Modelos de Predicción de Duración de Viajes BA Ecobici

Sofia Sapienza y Milagros Bochor
Universidad Tecnológica Nacional
Facultad Regional Buenos Aires
Cluster AI 2020

***Abstract*—El objetivo del siguiente artículo es analizar datos del sistema de Ecobicis de la Ciudad Autónoma de Buenos Aires con el fin de encontrar patrones de comportamiento que sean útiles para el análisis y mejora del mismo.**

***Keywords*—Ecobicis, CABA, Argentina**

I. Introducción

Partiendo de data sets que contienen información de cada uno de los viajes realizados utilizando el sistema de Ecobicis, y agregando a los mismos información relevante sobre el usuario o viajante, buscamos analizar algunos de los aspectos más relevantes del servicio de Ecobicis, tales como la evolución de la cantidad de viajes a lo largo del tiempo, las principales estaciones y la distribución de la duración de los viajes, entre otras. Una vez analizado el contexto de los datos, armamos un modelo que intenta predecir la duración de un viaje en Ecobici basado en las condiciones en las que se realizó el mismo.

II. Preparación de los datos

A. Elección de los datasets

Para el siguiente trabajo utilizamos datos públicos del sistema de Ecobicis de la Ciudad Autónoma de Buenos Aires, disponibles en: https://data.buenosaires.gob.ar/. Particularmente utilizamos datasets de recorridos realizados en bicicleta para los años 2018, 2019 y 2020 (con datos hasta 09/20), los cuales contienen información de id del usuario que realizó el viaje, latitud y longitud de las estaciones de origen y destino, fecha de inicio y finalización y duración, entre otras.

A su vez, para obtener información relevante de viajante, incorporamos datos de usuarios creados durante 2015, 2016, 2017 y 2018 para así obtener features como el género y la edad.

Incorporamos también un dataset calendario que indica los días de la semana por fecha y establece una columna binaria de ceros y unos según si se trató de un día laborable o no.

Por último, y con el objetivo de incorporar la feature de barrio destino y origen al modelo y además armar una visualización, agregamos también un dataset de estaciones, el cual se construyó a partir de datos de estaciones del nuevo y viejo sistema de Ecobicis, también disponibles en la web de datos de CABA.

B. Limpieza de los data sets

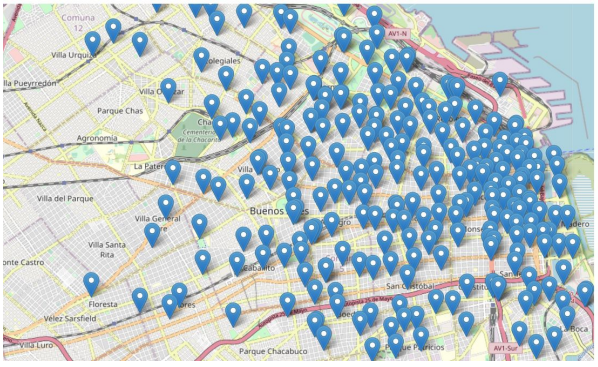
Dado que nuestro primer objetivo propuesto fue analizar la evolución de la cantidad de viajes realizados a lo largo de los años, armamos una tabla con recorridos totales, pero para ello primero debemos limpiar y organizar los datasets, dado que no tenían la misma cantidad de features, había nulos y muchos de los datos tenían caracteres al final como ceros o guiones. La metodología seguida fue conservar las features que eran comunes a los 3 sets y unificar todas ellas a mismas unidades y tipo de datos, eliminando además caracteres adicionales. Dentro de los cambios más relevantes para el presente trabajo, convertimos la variable duración de horas, segundos y minutos de los datasets de recorridos de 2018 y 2019 a segundos, tal como ya estaba expresada para los datos de 2020. Además, utilizamos la hora de inicio del viaje para crear una nueva feature llamada franja horaria, con 24 valores, uno por cada hora del día. Los nulos decidimos eliminarlos, dado que en todos los casos representaban menos del 1% de los datos.
Dada la dimensión de los datasets (2.619.968 para 2018, 6.367.314 para 2019 y 1.438.855 2020) debimos realizar estos pasos de limpieza por separado, como también el merge con las tablas soporte:

- Utilizando como índice la fecha origen del recorrido anexamos el calendario para poder saber en qué día de la semana se produjo el viaje y si fue un día laborable o no.
- Utilizando como índice la feature del id de estación origen y destino anexamos la tabla estaciones para poder saber los barrios de inicio y finalización del viaje.
- Utilizando como índice la feature id del usuario, anexamos la tabla de usuarios. Este paso solo fue realizado para la tabla 2018 ya que sólo se tenían datos de usuarios hasta ese período.

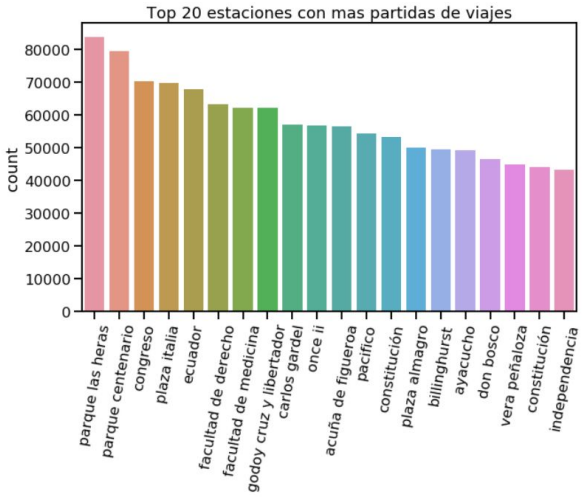
Una vez finalizado este proceso para cada año, formamos una única tabla de recorridos concatenando los datasets.

III. Exploración y análisis de datos

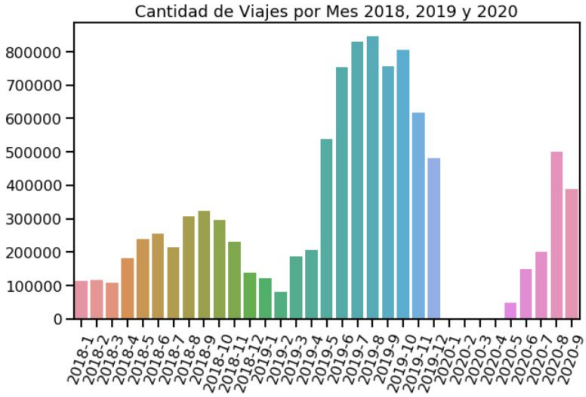
Lo primero que hicimos fue visualizar la totalidad de las estaciones del sistema en un mapa interactivo utilizando la librería Folium:



Son muchísimas, entonces nos preguntamos cuáles eran las estaciones más populares. A continuación sólo mostraremos el top 20 para aquellas con más partidas, dado que salvo leves diferencias, el ranking era el mismo para estaciones con más partidas y arribos.

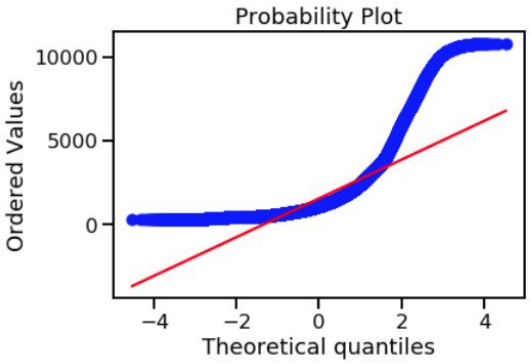


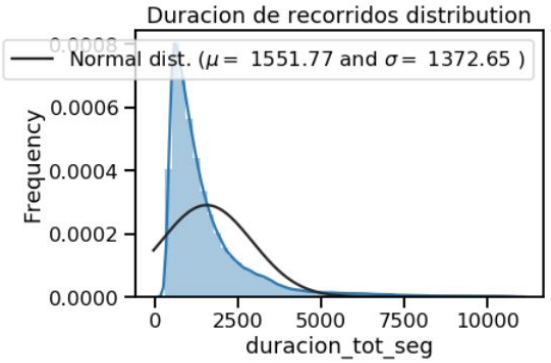
Luego, partiendo del insight de que el data set de 2019 tenía más del doble de datos que el del 2018, nos propusimos visualizar la evolución de la cantidad de viajes a lo largo de los años y obtuvimos como resultado lo siguiente:



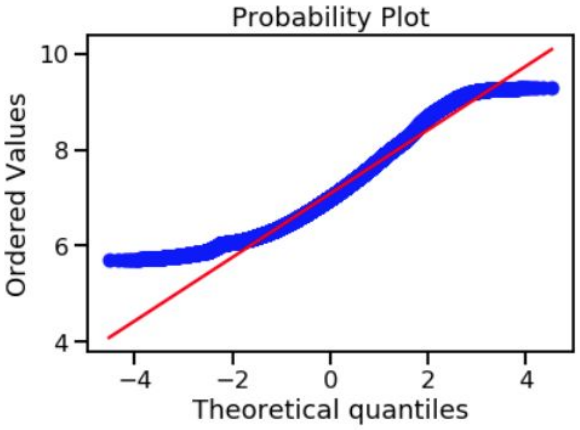
Efectivamente se puede evidenciar el aumento de los viajes de un año a otro. Además notamos que en los meses de alto verano la cantidad de viajes disminuye.

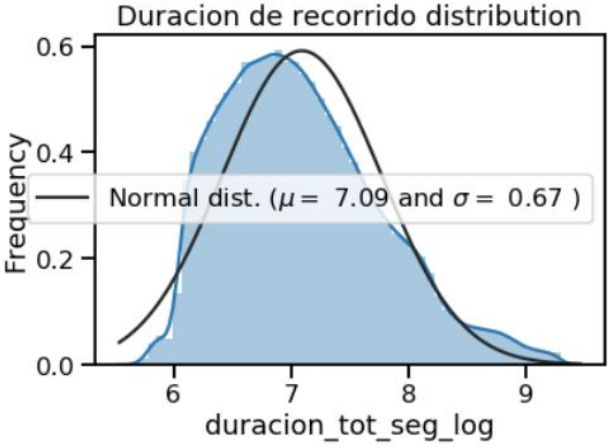
Luego de analizar todo este contexto, decidimos enfocarnos en la variable objetivo a predecir: **la duración del viaje**. Para ello analizamos en primera instancia la distribución de la misma. Utilizamos un QQ-Plot para ver qué tan bien o qué tan mal se ajusta la distribución original de los datos a una distribución conocida, en este caso una normal, y a su vez, ajustamos la duración en escala lineal a una normal para poder armar una visualización:





Tal como evidencian los gráficos, la variable duración en escala lineal no se ajusta a una distribución normal. Es por esto que pasamos a escala logarítmica la variable, y volvimos a armar el QQ-Plot [2] y a fitear los datos a una distribución normal para graficar y poder ver ahora los nuevos resultados de ajuste.

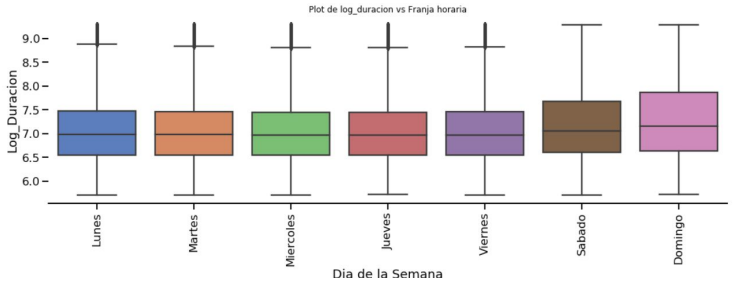




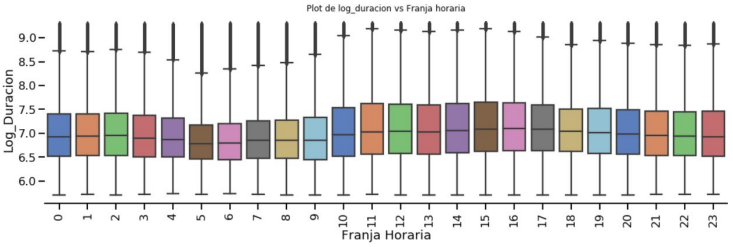
La duración en escala logarítmica muestra un buen ajuste a una distribución normal, por lo que de ahora en más utilizaremos esta feature.

Antes de comenzar con el modelo de predicción de la duración, la última y más importante pregunta que nos realizamos fue, ¿Cuáles de todas estas features que tenemos disponibles explican la duración de un viaje?

Realizamos un boxplot para visualizar la duración en escala logarítmica versus el día de la semana en el que se realizó el viaje.



Los viajes entre semana tienen la misma mediana. Los viajes de fin de semana suelen ser más largos, sobre todo los viajes de los domingos. Por último realizamos el mismo boxplot pero segmentando los datos por franja horaria. Los viajes de la madrugada suelen ser más cortos, y a partir de las 10 am y hasta las 18 pm son más largos.



IV. Selección del dataset para modelar

Luego de haber analizado los recorridos, las estaciones y los usuarios para tres años, decidimos quedarnos con los datos de Agosto de 2018, para los cuales tenemos información suficiente.

A. Eleccion de features

Analizando cuales son las variables que más nos explican la duración del recorrido, identificamos que tenemos dos tipos de features: continuas y discretas, por lo que, decidimos trabajarlas por separado para luego mandarlas a los modelos de regresión.

Respecto a las features continuas, seleccionamos:

- Latitud y longitud de estación origen
- Latitud y longitud estación destino
- Edad del usuario

Sobre las features categóricas, seleccionamos:

- Sexo del usuario
- Franja horaria
- Dia laboral o dia de fin de semana

B. Separación train y test

Ya sabiendo cuales son los atributos que vamos a enviarle al modelo, definimos al dataframe “X”, formado por las features y un dataframe “Y” que tendrá la variable a predecir: duración de un recorrido de bicicleta en la ciudad de Buenos Aires para el mes de Agosto de 2018. Luego separamos en train y test cada uno de estos, utilizando un 5% para entrenar y un 95% para testear, dando que la cantidad de datos que teníamos era muy grande. Para poder realizar esto importamos train test split de la librería Sklearn.

C. Tratamiento de features

C.1) Features categóricas: para poder trabajar este tipo de variables, necesitamos utilizar un objeto que nos permita tener cada “categoría” en columnas, ya que, si utilizamos una herramienta que únicamente codifique, podemos cometer el error de darle más peso a alguna variable, o asumir una cercanía entre ellas que no era verídica. Es por este motivo que decidimos utilizar dummies para el sexo del usuario (una columna para femenino y una para masculino), y para franja horaria (una columna por cada hora, desde las 00hs hasta las 23hs). En lo que respecta a la variable de Día laboral, al ser binaria, directamente la mantuvimos como estaba. De esta manera pasamos de tener 3 features, a tener 27.

C.2) Features continuas:

Para que los modelos a utilizar próximamente corran bien, tenemos que escalar las features continuas. Definimos un escalador de la librería Sklearn, llamado MinMaxScaler. Este objeto lo que hace es transformar las variables escalando cada una a un rango determinado. Este escalador lo fitteamos únicamente con las features continuas del dataframe del train, para que se ajuste a estos datos y recién luego podamos transformar el dataset de train y de test con el escalador ajustado correctamente.

Cabe aclarar que el proceso de tratamiento de las features lo repetimos tanto para las features lineales como las polinómicas. Para crear las polinómicas recurrimos nuevamente a Sklearn, al objeto PolynomialFeatures.

V. Modelos de regresión

Para abordar la problemática de predicar la duración de un recorrido decidimos optar por modelos de machine learning de aprendizaje supervisado. Este tipo de aprendizaje está enfocado para problemáticas en donde se conoce a las features y a las etiquetas, pero estas últimas son continuas. En nuestro caso, la duración del recorrido es una etiqueta de carácter continuo.

Decidimos iterar por diferentes modelos, entre los cuales elegimos:

- Regresión lineal

- Ridge Regression
- Support Vector Regression (SVR) [2]

Estos modelos los corrimos tanto para features lineales como para features polinómicas.

Las métricas que utilizamos para medir la performance de los modelos son:

- R^2 que muestra qué proporción de la varianza de las etiquetas reales es explicada por el modelo. Apuntamos a que este valor sea lo más cercano a 1.
- MSE (error cuadrático medio) que mide el error cuadrado promedio de las predicciones. Intentaremos que este valor sea lo más pequeño posible.
- MAE (media del error) que calcula el promedio de la diferencia absoluta entre el valor observado y los valores predichos. También intentaremos buscar el modelo que lo reduzca.

VI. Iteraciones de modelos

Para correr todos los modelos importamos de Sklearn, los objetos Linear SVR, Linear Regression y Ridge.

Para la elección de los mejores hiperparametros de cada modelo utilizamos Cross Validation [3] y Grid Search. La cross validation la realizamos en el dataset de entrenamiento, y definimos que divida el set en 5 porciones, para iterar esa cantidad de veces. Con esta técnica logramos simular el modelo para distintos hiperparametros, y poder elegir cuales son los mejores para estimar nuestra etiqueta.

La primera prueba la hicimos con el modelo lineal, que es aquel modelo que elige la familia de funciones lineales para separar a las muestras,calculando un peso w para cada feature. En primer lugar, corrimos para features lineales y obtuvimos los siguientes resultados:

	Model	Features	R2	MSE	MAE
0	Linear	Lineal	0.042368	0.435771	0.534492

Aunque no encontramos una correlación lineal entre las variables, no implica que no exista una correlación en otro grado. Por eso probamos con features polinómicas de diferentes grados. Estos fueron los resultados que obtuvimos:

1	Linear	Poly grado 5	0.163675	0.380570	0.472231
2	Linear	Poly grado 4	0.172195	0.376693	0.470192
3	Linear	Poly grado 3	0.164269	0.380300	0.476334
4	Linear	Poly grado 2	0.164721	0.380094	0.476191

Concluimos en que, de todas estas iteraciones con el modelo lineal, es cuando utilizamos features polinómicas de grado 4.

Continuamos probando el Ridge regression, que es el modelo que restringe al modelo lineal. Mediante un penalizador, va a ser más o menos estricto respecto a las muestras mal clasificadas. Penalizando más al modelo de regresión lineal, estos fueron los resultados:

5	Ridge	Lineal	0.001011	0.454591	0.544708
6	Ridge	Poly	0.002011	0.454135	0.544160

El R cuadrado para estas pruebas dieron muy por debajo del objetivo, por lo que descartamos esta posibilidad.

Por último, iteramos con el modelo SVR, que busca maximizar el margen , siendo la salida esperada del modelo un valor continuo. Determina el margen, con una función de costo y va intentar que todas las muestras caigan dentro de él. Los hiperparametros que tuvimos que elegir mediante CV son el costo y el Epsilon. Para este modelo, los resultados fueron:

7	SVR	Linear	0.087	0.415	0.473
8	SVR	Poly	0.031	0.441	0.469

Como resumen de todos los modelos probados:

	Model	Features	R2	MSE	MAE
0	Linear	Lineal	0.042	0.436	0.534
1	Linear	Poly grado 5	0.164	0.381	0.472
2	Linear	Poly grado 4	0.172	0.377	0.470
3	Linear	Poly grado 3	0.164	0.380	0.476
4	Linear	Poly grado 2	0.165	0.380	0.476
5	Ridge	Lineal	0.001	0.455	0.545
6	Ridge	Poly	0.002	0.454	0.544
7	SVR	Linear	0.087	0.415	0.473
8	SVR	Poly	0.031	0.441	0.469

VII. CONCLUSION FINAL

Al comienzo del trabajo, luego de haber realizado el análisis exploratorio de los datos, creímos que las features elegidas podrían explicarnos la duración de un recorrido de bicicleta mediante modelos de regresión. Tras intentar reiteradas veces encontrar un modelo con un error bajo y con una buena precisión, hemos concluido que las variables combinadas de la forma que lo hicimos no nos resuelven la problemática. Probamos con diferentes hiperparametros para poder mejorar el ajuste de los modelos pero no encontramos una razón por la cual estos no pudieron predecir la variable continua con un valor aceptable. El máximo R^2 logrado no superó el 20%, por lo que, dejamos sentadas las bases para retomar nuevamente este desafío con otras features y otras estrategias.

Referencias

[1] Drucker, H., Burges, C. J., Kaufman, L., Smola, A., & Vapnik, V. (1996). Support vector regression machines. Advances in neural information processing systems, 9, 155-161.

[2] Stirling, W. D. (1982). Enhancements to aid interpretation of probability plots. Journal of the Royal Statistical Society: Series D (The Statistician), 31(3), 211-220.

[3] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, 2825-2830.