Forecasting Using Model Studio in SAS® Viya®

Slides and Notes

Forecasting Using Model Studio in SAS® Viya® Slides and Notes was developed by Marc Huber, Danny Modlin, Jay Laramore, Terry Woodfield, and Chip Wells. Additional contributions were made by George Fernandez and Ari Zitin.

Forecasting Using Model Studio in SAS® Viya® Slides and Notes

---

Course code FVVF42, prepared date 30Jul2025. FVVF42_001

To learn more…

For information about other courses in the curriculum, contact the SAS Education Division at 1-800-333-7660, or send e-mail to training@sas.com. You can also find this information on the web at http://support.sas.com/training/ as well as in the Training Course Catalog.

For a list of SAS books (including e-books) that relate to the topics covered in this course notes, visit https://www.sas.com/sas/books.html or call 1-800-727-0025. US customers receive free shipping to US addresses.

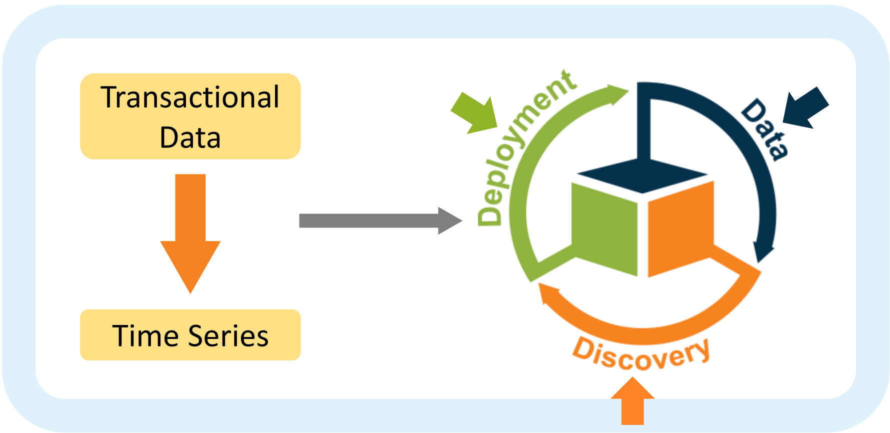
Getting Started with Your Forecasting Project

How to create and get started with your forecasting project

Lesson 01, Section 01

Analytics Life Cycle and Forecasting

SAS Visual Forecasting



Analytics starts with data. For forecasting, the starting point is transactional data. The analytics life cycle represents a series of activities whose goal is to extract value from that data.

We will use the functionality of SAS Visual Forecasting. SAS Visual Forecasting supports all three phases of the analytics life cycle: Data, Discovery, and Deployment.

Loading the data is only the beginning of the tasks that you perform at the Data phase. You might need to clean your data. For forecasting, you might need to accumulate the data into time series. In general, you want to explore and visualize the data. If you find problems with the data, this is the time to correct them. Exploring the data also helps you determine what models might be useful.

At the Discovery phase, you might want to build several models, compare them side by side, and choose a champion that meets the needs of your project. For forecasting, you learn to generate ARIMAX and exponential smoothing models. You select a champion and generate forecasts.

Finally, at Deployment, you put the model to work by using it to score new data.

Time Series Data for the Forecasting Project

Data Set Name	LOOKINGGLASS_FORECAST	**Observations**	53999
Member Type	DATA	**Variables**	23

time series

Variable	Label
Customer_ID	Customer ID
Txn_Month	Transaction Month
productline	Name of Product Line
productname	Product Name
sale	Units Sold
price	Unit Price
discount	Price Discount
cost	Unit Cost
…	…

You will work with a transactional data set named **lookingglass_forecast**. The data set contains more than 50,000 rows. The data consist of information from a telecommunications group whose goal is to predict sales.

This data set is already accumulated into a time series. This means that the observations are time stamped and successive observations are a fixed time interval apart.

Data Set Name	LOOKINGGLASS_FORECAST	Observations	53999
Member Type	DATA	Variables	23

time variable

Variable	Label
Customer_ID	Customer ID
Txn_Month	Transaction Month
productline	Name of Product Line
productname	Product Name
sale	Units Sold
price	Unit Price
discount	Price Discount
cost	Unit Cost
…	…

The data are accumulated monthly with **Txn_Month** as the time variable.

Data Set Name	LOOKINGGLASS_FORECAST	Observations	53999
Member Type	DATA	Variables	23

continuous response

predictors

Variable	Label
Customer_ID	Customer ID
Txn_Month	Transaction Month
productline	Name of Product Line
productname	Product Name
sale	Units Sold
price	Unit Price
discount	Price Discount
cost	Unit Cost
…	…

Potential predictor variables of the continuous response sale are **price**, **discount**, and **cost**.

Data Set Name	LOOKINGGLASS_FORECAST	Observations	53999
Member Type	DATA	Variables	23

BY variables

Variable	Label
Customer_ID	Customer ID
Txn_Month	Transaction Month
productline	Name of Product Line
productname	Product Name
sale	Units Sold
price	Unit Price
discount	Price Discount
cost	Unit Cost
…	…

The data set also contains the attribute variables that will be used in hierarchical forecasting, **productline** and **productname**. You learn more about attributes and hierarchical forecasting later.

HTML Placeholder

Data Dictionary: Time Series Data

You will work with a transactional data set named **lookingglass_forecast**. The data consist of information from a telecommunications group whose goal is to predict sales.

The data set is already accumulated into a time series with **Txn_Month** as the time variable. Potential predictor variables of the continuous response **sale** are **price**, **discount**, and **cost**. The data set also contains the attribute variables that will be used in hierarchical forecasting, **productline** and **productname**.

Variables in LOOKINGGLASS_FORECAST

Variable	Type	Length	Format	Informat	Label
Txn_Month	Num	8	MMDDYY10.	DATE15.	Transaction Date (Month)
productline	Char	8			Name of product line
productname	Char	12			Product name
sale	Num	8			Unit Sale
price	Num	8			Unit Price
discount	Num	8			Price Discount
cost	Num	8			Unit Cost

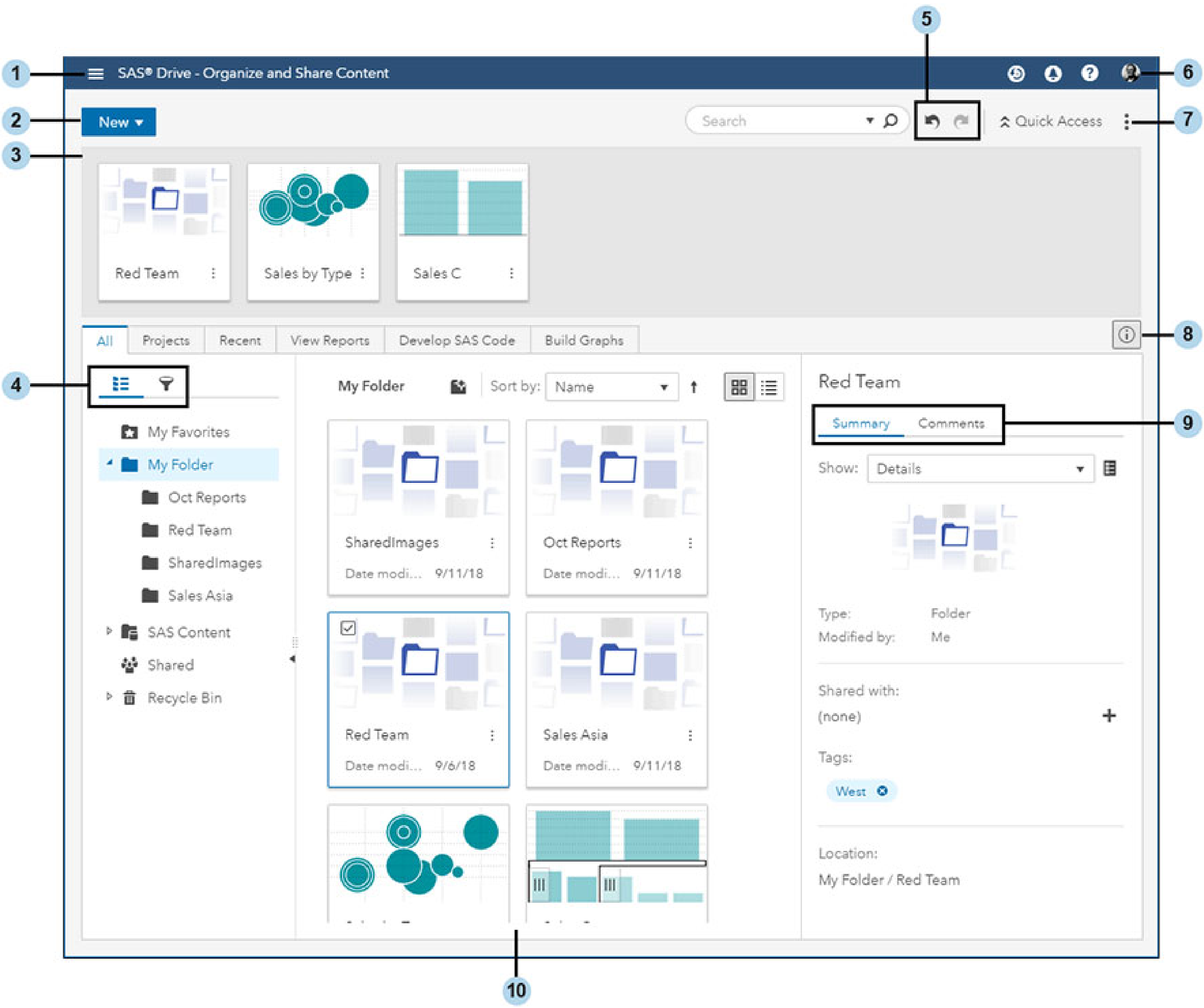

HTML Placeholder
SAS Drive and the Application Menu
sas

SAS Drive

SAS Drive is a common interface for SAS Viya applications, and it enables you to easily view, organize, and share your content from one place. From SAS Drive, you access the features of SAS Visual Forecasting through Model Studio.

To access SAS Drive, you use the standard sign-in window for SAS applications. To display a sign-in window, enter the URL provided by your administrator (for example, https://prod.host.com/SASDrive). **Note:** For this course, the instructions for accessing SAS Drive are included in the virtual lab instructions.

Standard features in SAS Drive include the following:



1. **Applications menu**. This menu appears in all applications that you access from SAS Drive. From any of these applications, you can use the Applications menu to return to SAS Drive (as well as access other applications).
2. **New item button**. Create new folders, links, shortcuts, and uploads.
3. **Quick Access area**. Access your most-used items.
4. **Folders and Filter**. Note: **My Folder** is a shortcut to **/SAS Content/Users/[userID]/MyFolder/**.
5. **Undo and Redo**. Click and hold on either icon to display a list of actions.
6. **Recent items, Notifications, Help, Settings, and Sign out**.
7. **Menu**. Create links or shortcuts, manage tabs, and upload content.

8. **Information pane button**. From the Information pane, view summary information about a selected item, add comments, and share your work.
9. **Summary and Comments tabs**.
10. **Canvas**.

Note: The availability of the features in SAS Drive depends on the applications that are installed and the features and permissions that your administrator has specified.

Applications Menu

The options on the Applications menu are actions that fall within the three phases of the analytics life cycle (Data, Discovery, and Deployment). For example, to access Model Studio from the Applications menu, select **Build Models**. Building models relates to the Discovery phase.

Model Studio, included in SAS Viya, is an integrated visual environment that provides a suite of analytic data mining tools to facilitate end-to-end data mining analysis. The data mining tools supported in Model Studio are designed to take advantage of the SAS Viya programming and cloud processing environments to deliver and distribute analytic model data mining champion models, score code, and results.

Here are other examples:

- To access SAS Model Manager, select **Manage Models**.
- To access SAS Visual Analytics, select **Explore and Visualize Data**. From SAS Visual Analytics, you can access the SAS Visual Statistics add-on functionality, which enables you to use pipelines. In this course, you do not use SAS Visual Analytics and SAS Visual Statistics.

From other applications, you can use the Applications menu to return to SAS Drive.

Note: Remember that access to specific applications is determined by the permissions that are associated with your account.

Demo: Creating a Forecasting Project and Loading the Data

This demonstration illustrates how to create a forecasting project and load the data.

Answer: Key Variables

What are the two key variables needed for a forecasting project?

time variable and BY variable

time variable and dependent variable

BY variable and input variable

attribute variable and input variable

Answer: B - The time variable and dependent, or target, variable are the two key variables needed for a forecasting project.

Visualizing Time Series Using Attribute Variables

How to visualize a time series using attribute variables

Lesson 01, Section 02

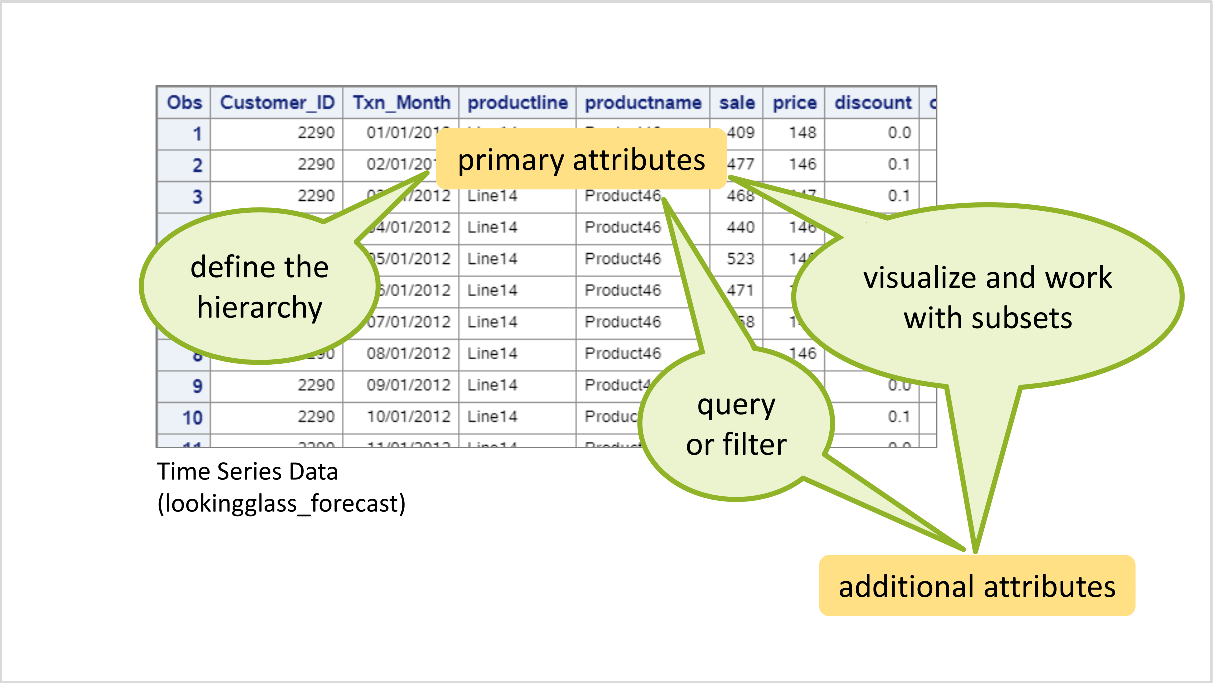
Attribute Variables

BY variables

Obs	Customer_ID	Txn_Month	productline	productname	sale	price	discount
1	2290	01/01/201			409	148	0.0
2	2290	02/01/201			477	146	0.1
3	2290	03/01/2012	Line14	Product46	468	147	0.1
4	2290	04/01/2012	Line14	Product46	440	146	0.0
5	2290	05/01/2012	Line14	Product46	523	146	0.0
6	2290	06/01/2012	Line14	Product46	471	151	0.0
7	2290	07/01/2012	Line14	Product46	458	148	0.1
8	2290	08/01/2012	Line14	Product46	424	146	0.0
9	2290	09/01/2012	Line14	Product46	491	122	0.0
10	2290	10/01/2012	Line14	Product46	369	143	0.1

Time Series Data
(lookingglass_forecast)

When we created the baseline sales forecast project, we defined two variables in the time series data as BY variables: **productline** and then **productname**.



BY variables are the primary attributes of a project.

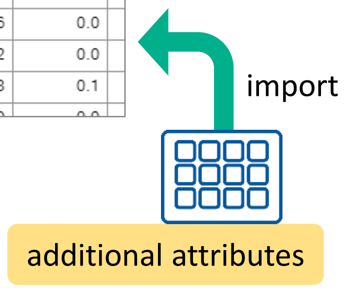
SAS Visual Forecasting uses the primary attributes to define the hierarchical structure of the time series.

A forecasting project can also have additional attributes.

All attributes are characteristics that you can use to query or filter specific subsets of your time series. You can visualize and work with subsets of project data based on specific values of the attributes. Attributes also affect how you can apply overrides or perform other post-modeling tasks.

Obs	Customer_ID	Txn_Month	productline	productname	sale	price	discount	c
1	2290	01/01/2012	Line14	Product46	409	148	0.0	
2	2290	02/01/2012	Line14	Product46	477	146	0.1	
3	2290	03/01/2012	Line14	Product46	468	147	0.1	
4	2290	04/01/2012	Line14	Product46	440	146	0.0	
5	2290	05/01/2012	Line14	Product46	523	146	0.0	
6	2290	06/01/2012	Line14	Product46	471	151	0.0	
7	2290	07/01/2012	Line14	Product46	458	148	0.1	
8	2290	08/01/2012	Line14	Product46	424	146	0.0	
9	2290	09/01/2012	Line14	Product46	491	122	0.0	
10	2290	10/01/2012	Line14	Product46	369	143	0.1	

Time Series Data
(lookingglass_forecast)



It is a common practice to import additional attributes to your forecasting project from another data source.

Obs	Customer_ID	Txn_Month	productline	productname	sale	price	discount
1	2290	01/01/2012	Line14	Product46	409	148	0.0
2	2290	02/01/2012	Line14	Product46	477	146	0.1
3	2290	03/01/2012	Line				
4	2290	04/01/2012	Line				
5	2290	05/01/2012	Line				
6	2290	06/01/2012	Line				
7	2290	07/01/2012	Line				
8	2290	08/01/2012	Line				
9	2290	09/01/2012	Line				
10	2290	10/01/2012	Line				

Time Series Data
(lookingglass_forecast)

Obs	productline	productname	Cust_Region	margin_cat
1	Line01	Product01	South	LOW
2	Line01	Product02	South	MED
3	Line01	Product03	Great Lakes	LOW
4	Line02	Product04	Greater Texas	LOW
5	Line02	Product05	Greater Texas	LOW
6	Line02	Product06	South	LOW
7	Line02	Product07	Pacific	LOW
8	Line03	Product08	Pacific	LOW
9	Line03	Product09	Greater Texas	HIG
10	Line03	Product10	Southwest	LOW

Attribute Data
(lg_attributes)

For the baseline sales forecast project, we have an additional data set, named **lg_attributes**, that contains additional attributes, **Customer Region** and **Margin Category**.

Imported attributes data must contain a row for each unique combination of the primary attributes, or BY variables, that are defined in the time series data.

Demo: Loading an Attributes Table to Subset the Time Series

This demonstration illustrates how to load an attributes table and how to visualize the modeling data.

Answer: Attribute Variables

What are attribute variables primarily useful for?

visualizing and post-processing

loading the data

accumulating the data

defining the hierarchy

Answer: A - The attribute variables are useful for visualization and post-processing.

Practice

Creating a Project and a Visualization


Questions?

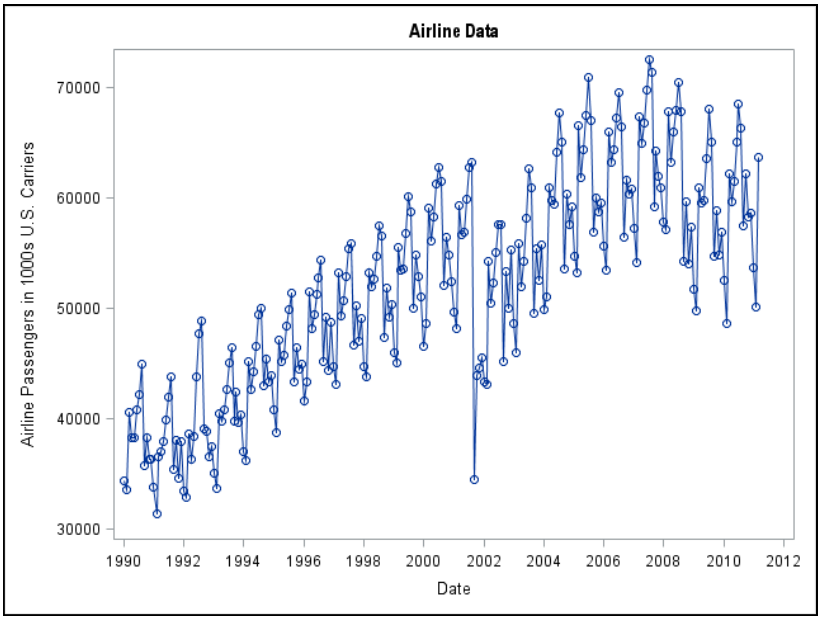
Definition and Creation of a Time Series

How to define and create a time series

Lesson 02, Section 01

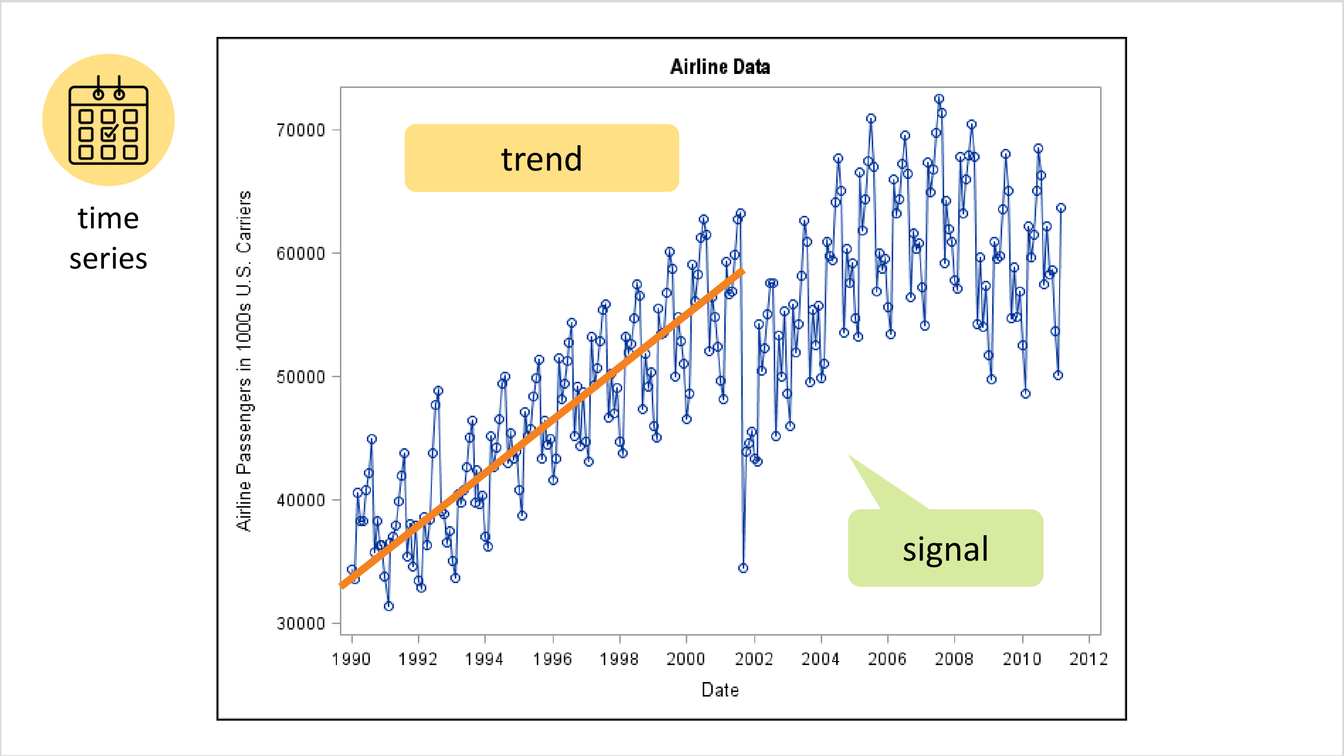
Time Series Creation

time series

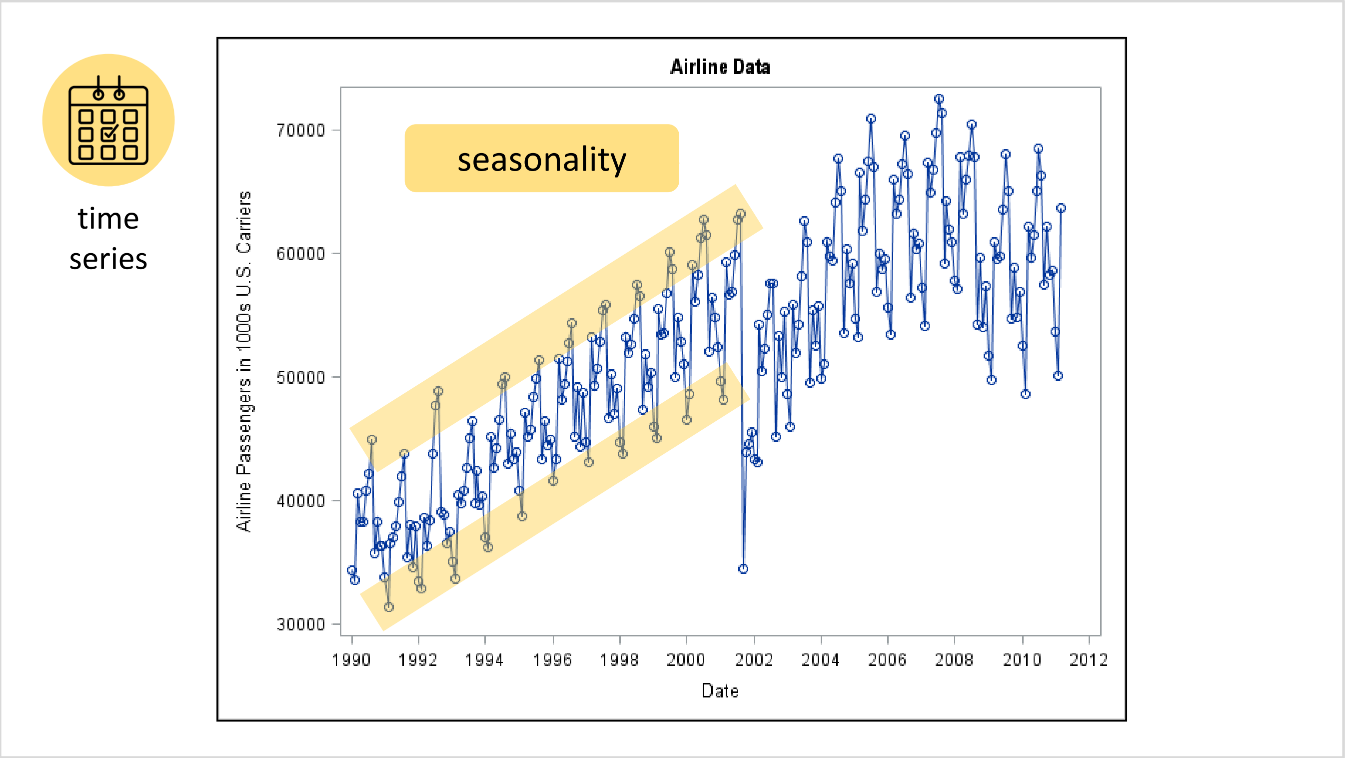


Now, let's look at a time series. A time series is simply an indexed set of equally spaced measurements, where the sequence is related to time.

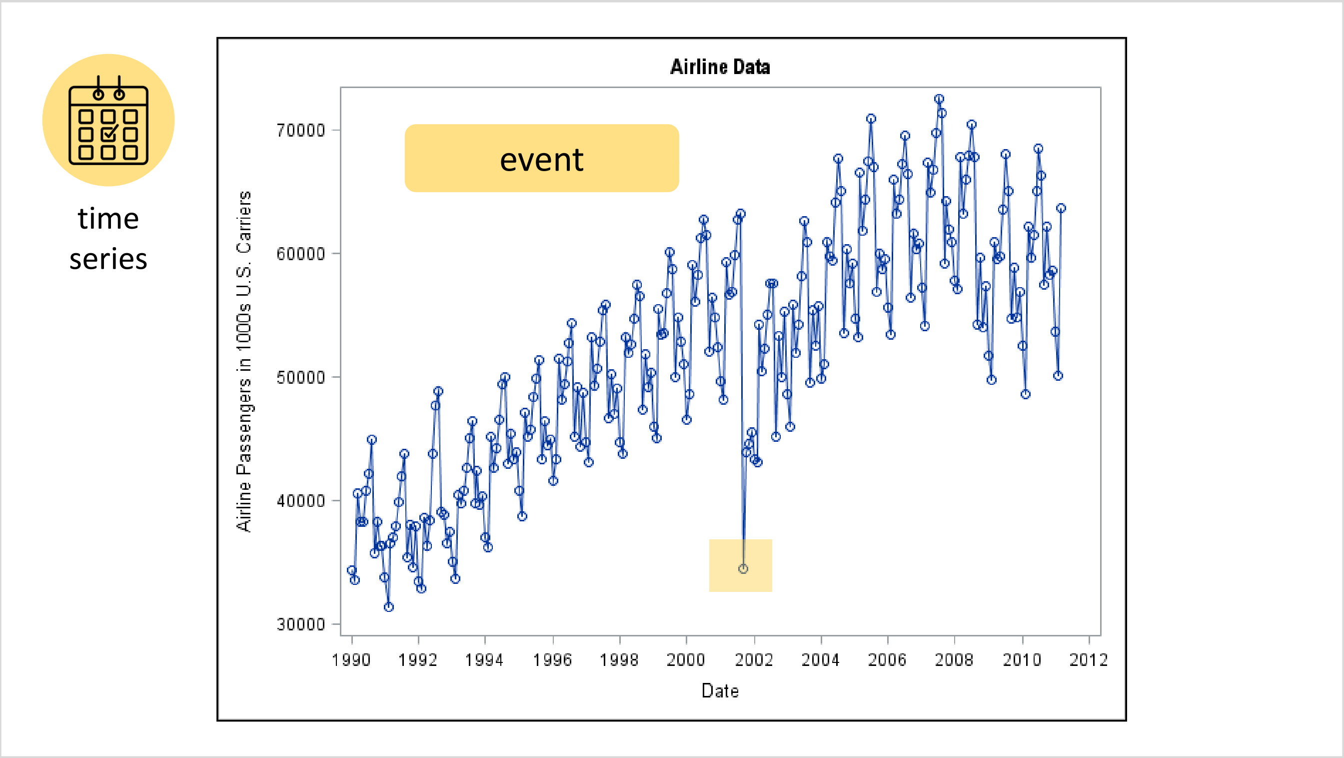
This particular plot displays a time series similar to one you might see if you opened a time series textbook. In the airline data time series displayed, the data points represent the number of passengers that flew on airplanes in the United States per month from 1990 to 2012. The equally spaced measurements, with respect to the date index, define these data as a time series. In this case, the chosen interval of the index is monthly. The airline data time series is considered textbook time series forecasting data because it has many attributes of a times series that you might want to model.



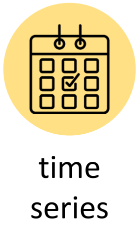
The data have systematic variation, often referred to as signal. In this example, the systematic variation includes a long-term trend, almost a straight-line linear *trend*, from 1990 through 2001.

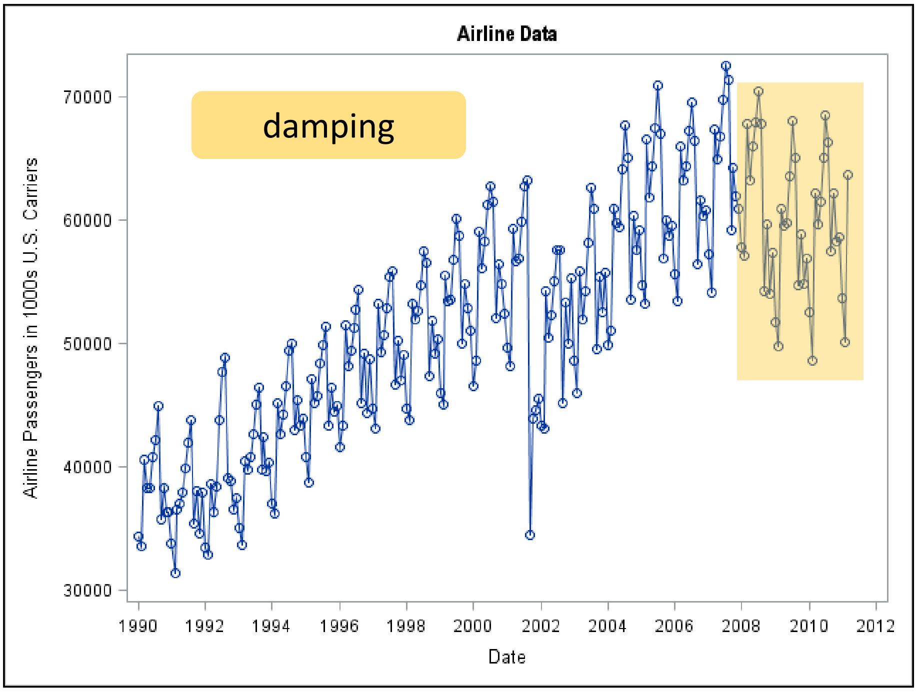


The airline data time series has *seasonality*, which can be thought of as a regular departure from the trend at a particular indexed period of a seasonal cycle. In this example, there are consistent August peaks and February troughs.



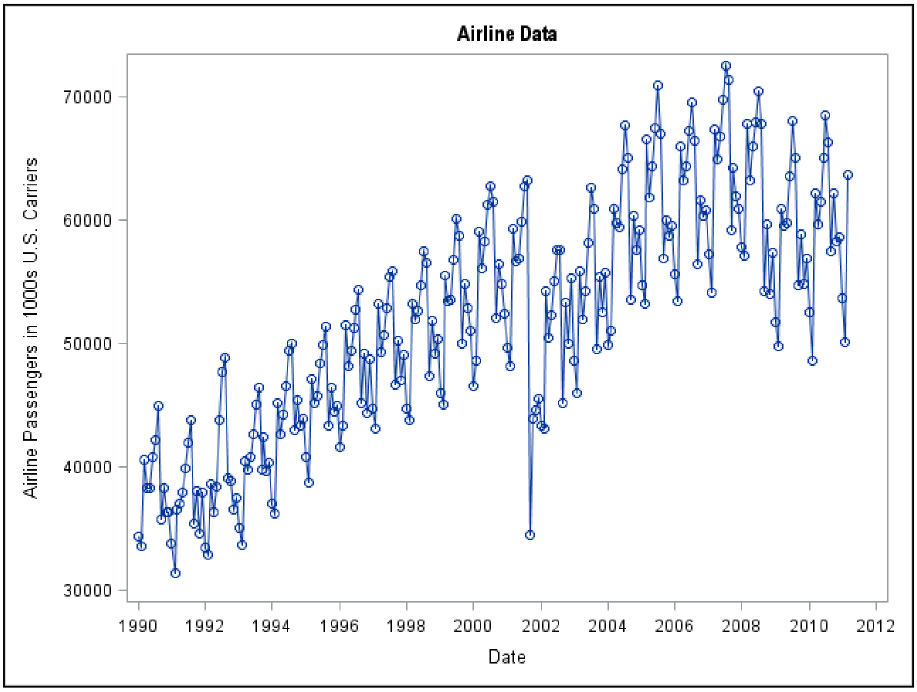
The data have at least one *event*, which is a circumstance that causes a sudden departure from other regular influences, such as trend and seasonality. In September 2001 and, to a lesser degree, in some other months, you see a sudden departure from the generally upward linear trend, modified by the annual cyclical pattern.





The series also shows evidence of *damping*, which is a slow and long-term reversal of a trend. In these data, damping starts at about the middle of 2007. The passenger numbers start to plateau year to year and even seem to slightly decline. This damping could be due to a ticket price effect or an oil price effect. You need to investigate to explain why this might happening.

time
series



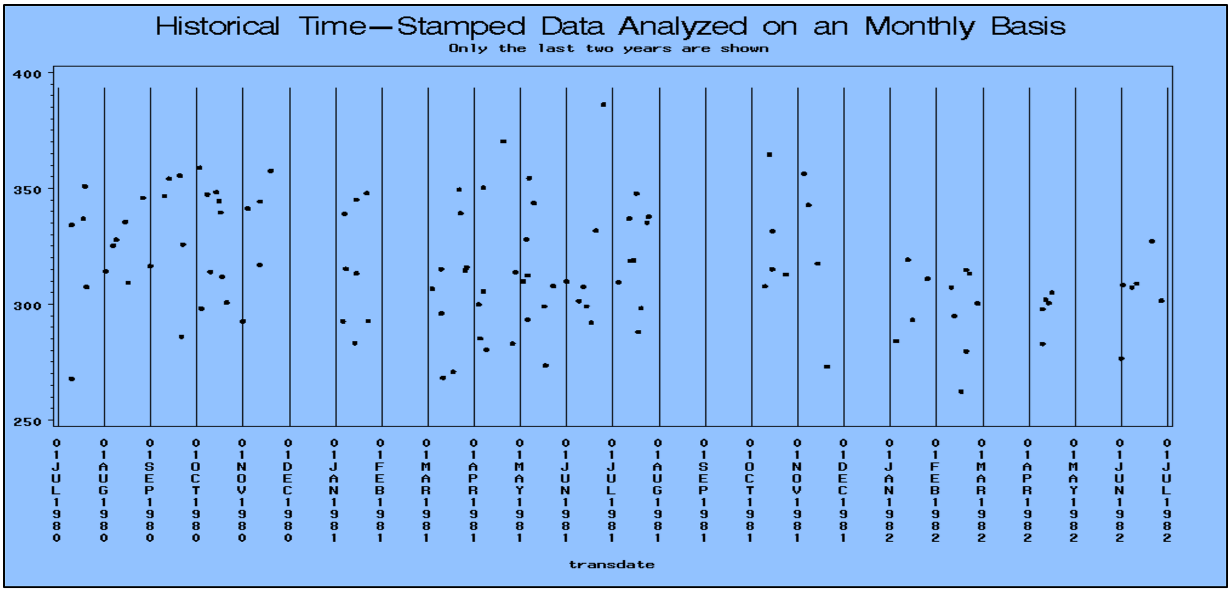
It is important to remember that what defines a time series is the index. In this case, the time series might have missing values for passengers, but it cannot have a missing value for the time index and still be considered a time series.

website visits

Date	Website Visits
10/3/1981	11
10/8/1981	14
10/12/1981	7
10/19/1981	17
10/23/1981	5
11/01/1981	12
11/09/1981	8
11/14/1981	13
11/20/1981	15

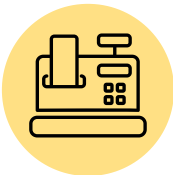
transactional data

In business applications, time series usually start as transactional, or timestamped, data. An example of transactional data is a record of customer visits to a website over a period of a year, where each visit is recorded with a customer identifier and a timestamp. Transactional data are not organized with respect to a time interval. They must be made equally spaced, or indexed, before time series models can be used to quantify the systemic variation contained in the data.



To create time series data from transactional data, you must accumulate transactional data to a specified time interval of the data. *Accumulation* is defined as the process of indexing, or transforming transactional data into a time series. After you choose your time index and an interval, you must decide how to summarize the data that share the same interval value.

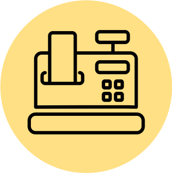
restaurant
sales



Transaction ID	Date	Cost
123000113	02/27/2019	$13.54
123000114	02/27/2019	$13.26
123000115	02/27/2019	$12.99
123000116	02/27/2019	$13.23
123000117	02/27/2019	$14.01
123000118	02/27/2019	$13.08
123000119	02/27/2019	$14.13
123000120	02/27/2019	$13.87
123000121	02/27/2019	$13.34

For example, let's consider transactional data from sales at a restaurant.

restaurant sales

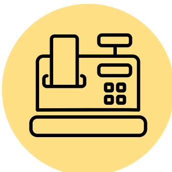


Transaction ID	Date	Cost
123000113	02/27/2019	$13.54
123000114	02/27/2019	$13.26
123000115	02/27/2019	$12.99
123000116	02/27/2019	$13.23
123000117	02/27/2019	$14.01
123000118	02/27/2019	$13.08
123000119	02/27/2019	$14.13
123000120	02/27/2019	$13.87
123000121	02/27/2019	$13.34

?

You might decide to use a daily interval as your time index. How will you represent sales for the 27th of February of 2019? You have thousands of register transactions from different times throughout this particular day. How will you represent all of the transactions with one value?

restaurant sales



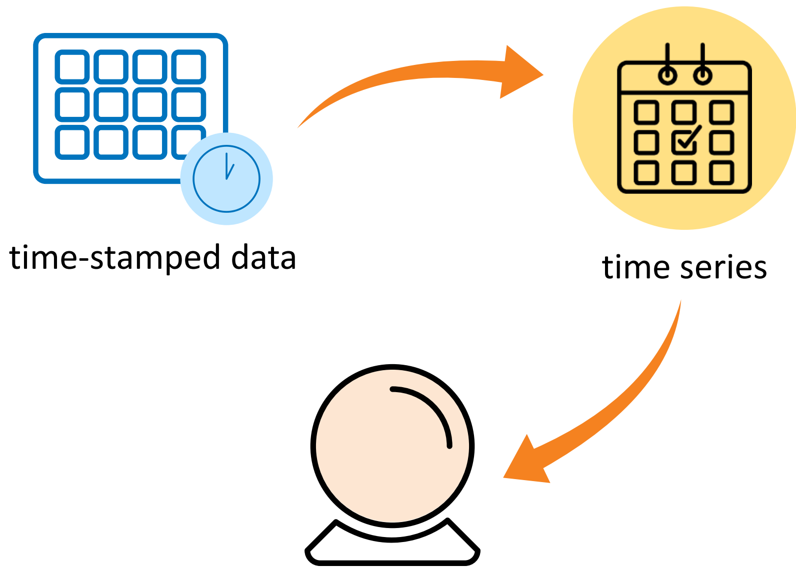
Transaction ID	Date	Cost
123000113	02/27/2019	$13.54
123000114	02/27/2019	$13.26
123000115	02/27/2019	$12.99
123000116	02/27/2019	$13.23
123000117	02/27/2019	$14.01
123000118	02/27/2019	$13.08
123000119	02/27/2019	$14.13
123000120	02/27/2019	$13.87
123000121	02/27/2019	$13.34

sum

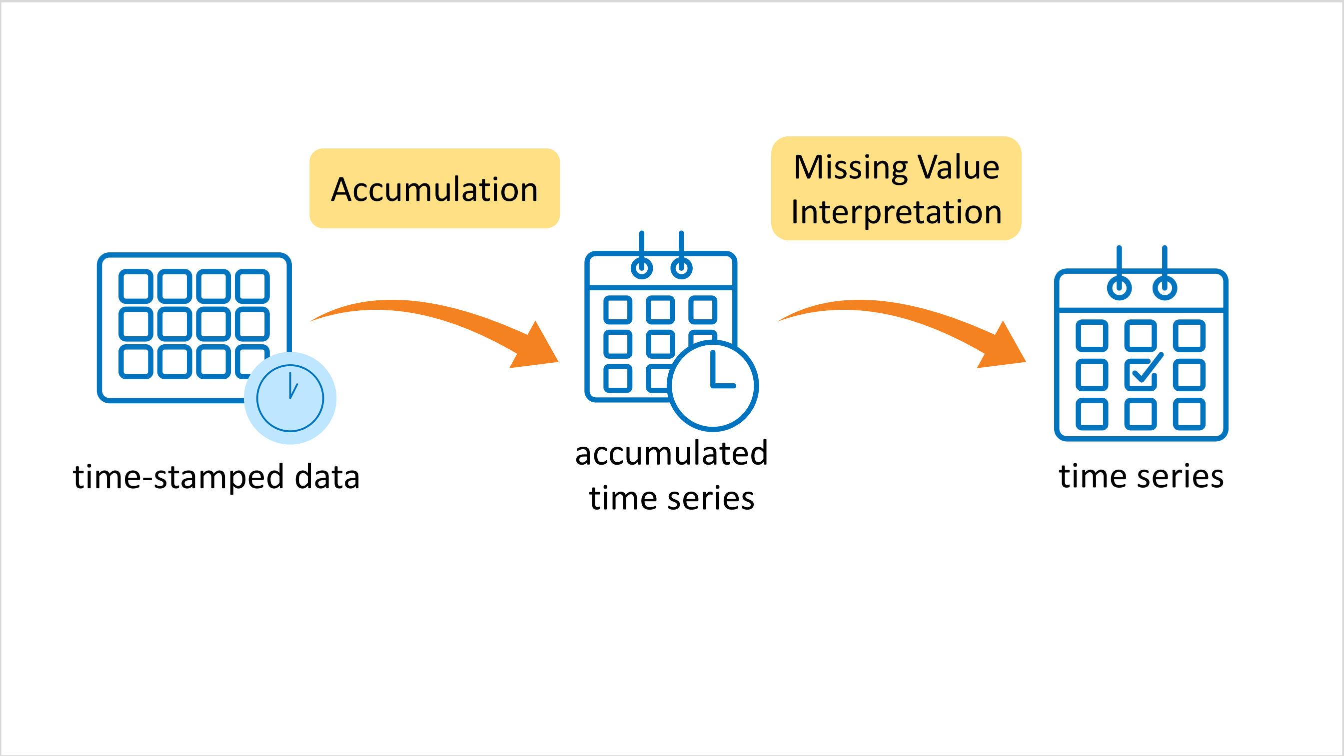
average

You might choose to represent the day with a sum of all transactions.

Or, instead, you might choose to represent the day with the average cash value of each transaction. There are many methods of accumulation, only a few of which we discuss.



Let's examine the process of converting timestamped data to time series data so that you can perform time series analysis and forecasting.



You begin with raw timestamped data, such as transactional data from sales at a restaurant.

Next, you must decide on the length of the time interval and the accumulation method that you'd like to use. Different accumulation choices can emphasize different systemic characteristics of the data, and impact model usefulness and precision.

After you make your choices, you accumulate the data into a time series. Then you have to deal with missing values for intervals with missing data. Should intervals with no data have series values set to zero? Although a zero value might be the correct value for those intervals, imputing a zero value for missing data in an interval might not give you the best model. The data might be an outlier that is not expected to repeat. For example, your restaurant might close for a day due to a power failure as a result of a large storm, causing missing values.

The time series data are ready to analyze after the missing values are imputed.

Answer: Time Series Creation Process

Which of the following steps must come first in the process of creating a time series for modeling?

Zero value interpretation

Missing value interpretation

Accumulation

Outlier detection

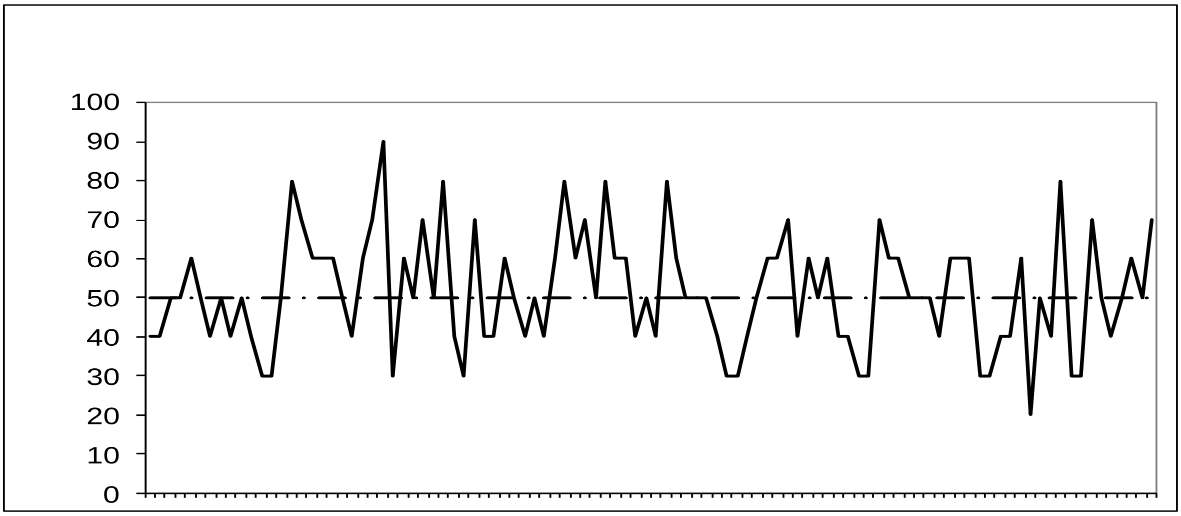
Answer: c - Accumulation must come first.

Fundamental Concepts in Time Series Modeling

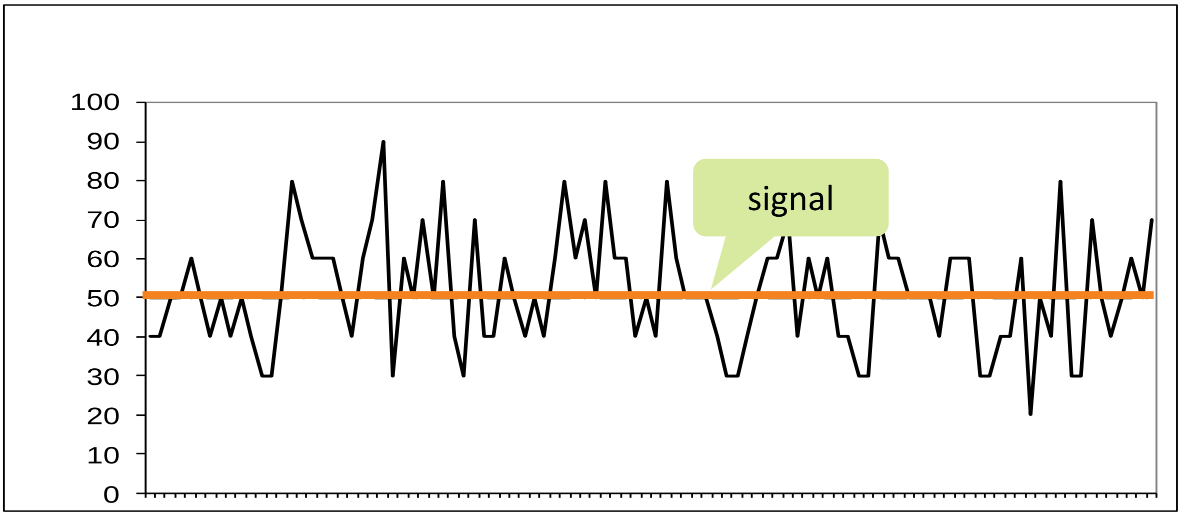
How to understand fundamental concepts in time series modeling

Lesson 02, Section 02

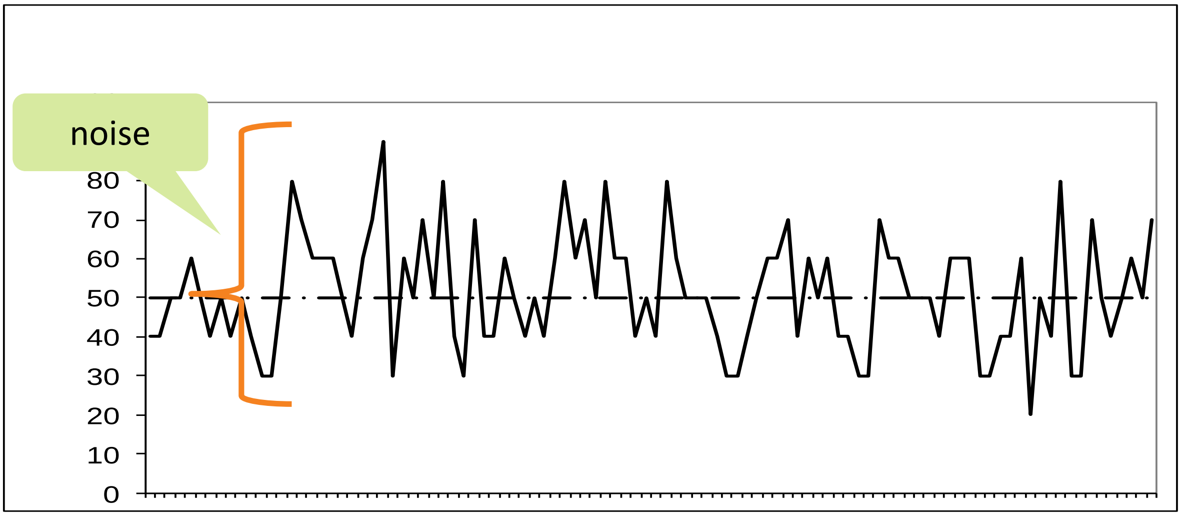
Variation in Time Series Data: Signal and Noise



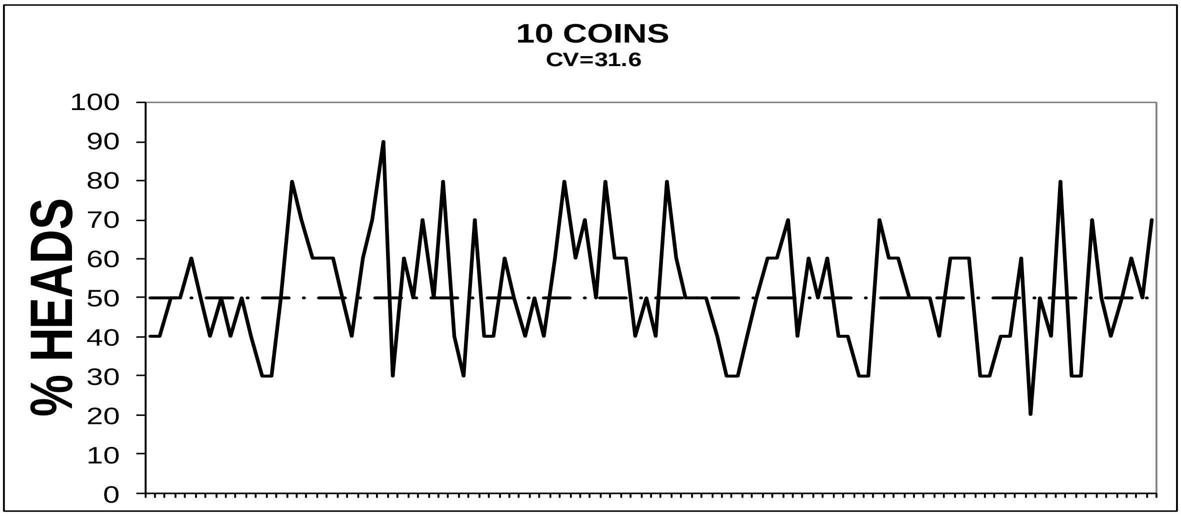
Now, you are ready to model your data. But first, we need to step back and talk about a fundamental concept in time series analysis: signal versus noise.



You can think of a signal in data as an identifiable and repeatable pattern. Signal can include things like an upward slope in average daily calls for the call center of a company that has been gaining more customers, or the relative increases in swimwear sales during the warmest months of the year. Patterns that are systematic and repeatable can be modeled and used to forecast future behavior of the series.



On the other hand, noise is a pattern in data that is not systematic and therefore not replicable. Noise is driven by what is called a *stochastic process*, one guided by randomness.



A time series that consists of only noise is often referred to as a *white noise series*. A white noise series is a series of independently, identically distributed random variables. Each distribution has a mean of 0 and the same finite variance. A white noise series from an underlying “white noise process" contains, by definition, no signal. The forecasted value for such a series, at any time point, does not change, and it is generally calculated to be zero. Where there is no signal, there are no patterns to help produce a better forecast than that.

Answer: Data Differences

What is the difference between time series data and transactional data?

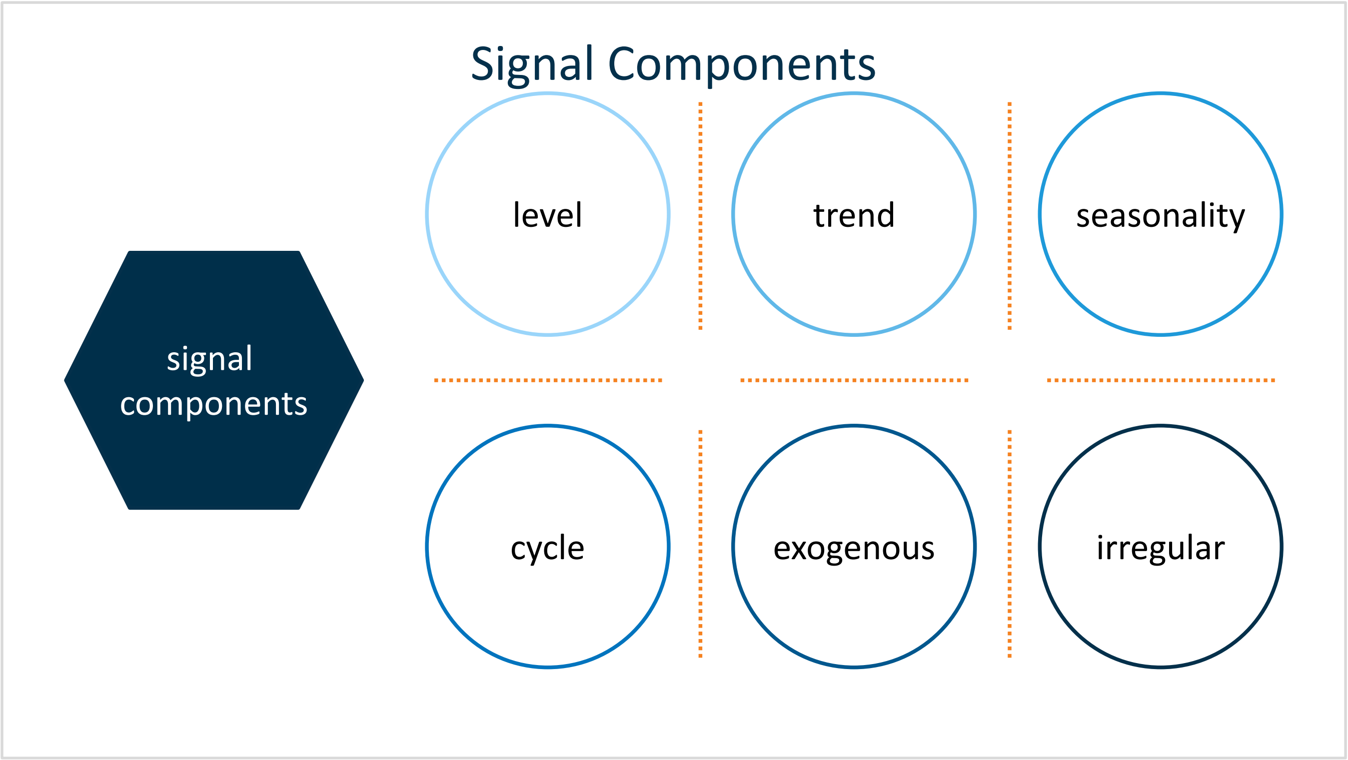
Time series data are sequential, transactional data are not.

Time series data contain explanatory variables, transactional data do not.

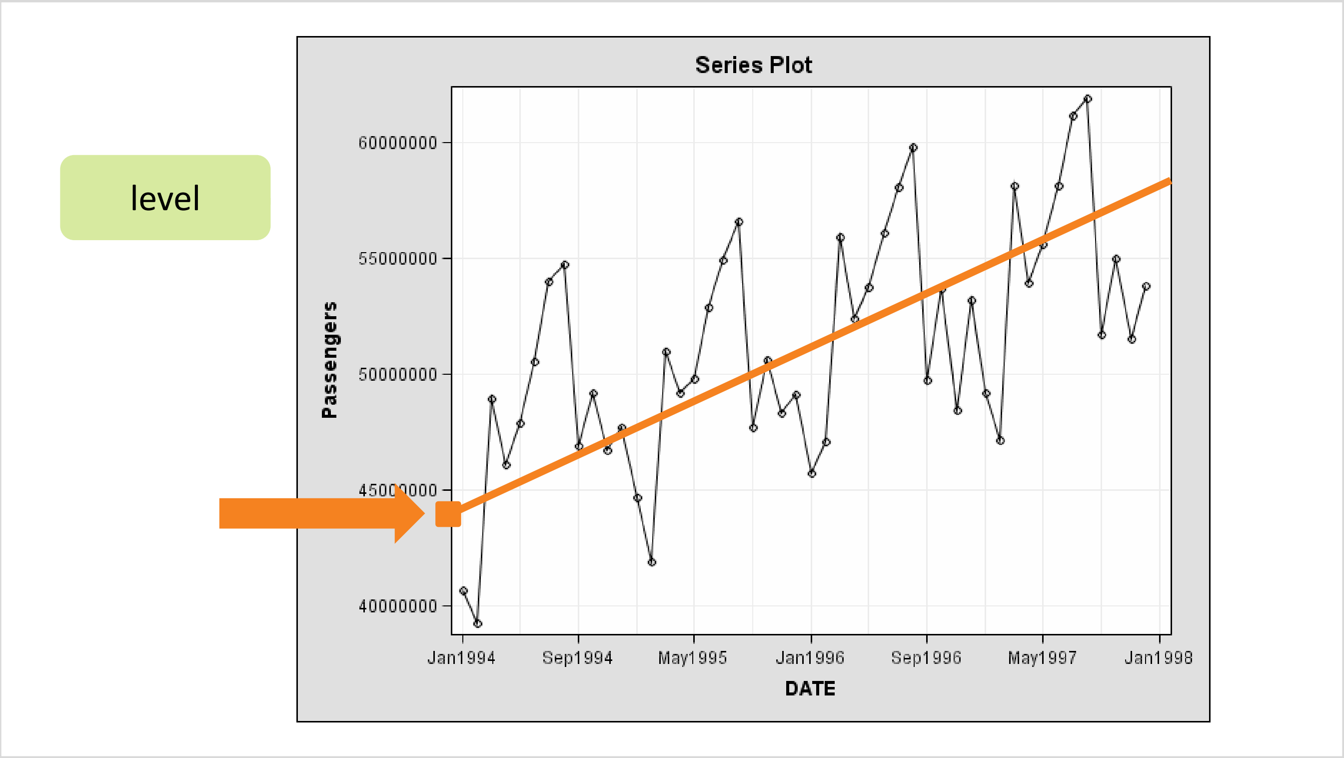
Time series data have trend, transactional data do not.

Time series data are equally spaced with respect to an index, transactional data are not.

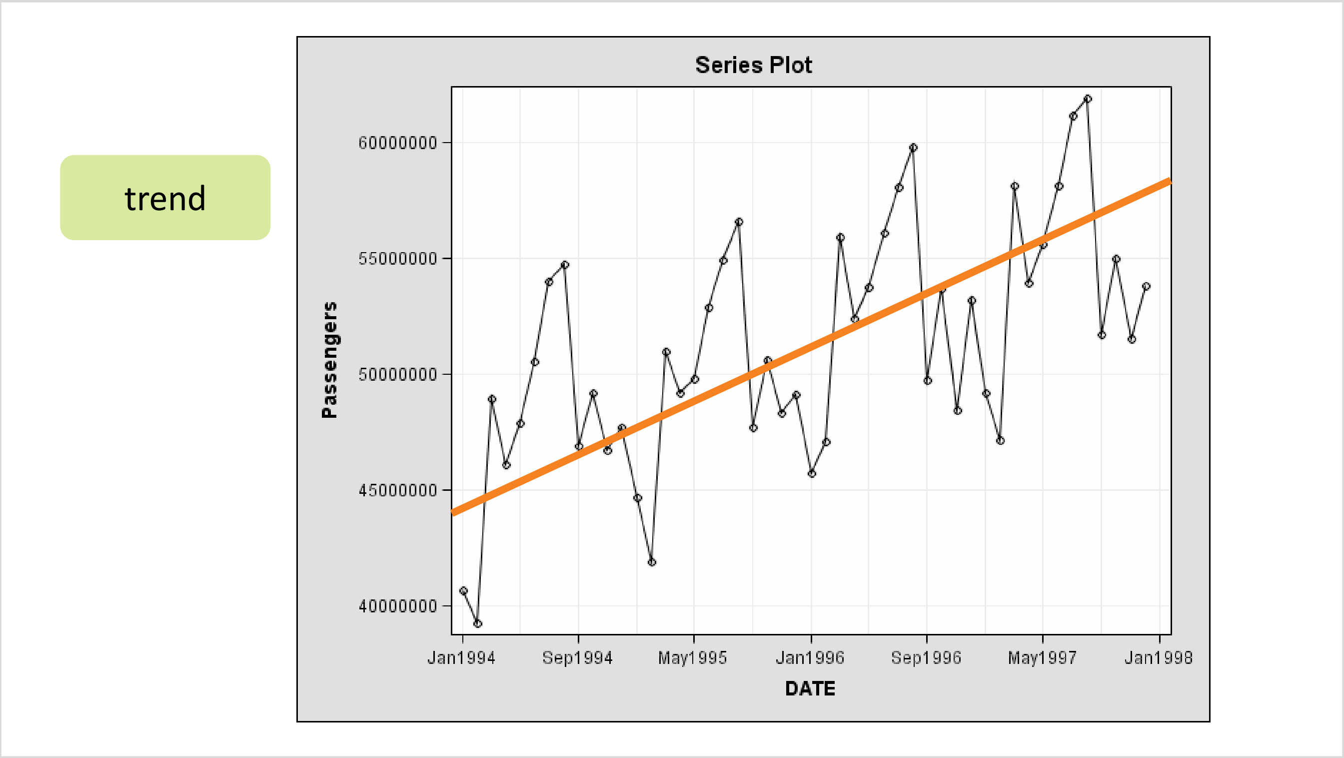
Answer: D - Time series data are equally spaced with respect to an index. Transactional data do not have to be equally spaced with respect to an index.



Conceptually, signal can consist of many different components , whereas noise cannot. The primary components of signal in a time series are level, trend, seasonality, other seasonal cycles, effects of exogenous series, and localized components, referred to as *irregular* components.

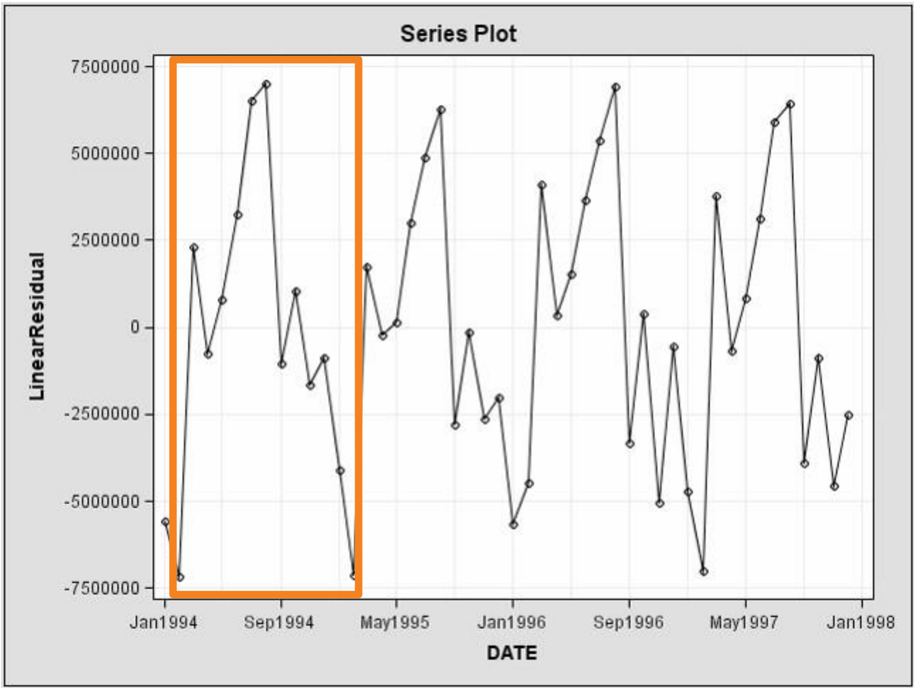


The first, and most basic, signal component is the series level. Level is the nonzero value of the series at a particular point in time. In some ways, level is like the intercept of a regression model, but the level itself can be a function of time.



Trend is the long-term pattern of the mean of a series. Trend can be linear, analogous to the slope in a regression equation. For example, the trend in the airline passenger data seems to be a generally upward-sloping, straight line. Trend doesn't have to be linear.

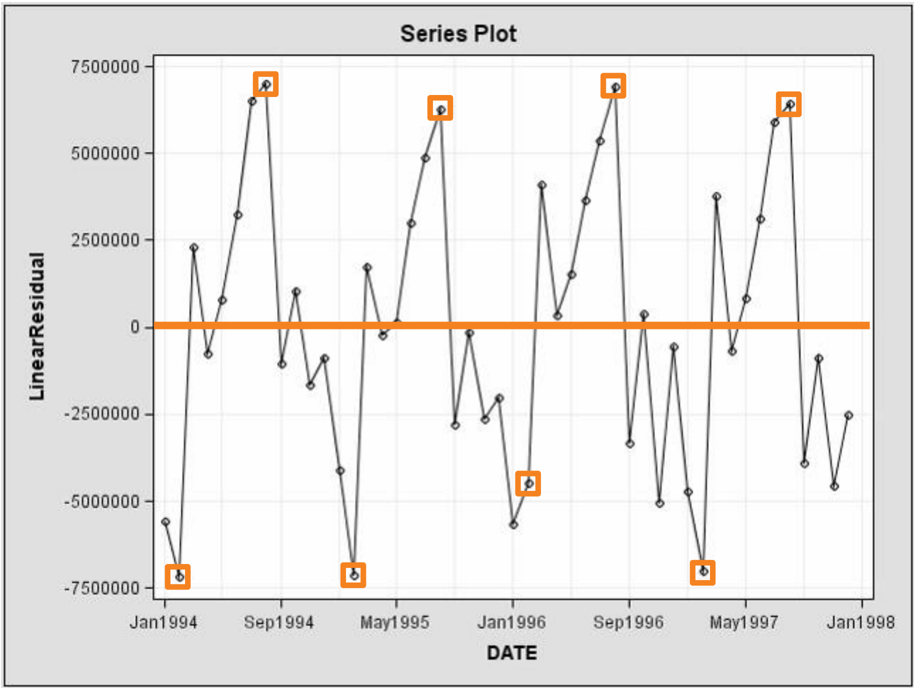
seasonal decomposition



Seasonality patterns can be thought of as predictable deviations of the series patterns from the trend component. They can be either smooth cycles or recurring effects of a season within a seasonal cycle.

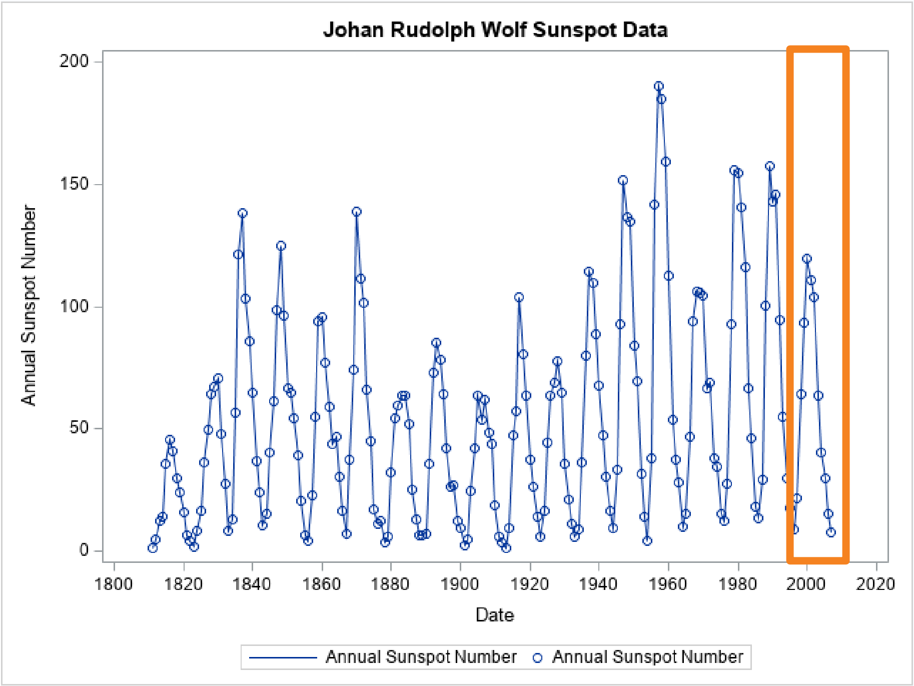
A seasonal cycle for the airline passenger data is 12 months. Certain patterns seem to occur every 12 months.

seasonal decomposition



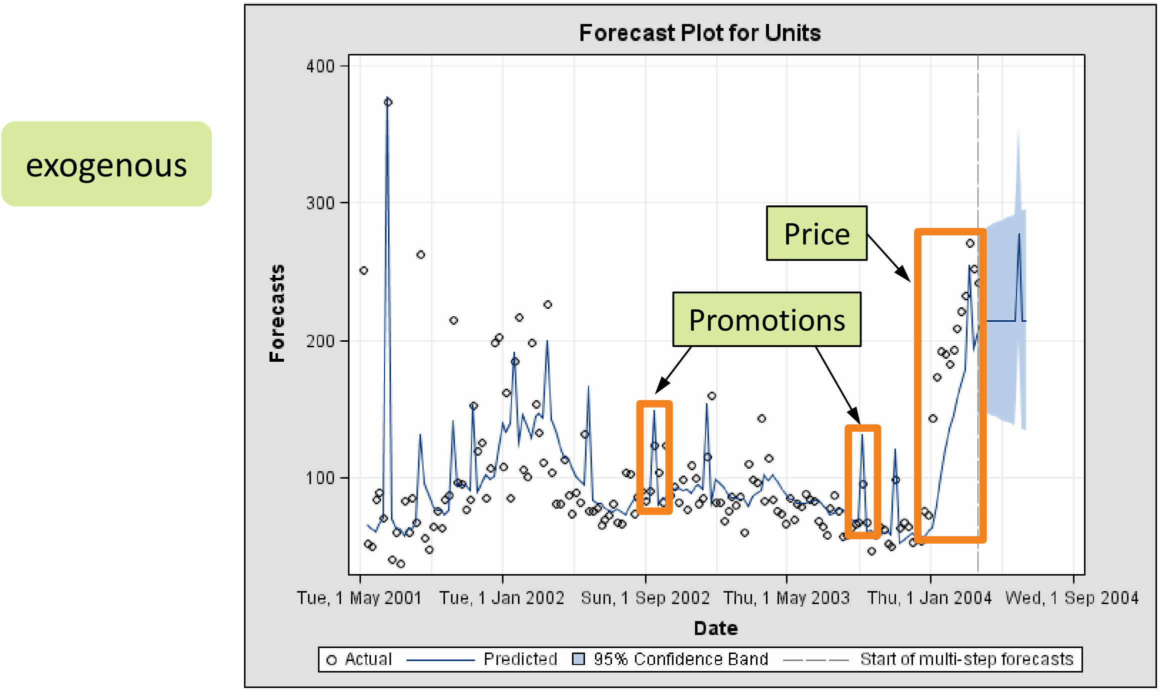
For example, every August the value of the series appears to be about 6.5 million passengers above the value predicted by the trend. Each February, the value of the series is about 7 million passengers below the trend line.

cycle



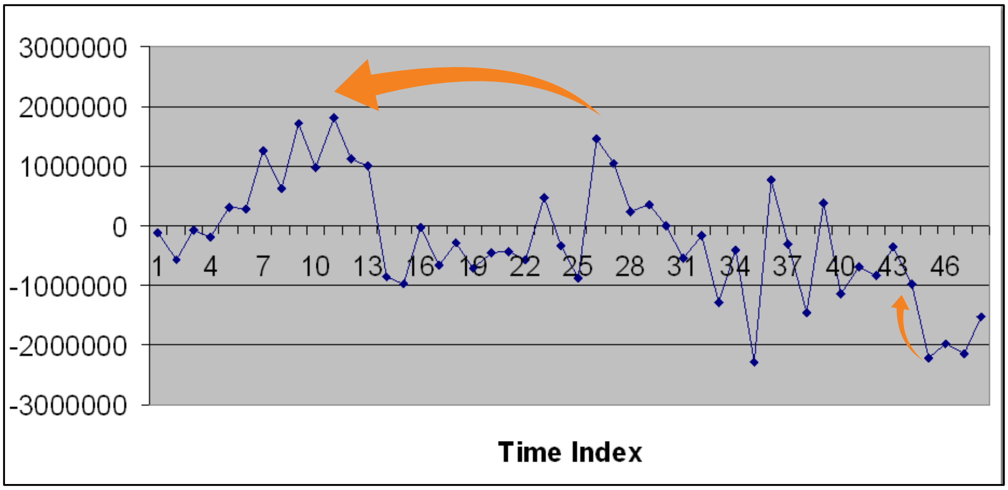
A longer trend cycle, or a cycle whose period varies, can also be a signal component. Examples can be found in economic data. The value of a particular currency tends to flow upward and downward, but not at regular intervals or intervals defined by yearly cycles.

It is important to note that cycle and seasonality are often confused. Seasonality refers to a regular, repeating cycle, whereas cycle is more general.



The effect of other series, external to the one of interest, is often referred to as an *exogenous effect*. For example, a series of sales data might spike at times when a particular sales promotion is offered. Another example of an exogenous effect is when sales start to rise during a time when the price of the product is reduced. The relationship with series external to sales forms the exogenous component of signal for the sales series. Exogenous effects are sometimes known as *explanatory effects*.

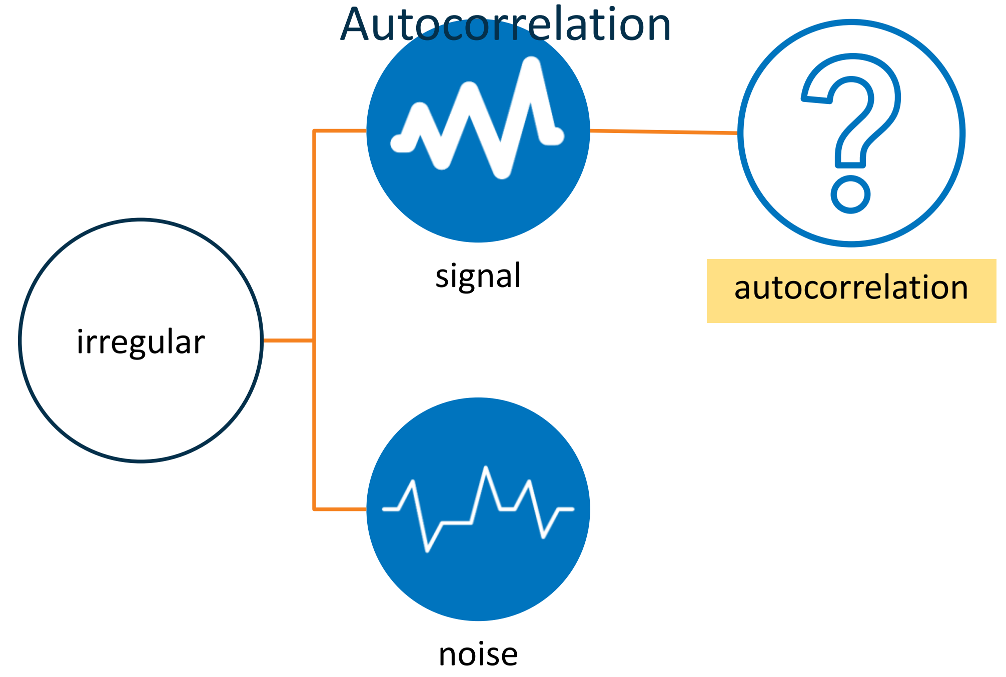
irregular



After you account for all of the previously discussed components and effects, you are left with the irregular component of a series. Some of the irregular component might still be part of the signal of the series, but the rest of it is likely just noise. The noise in the irregular component of the series is often referred to as *white noise* and is very similar to the error term in a regression model.

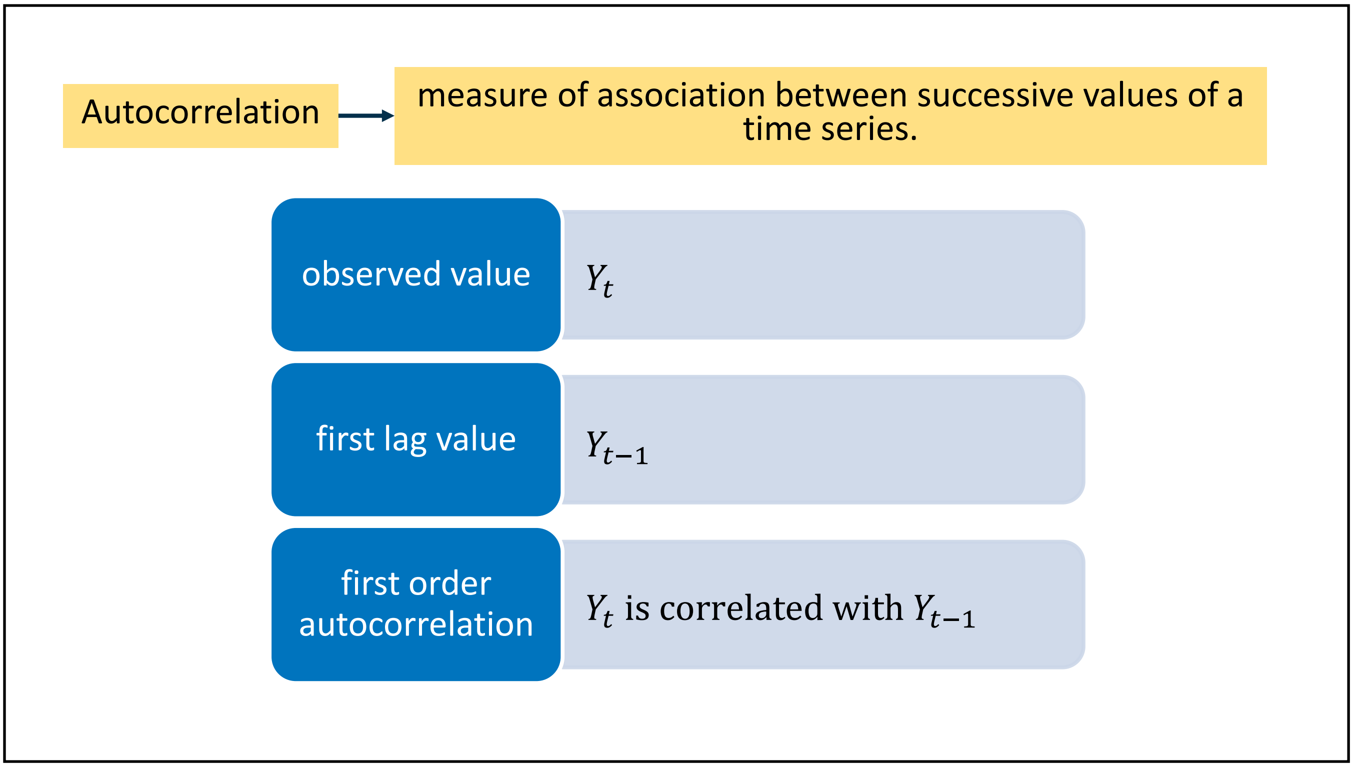
The signal part of the irregular component of a series distinguishes itself from other parts of the signal in that its pattern is local, rather than global, and it consists of short-term patterns, rather than long-term patterns.

Autocorrelation

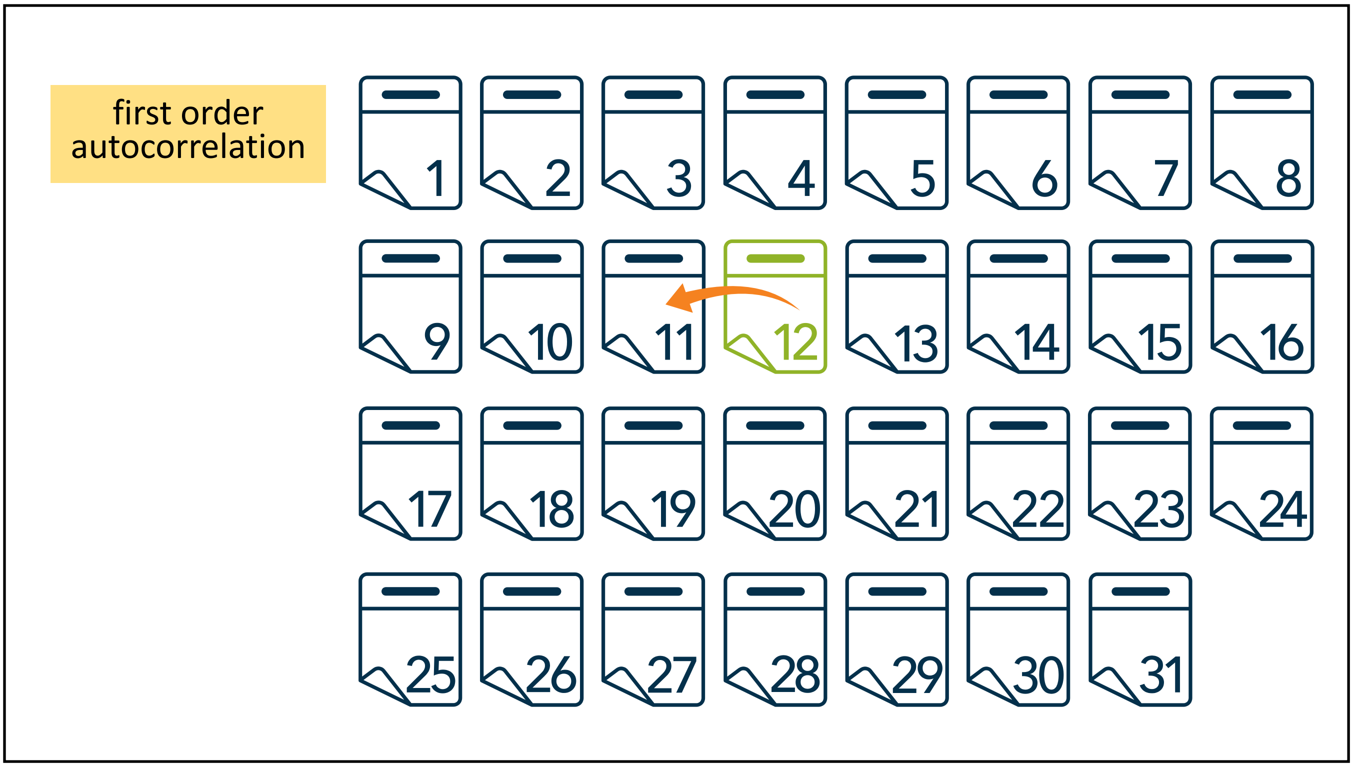


You now know that the irregular part of a series contains both noise and signal. What does the signal part of the irregular component of the series consist of?

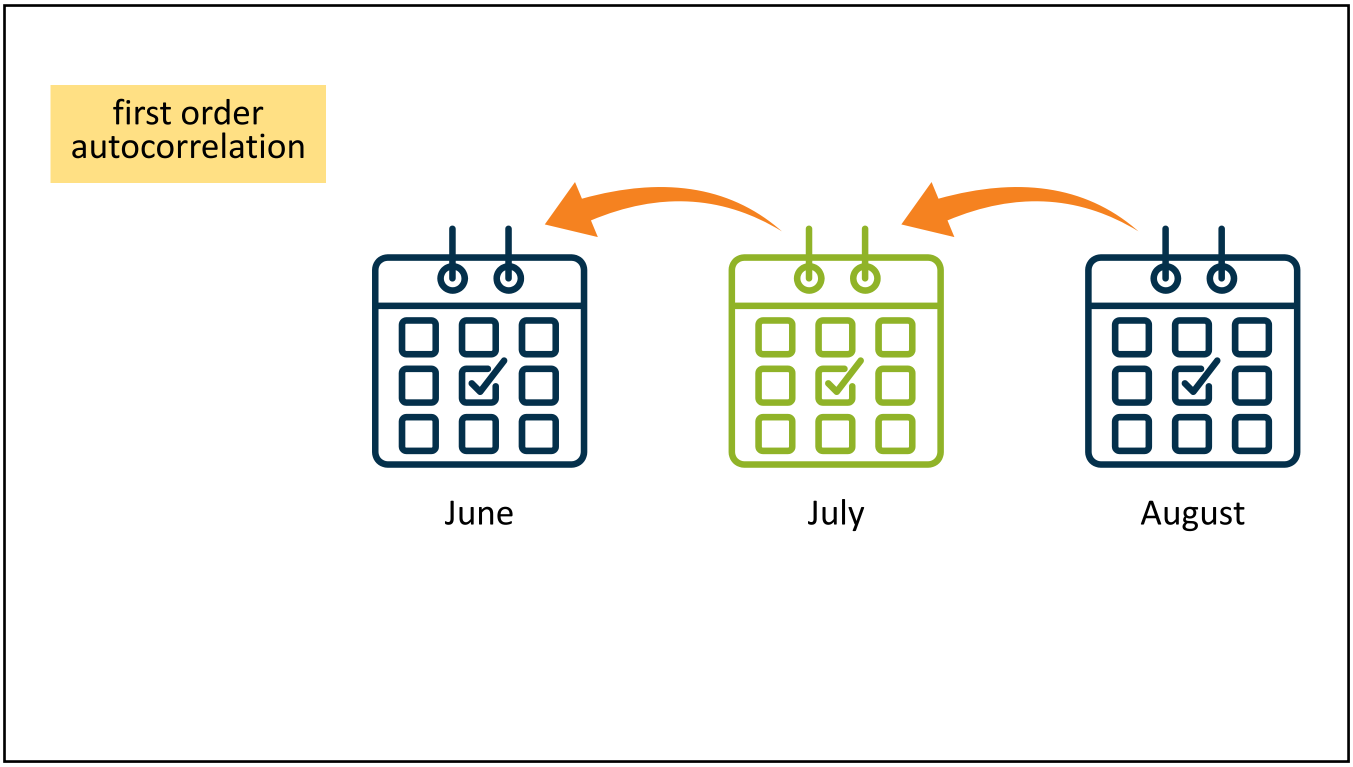
Autocorrelation is part of the signal in the irregular component.



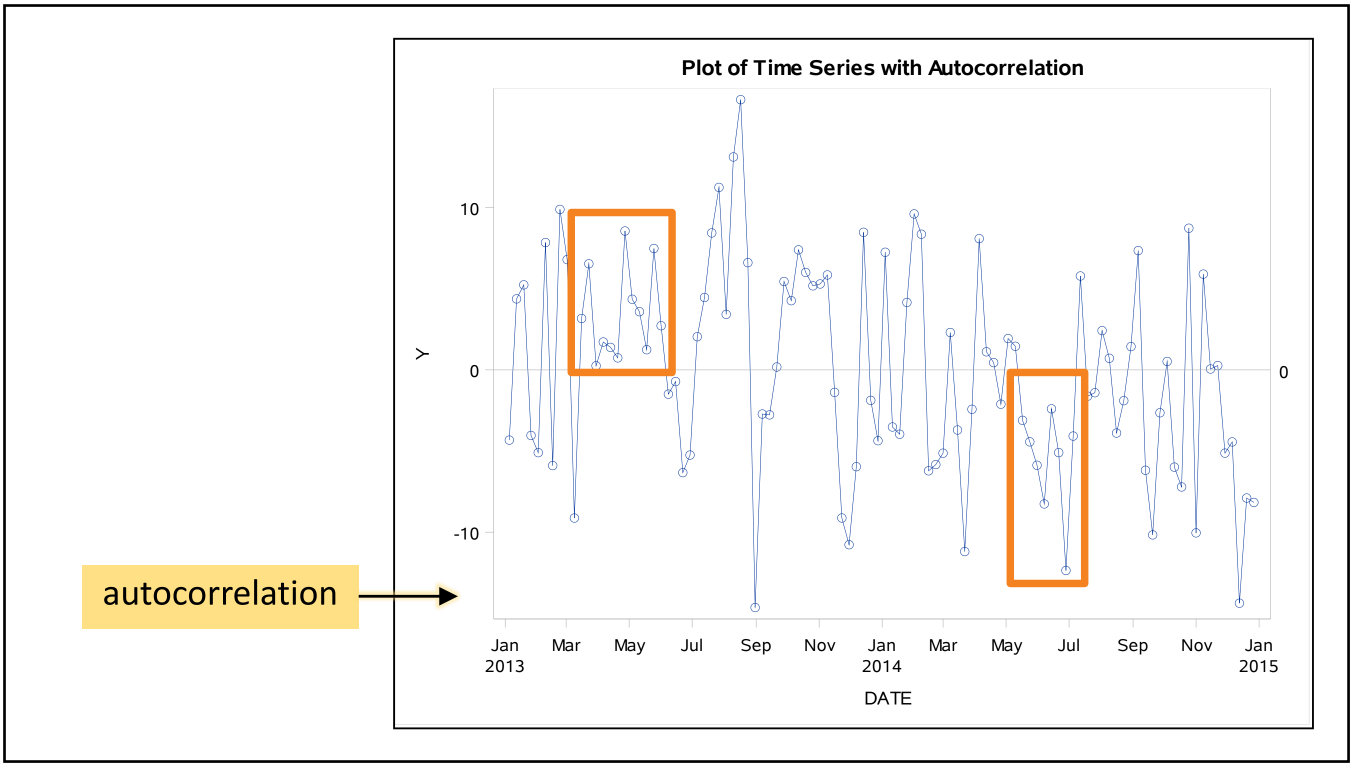
Autocorrelation is a measure of association between successive values of a time series. Series values immediately preceding current values are referred to as *lagged* values. The notation for an observed value, *Y*, as a function of time, *t*, is simply Y_t. Similarly, the first lag value, which is *Y* as a function of *t* minus 1 is represented as Y_{t-1}. The correlation between the current value and the first lag value is called *first-order autocorrelation*.



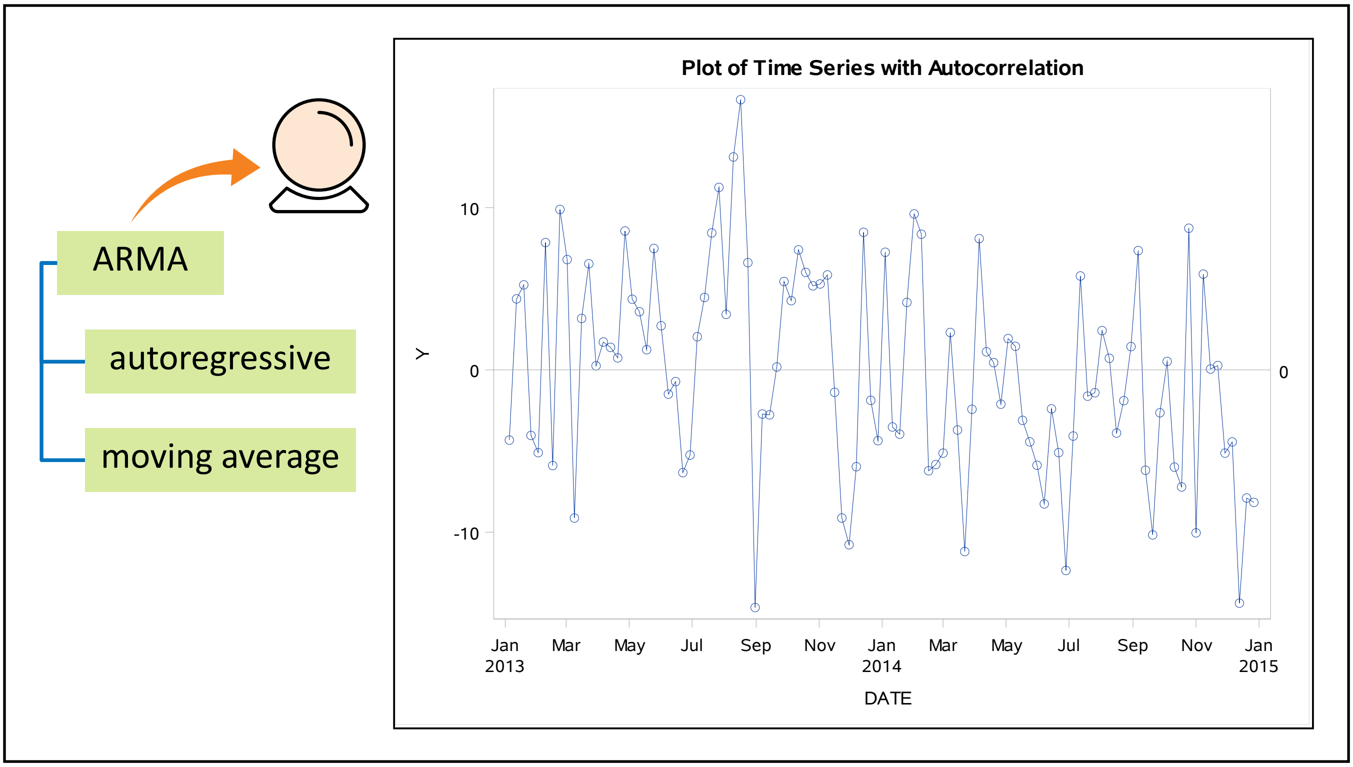
For daily series values, first-order autocorrelation indicates that today's observed value is correlated with the previous day's value.



In monthly series data, first-order correlation means that July is correlated with June, August is correlated with July, and so on.



The way autocorrelation manifests itself is that if the series has a value above the mean, where the mean in our example is 0, values tend to stay above the mean for a while. After values fall below the mean, future values tend to stay below the mean for a while. This differs from behavior of random data, where the probability of an above-mean value is unrelated to the occurrence of an above-mean value at previous times.

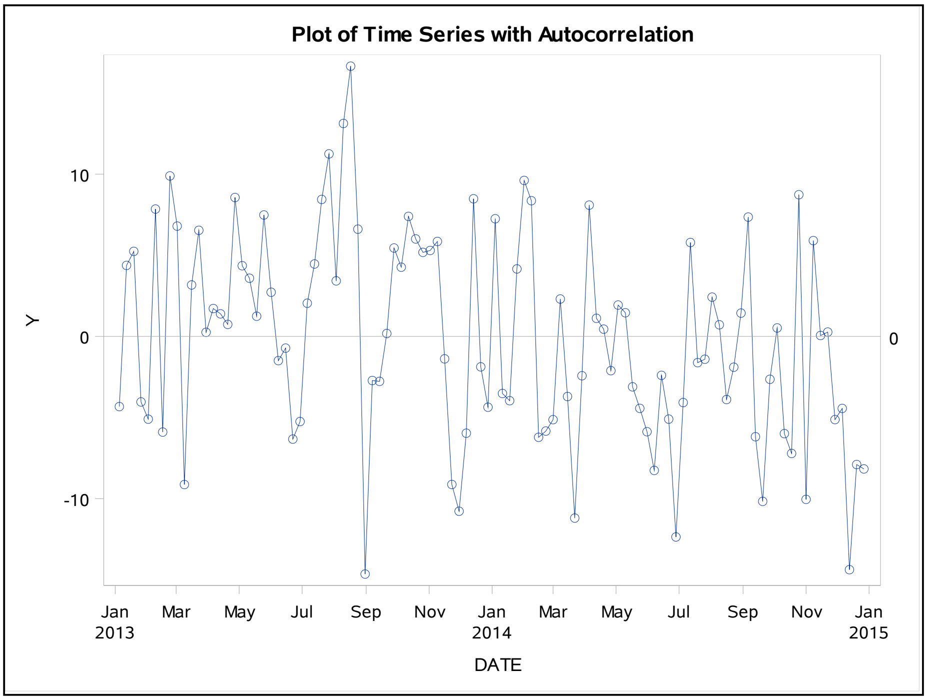


Modeling autocorrelation is done using autoregressive models. Autoregressive models, and their partners, moving average models, make up what is referred to as *ARMA models*. ARMA models try to capture local systematic variation. This can be useful for short-term forecasts.

trend

seasonality

autocorrelation



If a series contains trend, seasonality, or both, it contains autocorrelation. Even series that do not contain trend or seasonality can contain autocorrelation. This series does not appear to contain trend or seasonality, at least visually, but it does contain autocorrelation! When this is the case, detecting autocorrelation visually is, at best, difficult and, at worst, possibly misleading. It is best to focus on specific autocorrelation plots and diagnostics to evaluate the presence of autocorrelation in a time series.

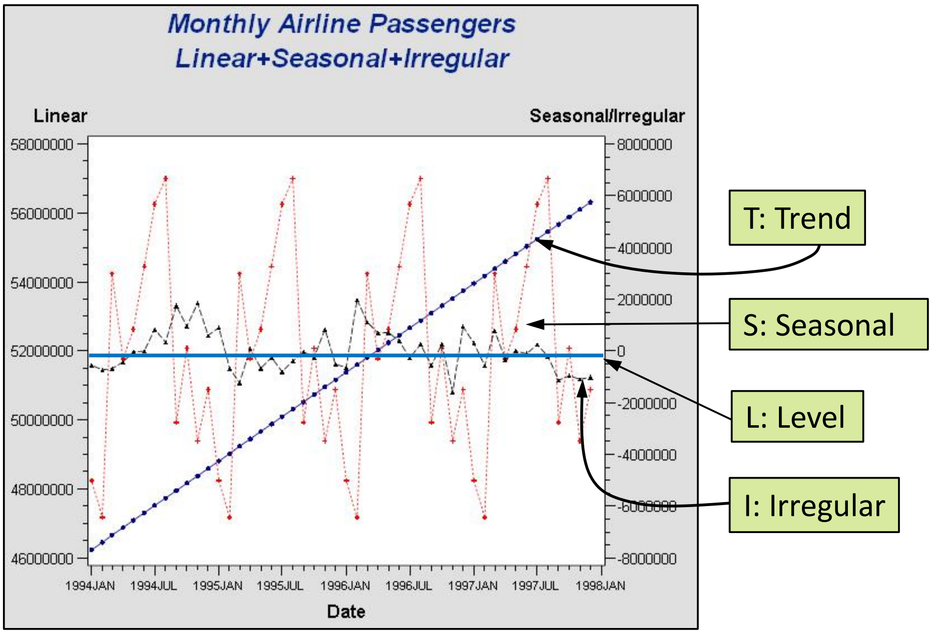
Signal Component Decomposition

time series decomposition

additive

or

multiplicative



The attempt at separating the signal of a series into its signal components is called *time series decomposition*. Notice that each component of the airline series here is represented separately in the plot.

Decomposition can be additive or multiplicative. Additive decomposition means that the total signal is a sum of the individual components, whereas multiplicative decomposition means that the total signal is the product of the individual components.

Classes of Time Series Models

How to classify time series models

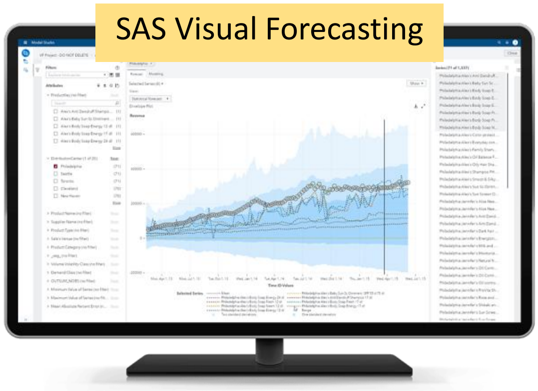
Lesson 02, Section 03

Classes of Models

time series models

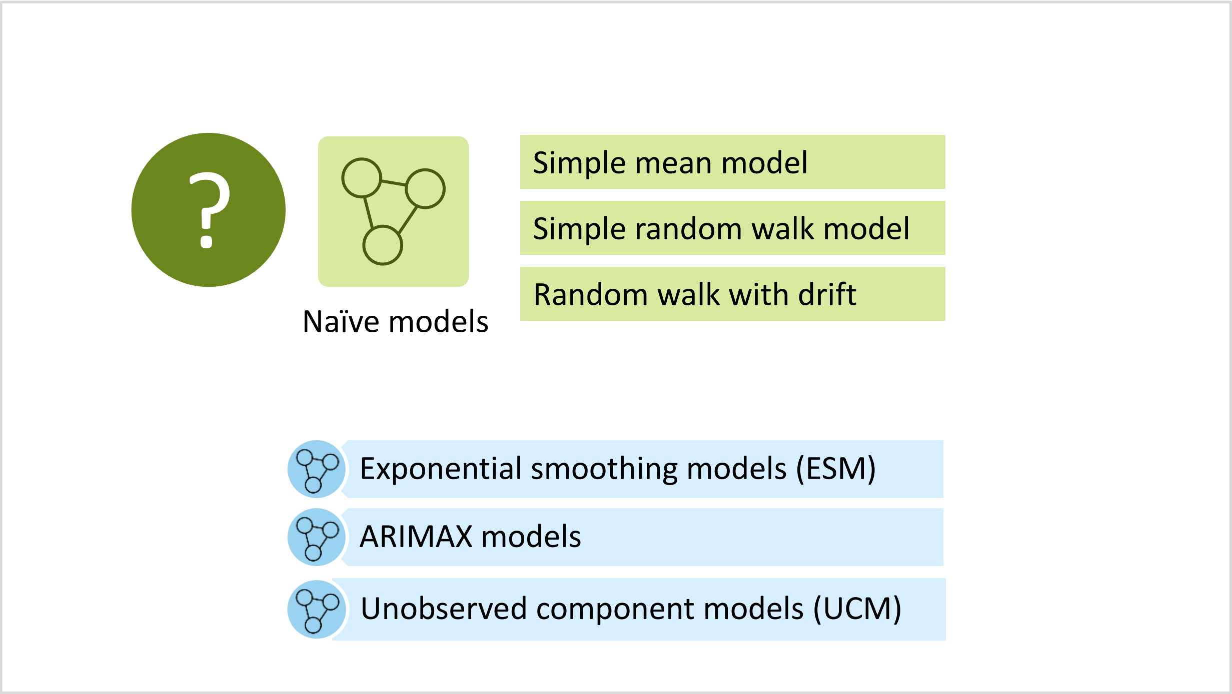
- Exponential smoothing models (ESM)
- ARIMAX models
- Unobserved component models (UCM)
- Simple regression models
- Simple models

Let's look at some classes of time series models that you will be able to work with. We discuss exponential smoothing models (ESM), ARIMAX models, and unobserved component models (UCM) in particular, but you might also use simple regression models or simple models. Simple regression models and other simple models are not discussed in this course.



- Exponential smoothing models (ESM)
- ARIMAX models
- Unobserved component models (UCM)

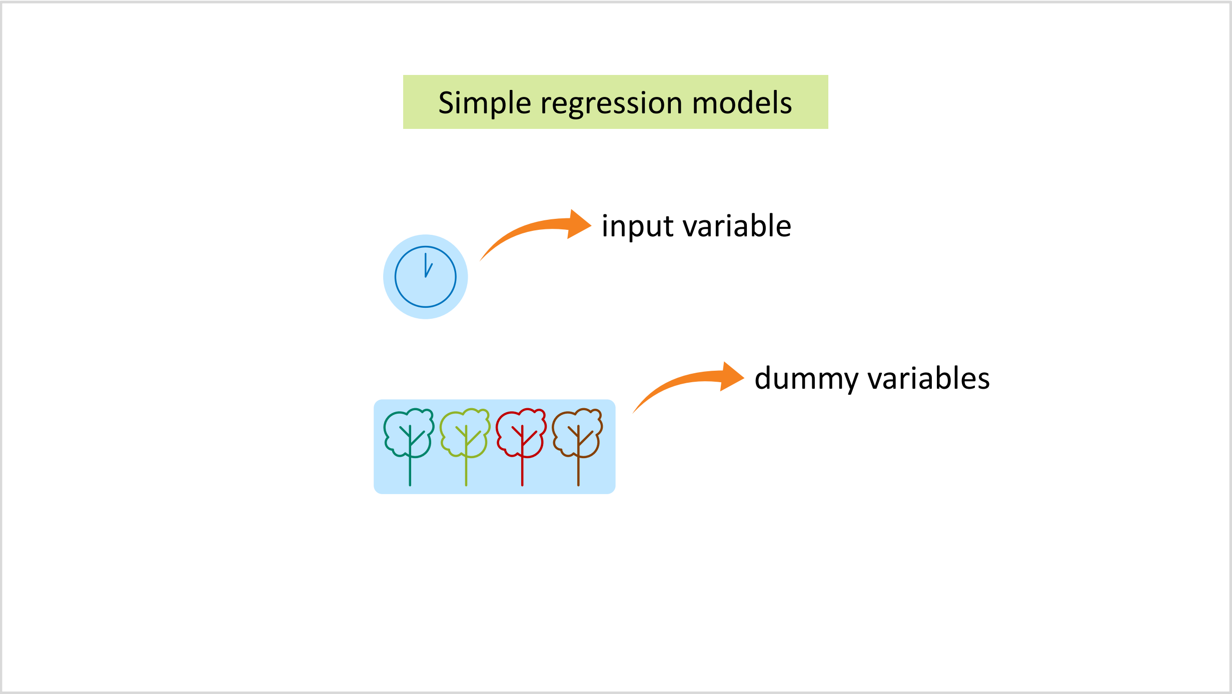
SAS Visual Forecasting tests multiple ESM and ARIMAX models by default. The UCM family can also be tested if you request it.



In addition to these three classes of models, time series can be modeled using naïve models. For example, in the simple mean model, forecasting the mean of the series is equivalent to a regression model where all slopes are set to the value zero.

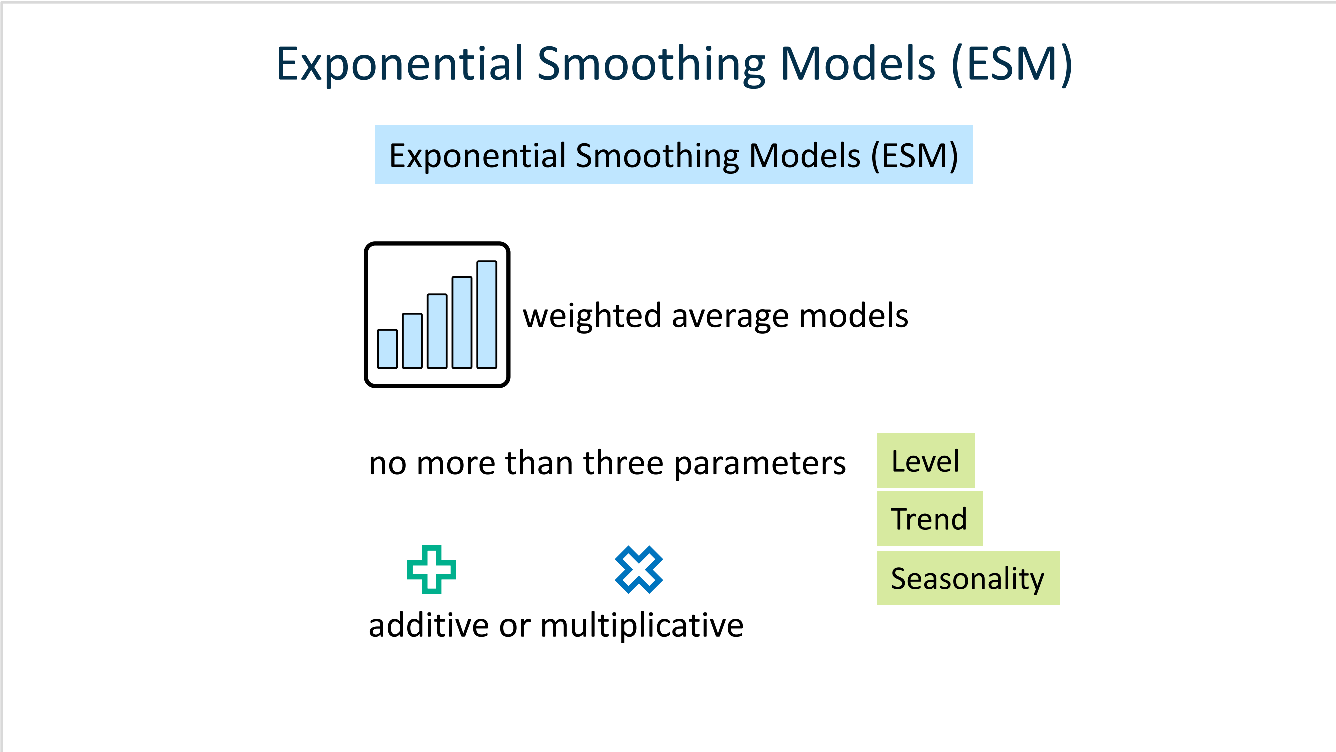
Why might you want to test a naïve model? The complex statistical models are valuable only if they perform better than what you could have done on your own without the models. It’s useful to have a reference to compare against.

Besides the simple mean model, you might choose a simple random walk model or random walk with drift as a naïve model. In these models, the forecasted value for the next time point is equal to the current observed value.



Simple regression models can also be used as time series forecasting models. Time can be used as an input variable. Seasonality can be modeled using dummy variables representing each season.

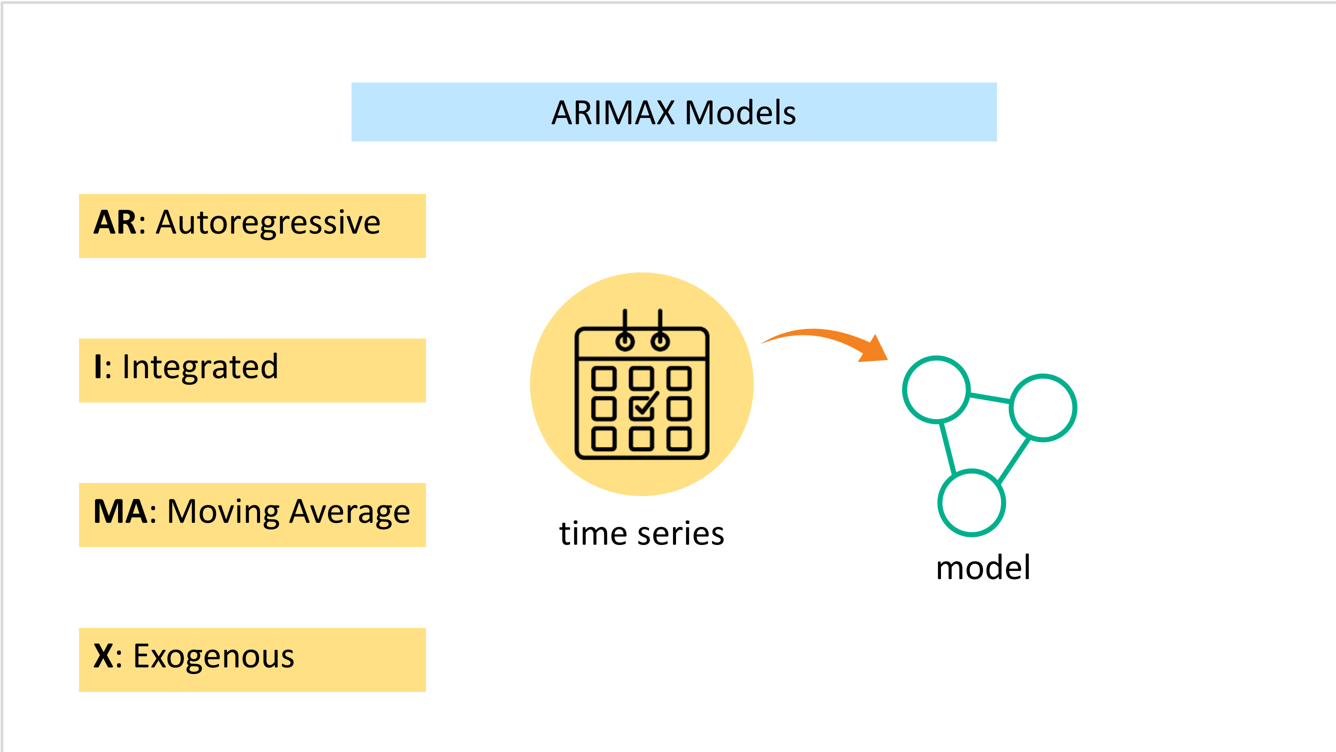
If creating an ESM, ARIMAX or UCM model becomes problematic, these simple model types become options.



Exponential smoothing models, or ESMs, are relatively simple models. ESMs are weighted average models that use no more than three parameters to estimate level, trend, and seasonality, or any subset of those components. ESMs are simple, but often adequate to produce a good model with very few parameters.

They can be additive or multiplicative. For additive models, the individual components are summed to produce the final model. Meanwhile, individual components are multiplied to produce the final model for multiplicative models.

Exponential smoothing models are intuitive and therefore quite popular. One drawback of ESM is that the models cannot handle the effects of exogenous factors.



ARIMAX, which stands for Auto-Regressive Integrated Moving Average with eXogenous factors, are perhaps the most widely used time series models. For all but one of the linear ESM models represented in the ESM procedure, there is an ARIMAX representation. The only exception is the damped trend ESM model. ARIMAX models are much more flexible than ESMs, not only because they can handle exogenous inputs, but because you have more parameters available to estimate models.

ARIMAX Models

AR: Autoregressive → Time series is a function of its own past.

I: Integrated

MA: Moving Average

X: Exogenous

As an autoregressive model, ARIMAX models use autocorrelation. This means that current values of the series are forecast from one or more previous measures from the series. The previous series values, called *lagged values*, are related to current values through a weighted function. This means that the time series is a function of its own past.

ARIMAX Models

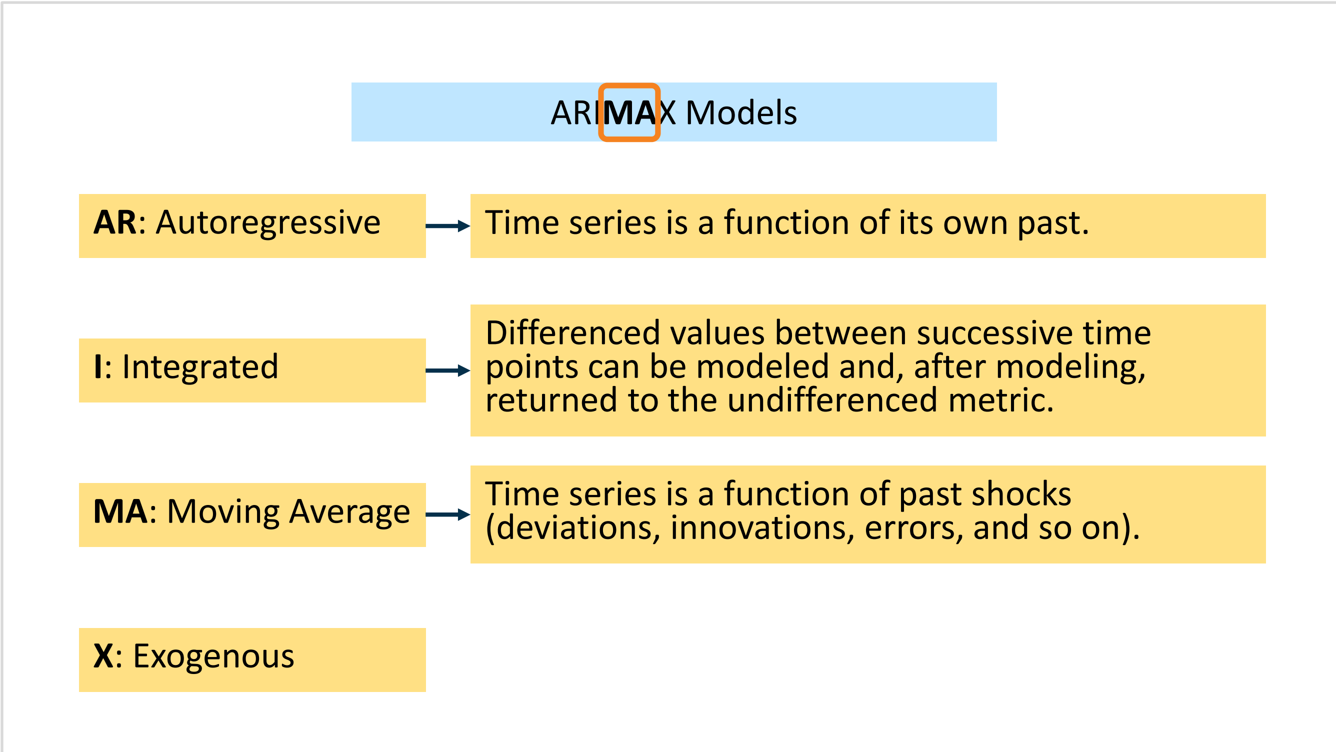
AR: Autoregressive → Time series is a function of its own past.

I: Integrated → Differenced values between successive time points can be modeled and, after modeling, returned to the undifferenced metric.

MA: Moving Average

X: Exogenous

The integrated part of ARIMAX is used when the series contains trend or seasonality. The traditional approach to accommodating trend and seasonality, introduced by statisticians Box and Jenkins, is to difference the data. If the data have trend, you first difference the data by taking the differences between successive values of the series. This process is sometimes referred to as *detrending*. If the series shows a repeating seasonal pattern, you can difference the data across a seasonal span, such as taking differences across successive February values. This process is often referred to as *deseasonalizing*. According to the Box and Jenkins textbook, if your data have trend and seasonality, perform the appropriate differences before you apply the autoregressive and moving average models. The modeling is done on a differenced series. Then, undo the differencing to keep trend and the seasonality in your final forecast. Returning to the undifferenced metric is called *integration*.



Moving average is similar to autoregressive types of variation, but the model is written in terms of lagged forecasting errors rather than on lagged series values. This is not quite as intuitive as autoregression.

ARIMA**X** Models

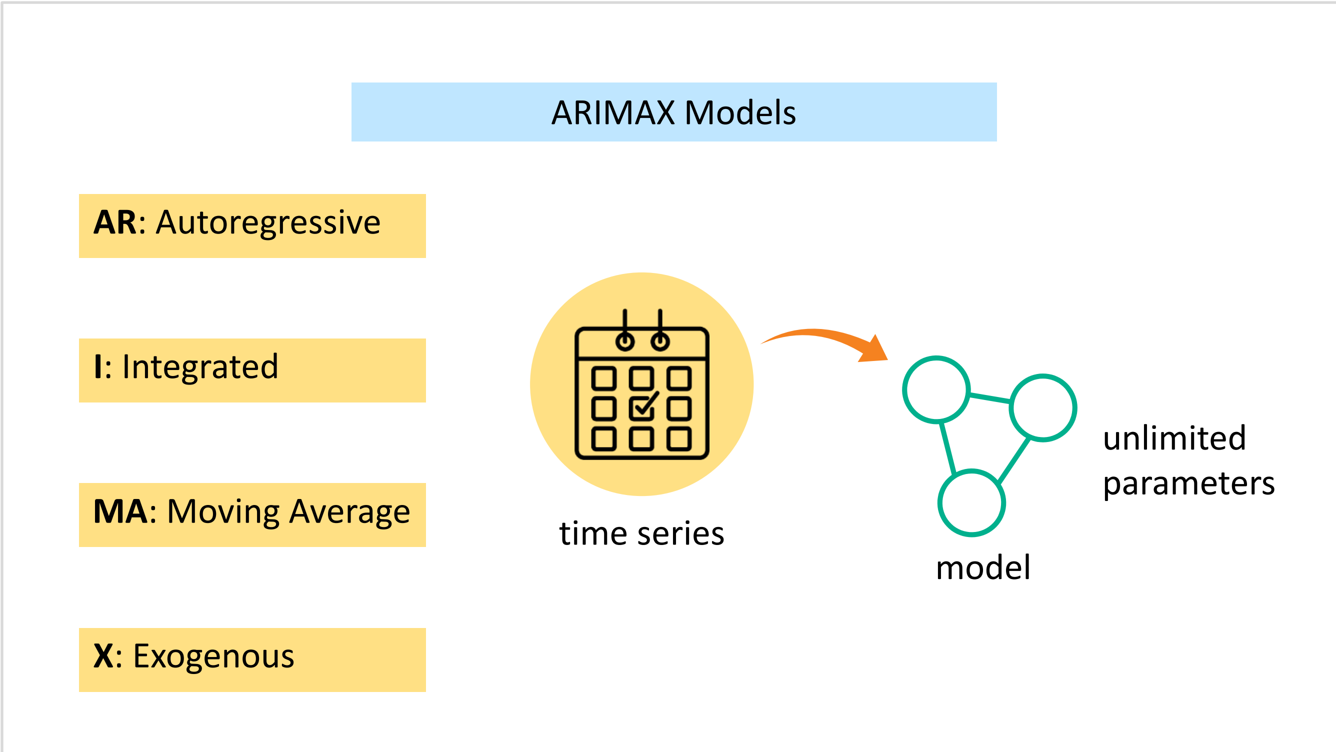
AR: Autoregressive → Time series is a function of its own past.

I: Integrated → Differenced values between successive time points can be modeled and, after modeling, returned to the undifferenced metric.

MA: Moving Average → Time series is a function of past shocks (deviations, innovations, errors, and so on).

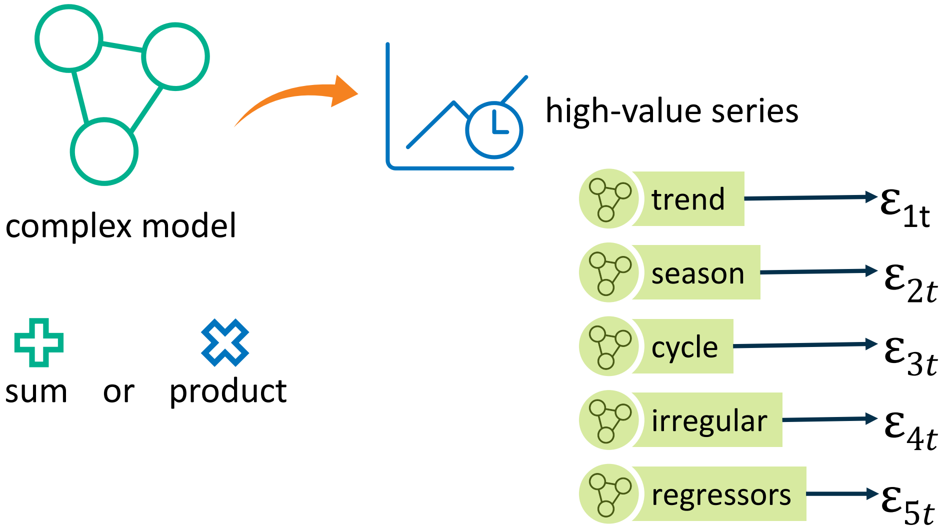
X: Exogenous → Time series is influenced by explanatory factors.

The X in ARIMAX stands for “eXogenous," as ARIMAX models can handle exogenous factors. The X is the time series regression part, where you model the effects of input series. The effects of the input series might be delayed, such as when a price increase this week affects demand the next week and beyond, defining what is known as a *dynamic* relationship between the exogenous series and the target series.



There is no real limit on the number of factors that can be estimated in an ARIMAX model, which means that this model can potentially fit a time series extremely well. However, no limit on the number of parameters makes ARIMAX models vulnerable to overfitting. Having a holdout sample can reduce that risk. You learn about honest assessment with holdout samples later.

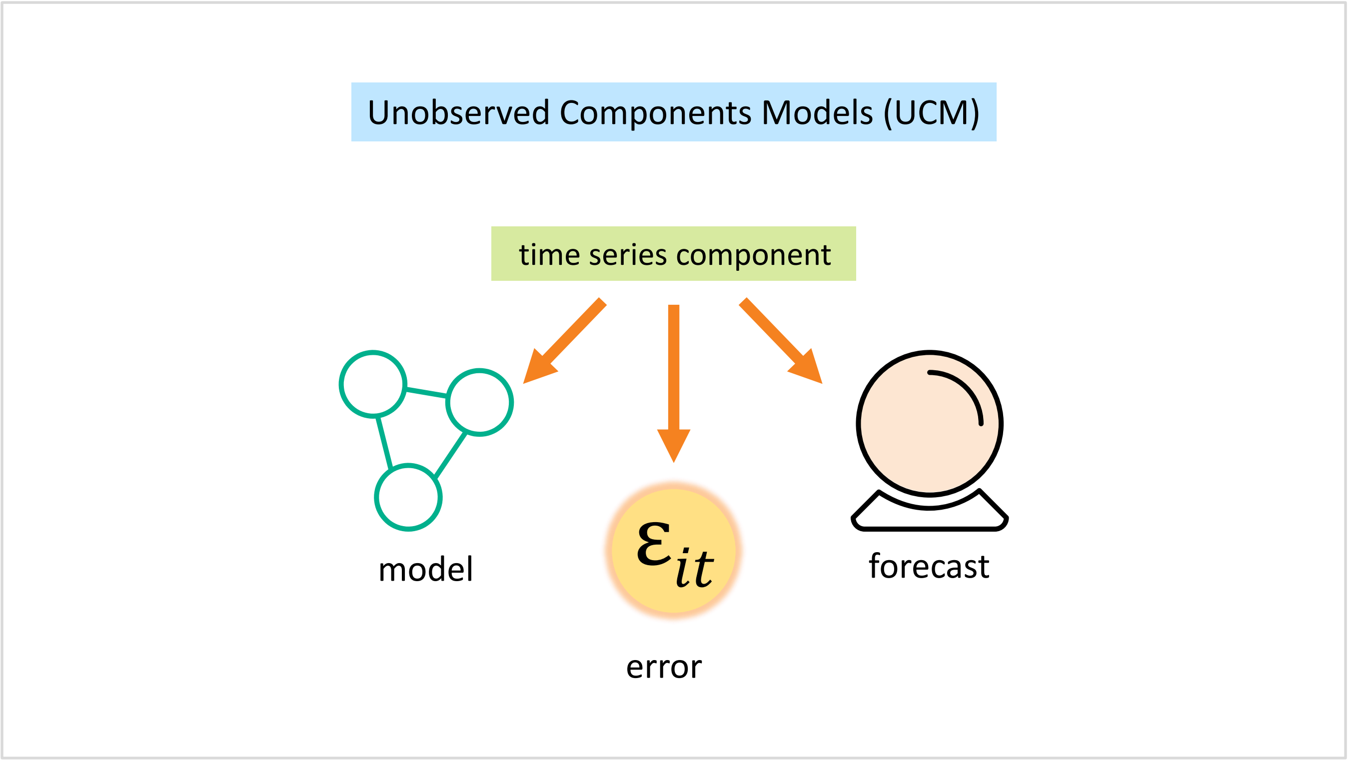
Unobserved Components Models (UCM)



Unobserved components models, or UCMs, are complex but intuitive models. UCM models are quite flexible, but the algorithm that is used to estimate the parameters is intensive. UCMs might be valuable when you are forecasting a few high-value series, but you're not going to want to test them on a million series at once.

UCMs decompose the time series in similar ways to ESM and ARIMA models. The key difference in UCMs is that each of the components has its own sub-model and its own error term. So the big difference between an unobserved components model and an ARIMA model, for example, is that the ARIMA model will have one error term, whereas the UCM will have an error term associated with each of the components. This is part of what makes UCMs slower to run.

The model of Y_t is modeled as either the sum or product of all relevant signal components. In addition, the error is a combination of the errors for each relevant component.



In summary, in a UCM, each component captures some important feature of the series dynamics. UCM model components have their own models, and each component has its own source of error and its own forecasts. The coefficients for trend, season, and cycle are dynamic, and the coefficients are testable.

HTML Placeholder

Performance

Simple Models

Simple models have no performance issues.

Exponential Smoothing Models

Exponential smoothing models can be constructed quickly and easily, so they always have good performance.

ARIMAX

ARIMAX models require many more computer cycles than simple or exponential smoothing models, so some approximations and shortcuts are used to speed performance. Thus, the ARIMAX model that is selected by the software is not necessarily the best ARIMAX model with respect to a given accuracy measure.

UCM

UCM models are very computer-intensive and should be tried only on small data sets or individual time series.

Demo: Performing Basic Forecasting with a Pipeline

This demonstration illustrates performing basic forecasting with a pipeline.

Answer: MAPE Distribution Histogram

The results of the Auto-forecasting node contain a MAPE Distribution histogram. This diagnostic illustrates which of the following?

the fit of the forecast models for all series contained in the data hierarchy

the types of models used to generate forecasts in the pipeline

the number of missing or anomalous observations across all series in the data

the fit of forecast models for series contained in the base level of the data hierarchy

Answer: d - The MAPE histogram displays the fit of the forecast models for all the models across the lowest level of the hierarchy.

Demo: Exploring Generated Models and Forecasts Using the Forecast Viewer

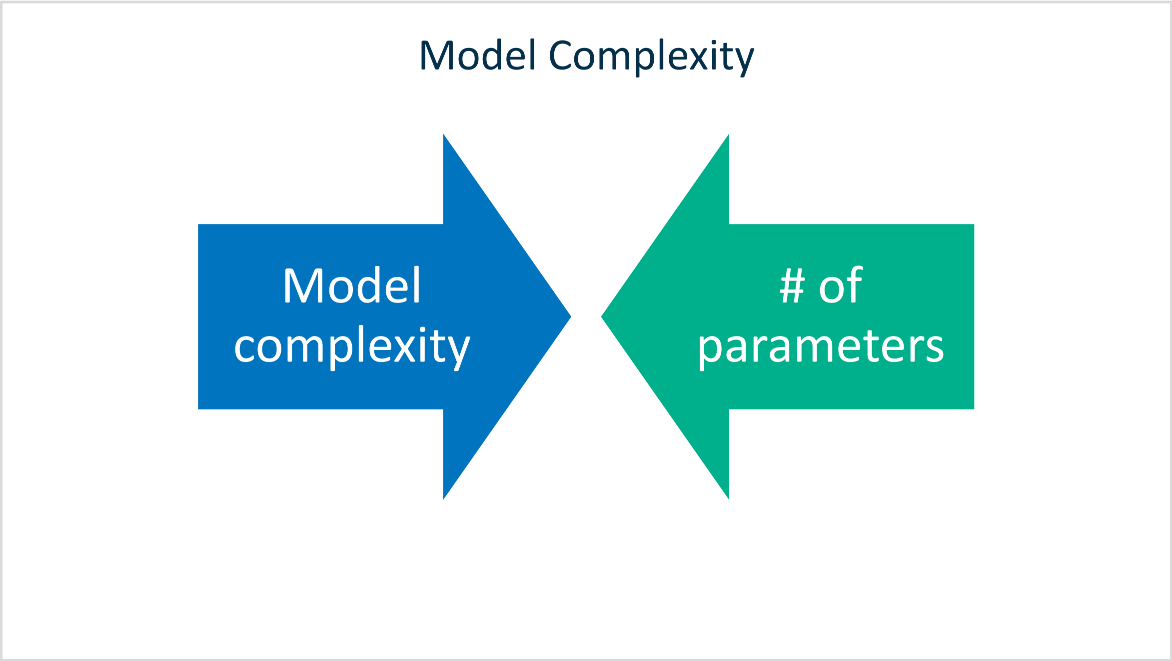
This demonstration illustrates how to explore generated models and forecasts using the forecast viewer.

Questions?

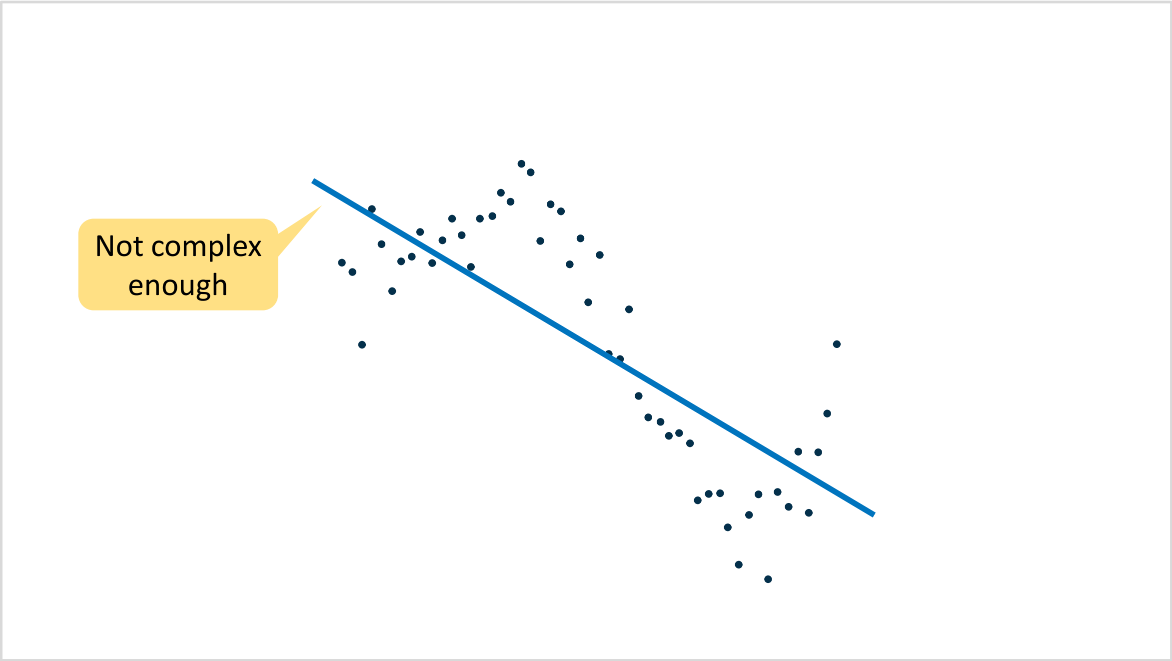
Model Comparison Using Honest Assessment

How to compare time series models using honest assessment

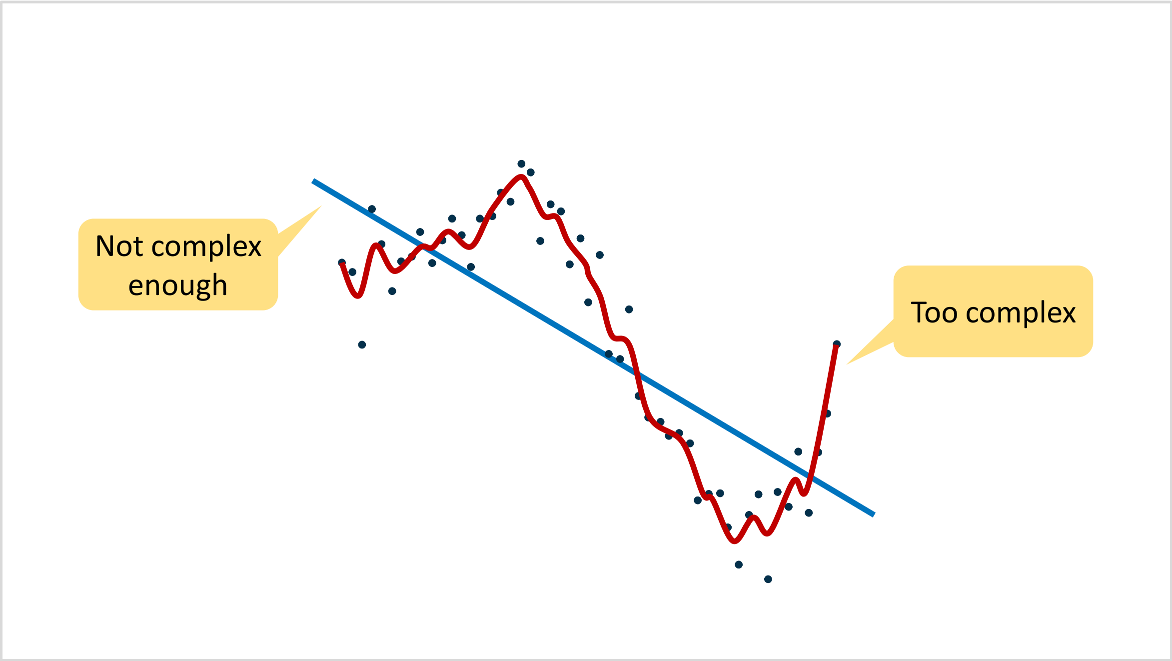
Lesson 02, Section 04



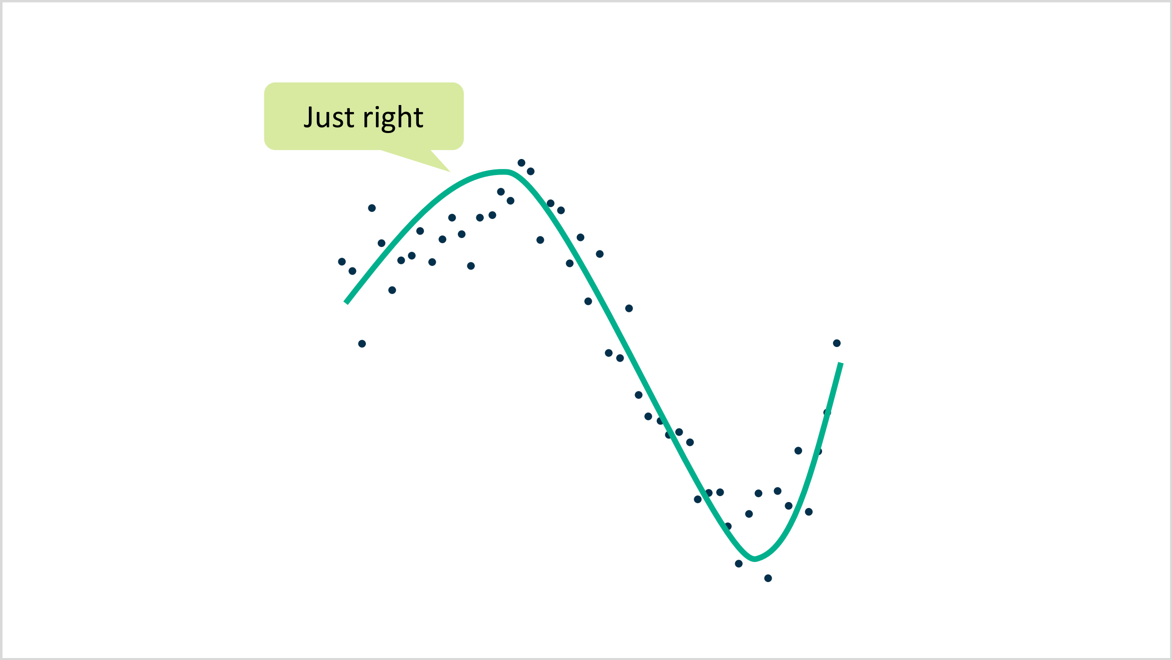
We define model complexity as the number of parameters that we are estimating. In general, complex models are more flexible, but they can also be too flexible.



Suppose you have some data points, as shown here. An insufficiently complex model might not be flexible enough, which can lead to underfitting—that is, systematically missing the signal.



A naïve modeler might assume that the most complex model should always outperform the others, but this is not the case. An overly complex model might be too flexible, which can lead to overfitting—that is, accommodating nuances of the random noise in the sample.



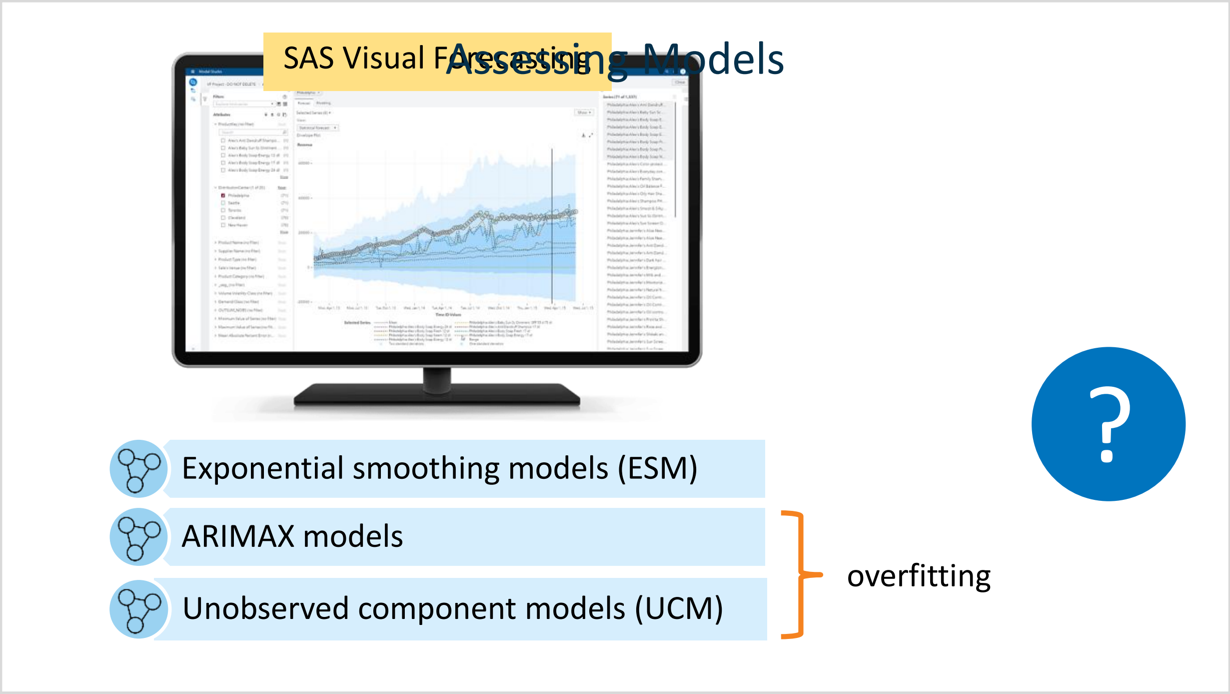
The idea with time series modeling is very similar to the idea when you're doing predictive modeling. We want a model that captures the signal but ignores the noise. A model with the right amount of flexibility gives the best generalization.

Assessing Models

SAS Visual Forecasting

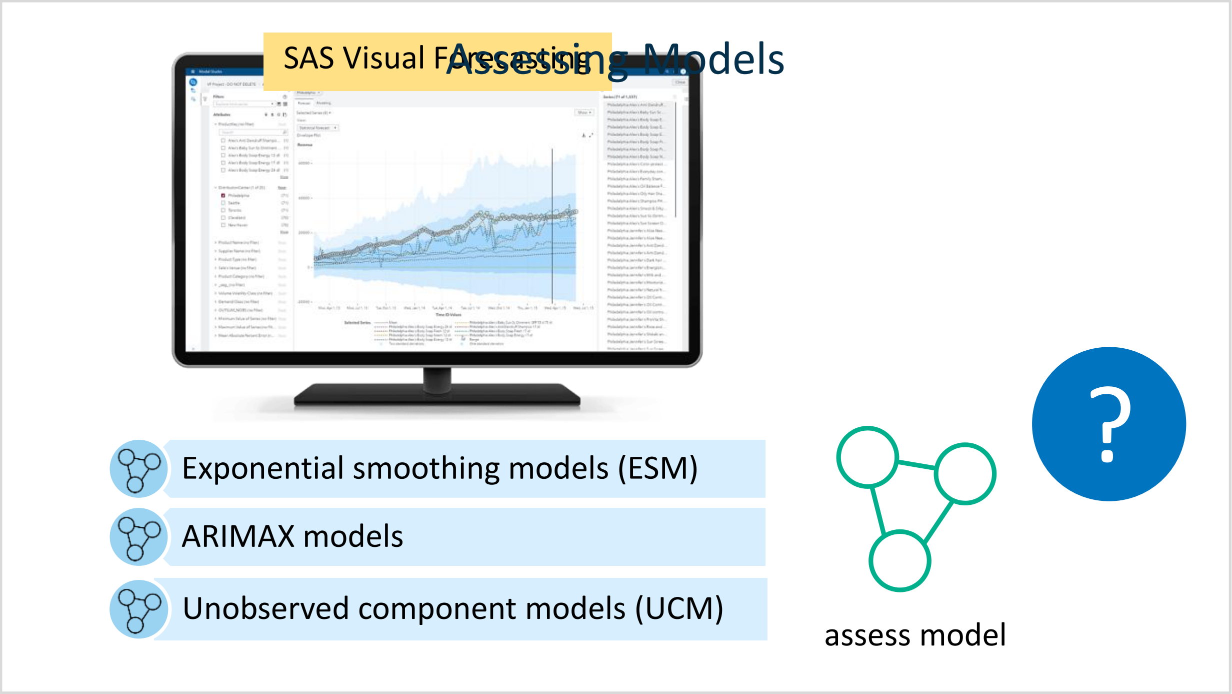
- Exponential smoothing models (ESM)
- ARIMAX models
- Unobserved component models (UCM)

So far, you have learned about the model classes available in Visual Forecasting. These are exponential smoothing models, autoregressive integrated moving average models with exogenous factors, and unobserved components models.

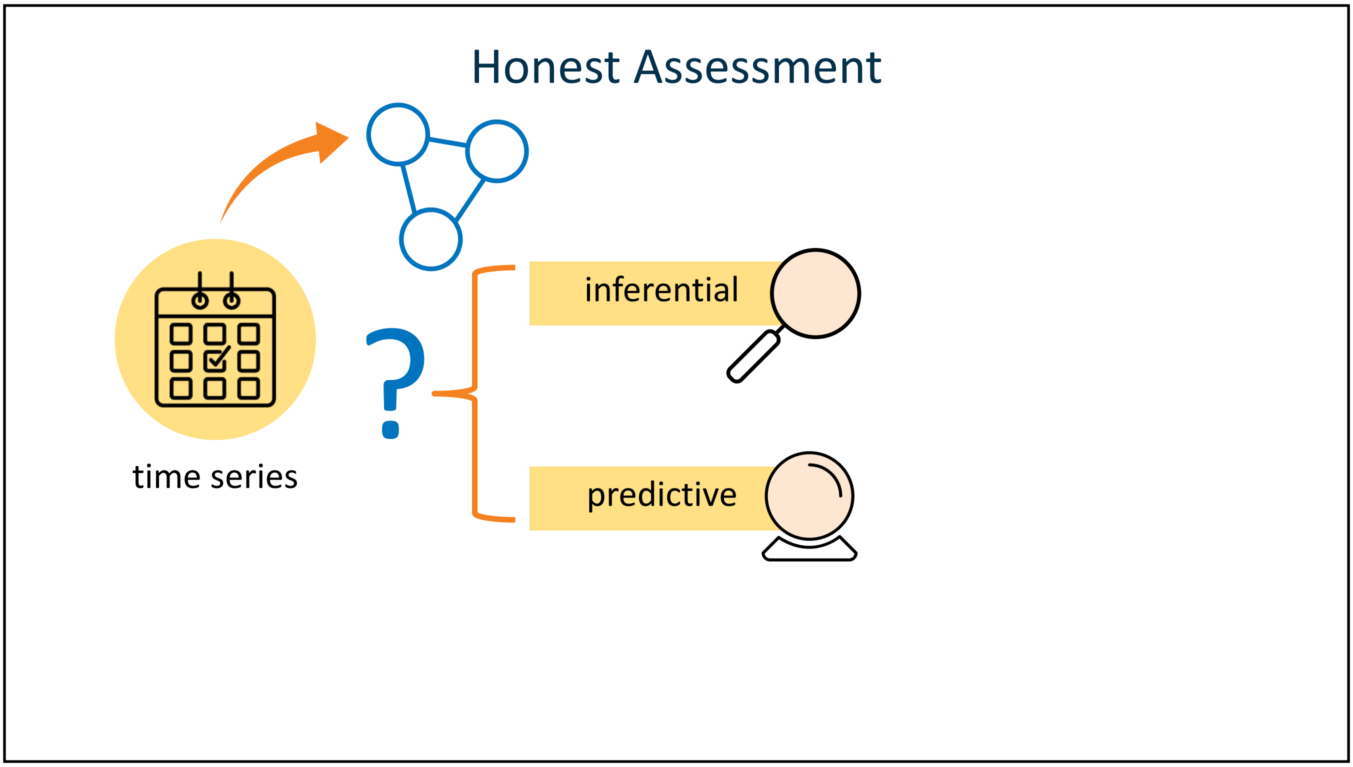


You know that ARIMAX and UCM models, in particular, are subject to overfitting. This means adding parameters that contribute to modeling the noise in the sample, but do not contribute to modeling signal in the future.

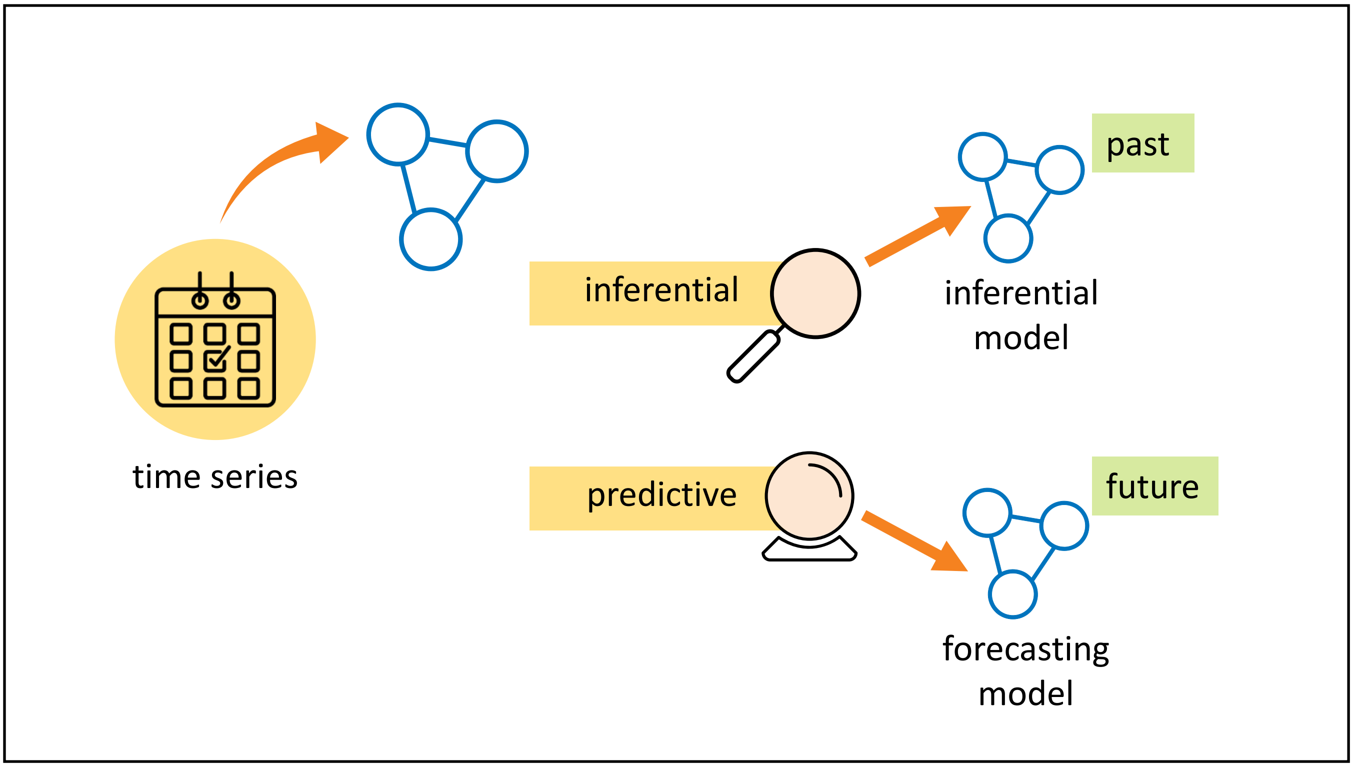
How do you protect yourself from overfitting?



By assessing your model.

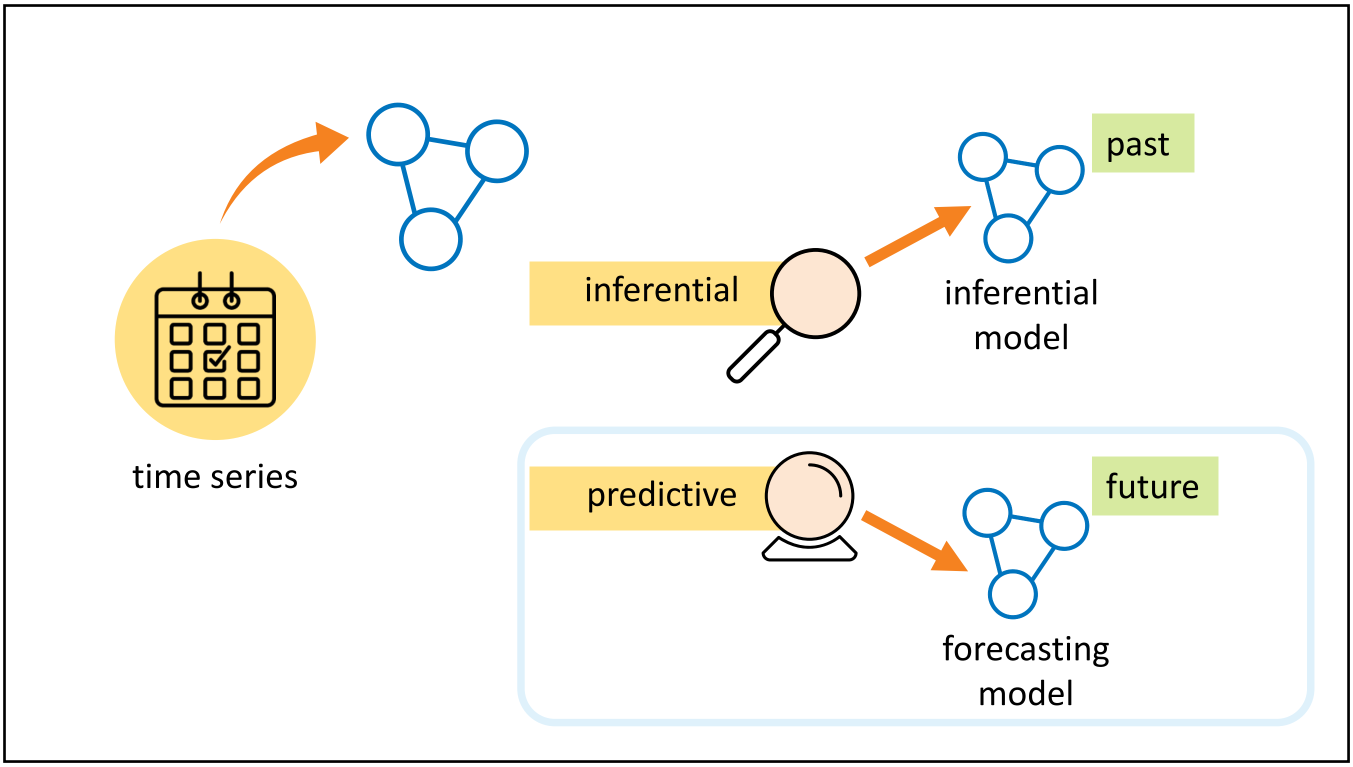


When you model time series data, you decide whether you are doing it for inferential purposes or predictive purposes.

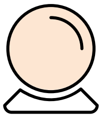


The goal in inferential models is to explain the past, or determine what factors explain past behavior of a series. The past values are all you are concerned with in inferential models.

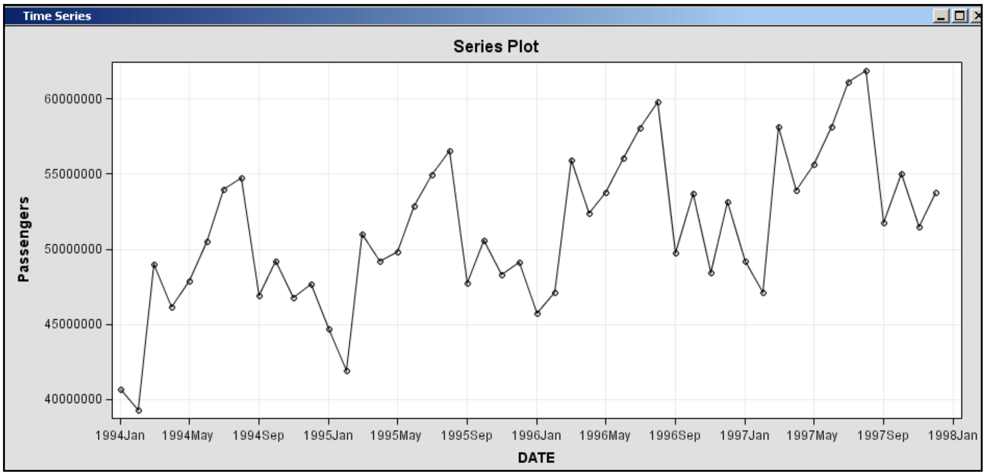
However, predictive models, or rather forecasting models, are used to produce estimates of what will happen in the future. Because one type of model is focused on the past and the other type of model is focused on the future, assessment of the models is done differently.



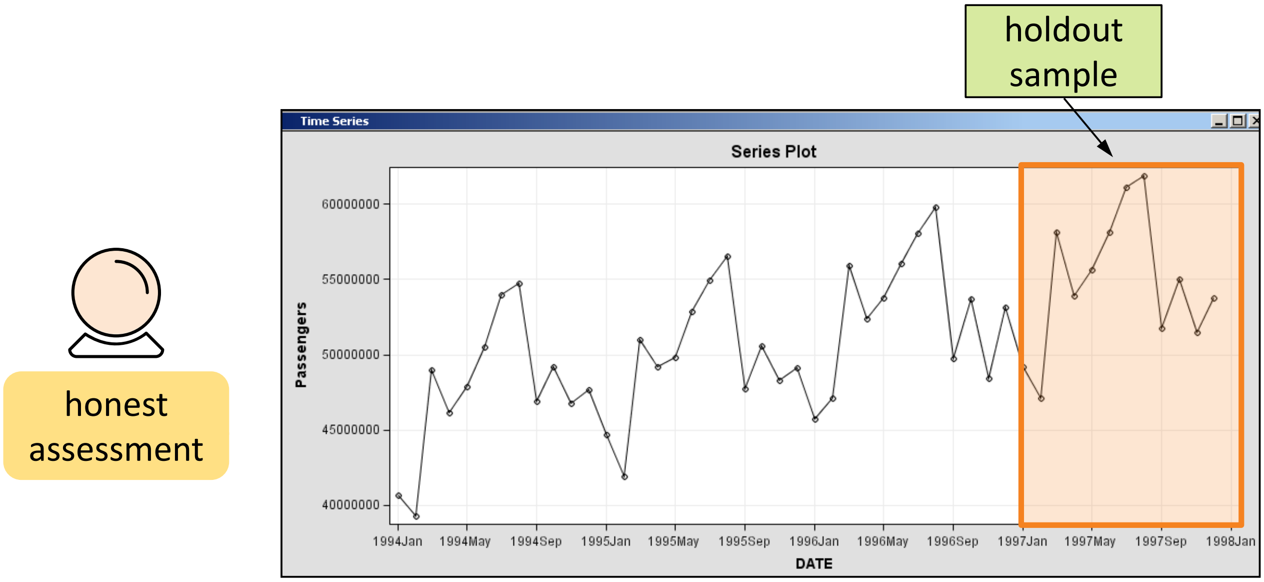
We are primarily focused on forecasting models.



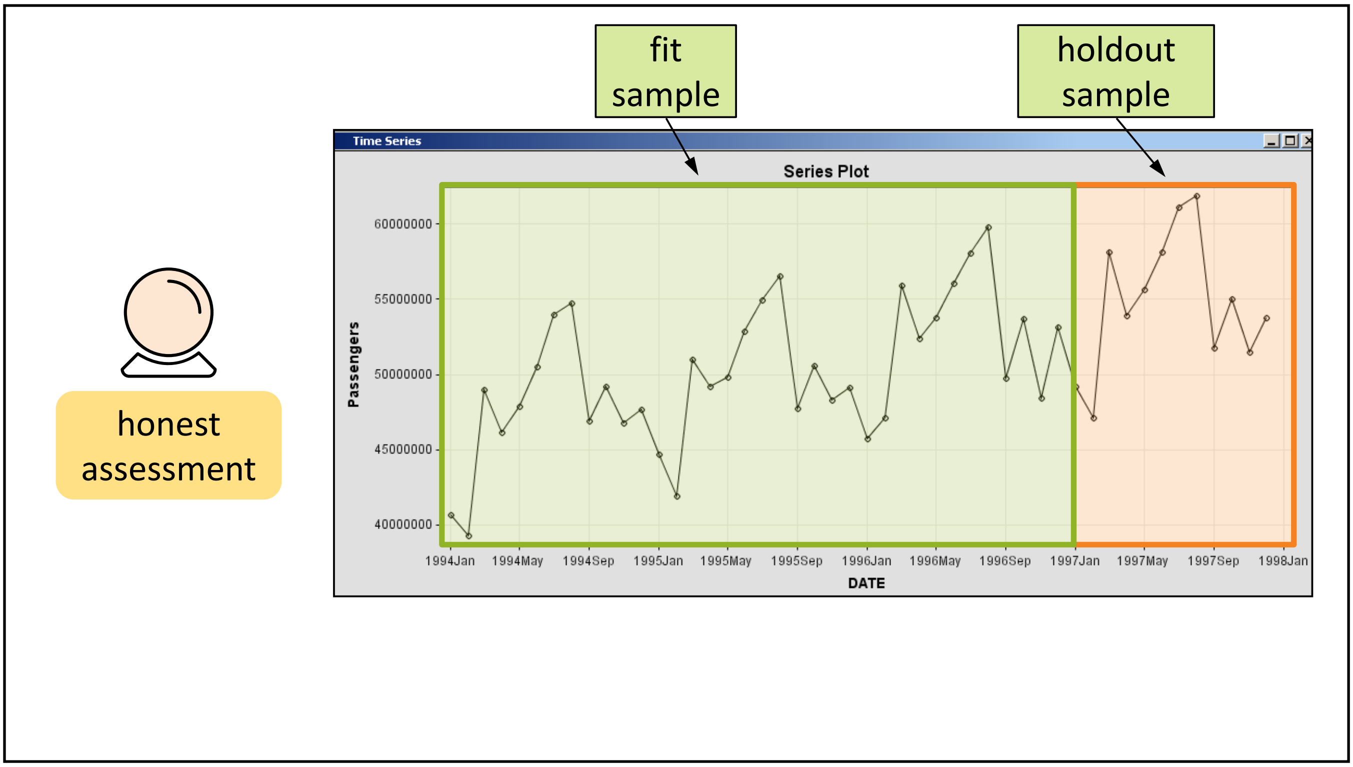
honest
assessment



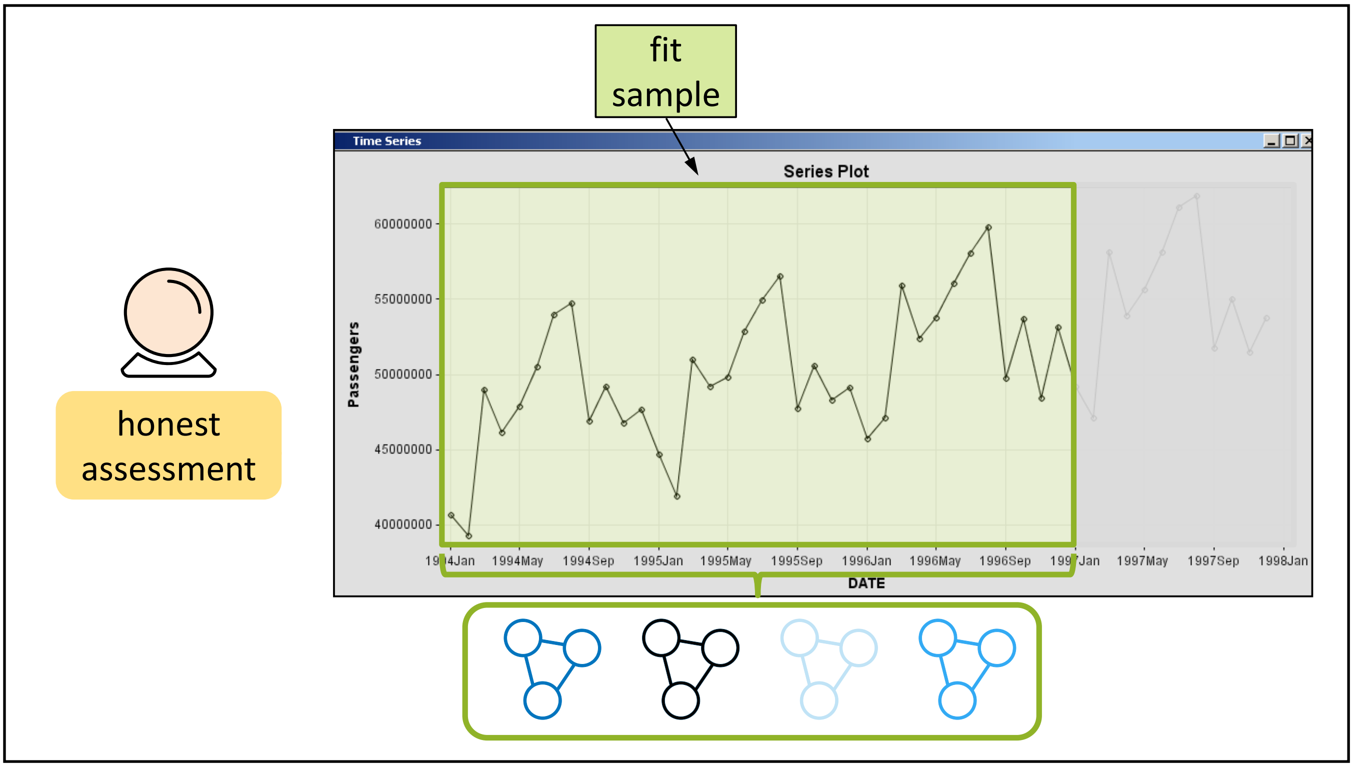
Forecasting models are assessed using a method known as *honest assessment*.



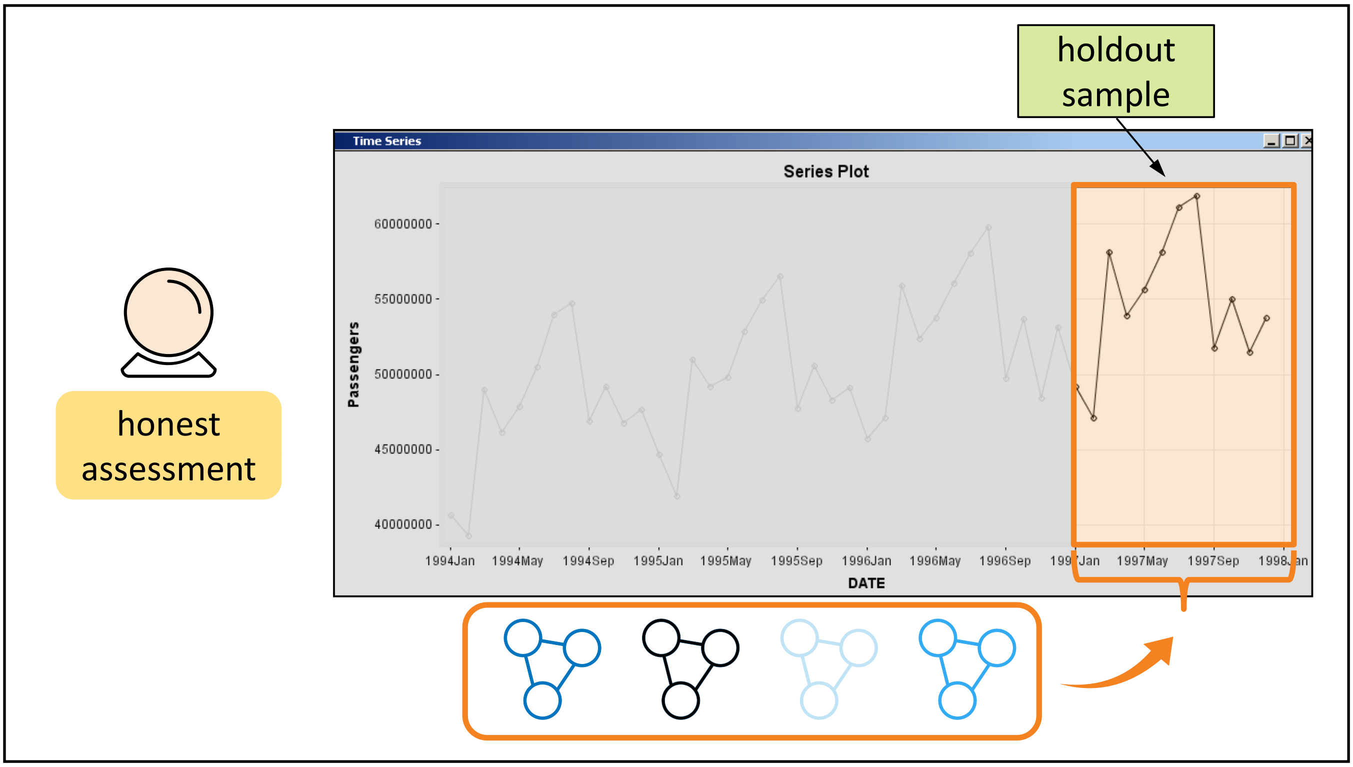
The idea behind honest assessment is that you can simulate forecasting future measurements by holding out the most recent measurements and pretending that you don't know those values. The most recent measurements are referred to as the *holdout sample*, which is also known as the *validation sample* in other areas of predictive modeling. The holdout sample is always at the end of the series.



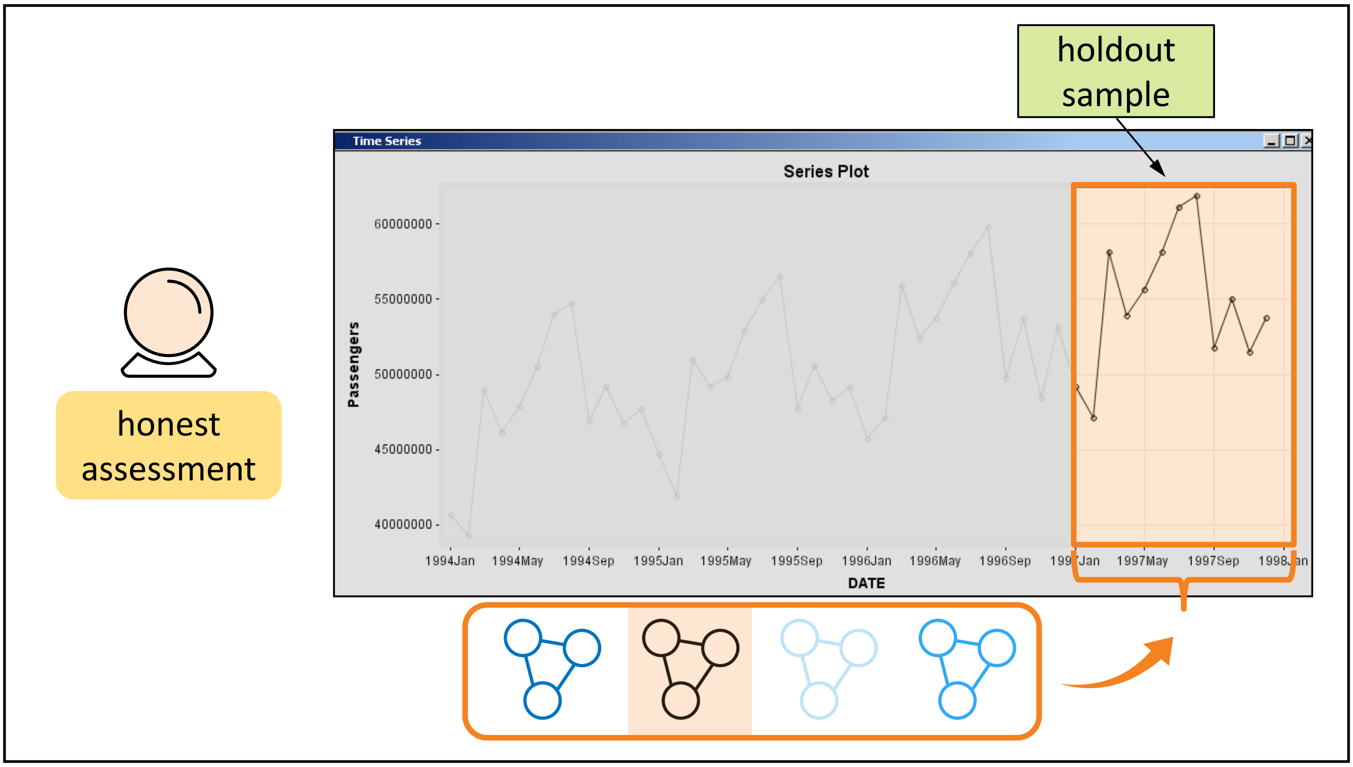
The remaining measurements are referred to as the *fit sample*. In other predictive modeling scenarios, the fit sample is known as the *training sample*. Here, you see the airline data split into fit sample and holdout sample.



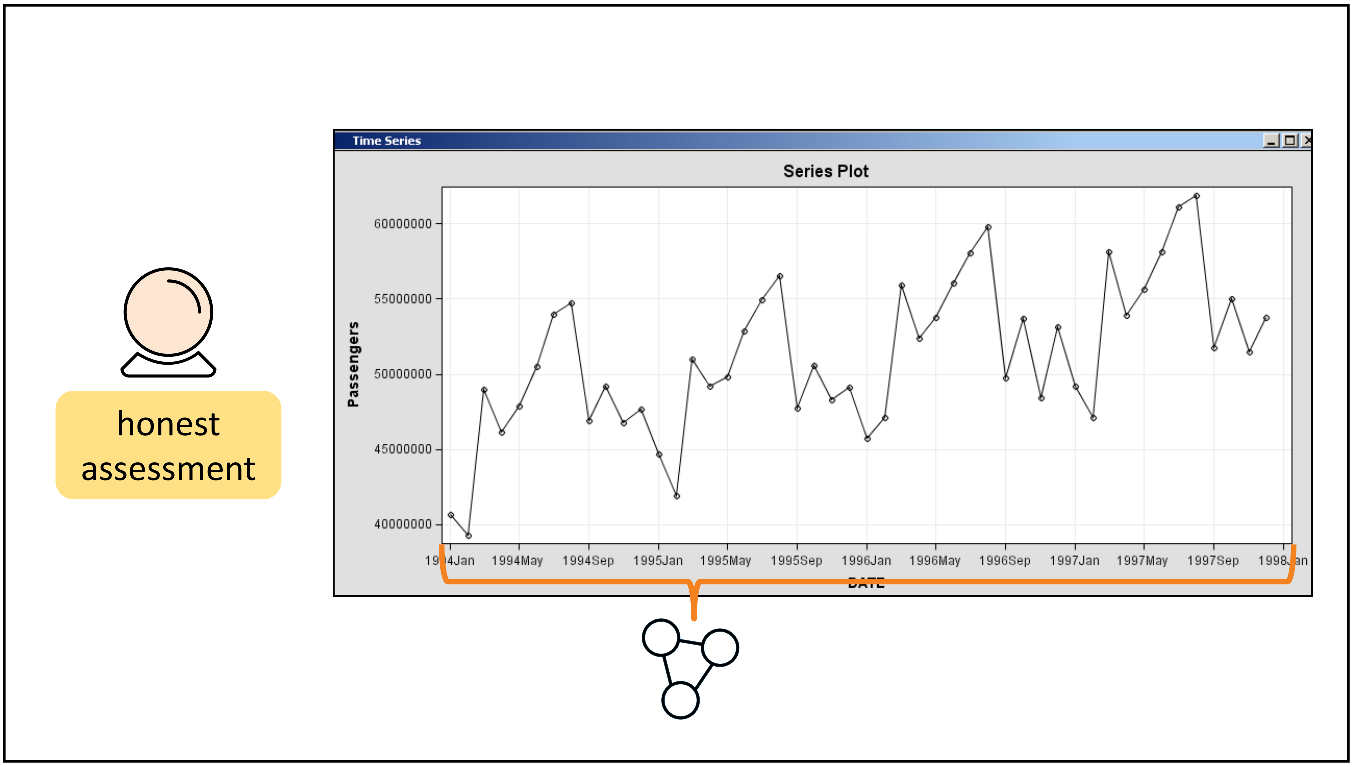
After the series is split into a fit sample and a holdout sample, you fit several candidate models to the fit sample using any modeling family that you want.



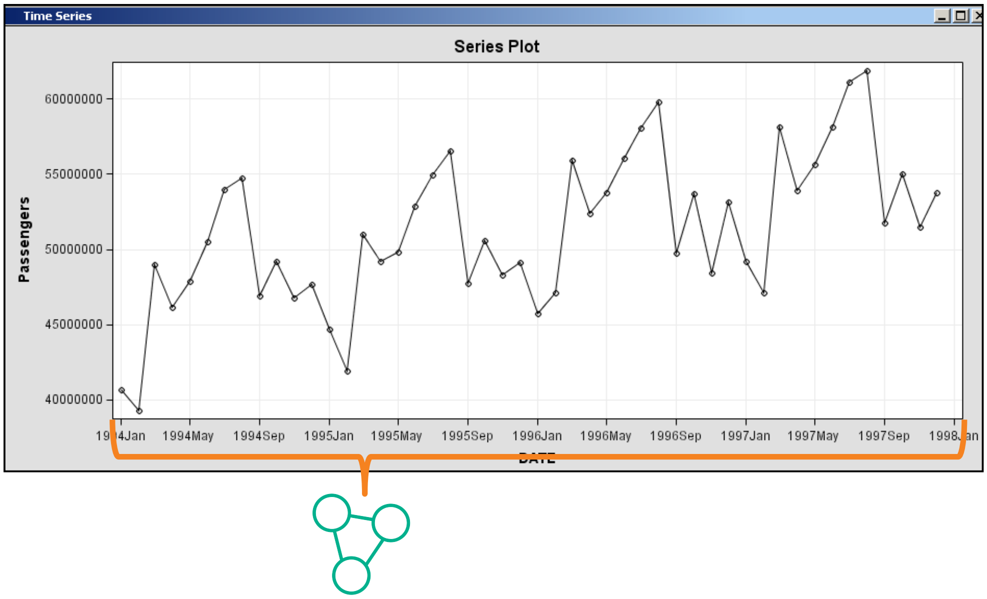
For each model estimated, you calculate forecast values for the time span of the holdout sample. Then you calculate an accuracy statistic value, based on how close the forecasted values are to the actual values.



The champion model is chosen from these accuracy statistics of your candidate models.



After you determine a champion, you then re-run your champion model on the full data, which is fit plus holdout, to obtain more complete parameter estimates for future forecasts.



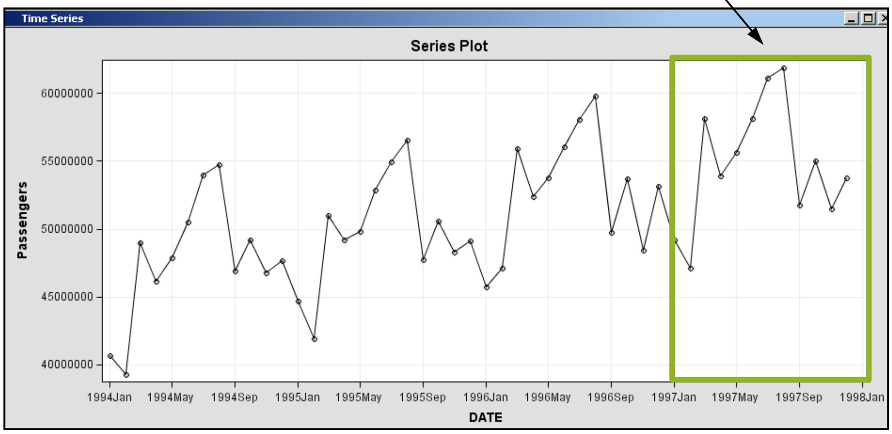
Sometimes, splitting your series into fit and holdout is a luxury that you cannot afford because either your sample is too small or the behavior of the series changes in the later measurements. In these cases, you can forecast and calculate accuracy measures based on the entire series. Doing so is not honest assessment, but sometimes it is the best that you can do.

best practices

Choosing the Holdout Sample

- at least one season of data
- avoid unique behavior
- four time points for every parameter
- no more than 25% of series

holdout sample



You have to select an appropriate holdout sample if you attempt honest assessment. There are a few key points to remember when choosing a holdout sample.

If you plan on modeling seasonal effects in any of your models, you need to have at least one seasonal period worth of data in the holdout sample. For example, if you have monthly sales data where the cycle is one year, you need at least 12 monthly measurements in your holdout sample.

If any unique behavior occurs in the period that you are considering for your holdout sample, then you should not use a holdout sample. For example, you might have 10 years of data from a restaurant at a beach, but a storm damaged the restaurant during the last year. Forecasts of sales during the last year would be adversely affected due to the unforeseen and likely unrepeatable event of the storm damage. Assessment of the model based on only the last year would not be a true assessment of the accuracy of the model during typical years.

You also need to consider sample size. A common rule of thumb is that the series requires at least four time points for every parameter to be estimated in the model. This would need to be calculated based on the most complex model you are considering. The fit sample is the primary concern. You should consider splitting your sample only when the fit sample size requirement is met and there will be enough data left over for a seasonal period worth of data in a holdout sample.

Finally, the holdout sample should rarely contain more than 25% of the data in the entire series. For example, in our airline data, we have four years of data. We can use three-quarters of the data for our fit sample. Then we can use a quarter of the data as our holdout sample.

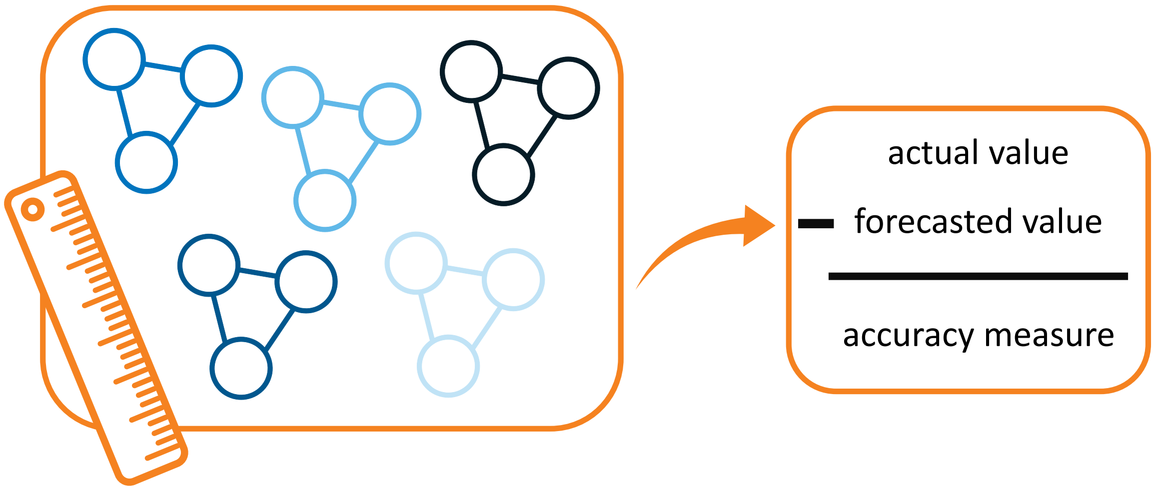
Answer: Holdout Samples

Which of the following is a best practice when selecting a holdout sample?

- Use the first 25% of the data.
- Choose data that exhibits unique behavior.
- Use no more than 25% of the series data.
- Include at least two seasons of data.

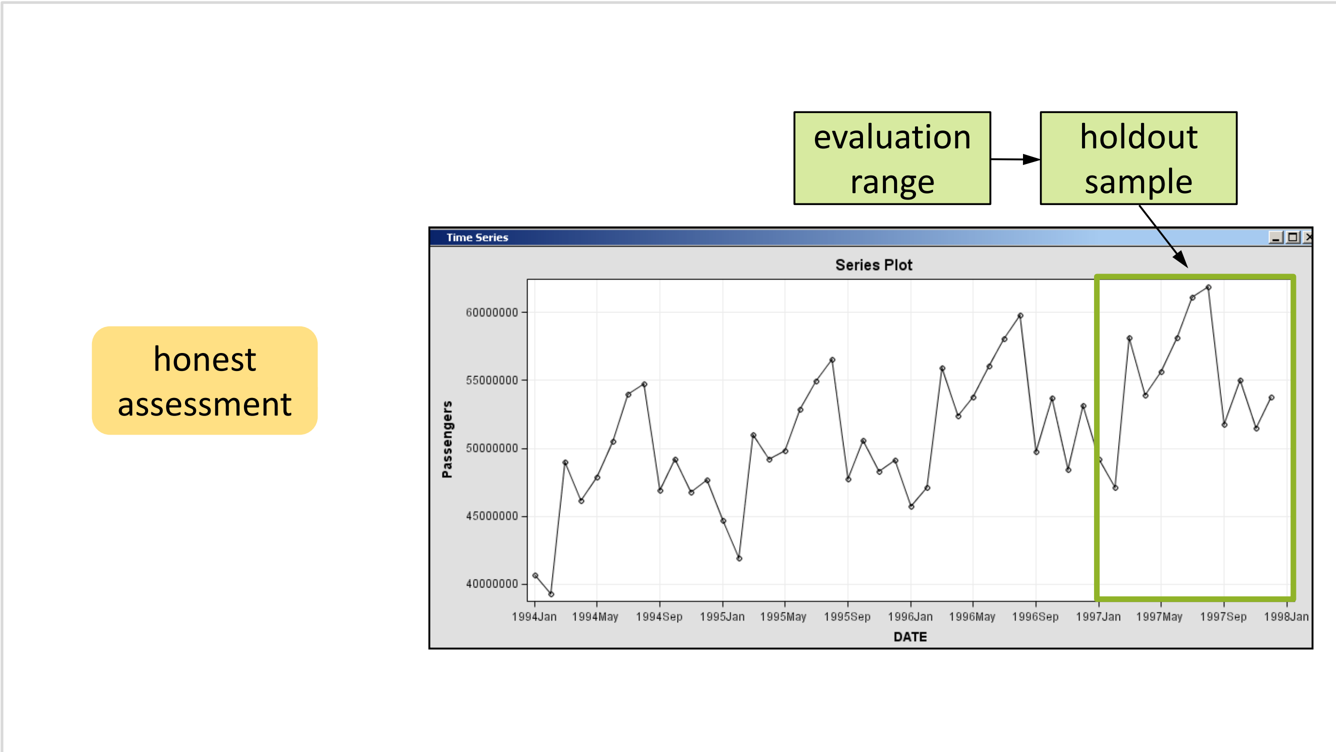
Answer: c - The holdout sample should include no more than 25% of the series data. Remember that the holdout sample should be taken from the end of the series, which includes the most recent series values. It is a best practice to include one season of data without unique behavior in the holdout sample. The holdout sample should also include four time points for every parameter.

Accuracy versus Goodness-of-Fit



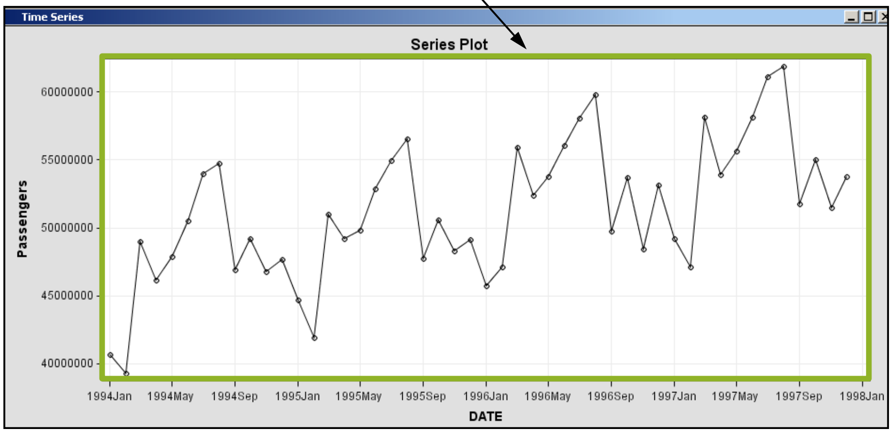
Now that you understand honest assessment, you must decide on an accuracy measure with which to choose a champion model.

Most accuracy measures are based on forecast error—that is, the difference between the actual value and the forecasted value in the evaluation range.



Remember that the evaluation range in true honest assessment is the holdout sample.

evaluation range

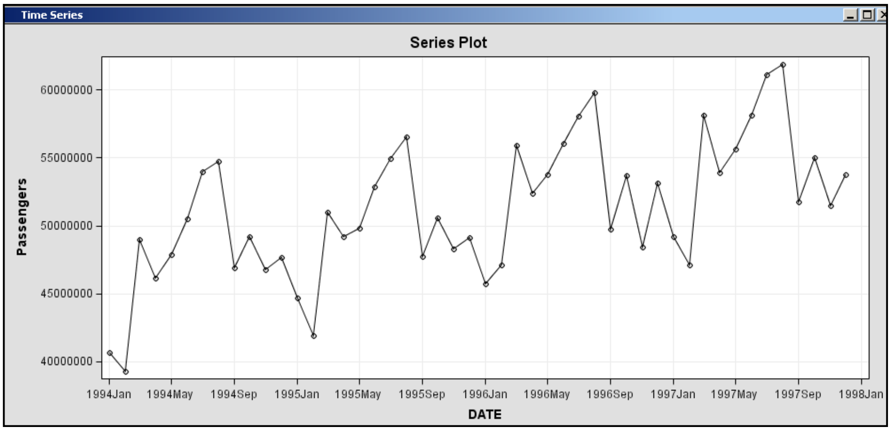


holdout sample

However, when a holdout sample is not feasible, evaluation can be done on the entire series.

one-step-ahead forecast

multi-step-ahead forecast

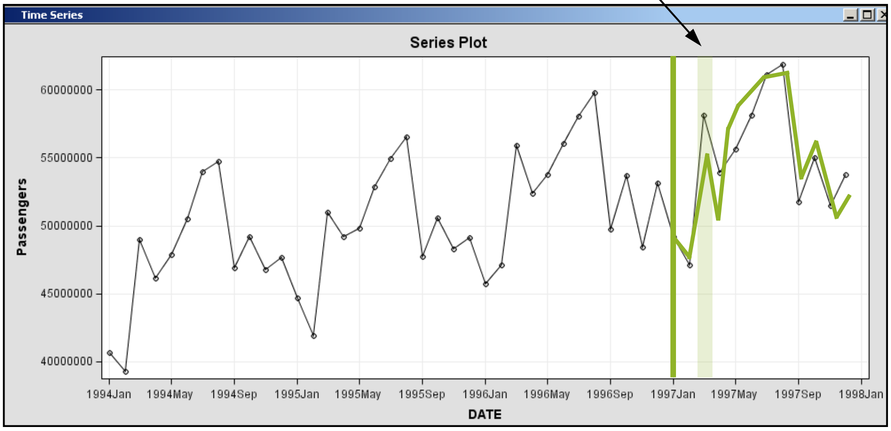


In addition, there is a difference between one-step-ahead forecasts and multi-step-ahead forecasts.

one-step-ahead forecast

multi-step-ahead forecast

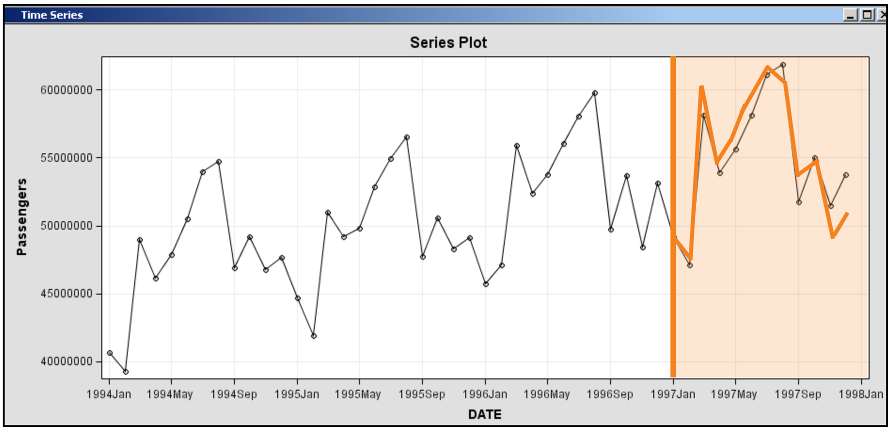
forecast error



In one-step-ahead forecasts, the forecast error is calculated for one time point only. Then the true value of that time point is added to the fit sample for forecasting the next value.

one-step-ahead forecast

multi-step-ahead forecast

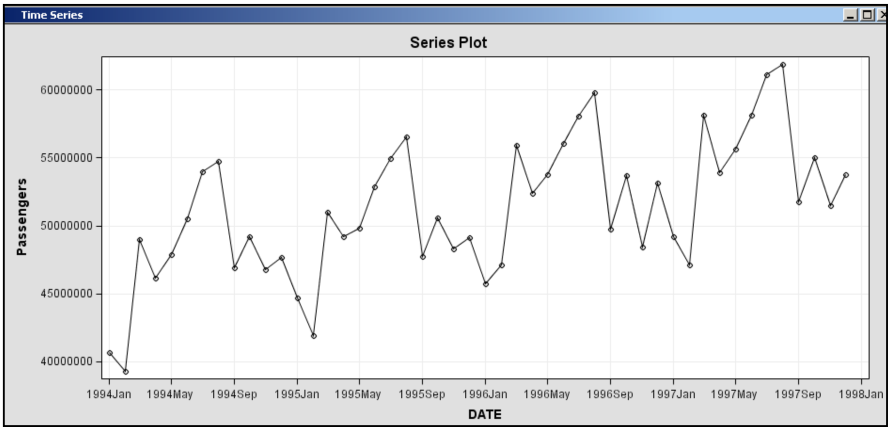


In multi-step-ahead forecasts, forecast values are generated for the entire forecast range without re-estimating the model parameters at each time point.

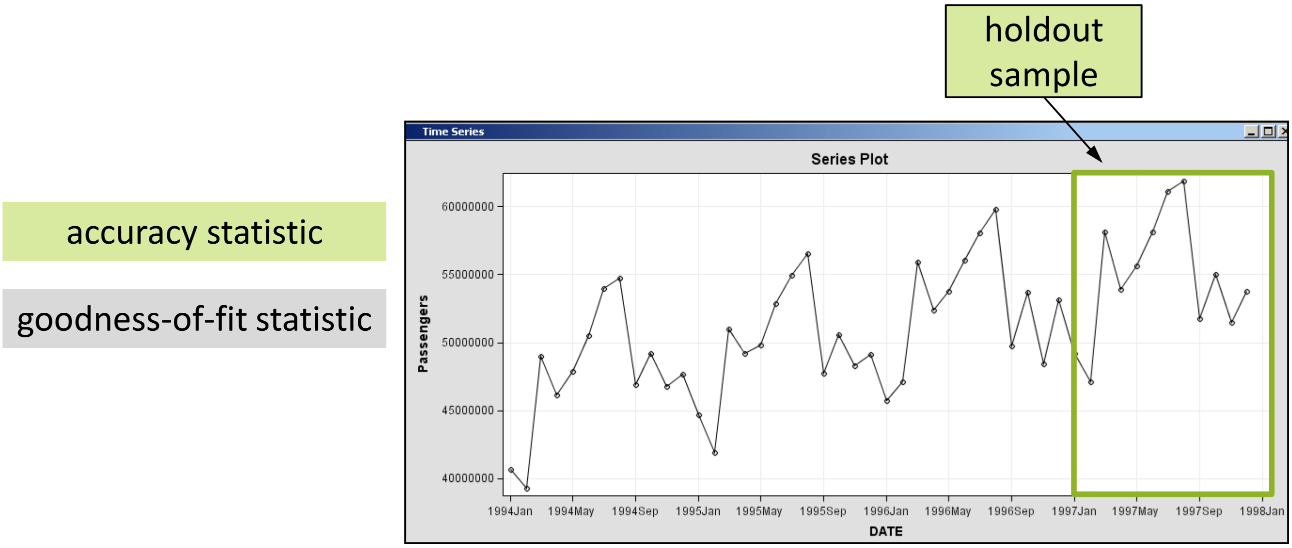
?

accuracy statistic

goodness-of-fit statistic



How do you know that you are performing an accuracy statistic versus a goodness-of-fit statistic?

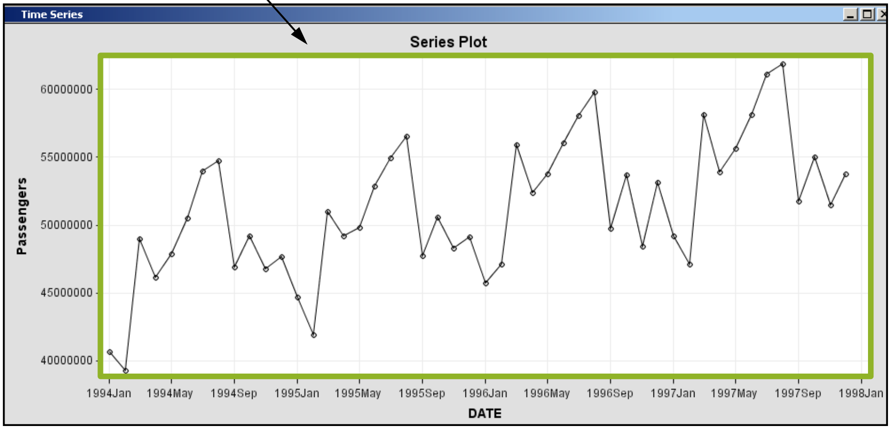


When you perform your calculations against an honest assessment holdout data set, that is classified as accuracy. This is because you are seeing how well your model from the fit sample adequately predicts data that it has not seen.

evaluation range

accuracy statistic

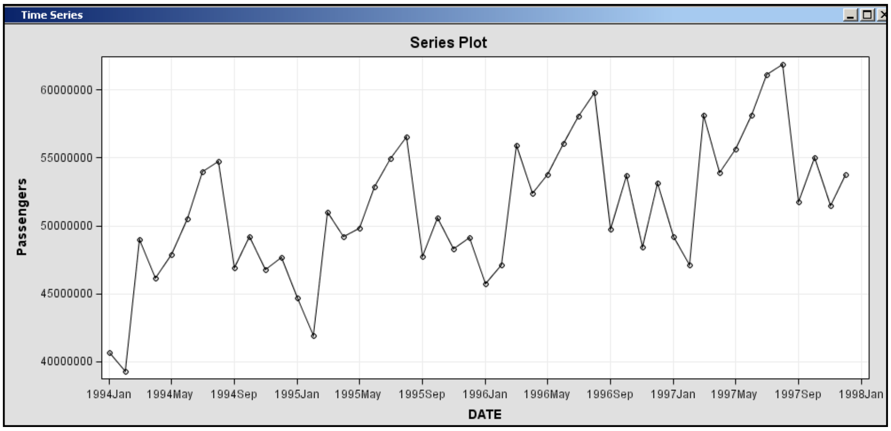
goodness-of-fit statistic



However, if the statistic is calculated against the full data set because holdout or honest assessment was not possible, this is a goodness-of-fit statistic. This is because you are seeing how well your model fits the data entirely, and all of this data was seen in the creation of the model.

accuracy statistic

goodness-of-fit statistic



All statistics that we are about to speak about can be used in either of these two manners.

Mean Absolute Percent Error (MAPE)

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n} |\, Y_t - \hat{Y}_t \,| \,/\, Y_t$$

MAPE → average of all of the individual absolute percent errors

Interpretation → average size of forecast error in percentage terms

- ✓ easy to assess accuracy
- ⊘ undefined when actual value of series is zero

Mean absolute percent error, or MAPE, is one of the most commonly used accuracy measures in business forecasting applications.

At each time point, the absolute value of the forecast error is divided by the actual series value. Thus, the absolute value of the forecast error is represented as a percent of the actual series value. The absolute percent errors are then summed across all time points in the assessment span, and the sum is divided by the number of measurements in the span.

As the average of all of the individual absolute percent errors, MAPE measures the size of forecast error relative to the magnitude of the actual value.

An advantage of using MAPE to assess the accuracy of the forecast is that a percentage is a standardized measure, which makes it relatively easy to assess accuracy based on the percentage value. A drawback of MAPE is that it behaves badly when the actual value of the series is zero at any time point in the assessment range. In these cases, MAPE is undefined.

Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n} | Y_t - \hat{Y}_t |$$

MAE → average of all of the individual absolute errors

Interpretation → average size of the forecast error in the units of the series

✓ almost always defined and finite

⊘ not as commonly used in business forecasting

Where MAPE is undefined, or even when its value is approaching infinity, a common alternative is the mean absolute error, or MAE.

Forecast errors in MAE are not divided by any values. So MAE is always defined and finite, unless a forecasted value is infinite. In that case, you have problems that go beyond accuracy statistics.

As the average of all of the individual absolute errors, MAE indicates the size of the forecast error.

MAPE

- ✓ easy to assess accuracy
- ⊘ undefined when actual value of series is zero
- ✓ standardized measure
- ✓ magnitude impacted by scale of series

MAE

- ✓ almost always defined and finite
- ⊘ not as commonly used in business forecasting
- ⊘ not a standardized measure
- ⊘ treats errors equivalently

Smaller values imply better accuracy.

For both MAPE and MAE, smaller values imply better forecast accuracy. However, there are some noteworthy differences between the two that make MAE less common.

Contrasted with MAPE, one drawback of MAE is that it is not a standardized measure. The range of MAE depends on the measurement scale of the series value.

Another disadvantage of MAE is that it treats absolute values of forecast errors equivalently, regardless of the seriousness of the error.

	Holiday Sales Day	Low Sales Day		
Actual	2,000	100		
Forecast	1,900	200	**Mean**	
APE	5%	100%	52.5%	**MAPE**
AE	100	100	100	**MAE**

An error of 100 on a large sales day is usually not as serious as an error of 100 on a low sales day, but MAE weights both equally.

For example, a forecast error of 100 when the actual series value is 2000 is not that great. A forecast error of 100 when the actual series value is 100 is quite serious. MAE treats both errors equivalently.

Root Mean Square Error (RMSE)

$$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n}(Y_t - \hat{Y}_t)^2, \quad \text{RMSE} = \sqrt{\text{MSE}}$$

RMSE →	square root of the average of all of the individual squared errors, adjusted for the number of estimated model parameters
Interpretation →	square root of the averaged squared forecast error

✓ commonly used in industrial, economic, and scientific forecasting.

Many other accuracy measures are available in Visual Forecasting. One additional measure is root mean square error, or RMSE. RMSE is commonly used as an accuracy measure in industrial, economic, and scientific forecasting.

In RMSE, at each time point in the assessment range, the forecast errors are squared. The squared errors are summed and the results are divided by the number of time points in the assessment range. This generates the mean square error, or MSE. Taking the square root of the MSE produces the RMSE. Like MAPE and MAE, smaller values imply better accuracy.

Answer: RMSE Meaning

How would you interpret a RMSE measure of 10?

- Over the range of data that the residuals are calculated on, the average difference between the forecast and actual is 10 units of the series.
- Over the range of data that the residuals are calculated on, the average difference between the forecast and actual is 10 percent.
- Over the range of data that the residuals are calculated on, the average difference between the forecast and actual is 10 standard deviations.
- Over the range of data that the residuals are calculated on, the average difference between the forecast and actual is 10 units of time.

Answer: a - The RMSE generates measures in the units of the series.

Demo: Honest Assessment

This demonstration illustrates model selection using accuracy measures.

HTML Placeholder

Summary of Honest Assessment

1. Divide the time series data into two segments.
    - The *fit sample* is used to derive a forecast model.
    - The *holdout sample* is used to evaluate forecast accuracy.

2. Derive a set of candidate models (ESM, ARIMAX, UCM, and so on).

3. Calculate the chosen model accuracy statistic for each model by using the fit sample to forecast the holdout sample.

4. Choose the model with the best accuracy statistic in the holdout sample.

5. Using the best model, generate forecasts for n future periods.

HTML Placeholder

Use of Accuracy Criteria

Two ways to judge accuracy:

1. Accuracy can be calculated for *one-step-ahead* forecasts over the ***entire range of the data***.

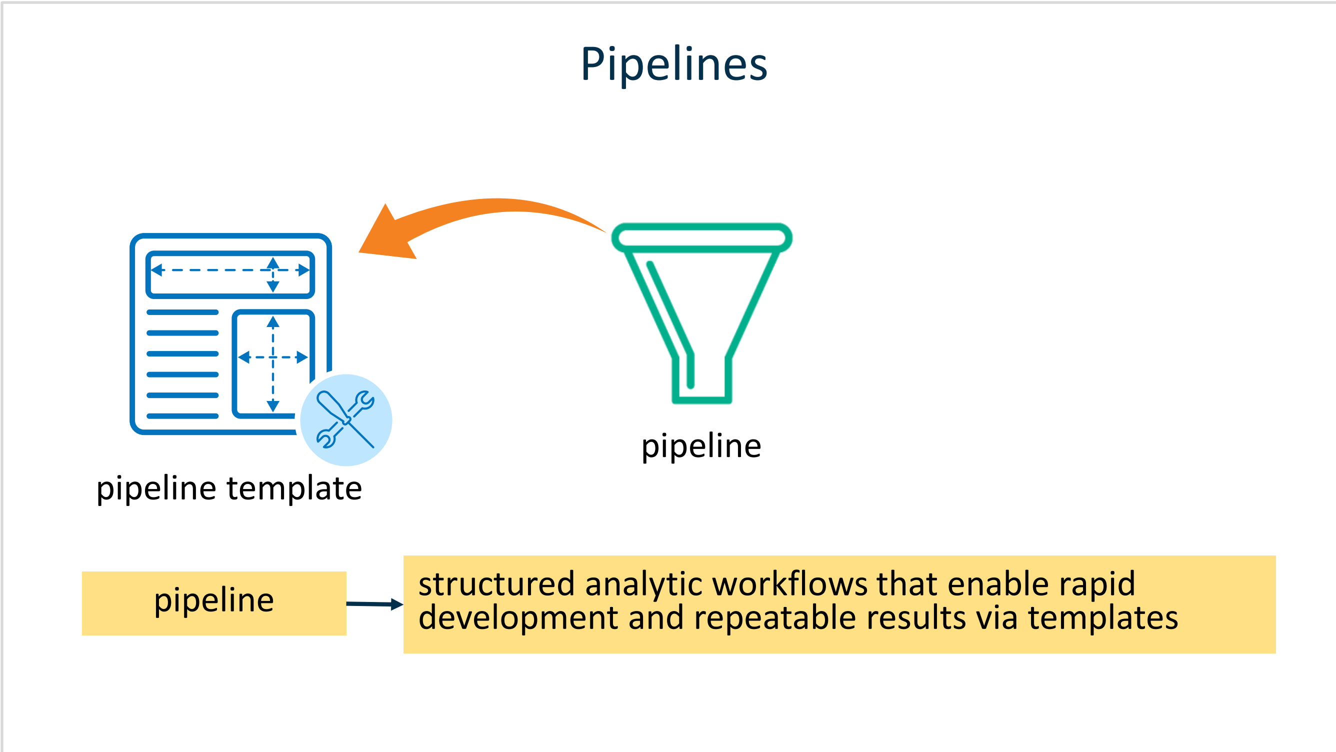
2. Accuracy can be calculated for a *holdout sample* of data at the ***end of each time series*** that was not used to construct models. A time series might be too short to enable use of a holdout sample. This method is preferred, but it is often not feasible. Using a holdout sample to judge accuracy is often referred to as *honest assessment* because it simulates fitting and deploying a model and then judging accuracy in the live environment.

Questions?

Pipeline Templates and Pipeline Comparison

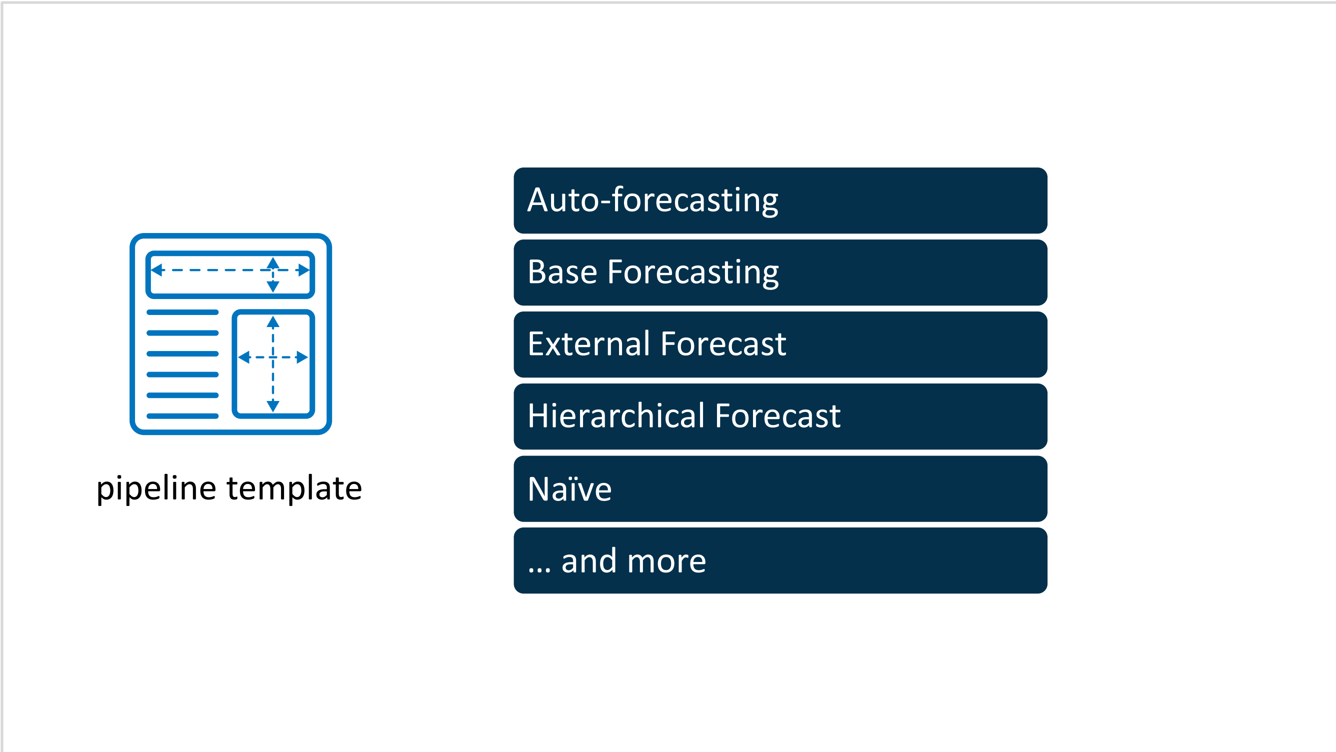
How to utilize pipeline templates and compare pipelines

Lesson 02, Section 05



Pipelines are structured analytic workflows that enable rapid development and repeatable results via templates.

Pipeline templates are also customizable, and you can share them.



You're already familiar with the auto-forecasting template, so let's look at some of the other pre-built forecast pipeline templates now.

The Base Forecasting template is a bare template. It simply includes a data node. All other nodes must be added manually.

The External Forecast template accommodates forecast models that are generated outside of Visual Forecasting. You can bring in forecasts, as well as confidence intervals and standard errors from those forecasting models. This enables you to compare models created externally with those created in Visual Forecasting.

Hierarchical Forecasting enables you to organize your forecast models into a hierarchy of attributes defined by your BY variables. You learn about these models in detail later.

The Naïve template includes mean and random walk models. These models are simplistic, but useful, for comparison with other models. If a complex model does not perform better than the naïve model, then the complex model isn't worth the extra work.

In addition to these pipeline templates, there are a number of others available in SAS Visual Forecasting to streamline your workflow.

Demo: Exploring More Pipeline Templates

This demonstration illustrates the use of the Auto-Forecasting template.

Questions?

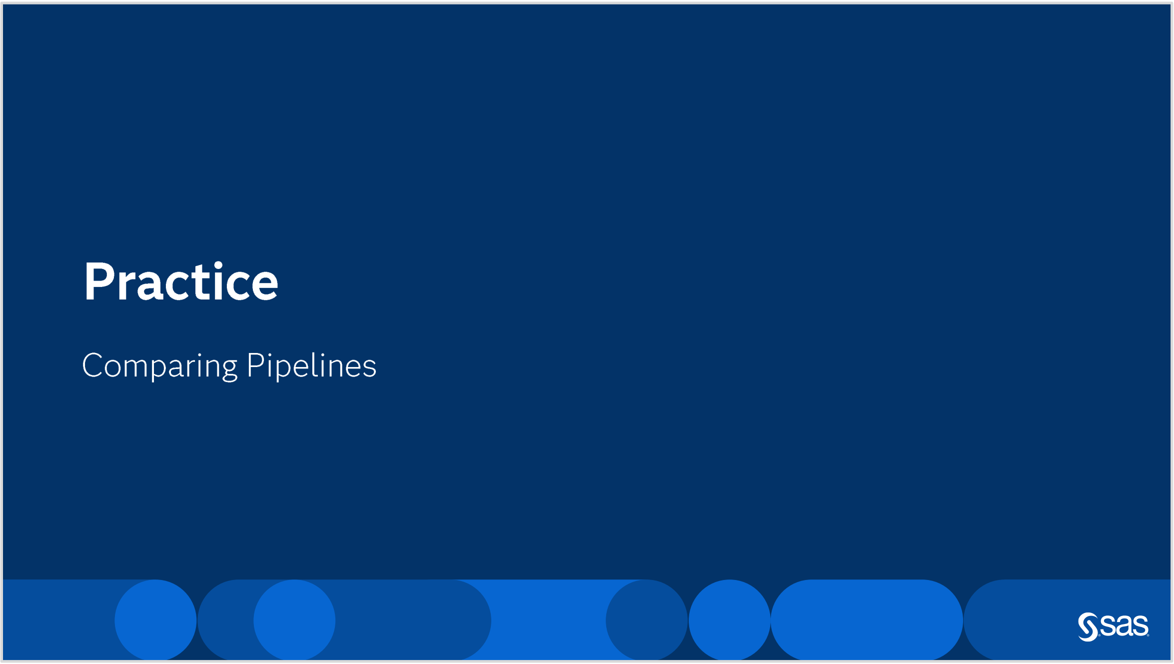
Practice

Working with Pipelines

Demo: Pipeline Comparison

This demonstration illustrates comparing pipelines.

Questions?

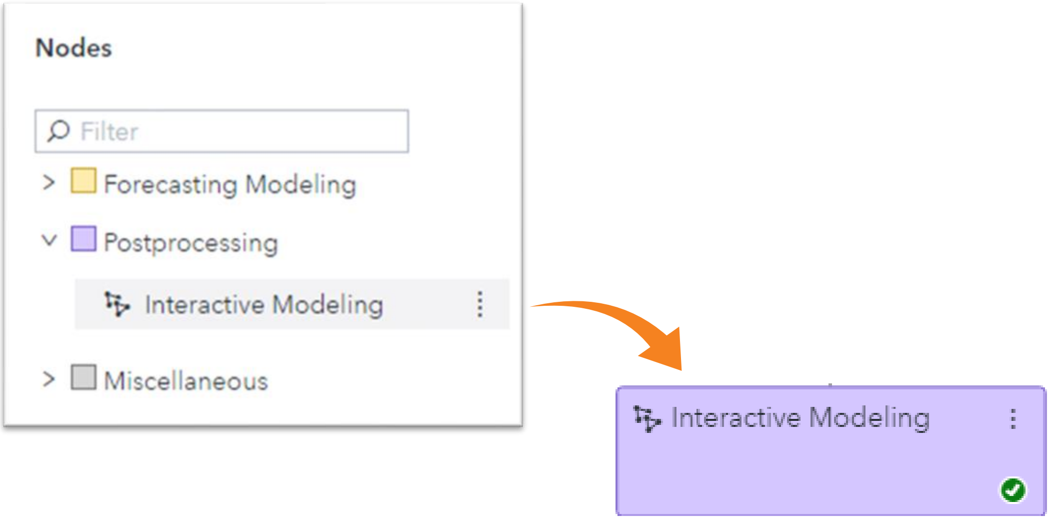

Practice
Comparing Pipelines
sas

Interactive Modeling Node

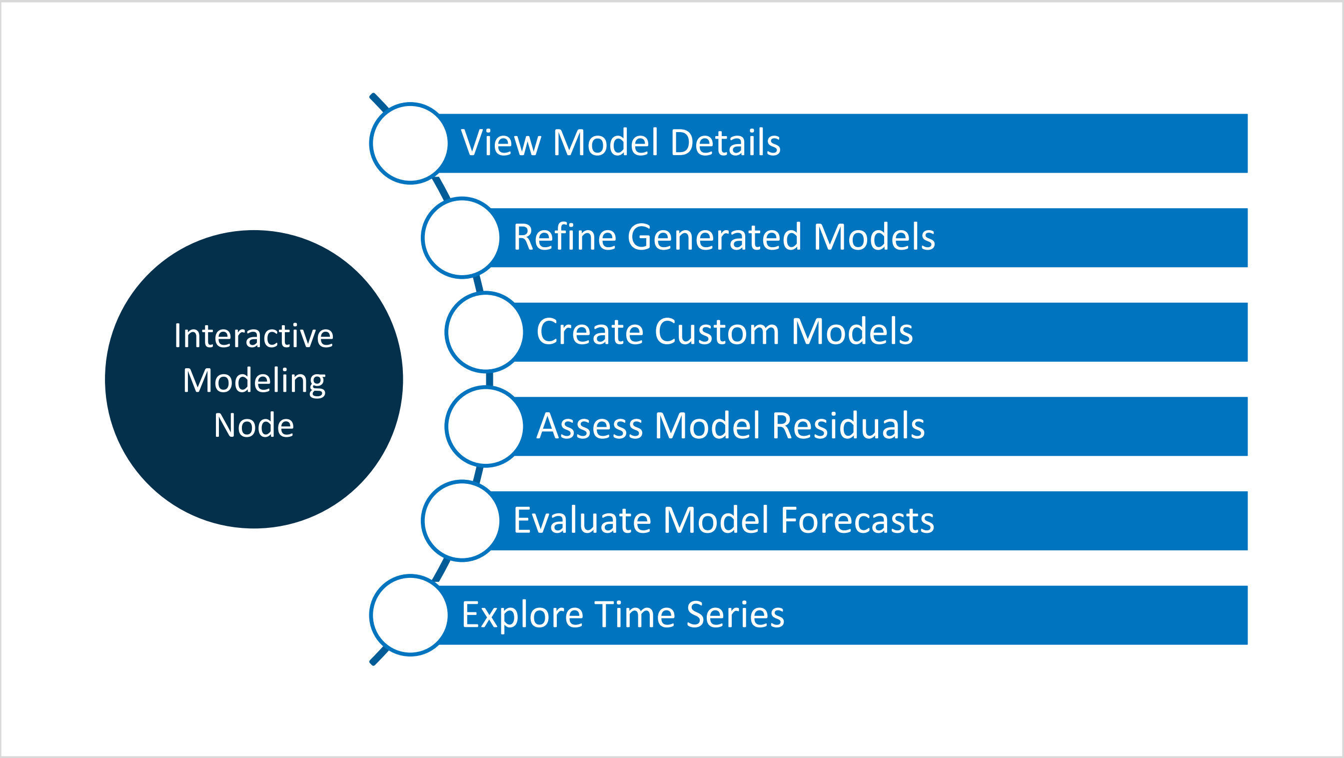
How to use the interactive modeling node when analyzing a time series

Lesson 02, Section 06

Interactive Modeling Node



The Interactive Modeling node is a Postprocessing node that attaches to a modeling node in a pipeline. It enables users to analyze and adjust models on a series-by-series basis.



The Interactive Modeling node contains details on the candidate models in the project. It lists parameter estimates and other diagnostics for all models on a selected series' Model Selection List. Additional functionality lets analysts modify and refine generated model specifications as well as create new, custom model specifications. The three views contained in the node provide a comprehensive look at the models considered for a selected series, the forecasts they generate, and the variation in the underlying time series.

Demo: Creating Custom Models with the Interactive Modeling Node

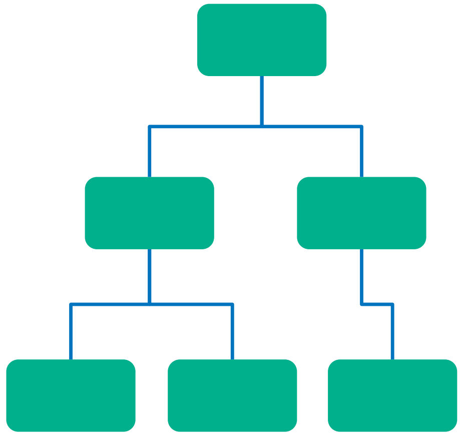
This demonstration illustrates creating custom models with the Interactive Modeling node.

Building a Hierarchical Forecasting Model

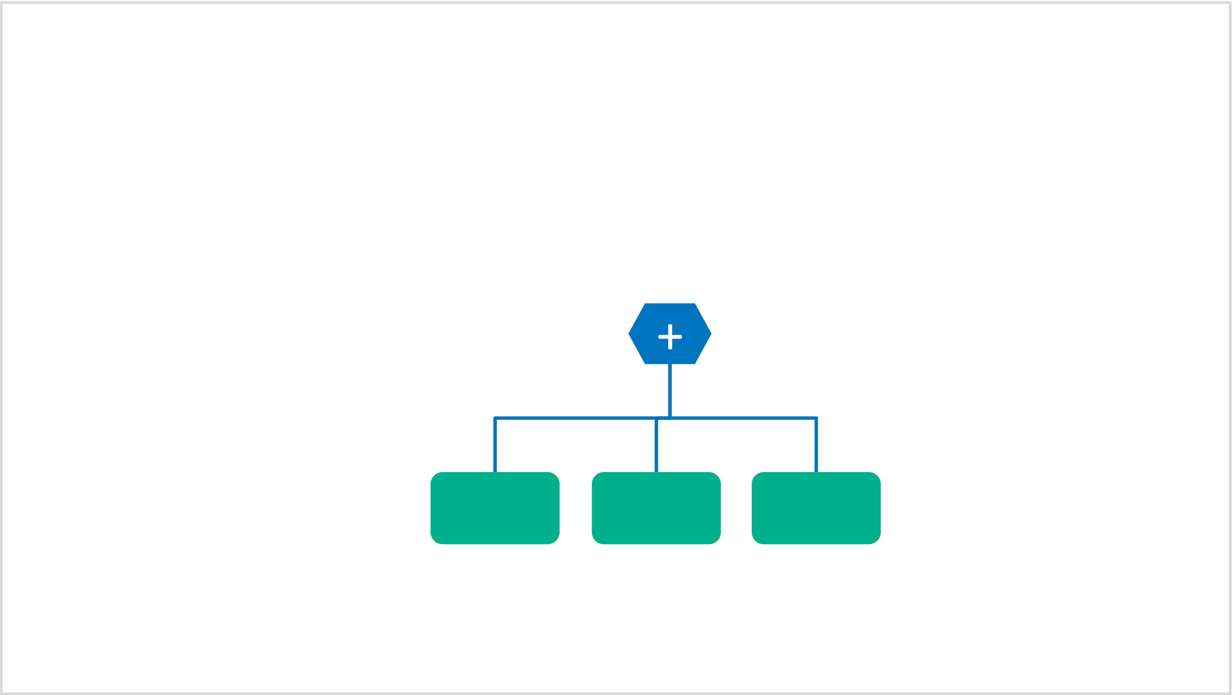
How to build a hierarchical forecasting model

Lesson 03, Section 01

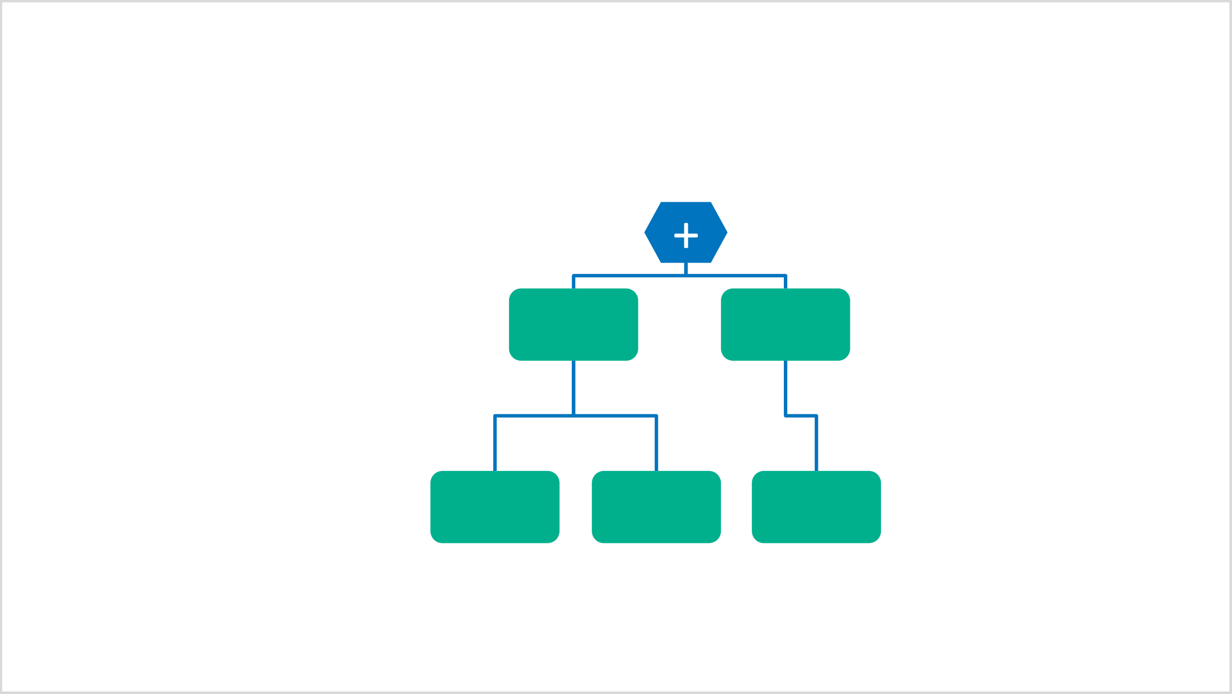
Hierarchical Forecasting



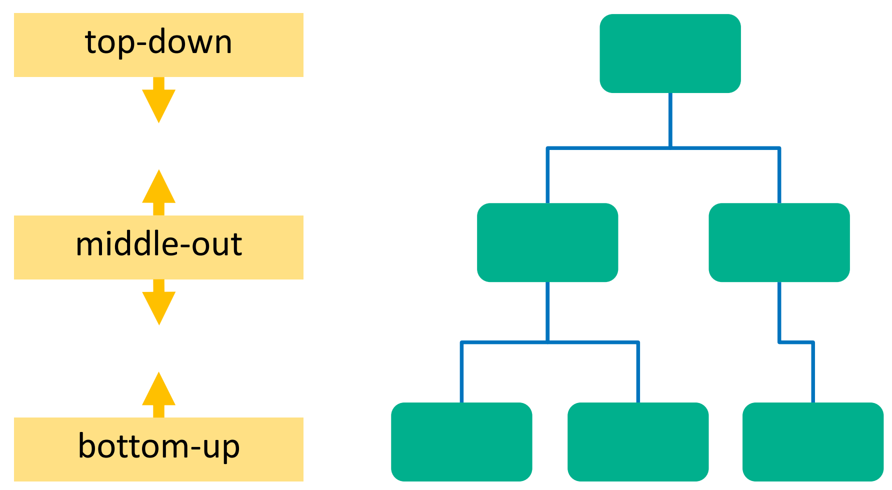
Modeling data with a hierarchical structure adds complexity and tasks to the forecasting process.



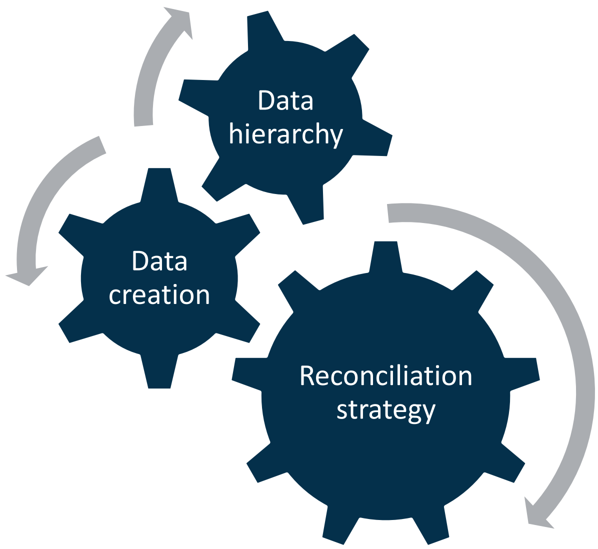
On the data preparation end, each transactional series in the base level of the hierarchy needs to be accumulated to an equally spaced interval.



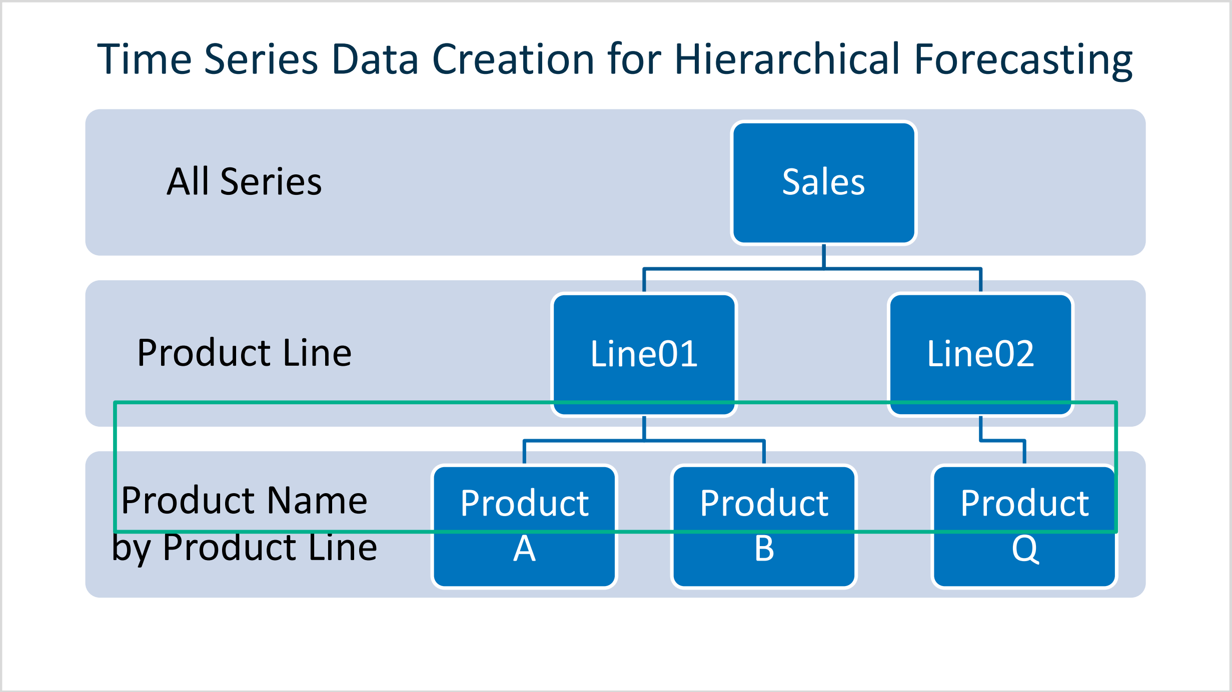
After this, time series data in the upper levels of the hierarchy are constructed from the base level data through the process of aggregation.



Usually, forecasters analyzing hierarchical data want their statistical forecasts to reconcile. The process of reconciliation is based on the choice of reconciliation type: bottom-up, middle-out, or top-down. Given a reconciliation type, forecast reconciliation is performed using the method of forecast proportions by default. We will explore reconciliation options available in the software.



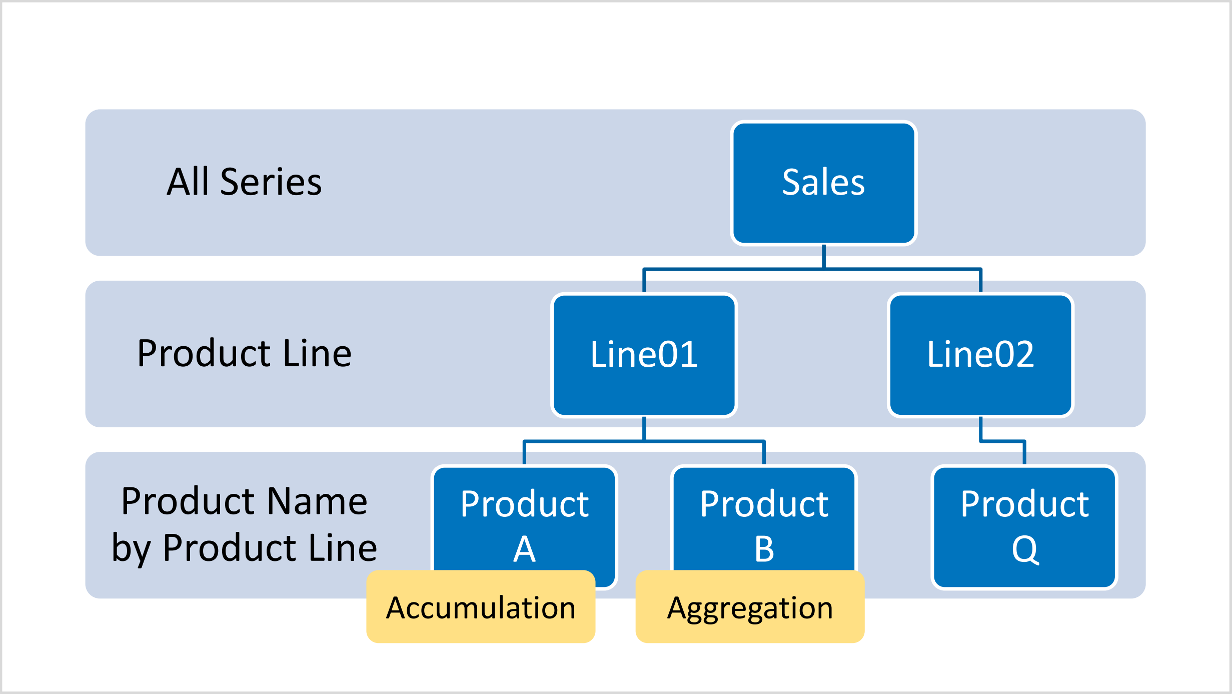
Finally, choices related to which products to include in the data hierarchy, data creation, and the implemented reconciliation strategy interact. These interactions have an impact on a project's forecasting performance.



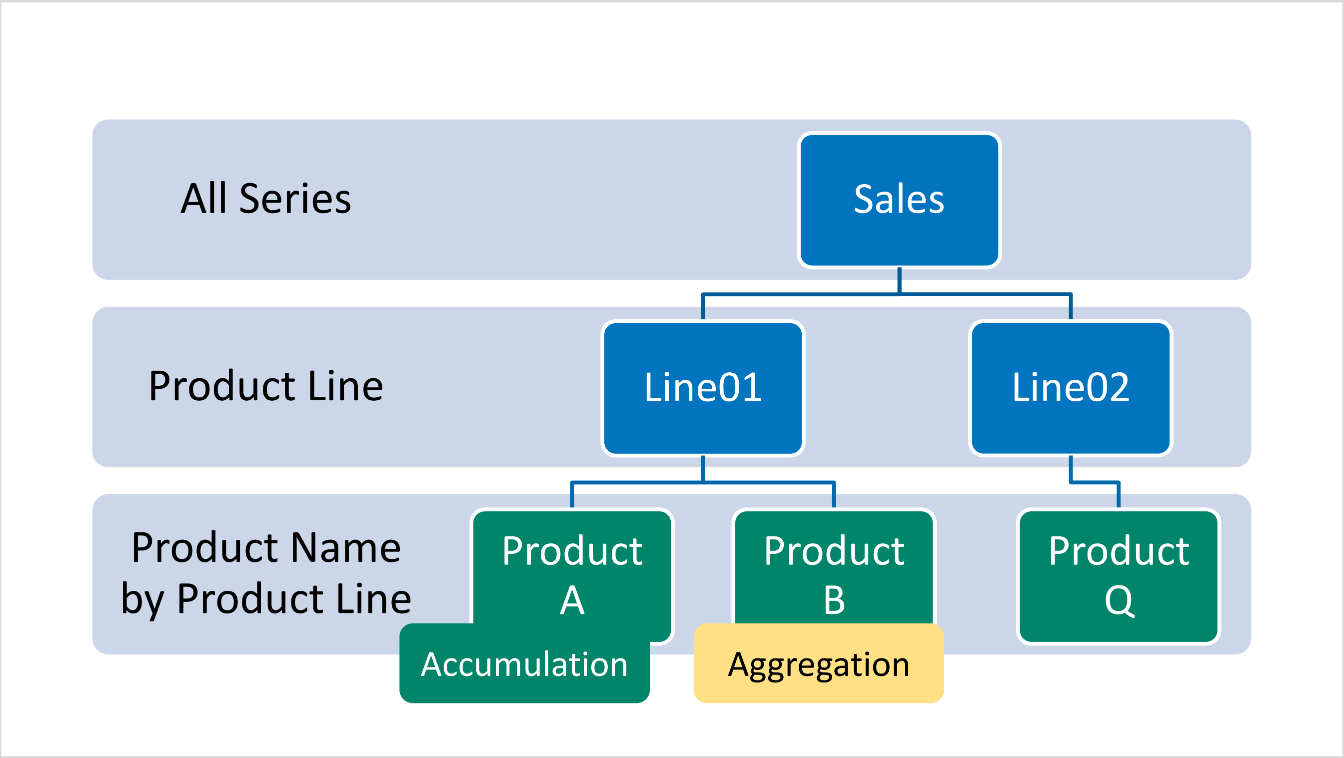
Recall that when you created the baseline sales forecast project, you defined two variables in the time series data as BY variables, **productline** and **productname**. At that time, we mentioned that SAS Visual Forecasting uses these primary attributes to define the hierarchical structure of the time series.

You included additional attribute variables, but they didn't define the hierarchy.

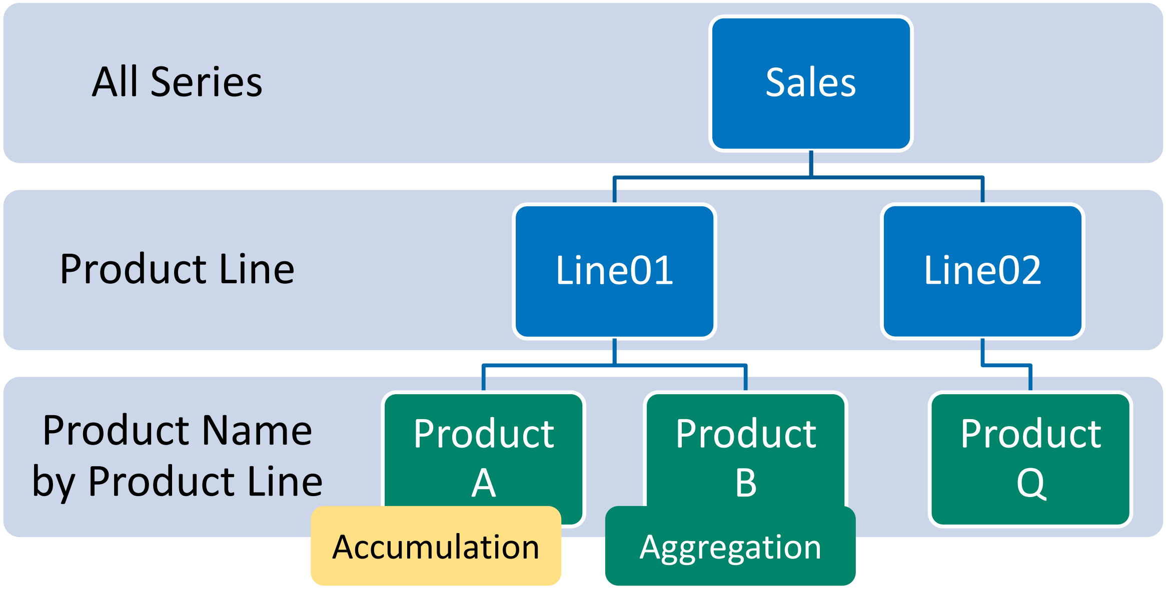
So far, you have seen results only at the base level of the hierarchy, where the series is broken down by **productline** and **productname** within **productline** . Let's discuss true hierarchical forecasting.



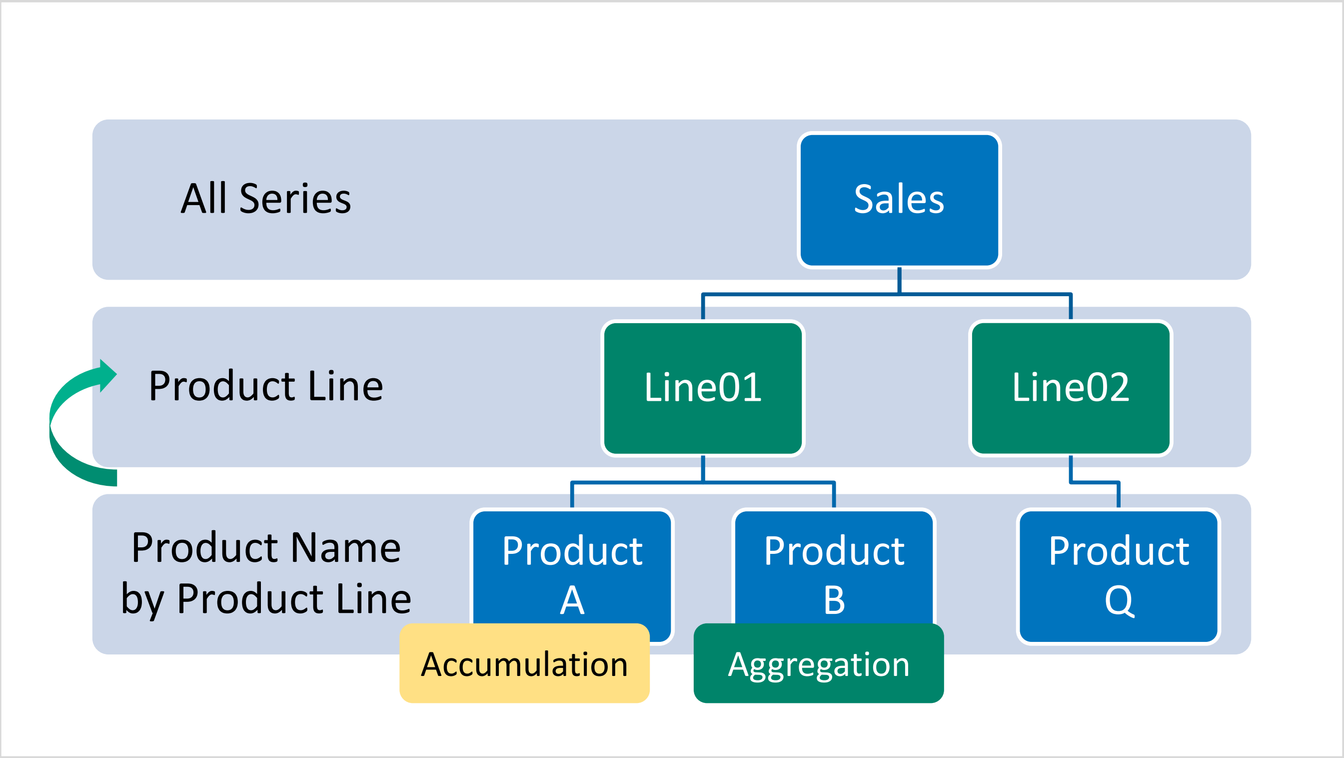
There are two options for creating time series data from transactional data in Model Studio: accumulation and aggregation.



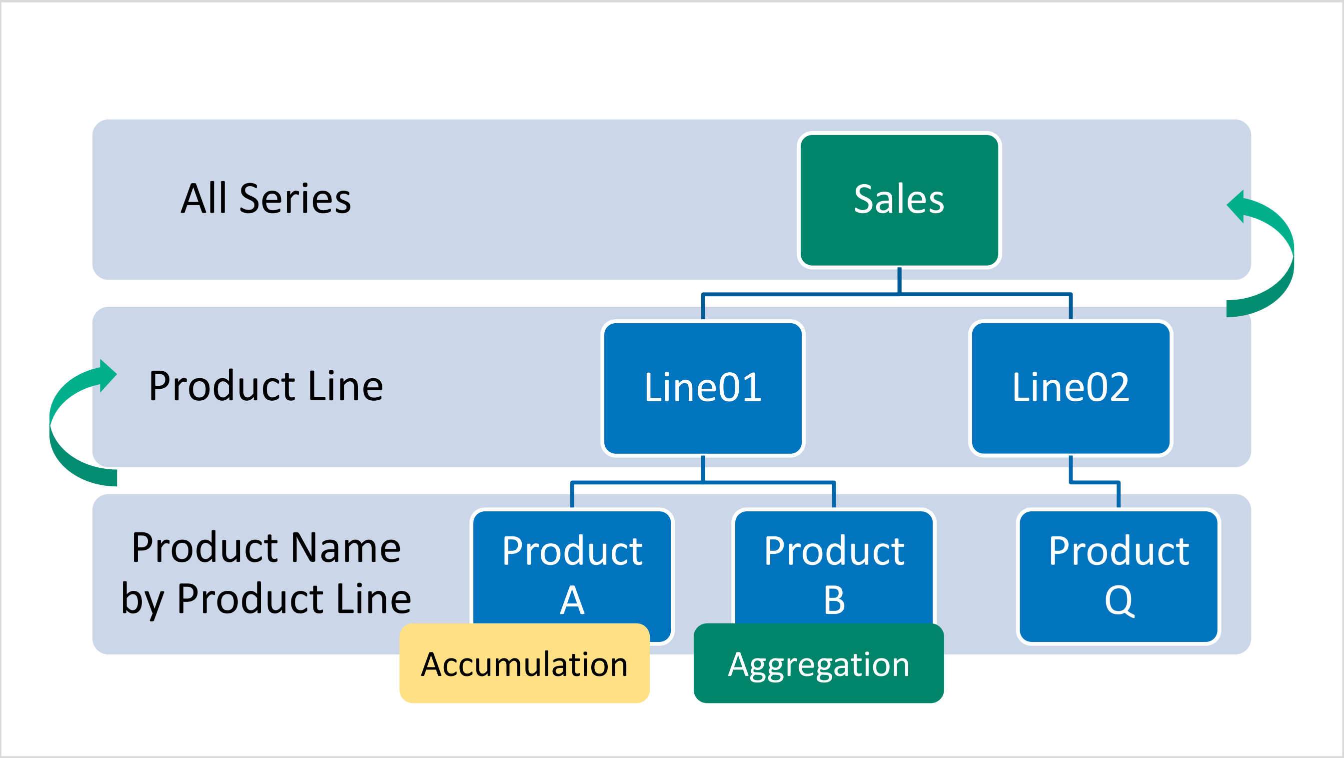
You already learned about accumulation. Data accumulation produces the time series at the bottom level of the hierarchy from transactional series in the data.



Data aggregation constructs the data hierarchy by aggregating the time series in the bottom level of the hierarchy.

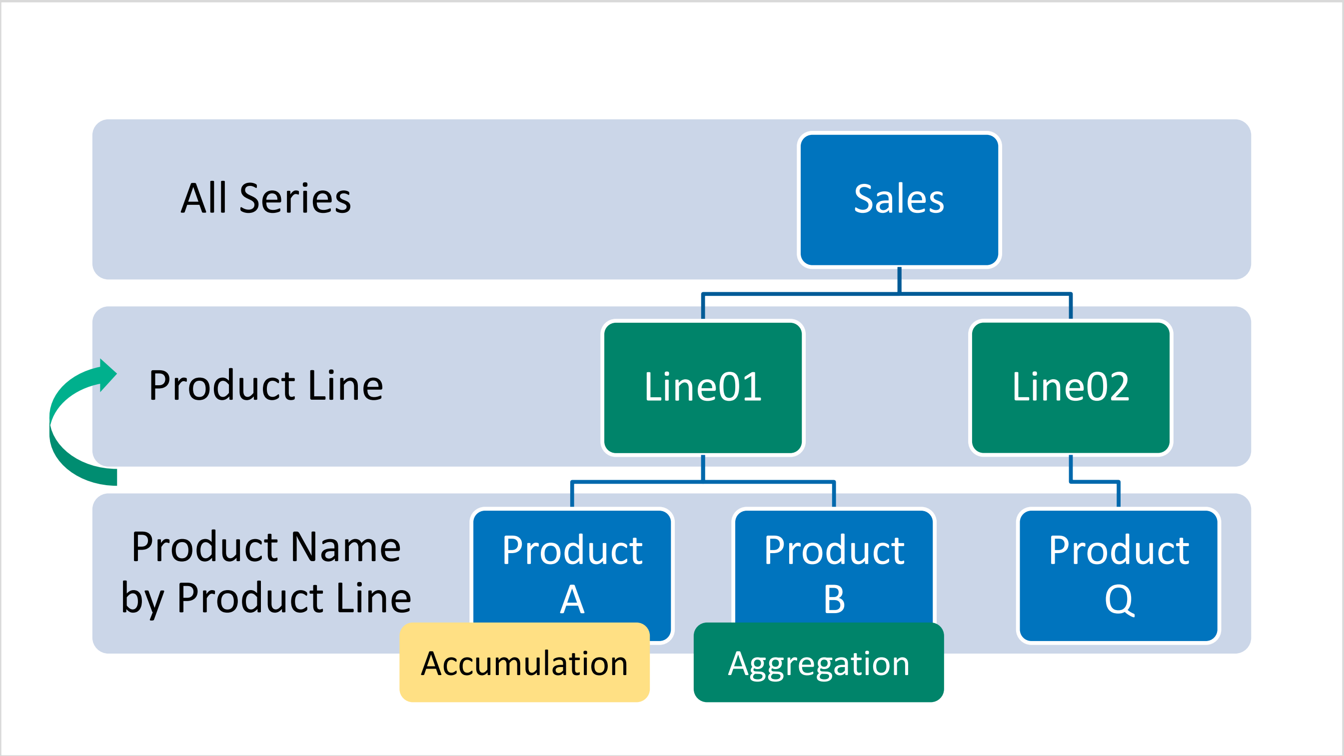


After the time series are created at the bottom level, you can create the middle level of the hierarchy by summarizing the series at the base level.

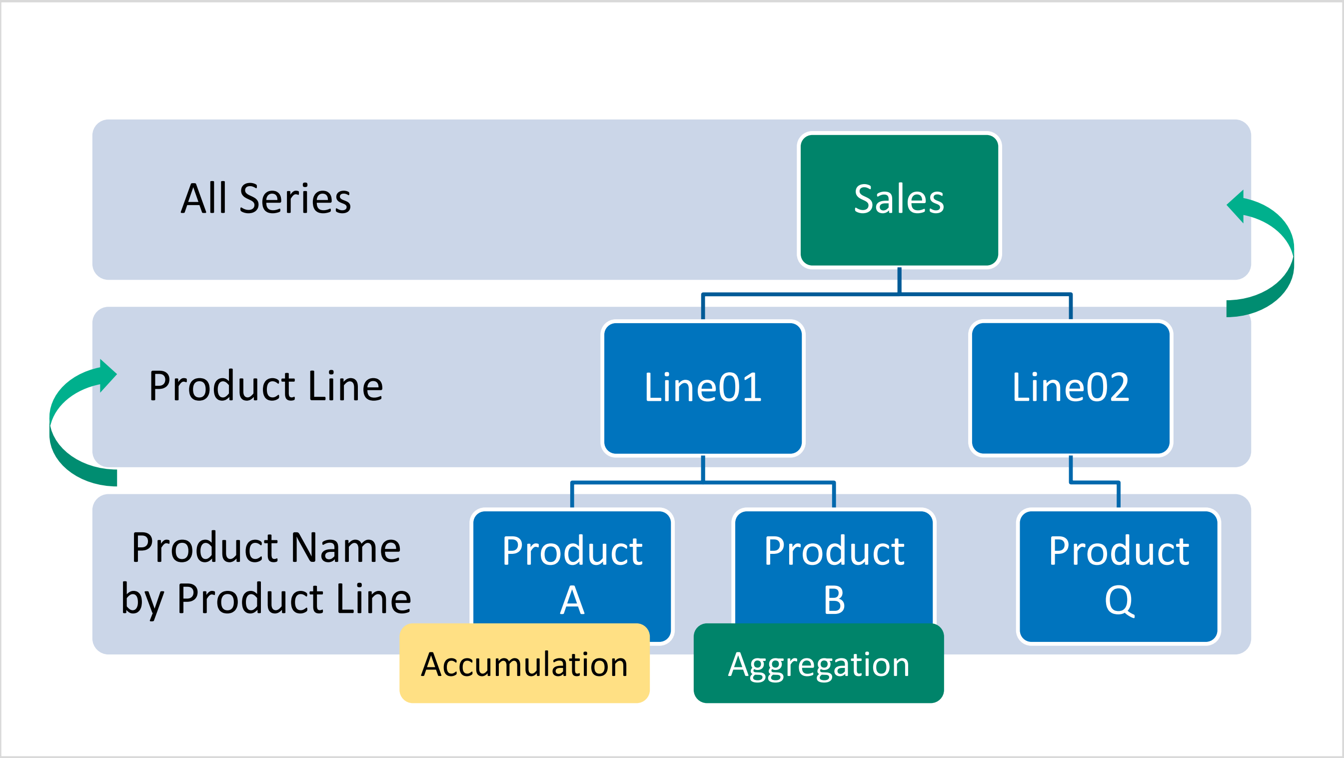


After the middle level series are created, you can create the top level series by summarizing the series at the middle level.

Aggregation produces time series in the middle and top levels of the hierarchy according to BY groups in the data. A statistical forecast is then produced for every series in every level of the data hierarchy.



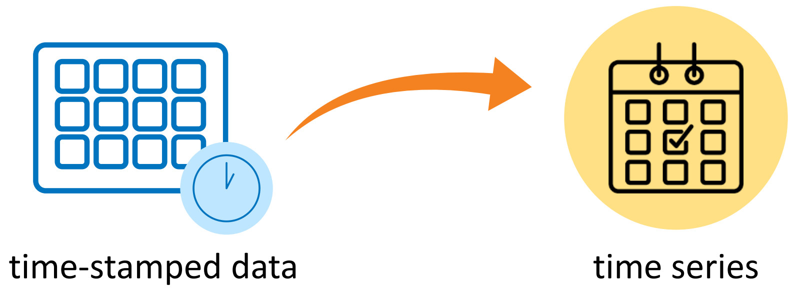
In our example, aggregation of the base level results produces product line series in the middle level of the hierarchy



and total sales in the top level of the hierarchy.

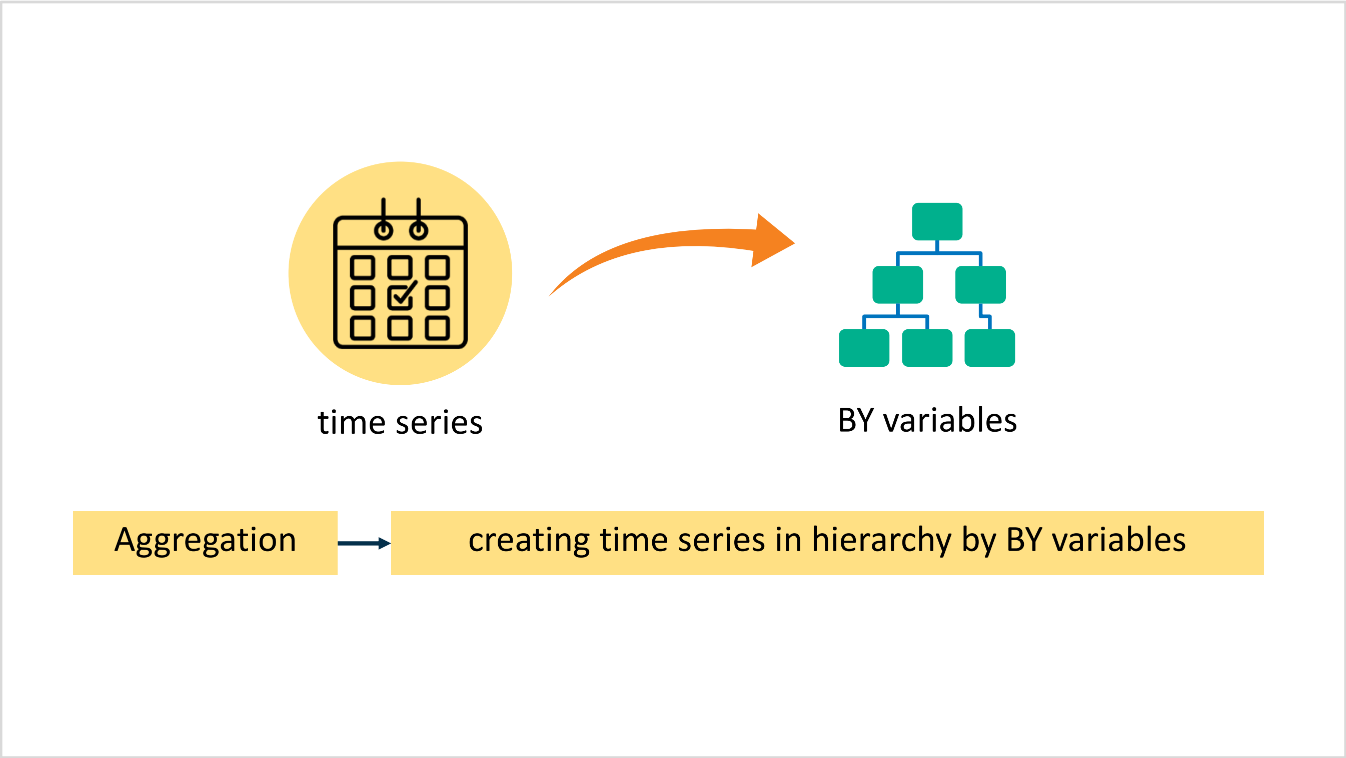
The sales for each month across all the different BY groups are summed.

Accumulation and Aggregation Options in SAS Visual Forecasting

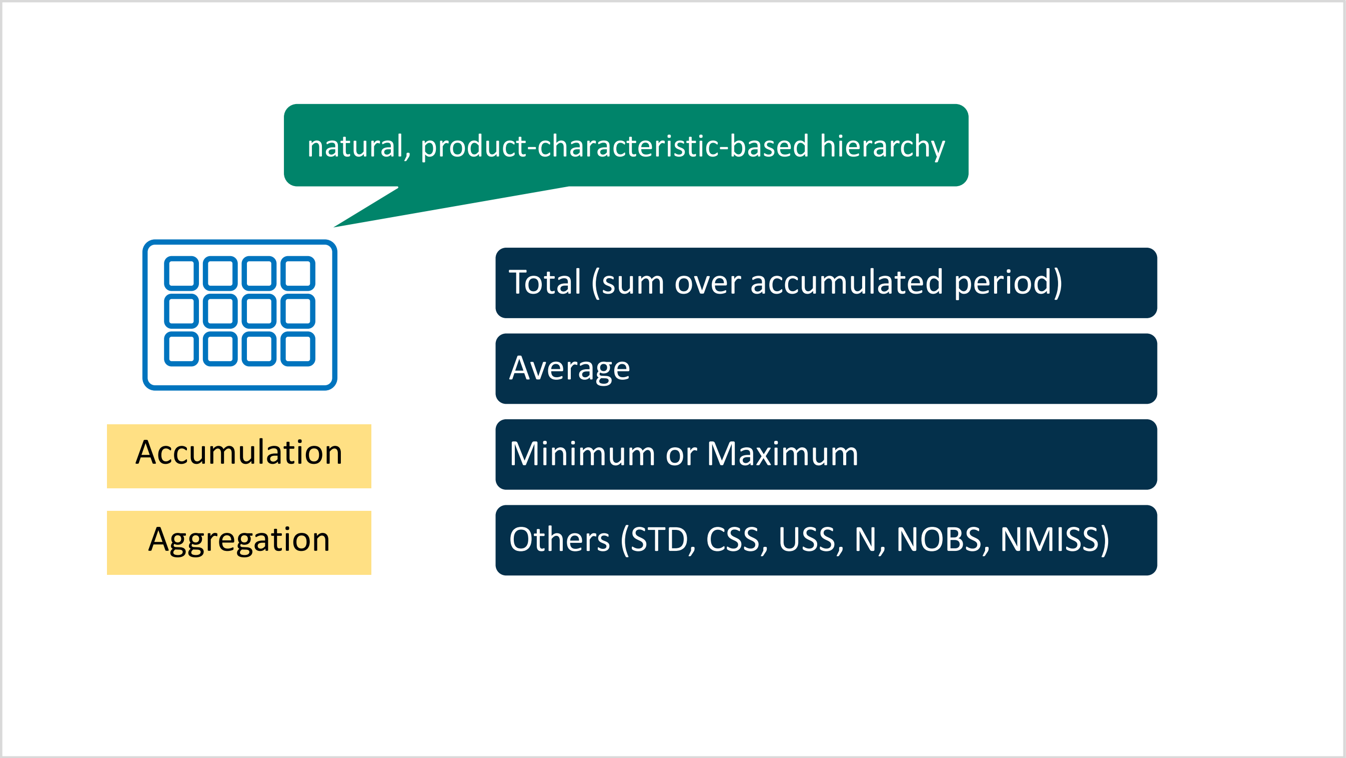


Accumulation → summarizing data within equally spaced time intervals

Accumulation is the process of taking timestamped data and rolling them up into time series by summarizing data within equally spaced time intervals.



Aggregation is the process of summarizing the time series themselves in a hierarchy defined by BY variables. Accumulation and aggregation are processes that create the series, so they must occur before any modeling is performed.



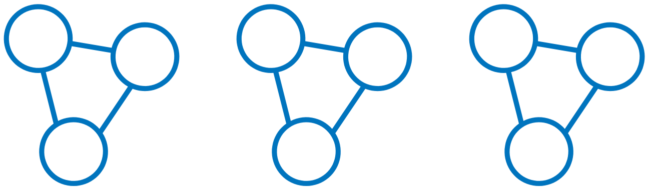
Although accumulation and aggregation options are identical, there are situations in which you will want to choose different options to operate on different levels of the hierarchy. The data used in this lesson's demonstration have a natural, product-characteristic-based hierarchy.

Total or sum is appropriate for the target in the case of demand. If your target is price or cost, there are times when average would be a better representation.

In instances where you have disproportionate costs, you might want to summarize according to minimum or maximum. This method builds in a safety measure by forcing you to over-forecast or under-forecast on average.

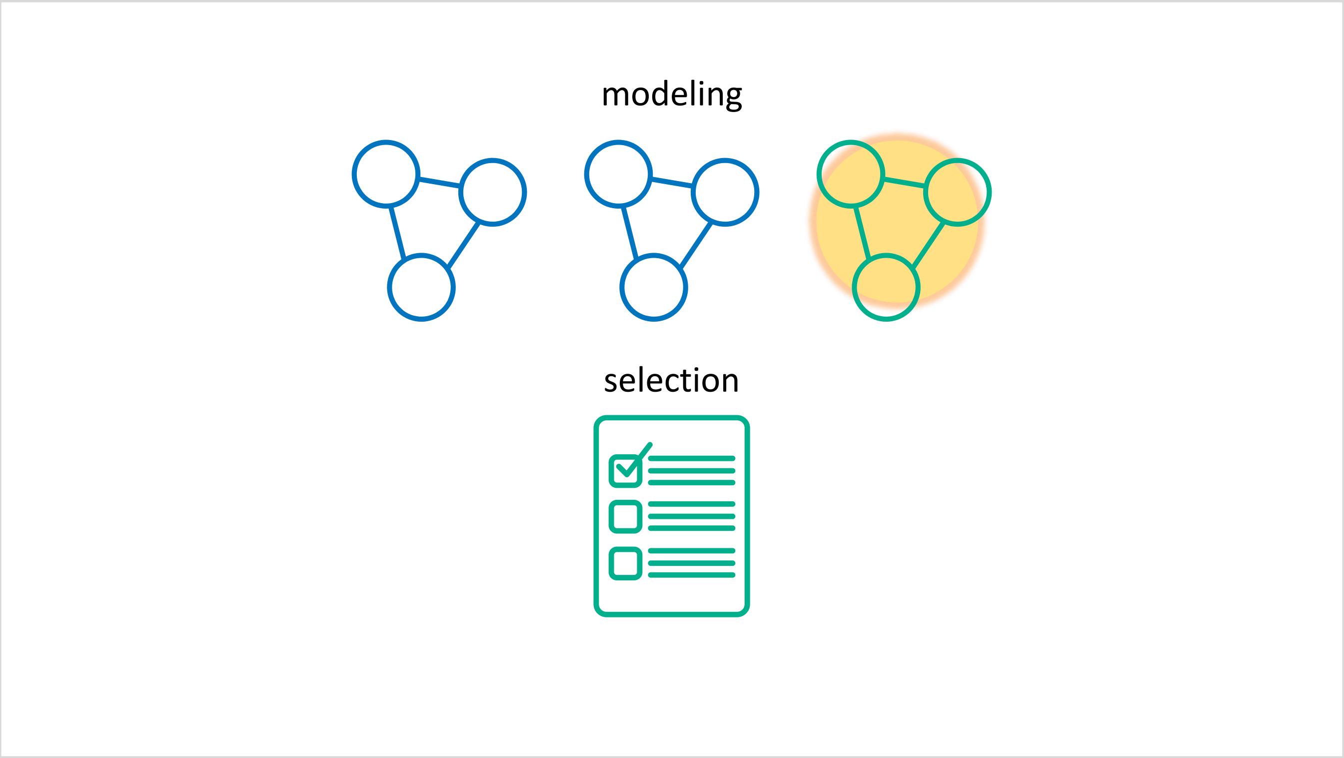
Forecast Reconciliation

modeling

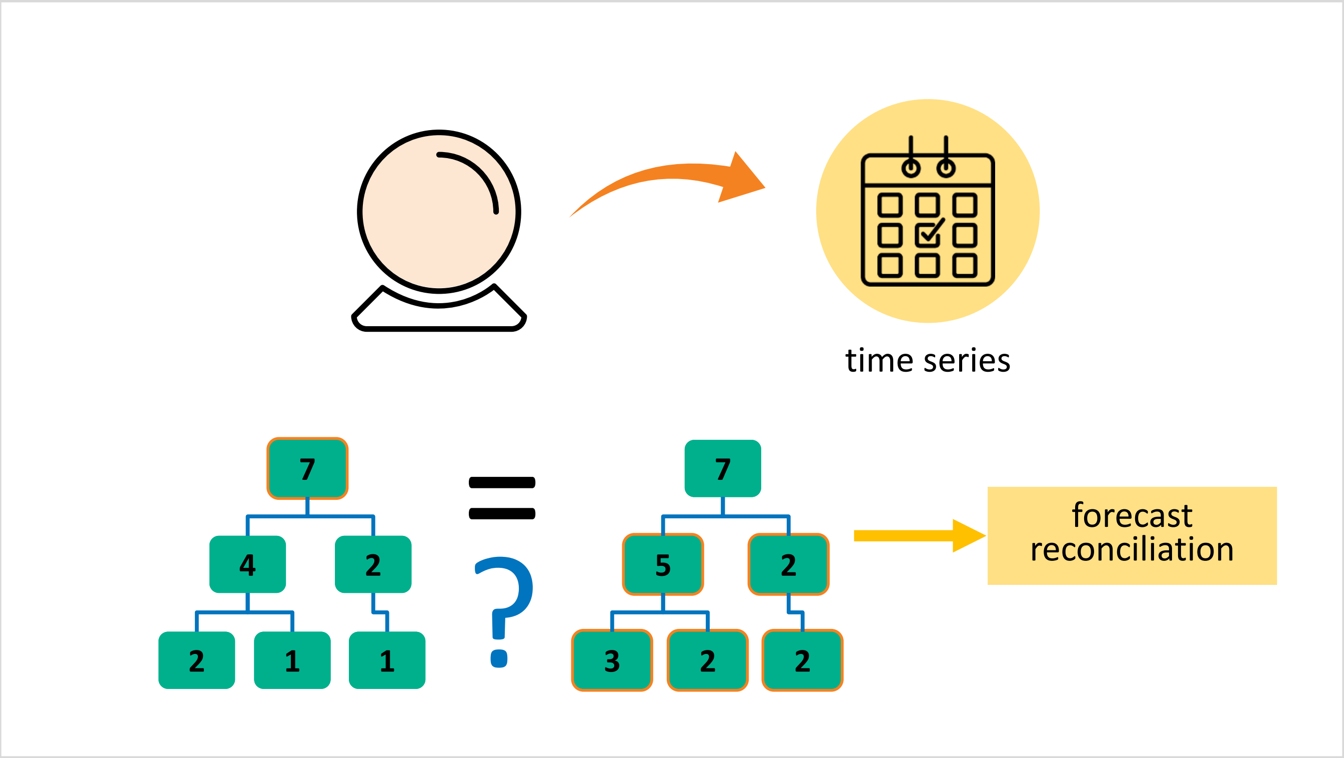


selection

After the series are created with accumulation or aggregation, modeling occurs, and many of the models that you've learned about are tested.



The champion models are selected for each series using honest assessment on a holdout sample, if possible, or accuracy on the fit sample if data splitting isn't feasible.

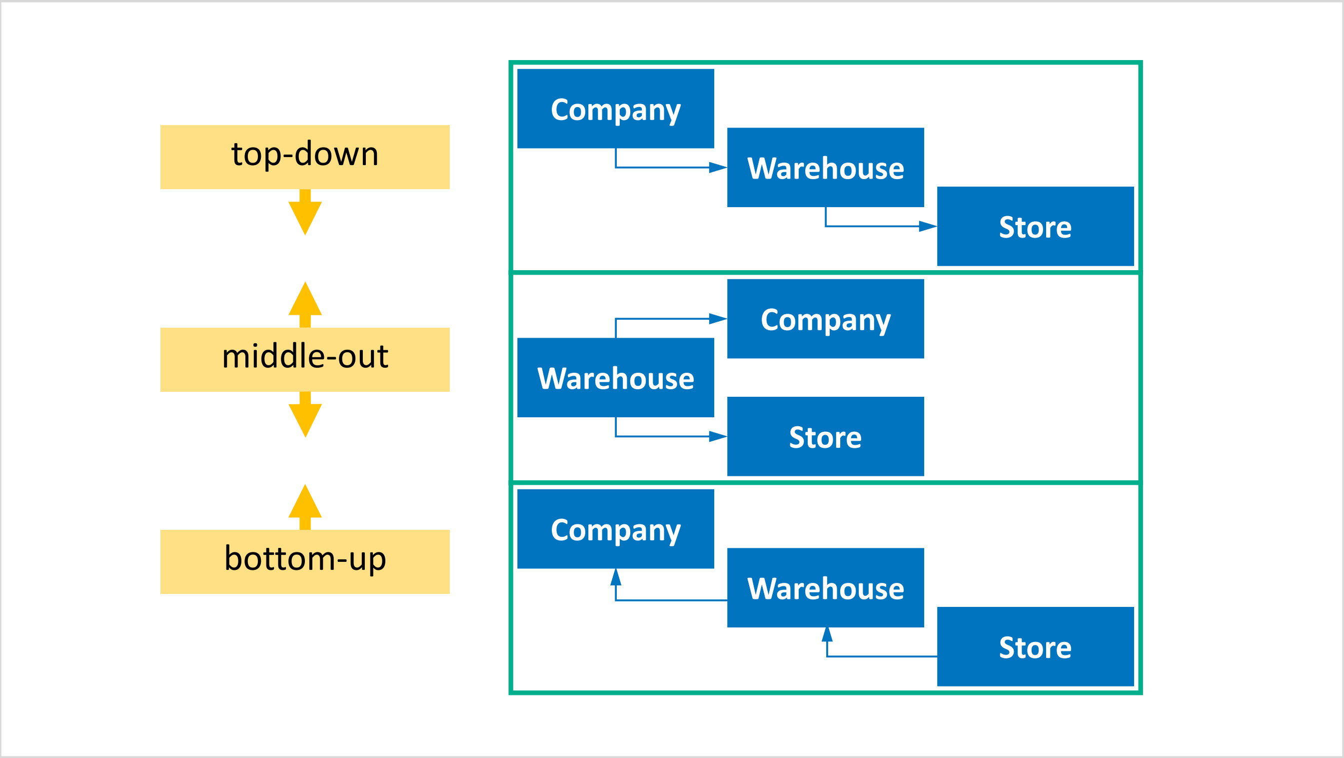


At this point, forecasts for the future are generated for all series.

However, there is no guarantee that the aggregation of forecasts from the lower to the upper levels of the hierarchy will match the actual forecasts from those upper levels.

Inconsistencies must be resolved before you can deploy the models.

The process of resolving the inconsistencies between levels of a hierarchy is called *forecast reconciliation*. In contrast to data creation, forecast reconciliation occurs at the other end of the forecasting process.

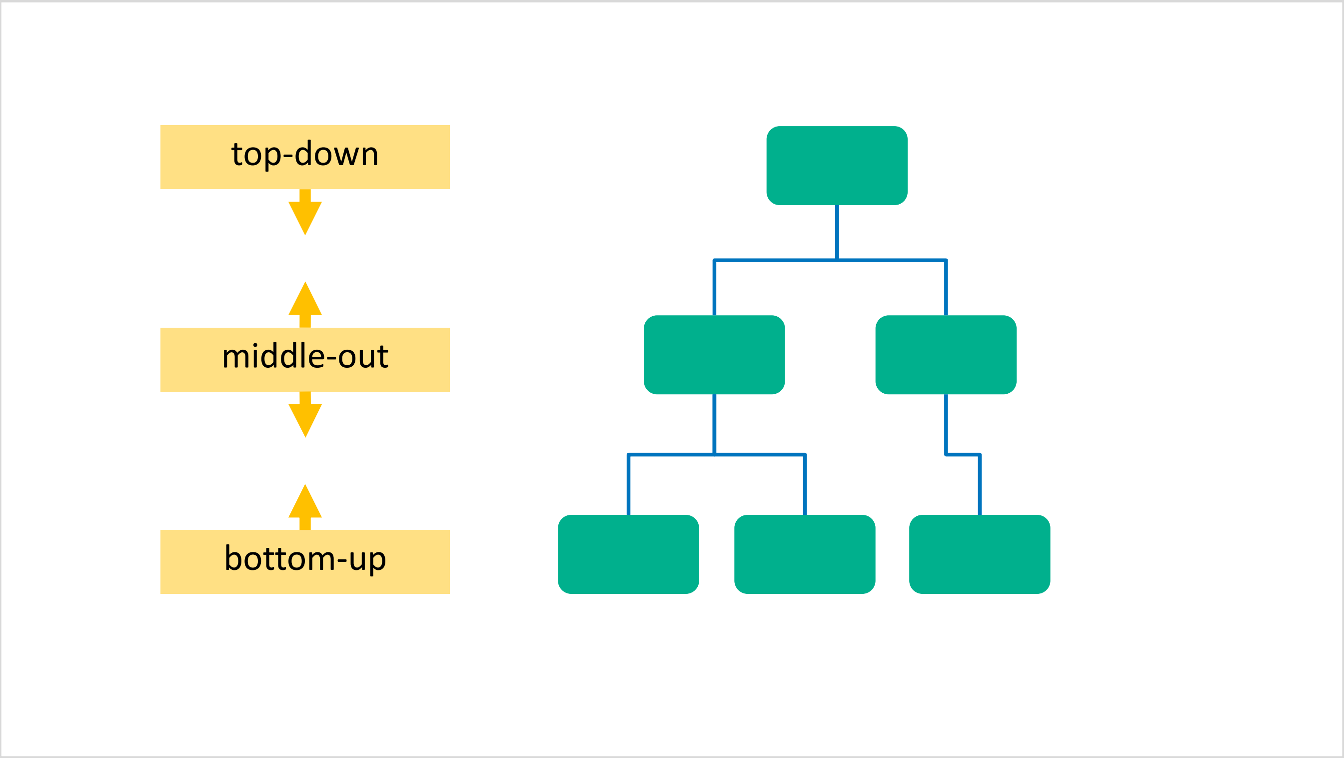


The three main reconciliation approaches are top-down, middle-out, and bottom-up. These approaches are fairly straightforward.

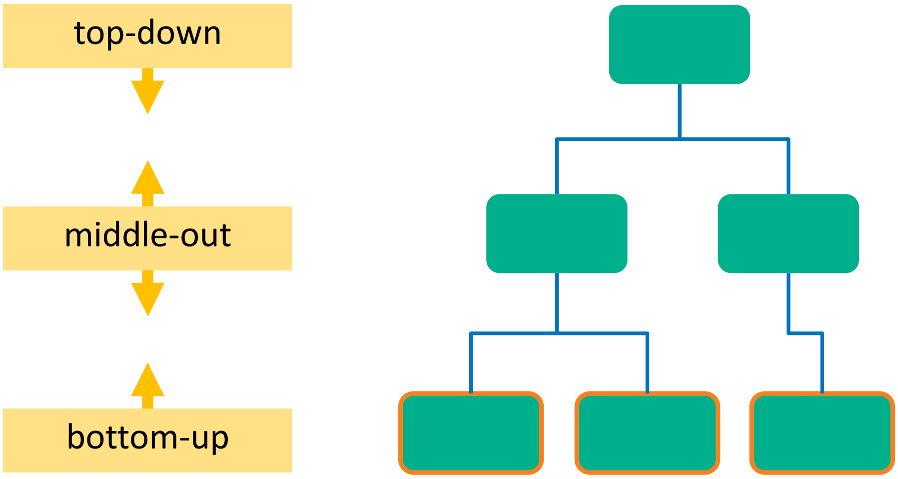
The top-down approach treats the statistical forecast at the top level as unchangeable. We adjust the middle-level forecasts to accommodate the top-level forecast, and then modify the bottom-level forecast to be consistent with the middle-level forecast.

When you use the middle-out approach, you select the middle-level forecast to be immutable. You resolve any inconsistences with the other levels of the hierarchy by adjusting those levels. There might be more than one middle level of the hierarchy. In this case, you specify which of the middle levels is held constant for reconciliation.

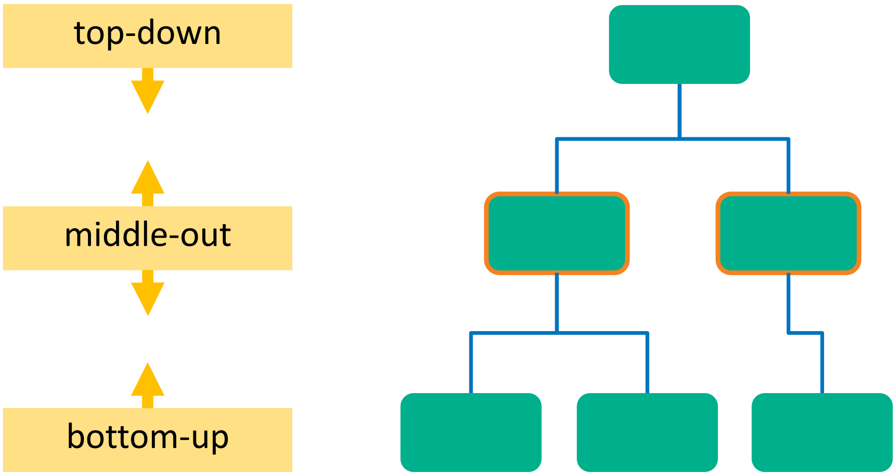
Finally, the bottom-up approach protects the bottom level of the hierarchy. You make adjustments to the forecasts at the middle level first to achieve consistency, and then you make adjustments to the highest-level forecasts to achieve consistency with the middle-level forecasts.



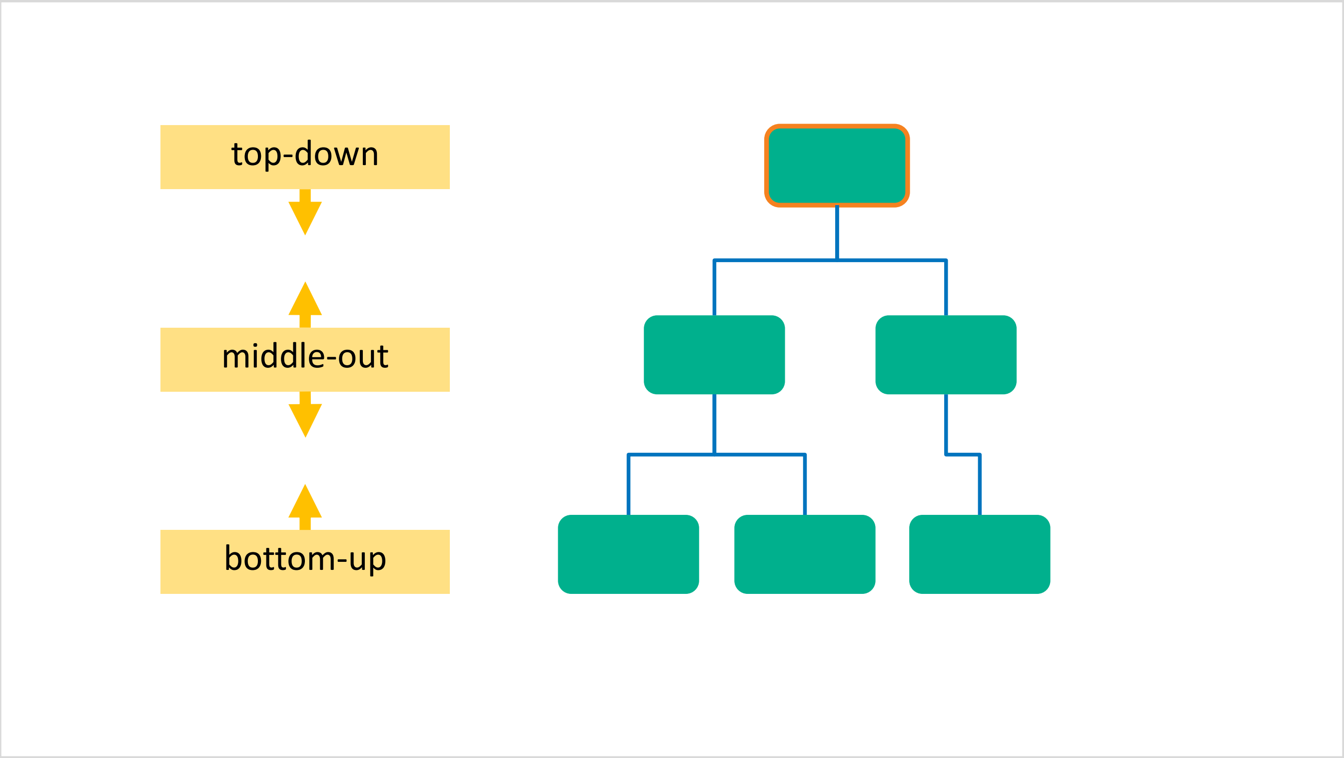
The data creation methods that we have discussed can play a role in choosing a reconciliation strategy.



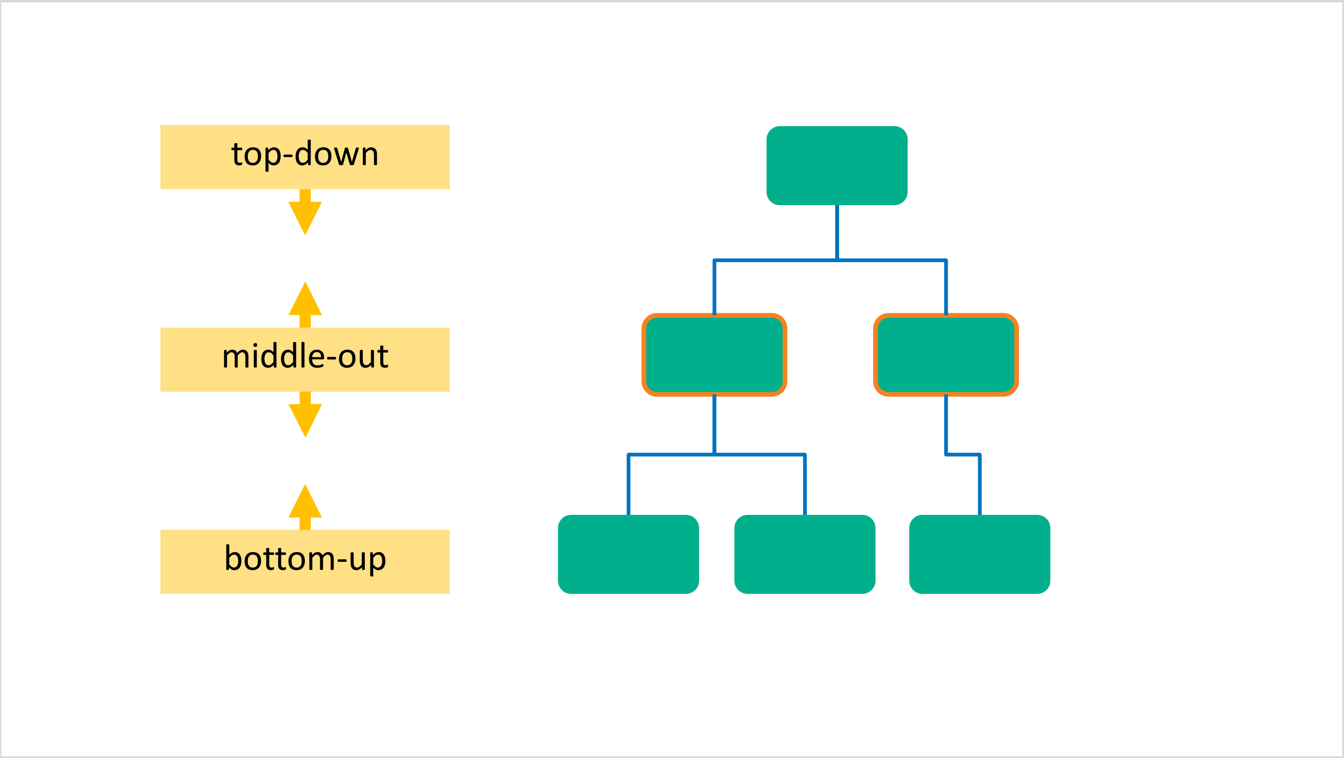
Data accumulated to the bottom level of the hierarchy are usually noisy and might be sparse. For example, consider store- or SKU-level data.



Data aggregation methods are smoothing processes, and sparseness is usually less of a problem at higher levels of the hierarchy. Patterns such as seasonality and trend are usually more easily discernible in data aggregated to the distribution center level, for example.

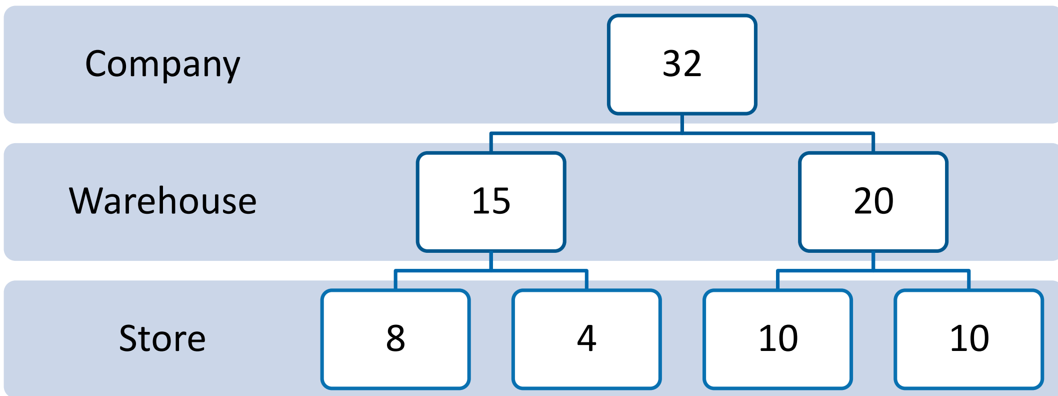


However, as data continue to be aggregated up the hierarchy, they can become overly smoothed. Interesting and predictive signals can be washed out.



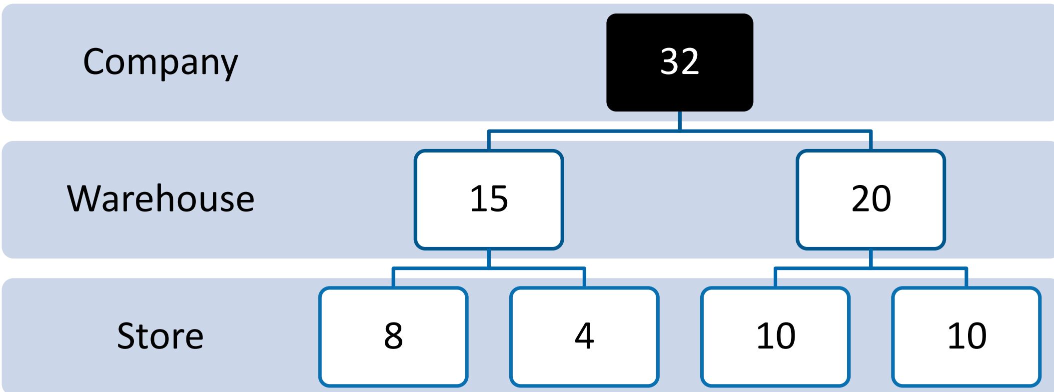
This leads to a rule of thumb for choosing a strategy for reconciliation. The best level to reconcile to is usually somewhere in the middle of the hierarchy. This level is where the models get the best look at the available signal in the data.

Middle-Out Reconciliation Using Forecast Proportions

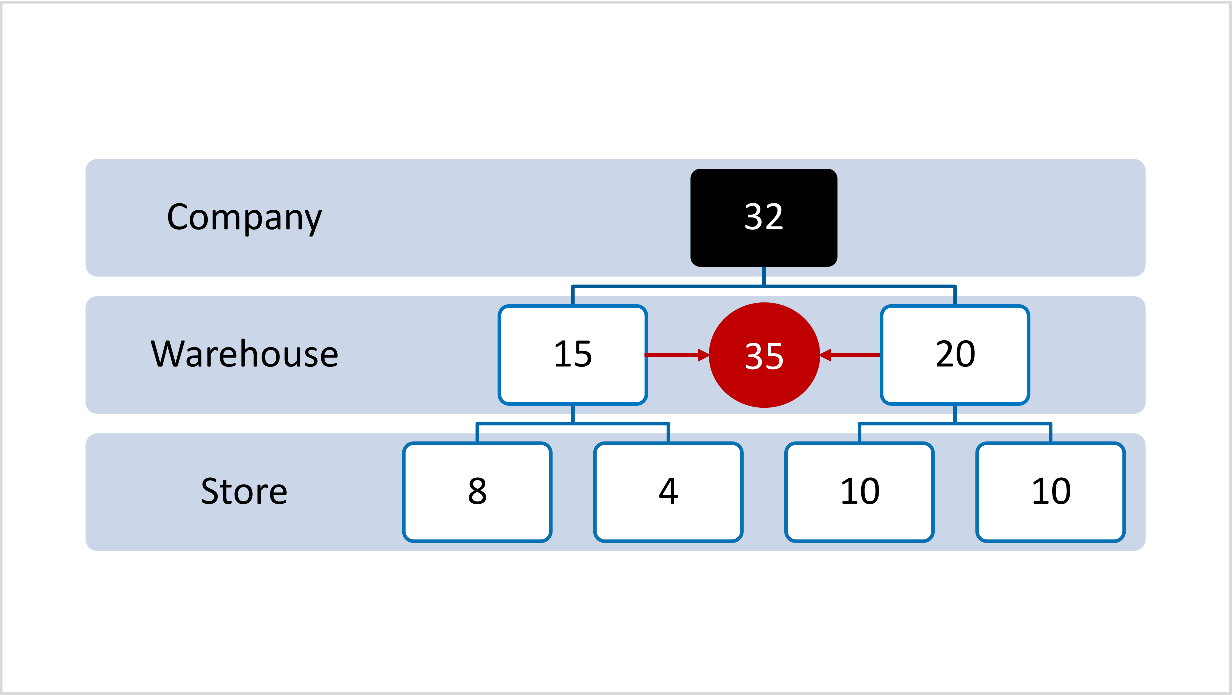


Let's look at a simple illustration of a middle-out reconciliation strategy using Forecast Proportions, the default methodology for distributing reconciliation effects.

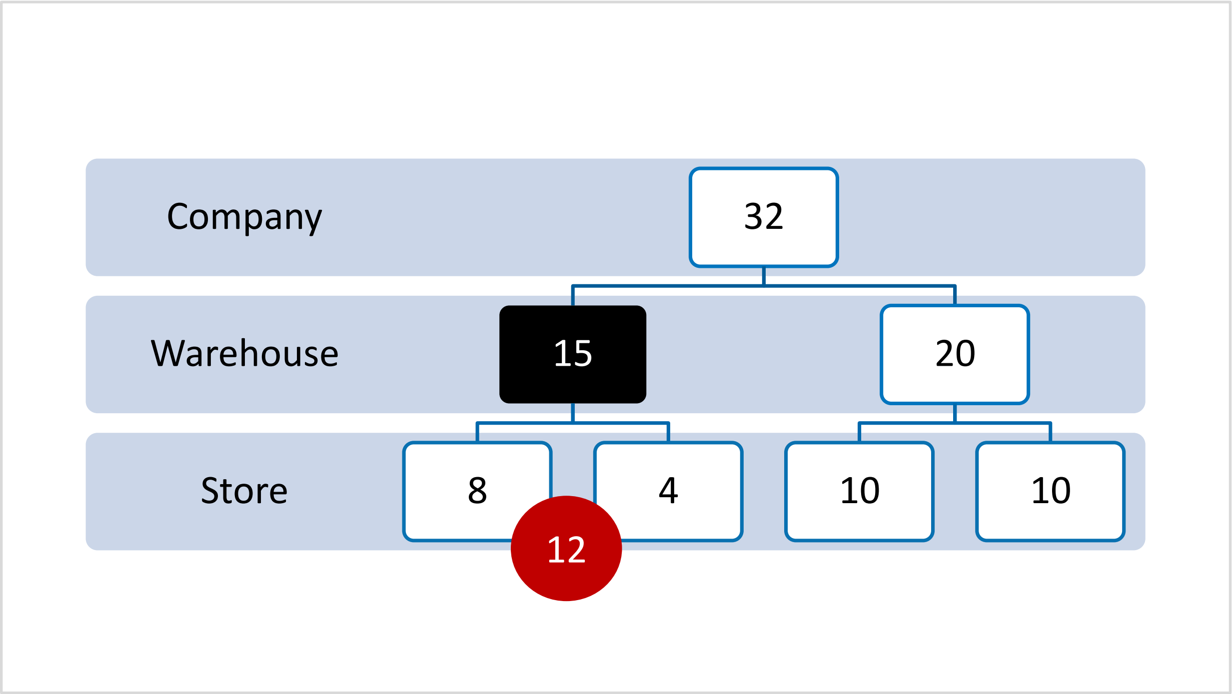
Statistical forecasts for January 2019 are shown in boxes on Company, Warehouse, and Store rows.



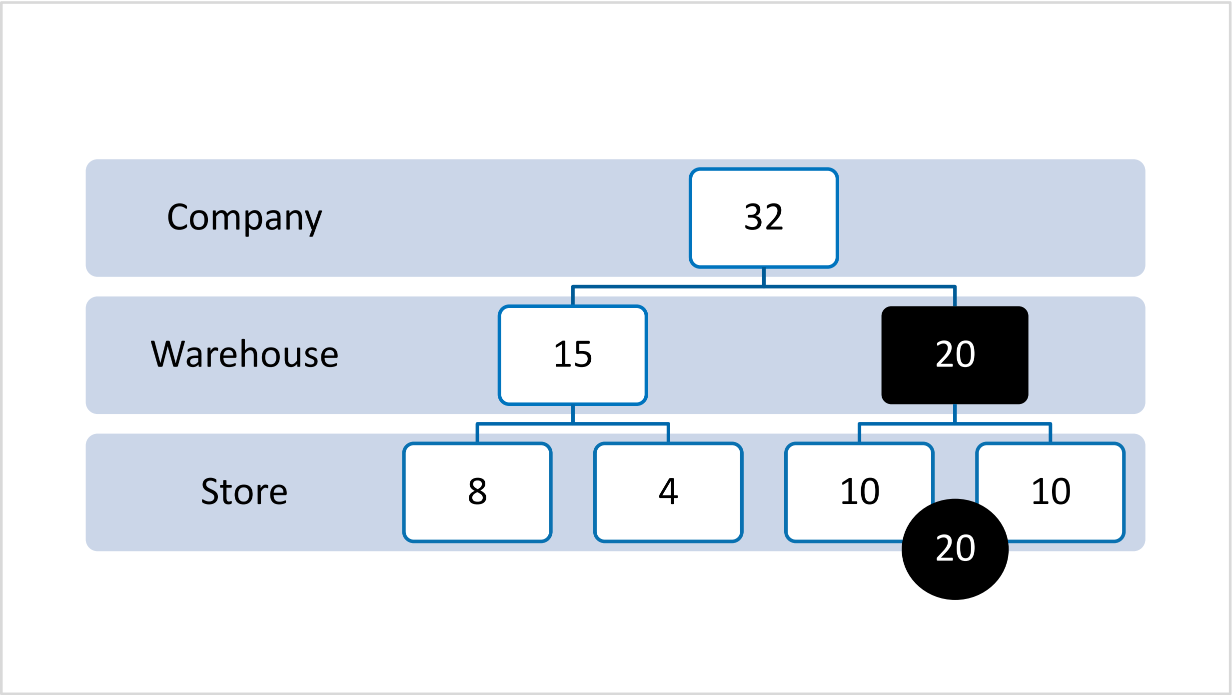
The company forecast is 32 units.



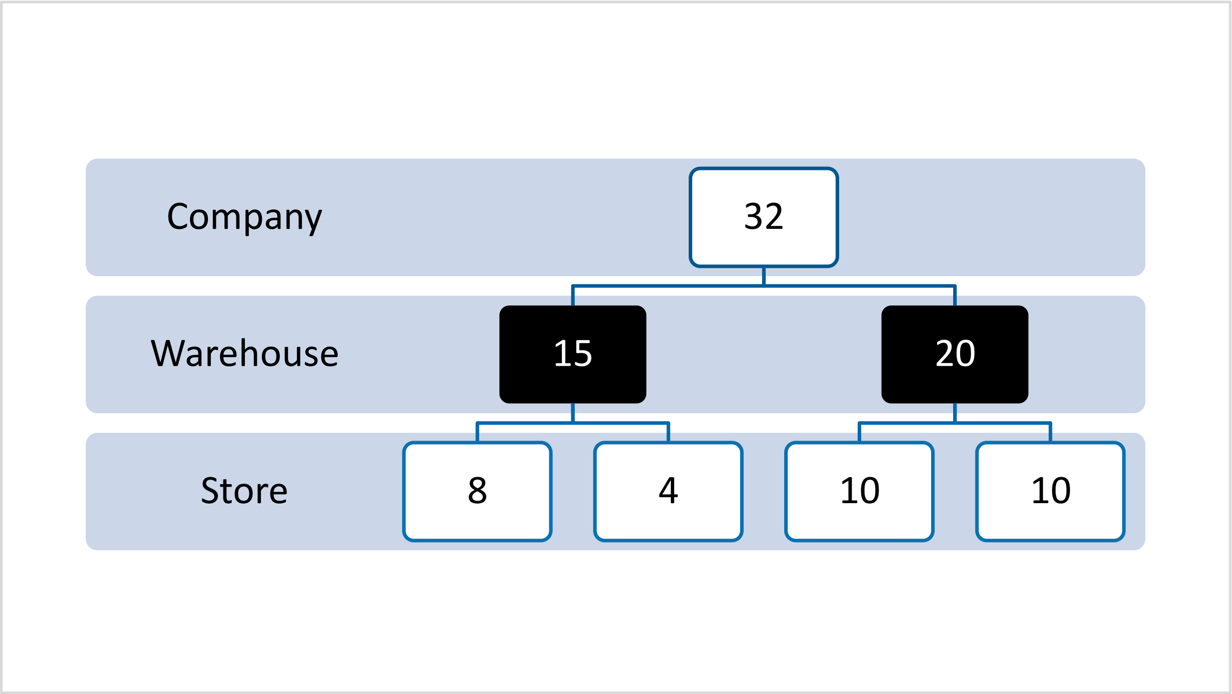
However, the two warehouses owned by the company have forecasts of 15 units and 20 units. The sum of the warehouse forecasts is 35 units, which is inconsistent with the company forecast of 32 units. This is possible because the company series was modeled independently from the warehouse series models.



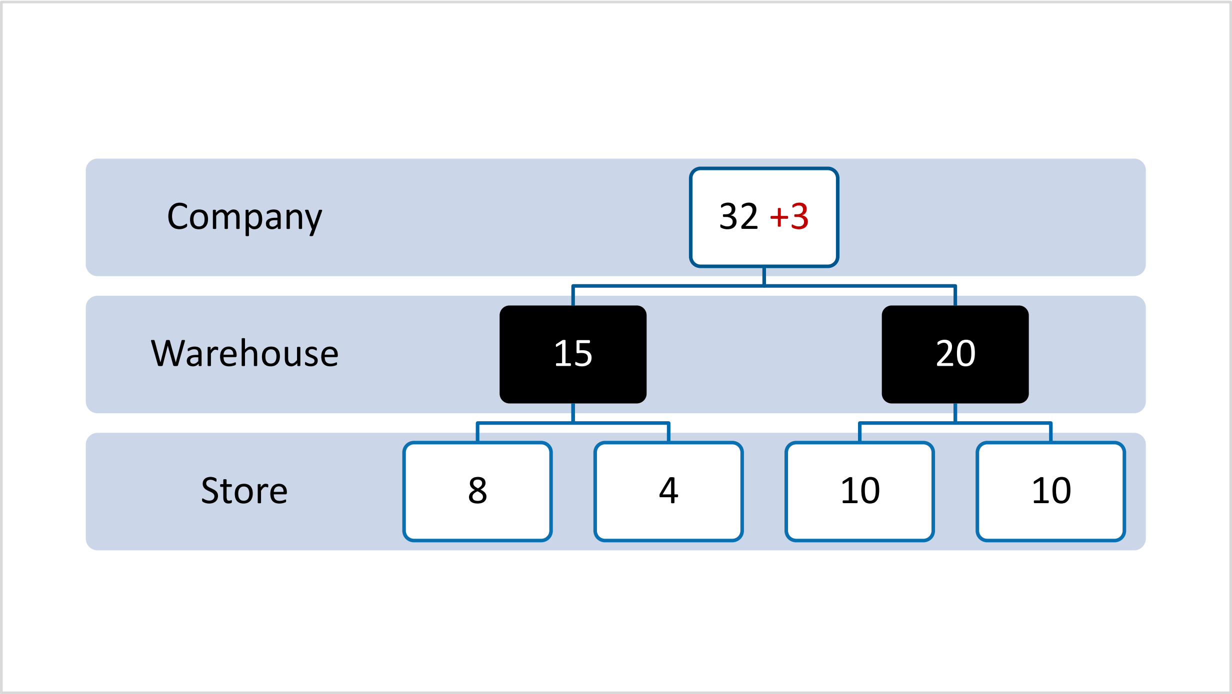
Similarly, the sum of the two store forecasts for the first warehouse is 12 units, whereas the forecast for the warehouse itself is 15 units.



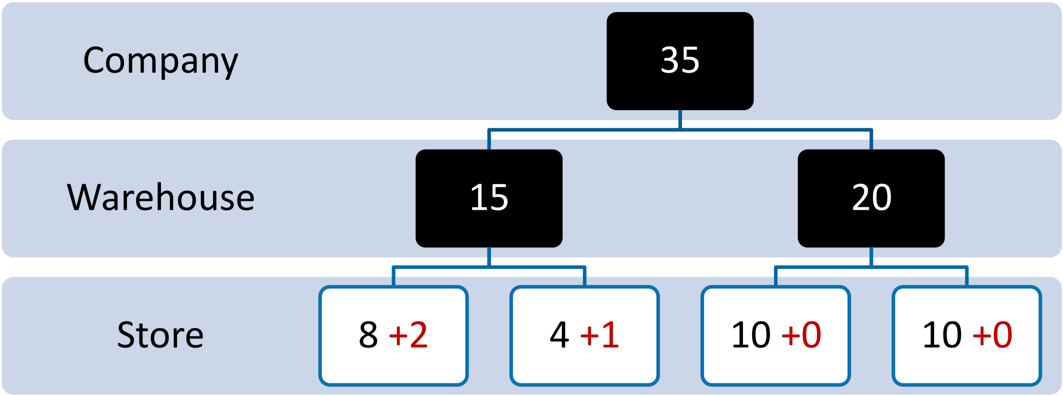
The sum of the store forecasts for the second warehouse is already consistent with the second warehouse forecast of 20 units.



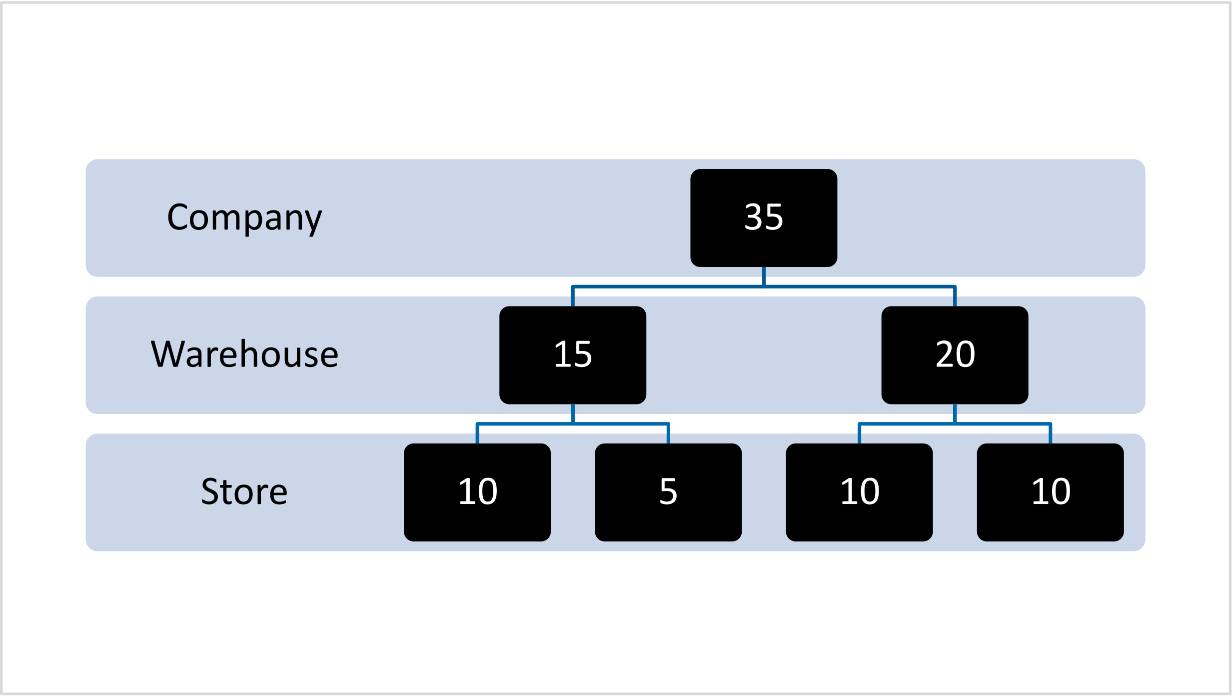
The company decides that the warehouse forecasts are the basis for reconciliation. Because warehouse is at the middle level of the hierarchy, we perform middle-out reconciliation. The warehouse forecast values, 15 and 20, are set as the standards for reconciliation.



The discrepancy between the top-level company forecast, 32, and the sum of the middle-level warehouse forecasts, 35, is three units. The company-level forecast is increased by three to make the top and middle levels consistent. To reconcile up a level of a hierarchy, find the difference between the higher level and the sum of the lower level of the hierarchy, and then add that value to the higher level.

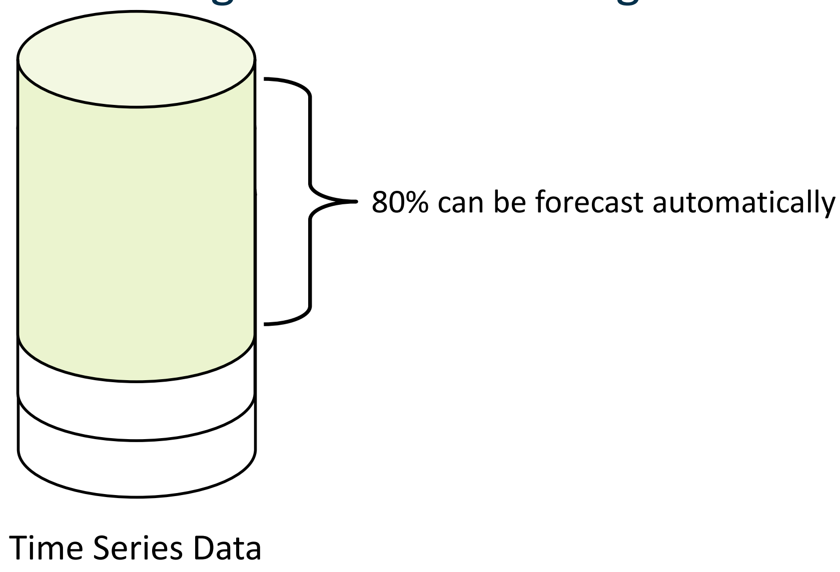


In order to reconcile the values down a level of a hierarchy, you must decide how to apportion the reconciliation between series at the lower level of the hierarchy. In this case, the difference between the first warehouse forecast and its two stores forecasts is three units. The company decides to use proportional apportioning of the three units. The first store accounts for 8/12, or 2/3, of the sum of the store-level forecasts. We apportion two-thirds of the three reconciliation units to the first store. The second store is apportioned the remaining unit.

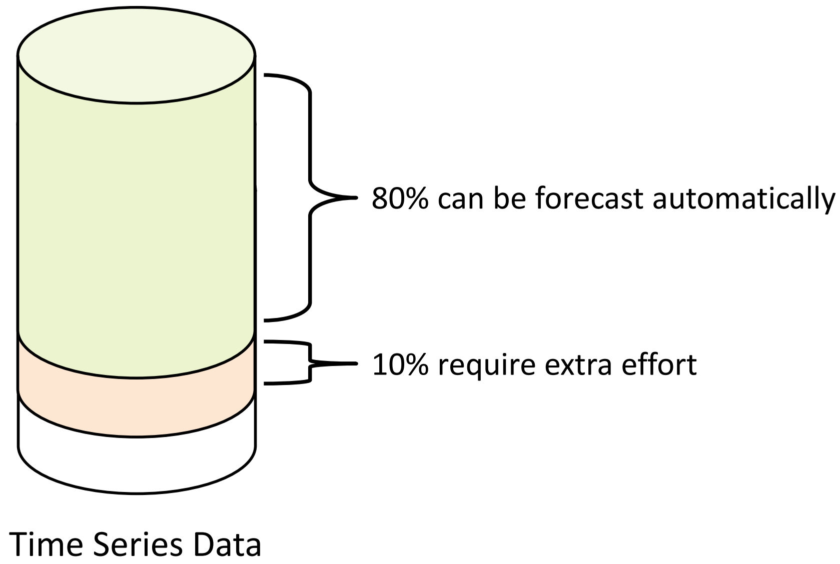


The reconciled forecasts are now consistent up and down the hierarchy.

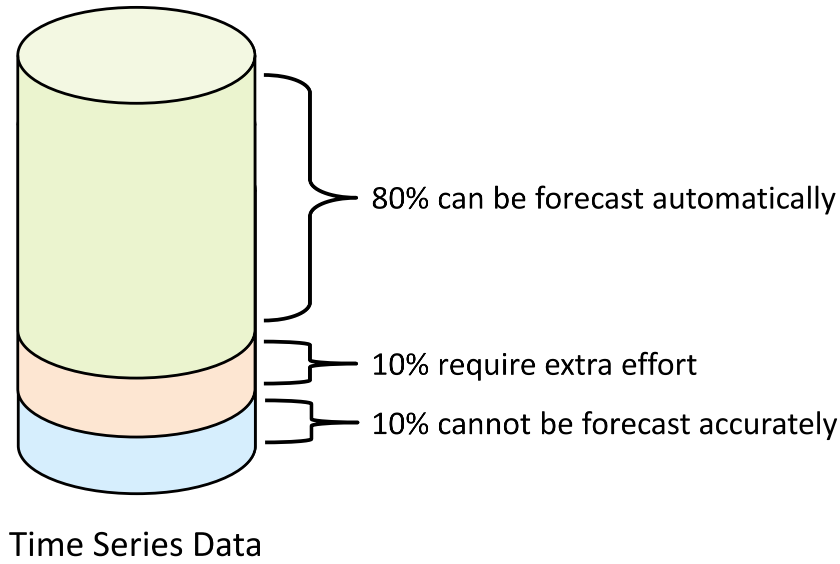
Large-Scale Forecasting



Automatic forecasting algorithms can satisfactorily handle the majority of forecasts for time series. That is, for a majority of the time series, the analyst does not need to visually scrutinize the data, identify an appropriate model, estimate parameters, or generate forecasts. Algorithms can handle these jobs in an automated way.



For another small percentage of the time series, the automatic forecasting algorithms are inadequate. The analyst needs to provide intervention and further refinement to produce forecasts with acceptable accuracy.



A final small percentage of the data consists of time series for which accurate forecasts are not possible. These series have a very low signal-to-noise ratio (for example, short series and random walks).

The process described is forecasting by exception. The majority of the time series to be forecast are handled in an automated way, leaving analysts to focus their time and expertise on problematic or high-value series.

Demo: Generating Hierarchical Forecasts with the Default Functionality

This demonstration illustrates the default functionality of the Hierarchical Forecasting node.

Idea Exchange

- How are the data that you are responsible for forecasting arranged or rolled up?
- At what level of aggregation do you get the best “look” at any signal the data might have? For example, if there is a seasonal cycle, is it most evident at a total sales or company level?
- If you perform reconciliation, what level of the hierarchy do you reconcile forecasts to?

Questions?

Practice

Hierarchical Forecasting

Selecting a Champion Model

How to select a champion model

Lesson 03, Section 02

HTML Placeholder

Combining the Forecasts from Multiple Models

Often, combining the forecasts from several models can result in superior forecasts when compared to a single forecasting model. Using SAS Visual Forecasting, you can create a combined model and add it to the model selection list for the current series. Options in SAS Visual Forecasting enable you to control different aspects of the forecast combination process for the candidate models. Given the response time series with previously generated forecasts from several models, a combined forecast can be created from these component forecasts by model combination using appropriate weights.

Understanding the Process for Combining Forecasts

SAS Visual Forecasting uses the following process to create a combined model:

1. **Forecast candidate tests** - By using tests for seasonality and intermittency, this step removes any models from the set that are not suitable for the targeted forecasting variable. For example, the target series is seasonal, and some models are not seasonal. These non-seasonal models would be removed from the list of candidates.
2. **Forecast missing value tests** - This step removes any models that have too many missing values. You can specify thresholds for the percentage of missing values in the horizon or in the estimated region.
3. **Encompassing tests** - This step removes any models that contain redundant information. This test is performed in a pairwise iteration over the ranked set of candidates that remain after the previous step. Performing an encompassing test is optional.
4. **Weight estimation** - This step assigns weights to the forecasts that were generated after running any encompassing tests. The weights are assigned based on the information in the combined model list.
5. **Forecast combination** - This step results in the weighted combination of the forecasts.
6. **Weighted average** - This step calculates the forecasts using a weighted average of the predictions from the models in the table. Models with missing values are not included in the forecast combination. Instead, these missing values are either dynamically rescaled using the weights of the nonmissing forecasts or generated from the combined forecast. The results of this step are the combined forecast, its residual, standard error, and the upper and lower confidence intervals.

Generating a Combined Model

About Automatically Generating a Combined Model

Using SAS Visual Forecasting, you can automatically generate a combined model for the set of time series models that is generated from the diagnosis of each series. The options for combined models are available in the Hierarchical Forecasting node. These options correspond to the options for the COMBINE statement in the HPFDIAGNOSE procedure.

Demo: Adding Combined Models to the Hierarchical Forecasting Pipeline

This demonstration illustrates expanding the functionality of the Hierarchical Forecasting node.

Questions?

Demo: Selecting Models Based on Forecast Accuracy

This demonstration illustrates incorporating best practices into the automatic model selection process.

Questions?

Demo: Sharing a Custom Pipeline via the Exchange

This demonstration illustrates how a custom pipeline can be shared using the Exchange.

Questions?

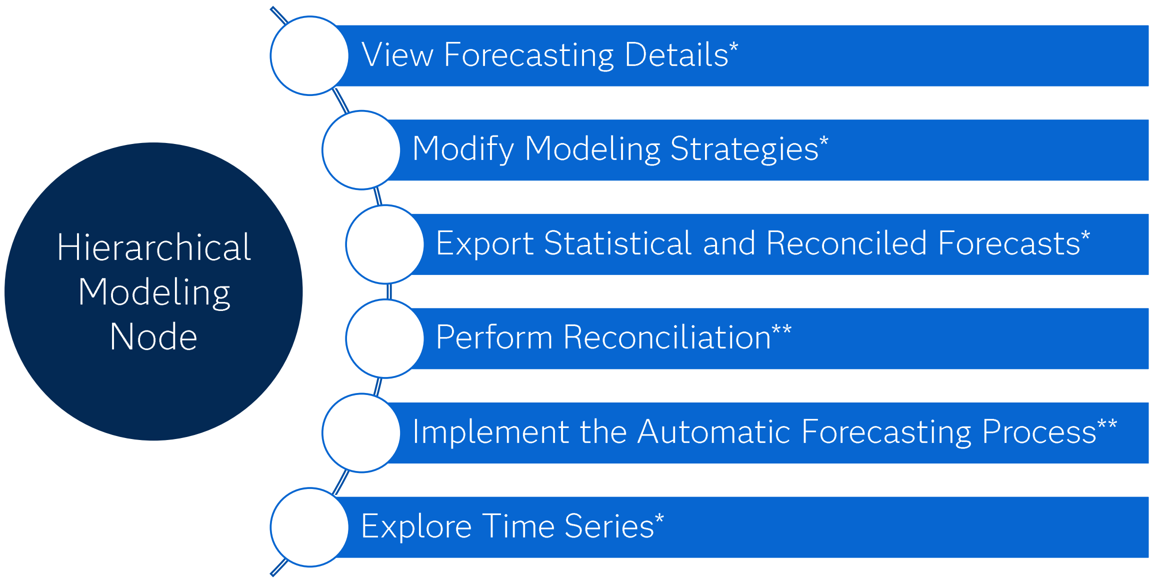
Practice

Using a Custom Pipeline

Hierarchical Modeling Node

How to use the Hierarchical Modeling Node

Lesson 03, Section 03



*For Any Level of a Project's Hierarchy
** Across all levels of a Project's Hierarchy

The Hierarchical Modeling node provides functionality to explore and modify the automatic forecasting processes at all levels of the data hierarchy in a project. The Hierarchical Modeling node creates a separate pipeline for each BY group or level in the in the data, and analysts can interact with pipelines to assess forecast results, modify modeling strategies and export generated tables corresponding to individual BY groups.

Demo: Implementing the Hierarchical Modeling Node and Generating Reconciled Forecasts for the Project

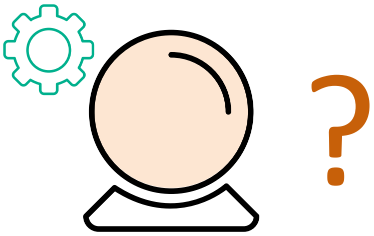
This demonstration illustrates the functionality of the Hierarchical Forecasting node.

Overrides and Exporting Generated Tables

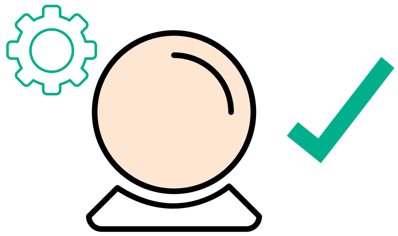
How to create overrides and export generated tables from a time series

Lesson 04, Section 01

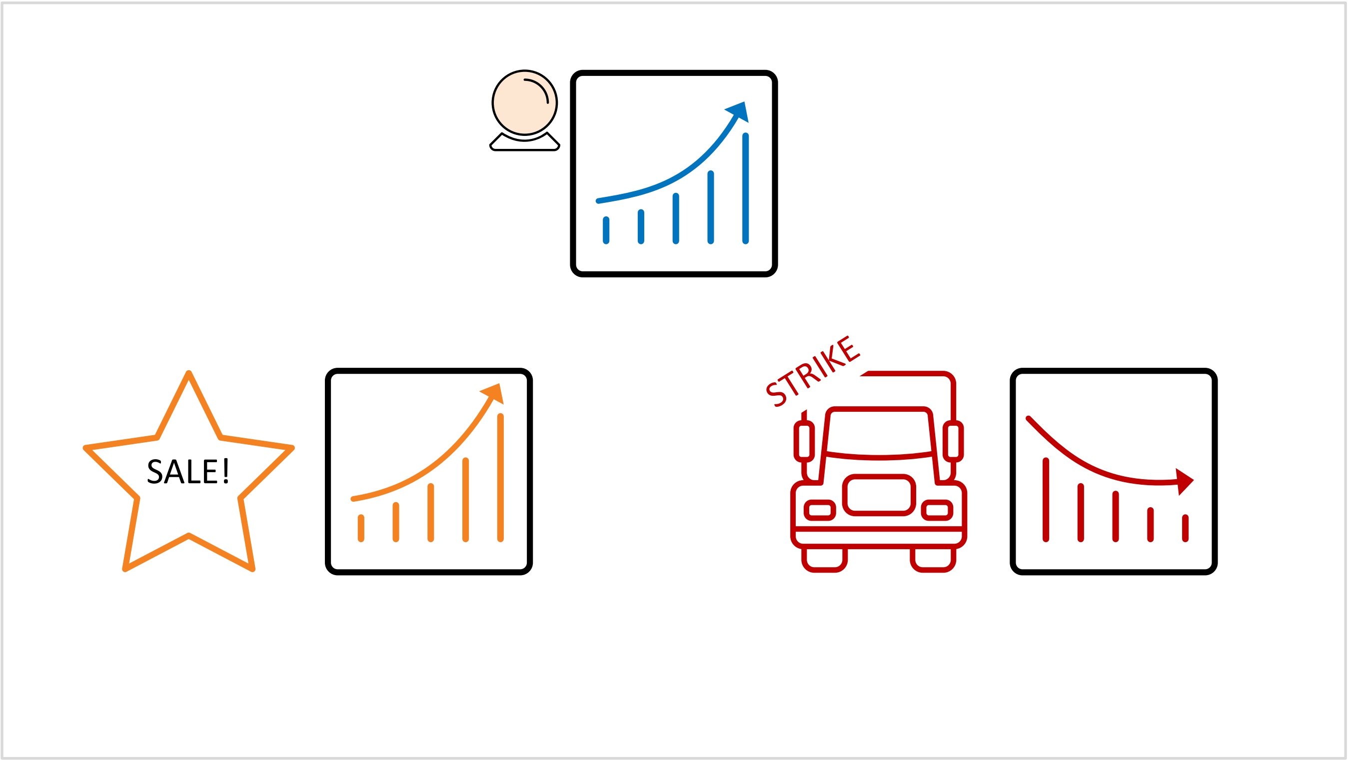
Best Practices for Forecast Adjustment



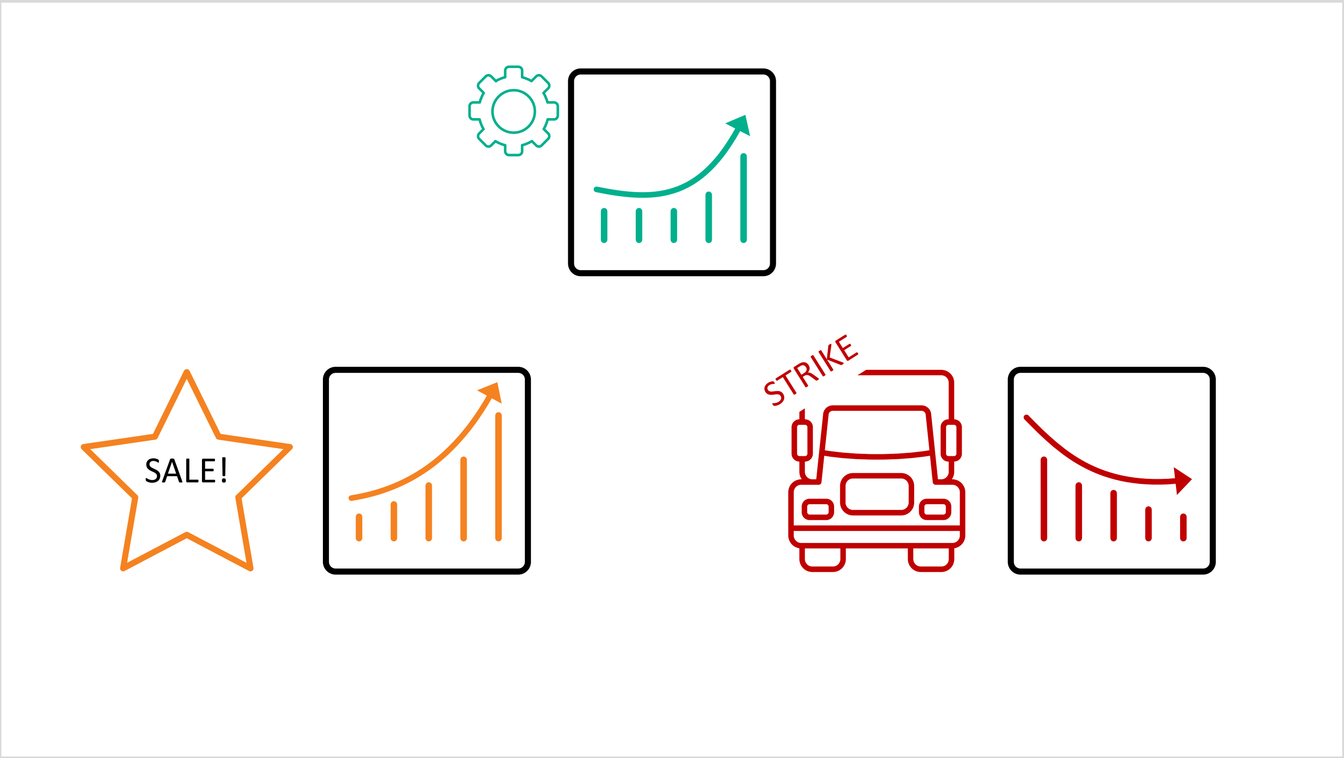
Is it a good idea to adjust statistical forecasts?



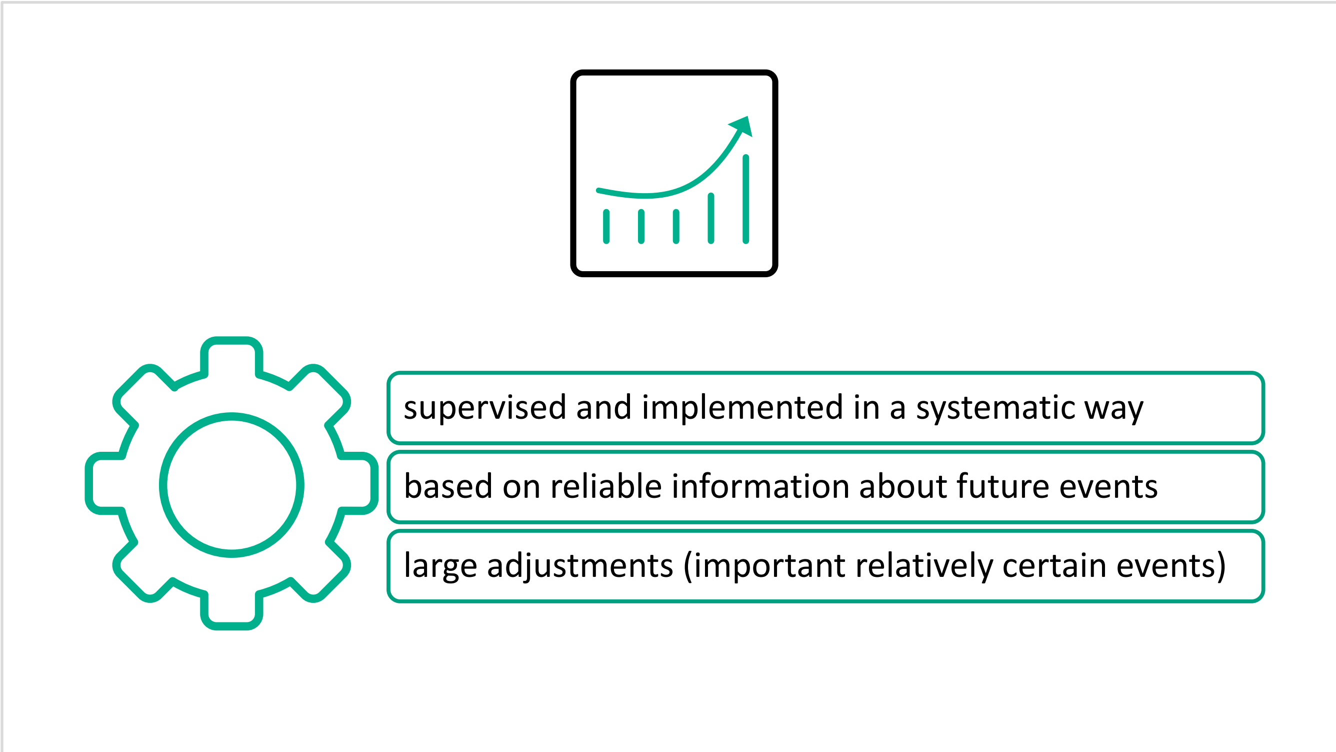
The answer is "Yes." We want our statistical forecast to be consistent with our business knowledge of the data.



For example, consider a forecast for sales data. We might have an upcoming promotion that we want to implement. Also, we might know that our truck delivery drivers are going on strike.



The future forecast needs to be modified to reflect these two events that the models do not know about. Adjusting forecasts to accommodate the hypothesized effects of these events can improve forecast accuracy.



Statistical forecast adjustments should be supervised and implemented in a systematic way. Adjustments should be based on reliable information about future events. It is best practice to perform large adjustments because they are usually motivated by knowledge of important and relatively certain events, such as natural disasters, pending law changes, or upcoming competitor activity.

Be wary of making smaller adjustments that can be motivated by factors not associated with the process generating the data. These include forcing something to seem to contribute to the process or forcing statistical forecasts to align with business goals. Smaller changes tend to degrade forecast accuracy.

Demo: Adding the Attributes Table to a Project

This demonstration illustrates how to add additional attributes via an Attributes table.

Questions?

Demo: Applying Overrides to Generated Forecasts

This demonstration illustrates the use of the overrides functionality in a forecasting project.

Questions?

Demo: Resolving Override Conflicts

This demonstration illustrates how override conflicts arise and how they can be resolved.

Questions?

Demo: Exporting Forecasts

This demonstration illustrates how to make forecasts available to team members and other stakeholders.

Questions?

Practice

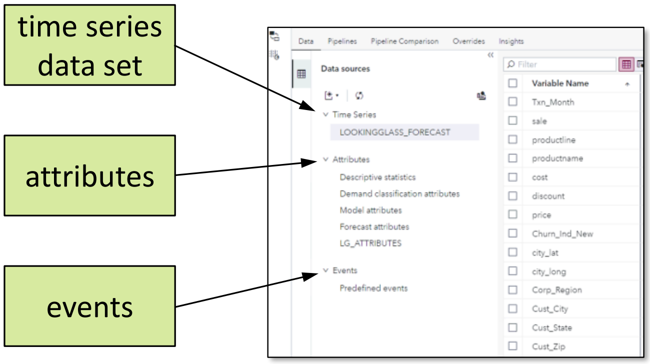
Using the Filter Option to Create a Histogram

New and Existing Attributes and Filters

How to utilize new and existing attributes and filters in a time series

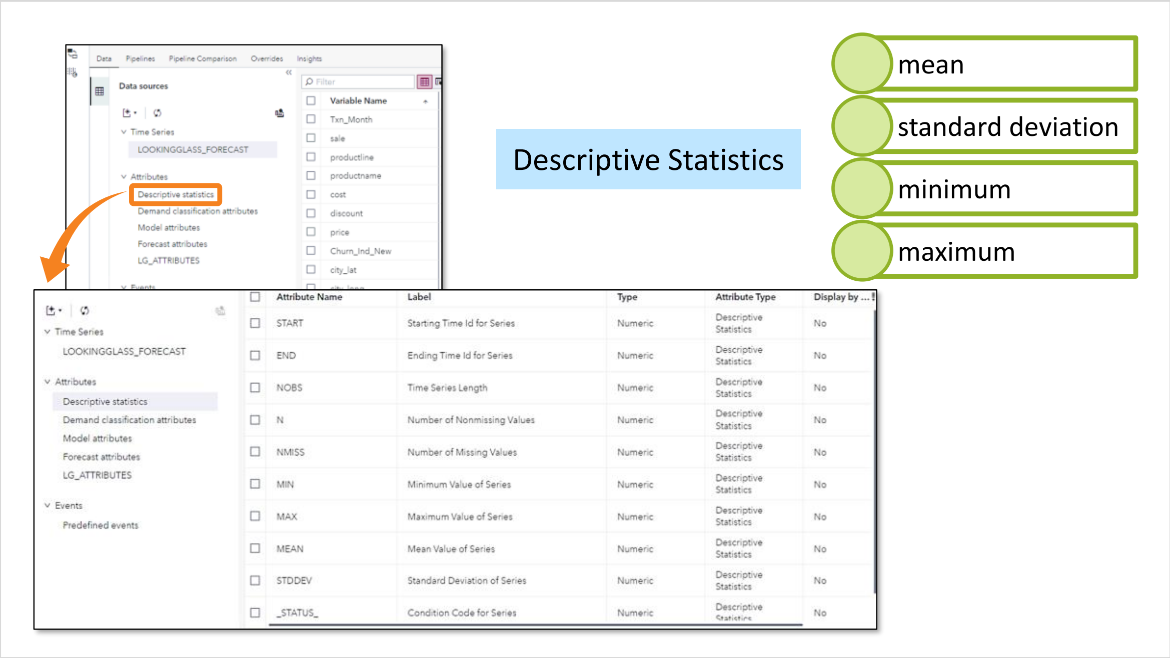
Lesson 05, Section 01

Additional Filters and Attributes

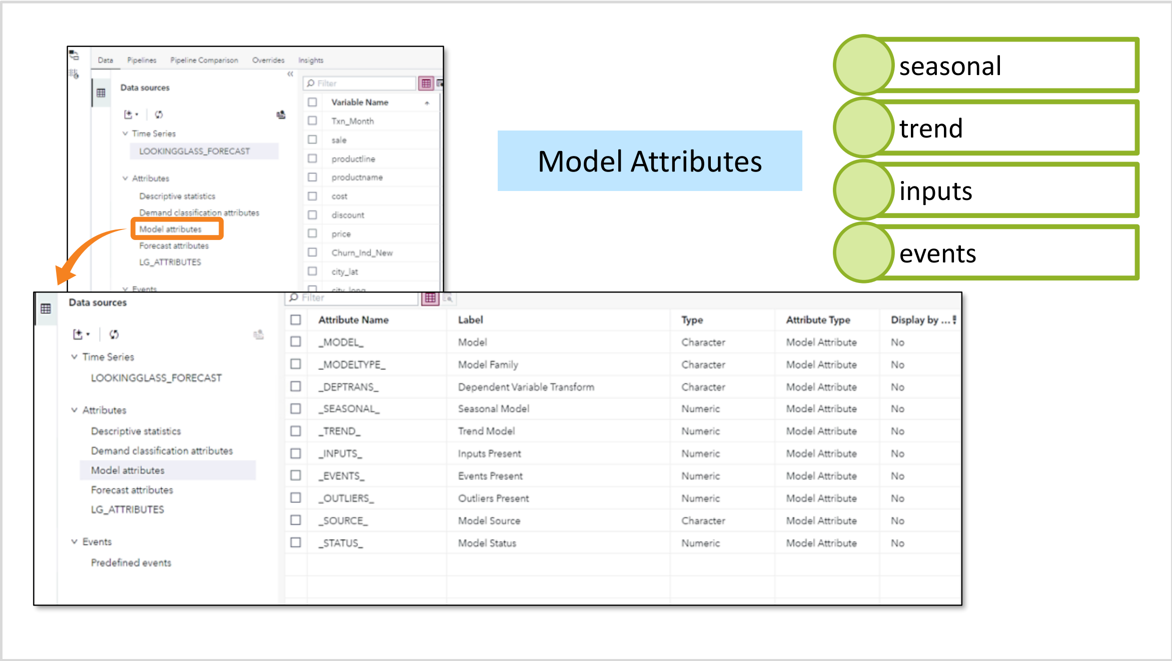


Previously, we discussed the usage of primary attribute variables from our time series data set and secondary attribute variables from an external attributes data set. We used them within the time series viewer and in overrides.

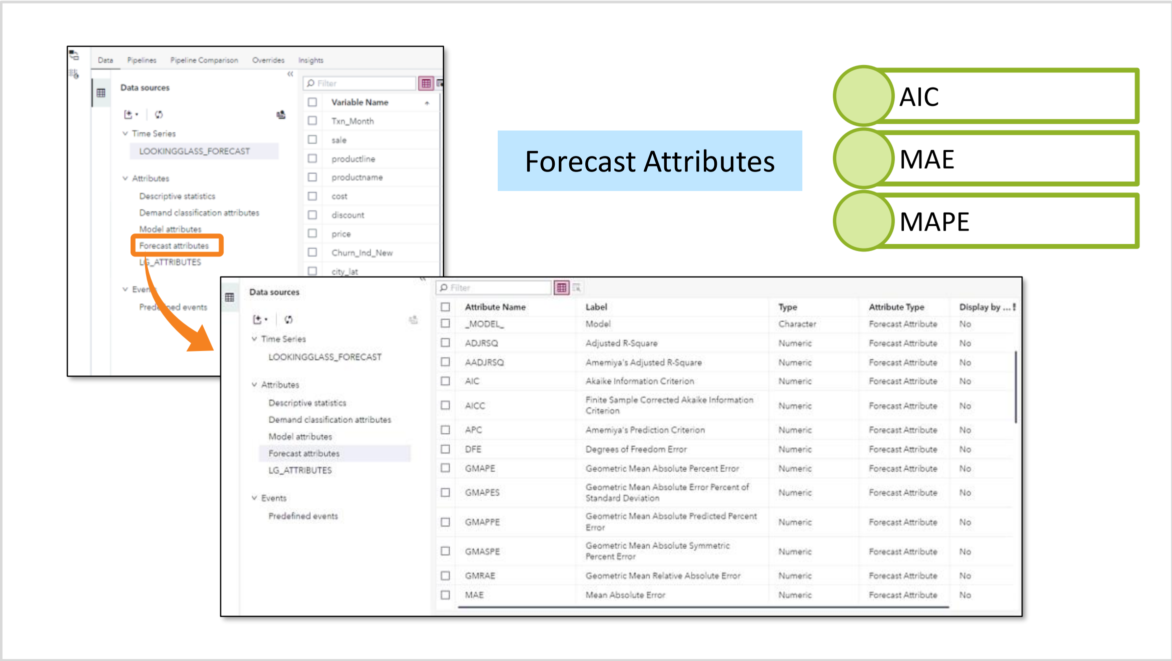
Let's expand our use of filters with additional attributes and events!



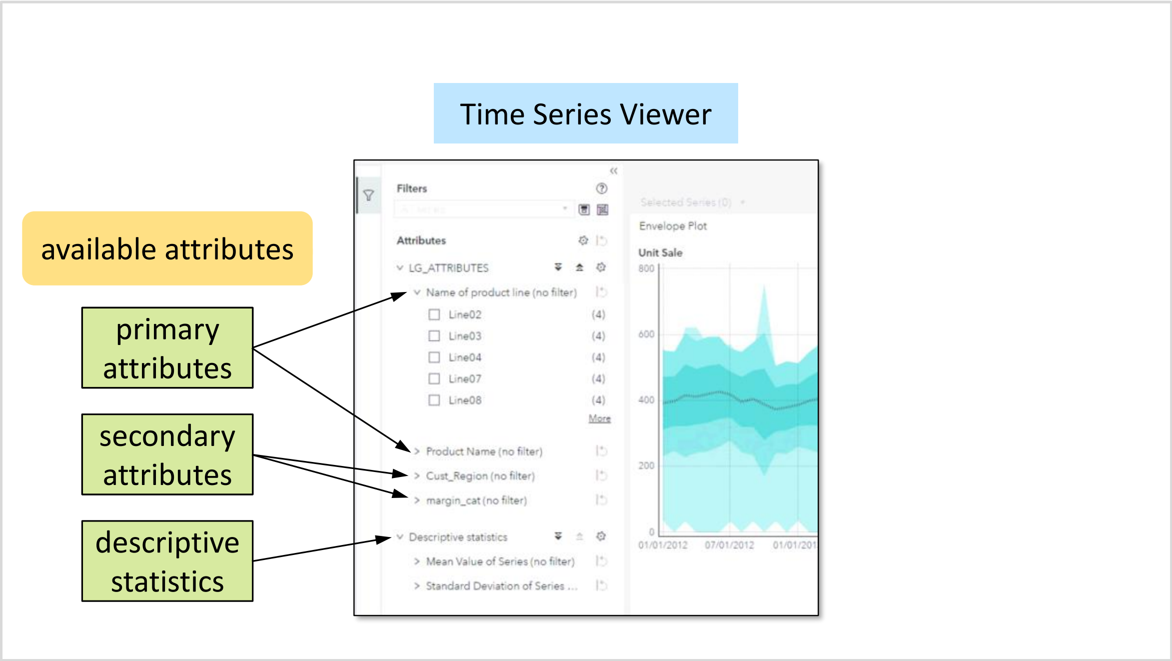
One type of additional attribute that we can use is descriptive statistics. Descriptive statistics are calculated from the time series data set directly. Examples of descriptive statistic attributes include mean, standard deviation, minimum, and maximum. Do you want to filter your exploration on time series with particular values for their mean? What about their maximum value? If so, then these attributes are for you.



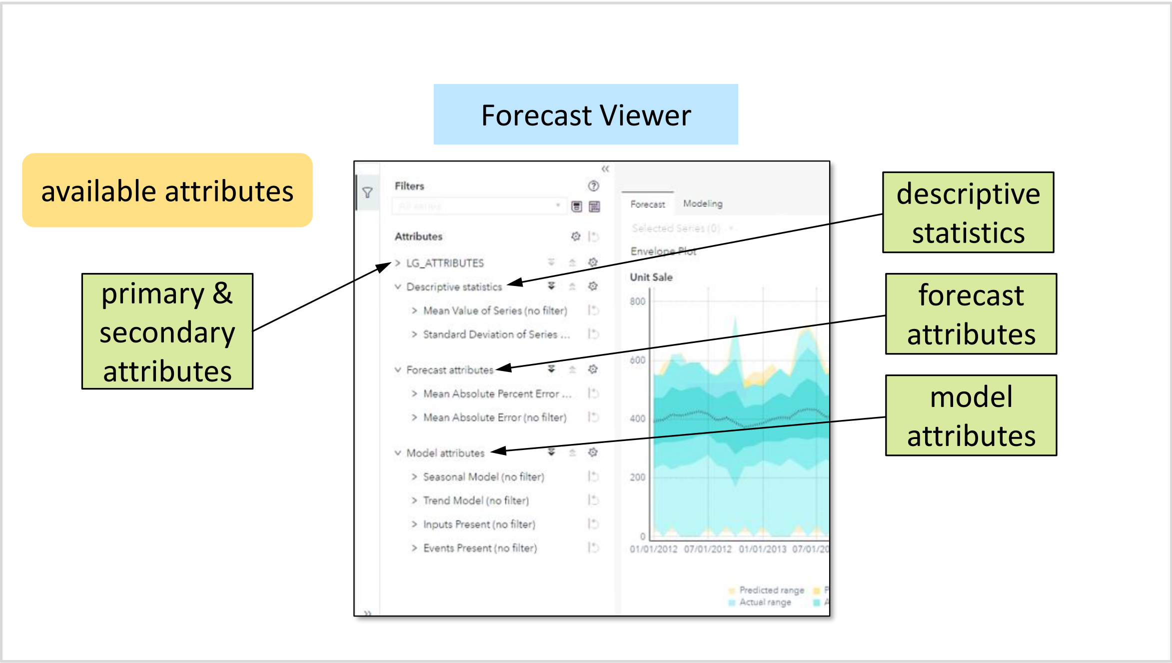
Another type of attribute that we can use is model attributes. Model attributes are calculated from the time series analysis and are not available until after an analysis node in a pipeline is executed. Examples of model attribute include seasonal, trend, inputs, and events. Do you want to filter your exploration on time series with seasonal components in their selected models? What about usage of inputs? If so, then these attributes are for you.



Forecast attributes are another type of attribute that we can use, and they are calculated from the time series analysis, similar to model attributes. However, they are not available until after an analysis node in a pipeline is executed. Examples of forecast attributes include AIC, MAE, and MAPE. Would you want to filter your exploration on time series within a particular interval for their MAPE? What about within a particular interval of AIC? If so, then these attributes are for you.



The time series viewer is available after processing the Data node of your pipeline. During your exploration using the time series viewer, you can use primary attributes, secondary attributes, and descriptive statistic attributes. These are the attributes that are drawn from the time series data set directly. No other attributes can be used as a filter on the time series viewer.



After the execution of a Time Series Analysis node in your pipeline, you will be able access the Forecast Viewer. This viewer is very similar to the Time Series Viewer. However, the attributes that can be used as filter for exploration are increased. The Forecast Viewer is able to use primary and secondary attributes as well as descriptive statistic attributes (just like in the time series viewer). Because a forecast model has been generated, we can now also include the forecast attributes and the model attributes to this Forecast Viewer.

Demo: Incorporating More Filters into the Time Series Viewer

This demonstration illustrates creating more filters, beyond the primary and secondary attribute variables, for use in the Time Series Viewer.

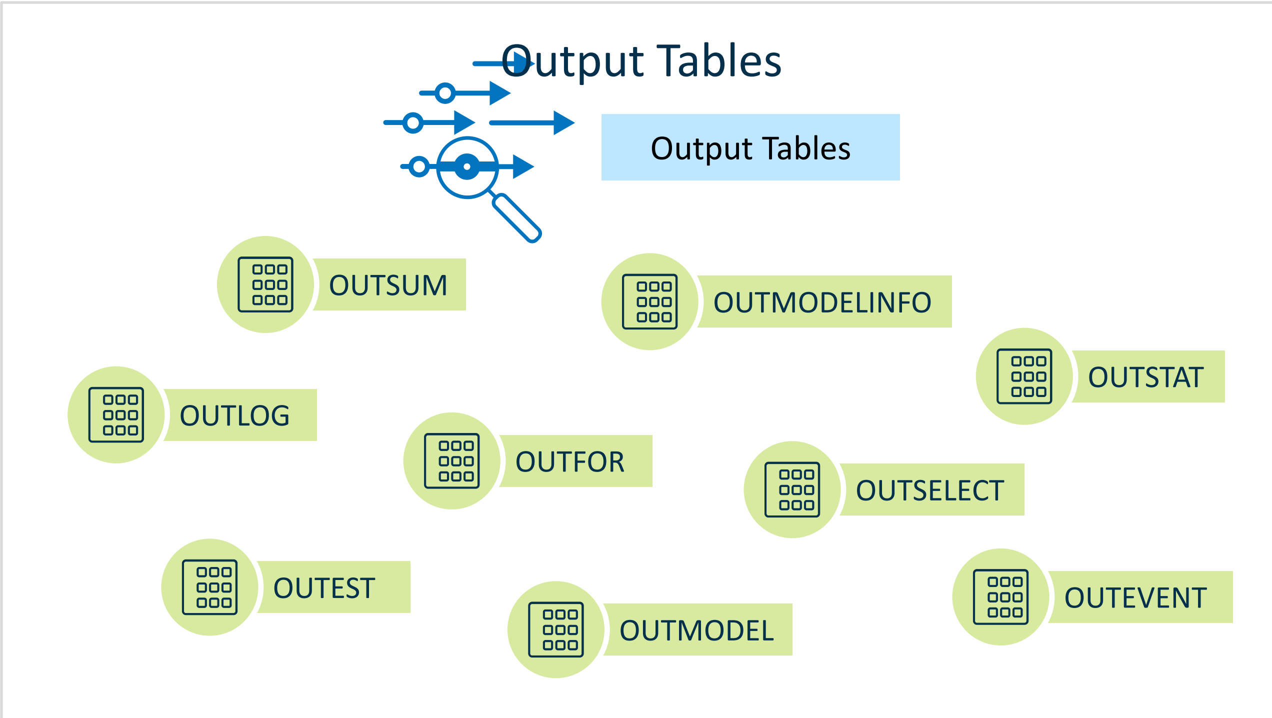
Demo: Using Filters in the Forecast Viewer

This demonstration illustrates using more filters, beyond the primary and secondary attribute variables, within the Forecast Viewer.

Data Export

How to export data from a time series analysis

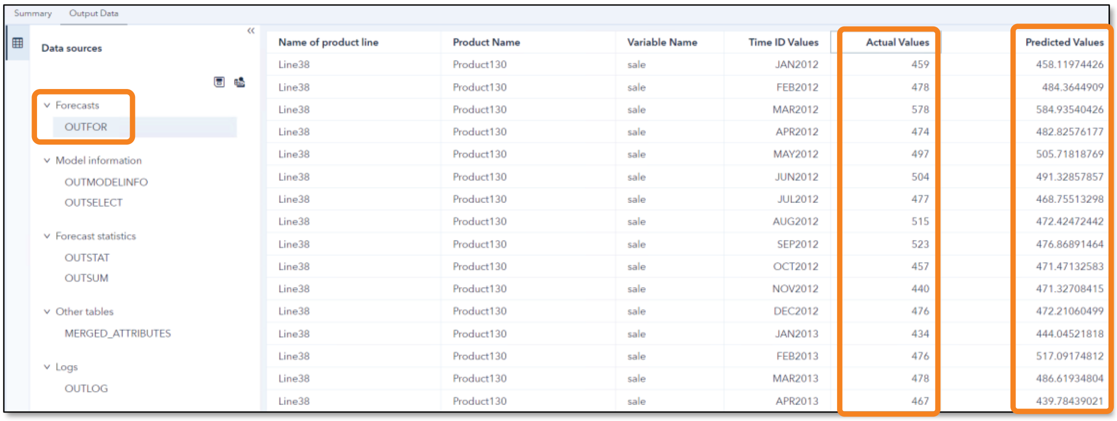
Lesson 05, Section 02



This section provides an overview of the output tables that are available in a Forecasting project and discusses details about the different ways that they can be disseminated outside of the Model Studio project.

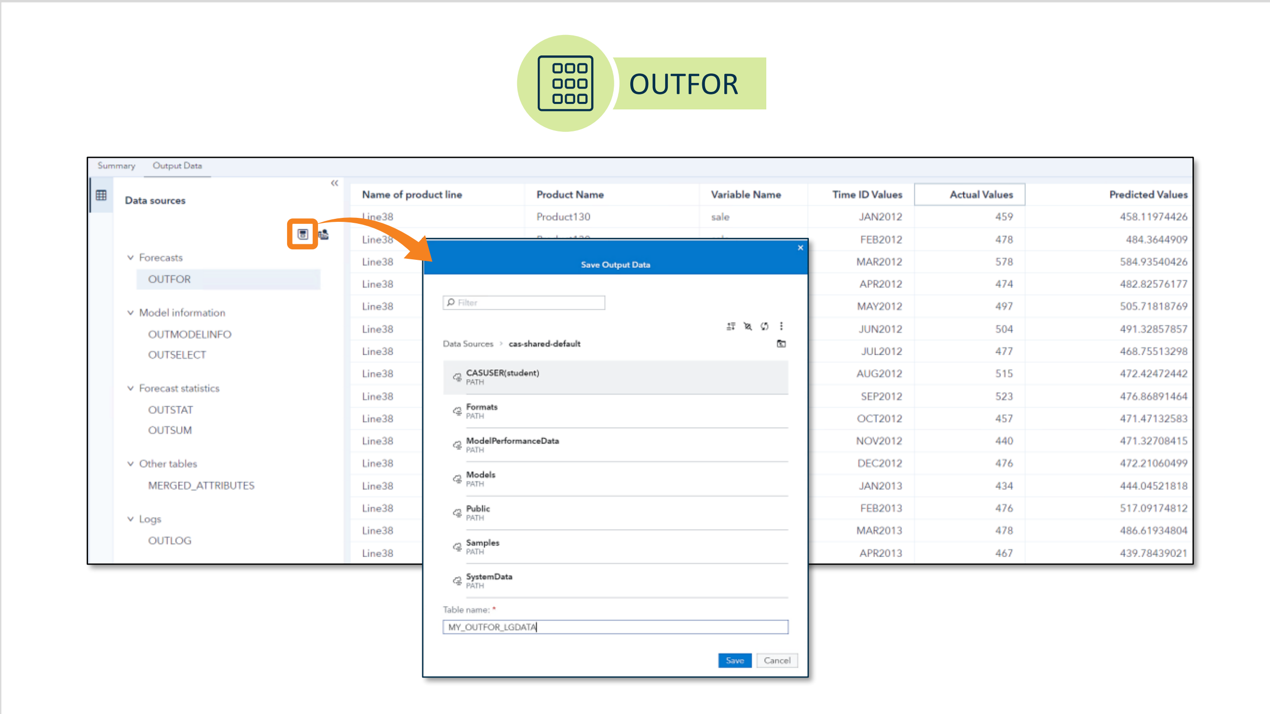
After a champion pipeline is selected and overrides are implemented, a next step is to disseminate the project's results to forecast consumers. Forecasting results are available in several output tables. Some generated tables can be exported or made available outside of Model Studio with just a couple of mouse clicks. Others need to be specifically requested prior to running a project or pipeline. We'll also introduce the Save Data node in this section. This node adds flexibility in the process of exporting a project's data sets.

OUTFOR



Name of product line	Product Name	Variable Name	Time ID Values	Actual Values	Predicted Values
Line38	Product130	sale	JAN2012	459	458.11974426
Line38	Product130	sale	FEB2012	478	484.3644909
Line38	Product130	sale	MAR2012	578	584.93540426
Line38	Product130	sale	APR2012	474	482.82576177
Line38	Product130	sale	MAY2012	497	505.71818769
Line38	Product130	sale	JUN2012	504	491.32857857
Line38	Product130	sale	JUL2012	477	468.75513298
Line38	Product130	sale	AUG2012	515	472.42472442
Line38	Product130	sale	SEP2012	523	476.86891464
Line38	Product130	sale	OCT2012	457	471.47132583
Line38	Product130	sale	NOV2012	440	471.32708415
Line38	Product130	sale	DEC2012	476	472.21060499
Line38	Product130	sale	JAN2013	434	444.04521818
Line38	Product130	sale	FEB2013	476	517.09174812
Line38	Product130	sale	MAR2013	478	486.61934804
Line38	Product130	sale	APR2013	467	439.78439021

The OUTFOR table is probably the most widely accessed of the system-generated tables. It contains the forecasts, or predicted values, actual values, and confidence limits for each BY group in the data. This table can be viewed in the Results of Forecasting Modeling nodes by selecting the Output Data tab.



The OUTFOR table can be exported by selecting the Save Table button and then choosing an existing CASLIB to store it in.

OUTSTAT

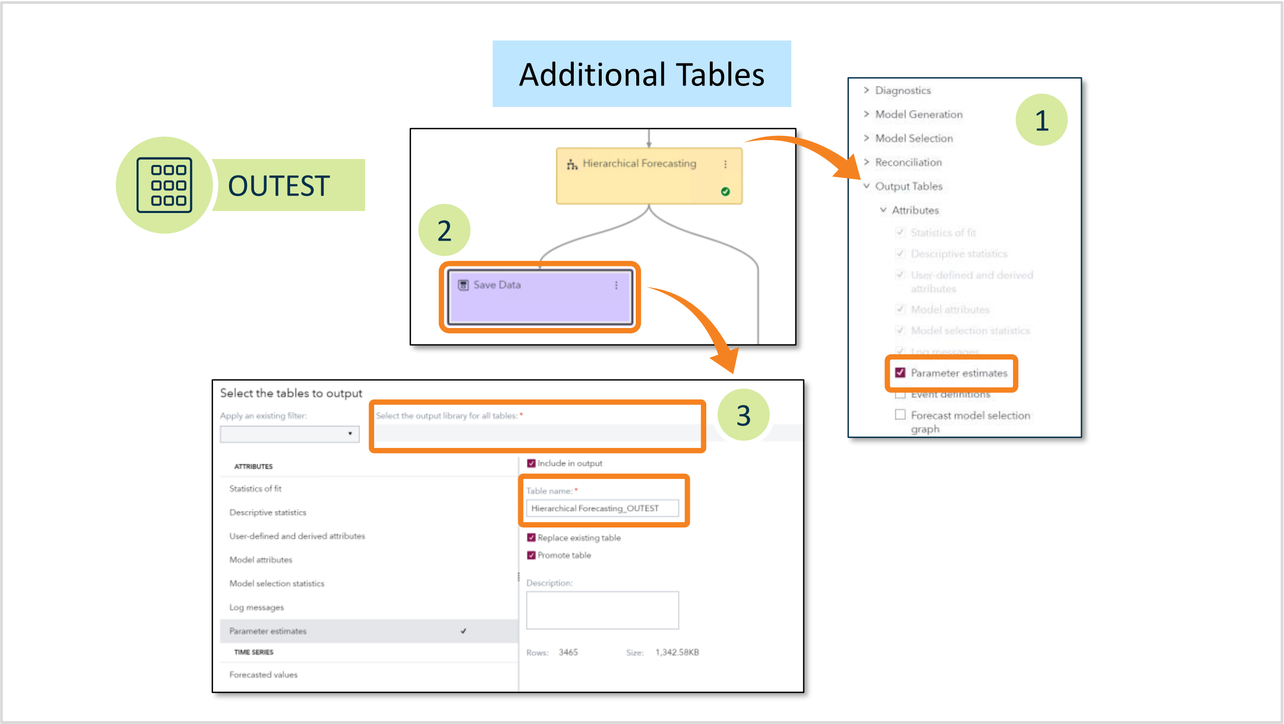
Summary Output Data

Data sources

- Forecasts
  - OUTFOR
- Model information
  - OUTMODELINFO
  - OUTSELECT
- Forecast statistics
  - OUTSUM
  - OUTSTAT
- Other tables
  - MERGED_ATTRIBUTES
- Logs
  - OUTLOG

Variable Name	Name of produ...	Product Name	Mean Square Error	Mean Absolute Percent Error	Degrees of Freedom Error
sale	Line107	Product361	903.39180071	5.5389492517	60
sale	Line107	Product362	782.63157517	4.8091510669	60
sale	Line107	Product363	1666.1083078	7.1350733317	60
sale	Line107	Product364	3658.8296173	6.5414723768	60
sale	Line125	Product424	542.96803642	3.7918302651	60
sale	Line125	Product425	478.47807565	4.4917195396	60
sale	Line143	Product484	1122.4336479	6.7104553929	60
sale	Line143	Product485	630.15498345	6.7521027322	60
sale	Line143	Product486	449.94000455	57.198123791	60
sale	Line143	Product487	1801.632586	5.1151938556	21
sale	Line161	Product545	2764.7222117	6.9838279659	60
sale	Line161	Product546	1981.2676795	7.1752437835	60
sale	Line161	Product547	609.72240821	5.809320153	60
sale	Line178	Product603	570.13507441	5.3508462672	60
sale	Line178	Product604	315.14326587	5.3226421301	60
sale	Line178	Product605	2052.0421222	7.2282596509	60
sale	Line178	Product606	932.190751	5.4456592302	60
sale	Line196	Product664	1415.2727883	6.4492132919	60
sale	Line196	Product665	2022.3602465	7.2209940724	60

OUTSTAT is another table that is automatically generated in a project and available for export. This table contains a variety of fit statistics, like Mean Square Error, or MSE, and Mean Absolute Percent Error, or MAPE, associated with champion models for each BY group or series in the data. Information about the contents of other generated tables is found in the SAS Visual Forecasting User's Guide in Working With Strategy Nodes > Viewing the Results of a Modeling Strategy Node > Output Data.



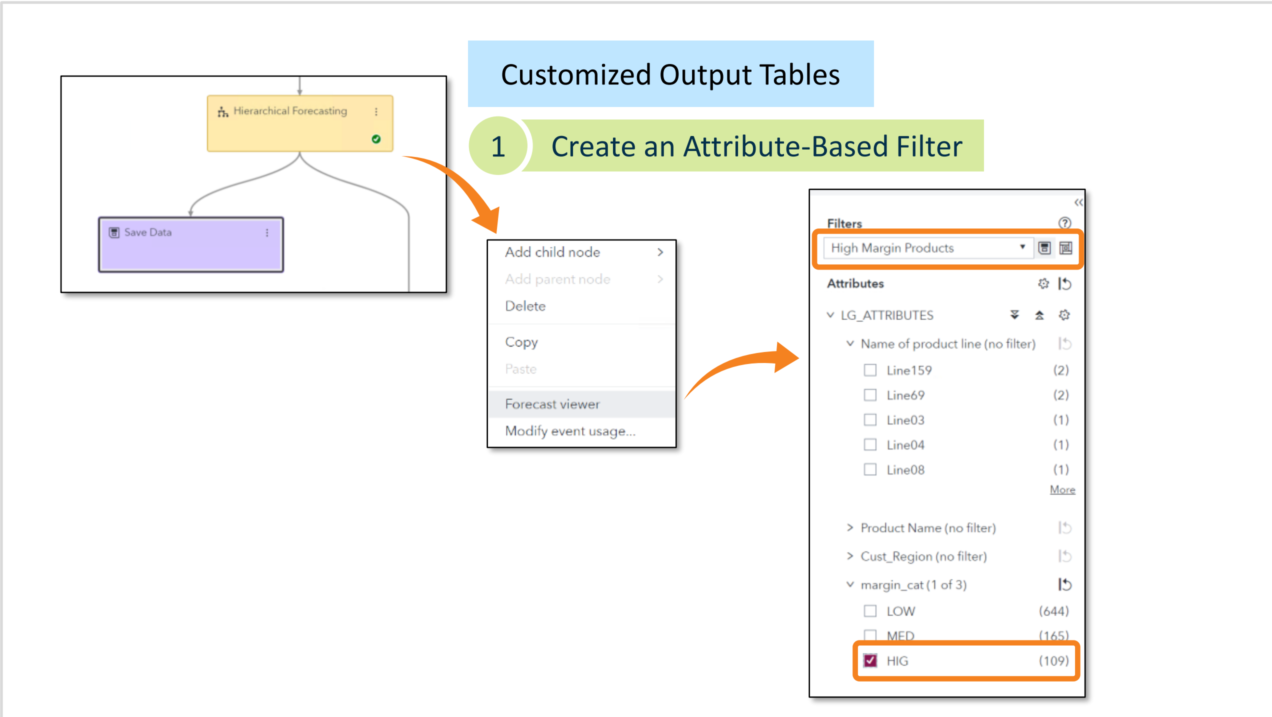
Other project output tables are available by request. For example, the OUTEST table contains parameter estimates and other information related to the champion model specifications for each BY group. This table can be generated in three steps. First, the table information is selected in the Output Tables property of the Forecasting Modeling node. Next, a Save Data node is added to the pipeline. Finally, using the **Edit save options** button in the properties of the Save Data node produces a window that enables users to name the table and choose a CASLIB destination for export.

Demo: Exporting Automatically Generated Tables

This demonstration illustrates exporting automatically generated tables from Modeling nodes and the Project.

Demo: Exporting Tables Generated by Request

This demonstration illustrates exporting tables generated by request from the Save Data node.

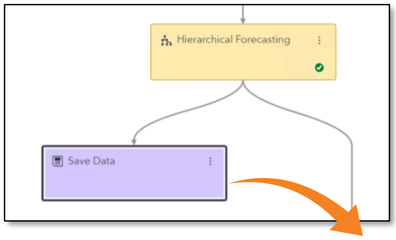


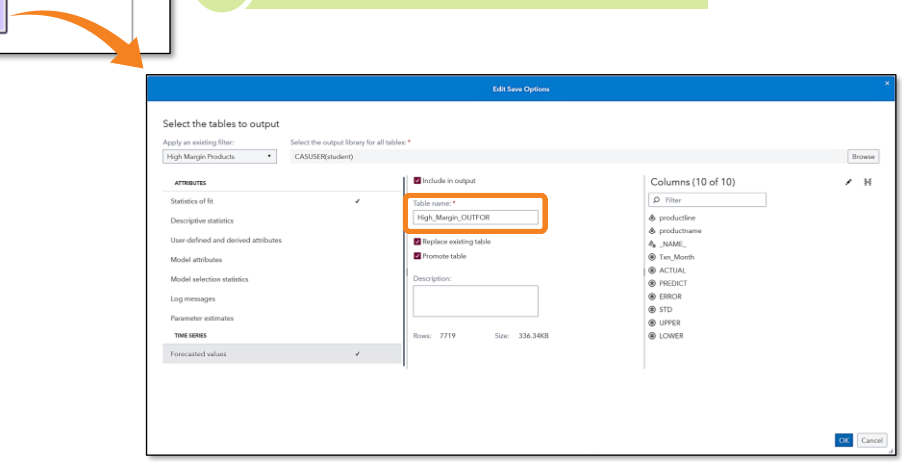
We've demonstrated the usefulness of attribute variables in other parts of the course. Attributes can be converted into filters and then used to customize post-forecasting tasks like applying overrides and the content of output tables. For instancewe can use the **Forecast viewer** of the Hierarchical Forecasting node to create a filter that selects series in the High Margin category.

Customized Output Tables

1 Create an Attribute-Based Filter

2 Apply the Filter





After the High_Margin_Products filter is created, it can be applied in the Save Data node to create a customized export table. The exported OUTFOR table High_Margin_OUTFOR will contain forecasts only for the 109 High Margin Category series in the data.

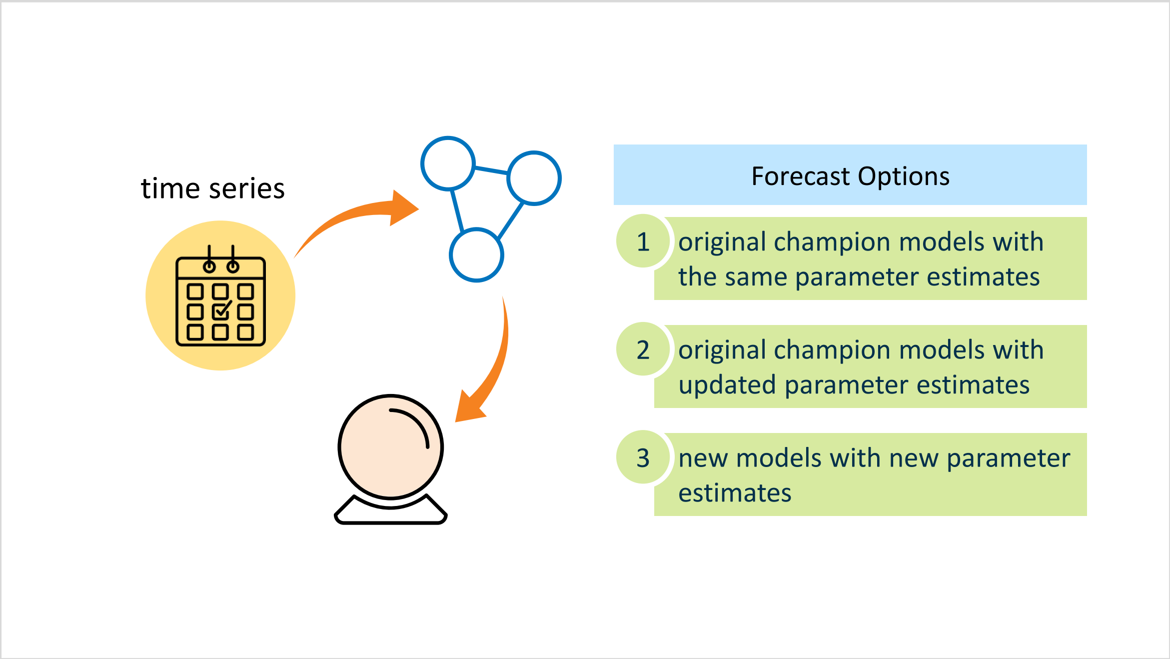
Demo: Creating Customized Output Tables

This demonstration illustrates customizing exported tables using filters and the Save Data node.

Forecast New Data and Task Options

How to forecast new data and exploring other task options in a time series

Lesson 05, Section 03



Let’s consider a scenario in which you select models for a series, calculating parameter estimates and forecasting using your preferred time horizon. Three months later, you'd like to forecast from the end of the series.

There are several options available in Model Studio to do this.

One option is to take the originally calculated champion models and use the same parameter estimates to forecast future series values.

A second option is to use the models selected from the original series, but update the parameter estimates based on the extended series. These parameter estimates will likely be somewhat different from the original parameter estimates, so the forecasts will also be somewhat different. This option is best if you want to refresh the forecasts without re-selecting the actual models.

If you feel that your models are stale or no longer reflective of the series dynamics, you can restart the entire process of selecting models, estimating parameters, and ultimately forecasting using the new models and parameters.

No matter which option you choose, remember to first select the updated data source into the project.

Demo: Updating Models and Forecasts

This demonstration illustrates building a Hierarchical Forecast Project based on monthly data and then updating models and forecasts.

Event Variables

How to utilize event variables in a time series

Lesson 05, Section 04

Introduction to Event Variables

events

negative article in major publication

advertising campaigns

scandals

mergers and acquisitions

strikes

retail promotions

disasters

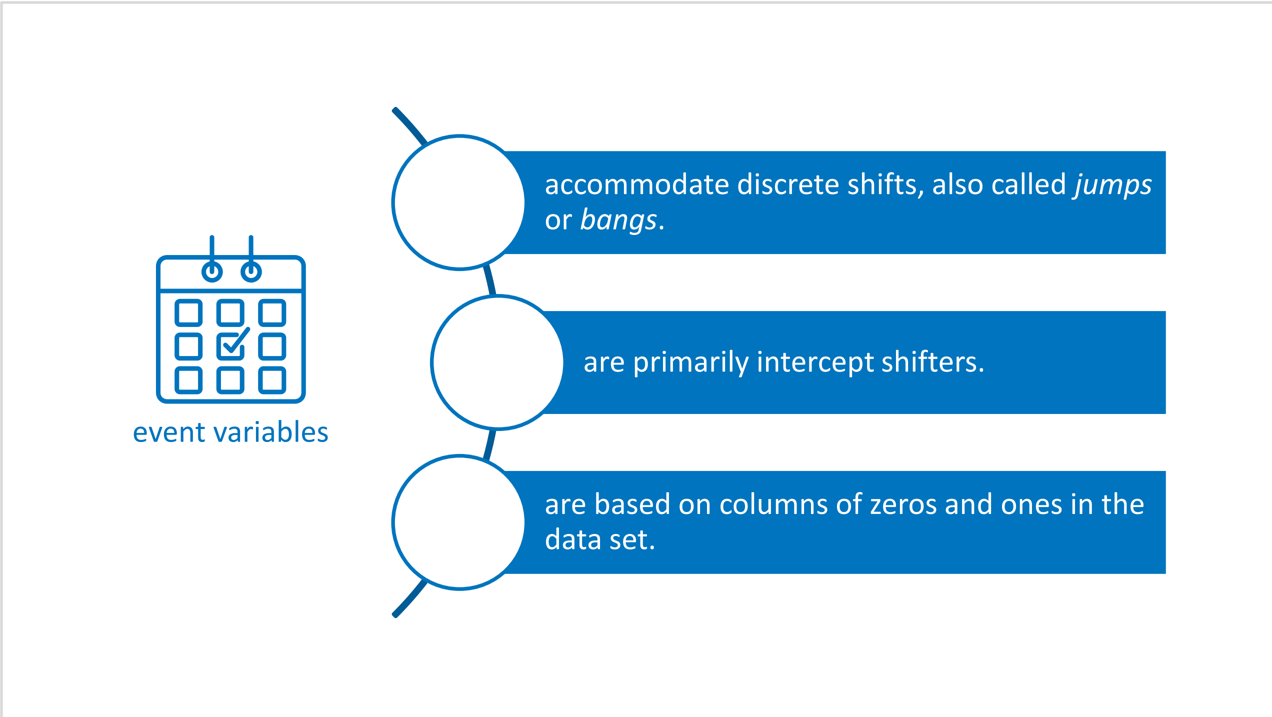
holidays

personnel/policy changes

key player injury, illness, or death

An *event* is anything that disrupts the underlying time series process that generates the data. Events generally manifest themselves as changes in the long-run components of time series. These include disruptions in level, trend, or seasonal components. Events include holidays, retail promotions, and natural disasters.

Event variables are included in the forecasting system to capture or accommodate variation associated with events in time series. An event variable is defined by the date or dates on which the event occurs. An event variable also 'qualifies' an event. That is, an event variable is defined in a way that tells a story about how the event occurrence manifests itself in the time series that is being modeled. For example, an event variable representing an immediate and permanent impact of a policy change might be qualified or coded as a step change. Another event variable that represents the impact of New Years can be coded as a pulse event that occurs only in the first week of the year, each year in the history, and in the lead forecast horizon.



Next, we'll discuss how event variables accommodate the effects of events in time series.

Event variables accommodate discrete shifts, also called *jumps* or *bangs.*

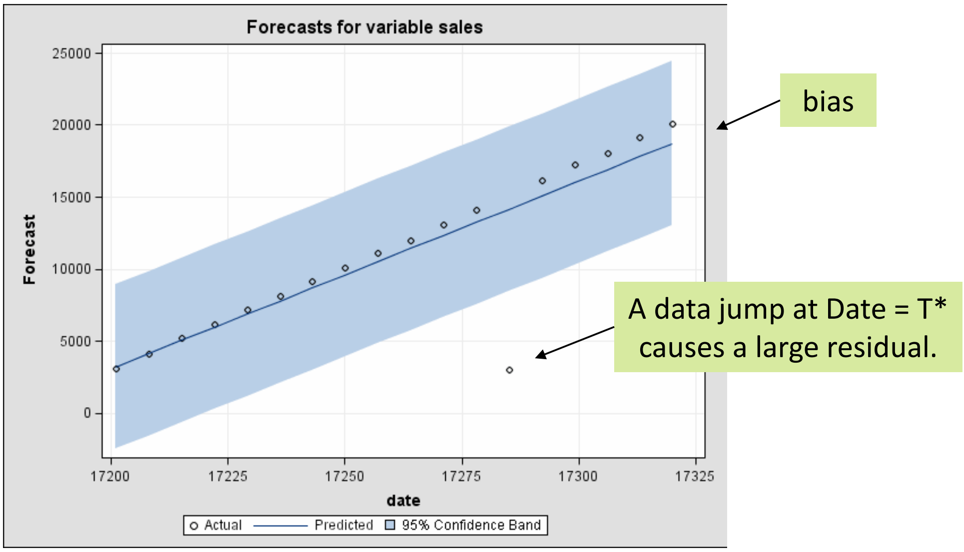
Event variables are primarily intercept shifters. Intercept shifters are included in the model as explanatory variables.

Event variables are based on columns of zeros and ones in the data set.

The focus is on how event variables are created and how they can improve the accuracy of models in the forecasting system.

The data is fit with a linear model:

$$sales_t = \mu + \beta\, trend_t$$



How do event variables improve accuracy?

Let's assume that the series we want to forecast follows a linear looking trend except for one observation. There are at least a couple reasons that you might want to accommodate the observation causing the large residual in the model. First, if the event is recurring, the event variable could be coded so that the variation associated with future instances is captured and future forecasts are improved. However, even if the event is believed to be a 'one-off' change, accommodating its variation in the model can reduce bias in the forecast.

$$sales_t = \mu + \beta\, trend_t$$

$$\Downarrow$$

$$sales_t = (\mu + \delta D) + \beta\, trend_t$$

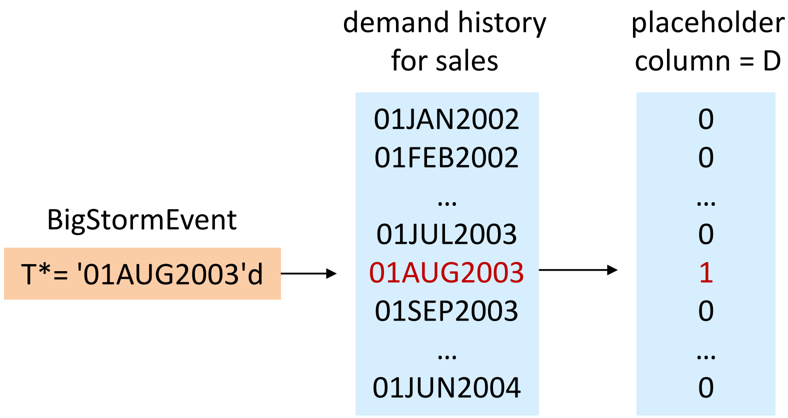
When date ≠ T*, *D* = 0, so the model's intercept = μ.

When date = T*, *D* = 1, so the model's intercept = $(\mu + \delta)$.

The linear model can be refined by modifying the intercept term.

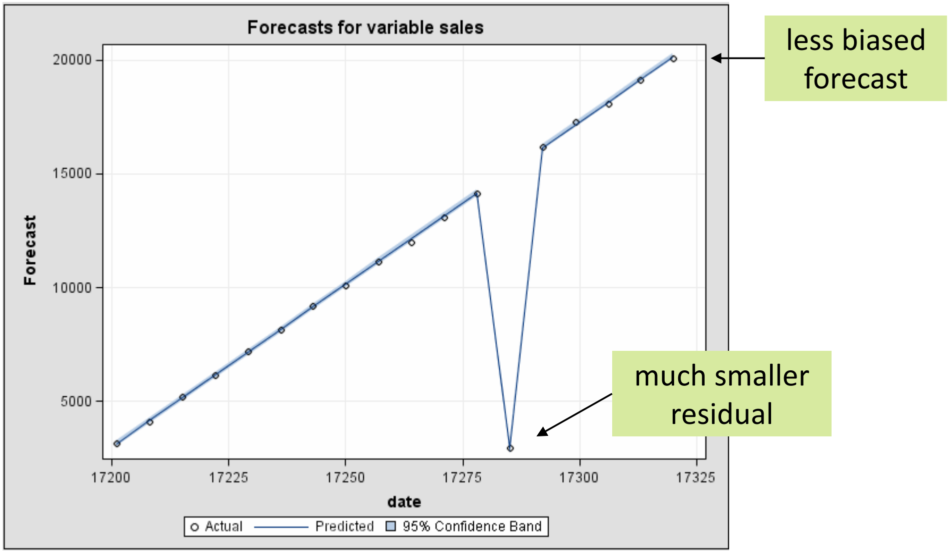
The event variable is represented as D in the equation. It acts like a switch to change the intercept of the model on the date of the event occurrence. At date T star, D is 'on', or is a one, so the intercept of the model represented by MU is modified by the value of the parameter DELTA. At all other dates, D is 'off', or equal to zero, so the intercept is just MU.

$$sales_t = (\mu + \delta D) + \beta\, trend_t$$



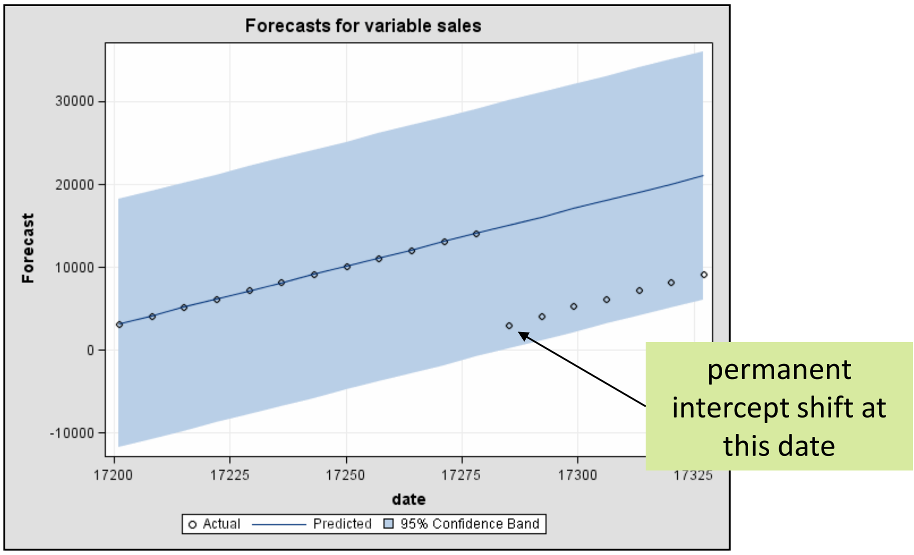
The placeholder coding of the event variable D that qualifies the event as an immediate and temporary change, or pulse, is shown. The temporary intercept shift is accomplished by adding a zero, one, or placeholder column to the data.

$$sales_t = (\mu + \delta D) + \beta\, trend_t$$



Here, the data is fit with a linear model that includes the pulse event variable. Adding the event variable to the model removes the large residual and mitigates the bias that it caused. Next, we will consider how to qualify a different event effect.

$$sales_t = \mu + \beta\, trend_t$$



Here, the model is initially fit to the pre-event data, and in this case, the event has manifested itself in what looks like a permanent change in the intercept.

$$sales_t = \mu + \beta\, trend_t$$

$$\Downarrow$$

$$sales_t = (\mu + \delta D) + \beta\, trend_t$$

When date < T*, *D* = 0, so the model's intercept = μ.

When date => T*, *D* = 1, so the model's intercept = $(\mu + \delta)$.

The model equation is the same as before, but the event variable D will be coded differently to qualify the event as a step, or immediate and permanent change to the level of the series. Before time T star, the intercept is MU. On and after T star, the intercept is MU plus DELTA.

$$sales_t = (\mu + \delta D) + \beta\, trend_t$$

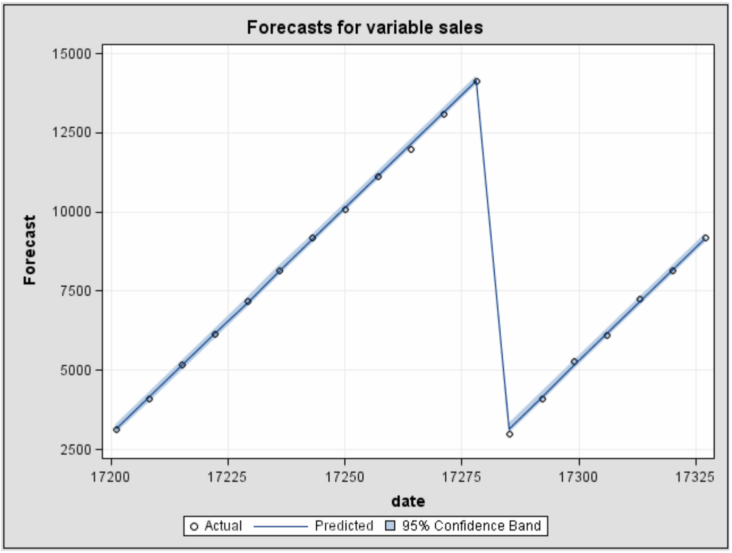
NewLawEnacted

T*= '01AUG2003'd →

demand history for sales	placeholder column = D
01JAN2002	0
01FEB2002	0
…	…
01JUL2003	0
01AUG2003	1
01SEP2003	1
…	…
01JUN2004	1

The permanent intercept shift is accomplished by adding a zero, one, or placeholder column to the data. The placeholder coding reflects the step qualification.

$$sales_t = (\mu + \delta D) + \beta\, trend_t$$



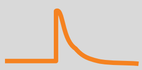
Here, the data is fit with a linear model and a step event variable. The permanent shift in the level of the series is accommodated in the model via the intercept and the step event variable, D.

Think About It: Event Variable Models

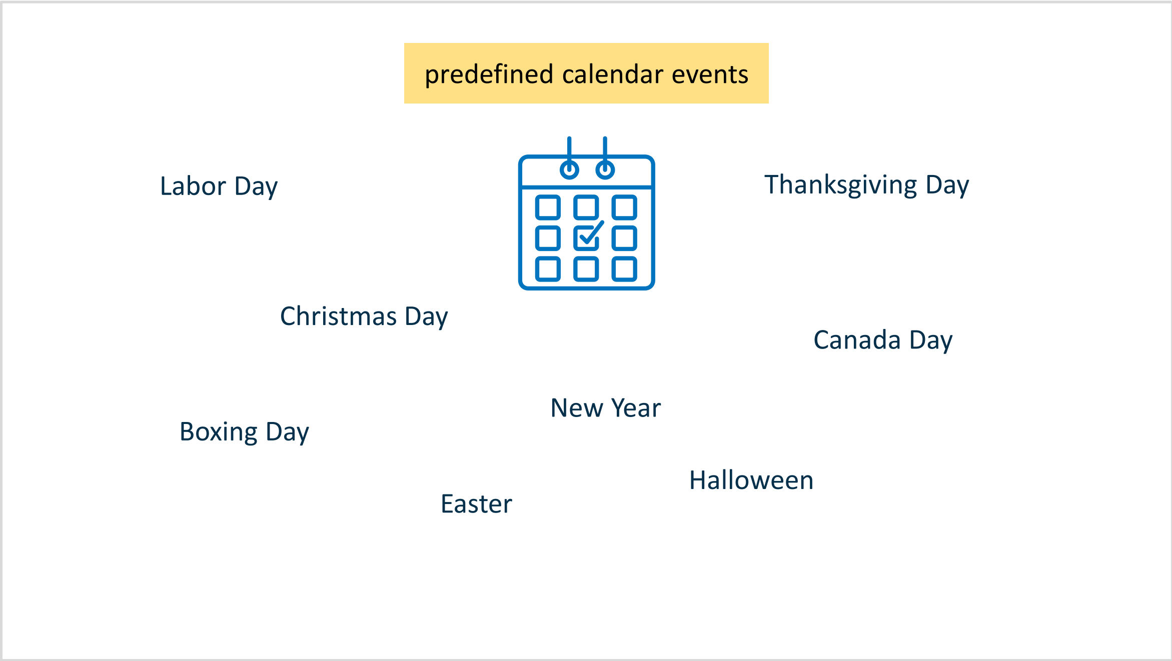
A team member suggests building event variables into the model for all intervals with large residuals in the forecast for a high value series. Why might this be a bad idea?

Event variables can be useful for improving the fit of the model. However, indiscriminately creating lots of event variables to effectively remove all large residuals usually results in overfitting the data. Large residuals can be a result of a large variance in the series or big noise, and fitting noise is overfitting by definition. Big noise cancels out over time. A good rule of thumb is to create event variables only for events that you or a subject matter expert can name and explain. For example, after finding a large unexplained residual, you investigate and discover that it is the result of an undocumented promotion. In this case, it can improve the precision of the forecast to accommodate the promotion's effect with an event variable in the model.

Event Variables in SAS Visual Forecasting

Event type	Shape of time series	Description
pulse		Temporary change in the magnitude of a time series process. The magnitude returns to the former level immediately after the change.
level shift		Persistent change in the level of a time series process.
ramp		Persistent change in the trend or slope of a time series process.
temporary change		Temporary change in the magnitude. The magnitude decays to the former level immediately after the change.

The basic event variable types available in SAS Visual Forecasting are shown here. Note that more flexible event qualifications are feasible. For example, steps don't have to be permanent and can be truncated. Pulses can be 'stretched' to accommodate variation that persists for more than one interval.



There are currently over 20 predefined event variable keywords in the software, and these accommodate recurring types of events in the data. For example, most of the series in your project might have a Christmas effect, year over year. Using the CHRISTMAS keyword, a placeholder variable can be built that flags each Christmas in the history and in the lead forecast horizon for each series.

You might also have an event that occurs relative to a holiday interval. For example, maybe you have an event that occurs two weeks before Easter week each year. The software is flexible enough to shift the qualification of an event variable based on the EASTER keyword to accommodate this pattern.

Demo: Using Keyword-Based Event Variables in an Automatic Forecasting System

This demonstration illustrates adding predefined event variables to the project.

Custom Event Variables

```
proc hpfevents;
     eventdef Sep01step = '01sep2001'd /
                    type=ls;
       …
     eventdata out=work.myhpfevents;
quit;
```

Let's talk about adding a custom event variables table to your project. One way to create custom event variables is to use the HPFEVENTS procedure. This procedure is based on SAS 9 functionality, but it can produce tables of custom event variables that can be read into memory and subsequently used in be used in Visual Forecasting projects. Here, an EVENTDEF statement is used to define a custom event variable. The EVENTDATA statement creates an event variables table that is used to reference the custom events.

Demo: Adding Custom Event Variables in an Automatic Forecasting System

This demonstration illustrates adding custom event variables to the project.

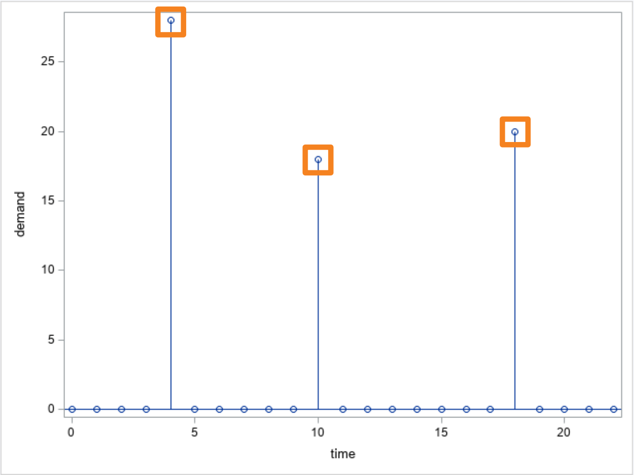
Intermittent Demand Models

How to determine if an intermittent demand model is needed in a time series

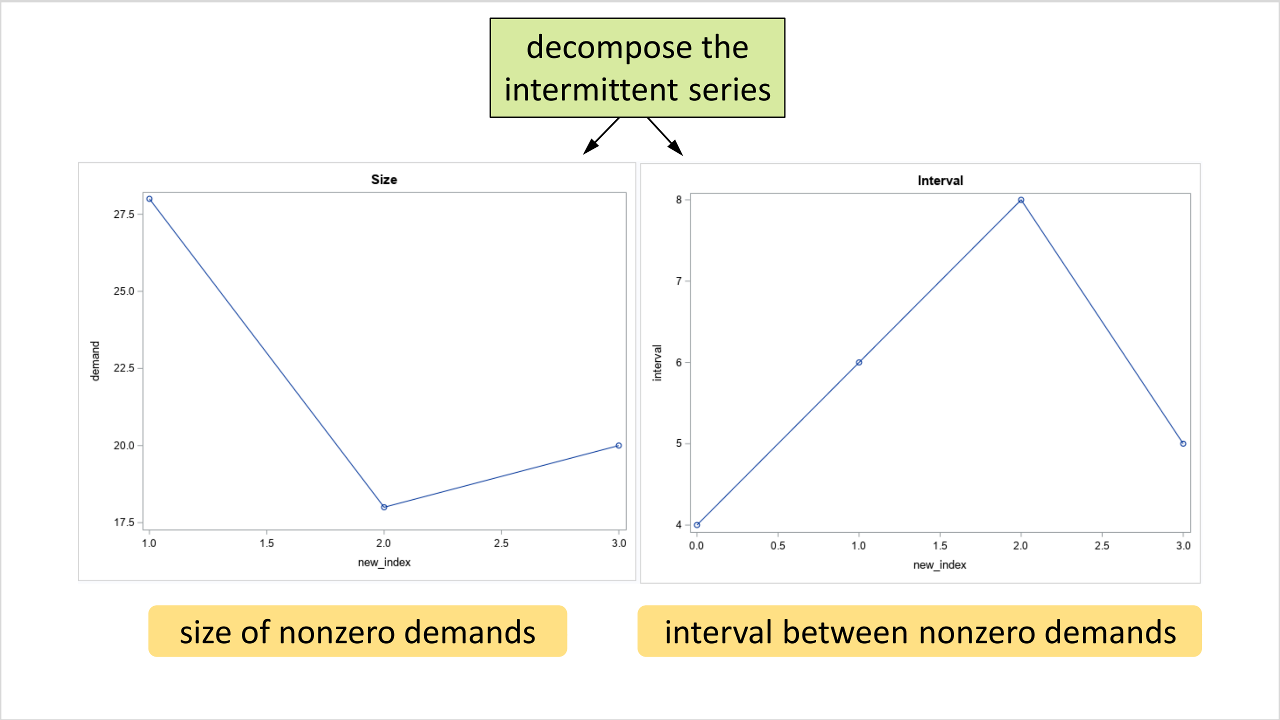
Lesson 05, Section 05

Intermittent Demand Model

intermittent
time series



An intermittent time series has observations that are mostly zero, or some other base value. In the series shown, there are 23 observations, and only three are nonzero. Because most of the values are zero, the most likely forecast, or expected value, for the next interval is zero. It's probably more useful to predict when the series departs from zero and the magnitude of this departure than it is to forecast future values. The goal of intermittent demand models is to predict the average departure from zero for each time period.



To understand the way an intermittent demand model or IDM works, think of decomposing the intermittent time series into two series: size and interval. The size component series measures and can be used to predict the magnitude of the departures from zero. The interval component measures and is used to predict the interval between departures from zero. The lead forecast from the IDM is the predicted average departure for future time periods.

partially observed

Obs	demand	interval	avg_demand
1	28	4	7.0
2	18	6	3.0
3	20	8	2.5

Average of fully known average demands = 2.75

An average demand series is constructed by dividing the size, or nonzero demand series, by the interval series. Note that the first interval observation is incompletely observed: the first nonzero demand occurs 4 intervals after the series starts. This means that the interval between the first observed nonzero demand and the previous one is at least 4. Only fully known demands and intervals are used to construct the average demand series. The average of fully known average demands is therefore 2.75.

Croston Model $\widehat{AD} = ESM(Size)/ESM(Interval)$

Average Demand Model $\widehat{AD} = ESM(AD)$

Croston and average demand are the two IDM models supported in the software to forecast the average demand per period. Both produce average demand predictions, denoted AD in these formulas, using forecasts from an exponential smoothing model, or ESM. The Croston prediction derives the average demand forecast by dividing the ESM prediction for the size series by the ESM prediction for the interval series. The average demand model's prediction is an ESM prediction of the average demand series.

Obs	demand	interval	avg_demand
1	28	4	7.0
2	18	6	3.0
3	20	8	2.5

default threshold = 2

When the IDM functionality is active, the software determines whether a series is intermittent by calculating the median of fully observed intervals between nonzero demands. If this value exceeds a threshold, 2 by default, then the series is declared intermittent. Only IDM models are considered in the automatic modeling process for series that have been declared intermittent.

Demo: Exploring IDM Models in Model Studio

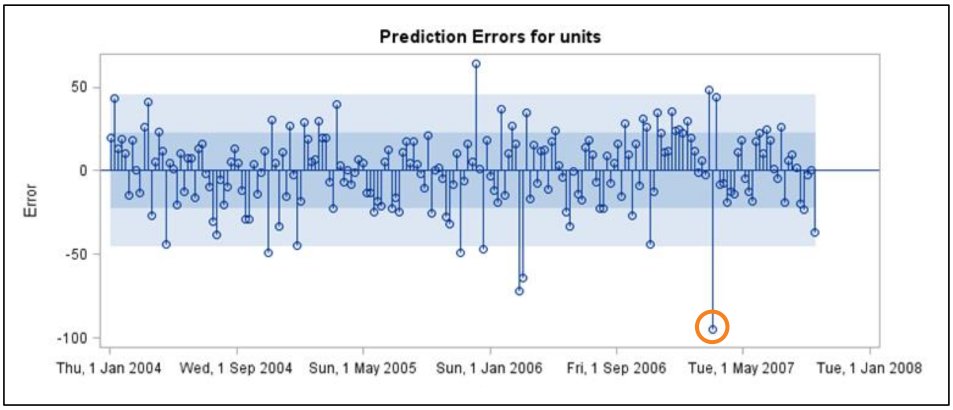
This demonstration explores IDM models in Model Studio.

Outlier Detection

How to perform outlier detection in a time series

Lesson 05, Section 06

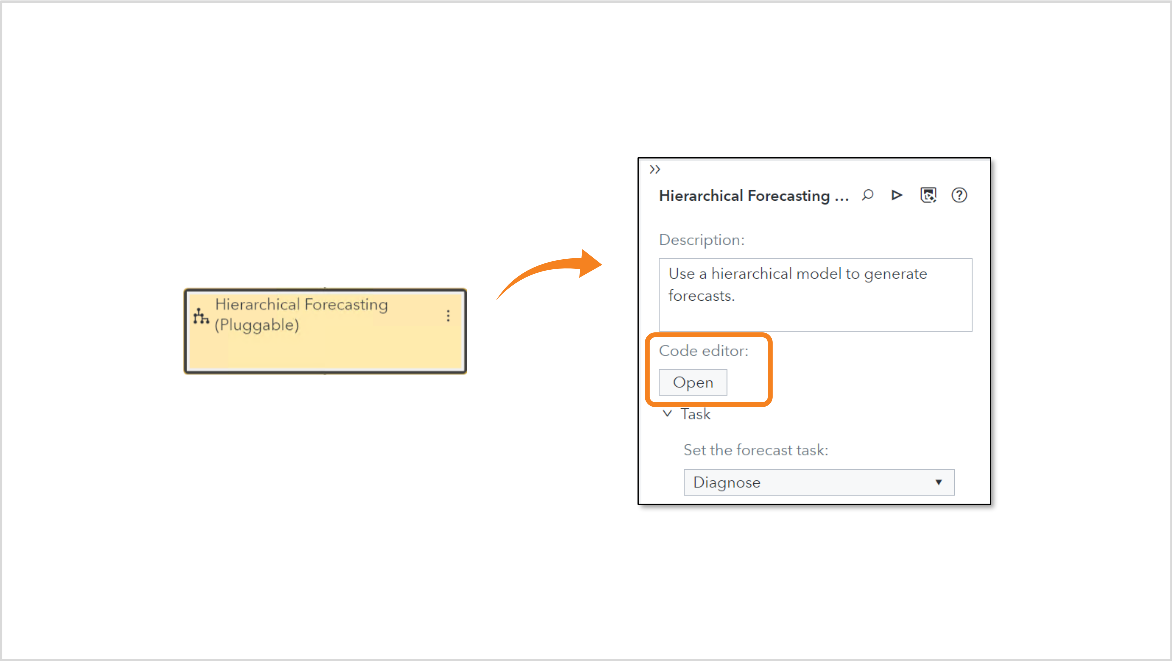
Outlier Detection



A common predicament in large-scale forecasting is that there's not enough time or resources to scrutinize each series that needs forecasting. An analyst might suspect that there are anomalies in the data, but lack the time and resources to find each change, build an appropriate event variable for each change, and incorporate the event variables into appropriate model specifications. Outlier detection is primarily useful in these circumstances.

An informal definition of an *outlier* is an event that the analyst did not know about. Outliers include undocumented promotions, competitor price movements, and random shocks.

Outlier detection is an algorithm that detects structural changes in the form of large residuals in system-generated ARIMAX models. The default definition of an outlier is any residual that is over three standard deviations away from the residual mean. Outlier variables are system-generated variables, and they're similar to event variables. When an outlier is detected, it can be classified by the software in three ways: as a step or LS, as a pulse or AO, or as a temporary change or TC. Outliers are not detected by default.



We'll use the in-line code that is available in a Hierarchical Forecasting Pluggable node to activate the outlier detection functionality. In-line code provides users access to the complete range of functionality in the Visual Forecasting, Automatic Time Series Modeling package and allows full customization in how the node is set up and run. In-line code is also available from the Auto-Forecasting node. Interested students can find more details in the Appendix.

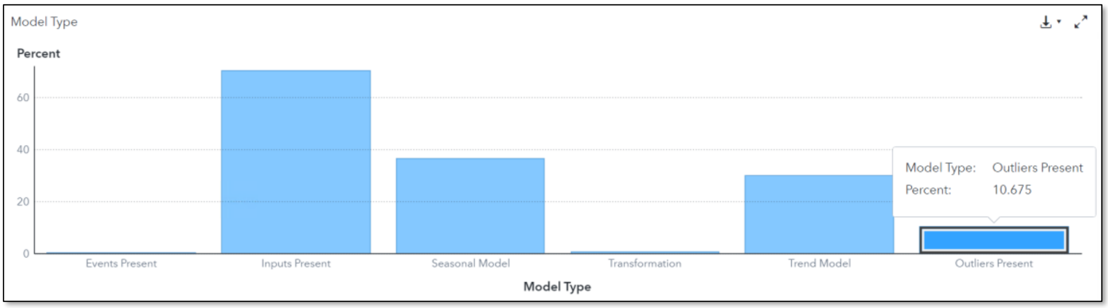
```
/*-------------------------------------------------------
 * Define the diagnose part of script to run in TSMODEL
 *-----------------------------------------------------*/
%macro tsmodelCodeDiagnose;

    /*setup time series diagnose specifications*/
    rc = diagSpec.open();
    %if %UPCASE("&_esmInclude") eq "TRUE" %then %do;
        rc = diagSpec.setESM();
    %end;
    %if %UPCASE("&_arimaxInclude") eq "TRUE" %then %do;
        rc = diagSpec.setARIMAX('IDENTIFY', 'BOTH');
        rc = diagSpec.setARIMAXRefine('ORDER', 'INPUT');
        rc = diagSpec.setARIMAXOutlier('DETECT', 'YES');
    %end;
```

The in-line code is object-oriented and consists of objects and methods. Outlier detection is available only for system-generated ARIMAX models. It's switched on in the SETARIMAXOUTLIER method associated with the DIAGSPEC type object. Changing this method's arguments to DETECT, YES specifies that outlier detection is performed, and that a detected outlier should be accommodated in a model if it improves the overall fit.

Adding lots of outlier variables to a model can easily lead to overfitting, and it's good practice to limit the number of outlier variables accommodated in a model. The default, maximum number of outliers detected for any series is 2.

Students interested in learning more about setting up their projects using TSMODEL syntax can find more information in the class Large-Scale Forecasting Using SAS Viya: A Programming Approach.



Here we see the model type results following outlier detection. About 11 percent of the series now have a generated ARIMAX model with one or more detected outlier variables as the champion model.

prod...	productname	_MODEL_	_MODELVAR_	_EST_
Line189	Product641	DIAG1_REGARIMA1	Y	760.63419121
Line189	Product641	DIAG1_REGARIMA1	Y	0.609131801
Line189	Product641	DIAG1_REGARIMA1	price	-3.042730273
Line189	Product641	DIAG1_REGARIMA1	price	1.3081371093
Line189	Product641	DIAG1_REGARIMA1	price	-0.261501159
Line189	Product641	DIAG1_REGARIMA1	price	-0.758493051
Line189	Product641	DIAG1_REGARIMA1	AO01May2012D	85.631815947
Line212	Product719	DIAG1_ARIMAX1	Y	-0.517001149
Line212	Product719	DIAG1_ARIMAX1	price	-6.840628072
Line212	Product719	DIAG1_ARIMAX1	AO01Aug2015D	111.85181991
Line262	Product888	DIAG1_REGARIMA1	Y	0
Line262	Product888	DIAG1_REGARIMA1	SUMMER	30.760620719
Line262	Product888	DIAG1_REGARIMA1	price	-2.274103166
Line262	Product888	DIAG1_REGARIMA1	AO01May2014D	97.783897696

An outlier variable's name that is prefixed by AO indicates an additive outlier, or a pulse. This means that an outlier that lasts for one interval was detected, and an AO type variable that codes it as a pulse was added to the model. The date part of the name tells you when the detected outlier occurs or starts.

Other outlier variable types are LS and TC. LS stands for level shift or step. TC, or temporary change, codes a detected outlier that persists for more than one interval, but that doesn't persist forever.

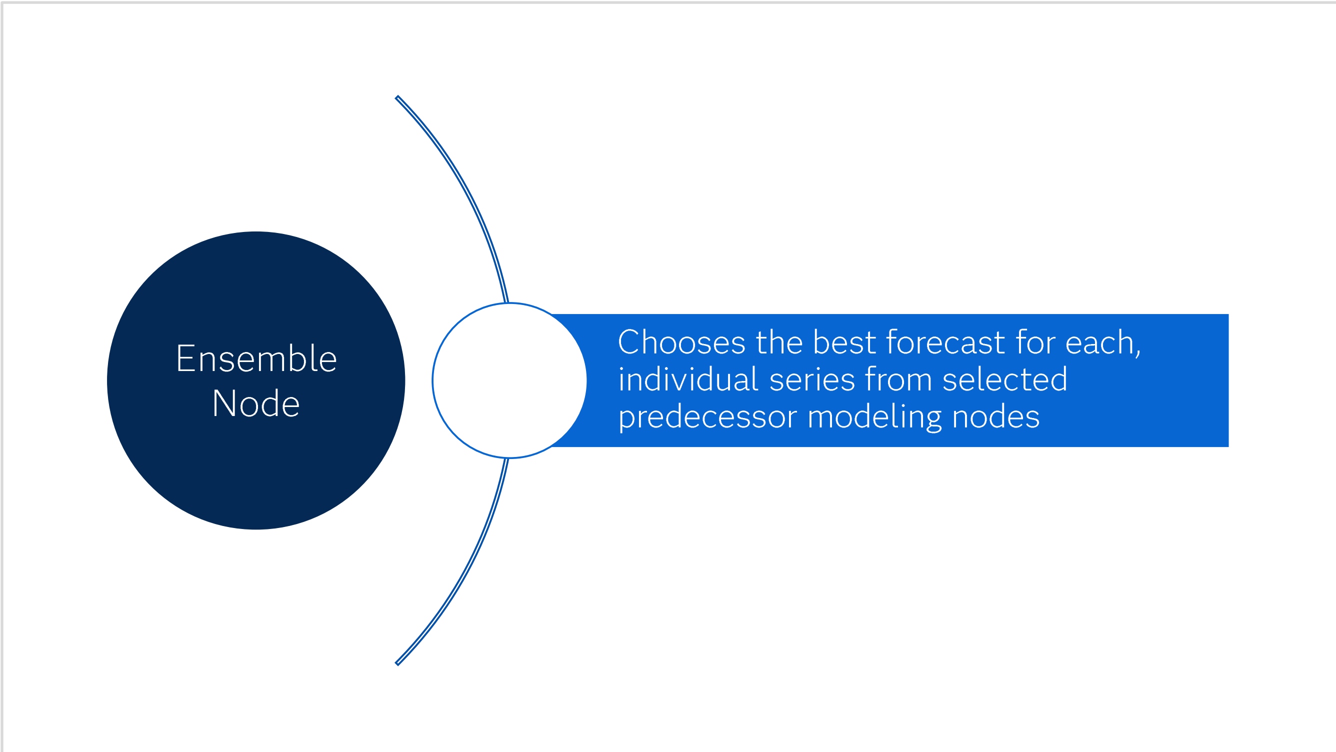
Demo: Adding Outlier Detection to the Forecasting System

This demonstration illustrates adding Outlier Detection to the forecasting project.

Ensemble Node

How to use the Ensemble Node

Lesson 05, Section 07



The champion modeling node within a pipeline is selected based on an aggregated measure like Weighted MAPE. While the champion modeling node has the best performance based on averaging fit measures across all forecasts, it may not contain the best forecast for a given individual series. The Ensemble node evaluates the champion forecasts from selected, predecessor modeling nodes in a pipeline and selects the best forecast for each series based on a fit statistic selected by the analyst.

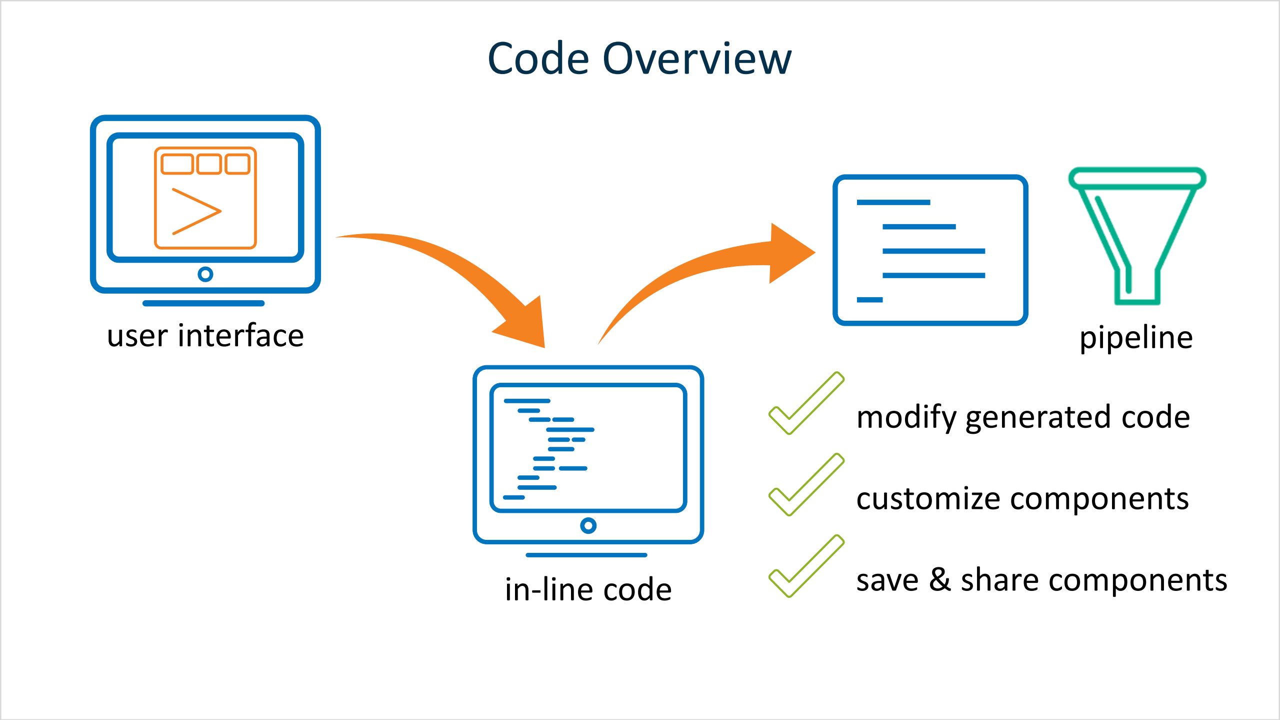
Demo: Using the Ensemble Node

This demonstration illustrates the functionality of the Ensemble node.

In-Line Code Access and Overview

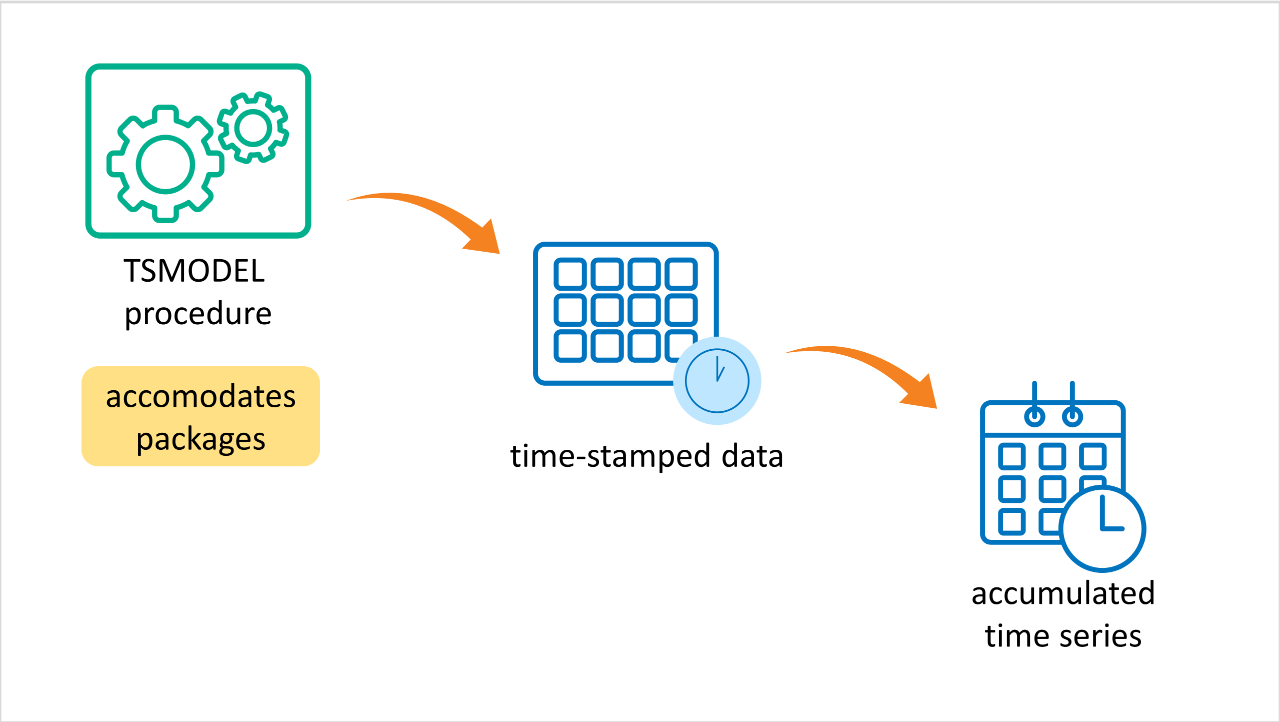
How to alter in-line code to adjust a time series analysis

Lesson 06, Section 01

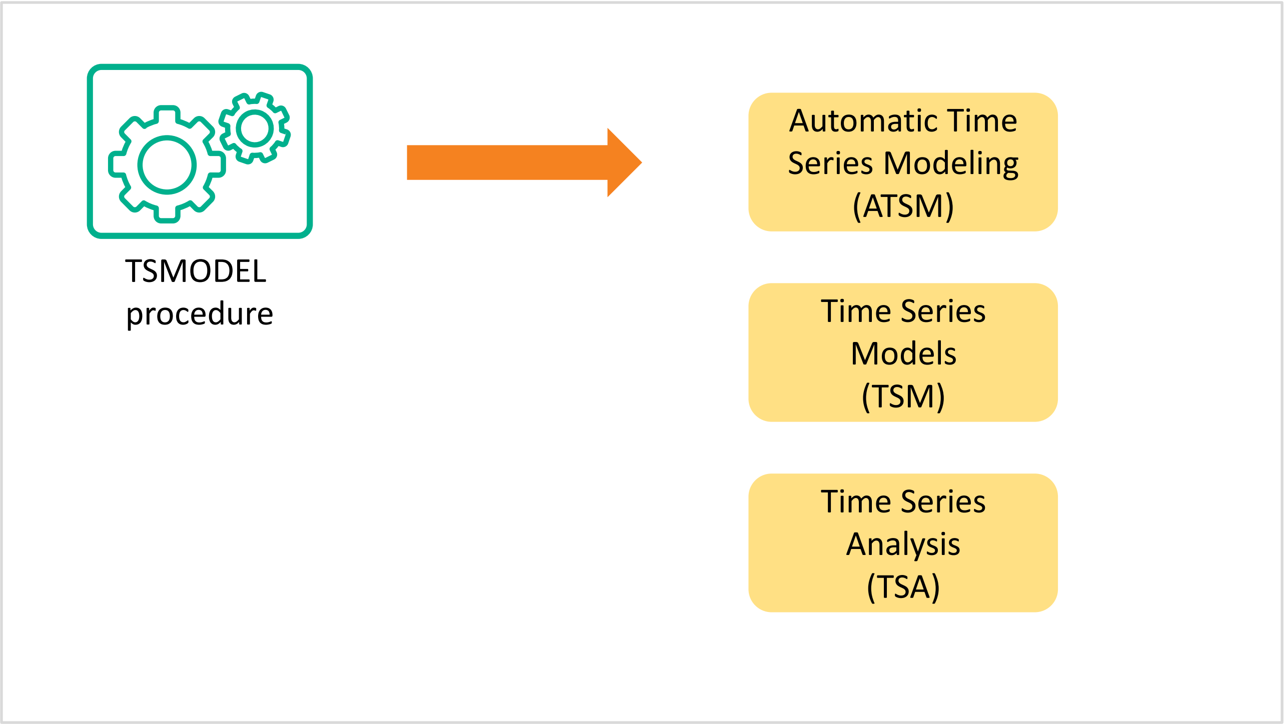


Let's switch our focus from the Model Studio user interface to the in-line code that is generated "under the hood." The goal is to overview how the syntax that is accessible from a pipeline is laid out. We'll describe what the main parts of the syntax do and illustrate how the information flows into and out of each part or chunk of code. Our goal is to familiarize analysts with the generated syntax to the extent that they are comfortable accessing it, making straightforward modifications to it, and saving it.

Access to the code expands the functionality of the UI. One of the useful features of SAS Visual Forecasting is the ability to modify the generated code and then rerun the project in Model Studio. Another unique and productive feature is the ability to customize components of the project via modifications to the code, and then save and share these components using the Exchange.



The TSMODEL procedure is a SAS Viya procedure that executes user-defined programs on time series data. The TSMODEL procedure analyzes timestamped transactional data and accumulates the data into time series. TSMODEL also accommodates packages, which are containers for objects that enhance the functionality of the TSMODEL procedure.



One package is the automatic time series modeling (ATSM) package for the TSMODEL procedure, which provides objects that support automatic time series modeling and automatic forecasting. There are other packages as well. For example, time series models (TSM) enable users to specify and fit their own custom forecasting models, and the time series analysis (TSA) package contains a set of time series diagnostic functions that can be used as part of the programming statements in the TSMODEL procedure. Your SAS license determines the packages that are available to you.

TSMODEL procedure

```
proc tsmodel data = …;
    …
  require ATSM;
  submit;
    …
    declare object mytsdata(tsdf);
    rc = mytsdata.initialize();
    rc = mytsdata.addY(mytarget);
    …
  endsubmit;
```

There are specific steps in using packages and the objects contained in them. Package objects need to be declared and then initialized. Following this, actions can be run on objects. Let's look at the main syntax involved in specifying a package, implementing one of its objects, and then running an action on the object. A time series data frame (TSDF) type object is declared and named **mytsdata**. After it is initialized, an **addY** method is run on it to populate the target variable name, **mytarget**, for the analysis.

Demo: Auto-forecasting Code Overview

This demonstration overviews the code that can be accessed from the Auto-forecasting node.

Questions?

Demo: Modifying the Auto-forecasting Code and Creating a Custom Forecast Node

This demonstration provides an example of how the functionality of a forecasting node can be customized via the code.

Questions?

Demo: Overview and Modification of the Hierarchical Forecasting (Pluggable) Node

This demonstration provides an example of how the functionality of a forecasting node can be customized via the code.

Questions?

Demo: Modifying and Saving a Hierarchical Forecasting (Pluggable) Node

This demonstration provides an example of how the functionality of a forecasting node can be customized via the code.

Questions?

Resources

Lesson 06, Section 02

HTML Placeholder

References

Brown, Robert Goodell. 1959. *Statistical Forecasting for Inventory Control*. McGraw-Hill: New York, NY.

Brown, Robert Goodell. 1962. *Smoothing, Forecasting, and Prediction of Discrete Time Series*. Prentice-Hall: Englewood Cliffs, NJ.

Chase, Charles. 2009. *Demand-Driven Forecasting: A Structured Approach to Forecasting*. Hoboken, NJ: John Wiley and Sons.

Fildes, Robert, and Paul Goodwin. 2007. "Good and Bad Judgement in Forecasting: Lessons from Four Companies." *Foresight: The International Journal of Forecasting* 8:5-10.

Fomby, Thomas B. June 2008. *Exponential Smoothing Models*. Southern Methodist University. Dallas, TX.

Gilliland, Michael. 2010. *The Business Forecasting Deal*. Hoboken, NJ: John Wiley and Sons.

Holt, Charles C. 1960. *Planning Production, Inventories, and Work Force*. Prentice-Hall: Englewood Cliffs, NJ. (Chapter 14).

SAS Institute Inc. 2017. *SAS/ETS 14.3 User's Guide*. Cary, NC: SAS Institute Inc.

Yaffee, Robert A., and Monnie McGee. 2000. *Introduction to Time Series Analysis and Forecasting with Applications of SAS and SPSS*. Academic Press: San Diego, CA.

Yule, George U. 1926. *Journal of the Royal Statistical Society*. 89(1):1 – 63.

ARIMA:

Box, G. E. P., G. M. Jenkins, and G. C. Reinsel. 1994. *Time Series Analysis: Forecasting and Control*. Upper Saddle River, NJ: Prentice Hall.

Brockwell, Peter J., and Richard A. Davis. 1992. *Time Series: Theory and Methods*. New York, NY: Springer-Verlag Inc.

UCM:

Harvey, Andrew C. 1989. *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge, UK: Cambridge University Press.

Koopman, Siem Jan, and James Durbin. 2001. *Time Series Analysis by State Space Methods*. Oxford, UK: Oxford University Press.