Modelo de predicción de precios de segunda mano

1. Introducción

Este proyecto tiene como objetivo desarrollar un modelo de Machine Learning para predecir el precio de vehículos usados, enfocado a una empresa de reventa de coches. El propósito es proporcionar una herramienta que ayude tanto a los usuarios que venden sus coches como a los que compran, dándoles una estimación precisa de los precios de compra y venta.

Se parte de un conjunto de datos que contiene información sobre diferentes características de los vehículos.

El enfoque del proyecto incluye el análisis exploratorio de datos (EDA), preprocesamiento, selección y ajuste de modelos, evaluación de resultados, y la interpretabilidad del modelo final.

2. Datos utilizados

El conjunto de datos contiene las siguientes variables:

- Características categóricas: marca, modelo, tipo de gasolina, color, tipo de coche, entre otros.
- Características numéricas: Kilómetros recorridos, potencia, precio, fecha de registro y fecha de venta.

La variable objetivo es el precio del coche, y el conjunto de datos cuenta con registros sobre coches de diferentes modelos y especificaciones técnicas.

3. Análisis Exploratorio de Datos (EDA)

3.1. Información general del DataFrame

Al inicio, el DataFrame (a partir de ahora, DF) está formado por un total de 18 variables y 4843 observaciones. No tiene observaciones duplicadas pero si valores nulos.

3.2. Distribución de variables numéricas y categóricas

Se exploraron las distribuciones, tanto de las variables numéricas como de las categóricas. También se realiza un primer análisis de los outliers y nulos.

3.3. Errores en variables string

Analizamos los valores de las variables string para localizar errores de escritura. En la variable *tipo_gasolina* encontramos "*Diesel"* y *"diesel"*. Lo corregimos.

4. Preprocesamiento de los Datos

4.1. Valores nulos

Se analiza cada una de las variables para determinar cuál es la mejor estrategia de imputación de nulos para cada una:

- Para las variables *modelo*, *tipo_gasolina*, *color*, *tipo_coche*, *volante_regulable*, *aire_acondicionado*, *cámara_trasera*, *asiento_traseros_plegables*, *elevalunas_electrico*, *bluetooth* y *alerta_lim_velocidad* se imputa la categoría *"unknown"*.
- Para las variables *km*, *potencia* y *precio* se imputa la media, tras haber analizado los outliers.

4.2. Análisis de outliers

Se realizaron análisis de outliers utilizando boxplots para identificar valores atípicos en variables clave como *km*, *potencia* y *precio*.

- En *km* se eliminaron las observaciones con más de 100.000km (solamente 1).
- En *potencia* se estudian las observaciones que tienen un valor de 0. En este caso, se imputa la media.
- Para *precio,* eliminamos las observaciones cuyo precio supera el valor 100.000.

4.3. Data engineering

A partir de las columnas *fecha_registro* y *fecha_venta* hemos creado la nueva variable *antiguedad*, resultante de la diferencia de las dos columnas fecha.

La variable *antiguedad* está expresada en años. Tratamos los outliers, ya que no puede haber valores negativos. Los outliers y nulos se imputan con el valor -1.

Eliminamos las variables *fecha_registro* y *fecha_venta* del dataset.

4.4. Codificación de variables categóricas

Para las variables booleanas (las cuales ahora mismo tienen 3 valores; *true, false* y *unknown)* asignamos los valores 1, 0 y -1.

Las variables categóricas con muchas categorías fueron agrupadas en clases más generales para reducir la cardinalidad.

- *modelo*: agrupamos los modelos que suponen menos del 1% del total de modelos en el valor "*Otro modelo"*.

- *tipo_gasolina*: agrupamos los tipos de gasolina, de manera que finalmente separamos en 1 para diésel y 0 para otro tipo de gasolina.
- *color*: agrupamos los colores que suponen menos del 5% del total de colores en el valor "*Otro color"*.
- *tipo_coche*: agrupamos los tipos de coche que suponen menos del 10% del total de tipos de coche en el valor "*Otro tipo"*.

Finalmente, se realizó One-Hot Encoding para las variables categóricas restantes (*modelo, color, tipo_coche)*.

4.5. Análisis target vs features

Marcamos *precio* como target y analizamos la dispersión del objetivo con cada una de las variables.

Estudiamos la relación entre las variables y con el objetivo, buscando aquellas con una correlación más alta de 0,95 para eliminarlas. También miramos las variables con varianza nula.

4.5. Escalado de datos

Se utilizó StandardScaler para estandarizar las variables numéricas, con el fin de que los algoritmos que lo requieran puedan trabajar de manera más eficiente.

5. Selección y evaluación de modelos

5.1. Separamos train y test

Decidimos que el tamaño de train supongo el 70% del dataset, es decir, 3.388 observaciones. Test supone un 30% (1.452 observaciones).

Analizamos las diferencias en la distribución de la variable objetivo (*precio*) tanto en train como en test.

5.2. Torneo de modelos

Con una función creada para evaluar los distintos algoritmos, comenzamos el torneo de modelos utilizando validación cruzada. Se utilizó el RMSE (Root Mean Squared Error) como métrica principal de evaluación para todos los modelos, tanto en el conjunto de train como en el de cross-validation.

Se evaluaron múltiples modelos de regresión, incluyendo:

- Regresión Lineal
- Lasso
- Ridge
- ElasticNet
- Stochastic Gradient Descent

- Gaussian Process Regressor
- KNeighborsRegressor
- Random Forest
- XGBoost Regressor

5.3. Mejores Modelos

De todos los modelos probados, el modelo de XGBoost Regressor resultó ser el mejor en términos de RMSE, mostrando un excelente rendimiento tanto en el conjunto de entrenamiento (RMSE = 1.247,93) como en validación cruzada (RMSE = 4.038,47).

5.4. Hiperparametrización

Se realizó una búsqueda de hiperparámetros utilizando RandomizedSearchCV para mejorar el rendimiento del modelo de XGBoost. Los parámetros ajustados incluyeron:

- n_estimators
- max_depth
- learning_rate
- subsample
- gamma
- colsample_bytree

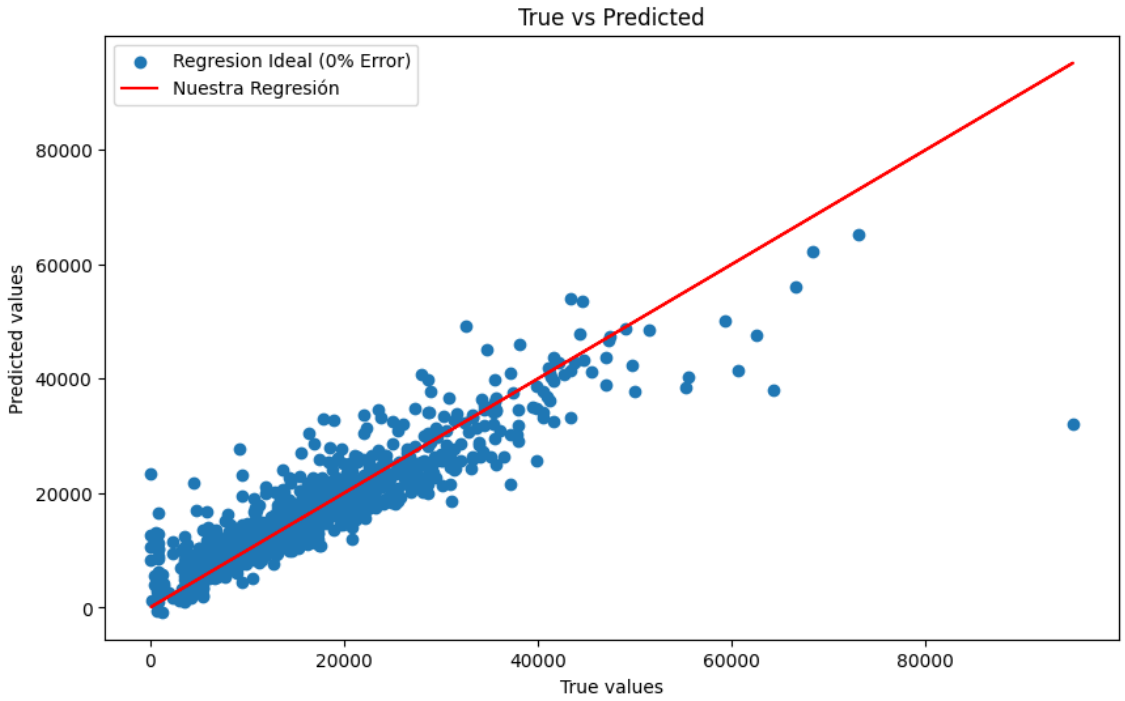
El mejor modelo fue seleccionado según el RMSE obtenido durante la validación cruzada (RMSE = 3.650,45).

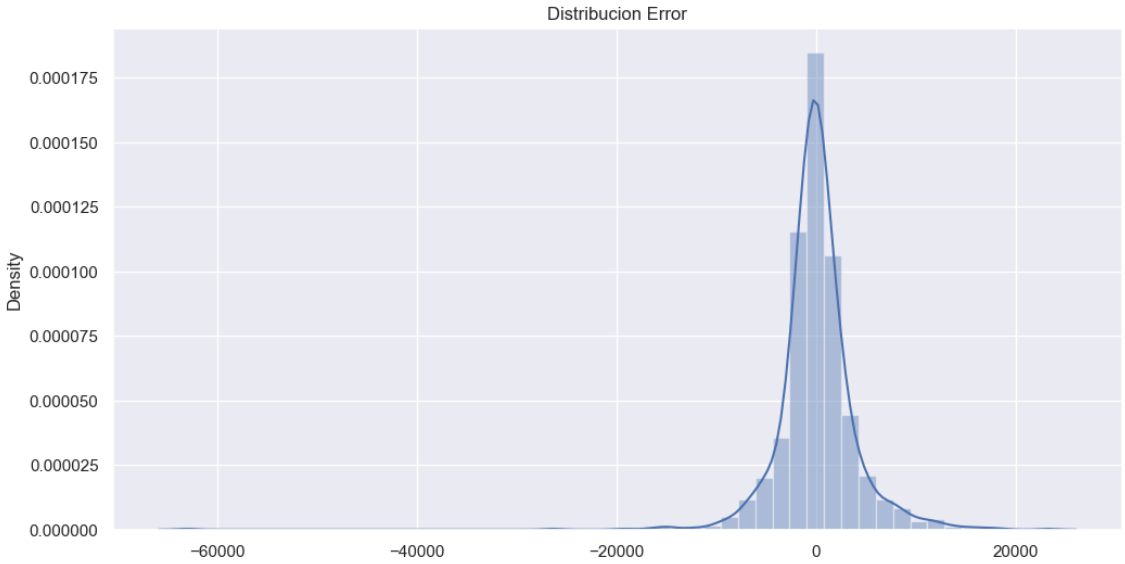
6. Evaluación en el conjunto de test

El mejor modelo, XGBoost con hiperparámetros ajustados, fue evaluado en el conjunto de test.

- RMSE en test: se calculó el RMSE para el conjunto de test, obteniendo un valor final de error que demuestra la robustez del modelo (RMSE = 4.013,935).
- R2-Score: se obtuvo un valor de R2 (qué porcentaje de la variabilidad en el precio es explicada por el modelo) del 80.82%.

Se graficó la comparación entre los valores reales y los predichos, y se analizó la distribución del error, mostrando que el modelo generaliza bien en el conjunto de test.



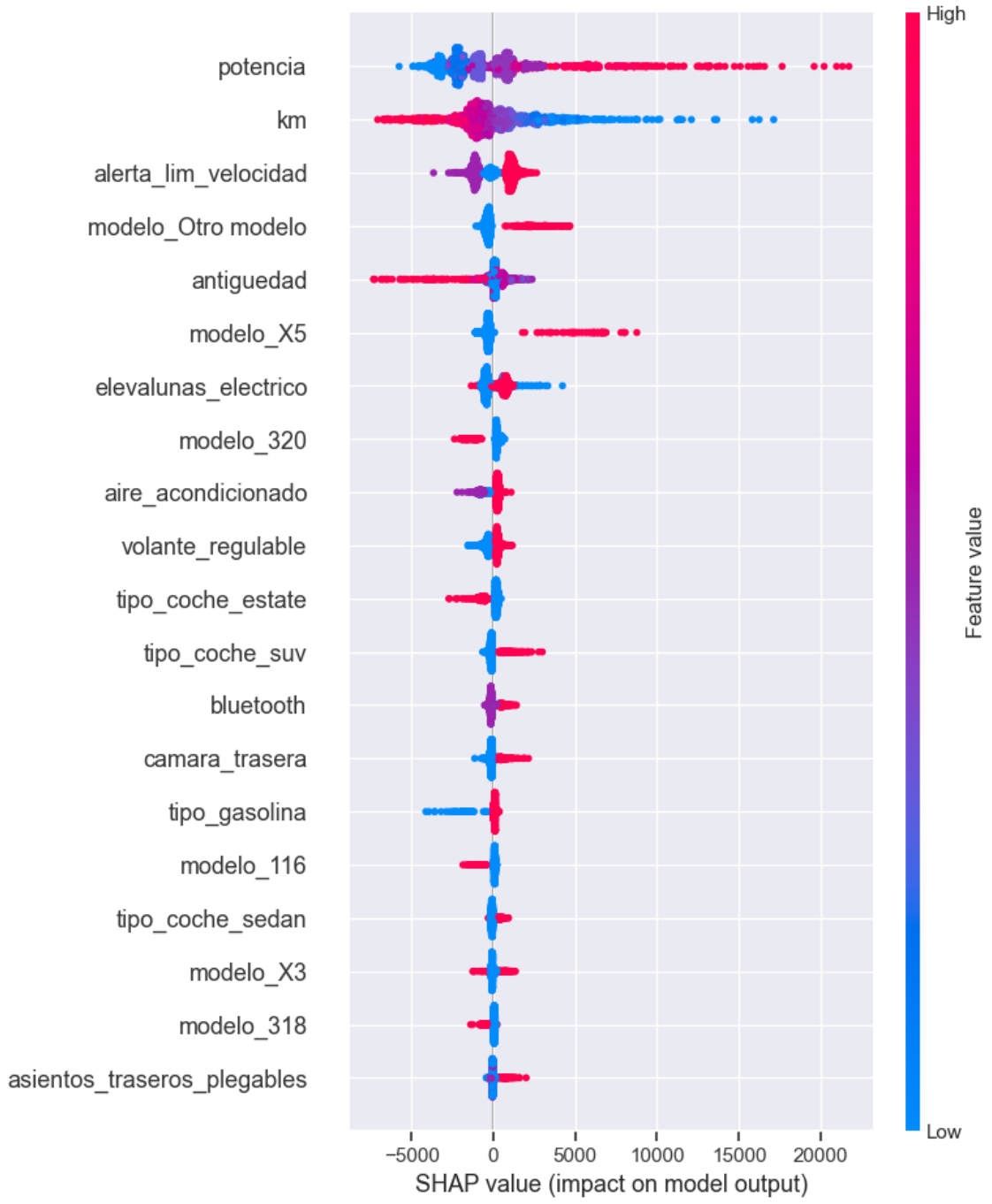


Por último, para una mayor comprensión del cliente, damos un intervalo de confianza del 95% del error (-6.957,8 ; 8.433,6).

7. Interpretabilidad del modelo

En primer lugar, utilizamos *plot_importance* de XGBoost para evaluar cuáles son las características que más peso tienen en la predicción, resultando *modelo_X5*, *potencia, modelo_Otro_modelo* y *alerta_lim_velocidad*.

A continuación, se utiliza SHAP para interpretar el modelo y entender cómo las características individuales contribuyen a las predicciones. Las características más importantes según los valores SHAP son *potencia* (a mayor potencia, mayor precio, y viceversa) y *km* (a menos km más precio y viceversa).



Esto proporciona una visión clara de las variables que más afectan el precio de venta de un coche en este contexto.

8. Conclusión y Próximos Pasos

El proyecto ha logrado desarrollar un modelo predictivo de alta precisión para estimar el precio de coches usados, basándose en las características del vehículo. El modelo de XGBoost, optimizado mediante hiperparametrización, resultó ser el más adecuado.

Próximos pasos:

- Implementación del modelo en producción para que los usuarios de la plataforma puedan obtener estimaciones en tiempo real.
- Recolectar más datos sobre el historial de ventas de coches para seguir refinando el modelo.
- Explorar modelos adicionales de regresión o incluso técnicas de deep learning para mejorar aún más la precisión, así como estudiar la eliminación de las variables con menos peso en el modelo para mejorar la eficiencia sin perder precisión.