Modularización del script de Coffee Forecasting

Estructura creada en `src/model/` con módulos:

- `config.py`: parámetros y flags globales (incluye hiperparámetros LSTM y opciones Tweedie/Poisson).
- `paths.py`: resolución robusta de rutas para `data/` y `results/`.
- `mlflow_utils.py`: inicialización y logging de métricas/artefactos en MLflow (tracking local bajo `data/results/mlruns`).
- `data.py`: carga y enriquecimiento del dataset (merge index_1, calendario rico, agregados de negocio, lags y rollings causales), snapshot de datos.
- `metrics.py`: métricas (MAE, RMSE, MAPE, sMAPE, cobertura) y agregación por horizonte/overall.
- `splits.py`: función `rolling_origins` para backtesting.
- `baselines.py`: Naïve, sNaïve7 y MA7.
- `lgbm_direct.py`: entrenamiento directo multi-horizonte con LightGBM y cuantiles p10/p90.
- `prophet_model.py`: Prophet con regresores opcionales.
- `sarimax_model.py`: SARIMAX con fallback sin exógenas y bandas al 80% (p10–p90).
- `lstm_direct.py`: LSTM multi-salida (HORIZON) con MC Dropout para bandas.
- `results.py`: normalización de columnas y guardado de métricas/forecasts.
- `main.py`: orquestador que reproduce el flujo del script original.

Cómo ejecutar

1. Ubica este paquete dentro del repo (por ejemplo, en la raíz del repo).
2. Asegúrate de tener la estructura:

```
<repo>/
  data/
    processed/
      coffee_ml_features.csv
      index_1.csv        (opcional)
```

3. Instala dependencias equivalentes a las del script original.
4. Ejecuta:

```
python -m src.model.main
```

5. Revisa resultados en `data/results/` y el tracking local de MLflow en `data/results/mlruns`.