Function: `get_efficiency_level`

Purpose:

The `get_efficiency_level` function calculates the efficiency level of a motor based on given speed and torque values by checking which polygon (formed by data points representing efficiency levels) contains the point defined by these values. The efficiency levels are nested within each other, and the function returns the highest efficiency level that contains the point, corresponding to the innermost polygon. Finding the innermost polygon is essential as many of the polygon areas are shared, so the innermost polygon corresponds to the correct efficiency level.

Parameters:

- **`speed`:** The motor speed value to evaluate (RPM).
- **`torque`:** The motor torque value to evaluate (Nm).
- **`csv_file_path`:** The file path to the CSV containing efficiency data points. Defaults to `'data/wpd_datasets.csv'`.

CSV File Structure:

- The CSV file is expected to have multiple columns, with each pair of columns corresponding to a specific efficiency level (e.g., "86-90", "90-94", etc.).
- The values of “86-90” and “90-94” were averaged to 88 and 92 respectively.
- Each pair of columns represents x and y coordinates (speed and torque), which are used to create polygons for each efficiency level.
  - Points were overlaid onto the official Emrax 228 efficiency map, therefore creating polygons from these points is essentially recreating the efficiency map.

Workflow:

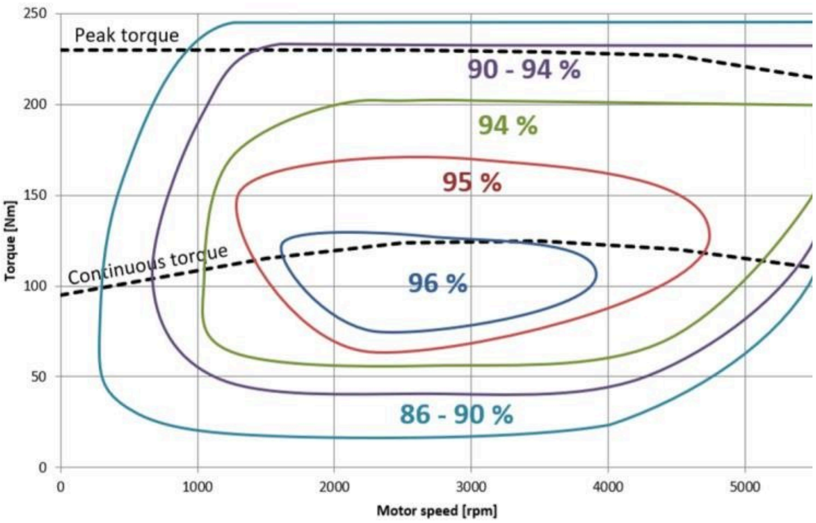
1. **Load CSV Data:**
   - The function reads the CSV file using and skips the first two rows, assuming the relevant data starts thereafter.
2. **Initialize Data Structures:**
   - A dictionary, `data_points`, is used to store lists of (x, y) for each efficiency level.
3. **Populate Data Points:**
   - Iterates through the CSV columns in pairs (since each efficiency level is represented by two columns: x and y).
   - For each pair, assigns the points to the correct efficiency level in `data_points`.

  - Stops reading further points if a NaN value is encountered in the current row, indicating the end of active data for that column pair.
4. **Create Polygons:**
   - For each efficiency level, if there are at least three points, a `Polygon` object is created from these points using the `shapely` library.
   - Polygons are stored in the `polygons` dictionary.
5. **Point Creation:**
   - A `Point` object is created from the input `speed` and `torque` values.
6. **Determine Efficiency Level:**
   - The function iterates over the efficiency levels in a predefined order from innermost to outermost (`["96", "95", "94", "92", "88"]`).
   - It checks if the point is contained within each polygon. The first matching polygon's efficiency level is returned.
   - If no polygons contain the point, the function returns `None`.

Returns:

- **Efficiency Level:** The efficiency level corresponding to the innermost polygon containing the point. If the point is not inside any polygon, returns `None`.

Pictured below is the Emrax 228 motor efficiency map that was recreated through linearized polygons:


Torque [Nm]
Peak torque
Continuous torque
Motor speed [rpm]