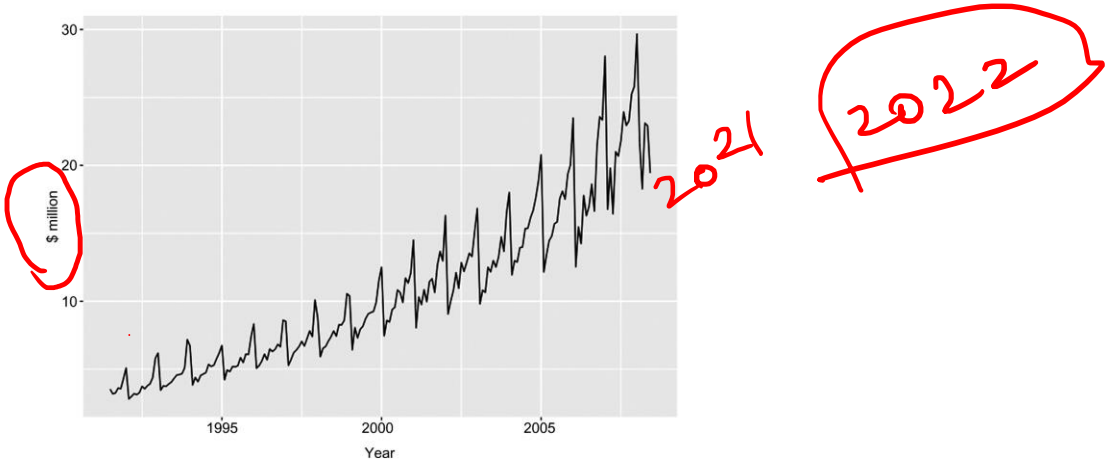
Time Series Forecasting, Anomaly Detection & Feature Selection

Dr. M. Brindha
Assistant Professor
Department of CSE
NIT, Trichy-15

Introduction

- **Time series** is a series of observations listed in the order of time.
- The data points in a time series are usually recorded at constant successive time intervals.
- **Time series analysis** is the process of extracting meaningful non-trivial information and patterns from time series.
- **Time series forecasting** is the process of predicting the future value of time series data based on past observations and other inputs.
- Time series forecasting is one of the oldest known predictive analytics techniques.
- It is widely used in every organizational setting and has deep statistical foundations.
- Drug sales:

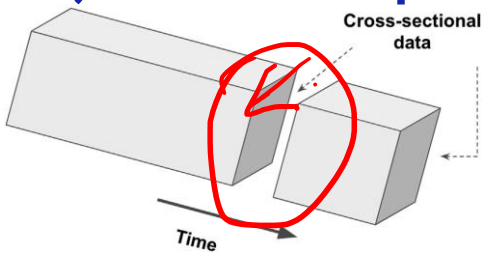


Introduction

- Supervised model building has been about collecting data from several different attributes of a system and using these attributes to fit a function to predict a desired quantity or target variable.
- For example, if the system was a housing market, the attributes may have been the price of a house, location, square footage, number of rooms, number of floors, age of the house, and so on.
- A multiple linear regression model or a neural network model could be built to predict the house price (target) variable given the independent (predictor) variables.
- Similarly, purchasing managers may use data from several different raw material commodity prices to model the cost of a product.
- The common thread among these predictive models is that predictors or independent variables that potentially influence a target (house price or product cost) are used to predict that target variable.
- The objective in time series forecasting is slightly different: to use historical information about a particular quantity to make forecasts about the value of the same quantity in the future.

Time series analysis Vs Supervised predictive models

- First, time is an important predictor in many applications.
- In time series analysis one is concerned with forecasting a specific variable, given that it is known how this variable has changed over time in the past.
- In all other predictive models, the time component of the data was either ignored or was not available. Such data are known as cross-sectional data.
- Consider the problem of predicting the house price based on location, square footage, number of rooms, number of floors, and age of the house.
- The predictors are observed in a point in time (like a cross-section of a block of wood in Fig. 12.2) and the price is predicted in the same point in time.

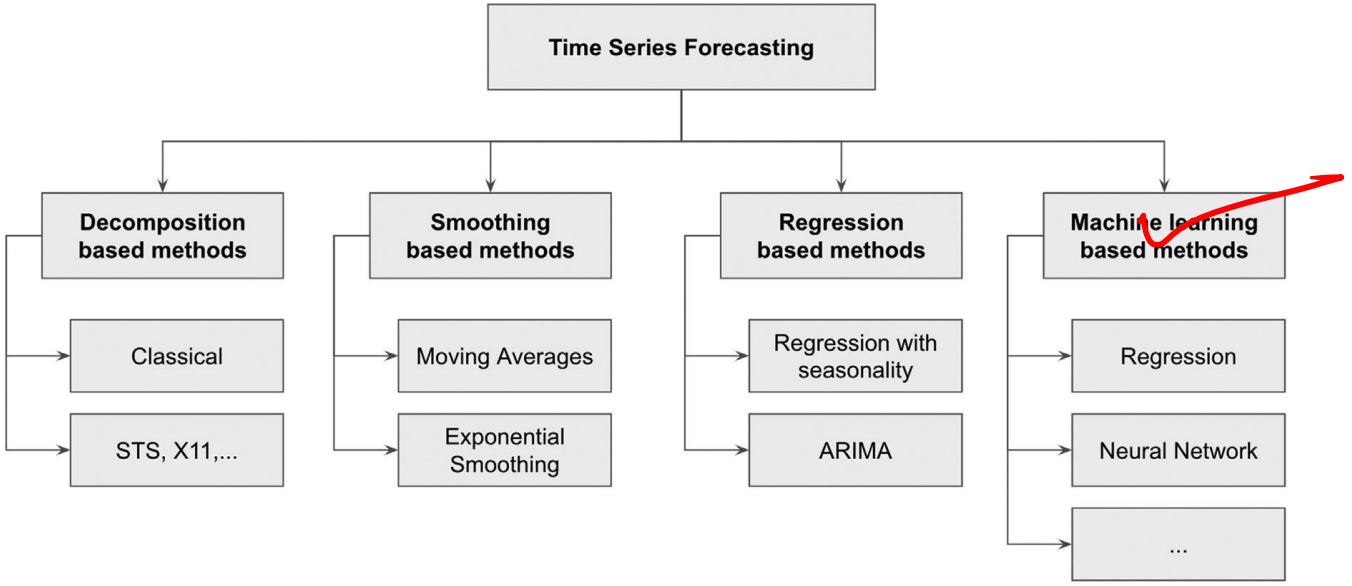


Time series analysis Vs Supervised predictive models

- However, it is important to take the time variable (length of wood) to predict something like house prices.
- It fluctuates up and down based on economic conditions, supply and demand, etc., which are all influenced by time.
- Second, one may not be interested in or might not even have data for other attributes that could potentially influence the target variable.
- In other words, independent or predictor variables are not strictly necessary for univariate time series forecasting but are strongly recommended for multivariate time series.

Taxonomy of Time series forecasting

- The investigation of time series can also be broadly divided into descriptive modeling, called time series analysis, and predictive modeling, called time series forecasting.
- Time Series forecasting can be further classified into four broad categories of techniques: Forecasting based on time series decomposition, smoothing based techniques, regression based techniques, and machine learning-based techniques



Taxonomy of Time series forecasting

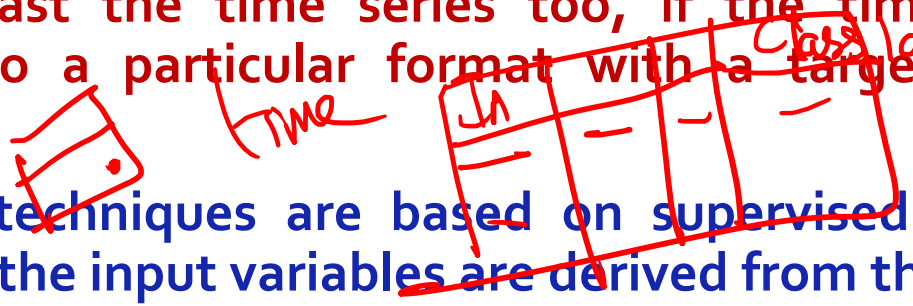
- Time series decomposition is the process of deconstructing a time series into the number of constituent components with each representing an underlying phenomenon.
- Decomposition splits the time series into a trend component, a seasonal component, and a noise component.
- The trend and seasonality components are predictable (and are called systematic components), whereas, the noise, by definition, is random (and is called the non-systematic component).
- Before forecasting the time series, it is important to understand and describe the components that make the time series.
- These individual components can be better forecasted using regression or similar techniques and combined together as an aggregated forecasted time series. This technique is called forecasting with decomposition.

Taxonomy of Time series forecasting

- Time series can be thought as past observations informing future predictions.
- To forecast future data, one can smooth past observations and project it to the future. Such time series forecasting methods are called smoothing based forecasting methods. In smoothing methods, the future value of the time series is the weighted average of past observations.
- Regression based forecasting techniques are similar to conventional supervised predictive models, which have independent and dependent variables, but with a twist: the independent variable is now time.
- The simplest of such methods is of course a linear regression model of the form: $y_t = a \times t + b$ where yt is the value of the target variable at time t.
- Given a training set, the values of coefficients a and b can be estimated to forecast future y values.
- Regression based techniques can get pretty complicated with the type of function used to model the relationship between future value and time.
- Commonly used functions are exponential, polynomial, and power law functions.

Taxonomy of Time series forecasting

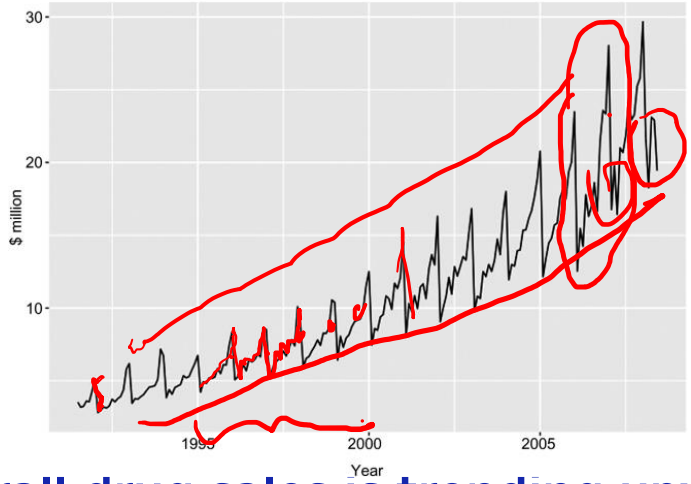
- Autocorrelation refers to the fact that data from adjacent time periods are correlated in a time series.
- The most well-known among these techniques is ARIMA, which stands for Auto Regressive Integrated Moving Average.
- Any supervised classification or regression predictive models can be used to forecast the time series too, if the time series data are transformed to a particular format with a target label and input variables.



- This class of techniques are based on supervised machine learning models where the input variables are derived from the time series using a windowing technique.
- The windowing technique transforms a time series to a crosssectional like dataset where the input variables are lagged data points for an observation.
- The artificial neural network-based time series forecasting has particular relevance because of its resemblance with the ARIMA technique.

Time series decomposition

- Time series data, are univariate, an amalgamation of multiple underlying phenomenon.
- Consider the example of monthly antidiabetic drug sales. A few observations can be made about this time series.



- Firstly, the overall drug sales is trending upward and the upward trend accelerates in the 2000s.
- Secondly, there is clear seasonality in the time series of drug sales. In particular, it is a yearly seasonality.

Time series decomposition

- There is a spike in drug sales at the start of the year and a dip in every February. This seasonal variation is consistent every year.
- However, even when accounting for the trend and the seasonal variations there is one more phenomenon that could not be explained.
- For example, the pattern in 2007 is odd when compared with prior years or 2008.
- This unattributable phenomenon can be assumed as noise in the time series.

Time series decomposition-Components

- **Trend: Trend is the long-term tendency of the data.**
- **It represents change from one period to next.**
- **If a trend line has been inserted in over a chart in a spreadsheet, it is the regression equation line that represents the trend component of a time series.**
- **The trends can be further split into a zero-basis trend and the level in the data.**
- **The level (average value) in the time series does not change with respect to time while the zero-basis trend does change with the time.**

Time series decomposition-Components

- **Seasonality:** Seasonality is the repetitive behavior during a cycle of time.
- These are repeated patterns appearing over and over again in the time series.
- Seasonality can be further split into hourly, daily, weekly, monthly, quarterly, and yearly seasonality.
- Consider the revenue generated by an online finance portal, like Yahoo Finance or Morningstar. The visits would clearly indicate a daily seasonality with the difference in the number of people accessing the portal or the app during the day time, especially during the market open hours, and the nighttime.
- Weekdays will have higher traffic (and revenue) than during the weekends. Moreover, the online advertising spend shows quarterly seasonality as the advertisers adjust marketing spend during the end of the quarter.
- The revenue will also show yearly seasonality where end of the year, after Christmas, shows weakness in revenue because of holidays.

Time series decomposition-Components

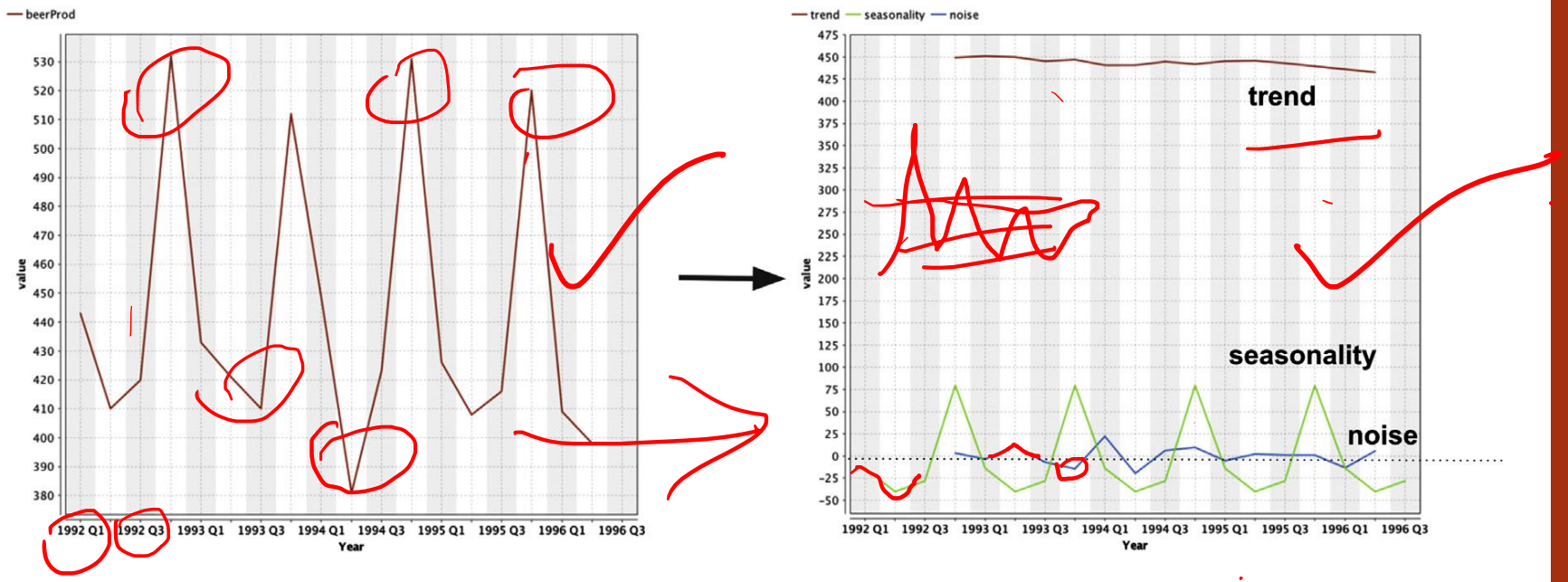
- **Cycle:** Cyclic component represents longer-than-a-year patterns where there is no specific time frames between the cycles.
- An example here is the economic cycle of booms and crashes.
- While the booms and crashes exhibit a repeated pattern, the length of a boom period, the length of a recession, the time between subsequent booms and crashes (and even two consecutive crashes— double dip) is uncertain and random, unlike the seasonality components.
- Figure, on the antidiabetic drug sales, shows an accelerated upward trend after the year 2000 which may represent a cyclic component or a non-linear trend.
- It is hard to conclude anything in this time series because of the limited time frame in the dataset.

Time series decomposition-Components

- **Noise:** In a time series, anything that is not represented by level, trend, seasonality, or cyclic component is the noise in the time series.
- The noise component is unpredictable but follows normal distribution in ideal cases.
- All the time series datasets will have noise.

Time series decomposition-Components

- Figure shows the decomposition of quarterly production data, into trend, seasonality, and noise components.



- The trend and seasonality are the systematic components of a time series.
- Systematic components can be forecasted.
- It is impossible to forecast noise—the non-systematic component of a time series.

Time series decomposition-Components

- In the time series decomposition can be classified into additive decomposition and multiplicative decomposition, based on the nature of the different components and how they are composed.
- In an additive decomposition, the components are decomposed in such a way that when they are added together, the original time series can be obtained.
- Time series=Trend+Seasonality+Noise
- In the case of multiplicative decomposition the components are decomposed in the such a way that when they are multiplied together, the original time series can be derived back.
- Time series=TrendXSeasonalityXNoise
- Both additive and multiplicative time series decompostions can be represented by these equations:

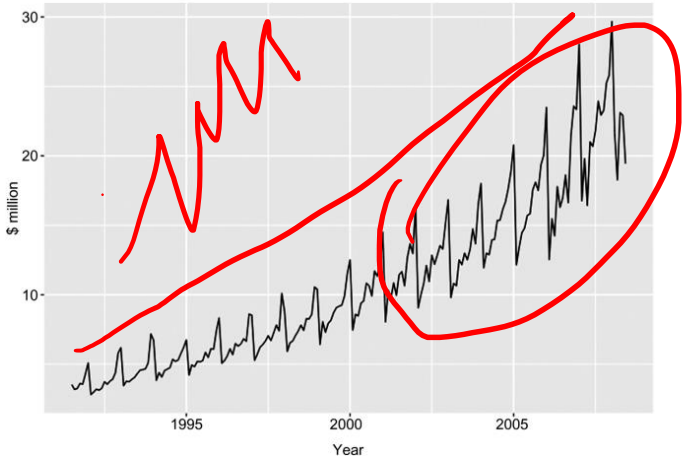
$$y_t = T_t + S_t + E_t$$

$$y_t = T_t \times S_t \times E_t$$

where Tt, St, and Et are trend, seasonal, and error components respectively.

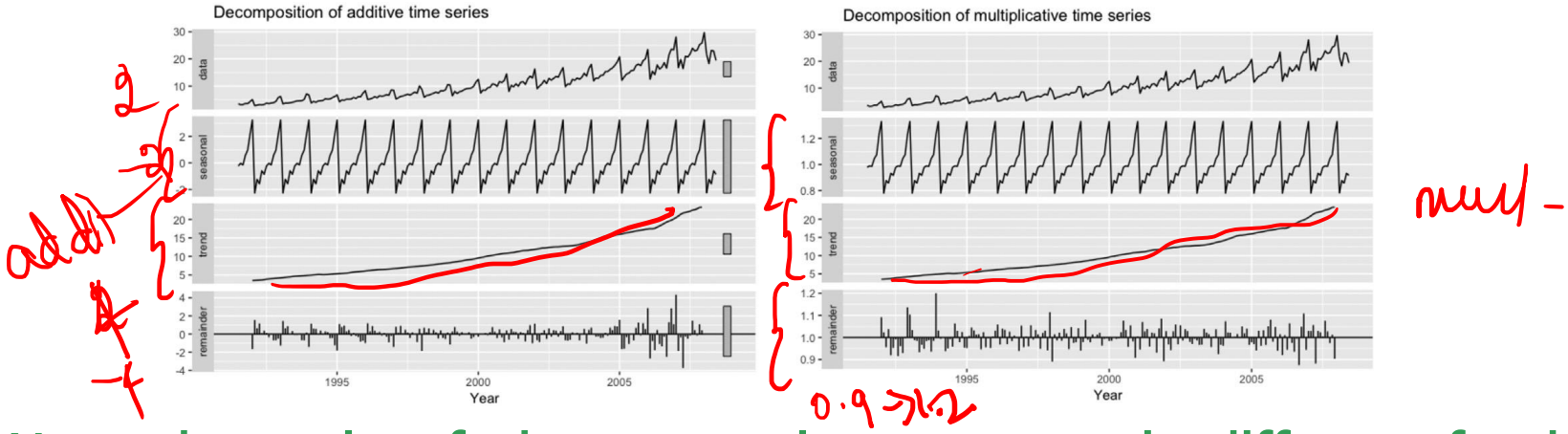
Time series decomposition-Components

- The original time series yt is just an additive or multiplicative combination of components.
- If the magnitude of the seasonal fluctuation or the variation in trend changes with the level of the time series, then multiplicative time series decomposition is the better model.
- The antidiabetic drug sales time series from shows increasing seasonal fluctuation and exponential rise in the long-term trend. Hence, a multiplicative model is more appropriate.



Time series decomposition-Components

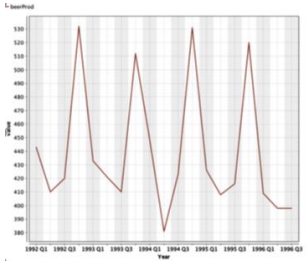
- Figure shows both the additive and multiplicative decomposition of the antidiabetic drug sales time series.



- Note the scale of the seasonal component is different for both decompositions.
- The noise in the additive decomposition is higher post 2005, as it ineffectively struggles to represent increases in the seasonal variation and trend changes.
- The noise is much smaller in the multiplicative decomposition.

Time series decomposition-Classical Decomposition

- To decompose the time series data to its individual components, there are a few different techniques available.



- The classical decomposition technique is simple, intuitive, and serves as a baseline of all other advanced decomposition methods.
- Suppose the time series has yearly seasonality with monthly data as shown in Figure. m represents seasonal period, which is 12 for monthly data with yearly seasonality.
- The classical decomposition technique first estimates the trend component by calculating the long term (say 12 month) moving average.
- The trend component is removed from the time series to get remaining seasonal and noise components.
- The seasonal component can be estimated by average Jan, Feb, Mar, . . ., variance in the remaining series. Once the trend and seasonal components are removed, what is left is noise.

Time series decomposition-Classical Decomposition

- The algorithm for classic additive decomposition is:

1. Estimate the trend Tt: If m is even, calculate 2Xm moving average (m-MA and then a 2-MA); if m is odd, calculate m-moving average. A moving average is the average of last m data points.
2. Calculate detrended series: Calculate yt - Tt for each data point in the series.
3. Estimate the seasonal component St: average (yt - Tt) for each m period. For example, calculate the average of all January values of (yt - Tt) and repeat for all the months. Normalize seasonal value in such a way that mean is zero.
4. Calculate the noise component Et: Et=(yt - Tt - St) for each data point in the series.

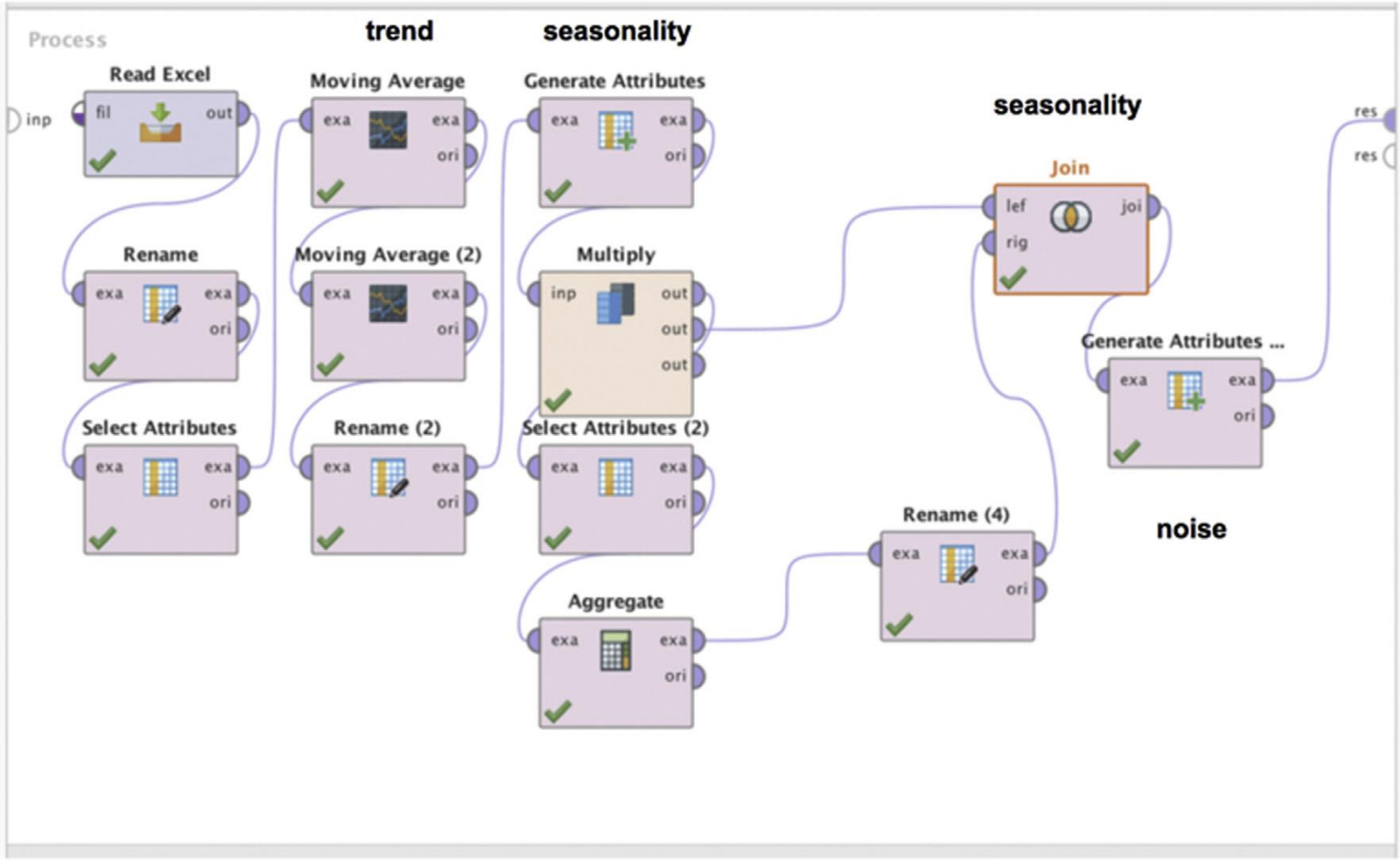
Multiplicative decomposition is similar to additive decomposition: Replace subtraction with division in the algorithm described.

An alternative approach is to convert multiplicative decomposition into additive by applying the logarithmic function on both the sides

$$y_t = T_t \times S_t \times E_t$$

Time series decomposition- Implementation

- A practical example of a time series additive decomposition using RapidMiner will be briefly described.
- The process shown in Figure has simple data pre-processing operators to estimate the trend, seasonal, and error components.



Time series decomposition- Implementation

- Steps to convert the time series into the components are:

Trend: Since the data show four quarter seasonality, a four period moving average operator and a two-period moving average operator is used to estimate the trend component.

Seasonality: The Generate attribute operator is used to calculate the detrended series by finding the difference between the time series and the 234 month moving average values.

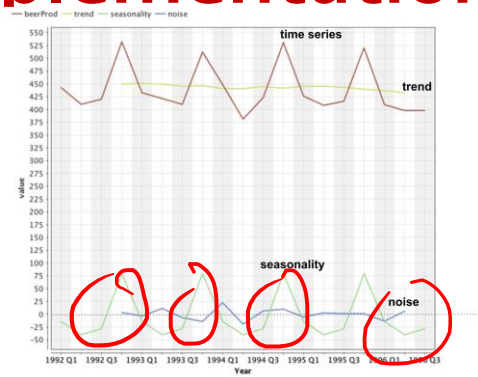
The same operator also extracts quarter value (Q1, Q2, Q3, Q4) from the Year (e.g., 1996 Q1) attribute.

The detrended series has seasonality and noise. To calculate the seasonality, the quarter values have to be averaged using the Aggregate operator. This gives seasonality values for one year. The Join operator is used to repeat the seasonal component for every Q1, Q2, Q3 and Q4 records.

Noise: Noise is calculated by the difference of the time series with the combination of trend and seasonality.

Time series decomposition- Implementation

- The result of the process is shown in Figure

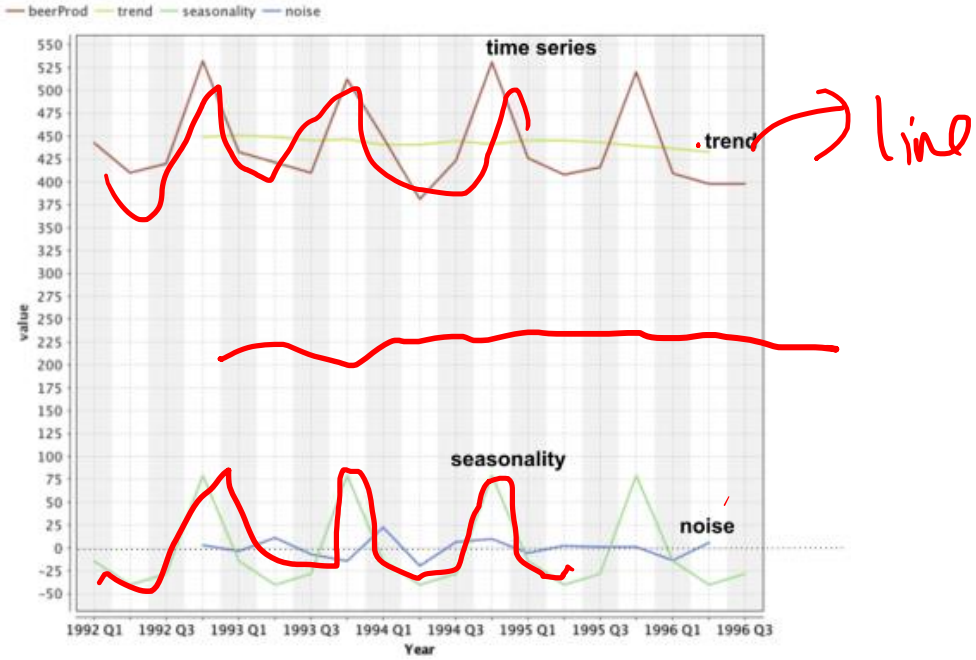


- The initial time series is decomposed into its components. Note that the noise is randomly distributed with the mean as zero.
- Even though the classical decomposition is straight forward, it has some serious limitations.
- Classical decomposition assumes the occurrence of the same seasonal pattern throughout the entire time series. That is too constrictive of an assumption for practical use cases.
- As the 2Xm moving average is used, the first m/2 and the last m/2 data points are not used in the modeling.
- Moreover, classical decomposition is inept at handling anomalies in the data. There are quite a range of advanced time series decomposition techniques. STL (Seasonal and Trend decomposition using Loess), SEATS (Seasonal Extraction in ARIMA Time Series), and X11 (from US Census Bureau)

Time series decomposition- Forecasting Using Decomposed Data

- While decomposing the time series is used to describe and increase the understanding of time series data, it is useful for forecasting as well.
- The idea is to breakdown the time series into its parts, forecast the parts, and put it back together for forecasted future time series values.
- Forecasting the time series through their components is a much easier task. $\hat{y}_t = \hat{S}_t + \hat{T}_t$
- It is assumed that the seasonal component of the time series does not change.
- Hence, the forecast of the seasonal component is the same as the values extracted from the time series. The time series data without the seasonal component is called seasonally adjusted time series.
- It contains just the trend component and noise in the data.
- The seasonally adjusted time series can be forecasted by relatively easier methods: linear or polynomial regression, Holt's method, or ARIMA.

Time series decomposition- Forecasting Using Decomposed Data



- For example, the trend components in Figures can be extended using linear regression.
- Noise is normally distributed with the mean as zero. Hence, it is not forecasted.
- The time series forecast for the future value is the sum of the seasonal forecast and the seasonally adjusted forecast of the trend component.

Smoothing Based Methods

- In the smoothing based approaches, an observation is a function of past few observations.

Time periods: t=1, 2, 3, . . ., n. Time periods can be seconds, days, weeks, months, or years depending on the problem.

Data series: Observation corresponding to each time period above: y1, y2, y3, . . . yn.

Forecasts: Fn+h is the forecast for the h^{th} time period following n. Usually h=1, the next time period following the last data point. However h can be greater than 1. h is called the horizon.

Forecast errors: et=yt - Ft for any given time, t.

- In order to explain the different methods, a simple time series data function will be used, Y(t).
- Y is the observed value of the time series at any time t.
- Furthermore, the observations are made at a constant time interval. If in the case of intermittent data, one can assume that an interpolation scheme is applied to obtain equally spaced (in time) data points.

Smoothing Based Methods- Simple Forecasting Methods

- **Naïve Method**: Probably the simplest forecasting "model." Here one simply assumes that Fn+1, the forecast for the next period in the series, is given by the last data point of the series, yn, Fn+1 = yn
- **Seasonal Naive Method**: If the time series is seasonal, one can forecast better than the naive point estimate. The forecast can assume the value as the previous value of the same season.
- For example, the next January revenue can be assumed as the last known January revenue. $F_{n+1} = y_{n-s}$
- where s is the seasonal period. In the case of monthly data with yearly seasonality, seasonal period is 12.

Average Method:

- Moving up a level, one could compute the next data point as an average of all the data points in the series. In other words, this model calculates the forecasted value, Fn+1, as: $F_{n+1} = \text{Average}(y_n, y_{n-1}, y_{n-2}, \ldots, y_1)$
- Suppose one has monthly data from January 2010 to December 2010 and they want to predict the next January 2011 value, they would simply average the values from January 2010 to December 2010.

Smoothing Based Methods- Simple Forecasting Methods

Moving Average Smoothing

- The obvious problem with a simple average is figuring out how many points to use in the average calculation.
- As the observational data grows (as n increases), should one still use all the n time periods to compute the next forecast? To overcome this problem, one can select a window of the last "k" periods to calculate the average, and as the actual data grows over time, one can always take the last k samples to average, that is, n, n - 1, . . ., n - k + 1.
- In other words, the window for averaging keeps moving forward and, thus, returns a moving average. Suppose in the simple example of the window k=3; then to predict the January 2021 data, a three-month average would be taken using the last three months. When the actual data from January comes in, the February 2021 value is forecasted using January 2021 (n), December 2020 (n - 1) and November 2020 (n - 3 + 1 or n - 2).
- This model will result in problems when there is seasonality in the data (e.g., in December for retail or in January for healthcare insurance), which can skew the average.

$$F_{n+1} = \text{Average}(y_n, y_{n-1}, \ldots, y_{n-k})$$

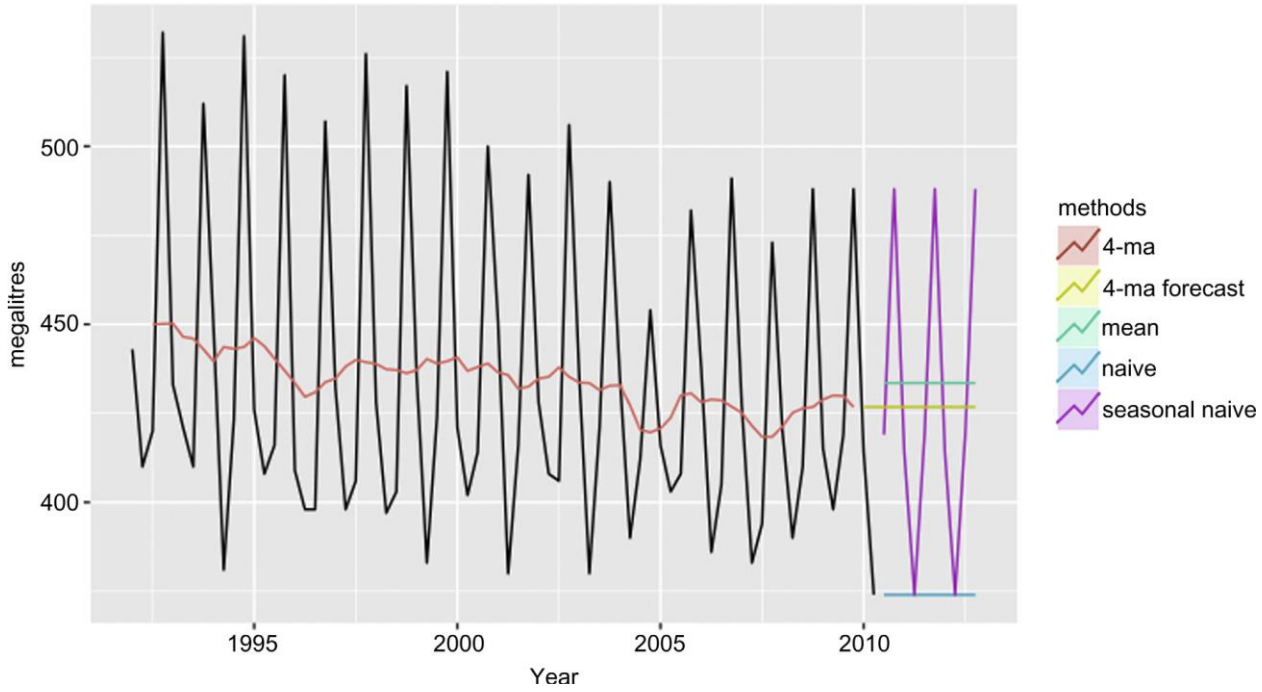
- Moving average smoothens the seasonal information in the time series.

Smoothing Based Methods- Simple Forecasting Methods

Weighted Moving Average Smoothing

- For some cases, the most recent value could have more influence than some of the earlier values.
- Most exponential growth occurs due to this simple effect. The forecast for the next period is given by the model: $F_{n+1} = (a \times y_n + b \times y_{n-1} + c \times y_{n-k})/(a + b + c)$
- where typically a > b > c.

Smoothing Based Methods- Simple Forecasting Methods



- Compares the forecast results for the simple time series introduced earlier.
- Note that all of the mentioned methods are able to make only one-step-ahead forecasts due to the nature of their formulation.
- The coefficients a, b, and c may be arbitrary, but are usually based on some previous knowledge of the time series.

Smoothing Based Methods

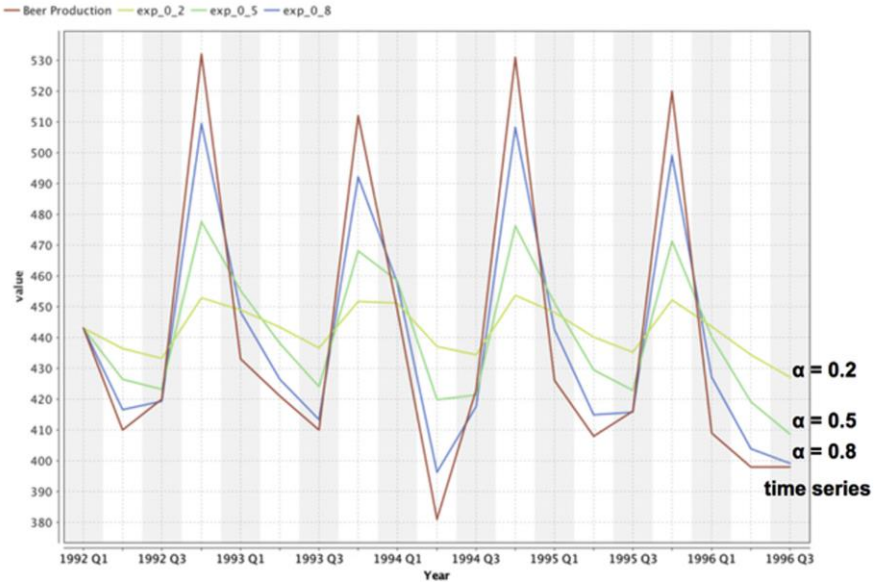
Exponential Smoothing

- Exponential smoothing is the weighted average of the past data, with the recent data points given more weight than earlier data points.
- The weights decay exponentially towards the earlier data points, hence, the name.
- The exponential smoothing is given by $F_{n+1} = \alpha y_n + \alpha(1-\alpha)y_{n-1} + \alpha(1-\alpha)^2 y_{n-2} + \cdots$

α is generally between 0 and 1. Note that α=1 returns the naïve forecast.

Smoothing Based Methods

Exponential Smoothing



- As seen in the charts in Figure, using a higher α results in putting more weight on actual values and the resulting curve is closer to the actual curve, but using a lower α results in putting more emphasis on previously forecasted values and results in a smoother but less accurate fit.
- Typical values for α range from 0.2 to 0.4 in practice.

Smoothing Based Methods

Exponential Smoothing for forecasting

- To forecast the future values using exponential smoothing, Eq. can be rewritten as: $F_{n+1} = \alpha \times y_n + (1 - \alpha) \times F_n$

  is more useful because it involves both actual value yn and forecasted value Fn.

- A higher α value gives an exact fit and a lower value gives a smoother fit.
- Going back to the monthly example, if one wanted to make the February 2011 forecast using not only the actual January 2011 value but also the previously forecasted January 2011 value, the new forecast would have "learnt" the data better.
- This is the concept behind basic exponential smoothing.
- This simple exponential smoothing is the basis for a number of common smoothing based forecasting methods.
- However, the model is suited only for time series without clear trend or seasonality.

Smoothing Based Methods

- The smoothing model has only one parameter, α, and can help smooth the data in a time series so that it is easy to extrapolate and make forecasts.
- Like many methods discussed earlier, the forecast will be flat that is, there is no trend or seasonality factored in.
- Only the level is forecasted. Also, if $F_{n+1} = \alpha \times y_n + (1 - \alpha) \times F_n$ were examined, one would see that forecasts cannot be made more than one-step ahead, because to make a forecast for step (n + 1), the data for the previous step, n, is needed.
- It is not possible to make forecasts several steps ahead, that is, (n + h), using the methods described (where it was simply assumed that Fn+h=Fn+1), where h is the horizon.
- This obviously has limited utility. For making longer horizon forecasts, that is, where h>>1, the trend and seasonality information also needs to be considered.
- Once trend and seasonality are captured, one can forecast the value for any time in the future, not just the values for one step ahead.

Smoothing Based Methods

Holt's Two-Parameter Exponential Smoothing

- A trend is an averaged long-term tendency of a time series. The simplified exponential smoothing model described earlier is not particularly effective at capturing trends.
- An extension of this technique, called Holt's two-parameter exponential smoothing, is needed to accomplish this.
- Recall that exponential smoothing simply calculates the average value of the time series at n + 1.
- If the series also has a trend, then an average slope of the series needs to be estimated as well.
- This is what Holt's two parameter smoothing does by means of another parameter, β.
- A smoothing equation similar to $F_{n+1} = \alpha \times y_n + (1 - \alpha) \times F_n$ is constructed for the average trend at n + 1.
- With two parameters, α and β, any time series with a trend can be modelled and, therefore, forecasted.

Smoothing Based Methods

Holt's Two-Parameter Exponential Smoothing

- The forecast can be expressed as a sum of these two components, average value or "level" of the series, Ln, and trend, Tn, recursively as:

$$F_{n+1} = L_n + T_n$$

where,

$$L_n = \alpha \times y_n + (1 - \alpha) \times (L_{n-1} + T_{n-1}) \text{ and } T_n = \beta \times (L_n - L_{n-1}) + (1 - \beta) \times T_{n-1}$$

- To make future a forecast over an horizon, one can modify Equation to

$$F_{n+h} = L_n + h \times T_n$$

- The values of the parameter can be estimated based on the best fit with the training (past) data.

Smoothing Based Methods

Holt-Winters' Three-Parameter Exponential Smoothing

- When a time series contains seasonality in addition to a trend, yet another parameter, γ, will be needed to estimate the seasonal component of the time series .
- The estimates for value (or level) are now adjusted by a seasonal index, which is computed with a third equation that includes γ.

$$F_{t+h} = (L_t + {}_hT_t)\, S_{t+h-p}$$

$$L_t = \alpha y_t / S_{t-p} + (1-\alpha)(L_{t-1} + T_{t-1})$$

$$T_t = \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1}$$

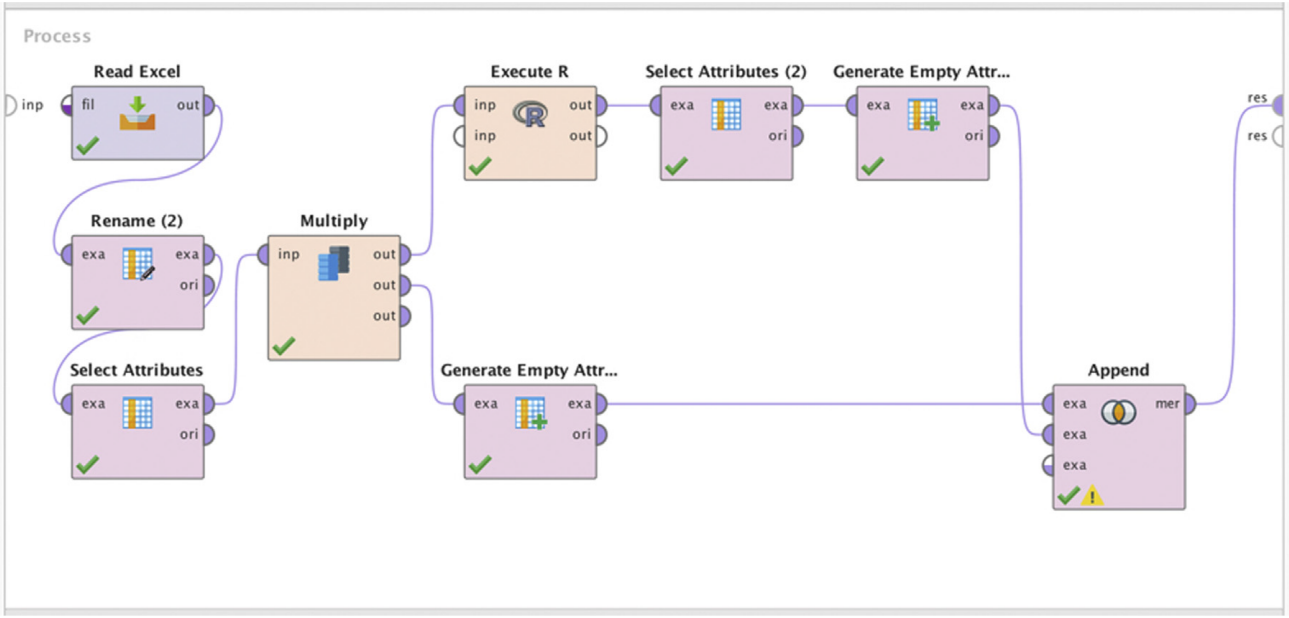
$$S_t = \gamma(y_t / L_t) + (1-\gamma)\, S_{t-p}$$

where p is the seasonality period.

- One can estimate the value of the parameters α, β, and γ from the fitting of the smoothing equation with the training data.

Smoothing Based Methods

Holt-Winters' Three-Parameter Exponential Smoothing-Implementation



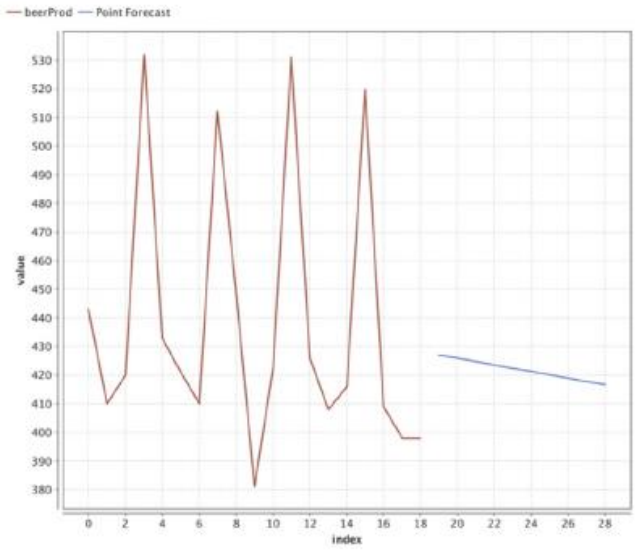
- The data used in the process is the Australian Beer Production time series dataset used in prior time series process.
- After a bit of data pre-processing, like renaming the attributes and selecting the production data, R operator is invoked for Holt and Holt Winters' forecasting function.
- The Execute R operator executes the R script entered in the operator.

Smoothing Based Methods

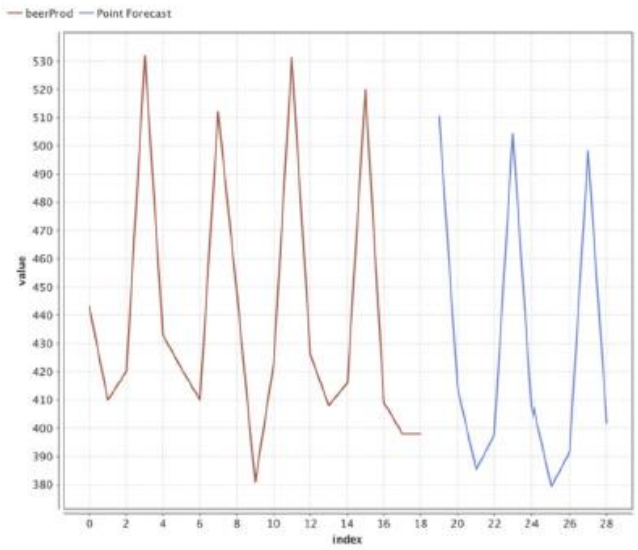
Holt-Winters' Three-Parameter Exponential Smoothing-Implementation

R Script for Holt-Winters' Forecasting

```
rm_main = function(data)
{
library(forecast)
  inp <- ts(data, freq = 4)
    y <- holt(inp, h = 10)   # or hw(inp, h= 10) for Holt Winters' smoothing
    df <- as.data.frame(y)
    return(list(df))
}
```



Holt's Two-Parameter Smoothing



Holt-Winters' Three-Parameter Smoothing

Holt's forecast, only trend and level are forecasted and in Holt Winters' forecasting both trend and seasonality are forecasted.

Regression Based Methods

- In the regression based methods, the variable time is the predictor or independent variable and the time series value is the dependent variable.
- Regression based methods are generally preferable when the time series appears to have a global pattern.
- The idea is that the model parameters will be able to capture these patterns and, thus, enable one to make predictions for any step ahead in the future under the assumption that this pattern is going to be repeated.
- For a time series with local patterns instead of a global pattern, using a regression based approach requires one to specify how and when the patterns change, which is difficult.
- For such a series, smoothing approaches work best because these methods usually rely on extrapolating the most recent local pattern as seen earlier.

Regression Based Methods

- Fig. A shows antidiabetic drug revenue and Fig.B shows the economy class passenger volume in Sydney- Melbourne4 route.
- A regression based forecasting method would work well for the antidiabetic drug revenue series because it has a global pattern.
- However the passenger volume series shows no clear start or end for any patterns. It is preferable to use smoothing based methods to attempt to forecast this second series.

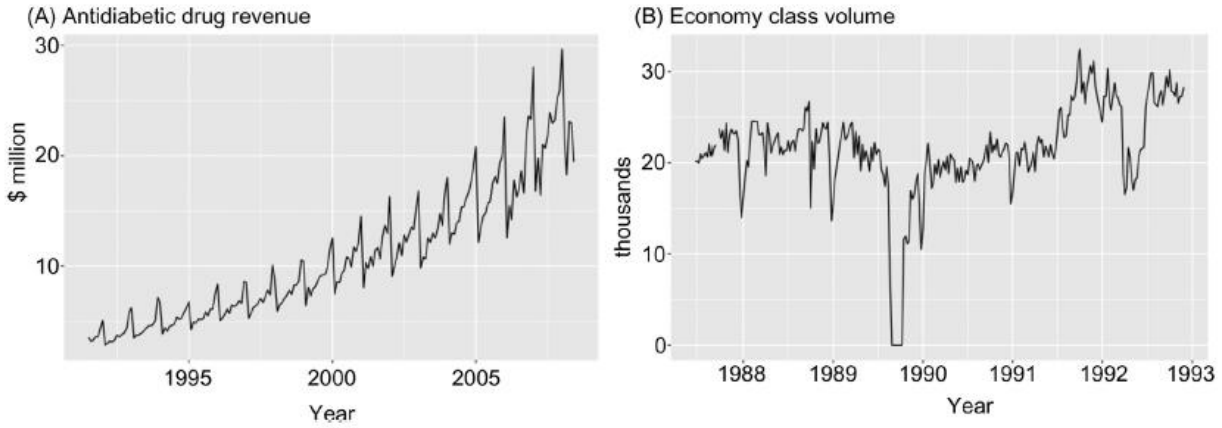


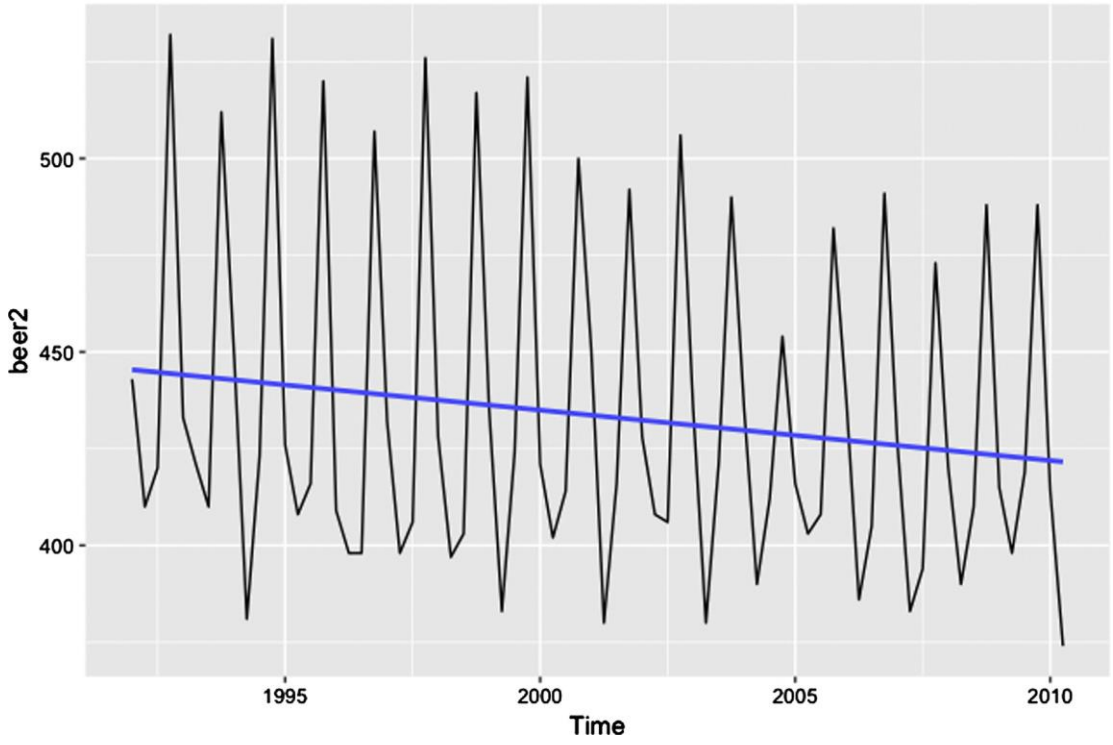
FIGURE 12.13

Global and local patterns. (A) Antidiabetic drug revenue - global pattern and (B) Airline economy class passenger volume between Sydney and Melbourne - local pattern.

Regression Based Methods

Regression

- The simplest of the regression based approaches for analyzing a time series is using linear regression.
- One assumes the time period is the independent variable and attempts to predict the time series value using this.
- For the Australian beer dataset used so far, the chart in Figure shows a linear regression fit.



Regression Based Methods

Regression

- As can be seen, the linear regression model is able to capture the long-term tendency of the series, but it does a poor job of fitting the data.
- Sophisticated polynomial functions can be used to improve the fit.
- Polynomial regression is similar to linear regression except that higher-degree functions of the independent variable are used (squares and cubes on the time variable).
- Since the global trend here is straight decline, it is difficult to argue that the cubic polynomial does a significantly better job.
- However in either of these cases, one is not limited to a one-step-ahead forecast of the simple smoothing methods.

Regression Based Methods

Regression with seasonality

- The linear regression trend-fitting model can be significantly improved by simply accounting for seasonality.
- This is done by introducing seasonal dummy variables for each period (quarter), of the series, which triggers either 1 or 0 in the attribute values as seen in Fig.

Year	time	Quarter = Q1	Quarter = Q2	Quarter = Q3	Quarter = Q4	beerProd
1992 Q1	1	1	0	0	0	443
1992 Q2	2	0	1	0	0	410
1992 Q3	3	0	0	1	0	420
1992 Q4	4	0	0	0	1	532
1993 Q1	5	1	0	0	0	433
1993 Q2	6	0	1	0	0	421
1993 Q3	7	0	0	1	0	410
1993 Q4	8	0	0	0	1	512
1994 Q1	9	1	0	0	0	449
1994 Q2	10	0	1	0	0	381

Regression Based Methods

Regression with seasonality

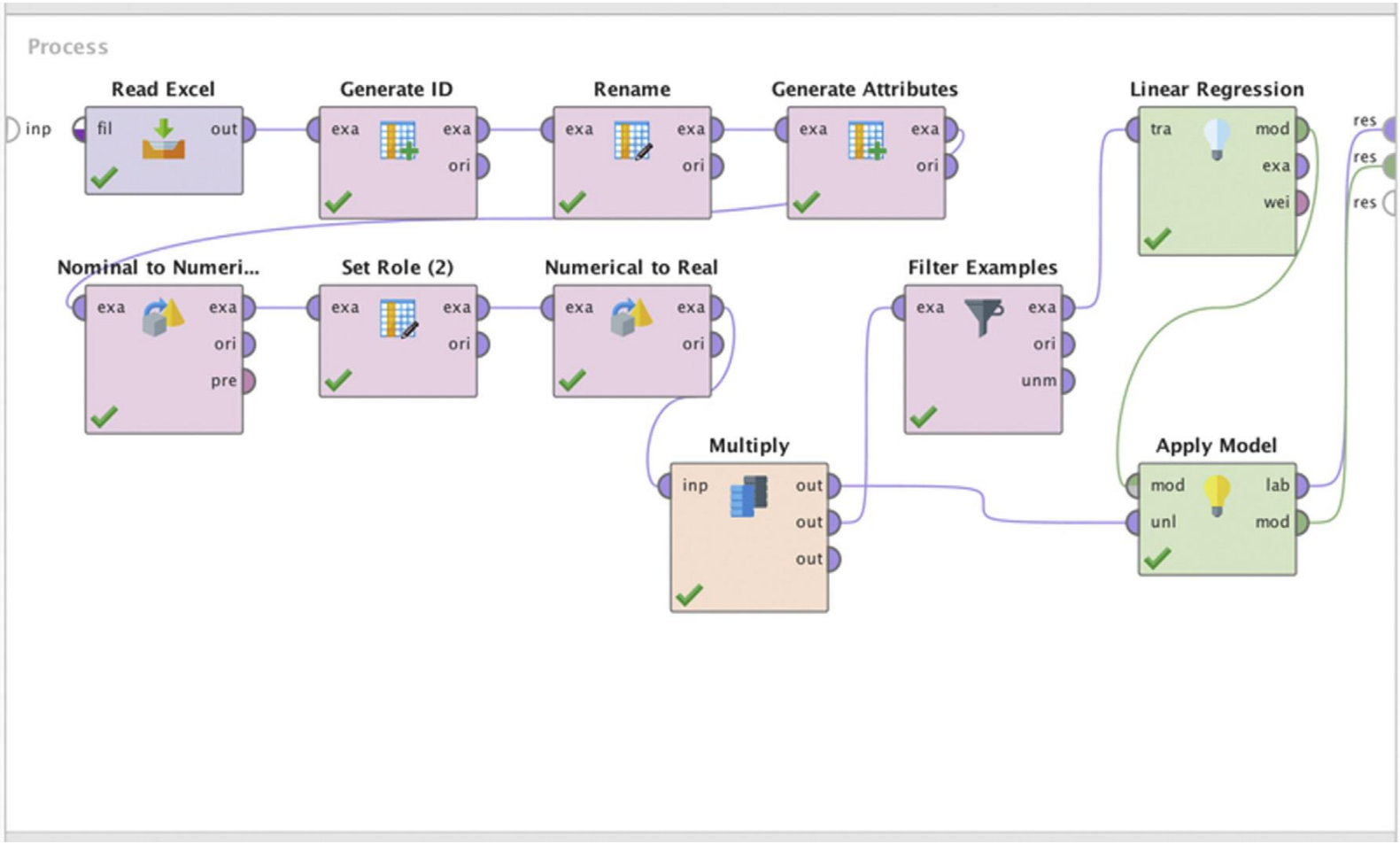
- In this example, four new attributes are added to the original dataset to indicate which quarter the record belongs to.
- The attribute value of (Quarter =Q1) is turned 1 if the quarter is Q1 and so on. Just this trivial addition to the predictors of the linear regression model can yield a surprisingly good fit in most datasets with clear seasonality.
- Although the model equation may appear a bit complicated, in reality it is just a linear regression model with four variables: the time period and four dummy variables for each quarter of a year.
- The independent variable time captures the level and the long-term trend. The four dummy seasonal variables capture the seasonality.
- This regression equation can be used for predicting any future value beyond n11, and thus, has significantly more utility than the simpler counterparts in the smoothing side.

$$\begin{aligned}\text{Forecast} = 442.189 - 1.132 \times \text{time} - 27.268 \times (\text{Quarter} = \text{Q2}) \\ - 16.336 \times (\text{Quarter} = \text{Q3}) + 92.882 \times (\text{Quarter} = \text{Q4})\end{aligned}$$

- There is of course no reason to use linear regression alone to capture both trend and seasonality.
- More sophisticated models can easily be built using polynomial equations along with the sine and cosine function to model seasonality.

Regression Based Methods

Regression with seasonality-Implementation



Regression Based Methods

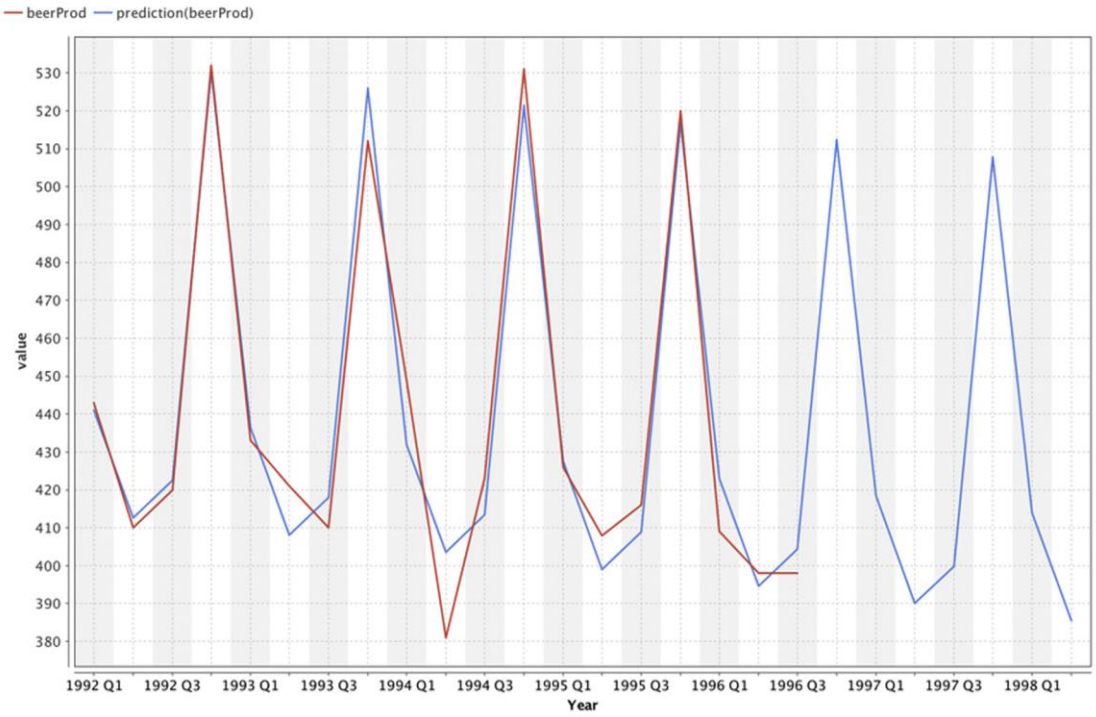
Regression with seasonality-Implementation

- **1. Add time attribute**: After the dataset is read, a new attribute "time" is generated to indicate the consecutive time period of each example record. First record "1992 Q1" is tagged as 1 and the next as 2, and so on. This is essential in all regression based forecasting because time is the independent variable and will be part of the regression equation. It is important to sort the dataset before adding the sequential time attribute.
- **2. Extract seasonal attribute**: The next step is to extract "Q1" from the "1992 Q1" attribute. It can be achieved with Generate attribute and cut () function to extract Q1 from the text. If the time series has weekly or monthly seasonality, corresponding identifiers like weekday or month id have to be extracted.
- **3. Generation of independent seasonal attributes**: The quarter attribute has to be pivoted to "Is Q1", "Is Q2", . . . attributes. The Nominal to Numerical conversion operator is used to generate these attributes. The value of the attribute is either 1 or 0, depending on the record's seasonal period.

Regression Based Methods

Regression with seasonality-Implementation

- **4. Modeling:** A linear regression model is used to fit the equation with the training data. A polynomial regression learner can be used for a better fit.
- **5. Forecasting:** The dataset also contains the placeholders for future periods. The Apply model operator is used to visualize the fit of the model for the training period and to forecast the future horizon.



Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

Building blocks- Auto Correlation

- Correlation measures how two variables are dependent on each other or if they have a linear relationship with each other.
- Consider the time series shown in Figure.

Year	prod	prod-1	prod-2	prod-3	prod-4	prod-5	prod-6
1992 Q1	443	?	?	?	?	?	?
1992 Q2	410	443	?	?	?	?	?
1992 Q3	420	410	443	?	?	?	?
1992 Q4	532	420	410	443	?	?	?
1993 Q1	433	532	420	410	443	?	?
1993 Q2	421	433	532	420	410	443	?
1993 Q3	410	421	433	532	420	410	443
1993 Q4	512	410	421	433	532	420	410
1994 Q1	449	512	410	421	433	532	420
1994 Q2	381	449	512	410	421	433	532
1994 Q3	423	381	449	512	410	421	433
1994 Q4	531	423	381	449	512	410	421
1995 Q1	426	531	423	381	449	512	410

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

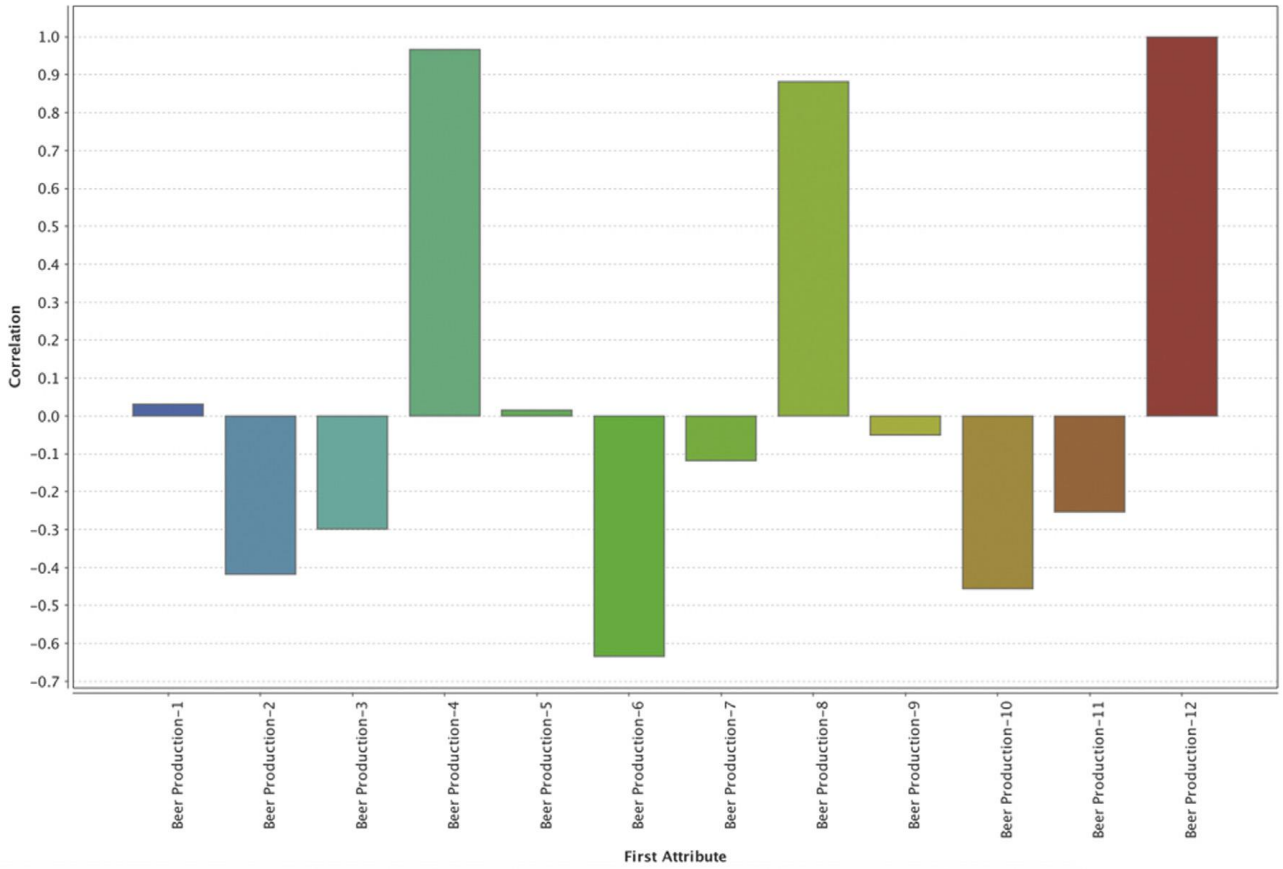
Building blocks- Auto Correlation

- The second column "prod" shows the data for the simple time series.
- In the third column, data are lagged by one step. 1992 Q1 data is shown in 1992 Q2.
- This new series of values is termed a "1-lag" series.
- There are an additional 2-lag, 3-lag, . . ., n-lag series in the dataset.
- Notice that there is a strong correlation between the original time series "prod" and 4-lag "prod-4." They tend to move together.
- This phenomenon is called autocorrelation, where the time series is correlated with its own data points, with a lag.
- As in a multivariate correlation matrix, one can measure the strength of correlation between the original time series and all the lag series.
- The plot of the resultant correlation matrix is called an Autocorrelation Function (ACF) chart.
- The ACF chart is used to study all the available seasonality in the time series.

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)

Building blocks- Auto Correlation

- From Figure it can be concluded that the time series is correlated with the 4th, 8th, and 12th lagged quarter due to the yearly seasonality.



Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

Building blocks-Autoregressive Models

- Autoregressive models are regression models applied on lag series generated using the original time series.
- Recall in multiple linear regression, the output is a linear combination of multiple input variables.
- In the case of autoregression models, the output is the future data point and it can be expressed as a linear combination for past p data points. p is the lag window.
- The autoregressive model can be denoted as the equation:

$$y_t = l + \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + \cdots + \alpha_p y_{t-p} + e$$

- where, l is the level in the dataset and e is the noise. α are the coefficients that need to be learned from the data.
- This can be referred to as an autoregressive model with p lags or an AR(p) model.
- In an AR(p) model, lag series is a new predictor used to fit the dependent variable, which is still the original series value, Yt.

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

Building blocks- Stationary Data

- In a time series with trends or seasonality, the value is affected by time.
- A time series is called stationary when the value of time series is not dependent on time
- For instance, random white noise is a stationary time series.
- Daily temperature at a location is not stationary as there will be a seasonal trend and it is affected by time.
- Meanwhile, the noise component of a time series is stationary.
- Stationary time series do not have any means of being forecasted as they are completely random.

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

Building blocks- Stationary Data

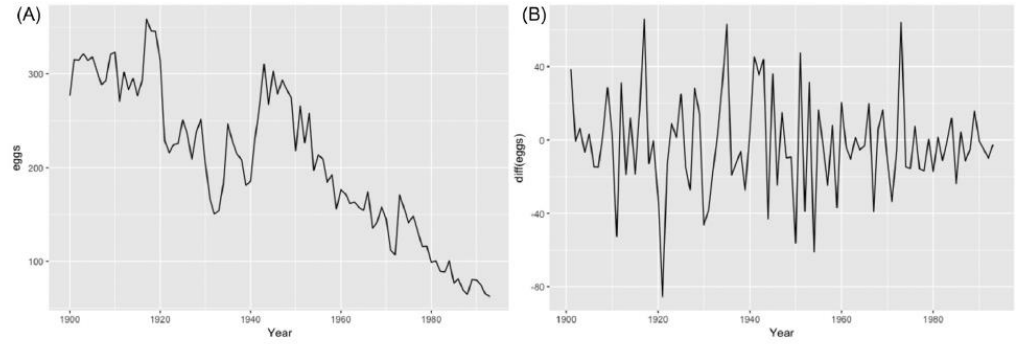


FIGURE 12.20
(A) Non-stationary Time series and (B) Stationary time series.

- **Stationary time series do not have any means of being forecasted as they are completely random.**
- **Figure A- Non-Stationary data because of the presence of both trend and seasonality and Fig. B is an example of stationary data because there is no clear trend or seasonality.**

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)

Building blocks- Differencing

- A non-stationary time series can be converted to a stationary time series through a technique called differencing.
- Differencing series is the change between consecutive data points in the series. $y'_t = y_t - y_{t-1}$
- This is called first order differencing. Previous figure shows a time series and a first order differenced time series.
- In some cases, just differencing once will still yield a nonstationary time series.
- In that case Second order differencing is the change between two consecutive data points in a first order differenced time series.
- a second order differencing is required.
- To generalize, differencing of order d is used to convert nonstationary time series to stationary time series.

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

Building blocks- Differencing

- **Seasonal differencing** is the change between the same period in two different seasons. Assume a season has period, m. $y'_t = y_t - y_{t-m}$
- This is similar to the Year-over-Year metric used commonly in business financial reports. It is also called as m-lag first order differencing.
- Fig. shows the seasonal differencing of the Australian Beer production dataset and the seasonal first order differencing of the same series with the seasonal lag as 4—to factor in the number of quarters in a year.

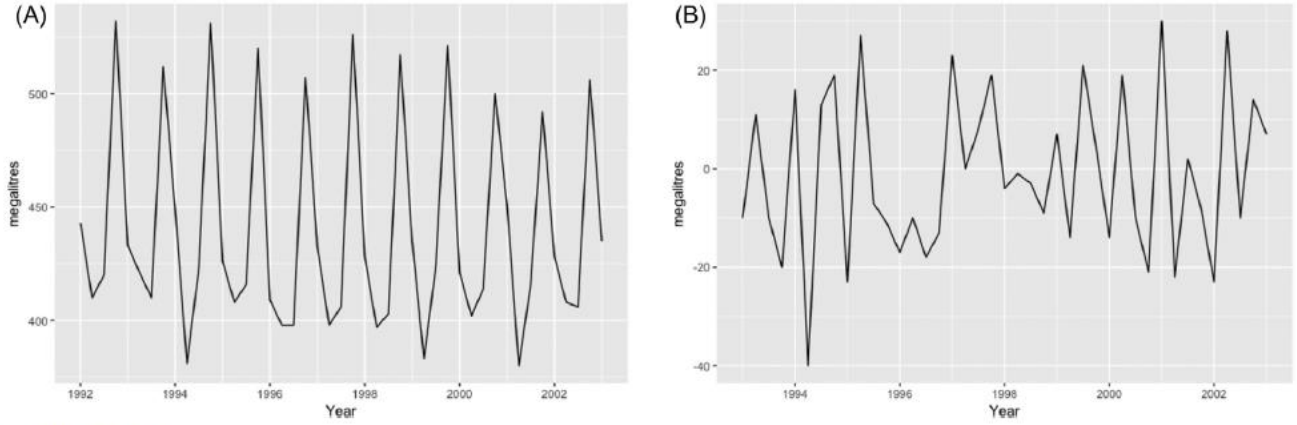


FIGURE 12.21
(A) Time series (B) Seasonal differencing of the time series.

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

Building blocks- Moving Average of Error

- In addition to creating a regression of actual past "p" values, one can also create a regression equation involving forecast errors of past data and use it as a predictor. $y_t = I + e_t + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots + \theta_q e_{t-q}$
- where ei is the forecast error of data point i.
- This makes sense for the past data points but not for data point t because it is still being forecasted.
- Hence, et is assumed as white noise. The regression equation for yt can be understood as the weighted (θ) moving average of past q forecast errors.
- This is called Moving Average with q lags model or MA(q).

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

Building blocks- Autoregressive Integrated Moving Average

- The Autoregressive Integrated Moving Average (ARIMA) model is a combination of the differenced autoregressive model with the moving average model.
- It is expressed as: $y'_t = I + \alpha_1 y'_{t-1} + \alpha_2 y'_{t-2} + \cdots + \alpha_p y'_{t-p} + e_t + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots + \theta_q e_{t-q}$
- The AR part of ARIMA shows that the time series is regressed on its own past data.
- The MA part of ARIMA indicates that the forecast error is a linear combination of past respective errors.
- The I part of ARIMA shows that the data values have been replaced with differenced values of d order to obtain stationary data, which is the requirement of the ARIMA model approach.
- Why would one need to get to this level of complexity? The ARIMA model is effective in fitting past data with this combination approach and help forecast future points in a time series.

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

Building blocks- Autoregressive Integrated Moving Average

- Eq. shows the predictors are the lagged p data points for the autoregressive part and the lagged q errors are for the moving average part, which are all differenced.
- The prediction is the differenced yt in the dth order.
- This is called the ARIMA(p,d,q) model. Estimating the coefficients α and θ for a given p,d,q is what ARIMA does when it learns from the training data in a time series.
- Specifying p,d,q can be tricky (and a key limitation) but one can tryout different combinations and evaluate the performance of the model.
- Once the ARIMA model is specified with the value of p,d,q, the coefficients of Eq. need to be estimated.
- The most common way to estimate is through the Maximum Likelihood Estimation.

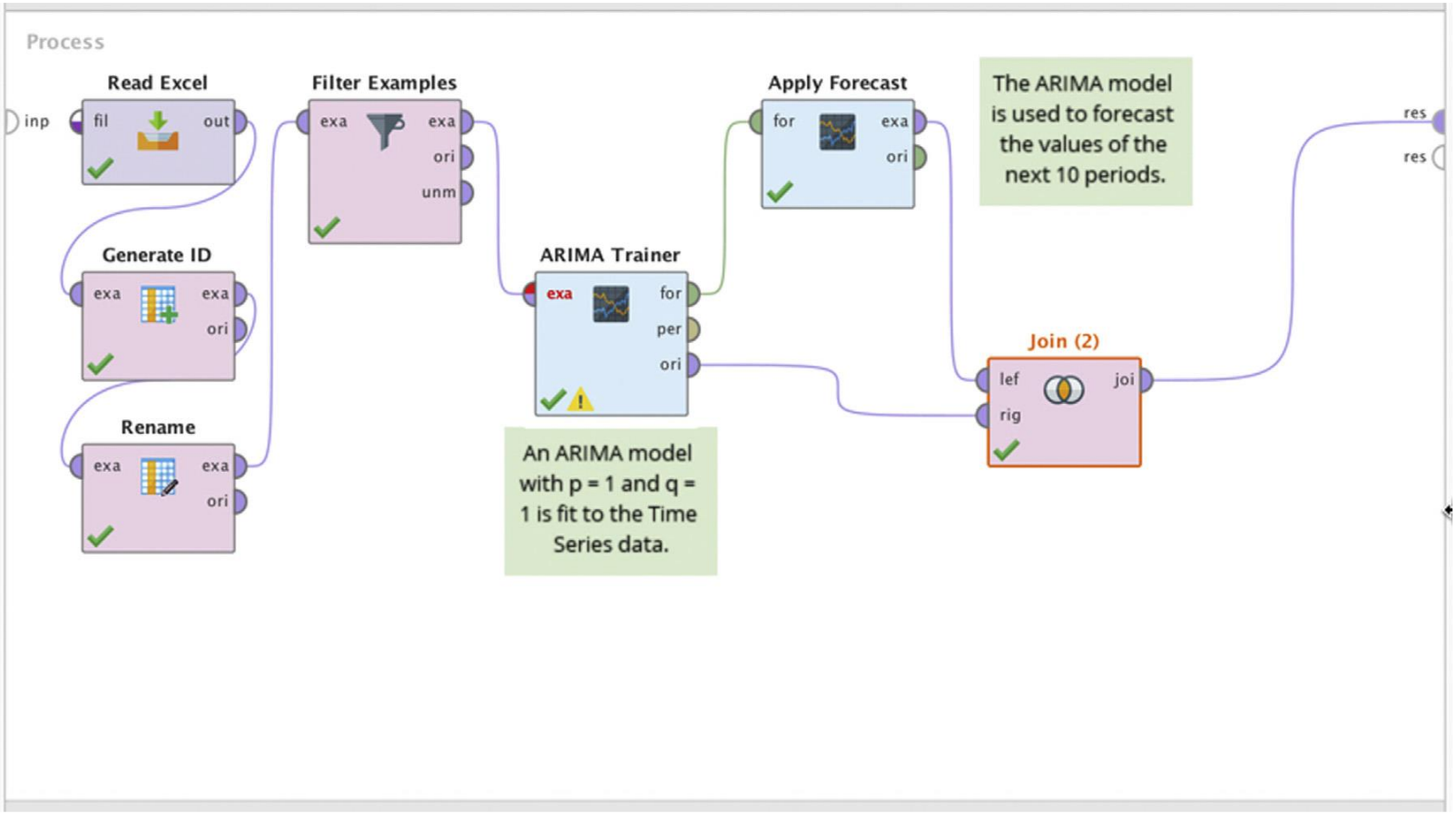
Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)

Autoregressive Integrated Moving Average

- It is similar to the Least Square Estimation for the regression equation, except MLE finds the coefficients of the model in such a way that it maximizes the chances of finding the actual data.
- ARIMA is a generalized model. Some of the models discussed in this chapter are special cases of an ARIMA model.
- For example,
- ARIMA (0,1,0) is expressed as $y_t = y_{t-1} + e$. It is the naive model with error, which is called the Random walk model.
- ARIMA (0,1,0) is expressed as $y_t = y_{t-1} + e + c$. It is a random walk model with a constant trend. It is called random walk with drift.
- ARIMA (0,0,0) is yt-e or white noise
- ARIMA (p,0,0) is the autoregressive model

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

How to Implement

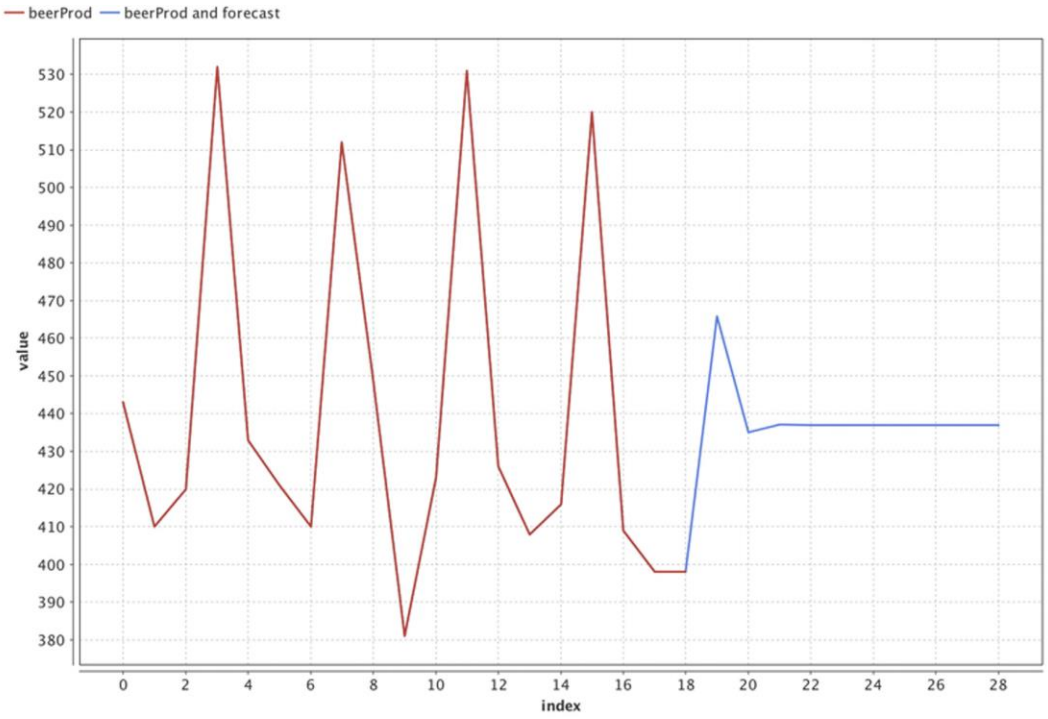


Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

- Step 1: The Australian beer production dataset has two attributes: Year and beer production value in megaliters.
- The role of Year should be set as ID and the only attribute in the dataset should be the time series value.
- A sequential time period is generated and added to the dataset as an anchor for the charts.
- Step 2: The ARIMA trainer operator can be found under Extension > Time Series > Forecast > ARIMA. The operator has these parameters:
- p: order of autoregression part. It is set as 1
- d: degree of differencing. Set as 0
- q: order of the moving average of the error part. Set as 1
- criterion: criterion to minimize for coefficient selection. Set as aic.
- As a start, the model parameters are specified as ARIMA(1,0,1). One can further all the combination of p,d,q are tried out to select the most optimal combination.

Regression Based Methods-Autoregressive Integrated Moving Average (ARIMA)-

- Step 3: The Apply the forecast operator takes the input of the forecasted model and applies the model for future horizon.
- Forecast horizon is set as 10 to predict 10 more data points at the end of the time series.
- The forecasted data is then joined with the original dataset using the Join operator



Regression Based Methods- Seasonal ARIMA

- The ARIMA model can be further enhanced to take into account of the seasonality in the time series. Seasonal ARIMA is expressed by the notion
- ARIMA(p,d,q)(P,D,Q)m where
- p is the order of nonseasonal autoregression
- d is the degree of differencing
- q is the order of nonseasonal moving average of the error
- P is the order of the seasonal autoregression
- D is the degree of seasonal differencing
- Q is the order of seasonal moving average of the error
- m is the number of observations in the year (for yearly seasonality)
- In terms of the equation, the seasonal part of ARIMA is similar to the terms used in the nonseasonal part, except that it is backshifted m times, the seasonal period.
- For example, the differencing is performed not with the consecutive data points but with the data points with m lags.
- If one has quarterly seasonality in monthly grain, each data point is differenced with the same month last quarter.

Regression Based Methods- Seasonal ARIMA

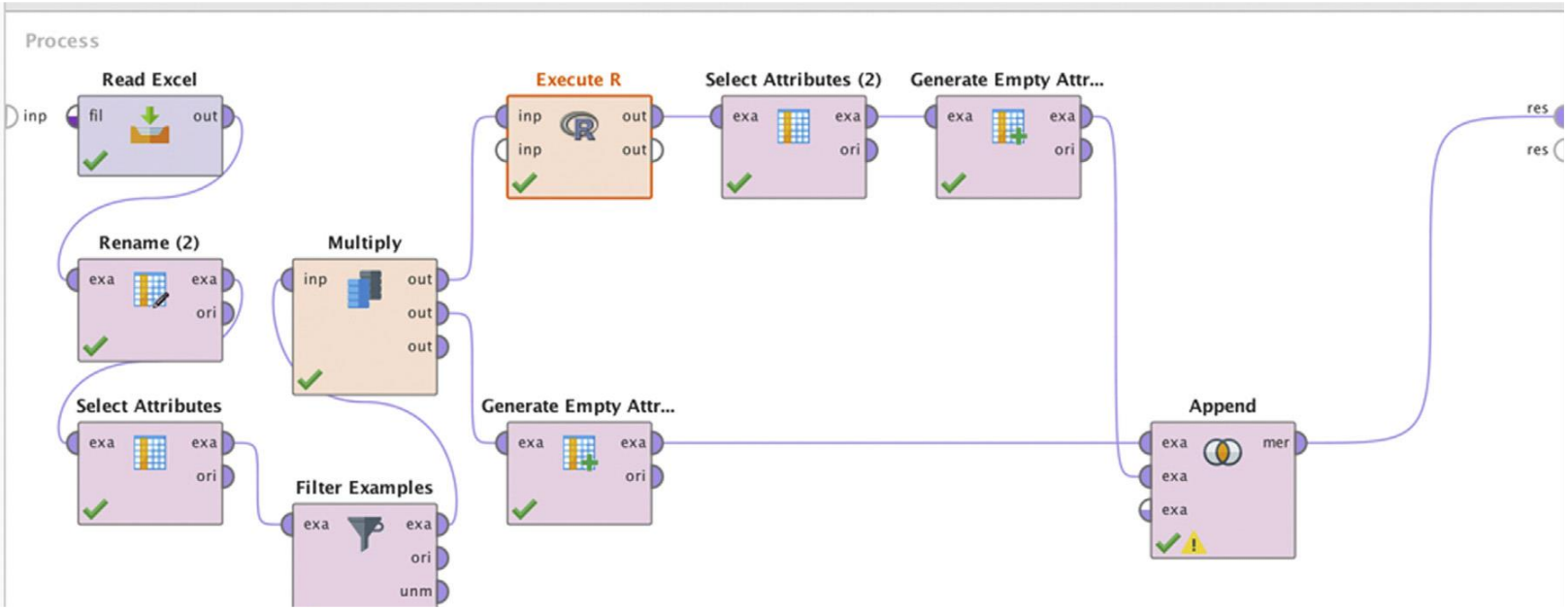
How to Implement

- To implement seasonal ARIMA, Execute R operator from the R extension for RapidMiner is used.
- The RapidMiner process shown in Fig. looks similar to the process built for the Holt-Winters' smoothing model.
- The difference is in the R code inside the Execute R operator. The process has a data preparation step before feeding data to R and a post process to combine the forecast and original data for visualization.
- The Australian beer production dataset is used for this process.
- The R code inside the Execute R operator is shown here.
- It uses the Forecast package in R to activate the arima() function. In this processes the auto.arima() function was used, which automatically selects the best parameters for (p,d,q)(P,D,Q)m.

Regression Based Methods- Seasonal ARIMA

How to Implement

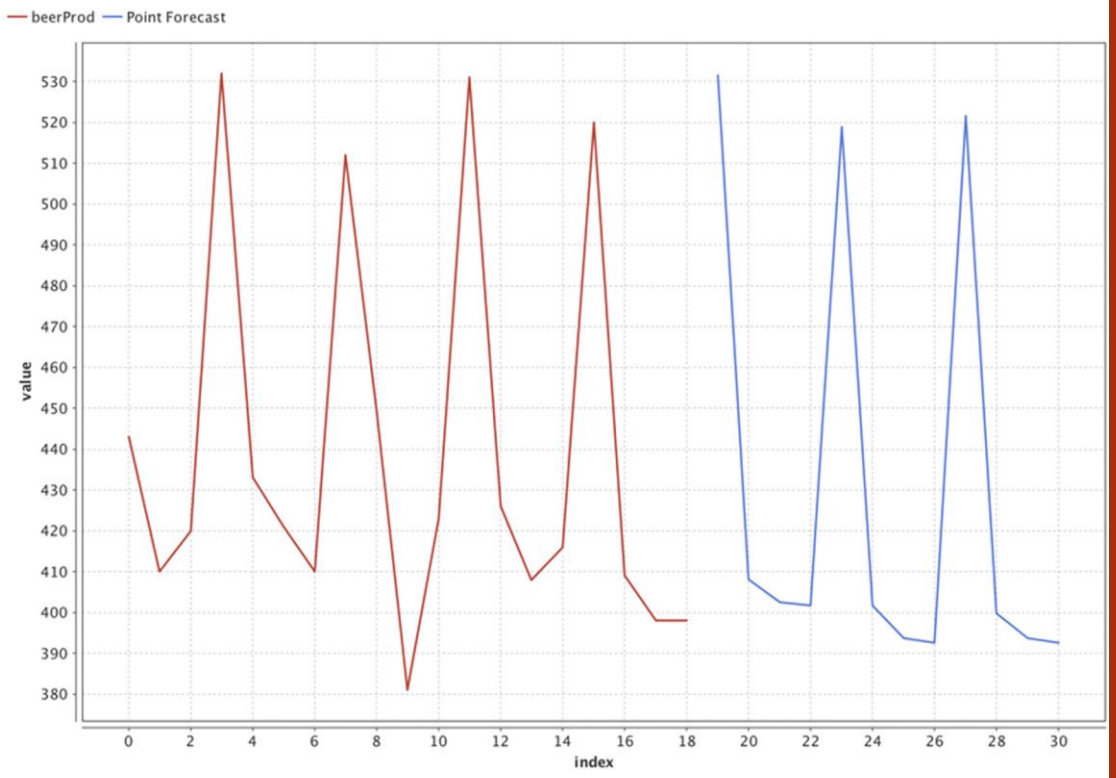
- The optimal parameters for the Beer production dataset is ARIMA(1,0,0)(1,1,0)4.
- The seasonal ARIMA model is used to forecast the future 12 data points using the forecast() function.



Regression Based Methods- Seasonal ARIMA

```
rm_main=function(data)
{
library(forecast)
  inp<- ts(data, freq=4)
  y<- auto.arima(inp)
    df<- as.data.frame(forecast(y, h=12))
    return(list(df))
}
```

The forecast output is shown in Figure.

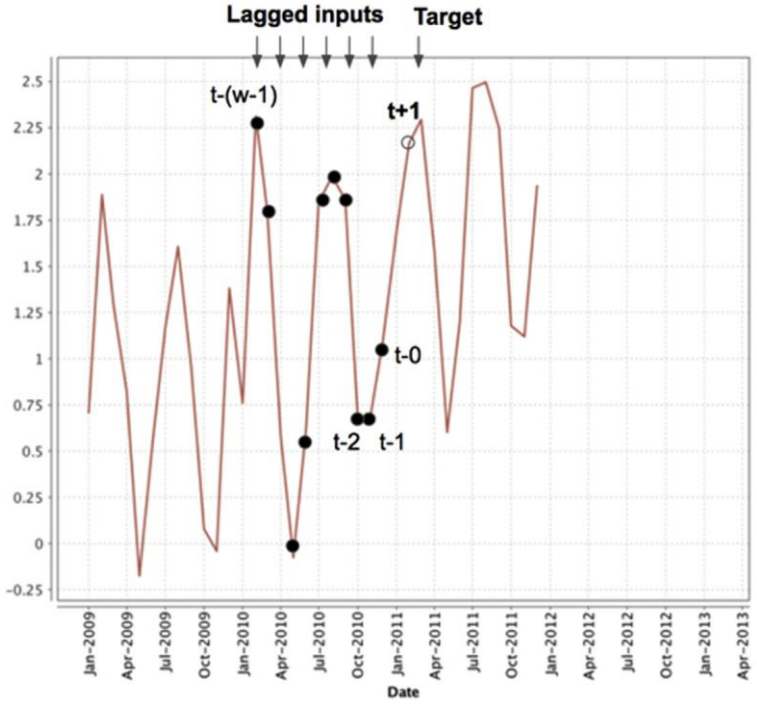


Regression Based Methods- Seasonal ARIMA

- The model seems to have accounted for both the trend and seasonality in the time series.
- ARIMA is one of the most used forecasting techniques in businesses today.
- It has strong underpinning in statistics, has been field tested for decades, and is effective in modeling seasonal and nonseasonal time series.
- Specifying the ARIMA modeling parameters beforehand might seem arbitrary.
- However, one can test the fit of the model for many combination or use meta functions like auto.arima() for the Optimize parameters operators.

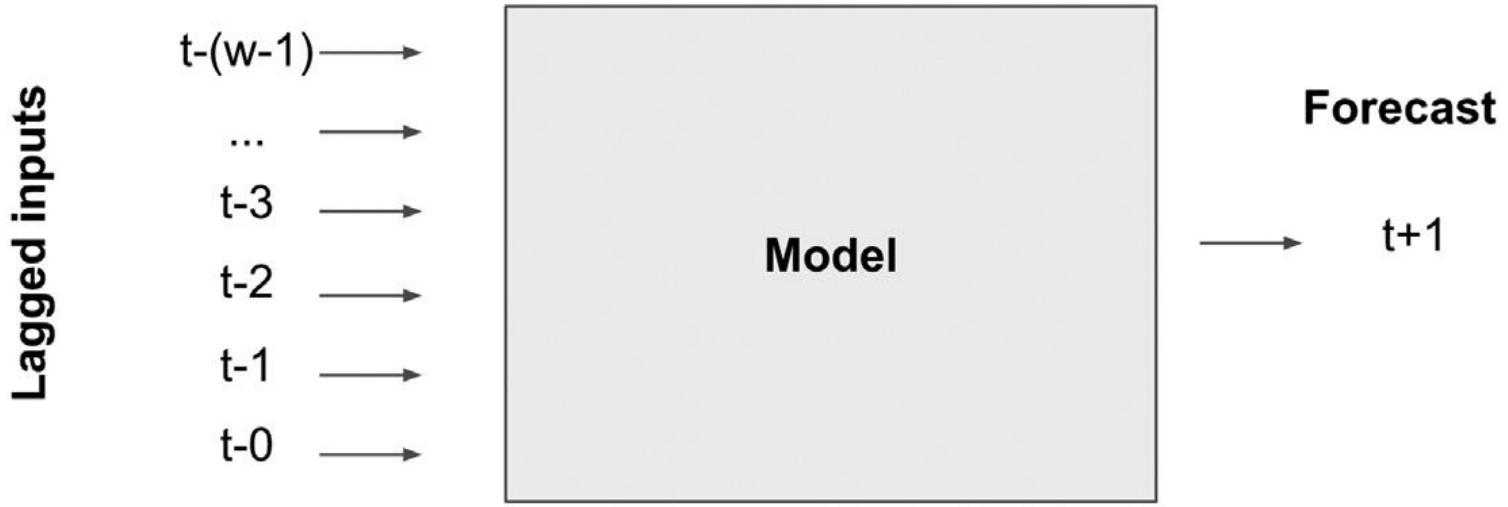
Machine Learning Methods

- A time series is a unique dataset where the information used to predict future data points can be extracted from past data points.
- A subset of the past known data points can be used as inputs to an inferred model to compute the future data point as an output.
- Figure shows the general framework of the approach, which is similar to supervised learning techniques.
- Standard machine learning techniques are used to build a model based on the inferred relationship between input (past data) and target (future data).



Machine Learning Methods

- In order to use the supervised learners on time series data, the series is transformed into cross-sectional data using a technique called windowing.
- This technique defines a set of consecutive time series data as a window, where the latest record forms the target while other series data points, which are lagged compared to the target, form the input variables.



Machine Learning Methods

- As consecutive windows are defined, the same data point in the series may function as the target for one cross-sectional window and the input variable for other cross-sectional windows.
- Once a sufficient number of windows are extracted from the dataset, a supervised model can be learned based on the inferred relationship between the lagged input variables and the target variable.
- This is similar to an autoregressive model where past p data points are used to predict the next data point.
- Any of the supervised learners discussed in the Classification or Regression can be applied to learn and predict the target variable—the next time step in the series.
- The inferred model can be used to predict a future time series data point based on the last window of the time series. This gives visibility into one future data point.
- The new predicted data point can be used to define a new window and predict one more data point into the future. This subroutine can be repeated until all future predictions are made.

Machine Learning Methods

- **Windowing:**
- **The purpose of windowing is to transform the time series data into a generic machine learning input dataset.**
- **Fig. shows a sample windowing and cross-sectional data extraction from the time series dataset.**

Date	inputYt
Jan 1, 2009	0.709
Feb 1, 2009	1.886
Mar 1, 2009	1.293
Apr 1, 2009	0.822
May 1, 2009	-0.173
Jun 1, 2009	0.552
Jul 1, 2009	1.169
Aug 1, 2009	1.604
Sep 1, 2009	0.949
Oct 1, 2009	0.080
Nov 1, 2009	-0.040
Dec 1, 2009	1.381
Jan 1, 2010	0.761
Feb 1, 2010	2.312
Mar 1, 2010	1.795
Apr 1, 2010	0.586
May 1, 2010	-0.077
Jun 1, 2010	0.613
Jul 1, 2010	1.845

(A) Time series data set

Window id	inputYt + 1 (horizon)	inputYt - 5	inputYt - 4	inputYt - 3	inputYt - 2	inputYt - 1	inputYt - 0
0	1.169	0.709	1.886	1.293	0.822	-0.173	0.552
1	1.604	1.886	1.293	0.822	-0.173	0.552	1.169
2	0.949	1.293	0.822	-0.173	0.552	1.169	1.604
3	0.080	0.822	-0.173	0.552	1.169	1.604	0.949
4	-0.040	-0.173	0.552	1.169	1.604	0.949	0.080
5	1.381	0.552	1.169	1.604	0.949	0.080	-0.040
6	0.761	1.169	1.604	0.949	0.080	-0.040	1.381
7	2.312	1.604	0.949	0.080	-0.040	1.381	0.761
8	1.795	0.949	0.080	-0.040	1.381	0.761	2.312
9	0.586	0.080	-0.040	1.381	0.761	2.312	1.795
10	-0.077	-0.040	1.381	0.761	2.312	1.795	0.586
11	0.613	1.381	0.761	2.312	1.795	0.586	-0.077
12	1.845	0.761	2.312	1.795	0.586	-0.077	0.613
13	1.984	2.312	1.795	0.586	-0.077	0.613	1.845
14	1.861	1.795	0.586	-0.077	0.613	1.845	1.984
15	0.661	0.586	-0.077	0.613	1.845	1.984	1.861

(B) Cross-sectional data set

Machine Learning Methods

Windowing:

- The characteristics of the windows and the cross-sectional data extractions are specified by the parameters of the windowing process.
- The following parameters of windowing allow for changing the size of the windows, the overlap between consecutive windows, and the prediction horizon which is used for forecasting.

1. Window Size: Number of lag points in one window excluding the target data point.

2. Step: Number of data points between the first value of the two consecutive windows. If the step is 1, maximum number of windows can be extracted from the time series dataset.

3. Horizon width: The prediction horizon controls how many records in the time series end up as the target variable. The common value for the horizon width is 1.

4. Skip: Offset between the window and horizon. If the skip is zero, the consecutive data point(s) from the window is used for horizon.

Machine Learning Methods

Windowing:

- In Fig., the window size is 6, step is 1, horizon width is 1, and skip is 0.
- Thus, the series data are now converted into a generic cross-sectional dataset that can be predicted with learning algorithms like regression, neural networks, or support vector machines.
- Once the windowing process is done, then the real power of machine learning algorithms can be brought to bear on a time series dataset.

Machine Learning Methods

Windowing-Model Training:

- Consider the time series dataset shown in Fig. A. The dataset refers to historical monthly profits from a product, from January 2009 to June 2012.
- Suppose the objective in this exercise is to develop profitability forecasts for the next 12 months.
- A linear regression model can be used to fit the crosssectional dataset shown in Fig. B Regression Methods. The model will be:

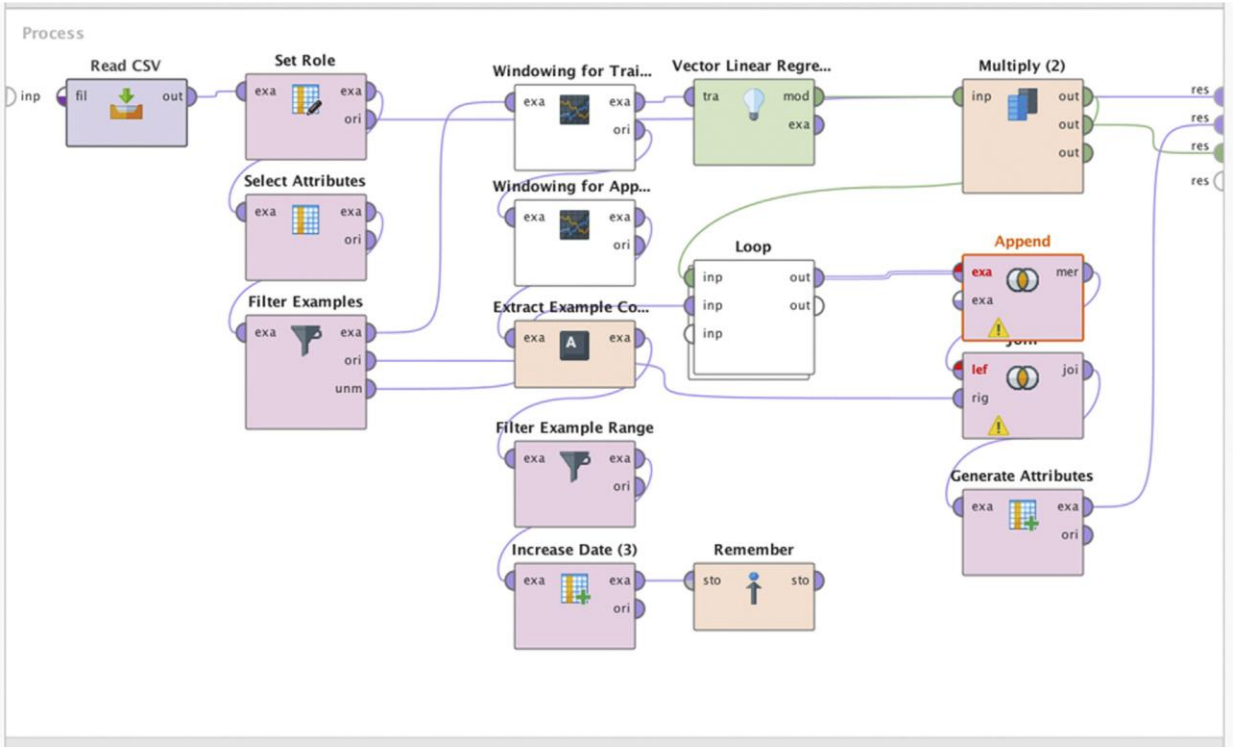
$$\text{input } Y_{t+1}(\text{label}) = 0.493 \times \text{input } Y_{t-5} + 0.258 \times \text{input } Y_{t-4} + 0.107 \times \text{input } Y_{t-3} - 0.098 \times \text{input } Y_{t-2} - 0.073 \times \text{input } Y_{t-1} + 0.329 \times \text{input } Y_{t-0} + 0.135$$

- Training the model is quite straightforward. The inferred relationship between a data point in the time series with the previous six data points is established.
- In other words, if one knows six consecutive data points in a time series, they can use the model to predict the seventh unseen data point.
- Since a new data point has been forecasted, it can be used along with the five preceding data points to predict one more data point and so on. That's time series forecasting one data point at a time!

Machine Learning Methods

Windowing-Implementation:

- Implementing a time series forecasting process using supervised learning is similar to a classification or regression process.
- The distinguishing step in time series forecasting is the conversion of a time series dataset to a crosssectional dataset and stacking the forecast one data point at a time.
- The RapidMiner process is shown in Figure.



Machine Learning Methods

Windowing-Implementation:

- It uses operators from the Time Series extension. Although the process looks complicated, it consists of three functional blocks: (1) conversion to cross-sectional data, (2) training an machine learning model, and (3) forecasting one data point at a time in a loop.
- The dataset used in the process is the Product profit5 dataset shown in Figure. The time series has two attributes: Date and InputYt.

Date	inputYt
Jan 1, 2009	0.709
Feb 1, 2009	1.886
Mar 1, 2009	1.293
Apr 1, 2009	0.822
May 1, 2009	-0.173
Jun 1, 2009	0.552
Jul 1, 2009	1.169
Aug 1, 2009	1.604
Sep 1, 2009	0.949
Oct 1, 2009	0.080
Nov 1, 2009	-0.040
Dec 1, 2009	1.381
Jan 1, 2010	0.761
Feb 1, 2010	2.312
Mar 1, 2010	1.795
Apr 1, 2010	0.586
May 1, 2010	-0.077
Jun 1, 2010	0.613
Jul 1, 2010	1.845

(A) Time series data set

Window id	inputYt + 1 (horizon)	inputYt - 5	inputYt - 4	inputYt - 3	inputYt - 2	inputYt - 1	inputYt - 0
0	1.169	0.709	1.886	1.293	0.822	-0.173	0.552
1	1.604	1.886	1.293	0.822	-0.173	0.552	1.169
2	0.949	1.293	0.822	-0.173	0.552	1.169	1.604
3	0.080	0.822	-0.173	0.552	1.169	1.604	0.949
4	-0.040	-0.173	0.552	1.169	1.604	0.949	0.080
5	1.381	0.552	1.169	1.604	0.949	0.080	-0.040
6	0.761	1.169	1.604	0.949	0.080	-0.040	1.381
7	2.312	1.604	0.949	0.080	-0.040	1.381	0.761
8	1.795	0.949	0.080	-0.040	1.381	0.761	2.312
9	0.586	0.080	-0.040	1.381	0.761	2.312	1.795
10	-0.077	-0.040	1.381	0.761	2.312	1.795	0.586
11	0.613	1.381	0.761	2.312	1.795	0.586	-0.077
12	1.845	0.761	2.312	1.795	0.586	-0.077	0.613
13	1.984	2.312	1.795	0.586	-0.077	0.613	1.845
14	1.861	1.795	0.586	-0.077	0.613	1.845	1.984
15	0.661	0.586	-0.077	0.613	1.845	1.984	1.861

(B) Cross-sectional data set

Machine Learning Methods

Windowing-Implementation:

Step 1: Set Up Windowing

- The process window in implementation figure shows the necessary operators for windowing.
- The time series dataset has a date column, and this must be treated with special care.
- The operator must be informed that one of the columns in the dataset is a date and should be considered as an "id."
- This is accomplished with the Set Role operator. If the input data has multiple time series, Select Attributes operator can be used to select the one to be forecasted.
- In this case, only a one value series is used and strictly speaking this operator is not needed. However, to make the process generic it has been included and the column labeled "inputYt" has been selected.
- Optionally, one may want to use the Filter Examples operator to remove any data points that may have missing values.
- The central operator for this step is the Windowing operator in the Time series extension.

Machine Learning Methods

Windowing-Implementation:

Step 1: Set Up Windowing

- The main parameters for the Windowing operator are:

1. Window size: Determines how many "attributes" are created for the cross-sectional data. Each row of the original time series within the window size will become a new attribute. In this example, w=6 was chosen.

2. Step size: Determines how to advance the window. S=1 was used.

3. Horizon width: Determines how far out to make the forecast. If the window size is 6 and the horizon is 1, then the seventh row of the original time series becomes the first sample for the "label" variable. H=1 was used.

- Figure shows the original data and the transformed output from the windowing process. The window operator adds six new attributes named input Yt-5 through input Yt-0.

Machine Learning Methods

Windowing-Implementation: Step 3: Generate the Forecast in a Loop

- Once the model fitting is done, the next step is to start the forecasting process.
- Note that given this configuration of the window size and horizon, one can now only make the forecast for the next step.
- In the example, the last row of the transformed dataset corresponds to December 2011.
- The independent variables are values from June-November 2011 and the target or label variable is December 2011. The regression equation is be used to predict December 2011 value.

Machine Learning Methods

Windowing-Implementation: Step 3: Generate the Forecast in a Loop

- The same regression equation is also used for predicting January 2012 value.
- All one needs to do is insert the values from July-December into the regression equation to generate the January 2012 forecast.
- Next, a new row of data needs to be generated that would run from August-January to predict February 2012 using the regression equation.
- All the (actual) data from August-December is available as well as the predicted value for January.
- Once the predicted February value is obtained, there is nothing preventing the actual data from September-December plus the predicted January and February values from being used to forecast March 2012.

Machine Learning Methods

Windowing-Implementation: Step 3: Generate the Forecast in a Loop

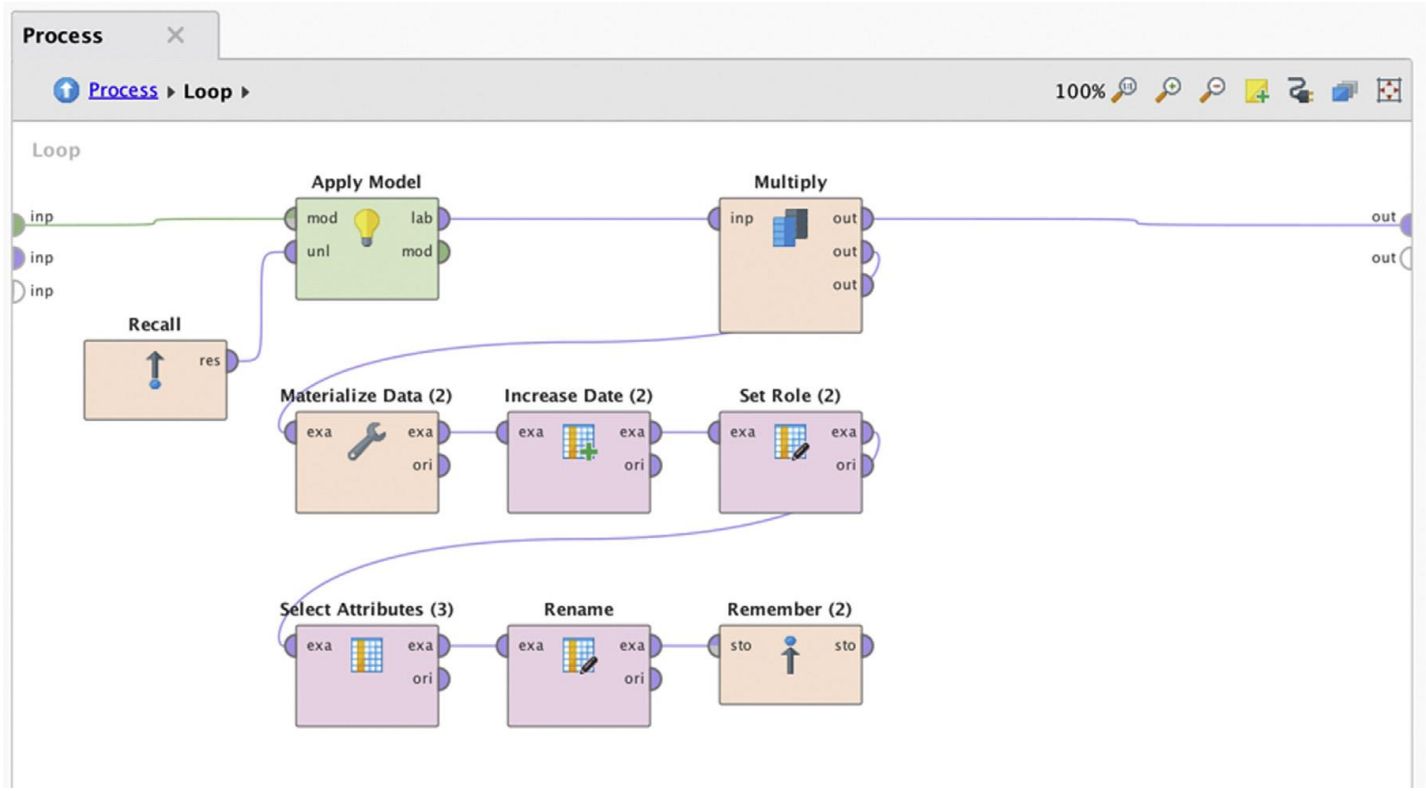
- To implement this in RapidMiner process, one would need to break this up into two separate parts.
- First, take the last forecasted row (in this case, December 2011), drop the current value of input Yt-5 (current value is 1.201), rename input Yt-4 to input Yt-5, rename input Yt-3 to input Yt-4, rename input Yt-2 to input Yt-3, rename input Yt-1 to input Yt-2, rename input Yt-0 to input Yt-1, and finally rename predicted label (current value is 1.934) to input Yt-0.
- With this new row of data, the regression model can be applied to predict the next date in the series: January 2012.

Date	prediction(l...	inputYt-5	inputYt-4	inputYt-3	inputYt-2	inputYt-1	inputYt-0
Dec 1, 2011...	1.694	1.201	2.466	2.497	2.245	1.179	1.119
Jan 1, 2012 ...	2.597	2.466	2.497	2.245	1.179	1.119	1.694
Feb 1, 2012...	2.693	2.497	2.245	1.179	1.119	1.694	2.597
Mar 1, 2012...	2.196	2.245	1.179	1.119	1.694	2.597	2.693
Apr 1, 2012...	1.457	1.179	1.119	1.694	2.597	2.693	2.196
May 1, 201...	1.457	1.119	1.694	2.597	2.693	2.196	1.457
Jun 1, 2012 ...	2.087	1.694	2.597	2.693	2.196	1.457	1.457
Jul 1, 2012 ...	2.784	2.597	2.693	2.196	1.457	1.457	2.087
Aug 1, 2012...	2.807	2.693	2.196	1.457	1.457	2.087	2.784
Sep 1, 2012...	2.265	2.196	1.457	1.457	2.087	2.784	2.807
Oct 1, 2012...	1.720	1.457	1.457	2.087	2.784	2.807	2.265
Nov 1, 2012...	1.816	1.457	2.087	2.784	2.807	2.265	1.720
Dec 1, 2012...	2.433	2.087	2.784	2.807	2.265	1.720	1.816
Jan 1, 2013 ...	2.974	2.784	2.807	2.265	1.720	1.816	2.433
Feb 1, 2013...	2.911	2.807	2.265	1.720	1.816	2.433	2.974

Machine Learning Methods

Windowing-Implementation: Step 3: Generate the Forecast in a Loop

- Last figure shows the sample steps. Next, this entire process need to be put inside a Loop operator that will allow these steps to be repeatedly run for as many future periods as needed.
- The Loop operator will contain all the mechanisms for accomplishing the renaming and, of course, to perform the forecasting.



Machine Learning Methods

Windowing-Implementation: Step 3: Generate the Forecast in a Loop

- Set the iterations in the Loop operator to the number of future months to forecast. (horizon).
- In this case, this is defined by a process variable called futureMonths whose value can be changed by the user before process execution.
- It is also possible to capture the Loop counts in a macro if the set iteration macro box is checked.
- A macro in RapidMiner is nothing but a process variable that can be called by other operators in the process.
- When set iteration macro is checked and a name is provided in the macro name box, a variable will be created with that name whose value will be updated each time, one loop is completed.
- An initial value for this macro is set by the macro start value option.
- Loops may be terminated by specifying a timeout, which is enabled by checking the limit time box.
- A macro variable can be used by any other operator by using the format %{macro name} in place of a numeric value.

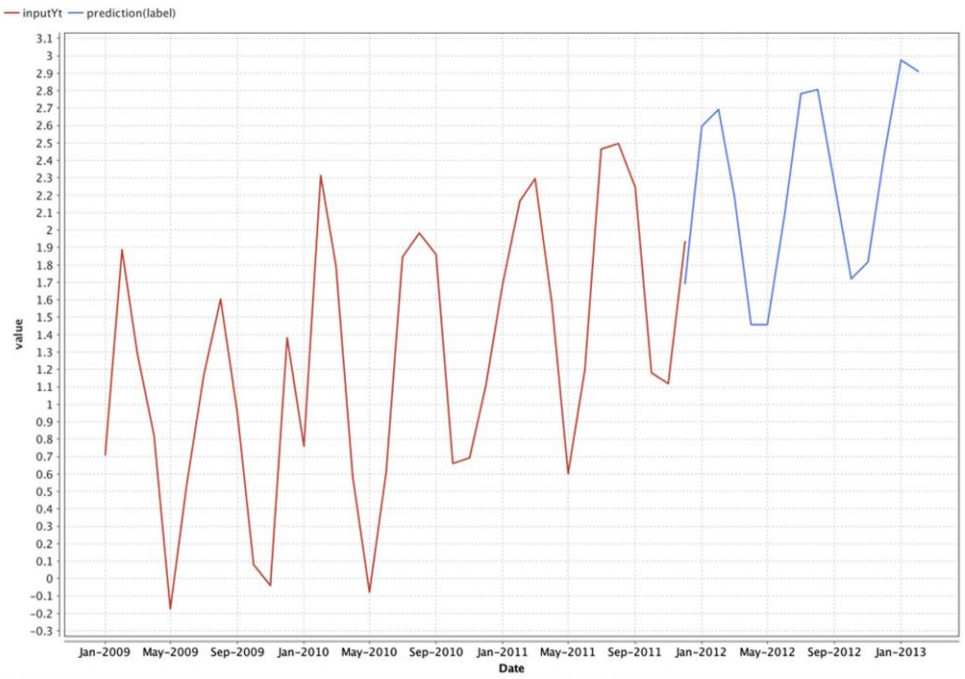
Machine Learning Methods

Windowing-Implementation: Step 3: Generate the Forecast in a Loop

- Before the looping is started, the last forecasted row needs to be stored in a separate data structure. This is accomplished by a new Windowing operator and the macro titled Extract Example Set.
- The Filter Example operator simply deletes all rows of the transformed dataset except the last forecasted row.
- Finally the Remember operator stores this in memory and allows one to "recall" the stored value once inside the loop.
- The loop parameter iterations will determine the number of times the inner process is repeated.
- Figure(Rapid Miner) shows that during each iteration, the model is applied on the last forecasted row, and bookkeeping operations are performed to prepare application of the model to forecast the next month.
- This includes incrementing the month (date) by one, changing the role of the predicted label to that of a regular attribute, and finally renaming all the attributes.
- The newly renamed data are stored and then recalled before the next iteration begins.

Machine Learning Methods

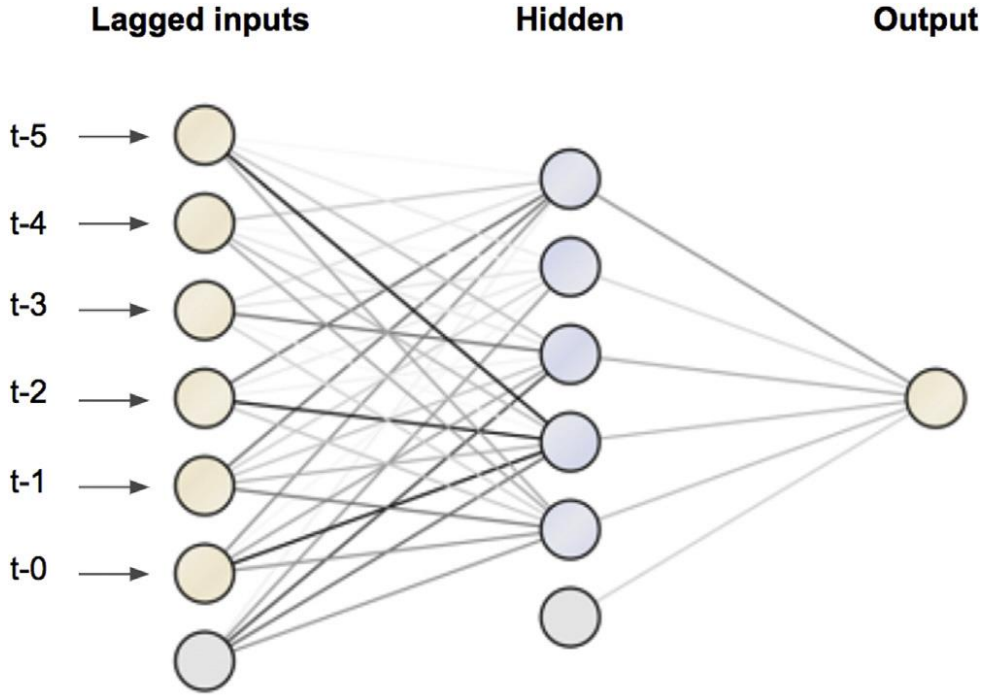
Windowing-Implementation: Step 3: Generate the Forecast in a Loop



- The output of this process is shown in Fig. as an overlay on top of the actual data.
- As seen, the simple linear regression model seems to adequately capture both the trend and seasonality of the underlying data.
- The Linear Regression operator of Step 2: Train the model can be quickly swapped to a Support Vector Machine operator and its performance tested without having to do any other programming or process modifications.

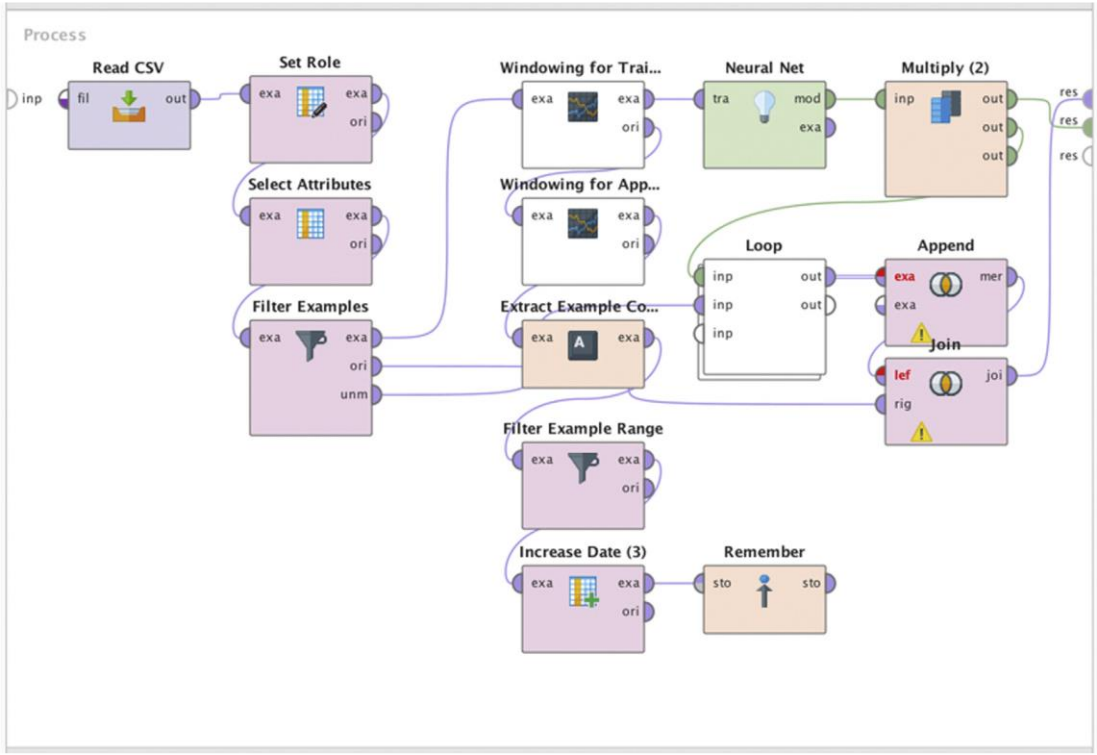
Machine Learning Methods -Neural Network Autoregressive

- The windowing process allows time series dataset to be transformed to a cross-sectional horizontal dataset that is conducive to learners to create a forecasting model.
- That includes artificial neural networks to forecast the time series.
- Consider a simple ANN architecture with six inputs, one hidden layer with five nodes and an output, as shown in Figure.
- It is multilayer feed forward network that maps nonlinear functions.



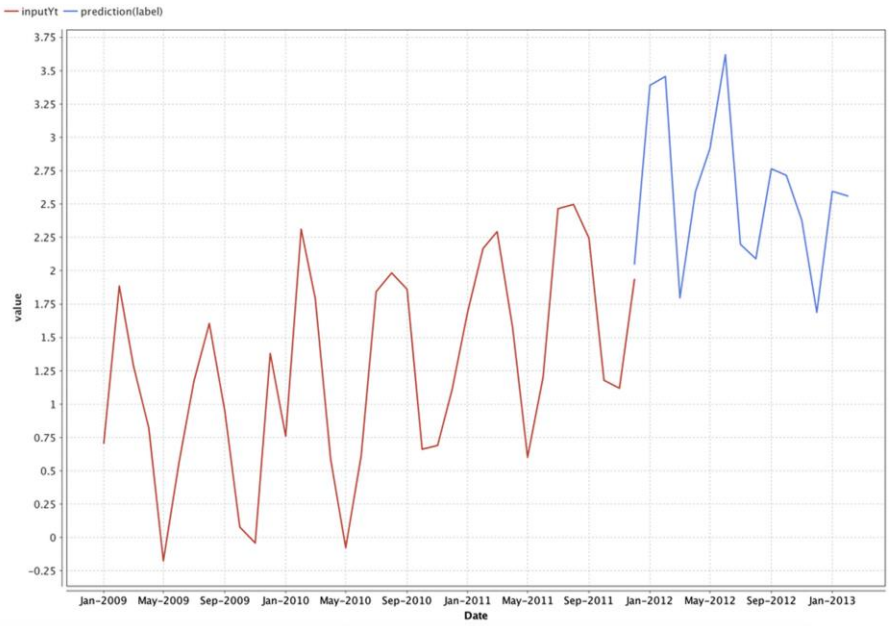
Machine Learning Methods -Neural Network Autoregressive

- For the NNAR model, the number of hidden layers is 1 and the number of nodes is one less than the number of the input nodes.
- Figure shows the Rapidminer process for NNAR. This process has six lagged inputs and the number of nodes in the hidden layer is 5.
- The forecasting is done one step ahead. The Loop operator takes the latest output and applies it to the input for the next iteration. The process can be saved and executed.



Machine Learning Methods -Neural Network Autoregressive

- The time series forecast is shown in Figure. It can be observed that the NNAR forecast is different from the forecast achieved with the linear regression model.
- An important point about any time series forecasting is that one should not place too much emphasis on the point forecasts. A complex quantity like sales demand for a product is influenced by too many factors and to claim that any forecasting will predict the exact value of demand three months in advance is unrealistic. However, what is far more valuable is the fact that recent undulations in the demand can be effectively captured and predicted.

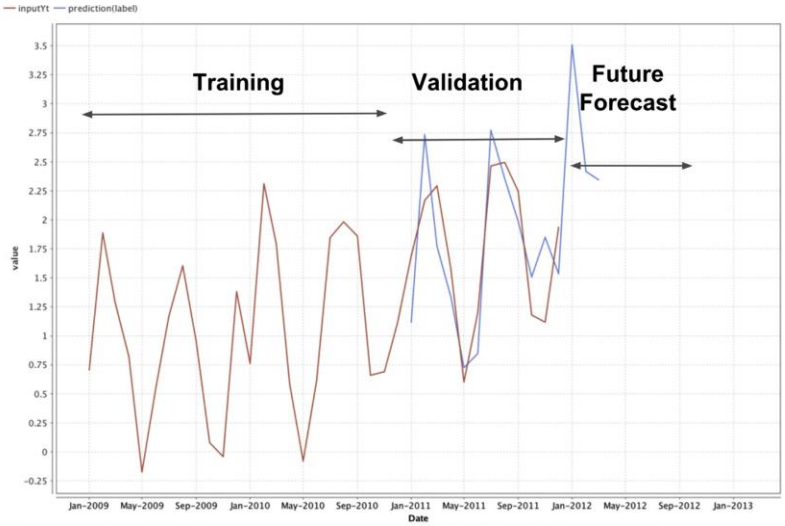


Performance Evaluation- Validation Dataset

- One way to measure accuracy is to measure the actual value post-fact and compare against the forecast and calculate the error.
- However, one would have to wait for the time to pass and the actual data to be measured.
- Residues are calculated when a forecast model is fitted on training data.
- By all means, the model might have overfitted the training data and perform poorly with unseen future forecasts.
- For this reason, training residuals are not a good way to measure how the forecast models are going to perform in the real world.
- Recall that in the case of the supervised learners, a set of known data is reserved for model validation.
- Similarly, some data points can be reserved in the time series as a validation dataset just to test the accuracy of the model.

Performance Evaluation- Validation Dataset

- The training process uses data by restricting it to a point and the rest of the later time series is used for validation as shown in Figure.
- As the model has not seen the validation dataset, deviation of actuals from the forecast is the forecast error of the model.
- The forecast error is the difference between the actual value yi and the forecasted value yi^



- The error or the residue for the ith data point is given by $e_i = y_i - \hat{y}_i$
- The forecast error shown in Eq. is scale dependent.
- The error measured for each of the data points and can be aggregated to one metric to indicate the error of the forecasting model.

Performance Evaluation- Forecast accuracy aggregate metrics

Mean Absolute Error

- The error of the individual data point may be positive or negative and may cancel each other out.
- To derive the overall forecast for the model, calculate the absolute error to aggregate all the residuals and average it.

$$\text{Mean absolute error} = \text{mean}(|e_i|)$$

- MAE is a simple metric and it is scale dependent.
- It is convenient to communicate the error of the revenue forecasting model as, for example, $900,000 per day.

Performance Evaluation- Forecast accuracy aggregate metrics

Root Mean Squared Error

- In some cases it is advantageous to penalize the individual point error with higher residues.
- Even though two models have the same MAE, one might have consistent error and the other might have low errors for some points and high error for other points. RMSE penalizes the latter.

$$\text{Root mean squared error} = \sqrt{\text{mean}\,(e^2)}$$

- RMSE is scale dependent and is used in situations where penalizing high relative residue is necessary.
- On the other hand, it is slightly difficult to understand the RMSE for a stand-alone model.

Performance Evaluation- Forecast accuracy aggregate metrics

Mean Absolute Percentage Error

- Percentage error of a data point is $p_i = 100 \cdot \frac{e_i}{y_i}$.
- It is a scale independent error that can be aggregated to form mean absolute percentage error. $\text{Mean absolute percentage error} = \text{mean}(|p_i|)$
- MAPE is useful to compare against multiple models across the different forecasting applications.
- For example, the quarterly revenue forecast, measured in USD, for a car brand might be 65% and the forecast for world-wide car demand, measured in quantity, might be 63%.
- The firm's ability to forecast the car demand is higher than the revenue forecast for one brand.
- Eventhough MAPE is easy to understand and scale independent, MAPE has a significant limitation when it applies to intermittent data where zero values are possible in actual time series.
- For example, profit or defects in a product. Zero value in the time series yields an infinite error rate (if the forecast is nonzero) and skews the result.
- MAPE is also meaningless when the zero point is not defined or arbitrarily defined, as in non-kelvin temperature scales.

Performance Evaluation- Forecast accuracy aggregate metrics

Mean Absolute Scaled Error

- MASE is scale independent and overcomes the key limitations of MAPE by comparing the forecast values against a naive forecast.
- Naive forecast is a simple forecast where the next data point has the same value as the previous data point.
- Scaled error is defined as:

$$\text{MASE} = \frac{\sum_{i=1}^{T} |e|}{\frac{T}{T-1}\sum_{i=2}^{T} |\bar{y}_i - \bar{y}_{i-1}|}$$

- T is the total number of data points.
- Scaled error is less than one if the forecast is better than naive forecast and greater than one if it is worse than naïve forecast.
- One would want a scaled error much less than one for a good forecasting model.

Performance Evaluation-Sliding Window Validation

- Sliding window validation is a process of backtesting time series models built through machine learning based methods.
- The whole cross-sectional dataset is divided into different training windows by specifying the window width.
- A model is trained using a training window and applied on the testing window to compute the performance for the first run.
- For the next run, the training window is slided to new set of training records and the process is repeated until all the training windows are used.
- By this technique, an average performance metric can be calculated across the entire dataset.
- The performance metric derived through sliding window validation is generally more robust than split validation technique.

Anomaly Detection

Anomaly Detection

- Anomaly detection is the process of finding outliers in a given dataset.
- Outliers are the data objects that stand out amongst other data objects and do not conform to the expected behavior in a dataset.
- Anomaly detection algorithms have broad applications in business, scientific, and security domains where isolating and acting on the results of outlier detection is critical.
- For identification of anomalies, algorithms such as classification, regression, and clustering can be used.
- If the training dataset has objects with known anomalous outcomes, then any of the supervised data science algorithms can be used for anomaly detection.

Anomaly Detection

- In addition to supervised algorithms, there are specialized (unsupervised) algorithms whose whole purpose is to detect outliers without the use of a labelled training dataset.
- In the context of unsupervised anomaly detection, algorithms can either measure distance from other data points or density around the neighborhood of the data point.
- Even clustering techniques can be leveraged for anomaly detection.
- The outlier usually forms a separate cluster from other clusters because they are far away from other data points.

Anomaly Detection-Concepts

- An outlier is a data object that is markedly different from the other objects in a dataset.
- Hence, an outlier is always defined in the context of other objects in the dataset.
- A high-income individual may be an outlier in a middle-class neighborhood dataset, but not in the membership of a luxury vehicle ownership dataset.
- By nature of the occurrence, outliers are also rare and, hence, they stand out amongst other data points.
- For example, the majority of computer network traffic is legitimate, and the one malicious network attack would be the outlier.

Concepts-Causes of Outliers

- Outliers in the dataset can originate from either error in the data or from valid inherent variability in the data.
- It is important to understand the provenance of the outliers because it will guide what action, if any, should be performed on the identified outliers.
- However, pinpointing exactly what caused an outlier is a tedious task and it may be impossible to find the causes of outliers in the dataset.

Concepts-Causes of Outliers

Data errors:

- Outliers may be part of the dataset because of measurement errors, human errors, or data collection errors.
- For example, in a dataset of human heights, a reading such as 1.70 cm is obviously an error and most likely was wrongly entered into the system.
- These data points are often ignored because they affect the conclusion of the data science task.
- Outlier detection here is used as a preprocessing step in algorithms such as regression and neural networks.
- Data errors due to human mistake could be either intentional introduction of error or unintentional error due to data entry error or significant bias.

Concepts-Causes of Outliers

Normal variance in the data:

- In a normal distribution, 99.7% of data points lie within three standard deviations from the mean.
- In other words, 0.26% or 1 in 370 data points lie outside of three standard deviations from the mean.
- By definition, they don't occur frequently and yet are a part of legitimate data.
- An individual earning a billion dollars in a year or someone who is more than 7 ft tall falls under the category of outlier in an income dataset or a human height dataset respectively.
- These outliers skew some of the descriptive statistics like the mean of the dataset.
- Regardless, they are legitimate data points in the dataset.

Concepts-Causes of Outliers

Data from other distribution classes:

- The number of daily page views for a customer-facing website from a user IP address usually range from one to several dozen.
- However, it is not unusual to find a few IP addresses reaching hundreds of thousands of page views in a day.
- This outlier could be an automated program from a computer (also called a bot) making the calls to scrape the content of the site or access one of the utilities of the site, either legitimately or maliciously.
- Even though they are an outlier, it is quite "normal" for bots to register thousands of page views to a website.
- All bot traffic falls under the distribution of a different class "traffic from programs" other than traffic from regular browsers that fall under the human user class.

Concepts-Causes of Outliers

Distributional assumptions

- Outlier data points can originate from incorrect assumptions made on the data or distribution.
- For example, if the data measured is usage of a library in a school, then during term exams there will be an outlier because of a surge in the usage of the library.
- Similarly, there will be a surge in retail sales during the day after Thanksgiving in the United States.
- An outlier in this case is expected and does not represent the data point of a typical measure.

Concepts-Causes of Outliers

- Understanding why outliers occur will help determine what action to perform after outlier detection.
- In a few applications, the objective is to isolate and act on the outlier as can be seen in credit card transaction fraud monitoring.
- In this case, credit card transactions exhibiting different behavior from most normal transactions (such as high frequency, high amounts, or very large geographic separation between points of consecutive transactions) has to be isolated, alerted, and the owner of the credit card has to be contacted immediately to verify the authenticity of the transaction.
- In other cases, the outliers have to be filtered out because they may skew the final outcome.
- Here outlier detection is used as a preprocessing technique for other data science or analytical tasks.
- For example, ultra-high-income earners might need to be eliminated in order to generalize a country's income patterns.
- Here outliers are legitimate data points but are intentionally disregarded in order to generalize conclusions.

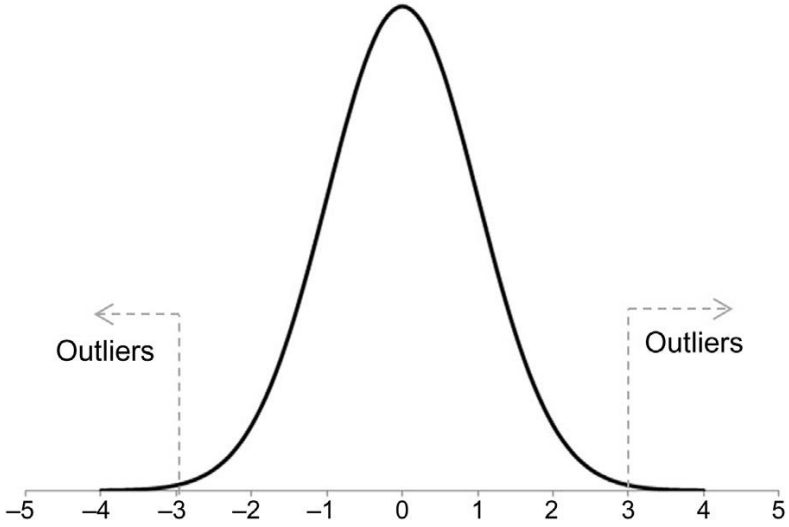
Concepts-Anomaly Detection Techniques

- Humans are innately equipped to focus on outliers.
- The news cycle experienced every day is mainly hinged on outlier events.
- The interest around knowing who is the fastest, who earns the most, and who wins the most medals or scores the most goals is in part due to increased attention to outliers.
- If the data is in one dimension like taxable income for individuals, outliers can be identified with a simple sorting function.
- Visualizing data by scatter, histogram, and box-whisker charts can help to identify outliers in the case of single attribute datasets as well.
- More advanced techniques would fit the data to a distribution model and use data science techniques to detect outliers.

Concepts-Anomaly Detection Techniques

Outlier Detection Using Statistical Methods

- Outliers in the data can be identified by creating a statistical distribution model of the data and identifying the data points that do not fit into the model or data points that occupy the ends of the distribution tails.
- The underlying distribution of many practical datasets fall into the Gaussian (normal) distribution.
- The parameters for building a normal distribution (i.e., mean and standard deviation) can be estimated from the dataset and a normal distribution curve can be created like the one shown in Figure.



Concepts-Anomaly Detection Techniques

Outlier Detection Using Statistical Methods

- Outliers can be detected based on where the data points fall in the standard normal distribution curve.
- A threshold for classifying an outlier can be specified, say, three standard deviations from the mean.
- Any data point that is more than three standard deviations is identified as an outlier.
- Identifying outliers using this method considers only one attribute or dimension at a time.
- More advanced statistical techniques take multiple dimensions into account and calculate the Mahalanobis distance instead of the standard deviations from the mean in a univariate distribution.
- Mahalanobis distance is the multivariate generalization of finding how many standard deviations away a point is from the mean of the multivariate distribution.

Concepts-Anomaly Detection Techniques

Outlier Detection Using Statistical Methods-limitations

- Outlier detection using statistics provides a simple framework for building a distribution model and for detection based on the variance of the data point from the mean.
- One limitation of using the distribution model to find outliers is that the distribution of the dataset is not previously known.
- Even if the distribution is known, the actual data don't always fit the model.

Concepts-Anomaly Detection Techniques

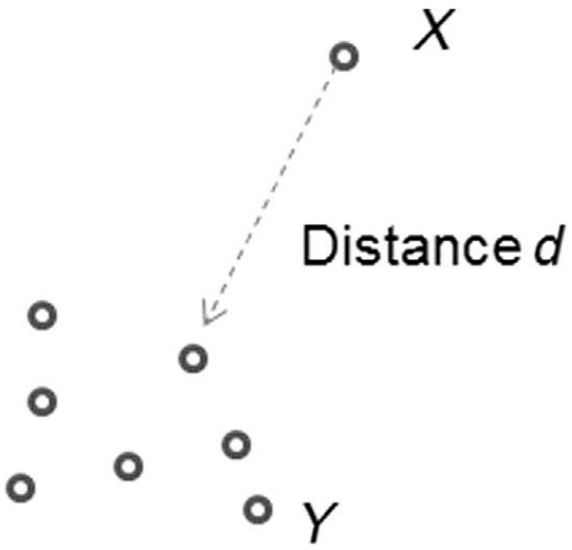
Outlier Detection Using Data Science

- Outliers exhibit a certain set of characteristics that can be exploited to find them.
- Following are classes of techniques that were developed to identify outliers by using their unique characteristics.
- Each of these techniques has multiple parameters and, hence, a data point labeled as an outlier in one algorithm may not be an outlier to another.
- Hence, it is prudent to rely on multiple algorithms before labelling the outliers.

Concepts-Anomaly Detection Techniques

Outlier Detection Using Data Science

- **Distance-based**: By nature, outliers are different from other data objects in the dataset.
- In multidimensional Cartesian space they are distant from other data points, as shown in Figure.



- If the average distance of the nearest N neighbors is measured, the outliers will have a higher value than other normal data points.
- Distance-based algorithms utilize this property to identify outliers in the data.

Concepts-Anomaly Detection Techniques

Outlier Detection Using Data Science

- **Density-based:** The density of a data point in a neighborhood is inversely related to the distance to its neighbors.
- Outliers occupy low-density areas while the regular data points congregate in high-density areas.
- This is derived from the fact that the relative occurrence of an outlier is low compared with the frequency of normal data points.
- **Distribution-based:** Outliers are the data points that have a low probability of occurrence and they occupy the tail ends of the distribution curve.
- So, if one tries to fit the dataset in a statistical distribution, these anomalous data points will stand out and, hence, can be identified.
- A simple normal distribution can be used to model the dataset by calculating the mean and standard deviation.

Concepts-Anomaly Detection Techniques

Outlier Detection Using Data Science

- **Clustering:** Outliers by definition are not similar to normal data points in a dataset.
- They are rare data points far away from regular data points and generally do not form a tight cluster.
- Since most of the clustering algorithms have a minimum threshold of data points to form a cluster, the outliers are the lone data points that are not clustered.
- Even if the outliers form a cluster, they are far away from other clusters.

Concepts-Anomaly Detection Techniques

Outlier Detection Using Data Science

- **Classification techniques:** Nearly all classification techniques can be used to identify outliers, if previously known classified data are available.
- In classification techniques for detecting outliers, a known test dataset is needed where one of the class labels should be called "Outlier."
- The outlier-detection classification model that is built based on the test dataset can predict whether the unknown data is an outlier or not.
- The challenge in using a classification model is the availability of previously labeled data.
- Outlier data may be difficult to source because they are rare.
- This can be partially solved by stratified sampling where the outlier records are oversampled against normal records.

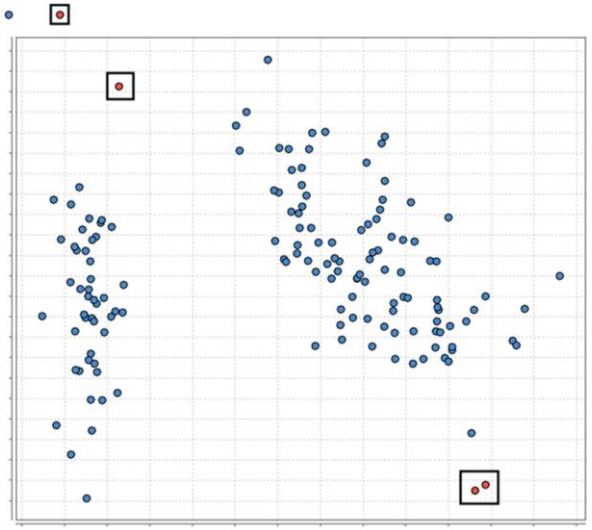
Concepts-Anomaly Detection Techniques

Outlier Detection Using Data Science

- **Classification techniques:** Nearly all classification techniques can be used to identify outliers, if previously known classified data are available.
- In classification techniques for detecting outliers, a known test dataset is needed where one of the class labels should be called "Outlier."
- The outlier-detection classification model that is built based on the test dataset can predict whether the unknown data is an outlier or not.
- The challenge in using a classification model is the availability of previously labeled data.
- Outlier data may be difficult to source because they are rare.
- This can be partially solved by stratified sampling where the outlier records are oversampled against normal records.

Distance-based Outlier Detection

- Distance or proximity-based outlier detection is one of the most fundamental algorithms for anomaly detection and it relies on the fact that outliers are distant from other data points.
- The proximity measures can be simple Euclidean distance for real values and cosine or Jaccard similarity measures for binary and categorical values.
- For the purpose of this discussion, consider a dataset with numeric attributes and Euclidean distance as the proximity measure.
- Figure shows a two-dimensional scatterplot of a sample dataset. Outliers are the data points marked as gray and can be visually identified away from the groups of data.
- However, when working with multidimensional data with more attributes, visual techniques show limitations quickly.

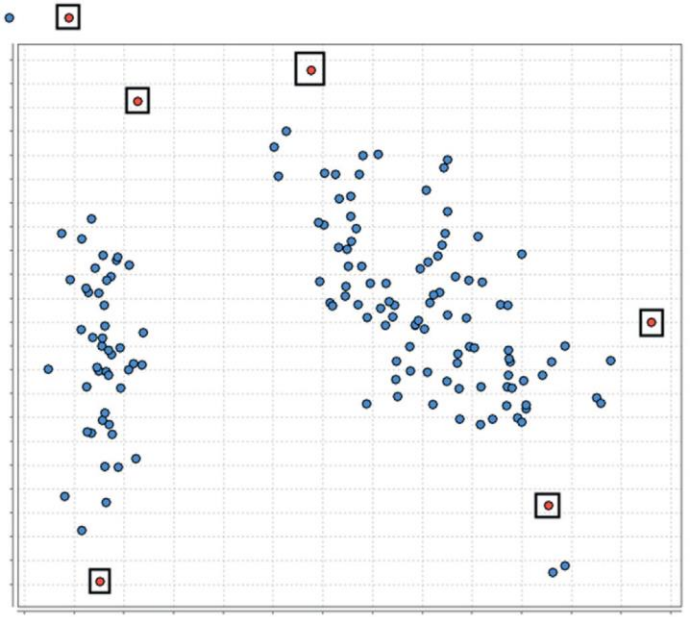


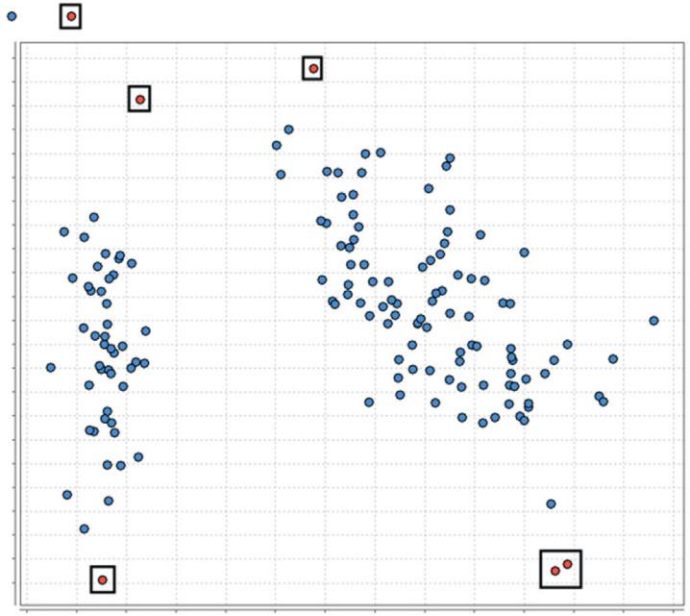
Distance-based Outlier Detection

- How it works:
- In distance-based outlier detection, there is a significant effect based on the value of k, as in the k-NN classification technique.
- If the value of k=1, then two outliers next to each other but far away from other data points are not identified as outliers.
- On the other hand, if the value of k is large, then a group of normal data points which form a cohesive cluster will be mislabeled as outliers, if the number of data points is less than k and the cluster is far away from other data points.
- With a defined value of k, once the distance scores have been calculated, a distance threshold can be specified to identify outliers or pick the top n objects with maximum distances, depending on the application and the nature of the dataset.

Distance-based Outlier Detection

- How it works:
- Figure shows the results of two different outlier-detection algorithms based on distance for the Iris dataset.
- Fig. A shows the outlier detection with k=1 and Fig. B shows the detection of the same dataset with k=5.





Distance-based Outlier Detection

- **How to implement:**
- **Commercial data science tools offer specific outlier-detection algorithms and solutions as part of the package either in the modeling or data cleansing sections.**
- **In RapidMiner, unsupervised outlier-detection operator can be found in Data Transformation >Data Cleansing > Outlier Detection > Detect Outlier Distance.**
- **The example set used in this process is the Iris dataset with four numerical attributes and 150 examples.**

Distance-based Outlier Detection

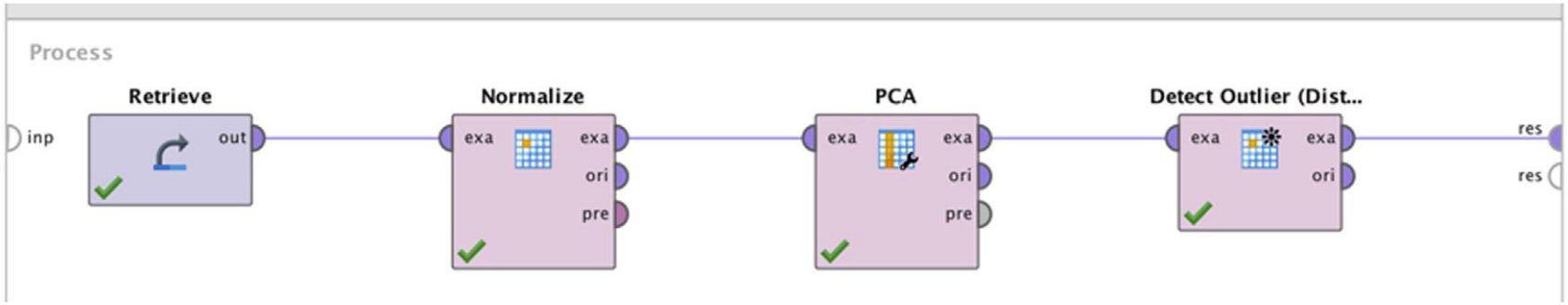
- How to implement: Step 1: Data Preparation
- Even though all four attributes of the Iris dataset measure the same quantity (length) and are measured on the same scale (centimeters), a normalization step is included as a matter of best practice for techniques that involve distance calculation.
- The Normalize operator can be found in Data Transformation > Value modification > Numerical.
- The attributes are converted to a uniform scale of mean 0 and standard deviation 1 using Ztransformation.
- For the purposes of this demonstration, a two-dimensional scatterplot with two attributes will be helpful to visualize outliers. However, the Iris dataset has four attributes.
- To aid in this visualization objective, the four numerical attributes will be reduced to two attributes (principal components) using the principal component analysis (PCA) operator.

Distance-based Outlier Detection

- How to implement: Step 2: Detect Outlier Operator
- The Detect Outlier (Distances) operator has a data input port and outputs data with an appended attribute called outlier.
- The value of the output outlier attribute is either true or false. The Detect Outlier (Distances) operator has three parameters that can be configured by the user.
- **Number of neighbors:** This is the value of k in the algorithm. The default value is 10. If the value is made lower, the process finds smaller outlier clusters with less data points.
- **Number of outliers:** The individual outlier score is not visible to the users. Instead the algorithm finds the data points with the highest outlier scores. The number of data points to be found can be configured using this parameter.
- **Distance function:** As in the k-NN algorithm, the distance measurement function needs to be specified. Commonly used functions are Euclidean and cosine (for document vectors).

Distance-based Outlier Detection

- How to implement: Step 2: Detect Outlier Operator
- In this example, k=1, number of outliers=10, and the distance function is set to Euclidian.
- The output of this operator is the example set with an appended outlier attribute.
- Figure provides the RapidMiner process with data extraction, PCA dimensional reduction, and outlier-detection operators.
- The process can now be saved and executed.

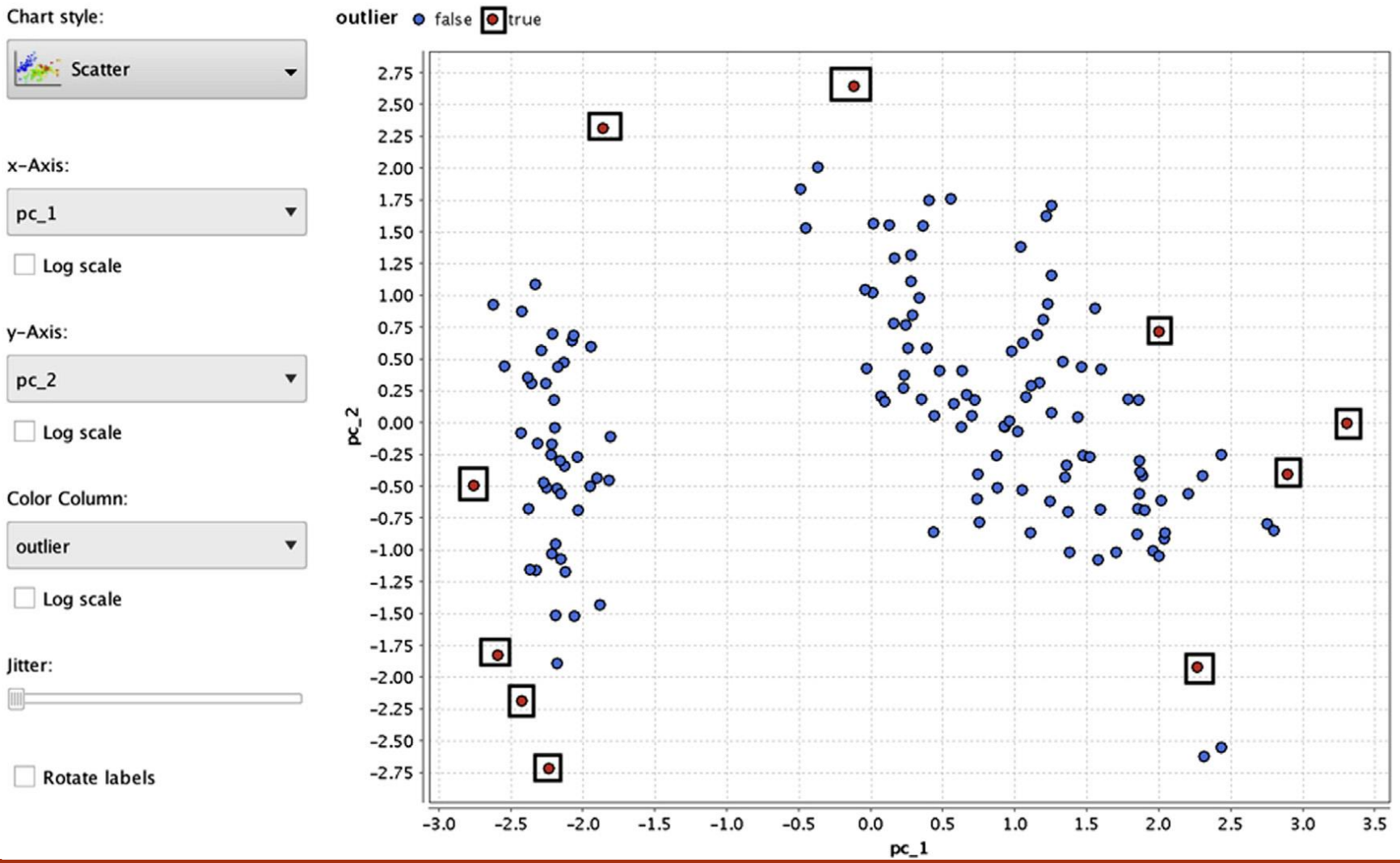


Distance-based Outlier Detection

- How to implement: Step 3: Execution and Interpretation
- The result dataset can be sorted by outlier attribute, which has either a true or false value.
- Since 10 outliers have been specified in the parameter of the Detect outlier operator, that number of outliers can be found in the result set.
- An efficient way of exploring the outliers is to look at the scatterplot in the Chart view of results set.
- The X- and Y-axes can be specified as the principal components and the color as the outlier attribute.
- Distance-based outlier detection is a simple algorithm that is easy to implement and widely used when the problem involves many numeric variables.
- The execution becomes expensive when the dataset involves a high number of attributes and records, because the algorithm has to calculate distances with other data points in high-dimensional space.

Distance-based Outlier Detection

- How to implement: Step 3: Execution and Interpretation
- The output scatterplot shows the outlier data points along with all the normal data points as shown in Figure.

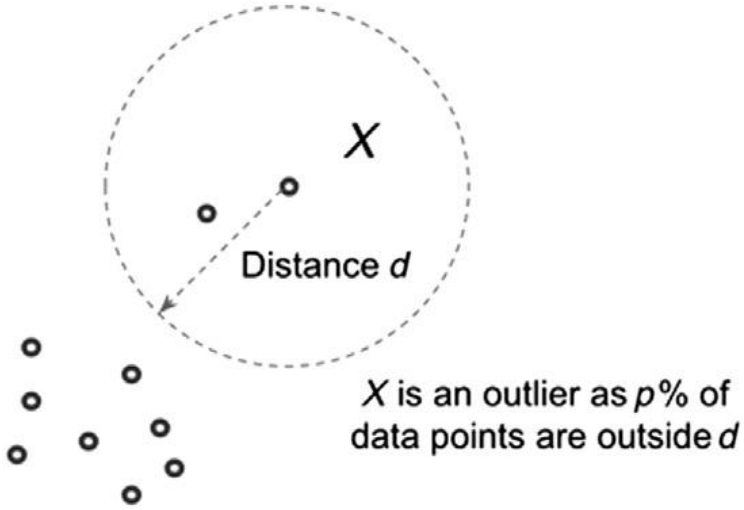


Density-based Outlier Detection

- Outliers, by definition, occur less frequently compared to normal data points.
- This means that in the data space outliers occupy low-density areas and normal data points occupy high-density areas.
- Density is a count of data points in a normalized unit of space and is inversely proportional to the distances between data points.
- The objective of a density-based outlier algorithm is to identify those data points from low-density areas.
- There are a few different implementations to assign an outlier score for the data points.
- The inverse of the average distance of all k neighbors can be found. The distance between data points and density are inversely proportional.
- Neighborhood density can also be calculated by calculating the number of data points from a normalized unit distance.
- The approach for density-based outliers is similar to the approach discussed for density-based clustering and for the k-NN classification algorithm.

Density-based Outlier Detection

- Since distance is the inverse of density, the approach of a density-based outlier can be explained with two parameters, distance (d) and proportion of data points (p).
- A point X is considered an outlier if at least p fraction of points lie more than d distance from the point.
- Figure provides a visual illustration of outlier detection. By the given definition, the point X occupies a low-density area.



Density-based Outlier Detection

- By the given definition, the point X occupies a low-density area.
- The parameter p is specified as a high value, above 95%. One of the key issues in this implementation is specifying distance.
- It is important to normalize the attributes so that the distance makes sense, particularly when attributes involve different measures and units.
- If the distance is specified too low, then more outliers will be detected, which means normal points have the risk of being labeled as outliers and vice versa.

Density-based Outlier Detection

How to Implement

- The RapidMiner process for outlier detection based on density is similar to outlier detection by distance, which was reviewed in the previous section.
- The process developed for previous distance-based outliers can be used, but the Detect Outlier (Distances) operator would be replaced with the Detect Outlier (Densities) operator.

Step 1: Data Preparation

- Data preparation will condition the data so the Detect Outlier (Densities) operator returns meaningful results.
- As with the outlier detection by distance technique, the Iris dataset will be used with normalization and the PCA operator so that the number of attributes is reduced to two for easy visualization.

Density-based Outlier Detection

How to Implement

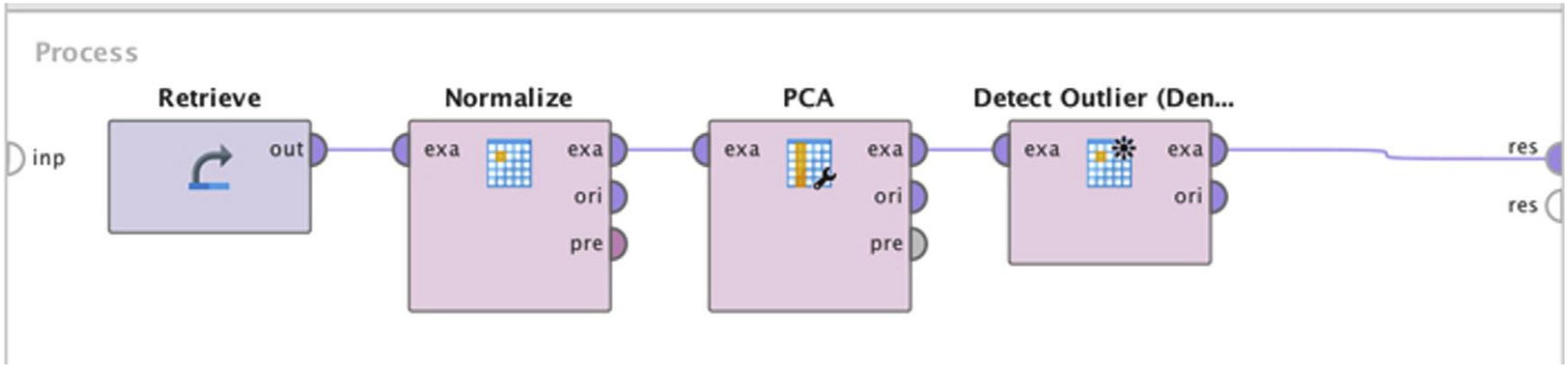
Step 2: Detect Outlier Operator

- The Detect Outlier (Densities) operator can be found in Data Transformation > Data Cleansing > Outlier Detection, and has three parameters:
- Distance (d): Threshold distance used to find outliers. For this example, the distance is specified as 1.
- Proportion (p): Proportion of data points outside of radius d of a point, beyond which the point is considered an outlier. For this example, the value specified is 95%.
- Distance measurement: A measurement parameter like Euclidean, cosine, or squared distance. The default value is Euclidean.
- Any data point that has more than 95% of other data points beyond distance d is considered an outlier.

Density-based Outlier Detection

How to Implement

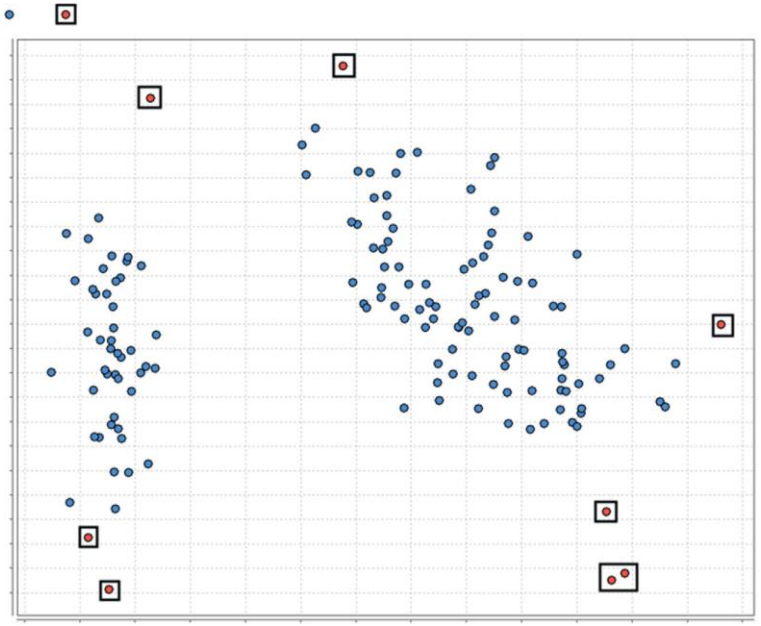
- Figure shows the RapidMiner process with the Normalization, PCA, and Detect Outlier operators.
- The process can be saved and executed.



Density-based Outlier Detection

How to Implement- Step 3: Execution and Interpretation

- The process adds an outlier attribute to the example set, which can be used for visualization using a scatterplot as shown in Figure.
- The outlier attribute is Boolean and indicates whether the data point is predicted to be an outlier or not.
- In the scatterplot, a few data points marked as outliers can be found.
- The parameters d and p of the Detect Outlier operator can be tuned to find the desired level of outlier detection.



Density-based Outlier Detection

How to Implement- Step 3: Execution and Interpretation

- Density-based outlier detection is closely related to distance-based outlier approaches and, hence, the same pros and cons apply.
- As with distancebased outlier detection, the main drawback is that this approach does not work with varying densities.
- The next approach, local outlier factor (LOF) is designed for such datasets.
- Specifying the parameter distance (d) and proportion (p) is going to be challenging, particularly when the characteristics of the data are not previously known.

Local Outlier Factor

- The LOF technique is a variation of density-based outlier detection, and addresses one of its key limitations, detecting the outliers in varying density.
- Varying density is a problem in simple density-based methods, including DBSCAN clustering.
- How it Works
- LOF takes into account the density of the data point and the density of the neighborhood of the data point as well.
- A key feature of the LOF technique is that the outlier score takes into account the relative density of the data point.
- Once the outlier scores for data points are calculated, the data points can be sorted to find the outliers in the dataset.

Local Outlier Factor

- **How it Works**
- The core of the LOF lies in the calculation of the relative density.
- The relative density of a data point X with k neighbors is given by the following equation:

$$\text{Relative density of } X = \frac{\text{Density of } X}{\text{Average density of all data points in the neighborhood}}$$

where the density of X is the inverse of the average distance for the nearest k data points.

- The same parameter k also forms the locality of the neighborhood.
- By comparing the density of the data point and density of all the data points in the neighborhood, whether the density of the data point is lower than the density of the neighborhood can be determined.
- This scenario indicates the presence of an outlier.

Local Outlier Factor

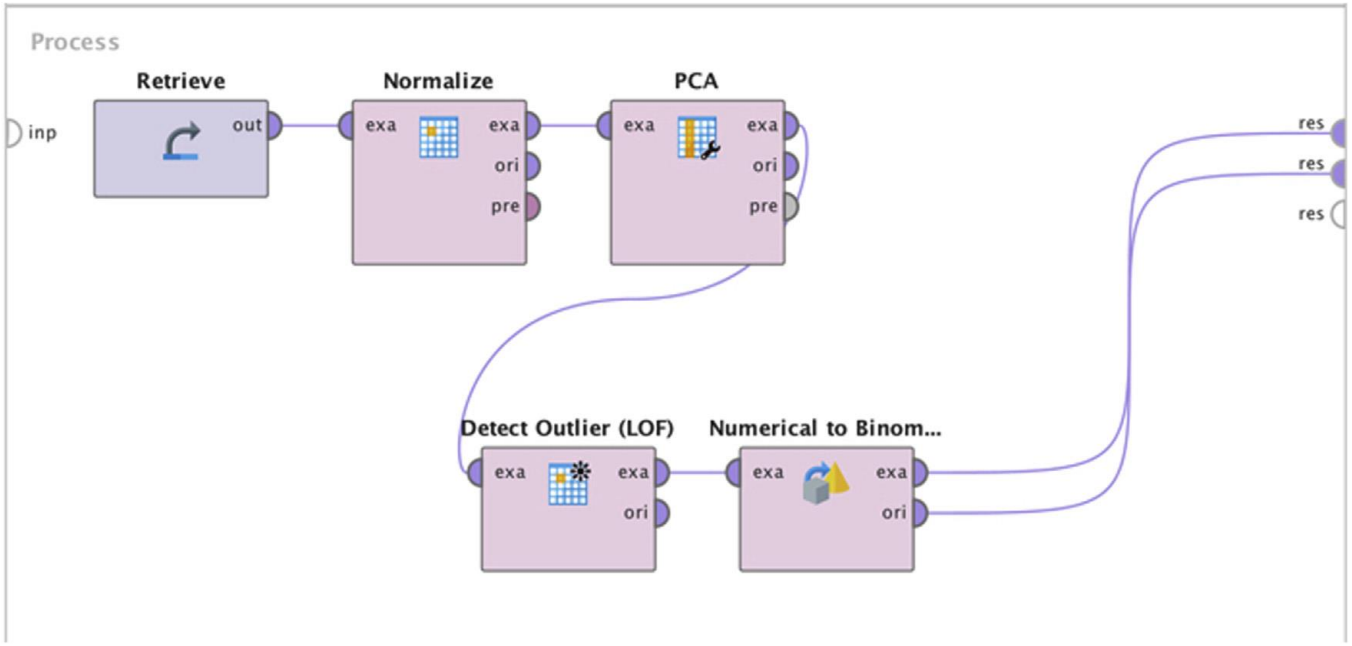
- **How to Implement-**
- A LOF-based data science process is similar to the other outlier processes explained in RapidMiner.
- The Detect Outlier (LOF) operator is available in Data Transformation > Data Cleansing > Outlier Detection.
- The output of the LOF operator contains the example set along with a numeric outlier score.
- The LOF algorithm does not explicitly label a data point as an outlier; instead the score is exposed to the user.
- This score can be used to visualize a comparison to a threshold, above which the data point is considered an outlier.
- Having the raw score means that the data science practitioner can "tune" the detection criteria, without having to rerun the scoring process, by changing the threshold for comparison.

Local Outlier Factor

- **How to Implement- Step 1: Data Preparation**
- **Similar to the distance- and density-based outlier-detection processes, the dataset has to be normalized using Normalize operator.**
- **The PCA operator is used to reduce the four-dimensional Iris dataset to two dimensions, so that the output can be visualized easily.**

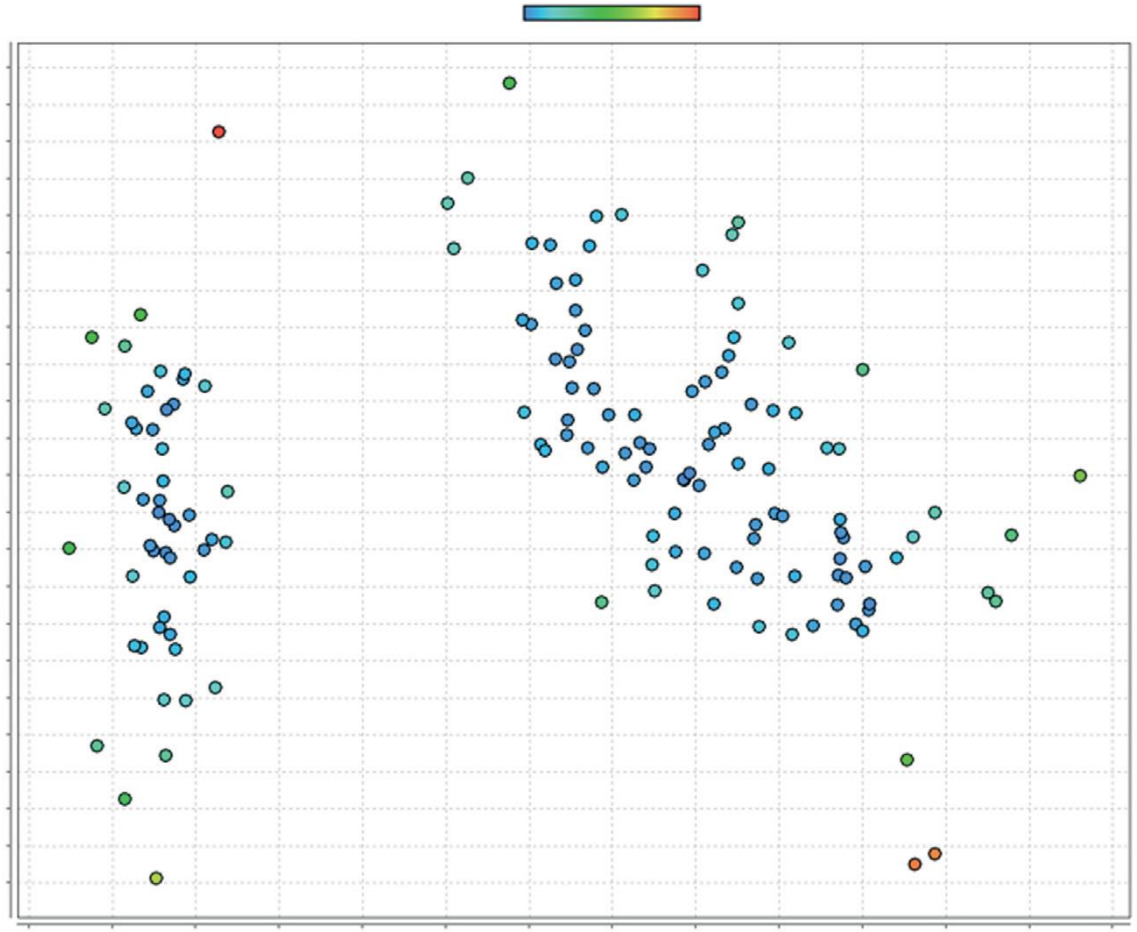
Local Outlier Factor

- **How to Implement-Step 2: Detect Outlier Operator**
- **The LOF operator has minimal points (MinPts) lower bound and upper bound as parameters.**
- **The MinPts lower bound is the value of k, the neighborhood number.**
- **The LOF algorithm also takes into account a MinPts upper bound to provide more stable results.**
- **Figure shows the RapidMiner process.**



Local Outlier Factor

- **How to Implement-Step 3: Results Interpretation**
- **After using the Detect Outlier operator, the outlier score is appended to the result dataset.**
- **Figure shows the result set with outlier score represented as the color of the data point.**



Local Outlier Factor

- **How to Implement-Step 3: Results Interpretation**
- **In the results window, the outlier score can be used to color the data points.**
- **The scatterplot indicates that points closer to the blue spectrum (left side of the outlier scale in the chart legend) are predicted to be regular data points and points closer to the red spectrum (right side of the outlier scale in the chart legend) are predicted to be outliers.**
- **If an additional Boolean flag indicating whether a data point is an outlier or not is needed, a Numeric to Binominal operator can be added to the result dataset.**
- **The Numeric to Binominal operator converts the numeric outlier score to a binominal true or false based on the threshold specification in the parameter of the operator and to the score output from the LOF operator.**

Feature Selection

Feature Selection

- An overused rubric in data science circles is that 80% of the analysis effort is spent on data cleaning and preparation and only 20% is typically spent on modeling.
- Feature selection in data science refers to the process of identifying the few most important variables or attributes that are essential to a model for an accurate prediction.
- This process is known by various terms: feature selection, dimension reduction, variable screening, key parameter identification, attribute weighting or regularization.
- There are fundamentally two types of feature selection processes: filter type and wrapper type.
- Filter approaches work by selecting only those attributes that rank among the top in meeting certain stated criteria.
- Wrapper approaches work by iteratively selecting, via a feedback loop, only those attributes that improve the performance of an algorithm.
- Among the filtertype methods, one can further classify based on the data types: numeric versus nominal.
- The most common wrapper-type methods are the ones associated with multiple regression: stepwise regression, forward selection, and backward elimination.

Classifying Feature Selection Methods

- There are two powerful technical motivations for incorporating feature selection in the data science process.
- First, a dataset may contain highly correlated attributes, such as the number of items sold, and the revenue earned by the sales of the item.
- Typically, there is no new information gained by including both of these attributes.
- Additionally, in the case of multiple regression-type models, if two or more of the independent variables (or predictors) are correlated, then the estimates of coefficients in a regression model tend to be unstable or counter intuitive. This is the multicollinearity.
- In the case of algorithms like naïve Bayesian classifiers, the attributes need to be independent of each other.
- Further, the speed of algorithms is typically a function of the number of attributes.
- So, by using only one among the correlated attributes the performance is improved.

Classifying Feature Selection Methods

- Second, a dataset may also contain redundant information that does not directly impact the predictions: as an extreme example, a customer ID number has no bearing on the amount of revenue earned from the customer.
- Such attributes may be filtered out by the analyst before the modeling process begins.
- However, not all attribute relationships are that clearly known in advance.
- In such cases, one must resort to computational methods to detect and eliminate attributes that add no new information.
- The key here is to include attributes that have a strong correlation with the predicted or dependent variable.
- So, to summarize, feature selection is needed to remove independent variables that may be strongly correlated to one another, and to keep independent variables that may be strongly correlated to the predicted or dependent variable.

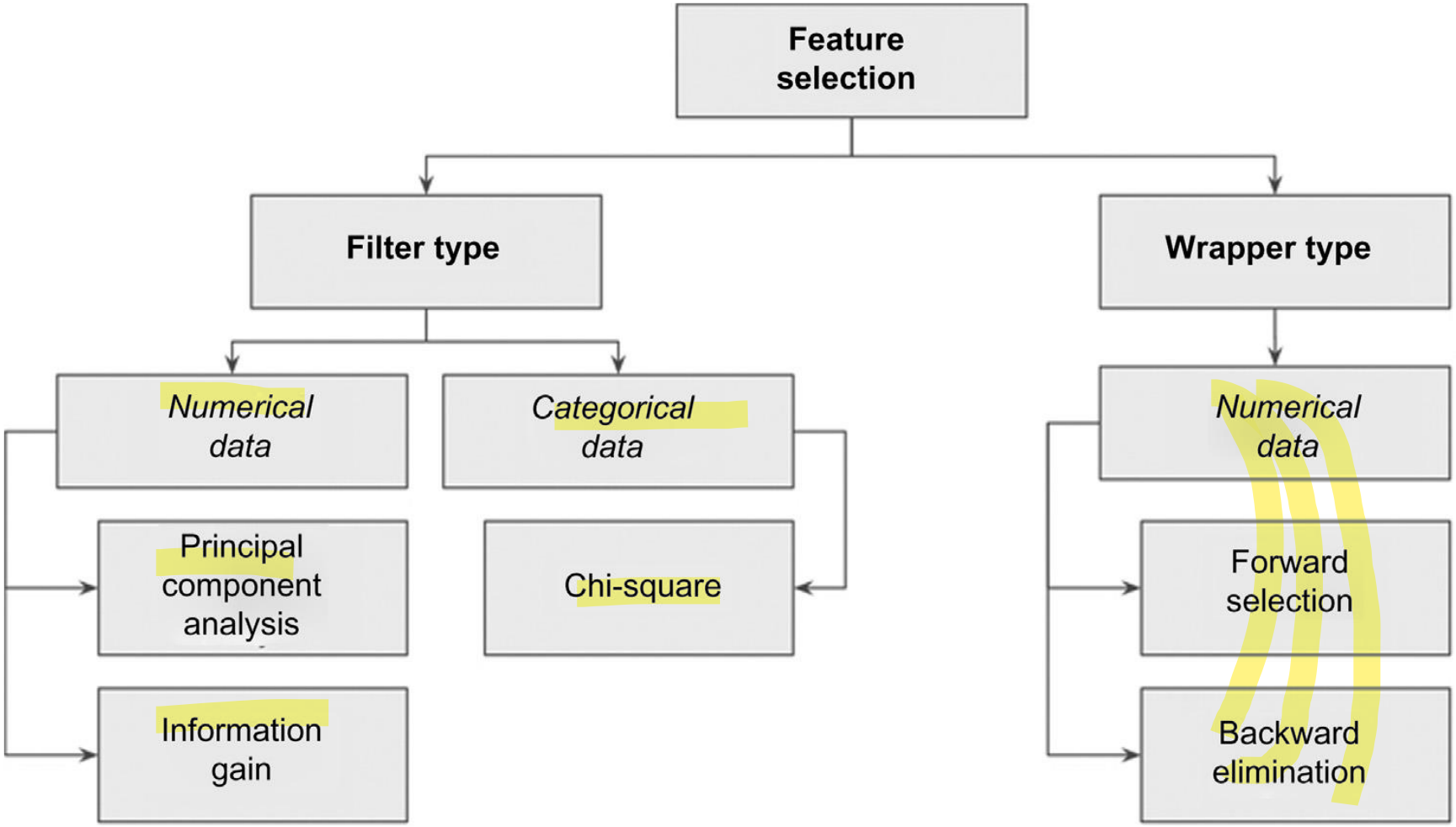
Classifying Feature Selection Methods

- Feature selection methods can be applied before the modeling process starts and, thus, unimportant attributes will get filtered out, or feature selection methods can be applied iteratively within the flow of the data science process.
- Depending on the logic, there are two feature selection schemes: filter schemes or wrapper schemes.
- The filter scheme does not require any learning algorithm, whereas the wrapper type is optimized for a particular learning algorithm.
- In other words, the filter scheme can be considered unsupervised and the wrapper scheme can be considered a supervised feature selection method.
- The filter model is commonly used:

1. When the number of features or attributes is really large
2. When computational expense is a criterion

Classifying Feature Selection Methods

- The chart summarizes a high-level taxonomy of feature selection methods.



Principal Component Analysis

- Assume that there is a dataset with m attributes.
- These could be for example, commodity prices, weekly sales figures, number of hours spent by assembly line workers, etc.; in short, any business parameter that can have an impact on a performance that is captured by a label or target variable.
- The question that PCA helps to answer is fundamentally this: Which of these m attributes explain a significant amount of variation contained within the dataset?
- PCA essentially helps to apply an 80/20 rule: Can a small subset of attributes explain 80% or more of the variation in the data?
- This sort of variable screening or feature selection will make it easy to apply other data science techniques and also make the job of interpreting the results easier.

Principal Component Analysis

- PCA captures the attributes that contain the greatest amount of variability in the dataset.
- It does this by transforming the existing variables into a set of principal components or new variables that have the following properties.

1. They are uncorrelated with each other.

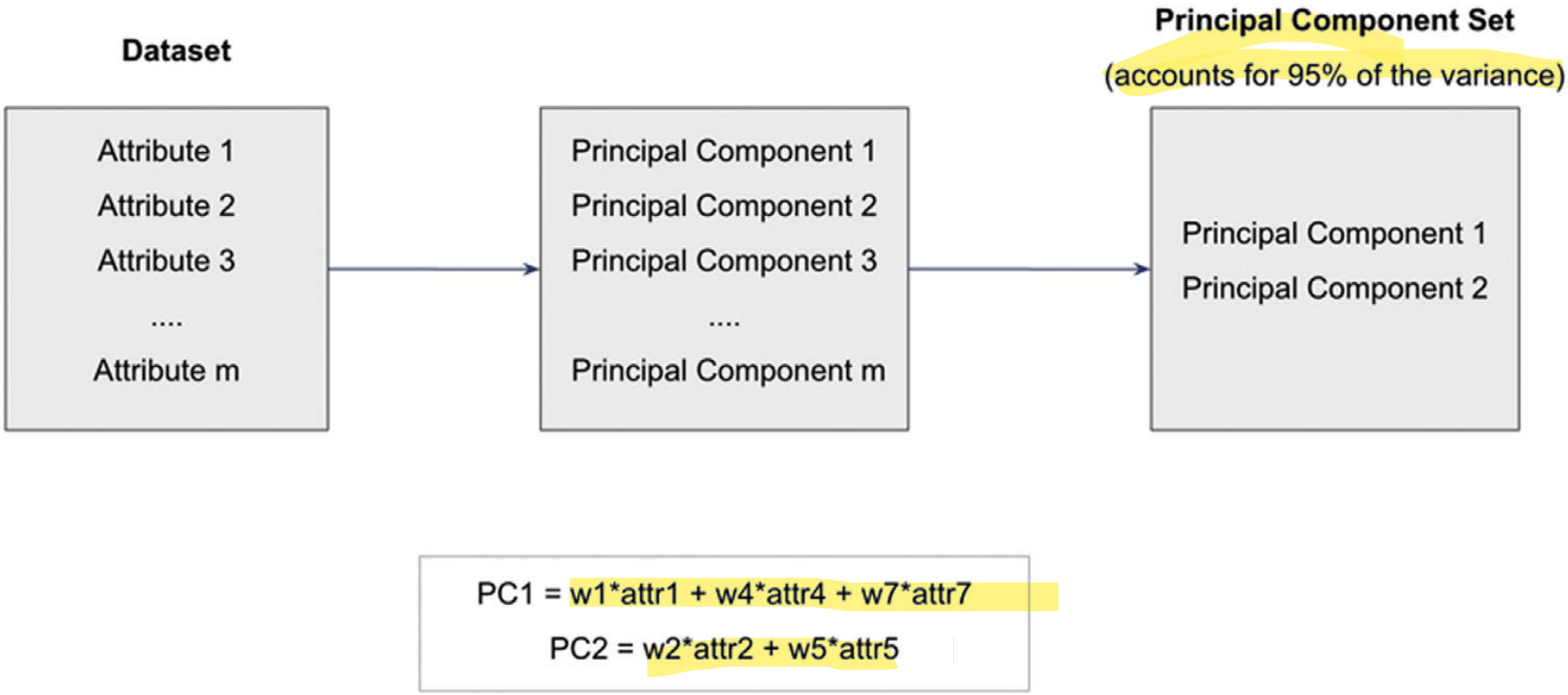
2. They cumulatively contain/explain a large amount of variance within the data.

3. They can be related back to the original variables via weighting factors.

- The original variables with very low weighting factors in their principal components are effectively removed from the dataset.

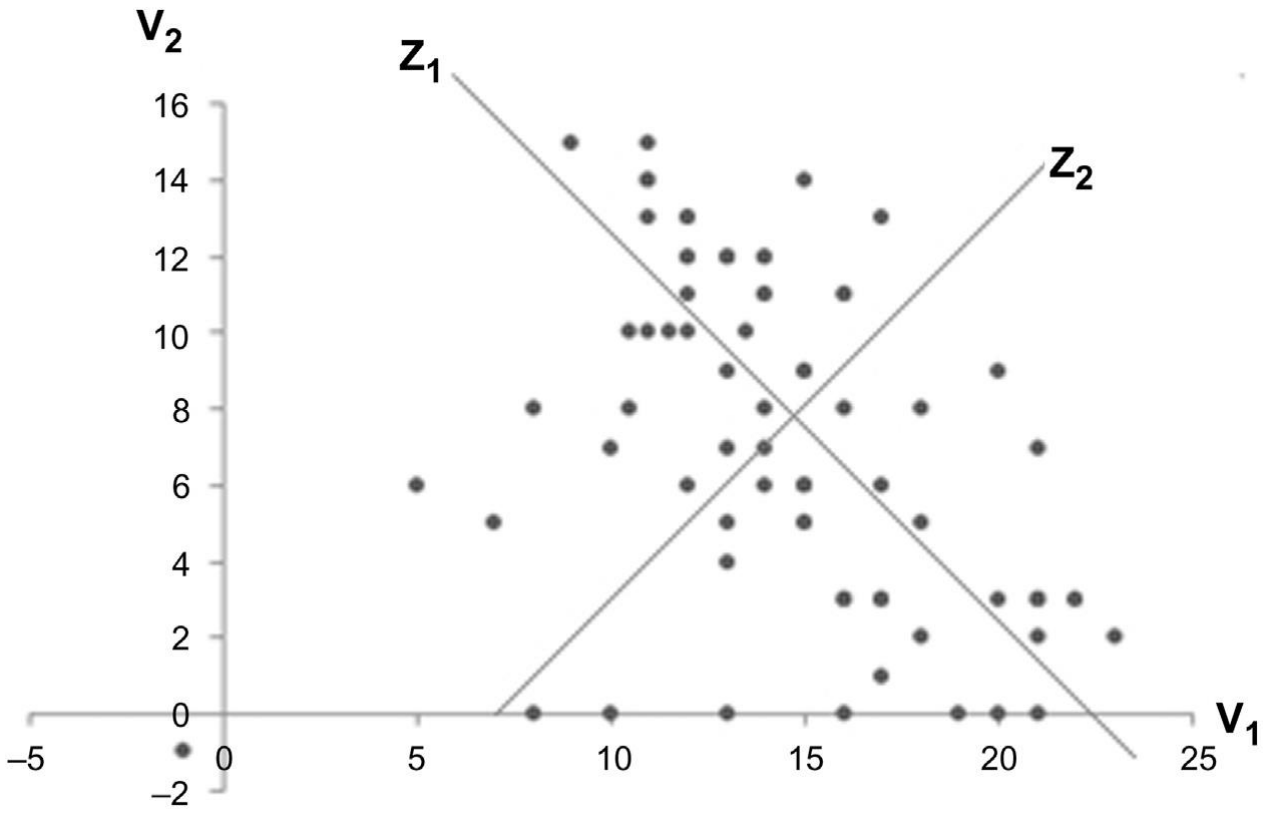
Principal Component Analysis

- The conceptual schematic in Figure illustrates how PCA can help in reducing data dimensions with a hypothetical dataset of m variables.



Principal Component Analysis-How it works

- The key task is computing the principal components, zm, which have the properties that were described.
- Consider the case of just two variables: v1 and v2.
- When the variables are visualized using a scatterplot, something like the one shown in Figure would be observed.



Principal Component Analysis-How it works

- As can be seen, v1 and v2 are correlated.
- But v1 and v2 could be transformed into two new variables z1 and z2, which meet the guidelines for principal components, by a simple linear transformation.
- As seen in the chart, this amounts to plotting the points along two new axes: z1 and z2.
- Axis z1 contains the maximum variability, and one can rightly conclude that z1 explains a significant majority of the variation present in the data and is the first principal component.
- z2, by virtue of being orthogonal to z1, contains the next highest amount of variability.
- Between z1 and z2, 100% of the total variability in the data can be accounted for (in this case of two variables).

Principal Component Analysis-How it works

- Furthermore, z1 and z2 are uncorrelated. As the number of variables, vm, increases it's possible that only the first few principal components are sufficient to express all the data variances.
- The principal components, zm, are expressed as a linear combination of the underlying variables, vm: $z_m = \sum w_i \times x_i$
- When this logic is extended to more than two variables, the challenge is to find the transformed set of principal components using the original variables.
- This is easily accomplished by performing an eigenvalue analysis of the covariance matrix of the original attributes.
- The eigenvector associated with the largest eigenvalue is the first principal component; the eigenvector associated with the second largest eigenvalue is the second principal component and so on.

Principal Component Analysis-How it works

- The covariance explains how two variables vary with respect to their corresponding mean values—if both variables tend to stay on the same side of their respective means, the covariance would be positive, if not it would be negative. (In statistics, covariance is also used in the calculation of correlation coefficient.)

$$\mathrm{Cov}_{ij} = E[V_i V_j] - E[V_i]E[V_j]$$

- where expected value $E[v] = v_k \, P(v = v_k)$
- For the eigenvalue analysis, a matrix of such covariances between all pairs of variables vm is created.

Principal Component Analysis-How to implement

- The dataset includes information on ratings and nutritional information on 77 breakfast cereals.
- There are a total of 16 variables, including 13 numerical parameters

Table 14.1 Breakfast Cereals Dataset for Dimension Reduction Using PCA

Name	mfr	Type	Calories	Protein	Fat	Sodium	Fiber	Carbo	Sugars	Potass	Vitamins	Shelf	Weight	Cups	Rating
100%_Bran	N	C	70	4	1	130	10	5	6	280	25	3	1	0.33	68.402973
100%_Natural_Bran	Q	C	120	3	5	15	2	8	8	135	0	3	1	1	33.983679
All-Bran	K	C	70	4	1	260	9	7	5	320	25	3	1	0.33	59.425505
All-Bran_with_Extra_Fiber	K	C	50	4	0	140	14	8	0	330	25	3	1	0.5	93.704912
Almond_Delight	R	C	110	2	2	200	1	14	8	-1	25	3	1	0.75	34.384843
Apple_Cinnamon_Cheerios	G	C	110	2	2	180	1.5	10.5	10	70	25	1	1	0.75	29.509541
Apple_Jacks	K	C	110	2	0	125	1	11	14	30	25	2	1	1	33.174094
Basic_4	G	C	130	3	2	210	2	18	8	100	25	3	1.33	0.75	37.038562

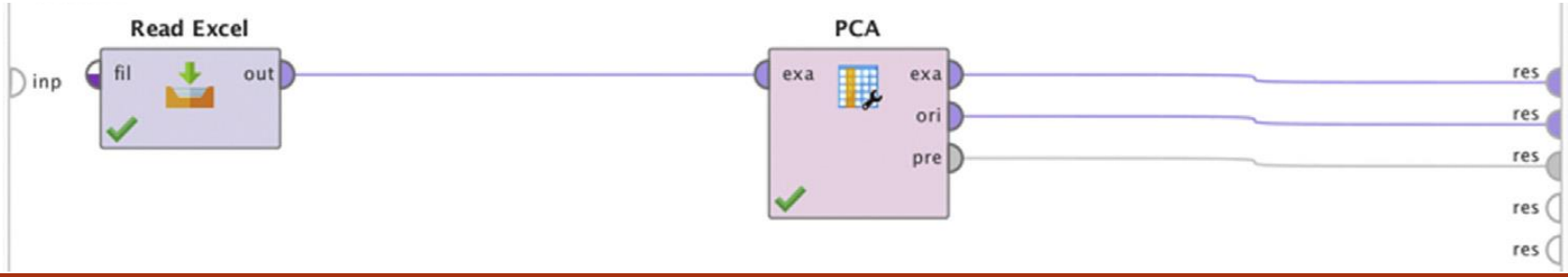
Principal Component Analysis-How to implement

Step 1: Data Preparation

- Remove the non-numeric parameters such as Cereal name, Manufacturer, and Type (hot or cold), as PCA can only work with numeric attributes.
- These are the first three columns in Table.
- In RapidMiner, these can be converted into ID attributes if needed for reference later.
- This can be done during the import of the dataset into RapidMiner during the next step if needed; in this case these variables will simply be removed.
- The Select Attributes operator may also be used following the Read Excel operator to remove these variables.
- Read the Excel file into RapidMiner: this can be done using the standard Read Excel operator as described in earlier sections.

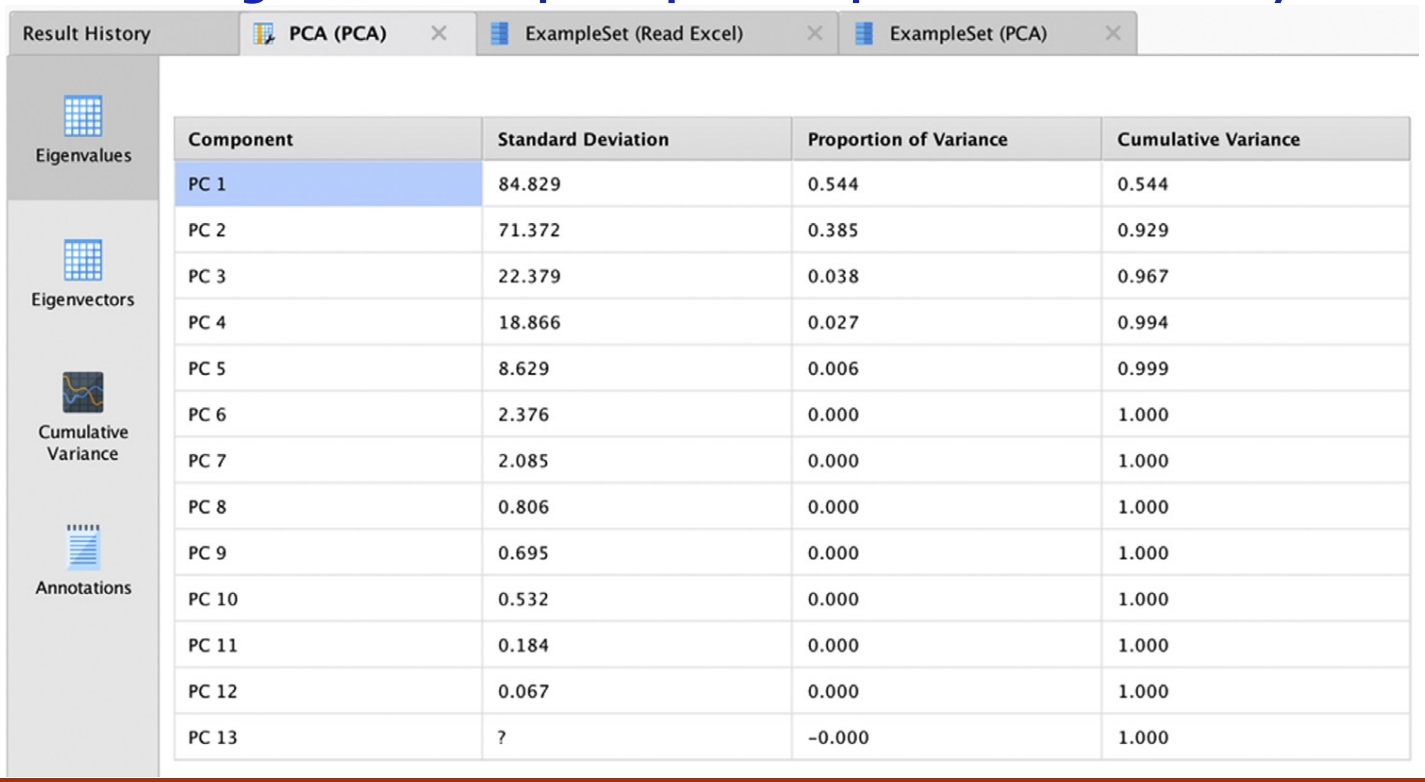
Principal Component Analysis-How to implement

- **Step 2: PCA Operator**
- Type in the keyword PCA in the operator search field and drag and drop the PCA operator into the main process window.
- Connect the output of Read Excel into the ports of the PCA operator.
- The three available parameter settings for dimensionality reduction are none, keep variance, and fixed number.
- Here use keep variance and leave the variance threshold at the default value of 0.95% or 95% .
- The variance threshold selects only those attributes that collectively account for or explain 95% (or any other value set by user) of the total variance in the data.
- Connect all output ports from the PCA operator to the results ports.



Principal Component Analysis-How to implement

- Step 3: Execution and Interpretation
- By running the analysis as configured above, RapidMiner will output several tabs in the results panel (Fig.).
- By clicking on the PCA tab, three PCA related tabs will be seen—Eigenvalues, Eigenvectors, and Cumulative Variance Plot.
- Using Eigenvalues, information can be obtained about the contribution to the data variance coming from each principal component individually and cumulatively.



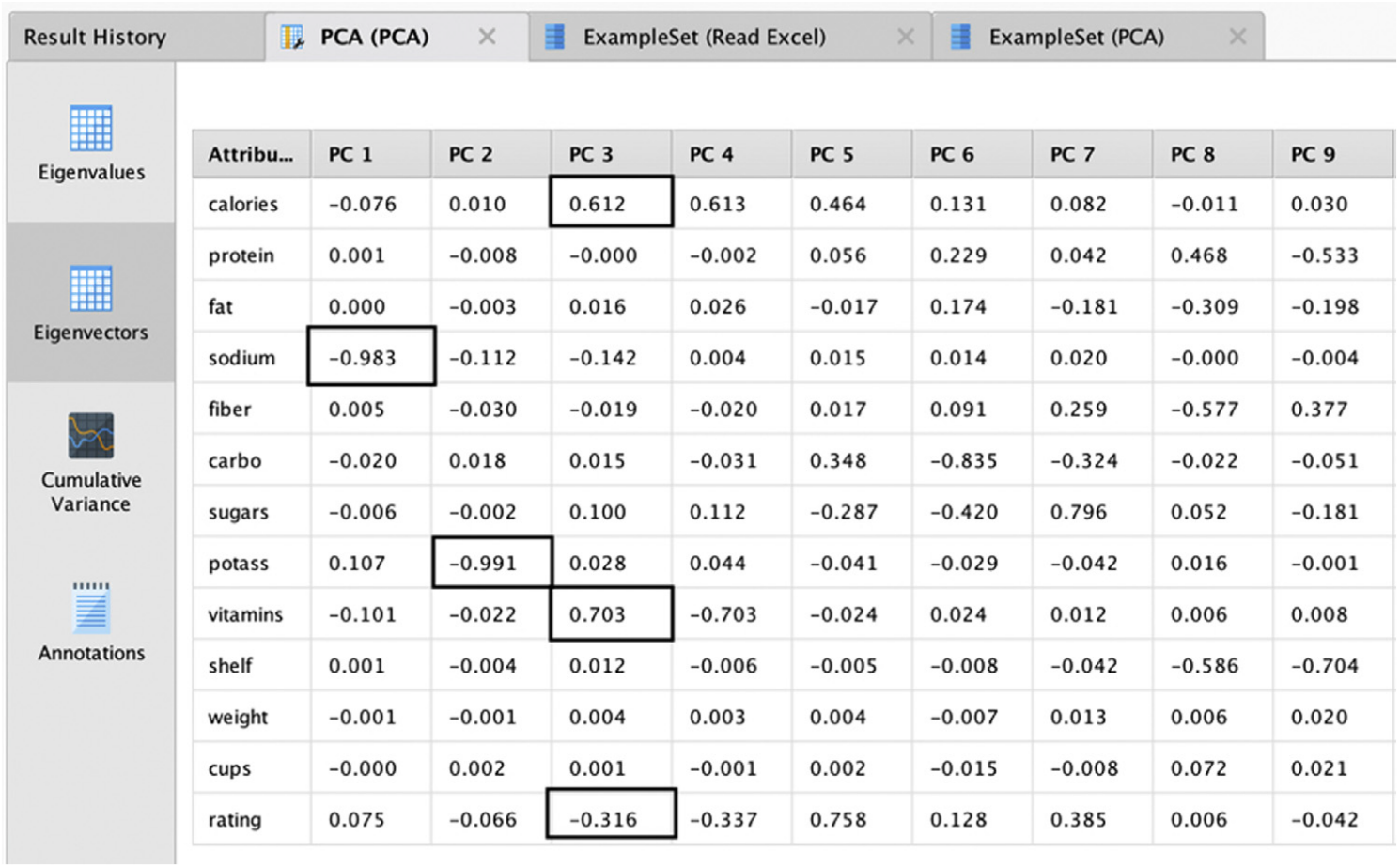
Component	Standard Deviation	Proportion of Variance	Cumulative Variance
PC 1	84.829	0.544	0.544
PC 2	71.372	0.385	0.929
PC 3	22.379	0.038	0.967
PC 4	18.866	0.027	0.994
PC 5	8.629	0.006	0.999
PC 6	2.376	0.000	1.000
PC 7	2.085	0.000	1.000
PC 8	0.806	0.000	1.000
PC 9	0.695	0.000	1.000
PC 10	0.532	0.000	1.000
PC 11	0.184	0.000	1.000
PC 12	0.067	0.000	1.000
PC 13	?	-0.000	1.000

Principal Component Analysis-How to implement

- **Step 3: Execution and Interpretation**
- If, for example, our variance threshold is 95%, then PC 1, PC 2, and PC 3 are the only principal components that need to be considered because they are sufficient to explain nearly 97% of the variance.
- PC 1 contributes to a majority of this variance, about 54%.
- One can then deep dive into these three components and identify how they are linearly related to the actual or real parameters from the dataset.
- At this point only those real parameters can be considered that have significant weight contribution to the each of the first three PCs.
- These will ultimately form the subset of reduced parameters for further predictive modeling.
- The key question is how does one select the real variables based on this information?
- RapidMiner allows the eigenvectors (weighting factors) to be sorted for each PC and one can decide to choose the two to three highest (absolute) valued weighting factors for PCs 1-3.

Principal Component Analysis-How to implement

- As seen from Figure, the highlighted real attributes have been chosen—calories, sodium, potassium, vitamins, and rating—to form the reduced dataset.
- This selection was done by simply identifying the top three attributes from each principal component.



Attribu...	PC 1	PC 2	PC 3	PC 4	PC 5	PC 6	PC 7	PC 8	PC 9
calories	-0.076	0.010	0.612	0.613	0.464	0.131	0.082	-0.011	0.030
protein	0.001	-0.008	-0.000	-0.002	0.056	0.229	0.042	0.468	-0.533
fat	0.000	-0.003	0.016	0.026	-0.017	0.174	-0.181	-0.309	-0.198
sodium	-0.983	-0.112	-0.142	0.004	0.015	0.014	0.020	-0.000	-0.004
fiber	0.005	-0.030	-0.019	-0.020	0.017	0.091	0.259	-0.577	0.377
carbo	-0.020	0.018	0.015	-0.031	0.348	-0.835	-0.324	-0.022	-0.051
sugars	-0.006	-0.002	0.100	0.112	-0.287	-0.420	0.796	0.052	-0.181
potass	0.107	-0.991	0.028	0.044	-0.041	-0.029	-0.042	0.016	-0.001
vitamins	-0.101	-0.022	0.703	-0.703	-0.024	0.024	0.012	0.006	0.008
shelf	0.001	-0.004	0.012	-0.006	-0.005	-0.008	-0.042	-0.586	-0.704
weight	-0.001	-0.001	0.004	0.003	0.004	-0.007	0.013	0.006	0.020
cups	-0.000	0.002	0.001	-0.001	0.002	-0.015	-0.008	0.072	0.021
rating	0.075	-0.066	-0.316	-0.337	0.758	0.128	0.385	0.006	-0.042

Principal Component Analysis-How to implement

- For this example, PCA reduces the number of attributes from 13 to 5, a more than 50% reduction in the number of attributes that any model would need to realistically consider.
- One can imagine the improvement in performance as one deals with the larger datasets that PCA enables.
- In practice, PCA is an effective and widely used tool for dimension reduction, particularly when all the attributes are numeric.
- It works for a variety of real-world applications, but it should not be blindly applied for variable screening.
- For most practical situations, domain knowledge should be used in addition to PCA analysis before eliminating any of the variables.

Principal Component Analysis-Risks

1. The results of a PCA must be evaluated in the context of the data.

- If the data is extremely noisy, then PCA may end up suggesting that the noisiest variables are the most significant because they account for most of the variation! An analogy would be the total sound energy in a rock concert.
- If the crowd noise drowns out some of the high-frequency vocals or notes, PCA might suggest that the most significant contribution to the total energy comes from the crowd—and it will be right!
- But this does not add any clear value if one is attempting to distinguish which musical instruments are influencing the harmonics, for example.

Principal Component Analysis-Risks

2. Adding uncorrelated data does not always help.

- Neither does adding data that may be correlated, but irrelevant.
- When more parameters are added to the dataset, and if these parameters happen to be random noise, effectively the same situation as the first point applies.
- On the other hand, caution also has to be exercised and spurious correlations have to looked out for.
- As an extreme example, it may so happen that there is a correlation between the number of hours worked in a garment factory and pork prices (an unrelated commodity) within a certain period of time.
- Clearly this correlation is probably pure coincidence. Such correlations again can muddy the results of a PCA.
- Care must be taken to winnow the dataset to include variables that make business sense and are not subjected to many random fluctuations before applying a technique like PCA.

Principal Component Analysis-Risks

3. PCA is very sensitive to scaling effects in the data.

- If the data in the example is closely examined, it will be observed that the top attributes that PCA helped identify as the most important ones also have the widest range (and standard deviation) in their values.
- For example, potassium ranges from 21 to 330 and sodium ranges from 1 to 320.
- Comparatively, most of the other factors range in the single or low double digits.
- As expected, these factors dominate PCA results because they contribute to the maximum variance in the data.
- What if there was another factor such as sales volume, which would potentially range in the millions (of dollars or boxes), that were to be considered for a modeling exercise? Clearly it would mask the effects of any other attribute.

Principal Component Analysis-Risks

- To minimize scaling effects, one can range normalize the data (using for example, the Normalize operator).
- When this data transformation is applied, all the attributes are reduced to a range between 0 and 1 and scale effects will not matter anymore. But what happens to the PCA results?

PCA (PCA) | ExampleSet (Normalize) | ExampleSet (PCA)

Eigenvalues

Eigenvectors

Cumulative Variance

Annotations

Component	Standard Deviation	Proportion of Variance	Cumulative Variance
PC 1	0.459	0.306	0.306
PC 2	0.392	0.224	0.530
PC 3	0.302	0.132	0.663
PC 4	0.287	0.120	0.782
PC 5	0.212	0.065	0.848
PC 6	0.187	0.051	0.898
PC 7	0.162	0.038	0.937
PC 8	0.140	0.029	0.965
PC 9	0.122	0.022	0.987
PC 10	0.067	0.006	0.993
PC 11	0.053	0.004	0.997
PC 12	0.043	0.003	1.000
PC 13	?	-0.000	1.000

Principal Component Analysis-Risks

- As Figure shows, eight PCs are now needed to account for the same 95% total variance. As an exercise, use the eigenvectors to filter out the attributes that are included in these eight PCs and it would be observed that (applying the top three rule for each PC as before), none of the attributes would be eliminated!
- This leads to the next section on feature selection methods that are not scale sensitive and also work with non-numerical datasets, which were two of the limitations with PCA.

Information Theory-based Filtering

- The concepts of information gain and gain ratio involve comparing the information exchanged between a given attribute and the target or label attribute.
- the key to feature selection is to include attributes that have a strong correlation with the predicted or dependent variable.
- With these techniques, one can rank attributes based on the amount of information gain and then select only those that meet or exceed some (arbitrarily) chosen threshold or simply sselect the top k (again, arbitrarily chosen) features.

Information Theory-based Filtering

- **The golf example.**

(A)

Row No.	id	Play	Outlook	Temperature	Humidity	Wind
1	1	no	sunny	85	85	false
2	2	no	sunny	80	90	true
3	3	yes	overcast	83	78	false
4	4	yes	rain	70	96	false
5	5	yes	rain	68	80	false
6	6	no	rain	65	70	true
7	7	yes	overcast	64	65	true
8	8	no	sunny	72	95	false
9	9	yes	sunny	69	70	false
10	10	yes	rain	75	80	false
11	11	yes	sunny	75	70	true
12	12	yes	overcast	72	90	true
13	13	yes	overcast	81	75	false
14	14	no	rain	71	80	true

Row No.	Play	Outlook
1	no	sunny
2	no	sunny
3	yes	overcast
4	yes	rain
5	yes	rain
6	no	rain
7	yes	overcast
8	no	sunny
9	yes	sunny
10	yes	rain
11	yes	sunny
12	yes	overcast
13	yes	overcast
14	no	rain

Information Theory-based Filtering

- When the information gain calculation methodology is applied to compute information gain for all attributes, the feature ranking in Fig. B will be reached, in terms of their respective influence on the target variable Play.
- This can be easily done using the Weight by Information Gain operator in RapidMiner.
- The output looks almost identical to the one shown in Table, except for the slight differences in the information gain values for Temperature and Humidity.

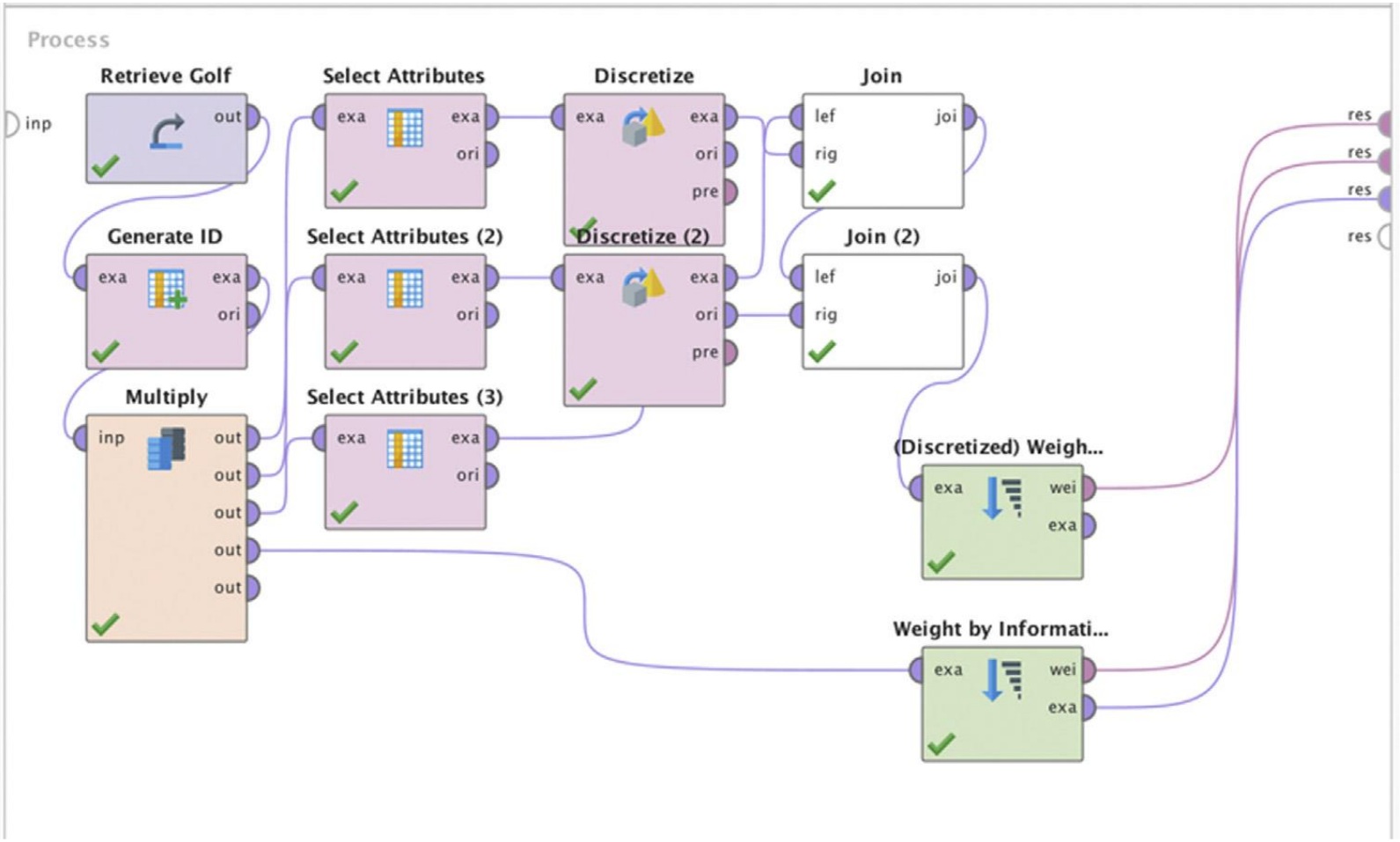
attribute	weight ↓
Outlook	0.247
Temperature	0.113
Humidity	0.102
Wind	0.048

Information Theory-based Filtering

- The reason is that for that dataset, the temperature and humidity had to be converted into nominal values before computing the gains.
- In this case, the numeric attributes are used as they are.
- So, it is important to pay attention to the discretization of the attributes before filtering.
- Use of information gain feature selection is also restricted to cases where the label is nominal.
- For fully numeric datasets, where the label variable is also numeric, PCA or correlation-based filtering methods are commonly used.

Information Theory-based Filtering

- Fig. describes a process that uses the sample Golf dataset available in RapidMiner.
- The various steps in the process convert numeric attributes, Temperature and Humidity, into nominal ones.



Information Theory-based Filtering

- In the final step, the Weight by Information Gain operator is applied to both the original data and the converted dataset in order to show the difference between the gain computed using different data types.
- The main point to observe is that the gain computation depends not only on the data types, but also on how the nominal data is discretized.
- For example, one gets slightly different gain values (see Table 14.2) if Humidity is divided into three bands (high, medium, and low) as opposed to only two bands (high and low). These variants can be tested easily using the process described.
- In conclusion, the top-ranked attributes are selected.
- In this case, they would be Outlook and Temperature if one chose the non-discretized version, and Outlook and Humidity in the discretized version.

Table 14.2 Results of Information Gain Feature Selection

Attribute	Info Gain Weight (Not Discretized)	Info Gain Weight (Discretized)
Outlook	0.247	0.247
Temperature	0.113	0.029
Humidity	0.102	0.104
Wind	0.048	0.048

Chi-square-based Filtering

- In many cases the datasets may consist of only categorical (or nominal) attributes.
- In this case, what is a good way to distinguish between high influence attributes and low or no influence attributes?
- A classic example of this scenario is the gender selection bias. Suppose one has data about the purchase of a big-ticket item like a car or a house.
- Can the influence of gender on purchase decisions be verified? Are men or women the primary decision makers when it comes to purchasing big-ticket items? For example, is gender a factor in color preference of a car? Here attribute 1 would be gender and attribute 2 would be the color.
- A chi-square test would reveal if there is indeed a relationship between these two attributes.
- If there are several attributes and one wishes to rank the relative influence of each of these on the target attribute, the chi-square statistic can still be used.

Chi-square-based Filtering

- The golf example in Figure—this time all numeric attributes have been converted into nominal ones.
- Chi-square analysis involves counting occurrences (number of sunny days or windy days) and comparing these variables to the target variable based on the frequencies of occurrences.

Row No.	id	Play	Humidity	Temperature	Outlook	Wind
1	1	no	High	hot	sunny	false
2	2	no	High	hot	sunny	true
3	3	yes	Normal	hot	overcast	false
4	4	yes	High	mild	rain	false
5	5	yes	Normal	cool	rain	false
6	6	no	Normal	cool	rain	true
7	7	yes	Normal	cool	overcast	true
8	8	no	High	mild	sunny	false
9	9	yes	Normal	cool	sunny	false
10	10	yes	Normal	mild	rain	false
11	11	yes	Normal	mild	sunny	true
12	12	yes	High	mild	overcast	true
13	13	yes	Normal	hot	overcast	false
14	14	no	Normal	mild	rain	true

Chi-square-based Filtering

- The chisquare test checks if the frequencies of occurrences across any pair of attributes, such as Outlook=overcast and Play=yes, are correlated.
- In other words, for the given Outlook type, overcast, what is the probability that Play=yes (existence of a strong correlation)?
- The multiplication law of probabilities states that if event A happening is independent of event B, then the probabilities of A and B happening together is simply pAXpB.
- The next step is to convert this joint probability into an expected frequency, which is given by pAXpBXN, where N is the sum of all occurrences in the dataset.

Chi-square-based Filtering

- For each attribute, a table of observed frequencies, such as the one shown in Table is built. This is called a contingency table.
- The last column and row (the margins) with heading 'Totals' are simply the sums in the corresponding rows or columns as can be verified.

Table 14.3 Contingency Table of Observed Frequencies for Outlook and the Label Attribute, Play

Outlook	Sunny	Overcast	Rain	Total
Play = no	3	0	2	**5**
Play = yes	2	4	3	**9**
Total	**5**	**4**	**5**	**14**

Chi-square-based Filtering

- Using the contingency table, a corresponding expected frequency table can be built using the expected frequency definition (pAXpBXN) from which the chi-square statistic is then computed by comparing the difference between the observed frequency and expected frequency for each attribute.
- The expected frequency table for Outlook is shown in Table.

Table 14.4 Expected Frequency Table

Outlook	Sunny	Overcast	Rain	Total
Play = no	1.785714	1.428571	1.785714	**5**
Play = yes	3.214286	2.571429	3.214286	**9**
Total	**5**	**4**	**5**	**14**

Chi-square-based Filtering

- The expected frequency for the event [Play=no and Outlook=sunny] is calculated using the expected frequency formula: (5/14X5/14X14) =1.785 and is entered in the first cell as shown.
- Similarly, the other expected frequencies are calculated.
- The formula for the chi-square statistic is the summation of the square of the differences between observed and expected frequencies, as given in Eq

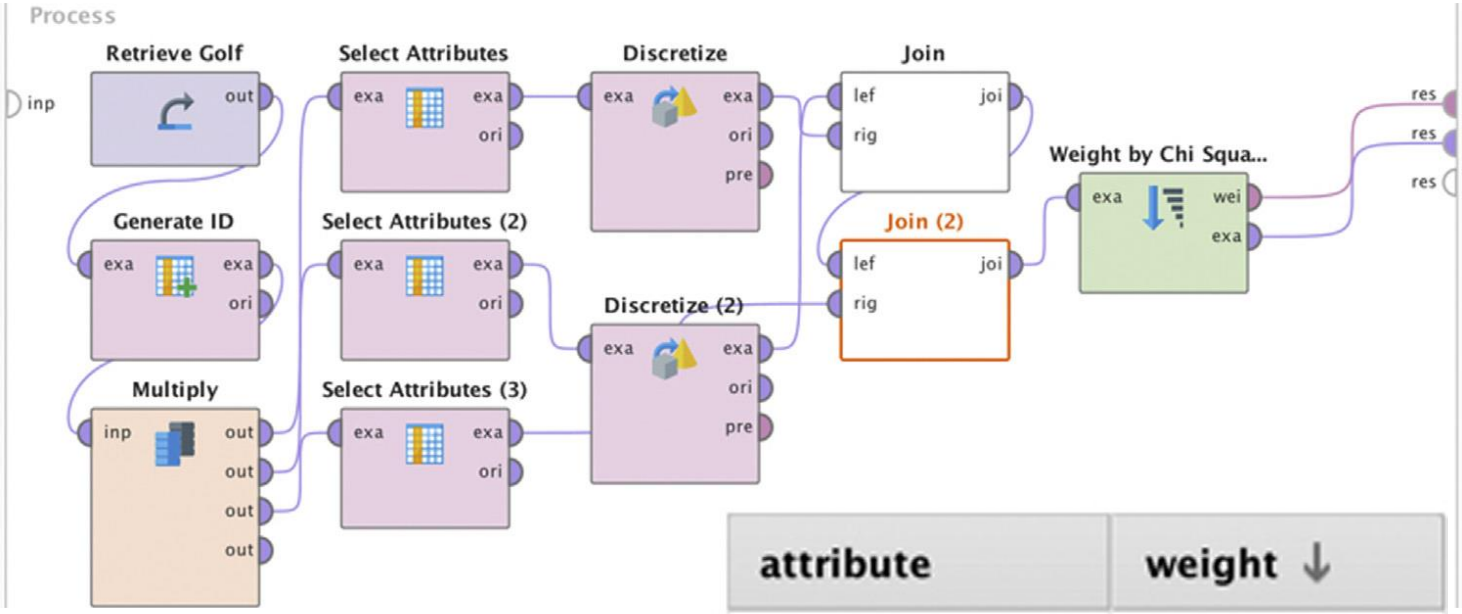
$$\chi^2 = \sum\sum \frac{(f_o - f_e)^2}{f_e}$$

where fo is the observed frequency and fe is the expected frequency.

- The test of independence between any two parameters is done by checking if the observed chi-square is less than a critical value which depends on the confidence level chosen by the user.
- In this case of feature weighting, all the observed chi-square values are simply gathered and used to rank the attributes.

Chi-square-based Filtering

- The ranking of attributes for the golf example is generated using the process described in Figure and is shown in the table of observed chi-square values in next figure.



attribute	weight ↓
Outlook	3.547
Humidity	1.998
Wind	0.933
Temperature	0.570

Chi-square-based Filtering

- Just like in information gain feature selection, most of the operators shown in the process are simply transforming the data into nominal values to generate it in the form shown in Figure A.

Row No.	id	Play	Humidity	Temperature	Outlook	Wind
1	1	no	High	hot	sunny	false
2	2	no	High	hot	sunny	true
3	3	yes	Normal	hot	overcast	false
4	4	yes	High	mild	rain	false
5	5	yes	Normal	cool	rain	false
6	6	no	Normal	cool	rain	true
7	7	yes	Normal	cool	overcast	true
8	8	no	High	mild	sunny	false
9	9	yes	Normal	cool	sunny	false
10	10	yes	Normal	mild	rain	false
11	11	yes	Normal	mild	sunny	true
12	12	yes	High	mild	overcast	true
13	13	yes	Normal	hot	overcast	false
14	14	no	Normal	mild	rain	true

attribute	weight ↓
Outlook	3.547
Humidity	1.998
Wind	0.933
Temperature	0.570

- Compare the output of the chi-square ranking to the information gain-based ranking (for the nominalized or discretized attributes) and it will be evident that the ranking is identical Figure B.
- Note that the Normalize weights option is sometimes also used, which is a range normalization onto the interval 0 to 1.

Wrapper-type Feature Selection

- In general, if a dataset contains k different attributes, then conducting all possible regression searches implies that one builds $2^k - 1$ separate regression models and picks the one that has the best performance.
- Clearly this is impractical.
- A better way, from a computational resource consumption point of view, to do this search would be to start with one variable, say v1, and build a baseline model.
- Then add a second variable, say v2, and build a new model to compare with the baseline.
- If the performance of the new model, for example, the R2, is better than that of the baseline, this model can be made to be the new baseline, add a third variable, v3, and proceed in a similar fashion.

Wrapper-type Feature Selection

- If, however, the addition of the second attribute, v2, did not improve the model significantly (over some arbitrarily prescribed level of improvement in performance), then a new attribute v3 can be chosen, and a new model built that includes v1 and v3.
- If this model is better than the model that included v1 and v2, proceed to the next step, where the next attribute v4 can be considered, and a model built that includes v1, v3, and v4.
- In this way, one steps forward selecting attributes one by one until the desired level of model performance is achieved.
- This process is called forward selection.

Wrapper-type Feature Selection

- A reverse of this process is where the baseline model with all the attributes is started, v1, v2, . . ., vk and for the first iteration, one of the variables, vj, is removed and a new model is constructed.
- However, how does one select which vj to remove? Here, it is typical to start with a variable that has the lowest t-stat value, seen in the case study described below.
- If the new model is better than the baseline, it becomes the new baseline and the search continues to remove variables with the lowest t-stat values until some stopping criterion is met (usually if the model performance is not significantly improved over the previous iteration).
- This process is called backward elimination.

Wrapper-type Feature Selection

- As observed, the variable selection process wraps around the modeling procedure, hence, the name for these classes of feature selection.
- A case study will now be examined, using data from the Boston Housing dataset, Regression Methods, to demonstrate how to implement the backward elimination method using RapidMiner.
- Recall that the data consists of 13 predictors and 1 response variable.
- The predictors include physical characteristics of the house (such as the number of rooms, age, tax, and location) and neighborhood features (schools, industries, zoning), among others.
- The response variable is the median value (MEDV) of the house in thousands of dollars.
- These 13 independent attributes are considered to be predictors for the target or label attribute.

Wrapper-type Feature Selection

- The snapshot of the data table is shown in Table.

Table 14.6 Sample View of the Boston Housing Dataset

CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9	5.33	36.2
0.02985	0	2.18	0	0.458	6.43	58.7	6.0622	3	222	18.7	394.12	5.21	28.7
0.08829	14.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.6	14.43	22.9
0.14455	14.5	7.87	0	0.524	6.172	96.1	5.9505	5	311	15.2	396.9	19.15	27.1

Wrapper-type Feature Selection

Backward Elimination

- In order to apply a wrapper-style feature selection method such as backward elimination, the training and testing process will need to be tucked inside another subprocess, a learning process.
- The learning process is now nested inside the Backward Elimination operator.
- Therefore, a double nesting as schematically shown in Figure is now obtained.
- Next, the Backward Elimination operator can be configured in RapidMiner.
- Double clicking on the Backward Elimination operator opens up the learning process, which can now accept the Split Validation operator that has been used many times.
- The Backward Elimination operator can now be filled in with the Split Validation operator and all the other operators and connections required to build a regression model.

Wrapper-type Feature Selection

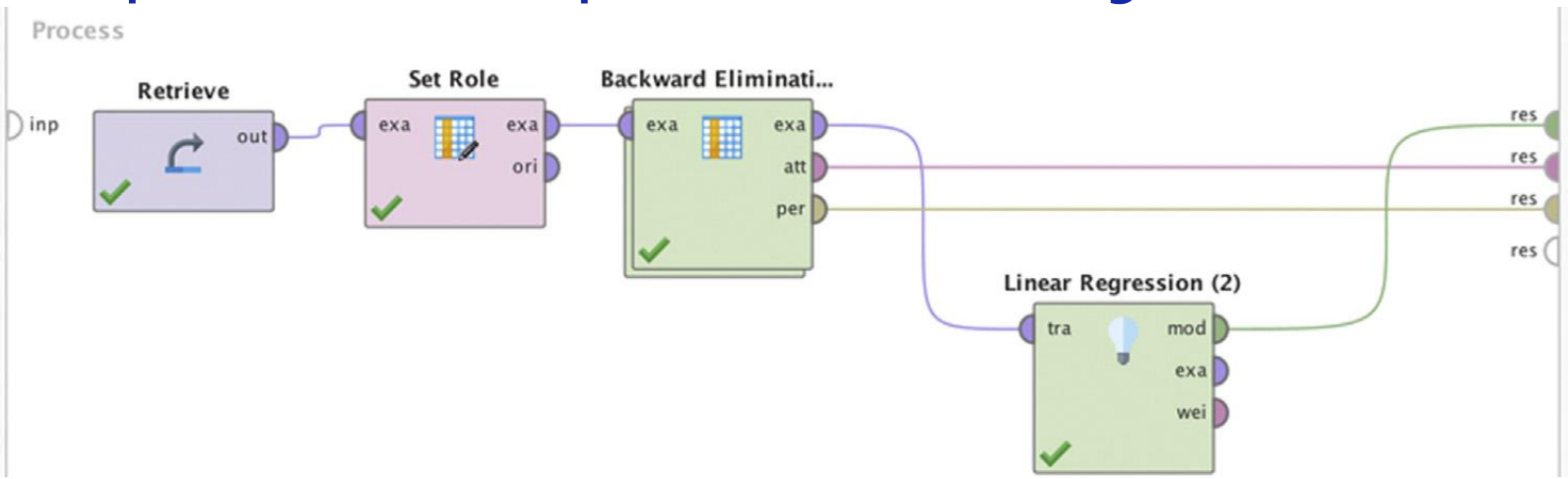
Backward Elimination

- The process of setting these up is exactly the same as discussed in Regression Methods, and, hence, is not repeated here.
- Now the configuration of the Backward Elimination operator will be examined.
- Here one can specify several parameters to enable feature selection. The most important one is the stopping behavior.
- The choices are "with decrease," "with decrease of more than," and "with significant decrease."
- The first choice is highly parsimonious—a decrease from one iteration to the next will stop the process. But if the second choice is chosen, one now has to now indicate a "maximal relative decrease."
- In this example, a 10% decrease has been indicated. Finally, the third choice is very stringent and requires achieving some desired statistical significance by allowing one to specify an alpha level.
-

Wrapper-type Feature Selection

Backward Elimination

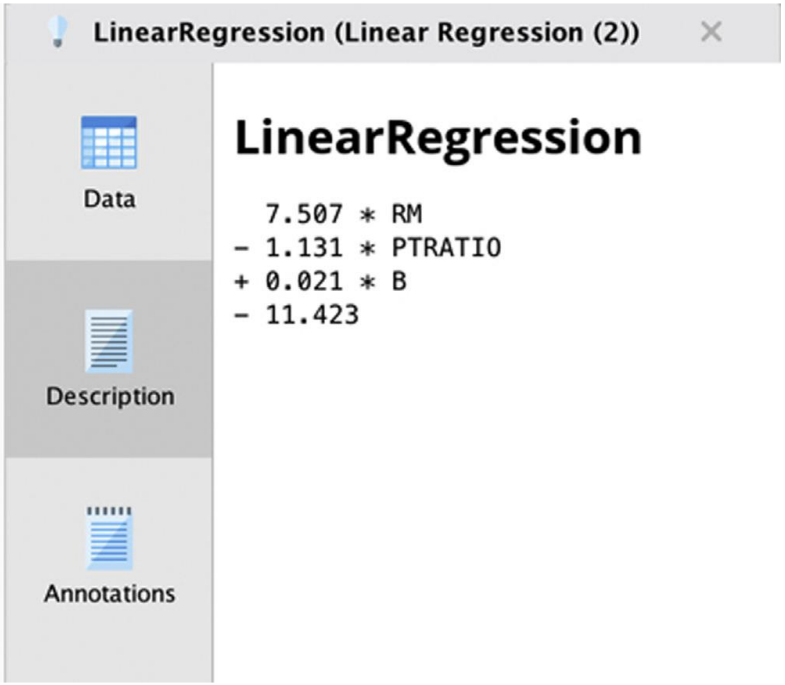
- The output of this operator will contain the model, which can be examined in the Results perspective.
- The top level of the final process is shown in Figure.



Wrapper-type Feature Selection

Backward Elimination

- It is evident that nine attributes have been eliminated. Perhaps the 10% decrease was too aggressive.
- As it happens, the R^2 for the final model with only three attributes was only 0.678.



Wrapper-type Feature Selection

Backward Elimination

- It is also evident that the regression coefficients for the two models are different as well.
- The final judgment on what is the right criterion and its level can only be made with experience with the dataset and of course, good domain knowledge.
- Each iteration using a regression model either removes or introduces a variable, which improves model performance.
- The iterations stop when a preset stopping criterion or no change in performance criterion (such as adjusted r2 or RMS error) is reached.
- The inherent advantage of wrapper-type methods are that multicollinearity issues are automatically handled.
- However, no prior knowledge about the actual relationship between the variables will be gained.
- Applying forward selection is similar and is recommended as an exercise.

Thank You!!!