We want to build a time series model that forecasts the monthly number of passengers travelling via this particular airline. This would be a valuable insight for the company since they want to tackle the demand of the passengers as well as set a price which optimizes the company’s profit. `

First of all, let’s check the contents of our data, using the following code:

**ODS HTML File = 'C:\Documents and Settings\STD\Desktop\SasClasses\Time Series Air\Contents of Air Data.xls';**

**Proc Contents Data = ts.air;**

**Run;**

**ODS HTML Close;**

|  |  |
| --- | --- |
| **Observations** | 144 |
| **Variables** | 2 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Variable** | **Type** | **Len** | **Format** | **Label** |
| **2** | AIR | Num | 8 |  | international airline travel (thousands) |
| **1** | DATE | Num | 8 | MONYY. |  |

So, we have 2 variables: DATE (which contains dates in the form of MONYY) and AIR (which contains the monthly number of passengers in thousands). Total number of observations is 144 i.e. Monthly data starting from Jan1949 to Dec1960. This can be explored by looking at the data.

Now, before any kind of analysis, we must prepare our data for analysis by checking for any outliers and missing data.

The following code gives us an insight about the distribution of the data:

**ODS HTML File = 'C:\Documents and Settings\STD\Desktop\SasClasses\Time Series Air\Outliers Check.xls';**

**Proc Univariate Data = ts.air;**

**Var air;**

**Run;**

**ODS HTML Close;**

|  |  |
| --- | --- |
| **Quantiles (Definition 5)** | |
| **Quantile** | **Estimate** |
| **100% Max** | 622 |
| **99%** | 606 |
| **95%** | 491 |
| **90%** | 461 |
| **75% Q3** | 361 |
| **50% Median** | 265.5 |
| **25% Q1** | 180 |
| **10%** | 135 |
| **5%** | 121 |
| **1%** | 112 |
| **0% Min** | 104 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Extreme Observations** | | | |
| **Lowest** | | **Highest** | |
| **Value** | **Obs** | **Value** | **Obs** |
| 104 | 11 | 535 | 138 |
| 112 | 1 | 548 | 127 |
| 114 | 23 | 559 | 128 |
| 115 | 13 | 606 | 140 |
| 118 | 12 | 622 | 139 |

The first table represents the various percentiles across the variable ‘AIR’. We can see that starting from 104, every next step shows a consistent growth which goes up till 622. Also, from the second table, we can see that the extreme 5 observations on both the ends have no ‘odd’ high or low values. Thus, we have no outliers in the data.

Next, let’s check for any missing values, using the following code:

**ODS HTML File = 'C:\Documents and Settings\STD\Desktop\SasClasses\Time Series Air\Missing Check.xls';**

**Proc Means data = ts.air nmiss;**

**Run;**

**ODS HTML CLOSE;**

|  |  |  |
| --- | --- | --- |
| **Variable** | **Label** | **N Miss** |
| **DATE** | **international airline travel (thousands)** | 0 |
| **AIR** | 0 |

The above table shows that both of the variables have no missing values.

So, there are no missing values and outliers in the data (LUCKY FOR ME!). Now, our data is ready to be analysed.

Before proceeding with it, we will first divide our dataset into 2 parts: training data and test data. But why do we do it?

Basically, we want to build a forecasting model using a subset of the whole data, so that later we can compare the estimated values with the actual values. Here, we will use the data (training data) from the first 11 years, to forecast the number of passengers for the 12th year. This way, we get to check the accuracy of our model by comparing the 12th year estimates with the actual values of 1960 (test data).

The following code will do the trick:

**Data ts.train ts.test;**

**Set ts.air;**

**If Year(Date) = 1960 then output ts.test;**

**Else output ts.train;**

**Run;**

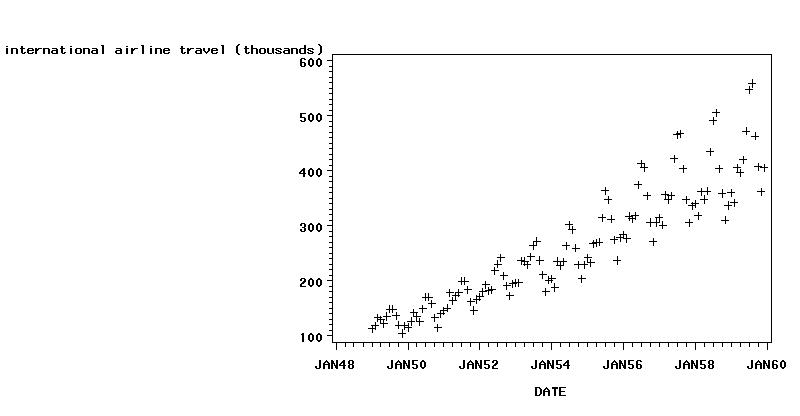
Okay! Now all the operations will be done on the training data (ts.train).

Now let's start with visualizing our data. The following code does that:

**Proc GPlot Data = ts.train;**

**Plot Air\*Date;**

**Run;**

****

We can clearly see that the above time series is a non-stationary time series because:

1. The mean is changing over time i.e. we can clearly see that there is a growing tendency in the time series. In mathematical terms, mean is a function os time.
2. The Variance is changing over time i.e. the spread is clearly increasing along y-axis . Variance is also a function of time here.

Another way of checking for stationarity of a time series is known as Augmented Dickey Fuller’s Test (or ADF Test). It is a statistical test, which is a scientific and more sophisticated approach than just inspection of the graph above. Also, it is one of the most popularly used statistical tests for this task. Here’s the code for it:

**Proc Arima Data = ts.train;**

**identify var = air stationarity = (ADF);**

**Run;**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Type** | **Lags** | **Rho** | **Pr < Rho** | **Tau** | **Pr < Tau** |
| **Zero Mean** | **0** | 0.1438 | 0.715 | 0.11 | 0.717 |
|  | **1** | -0.4523 | 0.5792 | -0.26 | 0.5912 |
|  | **2** | 0.0666 | 0.6968 | 0.05 | 0.6967 |
| **Single Mean** | **0** | -5.8105 | 0.3561 | -1.73 | 0.4156 |
|  | **1** | -10.8506 | 0.103 | -2.28 | 0.1782 |
|  | **2** | -7.1422 | 0.2589 | -1.79 | 0.3858 |
| **Trend** | **0** | -36.4924 | 0.0011 | -4.54 | 0.0019 |
|  | **1** | -112.588 | 0.0001 | -7.34 | <.0001 |
|  | **2** | -159.569 | 0.0001 | -6.88 | <.0001 |

The highlighted values are the p-values for the various estimates of ‘Rho’ and ‘Tau’ obtained under different variants of ADF test. **All** of these values must be significant (preferably less than 0.01) to conclude that we have a stationary time series.

Here, we have a non-stationary time series, as expected. Most of the real-life time series that we obtain are non-stationary in nature. We have to apply various transformations and use techniques to make it stationary. It is important because most of our most commonly used models work only with stationary series.

The idea is to transform the time series into a stationary one, build a suitable model, forecast the future values, and ‘de-transform’ the forecasted values to obtain the insightful forecasts.

There are some very sophisticated techniques to do this task such as decomposition of series, moving averages, (exponential) weighted moving averages, but let’s implement a basic technique of removing trend and seasonal components from our series.

We will try to transform the series, which is expected to remove trend component from the series. Let’s try taking the log and Square Root of the series:

**Data ts.train;**

**Set ts.train;**

**Log\_Air = Log(Air);**

**SQRT\_Air = SQRT(Air);**

**Run;**

A quick ADF Test and scatter plots of each of these series will again indicate non-stationary series:

**Proc Arima Data = ts.train;**

**identify var = log\_air stationarity = (ADF);**

**Run;**

**Proc GPLOT Data = ts.train;**

**Plot Log\_Air\*Date;**

**Run;**

**Proc GPLOT Data = ts.train;**

**Plot SQRT\_Air\*Date;**

**Run;**

**Proc Arima Data = ts.train;**

**identify var = sqrt\_air stationarity = (ADF);**

**Run;**

Let’s try taking lagged values (‘first order differences’) of these series, then we now get a mix of transformed-differenced time series, using a Lag(.) function:

**Data ts.train;**

**Set ts.train;**

**Dif\_log\_air = Dif(Log\_Air);**

**Dif\_SQRT\_air = Dif(SQRT\_air);**

**Run;**

And now, of course, run ADF test to see if these series are stationary or not:

**Proc Arima Data = ts.train;**

**identify var = dif\_log\_air stationarity = (adf);**

**Run;**

**Proc Arima Data = ts.train;**

**identify var = dif\_sqrt\_air stationarity = (adf);**

**Run;**

And indeed, both of these series come out to be stationary now. This can be easily verified using a scatter plot.

Now a simple variance check suggests that ‘dif\_log\_air’ has lower variance than ‘dif\_sqrt\_air’:

**Proc Means Data = ts.train mean std;**

**Var dif\_log\_air dif\_sqrt\_air;**

**Run;**

So, I decide to go with ‘dif\_log\_air’ as my stationary time series. This is not a ‘best’ way to transform any series. These techniques perform differently with different series. It is actually more of a hit and trial procedure. For the further analysis, even ‘dif\_sqrt\_air’ may be used, which may yield very similar results (or maybe not!).

Now the trend variations have been removed from the data, and now we need to eliminate seasonal variations, using Autocorrelations plot:

**ODS HTML FILE = 'C:\Documents and Settings\STD\Desktop\SasClasses\Time Series Air\Seasonality check.xls';**

**Proc Arima Data = ts.train;**

**identify var = dif\_log\_air nlag=70;**

**QUIT;**

**ODS HTML CLOSE;**

|  |  |  |  |
| --- | --- | --- | --- |
| **Lag** | **Covariance** | **Correlation** | **-1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1** |
| **0** | 0.011259 | 1 | |                    |\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*| |
| **1** | 0.0021204 | 0.18833 | |                 .  |\*\*\*\*                | |
| **2** | -0.0014316 | -0.12715 | |                .\*\*\*|   .                | |
| **3** | -0.0017371 | -0.15428 | |                .\*\*\*|   .                | |
| **4** | -0.003671 | -0.32605 | |             \*\*\*\*\*\*\*|   .                | |
| **5** | -0.0007441 | -0.06609 | |                .  \*|   .                | |
| **6** | 0.00045622 | 0.04052 | |                .   |\*  .                | |
| **7** | -0.0011062 | -0.09826 | |                . \*\*|   .                | |
| **8** | -0.0038662 | -0.34339 | |             \*\*\*\*\*\*\*|   .                | |
| **9** | -0.0012221 | -0.10855 | |                . \*\*|   .                | |
| **10** | -0.0013476 | -0.11969 | |                . \*\*|   .                | |
| **11** | 0.0022423 | 0.19916 | |               .    |\*\*\*\*.               | |
| **12** | 0.0093768 | 0.83283 | |               .    |\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   | |
| **13** | 0.0022286 | 0.19794 | |              .     |\*\*\*\* .              | |
| **14** | -0.0016113 | -0.14311 | |              .  \*\*\*|     .              | |
| **15** | -0.0012339 | -0.10959 | |              .   \*\*|     .              | |
| **16** | -0.0032466 | -0.28836 | |              \*\*\*\*\*\*|     .              | |
| **17** | -0.0005219 | -0.04636 | |              .    \*|     .              | |
| **18** | 0.00040322 | 0.03581 | |             .      |\*     .             | |
| **19** | -0.0011741 | -0.10428 | |             .    \*\*|      .             | |
| **20** | -0.0035209 | -0.31272 | |             .\*\*\*\*\*\*|      .             | |
| **21** | -0.0011947 | -0.10612 | |             .    \*\*|      .             | |
| **22** | -0.0009539 | -0.08473 | |             .    \*\*|      .             | |
| **23** | 0.0020862 | 0.1853 | |             .      |\*\*\*\*  .             | |
| **24** | 0.0080362 | 0.71376 | |             .      |\*\*\*\*\*\*\*\*\*\*\*\*\*\*      | |

The above table is a part the whole output where lag values go from 0 to 70 (Hence, NLAG=70 in the code). The right-most column represents the Autocorrelation (ACF) graph and the little dots represent the critical limits for each of the spikes (denoted by \*\*\*\*\*\*\*\*\*). A spike is significant if it exceeds the dots on either side.

On careful inspection, we find that there is a significant spike after every 12th data point. This strongly suggests unaccounted seasonal component in the time series. Here, the order of lag is 12.

By differencing this series by 12 lags, we can eliminate these seasonal variations:

**Data ts.train;**

**Set ts.train;**

**Dif12\_dif\_log\_air = Lag12(dif\_log\_air);**

**Run;**

Now, again we check for seasonality:

**Proc Arima Data = ts.train;**

**identify var = dif12\_dif\_log\_air nlag = 70;**

**Run;**

It can now be observed in ACF plot that only the 12th time point has a significant spike; the rest are not significant

which implies that the seasonality has been removed.

Now, the data is ready to be forecasted upon.

This means that we need to determine the order of our model i.e. the values of p and q, to apply ARIMA models. One way to do is by inspecting ACF and PACF plots in SAS.

But in SAS, we follow a more sophisticated procedure to determine these values, using BIC, AIC and SBC.

We start with BIC. The following code gives the values for BIC indicators:

**ODS HTML FILE = 'C:\Documents and Settings\STD\Desktop\SasClasses\Time Series Air\MinIC\_dif12\_dif\_log\_air.xls';**

**Proc Arima Data = ts.train;**

**identify var = dif12\_dif\_log\_air MinIC;**

**QUIT;**

**ODS HTML CLOSE;**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Minimum Information Criterion** | | | | | | |
| **Lags** | **MA 0** | **MA 1** | **MA 2** | **MA 3** | **MA 4** | **MA 5** |
| **AR 0** | -6.24946 | -6.32141 | -6.30667 | -6.32826 | -6.29985 | -6.27302 |
| **AR 1** | -6.33466 | -6.29602 | -6.28026 | -6.29538 | -6.26474 | -6.23915 |
| **AR 2** | -6.32028 | -6.28278 | -6.25595 | -6.25609 | -6.22534 | -6.2063 |
| **AR 3** | -6.3503 | -6.3141 | -6.27621 | -6.24523 | -6.24194 | -6.22247 |
| **AR 4** | -6.33057 | -6.29054 | -6.25187 | -6.25848 | -6.21998 | -6.1958 |
| **AR 5** | -6.30796 | -6.26784 | -6.22782 | -6.23527 | -6.19898 | -6.165 |

Minimum Table Value: BIC(3,0) = -6.3503

In our result, Minimum table value is at BIC(3,0). This is an indicator of the all the appropriate combinations

(p,q) that should be taken. BIC(3,0) implies that we should try all the combinations of (p,q) where p,q = 0,1,2,3.

Of course, we can check for more combinations, but that should be of no use from results point of view.

Now, what follows next is a bit of a SAS macro coding. But first, some background:

After getting these combinations { i.e. (0,0), (0,1),(0,2),…….,(3,2),(3,3) }, we shortlist some combinations based on the arithmetic mean of AIC and SBC, which we obtain using SAS as follows:

**Proc Arima Data = ts.train;**

**identify var = dif12\_dif\_log\_air;**

**Estimate p=0 q=0;**

**Run;**

One way to do it is manually running the codes by changing the values of p and q for all the combinations and this will give us AIC, SBC and hence we can manually calculate the average.

Then, we further shortlist those combinations with the lower averages, since those are the models which will be the more accurate than those with higher averages.

However, this approach is inefficient. So, instead, we will use a macro which will loop the code for different combinations (p,q). This eliminates the need to shortlist on the basis of AIC-SBC average since the task will be automated, provided that the number of combinations is not too large.

The following is the macro code for this task and then the explanation:

**%macro IC\_Forecast\_model\_Selection(number,data,var,date);**

**/\*** *The first line specifies the name of the macro and its parameters:*

*number = the numerical value obtained from BIC table (Here it is 3, highlighted in green)*

*data = the dataset on which we are performing operations (Here, ts.train)*

*var = the time series variable (dif12\_dif\_log\_air)*

*date = the Date variable (DATE)*\*/

**%do p = 0 %to &number;**

**%do q = 0 %to &number;**

**Proc Arima Data = &data;**

**identify var = &var;**

**estimate p=&p. q=&q. outstat = ts.stat\_&p.\_&q.;**

**forecast lead = 12 interval = month ID = &date out = ts.result\_&p.\_&q.;**

**Run;**

**Quit;**

**/\*** *This chunk of code starts with 2 do-loops since we have two numerical values in (p,q):*

1. *‘outstat = <new\_dataset>’ creates a SAS dataset which contains only the important statististics such as AIC, SBC and some more in a tabular manner, which can be further used for numerical computation.*
2. *‘lead = <numeric\_value>’ specifies the number of forecasted estimates (Jan1960-Dec1960).*
3. *‘interval = <unit of time for forecast>’ e.g. ‘interval = month’ for monthly forecast.*
4. *‘ID = <Date variable>*
5. *‘out = <new\_dataset>’ returns the forecasted values against the specified variable into a SAS dataset in tabular form.***\*/**

**Data ts.stat\_&p.\_&q.;**

**Set ts.stat\_&p.\_&q.;**

**p = &p.;**

**q = &q.;**

**Run;**

**/\*** This chunk of code creates 2 separate columns for the values of p and q in each of the outstat datafile. This is particularly useful only if we are interested in the AIC-SBC-average for verification purpose. \*/

**Data ts.result\_&p.\_&q.;**

**Set ts.result\_&p.\_&q.;**

**p=&p.;**

**q=&q.;**

**Run;**

**%end;**

**%end;**

**/\*** This chunk of code does the similar task as above, but in the forecasted result datafiles . This is useful for the next chunk of code.

Here both the loops end.

**\*/**

**Data ts.final\_stats;**

**Set %do p=0 %to &number;**

**%do q=0 %to &number;**

**ts.stat\_&p.\_&q.**

**%end;**

**%end;**

**;**

**Run;**

**/\*** This chunk stacks all the outstat datafiles vertically, along with the values of p and q in separate columns that we created in the second chunk. Since all of the outstat datafiles are structurally identical, the resulting datafile (ts.final\_stats) looks well structured.\*/

**Data ts.final\_result;**

**Set %do p=0 %to &number;**

**%do q=0 %to &number;**

**ts.result\_&p.\_&q.**

**%end;**

**%end;**

**;**

**Run;**

**%mend;**

/\* This chunk does the similar task, but with the forecasted values, resulting into ts.final\_result

Here, the macro ends. Next, we use this macro with appropriate parameters.\*/

**%IC\_Forecast\_model\_selection(3,ts.train,log\_air(1,12),Date);**

**/\****Notice log\_air(1,12) here instead of dif12\_dif\_log\_air. It is because we later have to convert the differenced time series to get the meaningful results, but we have an option to do the task of conversion here by mentioning the differencing as (1,12) where 1 = order of differencning for trend and 12 = order of differencing for seasonality. Finally, log\_air air be easily converted later by taking anti-log\*/*

We will obtain outstat and result files for each of the combination (p,q) as well as 2 more files: final\_stats and final\_result.

Next, if we are interested in identifying the combinations with low AIC-SBC average, consider the following code:

**Proc SQL;**

**Create table ts.final\_stats\_imp as**

**Select p,q,sum(\_value\_)/2 as Average**

**From ts.final\_stats**

**Where \_stat\_ in ('AIC', 'SBC')**

**Group by p,q**

**Order by Average**

**;**

**Quit;**

**/\*** *This will calculate the average of AIC and SBC from ts.final\_stats and arrange the combinations in the ascending order of the average***\*/**

**Data ts.final\_result\_imp;**

**Set ts.final\_result;**

**Forecast = exp(Forecast);**

**If Year(Date)=1960;**

**Keep Date Forecast p q;**

**Run;**

/**\****This will take anti-log of forecast variable, so that we have our forecast in the original form now. Also, we take only the values for the year 1960 for each combination (p,q), to compare them with the actual values of 1960\**/

**Proc SQL;**

**Create table ts.aa as**

**Select f.date, FORECAST, AIR, p, q**

**From ts.final\_result\_imp as f inner join ts.test as t on f.date = t.date;**

**Quit;**

/*\*This will join the forecast values and test data set such that the resulting SAS table has 1960 dates, forecasts, actual values, for each combination (p,q)\**/

**Proc SQL;**

**Create Table ts.mape as**

**Select p,q, Abs(Forecast-Air)/Air as ind\_mape**

**From ts.aa;**

**Quit;**

**/\****This will calculate Abs(Forecast-Air)/Air at each point of time for each combination (p,q)***\*/**

**Proc SQL;**

**Create Table ts.mape\_final as**

**Select p,q, Round(mean(ind\_mape)\*100,0.01) as mape**

**From ts.mape**

**Group by p,q**

**Order by mape;**

**Quit;**

**/\****This will calculate MAPE values for each combination (p,q) and return a SAS file***\*/**

**ODS HTML File = 'C:\Documents and Settings\STD\Desktop\SasClasses\Time Series Air\Mape Values.xls';**

**Proc Print Data = ts.mape\_final noobs;**

**Run;**

**ODS HTML Close;**

**/\****Finally, we obtain the MAPE values for each of the combination in an Excel file.***\*/**

|  |  |  |
| --- | --- | --- |
| **P** | **q** | **mape** |
| 0 | 3 | 6.65 |
| 1 | 3 | 6.83 |
| 2 | 2 | 6.85 |
| 0 | 2 | 7.44 |
| 2 | 3 | 7.57 |
| 0 | 1 | 7.58 |
| 3 | 0 | 7.59 |
| 1 | 2 | 8.09 |
| 3 | 1 | 8.15 |
| 3 | 3 | 8.29 |
| 1 | 0 | 8.3 |
| 3 | 2 | 8.34 |
| 2 | 0 | 8.34 |
| 1 | 1 | 8.68 |
| 2 | 1 | 9.07 |
| 0 | 0 | 9.12 |

Finally, we get (p,q)=(0,3) as our best ARMA model. Here order of differencing for Autoregressive model (AR) is 0 and that for Moving Average model (MA) is 3.

So, this model may be used for forecasts for the given problem.