IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Applied Data Science Capstone project

Sebastian Tølbøll Glavind

May 17, 2022

Outline

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

Executive Summary

Methodology

- Collected data from public SpaceX API and SpaceX Wikipedia page, as basis for predicting whether the first stage would land
- Filtered, cleaned and formatted the date for further analysis, e.g., class label and data imputation
- Explored data using SQL, matplotlib visualization, folium maps, and dashboards
- Defined features and outcome for classification models. Changed all categorical variables to binary using one hot encoding. Standardized the data. Used GridSearchCV to find best parameters for machine learning models. Visualize accuracy score of all models.

Results

- The four considered machine learning models, i.e., logistic regression, support vector machine, decision tree, and k-nearest-neighbors performed equally well in predicting the landing outcome of the first stage, i.e., test set accuracy of 0.83 and similar confusion matrix.
- We may be able to improve our predictions by accounting for class imbalance in the database.

Introduction

Background and context

- The commercial space age is here!
- Most cost efficient is SpaceX (Falcon 9) with a cost of 62 million dollars per launch compared to a cost upwards of 165 million dollars.
- The competitive advantage is SpaceX's ability to recover the first stage.
- SpaceY would like to compete with SpaceX

Problem

- Determine the price of each launch by;
  - Gather public information about spaceX to gather insights
  - Build a machine learning model to predict reuse of the first stage

Section 1

Methodology

Methodology

Executive Summary

- Data collection and wrangling:
  - Collect data from SpaceX API and Wikipedia page
  - Filter data to only include Falcon 9 launches, impute missing data, created a landing outcome variable
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Model were build using Scikit-learn, tuned using grid search, and evaluated by predictive accuracy and confusion matrix on a held-out test set.

Data Collection

- The data collection involved a combination of API requests to SpaceX's public API and web scraping from Space X's Wikipedia.
- The next slides will show the flowchart for data collection from API and Wikipedia, respectively.
- Information gathered from;

SpaceX API

- FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude

Wikipedia page

- Flight No., Launch site, Payload, PayloadMass, Orbit, Customer, Launch outcome, Version Booster, Booster landing, Date, Time

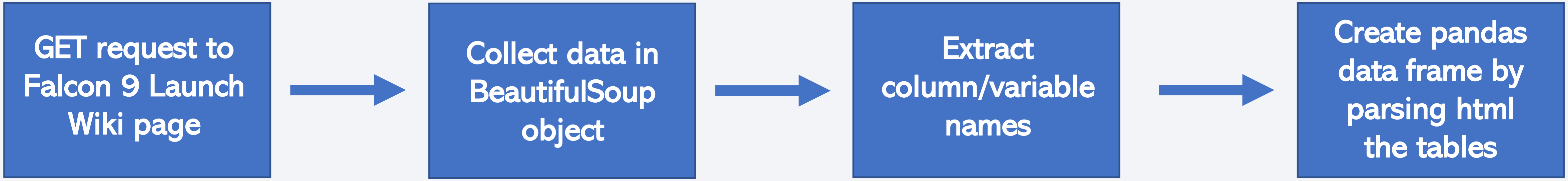
Data Collection – SpaceX API

GET request to SpaceX API → Collect data in pandas data frame → Filter data to include only Falcon 9 → Impute missing data by the mean

Github URL:

https://github.com/SebastianGlavind/IBM-DataScienceProfessionalCertificate/blob/main/AppliedDataScienceCapstone/Data%20Collection%20API.ipynb

Data Collection - Scraping



Github URL:

https://github.com/SebastianGlavind/IBM-DataScienceProfessionalCertificate/blob/main/AppliedDataScienceCapstone/Data%20Collection%20with%20Web%20Scraping.ipynb

Data Wrangling

- Initially some exploratory data analysis was performed to ensure data quality and encoding, i.e., check for missing values, and calculate: # launches from each site, # occurrence of each orbit, and # mission outcome per orbit type
- Create a landing outcome label from Outcome column: 0-1 encoding.
  - Failure (=0) for outcomes {'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None’}
  - Success (=1) for outcomes {'True Ocean', 'True RTLS', 'True ASDS'}

Initial EDA to ensure data quality and encoding → Include a class variable in the data set for landing outcome

Github URL:

https://github.com/SebastianGlavind/IBM-DataScienceProfessionalCertificate/blob/main/AppliedDataScienceCapstone/Data%20Wrangling%20(EDA).ipynb

EDA with Data Visualization

- Carplots to visualize the trend between two features considering the class label – was the launch successful or not for the feature configuration?
  - FlightNumber vs. PayloadMass with class overlay
  - FlightNumber vs. LaunchSite with class overlay
  - LaunchSite vs. PayloadMass with class overlay
  - FlightNumber vs. Orbit with class overlay
  - PayloadMass vs. Orbit with class overlay
- Histogram of success rate per orbit – are some orbits are more successful?
- Trend plot of launch successes yearly trend – what is success trend?
- Finally, dummy encoding was performed on the discrete features

Github URL:

https://github.com/SebastianGlavind/IBM-DataScienceProfessionalCertificate/blob/main/AppliedDataScienceCapstone/EDA%20with%20Pandas%20and%20Matplotlib.ipynb

EDA with SQL

- Loading the data into a IBM DB2 database, which enabled that a set of queries could be performed on the database with SQL using the Python integration.
- The following queries were made to gain insights about the launches
    - Unique launch sites
    - Information regarding specific launch sites, booster versions and their payload
    - Information regarding average payload mass caried by specific booster types
    - Information on successful/failed missions (amount, dates, booster vision given pay load mass range)
    - Booster versions that carried maximum payload
    - Successful and failed missions within specific time frames

Github URL:

https://github.com/SebastianGlavind/IBM-DataScienceProfessionalCertificate/blob/main/AppliedDataScienceCapstone/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- All launch sites were mapped to a folium.Map with folium.map.Marker's, and the launches from each site were marked according to the class variable (success/failure) using marker clusters. Finally, the distances between a launch site to its proximities were calculated.
- These operations enabled that the launch site could be located, the success rate of each launch site could be judged qualitatively, and distances between each sites and its proximities could be evaluated to define distance-metrics for placing launch sites.

Github URL:

https://github.com/SebastianGlavind/IBM-DataScienceProfessionalCertificate/blob/main/AppliedDataScienceCapstone/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

Build a Dashboard with Plotly Dash

- A dashboard with a pie chart and a scatter plot were created;
  - The pie chart can show: (i) the distribution of success landing across all launch sites (ALL SITES entry); and (ii) individual launch site success rates (specific launch site entry).
  - The scatter plot shows the correlation between pay load and successful landings. Again, considering either ALL SITES or a specific site.
- Thus, the pie chart shows the launch sites success rates, and the scatter plot shows how successes varies across launch sites, payloads and booster versions.

Github URL:

https://github.com/SebastianGlavind/IBM-DataScienceProfessionalCertificate/blob/main/AppliedDataScienceCapstone/spacex_dash_app.py

Predictive Analysis (Classification)

Load data in pandas data frame

Define input/feature data X and output data Y

Standardize features (StandardScaler)

Split the data into a training and a test set (train_test_split)

Fit classifiers on training set (GridSearchCV)

Score classifiers on test set (accuracy and confusion matrix)

Compare the classifier performances on test set

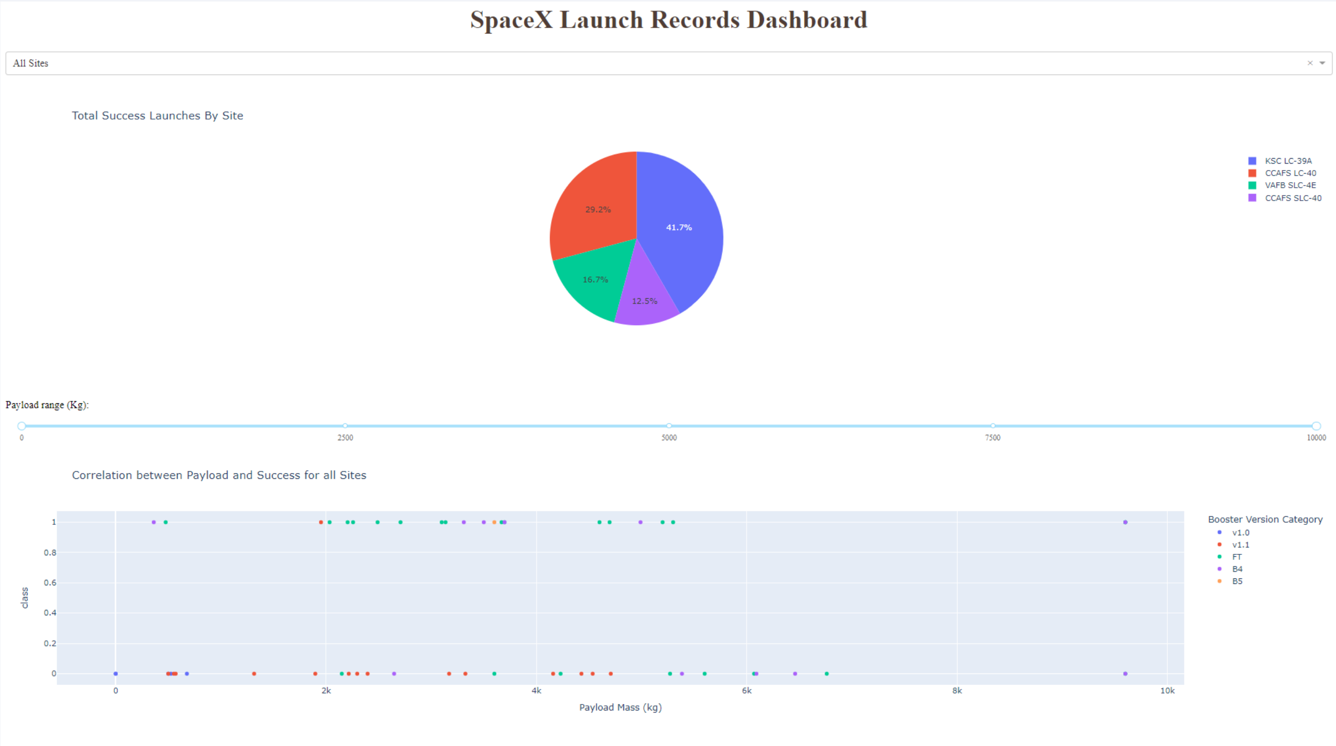
Considered classifiers:
- Logistic regression
- Support vector machine
- Decision tree
- k-nearest-neighbors

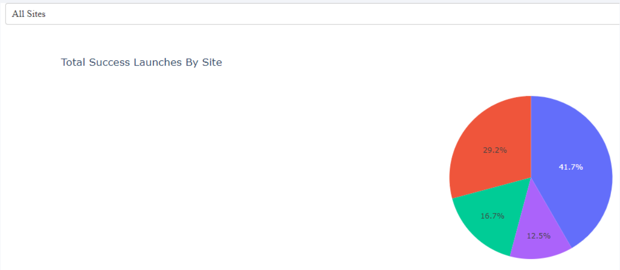
Github URL:

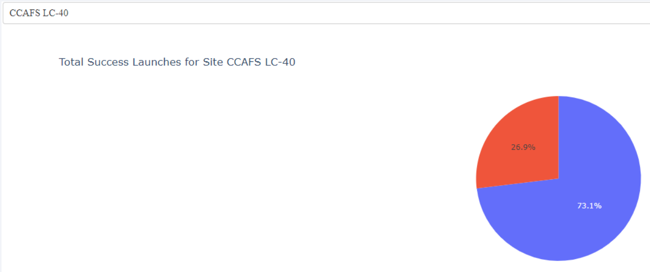
https://github.com/SebastianGlavind/IBM-DataScienceProfessionalCertificate/blob/main/AppliedDataScienceCapstone/Machine%20Learning%20Prediction.ipynb

Results

- The exploratory data analysis shows e.g., that different launch sites have different success rates, and it provides some explanations to why this is the case, e.g., different payload and orbit.
- The dashboard showed e.g., that site KSC LC-39A has the highest success rate and site VAFC SLC-4E has the larges successful launches in terms of payload.
- All classifiers performed equally well on the test set, i.e., all have an accuracy of 0.83 and similar confusion matrices.
- The following slides will show the results of the individual analysis in detail.





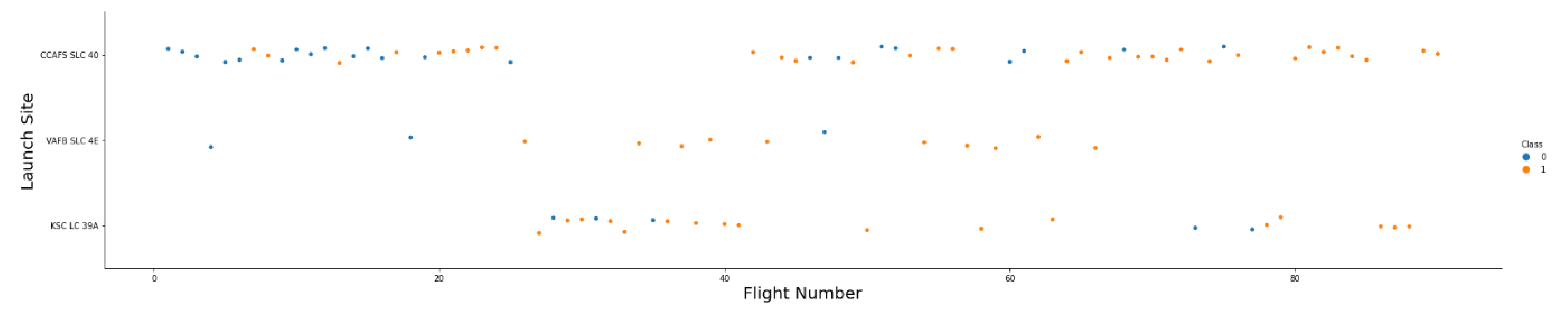


Section 2

Insights drawn from EDA

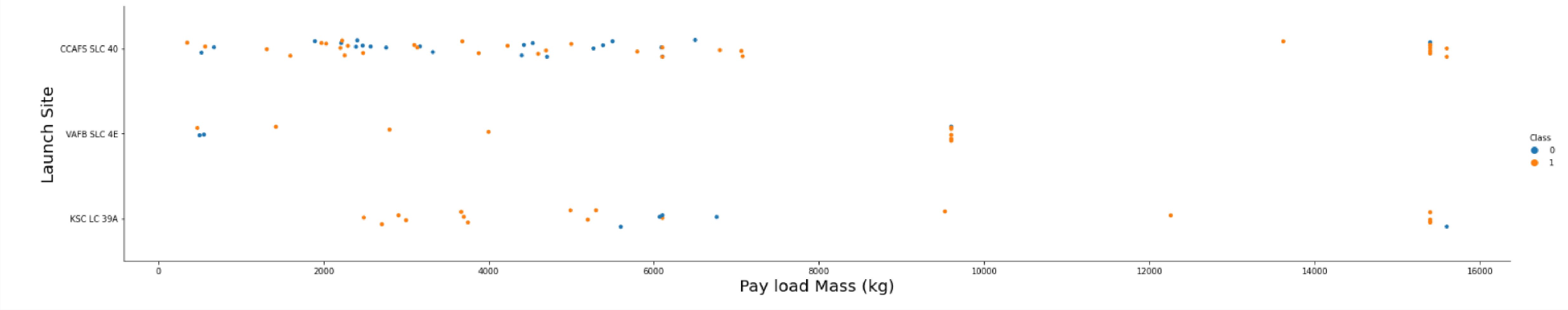
Flight Number vs. Launch Site

- Carplots to visualize the trend between two features considering the class label
  - **FlightNumber** vs. **LaunchSite** with **class** overlay
  - E.g., some site are more used and successful for given flights



Payload vs. Launch Site

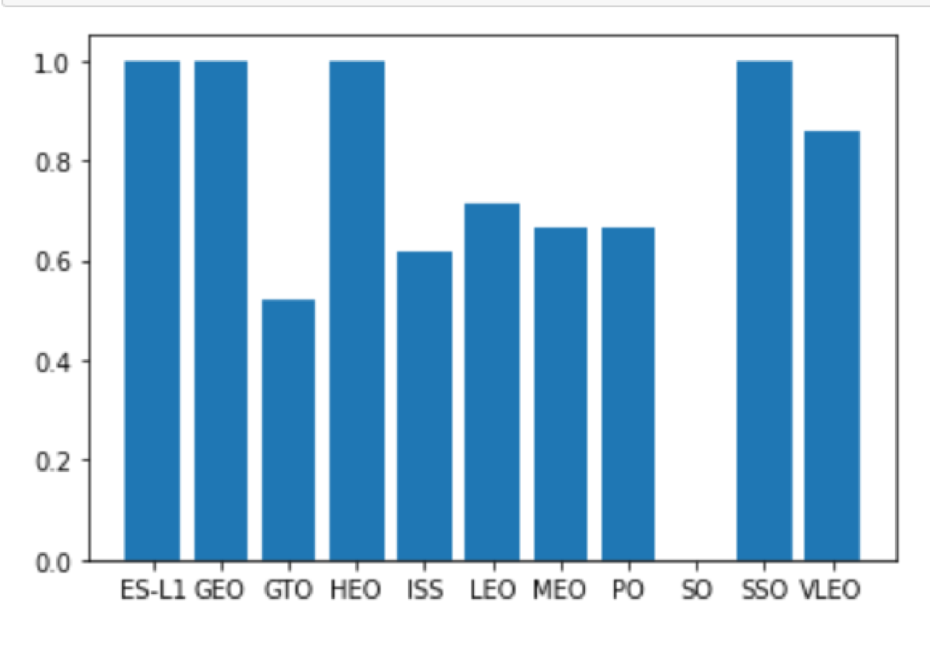
- Carplots to visualize the trend between two features considering the class label
  - **LaunchSite** vs. **PayloadMass** with **class** overlay
  - E.g., VAFB-SLC has no heavy masses (greater than 10000)



Success Rate vs. Orbit Type

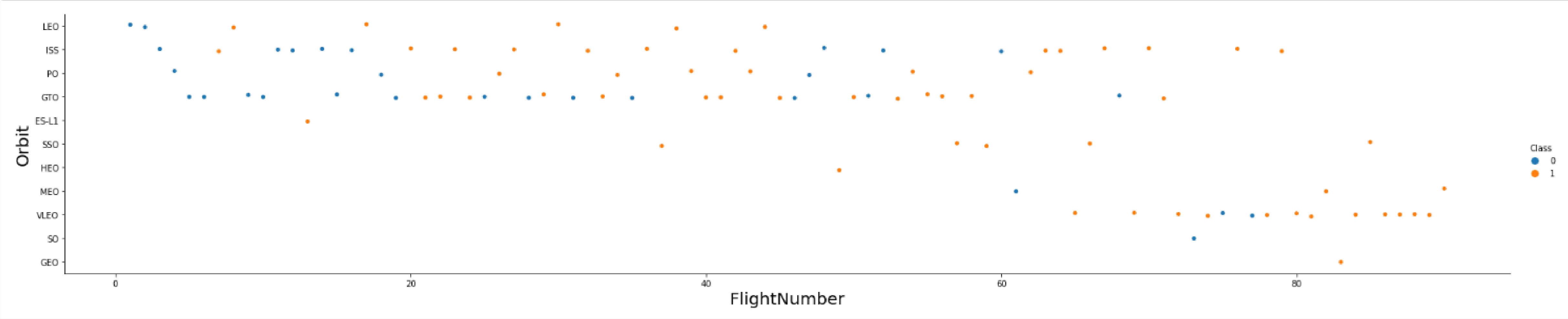
- Bar chart of success rate per orbit
  - E.g., some orbits are more successful, e.g., SSO (=1)

```
Orbit
ES-L1    1.000000
GEO      1.000000
GTO      0.518519
HEO      1.000000
ISS      0.619048
LEO      0.714286
MEO      0.666667
PO       0.666667
SO       0.000000
SSO      1.000000
VLEO     0.857143
Name: Class, dtype: float64
```



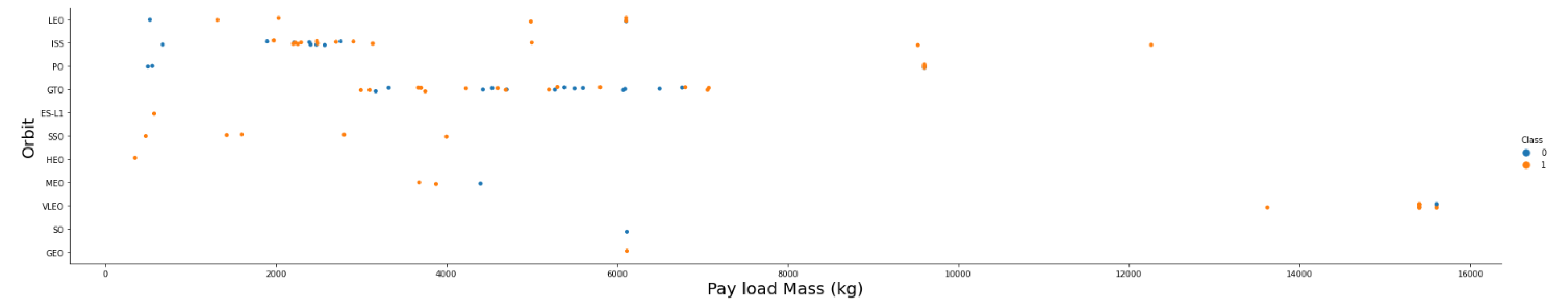
Flight Number vs. Orbit Type

- Carplots to visualize the trend between two features considering the class label
  - **FlightNumber** vs. **Orbit** with **class** overlay
  - E.g., success for some orbits depend on flight number



Payload vs. Orbit Type

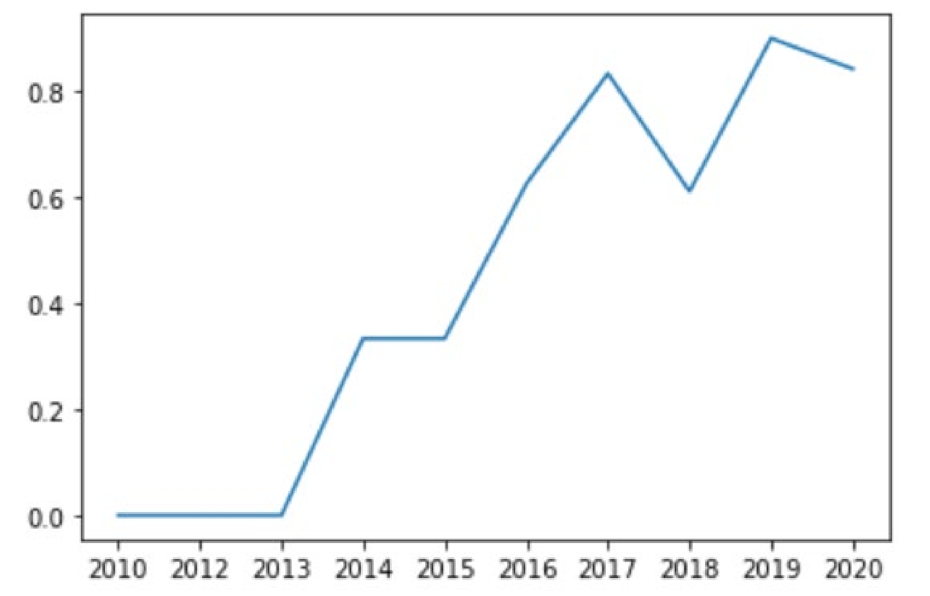
- Carplots to visualize the trend between two features considering the class label
  - **PayloadMass** vs. **Orbit** with **class** overlay
  - E.g., success for some orbits become higher with increasing mass (more valuable assets)



Launch Success Yearly Trend

- Line chart of yearly average success rate trend
  - E.g., more successful over the years

```
2010    0.000000
2012    0.000000
2013    0.000000
2014    0.333333
2015    0.333333
2016    0.625000
2017    0.833333
2018    0.611111
2019    0.900000
2020    0.842105
Name: Class, dtype: float64
```



All Launch Site Names

- Unique launch site names
  - SQL statements: SELECT, DISTINCT
  - E.g., there are four unique launch site names

```
In [7]: %sql SELECT DISTINCT launch_site FROM SPACEXDATASET;

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.

Out[7]:
```

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

Launch Site Names Begin with 'CCA'

- First 5 records where the launch site begin with `CCA`
  - SQL statements: SELECT, SUBSTRING, LIMIT (or SELECT, LIKE, LIMIT)
  - E.g., all these launches went to the LEO orbit with booster version F9 v1.0

```
In [13]: %sql SELECT * FROM SPACEXDATASET WHERE SUBSTRING( launch_site, 1, 3 )='CCA' LIMIT 5;

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[13]:

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

- Total payload carried by boosters from NASA
  - SQL statements: SELECT, (SUM,) WHERE, “=”

```
In [16]: %sql SELECT payload_mass__kg_ FROM SPACEXDATASET WHERE customer='NASA (CRS)';
 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[16]:

payload_mass__kg_
500
677
2296
2216
2395
1898
1952
3136
2257
2490
2708
3310
2205
2647
2697
2500
2495
2268
1977
2972

```
In [55]: %sql SELECT SUM(payload_mass__kg_) FROM SPACEXDATASET WHERE customer='NASA (CRS)';
 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[55]:

1
45596

Average Payload Mass by F9 v1.1

- Average payload mass carried by booster version F9 v1.1
  - SQL statements: SELECT, AVG, WHERE, “=”

```
In [20]: %sql SELECT payload_mass__kg_ FROM SPACEXDATASET WHERE booster_version='F9 v1.1';

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[20]:

payload_mass__kg_
3170
3325
2296
1316
4535

```
In [24]: %sql SELECT AVG(payload_mass__kg_) AS "avg_pay_load_mass" FROM SPACEXDATASET WHERE booster_version='F9 v1.1';

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[24]:

avg_pay_load_mass
2928

First Successful Ground Landing Date

- Date of the first successful landing outcome on ground pad
  - SQL statements: SELECT, MIN, WHERE, “=” (or SELECT, WHERE, “=”, OREDER BY, LIMIT)

```
In [27]: %sql SELECT MIN(DATE) AS "First success" FROM SPACEXDATASET WHERE landing__outcome='Success (ground pad)';

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[27]:

First success
2015-12-22

```
In [29]: %sql SELECT DATE FROM SPACEXDATASET WHERE landing__outcome='Success (ground pad)' ORDER BY DATE LIMIT 1;

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[29]:

DATE
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
  - SQL statements: SELECT, WHERE, AND, BETWEEN (or SELECT, WHERE, 2xAND)

```
In [33]: %sql SELECT DATE, booster_version FROM SPACEXDATASET WHERE (landing__outcome='Success (drone ship)') AND (payload_mass__kg_ BETW
EEN 4000 AND 6000);

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[33]:

DATE	booster_version
2016-05-06	F9 FT B1022
2016-08-14	F9 FT B1026
2017-03-30	F9 FT B1021.2
2017-10-11	F9 FT B1031.2

```
In [35]: %sql SELECT DATE, booster_version, payload_mass__kg_ FROM SPACEXDATASET WHERE (landing__outcome='Success (drone ship)') AND (pay
load_mass__kg_ > 4000) AND (payload_mass__kg_ < 6000);

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[35]:

DATE	booster_version	payload_mass__kg_
2016-05-06	F9 FT B1022	4696
2016-08-14	F9 FT B1026	4600
2017-03-30	F9 FT B1021.2	5300
2017-10-11	F9 FT B1031.2	5200

Total Number of Successful and Failure Mission Outcomes

- Total number of successful and failure mission outcomes
  - SQL statements: SELECT, WHERE, LIKE

```
In [37]: %sql SELECT COUNT(*) AS "Successful missions" FROM SPACEXDATASET WHERE mission_outcome LIKE 'Success%';

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[37]:

Successful missions
100

```
In [38]: %sql SELECT COUNT(*) AS "Failed missions" FROM SPACEXDATASET WHERE mission_outcome LIKE 'Failure%';

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[38]:

Failed missions
1

```
In [39]: %sql SELECT COUNT(*) AS "All missions" FROM SPACEXDATASET;

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[39]:

All missions
101

Boosters Carried Maximum Payload

- Names of the booster which have carried the maximum payload mass
  - SQL statements: SELECT, WHERE, SELECT, MAX (subquery)

In [41]:
```
%sql SELECT booster_version FROM SPACEXDATASET WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXDATASET );
```

```
 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

In [43]:
```
%sql SELECT MAX(payload_mass__kg_) AS "max_payload" FROM SPACEXDATASET;
```

```
 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[43]:

max_payload
15600

In [44]:
```
%sql SELECT COUNT(*) FROM SPACEXDATASET WHERE payload_mass__kg_ = 15600;
```

```
 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[44]:

1
12

Out[41]:

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

- Failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
  - SQL statements: SELECT, WHERE, “=“, AND, SUBSTRING

```
In [45]: %sql SELECT * FROM SPACEXDATASET WHERE landing__outcome='Failure (drone ship)' AND SUBSTRING(DATE,1,4)=2015;

 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[45]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2015-01-10	09:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)

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

- Ranking of the counts 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 statements: SELECT, COUNT, WHERE, BETWEEN, GROUP BY, ORDER BY DESC

In [71]:
```
%sql SELECT landing__outcome, COUNT(*) AS "num_landing_out" FROM SPACEXDATASET WHERE (DATE BETWEEN '2010-06-04' AND '2017-03-2
0') GROUP BY landing__outcome ORDER BY "num_landing_out" DESC;
```

```
 * ibm_db_sa://zvc27717:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/BLUDB
Done.
```

Out[71]:

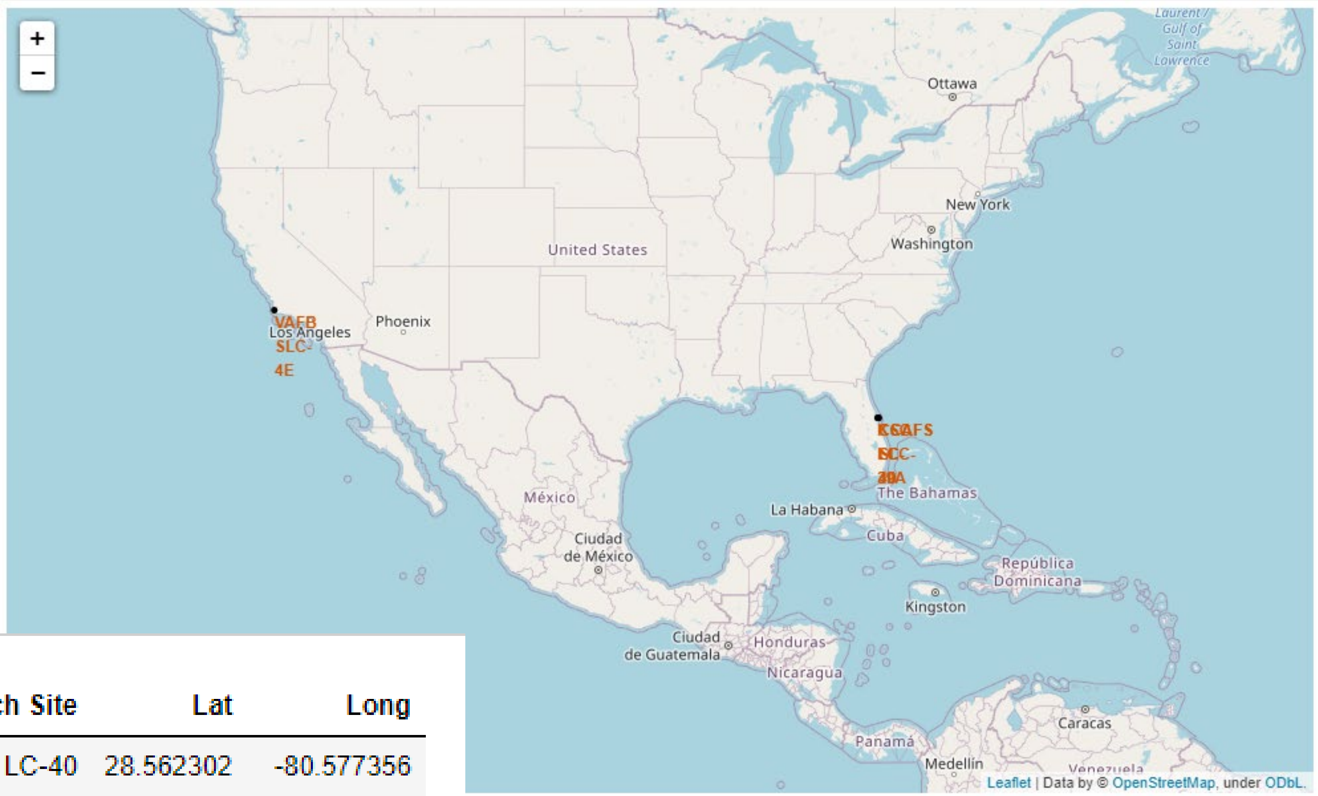
landing__outcome	num_landing_out
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

Launch sites on Folium Map

- The map shows the location of the launch sites, which are located on the west coast, and east coast, respectively.
- The coordinates are shown below the map.

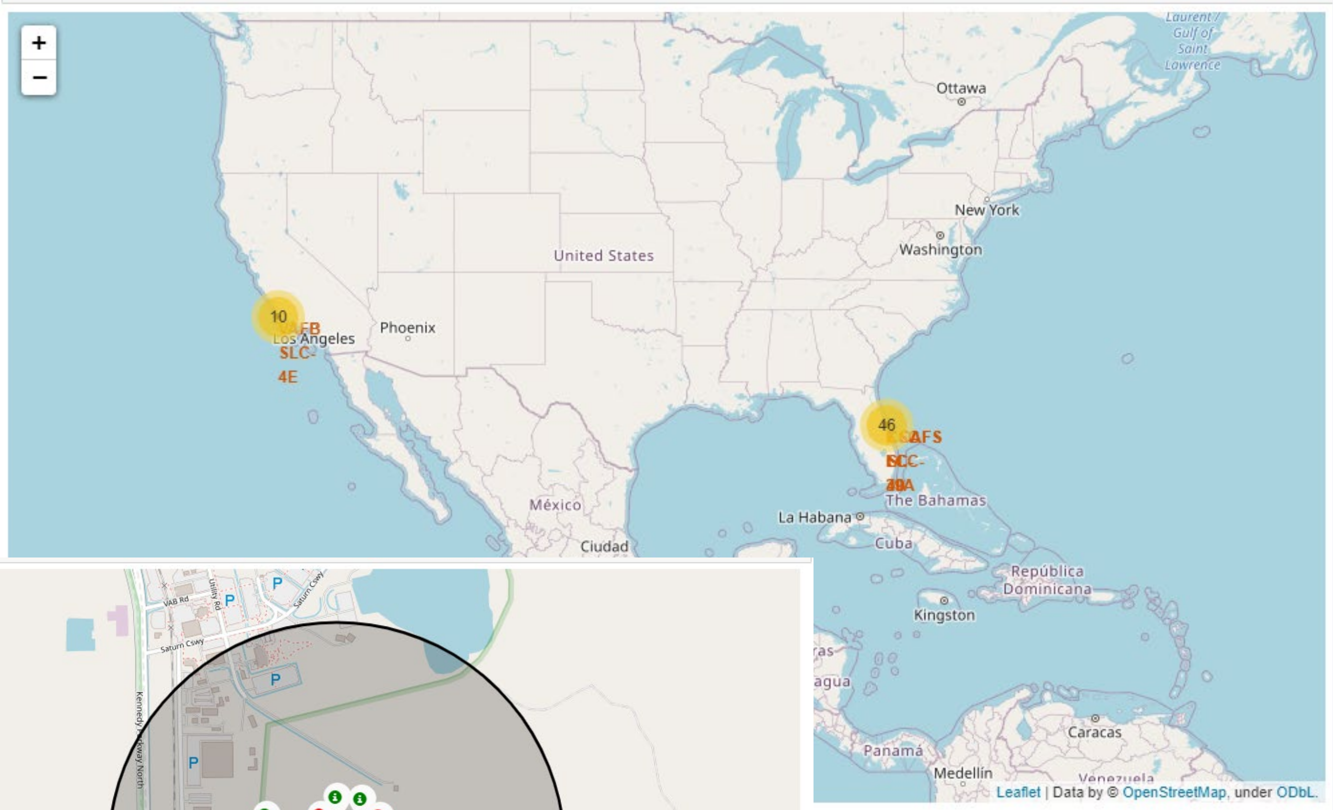


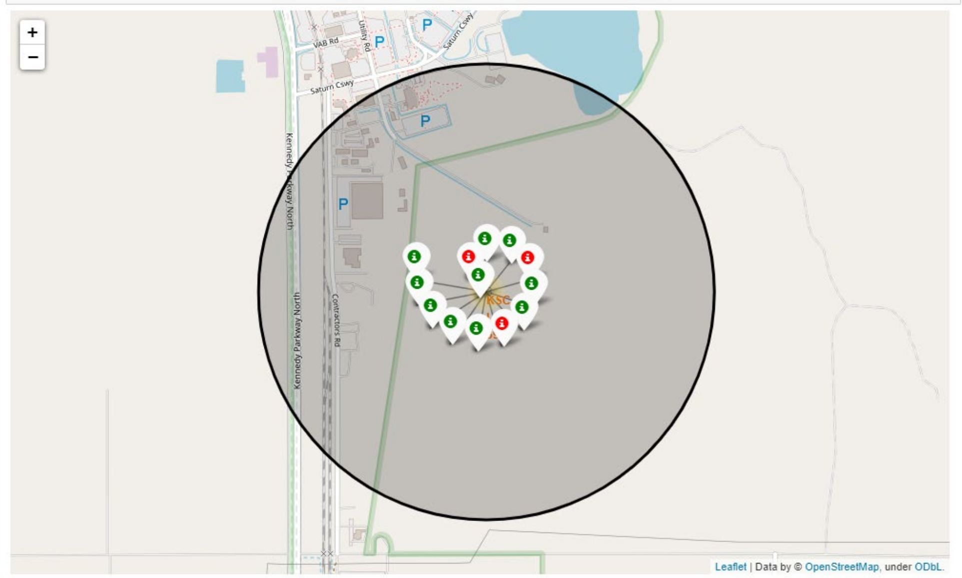
Out[125]:

	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	-120.610745

Grouped launches on Folium Map

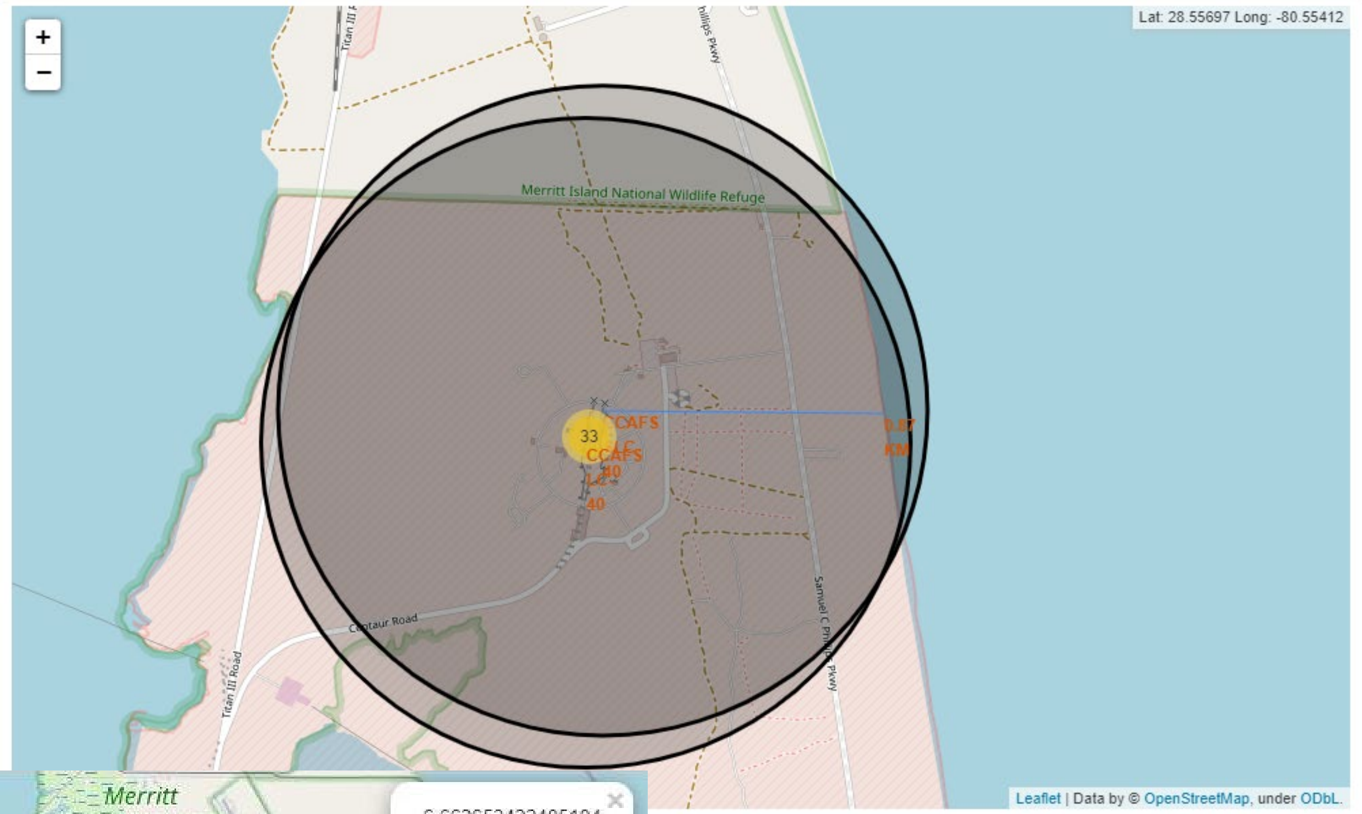
- The map shows the individual launches clustered according to launch site (coordinate).
- Expanding a cluster, we see the individual launches color-coded by class variable (success/failure).
- This is shown for the KSC launch site.





Launch site proximities on Folium Map

- The map show an example where the distance from CCAFS SLC-40 to the coast is considered.
- Close-up of proximities for KSC LC-39A is also shown.
- What make up an optimal launch site in terms of proximities?
    - Close to transport and supply lines, i.e., highway and railways.
    - Close to the coast (also partly related to transport).
    - Keep distance to cities.

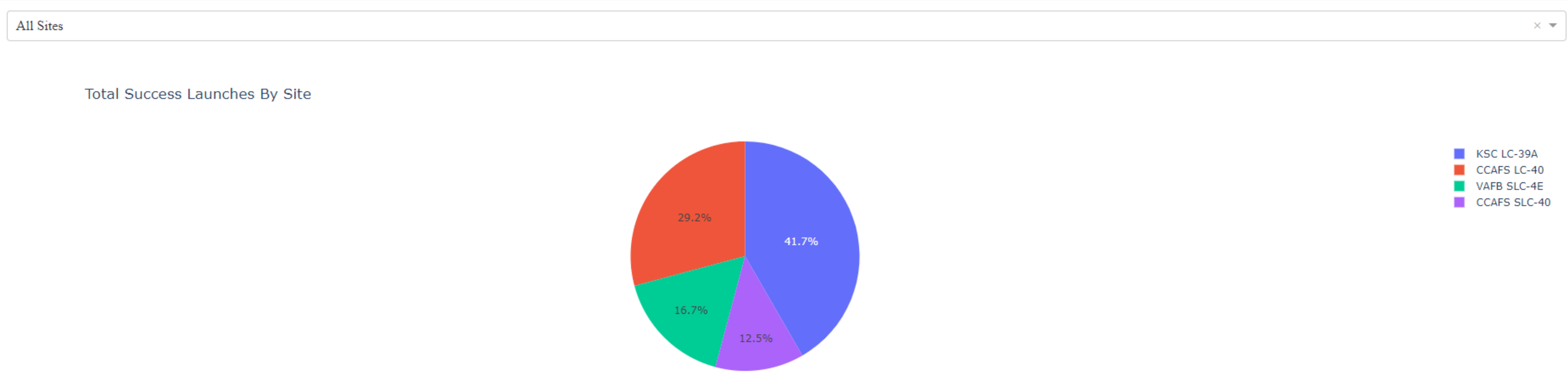


Section 4

Build a Dashboard with Plotly Dash

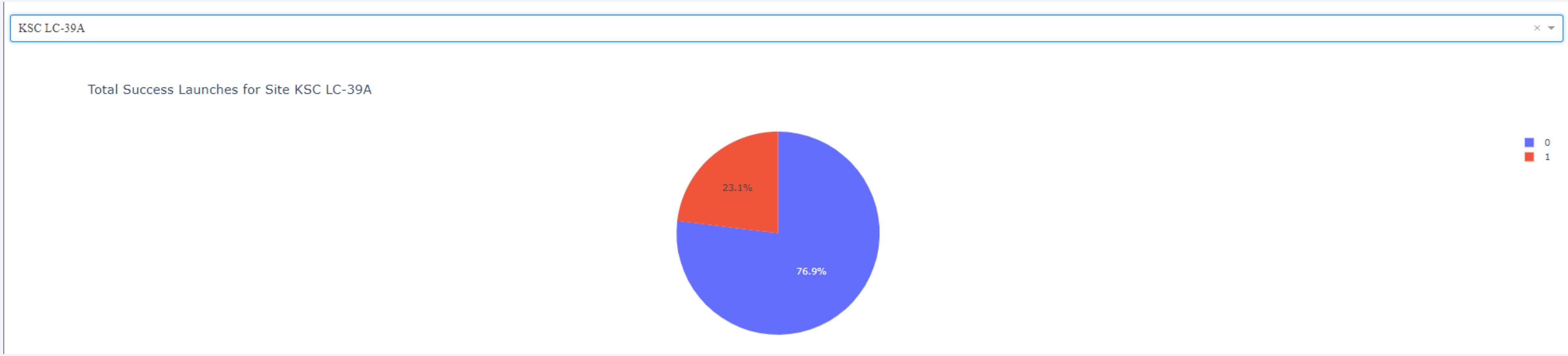
Dashboard for launch site successes – all sites

- Pie chart of launch success count for all sites
- The plot shows the distribution of successful launches over the launch sites
- We see e.g., that KSC LC-39A contribute with most of the successful launches (41.7%)



Dashboard for successful launches – KSC LC-39A

- Pie chart of launch success count for KSC LC-39A (most successful launches)
- The plot shows the distribution between successful (=1) and failed launches (=0)

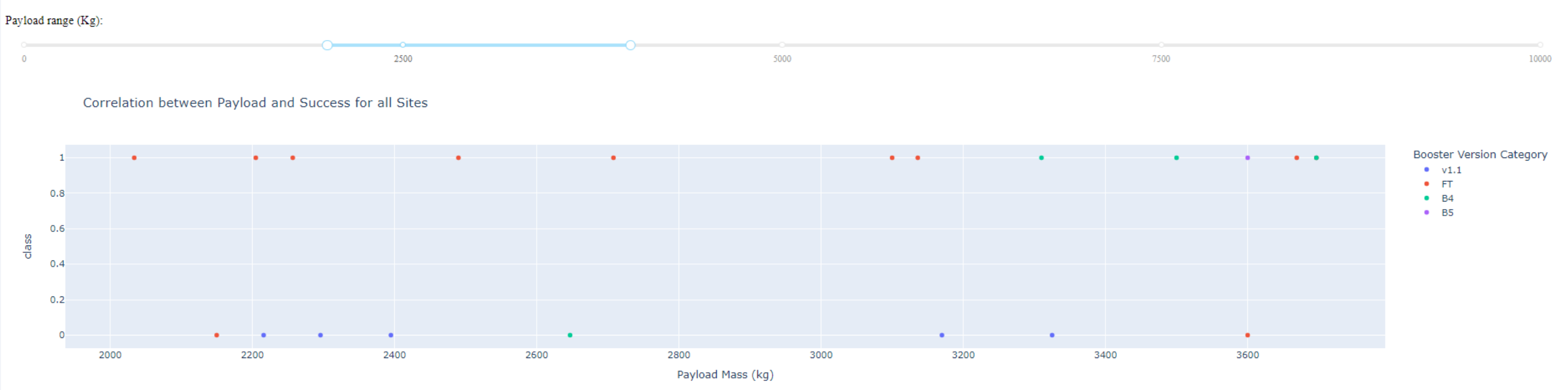


Dashboard for Payload vs. Launch Outcome (1)

- Scatterplot of Payload vs. Launch Outcome for all sites, considering;
  - The full payload range – booster version FT is the most successful over the full payload range

Close-up on the following slides

  - The payload range [2k-4k] – this payload range is the most successful (only boosters v1.1, FT, B4, B5)



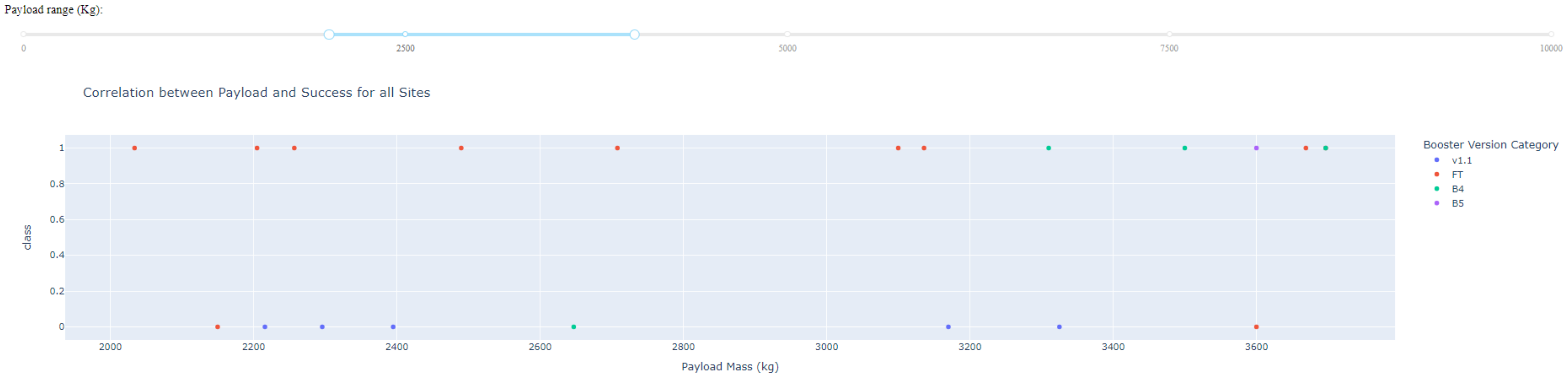
Close-up on the following slides

Dashboard for Payload vs. Launch Outcome (2)

- Scatterplot of Payload vs. Launch Outcome for all sites, considering;
  - The full payload range – booster version FT is the most successful over the full payload range

Dashboard for Payload vs. Launch Outcome (3)

- Scatterplot of Payload vs. Launch Outcome for all sites, considering;
  - The payload range [2k-4k] – this payload range is the most successful (only boosters v1.1, FT, B4, B5)

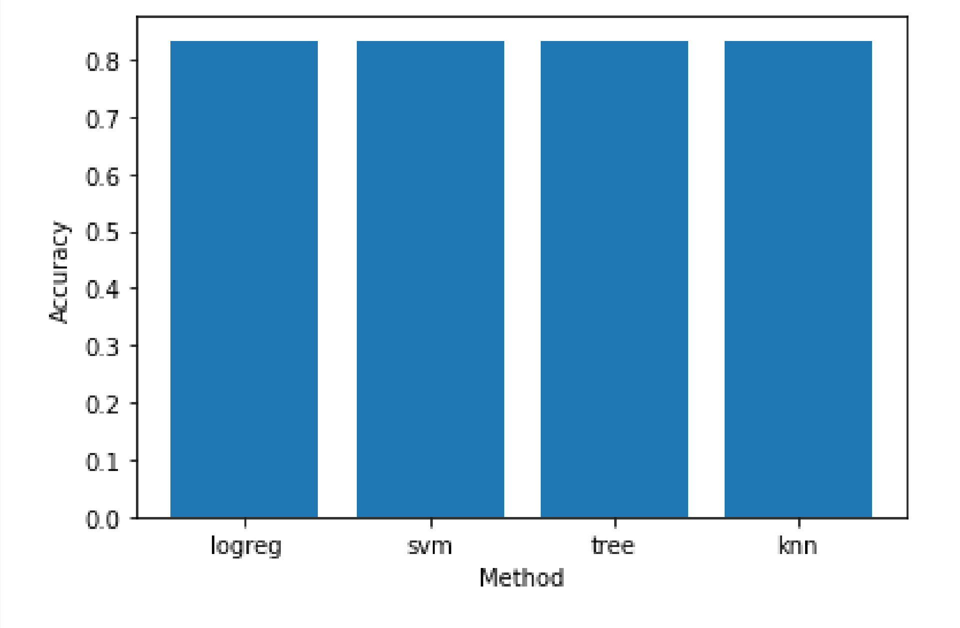


Section 5

Predictive Analysis (Classification)

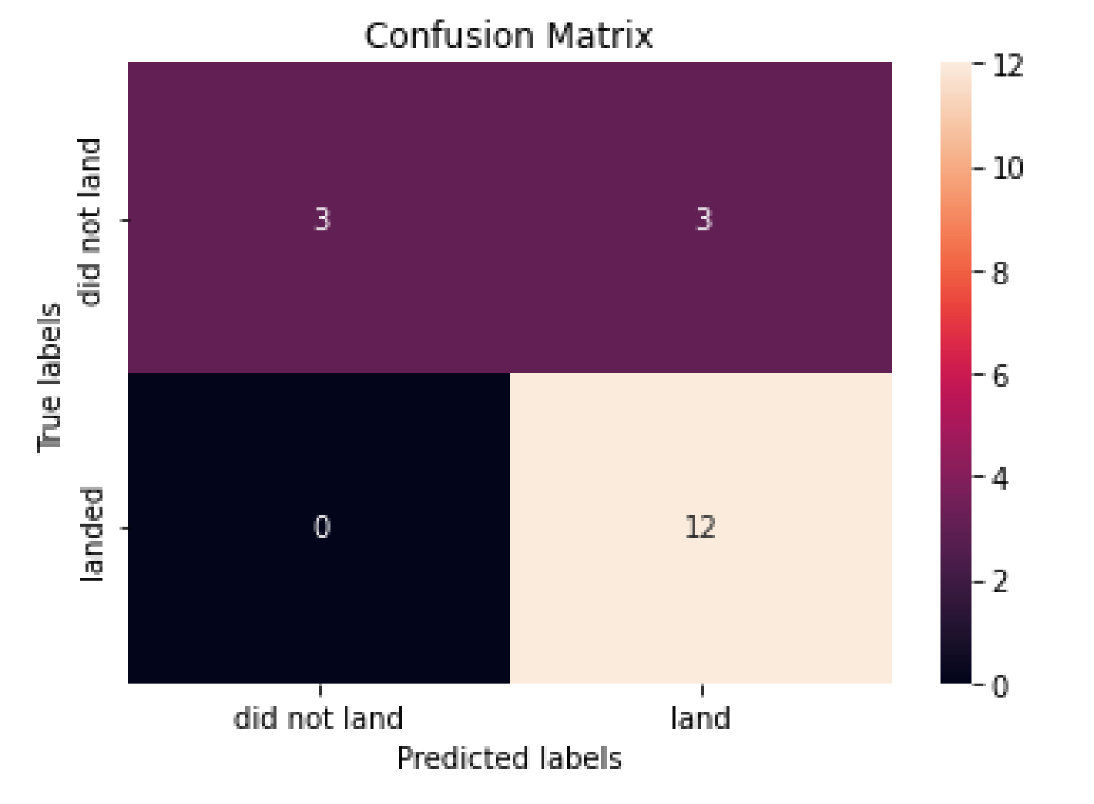
Classification Accuracy

- Bar chart of test set accuracy for the considered classifiers, i.e.,
  - Logistic regression (logreg)
  - Support vector machine (svm)
  - Decision tree (tree)
  - k-nearest-neighbors (knn)
- All classifiers performed equally well on the test set, i.e., accuracy 0.83



Confusion Matrix

- Confusion matrix for one of the models – all models resulted in the same confusion matrix
  - The main diagonal shows the correctly classified test data points
  - The off-diagonal elements shows the incorrectly classified test data points
  - We see that the classifier makes three errors prediction a landing when in fact it did not land.
  - We also see that the data is imbalanced in the classes, i.e., more landed examples (double), which could explain the bias towards a landing prediction.



Conclusions

- The goal of the analysis was to build a machine learning model to predict whether the first stage would land, such that it could be reused to reduce the cost of a launch
    - We have compiled and analyzed a data set composed of information on SpaceX Falcon 9 launches from SpaceX API and Wikipedia page
    - Filtered, cleaned, and formatted the data
    - The data show some clear patterns in e.g., the placement of launch site and success rate dependence on payload
    - Based on the data set, we were able to fit classifiers with a test set accuracy of 0.83
    - The data set is imbalanced in the classes, which result in a bias towards a landing prediction
- A test set accuracy of 0.83 is reasonable, but we could further improve our predictions by accounting for the class imbalance in the database.

Appendix

All available/additional course material can be could on my GitHub repo for this course, see

https://github.com/SebastianGlavind/IBM-DataScienceProfessionalCertificate/tree/main/AppliedDataScienceCapstone

Thank you!