Winning Space Race with Data Science

Alam Katon Agung
Jan 2, 2022

Outline

Executive Summary

- SpaceY is a new commercial rocket launch provider who wants to bid against SpaceX.
- SpaceX advertises launch services starting at $62 million for missions that allow some fuel to be reserved for landing the 1st stage rocket booster, so that it can be reused.
- SpaceX public statementsindicate a 1st stage Falcon 9 booster to cost upwards of $15 million to build without including R&D cost recoupment or profit margin.
- Given mission parameters such as payload mass and desired orbit, the models produced in this report were able to predict the first stage rocket booster landing successfully with an accuracy level of 83.3%.
- As a result, SpaceY will be able to make more informed bids against SpaceX by using 1st stage landing predictions as a proxy for the cost of a launch.

Introduction

Project background and context;

Here we will predict 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 each, much of the savings is because SpaceX can reuse the first stage. Therefore, if we can determine if the first stage will land successfully. This information can be used if an alternate company wants to bid aganst SpaceX for a rocket lauch.

Problem we want to find answer:

- With what factor, the rocket will land successfully?
- The effect of each relationship of rocket variable on outcome.
- Conditions which will aid SpaceX have to achieve the best results.

Section 1

Methodology

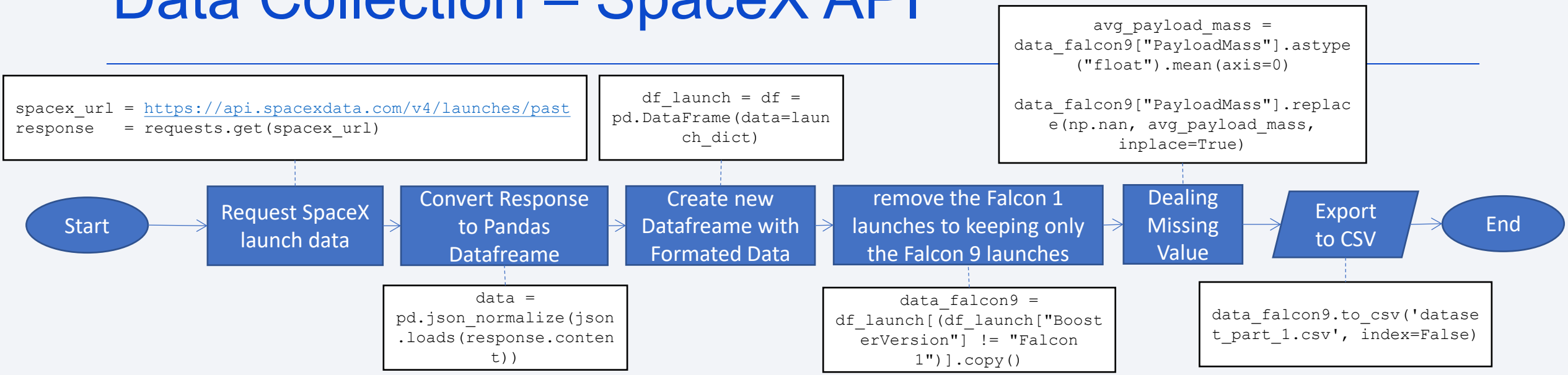
Methodology

- Data collection methodology:
  - Data was collected by requesting to SpaceX API and Wabscraping to Wikipedia
- Perform data wrangling
  - One hot encoding data fields for machine learning and dropping irrelevant columns (Transforming data for Machine Learning)
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Scatter and bar graphs to show patterns between data
- Perform interactive visual analytics using Folium and Plotly Dash
  - Using Folium and Plotly Dash Visualizations
- Perform predictive analysis using classification models
  - Build and evaluate classification models

Data Collection

Data collection is the process of gathering and measuring information on targeted variables.

Data Collection – SpaceX API



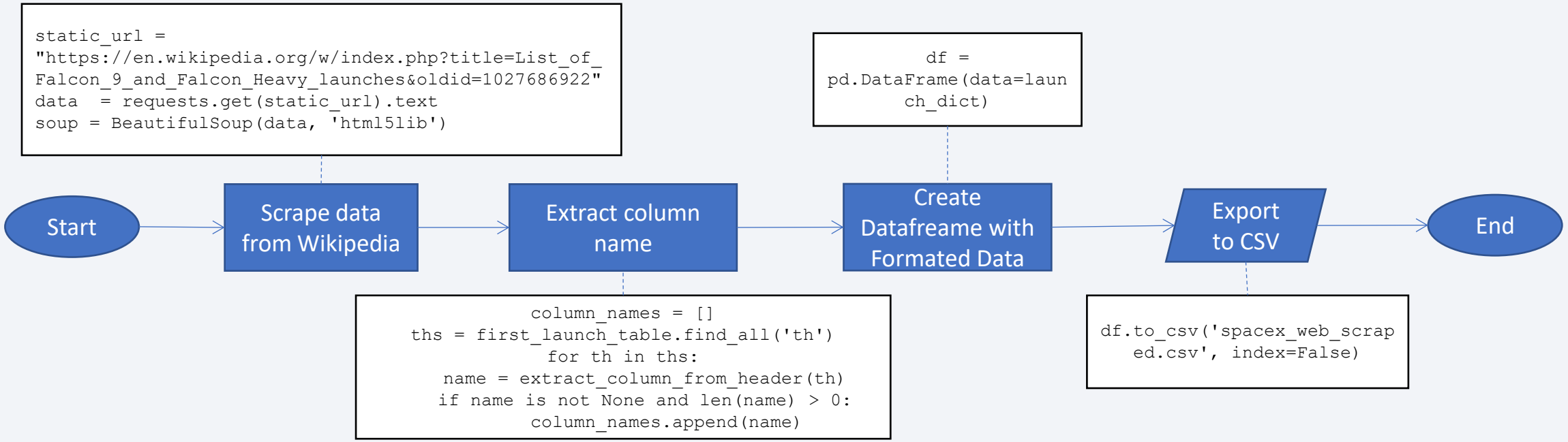
Github URL = https://github.com/alamkaton/coursera-capstone-project/blob/master/1.%20Data%20Collection%20with%20Web%20API.ipynb

In [24]:
```
# Show the head of the dataframe
df.head()
```

Out[24]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin1A	167.743129	9.047721
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2A	167.743129	9.047721
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2C	167.743129	9.047721
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin3C	167.743129	9.047721
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857

Data Collection - Scraping



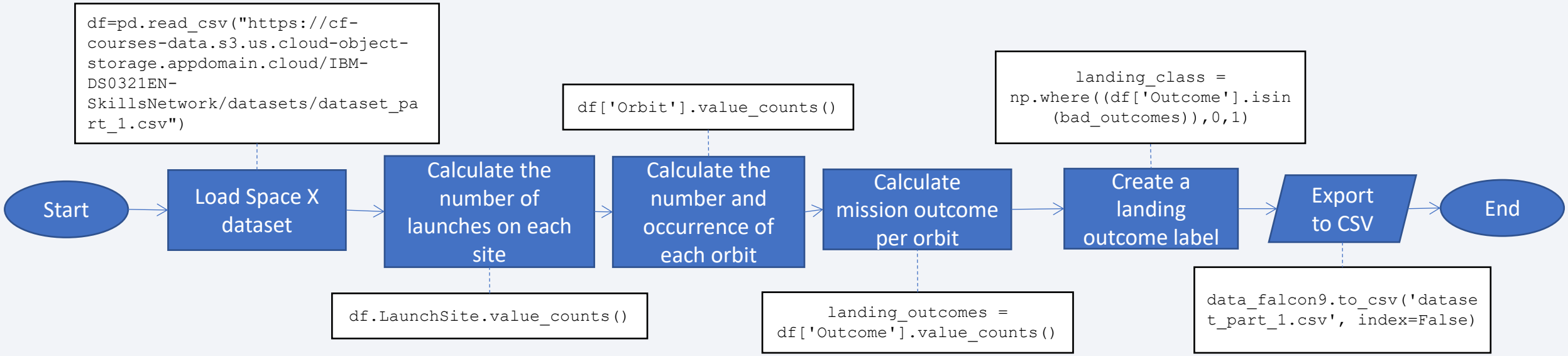
https://github.com/alamkaton/coursera-capstone-project/blob/master/2.%20Data%20Collection%20with%20Web%20Scraping.ipynb

```
In [14]: df=pd.DataFrame(launch_dict)
         df.head()
```

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	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10

Data Wrangling



https://github.com/alamkaton/coursera-capstone-project/blob/master/3.%20Data%20Wrangling.ipynb

```
In [29]: df.head(5)
```

Out[29]:

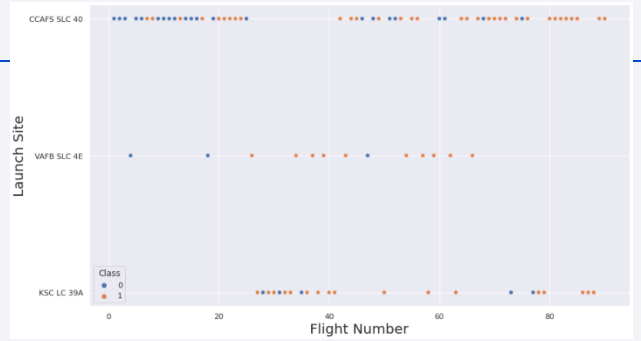
	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	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	28.561857	0
3	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	34.632093	0
4	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	28.561857	0

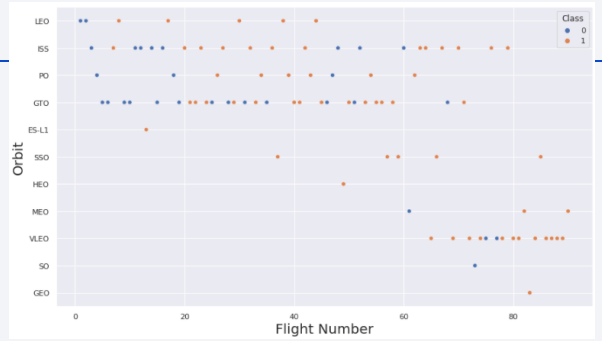
EDA with Data Visualization

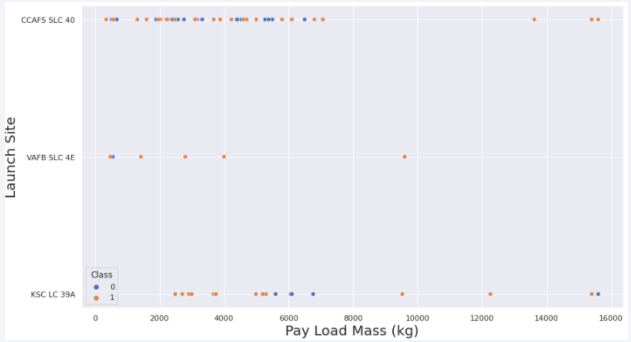
Scatter Graphs:

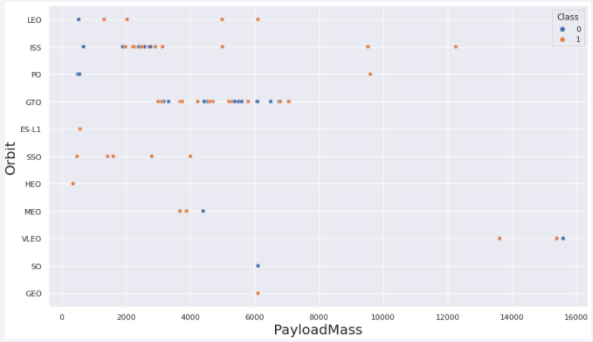
- Payload and Flight Number
- Fligh Number and Launch Site
- Payload and Launch Site
- Flight Numer and Orbit Type
- Payload and Orbit Type

Scatter plots show dependency of attributes on each other. Once a pattern is determined from the graphs it's very easy to predict which factor will lead to maximum probability of success in both outcome and landing.





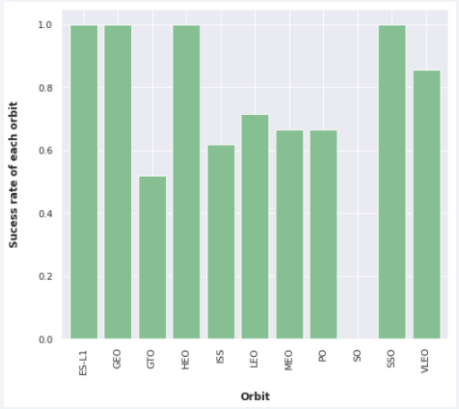




Bar Graphs:

- Success Rate and Orbit Type

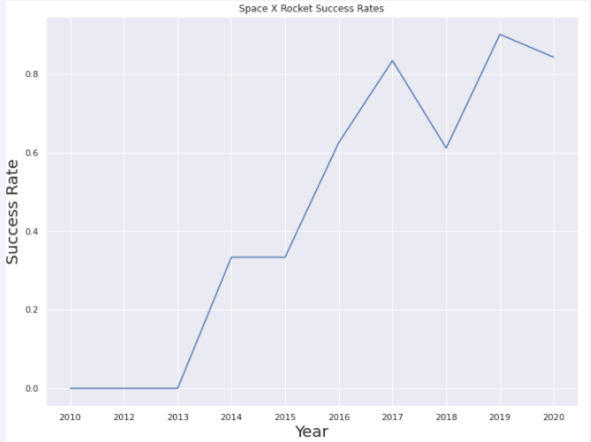
Bar graphs are esiest to interpret a relationship between attributes. Via this bar graph we can easily determine which orbits have the highest probability of success.



Line Graphs:

- Yearly success launch trend

Line graphs are useful in that they show trends clearly and can aid in predictions for the future.



https://github.com/alamkaton/coursera-capstone-project/blob/master/3.%20Data%20Wrangling.ipynb

EDA with SQL

We use IBM DB2 as a Database Service

Db2 is a family of data management products, including database servers, developed by IBM. They initially supported the relational model, but were extended to support object–relational features and non-relational structures like JSON and XML.

SQL have been performed

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA‘
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first successful landing outcome in ground pad was acheived.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the 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
- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- 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

`https://github.com/alamkaton/coursera-capstone-project/blob/master/5.%20EDA%20with%20SQL.ipynb`

Build an Interactive Map with Folium

folium makes it easy to visualize data that's been manipulated in Python on an interactive leaflet map. It enables both the binding of data to a map for choropleth visualizations as well as passing rich vector/raster/HTML visualizations as markers on the map.

The library has a number of built-in tilesets from OpenStreetMap, Mapbox, and Stamen, and supports custom tilesets with Mapbox or Cloudmade API keys. folium supports both Image, Video, GeoJSON and TopoJSON overlays.

Map Objects	Code	Result
Map Marker	folium.Marker()	Map object to make a mark on map.
Icon Marker	folium.icon()	Create an icon on map.
Circle Marker	folium.Circle()	Create a circle where Marker is being placed.
Polyline	folium.Polyline()	Create a line between points.
Marker Cluster Object	MarkerCluster()	This is a good way to simplify a map containing many markers having the same coordinate.

https://github.com/alamkaton/coursera-capstone-project/blob/master/6.%20Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

Build a Dashboard with Plotly Dash

- Pie Chart
  Showing the total success for all sites or by certain launch site.
- Scatter Plot
  Showing corelation between Payload and Success for all sites or by certain launch site.

Map Objects	Code	Result
Dash and its components	import dash import dash_html_components as html import dash_core_components as dcc from dash.dependencies import Input, Output	Plotly stewards Python's leading data viz and UI libraries. With Dash Open Source, Dash apps run on your local laptop or server. The Dash Core Component library contains a set of higher-level components like sliders, graphs, dropdown wns, table, and more. Dash provides all of the available HTML tags as user-friendly Python classes.
Pandas	import pandas as pd	Fetching values from CSV and creating a dataframe.
Plotly	import plotly.express as px	Plot the graphs with interactive plotly library.
Dropdown	dcc.Dropdown()	Create a dropdown for launch site.
Rangeslider	dcc.Rangeslider()	Create a rangeslider of Payload Mass range selection.
Pie Chart	px.pie()	Creating the Pie graph for Success percentage display.
Scatter Chart	px.scatter()	Creating the Scatter graph for correlation display.

https://github.com/alamkaton/coursera-capstone-project/blob/master/7.%20plotly_dash.py

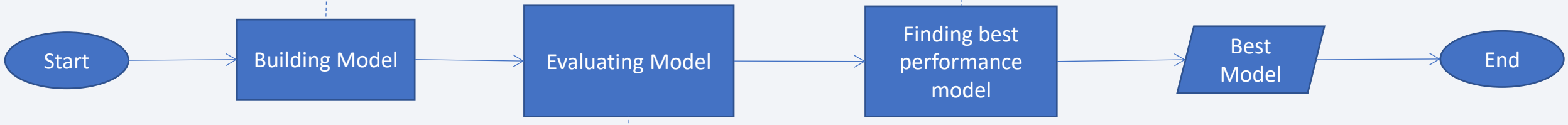
Predictive Analysis (Classification)

Building Model

- Load our feature engineered data into dataframe
- Transform it into Numpy arrays
- Standardize and transform data
- Split data into training and test data sets
- Check how many test samples has been created
- List down machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our model

```
algorithms = {
 'KNN'                 : knn_cv.best_score_,
 'Decision Tree'       : tree_cv.best_score_,
 'Logistic Regression' : logreg_cv.best_score_,
 'SVM'                 : svm_cv.best_score_
}

best_algorithm = max(algorithms, key= lambda x: algorithms[x])
```




```
yhat = logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

Cheack for each model

https://github.com/alamkaton/coursera-capstone-project/blob/master/8.%20Machine%20Learning%20Prediction.ipynb

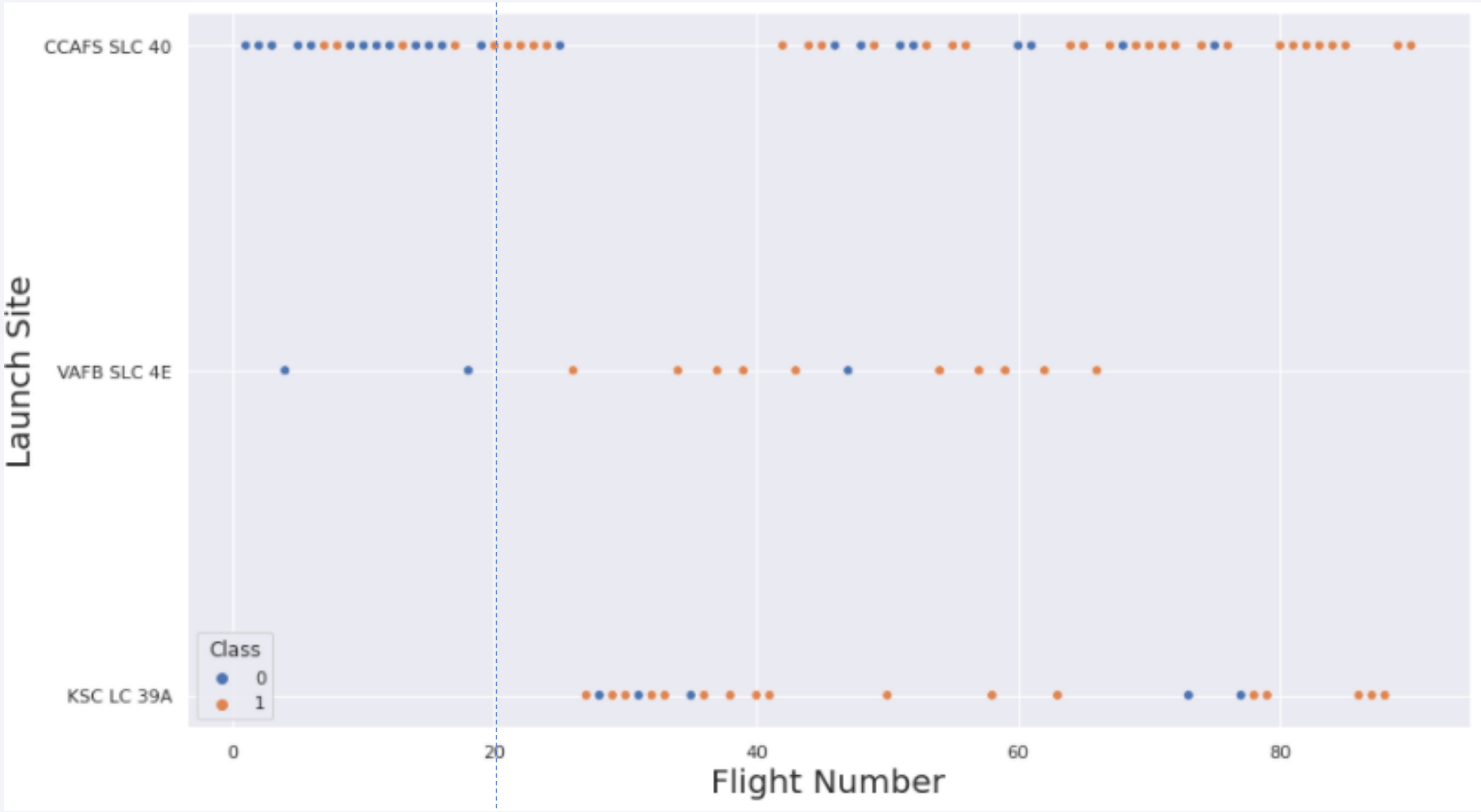
Results

- Exploratory data analysis results (17)
- Interactive analytics demo in screenshots (34)
- Predictive analysis results (38)

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site



Class

- 1 = Successful
- 2 = Unsuccessful

Flight number above 20 have more success launch.

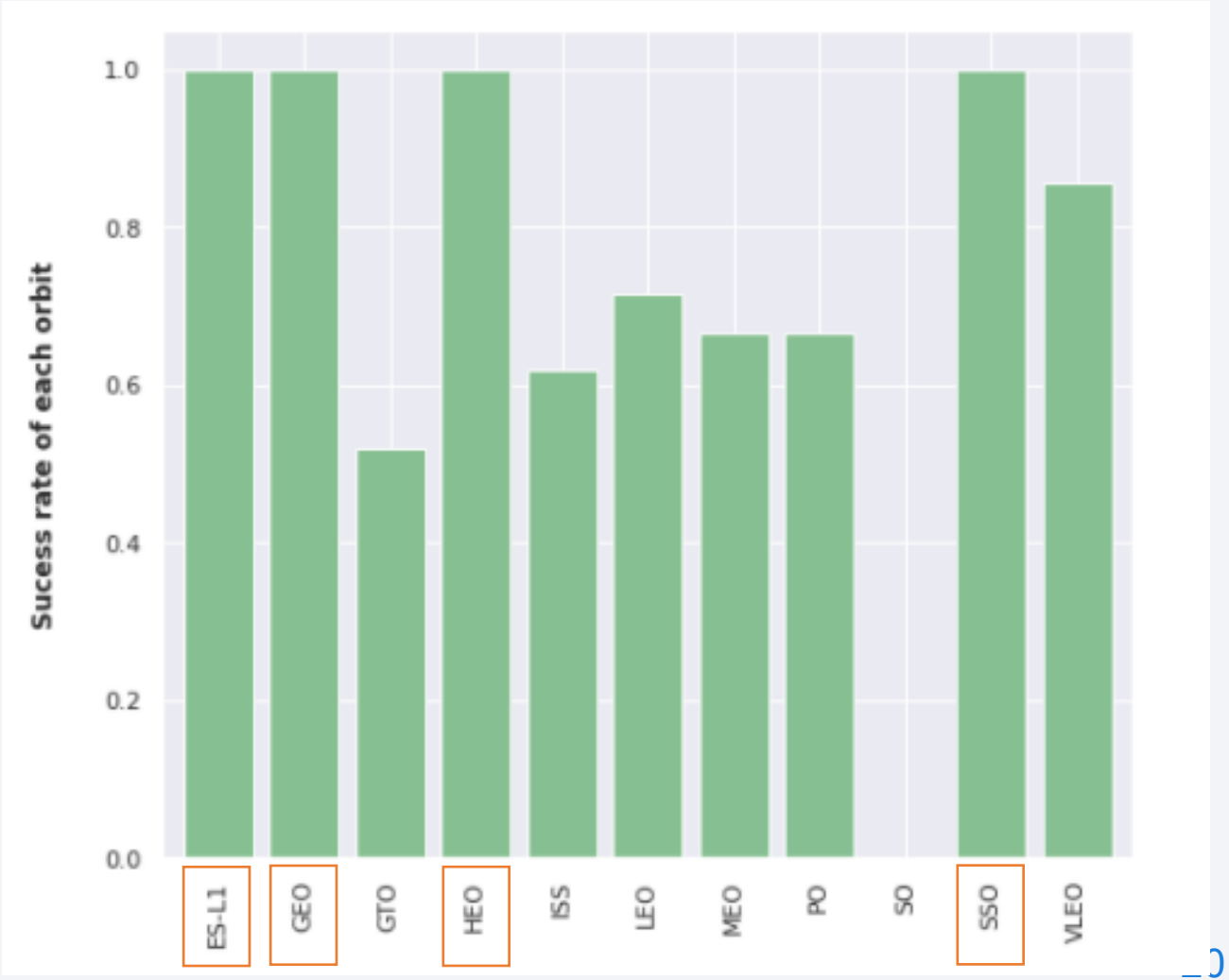
Payload vs. Launch Site

Class
- 1 = Successful
- 2 = Unsuccessful

Payload less than 4000 kg have more successful rate

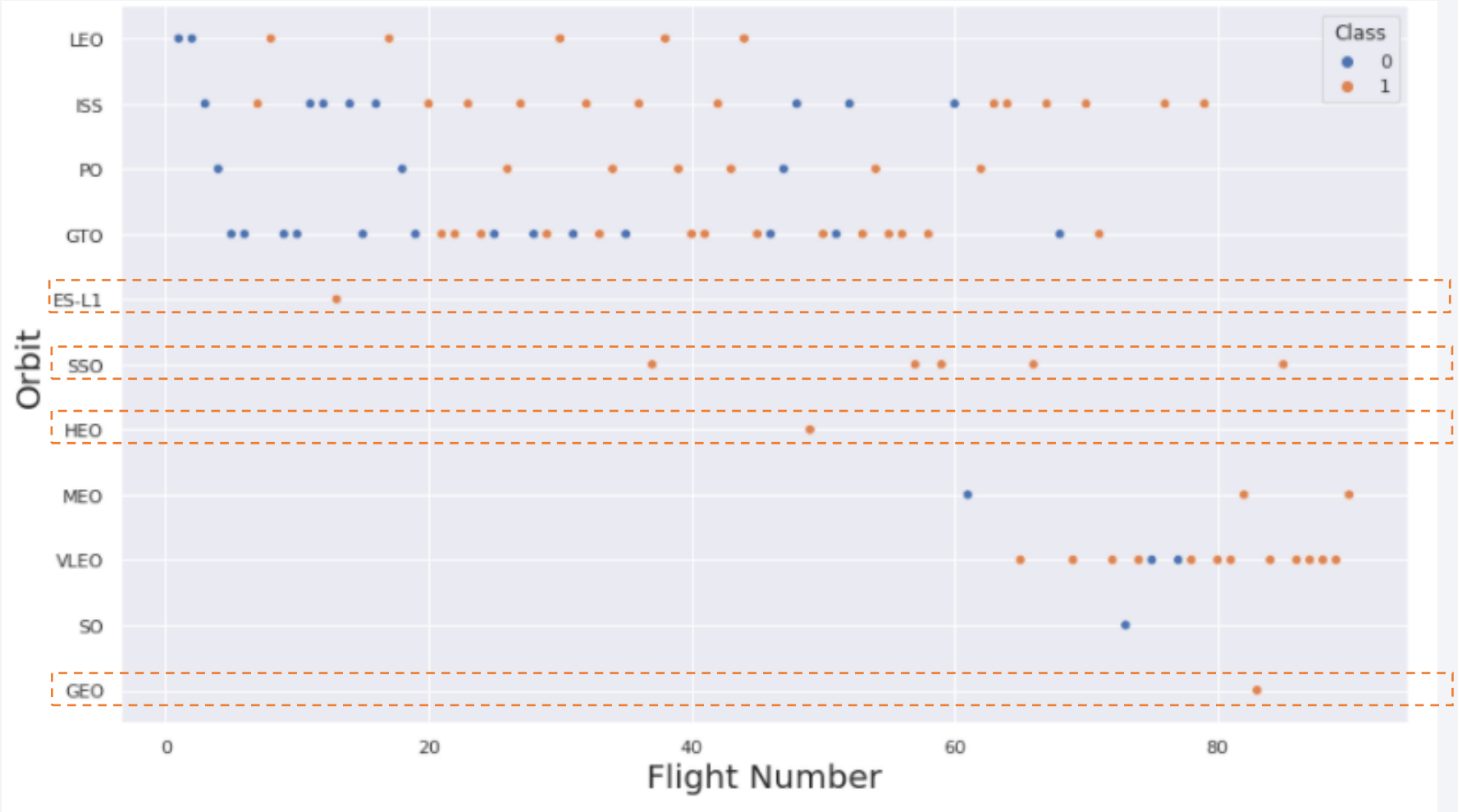
Success Rate vs. Orbit Type

ES-L1, GEO, HEO, SSO has perfect success rate



Flight Number vs. Orbit Type

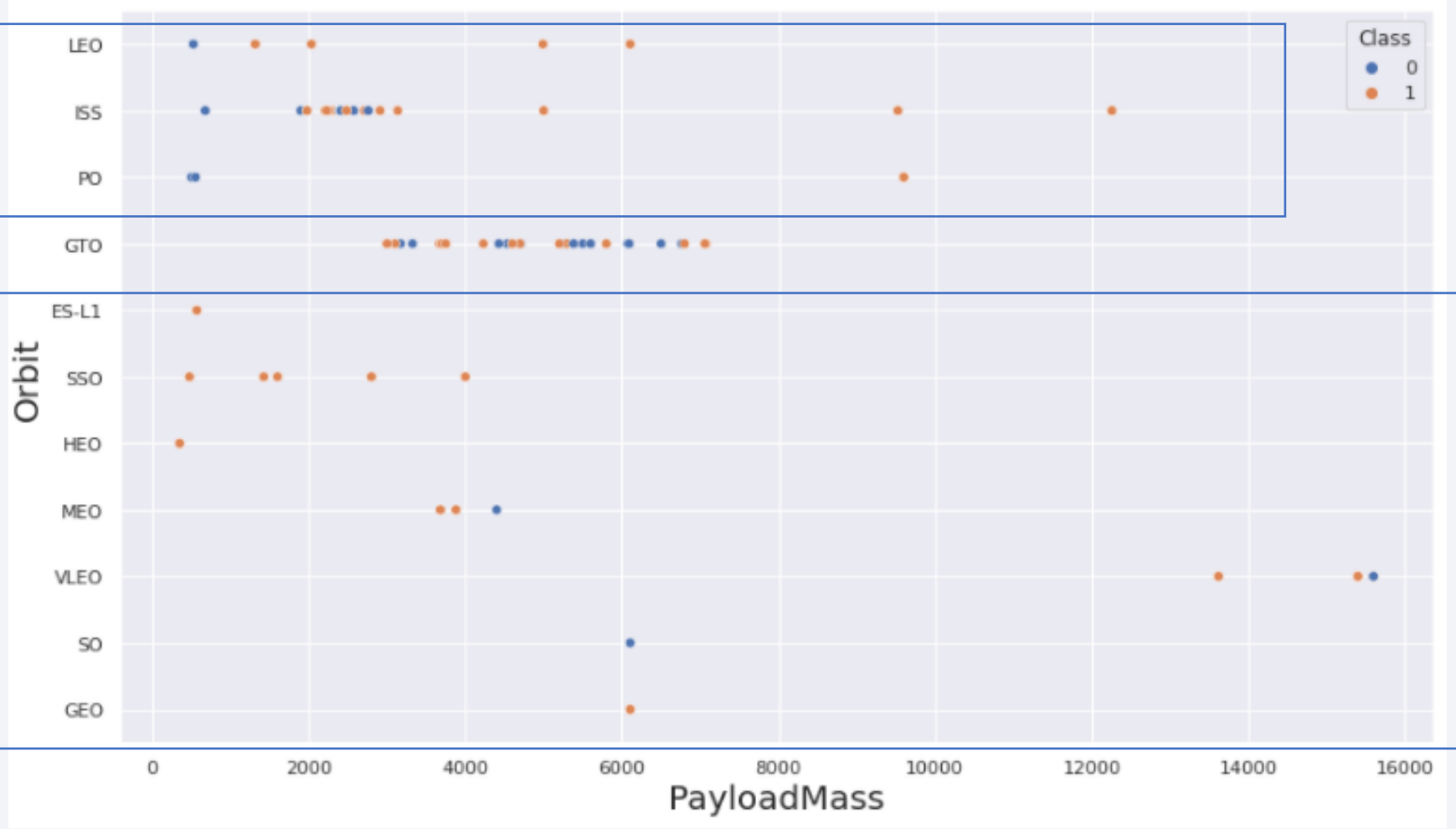
- We noticed from previous slide that ES-L1, GEO, HEO, SSO had perfect success rate.
- Validity cannot be acepted because if we caunt number of flight, it is to small compare to other orbit.



Payload vs. Orbit Type

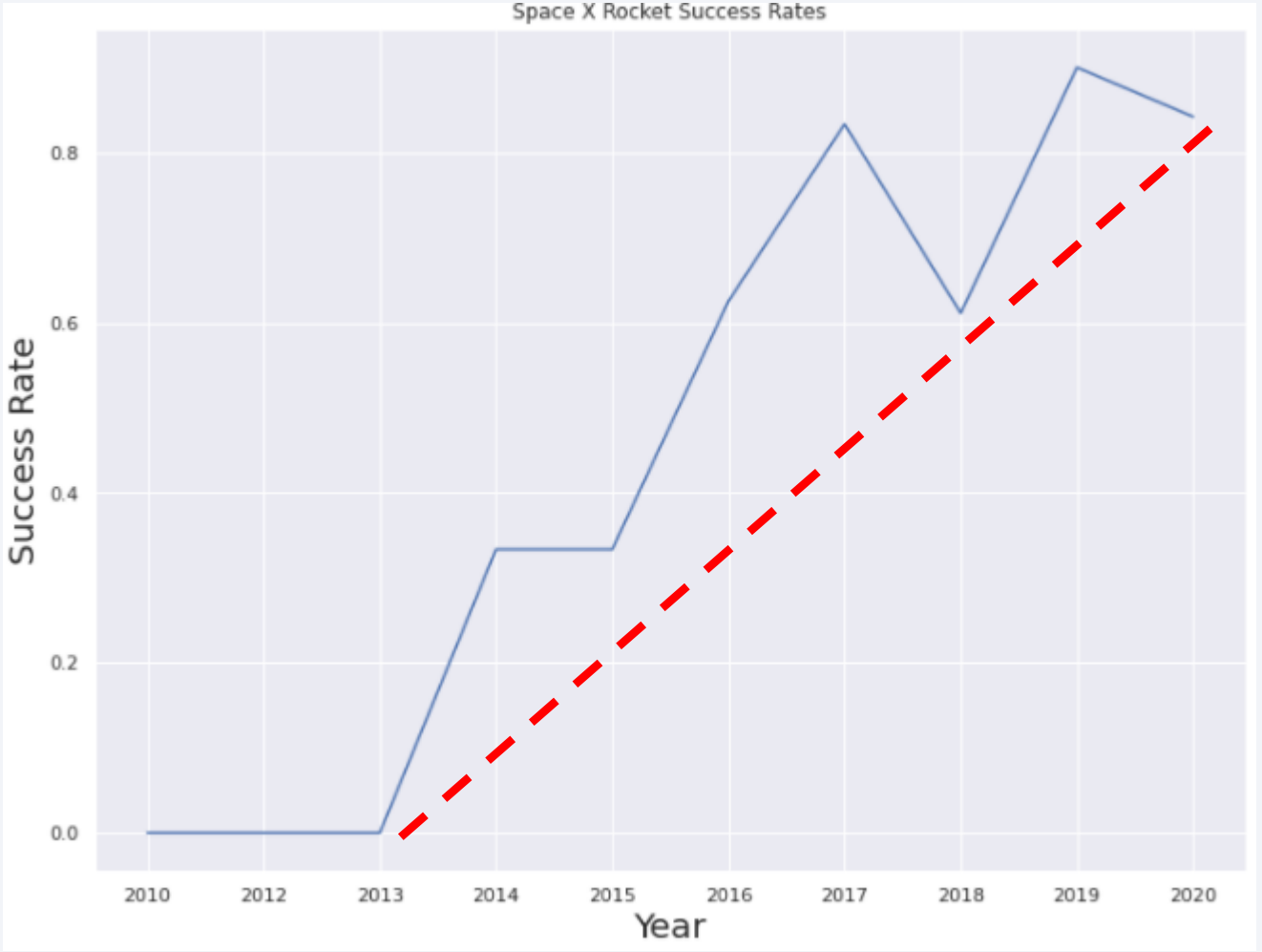
Higher payload mass, give more succes

Lower payload mass, give more succes



Launch Success Yearly Trend

Form years to years, since 2013 success rate kept increasing



All Launch Site Names

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

```
 * ibm_db_sa://rhh23184:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3
Done.
```

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

Explanation

Using keyword DISTINCT in the query we pull unique values for LAUNCH_SITE column from table SPACEXTBL

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

```
 * ibm_db_sa://rhh23184:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

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

Explanation

Using keyword ‘Limit 5’ in the query we fetch 5 records from SPACEXTBL table and with condition LIKE keyword with valid card ‘CCA’. The precentage in the end suggest that LAUNCH_SITE name must startwith ‘CCA’.

Total Payload Mass

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)" FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)';
 * ibm_db_sa://rhh23184:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/b
Done.
```

Total Payload Mass by NASA (CRS)
45596

Explanation

Using function SUM() calculates the total in the column PAYLOAD_MASS_KG _ and WHERE clause filters the data to fetch Customer ‘s by name “NASA (CRS)”.

Average Payload Mass by F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster Version F9 v1.1" FROM SPACEXTBL WHERE BOOSTER_VERSION =
'F9 v1.1';
```

```
 * ibm_db_sa://rhh23184:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Average Payload Mass by Booster Version F9 v1.1
2928

Explanation

Using function AVG() calculates the average in the column PAYLOAD_MASS_KG _ and WHERE clause filters the dataset to only perform calculations on BOOSTER_VERSION = 'F9 v1.1'

First Successful Ground Landing Date

```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad" FROM SPACEXTBL WHERE LANDING__OUTCOME = 'Success (groun
d pad)';
```

```
 * ibm_db_sa://rhh23184:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

Explanation

Using MIN() function to find minimum date in the column DATE and WHERE clause filters the dataset the data only perform calculations on LANDING__OUTCOME with values 'Success (ground pad)'

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE LANDING__OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYL
OAD_MASS__KG_ < 6000;
```

```
 * ibm_db_sa://rhh23184:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Explanation

Selecting only BOOSTER_VERSION, WHERE clause filters the dataset to LANDING_OUTCOME = 'Success (drone ship)' and clause specifies additional filter conditions PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT sum(case when MISSION_OUTCOME LIKE '%Success%' then 1 else 0 end) AS "Successful Mission", sum(case when MISSION_OUT
COME LIKE '%Failure%' then 1 else 0 end) AS "Failure Mission" FROM SPACEXTBL;
```

```
 * ibm_db_sa://rhh23184:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Successful Mission	Failure Mission
100	1

Explanation

Selecting multiple count is a complex query. I have used case clause within sub query for getting both success and failure counts in same query.

“Case when MISSION_OUTCOME like ‘%Success%’ then 1 else 0 end’ returns a Boolean value which we sum to get the result needed.

Boosters Carried Maximum Payload

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

```
 * ibm_db_sa://rhh23184:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Booster Versions which carried the Maximum Payload Mass
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

Explanation

Using max() function to calculates the maximum payload in the column PAYLOAD_MASS_KG in sub query. Where clause filters Booster Version which had that maximum payload.

2015 Launch Records

```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE DATE LIKE '2015-%' AND LANDING__OUTCOME = 'Failure (drone ship)';
```

```
 * ibm_db_sa://rhh23184:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

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

Explanation

We need to list the records which will display the month names, failure LANDING_OUTCOMES in drone ship, BOOSTER_VERSION, and LAUNCH_SITE for the months in year 2015.

Via year function we extract the year and future where clause ‘Failure (drone ship)’ fetches our required values.

Also, am using {fn MONTHNAME(DATE)} to get the Month name

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

```
%sql SELECT LANDING__OUTCOME as "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" FROM SPACEXTBL WHERE DATE BETWEEN '2
010-06-04' AND '2017-03-20' GROUP BY  LANDING__OUTCOME ORDER BY COUNT(LANDING__OUTCOME) DESC ;
```

```
 * ibm_db_sa://rhh23184:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Landing Outcome	Total Count
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

Explanation

Selecting only LANDING_OUTCOME, WHERE clause filtes the data with DATE between ‘2010-06-04’ and ‘2017-03-20’

Grouping by LANDING_OUTCOME and ordered by COUNT(LANDING_OUTCOME) in Descending order

Section 4

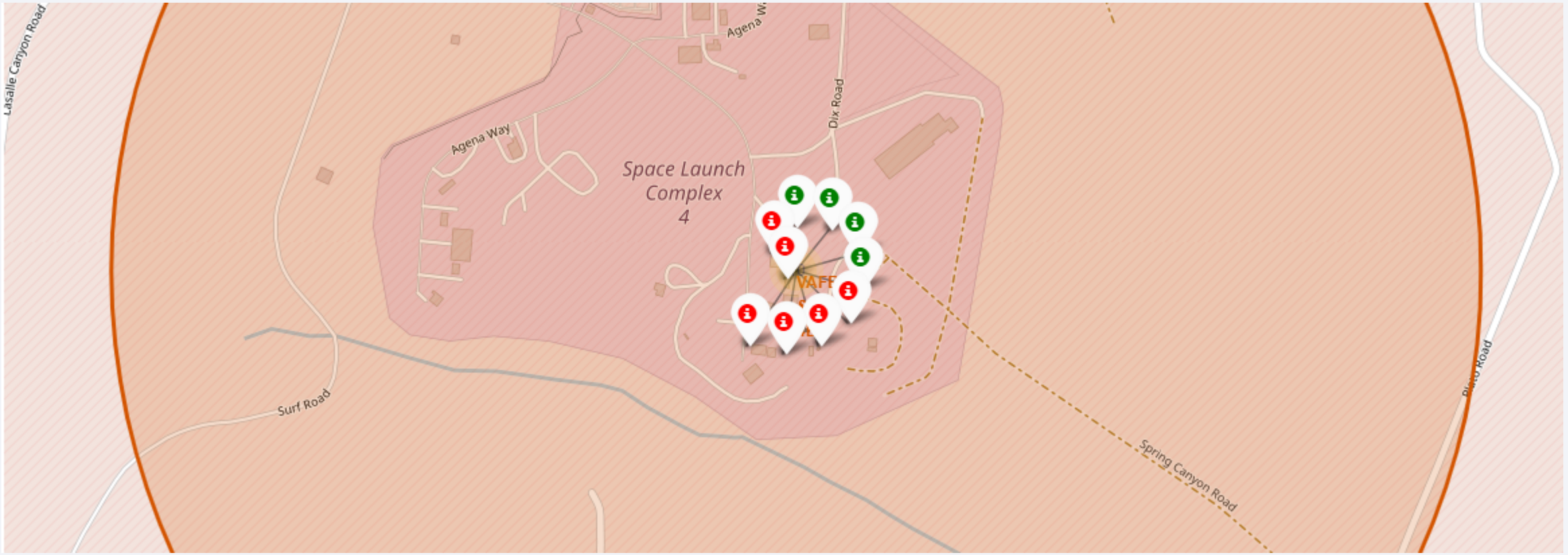
Launch Sites Proximities Analysis

All Launch Sites Location

- The SpaceX launch sites are near to the United States America coasts i.e, California and Florida Regions.

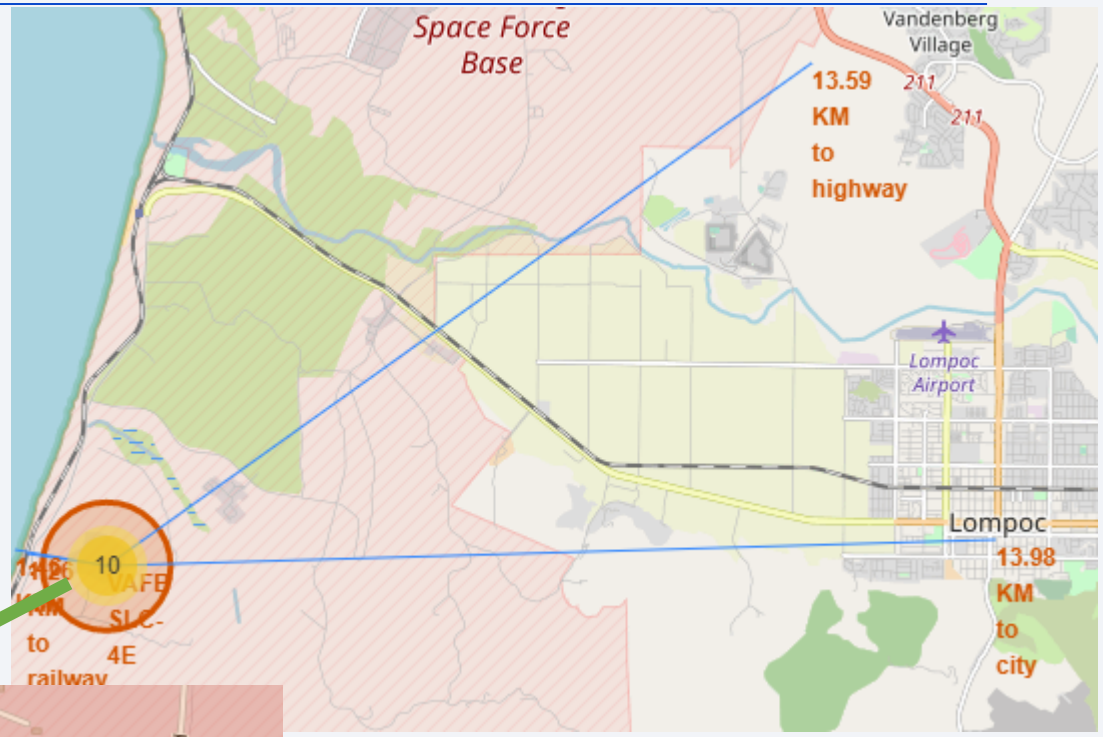
Color-labeled Launch Outcomes

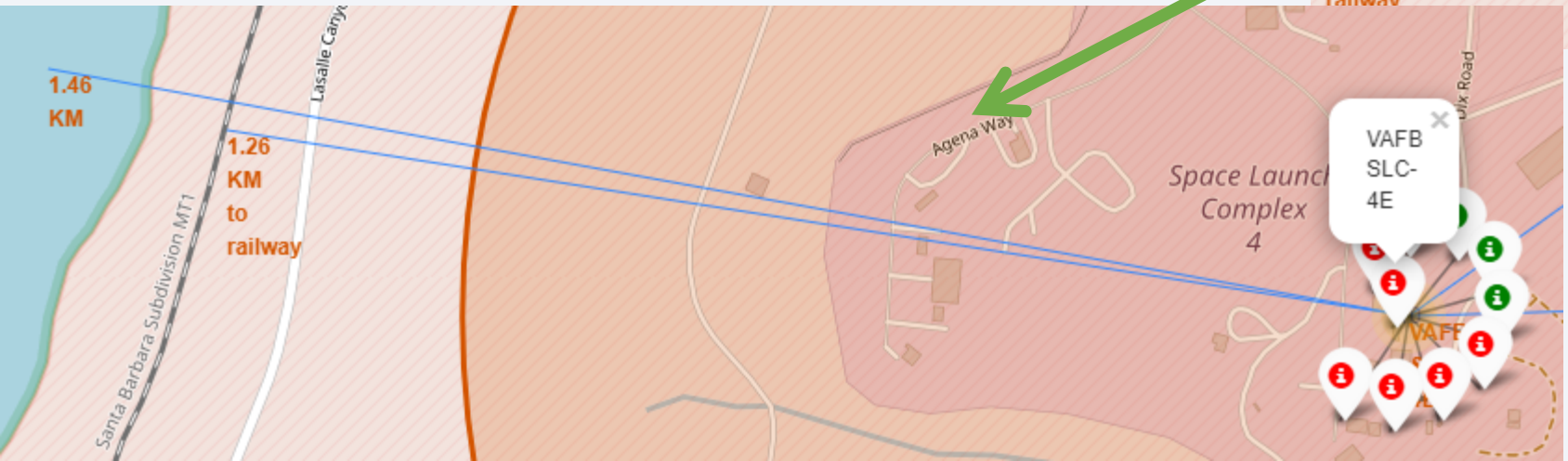
- Green marker shows successful launches
- Red Marker Shows failures

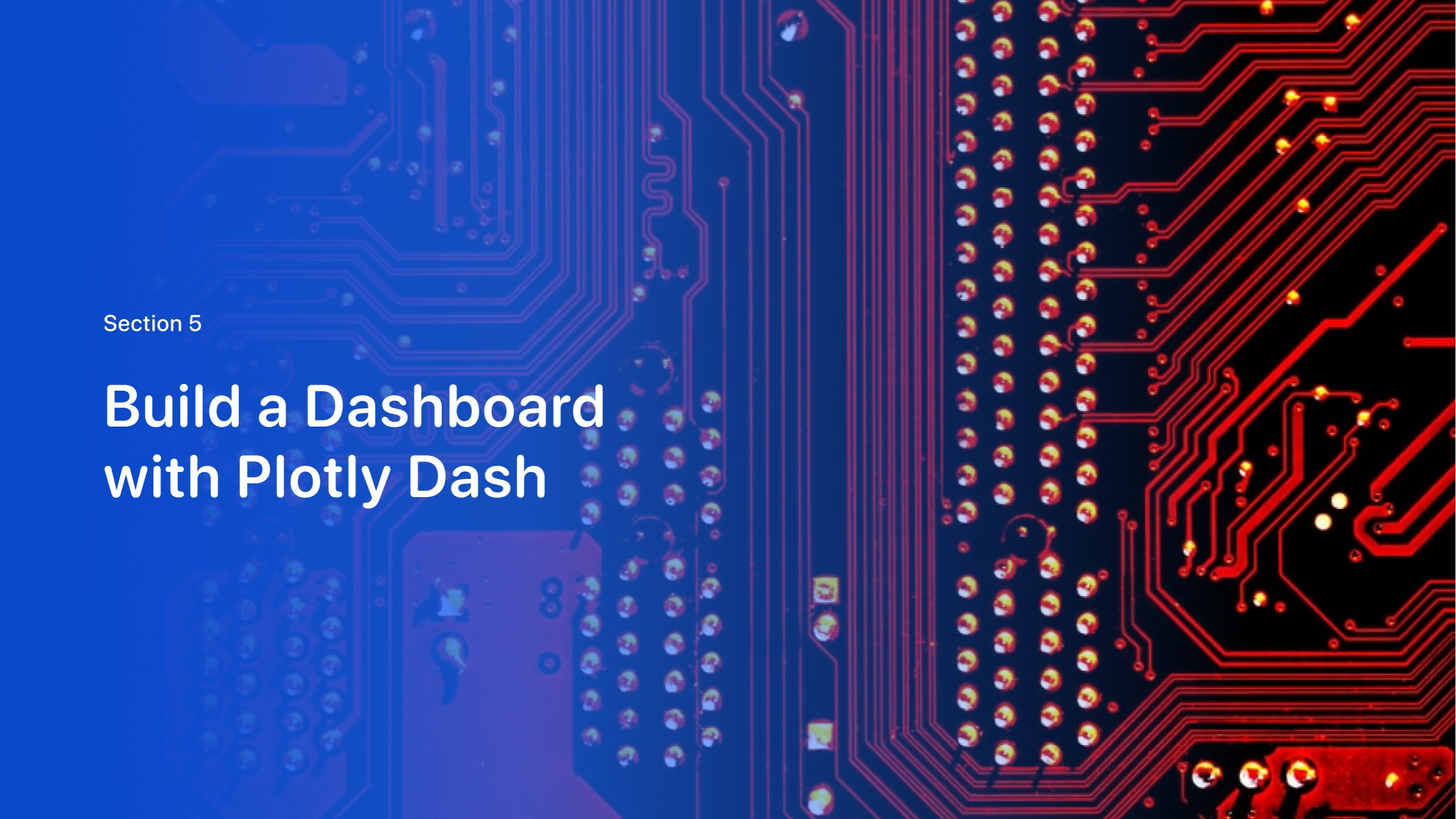


VAFB SLC-4E Launch Site Distance to to a closest City, Railway, Highway, and Coastline

- VAFB SLC-4E are 13.59 Km to Cbrillo Highway
- VAFB SLC-4E are 13.98 Km to Lompoc City
- VAFB SLC-4E are 1.26 Km to Santa Barbara Railway
- VAFB SLC-4E are 1.46 to closest Coastline



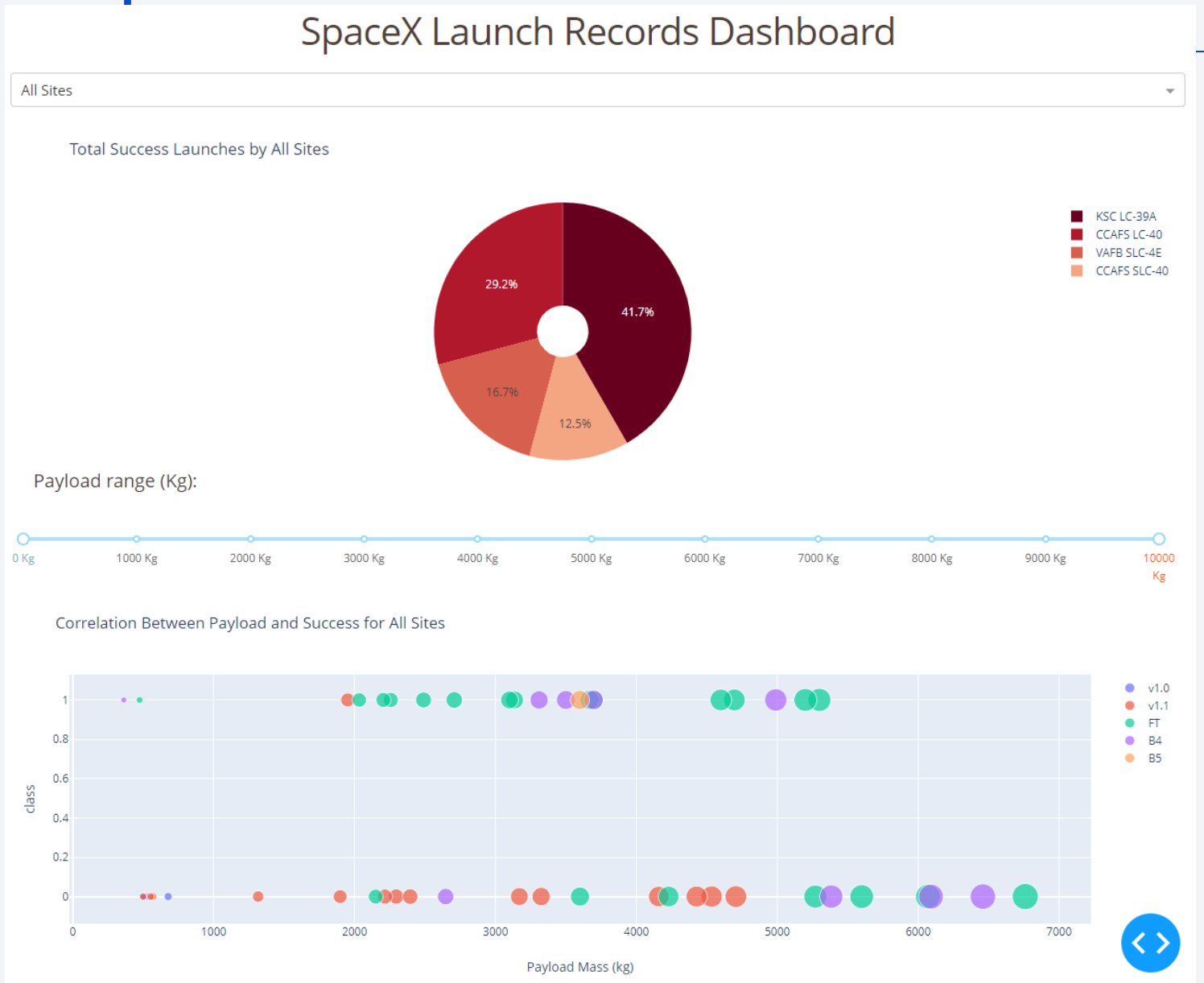




Section 5

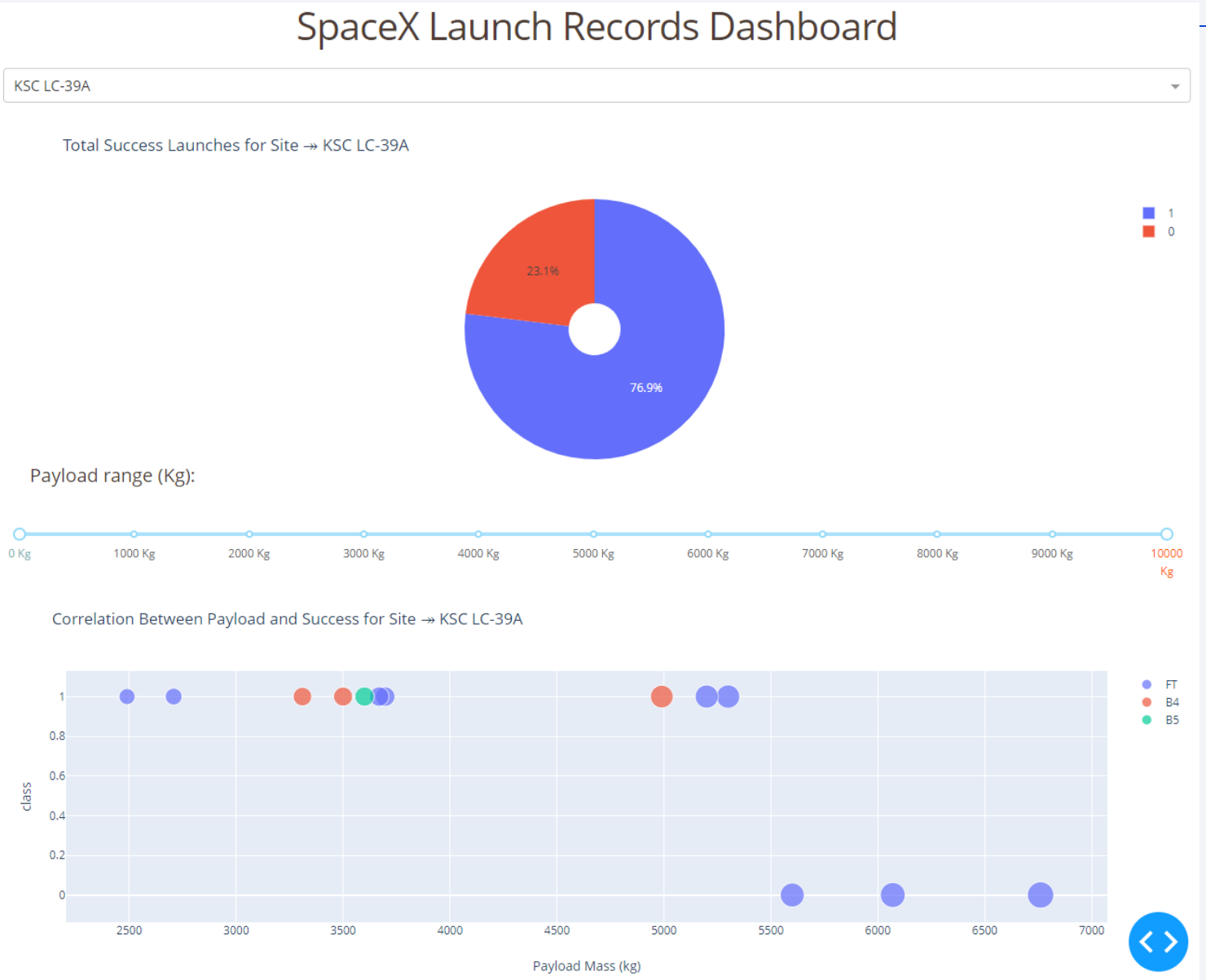
Build a Dashboard with Plotly Dash

SpaceX Launch Records Dashboard



- KSC LC-39A had the most successful launches from all the sites
- The success rate for low weighted payloads is higher than the heavy weighted payloads

The launch site with highest launch



- KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs Launch outcome

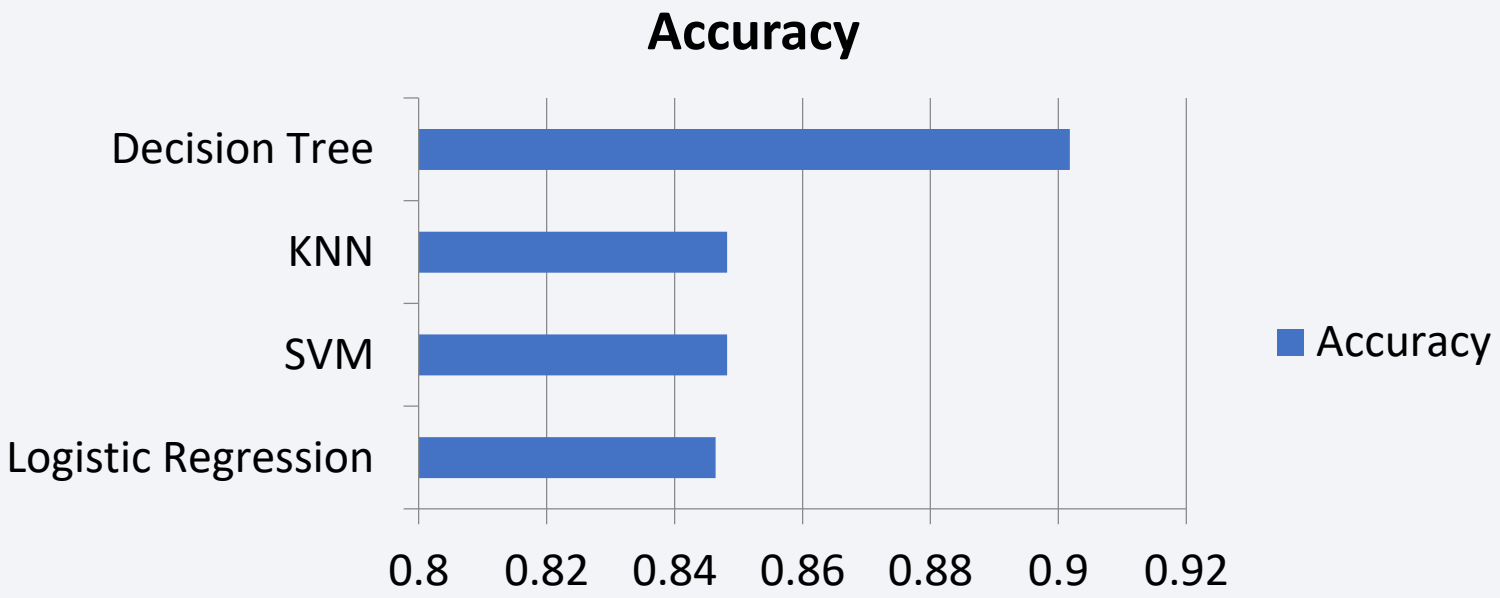
- Heavy payload have more failure result

Section 6

Predictive Analysis (Classification)

Classification Accuracy

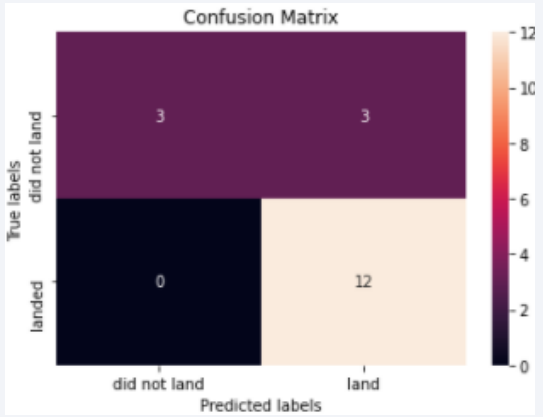
More or less all methods have same acuracy, but the winner is **Decision Tree**



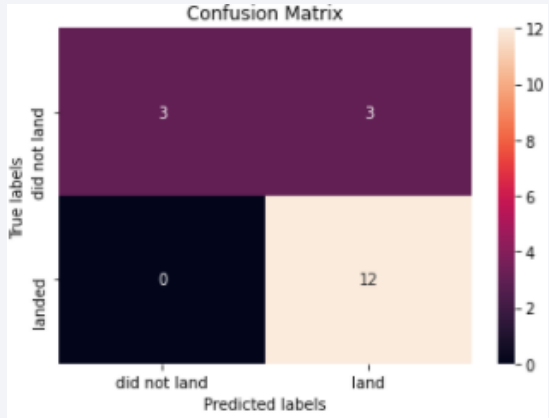
Confusion Matrix

Unfortunately all methods have same confussion matrix result.

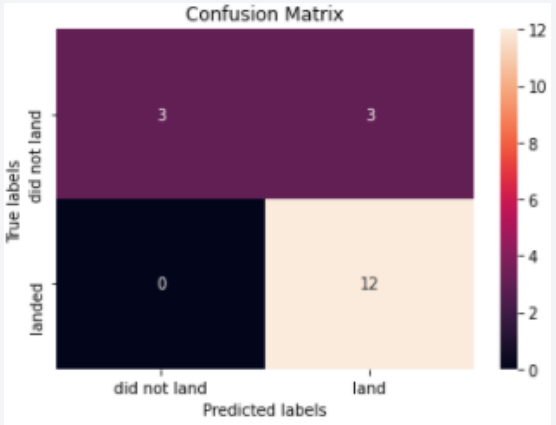
Logistic Regression



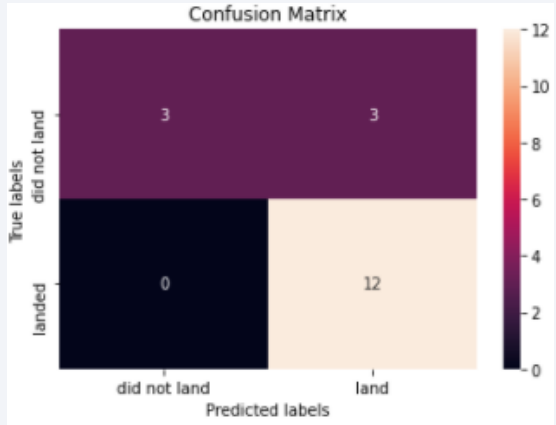
SVM



Decision Tree



KNN



Accuracy => (TP+TN) / Total => (12+3)/18 = 0.8333

Missclasification Rate => (FP+FN)/Total => (3+0)/18 = 0.1667

Precision => TP/Predicted Yes => 12/15 = 0.8

Prevalence => Actual yes / Total => 12 / 18 = 0.6667

Conclusions

- Orbits ES-L1, GEO, HEO, and SSO has highest Success rates.
- Success rates for SpaceX launches has been increasing relatively with time and it looks like soon they will reach the required target.
- KSC LC-39A had the most successful launches but increasing payload mass seems to have negative impact on success.
- Decision Tree Classifier Algorithm is the best for Machine Learning Model for provided dataset.

Appendix

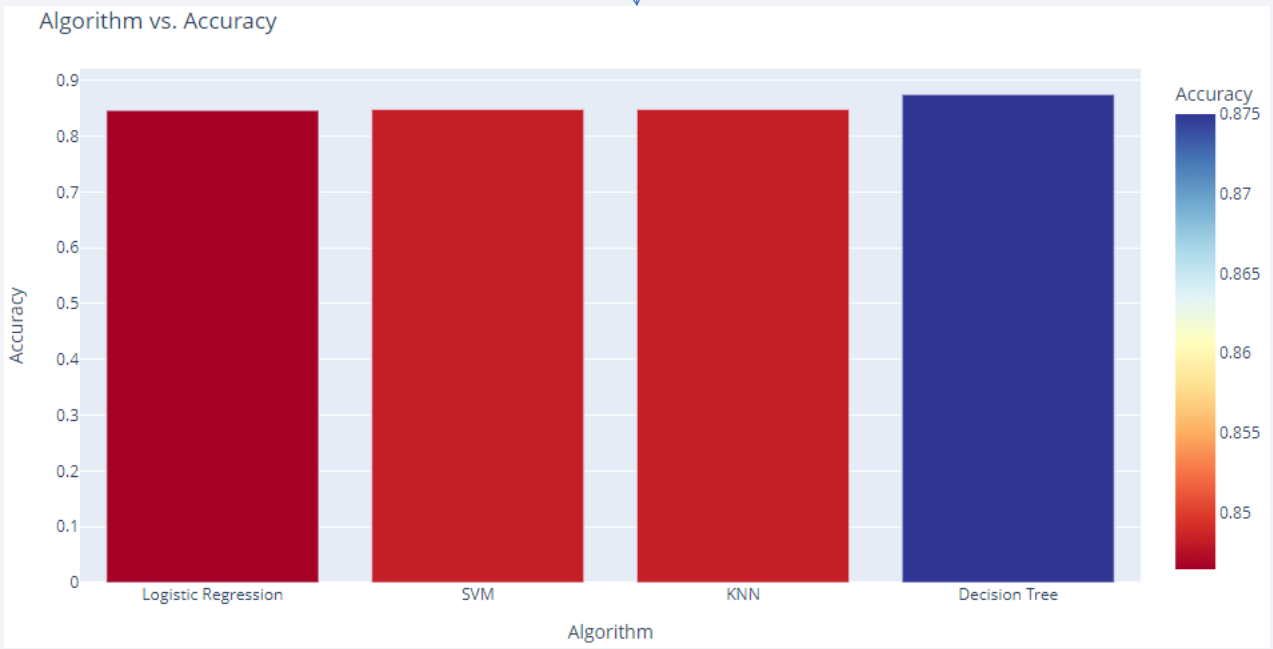
1. Get accuracy data and crete dataframe

```
algo_df
```

	Algorithm	Accuracy
0	Logistic Regression	0.846429
1	SVM	0.848214
2	KNN	0.848214
3	Decision Tree	0.875000

2. Visualize accuracy dataframe to show how much the different between all those models

```
fig = px.bar(
      algo_df,
      x='Algorithm',
      y='Accuracy',
      hover_data=['Algorithm', 'Accuracy'],
      color='Accuracy',
      color_continuous_scale='rdylbu'
)
fig.update_layout(title='Algorithm vs. Accuracy', xaxis_title='Algorithm', yaxis_title='Accuracy')
fig.show()
```



Explanation

Logistic Regression, SVM, and KNN models doesn't have pretty much different result, whereas Decision Tree gives best accuracy result with high rate.

Thank you!