Introduction to Feature engineering and Time Series

Dan Ofer

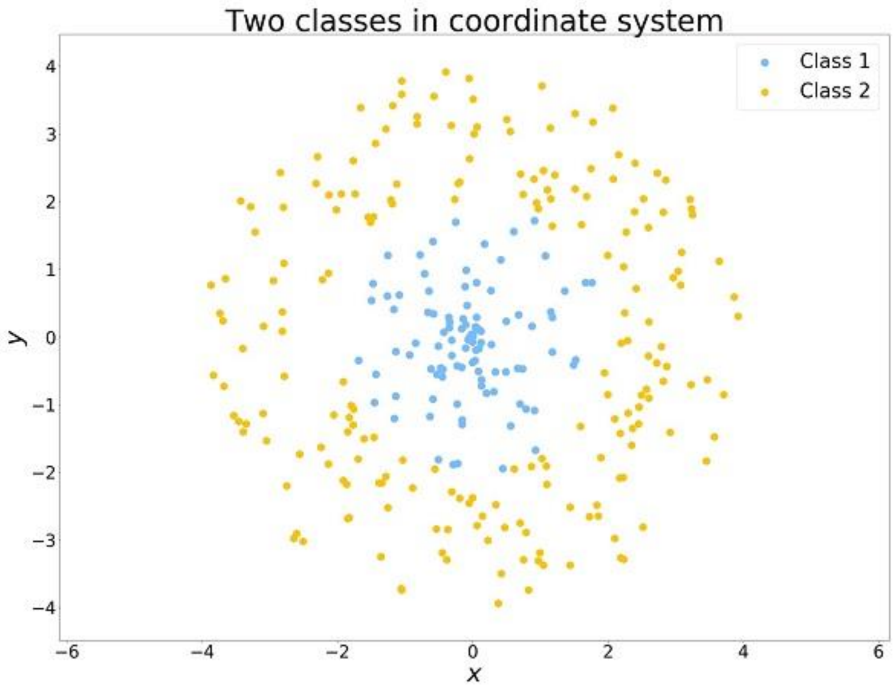
Why listen to me?

- Dan Ofer - Senior Data Scientist at Sparkbeyond 4.5Y
- Multiple ML projects with Fortune 500, HMOs, charities, including healthcare, insurance, CRM, chemicals etc'
- Top 0.8% on Kaggle (kaggle.com/danofer)

- MsC: Neuroscience & Bioinformatics (HUJI), thesis on protein feature engineering
- 1st Place in WiDS 2020
- Founder & captain of the Hebrew U. ML & DS team
- Probably took your picture at a convention/Midburn!

Feature Engineering

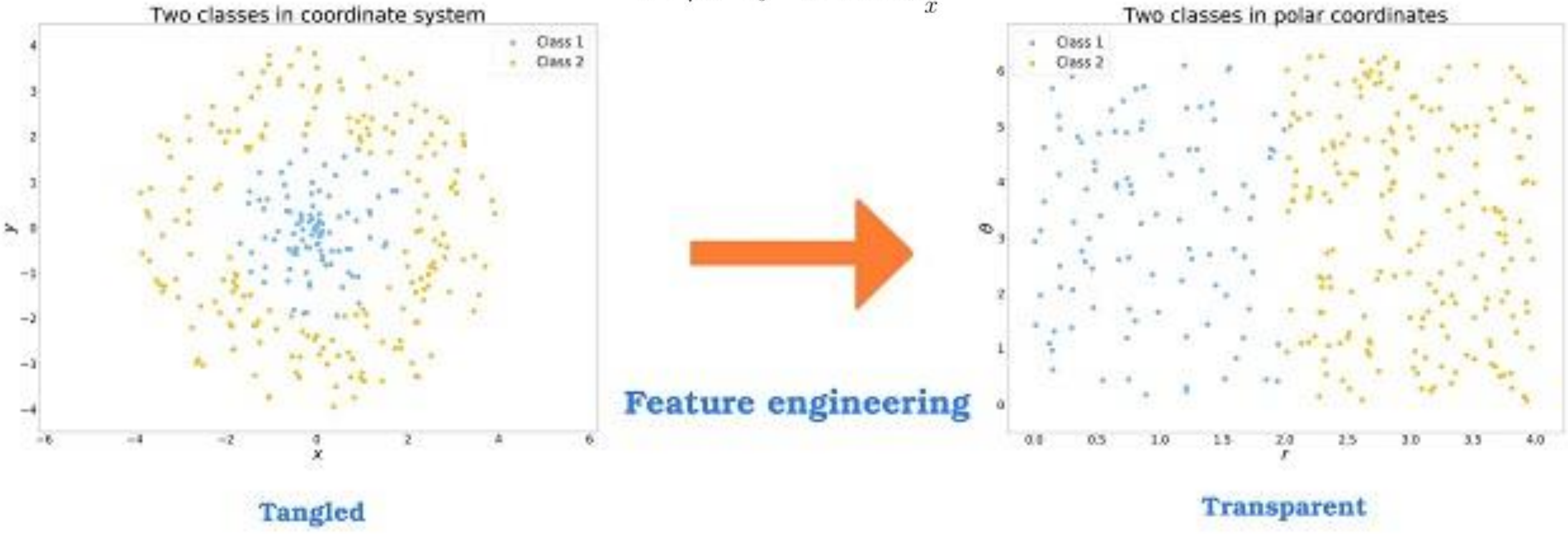
Feature Engineering: What is it good for?



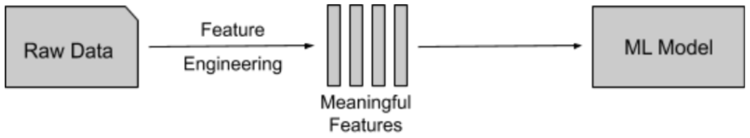
Feature Engineering: What is it good for?

Coordinate transformation

$$r = \sqrt{x^2 + y^2} \quad \theta = \arctan \frac{y}{x}$$



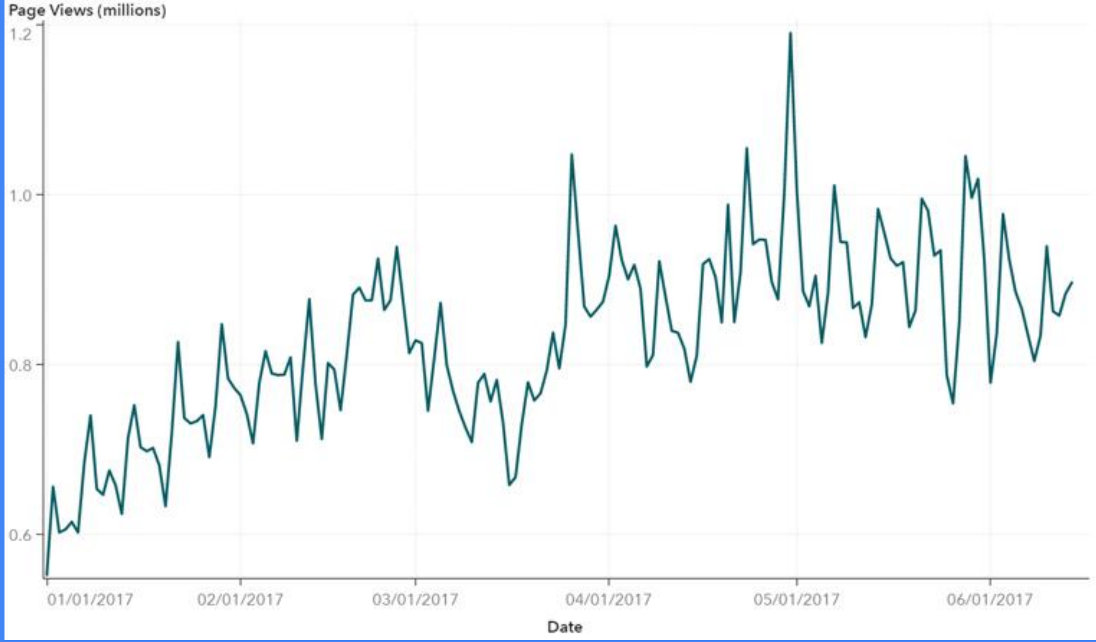
Feature Engineering



- Make *good* features
    - Highly predictive (of the target variable), succinct, interpretable, robust
- “Secret sauce of machine learning”
    - ~3d most important part of an machine learning project (after problem definition & data cleaning)
- Related, but different:
    - Feature extraction (raw features)
    - Feature selection
- Huge topic. More art than science
    - Extremely manual, handcrafted “dark magic” (For now)
- SparkBeyond (SOTA Automated Feature Engineering)
- Reading: Appendix + A few useful things to Know about machine Learning

Feature Extraction Vs Engineering

- Given a tweet, predict emotional sentiment
  - “Star Wars = Greatest. movie. Ever!! Awesome! :D “ : Positive
  - “Coronavirus quarantine/furlough makes me sad+bored” : Negative
- Feature extraction:
  - Tokenize and extract Bag of words from text. (Counts each possible word)
- (Simple) Feature engineering:
  - Lowercase text, normalize, lemmatize words (“Cats” -> “Cat”), extract word frequency instead of binary count
- Advanced Feature engineering:
  - As above + get a sentiment lexicon/dictionary (AFINN, depechemood), and use it to score text ({“awesome”:+2, “sad”:-1, “terrible”:-5...}


Page Views (millions)
1.2
1.0
0.8
0.6
01/01/2017
02/01/2017
03/01/2017
04/01/2017
05/01/2017
06/01/2017
Date

Time Series

Time Series - Data over Time

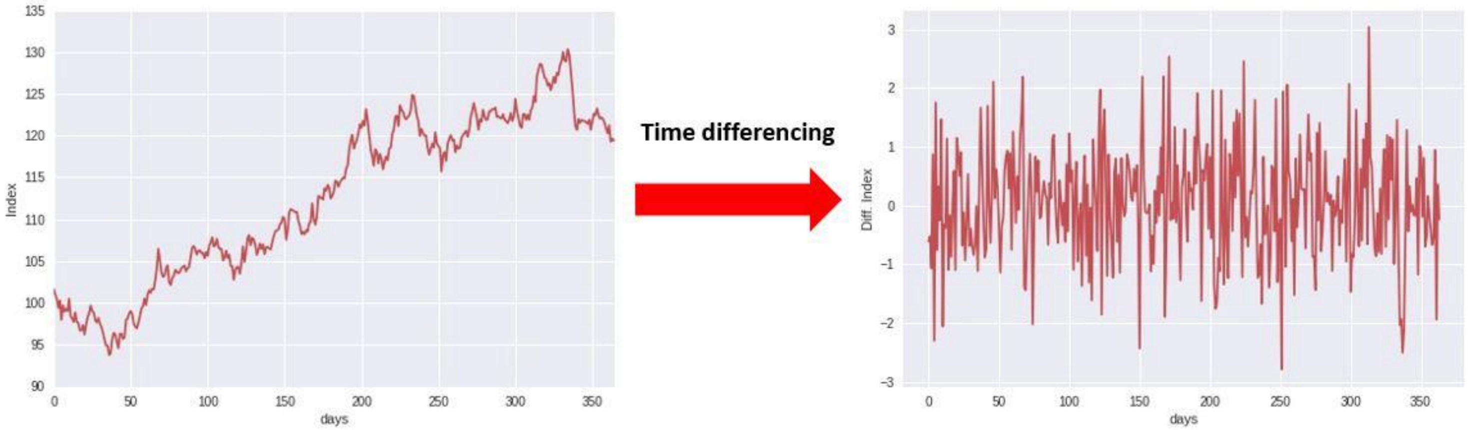
Can also be seen as:

- (X) Data with time
    - E.g. “Given each customers internet history, predict if they will click an Ad”
- (Y) Target over time
    - E.g.: “Evaluate a financial trading strategy, predict daily stock market price, per stock, for each day, for the next month”
    - How many coronavirus patients..

Time-series: Some vital statistics

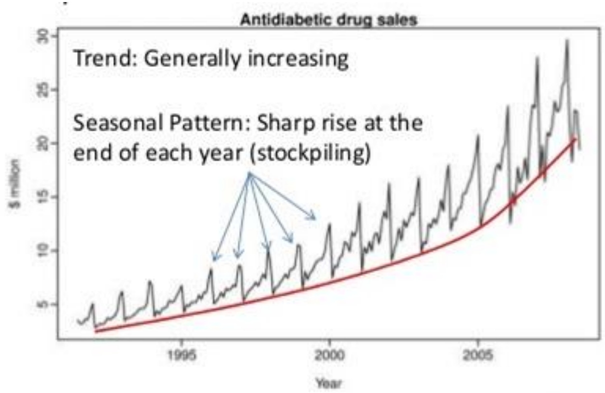
- Stationary (over time)?
  - Do statistical properties (e.g. mean, VAR) change? (e.g. inflation, growth)
  - Dickey–Fuller test (ADF)
  - If non stationary over time - detrend!

    https://people.duke.edu/~rnau/411diff.htm
    http://people.duke.edu/~rnau/whatuse.htm



Time-series: more vital statistics

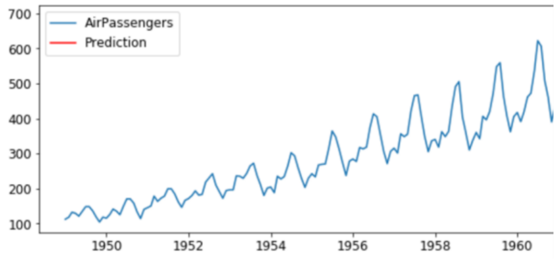
- Seasonality?
  - Seasonal components: Daily/monthly/hourly? Holiday?
- Noise?
  - Regularly sampled intervals? Future non-causal noise?
- Multiple variables or univariate?
  - Predict a stock by its history: univariate (+AutoRegression)
  - Using other stocks: multivariate
- How much history?
  - Less than a year?
- Outliers?
- [Breakpoints](), Level shifts?



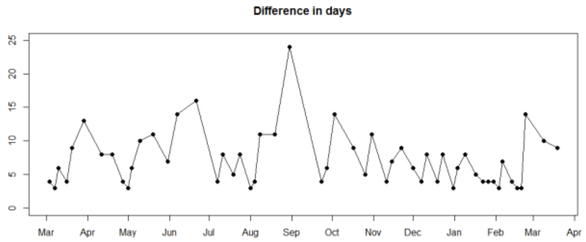
More about time-series

- Regular / non regular intervals?
    - I.e “data recorded every day, without missing values” vs “We have customer activity, on times when they visited on of our affiliates”

Regular:



Irregular :



Evaluating Forecasts: Common Gotchas

- Not splitting test-set by time
  - E.g. Stock prices. Highly dynamic, strong trend over time. Regular time series. A random sample of the data can be easy to solve by interpolating a stocks value from the dates before and after
- Target/Feature leakage:
  - Calculating features without accounting for **prediction horizon**.
  - E.g. : Aggregate mean of value, including future data points.
- Forgetting simple baselines
  - Last value, mean of value..
  - Overcomplicated models (often inferior to naive baselines!)
- Assuming everything is predictable
- Not accounting for non stationary values (Trends..)
  - Make it stationary by 1st/2d order differencing! https://people.duke.edu/~rnau/411diff.htm
  - Random forests & regression: can't predict target outside of range
    - Linear models can though! (But ARIMA, etc' don't like non-stationary..)

Simple Features

DateTime/Calendar features

```
df["Datetime"] = pd.to_datetime(df["Datetime"], infer_datetime_format=True)

df['year'] = df['Datetime'].dt.year

df['month'] = df['Datetime'].dt.month

df['week'] = df['Datetime'].dt.week

df['day'] = df['Datetime'].dt.day

df['hour'] = df['Datetime'].dt.hour

df['minute'] = df['Datetime'].dt.minute

df['dayofweek'] = df['Datetime'].dt.dayofweek
```

https://www.kaggle.com/danofer/datetime-embeddings-for-end-to-end-deep-learning

Datetime Properties

Series.dt.date	Returns numpy array of python datetime.date objects (namely, the date part of Timestamps without timezone information).
Series.dt.time	Returns numpy array of datetime.time.
Series.dt.year	The year of the datetime
Series.dt.month	The month as January=1, December=12
Series.dt.day	The days of the datetime
Series.dt.hour	The hours of the datetime
Series.dt.minute	The minutes of the datetime
Series.dt.second	The seconds of the datetime
Series.dt.microsecond	The microseconds of the datetime
Series.dt.nanosecond	The nanoseconds of the datetime
Series.dt.week	The week ordinal of the year
Series.dt.weekofyear	The week ordinal of the year
Series.dt.dayofweek	The day of the week with Monday=0, Sunday=6
Series.dt.weekday	The day of the week with Monday=0, Sunday=6
Series.dt.dayofyear	The ordinal day of the year
Series.dt.quarter	The quarter of the date
Series.dt.is_month_start	Logical indicating if first day of month (defined by frequency)
Series.dt.is_month_end	Indicator for whether the date is the last day of the month.
Series.dt.is_quarter_start	Indicator for whether the date is the first day of a quarter.
Series.dt.is_quarter_end	Indicator for whether the date is the last day of a quarter.
Series.dt.is_year_start	Indicate whether the date is the first day of a year.
Series.dt.is_year_end	Indicate whether the date is the last day of the year.
Series.dt.is_leap_year	Boolean indicator if the date belongs to a leap year.
Series.dt.daysinmonth	The number of days in the month
Series.dt.days_in_month	The number of days in the month
Series.dt.tz	
Series.dt.freq	

Source:
https://pandas.pydata.org/docs/user_guide/timeseries.html#time-date-components

Lag

- Variable's value, X "steps" ago.
- Strong baseline to beat.
  - Momentum strategy in stocks: "Stock will be the same as yesterday" (Lag 1 day)
  - Weather will be the same as it was last year (lag 365)

```
df["value_lag1"] = df["Value"].shift(1)
```

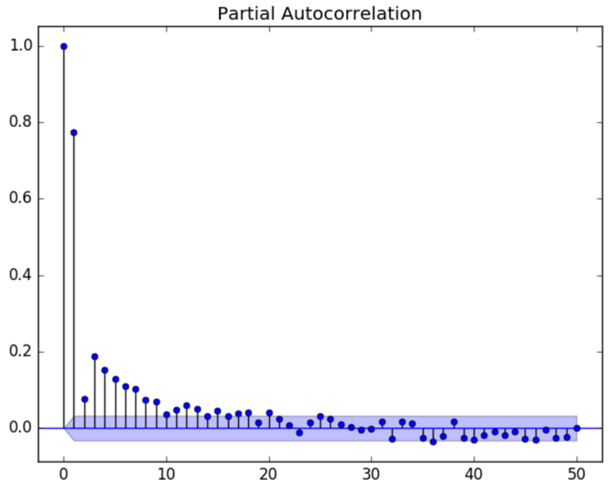
Date	Value	$Value_{t-1}$	$Value_{t-2}$
1/1/2017	200	NA	NA
1/2/2017	220	200	NA
1/3/2017	215	220	200
1/4/2017	230	215	220
1/5/2017	235	230	215
1/6/2017	225	235	230
1/7/2017	220	225	235
1/8/2017	225	220	225
1/9/2017	240	225	220
1/10/2017	245	240	225

Lag: How to pick lags?

1. Domain knowledge:
    - Store sales: same weekday last week (lag7) ; last month (lag30), last year (lag 365)
2. Partial autocorrelation plot: pick points with highest (absolute) correlation with target:

- `Pandas.plotting.autocorrelation_plot`

```
from matplotlib import pyplot
from statsmodels.graphics.tsaplots import plot_pacf
series = pd.read_csv('daily-temperature.csv')
plot_pacf(series, lags=50)
pyplot.show()
```



Sliding Window Statistics

"{Statistic} Over the last X points"

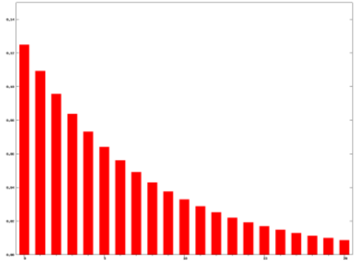
- Examples:
  - **Mean** sales over last **month**
  - **Max** sales over last **year**
  - Standard_Deviation in past 24 hours
  - **Sum**, Var, STD, skew, curtosis, etc' ...
- Can be combined with different window/weighting methods : https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.rolling.html#pandas.DataFrame.rolling

Example: Feature of Mean sales over past 30 days, with a prediction horizon of 7 days into the future, using daily data:

```
df["mean_30day_sales"] = df["sales"].shift(7).rolling(window="30D",on="Datetime").mean()
```

Sliding Window: EWMA (exponential weighted moving average)

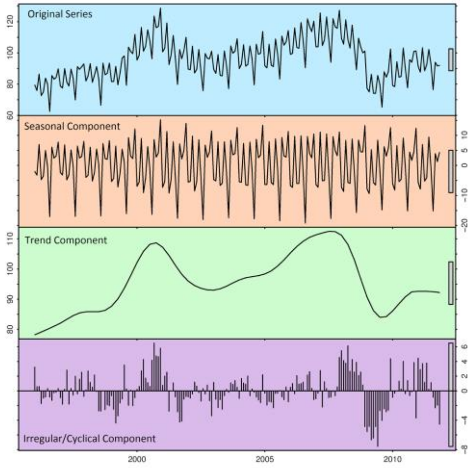
- Like sliding window, but give different weights to more recent points.
- E.g. average over last 3 years, but points 3 years ago have ¼ weight, 2 years ago ½ weight..
- https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.ewm.html



Advanced Features

Classical statistical time-series models

- Can outperform ML approaches, especially on small data, noisy problems, studied domains, highly seasonal problems, etc’
- We can use these models, or combine them with ML models!
- E.g. extract Trend, seasonal components from ARIMA/ statsmodels.seasonal decomposition/ FB prophet, and use as features!



Grouped time-series

- Features for entities/subsets within groups
- Features for entities across groups/time-series
- Store sales example:
    - Sales **per item within** the store
    - Sales of an item **across** stores

```
df[“mean_item_sales”] =
df.set_index(“datetime”).groupby([“store”,”item”])[“sales”].rolling(window=30).mean()
```

Interpolation

- Fill in missing values - required for most classical models
- This can cause a lot of bias, especially when the data is very uneven.
- I suggest only doing this when your data is from regular intervals overall, and with high temporal resolution (e.g. weekly, daily).
- Typical approaches: backfill, forward fill , fill in by average...
  - All supported in pandas

Transform temporal variables/inputs

- Differentiate/gradient, normalize by Z-score, percent change, etc’
  - E.g. Δ/rate of change in coronavirus patients (1st order diff) is more important than absolute count, in order to predict if rate (Δ) of infection is slowing or increasing

- Transform continuous variables/target to a normal distribution
  - Hit it with a log!
  - Important for many linear models!
  - Box-cox, other power transformations

Target Transformations: Change Y

- Make target stationary for modelling - e.g. subtract/divide/differentiate
    - Very important for most models!
        - Reading material in appendix
    - 1st/2d order differencing is common approach to detrending

- “Remove baseline”
    - “Subtract” mean
    - “Subtract”/divide by top feature (e.g. moving average)
    - Normalize (Z-score) by group
    - Decomposition components/forecast
      https://otexts.com/fpp2/decomposition.html

Categorical variables over time (Text)

- “Contains”.
- “Contains within last X”
- Frequency statistics (within time-window)
- Trends (e.g. breakout topics/keywords - a la Twitter)
- Word2Vec embeddings + window over time (No NN needed!)
- Order may be less important for these features, vs recency... problem dependent!
    - Workaround - looks at it as a set (ignore most ordering)
- Lots more!

Deep learning

- LSTM, GRU, RNNs..
- Convolutional neural networks (CNN)
- BiLSTM+Dilated CNN + Attention…
  - Etc'…
- Can combine with transformations, cleaning, target transformations
- In many forecasting competitions DL lost to classical TS approaches! But **combination** of both can be powerful

Thanks!

Further reading

Further reading:

- https://people.duke.edu/~rnau/411diff.htm
- Forecasting: Principles and Practice - Rob J Hyndman and G Athanasopoulos - https://otexts.com/fpp2/
- http://www.svds.com/avoiding-common-mistakes-with-time-series/
- TimeSeriesSplit - scikit-learn Time Series cross-validator
- https://machinelearningmastery.com/basic-feature-engineering-time-series-data-python/
- Kaggle: Learn Feature extraction/"engineering" https://www.kaggle.com/learn/feature-engineering

Further reading: Libraries

- Pandas - designed for tabular data and time-series! https://pandas.pydata.org/
- Automatic extraction of relevant features from time series: http://tsfresh.readthedocs.io
- Facebook Prophet - very easy to use bayesian TS -
- Irregularly spaced TS features : https://traces.readthedocs.io/en/latest/
- https://github.com/microsoft/forecasting
- Statsmodels - **Python** module that provides classes and functions for the estimation of different statistical models
- **Pmdarima = auto.arima for Python -** https://pypi.org/project/pmdarima/
- **Hyndman (R) Blog - auto.arima, Feasts etc' -** https://robjhyndman.com/hyndsight
- Feature Tools - Open-source Feature engineering library. Time-aware.
  https://www.featuretools.com/
- H20 - http://docs.h2o.ai/driverless-ai/latest-stable/docs/userguide/time-series.html
- SparkBeyond (Automated feature engineering & insights including complex multivariate, non numeric irregular time-series)