Identifying Key Entities in Recipe Data - CRF Model Report

1. Business Objective

The goal of this assignment was to develop a Conditional Random Fields (CRF) model for Named Entity Recognition (NER) on culinary recipe data. The task involved identifying entities such as ingredients, quantities, and units from structured recipe texts, helping in building structured representations for use in recipe systems, e-commerce, or dietary tracking apps.

2. Data Overview & Preparation

The dataset contains a JSON list of recipe ingredients with input strings and associated POS-like entity tags for each word. The data was converted into a structured dataframe, tokenized into input_tokens and pos_tokens. 5 rows with inconsistent token-label lengths were removed to ensure data quality.

3. Exploratory Data Analysis

- The most common ingredients include 'salt', 'onion', and 'oil'.
- Most frequent units include 'tsp', 'tbsp', and 'cups'.
- Tokens were flattened and categorized into labels for quantity, unit, and ingredient for training insights.

4. Feature Engineering

Feature extraction was performed using spaCy. Core lexical, grammatical and contextual features were extracted for each token such as part-of-speech tags, shapes, and regex matches for quantities and units. Class weights were computed to penalize over-represented labels like 'ingredient'.

5. Model Training & Evaluation

A CRF model was trained with lbfgs optimizer and regularization parameters. Training accuracy and confusion matrix showed effective learning of entity relationships. Validation performance was evaluated with classification report and confusion matrix.

6. Error Analysis

Misclassified tokens were analyzed, highlighting errors in ambiguous or context-sensitive cases. Tokens with nearby quantities or unusual units showed higher misclassification. Error data included token context and class weights to understand the mispredictions.

7. Insights & Outcomes

- CRF successfully identified structured entities from unstructured recipe text.
- Class weighting helped improve minority class predictions.
- Cleaned data and context-aware features enhanced model performance.
- Errors revealed opportunity to improve via better unit detection and ambiguity handling.