

Data Preparation

David Guzmán Leyva - A01706417
Enrique Santos Fraire - A01705746
Leonardo Alvarado Menéndez - A01705998
Oscar Enrique Delgadillo Ochoa - A01705935

Índice

Índice	2
DataSet	4
Descripción del data set	4
Seleccionar los datos	4
Base lógica para la inclusión/exclusión	4
Preparación de los datos	4
Not moving	5
Formateo	5
Atributos derivados	5
Formateo	5
Limpieza	5
Atributos derivados	5
Formateo	6
Limpieza	6
Formateo	6
Limpieza	6
Registros generados	6
Antenas	6
Transformar valores de atributos existentes	6
Atributos derivados	6
Registros generados	6
Atributos derivados	6
Limpieza	7
Registros generados	7
Cleaning data	7
Transformar valores de atributos existentes	7
Formateo	7
Registros generados	7
Atributos derivados	7
Limpieza	7
Formateo	7
Limpieza	8
Registros generados	8
Merge data	8
Atributos derivados	8
Transformar valores de atributos existentes	8
Atributos derivados	8
Registros generados	8
Formateo	8

Formateo	8
Limpieza	9
Registros generados	9
Viajes2	9
Transformar valores de atributos existentes	9
Atributos derivados	9
Limpieza	9
Registros generados	9
OrigenDestino	9
Transformar valores de atributos existentes	9
Registros generados	10
Heatmap	10
Limpieza	10
Atributos derivados	10
Registros generados	10

DataSet

Descripción del data set

Nuestro dataset final va a tener los siguientes parámetros:

- PHONE_ID: identificador único del usuario que está realizando el viaje..
- Comuna Origen: lugar en el que empezó el viaje.
- Comuna Destino: destino final.
- Start_time: hora en la que inició el viaje.
- End_time: hora en la que finalizó el viaje.

Seleccionar los datos

Base lógica para la inclusión/exclusión

Tomamos la decisión de no eliminar ningún atributo debido a que los datos proporcionados (timestamp, phone_id, bts_id, lat y lon) por el Centro de Data Science de Chile son útiles para obtener información relevante como, distancias, velocidades, tiempo de movilidad y comunas, que son datos primordiales para obtener la matriz de viajes solicitada.

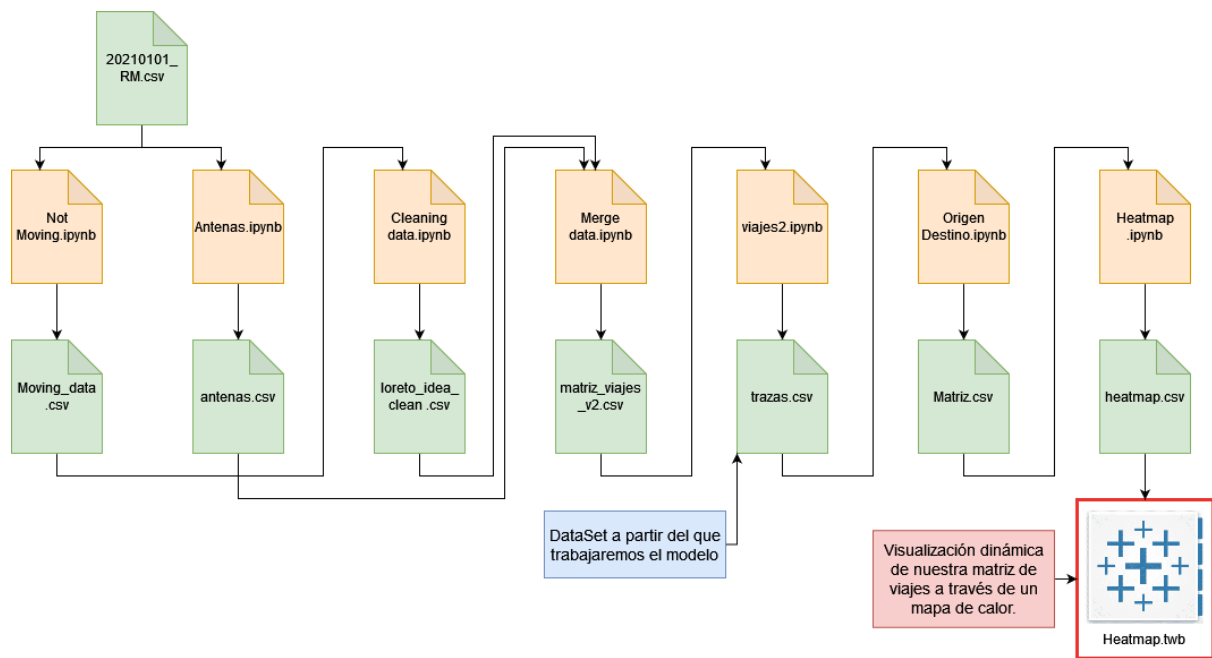
Preparación de los datos

A continuación se muestra un diagrama general sobre el proceso de preparación de nuestros datos. Consta de 7 partes distribuidas en jupyter notebooks, partiendo del csv que se nos otorgó, realizando las fases de preparación que aplicaban en cada uno y generando un csv nuevo por cada jupyter, finalizando con el DataSet limpio para la matriz de viajes, que posteriormente se verá representado en un mapa de calor dentro de Tableau.

En resumen cada fase realizó lo siguiente:

1. Not moving - DataSet original a DataSet con gente que si se mueve.
2. Antenas - Limpieza de antenas repetidas con sus coordenadas y comunas.
3. Cleaning data - Gente que se está moviendo a menos de 150 km/h.
4. Merge - Combinar los datos anteriores en un solo dataset de PHONE_ID, timestamp, coordinates, tiempo_min y comuna.
5. Viajes2 - Agrupar las conexiones a la misma coordenada en un rango continuo de tiempo.
6. OrigenDestino - Genera la matriz de viajes siguiendo las condiciones para que sean posibles.
7. Heatmap -Conteo de viajes que hay de una comuna a otra para poder visualizarse.

En seguida, se describen a detalle las actividades realizadas en cada fase.



Not moving

Formateo

Partimos del DataSet original y los organizamos por *PHONE_ID* y *timestamp*, de manera que sean separados cada usuario y su recorrido a lo largo de todo el día.

Atributos derivados

Mediante la fórmula de Haversine obtenemos la distancia entre dos puntos utilizando la latitud y longitud (columnas *lat* y *lon*) en una nueva columna *dis_trans*.

Formateo

Reiniciamos los índices ya que las operaciones los alteran.

Limpieza

Borramos la columna anterior de *index* ya que se reiniciaron anteriormente.

Atributos derivados

Reemplazamos los NaN's por ceros, esto debido a que al inicio de cada id no hay distancias que comparar y los NaN's interfieren en los cálculos.

Mediante una función iterativa y un acumulador, separamos aquellos usuarios que sí tuvieron un recorrido de los que no en 2 arreglos distintos.

Formateo

Asignamos al *PHONE_ID* como index.

Limpieza

Gracias al arreglo de los usuarios que no se mueven, utilizamos dicha variable como parámetro para borrar esos registros del DataSet.

Formateo

Reiniciamos los índices para recuperar el *PHONE_ID* como columna.

Limpieza

Eliminamos la columna de distancia recorrida (*dis_trans*), pues ya no es necesaria.

Registros generados

Se guarda el nuevo DataSet en un csv (*moving_data.csv*).

Antenas

Transformar valores de atributos existentes

Partimos del DataSet original y cambiamos el tipo de dato de las columnas de *lat* y *lon* de float a string para poder realizar una concatenación.

Atributos derivados

Creamos una columna de *coordinates*, que es la concatenación de *lat* y *lon* separadas por una coma.

Registros generados

Mediante la librería *geopy*, utilizamos las *coordinates* y en una columna *direction* se le aplica una función reversa con *geolocator* y obtenemos un objeto de datos en las coordenadas. extrayendo con una función adicional únicamente la comuna y verificando la unicidad de 52 comunas.

Atributos derivados

En una columna *direc* utilizamos una función lambda para extraer la dirección de la columna creada anteriormente.

A partir de *direc*, utilizamos una función que busca en la dirección la comuna en la que se encuentra, comparándolo con un arreglo que contiene las 52 comunas y guardando el resultado en la columna *comuna*.

Limpieza

Eliminamos las columnas de *direction* y *direc*, pues ya no son necesarias.

Verificamos que no haya datos nulos o inusuales. Encontramos que quedaba un registro nulo, por lo que manualmente buscamos su comuna y se la asignamos.

Registros generados

Se guarda el nuevo DataSet en un csv (*antenas.csv*).

Cleaning data

Transformar valores de atributos existentes

Partimos del DataSet *moving_data*, generado en el paso de **Not Moving**, y cambiamos el tipo de dato de la columna de *timestamp* a pandas datetime para realizar los cálculos.

Formateo

Organizamos los datos por *PHONE_ID* y *timestamp*, de manera que sean separados cada usuario y su recorrido a lo largo de todo el día.

Registros generados

Se genera una nueva columna llamada *drop* que contiene ceros, esto para identificar los registros que serán borrados bajo cierta condición.

Atributos derivados

Mediante la fórmula de Haversine obtenemos la distancia entre dos puntos utilizando la latitud y longitud (columnas *lat* y *lon*) en una variable. Posteriormente se obtiene el tiempo transcurrido restando la hora del registro anterior al actual en otra variable. Finalmente se obtiene la velocidad dividiendo la distancia sobre el tiempo y se guarda el resultado en una tercera variable.

En caso de que la velocidad sea mayor a 150, se registra un 1 en la columna *drop* del registro y se vuelven a hacer los cálculos con el registro siguiente hasta que la velocidad calculada sea menor a 150.

Limpieza

Creamos un nuevo dataset a través de una máscara que nos devuelva únicamente los registros que se hayan mantenido con un *drop* de 0.

Formateo

Reiniciamos los índices ya que las operaciones los alteran.

Limpieza

Borramos la columna anterior de *index* ya que se reiniciaron anteriormente.
Borramos la columna *Unnamed: 0* ya que no es necesaria.

Registros generados

Se guarda el nuevo DataSet en un csv (*loreto_idea_clean.csv*).

Merge data

Atributos derivados

Partimos de los DataSet *loreto_idea_clean*, generado anteriormente en el paso de **Cleaning data**. Creamos la columna de *tiempo_min* multiplicando *tiempo_trans* por 60, obteniendo así el tiempo en minutos.

Transformar valores de atributos existentes

Cambiamos el tipo de dato de la columna de *timestamp* a pandas datetime para realizar los cálculos.

Cambiamos el tipo de dato de las columnas de *lat* y *lon* de float a string para poder realizar una concatenación.

Atributos derivados

Creamos una columna de *tiempo_min*, que es la resta del tiempo del registro actual menos el anterior en horas.

Creamos una columna de *coordinates*, que es la concatenación de *lat* y *lon* separadas por una coma.

Registros generados

Generamos un nuevo DataSet donde conservamos solo *PHONE_ID*, *timestamp*, *coordinates* y *tiempo_min*.

Unimos este nuestros datos con el DataSet *antenas*, generado anteriormente en el paso de **Antenas**, utilizando como parámetro de unión la columna *coordinates*.

Formateo

Organizamos los datos por *PHONE_ID* y *timestamp*, de manera que sean separados cada usuario y su recorrido a lo largo de todo el día.

Formateo

Reiniciamos los índices ya que las operaciones los alteran.

Limpieza

Borramos la columna anterior de *index* ya que se reiniciaron anteriormente.
Borramos la columna *Unnamed: 0* ya que no es necesaria.

Registros generados

Se guarda el nuevo DataSet en un csv (*matriz_viajes_v2.csv*).

Viajes2

Transformar valores de atributos existentes

Partimos del DataSet *matriz_viajes_v2*, generado en el paso de **Merge Data**, y cambiamos el tipo de dato de la columna de *timestamp* a pandas datetime para realizar los cálculos.

Atributos derivados

Reemplazamos los NaN's por ceros, esto para facilitar las operaciones al inicio de cada id.

Limpieza

Borramos la columna *Unnamed: 0* ya que no es necesaria.

Registros generados

A través de un diccionario auxiliar se generó un nuevo DataFrame, donde en vez de tener cada registro de los id's conectados a la misma antena en diferentes horas ahora es un registro unico que indica la comuna en la que se encuentra, cuál fue su hora de entrada y su hora de salida de la conexión a esa antena.

Se guarda el nuevo DataSet en un csv (*trazas.csv*).

OrigenDestino

Transformar valores de atributos existentes

Partimos del DataSet *trazas*, generado en el paso de **Viajes2**, y cambiamos el tipo de dato de las columnas de *Hora_entrada* y *Hora_salida* a pandas datetime para realizar los cálculos.

Registros generados

A través de un diccionario auxiliar se generó un nuevo DataFrame, donde se guardará la matriz de viajes, indicando el traslado del id, desde su comuna de origen hasta su comuna de destino, con su hora de salida y su hora de llegada, así como la distancia

que recorrió y la duración del viaje. Esto siguiendo las condiciones de que para terminar un viaje debe mantenerse por lo menos 20 minutos quieto, su duración no debe exceder las 2 horas ni los 25 kilómetros de distancia.

Se guarda el nuevo DataSet en un csv (*Matriz.csv*).

Heatmap

Limpieza

Partimos del DataSet *Matriz*, generado en el paso de **OrigenDestino**, y borramos la columna Unnamed: 0 ya que no es necesaria.

Atributos derivados

Creamos una columna de *Concatenados*, que es la concatenación de *Comuna_Origen* y *Comuna_Destino* separadas por una coma.

Registros generados

Se crea un arreglo con el conteo de todos diferentes viajes y se guarda en un nuevo DataFrame.

Con un diccionario auxiliar se crea una matriz que guarda el origen, el destino y el número de viajes realizados entre ambos.

Se genera otro DataFrame con este diccionario.

Se guarda el nuevo DataSet en un csv (*heatmap.csv*).