PROYECTO FINAL
Fintech EasyMoney

Adrian Nora, Javier Alfonso y David Carlón

nuclio digital school

easyMoney

Índice de Contenido

0. Contexto del Proyecto

Hace casi cuatro años, easyMoney fue fundada por Carol Denver, una profesional consolidada en el sector de la banca de inversión. Tras más de una década trabajando para grandes firmas, Carol decidió emprender su propio proyecto: una plataforma multicanal enfocada en la comercialización de productos financieros (ahorro, inversión y financiación). El objetivo de easyMoney era ofrecer una interfaz amigable y accesible, donde los clientes pudieran encontrar soluciones financieras personalizadas y contratarlas de forma sencilla.

El primer producto lanzado fue la cuenta easyMoney, un servicio innovador que permite a los usuarios acumular dinero automáticamente mediante el redondeo de sus compras. Este producto fue un éxito rotundo, lo que impulsó la ampliación de su cartera de soluciones, incluyendo productos de inversión y tarjetas, entre otros. Para facilitar su operación, y debido a la falta de licencia bancaria propia, easyMoney estableció una alianza con easyBanking S.A., una entidad regulada por el Banco de España que actúa como proveedor de productos financieros. Sin embargo, la fuerte influencia de easyBanking en la estrategia comercial ha generado tensiones en la relación entre ambas empresas.

Gracias a la perseverancia y red de contactos de Carol, easyMoney logró superar sus objetivos iniciales tras dos exitosas rondas de financiación. Esto permitió a la empresa captar un volumen significativo de clientes durante su primer año de operación, alcanzando una facturación millonaria y una plantilla cercana a las 100 personas. No obstante, tras cuatro años de actividad, la empresa enfrenta varios desafíos que amenazan su sostenibilidad.

Entre los principales retos se encuentran la creciente presión de easyBanking para incorporar productos complejos, lo que ha desviado a la empresa de su visión original de ofrecer soluciones sencillas y enfocadas en las necesidades del cliente. Además, los fondos obtenidos en las rondas de financiación están cerca de agotarse y, a pesar de su crecimiento, easyMoney no ha alcanzado aún un EBITDA positivo que le permita operar de manera independiente. Su principal inversor, Lion Global Management, ha demandado una mayor rentabilidad de la base de clientes actual antes de comprometer más capital.

Asimismo, la alta rotación en el equipo de tecnología y la falta de inversión en infraestructura tecnológica están generando ineficiencias que afectan a diversas áreas de la empresa. Las tensiones internas han comenzado a erosionar el espíritu ágil que caracterizó los primeros años de desarrollo. Ante este panorama, y tras la salida de un miembro clave del equipo de marketing, la dirección de easyMoney ha decidido contratar a un Data Scientist. Este profesional tendrá la misión de ayudar a la empresa a mejorar la rentabilidad de su cartera de clientes, abordando los retos actuales y acompañando a easyMoney en su nueva etapa de crecimiento.

1. Preprocesamiento y Análisis

El preprocesamiento de datos es un paso fundamental en cualquier proyecto de **Data Science**. Consiste en preparar y limpiar los datos antes de su análisis, asegurando que la información con la que se trabajará sea precisa, completa y consistente. En el caso de *easyMoney*, el dataset proporcionado está dividido en tres secciones principales que agrupan distintos tipos de información:

1. **Características sociodemográficas:** Incluye variables relacionadas con los clientes, como su edad, país de residencia, género, provincia, y salario.
2. **Productos contratados:** Recoge información sobre los distintos productos financieros que cada cliente ha adquirido, como depósitos, tarjetas, y préstamos.
3. **Actividad comercial:** Proporciona datos tales como el segmento comercial de los clientes, canal de entrada, estado en la aplicación de EasyMoney y primera fecha de contratación.

Cada uno de estos conjuntos de datos contiene información valiosa y complementaria que, en conjunto, permitirá a *easyMoney* mejorar la segmentación de clientes y definir estrategias personalizadas. Sin embargo, antes de realizar cualquier tipo de análisis avanzado, es crucial llevar a cabo un preprocesamiento que permita:

- **Identificar y corregir errores:** Como datos faltantes, duplicados o inconsistentes.
- **Establecer la coherencia temporal:** Alineando las fechas y la evolución de las variables a lo largo del tiempo.
- **Transformar y estandarizar:** Adaptar las variables categóricas y numéricas a un formato adecuado para los modelos de machine learning.

El análisis de los datos comienza con una limpieza exhaustiva y una exploración preliminar. Este paso inicial no solo facilita la comprensión del estado actual de la base de datos, sino que también define el camino hacia la implementación de los modelos predictivos y el descubrimiento de patrones clave.

1.1. Sociodemographic DataBase

En esta primera parte del análisis, nos enfocamos en la sección de **características sociodemográficas** del dataset de *easyMoney*, que incluye información clave sobre los clientes, como su edad, país de residencia, género, región y salario. El objetivo de este apartado es realizar un preprocesamiento adecuado de estos datos, asegurando la calidad de la información antes de llevar a cabo análisis más detallados.

1.1.1. Preprocesameinto

El archivo fue cargado y posteriormente se realizaron algunas limpiezas y ajustes iniciales:

- **Eliminación de columnas innecesarias:** Se eliminaron columnas que no aportaban información relevante, como 'Unnamed: 0_x', 'Unnamed: 0_y', y otras que eran resultado de procesos previos o simplemente no contribuían al análisis.

- **Cambio de tipo de datos:** Se ajustaron los tipos de datos de varias columnas para asegurar una correcta manipulación posterior. Por ejemplo, se convirtieron fechas a datetime, algunas columnas a categoría, y variables numéricas a int32.

Se realizó un análisis preliminar que proporciona una visión detallada de las principales características de cada columna del dataset, incluyendo estadísticas descriptivas, número de valores únicos, porcentaje de valores nulos, entre otros.

- **Nulos:** Se identificaron valores nulos en las variables **gender** (25 casos), **region_code** (2,264 casos) y **salary** (más de 1.5 millones de casos), lo cual requerirá un tratamiento posterior.
- **Country y Deceased:** Se observó que la mayoría de los clientes residen en España y casi ningún cliente está marcado como fallecido, por lo que estas variables presentan poca variabilidad.

column	dtype	count	mean	std	min	25%	50%	75%	max	nunique	unique	cardinality %	mode	mode %	null_count	null %
pk_cid	int64	5962924	1234929.800000	162302.000000	15891	1112532.000000	1231097.000000	1352339.000000	1553689	456373		7.700000	17457	0.000000	0	0.000000
pk_partition	datetime64[ns]	5962924								17		0.000000	2019-05-28 00:00:00	7.400000	0	0.000000
age	int32	5962924	29.800000	12.000000	2	22.000000	25.000000	34.000000	105	104		0.000000	22	12.200000	0	0.000000
country_id	category	5962924								41		0.000000	ES	100.000000	0	0.000000
deceased	object	5962924								2	['N', 'S']	0.000000	N	100.000000	0	0.000000
gender	category	5962899								2	['H', 'V', nan]	0.000000	H	51.800000	25	0.000000
region_code	category	5960660								52		0.000000	28.000000	19.900000	2264	0.000000
salary	float64	4450821	115816.700000	199551.900000	1202.700000	61500.600000	88654.600000	131669.900000	28894395.500000	258629		4.300000	451931.220000	0.000000	1512103	25.400000

1.1.2. Edad (Age)

La variable **age** es una de las más importantes en el análisis de clientes. Se realizó un análisis detallado de su distribución:

- **Distribución de edades:** Se analizaron las edades más comunes de los clientes en la partición más reciente (mayo de 2019). Un gráfico interactivo mostró las 15 edades más frecuentes, con una **concentración significativa en torno a los 30 años**.

- **Clientes con edades mixtas:** Se detectaron **341,742 clientes** con múltiples valores de edad, lo que indica inconsistencias en la entrada de datos. Estos casos serán analizados y corregidos en las siguientes etapas.

```python
# Contar cuántos valores únicos de 'age' tiene cada cliente
estado_por_cliente_age = df_full.groupby('pk_cid')['age'].nunique()

# Identificar clientes con edades mixtas (múltiples valores en 'age')
clientes_mixtos_age = estado_por_cliente_age[estado_por_cliente_age > 1].index

# Excluir los clientes mixtos del análisis de las otras categorías
df_no_mixtos_age = df_full[~df_full['pk_cid'].isin(clientes_mixtos_age)]

# Identificar clientes que tienen siempre la misma edad
clientes_misma_edad = df_no_mixtos_age.groupby('pk_cid')['age'].nunique()
clientes_misma_edad = clientes_misma_edad[clientes_misma_edad == 1].index

# Identificar clientes con valores NaN en 'age' (excluyendo clientes mixtos)
clientes_con_nan_age = df_no_mixtos_age[df_no_mixtos_age['age'].isna()]['pk_cid'].unique()

# Contar el número de clientes en cada categoría (excluyendo mixtos)
n_misma_edad = len(clientes_misma_edad)
n_con_nan_age = len(clientes_con_nan_age)
n_mixtos_age = len(clientes_mixtos_age)

# Mostrar los resultados
print(f"Clientes que tienen siempre la misma edad (excluyendo mixtos): {n_misma_edad}")
print(f"Clientes con valores NaN en 'age' (excluyendo mixtos): {n_con_nan_age}")
print(f"Clientes con edades mixtas (diferentes valores de edad): {n_mixtos_age}")
```

```
Clientes que tienen siempre la misma edad (excluyendo mixtos): 114631
Clientes con valores NaN en 'age' (excluyendo mixtos): 0
Clientes con edades mixtas (diferentes valores de edad): 341742
```

Al analizar la variable **edad** en el dataset, se detectaron dos tipos de anomalías: clientes que se mantienen con una **edad constante**, siendo menores de edad, y aquellos con **edades mixtas**, donde se observan cambios incoherentes en la edad reportada a lo largo del tiempo. A continuación, se describe cómo se abordaron ambas situaciones.

Clientes Menores

En este caso, se identificaron clientes cuya edad es **menor a 18 años** y que mantienen la misma edad a lo largo del tiempo. Esto plantea un posible problema, ya que estos clientes, por su edad, no deberían tener acceso a ciertos productos financieros, lo cual podría ser indicativo de errores en los registros. Se procedió con el siguiente análisis:

- **Filtro de clientes menores de 18 años:** Se seleccionaron aquellos clientes cuya edad es inferior a los 18 años, obteniendo un resultado de 7368 clientes.
- **Análisis de productos activos:** Se evaluó cuántos de estos clientes tenían productos financieros contratados. De los productos disponibles, se detectaron muy pocos casos, 2 clientes con **nómina**, 9 con **planes de pensiones** y 3 con **cuenta EasyMoney.**
- **Clientes activos:** Se evaluó cuántos de estos menores eran marcados como **active_customer**, es decir, que habían realizado alguna transacción reciente. El análisis reveló un número pequeño de casos activos.

Dado que el volumen de usuarios menores de edad no representa una cantidad relevante en comparación al volumen del dataset completo, y considerando que es posible que estas cuentas estén siendo gestionadas por los padres como cuentas de ahorro para el futuro de los menores, se ha decidido mantener estos registros en el análisis.

Edad Mixta

La segunda anomalía encontrada se refiere a los clientes con **edades mixtas**, es decir, aquellos cuyos registros muestran varias edades distintas a lo largo del tiempo, lo cual podría deberse a errores de tipificación o de captura en el sistema. Esta situación se aborda desde dos perspectivas:

a) Variaciones Mayores a 3 Años

Para detectar errores en la tipificación de la edad, se analizó la diferencia entre la edad más reciente y la más antigua registrada para cada cliente. Dado que los datos cubren un periodo de 3 años, se estableció un **umbral de 3 años** como la variación máxima aceptable. Los resultados mostraron que **1,093 clientes** tenían una diferencia mayor a 3 años, con una variación promedio de **25 años**, lo cual es claramente incoherente.

Se decidió **corregir estas inconsistencias** utilizando la última edad registrada para cada cliente, bajo la premisa de que las entradas más recientes son más fiables.

b) Variaciones Negativas

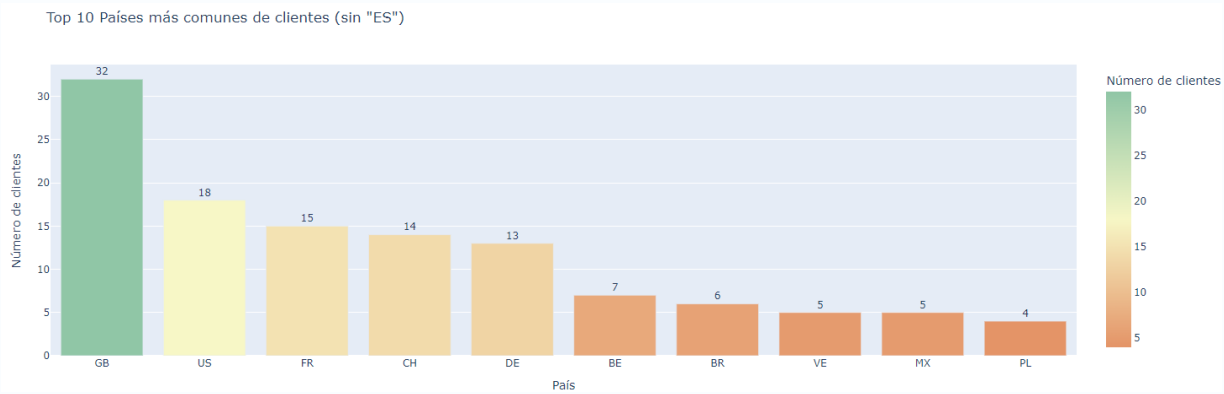
También se detectaron casos en los que la edad de los clientes **disminuía** en registros sucesivos, algo que es evidentemente incorrecto. En estos casos, se procedió de manera similar: se sustituyó la edad de estos clientes por la última edad conocida, ya que esta es probablemente la más correcta.

1.1.3 Country_ID

La variable country_id representa el país de residencia de los clientes y, en este dataset, muestra una predominancia clara de España ('ES'), con más de 5.9 millones de clientes registrados en este país. Esta variable es crucial para entender la distribución geográfica de los clientes, aunque con un claro sesgo hacia España. A continuación, se detallan los pasos realizados para la limpieza y preprocesamiento de esta variable.

Distribución General de country_id

Inicialmente, se analizaron los **10 países más frecuentes** excluyendo España, con el objetivo de identificar si había alguna tendencia geográfica notable fuera de este país. En este caso, el Reino Unido ('GB'), Francia ('FR'), y Alemania ('DE') figuran entre los países más comunes después de España, pero el volumen de clientes de estos países es significativamente menor (con menos de 500 clientes en cada uno).



Análisis de Clientes con country_id Mixto

Durante el preprocesamiento, se encontraron **25 clientes con valores mixtos** en la columna country_id, lo que significa que en diferentes registros un mismo cliente aparecía con distintos países de residencia. Estos casos pueden tener diversas explicaciones:

- **Errores en la captura de datos**, donde el país de residencia podría haber sido registrado incorrectamente.
- **Cambios reales de residencia**, aunque este tipo de movimientos suelen implicar cambios en otras variables, como el salario, lo cual no se observó en estos casos.

Para resolver esta inconsistencia, se tomó la decisión de **mantener el último país registrado** para cada cliente mixto. Esta decisión se basa en la lógica de que el último valor es probablemente el más actualizado y preciso. Además, dado que se trata de un número muy pequeño de clientes, su impacto en los resultados generales del análisis es mínimo.

```
# Contar cuántos valores únicos de 'country_id' tiene cada cliente
estado_por_cliente_country = df_full.groupby('pk_cid')['country_id'].nunique()

# Identificar clientes con 'country_id' mixtos (múltiples valores de 'country_id')
clientes_mixtos_country = estado_por_cliente_country[estado_por_cliente_country > 1].index

# Excluir los clientes mixtos del análisis de las otras categorías
df_no_mixtos_country = df_full[~df_full['pk_cid'].isin(clientes_mixtos_country)]

# Identificar clientes que tienen siempre el mismo 'country_id'
clientes_mismo_country = df_no_mixtos_country.groupby('pk_cid')['country_id'].nunique()
clientes_mismo_country = clientes_mismo_country[clientes_mismo_country == 1].index

# Identificar clientes con valores NaN en 'country_id' (excluyendo clientes mixtos)
clientes_con_nan_country = df_no_mixtos_country[df_no_mixtos_country['country_id'].isna()]['pk_cid'].unique()

# Contar el número de clientes en cada categoría (excluyendo mixtos)
n_mismo_country = len(clientes_mismo_country)
n_con_nan_country = len(clientes_con_nan_country)
n_mixtos_country = len(clientes_mixtos_country)

# Mostrar los resultados
print(f"Clientes que tienen siempre el mismo 'country_id' (excluyendo mixtos): {n_mismo_country}")
print(f"Clientes con valores NaN en 'country_id' (excluyendo mixtos): {n_con_nan_country}")
print(f"Clientes con 'country_id' mixtos (diferentes países): {n_mixtos_country}")
```

```
Clientes que tienen siempre el mismo 'country_id' (excluyendo mixtos): 456348
Clientes con valores NaN en 'country_id' (excluyendo mixtos): 0
Clientes con 'country id' mixtos (diferentes países): 25
```

Mantener el **último país de residencia** es la opción más adecuada, ya que refleja la situación más actual del cliente. Además, dado que el salario no cambió en ninguno de estos casos, es probable que los cambios de country_id sean errores en la captura de datos, lo que refuerza la decisión de optar por el dato más reciente.

1.1.4. Deceased

La variable deceased es una indicación de si el cliente ha fallecido o no, con los valores 'N' para los clientes vivos y 'S' para los fallecidos. La presencia de esta variable puede ser relevante desde una perspectiva comercial, ya que los clientes fallecidos naturalmente dejarían de interactuar con la plataforma. Sin embargo, durante el análisis se observaron varios factores que llevaron a reconsiderar su uso en los modelos.

Variabilidad y Distribución de deceased

En este dataset, la gran mayoría de los clientes están marcados como **vivos** ('N'), con solo **55 clientes fallecidos** ('S'). Además, se detectaron **74 casos mixtos (particiones)**, en los cuales el estado del cliente cambiaba entre 'N' y 'S' en diferentes registros. Esto no tiene sentido desde una perspectiva lógica, ya que una vez fallecido, un cliente no puede revertir su estado. Este hecho sugiere errores en la captura o en el registro de la variable.

Análisis de Clientes Mixtos y Constantes

- **Clientes constantes en deceased**: La mayoría de los clientes mantienen su estado de fallecimiento constante a lo largo del tiempo. Esto incluye tanto a los 55 clientes fallecidos como a los demás clientes vivos.
- **Clientes mixtos en deceased**: Los casos mixtos sugieren errores en la captura de los datos o falta de actualización en algunos registros. Dado que el estado fallecido es irreversible, estos casos fueron corregidos asignando el último estado registrado para cada cliente.

Eliminación de deceased en los Modelos Predictivos

Después de realizar este análisis, se concluyó que la variable deceased **no tiene suficiente capacidad explicativa** para ser utilizada en los modelos de segmentación. Además, se identificó una **alta correlación** entre esta variable y otras como la **edad**, lo que aumenta el riesgo de multicolinealidad si se incluye en los modelos predictivos. Dado que la mayoría de los clientes fallecidos pertenecen a grupos de mayor edad, esta variable no añade valor adicional y podría introducir redundancia.

1.1.5. Gender

La variable gender en este dataset cuenta con tres posibles valores: **H** para hombres, **V** para mujeres, y **Unknown** para los casos donde el género no está especificado. El análisis de esta variable es importante para evaluar la segmentación de la clientela en términos de género y para identificar cualquier inconsistencia en los datos.

Distribución General de gender

El análisis inicial mostró que la distribución por género es bastante equilibrada, con un **51.24% de hombres** y un **48.76% de mujeres**. Los valores **Unknown** son extremadamente raros, con solo **2 clientes** en esta categoría, lo cual representa un porcentaje insignificante.

```python
# Contar cuántos valores únicos tiene cada cliente en la columna gender
estado_por_cliente_gender = df_full.groupby('pk_cid')['gender'].agg(['nunique', 'count'])

# Identificar clientes con géneros mixtos (H y F o Unknown)
clientes_mixtos_gender = estado_por_cliente_gender[estado_por_cliente_gender['nunique'] > 1].index

# Excluir los clientes mixtos del análisis de las otras categorías
df_no_mixtos_gender = df_full[~df_full['pk_cid'].isin(clientes_mixtos_gender)]

# Identificar clientes que son siempre 'H'
clientes_siempre_H = df_no_mixtos_gender[df_no_mixtos_gender['gender'] == 'H']['pk_cid'].unique()

# Identificar clientes que son siempre 'V'
clientes_siempre_V = df_no_mixtos_gender[df_no_mixtos_gender['gender'] == 'V']['pk_cid'].unique()

# Identificar clientes "Unknown" en gender (excluyendo clientes mixtos)
clientes_siempre_Unknown = df_no_mixtos_gender[df_no_mixtos_gender['gender'] == "Unknown"]['pk_cid'].unique()

# Contar el número de clientes en cada categoría (excluyendo mixtos)
n_siempre_H = len(clientes_siempre_H)
n_siempre_V = len(clientes_siempre_V)
n_siempre_Unknown = len(clientes_siempre_Unknown)
n_mixtos_gender = len(clientes_mixtos_gender)

# Mostrar los resultados
print(f"Clientes que son siempre 'H' en 'gender' (excluyendo mixtos): {n_siempre_H}")
print(f"Clientes que son siempre 'V' en 'gender' (excluyendo mixtos): {n_siempre_V}")
print(f"Clientes que son todo 'Unknown' (excluyendo mixtos): {n_siempre_Unknown}")
print(f"Clientes con géneros mixtos (H y V o Unknown): {n_mixtos_gender}")
print(f"Total de clientes en el dataset: {n_siempre_H + n_siempre_V + n_siempre_Unknown + n_mixtos_gender}")
```

```
Clientes que son siempre 'H' en 'gender' (excluyendo mixtos): 233821
Clientes que son siempre 'V' en 'gender' (excluyendo mixtos): 222509
Clientes que son todo 'Unknown' (excluyendo mixtos): 2
Clientes con géneros mixtos (H y V o Unknown): 41
Total de clientes en el dataset: 456373
```

Casos Mixtos en gender

Al revisar los datos, se detectaron **41 casos mixtos**, es decir, clientes que en diferentes registros aparecían con distintos géneros. Esta situación, aunque rara, es un indicativo de posibles errores en la captura de datos. La presencia de géneros mixtos puede deberse a varias razones, como:

- **Errores en el registro** donde el género fue introducido incorrectamente en algún momento.
- **Cambios reales en el género**, aunque dado el bajo número de casos, esta explicación parece poco probable.

Para resolver estos casos, se decidió asignar el **último género registrado** para cada cliente, bajo la premisa de que este es el dato más actualizado y, por tanto, el más fiable.

Imputación de Unknown

En los pocos casos donde el género era **Unknown**, se decidió imputar este valor con **Hombre**, ya que ambos clientes superaban los **70 años** y la proporción de hombres es la mayor.

1.1.6. Region Code

La variable region code en el conjunto de datos indica la provincia de residencia de los clientes. Al igual que con otras variables, es crucial asegurarse de que esta información esté bien representada y limpia, ya que impactará en el análisis de segmentación y en el entendimiento del comportamiento del cliente según su ubicación geográfica. A continuación, se detallan los pasos realizados para limpiar y tratar esta variable.

Inicialización y Manejo de Valores Nulos

1. **Asignación de Unknown**:
    - Se identificaron y asignaron los valores nulos en region_code como 'Unknown'.
    - Esto se logró mediante el uso de cat.add_categories() para añadir 'Unknown' a las categorías existentes y luego reemplazar los valores nulos usando fillna().

2. **Conteo de Clientes**:

- Se contabilizaron los clientes por region_code para identificar cuántos clientes tenían códigos de región constantes, cuántos presentaban valores nulos, y cuántos tenían códigos mixtos (cambios en la región).
- Se encontraron **454,305 clientes con el mismo código de región**, **0 clientes con valores nulos** en region_code (excluyendo mixtos), y **2,068 clientes con códigos de región mixtos**. Este último grupo indica que algunos clientes han cambiado su región de residencia en diferentes momentos.

```
# Contar cuántos valores únicos de 'region_code' tiene cada cliente
estado_por_cliente_region = df_full.groupby('pk_cid')['region_code'].nunique()

# Identificar clientes con múltiples códigos de región (cambios de región)
clientes_mixtos_region = estado_por_cliente_region[estado_por_cliente_region > 1].index

# Excluir los clientes mixtos del análisis de las otras categorías
df_no_mixtos_region = df_full[~df_full['pk_cid'].isin(clientes_mixtos_region)]

# Identificar clientes que tienen siempre el mismo código de región
clientes_mismo_region = df_no_mixtos_region.groupby('pk_cid')['region_code'].nunique()
clientes_mismo_region = clientes_mismo_region[clientes_mismo_region == 1].index

# Identificar clientes con valores NaN en 'region_code' (excluyendo clientes mixtos)
clientes_con_nan_region = df_no_mixtos_region[df_no_mixtos_region['region_code'].isna()]['pk_cid'].unique()

# Contar el número de clientes en cada categoría (excluyendo mixtos)
n_mismo_region = len(clientes_mismo_region)
n_con_nan_region = len(clientes_con_nan_region)
n_mixtos_region = len(clientes_mixtos_region)

# Mostrar los resultados
print(f"Clientes que tienen siempre el mismo código de región (excluyendo mixtos): {n_mismo_region}")
print(f"Clientes con valores NaN en 'region_code' (excluyendo mixtos): {n_con_nan_region}")
print(f"Clientes con códigos de región mixtos (diferentes códigos): {n_mixtos_region}")
```

```
Clientes que tienen siempre el mismo código de región (excluyendo mixtos): 454305
Clientes con valores NaN en 'region_code' (excluyendo mixtos): 0
Clientes con códigos de región mixtos (diferentes códigos): 2068
```

Sustitución de Valores

3. **Sustitución de Unknown**:

- Para los casos de region_code que tenían el valor 'Unknown' y cuyo country_id era 'ES', se decidió sustituir por **28**, que corresponde a Madrid, ya que esta es la provincia más poblada y representativa de España. Para aquellos que no pertenecen a España, se sustituyó por **00**, indicando que no son clientes locales.
- Esta decisión se basa en el principio de que mantener la información más relevante y actualizada proporciona una visión más precisa del cliente.

4. **Mapeo de Códigos a Nombres**:

- Se realizó un mapeo adicional para convertir los códigos de región a sus nombres correspondientes. Esto facilita la interpretación y el análisis posterior, ya que trabajar con nombres en lugar de códigos es generalmente más comprensible.

5. **Asignación de 'Extranjero'**:

- En caso de que el region_code siguiera siendo nulo después de las imputaciones, se rellenó con el valor **'Extranjero'** para indicar que el cliente reside fuera de España. Esto es importante para clasificar adecuadamente a los clientes que no están localizados en las provincias españolas.

Visualización de la Distribución Regional

6. **Distribución de Usuarios en España**:

- Se creó un gráfico interactivo de dispersión en un mapa para visualizar la distribución de usuarios en España. Los puntos en el mapa representaban el número de clientes por cada provincia, ayudando a identificar áreas con mayor concentración de clientes.
- Este análisis geográfico es útil para la toma de decisiones estratégicas en marketing y segmentación.

1.1.7. Salary

La variable salary representa los ingresos brutos de la unidad familiar de cada cliente y es fundamental para evaluar el potencial financiero y el comportamiento de consumo de los clientes. A continuación, se detallan los pasos realizados para limpiar y preprocesar esta variable.

Identificación de Valores Nulos y Mixtos

1. **Conteo de Valores Únicos**:

- Se realizó un análisis para determinar cuántos valores únicos de salary tenía cada cliente. Esto permitió identificar a los clientes con múltiples valores de salario (indicando cambios en sus ingresos) y aquellos que tenían valores nulos.

- Se encontraron **299,443 clientes con el mismo salario**, **156,930 clientes con valores nulos** y **0 clientes con valores de salario mixtos**. Este último resultado sugiere que la mayoría de los clientes tienen un único valor de salario registrado.

```
# Contar cuántos valores únicos de 'salary' tiene cada cliente
estado_por_cliente_salary = df_full.groupby('pk_cid')['salary'].nunique()

# Identificar clientes con múltiples valores de salario (cambios en el salario)
clientes_mixtos_salary = estado_por_cliente_salary[estado_por_cliente_salary > 1].index

# Excluir los clientes mixtos del análisis de las otras categorías
df_no_mixtos_salary = df_full[~df_full['pk_cid'].isin(clientes_mixtos_salary)]

# Identificar clientes que tienen siempre el mismo salario
clientes_mismo_salary = df_no_mixtos_salary.groupby('pk_cid')['salary'].nunique()
clientes_mismo_salary = clientes_mismo_salary[clientes_mismo_salary == 1].index

# Identificar clientes con valores NaN en 'salary' (excluyendo clientes mixtos)
clientes_con_nan_salary = df_no_mixtos_salary[df_no_mixtos_salary['salary'].isna()]['pk_cid'].unique()

# Contar el número de clientes en cada categoría (excluyendo mixtos)
n_mismo_salary = len(clientes_mismo_salary)
n_con_nan_salary = len(clientes_con_nan_salary)
n_mixtos_salary = len(clientes_mixtos_salary)

# Mostrar los resultados
print(f"Clientes que tienen siempre el mismo salario: {n_mismo_salary}")
print(f"Clientes con valores NaN en 'salary' en todas las ingestas: {n_con_nan_salary}")
print(f"Clientes con valores de salario mixtos, varían con la ingesta de datos (diferentes valores): {n_mixtos_salary}")
```

```
Clientes que tienen siempre el mismo salario: 299443
Clientes con valores NaN en 'salary' en todas las ingestas: 156930
Clientes con valores de salario mixtos, varían con la ingesta de datos (diferentes valores): 0
```

2. **Descripción Estadística**:

Se generó un resumen estadístico de la variable salary, que reveló que el salario medio era de **115,816.72**, con un rango amplio que va desde **1,202.73** hasta **28,894,395.51**. Este rango amplio puede ser indicativo de outliers o errores en la captura de datos, lo que requiere atención adicional.

3. **Imputación de Valores Nulos**

Para abordar la problemática de los valores nulos, se optó por aplicar una estrategia de imputación basada en el **salario mediano**. La elección de la mediana es estratégica, ya que proporciona una medida más robusta en presencia de valores atípicos.

Creación de Grupos de Edad

Como los valores nulos están presentes en varios grupos de edad y países, se decidió segmentar a los clientes en grupos de edad definidos previamente, a saber:

```python
# Definir función para categorizar según grupos de edad
def categorizar_edad(edad):
    if edad < 18:
        return 'Menores'
    elif 18 <= edad < 25:
        return 'Jóvenes'
    elif 25 <= edad < 40:
        return 'Adultos jóvenes'
    elif 40 <= edad < 55:
        return 'Adultos'
    elif 55 <= edad < 65:
        return 'Adultos mayores'
    elif 65 <= edad < 75:
        return 'Ancianos'
    else:
        return 'Longevos'

# Crear nueva columna 'grupo_edad' en el DataFrame
df_full['grupo_edad'] = df_full['age'].apply(categorizar_edad)
```

La imputación de salarios nulos se realizó en dos pasos clave:

- **Calcular la Mediana**: Se calculó la mediana de salary para cada combinación de país, grupo de edad, segmento y código de región en el DataFrame sin duplicados. Este paso es crucial para garantizar que la mediana sea representativa y no se vea afectada por valores atípicos.

- **Imputar los Valores Nulos**: Se aplicó una función que, para cada fila del DataFrame, si el salario era nulo, se reemplazaba con la mediana correspondiente calculada en el paso anterior. Este método asegura que las imputaciones sean específicas y relevantes al contexto del cliente.

Comprobación de Resultados

A pesar de los esfuerzos por imputar los valores nulos, se constató que todavía quedaban algunos casos con salary como NaN. Para abordar estos restantes **1,856** casos, se decidió aplicar otra capa de imputación usando la mediana de salary agrupando por **grupo de edad** y **país**. Este enfoque fue seleccionado para maximizar la relevancia de los salarios imputados.

Se creó un nuevo DataFrame con las medianas correspondientes y se utilizó para reemplazar los valores nulos restantes. Tras la imputación, se realizó una verificación final para asegurar que no quedaran valores nulos en salary.

Este riguroso proceso de imputación no solo proporciona un enfoque sistemático para manejar los valores nulos en salary, sino que también permite asegurar que el análisis posterior se base en datos más completos y representativos. A través de la creación de grupos de edad y la utilización de la mediana como método de imputación, se busca construir un modelo más preciso y útil para la toma de decisiones en easyMoney.

1.1.8. Resumen de Imputaciones

1. Columnas Creadas

- **Mes_partition y Mes_nombre_partition**: Se crearon estas columnas para identificar el mes de pk_partition. Esto puede ser relevante en el futuro si se detectan estacionalidades en el comportamiento de los clientes a lo largo de los meses.
- **Grupo_edad**: Se segmentaron las edades en los siguientes grupos, lo cual es útil para un análisis más enfocado:

- **Menores**: Personas menores de dieciocho años.
- **Jóvenes**: Personas mayores de dieciocho años y menores de veinticinco.
- **Adultos jóvenes**: Personas mayores de veinticinco y menores de cuarenta años.
- **Adultos**: Personas mayores de cuarenta y menores de cincuenta y cinco.
- **Adultos mayores**: Personas de cincuenta y cinco a sesenta y cinco años.
- **Ancianos**: Personas de sesenta y cinco a setenta y cinco años.
- **Longevos**: Personas de setenta y cinco años en adelante.

2. Asignaciones de Valores en Variables

Edad (Age):

- **Variaciones + 3 años**: Se sustituyeron todos los valores de Age por el último valor conocido. Esto se justifica en que, se considera que los errores de tipificación se corrigen con el tiempo; así, una entrada más reciente es más probable que sea correcta.
- **Variación Negativa**: Se sustituyeron las edades con variaciones negativas por el último valor registrado para cada usuario, asegurando que se mantenga la consistencia en los datos.

Country_id Mixto: Para los casos de country_id mixto (25 clientes), se asignó el último país registrado para su pk_cid. Esto se debe a que se presume que la variabilidad en estos datos es un error en la recopilación de datos. Suponiendo que los cambios de residencia fueran reales, el salario debería verse afectado, lo que no se observa en estos casos. Por ello, se mantuvo la información más actualizada.

Deceased: La variable deceased presenta una variabilidad despreciable. Por lo tanto, no se considera útil para la construcción de modelos, ya que no aporta capacidad predictiva. Además, dada su alta correlación con otras variables, como el grupo de edad, es prudente eliminarla para evitar problemas de multicolinealidad en los modelos.

Gender:

- **Unknown Constante**: Se imputaron los valores Unknown y se cambiaron por "H" (Hombre) para los dos casos que superan los 70 años, considerando que es poco probable que estas personas se identifiquen como género fluido. La categoría masculina presenta la mayor proporción, lo que refuerza esta decisión.

- **Unknown Mixto**: Para los registros con gender mixto, se imputó el último género conocido.
- **Géneros Mixtos**: Se modificaron los registros de género mixto tomando el último valor registrado, ya que se considera que la última entrada es la más actualizada y, por ende, la más fiable.

Region_code:

- **Region Unknown**: Los registros con region_code como "Unknown" se sustituyeron por "28" en el caso de España (correspondiente a Madrid) y "00" para clientes fuera de España. Esto establece un valor más coherente para el análisis posterior.
- Posteriormente, se reemplazaron los códigos de provincia por el nombre correspondiente, y se etiquetaron como "Extranjero" aquellos registros que no pertenecían a España.

Imputación de Salario:

- **Imputación de Nulos**: Se decidió aplicar el salario mediano, segmentado por grupo de edad, país de residencia, región de residencia y actividad comercial. Esta imputación se llevó a cabo de forma iterativa; en caso de que no se encontrara un salario mediano aplicable, se iba reduciendo la restricción de las variables en el siguiente orden:
    - Primero se utilizaron **grupo_edad** y **country_id**.
    - Si aún quedaban casos con salarios nulos, se procedió a utilizar únicamente **grupo_edad**.

Es importante destacar que todas las imputaciones de salario se realizaron sobre el conjunto sin duplicados para garantizar que la mediana no se viera afectada por valores atípicos, lo que podría distorsionar los resultados finales.

1.2. Product Database

En este segmento del análisis, nos enfocamos en las variables relacionadas con los productos ofrecidos por easyMoney. A continuación, se presentan las variables y su respectiva explicación:

Variable	**Explicación**
credit_card	Tarjetas de crédito
debit_card	Tarjetas de débito
em_account_p	Cuenta easyMoney+
em_account_pp	Cuenta easyMoney++
em_acount	Cuenta easyMoney
emc_account	Cuenta easyMoney Crypto
funds	Fondos de inversión
loans	Préstamos
long_term_deposit	Depósitos a largo plazo
mortgage	Hipotecas
payroll	Domiciliaciones
payroll_account	Cuenta bonificada por domiciliaciones
pension_plan	Plan de pensiones
securities	Valores
short_term_deposit	Depósitos a corto plazo

1.2.1. Análisis Preliminar

Para iniciar el análisis, cargamos el dataset de productos desde AWS S3. Realizamos un análisis preliminar del DataFrame para observar la distribución de los datos y los valores nulos. Los resultados mostraron que hay **61 nulos** en las columnas **payroll** y **pension_plan** dentro de un dataset que contiene más de 5 millones de filas. Se procederá a imputar estos valores nulos utilizando la moda. Posteriormente, confirmamos que no haya valores nulos en el DataFrame:

```python
def build_my_info_table(df):
    # Seleccionar las columnas numéricas del DataFrame
    numerical_columns = df.select_dtypes(include=[np.number])

    # Crear un DataFrame con información detallada sobre las columnas
    df_info = pd.DataFrame({
        # Nombre de la columna
        'column':   [col             for col in df.columns],
        # Tipo de datos de la columna
        'dtype':    [df[col].dtype   for col in df.columns],
        # Número de valores no nulos en la columna
        'count':    [df[col].count() for col in df.columns],

        # Estadísticas descriptivas para columnas numéricas
        'mean':     [round(df[col].mean(), 1)         if col in numerical_columns else '' for col in df.columns],
        'std':      [round(df[col].std(), 1)          if col in numerical_columns else '' for col in df.columns],
        'min':      [round(df[col].min(), 1)          if col in numerical_columns else '' for col in df.columns],
        '25%':      [round(df[col].quantile(0.25), 1) if col in numerical_columns else '' for col in df.columns],
        '50%':      [round(df[col].median(), 1)       if col in numerical_columns else '' for col in df.columns],
        '75%':      [round(df[col].quantile(0.75), 1) if col in numerical_columns else '' for col in df.columns],
        'max':      [round(df[col].max(), 1)          if col in numerical_columns else '' for col in df.columns],

        # Número de valores únicos en la columna
        'nunique':  [df[col].nunique()                                                   for col in df.columns],

        # Lista de valores únicos si el número de valores únicos es menor que 30, de lo contrario, cadena vacía
        'unique':   [list(df[col].unique()) if df[col].nunique() < 10 else ''   for col in df.columns],

        # Porcentaje de cardinalidad (número de valores únicos)
        'cardinality %': [round(df[col].nunique() / df.shape[0] * 100, 1)      for col in df.columns],

        # Moda (valor más frecuente) en la columna
        'mode':     [df[col].mode()[0]                                           for col in df.columns],

        # Porcentaje de ocurrencias de la moda en la columna
        'mode %':   [round(df[col].value_counts().max() * 100 / df.shape[0], 1) for col in df.columns],

        # Número de valores nulos en la columna
        'null_count': [df[col].isnull().sum() for col in df.columns],

        # Porcentaje de valores nulos en la columna
        'null %':   [round(df[col].isnull().mean() * 100, 1)                     for col in df.columns],
    })
    return df_info
```

```python
# reemplazamos los valores nulos por la moda
prod_df['payroll'].fillna(prod_df['payroll'].mode()[0], inplace=True)
prod_df['pension_plan'].fillna(prod_df['pension_plan'].mode()[0], inplace=True)
# comprobamos que no haya valores nulos
prod_df.isna().sum().sum()
```

Tabla de Información General

Eliminamos columnas que no aportan información *'Unnamed: 0'* y creamos una tabla de información detallada:

	column	dtype	count	mean	std	min	25%	50%	75%	max	nunique	unique	cardinality %	mode	mode %	null_count	null %
0	pk_cid	int64	5962924	1234929.880000	162302.000000	15891	1112532.000000	1231097.000000	1352339.000000	1553689	456373		7.700000	17457	0.000000	0	0.000000
1	pk_partition	object	5962924								17		0.000000	2019-05-28	7.400000	0	0.000000
2	short_term_deposit	int64	5962924	0.000000	0.100000	0	0.000000	0.000000	0.000000	1	2	[0, 1]	0.000000	0	99.700000	0	0.000000
3	loans	int64	5962924	0.000000	0.000000	0	0.000000	0.000000	0.000000	1	2	[0, 1]	0.000000	0	100.000000	0	0.000000
4	mortgage	int64	5962924	0.000000	0.000000	0	0.000000	0.000000	0.000000	1	2	[0, 1]	0.000000	0	100.000000	0	0.000000
5	funds	int64	5962924	0.000000	0.100000	0	0.000000	0.000000	0.000000	1	2	[0, 1]	0.000000	0	99.700000	0	0.000000
6	securities	int64	5962924	0.000000	0.100000	0	0.000000	0.000000	0.000000	1	2	[0, 1]	0.000000	0	99.600000	0	0.000000
7	long_term_deposit	int64	5962924	0.000000	0.100000	0	0.000000	0.000000	0.000000	1	2	[0, 1]	0.000000	0	98.300000	0	0.000000
8	em_account_pp	int64	5962924	0.000000	0.000000	0	0.000000	0.000000	0.000000	0	1	[0]	0.000000	0	100.000000	0	0.000000
9	credit_card	int64	5962924	0.000000	0.100000	0	0.000000	0.000000	0.000000	1	2	[0, 1]	0.000000	0	98.800000	0	0.000000

Esto nos permite extraer las siguientes conclusiones de la tabla:

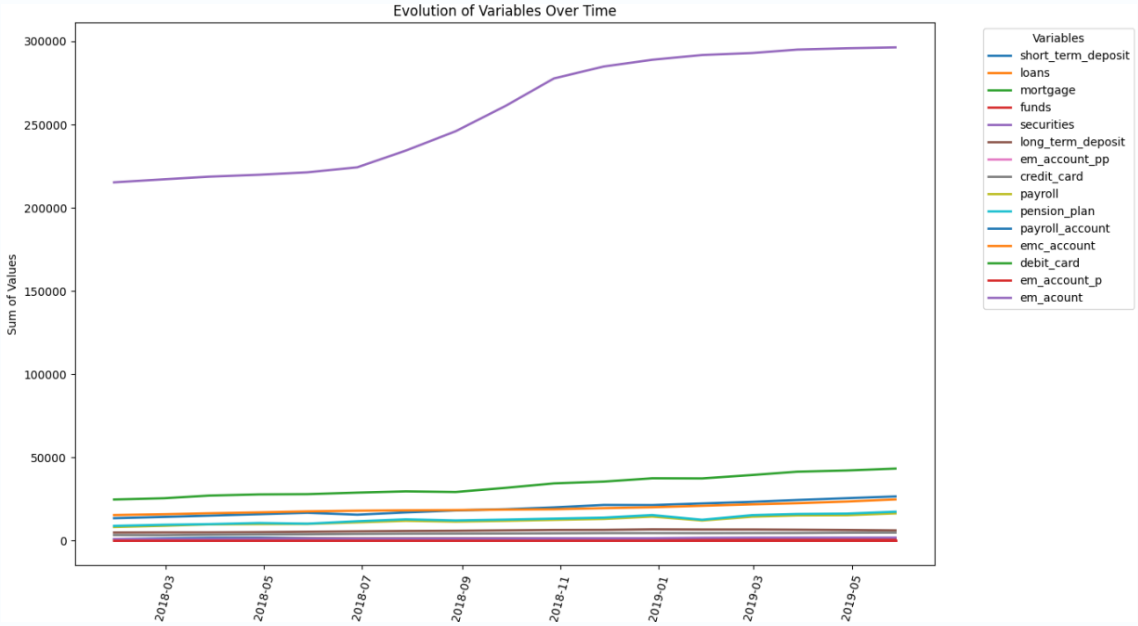
- Se han corregido todos los valores nulos observados.
- A excepción de la cuenta easyMoney, el resto de productos tienen una tasa de contratación/uso menor al 10% sobre el total de particiones.
- Algunos casos concretos, como **loans**, **mortgage**, **em_account_pp** y **em_account_p**, presentan una tasa de contratación del 0%.

Modificación de Tipos de Datos

Para optimizar el uso de memoria, modificamos el tipo de dato de las columnas relacionadas con los productos financieros.

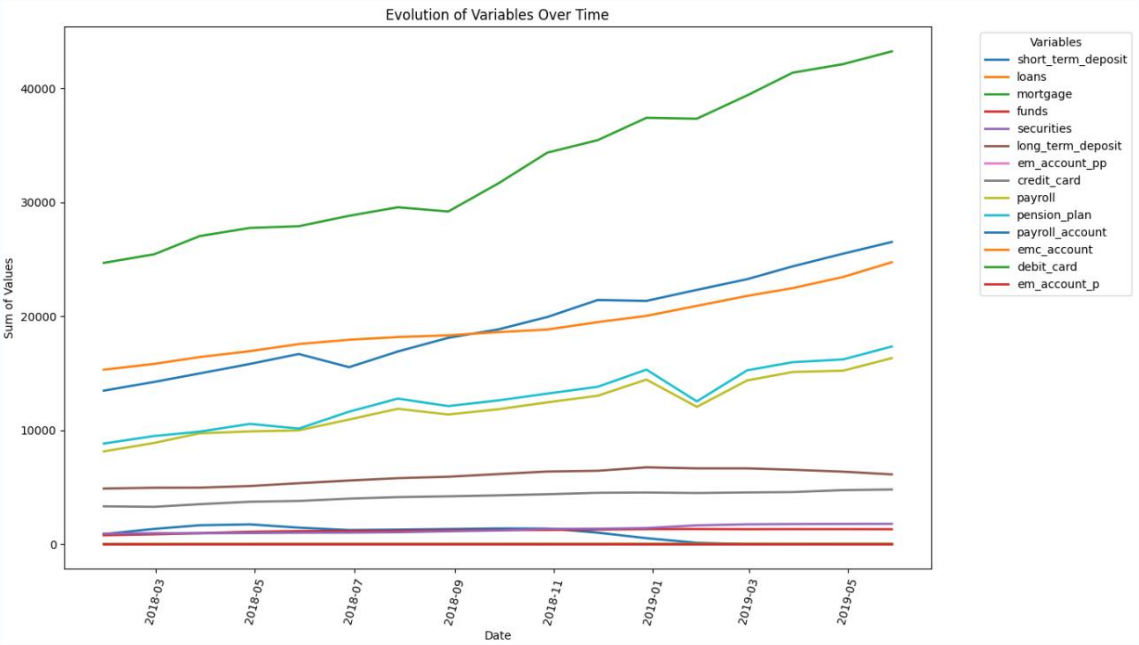
1.2.2. Comportamiento de los Productos a lo Largo de las Particiones

Agrupamos por **pk_partition** y sumamos las demás columnas para observar la evolución de los productos a lo largo del tiempo:



El gráfico resultante revela que **em_account** es el producto más utilizado, mientras que otros productos como **loans** y **mortgage** tienen una adopción considerablemente menor. Al eliminar **em_account** del análisis para observar el comportamiento de los demás productos, encontramos que:

- Productos como **tarjetas de débito** y **cuentas de nómina** presentan un crecimiento constante.
- Los productos como **fondos**, **depósitos a largo plazo** e **hipotecas** tienen un uso limitado y constante.



El gráfico muestra la evolución del uso de varios productos financieros a lo largo del tiempo:

- Productos más populares: Las **tarjetas de débito**, **las cuentas de nómina** y los **préstamos** son los más utilizados por los clientes, mostrando un crecimiento constante.
- Uso moderado: Las **tarjetas de crédito** y la **nómina** tienen un uso estable, con un crecimiento más leve.
- Bajo uso: Productos como **valores, fondos, depósitos a largo plazo e hipotecas** tienen un uso limitado y constante, indicando menor adopción.
- Adopción mínima: Productos como **em_account_pp, em_account_p, y emc_account** tienen una adopción casi inexistente.

En resumen, mientras algunos productos financieros son ampliamente utilizados, otros presentan oportunidades significativas para aumentar su adopción entre los clientes.

1.2.3. Análisis de Clientes Según Tipos de Producto

Al analizar el comportamiento de los productos según tipos de clientes, se observa que existen clientes que repiten en cada partición, así como otros que presentan un comportamiento variable.

La distribución de productos para cada cliente se evalúa y se generan gráficos que muestran el porcentaje de clientes que siempre utilizan un producto, aquellos que no lo usan y los que tienen un comportamiento mixto.

La imagen obtenida presenta un análisis visual del comportamiento de los productos ofrecidos por easyMoney. En el primer gráfico se observa la distribución porcentual de los clientes categorizados según su uso de los productos: aquellos que siempre utilizan el producto, aquellos que nunca lo utilizan (Siempre 0) y aquellos que presentan un comportamiento variable. Esta

categorización es crucial para entender las dinámicas de uso y aceptación de los diferentes productos en el portafolio de la empresa.

En general, la mayoría de los productos financieros analizados muestran una notable cantidad de clientes que no los utilizan "Siempre 0". Este hallazgo indica que hay un área significativa de mejora para la empresa, ya que existe una oportunidad para aumentar la adopción de estos productos entre los clientes. La empresa puede investigar las razones detrás de esta falta de uso: ¿son los productos poco conocidos? ¿Hay barreras de acceso? ¿Los clientes no ven el valor en los mismos? Estas preguntas pueden ayudar a identificar estrategias de marketing y educación que fomenten la utilización de productos financieros.

Por otro lado, productos como **em_account** han demostrado ser ampliamente utilizados y mantienen un uso constante. Esto se debe a que fue el producto original con el que comenzó la empresa, antes de pivotar hacia otros productos financieros.

Análisis Detallado del Comportamiento por Tipo de Cliente

El resto de los gráficos analizan más a fondo el comportamiento de los diferentes tipos de clientes a lo largo de las particiones temporales. Este análisis revela patrones específicos que son importantes para entender cómo los productos están siendo utilizados en el tiempo:

- **Clientes que nunca usan el producto**: El comportamiento de estos clientes es notablemente consistente a través de los productos. Antes de julio de 2018, el número de clientes que no utilizaban los productos se mantenía relativamente estable. Sin embargo, a partir de esta fecha, se ha observado un aumento en la inclusión de nuevos clientes en esta categoría. Este cambio puede indicar un desinterés creciente en los productos por parte de algunos usuarios existentes o la incapacidad de la empresa para atraer a nuevos clientes hacia la utilización de productos que previamente no usaban.
- **Clientes que siempre usan el producto**: Estos clientes presentan una tendencia estable, aunque hay una ligera inclinación ascendente en algunos productos después de 2018. Este crecimiento sugiere que, con el tiempo, algunos clientes han comenzado a adoptar productos que anteriormente no utilizaban de manera consistente.
- **Clientes con comportamiento variable**: Este grupo es especialmente interesante, ya que muestra fluctuaciones en su uso de productos. Inicialmente, muchos de estos clientes no utilizaban los productos, pero a partir de julio de 2018, se ha notado una mejora en la aceptación. Sin embargo, algunos productos como **em_account**, **long_term_deposit**, **loans** y **short_term_deposit** han visto una reducción en el interés por parte de los clientes. Esto puede indicar que, aunque la aceptación de productos está aumentando, ciertos productos están perdiendo su atractivo o que no cumplen con las expectativas de los usuarios. La empresa debe investigar más a fondo estos casos para entender las razones detrás de esta disminución.

1.2.4. Suma de Productos Contratados

Para entender mejor la interacción de los clientes con los productos financieros ofrecidos por EasyMoney, hemos creado una nueva columna en el DataFrame prod_df, denominada num_products_contracts, que representa la suma de productos contratados por cada registro. Este paso nos permite tener una visión más clara del comportamiento de los clientes en relación a la diversidad de productos que utilizan.

Se observa que la mayoría de los clientes (aproximadamente el 57%) han contratado al menos un producto, mientras que el 23% no tiene ningún producto contratado. Esta información es crucial, ya que sugiere que una parte significativa de la base de clientes todavía no ha explorado los productos disponibles, lo que podría ser una oportunidad para aumentar la adopción de productos.

Agrupando los Clientes por Productos Contratados

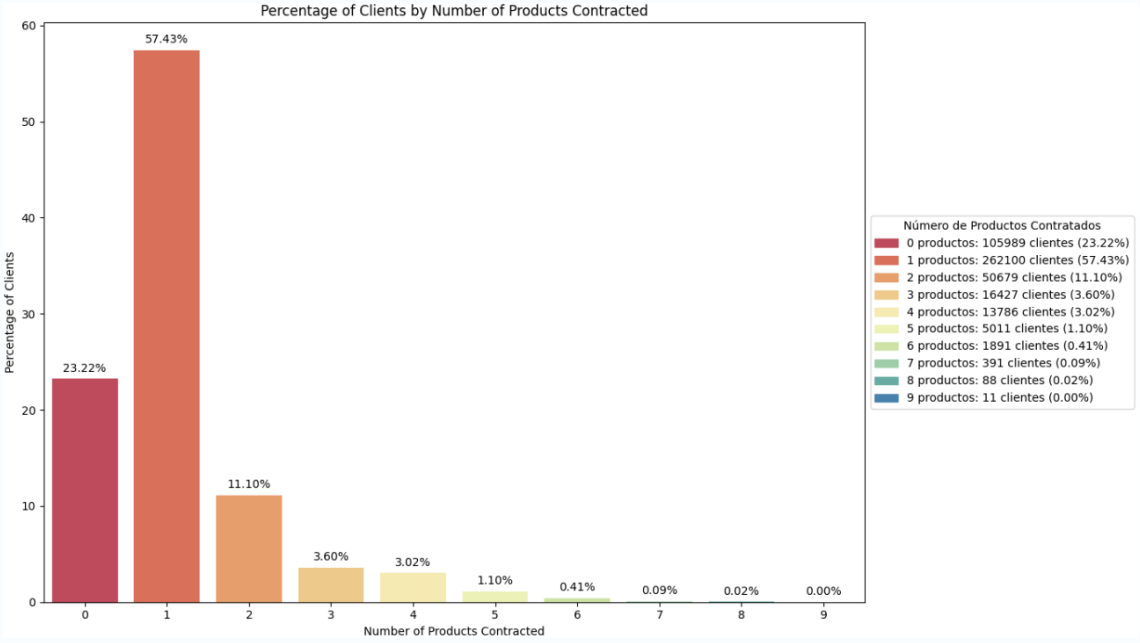
A continuación, agrupamos a los clientes por el número máximo de productos contratados a lo largo de su historia. Esto nos permitirá identificar el número total de clientes que poseen un determinado número de productos:

```
# Agrupar por 'pk_cid' para obtener el máximo número de productos contratados por cliente
max_products_per_client = prod_df.groupby('pk_cid')['num_products_contracts'].max()

# Contar cuántos clientes tienen un determinado número de servicios contratados
clients_per_service_count = max_products_per_client.value_counts().reset_index()

# Renombrar las columnas para mayor claridad
clients_per_service_count.columns = ['num_products_contracted', 'num_clients']

# Ordenar por número de productos contratados
clients_per_service_count.sort_values(by='num_products_contracted', inplace=True)
clients_per_service_count
```

	num_products_contracted	num_clients
1	0	105989
0	1	262100
2	2	50679
3	3	16427
4	4	13786
5	5	5011
6	6	1891
7	7	391
8	8	88
9	9	11

Porcentaje de Clientes por Número de Productos Contratados

Para profundizar en el análisis, calculamos el porcentaje de clientes según el número de productos contratados. A continuación, se presenta un gráfico de barras que ilustra la distribución porcentual de los clientes en función del número de productos contratados:



Analizando el gráfico resultante, se pueden extraer las siguientes conclusiones:

- **Más del 57% de los Clientes Tienen al Menos un Producto Contratado**: Este hallazgo indica que una parte significativa de la clientela ha comenzado a explorar los productos de easyMoney. Sin embargo, también implica que existe un porcentaje considerable de clientes (23%) que no ha contratado ningún producto, lo que podría reflejar desinterés, falta de conocimiento o barreras de acceso.
- **Uso de Productos por Parte de los Clientes**: El gráfico muestra que la mayoría de los clientes solo tiene uno o dos productos contratados. Esto sugiere que, si bien hay una adopción inicial, es necesario trabajar en estrategias que fomenten la contratación de más productos por parte de los clientes existentes.
- **Oportunidades para la Empresa**: La gran cantidad de clientes que no han contratado ningún producto presenta una clara oportunidad para easyMoney. Invertir en campañas de marketing y educación puede ser clave para aumentar la adopción de productos.

1.2.5. Creación de Variables por Segmentación de Productos

Para entender mejor el rendimiento de los productos ofrecidos por easyMoney, se sugiere agrupar los productos por familias, según su tipología y los ingresos que generan. Se crean las siguientes variables:

1. **p_cuenta_bancaria**: Esta variable se establece en 1 si el cliente tiene contratado en esa pk_partition cualquiera de los siguientes productos: em_account_pp, payroll, payroll_account, emc_account, debit_card, em_account_p, em_acount. En caso contrario, se asigna 0.
2. **cuentas_sum**: Esta columna sumará el número de productos de cuentas bancarias contratados por el usuario en ese momento.

3. **p_inversion**: Al igual que la variable anterior, se asigna un 1 si el cliente tiene al menos un producto de inversión en su pk_partition, y 0 en caso contrario. Los productos considerados son: short_term_deposit, funds, securities, long_term_deposit, pension_plan.
4. **inversion_sum**: Esta variable contará el número de productos de inversión que tiene el usuario.
5. **p_financiacion**: Esta columna recogerá un 1 si el cliente tiene contratado en esa pk_partition cualquiera de los siguientes productos: loans, mortgage, credit_card, y 0 si no.
6. **financiacion_sum**: Sumará el número de productos de financiación contratados por el usuario.

```
# Crear las nuevas columnas

# p_cuenta_bancaria: 1 si tiene al menos un producto de cuenta bancaria, 0 si no
prod_df['p_cuenta_bancaria'] = prod_df[cuenta_bancaria_products].max(axis=1)

# cuentas_sum: número de productos de cuenta bancaria que tiene el cliente
prod_df['cuentas_sum'] = prod_df[cuenta_bancaria_products].sum(axis=1)

# p_inversion: 1 si tiene al menos un producto de inversión, 0 si no
prod_df['p_inversion'] = prod_df[inversion_products].max(axis=1)

# inversion_sum: número de productos de inversión que tiene el cliente
prod_df['inversion_sum'] = prod_df[inversion_products].sum(axis=1)

# p_financiacion: 1 si tiene al menos un producto de financiación, 0 si no
prod_df['p_financiacion'] = prod_df[financiacion_products].max(axis=1)

# financiacion_sum: número de productos de financiación que tiene el cliente
prod_df['financiacion_sum'] = prod_df[financiacion_products].sum(axis=1)
```

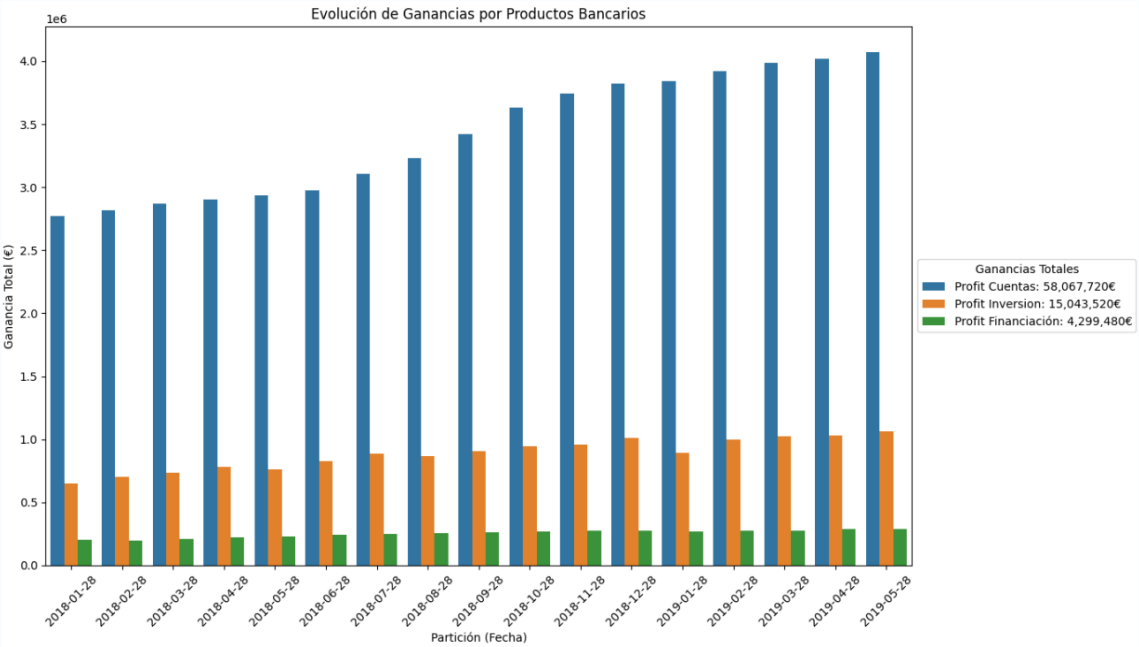
Una vez definidas las variables por tipología de productos, se procede a calcular los beneficios asociados a cada grupo. Los beneficios se estiman de la siguiente manera:

- **Profit Cuentas**: Se asigna un beneficio de 10€ por cada cuenta bancaria contratada.
- **Profit Inversión**: Se asigna un beneficio de 40€ por cada producto de inversión.
- **Profit Financiación**: Se asigna un beneficio de 60€ por cada producto de financiación.

```
# Calcular los beneficios asociados a cada grupo de productos

# profit_cuentas: 10€ * cuenta_sum
prod_df['profit_cuentas'] = 10 * prod_df['cuentas_sum']

# profit_inversion: 40€ * inversion_sum
prod_df['profit_inversion'] = 40 * prod_df['inversion_sum']

# profit_financiacion: 60€ * financiacion_sum
prod_df['profit_financiacion'] = 60 * prod_df['financiacion_sum']
```

Análisis de la Evolución de Ganancias

Para realizar un análisis más detallado sobre la evolución de las ganancias asociadas a los productos, se agrupan los datos por pk_partition y se suman las ganancias de cada tipo de producto. Luego, creamos un gráfico que ilustre la evolución de las ganancias a lo largo del tiempo:



La gráfica presentada muestra la evolución de las ganancias totales generadas por los diferentes grupos de productos bancarios a lo largo del tiempo, desde enero de 2018 hasta marzo de 2019.

1. **Ganancias Totales**: Las ganancias totales, muestran un crecimiento constante y significativo a lo largo del tiempo, lo que indica una tendencia positiva en la adquisición y uso de productos bancarios por parte de los clientes. Este incremento sugiere que la empresa ha logrado mejorar su oferta o fortalecer su relación con los clientes, alentando a más personas a contratar productos.

2. **Ganancias por Productos**: La gráfica incluye diferentes colores para representar las ganancias generadas por cuentas bancarias (azul), inversiones (naranja) y financiación (verde).

    - **Cuentas Bancarias**: Con ganancias totales de 58,067,720€, este grupo representa la mayor parte de los ingresos, lo que implica que los clientes están optando mayoritariamente por este tipo de productos.
    - **Inversiones**: Las ganancias de este segmento alcanzan 15,043,520€. Aunque son significativamente menores en comparación con las cuentas bancarias, el hecho de que haya ingresos procedentes de inversiones demuestra un interés por parte de los clientes en diversificar su patrimonio, lo que podría ser un área para explorar más a fondo y potenciar.
    - **Financiación**: Con 4,299,480€ en ganancias, el segmento de financiación muestra un interés moderado de los clientes en productos como préstamos y tarjetas de crédito. A pesar de ser el grupo con menores ingresos, es crucial para la salud financiera de la empresa y puede ser un área con potencial de crecimiento si se implementan estrategias adecuadas para incentivar su uso.

3. **Conclusiones**: La evolución de las ganancias indica que la empresa está en una trayectoria positiva, con un aumento notable en las ganancias totales. Sin embargo, se observa una dependencia elevada en productos de cuentas bancarias, lo que sugiere que la empresa debe seguir diversificando su oferta de productos para reducir el riesgo asociado a una única fuente de ingresos. Además, hay oportunidades para mejorar la adopción y uso de productos de inversión y financiación, lo que podría resultar en un aumento sustancial de los ingresos en el futuro.

1.2.6. Resumen

1. **Tratamiento de Valores Nulos:** Se identificaron 61 valores nulos en dos columnas clave, **pension_plan** y **payroll**. Aunque representan una pequeña fracción del total de datos, estos valores pueden afectar el análisis, se decidió reemplazar los valores nulos con la moda, dado que es una técnica efectiva para columnas categóricas en datasets de gran tamaño. Esto asegura que los registros estén completos sin introducir sesgos significativos:

2. **Identificación de Duplicados**: Una validación esencial del preprocesamiento fue la búsqueda de duplicados, los cuales pueden inflar resultados o distorsionar el análisis. No se encontraron duplicados, lo que indica que cada registro en el dataset es único y representa a un cliente o una partición temporal distinta.

3. **Popularidad de los Productos** El producto más utilizado a lo largo del tiempo por los clientes es **em_acount** (cuenta bancaria electrónica). Este comportamiento refleja una preferencia clara por parte de los usuarios hacia productos que permiten la gestión básica de sus finanzas de manera digital. Este hallazgo tiene implicaciones significativas para el diseño de futuras estrategias de marketing y desarrollo de productos.

4. **Análisis del Comportamiento de los Clientes en las Particiones** Se observó que hay clientes que mantienen su actividad en diferentes particiones del dataset. Estos clientes presentan distintos patrones de comportamiento:

   - Algunos mantienen valores constantes (0 o 1) en todas las particiones, lo que sugiere un comportamiento consistente en la contratación o no de productos.

   - Otros muestran variabilidad en su comportamiento, contratando o no productos en diferentes momentos.

5. **Agrupación de Clientes por Cantidad de Productos Contratados**: Uno de los puntos clave del análisis fue la creación de una nueva variable: **num_products_contracts**, que representa el número de productos contratados por cada cliente, con un rango que va de 0 a 9 productos contratados.

   Los resultados muestran que:

   - El **57%** de los clientes tienen al menos un producto contratado.

   - El **23%** no tiene ningún producto contratado.

   - El **20%** restante tiene más de un producto contratado.

6. **Evolución de Ganancias por Segmento de Productos**: A partir del análisis financiero, se observó un comportamiento diferencial en las ganancias generadas por cada segmento de productos financieros. Los resultados indican que:

    - **Cuentas**: Constituyen el producto más rentable, acumulando **58.067.720€**, y mostrando un crecimiento continuo a lo largo del periodo analizado.
    - **Inversiones**: Han generado **15.043.520€**, consolidándose como el segundo segmento más rentable.
    - **Financiación**: Aunque su aportación es menor, ha sumado **4.299.480€** a las ganancias totales.

    Este análisis sugiere que las cuentas bancarias son el producto financiero más valioso para la empresa, seguido por las inversiones. Las estrategias futuras podrían centrarse en potenciar estos segmentos, sin descuidar el desarrollo de productos de financiación.

1.3. Actividad Comercial

La siguiente sección está dedicada al análisis de la **actividad comercial** de los clientes en la aplicación easyMoney. El conjunto de datos correspondiente contiene información relevante sobre la interacción de los clientes con la plataforma, los canales de captación, fechas de contratación y otros detalles relacionados con su actividad comercial.

1.3.1. Descripción del dataset

El dataset proporcionado en esta sección consta de las siguientes variables:

- **active_customer**: Indicador de actividad del cliente en la aplicación.
- **entry_channel**: Canal de captación del cliente.
- **entry_date**: Fecha en la que el cliente realizó la primera contratación a través de easyMoney.
- **segment**: Segmento comercial al que pertenece el cliente.
- **pk_cid**: Identificador único del cliente.
- **pk_partition**: Fecha en la que los datos fueron ingeridos.

1.3.2. Preprocesamiento de datos

El primer paso en el preprocesamiento de los datos fue eliminar las columnas irrelevantes, como **Unnamed: 0**, que no aportaban información útil.

Se identificaron errores en las fechas de la columna **entry_date**, como la presencia de días no válidos (29 de febrero en años no bisiestos). Para corregir estos errores, se sustituyeron manualmente las fechas erróneas por las correctas:

A continuación, se generó una tabla descriptiva que resume las principales características de cada variable. Se incluyeron estadísticas como el número de valores únicos, moda, porcentajes de valores nulos, y estadísticas descriptivas para las variables numéricas.

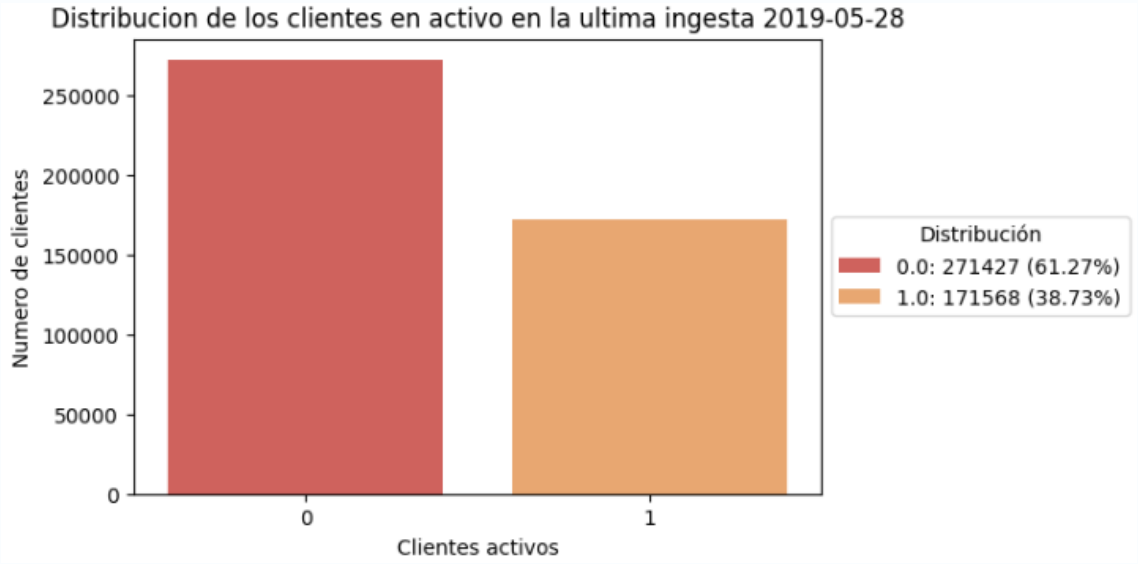
	column	dtype	count	mean	std	min	25%	50%	75%	max	nunique	unique	cardinality %	mode	mode %	null_count	null %
0	pk_cid	int64	5962924	1234929.800000	162302.000000	15891	1112532.000000	1231097.000000	1352339.000000	1553689	456373		7.700000	17457	0.000000	0	0.000000
1	pk_partition	datetime64[ns]	5962924								17		0.000000	2019-05-28 00:00:00	7.400000	0	0.000000
2	entry_date	datetime64[ns]	5962924								1497		0.000000	2017-07-28 00:00:00	1.000000	0	0.000000
3	entry_channel	category	5829891								68		0.000000	KHE	52.200000	133033	2.200000
4	active_customer	int32	5962924	0.400000	0.500000	0	0.000000	0.000000	1.000000	1	2	[1, 0]	0.000000	0	59.700000	0	0.000000
5	segment	category	5828980								3	['02 - PARTICULARES', '03 - UNIVERSITARIO', '01 - TOP', nan]	0.000000	03 - UNIVERSITARIO	65.400000	133944	2.200000

En un primer analisis, podemos comprobar bastantes datos a simple vista:

- El número total de clientes, valores únicos de "pk_cid" = 456373.
- Comprobamos que hay 17 fechas de “pk_partition” correspondientes a 17 ingestas de datos.
- Hay dos columnas con valores nulos “entry_channel” y “segment”.
- No tenemos filas duplicadas.

1.3.3. Clientes Activos e Inactivos

Para analizar la distribución de clientes activos e inactivos, se filtraron los datos de la última ingesta disponible, correspondiente al 28 de mayo de 2019. Los resultados mostraron una proporción significativa de clientes inactivos en la plataforma, con un **61% de inactividad** en ese periodo.



Este resultado refleja una problemática importante que deberá abordarse en la estrategia de retención de clientes, ya que un alto porcentaje de clientes no está utilizando activamente la plataforma.

1.3.4. Clientes duplicados por actividad

El análisis de la columna **active_customer** mostró que algunos clientes han cambiado su estado de actividad a lo largo del tiempo. Al agrupar los clientes por su estado de actividad, se identificó una discrepancia en el número de clientes únicos, lo que indica que hay clientes que presentan más de un estado de actividad en diferentes ingestas:

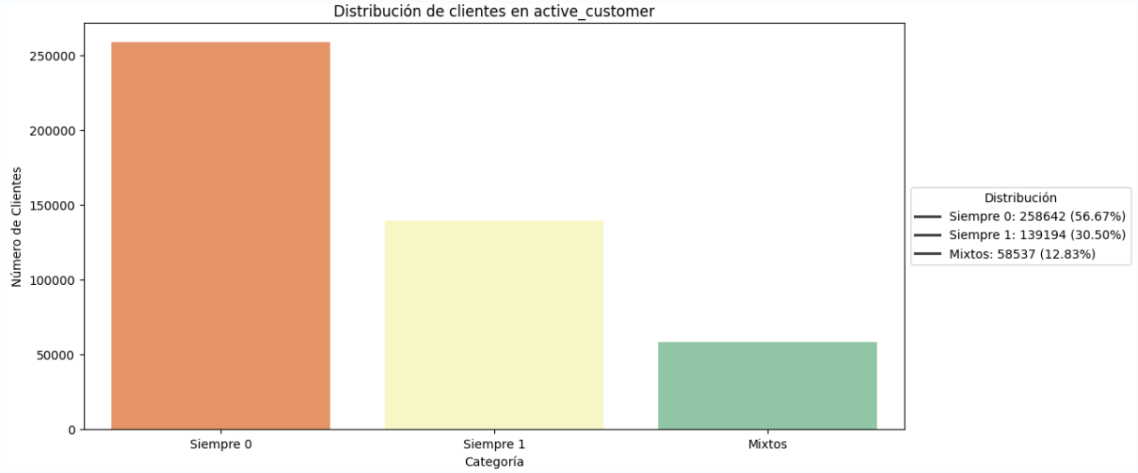
```
# Contar cuántos estados diferentes tiene cada cliente en active_customer
estado_por_cliente = ca_df.groupby('pk_cid')['active_customer'].agg(['nunique', 'count'])

# Identificar clientes con estados mixtos (0 y 1)
clientes_mixtos = estado_por_cliente[estado_por_cliente['nunique'] > 1].index

# Excluir los clientes mixtos del análisis de las otras categorías
df_no_mixtos = ca_df[~ca_df['pk_cid'].isin(clientes_mixtos)]

# Identificar clientes que son siempre 0
clientes_siempre_cero = df_no_mixtos.groupby('pk_cid')['active_customer'].min()
clientes_siempre_cero = clientes_siempre_cero[clientes_siempre_cero == 0].index

# Identificar clientes que son siempre 1
clientes_siempre_uno = df_no_mixtos.groupby('pk_cid')['active_customer'].max()
clientes_siempre_uno = clientes_siempre_uno[clientes_siempre_uno == 1].index

# Contar el número de clientes en cada categoría (excluyendo mixtos)
n_siempre_cero = len(clientes_siempre_cero)
n_siempre_uno = len(clientes_siempre_uno)

n_mixtos = len(clientes_mixtos)

# Mostrar los resultados
print(f"Clientes que son siempre 0 en 'active_customer' (excluyendo mixtos): {n_siempre_cero}")
print(f"Clientes que son siempre 1 en 'active_customer' (excluyendo mixtos): {n_siempre_uno}")
print(f"Clientes con estados mixtos (0 y 1): {n_mixtos}")
print(f"Total de clientes en el dataset: {n_siempre_cero + n_siempre_uno + n_mixtos}")
```

```
Clientes que son siempre 0 en 'active_customer' (excluyendo mixtos): 258642
Clientes que son siempre 1 en 'active_customer' (excluyendo mixtos): 139194
Clientes con estados mixtos (0 y 1): 58537
Total de clientes en el dataset: 456373
```

Esta observación refuerza la importancia de analizar el comportamiento de los clientes a lo largo del tiempo para obtener una visión más precisa de su ciclo de vida dentro de la plataforma.

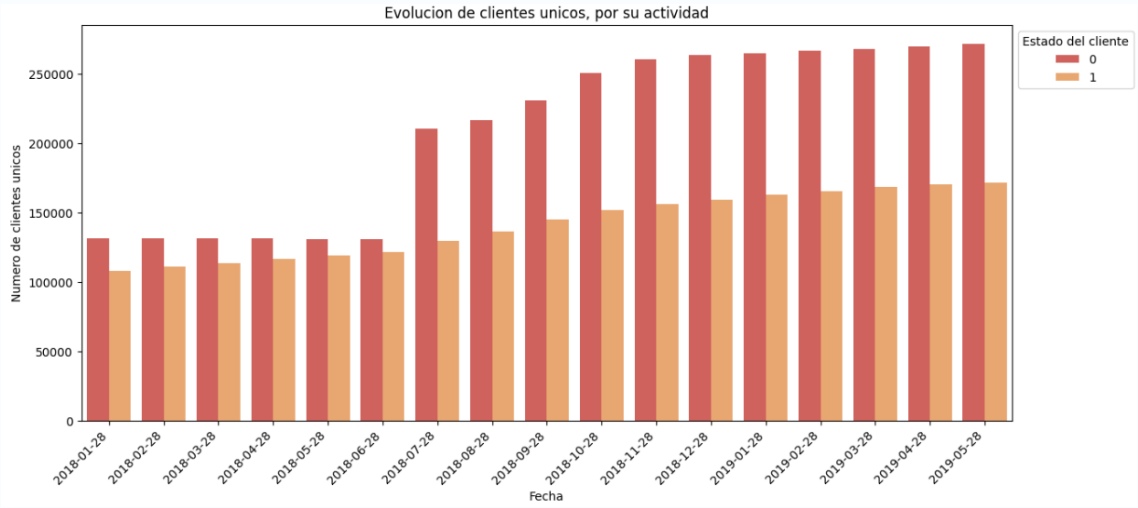
Gráfica de Distribución

La mayoría de los clientes (56.67%) se encuentran inactivos durante todo su ciclo de vida, lo cual sugiere un problema en la retención de clientes. Solo un pequeño porcentaje (12.83%) muestra un comportamiento variable en su estado de actividad.



Evolución Temporal de la Actividad

Además, se analizó la evolución temporal de la actividad de los clientes a lo largo del tiempo. Los resultados indican que, aunque la base de clientes ha crecido, una proporción significativa de nuevos clientes no se mantienen activos. Se observó un aumento en el número de clientes inactivos en julio de 2018, lo que puede requerir una estrategia de reactivación para evitar la pérdida de clientes.



1.3.5. Canal de Entrada (entry_channel)

Se exploraron los canales de entrada de los clientes para identificar patrones en la captación de usuarios. Tras el análisis de las categorías, se observó que un porcentaje significativo de los registros presentaba información mixta o faltante. Se tomó la decisión de:

1. **Reemplazar valores 'Unknown'**: Se reemplazó el valor Unknown por el último canal conocido para cada cliente.
2. **Reemplazar clientes con canales mixtos**: Los clientes que habían utilizado más de un canal de entrada fueron reasignados al canal más reciente registrado.

```python
# Contar cuántos valores únicos tiene cada cliente en la columna entry_channel
estado_por_cliente_channel = ca_df.groupby('pk_cid')['entry_channel'].agg(['nunique', 'count'])

# Identificar clientes con canales de entrada mixtos (más de un valor en entry_channel)
clientes_mixtos_channel = estado_por_cliente_channel[estado_por_cliente_channel['nunique'] > 1].index

# Excluir los clientes mixtos del análisis de las otras categorías
df_no_mixtos_channel = ca_df[~ca_df['pk_cid'].isin(clientes_mixtos_channel)]

# Identificar clientes que siempre usaron un solo canal de entrada (excluyendo mixtos)
clientes_un_canal = df_no_mixtos_channel.groupby('entry_channel')['pk_cid'].nunique()

# Contar el número de clientes en cada categoría (excluyendo mixtos)
n_un_canal = clientes_un_canal.sum()
n_mixtos_channel = len(clientes_mixtos_channel)

# Mostrar los resultados
print(f"Clientes que siempre usaron un solo canal de entrada (excluyendo mixtos): {n_un_canal}")
print(f"Clientes con canales de entrada mixtos: {n_mixtos_channel}")
```

```
Clientes que siempre usaron un solo canal de entrada (excluyendo mixtos): 331869
Clientes con canales de entrada mixtos: 124504
```

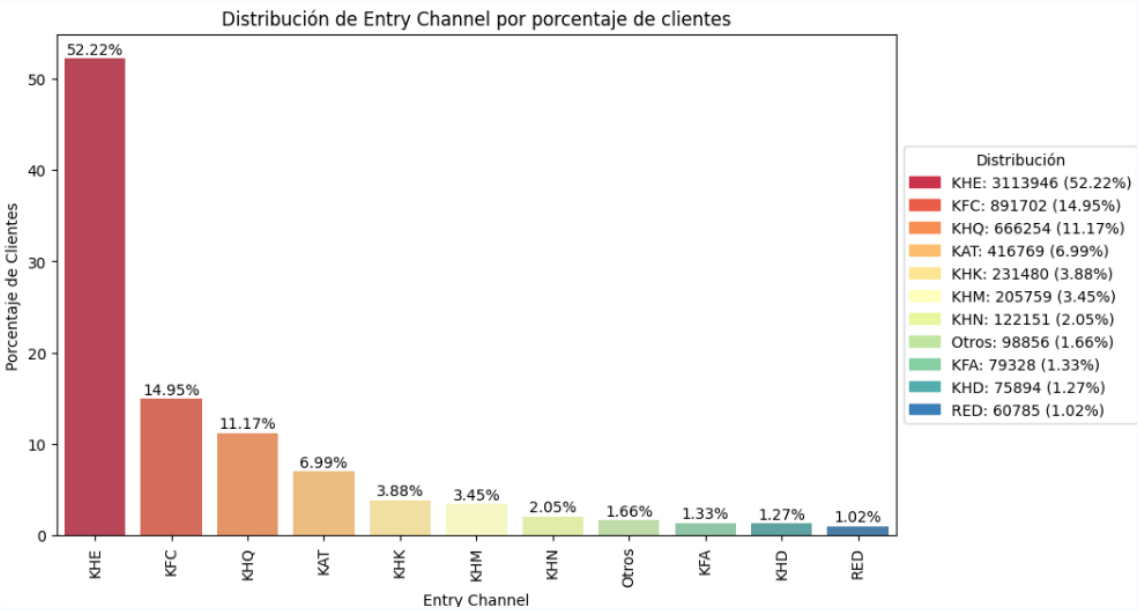
Distribución de Canales de Entrada

Una vez depurada la base de datos, se observó que la mayor parte de los clientes ingresó a través de los siguientes canales:

- **KHE**: 52.22% de los clientes.
- **KFC**: 14.95% de los clientes.
- **KHQ**: 11.17% de los clientes.
- **KAT**: 6.99% de los clientes.

Los canales que representaban menos del 1% de los clientes fueron agrupados bajo la categoría "Otros", lo cual proporciona una visión más clara de los principales canales de captación.

La siguiente gráfica muestra la distribución de clientes por los principales canales de entrada:



1.3.6. Fecha de Contratación (entry_date)

Evaluación de Fechas de Contratación

Se realizó un análisis exhaustivo de las fechas de contratación de los clientes para identificar la consistencia de los datos en la columna entry_date. Se comenzó por contar los valores únicos de entry_date para cada cliente, con el fin de determinar si había clientes con múltiples fechas de registro. A continuación, se identificaron **7 clientes** que presentaban fechas de registro mixtas. Sin embargo, la gran mayoría de los clientes, **456,366**, tenía siempre la misma fecha de registro.

```
# Contar cuántos valores únicos de 'entry_date' tiene cada cliente
estado_por_cliente_fecha = ca_df.groupby('pk_cid')['entry_date'].nunique()

# Identificar clientes con fechas de registro mixtas (múltiples fechas de registro)
clientes_mixtos_fecha = estado_por_cliente_fecha[estado_por_cliente_fecha > 1].index

# Excluir los clientes mixtos del análisis de las otras categorías
df_no_mixtos_fecha = ca_df[~ca_df['pk_cid'].isin(clientes_mixtos_fecha)]

# Identificar clientes que tienen siempre la misma fecha de registro
clientes_misma_fecha = df_no_mixtos_fecha.groupby('pk_cid')['entry_date'].nunique()
clientes_misma_fecha = clientes_misma_fecha[clientes_misma_fecha == 1].index


# Contar el número de clientes en cada categoría (excluyendo mixtos)
n_misma_fecha = len(clientes_misma_fecha)
n_mixtos_fecha = len(clientes_mixtos_fecha)

# Mostrar los resultados
print(f"Clientes que tienen siempre la misma fecha de registro: {n_misma_fecha}")
print(f"Clientes con fechas de registro mixtas (diferentes fechas): {n_mixtos_fecha}")
```

```
Clientes que tienen siempre la misma fecha de registro: 456366
Clientes con fechas de registro mixtas (diferentes fechas): 7
```

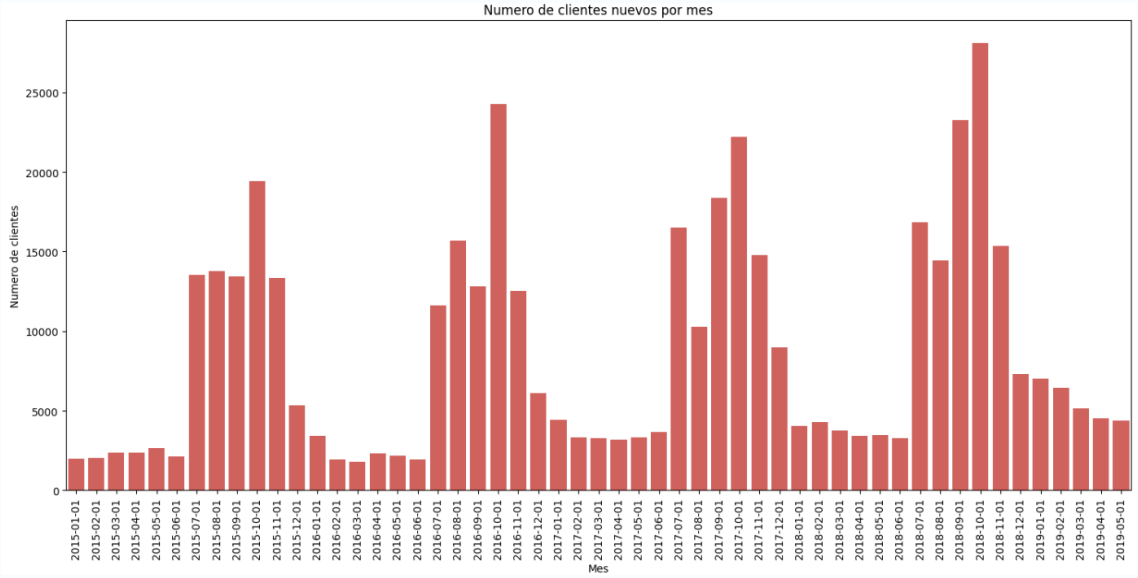
Corrección de Fechas de Registro

Se observó que las inconsistencias en entry_date podrían indicar errores en la carga de datos o actualizaciones incorrectas. Para corregir estos problemas, se decidió establecer la fecha de registro como el valor más temprano registrado para cada cliente. Este enfoque asegura que se mantenga la calidad y la integridad de los datos.

El proceso implicó ordenar los datos por pk_cid y pk_partition, y luego aplicar la fecha de la primera partición registrada a todos los registros correspondientes a cada cliente. Tras este ajuste, se verificó que **456,373** clientes ahora tenían la misma fecha de registro, eliminando así cualquier inconsistencia en los datos.

Comportamiento de Clientes Según Fecha de Contratación

Se creó un dataframe que contenía únicamente los clientes con fechas de registro consistentes. Posteriormente, se agruparon las fechas por mes para analizar la actividad de adquisición de nuevos clientes a lo largo del tiempo. El gráfico resultante mostró ciclos claros en la actividad de adquisición, con picos notables en la captación de nuevos clientes. Se identificaron períodos de alta actividad entre **mediados de julio hasta mediados de agosto** y también **a inicios de enero** de cada año.



1.3.7. Segmentación (Segment)

Análisis General de Segmentos

En el siguiente paso, se abordó la segmentación de clientes, comenzando por reemplazar los valores nulos en la columna segment con "Unknown". Se contó el número de valores únicos en segment para cada cliente, lo que reveló que **326,078** clientes tenían siempre el mismo segmento, mientras que **130,295** presentaban segmentos mixtos.

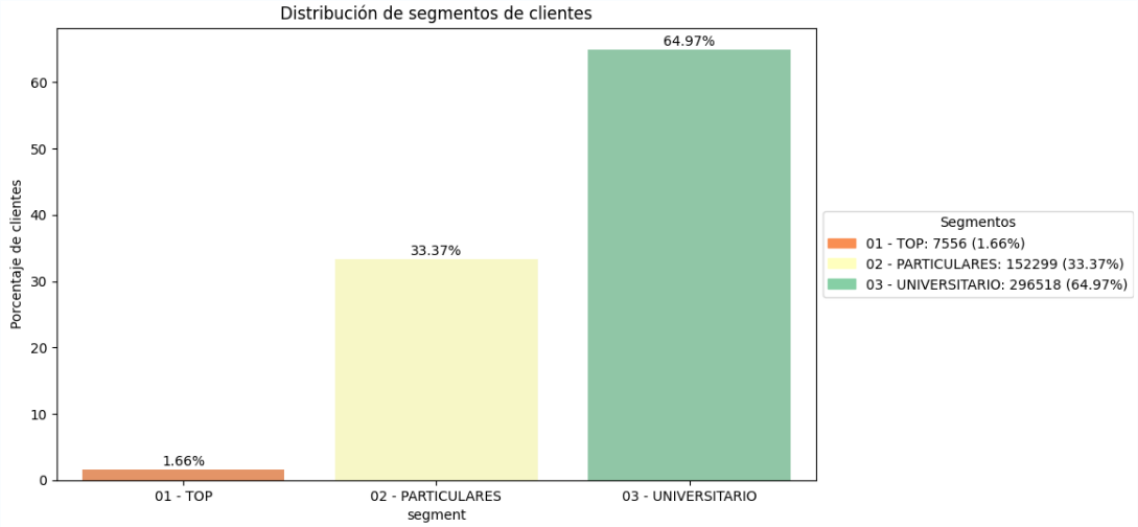
```
# Reemplazar valores nulos en la columna 'segment' por 'Unknown'
ca_df["segment"] = ca_df["segment"].cat.add_categories(["Unknown"]).fillna("Unknown")

# Contar cuántos valores únicos de 'segment' tiene cada cliente
estado_por_cliente_segment = ca_df.groupby('pk_cid')['segment'].nunique()

# Identificar clientes con segmentos mixtos (más de un segmento diferente)
clientes_mixtos_segment = estado_por_cliente_segment[estado_por_cliente_segment > 1].index

# Identificar clientes que tienen siempre el mismo segmento
clientes_mismo_segment = estado_por_cliente_segment[estado_por_cliente_segment == 1].index

# Filtrar DataFrames con clientes con segmentos mixtos y con el mismo segmento
df_mixtos_segment = ca_df[ca_df['pk_cid'].isin(clientes_mixtos_segment)]
df_mismo_segment = ca_df[ca_df['pk_cid'].isin(clientes_mismo_segment)]

# Mostrar los resultados
print(f"Clientes que tienen siempre el mismo 'segment': {len(clientes_mismo_segment)}")
print(f"Clientes con segmentos mixtos (diferentes segmentos): {len(clientes_mixtos_segment)}")
```

```
Clientes que tienen siempre el mismo 'segment': 326078
Clientes con segmentos mixtos (diferentes segmentos): 130295
```

Para manejar las inconsistencias, se definió una función que reemplazó los valores 'Unknown' por el segmento más reciente registrado. Después de aplicar este proceso, se confirmó que ya no había clientes con segmentos mixtos, y **456,373** clientes ahora tenían un único segmento asignado.

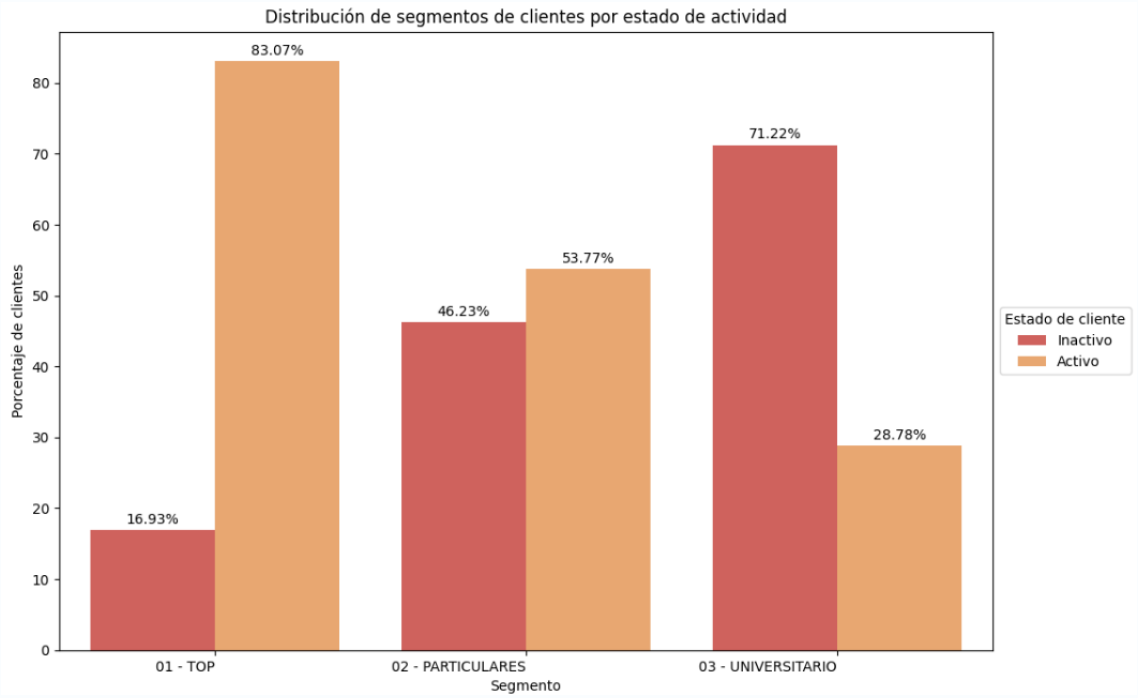
Distribución de Segmentos

La distribución final de los segmentos mostró que la mayoría de los clientes pertenecen al segmento **03 - UNIVERSITARIO** (87.24%), seguido de **02 - PARTICULARES** (12.33%) y **01 - TOP** (0.02%). Esto indica una gran concentración de clientes en el segmento universitario, lo que puede sugerir la necesidad de estrategias de marketing más específicas para los otros segmentos.



Análisis de Clientes Activos e Inactivos por Segmento

Se realizó un análisis adicional para observar la proporción de clientes activos e inactivos en cada segmento. Los resultados revelaron que los clientes en el segmento **TOP** eran los más activos, mientras que los **universitarios** mostraban una menor actividad en comparación con los demás segmentos.



1.3.8. Particiones (pk_partition)

Se realizó un análisis para determinar la presencia de clientes en distintas particiones (pk_partition). En primer lugar, se contó el número de particiones únicas asociadas a cada cliente. Los resultados mostraron que:

- **8,814** clientes están asociados a una única partición, es decir, no se repiten en más de una.
- **447,559** clientes están presentes en al menos dos particiones, lo que sugiere que son usuarios recurrentes.
- No se encontraron clientes con valores NaN en pk_partition.

Estos hallazgos resaltan que una gran mayoría de los clientes (aproximadamente el 98%) tiene actividad en múltiples particiones, lo cual es un indicador positivo de continuidad en el uso de la aplicación.

```
# Contar cuántos valores únicos de 'pk_partition' tiene cada cliente
estado_por_cliente_partition = ca_df.groupby('pk_cid')['pk_partition'].nunique()

# Identificar clientes con particiones mixtas (múltiples particiones)
clientes_mixtos_partition = estado_por_cliente_partition[estado_por_cliente_partition > 1].index

# Excluir los clientes mixtos del análisis de las otras categorías
df_no_mixtos_partition = ca_df[~ca_df['pk_cid'].isin(clientes_mixtos_partition)]

# Identificar clientes que están asociados a una única partición
clientes_una_particion = df_no_mixtos_partition.groupby('pk_cid')['pk_partition'].nunique()
clientes_una_particion = clientes_una_particion[clientes_una_particion == 1].index

# Identificar clientes con valores NaN en 'pk_partition' (excluyendo clientes mixtos)
clientes_con_nan_partition = df_no_mixtos_partition[df_no_mixtos_partition['pk_partition'].isna()]['pk_cid'].unique()

# Contar el número de clientes en cada categoría (excluyendo mixtos)
n_una_particion = len(clientes_una_particion)
n_con_nan_partition = len(clientes_con_nan_partition)
n_mixtos_partition = len(clientes_mixtos_partition)

# Mostrar los resultados
print(f"Clientes que están asociados a una única partición, que no se repiten más de una vez: {n_una_particion}")
print(f"Clientes con valores NaN en 'pk_partition': {n_con_nan_partition}")
print(f"Clientes que se repiten en al menos 2 particiones: {n_mixtos_partition}")
```

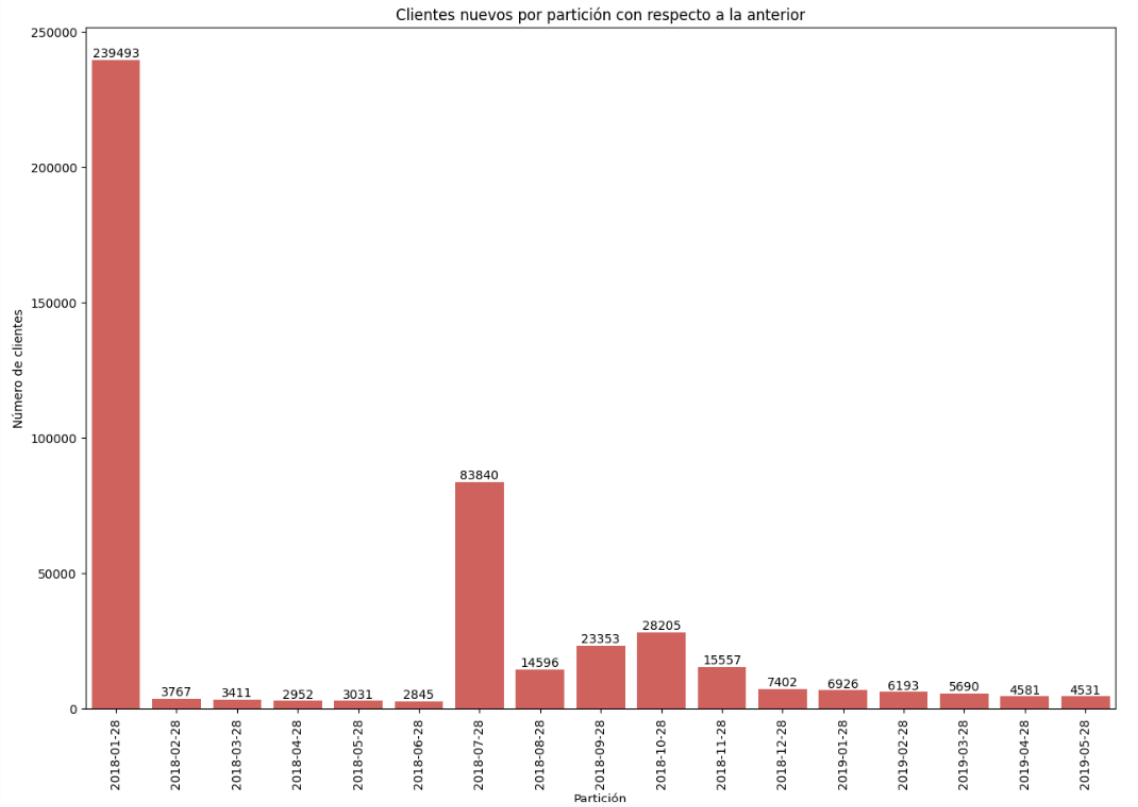
```
Clientes que están asociados a una única partición, que no se repiten más de una vez: 8814
Clientes con valores NaN en 'pk_partition': 0
Clientes que se repiten en al menos 2 particiones: 447559
```

Identificación de Clientes Nuevos en Cada Partición

A continuación, se procedió a analizar la adquisición de nuevos clientes en cada partición en comparación con la anterior. Para ello, se ordenaron los datos por pk_partition y pk_cid. Se creó un conjunto que almacenaba todos los clientes vistos hasta el momento y, al iterar sobre las particiones, se identificaron los nuevos clientes al restar los clientes actuales de los acumulados.

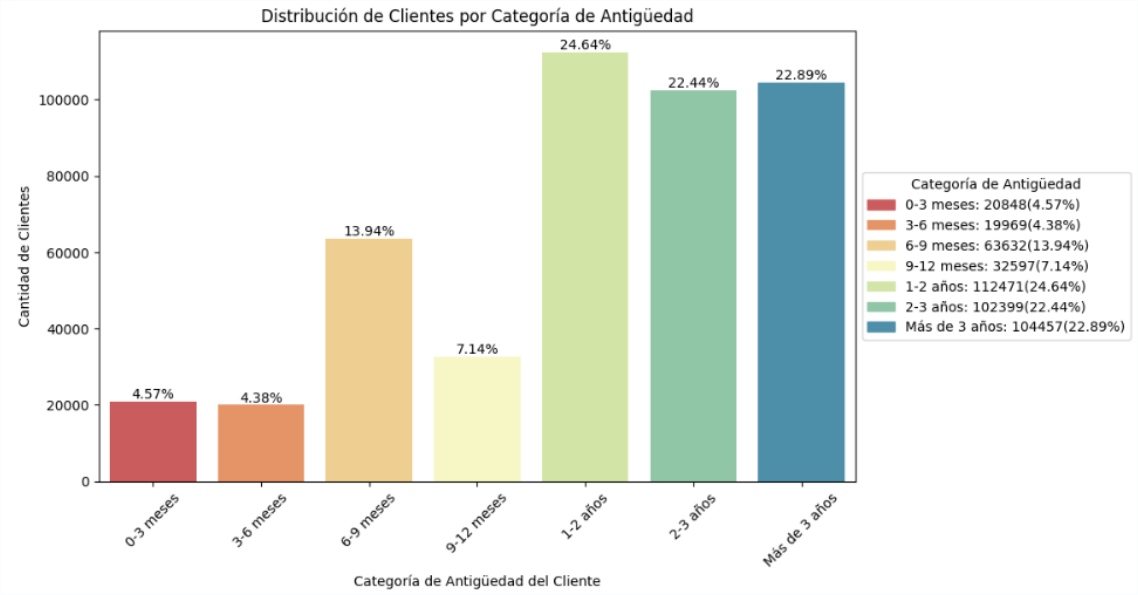
La suma total de nuevos clientes por partición coincide con el número total de clientes únicos en el dataset, **456,373**.

El gráfico resultante muestra picos de adquisición en varias particiones, lo que podría estar vinculado a campañas o eventos específicos. Hacia el final del período analizado, se observa una tendencia a la baja en la adquisición de nuevos clientes, sugiriendo la necesidad de nuevas estrategias para atraer a clientes o una posible saturación del mercado.



Distribución de Clientes según Antigüedad

Para complementar el análisis, se evaluó la antigüedad de los clientes en la plataforma. Se calculó la antigüedad en días a partir de la última fecha de partición (pk_partition) y la fecha de registro (entry_date), y se categorizó en intervalos de tiempo. Las categorías definidas fueron:



Los resultados mostraron que más del **70%** de los clientes tienen más de un año de antigüedad, lo cual es un indicativo positivo en términos de retención y lealtad. Sin embargo, se identificó una caída notable en la categoría de 9-12 meses en comparación con la de 6-9 meses, lo que sugiere que se debe prestar atención a este grupo para mejorar la retención. Además, la mayor

concentración de clientes se encontró en el rango de 1 a 3 años, lo que representa una oportunidad para implementar estrategias de fidelización dirigidas a este segmento.

1.3.9. Resumen

1. **Comportamiento de Clientes**: Se clasificaron a los clientes según su actividad:
    - **Siempre Activos**: 139,194 clientes.
    - **Nunca Activos**: 258,642 clientes.
    - **Actividad Variable**: 58,537 clientes. Este análisis revela que el 12% de los clientes cambian su estado de actividad, indicando una necesidad de estrategias para mejorar la retención.
2. **Análisis de entry_channel**: Se identificaron 124,504 clientes con canales de entrada mixtos (27% del total). Se reemplazaron los valores Unknown con el último canal conocido. Los principales canales son KHE (52.22%) y KFC (14.94%).
3. **Análisis de entry_date**: Se ajustaron las fechas de registro a la más temprana para los clientes con registros mixtos. Se observó un patrón de adquisición de clientes con picos que podrían estar relacionados con campañas específicas y una estabilización hacia el final del período.
4. **Segmentación de Clientes**: La mayoría de los clientes pertenecen al segmento 03 - UNIVERSITARIO, seguido de 02 - PARTICULARES y 01 - TOP. Se limpiaron los segmentos mixtos y se asignó el segmento más común a los clientes con información faltante.
5. **Clientes por pk_partition**: Se identificaron 8,814 clientes que solo están presentes en una partición. En total, 447.559 clientes se repiten en más de una partición. El análisis de nuevos clientes en cada partición reveló picos en la adquisición, lo que sugiere que podrían ser necesarios nuevos enfoques para atraer clientes.
6. **Distribución de Clientes según Antigüedad**: Más del 70% de los clientes tienen más de un año de antigüedad. Se identificó una caída en la retención entre los 9-12 meses, lo que sugiere áreas de mejora en la fidelización.

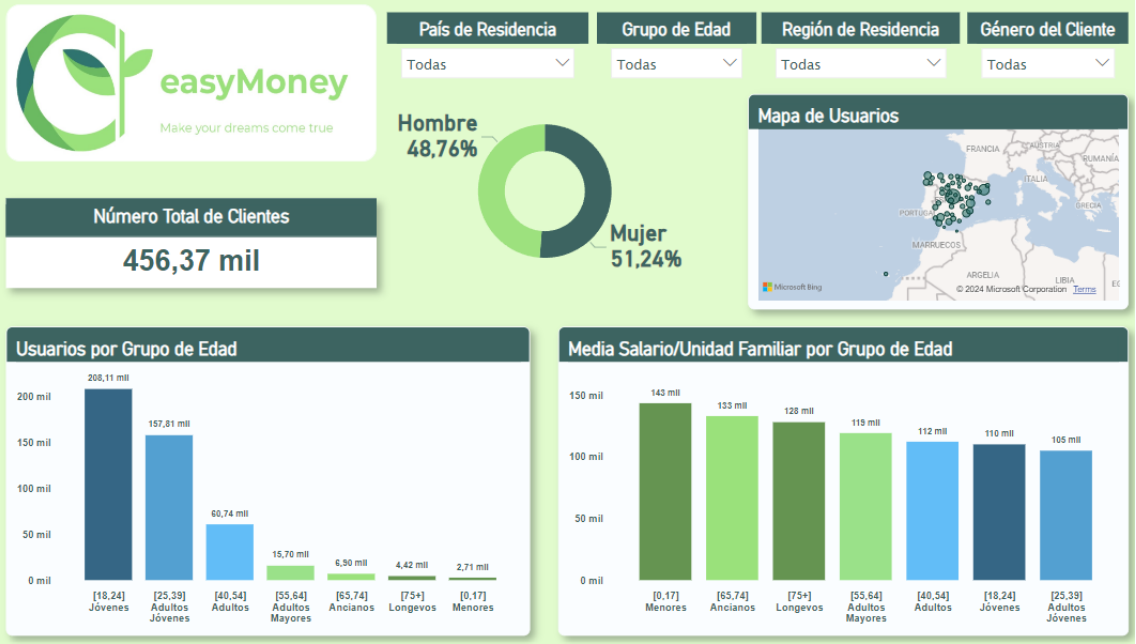
1.4. Repositorio Business Intelligence

El presente Dashboard ha sido desarrollado con el objetivo de apoyar la nueva estrategia comercial de la compañía, enfocada en aumentar la penetración de nuestra cartera actual de productos dentro del mercado. En línea con la matriz de Ansoff, el enfoque ha cambiado hacia la maximización de la rentabilidad de nuestra base de clientes existente, reduciendo de forma considerable la actividad intensiva de captación.

El Dashboard proporcionará una visión clara y detallada del rendimiento de ventas de nuestros productos, permitiendo identificar con precisión cuántos productos se han vendido en un periodo determinado y si estos han sido contratados por clientes nuevos o existentes.

Además, este autoservicio de BI no solo permitirá obtener datos en tiempo real, sino también segmentar y analizar las diferentes dinámicas de contratación, brindando al equipo de dirección una herramienta poderosa para optimizar la gestión comercial y fortalecer el crecimiento a largo plazo.

- Ejemplo del Dashboard



El Dashboard de mostrado presenta una distribución general de los clientes, con un total de **456,37 mil usuarios**. Destacan las siguientes características:

1. **Género**: Hay una ligera mayoría de mujeres (**51,24%**) frente a hombres (**48,76%**).
2. **Usuarios por Grupo de Edad**: Los **jóvenes (18-24 años)** son el grupo más numeroso (**200,11 mil**), seguidos por los **adultos jóvenes (25-39 años)** con **157,81 mil**.
3. **Salario Medio por Grupo de Edad**: Los **ancianos (65-74)** y **longevos (75+)** tienen los ingresos más altos, mientras que los grupos jóvenes presentan ingresos más bajos.
4. **Mapa de Usuarios**: Muestra la distribución geográfica, mayoritariamente concentrada en Europa.

Tarea 2 - Segmentación de Clientes

Este apartado tiene como objetivo segmentar la base de clientes de *Easy Money* en siete u ocho grupos diferenciados para personalizar sus estrategias comerciales y de marketing. A partir de la segmentación, se busca proporcionar insights que ayuden a la compañía a optimizar la relación con sus clientes y aumentar la efectividad de sus campañas publicitarias. Para ello, se han empleado técnicas avanzadas de análisis de datos, reducción de dimensionalidad y clustering.

2.1. Análisis de Variables

Para lograr una segmentación efectiva, se partió de la identificación y selección de las variables que proporcionaran la mayor relevancia en términos de comportamiento de los clientes. Entre las variables seleccionadas se encuentran:

- **Variables demográficas:** Edad, género, nivel educativo, y lugar de residencia. Estas variables son esenciales para diferenciar clientes por etapas de vida, necesidades y preferencias.
- **Comportamiento de compra:** Frecuencia y monto de transacciones, tipo de productos contratados (préstamos, inversión, servicios de pago, etc.). Estas características proporcionan información clave sobre los hábitos de consumo financiero de los clientes.
- **Interacción con productos fintech:** Frecuencia en el uso de productos digitales de la plataforma, como préstamos, créditos rápidos o soluciones de inversión.

La selección de estas variables es crucial ya que permiten una segmentación basada en características tanto estáticas (edad, género) como dinámicas (comportamiento de compra, uso de productos). Este enfoque híbrido garantiza que los segmentos resultantes sean significativos para la estrategia comercial.

Analizando más en detalle la aportación que puede ofrecer cada una de las variables

- **age (Edad):** Es fundamental para entender la etapa de vida en la que se encuentra el cliente, lo que afecta sus necesidades financieras.
- **country_id (País de residencia):** Podría ser útil para identificar variaciones geográficas en el comportamiento de los clientes.
- **region_code (Provincia):** Para segmentar a nivel más granular dentro de un país, especialmente si hay diferencias regionales significativas.
- **gender (Sexo):** Aunque menos relevante en algunos casos, podría influir en la preferencia por ciertos productos financieros.
- **salary (Ingresos brutos):** Indica la capacidad de compra e inversión del cliente, crucial para segmentar en términos de valor potencial.
- **active_customer (Actividad del cliente):** Indicador de si el cliente está activamente utilizando los productos, importante para identificar clientes leales o en riesgo de abandono.
- **entry_channel (Canal de captación):** Puede influir en el comportamiento inicial y la adopción de productos del cliente.
- **credit_card, debit_card (Tarjetas de crédito/débito**): Reflejan el uso de productos financieros cotidianos y la disposición del cliente a utilizar crédito.
- **em_account_p, em_account_pp, em_acount, emc_account (Cuentas de easyMoney):** Estas variables muestran el nivel de compromiso del cliente con los diferentes productos de la empresa, siendo clave para entender la amplitud de su relación con la empresa.

- **loans, mortgage (Préstamos e Hipotecas):** Indican el nivel de deuda y el compromiso financiero a largo plazo del cliente.
- **funds, securities (Fondos y Valores):** Reflejan la predisposición del cliente a invertir y su perfil de riesgo.
- **short_term_deposit, long_term_deposit (Depósitos a corto y largo plazo**): Muestran las preferencias del cliente por la seguridad y el rendimiento financiero a corto o largo plazo.
- **payroll, payroll_account (Domiciliaciones y cuentas bonificadas):** Indican el nivel de vinculación del cliente con la empresa a través de la domiciliación de su salario y otros ingresos recurrentes.
- **pension_plan (Plan de pensiones):** Refleja la planificación a largo plazo del cliente y su preocupación por la jubilación.
- **categoria_antiguedad:** Muestra la retención del cliente a largo plazo.

2.2. Preprocesamiento

El conjunto de datos utilizado para la segmentación proviene de diversas fuentes, incluyendo información transaccional, características demográficas y el uso de productos financieros. Para asegurar la calidad de los datos, se llevó a cabo un proceso de preprocesamiento exhaustivo.

- **Codificación de variables categóricas:** Todas las variables categóricas, como el canal de entrada y el grupo de edad, se transformaron a formato numérico mediante Frecuency Encoding. Este paso permitió que las categorías fueran correctamente interpretadas por los algoritmos de machine learning sin introducir relaciones artificiales entre las diferentes categorías.
- **Análisis de correlación:** Una vez que las variables categóricas fueron transformadas a formato numérico, se realizó un análisis de correlación para identificar variables altamente correlacionadas. Aunque algunas variables mostraron una correlación superior a 0.9, se decidió mantenerlas en el modelo debido a la posibilidad de que proporcionen insights distintos sobre los clientes.
- **Estandarización de variables numéricas:** Se aplicó el método RobustScaler para estandarizar las variables numéricas, lo cual fue fundamental para asegurar que todas las variables tuvieran el mismo peso durante el proceso de clustering. Este enfoque es especialmente útil en presencia de outliers, ya que el escalado se basa en la mediana y los cuartiles, protegiendo los resultados de distorsiones.

2.2.1. Agrupación por Productos Usados

El primer paso fue **agrupar a los clientes según los productos que tienen contratados** a lo largo de diferentes particiones temporales, calculando la media de cada producto por cliente. Esto permite evaluar el grado de uso de cada producto por parte de los clientes de forma precisa.

```
productos = df_full_clean.columns[2:17]
productos
df_productos_usos = df_full_clean.groupby("pk_cid")[productos].mean()
```

Este paso permite obtener un perfil detallado del uso de productos financieros por cliente, permitiendo que posteriormente los segmentos puedan diferenciarse en función de la cantidad y tipo de productos contratados. La media de cada producto por cliente refleja la intensidad de uso en cada categoría de productos.

2.2.2. Número Máximo de Productos Contratados por Cliente

Luego, se calculó el **número máximo de productos contratados por cada cliente hasta la fecha**. Esto proporciona una métrica clara del compromiso del cliente con la plataforma y el uso de sus servicios.

```
df_cant_prod_contratados = df_full_clean.groupby("pk_cid")["num_products_contracts"].max()
```

Este cálculo permite identificar a los clientes con un mayor nivel de vinculación con la empresa, ya que aquellos que han contratado más productos pueden considerarse más fieles o comprometidos. Esto es fundamental para detectar segmentos de alto valor o potencial.

2.2.3. Tratamiento de la Variable Region Code

Para la columna **region_code**, se barajaron distintas opciones, incluyendo **One-Hot Encoding** y **Label Encoding**, pero finalmente se decidió utilizar **Frequency Encoding**. Este enfoque se eligió para evitar la alta dimensionalidad y relaciones ordinales incorrectas, asegurando un mejor rendimiento del modelo de K-Means, ya que permite representar la distribución de clientes por región de manera efectiva, sin inflar el número de columnas como sucedería con One-Hot Encoding. Esto reduce la dimensionalidad del dataset y asegura que el modelo capture la densidad relativa de clientes en cada provincia.

Pasos seguidos:

1. Calcular la frecuencia de cada provincia en el dataset.
2. **Mapear las frecuencias** calculadas a cada cliente en función de su provincia de residencia.

```
# Paso 1: Calcular la frecuencia de cada provincia
frecuencias_provincia = df_full_clean['region_code'].value_counts(normalize=True)  # normalize=True para obtener las frecuencias relativas

# Paso 2: Mapear las frecuencias a la columna de provincia
df_full_clean['region_code_encoded'] = df_full_clean['region_code'].map(frecuencias_provincia)

df_region_code_frecuency = df_full_clean.groupby("pk_cid")["region_code_encoded"].last()

# Verificar los resultados
print(df_full_clean[['region_code', 'region_code_encoded']].head())
```

2.2.4. Transformación de la Fecha de Ingreso

Se transformó la columna **entry_date** a un formato numérico, ya que, si bien la fecha original no aporta información directa, al convertirla en su representación ordinal (número de días desde una fecha base), se puede utilizar en los modelos de machine learning. La transformación de fechas a un formato numérico permite incluir el factor tiempo en el análisis. En este caso, la fecha de ingreso de un cliente puede influir en su nivel de compromiso con la empresa o en el uso de productos financieros, por lo que es importante incluirla de una forma que sea compatible con los modelos de machine learning.

```
# eliminar la columna entry_date que tampoco aporta información
df_entry_date = df_full_clean.groupby("pk_cid")["entry_date"].first()

# llevar df_entry_date a formato numerico

df_entry_date = pd.to_datetime(df_entry_date)
df_entry_date = df_entry_date.apply(lambda x: x.toordinal())
df_entry_date
```

2.2.5. Eliminación de Columnas Irrelevantes

Algunas columnas fueron eliminadas por no aportar valor suficiente para el análisis de segmentación. Entre ellas, las columnas **country_id**, **deceased** y **em_account_pp**, las cuales no brindaban información relevante o habían sido tratadas previamente en la tarea 1. En este caso, estas variables no aportaban información adicional para la segmentación y, por tanto, su eliminación simplifica el análisis sin perder valor.

2.2.6. Cálculo del Grado Medio de Actividad

Para la variable **active_customer**, se agrupó por clientes y se calculó la media, lo que indica el grado promedio de actividad de los clientes en la plataforma. Este indicador refleja cuán activos son los clientes en el uso de los productos y servicios de *Easy Money*, proporcionando una métrica importante para diferenciar entre clientes que son usuarios frecuentes y aquellos que pueden estar en riesgo de abandono.

```
df_active_customer = df_full_clean.groupby("pk_cid")["active_customer"].mean()
```

2.2.7. Limpieza de Variables Finales

Para el resto de las variables que ya venían preprocesadas en tareas anteriores, se seleccionaron los valores más recientes (de la última partición) para mantener una única fila por cliente. Las variables seleccionadas fueron **gender, salary, mes_partition, grupo_edad, age, entry_channel, segment, categoria_antiguedad**.

Este último paso asegura que se mantenga la información más actualizada de cada cliente, necesaria para la segmentación. Mantener una sola fila por cliente facilita la implementación del modelo de clustering y asegura que se estén utilizando datos consistentes y recientes.

2.3. Matriz de Correlación y Variables Altamente Correlacionadas

Una vez que todas las variables fueron convertidas a un formato numérico, se calculó la **matriz de correlación**, la cual muestra las relaciones lineales entre las variables. Para visualizar la matriz, se generó un **heatmap** utilizando la biblioteca Seaborn, lo cual permite identificar fácilmente las variables que están altamente correlacionadas (aquellas con un coeficiente de correlación mayor a 0.9).

```
# Calcular la matriz de correlación
matriz_correlacion = df_numerico_segmetacion_1.corr()

# Visualizar la matriz de correlación utilizando un heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(matriz_correlacion, cmap='coolwarm', center=0)
plt.title('Heatmap de la Matriz de Correlación')
plt.show()
```

Se fijó un **umbral de 0.9** para considerar variables como altamente correlacionadas. En este caso, las parejas de variables que superaban este umbral fueron:

- 'payroll' y 'pension_plan'
- 'segment_02 - PARTICULARES' y 'segment_03 - UNIVERSITARIO'

A pesar de estar correlacionadas, ambas parejas de variables fueron conservadas en el análisis, ya que representan aspectos distintos del comportamiento de los clientes que podrían ofrecer insights valiosos.

```
# Identificar variables altamente correlacionadas
umbral = 0.9
correlaciones_altas = np.where(np.abs(matriz_correlacion) > umbral)
pares_correlacionados = [(matriz_correlacion.index[x], matriz_correlacion.columns[y]) for x, y in zip(*correlaciones_altas) if x != y and x < y]

print("Pares de variables altamente correlacionadas:")
for par in pares_correlacionados:
    print(par)
```

2.4. Estandarización de las Variables Numéricas

Una vez realizadas las transformaciones y el análisis de correlación, se procedió a la **estandarización de las variables numéricas**. Este proceso es necesario para que todas las variables tengan el mismo rango y no influyan desproporcionadamente en el modelo de clustering. Para ello, se utilizó **RobustScaler** de Scikit-learn. **RobustScaler** es particularmente útil cuando hay outliers en los datos, ya que utiliza la mediana y los cuartiles para escalar las variables, lo que lo hace más robusto ante la existencia de valores extremos.

La estandarización es un paso crucial para algoritmos como **K-Means**, que dependen de las distancias entre los puntos. Sin estandarización, las variables con valores absolutos más grandes (como salarios o montos financieros) podrían dominar el análisis y hacer que el clustering se sesgue hacia ellas.

```
# Initialize the scaler
scaler = RobustScaler()

# Fit and transform the data
df_estandarizado = scaler.fit_transform(df_numerico_segmetacion_1)

# Convert the result to a DataFrame
df_estandarizado = pd.DataFrame(df_estandarizado , columns=df_numerico_segmetacion_1.columns)

# Verify the results
df_estandarizado
```

2.5. Ejecución del Clustering

En esta sección, se evaluaron tres enfoques distintos para el clustering de los datos de clientes de *Easy Money*: clustering con el dataset completo, clustering con reducción de dimensionalidad mediante PCA y clustering con ingeniería de características. Cada uno de estos enfoques fue diseñado para evaluar la calidad de la segmentación obtenida en términos de inercia y la puntuación de silueta, con el objetivo de determinar cuál enfoque proporciona clusters más claros y útiles para el análisis comercial.

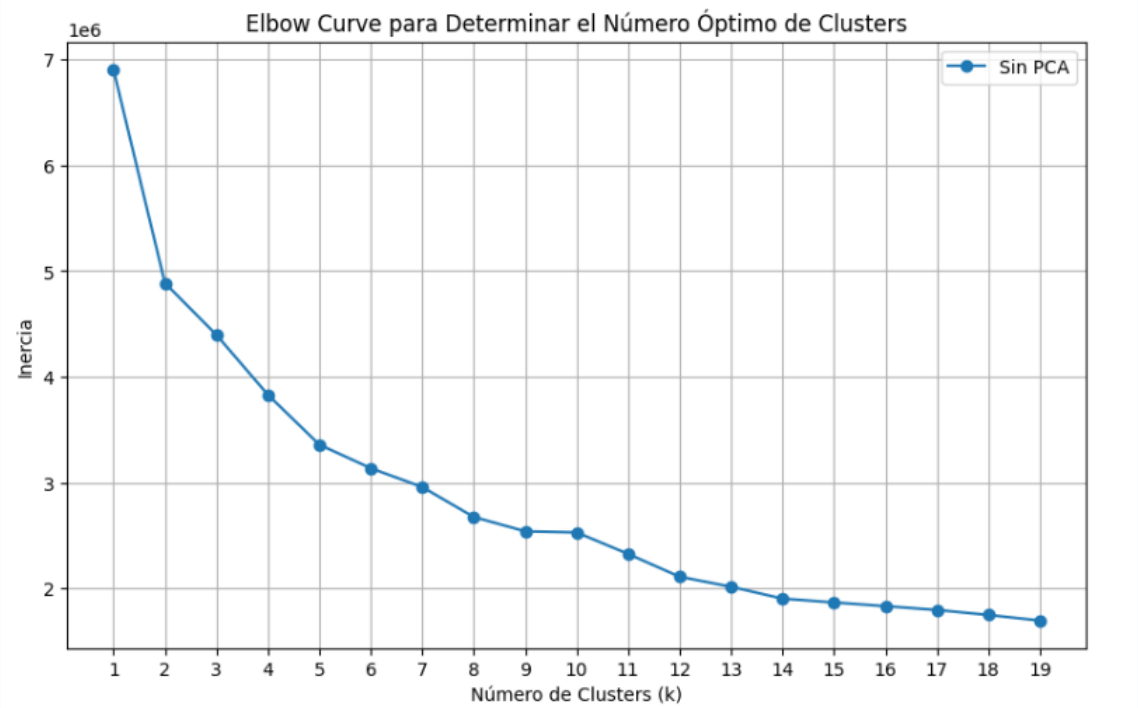
2.5.1. Clustering con el Dataset Completo

Este enfoque aplicó el algoritmo de **K-Means** directamente sobre el dataset estandarizado completo (sin reducción de dimensionalidad ni transformación adicional de las variables).

- **Definición de un rango de clusters:** Se probaron diferentes valores de **k** (número de clusters), variando entre 1 y 20, para encontrar el número óptimo de clusters. Para cada iteración, se calculó la **inercia** óptima para segmentar los datos.

```
# Definir un rango de k (número de clusters) para probar
rango_k = range(1, 20)
inercias = []

for k in rango_k:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(df_estandarizado)
    inercias.append(kmeans.inertia_)
```

- **Curva del Codo:** La gráfica de la **curva del codo** mostró que la inercia disminuía rápidamente al incrementar el número de clusters hasta **k = 5**. A partir de ese punto, la inercia se estabilizó, lo que indica que 5 clusters podrían ser suficientes para capturar la estructura de los datos.



Basado en el análisis, se seleccionó **5 clusters** como el número óptimo en este enfoque. Este resultado sugiere que el dataset completo estandarizado puede ser segmentado de manera eficiente en 5 grupos sin necesidad de reducción de dimensionalidad ni ingeniería de características adicionales.

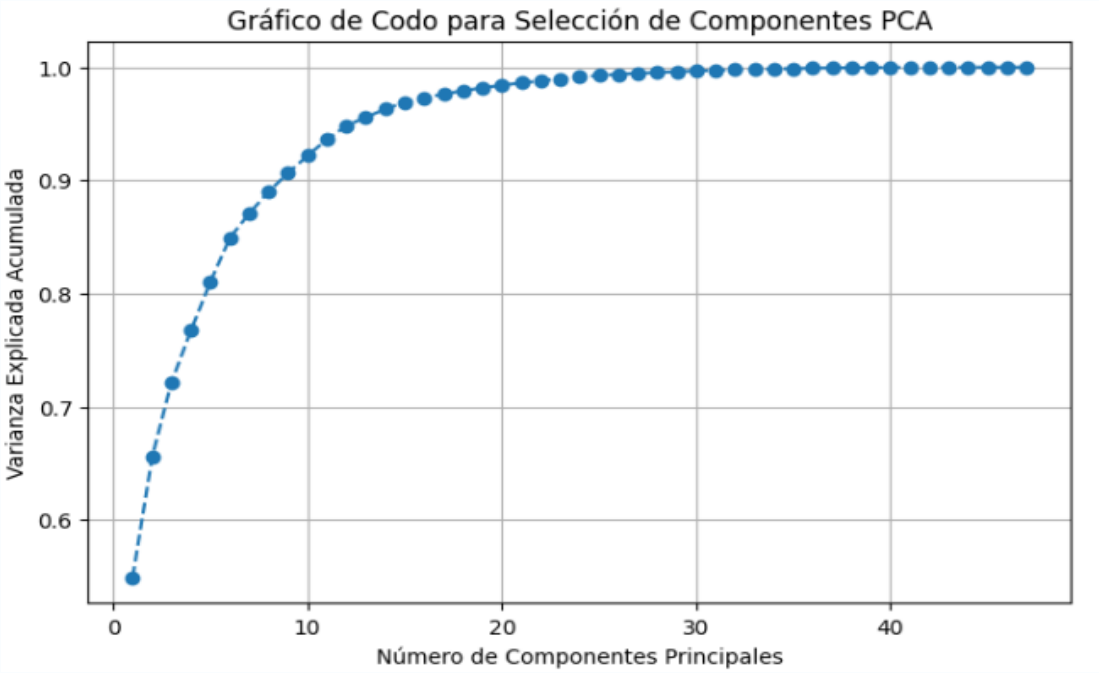
2.5.2. Clustering con Reducción de Dimensionalidad mediante PCA

En este enfoque, se utilizó **Análisis de Componentes Principales (PCA)** para reducir la dimensionalidad del dataset antes de aplicar el clustering. Este método busca simplificar los datos al identificar los componentes principales que capturan la mayor parte de la varianza.

```
# Aplicar PCA
pca = PCA()
pca.fit(df_estandarizado)

# Varianza explicada por cada componente
varianza_explicada = pca.explained_variance_ratio_
varianza_explicada_acumulada = np.cumsum(varianza_explicada)
```

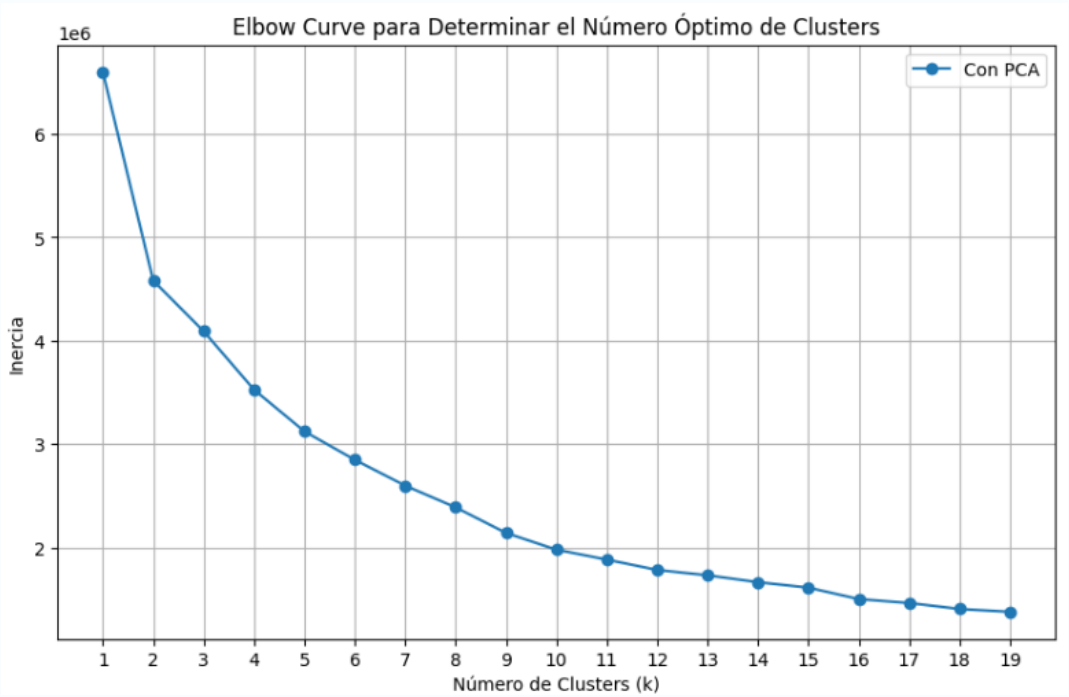
```
# Selección de número de componentes que expliquen al menos el 95% de la varianza
n_componentes = np.argmax(varianza_explicada_acumulada >= 0.95) + 1
print(f"Número de componentes: {n_componentes}")

# Aplicar PCA con el número óptimo de componentes
pca = PCA(n_components=n_componentes)
caracteristicas_reducidas = pca.fit_transform(df_estandarizado)

# Crear un DataFrame con los componentes principales
df_pca = pd.DataFrame(caracteristicas_reducidas, columns=[f'PC{i+1}' for i in range(n_componentes)])
```

- **Selección del número de componentes:** Se seleccionaron suficientes componentes principales para retener el **95% de la varianza** de los datos originales, lo que redujo el número de variables de 47 a **13 componentes** principales.




```
Número de componentes: 13
```

- **Curva del Codo para PCA:** Al igual que en el enfoque anterior, se aplicó el algoritmo de **K-Means** sobre el dataset reducido a 13 componentes principales, probando un rango de valores para **k** entre 1 y 20. El gráfico del codo mostró una reducción en la inercia similar a la observada sin PCA, pero con clusters más compactos, lo que sugiere que los grupos resultantes son más coherentes.



Aunque los resultados son similares al clustering sin PCA, la reducción de dimensionalidad permitió que los clusters fueran más compactos y eficientes. El análisis sugiere que **4 o 5 clusters** es el número óptimo, con una notable reducción en la inercia comparado con el dataset completo.

2.5.3. Clustering con Ingeniería de Características

En este enfoque, se realizó **ingeniería de características** para crear nuevas variables a partir de las originales, con el objetivo de mejorar la segmentación de clientes. El proceso consistió en agrupar productos financieros, categorizar canales de entrada, clasificar los salarios y eliminar variables redundantes, entre otros pasos. Tras realizar estos ajustes, se procedió al análisis de correlación, estandarización y aplicación del clustering.

Agrupación de Productos Financieros

Se crearon nuevas columnas que agrupan los productos en tres categorías clave: **cuentas**, **ahorro/inversión** y **financiación**. Estas agrupaciones permitieron simplificar la representación de los productos financieros sin perder información relevante.

- **Cuentas:** Se agruparon productos como cuentas de depósito a corto y largo plazo, cuentas de nómina y cuentas de ahorro.
- **Ahorro/inversión:** Incluye productos de fondos, valores y planes de pensiones.
- **Financiación:** Se agrupan productos como préstamos, tarjetas de crédito, hipotecas y tarjetas de débito.

La agrupación de productos simplifica el modelo y permite capturar de manera más general el comportamiento financiero de los clientes sin perder detalle relevante.

Categorías de Canales de Entrada

Se decidió agrupar los canales de entrada en **principales** y **secundarios**, lo que permite reducir la dimensionalidad y hacer más manejable el análisis.

Los canales principales seleccionados fueron: 'KHE', 'KHQ', 'KFC'. Todos los demás canales se clasificaron como secundarios. Agrupar los canales de entrada ayuda a reducir el número de categorías mientras se preserva la información sobre el origen de los clientes, facilitando la identificación de patrones de comportamiento en función del canal de ingreso.

Eliminación de Variables Redundantes

Se eliminaron varias variables que podrían introducir redundancias o sesgos en el modelo. Las variables eliminadas incluyen: **segment, grupo_edad, gender, region_code_encoded**

Eliminar variables redundantes es una práctica común para evitar la multicolinealidad y mejorar la eficiencia del modelo de clustering. Estas variables no aportaban nueva información significativa tras la creación de nuevas características.

Codificación de la Antigüedad

La variable **categoria_antiguedad** se codificó en función de la frecuencia de cada categoría en el dataset. Codificar las categorías de antigüedad mediante **frequency encoding** permite capturar la variabilidad en la distribución de los clientes sin inflar el número de columnas, lo que mantiene el modelo más compacto.

Clasificación de Salarios

La columna de **salary** se transformó en una variable categórica, dividiendo los salarios en tres rangos: **Ingreso Bajo**, **Ingreso Medio** e **Ingreso Alto**, según los percentiles 33% y 66%.

Clasificar los salarios en categorías simplifica el análisis y permite evaluar cómo los ingresos influyen en la segmentación de clientes, además de evitar que el salario tenga un impacto desproporcionado en el clustering.

Conversión de Variables Categóricas a Numéricas

Las variables categóricas restantes se convirtieron a formato numérico mediante **One-Hot Encoding** para que pudieran ser procesadas por el algoritmo de clustering.

El **One-Hot Encoding** asegura que las variables categóricas sean tratadas correctamente por el algoritmo de machine learning, sin introducir relaciones artificiales entre categorías.

Análisis de Correlación y Estandarización

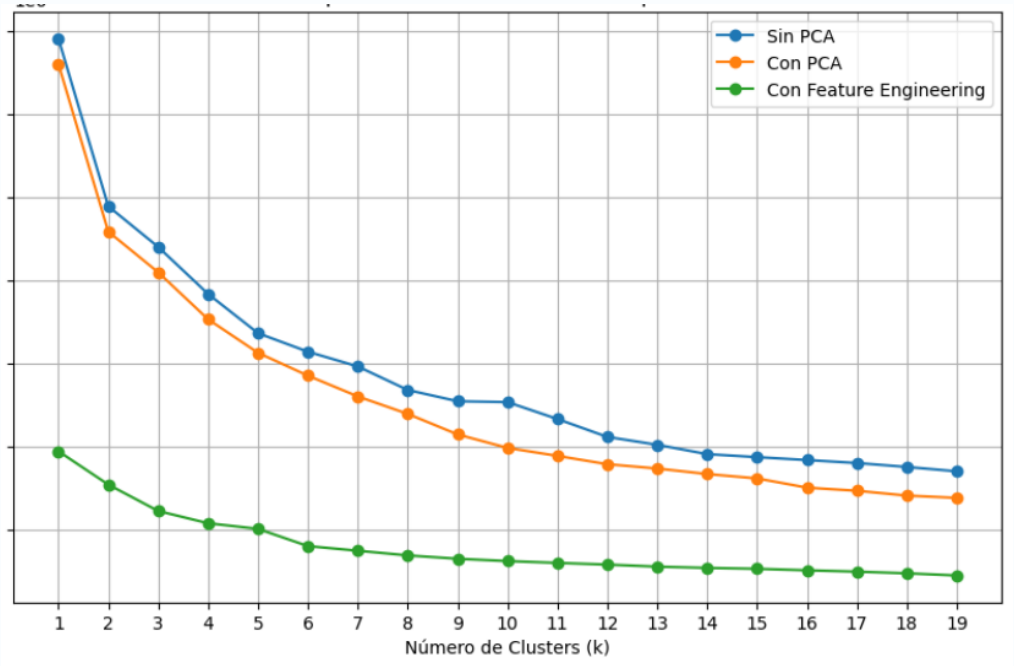
Se calculó la **matriz de correlación** para identificar variables altamente correlacionadas, que podrían ser eliminadas si presentaban redundancia. En este caso, no se identificaron variables altamente correlacionadas (con un umbral mayor a 0.9).

En esta sección se aplicó el algoritmo **K-Means** utilizando los tres enfoques previamente definidos: el **dataset original**, el dataset con **reducción de dimensionalidad mediante PCA**, y el dataset con **ingeniería de características**. El objetivo fue identificar el número óptimo de clusters que proporcionara una segmentación clara y manejable.

2.5.4. Definición del Rango de Clusters

Para cada uno de los tres enfoques, se probó un rango de **k** entre 1 y 20, calculando la **inercia** para cada valor de clusters. A continuación, se presenta un análisis comparativo basado en las inercias obtenidas y representadas en la **curva del codo**.

La **curva del codo** mostró el punto en el que agregar más clusters deja de reducir significativamente la inercia, indicando que los clusters adicionales no aportan información relevante. A continuación, se observa la visualización de la curva del codo para los tres enfoques:



- **Clustering sin PCA:** Proporciona una segmentación razonable con 5 clusters, pero los resultados son menos compactos en comparación con los otros enfoques.
- **Clustering con PCA:** Mejora la compactación de los clusters sin perder demasiada información, mostrando que 4 o 5 clusters son suficientes para capturar la estructura de los datos.
- **Clustering con Feature Engineering:** Este enfoque muestra la mayor reducción de inercia, lo que indica que las nuevas variables son más efectivas para segmentar a los clientes, con 5 a 7 clusters recomendados.

En conclusión, la **ingeniería de características** resultó ser el enfoque más efectivo para agrupar a los clientes, seguido por el uso de **PCA**. Ambos enfoques permitieron una segmentación más compacta y coherente, facilitando la interpretación y la aplicación de los resultados en estrategias comerciales.

2.6. Interpretación del Clustering

Una vez completado el proceso de clustering, es necesario realizar una interpretación y un análisis de los resultados obtenidos para seleccionar el número adecuado de clusters y comprender las características principales de cada grupo. El análisis incluyó la comparación de resultados con **4, 5 y 6 clusters**, y se tomó la decisión final de trabajar con **6 clusters**, que ofrecían el equilibrio adecuado entre detalle y manejabilidad.

2.6.1. Selección del Número de Clusters

El análisis de la **curva del codo** indicó que los valores óptimos para el número de clusters estaban entre 5 y 6, pero la elección final de **6 clusters** se basó en la observación de la distribución y el tamaño de los grupos.

- **Aplicación de K-Means con diferentes clusters:** Se probaron **4, 5 y 6 clusters** y se analizó la distribución de los clientes en cada uno.

- **Distribución de clientes:** El número de clientes por cluster fue evaluado para ver si cada cluster era lo suficientemente grande y balanceado para generar insights significativos. Al trabajar con 6 clusters, se observó una distribución equilibrada que facilitaba tanto la interpretación como la implementación de estrategias comerciales.

```
cluster_4
3    242079
1    107071
0     70878
2     36216
Name: count, dtype: int64
cluster_5
0    241365
1    101683
4     70079
2     36084
3      7033
Name: count, dtype: int64
```

```
cluster_6
0    121000
3    120365
2    101683
5     70079
4     36084
1      7033
Name: count, dtype: int64
```

- **Decisión de trabajar con 6 clusters:** Aunque la Directora General (Carol) había propuesto dividir los clientes en 7 u 8 grupos, se optó por trabajar con **6 clusters** porque la adición de más clusters no mejoraba significativamente la reducción de la inercia. Además, la distribución equilibrada en 6 grupos proporcionaba una base sólida para la segmentación, con clusters manejables y diferenciados.

Se seleccionó el número de **6 clusters** para obtener un balance adecuado entre la utilidad de la segmentación y la posible implementación. Este enfoque permite que los grupos sean lo suficientemente grandes para generar insights valiosos sin perder especificidad.

2.6.2. Resumen de Clusters y Características Medias

Se calcularon las **medias de las características** por cluster, lo que permitió identificar los perfiles de cada grupo, así como la cantidad de clientes en cada cluster.

- **Cálculo de las medias por cluster:** Se generó un resumen de las principales características de los clientes agrupados en cada cluster, incluyendo la cantidad de productos contratados, la actividad de los clientes y otros factores relevantes.

```
# Lista de columnas excluyendo las de los clusters específicos
cols = list(df_numerico_estandarizado_3.columns)
cols.remove('cluster_6')
cols.remove("cluster_5")
cols.remove("cluster_4")

# Calcular las medias para cada cluster
pt = pd.pivot_table(df_numerico_estandarizado_3, index='cluster_6', values=cols, aggfunc='mean')

# Calcular la cantidad de elementos en cada cluster
add = pd.pivot_table(df_numerico_estandarizado_3, index='cluster_6', values='num_products_contracts', aggfunc='count')
pt['count'] = add

# Añadir 'count' a la lista de columnas para la visualización final
cols.append('count')

# Mostrar la tabla con estilos y gradientes de color
pt[cols].style.background_gradient(cmap='coolwarm')
```

- **Desnormalización de las variables:** Para una mejor interpretación de los resultados, algunas variables, como **entry_date**, fueron desnormalizadas y transformadas a su formato original.

```
df_desnorm = df_seg_3.copy()
df_desnorm["cluster_6"] = df_numerico_estandarizado_3["cluster_6"]
# llevando la columna entry_date a su formato original
df_desnorm["entry_date"] = df_desnorm["entry_date"].apply(lambda x: datetime.fromordinal(x))
df_desnorm.groupby("cluster_6").mean()
```

2.6.3. Interpretación de los Clusters

A continuación, se describen los 6 clusters, sus características principales y las estrategias recomendadas para cada grupo:

Cluster 0: Jóvenes de Baja Actividad

- Edad media: 25.87 años.
- Productos contratados: 1.093 productos.
- Clientes activos: 28.4%.
- Perfil: Jóvenes con baja actividad y bajo compromiso con la entidad. Son clientes de bajo valor con poca participación en productos de financiación.
- Estrategias: Incentivar el uso de los servicios mediante programas de recompensas y ofertas personalizadas.

Cluster 1: Nuevos Clientes de Bajo Compromiso

- Edad media: 35.2 años.
- Productos contratados: 0.485 productos.
- Clientes activos: 16.6%.
- Perfil: Clientes de mediana edad, que ingresaron recientemente con baja actividad. La mayoría ingresó por canales secundarios.
- Estrategias: Focalizar en campañas de enganche personalizadas y mejorar la experiencia digital.

Cluster 2: Jóvenes Inactivos Sin Productos

- Edad media: 28.88 años.
- Productos contratados: 0.039 productos.
- Clientes activos: 3.7%.
- Perfil: Jóvenes con muy poca actividad y prácticamente ningún producto contratado. Están en riesgo de abandono.
- Estrategias: Ofrecer incentivos para reactivar la actividad, como descuentos y promociones.

Cluster 3: Jóvenes Activos en Crecimiento

- Edad media: 23.27 años.
- Productos contratados: 1.08 productos.
- Clientes activos: 40.6%.
- Perfil: El grupo más joven, con un alto nivel de actividad y potencial de crecimiento en el uso de productos financieros.
- Estrategias: Desarrollar programas educativos financieros y promociones adaptadas a sus necesidades.

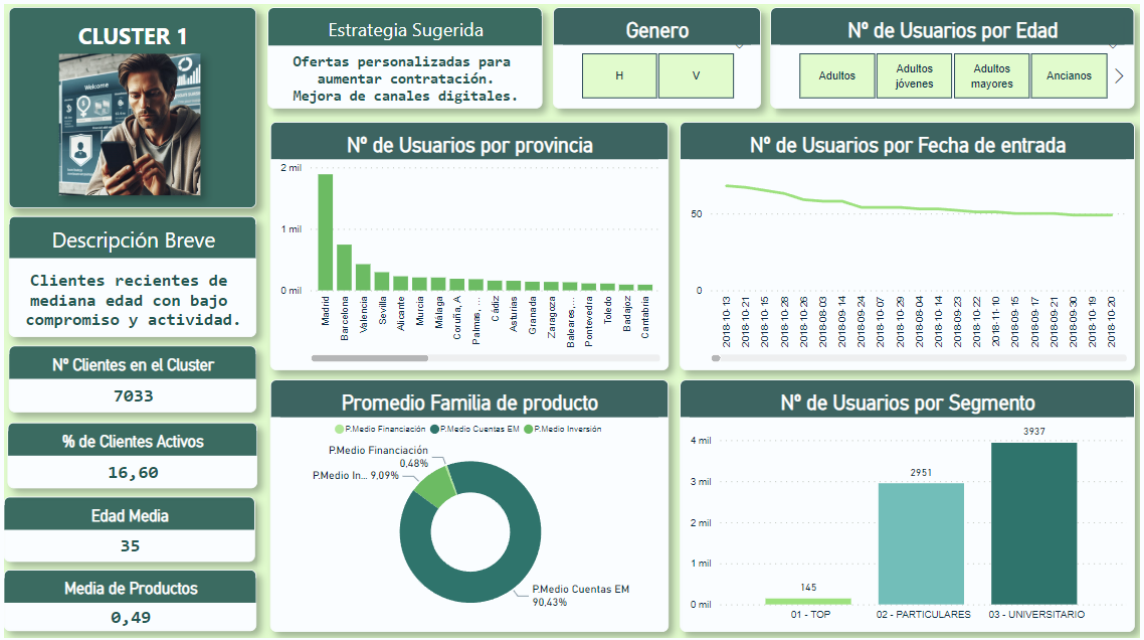
Cluster 4: Clientes Premium Altamente Comprometidos

- Edad media: 37.41 años.
- Productos contratados: 3.871 productos.
- Clientes activos: 94.5%.
- Perfil: Clientes con un alto compromiso y gran valor. Son usuarios intensivos de productos financieros complejos.
- Estrategias: Mantener la lealtad mediante programas de fidelización y asesoría personalizada.

Cluster 5: Seniors Comprometidos con Potencial

- Edad media: 49.63 años.
- Productos contratados: 1.401 productos.
- Clientes activos: 75.4%.
- Perfil: Clientes de mayor edad, con alta actividad y un nivel moderado de contratación de productos. Tienen potencial para adoptar productos más complejos.
- Estrategias: Ofrecer productos financieros adaptados y organizar eventos que fortalezcan la relación con la entidad.

También hemos elaborado un **Dashboard** relativo a la tarea de segmentación, con el propósito de permitir una toma de decisiones más informada en cuanto a las estrategias comerciales, identificando los grupos con mayor necesidad de atención, así como aquellos con el mayor potencial de crecimiento. Cada **cluster** representa un segmento específico con características y comportamientos claramente diferenciados, lo que facilita la implementación de **campañas personalizadas** y **enfoques más efectivos**.

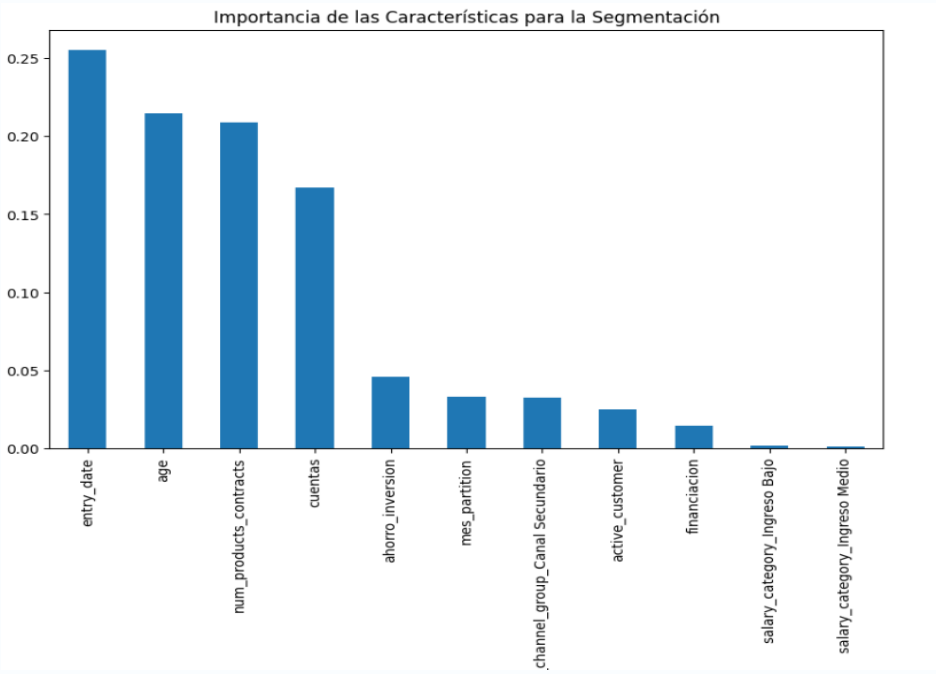


2.6.4. Análisis de la Importancia de Características

Para identificar las características más importantes que diferencian los clusters, se utilizó un modelo de **Random Forest**, que permitió evaluar el peso relativo de cada variable en la segmentación.

Las variables con mayor peso relativo en la segmentación de clientes son:

- **entry_date** y **age** resultaron ser las variables más relevantes para la segmentación, lo que indica que el comportamiento de los clientes varía en función de cuándo ingresaron al sistema y su edad.
- **num_products_contracts** (número de productos contratados) también fue una variable importante, reflejando el nivel de compromiso de los clientes con la entidad.
- Variables como **cuentas**, **ahorro_inversion** y **active_customer** también aportaron información relevante, aunque en menor medida.



Tarea 3 – Recomendación

3.1. Enfoque del Modelo y Variable Target

Para la recomendación de productos, tras explorar diversas opciones, decidimos codificar la variable objetivo del modelo en función de los productos **sale_credit_card** y **pension_plan**. Esta decisión se tomó con el propósito de maximizar los beneficios y la efectividad de la campaña de marketing, dado que estos productos pertenecen a las dos categorías que generan un mayor beneficio para la empresa: financiación (60€/producto) e inversión (40€/producto).

Este enfoque no solo nos permite aprovechar los productos más rentables, sino también diversificar el riesgo inherente a la oferta de productos de financiación. Aunque estos productos ofrecen un mayor retorno, también conllevan riesgos asociados a la prestación del servicio financiero.

Sin embargo, basándonos en los resultados obtenidos de la multiplicación entre la **probabilidad de compra** y el **retorno esperado**, decidimos dirigir nuestros esfuerzos exclusivamente hacia los potenciales compradores de **tarjetas de crédito**. Por tanto, la variable objetivo del modelo se definió como **1 (True)** si un cliente ha contratado una tarjeta de crédito en algún momento.

Adicionalmente, el modelo se utilizará para realizar predicciones sobre aquellos clientes que aún no han contratado una tarjeta de crédito, con el fin de identificar a los que tienen mayor probabilidad de hacerlo en el futuro. Este planteamiento asegura que se optimicen las campañas de marketing, centrándose en los clientes con mayor potencial de conversión.

A continuación, se detallan las razones que sustentan esta decisión:

1. **Producto de Tarjeta de Crédito**: Nos enfocamos en la tarjeta de crédito como producto objetivo de la campaña, ya que es el producto de financiación con mayor representación en nuestra base de clientes y con mayor rentabilidad (60€/producto). Esto maximiza el beneficio económico de la campaña.
2. **Captura del comportamiento histórico del cliente**: La creación de una variable *sale* para cada producto, incluida la tarjeta de crédito, permite consolidar el historial de contratación de productos de cada cliente. Este historial es un predictor clave en las decisiones de compra futuras.
3. **Optimización de recursos de marketing**: Al predecir qué clientes sin tarjeta de crédito tienen mayor probabilidad de contratar una, podemos enfocar los esfuerzos de marketing en estos clientes, mejorando la eficiencia, reduciendo costos y maximizando el retorno de inversión (ROI).
4. **Mejor representación de clientes con tarjeta de crédito**: Al utilizar *sale_credit_card* como objetivo, garantizamos una mayor representación de clientes con tarjeta, equilibrando las clases y mejorando la generalización del modelo.
5. **Reducción de duplicidad y ruido**: Al consolidar el comportamiento histórico en una única variable por producto, evitamos la redundancia y el ruido, mejorando la precisión del modelo.

3.2. Generación del modelo

3.2.1. Filtrado y preprocesamiento

Se creó una función para calcular si el cliente había adquirido algún producto financiero a lo largo de su historial en la plataforma. Posteriormente, el dataset fue filtrado para trabajar solo con la última partición de datos y eliminar duplicados. Se eliminaron también varias columnas irrelevantes, y se ajustaron las categorías de las variables.

```
def calculate_historical_sales(df, product_columns):
    # Crear un DataFrame temporal con solo las columnas de productos
    df_products = df[product_columns]

    # Calcular la suma de cada producto por cliente
    product_sums = df_products.groupby(df['pk_cid']).sum()

    # Convertir las sumas a 1 (si > 0) o 0
    product_sales = (product_sums > 0).astype(int)

    # Renombrar las columnas
    product_sales.columns = [f'sale_{col}' for col in product_sales.columns]

    # Fusionar los resultados con el DataFrame original
    return df.merge(product_sales, left_on='pk_cid', right_index=True, how='left')

# Lista de productos
product_columns = [
    'credit_card', 'short_term_deposit', 'loans', 'mortgage', 'funds',
    'securities', 'long_term_deposit', 'em_account_pp', 'payroll',
    'pension_plan', 'payroll_account', 'emc_account', 'debit_card',
    'em_account_p', 'em_acount'  # Nota: corregí 'em_acount' que estaba mal escrito en el original
]

# Aplicar la función
df_full = calculate_historical_sales(df_full, product_columns)
```

3.2.2. Matriz de correlación y eliminación de variables

Se calculó la matriz de correlación para las variables numéricas. Aquellas variables con una correlación superior al 0.90 fueron eliminadas para evitar la multicolinealidad, mejorando así la estabilidad y precisión del modelo.

Columnas que no pasan al siguiente análisis debido a alta correlación: ['pension_plan', 'cuentas_sum', 'inversion_sum', 'financiacion_sum', 'profit_cuentas', 'profit_inversion', 'profit_financiacion', 'sale_mortgage', 'sale_securities', 'sale_pension_plan', 'sale_payroll_account', 'sale_emc_account', 'sale_em_account_p']

3.2.3. Preparación de los datos para el modelo

El conjunto de datos se preparó mediante un proceso exhaustivo que incluyó la separación de las variables predictoras (X) de la variable objetivo (y), con el fin de aislar las características que ayudarían a predecir la contratación de tarjetas de crédito.

Dado que la mayoría de los clientes no había contratado una tarjeta de crédito, el conjunto de datos presentaba un desequilibrio significativo entre las clases, lo que podría afectar negativamente al rendimiento del modelo predictivo. La mayoría de los algoritmos de machine learning tienden a sesgarse hacia la clase mayoritaria en situaciones de desequilibrio, lo que

puede resultar en un modelo que simplemente prediga que los clientes no contratarán una tarjeta de crédito en la mayoría de los casos.

Para abordar este problema, se implementó una técnica de **undersampling** en la que se redujo el número de ejemplos de la clase mayoritaria (clientes sin tarjeta de crédito) para equilibrar el número de ejemplos de ambas clases. Concretamente, se utilizó un ratio de **0.6** entre los clientes sin tarjeta (clase 0) y los clientes con tarjeta (clase 1). Esto permitió que las clases estuvieran más equilibradas, reduciendo el riesgo de que el modelo sesgara sus predicciones hacia la clase dominante.

Este proceso de **undersampling** garantiza que el modelo tenga suficientes ejemplos de clientes con tarjeta de crédito para aprender patrones relevantes y mejorar su capacidad para identificar correctamente aquellos clientes que podrían estar interesados en adquirir una tarjeta. Aunque el undersampling puede reducir el tamaño total del conjunto de datos, en este caso resultó ser una solución efectiva dado el volumen de datos disponible, permitiendo así construir un modelo más robusto y preciso en la predicción de la clase minoritaria.

3.2.4. Codificación y escalado

Se aplicó **One-Hot Encoding** para las variables categóricas y se utilizó **RobustScaler** para escalar las variables numéricas, asegurando así que todas las características estén en un rango comparable antes de ser utilizadas en el modelo.

```
# One-Hot Encoding para las variables categóricas
categorical_cols = X_resampled.select_dtypes(include=['object', 'category']).columns
ohe = OneHotEncoder(sparse_output=False, drop='first')
X_ohe = pd.DataFrame(ohe.fit_transform(X_resampled[categorical_cols]),
                     columns=ohe.get_feature_names_out(categorical_cols), index=X_resampled.index)
```

```
# RobustScaler para las variables numéricas
numerical_cols = X_resampled.select_dtypes(include=['int64', 'float64','int32','int8']).columns
scaler = RobustScaler()
X_scaled = pd.DataFrame(scaler.fit_transform(X_resampled[numerical_cols]),
                        columns=numerical_cols, index=X_resampled.index)
```

3.2.5. Separación de los datos en Train y Test

El conjunto de datos fue dividido en entrenamiento y prueba utilizando una proporción del 80% para entrenamiento y el 20% restante para prueba. Esta división es crucial para evaluar el rendimiento del modelo en datos no vistos y prevenir el sobreajuste, asegurando que el modelo pueda generalizar correctamente a nuevos datos.

3.2.6. Selección del Modelo: XGBoost

El modelo seleccionado para esta tarea fue **XGBoost** (*Extreme Gradient Boosting*), un algoritmo que ha demostrado un buen rendimiento en aplicaciones prácticas, especialmente en problemas que involucran grandes volúmenes de datos y desequilibrios de clases, como es el caso en nuestro análisis.

XGBoost ofrece varias ventajas:

1. **Rendimiento**: Es conocido por su capacidad para manejar grandes conjuntos de datos con tiempos de entrenamiento relativamente cortos.
2. **Regularización**: XGBoost incluye términos de regularización que ayudan a evitar el sobreajuste, lo que mejora la capacidad del modelo para generalizar en nuevos datos.
3. **Manejo de datos faltantes**: El algoritmo puede tratar datos faltantes de manera eficiente, creando ramas alternativas en los árboles de decisión para los valores ausentes.

Optimización de Hiperparámetros

Dado que XGBoost tiene una serie de hiperparámetros que pueden afectar significativamente su rendimiento, se implementó **RandomizedSearchCV** para optimizar dichos parámetros.

Los hiperparámetros ajustados incluyeron:

- **n_estimators**: Número de árboles de decisión que se construirán.
- **max_depth**: La profundidad máxima de los árboles, que controla la complejidad del modelo.
- **learning_rate**: Tasa de aprendizaje que regula la contribución de cada árbol en la iteración final del modelo.
- **subsample**: Fracción de muestras utilizadas en cada árbol, para evitar sobreajuste.
- **colsample_bytree**: Fracción de características a utilizar en cada árbol, lo que puede reducir la correlación entre árboles.
- **gamma**: Umbral para reducir la partición de nodos, que ayuda a evitar particiones innecesarias.
- **min_child_weight**: Peso mínimo que debe tener una hoja, lo que permite controlar la sobreadaptación.

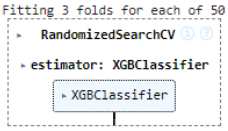
```
# Crear el modelo de XGBoost
xgb_model = xgb.XGBClassifier()
```

```
# Definir el grid de hiperparámetros a optimizar
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [3, 5, 7, 10],
    'learning_rate': [0.01, 0.1, 0.2],
    'subsample': [0.6, 0.8, 1.0],
    'colsample_bytree': [0.6, 0.8, 1.0],
    'gamma': [0, 0.1, 0.3],
    'min_child_weight': [1, 3, 5]
}
```

```
# Usar RandomizedSearchCV para buscar los mejores hiperparámetros
random_search = RandomizedSearchCV(xgb_model, param_distributions=param_grid, n_iter=50, scoring='accuracy',
                                   n_jobs=-1, cv=3, verbose=2, random_state=42)
```

```
# Entrenar con los datos de entrenamiento
random_search.fit(X_train, y_train,verbose=1)
```

```
Fitting 3 folds for each of 50 candidates, totalling 150 fits
```




```
# Ver los mejores hiperparámetros encontrados
print("Mejores hiperparámetros:", random_search.best_params_)
```

```
Mejores hiperparámetros: {'subsample': 1.0, 'n_estimators': 100, 'min_child_weight': 1, 'max_depth': 3, 'learning_rate': 0.2, 'gamma': 0.3, 'colsample_bytree': 1.0}
```

Utilizando **RandomizedSearchCV**, se evaluaron 50 combinaciones de estos hiperparámetros a través de validación cruzada con 3 *folds* (particiones del conjunto de datos). Esta técnica permite evaluar el rendimiento del modelo de manera más robusta, garantizando que el rendimiento no dependa del conjunto de prueba utilizado.

Tras la optimización, los mejores hiperparámetros obtenidos fueron:

- subsample: 1.0
- n_estimators: 100
- min_child_weight: 1
- max_depth: 3
- learning_rate: 0.2
- gamma: 0.3
- colsample_bytree: 1.0

Estos hiperparámetros optimizados sugieren que el modelo utiliza un enfoque prudente en cuanto a la profundidad de los árboles y el número de estimadores. El **learning rate** de 0.2 asegura que las actualizaciones se realicen de manera gradual para evitar ajustes demasiado drásticos en cada iteración. El valor de **gamma** también ayuda a prevenir la sobrefragmentación de los nodos, lo que es clave para evitar sobreajuste en árboles profundos.

3.3. Evaluación del Modelo

En este apartado, se evaluó el rendimiento del modelo XGBoost optimizado mediante la predicción sobre el conjunto de prueba. Los resultados obtenidos se analizaron utilizando diferentes métricas de clasificación, así como herramientas gráficas que ayudan a comprender mejor el comportamiento del modelo.

3.3.1. Predicciones y Cálculo de Métricas

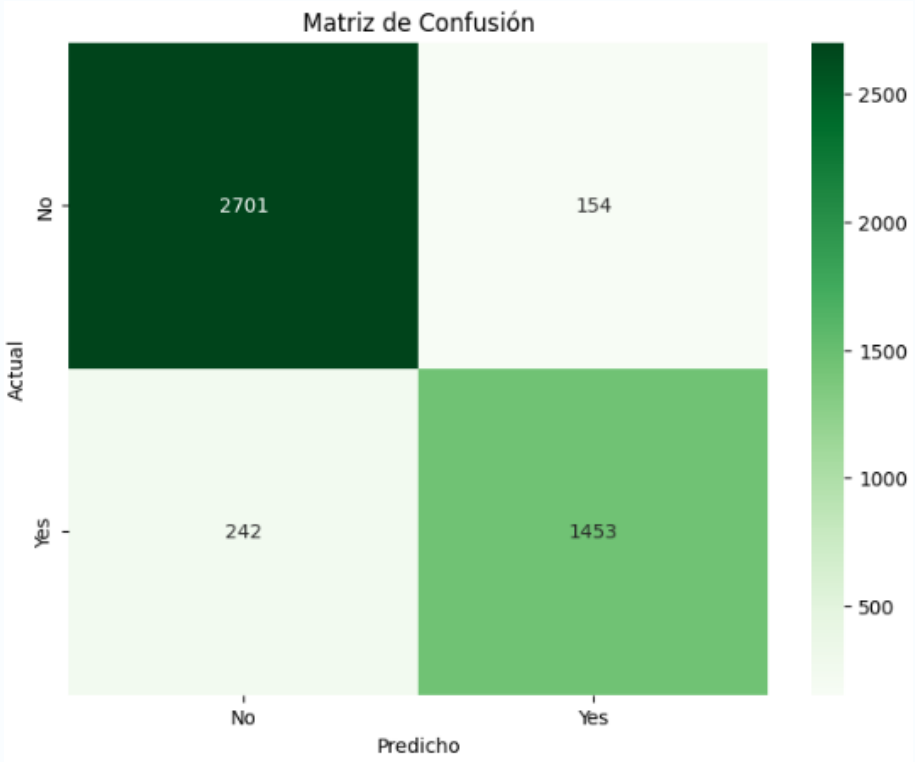
El modelo fue utilizado para hacer predicciones tanto de las clases como de las probabilidades de pertenencia a la clase positiva (contratar una tarjeta de crédito). Se calcularon las métricas clave de evaluación: **accuracy**, **ROC AUC**, **matriz de confusión** y el **reporte de clasificación**.

- **Accuracy**: El modelo alcanzó una precisión del **91%**, lo que indica que clasifica correctamente una alta proporción de las observaciones.
- **ROC AUC**: El área bajo la curva ROC fue de **0.97**, lo cual es notablemente alto, indicando una excelente capacidad del modelo para distinguir entre clientes que contratarán y no contratarán la tarjeta de crédito.
- **Reporte de clasificación**: El modelo presenta una precisión de **0.92** para la clase 0 (clientes que no contratarán la tarjeta) y de **0.90** para la clase 1 (clientes que sí la contratarán), lo que refleja un buen balance entre las clases. El **recall** para la clase 1 fue de **0.86**, lo que significa que el modelo es capaz de identificar el 86% de los clientes que realmente contratarán la tarjeta.

```
Accuracy: 0.91
ROC AUC: 0.97

Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.95      0.93      2855
           1       0.90      0.86      0.88      1695

    accuracy                           0.91      4550
   macro avg       0.91      0.90      0.91      4550
weighted avg       0.91      0.91      0.91      4550
```

3.3.2. Matriz de Confusión

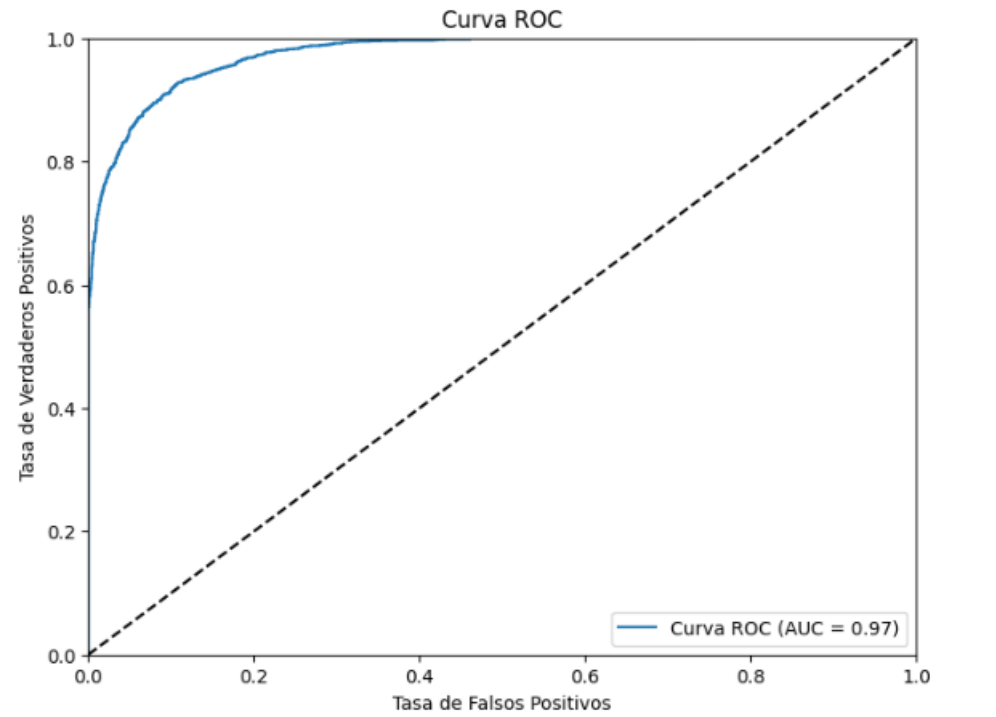
La matriz de confusión permite visualizar el número de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos, proporcionando una representación clara del rendimiento del modelo en cada clase.



La matriz muestra que el modelo clasifica correctamente a la mayoría de los clientes. Sin embargo, detectamos **154 falsos positivos** (clientes que no contratarán la tarjeta, pero fueron clasificados como que sí lo harán) y **242 falsos negativos** (clientes que sí contratarán la tarjeta, pero fueron clasificados como que no lo harán). A pesar de estos errores, el rendimiento general del modelo es muy aceptable, especialmente considerando la naturaleza del problema.

3.3.3. Curva ROC AUC

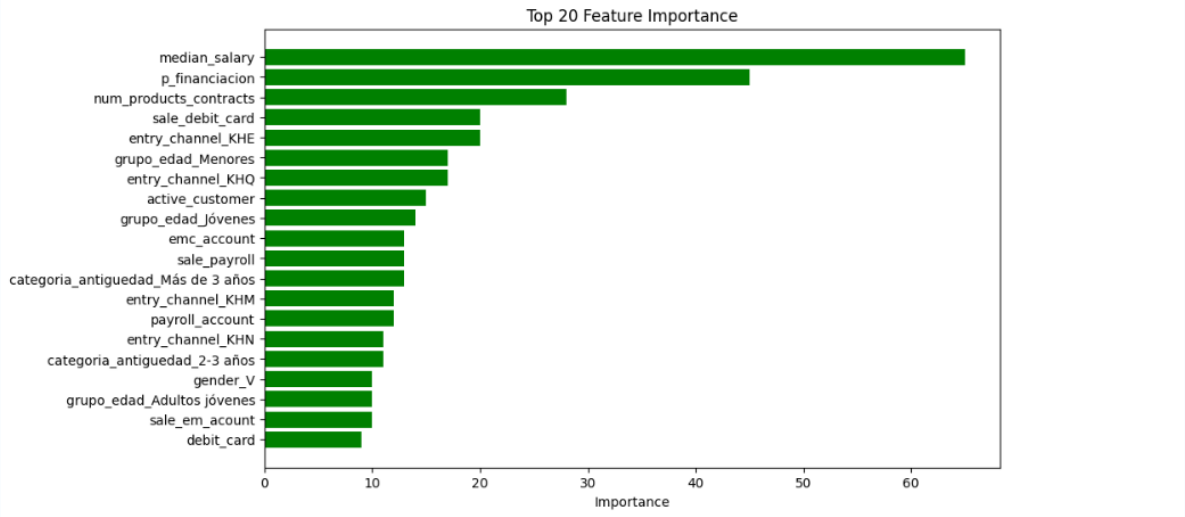
La curva ROC (Receiver Operating Characteristic) evalúa el rendimiento del modelo en distintos umbrales de clasificación. La curva ROC es una herramienta fundamental para medir la capacidad de un modelo de clasificación binaria para discriminar entre clases.



La curva ROC obtenida muestra un área bajo la curva de **0.97**, lo que implica una capacidad casi perfecta del modelo para discriminar entre las dos clases. Cuanto más cerca está la curva de la esquina superior izquierda, mejor es el rendimiento del modelo.

3.3.4. Importancia de las Características

Uno de los aspectos cruciales al entrenar un modelo de *machine learning* es comprender qué características contribuyen más a las predicciones. En este caso, se evaluó la importancia de las características utilizadas por el modelo XGBoost.



Principales Características:

1. **Salario (median_salary)**: Es la característica más importante en el modelo, lo cual tiene sentido, ya que los clientes con salarios más altos suelen tener mayor capacidad para gestionar créditos y tarjetas de crédito.

2. **Productos de Financiación Contratados (p_financiacion)**: Los clientes con productos de financiación tienden a estar más familiarizados con el uso de productos financieros complejos, lo que aumenta la probabilidad de que contraten una tarjeta de crédito.
3. **Número de Productos Contratados (num_products_contracts)**: Indica el compromiso del cliente con la entidad financiera. Cuantos más productos tiene un cliente, mayor es su predisposición a contratar una tarjeta de crédito.
4. **Venta de Tarjeta de Débito (sale_debit_card)**: Los clientes que ya tienen una tarjeta de débito suelen estar más familiarizados con los servicios bancarios, lo que los hace más propensos a contratar productos de crédito.

Estas características ofrecen una idea clara de los factores que más influyen en la decisión de los clientes de adquirir una tarjeta de crédito, proporcionando insights valiosos que podrían ser utilizados por el equipo de marketing para personalizar futuras campañas.

3.4. Predicciones para Clientes sin Tarjeta de Crédito

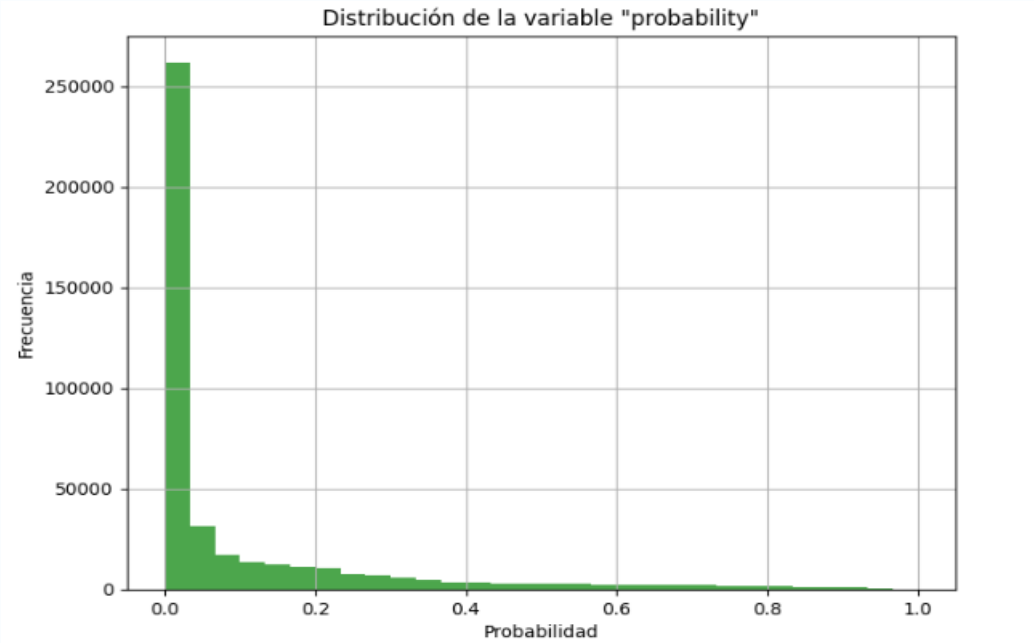
Una vez entrenado el modelo, el siguiente paso consistió en predecir qué clientes que actualmente no tienen una tarjeta de crédito tienen más probabilidades de contratar una. Esta etapa es fundamental para enfocar los esfuerzos de marketing hacia aquellos clientes con mayor probabilidad de conversión.

Se seleccionaron los **10,000 clientes** con mayor probabilidad de adquirir una tarjeta de crédito y se visualizó su distribución. Los **10,000 clientes seleccionados** representan la mejor oportunidad para maximizar la efectividad de la campaña de marketing, al enfocarse en aquellos con una mayor probabilidad de adquirir una tarjeta de crédito.

Una vez que se obtuvieron las probabilidades de contratación de la tarjeta de crédito para los clientes sin tarjeta en nuestro dataset, se procedió a realizar un análisis detallado de la distribución de dichas probabilidades. Esto nos permitió identificar a los **10,000 clientes** con las probabilidades más altas de contratar el producto, optimizando así los recursos para la campaña de marketing.

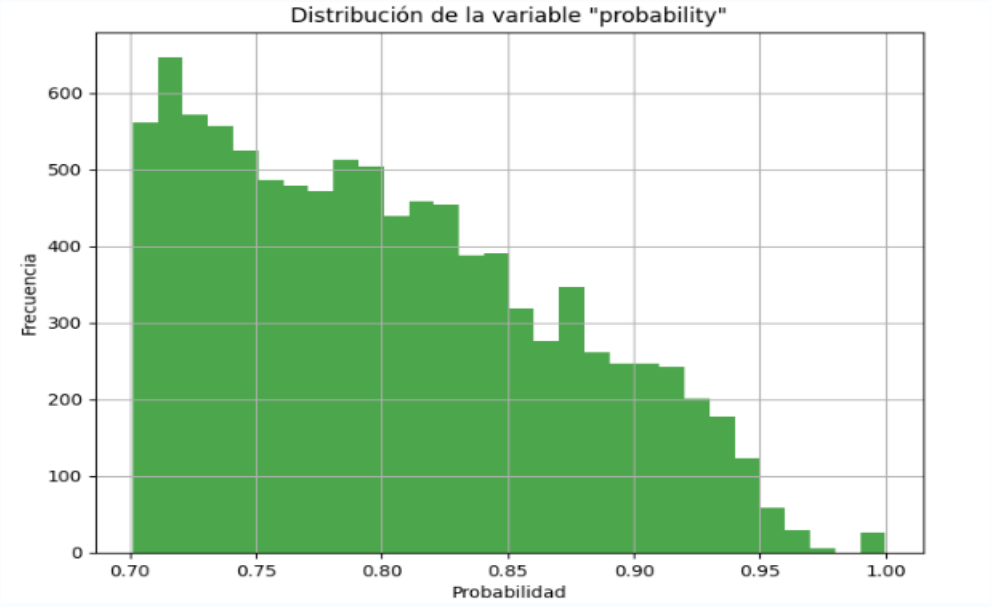
3.4.1. Análisis de la Distribución de Probabilidades

Al observar la distribución de la variable **"probability"**, notamos que la mayoría de los clientes sin tarjeta de crédito presentan una **baja probabilidad de adquirirla**, con muchas probabilidades cercanas a **0**. Este comportamiento es esperado, dado que una porción significativa de la base de clientes no parece estar interesada en contratar este producto o no presenta los atributos suficientes para que el modelo prediga una alta probabilidad de conversión.



3.4.2. Identificación de Clientes con Mayor Probabilidad

Estos 10.000 clientes representan la mejor oportunidad para maximizar la efectividad de la campaña de email, mejorando el **retorno sobre la inversión** (ROI) y reduciendo costos al dirigirse a un grupo específico de usuarios con mayor potencial de conversión.



El **top 10,000** de clientes fue seleccionado y almacenado para su uso en las siguientes etapas de la campaña.

- Suma de probabilidades: 8,029.626

Este valor indica que, en promedio, la probabilidad total de contratación en el grupo seleccionado es elevada, cercana al **80%**, lo que sugiere que la **campaña dirigida** tiene un alto potencial de éxito.

En cuanto a la distribución, la mayoría de estos clientes con mayores probabilidades se encuentran en un rango entre **0.7** y **0.85**, con menos clientes en los extremos superiores de la

probabilidad. Esto implica que, aunque se espera una **tasa de conversión alta** en este grupo, hay menos clientes con probabilidades extremadamente altas (mayores a **0.95**), lo que indica que un enfoque **personalizado** en los correos electrónicos podría maximizar las conversiones.

3.5. Recomendación de Productos de Inversión y Ahorro

En esta sección, desarrollamos un segundo modelo predictivo con el propósito de recomendar productos de **ahorro e inversión**. Este modelo tiene un enfoque más general en la capacidad de adaptación del sistema de machine learning a diferentes productos financieros, mostrando así su flexibilidad para campañas futuras más allá de la campaña actual centrada en la tarjeta de crédito. A continuación, se detallan los principales aspectos y la justificación detrás de este modelo:

3.5.1. Producto Representativo de Ahorro/Inversión

El modelo se enfocará en el **producto de ahorro o inversión más representativo** de nuestra base de datos, seleccionado en base a la frecuencia de contratación. Estos productos son fundamentales para aquellos clientes que buscan estabilidad financiera a largo plazo, generando ingresos recurrentes para la entidad. Según los datos disponibles, la **rentabilidad promedio** de estos productos es de **40€ por venta**, lo que destaca su importancia como fuente de ingresos continua.

3.5.2. Adaptabilidad del Modelo

El desarrollo de este modelo para productos de inversión y ahorro **refuerza la adaptabilidad** del enfoque de machine learning, evidenciando que puede ajustarse fácilmente para predecir la adquisición de productos financieros específicos. Aunque la campaña actual se centra en productos de financiación, como la tarjeta de crédito, este modelo demuestra que el sistema puede ser **reutilizado** para otros productos relevantes, como los de ahorro e inversión, proporcionando flexibilidad para futuras iniciativas comerciales.

La estructura del modelo sigue una **metodología similar** a la utilizada en el producto de tarjeta de crédito. Se definió una variable objetivo que representa si el cliente ha contratado algún **producto de inversión o ahorro** en el pasado, permitiendo al modelo predecir la probabilidad de que otros clientes puedan hacerlo en el futuro.

3.5.3. Priorización de Productos Rentables

Aunque este modelo no será implementado en la campaña de marketing actual, su valor radica en la **exploración de nuevas oportunidades** para maximizar ingresos en futuras campañas. Los productos de inversión y ahorro suelen atraer a un tipo de cliente más orientado a la **seguridad financiera** y, dado su perfil de cliente, representan una **fuente de ingresos valiosa**. Este enfoque de priorización permitiría a la compañía **diversificar** su estrategia de marketing en el futuro, maximizando la rentabilidad tanto de productos de financiación como de ahorro.

3.5.4. Captura del Comportamiento Histórico

Al igual que con el modelo de tarjeta de crédito, se generará una **variable de "venta"** (sale) para el producto de inversión o ahorro seleccionado, consolidando así el historial de contratación de

estos productos por parte de los clientes. Esto es crucial para poder identificar **patrones de comportamiento** y determinar cuáles clientes podrían estar más inclinados a adquirir dichos productos en el futuro. Este enfoque histórico es clave para mejorar la precisión del modelo y orientar de manera más eficaz las campañas futuras.

3.5.5. Focalización de Recursos para Futuras Campañas

La creación de este modelo no solo busca demostrar la capacidad de predicción ajustada a diversos productos financieros, sino que también permite a la compañía explorar la viabilidad de enfocarse en clientes con mayor probabilidad de contratación de productos de ahorro e inversión. En futuras campañas, este tipo de modelo podría ayudar a mejorar el **retorno de la inversión (ROI)**, permitiendo focalizar esfuerzos de marketing en segmentos con **mayor potencial de conversión**.

3.6. Productivización

En este apartado se desarrollan las especificaciones para la plataforma de producción que permitirá operar los modelos a medio plazo utilizando AWS. La propuesta incluye el almacenamiento de datos y modelos en Amazon S3, el uso de Amazon SageMaker para el entrenamiento del modelo y AWS Lambda para desplegar el modelo en producción.

Metodología Propuesta:

1. Ingesta y Almacenamiento de Datos:

Almacenamiento en Amazon S3: Se organizar los datos en buckets y carpetas lógicas para facilitar el acceso y la gestión. Todos los datos de actividad comercial, productos y sociodemografic se almacenan en formatos optimizados como Parquet o CSV. El nombre del bucket de S3 es easy-money-project-bucket y se puede acceder a los archivos a través de una url de la siguiente manera:

Seguridad de Datos: Es altamente recomendable implementar políticas de acceso (IAM roles y políticas de bucket) para asegurar que solo usuarios y servicios autorizados puedan acceder a los datos, en este caso, no se implementan y se da acceso al bucket a todos los usuarios.

2. Procesamiento y limpieza de Datos:

AWS Lambda es un servicio de computación serverless que permite ejecutar código sin la necesidad de gestionar servidores. Lambda es ideal para tareas pequeñas, con tiempos de ejecución cortos, y se escala automáticamente en función de la demanda.

Se utiliza la función AWS Lambda para la limpieza y transformación de los datos crudos. A partir de un script en Python se implementa la lógica de limpieza y transformación. Se guardar los datos transformados, en un área designada como s3://easy-money-project-bucket/feature-df_full_cleaned.parquet/ pudiéndose acceder a la misma de la siguiente forma:

df = pd.read_parquet("https://easy-money-project-bucket.s3.eu-west-3.amazonaws.com/df_full_cleaned.parquet")

3. Entrenamiento y Almacenamiento del Modelo:

Entrenamiento del Modelo: Para entrenar el modelo se utiliza el servicio Amazon SageMaker que permite crear, entrenar e implementar modelos de ML a escala mediante herramientas como cuadernos, depuradores, generadores de perfiles, pipelines, MLOps y más, todo en un único entorno de desarrollo integrado. Se desarrolla en python cargando el dataset limpio para el entrenamiento de los modelos.

Se recomienda para este apartado implementar un flujo completo para la colaboración y producción de modelos con MLflow, para que cada data scientist en el equipo puede entrenar y registrar sus modelos desde SageMaker Notebooks utilizando MLflow para rastrear experimentos, y de esta forma comparar los modelos y seleccionar el que tenga el mejor rendimiento.

Almacenamiento del Modelo: Se serializa el modelo entrenado y se almacena en Amazon S3 para su posterior carga en AWS Lambda.

4. Despliegue del Modelo en AWS Lambda:

Despliegue en producción: Se desarrolla otra función Lambda en Python que carga el modelo seleccionado desde S3 durante la inicialización, así como los datos que se van a utilizar para lanzar las predicciones.

Se guardan los resultados arrojados por el modelo en S3 nuevamente, con el objetivo que se pueda acceder a dichos datos por parte del departamento de Marketing, para continuar con el análisis de personalización de la campaña de email, pudiéndose acceder de la siguiente manera:

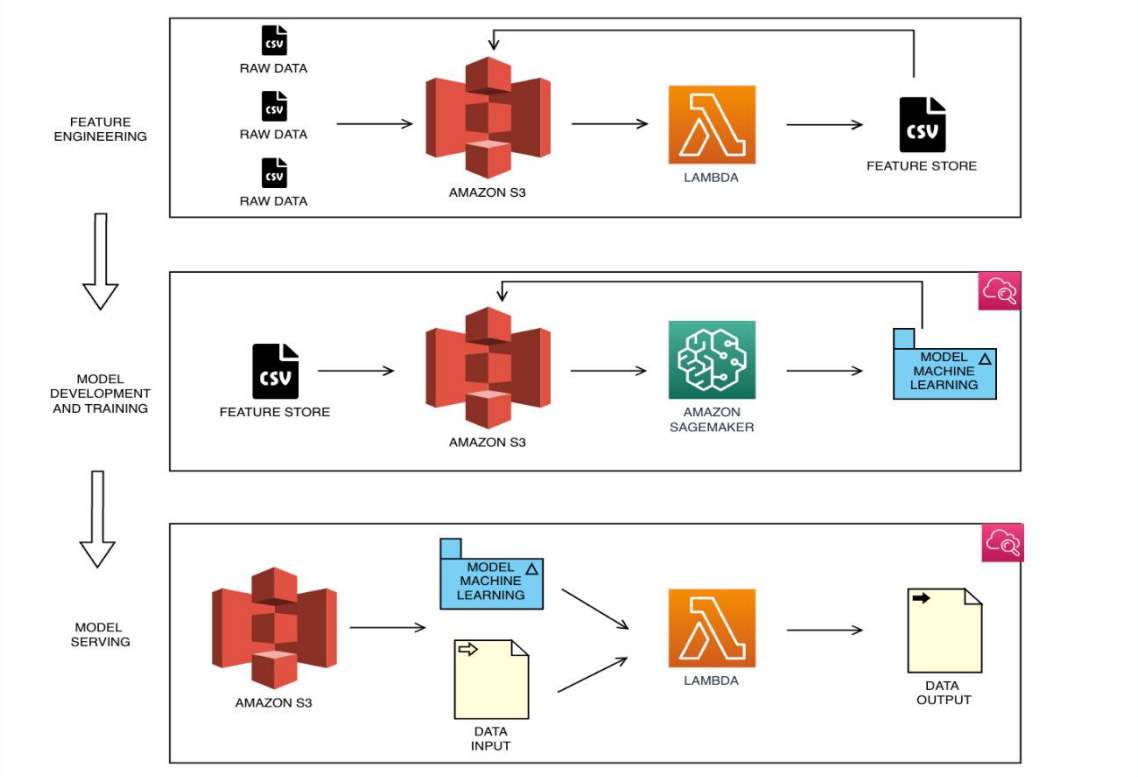
df = pd.read_parquet("https://easy-money-project-bucket.s3.eu-west-3.amazonaws.com/df_seleccionados.parquet")

5. Monitorización y Logging:

Amazon CloudWatch: Se recomienda configurar logs y métricas de la función Lambda en CloudWatch, estableciendo alarmas para detectar anomalías o errores en tiempo real.

Esta metodología permitirá desplegar y operar el modelo de recomendación de manera eficiente, segura y escalable, alineándose con los objetivos comerciales y técnicos de la organización.

DIAGRAMA DE PRODUCTIVIZACIÓN



3.7. ROI ESPERADO

Después de realizar un análisis exhaustivo de la campaña de marketing proyectada para 10,000 clientes con una probabilidad de compra del 80%, hemos obtenido resultados muy prometedores:

- **Beneficio Total Esperado**: Se anticipa un beneficio total de 481,740.00 €, lo que demuestra el alto potencial de retorno que esta campaña puede generar.

- **Beneficio Esperado de las Respuestas**: Basándonos en una tasa de respuesta realista del 50%, el beneficio esperado de las respuestas es de 240,870.00 €. Este cálculo es crucial para estimar el impacto real que podemos esperar de nuestros esfuerzos de marketing.

- **Costo Total de la Campaña**: Con un costo total de 50,000.00 €, que incluye gastos de envío, diseño, plataforma y costos de incentivos, la inversión inicial es razonable dado el potencial de retorno.

- **Beneficio Neto Real**: Tras restar los costos, se proyecta un beneficio neto de 190,870.00 €. Este beneficio neto subraya la viabilidad económica de la campaña.

- **ROI (Retorno sobre la Inversión)**: El ROI calculado es de 381.74%, lo que indica que, por cada euro invertido, se espera obtener aproximadamente 3.82 euros en retorno. Este es un indicador sólido de que la campaña puede ser altamente efectiva y rentable.

En resumen, la campaña presenta una oportunidad excelente para maximizar beneficios a través de una inversión controlada y una planificación estratégica. Recomendamos proceder con la

implementación, monitorizando de cerca las respuestas del mercado para ajustar las estrategias según sea necesario.

```
# 2. Definir y calcular todos los costos de la campaña
# aquí se definen algunos costos de la campaña estimados
costo_por_correo = 0.50
costo_diseño = 15000
costo_plataforma = 10000
costo_incentivos = 20000

costo_total = (10000 * costo_por_correo) + costo_diseño + costo_plataforma + costo_incentivos

# 3. Definir una tasa de respuesta realista basada en datos históricos o benchmarks
tasa_respuesta_real = 0.5

# 4. Calcular el beneficio esperado basado en la tasa de respuesta real
beneficio_respuestas_real = beneficio_total_esperado * tasa_respuesta_real

# 5. Calcular el beneficio neto
beneficio_neto_real = beneficio_respuestas_real - costo_total

# 6. Calcular el ROI realista
roi_real = (beneficio_neto_real / costo_total) * 100

# Mostrar los resultados
print(f"Beneficio Total Esperado: {beneficio_total_esperado:,.2f} €")
print(f"Beneficio Esperado de las Respuestas: {beneficio_respuestas_real:,.2f}€")
print(f"Costo Total de la Campaña: {costo_total:,.2f}€")
print(f"Beneficio Neto Real: {beneficio_neto_real:,.2f}€")
print(f"ROI: {roi_real:.2f}%")
```

```
Beneficio Total Esperado: 481,740.00 €
Beneficio Esperado de las Respuestas: 240,870.00€
Costo Total de la Campaña: 50,000.00€
Beneficio Neto Real: 190,870.00€
ROI: 381.74%
```

Tarea 4 – Personalización

El objetivo principal de este análisis es **segmentar** a los **10,000 clientes** con mayor probabilidad de compra de tarjetas de crédito. La segmentación permitirá personalizar la campaña de marketing a través de correos electrónicos dirigidos a grupos específicos, optimizando así la **efectividad de la campaña**. Este proceso se realiza utilizando técnicas de **clustering**, lo que nos permite agrupar a los clientes en base a características demográficas clave y a su **probabilidad de compra**, obtenida del modelo predictivo desarrollado previamente.

4.1. Contexto y Justificación

Siguiendo la recomendación del equipo de marketing, liderado por Erin, se ha decidido que la segmentación esté basada en **características demográficas** clave, tales como: edad, sexo, ingresos…

Este enfoque tiene como objetivo crear **perfiles personalizados** para cada grupo de clientes, de manera que las creatividades y los mensajes de la campaña de e-mail se alineen con las características y necesidades específicas de cada segmento, aumentando así la tasa de éxito y el retorno de los fondos empleados.

El uso de **K-Nearest Neighbors (KNN)** como algoritmo de clustering nos permitirá identificar el número óptimo de segmentos claramente diferenciados dentro de nuestra base de clientes seleccionados.

4.2. Preparación y Limpieza de Datos

4.2.1. Eliminación de Variables No Relevantes

Se han eliminado varias columnas que no aportan valor directo a la segmentación. Estas columnas incluyen principalmente información relacionada con productos financieros que los clientes ya han contratado, tales como préstamos o hipotecas, y que no son relevantes para nuestro análisis de segmentación demográfica y probabilidad de compra.

Además, se verificó que el conjunto de datos no contiene **valores nulos** ni **filas duplicadas**, asegurando así la consistencia de los datos.

4.2.2. Variables de Segmentación

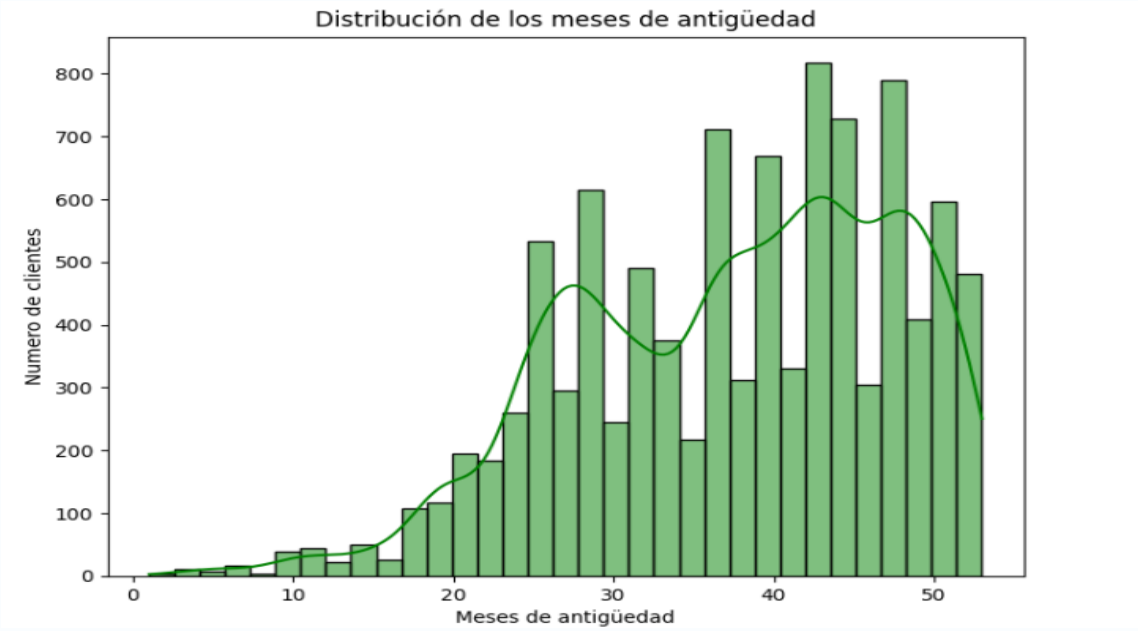
Las variables seleccionadas para la segmentación son: **Edad**, **Ingreso medio (salario)**, **Sexo**, **Meses de antigüedad** como cliente y **Probabilidad de compra** de la tarjeta de crédito

Estas variables permiten capturar tanto el **perfil sociodemográfico** de los clientes como su **comportamiento financiero** y su disposición a adquirir productos financieros.

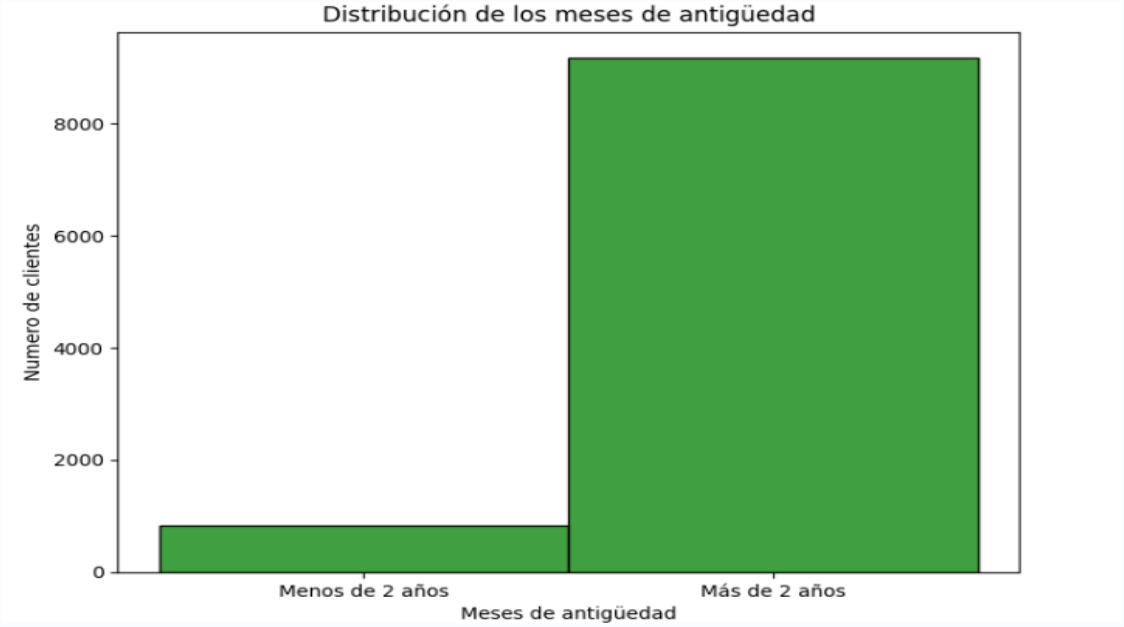
4.2.3. Creación de Nuevas Variables

Meses de Antigüedad

Para mejorar la interpretación de los datos y la segmentación, se calculó la antigüedad de los clientes en meses. Esta información permite clasificar a los clientes según su tiempo de relación con la entidad. Posteriormente, se creó una **categoría binaria** para clasificar a los clientes en dos grupos: aquellos con **menos de 2 años** de antigüedad y aquellos con **más de 2 años**.



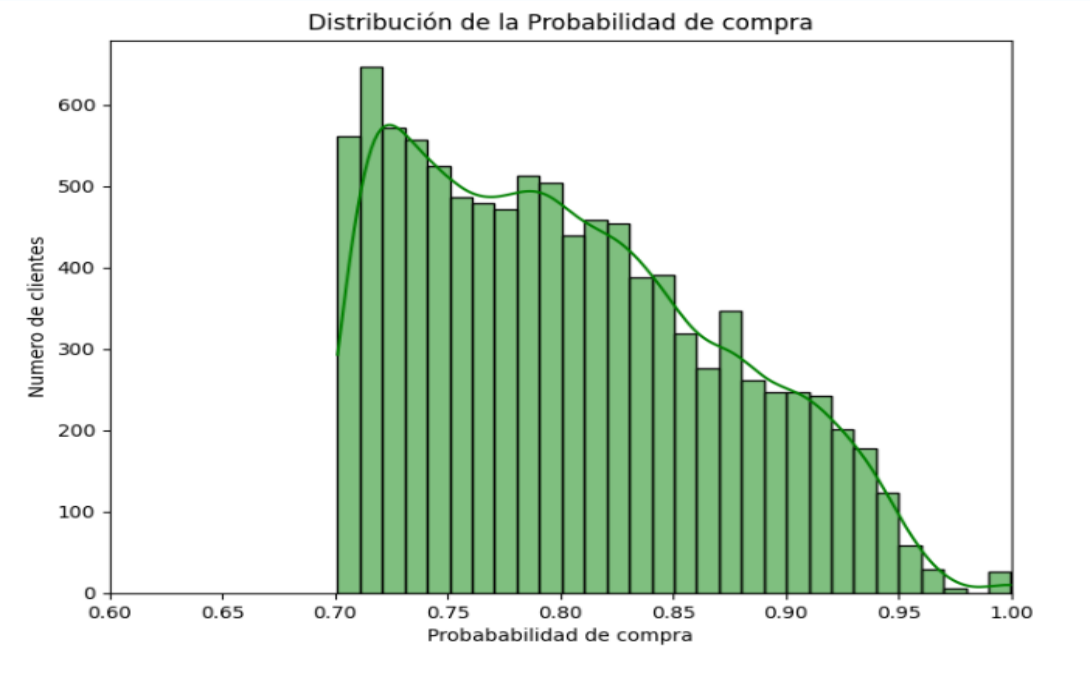
El gráfico de distribución de antigüedad muestra que la mayor parte de los clientes tiene una antigüedad significativa, lo que sugiere una base de clientes con **relaciones de largo plazo**.



4.2.4. Probabilidad de Compra

La **probabilidad de compra** es una variable crucial en este análisis, ya que proviene del modelo predictivo desarrollado previamente. Se ha categorizado en tres grupos:

- **Media** (0.7 - 0.8)
- **Alta** (0.8 - 0.9)
- **Muy Alta** (0.9 - 1)




```
# Crear las categorías para 'probabilidad_compra' utilizando pd.cut
df_joined['categoria_probabilidad'] = pd.cut(df_joined['probability'],
                            bins=[0, 0.80, 0.90, 1.0],
                            labels=['Media', 'Alta', ' Muy Alta'],
                            right=True)
```

Estas categorías nos permiten diferenciar la **intención de compra** y, por lo tanto, ajustar la estrategia de marketing para cada grupo. Clientes con una probabilidad **muy alta** recibirán un enfoque más directo y personalizado, mientras que aquellos con una probabilidad **media** podrían requerir incentivos adicionales para completar la compra.

4.2.5. Edad

Para hacer más manejable la variable de **edad** y facilitar la segmentación, se dividió en tres grupos basados en la distribución de los percentiles:

- **Jóvenes/Adultos**: Menores de 36 años
- **Adultos**: Entre 36 y 47 años
- **Senior**: Mayores de 47 años

Este enfoque permite diseñar estrategias de marketing específicas para cada grupo, que reflejan las **diferencias de comportamiento y preferencias** entre generaciones.

4.2.6. Salario Medio

El **salario medio** de los clientes es una variable importante que permite diferenciar a los grupos en función de su **capacidad económica**. Se han utilizado los percentiles 80 y 90 para identificar a los clientes con **ingresos más elevados**, ya que estos clientes podrían tener **necesidades financieras más sofisticadas** y requerir mensajes de marketing adaptados a productos premium.

4.2.7. Distribución de Género

La **variable de género** también es clave en la segmentación, ya que su distribución está ligeramente desbalanceada, con un **65.19% de hombres (V)** y un **34.81% de mujeres (H)**. Esta segmentación por género permitirá ajustar las campañas de marketing de manera más precisa, con mensajes dirigidos a cada grupo demográfico.

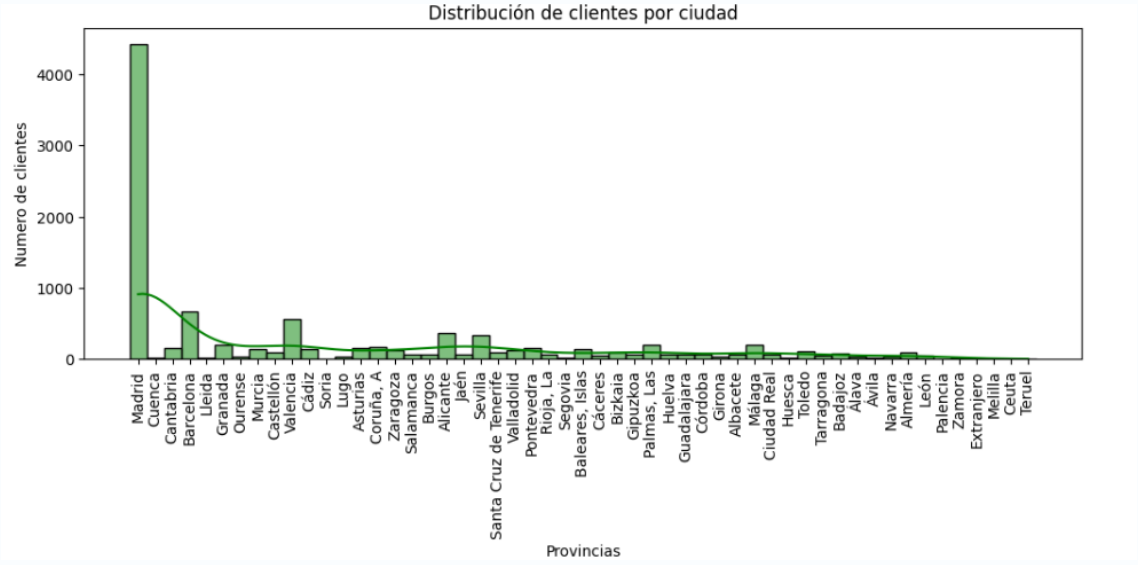
4.2.8. Meses de Antigüedad

Para profundizar en el análisis de antigüedad de los clientes, se calculó la **antigüedad en meses** desde su primera interacción con la entidad financiera. Esto nos permite clasificar a los clientes según su tiempo de relación, diferenciando entre clientes con menos de dos años de antigüedad y aquellos con una relación más establecida.

El análisis de la antigüedad de los clientes es relevante para personalizar las campañas, ya que **los clientes con mayor antigüedad** podrían tener un mayor **grado de confianza** en la entidad y estar más abiertos a recibir ofertas adicionales.

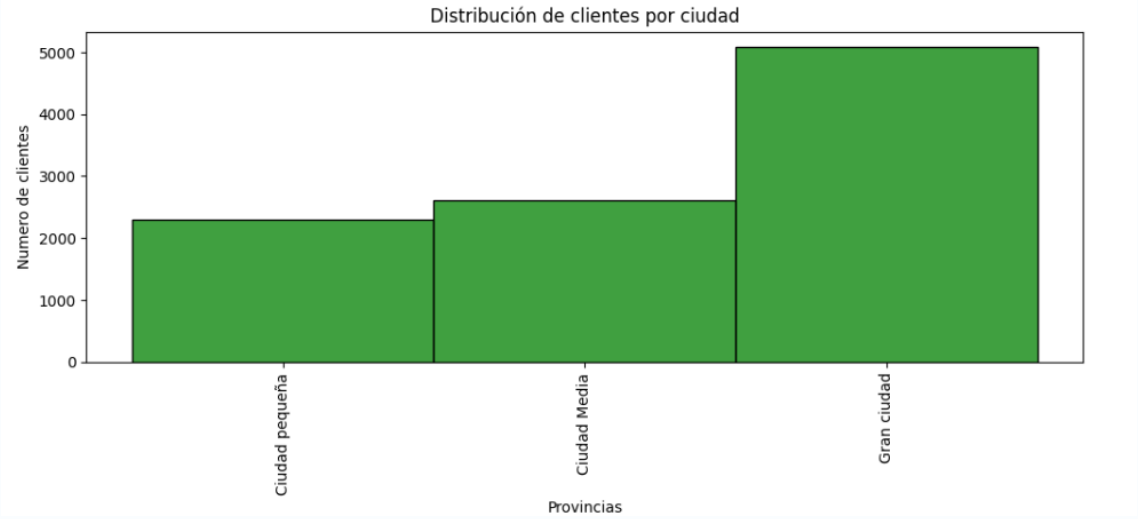
4.2.9. Segmentación por Región

El análisis de la **región** se llevó a cabo sustituyendo los códigos de las provincias por el número de habitantes ya que, haciendo un análisis inicial, observamos una descompensación entre Madrid y el resto de regiones del análisis. Esto nos proporciona una estimación del tamaño de las ciudades de los clientes, lo que nos ayudará a diferenciar entre aquellos que residen en **ciudades pequeñas**, **medias** o **grandes**.



```
[60] df_joined['city_size'] = pd.cut(df_joined['region_code'],
                                     bins=[0,1000000, 3000000, float('inf')],
                                     labels=['Ciudad pequeña',"Ciudad Media" ,'Gran ciudad'],
                                     right=False)
```

La segmentación por tamaño de la ciudad es útil porque los clientes de **ciudades grandes** pueden tener diferentes expectativas financieras en comparación con aquellos que viven en **ciudades pequeñas**. Este análisis nos permitirá enfocar la campaña con creatividades diferenciadas para cada perfil urbano.



4.3. Encoding y Preparación de los Datos para Segmentación

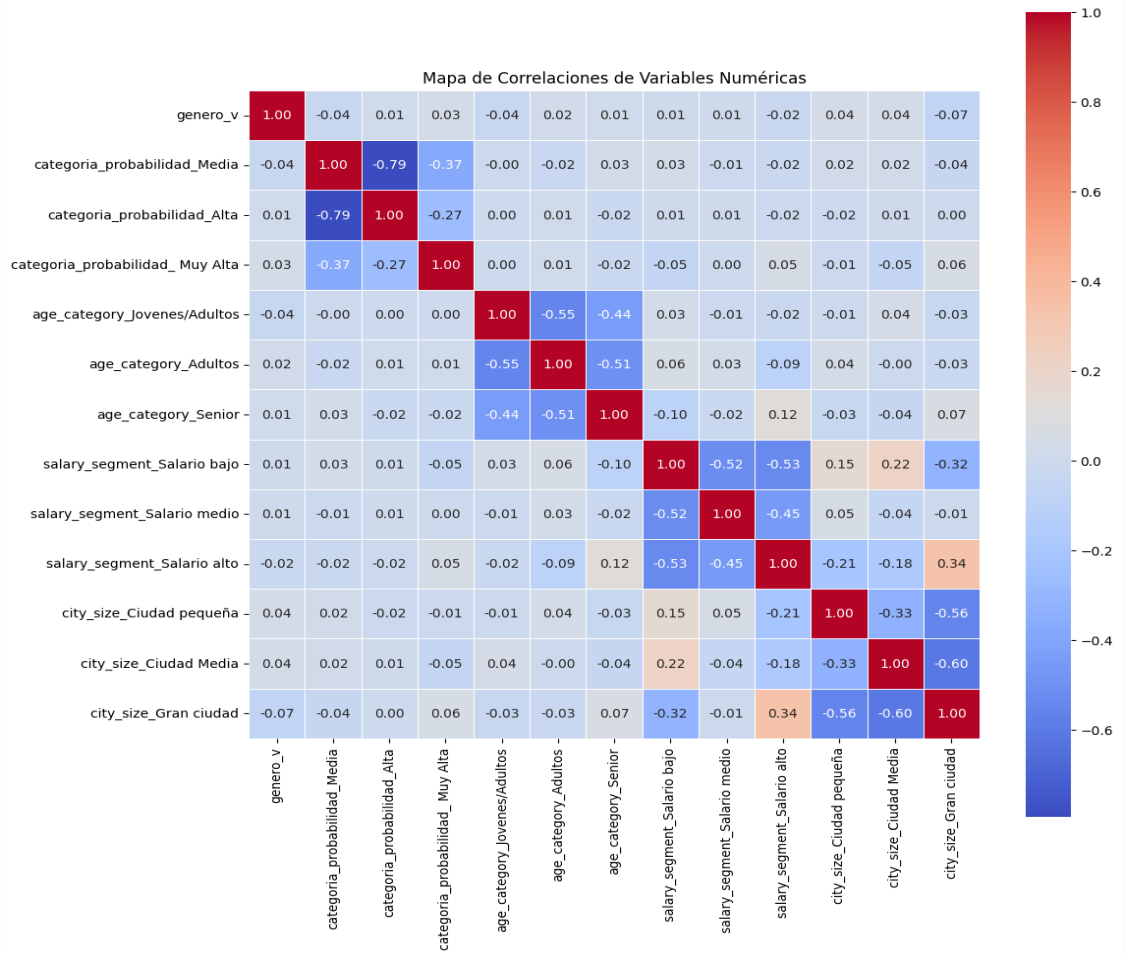
Se procedió a eliminar aquellas variables que no aportan valor para la segmentación, como es el caso de **mes_partition** y **entry_channel**.

Posteriormente, realizamos una limpieza y transformación de las columnas categóricas. Este paso es esencial, ya que el algoritmo de clustering K-Means que utilizaremos requiere datos numéricos. Por lo tanto, aplicamos **encoding** a varias columnas y realizamos ajustes para preparar los datos.

1. **Encoding de género**: Dado que la columna gender contiene valores de tipo texto ('V' para hombre y 'H' para mujer), transformamos estos valores en datos numéricos (1 para hombres y 0 para mujeres).

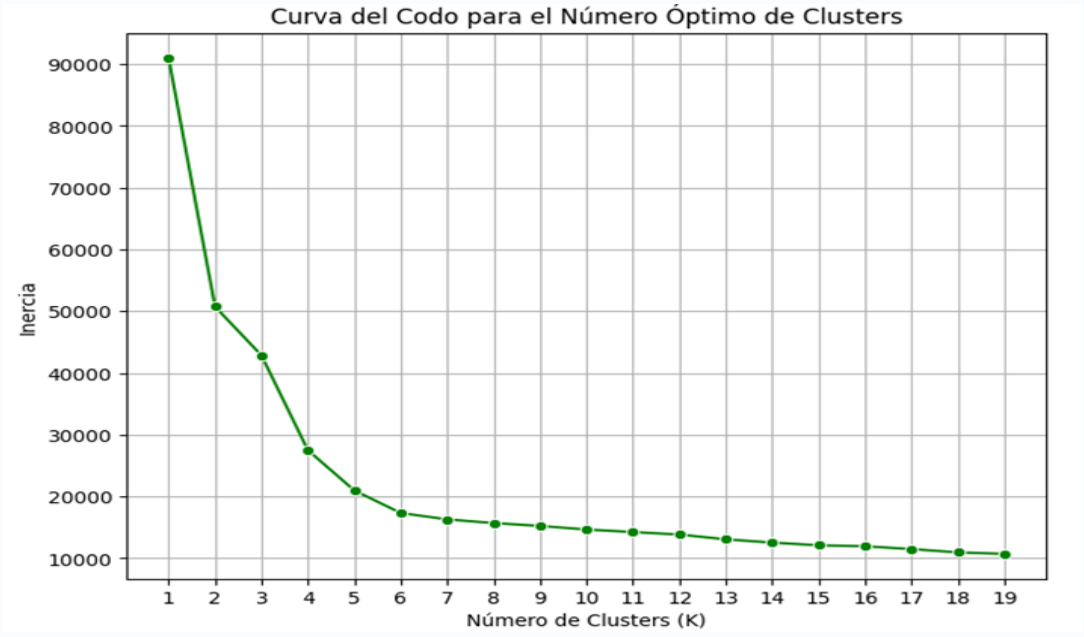
2. **Encoding de otras variables categóricas**: Además del género, otras variables categóricas como segment, categoria_probabilidad, age_category, salary_segment, y city_size fueron transformadas en variables dummy (codificación one-hot) para que el modelo de clustering pueda procesarlas.

3. **Matriz de Correlación**: Una vez transformados los datos, calculamos la matriz de correlación entre todas las variables numéricas. Este paso nos permite identificar relaciones lineales entre las variables, lo que es crucial para entender cómo interactúan y afectan los resultados del clustering.



El análisis del **mapa de correlación** nos ayuda a detectar si hay alguna redundancia entre variables, en este caso, podemos concluir que no existe ninguna correlación lo suficientemente fuerte entre las variables del modelo cómo para ser eliminadas.

4.3.1. Optimización del Número de Clusters: Curva del Codo



Una vez que los datos fueron escalados y preparados, aplicamos el algoritmo **K-Means** para realizar la segmentación. Sin embargo, antes de proceder con la segmentación final, es necesario determinar el número óptimo de clusters. Esto se logró utilizando la **Curva del Codo**.

Para este análisis, se observó que el número adecuado se encuentra entre **4 y 5 clusters**, ya que después de estos valores la inercia disminuye muy poco, lo que sugiere una segmentación eficiente sin sobreajuste.

4.3.2. Aplicación del Modelo K-Means

Una vez determinado el número óptimo de clusters, aplicamos el algoritmo **K-Means** con diferentes configuraciones (4, 5 y 6 clusters) para analizar cuál segmentación ofrecía los mejores resultados.

cluster_5	genero_v	categoria_probabilidad_Media	categoria_probabilidad_Alta	categoria_probabilidad_Muy Alta	age_category_Jovenes/Adultos	age_category_Adultos	age_category_Senior	salary_segment_Salario bajo
0	0.675971	0.498786	0.379045	0.122168	0.287217	0.480583	0.232201	1.000000
1	0.629147	0.802725	0.000000	0.197275	0.263033	0.000000	0.736967	0.114929
2	0.646729	0.000000	1.000000	0.000000	0.333084	0.371589	0.295327	0.166355
3	0.643085	0.864362	0.000000	0.135638	0.643085	0.202128	0.154787	0.231915
4	0.659144	0.822568	0.000000	0.177432	0.000000	1.000000	0.000000	0.176654

Las segmentaciones de 4, 5 y 6 clusters proporcionan diferentes distribuciones, pero tras analizar las características de cada uno, el modelo con **5 clusters** mostró un balance adecuado en términos de tamaño y diferenciación entre los segmentos. La opción de **5 clusters** fue seleccionada para la segmentación final.

4.4. Análisis de los Clusters

Posteriormente, calculamos las medias de todas las características para cada uno de los clusters obtenidos. Los **clusters resultantes** mostraron características demográficas diferenciadas, lo que permitirá la creación de campañas de marketing altamente personalizadas para cada uno de los grupos.

Clúster 1: "Urban Explorers"

- **Características principales**: Adultos y seniors, predominan los hombres, ingresos bajos, ubicados en ciudades pequeñas y medianas.
- **Probabilidad de compra**: Media (49.9%).
- **Mensaje de marketing**: Destacar beneficios locales y experiencias en su ciudad.

Clúster 2: "City High-Flyers"

- **Características principales**: Jóvenes adultos, mayoritariamente hombres, con ingresos altos, residentes en grandes ciudades.
- **Probabilidad de compra**: Media (80.3%).
- **Mensaje de marketing**: Enfocar en exclusividad y acceso VIP a servicios de lujo.

Clúster 3: "Balanced Lifestyles"

- **Características principales**: Mezcla de jóvenes adultos y seniors, ingresos medios, mayoría en grandes ciudades.
- **Probabilidad de compra**: Alta (100%).
- **Mensaje de marketing**: Resaltar la flexibilidad y balance entre vida personal y laboral.

Clúster 4: "Cultural Trendsetters"

- **Características principales**: Adultos, salarios medio a alto, ubicados en ciudades medianas.
- **Probabilidad de compra**: Muy alta (86.4%).
- **Mensaje de marketing**: Foco en experiencias culturales y tendencias locales.

Clúster 5: "Elite Professionals"

- **Características principales**: Adultos, ingresos altos, ubicados en grandes ciudades.
- **Probabilidad de compra**: Media (82.3%).
- **Mensaje de marketing**: Enfocar en exclusividad y beneficios premium, como viajes de lujo.

Tarea 5 – Seguimiento de Campaña

El éxito de una campaña de marketing digital depende de la capacidad de medir, interpretar y actuar sobre los datos de desempeño obtenidos. En el caso de EasyMoney, la campaña de marketing por correo electrónico dirigida a 10,000 clientes segmentados tiene como objetivo impulsar la adopción de un producto financiero clave: la tarjeta de crédito **EasyMoney**. Dado que la segmentación se ha realizado de manera cuidadosa, agrupando a los clientes en cinco clústeres distintos basados en características demográficas, socioeconómicas y de comportamiento, la medición del éxito de la campaña es fundamental para garantizar su efectividad.

Este plan de medición tiene el propósito de definir las métricas clave (KPIs) que permitirán evaluar de manera rigurosa la efectividad de la campaña. Los KPIs proporcionarán una visión cuantificable del rendimiento de la campaña, permitiendo ajustar las estrategias en función de los resultados obtenidos. Los indicadores seleccionados miden aspectos cruciales como la interacción con el contenido del correo, la conversión de clientes potenciales y el retorno económico, asegurando que la campaña no solo logre un impacto inmediato, sino que también fomente una relación a largo plazo con los clientes de EasyMoney.

5.1. KPIs (Key Performance Indicators)

5.1.1. Tasa de Apertura del Correo Electrónico (Open Rate)

La tasa de apertura mide el porcentaje de correos electrónicos que fueron abiertos en relación con el total de correos enviados. Es un indicador temprano de la efectividad del asunto y la relevancia del mensaje para el cliente.

$$Tasa\ de\ Apertura\ (\%) = \left(\frac{N\acute{u}mero\ Correos\ Abiertos}{Total\ Correos\ Enviados}\right) x\ 100$$

Este KPI es esencial para evaluar si los correos electrónicos están captando la atención de los clientes en los diferentes clústeres. Una tasa de apertura alta indicará que los mensajes son percibidos como relevantes y atractivos, lo cual es clave para inducir las siguientes interacciones.

5.1.2. Tasa de Clics (Click-Through Rate, CTR)

El CTR mide el porcentaje de clientes que hicieron clic en algún enlace dentro del correo electrónico en relación con el total de correos abiertos. Este indicador proporciona una visión clara sobre el nivel de interés generado por el contenido del correo y si el llamado a la acción (CTA) fue efectivo.

$$Tasa\ de\ Clics\ (CTR,\ \%) = \left(\frac{N\acute{u}mero\ de\ Clics}{Total\ Correos\ Abiertos}\right) x\ 100$$

El CTR es especialmente relevante cuando se trata de la venta de productos financieros, ya que el objetivo es motivar a los clientes a actuar, ya sea visitando la página de la tarjeta de crédito o iniciando el proceso de solicitud.

5.1.3. Tasa de Conversión (Conversion Rate)

La tasa de conversión mide el porcentaje de clientes que completaron la acción deseada, en este caso, solicitar la tarjeta de crédito, en relación con el total de correos abiertos. Este KPI es uno de los más críticos, ya que indica el éxito real de la campaña en términos de generación de ventas.

$$Tasa\ de\ Conversión = \left(\frac{Número\ de\ Conversiones}{Total\ Correos\ Abiertos}\right) x\ 100$$

Es importante medir esta métrica por separado para cada clúster, ya que la efectividad de los mensajes puede variar según las características demográficas y socioeconómicas de cada grupo.

5.1.4. Valor de Vida del Cliente (Customer Lifetime Value, CLV)

El CLV mide el valor económico total que un cliente aportará a la empresa a lo largo de su relación con la misma. Este KPI no solo se centra en la conversión inmediata, sino también en el impacto a largo plazo que tendrá la adquisición de nuevos clientes de tarjetas de crédito.

$$\text{CLV} = \left(\frac{\text{Ingreso Promedio por Cliente} \times \text{Margen Bruto}}{\text{Tasa de Churn}}\right)$$

El CLV ayuda a determinar si el costo de adquirir un cliente nuevo es justificable con respecto al valor total que ese cliente aportará. Es un KPI clave en una campaña de productos financieros, ya que se debe analizar el impacto a largo plazo más allá de la compra inicial.

* La tasa de Churn, también llamado tasa de cancelación, mide la rapidez con la que una empresa pierde clientes dentro de un plazo determinado. En el contexto de las aplicaciones móviles, representa el porcentaje de usuarios que han dejado de usar tu aplicación, ya sea que la desinstalen o cancelen su suscripción.

5.1.5. Retorno sobre la Inversión (ROI)

El ROI mide la rentabilidad de la campaña en términos del retorno generado por cada unidad monetaria invertida en la misma. En el caso de EasyMoney, es esencial conocer si la inversión en la campaña está generando un retorno positivo.

$$\text{ROI (\%)} = \left(\frac{\text{Ganancias de la Campaña} - \text{Coste de la Campaña}}{\text{Coste de la Campaña}}\right) \times 100$$

El ROI es un indicador clave para evaluar el éxito global de la campaña, especialmente cuando se utiliza para la venta de productos financieros, donde el costo de adquisición de clientes puede ser alto.

5.1.6. Retención de Clientes

La retención de clientes mide el porcentaje de clientes que, después de haber solicitado la tarjeta de crédito, siguen activos y utilizando los productos de EasyMoney a lo largo del tiempo. La retención es un KPI importante, ya que refleja la lealtad y el valor a largo plazo del cliente.

$$Tasa\ de\ Retención\ (\%) = \left(\frac{Clientes\ retenidos\ al\ Final\ del\ Periodo}{Clientes\ al\ Inicio\ del\ Periodo}\right) x\ 100$$

Este indicador es especialmente relevante para productos financieros como las tarjetas de crédito, donde el éxito no solo se mide por la adquisición, sino también por la utilización continua.

5.1.7. Tasa de Adopción del Producto

Este KPI mide la tasa a la cual los clientes que han recibido la tarjeta de crédito comienzan a utilizarla para realizar compras o acceder a los beneficios. Es crucial para medir no solo la conversión inicial, sino también el uso activo del producto.

$$Tasa\ de\ Adopción\ del\ Producto\ (\%) = \left(\frac{Número\ de\ Clientes\ Activos}{Total\ de\ Clientes\ Convertidos}\right) x\ 100$$

Este KPI es vital para asegurar que la campaña no solo genere solicitudes de tarjetas, sino que también promueva su uso continuo.

5.2 Estrategia de Medición Ampliada

La estrategia de medición automatizada debe permitir un seguimiento detallado de las interacciones con los correos electrónicos y sus efectos en los comportamientos de los clientes. La implementación de herramientas adecuadas para el análisis en tiempo real es fundamental para hacer ajustes oportunos y asegurar el máximo rendimiento de la campaña.

5.2.1 Pre-lanzamiento

Configuración de herramientas de análisis y segmentación:

- **Integración CRM y Email Marketing:** Es esencial que las plataformas de email marketing se sincronicen con un sistema de CRM (Customer Relationship Management) para permitir un seguimiento del comportamiento del cliente, como aperturas, clics y conversiones, en un solo sistema. Este paso garantiza que se pueda trazar un perfil detallado de cada clúster.

- **Definición de umbrales de éxito inicial:** Se deberán establecer valores umbral para cada KPI antes de iniciar la campaña. Por ejemplo, una tasa de apertura mínima esperada para ciertos clústeres o un CTR promedio que defina éxito o necesidad de ajuste.

- **Segmentación dinámica**: Basada en los clústeres predefinidos, el CRM debe estar configurado para ajustar automáticamente la segmentación si se detecta un cambio en las interacciones con el email, permitiendo la entrega de contenidos más personalizados en tiempo real.

5.2.2 Durante la campaña

Monitoreo diario de KPIs clave por clúster:

Visualización de datos en tiempo real: Es esencial utilizar paneles de control que permitan la visualización en tiempo real de KPIs como tasa de apertura, CTR, conversión y probabilidad de compra para cada clúster. Esto permitirá identificar tendencias emergentes.

Ajustes en tiempo real:

- A/B Testing continuo: Las pruebas A/B deben ejecutarse durante toda la campaña, modificando el asunto del correo, las imágenes o el contenido del cuerpo según la interacción de cada clúster. El sistema debe poder adaptar el contenido en base a los resultados de rendimiento observados.
- Personalización y segmentación: Si un clúster específico muestra una tasa de apertura baja o un CTR reducido, se puede ajustar en tiempo real el mensaje o las ofertas mostradas para mejorar el rendimiento.

Alertas automáticas:

Configuración de alertas automáticas cuando ciertos KPIs caen por debajo de los valores predeterminados o si alguno de los clústeres muestra un comportamiento anómalo, como una caída drástica en la tasa de conversión o un aumento inesperado de cancelaciones de suscripción.

5.2.3 Post-campaña

Evaluación exhaustiva de datos:

- Análisis por clúster: Se debe realizar un análisis detallado de los resultados por clúster, evaluando no solo el rendimiento en términos de KPIs tradicionales como CTR y tasa de conversión, sino también métricas más profundas como valor del ciclo de vida del cliente (CLV) y retención.
- Segmentación de ingresos y CLV: Es importante medir el retorno sobre la inversión (ROI) para cada clúster, calculando cuánto ha generado cada uno en términos de ingresos directos e indirectos. Además, se deben analizar las diferencias en el CLV por clúster para detectar cuáles podrían ser los grupos más rentables a largo plazo.

Evaluación del impacto a largo plazo:

Retención a 3 y 6 meses: Para evaluar el impacto duradero de la campaña, es clave analizar la retención de clientes a los 3 y 6 meses después de la campaña. ¿Se han mantenido más activos aquellos que recibieron la oferta? ¿Cómo varía la retención entre clústeres?

Lecciones aprendidas y optimización futura:

Optimización para campañas futuras: A partir de los resultados, se identificarán las estrategias más efectivas para cada clúster. Se propondrán mejoras para la próxima campaña, ajustando las estrategias de personalización, el uso de canales adicionales (como redes sociales o notificaciones push) y cualquier otra acción que mejore los resultados.

5.3. KPI Adaptados a los Clústeres

Clúster 0: "Urban Explorers"

- Enfoque en KPI: Tasa de apertura y conversión.
- Justificación: Este clúster tiene una probabilidad de compra media, y su poder adquisitivo es bajo. El objetivo será atraerlos con beneficios inmediatos, como descuentos locales.

Clúster 1: "City High-Flyers"

- Enfoque en KPI: Tasa de clics e ingreso promedio por cliente.
- Justificación: Dada la alta probabilidad de compra y los ingresos elevados, el foco será maximizar los ingresos generados por este segmento.

Clúster 2: "Balanced Lifestyles"

- Enfoque en KPI: Tasa de conversión y retención.
- Justificación: Este clúster tiene una alta probabilidad de compra, pero el objetivo a largo plazo será la retención, enfocándose en el valor a largo plazo.

Clúster 3: "Cultural Trendsetters"

- Enfoque en KPI: Tasa de apertura y retención.
- Justificación: Los clientes de este clúster son culturalmente sofisticados, y la campaña debe enfocarse en mantener su interés con beneficios que aseguren su lealtad.

Clúster 4: "Elite Professionals"

- Enfoque en KPI: ROI y CLV.
- Justificación: Debido a su perfil de altos ingresos y gran probabilidad de compra, el objetivo será maximizar el valor a largo plazo de este segmento.

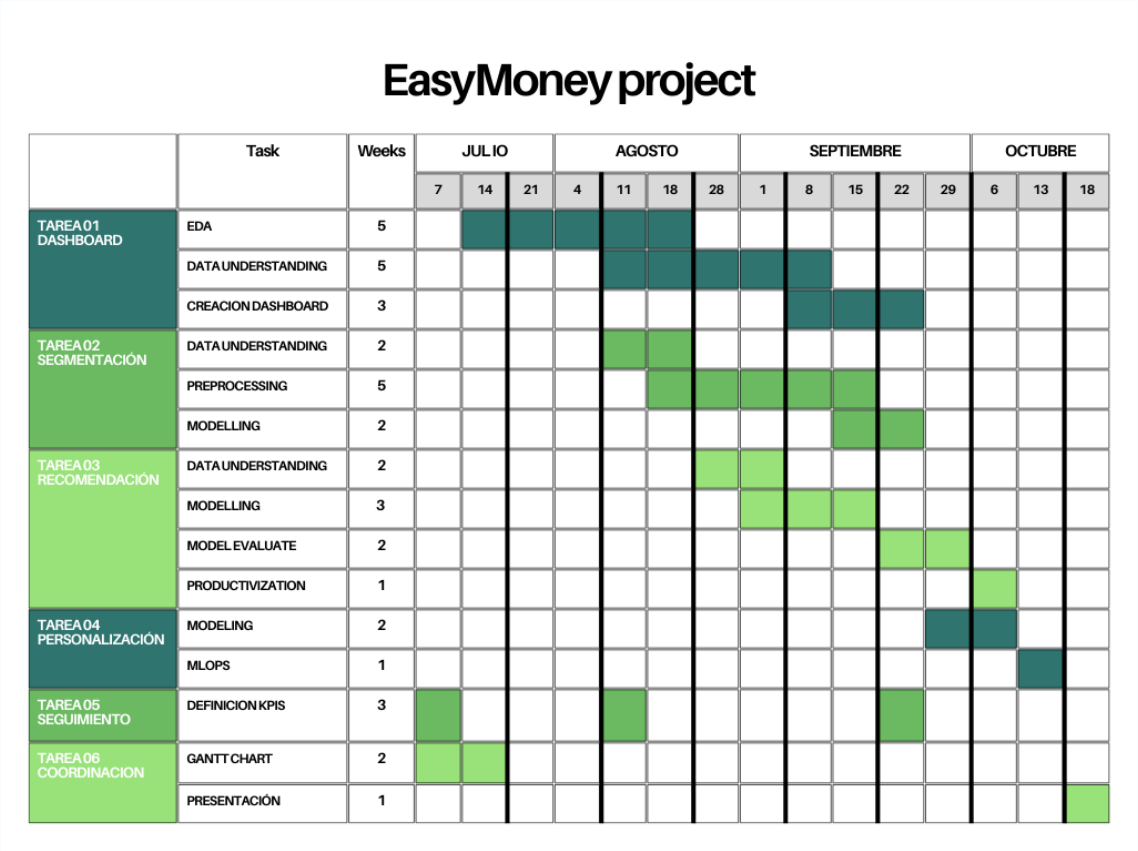
Tarea 6 - Coordinación de proyecto EASYMONEY

6.1. Introducción

El objetivo principal de esta tarea es establecer un proceso robusto de coordinación y validación de los cambios y mejoras que se implementen tanto en los modelos operativos como en el dashboard. Es crucial que exista un proceso de validación para los informes y análisis, así como un plan de proyecto claro sobre la gestión y coordinación de las tareas asignadas. Esto garantiza la calidad y la alineación de los resultados con las expectativas de las partes interesadas.

6.2. Diagrama de Gantt

El diagrama de Gantt presentado ilustra el cronograma de las tareas del proyecto desde julio hasta octubre. A continuación, se describen las tareas y sus respectivas fechas de inicio y duración:



6.2.1. Descripción de las Tareas

TAREA 01: DASHBOARD

- **Objetivo:** Realizar un análisis exploratorio de los datos (EDA) y desarrollar un dashboard para la visualización de todos los departamentos.
- **Duración:** 5 semanas.

- **Actividades Clave:** EDA y creación del dashboard.

TAREA 02: SEGMENTACIÓN

- **Objetivo:** Realizar un entendimiento de los datos y aplicar técnicas de segmentación, para generar una segmentación general de los clientes.
- **Duración:** 5 semanas.
- **Actividades Clave:** Preprocesamiento y modelado.

TAREA 03: RECOMENDACIÓN

- **Objetivo:** Desarrollar un modelo recomendación en este caso de nuestro producto credit card para una campaña de 10.000 clientes por email.
- **Duración:** 3 semanas.
- **Actividades Clave:** Entendimiento de datos y modelado.

TAREA 04: PERSONALIZACIÓN

- **Objetivo:** Segmentar esos 10.000 clientes para generar una personalización de las creatividades de la campaña por mail.
- **Duración:** 3 semanas.
- **Actividades Clave:** Segmentacion y MLOps.

TAREA 05: SEGUIMIENTO

- **Objetivo:** Definir KPIs y realizar un seguimiento de los modelos implementados.
- **Duración:** 3 semanas.
- **Actividades Clave:** Definición de KPIs.

TAREA 06: COORDINACIÓN

- **Objetivo:** Asegurar la correcta comunicación y coordinación entre los miembros del equipo.
- **Duración:** 2 semanas.
- **Actividades Clave:** Gantt chart y presentación final (22 de Octubre).

6.3. Gestión del Código con Git

La gestión del código es un aspecto esencial para el éxito del proyecto. Para ello, se utilizará Git, específicamente GitHub, como plataforma de control de versiones.

Ventajas de Git y GitHub

- **Control de Versiones:** Permite a los desarrolladores mantener un registro detallado de los cambios realizados en el código, facilitando el seguimiento y la recuperación de versiones anteriores si es necesario.

- **Colaboración Efectiva:** Múltiples desarrolladores pueden trabajar en diferentes características simultáneamente sin interferir en el trabajo de los demás gracias al uso de ramas.
- **Proceso de Aprobación:** A través de pull requests, los cambios pueden ser revisados y aprobados por un manager antes de ser fusionados a la rama principal, asegurando la estabilidad del código.

Flujo de Trabajo

- **Rama Principal:** La rama master contendrá el código que se lleva a producción.
- **Rama de Desarrollo:** La rama develop se utilizará para introducir cambios nuevos y características en desarrollo.
- **Revisiones:** Antes de realizar un merge a master, se debe realizar una revisión exhaustiva del código, y este debe ser aprobado por el manager del proyecto.

Documentación del Código

La documentación es vital para garantizar que el código sea comprensible y fácil de mantener. Deberá abarcar tanto una perspectiva técnica como una no técnica.

Recomendaciones de Documentación

- **Descripción Funcional:** Cada función debe tener una breve descripción de su propósito, los parámetros de entrada y salida, y la última revisión realizada.
- **Código Limpio:** Es importante seguir principios de **Clean Code**, asegurando que el código sea fácil de entender, mantener y escalar. Las funciones deben ser cortas y realizar una única acción.
- **Documentación Técnica y General:** Se debe crear un manual técnico para desarrolladores y otro más accesible para el personal no técnico. Esto facilita la incorporación de nuevos miembros al equipo y asegura que otros departamentos comprendan el funcionamiento del modelo.

6.3.1 Creación de un Paquete

Si el proyecto crece en complejidad, se recomienda construir un paquete propio en Python o R que contenga todas las funciones y estructuras utilizadas, lo que facilitará su reutilización y gestión.

6.4. Metodología de Trabajo

Se optará por la metodología **Agile**, específicamente por **Scrum**, que ha demostrado ser efectiva en proyectos de desarrollo de software.

Beneficios de Scrum

- **Iteraciones Rápidas:** Los proyectos se dividen en sprints de dos semanas, permitiendo ajustes y mejoras constantes.

- **Involucramiento del Cliente:** Los clientes y partes interesadas tienen la oportunidad de dar feedback continuo, lo que mejora la alineación del producto final con sus expectativas.

Estructura de Scrum

1. **Backlog:** Todas las tareas se recopilan en un backlog, que se prioriza y se detalla con historias que describen los objetivos de cada tarea.
2. **Planificación de Sprints:** Al inicio de cada sprint, el equipo selecciona las tareas del backlog que se abordarán, asignándolas a los desarrolladores
3. **Scrum Board:** Se utilizará un scrum board para visualizar el progreso de las tareas, con columnas para To Do, Doing, Review, y Done.
4. **Reuniones Diarias:** Se llevarán a cabo reuniones diarias de Scrum para revisar el avance y los bloqueos