

High Volume FHV trip data (VFH_HV)

Each row represents a single trip in an FHV dispatched by one of NYC's licensed High Volume FHV bases. On August 14, 2018, Mayor de Blasio signed Local Law 149 of 2018, creating a new license category for TLC-licensed FHV businesses that currently dispatch or plan to dispatch more than 10,000 FHV trips in New York City per day under a single brand, trade, or operating name, referred to as High-Volume For-Hire Services (HVFHS). This law went into effect on Feb 1, 2019.

Los datos de viajes de FHVHV (High Volume For-Hire Vehicle) representan los registros de viajes en los vehículos de alquiler con conductor (FHV) en la ciudad de Nueva York que fueron despachados por bases FHV con licencia de alta capacidad. Estos datos se recopilan y publican por la Comisión de Taxis y Limusinas (TLC) de la ciudad de Nueva York.

```
In [1]: import pandas as pd
```

```
In [3]: df = pd.read_parquet('fhvhv_tripdata_2023-01.parquet')
```

```
In [15]: # contar el número de NaN en cada columna
nan_counts = df.isna().sum()

# imprimir los resultados
print(nan_counts)
```

```
hvfhs_license_num          0
company                     0
dispatching_base_num       0
originating_base_num      4891992
request_datetime            0
on_scene_datetime          4891992
pickup_datetime              0
dropoff_datetime              0
PULocationID                  0
DOLocationID                  0
trip_miles                      0
trip_time                        0
base_passenger_fare            0
tolls                            0
bcf                             0
sales_tax                         0
congestion_surcharge            0
airport_fee                       0
tips                             0
driver_pay                         0
shared_request_flag              0
shared_match_flag                0
access_a_ride_flag                0
wav_request_flag                 0
wav_match_flag                   0
dtype: int64
```

```
In [4]: df.columns
```

```
Out[4]: Index(['hvfhhs_license_num', 'dispatching_base_num', 'originating_base_num',
   'request_datetime', 'on_scene_datetime', 'pickup_datetime',
   'dropoff_datetime', 'PULocationID', 'DOLocationID', 'trip_miles',
   'trip_time', 'base_passenger_fare', 'tolls', 'bcf', 'sales_tax',
   'congestion_surcharge', 'airport_fee', 'tips', 'driver_pay',
   'shared_request_flag', 'shared_match_flag', 'access_a_ride_flag',
   'wav_request_flag', 'wav_match_flag'],
  dtype='object')
```

col_0 'hvfhhs_license_num'

The TLC license number of the HVFHS base or business

As of September 2019, the HVFHS licensees are the following:

- HV0002: Juno
- HV0003: Uber
- HV0004: Via
- HV0005: Lyft

```
In [5]: df['hvfhhs_license_num'].unique()
Out[5]: array(['HV0003', 'HV0005'], dtype=object)
```

col_1 'company'

Crear columna 'company' con valores 'Uber' y 'Lyft' según correspondan con valores en columna 'hvfhhs_license_num'

```
In [6]: import numpy as np

# crea una nueva columna con valores vacíos
df.insert(1, "company", "")

# usa np.where() para asignar los valores 'Uber' o 'Lyft' según el valor de hvfhhs_license_num
df['company'] = np.where(df['hvfhhs_license_num'] == 'HV0003', 'Uber', df['company'])
df['company'] = np.where(df['hvfhhs_license_num'] == 'HV0005', 'Lyft', df['company'])
```

```
In [8]: # contar cuántos registros son de 'Uber' y cuántos de 'Lyft'
counts = df['company'].value_counts()

# imprimir los resultados
print("Registros de Uber:", counts['Uber'])
print("Registros de Lyft:", counts['Lyft'])
```

Registros de Uber: 13580152
 Registros de Lyft: 4898879

col_2 'dispatching_base_num'

The TLC Base License Number of the base that dispatched the trip

```
In [9]: df['dispatching_base_num'].unique()
Out[9]: array(['B03404', 'B03406', 'B02870', 'B02764', 'B02872', 'B02765',
   'B02879', 'B02876', 'B02889', 'B02835', 'B02887', 'B02871',
   'B02869', 'B02866', 'B02875', 'B02865'], dtype=object)
```

col_3 'originating_base_num'

base number of the base that received the original trip request

```
In [10]: df['originating_base_num'].unique()
```

```
Out[10]: array(['B03404', None, 'B02870', 'B02764', 'B03406', 'B02872', 'B02765',
   'B02026', 'B02879', 'B02876', 'B02889', 'B02835', 'B02887',
   'B02871', 'B02869', 'B03153', 'B02866', 'B02875', 'B02865'],
  dtype=object)
```

col_4 'request_datetime'

date/time when passenger requested to be picked up

```
In [16]: print(df['request_datetime'].min())
print(df['request_datetime'].max())
```

```
2022-12-31 20:30:00
2023-02-01 00:15:00
```

col_5 'on_scene_datetime'

date/time when driver arrived at the pick-up location (Accessible Vehicles-only)

```
In [18]: print(df['on_scene_datetime'].min())
print(df['on_scene_datetime'].max())
```

```
2022-12-31 21:23:03
2023-01-31 23:59:53
```

col_6 'pickup_datetime'

The date and time of the trip pick-up

```
In [19]: print(df['pickup_datetime'].min())
print(df['pickup_datetime'].max())
```

```
2023-01-01 00:00:00
2023-01-31 23:59:59
```

col_7 'dropoff_datetime'

The date and time of the trip drop-off

```
In [21]: print(df['dropoff_datetime'].min())
print(df['dropoff_datetime'].max())
```

```
2023-01-01 00:02:27
2023-02-01 01:47:23
```

col_8 'PULocationID'

TLC Taxi Zone in which the trip began

```
In [24]: # Los valores para este campo son INT
df['PULocationID'].nunique()
```

```
Out[24]: 262
```

col_9 'DOLocationID'

TLC Taxi Zone in which the trip ended

```
In [27]: # Los valores para este campo son INT
df['DOLocationID'].nunique()
```

```
Out[27]: 261
```

col_10 'trip_miles'

total miles for passenger trip

```
In [28]: print(df['trip_miles'].min())
print(df['trip_miles'].max())
```

```
0.0
407.563
```

col_11 'trip_time'

total time in seconds for passenger trip

```
In [30]: print(df['trip_time'].min())
print(df['trip_time'].max())
# Al parecer hay valores atípicos como este máximo que representa 9.82 horas
```

```
0
35359
```

col_12 'base_passenger_fare'

base passenger fare before tolls, tips, taxes, and fees

```
In [35]: print(df['base_passenger_fare'].nunique())
print(df['base_passenger_fare'].min())
print(df['base_passenger_fare'].max())
```

```
# ¿valores negativos? ¿Es posible?
```

```
# contar el número de valores iguales y menores a cero en la columna "base_passenger_fare"
negative_count = (df['base_passenger_fare'] < 0).sum()
zero_count = (df['base_passenger_fare'] == 0).sum()
```

```
# imprimir los resultados
```

```
print("valores menores a cero:", negative_count)
print("valores iguales a cero:", zero_count)
```

```
28613
-146.34
1455.12
valores menores a cero: 13440
valores iguales a cero: 98
```

col_13 'tolls'

total amount of all tolls paid in trip

Los "tolls" son peajes o tarifas que se pagan para usar ciertas carreteras, puentes o túneles. En el contexto de un viaje en taxi, los "tolls" se refieren a los peajes que se pagan durante el viaje, que son costos adicionales que se suman a la tarifa de viaje base.

```
In [36]: print(df['tolls'].min())
print(df['tolls'].max())
```

```
0.0
184.37
```

col_14 'bcf'

total amount collected in trip for Black Car Fund

El "Black Car Fund" es un fondo de compensación y seguro para los conductores de vehículos de alquiler de Nueva York que están afiliados a empresas de transporte de pasajeros con licencia. El fondo fue creado para proporcionar una red de seguridad financiera para los conductores, y está financiado por un cargo de 2.5% sobre el total de los ingresos brutos que los conductores ganan mientras trabajan para las empresas de transporte participantes.

En el contexto de un viaje en taxi, "total amount collected in trip for Black Car Fund" se refiere a la cantidad total de dinero que se ha recolectado durante un viaje en taxi para ser destinado al Black Car Fund. El monto se calcula como un porcentaje del ingreso bruto del conductor por el viaje.

El monto recolectado para el Black Car Fund se informa por separado de la tarifa de viaje base y otros cargos adicionales, y se detalla en el recibo que se le entrega al pasajero después del viaje.

```
In [37]: print(df['bcf'].min())
print(df['bcf'].max())
```

```
0.0
64.71
```

col_15 'sales_tax'

total amount collected in trip for NYS sales tax

El "total amount collected in trip for NYS sales tax" se refiere a la cantidad de impuestos sobre las ventas del Estado de Nueva York que se recolecta durante un viaje en taxi y se paga al gobierno estatal. En Nueva York, se aplica un impuesto estatal del 4% a las tarifas de taxi y un impuesto adicional del 0.375% para financiar el sistema de transporte público.

La cantidad total de impuestos sobre las ventas que se cobran durante un viaje en taxi se calcula como un porcentaje de la tarifa de viaje base y otros cargos adicionales, como los peajes. El monto total del impuesto se informa por separado en el recibo que se entrega al pasajero después del viaje.

```
In [38]: print(df['sales_tax'].min())
print(df['sales_tax'].max())
```

```
0.0
120.91
```

col_16 'congestion_surcharge'

total amount collected in trip for NYS congestion surcharge

El "total amount collected in trip for NYS congestion surcharge" se refiere a la cantidad de la tarifa de congestión que se recolecta durante un viaje en taxi en la ciudad de Nueva York y se paga al gobierno estatal. La tarifa de congestión fue introducida por el gobierno estatal en 2019 con el objetivo de reducir la congestión del tráfico en la ciudad de Nueva York y mejorar la calidad del aire.

La tarifa de congestión se aplica a todos los viajes en taxi y vehículos de transporte de pasajeros con licencia que pasan por el distrito de congestión en Manhattan, que es una zona del centro de la ciudad definida por el gobierno estatal. La tarifa se cobra en la parte superior de la tarifa de viaje base y otros cargos adicionales, y su monto depende del tipo de vehículo y la hora del día.

```
In [39]: print(df['congestion_surcharge'].min())
print(df['congestion_surcharge'].max())
```

```
0.0
8.25
```

col_17 'airport_fee'

\$2.50 for both drop off and pick up at LaGuardia, Newark, and John F. Kennedy airports

El "airport fee" es una tarifa adicional que se cobra a los pasajeros de taxis y vehículos de transporte de pasajeros con licencia que se recogen o dejan en los aeropuertos de LaGuardia, Newark y John F. Kennedy en la ciudad de Nueva York. Esta tarifa se cobra por separado de la tarifa de viaje base y otros cargos adicionales, y su monto es de \$2.50 por cada recogida o entrega en cualquiera de estos aeropuertos.

La tarifa del aeropuerto se utiliza para financiar la construcción y mantenimiento de instalaciones y servicios en los aeropuertos de la ciudad de Nueva York, como estacionamientos y terminales de pasajeros. La tarifa es obligatoria para todos los taxis y vehículos de transporte de pasajeros con licencia que operan en la ciudad de Nueva York y que prestan servicios de recogida o entrega en los aeropuertos cubiertos por la tarifa.

```
In [42]: print(df['airport_fee'].min())
print(df['airport_fee'].max())
print(df['airport_fee'].unique())

# ¿Cómo puede ser que un viaje tenga un cargo de 6.9 por concepto de 'airport_fee'?

0.0
6.9
[0.    2.5  1.25 5.    6.9  4.5  4.4  1.    0.5 ]
```

col_18 'tips'

total amount of tips received from passenger

```
In [43]: print(df['tips'].min())
print(df['tips'].max())

# ¿180 dólares de propina?

0.0
180.53
```

col_19 'driver_pay'

total driver pay (not including tolls or tips and net of commission, surcharges, or taxes)

El "total driver pay (not including tolls or tips and net of commission, surcharges, or taxes)" se refiere al monto total que el conductor de un taxi o vehículo de transporte de pasajeros con licencia gana por un viaje, después de deducir los costos asociados con la comisión, los recargos, los impuestos y cualquier otro costo que la empresa pueda deducir. Esta cantidad no incluye los peajes o propinas que el conductor pueda recibir durante el viaje.

Para calcular el "total driver pay", primero se suman todos los ingresos brutos que el conductor gana durante el viaje, incluyendo la tarifa de viaje base, los recargos y cualquier otra tarifa adicional que se aplique. Luego, se deducen de esta cantidad todos los costos que la empresa de transporte pueda deducir, como comisiones, recargos y impuestos.

Es posible que sea un valor negativo. Lo que significa que el conductor en realidad habrá perdido dinero durante el viaje. Esto es poco común, pero puede ocurrir en ciertas situaciones, como cuando el conductor viaja largas distancias para recoger a un pasajero, solo para descubrir que el pasajero canceló el viaje después de que el conductor llegó.

```
In [44]: print(df['driver_pay'].min())
print(df['driver_pay'].max())
```

-102.15
1285.04

col_20 'shared_request_flag'

Did the passenger agree to a shared/pooled ride, regardless of whether they were matched? (Y/N)

"shared_request_flag" se refiere a un indicador en los registros de datos de un servicio de transporte compartido, como un servicio de viajes compartidos en automóvil, que indica si el pasajero solicitó un viaje compartido/pool, independientemente de si se encontró una coincidencia con otros pasajeros.

Se establece en "Y" si el pasajero acordó compartir el viaje con otros pasajeros y en "N" si no lo hizo.

```
In [45]: # contar cuántos registros en 'shared_request_flag' son de 'Y' y cuántos 'N'
counts = df['shared_request_flag'].value_counts()

# imprimir los resultados
print("Y:", counts['Y'])
print("N:", counts['N'])
```

Y: 496066
N: 17982965

col_21 'shared_match_flag'

Did the passenger share the vehicle with another passenger who booked separately at any point during the trip? (Y/N)

"shared_match_flag" es un indicador que indica si un pasajero compartió un vehículo con otro pasajero que reservó por separado en algún momento durante el viaje en un servicio de transporte compartido. Se establece en "Y" si el pasajero compartió el vehículo con otro pasajero durante el viaje y en "N" si no lo hizo.

```
In [46]: # contar cuántos registros en 'shared_match_flag' son de 'Y' y cuántos 'N'
counts = df['shared_match_flag'].value_counts()

# imprimir los resultados
print("Y:", counts['Y'])
print("N:", counts['N'])
```

Y: 182366
N: 18296665

col_22 'access_a_ride_flag'

Was the trip administered on behalf of the Metropolitan Transportation Authority (MTA)? (Y/N)

"access_a_ride_flag" se refiere a un indicador en los registros de datos de servicios de transporte en la ciudad de Nueva York que indica si el viaje fue administrado en nombre de la Autoridad

Metropolitana de Transporte (MTA) de la ciudad de Nueva York. La MTA es la agencia que gestiona el transporte público en la ciudad de Nueva York y sus alrededores, y ofrece servicios de transporte para personas con discapacidades bajo el nombre de Access-A-Ride.

Este indicador se establece en "Y" si el viaje fue reservado y pagado por la MTA, o si fue coordinado a través del programa Access-A-Ride de la MTA. Esto significa que el viaje fue programado para un pasajero que cumple con los requisitos de elegibilidad para el programa Access-A-Ride y que es elegible para recibir servicios de transporte adaptados para personas con discapacidades.

```
In [50]: print(df['access_a_ride_flag'].unique())
# contar cuántos registros en 'access_a_ride_flag' son de ' ' y cuántos 'N'
print(df['access_a_ride_flag'].value_counts())
[' ' 'N']
13580152
N      4898879
Name: access_a_ride_flag, dtype: int64
```

col_23 'wav_request_flag'

Did the passenger request a wheelchair-accessible vehicle (WAV)? (Y/N)

```
In [51]: print(df['wav_request_flag'].unique())
# contar cuántos registros en 'wav_request_flag' de cada tipo
print(df['wav_request_flag'].value_counts())
['N' 'Y']
N      18451056
Y      27975
Name: wav_request_flag, dtype: int64
```

'wav_match_flag'

Did the trip occur in a wheelchair-accessible vehicle (WAV)? (Y/N)

```
In [52]: print(df['wav_match_flag'].unique())
# contar cuántos registros en 'wav_match_flag' de cada tipo
print(df['wav_match_flag'].value_counts())
['N' 'Y']
N      17262300
Y      1216731
Name: wav_match_flag, dtype: int64
```