SEGUNDA ENTREGA DE PROYECTO

POR:

Danilo Tovar Arias

Daniel Rosas Mendoza

MATERIA:

Introducción a la inteligencia artificial

PROFESOR:

Raúl Ramos Pollán

UNIVERSIDAD DE ANTIOQUIA

FACULTAD DE INGENIERIA

MEDELLIN 2023

CONTENIDO

Pág.

1. PLANTEAMIENTO DEL PROBLEMA

Un problema común que surge en el ámbito de la compra de alojamientos de una vivienda es la toma de decisiones sobre qué opción es la más adecuada en función de las circunstancias y las necesidades individuales de cada persona, así mismo, el precio del alojamiento en una propiedad puede estar influenciado por una variedad de factores, como la ubicación, el tamaño de la propiedad, el número de habitaciones, el estado de la propiedad, entre otros.
Con el fin de dar una respuesta a este problema, se tiene como objetivo la creación de un modelo de regresión que tenga en cuenta todas las características relevantes de la propiedad, utilizando el dataset de precios de vivienda de París como conjunto de datos de entrenamiento.

1.1. Dataset

El dataset seleccionado para trabajar en la solución de este problema, es un conjunto de archivos que podemos encontrar en una competencia de la página Kaggle, en su categoría de playground, temporada tres, episodio seis. Este dataset contiene tres archivos.

- **train.csv:** El conjunto de datos de entrenamiento; Obtener 'price' es el objetivo
- **test.csv:** El conjunto de datos de prueba; el objetivo es predecir acordemente 'price' en contraste de estos datos.
- **sample_submission.csv:** muestra de un archivo de envío en el formato correcto

El conjunto de datos para esta competencia (tanto de entrenamiento como de prueba) se generó a partir de un modelo de aprendizaje profundo entrenado en la Predicción del precio de la vivienda de París . Las distribuciones de

características son similares, pero no exactamente iguales, a las del original. Los datos contenidos dentro de estos archivos son:

- **squareMeters –** Metros Cuadrados totales
- **numberOfRooms –** Número de habitaciones
- **hasYard –** Tiene patio?
- **hasPool –** Tiene piscina?
- **floors –** Número de pisos
- **cityCode –** Código Zip
- **cityPartRange –** Entre mayor el rango, más exclusive es el vecindario.
- **numPrevOwners –** Número de propietarios anteriores
- **made –** Año de construcción
- **isNewBuilt –** es Nueva?
- **hasStormProtector –** Tiene protección ante tormentas?
- **basement –** Metros cuadrados de sótano
- **attic -** Metros cuadrados del ático
- **garage –** Tamaño del garaje
- **hasStorageRoom –** Tiene almacén?
- **hasGuestRoom –** Tiene habitación de invitados?
- **price –** valor predicho

1.2. Métrica

La métrica de evaluación principal para el modelo será la raíz del error cuadrático medio (RMSE, por sus siglas en inglés) es una medida comúnmente utilizada en estadística y en el campo del aprendizaje automático para evaluar la precisión de un modelo de predicción.

El RMSE se calcula como la raíz cuadrada de la media de los errores al cuadrado entre los valores reales y los valores predichos. En otras palabras, el RMSE mide la diferencia promedio entre los valores reales y los valores predichos y representa la cantidad de error en las predicciones del modelo.

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

RMSE expresa el valor del error, N es la cantidad total de observaciones en el dataset, finalmente, y_i es el precio real de la vivienda, y $\widehat{y_i}$ representa el valor predicho, en la instancia i.

El uso de la raíz cuadrada en el RMSE se realiza para que la medida esté en la misma escala que las unidades de la variable de respuesta, lo que facilita la interpretación y comparación de diferentes modelos. En resumen, la raíz del error cuadrático medio sirve para evaluar la precisión de un modelo de predicción.

Por otro lado, para la métrica de negocio, en que las predicciones sean suficientemente confiables para saber la evolución de los precios de acuerdo a las características de las viviendas. Con esta información se pueden realizar análisis financieros para la promoción de alojamientos específicos a las necesidades del cliente.

1.3. Variable Objetivo

Como se ha mencionado anteriormente, la variable objetivo que se desea predecir corresponde a “price”, que nos dice el precio de cada edificación, así se analizarán las demás variables para entender su comportamiento.

2. EXPLORACIÓN DE DATOS

Para iniciar, realizamos una exploración inicial del dataset, identificando las columnas de interés para el desarrollo.

2.1. Análisis de la variable objetivo

Se construyó una gráfica que nos permite observar la distribución que tiene la variable objetivo, como se muestra en la *Figura 1*, donde se puede apreciar que posee una asimetría muy baja hacia la izquierda. Sin embargo, posee un comportamiento adecuado para la realización del entrenamiento y las pruebas de algoritmo.

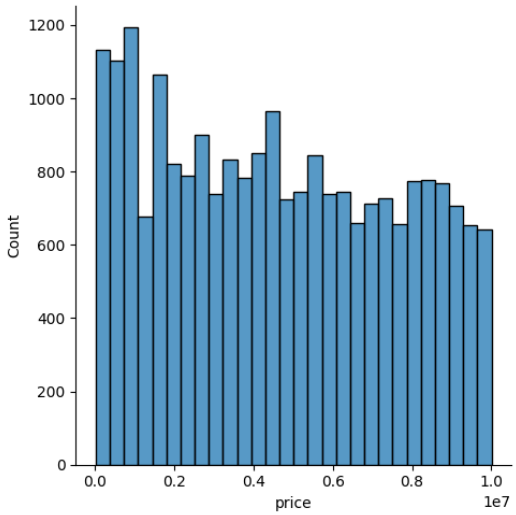

Figura 1. Distribución de la variable objetivo

2.2. Secciones de la ciudad

Entre las variables de interés se identificó *cityPartRange*, que nos dice la sección de la ciudad donde se encuentra ubicado el edificio. A esta variable se le realizó un análisis de frecuencia y de precio promedio por cada valor único, como se puede observar en las *Figura 2* y *Figura 3*, respectivamente.

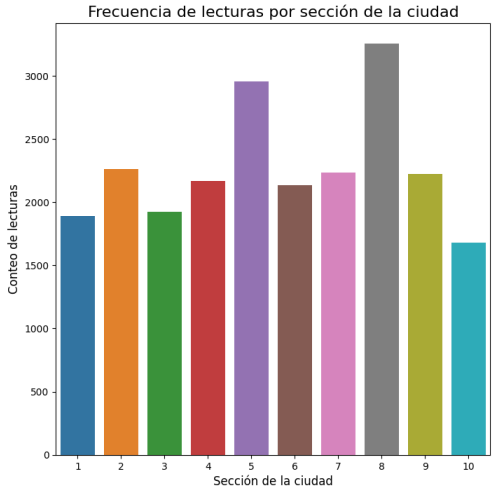

Figura 2. Frecuencia de los valores para la variable cityPartRange.

Se pudo observar que la sección 8 es aquella a la que pertenece el mayor porcentaje de edificios, mientras que la sección 10 se aquel al que pertenece la menor cantidad.

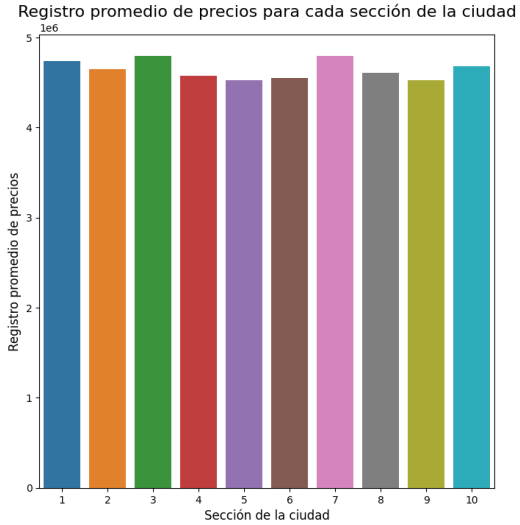

Figura 3. Promedio del precio por cada valor de la variable cityPartRange.

Se pudo observar que el promedio de la variable objetivo es relativamente constante entre las diferentes secciones de la ciudad, donde la sección 3 con el mayor valor se separa de la sección 9, correspondiente al menor, aproximadamente por un 5,6% (0.270909e+06).

2.3. Año de construcción

Otra de las variables de interés identificadas fue *made*, que nos dice el año de construcción del edificio. A esta variable se le realizó el mismo análisis que a la variable anterior. Los resultados se pueden observar en la *Figura 4* y *Figura 5*.

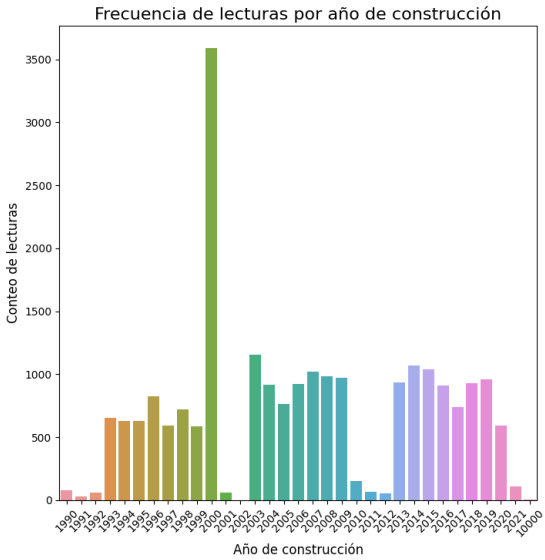

Figura 4. Frecuencia de los valores para la variable made.

Se pudo observar que el año 2000 es aquel en el que fueron construidos el mayor porcentaje de edificios, mientras que el año 2002 es aquel en el que fueron construidos la menor cantidad.

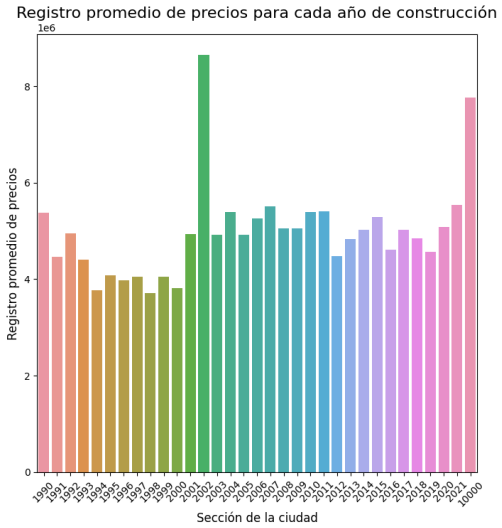

Figura 5. Promedio del precio por cada valor de la variable made.

Se pudo observar que el promedio de la variable objetivo es relativamente constante entre los diferentes años de construcción, a excepción del año 2000 y 10000. Así mismo, se pudo observar que existen una variable anómala correspondiente al año 10000, por lo que se eliminará en el preprocesado de datos.

2.4. Datos faltantes

Un elemento importante para el análisis son los datos faltantes. Luego de realizar un recorrido y conteo a través del database se logró comprobar que no existen datos faltantes dentro del mismo. Sin embargo, por motivos de aprendizaje y cumplir con los requerimientos del proyecto, se realizó una simulación para generar datos faltantes dentro del database la cual será explicada en la sección de preprocesado.

2.5. Correlación entre las diferentes variables

Otro factor importante que se tuvo en cuenta para el análisis fue la correlación entre las diferentes variables con la variable objetivo. A través de la *tabla 1* se puede observar que la variable squareMeters es aquella que posee la mayor correlación.

price	
price	1.000000
squareMeters	0.591749
numberOfRooms	0.091681
floors	0.038374
made	0.024270
cityCode	0.021986
hasStormProtector	0.020512
isNewBuilt	0.008080
hasPool	0.006023
hasStorageRoom	0.001567
hasYard	-0.002545
attic	-0.006851
id	-0.008060
numPrevOwners	-0.008546
hasGuestRoom	-0.009309
cityPartRange	-0.009366
basement	-0.034940
garage	-0.120137

Tabla 1. Correlación entre variable con price.

2.6. Distribución de las variables numéricas

Se realizó una muestra de las distribuciones de cada una de las variables. Sin embargo, no se observaron problemas relevantes dentro de las mismas.

3. TRATAMIENTO DE DATOS

3.1. Simulación de datos faltantes

Se tomaron aleatoriamente 3 columnas diferentes a id y precio, a las cuales se les aplicó una eliminación aleatoria de datos, de tal manera que entre 5 a 70% de los datos para dichas columnas serían ahora datos faltantes. Como resultado de esta

simulación, se vieron afectadas las columnas “*hasStormProtector*”, “*cityCode*” y “*made*”.

	Total	Percent
hasStormProtector	13561	59.661241
cityCode	2278	10.021997
made	1953	8.592169

Tabla 2. Cantidad y porcentaje de datos faltantes luego de la simulación

Así, se obtuvo un nuevo dataset (*train_mod*) que será utilizado como el nuevo *train*, al cual se le realizó el adecuado tratamiento de datos para su posterior aplicación en los diferentes algoritmos.

3.2. Eliminación de datos anómalos

Como se mencionó en la sección 2.3, dentro de la variable “*made*” existen datos anómalos correspondientes al año 10000, por lo que se eliminaron aquellas filas con dicho valor.

3.3. Eliminación de columnas con muchos datos faltantes

Como se puede observar, debido a la simulación la columna “*hasStormProtector*” terminó con una cantidad de datos faltantes superior al 50%. Por lo que se decidió eliminarla del database.

3.4. Relleno de datos faltantes

Así mismo, para el resto de columnas que se vieron afectadas cuyo porcentaje es inferior al 50%, se decidió rellenar los valores faltantes con la media correspondiente en cada una.

4. Métodos supervisados

4.1. Selección de modelos

4.1.1. Partición de datos

Para entrenar los modelos se usará el dataset que se generó, llamado "*train_mod*". De todo el conjunto de estos datos es necesario hacer una partición para entrenar y otra para prueba. De los datos que se usarán para entrenamiento, se hará una partición adicional en este conjunto de modo que se tenga una parte para entrenar y la otra parte para hacer validación. Cabe destacar que para entrenar los algoritmos se decidió por eliminar las variables ya que no tenían una correlación muy cercana a cero, es decir, aquellas cuyo valor absoluto de correlación era igual o menos a 0,01. De esta manera se conservan solo aquellas que tienen algo de correlación con la variable objetivo.

4.1.2. Selección de datos

Se plantearon inicialmente tres modelos: Regresión lineal, Árbol de decisión y RandomForest. De las tres posibilidades se realizó la selección de los dos modelos que den mejores resultados. Es importante resaltar que se implementó una metodología de cross-validation para realizar la selección de los modelos. Así, de este análisis se encontró que los modelos con el mejor desempeño fueron la Regrasión Lineal y el RandomForest.

4.2. Mejores hiperparámetros para el RandomForest

Debido a que el RandomForest hace uso de hiperparámetros, se realizó un estudio para determinar los mejores. Mediante el uso del modulo de la librería de sk.learn.model_selection llamado GridSearchCV, se utilizó un conjunto de hiperparámetros para realizar diferentes combinaciones y encontrar la mejor implementado una metodología de cross-validation. Finalmente, una vez se tienen dichos hiperparámetros, se obtiene el mejor estimador.

```
Mejor estimador Random Forest:  RandomForestRegressor(max_depth=7, n_estimators=10)
Mejores parámetros para el estimador Random Forest:  {'max_depth': 7, 'n_estimators': 10}
```

Figura 6. Resultados de la selección de hiperparámetros

A partir del análisis se obtuvo que los mejores valores de los hiperparámetros para el RandomForest es max_depth = 7 y n_estimators = 10.

5. Métodos supervisados y no supervisados

Una vez realizado el análisis con métodos supervisados, se realizó un análisis implementando métodos no supervisados a modo de tratamiento de los datos antes del entrenamiento de los métodos supervisados. En este caso se usó un único método no supervisados: PC. De esta manera se podrá reducir la dimensionalidad del dataset y conservar aquellos datos que conserven la mayor cantidad de información y con estos entrenar los algoritmos supervisados.

5.1. PCA y RandomForest

Para implementar el PCA, se realizó un código mediante el cual se exploraron varias opciones de hiperparámetro de este algoritmo. Tomando valores para n_components de 1,3,5,7 y 9. Se encontró que el mejor desempeño lo daba aquel modelo con n_components = 7.

Una vez obtenido el mejor PCA, se realizó nuevamente el procedimiento descrito anteriormente para obtener los mejores hiperparámetros del RandomForest, usando GridSearchCV. De esta manera se obtiene que para este caso los mejores hiperparámetros son n_estimators = 10 y max_depth = 5.

6. Curvas de aprendizaje

6.1. Métodos supervisados

Para obtener la curva de aprendizaje de los algoritmos se usa el módulo learning_curves de la librería de sk.learn.model_selection, la cual implementa una metodología de cross-validation para evaluar diferentes desempeños usando diferentes tamaños para el entrenamiento del modelo. La Figura 7 muestra la gráfica de aprendizaje para la regresión lineal. Cabe destacar que la línea verde llamada Cross-validation score, es la curva de aprendizaje en prueba. Se puede observar que inicialmente presenta buen desempeño en el train, sin embargo la distancia entre este valor y el error en la prueba demuestra que existe *overfitting* y, además, se presenta un comportamiento de Bias, puesto que a pesar de que se le dan más datos parece ser que el error se estabiliza y no disminuye a pesar de incrementar la cantidad de datos. Estos problemas se pueden solucionar trabajando en la complejidad del modelo o utilizando más datos.

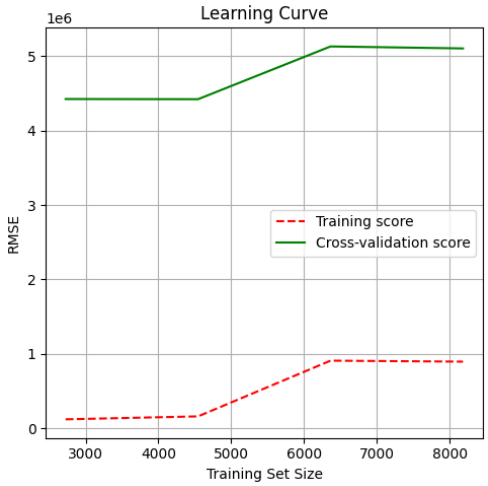

Figura 7. Curva de aprendizaje para la Regresión Lineal

En la Figura 8 se muestra la curva de aprendizaje para el algoritmo RandomForest implementado. Se observa que el error disminuye inicialmente, pero incrementa ligeramente al darle más datos al algoritmo. Esto da una pista acerca de la cantidad

de datos necesarios para entrenar el algoritmo de manera que su error sea bajo. Sin embargo, nuevamente se vuelve a evidenciar que existe *overfitting*.

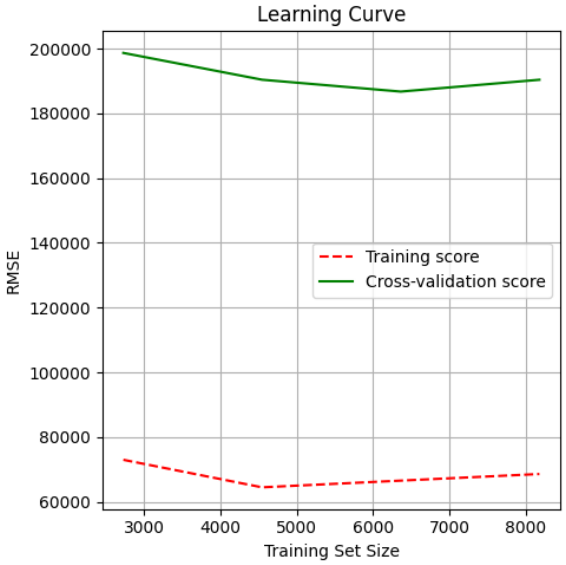

Figura 8. Curva de aprendizaje para el RandomForest

6.2. Métodos no supervisados

Para obtener la curva de aprendizaje de la combinación de los métodos no supervisados y supervisados, se implementa la misma metodología descrita anteriormente. La Figura 9 muestra la curva de aprendizaje para la combinación de PCA y RandomForest. Se observa que al igual que con los métodos supervisados, existe overfitting por lo que el problema puede deberse a la cantidad de datos utilizados para el training, en la selección de los modelos a utilizar.

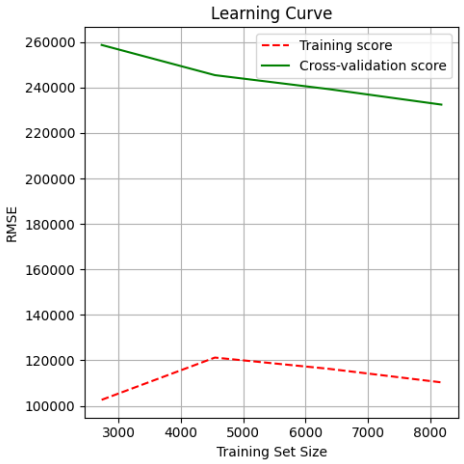

Figura 9. Curva de aprendizaje PCA+RandomForest

7. Retos y condiciones de despliegue modelo

Para evaluar el desempeño mínimo con el cual se consideraría que el modelo genera un beneficio, se debe comparar el precio generado por las predicciones del modelo con el precio real por el que se venderían las casas de Paris. Si el modelo permite obtener precios los cuales tanto compradores como vendedores reconozcan como apropiados, entonces se podría considerar que el modelo es apto para desplegarse en producción. Uno de los retos que se tiene para implementar este modelo, es que las edificaciones deben ser descritas según los parámetros definidos por el modelo, haciendo que no se puedan tener en cuenta elementos o características no empleadas.

8. Conclusiones

- Es necesario ajustar la selección y la complejidad de los modelos utilizados para reducir el *overfitting* dentro de los métodos aplicados.
- Es necesario hacer un análisis detallado de que variables tienen más peso en los modelos entrenados, ya que de esta manera se puede reducir el error.

- Lo métodos no supervisados pueden ser implementados como una forma de preprocesado de datos, que nos permita generar un dataset más compacto con las variables más significativas.

Bibliografía

- Kaggle. *Playground Series – Season 3, Episode 6*. Kaggle. (2023). https://www.kaggle.com/competitions/playground-series-s3e6/overview
- Raúl Ramos. *Ai4eng_example_project*. Github. (2022). https://github.com/rramosp/ai4eng_example_project