Predicting the Weights of Neural Networks using Meta-learning

---

Metadata Collection

Data Collection

- Loop over all dataset + group combinations.
  - **Tourism**: *Quarterly* & *Monthly*
  - **M3**: *Quarterly* & *Monthly*
  - **Gluonts_m1**: *Quarterly* & *Monthly*
- Load the dataset and its metadata
  - `lags`, `frequency`, `horizon`.
- Split the data into training and testing sets.

Baseline Modeling

- Train a **Seasonal Naive** model using training data.
- Predict and merge results with test data.
- Compute baseline **sMAPE** for future comparison.

Hyperparameter Search

- Generate combination of hyperparameters:

```
hyperparameters = {
    "hidden_size": [8, 16, 32, 64],
    "max_steps": [500],
    "num_layers": [3],
    "learning_rate": [1e-3, 5e-4, 1e-4],
    "batch_size": [16, 32, 64],
    "scaler_type": ['identity', 'standard', 'robust', 'minmax'],
    "seed": [42, 123, 456, 789, 1011]
}
```

Model Training & Evaluation

- For each hyperparameter set:
  - Train a MLP model with **custom Callback**.
  - Evaluate the model using **sMAPE**, **MSE**, **MAE**, and **R^2**.
  - Compare with the Seasonal Naive baseline.

- Store metadata and scores.

Training Callback

- Evaluate weight matrices and get model variance:
  - At the **start of training**: `on_train_start`
  - At the **end of training**: `on_train_end`
  - **During training**: `on_train_batch_end`
    - Custom training checkpoits: `[10, 25, 50, 100, 200, 300, 400, 500]`

Matrix Evaluation

- For each **weight tensor**:
  - Collect basic stats:
    - `shape`, `mean`, `std`, `min`, `max`, `var`, etc.
  - Calculate matrix norms:
    - `frobenius_norm` and `spectral_norm`
  - Attempt power-law distribution:
    - `alpha` and `weighted_alpha`

Train Metamodel

Model Configuration

- For **Classification**:
  - Use `XGBRFClassifier`.
  - Evaluate with: Accuracy, ROC AUC, Log Loss, F1 Score.
- For **Regression**:
  - Use `XGBRFRegressor`.
  - Evaluate with: MAE, MSE, R^2, Pearson, Kendall, Spearman.

Stagewise Evaluation

- **Iterative stage evaluation**:
  - Gradually add weight stats from:
    - `start`, `step_10`, `step_25`, `step_50`, ..., `step_500`.
- Results saved for each stage.
- Summary stored in `stagewise_summary.csv`.

Cross-Validation

- **Perform cross-validation** with `GroupKFold` for each `DATASET_GROUP`.
- **Fit model** and predict for each fold.
- Collect, for each fold:

- **evaluation metrics**;
- **classification reports**;
- **feature importances**.