Model Evaluation
Point and Density Forecasts

Romain Lafarguette, Ph.D.   Amine Raboun, Ph.D.

Quants & IMF External Experts

romainlafarguette.github.io/   amineraboun.github.io/

Singapore Training Institute, 19 April 2023

Model Parameters and Hyper-Parameters

OLS Example

$$y_{t+?} = \alpha + \beta_1 X_{1,t} + \cdots + \beta_K X_{k,t} + \epsilon_t$$

- α and $\beta_1, \ldots \beta_k$ are the parameters of the models. Once we have decided the specification (the DGP) then we can estimate the parameters

- However, there are many other choices:
  - Which variables $X_1 \ldots X_k$ to include? Do we need all of them? Tradeoff between exhaustive model and parsimonious model. Do we want to add-non linear effects like X_1^2 Do we want to add lags?
  - Do we want to control for some time periods, structural breaks, etc.? Forecast horizon?
  - Etc.

- These are called the **hyper-parameters**: they determine the DGP we want to estimate

Estimating Parameters and Optimizing Hyper-Parameters

- Parameters are linked to a specific data generating process (DGP): they can be estimated directly in the data, using standard approaches (regressions, maximum likelihood, methods of moments, etc.)

- Hyper-parameters represent different data generating process, we can not estimate them directly

- But we can try to fit different models with different hyper-parameters and **compare these models**

- How to compare models?
  - Intuitively, one might think that in-sample performance (such as R2, loglikelihood, AIC, BIC) would work the best
  - But...

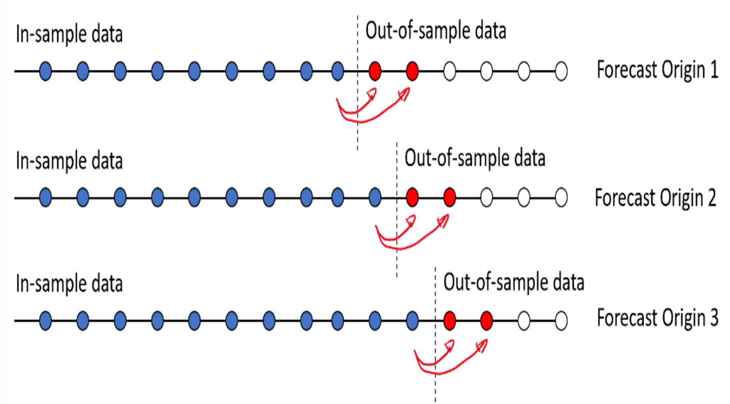
Fitting and Forecasting

Be careful

A model that fits the data well (in sample) might not necessarily forecast well

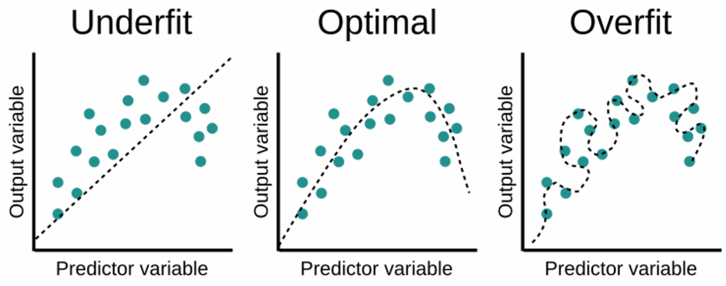
- A perfect in-sample fit can always be obtained by using a model with with enough parameters
- Over-fitting a model to data is just as bad as failing to identify a systematic pattern in the data
- Need to split the model between
- The test set must no be used to *any* aspect of model development or calculation of forecasts
- Forecast accuracy is only based on the test set

Out of Sample Concept



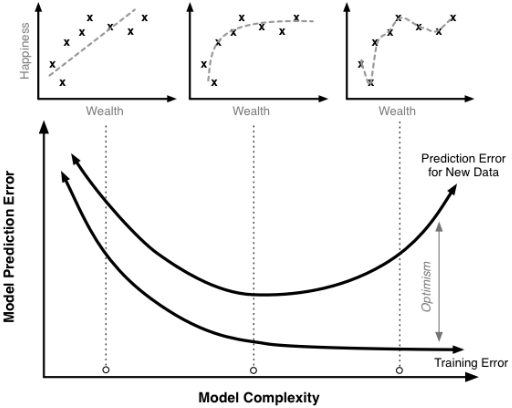
Source: *Author*

Underfit, Optimal, Overfit: Intuition



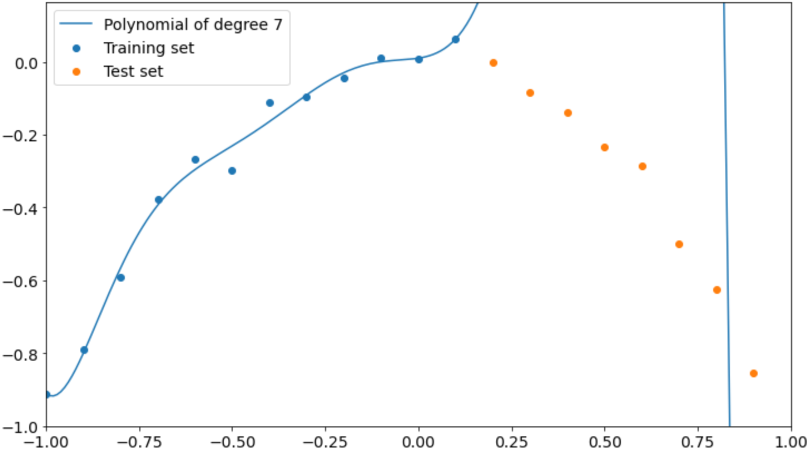
Source: *towardsdatascience*

Underfit, Optimal, Overfit and Model Complexity



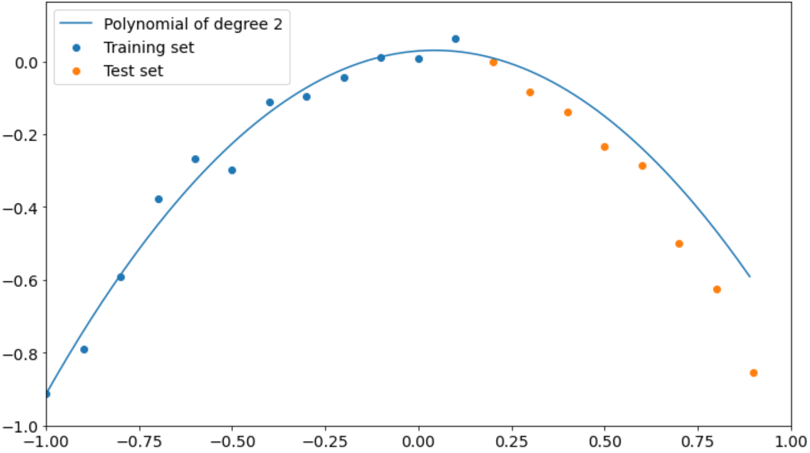
Source: *Scott Fortmann-Roe*

Out of Sample Example: Overfit



Source: *towardsdatascience.com/an-example-of-overfitting-and-how-to-avoid-it*

Out of Sample Example: Correct Fit

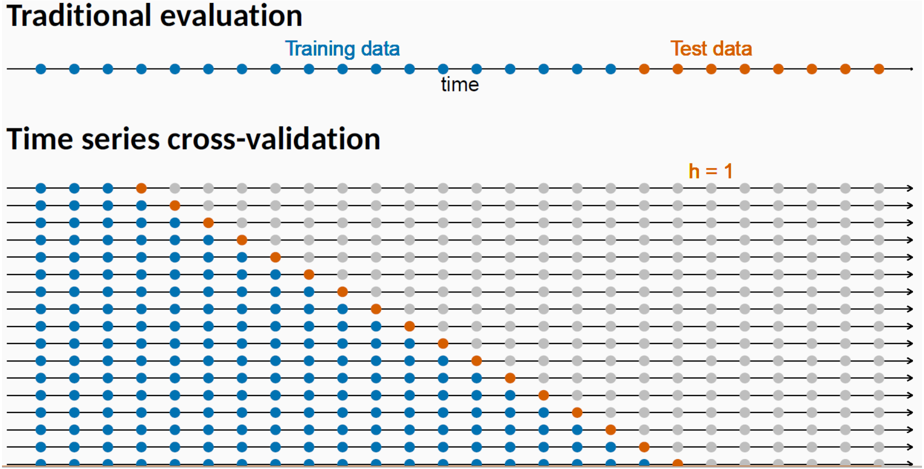


Source: *towardsdatascience.com/an-example-of-overfitting-and-how-to-avoid-it*

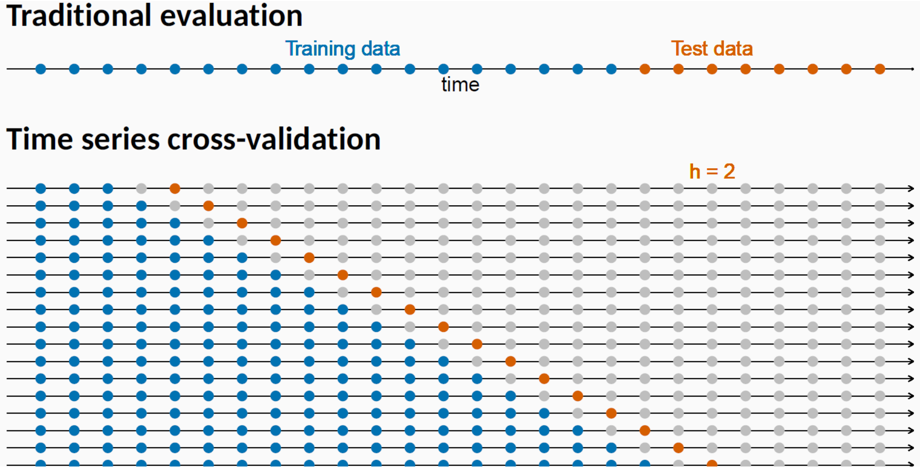
Choosing Hyper-Parameters

- The key aspect is to **optimize the hyper-parameters out-of-sample**
- How well does the model performs **"in real conditions"** if we would have estimated it in the past?
- Avoid overfitting the model, else we would end-up with models only good at explaning the past
- However, because we are working with time series, we need to follow a certain process

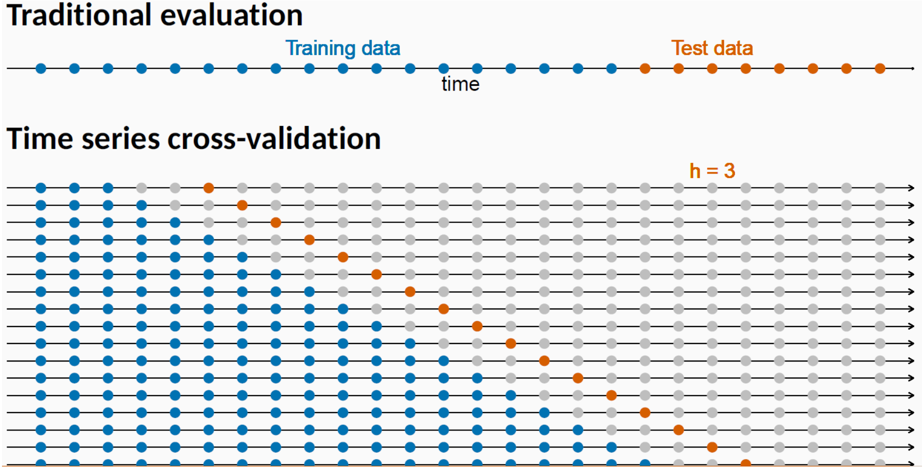
Time Series Cross-Validation



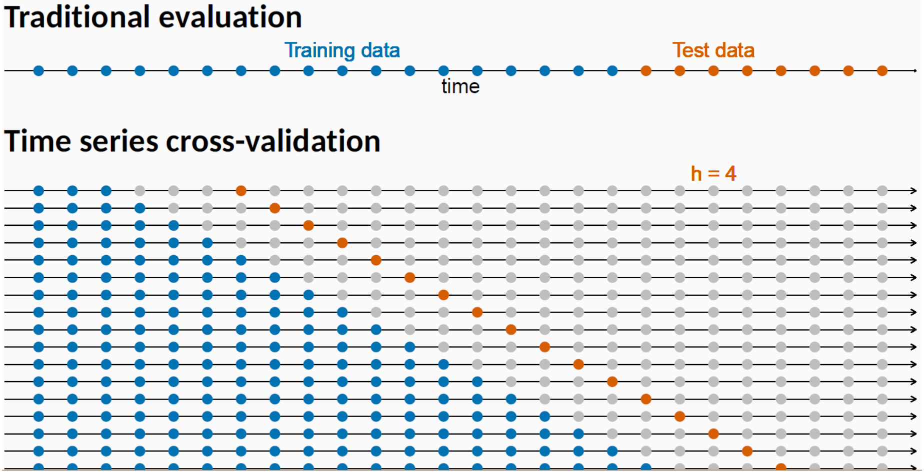
Time Series Cross-Validation



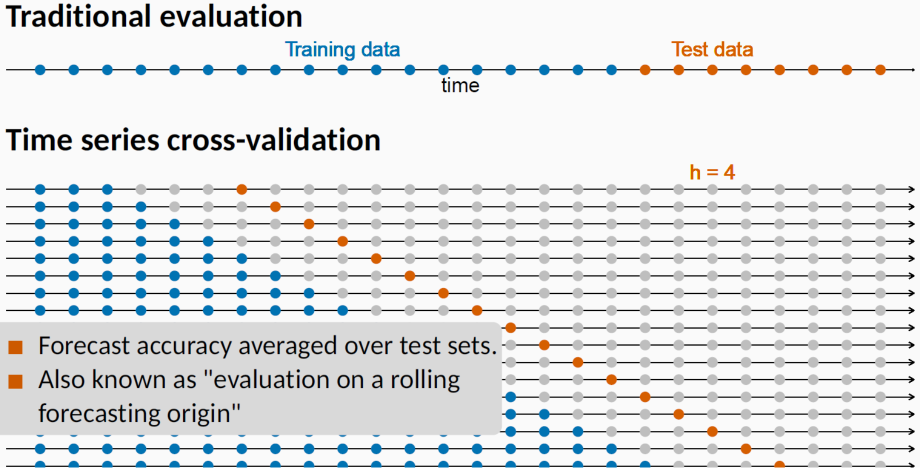
Time Series Cross-Validation



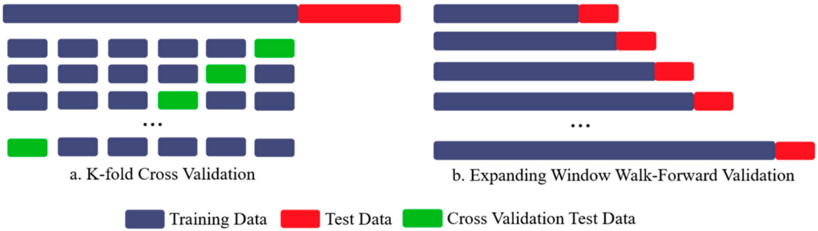
Time Series Cross-Validation



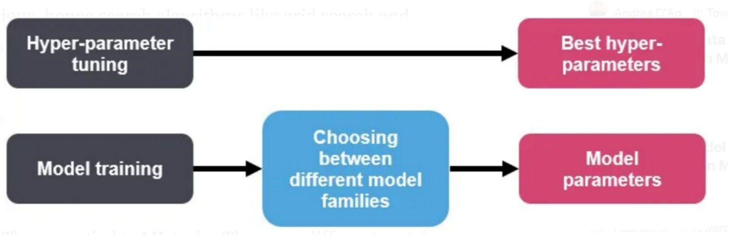
Time Series Cross-Validation



K-Fold Cross-Validation

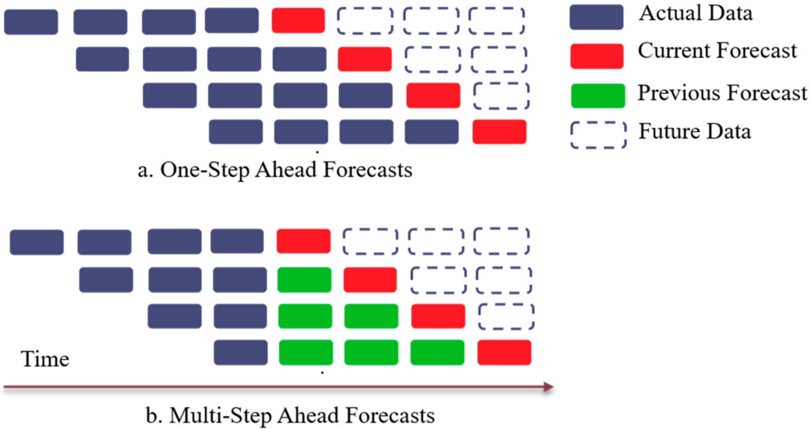


Parameters and Hyper-Parameters Process



Source: *Author*

Time Series Validation is Different from the Forecasting Horizon



Forecast Errors

- Evaluating point forecasts are relatively straightforward
- Ex-post (after the realization happened), we observe:
  - The true value y_{T+h} that has been realized
  - The expected value $\hat{y_{T+h}}$ that has been generated before, in time t

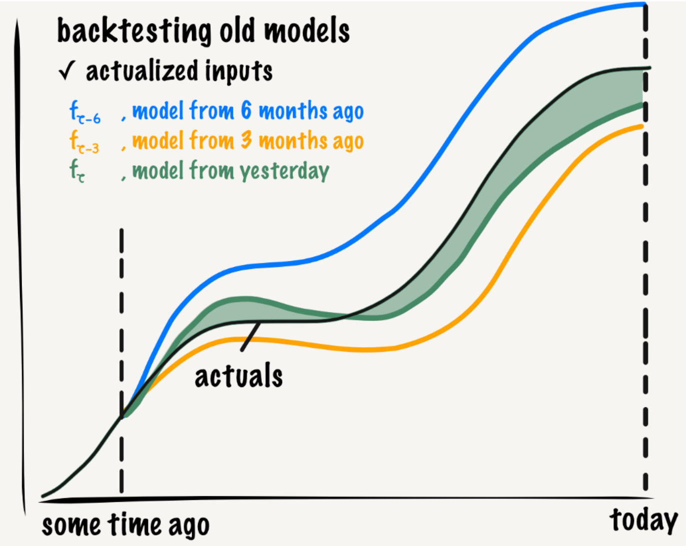
Definition: Forecast Errors

A forecast error is the ex-post difference between an observed value and its forecast

$$e_{T+h} = y_{T+h} - \hat{y}_{T+h} | Y_T, \ldots, Y_1$$

- Forecast evaluation metrics represent different variations on how to summarize the e_{T+h}
  - Are the forecast errors small on average?
  - Have we observed infrequent but large forecast errors (outliers)?
  - Are the forecast errors evenly distributed across the distribution of y? etc.

Updating Model



Source: *www.multithreaded.stitchfix.com*

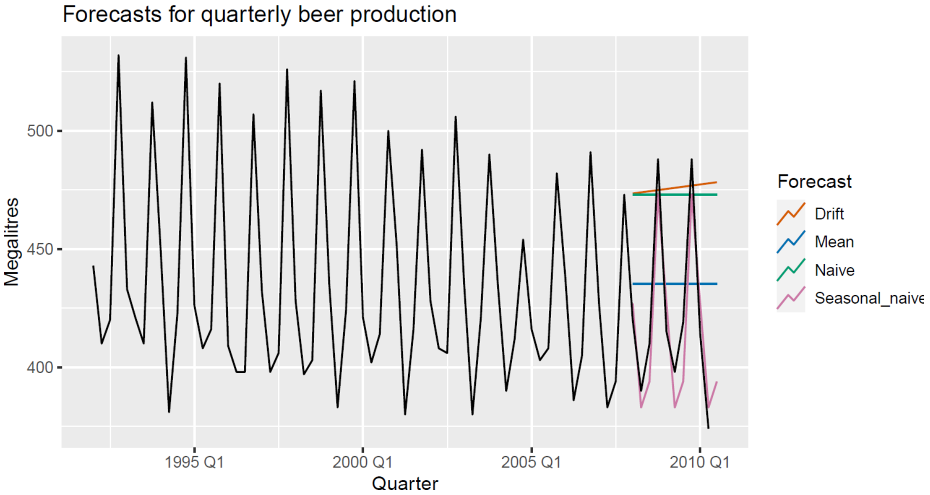
Forecast Errors with Train/Test Sets

Out of Sample

Measuring **accuracy** should be done out of sample. In-sample metrics inform on the how well the model **fits** the data

- The conditional set $Y_T, \ldots, Y_1$ should only be taken from the training dataset
- The true value y_{T+h} is taken from the test set
- Unlike residuals, forecast errors on the test involve multi-step forecasts
- These are the **true** forecast error, as the test data is not used to compute $\hat{y}_{T+h}$

Example: Forecasting Beer Production



Measures of Forecast Accuracy

Main Metrics

- **MAE**: mean absolute errors $\frac{1}{S}\sum_{s \in S}|e_{s,T+h}|$
- **MSE**: mean squared errors $\frac{1}{S}\sum_{s \in S}(e_{s,T+h})^2$
- **MAPE**: mean absolute percentage errors $\frac{1}{S}100 * \sum_{s \in S}\frac{|e_{s,T+h}|}{|y_{s,t+h}|}$
- **RMSE**: root mean squared errors: $\sqrt{\frac{1}{S}\sum_{s \in S}(e_{s,T+h})^2}$

With:

- y_{T+h}: T+h observation, h being the horizon (h = 1, 2, ..., H)
- $\hat{y}_{T+h|T}$: the forecast based on data up to time T
- $e_{T+h} = y_{T+h} - \hat{y}_{T+h|T}$: The forecast errors
- S is the testing sample

Scaling

- MAE, MSE and RMSE are all **scale dependent**
- MAPE is scale independent but is only sensible if $y_t >> 0 \qquad \forall\ t$
- **Most commonly used: Time Cross-Validation with the lowest RMSE**

Nemenyi Test

- We can rank the model by RMSE (or another metric), but are the RMSE significantly different?

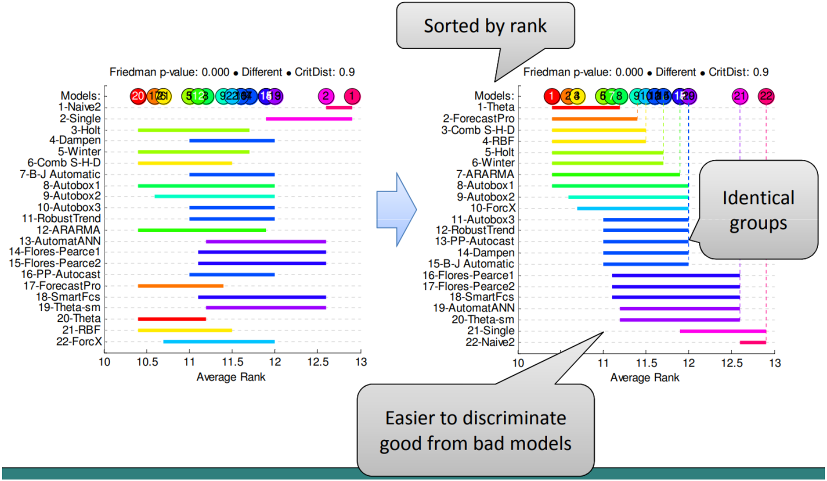
- Maybe Model 1 can have a lower RMSE than Model 2, but the difference in RMSE is non-significant

- In which case, we could pool the two models together

- Use a non-parametric test to test the hypothesis of equal RMSE, with the test statistic:

$$r_{\alpha,K,N} \approx \frac{q_{\alpha,K}}{\sqrt{2}}\sqrt{\frac{K(K+1)}{6N}}$$

Nemenyi Test in Practice



Source: *Nikolaos Kourentzes*

Aleatoric vs. Epistemic Uncertainty

- Fundamentally, there are two sources of uncertainty we want to quantify
  1. **Aleatoric Uncertainty**: Uncertainty because the model can not represent fully the reality
  2. **Epistemic Uncertainty**: Uncertainty because future reality is random, uncertain

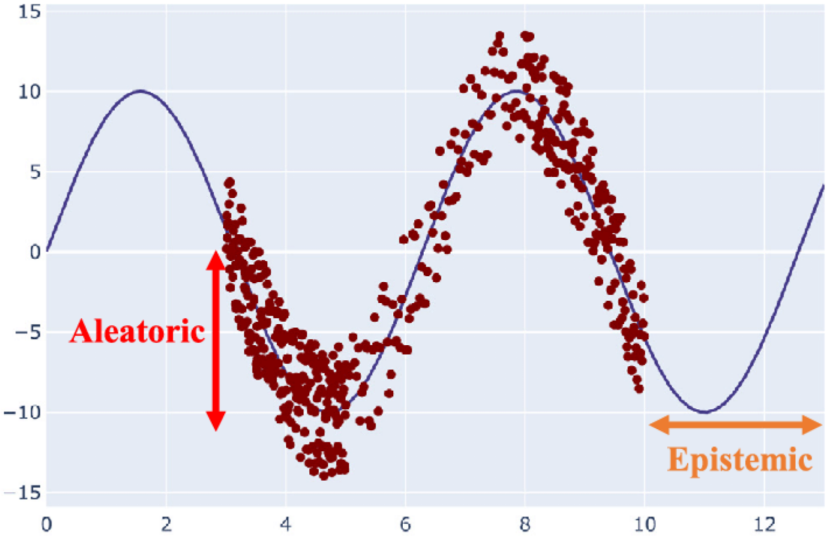
- Point forecasts evaluation measure the aleatoric uncertainty

- Yet they don't inform on the uncertainty of the future realizations.

  - Point forecasts evaluation only inform about the uncertainty of the point estimate (e.g. the uncertainty about the future values of the mean)

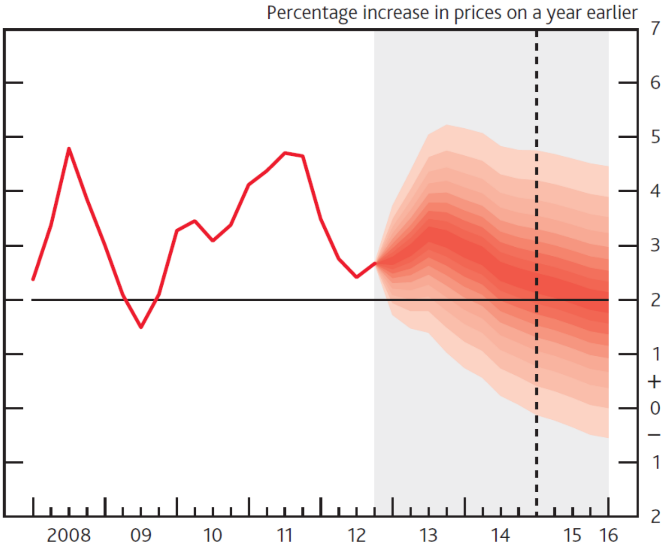
- If we want to "quantify" epistemic uncertainty, we need to use a density forecast

Aleatoric vs Epistemic Uncertainty



Source: *www.researchgate.net*

Bank of England Fanchart



Source: *Bank of England Fan Chart*

Challenges

- At the difference of point forecasts, density forecasts are never observed
  - We only observe **one** realization of the density
- Hence, for evaluating the quality of the density forecasts, we need to use specific tools
- The **model specification**: is my model "neutral", not over-optimistic, not over-pessimistic?
  - Use a **Probability Integral Transform (PIT) test**
- The **model performance**: attributing high *ex-ante* performance to *ex-post* realizations
  - Use **logscores** and asymmetric logscores

Probability Integral Transform Test (PIT)

Intuition

The forecasted quantiles from a correctly specified model should appear as frequently as their realizations. For instance, the true values should occur less than 10% of the 10th quantile

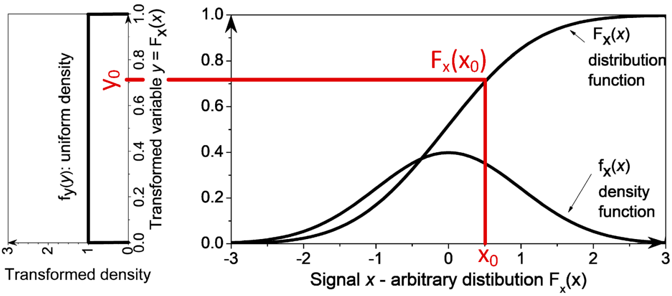
- **Pessimistic model**: if the true values below the forecasted 10th percentile appear significantly more than 10% of the time
- **Optimistic model**: if the true values below the forecasted 10th percentile appear significantly less than 10% of the time

- To quantify this approach, the PIT Test uses the concept of the probability integral transform

- A PIT is simply the evaluation of the cdf of a random variable (F_x) on its own realizations (X_t); the random variable $Y = F_X(X)$ should be uniformly distributed

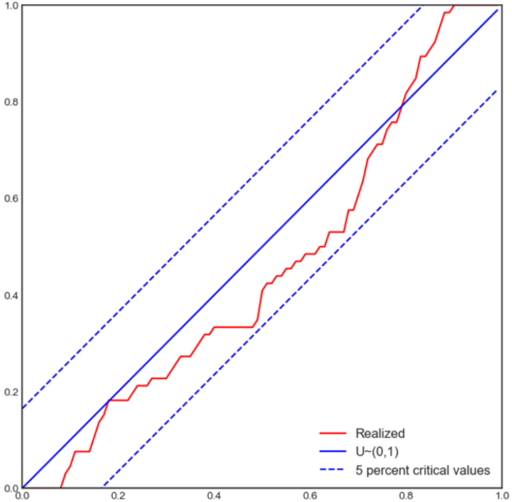
Probability Integral Transform

Figure 1. Probability integral transform. The random variable *x* with the density function $f_X(x)$ is transformed into the uniformly distributed random variable $y = F_X(x)$, with uniform density $f_y(y) = 1$.



Source: *Entropy*

Probability Integral Transform



Source: *Lafarguette (2019)*

Testing for the PIT

- It is possible to test for the specification of the model looking at the distance between the theoretical line of 45 degrees
- However, there are always some randomness in the data: at which point the deviation becomes significant?
- Use the confidence interval computed by ▸ Rossi and Sekhposyan (2019)
  - If the distribution crosses the confidence bands: the distribution is misspecified at this quantile

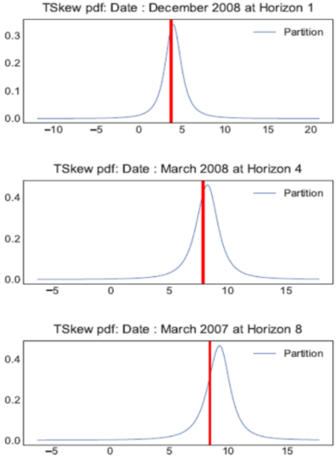
Scoring Tests

- PIT test answers the question: "is my model well specified"?
- But it doesn't inform about the performance. If two models are well specified, how can we distinguish between them?
- Idea: score them based on their *ex-post* performance of their *ex-ante* forecasts

Intuition

- Idea: what was the *ex-ante* probability of the *ex-post* realization?
- Scores are usually taken in log-form: $S\left[\hat{f}_t(y_{t+h})\right] = \log\left(\hat{f}_t(y_{t+h})\right)$

Ex-Ante Probability and Ex-Post Realizations



Source: *Lafarguette (2019)*

Tests for Equal Predictive Ability using Logscores

- A logscore is a relative metric, for a single model, it doesn't inform (at the difference of PIT tests)

- However, the difference of logscores between models informs whether a model performs better than another one and should be preferred

- Need to assess whether the difference is significant if we want to test a model $\hat{f}$ against another one $\hat{g}$

- $d^*_{t+h} = \log\left(\hat{f}_t(y_{t+h})\right) - \log\left(\hat{g}_t(y_{t+h}\right)$ $\bar{d^*}_{m,n} = \frac{1}{n}\sum_{t=m}^{T-1} d^*_{t+1}$

- Use the test of equal predictive ability via a Diebold-Mariano metric (1995)

$$t_{m,n} = \frac{\bar{d^*}_{m,n}}{\sqrt{\hat{\sigma}^2_{m,n}/n}} \xrightarrow[n]{d} \mathcal{N}(0,1)$$

Asymmetric Logscores

- The simple difference provides information about how models performs "on average"
- However, density forecasts are especially useful to inform about risks
- Hence, it makes sense to use **asymmetric logscores** to **test the performance in certain parts of the forecasted distribution**, especially on the tails

$$S^A(\hat{f}_t, y_{t+1}) = \mathbb{1}\left(y_{t+1} \in A_t\right) \log \hat{f}_t(y_{t+1}) + \mathbb{1}\left(y_{t+1} \in A_t^c\right) \log \left(\int_{A_t^c} \hat{f}_t(s) \mathrm{d}s \right)$$

Summary: Model Evaluation

- To evaluate the performance of a model, it is crucial to evaluate its **out-of-sample performances** using **train and test samples**
- The evaluation of a **point forecast**, for instance the mean, can be evaluated from the **forecasting errors**, using different metrics: RMSE, MAE, MAPE, etc.
- The evaluation of a density is more complicated:
  - To know if the density forecast is **properly specified**, use a **PIT test**
  - To assess the **accuracy** of the model, use a **logscore** or an **asymmetric logscore**
  - Note that other approaches, for instance based on **entropy**, exist: they try to minimize the amount of **information loss** between a density forecast and the true distribution

Application : Optimizing a GARCH model

Reminder: GARCH specification

$$
\begin{aligned}
r_{t+1} &= f(X_t) + \epsilon_t \qquad \text{(drift)} \\
\epsilon_t &= \sigma_t e_t \\
\sigma_t &= \omega + \alpha \epsilon_{t-1}^2 + \beta \sigma_{t-1}^2 \qquad \text{(volatility)}
\end{aligned}
$$

- There are many possible choices for the models on the mean/drift, the conditional variance or the distribution of the perturbations
- We would like to choose the most appropriate ones, using performance metrics via cross-validation
- Problem: the drift estimation depends on the volatility and the volatility estimation depends on the drift. How to optimize jointly?

Hyper Parameters Optimization via Cross Validation

Hyper-Parameters to Validate

- $r_{t+1} = f(X_t) + \epsilon_t$ the best combination of $f(X_t)$ to estimate the drift
- $\epsilon_t = \underbrace{\sigma_t}_{\text{Dynamic model}} \underbrace{e_t}_{\text{Distribution}}$

- We optimize the $r_{t+1} = f(X_t) + \epsilon_t$ using an RMSE/MAE/MAPE type of metric. We obtain the optimal out-of-sample error term ϵ_t^*
- Then we optimize both the dynamic model and the distribution $\epsilon_t^* = \underbrace{\sigma_t}_{\text{Dynamic model}} \underbrace{e_t}_{\text{Distribution}}$ using the PIT, logscore, and tail logscore
- Problem: we have a "Frankenstein model" where the hyperparameters - distribution of the error term and the specification of the drift - are optimized separately. How do we

Estimate the Parameters via The Zig-Zag Algorithm

- Idea: break down the estimation of the mean and the variance separately
  1. **First estimate of the mean** by assuming a constant variance model (σ^2 fixed) and estimate the mean of the drift equation $\hat{mu}$. This allows to obtain the estimated residuals $\hat{\epsilon_t} = r_t - \hat{\mu}$
  2. **Initial Volatility Model Estimation using the cross-validated density model** use a zero-mean GARCH model on the first-step residuals $\mathcal{E}[\hat{\epsilon_t}] = 0$ and obtain the dynamic of $\hat{\sigma_t^2}$. For instance, a standard GARCH, EGARCH, TGARCH, etc.
  3. **Re-estimate the drift using the cross-validated specification** and by plugging the volatility process estimated in the previous step. Because the new errors terms $\hat{\epsilon_t} = \hat{\sigma_t} e_t$ with $\hat{\sigma_t}$ estimated from the previous trend are derived from an estimation apply a GLS correction on the errors terms

- The Zigzag algorithm repeats steps 2 and 3 up to the point they converge