Nguyen Ngoc Gia Thinh

Student ID: 103809954

COS30018

Task B2: Weekly Report

Function will allow you to specify the start date and the end date for the whole dataset as inputs.

First, I will define everything related to the task in a function call “load_and process_data”.

```
def load_and_process_data(
    company: str, #Stock ticker symbol
    start_date: str, #Start date range for downloading stock data
    end_date: str, #End date range for downloading stock data
    features: list = ['Open', 'High', 'Low', 'Close', 'Volume'], #List of 
columns to be used as input features
    handle_nan: str = 'drop', #Determines how to handle missing values
    split_method: str = 'ratio', #Ratio for splitting training and test data
    train_ratio: float = 0.8,
    scale_data: bool = True, #If true, scales the data using MinMaxScaler
    save_local: bool = True, #Saves the dataset locally if True
    local_path: str = "data.csv" #Path where the data is saved
):
```

Dealing with NaN in data

At the start of the code, I will always check if there is any data exist in the local files. If not I will download the stock data from Yahoo finance and save in to my local as data.csv files.

```
if os.path.exists(local_path):
        df = pd.read_csv(local_path, index_col=0, parse_dates=True)
    else:
        df = yf.download(company, start=start_date, end=end_date)
        if save_local:
            df.to_csv(local_path)
```

After that I set the code to only fetching the features that I define earlier which is Open, High, Low, Close and Volume.

```
df = df[features]
```

Then, I will check if the data set is empty or not because some company change their logo (ticker symbol) like FB change to META. So if you trying to run the code with the symbol is FB, the data will be empty.

```
if df.empty:
        raise ValueError("Error: Stock data is empty. Check the ticker symbol
and date range.")
```

Also, sometime the data is missing, so I am using drop and fill function to fix the issues if it happens. However, in stock market, that rarely will happen but incase I will add them in.

```
if handle_nan == 'drop':
        df.dropna(inplace=True)
    elif handle_nan == 'fill':
        df.fillna(method='ffill', inplace=True)
```

Splitting the data into 80-20 ratio for train and test data.

```
#Splits the dataset into training (80%) and testing (20%) sets.
    if features.shape[0] > 1:
        X_train, X_test, y_train, y_test = train_test_split(features, target,
train_size=train_ratio, shuffle=False)
    else:
        raise ValueError("Not enough data to split. Check the dataset size.")

    return X_train, X_test, y_train, y_test, scalers if scale_data else None
```

Normalize the feature values between 0 and 1

```
#Normalize the feature values between 0 and 1 to improve neural network
performance
    scalers = {}
    if scale_data:
        feature_scaler = MinMaxScaler()
        features = feature_scaler.fit_transform(features)
        scalers['features'] = feature_scaler

        target_scaler = MinMaxScaler()
        target = target_scaler.fit_transform(target)
        scalers['target'] = target_scaler

    #Reshape the feature set into 3D format for LSTM input
    if features.shape[0] > 0 and features.shape[1] > 0:
        features = np.reshape(features, (features.shape[0], features.shape[1], 1))
    else:
        raise ValueError("Error: No data available after preprocessing.")
```