

Modelo de regresión lineal: How Long To Beat Video Game Playtime

Daniel Queijeiro Albo A01710441

ABSTRACT. El propósito de este documento es presentar todo el proceso para implementar un modelo de regresión lineal multivariable diseñado para predecir el tiempo de juego promedio de videojuegos, conocido como "All Styles", basándose en diversas características del juego. El modelo fue entrenado usando un dataset que contiene estadísticas de tiempo de juego de miles de videojuegos. Se discutirá el modelo propuesto, las razones de su elección, sus limitaciones y posibles mejoras. Los resultados indican que el modelo logra un coeficiente de determinación (R^2) de aproximadamente 0.66 en el conjunto de prueba, lo que se puede interpretar como que el modelo es capaz de explicar más de la mitad de la variabilidad en el tiempo de juego.

1. INTRODUCCIÓN

La industria de los videojuegos ha crecido exponencialmente desde los 90s hasta llegar a convertirse en una de las formas de entretenimiento más influyentes a nivel global. Para jugadores y desarrolladores, comprender el tiempo que se invierte en un juego es crucial. Los jugadores buscan estimaciones para ver si en verdad vale su precio, mientras que los desarrolladores pueden usar esta información para optimizar el diseño de sus juegos y las estrategias de marketing.

Tradicionalmente, la estimación del tiempo de juego se ha basado en la experiencia personal o en encuestas manuales. Sin embargo, con el avance del Machine Learning, es posible automatizar este proceso, proporcionando predicciones más precisas y basadas en datos.

Por lo tanto, el desarrollo de un modelo de Machine Learning que pueda predecir el

tiempo de juego de un videojuego, considerando atributos como el año de lanzamiento, el mes de lanzamiento, el tiempo de la historia principal y los géneros, se convierte en una herramienta valiosa.

Los datos utilizados para este proyecto fueron obtenidos de HowLongToBeat.com, una plataforma que recopila datos de tiempo de juego aportados por la comunidad. Este dataset será nuestra base para entrenar y evaluar el modelo.

2. DESCRIPCIÓN DEL DATASET

El dataset contiene más de 50,000 entradas entre videojuegos y expansiones, con variables que incluyen:

- **id:** ID único del juego en HowLongToBeat.
- **name:** Título del juego.

- **type:** Tipo de entrada (juego, DLC/expansión, enfocado en multijugador).
- **platform:** Lista de plataformas en las que se publicó el juego.
- **genres:** Lista del género (o géneros) de juego.
- **developer:** Desarrollador del juego.
- **release_date:** Fecha de lanzamiento oficial (YYYY-MM-DD).
- **release_precision:** Precisión de la fecha (día, mes, año).
- **release_year:** Componente numérico del año de lanzamiento.
- **release_month:** Componente numérico del mes de lanzamiento.
- **release_day:** Componente numérico del día de lanzamiento.
- **main_story_polled / main_story:** Conteo y tiempo promedio para la historia principal (horas).
- **main_plus_sides_polled / main_plus_sides:** Conteo y tiempo promedio para historia + misiones secundarias (horas).
- **completionist_polled / completionist:** Conteo y tiempo promedio para el 100% de completado (horas).
- **all_styles_polled / all_styles:** Conteo y tiempo promedio para todos los estilos de juego (horas).
- **single_player_polled / single_player:** Conteo y tiempo promedio para el modo un jugador.
- **co_op_polled / co_op:** Conteo y tiempo promedio para el modo cooperativo.
- **versus_polled / versus:** Conteo y tiempo promedio para el modo versus.

- **source_url:** URL original de la página de HowLongToBeat.
- **crawled_at:** Marca de tiempo del rastreo.

3. LIMPIEZA Y TRANSFORMACIÓN DE DATOS

Debido a la enorme cantidad de datos recopilados necesitamos asegurarnos de que nos sean útiles. Por lo que les realizamos una limpieza previo a su uso en el modelo.

Para empezar debemos de decidir nuestra variable objetivo, la cual es “all_styles”, que nos dice cuánto tiempo toma completar un juego promediando el total de horas entre solo completar la historia principal, completar todos los objetivos secundarios y llegando al 100% del juego.

Teniendo en cuenta que hay juegos más largos que otros, hacemos un box plot para verificar cuál rango es aceptable de tiempo máximo para evitar sesgos en nuestro modelo al permitir los casos “outlier” de juegos extremadamente largos.

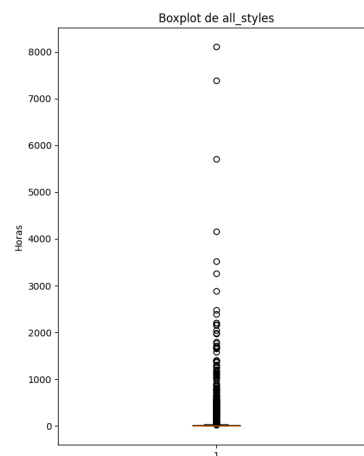


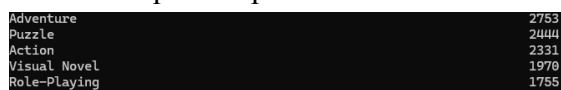
Figura 1.1

Como podemos ver en la figura 1.1, en efecto tenemos datos “outliers”, pero también vemos que la mayoría de datos se concentran hasta un rango de 2500 horas,

por lo que tomaremos eso como límite de nuestro modelo.

Después tenemos que decidir qué variables usar en nuestro modelo. Para evitar meter ruido al entrenamiento eliminamos aquellas variables que tengan demasiados valores vacíos, por lo que terminamos con las siguientes variables numéricas: [release_year, release_month, main_story y single_player]

Pero no solo nos interesa que el modelo aprenda a predecir basado en esas variables sino también dependiendo del género del juego. Así, buscamos cuales son los 5 géneros más populares en el dataset y aplicamos una técnica llamada “one-hot encoding” para convertir estas variables categóricas en variables numéricas que nos puedan ser de uso.



Adventure	2753
Puzzle	2444
Action	2331
Visual Novel	1970
Role-Playing	1755

Figura 1.2

Según los resultados mostrados en la figura 1.2, tenemos que “Adventure”, “Puzzle”, “Action”, “Visual Novel”, y “Role-Playing” son el top 5 de géneros, pero para no discriminar al resto de géneros también creamos otra variable llamada “Other”.

Ya con esto tenemos todas las variables que usaremos para el entrenamiento del modelo.

El siguiente paso para terminar con datos útiles de entrenamiento es normalizar los datos que nos quedaron después de filtrar. La normalización nos sirve para escalar todas las variables a un rango de 0 a 1, así evitamos que las variables que vengan con valores más grandes opaquen a las de menor valor.

Y finalmente, debemos separar nuestros datos en un grupo para entrenar el modelo y otro para probarlo. Gracias a que tenemos tantas filas de datos podemos usar una simple división de 80% de los datos para entrenamiento y el otro 20% para probar el modelo. En caso de que hubiéramos terminado con pocos datos después de la limpieza y transformación se podría emplear métodos como “K-folds”.

4. CONSTRUCCIÓN DEL MODELO

4a. Hipótesis

El modelo que vamos a implementar es una regresión lineal multivariable, donde la variable objetivo “all_styles” se estima como una combinación lineal de las variables de entrada.

La hipótesis se expresa como:

$$h\theta(x) = \theta_0 + \theta_1x_1 + \theta_2x_2 + \dots + \theta_nx_n$$

El objetivo del entrenamiento es ajustar los parámetros θ de manera que la función $h\theta(x)$ se aproxime lo más posible a los valores reales de la variable objetivo.

4b. Función de Costo

Para medir qué tan buenas son las predicciones en comparación con los valores reales, se utilizó Mean Squared Error(MSE):

$$mse = \frac{1}{2n} \sum (y_n - \hat{y})^2$$

El valor de esta función indica el error promedio que está cometiendo el modelo:

mientras menor sea el costo, mejor es el ajuste.

4c. Descenso de Gradiente

Para minimizar la función de costo y encontrar los parámetros óptimos θ , se implementó el algoritmo de Descenso de Gradiente.

En cada iteración, los parámetros se actualizan en la dirección contraria a la pendiente del gradiente:

$$\Theta_j := \Theta_j - \frac{\alpha}{m} \sum_{i=1}^m [(h_{\Theta}(x_i) - y)x_i]$$

Este procedimiento se repite hasta que el error converge o hasta alcanzar el número máximo de iteraciones definido.

Para evitar que el entrenamiento dure demasiado tiempo sin conseguir mejoras significativas, se implementó un early stopping en caso de que en 800 épocas no se detecte una mejora.

5. RESULTADOS

5a. Curva de Error

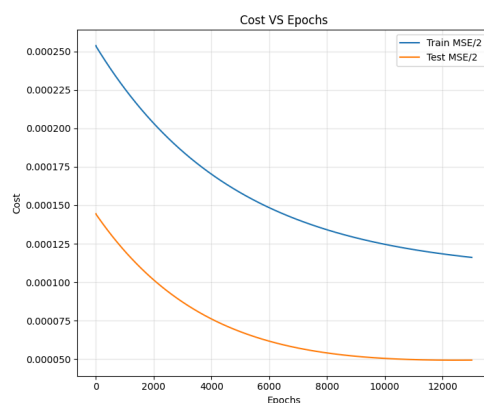


Figura 1.3

La figura 1.3 nos muestra cómo el error cuadrático medio (MSE) disminuye conforme avanza el entrenamiento y pruebas. El error baja rápido en las primeras iteraciones y después se estabiliza, lo que indica que el modelo alcanzó un punto de convergencia donde nuevas iteraciones ya no mejoran significativamente el ajuste.

5b. Predicciones vs Valores Reales

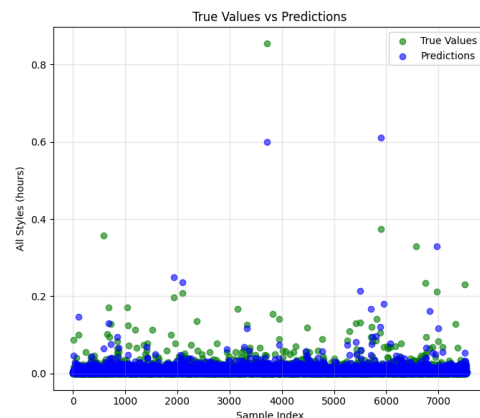


Figura 1.4

La figura 1.4 muestra la comparación entre los valores reales (verde) y los valores predichos (azul) de la variable objetivo en el conjunto de prueba. Aunque no coinciden exactamente, se observa que las predicciones siguen la tendencia general de los datos, lo que refleja que el modelo logra aproximar razonablemente el comportamiento del tiempo de juego promedio.

5c. Métricas

```
=== MÉTRICAS (normalizado [0,1]) ===  
R2(train): 0.5401  
R2(test) : 0.6640
```

Los resultados muestran que el modelo explica alrededor del 54% de la variabilidad en el conjunto de entrenamiento y cerca del 66% en el

conjunto de prueba. Esto indica que el modelo captura parcialmente la relación entre las características y el tiempo de juego, aunque aún deja una proporción considerable de varianza sin explicar.

7. CONCLUSIÓN

Al final, el modelo logra predecir con cierta efectividad el tiempo promedio para completar un videojuego, alcanzando un R^2 de aproximadamente 0.66 en el conjunto de prueba. El hecho de que el R^2 sea ligeramente mayor en prueba que en entrenamiento sugiere que el modelo no sufre de “overfitting”, pero tampoco alcanza un ajuste óptimo.

Lamentablemente, el modelo resultó ser demasiado sencillo y por lo tanto, no nos podrá ayudar a predecir cuánto durará GTA 6.