Predicción de Precios de SUVs Utilizando Machine Learning

Agustin Manzano
Departamento de Ingeniería
Universidad de San Andrés
Buenos Aires, Argentina
amanzano@udesa.edu.ar

Máximo Simian
Departamento de Ingeniería
Universidad de San Andrés
Buenos Aires, Argentina
msimian@udesa.edu.ar

***Abstract*—Este documento presenta un estudio detallado sobre la predicción de precios de SUVs utilizando técnicas de Machine Learning. Se analiza un dataset real de vehículos extraído de Mercado Libre, con el objetivo de desarrollar modelos predictivos precisos. Los modelos que mejor desempeño mostraron, fueron XGBoost y el Random Forest, ambos modelos que utilizan arboles y tienden a ser precisos cuando se trata con datos tabulares.**

***Index Terms*—Machine Learning, SUVs, Predicción de Precios, Optimización de Modelos**

I. Introducción

El presente trabajo tiene como objetivo principal desarrollar modelos predictivos para estimar los precios de venta de SUVs utilizando técnicas de Machine Learning. Se utilizó un dataset obtenido mediante scrapping extraído Mercado Libre, el cual contiene diversas características de los vehículos como modelo, año, kilometraje, entre otros. La tarea consiste en entrenar y evaluar modelos que puedan predecir de manera precisa el precio en dólares de nuevos vehículos basados en estas características.

II. Metodología

A. *Limpieza del Dataset*

El dataset inicial presentaba varios desafíos, como errores humanos y falta de estandarización en los datos proporcionados por los vendedores. Se realizó un proceso exhaustivo de limpieza y preprocesamiento:

- Conversión de precios a dólares: Utilizando el tipo de cambio del día de extracción del dataset, se convirtieron todos los precios de pesos a dólares para estandarizar la moneda.
- Filtrado por año de fabricación: Se eliminaron las muestras con años de fabricación inválidos, es decir, posteriores al año actual.
- Creación de nuevas características: Se agregaron nuevas columnas como la edad del vehículo a partir del año de fabricación.
- Eliminación de columnas irrelevantes: Se identificaron y removieron columnas como 'Puertas' y 'Tipo de Carrocería', que no aportaban información relevante debido a la naturaleza homogénea de los SUVs en el dataset.

Una de los enfoques principales utilizados para detectar anomalías y poder corregirlas fue utilizar convoluciones '1-D', donde las características válidas fueron encodeadas en 'kernels' que luego, con la operación de convolución, detectábamos si las características estaban presentes en cada muestra. Utilizamos este enfoque en casos como:

- Motor: Detectamos que muchas muestras tenían cargados valores erróneos en el campo 'Motor', por lo que optamos por corregirlas buscando el valor de la cilindrada en el campo 'Versión', donde mayormente estaba declarado.
- Colores: Detectamos que algunos colores eran clasificados como distintos, cuando en realidad eran el mismo color, sólo que fue cargado de distinta forma (ej: 'Grey','Gris' y 'gris' son el mismo color, pero son tomados como distintos 'strings'). Creamos kernels específicos para cada color (retomando el ejemplo, el kernel en ese caso fue 'gr').

Todo este trabajo fue llevado a cabo con el objetivo principal de no perder un porcentaje considerable de muestras, ya que esto posiblemente desembocaría en una *performance* peor para nuestro modelo a la hora de entrenar. Partiendo de un dataset con 22377 muestras, luego de limpiar los datos corruptos, la perdida fue de 84 muestras, en torno a un 0.38%.

B. *Feaeture Engineering y Oversampling*

Una vez obtenido el dataset limpio y completamente numérico, se realizo una instancia de *feature engineering*, en la cual, mediante combinaciones de ciertas columnas, obtuvimos nuevas características para las muestras. Algunas son simplemente comprobar si cierta feature es mayor a un numero limite, mientras que otras son ratios o multiplicaciones de columnas. Algunas de las columnas nuevas son:

- Alto kilometraje (si el auto tiene mas de 100.000 km)
- Es viejo (si el auto tiene mas de 10 años)
- Km por cilindro (ratio)

Lo que buscamos lograr mediante este proceso, es generar nuevas características para las muestras y así proveer a los modelos con mayor información sobre cada auto.

Por otro lado, al revisar el dataset, surgió el problema de que hay modelos de autos que tienen una baja cantidad de muestras en comparación a otros. Precisamente por esto se aplico un

algoritmo de *oversampling*, para así generar mas muestras de los modelos minoritarios. Mas específicamente, el algoritmo utilizado, toma todos los modelos con una representación menor a 1200 muestras y mediante una selección random, duplica muestras hasta llegar a un umbral de 1500. Una vez realizado esto, la cantidad de muestras por modelo de auto es mas similar, dándole a los modelos una cantidad de muestras mucho mayor a la inicial. Mas concretamente, de 22293 a 245959. Mas de 10 veces mas muestras que al principio.

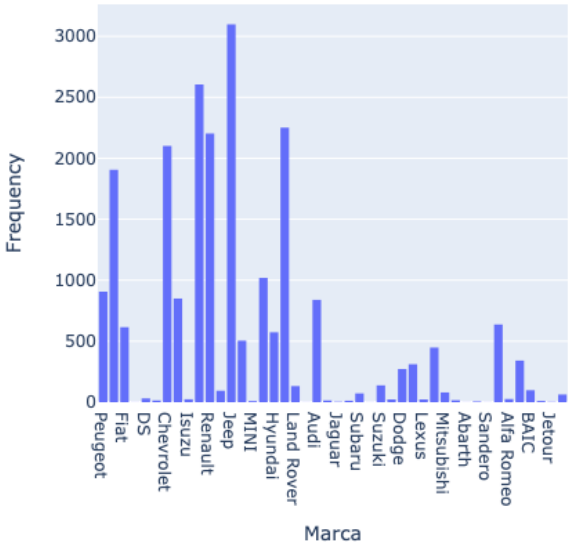

Fig. 1. Distribución de SUV's por marcas

En la figura anterior, podemos observar que hay marcas con mucha diferencia de cantidad de autos, por lo que aplicar oversampling ayuda a cubrir toda esa falta de muestras y proporcionar, no mas información, sino que hacer que los modelos tengan mas en cuenta a las muestras con menos apariciones.

C. *Desarrollo de Modelos*

Se exploraron varios modelos de Machine Learning para predecir los precios de los SUVs. Tuvimos un enfoque principal en modelos donde su estructura principal se basa en 'arboles de decisión', ya que empíricamente presentan una mejor performance para datos tabulares. Entre ellos, se destacó el modelo XGBoost, que demostró ser el más efectivo en términos de rendimiento predictivo. A continuación se detallan las técnicas de optimización empleadas:

- **Regresión Lineal:** Se implementó un modelo de regresión lineal como punto de partida. Aunque su rendimiento no fue el mejor, proporcionó una línea base para comparar con otros modelos más complejos.
- **Neural Network Model:** Se desarrolló un modelo de red neuronal con capas densas para capturar las relaciones complejas en los datos. Sin embargo, debido a que el desarrollo tuvo desde el inicio un enfoque en estructuras con árboles, no avanzamos con el hecho de
- **XGBoost Regressor:** El modelo XGBoost demostró ser el más efectivo. Se optimizaron hiperparámetros clave como la tasa de aprendizaje, la profundidad máxima del árbol, el número de estimadores y otros parámetros utilizando técnicas de optimización bayesiana ('*BayesSearchCV*').
- **Random Forest Regressor:** Este modelo se utilizó para capturar las relaciones no lineales entre las características y el precio de los SUVs. Se optimizaron parámetros como el número de árboles en el bosque y la profundidad máxima de los árboles utilizando '*BayesSearchCV*'.

D. *Optimización Bayesiana*

La optimización bayesiana es una técnica de optimización de hiperparámetros que utiliza un enfoque probabilístico para encontrar la configuración óptima de los parámetros. En lugar de probar todas las combinaciones posibles como en un GridSearchCV, la optimización bayesiana construye un modelo probabilístico del problema de optimización y utiliza este modelo para seleccionar las configuraciones de hiperparámetros más prometedoras.

- Construcción de un Modelo Surrogado: Se utiliza un modelo surrogado (como un proceso de Gauss o un modelo de regresión) para aproximar la función de pérdida basada en las configuraciones de hiperparámetros ya evaluadas.
- Selección de los Siguientes Puntos a Evaluar: Se seleccionan los puntos más prometedores del espacio de hiperparámetros mediante una función de adquisición que equilibra la explotación (probar configuraciones cercanas a las mejores conocidas) y la exploración (probar nuevas configuraciones).
- Actualización del Modelo: Después de evaluar los puntos seleccionados, el modelo surrogado se actualiza con los nuevos resultados, y el proceso se repite hasta que se alcanza un criterio de parada (como un número máximo de iteraciones).

III. Resultados y Análisis

El modelo XGBoost optimizado mostró un rendimiento superior en comparación con otros algoritmos evaluados. Se utilizaron métricas estándar como el Error Absoluto Medio (MAE), el Error Cuadrático Medio (MSE), la Raíz del Error Cuadrático Medio (RMSE) y el Coeficiente de Determinación (R2) para evaluar su desempeño. Los resultados de error de validación fueron los siguientes:

- MAE: 695.959
- MSE: 3941045.420
- RMSE: 1985.206
- R2: 0.997

Por otro lado, el modelo de regresión lineal, si bien resulto inferior a XGBoost, presento resultados que en un caso como este, se podrían considerar aceptables o incluso buenos. Pero dado que es un modelo ampliamente mas simple, es de esperar que no obtenga los mejores resultados. De todas formas lo consideramos como una buena base contra la cual

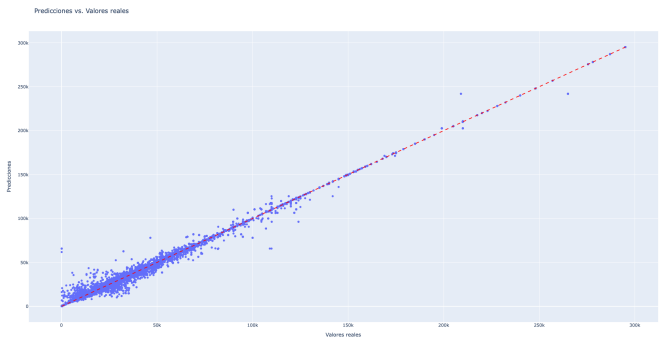

Fig. 2. Predicciones vs Real - XGBoost

evaluar nuestros otros modelos. Cualquier modelo inferior a la regresión, probablemente no nos sea de utilidad.

- MAE: 697.686
- MSE: 161850541.791
- RMSE: 12722.049
- R2: 0.895

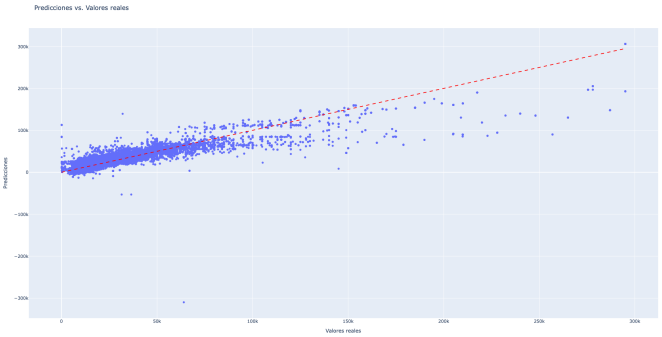

Fig. 3. Predicciones vs Real - Regresión Lineal

En el caso de la red neuronal, los resultados fueron altamente inferiores a los del resto de modelos. Dado que las redes neuronales necesitan de muchos mas datos que los arboles o XGBoost, es de esperar que con un dataset de esta escala no de buenos resultados. Además de esto, las redes neuronales funcionan mejor con datos no estructurados, mientras que los arboles son especialmente buenos con datos tabulares, ya que detectan mejor ciertos patrones y relaciones entre los datos.

- MAE: 5931.849
- MSE: 127350026.231
- RMSE: 11284.946
- R2: 0.670

Por ultimo, el random forest, dio resultados muy similares a los de XGBoost. Dado que las métricas son tan similares, se hace difícil determinar cual de los dos es mejor, ya que estos pueden variar sutilmente. Dado que ambos modelos utilizan arboles, una performance similar era de esperarse con una elección adecuada de hiperparametros.

- MAE: 410.419
- MSE: 3494576.506
- RMSE: 1869.378
- R2: 0.997

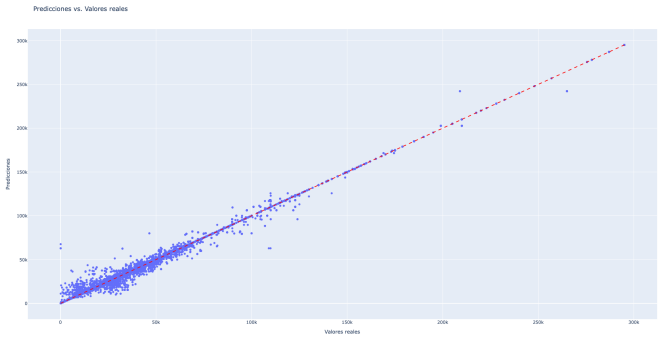

Fig. 4. Predicciones vs Real - Random Forest

IV. Conclusiones

En este trabajo, se desarrollaron y evaluaron varios modelos de Machine Learning para predecir los precios de SUVs utilizando un dataset real de Mercado Libre. Se concluye que tanto el modelo XGBoost optimizado como el Random Forest son igualmente precisos para esta tarea, demostrando una alta capacidad para manejar relaciones complejas entre las características del vehículo y su precio. Aunque se identificaron ciertas limitaciones en otros enfoques, como la regresión lineal y las redes neuronales, el uso de árboles de decisión optimizados probó ser altamente efectivo. Este estudio sugiere que la incorporación de técnicas avanzadas de ingeniería de características y modelos de ensamble puede seguir mejorando la precisión de las predicciones. Además, destaca la importancia de un exhaustivo preprocesamiento y limpieza de datos para lograr un rendimiento óptimo en los modelos predictivos.

A. *Trabajo a Futuro*

Se sugiere investigar técnicas avanzadas de feature engineering y explorar modelos ensamble para mejorar aún más la precisión de las predicciones. Además, sería beneficioso explorar la aplicación de redes neuronales profundas (DNN) para capturar relaciones aún más complejas en los datos.

B. *Agradecimientos*

Agradecemos a nuestros profesores tutores por su apoyo y orientación en el desarrollo de este trabajo.

References

[1] Bishop, C. M. (2023). Deep Learning: Foundations and Concepts. Springer.