

Resumen TFG Daniel Carrera Bonilla

TFG enfocado en el desarrollo y análisis de una plataforma predictiva de precios inmobiliarios para las provincias de Andalucía, utilizando técnicas de Machine Learning y Big Data.

Enlaces de interés

- Repositorio github: <https://github.com/danielcb0/TFG---Predictor-Precios-Vivienda-Andalucia>
- Overleaf para memoria prácticas: <https://www.overleaf.com/5811387216kmwbbwrtjpxt#d348c0>
- Despligue aplicación (inactiva debido al tamaño de procesamiento del modelo que excede el plan gratuito): <https://tfg-predictor-precios-vivienda-andalucia.onrender.com/predict>
- API: <https://rapidapi.com/apidojo/api/idealista2>

Estructura

La **estructura del proyecto** se organiza en los siguientes directorios principales:

Carpeta Data

data/: Contiene los scripts para:

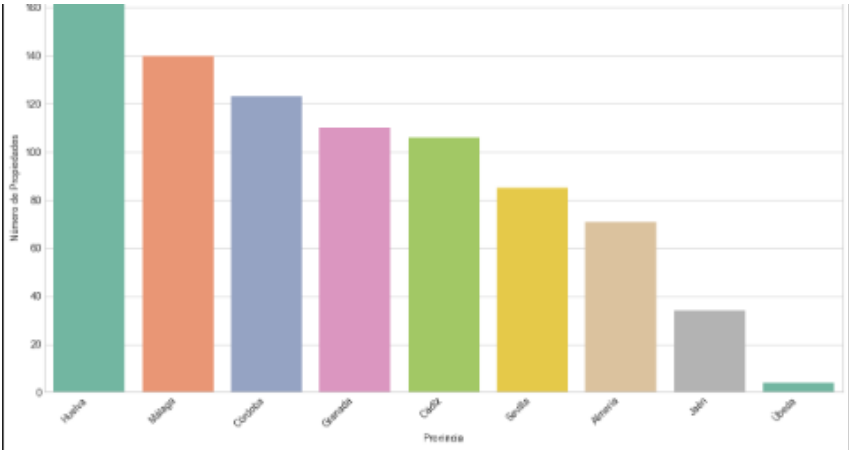
- La extracción de los datos mediante la api de rapid api de Idealista(
[extraccionViviendasMensualv2.py](#))
- Preprocesamiento para coger los datos crudos de todos los CSVs que se generan en la extracción, fusionarlos y conseguir un único csv([preprocess.py](#)).
- El almacenamiento de datos crudos ([raw/](#)), procesados ([processed/](#)) y limpios después de pasar por una serie de acciones en un notebook ([clean/](#)).

Carpeta Notebooks

notebooks/: Incluye los cuadernos de Jupyter para:

- El análisis exploratorio de datos por provincias ([\[analisis_visualizacion_por_provincia.ipynb\]](#))
- Análisis general del dataset ([\[analisis_visualizacion v2.ipynb\]](#)).
- La carga y limpieza del dataset de la carpeta processed se da en ([\[carga_limpieza.ipynb\]](#)).

- El desarrollo y la generación de modelos se da en el notebook ([[model_testing v3.ipynb](#)]).



```

advertencia: 23272 propiedades no pudieron ser asignadas a una provincia (etiquetadas como 'Desconocida').
Esto puede afectar la representatividad del análisis provincial.
Para mejorar esto, se necesitaría un mapeo más robusto, NLP, o geocodificación (lat/lon -> provincia).
Ejemplos de 'ubicacion' no mapeadas:

ubicacion
Barrio Brillante      185
Centro               147
Barrio Centro         138
Barrio Casco Histórico - Corredera - Ribera  118
Barrio Casco Histórico - Ollerías - Marrubial  85
Name: count, dtype: int64

4. Análisis Descriptivo de Variables Clave por Provincia

Una vez asignada la provincia, calcularemos estadísticas descriptivas (media, mediana, desviación estándar, mínimo, máximo) para las variables numéricas clave (precio, superficie, habitaciones, baños, precio_m2), agrupadas por la columna 'provincia'.

if not df.empty and 'provincia' in df.columns and df[df['provincia'] != 'Desconocida']['provincia'].nunique() > 0:
    print("\nAnálisis Descriptivo de Variables Clave por Provincia")
    print("=====")

    # Filtrar datos donde la provincia ha sido identificada y no es 'Desconocida' o 'No disponible'
    df_analisis_prov = df[(df['provincia'] != 'Desconocida') & (df['provincia'] != 'No disponible')].copy()

    if not df_analisis_prov.empty:
        variables_descriptivas = ['precio', 'superficie', 'habitaciones', 'baños', 'precio_m2']

        # Asegurarse de que las columnas existen
        variables_existentes = [var for var in variables_descriptivas if var in df_analisis_prov.columns]

        if variables_existentes:
            print(f"Analizando para las variables: {variables_existentes}\n")

            stats_por_provincia = df_analisis_prov.groupby('provincia')[variables_existentes].agg(
                ['mean', 'median', 'std', 'min', 'max', 'count']
            )

            # Mejorar la presentación de las estadísticas
            for var in variables_existentes:
                print(f"---- Estadísticas para '{var}' por Provincia ----")
                display(stats_por_provincia[var].sort_values(by='median', ascending=False))
                print('\n')
            else:
                print("Ninguna de las variables clave para análisis descriptivo se encuentra en el DataFrame.")

        else:
            print("No hay datos suficientes con provincias identificadas para realizar el análisis descriptivo.")
    else:
        print("El DataFrame está vacío, la columna 'provincia' no existe, o no hay provincias identificadas (excluyendo 'Desconocida').")

Análisis Descriptivo de Variables Clave por Provincia
=====
Analizando para las variables: ['precio', 'superficie', 'habitaciones', 'baños', 'precio_m2']

--- Estadísticas para 'precio' por Provincia ---

provincia    mean    median    std    min    max count
Huelva      3686136.69  4500000.00  5575413.04  260000.00  12900000.00  140
Málaga      3045483.57  1095000.00  4148333.53  270000.00  1680000.00  162
Cádiz       327837.83   925000.00  541775.52  200000.00  4990000.00  106
Sevilla     250585.59   850000.00  327565.80  105000.00  1580000.00  85
Córdoba     147175.12   700000.00  165052.96  200000.00  920000.00  123
Granada     442075.45   694500.00  1245788.28  200000.00  8800000.00  110
Jaén        91208.82    650000.00  100759.03  186000.00  590000.00  34
Úbeda       58750.00    628000.00  18675.03  320000.00  75000.00  4
Almería     107304.23   580000.00  202058.71  230000.00  1565000.00  71

--- Estadísticas para 'superficie' por Provincia ---

provincia    mean    median    std    min    max count
Málaga      561.65    647.00  451.20  40.00  2004.00  140
Huelva      549.00    640.00  450.00  40.00  2004.00  162
Cádiz       549.00    640.00  450.00  40.00  2004.00  106
Sevilla     549.00    640.00  450.00  40.00  2004.00  85
Córdoba     549.00    640.00  450.00  40.00  2004.00  123
Granada     549.00    640.00  450.00  40.00  2004.00  110
Jaén        549.00    640.00  450.00  40.00  2004.00  34
Úbeda       549.00    640.00  450.00  40.00  2004.00  4
Almería     549.00    640.00  450.00  40.00  2004.00  71
```

```
if not df_analisis_prov.empty and 'precio' in df_analisis_prov.columns:
    # Ordenar provincias por precio mediano para mejor visualización en boxplots
    order_provincias_precio = df_analisis_prov.groupby('provincia')['precio'].median().sort_values().index

    # Boxplot de 'precio' por provincia
    plt.figure(figsize=(14, 8))
    sns.boxplot(data=df_analisis_prov, x='precio', y='provincia', order=order_provincias_precio, palette='coolwarm')
    plt.title("Distribución de Precios por Provincia")
    plt.xlabel('Precio (€)')
    plt.ylabel('Provincia')
    plt.yscale('log') # Precio suele tener una distribución asimétrica
    plt.grid(True, axis='x', linestyle='--', alpha=0.7)
    plt.tight_layout()
    plt.show()

    # Diagrama de barras del precio mediano por provincia
    median_precio_prov = df_analisis_prov.groupby('provincia')['precio'].median().sort_values(ascending=False)
    plt.figure(figsize=(12, 7))
    sns.barplot(x=median_precio_prov.index, y=median_precio_prov.values, palette='coolwarm_r', order=median_precio_prov.index)
    plt.title("Precio Mediano por Provincia")
    plt.xlabel('Provincia')
    plt.ylabel('Precio Mediano (€)')
    plt.xticks(rotation=45, ha='right')
    plt.grid(True, axis='y', linestyle='--', alpha=0.7)
    plt.tight_layout()
    plt.show()

else:
    print("No hay datos de 'precio' o provincias identificadas para el análisis.")

if not df_analisis_prov.empty and 'precio_m2' in df_analisis_prov.columns:
    # Ordenar provincias por precio m2 mediano
    order_provincias_precio_m2 = df_analisis_prov.groupby('provincia')['precio_m2'].median().sort_values().index

    # Boxplot de 'precio_m2' por provincia
    plt.figure(figsize=(14, 8))
    sns.boxplot(data=df_analisis_prov, x='precio_m2', y='provincia', order=order_provincias_precio_m2, palette='viridis_r')
    plt.title("Distribución de Precio por m² por Provincia")
    plt.xlabel('Precio por m² (€/m²)')
    plt.ylabel('Provincia')
    plt.yscale('log') # Opcional, si precio_m2 también es muy asimétrico
    plt.grid(True, axis='x', linestyle='--', alpha=0.7)
    plt.tight_layout()
    plt.show()

    # Diagrama de barras del precio m2 mediano por provincia
    median_precio_m2_prov = df_analisis_prov.groupby('provincia')['precio_m2'].median().sort_values(ascending=False)
    plt.figure(figsize=(12, 7))
    sns.barplot(x=median_precio_m2_prov.index, y=median_precio_m2_prov.values, palette='viridis', order=median_precio_m2_prov.index)
    plt.title("Precio Mediano por m² por Provincia")
    plt.xlabel('Provincia')
    plt.ylabel('Precio Mediano por m² (€/m²)')
    plt.xticks(rotation=45, ha='right')
    plt.grid(True, axis='y', linestyle='--', alpha=0.7)
    plt.tight_layout()
    plt.show()

else:
    print("No hay datos de 'precio_m2' o provincias identificadas para el análisis.")

else:
    print("El DataFrame está vacío, la columna 'provincia' no existe, o no hay provincias identificadas (excluyendo 'Desconocida').")
```

Analisis Comparativo de Precios por Provincia

[C:\Users\dania\AppData\Local\Temp\ipykernel_17728\1360117082.py:12: FutureWarning:](#)

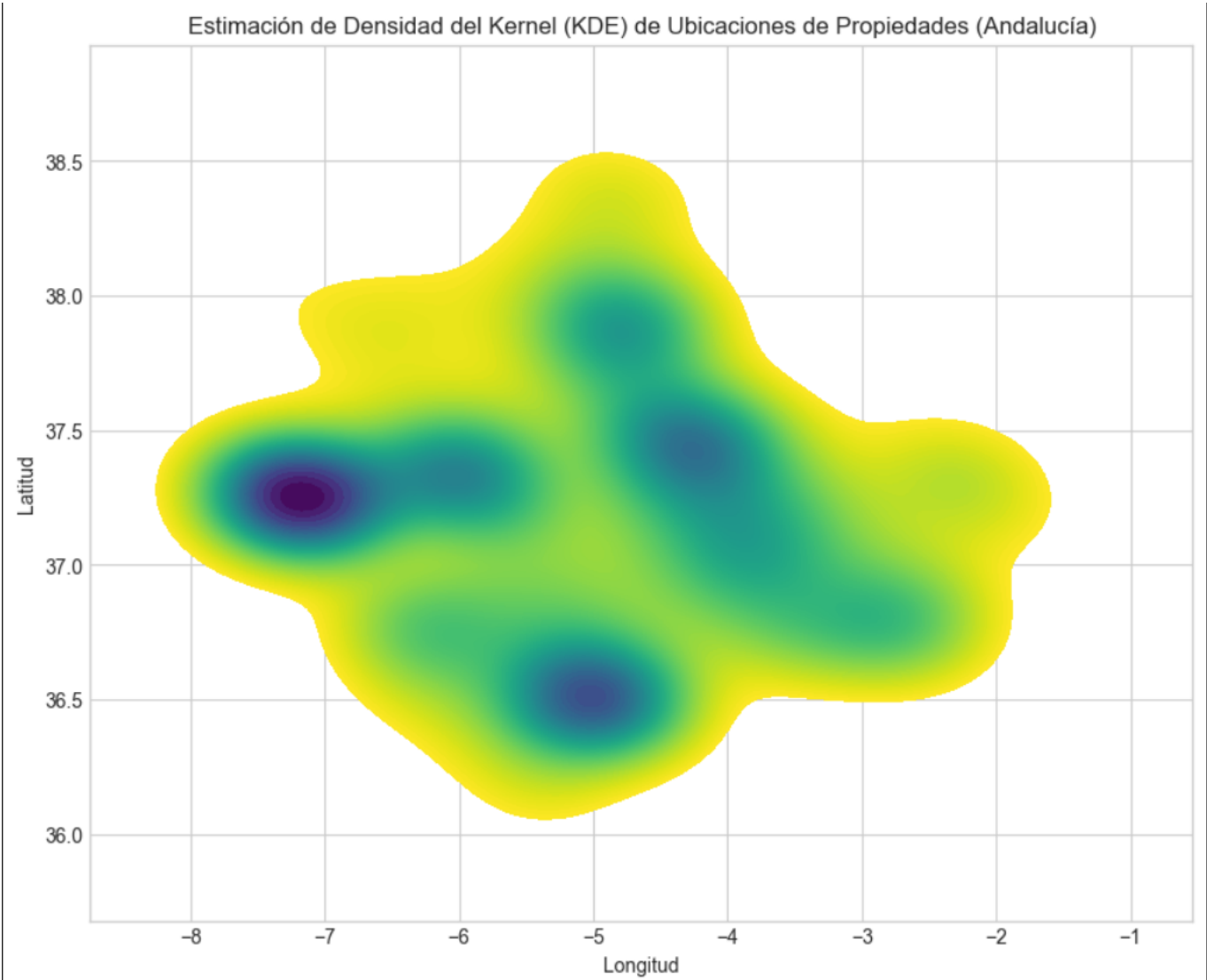
Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'y' variable to 'hue' and set 'legend=False' for the same effect.

```
sns.boxplot(data=df_analisis_prov, x='precio', y='provincia', order=order_provincias_precio, palette='coolwarm')
```

[C:\Users\dania\AppData\Local\Temp\ipykernel_17728\1360117082.py:20: FutureWarning:](#)

Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.

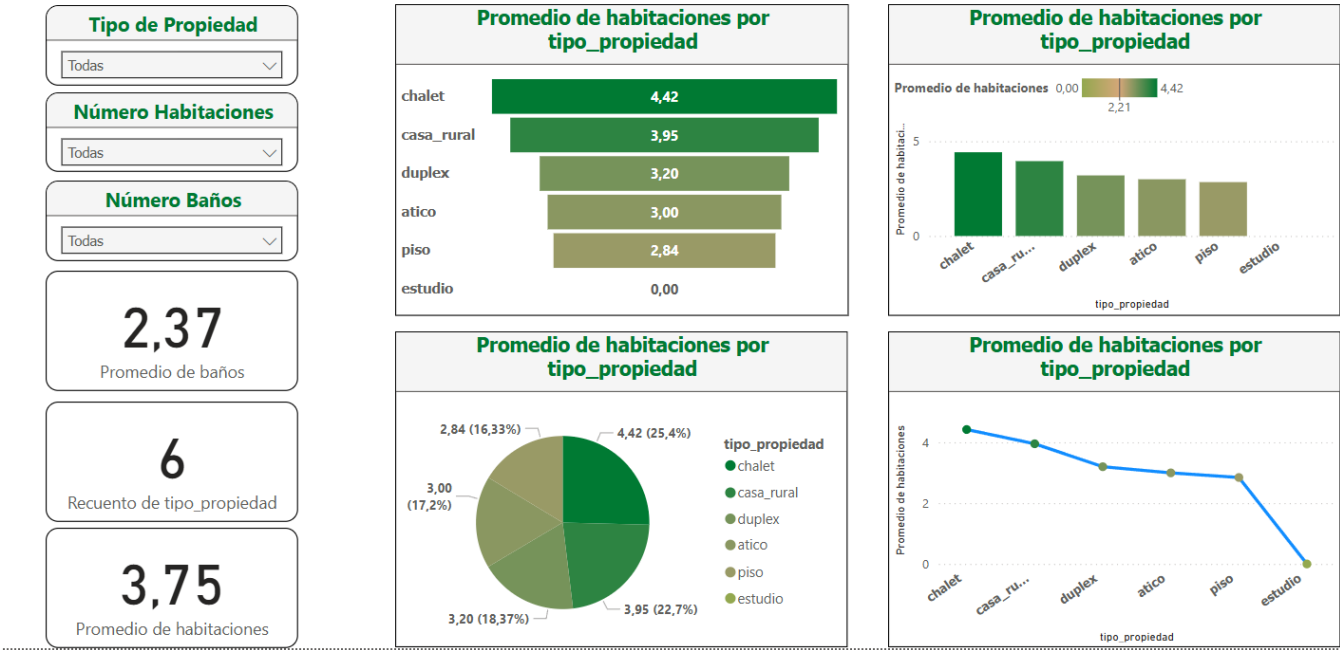
```
sns.barplot(x=median_precio_prov.index, y=median_precio_prov.values, palette='coolwarm_r', order=median_precio_prov.index)
```



Power BI

Como extra al análisis de nuestros datos he diseñado un cuadro de mando que nos pueda aportar una serie de datos directos sobre el dataset.

Cuadro de mando para análisis de viviendas en venta en Andalucía



Carpeta Models

- `models/`:
- Contiene los modelos entrenados.

Carpeta Backend

- `backend/`: Contiene la lógica de la aplicación del lado del servidor (`[app.py]`). Desarrollada con flask y flask-cors. Se crea una pequeña API que consume el modelo.

Carpeta Backend


- `frontend/`: Incluye los archivos de la interfaz de usuario (`[index.html]`).

La aplicación web es sencilla marcada por dos partes:

- Ruta para acceder al modelo predictor

- Ruta para acceder a los estudios de los datos con los que trabajamos (notebooks exportados a HTML)

en Andalucía



Superficie (m²):

120

Número de Habitaciones:

3

Número de Baños:

2

Latitud:

37,1969934816

Longitud:

-3,6164274080

Tipo de Propiedad:

Piso

Generar Predicción

Precio Estimado:

€ 111.209,99

El mapa con el que se obtienen las coordenadas está limitado para mostrar únicamente (y lo más preciso posible) solamente la parte de andalucía.

La idea es que la aplicación se despligue en un host gratuito como renders. Ahora mismo está ya, sin embargo, por el creciente tamaño del modelo este ha provocado que se superen los recursos de memoria RAM que proporciona el plan gratuito. Ahora mismo tengo dos opciones, pagar 25\$ o disminuir el tamaño del modelo optando por recortar ramas y partes del algoritmo que se elige ahora mismo, RandomForest, o escoger otro algoritmo que genere un modelo más liviano.

- `docs/`: Contiene la documentación del proyecto
- Otros archivos relevantes incluyen `Dockerfile`, `docker-compose.yml` para la contenerización (no dockerizado aún) y `requirements.txt` para las dependencias.

Las **partes o fases principales del proyecto** son:

1. **Introducción y Extracción de Datos:** Definición del alcance del proyecto y desarrollo de scripts para la recopilación automatizada de datos inmobiliarios de Andalucía.
2. **Preprocesamiento y Limpieza de Datos:** Transformación de los datos crudos en un formato limpio y estructurado, adecuado para el análisis y modelado. Esto incluye la gestión de valores nulos, la corrección de tipos de datos y la ingeniería de características.
3. **Análisis Exploratorio de Datos (EDA):** Investigación de los datos para descubrir patrones, tendencias, anomalías y relaciones entre variables, con el fin de obtener una comprensión profunda del mercado inmobiliario andaluz.
4. **Desarrollo y Evaluación de Modelos de Machine Learning:** Entrenamiento de diversos modelos predictivos (regresión) para estimar los precios de las viviendas. Se realiza una selección de características, optimización de hiperparámetros y evaluación exhaustiva para elegir los modelos con mejor rendimiento para cada provincia. Se presta especial atención a características geográficas como latitud y longitud.
5. **Implementación de la Plataforma Web:**
 - **Backend:** Desarrollo de una API para servir las predicciones de los modelos.
 - **Frontend:** Creación de una interfaz de usuario interactiva que permita a los usuarios introducir características de una vivienda (superficie, habitaciones, ubicación en un mapa, etc.) y obtener una predicción de precio.
6. **Documentación y Presentación de Resultados:** Elaboración de la memoria del TFG, incluyendo la descripción detallada de cada fase, los resultados obtenidos y las conclusiones del proyecto.