Módulo de especialización en Big Data & IA

HousePriceAI: Predicción Inteligente de Precios de Vivienda

Profesor: Sinesio David Carvajal Tabasco

Alumno: Franco Leonardo Ramírez Castro

Centro: Universidad Alfonso X El Sabio

Resumen

Este proyecto tiene como objetivo desarrollar un modelo de Machine Learning para predecir el precio de venta de viviendas, siguiendo la metodología CRISP-DM. Se realizó un Análisis Exploratorio de Datos (EDA) para limpiar y transformar las variables, reduciendo de 81 a 9 características clave mediante técnicas de selección de atributos. Se probaron Random Forest, XGBoost y LightGBM, seleccionando XGBoost optimizado ($R^2 = 0.8939$) tras ajustar hiperparámetros clave. La evaluación del modelo incluyó validación cruzada ($R^2 = 0.8174$), análisis de errores y métricas como RMSE y MAE, identificando un leve sesgo en precios altos. Para el despliegue, se guardó el modelo en formato .pkl, asegurando su reutilización con nuevos datos mediante un pipeline de transformación. Se generaron predicciones y se compararon con valores esperados para verificar su precisión. El modelo está listo para producción, con potencial de implementación en una API para automatizar la tasación de viviendas. Próximos pasos incluyen mejorar la precisión en diferentes rangos de precios y explorar más variables relevantes. Este proyecto demuestra el valor del Machine Learning en el sector inmobiliario, proporcionando una herramienta confiable y eficiente para la estimación de precios.

Metodogía

La metodología CRISP-DM (Cross Industry Standard Process for Data Mining) es un modelo estándar abierto que describe los enfoques comunes utilizados por los expertos en minería de datos. Divide el proceso de minería de datos en seis fases principales:

1. Comprensión del negocio: Esta fase se centra en entender los objetivos y requisitos del proyecto desde una perspectiva empresarial, para convertir este conocimiento en una definición técnica del problema.
2. Comprensión de los datos: Implica la recopilación inicial de datos y el inicio de actividades para familiarizarse con ellos, identificando problemas o conjuntos interesantes. Wikipedia Portugués+1Wikipedia+1
3. Preparación de los datos: Consiste en construir el conjunto de datos final a partir de los datos iniciales. Normalmente, esta fase se realiza varias veces en el proceso.
4. Modelado: En esta fase, se seleccionan las técnicas de modelado más apropiadas para el proyecto de minería de datos específico y se construyen los modelos.
5. Evaluación: Una vez construidos los modelos, se evalúan para asegurar que cumplen con los objetivos del negocio y se identifican los próximos pasos a seguir.
6. Despliegue: El conocimiento adquirido se organiza y presenta de una manera que el cliente pueda utilizar, lo que puede implicar la implementación de sistemas automatizados analíticos, predictivos y/o prospectivos.

Estas fases no son necesariamente secuenciales y pueden requerir iteraciones, permitiendo mejorar la aproximación obtenida en fases anteriores.

1. COMPRESIÓN DE NEGOCIO

En este proyecto, el objetivo principal es predecir el precio de venta de viviendas utilizando técnicas de aprendizaje automático. Este caso de uso es fundamental en sectores como bienes raíces, banca y aseguradoras, ya que permite estimar con precisión el valor de las propiedades basándose en características clave del inmueble.

1.1 Objetivos de Negocio:

- Automatizar la valuación de propiedades para facilitar decisiones de compra, venta e inversión.
- Optimizar el proceso de tasación mediante modelos de IA, reduciendo la dependencia de evaluaciones manuales.
- Mejorar la precisión de las estimaciones para evitar sobrevaloraciones o subvaloraciones en el mercado inmobiliario.

1.2 Beneficios Esperados:

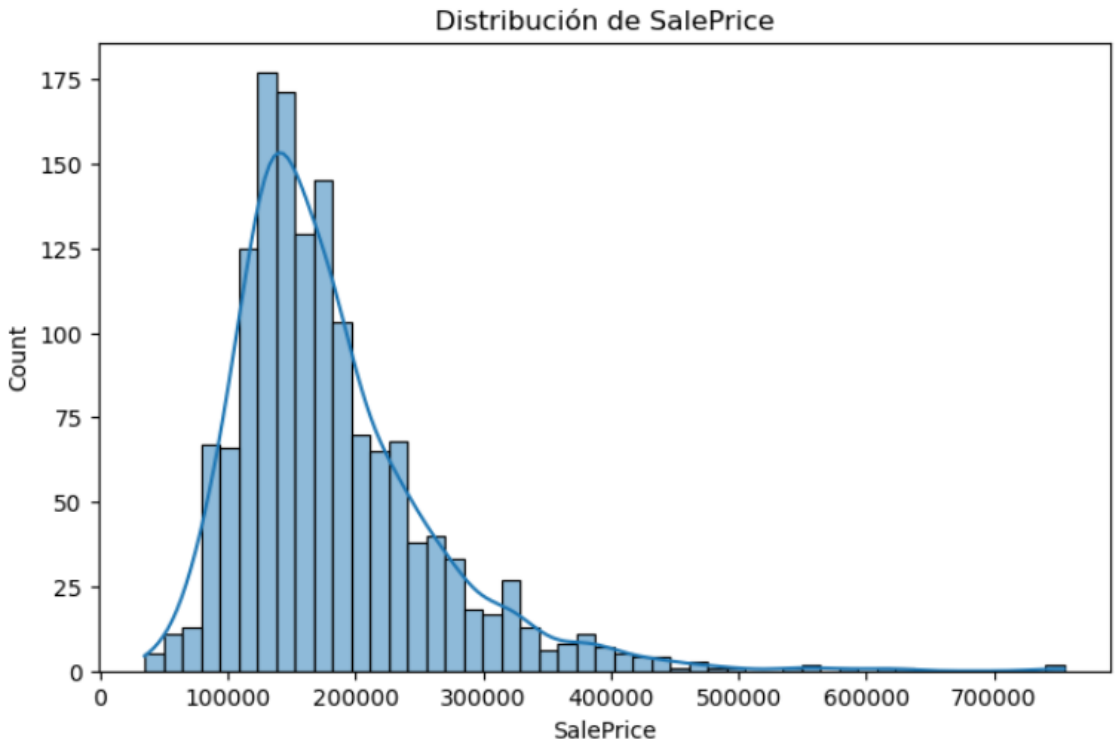
- Mayor eficiencia y rapidez en la tasación de inmuebles.
- Reducción de errores humanos en la estimación de precios.
- Herramienta escalable que puede aplicarse en distintos mercados inmobiliarios.
- Facilitación de decisiones de inversión basadas en datos precisos.

2. ENTENDIMIENTO DE LOS DATOS

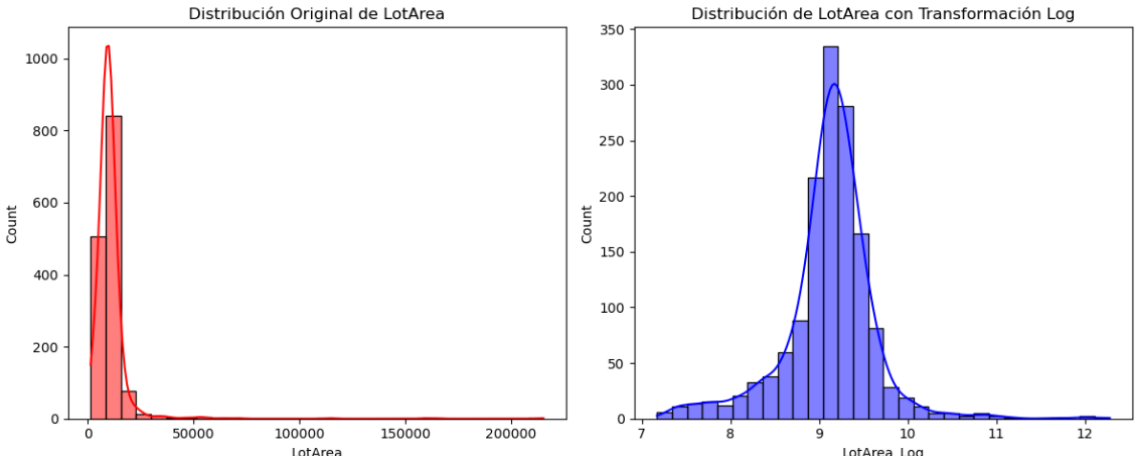
Para asegurar un modelo predictivo preciso, realizamos un Análisis Exploratorio de Datos detallado. Esta fase permitió identificar patrones, detectar valores atípicos y comprender la estructura de los datos antes de proceder a su limpieza y transformación.

2.1 Evaluación Inicial del Dataset

Se analizaron 81 variables en el dataset original, compuesto por variables categóricas y numéricas. Se identificó la variable objetivo: SalePrice (precio de venta de la vivienda). Se detectó una distribución sesgada en SalePrice, lo que motivó el uso de transformaciones.



Distribución después de la transformación logarítmica.



2.2 Manejo de Valores Nulos

Se encontraron varias columnas con valores nulos y se aplicaron estrategias diferenciadas:

- Imputación con cero (0): Para columnas que indicaban la ausencia de una característica (ejemplo: GarageCars, TotalBsmtSF).
- Imputación con la mediana: Para variables con distribuciones sesgadas (GrLivArea, 1stFlrSF).
- Conversión de valores nulos a "Sin información": En variables categóricas como PoolQC, Fence y Alley.

2.3 Análisis Univariado

Se analizaron distribuciones y sesgos en las principales variables predictoras (OverallQual, GrLivArea, TotalBsmtSF). Se aplicaron transformaciones

logarítmicas en variables con alta asimetría. Se detectaron valores atípicos (outliers) en variables como LotArea y GrLivArea, evaluando su impacto en el modelo.

2.4 Análisis Bivariado y Correlaciones

Se calcularon coeficientes de correlación entre SalePrice y otras variables. Se eliminaron características con baja importancia predictiva para reducir dimensionalidad. Se evaluaron relaciones no lineales entre algunas variables y la variable objetivo.

Esta fase permitió comprender la estructura del dataset, sus puntos críticos y las transformaciones necesarias antes del modelado.

3. PREPARACIÓN DE LOS DATOS

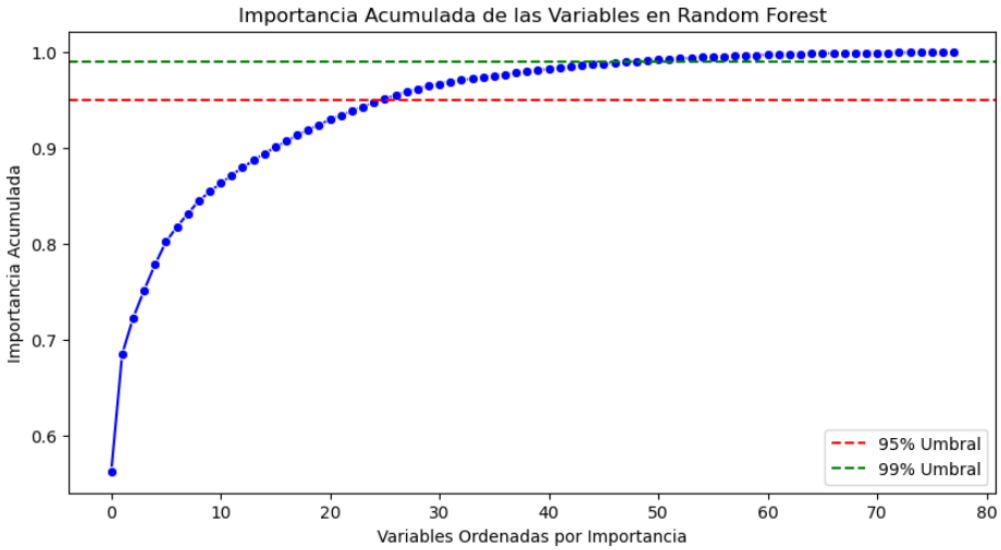
Una vez comprendidos los datos, se llevó a cabo su limpieza y transformación para generar un conjunto de datos listo para el entrenamiento del modelo.

Para determinar el umbral de importancia basado en la acumulación de impacto en SalePrice.

Teniendo en cuenta que despues de la codificación de las variables categóricas tenemos 305 filas.

- Resultados obtenidos:
    - Para alcanzar el 95% de la importancia total, necesitamos 26 variables.

- Para alcanzar el 99% de la importancia total, necesitamos 48 variables.



3.1Selección de Variables Relevantes

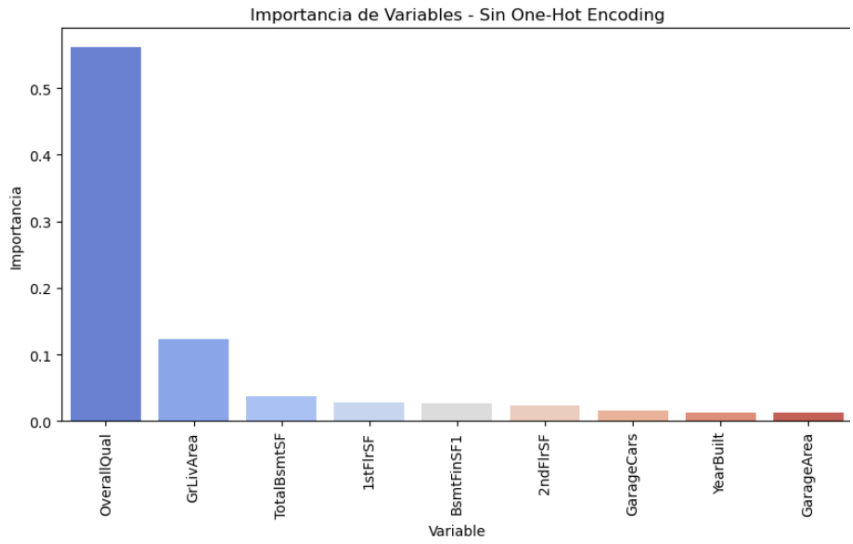
Se realizo una prueba iterativa eliminando variables poco importantes

- Entrenamos el modelo con todas las variables y luego vamos eliminando las de baja importancia hasta que el rendimiento empiece a empeorar.

Se redujo el número de variables de 81 a 9, seleccionando aquellas con mayor influencia en SalePrice:

- OverallQual (Calidad general de la vivienda).
- GrLivArea (Área habitable sobre el suelo).
- TotalBsmtSF (Área total del sótano).

- 1stFlrSF (Área del primer piso).
- BsmtFinSF1 (Área terminada del sótano).
- 2ndFlrSF (Área del segundo piso).
- GarageCars (Capacidad del garaje en autos).
- YearBuilt (Año de construcción).
- GarageArea (Área total del garaje en pies cuadrados).
- Codificación de Variables Categóricas



Se utilizó one-hot encoding para transformar variables categóricas en numéricas.

Se agruparon categorías con baja frecuencia para evitar sobreajuste.

3.2 Manejo de Valores Nulos (Aplicado en Transformaciones)

Se aplicaron estrategias de imputación consistentes con el EDA. Se aseguraron valores numéricos en todas las variables seleccionadas para evitar errores en el modelo.

3.3 Creación del Conjunto de Datos Final

Se generó un dataset limpio y optimizado: df_final_for_training.csv. Se garantizó que las variables estuvieran en el formato correcto para el entrenamiento del modelo.

Los datos fueron transformados y optimizados para asegurar la mejor precisión en el modelo predictivo.

4 MODELADO

El modelado consistió en la selección, entrenamiento y evaluación de modelos de Machine Learning con el objetivo de predecir con la mayor precisión posible el precio de venta de las viviendas (SalePrice). Se probaron diferentes algoritmos para analizar su desempeño y elegir el más adecuado según las métricas de evaluación.

4.1 Selección de Algoritmos

Para el entrenamiento, se optó por modelos basados en árboles de decisión, dada su capacidad para manejar relaciones no lineales entre variables y su robustez ante datos faltantes. Los modelos evaluados fueron:

- Random Forest: Un ensamble de múltiples árboles de decisión que mejora la generalización del modelo.
- XGBoost: Un modelo basado en boosting que ajusta progresivamente sus predicciones para minimizar errores.
- LightGBM: Un modelo optimizado para velocidad y eficiencia en grandes volúmenes de datos.

Cada modelo fue entrenado con el conjunto de datos procesado, utilizando una división 80/20 (80 % entrenamiento, 20 % prueba).

4.1Entrenamiento de los Modelos

Cada modelo fue entrenado con hiperparámetros predeterminados y posteriormente ajustado mediante optimización para mejorar su rendimiento.

El modelo XGBoost obtuvo la mejor precisión, con un $R^2 = 0.8868$, superando a los otros modelos. Posteriormente, se realizó una optimización de hiperparámetros, donde se ajustaron n_estimators y learning_rate, logrando un $R^2 = 0.8939$, mejorando el rendimiento.

4.3 Comparación de Modelos

Los modelos fueron evaluados con tres métricas clave:

- RMSE (Root Mean Squared Error): Indica cuánto se desvía la predicción del valor real.
- MAE (Mean Absolute Error): Muestra el error promedio absoluto de las predicciones.
- R^2 (Coeficiente de determinación): Mide qué porcentaje de la variabilidad en SalePrice es explicada por el modelo.

Modelo	RMSE	R^2	MAE
Random Forest	29502.68	0.8865	18972.12
XGBoost	29460.27	0.8868	19685.30
LightGBM	31342.71	0.8719	19314.01
XGBoost Optimizado	28524.28	0.8939	19039.14

El modelo XGBoost optimizado fue el seleccionado para su despliegue, ya que ofreció la mejor precisión en la predicción del precio de las viviendas.

4.4 Optimización de Hiperparámetros

Para mejorar la precisión del modelo XGBoost, se ajustaron hiperparámetros clave mediante búsqueda aleatoria (RandomizedSearchCV) y pruebas manuales de optimización.

- Primera optimización: Se ajustaron n_estimators y learning_rate, logrando un incremento en R^2.
- Segunda optimización: Se ajustaron max_depth y min_child_weight, logrando una reducción en el error absoluto medio.

Finalmente, se seleccionó el modelo con los hiperparámetros más eficientes y se validó en un conjunto de datos de prueba.

El modelo final se guardó en formato pkl para su posterior despliegue y prueba con nuevos datos.

5 EVALUACIÓN

Después del entrenamiento y optimización del modelo, se llevó a cabo la evaluación utilizando diferentes métricas de rendimiento. Estas métricas permitieron medir la precisión del modelo en la predicción de los precios de las viviendas y validar su efectividad en datos de prueba.

5.2 Métricas de Evaluación Utilizadas

Para evaluar el rendimiento del modelo, se utilizaron tres métricas clave:

- Root Mean Squared Error (RMSE): Mide la diferencia promedio entre los valores reales y predichos, penalizando errores grandes más que pequeños.
- Mean Absolute Error (MAE): Representa el error promedio absoluto en dólares, indicando cuánto se desvía la predicción del valor real en promedio.
- Coeficiente de Determinación (R^2): Indica el porcentaje de la variabilidad en SalePrice que el modelo puede explicar. Un valor cercano a 1 implica un mejor ajuste.

Se compararon los resultados obtenidos en los modelos probados:

Modelo	RMSE	R^2	MAE
Random Forest	29502.68	0.8865	18972.12
XGBoost	29460.27	0.8868	19685.30
LightGBM	31342.71	0.8719	19314.01
XGBoost Optimizado	28524.28	0.8939	19039.14

El modelo XGBoost Optimizado obtuvo el mejor rendimiento con un R^2 de 0.8939, indicando que el modelo explica casi el 89.4 % de la variabilidad en SalePrice.

5.3 Validación del Modelo con Cross-Validation

Para asegurar que el modelo no estuviera sobreajustado a los datos de entrenamiento, se aplicó validación cruzada con K-Fold (k=5).

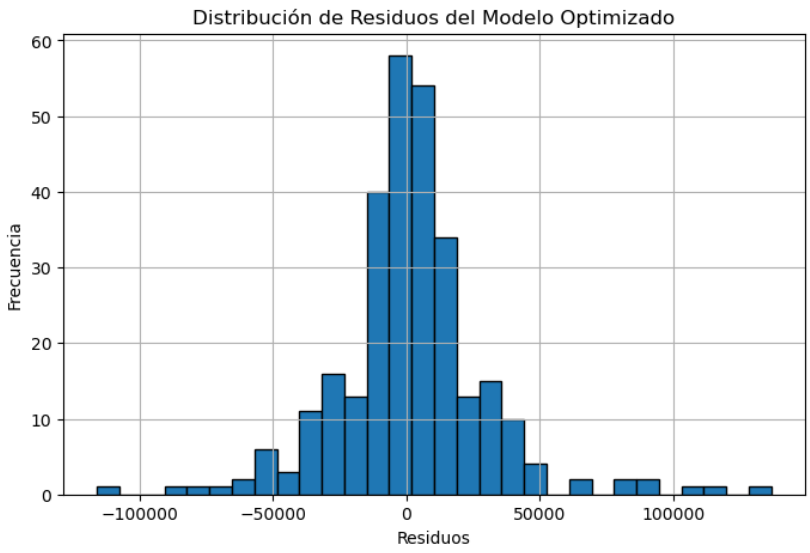
- R^2 promedio en validación cruzada: 0.8174
- Diferencia entre R^2 en prueba y en validación cruzada: 0.0765

Esto indica que el modelo mantiene una buena generalización sin caer en sobreajuste.

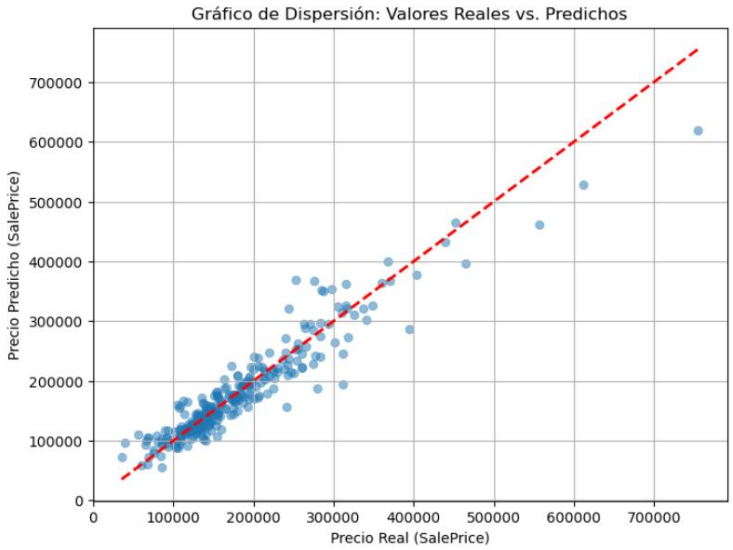
5.4 Visualizaciones de la Evaluación

Se realizaron las siguientes visualizaciones para analizar el desempeño del modelo:

1. Distribución de errores: Se generó un histograma de los residuos (diferencia entre valores reales y predichos) para verificar la dispersión de errores.



2. Gráfico de dispersión de valores reales vs. predichos: Se compararon los precios reales y predichos en un gráfico de dispersión. Se observó una alineación cercana a la diagonal ideal, lo que indica buena precisión.



3. Matriz de correlación de errores: Se analizó cómo varían los errores según el rango de SalePrice. El modelo tiene una ligera tendencia a subestimar los precios altos, ya que la correlación entre SalePrice y Error es positiva (0.39). Pero el ajuste de hiperparámetro disminuye el rendimiento del modelo.

5.5 Conclusiones Extraídas

- El modelo XGBoost Optimizado es el mejor candidato para despliegue, ya que ofrece el mejor balance entre precisión y estabilidad.
- El error absoluto medio (MAE) de 19,039 USD indica que el modelo puede desviarse en ese rango, lo cual es aceptable para este tipo de predicciones en el sector inmobiliario.
- El análisis de residuos muestra que el modelo no presenta sesgos significativos, aunque los valores más altos de SalePrice pueden tener mayor variabilidad en la predicción.
- La validación cruzada confirma que el modelo generaliza bien a nuevos datos, con un desempeño estable entre entrenamiento y validación.

Estos resultados respaldan la implementación del modelo en un entorno de producción para la predicción de precios de viviendas, con un margen de error razonable y una capacidad predictiva sólida.

6 Despliegue

El objetivo de esta fase es hacer que el modelo entrenado y validado esté disponible para su uso en predicciones en nuevos datos. Para ello, se guardó el

modelo en un formato que permite reutilizarlo y se estableció un proceso para recibir datos de entrada, realizar predicciones y devolver los resultados.

6.1 Guardado del Modelo

El modelo XGBoost optimizado se guardó en un archivo .pkl para poder reutilizarlo sin necesidad de volver a entrenarlo desde cero.

6.2 Carga del Modelo Desplegado

Cuando queramos utilizar el modelo para hacer predicciones en nuevos datos, simplemente lo cargamos con:

```
# Cargar el modelo entrenado
model_filename = "xgb_optimized_model.pkl"
xgb_loaded = joblib.load(model_filename)
```

6.3 Transformación de Datos para Predicción

Dado que el modelo espera recibir variables preprocesadas (OverallQual, GrLivArea, TotalBsmtSF, etc.), es necesario transformar los datos de entrada antes de hacer predicciones.

Para ello, se creó un script que toma test.csv y lo convierte en test_transformed.csv, asegurando que el modelo reciba los datos en el formato correcto.

6.4 Generación de Predicciones con el Modelo Desplegado

Una vez que los datos han sido transformados, el modelo se usa para hacer predicciones.

```
# Cargar nuevos datos (suponiendo que tienes un DataFrame con las mismas columnas usadas para entrenar)
new_data = pd.read_csv(r"C:\Users\franc\Desktop\ML_IoT\Nuevos_datos_preparados.csv", sep=',')

# Extraer el ID para mantenerlo en la salida final
df_test_ids = new_data[['Id']]

# Seleccionar solo las columnas necesarias para el modelo (excluyendo 'Id')
columns_needed = ['OverallQual', 'GrLivArea', 'TotalBsmtSF', '1stFlrSF',
                  'BsmtFinSF1', '2ndFlrSF', 'GarageCars', 'YearBuilt', 'GarageArea']

df_test_filtered = new_data[columns_needed]

# Hacer predicciones
predicted_price = xgb_loaded.predict(df_test_filtered)


submission = pd.DataFrame({
    "Id": df_test_ids["Id"],
    "SalePrice": predicted_price
})

# Guardar el archivo en formato CSV listo para envío
submission.to_csv("submission.csv", index=False)

print(f"Precio predicho de la vivienda: ${predicted_price[0]:,.2f}")
```

6.4 Validación de las Predicciones

Para evaluar la precisión del modelo en estos nuevos datos, se compararon los valores predichos con los valores de sample_submission.csv.

Se calculó el error absoluto relativo (% de error por predicción) y se sumó el error total acumulado.

6.5 Conclusión del Despliegue

- El modelo ha sido guardado y cargado correctamente para reutilización.
- Se ha establecido un proceso de transformación de datos que permite generar predicciones con nuevos datos.
- Las predicciones se guardan en submission.csv, manteniendo los IDs originales para su identificación para los casos de nuevos datos de testeo,
- Se implementó un método de validación para verificar el error en las predicciones del modelo desplegado.

Bibliografía

1. **Metodología CRISP-DM**
   - Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). *CRISP-DM 1.0: Step-by-step data mining guide*. SPSS Inc.
   - Wirth, R., & Hipp, J. (2000). *CRISP-DM: Towards a Standard Process Model for Data Mining*.
2. **Análisis Exploratorio de Datos (EDA) y Feature Engineering**
   - Han, J., Kamber, M., & Pei, J. (2011). *Data Mining: Concepts and Techniques*. Elsevier.
   - Kuhn, M., & Johnson, K. (2013). *Applied Predictive Modeling*. Springer.
3. **Modelos de Machine Learning (Random Forest, XGBoost, LightGBM)**
   - Breiman, L. (2001). *Random Forests*. Machine Learning, 45(1), 5-32.
   - Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System*. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16).
   - Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). *LightGBM: A Highly Efficient Gradient Boosting Decision Tree*. Advances in Neural Information Processing Systems (NeurIPS).
4. **Evaluación de Modelos de Machine Learning**
   - James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer.
   - Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media.
5. **Despliegue de Modelos en Producción**
   - Raschka, S., & Mirjalili, V. (2019). *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2*. Packt Publishing.
   - Kluyver, T., et al. (2016). *Jupyter Notebooks – a publishing format for reproducible computational workflows*.

Discusión

Referencias: