Tarea 2. Manipulación de Datos sobre Películas usando NumPy, Pandas y Matplotlib

1. **Manipulación y procesamiento de datos con Pandas y NumPy**

- Importar, limpiar y preparar los datos.
- Realizar transformaciones, agregaciones y cálculos estadísticos clave.
- Detectar valores faltantes o atípicos y decidir cómo tratarlos.
- Realizar análisis estadísticos básicos (promedios, máximos, mínimos, desviación estándar, etc.).

Esta sección aborda la manipulación y procesamiento de los datos mediante la librería Pandas y el uso de NumPy para cálculos avanzados. Se realizó la importación de los datos, limpieza, y transformaciones para obtener información clave de los datos. Además, se implementaron cálculos estadísticos básicos como el promedio, máximo, mínimo y desviación estándar, entre otros.

Código para importar y visualizar los datos:

```
import pandas as pd

df = pd.read_excel(r'C:\Users\alex_\Desktop\MATER CIENCIA DE DATOS\Primer cuatrimestre\Programación Ciencia de Datos\Examen 2\movies_dataset.xlsx', sheet_name='Sheet1')

print("Datos originales:")
print(df)
df

# INFORMACION DEL DATAFRAME

# 1. Verificar información general del DataFrame
print("\nInformación del DataFrame:")
print(df.info())
```

Esta es la información del DataFrame:

```
Información del DataFrame:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 10 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   movie_id          100 non-null    int64
 1   title             100 non-null    object
 2   genre             100 non-null    object
 3   release_year      100 non-null    int64
 4   critic_score      100 non-null    int64
 5   audience_score    100 non-null    int64
 6   box_office_millions  100 non-null    float64
 7   popularity        100 non-null    float64
 8   duration_minutes  100 non-null    int64
 9   budget_millions   100 non-null    float64
dtypes: float64(3), int64(5), object(2)
memory usage: 7.9+ KB
None
```

Se verificó si existían valores nulos y duplicados en el conjunto de datos. No se encontraron valores faltantes, pero se eliminaron valores duplicados para asegurar la consistencia del análisis.
Código para verificar y eliminar valores duplicados:

```
# Verificar si hay valores nulos
print("\nValores nulos por columna:")
print(df.isnull().sum())

df = df.dropna()

#Eliminar valores duplicados
df = df.drop_duplicates()

#Missing values
Missings = df.isnull().sum()

# Mostrar el DataFrame limpio
print("\nDatos limpios:")
print(df)
```

```
Datos limpios:
    movie_id                              title      genre  release_year  \
0          1                         The Green Mile    Fantasy          2019
1          2                                Se7en    Fantasy          2001
2          3  The Lord of the Rings: The Two Towers    Romance          2016
3          4                           Interstellar  Animation          1998
4          5                             Life of Pi     Comedy          1981
..       ...                                    ...        ...           ...
95        96                             Casablanca     Sci-Fi          2007
96        97                The Godfather: Part II   Thriller          2016
97        98               The Silence of the Lambs      Drama          1996
98        99                                  Alien     Action          2010
99       100                             The Matrix     Sci-Fi          2011

    critic_score  audience_score  box_office_millions  popularity  \
0             87              51                146.8         9.0
1             79              80                451.9         7.6
2             78              54                 37.0         6.9
3             82              71                410.4         7.6
4             61              61                155.6         9.2
..           ...             ...                  ...         ...
95            83              77                466.6         9.3
96            79              80                298.2         9.8
97            88              73                415.1         6.7
98            93              91                447.8         9.6
99            78              89                391.8         9.1

    duration_minutes  budget_millions
0                 89          237.14
1                128           62.59
2                 91           82.04
3                 80          114.75
4                131          178.47
..               ...             ...
95                93           79.72
96               136          213.39
97                97          104.86
98               135          161.53
99               129          177.25

[100 rows x 10 columns]
```

Figura 1: Esto sería la información con los datos limpios

Se realizan transformaciones de formato de la variable para un mejor análisis, además de agregar columnas nuevas para ver el rendimiento de cada película y género, así como el ROI. También se verifica qué películas están agotadas (Sold Out) y cuáles están en versión IMAX. Por último, se agregan cálculos estadísticos clave como la media, desviación estándar, moda y valores mínimos y máximos de cada película.

```
df['genre'] = df['genre'].astype('category')
df['movie_id'] = df['movie_id'].astype('object')
df['release_year'] = pd.to_datetime(df['release_year'], format='%Y')
# Mostrar el DataFrame actualizado
print(df)


import numpy as np

# Agrega las columnas con valores aleatorios
df['Sold Out'] = np.random.choice(['Sold Out', 'No'], df.shape[0])
df['Version IMAX'] = np.random.choice(['Si', 'No'], df.shape[0])
df['Ganancias_Perdidas'] = df['box_office_millions'] - df['budget_millions']
df['ROI'] = (df['box_office_millions']/df['budget_millions'])-1

# Mostrar el DataFrame actualizado
print(df)
```

```
   movie_id                                title      genre release_year  \
0         1                The Green Mile    Fantasy   2019-01-01
1         2                           Se7en    Fantasy   2001-01-01
2         3  The Lord of the Rings: The Two Towers    Romance   2016-01-01
3         4                  Interstellar  Animation   1998-01-01
4         5                    Life of Pi     Comedy   1981-01-01
..      ...                           ...        ...          ...
95       96                    Casablanca     Sci-Fi   2007-01-01
96       97        The Godfather: Part II   Thriller   2016-01-01
97       98      The Silence of the Lambs      Drama   1996-01-01
98       99                         Alien     Action   2010-01-01
99      100                    The Matrix     Sci-Fi   2011-01-01

    critic_score  audience_score  box_office_millions  popularity  \
0             87              51                146.8         9.0
1             79              80                451.9         7.6
2             78              54                 37.0         6.9
3             82              71                410.4         7.6
4             61              61                155.6         9.2
..           ...             ...                  ...         ...
95            83              77                466.6         9.3
96            79              80                298.2         9.8
97            88              73                415.1         6.7
98            93              91                447.8         9.6
99            78              89                391.8         9.1

    duration_minutes  budget_millions  Sold Out Version IMAX  \
0                 89          237.14        No           Si
1                128           62.59        No           No
2                 91           82.04  Sold Out           Si
3                 80          114.75        No           No
4                131          178.47  Sold Out           No
..               ...             ...       ...          ...
95                93           79.72        No           Si
96               136          213.39        No           No
97                97          104.86        No           No
98               135          161.53  Sold Out           Si
99               129          177.25        No           Si

    Ganancias_Perdidas       ROI
0               -90.34 -0.380956
1               389.31  6.220003
2               -45.04 -0.549000
3               295.65  2.576471
4               -22.87 -0.128145
..                 ...       ...
95              386.88  4.852985
96               84.81  0.397441
97              310.24  2.958611
98              286.27  1.772240
99              214.55  1.210437

[100 rows x 14 columns]
```

Figura 2 output: Después de agregar columnas y transformaciones

```
#Estadistica


Medias = {
   'critic_score': df['critic_score'].mean(),
   'audience_score': df['audience_score'].mean(),
   'box_office_millions': df['box_office_millions'].mean(),
   'duration_minutes': df['duration_minutes'].mean(),
   'budget_millions': df['budget_millions'].mean()
}


for columna, valor in Medias.items():
   print(f"Media {columna}: {valor}")


columns_to_check = ['release_year', 'critic_score', 'audience_score',
                'box_office_millions', 'popularity',
                'duration_minutes', 'budget_millions']

for column in columns_to_check:
   max_value = df[column].max()
   movie_title = df.loc[df[column] == max_value, 'title'].values[0]
   print(f"Máximo {column}: {max_value}, Película: {movie_title}")



columns_to_check = ['release_year', 'critic_score', 'audience_score',
                'box_office_millions', 'popularity',
                'duration_minutes', 'budget_millions']

for column in columns_to_check:
   min_value = df[column].min()  # Cambiado a min()
   movie_title = df.loc[df[column] == min_value, 'title'].values[0]  # Obtener el título de la película
   print(f"Mínimo {column}: {min_value}, Película: {movie_title}")

Desviaciones = {
   'critic_score': df['critic_score'].std(),
   'audience_score': df['audience_score'].std(),
   'box_office_millions': df['box_office_millions'].std(),
   'duration_minutes': df['duration_minutes'].std(),
   'budget_millions': df['budget_millions'].std()
}

for columna, valor in Desviaciones.items():
   print(f"Desviación estándar {columna}: {valor}")


columns_to_check = ['release_year', 'critic_score', 'audience_score',
                'box_office_millions', 'popularity',
                'duration_minutes', 'budget_millions']

for column in columns_to_check:
   mode_value = df[column].mode()[0]  # Calcular la moda
   # Obtener los títulos de las películas que tienen la moda
   movie_titles = df.loc[df[column] == mode_value, 'title'].values
   print(f"Moda {column}: {mode_value}, Películas: {', '.join(movie_titles)}")


imax_count = df['Version IMAX'].str.lower().value_counts().get('si', 0)
sold_out_count = df['Sold Out'].str.lower().value_counts().get('sold out', 0)

print(f"Películas en IMAX: {imax_count}")
print(f"Películas Sold Out: {sold_out_count}")
```

Este seria el putput de las estadisticas :

```
Media critic_score: 80.21
Media audience_score: 73.65
Media box_office_millions: 280.707
Media duration_minutes: 126.05
Media budget_millions: 171.82119999999998
Máximo release_year: 2022-01-01 00:00:00, Película: The Departed
Máximo critic_score: 99, Película: Her
Máximo audience_score: 98, Película: Pan's Labyrinth
Máximo box_office_millions: 495.8, Película: North by Northwest
Máximo popularity: 10.0, Película: Her
Máximo duration_minutes: 179, Película: Us
Máximo budget_millions: 299.49, Película: 12 Angry Men
Mínimo release_year: 1980-01-01 00:00:00, Película: Avengers: Endgame
Mínimo critic_score: 60, Película: Black Panther
Mínimo audience_score: 50, Película: Ratatouille
Mínimo box_office_millions: 20.1, Película: The Usual Suspects
Mínimo popularity: 6.0, Película: The Grand Budapest Hotel
Mínimo duration_minutes: 80, Película: Interstellar
Mínimo budget_millions: 50.5, Película: The Prestige
Desviación estándar critic_score: 11.225237618685833
Desviación estándar audience_score: 14.367032267020774
Desviación estándar box_office_millions: 137.90263968217985
Desviación estándar duration_minutes: 27.88771678489917
Desviación estándar budget_millions: 68.70202524884495
Moda release_year: 2011-01-01 00:00:00, Películas: The Great Gatsby, The Goonies, The
Shape of Water, Schindler's List, Gladiator, The Empire Strikes Back, The Matrix
Moda critic_score: 90, Películas: The Big Short, It, Inception, Jurassic Park, Rear Window, The
Usual Suspects
Moda audience_score: 71, Películas: Interstellar, Up, Her, The Martian, The Big Lebowski
Moda box_office_millions: 338.5, Películas: Whiplash, The Social Network
Moda popularity: 7.6, Películas: Se7en, Interstellar, The Goonies, Rocketman, La La Land
Moda duration_minutes: 93, Películas: Parasite, Saving Private Ryan, Gladiator, Spotlight,
Casablanca
Moda budget_millions: 50.5, Películas: The Prestige
Películas en IMAX: 47
Películas Sold Out: 56
[ ]:
```

Los resultados muestran un análisis de varias películas y sus características. En promedio, las

películas tienen una puntuación de críticos de 80.21 y una puntuación de audiencia de 73.65, lo que significa que a los críticos les gustan un poco más que al público. El presupuesto promedio para hacer estas películas es de 171.82 millones de dólares, y en taquilla, recaudan alrededor de 280.71 millones, lo que indica que generalmente ganan más de lo que costaron. La duración promedio de las películas es de 126 minutos.Entre las películas destacadas, "Her" tiene la mejor puntuación crítica (99) y es la más popular (10.0), mientras que "Pan's Labyrinth" tiene la mejor puntuación entre el público (98). "North by Northwest" es la que más dinero ha recaudado, con 495.8 millones. La película más reciente en la lista es "The Departed", de 2022, y la más antigua es "Avengers: Endgame", de 1980.Además, hay una gran variedad en las puntuaciones y el rendimiento financiero de las películas, lo que muestra que hay muchas diferencias en la calidad y el éxito. También se menciona que 47 películas se proyectaron en IMAX y 56 se agotaron, lo que indica que algunas producciones son muy populares entre los espectadores

2. **Análisis exploratorio de datos y visualización con Matplotlib**

- Generar visualizaciones claras y efectivas para mostrar tendencias clave, tales como:
    - **Distribución de puntuaciones** de las películas (por ejemplo, cómo se distribuyen las puntuaciones de los críticos o los espectadores).

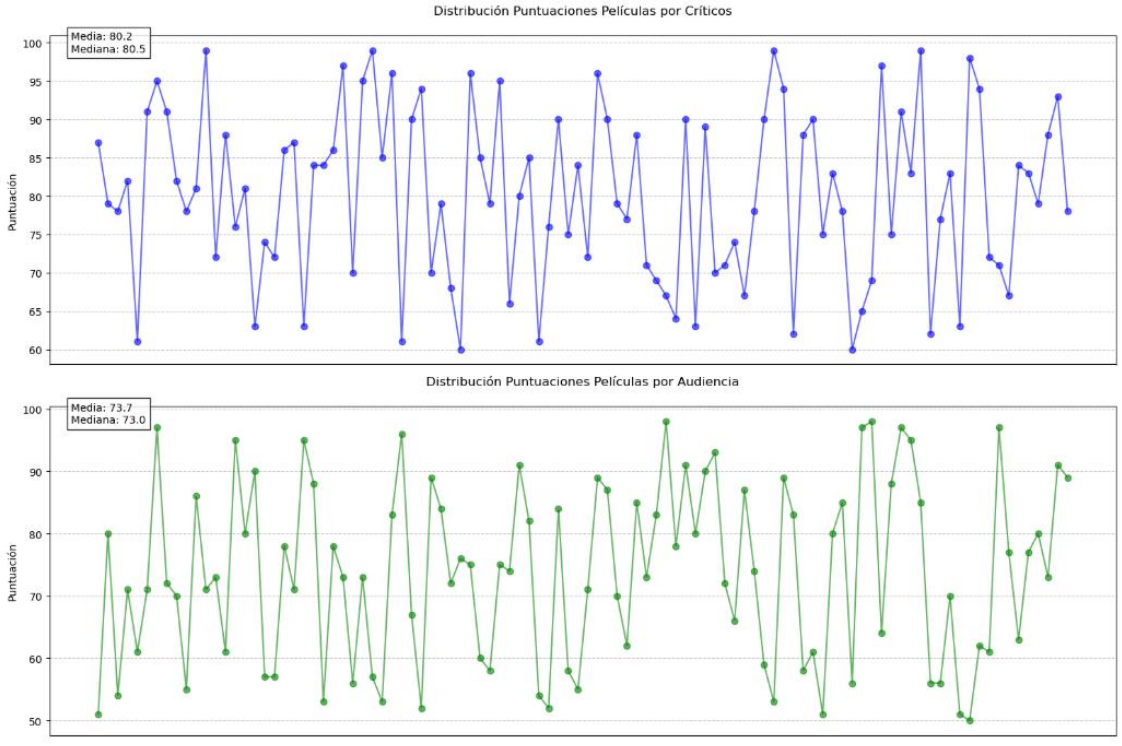
Ejecutamos el código

```
#2.1 Distribución de puntuaciones

import matplotlib.pyplot as plt


def visualizar_distribucion_puntuaciones(df):

  plt.figure(figsize=(15, 10))


  plt.subplot(2, 1, 1)

  plt.plot(range(len(df)), df['critic_score'],
```

```
        marker='o',
        linestyle='-',
        color='blue',
        markersize=6,
        alpha=0.6,
        label='Puntuación Críticos')
plt.title('Distribución Puntuaciones Películas por Críticos', pad=20)
plt.ylabel('Puntuación')
plt.grid(True, linestyle='--', alpha=0.7)
plt.xticks([])

stats_text = f"Media: {df['critic_score'].mean():.1f}\n"
stats_text += f"Mediana: {df['critic_score'].median():.1f}"
plt.text(0.02, 0.95, stats_text,
        transform=plt.gca().transAxes,
        bbox=dict(facecolor='white', edgecolor='black', alpha=0.8))

plt.subplot(2, 1, 2)
plt.plot(range(len(df)), df['audience_score'],
        marker='o',
        linestyle='-',
        color='green',
        markersize=6,
        alpha=0.6,
        label='Puntuación Audiencia')
plt.title('Distribución Puntuaciones Películas por Audiencia', pad=20)
plt.ylabel('Puntuación')
plt.grid(True, linestyle='--', alpha=0.7)
plt.xticks([])
stats_text = f"Media: {df['audience_score'].mean():.1f}\n"
```

```
stats_text += f"Mediana: {df['audience_score'].median():.1f}"
plt.text(0.02, 0.95, stats_text,
        transform=plt.gca().transAxes,
        bbox=dict(facecolor='white', edgecolor='black', alpha=0.8))
plt.tight_layout()
plt.show()
```

```
visualizar_distribucion_puntuaciones(df)
```

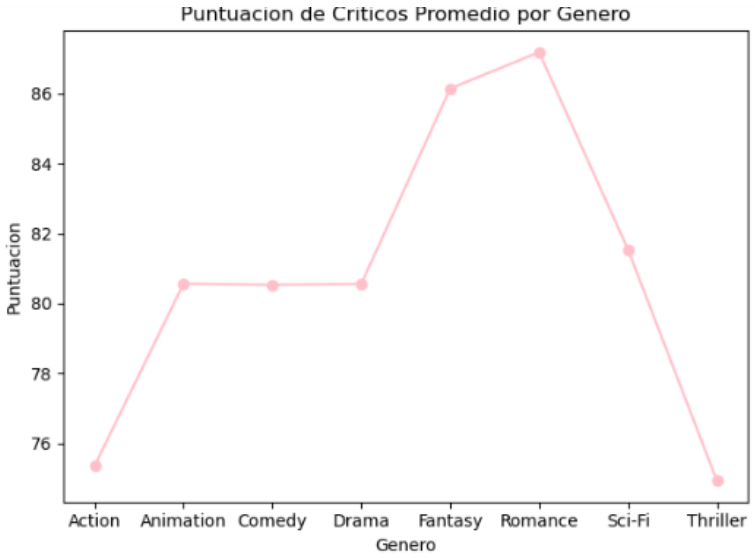


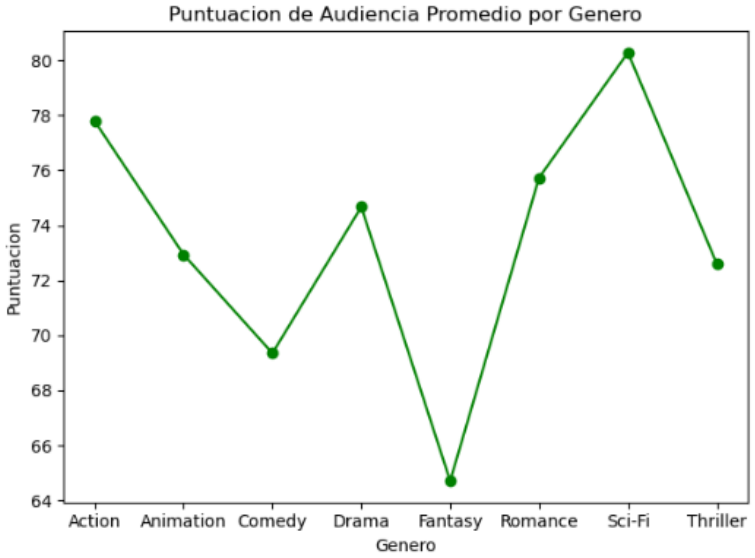
Los críticos tienden a dar puntuaciones más altas, con una media de 80.2 y una mediana de 80.5, mientras que la audiencia tiene una media de 73.7 y una mediana de 73.0. Aunque ambos grupos presentan fluctuaciones significativas, las puntuaciones de la audiencia son más dispersas y tienen más valores bajos en comparación con los críticos, quienes son más consistentes y positivos en sus valoraciones. Esto sugiere que los críticos valoran de forma más favorable y con menor variabilidad que la audiencia

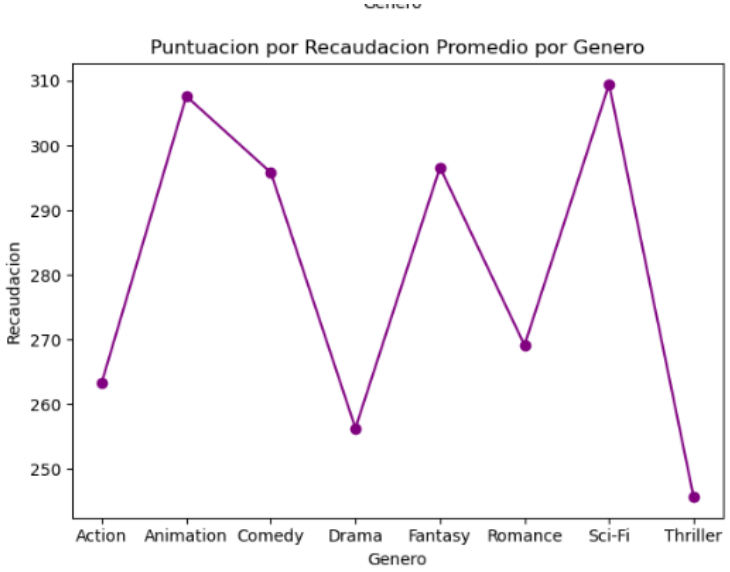
- **Géneros más populares**: Visualizar qué géneros tienden a tener mejores puntuaciones o recaudaciones.

```
#2.2 Géneros más populares
df_agrupado2 = df.groupby('genre').agg({'critic_score': 'mean','audience_score': 'mean', 'box_office_millions': 'mean'}).reset_index()


plt.plot(df_agrupado2['genre'], df_agrupado2['critic_score'], marker='o', linestyle='-', color='pink')
plt.title('Puntuacion de Criticos Promedio por Genero')
plt.xlabel('Genero')
plt.ylabel('Puntuacion')
plt.tight_layout()
plt.show()



plt.plot(df_agrupado2['genre'], df_agrupado2['audience_score'], marker='o', linestyle='-', color='green')
plt.title('Puntuacion de Audiencia Promedio por Genero')
plt.xlabel('Genero')
plt.ylabel('Puntuacion')
plt.tight_layout()
plt.show()



plt.plot(df_agrupado2['genre'], df_agrupado2['box_office_millions'], marker='o', linestyle='-', color='purple')
plt.title('Puntuacion por Recaudacion Promedio por Genero')
plt.xlabel('Genero')
plt.ylabel('Recaudacion')
```

```
plt.tight_layout()
    plt.show()
```







En el primer gráfico, los críticos valoran más alto los géneros "Fantasy" y "Sci-Fi", con puntuaciones alrededor de 84, mientras que "Comedy" y "Thriller" reciben puntuaciones más bajas, cercanas a 78. En el segundo gráfico, las puntuaciones de la audiencia son variables, destacando "Sci-Fi" con una valoración alta (alrededor de 78) y "Comedy" y "Drama" con puntuaciones más bajas (cerca de 70). Finalmente, el tercer gráfico muestra que los géneros "Action" y "Animation" generan mayores recaudaciones promedio, superando los 300 millones, mientras que géneros como "Thriller" y "Drama" tienen recaudaciones menores, cerca de 250 millones.

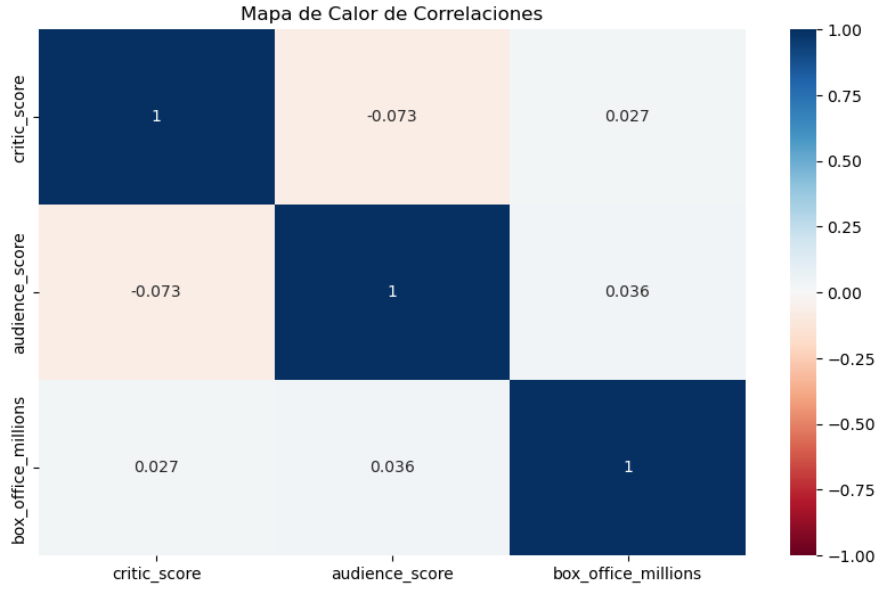
- **Relación entre la recaudación y la puntuación**: ¿Las películas más exitosas financieramente tienden a tener mejores puntuaciones?

#2.3 Relación entre la recaudación y la puntuación: ¿Las películas más exitosas financieramente tienden a tener mejores puntuaciones?

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

correlaciones = {
    'criticos_vs_recaudacion': df['critic_score'].corr(df['box_office_millions']),
    'audiencia_vs_recaudacion': df['audience_score'].corr(df['box_office_millions']),
    'criticos_vs_audiencia': df['critic_score'].corr(df['audience_score'])
}

def interpretar_correlacion(valor):
    if abs(valor) >= 0.7:
```

```
        return "Fuerte"
    elif abs(valor) >= 0.4:
        return "Moderada"
    else:
        return "Débil"

plt.figure(figsize=(10, 6))
sns.heatmap(df[['critic_score', 'audience_score', 'box_office_millions']].corr(),
            annot=True,
            cmap='RdBu',
            vmin=-1,
            vmax=1,
            center=0)
plt.title('Mapa de Calor de Correlaciones')
```

Hay películas muy exitosas financieramente con bajas puntuaciones y películas con excelentes puntuaciones que no generaron grandes ingresos.

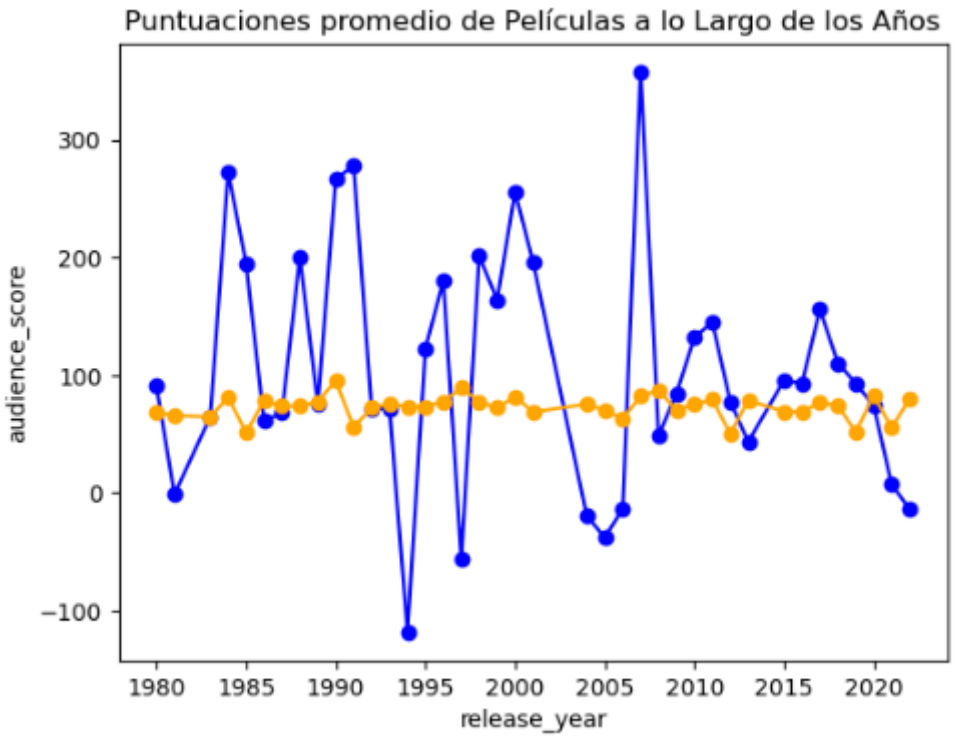


La correlación entre las puntuaciones de los críticos y el éxito financiero es de 0.027, y entre las puntuaciones de la audiencia y los ingresos de taquilla es de 0.036. Estos valores son muy bajos, lo que indica que no existe una relación significativa entre las puntuaciones (tanto de críticos como de audiencia) y el éxito financiero de las películas. Por lo tanto, las películas más exitosas financieramente no necesariamente tienen mejores puntuaciones.

- **Evolución temporal**: Gráficos que muestren cómo ha cambiado la recaudación o las puntuaciones de las películas a lo largo de los años.

#2.4 Evolución temporal: Gráficos que muestren cómo ha cambiado la recaudación o las puntuaciones de las películas a lo largo de los años.

```
df_agrupado = df.groupby('release_year').agg({'Ganancias_Perdidas': 'mean',
'critic_score': 'mean','audience_score': 'mean',}).reset_index()

plt.plot(df_agrupado['release_year'], df_agrupado['Ganancias_Perdidas'],
marker='o', linestyle='-', color='blue')
plt.title('Ganancias promedio de Películas a lo Largo de los Años')
plt.xlabel('release_year')
plt.ylabel('Ganancias_Perdidas')


plt.plot(df_agrupado['release_year'], df_agrupado['audience_score'], marker='o',
linestyle='-', color='orange')
plt.title('Puntuaciones promedio de Películas a lo Largo de los Años')
plt.xlabel('release_year')
plt.ylabel('audience_score')
```



La línea azul representa los puntajes individuales de películas, evidenciando una variabilidad extrema con picos y caídas significativas en diferentes periodos, especialmente en las décadas de los 80 y 2000. Por otro lado, la línea naranja sugiere un promedio general más constante de las puntuaciones de audiencia a lo largo de las décadas, con pocos cambios notables. El gráfico muestra que, aunque algunas películas alcanzan puntajes muy altos o bajos en ciertos años.

3. **Uso de Arrays y Funciones de NumPy**

- Implementar operaciones y cálculos utilizando **arrays de NumPy**. Por ejemplo:
    - Crear un array con las puntuaciones de las películas y otro con las recaudaciones. Luego, calcula la **media**, **máximo**, **mínimo** y **desviación estándar** de ambos arrays.

```
#3. Uso de Arrays y Funciones de NumPy

#Crear un array con las puntuaciones de las películas y otro con las recaudaciones.
    Luego, calcula la media, máximo, mínimo y desviación estándar de ambos arrays.

array_recaudaciones = np.array(df['box_office_millions'])

array_puntuaciones_publico = np.array(df['audience_score'])

array_puntuaciones_criticos = np.array(df['critic_score'])


#Estadisticos

def calcular_estadisticas(array, nombre):

   media = np.mean(array)

   maximo = np.max(array)

   minimo = np.min(array)

   desviacion_estandar = np.std(array)


   print(f"Estadísticas para {nombre}:")

   print(f"Media: {media}")

   print(f"Máximo: {maximo}")

   print(f"Mínimo: {minimo}")

   print(f"Desviación estándar: {desviacion_estandar}")

   print("-" * 30)


# Calcular estadísticas para cada array

calcular_estadisticas(array_recaudaciones, "Recaudaciones")

calcular_estadisticas(array_puntuaciones_publico, "Puntuaciones del público")

    calcular_estadisticas(array_puntuaciones_criticos, "Puntuaciones de críticos")
```

```
    #3. Uso de Arrays y Funciones de NumPy

#Crear un array con las puntuaciones de las películas y otro con las recaudaciones.
    Luego, calcula la media, máximo, mínimo y desviación estándar de ambos arrays.

array_recaudaciones = np.array(df['box_office_millions'])

array_puntuaciones_publico = np.array(df['audience_score'])

array_puntuaciones_criticos = np.array(df['critic_score'])


#Estadisticos

def calcular_estadisticas(array, nombre):

   media = np.mean(array)

   maximo = np.max(array)

   minimo = np.min(array)

   desviacion_estandar = np.std(array)


   print(f"Estadísticas para {nombre}:")

   print(f"Media: {media}")

   print(f"Máximo: {maximo}")

   print(f"Mínimo: {minimo}")

   print(f"Desviación estándar: {desviacion_estandar}")

   print("-" * 30)


# Calcular estadísticas para cada array

calcular_estadisticas(array_recaudaciones, "Recaudaciones")

calcular_estadisticas(array_puntuaciones_publico, "Puntuaciones del público")

calcular_estadisticas(array_puntuaciones_criticos, "Puntuaciones de críticos")
```

Estadísticas para Recaudaciones:

Media: 280.707

Máximo: 495.8

Mínimo: 20.1

Desviación estándar: 137.21139402760983

Estadísticas para Puntuaciones del público:

Media: 73.65

Máximo: 98

Mínimo: 50

Desviación estándar: 14.295016614191114

Estadísticas para Puntuaciones de críticos:

Media: 80.21

Máximo: 99

Mínimo: 60

Desviación estándar: 11.168970409129034

- Usa operaciones vectorizadas de **NumPy** para calcular el **rendimiento** de cada película (recaudación - presupuesto) y comparar el rendimiento medio entre diferentes géneros.

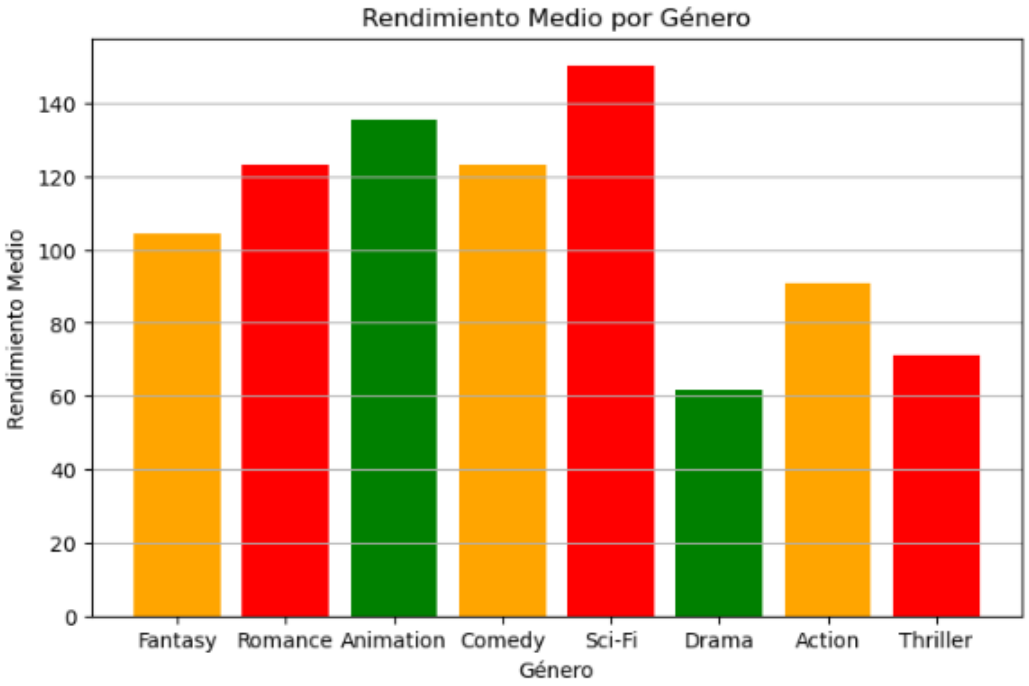
#Usa operaciones vectorizadas de NumPy para calcular el rendimiento de cada película (recaudación - presupuesto) y comparar el rendimiento medio entre diferentes géneros.

#El rendimiento de cada pelicula ya lo teniamos calculado, sin embargo, vamos a realizar este numeral como se propone con los rendimientos que ya fueron calculados previamente y dejamos el codigo escrito por redundancia.

```
array_rendimientos = df['Ganancias_Perdidas']

array_generos= df['genre']


#Diccionario para almacenar medias
```

```
rendimiento_por_genero = {}

#Se calcula la media por genero
for genero in array_generos:
    rendimiento_por_genero[genero] = np.mean(array_rendimientos[array_generos == genero])

#Para ver los datos numericamente
for genero, media in rendimiento_por_genero.items():
    print(f"Rendimiento medio para el género {genero}: {media:.2f}")

#Hacemos grafica para verlos visualmente
plt.figure(figsize=(8, 5))
plt.bar(list(rendimiento_por_genero.keys()), list(rendimiento_por_genero.values()),
color=['orange','red','green'])
plt.title('Rendimiento Medio por Género')
plt.xlabel('Género')
plt.ylabel('Rendimiento Medio')
plt.grid(axis='y')
plt.show()
```

```
Rendimiento medio para el género Fantasy: 104.26
Rendimiento medio para el género Romance: 123.01
Rendimiento medio para el género Animation: 135.59
Rendimiento medio para el género Comedy: 122.97
Rendimiento medio para el género Sci-Fi: 150.17
Rendimiento medio para el género Drama: 61.67
Rendimiento medio para el género Action: 90.68
Rendimiento medio para el género Thriller: 71.29
```



El gráfico muestra el rendimiento promedio de diferentes tipos de películas, con barras de varios colores que representan cada género. El género que mejor rendimiento tiene es el de Ciencia Ficción, seguido por Animación y Romance, lo que indica que estas películas suelen ser más exitosas en términos de taquilla o popularidad. Por otro lado, los géneros que tienen un rendimiento promedio más bajo son Drama y Thriller, lo que sugiere que estas películas no suelen tener tanto éxito como los otros géneros. En cambio, los géneros de Fantasía, Comedia y Acción tienen un rendimiento promedio intermedio, lo que significa que son populares pero no tanto como los primeros. Este análisis resalta qué tipos de películas son más preferidos por el público y cuáles tienen más posibilidades de éxito.