Deep Learning para series temporales

Part II

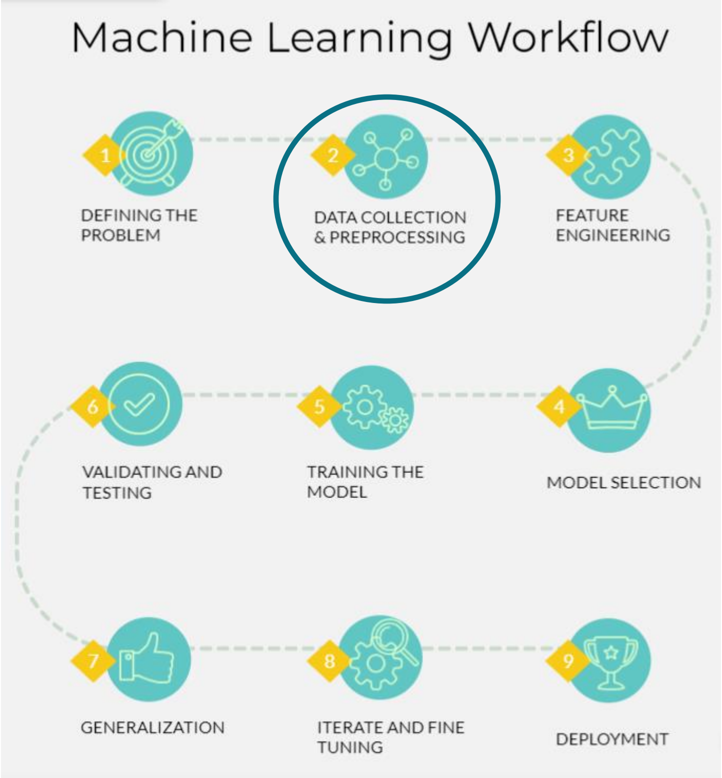
Preprocessing and Analysis

Contents

ETL

Preprocess the data

ML Workflow



https://www.linkedin.com/pulse/data-collection-preprocessing-dr-john-martin-5kj3f/

ETL (Extract + Transform + Load)

- Extract

Máster Deep Learning

ETL (Extract + Transform + Load)

- Transform

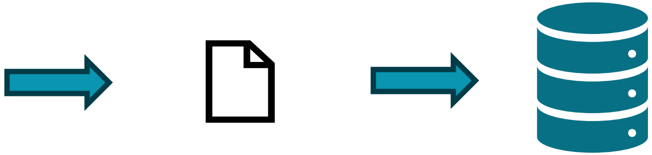
DataFrame & ndarray

Analyze ⇄ Transform as needed

ETL (Extract + Transform + Load)

- Load

DataFrame



Extract

Get & Open de data

Extract

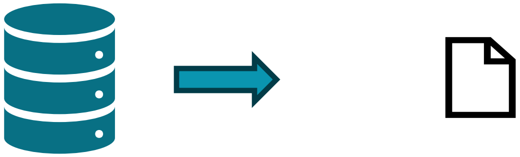
- **Files**: owned folder
- **Database**: SQL, Cassandra, MongoDB, PostgretSQL, Weight and Biases
- **Specific webpages**: Kaggle, zenodo, INE, HuggingFace
- **Other's datasets**: Github, GoogleDrive, e-mail, personal webpage

Extract

- Extract

Extract | Download

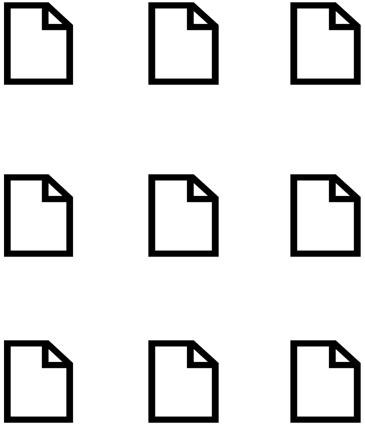
- Extract
  - **Download**
    - Can I use wget?
    - Is there an API?
      - Gdrive, OneDrive
      - Kaggle, Zenodo
      - Hugging Face
      - Weight and Biases
    - Is there a download button?



.csv, .txt, .tsf

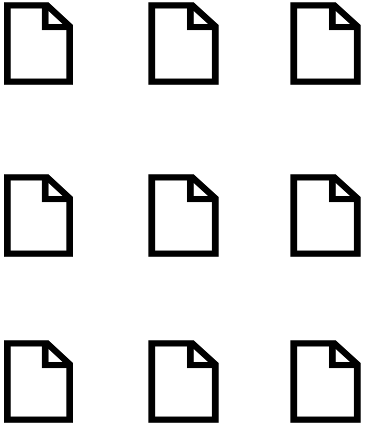
Extract | Take a look |

- Extract
  - Download
  - **Take a look to the files**
    - Are there duplicated files (different extensions, same data)?
    - Can I open the file? Is it corrupted? The type is adequate? Need to unzip/unrar?
      - Decompress
      - Use ! cat or open as normal for checking



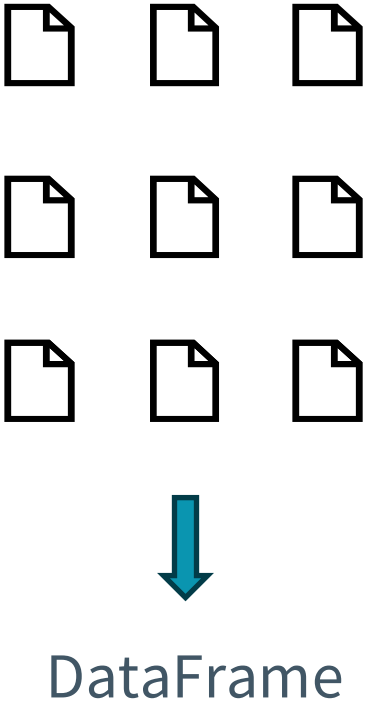
Extract | Take a look II

- Extract
  - Download
  - **Take a look to the files**
    - Are there multiple same-type files?
    - Do I need one or all of them?
    - Can I join the files?
      - Same information
      - Compatible indexes or rows that allow differencing datasets



Extract | Extract & Join |

- Extract
  - Download
  - Take a look to the files
  - **Extract & join**
    - Pandas IO tools
      - read_<csv, parquet, Excel, sql, json, hdf, table>
    - Pyarrow, tslearn
  - pd.concat([df1, df2, …])



Transform

Analyze and preprocess the data

Transform | Second Look

In this step we want to get a first touch with the time series data

- What columns does the dataFrame has?
- What variables do we want to analyze?
- Is the dataFrame well-structured for the analysis?
  - Time-index:
    - time in separated columns? Not the index?
  - Columns with “single data” (not composed strings)
  - Type of the data
  - Columns with unique value in all rows

Transform | Build the dataFrame |

Now we know a bit of our data, finish building your dataset

- Drop unnecessary columns
  - pd.drop
- Split columns with concatenated data
  - str.split(separator, expand = True)
- Ensure datatypes.astype(<type)>
- Filter the data you need
  - df = df.loc[df [<condition>]]
  - df = df.loc[df ["<colname>"] == <value we need>]

Máster Deep Learning

Transform | Build the dataFrame II

Now we know a bit of our data, finish building your dataset

- Ensure date is a timestamp and use it as index
- Ensure your data is a columna of the dataframe

In this part you should use **pd.melted** for pivoting the dataFrame

Transform | Build the dataFrame III

Now we know a bit of our data, finish building your dataset

- Is your data evenly spaced? How do you fix it?

Transform | Example | Non evenly -> evenly spaced |

4h
4h
3h 30 min
30 min
4h

Time	Available food (grs)
7:00 AM	**100**
11:00 AM	**50**
03:00 PM	**0**
06:30 PM	**100**
07:00 PM	**50**
11:00 PM	**0**

Non evenly spaced

Transform | Example | Non evenly -> evenly spaced II

4h
4h
3h 30 min
30 min
4h

Time	Available food (grs)
7:00 AM	**100**
11:00 AM	**50**
03:00 PM	**0**
06:30 PM	**100**
07:00 PM	**50**
11:00 PM	**0**

Non evenly spaced

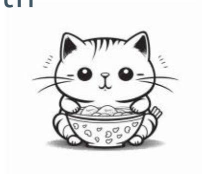
Units: h, min

Spaces:
4h | 3h 30 min **=** 240 m | 210 min

Greatest common divisor:

$$\gcd(240, 190) = \gcd(2^4\cdot 3\cdot 5, 2*3*5*7) =$$
$$= 2*3*5 = 30$$

=> The biggest space we can use to fit both space sizes is **h = 30 min**



Transform | Example | Non evenly -> evenly spaced III

Time	Available food (grs)	Time	Available food (grs)	Time	Available food (grs)	Time	Available food (grs)
07:00 AM	**100**	10:00 AM	**Unknown**	01:10 PM	**Unknown**	04:00 PM	**Unknown**
07:30 AM	**Unknown**	10:30 AM	**Unknown**	01:30 PM	**Unknown**	04:30 PM	**Unknown**
08:00 AM	**Unknown**	11:00 AM	**50**	02:00 PM	**Unknown**	05:00 PM	**Unknown**
08:30 AM	**Unknown**	11:30 AM	**Unknown**	02:30 PM	**Unknown**	05:30 PM	**Unknown**
09:00 AM	**Unknown**	00:00 AM	**Unknown**	03:00 PM	**0**	06:00 PM	**Unknown**
09:30 AM	**0**	00:30 PM	**Unknown**	03:30 PM	**Unknown**	06:30 PM	**100**

Evenly spaced 30 min

Máster Deep Learning

Transform | Example | Non evenly -> evenly spaced IV

Time	Available food (grs)	Time	Available food (grs)
07:00 PM	**Unknown**	10:30 PM	**Unknown**
07:30 PM	**Unknown**	10:30 PM	**Unknown**
08:00 PM	**Unknown**	11:00 PM	**0**
08:30 PM	**Unknown**		
09:00 PM	**100**		
09:30 PM	**50**		

Length:

Evenly spaced 30 min

Máster Deep Learning

Transform | Example | Non evenly -> evenly spaced V

Time	Available food (grs)	Time	Available food (grs)
07:00 PM	**Unknown**	10:30 PM	**Unknown**
07:30 PM	**Unknown**	10:30 PM	**Unknown**
08:00 PM	**Unknown**	11:00 PM	**0**
08:30 PM	**Unknown**		
09:00 PM	**100**		
09:30 PM	**50**		

Evenly spaced 30 min

Length:

7AM -> 11 AM => 16 h

=> 32 timestamps of 30 min

Length: 32

How do we apply this change in python?

?

Transform | Build the dataFrame IV

- df: DataFrame with time index
- Build differences
  - deltas = df.index.to_series().diff()
- Check if only one delta or different deltas
  - is_evenly_spaced = deltas.dropna().nunique() == 1
- Make it evenly spaced if it isn't

Now we got the evenly spaced...
What problem do we have?

?

Transform | Example 1 | What problem do we have?

- We have a lot of "unknown" values.
- What will pandas have done with them at creating the dataset?
- How can we control the value in those place?
- In this case, … do we know the values?
- What about the multivariate version… could we fill any of the columns with real data based on the known information?

Transform

Thus… it is the moment…

Analyze ⇄ Transform as needed

Let's learn:

- How to do an EDA (Exploratory Data Analysis)
- Transform techniques for time series

EDA

EDA

- Visualize the information within the dataset
  - Visualize the plot

```
df.plot( figsize = (10, 4))
```

  - Visualize one column

```
def plot_var(df, var):
    ax = df.plot(y = var, figsize = (10, 4))
    ax.set_xticklabels(
        ax.get_xticklabels(),
        rotation = 45
)
```

EDA | Scale |

- Are the plots easy to analyze?
  - Problems with the scale?
  - Use this option to plot at different scales

```
primary_vars = [<names vars to the left>]
secondary_vars = [<names vars to the right>]
ax = df[primary_vars + secondary_vars].plot(
 secondary_y=secondary_vars, …
)
```

Transform | EDA | Profile |

- Take a profile from the data to get a little nearer to its information.

```
! pip install ydata_profiling
from ydata_profiling import ProfileReport
profile = ydp.ProfileReport(
       #df.sample(10000),
       title="Pandas Profiling Report for <your dataset>'",
       explorative=True,
)
profile.to_notebook_iframe()
profile.to_file('report.html')
```

Máster Deep Learning

Transform | EDA | Profile II

Overview

Brought to you by YData

Overview | Alerts 6 | Reproduction

Dataset statistics

Number of variables	4
Number of observations	732
Missing cells	16
Missing cells (%)	0.5%
Duplicate rows	16
Duplicate rows (%)	2.2%
Total size in memory	28.6 KiB
Average record size in memory	40.0 B

Variable types

Numeric	4

Transform | EDA | Profile II

Overview

Brought to you by YData

Overview | Alerts 6 | Reproduction

Alerts

Dataset has 16 (2.2%) duplicate rows	Duplicates
MAGNITUD_10 is highly overall correlated with MAGNITUD_12 and 2 other fields	High correlation
MAGNITUD_12 is highly overall correlated with MAGNITUD_10 and 2 other fields	High correlation
MAGNITUD_7 is highly overall correlated with MAGNITUD_10 and 2 other fields	High correlation
MAGNITUD_8 is highly overall correlated with MAGNITUD_10 and 2 other fields	High correlation
MAGNITUD_10 has 13 (1.8%) missing values	Missing

Transform | EDA | Profile III

Overview

Brought to you by YData

Overview Alerts 6 Reproduction

Reproduction

Analysis started	2024-12-02 14:49:39.864079
Analysis finished	2024-12-02 14:49:47.674195
Duration	7.81 seconds
Software version	ydata-profiling vv4.12.0
Download configuration	config.json

Transform | EDA | Profile IV

Variables

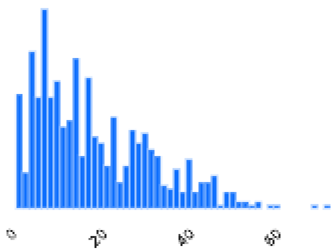
Select Columns

MAGNITUD_10

Real number (ℝ)

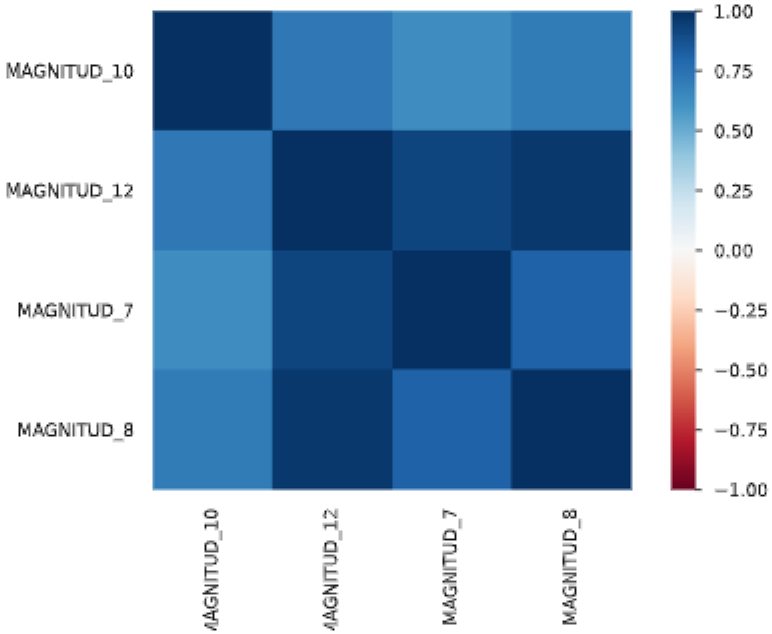
High correlation Missing

Distinct	60	Minimum	1
Distinct (%)	8.3%	Maximum	73
Missing	13	Zeros	0
Missing (%)	1.8%	Zeros (%)	0.0%
Infinite	0	Negative	0
Infinite (%)	0.0%	Negative (%)	0.0%
Mean	18.887344	Memory size	11.4 KiB



More details

Transform | EDA | Profile IV



Transform | EDA | Profile IV

Duplicate rows

Most frequently occurring

	MAGNITUD_10	MAGNITUD_12	MAGNITUD_7	MAGNITUD_8	# duplicates
4	4.0	11.0	1.0	9.0	3
5	5.0	5.0	1.0	3.0	3
0	2.0	10.0	1.0	8.0	2
1	3.0	18.0	1.0	16.0	2
2	4.0	6.0	1.0	4.0	2
3	4.0	7.0	1.0	6.0	2
6	5.0	9.0	1.0	7.0	2
7	5.0	13.0	1.0	11.0	2

Missing data - Imputation

Máster Deep Learning

Transform | Example 1 | Missing data - Imputation

- In the multivariate version, in the column "Refilled food (grs)" we know exactly the hours we filled the bowl, thus we can just use '0' at the unknown timestamps.

- In the 'Eaten food (grs)' or 'Available food (grs)' we don't know if the cat have eaten between timestamps, so, we must take a decision.

Transform | Missing data - Imputation

Definition: Imputation

Replacing missing values (MVs) with an arbitrary value to fill the empty timestamps.

The most similar to the real time series, the better we are doing.

Transform | Missing data - Imputation

- Use 'NaN' in the unknown positions.
- Constant (eg. use '0' for the unknown positions).
- Statistics: Mean/mode/median of the time series.

Máster Deep Learning

Transform | Missing data - Imputation

- Use 'NaN' in the unknown positions.
    - The same as doing nothing
- Constant (eg. use '0' for the unknown positions).
    - Usually skews the value distribution
- Statistics: Mean/mode/median of the time series.
    - Bad for high variance variables

Transform | Missing data - Imputation

- LOCF: Last Observation Carried Forward (makes sense?)
- NOCB: Next Observation Carried Backward (makes sense?)
- Rolling statistics: mean/mode/median across a *time window*

Transform | Missing data - Imputation

- LOCF: Last Observation Carried Forward (makes sense?)
  - Biases the time series if non-stationary
- NOCB: Next Observation Carried Backward (makes sense?)
  - Same as LOCF
- Rolling statistics: mean/mode/median across a *time window*
  - Adjusting window size is non-trivial, bad for large MV gaps

Transform | Missing data - Imputation

- Lineal interpolation
- Spline (polynomical) interpolation
- Additive/multiplicative/mixed decomposition

Transform | Missing data - Imputation

- Lineal interpolation
  - Assumes linearity in observations
- Spline (polynomical) interpolation
  - Assumes the variable is smooth without many perturbations
- Additive/multiplicative/mixed decomposition
  - Assumes non-relevant noise

Transform | Missing data - Imputation

- IA – based (Imputation): RNN, K-NN, Miss Forest
  - Much more computionally expensive than the alternatives
- Other methods:
  - Regression
  - Multiple imputation: create various imputated dataset and combine them in some way
  - Inverse probability weighting
  - ARIMA/SARIMA: approximate through statistics
  - Matrix completion

Transform | Missing data - Definitions

Definition: missing data

Missing value related to a timestamp in a timeseries.

Transform | Missing data - Definitions

Definition: Types of missing data

Missing

- **Completely at random (MCAR)**
  - Random positions
- **At random (MAR)**:
  - Random distribution related to other variable values
- **Not at random (MNAR)**:
  - Related to events

Transform

- LOCF: Last Observation Carried Forward (makes sense?)
  - Biases the time series if non-stationary
- NOCB: Next Observation Carried Backward (makes sense?)
  - Same as LOCF
- **Rolling statistics**: mean/mode/median across a ***time window***
  - Adjusting window size is non-trivial, bad for large MV gaps

Time window

Transform | Time window | Subsequence

Máster Deep Learning

Definition: Subsequence

If $x^m = \{x_0, \dots, x_m\}$ is a time series, a time window is a subsequence

$$x^{(k,n)} = \left\{x_{t_k}, \dots, x_{t_{k+n-1}}\right\}$$

$$k, n > 0;\ k + n - 1 < m$$

Transform | Time window | Subsequence II

Definition: Subsequence II

If x: $[1, m] \cap (\mathbb{N} \cup \{0\}) \rightarrow \mathbb{R}$ is a time series, a time window is a restriction

$$x\Big|_{[k,n]\cap(\mathbb{N}\cup\{0\})}$$

$$k, n \ > \ 0; \ k + n - 1 \ < \ m$$

Transform | Time window | Time window

Definition: Time window

A time window is a subsequence of a time series, typically smaller in relation to the length of the series, used to compute derived features or perform analyses on the series.

Commonly, these extra features allow us to better understand patterns within the time series.

Extra feature: rolling mean, mean of two variates

Feature == variate

Transform | Time window | Classification
Sliding Window |

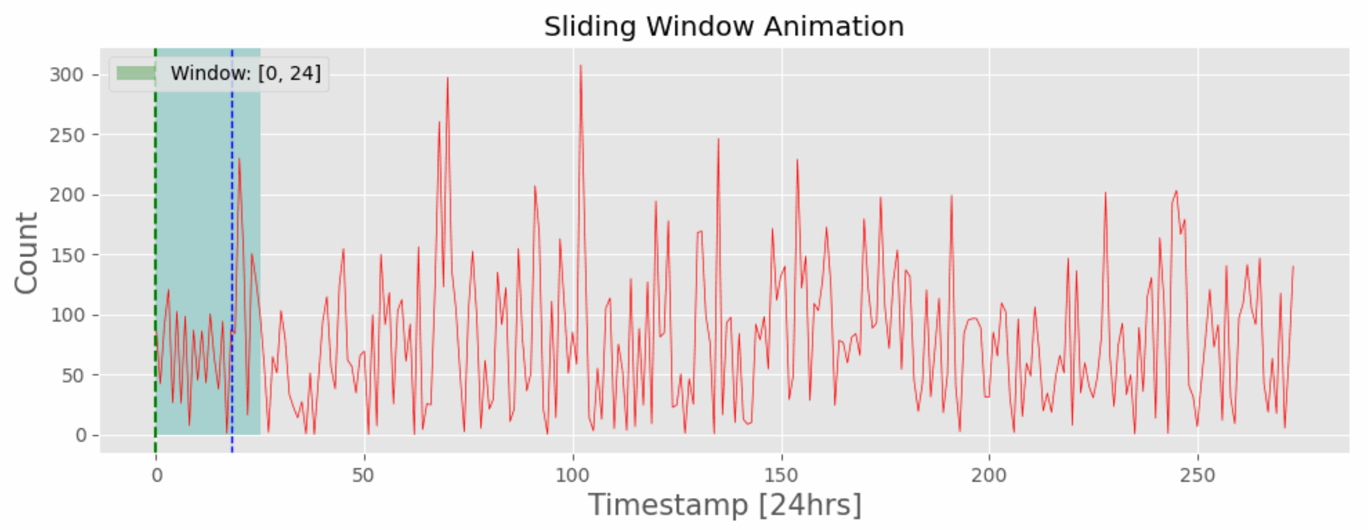
- Sliding window
- Tumbling window
- Rolling window
- Expanding window

Transform | Time window | Classification
Sliding window |

- **Sliding window**
  - $\{x^{(k+i\cdot stride,n)} \mid i \in \left[0, \frac{m-n-k}{stride}\right]\}$
  - **Fixed-size** window that moves through the time series in **steps** of an specific given '**stride**' size.
  - ARIMA, AI-based (MVP)
- Tumbling window
- Rolling window
- Expanding window

Transform | Time window | Classification
Sliding window II

- Data: [1,2,3,5,8,13, …]
- Stride: 6 | m = 25



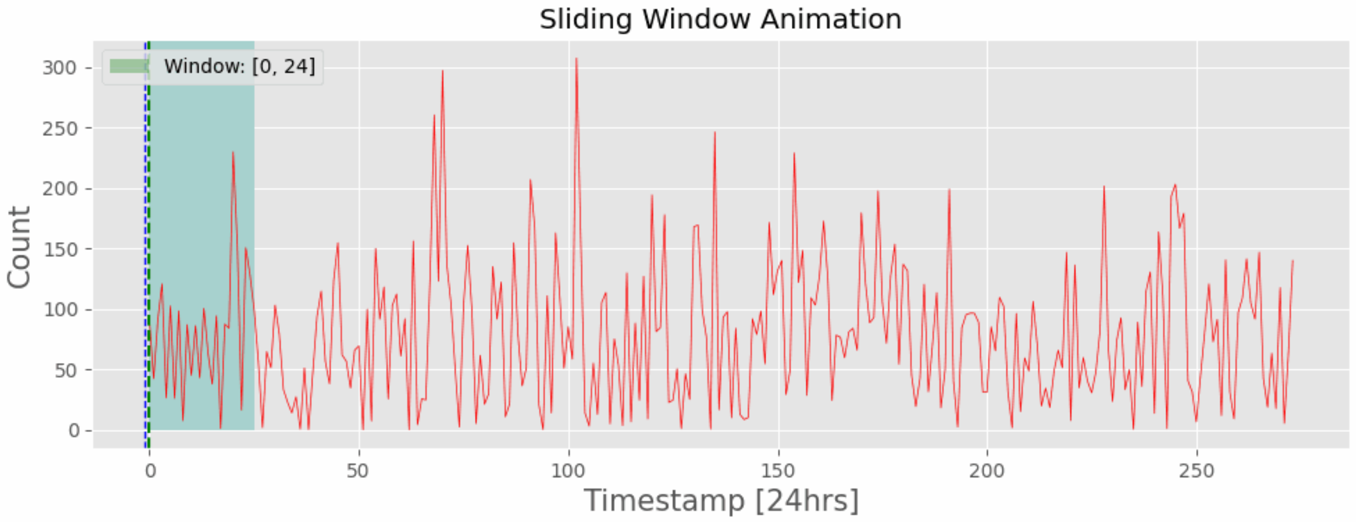
k	Timestamps (train)
0	$\{t_0, t_1, \dots, t_{24}\}$
1	$\{t_6, t_7, \dots, t_{30}\}$
2	$\{t_{12}, t_{13}, \dots, t_{36}\}$
3	$\{t_{18}, t_1, \dots, t_{42}\}$
4	$\{t_{24}, t_1, \dots, t_{48}\}$

https://stackoverflow.com/questions/76836503/how-can-reproduce-animation-for-rolling-window-over-time

Transform | Time window | Classification
Sliding window III

Máster Deep Learning

- Data: [1,2,3,5,8,13, …]
- Stride: 25 | m = 25

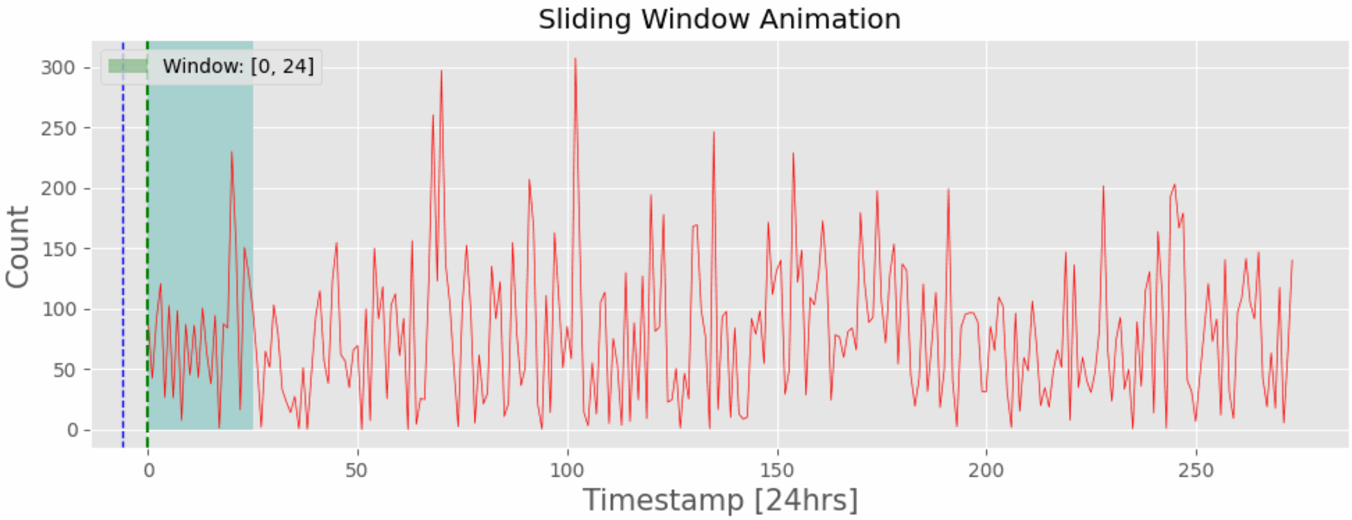


k	Timestamps (train)
0	$\{t_0, t_1, \dots, t_{24}\}$
1	$\{t_6, t_7, \dots, t_{30}\}$
2	$\{t_{12}, t_{13}, \dots, t_{36}\}$
3	$\{t_{18}, t_1, \dots, t_{42}\}$
4	$\{t_{24}, t_1, \dots, t_{48}\}$

https://stackoverflow.com/questions/76836503/how-can-reproduce-animation-for-rolling-window-over-time

Transform | Time window | Classification
Sliding window IV

- Data: [1,2,3,5,8,13, …]
- Stride: 30 | m = 25



k	Timestamps (train)
0	$\{t_0, t_1, \dots, t_{24}\}$
1	$\{t_6, t_7, \dots, t_{30}\}$
2	$\{t_{12}, t_{13}, \dots, t_{36}\}$
3	$\{t_{18}, t_1, \dots, t_{42}\}$
4	$\{t_{24}, t_1, \dots, t_{48}\}$

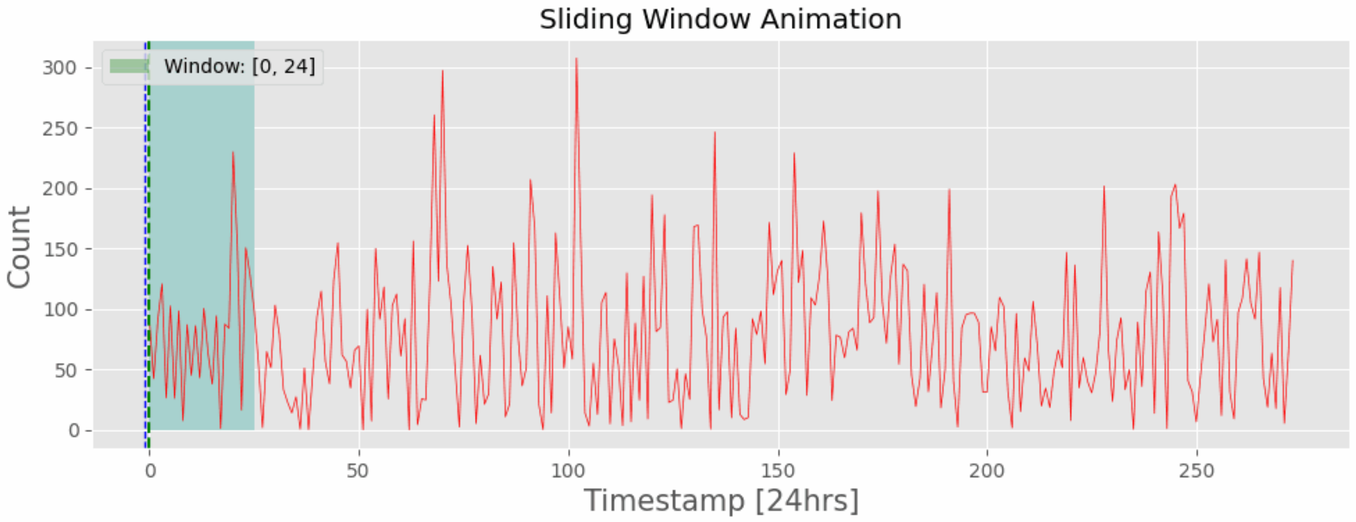
https://stackoverflow.com/questions/76836503/how-can-reproduce-animation-for-rolling-window-over-time

Transform | Time window | Classification
Sliding window VI: Tumbling window I

- **Sliding window**
- **Tumbling window**
  - $\{x^{(k+i \cdot stride, n)} \mid i \in \left[0, \frac{m-n-k}{stride}\right]\}, n = stride$
  - $\{x^{(k+i \cdot n, n)} \mid i \in \left[0, \frac{m-k}{n} - 1\right]\}$
  - **Fixed-size** window that moves through the time series in **steps** of an specific given 'stride' size.
- Rolling window
- Expanding window

Transform | Time window | Classification
Sliding window VII: Tumbling window II

- Data: [1,2,3,5,8,13, …]
- Stride: 25 | m = 25



k	Timestamps (train)
0	$\{t_0, t_1, \dots, t_{24}\}$
1	$\{t_6, t_7, \dots, t_{30}\}$
2	$\{t_{12}, t_{13}, \dots, t_{36}\}$
3	$\{t_{18}, t_1, \dots, t_{42}\}$
4	$\{t_{24}, t_1, \dots, t_{48}\}$

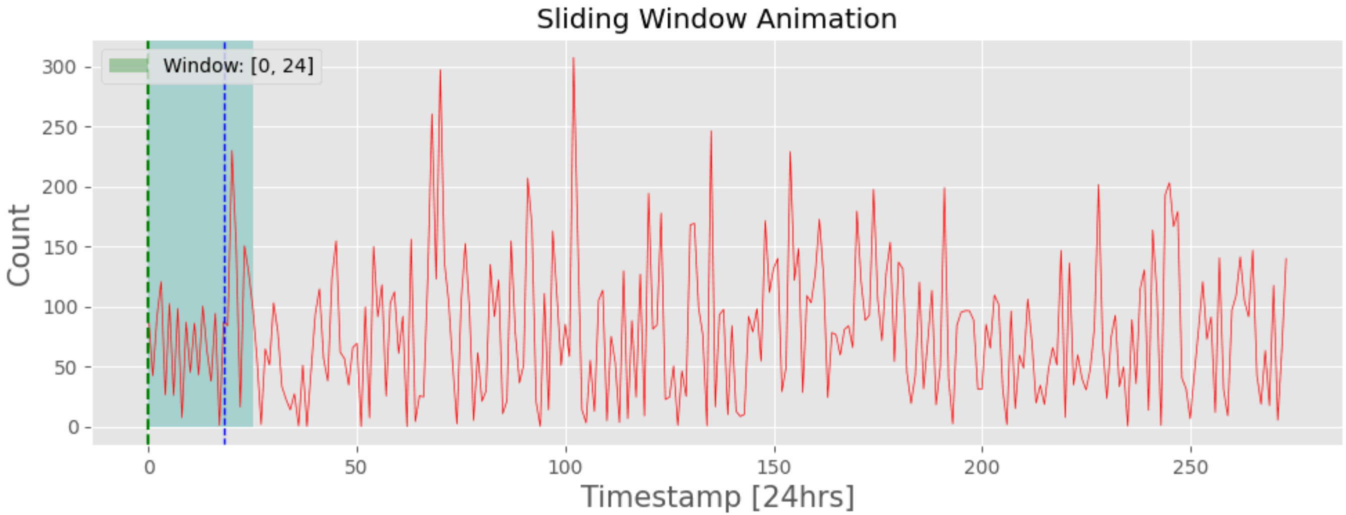
https://stackoverflow.com/questions/76836503/how-can-reproduce-animation-for-rolling-window-over-time

Transform | Time window | Classification
Sliding window VII: Rolling window I

- **Sliding window**
- Tumbling window
- **Rolling window**
  - $\{x^{(k+i \cdot stride, n)} \mid i \in \left[0, \frac{m}{stride} - n - k + 1\right]\}, stride < n$
  - Sliding window with overlap: each window includes part of the previows window
  - Rolling statistics (rolling mean), time series smoothing
- Expanding window

Transform | Time window | Classification
Sliding window VIII: Rolling window II

- Data: [1,2,3,5,8,13, …]
- Stride: 6 | m = 25



k	Timestamps (train)
0	$\{t_0, t_1, \dots, t_{24}\}$
1	$\{t_6, t_7, \dots, t_{30}\}$
2	$\{t_{12}, t_{13}, \dots, t_{36}\}$
3	$\{t_{18}, t_1, \dots, t_{42}\}$
4	$\{t_{24}, t_1, \dots, t_{48}\}$

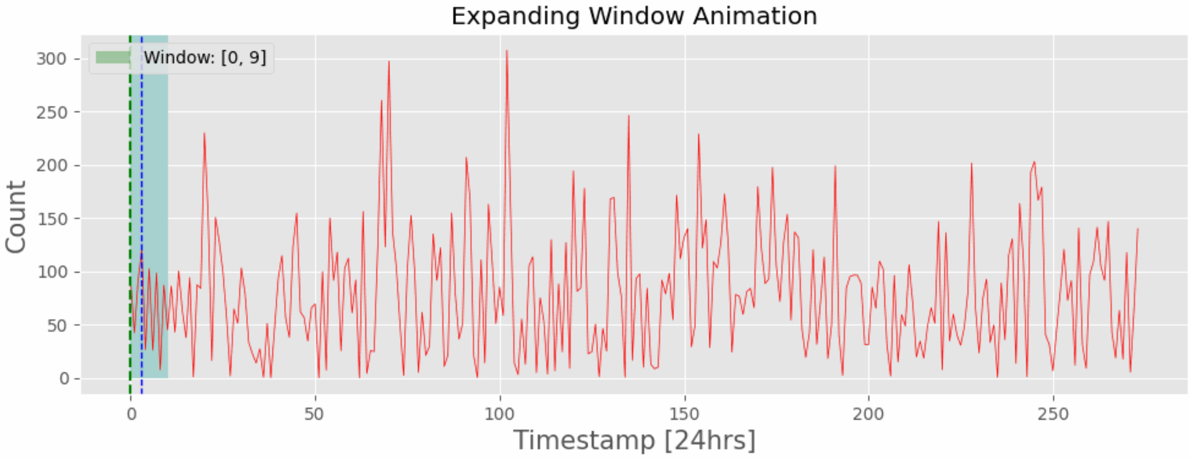
https://stackoverflow.com/questions/76836503/how-can-reproduce-animation-for-rolling-window-over-time

Transform | Time window | Classification
Expanding window |

- Sliding window
- Tumbling window
- Rolling window
- **Expanding window**
  - $\{x^{(k,n+i\cdot\Delta)} \mid i \in \left[0, \frac{m-n-k}{\Delta}\right]\}$
  - Each window expands the previous one a Δ size.
  - Financial forecasting, cumulative statistics

Transform | Time window | Classification
Expanding window II

- Data: [1,2,3,5,8,13, …]
- Delta: 6 | m = 25



k	Timestamps (train)
0	$\{t_0, t_1, \ldots, t_9\}$
1	$\{t_0, t_1, \ldots, t_{15}\}$
2	$\{t_0, t_1, \ldots, t_{21}\}$
3	$\{t_0, t_1, \ldots, t_{27}\}$
4	$\{t_0, t_1, \ldots, t_{33}\}$

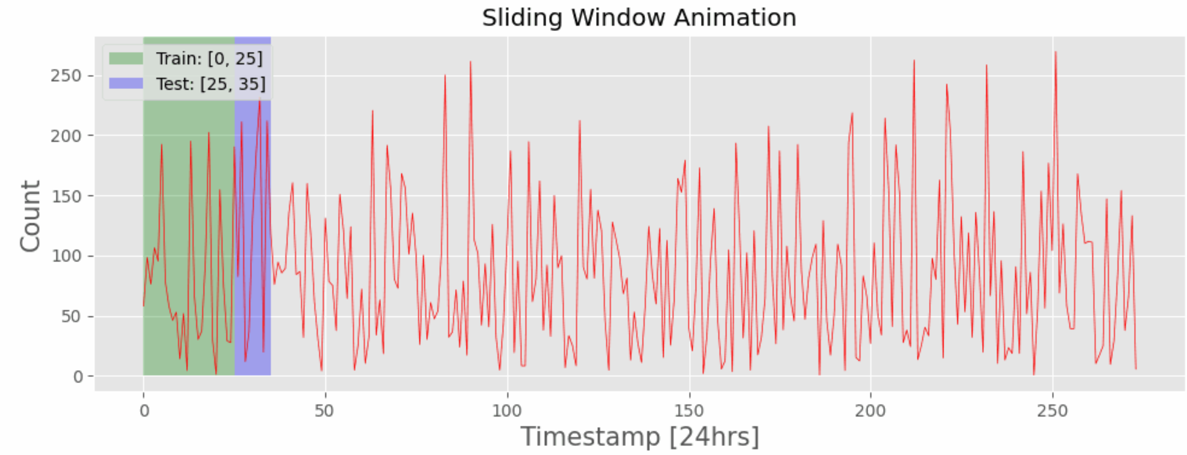
https://stackoverflow.com/questions/76836503/how-can-reproduce-animation-for-rolling-window-over-time

If we are splitting the dataset for a AI model into train + test +validation. How should the preprocessing windows look like?

?

Transform | Time windows | AI-based methods

- Data: [1,2,3,5,8,13, …]
- Stride: 6 | m_train = 25 | m_test = 10



k	Timestamps (train)	Timestamps (test)
0	$\{t_0, t_1, \ldots, t_{24}\}$	$\{t_{25}, t_{26}, \ldots, t_{34}\}$
1	$\{t_6, t_7, \ldots, t_{30}\}$	$\{t_{31}, t_{32}, \ldots, t_{40}\}$
2	$\{t_{12}, t_{13}, \ldots, t_{36}\}$	$\{t_{37}, t_{38}, \ldots, t_{46}\}$
3	$\{t_{18}, t_1, \ldots, t_{42}\}$	$\{t_{43}, t_{44}, \ldots, t_{52}\}$
4	$\{t_{24}, t_1, \ldots, t_{48}\}$	$\{t_{49}, t_{50}, \ldots, t_{58}\}$

https://stackoverflow.com/questions/76836503/how-can-reproduce-animation-for-rolling-window-over-time

Making data "similar"

Transform | Making similar

Having similar data in terms of

- Scale (range of values)
- Distribution

makes easier reducing the data with both classical an AI techniques

Transform | Making similar | Normalization

- **Normalization / Min-max scaling**
  - Scale values to an specific range $[r_{min}, r_{max}]$
  - Used for having same scale in all data.
  - Used when the features have different scales and no specif distribution is needed
    - RRNN, k-NN

Transform | Making similar | Normalization

- **Normalization**
  - $[0,1] \rightarrow x_{norm} = \frac{x - \min(x)}{\max(x) - \min(x)}$
  - $[-1,1] \rightarrow x_{norm} = 2 \cdot \frac{x - \min(x)}{\max(x) - \min(x)} - 1$
  - $[0,1] \rightarrow x_{norm} = (r_{max} - r_{min}) \cdot \frac{x - \min(x)}{\max(x) - \min(x)} + r_{min}$

Transform | Making similar | Standarization

- **Standarization**
  - Fix value to have mean = 0 and standard deviation = 1
  - Used for having same mean and deviation in all features
  - Used when data must follow a normal distribution
    - SVM – Support Vector Machine
    - PCA – Principal Component Analysis
    - Logistic regression

Máster Deep Learning

Transform | Making similar | Other

- **Robust scaling**: using percentile values
  - $x_{scaled} = \frac{x - Q1}{Q3 - Q1}$
  - Less affected by outliers
- **Logarithmic scaling**: logarithm
  - $x_{scaled} = \log(x + 1)$
  - Large differences in scale (economics)

Transform | Making similar | Other

- **Quantile normalization**: align different distribution quantiles
- Steps:
  - Order values from upper to lower
  - Compute the mean of each position of the quantile
  - Replace values by the mean
  - Bioinformatics, genetics

Máster Deep Learning

Transform | Making similar | Other scaling

A	B
5	7
8	6
12	10

pos	A
0	5
1	8
2	12

pos	B
0	6
1	7
2	10

pos	mean
0	5,5
1	7,5
2	11

A	B
5,5	7,5
7,5	5,5
11	11

Transform | Making similar | Other

- **MaxAbs scaling**: scale by absolute máximum per feature
- Scales in [-1,1]
- $x_{scaled} = \frac{x}{\max(|x|)}$
- Useful for:
  - Data with positive and negative models where wee need to preserve the sign: finances
  - Models sensitive to relative magnitudes that do no require centering (e.g. mean = 0): Lasso, ElasticNets, NN
  - Sparse data. Does not alter the structure. Sparse matrices (Word frequencies)
  - Preserve relative proportions: signal processing

Transform | Making similar | Other

- **Rolling <...>**
  - Make the operation within sliding windows
- **Expanding <...>**
  - Make the operation withing expanding windows
- **Seasonal <...>**
  - Makes the operation for specific time sizes (e.g. yearly)

Stationarity, Seasonality and Trend

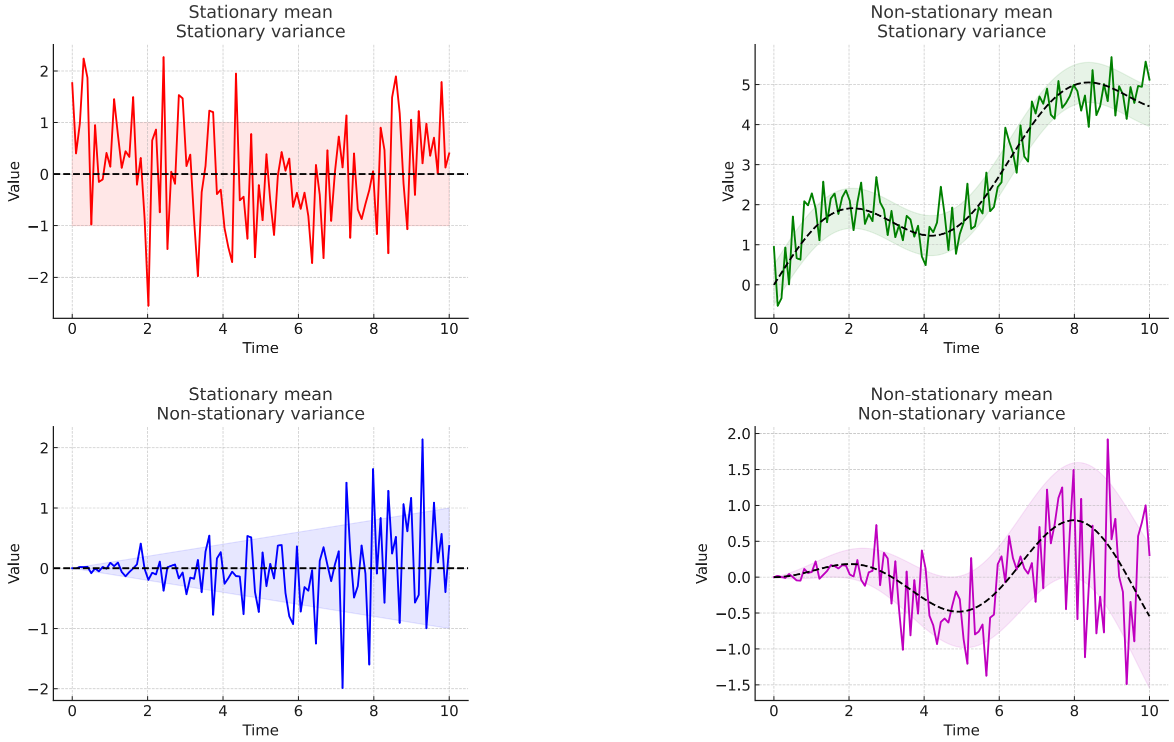
Transform | Stationarity

- **Qualitative**
  - Plot and check
- **Quantitative**
  - Get mean and standard deviation from random sliding windows
- **Statistical analysis**
  - Check through statistical analysis wether the time series is stationary or not

Transform | Stationarity

- **Qualitative**
  - Plot and check
- Quantitative
  - Get mean and standard deviation from random sliding windows
- Statistical analysis
  - Check through statistical analysis wether the time series is stationary or not

Transform | Stationarity | Qualitative



Transform | Stationarity

- Qualitative
  - Plot and check
- **Quantitative**
  - Get mean and standard deviation from random sliding windows
- Statistical analysis
  - Check through statistical analysis wether the time series is stationary or not

Transform | Stationarity | Qualitative

- **Mean stationarity**
  - Look for trends: any upward/downward tren?
  - Non-stationary data often show a trend,
  - Stationary data fluctuates around a constant mean.
- **Variance stationarity**
  - Non-stationary data may show increasing or decreasing variance over time
- **Auxiliar plots:** use the time series decomposition to get the trend plot for easier analysis (if not that clear)

Transform | Quantitative

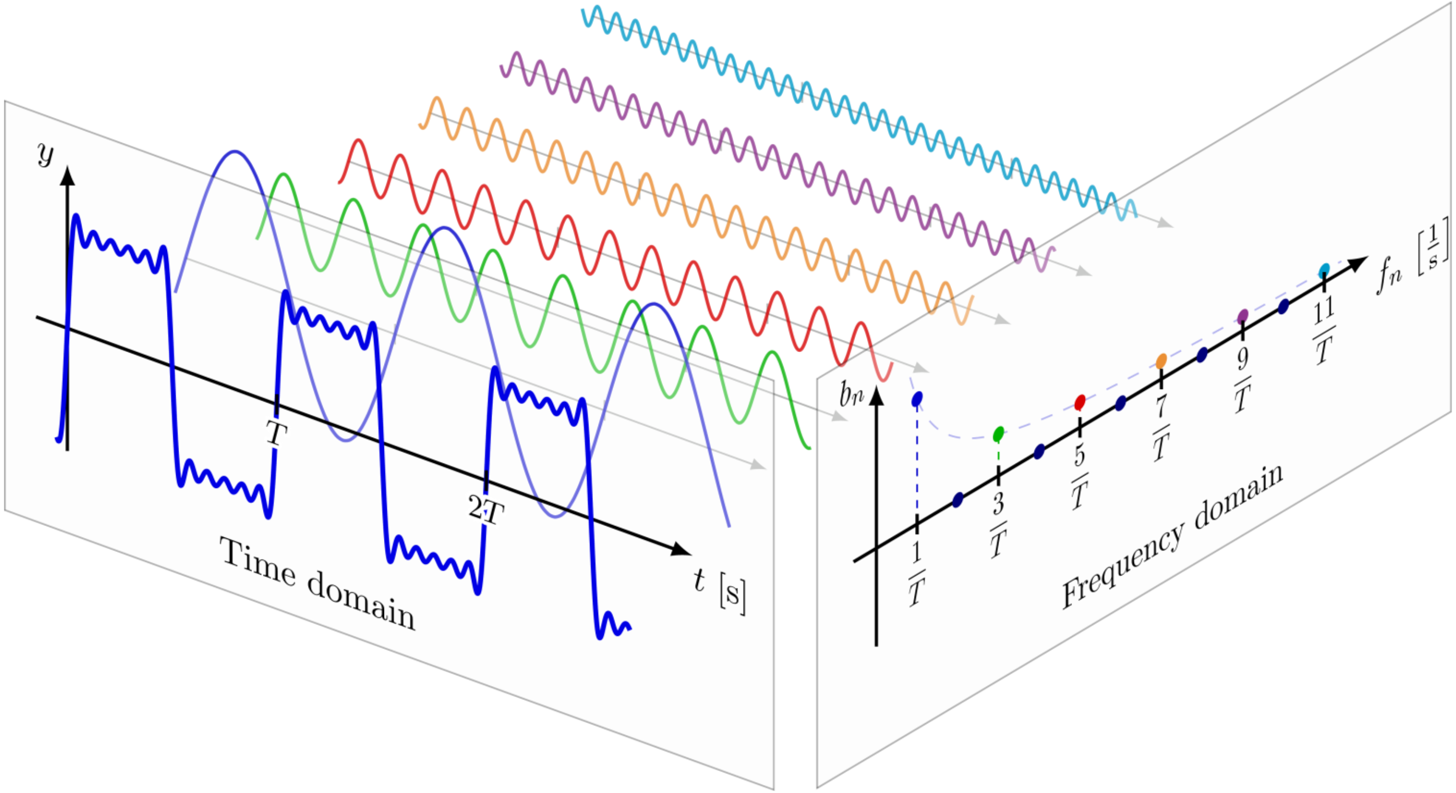
- Get mean and standard deviation from random sliding windows
  - Select representative window sizes: for example, computing Fourier predominant frequencies
  - Generate sliding window of those size in random positions
  - Compute mean and standard deviation

Máster Deep Learning

Transform | Quantitative

- If mean and deviation are ~constant between windows, the time series should be stationary
- Big changes between the values may indiquate seasonality, trends or non-variance-stationary

Transform | Quantitative | Wait... Fourier?



Transform | Quantitative | Wait… Fourier?

- One (of various) unsupervised window size selection technique.
- Fourier transform decomposes TS into sinusoid waves (Fourier coefs)
  - Represent magnitudes of associated frequencies
  - The most dominant sinusoid wave (one with largest magnitude) captures a signal's period best
- Using *CLASP.find_dominant_window_size* we can easily compute the most relevant frequencies.
- Check https://project.inria.fr/aaltd22/files/2022/09/Ermshaus_AALTD22.pdf
- https://www.youtube.com/watch?v=spUNpyF58BY
- aeon-toolkit/aeon: A toolkit for machine learning from time series

Transform | Stationarity

- Qualitative
  - Plot and check
- Quantitative
  - Get mean and standard deviation from random sliding windows
- **Statistical analysis**
  - Check through statistical analysis wether the time series is stationary or not

Transform | Statistical analysis

- **Augmented Dickey-Fuller (ADF) Test**
  - Tests for the presence of a unit root
- **Kwiatkowski-Phillips-Schmidt-Shin (KPSS) Test**
  - Tests for stationarity around a deterministic trend
- **Philips-Perron (PP) Test**
  - Similar to the ADF test but adjusts for heteroscedasticity (variance on errors) and autocorrelation (dependency on past values) in the data.

Transform | Statistical analysis

- Key values:

Test	p-value < 0,05	Null hypothesis (H_0):	Alternative hypothesis
ADF Test	Stationary	Non-stationary	Stationary
KPSS Test	Non-stationary	Stationary	Non-stationary
PP Test	Non-stationary	Non-stationary	X

Transform | Statistical analysis | Expanded table

Test	Purpose	Speed	Python module	Notes/Alternatives
ADF Test	Detect unit roots	Moderate	statsmodels	Fails for heteroscedastic data. Sensitive to lag selection.
KPSS Test	Check trend	Moderate	statsmodels	Often used alongside ADF to confirm stationary
PP Test	Adjusts for heterosedasticity	Fast	statsmodels	Similar to ADF but robust to serial correlation and heteroscedasticity
Perron Test	Multiple structural breaks	Slow	Limited external libraries	Best for datasets with complex strutural changes
Dickey-Fuller GLS (DF-GLS)	Improved ADF with Generalized Least Squares (GLS) detrending	Moderate	statsmodels	Powerfull with small sample sizes or data with trends
Hurst Exponent	Long-term memory or tren persistence	Moderate	hurst	Classifies Mean-reverting / random walk / Trending

Transform | Statistical analysis | Expanded table

Test	Purpose	Speed	Python module	Notes/Alternatives
ADF-GLS Test	Combines ADF with GLS	Moderate	statsmodels	Effective when trends are present but subtle
Wavelet Analysis	Examines stationarity at different scales	Slow	pywt (wavelet transforms)	Useful for non-stationarity signals with localized variation (e.g. seismic data)

Máster Deep Learning

Transform | Statistical analysis | Summary

Test	When to use	Strengths	Notes/Alternatives
ADF/KPSS/PP	General stationarity testing	Widely used, well documented	Sensitive to parameter tuning
Wavelet Analysis	Non-stationary signals with localized changes	Handles multi-scale analysis	Computationally intensive
Hurst exponent	Long-term detection	Hihlights persistence or mean-reversion	Less robust for short time series
ADF-GLS Test	When trends are present but subtle	More powerfull for small sample sizes	Combines ADF with GLS detrending
Perron Test	Multiple structural breaks	Handles structural changes in data	Limited external libraries, slower compared to others

Transform | Trend

- Trends and seasonality can affect model's convergence and results, it may be beneficial to remove it. How?

Transform | Trend

Method	Definition
Differencing	Substracting the observations from a previous time period
Decomposition	Breaking down the series into tren, seasonality, and residual components
Moving averages	Average over a sliding window
Polynomial fitting	Fitting a polynomial curve to the data and substract it
Trend filters	Using statistical tool to detect/remove trends
Log transformations	Applying logarithmic functions to remove exponential growth
Box-Cox transformations	Applying power transformations (mathematical transformations) to stabilize variance and make trends linear.

Máster Deep Learning

Transform | Trend

Method	Advantages	Disadvantages
Differencing	Simple, effective for removing trends	Can amplify noise; may lose interpretability of the data
Decomposition	Separate components clearly	Requires assumptions about the model (additive/multiplicative/mixed)
Moving averages	Smooths fluctuations, reduce noises	May remove important short-term patterns
Polynomial fitting	Handles non-linear trends effectively	Risk of overfitting; higher-degree polinomials can distort data
Trend filters	Effective for extracting smooth trends	Can be computionally expensive; parameters need fine-tuning
Log transformations	Stabilizes variance, simplifies multiplicative relationships	Non suitable for data with <= 0 values
Box-Cox transformations	Handles both tren and variance issues	Requires non-negative data; interpretation of transformed data is harder

Transform | Trend

Method	Recommendations
Differencing	When trends are linear and the data is heavily autocorrelated
Decomposition	Ideal for datasets with clear seasonality or multiple components
Moving averages	Best for smooting noise in non-seasonal series with light trends
Polynomial fitting	Use for series with non-linear trends; limit polynomial degree to avoid overfitting
Trend filters	Use for economic or financial data with noisy trends
Log transformations	Use for exponential growth trends (e.g. population, financial data)
Box-Cox transformations	Use for data with heteroscedasticity and exponential trends

Máster Deep Learning

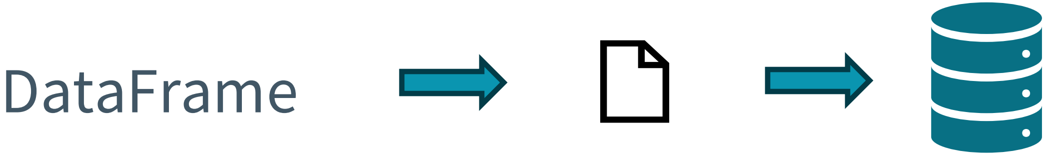
Transform | Seasonality

- Detect a seasonality period -> Remove seasonality
- Steps:
  - Plot the data: check the period of your target variable
  - Identify pattern period: hourly, daily, weekly
  - Treat seasons as ‘cyclical trends’
    - Differentiating: remove the value from a <insert detected period> before
    - Polynomial fitting: Fit a polynomial on a given season, later subtract it from each <insert detect period>.

Load

Save the data

Load



Load

- **Files**: owned folder
- **Database**: SQL, Cassandra, MongoDB, PostgretSQL, Weight and Biases
- **Specific webpages**: Kaggle, zenodo, HuggingFace
- **Own datasets**: Github, GoogleDrive, personal webpage

Miscelaneous

Auxiliar material

Máster Deep Learning

Further reading

- Research paper: Survey: Time-series data preprocessing: A survey and an empirical analysis
- Example I: IMSL
- Example II: Medium
- Example III: Kaggle
- Example IV: Advanced missing data reconstruction. UCO: Time series data mining: preprocessing, analysis, segmentation and prediction. Applications
- Example V: Linkedin schema
- Example VI: Linkedin | Dr. John Martin – AI | ML | Newsletter

Máster Deep Learning

Further reading

- Paper | Missing data: A comparison of three popular methods of handling missing data: complete-case analysis, inverse probability weighting, and multiple imputaiton
- OpenAccess paper: Recurrent Neural Networks for Multivariate Time Series with Missing Values
- Light lecture on ADF: Machine Learning Plus
- Original paper of ADF: Rizwan Mushtaq

Practice datasets

- Tourism: Tourists night stances in Tenerife
- Normalization and standarization example: Machine Learning Mastery
- Make evenly/non-evenly time series with pandas.

Google Collab

02_preprocessing.ipynb

?

Summary

Summary of the lesson

Máster Deep Learning

What this we just learned?

- What is a time series
- Where can we get time series from
- Types of time series
- The problem of evenly/non-evenly distribution
- Lenth
- Classical (additive decomposition): trend, seasonality, irregular/noise

Google Collab

02_preprocessing_exercises.ipynb

?

What is the next step?

- Start with Deep Learning!
- Time series forecasting:
  - How well can we guess the future based on the past values? Is it possible?
  - How?

To be continued...

Questions? mi.santamaria@upm.es

Deep Learning para series temporales

Part II

Preprocessing and Analysis

Máster Deep Learning

UNIVERSIDAD POLITÉCNICA DE MADRID

POLITÉCNICA

Máster Deep Learning