IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Naeem Ghahramanpour
2/23/2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive analysis
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

Introduction

Nowadays, companies make space travels affordable for everyone, such as Virgin Galactic, Blue Origin, Rocket Lab. Among these companies, perhaps the most successful one is SpaceX. One reason for SpaceX's success is that SpaceX can reuse the first stage. Therefore, this company can launch rockets at a lower cost than other companies. So if we can determine if the first stage will land, we can determine the launch cost.

In this project, we will predict if Falcon9, one of SpaceX rockets which it advertises on its website, the first stage will land successfully based on the information gathered from SpaceX launches data. Also, we will evaluate if SpaceX will reuse the first stage. For this project, we will train machine learning to predict if SpaceX will reuse the first stage or not based on public information instead of rocket science.

So hope you enjoy reading this report till the end.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Using SpaceX API
  - Scrapping SpaceX wikipage.
- Perform data wrangling
  - Encoding categorical data with one hot encoding for machine learning and also choose the most relevant and correlated columns with success rates.
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Using scatter plot, Bar charts, line plot to show relationships between variables and target data. Also using queries to get some important information about the SpaceX launches.
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

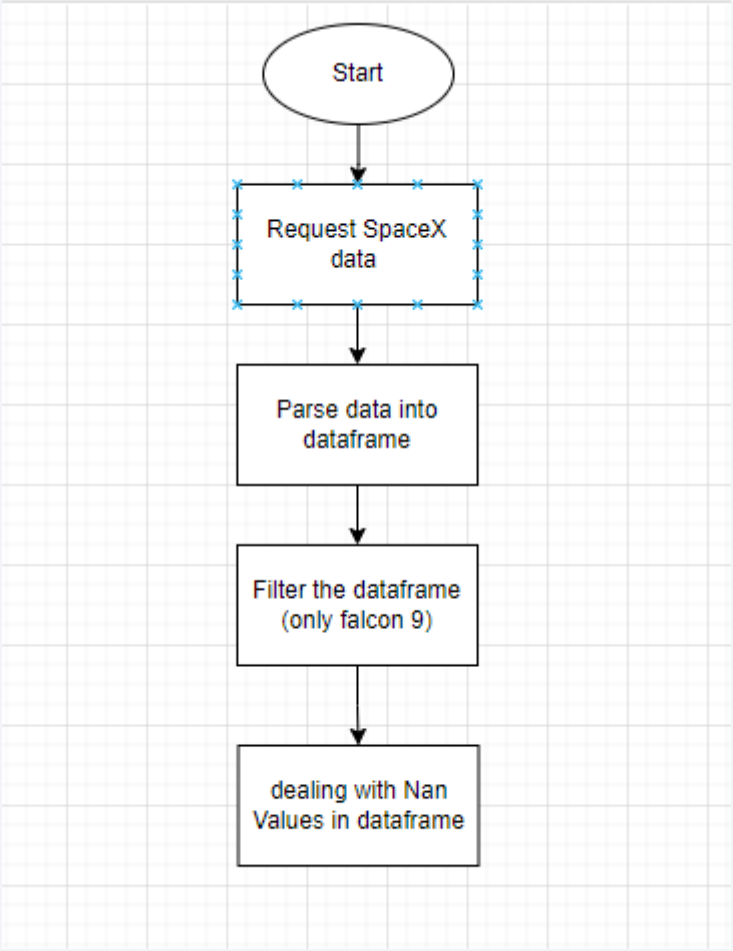
Data Collection

For collecting data, we use two methods:

- First by using SpaceX API. We gathered data by sending a request to SpaceX API and used columns relative to the project such as “FlightNumber”, ”Date”, “BoosterVersion”, “PayloadMass”, and for our project we need only data relative to Falcon9 launches, so we selected only rows with “BoosterVersion” Falcon 9
- Our second method was web scraping to collect Falcon 9 historical launch records from a Wikipedia page (titled as “List of Falcon 9 and Falcon Heavy launches”).

Data Collection – SpaceX API

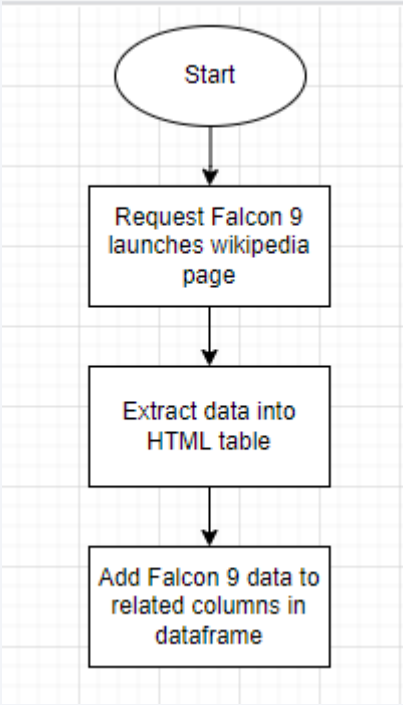
- In this method, after requesting data from SpaceX API and turning it into a pandas data frame, we select only data relative to falcon nine launches with specific columns that will help us in the project. In this step, we only handle Nan values in number-type columns.



- Github URL - notebook name: data_collector.ipynb

Data Collection - Scraping

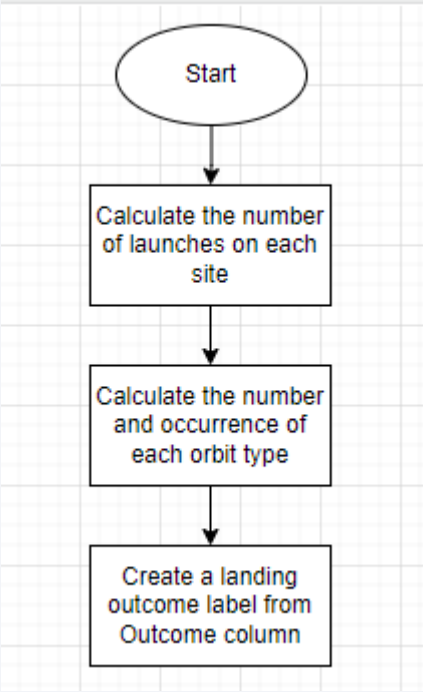
- In web scraping, we scraped the data from a snapshot of the “List of Falcon 9, and Falcon Heavy launches” wiki page. After getting data, we extracted variables related to columns we needed and parsed the HTML table to create a data frame.



- Github URL - notebook name: data_collection_web_scrapping.ipynb

Data Wrangling

- In this part of the project, we analyzed data gathered from previous methods and we calculated the number of launches on each site, the number and occurrence of each orbit, the number occurrence of missions outcome per orbit type. We also create a landing outcome label from the Outcome column, identifying successful and unsuccessful landings.



- Github URL - notebook name: data_exploratory_analysis.ipynb

EDA with Data Visualization

- The charts that we used in EDA was:
  - Scatter plot: We used this plot to show the relationship between different columns and the target column (launch outcome, which we named Class column) and how these columns affect the target column.
  - Bar chart: To check relationship between the orbit type and the success rate.
  - Line plot: To get the average launch success trend (get the average launch success for each year).
- In this part, after finding the most important variables, we create dummy variables from categorical columns to use them in next steps.

- Github URL - notebook name: EDA_visualization.ipynb

EDA with SQL

- SQL queries used in project:
  - We displayed the names of the unique launch sites in the space missions.
  - We displayed five records where launch sites begin with the string ‘CCA’
  - Displayed the total payload mass carried by booster launcher by NASA (CRS)
  - Displayed average payload mass carried by booster version F9 v1.1
  - Displayed list of date when the first successful landing outcome in ground pad was achieved
  - Listed the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - Listed the total number of successful and failure mission outcomes.
  - Listed the names of the booster versions which have carried the maximum payload mass.
  - Listed the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015
  - Ranked the count of landing outcomes or Success between the date 2010-06-04, 2017-03-20 in descending order
- Github URL - notebook name: EDA_with_sql.ipynb

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
- All objects that used in map are:
  - We marked all launch sites on the map with folium circles and used markers to write their names.
  - We used marker cluster object to mark the success/failed launches for each site on the map.
  - We also added a mouse position object on the map to get coordinate for a mouse over a point on the map.
  - Finally, with polyline, we show the distances between a launch site to its proximities.
- The reason behind showing these objects in an interactive map was to analyze each launch site position and its proximities. With this information, we figured out that it is vital that each launch site position must be away from cities and be close to coastlines, etc.

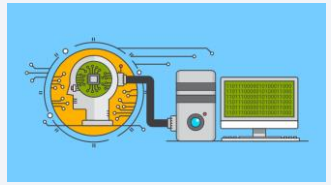
- Github URL - notebook name: Interactive Visual analytic.ipynb

Build a Dashboard with Plotly Dash

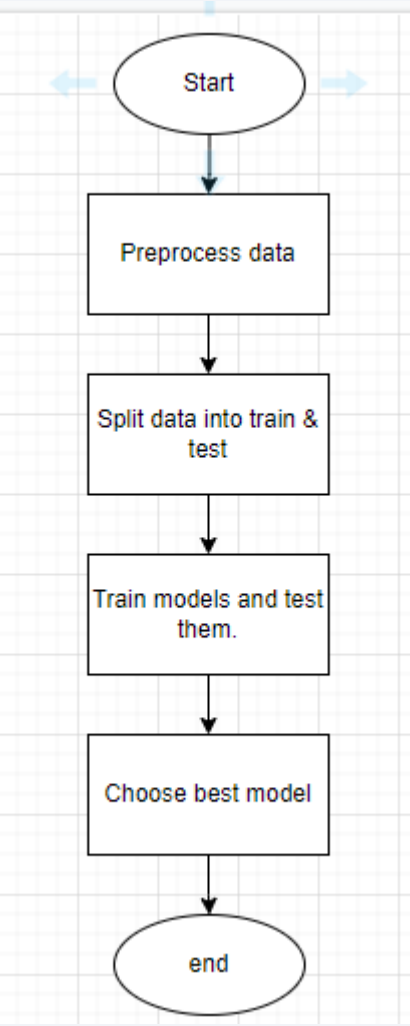
- In project's dashboard we used input components such as a dropdown and a range slider to interact with a pie chart and a scatter point chart. The dropdown in dashboard choose Launch site and with help of the range slider, we select payload amount.
- This dashboard helps us to perform interactive visual analytics on SpaceX launch data in real-time.

- Github URL - notebook name: spacex_dash_app.py

Predictive Analysis (Classification)



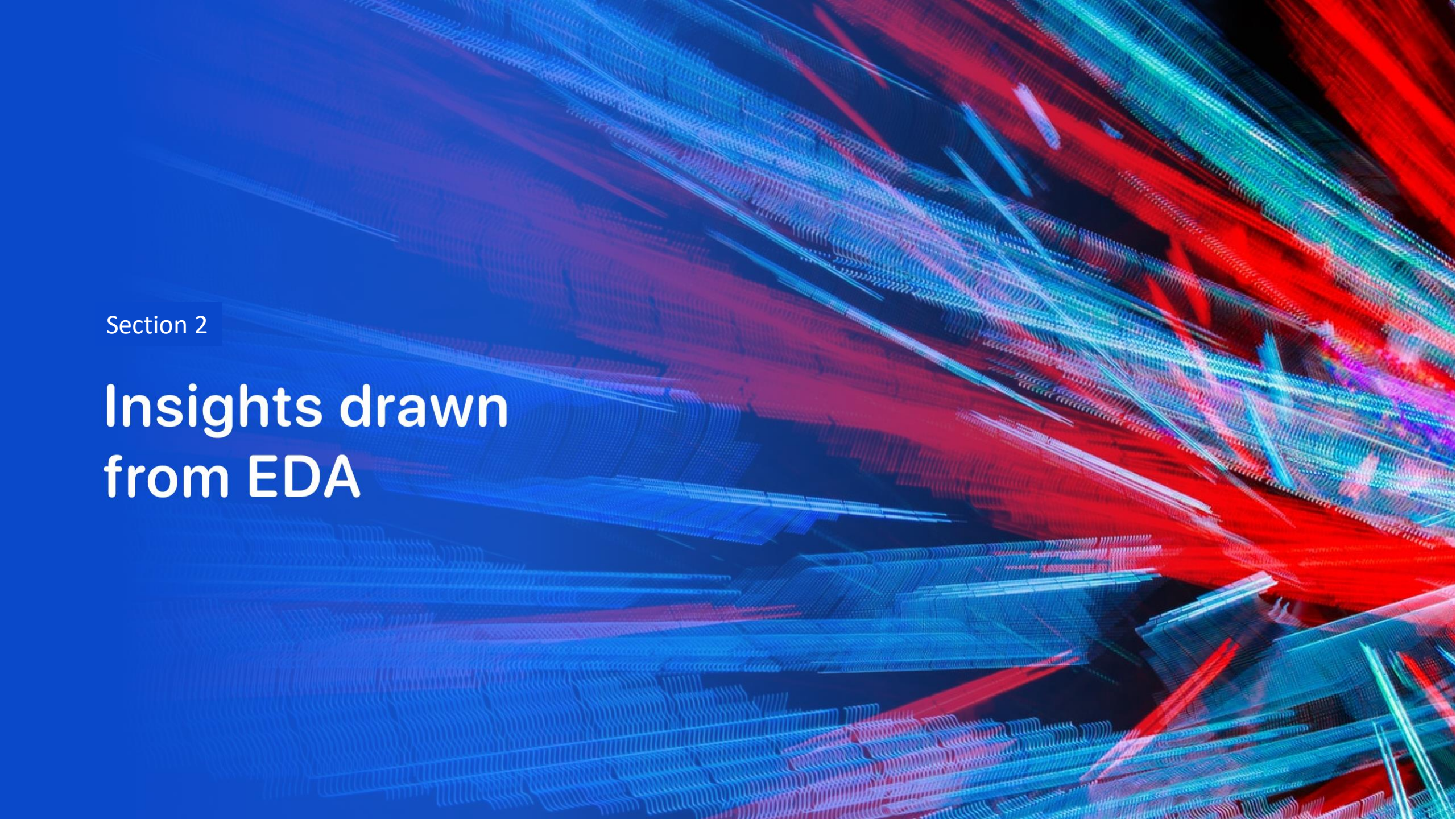
- For building different models, we first separate the Class column from the data frame and preprocess and normalize the other data, then split our data to train and test for evaluating the models' performance. For each model, we used the GridSearchCV to find the best hyperparameter for it. The models we examined in this project were Linear Regression, SVM, Decision tree, KNN.



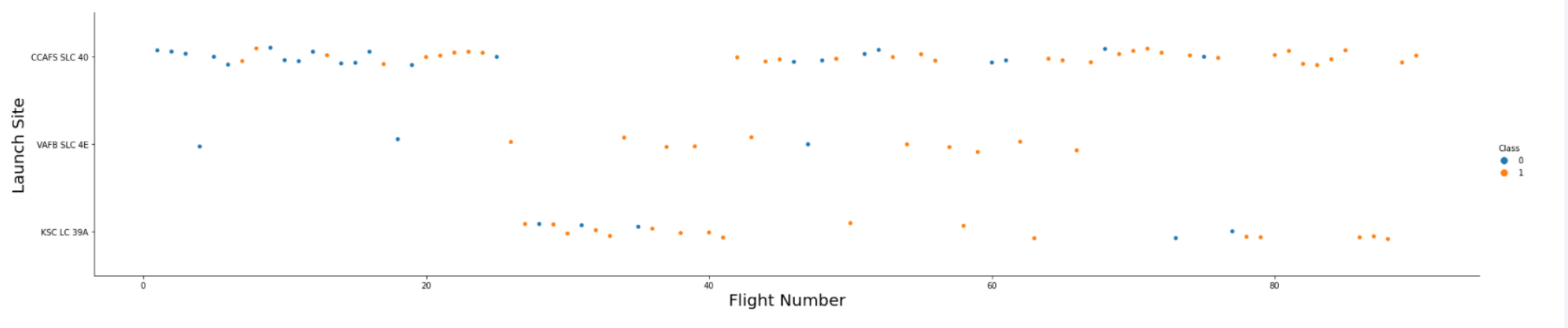
- Github URL - notebook name: IBM_MachineLearning_prediction.ipynb

Results

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

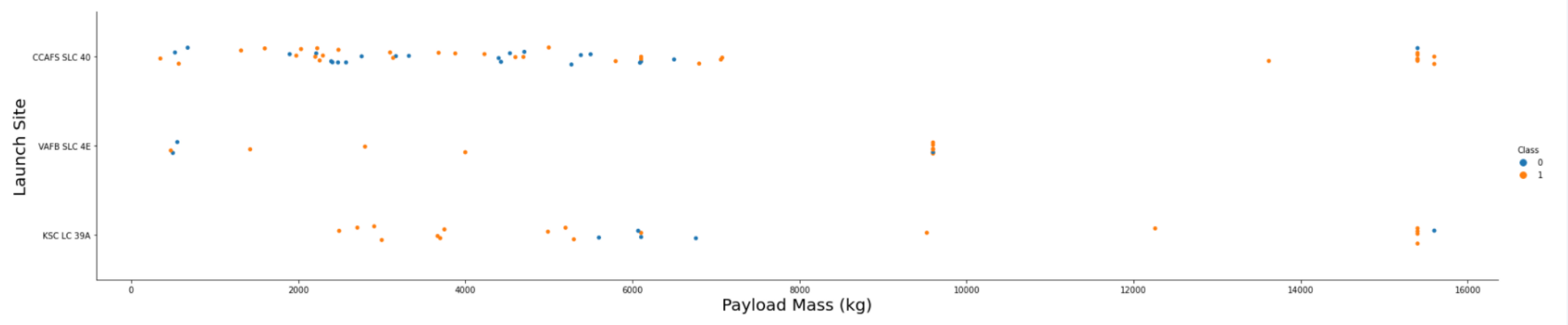

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



- As you can see, different launch sites have a different success rates. LC-39A has a success rate of 77.27 %, while SLC-4E has 76.92 %, and SLC-40 has a 60 % success rate. We see that as the Flight Number increases, the success rate for landing for launch sites increases generally.

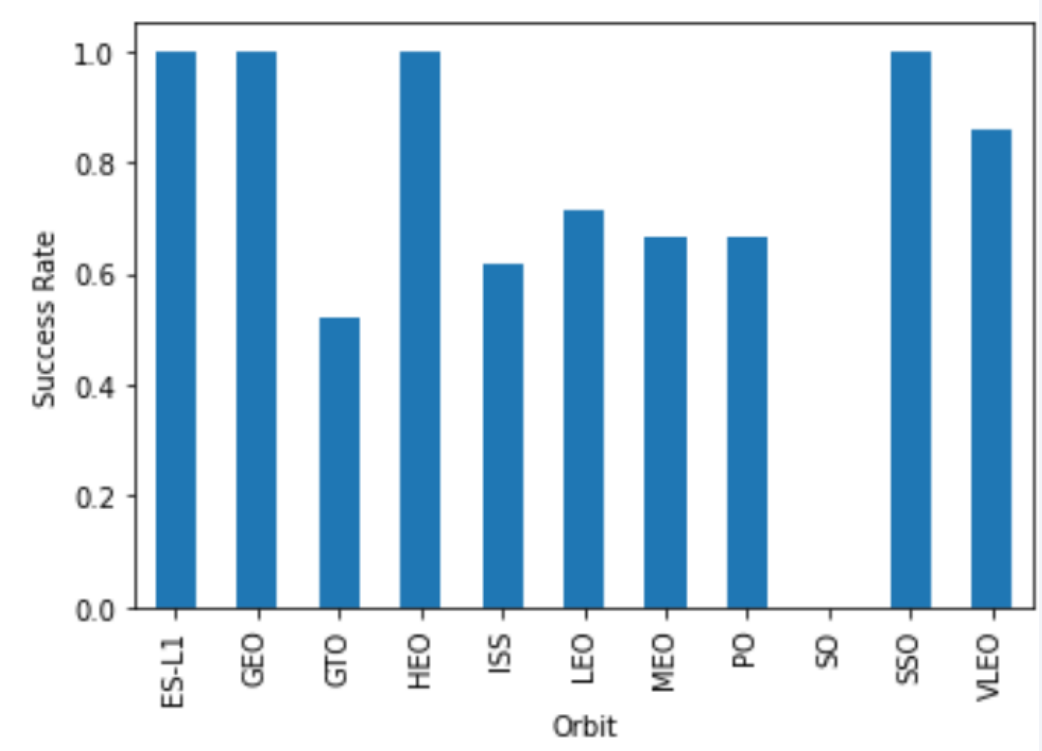
Payload vs. Launch Site



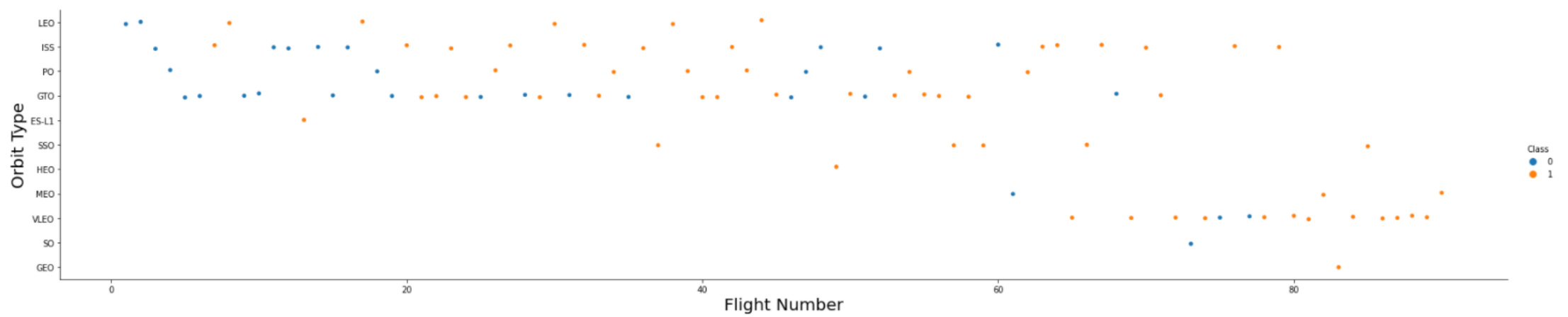
- In this figure, if you observe the chart, you can find that the SLC-4E launch site has no rocket launches with high payload Mass. Also, if you watch the other launch sites' launches, you can figure out that most of the launches are in low payload mass ranges.

Success Rate vs. Orbit Type

- As you can see, the ES-L1, GEO, HEO, SSO has the most success rate average among other orbit types with all launches were successful and with this information, we can understand that there is a relation between orbit type and success rate.
- It is worth mentioning that ES-L1, HEO has low information that we cannot rely only on them and deduce that every launch in these orbits will succeed

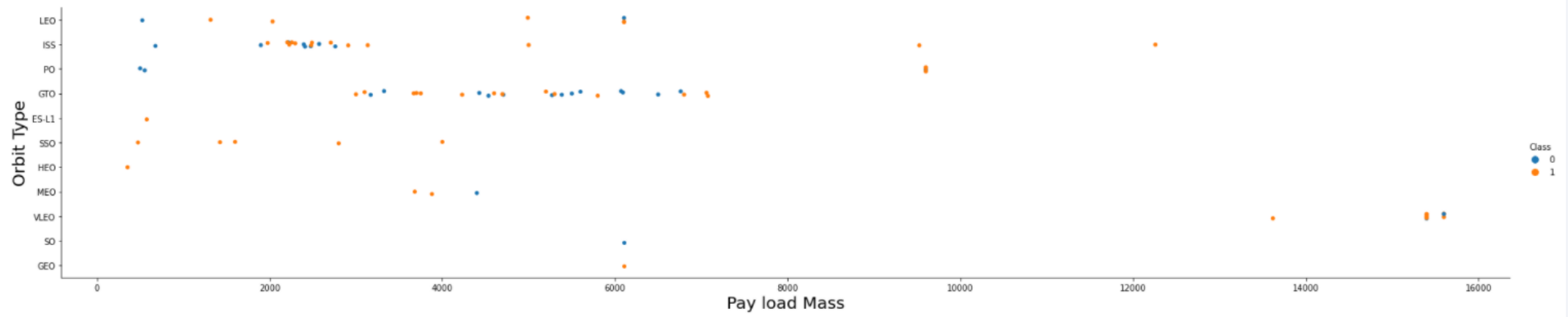


Flight Number vs. Orbit Type



- By looking at the figure, we can see that there is relation between flight number and LEO orbit's success rate. But, for GTO orbit, as you see from the figure, there seems to be no relation with flight number.

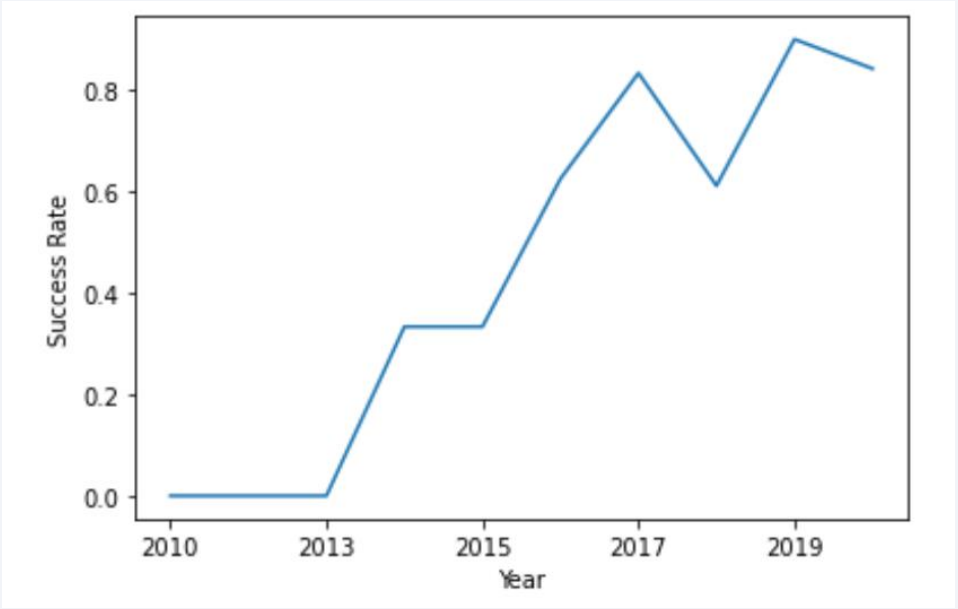
Payload vs. Orbit Type



- By increasing payload mass, we can observe that success rate increases for orbit types like LEO, ISS, PO. However, for GTO, we can’t find any relation between payload mass and orbit type success rate.

Launch Success Yearly Trend

- As you can see, generally the success rate since 2013 started to increase till 2020.



All Launch Site Names

- We've got the unique launch sites in the space mission and as you can see from results, there is four launch site which we will show their coordinates in the front, in launch site proximities section.
- For getting these values, we used unique method in SQL.

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

```
In [5]: %sql select unique(LAUNCH_SITE) from XJQ37119.SPACEXDATASET
```

Launch Site Names Begin with 'CCA'

- As you can see, we're showing five launch sites whose names begin with 'CCA'. For this query, we used LIKE method and LIMIT constraint to find launch sites names start with 'CCA' and limit the number of records to get and show us.

```
In [9]: %sql select * from XJQ37119.SPACEXDATASET where LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	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 this query, we calculate total payload mass carried by boosters launched by NASA (CRS).
- For getting the answer, we used sum method to find total mass and group by method to group records based on customer column and used having to set the condition to find NASA boosters launch.

```
In [15]: %sql select customer, sum(payload_mass__kg_) from XJQ37119.SPACEXDATASET group by customer having customer like 'NASA (CRS)'
 * ibm_db_sa://xjq37119:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

Out[15]:

customer	2
NASA (CRS)	45596

Average Payload Mass by F9 v1.1

- In this query, we found average payload mass carried by booster version F9 v1.1.
- For getting the answer, we used the avg method to find the average of the payload, and as the previous query, we used group-by to group records based on booster_version and used having for setting condition booster_version.

```
In [16]: %sql select booster_version, avg(payload_mass__kg_) from XJQ37119.SPACEXDATASET group by booster_version having booster_version like 'F9 v1.1'
 * ibm_db_sa://xjq37119:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

Out[16]:

booster_version	2
F9 v1.1	2928

First Successful Ground Landing Date

- For finding the first successful ground landing date, we grouped our records based on landing_outcome and got all records with “Success (ground pad)” landing_outcome, and by using the min method on the DATE column, we got the data of the first successful landing outcome.

```
In [18]: %sql select min(DATE) from XJQ37119.SPACEXDATASET group by landing__outcome having landing__outcome like 'Success (ground pad)'
 * ibm_db_sa://xjq37119:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
Out[18]:
1
2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- For getting List of successful drone ship landing with payload between 4000 and 6000, we used two condition on the WHERE that first one choose records with landin__outcome ‘Success (drone ship)’ and second one choose records with payload_mass between 4000 and 6000 kg.
- We link these two condition with AND in sql.

```
In [19]: %sql select booster_version from XJQ37119.SPACEXDATASET where landing__outcome like 'Success (drone ship)' and payload_mass__kg_ > 4000 and payload_mass__kg_ < 6000
 * ibm_db_sa://xjq37119:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

Out[19]:

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

Total Number of Successful and Failure Mission Outcomes

- For calculating the total number of successful and failure mission outcomes, the query we used tried to group records based on their mission_outcome; then, we counted the number of records per group with the COUNT method.

```
In [21]: %sql select mission_outcome, count(mission_outcome) from  XJQ37119.SPACEXDATASET group by mission_outcome
 * ibm_db_sa://xjq37119:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

Out[21]:

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

```
'select booster_version, payload_mass__kg_
from XJQ37119.SPACEXDATASET
where payload_mass__kg_ in (select payload_mass__kg_
                            from XJQ37119.SPACEXDATASET
                            order by payload_mass__kg_ desc limit 1)'
```

Out[25]:

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

- In this query, by using subqueries, we first find the maximum payload carried by boosters with the help of the ORDER BY method and order them in descending order, then select the first record.
- In the main query, we tried to find booster_versions with payload_mass equal to the maximum payload, which we found in a subquery.

2015 Launch Records

‘select landing__outcome, booster_version, launch_site
from XJQ37119.SPACEXDATASET
where extract(year from DATE) = '2015' and
 landing__outcome like 'Failure (drone ship)' ’

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- To acquire the 2015 failed launch records, we first extract the year value from the DATE columns to find records belonging to 2015.
- After finding records belonging to 2015, we filter landing__outcome to find “Failure (drone ship)” values.

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

‘select landing__outcome, count(landing__outcome) as num
from
(select *
from XJQ37119.SPACEXDATASET
where DATE between '2010-06-04' and '2017-03-20') as d
group by d.landing__outcome
order by num desc’

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

- Using subquery to find launches that happened between specified dates.
- Grouping selected records based on landing__outcome and counting each group’s members.
- Finally, ordering found groups based on the number of members that counted in descending order.

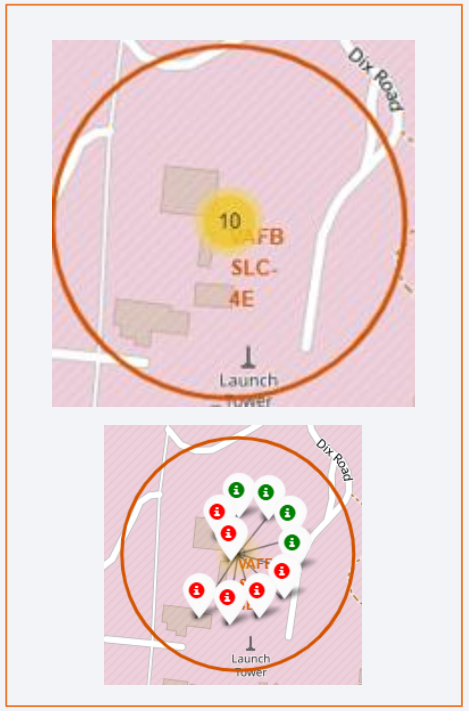
Section 3

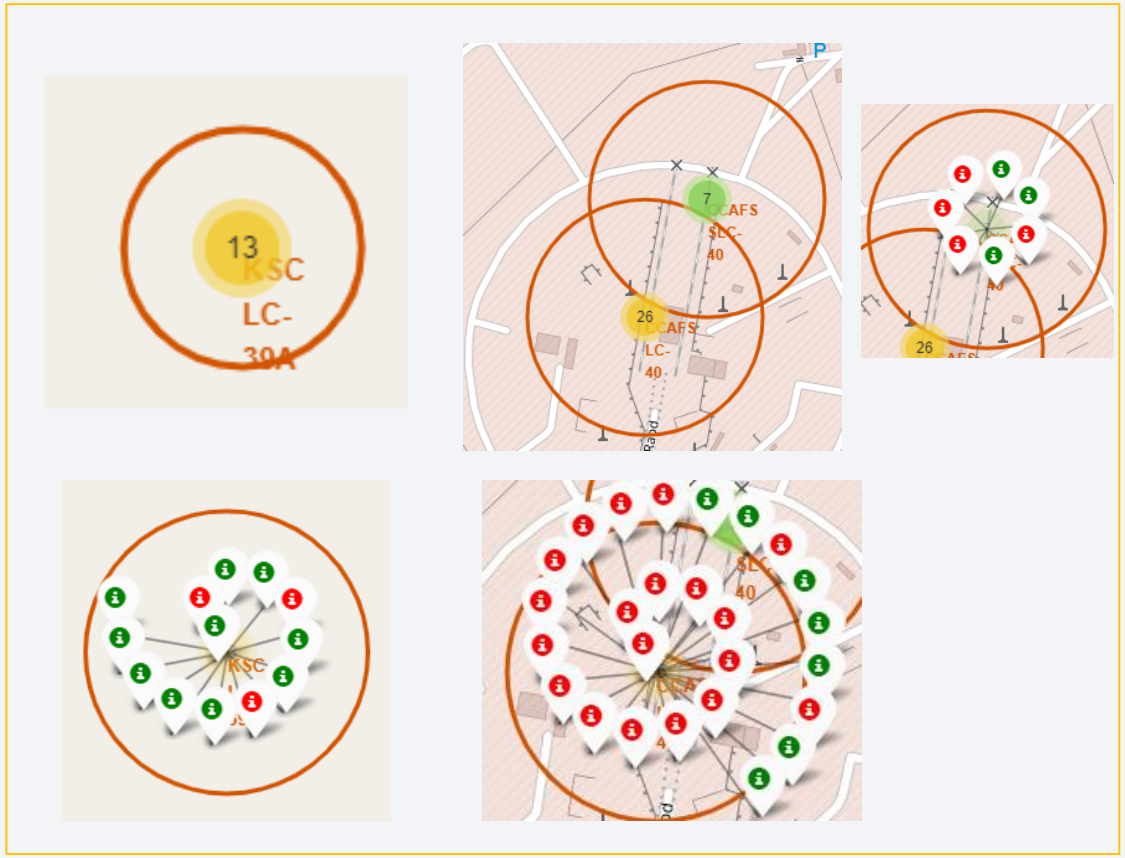
Launch Sites Proximities Analysis

All Launch Sites in the Global Map

- As you can see in the figure, all SpaceX launch sites are in the United States of America.
- There are four launch sites in there that three of them are on the east of the USA and the other one is on the west side.

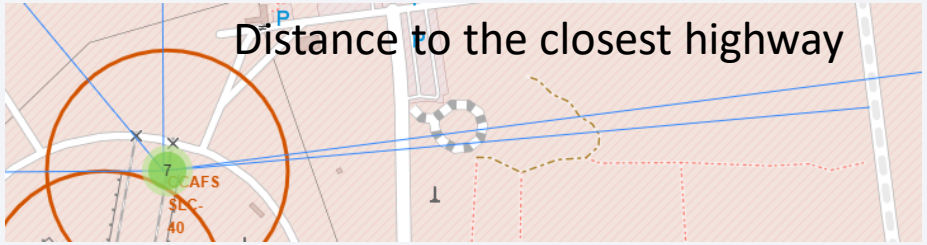
Launch Outcomes Labeled with Colors on the Map

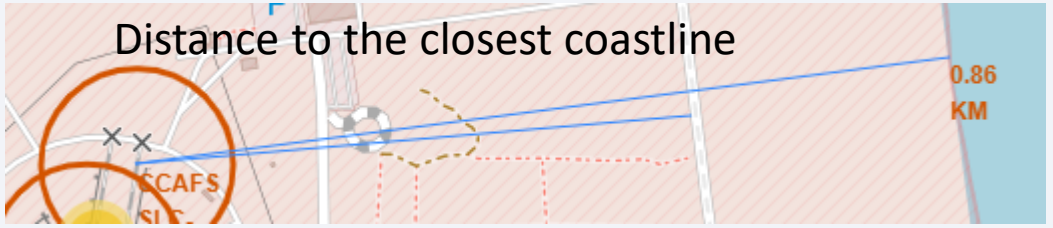




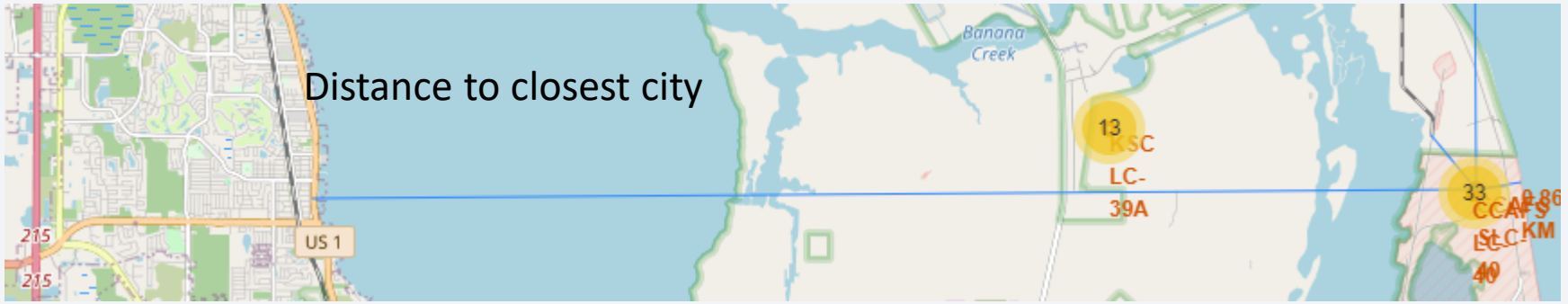
- Left images belong to the west side's launch site
- Right images belong to the east side's launch sites
- Green marker shows successful launches, and red marker shows failed ones.

CCAFS SLC-40 Launch Site Distance with its Proximities



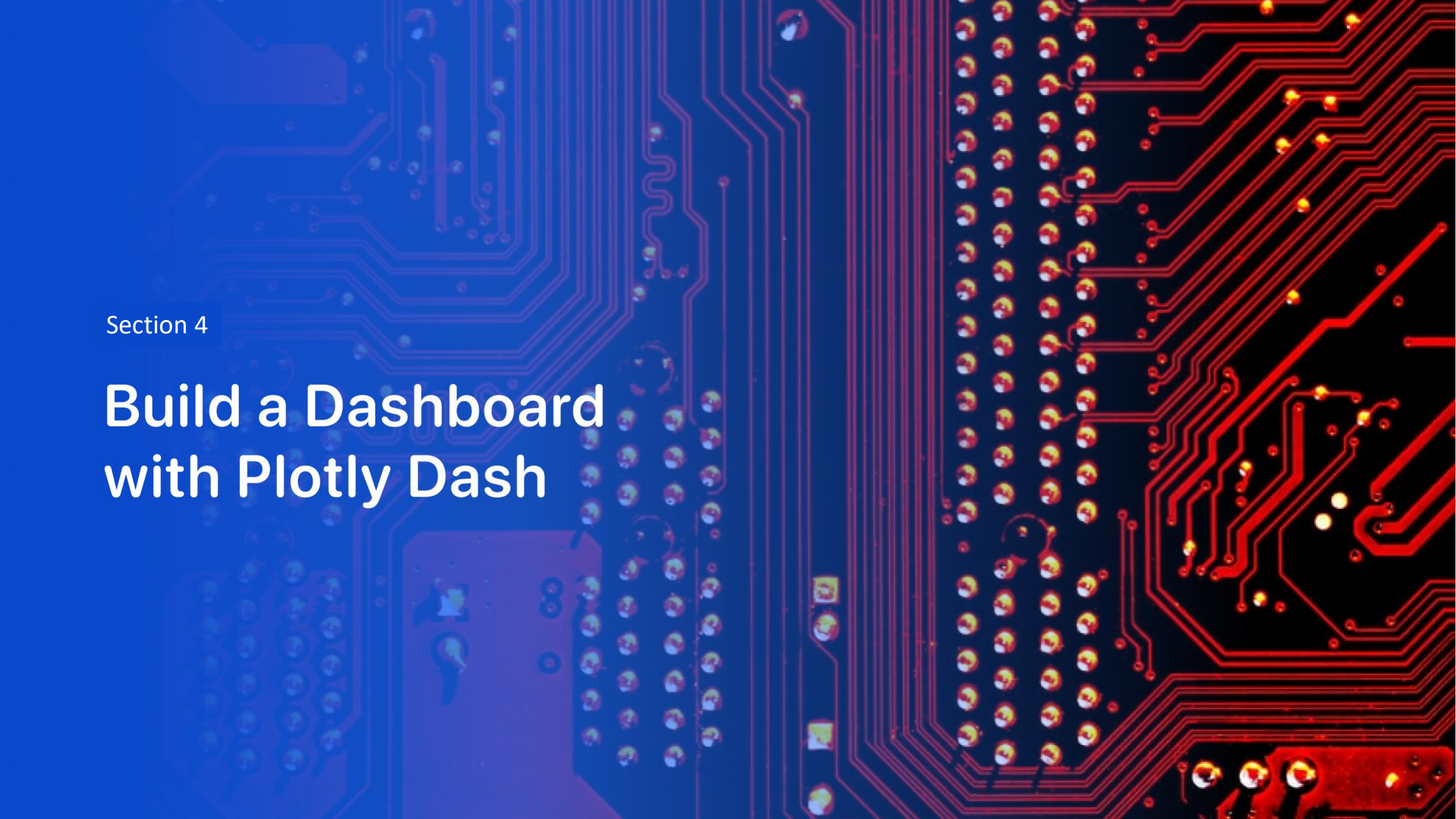


Distance to closest railway

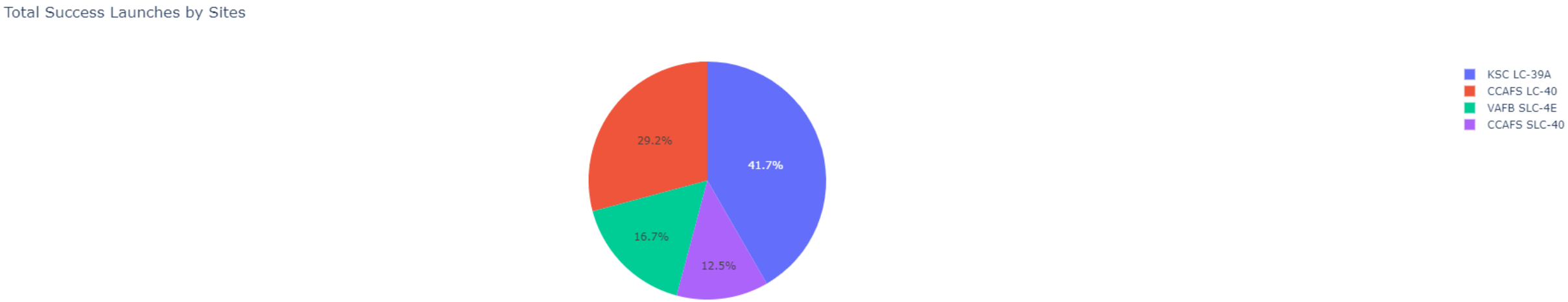


- Are launch sites in close proximity to railways? Yes
- Are launch sites in close proximity to highways? Yes
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Most of these answers to questions are valid for other launch sites too.

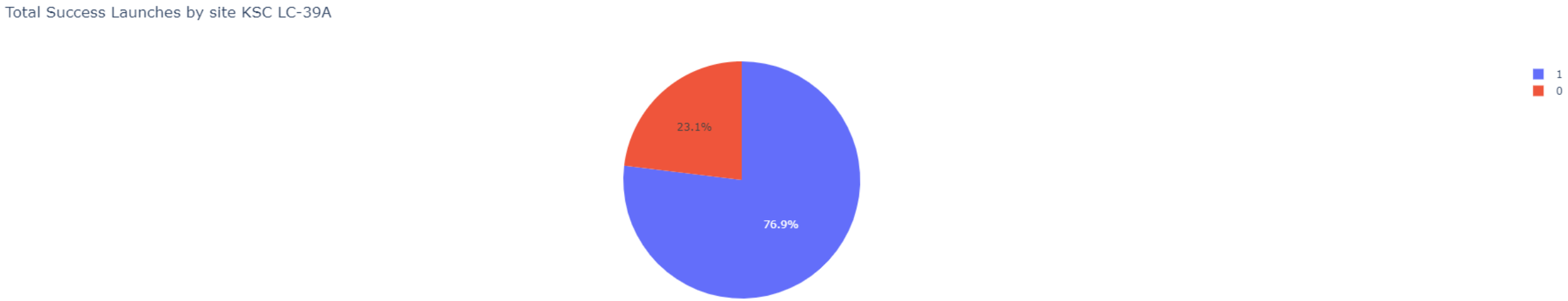

Section 4
Build a Dashboard with Plotly Dash

Launch Sites’ Success Rate



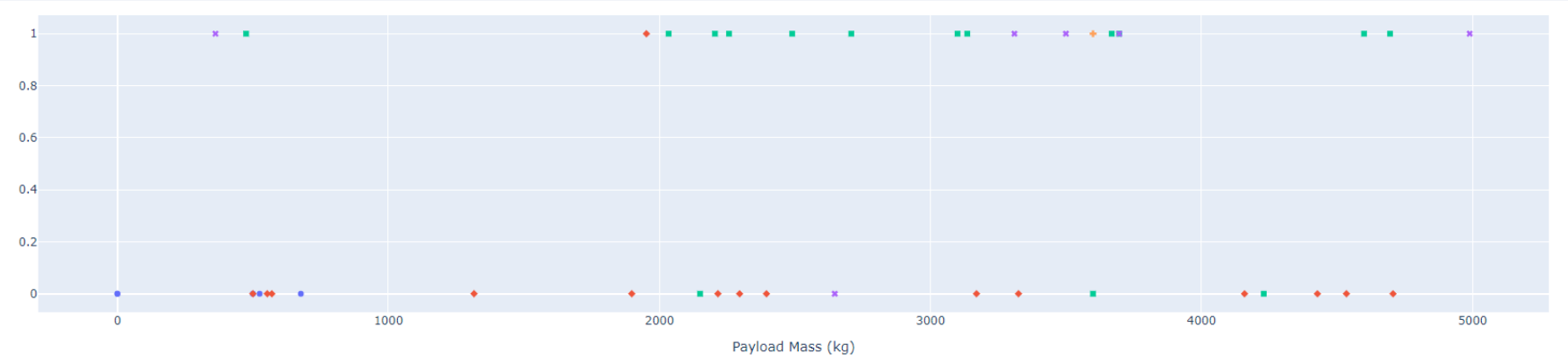
- As you can see from the above figure, KSC LC-39A had the largest successful launches rate among the launch sites.

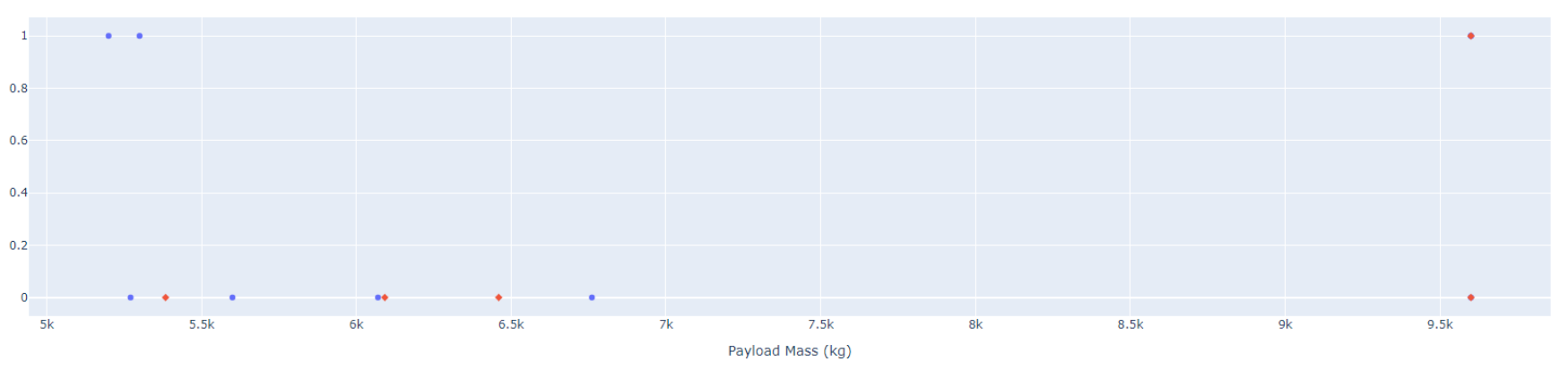
KSC LC-39A Success Ration Pie-chart



- As you can see, KSC LC-39A had a 76.9 % success rate while achieving a 23.1 % failure rate.
- This launch site had the highest success rate among the other launch sites.

Payload vs. Launch Outcome for All Sites within Different Range of Payload





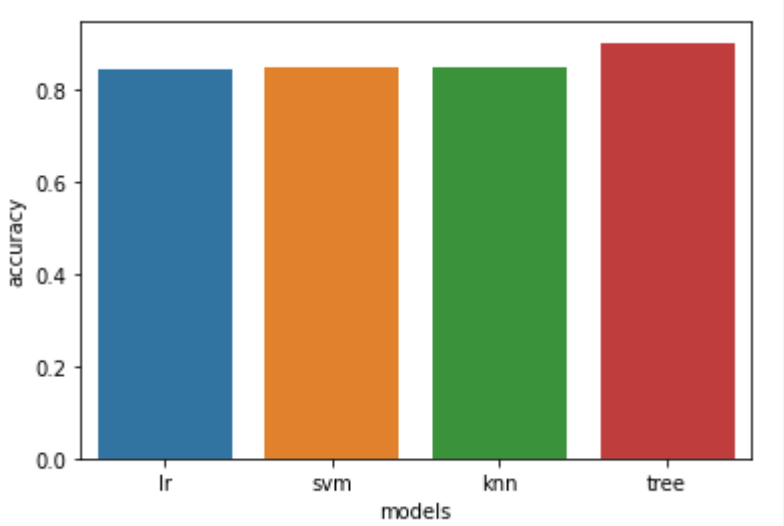
- As you can see, success rate in low payloads is higher than heavy payloads.
- Most of the launches had low weighted payload.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- As you can see from the figure, the Decision tree has the most accuracy among the models with 90 % accuracy.
- So, as the conclusion from the bar chart result and code below, we can deduce that the decision tree has the best performance with 90% accuracy, and you can observe decision tree parameters.

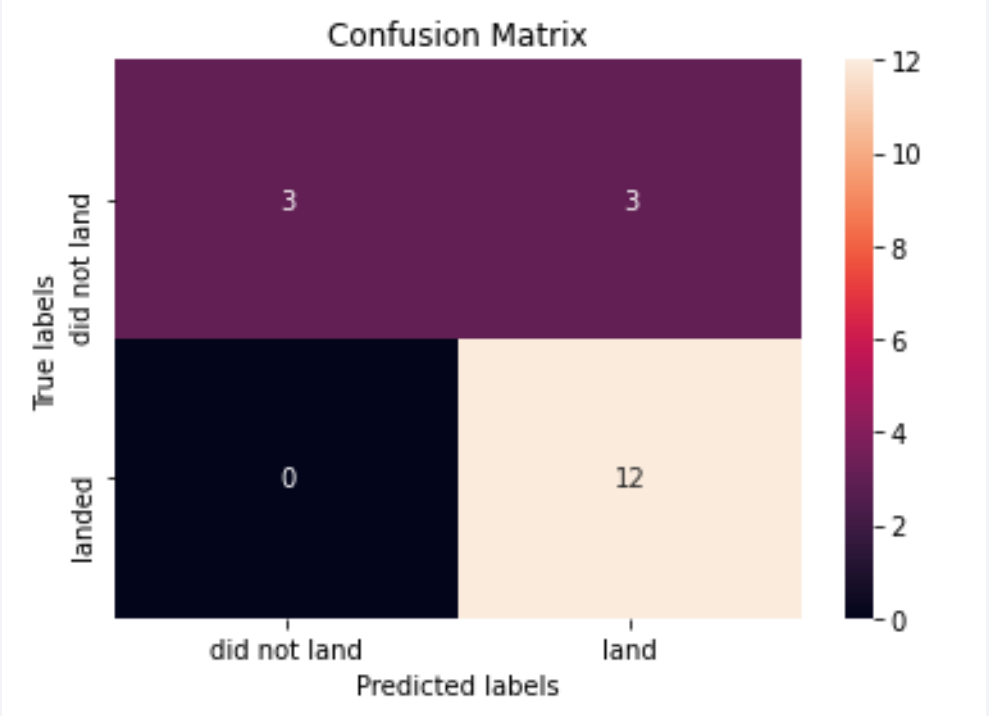



```
best_model = max(models, key= lambda x: models[x].best_score_)
print(f'best model: {models[best_model].best_params_},\naccuracy: {models[best_model].best_score_}')
```

```
best model: {'criterion': 'gini', 'max_depth': 14, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 10, 'splitter': 'best'},
accuracy: 0.9035714285714287
```

Confusion Matrix

- By analyzing the confustion matrix, we can understand that decision tree can distinguish between between different classes. And totally models performs well. But the main problem is False Negatives.
- Class 1 precision is: 80 %
- Class 1 Recall is 100 %



Conclusions

- By exploring data, we found the most correlated columns with the success rate.Those columns are “FlighNumber”, “PayloadMass”, “Orbit”, “LaunchSite”, “Flights”, “GridFins”, “Reused”, “legs”, “LandingPad”, “Block”, “ReusedCount”, “Serial”
- As we saw in proximities analysis, the launch site is near the coastline, railway and highway but far from the nearest city. This information lets us determine which locations are suitable for launching rockets and locating launch sites.
- Predictive analysis results show that among models that we used for training on data, the decision tree has the best performance on the train data with 90% accuracy and 83.4% on the test set.
- As we showed in previous sections, the ES-L1, GEO, HEO, SSO has the most success rate average among other orbit types with all launches were successful
- As we showed, low weighted payloads had more successful launches than the high weighted ones.
- As we showed in EDA section,generally success rates for SpaceX launches increases since 2013.

Appendix

- For predictive analysis, the decision tree model had the best accuracy, so in addition to the models mentioned, we used a random forest classifier to improve model accuracy.
- Random forest is a supervised machine learning algorithm used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.
- After using GridSearchCV, random forest accuracy reached 85% with an 83% score on the test dataset.
- Hyper parameters that have been chosen for analyzing are:

```
rand_clf = RandomForestClassifier(random_state=6)
parameters = {'criterion': ['gini', 'entropy'],
              'n_estimators': [90, 100, 115, 130],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt', 'log2'],
     'min_samples_leaf': range(2, 10, 1),
     'min_samples_split': range(2, 10, 1)}
```

Thank you!