Enrutamiento de vehículos para Lima

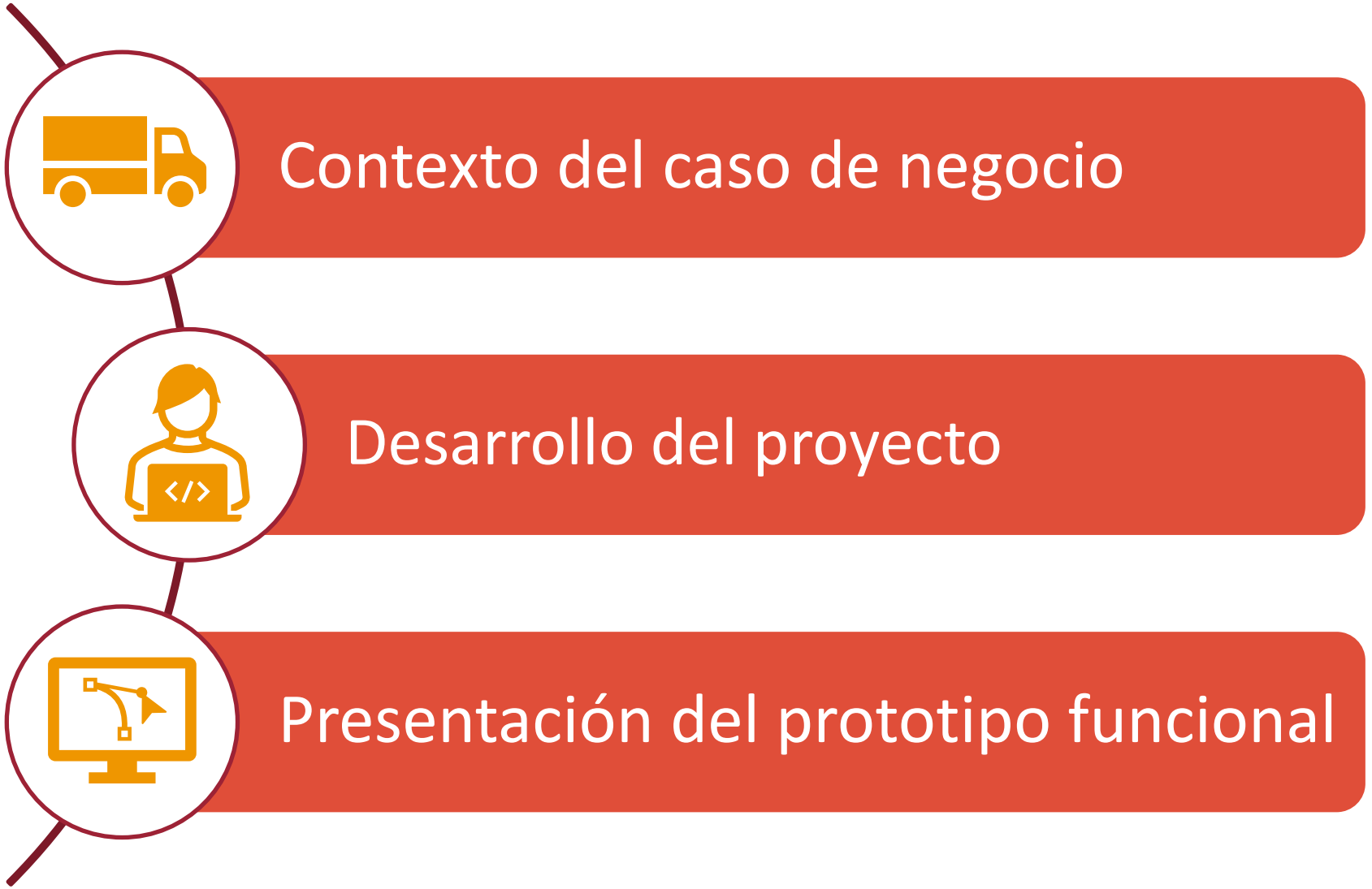
Sigma

TI LATAM

Bruno Miranda

Proyecto del curso Python Fundamentals

INDICE

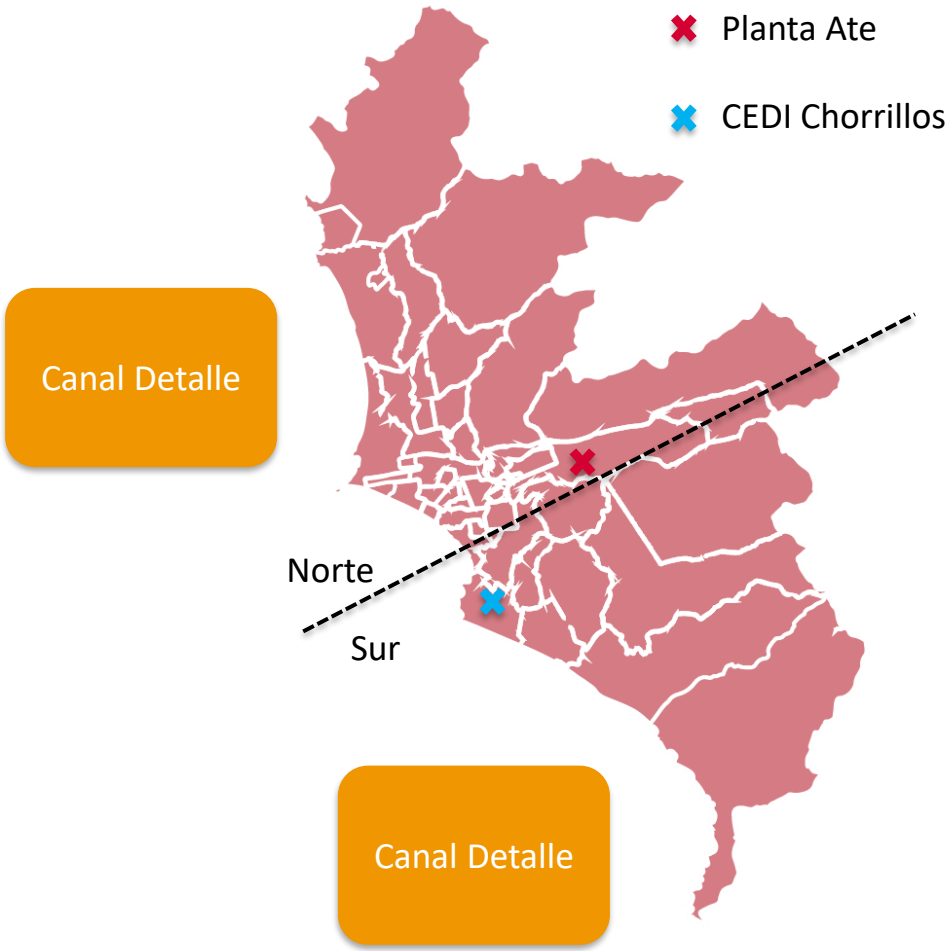


Sigma

Contexto del caso de negocio

Situación actual y proceso a abordar

DIVISIÓN ACTUAL DE RUTAS DE DISTRIBUCIÓN EN LIMA



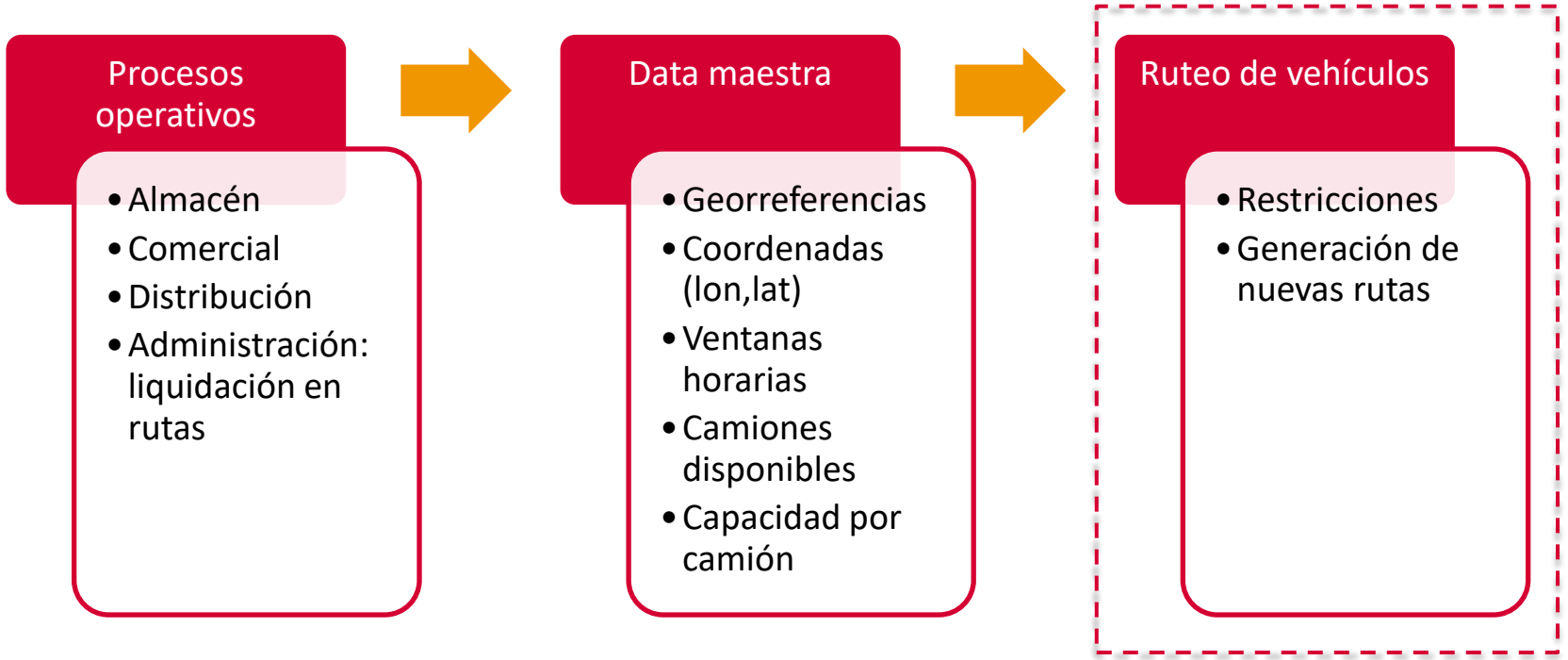
ATE:

- Distribución fija.
- Cuenta con vehículo propios.
- Ate envía a Chorrillos para que este distribuya en el sur.
- Manejan SAP para asignar rutas.
- Se extienden al sur para canales FOSE y Autoservicios

CHORRILLOS:

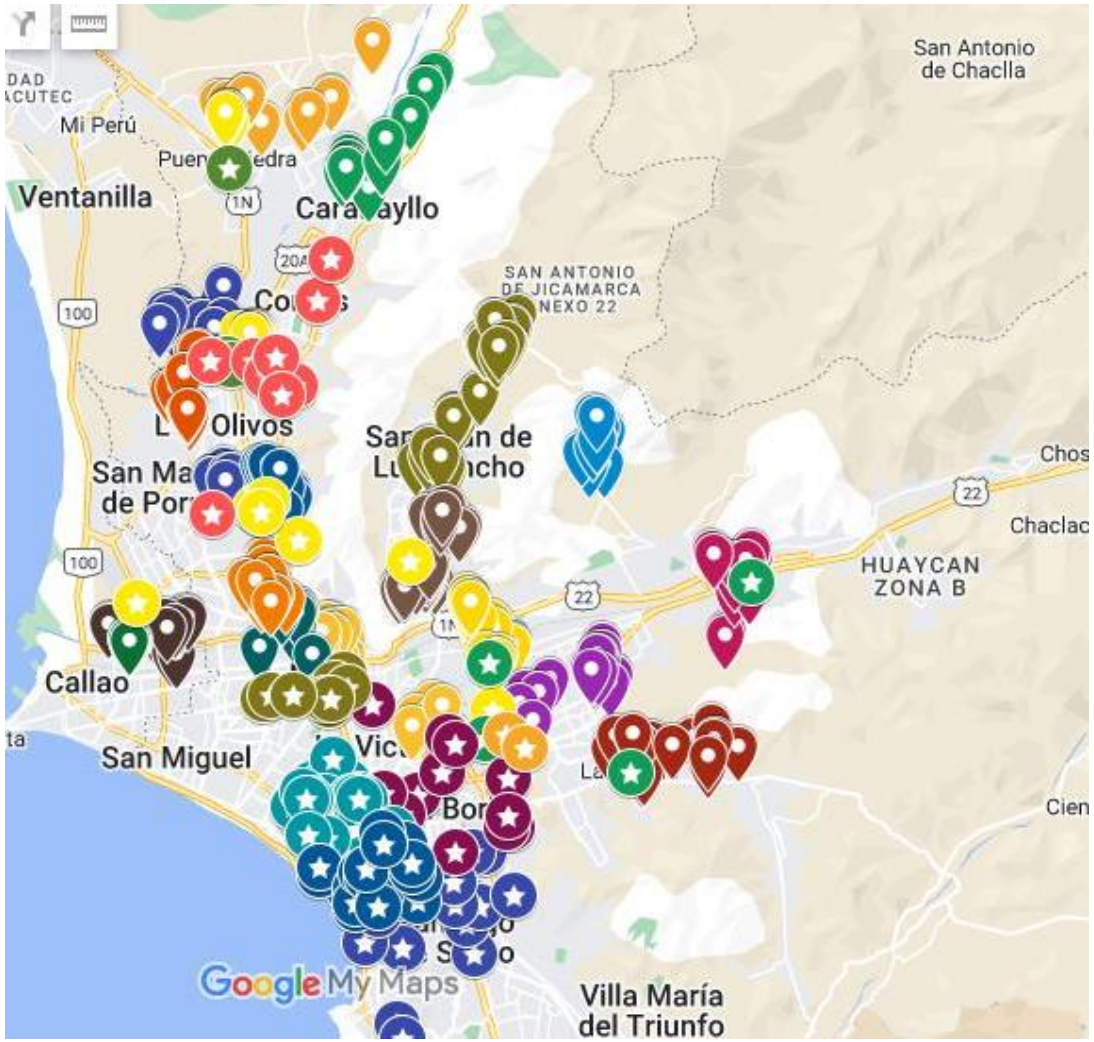
- FOSE, distribución dinámica.
- Moderno y Detalle, distribución fija.
- Cuenta con vehículos alquilados.
- Manejan SAP y GisAT para asignar rutas.
- Se extienden al norte para canales FOSE y Autoservicios.

Escenarios/Procesos a abordar



Con el fin de resolver los inconvenientes en el área de distribución, todos los procesos deben ser abordados

Asignación de ruta fija



Asignación de ruta fija

PROGRAMACION DE DISTRIBUCION - BRAEDT S.A.

martes, 2 de Agosto de 2022

SECUENCIA	PUERTA	1			2			3			4			5			6		
1	Placa	SERVICIO ESPECIAL			F8X-899			D1M-791			C7X-875			C7Y-915			D1P-751		
	Chofer	PRIORIDAD DE CARGA			GARCIA			EDER PEREZ			L. QUISPE			ALANIA			NUÑEZ		
	Auxiliar				HERNANDEZ		10	RIVAS	CORONADO	14	PUCAHUALA	POMA	17	LEON	ÑAUPARI	20	ARENA	TEODORO	16
5:00 am		TOTTUS	CASO	CACERES	210368	11	311	210357	32	293	210348	37	510	210354	84	1,005	210375	52	1718
5:40 am	Ruta	ACOPIO	RIVERA	A. PEREZ				210370	6	643	204570	10	309						
											204561	9	183						
2	Placa	AAV-711			C7O-880			A1J-882			F8X-854			F1E-704			A1J-880		
	Chofer	J. GUTIERREZ			FASABI			FERNANDEZ			ORE			TITO			BULLON		
	Auxiliar	LOPEZ	G. GERSON	18	VARELA	R.SARMIENTO	13	RIVADENEYRA	BASTIDAS	14	BARBOZA		11	NAVARRO		13	RUIZ	ZEVALLOS	13
5:40 am	Ruta	210346	67	491	210371	21	597	210374	14	325	210353	17	144	210351	27	256	210349	34	350
6:20 am					210370	4	355	210372	20	669	204568	1	19	204565	1	4			
														204568	1	93			
3	Placa	ABO-787			ABH-733			D3K-779			C4N793			D1F-719			F2A-848		
	Chofer	LIFONSO			VITOR			MICHEL			RUJEL			PALACIOS			HUAMAN		
	Auxiliar	TORRES	RIME	20	ALLPACCA	CHAVEZ	18	ROMAN			J.SILVA	EDEL R.	16	HUERTAS		12	ELMER RAMOS	ERICK	18
6:20 am	Ruta	210358	76	329	210355	63	TORRES	210373	19	1,400	210359	31	271	210369	21	1,509	210356	62	308
7:00 am		210370	4	76							204622	13	167						
											204617	1	21						
											210370	2	40						
4	Placa	F2A-929			F1G-866			D1F-760			BAI-836			ABE-738			B0P-919		
	Chofer	ALBORNOZ			VERGARA			PELAEZ			NUÑEZ			COILA			HINOSTROZA		
	Auxiliar	A.QUISPE	REA		D. SULCA		13	LIMAYMANTA			GUILLEN	EXTRA	15	ANCHANTE		13	I.VICTORIO		14
7:00 am	Ruta	204561	38	1,847	210347	29	258	210350	25	164	210360	34	270	210352	33	205	210362	31	156
7:40 am											205859	6	97	204565	1	3	204621	5	73
											210370	2	454						

SERVICIO ESPECIAL		
PRIORIDAD DE CARGA		
	TARDE	
CENCOSUD	D. MEZA	J. GUTIERREZ
TARDE	HUAMANI	L. HUAMAN

Vacaciones		
Conductor	Auxiliar	SUSP
PORRAS	P.PEREZ	
MEJIA	LEVANO	

DM	Permiso	Falto
		SARMIENTO
		TOVAR

Renuncia	Por confirmar	Disponibilidad
		Conductor
		M. GOMEZ

Disponibilidad		
Auxiliar	Por cubrir	Disponible
	PIPA	
	F.CASTRO	

Taller	Motivo	TALLER
C7O-879	MOTOR	
C8G-900	M.PREVENTIVO	

Horario de carga

Placa, chofer, auxiliar

Ruta

Cantidad de clientes

Volumen en kg

Hora fin de ruta

Caso de estudio - Resumen

Necesidad del negocio

> Organizar un Sistema de distribución uniforme para Ate y Chorrillos.
> Reducir tiempos y costos de operación en la distribución de los productos.

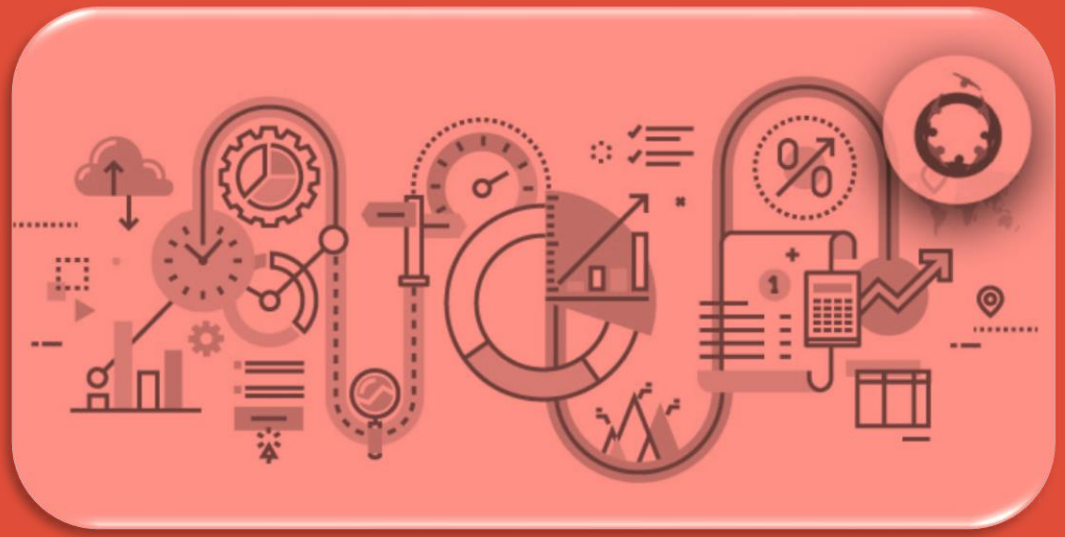
Solución

> Desarrollar/contratar/adquirir un Software de Ruteo de Vehículos que contemple las restricciones del área de distribución y realice seguimiento.
> Digitalizar los procesos que se llevan en la logística de la distribución.
> Trackear los vehículos de distribución.

Beneficios del negocio

> Obtención de un sistema robusto de ruteo de vehículos para programar la distribución.
> Reducción de tiempos de operación.
> Reducción de horas extras pagadas a vendedores/auxiliaries.
> Reducción de combustible empleado en las rutas.
> Ahorro en costos de operación.

Desarrollo del proyecto



Estructura del proyecto y métodos empleados

Taxonomía del proyecto

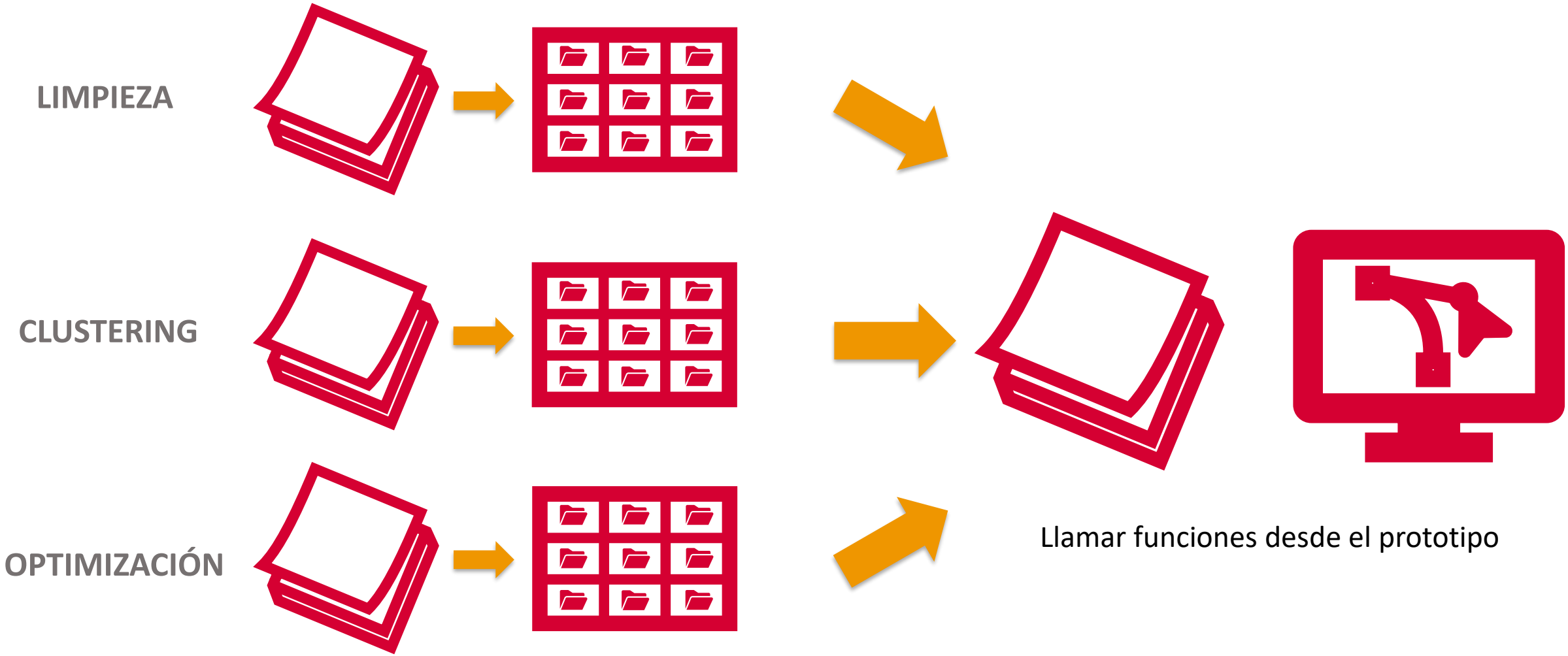
Proyecto_distribucion

```
---------- datasets----------  -
           |                   |---rutas.csv
           |
---------- modulos----------   -
           |                   |---init.py
           |                   |---crear_clusters-----------
           |                   |                            |---init.py
           |                   |                            |---plotear_clusters.py
           |
           |                   |---directorio----------------
           |                   |                            |---init.py
           |                   |                            |---filedialog.py
           |                   |---optimización-------------
           |                   |                            |---init.py
           |                   |                            |---ruteo_vehículos.py
           |                   |---preprocesamiento------
           |                                                |---init.py
           |                                                |---encoding.py
           |                                                |---limpieza_datos.py
           |
---------- notebooks--------  -
                               |---distribucion_limpieza.ipynb
                               |---distribución_clusters.ipynb
                               |---distribución_optimización.ipynb
                               |---distribución_prototipo.ipynb
```

Librerías usadas

Librerías	Uso
pandas	Manipulación de dataframes
numpy	Manipulación de arrays
sklearn	Preprocesamiento, clustering, entre otros
plotly	Ploteo interactivo de mapas
docplex	Algoritmos de optimización para rutas
tkinter	Abrir cuadros de diálogo para cargas de archivo
ipywidgets	Crear prototipo para product owner
IPython	Mostrar controles en ipywidgets
matplotlib	Ploteo en general
io	Leer archivos binarios
os	Cambiar directorio para llamar librerías
warnings	Ignorar advertencias del sistema
panel	Mejor visualización para dataframes
modulos	Librerías locales para el prototipo

Proceso de desarrollo del software



Sigma

LIMPIEZA

Exploración del dataframe, diccionario de datos, preprocesamiento

Diccionario de datos

Variable	Descripción
Fecha	Datetime del pedido
Ce.	Sociedad
Día	Día de la semana de lunes a sábado
Interlocut	Ruta de distribución actual
Distrito	Distrito donde se encuentra punto de venta
Cliente	Código de cliente al cual pertenece punto de venta
D.Solic/Ct	Crear prototipo para product owner
Dirección	Dirección del punto de venta
Latitud	Coordenada latitud del punto de venta
Longitud	Coordenada longitud del punto de venta
Pedido	Volumen en kg pedido por el cliente
Real	Volumen en kg que se puede despachar
T.Ruta	Tipo de ruta al cual pertenece el cliente: R.Primaria, R.Secundaria

Carga y exploración inicial del dataframe

Cargar archivos

```
import pandas as pd
import numpy as np
import os
from functools import partial
# establecer ruta de archivo
current_directory=os.getcwd()
os.chdir(os.path.split(current_directory)[0])
file=r'Datasets\rutas.xlsx'

# crear funcion para ir llamandola más adelante
read_excel=partial(pd.read_excel,io=file,engine='openpyxl')

# cargar el archivo de distribucion
df=read_excel()
df.columns=[i.strip() for i in df.columns]
df.head()
```

Leer con pd.read_excel()

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5565 entries, 0 to 5564
Data columns (total 14 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   Fecha       5565 non-null   datetime64[ns]
 1   Ce.         5565 non-null   object        
 2   Día         5565 non-null   object        
 3   Interlocut  5565 non-null   object        
 4   Ruta real   5565 non-null   object        
 5   Distrito    5565 non-null   object        
 6   Cliente     5565 non-null   object        
 7   D.Solic/Ct  5564 non-null   object        
 8   Dirección   5559 non-null   object        
 9   Latitud     5558 non-null   object        
 10  Longitud    5558 non-null   object        
 11  Pedido      5559 non-null   float64       
 12  Real        5559 non-null   float64       
 13  T.Ruta      5413 non-null   object        
dtypes: datetime64[ns](1), float64(2), object(11)
memory usage: 608.8+ KB
```

Shape de (5565,14)

Verificar le existencias de valores nulos

```
df.isnull().sum()
```

```
Fecha           0
Ce.             0
Día             0
Interlocut      0
Ruta real       0
Distrito        0
Cliente         0
D.Solic/Ct      1
Dirección       6
Latitud         7
Longitud        7
Pedido          6
Real            6
T.Ruta        152
dtype: int64
```

Valores nulos

Funciones creadas para la limpieza

```
def escoger_variables(df,columns):
    # limpiar columnas
    df.columns=[i.strip() for i in df.columns]
    return df[columns].copy()

def isfloat(num):
    try:
        float(num)
        return True

    except ValueError:
        return False
```

```
def limpiar_dataframe(df):
    # definir tipos de variable
    tipo_variable={'Fecha':np.datetime64,'Ce.':str,'Día':str,'Interlocut':str,'Ruta real':str,'Distrito':str,
                   'Cliente':str,'D.Solic/Ct':str,'Dirección':str,'Latitud':float,'Longitud':float,'Pedido':float,
                   'Real':float,'T.Ruta':str}

    # Asignar la menor clase
    df.loc[df['T.Ruta'].isnull(),'T.Ruta']='R.secundario'

    # Eliminar valores nulos
    df.dropna(subset=['Latitud','Longitud'],inplace=True,axis=0)

    # Eliminar registros indefinidos
    index_indefinidos=df.loc[(df['Latitud'].apply(lambda x:isfloat(x))==False)|(df['Longitud'].apply(lambda
x:isfloat(x))==False)].index.tolist()
    df=df.drop(index=index_indefinidos).reset_index(drop=True)

    # cambiar tipo de variable
    for i in df.columns:
        df[i]=df[i].astype(tipo_variable[i])

    # retornar dataframe limpio
    return df
```

	Fecha	Ce.	Día	Interlocut	Ruta real	Distrito	Cliente	D.Solic/Ct	Dirección	Latitud	Longitud	Pedido	Real	T.Ruta
0	2022-09-01	374	jueves	204471	Expendios	Indefinido	6290000	EXPENDIO ATE	NaN	NaN	NaN	82.50	82.50	NaN
1	2022-09-01	374	jueves	204560	Makros	Indefinido	6247246	SPSA MAKRO COMAS	Indefinido	Indefinido	Indefinido	198.92	195.08	NaN
2	2022-09-01	374	jueves	204563	Expendios / Makro	Indefinido	6247249	SPSA MAKRO CALLAO	Indefinido	Indefinido	Indefinido	290.29	282.85	NaN
3	2022-09-01	374	jueves	204563	Expendios / Makro	Indefinido	6247644	SUPERMERCADOS PERUANOS S.A.	Indefinido	Indefinido	Indefinido	349.83	338.55	NaN
4	2022-09-01	374	jueves	204565	204565	SAN MARTIN DE PORRES	11674146	DELIVERY HERO DMART PERU SAC	AV ALFREDO MENDIOLA 6334 ...	-11.9525	-77.0696	11.51	11.51	R.Secundario

Dataframe inicial

	Día	Interlocut	Cliente	Latitud	Longitud	Pedido	Real	T.Ruta
0	jueves	204565	11674146	-11.952544	-77.069646	11.51	11.51	R.Secundario
1	jueves	204565	11559915	-11.953273	-77.069556	1.80	1.80	R.Secundario
2	jueves	204565	11612248	-11.948416	-77.080081	1.68	1.68	R.Secundario
3	jueves	204565	11641999	-11.936530	-77.065580	31.00	31.00	R.Secundario
4	jueves	204565	11558695	-11.926137	-77.059138	3.28	3.28	R.Secundario

Dataframe limpio

Funciones y clases creadas para el encoding

```python
class encode_cat_ruta(BaseEstimator,TransformerMixin):
    '''Clase que permita realizar transformar la variable T.Ruta'''
    def fit(self,X,y=None):
        return self
    def transform(self,X,y=None):
        X=np.where(X=='R.Principal',1,0)
        return X

def escalar_variables(df):
    # separar variables en numericas y categoricas
    df_num=df.select_dtypes(include=np.number).copy()
    df_cat=df.select_dtypes(exclude=np.number).drop(columns=['Cliente','Interlocut','Día']).copy()

    # definir atributos
    num_attrib=df_num.columns.tolist()
    cat_attrib=df_cat.columns.tolist()

    # definir pipeline
    full_pipe=ColumnTransformer([('num',StandardScaler(),num_attrib),
                                 ('cat',encode_cat_ruta(),cat_attrib)])

    # crear dataframe transformado
    df_tr=pd.DataFrame(full_pipe.fit_transform(df),columns=num_attrib+cat_attrib)

    # concatenar df_cleaned[['Interlocut','Cliente']] con df_tr y retornarlo
    return pd.concat([df[['Día','Interlocut','Cliente']].copy(),df_tr],axis=1)
```

	Día	Interlocut	Cliente	Latitud	Longitud	Pedido	Real	T.Ruta
0	jueves	204565	11674146	-11.952544	-77.069646	11.51	11.51	R.Secundario
1	jueves	204565	11559915	-11.953273	-77.069556	1.80	1.80	R.Secundario
2	jueves	204565	11612248	-11.948416	-77.080081	1.68	1.68	R.Secundario
3	jueves	204565	11641999	-11.936530	-77.065580	31.00	31.00	R.Secundario
4	jueves	204565	11558695	-11.926137	-77.059138	3.28	3.28	R.Secundario

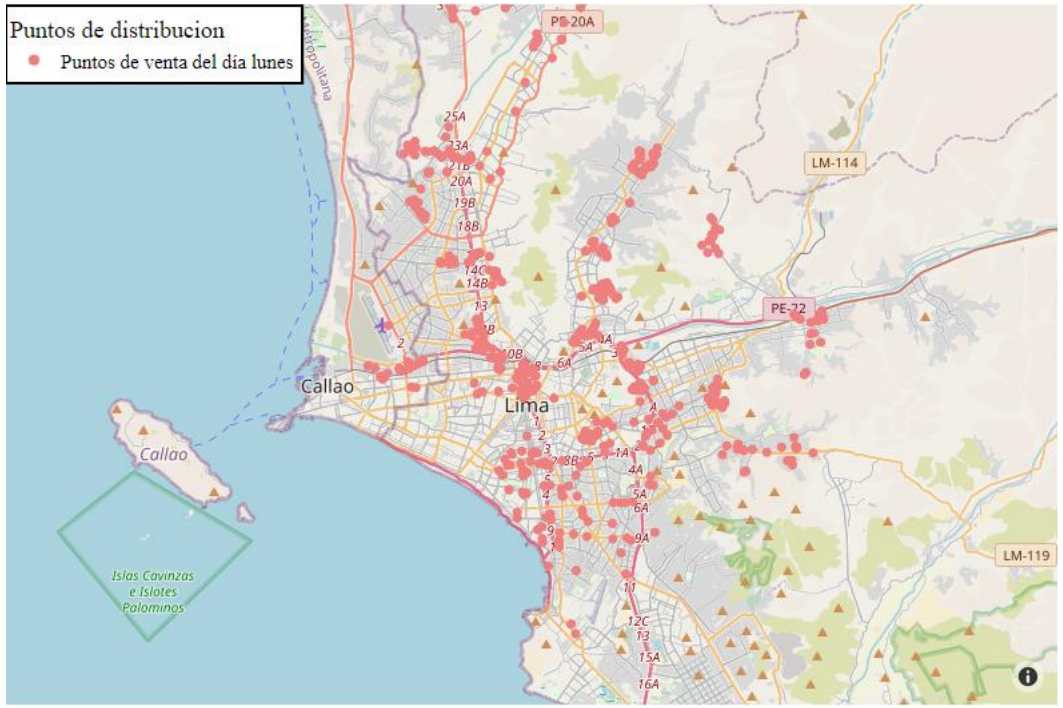
Dataframe limpio

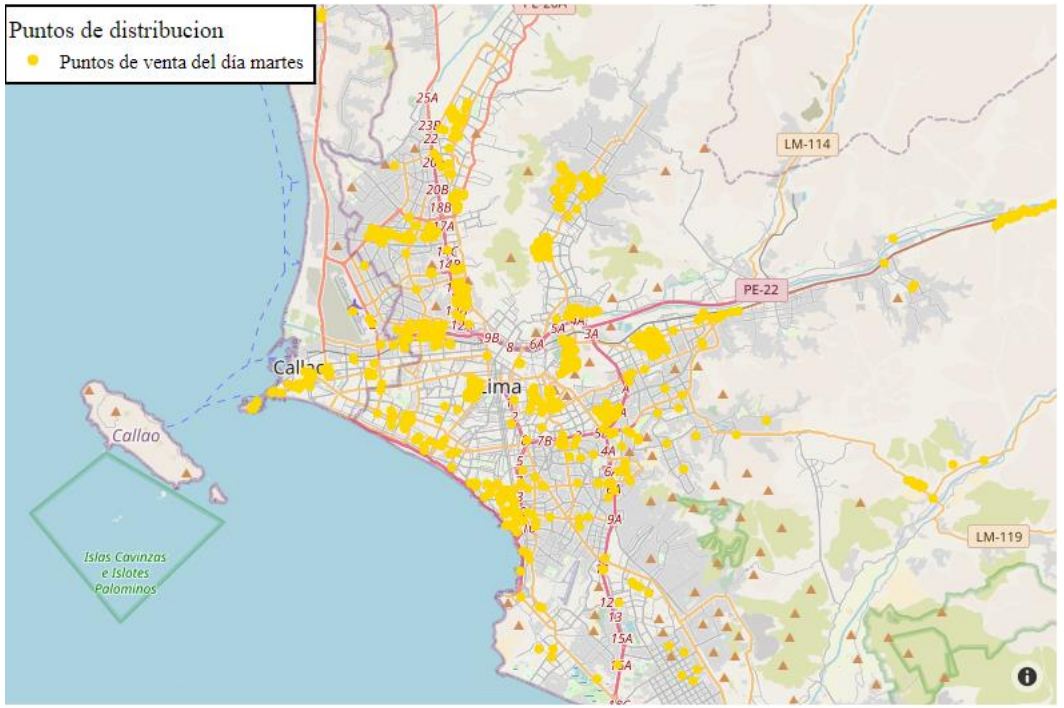
	Interlocut	Cliente	Latitud	Longitud	Pedido	Real	Día	T.Ruta
0	204565	11674146	1.144042	-0.784975	-0.062264	-0.058898	0.0	0.0
1	204565	11559915	1.132858	-0.783778	-0.234248	-0.232371	0.0	0.0
2	204565	11612248	1.207336	-0.924765	-0.236374	-0.234515	0.0	0.0
3	204565	11641999	1.389579	-0.730509	0.282944	0.289298	0.0	0.0
4	204565	11558695	1.548931	-0.644211	-0.208035	-0.205930	0.0	0.0

Dataframe escalado

Funciones creadas para el plot del rawmap

```
def plot_raw_map(df,dia):
    df_one=df.loc[df['Día']==dia].copy()
    # coordenadas del centro
    lat_center = df_one.Latitud.mean()
    lon_center = df_one.Longitud.mean()

    # inicializar la figura
    fig=go.Figure()
    # diccionario de colores
    dia_color={'lunes':'lightcoral','martes':'gold','miércoles':'lime','jueves':'hotpink','viernes':'dodgerblue','sábado':'peru'}
    # iterar por cada cluster
    fig.add_trace(go.Scattermapbox(
        name = f'Puntos de venta del día {dia}',
        lon = df_one.Longitud,
        lat = df_one.Latitud,
        mode="markers",
        marker =go.scattermapbox.Marker(size=8,color =dia_color[dia])))

    # crear leyendad
    legend=dict(
            x=0,
            y=1,
            title_font_family="Tunga",
            font=dict(
                family="Tunga",
                size=15,
                color="black"
            ),
            bgcolor="white",
            bordercolor="Black",
            borderwidth=2
        )
    # hacer update de leyenda
    fig.update_layout(legend=legend,legend_title_text='Puntos de distribucion',mapbox_style="open-street-map", margin=
{"r":0,"t":0,"l":0,"b":0},
            mapbox = {'center': {'lat': lat_center,
                                 'lon': lon_center
                                 },
                      'zoom': 10},
                      showlegend=True)
    # plotear
    fig.show()
```

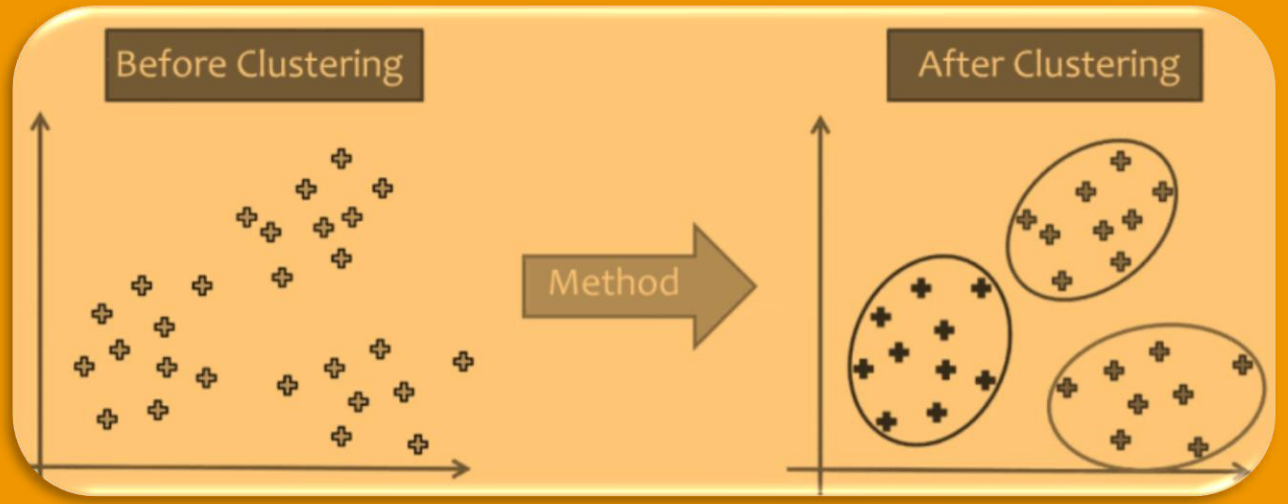




Uso de las función para el raw_map

Sigma

CLUSTERING



Aprendizaje no supervisado para obtener agrupaciones

Datos del negocio para el clustering

Variable	Descripción
Interlocut	Ruta de distribución actual
Distrito	Distrito donde se encuentra punto de venta
Cliente	Código de cliente al cual pertenece punto de venta
Latitud	Coordenada latitud del punto de venta
Longitud	Coordenada longitud del punto de venta
Real	Volumen en kg que se puede despachar
n_camiones	Número de clusters para las agrupaciones

Datos simplificados empleados:

Variable	Descripción
Latitud	Coordenada latitud del punto de venta
Longitud	Coordenada longitud del punto de venta
n_camiones	Número de clusters para las agrupaciones

Funciones para entrenar clusters

```python
def entrenar_cluster(df_done,df_cleaned,tipo_cluster,n_components):
    # definir dataframe para clustering
    df_cluster=df_done[['Latitud','Longitud']].copy()

    # entrenar modelo de clustering
    cluster=tipo_cluster(n_components).fit(df_cluster)

    # obtener labels
    labels=cluster.predict(df_cluster)

    # devovler dataframe con labels
    df_export=df_cleaned.copy()
    df_export.loc[:,'Cluster']=labels
    return df_export.sort_values(by='Cluster')
```

	Día	Interlocut	Cliente	Latitud	Longitud	Pedido	Real	T.Ruta
0	lunes	204565	11678464	0.348918	-1.193881	0.123503	0.136442	0.0
1	lunes	204565	11685152	1.143804	-1.211460	-0.293939	-0.282720	0.0
2	lunes	204565	11641999	1.139568	-0.918840	0.125002	0.137948	0.0
3	lunes	204565	6393786	1.061491	-0.833391	-0.316436	-0.305310	0.0
4	lunes	204565	11569310	1.018992	-0.642022	-0.222699	-0.211187	0.0

Dataframe del día escalado

	Día	Interlocut	Cliente	Latitud	Longitud	Pedido	Real	T.Ruta
0	lunes	204565	11678464	-11.990883	-77.080816	18.94	18.94	R.Secundario
1	lunes	204565	11685152	-11.936239	-77.081790	2.24	2.24	R.Secundario
2	lunes	204565	11641999	-11.936530	-77.065580	19.00	19.00	R.Secundario
3	lunes	204565	6393786	-11.941897	-77.060846	1.34	1.34	R.Secundario
4	lunes	204565	11569310	-11.944819	-77.050245	5.09	5.09	R.Secundario

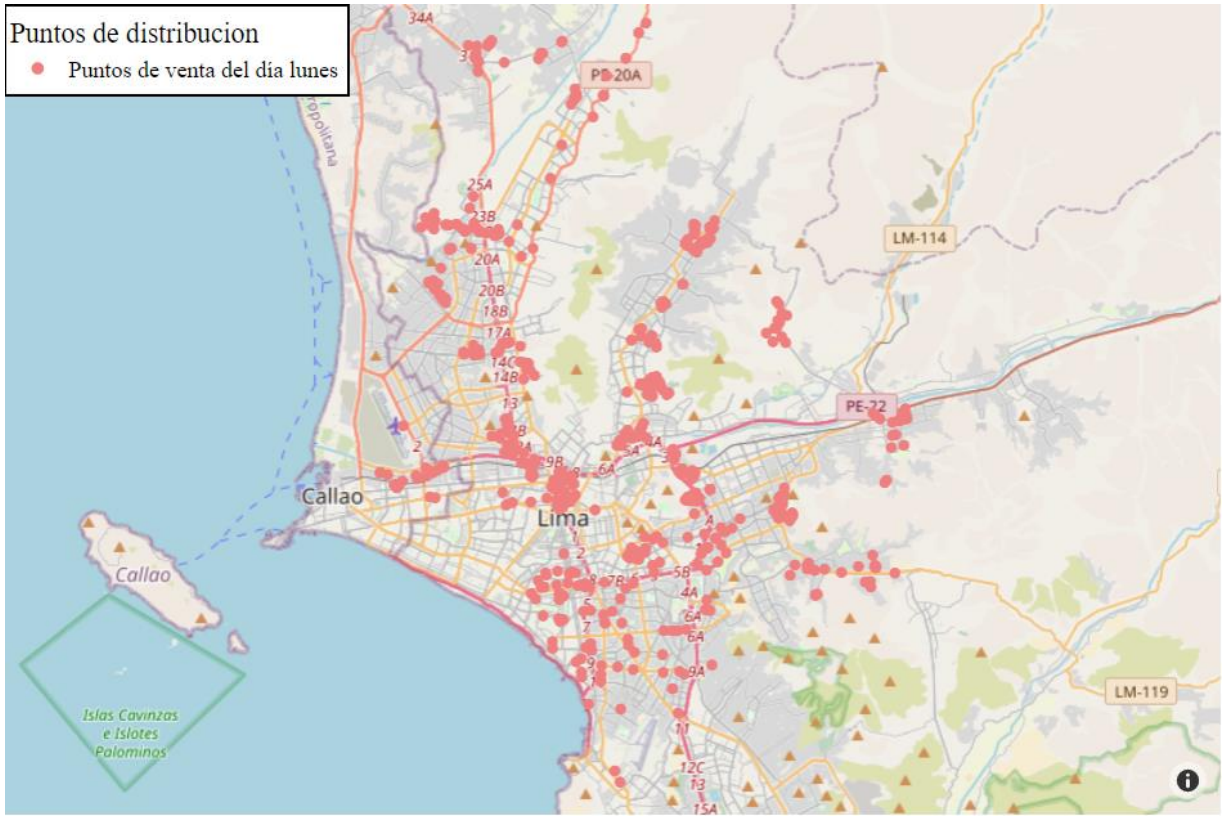
Dataframe del día sin escalar

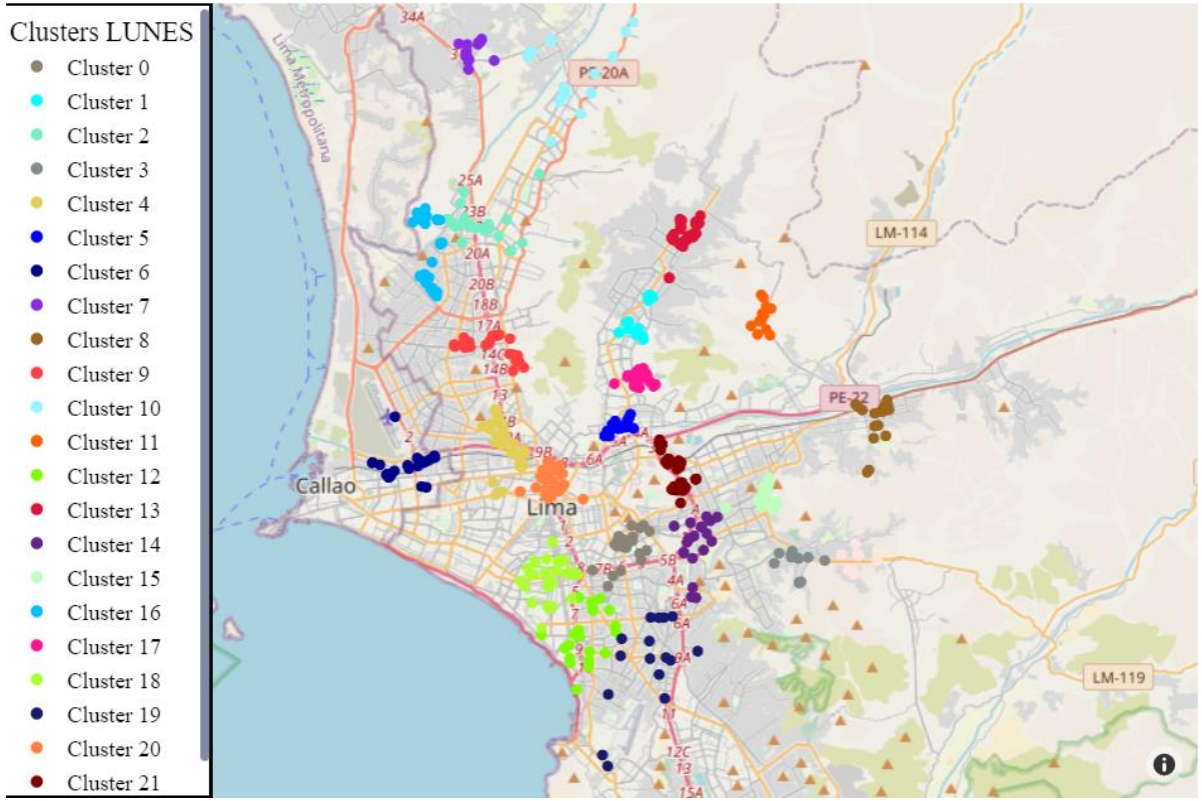
	Día	Interlocut	Cliente	Latitud	Longitud	Pedido	Real	T.Ruta	Cluster
437	lunes	210359	6361622	-12.095620	-76.924620	8.61	8.61	R.Principal	0
438	lunes	210359	6354140	-12.095664	-76.924533	7.15	7.15	R.Principal	0
439	lunes	210359	6349258	-12.084698	-76.925108	1.22	1.22	R.Principal	0
440	lunes	210359	6431025	-12.092074	-76.899924	2.19	2.19	R.Principal	0
441	lunes	210359	6341639	-12.078313	-76.899132	13.66	13.66	R.Principal	0
...	...	...	...	...	...	...	...	...	...
424	lunes	210358	11663924	-12.076040	-76.974470	1.22	1.22	R.Principal	22
423	lunes	210358	6438460	-12.077707	-76.975477	1.32	1.32	R.Principal	22
422	lunes	210358	11553834	-12.066326	-76.966684	2.41	2.41	R.Principal	22
483	lunes	210369	6340801	-12.111150	-76.988230	42.00	42.00	R.Principal	22
642	lunes	204622	6346596	-12.073840	-76.966530	8.00	8.00	R.Secundario	22

Dataframe del día sin escalar con clusters

Funciones para plotear clusters

```python
def plot_clusters(df_export):
    # crear diccionario de colores
    colors=['#8B8378','#00FFFF','#76EEC6','#838B8B','#E3CF57','#0000FF','#00008B','#8A2BE2','#9C661F','#FF4040',
    '#98F5FF','#FF6103','#7FFF00','#DC143C','#68228B','#C1FFC1','#00BFFF','#FF1493','#ADFF2F','#191970',
    '#FF8247','#800000','#FFE4E1']
    n_cluster=range(0,23)
    dic_colors=dict(zip(n_cluster,colors))

    # establecer centros para iniciar el mapa
    lat_center = df_export.Latitud.mean()
    lon_center = df_export.Longitud.mean()

    # inicializar la figura
    fig=go.Figure()

    # iterar por cada cluster
    for i in df_export.Cluster.unique():
        flag=df_export.loc[df_export.Cluster==i]
        fig.add_trace(go.Scattermapbox(
            name = f'Cluster {i}',
            lon = flag.Longitud,
            lat = flag.Latitud,
            mode="markers",
            marker =go.scattermapbox.Marker(size=8,color =dic_colors[i])))

    # crear leyenda
    legend=dict(
            x=0,
            y=1,
            title_font_family="Tunga",
            font=dict(
                family="Tunga",
                size=15,
                color="black"
            ),
            bgcolor="white",
            bordercolor="Black",
            borderwidth=2
        )
    # hacer update de leyenda
    fig.update_layout(legend=legend,legend_title_text=f"""Clusters {df_export['Día'][0].upper()}""",
                      mapbox_style="open-street-map", margin={"r":0,"t":0,"l":0,"b":0},
                      mapbox = {'center': {'lat': lat_center, 'lon': lon_center},
                            'zoom': 10},
                      showlegend=True)
    # plotear
    fig.show()
```





Uso de la clusterización

Sigma

OPTIMIZACIÓN

Restricciones de cantidad y capacidad de vehículos mediante cplex de IBM

Datos del negocio para la optimización

Variable	Descripción
Interlocut	Ruta de distribución actual
Distrito	Distrito donde se encuentra punto de venta
Cliente	Código de cliente al cual pertenece punto de venta
Latitud	Coordenada latitud del punto de venta
Longitud	Coordenada longitud del punto de venta
Real	Volumen en kg que se puede despachar
Capacidad	Kilogramos máximos transportador por camiones

Datos simplificados empleados:

Variable	Descripción
Latitud	Coordenada latitud del punto de venta
Longitud	Coordenada longitud del punto de venta
Real	Volumen en kg que se puede despachar
Capacidad	Kilogramos máximos transportador por camiones

Funciones para obtener las rutas

```python
def ruteo_dinamico(df,capacidad_camion):
    # definir datos de clientes
    n_clientes=len(df)
    clientes=list(range(1,n_clientes+1))
    nodos=[0]+clientes# puntos de los clientes más el nodo 0 que es el centro
    Q=capacidad_camion # capacidad de vehículos
    q=dict(zip(clientes,df.Real.tolist()))# capacidad de clientes

    # definir coordenadas
    centro=(-12.066883514538135, -76.97871779045312) # planta ATE
    coor_lat=np.insert(df.Latitud.values,0,centro[0],axis=0)# coordenadas de los puntos de venta + planta ATE
    coor_lon=np.insert(df.Longitud.values,0,centro[1],axis=0)

    # crear estructura de datos
    arcos={(i,j) for i in nodos for j in nodos if i!=j}
    distancia={(i,j):np.hypot(coor_lat[i]-coor_lat[j],coor_lon[i]-coor_lon[j]) for i in nodos for j in nodos if i!=j}

    # creación y optimización - CVRP
    mdl=Model('CVRP')

    # creando variables de decisión
    x=mdl.binary_var_dict(arcos,name='x')
    u=mdl.continuous_var_dict(nodos,ub=Q,name='u')

    # función objetivo: la menor ruta, minimizar
    mdl.minimize(mdl.sum(distancia[i,j]*x[i,j] for i,j in arcos))

    # establecer restricciones
    mdl.add_constraints(mdl.sum(x[i,j] for j in nodos if i!=j)==1 for i in clientes)# no se puede ir a varios nodos desde uno
    mdl.add_constraints(mdl.sum(x[i,j] for i in nodos if i!=j)==1 for j in clientes)
    mdl.add_indicator_constraints(mdl.indicator_constraint(x[i,j],u[i]+q[j]==u[j]) for i,j in arcos if i!=0 and j!=0)
    mdl.add_constraints(u[i]>=q[i]for i in clientes)

    # obtener la solución
    mdl.parameters.timelimit=120
    solucion=mdl.solve()

    # obtener los arcos donde se establece la ruta
    arcos_activos=[k for k in arcos if x[k].solution_value>0.9]

    # validar si estan ordenados
    arcos_activos_sorted=ordenar_arcos_activos(arcos_activos)

    # retornar estado de la solución y los arcos
    return mdl.get solve status(),arcos activos,arcos activos sorted
```

	Día	Interlocut	Cliente	Latitud	Longitud	Pedido	Real	T.Ruta	Cluster
0	lunes	210359	6361622	-12.095620	-76.924620	8.61	8.61	R.Principal	0
1	lunes	210359	6354140	-12.095664	-76.924533	7.15	7.15	R.Principal	0
2	lunes	210359	6349258	-12.084698	-76.925108	1.22	1.22	R.Principal	0
3	lunes	210359	6431025	-12.092074	-76.899924	2.19	2.19	R.Principal	0
4	lunes	210359	6341639	-12.078313	-76.899132	13.66	13.66	R.Principal	0

Dataframe del día sin escalar con cluster específico

```
mdl.get_solve_status()
```

```
<JobSolveStatus.FEASIBLE_SOLUTION: 1>
```

Status de la optimización

```
[(8, 5),
 (16, 20),
 (13, 18),
 (3, 19),
 (11, 10),
 (4, 1),
 (20, 6),
 (14, 0),
 (2, 15),
 (10, 8),
 (0, 4),
 (15, 16),
 (9, 7),
 (19, 12),
 (17, 9),
 (18, 14),
 (12, 11),
 (5, 17),
 (7, 13),
 (1, 2),
 (6, 3)]
```

Arcos de distribución

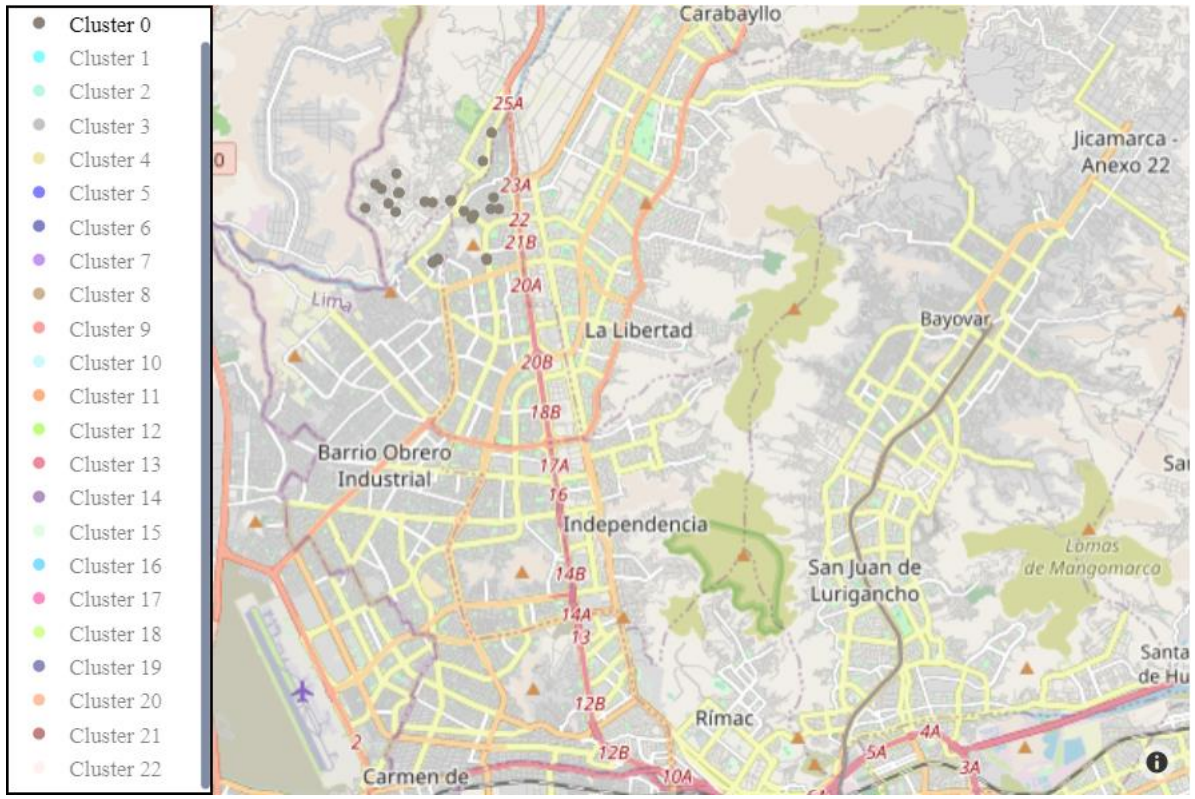
```
arcos_sorted
```

```
[0, 4, 1, 2, 15, 16, 20, 6, 3, 19, 12, 11, 10, 8, 5, 17, 9, 7, 13, 18, 14, 0]
```

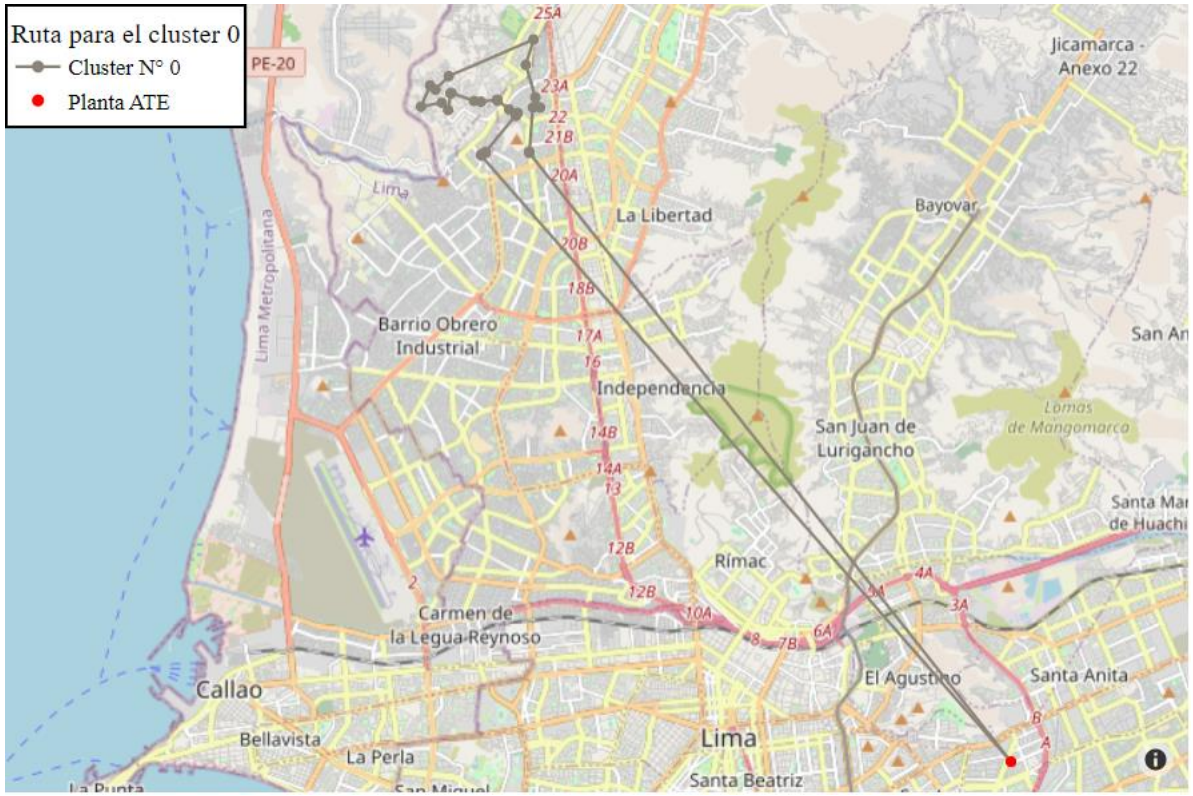
Arcos ordenados

Funciones para plotear mejor ruta

```
def plot_mejor_ruta(lat,lon,arcos_sorted,n_cluster,zoom=5):
    # crear diccionario de colores
    colors=['#8B8378','#00FFFF','#76EEC6','#838B8B','#E3CF57','#0000FF','#00008B','#8A2BE2','#9C661F','#FF4040',
    '#98F5FF','#FF6103','#7FFF00','#DC143C','#68228B','#C1FFC1','#00BFFF','#FF1493','#ADFF2F','#191970',
    '#FF8247','#800000','#FFE4E1']
    n_clusters=range(0,23)
    dic_colors=dict(zip(n_clusters,colors))

    # centros para iniciar el mapa
    lat_center = lat.mean()
    lon_center = lon.mean()

    # inicializar figura
    fig=go.Figure()

    # ordenar coordenadas
    lat_sorted=[lat[i] for i in arcos_sorted]
    lon_sorted=[lon[i] for i in arcos_sorted]

    # agregar coordenadas de cluster
    fig.add_trace(go.Scattermapbox(
            name = f'Cluster Nº {n_cluster}',
            lon = lon_sorted,
            lat = lat_sorted,
            mode="markers+lines+text",
            marker =go.scattermapbox.Marker(size=8,color =dic_colors[n_cluster])))

    # agregar coordenada central
    fig.add_trace(go.Scattermapbox(
            name = f'Planta ATE',
            lon = np.array(lon[0]),
            lat = np.array(lat[0]),
            mode="markers",
            marker =go.scattermapbox.Marker(size=8,color ='red')))

    # crear leyenda
    legend=dict(
            x=0,
            y=1,
            title_font_family="Tunga",
            font=dict(
                family="Tunga",
                size=15,
                color="black"
            ),
            bgcolor="white",
            bordercolor="Black",
            borderwidth=2)

    # hacer update de leyenda
    fig.update_layout(legend=legend,legend_title_text=f'Ruta para el cluster {n_cluster}',mapbox_style="open-street-map", margin=
{"r":0,"t":0,"l":0,"b":0},
            mapbox = {'center': {'lat': lat_center,
                                 'lon': lon_center
                                 },
                      'zoom': zoom},
                      showlegend=True)
    # plotear
    fig.show()
```



Cluster 0 del día lunes



Rutas trazadas en el mapa

Funciones para exportar ruta del día

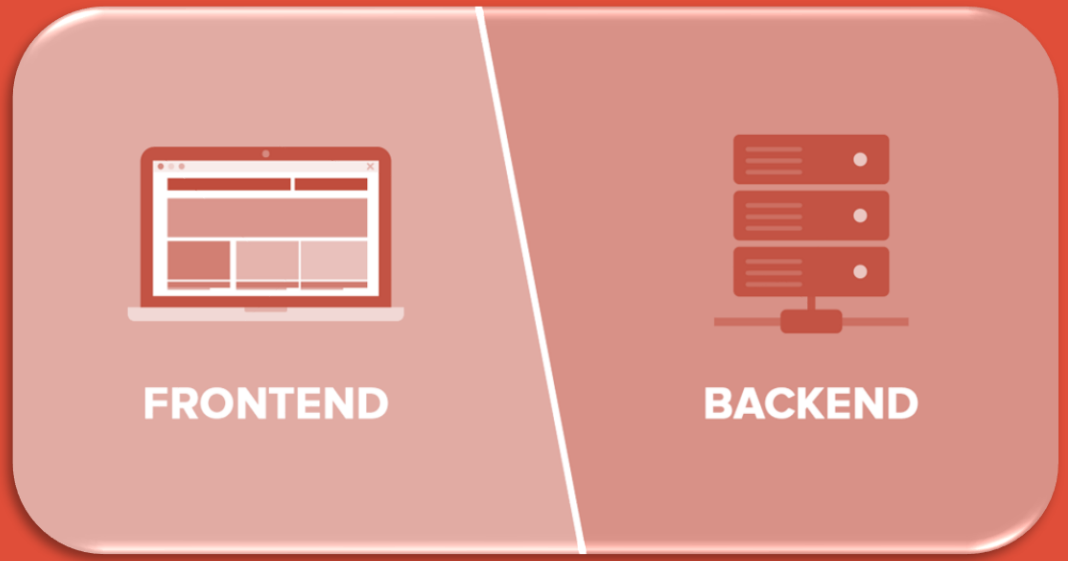
```python
def presentar_ruta(df_in,arcos):
    # quitar la planta ATE de los puntos (la cual está en los extremos) y restar el valor de 1
    # para que se acomode a los índices reales del dataframe
    list_puntos_venta=np.array(arcos[1:-1])-1

    # definir dataframe a devolver cuyas rutas están por orden acorde a los arcos
    df_sorted=pd.DataFrame()

    # llenar el dataframe acorde a list_puntos_venta los cuales están ordenados y los registros se obtienen
    # del .iloc del dataframe que ingresa
    for i in list_puntos_venta:
        df_sorted=df_sorted.append(df_in.iloc[i].to_dict(),ignore_index=True)

    # retornar dataframe ordenado por arcos a seguir según la ruta optimizada por cplex
    return df_sorted
```

Orden de la ruta

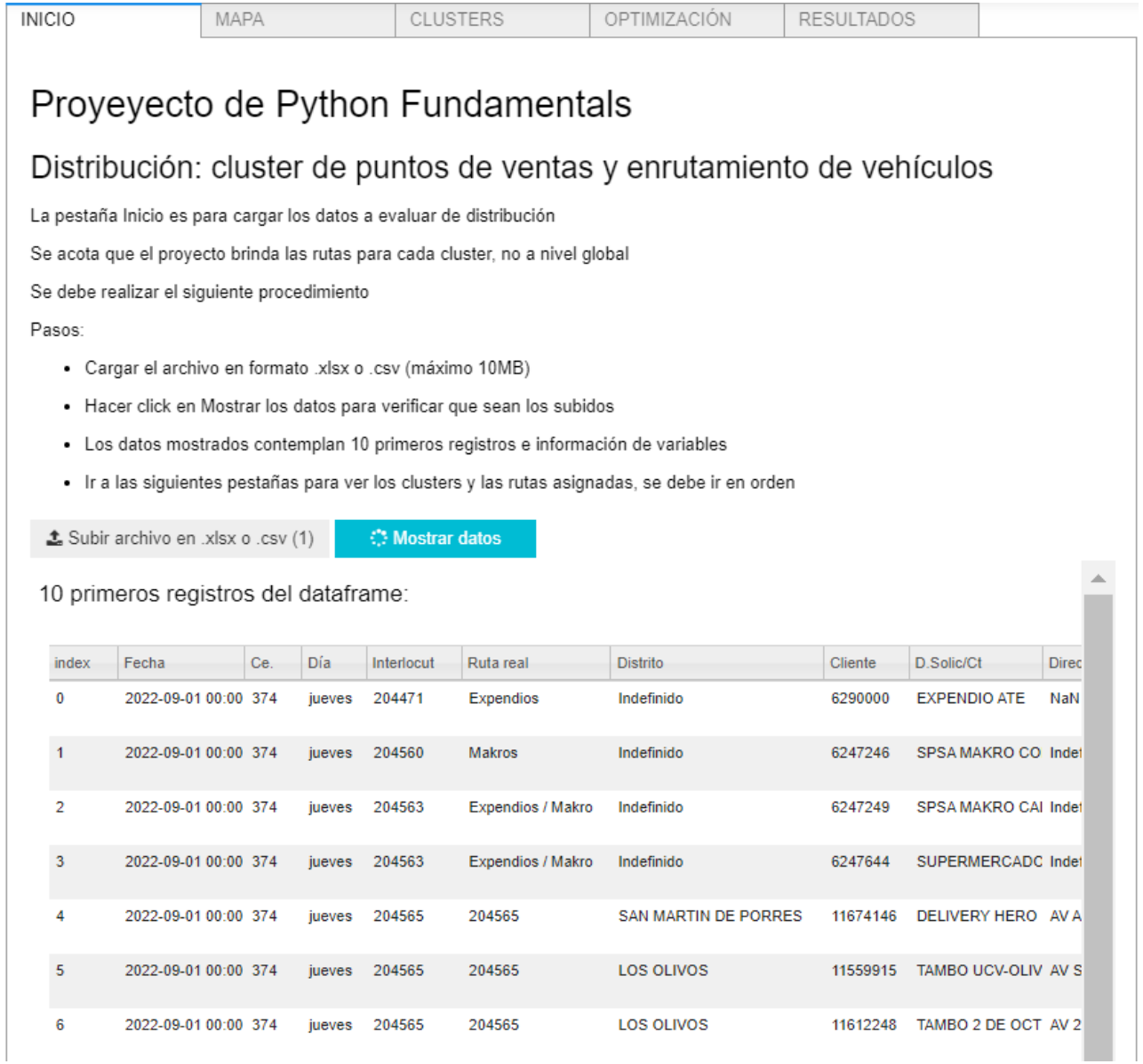
	Cliente	Cluster	Día	Interlocut	Latitud	Longitud	Pedido	Real	T.Ruta
0	11639925.0	1.0	lunes	210362.0	-11.966780	-77.088910	5.255	5.255	R.Principal
1	11641993.0	1.0	lunes	210362.0	-11.966657	-77.089026	16.720	16.720	R.Principal
2	11653623.0	1.0	lunes	210362.0	-11.966880	-77.089230	13.138	13.138	R.Principal
3	11683339.0	1.0	lunes	210362.0	-11.966737	-77.089584	1.400	1.400	R.Principal
4	11665129.0	1.0	lunes	210362.0	-11.966284	-77.089780	4.500	4.500	R.Principal
5	11637056.0	1.0	lunes	210362.0	-11.966330	-77.090090	4.350	4.350	R.Principal
6	11655220.0	1.0	lunes	210362.0	-11.965960	-77.090230	12.865	12.865	R.Principal
7	11637659.0	1.0	lunes	210362.0	-11.964909	-77.091197	5.740	5.740	R.Principal
8	11668296.0	1.0	lunes	210362.0	-11.960269	-77.092486	1.485	1.485	R.Principal
9	11663018.0	1.0	lunes	210362.0	-11.958810	-77.094560	0.375	0.375	R.Principal
10	11674826.0	1.0	lunes	210362.0	-11.959494	-77.096656	3.300	3.300	R.Principal
11	11647439.0	1.0	lunes	210362.0	-11.959730	-77.096480	12.040	12.040	R.Principal
12	11653622.0	1.0	lunes	210362.0	-11.959800	-77.096600	8.400	8.400	R.Principal
13	11663017.0	1.0	lunes	210362.0	-11.959880	-77.096460	6.060	6.060	R.Principal
14	11647438.0	1.0	lunes	210362.0	-11.962248	-77.094230	9.680	9.680	R.Principal
15	11667316.0	1.0	lunes	210362.0	-11.962872	-77.093625	1.480	1.480	R.Principal
16	11665372.0	1.0	lunes	210362.0	-11.965094	-77.091859	4.100	4.100	R.Principal
17	11661897.0	1.0	lunes	210362.0	-11.966910	-77.090630	2.675	2.675	R.Principal
18	11657968.0	1.0	lunes	210362.0	-11.966880	-77.090060	26.325	26.325	R.Principal
19	11683103.0	1.0	lunes	210362.0	-11.968491	-77.089054	7.200	7.200	R.Principal

Sigma

Presentación del prototipo funcional



Front-end para la generación de mejores rutas



PESTAÑA INICIO

INICIO | MAPA | CLUSTERS | OPTIMIZACIÓN | RESULTADOS

Distribución: Plot de puntos de ventas del día

En esta sección se muestra en un mapa los puntos de venta actuales

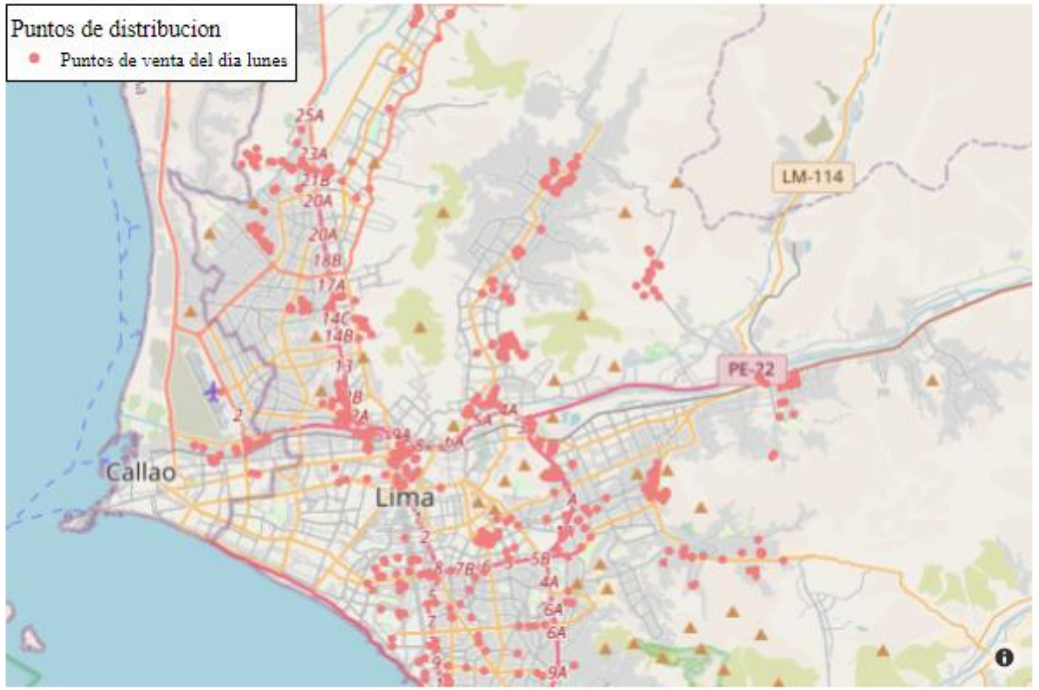
Se debe seleccionar el día y presionar el boton para plotear el mapa actual sin clusters

Después de ver los puntos, se debe dirigir a la pestaña de clusters

Día de la s... lunes martes miércoles jueves viernes

Plotear puntos de venta

	Día	Interlocut	Cliente	Latitud	Longitud	Pedido	Real	T.Ruta
0	lunes	204565	11678484	-11.990883	-77.080818	18.94	18.94	R.Secundario
1	lunes	204565	11685152	-11.936239	-77.081790	2.24	2.24	R.Secundario
2	lunes	204565	11641999	-11.936530	-77.065580	19.00	19.00	R.Secundario
3	lunes	204565	6393786	-11.941897	-77.060846	1.34	1.34	R.Secundario
4	lunes	204565	11569310	-11.944819	-77.050245	5.09	5.09	R.Secundario



PESTAÑA MAPA

INICIO | MAPA | CLUSTERS | OPTIMIZACIÓN | RESULTADOS

Distribución: Clustering por medio de KMeans y GaussianMixture

En esta sección se escoge también el dia y el modelo del cluster a emplear

Si bien un dia y tipo de clustering van en pareja, este proceso es estocástico, por lo que contar si se vuelve a presionar el botón botará otras agrupacoines

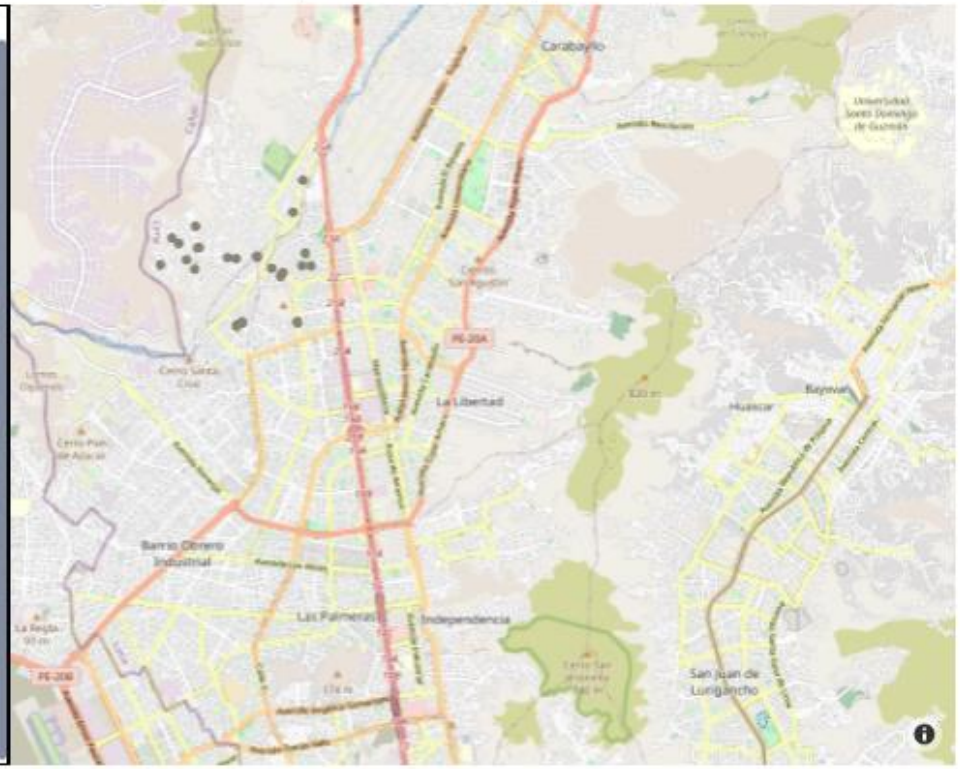
Después de realizar el clustering, proceder a la pestaña de optimización

Día de la s... lunes martes miércoles jueves viernes

Tipo cluster: KMeans GaussianMixture

Plotear clusters de los puntos de venta

Cluster 0
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 6
Cluster 7
Cluster 8
Cluster 9
Cluster 10
Cluster 11
Cluster 12
Cluster 13
Cluster 14
Cluster 15
Cluster 16
Cluster 17
Cluster 18
Cluster 19
Cluster 20
Cluster 21
Cluster 22



PESTAÑA CLUSTERS

INICIO | MAPA | CLUSTERS | OPTIMIZACIÓN | RESULTADOS

Distribución:

En esta sección se traza la mejor ruta para cada cluster obtenido en el pestaña previa

Este versión de Cplex de prueba solo permite obtener mejor enrutamiento para una cantidad determinada de puntos en un cluster, por lo que se realiza la optimización por cada cluster

Se presionar el boton para empezar con la generación de mejores rutas

Clusters para el día: LUNES

N° Cluster: Cluster 0 / Cluster 1 / Cluster 2 / Cluster 3 / Cluster 4

Plotear enrutamiento por cluster

	Día	Interlocut	Cliente	Latitud	Longitud	Pedido	Real	T.Ruta	Cluster
0	lunes	210349	11653816	-11.93585	-77.07690	2.92	2.92	R.Principal	0
1	lunes	210349	11659877	-11.92957	-77.09415	1.96	1.96	R.Principal	0
2	lunes	210349	11660584	-11.93295	-77.09360	2.52	2.52	R.Principal	0
3	lunes	210349	11661211	-11.93456	-77.08897	1.16	1.16	R.Principal	0
4	lunes	210349	11652864	-11.94498	-77.07760	2.68	2.68	R.Principal	0

La solución es de tipo: FACTIBLE

PESTAÑA OPTIMIZACIÓN

INICIO | MAPA | CLUSTERS | OPTIMIZACIÓN | RESULTADOS

Resultador del día LUNES para el Cluster 0

Rutas ordenadas de la planta

	Cliente	Cluster	Día	Interlocut	Latitud	Longitud	Pedido	Real	T.Ruta
0	11632268	0.0	lunes	210362	-11.945480	-77.087530	7.455	7.455	R.Principal
1	11655453	0.0	lunes	210362	-11.945230	-77.087450	17.520	17.520	R.Principal
2	11654182	0.0	lunes	210362	-11.945000	-77.086570	3.555	3.555	R.Principal
3	11639573	0.0	lunes	210362	-11.944750	-77.086570	10.340	10.340	R.Principal
4	11655226	0.0	lunes	210349	-11.937550	-77.080370	1.120	1.120	R.Principal
5	11653001	0.0	lunes	210349	-11.937180	-77.080240	0.600	0.600	R.Principal
6	11653181	0.0	lunes	210349	-11.936939	-77.079985	2.250	2.250	R.Principal
7	11685152	0.0	lunes	204565	-11.936239	-77.081790	2.240	2.240	R.Secundario
8	11685163	0.0	lunes	210349	-11.934453	-77.084160	3.570	3.570	R.Principal
9	11685164	0.0	lunes	210349	-11.934357	-77.084198	4.040	4.040	R.Principal
10	11685520	0.0	lunes	210349	-11.934327	-77.084261	2.765	2.765	R.Principal
11	11661705	0.0	lunes	210349	-11.934700	-77.087570	10.065	10.065	R.Principal
12	11661211	0.0	lunes	210349	-11.934560	-77.088970	1.160	1.160	R.Principal
13	11660584	0.0	lunes	210349	-11.932950	-77.093600	2.520	2.520	R.Principal
14	11662331	0.0	lunes	210349	-11.933020	-77.093810	1.900	1.900	R.Principal
15	11684336	0.0	lunes	210349	-11.936395	-77.094283	0.600	0.600	R.Principal
16	11668647	0.0	lunes	210349	-11.934903	-77.095605	2.970	2.970	R.Principal
17	11687763	0.0	lunes	210349	-11.935686	-77.099859	1.140	1.140	R.Principal
18	11668648	0.0	lunes	210349	-11.931434	-77.097985	1.160	1.160	R.Principal
19	11684455	0.0	lunes	210349	-11.932296	-77.096878	3.400	3.400	R.Principal
20	11659877	0.0	lunes	210349	-11.929570	-77.094150	1.960	1.960	R.Principal
21	11672371	0.0	lunes	210349	-11.922239	-77.076749	8.660	8.660	R.Principal
22	11667569	0.0	lunes	210349	-11.927317	-77.078331	5.700	5.700	R.Principal
23	6358022	0.0	lunes	210349	-11.933830	-77.076390	0.840	0.840	R.Principal
24	11547925	0.0	lunes	204568	-11.935853	-77.075411	2.700	2.700	R.Secundario
25	11653816	0.0	lunes	210349	-11.935850	-77.076900	2.920	2.920	R.Principal
26	11632267	0.0	lunes	210349	-11.944690	-77.077660	0.800	0.800	R.Principal
27	11652864	0.0	lunes	210349	-11.944980	-77.077600	2.680	2.680	R.Principal

PESTAÑA RESULTADOS

DEMO


GRACIAS