Data Science & Operational Research Team

NHS Somerset NHS Foundation Trust

Cheat Sheets

Time Series Forecasting

Methodologies

Naïve Models
Naïve models describes a series of techniques that are very simple, but surprisingly effective, especially as a benchmark for more complex techniques. For example a Naïve model could be repeating the previous value, taking a rolling average to predict the next value or using the previous similar value from the same seasonal component, i.e. predicting June 2027 based on June 2026s value.

Judgemental models
Judgemental models are a non code based form of predicting. It relies on business understanding to make well informed decisions and is typically used in the absence of any data in which to make a mathematical prediction. A good example of this is the Delphi method, where a large group of experts are asked to make a prediction, for example about whether tomorrow is going to be busy, with the majority being the prediction.

Time Series Regression
Regression models use the relationship from external variables to forecast future values of the target variable. There are many regression techniques that can be used in time series forecasting, ranging from linear regression to complex neural networks.

Time Series Decomposition
Time series decomposition involves breaking a time series down into individual components, where each component represents an individual process (this is explained in more detail on the next card).

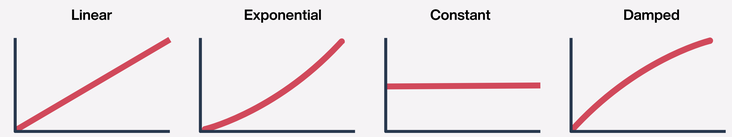
Exponential Smoothing
Exponential Smoothing methods are weighted averages of past observations, with exponentially weights decaying weights as the observations get older (i.e. the oldest values in the training data have the smallest impact on a new prediction). The more recent the observation the higher the associated weight on the prediction. An example of a Exponential Smoothing model is Holts-Winter.

ARIMA
ARIMA models forecasts a time series based on its past values. It uses both previous values (Auto Regression) and previous forecast errors (Moving Average) to estimate future values. ARIMA can model seasonal patterns (SARIMA) and incorporate external variables (SARIMAX).

Components of a Time Series

Trend
The trend forms the backbone of a forecast. It shows the general upward or downward movement of the whole or part of a time series. Depending on the model you are using the trend may be constant, moving or utilise changepoints (the trend changes at defined points)

Some of the most common trends you will see are

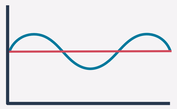


Seasonality
Seasonality represents regular patterns that deviate from the trend in line with fixed periods of time and calendar components. For example day of week may impact a businesses operating time, or a business may work on a fixed five week rota.

Examples of Seasonal components can include (but are not limited to): Hour of Day, Day of Week, Week of Year, Month, Day of Year

How this is applied to the trend will typically be in two ways, additively or multiplicatively.

Additive
In an additive model, the seasonality components are added to the trend.



`yhat = trend + season + cyclical + noise`

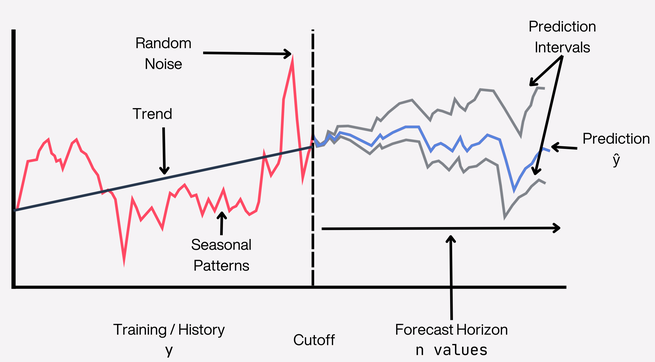
Multiplicative
In an multiplicative model the seasonal components are multiplied by the trend.

`yhat = trend * season * cylical * noise`

Cyclical patterns
A cyclical component is similar to seasonality but it does not happen for a fixed interval or line up with an aspect of a calendar. For example, the flu season has a cyclical pattern, but the size of its impact, when it occurs and for how long differ year on year.

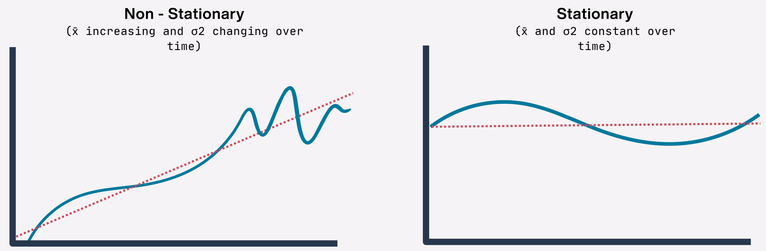
Noise
A cyclical component is similar to seasonality but it does not happen for a fixed interval or line up with an aspect of a calendar. For example, the flu season has a cyclical pattern, but the size of its impact, when it occurs and for how long differ year on year.

Anatomy of a forecast



Stationary vs Non-Stationary data

If a time series is stationary then its average (mean) and spread (variance) don't change over time. For a wide range of time series forecasting techniques, stationarity is vital for accurate predictions. Unfortunately, most real-world data isn't stationary, but there are techniques like differencing that can be used to make it stationary.



Additional Regressors

If external variables (exogenous) impact the value you are trying to forecast, you may want to incorporate them within your model. You can achieve this by adding the variable as a regressor to the model. Many time series models will accept external variables, including:

- fbProphet
- SARIMAX
- Most regression models

Typically, a linear relationship between the independent and dependant variables is modelled. If categorical variables are used, they will need to be converted to binary features (dummy variables (1,0)). In pandas, the get_dummies() method will help with this.

Auto Correlation

Autocorrelation is the relationship between a point in a time series, and the data immediately preceding it (lagged). Positive autocorrelation implies that (in daily data) that the value of the previous days influences todays prediction. Negative autocorrelation, in the same scenario would mean todays prediction would be the opposite of the previous days. No Autocorrelation indicates that the previous points have no influence over the preceding predictions.

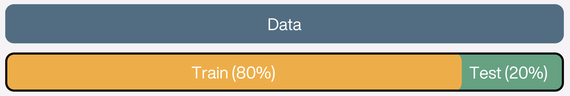
There are two commonly used plots to show autocorrelation, ACF and PACF.

ACF (Autocorrelation Function)
ACF is a (complete) auto-correlation function which gives us values of auto-correlation of any series with its lagged values.

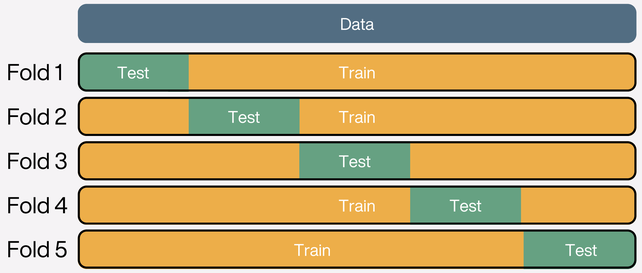
PACF (Partial Autocorrelation Function)
Instead of finding correlations of the present value with lags like ACF, it finds the correlation of the residuals (the remains after removing the effects of the already explained earlier lags).

Cross Validation

Train test split
A train test split is used to split the input data into a training set(to model the relationships) and test set (to evaluate performance). Typically, a 80:20 split (80% for training & 20% for testing) is used. **In forecasting, we have to respect the temporal dependence, so we cannot shuffle the data.**



This concept can be expanded upon by conducting multiple train test splits. Cross Validation allows for a more robust estimate for the performance of a model (compared with a single train test split).



Validation Metrics
Many different metrics can be used to evaluate model performance on the test data, including:

- Mean Absolute Error (MAE)
- Mean Error (ME)
- Mean Absolute Percentage Error (MAPE)
- Mean Absolute Scaled Error (MASE)
- Mean Percentage Error (MPE)
- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)
- Coverage

Residuals

The residuals of a model represent the error during fitting (i.e., what the model missed). In mathematical notation, this is given by $y - \hat{y}$

A model which accurately models the data will result in residuals with:
- Mean equal to zero
- No autocorrelation (this can be verified using an ACF plot)

Furthermore, prediction intervals will be easier to develop if residuals exhibit:
- Constant variance
- A normal distribution

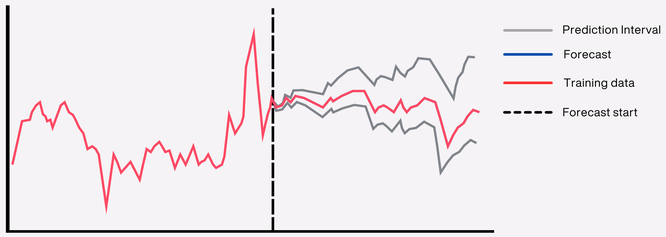
Prediction Intervals

Prediction intervals gives an estimate of an interval in which a forecasted value is likely to fall within, with specified probability (usually 95%).

Prediction intervals show the level of certainty in a forecast estimate. A point forecast (i.e., a single forecast value) will not include any information on how confident we are in the forecast. Therefore, without prediction intervals, point forecasts offer little value.

Some time series models are able to automatically produce prediction intervals, including:
- Facebook Prophet
- ARIMA
- Exponential smoothing

Otherwise, prediction intervals will need to be estimated from the model residuals. Many techniques can be used to estimate prediction intervals, including bootstrapping.



Simple Forecast Example

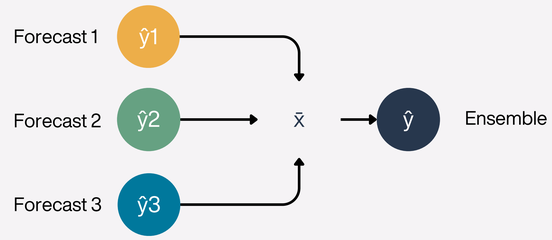
Here is a simple code block on using the Prophet library in Python. It uses a decomposable time series which allows for it to be highly flexible to a wide range of use cases.

```
import pandas as pd
from prophet import Prophet

df = pd.read_excel('excelfile.xlsx') # file containing a ds column and a y column
holidays = pd.read_excel('holidayfile.xlsx')
regressor = pd.read_excel('regressor.xlsx')

m = Prophet(holidays=holidays,
    seasonality_mode='additive',
    changepoint_prior_scale=0.5,
    growth='linear',
    yearly_seasonality='auto',
    weekly_seasonality='auto',
    daily_seasonality='auto',
    growth_cap=None
    )

m.add_regressor(regressor) # Add regressors to the initialised model

m.fit(df) # Fit data to model

future = m.make_future_dataframe(periods=30) # Make a new dataframe with 30 new data points

forecast = m.predict(future) # Make the prediction based on the size in the previous line

fig = m.plot(forecast) # Create a plot with the original data and forecast

fig.show() # Make Python render the forecast image
```

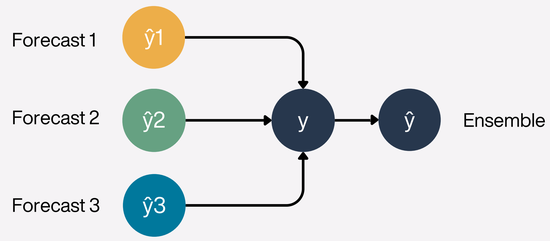
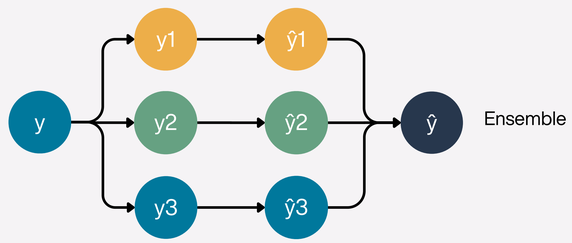
Ensembling

Ensembling is a technique that combines the predictions of multiple models to improve the accuracy of our model. The basic idea behind ensembling is that by aggregating the forecasts of multiple models, you can often achieve better results than any individual model can provide. Commonly used methods in time series forecasting are

Averaging / Weighted Averages
The simplest method is to average all the results together to make the final forecast, alternatively, when we know that one of the models has a more desirable impact then the others we may increase the weight so that it has more influence over the final result.



Stacking
Stacking is a technique where multiple forecast predictions are used as the training data for a new meta-model, which then makes the final forecast. Stacking can be adapted to weigh the individual model's predictions, giving more importance to better-performing models.



Bootstrap Aggregating (Bagging)
Bagging is a technique where you create multiple subsets of the time series data through a technique called bootstrapping. Bootstrapping is a resampling technique where you randomly sample the time series data (y) with replacement to create new datasets of the same size as the original data. These new datasets (y1, y2, y3) are referred to as "bootstrap samples." From these subsets we can then train individual models (ŷ1, ŷ2,ŷ 3) and combine them together to make our ensemble.