Universidad Autónoma de Baja California

Facultad de Ciencia

Licenciatura de Ciencia de Datos

Proyecto final

Programación para Ciencia de Datos

Emilio Daniel Garcia Perez

372771

Ensenada Baja California. 06 de Junio del 2024

Incendios Forestales 2015 al 2023

Pregunta: ¿Cuáles son las tendencias espaciales y temporales de los incendios forestales en México?

Paso 1: Cargar la base de datos en el entorno

En este paso, el objetivo es cargar los datos de incendios forestales desde un **archivo CSV.** Se utiliza la función cargar datos, la cual toma como entrada la ruta del archivo CSV y devuelve un DataFrame de pandas que contiene los datos.

Dentro de la función cargar datos, se utiliza la función **pd.read_csv** de la librería Pandas para leer el archivo CSV especificado por la ruta. Se especifica la codificación **encoding='iso-8859-1'** para manejar posibles caracteres especiales en el archivo CSV.

Luego de cargar los datos, se imprime información básica sobre el Data Frame utilizando el método **info()**. Esto incluye el número de filas y columnas, los nombres de las columnas, y los tipos de datos de cada columna.

Después, se muestran las primeras y últimas filas del Data Frame utilizando los métodos **head()** y **tail()** respectivamente. Esto proporciona una vista previa de los datos y ayuda a verificar si la carga se realizó correctamente.

El propósito de este paso es familiarizarse con los datos y asegurarse de que se hayan cargado correctamente antes de continuar con el análisis y la visualización.

Código que se describió previamente:

```
[177] #Importar la librerias
      import pandas as pd
      import matplotlib.pyplot as plt
      import seaborn as sns
      #Funcion para la carga de datos
      def cargar_datos(ruta_archivo):
          """
          Carga los datos de incendios desde un archivo CSV.

          Parámetros:
          ruta_archivo (str): Ruta del archivo CSV.

          Returns:
          DataFrame: DataFrame con los datos de incendios.
          """
          #Cargar los datos
          fire_df = pd.read_csv('/content/drive/MyDrive/Colab_EDyA/incendios.csv', encoding='iso-8859-1', dtype=str)

          #Imprimir el tipo de objeto
          print("Tipo de objeto:", type(fire_df))
          print("----------------------------------------------------------------")

          #Mostrar información sobre el DataFrame
          print("\nDescibe los dimensiones de los datos de la data frame")
          print(fire_df.info())

          #Mostrar las primeras filas del DataFrame
          print("\nDescribe los 10 primeros datos de la data frame")
          print(fire_df.head(10))

          #Mostrar las últimas filas del DataFrame
          print("\nDescribe los 10 ultimos datos de la data frame")
          print(fire_df.tail(10))

          return fire_df
      data_frame = 'incendios.csv'
      fire_df = cargar_datos(data_frame)
```

Resultados:

- Se obtiene un Data Frame que contiene los datos de incendios forestales.
- Se imprime información básica sobre el DataFrame, incluyendo número de filas y columnas, y tipos de datos de las columnas.
- Se muestran las primeras y últimas filas del Data Frame para verificar la carga de los datos.

Paso 2: Limpieza de Datos

La limpieza de datos es un paso crucial en cualquier análisis de datos. En esta etapa, se realizan una serie de operaciones para asegurar que los datos estén en un formato adecuado y sean coherentes para su posterior análisis. En este caso específico, la limpieza de datos se lleva a cabo en la función limpiar datos, que toma como entrada el Data Frame cargado previamente. Dentro de esta función, se realizan las siguientes operaciones:

- Conversión de columnas a Tipos de Datos Numéricos: Se identifican las columnas que deberían representar valores numéricos, pero que están en formato de texto. Utilizando la función **pd.to_numeric**, se convierten estas columnas al tipo de datos numérico adecuado. Esto facilita operaciones matemáticas y análisis estadísticos posteriores.
- Eliminación de columnas irrelevantes: Algunas columnas pueden no ser relevantes para el análisis que se va a realizar. En este caso, se eliminan columnas como 'Clave del incendio', 'Clave Municipio', 'CVE_ENT', 'CVE_MUN', 'CVEGEO' y 'Predio' utilizando el método **drop()**.
- Formato de Fecha y Hora: Las columnas que contienen fechas ('Fecha Inicio' y 'Fecha Termino') se convierten al formato adecuado utilizando **pd.to_datetime**. Esto permite realizar operaciones de tiempo y análisis de series temporales.
- Imputación de Valores Faltantes: Se manejan los valores faltantes en algunas columnas utilizando la imputación de valores promedio. Esto se logra mediante el método **fillna()**.

Después de realizar estas operaciones, se imprime información adicional sobre el Data Frame para verificar que los cambios se hayan aplicado correctamente. Esto incluye una descripción de las columnas numéricas y una descripción estadística de los datos.

Código que se describió previamente:

```
# Convertir columnas numéricas a tipo float
Columnas_numericas = ['latitud_grados', 'latitud_minutos', 'latitud_segundos', 'longitud_grados',
                      'longitud_minutos', 'longitud_segundos', 'Duración días', 'Arbolado Adulto',
                      'Renuevo', 'Arbustivo', 'Herbáceo', 'Hojarasca', 'Total hectáreas']
for col in Columnas_numericas:
    # Convertir la columna a tipo float, si no es posible, se asigna NaN
    df[col] = pd.to_numeric(df[col], errors='coerce')

# Eliminar columnas irrelevantes para el análisis
df = df.drop(['Clave del incendio', 'Clave Municipio', 'CVE_ENT', 'CVE_MUN', 'CVEGEO', 'Predio'], axis=1)

# Convertir columnas numéricas a tipos de datos numéricos
Columnas_numericas = ['Año', 'Duración días', 'Arbolado Adulto', 'Renuevo', 'Arbustivo', 'Herbáceo', 'Hojarasca', 'Total hectáreas']
for col in Columnas_numericas:
    # Convertir la columna a tipo numérico, si no es posible, se asigna NaN
    df[col] = pd.to_numeric(df[col], errors='coerce')

# Especificar el formato de fecha y hora para las columnas de fecha
df['Fecha Inicio'] = pd.to_datetime(df['Fecha Inicio'], format='%d/%m/%Y', errors='coerce')
df['Fecha Termino'] = pd.to_datetime(df['Fecha Termino'], format='%d/%m/%Y', errors='coerce')

# Convertir columnas de latitud y longitud a tipo float
df['Latitud'] = pd.to_numeric(df['Latitud'], errors='coerce')
df['Longitud'] = pd.to_numeric(df['Longitud'], errors='coerce')

# Imputar valores faltantes usando la media de cada columna
for col in ['Duración días', 'Arbolado Adulto', 'Renuevo', 'Arbustivo', 'Herbáceo', 'Hojarasca']:
    # Llenar los valores faltantes de la columna con la media de esa columna
    df[col] = df[col].fillna(df[col].mean())

# Mostrar información básica del DataFrame después de la conversión e imputación
print("\nInformación después de la conversión e imputación de valores faltantes:")
print(df.info())

# Mostrar descripción estadística de las columnas numéricas
print("\nDescripción estadística de las columnas numéricas:")
print(df.describe())
```

Resultados:

- Se convierten las columnas relevantes a tipos de datos numéricos.
- Se eliminan columnas irrelevantes.
- Se formatean las columnas de fecha y hora.
- Se manejan los valores faltantes mediante imputación de valores promedio.

Paso 3: Ejecuciones de gráficas

En esta parte se crearon diferentes gráficas para visualizar de diferentes maneras al Data Frame para tener un mejor entendimiento.

1. Número de Incendios por Estado: Esta función genera un gráfico de barras que muestra la distribución del número de incendios en diferentes estados. Utiliza la biblioteca Seaborn para crear un gráfico de conteo, donde cada barra representa un estado y su altura corresponde al número de incendios registrados en ese estado. También permite visualizar si hay algún estado con una incidencia notablemente mayor o menor de incendios en comparación con otros.

```python
def grafico_incendios_por_estado(fire_df):
    """
    Esta función genera un gráfico de barras que muestra el número de incendios
    por estado utilizando los datos proporcionados en el DataFrame fire_df.

    Parámetros:
    fire_df (DataFrame): DataFrame con los datos de incendios.
    """
    #Crear una nueva figura para el gráfico con el tamaño especificado
    plt.figure(figsize=(15, 8))

    #Crear un gráfico de barras contando el número de incendios por estado
    #sns.countplot es una función de seaborn que genera un gráfico de conteo de los valores de una columna categórica
    sns.countplot(data=fire_df, y='Estado', order=fire_df['Estado'].value_counts().index, hue='Estado', legend=False)

    #Agregar título al gráfico
    plt.title('Número de Incendios por Estado', fontsize=16)

    #Etiquetas de los ejes x e y
    plt.xlabel('Número de Incendios', fontsize=14)
    plt.ylabel('Estado', fontsize=14)

    #Tamaño de las etiquetas en los ejes x e y
    #fontsize especifica el tamaño de la fuente para los ejes x e y
    plt.xticks(fontsize=12)
    plt.yticks(fontsize=12)

    #Agregar una descripción debajo del gráfico
    plt.text(0.5, -0.15, "Este gráfico muestra la distribución del número de incendios por estado.", ha='center', fontsize=12, transform=plt.gca().transAxes)

    #Mostrar el gráfico
    plt.show()

    #Cerrar la figura para liberar memoria
    plt.close()
```

Función: `grafico_incendios_por_estado(df)`

Comandos y funciones utilizados:

- `plt.figure(figsize=(15, 8))`: Crea una figura de matplotlib con un tamaño específico.
- `sns.countplot()`: Crea un gráfico de conteo utilizando Seaborn.
- `plt.title()`, `plt.xlabel()`, `plt.ylabel()`, `plt.xticks()`, `plt.yticks()`: Agrega títulos y etiquetas a los ejes del gráfico.
- `plt.text()`: Añade un texto explicativo en el gráfico.
- `plt.show()`, `plt.close()`: Muestra y cierra el gráfico respectivamente.

2. Número de Incendios a lo Largo del Tiempo (por Año): Esta función crea un gráfico de líneas que muestra la evolución del número de incendios a lo largo de los años. Agrupa los datos por año y cuenta el número de incendios ocurridos en cada año. Luego, traza estos datos en un gráfico de líneas para visualizar las tendencias a lo largo del tiempo.

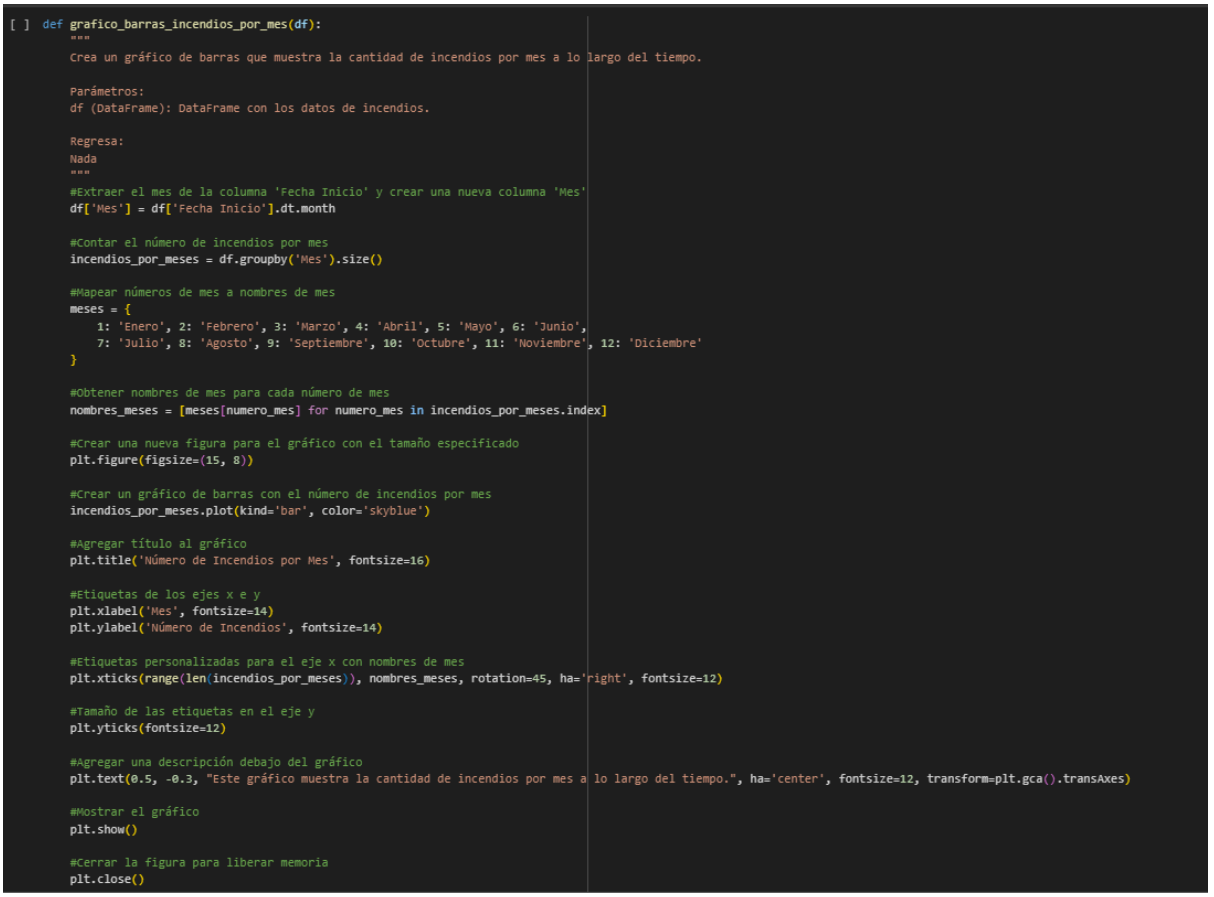
```python
def grafico_lineas_incendios_por_año(df):
    """
    Esta función crea un gráfico de líneas que muestra la cantidad de incendios a lo largo del tiempo utilizando los datos proporcionados en el DataFrame.

    Parámetros:
    df (DataFrame): DataFrame con los datos de incendios.

    Regresa:
    Nada
    """
    #Crear una nueva figura para el gráfico con el tamaño especificado
    plt.figure(figsize=(15, 8))

    #Convertir la columna 'Año' a formato de fecha
    df['Año'] = pd.to_datetime(df['Año'], format='%Y')

    #Agrupar los datos por año y contar el número de incendios por año
    incendios_por_años = df.groupby('Año').size()

    #Crear un gráfico de líneas con los datos de incendios por año
    #kind='line' especifica que se debe generar un gráfico de líneas
    #marker='o' agrega puntos circulares en cada punto de datos en el gráfico de líneas
    #color='green' especifica el color de las líneas en el gráfico
    incendios_por_años.plot(kind='line', marker='o', color='green')

    #Agregar título al gráfico
    plt.title('Número de Incendios a lo Largo del Tiempo', fontsize=16)

    #Etiquetas de los ejes x e y
    plt.xlabel('Año', fontsize=14)
    plt.ylabel('Número de Incendios', fontsize=14)

    #Tamaño de las etiquetas en los ejes x e y
    plt.xticks(fontsize=12)
    plt.yticks(fontsize=12)

    #Habilitar la cuadrícula en el gráfico
    plt.grid(True)

    #Agregar una descripción debajo del gráfico
    plt.text(0.5, -0.15, "Este gráfico muestra la cantidad de incendios a lo largo del tiempo en forma de serie temporal.", ha='center', fontsize=12, transform=plt.gca().transAxes)

    #Mostrar el gráfico sin bloquear la ejecución del código
    plt.show(block=False)

    #Cerrar la figura para liberar memoria
    plt.close()
```

Función: `grafico_lineas_incendios_por_año(df)`

Comandos y funciones utilizados:

- `plt.figure(figsize=(15, 8))`: Crea una figura de matplotlib con un tamaño específico.
- `pd.to_datetime()`: Convierte la columna de año a formato de fecha.
- `groupby().size()`: Agrupa los datos por año y cuenta el número de incendios en cada año.
- `plot(kind='line', marker='o')`: Crea un gráfico de líneas con marcadores.
- `plt.grid()`: Agrega una cuadrícula al gráfico.
- `plt.show(block=False)`: Muestra el gráfico sin bloquear la ejecución del código.
- `plt.close()`: Cierra el gráfico después de mostrarlo.

3. Número de Incendios por Mes: Genera un gráfico de barras que muestra la distribución del número de incendios por mes a lo largo del tiempo. Utiliza la función **groupby** de pandas para agrupar los datos por mes y cuenta el número de incendios en cada mes. Luego, traza estos datos en un gráfico de barras para identificar si hay meses específicos con una mayor incidencia de incendios.



```
[ ]  def grafico_barras_incendios_por_mes(df):
        """
        Crea un gráfico de barras que muestra la cantidad de incendios por mes a lo largo del tiempo.

        Parámetros:
        df (DataFrame): DataFrame con los datos de incendios.

        Regresa:
        Nada
        """
        #Extraer el mes de la columna 'Fecha Inicio' y crear una nueva columna 'Mes'
        df['Mes'] = df['Fecha Inicio'].dt.month

        #Contar el número de incendios por mes
        incendios_por_meses = df.groupby('Mes').size()

        #Mapear números de mes a nombres de mes
        meses = {
            1: 'Enero', 2: 'Febrero', 3: 'Marzo', 4: 'Abril', 5: 'Mayo', 6: 'Junio',
            7: 'Julio', 8: 'Agosto', 9: 'Septiembre', 10: 'Octubre', 11: 'Noviembre', 12: 'Diciembre'
        }

        #Obtener nombres de mes para cada número de mes
        nombres_meses = [meses[numero_mes] for numero_mes in incendios_por_meses.index]

        #Crear una nueva figura para el gráfico con el tamaño especificado
        plt.figure(figsize=(15, 8))

        #Crear un gráfico de barras con el número de incendios por mes
        incendios_por_meses.plot(kind='bar', color='skyblue')

        #Agregar título al gráfico
        plt.title('Número de Incendios por Mes', fontsize=16)

        #Etiquetas de los ejes x e y
        plt.xlabel('Mes', fontsize=14)
        plt.ylabel('Número de Incendios', fontsize=14)

        #Etiquetas personalizadas para el eje x con nombres de mes
        plt.xticks(range(len(incendios_por_meses)), nombres_meses, rotation=45, ha='right', fontsize=12)

        #Tamaño de las etiquetas en el eje y
        plt.yticks(fontsize=12)

        #Agregar una descripción debajo del gráfico
        plt.text(0.5, -0.3, "Este gráfico muestra la cantidad de incendios por mes a lo largo del tiempo.", ha='center', fontsize=12, transform=plt.gca().transAxes)

        #Mostrar el gráfico
        plt.show()

        #Cerrar la figura para liberar memoria
        plt.close()
```

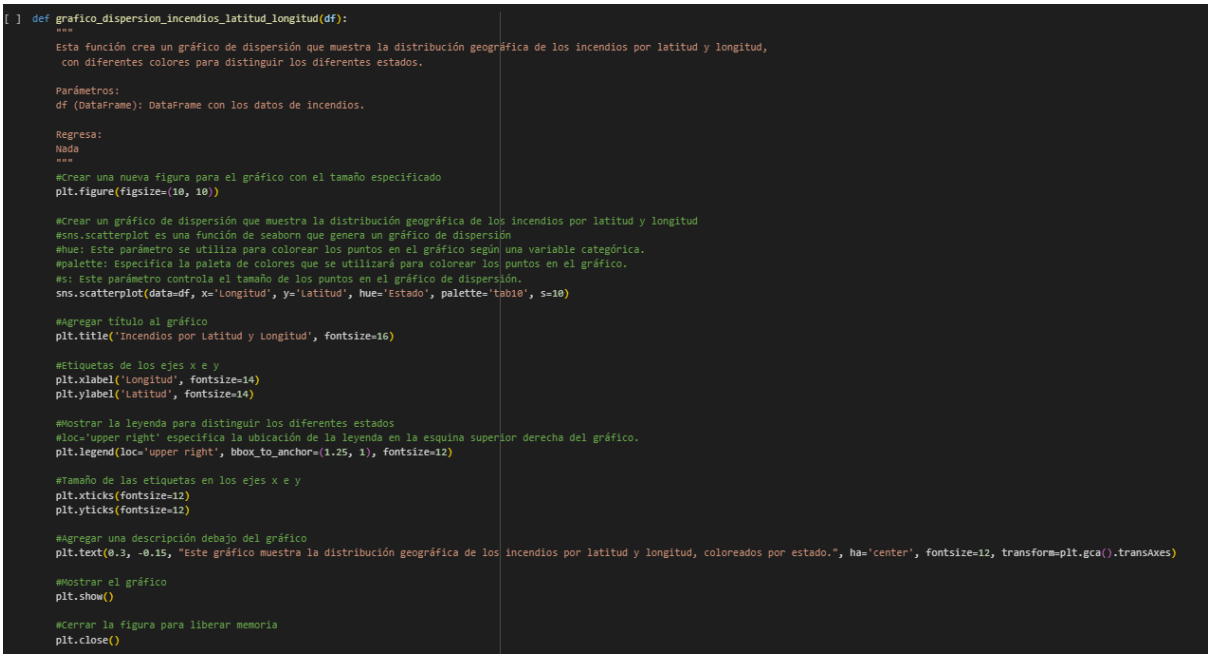
Función: `grafico_barras_incendios_por_mes(df)`

Comandos y funciones utilizados:

- `groupby().size()`: Agrupa los datos por mes y cuenta el número de incendios en cada mes.
- `plot(kind='bar')`: Crea un gráfico de barras.
- `plt.xticks()`, `plt.yticks()`: Ajusta las etiquetas de los ejes x e y.
- `plt.text()`: Añade un texto explicativo en el gráfico.

- `plt.show()`, `plt.close()`: Muestra y cierra el gráfico respectivamente.

4. Distribución Geográfica de los Incendios: Esta función crea un gráfico de dispersión que muestra la distribución geográfica de los incendios por latitud y longitud. Utiliza la biblioteca Seaborn para trazar los datos en un gráfico de dispersión, donde cada punto representa un incendio y su posición está determinada por sus coordenadas geográficas. Los puntos están coloreados según el estado en el que ocurrió el incendio.



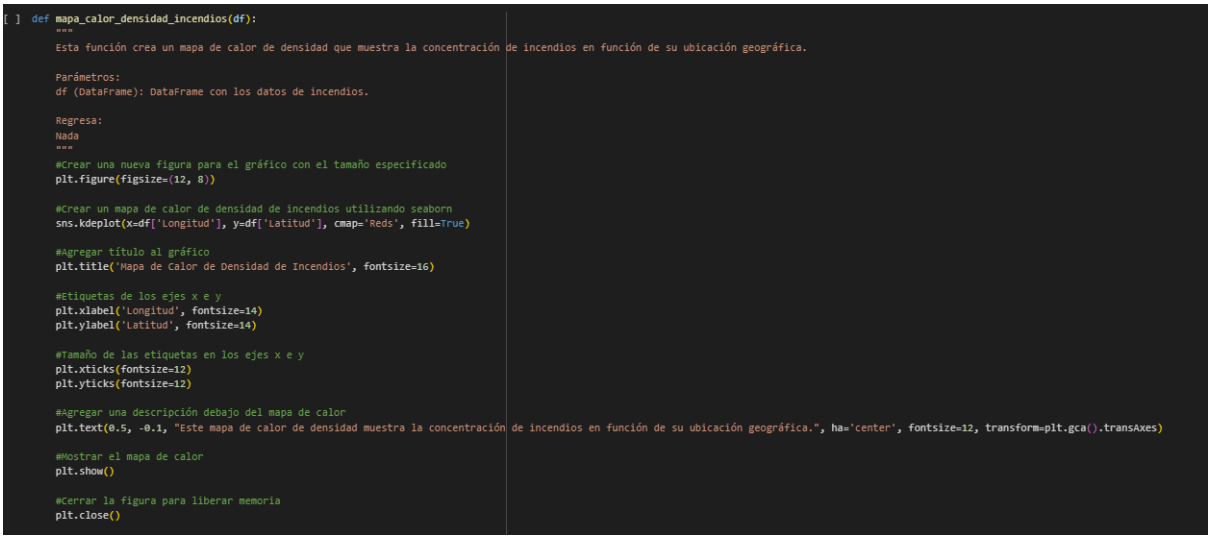
```
[ ] def grafico_dispersion_incendios_latitud_longitud(df):
        """
        Esta función crea un gráfico de dispersión que muestra la distribución geográfica de los incendios por latitud y longitud,
         con diferentes colores para distinguir los diferentes estados.

        Parámetros:
        df (DataFrame): DataFrame con los datos de incendios.

        Regresa:
        Nada
        """
        #Crear una nueva figura para el gráfico con el tamaño especificado
        plt.figure(figsize=(10, 10))

        #Crear un gráfico de dispersión que muestra la distribución geográfica de los incendios por latitud y longitud
        #sns.scatterplot es una función de seaborn que genera un gráfico de dispersión
        #hue: Este parámetro se utiliza para colorear los puntos en el gráfico según una variable categórica.
        #palette: Especifica la paleta de colores que se utilizará para colorear los puntos en el gráfico.
        #s: Este parámetro controla el tamaño de los puntos en el gráfico de dispersión.
        sns.scatterplot(data=df, x='Longitud', y='Latitud', hue='Estado', palette='tab10', s=10)

        #Agregar título al gráfico
        plt.title('Incendios por Latitud y Longitud', fontsize=16)

        #Etiquetas de los ejes x e y
        plt.xlabel('Longitud', fontsize=14)
        plt.ylabel('Latitud', fontsize=14)

        #Mostrar la leyenda para distinguir los diferentes estados
        #loc='upper right' especifica la ubicación de la leyenda en la esquina superior derecha del gráfico.
        plt.legend(loc='upper right', bbox_to_anchor=(1.25, 1), fontsize=12)

        #Tamaño de las etiquetas en los ejes x e y
        plt.xticks(fontsize=12)
        plt.yticks(fontsize=12)

        #Agregar una descripción debajo del gráfico
        plt.text(0.3, -0.15, "Este gráfico muestra la distribución geográfica de los incendios por latitud y longitud, coloreados por estado.", ha='center', fontsize=12, transform=plt.gca().transAxes)

        #Mostrar el gráfico
        plt.show()

        #Cerrar la figura para liberar memoria
        plt.close()
```

Función: `grafico_dispersion_incendios_latitud_longitud(df)`

Comandos y funciones utilizados:

- `plt.figure(figsize=(10, 10))`: Crea una figura de matplotlib con un tamaño específico.
- `sns.scatterplot()`: Crea un gráfico de dispersión utilizando Seaborn.
- `plt.title()`, `plt.xlabel()`, `plt.ylabel()`, `plt.legend()`, `plt.xticks()`, `plt.yticks()`: Agrega títulos, etiquetas y leyendas al gráfico.
- `plt.text()`: Añade un texto explicativo en el gráfico.
- `plt.show()`, `plt.close()`: Muestra y cierra el gráfico respectivamente.

5. Mapa de calor Geográfica de los Incendios: Genera un mapa de calor de densidad que muestra la concentración de incendios en función de su ubicación geográfica. Utiliza la función **kdeplot** de Seaborn para crear un mapa de calor, donde las áreas con colores más intensos indican una mayor concentración de incendios. Esta visualización permite identificar áreas con alta densidad de incendios.



```
[ ] def mapa_calor_densidad_incendios(df):
        """
        Esta función crea un mapa de calor de densidad que muestra la concentración de incendios en función de su ubicación geográfica.

        Parámetros:
        df (DataFrame): DataFrame con los datos de incendios.

        Regresa:
        Nada
        """
        #Crear una nueva figura para el gráfico con el tamaño especificado
        plt.figure(figsize=(12, 8))

        #Crear un mapa de calor de densidad de incendios utilizando seaborn
        sns.kdeplot(x=df['Longitud'], y=df['Latitud'], cmap='Reds', fill=True)

        #Agregar título al gráfico
        plt.title('Mapa de Calor de Densidad de Incendios', fontsize=16)

        #Etiquetas de los ejes x e y
        plt.xlabel('Longitud', fontsize=14)
        plt.ylabel('Latitud', fontsize=14)

        #Tamaño de las etiquetas en los ejes x e y
        plt.xticks(fontsize=12)
        plt.yticks(fontsize=12)

        #Agregar una descripción debajo del mapa de calor
        plt.text(0.5, -0.1, "Este mapa de calor de densidad muestra la concentración de incendios en función de su ubicación geográfica.", ha='center', fontsize=12, transform=plt.gca().transAxes)

        #Mostrar el mapa de calor
        plt.show()

        #Cerrar la figura para liberar memoria
        plt.close()
```

Función: `mapa_calor_densidad_incendios(df)`

Comandos y funciones utilizados:

- `plt.figure(figsize=(12, 8))`: Crea una figura de matplotlib con un tamaño específico.
- `sns.kdeplot()`: Crea un mapa de calor de densidad utilizando Seaborn.
- `plt.title()`, `plt.xlabel()`, `plt.ylabel()`, `plt.xticks()`, `plt.yticks()`: Agrega títulos y etiquetas a los ejes del gráfico.
- `plt.text()`: Añade un texto explicativo en el gráfico.
- `plt.show()`, `plt.close()`: Muestra y cierra el gráfico respectivamente.

6. Porcentaje de incendios por temporada: Crea un gráfico de pastel que muestra el porcentaje de incendios por temporada. Utiliza la función **value_counts** de pandas para contar el número de incendios en cada temporada y luego traza estos datos en un gráfico de pastel para visualizar la distribución estacional de los incendios.

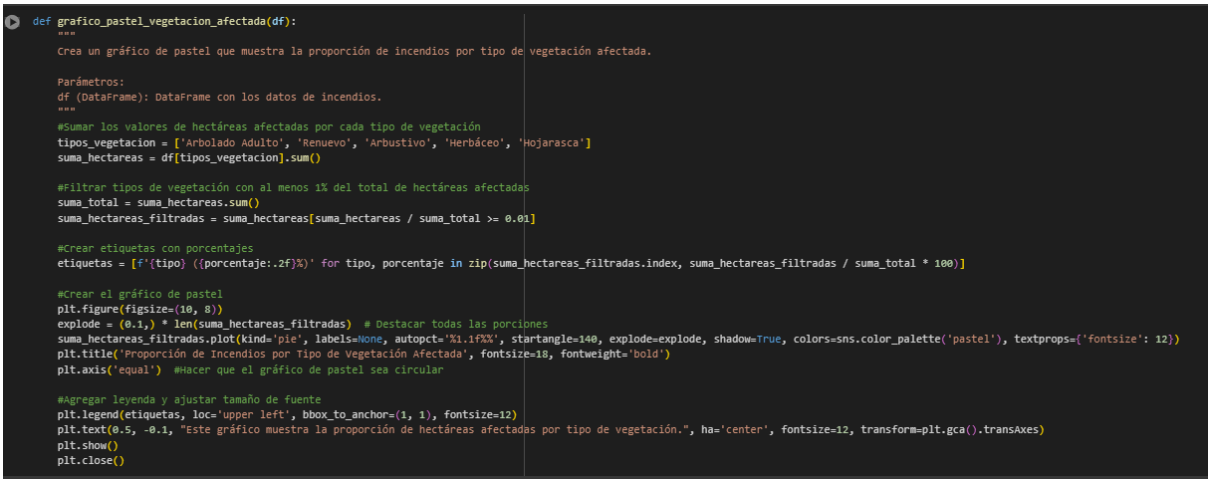
```python
[ ] def grafico_porcentaje_incendios_temporada(df):
        """
        Crea un gráfico de pastel que muestra el porcentaje de incendios por temporada.

        Parámetros:
        df (DataFrame): DataFrame con los datos de incendios.

        Retorna:
        Nada
        """
        #Mapear el mes de inicio de cada incendio a la temporada correspondiente
        df['Temporada'] = df['Fecha Inicio'].dt.month.map({1: 'Invierno', 2: 'Invierno', 3: 'Primavera', 4: 'Primavera',
                                                           5: 'Primavera', 6: 'Verano', 7: 'Verano', 8: 'Verano',
                                                           9: 'Otoño', 10: 'Otoño', 11: 'Otoño', 12: 'Invierno'})

        #Crear una nueva figura para el gráfico con el tamaño especificado
        plt.figure(figsize=(10, 8))

        #Destacar la primera porción (Invierno)
        explode = (0.1, 0, 0, 0)

        #Contar el número de incendios por temporada
        conteo_temporada = df['Temporada'].value_counts()

        #Crear un gráfico de pastel con el porcentaje de incendios por temporada
        conteo_temporada.plot(kind='pie', autopct='%1.1f%%', colors=sns.color_palette('pastel'), labels=None, explode=explode, shadow=True, startangle=140, textprops={'fontsize': 12})

        #Agregar título al gráfico
        plt.title('Porcentaje de Incendios por Temporada', fontsize=18, fontweight='bold')

        #Hacer que el gráfico de pastel sea circular
        plt.axis('equal')

        #Eliminar etiqueta del eje y
        plt.ylabel('')

        #Agregar leyenda y ajustar tamaño de fuente
        plt.legend(conteo_temporada.index, loc='upper left', bbox_to_anchor=(1, 1), fontsize=12)

        #Agregar una descripción debajo del gráfico
        plt.text(0.5, -0.1, "Este gráfico de pastel muestra el porcentaje de incendios por temporada.", ha='center', fontsize=12, transform=plt.gca().transAxes)

        #Mostrar el gráfico de pastel
        plt.show()

        #Cerrar la figura para liberar memoria
        plt.close()
```

Función: `grafico_porcentaje_incendios_temporada(df)`

Comandos y funciones utilizados:

- `groupby().size()`: Agrupa los datos por temporada y cuenta el número de incendios en cada temporada.
- `plot(kind='pie')`: Crea un gráfico de pastel.
- `plt.title()`, `plt.axis()`, `plt.ylabel()`, `plt.legend()`, `plt.text()`: Agrega títulos, leyendas y etiquetas al gráfico.
- `plt.show()`, `plt.close()`: Muestra y cierra el gráfico respectivamente.

7. Porcentaje incendios por tipo de vegetación afectada: Genera un gráfico de pastel que muestra la proporción de incendios por tipo de vegetación afectada. Utiliza los datos sobre la cantidad de hectáreas afectadas por cada tipo de vegetación para calcular el porcentaje de incendios que afectaron a cada tipo. Luego, traza estos datos en un gráfico de pastel para identificar qué tipos de vegetación son más susceptibles a los incendios.



```python
def grafico_pastel_vegetacion_afectada(df):
    """
    Crea un gráfico de pastel que muestra la proporción de incendios por tipo de vegetación afectada.

    Parámetros:
    df (DataFrame): DataFrame con los datos de incendios.
    """
    #Sumar los valores de hectáreas afectadas por cada tipo de vegetación
    tipos_vegetacion = ['Arbolado Adulto', 'Renuevo', 'Arbustivo', 'Herbáceo', 'Hojarasca']
    suma_hectareas = df[tipos_vegetacion].sum()

    #Filtrar tipos de vegetación con al menos 1% del total de hectáreas afectadas
    suma_total = suma_hectareas.sum()
    suma_hectareas_filtradas = suma_hectareas[suma_hectareas / suma_total >= 0.01]

    #Crear etiquetas con porcentajes
    etiquetas = [f'{tipo} ({porcentaje:.2f}%)' for tipo, porcentaje in zip(suma_hectareas_filtradas.index, suma_hectareas_filtradas / suma_total * 100)]

    #Crear el gráfico de pastel
    plt.figure(figsize=(10, 8))
    explode = (0.1,) * len(suma_hectareas_filtradas)  # Destacar todas las porciones
    suma_hectareas_filtradas.plot(kind='pie', labels=None, autopct='%1.1f%%', startangle=140, explode=explode, shadow=True, colors=sns.color_palette('pastel'), textprops={'fontsize': 12})
    plt.title('Proporción de Incendios por Tipo de Vegetación Afectada', fontsize=18, fontweight='bold')
    plt.axis('equal')  #Hacer que el gráfico de pastel sea circular

    #Agregar leyenda y ajustar tamaño de fuente
    plt.legend(etiquetas, loc='upper left', bbox_to_anchor=(1, 1), fontsize=12)
    plt.text(0.5, -0.1, "Este gráfico muestra la proporción de hectáreas afectadas por tipo de vegetación.", ha='center', fontsize=12, transform=plt.gca().transAxes)
    plt.show()
    plt.close()
```

Función: `grafico_pastel_vegetacion_afectada(df)`

Comandos y funciones utilizados:

- `plot(kind='pie')`: Crea un gráfico de pastel.
- `plt.title()`, `plt.axis()`, `plt.legend()`, `plt.text()`: Agrega títulos, leyendas y etiquetas al gráfico.
- `plt.show()`, `plt.close()`: Muestra y cierra el gráfico respectivamente.

8. Comparación de incendios por región: Crea un gráfico de barras que compara el total de hectáreas quemadas por región. Utiliza la función groupby de pandas para agrupar los datos por región y calcular la mediana del total de hectáreas quemadas en cada región. Luego, traza estos datos en un gráfico de barras para identificar las regiones con mayores áreas afectadas por incendios.

```python
[ ] def grafico_comparacion_total_hectareas_regiones(df):
        """
        Crea un gráfico de barras que compara el total de hectáreas quemadas por región.

        Parámetros:
        df (DataFrame): DataFrame con los datos de incendios.

        Retorna:
        Nada
        """
        #Crear una nueva figura para el gráfico con el tamaño especificado
        plt.figure(figsize=(12, 8))

        #Calcular la mediana del total de hectáreas quemadas por región y ordenar los valores
        df.groupby('Región')['Total hectáreas'].median().sort_values().plot(kind='bar', color='skyblue')

        #Agregar título al gráfico
        plt.title('Comparación del Total de Hectáreas Quemadas por Región', fontsize=16)

        #Etiquetas de los ejes x e y
        plt.xlabel('Región', fontsize=14)
        plt.ylabel('Total de Hectáreas Quemadas', fontsize=14)

        #Rotar las etiquetas del eje x para mejor visualización
        plt.xticks(rotation=45, ha='right', fontsize=12)

        #Tamaño de las etiquetas en el eje y
        plt.yticks(fontsize=12)

        #Mostrar el gráfico de barras
        plt.show()

        #Cerrar la figura para liberar memoria
        plt.close()
```

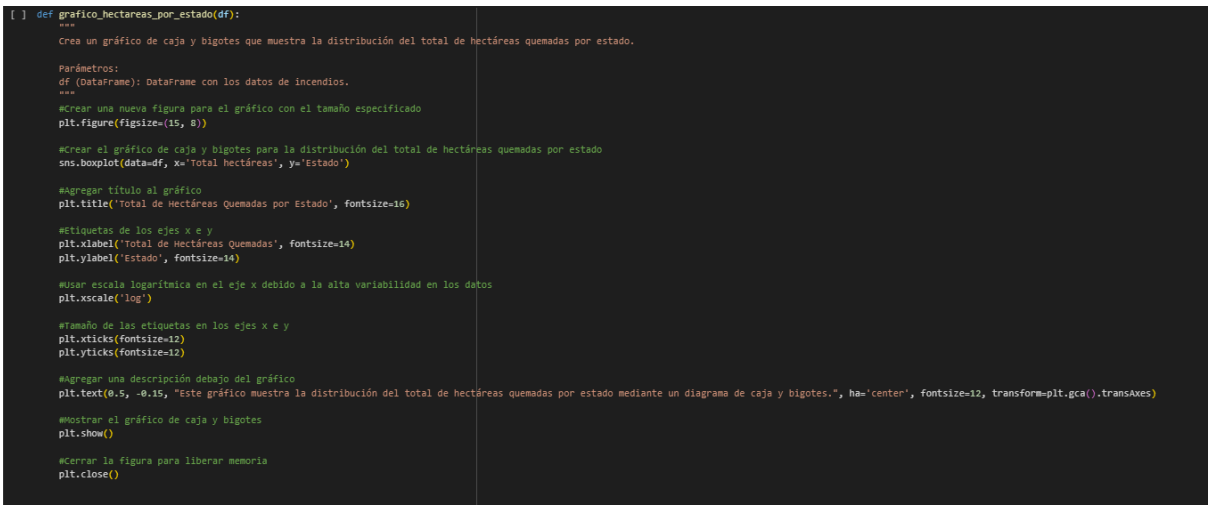
Función: `grafico_comparacion_total_hectareas_regiones(df)`

Comandos y funciones utilizados:

- `groupby().median().sort_values().plot(kind='bar')`: Agrupa los datos por región, calcula la mediana del total de hectáreas quemadas en cada región y traza estos datos en un gráfico de barras.
- `plt.title()`, `plt.xlabel()`, `plt.ylabel()`, `plt.xticks()`, `plt.yticks()`: Agrega títulos y etiquetas a los ejes del gráfico.
- `plt.show()`, `plt.close()`: Muestra y cierra el gráfico respectivamente.

9. Caja de bigote que muestra hectáreas quemadas por estado: Esta función genera un gráfico de caja y bigotes que muestra la distribución del total de hectáreas quemadas por estado. Utiliza la biblioteca Seaborn para crear un gráfico de caja y bigotes, donde cada caja representa un estado y su altura y

longitud indican la variabilidad en el total de hectáreas quemadas en ese estado.



```
[ ]  def grafico_hectareas_por_estado(df):
        """
        Crea un gráfico de caja y bigotes que muestra la distribución del total de hectáreas quemadas por estado.

        Parámetros:
        df (DataFrame): DataFrame con los datos de incendios.
        """
        #Crear una nueva figura para el gráfico con el tamaño especificado
        plt.figure(figsize=(15, 8))

        #Crear el gráfico de caja y bigotes para la distribución del total de hectáreas quemadas por estado
        sns.boxplot(data=df, x='Total hectáreas', y='Estado')

        #Agregar título al gráfico
        plt.title('Total de Hectáreas Quemadas por Estado', fontsize=16)

        #Etiquetas de los ejes x e y
        plt.xlabel('Total de Hectáreas Quemadas', fontsize=14)
        plt.ylabel('Estado', fontsize=14)

        #Usar escala logarítmica en el eje x debido a la alta variabilidad en los datos
        plt.xscale('log')

        #Tamaño de las etiquetas en los ejes x e y
        plt.xticks(fontsize=12)
        plt.yticks(fontsize=12)

        #Agregar una descripción debajo del gráfico
        plt.text(0.5, -0.15, "Este gráfico muestra la distribución del total de hectáreas quemadas por estado mediante un diagrama de caja y bigotes.", ha='center', fontsize=12, transform=plt.gca().transAxes)

        #Mostrar el gráfico de caja y bigotes
        plt.show()

        #Cerrar la figura para liberar memoria
        plt.close()
```

Función: `grafico_hectareas_por_estado(df)`

Comandos y funciones utilizados:

- `sns.boxplot()`: Crea un gráfico de caja y bigotes utilizando Seaborn.
- `plt.title()`, `plt.xlabel()`, `plt.ylabel()`, `plt.xscale()`, `plt.xticks()`, `plt.yticks()`: Agrega títulos y etiquetas a los ejes del gráfico.
- `plt.text()`: Añade un texto explicativo en el gráfico.
- `plt.show()`: Muestra el gráfico.

10. Mapa de calor de correlaciones entre variables numéricas: Crea un mapa de calor que muestra las correlaciones entre variables numéricas en el conjunto de datos. Utiliza la función **corr** de pandas para calcular la matriz de correlación entre todas las variables numéricas y luego traza esta matriz en un mapa de calor utilizando la biblioteca Seaborn. Los colores más intensos indican correlaciones más fuertes entre las variables. Esta visualización permite identificar relaciones y patrones entre las variables numéricas en el conjunto de datos.

```python
[ ] def mapa_calor_correlaciones(df):
        """
        esta función proporciona una herramienta visual para explorar y comprender las relaciones entre variables numéricas en un DataFrame.

       Parámetros:
        df (DataFrame): DataFrame con las variables.

        Retorna:
        Nada
        """
        #Filtra solo las columnas numéricas para el cálculo de la correlación
        columnas_numericas = df.select_dtypes(include='number').columns  # Selecciona las columnas numéricas
        df_numeric = df[columnas_numericas]  # Crea un nuevo DataFrame con las columnas numéricas

        #Calcula la matriz de correlación
        correlaciones = df_numeric.corr()  # Calcula la matriz de correlación entre las variables numéricas

        #Crea el mapa de calor de las correlaciones
        plt.figure(figsize=(12, 8))  #Crea una nueva figura con el tamaño especificado
        sns.heatmap(correlaciones, annot=True, cmap='coolwarm', linewidths=0.5)  #Crea el mapa de calor
        plt.title('Mapa de Calor de Correlaciones entre Variables Numéricas', fontsize=16)  #Agrega título al gráfico
        plt.xticks(rotation=45, ha='right', fontsize=12)  #Ajusta las etiquetas del eje x
        plt.yticks(rotation=0, fontsize=12)  #Ajusta las etiquetas del eje y
        plt.show()  #Muestra el gráfico
        plt.close()  #Cierra la figura
```

Función: `mapa_calor_correlaciones(df)`

Comandos y funciones utilizados:

- `corr()`: Calcula la matriz de correlación entre todas las variables numéricas en el conjunto de datos.
- `sns.heatmap()`: Crea un mapa de calor de correlaciones utilizando Seaborn.
- `plt.title()`: Agrega un título al gráfico.
- `plt.show()`: Muestra el gráfico.

Paso 4: Menú interactivo y visualización.

Esta parte del análisis proporciona una interfaz interactiva para seleccionar y visualizar diferentes gráficos relacionados con los incendios forestales. Los usuarios pueden elegir entre una variedad de opciones numeradas que corresponden a diferentes tipos de visualizaciones. Una vez seleccionada una opción, el programa genera automáticamente la visualización correspondiente utilizando los datos proporcionados sobre incendios forestales. Esta funcionalidad permite a los usuarios explorar y analizar los datos de manera dinámica y personalizada."

```python
while True:
    # Mostrar el menú de opciones
    print("\nSeleccione el gráfico que desea visualizar:")
    print("1. Número de Incendios por Estado")
    print("2. Número de Incendios a lo Largo del Tiempo (por Año)")
    print("3. Número de Incendios por Mes")
    print("4. Distribución Geográfica de los Incendios")
    print("5. Mapa de calor Geográfica de los Incendios")
    print("6. Porcentaje de Incendios por Temporada")
    print("7. Porcentaje de Incendios por Tipo de Vegetación Afectada")
    print("8. Comparación del Total de Hectáreas Quemadas por Región")
    print("9. Gráfico de Caja y Bigotes que Muestra Hectáreas Quemadas por Estado")
    print("10. Mapa de Calor de Correlaciones entre Variables Numéricas")
    print("11. Salir")

    opcion = input("Ingrese el número de su elección: ")  # Solicitar al usuario que ingrese una opción

    # Validar la opción ingresada por el usuario y llamar a la función correspondiente
    if opcion == '1':
        #Gráfico de barras que muestra el número de incendios por estado
        grafico_incendios_por_estado(fire_df)
    elif opcion == '2':
        #Gráfico de líneas que muestra la cantidad de incendios a lo largo del tiempo
        grafico_lineas_incendios_por_año(fire_df)
    elif opcion == '3':
        #Gráfico de barras que muestra el número de incendios por mes
        grafico_barras_incendios_por_mes(fire_df)
    elif opcion == '4':
        #Gráfico de dispersión que muestra la distribución geográfica de los incendios
        grafico_dispersion_incendios_latitud_longitud(fire_df)
    elif opcion == '5':
        #Mapa de calor que muestra la concentración de incendios en función de su ubicación geográfica
        mapa_calor_densidad_incendios(fire_df)
    elif opcion == '6':
        #Gráfico de pastel que muestra el porcentaje de incendios por temporada
        grafico_porcentaje_incendios_temporada(fire_df)
    elif opcion == '7':
        #Gráfico de pastel que muestra la proporción de incendios por tipo de vegetación afectada
        grafico_pastel_vegetacion_afectada(fire_df)
    elif opcion == '8':
        #Gráfico de barras que compara el total de hectáreas quemadas por región
        grafico_comparacion_total_hectareas_regiones(fire_df)
    elif opcion == '9':
        #Gráfico de caja y bigotes que muestra la distribución del total de hectáreas quemadas por estado
        grafico_hectareas_por_estado(fire_df)
    elif opcion == '10':
        #Mapa de calor que muestra las correlaciones entre variables numéricas
        mapa_calor_correlaciones(fire_df)
    elif opcion == '11':
        print("Saliendo...")
        break  #Salir del bucle while si el usuario elige salir
    else:
        print("Opción no válida. Por favor, intente nuevamente.")  #Mensaje de error si el usuario ingresa una opción no válida
```

Función: `menu_interactivo(fire_df):`

- Ofrece al usuario un menú numerado de opciones de visualización.
- Solicita al usuario que elija una opción ingresando un número.
- Ejecuta la función correspondiente según la opción seleccionada.
- El programa continúa ejecutándose hasta que el usuario elige salir.

Opciones del menú:

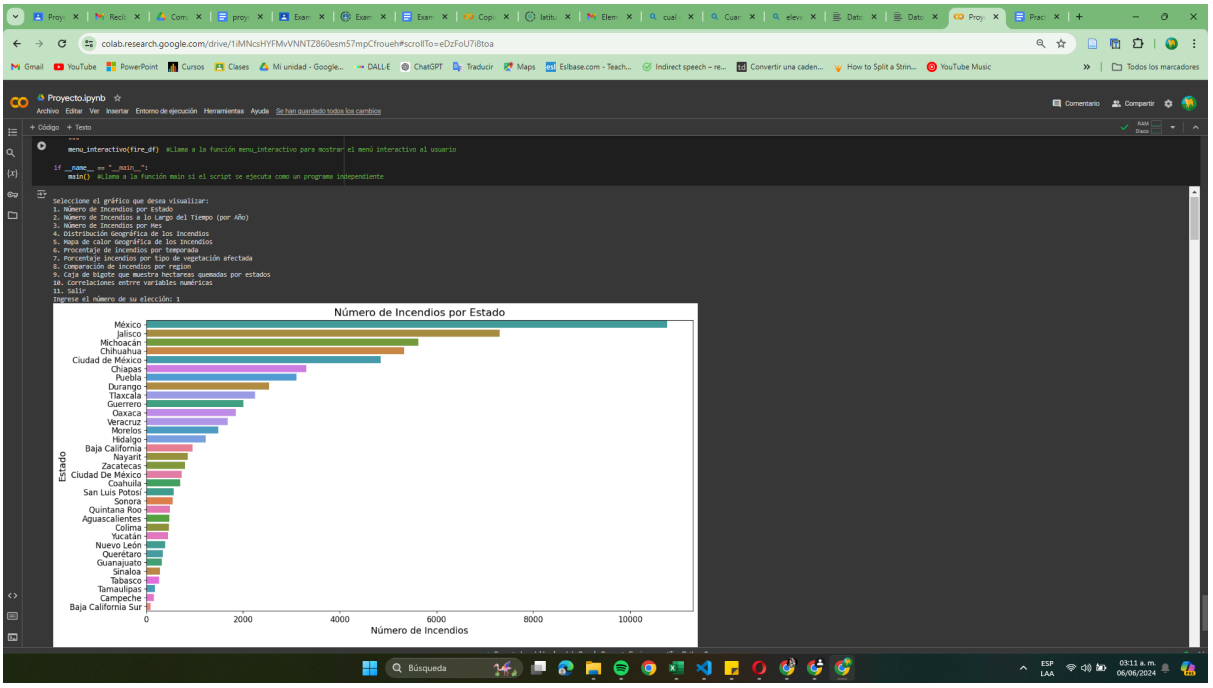
- Cada opción del menú corresponde a una función específica que genera un tipo de visualización.
- Las funciones utilizan técnicas de visualización como gráficos de barras, líneas, dispersión, etc., para mostrar los datos de manera comprensible.

Interacción del usuario:

- El usuario puede explorar diferentes visualizaciones simplemente ingresando el número de opción deseado.
- Una vez que el usuario termina de explorar, puede elegir salir para finalizar el programa.

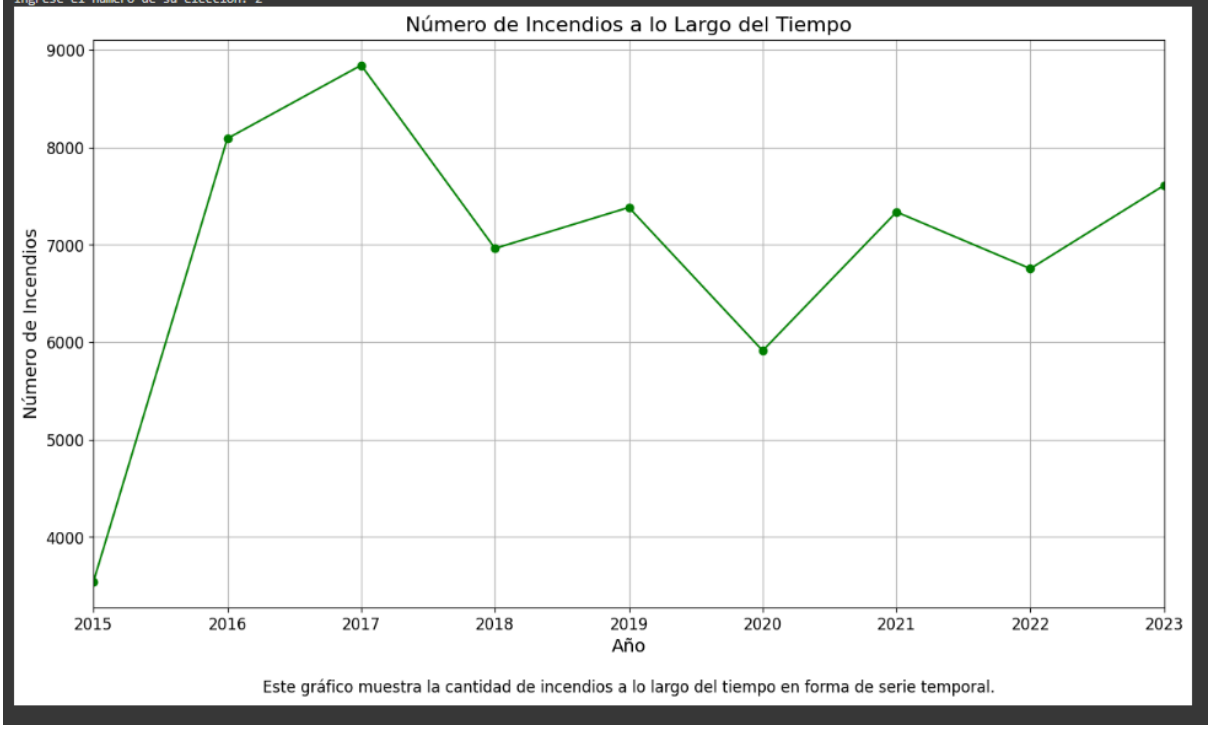
*Visualización

1



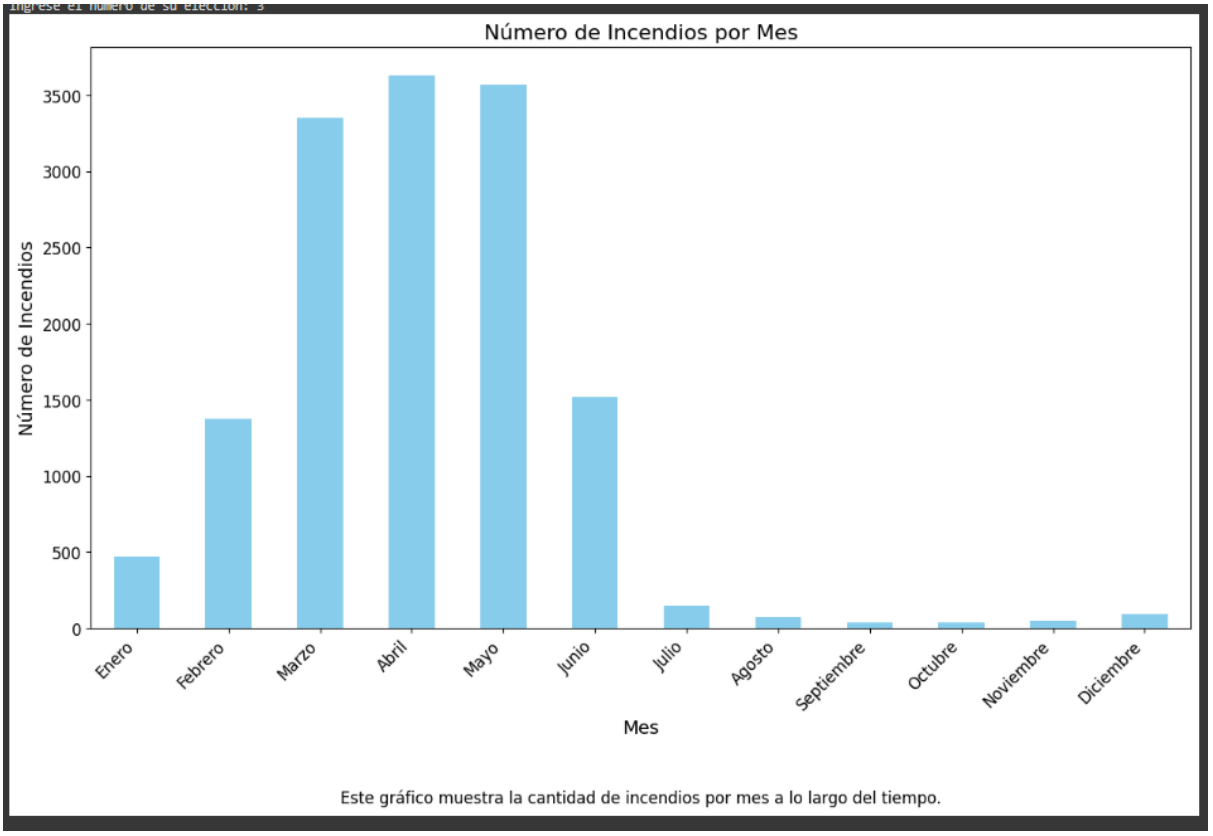
2

```
Seleccione el gráfico que desea visualizar:
1. Número de Incendios por Estado
2. Número de Incendios a lo Largo del Tiempo (por Año)
3. Número de Incendios por Mes
4. Distribución Geográfica de los Incendios
5. Mapa de calor Geográfica de los Incendios
6. Procentaje de incendios por temporada
7. Porcentaje incendios por tipo de vegetación afectada
8. Comparación de incendios por region
9. Caja de bigote que muestra hectareas quemadas por estados
10. Correlaciones entrre variables numéricas
11. Salir
Ingrese el número de su elección: 2
```



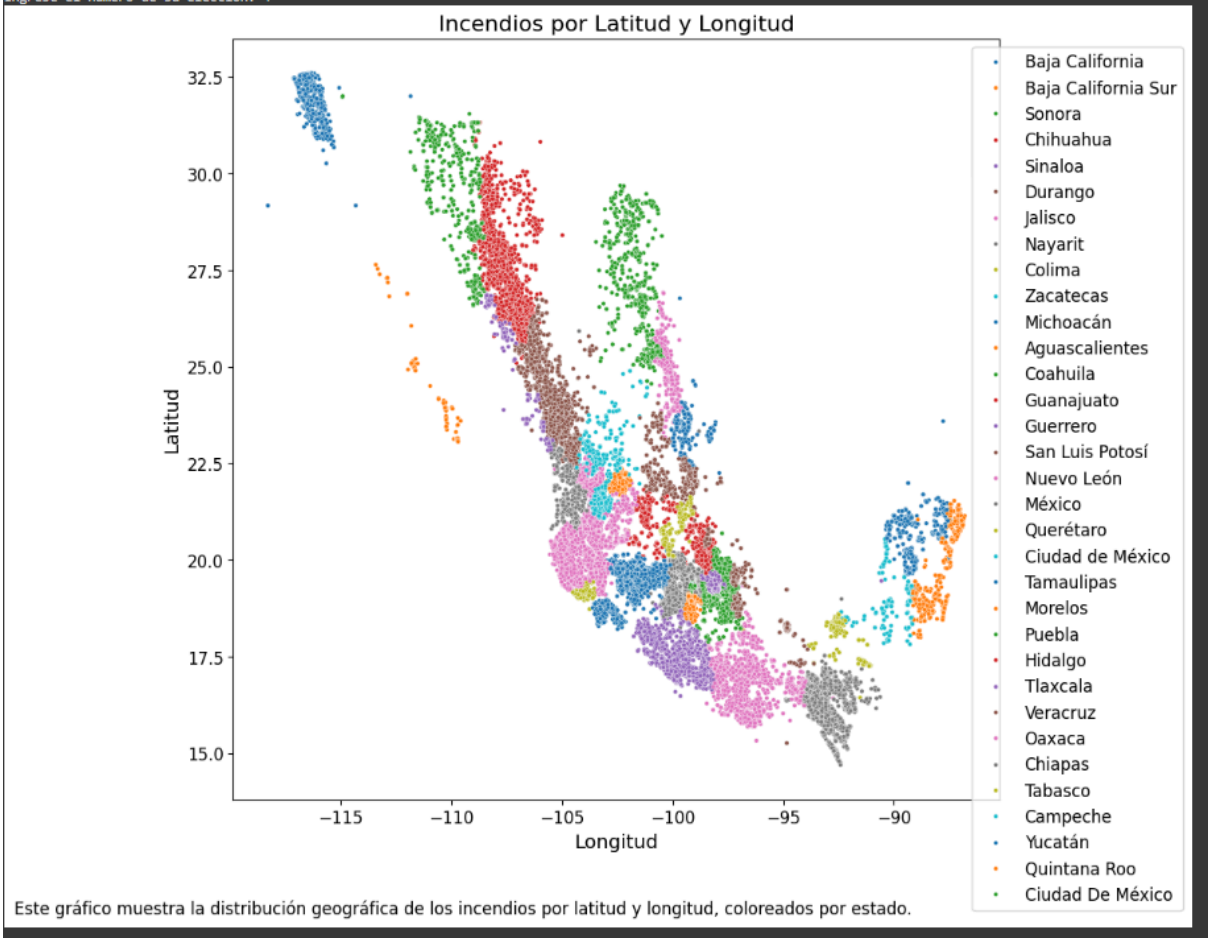
Este gráfico muestra la cantidad de incendios a lo largo del tiempo en forma de serie temporal.

3


Ingrese el número de su elección: 3
Número de Incendios por Mes
Número de Incendios
0
500
1000
1500
2000
2500
3000
3500
Enero
Febrero
Marzo
Abril
Mayo
Junio
Julio
Agosto
Septiembre
Octubre
Noviembre
Diciembre
Mes
Este gráfico muestra la cantidad de incendios por mes a lo largo del tiempo.

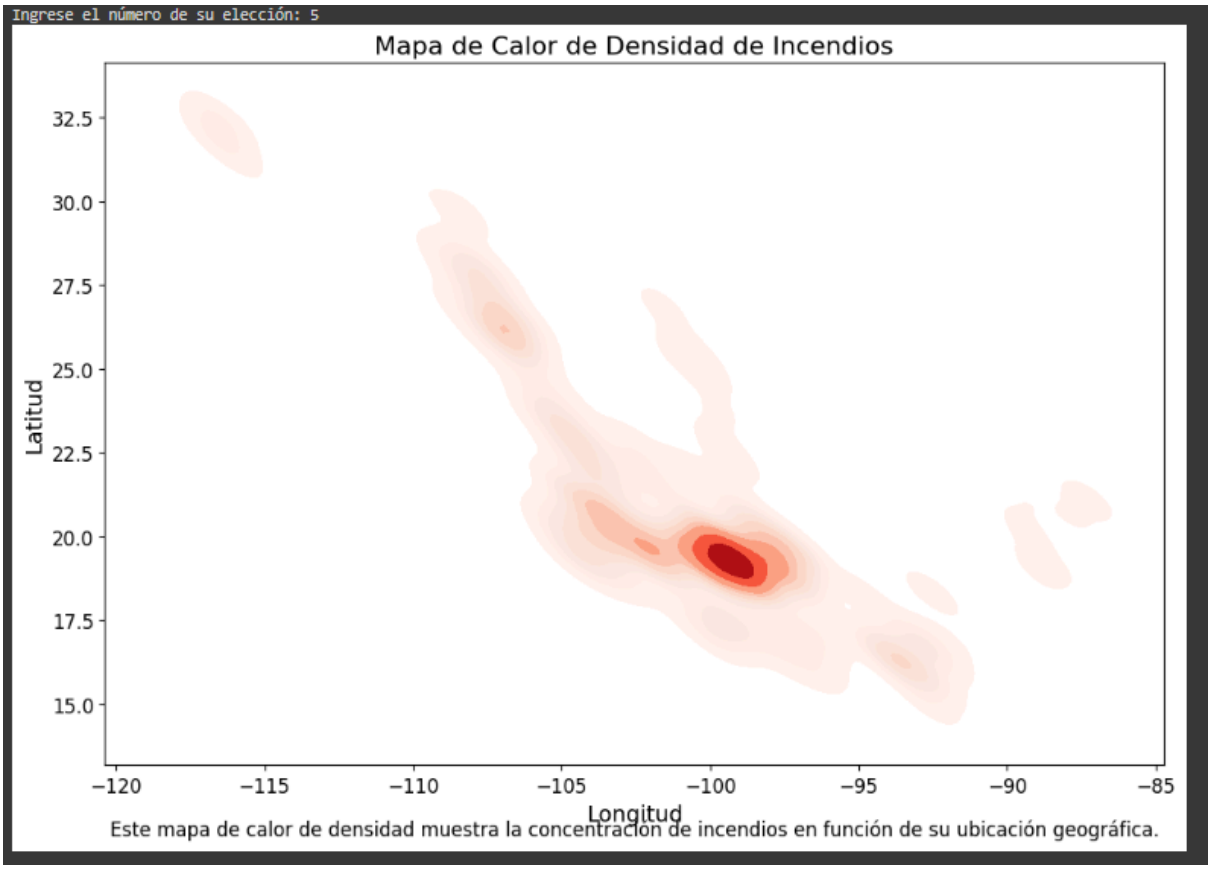
4

```
Ingrese el número de su elección: 4
```

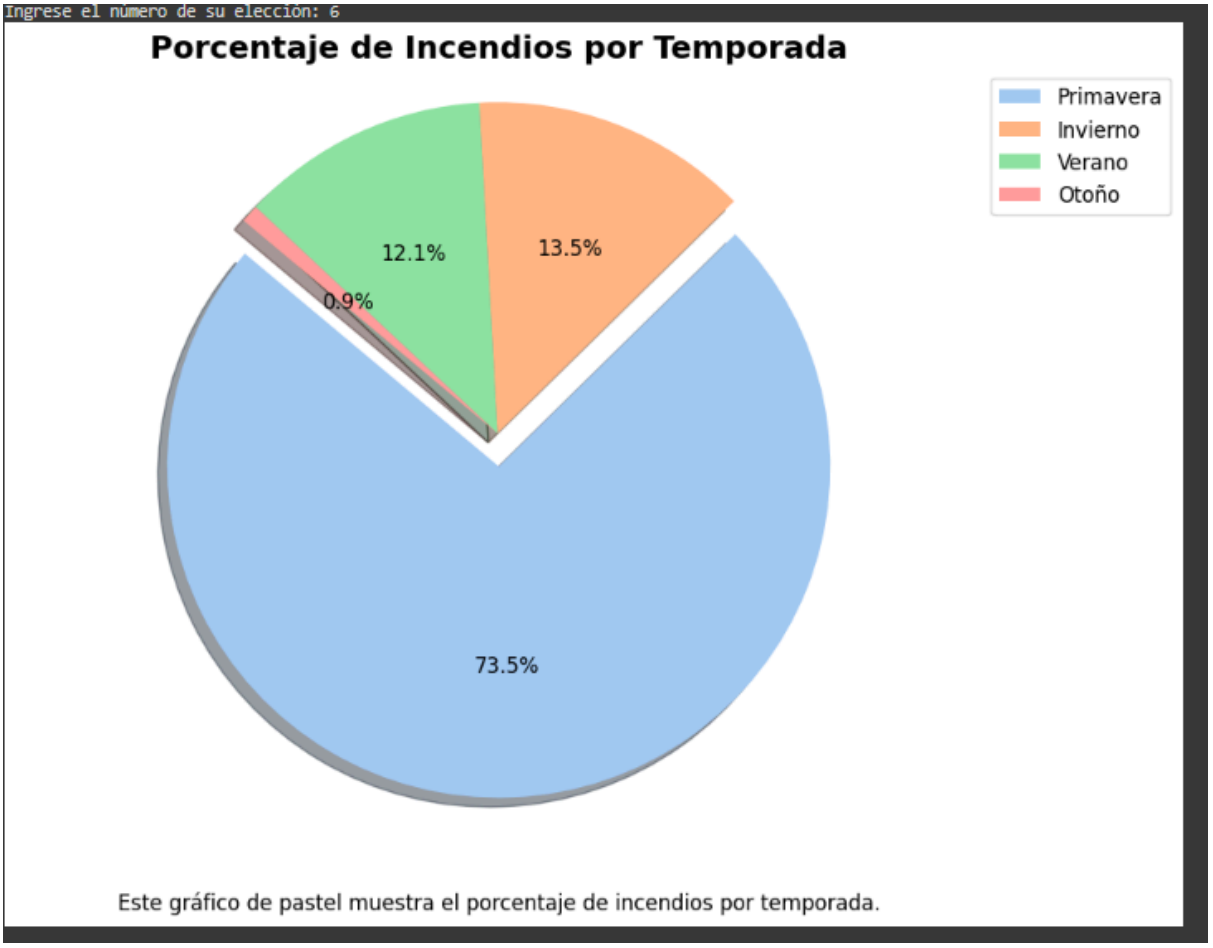


Este gráfico muestra la distribución geográfica de los incendios por latitud y longitud, coloreados por estado.

5.

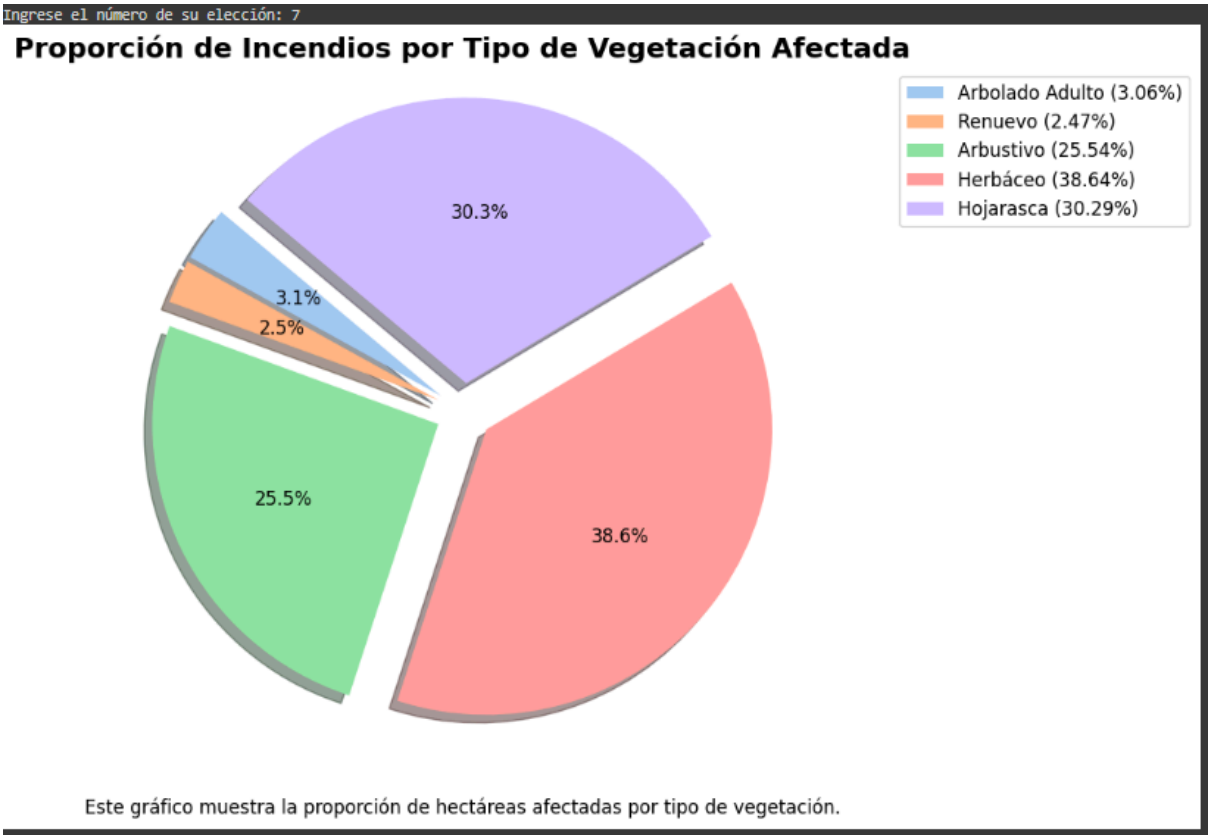

Ingrese el número de su elección: 5
Mapa de Calor de Densidad de Incendios
Latitud
32.5
30.0
27.5
25.0
22.5
20.0
17.5
15.0
−120
−115
−110
−105
−100
−95
−90
−85
Longitud
Este mapa de calor de densidad muestra la concentración de incendios en función de su ubicación geográfica.

6.

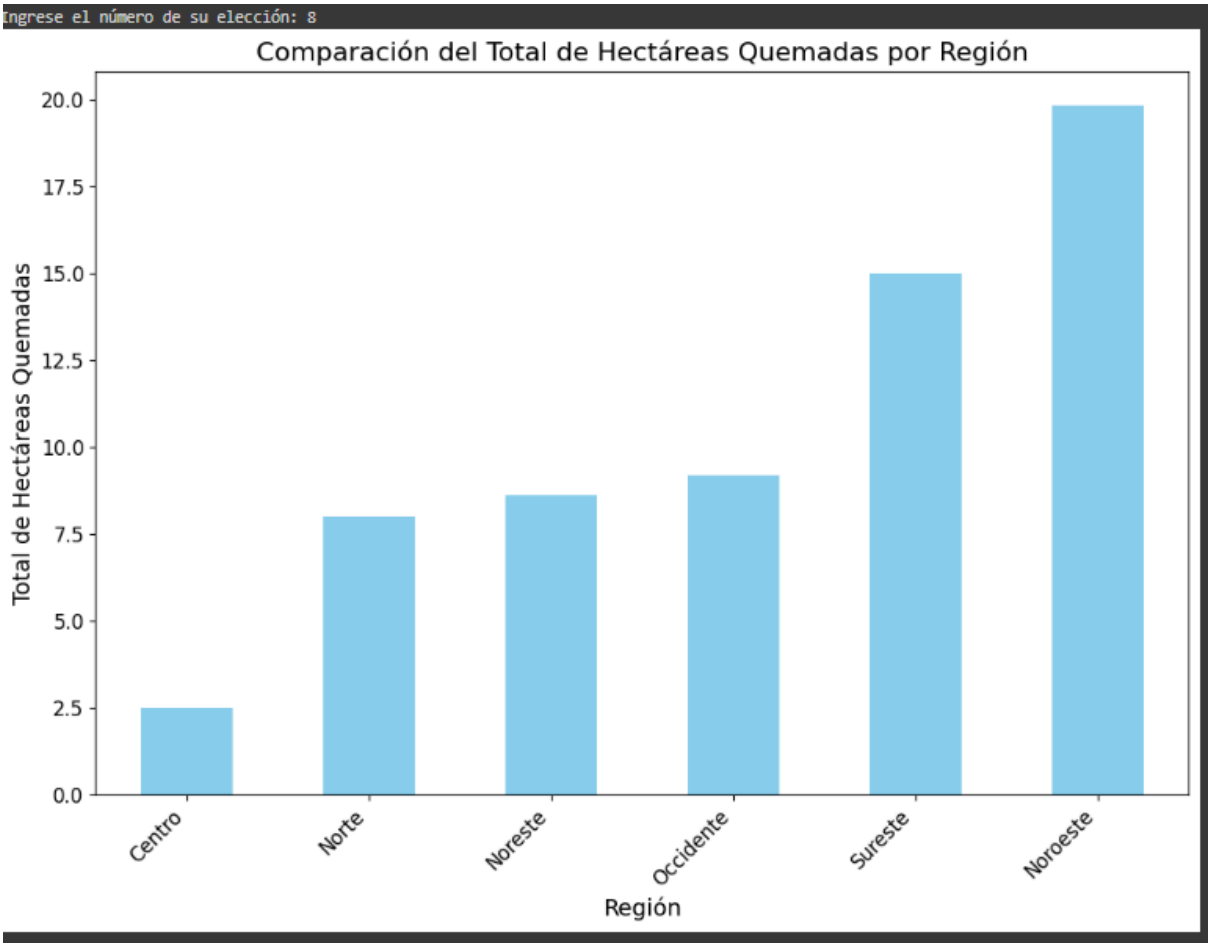

Ingrese el número de su elección: 6
Porcentaje de Incendios por Temporada
Primavera
Invierno
Verano
Otoño
12.1%
13.5%
0.9%
73.5%
Este gráfico de pastel muestra el porcentaje de incendios por temporada.

7.

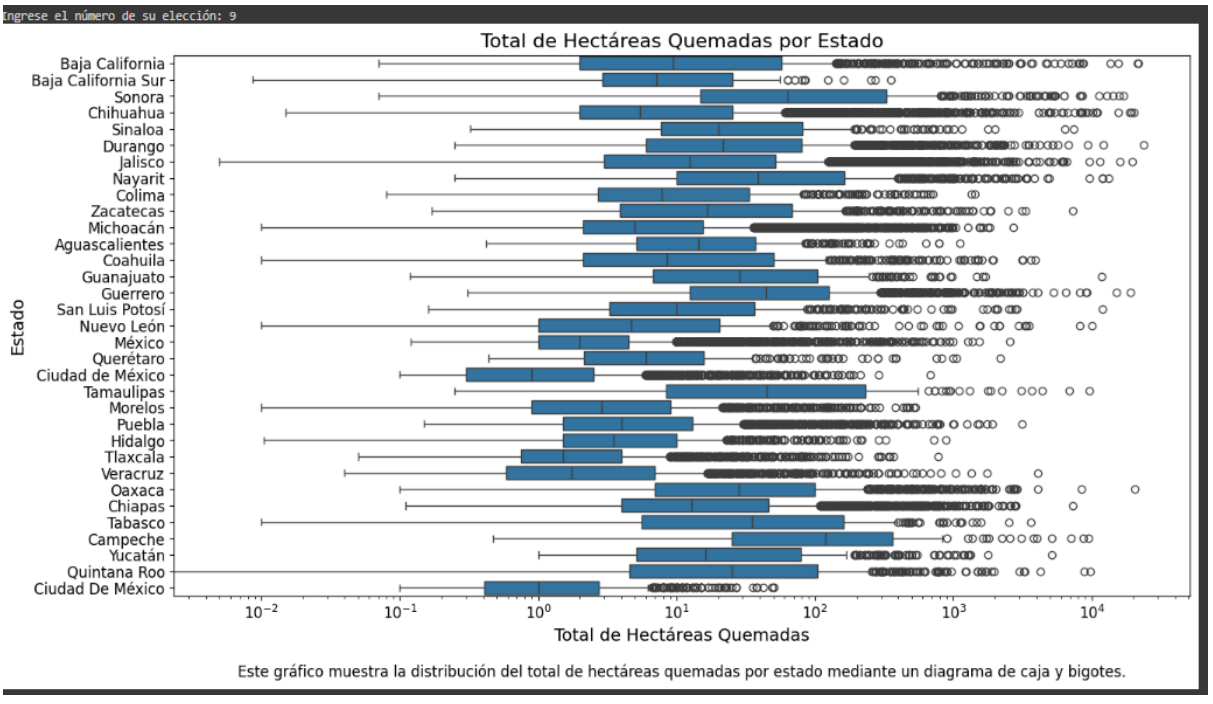
```
Ingrese el número de su elección: 7
```



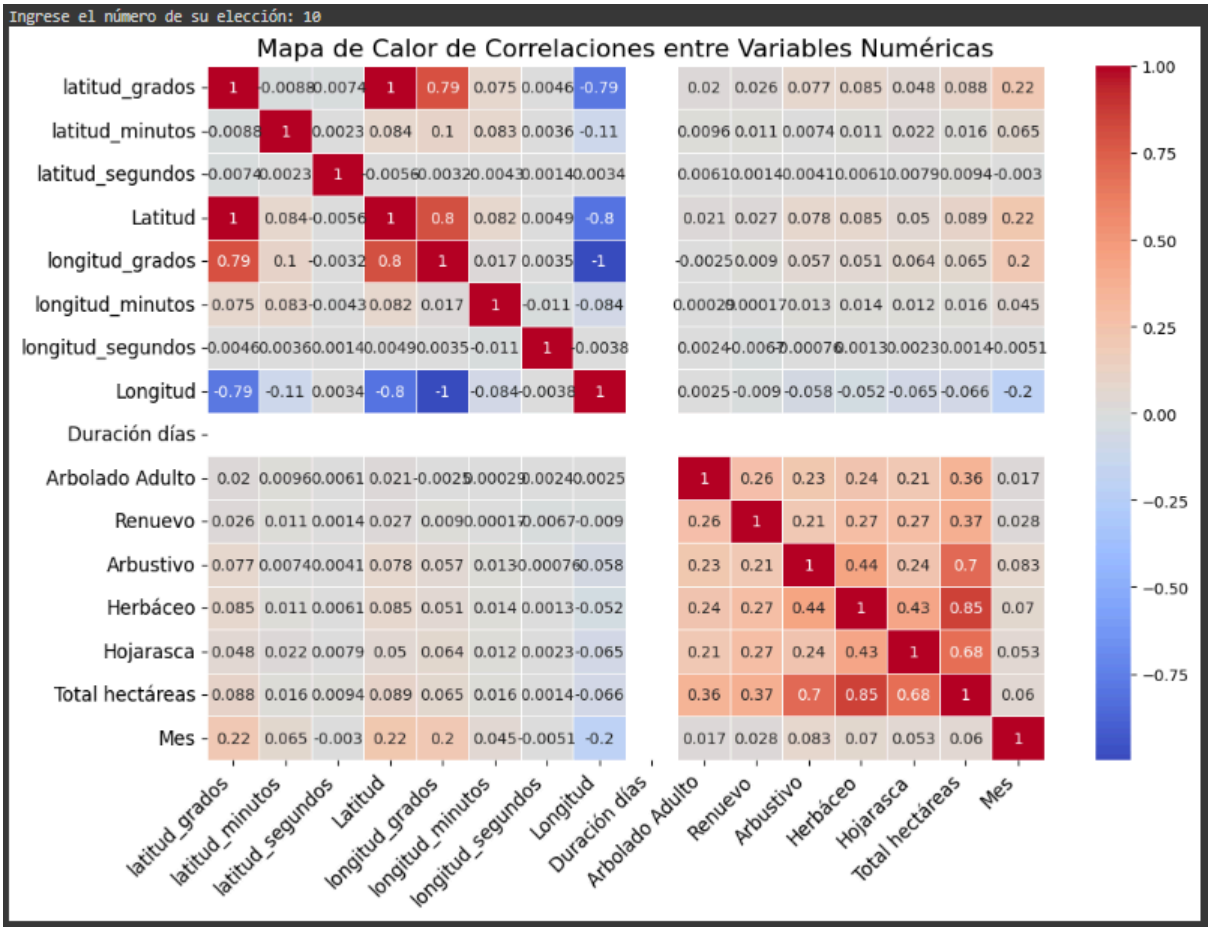
8.


Ingrese el número de su elección: 8
Comparación del Total de Hectáreas Quemadas por Región
Total de Hectáreas Quemadas
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Centro
Norte
Noreste
Occidente
Sureste
Noroeste
Región

9.


Ingrese el número de su elección: 9
Total de Hectáreas Quemadas por Estado
Baja California
Baja California Sur
Sonora
Chihuahua
Sinaloa
Durango
Jalisco
Nayarit
Colima
Zacatecas
Michoacán
Aguascalientes
Coahuila
Guanajuato
Guerrero
San Luis Potosí
Nuevo León
México
Querétaro
Ciudad de México
Tamaulipas
Morelos
Puebla
Hidalgo
Tlaxcala
Veracruz
Oaxaca
Chiapas
Tabasco
Campeche
Yucatán
Quintana Roo
Ciudad De México
Estado
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
Total de Hectáreas Quemadas
Este gráfico muestra la distribución del total de hectáreas quemadas por estado mediante un diagrama de caja y bigotes.

10.


Ingrese el número de su elección: 10
Mapa de Calor de Correlaciones entre Variables Numéricas
latitud_grados
latitud_minutos
latitud_segundos
Latitud
longitud_grados
longitud_minutos
longitud_segundos
Longitud
Duración días
Arbolado Adulto
Renuevo
Arbustivo
Herbáceo
Hojarasca
Total hectáreas
Mes
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

Referencia:

https://datos.gob.mx/busca/dataset/incendios-forestales