

INFORME

PROYECTO

Alumnos: Casas Uriel Maximiano y Fustet Arnaldo Antonio

Materia: Aproximación al Campo Laboral

Año Lectivo: 2025

Año de Carrera: 1er Año

Profesor: Simón Polizzi

Índice

Fundamentación Problema/s específico/s a resolver	3
Objetivos.....	4
Objetivo General	4
Objetivos Específicos	4
Fuentes de Datos y Datasets.....	5
Visualización de Datasets.....	7
Algoritmos y Justificación	10
Resultados Esperados.....	12
Conclusión y Proyección	13
Proyección.....	14
Glosario.....	15
Referencias y Fuentes.....	17
Software utilizado:	17

Fundamentación Problema/s específico/s a resolver

Con datos de distintas líneas de colectivos en todo Buenos Aires, queremos revisar/observar la cantidad de colectivos que hay disponibles por parada, es decir, por cada parada pueden subir más o menos pasajeros (teniendo en cuenta que puede variar la cantidad por el día de la semana, algún evento y/o durante los feriados), y si suben pocos pasajeros de una línea determinada, podríamos optimizar la cantidad de colectivos de determinada línea distribuidos por la provincia reduciendo su cantidad de medio de transporte, y priorizando así, aumentar la cantidad de unidades permitiendo transportar más pasajeros o distribuirlos de una manera más optimizada de así poder cubrir las demandas.

Un caso real es la línea 410 de destino LUJÁN – MORENO que pasa por la RUTA 7. El viaje desde una ciudad hasta la otra dura aproximadamente 50 minutos, pero por quejas de pasajeros pueden estar mal distribuidas. En la página “Moovit” está informado que siempre aparecerá un colectivo cada 3 y 20 minutos, pero actualmente, eso no se cumple, en realidad es cada 40 minutos, y con eso, se hace notar por la cantidad de personas esperando en las paradas, y eso genera problemas también en el viaje de los pasajeros, debido a que los pasajeros del primer y segundo colectivo van llenos, y el tercero no sirve prácticamente en ese momento y hasta incluso los choferes no se detienen en las paradas a menos que uno que se encuentre dentro toque el timbre para detenerse. Una mejor distribución de horarios sería eficaz para aumentar la calidad del servicio de la línea 410 para los pasajeros.

Objetivos

Objetivo General

El objetivo del presente informe es analizar la distribución del transporte público, en esta ocasión, los transportes de las distintas líneas de colectivos en todo Buenos Aires. Para esto, se analiza la cantidad de pasajeros que son llevados en cada línea por día, y con esto, se busca mejorar la calidad del servicio del transporte.

Objetivos Específicos

- Optimizar Horarios (En la distribución de colectivos por cada línea, es decir, generalmente los transportes no llegan al horario acordado, o llegan las unidades muy juntas, lo que genera mayor tiempo de espera para los pasajeros que no abordaron a los transportes, y deben esperar más tiempo hasta el siguiente colectivo).
- Optimizar cantidad de colectivos de determinada línea (en base a muchos pasajeros en el transporte por día, aumentar la cantidad de colectivos de esa línea, o caso contrario, disminuir la cantidad de colectivos en determinada línea y priorizarla en otra que la necesite).
- Reducir tiempos de espera para los pasajeros (Con una mejor distribución de colectivos, habrá una mayor calidad de servicio para el cliente será mejor).
- Evitar sobrecarga de pasajeros (Habrá ocasiones donde ciertas líneas suban muchos pasajeros por día, dependiendo la parada también, entonces, al distribuir mejor por horario, o aumentar la cantidad de horarios se espera mejorar la calidad de servicio de transporte).

Fuentes de Datos y Datasets

Se utilizó datos Open Source de Buenos Aires por no contar con información de transporte más cercanas a la localidad de Luján, esto quiere decir, que utilizamos un **dataset** con información más precisa y cercana a la zona provincia, la que además cuenta con gran cantidad de registros para ser utilizada en este proyecto. El dataset fue extraído de una página de Argentina con grandes cantidades de datos de transportes, llamado: “[Datos abiertos de la Secretaría de Transporte](#)”. Dentro contiene datos desde el 2020 de enero, hasta el 2025 de octubre de líneas de colectivo de todo Buenos Aires.

Los datos se analizaron y procesaron utilizando diferentes scripts de python. Se normalizaron fechas y se cambió el **separador** de campo por “,”.

En cuanto al contenido, se cambió el **etiquetado de las columnas** y se realizó **poda** de datos irrelevantes, por ejemplo hubo un show musical, donde los viajes en transporte dispararon en cantidad de pasajeros en un día determinado, eso afectó en mala orientación a la media, lo cual hubo que eliminar el registro, otro caso que afectaron los datos, fueron las cantidades de pasajeros negativas, como los registros no tienen lógica alguna, tuvieron que ser descartadas. Estos registros que distorsionan el análisis por sus valores extremos son llamados **outliers**. Se cruzó información con otras fuentes para agregar información referente a periodos lectivos, estación del año y días hábiles.

Con esto nos quedaron los siguientes **atributos (o features)**:

DiaSemana → Guarda el día de la semana asociado a un número, por ejemplo, el 0 es Lunes, mientras que el 6 es el Domingo, el primer registro guardado es el día 1, es decir, Martes.

Feriado → También considerado de tipo lógico, contiene los datos False y True que determina si en el registro correspondiente era o no feriado, sin importar qué evento era.

Estacion → Época o temporada del año, puede ser VERANO, PRIMAVERA, OTOÑO o por consiguiente, INVIERNO.

Clases → También considerado de tipo lógico, contiene los datos False y True que determina si en el registro correspondiente hubo o no clases.

Pandemia → También considerado de tipo lógico, contiene los datos False y True que determina si en el registro correspondiente hubo o no cuarentena.

Estos atributos son mixtos, mientras que DiaSemana; Feriado; Clases; y Pandemia son de tipo numérico (o lógico Feriado, Clases y Pandemia), el restante (Estacion) es de tipo categóricos.

El **target (o label)** asociado a los atributos ya mencionados es el encabezado “Cantidad”, el cual representa la cantidad de pasajeros que abordaron el medio de transporte correspondiente en un día determinado, dada las condiciones por los atributos.

Ahora con el separador, podremos visualizar mejor los registros, debido a que serán separados las columnas para poder visualizar mejor los datos y decidir cuáles serán los features y el target:

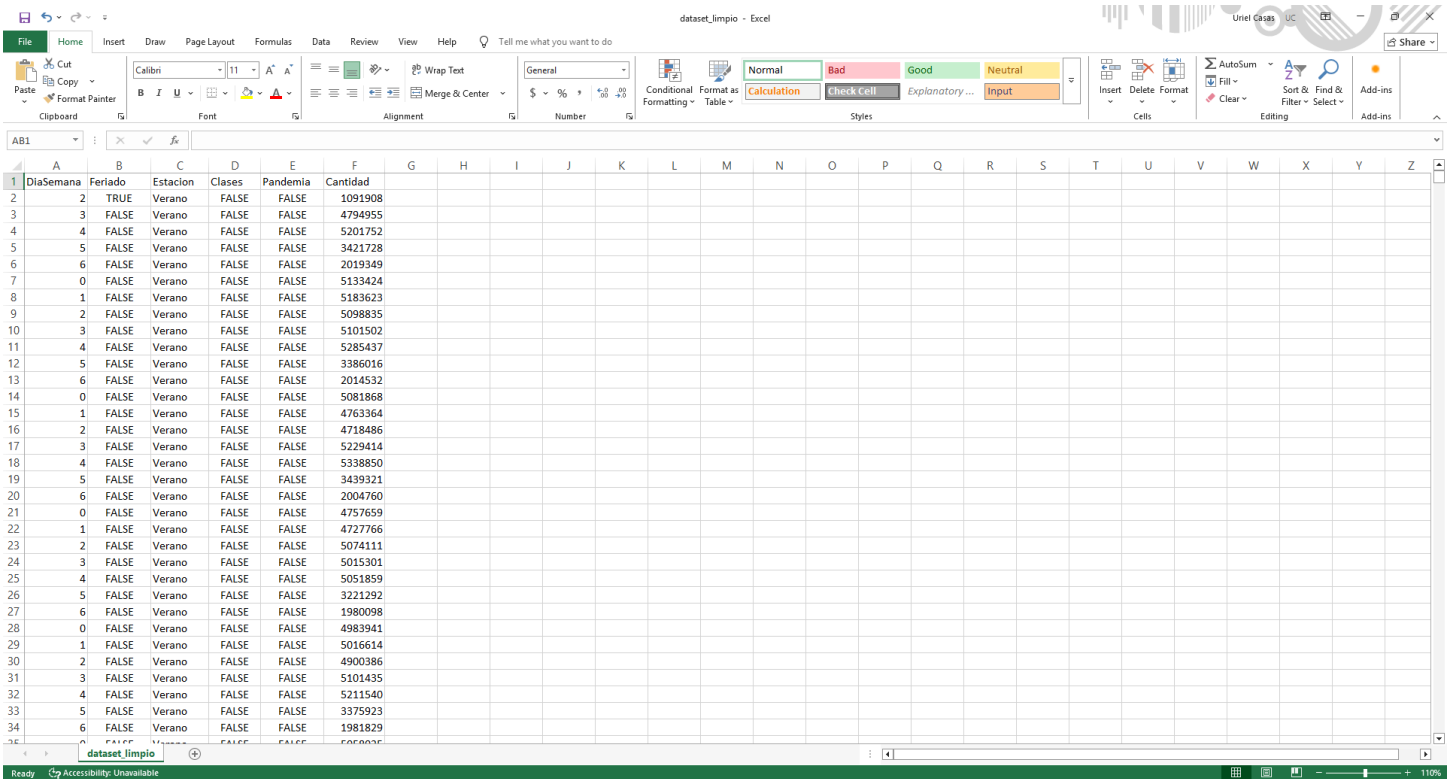
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	DIA_TRANSPORTE	AMBA	MOTIVO_ATSF	GENERO	TIPO_TRANSPORTE	CANT_TRJ	DATO_PRELIMINAR																		
2	1/1/2020	NO			COLECTIVO	48306	NO																		
3	1/1/2020	NO			LANCHA	0	NO																		
4	1/1/2020	NO			SUBTE	0	NO																		
5	1/1/2020	NO			TOTAL	48306	NO																		
6	1/1/2020	NO			TREN	0	NO																		
7	1/1/2020	NO		F	COLECTIVO	20057	NO																		
8	1/1/2020	NO		F	LANCHA	9	NO																		
9	1/1/2020	NO		F	SUBTE	0	NO																		
10	1/1/2020	NO		F	TOTAL	20066	NO																		
11	1/1/2020	NO		F	TREN	0	NO																		
12	1/1/2020	NO		M	COLECTIVO	16260	NO																		
13	1/1/2020	NO		M	LANCHA	14	NO																		
14	1/1/2020	NO		M	SUBTE	0	NO																		
15	1/1/2020	NO		M	TOTAL	16274	NO																		
16	1/1/2020	NO		M	TREN	0	NO																		
17	1/1/2020	NO	AUH	F	COLECTIVO	11000	NO																		
18	1/1/2020	NO	AUH	F	LANCHA	2	NO																		
19	1/1/2020	NO	AUH	F	SUBTE	0	NO																		
20	1/1/2020	NO	AUH	F	TOTAL	11002	NO																		
21	1/1/2020	NO	AUH	F	TREN	0	NO																		
22	1/1/2020	NO	AUH	M	COLECTIVO	233	NO																		
23	1/1/2020	NO	AUH	M	LANCHA	0	NO																		
24	1/1/2020	NO	AUH	M	SUBTE	0	NO																		
25	1/1/2020	NO	AUH	M	TOTAL	233	NO																		
26	1/1/2020	NO	AUH	M	TREN	0	NO																		
27	1/1/2020	NO	JUBILACION	F	COLECTIVO	11129	NO																		
28	1/1/2020	NO	JUBILACION	F	LANCHA	8	NO																		
29	1/1/2020	NO	JUBILACION	F	SUBTE	0	NO																		
30	1/1/2020	NO	JUBILACION	F	TOTAL	11137	NO																		
31	1/1/2020	NO	JUBILACION	F	TREN	0	NO																		
32	1/1/2020	NO	JUBILACION	M	COLECTIVO	5847	NO																		
33	1/1/2020	NO	JUBILACION	M	LANCHA	6	NO																		
34	1/1/2020	NO	JUBILACION	M	SUBTE	0	NO																		
35	1/1/2020	NO	JUBILACION	M	TOTAL	5853	NO																		
36	1/1/2020	NO	JUBILACION	M	TREN	0	NO																		
37	1/1/2020	NO	MONOTRIBUTO SOC F		COLECTIVO	2494	NO																		
38	1/1/2020	NO	MONOTRIBUTO SOC F		LANCHA	0	NO																		

Luego de decidir cuáles serían los features y el target, se eliminó el separador que se encontraba dentro del código, y dentro del programa se limpió los datos de la siguiente manera:

1. La etiqueta “DIA_TRANSPORTE” fue cambiado a “DiaSemana” y en vez de contener la fecha con el formato “%Y/%m/%d”, cambio a uno que contiene solo el día de la semana en un codificado entre el 0 y el 6 (0 = lunes, 6 domingo).
2. La columna “AMBA”, “MOTIVO_ATSF”, “GENERO”, “TIPO_TRANSPORTE” y “DATO_PRELIMINAR” fueron podados porque no eran relevantes para la investigación.

3. Se agregaron cuatro columnas al conjunto de datos para poder analizar mejor el target en base a los features: “Feriado”, “Estacion”, “Clases”, y “Pandemia”.
4. La etiqueta “CANT_TRJ” fue cambiado a “Cantidad”, fue colocado en la última columna.
5. Los registros fueron totalizados en base a “DiaSemana” para evitar la redundancia en el conjunto de datos.

El dataset limpio se podrá observar de la siguiente manera (se muestra el dataset con un separador insertado en el código):



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	DiaSemana	Feriado	Estacion	Clases	Pandemia	Cantidad																				
2	2	TRUE	Verano	FALSE	FALSE	1091908																				
3	3	FALSE	Verano	FALSE	FALSE	4794955																				
4	4	FALSE	Verano	FALSE	FALSE	5201752																				
5	5	FALSE	Verano	FALSE	FALSE	3421728																				
6	6	FALSE	Verano	FALSE	FALSE	2019349																				
7	0	FALSE	Verano	FALSE	FALSE	5133424																				
8	1	FALSE	Verano	FALSE	FALSE	5183623																				
9	2	FALSE	Verano	FALSE	FALSE	5098835																				
10	3	FALSE	Verano	FALSE	FALSE	5101502																				
11	4	FALSE	Verano	FALSE	FALSE	5285437																				
12	5	FALSE	Verano	FALSE	FALSE	3386016																				
13	6	FALSE	Verano	FALSE	FALSE	2014532																				
14	0	FALSE	Verano	FALSE	FALSE	5081868																				
15	1	FALSE	Verano	FALSE	FALSE	4763364																				
16	2	FALSE	Verano	FALSE	FALSE	4718486																				
17	3	FALSE	Verano	FALSE	FALSE	5229414																				
18	4	FALSE	Verano	FALSE	FALSE	5338850																				
19	5	FALSE	Verano	FALSE	FALSE	3439321																				
20	6	FALSE	Verano	FALSE	FALSE	2004760																				
21	0	FALSE	Verano	FALSE	FALSE	4757659																				
22	1	FALSE	Verano	FALSE	FALSE	4727766																				
23	2	FALSE	Verano	FALSE	FALSE	5074111																				
24	3	FALSE	Verano	FALSE	FALSE	5015301																				
25	4	FALSE	Verano	FALSE	FALSE	5051859																				
26	5	FALSE	Verano	FALSE	FALSE	3221292																				
27	6	FALSE	Verano	FALSE	FALSE	1980098																				
28	0	FALSE	Verano	FALSE	FALSE	4983941																				
29	1	FALSE	Verano	FALSE	FALSE	5016614																				
30	2	FALSE	Verano	FALSE	FALSE	4900386																				
31	3	FALSE	Verano	FALSE	FALSE	5101435																				
32	4	FALSE	Verano	FALSE	FALSE	5211540																				
33	5	FALSE	Verano	FALSE	FALSE	3375923																				
34	6	FALSE	Verano	FALSE	FALSE	1981829																				
35	0	FALSE	Verano	FALSE	FALSE	5050026																				

Algoritmos y Justificación

Para el desarrollo se seleccionó el **Algoritmo de Regresión Logística**, perteneciente al **aprendizaje supervisado**. Este algoritmo es un modelo estadístico supervisado, utilizado para predecir la **probabilidad** de **ocurrencia** de un **evento binario** o **categorico** en función de varias variables independientes, que pueden ser continuas o categóricas. Esto quiere decir, el algoritmo utilizado obtuvo **entradas(x)** y **salidas conocidas(y)**, cuando el modelo aprendió de la relación entre X e Y, será capaz de predecir **Y** cuando solo tenga **X** usando la **función sigmoide**.

Con esto podemos estimar la **probabilidad** que habrá de alta o baja demanda, y en base a esto, optimizar las decisiones (por ejemplo, aumentar o disminuir la cantidad en un día o temporada determinado).

El modelo lineal estimara la probabilidad de demanda en base a los **features** (Feriado, DiaSemana, etc), y el **tarjet** (Cantidad). Esto permite anticipar comportamientos en base al análisis de los datos, reduciendo decisiones intuitivas y más basadas en datos.

El dataset utilizado es de tipo tabular donde el archivo es un CSV, es decir, está organizado en filas y columnas, separando los dataset por una coma o punto y coma. Cada instancia u observación corresponde a un día específico de la semana, mientras que el resto de features describe una característica del fenómeno analizado:

- **DiaSemana** → Día de la semana, codificado entre el 0 y el 6 (0 = lunes, 6 domingo).
- **Feriado** → Indica si el día es feriado (False = no, True = sí).

- **Estacion** → Indica cuál de las cuatro estaciones del año es ('PRIMAVERA', 'VERANO', 'OTOÑO', 'INVIERNO')
- **Clases** → Indica si el día hubo clases (False = no, True = sí).
- **Pandemia** → Indica si el día corresponde a pandemia (False = no, True = sí).

En base a esto, contiene otra columna más, la cuál es la variable target:

- **Cantidad** → Cantidad de pasajeros registrados, utilizada para definir la variable objetivo del modelo.

Con estos datos históricos, permite entrenar al modelo a partir de observaciones reales, así generar predicciones confiables y facilitar la identificación de patrones y tendencias.

Para el presente proyecto se utilizaron los siguientes programas:

- **Excel** para la lectura y limpieza de los datasets. Versión: 2508 Build 16.0.19127.20302.
- **Visual Studio Code** donde se escribirá y gestionará el código fuente de manera eficiente utilizando **Python**. Versión 1.106.3.
- **Python** para la normalización y análisis de los datos. Versión: 3.14.0.
- **Librerías Python:**
 - **Pandas** para manipular, limpiar y analizar los datos tabulares (DataFrames). Versión: 2.3.3+.
 - **Numpy** para hacer los cálculos y operaciones numéricas necesarios así obtener el promedio, la mediana, la moda, etc. Versión: 2.3.4+.

- **Matplotlib** para mostrar los datos calculados en tablas y gráficos 2D.
Versión: 3.10.7+.
- **Scikit – Learn** para utilizar el Machine Learning y pueda ser entrenada y capaz de analizar los datos, y así, mostrar los resultados esperados.
Versión 1.7.2.
- **Tkinter** para crear una ventana donde se mostrarán los gráficos con los resultados esperados, así el usuario tendrá la posibilidad de aplicar el dataset y ver los resultados del algoritmo elegido. Versión: Tcl/Tk 8.6.15+.
- **Pillow** librería para manipulación de imágenes. Versión: 12.0.0+.
- **PyMuPDF** librería para manipulación de archivos pdf. Versión: 1.26.7+.

Resultados Esperados

Poder estimar con el mayor grado de certeza posible, en base a la información suministrada por los Datasets, la cantidad de unidades requeridas para cubrir las demandas de transporte de los usuarios de la tarjeta SUBE (en este caso sólo de colectivos) con el menor costo posible para la empresa prestadora del servicio, logrando un punto óptimo costo/beneficio.

Conclusión y Proyección

Tras el análisis de los datos, se observó que la mayor cantidad de pasajeros fue en el día de la semana **Miércoles**, a menor **Martes**, luego **Jueves**, **Lunes**, **Viernes**, **Sábado** y con la menor cantidad de pasajeros de la semana, **Domingo**, sin embargo, **Sábado** como **Domingo** cuentan con muy poca cantidad de pasajeros a comparación del resto de días de la semana. Con respecto a la temporada del año, **invierno** como **primavera** están a la cabeza por la cantidad de pasajeros transportador, y aunque muy igualados, a menor cantidad **Otoño** y por último **Verano**.

Podríamos concluir que, dentro de los días de la semana, **Miércoles** es el día con mayor demanda de transporte, lo cual se encuentra en la semana laboral (**Lunes - Viernes**), como días escolares para estudiantes y/o universitarios, y además, pese a que el **Sábado** en distintas localidades de Buenos Aires suceda que hay personas que estudien o trabajen, la cantidad de pasajeros sigue siendo muy baja a comparación de los días laborales, y su cantidad sea similar con el **Domingo**. En base a todo esto, habría que focalizar en disminuir las líneas de colectivo o choferes de los fines de semana y movilizarlos en los días laborales, principalmente los **Miércoles** por los datos observados.

Por otro lado, se observó que al momento que empezó y finalizó la pandemia COVID-19, hubo una gran baja de pasajeros en los transportes, pero también, en días feriados, la cantidad de pasajeros baja drásticamente.

Para asegurar la fiabilidad y efectividad de los datos, se utilizaron dos métricas:

- **F1 SCORE:** Es una medida cuantitativa y escalar de que tan bien el modelo logra clasificar en 0 y 1 después de aplicar un umbral. Mide el equilibrio entre la precisión y la recuperación de un modelo, donde 1 indica una precisión y una

recuperación perfectas, y 0 implica un rendimiento deficiente. En el caso de este modelo, tuvo un puntaje del **0.89**, lo cual entra en una **muy buena** puntuación ($F1 \geq 0.85$), permitiendo confiabilidad en la decisión binaria que toma el modelo.

- **LOG LOSS:** Mide qué tan buenas y realistas son las probabilidades que produce la regresión logística. Evalúa si el modelo asigna probabilidades altas a lo que realmente ocurre y bajas a lo que no ocurre. Si el valor es bajo, las probabilidades del modelo coinciden con lo que realmente pasa, cuanto más bajo es, mejor calidad son las probabilidades. En el caso de este modelo, tuvo un puntaje del **0.29**, lo cual es cercano al 0, no es perfecto, ni excelente, pero aun así está en las expectativas, el modelo tiene una **buena** puntuación (dentro del rango 0.20 – 0.40), respecto a qué tan creíbles son las probabilidades que estima el modelo.

En resumen, el modelo utilizado con el presente dataset es confiable en sus decisiones y creíble con lo que ha estimado. Podemos confiar en su efectividad y análisis en el conjunto de datos.

Proyección

Actualmente se está actualizando el dataset 2026, lo cual, cuando el año finalice, se actualizará en la base de datos del modelo, y con eso, podrá ser utilizado para predecir con mayor precisión el año 2027, lo mismo con el dataset 2027 para predecir en 2028, y así sucesivamente.

Continuando con el análisis de la hipótesis, se podría profundizar el análisis con las ubicaciones donde los pasajeros abordaron el transporte, para poder predecir con mayor precisión en distintas localidades y/o ciudades de Buenos Aires.

Incluir la posibilidad de que el usuario pueda abrir dentro del programa, un calendario para ingresar los feriados que necesite para poder actualizar el modelo en el caso que los datasets hayan sido actualizados y necesite ingresar los días fueron feriados.

Glosario

Dataset: Conjunto de datos estructurado diseñado para su análisis o uso en un modelo de Machine Learning. Contiene registros o instancias, los registros se componen de un conjunto de variables o atributos (features), que describen sus características, y opcionalmente de una etiqueta o variable objetivo (target), que indica el resultado asociado a esa observación específica.

Fuente de Datos: Origen de los datos donde se utilizaron en el proyecto y cómo fueron tratados o creados antes del análisis.

Normalización: Conjunto de reglas y técnicas que se aplican para minimizar la redundancia de datos y reducir la probabilidad de encontrar anomalías en la información, así asegurar la integridad de los datos.

Separador: Carácter específico utilizado para distinguir entre campos de datos (columnas) dentro de una misma fila.

Etiquetado de Campos de Columna: Primera fila del documento, contiene los encabezados o títulos que definen que tipo de dato contiene cada columna.

Poda: Limpieza o reducción del conjunto de datos, eliminando filas (registros) innecesarias, irrelevantes o valores nulos para facilitar el análisis.

Outliers: Son los valores que se alejan mucho del resto de los datos. Puede ser por:

- Un valor muy alto comparado con los demás.
- Un valor muy bajo comparado con los demás.

Aparecen por errores de medición, datos excepcionales reales o cambios o casos muy especiales. El objetivo es corregirlos, eliminarlos o analizarlos individualmente.

Referencias y Fuentes

None | *SUBE - Cantidad de transacciones (usos) por fecha.* (s. f.).

<https://datos.transporte.gob.ar/dataset/sube-cantidad-de-transacciones-usos-por-fecha>

None | *SUBE - Cantidad de tarjetas (usuarios) por fecha.* (s. f.).

<https://datos.transporte.gob.ar/dataset/sube-cantidad-de-tarjetas-usuarios-por-fecha>

None | *SUBE - Cantidad de tarjetas (usuarios) por día.* (s. f.).

<https://datos.transporte.gob.ar/dataset/sube-cantidad-de-tarjetas-usuarios-por-dia>

colaboradores de Wikipedia. (2025, 4 noviembre). *Pandemia de COVID-19*. Wikipedia, la Enciclopedia Libre. https://es.wikipedia.org/wiki/Pandemia_de_COVID-19

colaboradores de Wikipedia. (2025, 4 noviembre). *Pandemia de COVID-19*. Wikipedia, la Enciclopedia Libre. https://es.wikipedia.org/wiki/Pandemia_de_COVID-19

Argentina.gob.ar. (s. f.). <https://www.argentina.gob.ar/>

Software utilizado:

Welcome to Python.org. (2025, 19 noviembre). Python.org. <https://www.python.org/>

NumPy. (s. f.). <https://numpy.org/>

Numpy. (s. f.). *GitHub - numpy/numpy: The fundamental package for scientific computing with Python.* GitHub. <https://github.com/numpy/numpy>

pandas - Python Data Analysis Library. (s. f.). <https://pandas.pydata.org/>

Pandas-Dev. (s. f.). *GitHub - pandas-dev/pandas: Flexible and powerful data analysis / manipulation library for Python, providing labeled data structures similar to R data.frame objects, statistical functions, and much more.* GitHub.

<https://github.com/pandas-dev/pandas>

scikit-learn: machine learning in Python — scikit-learn 1.7.2 documentation. (s. f.).

<https://scikit-learn.org/stable/>

Scikit-Learn. (s. f.). *GitHub - scikit-learn/scikit-learn: scikit-learn: machine learning in Python.*

GitHub. <https://github.com/scikit-learn/scikit-learn>

Matplotlib — Visualization with Python. (s. f.). <https://matplotlib.org/>

Matplotlib. (s. f.). *GitHub - matplotlib/matplotlib: matplotlib: plotting with Python.* GitHub.

<https://github.com/matplotlib/matplotlib>

Python-Pillow. (s. f.). *GitHub - python-pillow/Pillow: Python Imaging Library (Fork).*

GitHub. <https://github.com/python-pillow/Pillow>

Pymupdf. (s. f.). *GitHub - pymupdf/PyMuPDF: PyMuPDF is a high performance Python library for data extraction, analysis, conversion & manipulation of PDF (and other) documents.*

GitHub. <https://github.com/pymupdf/PyMuPDF>