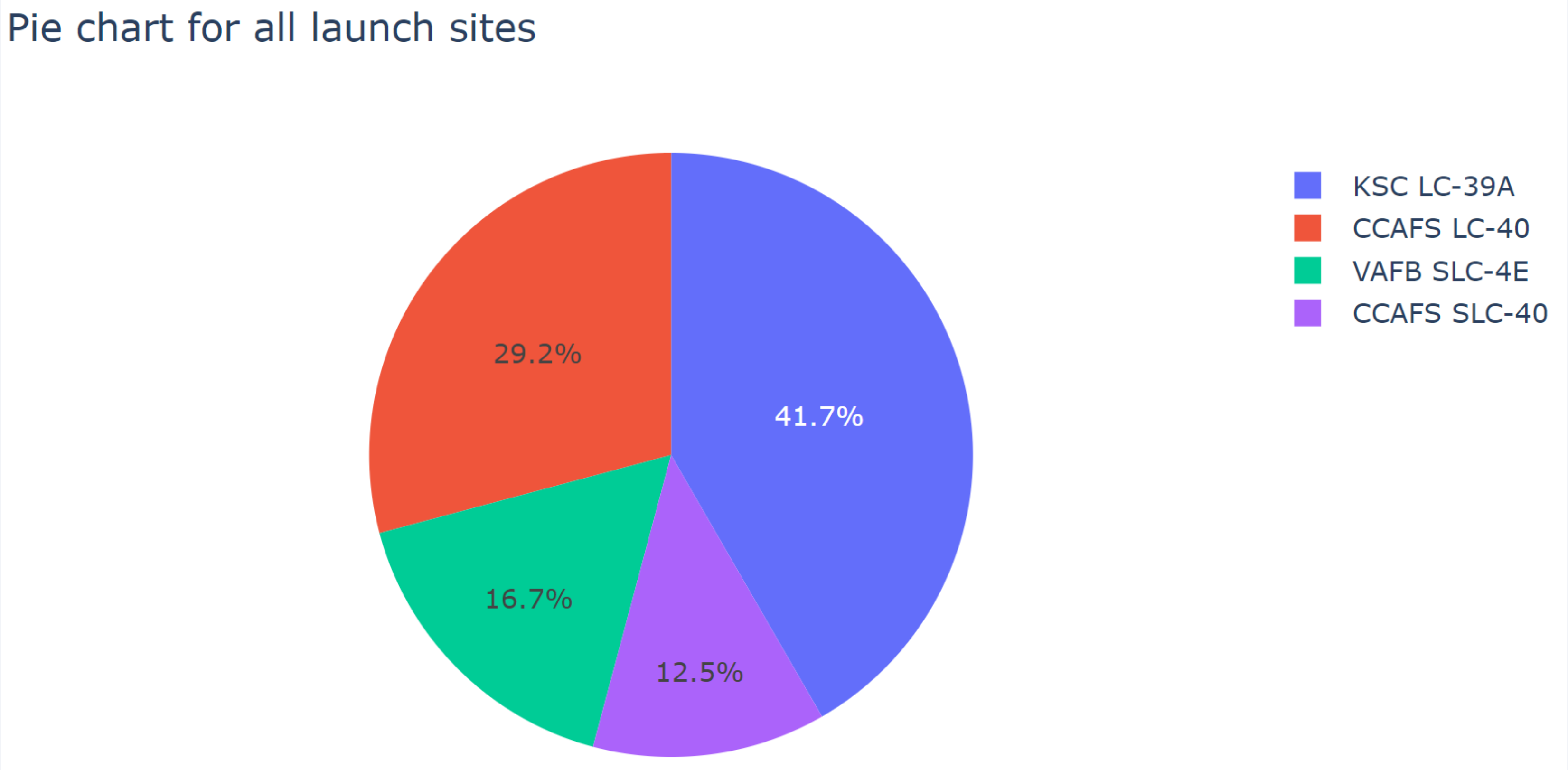


Section 5

Build a Dashboard with Plotly Dash

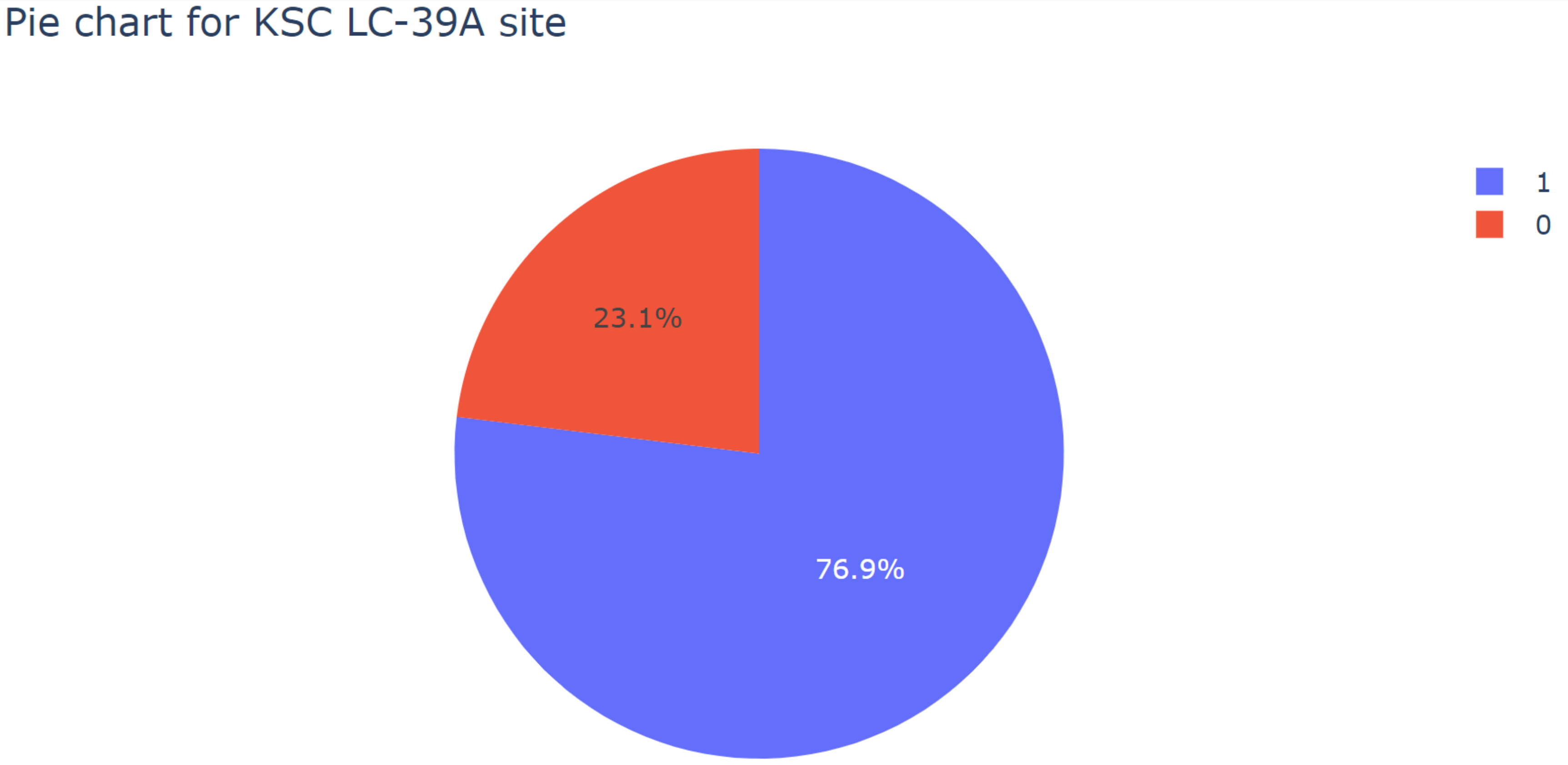
The rate of Launch Sites

- Based on this chart, we can be easy to discover that the KCS LC-39A accounts a large proportion.



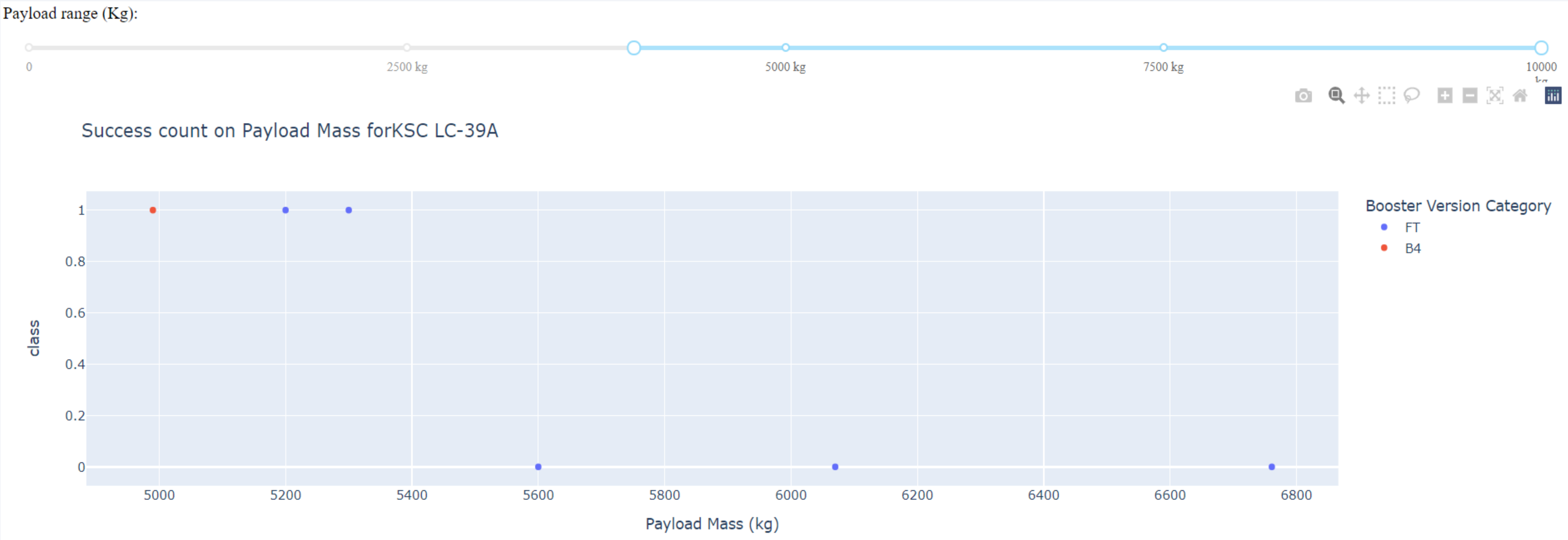
Setting the payload slider

- Based on this chart of KCS LC-39A category, we can be easy to discover that “class 1” accounts a large proportion.



Payload mass chart with range set in slider

- In range 5000-7000, the classes are clearer and FT accounts a large proportion.



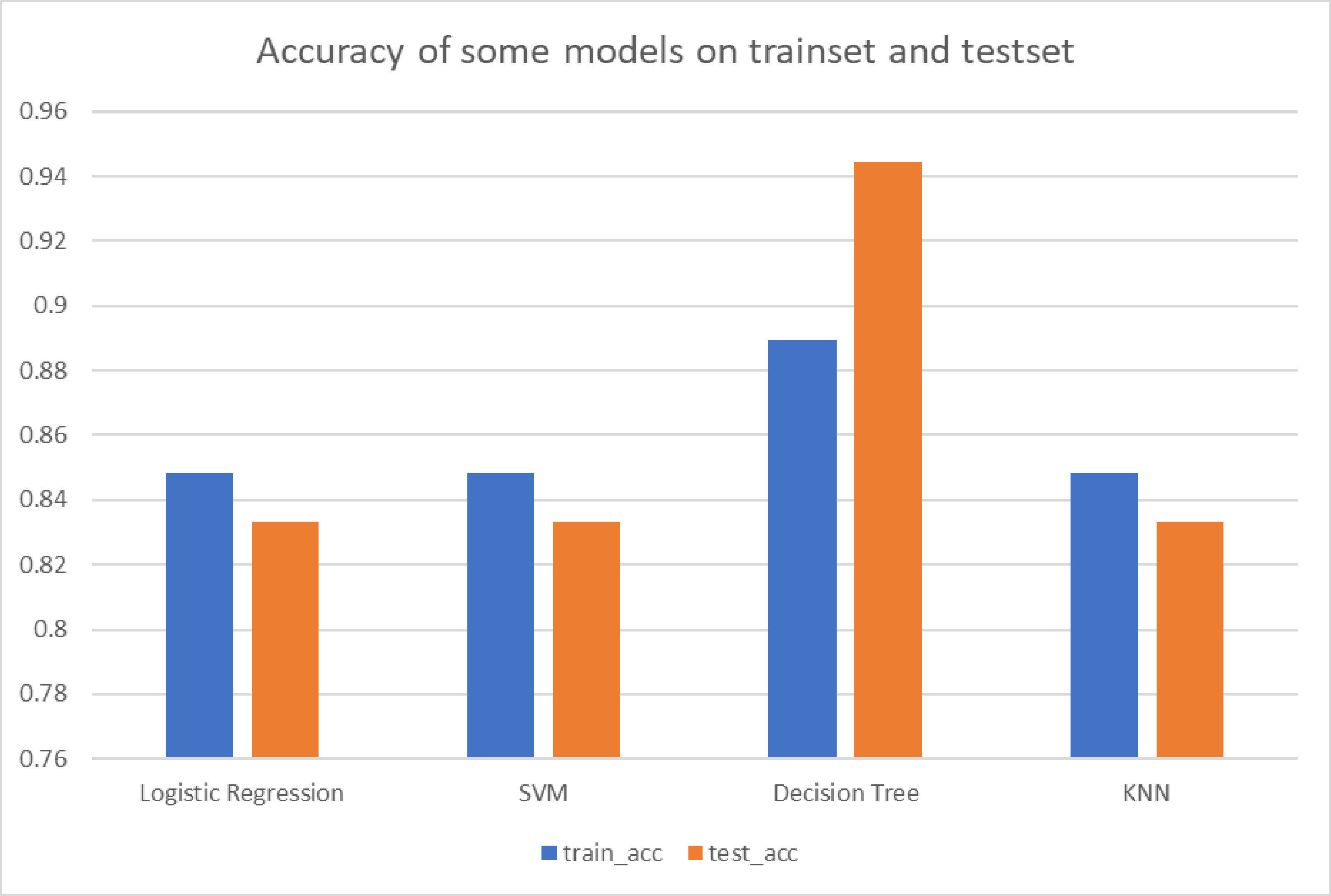
Section 6

Predictive Analysis (Classification)

Classification Accuracy

- The highest classification accuracy: 94.44%

➔ The best model is: Decision Tree model



Confusion Matrix

- The confusion matrix of the best performing model:

- Explanation

- Even though the accuracy on testset of model is high (94.44%), the f-score is small() because of the un-balanced data.

Precision = 0.5

Recall = 0.2

F-score = 2*0.2*0.5/0.7=0.2857

Conclusions

Task	Point
Uploaded the URL of your GitHub repository including all the completed notebooks and Python files (1 pt)	1
Uploaded your completed presentation in PDF format (1 pt)	1
Completed the required Executive Summary slide (1 pt)	1
Completed the required Introduction slide (1 pt)	1
Completed the required data collection and data wrangling methodology related slides (1 pt)	1
Completed the required EDA and interactive visual analytics methodology related slides (3 pts)	3
Completed the required predictive analysis methodology related slides (1 pt)	1
Completed the required EDA with visualization results slides (6 pts)	6
Completed the required EDA with SQL results slides (10 pts)	10
Completed the required interactive map with Folium results slides (3 pts)	3
Completed the required Plotly Dash dashboard results slides (3 pts)	3
Completed the required predictive analysis (classification) results slides (6 pts)	6
Completed the required Conclusion slide (1 pts)	1
Applied your creativity to improve the presentation beyond the template (1 pts)	1
Displayed any innovative insights (1 pts)	1

Appendix

- My all links of exercises of 10 courses of IBM Data Science Courses

Github:

https://github.com/WindPham/Cousera/tree/master/IBM_Data_Science

Thank you!