IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Vijay Jawali

Outline

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

Executive Summary

- At the beginning, data is collected using web scraping and relevant data is extracted as much as possible
- Data is collected from various sources. After your raw data has been collected, it will required to improve the quality by performing data wrangling.
- Then we explore the processed data using SQL to gather insights
- To gain further insights into the data, we apply some basic statistical analysis and data visualization, and will able to see directly how variables might be related to each other.
- Further, data is drilled down into finer levels of detail by splitting the data into groups defined by categorical variables or factors.
- Next stage is to build, evaluate, and refine predictive models for discovering more exciting insights.

Introduction

In this capstone, we will predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward 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. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
    - We start by requesting rocket launch data from SpaceX API with the following URL `https://api.spacexdata.com/v4/launches/past`
    - We request and parse data using get requests and convert it to JSON, which is further normalized into a data frame that is readable.
- Perform data wrangling
    - In this section, we perform some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.

Methodology

Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

  At this stage we perform exploratory Data Analysis and determine Training Labels
- create a column for the class
- Standardize the data
- Split into training data and test data
- Find best Hyperparameter for SVM, Classification Trees and Logistic Regression
- Find the method performs best using test data

Data Collection

The Following insights are derived from raw data at Data Collection Stage

- From the rocket we would like to learn the booster name
- From the payload we would like to learn the mass of the payload and the orbit that it is going to
- From the launchpad we would like to know the name of the launch site being used, the longitude, and the latitude.
- From cores we would like to learn the outcome of the landing, the type of the landing, number of flights with that core, whether gridfins were used, whether the core is reused, whether legs were used, the landing pad used, the block of the core which is a number used to seperate version of cores, the number of times this specific core has been reused, and the serial of the core.

Data Collection – SpaceX API

- https://github.com/vijayjawali/IBM-Data-Science-Professional-Capstone/blob/main/SpaceX%20Data%20Collection.ipynb

request rocket launch data from SpaceX API

Define and apply the functions on raw data to get the filtered data getBoosterVersion, getLaunchSite, getPayloadData, getCoreData etc

Decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()` add text

construct our dataset using the data we have obtained

Filter the dataframe to only include Falcon 9 launche

Dealing with Missing Values =>
Calculate the mean for the PayloadMass using the .mean(). Then use the mean and the .replace() function to replace np.nan values in the data with the meancalculated

Data Collection - Wrangling

- https://github.com/vijayjawali/IBM-Data-Science-Professional-Capstone/blob/main/Data%20Wrangling.ipynb

Load Space X dataset, from last section.

Identify and calculate the percentage of the missing values in each attribute

Identify which columns are numerical and categorical

Calculate the number of launches on each site

Calculate the number and occurrence of each orbit

Calculate the number and occurence of mission outcome per orbit type

Create a landing outcome label from Outcome column

EDA with Data Visualization

The Following Visualizations are created to understand the correlation between various parameters after the data was cleaned up

- First we look at how the Flight Number and Payload variables would affect the launch outcome.
- Visualize the relationship between Flight Number and Launch Site
- Visualize the relationship between Payload and Launch Site
- Visualize the relationship between success rate of each orbit type
- Visualize the relationship between FlightNumber and Orbit type
- Visualize the relationship between Payload and Orbit type
- https://github.com/vijayjawali/IBM-Data-Science-Professional-Capstone/blob/main/EDA%20with%20Visualization%20lab.ipynb

EDA with SQL

The Following Queries were executed to derive insights on DataSets by Connecting to DB2 on IBM service and querying it using SQL

- Display the names of the unique launch sites in the space mission
- SELECT DISTINCT(launch_site) FROM SPACEXTBL;
- Display 5 records where launch sites begin with the string 'CCA'
- SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
- Display the total payload mass carried by boosters launched by NASA (CRS)
- SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER LIKE 'NASA%';
- Display average payload mass carried by booster version F9 v1.1
- SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION LIKE 'F9 v1.1%';
- List the date when the first successful landing outcome in ground pad was acheived.
  SELECT MIN(DATE) FROM SPACEXTBL WHERE landing__outcome = 'Success (ground pad)'
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  SELECT distinct(booster_version) from spacextbl where landing__outcome = 'Success (drone ship)' AND payload_mass__kg_ > 4000 AND payload_mass__kg_ < 6000;

EDA with SQL

- List the total number of successful and failure mission outcomes
  SELECT (SELECT COUNT(*) FROM SPACEXTBL WHERE LANDING__OUTCOME LIKE 'Success%') as Success, (SELECT COUNT(*) FROM SPACEXTBL WHERE LANDING__OUTCOME LIKE 'Failure%') as Failure FROM SPACEXTBL LIMIT 1;
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- SELECT distinct BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (select max(payload_mass__kg_) from spacextbl)
- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- select landing__outcome, booster_version, launch_site from (select * from spacextbl where year(date) = '2015') where landing__outcome = 'Failure (drone ship)';
- 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
- select count (*) as total, landing__outcome from spacextbl where date >= '2010-06-04' AND date <= '2017-03-20' group by landing__outcome order by total desc
- https://github.com/vijayjawali/IBM-Data-Science-Professional-Capstone/blob/main/EDA%20with%20SQL%20lab.ipynb

Build an Interactive Map with Folium

The Following Tasks were completed as part of this exercise

- Mark all launch sites on a map
- Mark the success/failed launches for each site on the map
- Calculate the distances between a launch site to its proximities

First we get the Launch site co-ordinates for each site and create a Folium map object, the launch co-ordinates are added to the folium map

Next, we mark the success/failed launches for each site on the map, Marker clusters are used to simplify map containing many markers having the same coordinate

Finally, we Calculate the distances between a launch site to its proximities such as Coastal Distance, railway, highway and nearest city

- https://github.com/vijayjawali/IBM-Data-Science-Professional-Capstone/blob/main/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

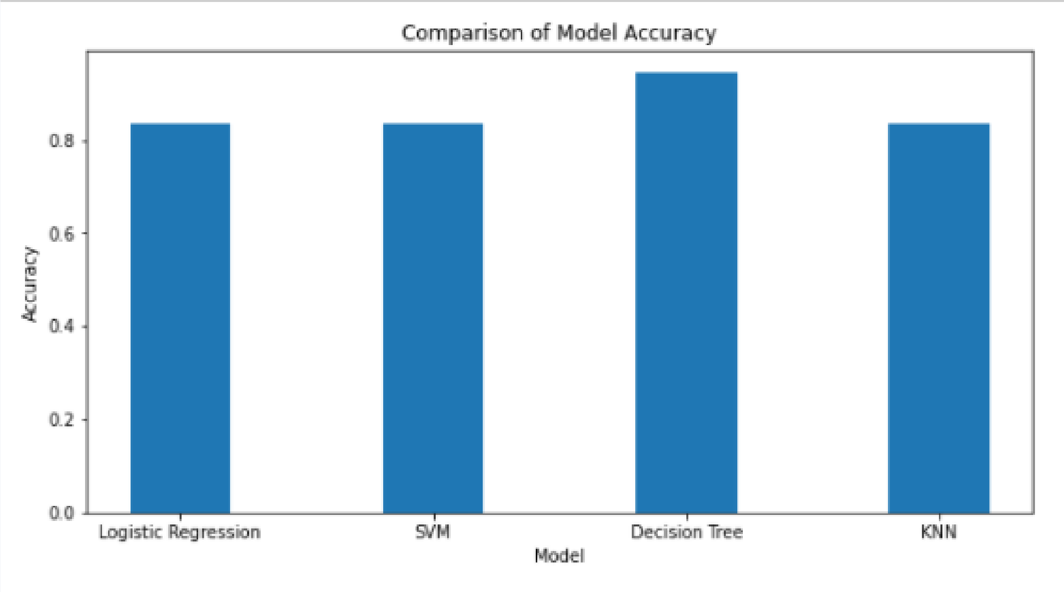
Predictive Analysis (Classification)

Results

- Exploratory data analysis results

The preceding slides illustrate the co-relation between different variables and show the relationship between them to explore furthur the main objective to determing the cost of Launch with predictive analysis

- Interactive analytics demo in screenshots
- 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
- Predictive analysis results



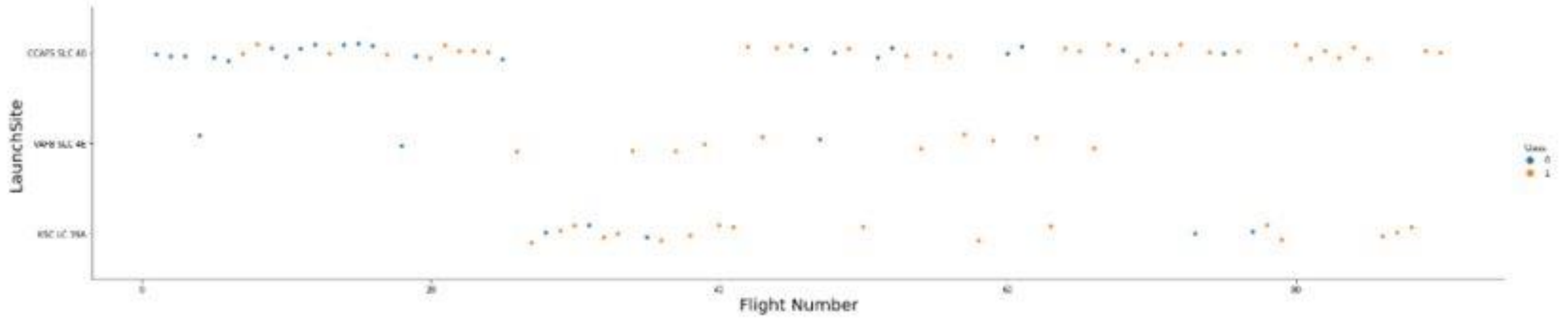
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site

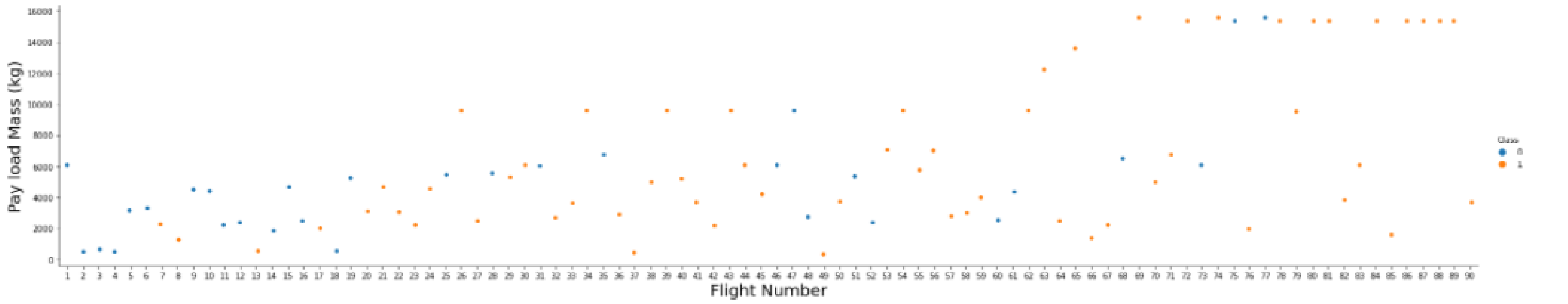
```python
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()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



Flight Number vs. PayLoad Mass

- Show a scatter plot of Flight Number vs. Launch Site

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



Payload vs. Launch Site

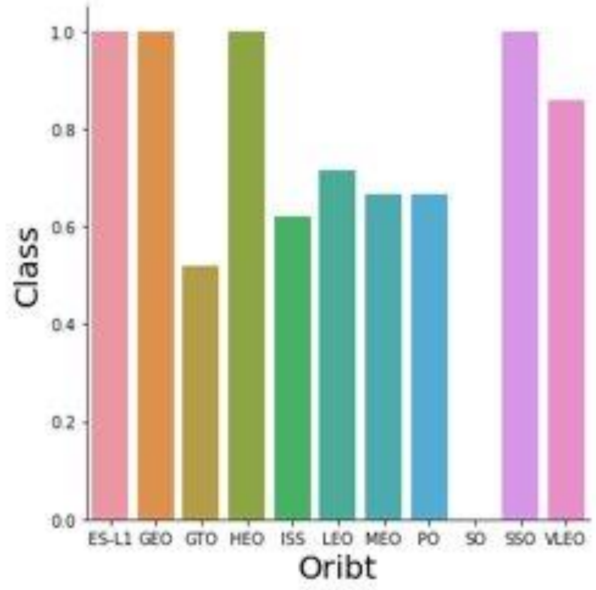
- Show a scatter plot of Payload vs. Launch Site

```
# 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="PayloadMass", x="LaunchSite", hue="Class", data=df, aspect = 5)
plt.xlabel("Launch Site",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```

Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type

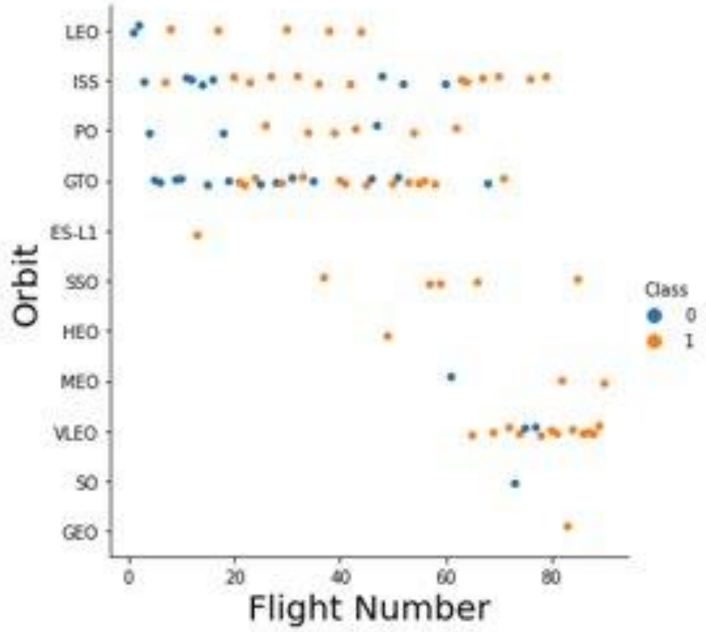
```
# HINT use groupby method on Orbit column and get the mean of Class column
sns.catplot(x=df.groupby('Orbit')['Class'].mean().to_frame().index,y='Class',kind='bar',data=new_df)
plt.xlabel("Oribt",fontsize=20)
plt.ylabel("Class",fontsize=20)
plt.show()
```



Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type

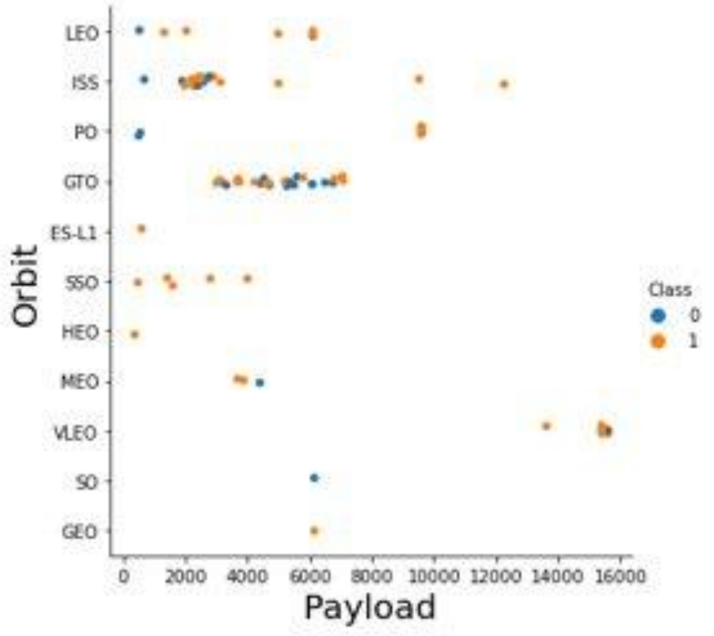
```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(x='FlightNumber', y='Orbit',hue='Class',data=df)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type

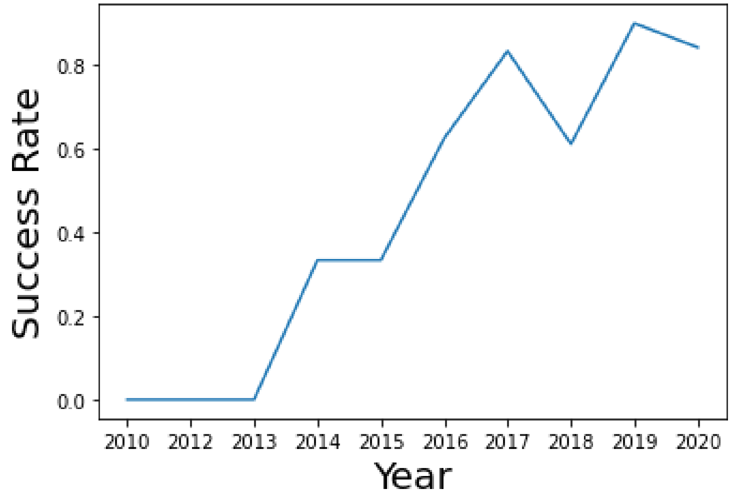
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(x='PayloadMass', y='Orbit',hue='Class',data=df)
plt.xlabel("Payload",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Launch Success Yearly Trend

- Show a line chart of yearly average success rate

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
sns.lineplot(data=new_df, x=new_df.index, y='Class')
plt.xlabel("Year",fontsize=20)
plt.ylabel("Success Rate",fontsize=20)
plt.show()
```



All Launch Site Names

- Find the names of the unique launch sites

Display the names of the unique launch sites in the space mission

```
%%sql
SELECT DISTINCT(launch_site) FROM SPACEXTBL
```

```
 * ibm_db_sa://nlw98739:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

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

Launch Site Names Begin with 'CCA'

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

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

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

```
 * ibm_db_sa://nlw98739:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

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

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

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

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER LIKE 'NASA%';
```

```
 * ibm_db_sa://nlw98739:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

1
99980

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

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

```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION LIKE 'F9 v1.1%';
```

```
 * ibm_db_sa://nlw98739:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

1
2534

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

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

Hint:Use min function

```
%%sql
SELECT MIN(DATE) FROM SPACEXTBL WHERE landing__outcome = 'Success (ground pad)'
```

```
 * ibm_db_sa://nlw98739:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 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 distinct(booster_version) from spacextbl where landing__outcome = 'Success (drone ship)' AND payload_mass__kg_ > 4000 AND payl
oad_mass__kg_ < 6000;
```

```
 * ibm_db_sa://nlw98739:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

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

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

List the total number of successful and failure mission outcomes

```
%%sql
SELECT (SELECT COUNT(*) FROM SPACEXTBL WHERE LANDING__OUTCOME LIKE 'Success%') as Success, (SELECT COUNT(*) FROM SPACEXTBL WHERE LAND
ING__OUTCOME LIKE 'Failure%') as Failure FROM SPACEXTBL LIMIT 1;
```

```
 * ibm_db_sa://nlw98739:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

success	failure
61	10

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

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

```
%%sql
SELECT distinct BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (select max(payload_mass__kg_) from spacextbl)
```

```
 * ibm_db_sa://nlw98739:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

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

2015 Launch Records

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

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

```
%%sql
select landing__outcome, booster_version, launch_site from (select * from spacextbl where year(date) = '2015') where landing__outcome
= 'Failure (drone ship)';
```

```
 * ibm_db_sa://nlw98739:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

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

- 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

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

```
%%sql
select count (*) as total, landing__outcome from spacextbl where date >= '2010-06-04' AND date <= '2017-03-20' group by landing__outcome order by total desc
```

```
 * ibm_db_sa://nlw98739:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

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

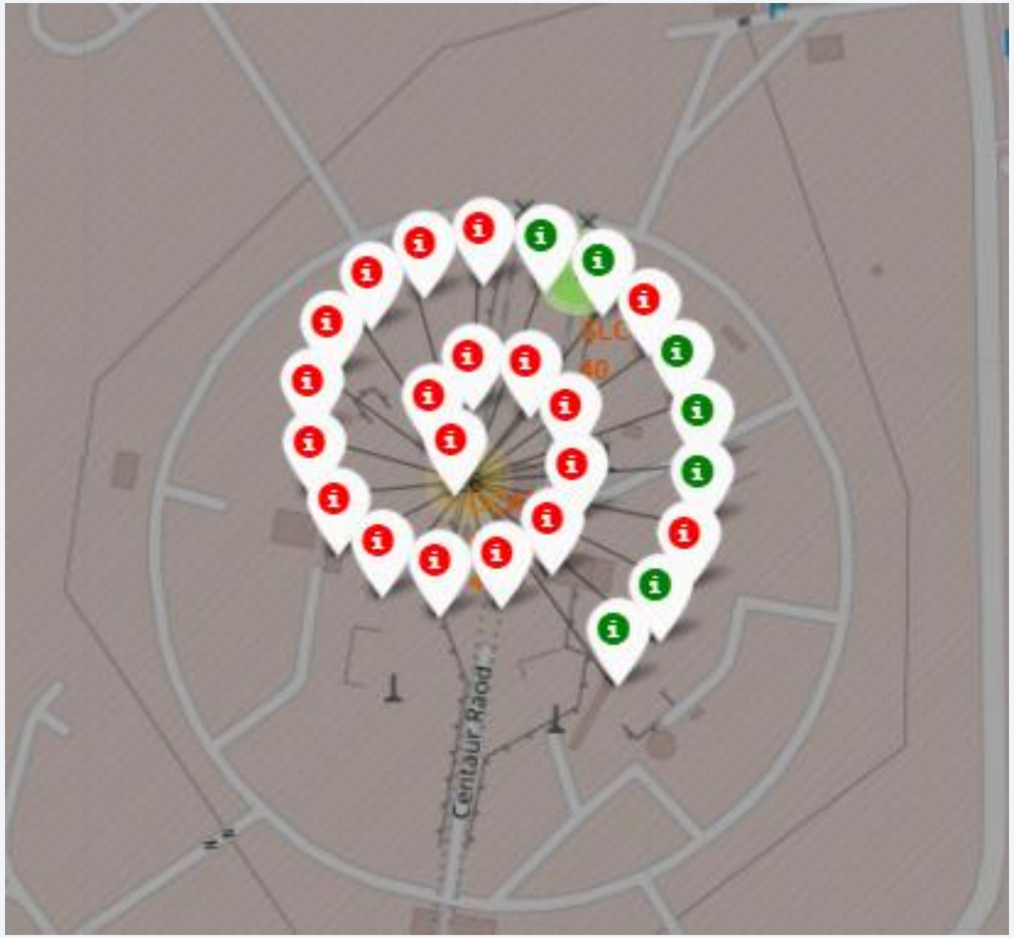
Section 4

Launch Sites Proximities Analysis

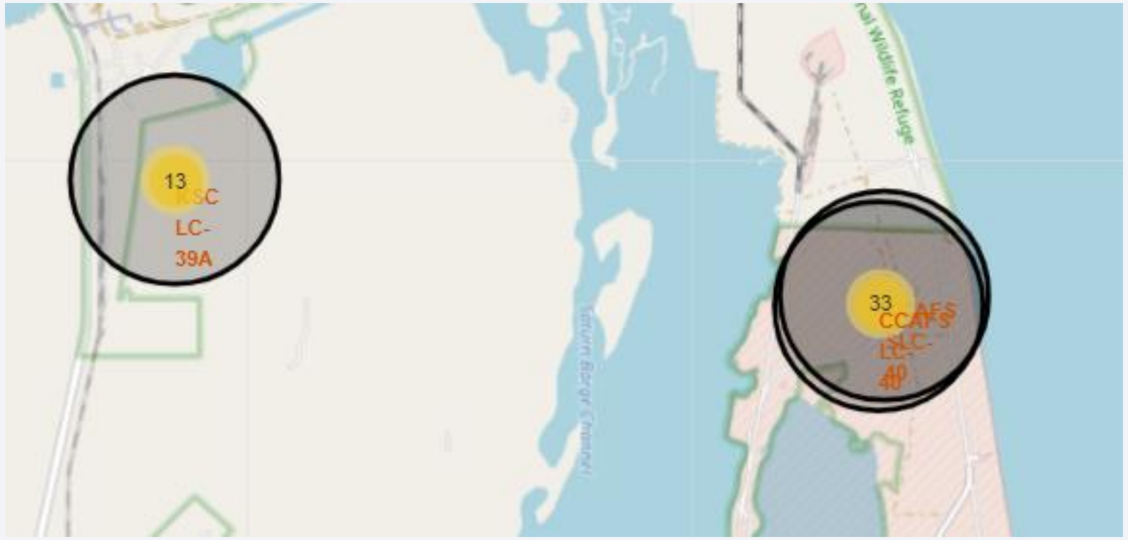
Launch site Map

- the generated folium map below includes all launch sites' location markers on a global map

Success/failed launches for each site on the map

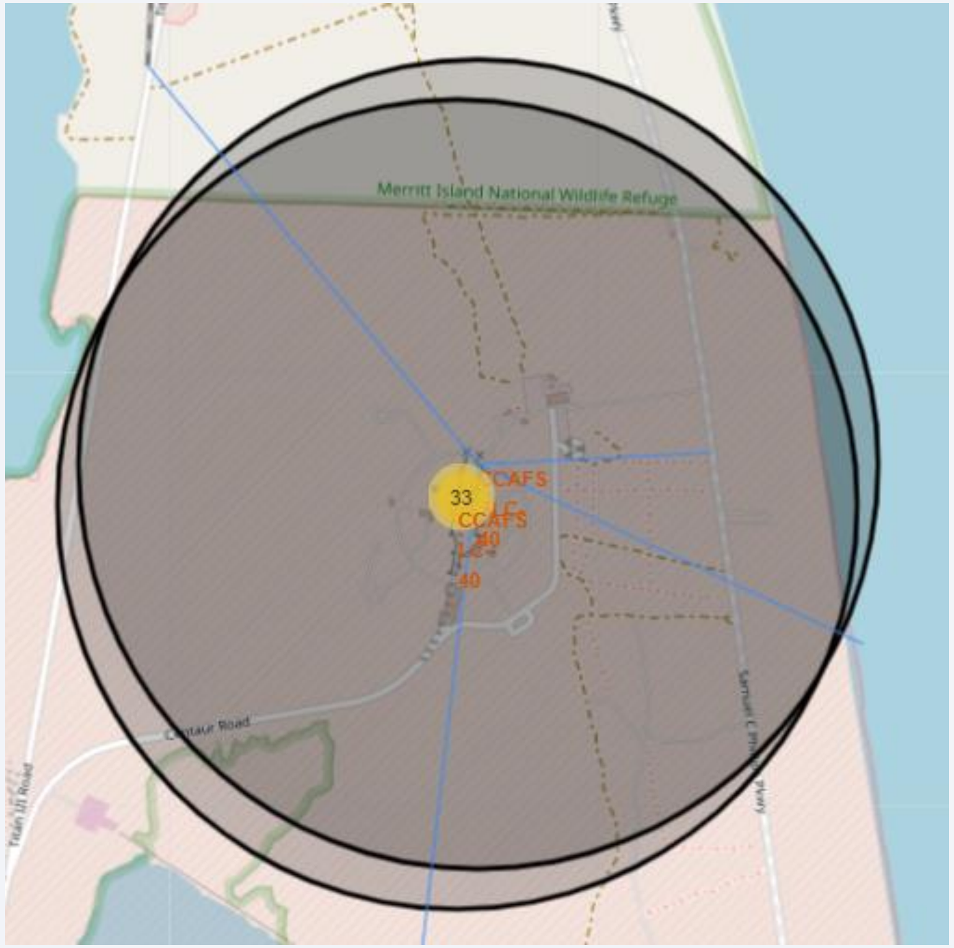


Markers with Red Color indicate Failed Launches and green markers indicate successful launches for a particular launch site.



Launch site Promities

The following map shows the Polyline bwtween Launch site and proximities such as coastline, highway, railway and city.



Section 5

Build a Dashboard with Plotly Dash

<Dashboard Screenshot 1>

- Replace <Dashboard screenshot 1> title with an appropriate title

- Show the screenshot of launch success count for all sites, in a piechart

- Explain the important elements and findings on the screenshot

<Dashboard Screenshot 2>

- Replace <Dashboard screenshot 2> title with an appropriate title
- Show the screenshot of the piechart for the launch site with highest launch success ratio
- Explain the important elements and findings on the screenshot

<Dashboard Screenshot 3>

- Replace <Dashboard screenshot 3> title with an appropriate title

- Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

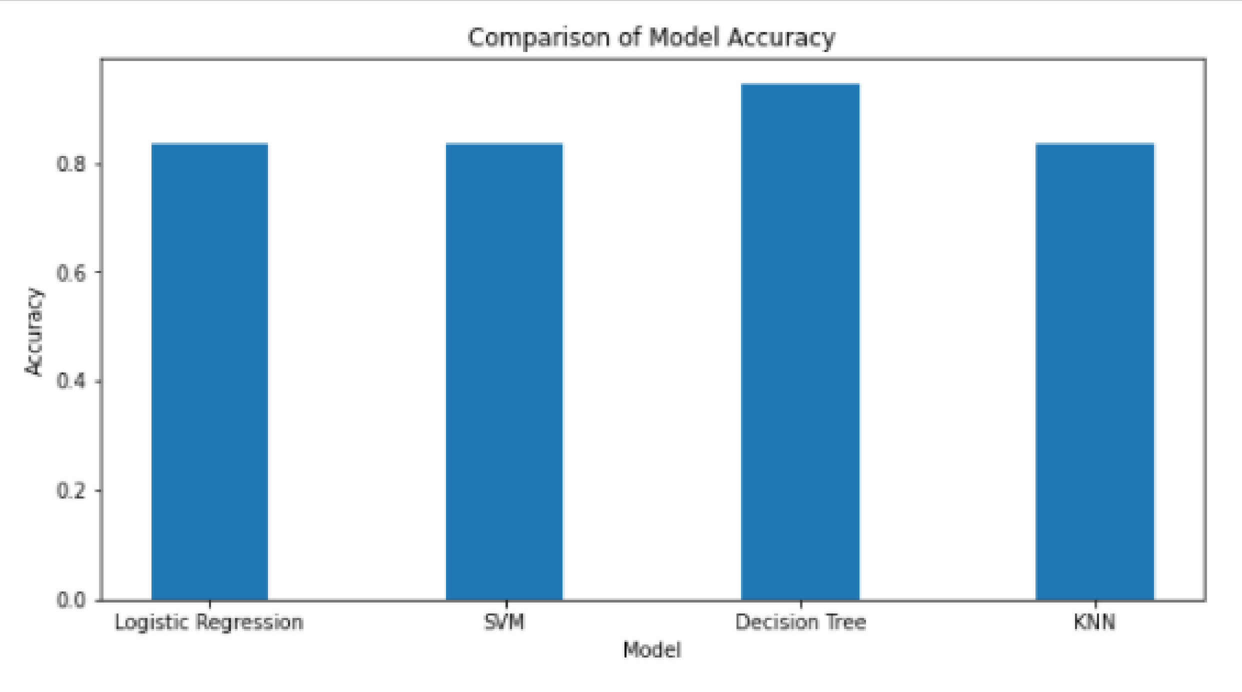
- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.

Section 6

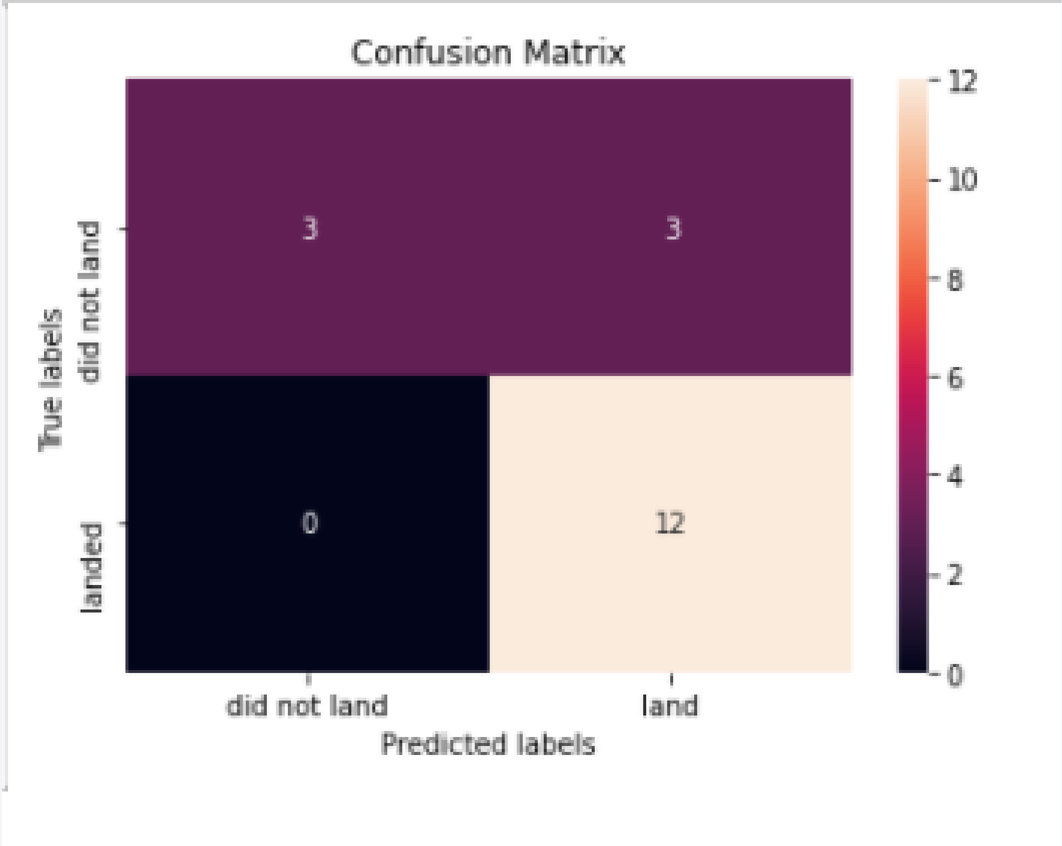
Predictive Analysis (Classification)

Classification Accuracy

- Visualize the built model accuracy for all built classification models, in a bar chart
- Decision tree has the highest accuracy with 94.4 % followed by others with similar accuracy at 83.34 %



Confusion Matrix



Conclusions

- Success rate of launches have increased over the years
- Different Models predict the outcome with similar accuracy
- Decision Tree has the highest accuracy over other models for test data

```
best_method = {'Logistic Regression':accuracy_score(Y_test,Y_pred1),
               'SVM':accuracy_score(Y_test,Y_pred2),
               'Decision Tree':accuracy_score(Y_test,Y_pred3),
                 'KNN':accuracy_score(Y_test,Y_pred4)}
best_method
```

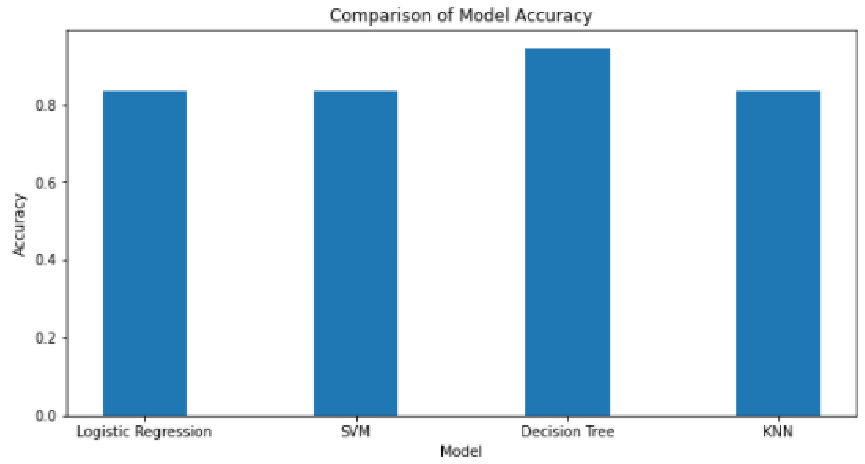
```
0]: {'Logistic Regression': 0.8333333333333334,
     'SVM': 0.8333333333333334,
     'Decision Tree': 0.9444444444444444,
     'KNN': 0.8333333333333334}
```

```
models = list(best_method.keys())
accuracy = list(best_method.values())

fig = plt.figure(figsize = (10, 5))

# creating the bar plot
plt.bar(models, accuracy,width = 0.4)

plt.xlabel("Model")
plt.ylabel("Accuracy")
plt.title("Comparison of Model Accuracy")
plt.show()
```



Thank you!