# Abstracción

La abstracción en ciencias y matemáticas es un proceso mental crucial que permite simplificar la complejidad de la realidad, enfocándose en los elementos esenciales de un fenómeno, sistema u objeto, mientras se desestiman detalles que pueden ser irrelevantes para el análisis. Este enfoque no solo facilita la creación de modelos simplificados, sino que también permite a los investigadores realizar generalizaciones que son fundamentales para el avance del conocimiento. En el contexto del análisis de datos, la abstracción se traduce en la capacidad de construir representaciones que preserven la esencia de los datos mientras se eliminan las variaciones irrelevantes, lo cual es vital para obtener insights significativos y patrones que guíen la toma de decisiones.

La capacidad de operar en diferentes niveles de abstracción, desde los modelos más detallados hasta las simplificaciones generales, es especialmente valiosa en el análisis de datos. Aquí, los investigadores pueden crear capas de abstracción que encapsulan solo los detalles críticos necesarios para cada nivel de análisis, promoviendo una comprensión más clara y efectiva de los datos.

# Cohorte

El término "cohorte" proviene del latín "cohors, cohortis" y en la antigua Roma designaba a una unidad militar compuesta por varias centurias. En la actualidad, se ha expandido para referirse a cualquier grupo de individuos que comparten una o varias características en común, lo que resulta esencial para la investigación longitudinal. Este concepto permite a los investigadores observar y analizar el comportamiento de grupos homogéneos a lo largo del tiempo, controlando factores que pueden influir en los resultados.

## Aplicaciones del Concepto de Cohorte

Las cohortes son herramientas poderosas en diversas disciplinas, desde la epidemiología hasta el análisis de mercado. En la salud pública, por ejemplo, la identificación de cohortes específicas permite evaluar cómo diferentes factores de riesgo afectan a grupos particulares de población. Además, en el ámbito educativo, las cohortes pueden ser utilizadas para investigar el rendimiento académico y las tasas de graduación de grupos específicos de estudiantes.

## Tipos de Cohortes

1. **Cohorte de Nacimiento**: Agrupa individuos nacidos en el mismo año o periodo, lo que facilita el análisis de generaciones y sus cambios en actitudes, comportamiento y salud a lo largo del tiempo. Esto es útil para entender cómo los eventos históricos y las políticas sociales impactan a generaciones específicas.

2. **Cohorte de Exposición**: Incluye a personas que han estado expuestas a un mismo factor, como un medicamento o un agente ambiental. Este tipo de cohorte es crucial para estudios que buscan identificar efectos adversos o beneficios de tratamientos específicos.

3. **Cohorte de Comportamiento o Intervención**: Estudia grupos que han experimentado una intervención o que siguen un mismo patrón conductual,

permitiendo observar cómo estas experiencias influyen en sus comportamientos o resultados de salud.

# Análisis de Cohortes

El análisis de cohortes es una metodología de investigación longitudinal que se centra en observar cómo variables específicas afectan los resultados a lo largo del tiempo dentro de un grupo determinado. Este enfoque permite comparar la incidencia de resultados entre un grupo expuesto a un factor de interés y un grupo no expuesto, facilitando la comprensión de posibles relaciones causales. Su aplicación se ha vuelto indispensable en la ciencia de datos y el análisis de clientes en empresas, donde se busca entender el comportamiento de segmentos específicos a lo largo del tiempo.

## Tipos de Análisis de Cohortes

1. **Cohorte Prospectiva**: Se observa el desarrollo de resultados en una cohorte desde el presente hacia el futuro. Este enfoque es fundamental para detectar la aparición de nuevos eventos o enfermedades, midiendo el efecto de factores de riesgo en tiempo real.

2. **Cohorte Retrospectiva**: Utiliza datos de eventos que ya ocurrieron, permitiendo realizar inferencias sobre el impacto de ciertos factores utilizando información pasada. Este tipo de análisis es eficaz cuando se dispone de datos históricos bien documentados, como en investigaciones de salud pública.

3. **Cohorte Fija vs. Dinámica**: Una cohorte fija se mantiene constante, mientras que una dinámica permite la incorporación de nuevos participantes, lo que es útil para estudios demográficos y poblacionales a largo plazo.

## Ventajas del Análisis de Cohortes

1. **Relación Causa-Efecto**: Permite estudiar posibles relaciones causales observando los efectos de una exposición a lo largo del tiempo, lo que es vital en la investigación científica y médica.

2. **Medición de Incidencia de Eventos Raros**: Útil para estudiar enfermedades o comportamientos poco frecuentes al observar amplias muestras durante largos períodos, lo que facilita la identificación de patrones de salud pública.

3. **Estudio de Múltiples Resultados y Variables**: Permite analizar múltiples resultados y factores, capturando relaciones complejas entre variables y ofreciendo un análisis multifacético del comportamiento o las condiciones de la cohorte.

## Desventajas del Análisis de Cohortes

1. **Costos y Tiempos Extensos**: Los estudios de cohorte requieren financiamiento significativo y largos periodos de seguimiento, lo que puede limitar su viabilidad en algunos contextos.

2. **Pérdida de Seguimiento y Sesgo**: Es posible que los participantes abandonen el estudio con el tiempo, lo que puede introducir sesgos en los resultados y afectar la validez de las conclusiones.

3. **Representatividad Limitada**: La selección de la cohorte inicial puede no ser representativa de la población general, lo que afecta la generalización de los resultados a otros grupos.

# Aplicaciones en el Análisis de Datos

El análisis de cohortes se utiliza en múltiples sectores para estudiar tendencias a lo largo del tiempo y el comportamiento de grupos específicos. Algunas aplicaciones relevantes incluyen:

1. **Marketing y Análisis de Clientes**: Las empresas utilizan el análisis de cohortes para estudiar la retención de clientes y el comportamiento de consumo. Esto permite segmentar a los clientes según su lealtad, identificar patrones de compra y abandono, y desarrollar estrategias personalizadas que aumenten la fidelización.

2. **Desarrollo de Productos y Mejora Continua**: Las empresas tecnológicas utilizan el análisis de cohortes para monitorear la aceptación de nuevos productos y la satisfacción del usuario a lo largo del tiempo. Este análisis ayuda a identificar mejoras o cambios necesarios en función de la interacción de los usuarios con distintas versiones del producto, guiando el desarrollo de características que aumenten la retención de usuarios.

3. **Salud Pública y Epidemiología**: En epidemiología, se utiliza el análisis de cohortes para observar la incidencia y prevalencia de enfermedades en grupos expuestos a factores de riesgo. Esto permite identificar patrones de transmisión y evaluar medidas de prevención. Además, el análisis de cohortes se utiliza para evaluar la efectividad de intervenciones de salud pública, facilitando el diseño de políticas basadas en evidencia.

4. **Educación**: Permite el análisis de rendimiento académico en estudiantes, ayudando a identificar factores clave que influyen en el éxito académico. Las instituciones educativas pueden analizar cohortes de estudiantes desde su ingreso hasta la graduación para optimizar políticas y estrategias de apoyo, diseñando programas de intervención específicos para estudiantes en riesgo de deserción.

5. **Finanzas y Comportamiento del Consumidor**: En finanzas, permite analizar comportamientos de clientes en productos financieros como préstamos, tarjetas de crédito y cuentas de ahorro. Esto ayuda a personalizar estrategias de retención y mejorar la experiencia del cliente, así como a identificar patrones de riesgo en el comportamiento de pagos.

## Aplicaciones Avanzadas del Análisis de Cohortes en Ciencia de Datos

1. **Análisis de la Vida Útil de los Clientes (Customer Lifetime Value - CLV)**: A través del análisis de cohortes, se puede evaluar el valor potencial de clientes a lo largo de su relación con la empresa. Esto permite optimizar los esfuerzos de retención y

personalizar ofertas que maximicen el CLV en función de las características y patrones de comportamiento de las cohortes.

2. **Segmentación de Usuarios en Aplicaciones y Juegos**: En el sector de tecnología y entretenimiento, el análisis de cohortes ayuda a identificar patrones de uso entre diferentes generaciones de usuarios o versiones de aplicaciones. Esto es útil para detectar oportunidades de mejoras en el producto o para redirigir los esfuerzos de marketing hacia los usuarios que tienen mayor probabilidad de permanecer activos.

3. **Predicción de Churn o Abandono**: Al estudiar cohortes de clientes en función de su comportamiento, se pueden identificar patrones que predicen la probabilidad de abandono. Estas predicciones permiten a las empresas implementar estrategias proactivas de retención, tales como promociones especiales o personalización de servicios para retener a clientes en riesgo de abandono.

4. **Análisis de Cohortes en Redes Sociales**: Los algoritmos de redes sociales aplican análisis de cohortes para segmentar y comprender mejor el comportamiento de los usuarios, optimizando la segmentación publicitaria y el contenido compartido. Además, ayuda a personalizar la experiencia del usuario y aumentar la relevancia de las recomendaciones de contenido o contactos.

5. **Análisis de Efectividad de Campañas de Marketing Digital**: Las cohortes en campañas de marketing digital permiten medir el retorno de inversión (ROI) al observar a los clientes captados en distintas campañas y su comportamiento a lo largo del tiempo. Esto ayuda a entender qué tácticas son más efectivas en términos de conversión y retención, permitiendo ajustes y optimización en tiempo real.

## Proceso para Llevar a Cabo un Análisis de Cohortes en Análisis de Datos

1. **Definición de la Cohorte**: Identificar y establecer los criterios de inclusión de la cohorte, asegurándose de que los individuos compartan características relevantes para el estudio.
2. **Selección y Preparación de Datos**: Recopilar y limpiar los datos relevantes de la cohorte, asegurando precisión en variables clave como características demográficas, exposición y resultados.
3. **Segmentación Temporal**: Definir periodos temporales de interés, que pueden ser semanales, mensuales, trimestrales o anuales, dependiendo del enfoque del estudio.
4. **Cálculo de Métricas Relevantes**: Calcular métricas como tasas de incidencia, mortalidad, retención, recurrencia de compra o interacción, que aporten valor al análisis en cada período y para cada grupo.
5. **Análisis Comparativo entre Grupos y Periodos**: Comparar las métricas obtenidas en distintos periodos o entre grupos de cohortes para identificar tendencias, patrones y efectos de variables.
6. **Interpretación de Resultados**: Evaluar los patrones y tendencias identificados en función de los objetivos del estudio, analizando posibles relaciones causa-efecto y considerando limitaciones o sesgos.

7. **Visualización y Comunicación de Hallazgos**: Presentar los resultados del análisis de cohortes de manera clara y comprensible, utilizando gráficos, tablas y otros medios visuales si es necesario, y discutir las implicaciones prácticas de los hallazgos.

El análisis de cohortes permite a los investigadores y analistas segmentar los datos en función de factores temporales y específicos de los grupos estudiados, obteniendo insights clave para la toma de decisiones estratégicas. Este enfoque es especialmente útil para identificar y medir tendencias longitudinales, realizar proyecciones y mejorar el diseño de productos, políticas y servicios en sectores tan diversos como el comercio, la salud, la educación y las finanzas.

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

# Crear un DataFrame de ejemplo con datos
data = {
    'ID_Cliente': [
        1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
        1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
        1, 3, 5, 6, 8, 9, 10, 2, 4, 6,
        2, 4, 7, 1, 5, 9, 3, 6, 7, 8
    ],
    'Fecha_Compra': [
        '2019-01', '2019-03', '2019-05', '2019-01', '2019-03',
        '2019-02', '2019-04', '2019-06', '2019-02', '2019-04',
        '2019-03', '2019-05', '2019-01', '2019-03', '2019-05',
        '2019-04', '2019-06', '2019-02', '2019-04', '2019-06',
        '2019-05', '2019-01', '2019-03', '2019-05', '2019-01',
        '2019-06', '2019-02', '2019-04', '2019-06', '2019-02',
        '2020-01', '2020-03', '2020-05', '2020-01', '2020-03',
        '2020-02', '2020-04', '2020-06', '2020-02', '2020-04'
    ],
    'Monto_Compra': [
        100, 150, 200, 80, 120, 160, 90, 100, 110, 200,
        250, 300, 90, 400, 450, 500, 600, 650, 700, 750,
        200, 180, 190, 260, 310, 120, 210, 300, 350, 400,
        80, 90, 250, 230, 340, 220, 290, 370, 330, 290
    ],    'Canal_Venta': [
        'Online', 'Online', 'Tienda', 'Tienda', 'Online',
        'Tienda', 'Online', 'Online', 'Tienda', 'Tienda',
        'Online', 'Online', 'Tienda', 'Tienda', 'Online',
        'Tienda', 'Tienda', 'Online', 'Online', 'Tienda',
        'Online', 'Tienda', 'Tienda', 'Online', 'Online',
        'Tienda', 'Tienda', 'Online', 'Online', 'Tienda',
        'Online', 'Tienda', 'Tienda', 'Online', 'Online',
        'Tienda', 'Tienda', 'Online', 'Online', 'Tienda'
    ]
}

# Crear DataFrame
```

```
df = pd.DataFrame(data)

# Visualizar el DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40 entries, 0 to 39
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   ID_Cliente     40 non-null     int64
 1   Fecha_Compra   40 non-null     object
 2   Monto_Compra   40 non-null     int64
 3   Canal_Venta    40 non-null     object
dtypes: int64(2), object(2)
memory usage: 1.4+ KB
```

```
# Convertir las fechas a tipo datetime
df['Fecha_Compra'] =
pd.to_datetime(df['Fecha_Compra']).dt.to_period('M')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40 entries, 0 to 39
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   ID_Cliente     40 non-null     int64
 1   Fecha_Compra   40 non-null     period[M]
 2   Monto_Compra   40 non-null     int64
 3   Canal_Venta    40 non-null     object
dtypes: int64(2), object(1), period[M](1)
memory usage: 1.4+ KB
```

# Cohortes

En el contexto del análisis de cohortes, una cohorte se refiere a un grupo de clientes que comparten una característica o experiencia común en un período de tiempo determinado, generalmente definido por la fecha de su primera interacción con el producto o servicio. Por ejemplo, una cohorte puede ser el grupo de clientes que se registraron en un sitio web durante un mes específico.

```
# Calcular el mes de la primera compra de cada cliente y asignarlo
como la Cohorte
df['Cohorte'] = df.groupby('ID_Cliente')
['Fecha_Compra'].transform('min')
df.head()
```

```
   ID_Cliente Fecha_Compra  Monto_Compra Canal_Venta  Cohorte
0           1      2019-01           100      Online  2019-01
1           2      2019-03           150      Online  2019-03
2           3      2019-05           200      Tienda  2019-01
3           4      2019-01            80      Tienda  2019-01
4           5      2019-03           120      Online  2019-03
```

# Tasa de Retención

La tasa de retención es una métrica clave utilizada en análisis de cohortes y gestión de clientes para medir la lealtad y la retención de los clientes a lo largo del tiempo. Se define como la proporción de clientes que continúan utilizando un producto o servicio en períodos de tiempo posteriores en comparación con el número total de clientes en un período inicial.

La fórmula para calcular la tasa de retención es:

$$\text{Tasa de Retención} = \frac{\text{Clientes Retenidos}}{\text{Clientes Iniciales}}$$

```
# Calcular la cantidad de clientes de cada cohorte en cada mes
cohort_sizes_monthly = df.groupby(['Cohorte',
'Fecha_Compra']).agg(Num_Clientes=('ID_Cliente',
'nunique')).reset_index()
cohort_sizes_monthly
```

```
    Cohorte Fecha_Compra  Num_Clientes
0   2019-01      2019-01             4
1   2019-01      2019-02             1
2   2019-01      2019-03             2
3   2019-01      2019-05             2
4   2019-01      2019-06             2
5   2019-01      2020-01             1
6   2019-01      2020-03             1
7   2019-01      2020-04             2
8   2019-02      2019-02             3
9   2019-02      2019-04             3
10  2019-02      2019-05             1
11  2019-02      2019-06             2
12  2019-02      2020-02             1
13  2019-02      2020-06             1
14  2019-03      2019-03             2
15  2019-03      2019-04             1
16  2019-03      2019-05             2
17  2019-03      2020-01             1
18  2019-03      2020-03             1
19  2019-04      2019-04             1
20  2019-04      2019-06             1
21  2019-04      2020-02             1
22  2019-04      2020-05             1
```

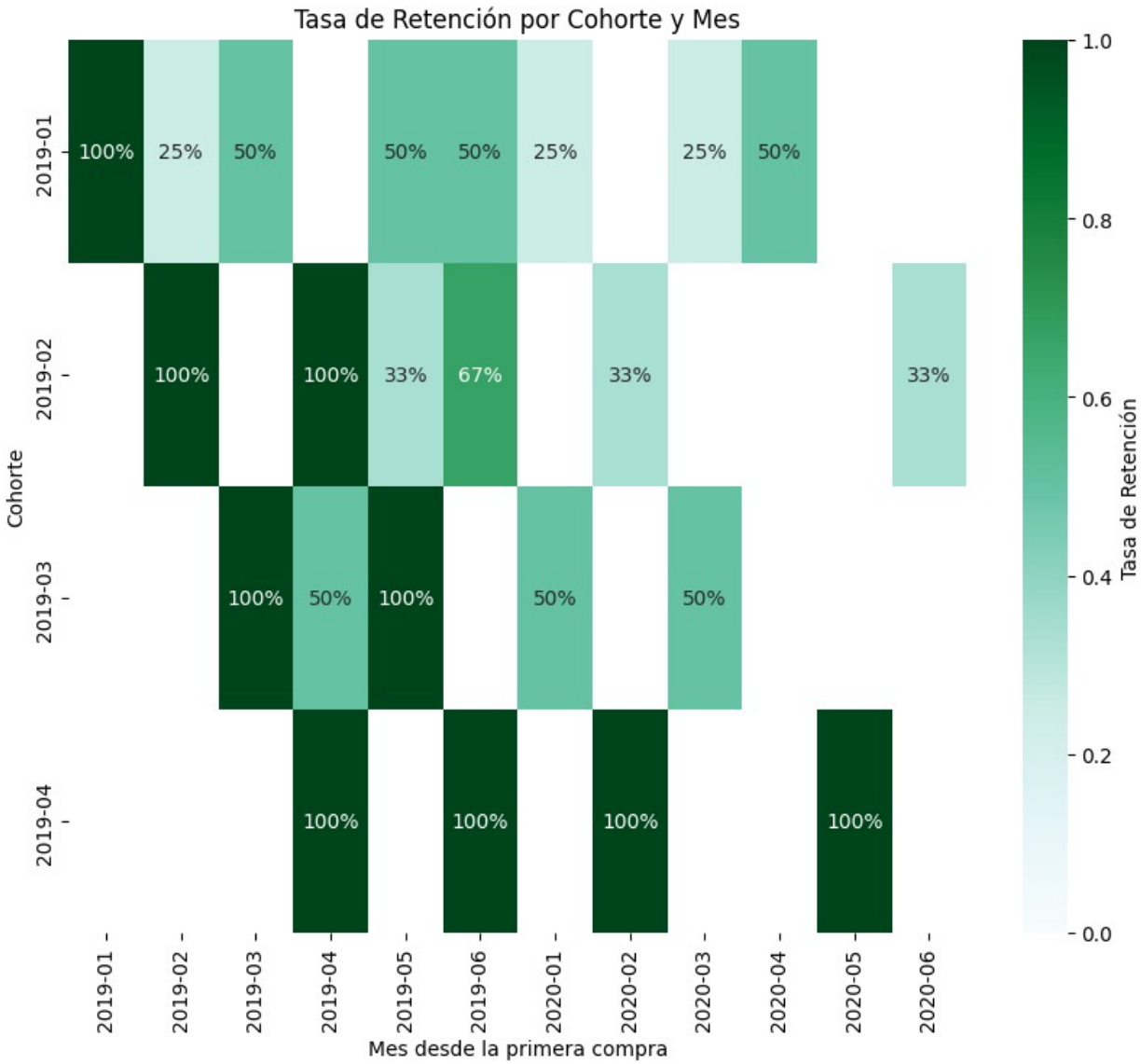
```
# Calcular la cantidad de clientes en la primera compra de cada cohorte
cohort_sizes_initial = cohort_sizes_monthly.groupby('Cohorte').first()
cohort_sizes_initial
```

```
        Fecha_Compra  Num_Clientes
Cohorte
2019-01      2019-01             4
2019-02      2019-02             3
2019-03      2019-03             2
2019-04      2019-04             1
```

```
# Calcular la tasa de retención
retention = cohort_sizes_monthly.pivot(index='Cohorte',
columns='Fecha_Compra',
values='Num_Clientes').divide(cohort_sizes_initial['Num_Clientes'],
axis=0)
retention
```

```
Fecha_Compra  2019-01  2019-02  2019-03  2019-04   2019-05   2019-06
2020-01  \
Cohorte

2019-01           1.0     0.25      0.5      NaN  0.500000  0.500000
0.25
2019-02           NaN     1.00      NaN      1.0  0.333333  0.666667
NaN
2019-03           NaN      NaN      1.0      0.5  1.000000       NaN
0.50
2019-04           NaN      NaN      NaN      1.0       NaN  1.000000
NaN

Fecha_Compra   2020-02  2020-03  2020-04  2020-05   2020-06
Cohorte
2019-01            NaN     0.25      0.5      NaN       NaN
2019-02       0.333333      NaN      NaN      NaN  0.333333
2019-03            NaN     0.50      NaN      NaN       NaN
2019-04       1.000000      NaN      NaN      1.0       NaN
```

```
# Graficar la tasa de retención
plt.figure(figsize=(10, 8))
plt.title('Tasa de Retención por Cohorte y Mes')
sns.heatmap(data=retention, annot=True, fmt='.0%', vmin=0.0, vmax=1.0,
cmap='BuGn', cbar_kws={'label': 'Tasa de Retención'})
plt.xlabel('Mes desde la primera compra')
plt.xticks(rotation=90)
plt.ylabel('Cohorte')
plt.show()
```




- La tasa de retención se calcula típicamente como el número de clientes que realizan una acción específica (como una compra) en un período de tiempo posterior (por ejemplo, el segundo mes, tercer mes, etc.) dividido por el número total de clientes en la cohorte inicial.
- Una tasa de retención alta indica que una gran proporción de clientes continúa utilizando el producto o servicio a lo largo del tiempo, lo que sugiere una mayor lealtad y satisfacción del cliente. Por otro lado, una tasa de retención baja puede indicar problemas de satisfacción del cliente, competencia o cambios en las necesidades del mercado.

# Calcular la tasa de abandono

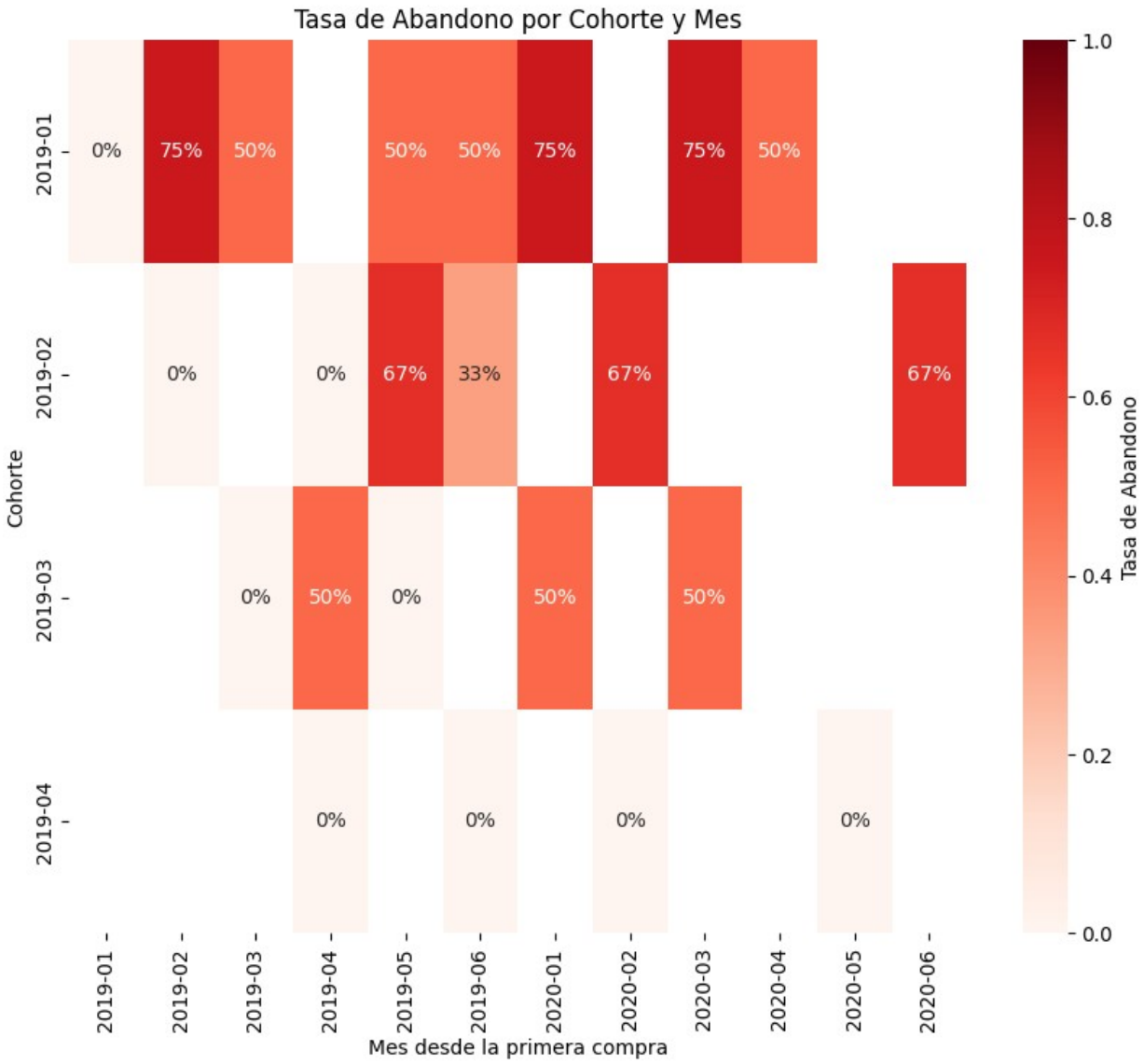
La tasa de abandono se refiere al porcentaje de clientes que dejan de interactuar con el producto o servicio. Se puede calcular como:

Tasa de Abandono = 1 − Tasa de Retención

```
abandon = 1 - retention
abandon
```

```
Fecha_Compra  2019-01  2019-02  2019-03  2019-04   2019-05   2019-06
2020-01  \
Cohorte

2019-01           0.0     0.75      0.5      NaN  0.500000  0.500000
0.75
2019-02           NaN     0.00      NaN      0.0  0.666667  0.333333
NaN
2019-03           NaN      NaN      0.0      0.5  0.000000       NaN
0.50
2019-04           NaN      NaN      NaN      0.0       NaN  0.000000
NaN

Fecha_Compra   2020-02  2020-03  2020-04  2020-05   2020-06
Cohorte
2019-01            NaN     0.75      0.5      NaN       NaN
2019-02       0.666667      NaN      NaN      NaN  0.666667
2019-03            NaN     0.50      NaN      NaN       NaN
2019-04       0.000000      NaN      NaN      0.0       NaN
```

```
# Graficar la tasa de abandono
plt.figure(figsize=(10, 8))
plt.title('Tasa de Abandono por Cohorte y Mes')
sns.heatmap(data=abandon, annot=True, fmt='.0%', vmin=0.0, vmax=1.0,
cmap='Reds', cbar_kws={'label': 'Tasa de Abandono'})
plt.xlabel('Mes desde la primera compra')
plt.xticks(rotation=90)
plt.ylabel('Cohorte')
plt.show()
```




# Frecuencia de Compra

La frecuencia de compra mide cuántas veces en promedio los clientes de una cohorte realizan compras. Se puede calcular con la siguiente fórmula:

$$\text{Frecuencia Media de Compra} = \frac{\text{Total de Compras}}{\text{Clientes en la Cohorte}}$$

```
# Calcular la frecuencia de compras por cohorte y mes
frecuencia_media = df.groupby(['Cohorte',
'Fecha_Compra']).agg(Total_Compras=('ID_Cliente',
'count')).reset_index()
frecuencia_media =
frecuencia_media.merge(cohort_sizes_monthly.groupby('Cohorte')
```

```
['Num_Clientes'].first().reset_index(), on='Cohorte')
frecuencia_media
```

```
    Cohorte Fecha_Compra  Total_Compras  Num_Clientes
0   2019-01      2019-01              5             4
1   2019-01      2019-02              1             4
2   2019-01      2019-03              2             4
3   2019-01      2019-05              2             4
4   2019-01      2019-06              2             4
5   2019-01      2020-01              1             4
6   2019-01      2020-03              1             4
7   2019-01      2020-04              2             4
8   2019-02      2019-02              4             3
9   2019-02      2019-04              3             3
10  2019-02      2019-05              1             3
11  2019-02      2019-06              2             3
12  2019-02      2020-02              1             3
13  2019-02      2020-06              1             3
14  2019-03      2019-03              3             2
15  2019-03      2019-04              1             2
16  2019-03      2019-05              2             2
17  2019-03      2020-01              1             2
18  2019-03      2020-03              1             2
19  2019-04      2019-04              1             1
20  2019-04      2019-06              1             1
21  2019-04      2020-02              1             1
22  2019-04      2020-05              1             1
```

```
frecuencia_media['Frecuencia_Media'] =
frecuencia_media['Total_Compras'] / frecuencia_media['Num_Clientes']
frecuencia_media
```

```
    Cohorte Fecha_Compra  Total_Compras  Num_Clientes
Frecuencia_Media
0   2019-01      2019-01              5             4
1.250000
1   2019-01      2019-02              1             4
0.250000
2   2019-01      2019-03              2             4
0.500000
3   2019-01      2019-05              2             4
0.500000
4   2019-01      2019-06              2             4
0.500000
5   2019-01      2020-01              1             4
0.250000
6   2019-01      2020-03              1             4
0.250000
7   2019-01      2020-04              2             4
0.500000
```

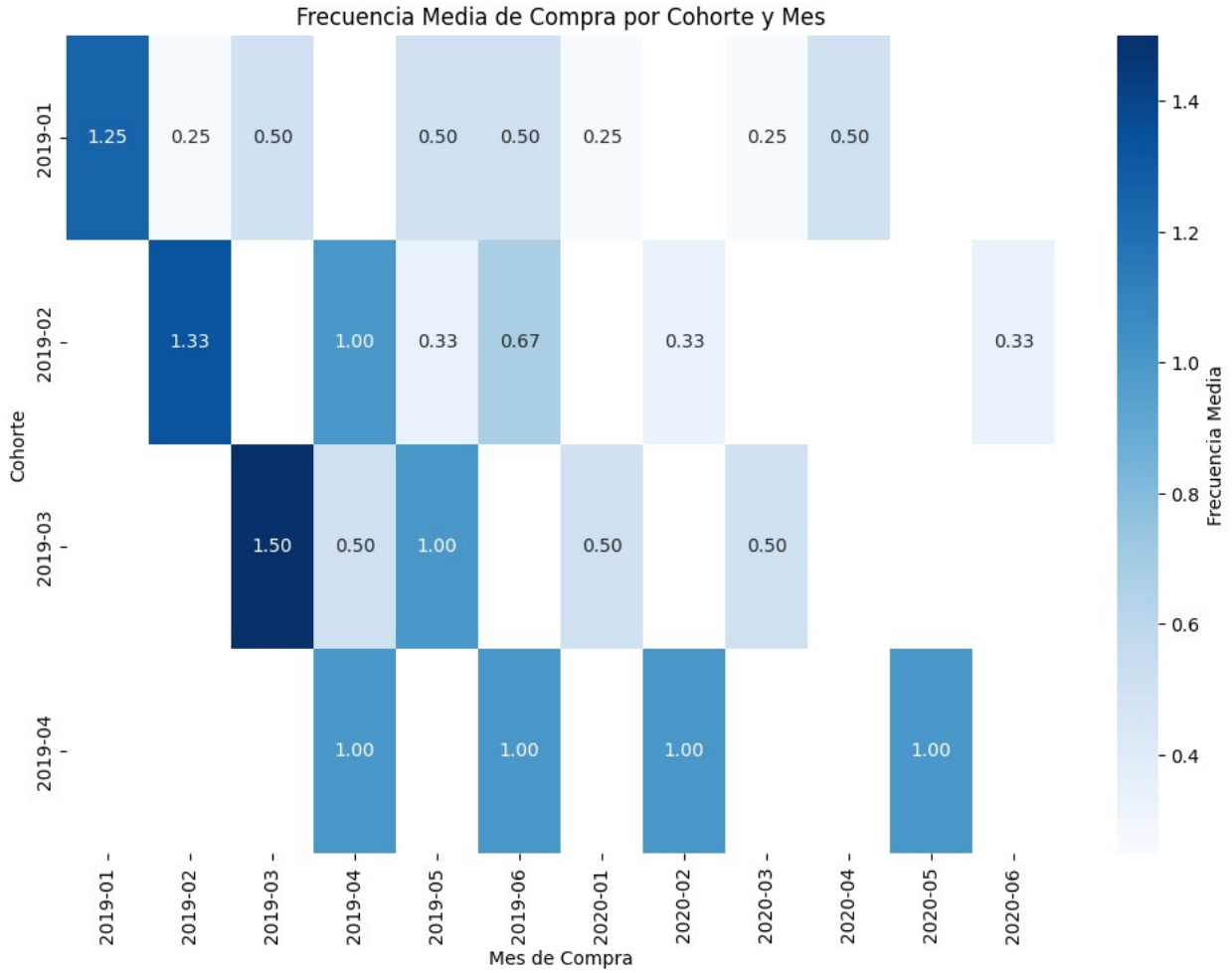
```
8   2019-02      2019-02                4             3
1.333333
9   2019-02      2019-04                3             3
1.000000
10  2019-02      2019-05                1             3
0.333333
11  2019-02      2019-06                2             3
0.666667
12  2019-02      2020-02                1             3
0.333333
13  2019-02      2020-06                1             3
0.333333
14  2019-03      2019-03                3             2
1.500000
15  2019-03      2019-04                1             2
0.500000
16  2019-03      2019-05                2             2
1.000000
17  2019-03      2020-01                1             2
0.500000
18  2019-03      2020-03                1             2
0.500000
19  2019-04      2019-04                1             1
1.000000
20  2019-04      2019-06                1             1
1.000000
21  2019-04      2020-02                1             1
1.000000
22  2019-04      2020-05                1             1
1.000000
```

```
# Crear una tabla pivotante para la frecuencia media de compra
frecuencia_heatmap = frecuencia_media.pivot_table(index='Cohorte',
columns='Fecha_Compra', values='Frecuencia_Media')
frecuencia_heatmap
```

```
Fecha_Compra  2019-01   2019-02  2019-03  2019-04   2019-05   2019-06
\
Cohorte

2019-01          1.25  0.250000      0.5      NaN  0.500000  0.500000

2019-02           NaN  1.333333      NaN      1.0  0.333333  0.666667

2019-03           NaN       NaN      1.5      0.5  1.000000       NaN

2019-04           NaN       NaN      NaN      1.0       NaN  1.000000


Fecha_Compra  2020-01   2020-02  2020-03  2020-04  2020-05   2020-06
```

```
Cohorte
2019-01          0.25       NaN     0.25       0.5       NaN       NaN
2019-02           NaN  0.333333      NaN       NaN       NaN  0.333333
2019-03          0.50       NaN     0.50       NaN       NaN       NaN
2019-04           NaN  1.000000      NaN       NaN       1.0       NaN
```

```python
# Graficar frecuencia media de compra como heatmap
plt.figure(figsize=(12, 8))
plt.title('Frecuencia Media de Compra por Cohorte y Mes')
sns.heatmap(data=frecuencia_heatmap, annot=True, fmt='.2f',
cmap='Blues', cbar_kws={'label': 'Frecuencia Media'})
plt.xlabel('Mes de Compra')
plt.xticks(rotation=90)
plt.ylabel('Cohorte')
plt.show()
```




- Una alta frecuencia media de compra indica que los clientes de la cohorte están comprando más a menudo, lo que es un signo positivo de lealtad y satisfacción. Una frecuencia baja puede sugerir que los clientes están menos comprometidos con la marca.

# Valor Medio de Compra

El valor medio de compra indica cuánto gasta, en promedio, cada cliente por compra. La fórmula es:

$$\text{Valor Medio de Compra} = \frac{\text{Monto Total de Compras}}{\text{Número de Compras}}$$

```
# Calcular el valor medio de compra por cohorte y mes
valor_medio_compra = df.groupby(['Cohorte', 'Fecha_Compra'])
['Monto_Compra'].mean().reset_index()
valor_medio_compra
```

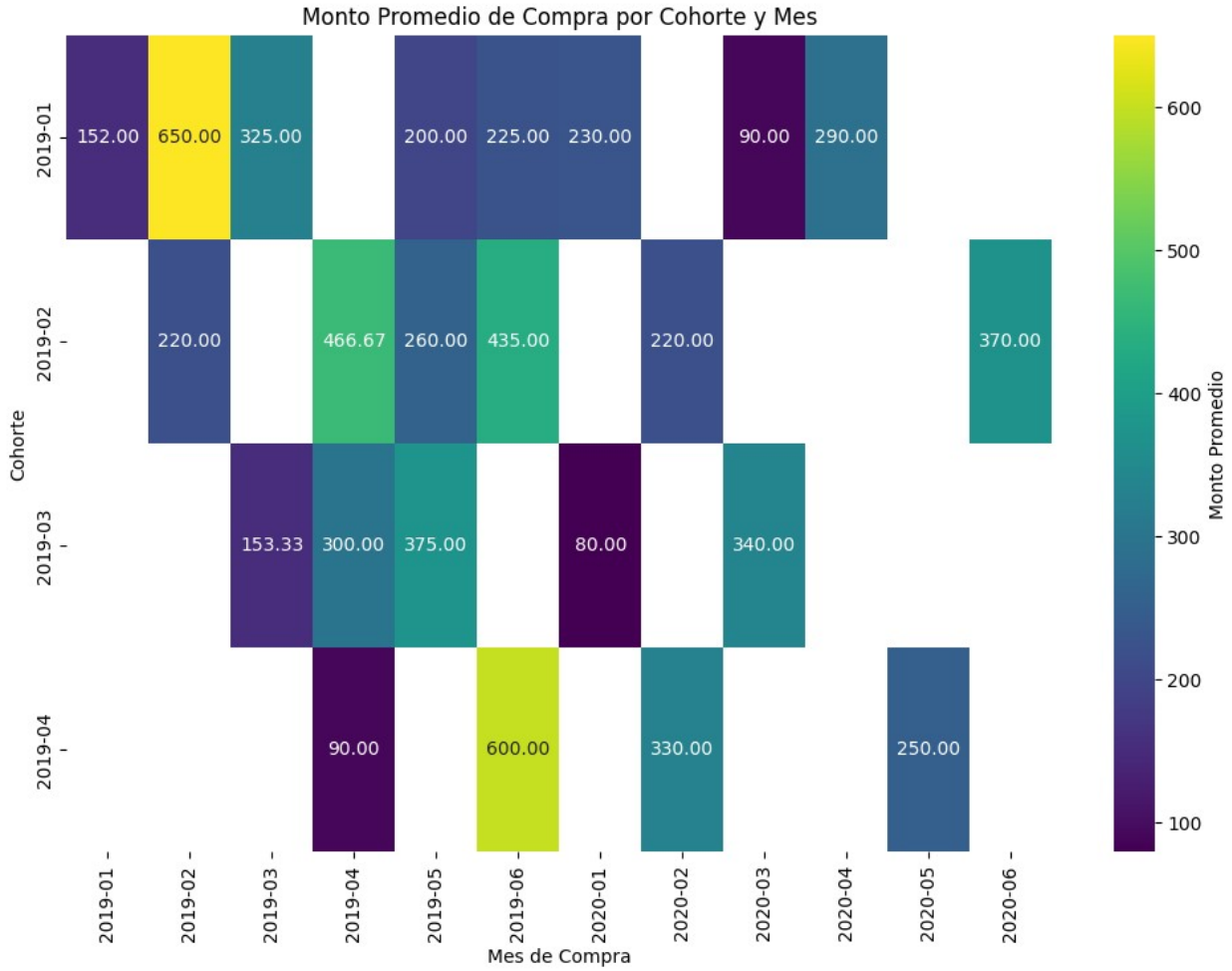
```
    Cohorte Fecha_Compra  Monto_Compra
0   2019-01      2019-01    152.000000
1   2019-01      2019-02    650.000000
2   2019-01      2019-03    325.000000
3   2019-01      2019-05    200.000000
4   2019-01      2019-06    225.000000
5   2019-01      2020-01    230.000000
6   2019-01      2020-03     90.000000
7   2019-01      2020-04    290.000000
8   2019-02      2019-02    220.000000
9   2019-02      2019-04    466.666667
10  2019-02      2019-05    260.000000
11  2019-02      2019-06    435.000000
12  2019-02      2020-02    220.000000
13  2019-02      2020-06    370.000000
14  2019-03      2019-03    153.333333
15  2019-03      2019-04    300.000000
16  2019-03      2019-05    375.000000
17  2019-03      2020-01     80.000000
18  2019-03      2020-03    340.000000
19  2019-04      2019-04     90.000000
20  2019-04      2019-06    600.000000
21  2019-04      2020-02    330.000000
22  2019-04      2020-05    250.000000
```

```
# Crear una tabla pivotante para el valor medio de compra
valor_medio_heatmap = valor_medio_compra.pivot_table(index='Cohorte',
columns='Fecha_Compra', values='Monto_Compra')
valor_medio_heatmap
```

```
Fecha_Compra  2019-01  2019-02     2019-03     2019-04  2019-05  2019-
06  \
Cohorte

2019-01         152.0    650.0  325.000000         NaN    200.0
225.0
2019-02           NaN    220.0         NaN  466.666667    260.0
```

```
435.0
2019-03           NaN      NaN  153.333333  300.000000    375.0
NaN
2019-04           NaN      NaN         NaN   90.000000      NaN
600.0

Fecha_Compra  2020-01  2020-02  2020-03  2020-04  2020-05  2020-06
Cohorte
2019-01         230.0      NaN     90.0    290.0      NaN      NaN
2019-02           NaN    220.0      NaN      NaN      NaN    370.0
2019-03          80.0      NaN    340.0      NaN      NaN      NaN
2019-04           NaN    330.0      NaN      NaN    250.0      NaN
```

```python
# Graficar valor medio de compra como heatmap
plt.figure(figsize=(12, 8))
plt.title('Monto Promedio de Compra por Cohorte y Mes')
sns.heatmap(data=valor_medio_heatmap, annot=True, fmt='.2f',
cmap='viridis', cbar_kws={'label': 'Monto Promedio'})
plt.xlabel('Mes de Compra')
plt.xticks(rotation=90)
plt.ylabel('Cohorte')
plt.show()
```




- Un aumento en el valor medio de compra podría sugerir que los clientes están dispuestos a gastar más, lo que podría ser el resultado de un aumento en la calidad del producto o una estrategia de marketing efectiva.

# Análisis de ingresos acumulados

Se puede calcular la suma acumulativa de los ingresos generados por cada cohorte a lo largo del tiempo.

```
# Calcular ingresos acumulados por cohorte y mes
ingresos_acumulados = df.groupby(['Cohorte',
'Fecha_Compra']).agg(Ingresos=('Monto_Compra', 'sum')).reset_index()
ingresos_acumulados
```

```
    Cohorte Fecha_Compra  Ingresos
0   2019-01      2019-01       760
1   2019-01      2019-02       650
2   2019-01      2019-03       650
3   2019-01      2019-05       400
4   2019-01      2019-06       450
```

```
5   2019-01      2020-01       230
6   2019-01      2020-03        90
7   2019-01      2020-04       580
8   2019-02      2019-02       880
9   2019-02      2019-04      1400
10  2019-02      2019-05       260
11  2019-02      2019-06       870
12  2019-02      2020-02       220
13  2019-02      2020-06       370
14  2019-03      2019-03       460
15  2019-03      2019-04       300
16  2019-03      2019-05       750
17  2019-03      2020-01        80
18  2019-03      2020-03       340
19  2019-04      2019-04        90
20  2019-04      2019-06       600
21  2019-04      2020-02       330
22  2019-04      2020-05       250
```

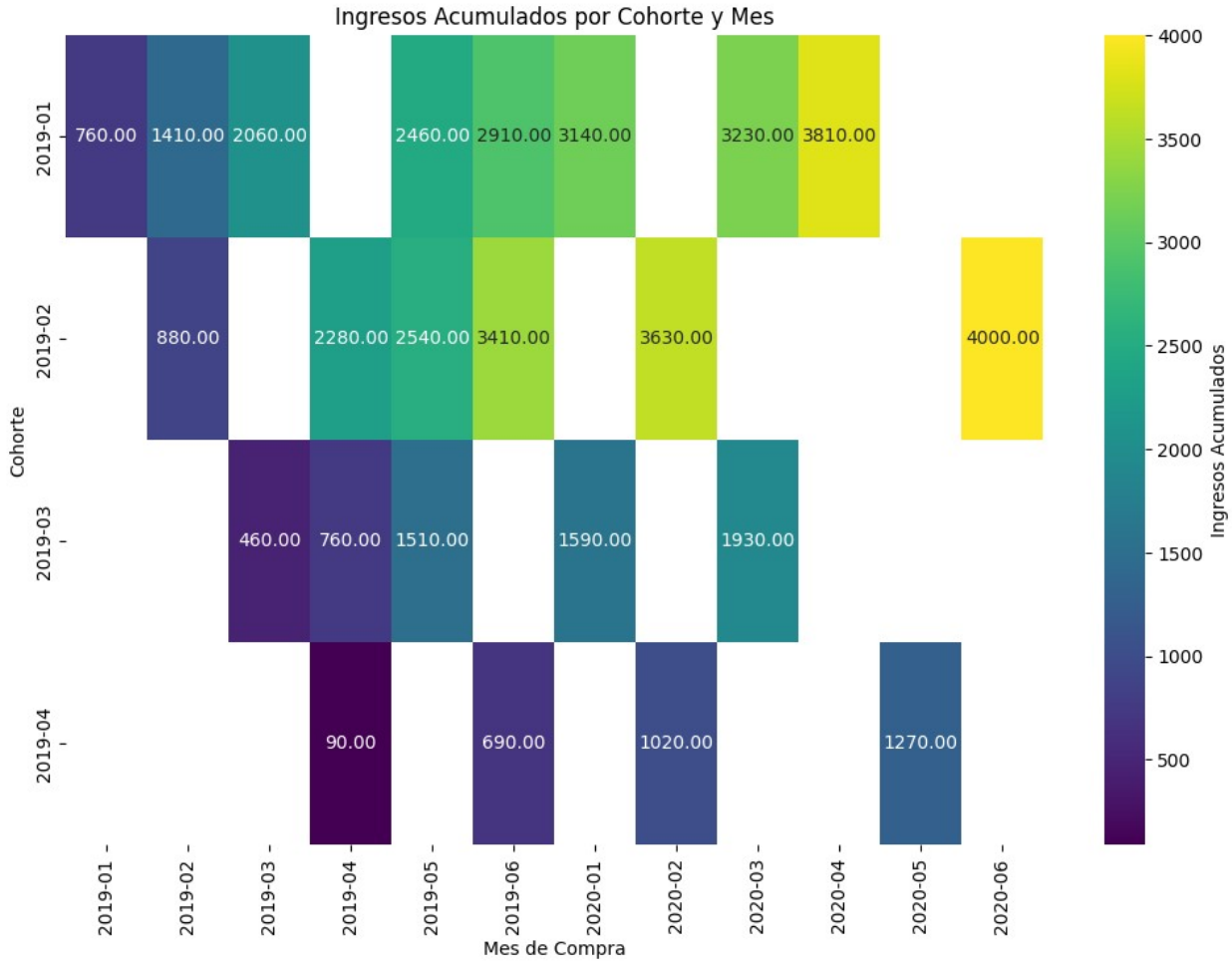
```
# Calcular los ingresos acumulados
ingresos_acumulados['Ingresos_Acumulados'] =
ingresos_acumulados.groupby('Cohorte')['Ingresos'].cumsum()
ingresos_acumulados
```

```
    Cohorte Fecha_Compra  Ingresos  Ingresos_Acumulados
0   2019-01      2019-01       760                  760
1   2019-01      2019-02       650                 1410
2   2019-01      2019-03       650                 2060
3   2019-01      2019-05       400                 2460
4   2019-01      2019-06       450                 2910
5   2019-01      2020-01       230                 3140
6   2019-01      2020-03        90                 3230
7   2019-01      2020-04       580                 3810
8   2019-02      2019-02       880                  880
9   2019-02      2019-04      1400                 2280
10  2019-02      2019-05       260                 2540
11  2019-02      2019-06       870                 3410
12  2019-02      2020-02       220                 3630
13  2019-02      2020-06       370                 4000
14  2019-03      2019-03       460                  460
15  2019-03      2019-04       300                  760
16  2019-03      2019-05       750                 1510
17  2019-03      2020-01        80                 1590
18  2019-03      2020-03       340                 1930
19  2019-04      2019-04        90                   90
20  2019-04      2019-06       600                  690
21  2019-04      2020-02       330                 1020
22  2019-04      2020-05       250                 1270
```

```
# Crear una tabla pivotante para los ingresos acumulados
ingresos_heatmap = ingresos_acumulados.pivot_table(index='Cohorte',
columns='Fecha_Compra', values='Ingresos_Acumulados')
ingresos_heatmap
```

```
Fecha_Compra  2019-01  2019-02  2019-03  2019-04  2019-05  2019-06
2020-01  \
Cohorte

2019-01         760.0   1410.0   2060.0      NaN   2460.0   2910.0
3140.0
2019-02           NaN    880.0      NaN   2280.0   2540.0   3410.0
NaN
2019-03           NaN      NaN    460.0    760.0   1510.0      NaN
1590.0
2019-04           NaN      NaN      NaN     90.0      NaN    690.0
NaN

Fecha_Compra  2020-02  2020-03  2020-04  2020-05  2020-06
Cohorte
2019-01           NaN   3230.0   3810.0      NaN      NaN
2019-02        3630.0      NaN      NaN      NaN   4000.0
2019-03           NaN   1930.0      NaN      NaN      NaN
2019-04        1020.0      NaN      NaN   1270.0      NaN
```

```
# Graficar ingresos acumulados como heatmap
plt.figure(figsize=(12, 8))
plt.title('Ingresos Acumulados por Cohorte y Mes')
sns.heatmap(data=ingresos_heatmap, annot=True, fmt='.2f',
cmap='viridis', cbar_kws={'label': 'Ingresos Acumulados'})
plt.xlabel('Mes de Compra')
plt.xticks(rotation=90)
plt.ylabel('Cohorte')
plt.show()
```




- El análisis de ingresos acumulados proporciona información sobre la estabilidad y crecimiento de los ingresos a lo largo del tiempo. Un crecimiento constante indica un negocio saludable, mientras que una caída podría requerir un examen más detallado.

# Análisis de Segmentación

Un análisis de segmentación es el proceso de dividir una base de datos de clientes, productos o datos de comportamiento en subgrupos o segmentos más pequeños, que comparten características similares. Esto permite identificar patrones específicos dentro de cada segmento, facilitando la personalización de estrategias y optimizando recursos en áreas clave como marketing, ventas, retención de clientes y desarrollo de productos.

**Segmentación de Ingresos por Canal de Venta**: Este análisis examina los ingresos generados por cada cohorte a través de diferentes canales de venta (por ejemplo, "Online" y "Tienda") en cada período de compra. Al segmentar los ingresos de esta forma, se busca entender cómo cada cohorte prefiere comprar y si existen patrones de mayor gasto en algún canal específico.

```
# Calcular ingresos por canal de venta y cohorte
ingresos_por_canal = df.groupby(['Cohorte',
```

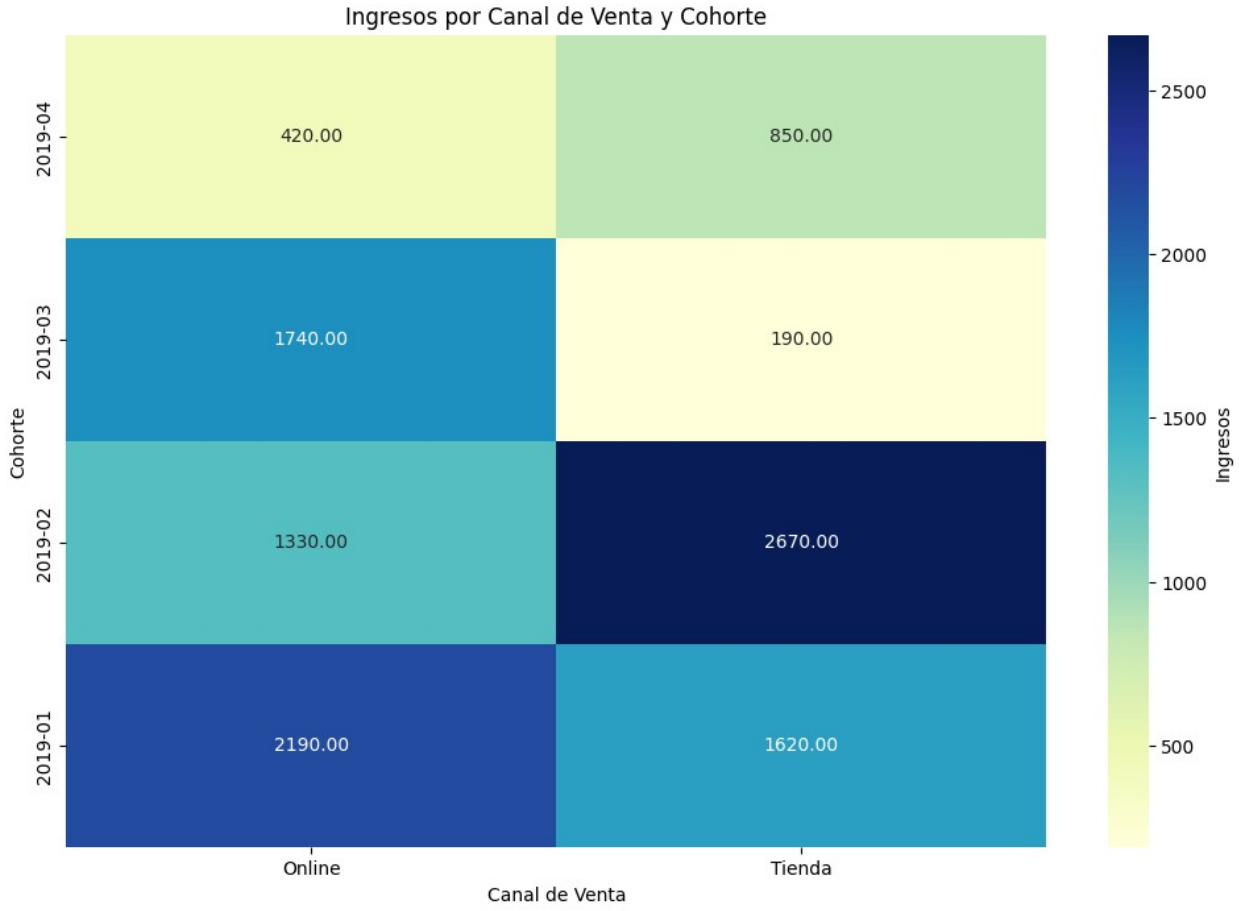
```
'Canal_Venta']).agg(Ingresos=('Monto_Compra', 'sum')).reset_index()
ingresos_por_canal
```

```
   Cohorte Canal_Venta  Ingresos
0  2019-01      Online      2190
1  2019-01      Tienda      1620
2  2019-02      Online      1330
3  2019-02      Tienda      2670
4  2019-03      Online      1740
5  2019-03      Tienda       190
6  2019-04      Online       420
7  2019-04      Tienda       850
```

```
# Crear una tabla pivotante para los ingresos por canal de venta
ingresos_canal_heatmap =
ingresos_por_canal.pivot_table(index='Cohorte', columns='Canal_Venta',
values='Ingresos')
ingresos_canal_heatmap =
ingresos_canal_heatmap.sort_index(ascending=False)
ingresos_canal_heatmap
```

```
Canal_Venta  Online  Tienda
Cohorte
2019-04         420     850
2019-03        1740     190
2019-02        1330    2670
2019-01        2190    1620
```

```
# Graficar ingresos por canal de venta como heatmap
plt.figure(figsize=(12, 8))
plt.title('Ingresos por Canal de Venta y Cohorte')
sns.heatmap(data=ingresos_canal_heatmap, annot=True, fmt='.2f',
cmap='YlGnBu', cbar_kws={'label': 'Ingresos'})
plt.xlabel('Canal de Venta')
plt.ylabel('Cohorte')
plt.show()
```




**Segmentación de Ingreso Acumulado por Canal de Venta**: Este análisis se enfoca en el ingreso total acumulado que cada cohorte ha generado a lo largo del tiempo en cada canal de venta. Se suma el ingreso de cada período para mostrar el ingreso acumulado hasta ese momento, lo cual permite ver el crecimiento en el valor de la cohorte a lo largo del tiempo.

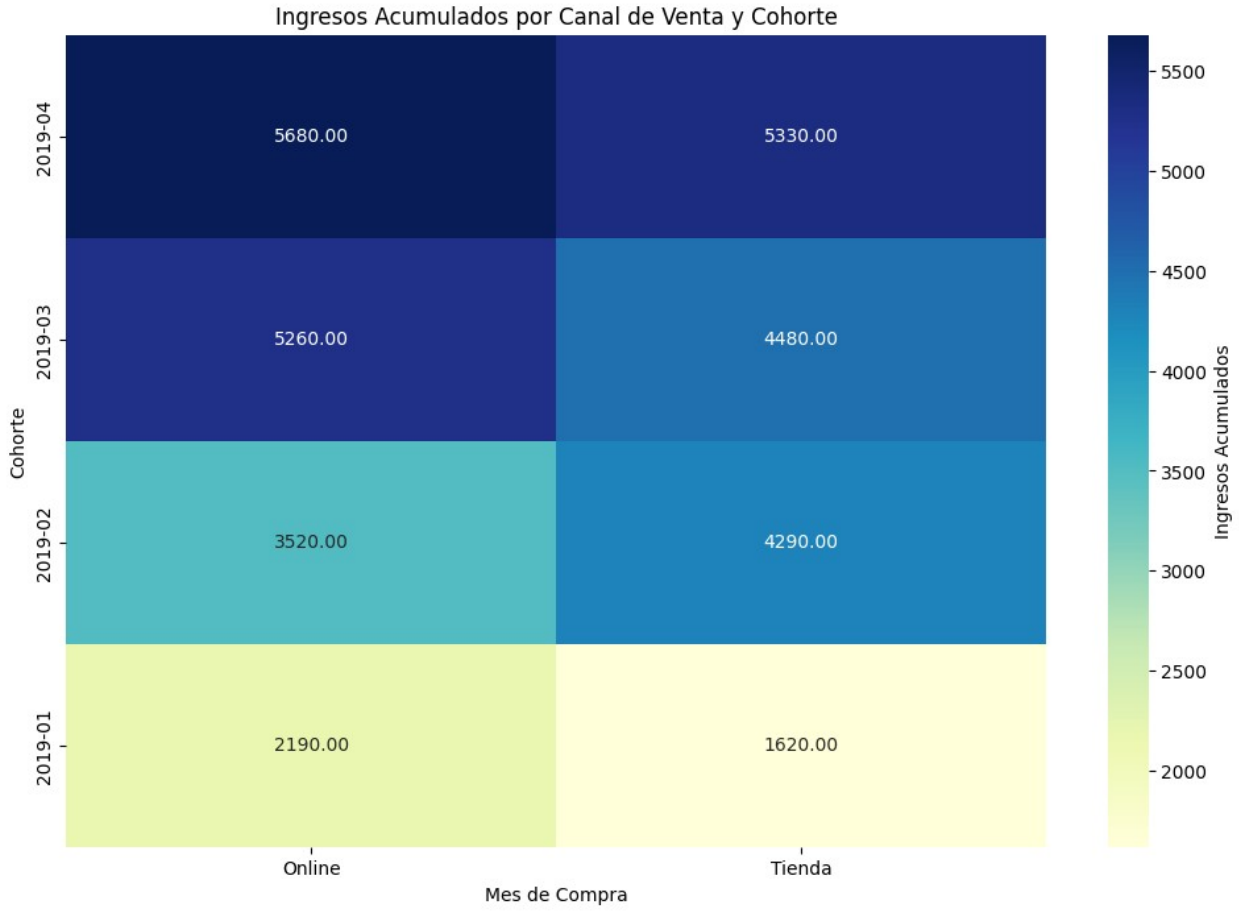
```
# Calcular ingresos acumulados por canal de venta dentro de cada cohorte
ingresos_por_canal['Ingresos_Acumulados'] =
ingresos_por_canal.groupby('Canal_Venta')['Ingresos'].cumsum()
ingresos_por_canal
```

| | Cohorte | Canal_Venta | Ingresos | Ingresos_Acumulados |
|---|---|---|---|---|
| 0 | 2019-01 | Online | 2190 | 2190 |
| 1 | 2019-01 | Tienda | 1620 | 1620 |
| 2 | 2019-02 | Online | 1330 | 3520 |
| 3 | 2019-02 | Tienda | 2670 | 4290 |
| 4 | 2019-03 | Online | 1740 | 5260 |
| 5 | 2019-03 | Tienda | 190 | 4480 |
| 6 | 2019-04 | Online | 420 | 5680 |
| 7 | 2019-04 | Tienda | 850 | 5330 |

```
# Crear una tabla pivotante para los ingresos acumulados por canal de venta
ingresos_acumulados_heatmap = ingresos_por_canal.pivot_table(index='Cohorte', columns='Canal_Venta', values='Ingresos_Acumulados')
ingresos_acumulados_heatmap = ingresos_acumulados_heatmap.sort_index(ascending=False)
ingresos_acumulados_heatmap
```

| Canal_Venta | Online | Tienda |
|---|---|---|
| Cohorte | | |
| 2019-04 | 5680 | 5330 |
| 2019-03 | 5260 | 4480 |
| 2019-02 | 3520 | 4290 |
| 2019-01 | 2190 | 1620 |

```
# Graficar ingresos acumulados por canal de venta como heatmap
plt.figure(figsize=(12, 8))
plt.title('Ingresos Acumulados por Canal de Venta y Cohorte')
sns.heatmap(data=ingresos_acumulados_heatmap, annot=True, fmt='.2f', cmap='YlGnBu', cbar_kws={'label': 'Ingresos Acumulados'})
plt.xlabel('Mes de Compra')
plt.ylabel('Cohorte')
plt.show()
```




**Segmentación de Frecuencia de Compra por Canal de Venta**: Este análisis mide la frecuencia promedio de compra en cada canal por cada cohorte, lo cual ayuda a conocer cuántas veces los clientes compran en diferentes canales en un período determinado.

```
# Calcular la frecuencia de compra por canal y cohorte
frecuencia_por_canal = df.groupby(['Cohorte',
'Canal_Venta']).agg(Frecuencia=('ID_Cliente', 'count')).reset_index()
frecuencia_por_canal
```

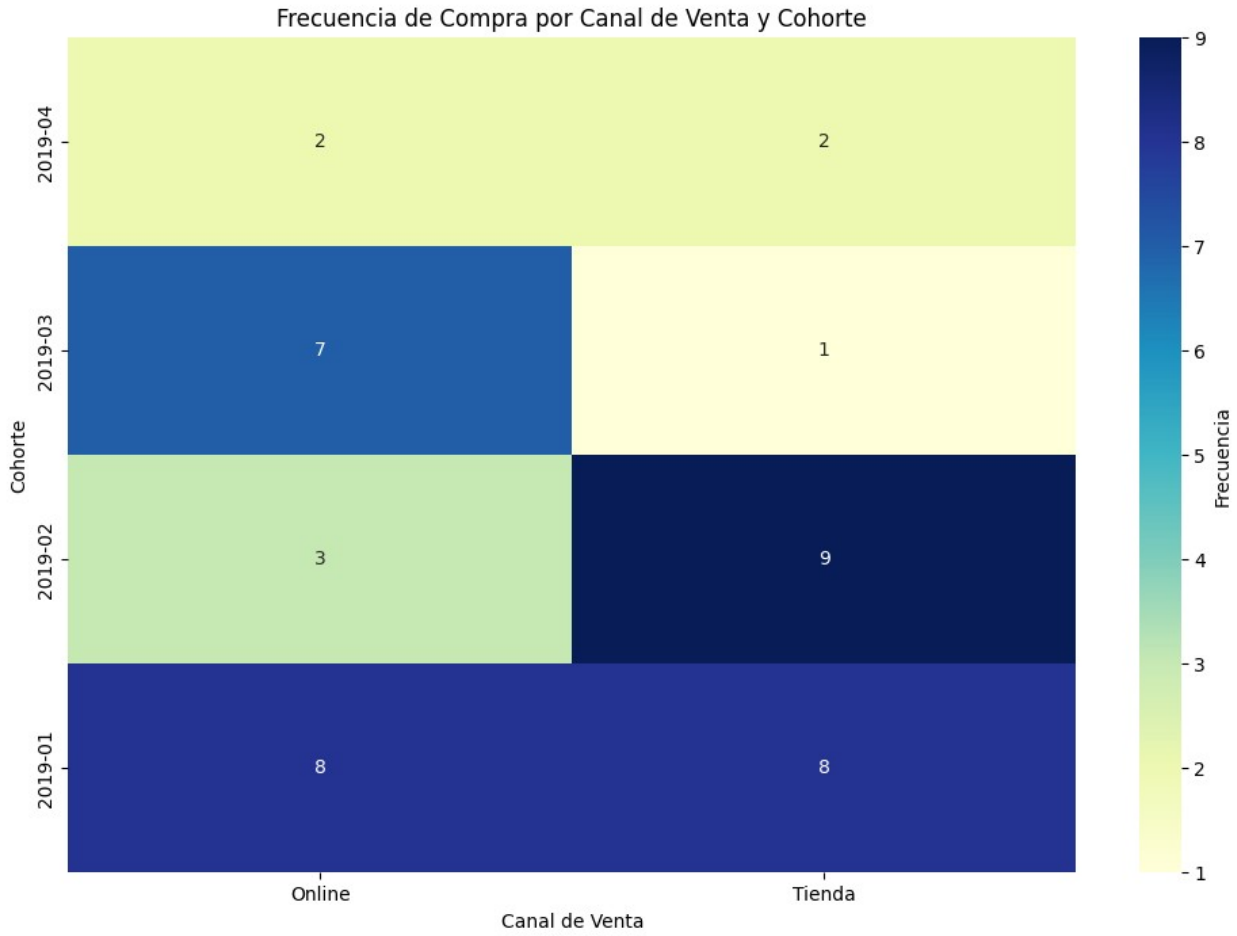
| | Cohorte | Canal_Venta | Frecuencia |
|---|---|---|---|
| 0 | 2019-01 | Online | 8 |
| 1 | 2019-01 | Tienda | 8 |
| 2 | 2019-02 | Online | 3 |
| 3 | 2019-02 | Tienda | 9 |
| 4 | 2019-03 | Online | 7 |
| 5 | 2019-03 | Tienda | 1 |
| 6 | 2019-04 | Online | 2 |
| 7 | 2019-04 | Tienda | 2 |

```
# Crear una tabla pivotante para la frecuencia de compra por canal
frecuencia_canal_heatmap =
frecuencia_por_canal.pivot_table(index='Cohorte',
```

```
columns='Canal_Venta', values='Frecuencia')
frecuencia_canal_heatmap =
frecuencia_canal_heatmap.sort_index(ascending=False)
frecuencia_canal_heatmap
```

| Canal_Venta | Online | Tienda |
|---|---|---|
| Cohorte | | |
| 2019-04 | 2 | 2 |
| 2019-03 | 7 | 1 |
| 2019-02 | 3 | 9 |
| 2019-01 | 8 | 8 |

```
# Graficar la frecuencia de compra por canal como heatmap
plt.figure(figsize=(12, 8))
plt.title('Frecuencia de Compra por Canal de Venta y Cohorte')
sns.heatmap(data=frecuencia_canal_heatmap, annot=True, fmt='.0f',
cmap='YlGnBu', cbar_kws={'label': 'Frecuencia'})
plt.xlabel('Canal de Venta')
plt.ylabel('Cohorte')
plt.show()
```



- Comprender qué canal de venta es más efectivo para diferentes cohortes permite a la empresa optimizar sus esfuerzos de marketing y asignar recursos de manera más efectiva.