Question 1:

You are provided with a list of bounding boxes, each representing a car detected on a road within an image. Every bounding box is defined by two sets of coordinates:

- Top-left corner: (x_min, y_min)
- Bottom-right corner: (x_max, y_max)

Now, imagine dividing this image into an N x N grid (i.e., the image is split into N equally sized rows and N equally sized columns). Each grid cell covers a specific region of the image. We want to tell how many cars lie in each grid cell of image so we can know where in image there are maximum number of cars.

Your task is to write a Python function that:

Assigns each bounding box (representing a car) to one of the grid cells, based on where the box appears in the image. Counts how many bounding boxes lie inside each grid cell. You can assign a bounding box to a grid cell based on it's center.

For example, if N = 4, the image will be divided into a 4x4 grid, which means there are 16 grid cells. Some bounding boxes may span across multiple grid cells, and in such cases, check in which grid cell the center of box lies. Whichever grid cell the center of box belongs to we will assign the car to it.

In the end, your function should return a matrix that tell how many bounding boxes are present in each cell. Please refer the sample output fo clarity

Input:

boxes = [(10, 20, 50, 60), (110, 120, 150, 160), (210, 220, 250, 260), (310, 320, 350, 360)] img_width = 400 img_height = 400 n_rows = 4 n_cols = 4

Output:

[1, 0, 0, 0]

[0, 1, 0, 0]

[0, 0, 1, 0]

[0, 0, 0, 1]

Input:

boxes = [(10, 20, 50, 60), (20, 25, 55, 65), (35, 45, 75, 85), (40, 50, 80, 90), (50, 60, 100, 110), (110, 120, 150, 160), (130, 140, 170, 180), (160, 170, 200, 210), (175, 185,

195, 205)] img_width = 200 img_height = 200 n_rows = 5 n_cols = 5

Output:

[0, 0, 0, 0, 0]

[2, 2, 0, 0, 0]

[0, 1, 0, 0, 0]

[0, 0, 0, 1, 0]

[0, 0, 0, 1, 2]

Question 2:

You need to implement a sentiment analysis system that classifies text into positive, negative, or neutral categories.

Requirements:

- Create a baseline model using traditional ML methods (TF-IDF + SVM)
- Implement a transformer-based fine-tuned classifier
- Compare performance between approaches
- Evaluate using precision, recall, F1-score, and confusion matrices
- Implement methods to explain predictions (such as feature importance, attention visualization)

Question 3:

You will in this question develop a zero-shot classification pipeline using LLMs without task-specific fine-tuning.

Requirements:

- Implement prompt engineering techniques for classification tasks
- Compare performance across different prompt templates
- Evaluate on a dataset where classes were not seen during pre-training
- Analyse classification confidence and threshold selection
- Implement methods to improve zero-shot performance

Optional Question:

Design a prompt template for extracting structured information (e.g., names, dates, locations) from unstructured text. What considerations would you make to ensure consistent extraction?

note: use any unstructured text available.

Deadline: You are requested to submit the assignment within 3 days from the date it is shared with you.