Un état de l'art sur le forecasting des séries temporelles

VERDIERE Jeffrey

2022-2023

Table of Contents

Introduction

A definition

In today's data-driven world, organizations are constantly searching for ways to use historical data to predict future trends and make informed decisions. Time series data, which records information over time, plays a crucial role in this quest. Whether it is predicting sales, stock prices, or weather patterns, time series forecasting has become an essential tool for businesses, researchers, and policymakers.

This whitepaper is a comprehensive guide to time series forecasting. We will explore the basics, techniques, and best practices that enable accurate predictions in various fields. Whether you are a data scientist, business analyst, or simply interested in forecasting, this whitepaper will provide valuable insights to help you make better decisions.

Many fields, many use cases and many forms

Time series forecasting is a versatile tool widely used in many fields. For instance, it is crucial in tasks like predictive maintenance, where it predicts when machines need repairs to prevent breakdowns. In IT, it helps predict server loads and network traffic for better resource management. In finance, it aids in predicting stock prices and market trends. In retail, it assists in demand forecasting and inventory management. In fact, time series forecasting is used extensively in various industries, highlighting its widespread relevance and importance.

Moreover, time series data comes in various forms, including time series with multiple variables, external data, trends, seasonalities, holes or irregularities. All those different kinds of time series should be addressed in various ways in order to harness the data and enhance algorithmic performance.

Therefore, embarking on a journey through time series analysis, a data scientist must pivot through numerous vital questions to navigate the intricate tapestry of temporal data modeling.

Therefore, we can summarize all the major questions a data scientist will need to ask to address a business problem involving time series in the following table

The issues and questions to consider regarding time series forecasting

 **Business issue**	What underlying business or scientific challenge is at the core of the problem? (Time series forecasting, classification...)
 **Data Quality issue**	Are the data consistent and complete across the temporal spectrum How will inconsistencies or missing values be addressed?
 **Feature engineering and preprocessing**	How can historical data be leveraged to create informative features? Are there any available/useful exogenous data or covariates? What preprocessing steps, like detrending, filtering or deseasonalizing, are warranted?
 **Model selection and hypothesis testing**	Which model (e.g ARIMA, LSTM, Prophet) is appropriate given the data characteristics and problem complexity?
 **Validation and Model Assessment**	What strategy will ensure robust model validation (e.g, time-based split, rolling forecast origin) Which metrics will accurately reflect the model's predictive prowess and reliability?
 **Deployment and maintenance**	How will the model be deployed, considering real-time or batch predictions? What approach will be adopted for model monitoring and periodic recalibration?

How to frame and explore a time series problem?

In this section, we will detail the main question to ask when facing a time series, and the initial steps to properly understand the nature of the data. We will start by discussing the questions to consider before embarking on the project. Then, we will look at the main steps of data exploration: **data structuring, data quality analysis**, and the **exploratory phase**.

The business questions to ask

Before starting a time series problem, it is often necessary to ask the following business questions.

Questions	To whom?
• What is the volume of available data?	Operational roles/Data Scientist/Data Engineer/Business owner
• What is the temporal granularity of the data?	Data Scientist/Data Engineer
• What is the time series granularity of the study (SKUs, product)?	Business owner
• Over what horizon and at what frequency do we want to make a forecast? Are there any exogenous data or covariates available/useful?	Business owner
• Are there any covariates/exogenous data available?	Business owner/data scientist/Data Engineer
• Is the data stationary?	Data Scientist
• Do we need a probabilistic or deterministic model?	Data Scientist/Business owner
• Is the data univariate or multivariate?	Data scientist
• Is there seasonality in the data?	Data scientist
• What is the importance of explainability in the use case?	Business owner

The following questions usually deal with data quality issues and exploration, which we will see in the subsequent sections.

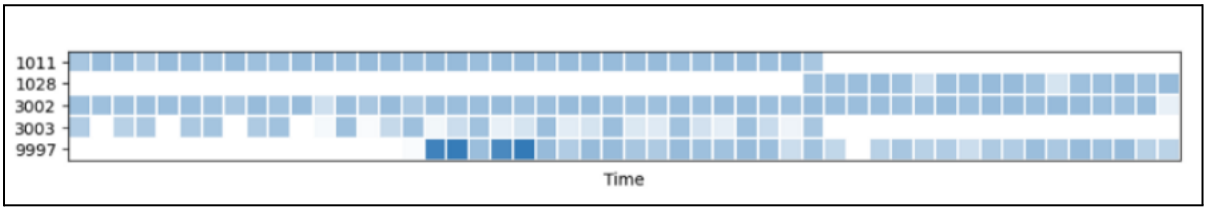
Evaluation of data quality

Time series analysis can present numerous challenges, owing to the inherent nature of data production. Thus, it's important to pay attention to data quality, particularly during extraction. In time series analysis, one may encounter four major types of data quality issues explained in the following table.

Problem	Description
• **Validity**	Values outside the range, impossible rapid variations, inaccurate data order.
• **Precision**	Inconsistent sampling, poor sensor accuracy
• **Completeness**	Empty fields, no keys/IDs, no metadata, no source information
• **Temporality**	Discrepancy between the data and reality

Exploratory data analysis (EDA)

EDA (Exploratory Data Analysis) is a crucial step in understanding the data used before moving on to models. To start, it's essential to grasp how much data we have, particularly the **number of columns**: are these **univariate, multivariate?** Some tools such as the **heatmap** can be very effective to quickly visualize the amount of data in each column and the frequency measurements.



We can then see here that not all data started to have records at the same time, and the colors show that some data may have more points over a given period. In this case, it might be useful to use a dynamic time range rather than a predetermined one to address these issues.

Now that we have seen the initial steps to explore a time series problem, the first step before applying forecasting models is to preprocess the data.

Preprocessing challenges

In time series analysis, **preprocessing** is crucial for ensuring data quality and reliability. This involves **removing outliers**, **imputing missing values**, and **applying filtering** and **noise reduction** techniques to achieve more accurate and meaningful insights. Additionally, it is essential to properly split the dataset into training and test sets and to utilize the **autocorrelation function** to understand the **data's inherent patterns** and **dependencies**.

Removing outliers

Detecting and removing **outliers** in time series data is a crucial step in data analysis, with far-reaching implications for decision-making and model accuracy. Outliers, those aberrant data points that deviate significantly from the expected pattern, can distort statistical summaries, lead to inaccurate forecasts, and even misguide important conclusions. **Their removal is essential** for obtaining a clearer and more accurate understanding of the underlying patterns and trends within the data.

However, handling outliers in time series data can be highly non-trivial. Sometimes, distinguishing between genuine anomalies and measurement errors can be challenging. To address this, various techniques are employed. Smoothing methods, like moving averages or exponential smoothing, can help mitigate the impact of outliers by assigning more weight to nearby data points. Alternatively, statistical tests, such as the **Z-score** or **modified Z-score**, can flag data points as outliers if they deviate significantly from the mean or median of the series. Moreover, advanced methods like robust regression models or anomaly detection algorithms like **Isolation Forests** and **One-Class SVMs** are also used to identify and manage outliers in a more sophisticated manner.

Ultimately, the careful detection and removal of outliers in time series data contribute to **improved data quality, more reliable insights, and more accurate predictive models**.

Missing values imputation

Addressing **missing values** in time series data is equally crucial for accurate analysis and modeling. Missing values can distort statistical summaries, hinder the detection of patterns, and lead to unreliable forecasts. **Imputation**, or the process of filling in missing values with estimated or substituted values, is essential to maintain data completeness and integrity.

Imputation is essential because it allows you to retain valuable information that would otherwise be lost due to missing data. By imputing missing values, you can ensure that your time series remains consistent, enabling you to draw meaningful insights and make reliable predictions.

However, imputing missing values in time series data is **not always straightforward**. Time series data often exhibits temporal dependencies, where the value at one time point is related to previous and future values. Therefore, imputed values should not disrupt the temporal structure of the data. Common imputation methods include **forward fill** (using the last observed value), **backward fill** (using the next observed value), **linear interpolation**, and **seasonal decomposition** for more complex imputations.

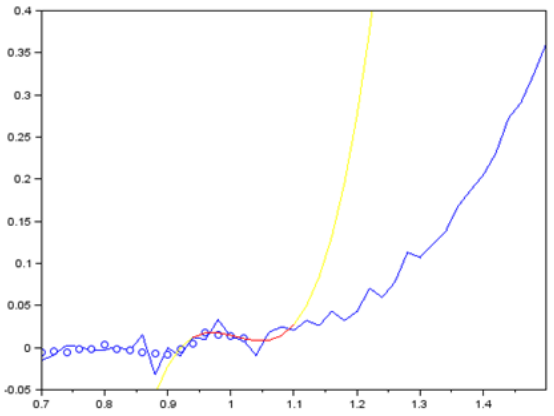
Sophisticated techniques like **regression imputation**, where missing values are estimated based on relationships with other variables, or machine learning-based imputation, such as **k-nearest neighbors imputation**, can be employed to handle more complex scenarios.

In summary, imputing missing values in time series data is essential for maintaining **data completeness and integrity**, ensuring that valuable information is not lost, and enabling meaningful analysis and modeling. The choice of imputation method depends on the nature of the data and the specific requirements of the analysis.

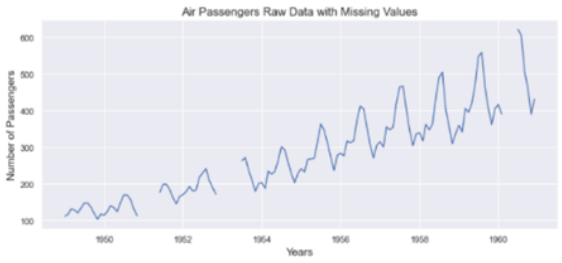
Other transformation techniques

In addition to handling outliers and missing values, there are several other preprocessing techniques that can be applied to time series data, depending on the specific characteristics of the data and the goals of the analysis. **Normalization** scales the values in the time series to a specific range, such as [0, 1], making it easier to compare time series with different units or scales. **Detrending** involves removing the underlying trend or long-term component from a time series, which can help in analyzing and modeling the remaining fluctuations. **Filtering**, like the *Savitzky-Golay filter*, can smooth noisy time series data, reducing random fluctuations while preserving essential features. **Differencing** is a common technique to stabilize non-stationary time series by computing the differences between consecutive data points. These preprocessing steps play a vital role in enhancing the quality and suitability of time series data for analysis and modeling.

The Savitsky-Golay filter is an example of a sophisticated method to smooth time series data. It is illustrated in this figure.



Original time series, with missing values:

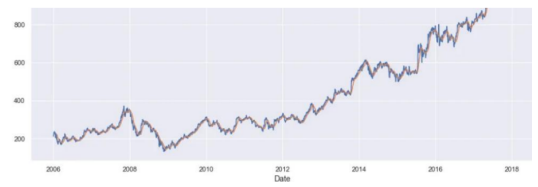
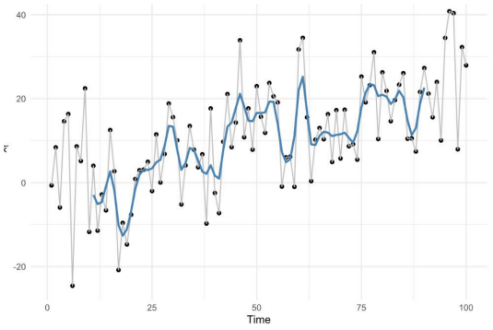
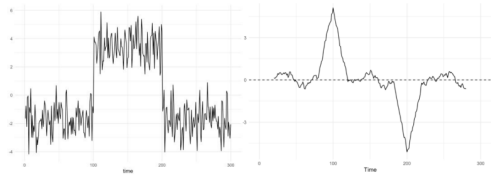


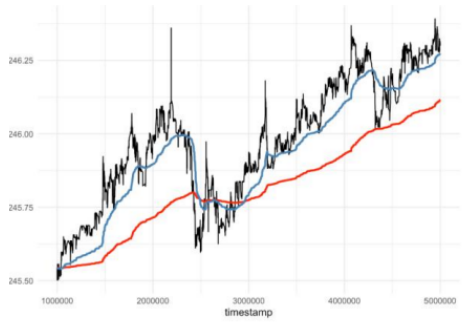
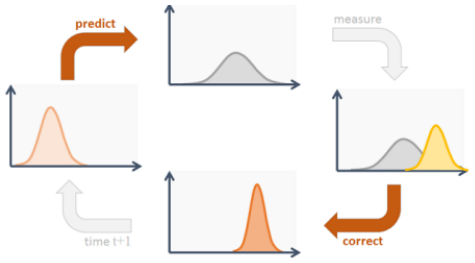
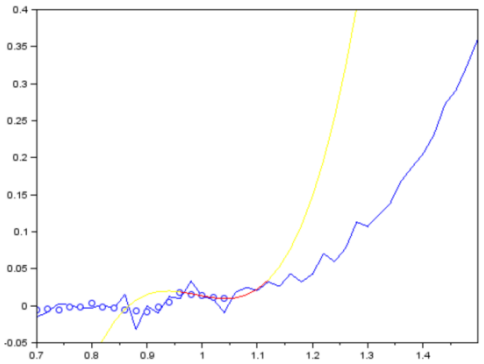
Many interpolation techniques exist, some of them (Linear, Spline and Time Based) are illustrated in these figures.

Imputed time series, using various

methods : Linear, Spline, Time Based

Noise in time series can cause significant problems, and it is generally recommended to remove the noise from the time series before analysis. This process is called **denoising**. Most filters can be summarized in the following table.

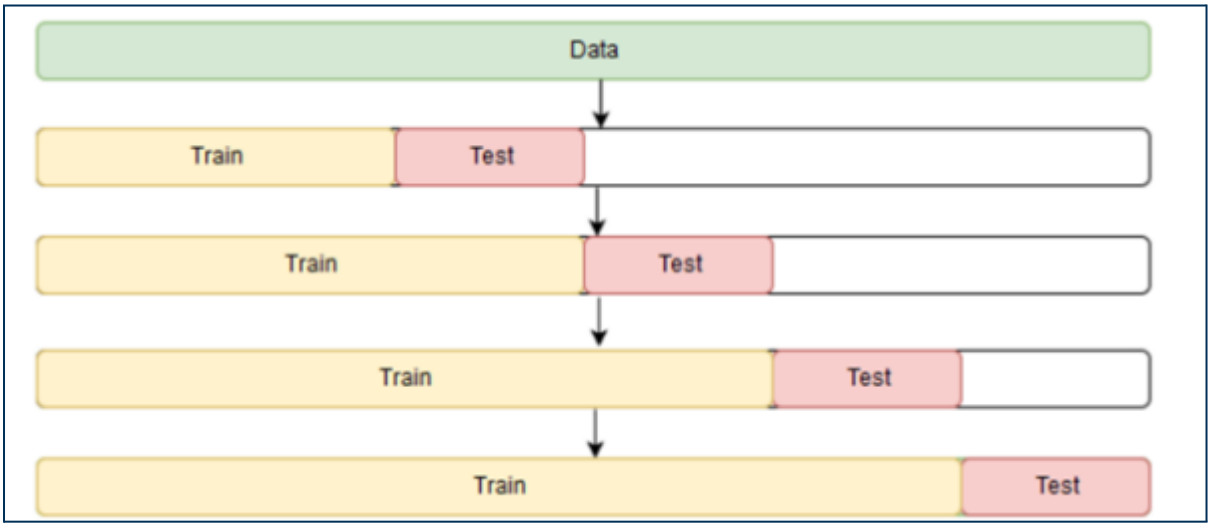
Type of filter	In which case	Illustration
Moving average	**Smoothing out** fluctuations by calculating the average over a sliding window, useful for identifying short-terms trends	
Fourier transform	Converting the series into **frequency domain**, allowing for the analysis of periodic components present;	
Low-pass transform	**Attenuating high frequencies** to preserve slow variations, effective for removing noise and highlighting long-term trends	
Matching filter	Detection of a **specific pattern** in a time series, often used in the detection of signals or specific events	

Exponential Smoothing	**Weighted** combination of past values to give more weight to recent observations	
Kalman filter	**Correcting** the prediction by **comparison** with real values	
Savitzky-Golay filter	**Polynomial** approximation over a sliding window	

Now that we have preprocessed the data, we generally need to think about strategies for separating the test and training datasets.

How to split my time series data set

The train-test split of time series data is unique because we must ensure that the test set is older than the training set. Therefore, we need to keep the latest values of the series for comparison with models during the train-test split. **Cross-validation** also has a specific format as the training values must always precede the prediction values. We should not use validation data for model training to avoid overfitting. The randomness in the split arises only from the size allocated to the training and test sets. Another strategy is to maintain a constant training window size to minimize the influence of distant values. We can then test at different periods. This split can be done manually by deciding a cutoff index that separates the two sets. There is also a function in the scikit-learn library that allows doing this by making several (n_splits) different separations while maintaining the chronological constraint between the two sets



Forecasting models

In this section, we will discuss the different types of models that allow for predictions based on a pre-processed, imputed, and filtered time signal such as shown in the previous section. We will introduce the main types of models, the theory behind them, their advantages and disadvantages, as well as their implementation. We will cover three major model families: **statistical models**, **machine learning models**, and **deep learning models**.

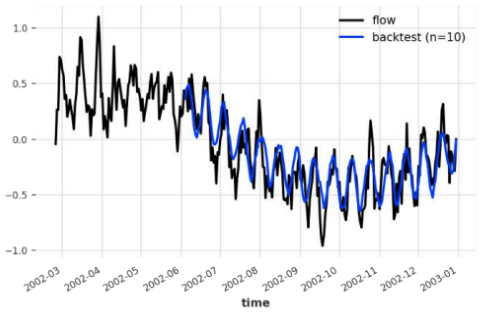
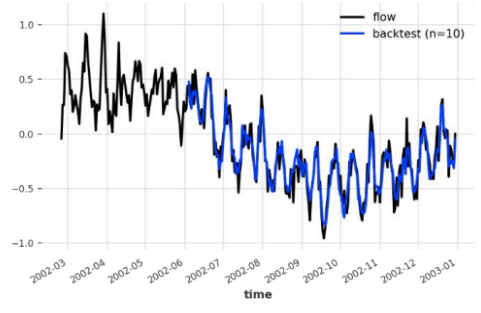
We can the summarize the advantages and disadvantages of the various statistical, machine learning, and deep learning models in the following matrix:

Family	Model	Need for stationarity	Hyperparameter fine-tuning	Univariate/ Multivariate Data	Model allow the inclusion of exogenous variables	Takes into account past/static/ future variables	Main python libraries
Statistic models	ARIMA	✔️	GridSearch	✔️/❌	✔️	❌/✔️/❌	Statsmodels
	SARIMA	✔️	GridSearch	✔️/❌	✔️	❌/✔️/❌	Statsmodels
	VARIMA	✔️	GridSearch	❌/✔️	❌	❌/✔️/❌	Statsmodels
	Exponential smoothing	❌	GridSeach	✔️/❌	❌	❌/❌/❌	Statsmodels
	Prophet	❌	-	✔️/❌	❌	✔️/✔️/✔️	prophet
	Croston method	❌	-	✔️/❌	❌	✔️/✔️/✔️	
Machine Learning models	Linear regression	❌	-	✔️/✔️	✔️	✔️/✔️/✔️	sklearn
	RandomForest	❌	-	✔️/✔️	✔️	✔️/✔️/✔️	sklearn
	Decision Tree	❌	-	✔️/✔️	✔️	✔️/✔️/✔️	sklearn
	XGBoost	❌	-	✔️/✔️	✔️	✔️/✔️/✔️	xgboost
Deep Learning method	RNN	❌	GridSearch	✔️/✔️	✔️	❌/✔️/❌	keras
	NBeats	❌	GridSearch	✔️/✔️	✔️	✔️/❌/❌	NBEATS
	Transformer	❌	GridSearch	✔️/✔️	✔️	✔️/❌/❌	gluon

Exogenous variables and covariates

To improve **model performance**, it is often necessary to add **exogenous, external data** that has significant added value. Not all models take this type of data into account, but it is often possible to incorporate these variables.

- **Statistical models** often offer a variant with X (e.g ARIMAX, SARIMAX) to incorporate exogenous variables in the 'exog' part, which must be the same length as the endogenous variable. However, it is also necessary to include them in the prediction part: they must be known in the future. This is why they are often calendar values (holidays, vacations, special events, etc.).
- **Deep Learning methods:** there is a specific terminology for exogenous data: they are referred to as covariates. It's exactly the same concept, but covariates are divided into 3 types. We detailed each of them in the following matrix with the example of a time series of a river flow forecasted. In this following example, we predict the river flow using additional covariates to enhance the quality of the predictions.

Covariate name	Description	Example
Static covariates	They represent a value that **does not change** over time but helps to identify multiple elements. Therefore, they are often identifiers.	
Past covariates	These are **exogenous** data for which we only have values from the past. For example, a past covariate will allow the model to understand that the drop in sales in March 2020 was due to Covid, and therefore not to replicate this behavior subsequently.	
Future covariates	For these covariates, we have values in both the **past** and the **future.** As with the previous models, these variables are often predictable calendar events, weather information.	

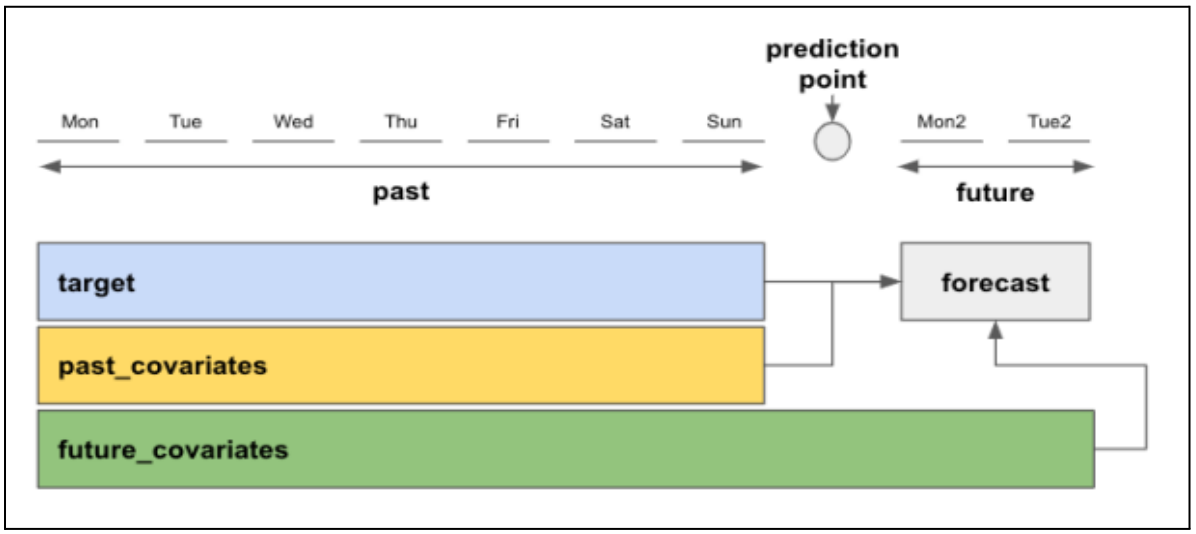
Statistics: integration of data columns into the exogenous parameters of models like ARIMAX, SARIMAX, etc

Machine Learning: addition of exogenous data as features in the models.

Deep Learning: integration of covariates into models that account for them:

- **Static covariates:** data used to differentiate time series.
- **Past covariates:** exogenous data of which only past values are available.
- **Future covariates:** exogenous data of which both past and future values are available.

The addition of covariates to the prediction can be summarized in the following figure:



Evaluation

In this section, we delve into the critical aspects of model evaluation, an essential phase in the lifecycle of predictive modeling. We begin with a thorough exploration of **cross-validation techniques**, which are pivotal in assessing the **robustness** and **reliability** of our models. This is followed by a discussion on various **prediction strategies**, each offering unique approaches to forecasting and analysis. We then shift our focus to the selection of **appropriate metrics**, essential for quantifying model performance accurately. Lastly, we outline the **criteria for model selection**, guiding us in choosing the most effective models for our specific needs.

Cross-validation

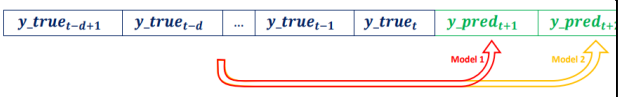
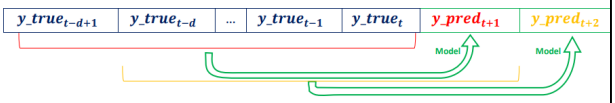
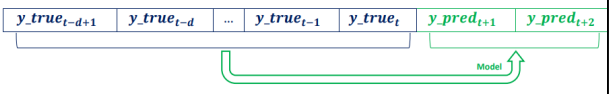
The evaluation of time series model performance is very important, not only from a technical standpoint in finding the best parameters but also from a **business perspective** when the results significantly support **decision-making**. Therefore, it is vital to understand model performance to gauge how much we can trust its predictions. For this purpose, we will use the test set created for cross-validation.

Even though a model's capability is measured by its performance, it is essential not to overlook other factors that can influence the choice of a model:

- Complexity and number of parameters
- Training time
- Explainability (of the decision, underlying causes, etc.)
- Confidence intervals

Prediction strategies

Time series models are sometimes used when it is necessary to predict **several future values:** this is referred to as multi-step time series forecasting. There are four **major prediction strategies** in this case:

Strategy	Description	Details
• Direct prediction strategy	Use different models for each prediction horizon	
• Recursive prediction strategy	Use a single model, with previous predictions as inputs	
• Hybrid direct-recursive prediction strategy	Combine the previous two	
• Multiple Output Prediction Strategy	Predict an entire sequence in one go	

Metric

In time series analysis, a wide range of evaluation metrics is available, each offering unique insights. The choice of metric depends on specific criteria like **data nature**, **model purpose**, and **required precision**. This ensures a tailored, effective model assessment.

The metrics			
Metric	**Use**	**Advantages**	**Disadvantages**
R^2	When the relationship between predictions and actual values is known	Compare performance against the average.	Provides only information on the quality of the fit
MAE	When there are few extreme values	- Easy to understand	Ineffective with extreme values
MAPE	When there are few extreme values	- Easy to understand - Useful for comparing models and datasets	Ineffective with extreme values
sMAPE	When there are few extreme values	- Has minimum (0%) and maximum (200%) threshold value	Does not fully address symmetry issues
MSE	For extended values	- Useful for very wide-ranging values - More stringent with larger values	Difficult to interpret
RMSE	For extended values and interpretability	- Useful for very wide-ranging values - More stringent with larger values - Interpretable as it is in the same unit	Sensitive to outliers
NRMSE	For extended values and comparing models or datasets.	- Useful for very wide-ranging values - Strict with larger values	Loss of measurement unit and less interpretable

The criteria for selection a model

There are also **model selection methods** that do not solely rely on performance to choose the best model. For instance, when comparing different statistical models, it is important to consider both the performance and the number of parameters in the evaluation. Indeed, the number of parameters greatly influences the training time of the models. With two nearly equivalent models, the one with the fewer parameters should be favored. To this end, there are criteria, of which the two main ones are:

- Akaike Information Criterion (AIC): $AIC = n.\ log(\sum (y\_truei - y\_predi)^2 + 2k$, where n is the number of observations in the series, and k is the number of model parameters. The goal is to select the model with the lowest possible AIC score.
- Bayesian Information Criterion (BIC): $BIC = n.\ log(\sum (y\_truei - y\_predi)^2 + k.\ log(n)$. The goal is to select the model with the lowest BIC score.

Compared to BIC, the AIC criterion penalizes complex models less: BIC assigns higher penalties to complex models than AIC. Both criteria are very useful and can be used together to compare different models and select the most relevant one.

Overview/maintenance/production

- **Model maintenance** is a crucial step in the CI/CD pipeline. A common issue is that the model performs well in the development, validation, or test phase but poorly in production. This is especially true for time series, where data changes can be rapid. There are three stages of time series monitoring during the development and production phases.
- First, as we saw in the previous section, it is necessary to choose the **most appropriate metrics** and measure them in the development phase. This allows for assessing the quality of the models, comparing them, improving performance, and possibly detecting overfitting between the test and validation sets. For Deep Learning methods, it is important to visualize learning curves, as these can indicate overfitting with good performance in training but poor in validation, and implement solutions like regularization or early stopping. This also helps to monitor the model's convergence and detect if it is converging to a suboptimal solution.
- Next, during the model training phase, it is important to monitor **hardware performance.** Monitoring hardware performance during the training of a DL model can help identify bottlenecks in the system. For example, **monitoring GPU** memory usage ensures that the model does not run out of memory and that training does not abruptly stop. This also ensures that hardware is used efficiently.
- Finally, once in production, it is necessary to continuously **monitor model** performance. It is also important to track **data quality**. By regularly monitoring the **evolution of metrics**, generally only small fluctuations are observed that do not require retraining of the model. However, if performance suddenly drops, one reason could be a change in data distribution. This is known as drift. To detect and address it, one solution can be to use two datasets for performance comparison: one dataset with only old values as a reference, and another dataset where new values are added. Then, measure performance on both datasets, or use other techniques like interpretability methods. Indeed, by comparing the most important features, one can ensure, for example, that the main features remain unchanged between the two tests.

Conclusion

- Beyond **preprocessing and modeling**, two points seem particularly crucial for having good time series models. First, the integration of **exogenous data** into the modeling process has proven its importance for improving forecast accuracy, taking into account external factors that can influence time series. On the other hand, **rigorous evaluation of model performance** is also crucial to ensure their relevance in real-world applications. Therefore, choosing relevant metrics is a step that should not be overlooked, and one should not hesitate to **create custom metrics** depending on the case studied.