High Integrity Systems Project Time Series Analysis

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Assignment 4

Hellyben Shah

November 19, 2024

1 Summary of Modern Time Series Forecasting with Python- Chapter 9

Ensemble methods in time-series forecasting involve combining multiple individual forecasts to create a single, more accurate prediction. This chapter explores various strategies for combining forecasts and introduces the concept of stacking or blending.

After generating forecasts by many techniques, ultimately we would be needing a single forecast which can be either selected by choosing the best among all or combining all the forecast.

$$Y = F(Y_1, Y_2, ...Y_N) \tag{1}$$

Here, F is the function that combines N forecasts.

Best fit

This strategy involves selecting the best-performing forecast for each time series based on validation metrics. It's widely used in automated forecasting software and is often referred to as the "best fit" forecast.

Algorithm

1. Evaluate each forecast model using a validation dataset.
2. For each time series, select the forecast from the model that performed best in validation.
3. Apply this selected model to generate forecasts for the test dataset.

Surprisingly, the Best Fit strategy underperformed compared to the best individual model (LightGBM) in terms of Mean Absolute Error (MAE).

Drawbacks

1. **Assumption Flaw**: The strategy assumes that the model performing best in the validation period will also perform best in the test period. This assumption often doesn't hold due to the dynamic nature of time series data.
2. **Forecast Instability**: In a live environment with frequent retraining, this approach can lead to significant instability in forecasts. The selected model for a given time series may change frequently, resulting in inconsistent predictions from week to week.
3. **Impact on Downstream Actions**: The instability in forecasts can negatively affect any downstream decision-making or actions that rely on these predictions.

Measures of central tendancy

Mean and Median Ensembles

The simplest ensemble methods involve taking the mean or median of multiple forecasts. However, these methods may not always outperform the best individual model, especially when some models perform significantly worse than others. This underperformance may be due to including poorly performing methods (e.g., Theta and FFT) without considering validation data.

They are independent of validation metrics, which prevents overfitting but may include poor-performing model.

Trimming and Skimming

More advanced techniques include:

- Trimming: Discarding the worst-performing models in the ensemble
- Skimming: Selecting only the best few models in the ensemble

These methods, while effective, can be subjective and challenging to implement when dealing with numerous models.

Optimization-Based Ensemble Methods

The problem of selecting the best combination of models can be framed as a combinatorial optimization problem. The goal is to minimize a loss function (e.g., Mean Absolute Error) by selecting the optimal weights for each base forecast.

Mathematically, this can be expressed as:

$$Y = \arg\min_{w} \mathcal{L}\left(\frac{1}{\sum_{i=1}^{N} w_i} \sum_{i=1}^{N} w_i \times \hat{Y}_i, Y\right) \tag{2}$$

Where:

- $\mathcal{L}$ is the loss function
- $w_i \in [0, 1]$ are binary weights for each base forecast
- $\hat{Y}_i$ are the base forecasts
- Y are the observed values

The most straightforward solution is to find w, which minimizes this function on validation data. But there are two problems with this approach:

1. The number of possible combinations increases exponentially with the number of base forecasts.
2. Selecting the global minimum in the validation period may lead to overfitting.

Heuristic-Based Solutions

Simple Hill Climbing for Forecast Combination

Simple hill climbing is a greedy optimization algorithm used for combining forecasts in time series analysis.

Algorithm Overview

- Builds a solution stage by stage, selecting a local optimum at each step
- Aims to find the best combination of forecasts by "climbing" the objective function surface

Process

1. Initializes with the best-performing individual candidate
2. Iteratively evaluates all remaining candidates
3. Adds the candidate that most improves the objective function
4. Continues until no further improvement is possible

Performance

- Often outperforms individual models and simpler ensemble techniques

Limitations

- Runtime issues with large numbers of candidates
- Short-sightedness in always selecting the immediate best option
- Inability to remove previously added candidates

Applications

- Useful for combining multiple forecasts in time series analysis
- Similar approach used in decision trees and gradient-boosted trees

Stochastic Hill Climbing for Forecast Combination

Stochastic hill climbing is an optimization algorithm for combining forecasts in time series analysis.

Core Concept

- Randomly selects candidates instead of evaluating all options
- Adds a candidate to the solution if it improves the current best

Algorithm Steps

1. Initialize with a starting solution (random or best-performing model)
2. Set the initial objective function value

3. Iterate for a specified number of times:

    - Randomly select a candidate

    - Evaluate the candidate

    - If better than current best, update the solution

Advantages over Simple Hill Climbing

- Addresses runtime issues by not evaluating all combinations
- Partially solves short-sightedness through randomness
- Can potentially avoid local optima

Limitations

- Still only selects solutions better than the current best
- May not always outperform greedy approaches

Performance

- Generally better than mean, median, and best fit ensembles
- May not surpass simple hill climbing in all cases

Applications

- Useful for combining multiple forecasts in time series analysis
- Effective when dealing with numerous models to combine

Stochastic hill climbing offers a balance between exploration and exploitation in forecast combination, making it a valuable tool in ensemble methods for time series forecasting.

Simulated Annealing for Forecast Combination

Simulated annealing is an optimization technique for combining forecasts in time series analysis, inspired by the physical process of annealing in metallurgy.

Core Concept

- Starts with a high "temperature" to encourage exploration
- Gradually decreases temperature to refine the solution
- Helps avoid getting stuck in local optima

Algorithm Steps

1. Initialize with a starting solution
2. Set initial temperature

3. Iterate for a specified number of times:

   - Randomly select a candidate

   - Evaluate the candidate

   - Accept improvements

   - Sometimes accept worse solutions based on a probability function

   - Decrease temperature over time

Key Parameters

- Maximum number of iterations
- Maximum temperature
- Temperature decay rate

Advantages

- Addresses runtime issues by not evaluating all combinations
- Solves short-sightedness by accepting suboptimal solutions early on
- Balances exploration and exploitation

Implementation

- Available in `src.forecasting.ensembling.py` as the `simulated_annealing` function
- Allows setting starting and ending probabilities for accepting worse solutions

Performance

- Generally outperforms stochastic hill climbing
- Effective at avoiding local optima

Limitations

- Still a forward-only procedure
- Setting the right temperature is crucial and can be challenging

Stacking and Blending in Time Series Forecasting

Concept of Stacking

- Uses machine learning to combine predictions from base models
- Trains a "meta-model" on the predictions of base learners
- Typically performs equal to or better than individual base models

Advantages

- Combines diverse strong learners (unlike bagging or boosting)
- Captures different properties of the problem from various models

Stacking vs. Blending

- Stacking: Uses entire training dataset with out-of-sample predictions so can work better
- Blending: Splits data into train and holdout sets

Implementation Steps For Stacking

1. Split training data into k parts
2. Train base models on k-1 parts, predict on kth part
3. Train meta-model on out-of-sample predictions

For Blending

1. Split data into train and holdout sets
2. Train base models on training set, predict on holdout
3. Train meta-model on holdout predictions

Considerations

- Stacking assumes independent and identically distributed (iid) data, which may not hold for time series
- Blending may be better for time series with changing distributions

Best Practices

- Use simple models (e.g., linear regression) for meta-models
- Consider regularized models for implicit base model selection
- Use separate validation set or cross-validation for model selection

Performance

- Can outperform simple averaging methods
- Choice of metric and combination model is crucial

Advanced Techniques

- Feature-Based Forecast Model Averaging (FFORMA)
- Classifier-based approach for predicting best base learner

Stacking and blending offer powerful ways to combine multiple forecasts, potentially improving overall prediction accuracy in time series analysis.