



GUÍA 4

DATASET CRÍMENES

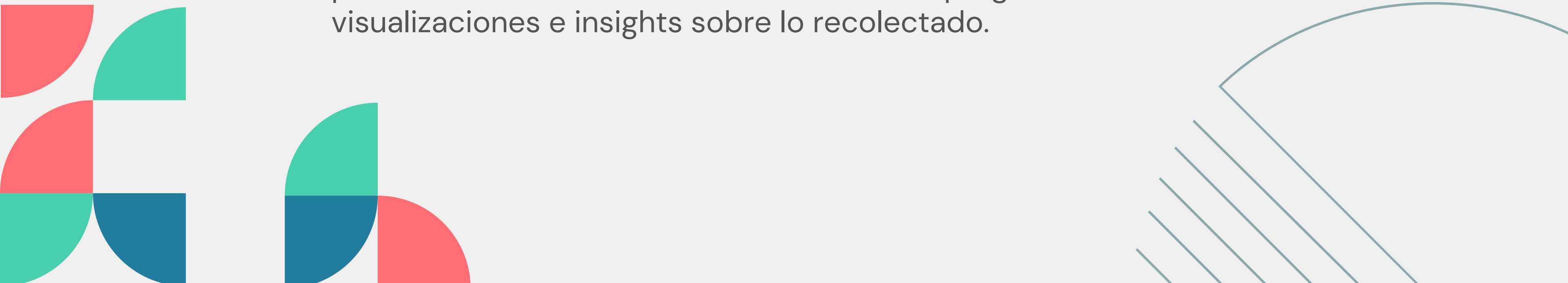
CHICAGO 19-24



INTRO

Se analizará un periodo del dataset de inicios del 2019 hasta la actualidad del 2024. (Aproximadamente 5 años). Se busca identificar cambios y a la medida de lo posible sugerir mejoras en cuanto a la seguridad en el estado de Chicago

El dataset cuenta con 1,363,148 registros de los cuales se hara el procesamiento de datos, formulacion de preguntas a resolver, visualizaciones e insights sobre lo recolectado.



DATA PROCESING

Como cualquier dataset no se presentará la información fácil de interpretar, se tendrá que procesar para poder resolver preguntas planteadas.



01 - DATA FALTANTE

02 - CODIFICACIÓN

03 - ESCALA

04 - SELECCIÓN

PROCESANDO LA DATA

01 - FALTANTE

Como se observa, de la totalidad del dataset sobrepasa los 1.3 millones de registros de crímenes y en ningún atributo (columna) se sobrepasa de 25 mil valores faltantes.

Siendo una cantidad baja se procede a trabajar con esa data y tratándola con imputaciones de valores faltantes usando media y moda para variables continuas y categóricas.



ID	0
Case Number	0
Date	0
Block	0
IUCR	0
Primary Type	0
Description	0
Location Description	7384
Arrest	0
Domestic	0
Beat	0
District	0
Ward	48
Community Area	2
FBI Code	0
X Coordinate	18660
Y Coordinate	18660
Year	0
Updated On	0
Latitude	18660
Longitude	18660
Location	18660
Historical Wards 2003-2015	23439
Zip Codes	18660
Community Areas	22824
Census Tracts	23073
Wards	22814
Boundaries - ZIP Codes	22817
Police Districts	22700
Police Beats	22697

dtype: int64

PROCESANDO LA DATA

01 - FALTANTE

```
# Imputar valores faltantes para variables continuas usando la media
df['Latitude'].fillna(df['Latitude'].mean(), inplace=True)
df['Longitude'].fillna(df['Longitude'].mean(), inplace=True)

# Imputar valores faltantes para variables categóricas con la moda
df['Location Description'].fillna(df['Location Description'].mode()[0], inplace=True)

# Usar KNN para imputar datos continuos (si hay más columnas numéricas con valores faltantes)
knn_imputer = KNNImputer(n_neighbors=5)
df[['Latitude', 'Longitude']] = knn_imputer.fit_transform(df[['Latitude', 'Longitude']])
```

```
columns_to_check = ['X Coordinate', 'Y Coordinate', 'Location', 'Historical Wards 2003-2015',
                    'Boundaries - ZIP Codes', 'Zip Codes', 'Census Tracts', 'Wards',
                    'Police Districts', 'Police Beats']

# Eliminar filas que tengan valores nulos en cualquiera de estas columnas
df = df.dropna(subset=columns_to_check)
```

```
# 2. Imputar valores faltantes en columnas con pocos valores faltantes (moda)
df['Ward'].fillna(df['Ward'].mode()[0], inplace=True)
df['Community Areas'].fillna(df['Community Areas'].mode()[0], inplace=True)
```

```
# 3. Imputar categóricas faltantes con una nueva categoría 'Desconocido'
df['Location Description'].fillna('Desconocido', inplace=True)
```

- Imputar valores faltantes para variables continuas usando la media
- Imputar valores faltantes para variables categóricas con la moda
- En el caso de haber mas columnas numericas con valores faltantes como el Latitude y Longitude se emplea KNN para imputar datos continuos
- Eliminar filas con valores faltantes y que no son beneficiosas para el análisis se eliminan del conjunto de datos.
- Las columnas con pocos valores faltantes, como Ward y Community Area, son imputadas con la moda, ya que estas variables suelen tener valores repetidos que representan zonas administrativas.
- Para las columnas categóricas como Location Description, creamos una nueva categoría para manejar los valores faltantes, preservando los datos sin necesidad de eliminarlos.

PROCESANDO LA DATA

02 - CODIFICACIÓN

```
df['Arrest'] = df['Arrest'].astype(int)
df['Domestic'] = df['Domestic'].astype(int)
```

Se van a cambiar el tipo de variables de Arrest y Domestic de categóricas a numericas para facilitar el uso de la data antes siendo un booleano a numérico

```
# Codificación One-Hot para las variables categóricas
df = pd.get_dummies(df, columns=['Location Description'], drop_first=True)
```

Es funcional puesto que One-Hot Encoding crea nuevas columnas binarias para cada categoría única en las variables categóricas, permitiendo que se usen en modelos que no pueden manejar directamente variables categóricas

```
# Crear cuadrantes de coordenadas geográficas (esto simplifica el análisis geográfico)
df['LatGroup'] = pd.cut(df['Latitude'], bins=5, labels=False)
df['LonGroup'] = pd.cut(df['Longitude'], bins=5, labels=False)
```

Contamos con Latitud y Longitud, se puede agrupar estas coordenadas en áreas más grandes, como distritos o zonas específicas, para facilitar el análisis geoespacial. Así se puede usar clustering para agrupar los puntos de coordenadas cercanas entre sí, facilitando la visualización y el análisis espacial.

PROCESANDO LA DATA

03 - ESCALADO

- Estandarización avanzada: Además de escalar variables como Latitude y Longitude, también sería interesante escalar todas las variables numéricas relevantes para normalizar las entradas al mismo rango que mejoraría para el uso con modelos predictivos.
- Normalización Min-Max: Si además se trataría de utilizar algoritmos basados en distancia como el Knn o clustering, la normalización Min-Max puede ser una mejor opción.

```
# Escalar variables numéricas con estandarización (media 0, desviación estándar 1)
scaler = StandardScaler()
numerical_cols = ['Latitude', 'Longitude', 'Beat', 'District', 'Ward', 'Community Areas']
df[numerical_cols] = scaler.fit_transform(df[numerical_cols])

# Si usas Min-Max para ciertos algoritmos (0 a 1):
# minmax_scaler = MinMaxScaler()
# df[numerical_cols] = minmax_scaler.fit_transform(df[numerical_cols])
```

#cantidad de valores nulos
df.isnull().sum()

	0
ID	0
Case Number	0
Date	0
Block	0
IUCR	0
...	...
Month	0
DayOfWeek	0
Hour	0
LatGroup	0
LonGroup	0

252 rows × 1 columns
dtype: int64

PROCESANDO LA DATA

04 - SELECCIÓN

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import SelectFromModel

# Asegurarnos de que solo queden columnas numéricas
df_numeric = df.select_dtypes(include=['float64', 'int64'])

# Preparamos las características (X) y la variable objetivo (y)
X = df_numeric.drop('Arrest', axis=1)
y = df_numeric['Arrest']

# Entrenar el modelo de Random Forest con optimizaciones para reducir tiempo de ejecución
rf = RandomForestClassifier(n_estimators=50, # Reducir el número de árboles
                           max_depth=10, # Limitar la profundidad de los árboles
                           n_jobs=-1, # Utilizar todos los núcleos de CPU
                           max_features='sqrt', # Considerar sqrt(n_features) en cada división
                           random_state=42) # Reproducibilidad

# Entrenar el modelo
rf.fit(X, y)

# Seleccionar características basadas en importancia
model = SelectFromModel(rf, prefit=True)
X_new = model.transform(X)

# Mostrar las características más importantes
importances = rf.feature_importances_
feature_names = X.columns
important_features = pd.Series(importances, index=feature_names).sort_values(ascending=False)
print(important_features.head(10)) # Las 10 características más importantes
```

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:458: UserWarning: X has feature names, but SelectFromModel does not. This will lead to errors in attribute access and may cause degraded performance in some models.
warnings.warn(

ID	0.223230
X Coordinate	0.096655
Longitude	0.087699
Latitude	0.079070
Y Coordinate	0.078998
Domestic	0.076673
Beat	0.067611
Police Districts	0.059736
Community Area	0.042057
District	0.031607

dtype: float64

Se busco trabajar con el método embebido puesto que abarca los beneficios tanto de los métodos envolventes como de los métodos de filtro, al incluir interacciones entre características pero también al mantener costos computacionales razonables.

PROCESANDO LA DATA

ING CARACTERÍSTICAS

```
# Crear nuevas columnas para día, mes y año a partir de la fecha
df['Date'] = pd.to_datetime(df['Date'])
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
df['DayOfWeek'] = df['Date'].dt.day_name()

# Crear interacciones entre columnas, por ejemplo entre tipo de crimen y distrito
df['Crime_District_Interaction'] = df['Primary Type'].astype(str) + "_" + df['District'].astype(str)
```

Generación del csv limpio

```
[ ] # Guardar el dataset limpio en un archivo CSV
df.to_csv('crimenes_cleaned.csv', index=False)
```

También sería interesante crear características que podrían ser útiles en la predicción de arrestos.

- Una idea es de agrupaciones temporales, creando una columna que agrupe los crímenes por día de la semana, mes, o incluso estaciones.
- Así también crear variables que representen interacciones entre múltiples características, como la combinación entre el tipo de crimen y el distrito.

01 - PREGUNTA

¿Cómo ha evolucionado la frecuencia de crímenes violentos en comparación con delitos no violentos en diferentes zonas geográficas entre 2019 y 2024?

02 - PREGUNTA

¿Cuál es la relación entre el tipo de crimen y la probabilidad de arresto, considerando factores como el lugar del crimen y si el crimen fue doméstico?

03 - PREGUNTA

¿Cuál es la relación entre el tipo de crimen y la estacionalidad (mes del año)?



04 - PREGUNTA

¿Cuáles son las zonas geográficas con mayor prevalencia de crímenes violentos vs. no violentos?

05 - PREGUNTA

¿Cuáles son las áreas con más crímenes recurrentes en ciertos días de la semana?



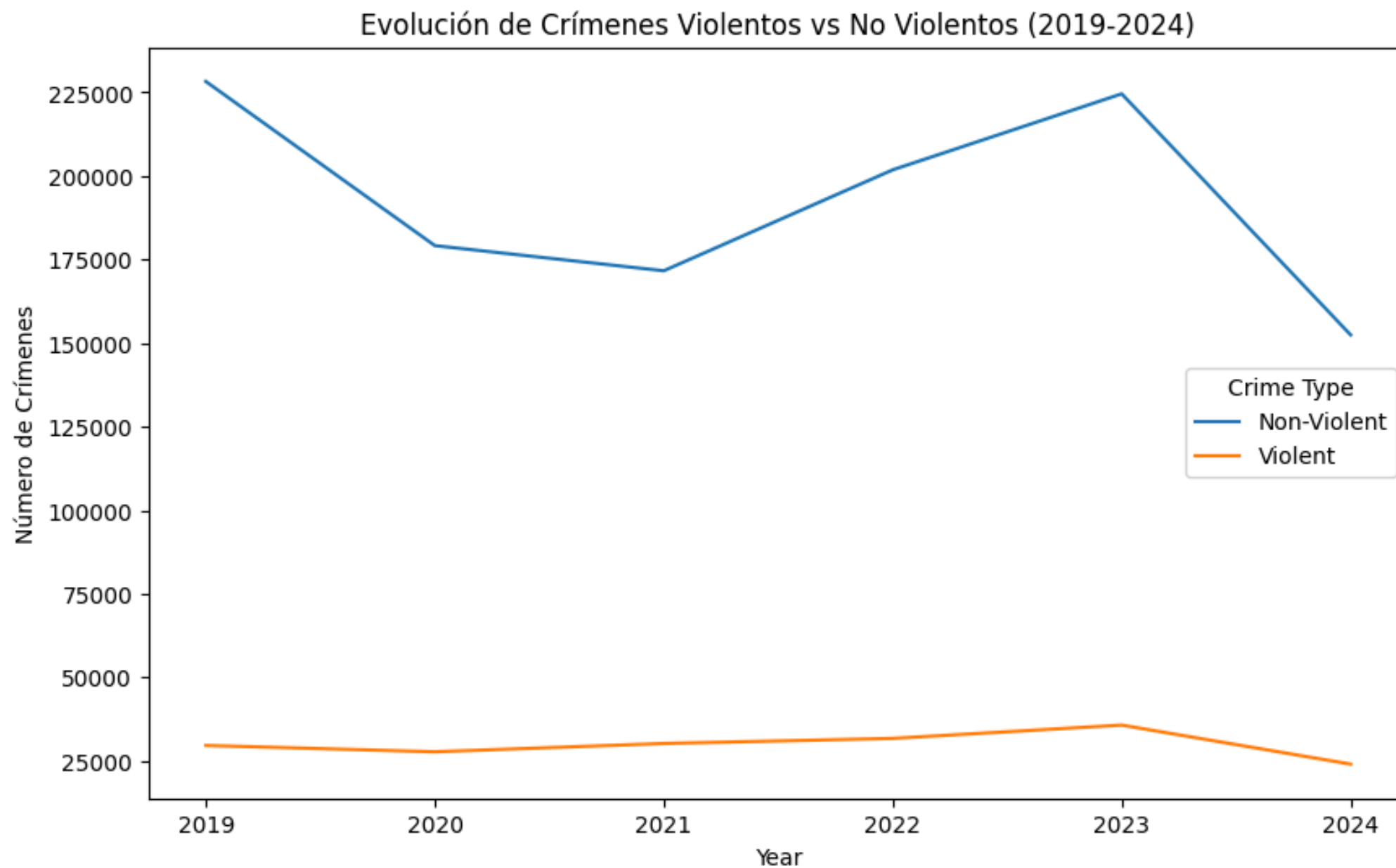
RELEVANCIA E HIPÓTESIS DE LAS PREGUNTAS

- Importante ver zonas mas afectadas puesto que podria haber mejorado la seguridad en ciertas zonas del Estado de Chicago pero pueden haber otras con carencias de seguridad
- Explora sobre qué tipos de **crímenes tienen una mayor o menor probabilidad de resultar en un arresto** puede ayudar a identificar las debilidades en la persecución y resolución de ciertos delitos.
- Los **patrones de crímenes pueden estar influidos por la estacionalidad**, con aumentos o disminuciones durante **ciertas épocas del año**, comprender este hecho podria ayudar significativamente a las autoridades a **planificar** mejor los efectivos y **recursos policiales** para su uso.
- Se busca identificar las **áreas más propensas a crímenes violentos** o no violentos puede ayudar a los responsables de la seguridad a dirigir mejor los recursos y mejorar la vigilancia en esas zonas.
- Bueno en esta pregunta se busca determinar si hay ciertos días con una mayor recurrencia de **crímenes en áreas específicas** similar a la pregunta 3 pero ahora por zonas para mejorar las estrategias de patrullaje.



PREGUNTA 1

Evolución de crímenes violentos y no violentos

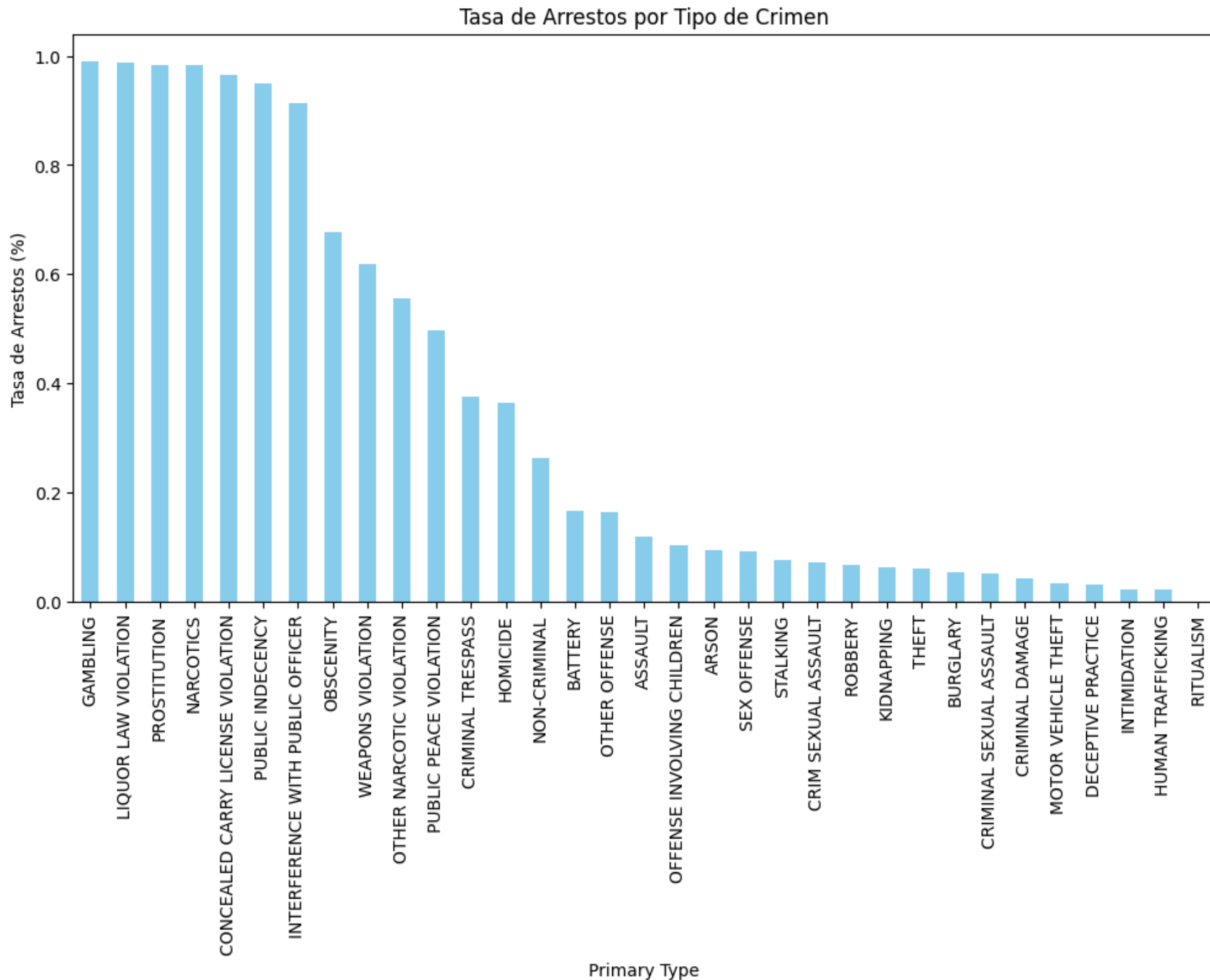


En la evolución de Crímenes Violentos vs No Violentos (2019-2024):

- Los crímenes no violentos aumentan hasta 2023, con una caída en 2024.
- Los crímenes violentos se mantienen relativamente estables durante el período.

PREGUNTA 2

Relación entre tipo de crimen y arresto



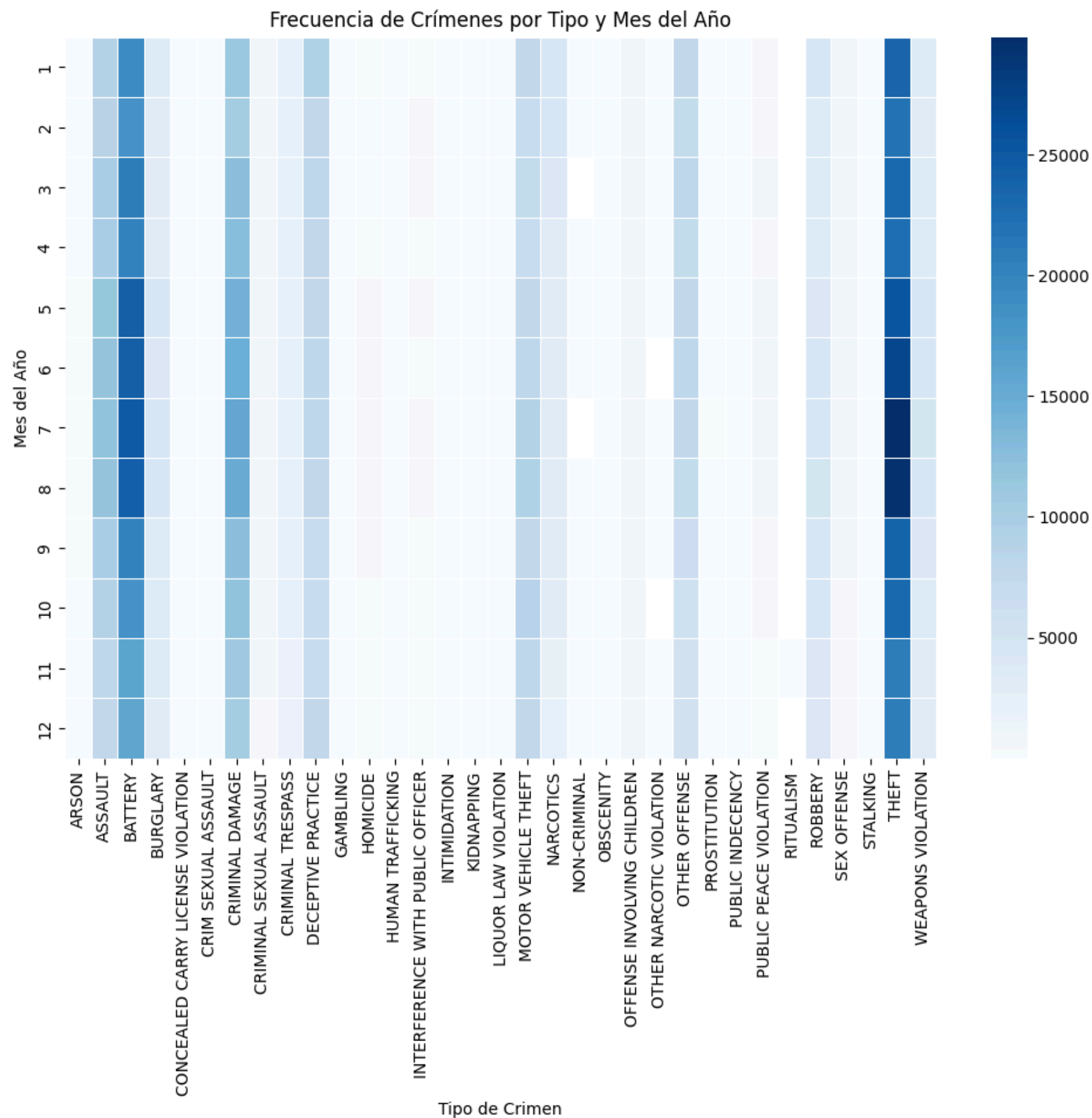
En la tasa de Arrestos por Tipo de Crimen:

- Juego ilegal y violaciones de armas tienen tasas de arresto cercanas al 100%.
- Robo y asalto sexual presentan tasas de arresto mucho más bajas.



PREGUNTA 3

Frecuencia de Crímenes por Tipo y Mes del Año



En la frecuencia de Crímenes por Tipo y Mes del Año:

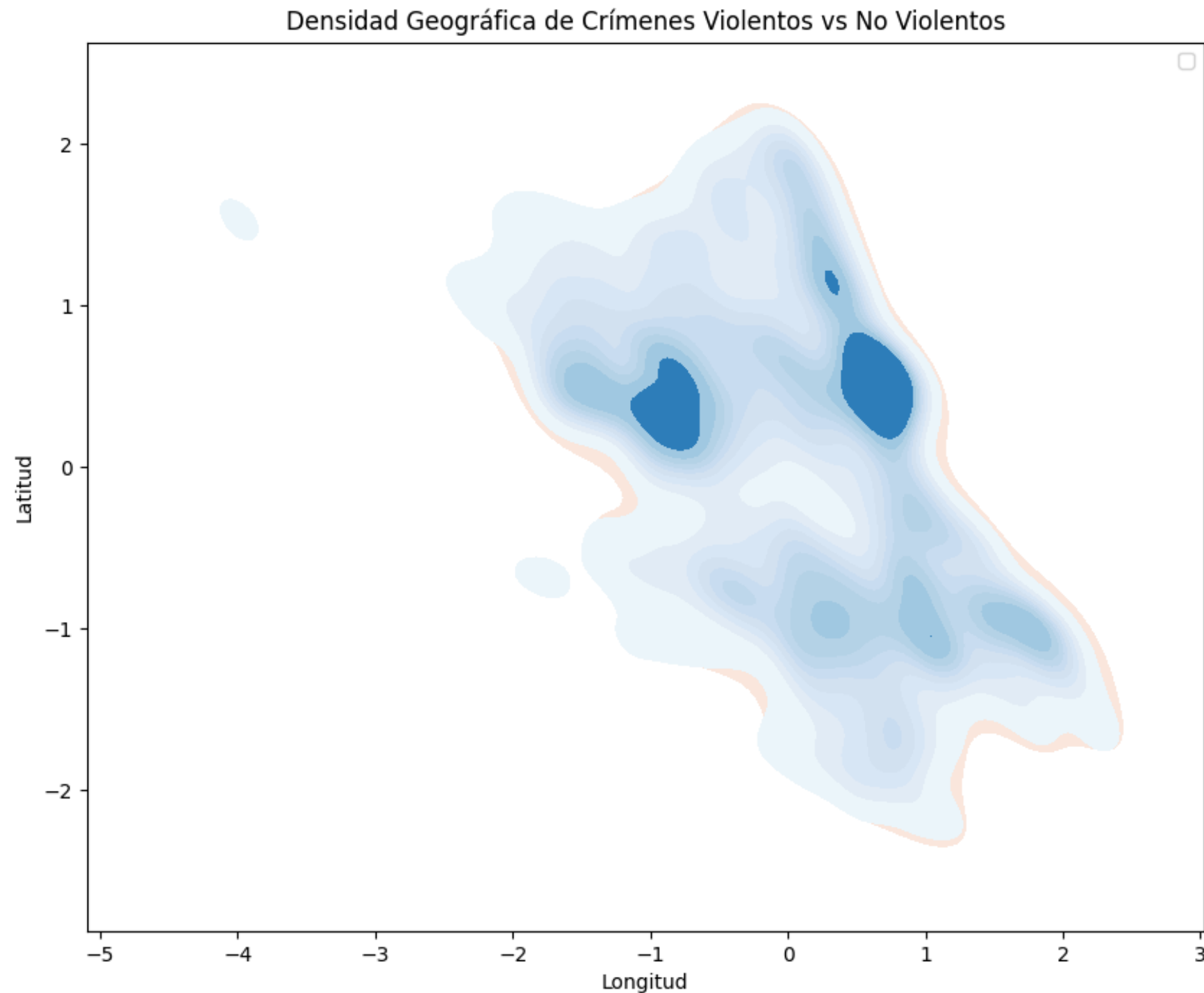
- El robo y las violaciones de armas son más frecuentes en diciembre y enero.
- La estacionalidad parece influir en el comportamiento criminal, especialmente en los meses de verano entre junio a agosto.

Primavera: Del 20 de marzo al 21 de junio.
Verano: Del 21 de junio al 22 de septiembre
Otoño: Del 22 de septiembre al 21 de diciembre.

Invierno: Del 21 de diciembre al 20 de marzo.

PREGUNTA 4

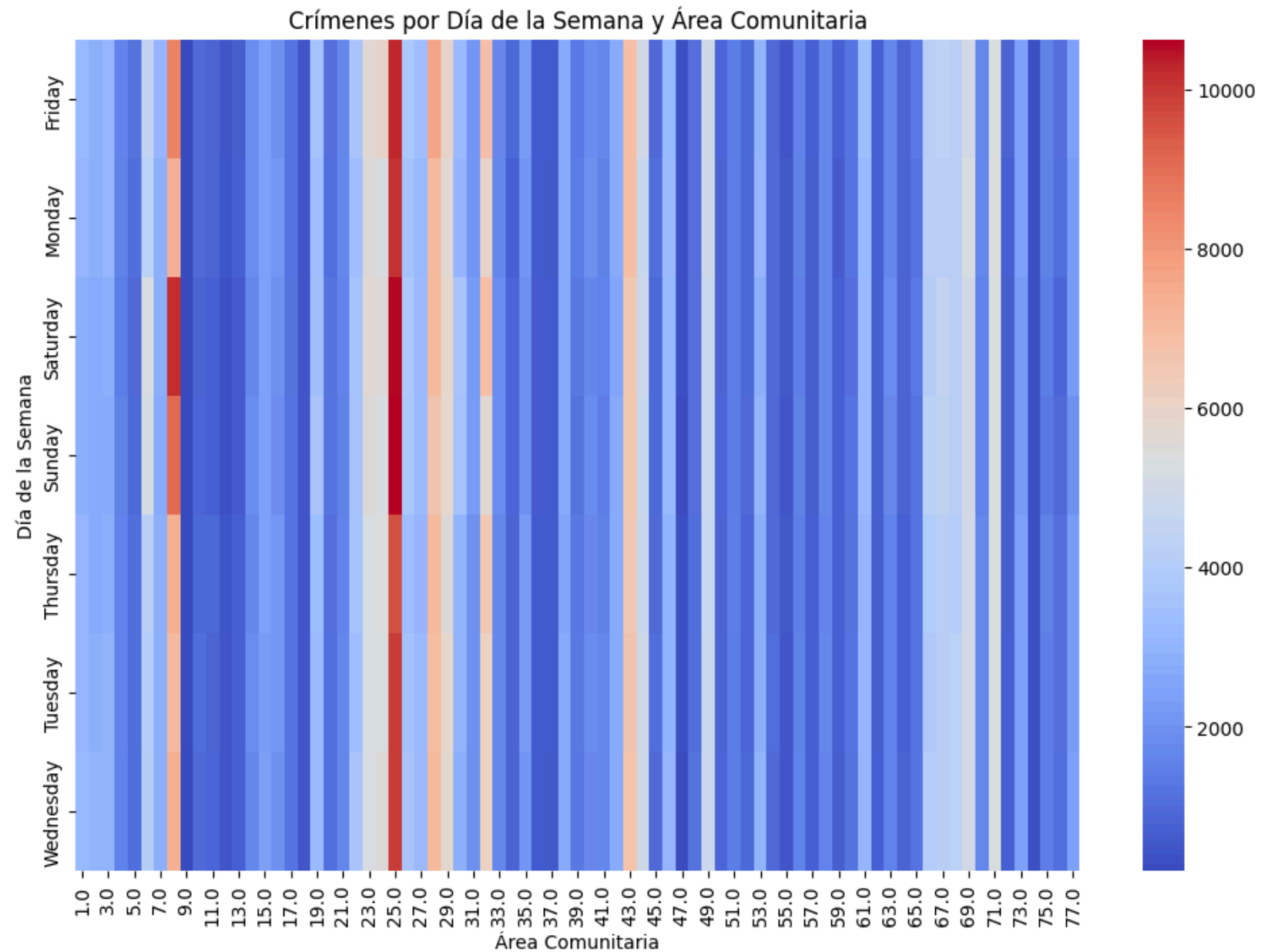
Zonas Geográficas con Mayor Prevalencia de Crímenes Violentos vs No Violentos



- Las áreas comunitarias 7.0, 25.0 y 22.0 tienen mayor actividad criminal los viernes y martes, sugiriendo la necesidad de medidas específicas en esos días.
- La concentración de crímenes varía entre vecindarios, lo que ayuda a priorizar recursos en días y áreas con mayor incidencia.
- Las áreas de mayor concentración de crímenes son claras, pero la diferenciación entre crímenes violentos y no violentos necesita mejor visualización para poder actuar de manera más efectiva.

PREGUNTA 5

Crímenes por Día de la Semana y Área Comunitaria:



En la frecuencia de Crímenes por Tipo y Mes del Año:

- El robo y las violaciones de armas son más frecuentes en diciembre y enero.
- La estacionalidad parece influir en el comportamiento criminal, especialmente en los meses de verano entre junio a agosto.



The background features four decorative geometric patterns in the corners. The top-left corner has a series of parallel diagonal lines in a light blue-grey color. The top-right corner contains a cluster of overlapping semi-circles in yellow, red, teal, and dark blue. The bottom-left corner also features a cluster of overlapping semi-circles in red, teal, and dark blue. The bottom-right corner has a series of parallel diagonal lines in a light blue-grey color, mirroring the top-left pattern.

GRACIAS