IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Shilpa Nidhi Kirubanidhi
Jan 13, 2022

Outline

Executive Summary

- In this capstone, the methodology will be to use the SpaceX data to analyze and predict the cost of each launch for the Company SpaceY. We will also be analyzing the launch details of SpaceX rockets.
- We have performed a complete analysis of SpaceX data and built dashboards to find the relationship between various factors affecting the success rate of the launches. We have also presented folium maps to find the proximities of each launch site to the nearest landmarks.

Introduction

- Project background and context:

The commercial space age is here, companies are making space travel affordable for everyone. Blue Origin manufactures sub-orbital and orbital reusable rockets. Perhaps the most successful is SpaceX. SpaceX's accomplishments include: Sending spacecraft to the International Space Station. Starlink, a satellite internet constellation providing satellite Internet access. One reason SpaceX can do this is the rocket launches are relatively inexpensive. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards of 165 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, we can determine the cost of a launch.

- Problems to find answers for:

1. The cost of each launch
2. The relationship between various factors of the launches like number of flights, payload and the success rate
3. In-detail analysis of rocket launch sites.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data is collected from SPACEX REST API api.spacexdata.com/v4/launches/past. The response will be in the form of a JSON (a list of JSON objects). We use the json_normalize() function to convert it into a dataframe.
- Perform data wrangling
  - We will use beautiful soup object to read the data from the HTML tables , and create a meaningful dataframe. We will remove the unnecessary fields out of the dataframe and also remove NULL values.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

- In this capstone assignment, we will be working with SpaceX launch data that is gathered from an API, specifically the SpaceX REST API. This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome. The SpaceX REST API endpoints, or URL, starts with api.spacexdata.com/v4/. We have the different end points, for example: /capsules and /cores We will be working with the endpoint api.spacexdata.com/v4/launches/past.
- Since we are using an API, we get a response it is in the form of a JSON. Specifically, we have a list of JSON objects which each represent a launch. To convert this JSON to a dataframe, we can use the json_normalize function. This function will allow us to "normalize" the structured json data into a flat table.

Data Collection – SpaceX API

- We use the below functions:
- getBoosterVersion: Takes the dataset and uses the rocket column to call the API and append the data to the list
- getLaunchSite: Takes the dataset and uses the launchpad column to call the API and append the data to the list
- getPayloadData: Takes the dataset and uses the payloads column to call the API and append the data to the lists
- getCoreData: Takes the dataset and uses the cores column to call the API and append the data to the lists
- GITHUB link:

  https://github.com/ShilpaKirubanidhi/IBM-SpaceX-Capstone/blob/b34bb89135013b1ca78dd106176345b18125e4d5/Spacex%20data%20collection%20API.ipynb

Import required libraries and define the functions

↓

Define the spacex URL and use get() method to fetch the data

↓

Normalize the data using json_normalize() and remove unnecessary features from the dataframe

↓

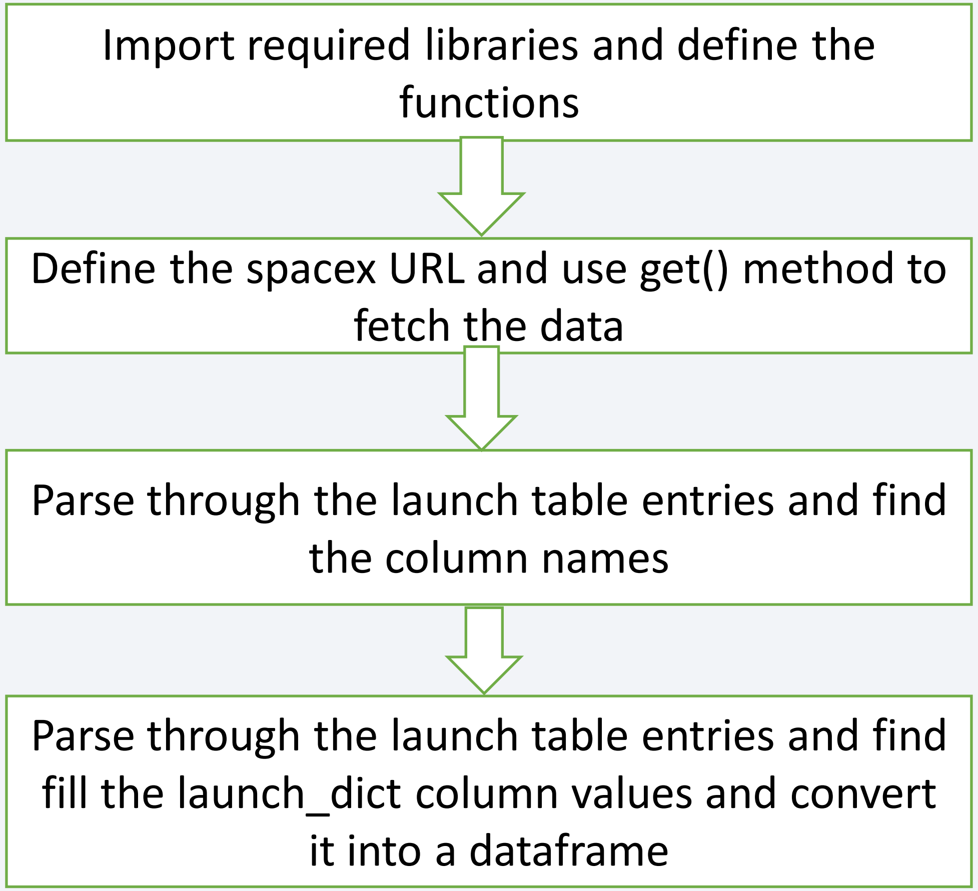
Filter out the Falcon9 data from the dataframe

↓

Find missing/NULL values and replace them with the mean value of the column

Data Collection - Scraping

- Extract a Falcon 9 launch records HTML table from Wikipedia URL "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
- Parse the table and convert it into a Pandas data frame using find_all() function
- GitHub URL: https://github.com/ShilpaKirubanidhi/IBM-SpaceX-Capstone/blob/623dd0c1d3d1e5447696ef4ab2ba53c4265ef391/Data%20collection%20with%20web%20scraping.ipynb



Data Wrangling

- Exploratory data analysis (data wrangling) is done on the Falcon 9 launch records to determine the success rate of the launches.
- We will calculate the field "landing_class" (1=success and 0=failure) by exploring the columns "outcome" and "orbit".
- GitHub URL: https://github.com/ShilpaKirubanidhi/IBM-SpaceX-Capstone/blob/623dd0c1d3d1e5447696ef4ab2ba53c4265ef391/EDA%20lab.ipynb

Import required libraries and define auxiliary functions

↓

Load SpaceX data (from CSV file) into a dataframe

↓

Calculate number of launches in each site and each orbit

↓

Define the list of bad outcome values, and create a landing_class column where 0=failure(if it's in the list of bad outcome) and 1=success

EDA with Data Visualization

- Objective is to perform exploratory Data Analysis and Feature Engineering using Pandas and Matplotlib. The purpose of the charts is to find any data patterns that would help in drawing insights. Charts plotted were:
  - FlightNumber vs. LaunchSite
  - Payload vs. Launch site
  - Success rate vs. Orbit
  - Flightnumber vs. orbit
  - Payload Vs. orbit
  - Yearly success trend
- GitHub URL: https://github.com/ShilpaKirubanidhi/IBM-SpaceX-Capstone/blob/623dd0c1d3d1e5447696ef4ab2ba53c4265ef391/EDA%20with%20visualization.ipynb

EDA with SQL

- SQL queries 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 using 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
- GitHub URL: https://github.com/ShilpaKirubanidhi/IBM-SpaceX-Capstone/blob/623dd0c1d3d1e5447696ef4ab2ba53c4265ef391/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

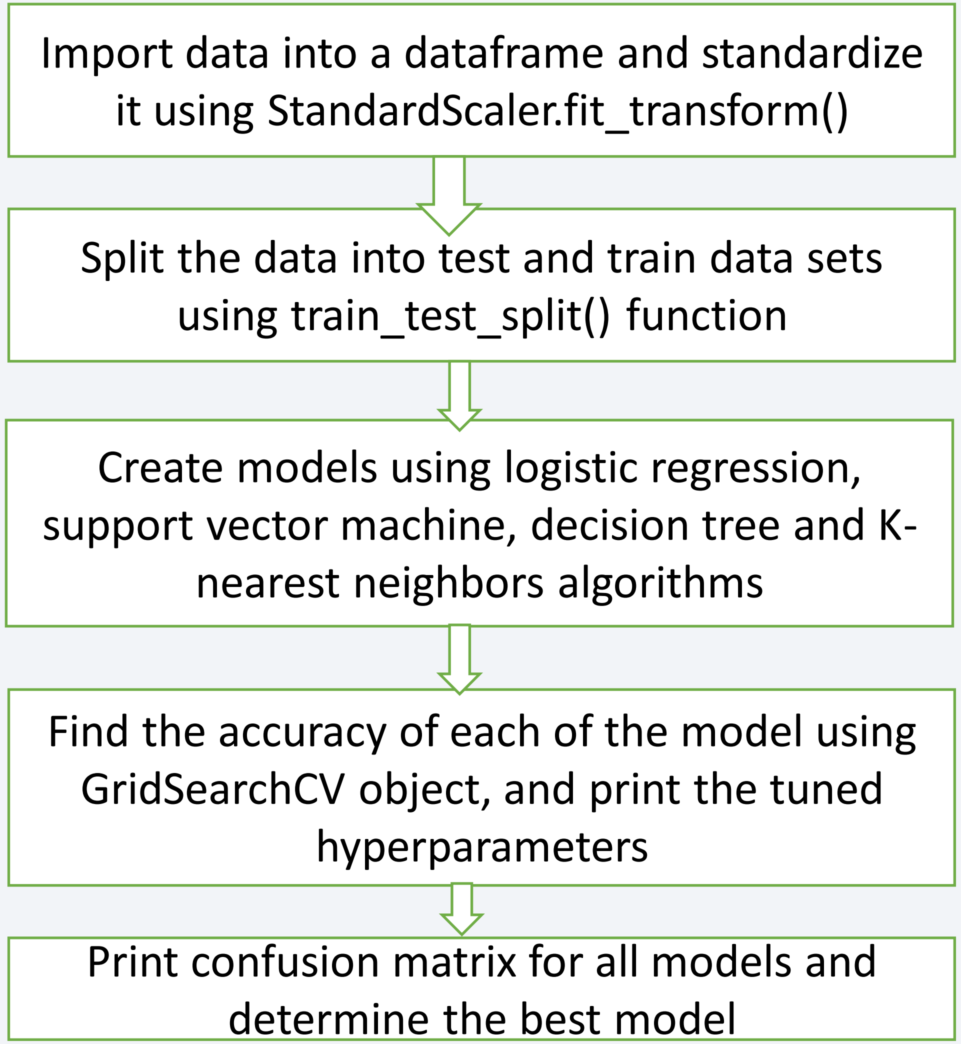
- Launch sites were marked using a Folium.marker object. MarkerCluster() object was used to show the successful(green) and failed(red) icons in each launch site.
- Folium.marker object was used to depict the nearest coastline, railroad, highway and city for each launch site. The function calculate_distance() was defined to find the distance in km between any two points on the map.
- Folium.PolyLine object was used to draw a line between the 2 points on the map to show the proximities of each launch site to the landmark, along with the distance in km.
- GitHub URL: https://github.com/ShilpaKirubanidhi/IBM-SpaceX-Capstone/blob/623dd0c1d3d1e5447696ef4ab2ba53c4265ef391/Interactive%20visual%20analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- The SpaceX Launch records dashboard has the below components:
  - A dropdown menu with the launch site (individual and all sites) to display the success rate
  - A pie chart of the successful and failed launches for the selected option
  - A slider for payload range
  - A scatter plot to display correlation between success rate and payload (from input selected in the initial drop down box)
- The purpose of these charts are to answer the questions:
  - Which site has the largest successful launches?
  - Which site has the highest launch success rate?
  - Which payload range(s) has the highest launch success rate?
  - Which payload range(s) has the lowest launch success rate?
  - Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest
- GitHub URL: https://github.com/ShilpaKirubanidhi/IBM-SpaceX-Capstone/blob/fcb7bb2b6709a12a118fce573aef2eb7fe619f8e/spacex_dash_app.py

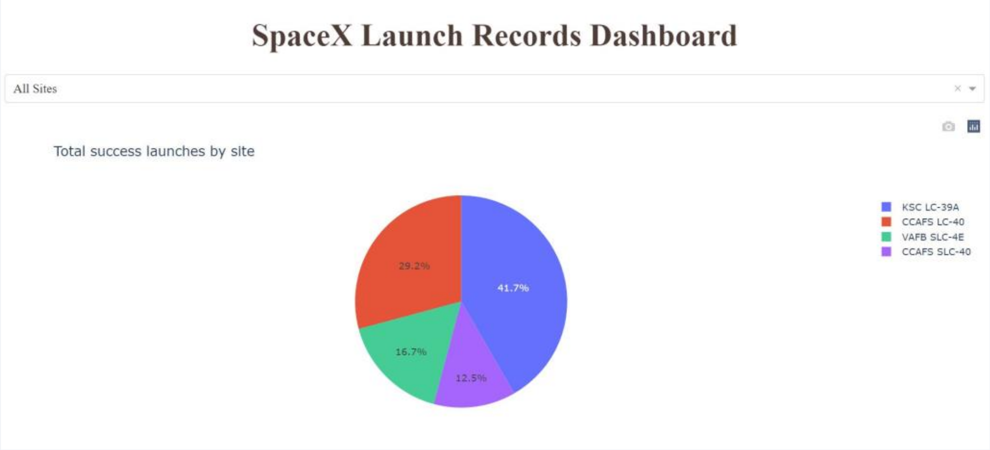
Predictive Analysis (Classification)

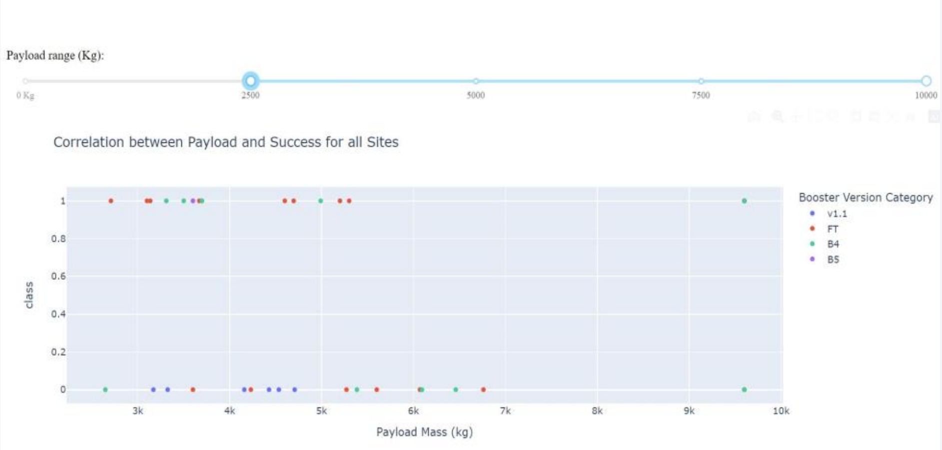
- We will evaluate the accuracy of the models built using logistic regression, support vector machine, decision tree and K-nearest neighbors. GridsearchCV object is used to find the tuned hyperparameters and the accuracy of each model.
- The confusion matrix is built for each model using the auxiliary function plot_confusion_matrix(). The best model is decided based on the accuracy score and confusion matrix.
- GitHub URL: https://github.com/ShilpaKirubanidhi/IBM-SpaceX-Capstone/blob/5cb10952baffe0585835673ab1ed1f2857392615/Machine%20learning%20prediction-new.ipynb



Results

- Exploratory data analysis results:
    - The various charts plotted in the exploratory data analysis help in deciphering the various relationships between factors such as flight number, launch site, orbit type and success rates of the launch. The yearly success rate trend chart shows that the success rate increases year by year, and the best success rate was in 2020.
- Interactive analytics demo screenshots:





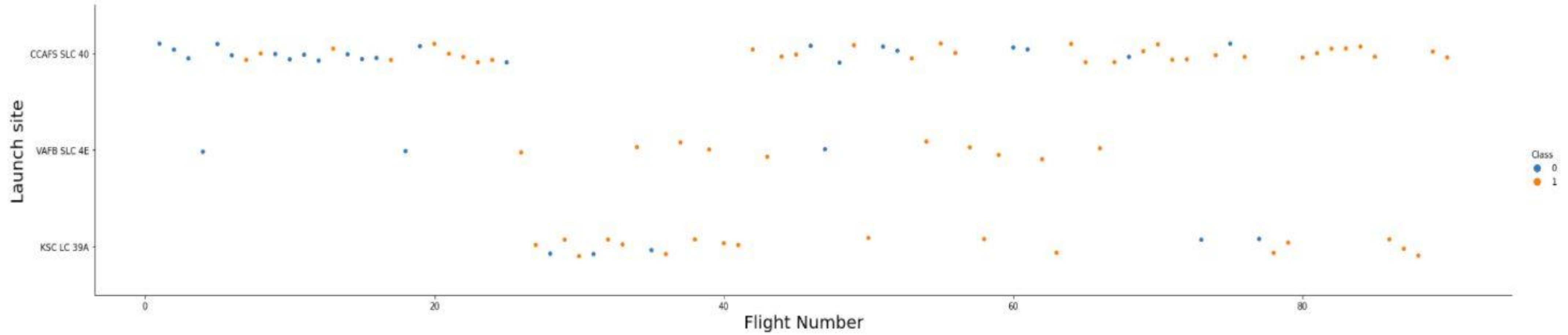
- Predictive analysis results:
    - Based on the models built using logistic regression, decision tree, support vector machine and KNN algorithms, we found the accuracy of each model. The best performing model is decision tree with an accuracy of 89%.

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

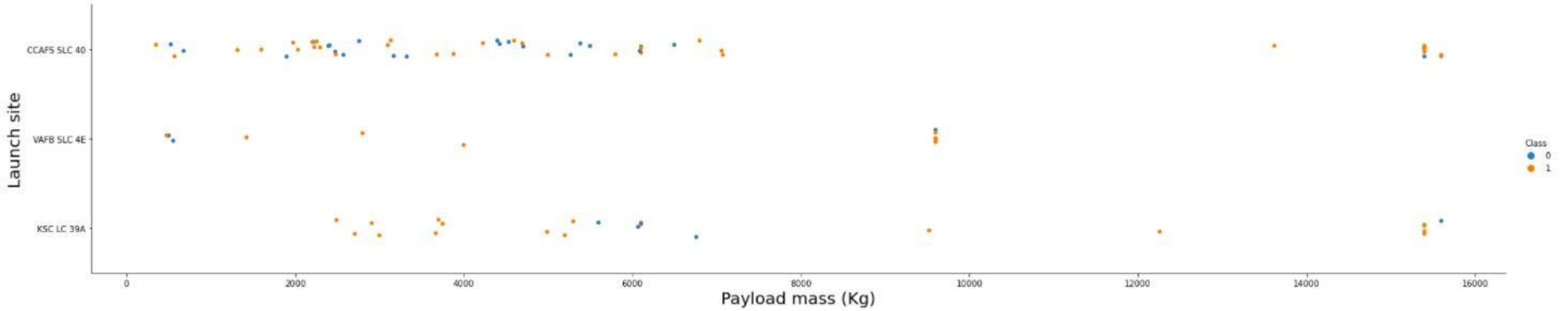
```
In [4]: # Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch site",fontsize=20)
plt.show()
```



- Flight number Vs. launch site graph shows that site VAFB SLC 4E has the best success rate. KSLC 39A has only greater than 20 flight numbers, CCAFS SLC 40 has the most number of flights.

Payload vs. Launch Site

In [5]:
```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 valu
e
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Payload mass (Kg)",fontsize=20)
plt.ylabel("Launch site",fontsize=20)
plt.show()
```



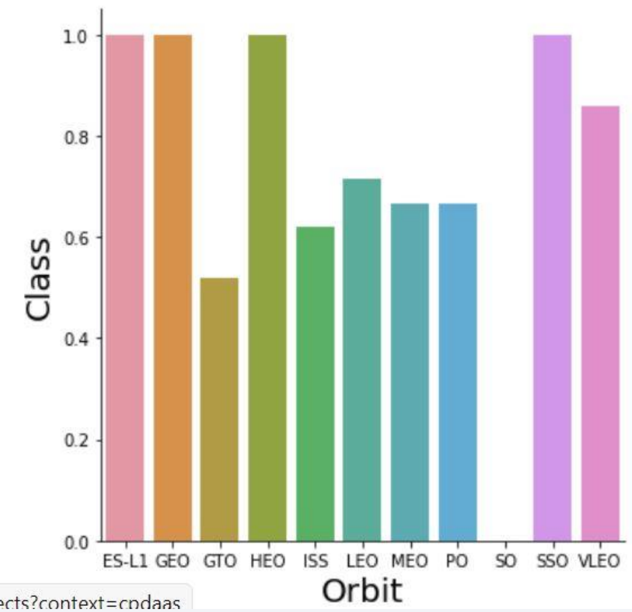
- Payload Vs. Launch site graph shows that VAFB-SLC 4E site has are no rockets launched for heavypayload mass(greater than 10000). Most of the flights for CCAFS SLC 40 has most of its flights with payload less than 8000, and for KSLC 39A, the minimum payload is 2000.

Success Rate vs. Orbit Type

- Sucess rate Vs. Orbit bar chart shows that the orbits with maximum success rates are ES-L1, GEO, HEO, SSO.

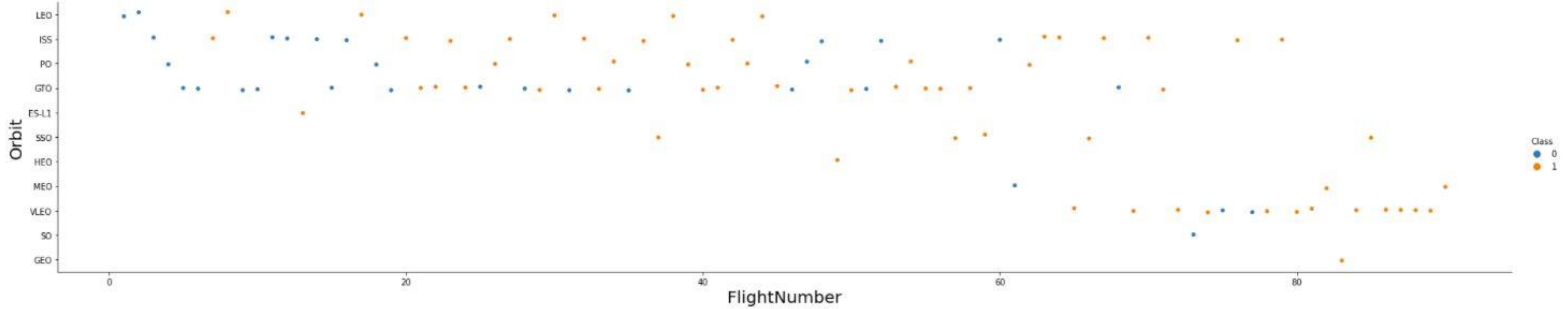
```
In [36]: # HINT use groupby method on Orbit column and get the mean of Class column
df2=df
orbit_1=df2.groupby(['Orbit'])['Class'].mean().reset_index()
orbit_df=pd.DataFrame(orbit_1)
orbit_df

sns.catplot(y="Class", x="Orbit", kind='bar',data=orbit_df)
plt.xlabel("Orbit",fontsize=20)
plt.ylabel("Class",fontsize=20)
plt.show()
```



Flight Number vs. Orbit Type

In [37]:
```
# 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("FlightNumber",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



- Flight number Vs. Orbit type graph shows that only the orbits GTO, PO, ISS and LEO have flight numbers ranging from 0 to 100. The other orbits have a minimum of 40 - 60 flight numbers, and a maximum of 100.

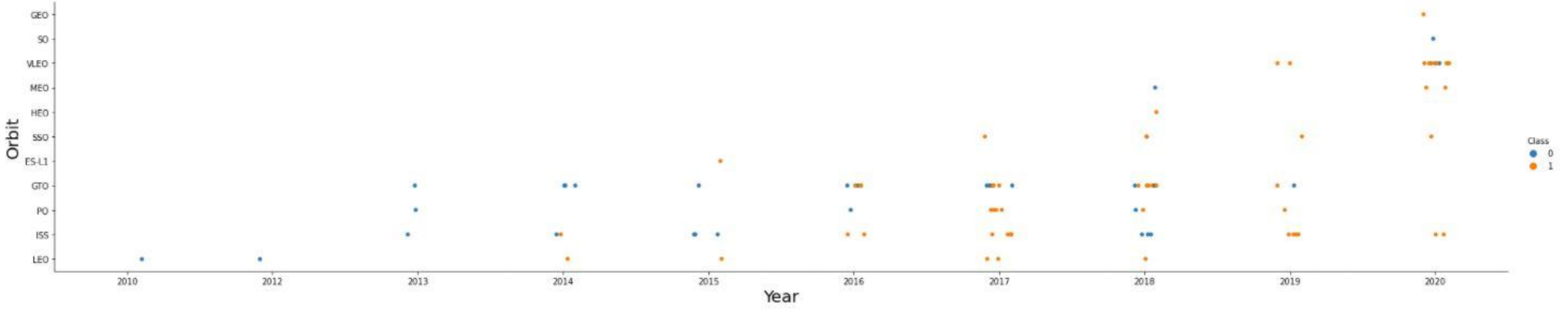
Payload vs. Orbit Type

In [38]:
```
# 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("PayloadMass(kg)",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```

- Payload mass Vs. Orbit type plot depicts that there are only very few launches with payloadmass>8000 kg, and most of them are successful in the orbits VLEO, ISS and PO. All the other orbits have flights with payload mass<8000 kg.

Launch Success Yearly Trend

```
In [40]: # Plot a line chart with x axis to be the extracted year and y axis to be the success rate
         sns.catplot(y="Orbit", x="Year", hue="Class", data=df1 ,aspect = 5)
         plt.xlabel("Year",fontsize=20)
         plt.ylabel("Orbit",fontsize=20)
         plt.show()
```



- Launch success yearly trends graph shows that the number of launches and the success rates are increasing year by year, and the maximum success rate was in 2020.

All Launch Site Names

- Select the distinct launch_site from the spacex table using following query:
- %sql select distinct launch_site from spacextbl

```
In [7]: %sql select distinct launch_site from spacextbl

 * ibm_db_sa://dqq29189:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Out[7]:

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

Launch Site Names Begin with 'CCA'

- Select records from spacex table with condition ‘CCA%’ to filter based on launch_site beginning with CCA and limit to 5 records
- %sql select * from spacextbl where launch_site like 'CCA%' limit 5

Display 5 records where launch sites begin with the string 'CCA'

In [8]:
```
%sql select * from spacextbl where launch_site like 'CCA%' limit 5
```

```
 * ibm_db_sa://dqq29189:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Out[8]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Sum keyword used to get the total payload mass and filter by customer name as NASA CRS.
- %sql select sum(payload_mass__kg_) from spacextbl where customer like 'NASA%CRS%'

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

In [20]:
```
%sql select sum(payload_mass__kg_) from spacextbl where customer like 'NASA%CRS%'
```

```
 * ibm_db_sa://dqq29189:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Out[20]:

1
48213

Average Payload Mass by F9 v1.1

- AVG keyword is used to find the average payload mass from spacex table with filter condition of booster version = “F9 V1.1”

Task 4

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

```
In [22]: %sql select avg(payload_mass__kg_) from spacextbl where booster_version='F9 v1.1'

 * ibm_db_sa://dqq29189:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Out[22]:

1
2928

First Successful Ground Landing Date

- Select the minimum of the date column with a filter of landing outcome = Success groun pad.
- %sql select min(date) from spacextbl where landing__outcome='Success (ground pad)'

List the date when the first successful landing outcome in ground pad was acheived.

Hint:Use min function

```
In [26]: %sql select min(date) from spacextbl where landing__outcome='Success (ground pad)'

 * ibm_db_sa://dqq29189:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Out[26]:

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Select all unique values of bosster version using the distinct keyword and filter the payload mass to be> 4000 and <6000.
- %sql select distinct booster_version from spacextbl where landing__outcome ='Success (drone ship)' and payload_mass__kg_>4000 and payload_mass__kg_<6000

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

```
In [28]: %sql select distinct booster_version from spacextbl where landing__outcome ='Success (drone ship)' and payload_mass__kg_>4000 an
d payload_mass__kg_<6000

 * ibm_db_sa://dqq29189:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.

Out[28]:
```

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

Total Number of Successful and Failure Mission Outcomes

- Count(*) is used along with group_by clause to get the number of success and failure outcomes.
- %sql select distinct mission_outcome,count(*) from spacextbl group by mission_outcome

List the total number of successful and failure mission outcomes

In [30]:
```
%sql select distinct mission_outcome,count(*) from spacextbl group by mission_outcome
```

```
 * ibm_db_sa://dqq29189:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Out[30]:

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

Boosters Carried Maximum Payload

- Used a subquery to find the maximum payload mass and used that in the main query to find the booster version carrying that payload mass
- %sql select booster_version from spacextbl where payload_mass__kg_ in(select max(payload_mass__kg_) from spacextbl)

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
In [34]: %sql  select booster_version from spacextbl where payload_mass__kg_ in(select max(payload_mass__kg_) from spacextbl)

 * ibm_db_sa://dqq29189:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.

Out[34]:
```

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

2015 Launch Records

- Fetch the columns landing outcome, booster version and launch site from spacex table with filter for failed outcomes and date is in 2015.
- %sql select landing__outcome,booster_version,launch_site from spacextbl where landing__outcome='Failure (drone ship)'and date>='01-01-2015'and date<='12-31-2015';

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [35]:
```
%sql select landing__outcome,booster_version,launch_site from spacextbl where landing__outcome='Failure (drone ship)'and date>
='01-01-2015'and date<='12-31-2015';
```

```
 * ibm_db_sa://dqq29189:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Out[35]:

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

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

- Fetch the landing outcome and count of records grouped by landing outcome, and order it in descending order to obtain the rank. Filter recordsby date >='06-04-2010’and <='03-20-2017'
- %sql select landing__outcome,count(*) as counts from spacextbl where date>='06-04-2010'and date<='03-20-2017' group by landing__outcome order by counts desc

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

In [36]:
```
%sql select landing__outcome,count(*) as counts from spacextbl where date>='06-04-2010'and date<='03-20-2017' group by landing__
outcome order by counts desc
```

```
 * ibm_db_sa://dqq29189:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/bludb
Done.
```

Out[36]:

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

Section 4

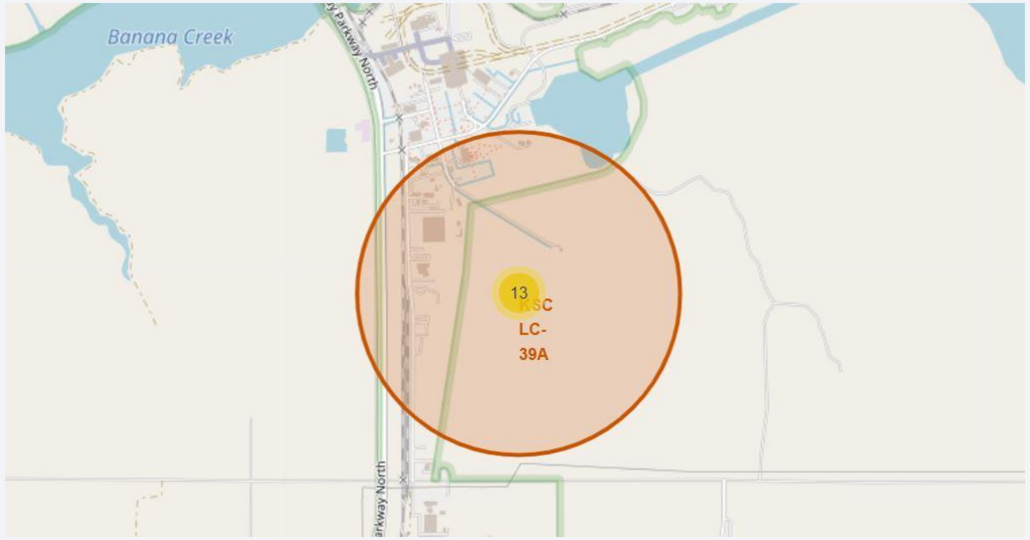
Launch Sites Proximities Analysis

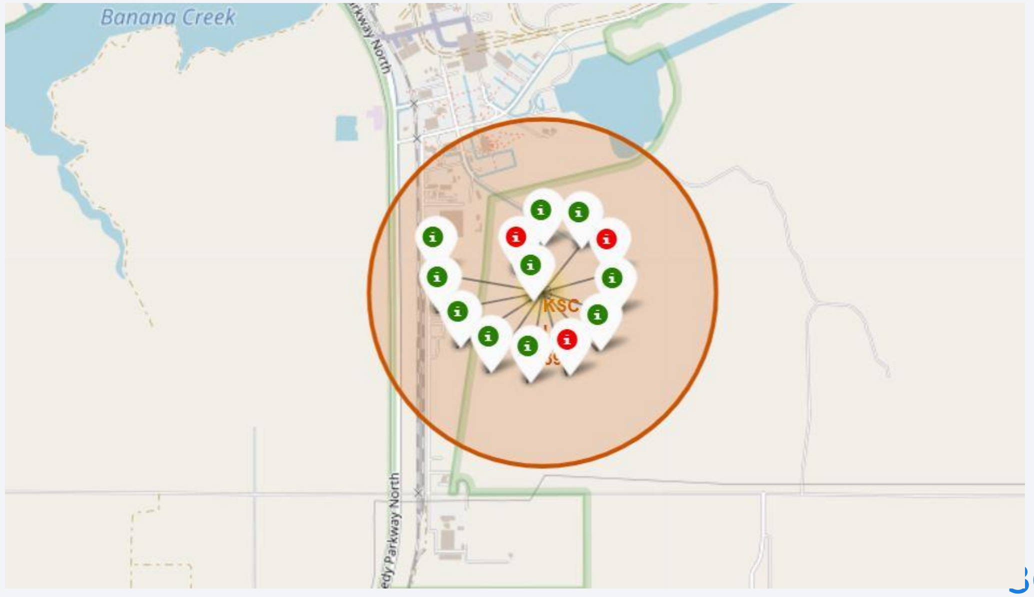
Folium map for all launch sites

- The folium map to show all launch sites is built using folium.map() function. Using the latitude and longitude for each site, we add the folium.circle() and folium.map.marker() objects to show the site locations. Add_child() function is used to add the location sites to the map.
- The map shows that 3 of the launch sites CCAFS LC-40, CCAFS SLC-40, KSC LC-39A are in the eastern side of the map, nearer to each other. The site VAFB SLC-4E is in the western side of the USA.

Folium map showing color-coded outcomes for each site

- This folium map shows the outcomes for each launch site (green=success and red=failed outcomes) when we click on the site location. Assign_marker_color() function is defined to assign the color codes. MarkerCluster() object is used to add the outcomes for each site on the map.
- The first screenshot of launch site KSC LC-39A shows the total number of outcomes (13). When we click on the site icon, it displays the landing outcomes that are color coded (red-failed, green=success) as shown on screenshot 2.





Folium map for launch site proximities

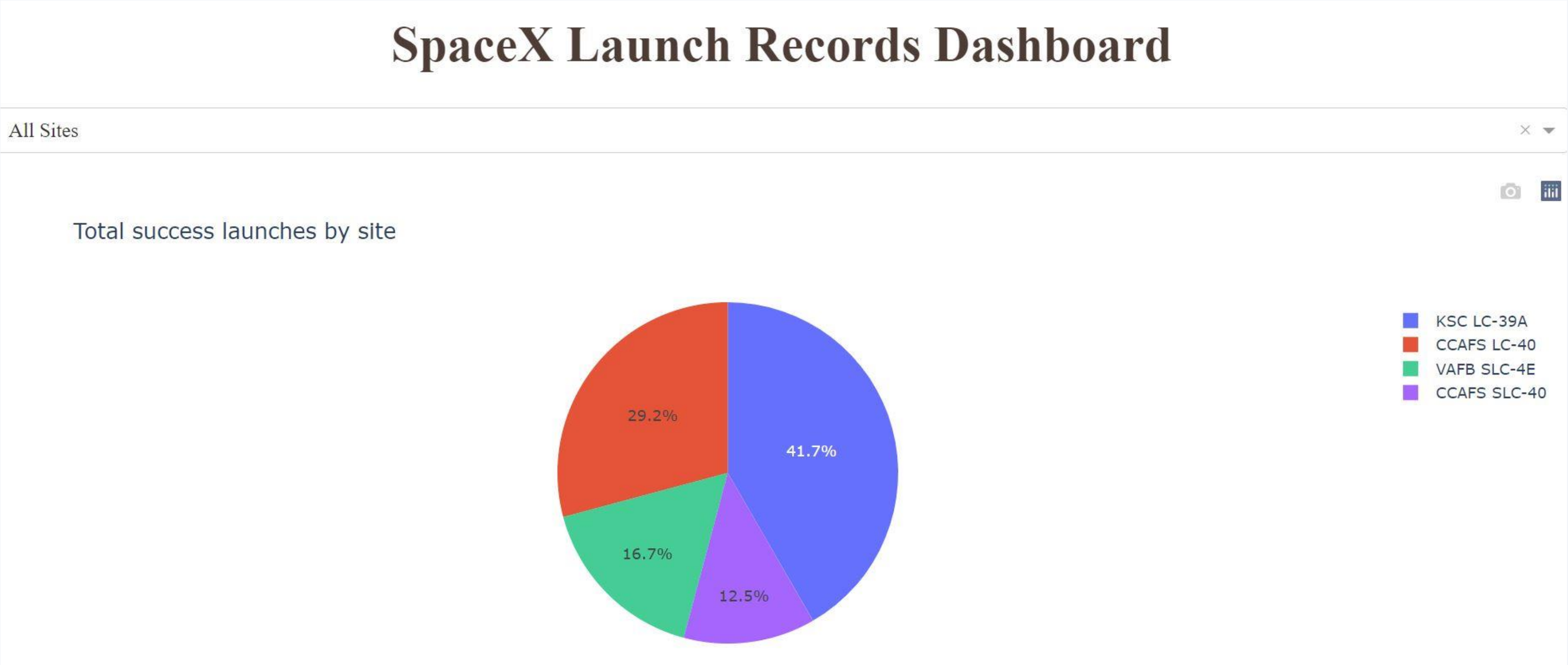
- The distance from each launch site to the nearest coastline, highway, railroad and city are calculated using the function calc_distance(). The latitude and longitude of each location is passed as input. Folium.marker object is used to mark the sites and add_child() is used to add them to the map. Folium.polyline() is used to draw a line between each site and its proximities, depicting the distance in kms. This screenshot shows the proximities for the site VAFB SLC-4E.
- Findings from the map:
  - The launch sites are in close proximity to coastline.
  - The launch sites are pretty close to railroads and highways.
  - The launch sites maintain a certain distance from the nearest city, minimum being 15 kms.

Section 5

Build a Dashboard with Plotly Dash

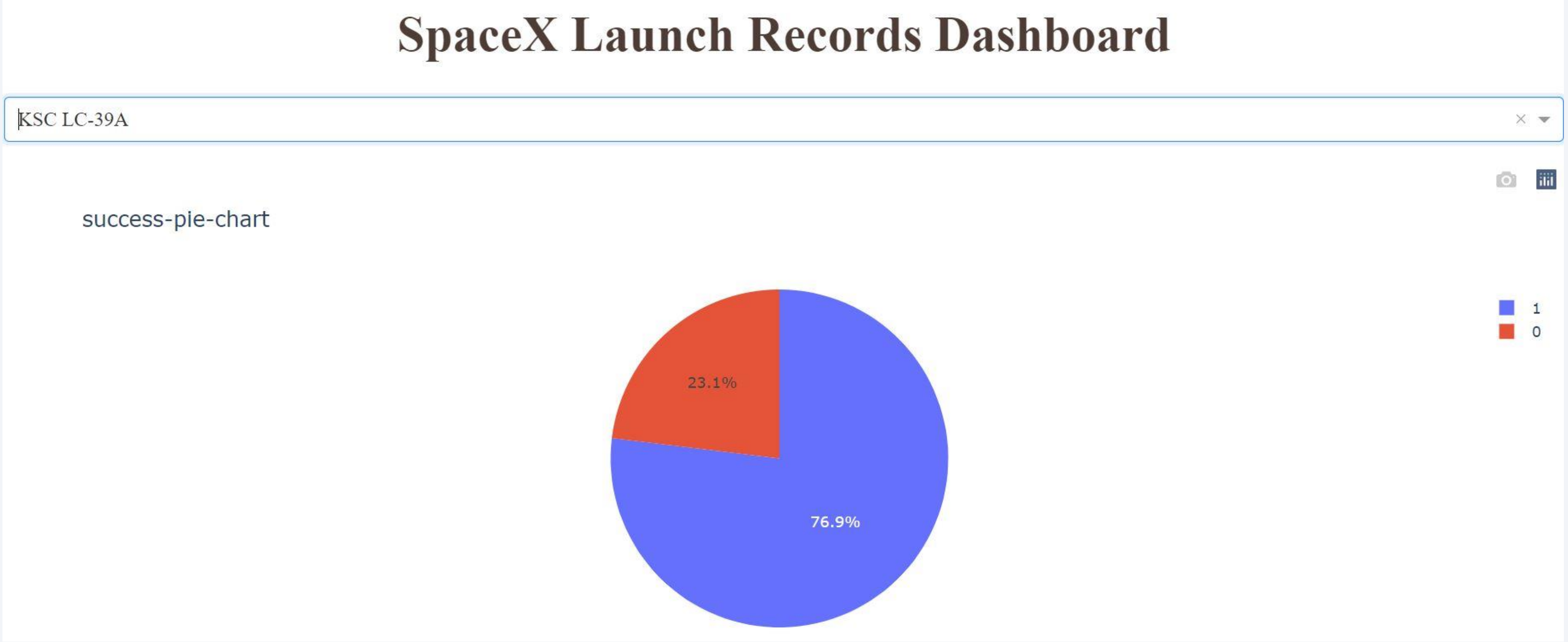
Dashboard – pie chart for launches

- Input is given in the form of a drop down box, containing options for all sites and each site individually. For each site selected, a pie chart is displayed showing the success and failure launch percentages.
- The graph for “All sites” option shows that the highest success rate is for the site KSC LC-39A and the least is for the site CCAFS SLC-40.



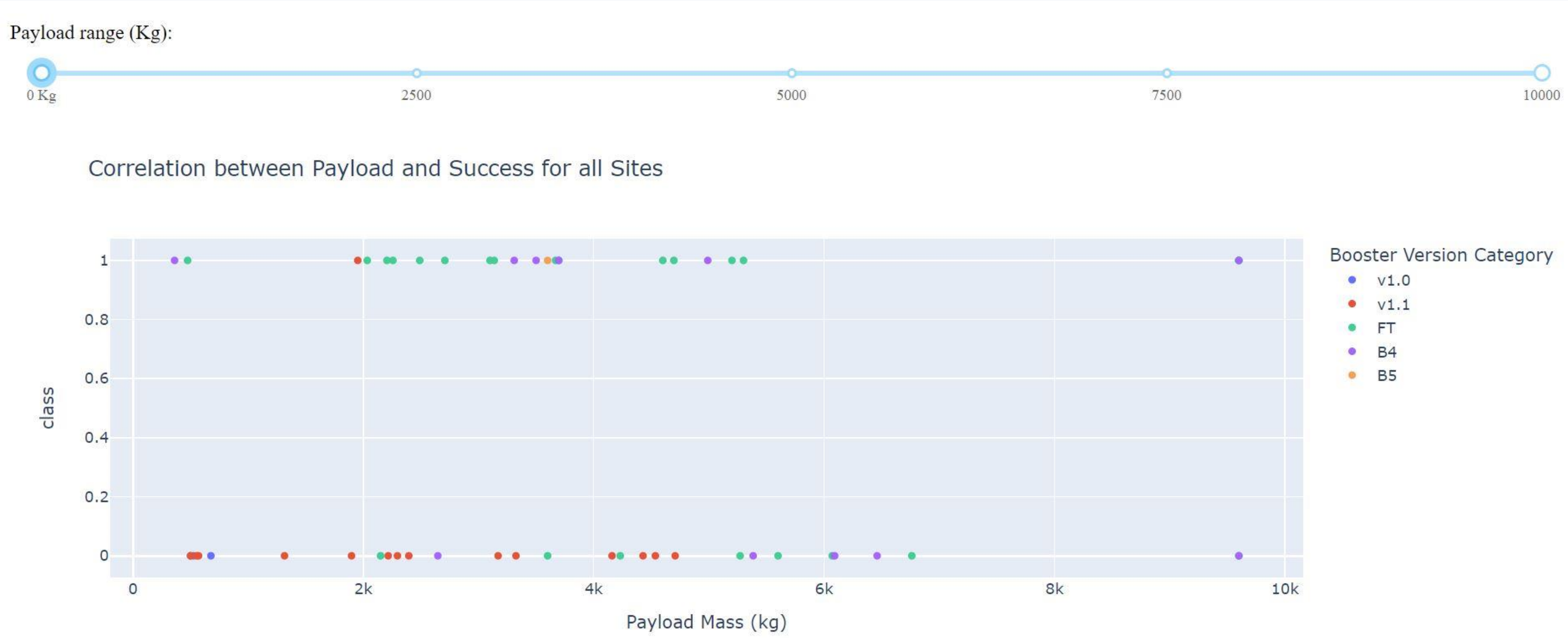
Dashboard showing success rate for each site

- The launch site with highest launch success ratio is KSC LC-39A, displayed in screenshot. The graph for other sites are provided in Appendix A.
- The success percentage is shown in purple color in the pie chart and failure percentage in red. The legend on the side displays the “class” field (0=failure and 1=success).



Dashboard with slider and scatter plot

- Based on the input provided in the drop down box and the payload mass selected in the slider, the scatter plot is displayed. It shows the correlation between payload mass and success rate for all sites, along with the booster version details.
- When the slider is kept at 0 to 10000 , it displays all of the launches. We can see that booster version FT has most successes and V1.1 has the most failures. For payload mass >2500 booster version FT has 7 success and 6 failed launches (highest in both success and failures). There are very few launches in payload mass >5000 and the most successful is FT. There are only 2 launches in payload mass >7500, 1 of which is successful and 1 failed, both being booster version B4. The graphs for other payload mass ranges are in the Appendix B.
- The maximum success rate is for payload mass<5000. The maximum payload mass ever launched was 9600 kg.

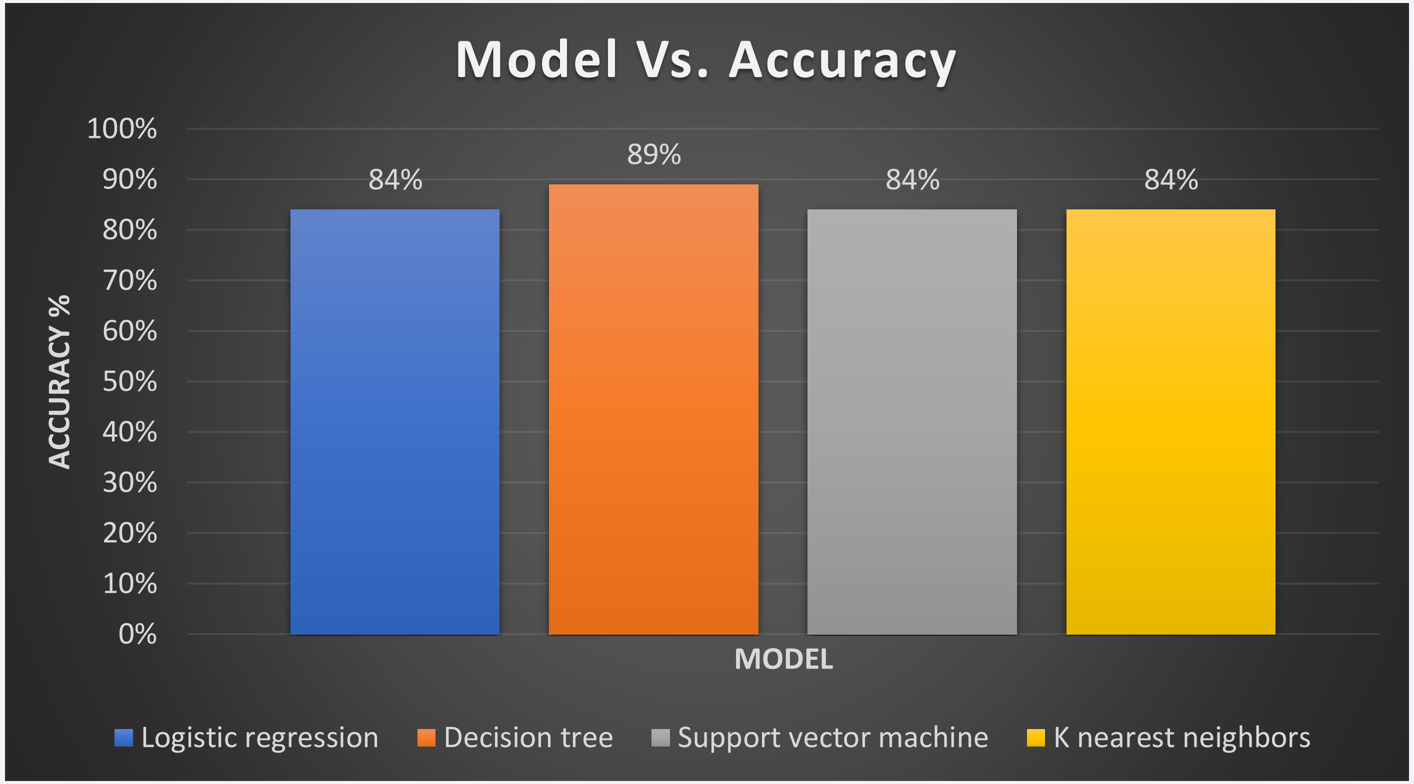


Section 6

Predictive Analysis (Classification)

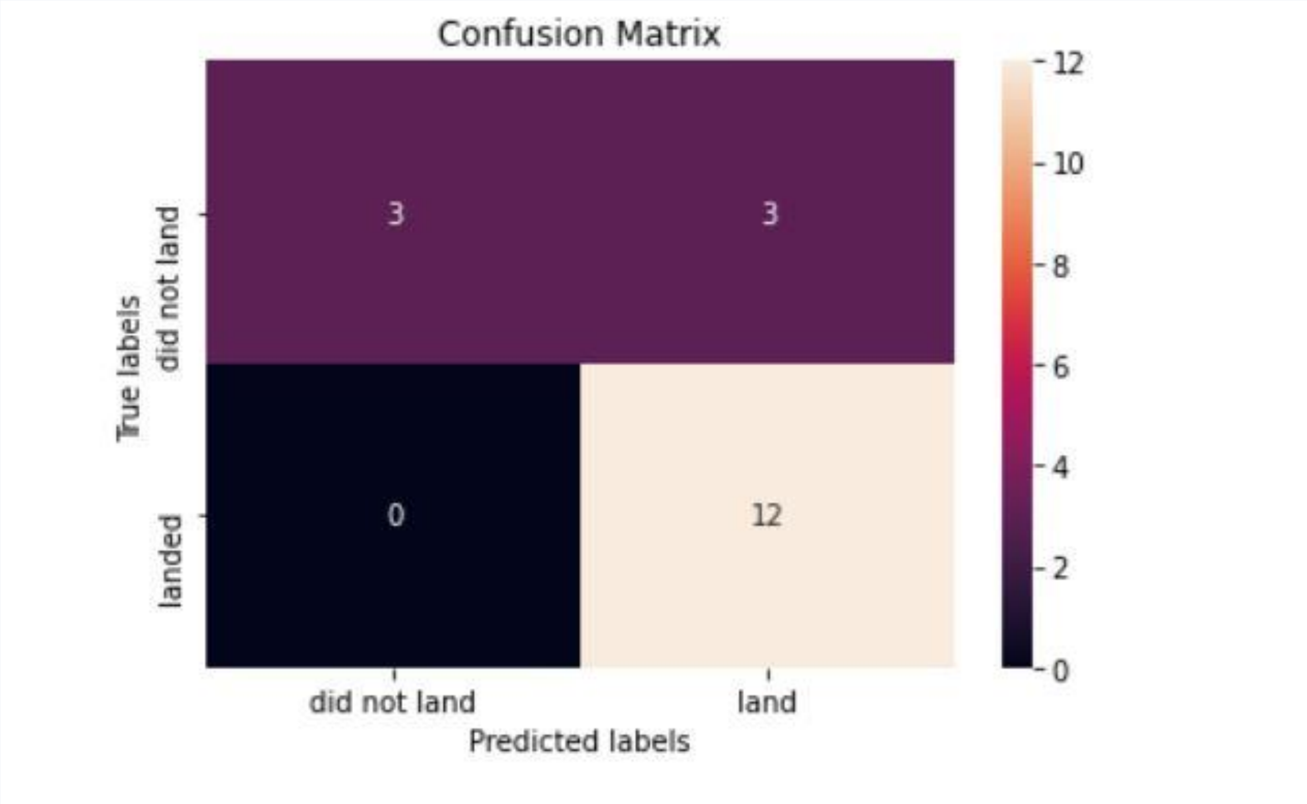
Classification Accuracy

- This bar chart shows the accuracy of each model built.
- The model having the highest classification accuracy is decision tree, with 89%. The other 3 models have the same accuracy of 84%.



Confusion Matrix

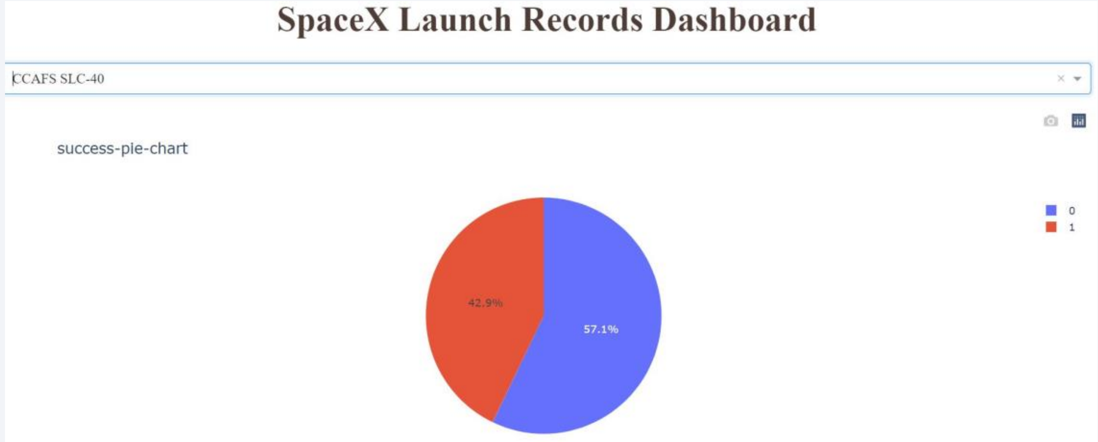
- This is the confusion matrix of decision tree based model.
- It has 3 false positives(predicted=landed and actual=did not land) and 0 false negative (predicted=did not land and actual=landed).
- It has 3 true negatives (actual=did not land and predicted=did not land) and 12 true positives (actual=landed and predicted=landed).

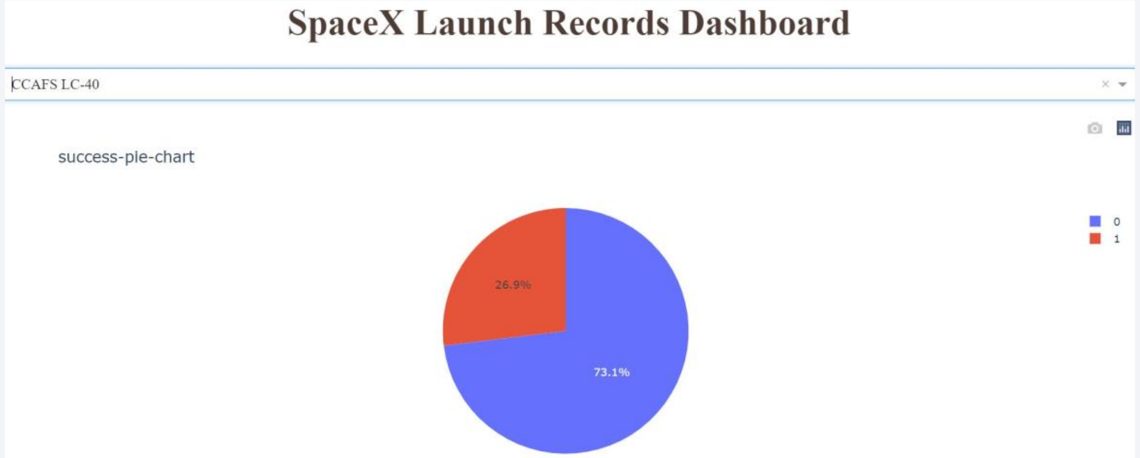


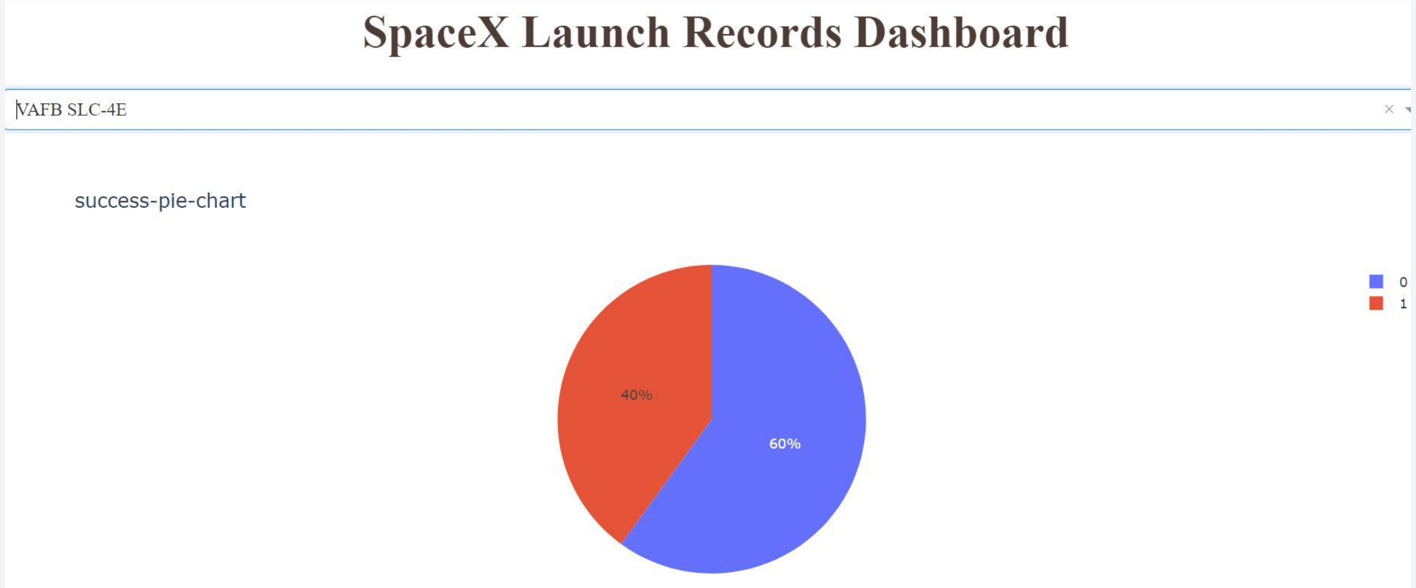
Conclusions

- The best performing model is decision tree with an accuracy of 89%. The cost of each launch and the landing status can be predicted accurately using this model.
- The launch sites analysis show that the sites are in close proximity to coast lines and pretty far off from cities.
- The dashboard visualization of launch data gives the following conclusions:
  - most successful year was 2020, and also the year with most launches.
  - Most successful site was KSC LC-39A, and most successful booster version was FT.
  - The least successful site was CCAFS SLC-40 and most unsuccessful booster version was V1.1.
  - The best payload mass range for success was between 2000kgs and 5500kgs.

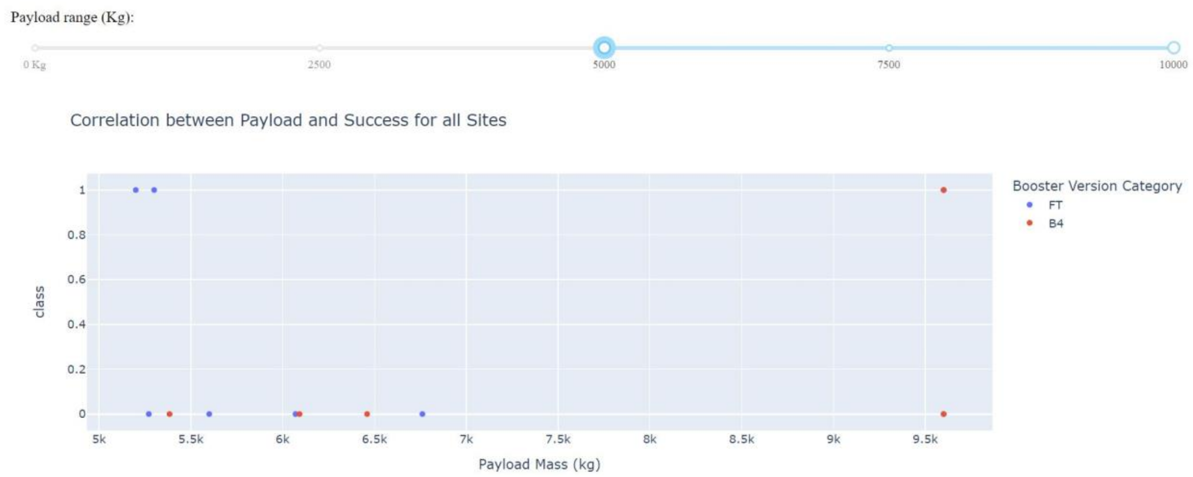
Appendix A – success rate pie chart for each launch site







Appendix B – slider input and scatter plot for different payload mass values



Thank you!