

Análisis exploratorio de datos:

Gym



Introducción a los **datos**



2 Datasets



Datasets

users_history.csv

- **Descripción:** Este archivo rastrea la actividad histórica de los miembros del gimnasio. Registra sus interacciones con las instalaciones y servicios del gimnasio.

users_info.csv

- **Descripción:** Este archivo contiene información estática sobre los miembros del gimnasio, incluyendo sus datos demográficos, detalles de membresía, género, etc.

users_history.csv : 300 000 filas 9 columnas



	A	B	C	D	E	F	G	H	I
1	user_id	gym_id	checkin_time	checkout_time	workout_type	calories_burned	location	gym_type	facilities
2	user_3291	gym_6	2023-09-10 15:55:00	2023-09-10 16:34:00	Weightlifting	462	Philadelphia	Budget	Swimming Pool, Clir
3	user_1944	gym_2	2023-04-13 20:07:00	2023-04-13 22:43:00	Yoga	1278	Los Angeles	Budget	Climbing Wall, Yoga
4	user_958	gym_7	2023-06-10 12:24:00	2023-06-10 13:49:00	Cardio	858	San Antonio	Premium	Sauna, Basketball Co
5	user_811	gym_2	2023-05-23 17:11:00	2023-05-23 20:01:00	Yoga	1134	Los Angeles	Budget	Climbing Wall, Yoga
6	user_4923	gym_10	2023-02-21 6:20:00	2023-02-21 8:02:00	Weightlifting	1049	San Jose	Premium	Swimming Pool, Sau
7	user_1534	gym_5	2023-02-05 19:33:00	2023-02-05 22:18:00	CrossFit	1482	Phoenix	Standard	Basketball Court, Cro
8	user_4431	gym_8	2023-06-10 13:08:00	2023-06-10 14:07:00	Cardio	1618	San Diego	Standard	Basketball Court, Sau
9	user_410	gym_1	2023-03-26 14:04:00	2023-03-26 16:03:00	Yoga	751	New York	Premium	Climbing Wall, Swi
10	user_54		2023-04-06 6:17:00	2023-04-06 9:10:00	Swimming	706	San Jose	Premium	Swimming Pool, Sau
11	user_4797	gym_7	2023-02-15 19:37:00	2023-02-15 21:02:00	Weightlifting	713	San Antonio	Premium	Sauna, Basketball Co

users_info.csv

- 5 000 filas
- 11 columnas



	A	B	C	D	E	F	G	H	I	J	K
1	user_id	first_name	last_name	age	gender	birthdate	sign_up_date	user_location	subscription_plan	price_per_month	features
2	user_1	Chris	Wilson		56 Female	29\$02\$2000	2023-02-06	Denver	Basic	19.99	Access to basic gym
3	user_2	Michael	Miller		46 Non-binary	22\$12\$1978	2023-08-08	Orlando	Pro	49.99	Access to all facilities
4	user_3	Daniel	Smith		32 Female	30\$08\$1962	2021-01-11	Orlando	Basic	19.99	Access to basic gym
5	user_4	David	Smith		60 Male	05\$12\$2003	2023-08-07	Denver	Pro	49.99	Access to all facilities
6	user_5	Chris	Jones		25 Female	25\$08\$2004	2021-01-08	Denver	Basic	19.99	Access to basic gym
7	user_6	Jessica	Rodriguez		38 Male	11\$09\$1969	2022-02-13	Austin	Student	9.99	Access to basic facilities
8	user_7	Linda	Jones		56 Male	09\$07\$1982	2023-07-25	Seattle	Student	9.99	Access to basic facilities
9	user_8	Jane	Brown		36 F	17\$05\$1989	2021-02-01	Atlanta	Pro	49.99	Access to all facilities
10	user_9	Robert	Davis		40 Non-binary	10\$04\$1976	2023-09-11	Orlando	Student	9.99	Access to basic facilities
11	user_10	John	Jones		28 N	30\$10\$2005	2022-06-24	Detroit	Basic	NOT SURE :(Access to basic gym

Detección y corrección de
missings y *outliers*



Detección y corrección de
missings y *outliers*

users_info.csv

Análisis de las Variables

	Problema Inicial	Acciones Tomadas	Resultado
Edad (Age)	Incoherencias entre fechas de nacimiento y edades reportadas.	Recálculo de la edad en el 01/11/2023.	Relación entre edad y planes de suscripción más clara
Género (Gender)	Valores inconsistentes en la variable gender ['Female' 'Non-Binary' 'Male' 'F' 'N' 'M']	Estandarizamos las categorías en: Female ➔ F, Male ➔ M, Non-Binary ➔ N	Ánálisis más claro de la distribución de género por gimnasio y plan.
Precio por Mes (Price per Month)	Valores anómalos como "4999", lo que generaba errores en el análisis.	<ul style="list-style-type: none">Corregimos los valores fuera de rango.Rellenamos valores nulos basándonos en los planes asociados: \$9.99 para Student., \$19.99 para Basic., \$49.99 para Pro.	Precios estandarizados y consistentes
Planes de Suscripción (Subscription_plan)	Plan Student incluía usuarios mayores de 35 años	Reasignamos usuarios mayores de 35 años al plan Basic	Edades coherentes por plan: Student: Edad promedio de 26 años., Basic: Edad promedio de 45 años., Pro: Edad promedio de 40 años.
Características (Features)	Teníamos toda la información en una única cadena de texto	Separamos la columna features en categorías específicas como: <ul style="list-style-type: none">Tipo de acceso.Beneficios para invitados.Beneficios adicionales.	<ul style="list-style-type: none">Relación clara entre las características ofrecidas y el tipo de plan.Pro: Más beneficios exclusivos, como acceso ilimitado y pases para invitados.

Unión de base de datos y transformación de variables



Merge() de los data sets



- Utilizamos **merge()** debido a que ambas tablas comparten una columna en común, **user_id**.
- En comparación con **join()**, **merge()** es más flexible, ya que permite unir 2 tablas a través de columnas en lugar de índices.
- Al unir los datasets terminamos con:
 - 21 columnas
 - 300000 filas

```
[ ] # Cambiamos los valores de las columnas a formato Title Case para tener los mismos formatos en todas las columnas texto
df_history['gym_id'] = df_history['gym_id'].str.title()
df_history['workout_type'] = df_history['workout_type'].str.title()
df_history['location'] = df_history['location'].str.title()
df_history['gym_type'] = df_history['gym_type'].str.title()
df_history['facilities'] = df_history['facilities'].str.title()
df_history['user_id'] = df_history['user_id'].str.title()

[ ] # Aplicamos el Merge
merged_df = pd.merge(df_info, df_history, on = 'user_id' , how = "inner")
merged_df.head(3)
merged_df_copia = merged_df.copy()

[ ] merged_df_copia.columns
Index(['user_id', 'age', 'gender', 'birthdate', 'sign_up_date',
       'user_location', 'subscription_plan', 'new_age', 'new_price',
       'acceso_a', 'tipo_acceso', 'beneficios_invitados',
       'beneficios_adicionales', 'gym_id', 'checkin_time', 'checkout_time',
       'workout_type', 'calories_burned', 'location', 'gym_type',
       'facilities'],
      dtype='object')
```

Limpieza después de `merge()`

- **Gym_type**
 - Se agrupó por 3 categorías:
 - **Budget**: Budget y Economico
 - **Standard**: Normalillo y Standard
 - **Premium**: Premium y Muy Caro
- Cambio de nombre de columna **location** a **gym_location**
- **Gym_id**
 - Identificamos los **gym_location** de cada **gym_id**.
 - Valores nulos fueron reemplazados por **gym_id** donde **gym_location** coincidían.



#	Column	Non-Null Count	Dtype
0	user_id	300000	non-null object
1	age	300000	non-null int64
2	gender	300000	non-null object
3	birthdate	300000	non-null datetime64[ns]
4	sign_up_date	300000	non-null object
5	user_location	300000	non-null object
6	subscription_plan	300000	non-null object
7	new_age	300000	non-null int64
8	new_price	300000	non-null float64
9	acceso_a	300000	non-null object
10	tipo_acceso	300000	non-null object
11	beneficios_invitados	300000	non-null object
12	beneficios_adicionales	202267	non-null object
13	gym_id	299807	non-null object
14	checkin_time	300000	non-null object
15	checkout_time	300000	non-null object
16	workout_type	300000	non-null object
17	calories_burned	299500	non-null float64
18	location	300000	non-null object
19	gym_type	300000	non-null object
20	facilities	299801	non-null object

Limpieza después de `merge()`

- **User_id y gym_location**
 - Para usuarios que van a varios gimnasios, determinamos cuál gym visitan más utilizando la moda.
 - Reemplazamos los otros gimnasios que visitan por el gym más frecuente.
- **Calories_burned**
 - Valores nulos fueron reemplazados por la mediana de cada deporte y por género.
 - Nueva columna ***calories_burned_new*** fue rellenada con los valores faltantes.
 - ***Calories_burned*** fue eliminada.

#	Column	Non-Null Count	Dtype
0	user_id	300000	non-null object
1	age	300000	non-null int64
2	gender	300000	non-null object
3	birthdate	300000	non-null datetime64[ns]
4	sign_up_date	300000	non-null object
5	user_location	300000	non-null object
6	subscription_plan	300000	non-null object
7	new_age	300000	non-null int64
8	new_price	300000	non-null float64
9	acceso_a	300000	non-null object
10	tipo_acceso	300000	non-null object
11	beneficios_invitados	300000	non-null object
12	beneficios_adicionales	202267	non-null object
13	gym_id	299807	non-null object
14	checkin_time	300000	non-null object
15	checkout_time	300000	non-null object
16	workout_type	300000	non-null object
17	calories_burned	299500	non-null float64
18	location	300000	non-null object
19	gym_type	300000	non-null object
20	facilities	299801	non-null object

Limpieza después de `merge()`

- **Facilities**

- Se observó que algunas personas practican deportes en gimnasios que no cuentan con las instalaciones necesarias. Estas filas fueron eliminadas.

- **Sign_up_date**

- Filas donde el **checkin_time** fue antes que la fecha de inscripción de la persona fueron eliminadas.

- **Checkin_time**

- Mantuvimos la primera ocurrencia de usuarios que hicieron checkin más de una vez al mismo tiempo.

RangeIndex: 300000 entries, 0 to 299999 Data columns (total 21 columns):			
#	Column	Non-Null Count	Dtype
0	user_id	300000	non-null object
1	age	300000	non-null int64
2	gender	300000	non-null object
3	birthdate	300000	non-null datetime64[ns]
4	sign_up_date	300000	non-null object
5	user_location	300000	non-null object
6	subscription_plan	300000	non-null object
7	new_age	300000	non-null int64
8	new_price	300000	non-null float64
9	acceso_a	300000	non-null object
10	tipo_acceso	300000	non-null object
11	beneficios_invitados	300000	non-null object
12	beneficios_adicionales	202267	non-null object
13	gym_id	299807	non-null object
14	checkin_time	300000	non-null object
15	checkout_time	300000	non-null object
16	workout_type	300000	non-null object
17	calories_burned	299500	non-null float64
18	location	300000	non-null object
19	gym_type	300000	non-null object
20	facilities	299801	non-null object

Nuevas columnas después de limpiar



```
merged_df_copia.info()  
  
→ <class 'pandas.core.frame.DataFrame'>  
RangeIndex: 300000 entries, 0 to 299999  
Data columns (total 21 columns):  
 # Column Non-Null Count Dtype  
 ---  
 0 user_id 300000 non-null object  
 1 age 300000 non-null int64  
 2 gender 300000 non-null object  
 3 birthdate 300000 non-null datetime64[ns]  
 4 sign_up_date 300000 non-null object  
 5 user_location 300000 non-null object  
 6 subscription_plan 300000 non-null object  
 7 new_age 300000 non-null int64  
 8 new_price 300000 non-null float64  
 9 acceso_a 300000 non-null object  
 10 tipo_acceso 300000 non-null object  
 11 beneficios_invitados 300000 non-null object  
 12 beneficios_adicionales 202267 non-null object  
 13 gym_id 299807 non-null object  
 14 checkin_time 300000 non-null object  
 15 checkout_time 300000 non-null object  
 16 workout_type 300000 non-null object  
 17 calories_burned 299500 non-null float64  
 18 location 300000 non-null object  
 19 gym_type 300000 non-null object  
 20 facilities 299801 non-null object  
  
dtypes: datetime64[ns](1), float64(2), int64(2), object(16)  
memory usage: 48.1+ MB
```



```
merged_df_copia.info()  
  
→ <class 'pandas.core.frame.DataFrame'>  
Index: 150544 entries, 25 to 60080  
Data columns (total 26 columns):  
 # Column Non-Null Count Dtype  
 ---  
 0 user_id 150544 non-null object  
 1 age 150544 non-null int64  
 2 gender 150544 non-null object  
 3 birthdate 150544 non-null datetime64[ns]  
 4 sign_up_date 150544 non-null datetime64[ns]  
 5 user_location 150544 non-null object  
 6 subscription_plan 150544 non-null object  
 7 new_age 150544 non-null int64  
 8 new_price 150544 non-null float64  
 9 acceso_a 150544 non-null object  
 10 tipo_acceso 150544 non-null object  
 11 beneficios_invitados 150544 non-null object  
 12 beneficios_adicionales 101380 non-null object  
 13 checkin_time 150544 non-null datetime64[ns]  
 14 checkout_time 150544 non-null datetime64[ns]  
 15 workout_type 150544 non-null object  
 16 date 150544 non-null object  
 17 gym_location 150544 non-null object  
 18 gym_id 150544 non-null object  
 19 facilities 150544 non-null object  
 20 gym_type_categorizado 150544 non-null object  
 21 calories_burned_new 150544 non-null float64  
 22 checkin_month 150544 non-null int32  
 23 checkin_year 150544 non-null int32  
 24 sign_year 150544 non-null int32  
 25 sign_month 150544 non-null int32  
  
dtypes: datetime64[ns](4), float64(2), int32(4), int64(2), object(14)  
memory usage: 28.7+ MB
```

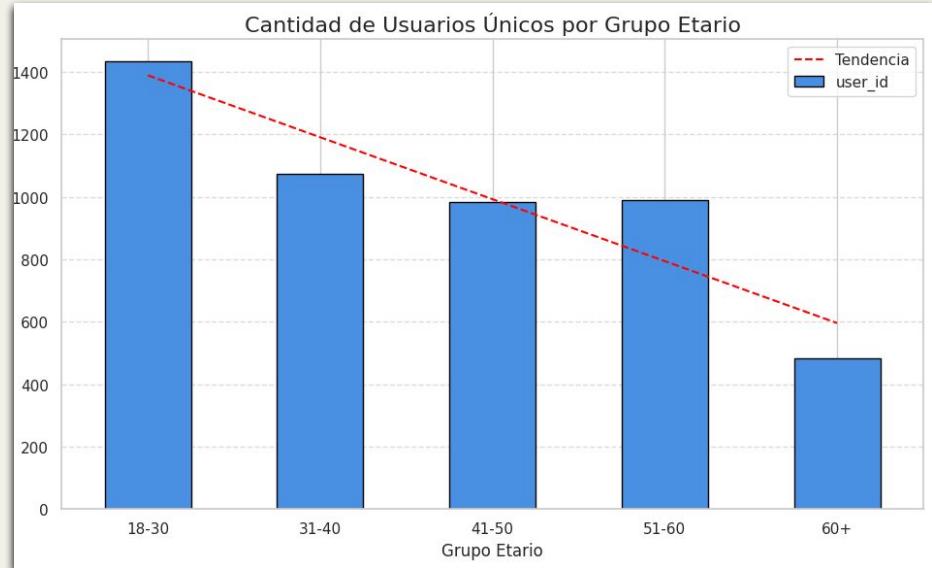
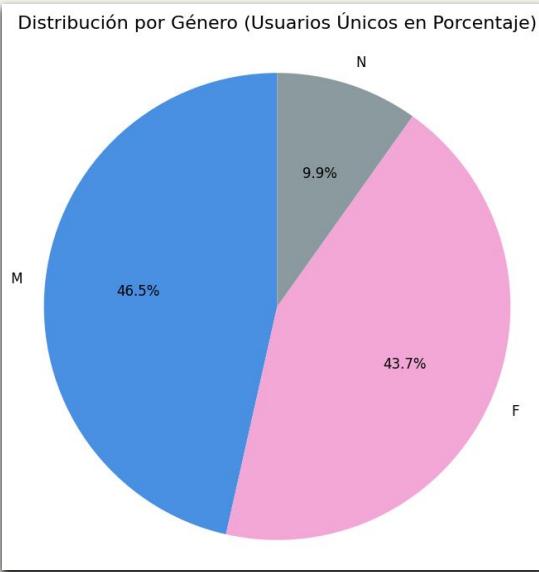
Exploración de variables numéricas y categóricas



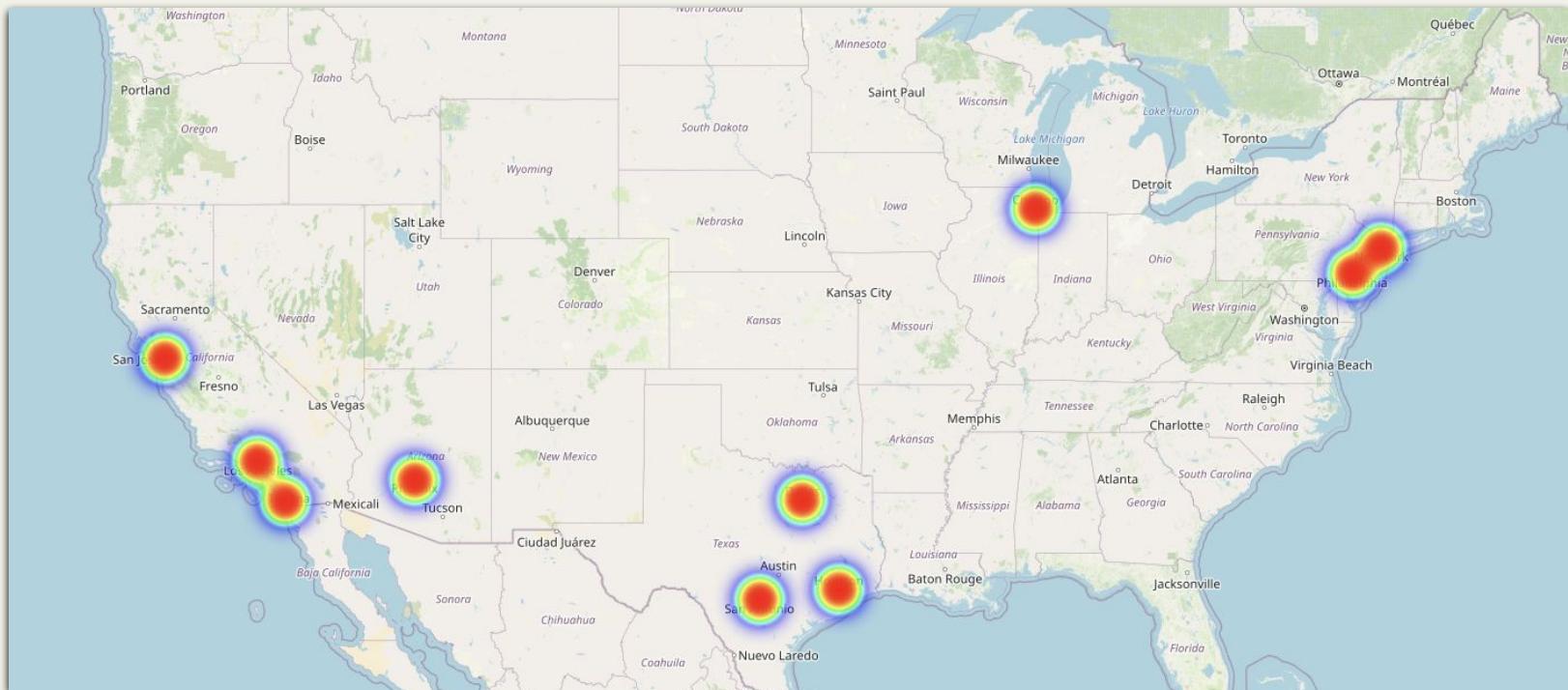


- ¿Hay diferencias entre los usuarios de distintos gyms/ tarifas?
 - ¿Qué actividades son las más realizadas?
 - ¿Cuál es el tiempo medio de entreno?
 - ¿Hay diferencias según el perfil del usuario?
 - ¿Hay diferencias entre meses del año?
-

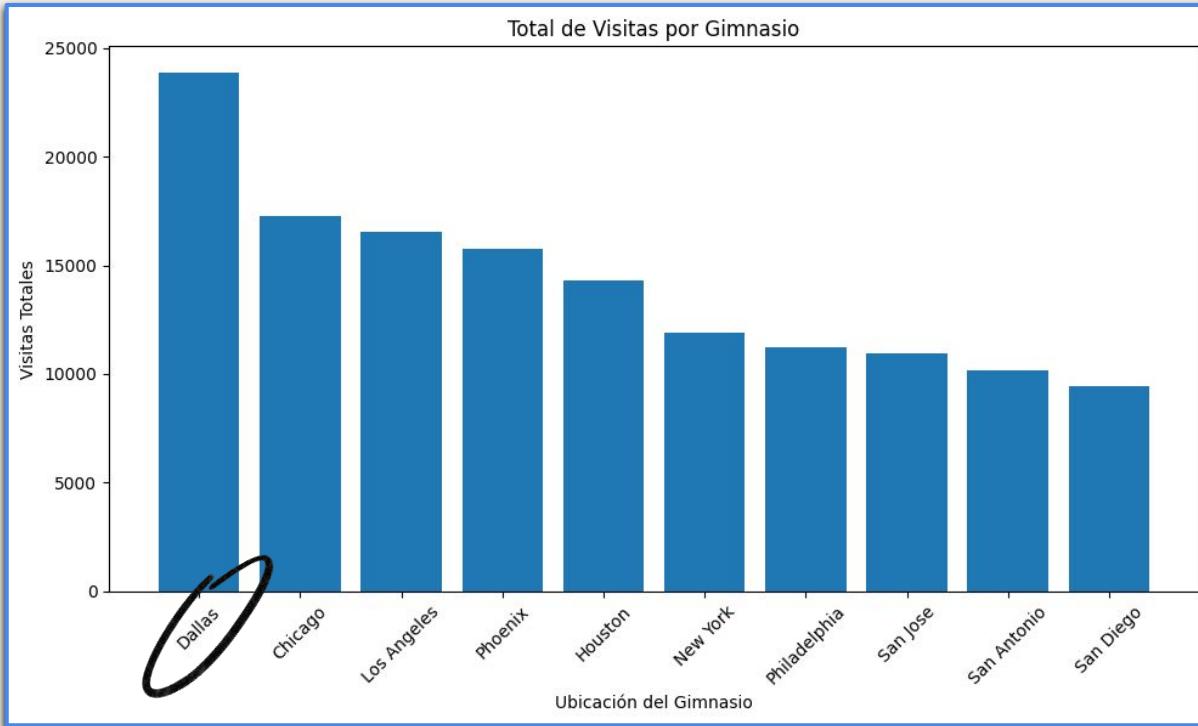
Análisis demográfico



gym_location

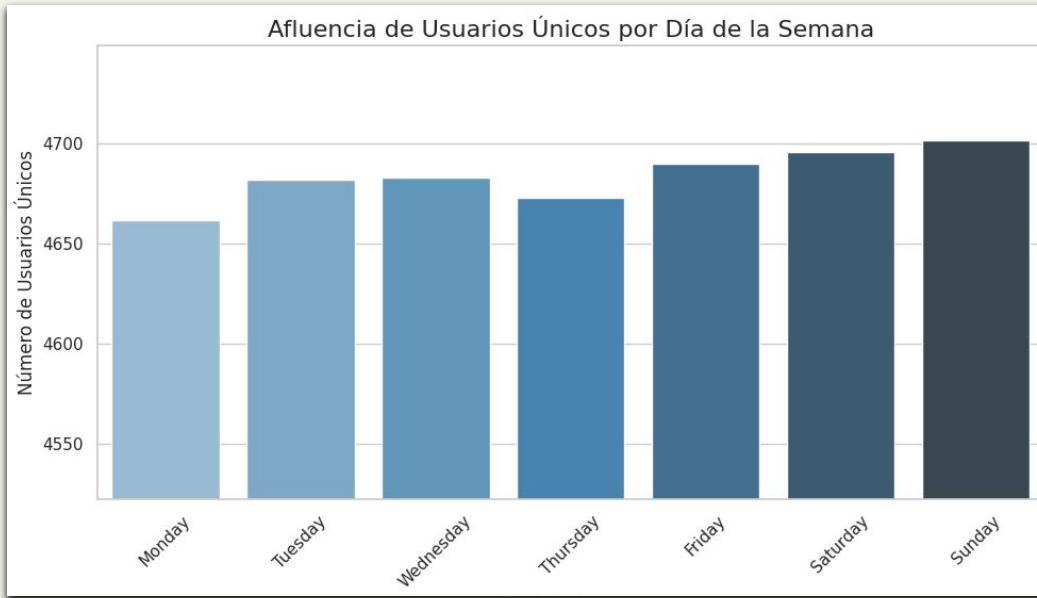


gym_location

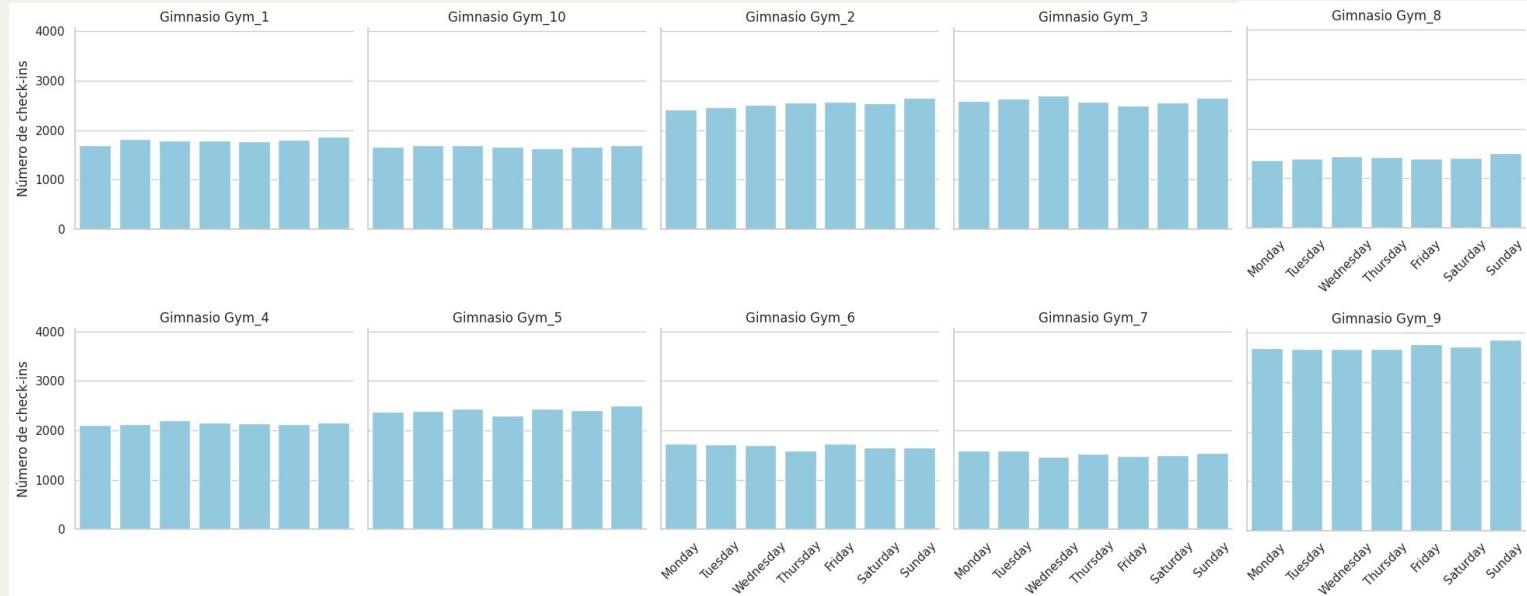


Análisis de suscripciones

Afluencia según el día de la semana

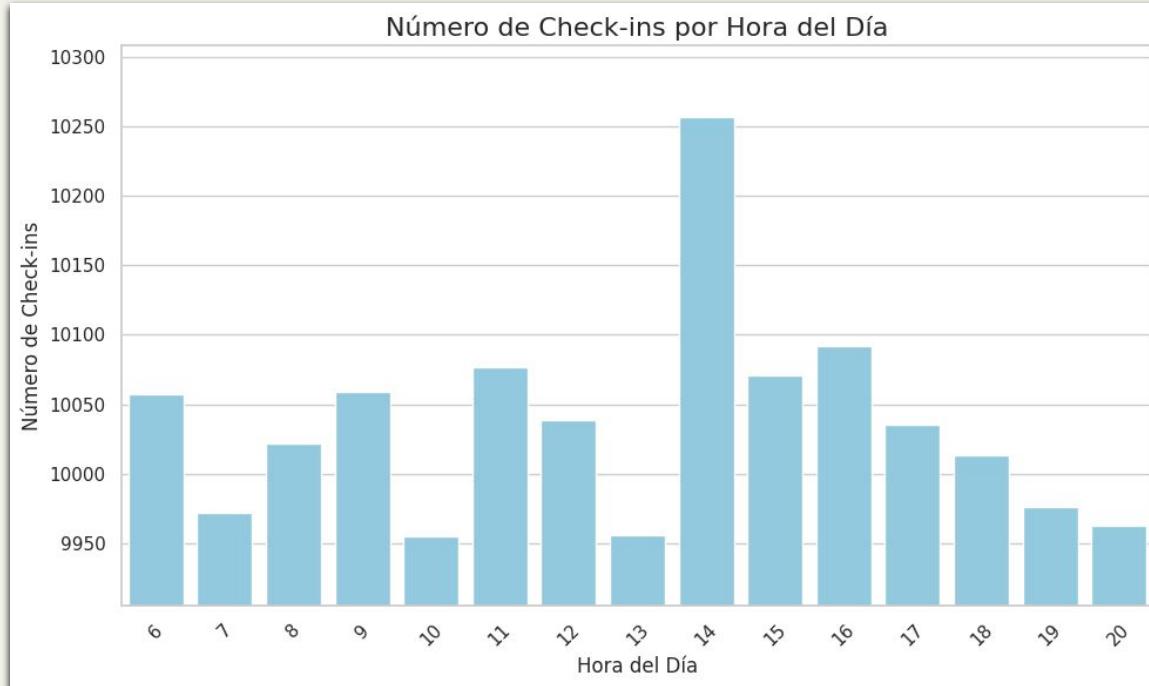


Análisis de suscripciones



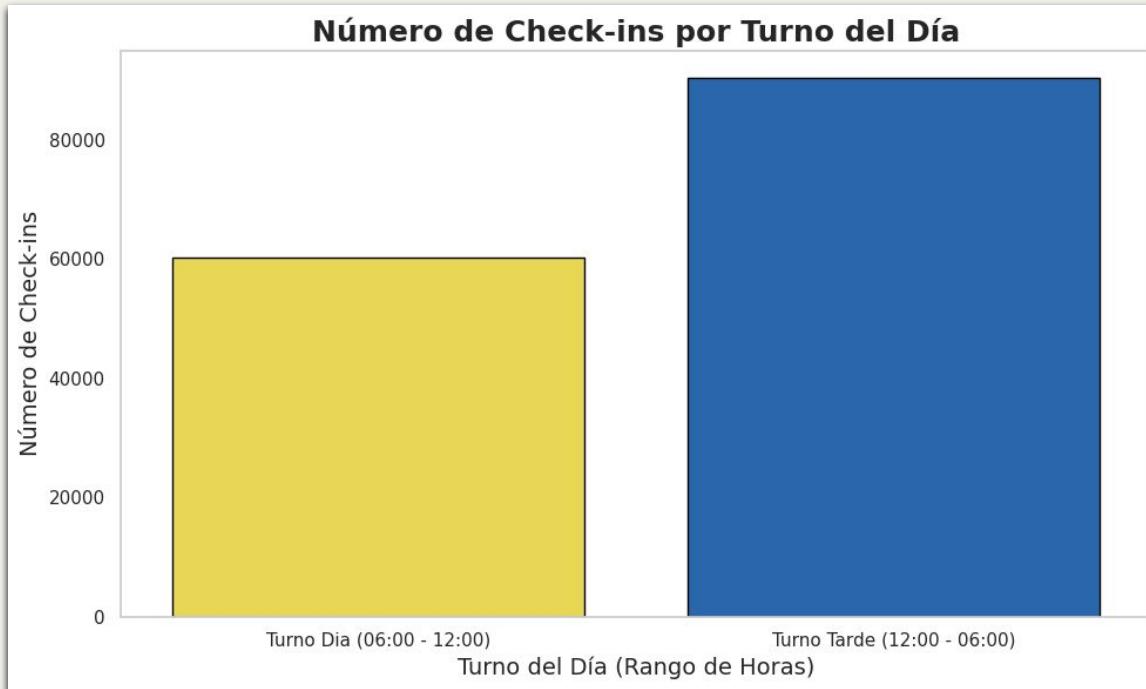
Análisis de suscripciones

Afluencia según la hora del día



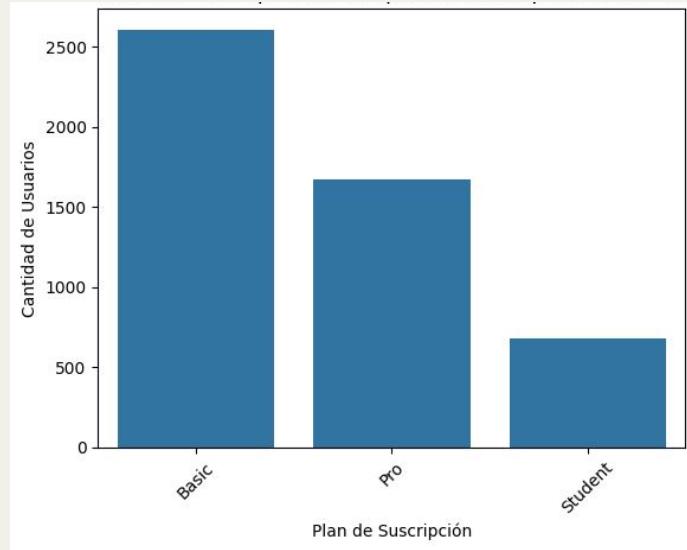
Análisis de suscripciones

Afluencia según la franja horaria

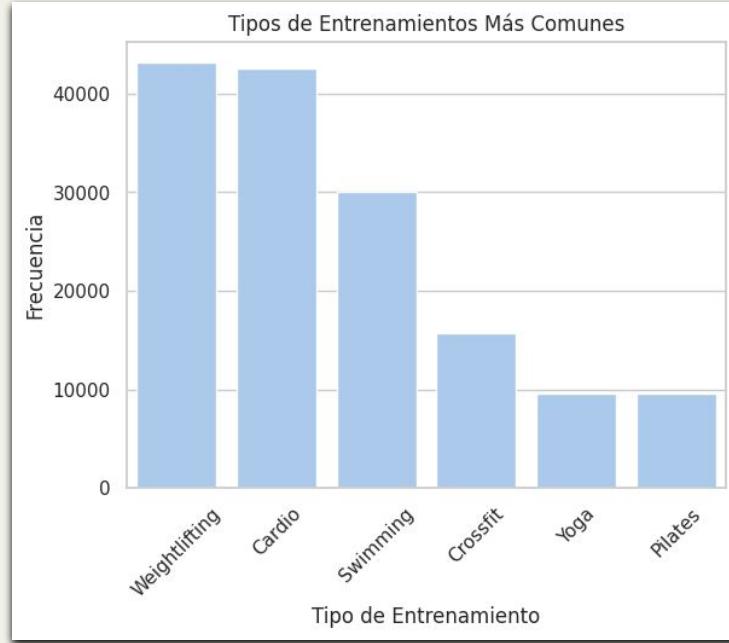


Análisis de suscripciones

¿Cuál es el plan de suscripción más popular?

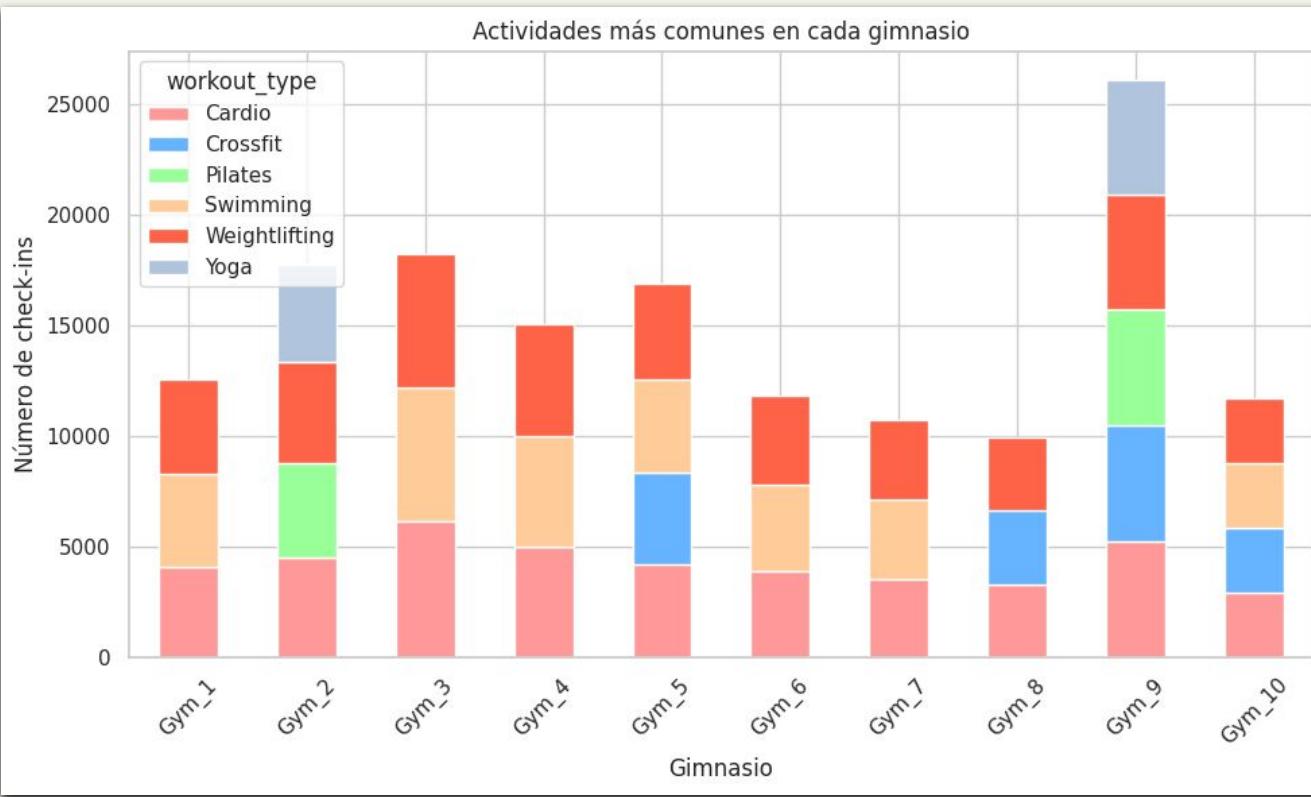


Análisis de suscripciones



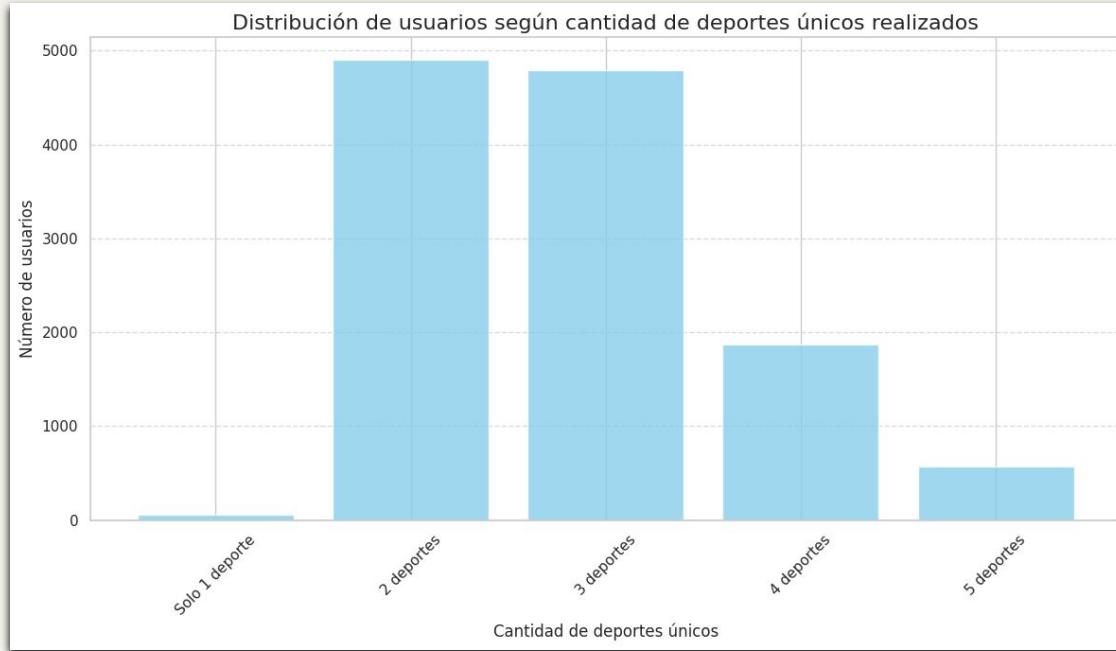
Los entrenamientos más populares son **Weightlifting** y **Cardio**, mientras el **Yoga** y **Pilates** son menos comunes.





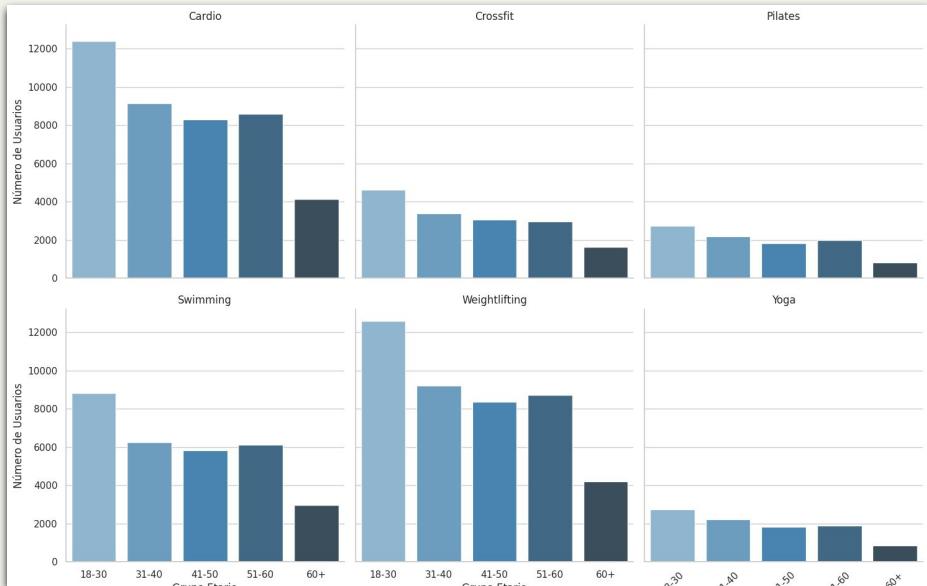
Podemos observar que solo hay dos gimnasios que ofrecen **Pilates** y **Yoga**. También, vemos que 4 ofrecen **Crossfit** y que la mayoría de los gimnasios tienen **piscinas**. Además, todos los gimnasios ofrecen **Cardio** y **Weightlifting**.

Análisis de suscripciones

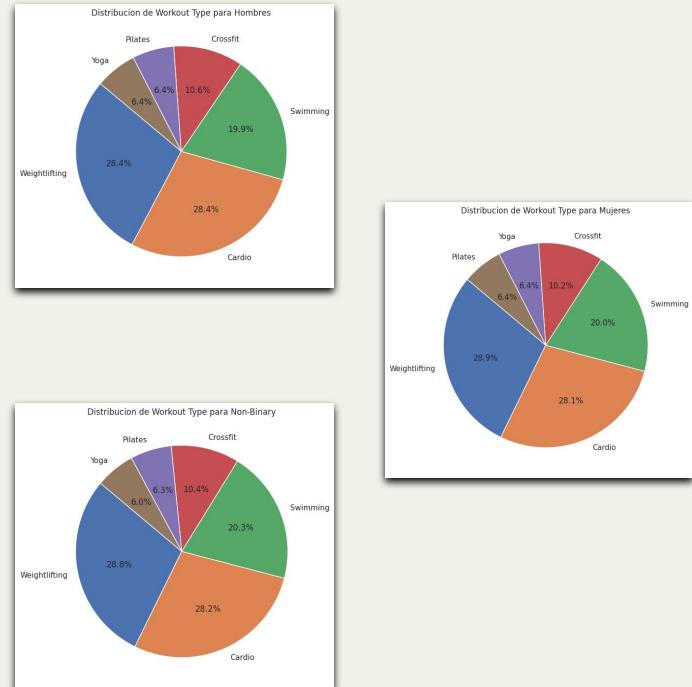


Análisis de suscripciones

Tipo de entrenamiento por edad



Tipo de entrenamiento por género



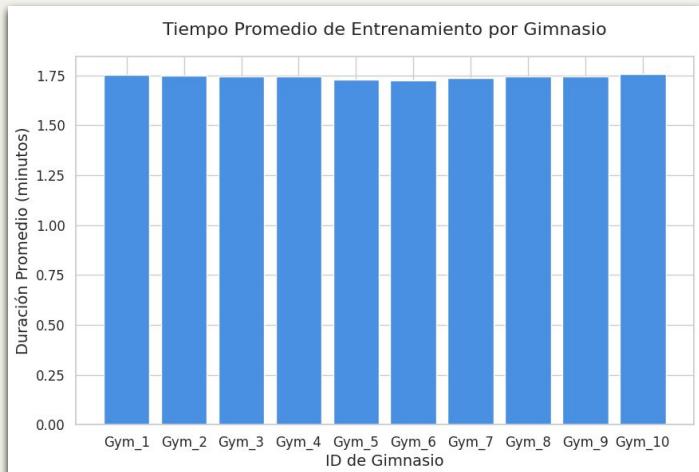
Duración de cada sesión de entrenamiento

Hemos creado la columna '**workout_duration**', que representa la duración de cada sesión de entrenamiento.

Esta información se deriva de la diferencia entre el `checkin_time` y el `checkout_time`.

	workout_duration
workout_type	
Cardio	0 days 01:44:34.651780467
Crossfit	0 days 01:44:07.774874115
Pilates	0 days 01:44:50.502559816
Swimming	0 days 01:44:38.539621260
Weightlifting	0 days 01:44:22.314935064
Yoga	0 days 01:44:44.956122022

dtype: timedelta64[ns]



Análisis de tendencias temporales

■ Análisis del comportamiento teniendo en cuenta las estaciones del año.

Teniendo en cuenta que en los Estados Unidos, las estaciones del año son:

- Primavera: de marzo a mayo.
- Verano: de junio a agosto.
- Otoño: de septiembre a noviembre.
- Invierno: de diciembre a febrero.

Se ha decidido agrupar de esa manera los datos.

1. Por suscripciones
2. Registro de visitas



```

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

# Asegurarnos de que 'sign_up_date' sea de tipo datetime
merged_df_copia['sign_up_date'] = pd.to_datetime(merged_df_copia['sign_up_date'])

# Creamos una columna de 'estación' basada en el mes de 'sign_up_date'
def assign_season(month):
    if month in [12, 1, 2]:
        return 'Invierno'
    elif month in [3, 4, 5]:
        return 'Primavera'
    elif month in [6, 7, 8]:
        return 'Verano'
    else:
        return 'Otoño'

# Asignamos la estación a cada registro
merged_df_copia['season'] = merged_df_copia['sign_up_date'].dt.month.apply(assign_season)

# Agrupamos por estación para contar el número de suscripciones por estación
season_subscription_counts = merged_df_copia['season'].value_counts()

# Asegurarnos de que las estaciones estén en el orden correcto
season_order = ['Primavera', 'Verano', 'Otoño', 'Invierno']
season_subscription_counts = season_subscription_counts[season_order]

# Colores representativos de cada estación
season_colors = {
    'Primavera': '#88C784', # Verde suave
    'Verano': '#FFB300', # Amarillo dorado
    'Otoño': '#FF7043', # Naranja cálido
    'Invierno': '#1E3ASF' # Azul oscuro
}

# Creamos una figura con un tamaño atractivo
plt.figure(figsize=(8, 6))

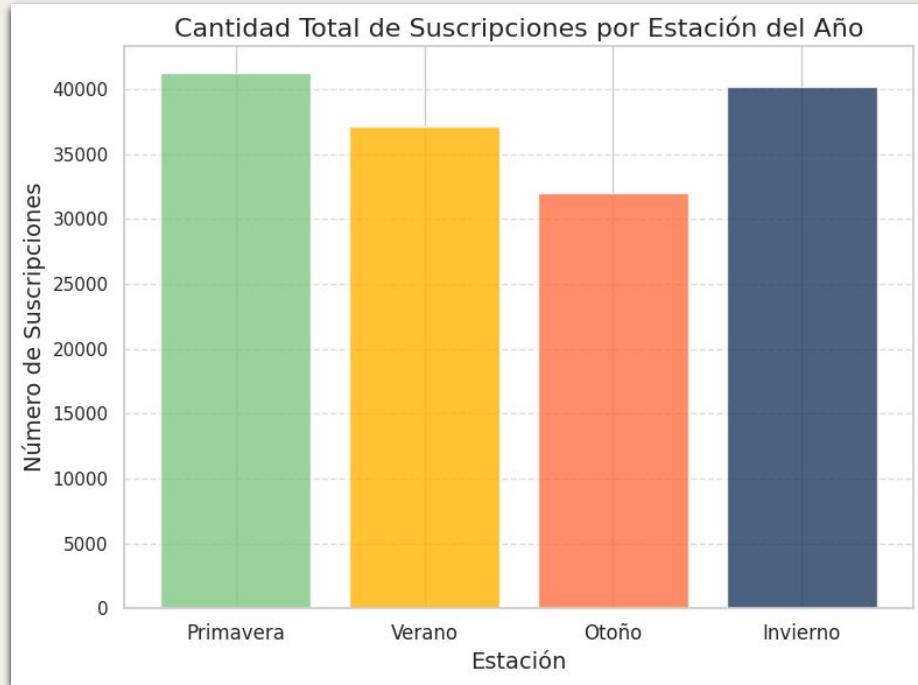
# Creamos un gráfico de barras con colores representativos de cada estación
plt.bar(season_subscription_counts.index, season_subscription_counts.values,
        color=[season_colors[season] for season in season_subscription_counts.index], alpha=0.8)

# Ajustamos títulos y etiquetas
plt.title('Cantidad Total de Suscripciones por Estación del Año', fontsize=16)
plt.xlabel('Estación', fontsize=14)
plt.ylabel('Número de Suscripciones', fontsize=14)

# Mejoramos la visualización de las etiquetas
plt.xticks(fontsize=12)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()

# Mostramos el gráfico
plt.show()

```



Observamos que en Otoño hay un menor número de suscripciones y la primavera es la estación del año donde hay más suscripciones.

```
monthly_activity = merged_df_copia.groupby('sign_month')['user_id'].count()  
monthly_activity.plot(kind='line', figsize=(10, 6))  
plt.title('Evolución Mensual de Incripciones al gym a durante el año')  
plt.show()
```



Se observa que las personas se inscriben **más al comienzo del año** quizás por la motivación de un nuevo año.

```

import matplotlib.pyplot as plt
import pandas as pd

# Asegurarnos de que 'checkin_time' sea de tipo datetime
merged_df_copia['checkin_time'] = pd.to_datetime(merged_df_copia['checkin_time'])

# Creamos una columna de 'estación' basada en el mes de 'checkin_time'
def assign_season(month):
    if month in [12, 1, 2]:
        return 'Invierno'
    elif month in [3, 4, 5]:
        return 'Primavera'
    elif month in [6, 7, 8]:
        return 'Verano'
    else:
        return 'Otoño'

# Asignamos la estación a cada registro
merged_df_copia['season'] = merged_df_copia['checkin_time'].dt.month.apply(assign_season)

# Contamos la cantidad total de check-ins por estación
season_checkin_counts = merged_df_copia['season'].value_counts().reindex(['Invierno', 'Primavera', 'Verano', 'Otoño'])

# Mostramos los resultados
print("Cantidad total de check-ins por estación:")
print(season_checkin_counts)

# Colores representativos de cada estación
season_colors = {
    'Primavera': '#8C1784', # Verde suave
    'Verano': '#FFB300', # Amarillo dorado
    'Otoño': '#FF7043', # Naranja cálido
    'Invierno': '#1E3A5F' # Azul oscuro
}

# Creamos una figura con un tamaño atractivo
plt.figure(figsize=(8, 6))

# Creamos un gráfico de barras con colores representativos de cada estación
plt.bar(season_checkin_counts.index, season_checkin_counts.values,
        color=[season_colors[season] for season in season_checkin_counts.index], alpha=0.8)

# Ajustamos títulos y etiquetas
plt.title('Cantidad Total de Check-ins por Estación del Año', fontsize=16)
plt.xlabel('Estación', fontsize=14)
plt.ylabel('Número de Check-ins', fontsize=14)

# Mejoramos la visualización de las etiquetas
plt.xticks(fontsize=12)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()

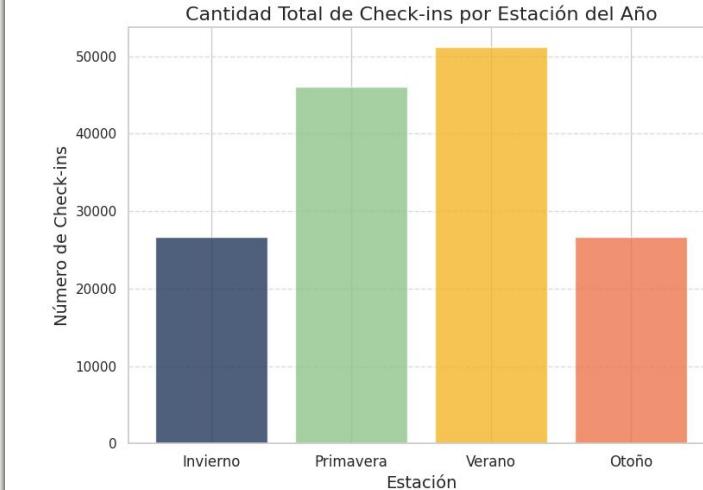
# Mostramos el gráfico
plt.show()

```

```

Cantidad total de check-ins por estación:
season
Invierno      26638
Primavera     46041
Verano       51225
Otoño        26640
Name: count, dtype: int64

```



Observamos que las personas visitan más el gimnasio durante el verano en comparación con invierno y otoño.

Conclusiones clave



CONCLUSION 1

Monitorear la evolución de las edades: Es importante realizar análisis periódicos para verificar si los perfiles de edad se mantienen estables o si hay cambios significativos.

CONCLUSION 2

Registros de visitas: Solo están disponibles desde enero hasta octubre de 2023. Esto significa que las extracción de datos se realizó en Noviembre del 2023. Esto es una limitación temporal importante.

CONCLUSION 3

Hábitos de ejercicio: la disminución de la inscripción en gimnasios a medida que aumenta la edad es un fenómeno complejo con múltiples factores causales. También los problemas de salud crónicos pueden limitar la capacidad de realizar ejercicios intensos en un gimnasio.

Este data set genera más preguntas.



1. ¿Qué tipo de plataforma o sistema generó estos datos?
2. ¿Existen eventos o campañas específicas que podrían haber influido en el número de visitas durante el año?
3. ¿Existen datos sobre el tipo de lesiones o enfermedades crónicas más comunes entre los miembros más mayores del gimnasio?
4. ¿Hay información sobre el nivel de actividad física antes de unirse al gimnasio?
5. ¿Cuál es la frecuencia con la que estos usuarios visitan el gimnasio fuera de su estado?
6. ¿Se han considerado factores como los viajes de negocios o las mudanzas recientes al analizar estos datos?

FAQs



ANY QUESTIONS?

Merci!
Thank You!
Gracias!

