UNIVERSIDAD TECNOLOGICA NACIONAL

Tecnicatura Universitaria en Programación
Universidad Tecnológica Nacional.

Trabajo Integrador:

Detección de picos de audiencia mediante árboles de decisión implementados en Python.

PROGRAMACIÓN I

ALUMNOS:

Pablo León - Piuzzi Andrés

Profesora: Cinthia Rigoni

Tutora: Ana Mutti

Comisión 16

09/06/2025

Indice:

1. Introducción

La identificación de picos de audiencia en plataformas de streaming es una tarea clave para analizar el comportamiento de los espectadores e identificar tendencias. En este trabajo se plantea la construcción e implementación de un modelo predictivo basado en árboles de decisión, con el objetivo de determinar cuándo un canal está experimentando un momento de alta audiencia.

2. Marco Teórico

Un árbol de decisión es una estructura jerárquica que representa un conjunto de decisiones a tomar, cada una basada en una condición lógica sobre los datos de entrada. Su funcionamiento se puede describir como una secuencia de preguntas binarias del tipo "sí/no", que dividen iterativamente el conjunto de datos en subconjuntos más homogéneos. Esta división se realiza según una métrica de impureza, siendo la entropía una de las más comunes.

En cada nodo del árbol se evalúa una condición sobre una de las variables predictoras. A partir de esta evaluación, los datos se bifurcan en ramas. El proceso continúa de manera recursiva hasta que se alcanza un criterio de detención, como una profundidad máxima o la homogeneidad total de los datos en una hoja. El resultado final es una serie de hojas terminales que contienen la clase predicha para los datos que lleguen hasta ellas.

3. Caso Práctico

El caso práctico consistió en la aplicación de un árbol de decisión con el objetivo de predecir picos de audiencia en transmisiones en vivo. Estos picos se definieron como aquellos momentos en los que la cantidad de espectadores superaba el percentil 75 dentro del historial del propio canal. La base de datos empleada abarca prácticamente la totalidad de los canales de streaming en Argentina durante el mes de abril. Los datos fueron recolectados mediante técnicas de *scraping*, con una frecuencia de muestreo de cinco minutos por canal, durante las 24 horas del día. Esta metodología garantiza una cobertura amplia y representativa, lo que permite identificar y generalizar tendencias de comportamiento. Las variables incluidas en el conjunto de datos comprenden marcas temporales, número de espectadores, tipo de canal (TV, PRENSA, STREAMING, RADIO) y nombre del canal.

La variable objetivo ("pico" o "no pico") fue determinada dinámicamente para cada canal mediante el cálculo de su cuartil superior. Posteriormente, se procedió al agrupamiento de canales poco frecuentes bajo una categoría genérica (“otros”) para evitar un sobreajuste del modelo debido a etiquetas demasiado específicas.

Uno de los principales desafíos fue el desbalance entre clases: los eventos sin pico eran significativamente más numerosos que los picos. Para compensar esta desigualdad, se utilizó una técnica de submuestreo aleatorio, igualando la cantidad de ejemplos positivos y negativos. Esto permitió entrenar un modelo más equilibrado y representativo.

El modelo **no utiliza directamente la cantidad de espectadores para predecir un pico**, sino que aprende a identificar patrones temporales y contextuales asociados a la ocurrencia de picos. Esto significa que, conociendo solo la hora del día, el día de la semana y el tipo o nombre del canal, el modelo estima la probabilidad de que se presente un pico.

Esta estrategia es valiosa porque permite anticipar picos en escenarios donde no se dispone aún de la información puntual de espectadores, basándose en patrones históricos aprendidos. Por ejemplo, un canal de TV puede tener picos recurrentes a las 21 horas, mientras que un canal de streaming puede tenerlos en horarios distintos; el modelo aprende estas tendencias y las utiliza para clasificar nuevos datos.

4. Metodología Utilizada

El desarrollo del trabajo se estructuró en varias etapas:

- **Preprocesamiento de datos**: Se realizó la conversión de fechas, eliminación de valores nulos, transformación de variables categóricas y normalización horaria. También se filtraron aquellas franjas horarias poco representativas dentro de cada canal.

- **Definición de la variable objetivo**: Se calculó el umbral de pico para cada canal en función del percentil 75 de espectadores, generando una variable booleana indicadora.

- **Transformación de variables**: Las variables categóricas como el tipo de canal y el nombre del canal fueron codificadas mediante variables dummies para su uso en el árbol.

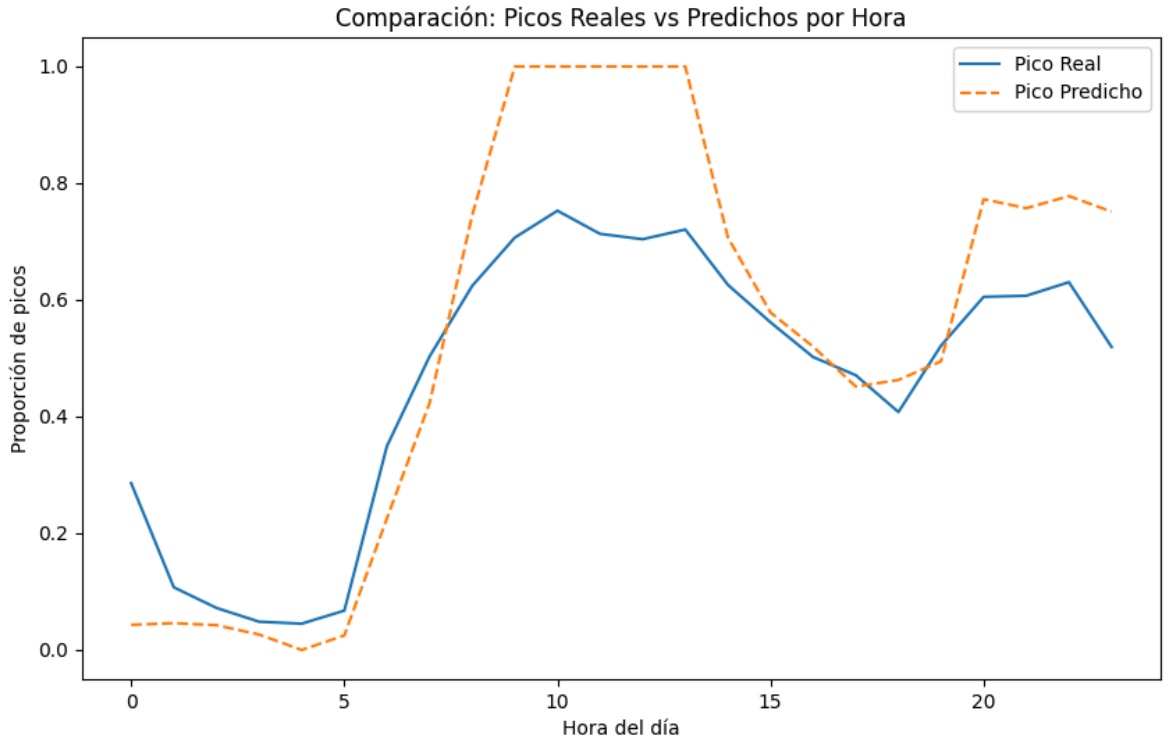
- **Construcción del árbol**: A través de funciones propias, se implementaron los cálculos de entropía, ganancia de información, y un algoritmo recursivo para construir el árbol de decisión. El árbol fue limitado a una profundidad máxima de 6 niveles para evitar un sobreajuste.

- **Evaluación del modelo**: Se realizaron predicciones sobre los mismos datos balanceados y se calcularon métricas manualmente: precisión global, precisión para la clase positiva, recall y F1-score.

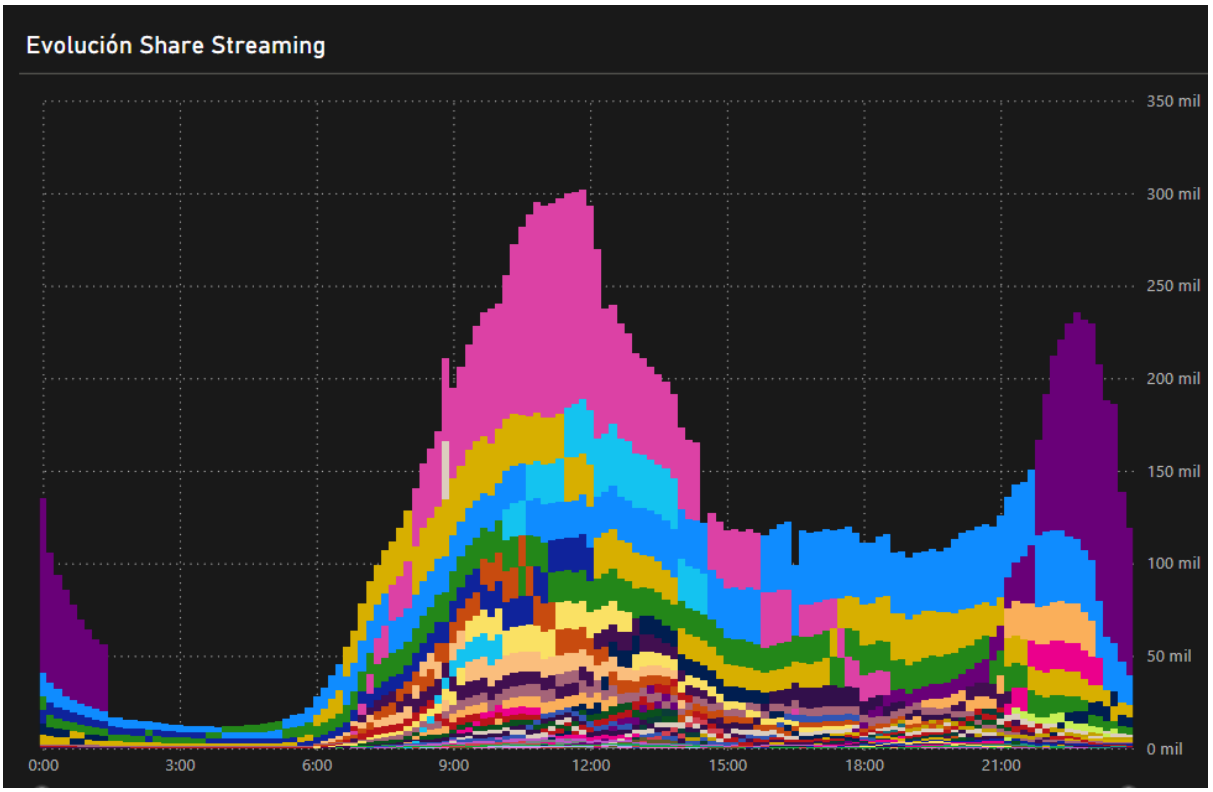
5. Resultados Obtenidos

El modelo logró una **precisión global del 74%**, lo cual indica un desempeño razonable en la clasificación de picos de audiencia. Las métricas específicas para la clase “pico” muestran un comportamiento balanceado entre precisión y recall, evidenciando que el modelo no solo predice correctamente muchos de los picos, sino que también mantiene un bajo nivel de falsos positivos.

El análisis descriptivo del modelo muestra una concentración recurrente de picos de audiencia en dos franjas horarias: entre las 10:00 y las 12:00, y entre las 21:00 y las 23:00 horas. Este patrón se corrobora con un análisis posterior realizado el día 4 de junio, donde se observa que la sumatoria de espectadores simultáneos en los canales también alcanza sus valores máximos en esas mismas franjas. Esto sugiere que dichas ventanas horarias concentran los momentos de mayor actividad en términos de audiencia, más allá de las predicciones del modelo.



Picos a partir de los datos originales comparado con los resultados del modelo



Sumatoria de espectadores por canal de streaming. Datos del 04/06/2025

6. Conclusiones

El desarrollo de este proyecto permitió no solo aplicar un modelo de árbol de decisión, sino también comprender en profundidad su lógica de construcción a partir de funciones propias. Esta aproximación, a diferencia de utilizar bibliotecas como *sklearn*, permite visualizar cada paso del proceso de decisión y analizar detalladamente cómo se comporta el modelo ante distintos escenarios.

No obstante, el resultado alcanzado (74% de precisión) sugiere que hay margen para mejorar. En particular, el proceso de **preprocesamiento de datos** podría afinarse para proporcionar un mejor soporte al modelo. Algunas estrategias que podrían explorarse incluyen:

- Incorporar nuevas variables derivadas del comportamiento histórico del canal.
- Ajustar el umbral de clasificación de picos utilizando métodos más sofisticados (por ejemplo, técnicas de clustering o percentiles adaptativos).
- Probar otros esquemas de codificación para variables categóricas que conserven más información (por ejemplo, embeddings o codificación de frecuencia).
- Ampliar el conjunto de entrenamiento con técnicas de sobremuestreo en lugar de submuestreo para no perder información valiosa.

En definitiva, si bien se logró un funcionamiento correcto y se cumplió con el objetivo principal del trabajo, este caso evidencia cómo la calidad del preprocesamiento y la elección de variables impactan directamente en la capacidad predictiva del modelo. Las futuras mejoras deberían orientarse a enriquecer los datos de entrada para aprovechar aún más el potencial de este tipo de algoritmos.

7. Bibliografía

A continuación, se enumeran las principales fuentes teóricas y técnicas consultadas durante el desarrollo del trabajo:

- Documentación oficial de Python: https://docs.python.org/3
- Pandas Documentation: https://pandas.pydata.org/docs/.
- Numpy Documentation: https://numpy.org/doc/.
- Tecnicatura Universitaria en Programación I.

8. Anexos:

🔗 **Video de presentación:**

- https://youtu.be/5XGH7UwnH4w

🔗 **Link al repositorio en GitHub:**

- https://github.com/pablol314/programacion_tp_integral