



Medical Time Series Forecasting

24 Feb 2023, Leuven, Belgium

Ioanna Miliou

Assistant Professor, Data Science Group, Stockholm University

Outline

Intro to Time Series Forecasting

Statistical Forecast Methods

Machine Learning Methods

Deep Learning Models

Evaluation Metrics

Nowcasting



Time Series



**A sequence of observations
obtained in successive order over
some period**

Financial Data

Heights of Ocean Tides

Weather measurements

Product Sales

...



**Very common in the medical
domain**

Electroencephalographic signals

Heart rate

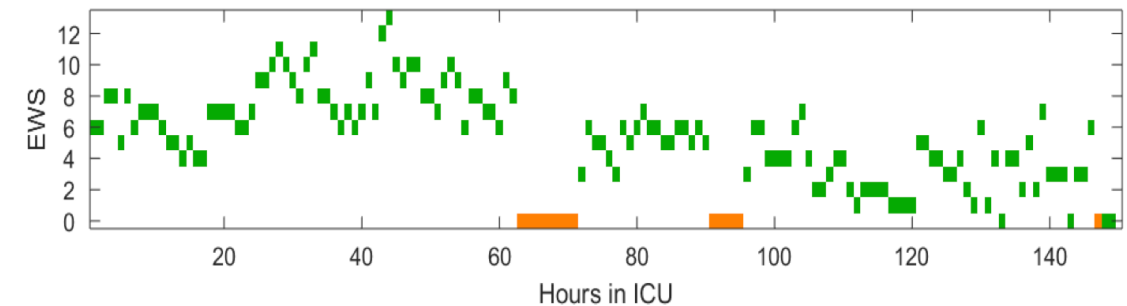
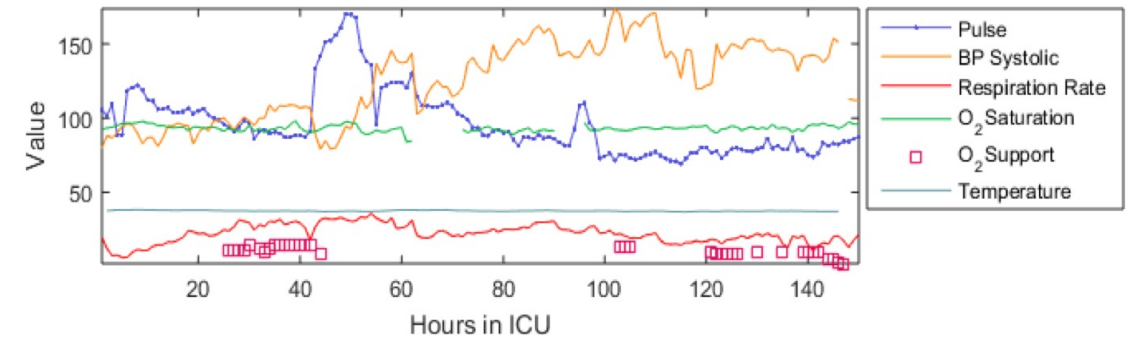
Respiratory rate

Temperature

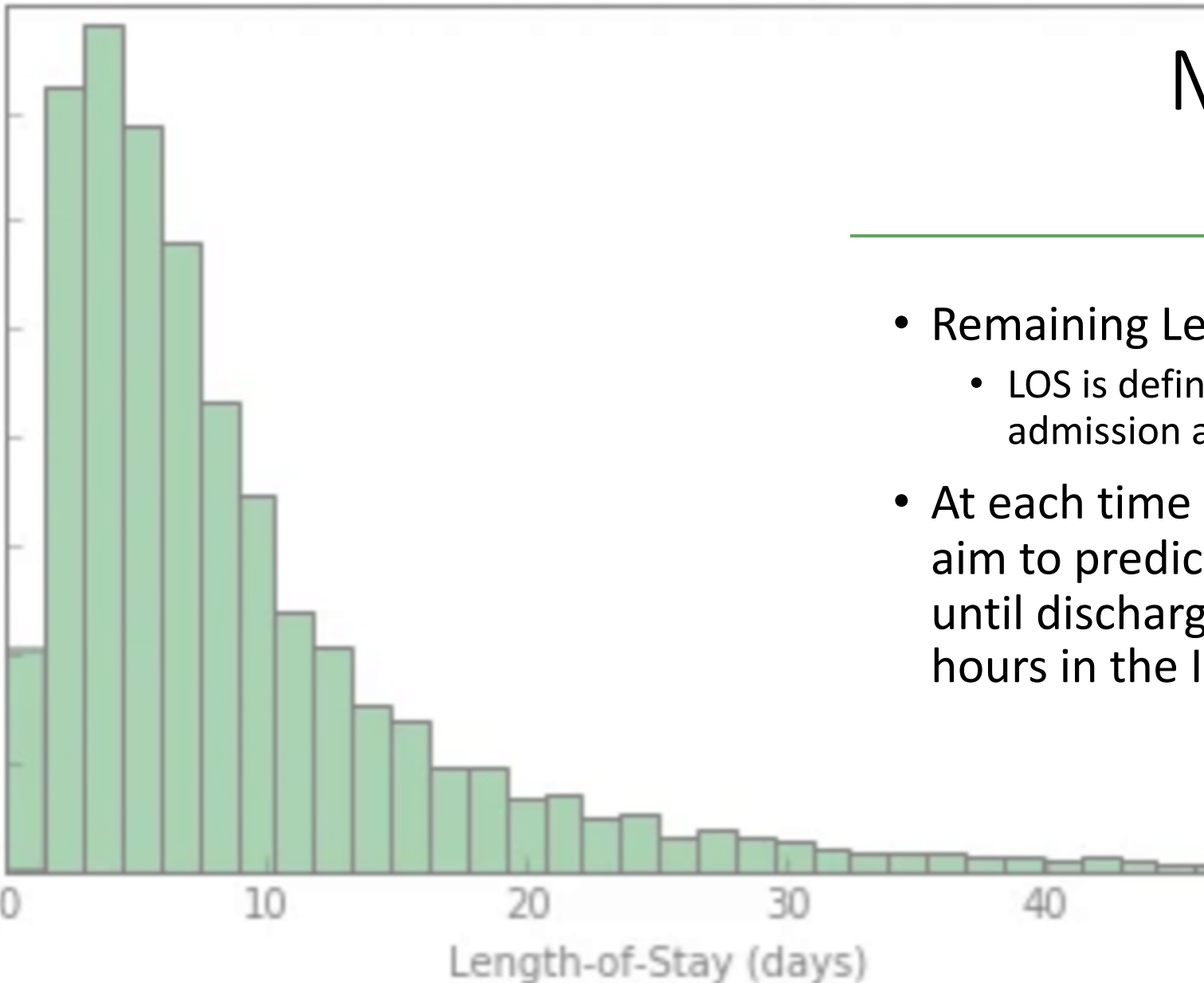
...

Medical Time Series Forecasting

- Intensive Care Unit (ICU) patient's progression
- Vital signs deterioration prediction
- At each precise time step t in the admission period, we aim to predict the values for the same variables in the forward time of h hours using the records of p past hours in the ICU



Distribution of LOS for all hospital admissions



Medical Time Series Forecasting

- Remaining Length Of Stay (LoS) Prediction
 - LOS is defined as the time between hospital admission and discharge measured in days
- At each time step t at the admission period, we aim to predict the remaining number of days until discharge, using the records of the p past hours in the ICU

Formulating the problem



In a typical scenario, we have an outcome measurement, usually **categorical** (such as heart attack/no heart attack) or **quantitative** (such as a temperature measurement), that we wish to predict based on a set of features

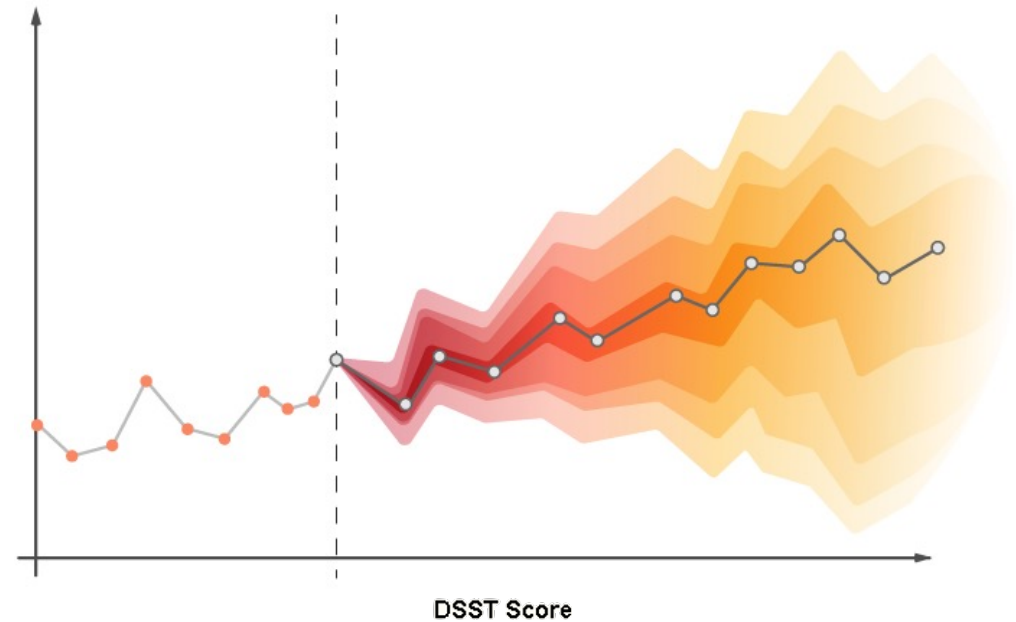
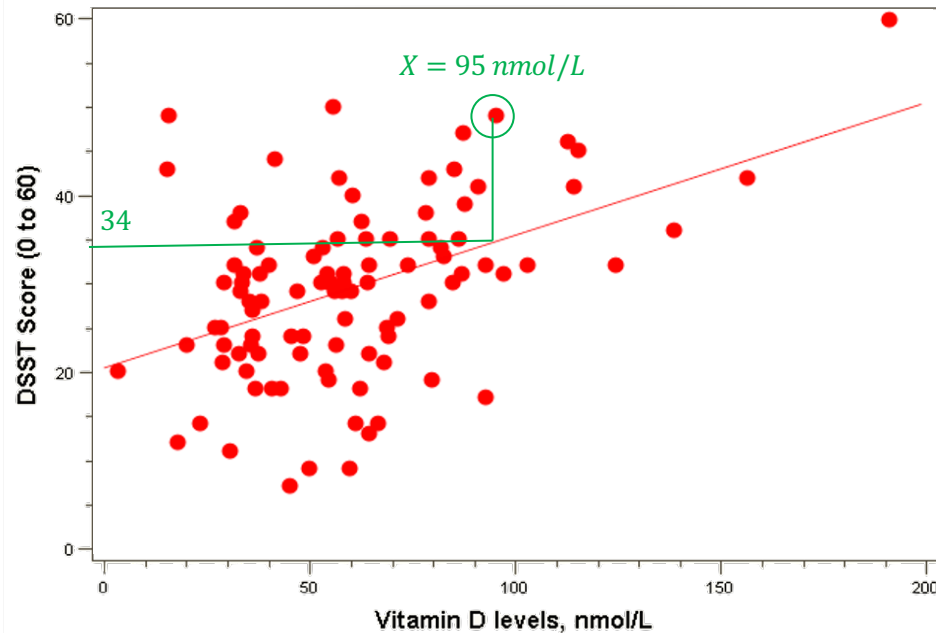
- **Classification**: when we predict categorical (qualitative) outputs
- **Regression**: when we predict quantitative outputs

Regression

- Statistical measure that attempts to determine the **relationship** between the **dependent variable Y** and a set of **independent variables X**
 - **Dependent or response variable:** the variable whose values change as a consequence of changes in other values in the system
 - **Independent or predictors or explanatory variables:** regarded as inputs to the system and may take on different values freely
- Widely used for **predicting** the next value or values of the **dependent variable Y** from the values of the **independent variables X**

Regression vs Forecasting

- **Regression**: determine the relationship between two time series
- **Forecasting**: predict the next value of a time series from the values of one or multiple time series in a dataset using the learned relationship

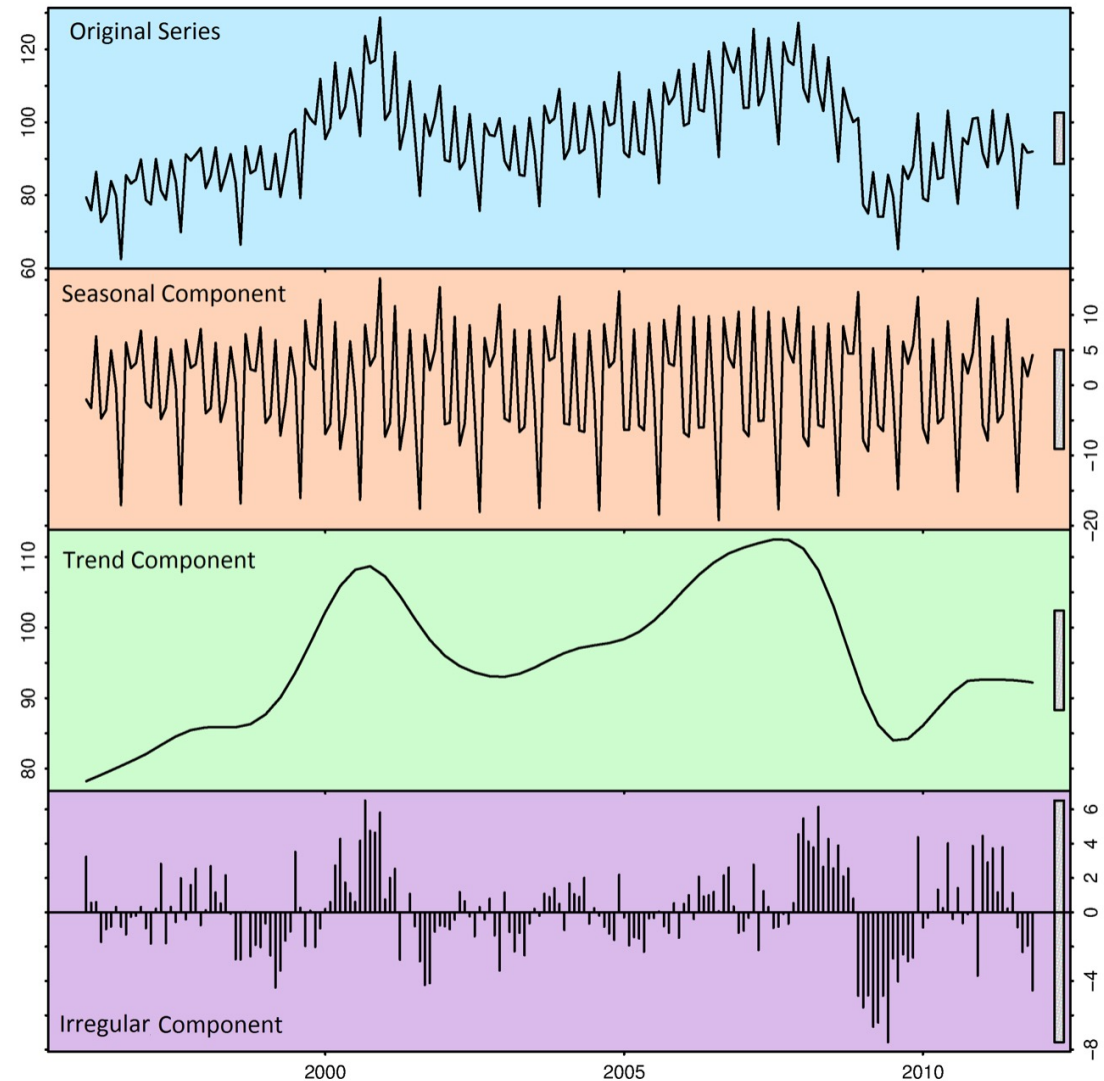


Dealing with time series

- Is there a trend? Seasonality?
- Is the time series stationary?
- Is it autocorrelated?

Components of a Time Series

- Time series are full of patterns and relationships
- We can separate a time series into three components:
 - Seasonal Component (S)
 - Long – term Trend (T)
 - Random (Irregular) variation (R)



Why decomposing a time series?



Analysis and understanding of historical time series



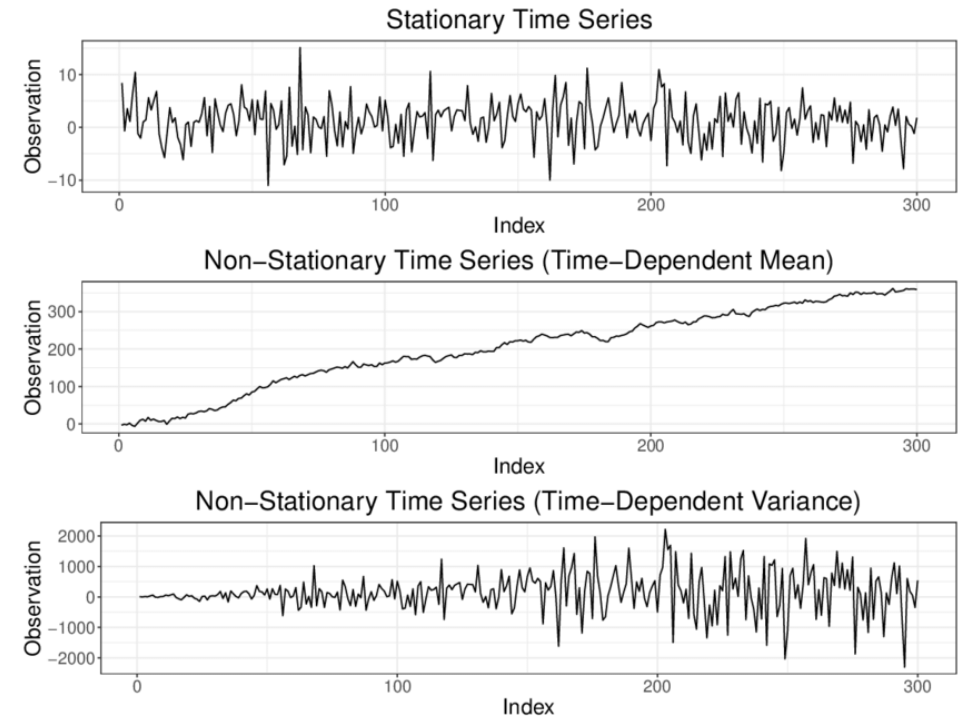
Seasonal adjustment



Time series prediction – forecasting and nowcasting

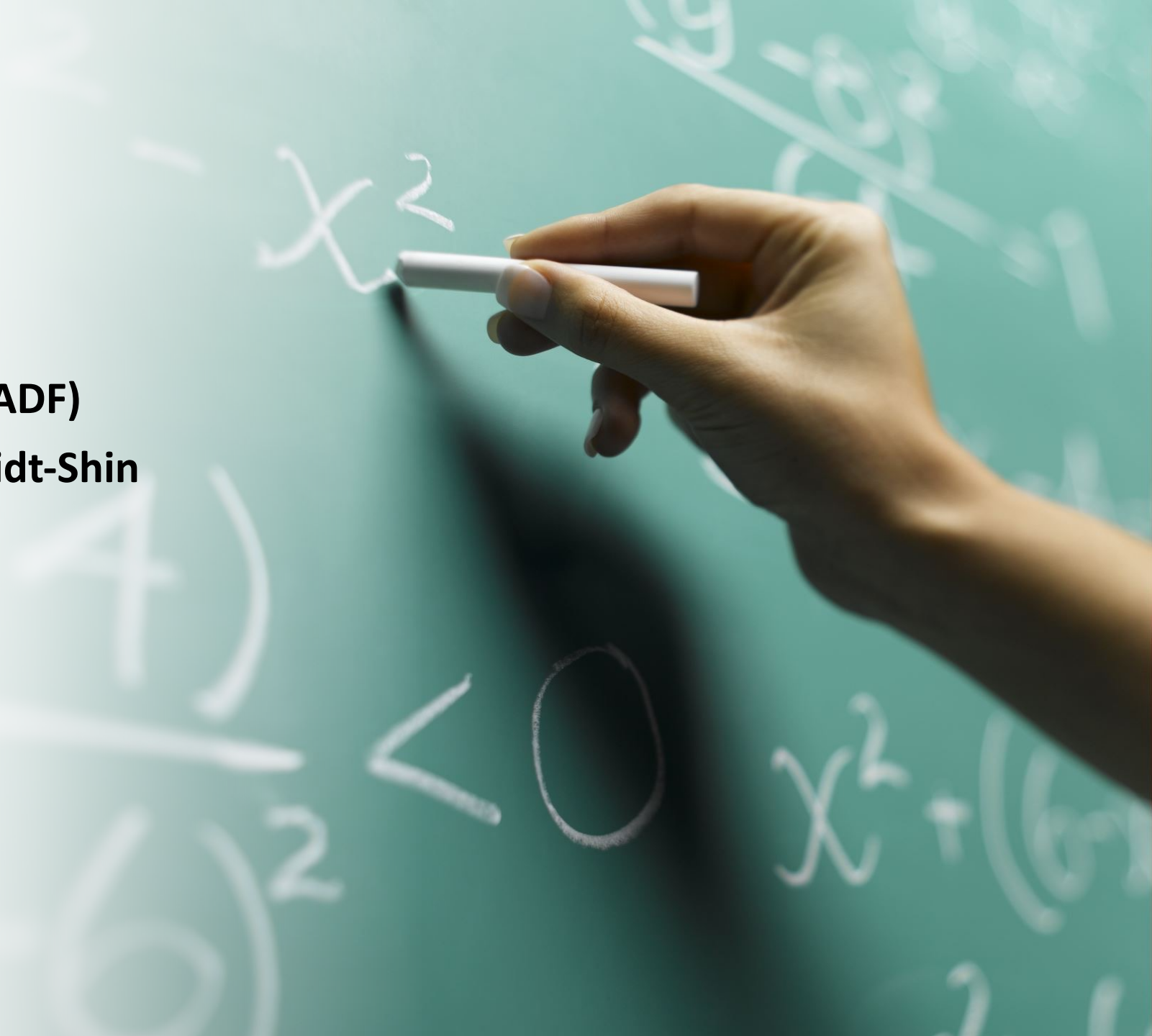
Stationarity

- Prediction is easier when a time series is stationary
 - its distribution does not change over time
- Stationarity can be defined in precise mathematical terms, but in general that means a flat looking series **without trend**, **constant variance** over time, **a constant autocorrelation** structure over time and **no periodic fluctuations** (seasonality)



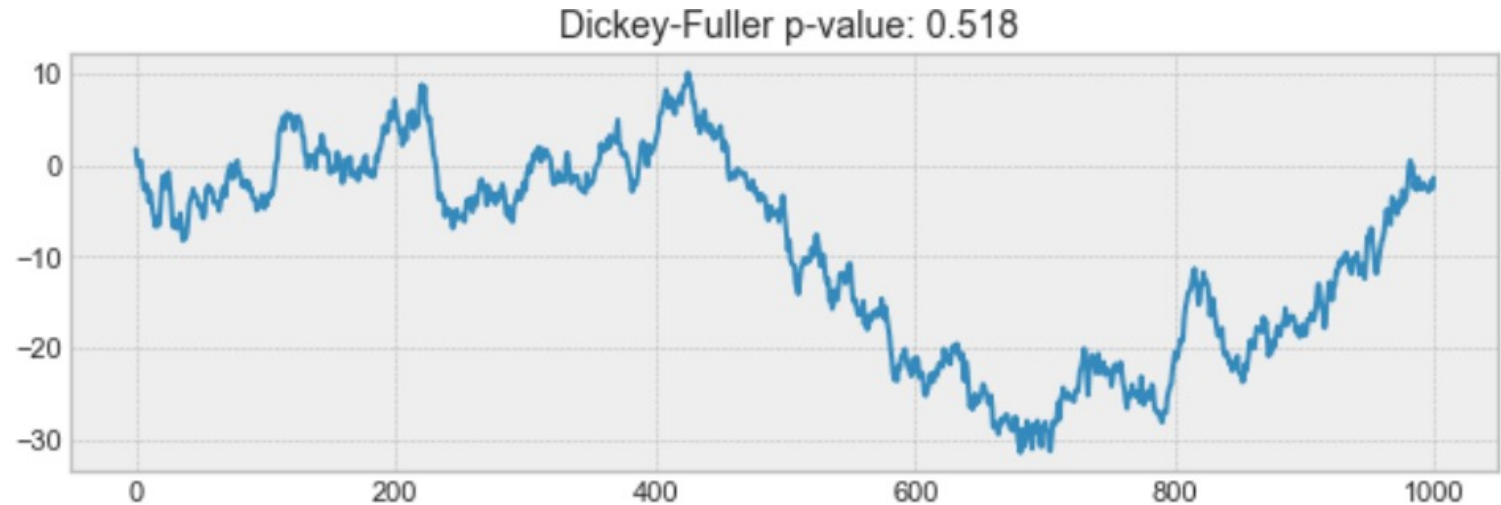
Tests for Stationarity

- **Augmented Dickey-Fuller (ADF)**
- **Kwiatkowski-Phillips-Schmidt-Shin (KPSS)**
- Elliott-Rothenberg-Stock
- Phillips-Perron test
- Schmidt-Phillips test
- Zivot-Andrews test



Dickey-Fuller test

- **Dickey-Fuller** is a statistical test that determines if a time series is stationary or not
- It tests the null hypothesis that a **unit root** is present or that the time series is **not stationary**
 - If $p > 0.05$, the process is not stationary
 - Otherwise, $p < 0.05$, the null hypothesis is rejected, and the process is considered to be stationary





KPSS test

- **KPSS** is a statistical test that determines if a time series is stationary or not
- It tests the null hypothesis that a **unit root** is **not** present or that the time series is **stationary** around a **deterministic trend**
 - If $p < 0.05$, the process is trend stationary
 - Otherwise, $p > 0.05$, the null hypothesis is rejected, and the process is not stationary

Stationarization

- To stationarize we have to **difference** the time series
 - compute the differences between consecutive observations
- Lag-1 difference operator:

$$\nabla X_t = X_t - X_{t-1}$$

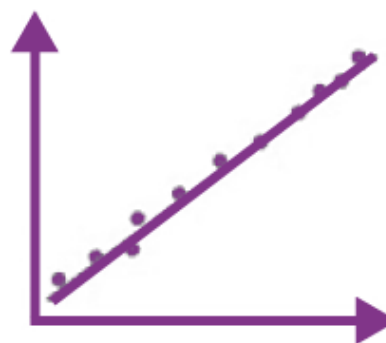
- We repeat the differencing until the time series is stationary
- Lag-k difference operator:

$$\nabla_k X_t = X_t - X_{t-k}$$

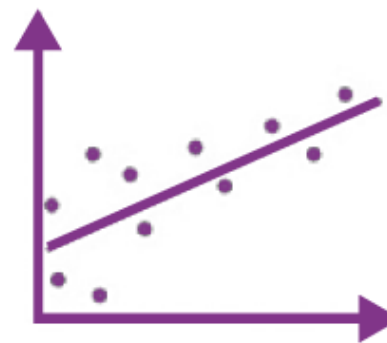
Independent and identically distributed random variables

- A collection of random variables is **independent and identically distributed (i.i.d.)** if each random variable has the same probability distribution as the others and all are mutually independent
- Random variables are **independent** if they satisfy a mathematical property of probabilistic independence
 - **Correlation**: measure of how two or more variables are related to one another
 - **Mutual Information**: a measure of the mutual dependence between the two variables by quantifying the amount of information obtained about one random variable by observing the other random variable

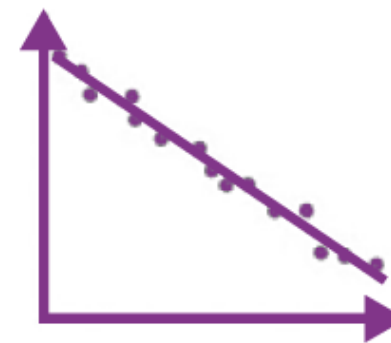
Correlation



Strong positive correlation



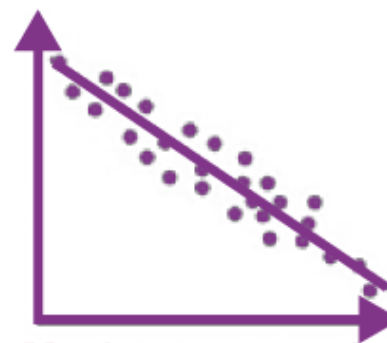
Weak positive correlation



Strong negative correlation



Weak negative correlation



Moderate negative correlation



No correlation

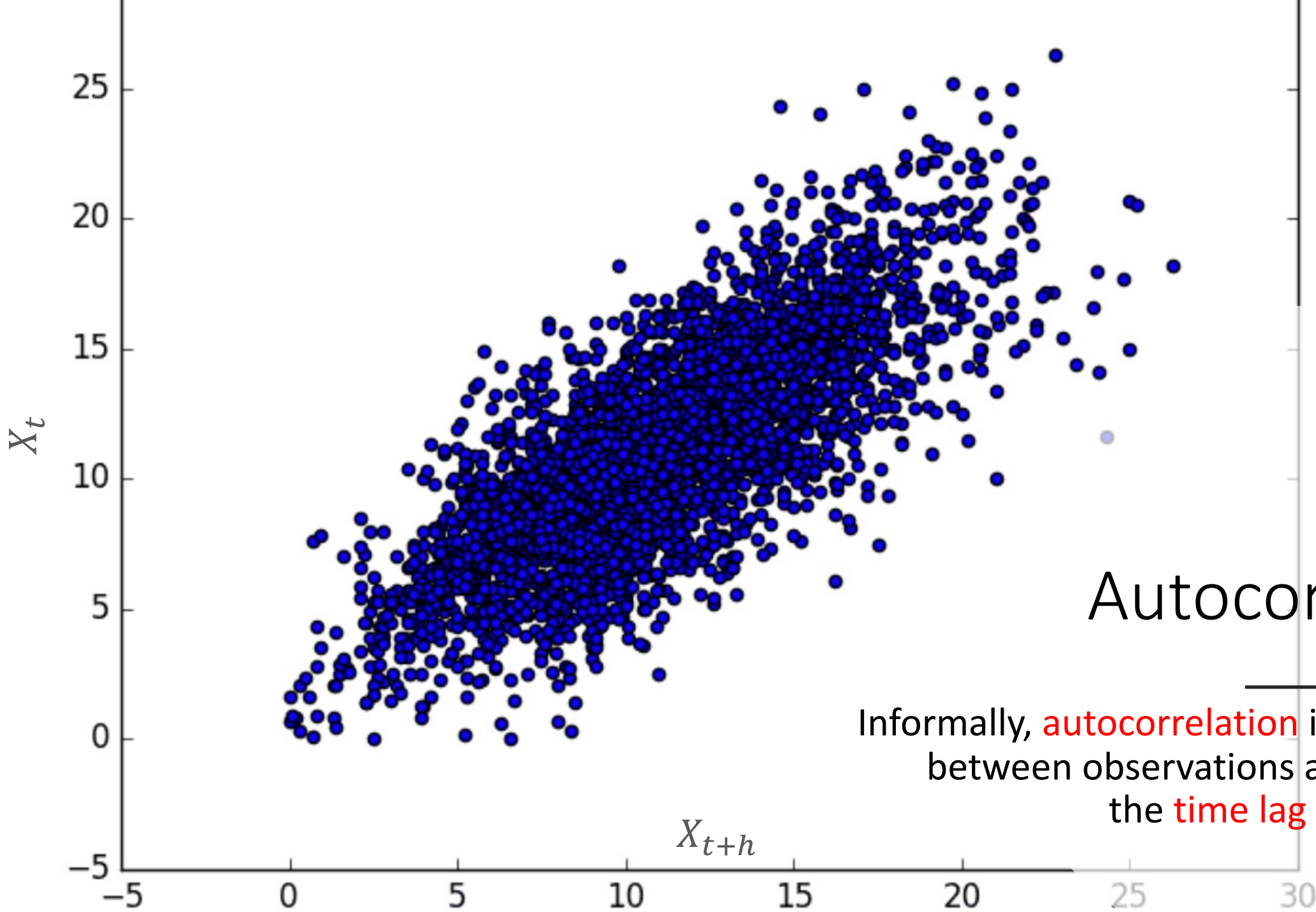
Correlation

- Given two time series X and Y , how do they correlate?
- **Pearson Correlation** is a measure of the linear dependence between two variables during period $[t_1, t_n]$:

$$r = \frac{\sum_{i=1}^n (Y_i - \bar{Y})(X_i - \bar{X})}{\sqrt{\sum_{i=1}^n (Y_i - \bar{Y})^2} \sqrt{\sum_{i=1}^n (X_i - \bar{X})^2}}$$

Lag

- To compute correlation, we need two variables
- But if we only have a time series, we need to compute the correlation of the time series with a k -th **lagged** version of itself
- **Lag**: for a specific time point t , the observation X_{t-k} is called the k -th lag of X_t
- A time series with lag $k = 1$ is a version of the original time series that is 1 period behind in time



Autocorrelation

Informally, **autocorrelation** is the similarity between observations as a function of the **time lag** between them

Autocorrelation Function

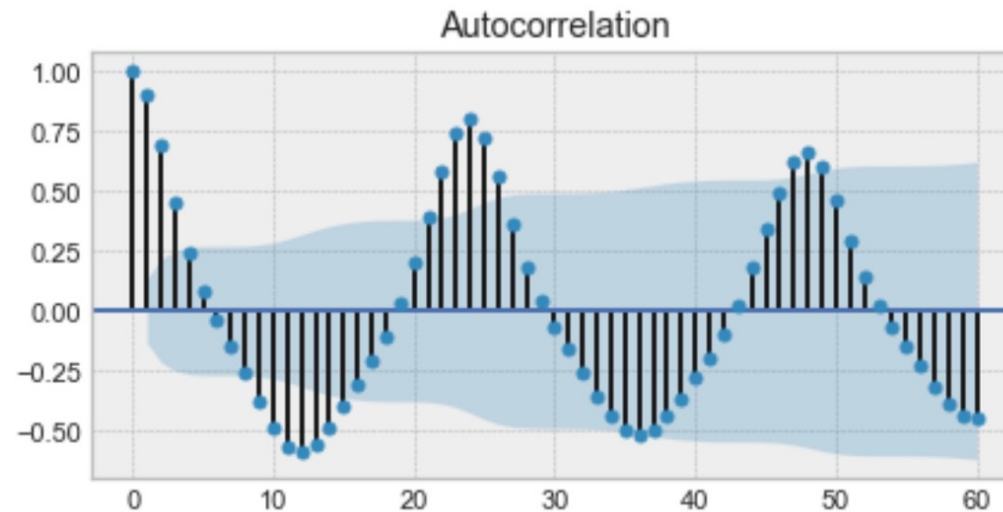
- The autocorrelation of a time series is the **Pearson correlation** between values of the time series at different times, as a function of the two times or of the time lag
- **Autocorrelation function (ACF)**: $\rho_X(k) = \text{Corr}(X_{t-k}, X_t)$
- **Partial autocorrelation function (PACF)**: the correlation between the time series and its lag after excluding the contributions from the intermediate lags
 - The partial autocorrelation at lag k is the autocorrelation between X_t and X_{t-k} that is not accounted for by lags 1 through $k-1$
- The difference between ACF and PACF is the inclusion or exclusion of **indirect correlations** in the calculation

ACF plot

The ACF plot can provide answers to the following questions:

- Is the observed time series **white noise/random**?
- Is an observation related to an adjacent observation, an observation twice-removed, and so on?

- The first value and the 24th value have a high autocorrelation
- The 12th and 36th observations are highly correlated
- We will find a very similar value at every 24 unit of time

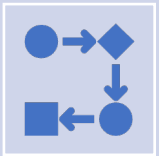


- The plot looks like sinusoidal function
 - This is a hint for seasonality
 - We can find its value by finding the period in the plot which would give 24h

Main takeaway



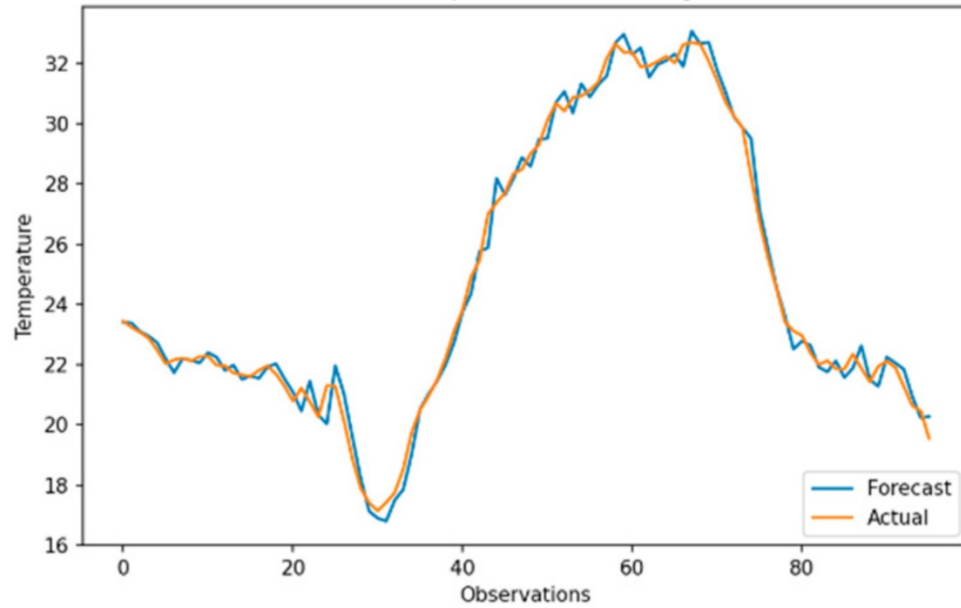
Before modelling, we must first to check for **Stationarity**, **Seasonality** and **Autocorrelation**



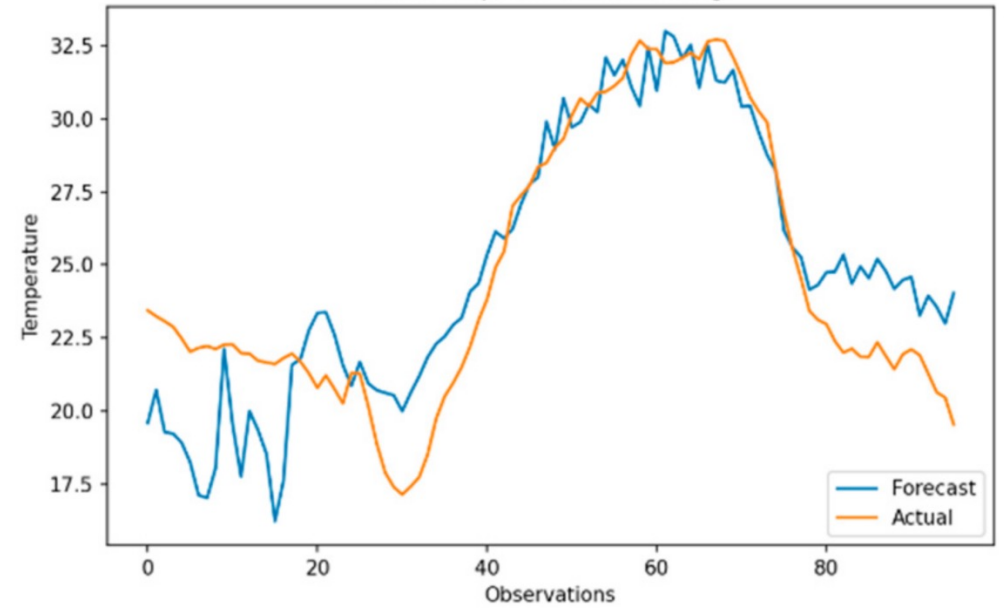
Then, we apply the right **transformations** to our time series to remove seasonality and any non-stationary behaviors

Time Series Forecasting

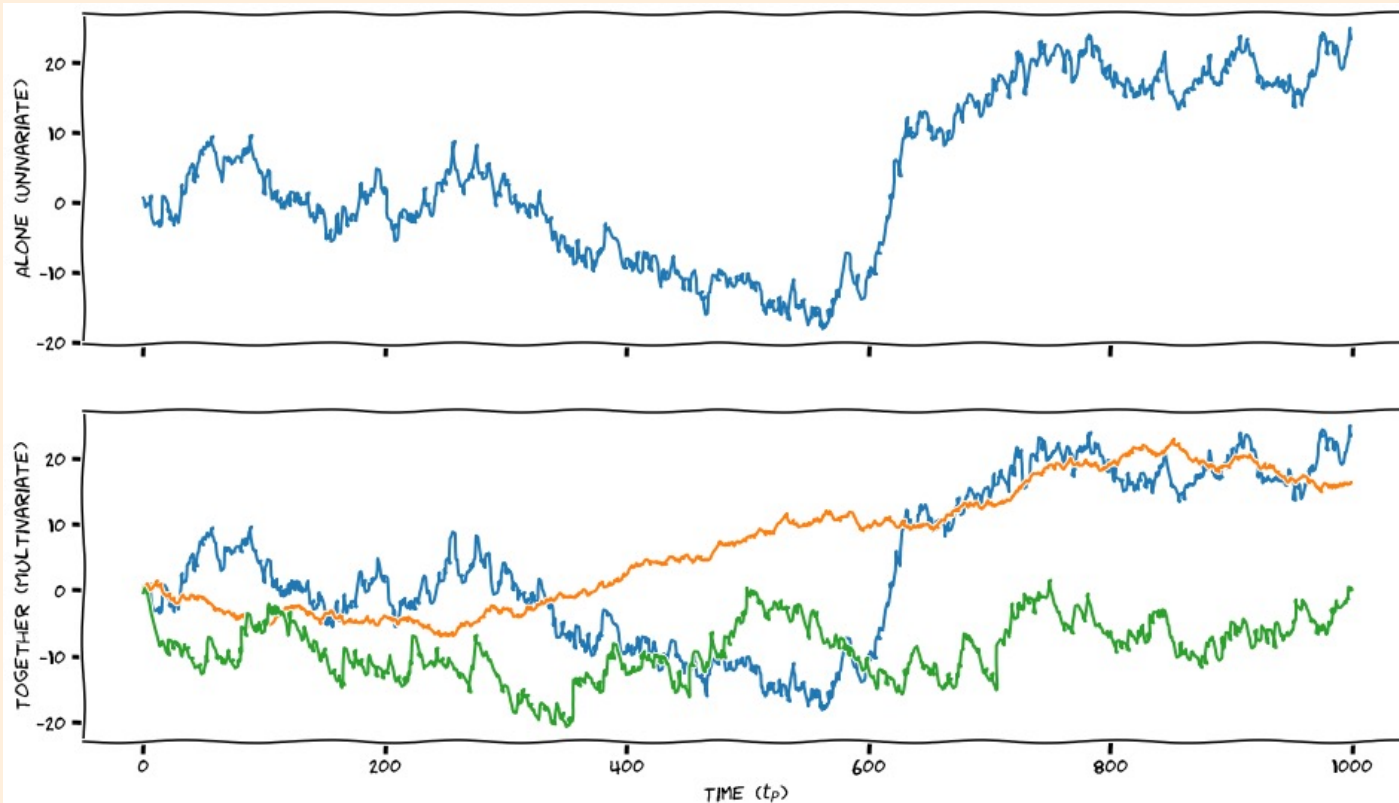
One-step Ahead Forecasting



Multi-step Ahead Forecasting



Univariate vs Multivariate TS



- Time series can be classified as **univariate** and **multivariate**
- It is an important distinction as the **selection** of the forecasting method depends, among other factors, on this
- **Univariate time series** is observation of one variable over time
- **Multivariate time series** is a collection of several univariate time series where a series can impact others

Methods for Time Series Forecasting

- **Statistical Methods**
 - extract relevant information from historical data
 - inference about the relationships between variables and their significance
 - highly interpretable
- **Traditional Machine Learning Methods**
 - accurate predictions
 - sacrifice interpretability for predictive power
- **Deep Learning Methods**
 - accurate predictions
 - added complexity – sacrifice interpretability
 - learn features and dynamics from the data – no need for feature engineering

Outline

Intro to Time Series Forecasting

Statistical Forecast Methods

Machine Learning Methods

Deep Learning Models

Evaluation Metrics

Nowcasting





Statistical Models

- AutoRegression (AR)
- Moving Average (MA)
- AR(I)MA
- ARIMAX
- SARIMA(X)
- VARMA(X)
- Exponential Smoothing
- Holt Winter's Exponential Smoothing
- ...

Autoregression (AR)

- AR states that the next observation depends on the past observations with some lag p which represents the **maximum lag**
- Linear function of its past values plus a random noise/error
- We assume that X_t is **stationary**
- **$AR(p)$** : autoregressive model of **order p**

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \dots + \phi_p X_{t-p} + w_t$$

where $\phi_1, \phi_2, \dots, \phi_p$ are constants

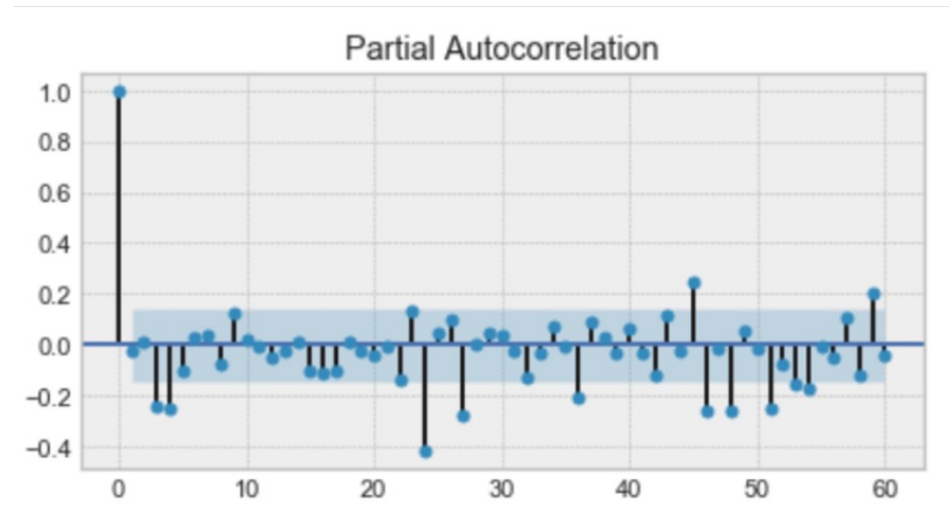
Moving Average (MA)

- **MA** simply states that the next observation is the mean of the past observations
- It uses the **last t time points** in order to predict time point **$t + 1$**
- Values are generated by a normal distribution with 0 mean
- Linear combination of the past values:

$$X_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \dots + \theta_q \varepsilon_{t-q}$$

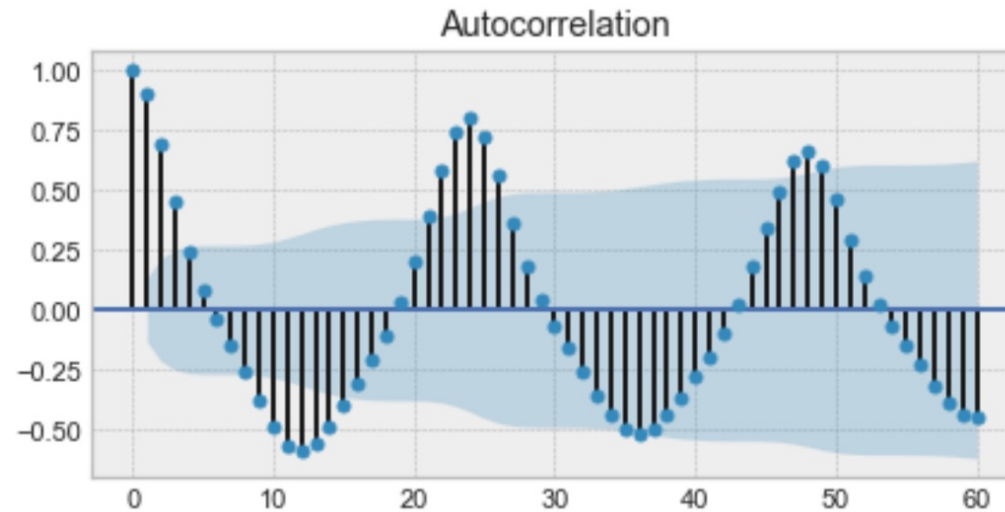
ARIMA

- At first, we have the autoregression model $AR(p)$
 - This takes a parameter p which represents the maximum lag after which most lags are not significant



ARIMA

- At first, we have the **autoregression model $AR(p)$**
- Then, we add the **moving average model $MA(q)$**
 - This takes a **parameter q** which represents the **biggest lag** after which other lags are not significant on the autocorrelation plot



ARIMA

- At first, we have the autoregression model $AR(p)$
- Then, we add the moving average model $MA(q)$
- After, we add the order of integration $I(d)$
 - The parameter d represents the number of differences required to make the series stationary, because a model cannot forecast on non-stationary time series data

SARIMA

- At first, we have the autoregression model $AR(p)$
- Then, we add the moving average model $MA(q)$
- After, we add the order of integration $I(d)$
- Finally, we add the final component: seasonality $S(P, D, Q, s)$, where s is simply the season's length
 - Furthermore, this component requires the parameters P and Q which are the same as p and q , but for the seasonal component
 - Finally, D is the order of seasonal integration representing the number of differences required to remove seasonality from the series
- **$SARIMA(p, d, q)(P, D, Q, s)$ model**

(S)ARIMAX

- But ARIMA models do not fully **model reality**
- There are often other **exogenous** variables that affect the time series
- Another class of models that incorporated ARIMA with the **explanatory variables** approach of standard econometrics
- **ARIMAX**: ARIMA with additional explanatory variables provided by economic theory
 - X: exogenous variable(s)

Box-Jenkins Methodology

- Box-Jenkins is a set of approaches for time series analysis and for finding out the best fit for ARIMA models
- It refers to the iterative application of the following three steps:
 - **Identification:** A class of simple ARIMA models is selected using plots of the data, autocorrelations, partial autocorrelations, and other information. This amounts to estimating appropriate values for p , d , and q .
 - **Estimation:** The phis and thetas of the selected model are estimated using maximum likelihood techniques, backcasting, etc.
 - **Diagnostic Checking:** The fitted model is checked for inadequacies by considering the autocorrelations of the residual series (the series of residual/error values).
- These steps are applied iteratively until step three does not improve the model any further

M-competitions

- **Time series forecast competitions** that were held starting from 1982 organized by Spyros Markidakis
- Groups of experts try to apply different forecasting methods to time series to compare the accuracy and uncertainty estimation
- The results of the **M4 competition** (2018) showed that the pure ML methods were largely outperformed by **the traditional statistical approaches**
- However, in the most recent M5 competition (2020), with a more creative dataset, the top spot submissions featured only **Machine Learning (ML) and Deep Learning (DL) methods**
 - This competition saw the rise of LightGBM (used for time series forecasting) and the debut of Amazon's DeepAR and N-BEATS

Time

Limitations

- The observations from the M4 competition hold but with serious **limitations**:
 - The amount of data should be limited
 - The time series should be univariate
 - The time series should be stationary or at least not too volatile due to external factors
- But in reality, in the era of Big Data, we rather face **large datasets** than small, time series in most real world cases are **multivariate** and **external factors** influence time series more often than not influence (pandemic, economic crisis, etc.)
- Therefore, the **future** is with ML and DL time series forecasts methods with classical methods being organically integrated rather than opposed to it

Outline

Intro to Time Series Forecasting

Statistical Forecast Methods

Machine Learning Methods

Deep Learning Models

Evaluation Metrics

Nowcasting



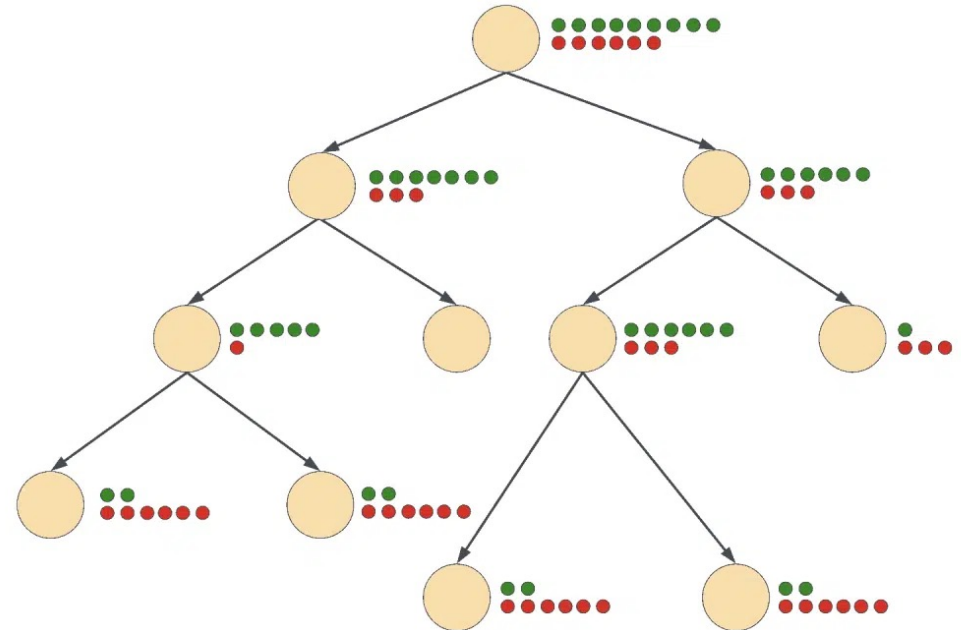


Machine Learning Models

- Decision Tree
 - Random Forest
 - XGBoost
 - LightGBM
 - Support Vector Regression (SVR)
- Linear Regression
 - Ridge Regression
 - Lasso Regression
 - Elastic Net
 - ...

Decision Tree

- Decision tree builds **regression** or **classification** models in the form of a tree structure
- A regression tree is basically a decision tree that is used for predicting **continuous valued outputs** instead of discrete outputs
- Due to tree-like structure, decision trees can capture **non-linear relationship** between variables and therefore can provide good results in forecasting complex data



Decision Tree

- To split the nodes at the most informative features, we need to define an **objective function** that we want to optimize via the tree learning algorithm
- We want to maximize the **information gain** at each split, which we define as follows:

$$IG(D_p, f) = I(D_p) - \left(\frac{N_{left}}{N_p} I(D_{left}) + \frac{N_{right}}{N_p} I(D_{right}) \right)$$

- Need for an **impurity metric** I that is suitable for continuous variables, so we define the impurity measure using the mean squared error (MSE) of the children's nodes:

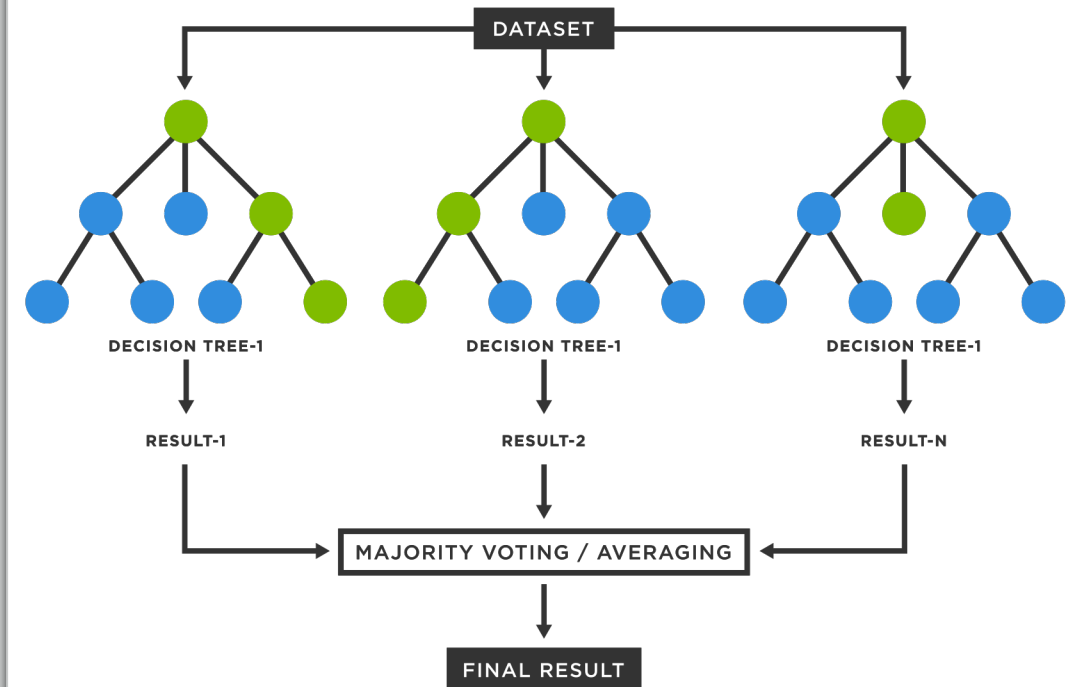
$$MSE(t) = \frac{1}{N_t} \sum_{i \in D_t} (Y^{(i)} - \hat{Y}_t)^2$$

Decision Tree

- Decision trees are very easy to understand and **interpret** because they can be visualized
- However, they are prone to **overfitting**
- A decision tree can be considered as a **weak learner**, which means that the model is simple and has limited predictive power

Random Forest

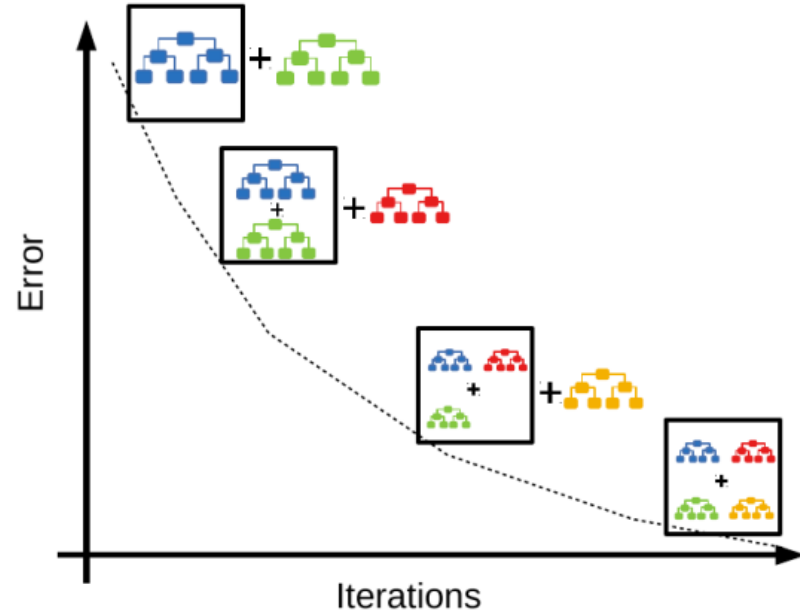
- **Ensemble methods** in ML are based on combination of outcomes of many weak learners
- Random Forest is called a forest because it grows a **forest of decision trees**
- Thus, many decision trees can be built using the same dataset, and the final outcome is the **average** of the predictions of all trees



Random Forest

- **Bootstrapping**: Random Forest randomly performs row sampling and feature sampling from the dataset forming sample datasets for every model
- Random Forest is very versatile
 - It can be used for both classification and **regression** tasks
 - It can easily handle binary and numerical features as well as categorical ones, with no need for transformation or rescaling
 - Unlike other models, it is incredibly efficient with all types of data
- Random forest **prevents overfitting** by building different sized trees from subsets and combining the results
- Similarly to Decision Trees, the **loss function** for the model can be MSE

Gradient Boosting



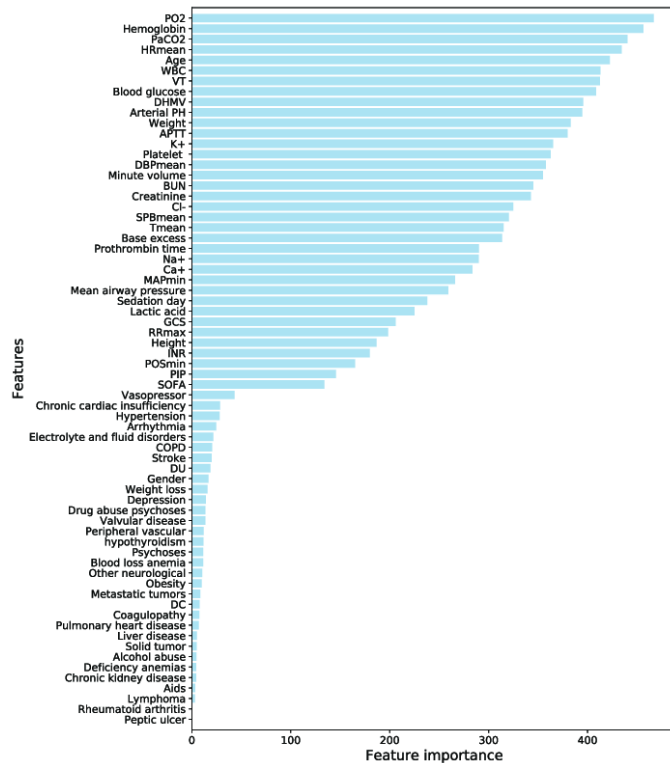
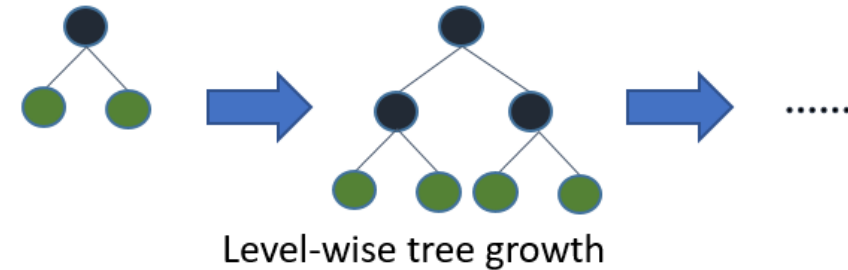
- It is an ML methodology where an **ensemble** of weak learners is used to improve the model performance in terms of efficiency, accuracy, and interpretability
 - These learners are defined as having better performance than random chance
 - Such models are typically **decision trees** and their outputs are combined for better overall results
- The hypothesis is to filter out instances that are **difficult** to accurately predict and develop new weak learners to handle them

eXrteme Gradient Boosting (XGBoost)

- XGBoost is built on top of **gradient boosted ensemble** of decision trees
- XGBoost is often used for time series analysis due to its ability to handle **non-linear relationship** between features
- It also includes **regularisation** techniques such as L1 and L2 to prevent overfitting
- Tree pruning using depth-first approach
- Utilization of all the available cores of a CPU during tree construction for **parallelization**

eXrteme Gradient Boosting (XGBoost)

- Level-wise (horizontal) growth



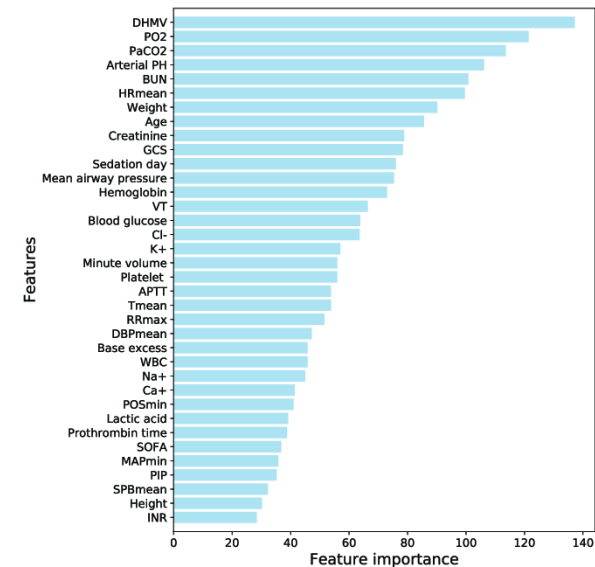
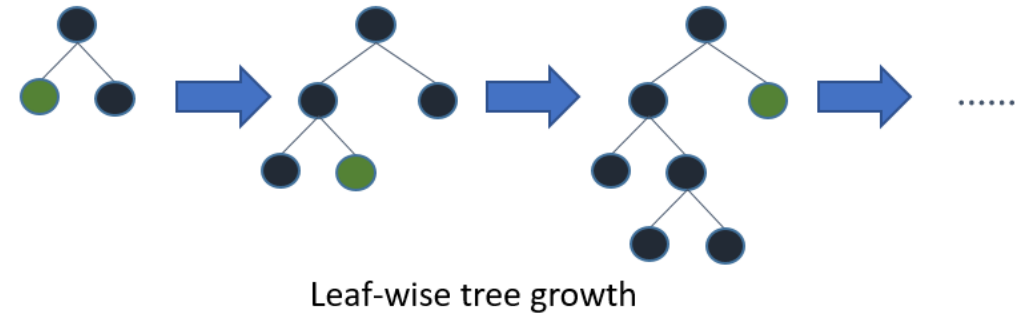
- XGBoost can calculate the importance of every feature, which can be used for feature engineering and **explainability** of the model

LightGBM

- LightGBM by Microsoft is a distributed high-performance framework that uses **decision trees** for ranking, classification, and regression tasks
- **Histogram-based algorithm** that performs bucketing of values, that also requires lesser memory
- Support for both **parallel learning** and **GPU learning**
- **Faster rate of execution** along with being able to maintain **good accuracy** levels primarily due to the utilization of two novel techniques:
 - **Gradient-Based One-Side Sampling (GOSS)**: data instances with larger gradients contribute more towards information gain and in addition, it performs random sampling on instances with smaller gradient
 - **Exclusive Feature Bundling (EFB)**: a near lossless method to reduce the number of effective features

LightGBM

- In contrast to the level-wise (horizontal) growth in XGBoost, LightGBM carries out **leaf-wise (vertical) growth** that results in more loss reduction and in turn higher accuracy while being faster
- LightGBM can also calculate the **importance** of every feature for making the predictions



Outline

Intro to Time Series Forecasting

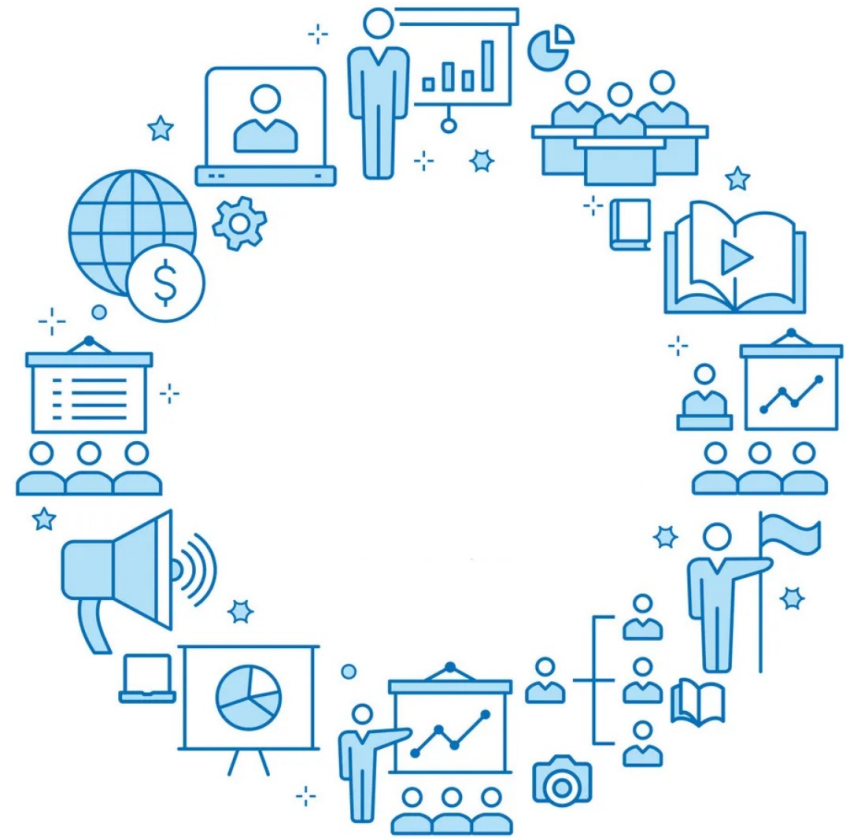
Statistical Forecast Methods

Machine Learning Methods

Deep Learning Models

Evaluation Metrics

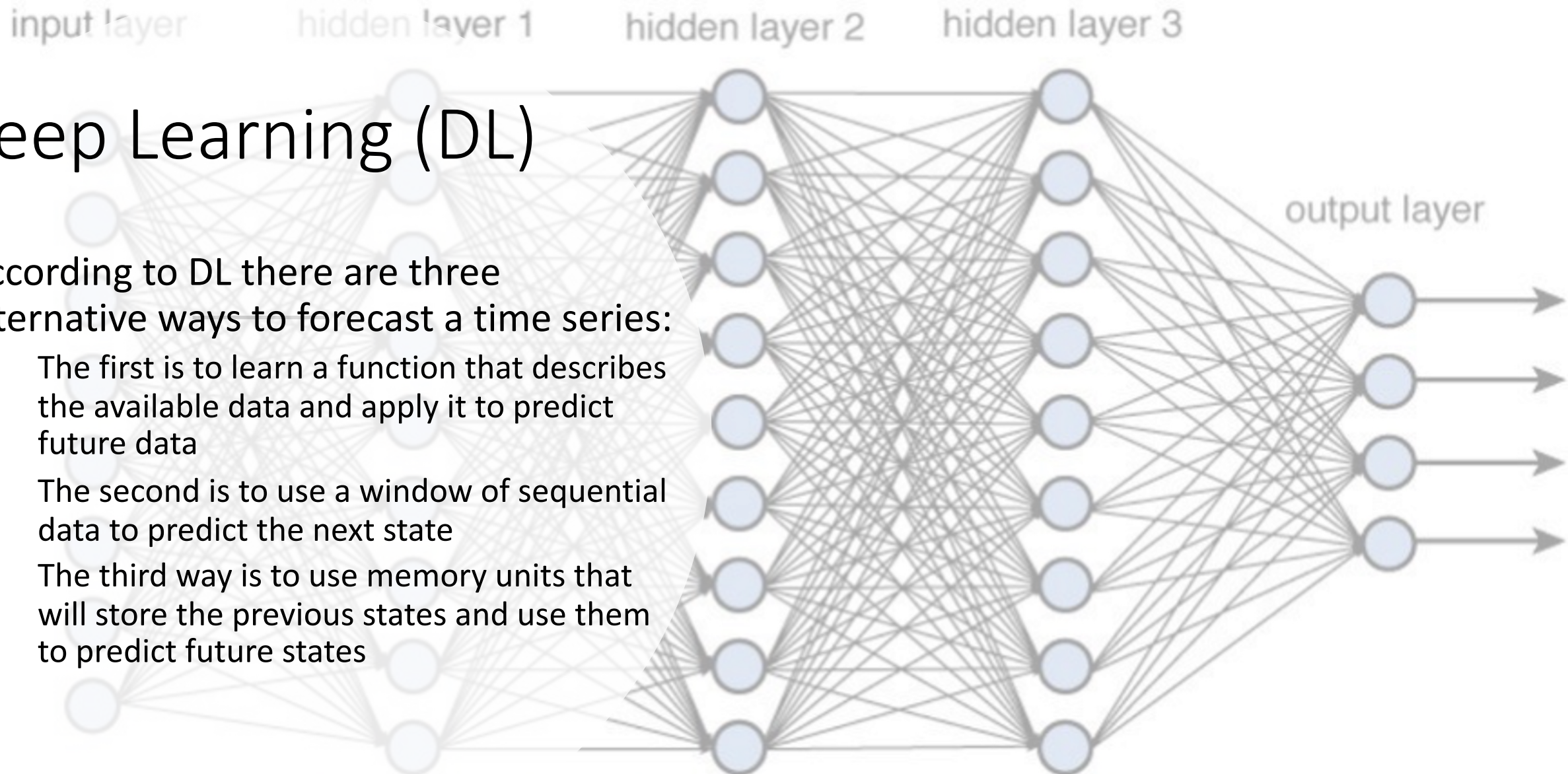
Nowcasting

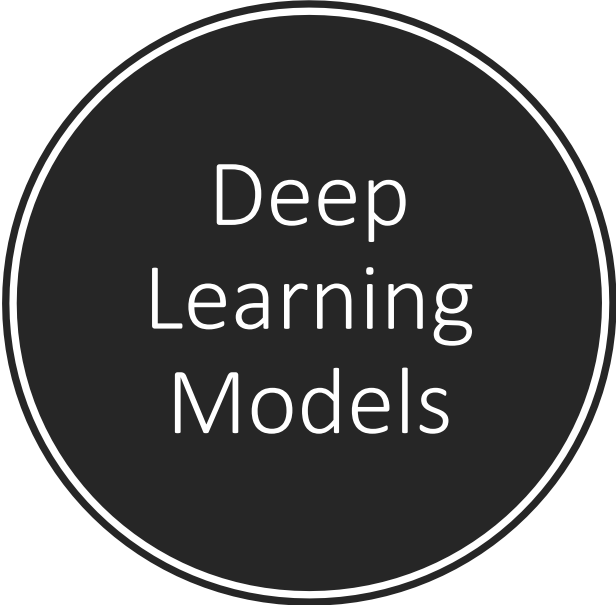


Deep Learning (DL)

According to DL there are three alternative ways to forecast a time series:

- The first is to learn a function that describes the available data and apply it to predict future data
- The second is to use a window of sequential data to predict the next state
- The third way is to use memory units that will store the previous states and use them to predict future states



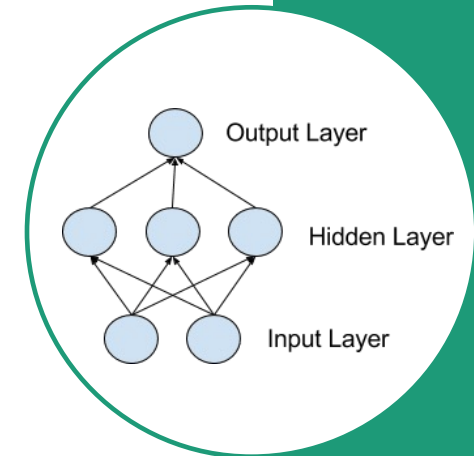


Deep Learning Models

- Multilayer Perceptrons (MLPs)
- Convolutional NNs (CNNs)
- Recurrent NNs (RNNs)
 - LSTM, GRU
- N-BEATS
- DeepAR
- Encoder-Decoder Model
- Attention Mechanism
- ...

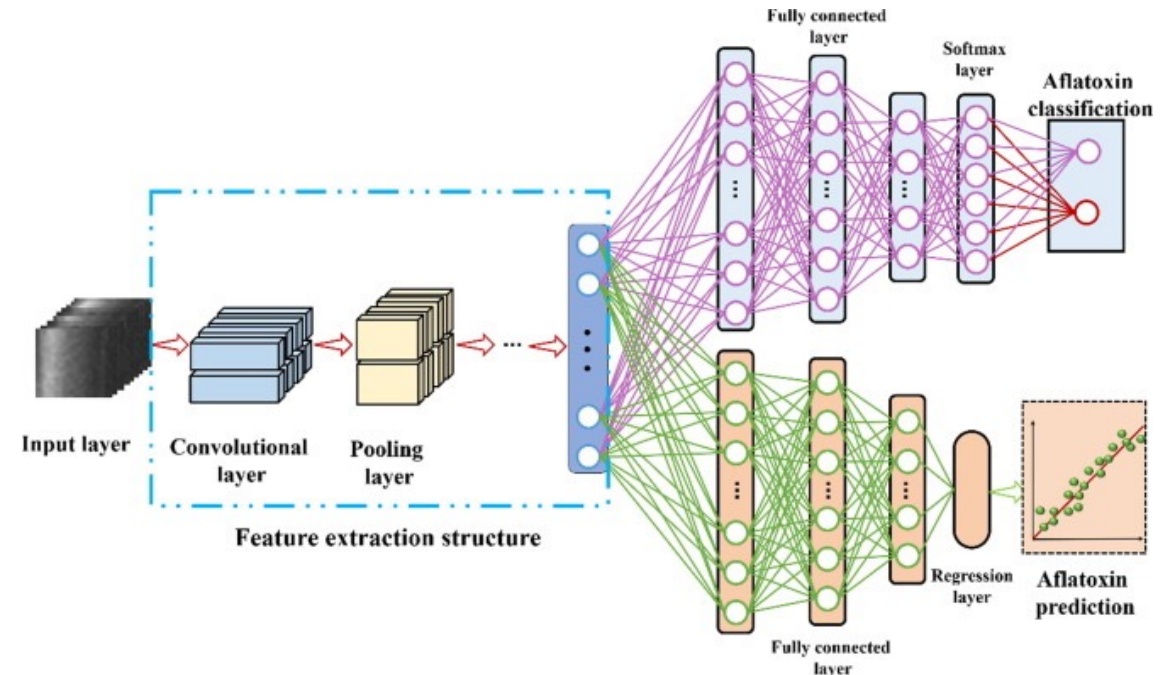
Multilayer Perceptrons (MLPs)

- MLPs are the **classical** type of Neural Network
- MLP models are **memory-less**, and they use the feed forward neural network architecture, which applies the back propagation algorithm for training the neural network
- Usually **fully-connected** networks, each neuron in one layer is connected to all neurons of the next layer
 - “full connectivity” makes them prone to **overfitting**
- MSE as the **regression loss**



Convolutional Neural Networks (CNNs)

- CNNs are **regularized** versions of multilayer perceptrons
- A CNN is a neural network with **convolutional layers**
 - A convolutional layer has a number of filters that perform the operation of **convolution** (an integral that expresses the amount of overlap of one function as it is shifted over another function)
- They take advantage of **local and hierarchical patterns** in the data



Implementing a CNN for **regression** prediction is as simple as:

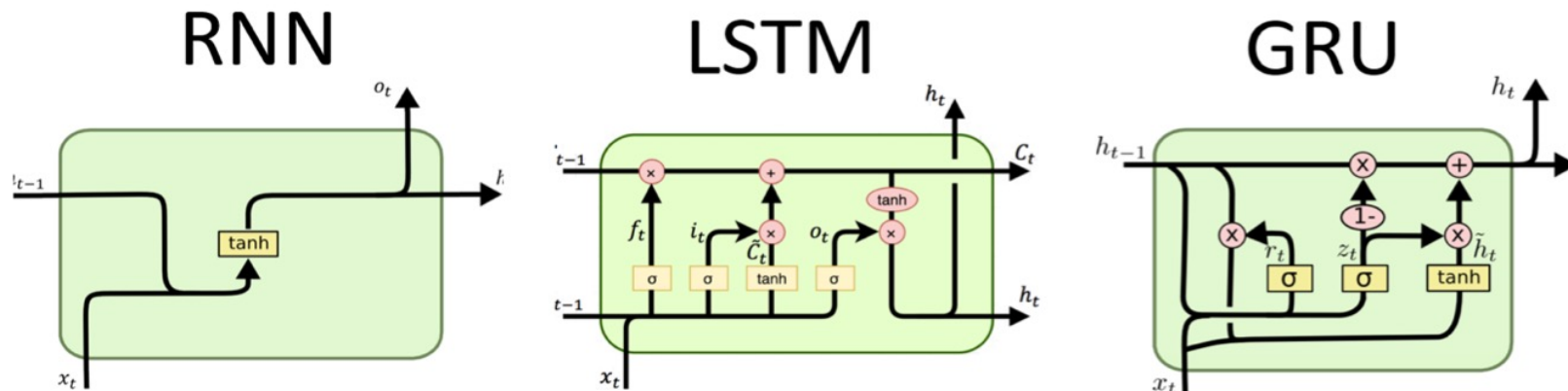
- Replacing the fully-connected softmax classifier layer (classification) with a fully-connected layer with a single node along with a linear activation function
- Training the model with a continuous value prediction loss function such as MSE

Recurrent Neural Networks (RNNs)

- RNNs are deep learning models, typically used to solve problems with **sequential input data**, such as time series
- They can have **multiple hidden layers** to solve more complex sequential problems
- They retain a **memory** of what it has already processed and, therefore, a sense of time
 - Achieved by implementing several neurons receiving as input the output of one of the hidden layers and injecting their output again in that layer
- They share the **same weights** across several time steps

Gated RNNs

- The most effective RNNs are the **Gated RNNs**:
 - **Long Short-Term Memory (LSTM)**: more accurate with longer sequences
 - **Gated Recurrent Unit (GRU)**: faster than LSTM with shorter sequences
- They allow the network to **accumulate information** over a long duration
- Once this information has been used the neural network can **forget the old state**
- Instead of manually deciding when to **clear the state**, we want the neural network to learn to decide when to do it



Long Short-Term Memory (LSTM)

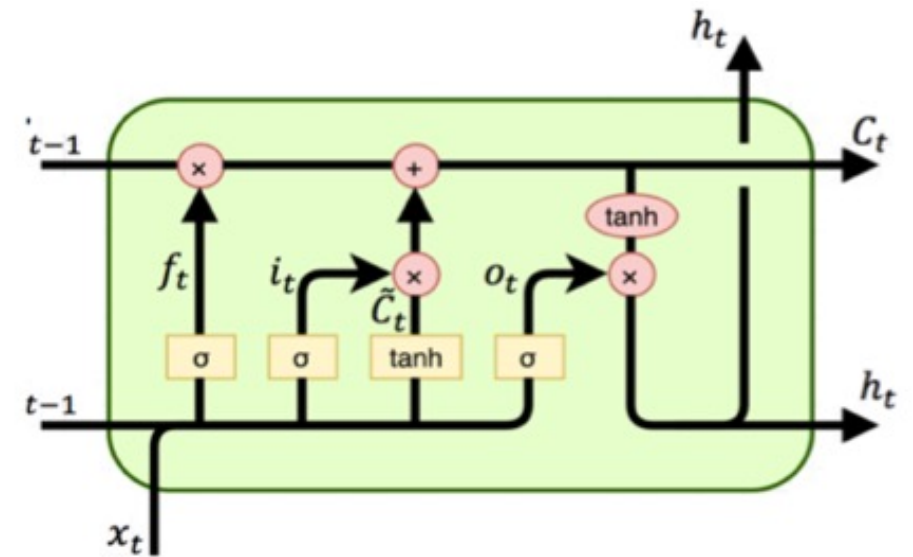
- LSTM is a type of recurrent neural network that can learn the **order dependence** between items in a sequence
- LSTM architecture allows it to be effective for time series forecasting due to its ability to capture **short term and long-term dependencies** and complex **non-linear** relationships
- It makes predictions according to the data of previous times
- LSTM works well for **multivariate time series** and can handle missing data
- Pre-processing data is a very important step for the performance of an LSTM

LSTM

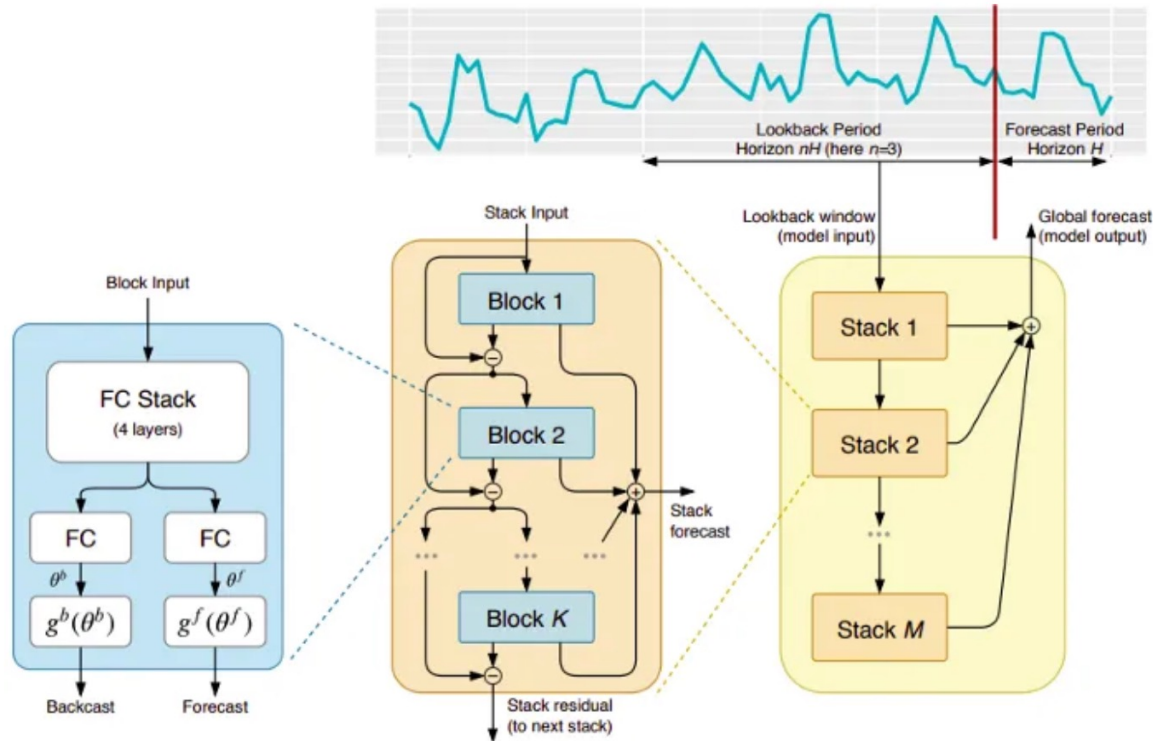
LSTM consists of **three gates** that control flow of information and maintain long-term dependencies in the data

- **Input gate:** responsible for the **information** that can enter the cell state
- **Forget gate:** controls how much of previous information should be **ignored** and how much should be **kept**
- **Output gate:** decides the information that should be sent to the **next state**

The combination of the gates allows LSTM to remember information over longer periods and therefore makes LSTM suitable for processing **time series** data



N-BEATS



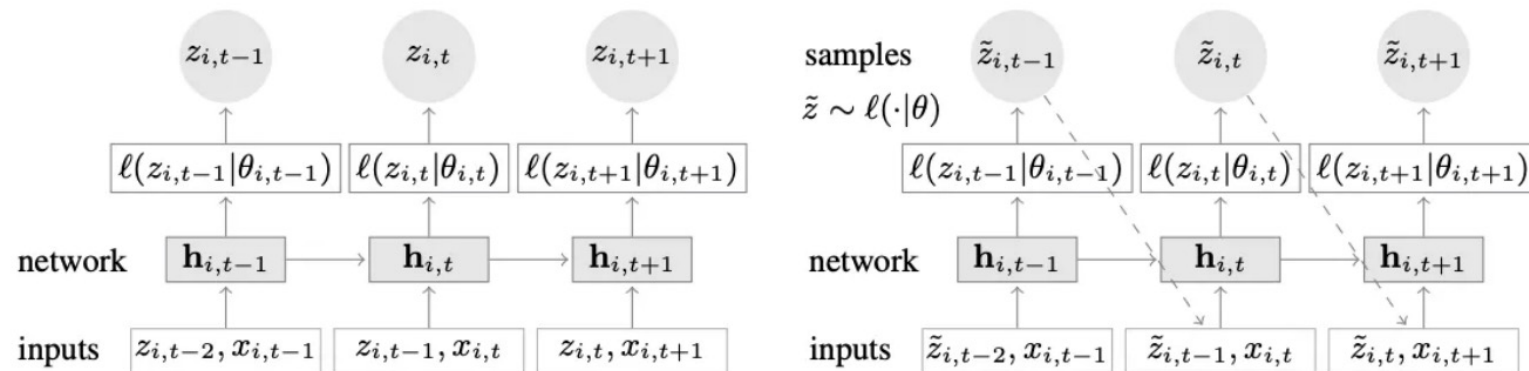
- N-BEATS is a custom DL algorithm which is based on backward and forward residual links for **univariate time series** point forecasting
- Essentially, N-BEATS is a pure DL architecture based on a deep stack of **ensembled feed forward networks** that are also stacked by interconnecting backcast and forecast links
- Each successive block models only the residual error due to the reconstruction of the backcast from the previous block and then updates the forecast based on that error
 - This process mimics the Box-Jenkins method when fitting ARIMA models

N-BEATS

- **Simple** to understand and has a **modular structure** (blocks and stacks)
- It has the ability to generalize on multiple time-series through **meta-learning**
 - **Inner learning**: takes place inside blocks and helps the model capture local temporal characteristics
 - **Outer learning**: takes place inside stacks and helps the model learn global characteristics across all time-series
- The model has two variants:
 - **General**: the final weights in the fully-connected layers of each block are learned by the network arbitrarily
 - **Interpretable**: the last layer of each block is removed, then the backcast and forecast branches are multiplied by specific matrices that mimic trend (monotonic function) and seasonality (periodic cyclical function)

DeepAR

- **DeepAR** developed by Amazon is a novel probabilistic forecasting model based on autoregressive RNNs
- It combines both DL and **autoregressive** characteristics
- No need to scale the time sequence using a normalization or standardization technique as in other DL models, since DeepAR **scales** the autoregressive input z of each time series i with a scaling factor v_i



DeepAR

- DeepAR works really well with **multiple time series**
 - A global model is built by using multiple time series with slightly different distributions
- Apart from historical data, DeepAR also allows the use of known **future time sequences** (a characteristic of auto-regressive models) and extra static attributes for series
- It makes **probabilistic forecasts** instead of directly outputting the future values
 - in the form of Monte Carlo samples
 - these forecasts are used to compute quantile forecasts, by using the **quantile loss function**, used to calculate not only an estimate, but also a prediction interval around that value

Outline

Intro to Time Series Forecasting

Statistical Forecast Methods

Machine Learning Methods

Deep Learning Models

Evaluation Metrics

Nowcasting

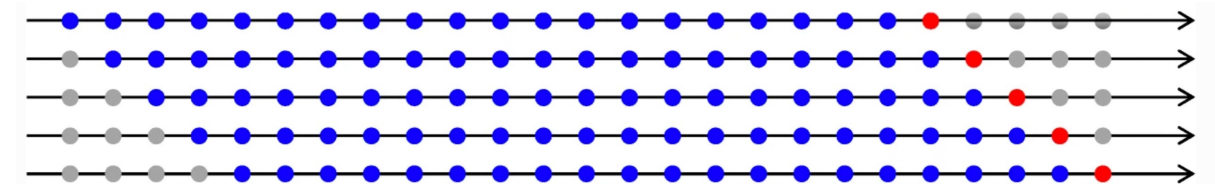
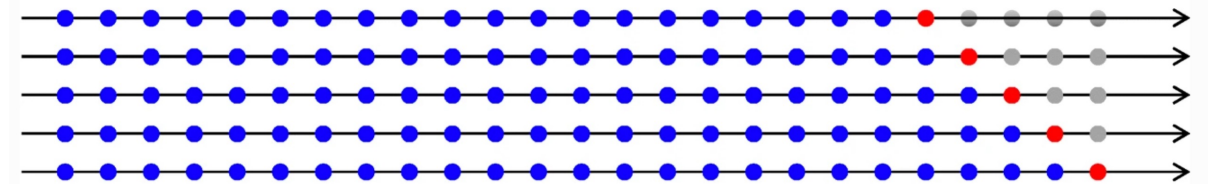


Rolling Forecast: Cross validation & Predictions

1. Minimum number of observations
required to train the model

2. Sliding window:

- the model will be trained on all the data it has available
- the model will be trained only on the most recent observations





Evaluation Metrics

- We cannot calculate accuracy for a forecasting model. The performance of the model must be reported as an **error** for the predictions
- Four commonly used error metrics for evaluating and reporting the performance of a forecasting model:
 - **Mean Squared Error (MSE)**
 - **Root Mean Squared Error (RMSE)**
 - **Mean Absolute Error (MAE)**
 - **Mean Absolute Percentage Error (MAPE)**
- In addition, we calculate the **correlation** between the predictions and the real values:
 - **Pearson's Correlation**
 - **Spearman's Rank Correlation**

Mean Squared Error

- **MSE** is calculated as the mean or average of the squared differences between predicted and expected target values in a dataset

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

- The difference between these two values is squared, which has the effect of removing the sign, resulting in a **positive error value**
- The squaring also has the effect of **inflating** or magnifying **large errors**; this has the effect of “punishing” models by inflating the average error score when used as a metric

Root Mean Squared Error

- **RMSE** is calculated as the square root of the MSE, which means that the **units of the error** are the **same** as the **units of the target** value that is being predicted:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2}$$

- It may be common to use MSE loss to train a regression predictive model, and to use RMSE to evaluate and report its performance
- MSE uses the square operation to remove the sign of each error value and to punish large errors; the **square root** reverses this operation, although it ensures that the result remains positive

Mean Absolute Error

- **MAE** is calculated as the average of the absolute error values, and like RMSE, the **units of the error** score **match** the **units of the target** value that is being predicted:

$$MAE = \frac{1}{n} \sum_{i=1}^n |Y_i - \hat{Y}_i|$$

- Unlike the RMSE, the **changes** in MAE are **linear** and therefore **intuitive**; the MAE does not give more or less weight to different types of errors (like MSE and RMSE) and instead **the scores increase linearly** with increases in error

Mean Absolute Percentage Error

- **MAPE** is the percentage equivalent of MAE. Each residual $(Y_i - \hat{Y}_i)$ is scaled against the actual value:

$$MAPE = \frac{100\%}{n} \sum_{i=1}^n \frac{|Y_i - \hat{Y}_i|}{Y_i}$$

- MAPE has a **clear interpretation** since percentages are easier for people to conceptualize
- Both MAPE and MAE are **robust** to the **effects of** outliers thanks to the use of absolute value
- Since everything is scaled by the actual value, MAPE is **undefined** for data points where the value is 0; similarly, the MAPE can grow **unexpectedly large** if the actual values are exceptionally small

Pearson's Correlation Coefficient

- It attempts to establish a **line of best fit** between two variables
- The resulting Pearson's correlation coefficient indicates **how far away** the actual values are from the mean values:

$$r_{XY} = \text{corr}(X, Y)$$

- The value of r_{XY} ranges between -1 and $+1$
 - the closer it is to either -1 or 1 , the stronger the correlation between the variables
 - as it approaches 0 there is less of a relationship
- If X and Y independent: $r_{XY} = 0$
 - X and Y independent and uncorrelated

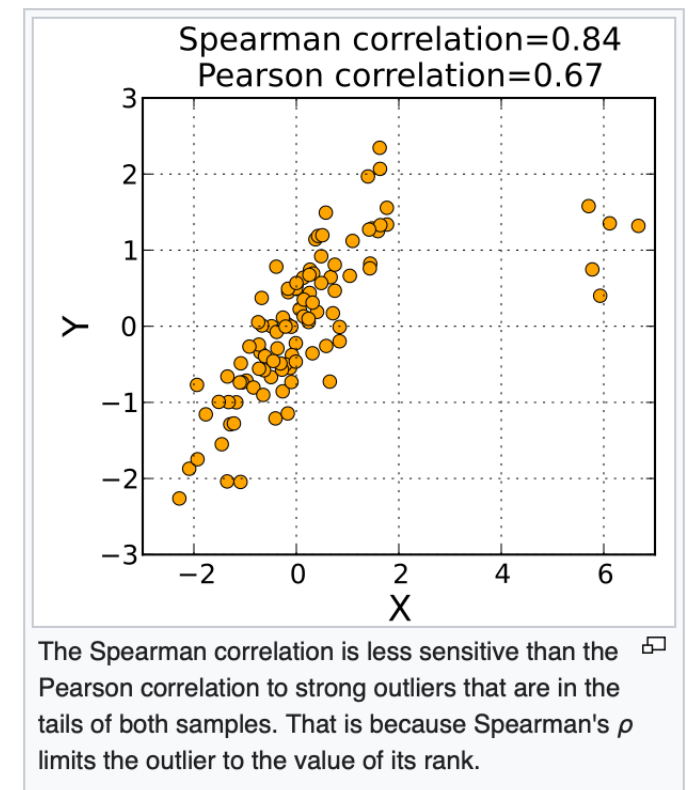
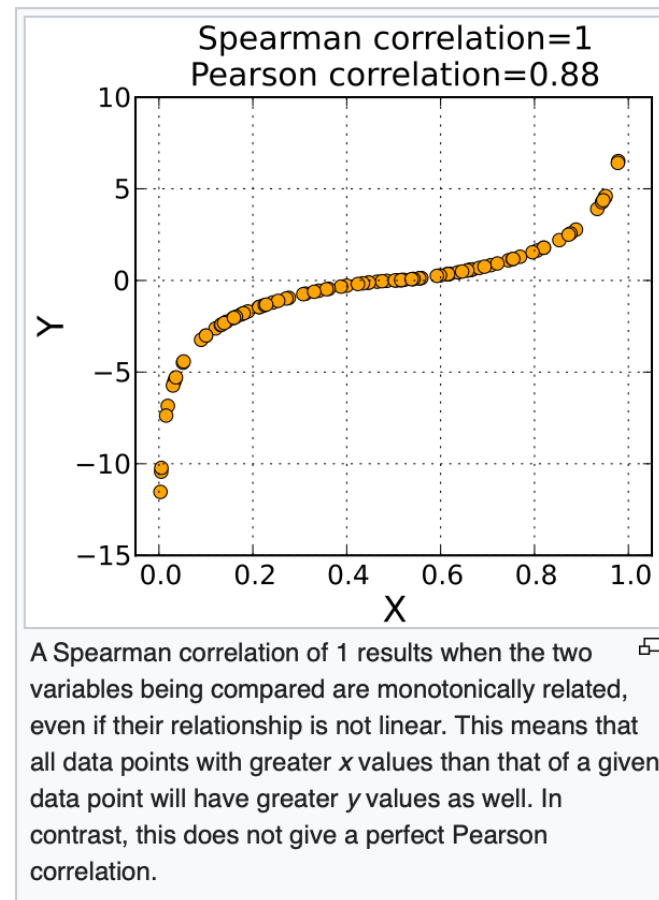
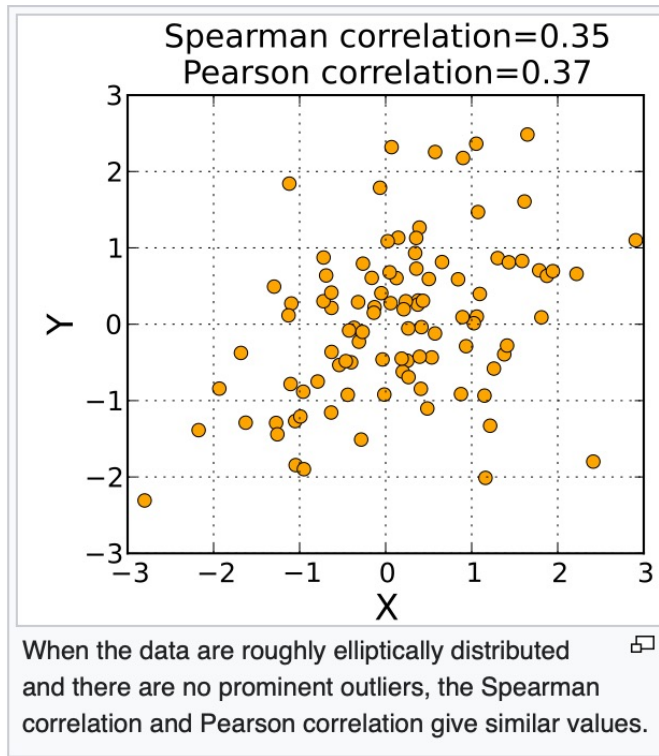
Correlation Coefficient Value	Indication
± 0.8 to ± 1.00	High correlation
± 0.6 to ± 0.79	Moderately high correlation
± 0.4 to ± 0.59	Moderately correlation
± 0.2 to ± 0.39	Low correlation
± 0.1 to ± 0.19	Negligible correlation

Spearman's Rank Correlation Coefficient

- It measures the **extent** to which, as one variable **increases**, the other variable tends to increase, without requiring that increase to be represented by a linear relationship
- The Spearman correlation coefficient is defined as the Pearson correlation coefficient between the **rank variables** $R(X)$ and $R(Y)$:

$$r_s = r_{R(X),R(Y)}$$

- If, as the one variable increases, the other decreases, the rank correlation coefficient will be **negative**



Pearson's Vs Spearman's

Outline

Intro to Time Series Forecasting

Statistical Forecast Methods

Machine Learning Methods

Deep Learning Models

Evaluation Metrics

Nowcasting



Nowcasting

- Predict the **short-term future** values given past observations of several indicators
- Early detection of a **critical event**
- Widely used by epidemiologists since they are interested in **timelier forecasts** of disease spreading



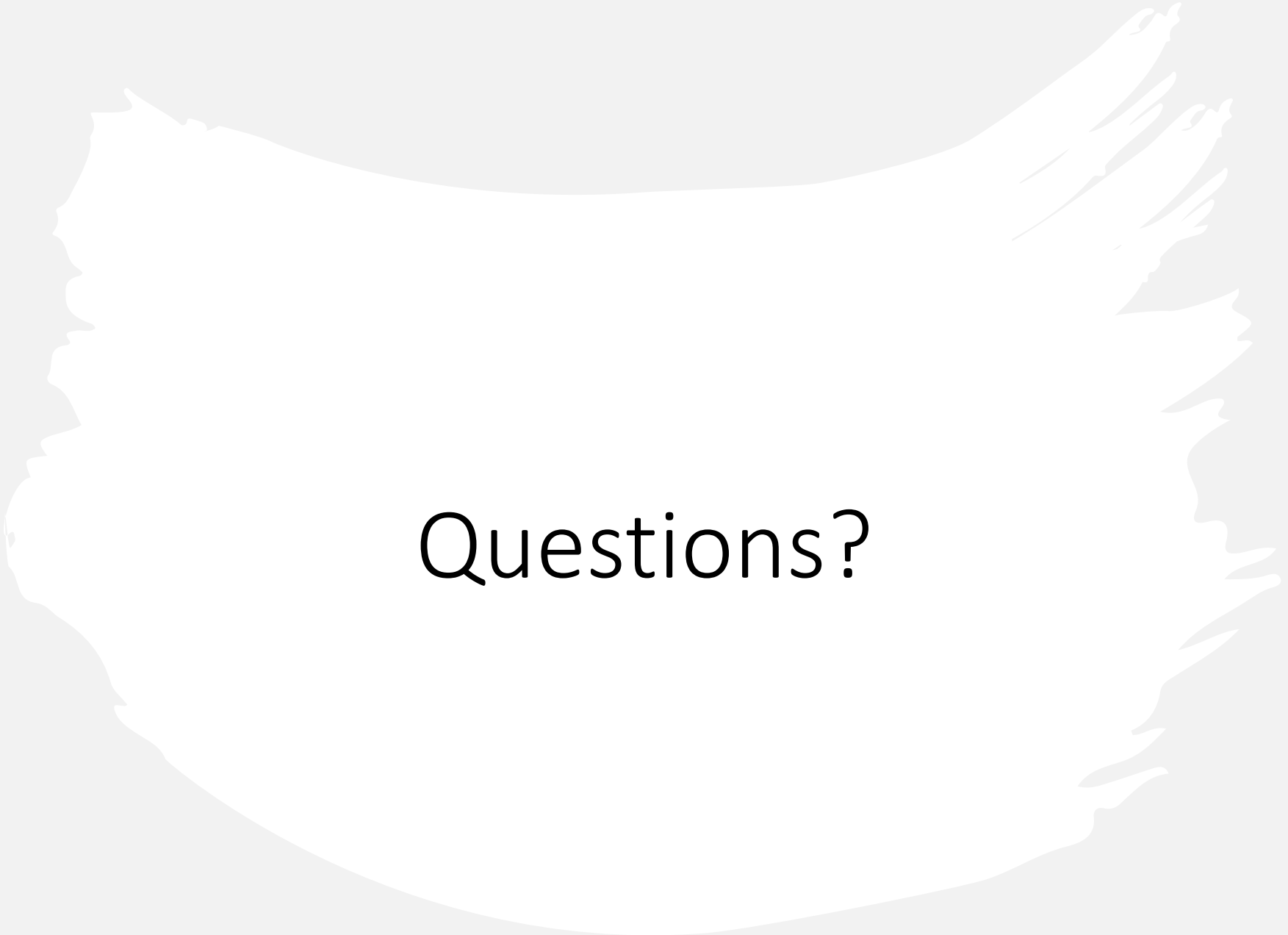
Nowcasting problem

- Time **dependent** variable $Y(t)$ that cannot be measured instantaneously
- Value of $Y(t)$ available at time $t+d$, where d is the delay
- We approximate the value of $Y(t)$ at time t , based on a set of n **independent** variables (**predictors**) $X_1(t), X_2(t), \dots, X_n(t)$, which can be measured at time t or time $t+d_{x_{1,2,\dots,n}}$ where $d_{1,2,\dots,n} < d$



Google Flu Trends

- Search data can help predict the incidence of influenza-like diseases
- Close relationship between number of people searching for flu-related topics and how many people have symptoms
- Prediction models compared to real-world cases of flu



Questions?

A decorative background element consisting of a stylized, light blue leafy branch that curves from the bottom left towards the top right, partially obscuring the text.

Thank you

Ioanna Miliou, PhD

ioanna.miliou@dsv.su.se



Stockholm
University