

## Descripción general

El repositorio Accidents\_usa proporciona un conjunto de datos completo de accidentes que han ocurrido en los Estados Unidos, con un enfoque en los accidentes de Nueva York. El conjunto de datos incluye varios atributos relacionados con los accidentes, como ubicación, fecha, hora y gravedad. Los datos se obtienen mediante el consumo de una API de terceros que proporciona datos de accidentes en tiempo real para Nueva York. Además, el repositorio utiliza Apache Airflow para administrar las tareas de extracción, transformación y carga (ETL), lo que permite la creación de un conjunto de datos fusionado que combina el conjunto de datos de accidentes de EE. UU. y el conjunto de datos de accidentes de Nueva York proporcionado por la API.

## Descripción del conjunto de datos

El conjunto de datos consta de [insertar número] filas y [insertar número] columnas. Las columnas incluyen:

- **Fecha** : La fecha de
- **Hora** : El
- **Ubicación** : La ubicación del accidente (ciudad, estado, código postal)
- **Gravedad** : La gravedad de la
- **Tipo de vehículo** : El tipo
- **Condiciones meteorológicas** : Las condiciones meteorológicas en el

## Integración API

El conjunto de datos se obtiene mediante el consumo de la API [insertar nombre de API], que proporciona datos de accidentes en tiempo real para Nueva York. La API se consulta periódicamente para actualizar el conjunto de datos con la información más reciente sobre accidentes.

## Proceso ETL

El repositorio utiliza Apache Airflow para administrar el proceso ETL, que implica:

- **Extraer**
- **Transformar**
- **Cargar**

## Conjunto de datos fusionados

El conjunto de datos fusionado combina el conjunto de datos de accidentes de EE. UU. y el conjunto de datos de accidentes de Nueva York proporcionado por API, lo que proporciona una visión integral de los accidentes en los Estados Unidos con un enfoque en Nueva York.

## Visualizaciones

El conjunto de datos fusionado se utiliza para crear varias visualizaciones, entre ellas:

- Frecuencia de accidentes por ubicación
- Gravedad del accidente
- Distribución del tipo de vehículo
- Impacto de las condiciones climáticas en la frecuencia de accidentes

## Fuentes de datos

Los datos provienen de la API [insertar nombre de API], que proporciona información precisa y confiable sobre accidentes en Nueva York y el conjunto de datos de accidentes de EE. UU.

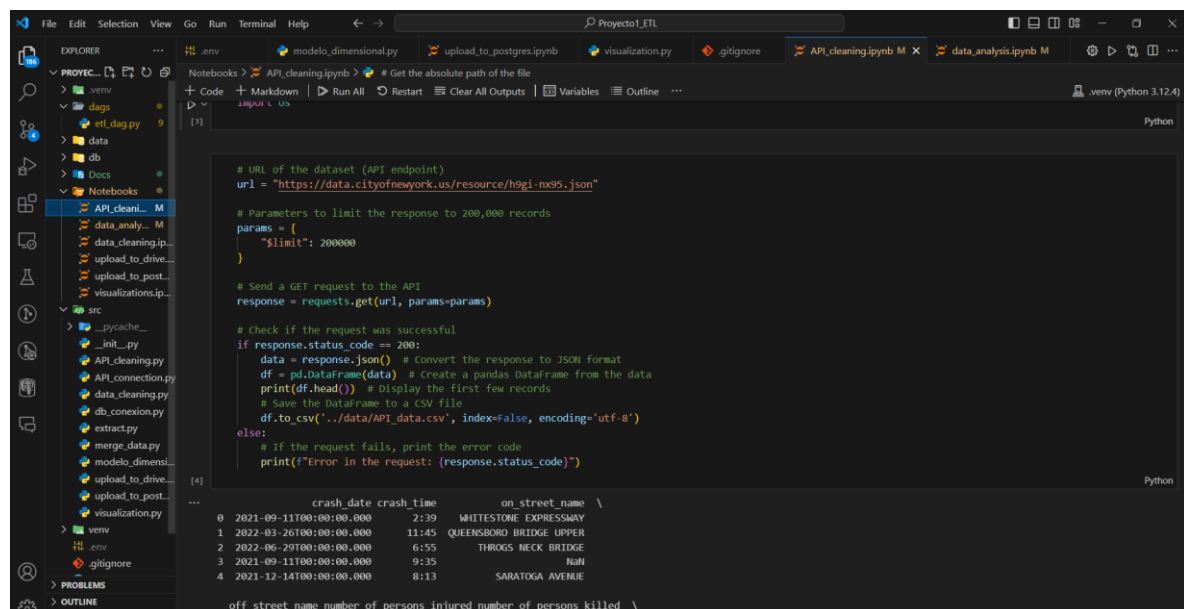
## Uso

El conjunto de datos se puede utilizar para diversos fines, tales como:

- Análisis de tendencias y patrones de accidentes
- Identificación de zonas de alto riesgo de accidentes en
- Desarrollo de la infraestructura en sitios clave donde ocurren mas accidentes

Aquí algunas evidencias

Esta fue la api que consumimos:



```
# URL of the dataset (API endpoint)
url = "https://data.cityofnewyork.us/resource/h9gi-nx95.json"

# Parameters to limit the response to 200,000 records
params = {
    "limit": 200000
}

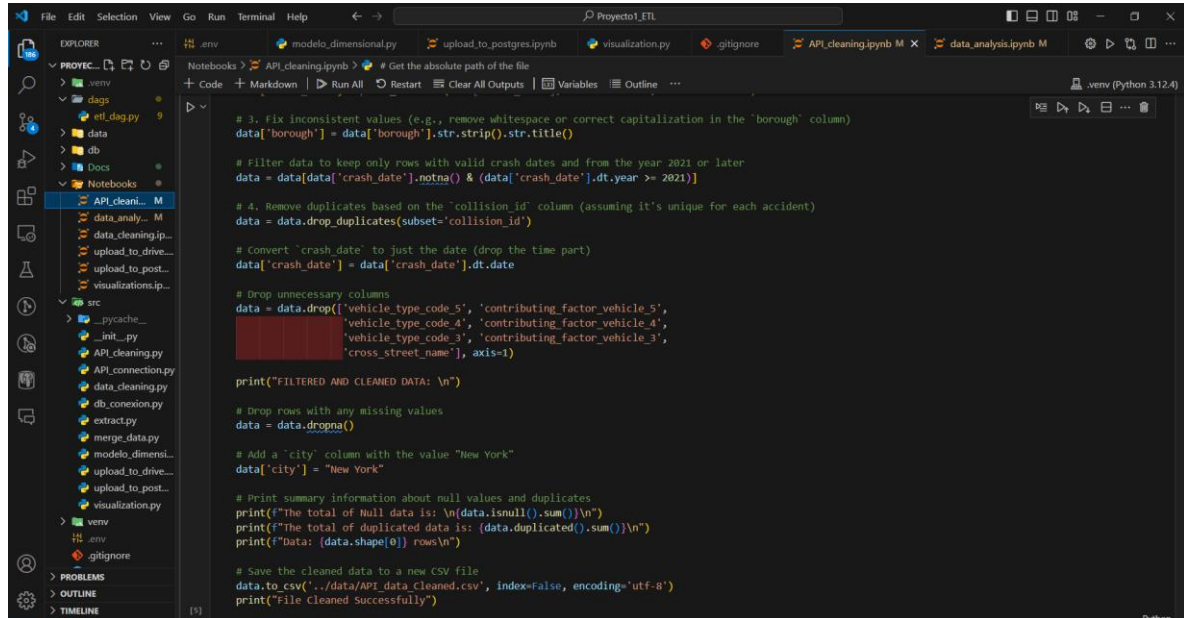
# Send a GET request to the API
response = requests.get(url, params=params)

# Check if the request was successful
if response.status_code == 200:
    data = response.json() # Convert the response to JSON format
    df = pd.DataFrame(data) # Create a pandas DataFrame from the data
    print(df.head()) # Display the first few records
    # Save the DataFrame to a CSV file
    df.to_csv('../data/API_data.csv', index=False, encoding='utf-8')
else:
    # If the request fails, print the error code
    print(f"Error in the request: {response.status_code}")
```

	crash_date	crash_time	on_street_name	
0	2021-09-11T00:00:00.000	2:39	WHITESTONE EXPRESSWAY	
1	2022-03-26T00:00:00.000	11:45	QUEENSBORO BRIDGE UPPER	
2	2022-06-29T00:00:00.000	6:55	THROGS NECK BRIDGE	
3	2021-09-11T00:00:00.000	9:35	NaN	
4	2021-12-14T00:00:00.000	8:13	SARATOGA AVENUE	

```
off_street_name number_of_persons_injured number_of_persons_killed \
```

Eliminamos columnas que no necesitábamos:



```
# 3. Fix inconsistent values (e.g., remove whitespace or correct capitalization in the 'borough' column)
data['borough'] = data['borough'].str.strip().str.title()

# Filter data to keep only rows with valid crash dates and from the year 2021 or later
data = data[data['crash_date'].notna() & (data['crash_date'].dt.year >= 2021)]

# 4. Remove duplicates based on the 'collision_id' column (assuming it's unique for each accident)
data = data.drop_duplicates(subset='collision_id')

# Convert 'crash_date' to just the date (drop the time part)
data['crash_date'] = data['crash_date'].dt.date

# Drop unnecessary columns
data = data.drop(['vehicle_type_code_5', 'contributing_factor_vehicle_5',
                 'vehicle_type_code_4', 'contributing_factor_vehicle_4',
                 'vehicle_type_code_3', 'contributing_factor_vehicle_3',
                 'cross_street_name'], axis=1)

print("FILTERED AND CLEANED DATA: \n")

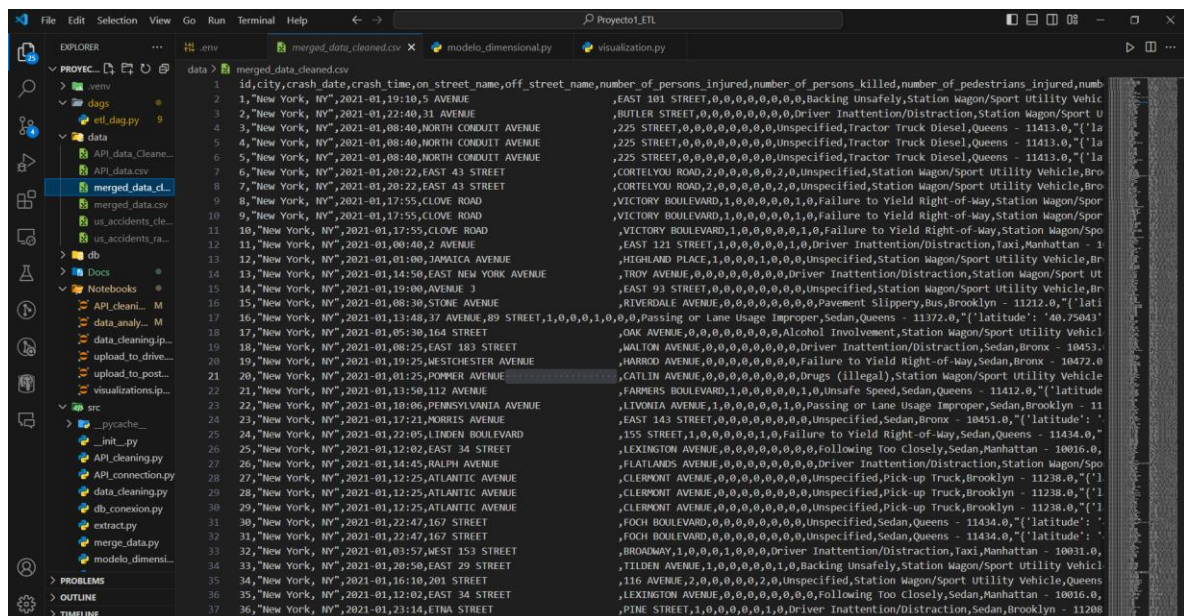
# Drop rows with any missing values
data = data.dropna()

# Add a 'city' column with the value "New York"
data['city'] = "New York"

# Print summary information about null values and duplicates
print(f"The total of Null data is: \n{data.isnull().sum()}\n")
print(f"The total of duplicated data is: {data.duplicated().sum()}\n")
print(f"Data: {data.shape[0]} rows\n")

# Save the cleaned data to a new CSV file
data.to_csv("../data/API_data_cleaned.csv", index=False, encoding='utf-8')
print("File Cleaned Successfully")
```

Nos quedamos con una tabla completamente larga, así se vea el csv del merge ya limpio



	merged_data_cleaned.csv
1	10,"New York, NY",2021-01,19:10,5 AVENUE ,EAST 101 STREET,0,0,0,0,0,0,0,Backing Unsafely,Station Wagon/Sport Utility Vehic
2	11,"New York, NY",2021-01,22:40,31 AVENUE ,BUTLER STREET,0,0,0,0,0,0,0,Driver Inattention/Distractio
3	12,"New York, NY",2021-01,08:40,NORTH CONDUIT AVENUE ,225 STREET,0,0,0,0,0,0,0,Unspecified,Tractor Truck Diesel,Queens - 11413.0,"[la
4	13,"New York, NY",2021-01,08:40,NORTH CONDUIT AVENUE ,225 STREET,0,0,0,0,0,0,0,Unspecified,Tractor Truck Diesel,Queens - 11413.0,"[la
5	14,"New York, NY",2021-01,08:40,NORTH CONDUIT AVENUE ,225 STREET,0,0,0,0,0,0,0,Unspecified,Tractor Truck Diesel,Queens - 11413.0,"[la
6	15,"New York, NY",2021-01,20:22,EAST 43 STREET ,CORTELYOU ROAD,2,0,0,0,0,0,2,Unspecified,Station Wagon/Sport Utility Vehicle,Bro
7	16,"New York, NY",2021-01,20:22,EAST 43 STREET ,CORTELYOU ROAD,2,0,0,0,0,0,2,Unspecified,Station Wagon/Sport Utility Vehicle,Bro
8	17,"New York, NY",2021-01,17:55,CLOVE ROAD ,VICTORY BOULEVARD,1,0,0,0,0,0,1,Failure to Yield Right-of-Way,Station Wagon/Spor
9	18,"New York, NY",2021-01,17:55,CLOVE ROAD ,VICTORY BOULEVARD,1,0,0,0,0,0,1,Failure to Yield Right-of-Way,Station Wagon/Spor
10	19,"New York, NY",2021-01,17:55,CLOVE ROAD ,VICTORY BOULEVARD,1,0,0,0,0,0,1,Failure to Yield Right-of-Way,Station Wagon/Spo
11	20,"New York, NY",2021-01,00:40,2 AVENUE ,HIGHLAND PLACE,1,0,0,0,1,0,0,Unspecified,Station Wagon/Sport Utility Vehicle,Br
12	21,"New York, NY",2021-01,01:00,JAMAICA AVENUE ,TROY AVENUE,0,0,0,0,0,0,0,Driver Inattention/Distractio
13	22,"New York, NY",2021-01,14:50,EAST NEW YORK AVENUE ,EAST 91 STREET,0,0,0,0,0,0,0,Unspecified,Station Wagon/Sport Utility Vehicle,Br
14	23,"New York, NY",2021-01,19:00,AVENUE J ,RIVERDALE AVENUE,0,0,0,0,0,0,0,Pavement Slippery,Bus,Brooklyn - 11212.0,"[lati
15	24,"New York, NY",2021-01,13:48,37 AVENUE ,89 STREET,1,0,0,0,1,0,0,Passing or Lane Usage Improper,Sedan,Queens - 11372.0,"[latitude: '40.75043"
16	25,"New York, NY",2021-01,05:30,164 STREET ,OAK AVENUE,0,0,0,0,0,0,0,Alcohol Involvement,Station Wagon/Sport Utility Vehicl
17	26,"New York, NY",2021-01,08:25,EAST 183 STREET ,WALTON AVENUE,0,0,0,0,0,0,0,Driver Inattention/Distractio
18	27,"New York, NY",2021-01,19:25,WESTCHESTER AVENUE ,HARROD AVENUE,0,0,0,0,0,0,0,Failure to Yield Right-of-Way,Sedan,Bronx - 10472.0
19	28,"New York, NY",2021-01,01:25,POMMER AVENUE ,CATLIN AVENUE,0,0,0,0,0,0,0,Drugs (illegal),Station Wagon/Sport Utility Vehicle
20	29,"New York, NY",2021-01,13:50,112 AVENUE ,FARMERS BOULEVARD,1,0,0,0,0,0,1,Unsafe Speed,Sedan,Queens - 11412.0,"[latitude
21	30,"New York, NY",2021-01,10:06,PENNSYLVANIA AVENUE ,LIVONIA AVENUE,1,0,0,0,0,0,1,Passing or Lane Usage Improper,Sedan,Brooklyn - 11
22	31,"New York, NY",2021-01,17:21,MORRIS AVENUE ,EAST 143 STREET,0,0,0,0,0,0,0,Unspecified,Sedan,Bronx - 10451.0,"[latitude: "
23	32,"New York, NY",2021-01,22:05,LINDEN BOULEVARD ,155 STREET,1,0,0,0,0,0,1,Failure to Yield Right-of-Way,Sedan,Queens - 11434.0,"
24	33,"New York, NY",2021-01,12:02,EAST 34 STREET ,LEXINGTON AVENUE,0,0,0,0,0,0,0,Following Too Closely,Sedan,Manhattan - 10016.0,
25	34,"New York, NY",2021-01,14:45,RALPH AVENUE ,FLATLANDS AVENUE,0,0,0,0,0,0,0,Driver Inattention/Distractio
26	35,"New York, NY",2021-01,12:25,ATLANTIC AVENUE ,CLERMONT AVENUE,0,0,0,0,0,0,0,Unspecified,Pick-up Truck,Brooklyn - 11238.0,"[l
27	36,"New York, NY",2021-01,12:25,ATLANTIC AVENUE ,CLERMONT AVENUE,0,0,0,0,0,0,0,Unspecified,Pick-up Truck,Brooklyn - 11238.0,"[l
28	37,"New York, NY",2021-01,22:47,167 STREET ,FOCH BOULEVARD,0,0,0,0,0,0,0,Unspecified,Sedan,Queens - 11434.0,"[latitude: "
29	38,"New York, NY",2021-01,22:47,167 STREET ,FOCH BOULEVARD,0,0,0,0,0,0,0,Unspecified,Sedan,Queens - 11434.0,"[latitude: "
30	39,"New York, NY",2021-01,03:57,WEST 153 STREET ,BROADWAY,1,0,0,0,1,0,0,Driver Inattention/Distractio
31	40,"New York, NY",2021-01,20:50,EAST 29 STREET ,TILDEN AVENUE,1,0,0,0,0,0,1,Backing Unsafely,Station Wagon/Sport Utility Vehicl
32	41,"New York, NY",2021-01,16:10,201 STREET ,116 AVENUE,2,0,0,0,0,0,2,Unspecified,Station Wagon/Spor Utility Vehicle,Queens
33	42,"New York, NY",2021-01,12:02,EAST 34 STREET ,LEXINGTON AVENUE,0,0,0,0,0,0,0,Following Too Closely,Sedan,Manhattan - 10016.0,
34	43,"New York, NY",2021-01,23:14,ETNA STREET ,PINE STREET,1,0,0,0,0,0,1,Driver Inattention/Distractio

Estos son algunos análisis interesantes donde el tipo de carro que mas tenia accidentes era el de tipo sedan:

```
[50] # 2. Distribución por tipo de vehículo involucrado
vehicle_distribution = merged_df['vehicle_type_code1'].value_counts()
print("\nDistribución por tipo de vehículo:")
print(vehicle_distribution)

..
Distribución por tipo de vehículo:
vehicle_type_code1
Sedan                22789
Station Wagon/Sport Utility Vehicle  16465
Taxi                 1414
Bus                 1122
Pick-up Truck       1022
...
TRACTOR              1
MTA bus              1
Vanette              1
REFG                1
Van Camper           1
Name: count, Length: 151, dtype: int64
```

Esta es la relación entre el clima y el numero de heridos

```
[53] # 3. Relación entre condiciones climáticas y gravedad del accidente
climate_vs_injuries = merged_df.groupby('weather_condition')['number_of_persons_injured'].sum()
print("\nRelación entre clima y número de personas heridas:")
print(climate_vs_injuries.sort_values(ascending=False))

...
Relación entre clima y número de personas heridas:
weather_condition
Fair                18005
Cloudy              6287
Light Rain         1848
Mostly Cloudy      1775
Partly Cloudy      1332
Heavy Rain         298
Rain               261
Fog                175
Light Snow         128
Haze               64
Snow              15
Name: number_of_persons_injured, dtype: int32
```

Esta comparación fue de los accidentes que ocurrían de noche y de día

```
# 4. Comparación entre accidentes de día y de noche
accidents_day_night = merged_df['sunrise_sunset'].value_counts()
print("\nComparación de accidentes entre día y noche:")
print(accidents_day_night)
```

[54]

...

Comparación de accidentes entre día y noche:

sunrise_sunset	
Day	32483
Night	15518

Name: count, dtype: int64

```
# 5. Factores contribuyentes más comunes en accidentes graves (con heridos)
factors_in_grave_accidents = merged_df[merged_df['number_of_persons_injured'] > 0]['contributing_factor_vehicle_1'].value_counts()
print("\nFactores contribuyentes más comunes en accidentes graves:")
print(factors_in_grave_accidents)
```

[55]

...

Factores contribuyentes más comunes en accidentes graves:

contributing_factor_vehicle_1	
Driver Inattention/Distracted	5940
Unspecified	3639
Failure to Yield Right-of-Way	2880
Traffic Control Disregarded	1757
Following Too Closely	1264
Unsafe Speed	837
Passing or Lane Usage Improper	774
Turning Improperly	634
Other Vehicular	502
Pedestrian/Bicyclist/Other Pedestrian Error/Confusion	434
Driver Inexperience	367
Unsafe Lane Changing	338

Así y muchos más análisis están en el notebook de data\_analysis

Así se ve el flujo de trabajo del airflow

