Winning Space Race with Data Science

Seokhyun Yoon
21/NOV/2023

Outline

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

Executive Summary

- Using API will request the data through URL
- Using Data wragling, process the data in a way we needed
- Extract the necessary data from the dataset using SQL
- Visualized data so that audience can understand the data
- Build the interactive map
- Build the dashboard

Introduction

- The data is mixed with the other payload model such as Falcon 1
- Deliver data driven insights to determine if the first stage of Falcon 9 will land successfully

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Using SpaceX Rest API
- Perform data wrangling
  - Using API, get the response and transform from json into useful 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
  - Used a predictive model such as SVM, logistic regression and etc to find the best model

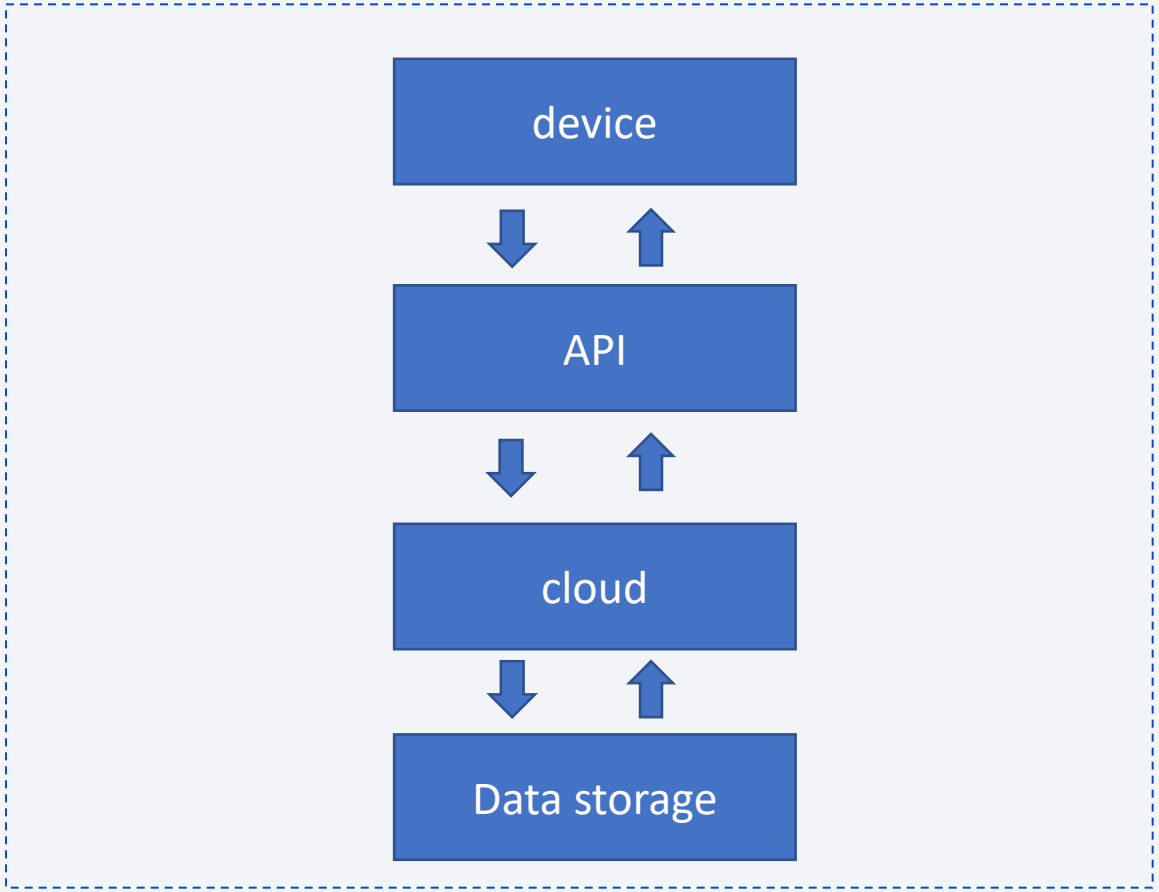
Data Collection

- Data was collected using SPACEX rest API about launches:
  - Rocket used
  - Payload delivered
  - Launch specifications
  - Landing out comes
  - And more

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

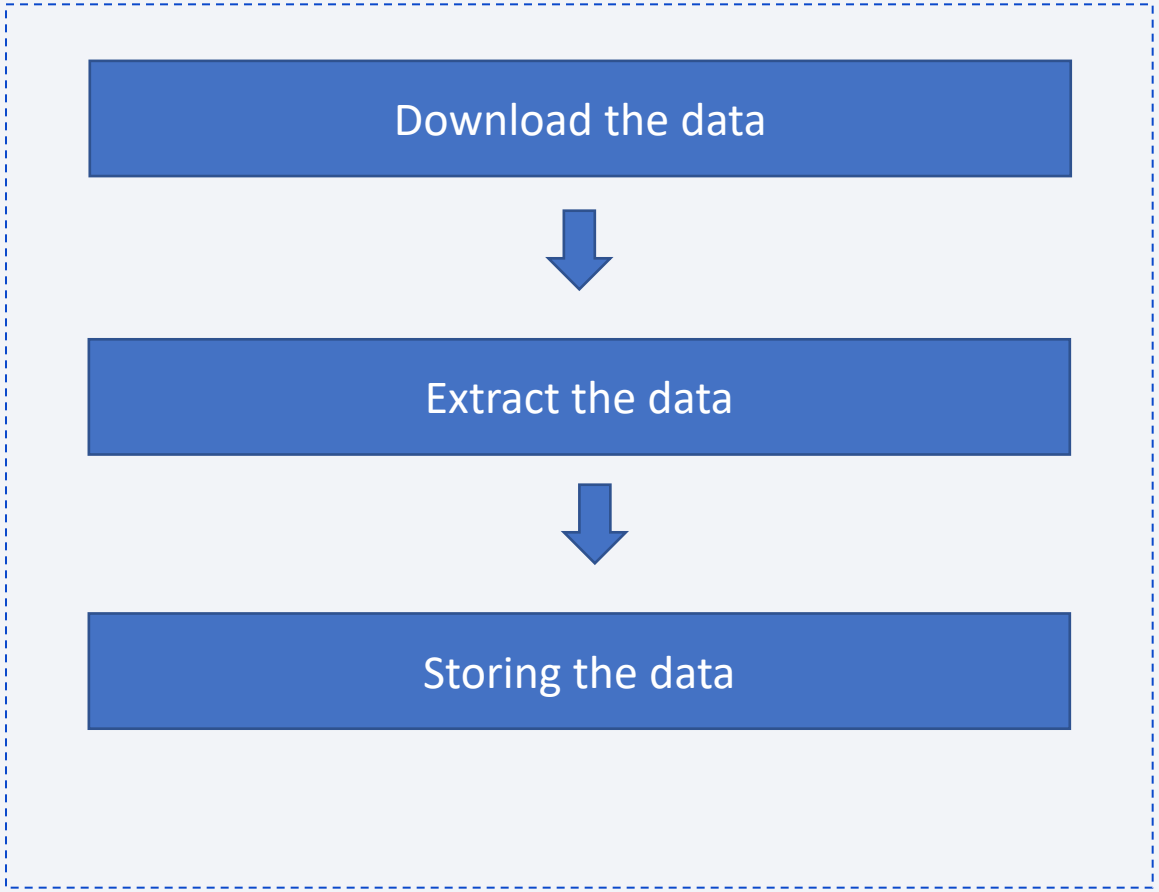
- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose



Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose



Data Wrangling

Looking at the detailed data

- Replace the null value with the mean value of the column
- Classify landing outcomes as success and failure and add the value into the new column ‘Class’
- Which helps to overlook the success rate of the rocket
- Look for a detailed understanding of each column.
  - Such as Orbit what is LEO, VLEO, GTO, etc

EDA with Data Visualization

- Scatter plot payload mass vs launch site
  - For CCAFS SLC 40, it tends to have less payload mass
  - For VAFB SLC 4E, it has only 5 points but it varies from 0 to 10000
  - For KSC LC 39A, it varies from 2000 to 16000

- Scatter plot flight number vs launch site
  - For CCAFS SLC 40, from 0-30 and from 40 it is used frequently
  - For VAFB SLC 4E only used time to time from 0 to 60
  - For KSC LC 39A started to be used around 25 and used from then

EDA with SQL

- Find the unique launch sites
- Find the launch site name beginning with “CCA”
- Find the total payload mass carried by boosters from NASA
- Find the average payload mass by F9 v1.1
- Find the dates of the first successful landing outcome on a ground pad
- Find the name of the boosters that have successfully landed on a drone ship and had payload mass greater than 4000 but less than 6000

Build an Interactive Map with Folium

- Add the Circle around the launch sites
- Add the marker to the nearest cost line (In my case, I made a marker at the middle of the ocean so that I can see the clear line)
- Add the line between the marker and one of the launch sites

Build a Dashboard with Plotly Dash

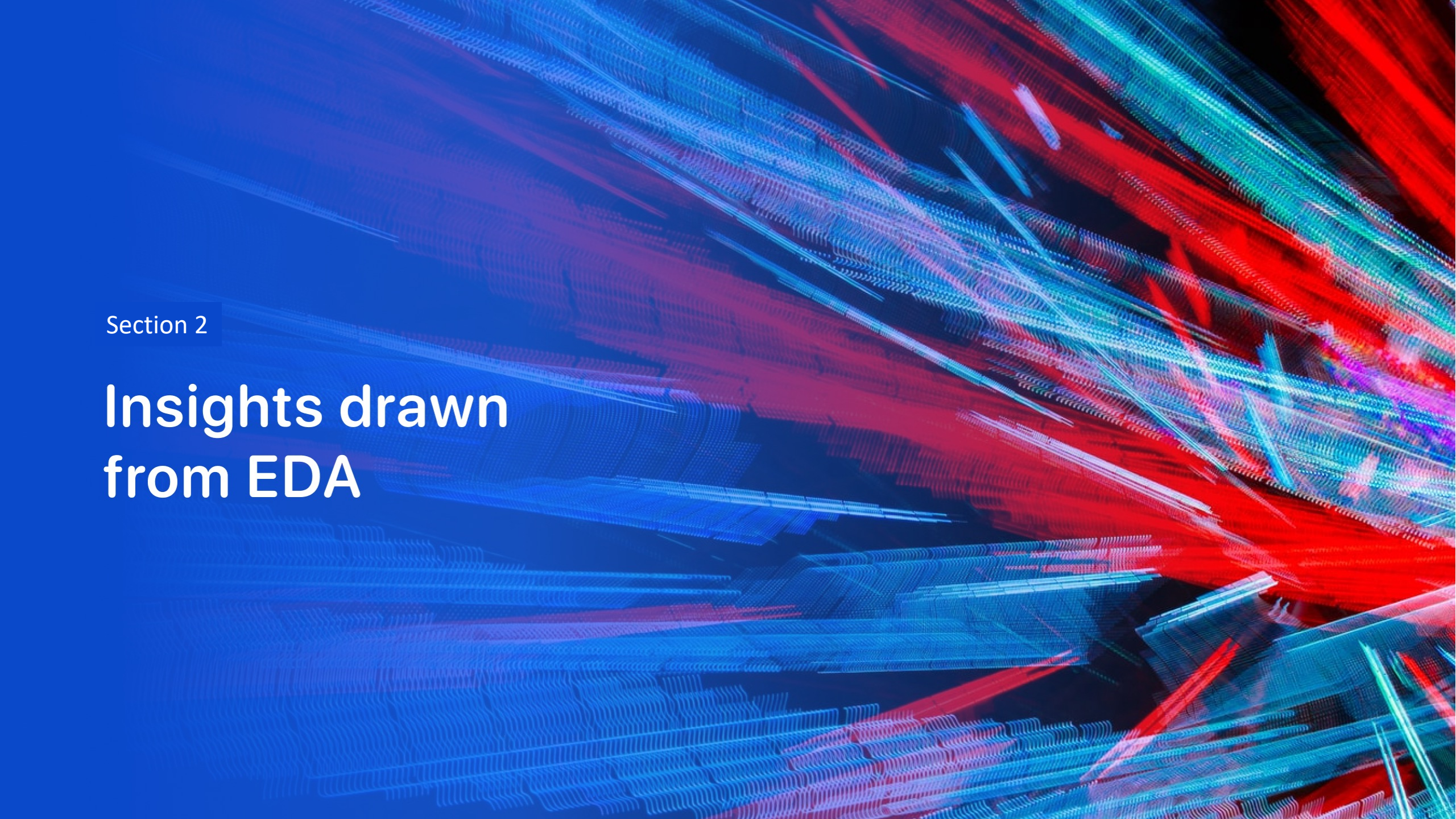
- Summarize what plots/graphs and interactions you have added to a dashboard
- Explain why you added those plots and interactions
- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

Predictive Analysis (Classification)

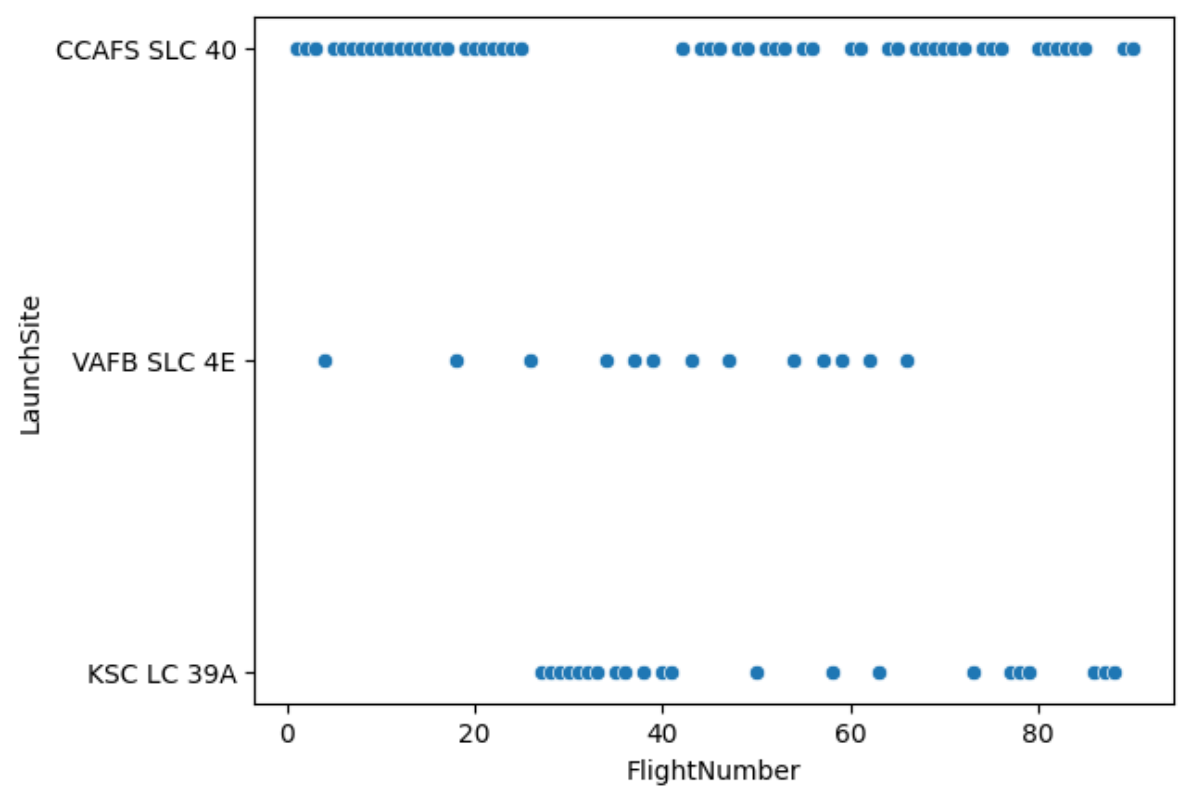
- Calculate the accuracy for each model
- Visualize the model accuracy for
  - Logistic regression model
  - Support vector machine
  - Decision tree classifier object

Results

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

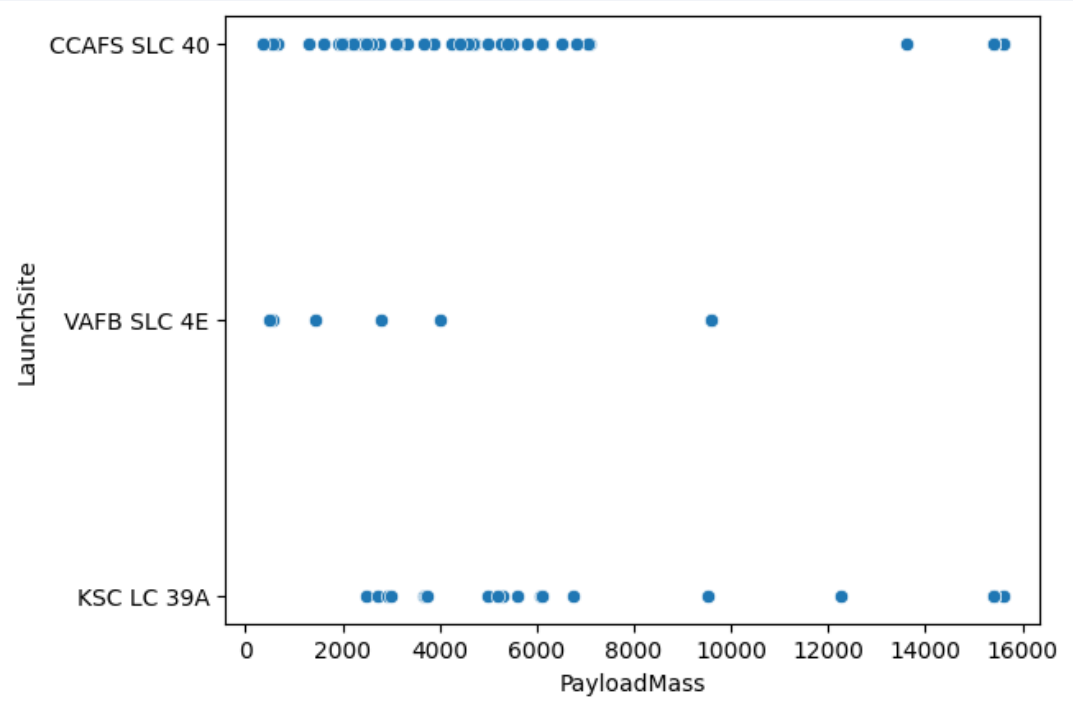

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



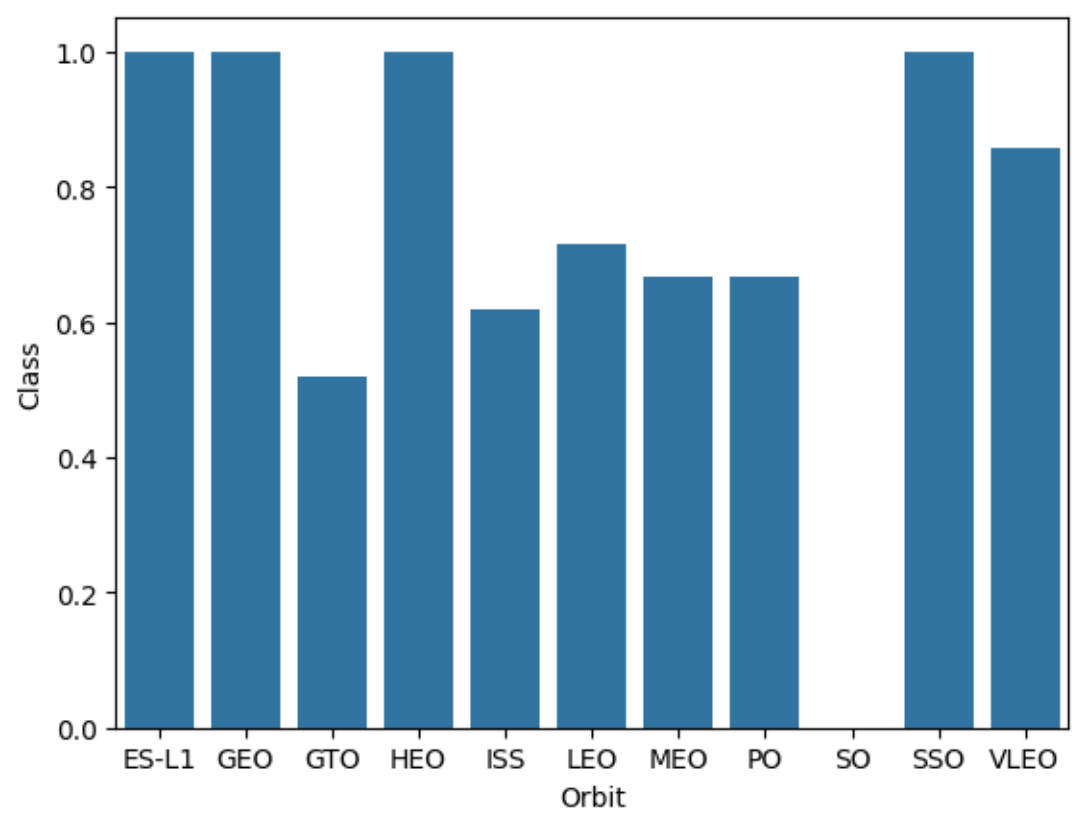
- Most of the lunch were held on first lunch site “CCAFS SLC 40”
- Second lunch site “VAFB SLC 4E” only made occasionally through out flight number
- Third lunch site “ KSC LC 39A” were started to lumch from flight number starting from around 25.

Payload vs. Launch Site



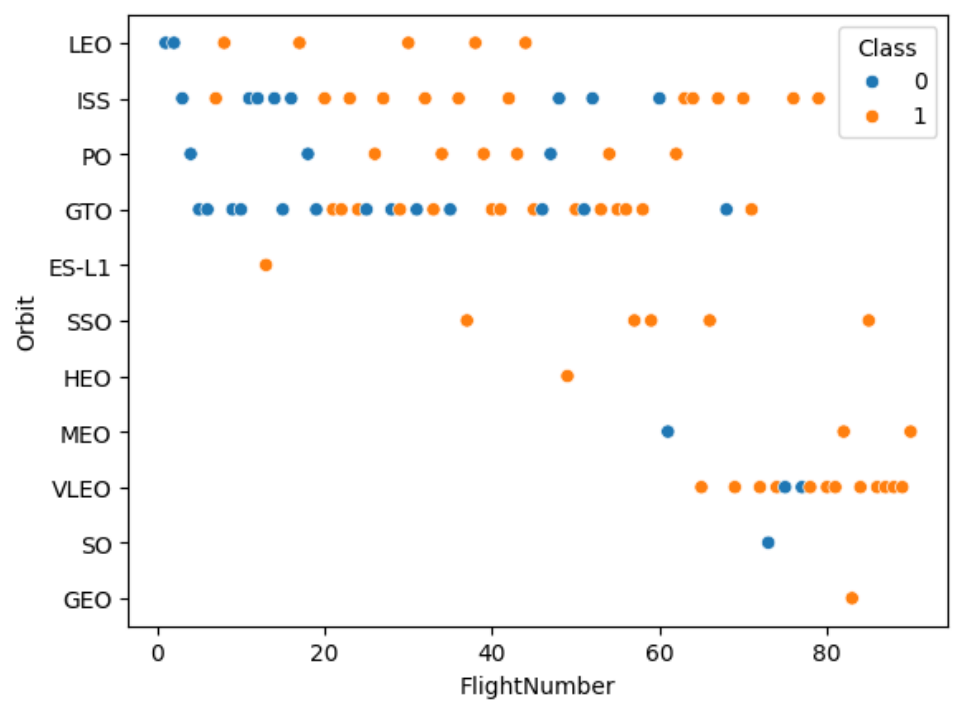
- From the first lunch site “CCAFS SLC 40” most of the payload mass was below 8000kg while only 3 payload was out of bound.
- Second lunch site “ VAFB SLC 4E” were tend to have distributed payload from 0 to 10000kg.

Success Rate vs. Orbit Type



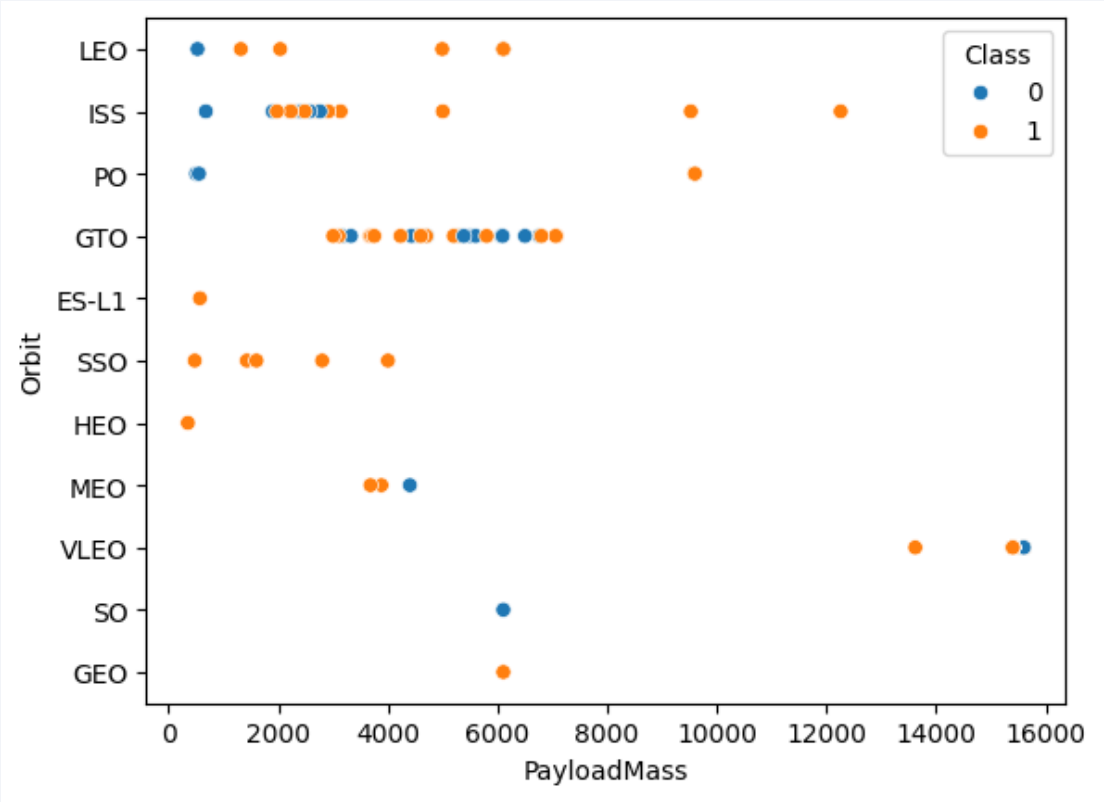
- The success rate of orbit ES-L1, GEO, HEO, SSO was 100% for all the lunches.
- Orbit GTO, ISS, LEO, MEO, PO success rates varied from 50% to 70$ which is pretty low compares to the orbits above which scores 100% success rate

Flight Number vs. Orbit Type



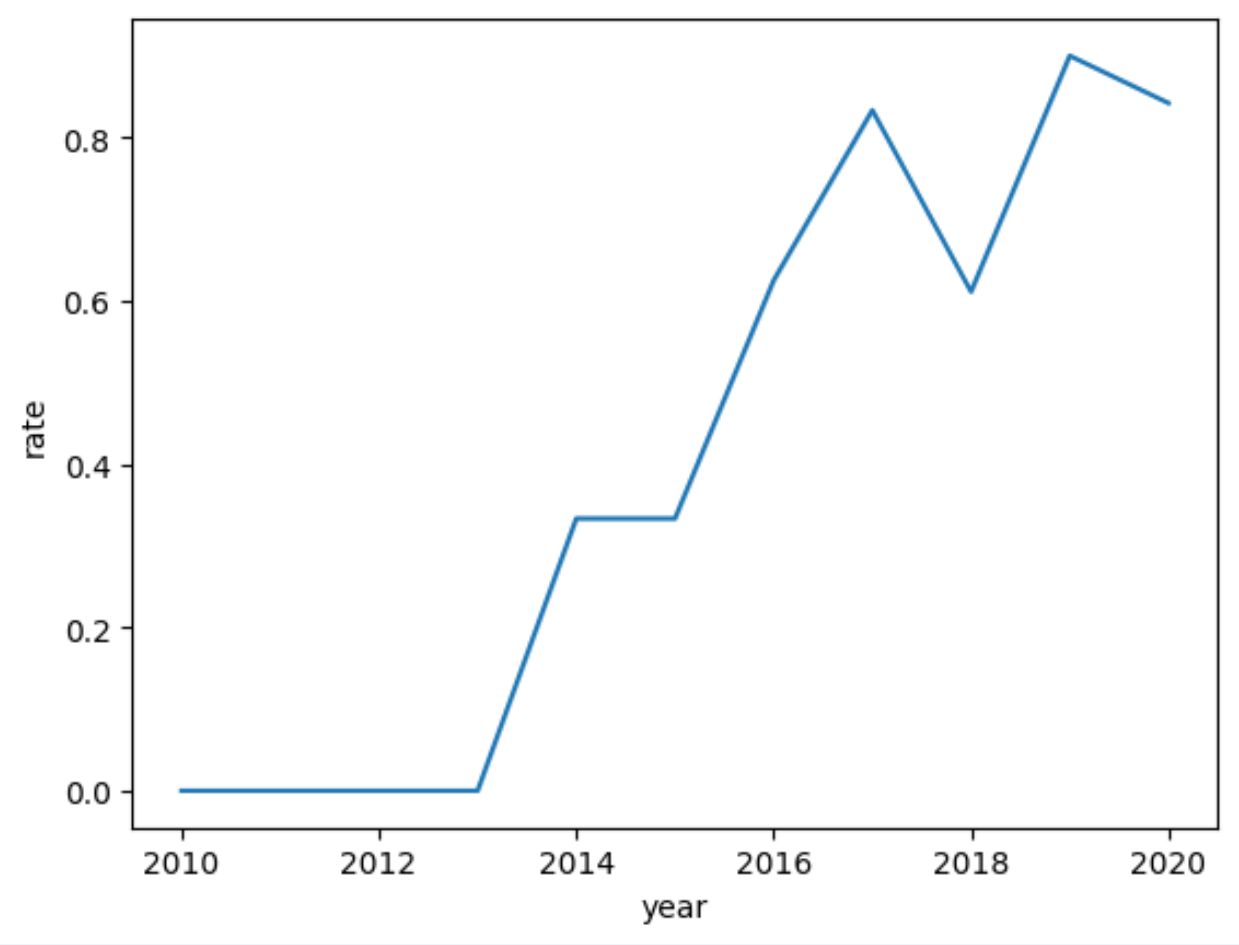
- When the flight number increases, the success rate increases as well.
- For example, from flight number 60 till the end, only around 5 failures were found.
- While flight numbers from 0-20, there are more failures than successes.

Payload vs. Orbit Type



- From the plot above orbit vs success rate, GTO was doing worst and it seems reasonable that GTO has to highest density of the payload among the other orbit
- The failures seem to vary all over the payload mass which might suggest that payload mass and success might not be correlated

Launch Success Yearly Trend



- The line plot clearly shows that the success rate increased by the year which was 0% from 2010 to 2013.
- The success rate increases sharply from 2013 to 2017 reaching 80% from 0%.
- From 2017 to 2020, the success rate varies from 60% to 90% which is a clear increase from 2010 to 2017.

All Launch Site Names

```
%sql select distinct Launch_Site from SPACEXTABLE
```

```
* sqlite:///my_data1.db
one.
```

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

- All lunch site is “CCAFS LC -40”, “VAFB SLC-4E”, ”KSC LC-39A”, “CCAFS SLC-40”
- Used ”distinct” query to extract the unique value from the columns “Launch_Site” from dataset SPACEXTABLE

Launch Site Names Begin with 'CCA'

File display

```
%sql select * from SPACEXTABLE where Launch_Site like 'CCA%' limit 5
```

```
* sqlite:///my_data1.db
one.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_O
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (pa
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 (pa
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No

- Used like “CCA%” that starts with CCA and limits 5 to present only the first 5 results.

Total Payload Mass

```
%sql select sum(PAYLOAD_MASS__KG_) as total_mass from SPACEXTABLE where Customer like 'NASA%'
```

```
* sqlite:///my_data1.db
one.
```

total_mass
99980

- Used sum() query to sum all the elements in the column “PAYLOAD_MASS” and used ”as” query to name the new column as ”total_mass”
- Used ”NASA%” to fine the result start with NASA

Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) as average_mass from SPACEXTABLE where Booster_Version like 'F9%'
```

```
 * sqlite:///my_data1.db
Done.
```

average_mass
6138.287128712871

- Calculated the average payload mass carried by booster version F9 v1.1
- Used avg() query to average the values from the column “PAYLOAD_MASS__KG_) and assigned the new name as “average_mass” by using the” as” query.
- Also used the ”like” query with “F9%” so that capture the column name “Booster_Version” start with F9.

First Successful Ground Landing Date

```
%sql select min(Date) from SPACEXTABLE where Landing_Outcome like 'Success'
```

```
 * sqlite:///my_data1.db
Done.
```

min(Date)
2018-07-22

- Found the dates of the first successful landing outcome on ground pad
- Used the “min()” query to min value of the “Date” column which would be considered as the first date where the “Landing_Outcome” is like ”Success”

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select distinct(Booster_Version) from SPACEXTABLE where Landing_Outcome
is 'Success (drone ship)' and (PAYLOAD_MASS__KG_ >4000 and PAYLOAD_MASS__KG_ <6000)
```

Python

```
 * sqlite:///my_data1.db
Done.
```

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

- Listed the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- Used after “where”, added ”is” to indicate the columns and specify the range needed, which is 4000 to 6000 using and.
- *(The screenshot is different because the query was too long to capture from GitHub so I have to capture it from visual studio)*

Total Number of Successful and Failure Mission Outcomes

```
%sql select count(*) as count from SPACEXTABLE group by Mission_Outcome is 'Success'
```

```
* sqlite:///my_data1.db
one.
```

count
3
98

- Calculated the total number of successful and failure mission outcomes
- Used “count()” function using “*” to count every rows from SPACEXTABLE and group them by “Mission_Outcome” columns where it is success

Boosters Carried Maximum Payload

```
%sql select Booster_Version From SPACEXTABLE where max(PAYLOAD_MASS__KG_)
```

```
* sqlite:///my_data1.db
sqlite3.OperationalError) misuse of aggregate function max()
SQL: select Booster_Version From SPACEXTABLE where max(PAYLOAD_MASS__KG_)]
Background on this error at: http://sqlalche.me/e/e3q8)
```

```
%sql select * from (select max(PAYLOAD_MASS__KG_) from SPACEXTABLE)
```

```
* sqlite:///my_data1.db
one.
```

max(PAYLOAD_MASS__KG_)
15600

- Listed the names of the booster which have carried the maximum payload mass
- First, extracted the “Booster_Version” with the maximum “PAYLOAD_MASS_KG_” and present the element

2015 Launch Records

```
%sql select substr(Date,6,2) as month_names, Landing_Outcome as failure_Landing_Outcome, Booster_Version, Launch_Site, Date
from SPACEXTABLE where Date like '2015%' and Landing_Outcome like 'Failure%'
```

Python

```
 * sqlite:///my_data1.db
Done.
```

month_names	failure_Landing_Outcome	Booster_Version	Launch_Site	Date
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40	2015-01-10
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40	2015-04-14

- Listed the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- First, extract the Year from “Date” as ”month_names”
- Extract the ”Landing_Outcome” as ”failure_Landing_Outcome”, and more columns from SPACEXTABLE, used “like” function to extract 2015 and failure from “Date” and “Landing_Outcome”

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

```
%sql select Landing_Outcome, count(*) as count from SPACEXTABLE where (Date > '2010-06-04' and Date < '2017-03-20')
group by Landing_Outcome order by count desc
```

✓ 0.0s Python

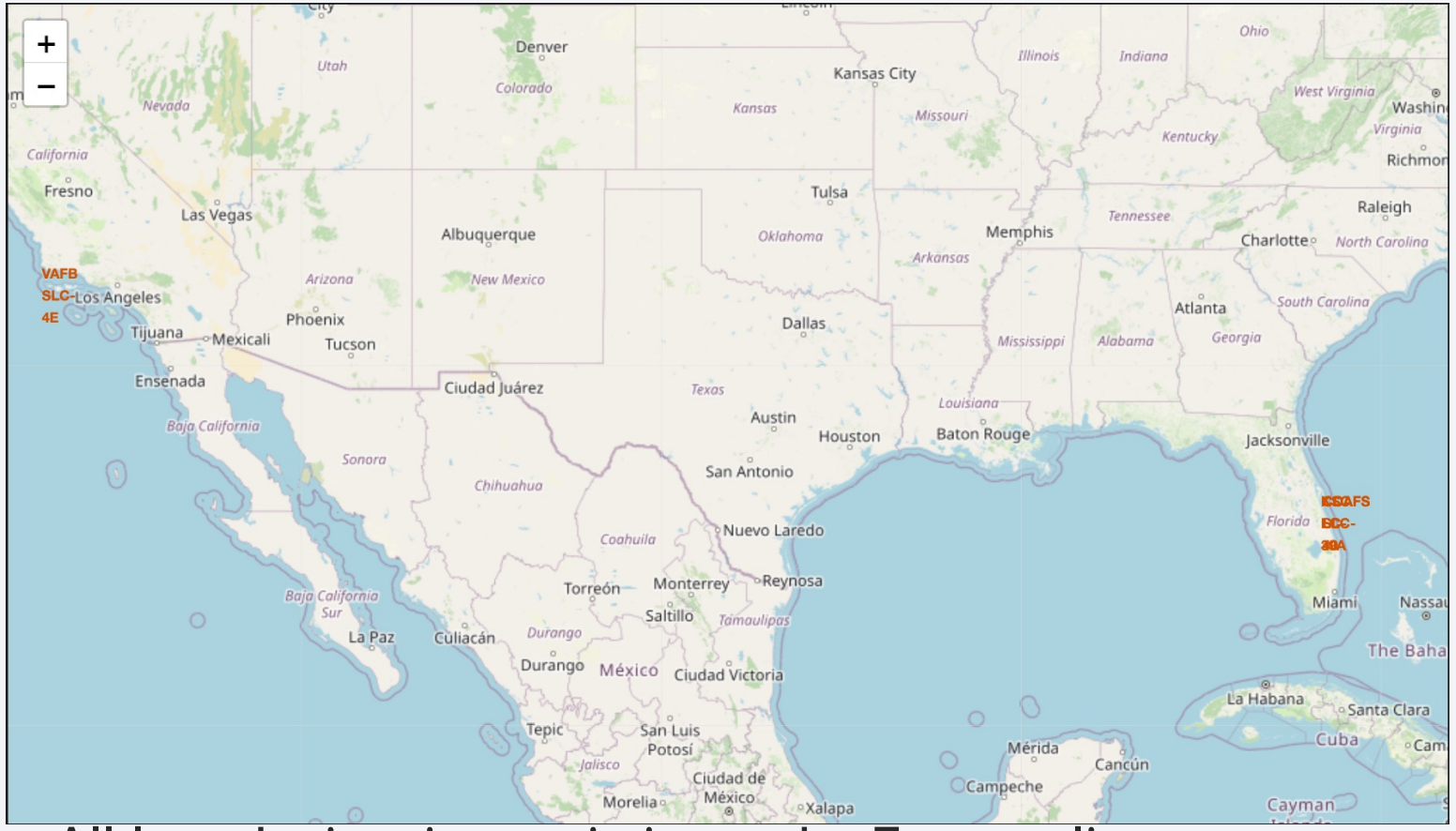
```
 * sqlite:///my_data1.db
Done.
```

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

- Ranked 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
- Used the ”group by” to extract distinct ”Landing_Outcome” and used “order by” function to make them descending order


Section 3
Launch Sites
Proximities Analysis

<All launch sites location markers on global map>



- All launch sites in proximity to the Equator line
- All launch sites in very close proximity to the coast

<Color-labeled launch outcomes>

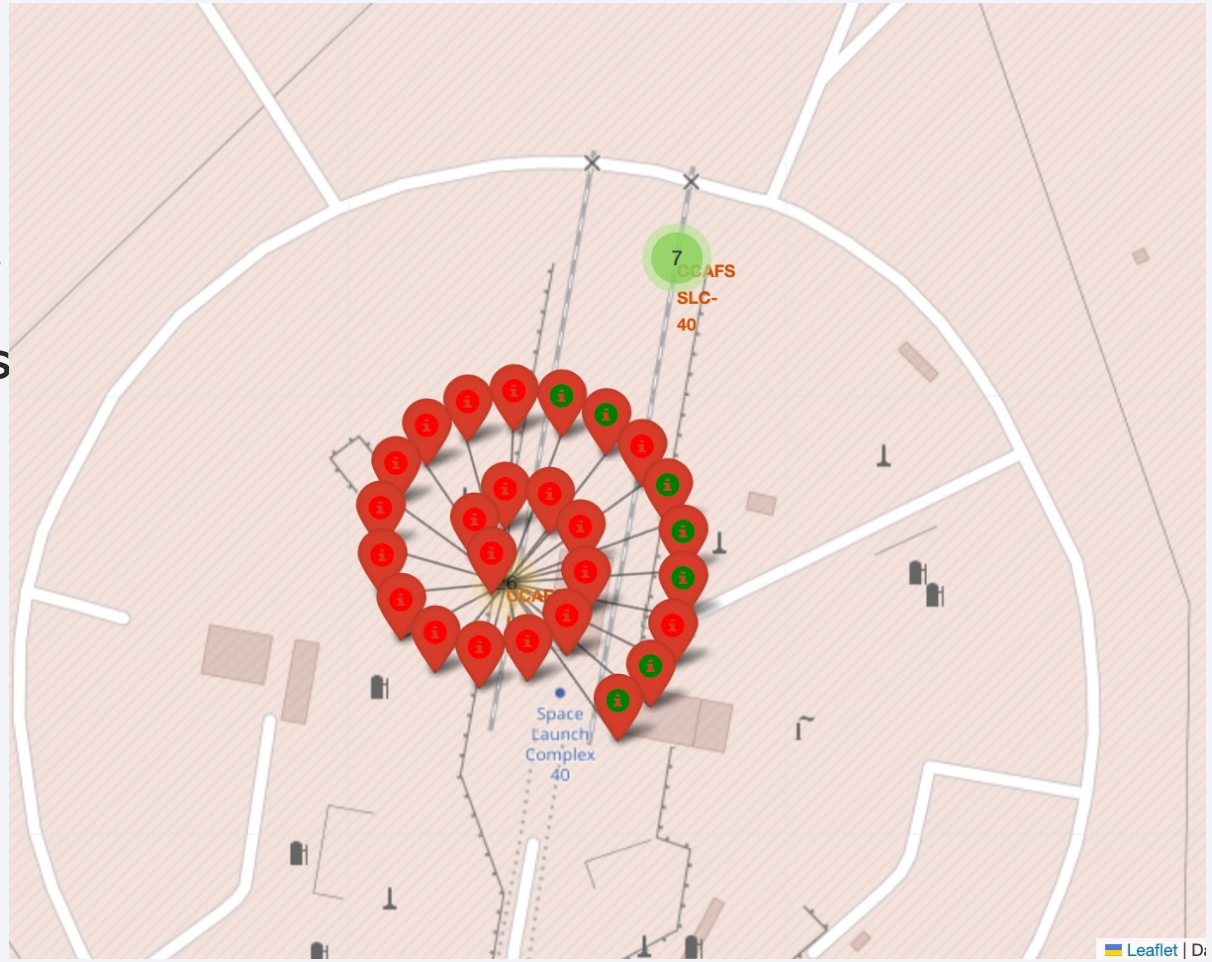
- The color labeled green a red

  it is easier to identify the success

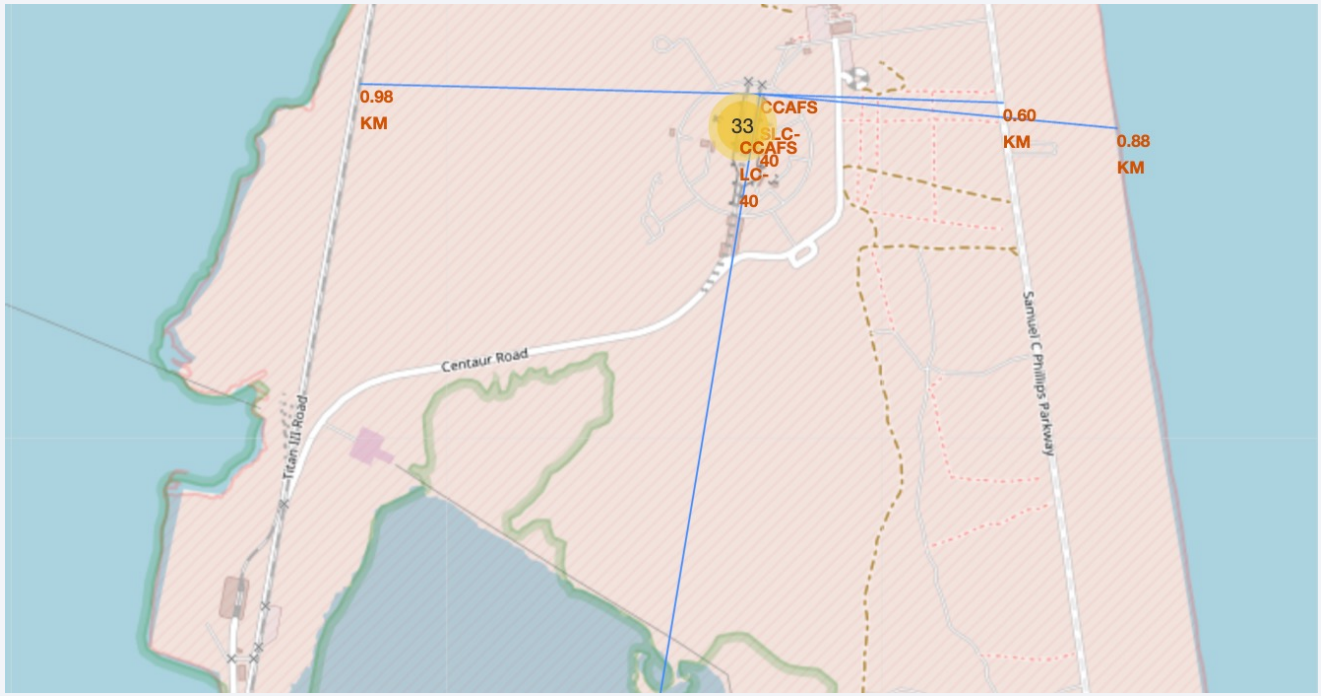
  and failure from each launch sites

- Also, the number of launches

  for each launch sites



<Distance calculated to railway, highway, coastline>



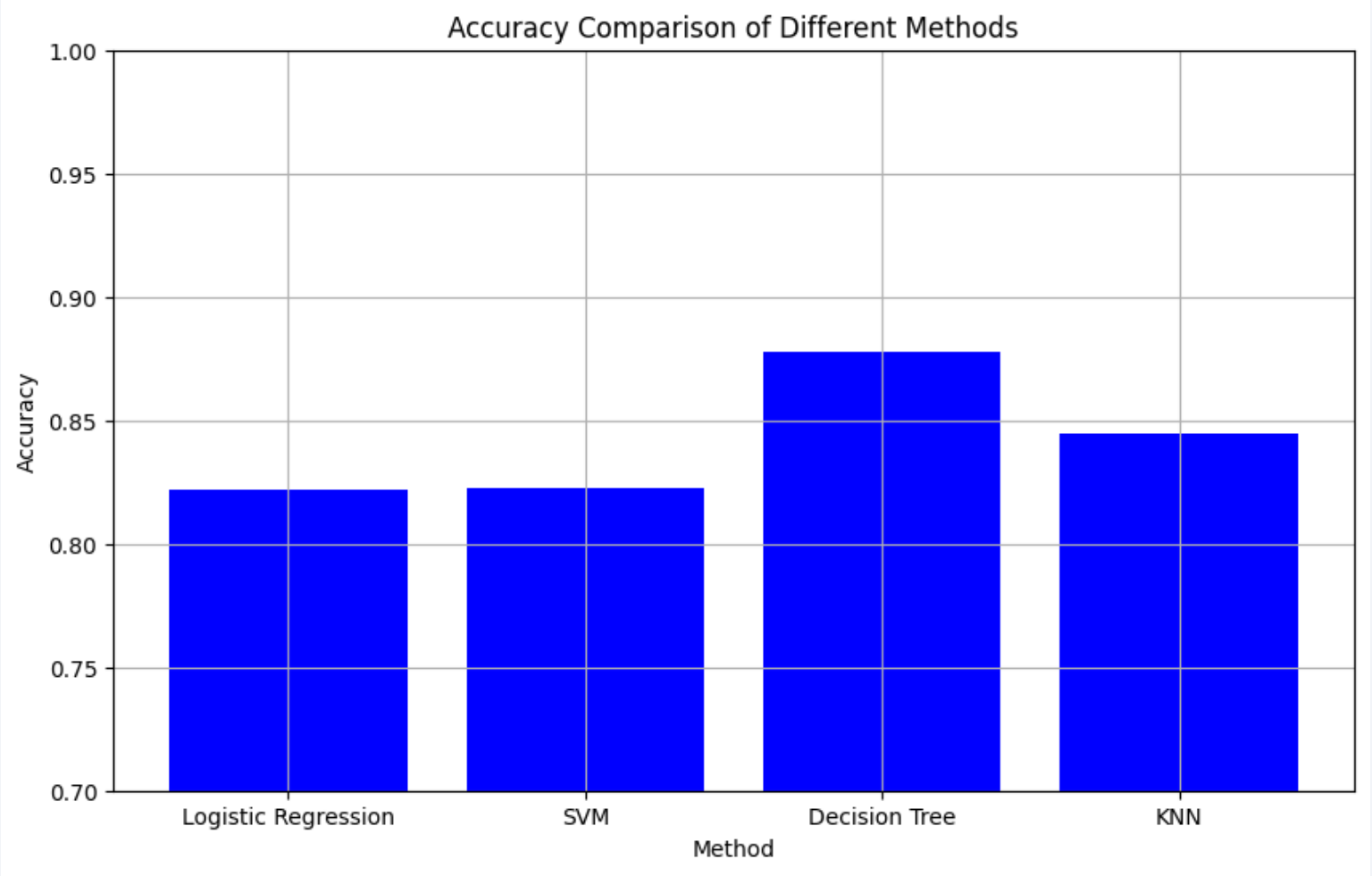
- The launch site is very close to railway, highway, and coastline
- Also, the line toward the south side is toward the closest city. It makes sure that the launch site keeps certain distance between the city

Section 5

Predictive Analysis (Classification)

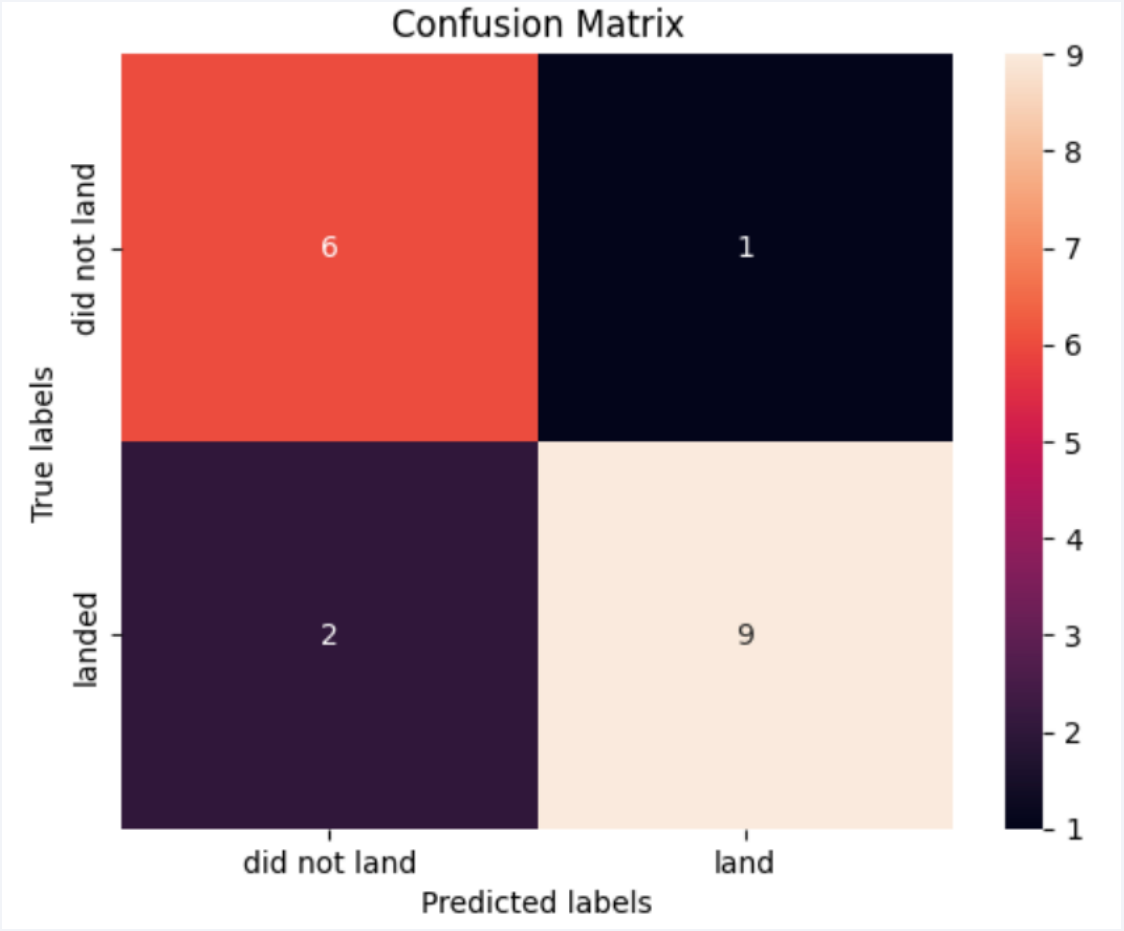
Classification Accuracy

- The best model among logistic regression, SVM, Decision Tree, KNN is the Decision Tree method.
- The decision tree method scores about 0.88



Confusion Matrix – Decision Tree

- Confusion matrix of the decision tree model
- This confusion matrix shows how accurate the true value is.
- True labels and Predicted labels with bot did not land and land has much higher count than mismatch



Conclusions

- From dataset related to spaceX, shows that the success rate of the payload is related to flight number which can be considered as a year or time.
- Other than flight number, orbits, launch sites or payload mass do not seem to correlate to the success of the launches
- Among logistic regression, SVM, decision tree, and KNN, by comparing the best accuracy for predicitoin, the decision tree shows the best score.
- The above claim is supported by the confusion matrix of the decision tree.

Thank you!