Practice Assignment - Part 1: Analyzing wildfire activities in Australia

Estimated time needed: **40** minutes

Table of Contents

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.

In [1]:
```
# All Libraries required for this lab are listed below. The libraries pre-installed on :
#%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"
```

In [2]:
```
%pip install seaborn
%pip install folium
```

Importing Required Libraries

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

In [4]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import folium
%matplotlib inline
```

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

In [5]:
```
from js import fetch
import io

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

```
Data read into a pandas dataframe!
```

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

In [6]:
```
df.head()
```

Out[6]:

	Region	Date	Estimated_fire_area	Mean_estimated_fire_brightness	Mean_estimated_fire_radiative_p
0	NSW	1/4/2005	8.68000	312.266667	42.40
1	NSW	1/5/2005	16.61125	322.475000	62.36
2	NSW	1/6/2005	5.52000	325.266667	38.40
3	NSW	1/7/2005	6.26400	313.870000	33.80
4	NSW	1/8/2005	5.40000	337.383333	122.53

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

In [7]:
```
#Column names
df.columns
```

Out[7]:
```
Index(['Region', 'Date', 'Estimated_fire_area',
       'Mean_estimated_fire_brightness', 'Mean_estimated_fire_radiative_power',
       'Mean_confidence', 'Std_confidence', 'Var_confidence', 'Count',
       'Replaced'],
      dtype='object')
```

In [8]:
```
#data type
df.dtypes
```

Out[8]:
```
Region                                 object
Date                                   object
Estimated_fire_area                   float64
Mean_estimated_fire_brightness        float64
Mean_estimated_fire_radiative_power   float64
```

```
Mean_confidence                        float64
Std_confidence                         float64
Var_confidence                         float64
Count                                    int64
Replaced                                object
dtype: object
```

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

In [9]:
```
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

In [10]:
```
df.dtypes
```

Out[10]:
```
Region                                  object
Date                                    object
Estimated_fire_area                    float64
Mean_estimated_fire_brightness         float64
Mean_estimated_fire_radiative_power    float64
Mean_confidence                        float64
Std_confidence                         float64
Var_confidence                         float64
Count                                    int64
Replaced                                object
Year                                     int32
Month                                    int32
dtype: object
```

▶ Click here for Solution

Practice Tasks

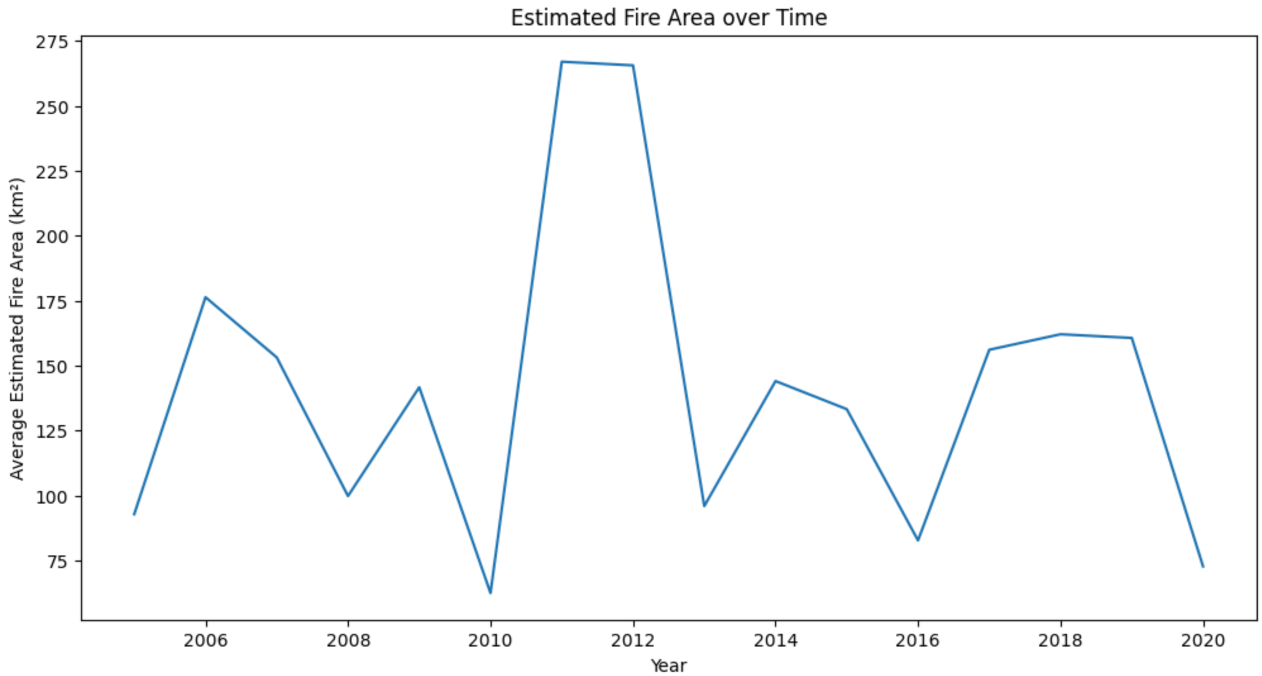
TASK 1.1: Let's try to understand the change in average estimated fire area over time (use pandas to plot)

▶ Click here for a Hint

In [11]:
```
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()
```



▶ Click here for Solution

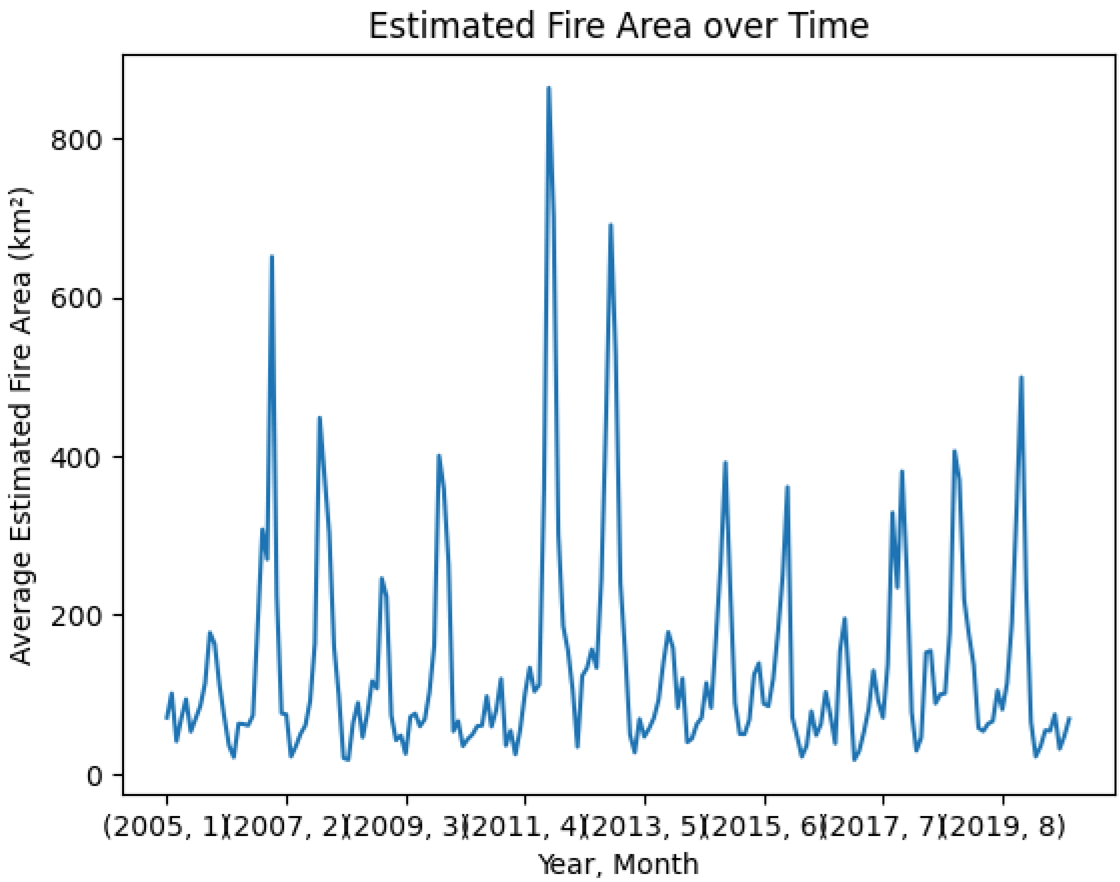
TASK 1.2: You can notice the peak in the plot between 2010 to 2013. Let's narrow down our finding, by plotting the estimated fire area for year grouped together with month.

▶ Click here for a Hint

In [12]:

```
df_new = df.groupby(['Year','Month'])['Estimated_fire_area'].mean()

df_new.plot(x=df_new.index, y=df_new.values)
plt.xlabel('Year, Month')
plt.ylabel('Average Estimated Fire Area (km²)')
plt.title('Estimated Fire Area over Time')
plt.show()
```



▶ Click here for Solution

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)

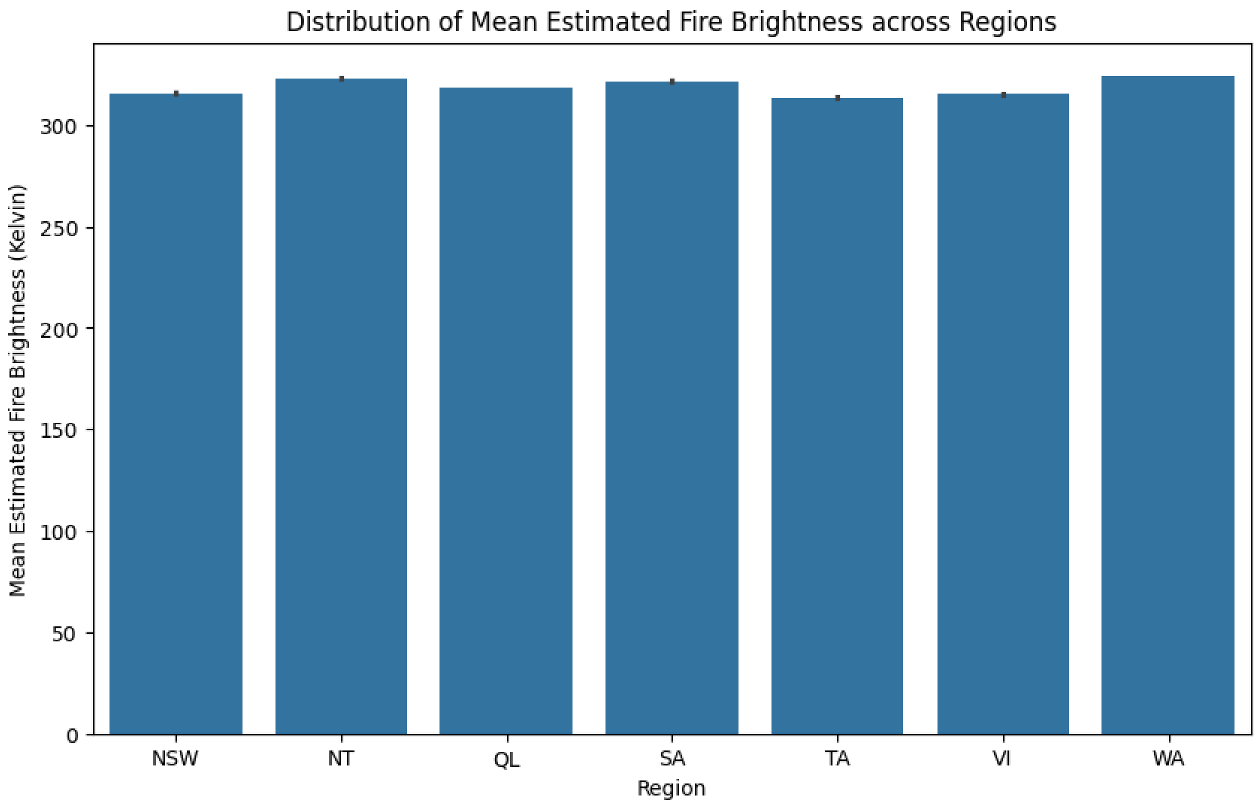
In [13]:
```
df['Region'].unique()
```

Out[13]:
```
array(['NSW', 'NT', 'QL', 'SA', 'TA', 'VI', 'WA'], dtype=object)
```

▶ Click here for a Hint

In [14]:
```
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()
```



▶ Click here for Solution

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

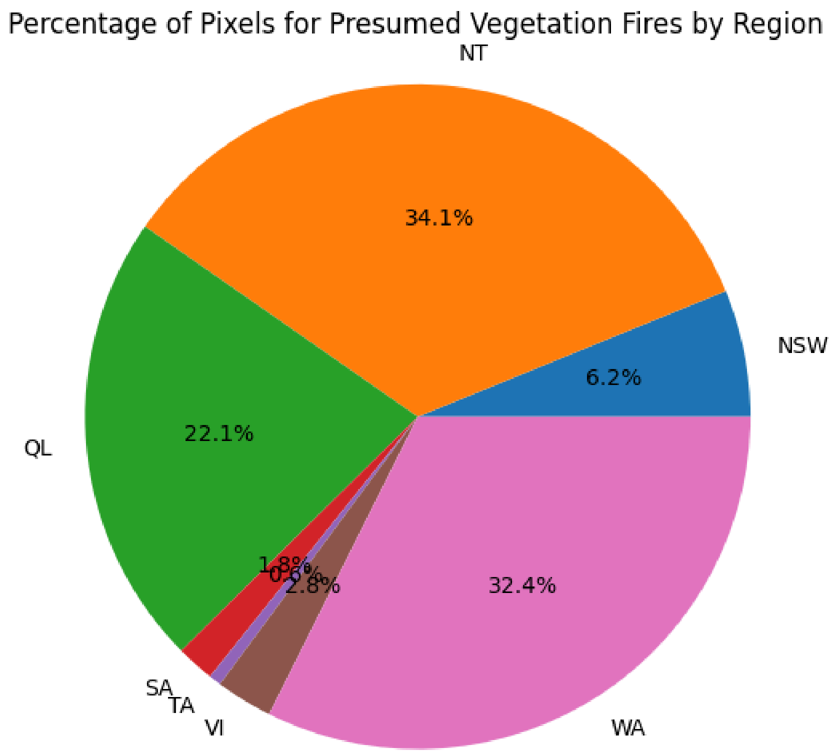
▶ Click here for a Hint

In [15]:

```
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()
```



▶ Click here for Solution

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

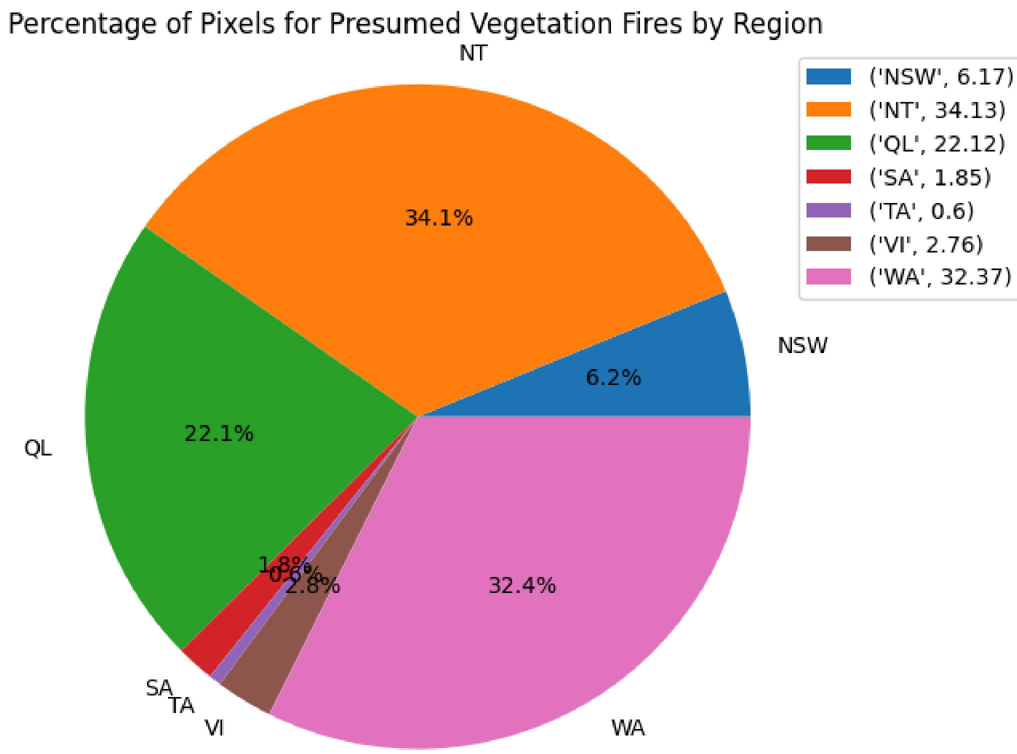
remove the autopct fromm pie function and pass the following to plt.legend() after plt.title()
`[(i,round(k/region_counts.sum()*100,2)) for i,k in zip(region_counts.index, region_counts)]`

In [16]:

```
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.legend([(i,round(k/region_counts.sum()*100,2)) for i,k in zip(region_counts.index,
plt.axis('equal')
plt.show()
```

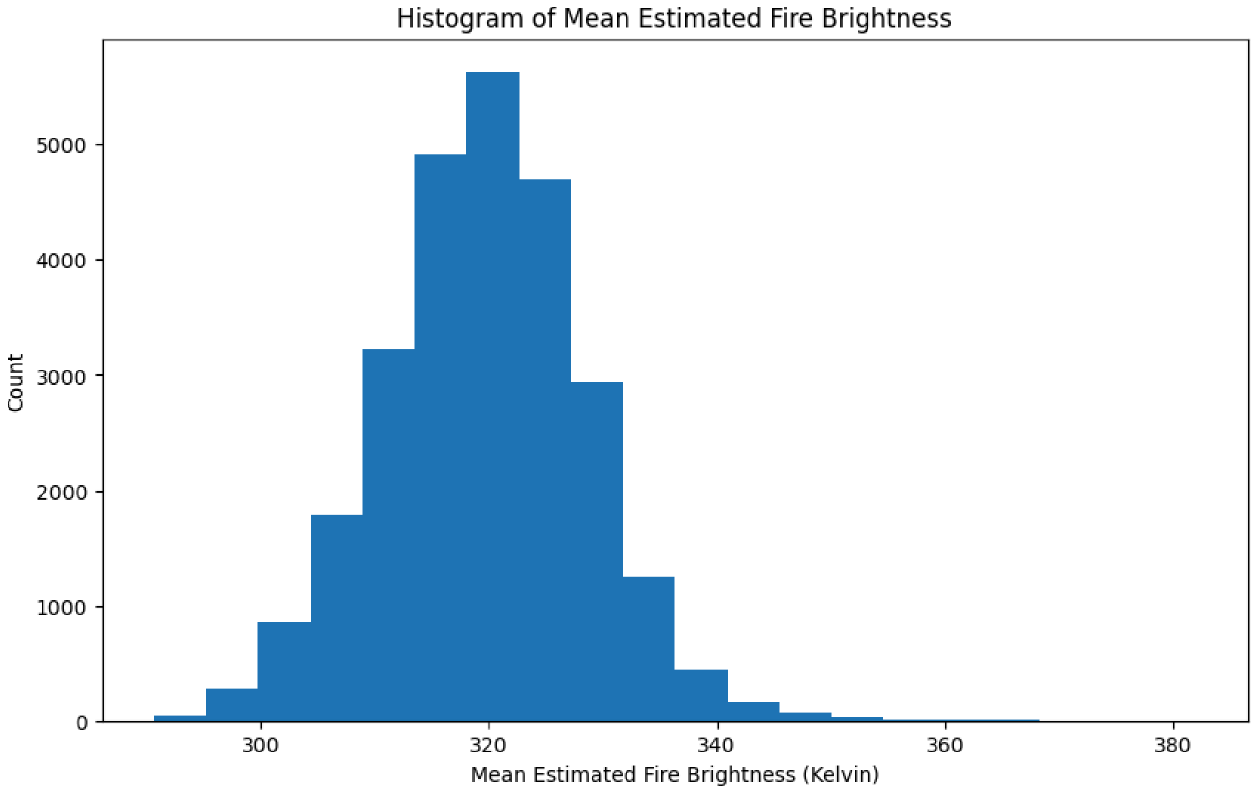


TASK 1.6: Let's try to develop a histogram of the mean estimated fire brightness
Using Matplotlib to create the histogram

▶ Click here for a Hint

In [17]:
```
plt.figure(figsize=(10, 6))

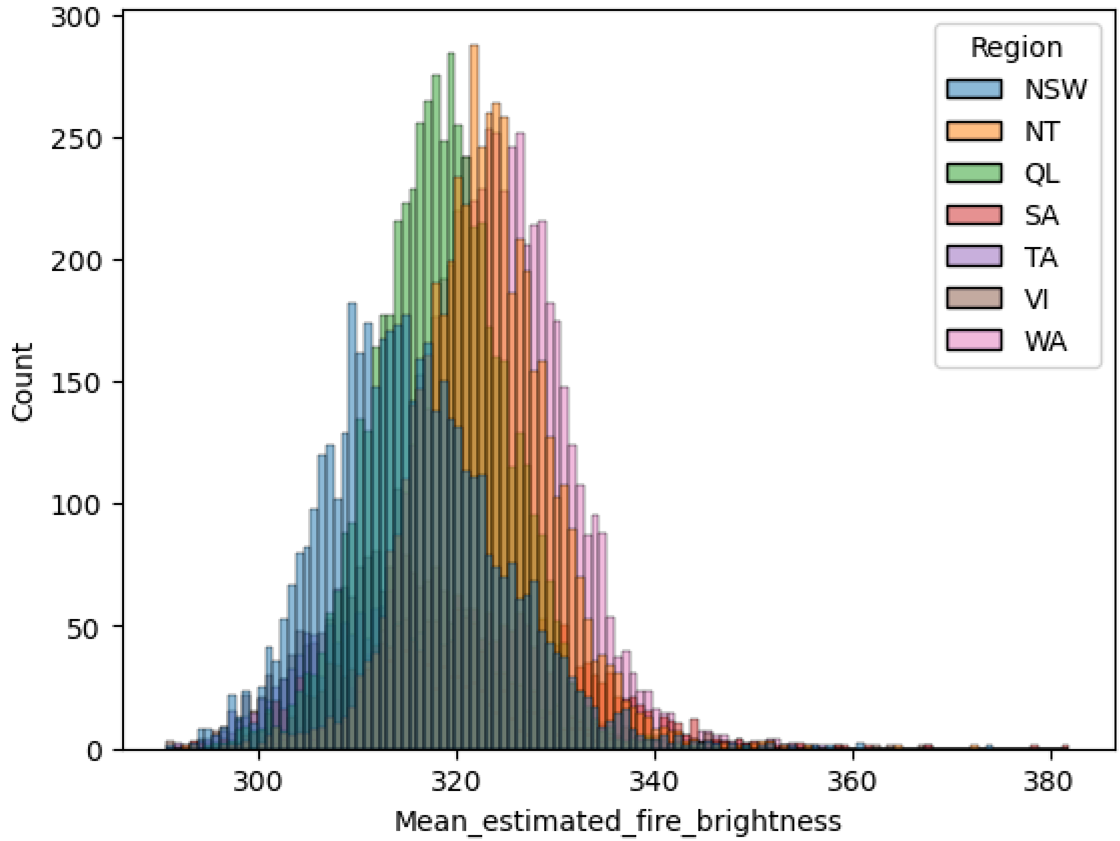
plt.hist(x=df['Mean_estimated_fire_brightness'], bins=20)
plt.xlabel('Mean Estimated Fire Brightness (Kelvin)')
plt.ylabel('Count')
plt.title('Histogram of Mean Estimated Fire Brightness')
plt.show()
```



▶ Click here for Solution

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

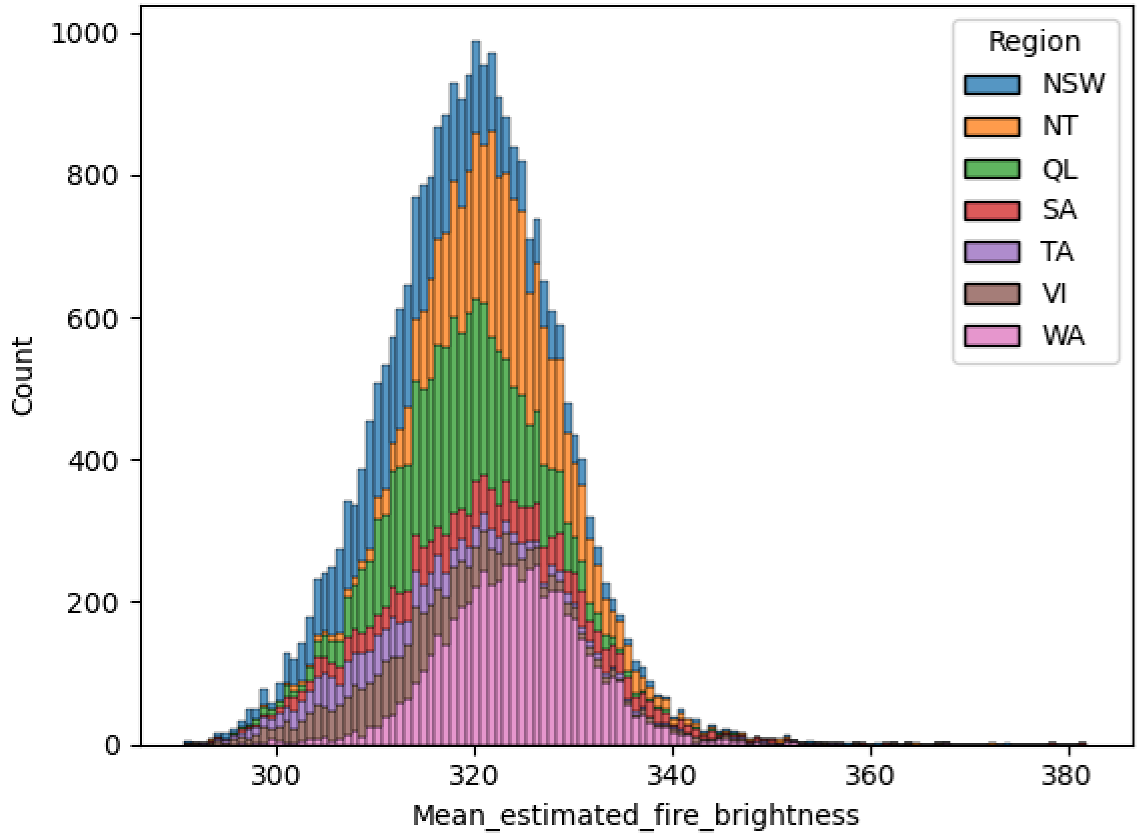
In [18]:
```
sns.histplot(data=df, x='Mean_estimated_fire_brightness', hue='Region')
plt.show()
```



▶ Click here for Solution

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

In [19]:
```
sns.histplot(data=df, x='Mean_estimated_fire_brightness', hue='Region', multiple='stack
plt.show()
```



▶ Click here for Solution

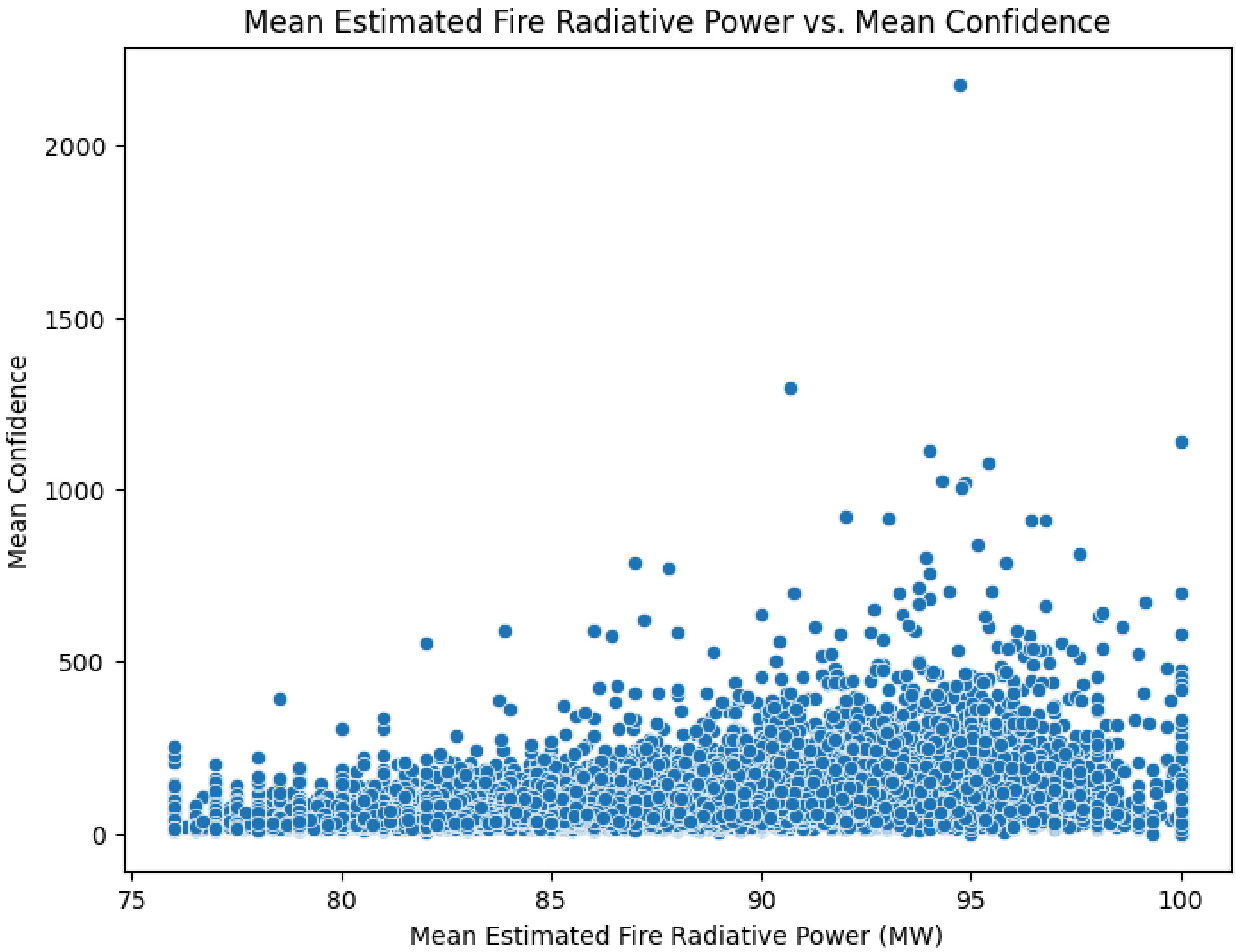
TASK 1.8: Let's try to find if there is any correlation between mean estimated fire radiative power and mean confidence level?

▶ Click here for a Hint

In [20]:

```
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()
```



▶ Click here for Solution

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

In [21]:
```
region_data = {'region':['NSW','QL','SA','TA','VI','WA','NT'], 'Lat':[-31.8759835,-22.1
               'Lon':[147.2869493,144.5844903,135.6301212,146.6366887,144.6780052,121.0
reg=pd.DataFrame(region_data)
reg
```

Out[21]:

	region	Lat	Lon
0	NSW	-31.875984	147.286949
1	QL	-22.164678	144.584490
2	SA	-30.534367	135.630121
3	TA	-42.035067	146.636689
4	VI	-36.598610	144.678005

	region	Lat	Lon
5	WA	-25.230300	121.018725
6	NT	-19.491411	132.550964

In [22]:

```
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)
```

Out[22]:

▶ Click here for Solution

Congratulations! You have completed the lab

Authors

Dr. Pooja

`{toggle}## Change Log`

`{toggle}|Date (YYYY-MM-DD)|Version|Changed By|Change Description|

```
```{toggle}|-|-|-|-|
```

`{toggle}|2023-06-281|0.2|Dr. Pooja|Initial Lab Creation|

```
```{toggle}|2023-05-01|0.1|Shengkai|Create Lab Template|
```