Bike Availability Prediction

Predict number of bikes using Bicing real data

(Capstone Project)

Nil Ramos Díaz

User Kaggle: nilramos

1. Objetivos y Alcance del Estudio

El objetivo principal de este trabajo es desarrollar un modelo predictivo del porcentaje de docks disponibles en las estaciones de Bicing de Barcelona en cualquier momento dado. Además de mejorar la experiencia del usuario, también aspiramos a proponer la ubicación estratégica de nuevas estaciones de Bicing mediante el análisis de datos. Visualizar y analizar los datos históricos y en tiempo real nos permitirá identificar patrones y tendencias que podrían sugerir la necesidad de nuevas infraestructuras. Este proyecto combina técnicas avanzadas de ciencia de datos, como el aprendizaje automático y la visualización de datos, para abordar un problema complejo con implicaciones directas en la movilidad urbana y la calidad de vida de los ciudadanos.

2. Extracción y origen de los datos.

Los datos han sido obtenidos mediante este script, siendo la página web de origen Open Data de Ajuntament de Barcelona.

```
import os

i2m = list(zip(range(1,13),
['Gener','Febrer','Marc','Abril','Maig','Juny','Juliol','Agost','Setembre','Oc
tubre','Novembre','Desembre']))
for year in [2022, 2021, 2020, 2019]:
for month, month_name in i2m:
os.system(f"wget 'https://opendata-
ajuntament.barcelona.cat/resources/bcn/BicingBCN/{year}_{month:02d}_{month_nam
e}_BicingNou_ESTACIONS.7z'")
os.system(f"7z x '{year}_{month:02d}_{month_name}_BicingNou_ESTACIONS.7z'")
os.system(f"rm '{year}_{month:02d}_{month_name}_BicingNou_ESTACIONS.7z'")
```

En la extracción, obtuvimos 4 carpetas correspondientes a los años 2020, 2021, 2022 y 2023 con un fichero parquet de alrededor de 25MB, por lo que en total, trabajaremos con aproximadamente 4,5 GB’s de data.

3. Perímetro y selección de los datos.

Para este trabajo, el perímetro definido por eficiencia y sentido de los datos ha sido 2023, 2022 y 2021.

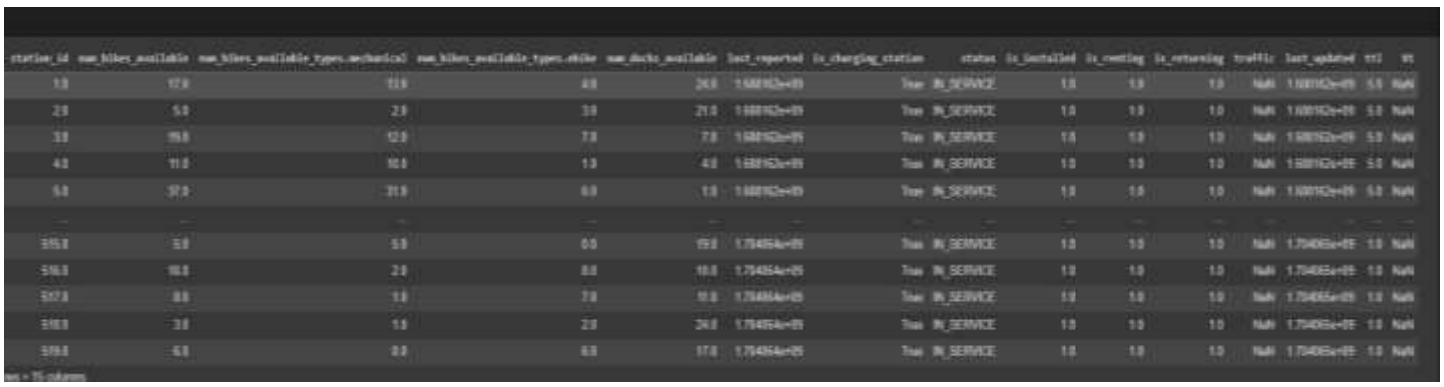
La primera decisión fue descartar 2020 ya que durante la pandemia el sistema de Bicing dejó de estar operativo en confinamiento y nos faltan bastantes datos para nuestro entrenamiento. Además, la segunda mitad del año 2020 fue atípica y no es representativa para entrenar y predecir en 2024.

Por otro lado, los lados son recibidos en secuencias de cada 5 minutos, con una serie de características e información asociada al número de bicicletas disponibles en la estación. Una de las decisiones iniciales será agrupar los datos por hora del día, día y mes de cada estación, teniendo así la media por hora como valor principal.

4. Pre procesamiento de los datos – primer paso.

Como bien se ha indicado en el punto anterior, para poder tratar y agrupar los datos correctamente y poder procesar tal cantidad de información sin tener problemas de rendimiento hemos seguido los siguientes pasos:

1. Este era el dataset original, lo primero ha sido eliminar esas columnas que no serán necesarias, tanto las que tienen una alta cardinalidad como aquellas que con una baja variabilidad.

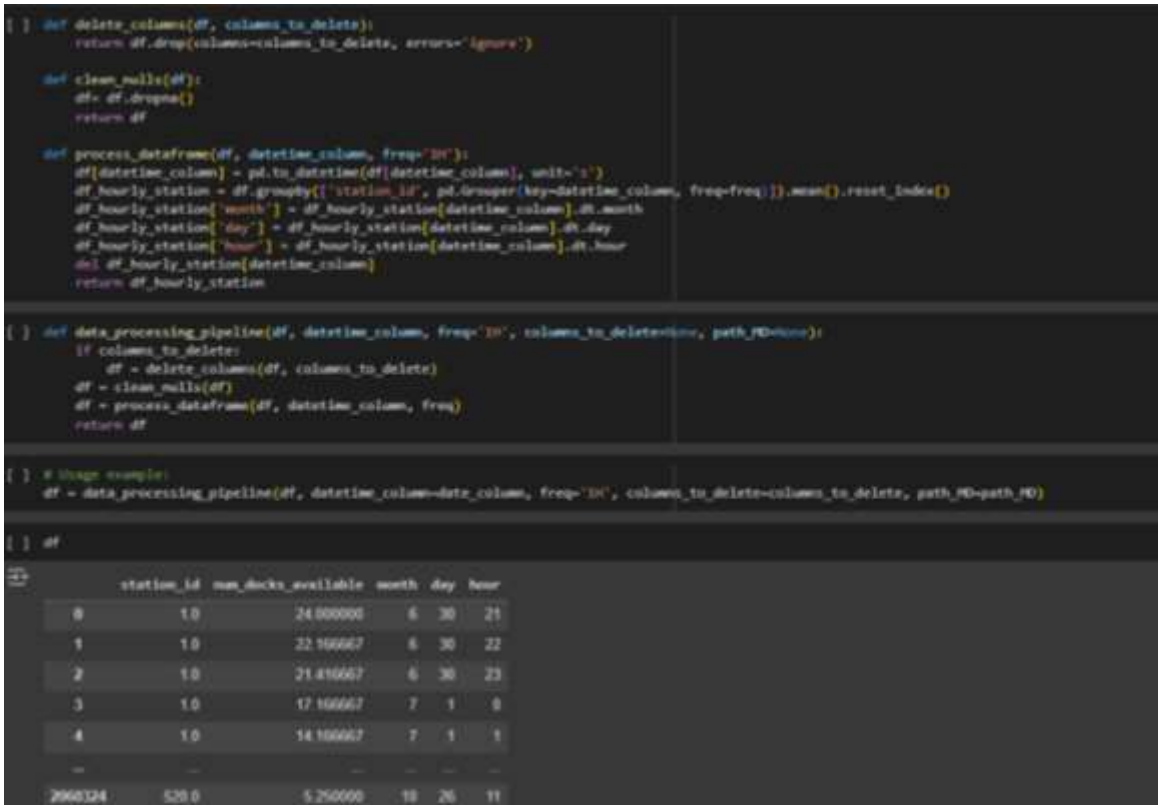


2. El siguiente paso fue eliminar todos los nulos existentes en la columna ‘num_docks_available’, para así evitar problemas en la agrupación por horas.

3. Pasamos la columna que utilizaremos como datetime a un formato correcto mediante *“pd.to_datetime(df[datetime_column], unit='s')”.*

4. Posteriormente, agrupamos por station id y por datetime, agrupando por bloques de una hora, teniendo así la media por hora.

Una vez realizados estos pasos, tal y como se enseña en la imagen, terminamos con el siguiente dataframe.



	station_id	num_docks_available	month	day	hour
0	1.0	24.000000	6	30	21
1	1.0	22.166667	6	30	22
2	1.0	21.416667	6	30	23
3	1.0	17.166667	7	1	0
4	1.0	14.166667	7	1	1
...	...	...	...	...	...
2968324	520.0	5.250000	10	26	11

Por último, le añadimos una columna que diga el año al que corresponde y lo guardamos en un parquet.

El objetivo de este pre procesamiento inicial es tener 2 parquets por año, correspondientes a 6 meses de datos de cada año. Por motivos de insuficiencia de memoria y de computación, es lo más eficiente para agrupar los datos.

5. Pre-procesamiento de los datos – segundo paso.

El objetivo de este último paso es simple – agrupar los parquets de 6 meses y concatenarlos para tener un contexto y unas columnas extras que enriqueceran nuestra data.

```
dfs = [df2021_1, df2021_2, df2022_1, df2022_2, df2023_1, df2023_2]
df = pd.concat(dfs, ignore_index=True)
```

En primer lugar, vamos a añadir las variables procedentes de la tabla de información, en esta tabla se encuentra la información de cada estación. Las columnas que traeremos a nuestro dataframe serán: 'capacity','lat','lon','altitude','address'. Lo que buscaremos será una union inner, para descartar esos stations id que no tengamos en la tabla de información.

En segundo lugar, creamos la variable “percentage_docks_available” que será la variable que usaremos para predecir en el futuro. Para crearla es tan sencillo como dividir el numero de docks avaiable entre la capacidad total de la estación, para así obtener el procentaje de docks libres que tendrá la estación en un determinado momento.

Como los datos de capacidad son de la última actualización de las estaciones, puede ser que alguna se haya visto reducida y por tanto en algunos momentos tenga mas huecos libres que capacidad tiene la estación, para ello hemos implementado la norma de que no puede haber mas de un 100% de huecos libres.

Por último, añadimos 4 variables de lag, añadiendo a los datos el porcentage de docks libres de las últimas 4 horas.

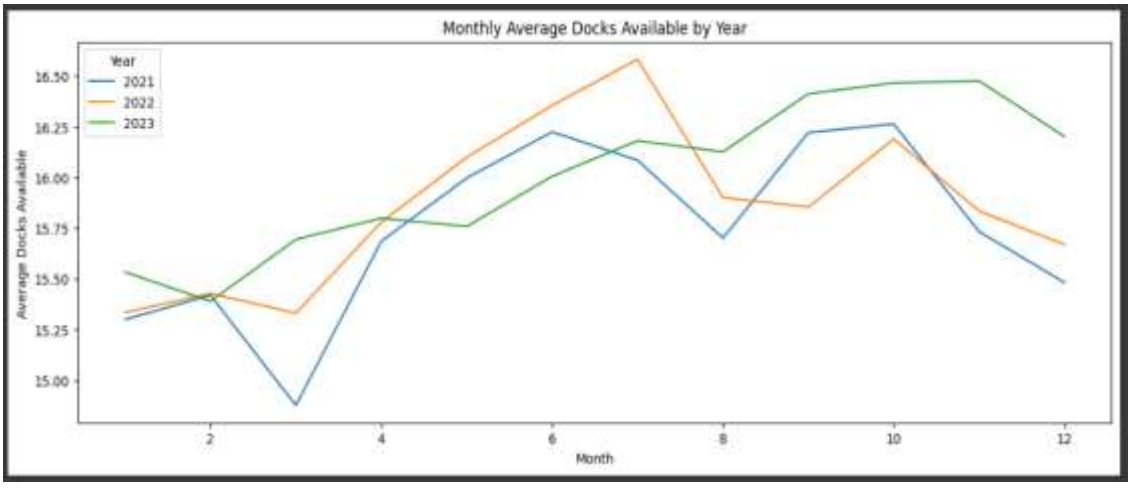
	station_id	num_docks_available	month	day	hour	year	capacity	lat	lon	altitude	address	percentage_docks_available	ctx_1	ctx_2	ctx_3	ctx_4
0	1	5.500000	1	1	0	2021	45	41.397978	2.180107	16.0	GRAN VIA CORTS CATALANES, 760	0.12	NaN	NaN	NaN	NaN
1	1	5.500000	1	1	1	2021	45	41.397978	2.180107	16.0	GRAN VIA CORTS CATALANES, 760	0.11	0.12	NaN	NaN	NaN
2	1	5.000000	1	1	2	2021	45	41.397978	2.180107	16.0	GRAN VIA CORTS CATALANES, 760	0.11	0.11	0.12	NaN	NaN
3	1	5.000000	1	1	3	2021	45	41.397978	2.180107	16.0	GRAN VIA CORTS CATALANES, 760	0.11	0.11	0.11	0.12	NaN
4	1	5.000000	1	1	4	2021	45	41.397978	2.180107	16.0	GRAN VIA CORTS CATALANES, 760	0.11	0.11	0.11	0.11	0.12
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
12073737	519	18.416667	12	31	19	2023	24	41.424655	2.166289	110.0	C/ PEDRELL, 52	0.77	0.87	0.85	0.83	0.89
12073738	519	17.166667	12	31	20	2023	24	41.424655	2.166289	110.0	C/ PEDRELL, 52	0.72	0.77	0.87	0.85	0.83
12073739	519	16.416667	12	31	21	2023	24	41.424655	2.166289	110.0	C/ PEDRELL, 52	0.68	0.72	0.77	0.87	0.85
12073740	519	16.000000	12	31	22	2023	24	41.424655	2.166289	110.0	C/ PEDRELL, 52	0.67	0.68	0.72	0.77	0.87
12073741	519	16.333333	12	31	23	2023	24	41.424655	2.166289	110.0	C/ PEDRELL, 52	0.68	0.67	0.68	0.72	0.77

12085794 rows × 16 columns

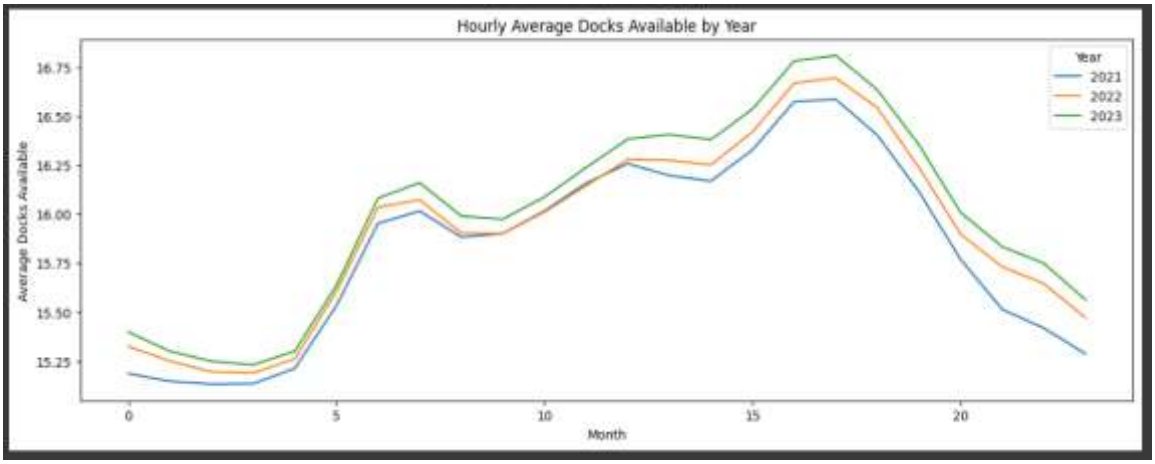
6. Exploratory Data Analysis (EDA)

Tras agrupar correctamente todos los datos que utilizaremos para predecir el porcentaje de docks disponibles, vamos a realizar un análisis exploratorio para obtener el máximo conocimiento posible de los datos que tenemos.

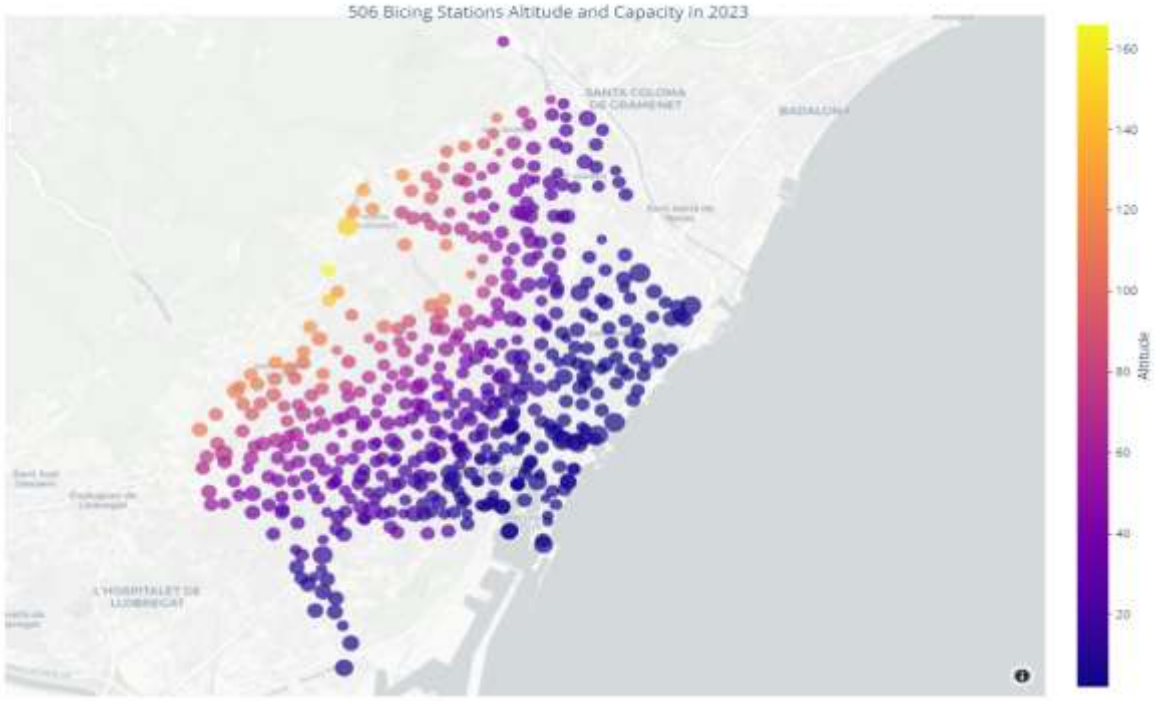
Anualmente se ve una misma tendencia en el aumento de disponibilidad de los meses de verano y una disminución en otoño y en los meses de navidades.



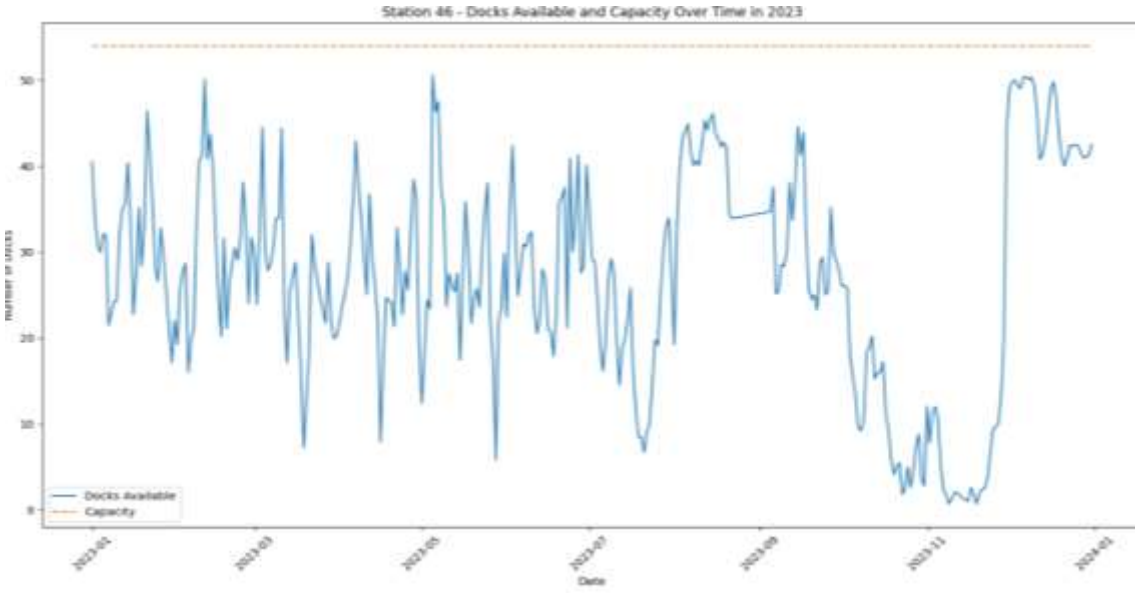
En cuanto a las horas, se ve como baja la disponibilidad después de fuera del horario laboral y como de madrugada, gracias a la redistribución que hace la empresa, muchas se recargan y otras se vacían.



También es interesante analizar la distribución de las estaciones en la ciudad, en el eje Y se ve como se distribuye el color dependiendo de la altitud de la misma, cuanto más cerca de la montaña del Tibidabo, más crece la inclinación.



Otro tipo de gráfico fue entrar directamente a la maxima granularidad posible viendo como fluctuaba la capacidad de una estación durante el tiempo. Pudiendose ver en este ejemplo como esta estación fue claramente cerrada y ampliada a finales de 2023:



7. *Feature Engineering – enriquecimiento del dataset*

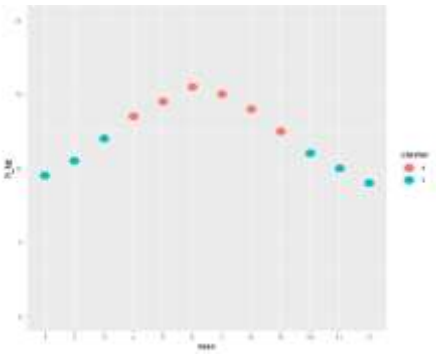
Una vez hecha una primera exploración de los datos, vamos a pasar a enriquecerlos y a prepararlos para el entrenamiento del modelo.

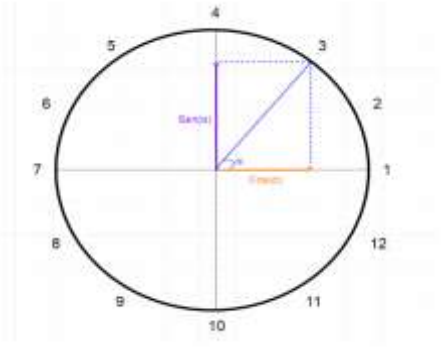
Las principales modificaciones que haremos serán

1. Variables cíclicas
2. Temperaturas
3. Precipitaciones

En cuanto a las variables cíclicas, a partir de la hora del día y del mes se han creado 4 nuevas columnas con el seno y el coseno de las mismas. La idea detrás de esto es que el modelo entienda que la diferencia entre el mes 12 y el 1, es de 1, y no de 11 meses de distancia, y lo mismo con la hora. De esta manera, interpreta con mayor facilidad y predice favorablemente los meses de los extremos.

Aquí unos ejemplos de como actúa el uso de variables ciclícas[1]:



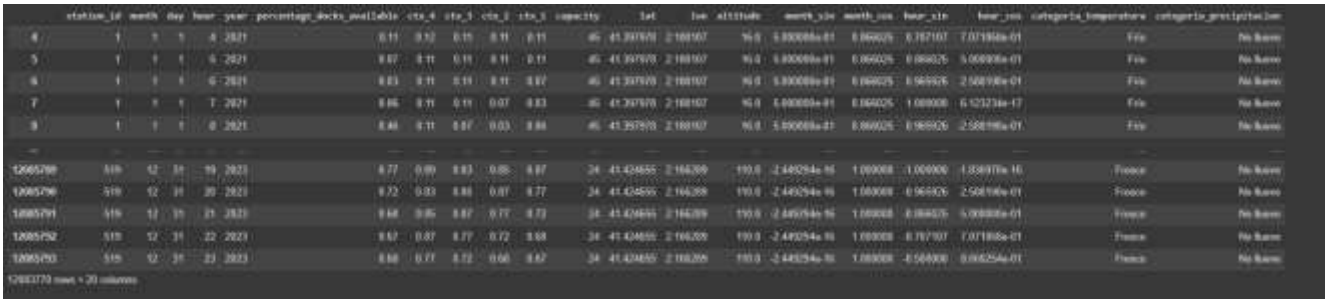


Por otro lado, en cuanto a precipitaciones y temperaturas, se han utilizado los datos libres de uso que proporciona la AEMET[2], obteniendo por día, la temperatura media y la precipitación en milímetros.

Para la temperatura, en lugar de mantener los datos como grados, la convertiremos en una columna categórica con los siguientes rangos: frío, fresco, templado, cálido y caluroso. Sin embargo, para la temperatura solo habrán dos opciones: llueve o no llueve.

[1] https://aukera.es/blog/variables-ciclicas/ fecha de consulta: 08/06/2024
[2] https://www.aemet.es/es/datos_abiertos fecha de consulta: 10/06/2024

Este es el dataframe que nos llevaremos a la última parte: el entrenamiento del modelo.



8. *Modelo e interpretabilidad*

Cargamos el parquet post-feature engineering y dividimos el dataset en train y test: entrenaremos con los datos que pertenezcan a 2022 y 2023, de este último, solo los meses de enero a julio. Dejando el resto de 2023 para validar.

Tras este split, nos quedamos con 6,4M de filas para entrenar y 1,6M para validar, aproximadamente el 26%.

En cuanto a los datos, preparemos un one hot encoding para la columna categórica de la temperatura, y para la de precipitación al ser booleana marcaremos el “No llueve” como 0 y el “Llueve” como 1, el resto, al ser todo númerico simplemente esperaremos al siguiente paso, que es escalarlo con el StandardScaler de Scikit-learn.

El modelo utilizado y el que mejor ha funcionado es el siguiente:

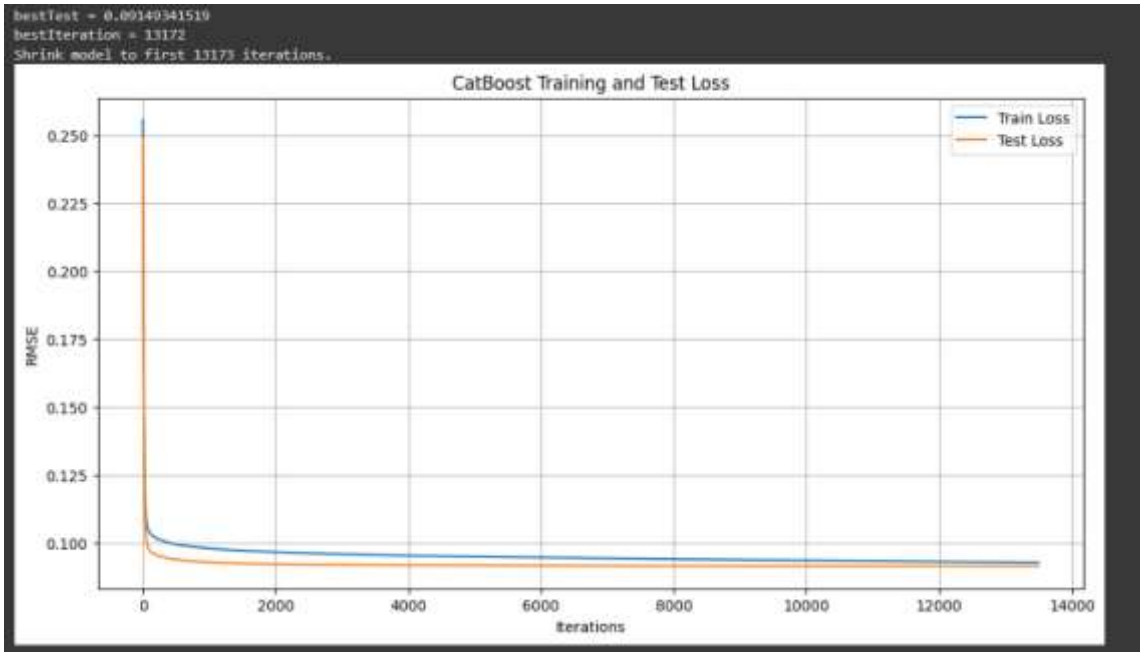
```
print("\nTraining CatBoost...")
catboost_reg = CatBoostRegressor(
    iterations=13500,
    depth=8,
    learning_rate=0.05,
    boosting_type='Plain',
    task_type='GPU',
    devices='0:1',
    verbose=400,
    random_strength=12,
    l2_leaf_reg=8,
    bagging_temperature=1.5
)
catboost_reg.fit(X_train_scaled, y_train)
train_score_catboost = catboost_reg.score(X_train_scaled, y_train)
test_score_catboost = catboost_reg.score(X_test_scaled, y_test)
y_pred_train_catboost = catboost_reg.predict(X_train_scaled)
y_pred_test_catboost = catboost_reg.predict(X_test_scaled)
rmse_train_catboost = calculate_rmse(y_train, y_pred_train_catboost)
rmse_test_catboost = calculate_rmse(y_test, y_pred_test_catboost)
print("CatBoost: Train Score =", train_score_catboost, ", Test Score =", test_score_catboost)
print("CatBoost: RMSE Train =", rmse_train_catboost, ", RMSE Test =", rmse_test_catboost)
```

Y los resultados obtenidos fueron:

```
10400:  learn: 0.0934643        total: 11m 18s  remaining: 3m 22s
10800:  learn: 0.0933755        total: 11m 44s  remaining: 2m 56s
11200:  learn: 0.0932878        total: 12m 9s   remaining: 2m 29s
11600:  learn: 0.0932025        total: 12m 35s  remaining: 2m 3s
12000:  learn: 0.0931232        total: 13m 1s   remaining: 1m 37s
12400:  learn: 0.0930383        total: 13m 27s  remaining: 1m 11s
12800:  learn: 0.0929595        total: 13m 52s  remaining: 45.5s
13200:  learn: 0.0928829        total: 14m 19s  remaining: 19.5s
13499:  learn: 0.0928264        total: 14m 37s  remaining: 0us
CatBoost: Train Score = 0.8781434959507974 , Test Score = 0.8751013535088383
CatBoost: RMSE Train = 0.09282637117561816 , RMSE Test = 0.09150850430720885
```

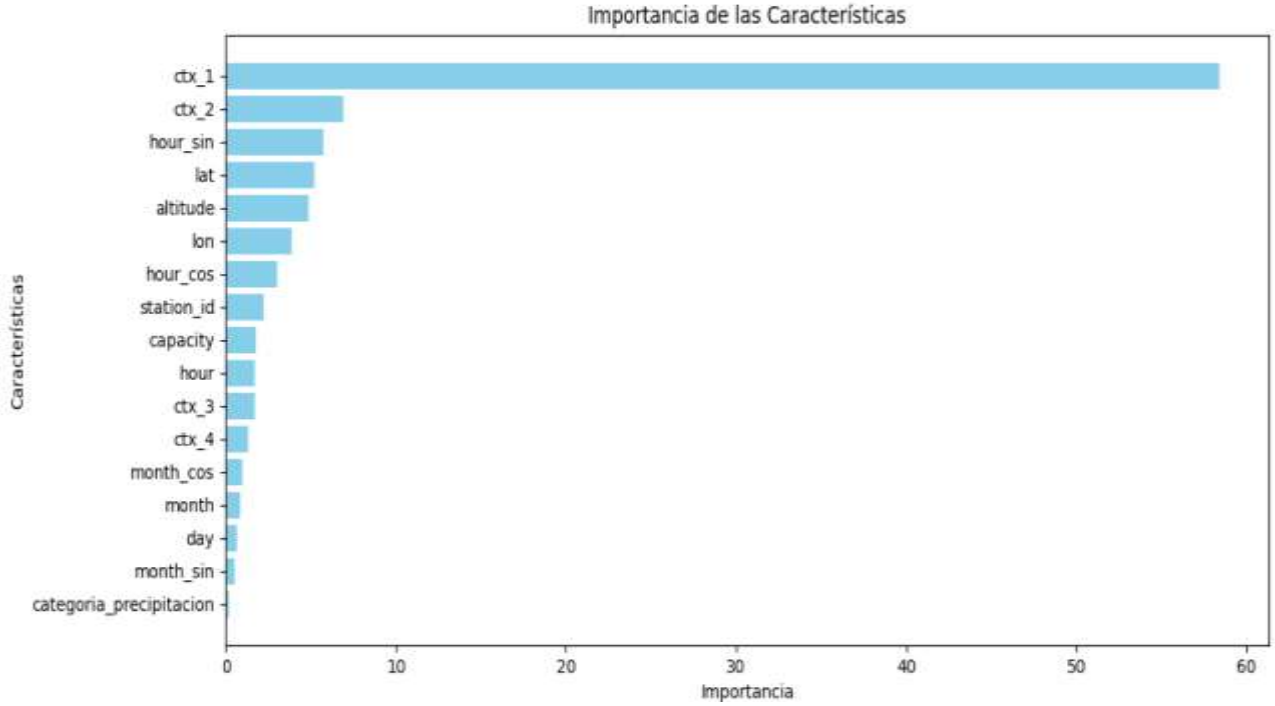
Con un RMSE de 0.09 en el conjunto de entrenamiento, podemos considerar que el resultado obtenido es adecuado. Dado que la capacidad media de las estaciones es de aproximadamente 35 docks, esto implica un error medio de alrededor de 3-4 docks. Este margen de error se encuentra dentro de la tolerancia asumida y es aceptable para nuestros propósitos.

Este es el resumen del RMSE tanto en train como en test. Como se puede observar la mejor iteración fue la 13172 y el mejor RMSE de 0.091



En cuanto a la interpretabilidad del modelo y la importancia de las características, se ve claramente como gran parte de predicción fue tomando en cuenta los valores ctx_1 y ctx_2, también fueron muy importantes la coordenadas, y tiene sentido, ya que realmente más que una predicción conjunta lo que se intentaba es que el modelo entendiese que cada estación era distinta y se comportaba de manera diferente.

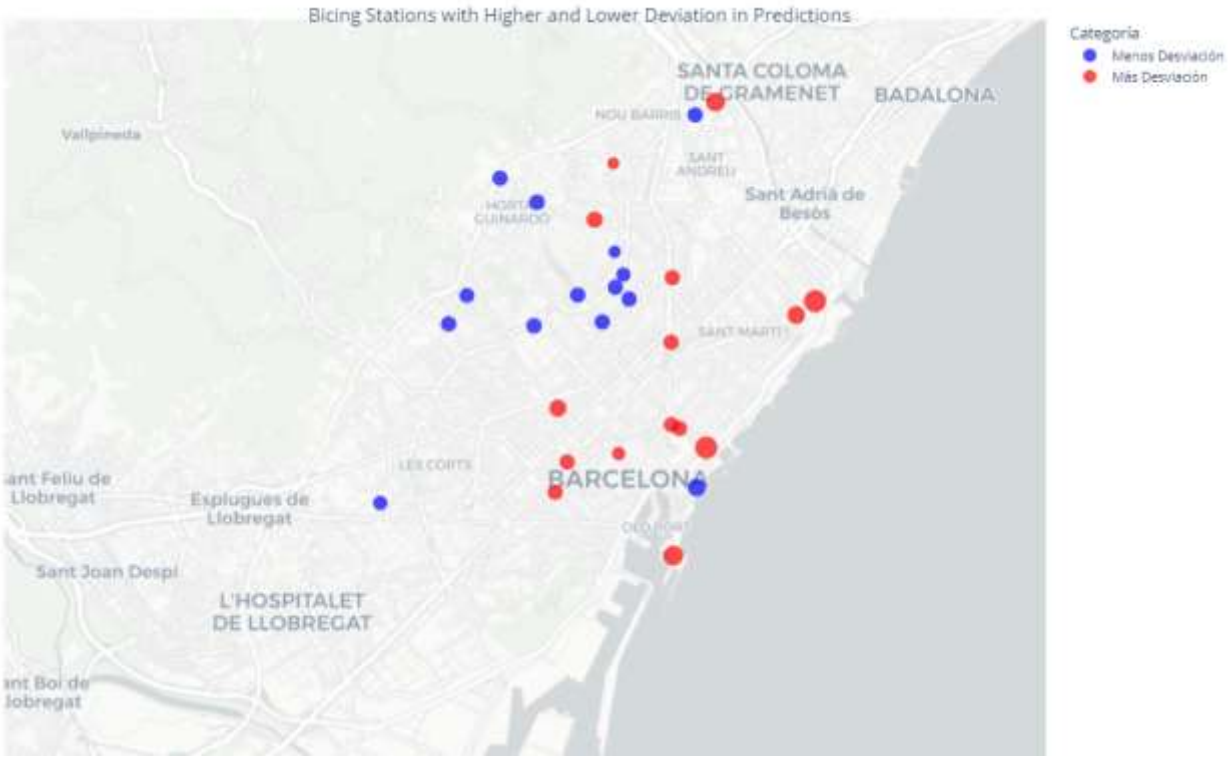
Sorprende también que las variables cíclicas para la hora funcionaron mejor que las estándar, y contrariamente, parece ser que la precipitación no influyó en el modelo. Una de las causas puede que sea el hecho de que solo se consiguieron datos diarios, imputando para todas las horas lo que ocurrió durante todo el día.



9. Visualizaciones sobre las predicciones

Una vez predecidos los resultados, es interesante observar como se distribuyen las estaciones que mejor hemos predecido, y las que peor.

En este mapa se puede observar en color rojo las peores estaciones en cuanto ha resultados obtenidos en la predicción, y en azul, las que mejor resultado han obtenido (algunas muy por debajo del RMSE general).



Como se puede observar, la tendencia es que las estaciones ubicadas en el centro de la ciudad tienen más tendencia a predecir peor que las ubicadas en los alrededores. La causa podría ser originada porque en el centro de la ciudad, el tráfico de llegada y salida de las bicicletas no sigue ninguna tendencia ya que es la zona más turísticas y se rige por la impredecibilidad derivada de ello. Sin embargo, en localizaciones más alejadas, es donde los usuarios de bicing habituales utilizan éstas para ir al centro a acortar su camino y generalmente siguen una rutina o utilizan las bicicletas con una frecuencia que se puede medir.

10. Caso de estudio: Nuevas estaciones para absorber la demanda.

Con los datos utilizados durante este trabajo, vamos a intentar generar 5 ubicaciones donde sería óptimo (teniendo en cuenta que no sabemos la disponibilidad que tiene el sitio) implantar estaciones de Bicing. Para ello vamos a estudiar ciertas características que tendría que tener la ubicación "óptima" para poder albergar una estación:

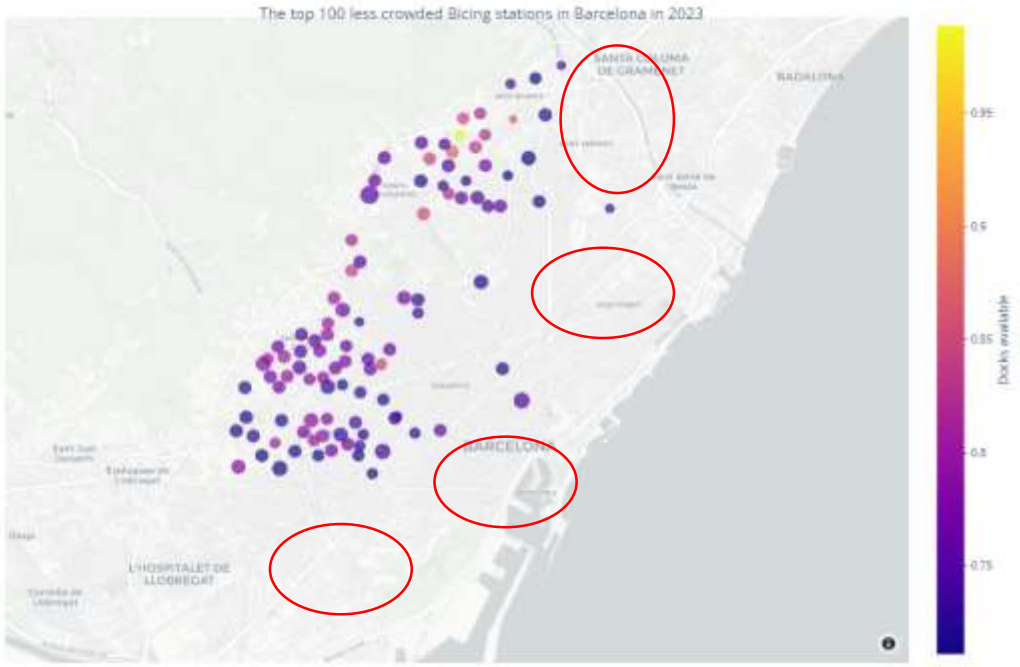
1. Estar en una zona con una alta densidad de población.
2. Cerca de estaciones de transporte público para facilitar la conexión.
3. Con carriles bicis cercanos.
4. Ubicaciones donde se prevea que habra una alta demanda en el futuro.

Para encontrar esas ubicaciones, vamos a estudiar primeramente cuales son las estaciones con un mayor aumento de la demanda:

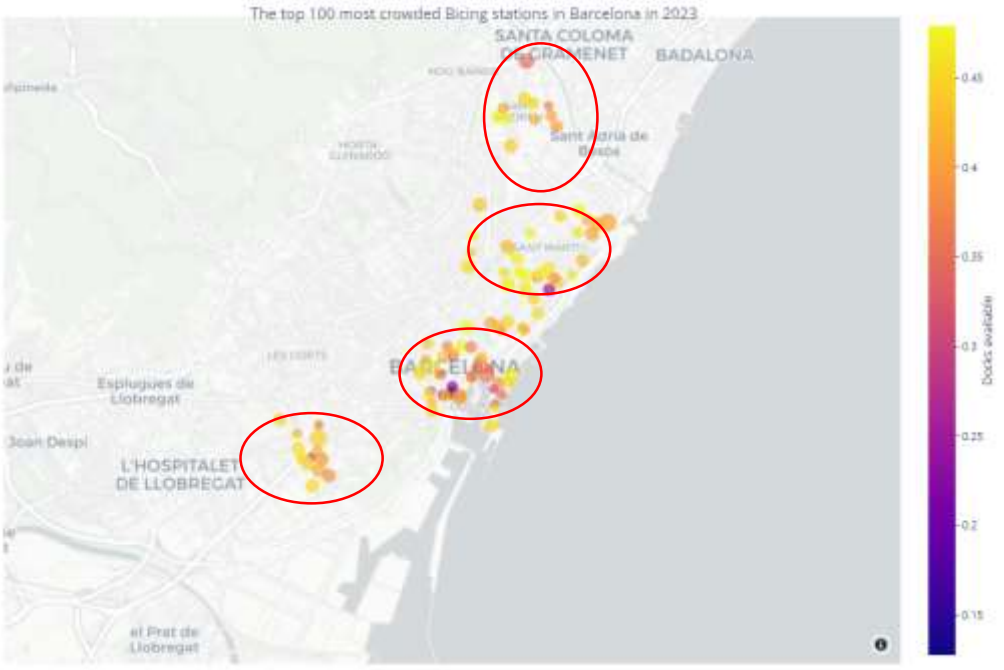
Se puede observar como las zonas con mas demanda se pueden dividir en tres: la zona cercana a l'Hospitalet, Barcelona centro histórico y la zona de Sant Martí / Sant Adrià del Besòs. Esta información es útil para ver las localizaciones donde más está creciendo el uso de las bicicletas.

El siguiente paso que tomaremos es ver cuales son las cien estaciones menos utilizadas (con mayor porcentaje de bicis disponibles de media) para ver si concuerda con que ninguna está cerca de las ubicaciones con mayor crecimiento, y parece que es así.

En circulos rojos, los lugares donde más crecimiento había:

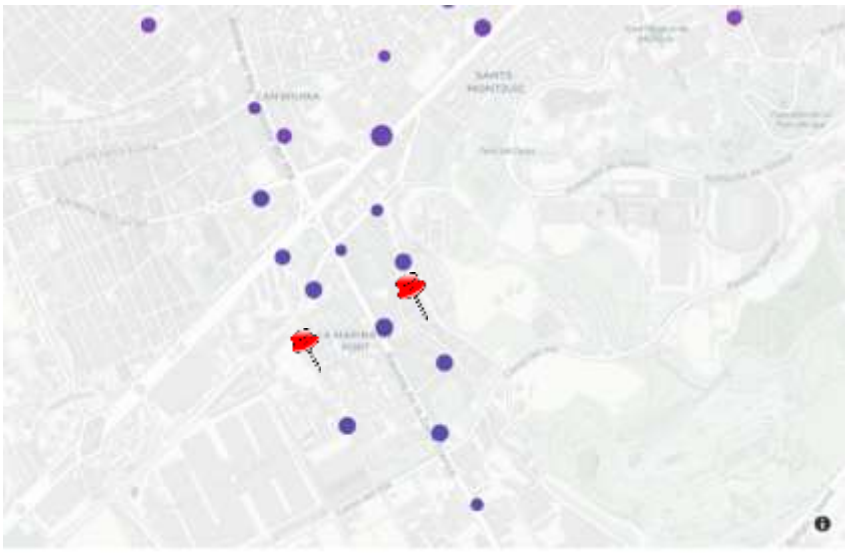


Haciendo el ejercicio contrario y analizando las cien estaciones con menos bicis disponibles de media, encontraremos que la gran mayoría de estaciones casi al tope de su capacidad, se encuentran justamente en esas áreas.



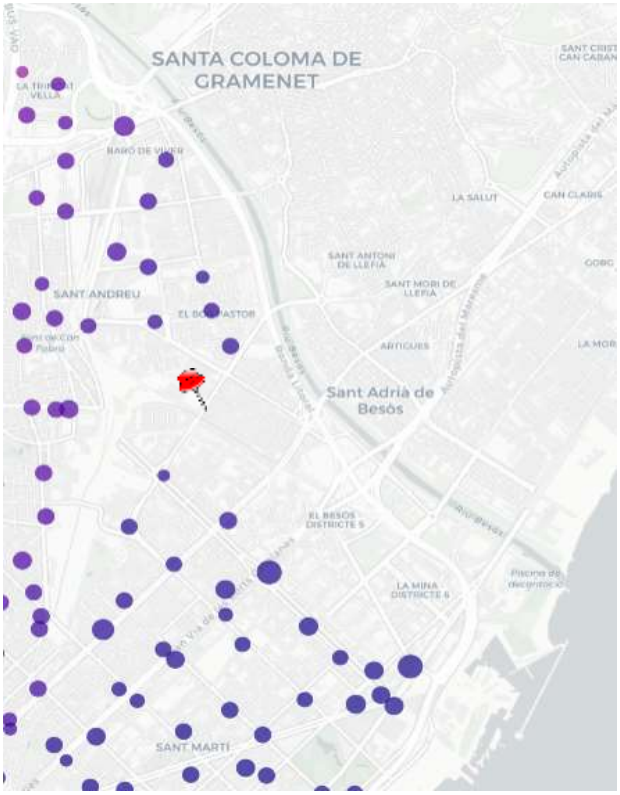
Una vez encontradas las ubicaciones donde sería perfecto colocar las estaciones nuevas, vamos a ponerlas en el mapa:

Ubicaciones 1 y 2:



Ubicación 3 y 4:

Ubicación 5:



En definitiva, la red de bicing está en una fase ya muy madura y su funcionamiento es muy correcto. Pese a existir un top 100 de estaciones más saturadas, es importante incidir en que solo 5 de ellas están por encima del 80% de saturación.

También es importante comentar que existe un trabajo en la sombra de los operarios de bicing, encargados de restribuir las bicicletas a las estaciones que se quedan más vacías a la noche. Esto produce que se creen fluctuaciones muy grandes en determinadas horas de la noche, que afectan negativamente a la posibilidad de predecir la disponibilidad.