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

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” esta 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 cuál 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.

Tipo → Es el tipo de transporte asociado al registro, hay cuatro categorías en el dataset: COLECTIVO, TREN, SUBTE y LANCHA.

Feriado → También considerado de tipo lógico, contiene los datos 0 (False) y 1 (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 0 (False) y 1 (True) que determina si en el registro correspondiente hubo o no clases.

Pandemia → También considerado de tipo lógico, contiene los datos 0 (False) y 1 (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), los restantes dos (Tipo y Estacion) son 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.

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 **categórico** 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 de un tipo de transporte en ciertas líneas o un día o temporada determinado).

El modelo lineal estimara la probabilidad de demanda en base a los **features** (Tipo, 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).
- **Tipo** → Tipo de transporte, representado por cuatro categorías ('COLECTIVO', 'LANCHA', 'SUBTE', 'TREN').

- **Feriado** → Indica si el día es feriado (0 = no, 1 = sí).
- **Estacion** → Indica cuál de las cuatro estaciones del año es ('PRIMAVERA', 'VERANO', 'OTOÑO', 'INVIERNO')
- **Clases** → Indica si el día hubo clases (0 = no, 1 = sí).
- **Pandemia** → Indica si el día corresponde a pandemia (0 = no, 1 = 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 taburales (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.

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.

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.

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