COMPETENCIA ESTRATÉGICA EN STREAMING CON ESTIMACIÓN DE PAGOS Y EQUILIBRIOS DE NASH

Ing. Ivan Mamani Condori
mamaniconivan@gmail.com – JalaUniversity.
Gimena Javier Vargas
vargasgimena22@gmail.com – U.A.T.F.
Christian Giovanni Ortiz
ortizchristiangiovanni13@gmail.com – Banco FIE

RESUMEN

El estudio, enmarcado en la teoría de juegos y la economía experimental, analiza la competencia estratégica en servicios OTT (Over-The-Top, video por internet) entre Netflix (2 estrategias), Disney+ (3) y Max (2) mediante un juego estático 2×3×2 en forma normal. Con un panel trimestral 2020–2024, estimado por OLS el efecto de las decisiones propias y rivales; contenido original, política de precios y bundle sobre altas netas y ARPU (Average Revenue Per User). A partir de dichas predicciones se construyen los pagos (beneficios) por perfil estratégico, se identifican mejores respuestas, equilibrios de Nash (puros y débiles) y el frente de Pareto, y se ejecutan contrafactuales reproducibles. El análisis se complementa con gráficos comparativos que documentan la evolución de suscriptores y el posicionamiento relativo de las plataformas. Asimismo, se realizan ejercicios de sensibilidad a costos fijos/variables y a la especificación empírica, mostrando cuándo conviene diferenciarse (originales/bundle) frente a competir en precio. El trabajo aporta una plantilla replicable, propia de la economía experimental para mapear datos a utilidades y evaluar la interacción estratégica en mercados OTT.

PALABRAS CLAVE

Teoría de juegos; Economía experimental; OTT (streaming); Equilibrio de Nash; Competencia estratégica.

ABSTRACT

The study, grounded in game theory and experimental economics, analyzes strategic competition in OTT services (Over-The-Top, internet-delivered video) among Netflix (2 strategies), Disney+ (3), and Max (2) using a static 2×3×2 normal-form game. Using a quarterly panel for 2020–2024, the impact of own and rival decisions, original content, pricing policy, and bundling on net additions and ARPU (Average Revenue Per User) is estimated via OLS. Based on these predictions, payoffs (profits) are constructed for each strategic profile; best responses, Nash equilibria (pure and weak), and the Pareto frontier are identified; and reproducible counterfactuals are executed. The analysis is complemented with comparative graphics that document the evolution of subscribers and the platforms' relative positioning. Sensitivity exercises to fixed/variable costs and to the empirical specification are also conducted, showing when differentiation (originals/bundling) is preferable to price competition. The study provides a replicable template, consistent with experimental economics practice for mapping data to utilities and evaluating strategic interaction in OTT markets.

KEYWORDS

Game theory; Experimental economics; OTT (streaming); Nash equilibrium; Strategic competition.

1. INTRODUCCIÓN

En la actualidad, la industria del entretenimiento digital ha experimentado un crecimiento acelerado, impulsado por la expansión de plataformas de streaming que ofrecen contenido audiovisual bajo demanda. Esta competencia estratégica ha generado un entorno altamente dinámico, donde las decisiones de cada plataforma afectan directamente la participación de mercado, los ingresos y la preferencia de los consumidores.

El presente estudio se centra en el análisis de la competencia entre plataformas de streaming mediante un enfoque de teoría de juegos, utilizando un diseño de tipo 2×3×2, que permite modelar de manera detallada las estrategias disponibles para los agentes, los posibles escenarios de interacción y los resultados esperados en términos de pagos o utilidades. Este enfoque cuantitativo permite no solo identificar los equilibrios de Nash de la interacción estratégica, sino también comprender cómo las decisiones simultáneas de los competidores pueden influir en la rentabilidad y sostenibilidad del negocio en un mercado altamente competitivo.

Para este análisis, se seleccionan tres empresas líderes en el sector:

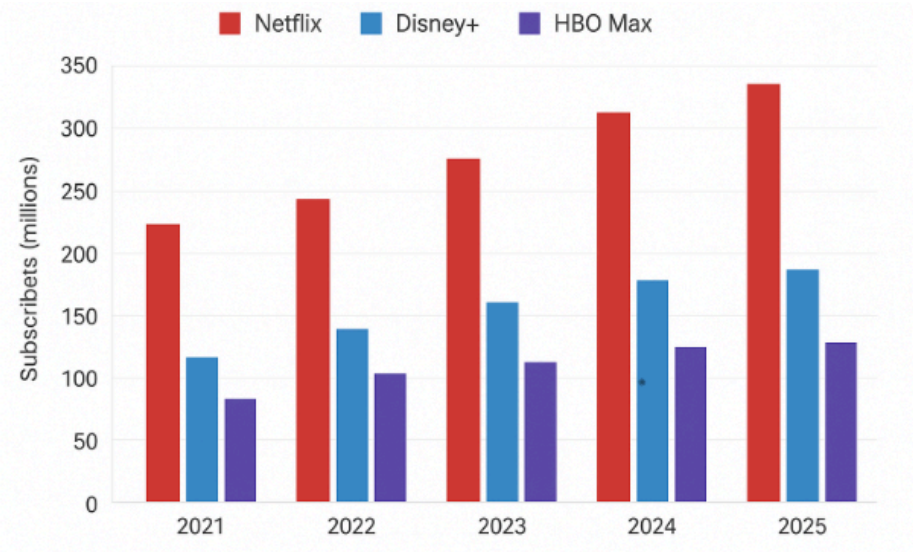
- Netflix: Considerada pionera en la industria del streaming, Netflix ha logrado consolidarse como la plataforma con mayor número de suscriptores a nivel mundial. Su estrategia se basa en la producción de contenido original, la adquisición de licencias exclusivas y la expansión internacional, buscando mantener una ventaja competitiva sostenida frente a sus rivales.
- HBO Max: Parte del ecosistema de Amazon, esta plataforma combina el servicio de streaming con beneficios adicionales para los suscriptores Prime, como envíos gratuitos y acceso a otros servicios de Amazon. Su modelo se enfoca en la diversificación de contenido, incluyendo películas, series, documentales y deportes, con el objetivo de incrementar la retención de usuarios y atraer nuevos segmentos de mercado.
- Disney+: Aprovechando el amplio portafolio de marcas icónicas como Disney, Pixar, Marvel y Star Wars, Disney+ ha experimentado un crecimiento rápido

desde su lanzamiento. Su estrategia se centra en la exclusividad de contenidos familiares y franquicias consolidadas, lo que le permite diferenciarse en un mercado competitivo y captar audiencias específicas a nivel global.

El mercado global de streaming ha mostrado un crecimiento sostenido durante los últimos cinco años, con un incremento en la cantidad de suscriptores y en la diversidad de contenidos ofrecidos. Plataformas como Netflix, Disney+ y HBO Max han implementado estrategias diferenciadas para captar y retener audiencias, compitiendo no solo por la cantidad de suscriptores, sino también por la exclusividad y calidad del contenido.

Gráficos comparativos del número de suscriptores y cuotas de mercado muestran la evolución de estas plataformas en los últimos años, evidenciando patrones de crecimiento y fluctuaciones que reflejan las decisiones estratégicas adoptadas por cada empresa. tal como se muestra en las siguiente tabla.

Año	Netflix	Disney+	HBO Max
2021	207M	116M	67M
2022	222M	129M	73M
2023	231M	142M	87M
2024	301M	157M	97M
2025	310M	165M	102M



El objetivo principal de este trabajo es estimar los pagos asociados a cada combinación de estrategias y determinar los equilibrios de Nash, proporcionando así un marco analítico que puede orientar la toma de decisiones estratégicas en la industria del streaming. Asimismo, se busca demostrar cómo la aplicación de modelos de teoría de juegos contribuye a la comprensión de la competencia en mercados digitales, ofreciendo herramientas predictivas y de planificación estratégica tanto para académicos como para profesionales del sector.

2. MARCO TEÓRICO

2.1. Competencia Estratégica y Teoría de Juegos

La competencia estratégica es el análisis de las decisiones de las empresas considerando las posibles respuestas de sus competidores. En el contexto de plataformas de streaming, estas decisiones incluyen precios de suscripción, inversión en contenido y estrategias de marketing, las cuales dependen de la interacción con otras plataformas. Como señalan Dixit y Nalebuff (2008), la comprensión de la estrategia competitiva requiere anticipar las acciones y reacciones de los rivales, mientras que Tirole (1988) destaca la importancia de estas interacciones en mercados oligopólicos, donde pocas empresas compiten por una base de consumidores limitada. En este estudio, la competencia estratégica se aborda mediante un modelo de teoría de juegos que permite analizar cómo las plataformas ajustan sus estrategias de manera interdependiente.

2.2. Mercados de Streaming y Diferenciación de Contenido

Las plataformas de streaming, como Netflix, Disney+ y Amazon Prime Video, operan en mercados caracterizados por la diferenciación de contenido y la importancia de los efectos de red. Los estudios de Smith, Johnson y Lee (2022) y Hagiu y Wright (2020) evidencian que las decisiones de precios y contenido influyen directamente en la captación de suscriptores y en la cuota de mercado. En este contexto, la competencia no solo se basa en el precio, sino también en la exclusividad y calidad del contenido ofrecido, así como en la capacidad de retener usuarios frente a alternativas disponibles.

2.3. Juegos Estratégicos y Diseño Experimental 2×3×2

Para analizar la interacción estratégica entre plataformas de streaming, este estudio emplea un diseño experimental 2×3×2: tres empresas, dos estrategias posibles para la primera empresa, tres estrategias para la segunda empresa y dos para la tercera empresa. Este enfoque permite observar cómo cambian las utilidades esperadas de cada plataforma según las estrategias adoptadas por su competidor y las condiciones del mercado. Osborne y Rubinstein (1994) y Myerson (1991) destacan que los diseños experimentales en teoría de juegos facilitan la estimación de pagos y la identificación de equilibrios estratégicos bajo diferentes supuestos.

2.4. Equilibrio de Nash y Estimación de Pagos

El equilibrio de Nash se alcanza cuando ninguna empresa puede mejorar su resultado cambiando unilateralmente su estrategia, dado lo que hace el competidor (Fudenberg & Tirole, 1991). Este concepto permite identificar combinaciones de precios o inversión en contenido en las que las plataformas no tienen incentivos para modificar sus decisiones. La estimación de los pagos, es decir, de las utilidades esperadas de cada estrategia, se realiza mediante modelos econométricos y análisis de mercado que consideran suscriptores, cuota de mercado y elasticidad de la demanda (Mas-Colell, Whinston & Green, 1995; Varian, 2014).

2.5. Aplicaciones Empíricas

Estudios recientes muestran cómo la teoría de juegos se aplica a la competencia entre plataformas de streaming. Chen y Zhang (2021) analizan el comportamiento del consumidor frente a estrategias de precios y contenido, mientras que Borges y Ferreira (2020) investigan la retención de suscriptores mediante juegos de pricing en servicios digitales. Estos estudios respaldan la relevancia del modelo 2×3×2 para estimar pagos y analizar equilibrios de Nash, ofreciendo un marco teórico sólido para evaluar la competencia estratégica en el mercado de streaming.

3. CONTRIBUCIONES METODOLÓGICAS

Se presenta un procedimiento reproducible para mapear datos a utilidades en un juego 2×3×2: (i) estimación OLS con efectos fijos y errores robustos de altas netas y ARPU sobre un panel trimestral; (ii) construcción de una función de pagos que integra predicciones y costos (variables y fijos) para simular contrafactuales en los 12 perfiles; y (iii) generación automática de matrices de pagos, mejores respuestas, equilibrios de Nash y frente de Pareto, con pruebas de sensibilidad.

3.1. Modelo del Juego

El modelo presentado constituye un juego estático de tres jugadores con información completa. Los jugadores son Netflix, Disney+ y HBO Max, cada jugador debe seleccionar simultáneamente una estrategia dentro de un conjunto de acciones disponibles, con el objetivo de maximizar su utilidad, misma que depende de las elecciones de los otros jugadores.

Las estrategias que se tienen son 3:

- O: Invertir en contenido original, es decir, incrementar la producción propia siendo contenido exclusivo de cada plataforma.
- P: Competir con precios bajos, impulsando las ventas mediante descuentos y promociones.
- B: Bundle, incremento de las franquicias ofertando varios productos como un solo paquete, estrategia exclusiva de Disney+.

Con esta información se tendría la matriz de 2x3x2, en la misma está expresado todos las utilidades que estas empresas generarían según la estrategia que adopte cada jugador. Esta matriz quedaría de esta manera, después de realizar los cálculos se tendrá una matriz con las utilidades respectivas.

HBO MAX: O

NETFLIX \ DISNEY	O	P	B
O	O,O,O	O,P,O	O,B,O
P	P,O,O	P,P,O	P,B,O

HBO MAX: P

NETFLIX \ DISNEY	O	P	B
O	O,O,P	O,P,P	O,B,P
P	P,O,P	P,P,P	P,B,P

Tabla 1: Matriz de pagos inicial, elaboración propia.

Estructura del Juego

Para poder tener estos datos de la matriz, se tiene a continuación la explicación matemática de este modelo donde se considera un juego estático de información completa $G = (N, \{A_i\}, \{\Pi_i\})\ donde\ i \epsilon N$ en forma normal. El conjunto de jugadores es:

$$N = \{Netflix,\ Disney,\ Max\}$$

Los conjuntos de estrategias puras son:

$$A_{Netflix} = \{O,\ P\},\ A_{Disney} = \{O,\ P,\ B\}\ y\ A_{Max} = \{O,\ P\}$$

Un perfil es:

$$s = (a_{Netflix},\ a_{Disney},\ a_{Max}) \epsilon A_{Netflix} \cdot A_{Disney} \cdot A_{Max}$$

Cada perfil induce un vector de pagos:

$$\Pi(s) = (\Pi_{Netflix}(s),\ \Pi_{Disney}(s),\ \Pi_{Max}(s))$$

La representación normal se organiza como un ***tensor 2×3×2***; para su lectura se emplean "rebanadas" fijando la estrategia de Max y mostrando matrices 2×3 (filas: Netflix; columnas: Disney).

Información y Timing

El juego es simultáneo y de información completa: cada jugador conoce los espacios de estrategias y la función de pagos, y elige sin observar ex ante la acción efectiva de los rivales.

Función de Pagos (Mapeo datos→utilidades).

Para cada jugador i y perfil s, el pago trimestral se modela contablemente como:

$$\Pi_i(s) = (S_i^0 + \widehat{net\_adds}_i(s))\cdot(\widehat{ARPU}_i(s) - c_v)\cdot 3 - c_i^{fix}(a_i(s))$$

donde S_i^0 es un stock de referencia, $\widehat{net\_adds}_i(s)$ y $\widehat{ARPU}_i(s)$, son predicciones obtenidas de las ecuaciones empíricas, c_v es el costo variable mensual por usuario y $c_i^{fix}(\cdot)$ el costo fijo trimestral de la estrategia elegida. Este mapeo de datos a utilidades mantiene separadas las dimensiones de **crecimiento** y **monetización**, tal como recomienda la práctica estándar en juegos aplicados y microeconomía empírica.

Mejor Respuesta y Equilibrio de Nash

Para cada i∈N, la correspondencia de **mejor respuesta** es:

$$BR_i(s_{-i}) = argmax\Pi_i(a, s_{-i})$$

Donde s_{-i} denota las estrategias de los rivales. Un **equilibrio de Nash (puro)** es un perfil s^* tal que $s^* \in BR_i(s^*_{-i})$, para todo i. En juegos finitos siempre **existe al menos un equilibrio en estrategias mixtas** (Teorema de Nash). En este trabajo se reportan equilibrios puros (y débiles cuando hay empates), y se discuten mixtos cuando corresponde.

Eficiencia de Pareto

Un perfil *s* es **Pareto-eficiente** si no existe *s′* tal que $\Pi_j(s') >= \Pi_j(s)$ para todo j y $\Pi_k(s') > \Pi_k(s)$ para algún k. El **frente de Pareto** se obtiene identificando los perfiles no dominados dentro del tensor de pagos.

Supuestos y Diseño (vínculo con Economía Experimental)

El juego es **estático** y de **información completa**; los pagos $\Pi_i(s)$ se tratan como realizaciones observables (o simuladas) por perfil, análogo a un diseño factorial **2×3×2** con tratamientos bien definidos por jugador. La notación y los conceptos de forma normal, mejor respuesta y equilibrio siguen los textos canónicos de teoría de juegos.

Ilustración con las Estimaciones Base

```
=== Max: O ===
               | Disney:O              | Disney:P              | Disney:B
-------------------------------------------------------------------------------
Netflix:O      | (4191.5, 727.8, 831.9) | (4084.5, 674.3, 802.4) | (4195.0, 764.9, 833.8)
Netflix:P      | (3687.2, 689.6, 802.4) | (3580.3, 636.3, 772.8) | (3690.3, 726.7, 804.2)

=== Max: P ===
               | Disney:O              | Disney:P              | Disney:B
-------------------------------------------------------------------------------
Netflix:O      | (4084.5, 689.6, 736.8) | (3977.4, 636.3, 707.4) | (4087.9, 726.7, 738.4)
Netflix:P      | (3580.3, 651.4, 707.4) | (3473.4, 598.2, 678.0) | (3583.4, 688.4, 709.0)
```

Bajo los parámetros y estimaciones de referencia, las rebanadas reportadas muestran que:

a) La mejor respuesta de Netflix, dado (D, M), es **O**.
b) La de Disney, dado (N, M), es **B**.
c) La de Max, dado (N, D), es **O**.

Por tanto, $(N = O, D = B, M = O)$, constituye un **equilibrio de Nash estricto** y, en el conjunto de 12 perfiles evaluados, resulta además **Pareto-eficiente** (véase la sección para la tabla de pagos y los valores numéricos).

3.2. Dataset: Preparación e Integración

Se utilizaron los sientes datasets disponibles en la red:

- **Netflix: "Netflix OTT Revenue and Subscribers (CSV File)" (Kaggle).**
  Tabla con la serie histórica de ingresos y número de suscriptores de pago de Netflix (nivel agregado), a partir de 2012. Se utiliza como base para derivar métricas de monetización (ARPU aproximado) y escala de usuarios en el periodo de interés.
- **Disney: "Walt Disney OTT Platforms Revenue and Subscribers" (Kaggle).**
  Compendio con ingresos y suscriptores de las plataformas OTT de Disney (Disney+, Disney+ Hotstar, Hulu y ESPN+). Para el proyecto, se toma como proxy del negocio DTC la serie de Disney+ (y, cuando corresponde, la agregación DTC reportada en la fuente) para alinear la comparación con Netflix y Max.
- **Max: Investor Relations de Warner Bros. Discovery (WBD).**
  Sitio oficial con resultados trimestrales, comunicados y presentaciones que incluyen métricas de suscriptores y resultados del segmento DTC (HBO Max/Max). Se emplea para obtener los puntos trimestrales de referencia en 2020–2024.

Dataset Integrado del Proyecto.

A partir de estas fuentes se elaboró un **panel trimestral armonizado 2020Q1–2024Q4** (formato `YYYYQn`) con las variables: `subs_start_millions`, `net_adds_millions`, `subs_end_millions`, `arpu`, e indicadores de **estrategia** (`flag_O`, `flag_P`, `flag_B`) y **presión rival** (`riv_flag_O`, `riv_flag_P`, `riv_flag_B`). Cuando alguna serie pública estuvo anual o no alineada con el trimestre calendario, se aplicaron transformaciones estándar (interpolación/redistribución conservadora y chequeos de consistencia) para obtener un panel homogéneo. Este panel integrado es el que alimenta las estimaciones OLS y la **matriz de pagos 2×3×2** utilizadas en el análisis.

3.3. Cálculo de Utilidades

Esta sección describe el **pipeline datos → predicciones → pagos** que permite llenar el tensor 2×3×2 con beneficios trimestrales $\Pi_i(s)$ para cada plataforma $i \in \{Netflix,\ Disney,\ max\}$ y cada perfil estratégico ***s***.

3.3.1. Especificación Econométrica (OLS con FE)

Se estiman dos ecuaciones lineales: una para **altas netas** y otra para **ARPU**, que relacionan decisiones propias y presión rival con resultados operativos:

$$net\_adds_{it} = \alpha_0 + \alpha_1 O_{it} + \alpha_2 P_{it} + \alpha_3 B_{it} + \rho_1 R^O_{it} + \rho_2 R^P_{it} + \rho_3 R^B_{it} + \gamma C(platform_i) + \varepsilon_{it}$$

$$ARPU_{it} = \beta_0 + \beta_1 O_{it} + \beta_2 P_{it} + \beta_3 B_{it} + \eta_1 R^O_{it} + \eta_2 R^P_{it} + \eta_3 R^B_{it} + \delta C(platform_i) + u_{it}$$

Donde:

- O_{it}, P_{it}, B_{it}, son **dummies de estrategia propia** (con B solo para Disney).
- R^O_{it}, R^P_{it}, R^B_{it}, son **conteos de rivales** que activan cada estrategia en *t*.
- $C(platform_i)$, introduce **efectos fijos** por plataforma (baseline: **Netflix**).
- La estimación se realiza por **OLS**, reportando **errores robustos** (HC3 o *cluster* por plataforma).

Notas de Implementación:

a) `platform` se trata como categórica con `Treatment("Netflix")`.
b) Se tipan columnas numéricas y se recalculan los conteos rivales para cada periodo.
c) Se aplican recortes suaves a outliers en `arpu`/`net_adds_millions` si es necesario para estabilidad numérica.

3.3.2. Predicción Contrafactual y Función de Pagos

Para **cada perfil estratégico** $s = (a_N, a_D, a_M)$ se construye, jugador por jugador, la fila de regresores $X_i(s)$ con:

a) dummies propias O, P, B.

b) **conteos rivales** R^O, R^P, R^B**.**

c) la plataforma correspondiente en la misma codificación categórica usada en el entrenamiento. Con los modelos OLS se obtienen:

$$\widehat{net\_adds}_i(s),\ \widehat{ARPU}_i(s)$$

Luego se mapea a **beneficio trimestral** mediante una identidad contable:

$$\Pi_i(s) = \underbrace{\left(S_i^0 + \widehat{\text{net\_adds}}_i(s)\right)}_{\text{stock fin}} \cdot \underbrace{\left(\widehat{\text{ARPU}}_i(s) - c_v\right)}_{\text{contribución mensual por usuario}} \cdot 3 - \underbrace{c_i^{\text{fix}}(a_i(s))}_{\text{costo fijo por estrategia}},$$

donde:

- S_i^0 es el **stock de referencia** (promedio histórico de `subs_start_millions` de *i*).
- c_v es el **costo variable mensual** por usuario (parámetro expuesto, **2 USD** por defecto).
- c_i^{fix} es el **costo fijo trimestral** asociado a la estrategia ***a*** (diccionario parametrizable; p. ej., $O > P, y\ B$ solo para Disney).

3.3.3. Enumeración del juego 2×3×2 y Artefactos

Se generan los **12 perfiles** combinando $A_N = \{O, P\}$, $A_D = \{O, P, B\}$, $A_M = \{O, P\}$. Para cada perfil se calculan $\Pi_N(s)$, $\Pi_D(s)$, $\Pi_M(s)$ y se arma la **tabla de pagos**. Con esa tabla se:

a. **Identifican mejores respuestas** por jugador (máximos por columnas/filas dentro de cada rebanada).
b. **Detectan equilibrios de Nash puros** (intersección de mejores respuestas) y, cuando corresponda, **Nash débiles** (empates).
c. **Computa el frente de Pareto** (perfiles no dominados).

3.3.4. Matriz de Utilidades 2×3×2

El procedimiento descrito en las secciones anteriores, culmina en la **matriz de pagos** $\Pi(s)$ del juego 2×3×2, donde cada celda corresponde a un perfil $s = (a_N, a_D, a_M)$ y reporta el vector $(\Pi_{Netflix}(s), \Pi_{Disney}(s), \Pi_{Max}(s))$ en **millones USD por trimestre**.

Para facilitar la trazabilidad y la replicación, el notebook:

- **Enumera** los 12 perfiles $(N \in \{O, P\}, D \in \{O, P, B\}, M \in \{O, P\})$.
- **Predice** $\widehat{net\_adds}_i(s), y\ \widehat{ARPU}_i(s)$ con los modelos OLS.
- **Aplica** la función contable de pagos $\Pi_i(s)$ con los parámetros de costos $(c_v, c_i^{fix}(a))$.

```
=== Pagos (millones) por perfil 2×3×2 ===
```

	profile	Netflix_profit_m	Disney_profit_m	Max_profit_m
0	{'Netflix': 'O', 'Disney': 'O', 'Max': 'O'}	4191.529452	727.778294	831.901556
1	{'Netflix': 'O', 'Disney': 'O', 'Max': 'P'}	4084.520784	689.614826	736.801968
2	{'Netflix': 'O', 'Disney': 'P', 'Max': 'O'}	4084.520784	674.321017	802.408497
3	{'Netflix': 'O', 'Disney': 'P', 'Max': 'P'}	3977.439587	636.291027	707.442385
4	{'Netflix': 'O', 'Disney': 'B', 'Max': 'O'}	4194.959411	764.852429	833.765782
5	{'Netflix': 'O', 'Disney': 'B', 'Max': 'P'}	4087.913219	726.662962	738.381987
6	{'Netflix': 'P', 'Disney': 'O', 'Max': 'O'}	3687.180980	689.614826	802.408497
7	{'Netflix': 'P', 'Disney': 'O', 'Max': 'P'}	3580.305789	651.378829	707.442385
8	{'Netflix': 'P', 'Disney': 'P', 'Max': 'O'}	3580.305789	636.291027	772.842907
9	{'Netflix': 'P', 'Disney': 'P', 'Max': 'P'}	3473.358069	598.188506	678.010272
10	{'Netflix': 'P', 'Disney': 'B', 'Max': 'O'}	3690.326733	726.662962	804.235198
11	{'Netflix': 'P', 'Disney': 'B', 'Max': 'P'}	3583.414018	688.400966	708.984881

Para su visualización, el notebook genera **rebanadas** fijando la acción de Max (panel **Max = O** y panel **Max = P**), en formato 2×3 (filas: Netflix; columnas: Disney).

```
=== Max: O ===
               | Disney:O              | Disney:P              | Disney:B
-------------------------------------------------------------------------------------
Netflix:O      | (4191.5, 727.8, 831.9) | (4084.5, 674.3, 802.4) | (4195.0, 764.9, 833.8)
Netflix:P      | (3687.2, 689.6, 802.4) | (3580.3, 636.3, 772.8) | (3690.3, 726.7, 804.2)

=== Max: P ===
               | Disney:O              | Disney:P              | Disney:B
-------------------------------------------------------------------------------------
Netflix:O      | (4084.5, 689.6, 736.8) | (3977.4, 636.3, 707.4) | (4087.9, 726.7, 738.4)
Netflix:P      | (3580.3, 651.4, 707.4) | (3473.4, 598.2, 678.0) | (3583.4, 688.4, 709.0)
```

3.4. Análisis de Resultados: Matriz de Pagos y Equilibrios

Mediante la base de datos que se obtiene de los 3 jugadores, donde se puede analizar los comportamientos en los últimos 5 años de estas compañías, con estas gráficas podemos ver la clara dominación que tiene NETFLIX respecto de HBO Max y Disney+, teniendo una mayor participación del mercado, con un número mucho mayor de suscriptores, y por ende de Ingresos representados de forma trimestral.

Disney fue superando en el tiempo con una tendencia alcista a HBO Max, gracias a las alianzas estratégicas y potenciamiento de franquicias que realizó. En base a estos datos obtenidos se generaron los cálculos donde se obtuvo las utilidades para cada jugador, en cada estrategia.

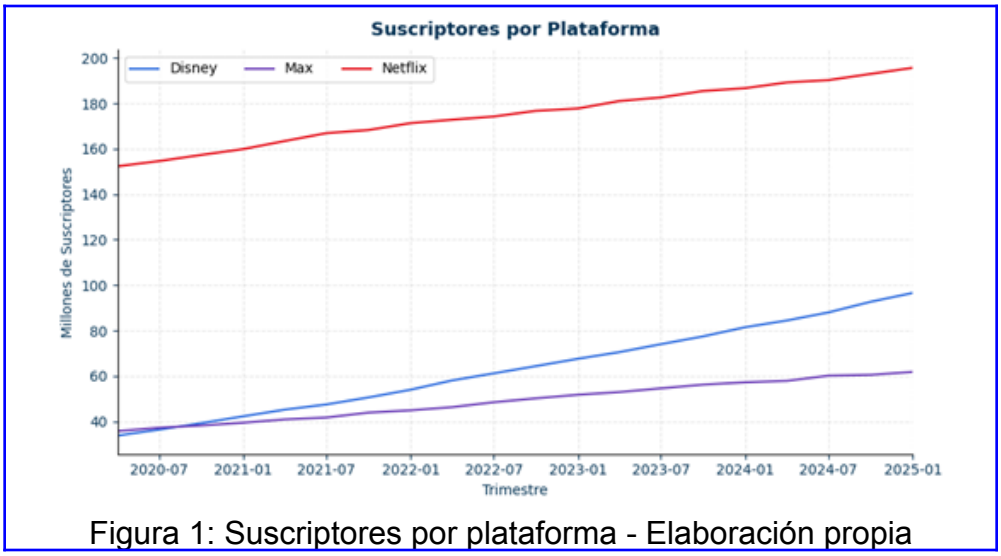


Figura 1: Suscriptores por plataforma - Elaboración propia

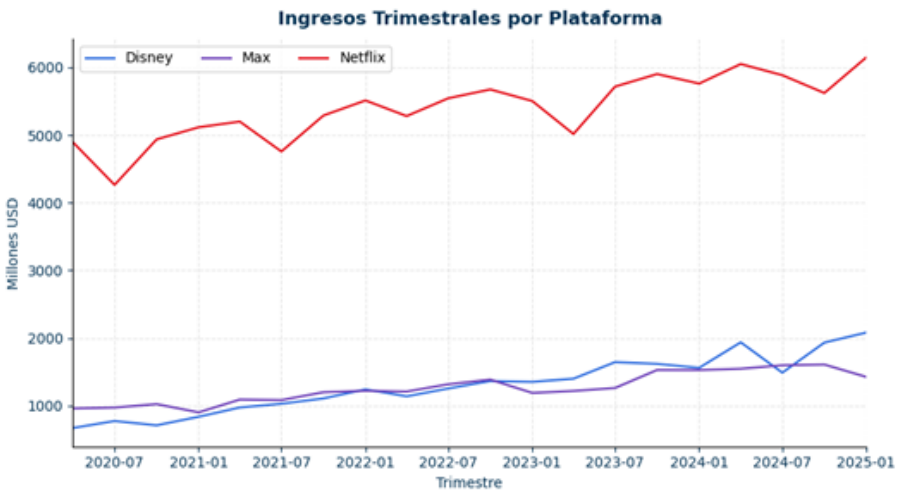


Figura 2: Ingresos trimestrales por plataforma - Elaboración propia

Una vez realizado los cálculos, con ayuda de python, mismo se encuentra en ANEXOS, determinamos los valores de la matriz de pagos donde esta estructura está dividida en dos secciones para cada estrategia de HBO Max, y los otros dos jugadores respectivamente, como se aprecia en la Tabla 1.

```
=== Max: O ===
              | Disney:O              | Disney:P              | Disney:B
-------------------------------------------------------------------------------------
Netflix:O     | (4191.5, 727.8, 831.9) | (4084.5, 674.3, 802.4) | (4195.0, 764.9, 833.8)
Netflix:P     | (3687.2, 689.6, 802.4) | (3580.3, 636.3, 772.8) | (3690.3, 726.7, 804.2)

=== Max: P ===
              | Disney:O              | Disney:P              | Disney:B
-------------------------------------------------------------------------------------
Netflix:O     | (4084.5, 689.6, 736.8) | (3977.4, 636.3, 707.4) | (4087.9, 726.7, 738.4)
Netflix:P     | (3580.3, 651.4, 707.4) | (3473.4, 598.2, 678.0) | (3583.4, 688.4, 709.0)
```

Tabla 2: Matriz de pagos, elaboración propia

Realizando el análisis de la matriz de pagos, donde se debe destacar que Netflix tiene una mayor ventaja en la estrategia O, independientemente de la estrategia que cada empresa elija, ya que al invertir en producciones propias Netflix tiene una ventaja, por el desarrollo que tiene en sus producciones, y lo mejor posicionada que se encuentra la

compañía. Entonces tendríamos para Netflix que la estrategia O es fuertemente dominante respecto de P.

También debemos destacar que la estrategia P de adoptar precios bajos o promociones, no favorece a ninguna empresa, convirtiéndose en una guerra de precios que es destructiva para toda la industria, por ende ninguna empresa tomaría esta estrategia, ya que disminuye los ingresos considerablemente, para HBO Max la estrategia a considerar es O.

Por último tenemos que analizar la estrategia de Disney+, la que mayores utilidades le reporta es B la de los paquetes, siendo esta su principal ventaja respecto a sus competidores, ya que derivada de su extenso catálogo de franquicias, mismas que no solo las adquiere, sino las desarrolla en producciones exclusivas, donde deriva en múltiples ingresos por otros conceptos como mercadería, parques temáticos, entre otros, dando así un valor agregado único en el mercado.

De esta manera quedaría el análisis de la matriz de pagos, tras descartar las otras estrategias mediante el análisis económico, donde vemos el mejor escenario para las tres compañías, que les genera mayores utilidades teniendo así todos los jugadores estrategias estrictamente dominantes, encontrándose en {O,B,O}.

```
=== Max: O ===
              | Disney:O              | Disney:P              | Disney:B
-------------------------------------------------------------------------------
Netflix:O     | (4191.5, 727.8, 831.9) | (4084.5, 674.3, 802.4) | (4195.0, 764.9, 833.8)
Netflix:P     | (3687.2, 689.6, 802.4) | (3580.3, 636.3, 772.8) | (3690.3, 726.7, 804.2)

=== Max: P ===
              | Disney:O              | Disney:P              | Disney:B
-------------------------------------------------------------------------------
Netflix:O     | (4084.5, 689.6, 736.8) | (3977.4, 636.3, 707.4) | (4087.9, 726.7, 738.4)
Netflix:P     | (3580.3, 651.4, 707.4) | (3473.4, 598.2, 678.0) | (3583.4, 688.4, 709.0)
```

Tabla 3: Matriz de pagos - estrategia estrictamente dominante, elaboración propia

También podemos apreciar que existe un **Equilibrio de Nash** estricto o puro, teniendo un equilibrio de estrategias dominantes de parte de todos los jugadores donde recae en

{O,B,O}, y de igual forma tenemos un **Óptimo de Pareto** en el mismo cuadrante donde todos los jugadores mejoran su posición.

4. Discusión

4.1. Lectura Estratégica del Equilibrio

El perfil $(N = O,\ D = B,\ M = O)$, sintetiza un patrón de **diferenciación eficiente**:

a) Para **Netflix** y **Max**, la inversión en **originales (O)** eleva simultáneamente las **altas netas** y la **monetización** (ARPU), manteniendo ventajas incluso cuando los rivales ajustan precio.
b) Para **Disney**, el **bundle (B)** actúa como palanca de **captura y retención** (amplía la base y suaviza churn), superando sistemáticamente a sus alternativas O y P. El resultado es **privadamente óptimo** para cada firma y **socialmente eficiente** dentro del conjunto evaluado (Pareto).

4.2. Mecanismos Económicos

a. **Economías de escala en contenido original.** A mayor base, la amortización del costo fijo $c^{fix}(O)$ hace que el retorno marginal de *O* sea superior a competir en precio *P* (que tiende a erosionar ARPU).
b. **Complementariedad en Bundle.** Disney internaliza sinergias entre verticales (catálogo familiar/deportes/alianzas), generando valor adicional por usuario que no depende solo de precio, sino del **portafolio**.
c. **Efectos de red y multihoming.** El hecho de que el equilibrio involucre O y B sugiere un entorno con **multihoming** de consumidores (suscripción simultánea a varias OTT), donde la diferenciación es más rentable que una guerra de precios.

4.3. Implicaciones Gerenciales

a. **Asignación de Presupuesto.** Para plataformas con escala (Netflix, Max), el **capex en originales** domina a ajustes de precio en el margen; para Disney,

acelerar bundles y *cross-selling* entre líneas de negocio refuerza ventaja relativa.

b. **Gestión del Roadmap.** La matriz de pagos permite **secuenciar** apuestas: si el rival traslada peso a P, **no conviene** seguirlo; preservar O o B mantiene la mejor respuesta.
c. **Diseño de Planes.** Los resultados favorecen **segmentar precio** vía niveles (con/sin anuncios) sin que ello desplace el núcleo estratégico (originales/bundle).

4.4. Lectura por Plataforma.

a. **Netflix.** El retorno marginal de O domina frente a P; la prioridad es sostener **pipeline** y **franquicias**, integrando planes con anuncios como táctica de monetización, no como eje estratégico.
b. **Disney.** El **bundle** cristaliza su ventaja comparativa (convergencia DTC); el desafío es **optimizar el mix** (precio del bundle, canibalización entre servicios).
c. **Max.** La apuesta por O resulta central; la elasticidad de ARPU frente a P es baja en el ejercicio, por lo que **precio** no compite con **calidad/catálogo**.

5. Conclusiones y Recomendaciones Futuras

El estudio mostró, con un juego estático 2×3×2 alimentado por un panel trimestral 2020–2024 y predicciones OLS de ***altas netas y ARPU***, que la matriz de pagos resultante identifica un ***equilibrio único de estrategias dominantes: originales*** para Netflix y Max, y ***bundle*** para Disney. Dicho perfil (O, B, O) es ***Nash puro y estricto*** y, dentro del espacio evaluado, ***Pareto-eficiente***, lo que sugiere que en OTT con economías de escala y multihoming ***la diferenciación supera sistemáticamente a la guerra de precios***.

Desde la perspectiva gerencial, ***los resultados respaldan priorizar inversión en originales*** cuando la escala amortiza costos fijos, ***y profundizar bundles*** en ecosistemas multiproducto para capturar y retener demanda. ***La política de precio*** emerge como táctica subordinada: ***tiende a erosionar ARPU sin generar suficientes altas netas compensatorias***. La matriz de pagos provee, además, un tablero

operativo para evaluar ex ante efectos cruzados de decisiones propias y rivales y para evitar espirales de descuentos no rentables.

Metodológicamente, el trabajo ***aporta un pipeline replicable*** que mapea datos a utilidades (OLS con efectos fijos, simulación contable de pagos, matriz 2×3×2, mejores respuestas, Nash y Pareto) y habilita ***análisis de sensibilidad*** a costos fijos/variables y supuestos de especificación. No obstante, su alcance es estático y se basa en datos agregados; pueden persistir endogeneidades en la elección de estrategias y la forma lineal puede omitir no linealidades relevantes. Los parámetros de costos, aunque plausibles y testeados, son supuestos.

Como líneas futuras, el análisis puede pasar de un juego estático a uno **dinámico** que capture la inercia (hábitos) y los costos de cambio (fricciones al migrar de plan o plataforma). También puede incorporar estrategias mixtas y aprendizaje de creencias (los jugadores alternan acciones y actualizan expectativas), ***trabajar con datos más granulares por país y tipo de plan***, e incorporar explícitamente el efecto de la **publicidad**. Para acercarse a relaciones causales, conviene usar instrumentos (IV) o diferencias-en-diferencias (DiD) con efectos fijos de dos vías. Asimismo, puede **ampliarse el espacio estratégico**, por ejemplo a 3×3×2 o 3×3×3 para incluir **FAST** (TV gratuita con anuncios) o **complementos deportivos**, y medir de forma explícita el **bienestar del consumidor** (precios, variedad, carga publicitaria).

6. Bibliografía

Borges, R., & Ferreira, P. (2020). *Pricing games and subscriber retention in digital services*. Journal of Digital Economics, 12(3), 45–62.

Chen, L., & Zhang, Y. (2021). *Consumer behavior and platform competition in streaming services*. International Journal of Media Economics, 14(2), 101–119.

Dixit, A., & Nalebuff, B. (2008). *Thinking strategically: The competitive edge in business, politics, and everyday life* (Rev. ed.). W. W. Norton & Company.

Fudenberg, D., & Tirole, J. (1991). *Game theory*. MIT Press.

Hagiu, A., & Wright, J. (2020). *Marketplace or reseller? Platforms in the digital economy*. Journal of Economic Perspectives, 34(2), 25–46.

Mas-Colell, A., Whinston, M., & Green, J. (1995). *Microeconomic theory*. Oxford University Press.

Myerson, R. (1991). *Game theory: Analysis of conflict*. Harvard University Press.

Smith, J., Johnson, P., & Lee, K. (2022). *Competitive strategies in streaming platforms: Price and content analysis*. Media Economics Review, 18(1), 77–95.

Tirole, J. (1988). *The theory of industrial organization*. MIT Press.

Varian, H. R. (2014). *Intermediate microeconomics: A modern approach* (9th ed.). W. W. Norton & Company.

7. Anexos

Anexo 1: DATASET

Dataset	Proveedor	URL	Fecha de consulta	Cobertura
Netflix OTT revenue and subscribers	Kaggle (Shivam)	https://www.kaggle.com/datasets/mauryansshivam/netflix-ott-revenue-and-subscribers-csv-file	07/09/2025	2012–2024 (varía)
Walt Disney OTT platforms revenue and subscribers	Kaggle (Shivam)	https://www.kaggle.com/datasets/mauryansshivam/walt-disney-ott-platforms-revenue-and-subscribers	07/09/2025	2019–2024 (varía)
Quarterly Results (Max/HBO Max/Segmento DTC)	Warner Bros. Discovery IR	https://ir.wbd.com/financials/quarterly-results/default.aspx	07/09/2025	2020–2024

Anexo 2: REPOSITORIO GITHUB CON DATASET Y NOTEBOOK (PYTHON).

https://github.com/mc-ivan/proyecto-teoria-juegos/tree/main

Anexo 3: ELABORACIÓN DEL MODELO DE TEORÍA DE JUEGOS MEDIANTE PYTHON.

Competencia Estratégica en Streaming con Estimación de Pagos y Equilibrios de Nash en un Diseño 2×3×2

Integrantes:

- Gimena Javier
- Giovanni Ortiz
- Ivan Mamani

```
# Librerias
```

```
#!pip -q install statsmodels

import numpy as np, pandas as pd, statsmodels.api as sm
from pandas.api.types import CategoricalDtype
from pathlib import Path
from IPython.display import display
import matplotlib.pyplot as plt
import io, json, math, os, warnings
warnings.filterwarnings("ignore")


# 0) LECTURA DE DATASET
GITHUB_RAW_URL =
"https://raw.githubusercontent.com/mc-ivan/proyecto-teoria-juegos/main/dataset/ott_pan
el_quarterly_synthetic_2020_2024.csv"
OUT = "outputs_project"
os.makedirs(OUT, exist_ok=True)

df = pd.read_csv(GITHUB_RAW_URL)
df.columns = df.columns.str.strip()

# Chequeo de columnas esperadas
needed = ["period","platform","subs_start_millions","subs_end_millions",
"net_adds_millions","arpu",
          "flag_O","flag_P","flag_B","riv_flag_O","riv_flag_P","riv_flag_B"]

missing = [c for c in needed if c not in df.columns]
if missing:
    raise ValueError(f"Faltan columnas en el CSV leído desde GitHub: {missing}")

# Normaliza periodo al rango 2020Q1-2024Q4
def _toQ(s):
    s = str(s).upper().replace(" ", "")
    if "Q" in s:
        y = s.split("Q")[0].replace("-","")
        q = s.split("Q")[1][0]
        return f"{y}Q{q}"
    return s

df["period"] = df["period"].map(_toQ)
df = df[(df["period"] >= "2020Q1") & (df["period"] <= "2024Q4")].copy()
df = df.sort_values(["platform","period"]).reset_index(drop=True)

print("Preview de datos:")
display(df)
```

```
Preview de datos:
```

	period	platform	subs_start_millions	subs_end_millions	net_adds_millions	revenue_millions	arpu	flag_O	flag_P	flag_B	riv_flag_O	riv_flag_P	riv_flag_B
0	2020Q1	Disney	30.000	33.804	3.804	669.942	7.00	0	0	1	0	0	0
1	2020Q2	Disney	33.804	36.401	2.597	772.957	7.34	0	0	0	0	1	0
2	2020Q3	Disney	36.401	39.282	2.881	709.528	6.25	0	1	0	1	0	0

3	2020Q4	Disney	39.282	42.307	3.025	835.879	6.83	0	0	1	1	1	0
4	2021Q1	Disney	42.307	45.217	2.910	972.829	7.41	0	0	0	1	0	0
5	2021Q2	Disney	45.217	47.503	2.286	1027.801	7.39	0	0	0	1	1	0
6	2021Q3	Disney	47.503	50.598	3.095	1106.579	7.52	0	0	0	1	0	0
7	2021Q4	Disney	50.598	53.995	3.397	1240.996	7.91	1	0	0	1	0	0
8	2022Q1	Disney	53.995	58.053	4.058	1137.847	6.77	1	1	0	0	0	0
9	2022Q2	Disney	58.053	61.176	3.123	1253.693	7.01	0	0	1	1	0	0
10	2022Q3	Disney	61.176	64.360	3.184	1363.321	7.24	0	0	1	2	0	0
11	2022Q4	Disney	64.360	67.599	3.239	1351.920	6.83	0	0	1	0	1	0
12	2023Q1	Disney	67.599	70.507	2.908	1398.323	6.75	0	0	1	1	2	0
13	2023Q2	Disney	70.507	74.018	3.511	1643.249	7.58	1	0	0	0	1	0
14	2023Q3	Disney	74.018	77.441	3.423	1617.582	7.12	0	0	1	1	0	0
15	2023Q4	Disney	77.441	81.475	4.034	1558.966	6.54	1	1	0	0	0	0
16	2024Q1	Disney	81.475	84.438	2.963	1936.205	7.78	1	0	0	1	0	0
17	2024Q2	Disney	84.438	87.999	3.561	1487.269	5.75	0	1	1	1	0	0
18	2024Q3	Disney	87.999	92.675	4.676	1932.308	7.13	1	0	1	1	1	0
19	2024Q4	Disney	92.675	96.564	3.889	2080.683	7.33	1	0	1	1	1	0
20	2020Q1	Max	35.000	35.767	0.767	958.539	9.03	0	0	0	0	0	1
21	2020Q2	Max	35.767	37.283	1.516	971.930	8.87	0	0	0	0	1	0
22	2020Q3	Max	37.283	38.220	0.937	1022.688	9.03	0	0	0	1	1	0
23	2020Q4	Max	38.220	39.479	1.259	900.920	7.73	0	1	0	1	0	1
24	2021Q1	Max	39.479	40.912	1.433	1090.102	9.04	0	0	0	1	0	0
25	2021Q2	Max	40.912	41.748	0.836	1082.433	8.73	0	0	0	1	1	0
26	2021Q3	Max	41.748	43.918	2.170	1198.896	9.33	1	0	0	0	0	0
27	2021Q4	Max	43.918	44.925	1.007	1219.370	9.15	0	0	0	2	0	0
28	2022Q1	Max	44.925	46.306	1.381	1208.355	8.83	0	0	0	1	1	0
29	2022Q2	Max	46.306	48.467	2.161	1316.397	9.26	1	0	0	0	0	1
30	2022Q3	Max	48.467	50.190	1.723	1385.144	9.36	1	0	0	1	0	1
31	2022Q4	Max	50.190	51.770	1.580	1188.344	7.77	0	1	0	0	0	1
32	2023Q1	Max	51.770	52.938	1.168	1217.230	7.75	0	1	0	1	1	1
33	2023Q2	Max	52.938	54.582	1.644	1261.210	7.82	0	1	0	1	0	0
34	2023Q3	Max	54.582	56.211	1.629	1527.282	9.19	0	0	0	1	0	1
35	2023Q4	Max	56.211	57.240	1.029	1526.483	8.97	0	0	0	1	1	0
36	2024Q1	Max	57.240	57.874	0.634	1545.405	8.95	0	0	0	2	0	0
37	2024Q2	Max	57.874	60.156	2.282	1596.946	9.02	1	0	0	0	1	1
38	2024Q3	Max	60.156	60.514	0.358	1607.324	8.88	0	0	0	2	1	1
39	2024Q4	Max	60.514	61.823	1.309	1424.003	7.76	0	1	0	2	0	1
40	2020Q1	Netflix	150.000	152.299	2.299	4892.709	10.79	0	0	0	0	0	1
41	2020Q2	Netflix	152.299	154.596	2.297	4262.772	9.26	0	1	0	0	0	0
42	2020Q3	Netflix	154.596	157.306	2.710	4935.849	10.55	1	0	0	0	1	0
43	2020Q4	Netflix	157.306	159.889	2.583	5114.769	10.75	1	0	0	0	1	1
44	2021Q1	Netflix	159.889	163.421	3.532	5198.825	10.72	1	0	0	0	0	0
45	2021Q2	Netflix	163.421	166.882	3.461	4756.363	9.60	1	1	0	0	0	0
46	2021Q3	Netflix	166.882	168.263	1.381	5288.588	10.52	0	0	0	1	0	0
47	2021Q4	Netflix	168.263	171.290	3.027	5510.945	10.82	1	0	0	1	0	0
48	2022Q1	Netflix	171.290	172.786	1.496	5279.846	10.23	0	0	0	1	1	0

49	2022Q2	Netflix	172.786	174.190	1.404	5542.942	10.65	0	0	0	1	0	1
50	2022Q3	Netflix	174.190	176.689	2.499	5673.713	10.78	1	0	0	1	0	1
51	2022Q4	Netflix	176.689	177.726	1.037	5502.293	10.35	0	0	0	0	1	1
52	2023Q1	Netflix	177.726	181.005	3.279	5015.059	9.32	1	1	0	0	1	1
53	2023Q2	Netflix	181.005	182.590	1.585	5715.713	10.48	0	0	0	1	1	0
54	2023Q3	Netflix	182.590	185.434	2.844	5901.265	10.69	1	0	0	0	0	1
55	2023Q4	Netflix	185.434	186.671	1.237	5760.185	10.32	0	0	0	1	1	0
56	2024Q1	Netflix	186.671	189.189	2.518	6049.467	10.73	1	0	0	1	0	0
57	2024Q2	Netflix	189.189	190.220	1.031	5884.634	10.34	0	0	0	1	1	1
58	2024Q3	Netflix	190.220	192.946	2.726	5621.045	9.78	1	1	0	1	0	1
59	2024Q4	Netflix	192.946	195.628	2.682	6149.184	10.55	1	0	0	1	1	1

```
# 0.1) Preprocesamiento y limpieza de Datos
# A) Duplicados y orden básico
df =
df.drop_duplicates(subset=["period","platform"]).sort_values(["platform","period"]).re
set_index(drop=True)

# B) Estandariza 'platform' y fija categorías (baseline estable = Netflix)
# (Si en la fuente aparecen variantes, se mapean aquí)
map_platform = {
    "Disney+": "Disney",
    "HBO Max": "Max",
    "WBD Max": "Max",
    "Warner Max": "Max"
}

df["platform"] = df["platform"].replace(map_platform).astype(str).str.strip()
EXPECTED_PLATFORMS = ["Netflix","Disney","Max"]
df["platform"] = pd.Categorical(df["platform"], categories=EXPECTED_PLATFORMS,
ordered=False)

# C) Tipado numérico y saneo de flags/targets
FLAG_COLS = ["flag_O","flag_P","flag_B","riv_flag_O","riv_flag_P","riv_flag_B"]
NUM_COLS  = ["subs_start_millions","subs_end_millions","net_adds_millions","arpu"]

# crear flags que falten (en 0) por si acaso
for c in FLAG_COLS:
    if c not in df.columns: df[c] = 0

# tipar
for c in FLAG_COLS:
    df[c] = pd.to_numeric(df[c], errors="coerce").fillna(0).astype(int)
for c in NUM_COLS:
    df[c] = pd.to_numeric(df[c], errors="coerce")

# D) Recalcular las flags de rivales (consistencia por periodo)
def recompute_rival_flags(frame):
    out = []
    for per, g in frame.groupby("period", sort=False):
        sums = g[["flag_O","flag_P","flag_B"]].sum(numeric_only=True)
        for _, r in g.iterrows():
            rec = r.copy()
```

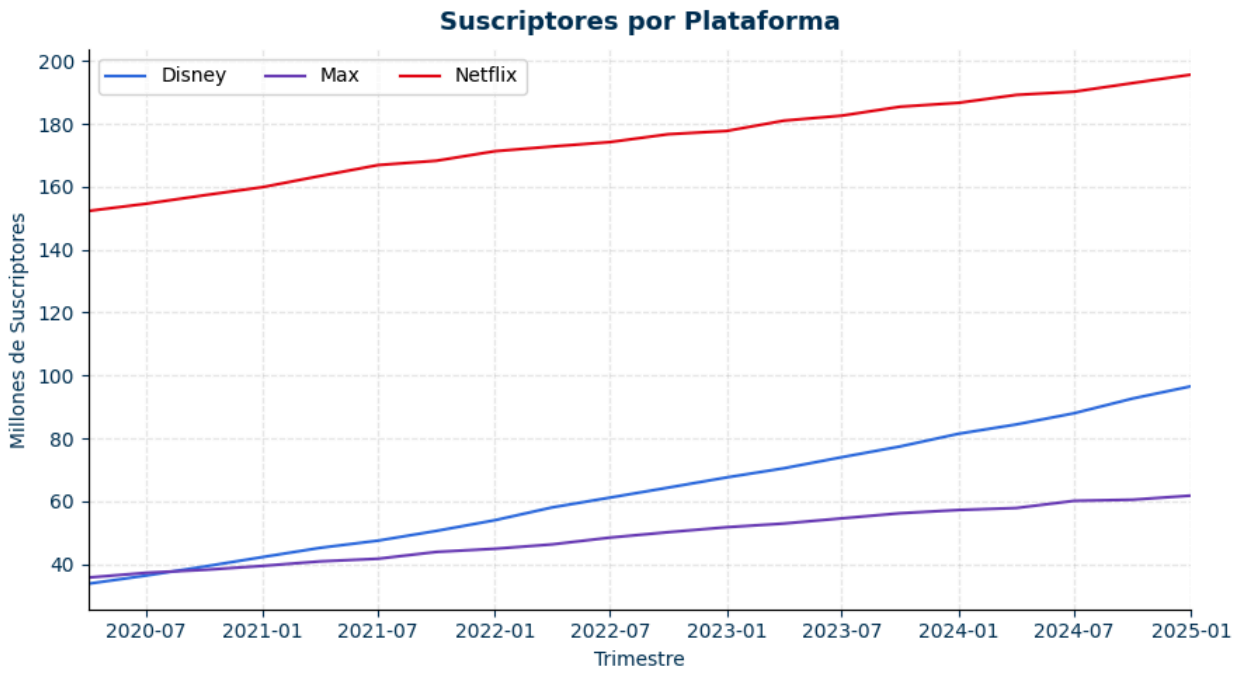
```
            if pd.isna(rec["platform"]):
                rec["riv_flag_O"] = rec["riv_flag_P"] = rec["riv_flag_B"] = 0
            else:
                rec["riv_flag_O"] = int(sums["flag_O"] - int(r["flag_O"]))
                rec["riv_flag_P"] = int(sums["flag_P"] - int(r["flag_P"]))
                rec["riv_flag_B"] = int(sums["flag_B"] - int(r["flag_B"]))
            out.append(rec)
    return pd.DataFrame(out)

df = recompute_rival_flags(df)

# E) Salvaguardar (evita outliers que desestabilicen OLS)
df["arpu"] = df["arpu"].clip(lower=3, upper=50)
df["net_adds_millions"] = df["net_adds_millions"].clip(lower=-5, upper=10)

# F) Chequeos rápidos
print("Filas por plataforma (incluyendo categorías definidas):")
print(df["platform"].value_counts(dropna=False))
print("\nRango de periodos:", df["period"].min(), "→", df["period"].max())
print("Preprocesamiento OK → dataset listo para OLS.")
```

```
Filas por plataforma (incluyendo categorías definidas):
platform
Disney     20
Max        20
Netflix    20
Name: count, dtype: int64

Rango de periodos: 2020Q1 → 2024Q4
Preprocesamiento OK → dataset listo para OLS.
```

```
# 0.2) Estadísticas generales y visualización (por plataforma)

# Colores corporativos y utilidades de estilo
COLORS = {
    "Netflix": "#E50914",  # rojo Netflix
    "Disney":  "#2D6CDF",  # azul Disney
    "Max":     "#6C3EB8",  # púrpura Max (HBO Max)
}
TITLE_COLOR = "#003153"
AXIS_COLOR = "#003153"

def savefig(name):
    path = os.path.join(OUT, name)
    plt.savefig(path, bbox_inches="tight", dpi=140)
    print(f"[fig] {path}")

def style_ax(ax, title):
    ax.set_title(title, fontsize=13, fontweight="bold", color=TITLE_COLOR, pad=10)
    for spine in ["top", "right"]:
        ax.spines[spine].set_color(AXIS_COLOR)
        ax.spines[spine].set_visible(False)
    ax.grid(True, linestyle="--", alpha=0.3)
```

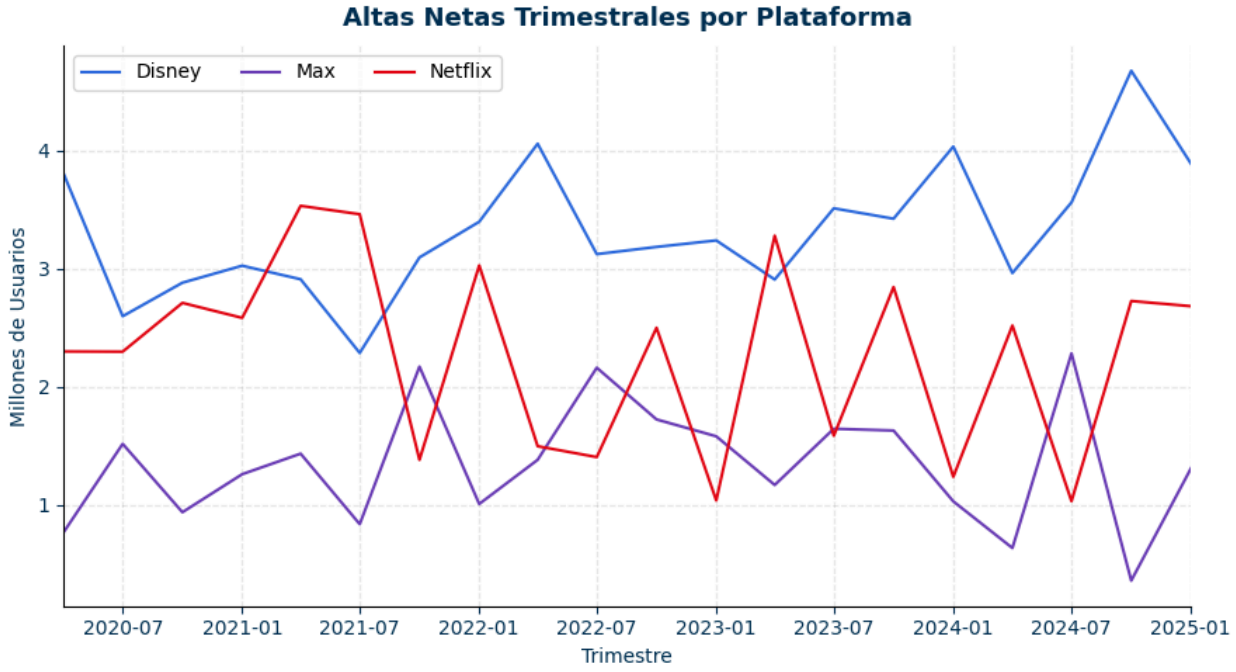
```
    ax.set_facecolor("white")
    ax.tick_params(colors=AXIS_COLOR)

# A) Preparación temporal
df_vis = df.copy()
# period 'YYYYQn' -> PeriodIndex (Q) -> Timestamp (fin de trimestre) para eje X
df_vis["period_q"] = pd.PeriodIndex(df_vis["period"], freq="Q")
df_vis["date"] = df_vis["period_q"].dt.to_timestamp(how="end")
df_vis = df_vis.sort_values(["platform", "date"]).reset_index(drop=True)

# B) Función de gráfico de líneas multi-plataforma (una métrica por gráfico)
def plot_metric(df_in, value_col, title, ylabel):
    pivot = (df_in.pivot_table(index="date", columns="platform", values=value_col,
aggfunc="mean").sort_index())
    fig, ax = plt.subplots(figsize=(9,5))
    for plat in pivot.columns:
        ax.plot(
            pivot.index, pivot[plat],
            label=str(plat),
            color=COLORS.get(plat, AXIS_COLOR),
            linewidth=1.5,
        )
        ax.margins(x=0)

    ax.set_xlabel("Trimestre", color=AXIS_COLOR)
    ax.set_ylabel(ylabel, color=AXIS_COLOR)

    # Leyenda sin marco
    leg = ax.legend(ncol=len(pivot.columns), frameon=True, loc="upper left")
    style_ax(ax, title)
    plt.tight_layout()


# C) Gráficos principales
# Suscriptores por plataforma
plot_metric(df_vis, "subs_end_millions", "Suscriptores por Plataforma", "Millones de
Suscriptores")
savefig("01_subscriptores_por_plataforma.png"); plt.show(); plt.close()
```

```
[fig] outputs_project/01_subscriptores_por_plataforma.png
```



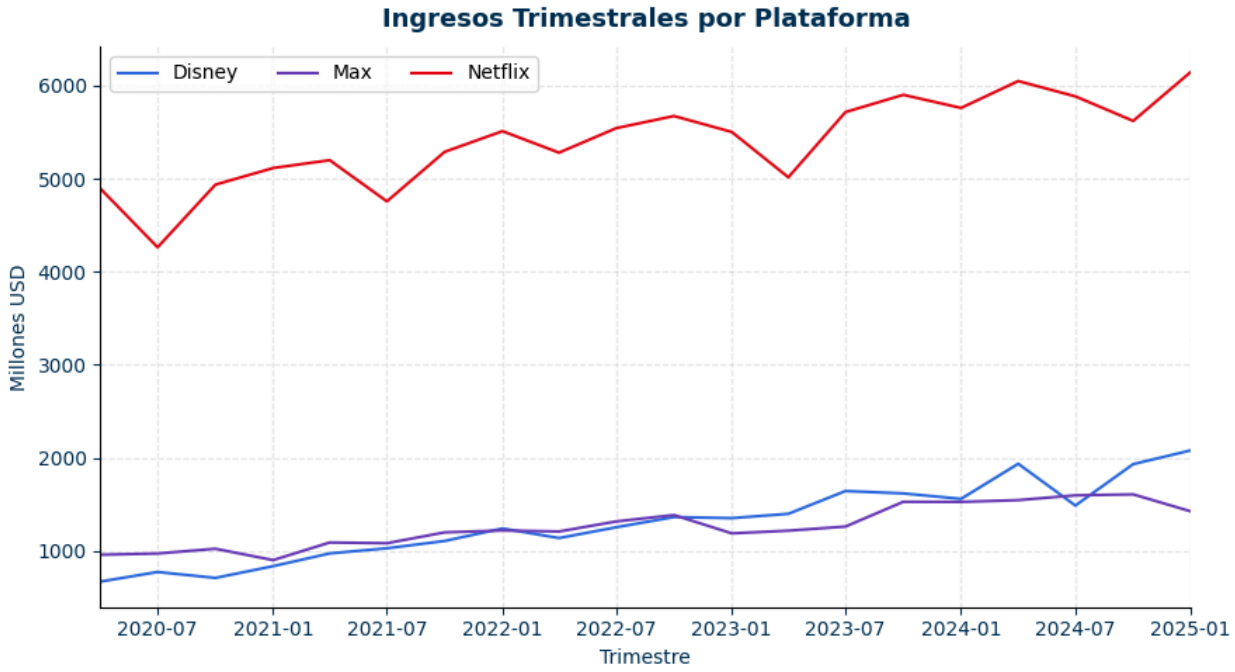

```
# Altas netas trimestrales por plataforma
plot_metric(df_vis, "net_adds_millions", "Altas Netas Trimestrales por Plataforma",
"Millones de Usuarios")
savefig("02_altas_trimestrales_por_plataforma.png"); plt.show(); plt.close()
```

```
[fig] outputs_project/02_altas_trimestrales_por_plataforma.png
```



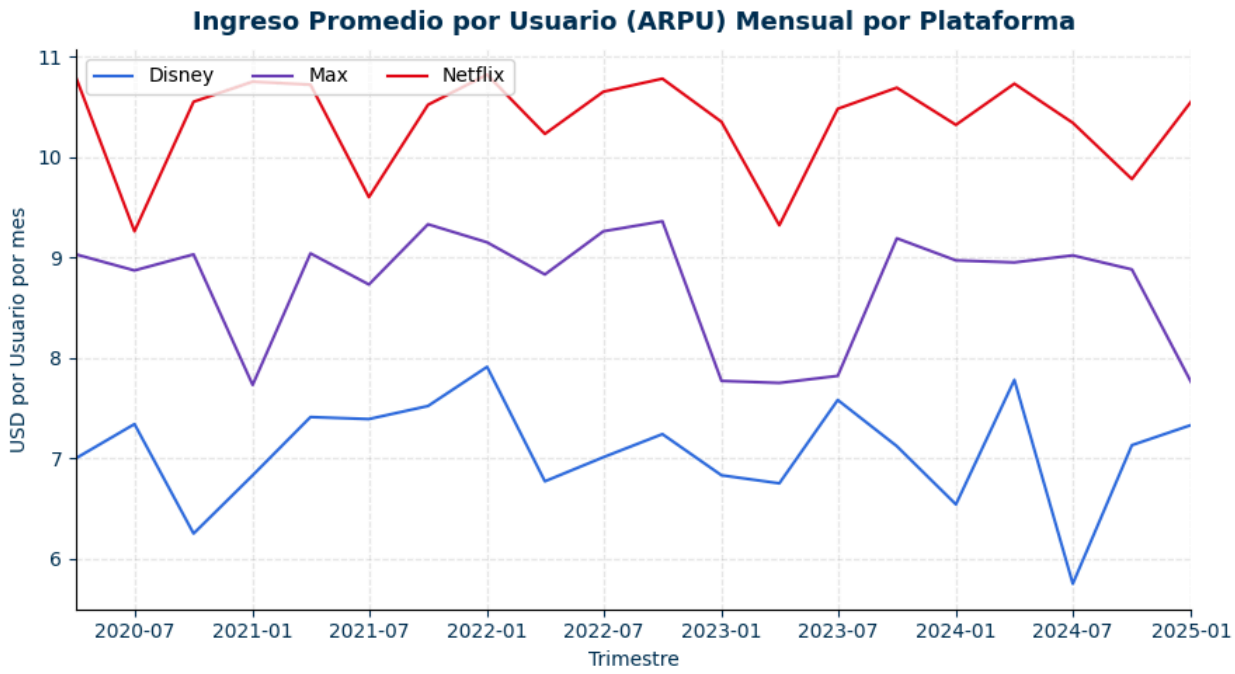
```
# Ingresos trimestrales por plataforma
plot_metric(df_vis, "revenue_millions", "Ingresos Trimestrales por Plataforma",
"Millones USD")
savefig("03_ingresos_trimestrales_por_plataforma.png"); plt.show(); plt.close()
```

```
[fig] outputs_project/03_ingresos_trimestrales_por_plataforma.png
```



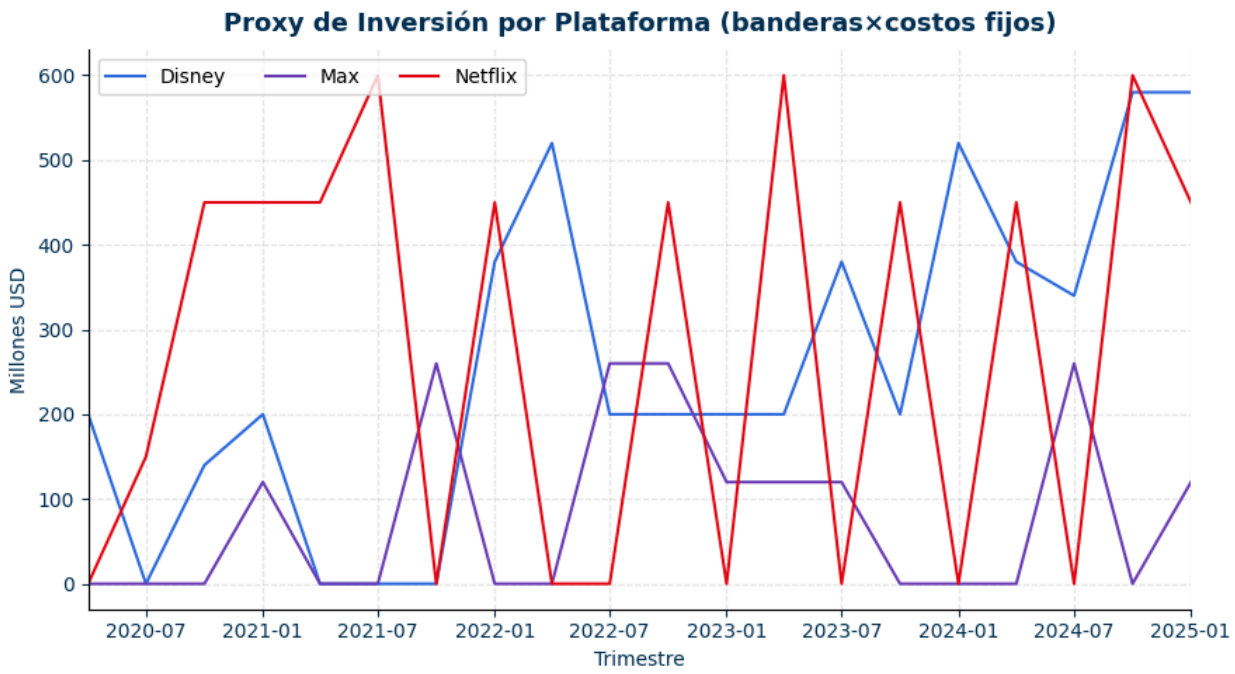

```
# ARPU Average Revenue Per User (ingreso promedio por usuario) mensual por plataforma
plot_metric(df_vis, "arpu", "Ingreso Promedio por Usuario (ARPU) Mensual por
Plataforma", "USD por Usuario por mes")
savefig("04_arpu_mensual_por_plataforma.png"); plt.show(); plt.close()
```

```
[fig] outputs_project/04_arpu_mensual_por_plataforma.png
```




```
# D) Proxy de inversión basada en flags y costos supuestos
USE_INVESTMENT_PROXY = True

if USE_INVESTMENT_PROXY:
    # Costos fijos "tipo" por trimestre (millones USD) coherentes con el bloque de
parámetros del juego
    COST_FIXED = {
        "Netflix": {"O": 450.0, "P": 150.0},
        "Disney":  {"O": 380.0, "P": 140.0, "B": 200.0},
        "Max":     {"O": 260.0, "P": 120.0},
    }
    # Proxy lineal: suma de costos activados por las banderas (pueden coexistir O y P
en un trimestre)
    def investment_proxy(row):
        p = row["platform"]
        cf = COST_FIXED.get(p, {})
        val = 0.0
        val += cf.get("O", 0.0) * float(row.get("flag_O", 0))
        val += cf.get("P", 0.0) * float(row.get("flag_P", 0))
        val += cf.get("B", 0.0) * float(row.get("flag_B", 0))
        return val

    df_vis["invest_proxy_millions"] = df_vis.apply(investment_proxy, axis=1)
    plot_metric(df_vis, "invest_proxy_millions", "Proxy de Inversión por Plataforma
(banderas×costos fijos)", "Millones USD")
    savefig("05_inversion_trimestral_por_plataforma.png"); plt.show(); plt.close()
```

```
[fig] outputs_project/05_inversion_trimestral_por_plataforma.png
```




```
# ---- E) Descriptivas para el paper ----
cols = ["subs_end_millions","net_adds_millions","arpu"]
if "revenue_millions" in df_vis.columns:
    cols.append("revenue_millions")
desc_by_platform =
(df_vis.groupby("platform")[cols].agg(["mean","std","min","max"]).round(2))
print("\n=== Descriptivas por plataforma ===")
display(desc_by_platform)
```

```
=== Descriptivas por plataforma ===
```

	subs_end_millions				net_adds_millions				arpu				revenue_millions			
	mean	std	min	max	mean	std	min	max	mean	std	min	max	mean	std	min	max
platform																
Disney	63.27	19.48	33.80	96.56	3.33	0.56	2.29	4.68	7.07	0.52	5.75	7.91	1304.89	412.95	669.94	2080.68
Max	49.02	8.47	35.77	61.82	1.34	0.52	0.36	2.28	8.72	0.59	7.73	9.36	1262.45	222.22	900.92	1607.32
Netflix	174.95	13.11	152.30	195.63	2.28	0.81	1.03	3.53	10.36	0.49	9.26	10.82	5402.81	478.08	4262.77	6149.18

```
# 1) ESPECIFICACIÓN EMPÍRICA (versión con fórmulas): evita dummies manuales y
problemas de colinealidad

# Asegura que las flags (propias y rivales) sean numéricas; NaN→0 para no romper OLS
for c in ["flag_O","flag_P","flag_B","riv_flag_O","riv_flag_P","riv_flag_B"]:
    df[c] = pd.to_numeric(df[c], errors="coerce").fillna(0)

# Declara 'platform' como categórica; el baseline (referencia) lo define la fórmula
(Netflix)
df["platform"] = df["platform"].astype("category")
```

```
# Calculo de OLS (Mínimos Cuadrados Ordinarios)
# Modelo 1: altas netas (millones) como función de estrategias propias, presión rival
y FE de plataforma
m_net = smf.ols(
    'net_adds_millions ~ flag_O + flag_P + flag_B + riv_flag_O + riv_flag_P +
riv_flag_B '
    '+ C(platform, Treatment("Netflix"))',
    data=df
).fit()

# Modelo 2: ARPU (USD/usuario/mes) con la misma especificación explicativa
m_arp = smf.ols(
    'arpu ~ flag_O + flag_P + flag_B + riv_flag_O + riv_flag_P + riv_flag_B '
    '+ C(platform, Treatment("Netflix"))',
    data=df
).fit()

# Resumen compacto de coeficientes (magnitud, error estándar, t, p, intervalos)
print("=== OLS net_adds (coeficientes) ===")
display(m_net.summary().tables[1])

print("\n=== OLS ARPU (coeficientes) ===")
display(m_arp.summary().tables[1])
```

```
=== OLS net_adds (coeficientes) ===
```

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2.0212	0.116	17.373	0.000	1.788	2.255
C(platform, Treatment("Netflix"))[T.Disney]	0.7792	0.123	6.357	0.000	0.533	1.025
C(platform, Treatment("Netflix"))[T.Max]	-0.5229	0.098	-5.359	0.000	-0.719	-0.327
flag_O	0.9969	0.088	11.383	0.000	0.821	1.173
flag_P	0.3554	0.097	3.667	0.001	0.161	0.550
flag_B	0.8291	0.135	6.149	0.000	0.558	1.100
riv_flag_O	-0.2775	0.069	-4.048	0.000	-0.415	-0.140
riv_flag_P	-0.2190	0.075	-2.904	0.005	-0.370	-0.068
riv_flag_B	-0.2160	0.094	-2.306	0.025	-0.404	-0.028

```
=== OLS ARPU (coeficientes) ===
```

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	10.5113	0.040	264.523	0.000	10.432	10.591
C(platform, Treatment("Netflix"))[T.Disney]	-2.9902	0.042	-71.421	0.000	-3.074	-2.906
C(platform, Treatment("Netflix"))[T.Max]	-1.4985	0.033	-44.960	0.000	-1.565	-1.432
flag_O	0.2735	0.030	9.142	0.000	0.213	0.334
flag_P	-1.2306	0.033	-37.179	0.000	-1.297	-1.164
flag_B	-0.4670	0.046	-10.139	0.000	-0.559	-0.375
riv_flag_O	0.0248	0.023	1.058	0.295	-0.022	0.072
riv_flag_P	-0.1818	0.026	-7.058	0.000	-0.233	-0.130
riv_flag_B	0.0282	0.032	0.881	0.382	-0.036	0.092

```
# 2) PARÁMETROS DEL JUEGO
# Conjunto de jugadores (orden útil para construir perfiles y mejores respuestas)
PLAYERS = ["Netflix","Disney","Max"]
# Estrategias disponibles por jugador:
# - Netflix: O = "originales", P = "precio/monetización"
# - Disney:  O, P y B = "bundle" (p.ej. Disney+ con Hulu/ESPN+)
# - Max:     O, P (no consideramos bundle aquí)
STRATS  = {"Netflix":["O","P"], "Disney":["O","P","B"], "Max":["O","P"]}

# Stock de referencia S0 (millones de suscriptores)
# Usamos el PROMEDIO histórico del inicio de trimestre por plataforma para suavizar
ruido.
S0 = df.groupby("platform")["subs_start_millions"].mean().to_dict()

# Estructura de costos:
# COST_FIXED: costo FIJO por TRIMESTRE (en millones USD) asociado a la estrategia
elegida por cada jugador.
#   - O suele ser más caro (producción de originales)
#   - P implica costes comerciales/tecnológicos (pricing/ads), típicamente menores que
O
#   - B (solo Disney) tiene un costo intermedio (gestión de empaquetado y descuentos)
COST_FIXED = {
    "Netflix": {"O": 450.0, "P": 150.0},
    "Disney":  {"O": 380.0, "P": 140.0, "B": 200.0},
    "Max":     {"O": 260.0, "P": 120.0},
}
# Costo VARIABLE mensual por usuario (USD/usuario/mes):
# Aproxima gastos de distribución, CDN, pagos, soporte, etc.
CV_MONTHLY = 2.0
# Conversión de trimestre a meses:
# ARPU viene mensual; multiplicamos por 3 para llevar la contribución a base
trimestral.
MONTHS_PER_Q = 3.0


# 3) PREDICCIÓN POR PERFIL
# Objetivo: dado un perfil de estrategias (NetFlix,Disney,Max), construir la fila de X
que entiende
# el modelo por fórmulas, predecir (net_adds, arpu) por jugador y calcular su
beneficio.
PLATFORM_CAT = CategoricalDtype(categories=["Netflix","Disney","Max"], ordered=False)

def _exog_row(player, profile):
    """Construye el exógeno para un jugador dado el perfil (DataFrame con
'platform')."""
    # Flags propias del jugador (O/P/B). Nota: B aplica efectivamente solo a Disney.
    fO = 1.0 if profile[player] == "O" else 0.0
    fP = 1.0 if profile[player] == "P" else 0.0
    fB = 1.0 if profile[player] == "B" else 0.0  # solo Disney usará B

    # Conteos de estrategias rivales en el mismo perfil (O, P, B)
    rO = sum(1 for j in PLAYERS if j != player and profile[j] == "O")
    rP = sum(1 for j in PLAYERS if j != player and profile[j] == "P")
    rB = sum(1 for j in PLAYERS if j != player and profile[j] == "B")
```

```
    # DataFrame con los nombres EXACTOS de la fórmula (incluye 'platform')
    X = pd.DataFrame([{
        "flag_O": fO,
        "flag_P": fP,
        "flag_B": fB,
        "riv_flag_O": float(rO),
        "riv_flag_P": float(rP),
        "riv_flag_B": float(rB),
        "platform": player, # requerido por C(platform, Treatment("Netflix"))
    }])

    # Asegura que 'platform' tenga el dtype categórico consistente con el modelo
    X["platform"] = X["platform"].astype(PLATFORM_CAT)
    return X

def predict_netadds_arpu(player, profile):
    """Predice (net_adds, arpu) para 'player' bajo 'profile' usando los modelos
OLS."""
    X = _exog_row(player, profile)
    net  = float(m_net.predict(X).iloc[0])
    arpu = float(m_arp.predict(X).iloc[0])
    # salvaguardas razonables
    return max(-1.0, net), max(4.0, arpu)

def profit(player, profile):
    """Calcula el beneficio trimestral (millones USD) del jugador bajo el perfil."""
    net, arpu = predict_netadds_arpu(player, profile)
    S_end = S0.get(player, 10.0) + net
    contrib_per_user_Q = (arpu - CV_MONTHLY) * MONTHS_PER_Q
    contrib_millions   = S_end * max(0.0, contrib_per_user_Q)
    cf = COST_FIXED[player][profile[player]]
    return contrib_millions - cf


# 4) ENUMERAR LOS 12 PERFILES (2×3×2)
from itertools import product

# Genera todas las combinaciones puras de estrategias:
# N ∈ {O,P}, D ∈ {O,P,B}, M ∈ {O,P} → 2×3×2 = 12 perfiles
profiles = [{"Netflix":n, "Disney":d, "Max":m}
            for n in STRATS["Netflix"]
            for d in STRATS["Disney"]
            for m in STRATS["Max"]]

records = []
for prof in profiles:
    # Para cada perfil, calcula el pago (beneficio trimestral, en millones)
    rec = {"profile": prof}
    rec["Netflix_profit_m"] = profit("Netflix", prof)
    rec["Disney_profit_m"]  = profit("Disney",  prof)
    rec["Max_profit_m"]     = profit("Max",     prof)
    records.append(rec)
```

```
# Tabla final de pagos por perfil (cada fila = un perfil N,D,M)
payoff_df = pd.DataFrame(records)
print("\n=== Pagos (millones) por perfil 2×3×2 ===")
display(payoff_df)
```

```
=== Pagos (millones) por perfil 2×3×2 ===
```

	profile	Netflix_profit_m	Disney_profit_m	Max_profit_m
0	{'Netflix': 'O', 'Disney': 'O', 'Max': 'O'}	4191.529452	727.778294	831.901556
1	{'Netflix': 'O', 'Disney': 'O', 'Max': 'P'}	4084.520784	689.614826	736.801968
2	{'Netflix': 'O', 'Disney': 'P', 'Max': 'O'}	4084.520784	674.321017	802.408497
3	{'Netflix': 'O', 'Disney': 'P', 'Max': 'P'}	3977.439587	636.291027	707.442385
4	{'Netflix': 'O', 'Disney': 'B', 'Max': 'O'}	4194.959411	764.852429	833.765782
5	{'Netflix': 'O', 'Disney': 'B', 'Max': 'P'}	4087.913219	726.662962	738.381987
6	{'Netflix': 'P', 'Disney': 'O', 'Max': 'O'}	3687.180980	689.614826	802.408497
7	{'Netflix': 'P', 'Disney': 'O', 'Max': 'P'}	3580.305789	651.378829	707.442385
8	{'Netflix': 'P', 'Disney': 'P', 'Max': 'O'}	3580.305789	636.291027	772.842907
9	{'Netflix': 'P', 'Disney': 'P', 'Max': 'P'}	3473.358069	598.188506	678.010272
10	{'Netflix': 'P', 'Disney': 'B', 'Max': 'O'}	3690.326733	726.662962	804.235198
11	{'Netflix': 'P', 'Disney': 'B', 'Max': 'P'}	3583.414018	688.400966	708.984881

```
# 5) MEJORES RESPUESTAS Y NASH PUROS
def best_responses(df):
    # Diccionario: para cada jugador, el conjunto de índices de filas (perfiles) donde
juega su mejor respuesta
    BR = {p:set() for p in PLAYERS}

    for p in PLAYERS:
        # Conjunto de "los otros" jugadores (para fijar sus estrategias)
        others = [q for q in PLAYERS if q!=p]
        groups = {}

        # Agrupa filas (perfiles) por las estrategias de los rivales s_{-p}
        # key = tupla ordenada con (jugador, estrategia) de los rivales
        for idx, row in df.iterrows():
            key = tuple((q, row["profile"][q]) for q in others)
            groups.setdefault(key, []).append(idx)

        # En cada grupo (rivales fijos), identifica las filas donde el pago de p es
máximo
        col = f"{p}_profit_m"
        for _, idxs in groups.items():
            sub = df.loc[idxs]
            mx  = sub[col].max()
            # Añade todos los empates en el máximo (mejor respuesta puede no ser
única)
            BR[p].update([i for i in idxs if abs(df.loc[i,col]-mx) < 1e-12])
    return BR
```

```
# Conjunto de índices que son mejor respuesta simultánea para TODOS los jugadores → Nash puro
BR = best_responses(payoff_df)
nash_idx = set.intersection(*BR.values())

# Tabla con los perfiles de equilibrio de Nash puro (si existen)
nash_df  = payoff_df.loc[sorted(nash_idx)].reset_index(drop=True)
print("\n=== Equilibrios de Nash PUROS ===")
display(nash_df if len(nash_df) else pd.DataFrame({"msg":["No se detectaron Nash puros con estos supuestos/costos."]}))
```

```
=== Equilibrios de Nash PUROS ===
```

	profile	Netflix_profit_m	Disney_profit_m	Max_profit_m
0	{'Netflix': 'O', 'Disney': 'B', 'Max': 'O'}	4194.959411	764.852429	833.765782

```
# 6) PARETO-EFICIENCIA
def pareto_front(df):
    # Matriz de pagos (N, D, M) para cada perfil (filas)
    vals = df[["Netflix_profit_m","Disney_profit_m","Max_profit_m"]].values
    efficient = []

    # Revisa, para cada perfil i, si existe algún perfil j que lo domine
    for i in range(len(vals)):
        v = vals[i]; dominated = False
        for j in range(len(vals)):
            if j==i: continue
            w = vals[j]
            # Dominancia de Pareto: w es >= en todos y > en alguno → i está dominado
            if np.all(w >= v) and np.any(w > v):
                dominated = True; break
        # Si no fue dominado por ninguno, es Pareto-eficiente (no dominado)
        if not dominated:
            efficient.append(i)

    # Devuelve solo los perfiles no dominados (frente de Pareto)
    return df.iloc[efficient].reset_index(drop=True)

pareto_df = pareto_front(payoff_df)
print("\n=== Perfiles Pareto-eficientes ===")
display(pareto_df)
```

```
=== Perfiles Pareto-eficientes ===
```

	profile	Netflix_profit_m	Disney_profit_m	Max_profit_m
0	{'Netflix': 'O', 'Disney': 'B', 'Max': 'O'}	4194.959411	764.852429	833.765782

```
# 7) REBANADAS (FIJANDO MAX)
def show_slices(df, fixed_player="Max", rows_player="Netflix", cols_player="Disney"):
    # Recorre cada valor posible de la estrategia del jugador fijado (p.ej., Max ∈ {O,P})
    for val in STRATS[fixed_player]:
        print(f"\n=== {fixed_player}: {val} ===")

        # Encabezado de la tabla: columnas son las estrategias del jugador en columnas (Disney)
        header = [""] + [f"{cols_player}:{c}" for c in STRATS[cols_player]]
```

```
        print("{:<14} | {:<22} | {:<22} | {:<22}".format(*header))
        print("-"*88)

        # Filas: estrategias del jugador en filas (Netflix)
        for r in STRATS[rows_player]:
            cells = []
            # Para cada combinación (fila r, columna c), busca el perfil correspondiente en df
            for c in STRATS[cols_player]:
                mask = df["profile"].apply(lambda d: d[fixed_player]==val and
d[rows_player]==r and d[cols_player]==c)
                sub = df[mask]
                if len(sub)==1:
                    # Extrae y formatea los pagos (N, D, M) de esa celda
                    n = sub["Netflix_profit_m"].iloc[0]
                    d = sub["Disney_profit_m"].iloc[0]
                    m = sub["Max_profit_m"].iloc[0]
                    cells.append(f"({n:.1f}, {d:.1f}, {m:.1f})")
                else:
                    # Si no hay exactamente un perfil (no debería pasar), deja un guion
                    cells.append("—")

            # Imprime la fila completa para la estrategia r del jugador en filas (Netflix)
            print("{:<14} | {:<22} | {:<22} | {:<22}".format(f"{rows_player}:{r}", *cells))

# Visualiza las dos rebanadas 2×3 fijando Max (O y P);
# filas = estrategias de Netflix, columnas = estrategias de Disney.
show_slices(payoff_df, fixed_player="Max", rows_player="Netflix", cols_player="Disney")
```

```
=== Max: O ===
               | Disney:O               | Disney:P               | Disney:B
----------------------------------------------------------------------------------------
Netflix:O      | (4191.5, 727.8, 831.9) | (4084.5, 674.3, 802.4) | (4195.0, 764.9, 833.8)
Netflix:P      | (3687.2, 689.6, 802.4) | (3580.3, 636.3, 772.8) | (3690.3, 726.7, 804.2)

=== Max: P ===
               | Disney:O               | Disney:P               | Disney:B
----------------------------------------------------------------------------------------
Netflix:O      | (4084.5, 689.6, 736.8) | (3977.4, 636.3, 707.4) | (4087.9, 726.7, 738.4)
Netflix:P      | (3580.3, 651.4, 707.4) | (3473.4, 598.2, 678.0) | (3583.4, 688.4, 709.0)
```

```
# 8) EXPORTS ÚTILES (para el paper)
out_dir = Path(OUT)
payoff_df.to_csv(out_dir/"payoffs_2x3x2.csv", index=False)
if len(nash_df):
    nash_df.to_csv(out_dir/"nash_puros.csv", index=False)
pareto_df.to_csv(out_dir/"pareto_front.csv", index=False)
print("\nGuardado en /content: payoffs_2x3x2.csv, nash_puros.csv (si hay),
pareto_front.csv")
```

```
Guardado en /content: payoffs_2x3x2.csv, nash_puros.csv (si hay), pareto_front.csv
```