Predicción del valor de la renta en bogotá

CHRISTIAN NICOLÁS OSPINA LÓPEZ

##### 31 DE AGOSTO DEL AÑO 2020

Contenido

[RESUMEN 3](#_Toc49795112)

[INTRODUCCIÓN 4](#_Toc49795113)

[OBJETIVOS 5](#_Toc49795114)

[DESARROLLO DEL MODELO 6](#_Toc49795115)

[RESUMEN DE LOS DATASETS 6](#_Toc49795116)

[PROCESO DE CONSTRUCCIÓN DEL DATAFRAME 8](#_Toc49795117)

[SELECCIÓN DE VARIABLES 10](#_Toc49795118)

[SELECCIÓN DEL MODELO 12](#_Toc49795119)

[REGRESIÓN LINEAL SIMPLE 12](#_Toc49795120)

[MODELO LOG-POLINOMIO 13](#_Toc49795121)

[RIDGE Y LASSO 14](#_Toc49795122)

[RED NEURONAL 15](#_Toc49795123)

[RESULTADOS FINALES 16](#_Toc49795124)

[DISCUSIÓN 18](#_Toc49795125)

[CONCLUSIONES 19](#_Toc49795126)

[REFERENCIAS 20](#_Toc49795127)

[ANEXOS 21](#_Toc49795128)

# RESUMEN

La ciudad de Bogotá es la capital de Colombia, ubicada geográficamente en el centro del país se caracteriza por ser el centro político, financiero y económico más grande del país. Según la última encuesta realizada en 2018 por el Departamento Administrativo Nacional de Estadística (DANE, 2019), la población de la ciudad es de alrededor de 7’200.000 habitantes lo cual la hace la ciudad más poblada. Estos aspectos hacen de Bogotá, un lugar muy demandado no solamente para vivir sino también para albergar personas que deben hospedarse en la ciudad por motivos de turismo o negocios en el país.

La renta o el arriendo se pactan por el mercado y básicamente no hay una regla general o una norma que pueda dictaminar este precio, lo que hace que las personas que decidan generar esta renta tengan que hacer una investigación de mercado para decidir cual debe ser el valor óptimo, tal que, sea atractivo para la persona que va a arrendar y a su vez sea la cantidad justa para el dueño de la propiedad.

Por otro lado, existe un problema que está arraigado por la demanda a estas grandes ciudades y es que generalmente al ser centros muy atractivos para las personas, cada año están recibiendo miles de personas de otros lugares y su demanda cada vez es mayor, de manera que la finca raíz cada año va incrementando los precios de la vivienda, lo que a su vez eleva los precios de los arriendos, de manera que cada año debe estar ajustándose estos valores.

Con todo esto en mente, el presente trabajo está enfocado en crear un modelo que prediga cual debe ser el valor de un arriendo o renta de vivienda en la ciudad de Bogotá, dependiendo de diferentes factores que pueden llegar a influir en cual debería ser la cifra óptima según el mercado.

# INTRODUCCIÓN

El presente trabajo pretende abordar un problema que se ha venido presentado en las grandes ciudades, las cuales reciben cada año a miles de personas de otros lugares y se enfrentan a tener un mercado inmobiliario muy cambiante, en el cual se demanda cada vez más la vivienda y los arriendos suelen ser difíciles de conciliar. De manera que, el propósito principal es crear un modelo que pueda solucionar este problema de predecir el valor óptimo de un arriendo, específicamente en la ciudad de Bogotá, la cual es considerada como una de las más importantes no solo de Colombia sino también de Latinoamérica.

Este modelo se alimentó de un dataset sobre los arriendos promedio de Airbnb, el cual cuenta con variables como, la tasa de ocupación, el barrio, el número de habitaciones promedio del sector, el número de baños promedio, la media de estos arriendos, entre otras, y por otro lado se incluyeron las coordenadas del barrio, las cuales también fueron importantes para determinar el valor de este arriendo ya que dependiendo el sector de la ciudad es más o menos atractivo para las personas.

La característica principal de este tipo de problemas es que sus resultados son un número, por lo que nos enfrentamos a un problema de regresión y el modelo está enfocado a predecir este valor numérico, de manera que, después de diferentes validaciones y modelos probados, el modelo más robusto fue el que finalmente se escogió en términos del error mínimo promedio y su capacidad de predicción que en este caso más precisamente fue una red neuronal validada por medio de la validación cruzada.

# OBJETIVOS

Los objetivos de este proyecto están enfocados a la predicción u optimización del valor de un arriendo, en la ciudad de Bogotá, enfocado también en realizar un análisis sobre el mercado inmobiliario en la ciudad y como está sectorizada la ciudad en función de este. Así pues, los objetivos son:

* Identificar las variables más importantes a la hora de predecir el valor de un arriendo diario en términos de cuales pueden alimentar de forma óptima el modelo.
* Atribuir al valor del arrendamiento diario, la ubicación del barrio específicamente las coordenadas.
* Identificar mediante el comportamiento de los datos, cuales pueden ser las posibles transformaciones que se deben hacer a los mismos para poder correr de forma óptima el modelo.
* Predecir cual debe ser el valor de una renta diaria en la ciudad, identificando también, cuando un predio esta sobre valorado y cuando esta subvalorado según el mercado.
* Identificar el modelo más preciso, tal que, no sufra de sobreajustes y se cumplan los supuestos.

# DESARROLLO DEL MODELO

## RESUMEN DE LOS DATASETS

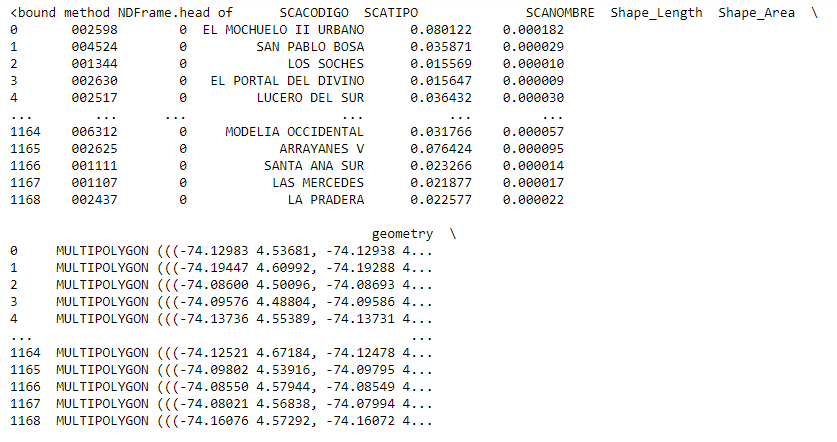
Para el modelo, se tuvieron en cuenta 2 fuentes de información diferentes, la primera es la base de datos “barrios\_airbnb-bogota-2015-2019-mensual.xls” es un Excel que contiene las variables de Airbnb por barrios y la segunda es “scatgeojson.geojson”, la cual es un archivo geojson que contiene las coordenadas de los barrios.

El primer dataset es una recopilación por mes de algunas variables numéricas y calificativas por barrio en la ciudad de Bogotá, esta fue obtenida del repositorio de datos abiertos publicados por el distrito en su página web y se encuentran los agregados mensuales desde el año 2015 hasta el año 2019. A continuación, se muestra el detalle presentado dentro del dataset, sobre cada una de las variables contenidas en él.

|  |  |
| --- | --- |
| **Variable** | **Definición** |
| Cod\_Barrio | Identificador único del Sector Catastral (Barrio) |
| Barrio | Nombre del Sector Catastral (Barrio) |
| Vigencia | Año en el cual fue creado el anuncio |
| Fecha\_Corte | Mes de reporte |
| Tipo\_Aloj(M)1 | (Moda estadística) del tipo de alojamiento por Sector Catastral |
| Tipo\_Alqu(M) | Tipo de alquiler; refiere el tipo de alquiler (Moda) en el Sector Catastral |
| Habitaciones(P)2 | Número promedio de habitaciones en el Sector Catastral (Barrio) |
| Tasa\_O2(P) | Promedio de la Tasa de Ocupación; Obedece al promedio de: Suma de días reservados en el mes por barrio / Suma de días disponibles en el mes por barrio |
| Ingreso\_USD(S)3 | Suma por Sector Catastral (Barrio), de los ingresos totales (en dólares estadounidenses) obtenidos durante el período del informe. Incluye el precio anunciado desde el momento de la reserva, así como los gastos de limpieza. |
| Ingreso\_Hab\_Dia (S) | Suma por Sector Catastral (Barrio), de los ingresos diarios por habitación (en dólares estadounidenses) |
| ADR\_USD(P) | Tarifa diaria promedio (ADR) de noches reservadas en USD. ADR = Ingresos totales / Noches reservadas |
| Días\_Reservados(P) | Número promedio de días calendario clasificados como reservados por Sector Catastral (Barrio) |
| Días\_Disponibles (P) | Número promedio de días calendario clasificados como disponibles por Sector Catastral (Barrio) |
| Baños (P) | Número promedio de baños por Sector Catastral (Barrio) |
| Max\_Huéspedes (P) | El número máximo promedio de huéspedes que la propiedad de alquiler de vacaciones puede acomodar |
| Deposito\_USD(P) | Promedio del depósito de seguridad en dólares estadounidenses en el Sector Catastral (Barrio) |
| Limpieza\_USD(P) | Promedio del coste de limpieza cobrado por reserva en dólares estadounidenses en el Sector Catastral (Barrio) |
| ExtraPeople\_USD(P) | Promedio de la tarifa de personas extra en dólares estadounidenses en el Sector Catastral (Barrio) |
| Cancelación(M) | (Moda estadística) de la política de cancelación de la lista de alquileres vacacionales por Sector Catastral (Barrio) |
| Public\_Noct\_USD(P) | Promedio de la tarifa nocturna predeterminada para una lista de alquileres de vacaciones en dólares estadounidenses |
| Fotos(P) | Número promedio de fotos publicadas por Sector Catastral (Barrio) |
| Calif\_Comunicación(P) | Valor promedio de la calificación por comunicación |
| Calif-Limpieza(P) | Valor promedio de la calificación por limpieza |
| Calif\_Localización(P) | Valor promedio de la calificación por localización |
| Calif\_Valores(P) | Valor promedio de la calificación Valo (Costo) |
| Mascotas(M) | (Moda estadística) del permiso o no de ingreso de animales de compañía |
| Comodidades(M) | (Moda estadística) según la cantidad de comodidades anexas |
| Localidad | Espacio geográfico en que se divide el área urbano y rural del municipio de Bogotá. |
| Cant\_Propiedades(S) | Suma por Sector Catastral (Barrio), de la cantidad de propiedades activas en el barrio durante el mes |
| Estrato(M) | (Moda estadística) del Estrato Socioeconómico |
| Percapita | Impuesto promedio por unidad predial: (Recaudo del predial por barrio/Número de unidades prediales en el barrio) |
| Media\_Arriendo | corresponde al canon promedio de arriendo resultado de la encuesta multipropósito 2017 realizada por el Departamento Administrativo Nacional de Estadística – Dane, en la cual se presentan los valores de arriendo por Unidad de Planeación Zonal – UPZ, estos valores se adoptaron para los barrios que integran cada UPZ |
| Observación | Indica si el barrio tuvo actividad o no durante el año. |

Nota: Recuperado de: <https://datosabiertos.bogota.gov.co/dataset/barrios_airbnb-bogota-2015-2019-mensual>, barrios\_airbnb-bogota-2015-2019-mensual.xlsx.

Por otro lado, para obtener la ubicación geográfica en términos de coordenadas de cada uno de los barrios también se obtuvo un dataset de la misma página del distrito, el cual es un archivo geojson que contiene las coordenadas en polígonos de cada uno de los barrios en la ciudad. La siguiente es un head de la tabla con sus correspondientes campos.



*Fuente: Elaboración propia*

Realmente de este dataset, precisamos el nombre del barrio y el campo geometry, el cual es el polígono de coordenadas que encierra al mismo. Es importante destacar también, que nuestra variable objetivo en este caso va a ser el campo ‘ADR\_USD(P)’ proveniente del primer dataset pues es el valor diario de la tarifa por noche.

## PROCESO DE CONSTRUCCIÓN DEL DATAFRAME

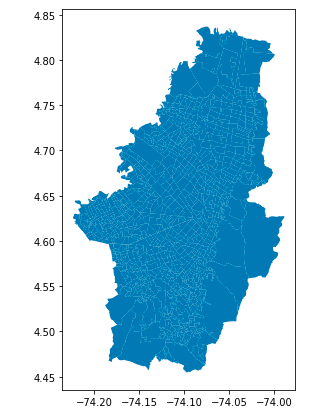
En primera instancia, dado que el archivo de Airbnb es un Excel que contiene las bases de datos distribuido en pestañas por cada uno de los años, se tuvieron que unir todos los años para condensarlos en un solo archivo de Excel llamado “CONSOLIDADO AIRBNB.xls”. En cuanto al otro archivo geojson, dado que las coordenadas que tenemos por barrio son polígonos, realmente esta entrada en el modelo la tenemos que resumir de alguna forma para poder dejarla en términos de variables continuas, de manera que, en primera instancia se calculó para cada barrio el centroide de su polígono y luego se separó la longitud y la latitud del centroide en 2 columnas nuevas, las cuales serán las entradas del modelo.

Una vez realizadas estas modificaciones, se unieron las tablas teniendo como llave el nombre del barrio, el cual es exactamente igual en las 2 bases de datos. A continuación, se encuentra el código mediante el cual se crean las columnas de centroides y se unen las 2 bases de datos:



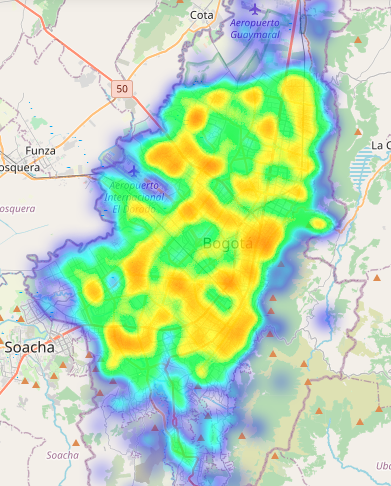
*Fuente: Elaboración propia*

En el código también se aprecia tiene como salida un mapa que se hace con los polígonos del scatgeojson. Se hace un filtro en la latitud debido a que la ciudad esta compuesta por una zona urbana y una zona rural, por lo tanto, se está mostrando solamente los barrios de la zona urbana.



*Fuente: Elaboración propia*

Es importante destacar que, en la base de datos de Airbnb no todos los barrios tienen un dato de arrendamiento, así pues, es importante tener en cuenta cuales son las zonas de la ciudad que tienen cobertura de Airbnb. A continuación, se muestra un mapa de calor de los lugares que tienen datos de arrendamiento en la ciudad.

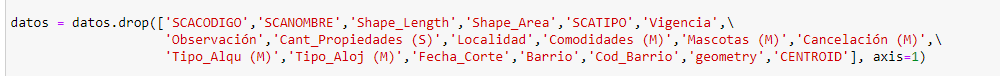


*Fuente: Elaboración propia*

Como se puede apreciar en el mapa, casi toda la ciudad tiene cobertura por la base de datos de Airbnb, de manera que es probable que nuestras variables de coordenadas pueden ser significativas en el modelo.

## SELECCIÓN DE VARIABLES

Para la selección de variables en primera instancia se eliminaron las columnas que por su naturaleza no deben hacer parte de la parte explicativa del modelo, hay tanto variables que se pueden clasificar como descriptivas y otras numéricas. El siguiente es un grupo de variables que se descartan por ser descriptivas.



*Fuente: Elaboración propia*

En este caso las variables SCACODIGO, SCANOMBRE, Shape\_length, Shape\_Area, SCATIPO, Vigencia, Observación, Localidad, Fecha\_Corte, Barrio, Cod\_Barrio, geometry y CENTROID

son columnas descriptivas que básicamente contienen los nombres del barrio, el código, la fecha, el área, el polígono y demás. Dado que nuestra variable de coordenadas ya contiene en sí la ubicación del barrio y en nuestro problema de regresión realmente no necesitamos tener el tamaño del sector o variables de fecha, por estas razones eliminamos desde un principio este conjunto de columnas.

En cuanto a las otras variables, Cant\_Propiedades (S), Comodidades (M), Mascotas (M), Cancelación (M) y Tipo\_Alqu (M) son en su mayoría modas estadísticas, es decir es la moda del dato en un mes para las propiedades de Airbnb, y al ser resultado de una moda, no se incluyeron en el modelo debido a que pueden sesgar un poco el resultado ya que no necesariamente una moda es lo que explica el valor del arriendo en el sector.

Por otro lado, se eliminan estas variables numéricas:



*Fuente: Elaboración propia.*

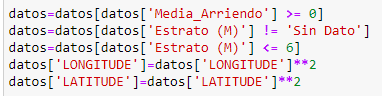
Las variables 'Días\_Reservados (P)' y 'Días\_Disponibles (P)' no se tienen en cuenta para el modelo ya que son el promedio del mes de la disponibilidad en el barrio la cual en este casi no es necesaria porque la variable independiente es el valor diario del arrendamiento.

En lo que se refiere a las columnas 'Fotos (P)', 'Calif\_Comunicación (P)', 'Calif\_Valores (P)', 'Calif-Limpieza (P)' y 'Calif\_Localización (P)' al ser un tipo de calificaciones promedio del mes desde 1 a 10, pueden también generar un sesgo ya que primero, no sabemos las desviaciones estándar y no es seguro atribuir una calificación promedio a un barrio y segundo porque el fin de nuestro modelo es que las variables que expliquen el arriendo sean datos de variables reales como el numero de baños, o el máximo de huéspedes, más que por el promedio de las calificaciones del sector o la cantidad de fotos de la propiedad.

Por último, las columnas 'Ingreso\_USD (S)' y 'Ingreso\_Hab\_Dia (S)' son los ingresos totales y diarios por habitación, de manera que claramente van a ser variables que están relacionadas entre si y es probable que expliquen en gran medida la variabilidad de la variable independiente al ser ingresos, por lo que es claro que es necesario quitarlas para evitar problemas de multicolinealidad.

## SELECCIÓN DEL MODELO

Una vez realizada la primera selección de variables explicativas se procede a realizar una selección del modelo más apropiado, al tratarse de un problema de regresión en primera instancia se evalúa un modelo de regresión lineal simple. Antes de correr los modelos, cabe destacar que se hicieron algunos filtros a los datos y se transformaron las siguientes columnas:



*Fuente: Elaboración propia*

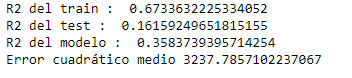
El filtro sobre la media de arriendo es debido a que existen datos en ceros para algunos barrios porque en el mes no tuvieron propiedades utilizadas en Airbnb, de manera que, estamos colocando solamente los barrios que tuvieron actividad.

La variable estrato es una clasificación que se hace en Colombia para identificar por medio de los ingresos de las personas del sector, si en este habitan personas de clase baja, media o alta, y esta clasificación son números del 1 al 6 siendo uno un sector con la clase baja y 6 el sector con la clase mas alta. Con esto en mente, es preciso colocar filtros para que se tengan en cuenta los barrios que cuentan con este número del 1 al 6 dado que, en el dataset existen por error algunos barrios que no tienen este dato o existen otros que tienen un numero mayor que 6, lo cual en la realidad no es posible tampoco.

Por último, se elevan al cuadrado las columnas que contienen la longitud y latitud debido a que estos números por naturaleza dentro de una ciudad tienden a ser muy similares y las diferencias entre extremos tienden a ser por decimales, de manera que, se elevan al cuadrado para que estas diferencias dentro de la ciudad sean más significativas en el modelo.

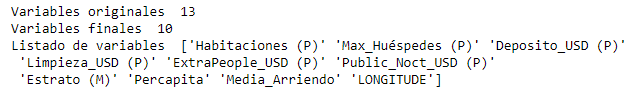
### REGRESIÓN LINEAL SIMPLE

Después de hacer estos filtros y las transformaciones pertinentes, ahora si se corre el primer modelo de una regresión lineal simple, El siguiente es el resultado de aquel primer modelo:



*Fuente: Elaboración propia*

Como se puede notar, solamente con el R2 del modelo en general y del test la variabilidad de la parte explicativa explica muy poco la variabilidad de la variable objetivo. Para verificar que variables pueden estar sesgando el resultado se hizo una prueba con el variance threshlod, para verificar que variables podemos eliminar porque no explican la variabilidad de nuestro dato objetivo. El siguiente fue el resultado:

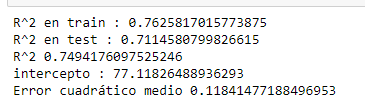


*Fuente: Elaboración propia*

Esto nos indica que la mayoría de las columnas son importantes en el modelo y realmente no es un problema estructural de este tipo. Para solucionar este R2 tan bajo fue necesario transformar las variables para que las escalas de estas no afectaran estos resultados y en cierta forma estuviéramos tratando cifras similares, esto porque nuestras variables tienen escalas muy diferentes y los arriendos de por si son números que pueden tener valores muy diferentes a la cantidad de habitaciones de un apartamento, por ejemplo.

### MODELO LOG-POLINOMIO

Teniendo en cuenta esta premisa, para el siguiente modelo se hizo una transformación polinomial en base 2 a las variables explicativas y se puso en logaritmo la variable independiente “ADR\_USD (P)”, quedando un modelo log-(polinomio en base 2). El siguiente es el resultado de esta nueva transformación.



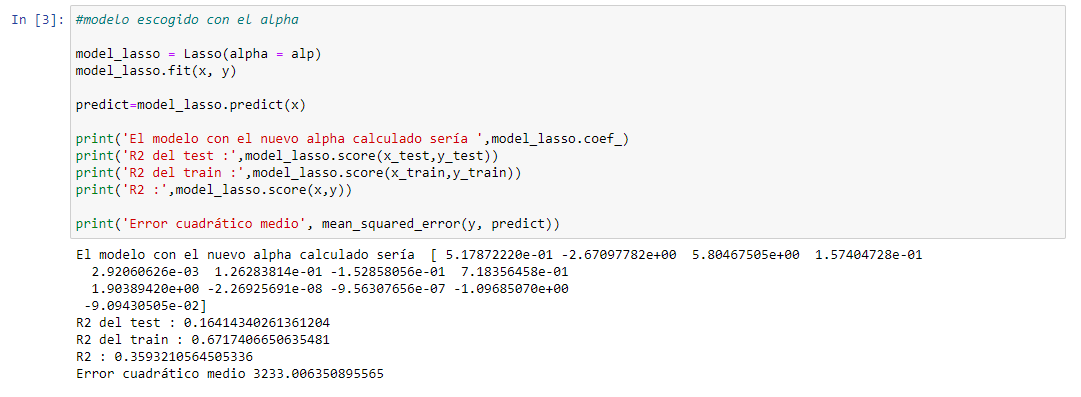
*Fuente: Elaboración propia*

Como se puede notar, el performance mejoró bastante tanto para el R2 como para el error cuadrático medio por lo que la solución al parecer funcionó.

### RIDGE Y LASSO

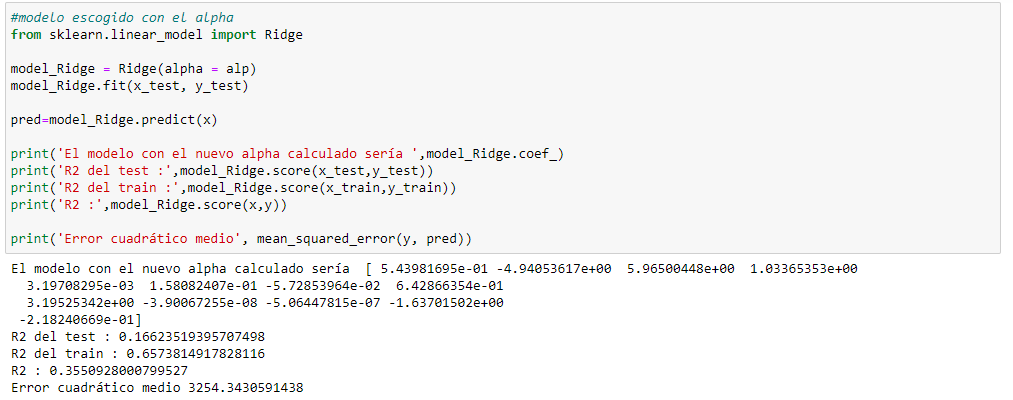
Otro modelo que se intento realizar fue el uno modelo de Ridge y otro de Lasso, entendiendo que estos también podrían tener un buen performance. A continuación, se muestra el resultado de los 2 modelos después de haber calculado el Alpha óptimo para cada uno:

#### LASSO



*Fuente: Elaboración propia*

#### RIDGE



*Fuente: Elaboración propia*

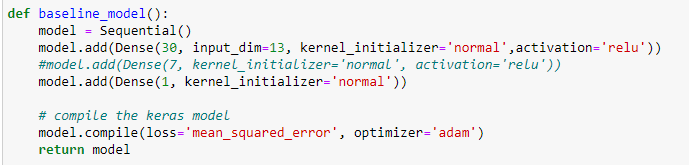
Al igual que en el primer modelo, tanto en Rigde como en Lasso no hay un buen performance ni en R2 ni en el error cuadrático que realmente es muy alto.

### RED NEURONAL

Después de haber probado diferentes modelos y de conocer un poco más el dataset, el modelo que finalmente se desarrolló y se escogió, fue una red neuronal mediante Keras, la cual es una herramienta para realizar redes neuronales profundas de forma muy práctica ya que se pueden añadir nodos y capas de manera sencilla, así como escoger las funciones de activación.

Antes de definir la red neuronal fue necesario hacer las transformaciones pertinentes que habíamos realizado anteriormente, de manera que, fue necesario dejar la variable independiente en logaritmo e igualmente para tratar estas escalas diferentes que tienen las variables explicativas, fue necesario estandarizar el dataset por medio de la herramienta StandarScaler. Adicionalmente se hizo una validación cruzada para dividir el dataset de forma aleatoria y verificar el performance de este.

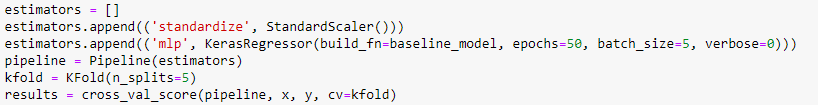
En primera instancia, se definió la red neuronal de la siguiente forma:



*Fuente: Elaboración propia*

Como se puede observar, se añadieron 2 capas, una capa de entrada con 30 nodos para las 13 variables explicativas y una capa de salida la cual es el valor del salario diario, es importante notar que al ser un numero entero que no necesariamente esta entre -1 y 1 la función de activación escogida fue la relu. Este modelo se escogió después de haber realizado pruebas con diferentes nodos y capas, se puede notar que en este fue óptimo crear una red con más nodos que una red con más capas.

Una vez definido el modelo, se definen los estimadores para alimentar el cross\_val\_score que en definitiva nos dará el performance del KerasRegressor el cual es el que en definitiva llama al modelo que ya se había definido. Para poder incluir dentro de esta validación las variables estandarizadas, se tuvieron que trabajar los estimadores por medio de un pipeline de la siguiente manera:



*Fuente: Elaboración propia*

### RESULTADOS FINALES

La validación cruzada en este caso está dividiendo el dataset en 5 porciones, de forma que, el 80% estima el modelo y el otro 20% es el test para evaluar el comportamiento. Los resultados fueron los siguientes:

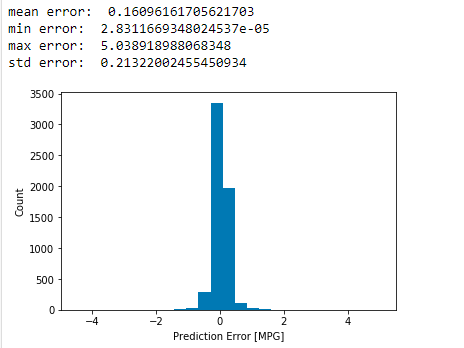


*Fuente: Elaboración propia*

El primero es el error medio y el error estándar del logaritmo de nuestra variable objetivo y el segundo resultado estandarizado es, también el error medio y el estándar después de aplicar el exponencial al logaritmo de “y”, es decir, sería el dato real del error en términos de dólares. Así pues, esto nos indica que el error medio es de 0.91 dólares, lo cual es bastante bajo, teniendo en cuenta que en nuestro dataset tenemos un promedio de 35,9 dólares como tarifa diaria en la ciudad para estos años.

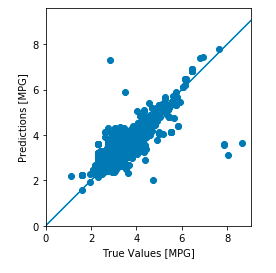
Haciendo un ejercicio de predicción con este pipeline creado que contiene los estimadores y es el que llama al modelo como tal, se separan los datos en train y test para validar de nuevo el comportamiento de los errores y saber que tan parecidas son las predicciones al dataset real. A continuación, se muestran los resultados:

#### Comportamiento de los errores



*Fuente: Elaboración propia*

#### Predicciones vs valor real



*Fuente: Elaboración propia*

Como se puede notar, los errores tienen un comportamiento que puede ser similar a la distribución normal y la mayoría de los errores tiende a ser cero lo que es bastante bueno porque cumplimos el supuesto de las regresiones en las que los errores deben tener esta distribución y garantizamos que la red neuronal esta prediciendo correctamente. Dado que, no hay datos en ceros y que los errores a pesar de que son muy bajos están en un rango entre 0 y 5, podemos concluir que el modelo no está sobreajustado puesto que al ser los resultados en logaritmos, hay datos atípicos que nos hacen intuir que el modelo no está replicando nuestro dataset.

Por otro lado, en nuestra segunda gráfica de predicciones versus valor real podemos corroborar lo que nos muestra el comportamiento de los errores, y es que el modelo predice de manera muy acertada y aproximada los datos que tenemos en el dataset. En definitiva, al trabajar con una red neuronal fue mucho más potente el resultado, tanto los errores como las predicciones fueron más acertadas y después de haber validado diferentes tipos de modelos este último fue el mejor modelo y el que finalmente se escoge porque tuvo un performance óptimo a comparación de los demás.

Con este modelo podremos no solamente tener una buena predicción sobre el arriendo óptimo que podríamos poner dentro de la ciudad, sino también para los consumidores o demandantes, podrían saber si el valor de un arriendo puede estar muy alto, en términos más técnicos ellos podrían validar si el arriendo está 2 desviaciones estándar por encima de la media del sector.

# DISCUSIÓN

Hay que tener en cuenta una limitación importante y es que este modelo fue realizado con un dataset que resumió la actividad de los Airbnb en cada uno de los barrios por mes, lo que influye mucho en el resultado pues desde la base de datos ya están resumida la información en términos de promedios y modas estadísticas. Claramente, si existiese un dataset con los datos concretos de cada una de las propiedades y su localización especifica sería mucho más potente el resultado del modelo.

Otra limitante que se presenta actualmente en el año 2020 es la coyuntura social que se dio por efectos del COVID-19 lo cual generará inconsistencias en la información de este año, es importante tener en cuenta que es probable que el modelo este un poco sesgado si se alimenta información actual y es necesario intentar generar de alguna forma un parámetro o un modelo más fuerte que pueda aprender de esta coyuntura que estamos viviendo.

Este modelo creado también abre las puertas a poder tratar estos datos desde otro tipo de modelos más potentes a las regresiones lineales simples y en concreto se podría también evaluar la posibilidad de haber creado una solución de regresión, a partir de un árbol de decisión u otro tipo de algoritmo que pueda igualar e incluso mejorar este modelo creado. Igualmente, se puede tener en cuenta algún dataset de seguridad por sector en la ciudad o población por barrios para alimentar de otras fuentes y verificar el impacto de estas nuevas variables dentro del valor de un arriendo en la zona.

# CONCLUSIONES

Se pudo constatar que, después de haber validado diferentes modelos, evaluado cada una de las variables y probado por métodos como el variance threshold para la selección de estas, se escogieron las variables óptimas en función de obtener el valor diario de arriendo, entendiendo también los campos con los que se contaban y los recursos de los que se disponían. Realmente esta selección pasó más por el tipo de dato que brindaba el dataset y pensando también en variables que no se tuvieran que consultar estrictamente de Airbnb.

En cuanto a incluir dentro del modelo la ubicación en coordenadas de cada uno de los barrios, fue posible y el modelo se adapto a estos nuevos campos ya que la variabilidad de estos explicaba en cierta forma el valor del arriendo en la ciudad. Aquí también se pudo constatar que la mejor transformación a los datos fue poner el valor de la variable independiente en logaritmos y a su vez quitar el efecto que tenía el tener campos en diferentes escalas, realmente sin estas transformaciones no hubiera sido posible crear un modelo que explique el valor del arriendo la información que se contaba.

Por último, realmente se pretendió dar respuesta al problema probando diferentes modelos y se intento buscar la forma de predecir este arriendo con el modelo más robusto posible, de manera que, después de haber estudiado diferentes opciones se pudo llegar a la mejor opción que en este caso fue la red neuronal, un modelo quizás más robusto que una simple regresión y que realmente tuvo un performance muy bueno sin caer en un sobreajuste.

Es probable que alimentar este modelo de nuevo con datos actualizados a 2020 puede generar algunos sesgos, sin embargo, valdría la pena evaluar algunas otras aproximaciones como una regresión mediante un árbol de decisión u algunos otros algoritmos existentes. Igualmente, el modelo podría tener un performance mucho mejor si se obtuviese información mas detallada, no el resumen mensual por barrio del cual alimentamos la red neuronal.

# REFERENCIAS

ARCGIS. (25 de Agosto de 2020). *ArcGIS.* Obtenido de https://desktop.arcgis.com/es/arcmap/10.3/analyze/commonly-used-tools/overlay-analysis.htm

Brownlee, J. (9 de Junio de 2016). *Machine Learning Mastery.* Obtenido de https://machinelearningmastery.com/regression-tutorial-keras-deep-learning-library-python/

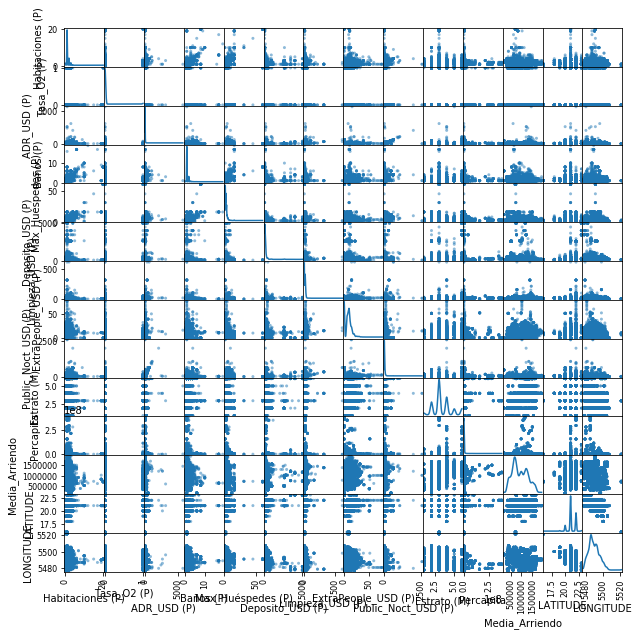
Brownlee, J. (24 de Julio de 2019). *Machine Learning Mastery.* Obtenido de https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/

DANE. (30 de Agosto de 2019). *Censo Nacional de Población y Vivienda.* Obtenido de https://sitios.dane.gov.co/cnpv/#!/

Instituto Distrital de Turismo. (30 de Diciembre de 2019). *datos abiertos bogota.* Obtenido de https://datosabiertos.bogota.gov.co/dataset/barrios\_airbnb-bogota-2015-2019-mensual

# ANEXOS

##### Comportamiento de cada una de las variables del modelo



##### CÓDIGO EN PYTHON DEL MODELO FINAL

import geojson

import tensorflow as tf

import numpy as np

from numpy import array

from keras.models import Sequential

from keras.layers import Dense

from keras.wrappers.scikit\_learn import KerasRegressor

from sklearn.model\_selection import cross\_val\_score

from sklearn.model\_selection import KFold

from sklearn.preprocessing import StandardScaler

from sklearn.pipeline import Pipeline

from sklearn.preprocessing import PolynomialFeatures

barrios=gpd.read\_file('scatgeojson.geojson')

base = pd.read\_excel('CONSOLIDADO AIRNBN.xlsx', sheet\_name='BASE')

barrios['CENTROID']=barrios['geometry'].centroid

barrios['LATITUDE']=barrios['CENTROID'].y

barrios['LONGITUDE']=barrios['CENTROID'].x

base['Cod\_Barrio']=base['Cod\_Barrio'].astype(str)

datos = pd.merge(base,barrios,left\_on='Barrio',right\_on='SCANOMBRE')

datos = datos.drop(['SCACODIGO','SCANOMBRE','Shape\_Length','Shape\_Area','SCATIPO','Vigencia',\

'Observación','Cant\_Propiedades (S)','Localidad','Comodidades (M)','Mascotas (M)','Cancelación (M)',\

'Tipo\_Alqu (M)','Tipo\_Aloj (M)','Fecha\_Corte','Barrio','Cod\_Barrio','geometry','CENTROID'], axis=1)

datos=datos[datos['Media\_Arriendo'] >= 0]

datos=datos[datos['Estrato (M)'] != 'Sin Dato']

datos['LONGITUDE']=datos['LONGITUDE']\*\*2

datos['LATITUDE']=datos['LATITUDE']\*\*2

datos = datos.drop(['Días\_Reservados (P)','Días\_Disponibles (P)','Fotos (P)','Calif\_Comunicación (P)',\

'Calif\_Valores (P)','Calif-Limpieza (P)','Calif\_Localización (P)','Ingreso\_USD (S)',\

'Ingreso\_Hab\_Dia (S)'], axis=1)

datos = datos.fillna(0)

for i in datos:

datos[i]=datos[i].astype(int)

datos=datos[datos['Estrato (M)'] <= 6]

x = datos.drop(['ADR\_USD (P)'], axis=1)

x = x.iloc[:,0:13]

y = datos.iloc[:,2]

y = np.log(y)

def baseline\_model():

model = Sequential()

model.add(Dense(30, input\_dim=13, kernel\_initializer='normal',activation='relu'))

#model.add(Dense(7, kernel\_initializer='normal', activation='relu'))

model.add(Dense(1, kernel\_initializer='normal'))

# compile the keras model

model.compile(loss='mean\_squared\_error', optimizer='adam')

return model

# fit the keras model on the dataset

#model.fit(X, Y, epochs=150, batch\_size=10, verbose=0)

# evaluate the keras model

#estimator = KerasRegressor(build\_fn=baseline\_model, epochs=100, batch\_size=5, verbose=0)

#kfold = KFold(n\_splits=10)

#results = cross\_val\_score(estimator, X, Y, cv=kfold)

#print("Baseline: %.2f (%.2f) MSE" % (results.mean(), results.std()))

# evaluate model with standardized dataset

estimators = []

estimators.append(('standardize', StandardScaler()))

estimators.append(('mlp', KerasRegressor(build\_fn=baseline\_model, epochs=50, batch\_size=5, verbose=0)))

pipeline = Pipeline(estimators)

kfold = KFold(n\_splits=5)

results = cross\_val\_score(pipeline, x, y, cv=kfold)

print("Standardized: %.2f (%.2f) MSE" % (results.mean(), results.std()))

print("Standardized: %.2f (%.2f) MSE" % (np.exp(results.mean()), np.exp(results.std())))

print(results)

##### CÓDIGO EN PYTHON DEL GRAFICO DE DISTRIBUCIÓN DE ERRORES

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

#print("Standardized: %.2f (%.2f) MSE" % (np.exp(results.mean()), np.exp(results.std())))

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,random\_state=1)

pipeline.fit(x\_train, y\_train)

prediction = pipeline.predict(x\_test)

train\_error = np.abs(y\_test - prediction)

mean\_error = np.mean(train\_error)

min\_error = np.min(train\_error)

max\_error = np.max(train\_error)

std\_error = np.std(train\_error)

error = y\_test - prediction

print("mean error: ", mean\_error)

print("min error: ", min\_error)

print("max error: ", max\_error)

print("std error: ", std\_error)

#train\_error.plot()

#estimator.score(x, y)

plt.hist(error, bins = 25)

plt.xlabel("Prediction Error [MPG]")

\_ = plt.ylabel("Count")

##### CÓDIGO EN PYTHON DE LA PREDICCION VS REAL

from tensorflow import keras

plt.scatter(y\_test, prediction)

plt.xlabel('True Values [MPG]')

plt.ylabel('Predictions [MPG]')

plt.axis('equal')

plt.axis('square')

plt.xlim([0,plt.xlim()[1]])

plt.ylim([0,plt.ylim()[1]])

\_ = plt.plot([-100, 100], [-100, 100])

##### CÓDIGO EN PYTHON PARA VER LOS POLIGONOS DE LOS BARRIOS

#hacer mapa

import pandas as pd

import geopandas as gpd

from geojson import Polygon

#from matplotlib.pyplot import figure

##se obtienen las fuentes de datos principales y se crean los centroides de cada p

barrios=gpd.read\_file('scatgeojson.geojson')

base = pd.read\_excel('CONSOLIDADO AIRNBN.xlsx', sheet\_name='BASE')

barrios['CENTROID']=barrios['geometry'].centroid

barrios['LATITUDE']=barrios['CENTROID'].y

barrios['LONGITUDE']=barrios['CENTROID'].x

#barrios['SCANOMBRE']=barrios['SCANOMBRE'].astype(str)

base['Cod\_Barrio']=base['Cod\_Barrio'].astype(str)

datos = pd.merge(base,barrios,left\_on='Barrio',right\_on='SCANOMBRE')

#datos.head

#barrios.head

barrios2=barrios[barrios['LATITUDE'] >= 4.469118]

barrios2.geometry.plot(figsize=(10, 7))

##### CÓDIGO EN PYTHON DEL MAPA DE CALOR INTERACTIVO

import folium

from folium import Choropleth, Circle, Marker

from folium.plugins import HeatMap, MarkerCluster

m = folium.Map(location=[4.6176,-74.0626], zoom\_start=11)

#HeatMap(data=releases[['LATITUDE', 'LONGITUDE']], radius=15).add\_to(m)

#for idx, row in datos.iterrows():

# Marker([row['LATITUDE'], row['LONGITUDE']]).add\_to(m)

#m

HeatMap(data=datos[['LATITUDE', 'LONGITUDE', 'ADR\_USD (P)']].groupby(['LATITUDE', 'LONGITUDE']).mean().reset\_index().values.tolist(), radius=8, max\_zoom=13).add\_to(m)

M

##### ARCHIVO DE EXCEL ORIGINAL

Este Excel se encuentra adjunto en el archivo zip enviado llamado “barrios\_airbnb-bogota-2015-2019-mensual.xls”.

##### ARCHIVO DE AIRBNB CON LOS AÑOS CONSOLIDADOS

Este Excel se encuentra adjunto en el archivo zip enviado llamado “CONSOLIDADO AIRNBN.xls”.

##### ARCHIVO DE BARRIOS DE BOGOTÁ POR POLIGONOS

Este Excel se encuentra adjunto en el archivo zip enviado llamado “scatgeojson.geojson”.

##### ARCHIVO PYTHON CON TODOS LOS MODELOS Y GRÁFICAS

Este Excel se encuentra adjunto en el archivo zip enviado llamado “TRABAJO FINAL MASTER.ypynb”.