**Universidad Mayor de San Andres**

**Facultad de Ciencias Puras y Naturales**



**Proyecto**

**Ventas de videojuegos**

**INF - 354**

**Integrantes:**

* Est. Alex Roque Mendoza

**Carrera:** Informática

**Docente:** M.Sc. Moises Silva

**Paralelo:** A

**La Paz - Bolivia**

* **Objetivo principal:**
* Construir un modelo predictivo para las ventas de nuevos videojuegos en función de diferentes características, como el género, la plataforma, el año de lanzamiento comparar los datos obtenidos a los datos dados en el dataset mediante un modelo de regresión lineal múltiple.

**Descripción detallada de de los campos del dataset**

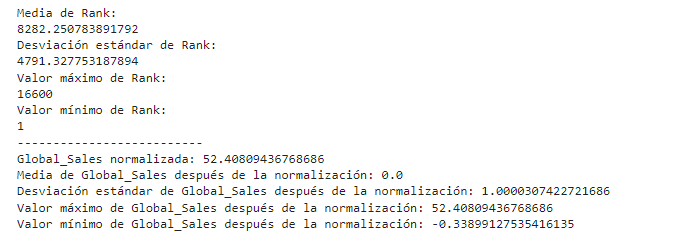
Existen 11 columnas y 16500 filas

|  |  |
| --- | --- |
| Número de columna | Descripción |
| 1(Rank) | El rango o posición del juego en términos de ventas totales. Representa la clasificación del juego en función de sus ventas globales. |
| 2(Name) | Nombre del VideoJuego |
| 3 (Platform) | Plataforma en la que se lanzo el videojuego(por ejemplo,ps4,Xbox One, PC, etc.). |
| 4 (Year\_of\_Release) | Año de lanzamiento del videojuego. |
| 5(Genre) | Género del videojuego (por ejemplo, acción, aventura, deportes, etc.). |
| 6 (Publisher) | Nombre de la empresa editora del videojuego. |
| 7 (NA\_Sales) | Ventas en América del Norte (en millones de unidades). |
| 8 (EU\_Sales) | Ventas en Europa (en millones de unidades). |
| 9(JP\_Sales) | Ventas en Japón (en millones de unidades). |
| 10 (Other\_Sales) | Ventas en otras regiones (en millones de unidades). |
| 11 (Global\_Sales) | Ventas totales a nivel mundial (en millones de unidades). |

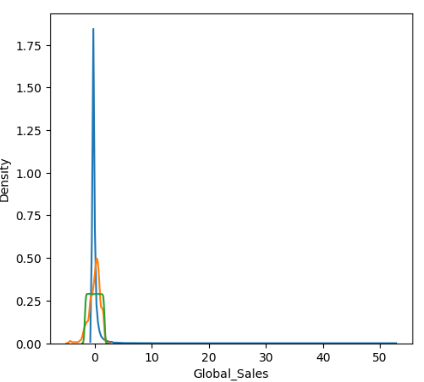
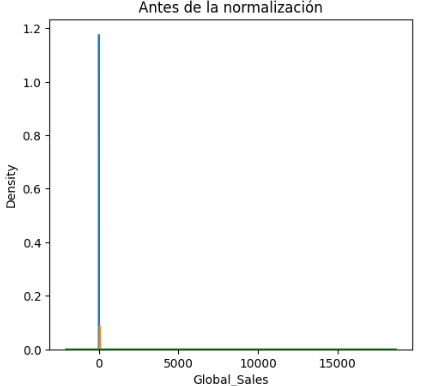
* Preprocesamiento (al menos una válida, otros dos por ver los resultados, si no se aplica justifique porque), Balanceo de datos

<https://github.com/AlexRMendoza989/PROYECTO-INF354/blob/main/normalizacion.ipynb>

Al aplicar normalización para el balanceo de dato se obtiene una clara diferencia en cuantos algunos parámetros como se pueden observar a continuación:



También se observa un cambio importante viendo los gráficos

 **Antes de normalizar Después de normalizar**

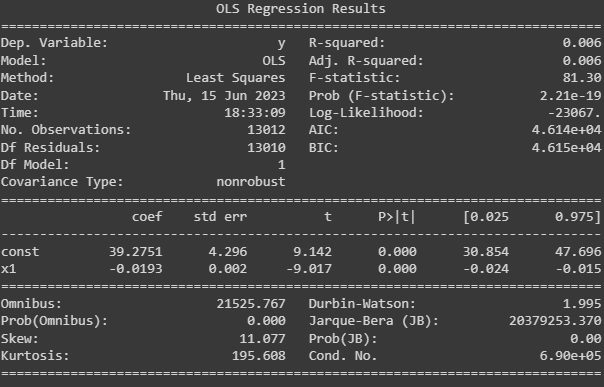
* Selección del clasificador (acorde a los datos supervisado, no supervisado). El clasificador puede, pero no necesariamente depender del preprocesamiento.

Por las razone que se verán a continuación es por esto que no es viable aplicar un modelo de clasificación

|  |  |
| --- | --- |
| Clasificación | Regresión |
| Cuando usamos clasificación, el resultado es una clase, entre un número limitado de clases. Con clases nos referimos a categorías arbitrarias según el tipo de problema. | Cuando usamos regresión, el resultado es un número. Es decir, el resultado de la técnica de machine learning que estemos usando será un valor numérico, dentro de un conjunto infinito de posibles resultados. |

Para tener una mejor comprensión se aplicó 3 modelos de regresión, uno de ellos con el uso de StatsModels

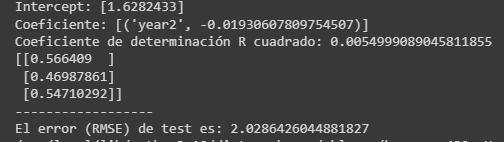
* <https://github.com/AlexRMendoza989/PROYECTO-INF354/blob/main/Prediccion1.ipynb>
* <https://github.com/AlexRMendoza989/PROYECTO-INF354/blob/main/Prediccion2.ipynb>
* <https://github.com/jglaurel/Proyecto-354/blob/main/Proyecto-354/prediccionstatmodels.py>
* <https://github.com/AlexRMendoza989/PROYECTO-INF354/blob/main/prediccionstatmodels.ipynb>



Al usar el modelo de regresión se obtiene los siguientes resultados

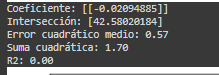
* Primera ejecución: Confiabilidad, matriz de confusión

Para medir la confiabilidad del modelo (robustez) se hace uso de la R cuadrado (coeficiente) de determinación en lugar de una matriz de confusión que es generalmente para modelos de datos binarios o categóricos con 2 respuestas.



De todas forma se aplicó una matriz de confusión la cual es:

* [https://github.com/AlexRMendoza989/PROYECTO-INF354/blob/main/arbol.ipynb](https://github.com/AlexRMendoza989/PROYECTO-INF354/blob/main/arbol.ipynb%20)
* Splits: al menos 100 asignaciones, la mediana de la confiabilidad. Académico (primera ejecucion) 80(train)/20(test) – Investigación 50/50 (segunda ejecución)
* https://github.com/AlexRMendoza989/PROYECTO-INF354/blob/main/Prediccion1.ipynb



* Aplicar Componentes principales (PCA), determinar la cantidad óptima para mejorar o llegar al resultado anterior. Al menos unas 5 ejecuciones. (12 columnas, 10, 11, 9, 5, 3)
* <https://github.com/AlexRMendoza989/PROYECTO-INF354/blob/main/CPA.ipynb>
* **Cómo funciona PCA.-**
* Es un algoritmo matemático para reducir las dimensiones de conjuntos de datos reteniendo la mayoría de la variación de ellos en vectores llamados componentes principales. Las componentes principales son combinaciones lineales no correlacionadas entre sí de las variables originales y maximizan la varianza de las observaciones. Este algoritmo es ampliamente utilizado para identificar patrones en conjuntos de datos con un número dimensiones considerable. La primera componente principal captura la mayor cantidad de la varianza de los datos, la segunda componente captura la segunda mayor cantidad de la varianza, y así sucesivamente. El número de componentes principales que puede ser obtenido de un conjunto de datos es igual a número de dimensiones que éste posea.
  + Calcular la media de cada una de las variables (dimensiones)
  + Restar la media de cada una de las variables a cada una de las observaciones (restar el vector de medias a cada una de las filas -observaciones- de la matriz)
  + Calcular la matriz de covarianza
  + Calcular los eigenvectores y eigenvalores de la matriz de covarianza
  + Graficar componentes principales