1. Título del proyecto

Predicción de precios de videojuegos de Steam y exploración de datos con Pandas

2. Descripción breve del proyecto

En este proyecto he trabajado con un dataset real de videojuegos de la plataforma Steam.
Mi objetivo ha sido doble:

1. Explorar los datos con Pandas para entender mejor cómo son los juegos del dataset.
2. Construir un modelo sencillo que me ayude a responder la pregunta:
   "¿Qué hace que un juego de Steam sea más caro o más barato?"

Más que buscar el modelo perfecto, me he centrado en aprender a explorar, limpiar, visualizar y analizar los datos de forma razonada.

3. Contexto y motivación

Elegí este proyecto por varias razones:

- Me gustan los videojuegos y me resultaba motivador trabajar con datos de Steam.
- Es un buen ejemplo de problema de regresión: queremos aproximar un **precio** a partir de varias características (reseñas, jugadores simultáneos, logros, DLC, etc.).
- Me permitía practicar lo que hemos visto en el bootcamp: carga de datos, limpieza, exploración, visualización con Matplotlib/Seaborn y modelos sencillos con scikit-learn.

4. Datos utilizados

He trabajado con un fichero de Kaggle con juegos de steam, en los que habian unos 120.000 juegos aprox.

Algunas de las variables más relevantes que he utilizado son:

- `Name`: nombre del juego.

- `Price`: precio del juego en dólares.
- `Positive`: número de reseñas positivas.
- `Negative`: número de reseñas negativas.
- `Peak_CCU`: número máximo de jugadores simultáneos (popularidad).
- `DLC_count`: número de DLCs.
- `Achievements`: número de logros.
- `Metacritic_score`: puntuación de Metacritic (cuando existe).

Problemas detectados en los datos

Durante la exploración con Pandas encontré varios problemas importantes:

- Muchos juegos con **precio 0** (free-to-play) que siguen otra lógica de negocio.
- Algunos precios extraños, como juegos a **$999**, que parecen errores.
- Columnas con casi todos los valores vacíos, como `Metacritic_url` o `Score_rank`.
- Muchos juegos sin reseñas: un porcentaje elevado tiene `Positive = 0`.

Esto me obligó a tomar decisiones de limpieza y filtrado antes de entrenar ningún modelo.

5. Limpieza y preparación de los datos

Las principales decisiones de limpieza han sido:

1. **Renombrar columnas**: el CSV venía sin nombres claros y reasigné los nombres a mano para trabajar más cómodo.
2. **Filtrar por rango de precio**:
    a. Primero probé a quitar solo los juegos gratis (Price = 0), pero seguían apareciendo valores extremos.
    b. Después, como parte de mi proceso de ensayo y error, decidí quedarme únicamente con juegos cuyo precio está entre **$0.99 y $80**.
    De esta forma:
        i. Elimino los free-to-play (otra lógica).
        ii. Elimino precios sin sentido como $999.
        iii. Me centro en el rango “normal” donde están la mayoría de juegos.

3. **Crear una variable de total de reseñas**:
   `Total_reviews = Positive + Negative`
   Esta variable resume cuánta atención ha recibido un juego.
4. **Eliminar columnas casi vacías** para no complicar el análisis (Como comenté antes, `Metacritic_url`, `Score_rank`).
5. **Quitar filas con NaN** en las columnas que uso para el modelo, para que scikit-learn pueda entrenar sin problemas.

6. Análisis exploratorio (EDA)

La exploración la he hecho principalmente con **Pandas**, **Matplotlib** y **Seaborn,** a base de ensayo y error**.**

6.1 Distribución de precios

Primero hice un histograma simple de `Price`. Al principio el gráfico se veía raro porque:

- Mezclaba juegos gratis, muy baratos y precios extremos.
- La escala se "aplastaba".

Tras el filtrado a **$0.99–80**, repetí el histograma:

- Probé con pocos "bins" y se veía muy plano.
- Mejoré el gráfico añadiendo:
  - Más bins.
  - Borde negro en las barras.
  - Líneas verticales para la **media** y la **mediana**.

Conclusión: la mayoría de juegos se concentran en precios bajos (sobre todo entre $1 y $10), y hay pocos juegos caros.

También hice una división por rangos ($1–5, $5–10, $10–20, $20–40, $40–80) para ver cuántos juegos hay en cada grupo y qué porcentaje suponen.

6.2 Relación entre popularidad y precio

Quería saber si los juegos más populares son más caros. Para eso usé:

- `Peak_CCU` (jugadores simultáneos).
- `Total_reviews` (reseñas totales).

Primero hice un scatter plot (gráfico de dispersión) simple de `Peak_CCU` vs `Price`. Se veía un punto enorme arriba a la derecha y el resto muy aplastado cerca del origen.
Mejoré el gráfico:

- Limitando el eje X (por ejemplo hasta 50.000).
- Añadiendo transparencia (`alpha`) y tamaños pequeños de punto.

Así pude ver mejor que, en general:

- Los juegos con más jugadores tienden a tener precios más altos.

Para confirmarlo, calculé la **correlación** entre `Peak_CCU` y `Price`. El valor era positivo y moderado, lo que apoya lo que se ve en el gráfico.

Repetí algo parecido con `Total_reviews`, `Achievements` y `DLC_count`.

6.3 Correlaciones

Construí una tabla de correlaciones entre `Price` y varias variables:

- `Positive`
- `Negative`
- `Total_reviews`
- `Peak_CCU`
- `Achievements`
- `DLC_count`
- `Metacritic_score`

Después hice un gráfico de barras horizontal para verlo más claro.

Conclusiones principales:

- **`Peak_CCU`** es la variable que **más se relaciona con el precio**.
- Le siguen `Achievements` y `Total_reviews`.
- `Metacritic_score` tiene muy poca correlación, entre otras cosas porque muchos juegos no tienen puntuación.

7. Variables finales utilizadas en el modelo

Para el modelo de predicción escogí un conjunto reducido y sencillo de variables numéricas:

- `Positive`
- `Negative`
- `Peak_CCU`
- `Achievements`
- `DLC_count`
- `Total_reviews`

La variable objetivo (y) es **`Price`**.

No he hecho un feature engineering muy complejo, porque mi objetivo no era optimizar al máximo el modelo, sino entender las relaciones básicas.

8. Modelo utilizado

He usado un modelo de **RandomForestRegressor** de scikit-learn.

Proceso:

1. He preparado X (características) e y (precio).
2. He dividido el dataset en entrenamiento y test (80% / 20%).
3. He entrenado primero un modelo muy simple:
    a. `n_estimators = 50`
    b. Parámetros por defecto.
4. He evaluado el modelo con:
    a. **MAE** (Mean Absolute Error).
    b. **R^2** (coeficiente de determinación).

Después, en modo ensayo y error, he mejorado ligeramente el modelo:

- Aumentando el número de árboles a `n_estimators = 100`.
- Limitando la profundidad con `max_depth = 10` para evitar overfitting.

9. Resultados

Los resultados exactos dependen de la ejecución, pero de forma aproximada he obtenido:

- **MAE** alrededor de **4–5 dólares**.
- **R^2** alrededor de **0.20–0.30**.

Interpretación:

- El error medio absoluto de 4–5 dólares quiere decir que, en promedio, me estoy equivocando esa cantidad al predecir el precio de un juego dentro del rango $0.99–80.
- Un R^2 del 20–30% indica que el modelo explica solo una parte de la variación del precio. Esto tiene sentido porque faltan variables muy importantes (género del juego, calidad real, campañas de marketing, etc.).

También he visualizado:

- Un gráfico de **precio real vs precio predicho**:
  Puntos cerca de la diagonal = buena predicción; puntos lejos = errores grandes.
- La **importancia de las variables** en el Random Forest:
  - La más importante suele ser `Peak_CCU`.
  - Después vienen `Total_reviews` y `Achievements`.

10. Conclusiones del proyecto

¿Qué hace que un juego sea más caro según mi análisis?

En el rango de **$0.99 a $80**, los juegos más caros suelen tener:

1. **Más jugadores simultáneos (Peak_CCU)**
   Es la variable que más pesa. Los juegos más populares tienden a tener precios más altos.
2. **Más contenido (Achievements)**
   Los juegos con más logros suelen ser más grandes y completos, y eso se refleja en el precio.
3. **Más reseñas (Total_reviews)**
   Indica que el juego ha llegado a más gente. Los títulos más conocidos tienden a ser más caros.
4. **Más DLCs (DLC_count)**
   Suelen ser juegos con soporte posterior al lanzamiento y más contenido adicional.

Lo que he aprendido

- A **cargar, limpiar y explorar** un dataset real con Pandas.
- A detectar problemas en los datos (valores extremos, columnas casi vacías, juegos gratis...
- A mejorar gráficos y análisis:
  - Empiezo con algo sencillo.
  - Si el gráfico "no se ve bien", lo ajusto (límites, colores, bins, etc.).

11. Limitaciones y posibles mejoras

Limitaciones principales:

- No he utilizado información de **género del juego** (RPG, acción, deportes...).
- No he tenido en cuenta el **desarrollador/editor** (no es lo mismo un juego indie que uno de una gran empresa).
- No he incluido variables de marketing (publicidad, rebajas, campañas de Steam, etc.).
- El modelo solo ve números, no sabe si el juego es bueno, innovador o famoso en redes sociales.

Mejoras que haría con más tiempo:

- Extraer variables a partir de `Genres` y `Publishers`.

- Probar otros modelos.
- Hacer una evaluación más completa con validación cruzada.

12. Valor personal del proyecto

Ha sido un proyecto duro porque he tenido que abandonar la idea de "modelo perfecto" debido a varios problemas que he tenido a lo largo del curso... y centrarme en algo más básico pero entendible. Me ha obligado a:

- Ser honesto con lo que sé y lo que no.
- Trabajar mucho con **Pandas** y visualización.
- Aceptar que un modelo con R^2 bajo también puede ser útil para aprender, si se entiende por qué.

13. AGRADECIMIENTO

Al final, siento que este proyecto refleja la lucha personal que he tenido para adaptarme, emprenderme en este mundo, el superarme a mí mismo por más ganas de abandonar que me dieran y a saber mantener un ritmo, una estabilidad, aunque haya tenido mis altibajos.

También quiero agradecer enormemente a Matías, Miquel y mis compañeros por haber estado apoyándome todo el curso. Estos profesores hacen el Bootcamp especial, y una experiencia de las que no se olvidan jamás. He tenido una segunda familia con vosotros, justo cuando más lo necesitaba... después de atravesar tantos cambios.