Report: Identifying Key Entities in Recipe Data

Name: Suraj Bhadra Cohort: C75

Include your visualisations, analysis, results, insights, and outcomes. Explain your methodology and approach to the tasks. Add your conclusions to the sections.

Business Objective: The goal of this assignment is to train a Named Entity Recognition (NER) model using Conditional Random Fields (CRF) to extract key entities from recipe data. The model will classify words into predefined categories such as ingredients, quantities and units, enabling the creation of a structured database of recipes and ingredients that can be used to power advanced features in recipe management systems, dietary tracking apps, or e-commerce platforms.

Data Description:

The given data is in JSON format, representing a **structured recipe ingredient list** with **Named Entity Recognition (NER) labels**. Below is a breakdown of the data fields:


```
[
  {
    "input": "6 Karela Bitter Gourd Pavakkai Salt 1 Onion 3 tablespoon Gram flour besan 2 teaspoons Turmeric powder Haldi Red Chilli Cumin seeds Jeera Coriander Powder Dhania Amchur Dry Mango Sunflower Oil",
    "pos": "quantity ingredient ingredient ingredient ingredient ingredient quantity ingredient quantity unit ingredient ingredient ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient"
  },
  {
    "input": "2-1/2 cups rice cooked 3 tomatoes teaspoons BC Belle Bhat powder 1 teaspoon chickpea lentils 1/2 cumin seeds white urad dal mustard green chilli dry red 2 cashew or peanuts 1-1/2 tablespoon oil asafoetida",
    "pos": "quantity unit ingredient ingredient quantity ingredient unit ingredient ingredient ingredient ingredient quantity unit ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient quantity unit ingredient ingredient"
  }
]
```

Key	Description
`input`	Contains a raw ingredient list from a recipe.
`pos`	Represents the corresponding part-of-speech (POS) tags or NER labels, identifying quantities, ingredients, and units.

1 Import libraries

1.1 Installation of sklearn-crfsuite

- sklearn-crfsuite is a Python wrapper for CRFsuite, a fast and efficient implementation of Conditional Random Fields (CRFs). It is designed to integrate seamlessly with scikit-learn for structured prediction tasks such as Named Entity Recognition (NER), Part-of-Speech (POS) tagging, and chunking.

1.2 Import necessary libraries

```
# Import necessary libraries
import json  # For handling JSON data
import pandas as pd  # For data manipulation and analysis
import re  # For regular expressions (useful for text preprocessing)
import matplotlib.pyplot as plt  # For visualisation
import seaborn as sns  # For advanced data visualisation
import sklearn_crfsuite  # CRF (Conditional Random Fields) implementation for sequence modeling
import numpy as np  # For numerical computations
# Saving and loading machine learning models
import joblib
import random
import spacy
from IPython.display import display, Markdown # For displaying well-formatted output

from fractions import Fraction  # For handling fractional values in numerical data
# Importing tools for feature engineering and model training
from collections import Counter  # For counting occurrences of elements in a list
from sklearn.model_selection import train_test_split  # For splitting dataset into train and test sets
from sklearn_crfsuite import metrics  # For evaluating CRF models
from sklearn_crfsuite.metrics import flat_classification_report
from sklearn.utils.class_weight import compute_class_weight
from collections import Counter
from sklearn.metrics import confusion_matrix
```

2 Data Ingestion and Preparation [25 marks]

2.1 *Read Recipe Data from Dataframe and prepare the data for analysis* [12 marks]

Read the data from JSON file, print first five rows and describe the dataframe

2.1.1 Define a *load_json_dataframe* function [7 marks]

Define a function that takes path of the ingredient_and_quantity.json file and reads it, convert it into dataframe - df and return it.

```python
# define a function to load json file to a dataframe

def load_json_to_dataframe(filepath: str) -> pd.DataFrame:
    """
    Load the ingredient_and_quantity.json file and convert it into a pandas DataFrame.

    Parameters:
    -----------
    filepath : str
        Full path to the ingredient_and_quantity.json file.

    Returns:
    --------
    pd.DataFrame
        DataFrame created from the JSON file. Expects keys like 'input' and 'pos'.
    """
    # Open and read JSON file
    with open(filepath, 'r', encoding='utf-8') as f:
        data = json.load(f)

    # Convert JSON list/dict to DataFrame
    df = pd.DataFrame(data)

    # Validate essential columns
    required_cols = ['input', 'pos']
    for col in required_cols:
        if col not in df.columns:
            raise ValueError(f"Missing required column: {col}")

    return df
```

2.1.2 Execute the *load_json_dataframe* function [2 marks]

```python
# read the json file by giving the file path and create a dataframe

# File path to your JSON file
filepath = "ingredient_and_quantity.json"

# Load the DataFrame
df = load_json_to_dataframe(filepath)
```

2.1.3 Describe the dataframe [3 marks]

Print first five rows of dataframe along with dimensions. Display the information of dataframe

First five rows:

[222]:

	input	pos
0	6 Karela Bitter Gourd Pavakkai Salt 1 Onion 3 tablespoon Gram flour besan 2 teaspoons Turmeric powder Haldi Red Chilli Cumin seeds Jeera Coriander Powder Dhania Amchur Dry Mango Sunflower Oil	quantity ingredient ingredient ingredient ingredient ingredient quantity ingredient quantity unit ingredient ingredient ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient
1	2-1/2 cups rice cooked 3 tomatoes teaspoons BC Belle Bhat powder 1 teaspoon chickpea lentils 1/2 cumin seeds white urad dal mustard green chilli dry red 2 cashew or peanuts 1-1/2 tablespoon oil asafoetida	quantity unit ingredient ingredient quantity ingredient unit ingredient ingredient ingredient ingredient quantity unit ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient quantity ingredient ingredient quantity unit ingredient ingredient
2	1-1/2 cups Rice Vermicelli Noodles Thin 1 Onion sliced 1/2 cup Carrots Gajjar chopped 1/3 Green peas Matar 2 Chillies 1/4 teaspoon Asafoetida hing Mustard seeds White Urad Dal Split Ghee sprig Curry leaves Salt Lemon juice	quantity unit ingredient ingredient ingredient ingredient quantity ingredient ingredient quantity unit ingredient ingredient ingredient quantity ingredient ingredient ingredient quantity ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient
3	500 grams Chicken 2 Onion chopped 1 Tomato 4 Green Chillies slit inch Ginger finely 6 cloves Garlic 1/2 teaspoon Turmeric powder Haldi Garam masala tablespoon Sesame Gingelly Oil 1/4 Methi Seeds Fenugreek Coriander Dhania Dry Red Fennel seeds Saunf cups Sorrel Leaves Gongura picked and	quantity unit ingredient quantity ingredient ingredient quantity ingredient quantity ingredient ingredient ingredient unit ingredient ingredient quantity unit ingredient quantity unit ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient
4	1 tablespoon chana dal white urad 2 red chillies coriander seeds 3 inches ginger onion tomato Teaspoon mustard asafoetida sprig curry	quantity unit ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient quantity unit ingredient ingredient ingredient unit ingredient ingredient unit ingredient

[224]:
```
# print the dimensions of dataframe - df
df.shape
```

[224]:
```
(285, 2)
```

[226]:
```
# print the information of the dataframe
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 285 entries, 0 to 284
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   input   285 non-null    object
 1   pos     285 non-null    object
dtypes: object(2)
memory usage: 4.6+ KB
```

[228]:
```
df.describe(include='all')
```

[228]:

	input	pos
count	285	285
unique	285	284
top	6 Karela Bitter Gourd Pavakkai Salt 1 Onion 3 tablespoon Gram flour besan 2 teaspoons Turmeric powder Haldi Red Chilli Cumin seeds Jeera Coriander Powder Dhania Amchur Dry Mango Sunflower Oil	quantity ingredient ingredient ingredient ingredient quantity unit ingredient quantity unit ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient
freq	1	2

2.2 *Recipe Data Manipulation* [13 marks]

Create derived metrics in dataframe and provide insights of the dataframe

2.2.1 Create input_tokens and pos_tokens columns by splitting the input and pos from the dataframe [3 marks]

Split the input and pos into input_tokens and pos_tokens in the dataframe and display it in the dataframe

```
# split the input and pos into input_tokens and pos_tokens in the dataframe

# Tokenize input
df["input_tokens"] = df["input"].apply(lambda x: x.split())

# Tokenize POS
df["pos_tokens"] = df["pos"].apply(lambda x: x.split())
```

[234]:

	input	pos	input_tokens	pos_tokens
0	6 Karela Bitter Gourd Pavakkai Salt 1 Onion 3 tablespoon Gram flour besan 2 teaspoons Turmeric powder Haldi Red Chilli Cumin seeds Jeera Coriander Powder Dhania Amchur Dry Mango Sunflower Oil	quantity ingredient ingredient ingredient ingredient ingredient quantity ingredient quantity unit ingredient ingredient ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient	[6, Karela, Bitter, Gourd, Pavakkai, Salt, 1, Onion, 3, tablespoon, Gram, flour, besan, 2, teaspoons, Turmeric, powder, Haldi, Red, Chilli, Cumin, seeds, Jeera, Coriander, Powder, Dhania, Amchur, Dry, Mango, Sunflower, Oil]	[quantity, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, quantity, unit, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient]
1	2-1/2 cups rice cooked 3 tomatoes teaspoons BC Belle Bhat powder 1 teaspoon chickpea lentils 1/2 cumin seeds white urad dal mustard green chilli dry red 2 cashew or peanuts 1-1/2 tablespoon oil asafoetida	quantity unit ingredient ingredient quantity ingredient unit ingredient ingredient ingredient ingredient quantity unit ingredient ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient quantity unit ingredient ingredient	[2-1/2, cups, rice, cooked, 3, tomatoes, teaspoons, BC, Belle, Bhat, powder, 1, teaspoon, chickpea, lentils, 1/2, cumin, seeds, white, urad, dal, mustard, green, chilli, dry, red, 2, cashew, or, peanuts, 1-1/2, tablespoon, oil, asafoetida]	[quantity, unit, ingredient, ingredient, quantity, ingredient, unit, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient]
2	1-1/2 cups Rice Vermicelli Noodles Thin 1 Onion sliced 1/2 cup Carrots Gajjar chopped 1/3 Green peas Matar 2 Chillies 1/4 teaspoon Asafoetida hing Mustard seeds White Urad Dal Split Ghee sprig Curry leaves Salt Lemon juice	quantity unit ingredient ingredient ingredient ingredient quantity ingredient ingredient quantity unit ingredient ingredient ingredient quantity ingredient ingredient ingredient quantity ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient	[1-1/2, cups, Rice, Vermicelli, Noodles, Thin, 1, Onion, sliced, 1/2, cup, Carrots, Gajjar, chopped, 1/3, Green, peas, Matar, 2, Chillies, 1/4, teaspoon, Asafoetida, hing, Mustard, seeds, White, Urad, Dal, Split, Ghee, sprig, Curry, leaves, Salt, Lemon, juice]	[quantity, unit, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, quantity, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient]
3	500 grams Chicken 2 Onion chopped 1 Tomato 4 Green Chillies slit inch Ginger finely 6 cloves Garlic 1/2 teaspoon Turmeric powder Haldi Garam masala tablespoon Sesame Gingelly Oil 1/4 Methi Seeds Fenugreek Coriander Dhania Dry Red Fennel seeds Saunf cups Sorrel Leaves Gongura picked and	quantity unit ingredient quantity ingredient ingredient quantity ingredient quantity ingredient ingredient ingredient unit ingredient ingredient quantity unit ingredient quantity unit ingredient ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient	[500, grams, Chicken, 2, Onion, chopped, 1, Tomato, 4, Green, Chillies, slit, inch, Ginger, finely, 6, cloves, Garlic, 1/2, teaspoon, Turmeric, powder, Haldi, Garam, masala, tablespoon, Sesame, Gingelly, Oil, 1/4, Methi, Seeds, Fenugreek, Coriander, Dhania, Dry, Red, Fennel, seeds, Saunf, cups, Sorrel, Leaves, Gongura, picked, and]	[quantity, unit, ingredient, quantity, ingredient, ingredient, quantity, ingredient, quantity, ingredient, ingredient, ingredient, unit, ingredient, ingredient, quantity, unit, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient]
4	1 tablespoon chana dal white urad 2 red chillies coriander seeds 3 inches ginger onion tomato Teaspoon mustard asafoetida sprig curry	quantity unit ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient quantity unit ingredient ingredient ingredient unit ingredient ingredient unit ingredient	[1, tablespoon, chana, dal, white, urad, 2, red, chillies, coriander, seeds, 3, inches, ginger, onion, tomato, Teaspoon, mustard, asafoetida, sprig, curry]	[quantity, unit, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, unit, ingredient, ingredient, unit, ingredient]

2.2.2 Provide the length for input_tokens and pos_tokens and validate their length [2 marks]

Create input_length and pos_length columns in the dataframe and validate both the lengths. Check for the rows that are unequal in input and pos length

```
# create input_length and pos_length columns for the input_tokens and pos-tokens
df["input_length"] = df["input_tokens"].apply(len)
df["pos_length"] = df["pos_tokens"].apply(len)
```

```
# check for the equality of input_length and pos_length in the dataframe
unequal_rows = df[df["input_length"] != df["pos_length"]]
unequal_rows
```

[239]:

	input	pos	input_tokens	pos_tokens	input_length	pos_length
17	2 cups curd 1 cup gourd cucumber green cor coriander 1/2 teaspoon cumin powder salt	quantity unit ingredient quantity unit ingredient ingredient ingredient ingredient quantity unit ingredient ingredient ingredient	[2, cups, curd, 1, cup, gourd, cucumber, green, cor, coriander, 1/2, teaspoon, cumin, powder, salt]	[quantity, unit, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient]	15	14
27	1 Baguette sliced 1 1/2 tablespoon Butter 1/2 Garlic minced cup Spinach Leaves Palak Red Bell pepper Capsicum Tomato finely chopped Onion Black powder Italian seasoning teaspoon Fresh cream Cheddar cheese grated Salt Roasted tomato pasta sauce	quantity ingredient ingredient quantity unit ingredient quantity ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient	[1, Baguette, sliced, 1, 1/2, tablespoon, Butter, 1/2, Garlic, minced, cup, Spinach, Leaves, Palak, Red, Bell, pepper, Capsicum, Tomato, finely, chopped, Onion, Black, powder, Italian, seasoning, teaspoon, Fresh, cream, Cheddar, cheese, grated, Salt, Roasted, tomato, pasta, sauce]	[quantity, ingredient, ingredient, quantity, unit, ingredient, quantity, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient]	37	36
79	1/2 cup Poha Flattened rice 2 tablespoons Rice flour 2 1/2 liter Milk 1 Nolen Gur or brown sugar Cardamom Elaichi Pods/Seeds 8-10 Mixed nuts almonds/cashews tablespoon Raisins pinch Saffron strands and a little more for garnish Salt	quantity unit ingredient ingredient ingredient quantity unit ingredient ingredient quantity unit ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient unit ingredient unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient	[1/2, cup, Poha, Flattened, rice, 2, tablespoons, Rice, flour, 2, 1/2, liter, Milk, 1, Nolen, Gur, or, brown, sugar, Cardamom, Elaichi, Pods/Seeds, 8-10, Mixed, nuts, almonds/cashews, tablespoon, Raisins, pinch, Saffron, strands, and, a, little, more, for, garnish, Salt]	[quantity, unit, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, quantity, unit, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, unit, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient]	38	37
	1/2 cup All Purpose Flour Maida Whole Wheat 1/4 Hung Curd Greek Yogurt 250 grams Chicken minced 1 Spinach Leaves Palak	quantity unit ingredient ingredient ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient quantity unit ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient	[1/2, cup, All, Purpose, Flour, Maida, Whole, Wheat, 1/4, Hung, Curd, Greek, Yogurt, 250, grams, Chicken, minced, 1, Spinach, Leaves, Palak, finely, chopped	[quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient		
164	1/2 cup All Purpose Flour Maida Whole Wheat 1/4 Hung Curd Greek Yogurt 250 grams Chicken minced 1 Spinach Leaves Palak finely chopped Onion 4 cloves Garlic Tomatoes tablespoon Cumin powder Jeera Coriander Powder Dhania 1 1/2 teaspoon Paprika Black pepper 3 sprig Mint Pudina 10 Spring Bulb & Greens 100 Feta Cheese crumbled	quantity unit ingredient ingredient ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient quantity unit ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient quantity unit ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient ingredient quantity unit ingredient ingredient ingredient quantity unit ingredient ingredient quantity ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient	[1/2, cup, All, Purpose, Flour, Maida, Whole, Wheat, 1/4, Hung, Curd, Greek, Yogurt, 250, grams, Chicken, minced, 1, Spinach, Leaves, Palak, finely, chopped, Onion, 4, cloves, Garlic, Tomatoes, tablespoon, Cumin, powder, Jeera, Coriander, Powder, Dhania, 1, 1/2, teaspoon, Paprika, Black, pepper, 3, sprig, Mint, Pudina, 10, Spring, Bulb, &, Greens, 100, Feta, Cheese, crumbled]	[quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient]	54	53
207	1 cup Cashew nuts Badam Almond 1 1/4 cups Sugar 1/2 Water teaspoon Cardamom Powder Ghee for greasing	quantity unit ingredient ingredient ingredient ingredient quantity unit ingredient quantity ingredient unit ingredient ingredient ingredient unit ingredient	[1, cup, Cashew, nuts, Badam, Almond, 1, 1/4, cups, Sugar, 1/2, Water, teaspoon, Cardamom, Powder, Ghee, for, greasing]	[quantity, unit, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, quantity, ingredient, unit, ingredient, ingredient, ingredient, unit, ingredient]	18	17

2.2.3 Define a unique_labels function and validate the labels in pos_tokens [2 marks]

Define a unique_labels function which checks for all the unique pos labels in the recipe & execute it.

```
unique_pos_tags = unique_labels(df)
```

```
Unique POS Labels in the Recipe Dataset:
ingredient
quantity
unit
```

2.2.3 Provide the insights seen in the recipe data after validation [1 marks]

Provide the indexes that requires cleaning and formatting in the dataframe

```
unequal_rows.index.tolist()
```

```
[17, 27, 79, 164, 207]
```

```
invalid_indexes = [17, 27, 79, 164, 207]
```

2.2.4 Drop the rows that have invalid data provided in previous cell [2 marks]

```
# drop the irrelevant recipe data
df_cleaned = df.drop(index=invalid_indexes).reset_index(drop=True)

# display the cleaned dataframe
df_cleaned.head()
```

	input	pos	input_tokens	pos_tokens	input_length	pos_length
0	6 Karela Bitter Gourd Pavakkai Salt 1 Onion 3 tablespoon Gram flour besan 2 teaspoons Turmeric powder Haldi Red Chilli Cumin seeds Jeera Coriander Powder Dhania Amchur Dry Mango Sunflower Oil	quantity ingredient ingredient ingredient ingredient ingredient quantity ingredient quantity unit ingredient ingredient ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient	[6, Karela, Bitter, Gourd, Pavakkai, Salt, 1, Onion, 3, tablespoon, Gram, flour, besan, 2, teaspoons, Turmeric, powder, Haldi, Red, Chilli, Cumin, seeds, Jeera, Coriander, Powder, Dhania, Amchur, Dry, Mango, Sunflower, Oil]	[quantity, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, quantity, unit, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient]	31	31
1	2-1/2 cups rice cooked 3 tomatoes teaspoons BC Belle Bhat powder 1 teaspoon chickpea lentils 1/2 cumin seeds white urad dal mustard green chilli dry red 2 cashew or peanuts 1-1/2 tablespoon oil asafoetida	quantity unit ingredient ingredient quantity ingredient unit ingredient ingredient ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient quantity unit ingredient ingredient	[2-1/2, cups, rice, cooked, 3, tomatoes, teaspoons, BC, Belle, Bhat, powder, 1, teaspoon, chickpea, lentils, 1/2, cumin, seeds, white, urad, dal, mustard, green, chilli, dry, red, 2, cashew, or, peanuts, 1-1/2, tablespoon, oil, asafoetida]	[quantity, unit, ingredient, ingredient, quantity, ingredient, unit, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient]	34	34
2	1-1/2 cups Rice Vermicelli Noodles Thin 1 Onion sliced 1/2 cup Carrots Gajjar chopped 1/3 Green peas Matar 2 Chillies 1/4 teaspoon Asafoetida hing Mustard seeds White Urad Dal Split Ghee sprig Curry leaves Salt Lemon juice	quantity unit ingredient ingredient ingredient ingredient quantity ingredient ingredient quantity unit ingredient ingredient ingredient quantity ingredient ingredient ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient	[1-1/2, cups, Rice, Vermicelli, Noodles, Thin, 1, Onion, sliced, 1/2, cup, Carrots, Gajjar, chopped, 1/3, Green, peas, Matar, 2, Chillies, 1/4, teaspoon, Asafoetida, hing, Mustard, seeds, White, Urad, Dal, Split, Ghee, sprig, Curry, leaves, Salt, Lemon, juice]	[quantity, unit, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient]	37	37
3	500 grams Chicken 2 Onion chopped 1 Tomato 4 Green Chillies slit inch Ginger finely 6 cloves Garlic 1/2 teaspoon Turmeric powder Haldi Garam masala tablespoon Sesame Gingelly Oil 1/4 Methi Seeds Fenugreek Coriander Dhania Dry Red Fennel seeds Saunf cups Sorrel Leaves Gongura picked and	quantity unit ingredient quantity ingredient ingredient quantity ingredient ingredient quantity ingredient ingredient unit ingredient ingredient quantity unit ingredient ingredient quantity unit ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient	[500, grams, Chicken, 2, Onion, chopped, 1, Tomato, 4, Green, Chillies, slit, inch, Ginger, finely, 6, cloves, Garlic, 1/2, teaspoon, Turmeric, powder, Haldi, Garam, masala, tablespoon, Sesame, Gingelly, Oil, 1/4, Methi, Seeds, Fenugreek, Coriander, Dhania, Dry, Red, Fennel, seeds, Saunf, cups, Sorrel, Leaves, Gongura, picked, and]	[quantity, unit, ingredient, quantity, ingredient, ingredient, quantity, ingredient, quantity, ingredient, ingredient, unit, ingredient, ingredient, quantity, unit, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient]	46	46
4	1 tablespoon chana dal white urad 2 red chillies coriander seeds 3 inches ginger onion tomato Teaspoon mustard asafoetida sprig curry	quantity unit ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient quantity unit ingredient ingredient unit ingredient ingredient unit ingredient	[1, tablespoon, chana, dal, white, urad, 2, red, chillies, coriander, seeds, 3, inches, ginger, onion, tomato, Teaspoon, mustard, asafoetida, sprig, curry]	[quantity, unit, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, unit, ingredient, ingredient, unit, ingredient]	21	21

2.2.5 Update the input_length & pos_length in dataframe [2 marks]

```
# update the input and pos length in input_length and pos_length
# Recalculate lengths after cleaning
df_cleaned["input_length"] = df_cleaned["input_tokens"].apply(len)
df_cleaned["pos_length"] = df_cleaned["pos_tokens"].apply(len)

# Display to verify
df_cleaned[["input", "input_length", "pos_length"]].head()
```

	input	input_length	pos_length
0	6 Karela Bitter Gourd Pavakkai Salt 1 Onion 3 tablespoon Gram flour besan 2 teaspoons Turmeric powder Haldi Red Chilli Cumin seeds Jeera Coriander Powder Dhania Amchur Dry Mango Sunflower Oil	31	31
1	2-1/2 cups rice cooked 3 tomatoes teaspoons BC Belle Bhat powder 1 teaspoon chickpea lentils 1/2 cumin seeds white urad dal mustard green chilli dry red 2 cashew or peanuts 1-1/2 tablespoon oil asafoetida	34	34
2	1-1/2 cups Rice Vermicelli Noodles Thin 1 Onion sliced 1/2 cup Carrots Gajjar chopped 1/3 Green peas Matar 2 Chillies 1/4 teaspoon Asafoetida hing Mustard seeds White Urad Dal Split Ghee sprig Curry leaves Salt Lemon juice	37	37
3	500 grams Chicken 2 Onion chopped 1 Tomato 4 Green Chillies slit inch Ginger finely 6 cloves Garlic 1/2 teaspoon Turmeric powder Haldi Garam masala tablespoon Sesame Gingelly Oil 1/4 Methi Seeds Fenugreek Coriander Dhania Dry Red Fennel seeds Saunf cups Sorrel Leaves Gongura picked and	46	46
4	1 tablespoon chana dal white urad 2 red chillies coriander seeds 3 inches ginger onion tomato Teaspoon mustard asafoetida sprig curry	21	21

2.2.6 Validate the input_length and pos_length by checking unequal rows [1 marks]

```
# validate the input length and pos length as input_length and pos_length
df_cleaned[df_cleaned["input_length"] != df_cleaned["pos_length"]]
```

input	pos	input_tokens	pos_tokens	input_length	pos_length

3 Train Validation Split (70 train - 30 val) [6 marks]

3.1 *Perform train and validation split ratio* [6 marks]

Split the dataset with the help of input_tokens and pos_tokens and make a ratio of 70:30 split for training and validation datasets.

3.1.1 Split the dataset into train_df and val_df into 70:30 ratio [1 marks]

```
# split the dataset into training and validation sets
train_df, val_df = train_test_split(
    df_cleaned,
    test_size=0.30,      # 30% for validation
    random_state=42,     # ensures reproducibility
    shuffle=True
)

# display sizes
print("Training set size:", len(train_df))
print("Validation set size:", len(val_df))
```

```
Training set size: 196
Validation set size: 84
```

3.1.2 Print the first five rows of train_df and val_df [1 marks]

```python
# print the first five rows of train_df
train_df.head()
```

	input	pos	input_tokens	pos_tokens	input_length	pos_length
175	250 grams Okra Oil 1 Onion finely chopped Tomato Grated teaspoon Ginger 2 Garlic Finely 1/2 Cumin seeds 1/4 Teaspoon asafoetida cup cottage cheese pinched coriander powder mango red chilli turmeric	quantity unit ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient unit ingredient quantity ingredient ingredient quantity ingredient ingredient quantity unit ingredient unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient	[250, grams, Okra, Oil, 1, Onion, finely, chopped, Tomato, Grated, teaspoon, Ginger, 2, Garlic, Finely, 1/2, Cumin, seeds, 1/4, Teaspoon, asafoetida, cup, cottage, cheese, pinched, coriander, powder, mango, red, chilli, turmeric]	[quantity, unit, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient, unit, ingredient, quantity, ingredient, ingredient, quantity, ingredient, ingredient, quantity, unit, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient]	31	31
55	200 grams Paneer Homemade Cottage Cheese 2 Potato Aloo Bay leaf tej patta Dry Red Chilli 1 tablespoon Panch Phoran Masala roasted and powdered Tomato big sized teaspoon Turmeric powder Haldi Cumin seeds Jeera Ginger grated Salt 1/2 Sugar Sunflower Oil	quantity unit ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient	[200, grams, Paneer, Homemade, Cottage, Cheese, 2, Potato, Aloo, Bay, leaf, tej, patta, Dry, Red, Chilli, 1, tablespoon, Panch, Phoran, Masala, roasted, and, powdered, Tomato, big, sized, teaspoon, Turmeric, powder, Haldi, Cumin, seeds, Jeera, Ginger, grated, Salt, 1/2, Sugar, Sunflower, Oil]	[quantity, unit, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient]	41	41
109	500 grams Cabbage Patta Gobi Muttaikose 1 teaspoon Mustard seeds 1-1/2 White Urad Dal Split sprig Curry leaves Green Chilli 1/4 cup Fresh coconut Salt	quantity unit ingredient ingredient ingredient ingredient quantity unit ingredient ingredient quantity ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient quantity unit ingredient ingredient ingredient	[500, grams, Cabbage, Patta, Gobi, Muttaikose, 1, teaspoon, Mustard, seeds, 1-1/2, White, Urad, Dal, Split, sprig, Curry, leaves, Green, Chilli, 1/4, cup, Fresh, coconut, Salt]	[quantity, unit, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient]	25	25
213	500 grams Fresh Figs 1/4 cup Lemon juice 1 teaspoon zest 2 Red Chilli flakes 1/2 Honey Brown Sugar (Demerara Sugar)	quantity unit ingredient ingredient quantity unit ingredient ingredient quantity unit ingredient quantity ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient	[500, grams, Fresh, Figs, 1/4, cup, Lemon, juice, 1, teaspoon, zest, 2, Red, Chilli, flakes, 1/2, Honey, Brown, Sugar, (Demerara, Sugar)]	[quantity, unit, ingredient, ingredient, quantity, unit, ingredient, ingredient, quantity, unit, ingredient, quantity, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient]	21	21
38	2 cups Water 1 teaspoon Tea leaves 1/4 Milk 10 Saffron strands	quantity unit ingredient quantity unit ingredient ingredient quantity ingredient quantity ingredient ingredient	[2, cups, Water, 1, teaspoon, Tea, leaves, 1/4, Milk, 10, Saffron, strands]	[quantity, unit, ingredient, quantity, unit, ingredient, ingredient, quantity, ingredient, quantity, ingredient, ingredient]	12	12

```python
# print the first five rows of the val_df
val_df.head()
```

	input	pos	input_tokens	pos_tokens	input_length	pos_length
33	1 cup Ada 2 liter Milk 3/4 Sugar tablespoon Ghee 1/2 teaspoon Cardamom Powder Elaichi	quantity unit ingredient quantity unit ingredient quantity ingredient unit ingredient quantity unit ingredient ingredient ingredient	[1, cup, Ada, 2, liter, Milk, 3/4, Sugar, tablespoon, Ghee, 1/2, teaspoon, Cardamom, Powder, Elaichi]	[quantity, unit, ingredient, quantity, unit, ingredient, quantity, ingredient, unit, ingredient, quantity, unit, ingredient, ingredient, ingredient]	15	15
108	1 Carrot Gajjar chopped 7 Potatoes Aloo 2 cups Cauliflower gobi cut to small florets Onion tablespoon Ginger Garlic Paste Salt teaspoons Sunflower Oil 1/2 cup Fresh coconut grated teaspoon Whole Black Peppercorns Green Chillies Fennel seeds Saunf Poppy 6 Cashew nuts inch Cinnamon Stick Dalchini Star anise 3 Cloves Laung Cardamom Elaichi Pods/Seeds Cumin Jeera	quantity ingredient ingredient ingredient quantity ingredient ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient quantity unit ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient quantity ingredient ingredient unit ingredient ingredient ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient	[1, Carrot, Gajjar, chopped, 7, Potatoes, Aloo, 2, cups, Cauliflower, gobi, cut, to, small, florets, Onion, tablespoon, Ginger, Garlic, Paste, Salt, teaspoons, Sunflower, Oil, 1/2, cup, Fresh, coconut, grated, teaspoon, Whole, Black, Peppercorns, Green, Chillies, Fennel, seeds, Saunf, Poppy, 6, Cashew, nuts, inch, Cinnamon, Stick, Dalchini, Star, anise, 3, Cloves, Laung, Cardamom, Elaichi, Pods/Seeds, Cumin, Jeera]	[quantity, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, unit, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient]	56	56
240	1 tablespoon Sunflower Oil 3 Potato Aloo Ginger paste Green Chilli chopped 1-1/12 tablespoons Sesame seeds Til teaspoon Red powder Cumin Jeera Coriander Powder Dhania 1/2 Garam masala 2 Sweet Chutney Date Tamarind Leaves few	quantity unit ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient quantity unit ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient quantity ingredient ingredient quantity ingredient ingredient ingredient ingredient ingredient ingredient	[1, tablespoon, Sunflower, Oil, 3, Potato, Aloo, Ginger, paste, Green, Chilli, chopped, 1-1/12, tablespoons, Sesame, seeds, Til, teaspoon, Red, powder, Cumin, Jeera, Coriander, Powder, Dhania, 1/2, Garam, masala, 2, Sweet, Chutney, Date, Tamarind, Leaves, few]	[quantity, unit, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, ingredient]	35	35
259	1 cup green peas gram flour 1/2 cheese tsp ginger 2 chillies turmeric powder cumin teaspoon salt oil	quantity unit ingredient ingredient ingredient ingredient quantity ingredient unit ingredient quantity ingredient ingredient ingredient ingredient unit ingredient ingredient	[1, cup, green, peas, gram, flour, 1/2, cheese, tsp, ginger, 2, chillies, turmeric, powder, cumin, teaspoon, salt, oil]	[quantity, unit, ingredient, ingredient, ingredient, ingredient, quantity, ingredient, unit, ingredient, quantity, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient]	18	18
154	2 cups Brown Rice cooked tablespoons Garlic chopped 1 Green Chilli 1/2 cup Carrots (Gajjar) beans (French Beans) Bell Pepper (Capsicum) Onion Cabbage (Patta Gobi/ Muttaikose) tablespoon Roasted tomato pasta sauce - or store bought Red teaspoon Soy Ginger freshly grated Spring Greens Salt Vinegar Extra Virgin Olive Oil as required	quantity unit ingredient ingredient ingredient unit ingredient ingredient quantity ingredient ingredient quantity unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient unit ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient ingredient	[2, cups, Brown, Rice, cooked, tablespoons, Garlic, chopped, 1, Green, Chilli, 1/2, cup, Carrots, (Gajjar), beans, (French, Beans), Bell, Pepper, (Capsicum), Onion, Cabbage, (Patta, Gobi/, Muttaikose), tablespoon, Roasted, tomato, pasta, sauce, -, or, store, bought, Red, teaspoon, Soy, Ginger, freshly, grated, Spring, Greens, Salt, Vinegar, Extra, Virgin, Olive, Oil, as, required]	[quantity, unit, ingredient, ingredient, ingredient, unit, ingredient, ingredient, quantity, ingredient, ingredient, quantity, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, unit, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient, ingredient]	51	51

3.1.3 Extract the dataset into train_df and val_df into X_train, X_val, y_train and y_val and display their length [2 marks]

Extract X_train, X_val, y_train and y_val by extracting the list of input_tokens and pos_tokens from train_df and val_df and also display their length

```
# extract the training and validation sets by taking input_tokens and pos_tokens
X_train = train_df["input_tokens"].tolist()
y_train = train_df["pos_tokens"].tolist()

X_val = val_df["input_tokens"].tolist()
y_val = val_df["pos_tokens"].tolist()
```

```
Length of X_train: 196
Length of y_train: 196
Length of X_val: 84
Length of y_val: 84
```

3.1.4 Display the number of unique labels present in y_train [2 marks]

```
Number of unique labels in y_train: 3
Unique labels: {'ingredient', 'quantity', 'unit'}
```

4 Exploratory Recipe Data Analysis on Training Dataset [16 marks]

4.1 *Flatten the lists for input_tokens & pos_tokens* [2 marks]

Define a function **flatten_list** for flattening the structure for input_tokens and pos_tokens. The input parameter passed to this function is a nested list.
Initialise the dataset_name with a value ***'Training'***

```python
# flatten the list for nested_list (input_tokens, pos_tokens)
def flatten_list(nested_list):
    """
    Flattens a nested list (list of lists) into a single list.
    Example: [['add','salt'], ['mix','well']] -> ['add','salt','mix','well']
    """
    flat = [item for sublist in nested_list for item in sublist]
    return flat
```

```python
# initialise the dataset_name
dataset_name = 'Training'
```

4.2 *Extract and validate the tokens after using the flattening technique* [2 marks]

Define a function named ***extract_and_validate_tokens*** with parameters dataframe and dataset_name (Training/Validation), validate the length of input_tokens and pos_tokens from dataframe and display first 10 records for both the input_tokens and pos_tokens. Execute this function

```
Dataset: Training
Length of flattened input_tokens: 7114
Length of flattened pos_tokens: 7114
✓ Validation Passed: Both flattened lists have equal length.

First 10 input tokens: ['250', 'grams', 'Okra', 'Oil', '1', 'Onion', 'finely', 'chopped', 'Tomato', 'Grated']
First 10 POS tokens:   ['quantity', 'unit', 'ingredient', 'ingredient', 'quantity', 'ingredient', 'ingredient', 'ingredient', 'ingredient', 'ingredien
t']
```

4.3 *Categorise tokens into labels (unit, ingredient, quantity)* [2 marks]

Define a function ***categorize_tokens*** to categorise tokens into ingredients, units and quantities by using extracted tokens in the previous code and return a list of ingredients, units and quantities. Execute this function to get the list.

```
unique_pos_tags = set(pos_tags)
unique_pos_tags
```

```
{'ingredient', 'quantity', 'unit'}
```

```
Found 5323 ingredient tokens, 1791 unit tokens, 0 quantity tokens.
Ingredients (sample): ['Okra', 'Oil', 'Onion', 'finely', 'chopped', 'Tomato', 'Grated', 'Ginger', 'Garlic', 'Finely', 'Cumin', 'seeds', 'asafoetida', 'co
ttage', 'cheese', 'pinched', 'coriander', 'powder', 'mango', 'red']
Units (sample):      ['250', 'grams', '1', 'teaspoon', '2', '1/2', '1/4', 'Teaspoon', 'cup', '200', 'grams', '2', '1', 'tablespoon', 'teaspoon', '1/2',
'500', 'grams', '1', 'teaspoon']
Quantities (sample):  []
```

4.4 *Top 10 Most Frequent Items* [3 marks]

Define a function ***get_top_frequent_items*** to display top 10 most frequent items
Here, item_list is used as a general parameter where you will call this function for ingredient and unit list
Execute this function separately for top 10 most units and ingredients

```
# get the top ingredients which are frequently seen in the recipe
top_ingredients = get_top_frequent_items(ingredients, pos_label="Ingredient", dataset_name=dataset_name, top_n=10)
```

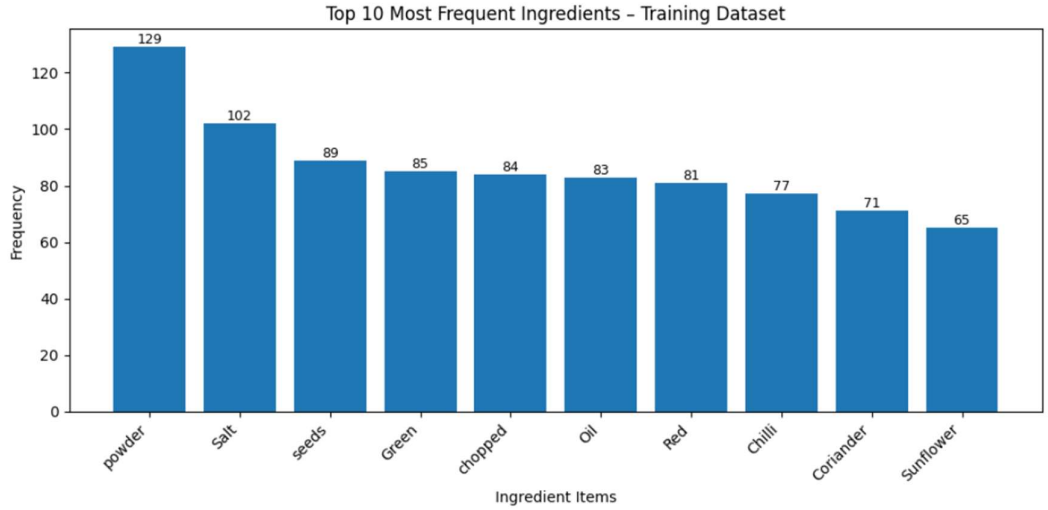
```
[Training | Ingredient] Top 10 items:
 1. 'powder' — 129
 2. 'Salt' — 102
 3. 'seeds' — 89
 4. 'Green' — 85
 5. 'chopped' — 84
 6. 'Oil' — 83
 7. 'Red' — 81
 8. 'Chilli' — 77
 9. 'Coriander' — 71
10. 'Sunflower' — 65
```

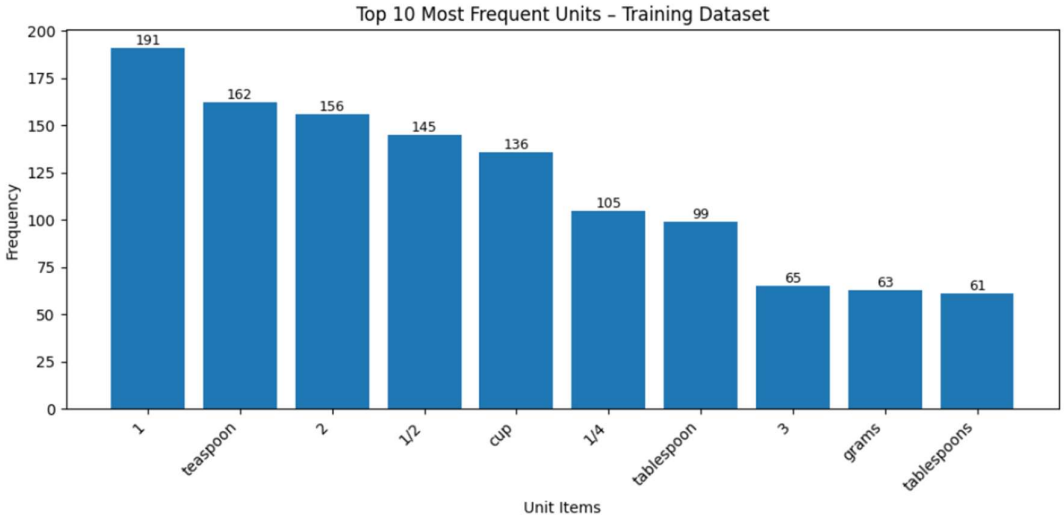
```
# get the top units which are frequently seen in the recipe
top_units = get_top_frequent_items(units, pos_label="Unit", dataset_name=dataset_name, top_n=10)
```

```
[Training | Unit] Top 10 items:
 1. '1' — 191
 2. 'teaspoon' — 162
 3. '2' — 156
 4. '1/2' — 145
 5. 'cup' — 136
 6. '1/4' — 105
 7. 'tablespoon' — 99
 8. '3' — 65
 9. 'grams' — 63
10. 'tablespoons' — 61
```

4.5 *Plot Top 10 most frequent items* [2 marks]

Define a function ***plot_top_items*** to plot a bar graph on top 10 most frequent items for units and ingredients
Here, item_list is used as a general parameter where you will call this function for ingredient and unit list





6 Feature Extraction For CRF Model [30 marks]

6.1 *Define a feature functions to take each token from recipe* [10 marks]
Define a function as ***word2features*** which takes a particular recipe and its index to work with all recipe input tokens and include custom key-value pairs.
Also, use feature key-value pairs to mark the beginning and end of the sequence and to also check whether the word belongs to unit, quantity etc. Use keyword sets for unit and quantity for differentiating feature functions well. Also make use of relevant regex patterns on fractions, whole numbers etc.

6.1.1 Define keywords for unit and quantity and create a quantity pattern to work on fractions, numbers and decimals [3 marks]

Create sets for **unit_keywords** and ***quantity_keywords*** and include all the words relevant for measuring the ingredients such as cup, tbsp, tsp etc. and in quantity keywords, include words such as half, quarter etc.
Also suggested to use regex pattern as ***quantity_pattern*** to work with quantity in any format such as fractions, numbers and decimals.

Then, load the spacy model and process the entire sentence

```
# define unit and quantity keywords along with quantity pattern
import re

# Unit keywords commonly found in recipes
unit_keywords = {
    "cup", "cups",
    "tbsp", "tablespoon", "tablespoons", "tbsps",
    "tsp", "teaspoon", "teaspoons",
    "gram", "grams", "g",
    "kg", "kilogram", "kilograms",
    "mg", "milligram", "milligrams",
    "ml", "milliliter", "milliliters",
    "l", "liter", "litre", "liters", "litres",
    "oz", "ounce", "ounces",
    "lb", "lbs", "pound", "pounds",
    "pinch", "pinches",
    "dash", "dashes",
    "clove", "cloves",
    "slice", "slices",
    "can", "cans",
    "packet", "packets",
    "package", "packages",
    "stick", "sticks",
    "piece", "pieces",
    "bunch", "bunches"
}

# Quantity keywords used in recipes
quantity_keywords = {
    "half", "quarter", "third",
    "one", "two", "three", "four", "five", "six",
    "seven", "eight", "nine", "ten",
    "dozen",
    "few", "several", "couple"
}
```

```
# Unicode fraction symbols commonly found
unicode_fractions = "¼|½|¾|⅐|⅑|⅒|⅓|⅔|⅕|⅖|⅗|⅘|⅙|⅚|⅛|⅜|⅝|⅞"

# Quantity regex pattern:
# matches integers, decimals, fractions, mixed numbers, unicode fractions
quantity_pattern = re.compile(
    rf"""^(
        (\d+(\.\d+)?) |            # integer or decimal -> 1, 1.5
        (\d+[\s-]+\d+/\d+) |      # mixed fraction -> 1 1/2 or 1-1/2
        (\d+/\d+) |               # simple fraction -> 1/2
        ({unicode_fractions})     # unicode fraction -> ½, ¼
    )$""",
    re.VERBOSE
)
```

```
# load spaCy model
import spacy

try:
    # Try loading the small English model
    nlp = spacy.load("en_core_web_sm")
    print("Loaded spaCy model: en_core_web_sm")

except Exception as e:
    print("spaCy model 'en_core_web_sm' not found. Falling back to blank English model.")

    # Fallback tokenizer-only model (no POS, no NER)
    nlp = spacy.blank("en")

nlp  # returns the loaded spaCy pipeline
```

```
Loaded spaCy model: en_core_web_sm
<spacy.lang.en.English at 0x243157faae0>
```

6.1.2 Define feature functions for CRF [7 marks]

Define ***word2features*** function and use the parameters such as sentence and its indexing as ***sent*** and ***i*** for extracting token level features for CRF Training. Build ***features*** dictionary, also mark the beginning and end of the sequence and use the ***unit_keywords***, ***quantity_keywords*** and ***quantity_pattern*** for knowing the presence of quantity or unit in the tokens

```python
# define word2features for processing each token in the sentence sent by using index i.
# use your own feature functions
_fraction_re = re.compile(r'^\d+/\d+$')
_decimal_re = re.compile(r'^\d+\.\d+$')
_mixed_re = re.compile(r'^\d+[\s-]+\d+/\d+$')

def is_fraction_text(s: str) -> bool:
    s = s.strip()
    return bool(_fraction_re.match(s) or _mixed_re.match(s) or re.search(unicode_fractions, s))

def is_decimal_text(s: str) -> bool:
    return bool(_decimal_re.match(s))

def matches_quantity_pattern(s: str) -> bool:
    return bool(quantity_pattern.match(s.strip()))

def is_unit_text(s: str) -> bool:
    return s.lower() in unit_keywords

def is_quantity_keyword_text(s: str) -> bool:
    return s.lower() in quantity_keywords
```

```python
# Process the entire sentence with spaCy
# main function
def word2features(sent, i):
    """
    sent: spaCy Doc or list of spaCy Token objects OR a raw string (will be processed by nlp)
    i: index of token in sent
    returns: dict of features for token at position i
    """
    # require nlp to be available
    try:
        nlp  # noqa: F821
    except NameError:
        raise RuntimeError("spaCy pipeline 'nlp' not found. Load spaCy (e.g., nlp = spacy.load('en_core_web_sm')) before calling this function.")

    # if user passed a raw string, process it
    if isinstance(sent, str):
        doc = nlp(sent)
        tokens = list(doc)
    else:
        tokens = list(sent)  # assume spaCy tokens or strings

    # ensure i is valid
    if not (0 <= i < len(tokens)):
        raise IndexError("Index i out of range for sentence tokens.")

    # pick token object or string text
    use_spacy_token = hasattr(tokens[0], "text")
    tok = tokens[i]
    text = tok.text if use_spacy_token else str(tok)
    lower = text.lower()
```

```python
# --- Core Features ---
features = {
    "bias": 1.0,
    "token": lower,
    "shape": tok.shape_ if use_spacy_token else ("X" if any(c.isalpha() for c in text) else text),
    "is_stop": tok.is_stop if use_spacy_token else False,
    "is_digit": tok.is_digit if use_spacy_token else text.isdigit(),
    "has_digit": any(ch.isdigit() for ch in text),
    "has_alpha": any(ch.isalpha() for ch in text),
    "hyphenated": "-" in text,
    "slash_present": "/" in text,
    "is_title": text.istitle(),
    "is_upper": text.isupper(),
    "is_punct": tok.is_punct if use_spacy_token else (len(text) == 1 and not text.isalnum()),
}

# POS/lemma/tag/dep when spaCy available; fallback placeholders otherwise
if use_spacy_token:
    features.update({
        "lemma": tok.lemma_.lower(),
        "pos_tag": tok.pos_,
        "tag": tok.tag_,
        "dep": tok.dep_,
    })
else:
    features.update({
        "lemma": lower,
        "pos_tag": "X",
        "tag": "X",
        "dep": "X",
    })
```

```python
# --- Improved Quantity & Unit Detection ---
fq_is_quantity_kw = is_quantity_keyword_text(lower)
fq_is_unit_kw = is_unit_text(lower)
fq_matches_qpat = matches_quantity_pattern(lower)
fq_is_fraction = is_fraction_text(lower)
fq_is_decimal = is_decimal_text(lower)
# spaCy tokens have .like_num boolean which is robust for numbers; fallback to pattern checks
if use_spacy_token:
    like_num = tok.like_num
else:
    like_num = bool(lower.isdigit() or fq_is_decimal or fq_is_fraction or fq_matches_qpat)

features.update({
    "is_quantity": bool(fq_is_quantity_kw or fq_matches_qpat or like_num),
    "is_unit": bool(fq_is_unit_kw),
    "is_numeric": bool(like_num),
    "is_fraction": bool(fq_is_fraction),
    "is_decimal": bool(fq_is_decimal),
})
```

```
# --- Contextual Features ---
# preceding token details
if i > 0:
    prev_tok = tokens[i - 1]
    prev_text = prev_tok.text if use_spacy_token else str(prev_tok)
    prev_lower = prev_text.lower()
    prev_is_quantity = is_quantity_keyword_text(prev_lower) or matches_quantity_pattern(prev_lower) or (prev_tok.like_num if use_spacy_token else any
    prev_is_digit = (prev_tok.like_num if use_spacy_token else prev_text.isdigit())
    features.update({
        "preceding_word": prev_text,
        "prev_token": prev_lower,
        "prev_is_quantity": bool(prev_is_quantity),
        "prev_is_digit": bool(prev_is_digit),
    })
else:
    features["BOS"] = True
    features.update({
        "preceding_word": "",
        "prev_token": "",
        "prev_is_quantity": False,
        "prev_is_digit": False,
    })
```

```
  # following token details
  if i < len(tokens) - 1:
      next_tok = tokens[i + 1]
      next_text = next_tok.text if use_spacy_token else str(next_tok)
      next_lower = next_text.lower()
      next_is_unit = is_unit_text(next_lower)
      next_is_quantity = is_quantity_keyword_text(next_lower) or matches_quantity_pattern(next_lower)
      # next_is_ingredient heuristic: not a unit and not a quantity (weak signal)
      next_is_ingredient = not (next_is_unit or next_is_quantity)
      features.update({
          "following_word": next_text,
          "next_token": next_lower,
          "next_is_unit": bool(next_is_unit),
          "next_is_ingredient": bool(next_is_ingredient),
      })
  else:
      features["EOS"] = True
      features.update({
          "following_word": "",
          "next_token": "",
          "next_is_unit": False,
          "next_is_ingredient": False,
      })

  return features
```

```
# Utility: produce features for entire sentence (CRF-ready)
def sent2features(sent):
    if isinstance(sent, str):
        doc = nlp(sent)
        tokens = list(doc)
    else:
        tokens = list(sent)
    return [word2features(tokens, i) for i in range(len(tokens))]
```

6.2 *Preparation of Recipe level features* [2 marks]

6.2.1 Define function to work on all the recipes and call word2features for each recipe [2 marks]

Define ***sent2features*** function and inputs ***sent*** as a parameter and correctly generate feature functions for each token present in the sentence

```
# define sent2features by working on each token in the sentence and correctly generate dictionaries for features
def sent2features(sent):
    """
    Generate CRF feature dictionaries for every token in a sentence.
    Accepts:
      - sent: spaCy Doc, list of spaCy Token objects, list of strings, or raw string
    Returns:
      - List[Dict] feature dict for each token, produced by calling word2features(tokens, i)
    """
    # ensure word2features and nlp exist
    try:
        word2features  # noqa: F821
    except NameError:
        raise RuntimeError("word2features not defined. Define word2features(sent, i) before using sent2features.")

    # If raw string, process with spaCy to get tokens
    if isinstance(sent, str):
        try:
            doc = nlp(sent)   # nlp should be loaded earlier
            tokens = list(doc)
        except NameError:
            raise RuntimeError("spaCy pipeline 'nlp' not found. Load spaCy or pass tokenized input.")
    else:
        # assume sent is already tokenized (spaCy tokens or plain token list)
        tokens = list(sent)

    # call word2features for each token index
    return [word2features(tokens, i) for i in range(len(tokens))]
```

```
def recipes2features(recipes):
    """
    Apply sent2features over a collection of recipe sentences.
    Accepts:
      - recipes: iterable of sentences (each can be a string or token list/doc)
    Returns:
      - List[List[Dict]]: outer list per recipe, inner list of token feature dicts
    """
    all_features = []
    for sent in recipes:
        feats = sent2features(sent)
        all_features.append(feats)
    return all_features
```

6.3 *Convert X_train, X_val, y_train and y_val into train and validation feature sets and labels* [6 marks]

6.3.1 Convert recipe into feature functions by using X_train and X_val [2 marks]

Create ***X_train_features*** and ***X_val_features*** as list to include the feature functions for each recipe present in training and validation sets

```
Number of training sentences processed: 196
Number of validation sentences processed: 84
```

6.3.2 Convert lables of y_train and y_val into list [2 marks]

Create ***y_train_labels*** and ***y_val_labels*** by using the list of y_train and y_val

```
Number of training label sequences: 196
Number of validation label sequences: 84
```

6.3.3 Print the length of val and train features and labels [2 marks]

```
# print the length of train features and labels
print("Length of X_train_features:", len(X_train_features))
print("Length of y_train_labels:", len(y_train_labels))
```

```
Length of X_train_features: 196
Length of y_train_labels: 196
```

```
# print the length of validation features and labels
print("Length of X_val_features:", len(X_val_features))
print("Length of y_val_labels:", len(y_val_labels))
```

```
Length of X_val_features: 84
Length of y_val_labels: 84
```

6.4 *Applying weights to feature sets* [12 marks]

6.4.1 Flatten the labels of y_train [2 marks]

Create ***y_train_flat*** to flatten the structure of nested y_train

```
Total labels after flattening: 7114
```

6.4.2 Count the labels present in training target dataset [2 marks]

Create ***label_counts*** to count the frequencies of labels present in y_train_flat and retrieve the total samples by using the values of label_counts as ***total_samples***

```
Label Counts: Counter({'ingredient': 5323, 'quantity': 980, 'unit': 811})
Total Samples: 7114
```

6.4.3 Compute weight_dict by using inverse frequency method for label weights [2 marks]

- Create ***weight_dict*** as dictionary with label and its inverse frequency count in ***label_counts***
- Penalise ingredient label in the dictionary

```
# Compute class weights (inverse frequency method) by considering total_samples and label_counts
weight_dict = {}
for label, count in label_counts.items():
    weight = total_samples / count   # inverse frequency
    weight_dict[label] = weight
```

```
# penalise ingredient label
for ing_label in ["B-ING", "I-ING"]:
    if ing_label in weight_dict:
        weight_dict[ing_label] *= 1.2   # apply 20% penalty boost

print("weight_dict:", weight_dict)
```

```
weight_dict: {'quantity': 7.259183673469388, 'unit': 8.771886559802713, 'ingredient': 1.336464399774563}
```

6.4.4 Extract features along with class weights [4 marks]

Define a function ***extract_features_with_class_weights*** to work with training and validation datasets and extract features by applying class weights

```
# Apply weights to feature extraction in extract_features_with_class_weights by using parameters such as X (input tokens), y(l
def extract_features_with_class_weights(X, y, weight_dict):
    """
    X : list of input recipe sentences
    y : list of label sequences corresponding to X
    weight_dict : dict containing class weights (inverse frequency)

    Returns:
        feature_list_all : list of feature dicts per token per sentence
        weight_list_all  : list of weights aligned with each token label
    """

    feature_list_all = []
    weight_list_all = []

    # Loop through all sentences and their labels
    for sent, labels in zip(X, y):

        # Extract token-level features for the sentence
        sent_features = sent2features(sent)
        feature_list_all.append(sent_features)

        # Apply class weights to each token label inside the sentence
        sent_weights = [weight_dict[label] for label in labels]
        weight_list_all.append(sent_weights)

    return feature_list_all, weight_list_all
```

6.4.5 Execute extract_features_with_class_weights on training and validation datasets [2 marks]

Create ***X_train_weighted_features*** and ***X_val_weighted_features*** for extracting training and validation features along with their weights by
calling ***extract_features_with_class_weights*** on the datasets

```
Weighted Training Sentences: 196
Weighted Validation Sentences: 84
```

7 Model Building and Training [10 marks]

7.1 *Initialise the CRF model and train it* [5 marks]

Train the CRF model with the specified hyperparameters such as

```python
# initialise CRF model with the specified hyperparameters and use weight_dict
import sklearn_crfsuite

crf = sklearn_crfsuite.CRF(
    algorithm='lbfgs',
    c1=0.5,
    c2=1.0,
    max_iterations=100,
    all_possible_transitions=True
)

# train the CRF model with the weighted training data
crf.fit(
    X_train_weighted_features,
    y_train_labels,
)

print("CRF model training completed.")
```

```
CRF model training completed.
```

7.2 *Evaluation of Training Dataset using CRF model* [4 marks]

Evaluate on training dataset using CRF by using flat classification report and confusion matrix

```python
# evaluate on the training dataset
y_pred_train = crf.predict(X_train_weighted_features)
```

```
Classification Report (Training Data):

              precision    recall  f1-score   support

  ingredient       0.99      1.00      0.99      5323
    quantity       1.00      0.98      0.99       980
        unit       0.98      0.96      0.97       811

    accuracy                           0.99      7114
   macro avg       0.99      0.98      0.98      7114
weighted avg       0.99      0.99      0.99      7114
```

```
Confusion Matrix (Training Data):
```

	ingredient	quantity	unit
ingredient	5307	2	14
quantity	19	961	0
unit	32	0	779

7.3 *Save the CRF model* [1 marks]
Save the CRF model

```
# dump the model using joblib as crf_model.pkl
joblib.dump(crf, "crf_model.pkl")

print("CRF model saved successfully as crf_model.pkl")
```

```
CRF model saved successfully as crf_model.pkl
```

8 Prediction and Model Evaluation [3 marks]

8.1 *Predict and Evaluate the CRF model on validation set* [3 marks]
Evaluate the metrics for CRF model by using flat classification report and confusion matrix

```
# predict the crf model on validation dataset
y_pred_val = crf.predict(X_val_weighted_features)
```

Classification Report (Validation Data):

	precision	recall	f1-score	support
ingredient	0.98	0.99	0.99	2107
quantity	0.99	0.99	0.99	411
unit	0.96	0.91	0.93	358
accuracy			0.98	2876
macro avg	0.98	0.96	0.97	2876
weighted avg	0.98	0.98	0.98	2876

Confusion Matrix (Validation Data):

	ingredient	quantity	unit
ingredient	2094	3	10
quantity	4	405	2
unit	34	0	324

9 Error Analysis on Validation Data [10 marks]

Investigate misclassified samples in validation dataset and provide the insights
9.1 *Investigate misclassified samples in validation dataset* [8 marks]
9.1.1 Flatten the labels of validation data and initialise error data [2 marks]

Flatten the true and predicted labels and initialise the error data as ***error_data***

```
Flattened validation labels length: 2876
Number of mismatches (tokens): 53
Initialized error_data (list) and empty error_df (DataFrame) for storing misclassifications.
```

9.1.2 Iterate the validation data and collect Error Information [2 marks]

Iterate through validation data (X_val, y_val_labels, y_pred_val) and compare true vs. predicted labels. Collect error details, including surrounding context, previous/next tokens, and class weights, then store them in error_data

```python
# iterate and collect Error Information
error_data = []
try:
    y_pred_val
except NameError:
    y_pred_val = crf.predict(X_val_weighted_features)

def _tok_text(t):
    return t.text if hasattr(t, "text") else str(t)

for sent_idx, (x_sent, true_labels, pred_labels) in enumerate(zip(X_val, y_val_labels, y_pred_val)):
    # ensure we have a list of token texts
    if isinstance(x_sent, str):
        doc = nlp(x_sent)
        tokens = list(doc)
    else:
        tokens = list(x_sent)
    token_texts = [_tok_text(t) for t in tokens]

    # align lengths (truncate to shortest to avoid index errors)
    L = min(len(token_texts), len(true_labels), len(pred_labels))

    for tok_idx in range(L):
        true_lbl = true_labels[tok_idx]
        pred_lbl = pred_labels[tok_idx]
        if true_lbl == pred_lbl:
            continue

        token_str = token_texts[tok_idx]

        # get previous and next tokens with handling for boundary cases
        if tok_idx > 0:
            prev_token = token_texts[tok_idx - 1]
        else:
            prev_token = ""    # BOS

        if tok_idx < L - 1:
            next_token = token_texts[tok_idx + 1]
        else:
            next_token = ""    # EOS

        error_data.append({
            "sent_idx": sent_idx,
            "token_idx": tok_idx,
            "token_text": token_str,
            "true_label": true_lbl,
            "pred_label": pred_lbl,
            "prev_token": prev_token,
            "next_token": next_token
        })
```

9.1.3 Create dataframe from error_data and print overall accuracy [1 marks]

Change error_data into dataframe and then use it to illustrate the overall accuracy of validation data

Error DataFrame:

	sent_idx	token_idx	token_text	true_label	pred_label	prev_token	next_token
0	2	34	few	ingredient	quantity	Leaves	
1	5	18	cloves	ingredient	unit	3	garlic
2	5	21	Spoon	unit	ingredient	big	oil
3	6	7	cloves	unit	ingredient	seeds	garlic
4	6	35	pieces	ingredient	unit	small	1/4

Overall Accuracy on Validation Data: 0.9815716272600834

9.1.4 Analyse errors by label type [3 marks]

Analyse errors found in the validation data by each label and display their class weights along with accuracy and also display the error dataframe with token, previous token, next token, true label, predicted label and context

Per-label error summary (validation):

label	total	errors	accuracy	class_weight
ingredient	2107	13	0.993830	1.336464
quantity	411	6	0.985401	7.259184
unit	358	34	0.905028	8.771887

9.2 *Provide insights from the validation dataset* [2 marks]

The CRF model performs strongly on the validation dataset with an overall token-level accuracy of 98.16%. The classification report and confusion matrix highlight a high level of consistency across all three entity types: ingredient, quantity, and unit, with especially strong performance on high-support classes.

1. Ingredient Labels
    - Highest accuracy: 99.38%
    - Very few mistakes (13 out of 2107 tokens).
    - Most ingredient errors involve tokens like "cloves" or "pieces" incorrectly predicted as unit due to their dual semantic usage in cooking contexts (e.g., “3 cloves garlic”).
    - Since "cloves" and "pieces" frequently behave like units in natural language, misclassification is understandable.

2. Quantity Labels
    - Accuracy of 98.54%.
    - Errors mostly occur when the model confuses quantity tokens such as “few” or “3” with ingredient or unit labels, especially when the surrounding context is ambiguous.
    - Example: “few Leaves” → “few” predicted as quantity instead of ingredient due to misleading context structure.
3. Unit Labels
    - Lowest accuracy: 90.50%
    - Units account for 34 of the 53 total errors, making them the weakest component.
    - Units like “spoon”, “cloves”, “pieces” are often confused with ingredient labels.
    - These errors typically occur when:
    - The unit is not part of the predefined unit_keywords.
    - The unit token looks like a noun that could also be an ingredient.
    - The model lacks contextual cues from noisy or incomplete sentences.
4. General Misclassification Patterns
    - Ingredient ↔ Unit swaps are the most common type of error (especially for multi-use nouns).
    - Tokens with weak preceding quantities (e.g., “big oil”, “seeds garlic”) cause confusion.
    - Some tokens like “few”, categorized as quantity keywords, are labeled as ingredients depending on the dataset annotation rules.
    - Sentences with irregular formatting or missing delimiters increase error frequency.
5. Class Weight Impact
    - Unit class had the highest class weight (~8.77), reflecting its rarity and helping the model learn better.
    - Despite high weight, the model still finds unit detection challenging, suggesting:
    - Need for broader unit keyword list.
    - Additional features for noun-based units.
    - Possibly richer context windows or dependency-based features.

10 Conclusion (Optional) [0 marks]

The CRF model demonstrates strong performance on the validation dataset with an overall accuracy of 98%, effectively identifying ingredient, quantity, and unit entities. Most errors arise from ambiguous tokens such as cloves, pieces, and spoon, which can function as both units and ingredients depending on context. Unit detection remains the weakest category, suggesting the need for an expanded unit vocabulary and richer contextual features. Overall, the model generalizes well and provides reliable entity extraction for recipe text.