Taxi Fare Data Generation: Explanation of Variable Choices

This document outlines the rationale behind each variable, multiplier, and assumption used in generating synthetic taxi ride data for Barcelona in 2024. The goal is to ensure the dataset is both realistic and ML-worthy while reflecting the actual operational behavior of the city's taxi system.

1. Tariffs and Constants

Barcelona taxi tariffs which are fixed by the authorities, so in the generation of our data we will take these values as they are.

```
BASE_FARE = 2.75                  # Fixed base fare

PRICE_PER_KM_DAY = 1.32           # Price per km (daytime 08:00 - 20:00)

PRICE_PER_KM_NIGHT = 1.62         # Price per km (nighttime 20:00 - 08:00)

AIRPORT_SURCHARGE = 4.50          # Fixed

HOLIDAY_SURCHARGE = 3.50          # Fixed

PASSENGERS_SURCHARGE = 4.50       # Passenger count more than 4 (from 5-8)

MIN_FARE = 7                      # Minimum total

MIN_FARE_AIRPORT = 21             # Minimum airport ride
```

2. Base Duration Per KM

- **`BASE_DURATION_KM = 2.5` minutes per km**

According to the TomTom Traffic Index for 2024, the average time it takes to travel 1 km in Barcelona ranges between **3 minutes 30 seconds and 3 minutes 40 seconds**.

In the rush hours the time per km goes up to 4 min and more and can come down to 2 min on the midnight hours. We can see this below in the captures taken directly from https://www.tomtom.com/traffic-index/barcelona-traffic/

Rush hour

Morning

Time taken to travel 1 km

3 min 31 sec

17.0 km/h

Average speed

45%

Congestion level

Time taken to travel 1 km

3 min 41 sec

16.2 km/h

Average speed

47%

Congestion level

Rush hour

Evening

How much extra time did we spend driving in rush hours over the year?

8 hours

↑ 32 min more than in 2023

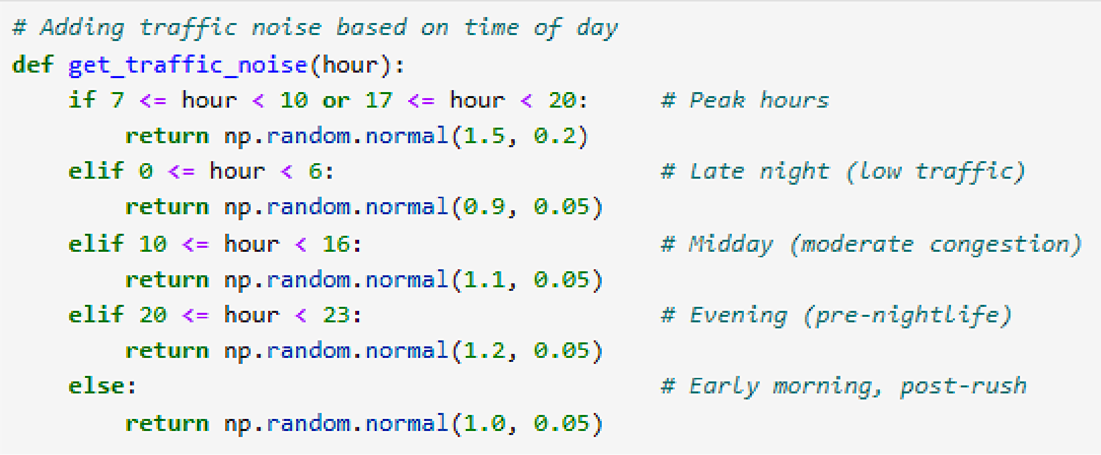
	Sun	Mon	Tue	Wed	Thu	Fri	Sat
12:00 AM	3 min	3 min	3 min	3 min	3 min	3 min	3 min
	3 min	3 min	3 min	3 min	3 min	3 min	3 min
02:00 AM	3 min	3 min	3 min	3 min	3 min	3 min	3 min
	3 min	3 min	3 min	3 min	3 min	3 min	3 min
04:00 AM	3 min	2 min	3 min	3 min	3 min	3 min	3 min
	2 min	2 min	2 min	2 min	2 min	2 min	2 min
06:00 AM	2 min	2 min	2 min	2 min	2 min	2 min	2 min
	2 min	3 min	3 min	3 min	3 min	3 min	2 min
08:00 AM	2 min	4 min	4 min	4 min	4 min	3 min	2 min
	2 min	3 min	3 min	3 min	3 min	3 min	3 min
10:00 AM	3 min	3 min	3 min	3 min	3 min	3 min	3 min
	3 min	3 min	3 min	3 min	3 min	3 min	3 min
12:00 PM	3 min	3 min	3 min	3 min	4 min	4 min	3 min
	3 min	3 min	3 min	3 min	4 min	4 min	3 min
02:00 PM	3 min	3 min	3 min	3 min	3 min	4 min	3 min
	3 min	3 min	3 min	3 min	3 min	4 min	3 min
04:00 PM	3 min	3 min	3 min	3 min	3 min	4 min	3 min
	3 min	3 min	4 min	4 min	4 min	4 min	3 min
06:00 PM	3 min	4 min	4 min	4 min	4 min	4 min	3 min
	3 min	3 min	3 min	4 min	4 min	3 min	3 min
08:00 PM	3 min	3 min	3 min	3 min	3 min	3 min	3 min
	3 min	3 min	3 min	3 min	3 min	3 min	3 min
10:00 PM	3 min	3 min	3 min	3 min	3 min	3 min	3 min
	3 min	3 min	3 min	3 min	3 min	3 min	3 min

However, taxis often benefit from **dedicated lanes and priority routing**, which means their average travel time per km is typically lower.

To reflect this more realistic taxi-specific behavior, we use a base duration of **2.8 minutes per km**, assuming average traffic flow for taxis in central urban areas.

We can’t just only use this value for all of the hours, we need to adjust it based on different times of the hour to reflect the reality.
That’s why I created the following function which adjusts the travel time based on the hour of the day, it takes in account rush hours and normal hours as well.

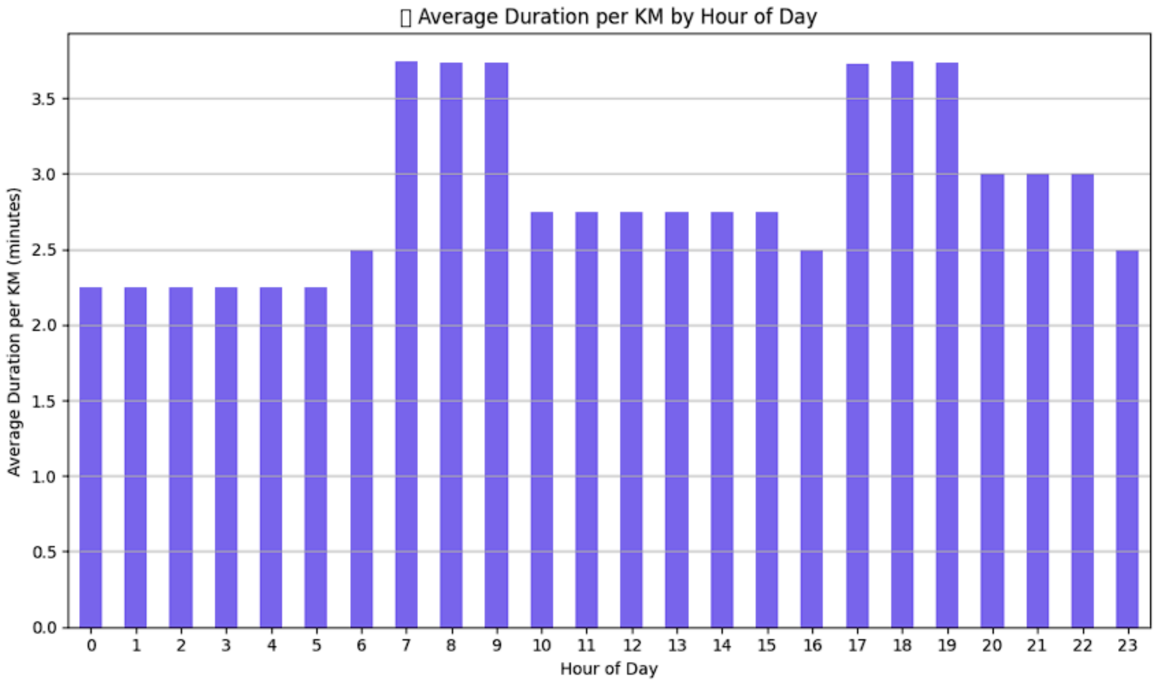
Function to simulate traffic-induced variation in ride durations:



```python
# Adding traffic noise based on time of day
def get_traffic_noise(hour):
    if 7 <= hour < 10 or 17 <= hour < 20:     # Peak hours
        return np.random.normal(1.5, 0.2)
    elif 0 <= hour < 6:                       # Late night (low traffic)
        return np.random.normal(0.9, 0.05)
    elif 10 <= hour < 16:                     # Midday (moderate congestion)
        return np.random.normal(1.1, 0.05)
    elif 20 <= hour < 23:                     # Evening (pre-nightlife)
        return np.random.normal(1.2, 0.05)
    else:                                     # Early morning, post-rush
        return np.random.normal(1.0, 0.05)
```

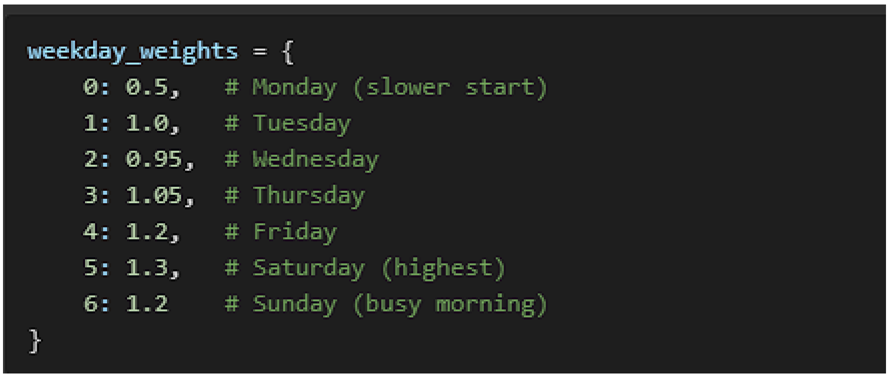
This adds realistic, time-based variation in ride duration, making the data less deterministic and more ML-friendly.

So by making this change we get the following duration per km and average by hour

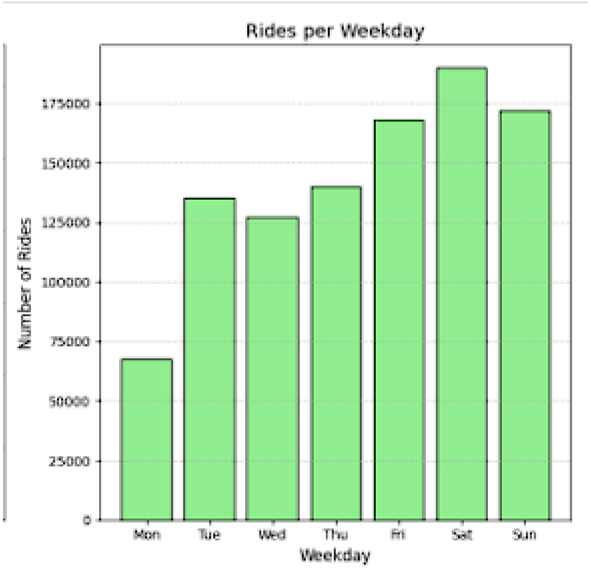


3. Weekday Weights

Used to simulate realistic weekly demand variation:

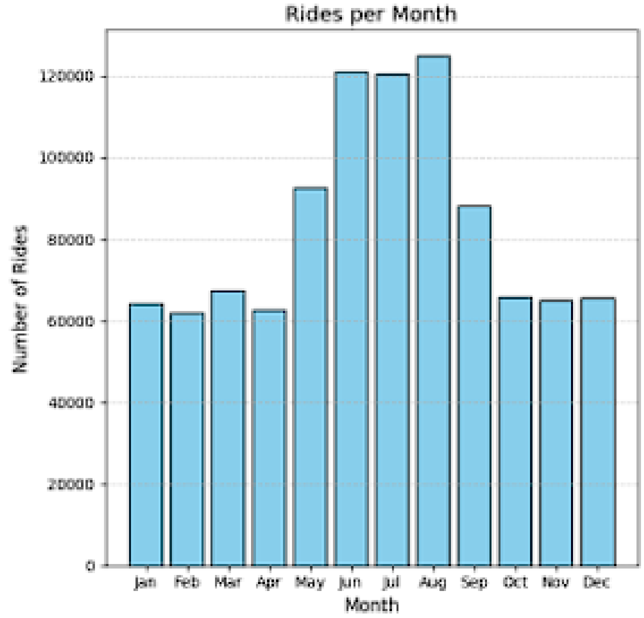


```
weekday_weights = {
    0: 0.5,    # Monday (slower start)
    1: 1.0,    # Tuesday
    2: 0.95,   # Wednesday
    3: 1.05,   # Thursday
    4: 1.2,    # Friday
    5: 1.3,    # Saturday (highest)
    6: 1.2     # Sunday (busy morning)
}
```



These weights reflect general commuting behavior, social activity, and late-night travel patterns.

4. Seasonal Weights



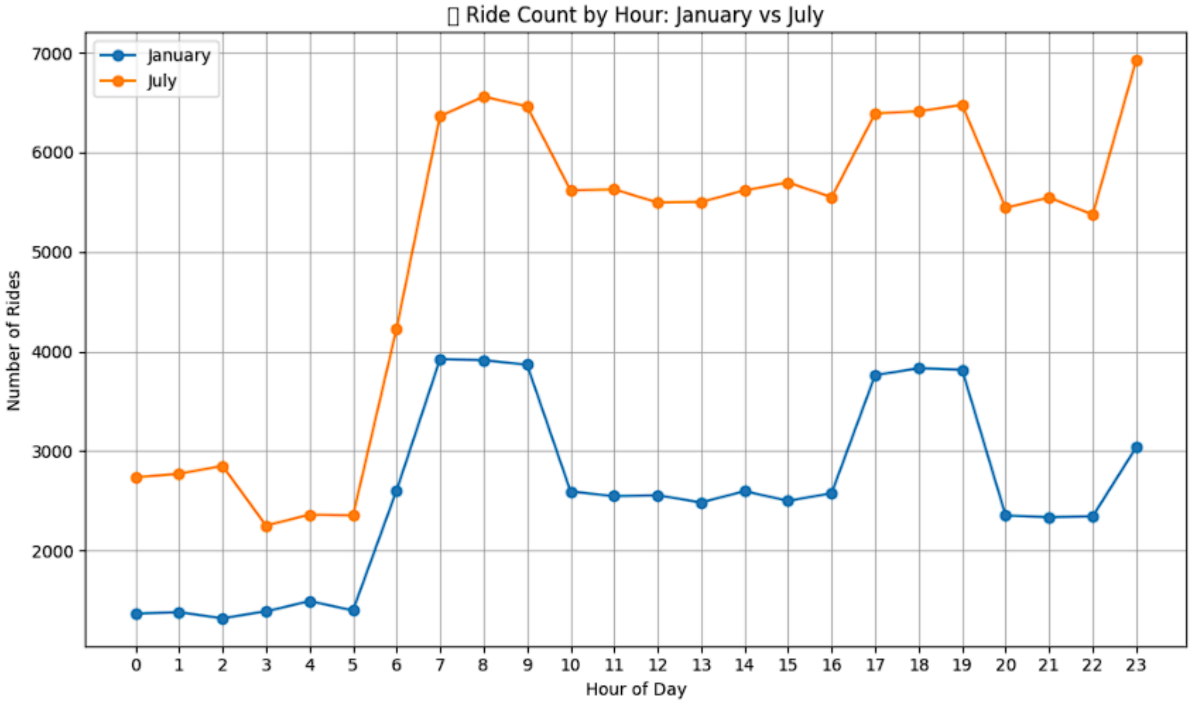
- **Peak season (June, July, August):** `1.6x`

Reflects increased tourism and leisure activity.

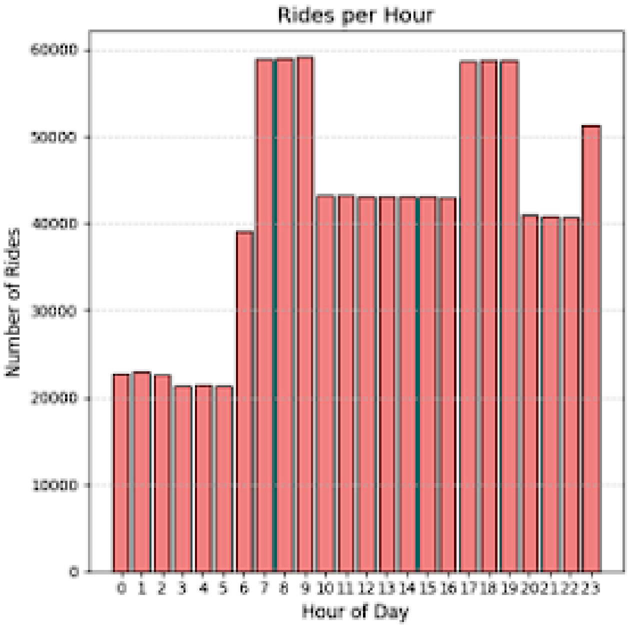
- **Shoulder season (May, September):** `1.4x`

Still moderately high due to good weather and partial tourism.

Below is the comparison between two months: January (off-season) vs July (peak-season)



5. Hourly Demand Patterns



- **Commuting hours (7-10 AM, 5-8 PM): `1.5x`**

- **Late night (0-6 AM): `0.5x`**

- **Evening leisure (8-11 PM): `0.8x`**

This mirrors real-world behavior: heavy demand in rush hours, lower during early mornings, and a slight rise in the evening.

6. Special Nightlife Boosts

- **Friday after 20:00: `1.3x`**

- **Saturday after 20:00 & Sunday early morning: `1.55x`**

These values account for nightlife-related demand surges on weekends.

8. Airport Ride Probability

- **20% of rides are airport rides**

This reflects a rough estimate based on real-world proportions of airport pickups in major cities, accounting for both tourists and early morning/late night rides.

9. Global Randomness Factor

- `np.random.normal(1.0, 0.05)`

Applied at the end of each weight calculation to introduce controlled randomness and prevent overly deterministic distributions.

This structured approach ensures that the generated dataset is a strong proxy for real-world taxi activity, enabling realistic ML training and analysis.