IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

André Roussel
April 2nd, 2025

Outline

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

Executive Summary

- The goal of this project is to predict the success of Space X Falcon 9 first-stage landings to build a business case for reusable first-stage rockets. According to available data, he ability to reuse the first-stage reduces the price of a lauch from roughly $165 millions dollars to around $62 million dollars which is a substantial savings.
- The approach is:
  - Collect data from publicly available databases and websites
  - Explore and wrangle data to analyze and summarize datasets to understand their main characteristics and uncover patterns.
  - Build and compare machine learning modules to predict the success of first-stage landings
- The tools used are Python, Dash, Folium, SQL, Scikit-learn and multiple other data science libraries
- From multiple models, the decision tree shows the most promise with a prediction accuracy on the testing data of 88% *(the decision tree parameters where adjusted due to an issue with the auto criterion in max_features parameter)*.

Introduction

Background

- SpaceX launches are significantly cheaper due to reusability.
- Predicting landing success helps understand launch economics.
- Project explores this using real launch data from various sources.

Problem Statement

- Can we accurately predict whether a Falcon 9 first-stage will land successfully?
- Binary classification problem: 0 = Failure, 1 = Success.
- Focus on identifying key features affecting landing outcome.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collection was done using APIs and Web Scraping
- Perform data wrangling
  - Data was explored to identify missing values, identify characteristics of the data, etc…
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - After the data was cleaned and understood (characteristics), it was standardized, split into training and testing sets, and used to build and compare Logistic Regression, Support Vector Machine (SVM), Classification tree and K nearest neighbor (KNN) models.

Data Collection

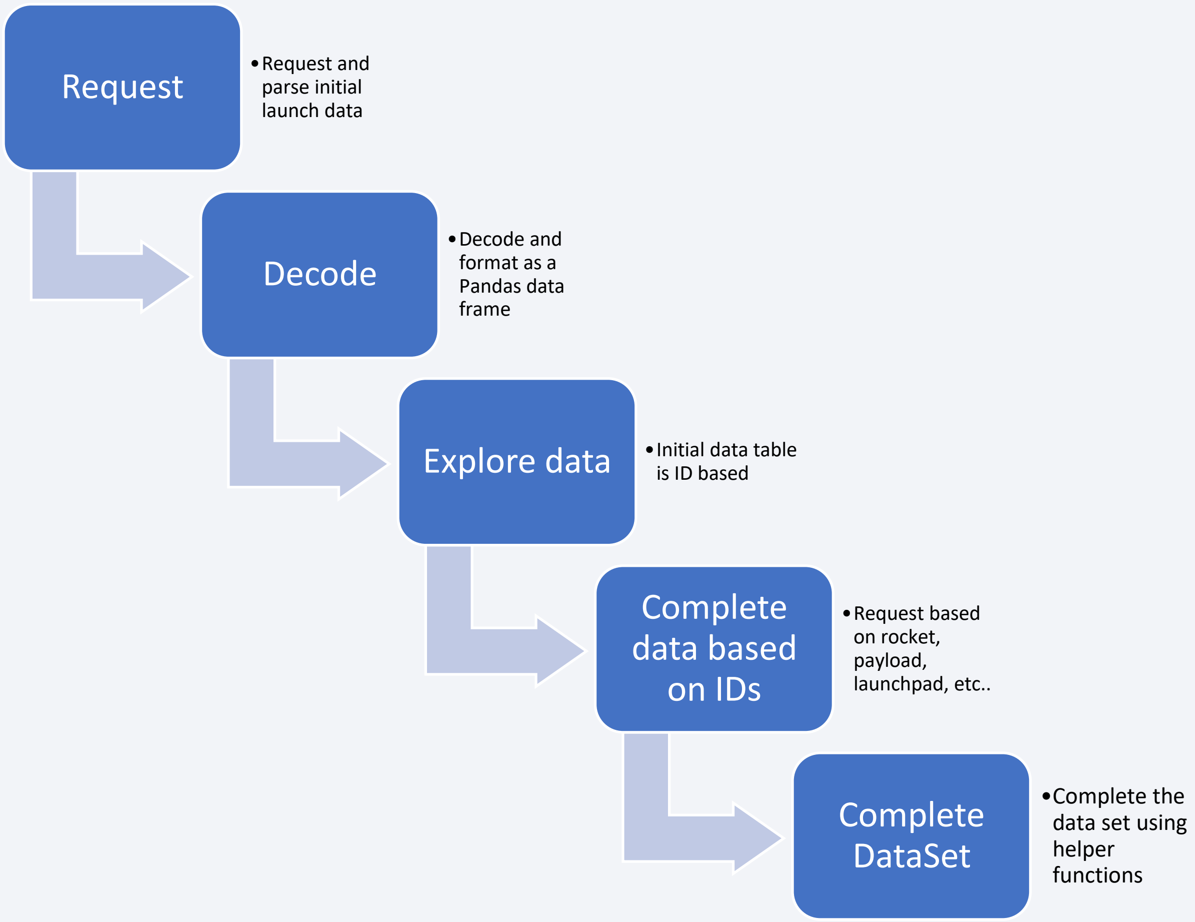
- Data Sources Overview
  - SpaceX REST API – launch data.
  - Wikipedia – rocket details and historical launches.
  - External JSON/CSV datasets.
  - Data integrated and cleaned into a unified pandas DataFrame for modeling.

Data Collection – SpaceX API

- Collected historical launch data using SpaceX REST API.
- Data includes launch site, payload mass, orbit, and landing outcome.
- Sample API response parsed from JSON to DataFrame.
- Challenges: handling inconsistent or missing fields.

- Code available at (may have to download it to visualize it)

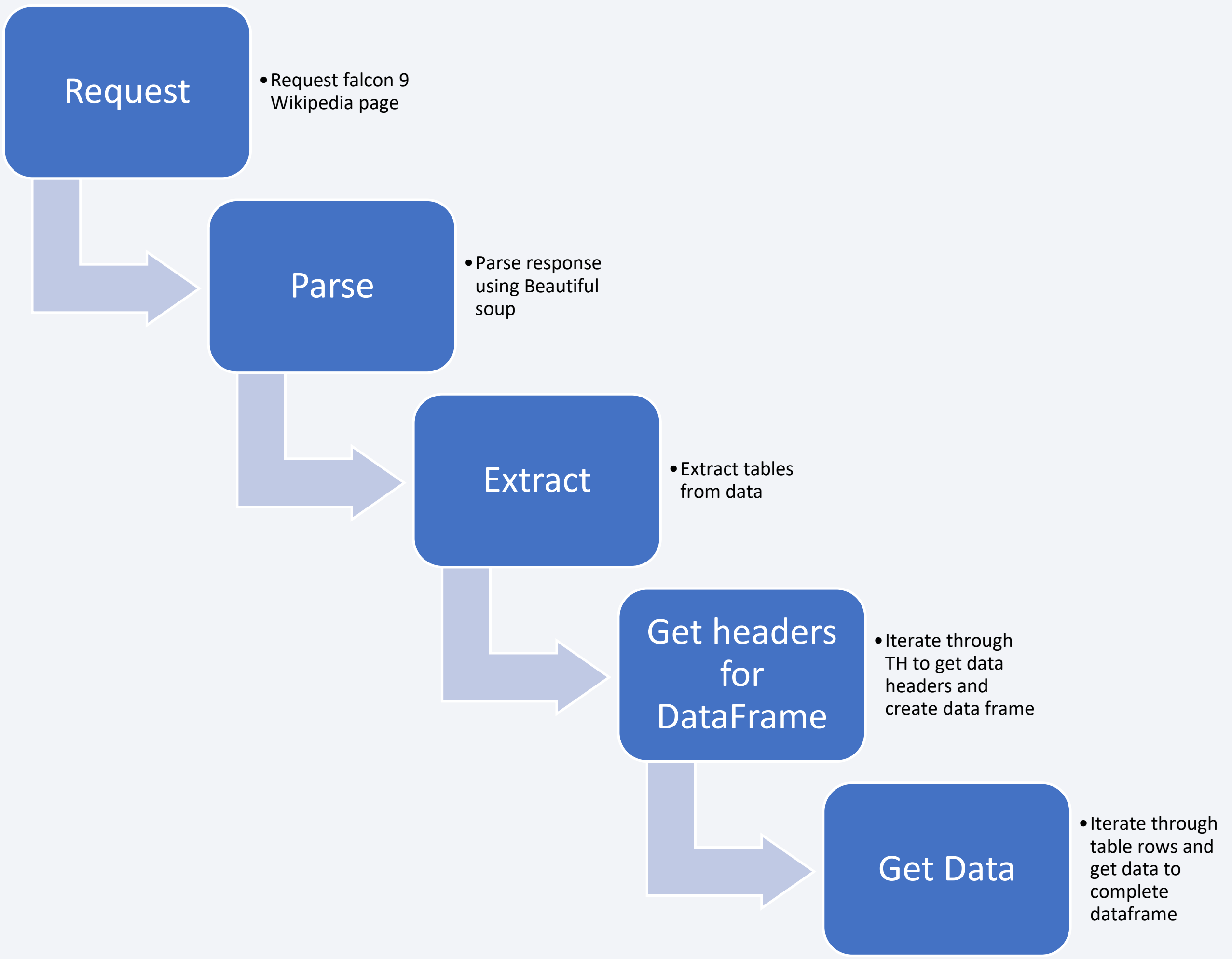
  https://github.com/Andre-Roussel/IBM-Skills-Build-DataScience-Capstone/blob/6fb62b72c62eeebb5b8f185cafbf17010d40c84f/jupyter-labs-spacex-data-collection-api-v2.ipynb



Data Collection - Scraping

- Supplemented API data with Wikipedia scraping for rocket specs.
- Used BeautifulSoup to extract tabular data on Falcon 9 launches.
- Data merged and deduplicated based on flight number and date.
- Code available at (may need to be downloaded):

  https://github.com/Andre-Roussel/IBM-Skills-Build-DataScience-Capstone/blob/6fb62b72c62eeebb5b8f185cafbf17010d40c84f/jupyter-labs-webscraping.ipynb



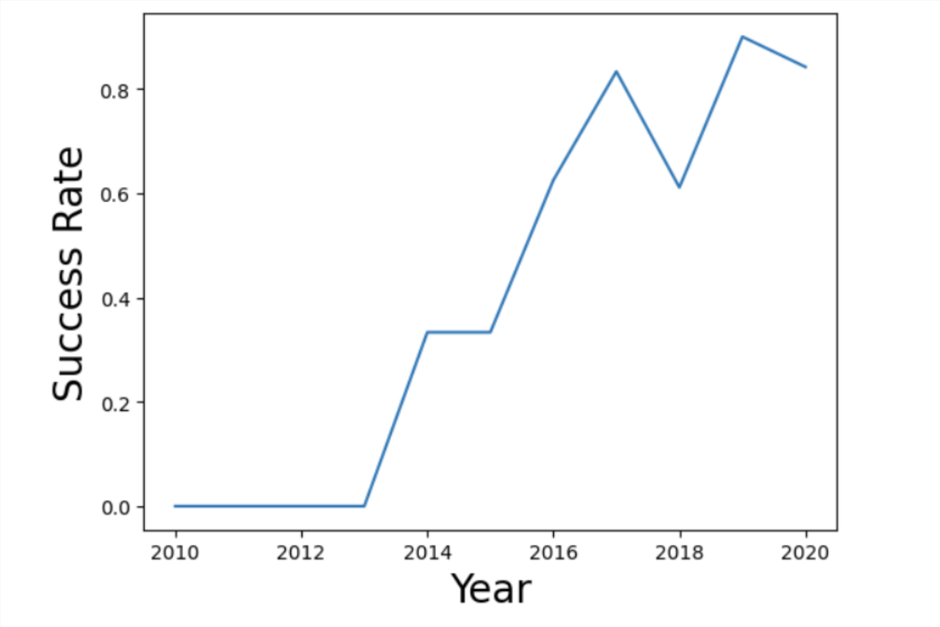
Data Wrangling

- Merged datasets from API and web sources.
- Dropped nulls and redundant columns.
- Created new features: booster version, landing pad, etc.
- Filtered for Falcon 9 missions only.

EDA with Data Visualization

- Used Seaborn and Matplotlib for visual analysis.
- Key charts: payload vs. success, launch site success rate, orbit type trends.
- Detected correlations and candidate features for modeling.
- Code available at (may need to be downloaded):

  https://github.com/Andre-Roussel/IBM-Skills-Build-DataScience-Capstone/blob/main/jupyter-labs-eda-dataviz-v2.ipynb



EDA with SQL

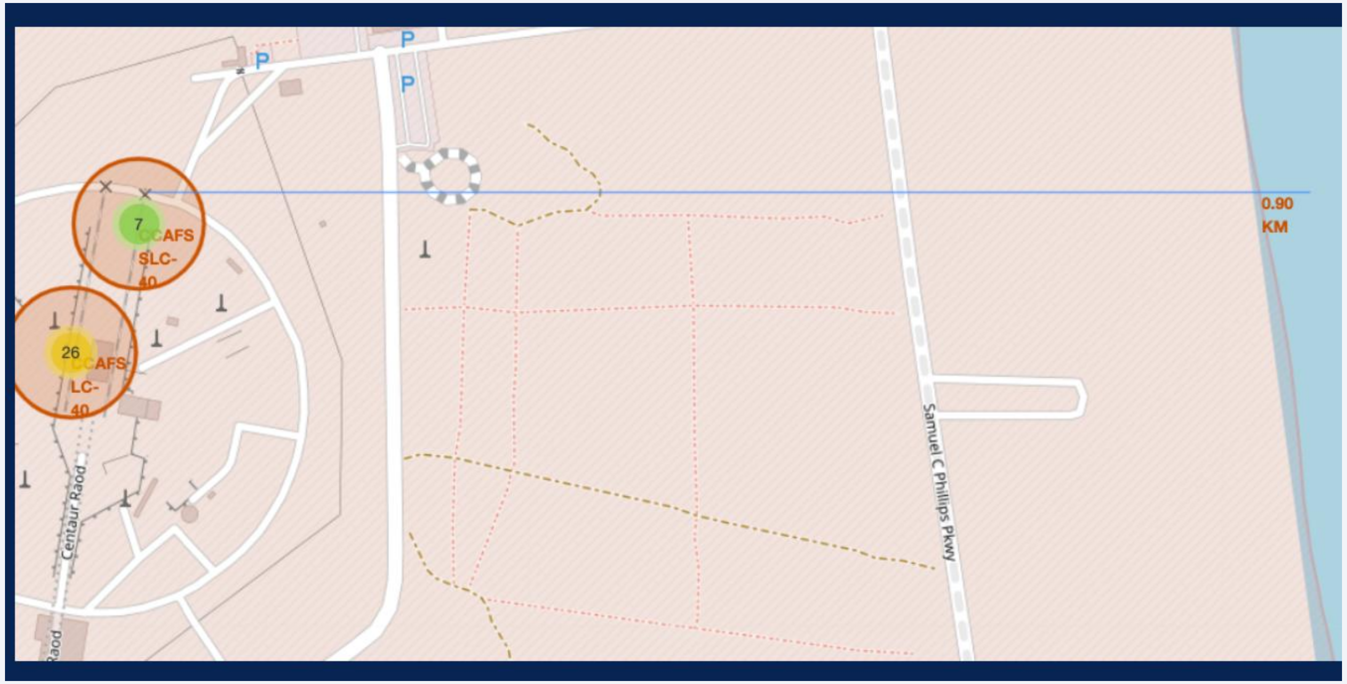
- Queried launch site names, payload distributions, and mission outcomes.
- Example: total payload mass from NASA launches.
  - %sql select sum(PAYLOAD_MASS__KG_) as Total_Payload_Mass from SPACEXTABLE where Customer = 'NASA (CRS)'
- Used SQLite to interact with structured dataset and confirm patterns.
- Code available at (may need to be downloaded):

  https://github.com/Andre-Roussel/IBM-Skills-Build-DataScience-Capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

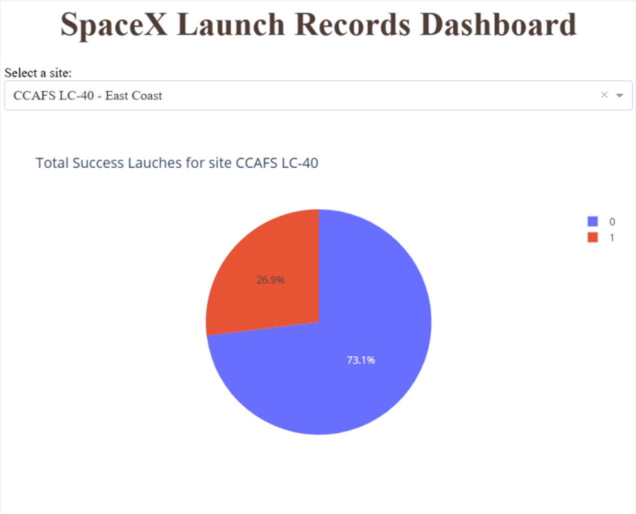
- Mapped launch sites using Folium.
- Markers show site names, success/failure, and coordinates.
- Used Circle and PolyLine to highlight proximity to coastlines, highways.
- Code available at (may need to be downloaded):

  https://github.com/Andre-Roussel/IBM-Skills-Build-DataScience-Capstone/blob/main/lab-jupyter-launch-site-location-v2.ipynb



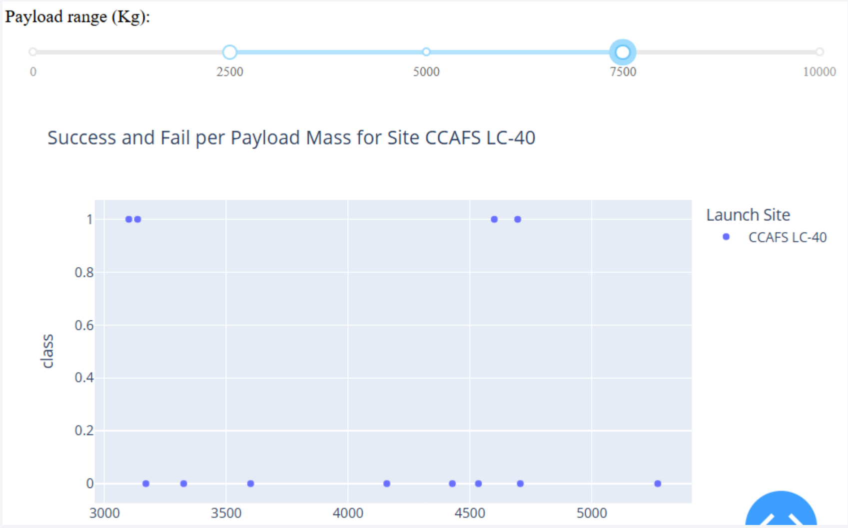
Build a Dashboard with Plotly Dash

- Pie chart shows success count per site or for all sites combined.
  - Highlights which sites have higher reliability.
- Scatter plot visualizes relationship between payload mass and landing outcome.
  - Interactive payload slider allows dynamic filtering.
  - Insights: mid-range payloads often show higher success.



- Code available at (may need to be downloaded):

https://github.com/Andre-Roussel/IBM-Skills-Build-DataScience-Capstone/blob/main/spacex_dash_app.py



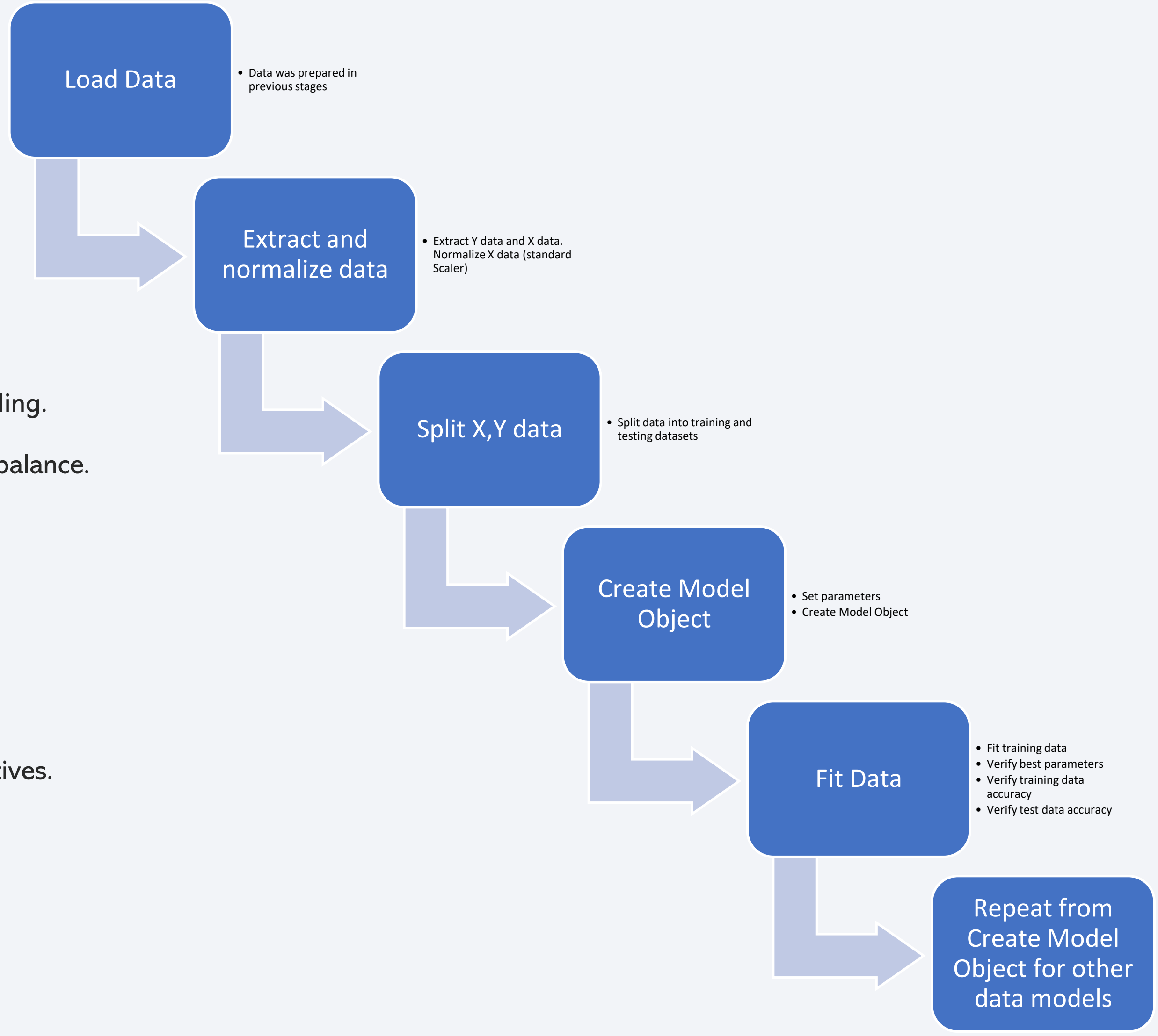
Note: This was coded to run on my RaspberryPi at home which was actually kinda fun…

Predictive Analysis (Classification)

- Overview
  - Objective: classify if a launch will be successful.
  - Features: payload, orbit, booster version, launch site, etc.
  - Target: landing success (1) or failure (0).
  - Models tested: Logistic Regression, Decision Tree, SVM, KNN
- Model building and Preprocessing
  - Preprocessing steps: encoding categorical variables, feature scaling.
  - Train-test split: 80/20 ratio with stratification to preserve class balance.
  - StandardScaler used for numerical features like payload mass.
  - OneHotEncoder used for orbit, launch site, and booster version.
- Model Evaluation Metrics
  - Used Accuracy, Precision, Recall, and F1-Score.
  - Confusion Matrix used to analyze false positives and false negatives.
  - Cross-validation applied to improve reliability of scores.

- Code available at (may need to be downloaded):

https://github.com/Andre-Roussel/IBM-Skills-Build-DataScience-Capstone/blob/main/SpaceX-Machine-Learning-Prediction-Part-5-v1.ipynb



Results

The next section, the following topics are going to be explored

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

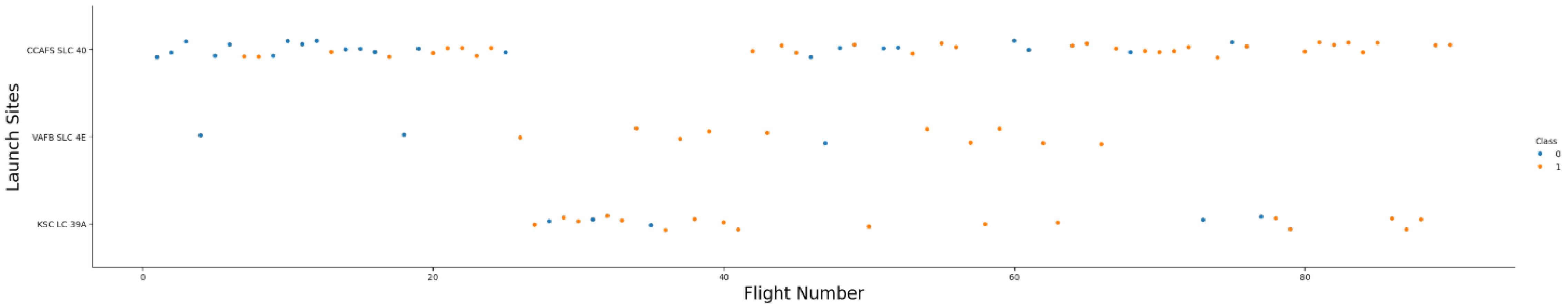
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

```python
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Sites",fontsize=20)
plt.show()
```

✓ 0.1s Python



Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

As the flight number increases, the more succesful are the landings which makes sense since greather experience and knowledge is accumulated. Flights from site CCAFS where stopped (flights 26 to 41) to the benefit of site KSC. Site VAFB was not used after flight 66. The ration of success seems to increase sharply after flight 50

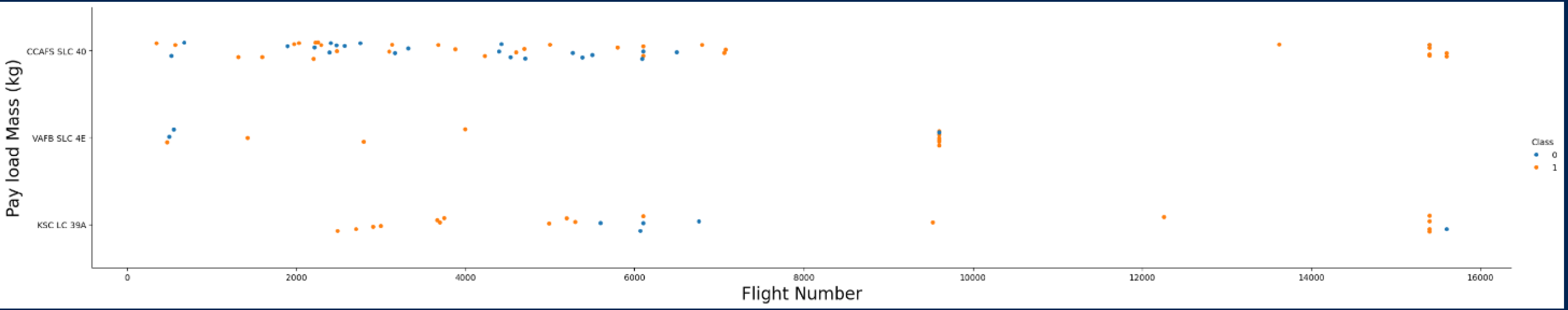
markdown

Payload vs. Launch Site

```python
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value

sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```

Python



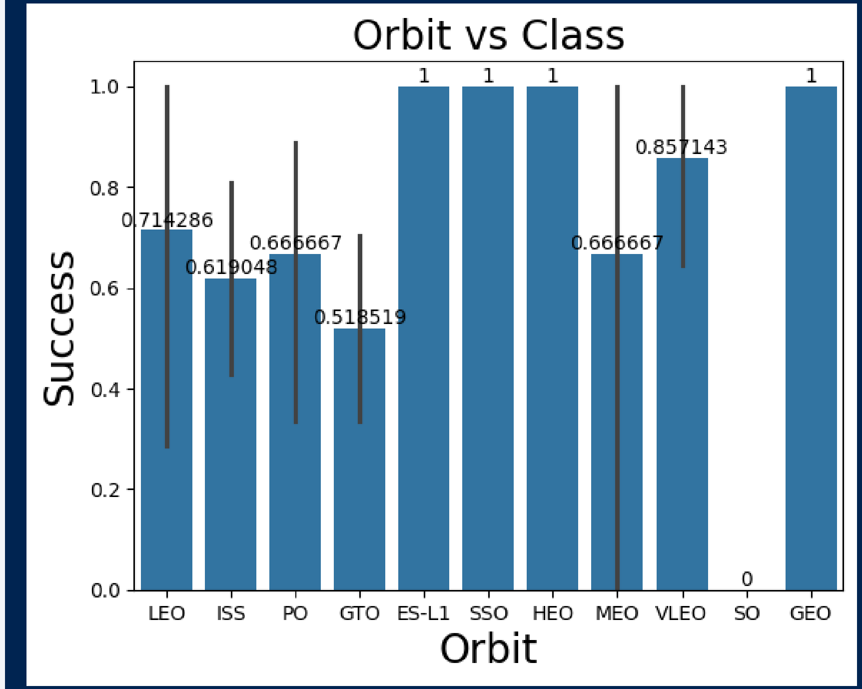

```
Launch site VAFB has no launches for payloads greather tahn 9600 kg.Also site CCAFS was not used to launch payloads between 10 000 kg and arounf 13500 kg.  Higher payload seem to have a higher coralation to success but that might also be due to latter flights with more knowledge and experience.  Site CCAFS and KSC seem to be able to lauch the heavyer paylaods
```

markdown

Success Rate vs. Orbit Type

```
sns.barplot( data=df, y="Class", x="Orbit")
plt.xlabel("Orbit",fontsize=20)
plt.ylabel("Success",fontsize=20)
plt.bar_label(plt.gca().containers[0])
plt.title("Orbit vs Class",fontsize=20)
plt.show()
```

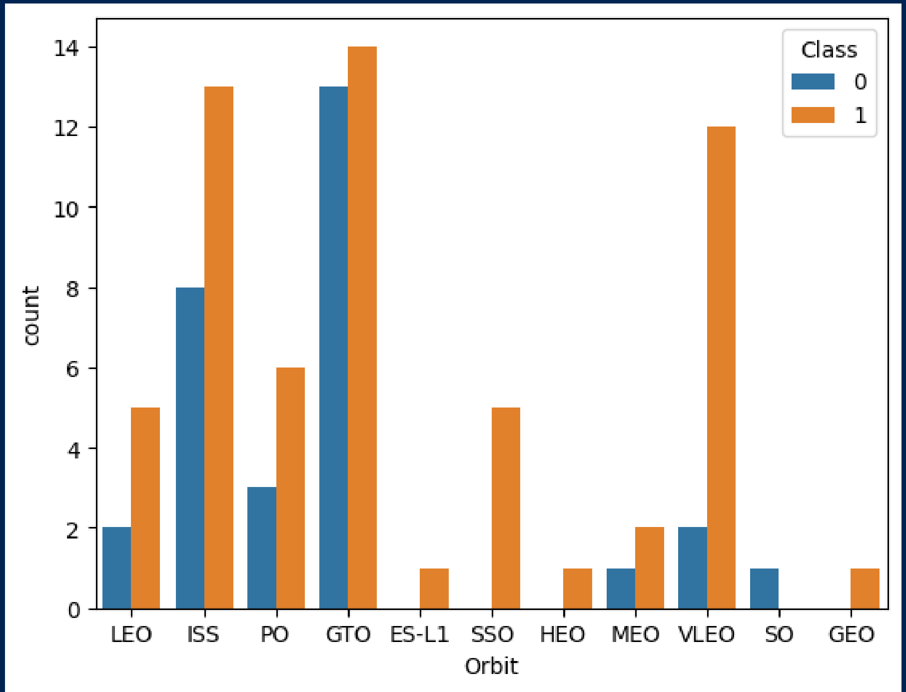
✓ 0.1s




```
sns.countplot(data=df, x="Orbit", hue="Class")
```

✓ 0.1s

```
<Axes: xlabel='Orbit', ylabel='count'>
```

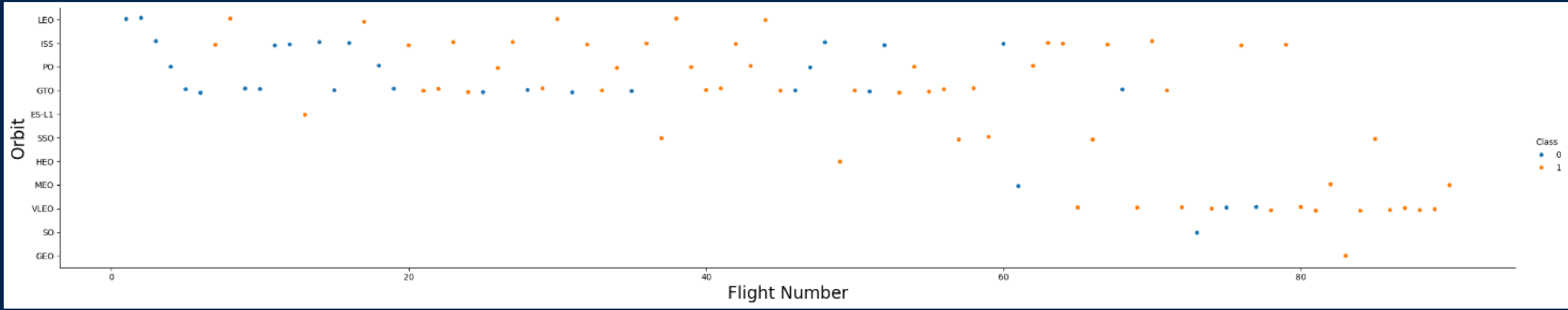


From the ration chart, the ES-L1, SSO, HEO and GEO have the highest success rate but they are also the ones with the least number of flights (1 for each).
SO orbit flights have never been succesful. VLEO Orbit has one of the best ratio of success

Flight Number vs. Orbit Type

```python
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```

Python



LEO success seems to be related to flight number but it is the same with VLEO (Low earth orbit and very low earth orbit). GTO orbit does not seem to be related to flight number

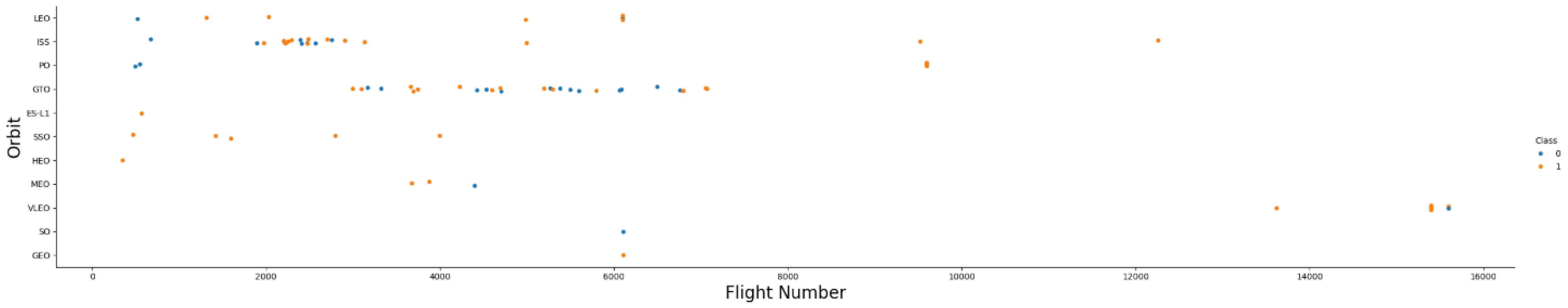
markdown

Payload vs. Orbit Type

Similarly, we can plot the Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type

```python
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```

Python



With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

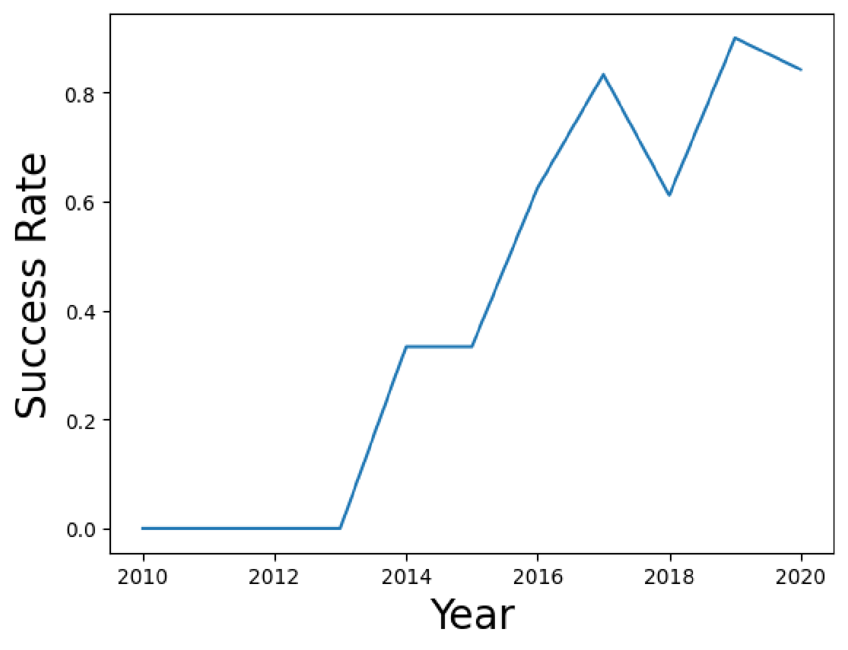
However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
year = Extract_year(df["Date"])
df["Year"] = year
df["Year"] = pd.to_numeric(df["Year"])
success_rate = df.groupby("Year")["Class"].mean()
success_rate.plot()
plt.xlabel("Year",fontsize=20)
plt.ylabel("Success Rate",fontsize=20)
plt.show()
```

[11]

...



You can observe that the success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.

All Launch Site Names

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

[28] ✓ 0.0s

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

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

Using the SQLite distinct clause, we can extract the unique Launch Sites.

Launch Site Names Begin with 'CCA'

```
#%sql select * from SPACEXTABLE where Booster_Version like '%F9 v1.1%'
%sql select * from SPACEXTABLE where Launch_Site like '%CCA%' Limit 5
```

✓ 0.0s

```
* sqlite:///my_data1.db
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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

The like clause enable lookup of fields containing a specific string. The Limit clause limits the number of output rows

Total Payload Mass

Display the total payload mass carried by boosters launched by NASA (CRS)

```
%sql select sum(PAYLOAD_MASS__KG_) as Total_Payload_Mass from SPACEXTABLE where Customer = 'NASA (CRS)'
```

✓ 0.0s

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

Total_Payload_Mass
45596

The sum clause enable the sum of all records meeting the where clause 'NASA (CRS)'

Average Payload Mass by F9 v1.1

Display average payload mass carried by booster version F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) as AVG_Payload_Mass from SPACEXTABLE where Booster_Version = 'F9 v1.1'
```

[4] ✓ 0.0s

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

AVG_Payload_Mass
2928.4

The avg clause averages the rows of the result set where the Booster Version is 'F0 v1.1'

First Successful Ground Landing Date

```
%sql select * from SPACEXTABLE where Landing_Outcome = 'Success (ground pad)' order by Date asc limit 5
```

✓ 0.0s Python

* sqlite:///my_data1.db
Done.

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)
2016-07-18	4:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-05-01	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
2017-06-03	21:07:00	F9 FT B1035.1	KSC LC-39A	SpaceX CRS-11	2708	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)

By using the Landing Outcome 'Success (ground pad)' in the where clause, ordering by Date asc with a limit of 5 to check the results, we see that the first succesful landing on a ground pad was on december 22nd 2015.

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
%sql select * from SPACEXTABLE where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000 order by Date asc
```

✓ 0.0s

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2016-05-06	5:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-08-14	5:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
2017-10-11	22:53:00	F9 FT B1031.2	KSC LC-39A	SES-11 / EchoStar 105	5200	GTO	SES EchoStar	Success	Success (drone ship)

By using the and clause, the data for a payload mass in the range of 4000 to 6000 kg

can be seclected where the Drone ship have been succesful.

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
%sql select Mission_Outcome , count(*) as 'Mission Outcomes' from  SPACEXTABLE  group by Mission_Outcome
#%sql select distinct Mission_Outcome from  SPACEXTABLE
```

✓ 0.0s

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

Mission_Outcome	Mission Outcomes
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Use the count clause to get the count of different mission outcomes. There are two Success mission outcomes are related

to the fact the for the client USAF, there is an extra space at the end od the 'sucess ' string.

That should be corrected if we further need those results.

Boosters Carried Maximum Payload

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

[54] ✓ 0.0s

* sqlite:///my_data1.db
Done.

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2019-11-11	14:56:00	F9 B5 B1048.4	CCAFS SLC-40	Starlink 1 v1.0, SpaceX CRS-19	15600	LEO	SpaceX	Success	Success
2020-01-07	2:33:00	F9 B5 B1049.4	CCAFS SLC-40	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600	LEO	SpaceX	Success	Success
2020-01-29	14:07:00	F9 B5 B1051.3	CCAFS SLC-40	Starlink 3 v1.0, Starlink 4 v1.0	15600	LEO	SpaceX	Success	Success
2020-02-17	15:05:00	F9 B5 B1056.4	CCAFS SLC-40	Starlink 4 v1.0, SpaceX CRS-20	15600	LEO	SpaceX	Success	Failure
2020-03-18	12:16:00	F9 B5 B1048.5	KSC LC-39A	Starlink 5 v1.0, Starlink 6 v1.0	15600	LEO	SpaceX	Success	Failure
2020-04-22	19:30:00	F9 B5 B1051.4	KSC LC-39A	Starlink 6 v1.0, Crew Dragon Demo-2	15600	LEO	SpaceX	Success	Success
2020-06-04	1:25:00	F9 B5 B1049.5	CCAFS SLC-40	Starlink 7 v1.0, Starlink 8 v1.0	15600	LEO	SpaceX, Planet Labs	Success	Success
2020-09-03	12:46:14	F9 B5 B1060.2	KSC LC-39A	Starlink 11 v1.0, Starlink 12 v1.0	15600	LEO	SpaceX	Success	Success
2020-10-06	11:29:34	F9 B5 B1058.3	KSC LC-39A	Starlink 12 v1.0, Starlink 13 v1.0	15600	LEO	SpaceX	Success	Success
2020-10-18	12:25:57	F9 B5 B1051.6	KSC LC-39A	Starlink 13 v1.0, Starlink 14 v1.0	15600	LEO	SpaceX	Success	Success
2020-10-24	15:31:34	F9 B5 B1060.3	CCAFS SLC-40	Starlink 14 v1.0, GPS III-04	15600	LEO	SpaceX	Success	Success
2020-11-25	2:13:00	F9 B5 B1049.7	CCAFS SLC-40	Starlink 15 v1.0, SpaceX CRS-21	15600	LEO	SpaceX	Success	Success

Using the max clause in a subquery, we can extract the max Payload mass and use it as the where clause value

2015 Launch Records

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
%%sql select substr(Date,6,2) as MonthName,* from spacextable where substr(Date,0,5) = '2015' and landing_outcome = 'Failure (drone ship)'
```

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

MonthName	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
01	2015-01-10	9:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
04	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)

Using the substr function, we can extract the proper month and date with the desired landing outcome 'Failure (drone ship)'

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

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

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

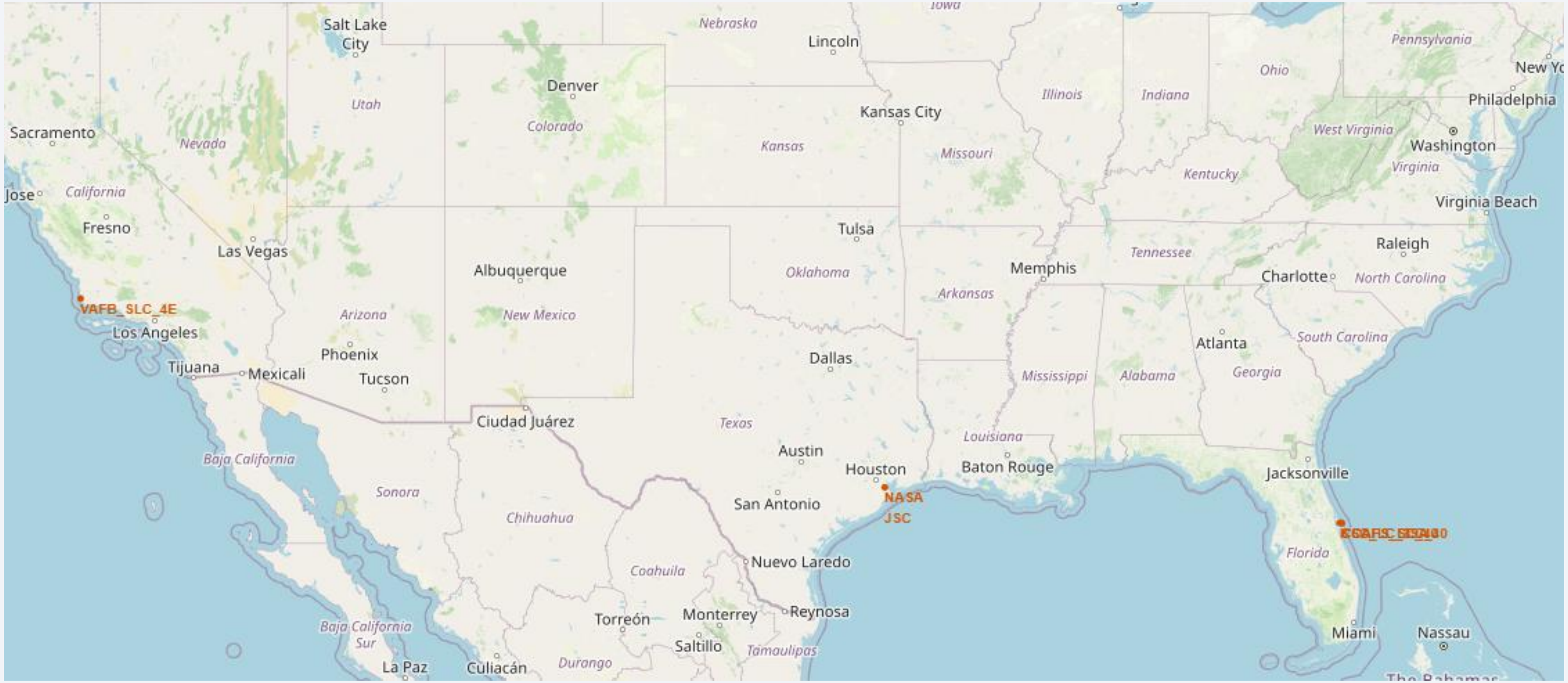
Landing_Outcome	Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Using the count clause, the where clause to select the proper dates, grouping by Landing Outcome and then ordering, ww get the desired results.

Section 3

Launch Sites Proximities Analysis

Map of Launch Sites including Huston NASA site



The first thing that we notice is that the launch sites are are as south as possible in the US, very flat areas, close to the sea and with low population and traffic.

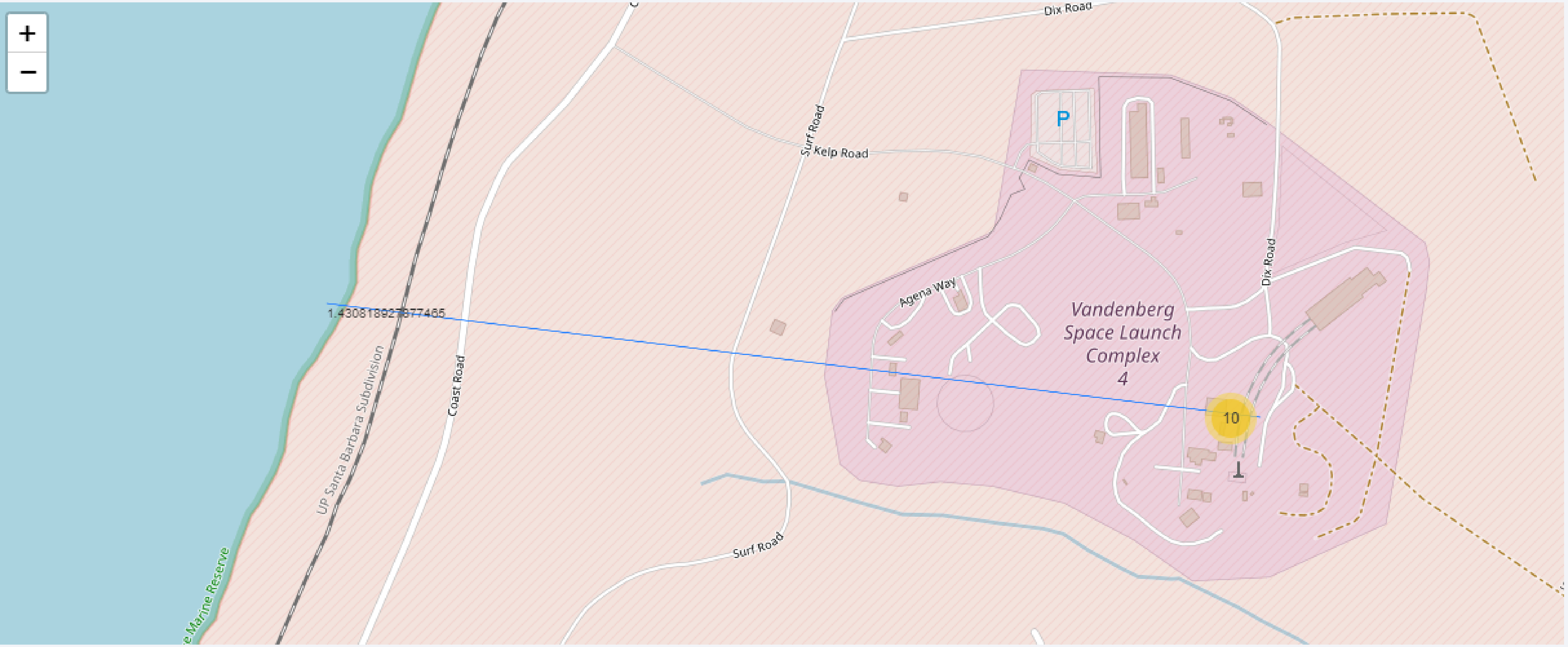
Color coded launch outcomes of all 4 sites.

Cap Canaveral Sites

Vandenberg Site

- The most used site it Cap Canaveral. It also has the most successful launch outcomes

Distance from Vandenberg Launch Site to Sea

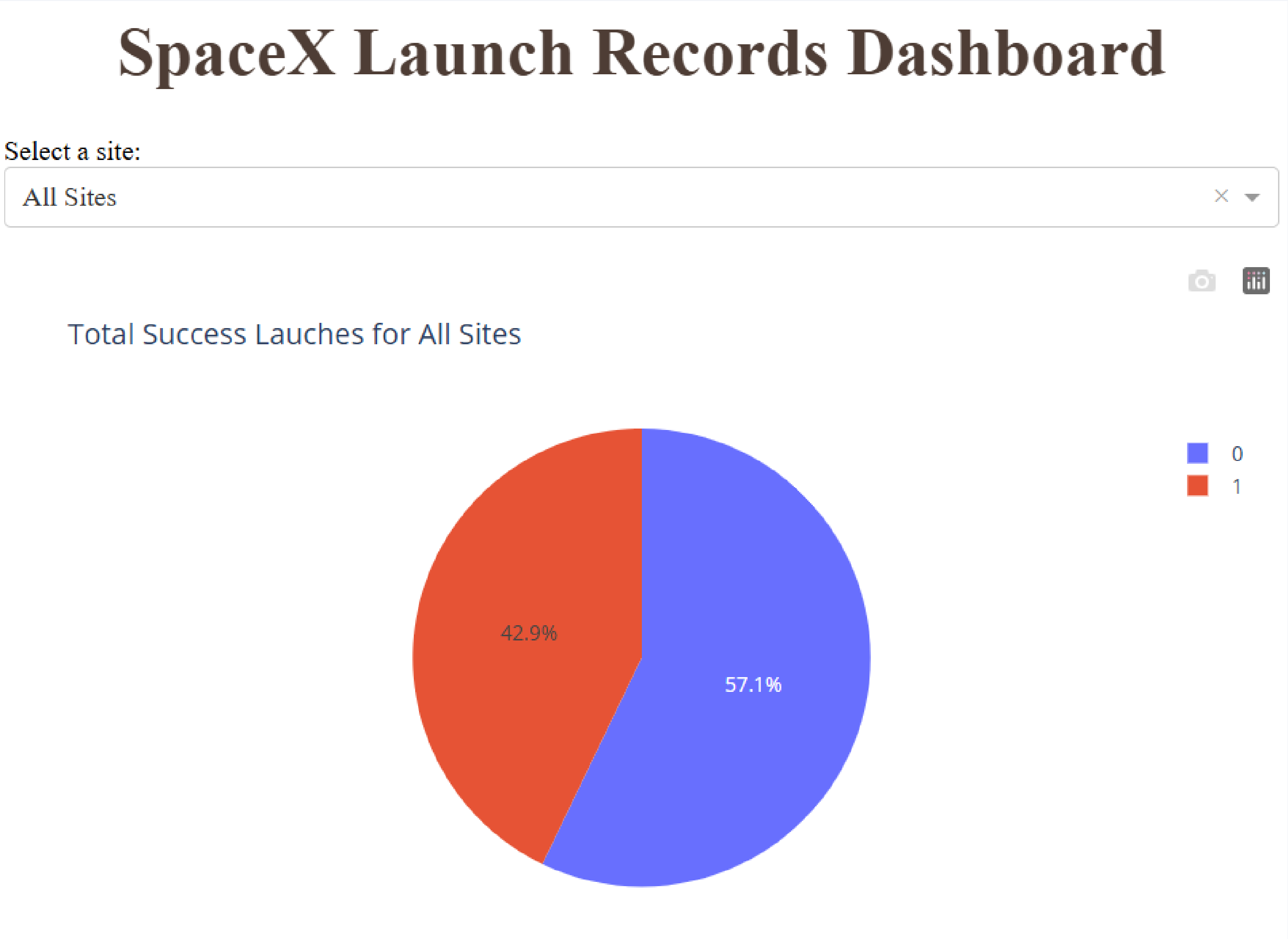


Can be used to calculate distance to point of interest for informational, environmental or safety pruposes.

Section 4

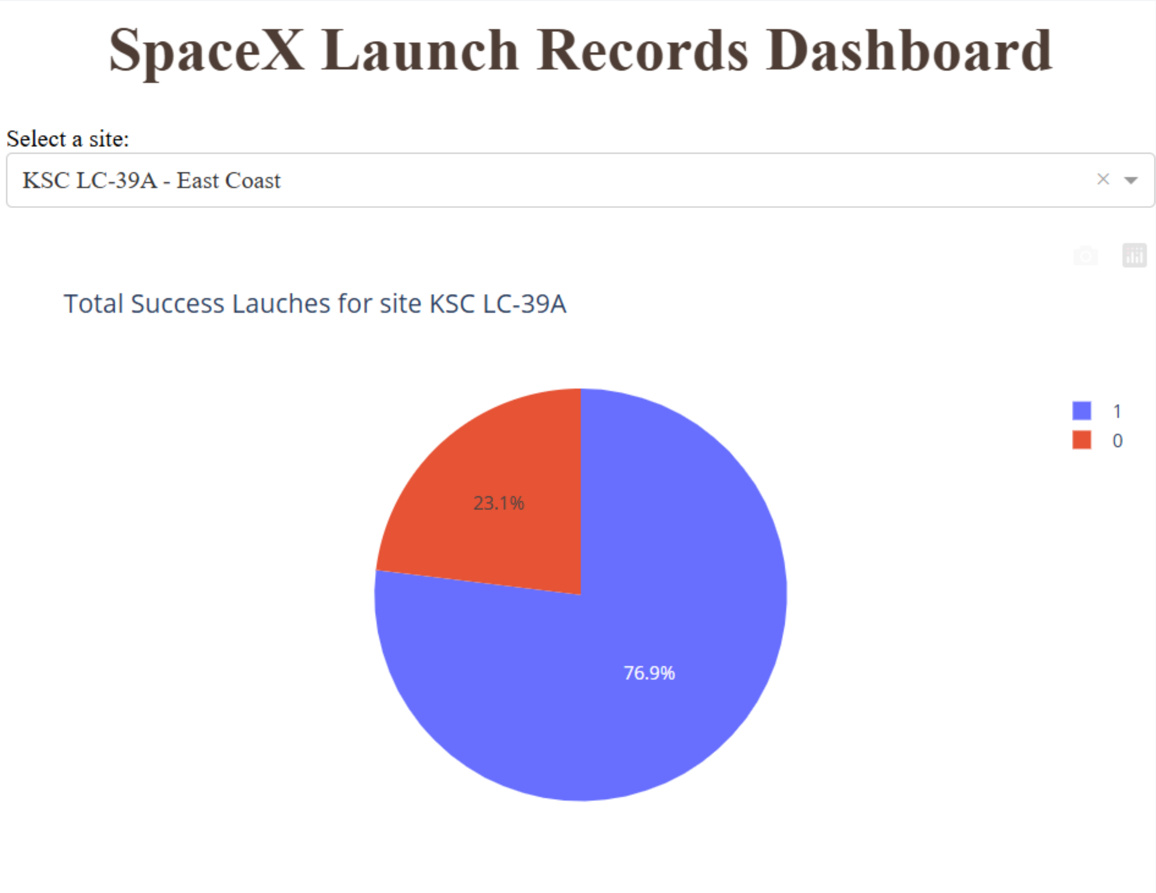
Build a Dashboard with Plotly Dash

Succes and Failure for All Sites



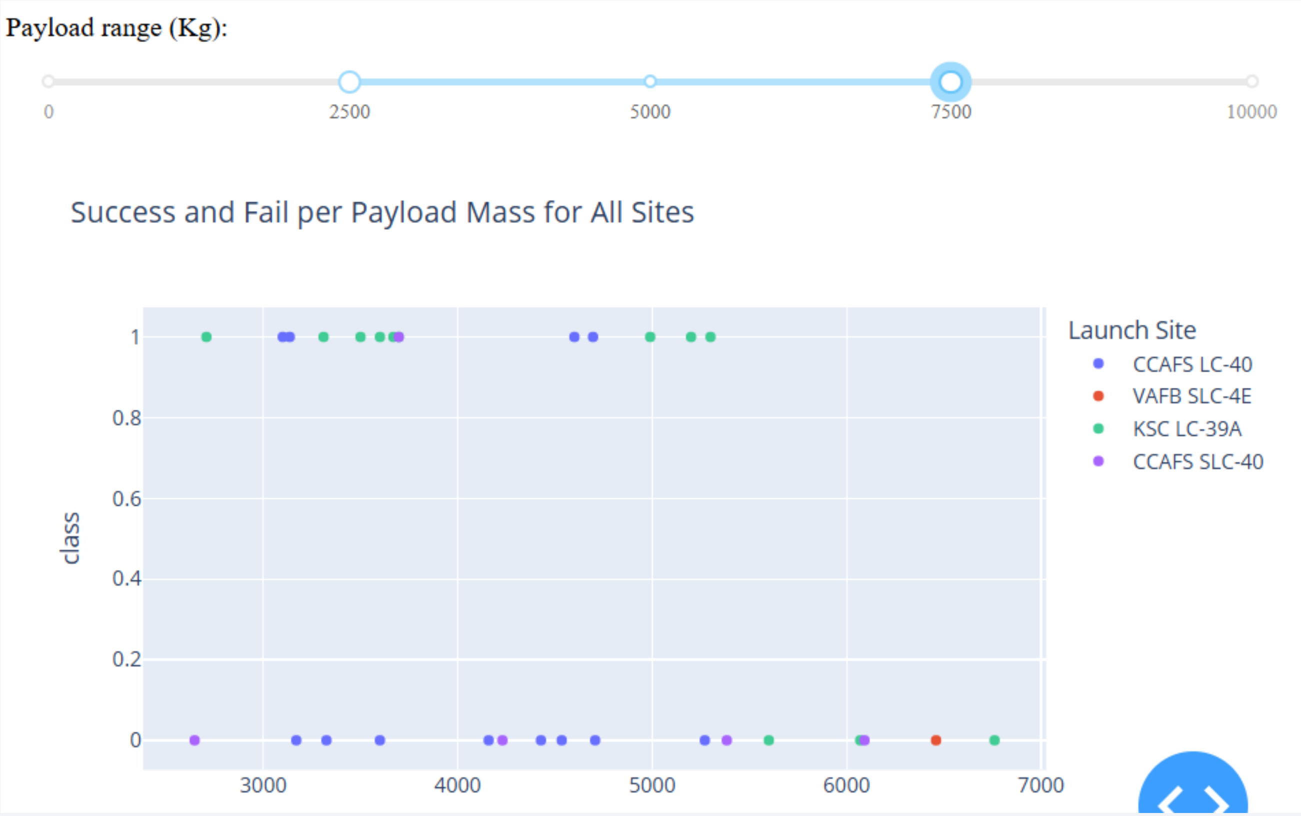
- The conclusion from this is that it is a very expensive undertaking as only 57.1% of the outcomes are succesful

Site with the best outcome



- The launch site with the best outcome is KSC LC-39A.

Outcomes for all site for selected payload.

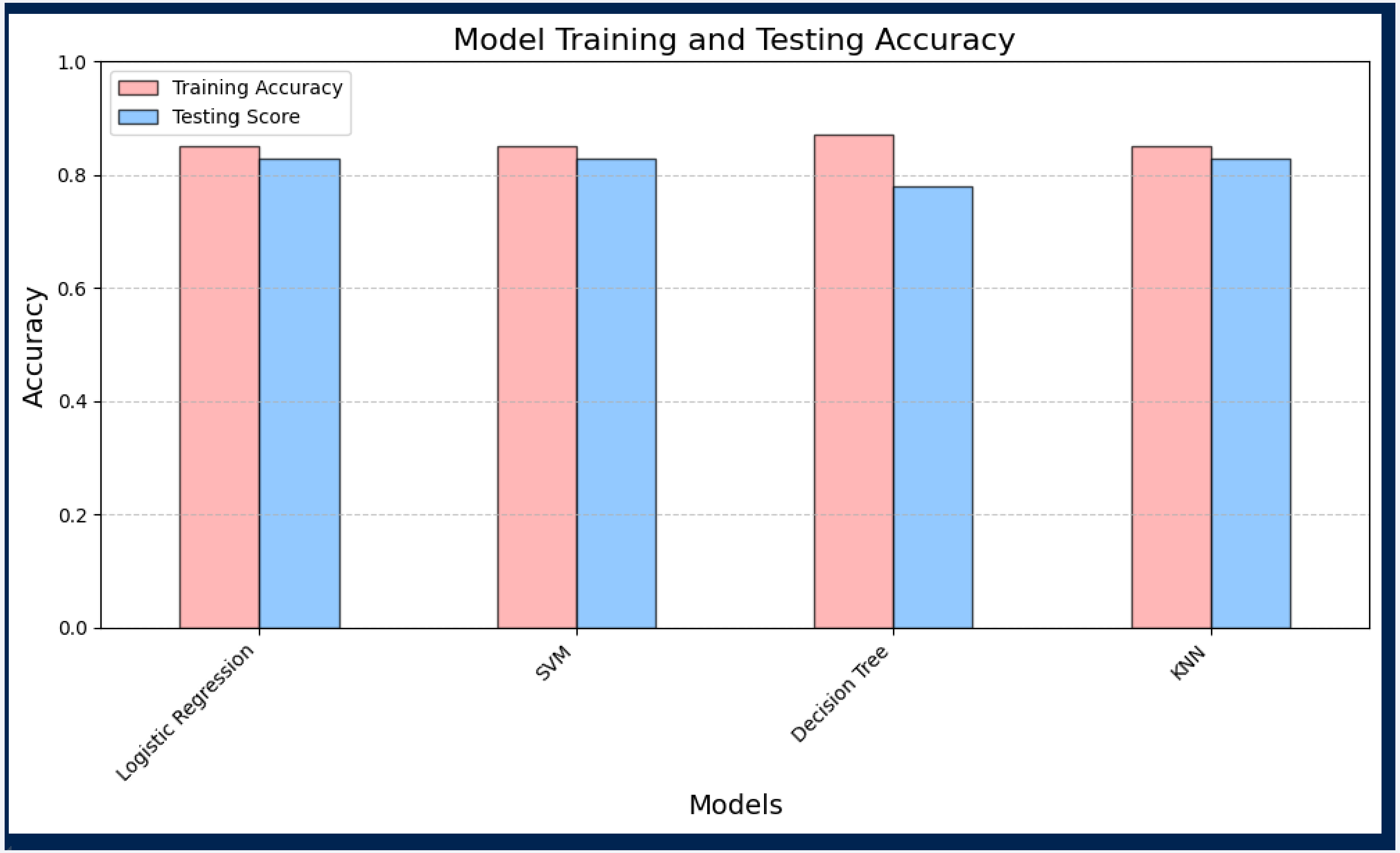


- Outcomes for all sites for Payloads between 2500 and 7500 kg. Not very successful is it?

Section 5

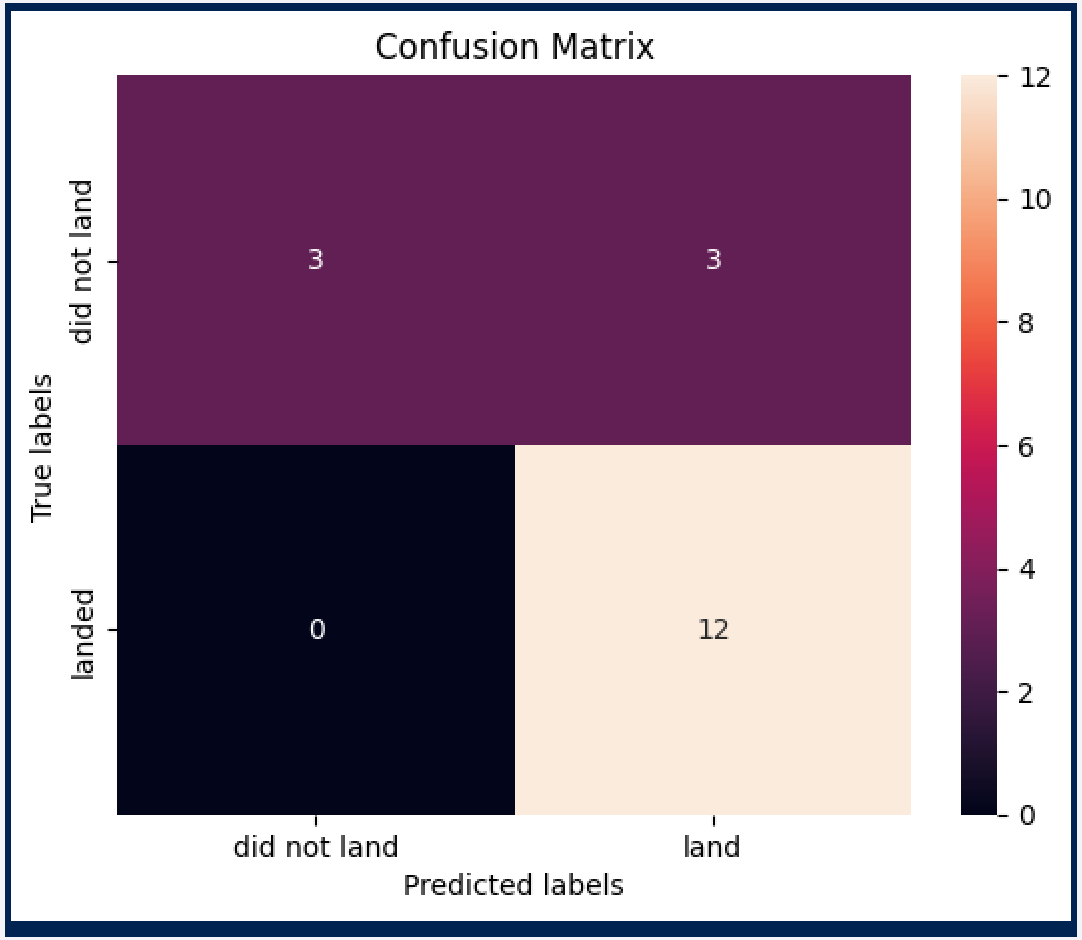
Predictive Analysis (Classification)

Classification Accuracy



The decision tree with the Log2, sqrt and none has the highest training accuracy but the lowest score on the test data. All other models have the same score for the test data.

Confusion Matrix



The Logistic Regression, SVM and KNN models essentially perform all the same on the testing data by correctly predicting the non successful launches and by mis predicting 3 successful landing when the where actually no successful.

Conclusions

Overall Project Results

- Achieved 83% score on testing data for LogReg, SVM and KNN model and only 77% on Tree classifier model.
- Identified payload mass and site as top predictors.
- Interactive dashboards and geospatial analysis enriched insights.

Business Implications

- Predictive models help forecast cost and risk.
- Can be used by competitors or partners to benchmark performance.
- Supports data-driven decision-making in aerospace logistics.

Business Implications

- Predictive models help forecast cost and risk.
- Can be used by competitors or partners to benchmark performance.
- Supports data-driven decision-making in aerospace logistics.

Recommendations

- Incorporate live API data for real-time prediction updates.
- Expand dataset with additional providers for broader comparison.
- Deploy the model in a monitoring dashboard for internal use.

Appendix

- All notebooks are available at https://github.com/Andre-Roussel/IBM-Skills-Build-DataScience-Capstone

Thank you!