Ciencia de Datos

Trabajo Práctico Final

Profesores:

- Palazzo, Martín
- Aguirre, Nicolás
- Chas, Santiago

Integrantes:

- Chodaszewski, Luciano
- Neustadt, Tomás

Año: 2022

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Introducción y objetivos

En este informe se detalla el desarrollo de un modelo de regresión lineal realizado en el lenguaje de programación Python utilizando librerías para el preprocesamiento y análisis de datos tales como Pandas[1], Seaborn[2], Matplotlib[3] y Numpy[4], así también como la librería Scikit Learn[5] para el aprendizaje automático de modelos.

Partiendo de fundamentos y métodos clásicos de Data Science, se busca aplicar conocimientos para la realización de un modelo de Machine Learning que permita predecir el valor de una propiedad mediante aprendizaje supervisado. Para ello partiremos de un set de datos, al cual se le realizará un pre-procesamiento y posterior Análisis Exploratorio de Datos para entender su estructura y cómo se relacionan sus dimensiones.

Una vez procesado el set de datos, quedando así las features más relevantes que explican la estructura de los datos, crearemos distintos tipos de modelos de regresión lineal con varios hiper parámetros y también utilizando la reducción de la dimensionalidad para encontrar el modelo que mejor prediga, comparando entre sí el score y error cuadrático medio de los modelos generados.

Descripción del dataset

El dataset a utilizar está compuesto por 26 columnas ó features y 38.656 samples, resultando en un S2FR (Sample to Feature Ratio) mayor a 1.

A continuación se detallan los nombres de cada columna con su significado:

Variable	***Significado***
Unnamed:	Índice del dataset
id	Número identificatorio de la publicación
ad_type	Tipo de publicación
start_date	Fecha de inicio de la publicación
end_date	Fecha de fin de la publicación
created_on	Fecha creada de la publicación
lat	Latitud de la propiedad
lon	Longitud de la propiedad
l1	Ubicación 1 de la propiedad
l2	Ubicación 2 de la propiedad
l3	Ubicación 3 de la propiedad
l4	Ubicación 4 de la propiedad
l5	Ubicación 5 de la propiedad
l6	Ubicación 6 de la propiedad

[1] https://pandas.pydata.org/
[2] https://seaborn.no/
[3] https://matplotlib.org/
[4] https://numpy.org/
[5] https://scikit-learn.org/stable/

rooms	Cantidad de ambientes de la propiedad
bedrooms	Cantidad de habitaciones de la propiedad
bathrooms	Cantidad de baños de la propiedad
surface_total	m2 total de la propiedad
surface_covered	m2 cubiertos de la propiedad
precio	Precio de la propiedad
currency	Moneda del precio
price_period	periodo del precio
title	Título de la publicación
description	Descripción de la publicación
property_type	Tipo de la propiedad
operation_type	Tipo de operación

Para su preprocesamiento, se eliminaron en una primera instancia todas las columnas con valores nulos, siendo las mismas l4, l5, l6 y price_period.

Luego se eliminaron todos los samples (filas) que contengan al menos un valor nulo en las columnas lat, lon, bathrooms y l3.

También se filtraron las columnas que tuvieran un único valor, ya que no tienen relevancia en la estructura de los datos, siendo ad_type, l1, l2, currency, operation_type.

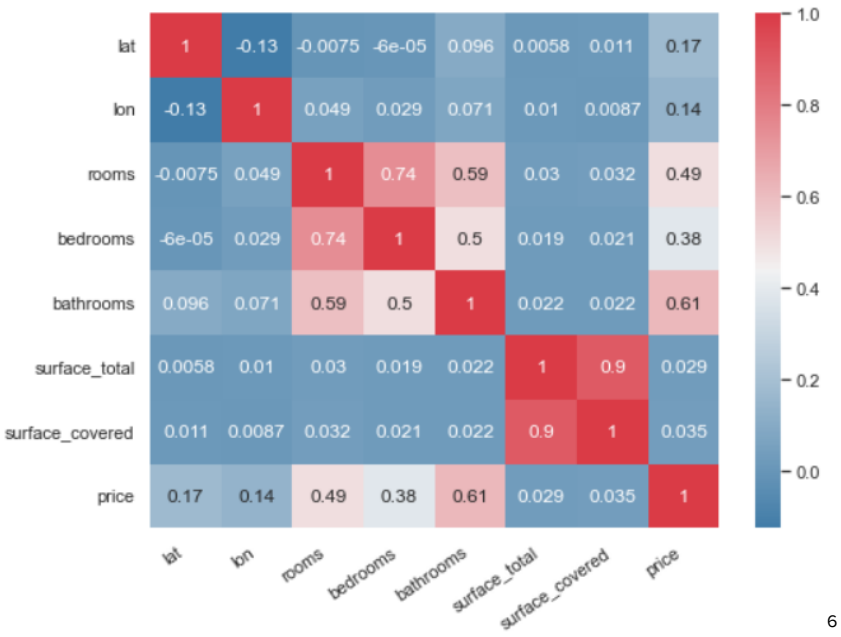
Se buscaron valores repetidos en la columna de identificación única de cada publicación (id), quitándose del dataset los samples con id repetido. Una vez hecho esto, se quitaron las columnas id, start_date, end_date, created_on, title y description ya que tampoco aportan relevancia para el modelado de la regresión lineal.

Finalmente se reemplazaron los valores nulos en columnas donde los hubiese para evitar errores de formatos.

Análisis exploratorio de datos

Una vez filtrado el dataset, se realizó un análisis de los datos del mismo con el fin de determinar cómo se comportan las variables entre sí para explicar la estructura.

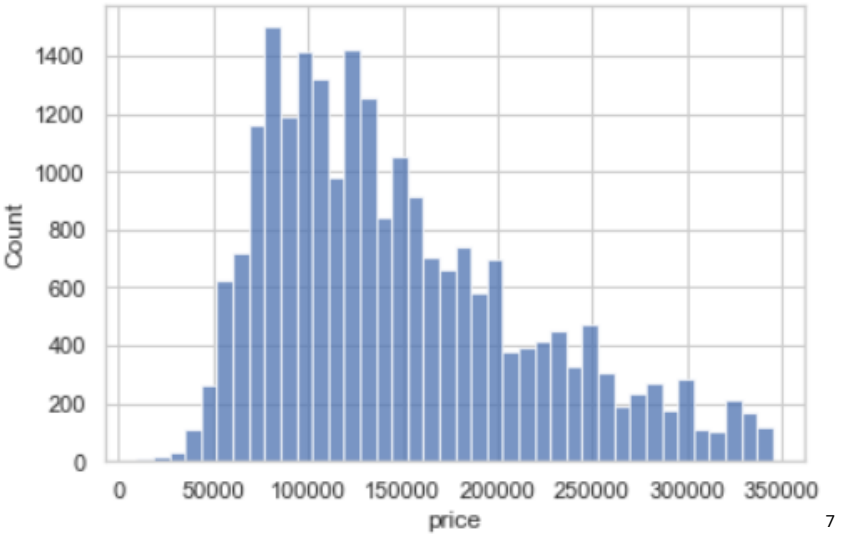
En primera medida se realizó una matriz de correlación lineal entre todas las variables:



[6]

Observamos como las features rooms y bathrooms tiene la mayor correlación con la feature price, es decir que son 3 features que influyen en gran medida en el precio de la propiedad.

Un histograma de la variable price con bins de tamaño 40 refleja la distribución empírica de los precios de las casas según su tasa de ocurrencia en el modelo, con una fuerte tendencia hacia propiedades entre 75.000 y 150.000 dólares.



[7]

Finalmente mediante un countplot graficamos los top 5 barrios donde más se encuentran las propiedades del dataset, siendo las mismas Palermo, Caballito, Belgrano, Almagro, y Villa Crespo.

[6] Matriz de Correlación del EDA
[7] Histograma de frecuencias del EDA

[8]

Materiales y métodos (algoritmos utilizados)

En primera instancia, el primer paso del pipeline llevado a cabo es la separación del dataset en variables de train y test para entrenar el modelo y luego medir su performance.

Para ello se debe primero crear las variables dummies de las features categóricas l3 y property_type que quedaron en el dataset utilizando la funcionalidad de la librería Pandas *get_dummies*.

Luego del concatenado de dichas variables al dataset y eliminado de las columnas categóricas, se separó en train y test con una posterior estandarización de los datos con la función *preprocessing.StandardScaler()*.

Finalizada la conformación de los dataset de train y test, seguimos con la generación de los modelos, para lo cual utilizaremos un modelo KNNeighbors Regressor y un Support Vector Regression.

El método KNN Regressor es un modelo particular utilizado para regresión lineal que parte de la base de que utiliza distancias euclídeas como la similaridad entre features para predecir a través de un factor K que determina la cantidad de puntos vecinos que comparten similaridades en su features. De esta manera el KNNeighbors determina el K óptimo a través de un Grid Search.

El método SVR busca predecir nuevos puntos de la forma más cercana a la función teórica f(x) que explica las relación entre las variables con un costo epsilon que permite un error de ± ε de la f(x).

Para la profundización en la investigación de modelos de regresión, se incluyó la utilización del método de Principal Component Analysis (PCA) para la reducción de la dimensionalidad del dataset y la posterior modelización con el fin de analizar si la reducción de la dimensionalidad en nuestro dataset nos permite encontrar un modelo con mayor accuracy (score) y menor error cuadrático medio. La cantidad de dimensiones deseadas es un parámetro seleccionado de forma manual. Para la reducción de la dimensionalidad se parte de la base de una matriz de correlación de las variables del dataset, para luego calcular los autovectores y los autovalores de cada uno de ellos para la posterior selección de una cierta

[8] Histograma de frecuencias de aparición de propiedades

cantidad de dimensiones deseadas que explique de la mejor forma el modelo, siendo estos los autovalores con mayor tamaño.

Experimentos y resultados

KNNeighbors

Para el modelo KNNeighbors se utilizó un Grid Search y Cross Validation de valor 5 para hallar los mejores hiper parámetros que devuelvan la mejor regresión

Con un total de 20 fits, el valor hallado de K que mejor modela es de valor 14 para un error cuadrático medio de 37.880,65.

SVR

Incluyendo hiper parámetros como Kernels del tipo lineal y RBF, los cuales permiten el modelado mediante la deformación del espacio de D dimensiones de features para encontrar un hipercubo lineal en este caso que permita la predicción. También se añadieron valores de 1, 10 y 100 para el hiper parámetro de costo C y valores de 0.1, 1 y 10 para el coeficiente de las funciones del Kernel.

Un total de 90 fits estimaron los siguientes como los mejores hiper parámetros: *{'C': 100, 'gamma': 0.1, 'kernel': 'linear'}*.

Un score de 0.5075 y error cuadrático medio de 47.183,49.

PCA

Utilizando sklearn como librería importamos el decomposition method, eligiendo un valor de 2 para las dimensiones deseadas. Se realiza el fit del dataset de train y la transformación de xtrain y xtest, resultando en un variance_ratio de [0.33370439, 0.31367324] para cada una de las dimensiones obtenidas.

KNNeighbors con PCA

De igual manera, se entrena el modelo de KNNeighbors pero ahora utilizando nuestro nuevo dataset con dimensionalidad reducida.

Se obtiene un valor de K = 14 y un error cuadrático medio de 39.996,30.

SVR con PCA

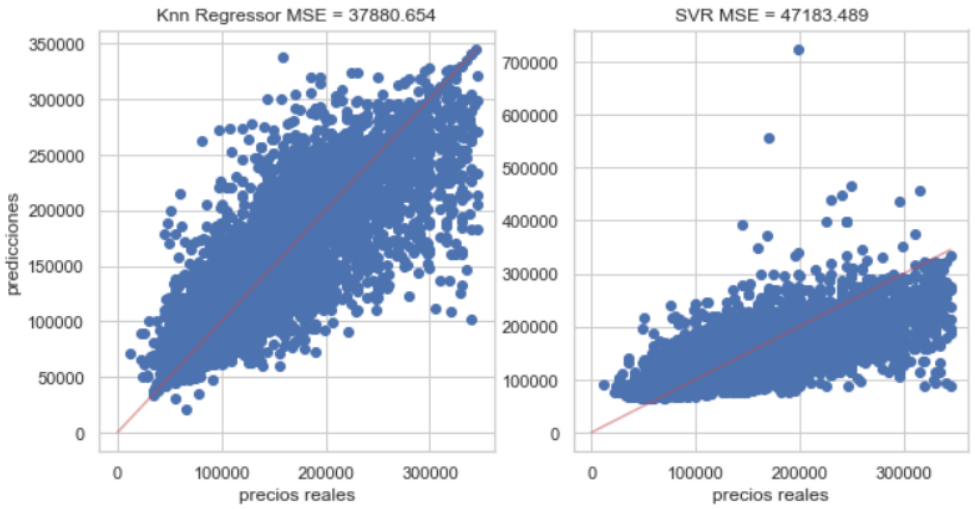
Se realiza la modelización con el dataset de dimensionalidad reducida igual que en KNNeighbors pero en el modelo de SVR con nuevos hiper parámetros para el Grid Search.

Un total de 72 fits estimaron los siguientes como los mejores hiper parámetros: *{'C': 100, 'gamma': 0.1, 'kernel': rbf}*.

Un score de 0.4274 y error cuadrático medio de 50.962,64

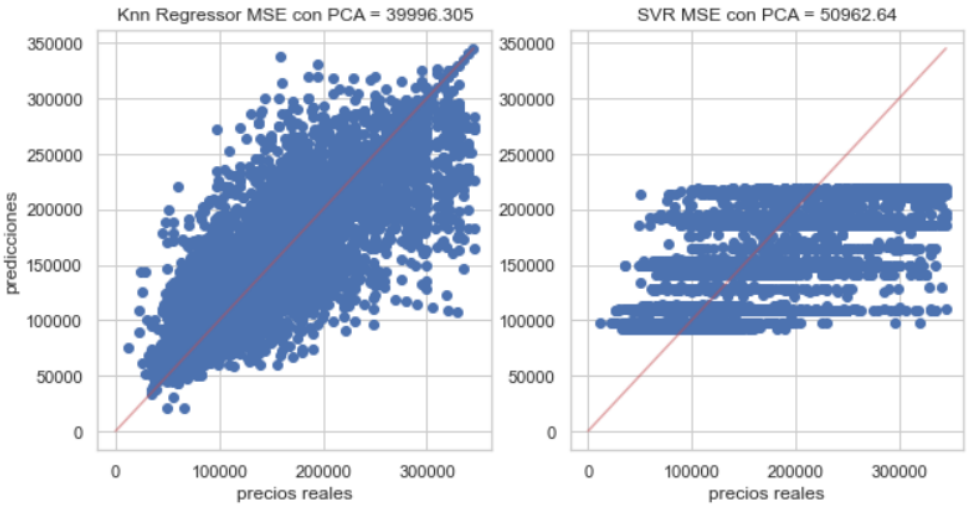
Discusión y conclusiones

KNNeighborsRegressor vs. SVR (sin PCA)



Se puede apreciar que la comparación de la performance del MSE entre ambos modelos determina que el KNNeighbors se ajustó de mejor manera a nuestro modelo ya que el MSE es menor.

KNNeighborsRegressor vs. SVR (con PCA)



Incluyendo el PCA en el pipeline para la reducción de la dimensionalidad, vemos cómo esto provocó un aumento en el MSE tanto de la KNNeighbors como de la SVR sin PCA.

De esta forma podemos concluir que el mejor modelo de los 4 realizados es el KNNeighbors sin PCA, ya que su performance es la mejor de los 4 modelos generados.

Referencias

- Support Vector Regression (SVR) por Jacob Avila Camacho[9]
- What Is K-Nearest Neighbor? An ML Algorithm to Classify Data por Amal Joby photo by Amal Joby[10]
- Análisis de Componentes Principales (Principal Component Analysis, PCA) por Joaquín Amat Rodrigo[11]
- scikit-learn[12]

[9]https://www.jacobsoft.com.mx/es_mx/support-vector-regression/
[10] https://learn.g2.com/k-nearest-neighbor
[11] https://www.cienciadedatos.net/documentos/35_principal_component_analysis
[12] https://scikit-learn.org/stable/