# Practice Assignment - Part 1: Analyzing wildfire activities in Australia

Estimated time needed: **40** minutes

# Table of Contents

```
<li><a href="#Objectives">Objectives</a></li>
<li>
    <a href="#Setup">Setup</a>
    <ol>
        <li><a href="#Installing-Required-Libraries">Installing
Required Libraries</a></li>
        <li><a href="#Importing-Required-Libraries">Importing Required
Libraries</a></li>
</li>
<li>
    <a href="#Dataset">Dataset</a>
</li>
<li><a href="#Importing Dataset">Importing Dataset</a></li>
<li><a href="#Practice Tasks">Practice Tasks</a></li>
```

---

# Objectives

After completing this lab you will be able to:

- Use visualization libraries such as Matplotlib, Pandas, Seaborn and Folium to create informative plots and charts

---

# Setup

For this lab, we will be using the following libraries:

- `pandas` for managing the data.
- `numpy` for mathematical operations.
- `seaborn` for visualizing the data.
- `matplotlib` for additional plotting tools.

## Installing Required Libraries

The following required libraries are pre-installed in the Skills Network Labs environment. However, if you run this notebook commands in a different Jupyter environment (e.g. Watson Studio or Ananconda), you will need to install these libraries by removing the `#` sign before `%pip` in the code cell below.

```
# All Libraries required for this lab are listed below. The libraries pre-installed on Skills Network Labs are commented.
#%pip install -qy pandas==1.3.4 numpy==1.21.4 seaborn==0.9.0 matplotlib==3.5.0 folium
# Note: If your environment doesn't support "%pip install", use "!mamba install"

%pip install seaborn
%pip install folium
```

## Importing Required Libraries

*We recommend you import all required libraries in one place (here):*

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import folium
%matplotlib inline
print ('Done')
```

---

# Dataset

**Historical Wildfires**

This wildfire dataset contains data on fire activities in Australia starting from 2005. Additional information can be found here.

Variables

- Region: the 7 regions
- Date: in UTC and provide the data for 24 hours ahead
- Estimated_fire_area: daily sum of estimated fire area for presumed vegetation fires with a confidence > 75% for a each region in km2
- Mean_estimated_fire_brightness: daily mean (by flagged fire pixels(=count)) of estimated fire brightness for presumed vegetation fires with a confidence level > 75% in Kelvin

- Mean_estimated_fire_radiative_power: daily mean of estimated radiative power for presumed vegetation fires with a confidence level > 75% for a given region in megawatts
- Mean_confidence: daily mean of confidence for presumed vegetation fires with a confidence level > 75%
- Std_confidence: standard deviation of estimated fire radiative power in megawatts
- Var_confidence: Variance of estimated fire radiative power in megawatts
- Count: daily numbers of pixels for presumed vegetation fires with a confidence level of larger than 75% for a given region
- Replaced: Indicates with an Y whether the data has been replaced with standard quality data when they are available (usually with a 2-3 month lag). Replaced data has a slightly higher quality in terms of locations

---

## Importing Data

```
from js import fetch
import io

URL = "https://cf-courses-data.s3.us.cloud-object-
storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DV0101EN-
SkillsNetwork/Data%20Files/Historical_Wildfires.csv"
resp = await fetch(URL)
text = io.BytesIO((await resp.arrayBuffer()).to_py())
df = pd.read_csv(text)
print('Data read into a pandas dataframe!')
```

Let's look at some samples rows from the dataset we loaded:

```
df.head()
```

---

Let's verify the column names and the data type of each variable

```
#Column names
df.columns

#data type
df.dtypes
```

Notice the type of 'Date' is object, let's convert it to 'datatime' type and also let's extract 'Year' and 'Month' from date and include in the dataframe as separate columns

```
import datetime as dt

df['Year'] = pd.to_datetime(df['Date']).dt.year
df['Month'] = pd.to_datetime(df['Date']).dt.month
```

**Verify the columns again**

```
#verify the columns again
df.dtypes
```

---

## Practice Tasks

### TASK 1.1: Let's try to understand the change in average estimated fire area over time (use pandas to plot)

```
plt.figure(figsize=(12, 6))
df_new=df.groupby('Year')['Estimated_fire_area'].mean()
df_new.plot(x=df_new.index, y=df_new.values)
plt.xlabel('Year')
plt.ylabel('Average Estimated Fire Area (km²)')
plt.title('Estimated Fire Area over Time')
plt.show()
```

---

### TASK 1.2: You can notice the peak in the plot between 2010 to 2013. Let's narrow down our finding, by plotting the estimated fire area for year grouped together with month.

```
plt.figure(figsize=(12, 6))
df_new1=df.groupby(['Year','Month'])['Estimated_fire_area'].mean()
df_new1.plot(x=df_new1.index, y=df_new1.values)
plt.xlabel('Year, Month')
plt.ylabel('Average Estimated Fire Area (km²)')
plt.title('Estimated Fire Area over Time')
plt.show()
```

This plot represents that the estimated fire area was on its peak after 2011, April and before 2012. You can verify on google/news, this was the time of maximum wildfire hit in Austrailia

---

### TASK 1.3: Let's have an insight on the distribution of mean estimated fire brightness across the regions use the functionality of seaborn to develop a barplot

before starting with the plot, why not know the regions mentioned in the dataset?. Make use of unique() to identify the regions in the dataset (apply it on series only)

```
df['Region'].unique()
```

```
plt.figure(figsize=(10, 6))
sns.barplot(data=df, x='Region', y='Mean_estimated_fire_brightness')
plt.xlabel('Region')
plt.ylabel('Mean Estimated Fire Brightness (Kelvin)')
plt.title('Distribution of Mean Estimated Fire Brightness across
Regions')
plt.show()
```

---

TASK 1.4: Let's find the portion of count of pixels for presumed vegetation fires vary across regions we will develop a pie chart for this

```
#Group the data:
df_new = df.groupby('Region')['Count'].sum()
df_new.head()

#Creating a pie chart:
colors_list = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue',
'lightgreen', 'pink']
explode_list = [0.1, 0, 0, 0, 0.1, 0.1] # ratio for each continent
with which to offset each wedge.

df_new.plot(kind='pie',
            figsize=(10, 6),
            autopct='%1.1f%%',
            startangle=90,
            shadow=True,
            labels=None,
            pctdistance=1.12,
            )

plt.title('Percentage of Pixels for Presumed Vegetation Fires by
Region', y=1.12, fontsize = 15)

plt.axis('equal')

# add legend
plt.legend(labels=df_new.index, loc='upper left', fontsize=7)

plt.show()

plt.figure(figsize=(10, 6))
region_counts = df.groupby('Region')['Count'].sum()
plt.pie(region_counts, labels=region_counts.index, autopct='%1.1f%%')
plt.title('Percentage of Pixels for Presumed Vegetation Fires by
Region')
plt.axis('equal')
plt.show()
```

## TASK 1.5: See the percentage on the pie is not looking so good as it is overlaped for Region SA, TA, VI

remove the autopct from pie function and pass the following to plt.legend() after plt.title()
`[(i,round(k/region_counts.sum()*100,2)) for i,k in zip(region_counts.index, region_counts)]`

```
plt.figure(figsize=(10, 6))
region_counts = df.groupby('Region')['Count'].sum()
plt.pie(region_counts, labels=region_counts.index, )
plt.title('Percentage of Pixels for Presumed Vegetation Fires by Region')
plt.legend([(i,round(k/region_counts.sum()*100,2)) for i,k in zip(region_counts.index, region_counts)])
plt.axis('equal')
plt.show()
```

---

## TASK 1.6: Let's try to develop a histogram of the mean estimated fire brightness Using Matplotlib to create the histogram

```
df['Mean_estimated_fire_brightness'].head()

count, bin_edges = np.histogram(df['Mean_estimated_fire_brightness'])
print(count) # frequency count
print(bin_edges) # bin ranges, default = 10 bins

# 'bin_edges' is a list of bin intervals
count, bin_edges = np.histogram(df['Mean_estimated_fire_brightness'])

df['Mean_estimated_fire_brightness'].plot(kind='hist', figsize=(8, 5), xticks=bin_edges)

plt.title('Histogram of mean estimated fire brightness') # add a title to the histogram
plt.ylabel('Count') # add y-label
plt.xlabel('Mean estimated fire brightness') # add x-label

plt.show()

plt.figure(figsize=(10, 6))
plt.hist(x=df['Mean_estimated_fire_brightness'], bins=20)
plt.xlabel('Mean Estimated Fire Brightness (Kelvin)')
plt.ylabel('Count')
plt.title('Histogram of Mean Estimated Fire Brightness')
plt.show()
```

TASK 1.7: What if we need to understand the distribution of estimated fire brightness across regions? Let's use the functionality of seaborn and pass region as hue

```
# let's quickly view the dataset
sns.histplot(data=df, x='Mean_estimated_fire_brightness',
hue='Region')
plt.show()
```

looks better!, now include the parameter `multiple='stack'` in the histplot() and see the difference. Include labels and titles as well

```
sns.histplot(data=df, x='Mean_estimated_fire_brightness',
hue='Region', multiple='stack')
plt.title('Histogram of Mean Estimated Fire Brightness by region')
plt.ylabel('Count')
plt.xlabel('Mean estimated fire brightness (Kelvin)')
plt.show()
```

---

TASK 1.8: Let's try to find if there is any correlation between mean estimated fire radiative power and mean confidence level?

```
x = df['Mean_confidence']
y = df['Mean_estimated_fire_radiative_power']
fit = np.polyfit(x, y, deg=1)
fit

df.plot(kind='scatter', x='Mean_confidence',
y='Mean_estimated_fire_radiative_power', figsize=(10, 6),
color='blue')

plt.title('Mean_estimated_fire_radiative_power and mean confidence')
plt.xlabel('Mean_estimated_fire_radiative_power')
plt.ylabel('Mean_confidence')

# plot line of best fit
plt.plot(x, fit[0] * x + fit[1], color='red') # recall that x is the
Years
plt.annotate('y={0:.0f} x + {1:.0f}'.format(fit[0], fit[1]), xy=(2000,
150000))

plt.show()

plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='Mean_confidence',
y='Mean_estimated_fire_radiative_power')
plt.xlabel('Mean Estimated Fire Radiative Power (MW)')
```

```
plt.ylabel('Mean Confidence')
plt.title('Mean Estimated Fire Radiative Power vs. Mean Confidence')
plt.show()
```

---

## TASK 1.9: Let's mark these seven regions on the Map of Australia using Folium

we have created a dataframe for you containing the regions, their latitudes and longitudes. For australia use [-25, 135] as location to create the map

```
region_data = {'region':['NSW','QL','SA','TA','VI','WA','NT'], 'Lat':
[-31.8759835,-22.1646782,-30.5343665,-42.035067,-36.5986096,-
25.2303005,-19.491411],
               'Lon':
[147.2869493,144.5844903,135.6301212,146.6366887,144.6780052,121.01872
46,132.550964]}
reg=pd.DataFrame(region_data)
reg

# define the world map centered around Australia with a low zoom level
Australia_map = folium.Map(location=[-25, 135], zoom_start=4)

# display Australia map
Australia_map

# instantiate a feature group for the incidents in the dataframe
incidents = folium.map.FeatureGroup()

# loop through the 100 crimes and add each to the incidents feature
group
for lat, lng, in zip(reg['Lat'], reg['Lon']):
    incidents.add_child(
        folium.vector_layers.CircleMarker(
            [lat, lng],
            radius=5,
            color='yellow',
            fill=True,
            fill_color='blue',
            fill_opacity=0.6
        )
    )

# add incidents to map
Australia_map.add_child(incidents)

# instantiate a feature group
aus_reg = folium.map.FeatureGroup()
```

```
# Create a Folium map centered on Australia
Aus_map = folium.Map(location=[-25, 135], zoom_start=4)

# loop through the region and add to feature group
for lat, lng, lab in zip(reg.Lat, reg.Lon, reg.region):
    aus_reg.add_child(
        folium.features.CircleMarker(
            [lat, lng],
            popup=lab,
            radius=5, # define how big you want the circle markers to
be
            color='red',
            fill=True,
            fill_color='blue',
            fill_opacity=0.6
        )
    )

# add incidents to map
Aus_map.add_child(aus_reg)
```

---

# Congratulations! You have completed the lab

## Authors

Dr. Pooja

## Change Log

| Date (YYYY-MM-DD) | Version | Changed By | Change Description |
|---|---|---|---|
| 2023-06-281 | 0.2 | Dr. Pooja | Initial Lab Creation |
| 2023-05-01 | 0.1 | Shengkai | Create Lab Template |