IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Jirayu KaewprateepMar
27, 2023

Outline

- Executive Summary : Applied Data Science Capstone projects and hand on assignments.
- Introduction : This is part of IBM data sciences class on Coursera, results from laboratories and exames are display and interviews with comments from instructors.
- Methodology : I working on the projects, laboratory, VDO, reading part, textbook followed by the course out-line and study actively with online information I could find and exames and assignments practice from the course.
- Results : I done all the labs even only few lab required time for running that is because there are some technical issues I found in many labs and I complete codes and configurations or do it off line on my PC that old but I finish all with some warning or errors and time spending. The benefits from this is I need to plan and resovled problems but one thing I forgot can anyone pay the IBM DB Clound Bills for me? That is because of the Watson gives you credits but DB using more time but not have credits and all those busy tasks along months of me trying to complete this subject I opened DB Clound for week ; (
- Conclusion : Working on course assignments are not too hard and some time it is too easy, I do it during weekend and with my environments I need to do it quitely within short of time, all assignments and quizs had there answer.
- Appendix : The capstone does not provided Appendix as other subject in the outline

Executive Summary

- Summary of methodologies : Start from this line, how do we achieved new knowledge in a short of time and do the computer do the same way with us? We setup experiments that students and computer can do without difficulty by learning a new sciences that you may know from different visualizing or views and how fast you can learn to have the same performance. This is the first time I use Scikit for data sciences but I used to Tensorflows and answer questions on the StackOverflow, I spend sometimes with reading and try to use Panda DataFrame which I know it a bit when I am using Tensoflow but now we running into it features, Numpy, easyplot, dash and SciKit learn. Computer can run what ever I input but I need to complied new time of Panda and components that also including updated VSCode, C++ runtimes distribution, MySQL setup, MYSQLClient for Python, Frameworks and etc. The computer never reject is you create correct input but it may response in different way. Human is more emotional now I am hungry and my cats (pet) too but I learn I need to finish something to earns some money for the next day.

- Summary of all results : We are smart in different way computer does not need to go out find jobs and food it consume electrical power and spare part or upgrade but it working more speed than us but we the user need to input the correct way same as my cats (pet) they going to bites me if I am not feed them now.

RACE	Basic requirements	Input from human	Output to human
Human (user)	Food, activities	Food and activities	Lives continue, flexibility and kind responses
Computer	Electronics power and upgrade part	Frameworks, power, parts, objectives and management skills	Output from objectives and errors
Cat	Food, activities and toys	Food, shelters and loves	Lives continue, flexibility and kind responses

Introduction

- Project background and context : We setup our projects from learning study new thing when I am working on different skills (direct). Starting from use an old PC setup program frameworks and connectivities until finish the projects and have the result verified. The controls parameters is the course learning time estimates, scores output and assignments done with programming and documents created by myself when I need to take care of the computer, cats (pet), home and working for servibility. I have some experiences in deep-learning and machine learning I create this project by the course assignment but I will not make it too boring.
- Problems you want to find answers : How much of time people can learn a new thing in sciences with goals for themselves future and how difficult they stop do it. The answer I found is most of effectives variables is not people or computer or cats but environments have some effects and budgets is most effect to our main objectives.

RACE	V1	V2	V3	V4	V5
HUMAN	energy	Time available	School subjects	Budgets	Plan and objectives
COMPUTER	energy	Conditions	Software requirements	Human patience	Human patience
CATS	energy	Air environments	Air environments	Air environments	Air environments

Section 1

Methodology

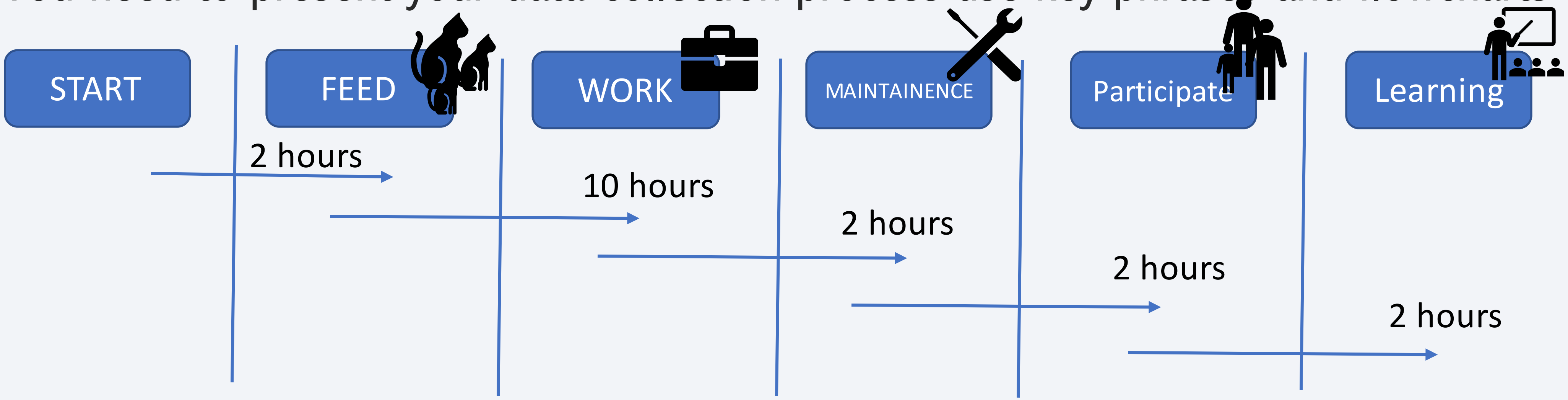
Methodology

Executive Summary

- **Data collection methodology:**
  - We setup goals by learning results and knowledge effectives by remains of all in party. Our data are come from learning results, working on assignments and improving goals.
- **Perform data wrangling**
  - Myself, my old PC and my cats during available time and study time (this month for all three courses) by going to works, travels for 3 hours each day have the same activities as we assumed I am ordinary student, my cats too but my PC is stonehead util the results.
- **Perform exploratory data analysis (EDA) using visualization and SQL**
  - To find some data patterns and visualize, I concluded in submitting laboratory files but for the experiments table draws show you some patterns.
- **Perform interactive visual analytics using Folium and Plotly Dash**
  - The laboratory files are submitted but plotting and Folium can perform by tools such as Google Traffics or 360 map that contained my foot prints and time steps.
- **Perform predictive analysis using classification models**
  - We can build a table and patterns and we can expecting results from it, the laboratory files also submitted.

Data Collection

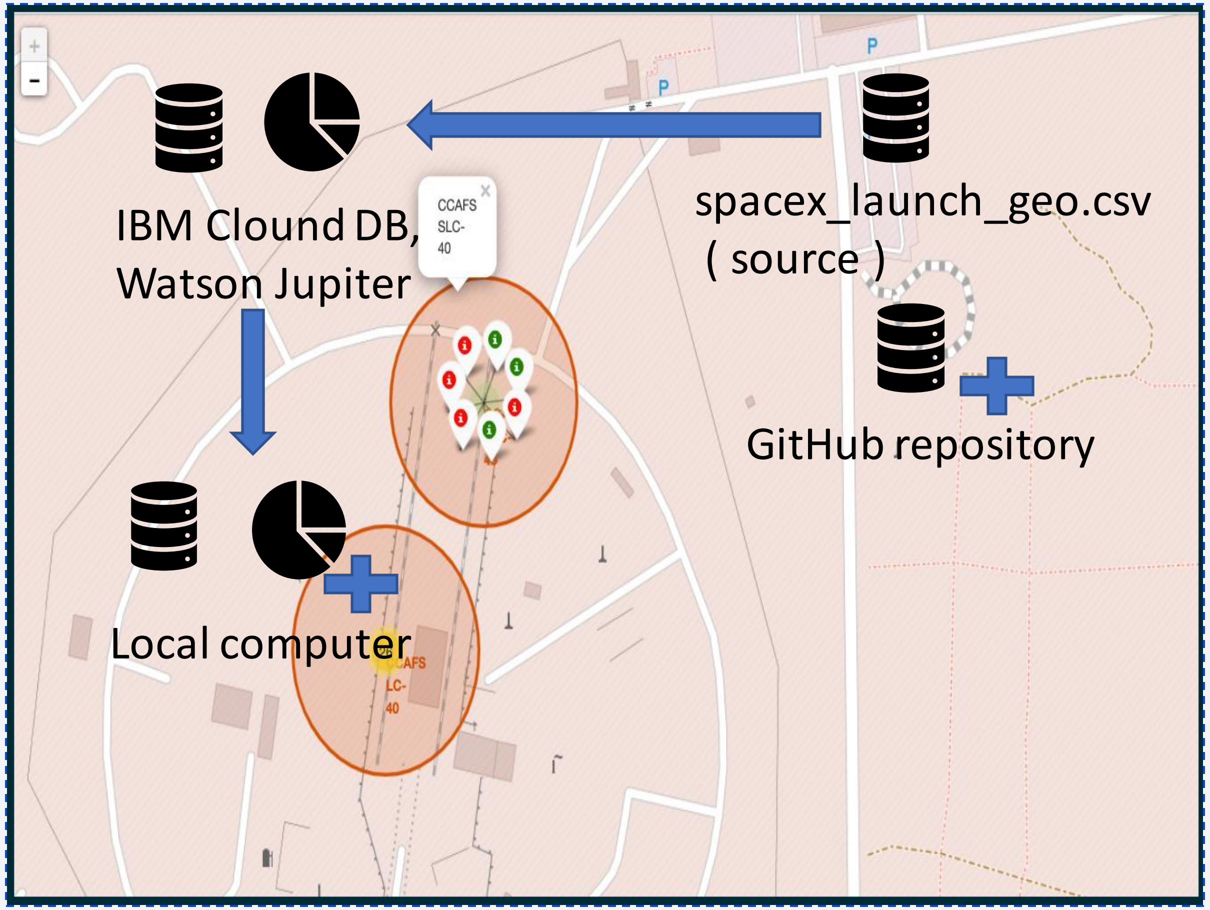
- Describe how data sets were collected.
  - By GPRS tracking in Google, mobile applications and my activities.
  - Learning result during past 2 months and living with manage live and servivibitity.
  - Household and activities because have pets and environments.
- You need to present your data collection process use key phrases and flowcharts



Data Collection – SpaceX API

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

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

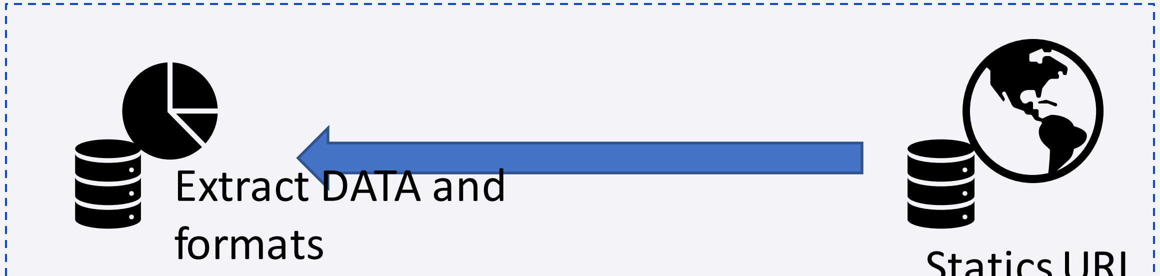


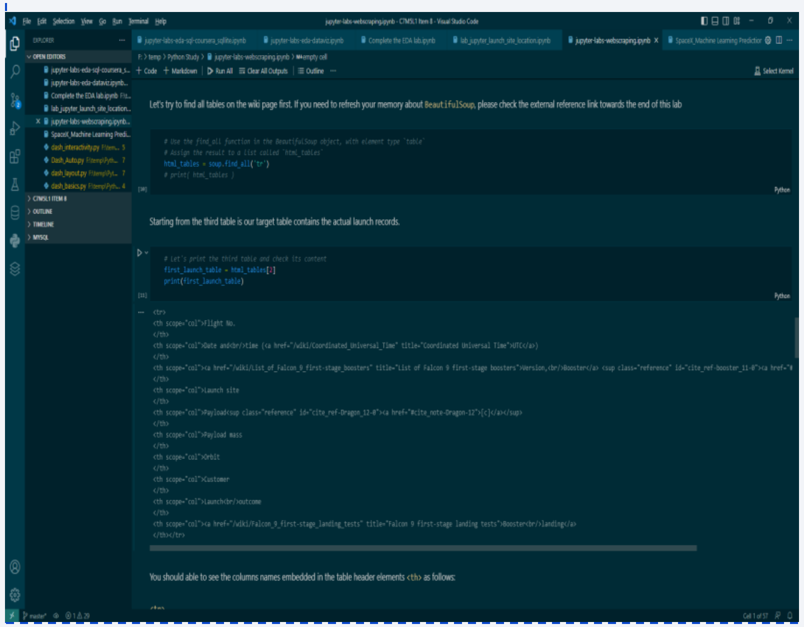
[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)](https://github.com/jkaewprateep/Applied-Data-Science-Capstone)

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

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




jupyter-labs-webscraping.ipynb - C7MSL1 Item 8 - Visual Studio Code
jupyter-labs-eda-sql-coursera_sqllite.ipynb
jupyter-labs-eda-dataviz.ipynb
Complete the EDA lab.ipynb
lab_jupyter_launch_site_location.ipynb
jupyter-labs-webscraping.ipynb
SpaceX_Machine Learning Prediction
+ Code + Markdown Run All Clear All Outputs Outline
Select Kernel

Let's try to find all tables on the wiki page first. If you need to refresh your memory about BeautifulSoup, please check the external reference link towards the end of this lab

```python
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('tr')
# print( html_tables )
```

Starting from the third table is our target table contains the actual launch records.

```python
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

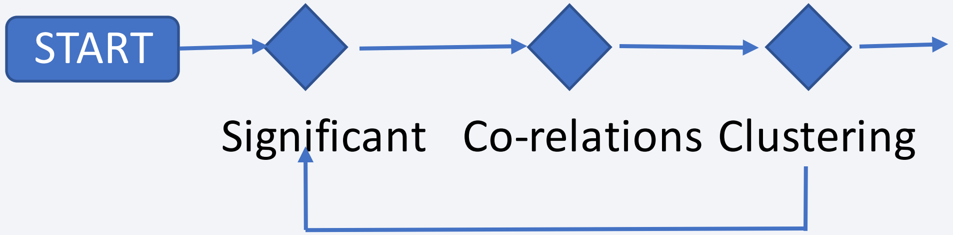
```
<tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">Version,<br/>Booster</a> <sup class="reference" id="cite_ref-booster_11-0"><a href="#
</th>
<th scope="col">Launch site
</th>
<th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-0"><a href="#cite_note-Dragon-12">[c]</a></sup>
</th>
<th scope="col">Payload mass
</th>
<th scope="col">Orbit
</th>
<th scope="col">Customer
</th>
<th scope="col">Launch<br/>outcome
</th>
<th scope="col"><a href="/wiki/Falcon_9_first-stage_landing_tests" title="Falcon 9 first-stage landing tests">Booster<br/>landing</a>
</th></tr>
```

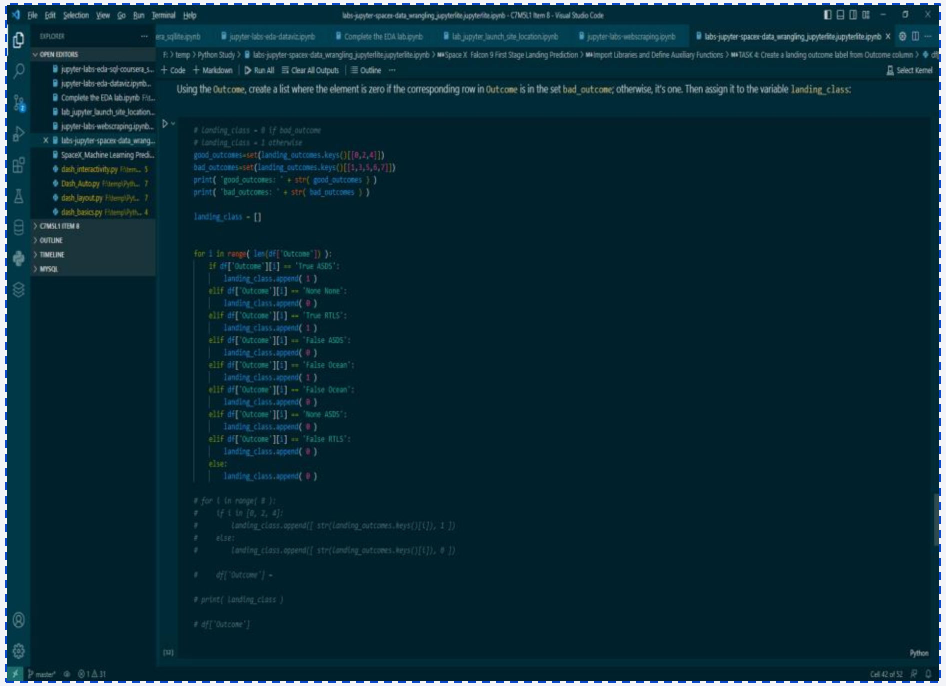
You should able to see the columns names embedded in the table header elements <th> as follows:

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Data Wrangling

- Describe how data were processed
  - You simply categorized from input data, data present into groups or clusters and removed data or add some fields such as average, total, mean and standards suitable for your learning machine, methods and objectives.
  - Toal, mean, summarized and standards can perform by tools and easy to find it relationship with the input data or selected when these process may require time running when machine unsupervised learning.
- You need to present your data wrangling process using key phrases and flowcharts
- Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose





Using the Outcome, create a list where the element is zero if the corresponding row in Outcome is in the set bad_outcome; otherwise, it's one. Then assign it to the variable landing_class:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
good_outcomes=set(landing_outcomes.keys()[[0,2,4]])
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
print( 'good_outcomes: ' + str( good_outcomes ) )
print( 'bad_outcomes: ' + str( bad_outcomes ) )

landing_class = []


for i in range( len(df['Outcome']) ):
    if df['Outcome'][i] == 'True ASDS':
        landing_class.append( 1 )
    elif df['Outcome'][i] == 'None None':
        landing_class.append( 0 )
    elif df['Outcome'][i] == 'True RTLS':
        landing_class.append( 1 )
    elif df['Outcome'][i] == 'False ASDS':
        landing_class.append( 0 )
    elif df['Outcome'][i] == 'False Ocean':
        landing_class.append( 1 )
    elif df['Outcome'][i] == 'False Ocean':
        landing_class.append( 0 )
    elif df['Outcome'][i] == 'None ASDS':
        landing_class.append( 0 )
    elif df['Outcome'][i] == 'False RTLS':
        landing_class.append( 0 )
    else:
        landing_class.append( 0 )

# for i in range( 8 ):
#     if i in [0, 2, 4]:
#         landing_class.append([ str(landing_outcomes.keys()[i]), 1 ])
#     else:
#         landing_class.append([ str(landing_outcomes.keys()[i]), 0 ])

#     df['Outcome'] =

# print( landing_class )

# df['Outcome']
```

[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)](https://github.com/jkaewprateep/Applied-Data-Science-Capstone)

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts
    - From the course outline
    - Working from initial to end of the machine learning and data visualization, presents of works in each step and proved by compared them with multiple methods and present of its meaning or theoretical and mathematical methods sample co-relation, tree and graphs.
    - Create present of data layout for presentations
    - Implementation of data integrations and application frameworks
- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed

Most of the query are from the same pattern which are filter, date time inputs, conditions, transform and removed fields then summarized.

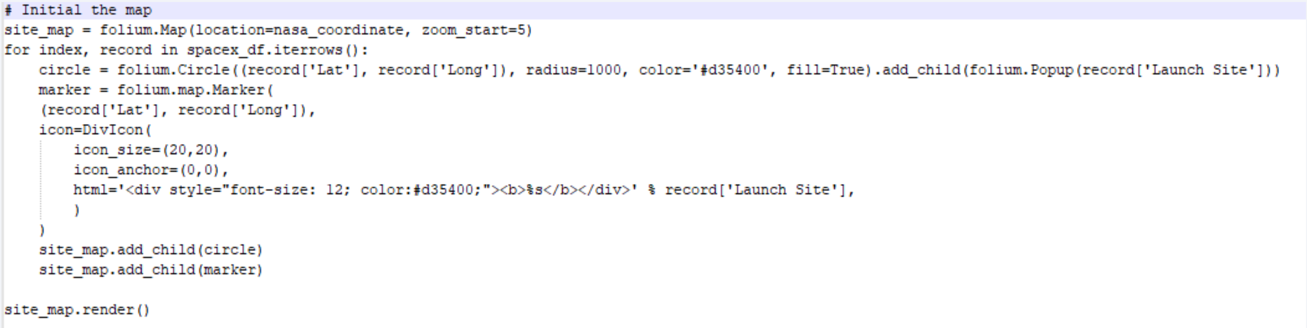
```
result = pd.DataFrame( df[(df['Landing_Outcome']  == 'Success (ground pad)') | (df['Landing_Outcome']  == 'Success') | (df['Landing_Outcome']  == 'Success (drone ship)')] )
result = result[ result['Date'] >= '04-06-2010' ]
result = result[ result['Date'] <= '20-03-2017' ]
result = result.sort_values(by=['Date'])

ls_landing_outcomes = pd.DataFrame([ 'Success (ground pad)', 'Success (drone ship)', 'Success' ])
ls_landing_outcomes
ls_landing_outcomes = ls_landing_outcomes.sort_values(by=[0], ascending=False).values.tolist()
ls_landing_outcomes

for landing_outcomes in ls_landing_outcomes:
    count = result[result['Landing_Outcome'] == landing_outcomes[0]]['Booster_Version'].count()
    print( landing_outcomes[0] + ': ' + str( count ) )
```

- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map



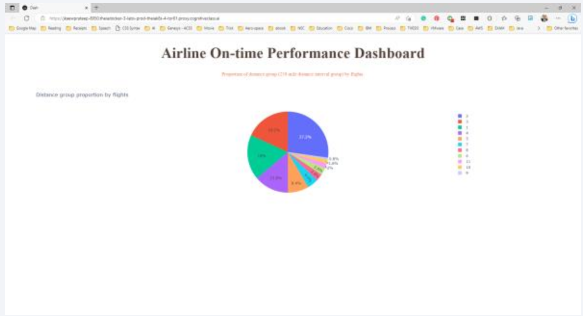
```
# Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
for index, record in spacex_df.iterrows():
    circle = folium.Circle((record['Lat'], record['Long']), radius=1000, color='#d35400', fill=True).add_child(folium.Popup(record['Launch Site']))
    marker = folium.map.Marker(
    (record['Lat'], record['Long']),
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % record['Launch Site'],
        )
    )
    site_map.add_child(circle)
    site_map.add_child(marker)

site_map.render()
```

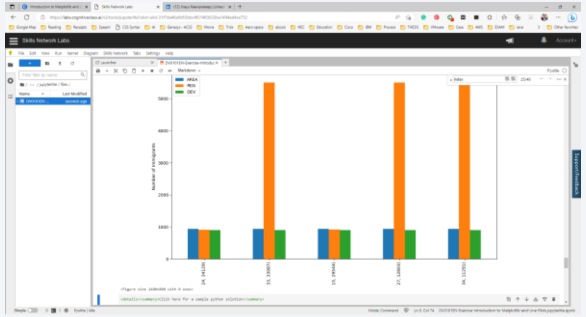
- Explain why you added those objects
  - The assignment objective using map is located, remarks and create pop-up when co-ordinate and location are from input data
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

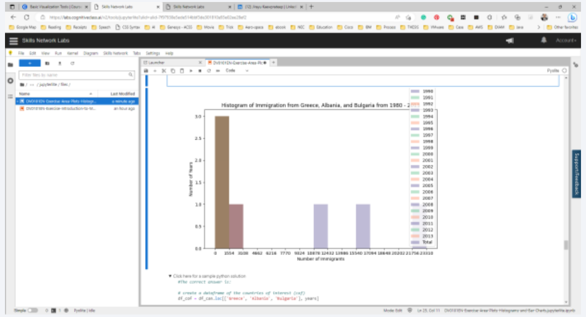
GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard


Airline On-time Performance Dashboard
Distance group proportion by flights




Skills Network Labs
Histogram of Immigration from Greece, Albania, and Bulgaria from 1980 - 2013
Number of Years
Number of Immigrants

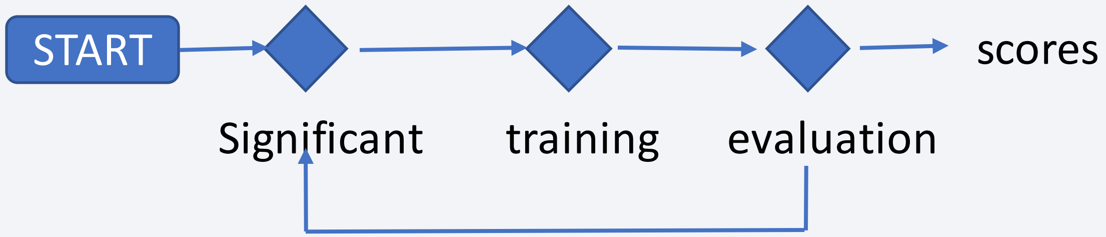
```
#The correct answer is:

# create a dataframe of the countries of interest (cof)
df_cof = df_can.loc[['Greece', 'Albania', 'Bulgaria'], years]
```

- Explain why you added those plots and interactions
    - It is presentation with users requirement, I am keep doing all assignments, I will submit and continue study. All assignments are done but have some issues I fixed it before have these graphs present.
- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
  - From exploring data, analysis categorize by grouping and patterns or significant finding, training and evaluate with models, decision tree, matrix calculation and classification.
  - Evaluating by data input folds, matrices of loss and accuracy and scores mapping output and compared results.
  - Training and testing, references and re-perform solution and experiments
- You need present your model development process using key phrases and flowchart

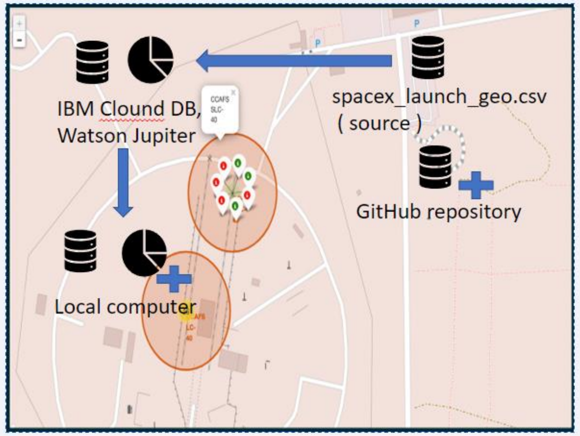


- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Results

- Exploratory data analysis results
  - Model evaluation and catagorrize targets
  - Presentation and data visualization
- Interactive analytics demo in screenshots
  - Graph and response data input dash board and dataset
  - Machine learning models
- Predictive analysis results
  - Model prediction scores

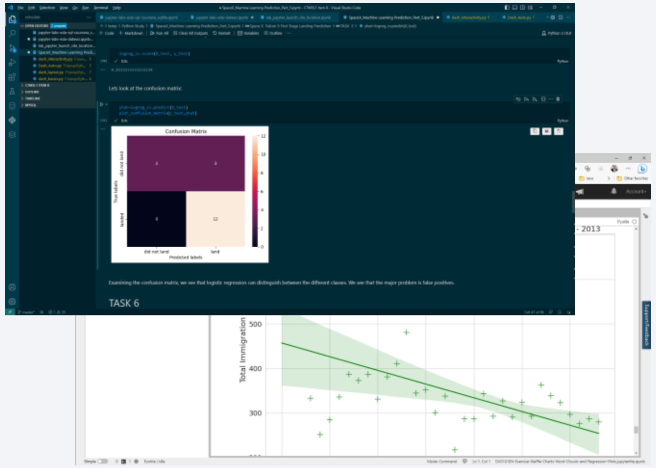


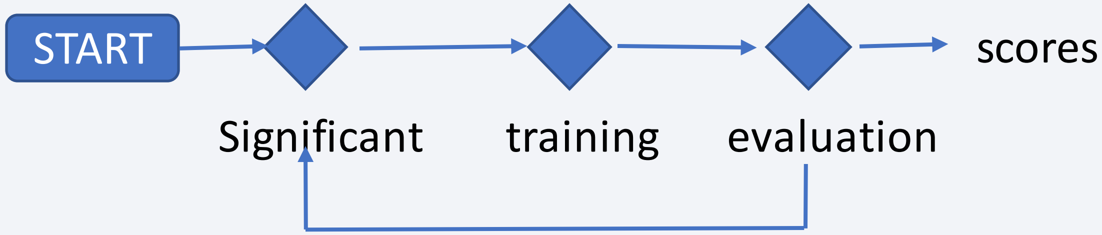
IBM Clound DB, Watson Jupiter

spacex_launch_geo.csv (source)

GitHub repository

Local computer





START

Significant

training

evaluation

scores

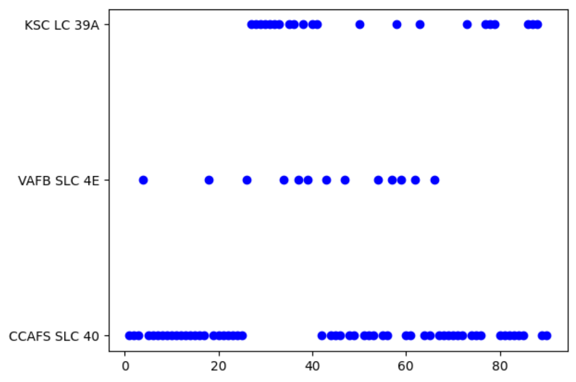
[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)]

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

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

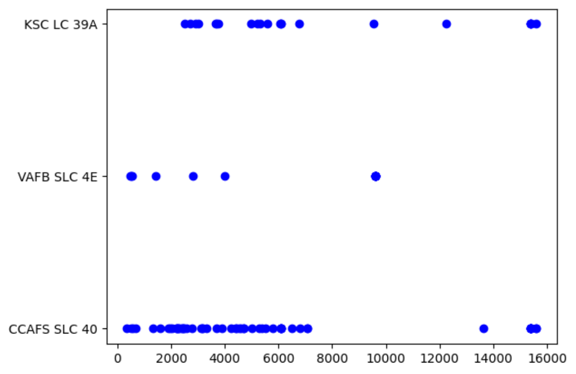


- Show the screenshot of the scatter plot with explanations

1. From question to evaluation of Launch sites and Flight number in order, famous and plan for readiness
2. Number of distribution of flight evaluate
3. Comparable and measurable sample

[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)](https://github.com/jkaewprateep/Applied-Data-Science-Capstone)

Payload vs. Launch Site



- Show a scatter plot of Payload vs. Launch Site

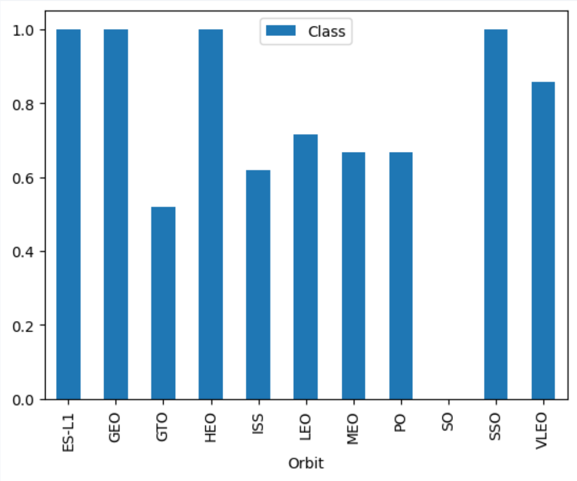
- Show the screenshot of the scatter plot with explanations

1. From question to evaluation of Launch sites and Payload in order, famous and plan for readiness
2. Number of distribution of flight evaluate
3. Comparable and measurable sample

[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)]

Success Rate vs. Orbit Type

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



- Show the screenshot of the scatter plot with explanations

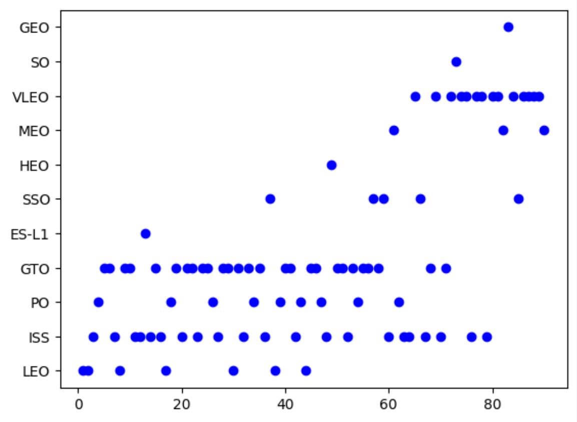
1. From question to evaluation of success rate and orbit type, famous and plan for readiness
2. Number of distribution of orbit evaluate
3. Comparable and measurable sample

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Flight Number vs. Orbit Type

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

- Show the screenshot of the scatter plot with explanations



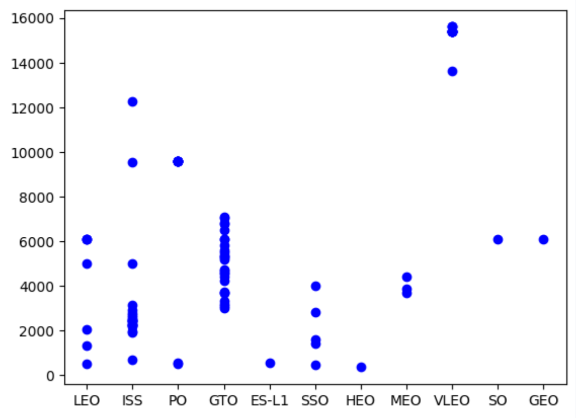
1. From question to evaluation of Flight Number and orbit type, famous and plan for readiness
2. Number of distribution of orbit evaluate
3. Comparable and measurable sample

[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)](https://github.com)

Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type

- Show the screenshot of the scatter plot with explanations



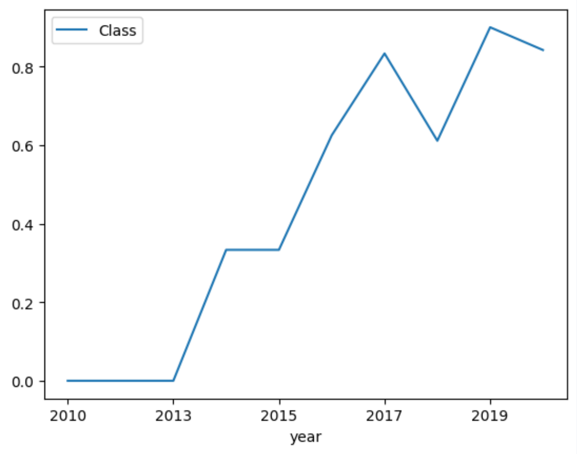
1. From question to evaluation of payload and orbit type, famous and plan for readiness
2. Number of distribution of payload evaluate
3. Comparable and measurable sample

[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)](#)

Launch Success Yearly Trend

- Show a line chart of yearly average success rate

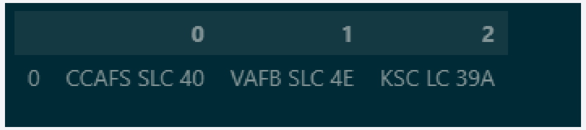
- Show the screenshot of the scatter plot with explanations



1. From question to evaluation of yearly average success rate, famous and plan for readiness
2. Number of distribution of yearly average success rate evaluate
3. Comparable and measurable sample

[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)]

All Launch Site Names



	0	1	2
0	CCAFS SLC 40	VAFB SLC 4E	KSC LC 39A

- Find the names of the unique launch sites

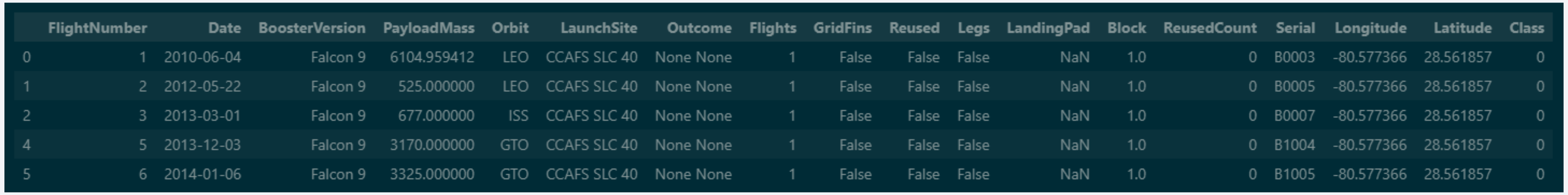
```
result = data['LaunchSite'].unique()
result = pd.DataFrame([ result ])
result.head(100)
```

- Present your query result with a short explanation here
  - Unique value from field

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Launch Site Names Begin with 'CCA'

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



	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0
5	6	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1005	-80.577366	28.561857	0

- Present your query result with a short explanation here

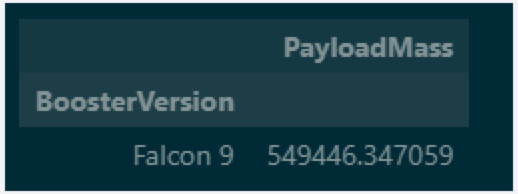
```
result = pd.DataFrame(data[data['LaunchSite'].str.contains('CCA')])
result.head(5)
```

1. String contain 'CCA' for data as dataframe

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Total Payload Mass

- Calculate the total payload carried by boosters from NASA



	PayloadMass
BoosterVersion	
Falcon 9	549446.347059

- Present your query result with a short explanation here

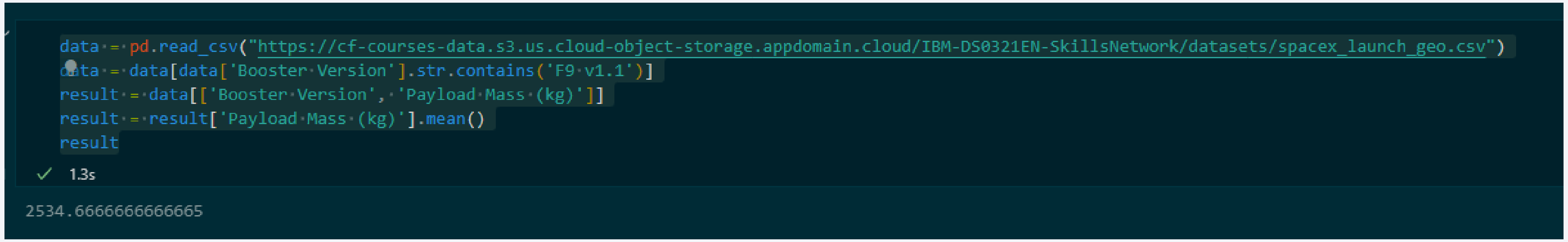
```
result = data[['BoosterVersion', 'PayloadMass']]
result = result.groupby(['BoosterVersion']).sum()
result
```

1. Group by 'BoosterVersion' and summary number of 'PayloadMass'.

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Average Payload Mass by F9 v1.1

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



```
data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_geo.csv")
data = data[data['Booster Version'].str.contains('F9 v1.1')]
result = data[['Booster Version', 'Payload Mass (kg)']]
result = result['Payload Mass (kg)'].mean()
result
```

✓ 1.3s

```
2534.6666666666665
```

- Present your query result with a short explanation here

data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_geo.csv")

data = data[data['Booster Version'].str.contains('F9 v1.1')]

result = data[['Booster Version', 'Payload Mass (kg)']]

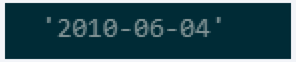
result = result['Payload Mass (kg)'].mean()

Result

1. Average pay load of specific engines

First Successful Ground Landing Date

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



```
'2010-06-04'
```

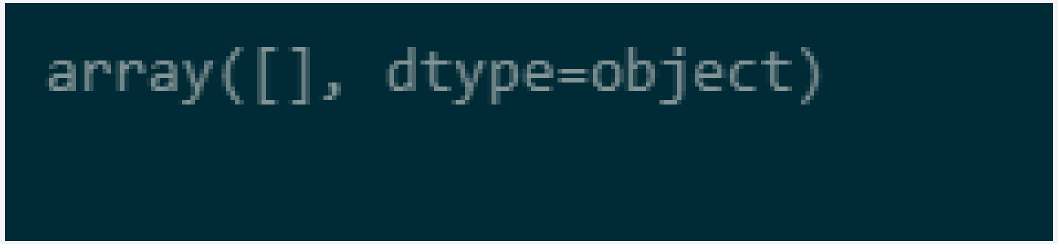
- Present your query result with a short explanation here

```
data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_geo.csv")
result = data[ ( data['Landing Outcome'].str.contains('Success  (ground pad)') ) | ( data['Landing Outcome'].str.contains('Success (ground pad)' ) )]
data['Date'].min()
```

1. Landing Outcome of successful landing on ground pad.

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



```
array([], dtype=object)
```

- Present your query result with a short explanation here

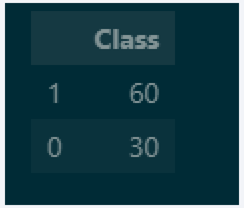
```
data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_geo.csv")
result = data[ ( data['Landing Outcome'].str.contains('Success (drone ship)') ) | ( data['Landing Outcome'].str.contains('Success  (drone ship)' ) )]
result = result[ data['Payload Mass (kg)'] > 4000 ]
result = result[ data['Payload Mass (kg)'] < 6000 ]
result['Booster Version'].unique()
```

1. Condition by weights, drone ship with over 4000 Kgs is 0

[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)]

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

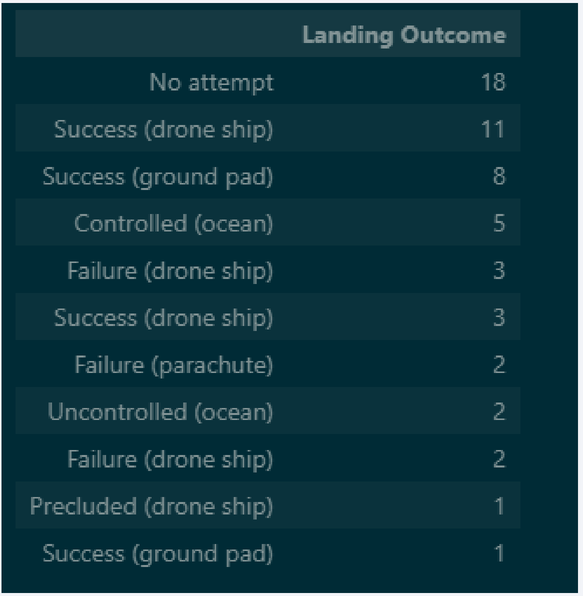


	Class
1	60
0	30

- Present your query result with a short explanation here

```
data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud
SkillsNetwork/datasets/spacex_launch_geo.csv")
result = pd.DataFrame( data['Landing Outcome'].value_counts() )
result
```

1. Counting by Landing outcomes



	Landing Outcome
No attempt	18
Success (drone ship)	11
Success (ground pad)	8
Controlled (ocean)	5
Failure (drone ship)	3
Success (drone ship)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Failure (drone ship)	2
Precluded (drone ship)	1
Success (ground pad)	1

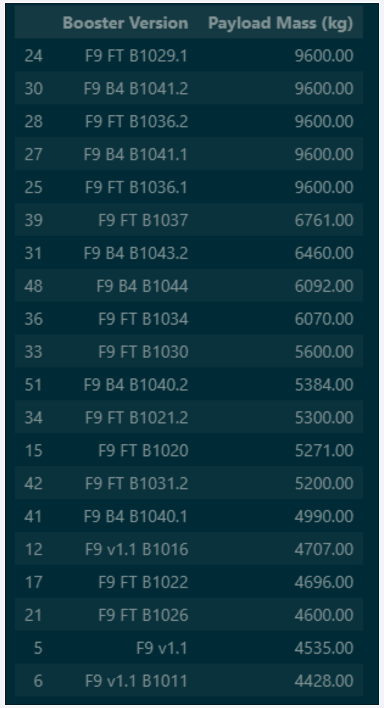
[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)](https://github.com)

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Present your query result with a short explanation here

```
ls_booster = data[ 'Booster Version' ].unique().tolist()
ls_result = []
result = data[['Booster Version', 'Payload Mass (kg)']]
for booster in ls_booster:
   ls_result.append( result[ result[ 'Booster Version' ] == booster].max() )
result = pd.DataFrame( ls_result )
result = result.sort_values(by=['Payload Mass (kg)'], ascending=False)
result
```

1. Selecting by name and maximum weight loads

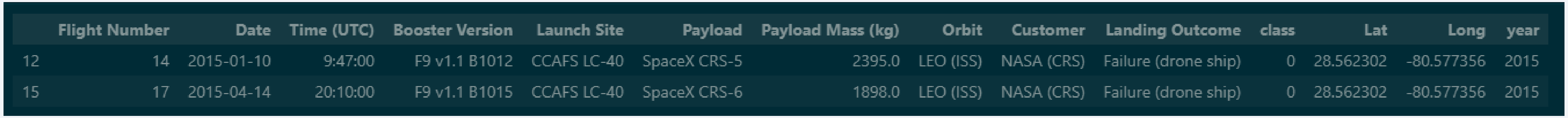


	Booster Version	Payload Mass (kg)
24	F9 FT B1029.1	9600.00
30	F9 B4 B1041.2	9600.00
28	F9 FT B1036.2	9600.00
27	F9 B4 B1041.1	9600.00
25	F9 FT B1036.1	9600.00
39	F9 FT B1037	6761.00
31	F9 B4 B1043.2	6460.00
48	F9 B4 B1044	6092.00
36	F9 FT B1034	6070.00
33	F9 FT B1030	5600.00
51	F9 B4 B1040.2	5384.00
34	F9 FT B1021.2	5300.00
15	F9 FT B1020	5271.00
42	F9 FT B1031.2	5200.00
41	F9 B4 B1040.1	4990.00
12	F9 v1.1 B1016	4707.00
17	F9 FT B1022	4696.00
21	F9 FT B1026	4600.00
5	F9 v1.1	4535.00
6	F9 v1.1 B1011	4428.00

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

2015 Launch Records

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



	Flight Number	Date	Time (UTC)	Booster Version	Launch Site	Payload	Payload Mass (kg)	Orbit	Customer	Landing Outcome	class	Lat	Long	year
12	14	2015-01-10	9:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395.0	LEO (ISS)	NASA (CRS)	Failure (drone ship)	0	28.562302	-80.577356	2015
15	17	2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898.0	LEO (ISS)	NASA (CRS)	Failure (drone ship)	0	28.562302	-80.577356	2015

- Present your query result with a short explanation here

```
data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_geo.csv")
result = data
result['year'] = result['Date'].str[0:4:1]
result = result[(result['Landing Outcome'] == 'Failure (drone ship)') | (result['Landing Outcome'] == 'Failure (drone ship)')]
result = result[result['year']=='2015']
```

1. Filter by custom filed 'year' and Landing Outcome

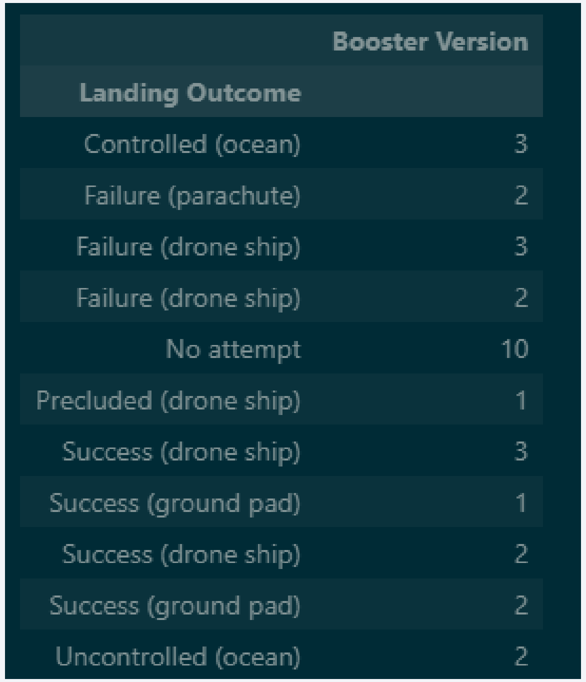
GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

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
- Present your query result with a short explanation here

```
data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_geo.csv")
result = data[['Landing Outcome', 'Date', 'Booster Version']]
result = result[result['Date'] >= '2010-06-04']
result = result[result['Date'] <= '2017-03-20']
result = result[['Landing Outcome', 'Booster Version']]
result = result.groupby(['Landing Outcome']).count()
result
```

1. Filters by date and group by Landing Outcome



	Booster Version
Landing Outcome	
Controlled (ocean)	3
Failure (parachute)	2
Failure (drone ship)	3
Failure (drone ship)	2
No attempt	10
Precluded (drone ship)	1
Success (drone ship)	3
Success (ground pad)	1
Success (drone ship)	2
Success (ground pad)	2
Uncontrolled (ocean)	2

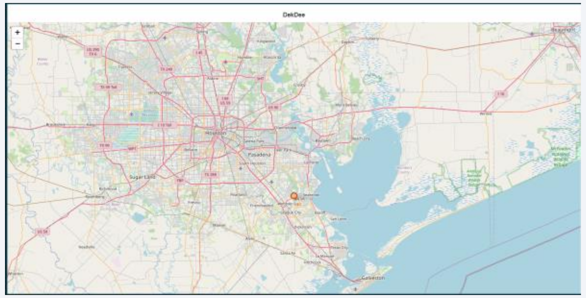
GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Section 3

Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

- Replace <Folium map screenshot 1> title with an appropriate title
- Explore the generated folium map and make a proper screenshot to include all launch sites' location markers on a global map
- Explain the important elements and findings on the screenshot

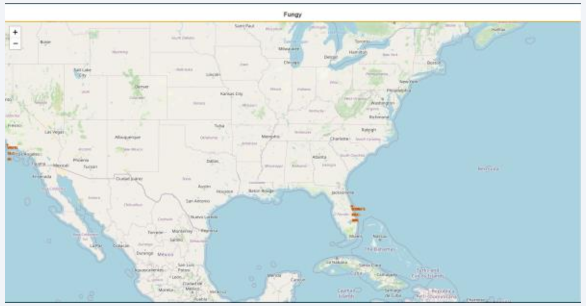


```
   nasa_coordinate = [29.559684888503615, -95.0830971930759]
   site_map = folium.Map(location=nasa_coordinate, zoom_start=10)
   loc = 'DekDee'
   title_html = '''
   <h3 align="center" style="font-size:16px"><b>{}</b></h3>
   '''.format(loc)
   site_map.get_root().html.add_child(folium.Element(title_html))
   site_map.render()

   circle = folium.Circle(nasa_coordinate, radius=1000, color='#d35400', fill=True).add_child(folium.Popup('NASA Johnson Space Center'))
   # Create a blue circle at NASA Johnson Space Center's coordinate with a icon showing its name
   marker = folium.map.Marker(
     nasa_coordinate,
     # Create an icon as a text label
     icon=DivIcon(
       icon_size=(20,20),
       icon_anchor=(0,0),
       html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % 'NASA JSC',
       )
     )

   site_map.add_child(circle)
site_map.add_child(marker)
```

1. Change title and create remarks

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

<Folium Map Screenshot 2>



- Replace <Folium map screenshot 2> title with an appropriate title
- Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map
- Explain the important elements and findings on the screenshot
  - *# Initial the map*
  - site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
  - *# For each launch site, add a Circle object based on its coordinate (Lat, Long) values. In addition, add Launch site name as a popup label*
  - loc = 'Fungy'
  - title_html = ''' <h3 align="center" style="font-size:16px"><b>{}</b></h3>''.format(loc)
  - site_map.get_root().html.add_child(folium.Element(title_html))
  - site_map.render()

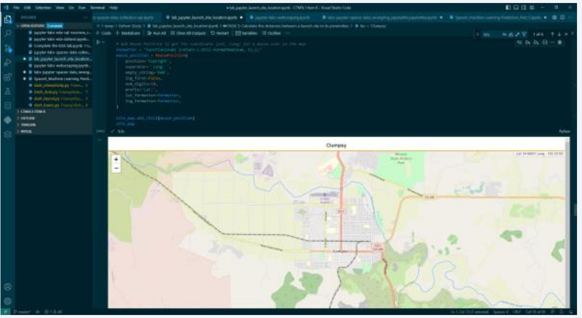
```
site_map.add_child(marker_cluster)
for index, record in spacex_df.iterrows():
   marker_cluster.add_child(marker)
site_map
```

1. Add title name and color labels

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

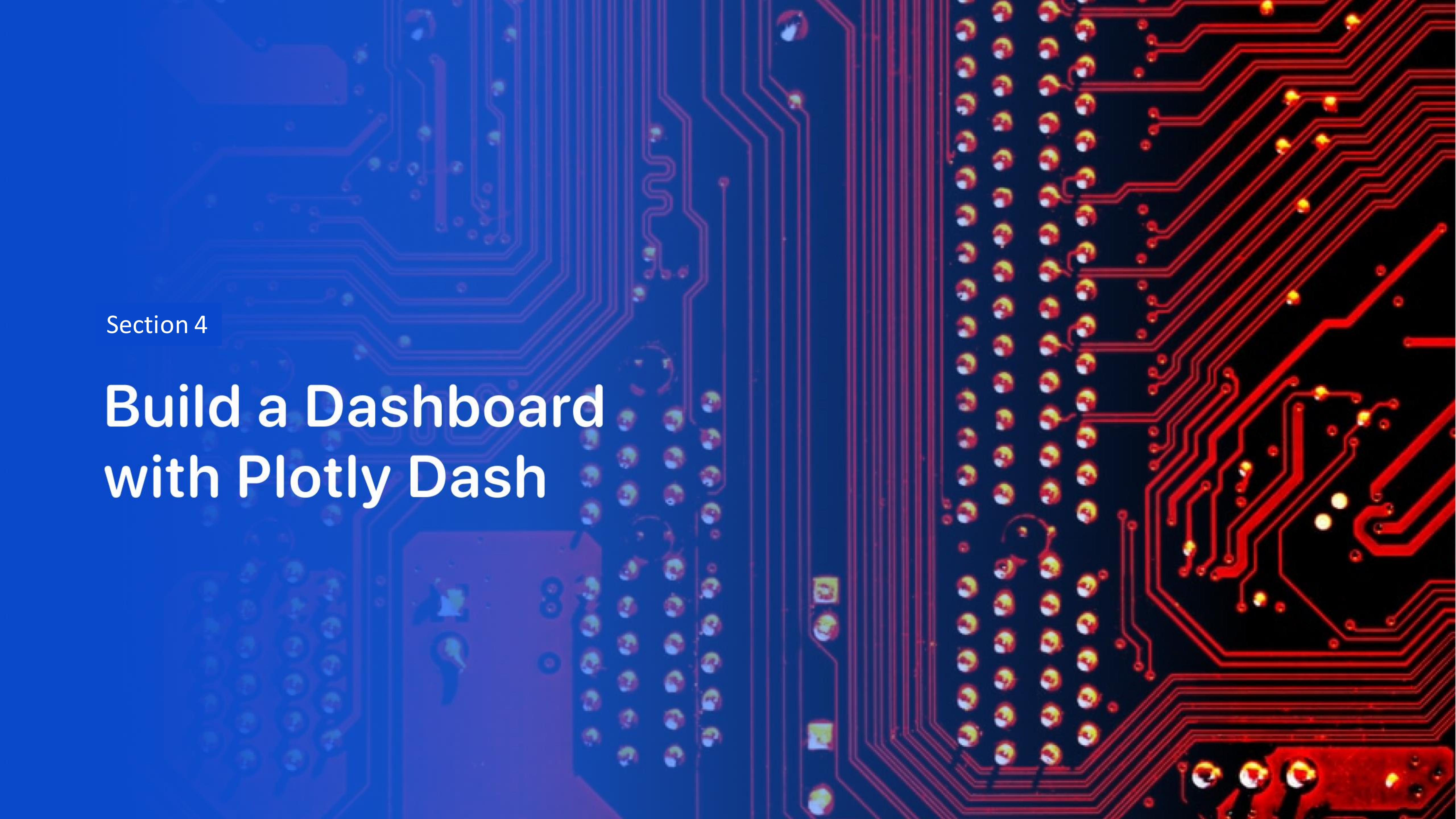
<Folium Map Screenshot 3>

- Replace <Folium map screenshot 3> title with an appropriate title
- Explore the generated folium map and show the screenshot of a selected launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed
- Explain the important elements and findings on the screenshot



```
# Add Mouse Position to get the coordinate (Lat, Long) for a mouse over on the map
formatter = "function(num) {return L.Util.formatNum(num, 5);};"
mouse_position = MousePosition(
    position='topright',
    separator=' Long: ',
    empty_string='NaN',
    lng_first=False,
    num_digits=20,
    prefix='Lat:',
    lat_formatter=formatter,
    lng_formatter=formatter,
)

site_map.add_child(mouse_position)
site_map
```

1. Proximates rails ways and high ways

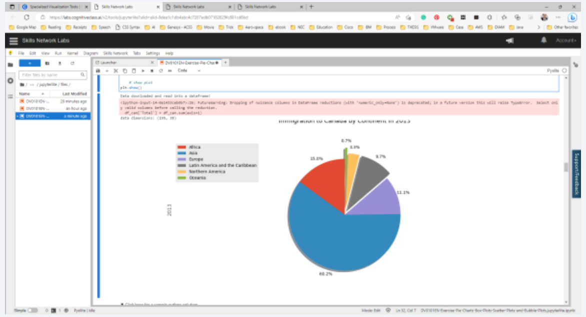


Section 4

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



```
    # show plot
plt.show()
```

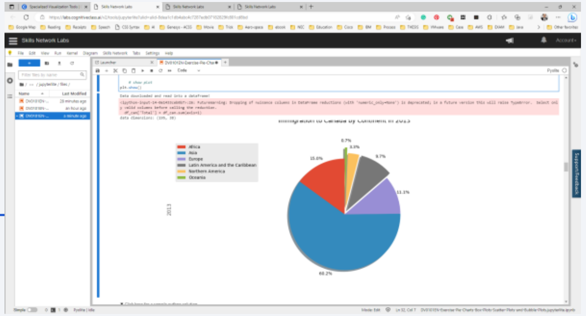
```
Data downloaded and read into a dataframe!
<ipython-input-14-[illegible]>:26: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  df_can['Total'] = df_can.sum(axis=1)
data dimensions: (195, 38)
```

```
@app.callback(Output(component_id='success-pie-chart', component_property='figure'),
              Input(component_id='site-dropdown', component_property='value'))
def get_pie_chart(entered_site):
    ...
```

1. Dash Pie chart (in last lab I am fixing)

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

<Dashboard Screenshot 2>



```
# show plot
plt.show()
```

```
Data downloaded and read into a dataframe!
<ipython-input-14-[illegible]>:16: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select onl
y valid columns before calling the reduction.
  df_can['Total'] = df_can.sum(axis=1)
data dimensions: (195, 39)
```

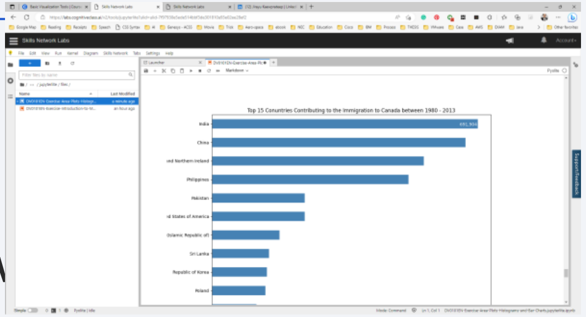
- Replace <Dashboard screenshot 2> title with an appropriate title
- Show the screenshot of the piechart for the launch site with highest launch success ratio
- Explain the important elements and findings on the screenshot

```
@app.callback(Output(component_id='success-pie-
chart', component_property='figure'),
              Input(component_id='site-dropdown', component_property='value'))
def get_pie_chart(entered_site):
    …
```

1. Dash Pie chart (in last lab I am fixing)

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

<Dashboard Screenshot 3>



- Replace <Dashboard screenshot 3> title with an appropriate title
- Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, w 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.

```
@app.callback(Output(component_id='success-pie-chart', component_property='figure'),
          Input(component_id='site-dropdown', component_property='value'))
def get_bar_chart(entered_site):
    ...
```

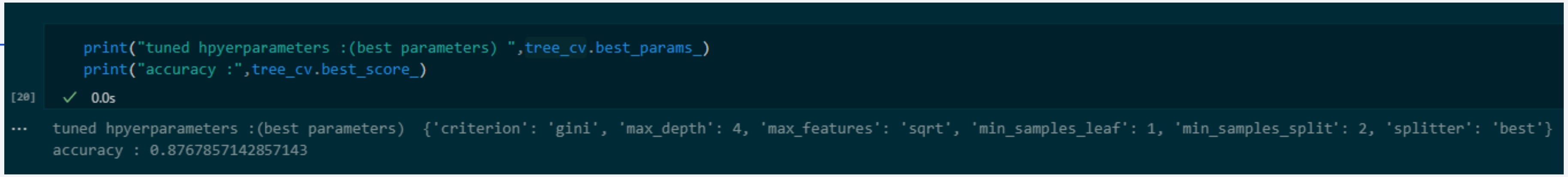
1. Dash BAR chart (in last lab I am fixing)

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Section 5

Predictive Analysis (Classification)

Classification Accuracy



```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

[20] ✓ 0.0s

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
accuracy : 0.8767857142857143
```

- Visualize the built model accuracy for all built classification models, in a bar chart

  tuned hpyerparameters :(best parameters) {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'} accuracy : 0.8767857142857143

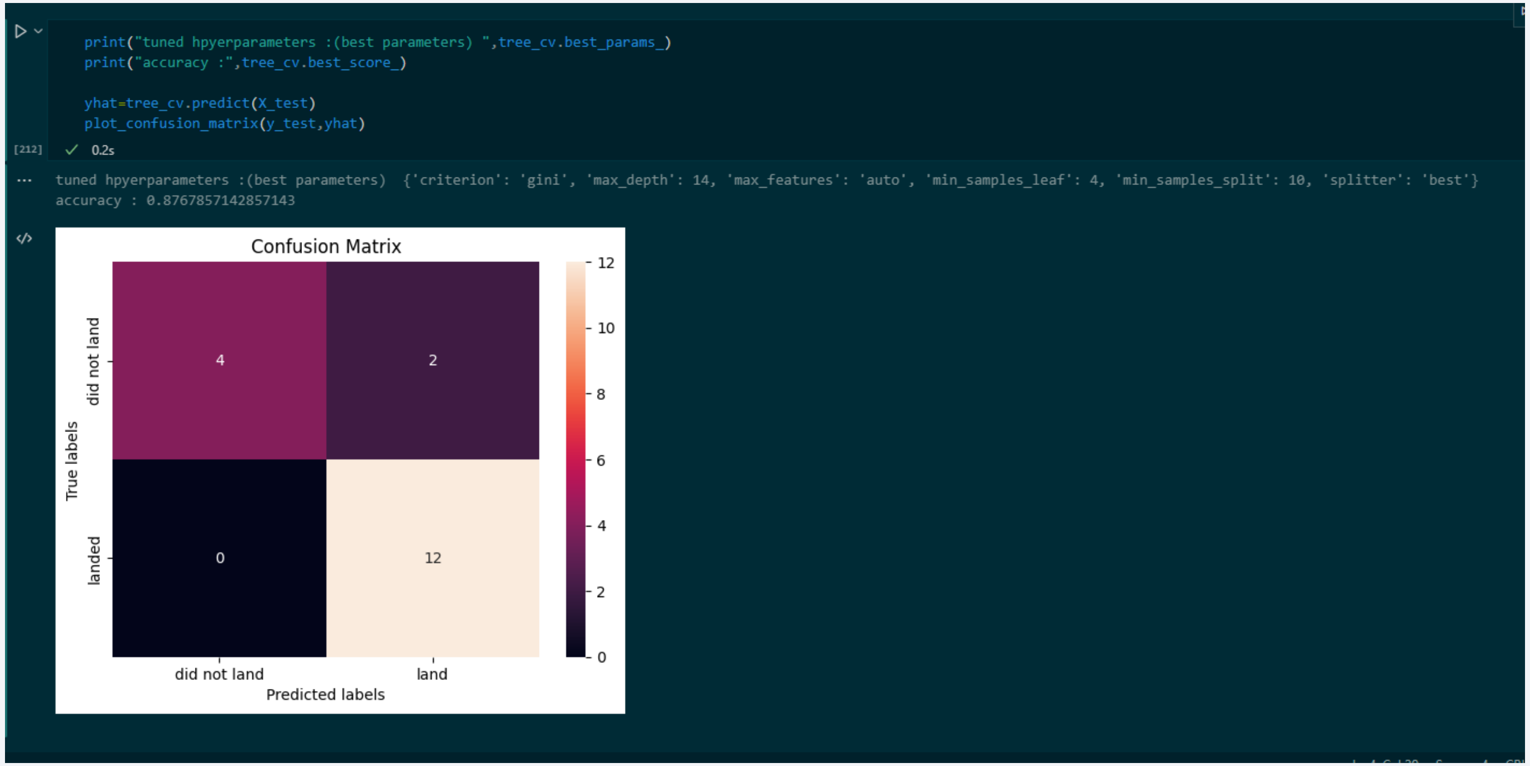
- Find which model has the highest classification accuracy

  1. Tree model

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation



```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)

yhat=tree_cv.predict(X_test)
plot_confusion_matrix(y_test,yhat)
```

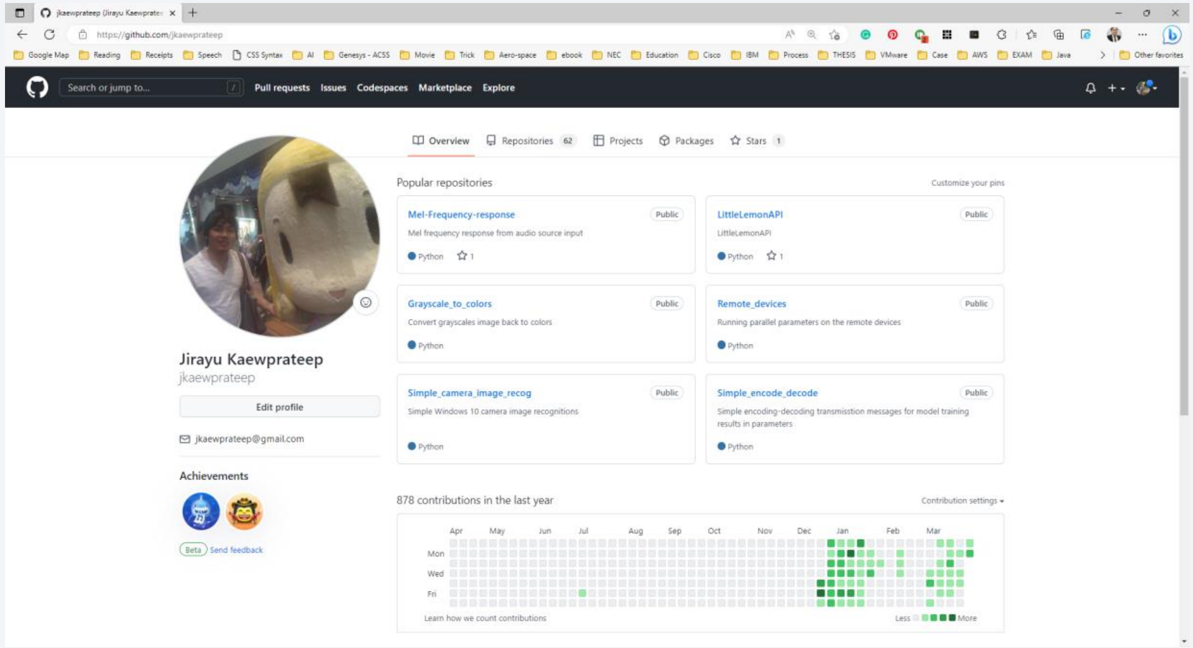
[212] ✓ 0.2s

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 14, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'best'}
accuracy : 0.8767857142857143
```

[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)]

Conclusions

- All assignments are good but please weights too all subjects.
- I will comeback learn and study it again
- New for me because I use Tensorflow
- See you next time



Jirayu Kaewprateep
jkaewprateep

jkaewprateep@gmail.com

Achievements

Popular repositories

Mel-Frequency-response (Public): Mel frequency response from audio source input. Python ☆ 1

LittleLemonAPI (Public): LittleLemonAPI. Python ☆ 1

Grayscale_to_colors (Public): Convert grayscales image back to colors. Python

Remote_devices (Public): Running parallel parameters on the remote devices. Python

Simple_camera_image_recog (Public): Simple Windows 10 camera image recognitions. Python

Simple_encode_decode (Public): Simple encoding-decoding transmisstion messages for model training results in parameters. Python

878 contributions in the last year

[GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)](https://github.com/jkaewprateep/Applied-Data-Science-Capstone)

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project
- I updated codes and documents into GitHub

GitHub: jkaewprateep/Applied-Data-Science-Capstone: Applied Data Science Capstone (github.com)

Thank you!