Australian beer production forecasting

Manish Kumar

8/14/2018

Case Study

Analyze the historical beer production data and use time series forecasting techniques to forecast future beer production.

load packages

```
library(fpp2)
library(astsa)
library(DT)
library(dygraphs)
```

load data

```
beer <- read.csv("data/monthly-beer-production-australia.csv")
head(beer)
```

```
##     Month Monthly.beer.production.in.Australia
## 1 1956-01                                 93.2
## 2 1956-02                                 96.0
## 3 1956-03                                 95.2
## 4 1956-04                                 77.1
## 5 1956-05                                 70.9
## 6 1956-06                                 64.8
```

basic analysis

```
tail(beer)
```

```
##       Month Monthly.beer.production.in.Australia
## 471 1995-03                                  152
## 472 1995-04                                  127
## 473 1995-05                                  151
## 474 1995-06                                  130
## 475 1995-07                                  119
## 476 1995-08                                  153
```

```
summary(beer)
```

```
##      Month     Monthly.beer.production.in.Australia
##  1956-01:  1   Min.   : 64.8
##  1956-02:  1   1st Qu.:112.9
##  1956-03:  1   Median :139.2
```

```
##   1956-04:  1   Mean   :136.4
##   1956-05:  1   3rd Qu.:158.8
##   1956-06:  1   Max.   :217.8
##   (Other):470
```

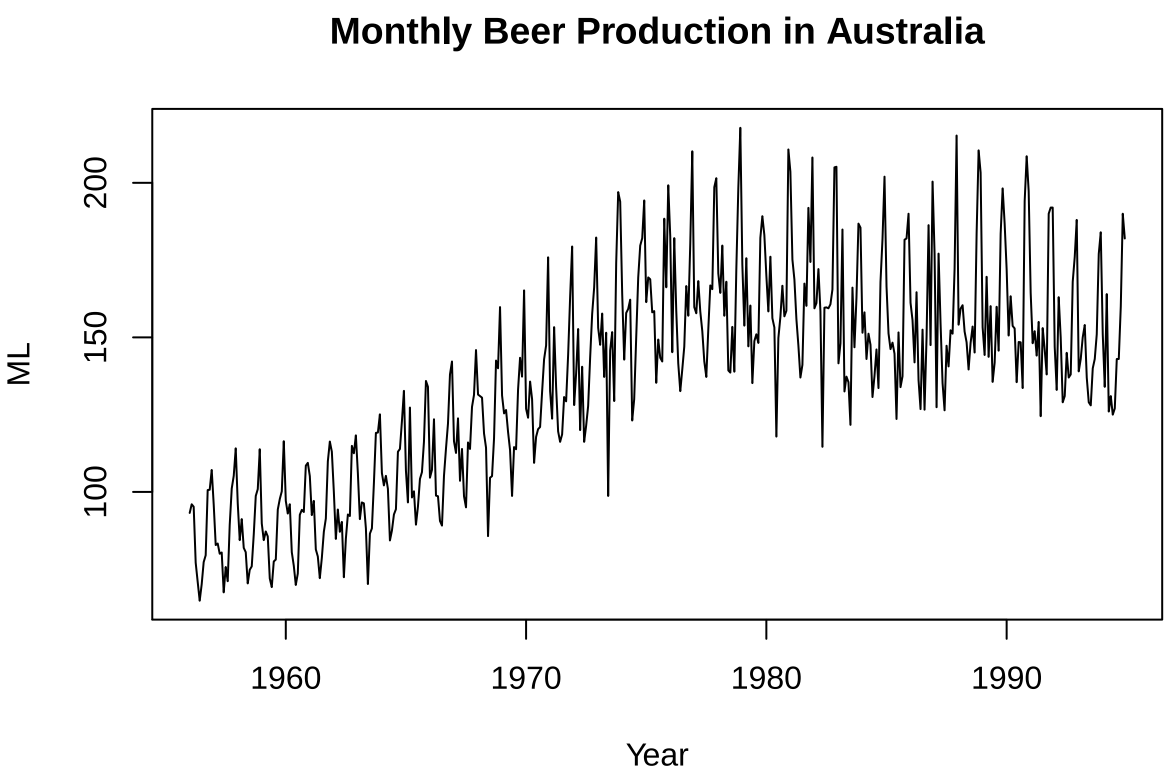
```
beer.ts <- ts(beer, frequency = 12, start = c(1956,1), end = c(1994,12))
```

```
head(beer.ts)
```

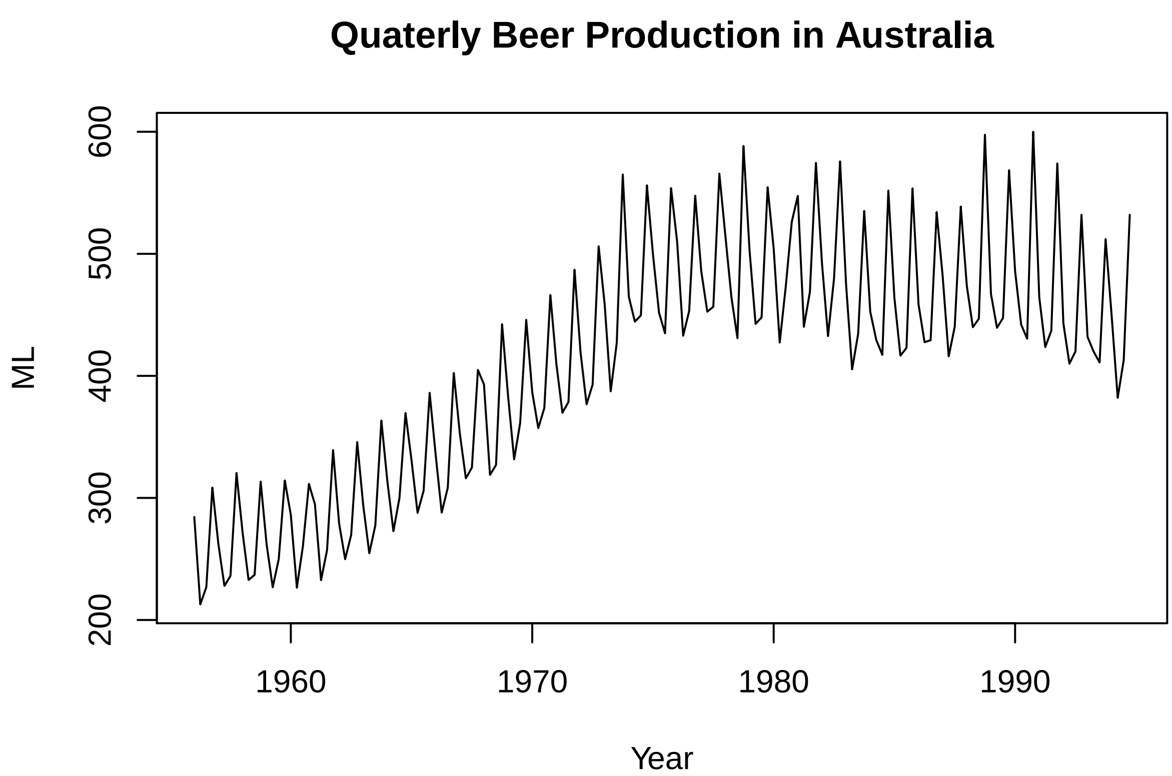
```
##          Month Monthly.beer.production.in.Australia
## Jan 1956     1                                 93.2
## Feb 1956     2                                 96.0
## Mar 1956     3                                 95.2
## Apr 1956     4                                 77.1
## May 1956     5                                 70.9
## Jun 1956     6                                 64.8
```

time series plots

```
plot.ts(beer.ts[,2], main = "Monthly Beer Production in Australia", xlab = "Year", ylab = "ML")
```

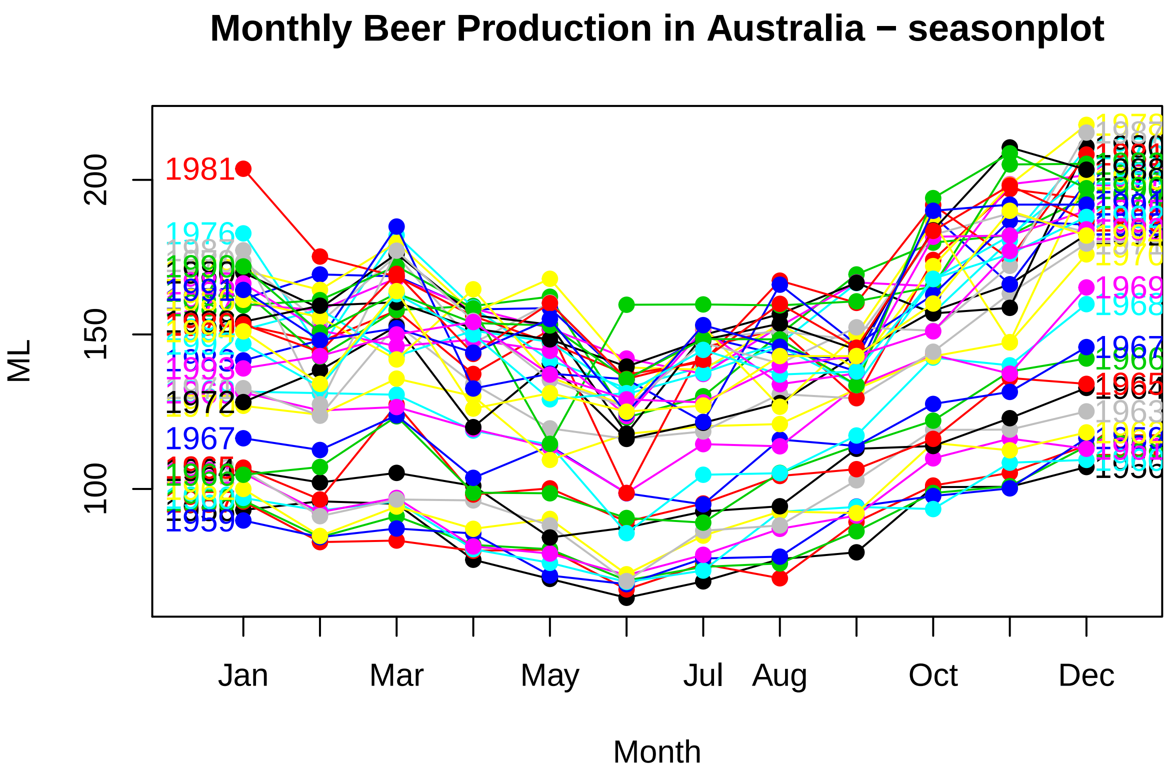



```
# aggregared on quater level
beer.ts.qtr <- aggregate(beer.ts, nfrequency=4)
plot.ts(beer.ts.qtr[,2], main = "Quaterly Beer Production in Australia", xlab = "Year", ylab = "ML")
```

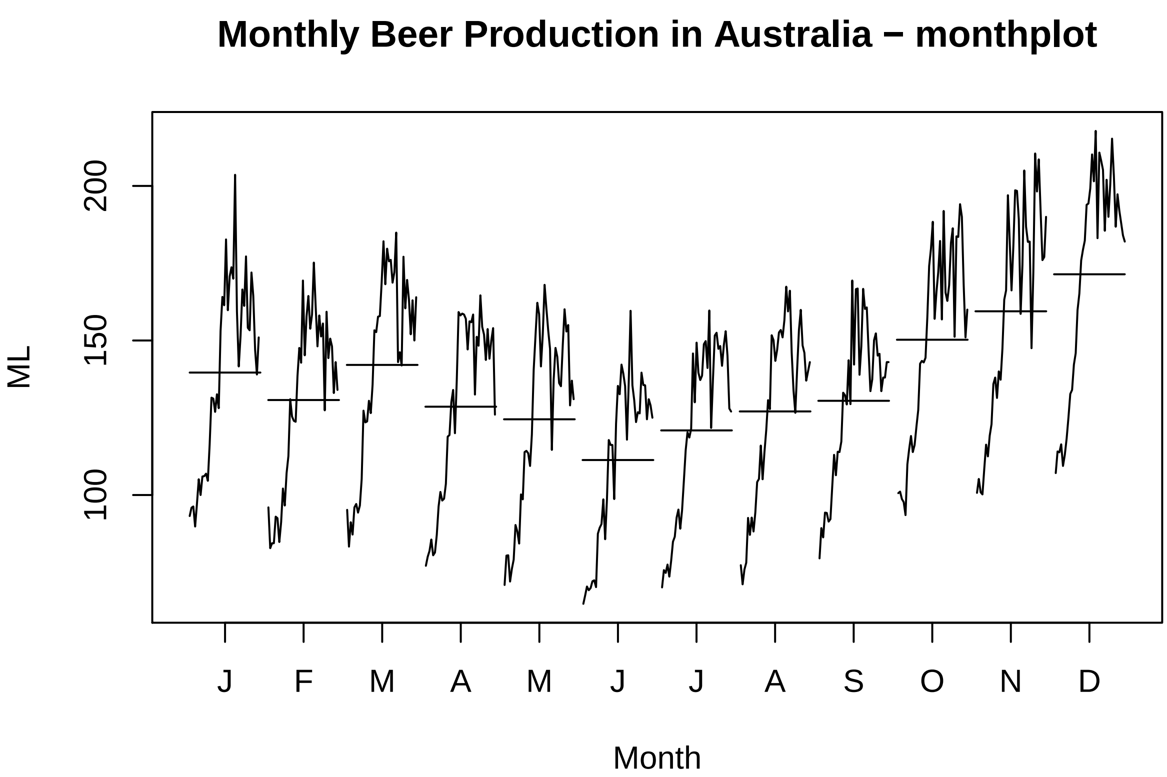



```r
# aggregared on year level
beer.ts.yr <- aggregate(beer.ts, nfrequency=1)
plot.ts(beer.ts.yr[,2], main = "Yearly Beer Production in Australia", xlab = "Year", ylab = "ML")
```

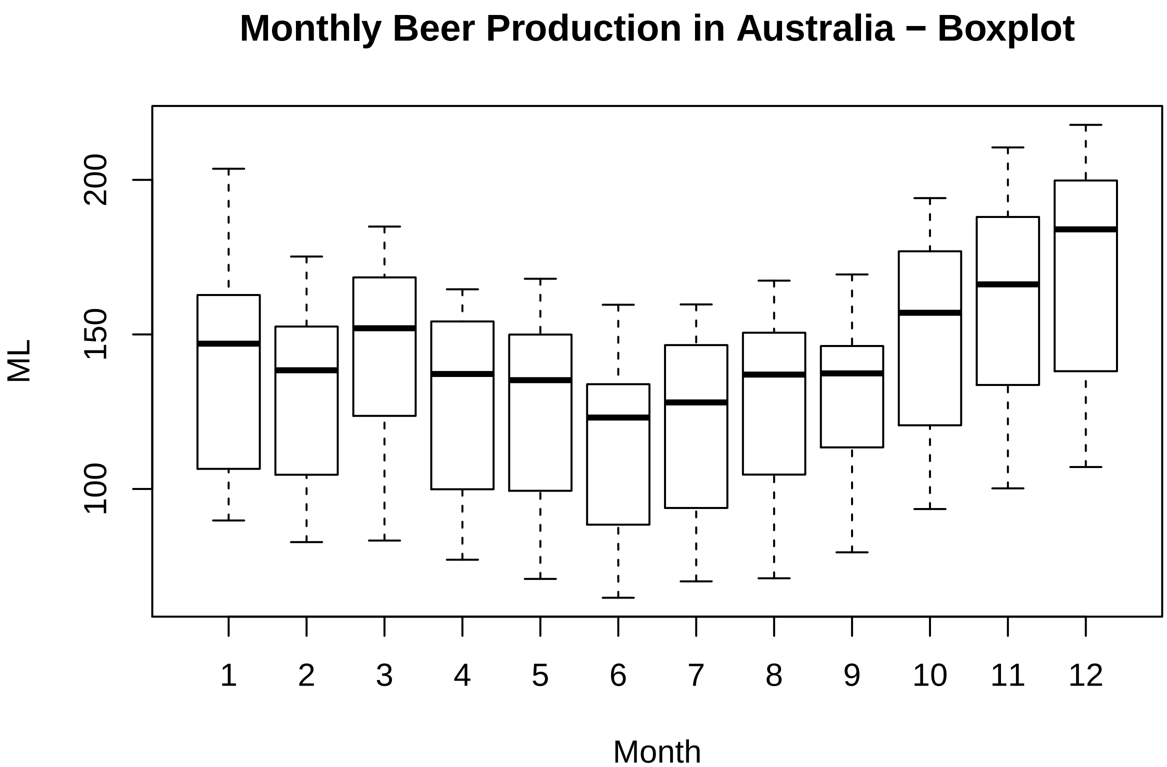
```r
seasonplot(beer.ts[,2], year.labels = TRUE, year.labels.left=TRUE, col=1:40, pch=19, main = "Monthly Bee
```



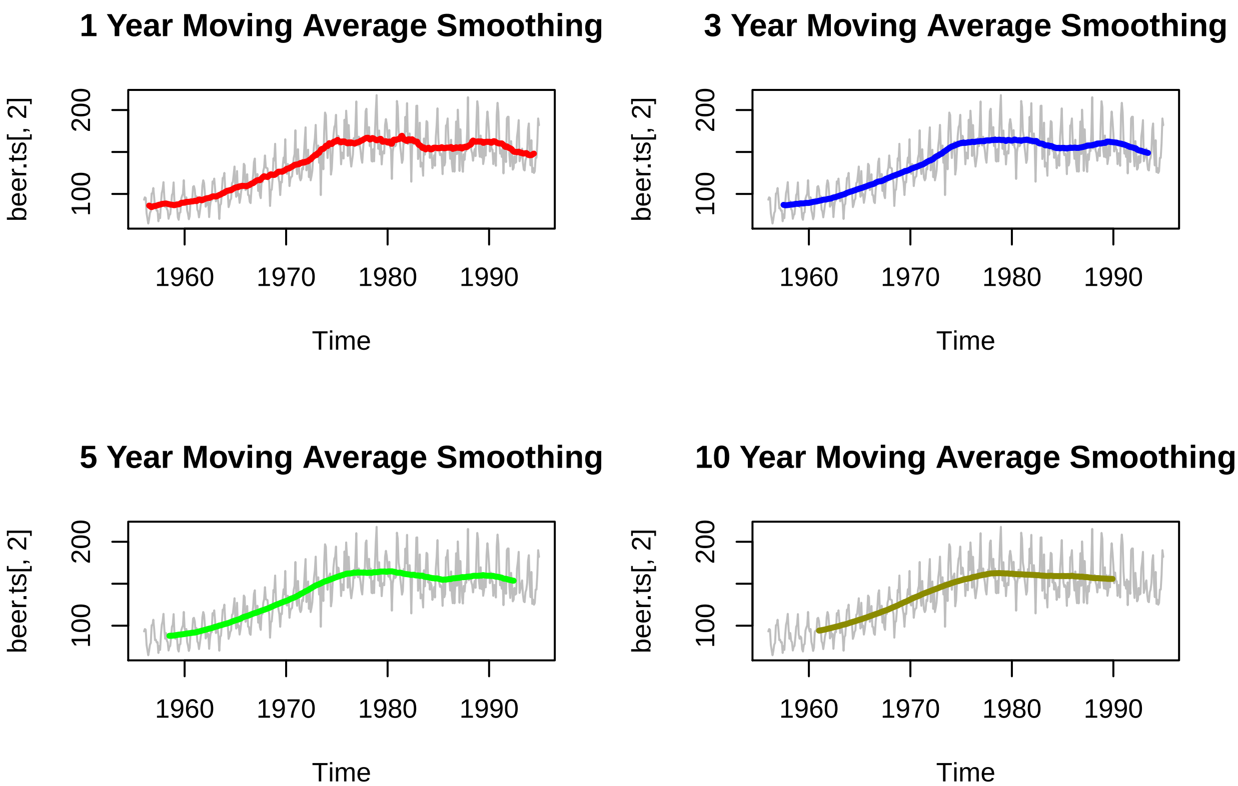

```r
# monthly plot aggreated the data of all year for montly analysis
# each month plots show the variation for entire data in each month
monthplot(beer.ts[,2], main = "Monthly Beer Production in Australia - monthplot", xlab = "Month", ylab =
```



```r
boxplot(beer.ts[,2] ~ cycle(beer.ts[,2]), xlab = "Month", ylab = "ML", main = "Monthly Beer Production i
```




```r
# moving average is useful when we need to analyse trend for the underlying data
# here, we see the moving average for 1 year, 3 year 5 year and 10 year
# Note : If there is not trend, average is good enough for the analysis
par(mfrow = c(2,2))
plot(beer.ts[,2], col="gray", main = "1 Year Moving Average Smoothing")
lines(ma(beer.ts[,2], order = 12), col = "red", lwd=3)
plot(beer.ts[,2], col="gray", main = "3 Year Moving Average Smoothing")
lines(ma(beer.ts[,2], order = 36), col = "blue", lwd=3)
plot(beer.ts[,2], col="gray", main = "5 Year Moving Average Smoothing")
lines(ma(beer.ts[,2], order = 60), col = "green", lwd=3)
plot(beer.ts[,2], col="gray", main = "10 Year Moving Average Smoothing")
lines(ma(beer.ts[,2], order = 120), col = "yellow4", lwd=3)
```



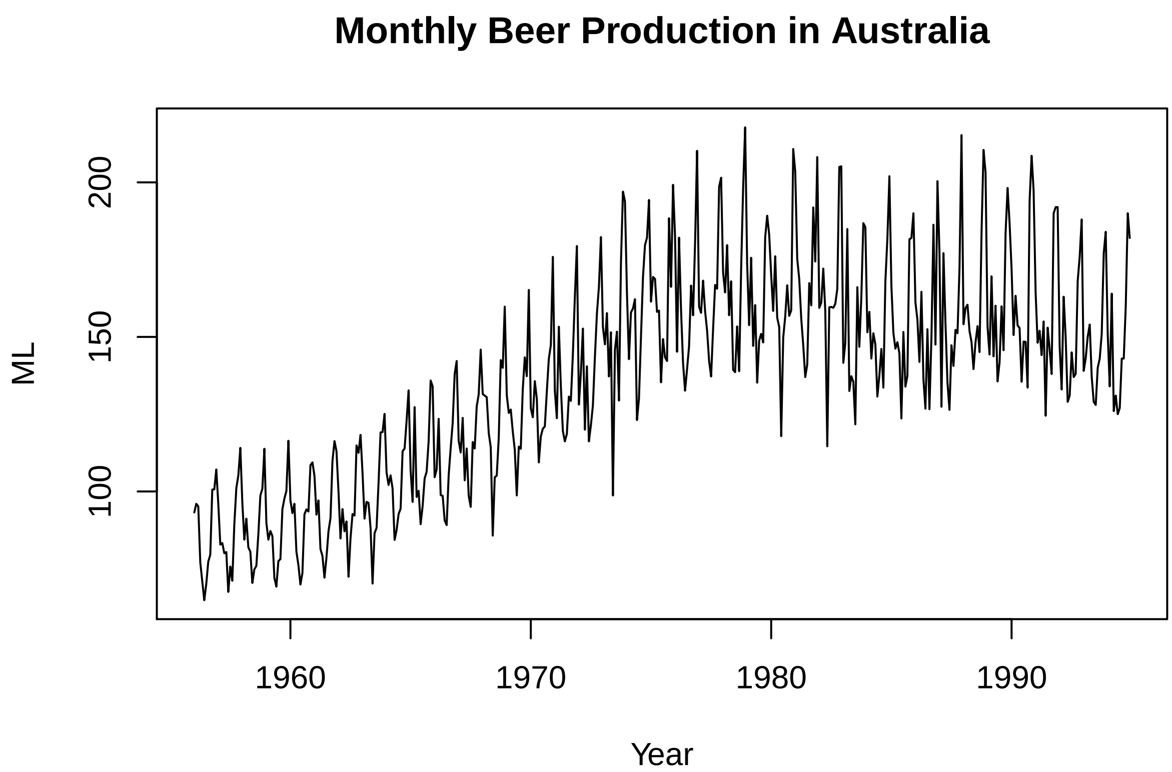
data transformations and adjustments

```
# adjusting and transformating data can makes the historical data less complex so a simpler
# forecast model can be used. It is also a good idea to remove the underline factors affected
# the time series like workday, inflation, population, and currancy exchange rate etc.
```

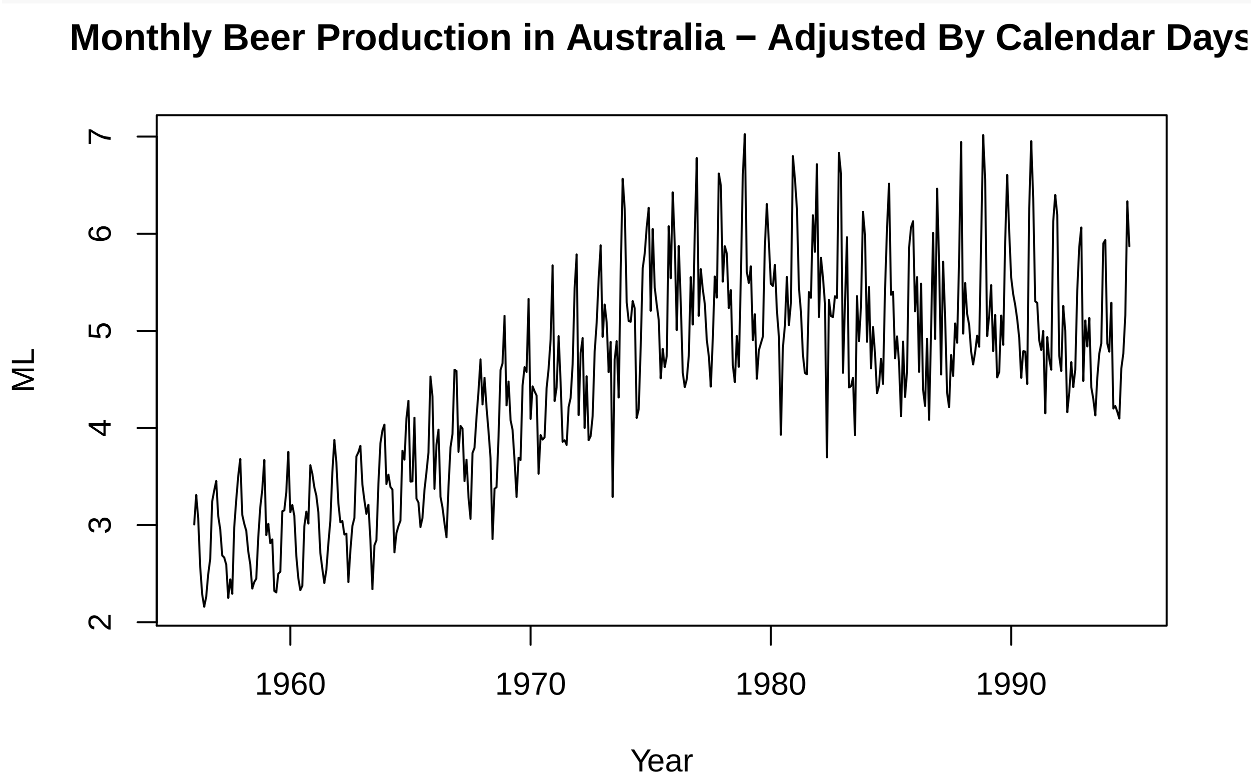
calendar adjustments

```
# not all month have same days, so this variation should be removed for analysis,
# also, so we will look at monthly sales dividing the number of working days in a month
# for better analysis
```

```
plot.ts(beer.ts[,2], main = "Monthly Beer Production in Australia", xlab = "Year", ylab = "ML")
```




```
plot.ts(beer.ts[,2]/monthdays(beer.ts[,2]), main = "Monthly Beer Production in Australia - Adjusted By C
```



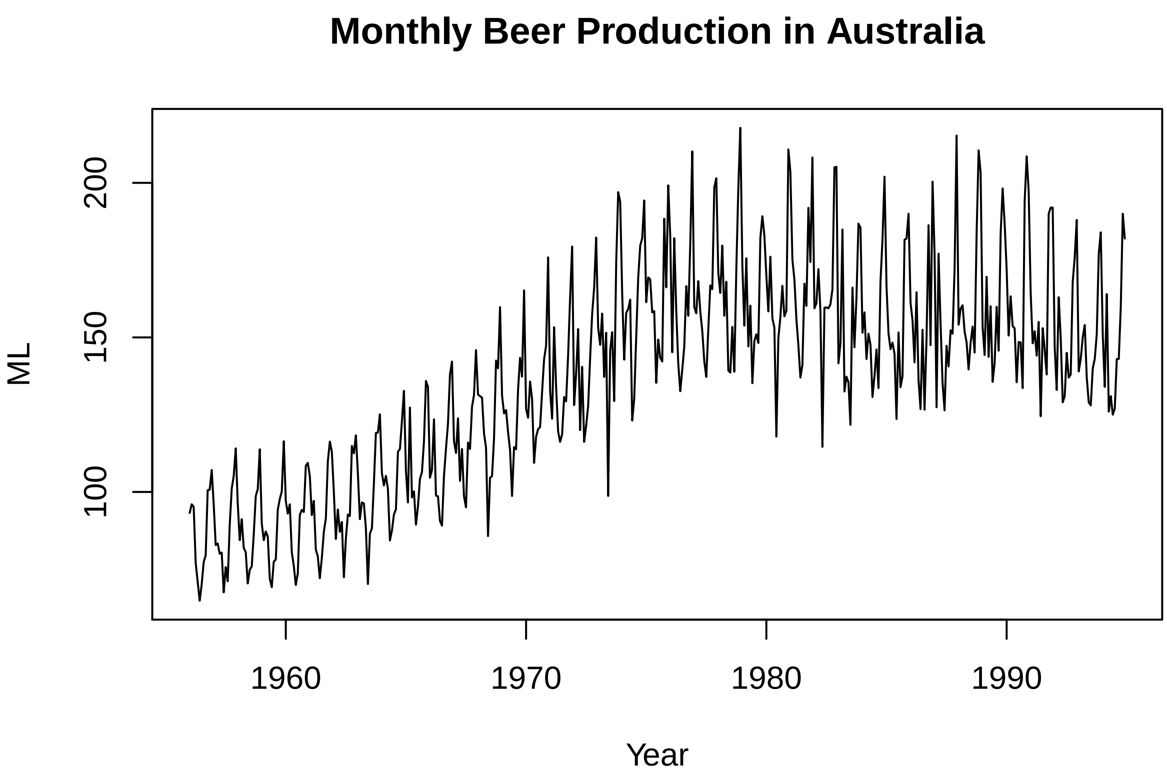
population adjustment

```
# The beer production has been increased since last 50 years but
# one thing to notice is that the population has also been increased in these year.
# So, we need to analyse the data removing this factor.
# Note: australian_population data is not available as of now, so we will not
# do this adjustment but it needs to be taken care depeding on use-case.
```

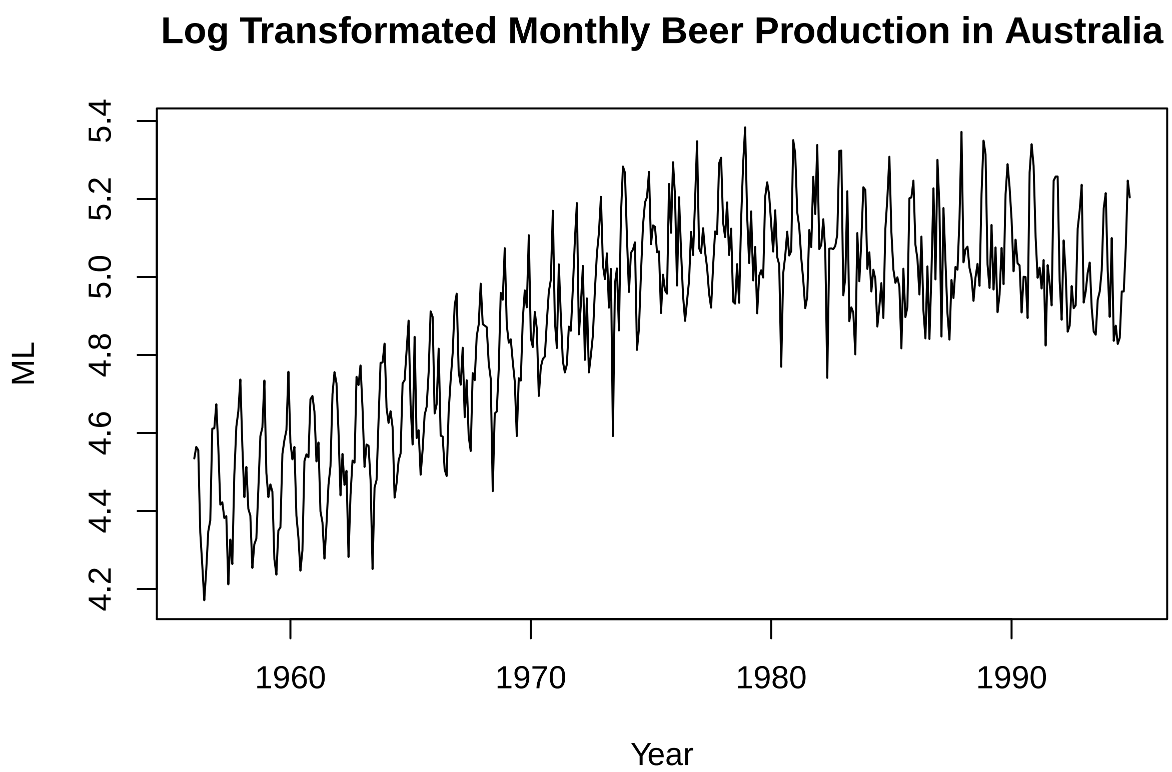
logarithmic transformation

```
# Transform exponential trend line into a linear line using such transformations.
# It is used before differencing data to improve stationarity.
```

```
plot.ts(beer.ts[,2], main = "Monthly Beer Production in Australia", xlab = "Year", ylab = "ML")
```



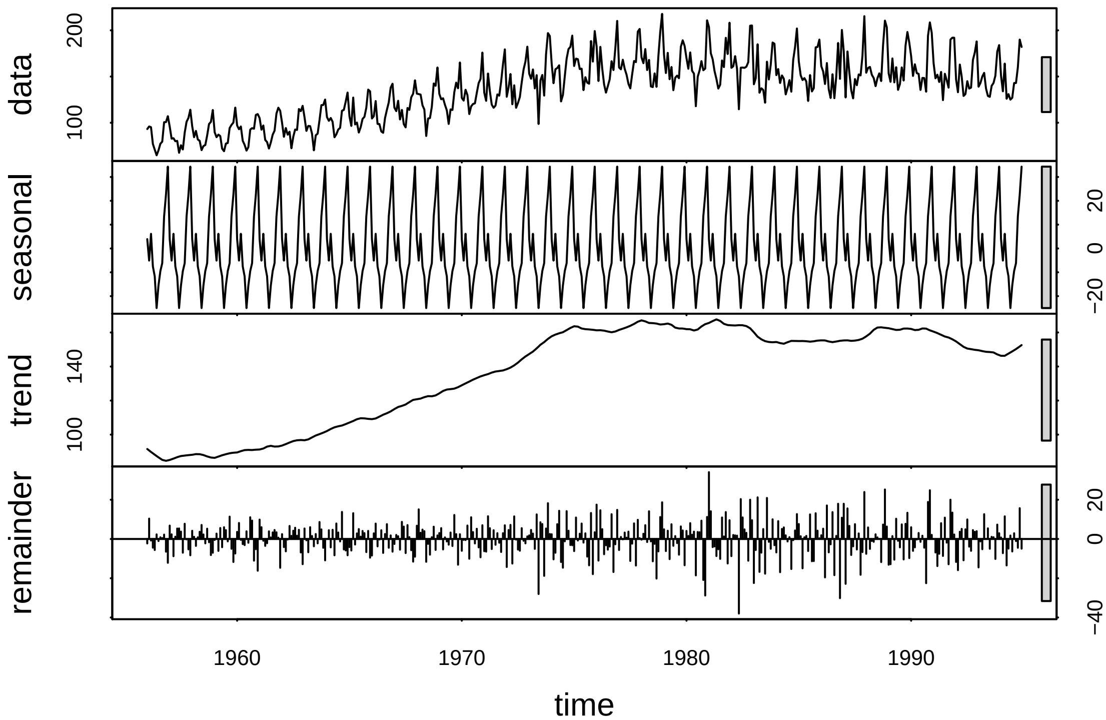

```
plot.ts(log(beer.ts[,2]), main = "Log Transformated Monthly Beer Production in Australia", xlab = "Year
```



decomposition

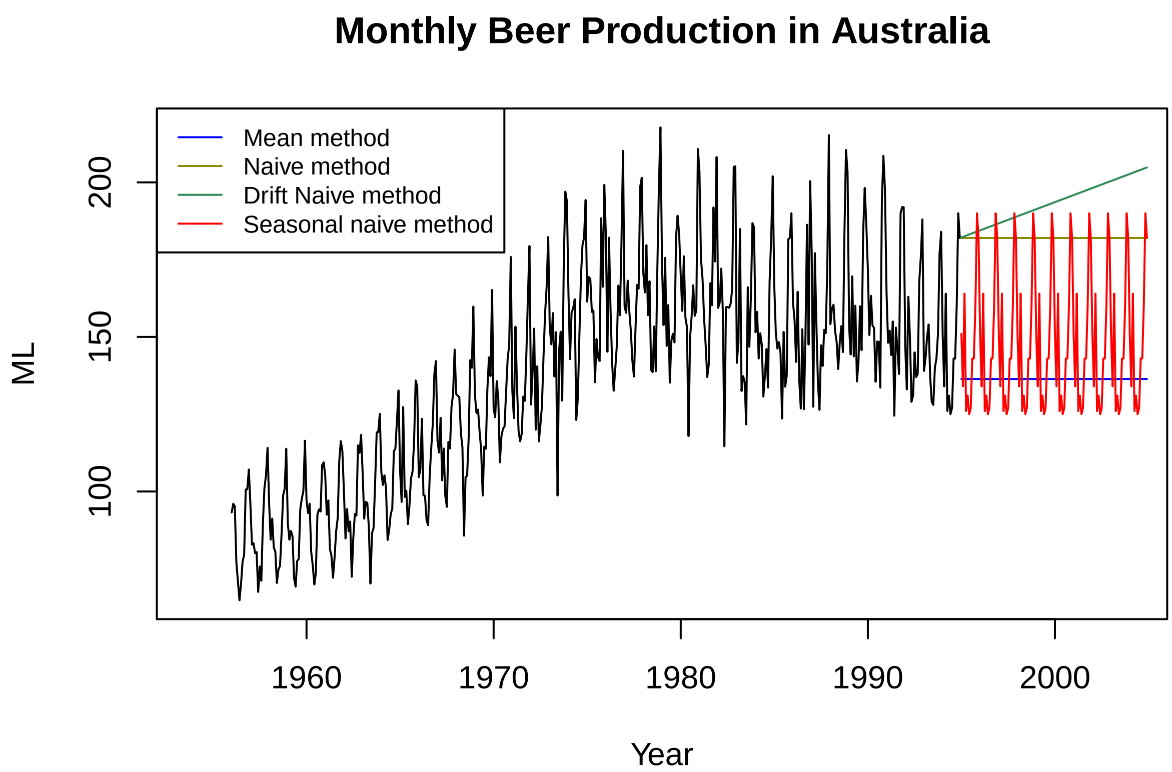
```
# Decomposition is a tool that we can seperate different components in a time series data so we can see
# seasonility, and random noises individually
```

```
plot(stl(beer.ts[,2], s.window="periodic"))
```



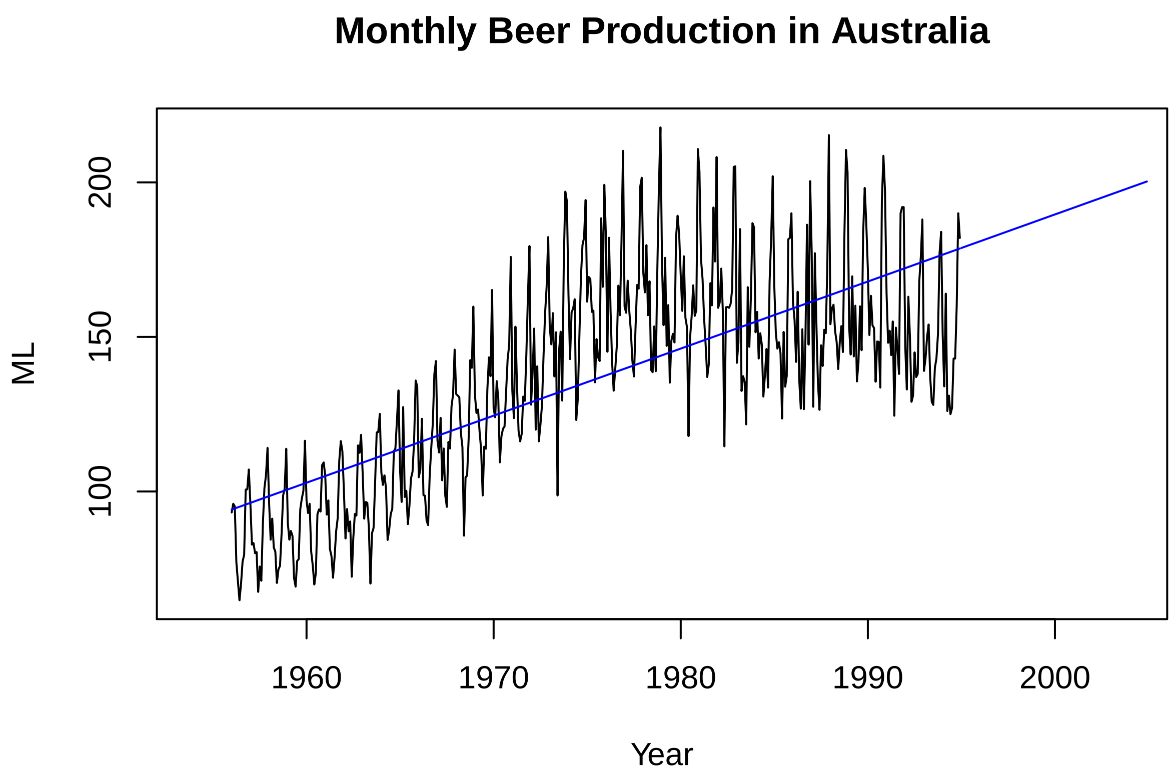
basic forecasting techniques (SNSD)

```
# Simple Average - simple average of all data points
# Naive Method - the last observation value
# Seasonal Navie - the last observation value from previous seasonal cycle
# Drift Method - forecast value increase or decrease over time based on average change in historical da

# Note: These models are basic model to forecasting data with no complex trends or seasonal behaviour
# but they can be useful as the benchmarking models for reference.

beer.fit.a <- meanf(beer.ts[,2], h = 120)
beer.fit.n <- naive(beer.ts[,2], h = 120)
beer.fit.sn <- snaive(beer.ts[,2], h = 120)
beer.fit.dri <- rwf(beer.ts[,2], h = 120, drift = TRUE)
plot.ts(beer.ts[,2], main = "Monthly Beer Production in Australia", xlab = "Year", ylab = "ML", xlim =
lines(beer.fit.a$mean, col = "blue")
lines(beer.fit.n$mean, col = "yellow4")
lines(beer.fit.dri$mean, col = "seagreen4")
lines(beer.fit.sn$mean, col = "red")
legend("topleft",lty=1,col=c("blue","yellow4","seagreen4", "red"), cex = 0.75,
       legend=c("Mean method","Naive method","Drift Naive method", "Seasonal naive method"))
```

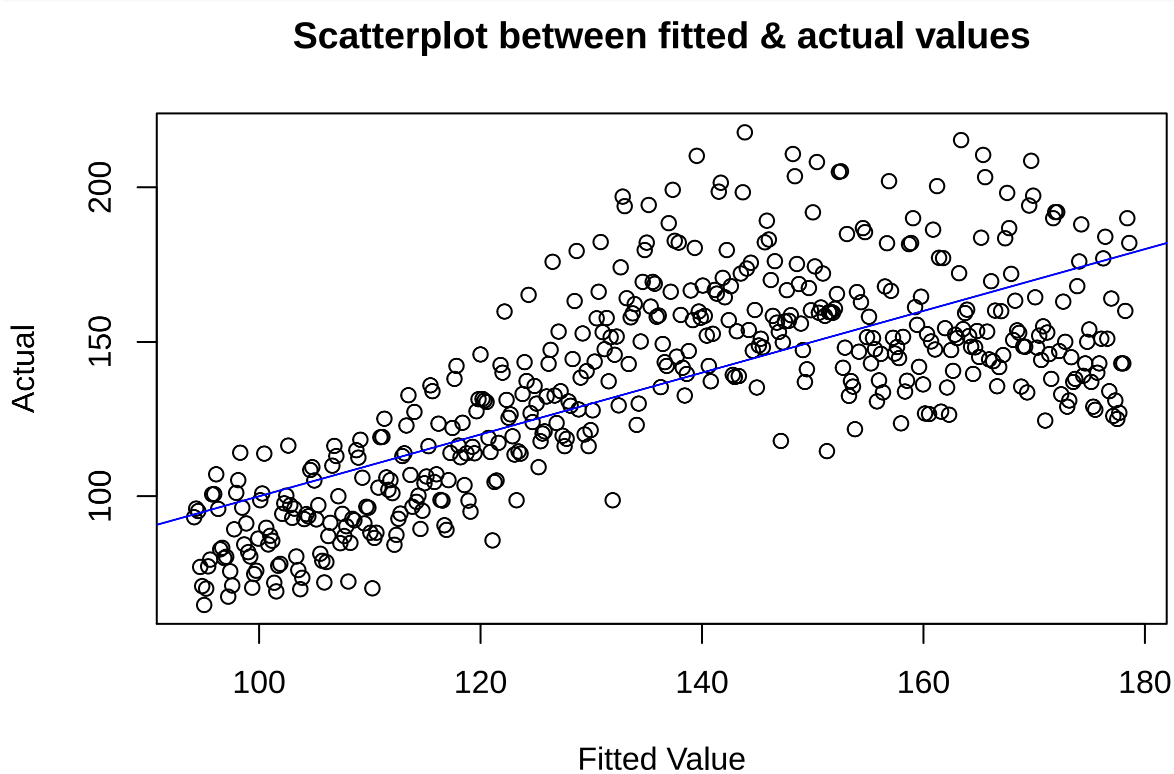


simple regression analysis

```
# simple regression model for trend analysis

beer.fit.lm <- tslm(beer.ts[,2] ~ trend)
f <- forecast(beer.fit.lm, h = 120, level = c(80,95))
plot.ts(beer.ts[,2], main = "Monthly Beer Production in Australia", xlab = "Year", ylab = "ML", xlim = c
lines(f$fitted, col = "blue")
lines(f$mean, col = "blue")
```




```
# simple regression model for trend analysis
# fitted versus actul plot

plot(beer.fit.lm$fitted, beer.ts[,2], main = "Scatterplot between fitted & actual values", xlab = "Fitt
abline(0, 1, col="blue")
```



```r
# simple regression model for trend analysis
# model summary

summary(beer.fit.lm)
```

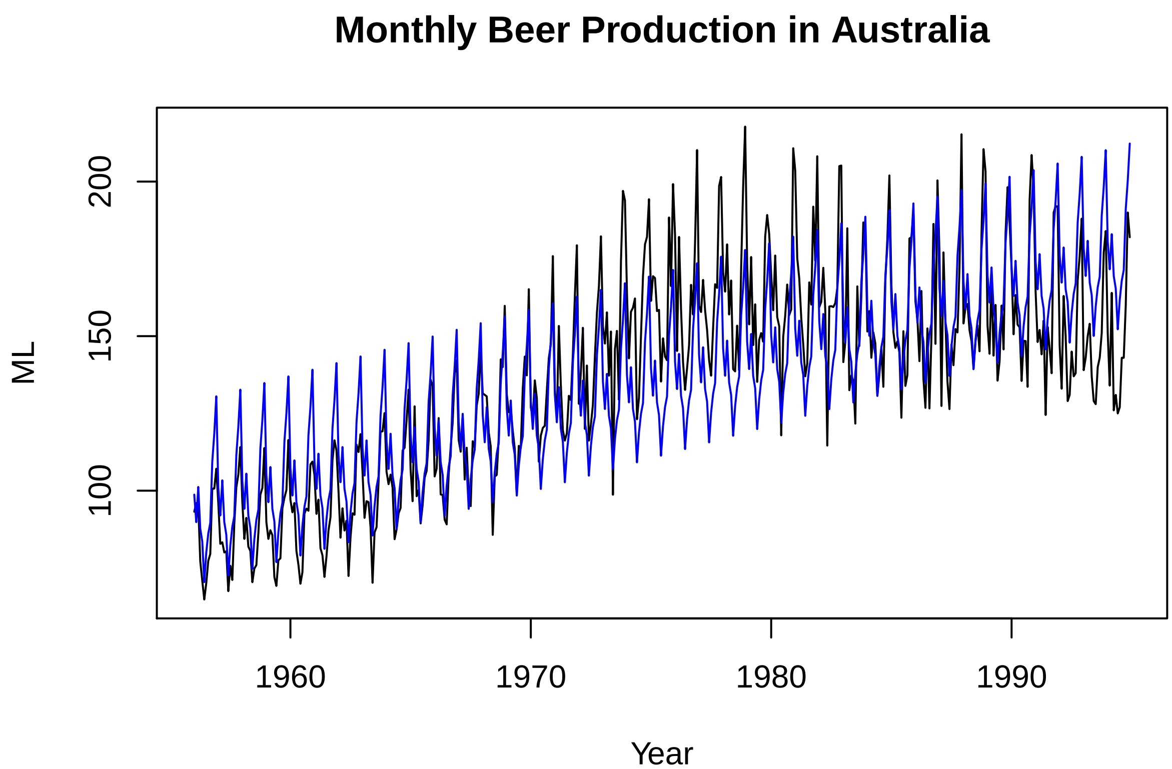
```
##
## Call:
## tslm(formula = beer.ts[, 2] ~ trend)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -52.50 -16.92  -3.28  13.77  73.93
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 93.957034   2.188016   42.94   <2e-16 ***
## trend        0.180839   0.008085   22.37   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.63 on 466 degrees of freedom
## Multiple R-squared:  0.5178, Adjusted R-squared:  0.5167
## F-statistic: 500.3 on 1 and 466 DF,  p-value: < 2.2e-16
```

```r
# simple regression model for (trend + seasonal) analysis
# model summary

beer.fit.lm2 <- tslm(beer.ts[,2] ~ trend + season)
summary(beer.fit.lm2)
```

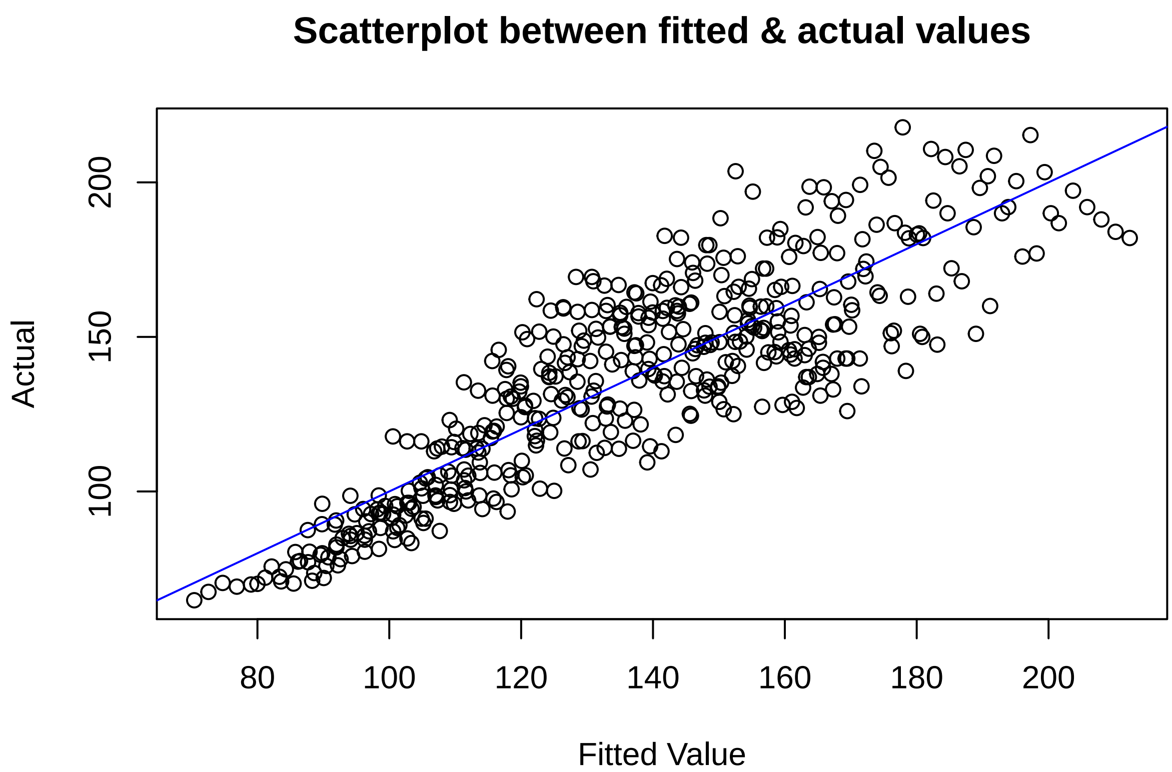
```
##
## Call:
## tslm(formula = beer.ts[, 2] ~ trend + season)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -43.472 -12.449  -2.105  12.632  51.070
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  98.538349   3.119771  31.585  < 2e-16 ***
## trend         0.179376   0.005989  29.949  < 2e-16 ***
## season2      -9.066555   3.962776  -2.288 0.022599 *
## season3       2.120736   3.962790   0.535 0.592799
## season4     -11.579153   3.962812  -2.922 0.003652 **
## season5     -15.835452   3.962844  -3.996 7.51e-05 ***
## season6     -29.196879   3.962885  -7.368 8.24e-13 ***
## season7     -19.786511   3.962934  -4.993 8.49e-07 ***
## season8     -13.822297   3.962993  -3.488 0.000534 ***
## season9     -10.573467   3.963061  -2.668 0.007903 **
## season10      9.011260   3.963138   2.274 0.023445 *
## season11     18.037012   3.963224   4.551 6.86e-06 ***
## season12     29.831995   3.963319   7.527 2.81e-13 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.5 on 455 degrees of freedom
## Multiple R-squared:  0.7418, Adjusted R-squared:  0.7349
## F-statistic: 108.9 on 12 and 455 DF,  p-value: < 2.2e-16
```

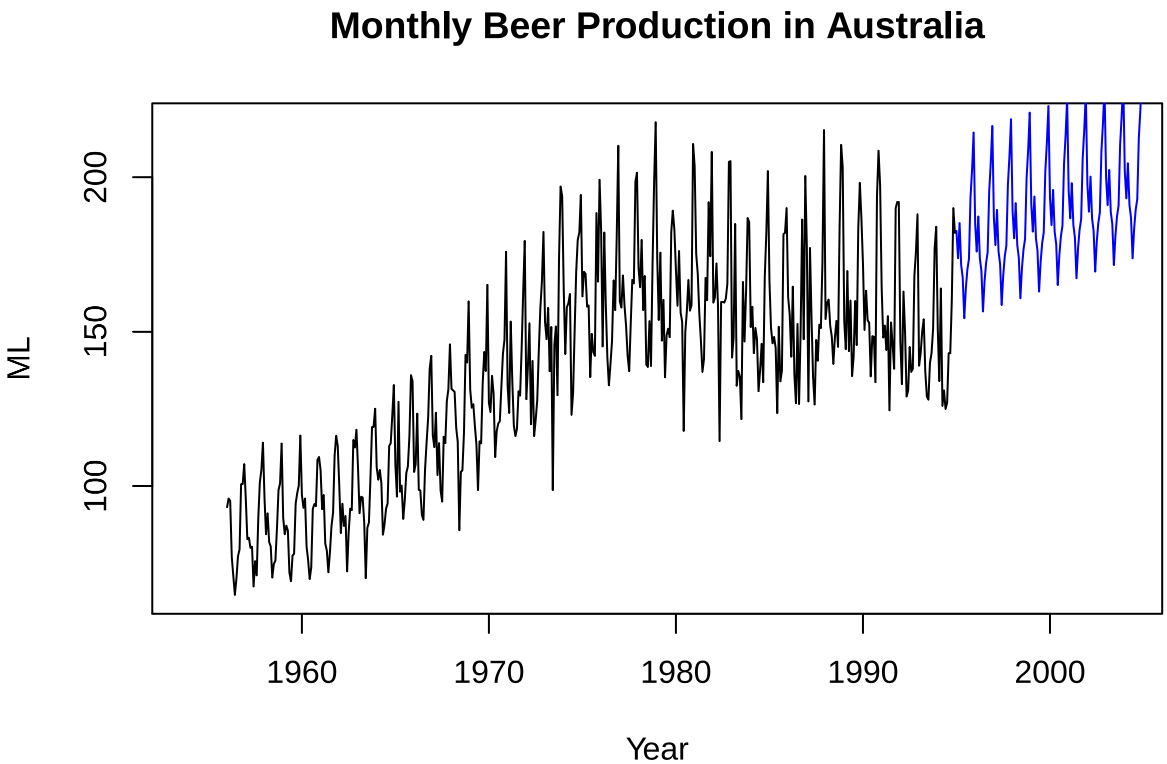
```
# simple regression model for (trend + seasonal) analysis
# fitted values

plot.ts(beer.ts[,2], main = "Monthly Beer Production in Australia", xlab = "Year", ylab = "ML")
lines(beer.fit.lm2$fitted.values, col = "blue")
```




```
# simple regression model for (trend + seasonal) analysis
# scatter plot

plot(beer.fit.lm2$fitted, beer.ts[,2], main = "Scatterplot between fitted & actual values", xlab = "Fit
abline(0, 1, col="blue")
```

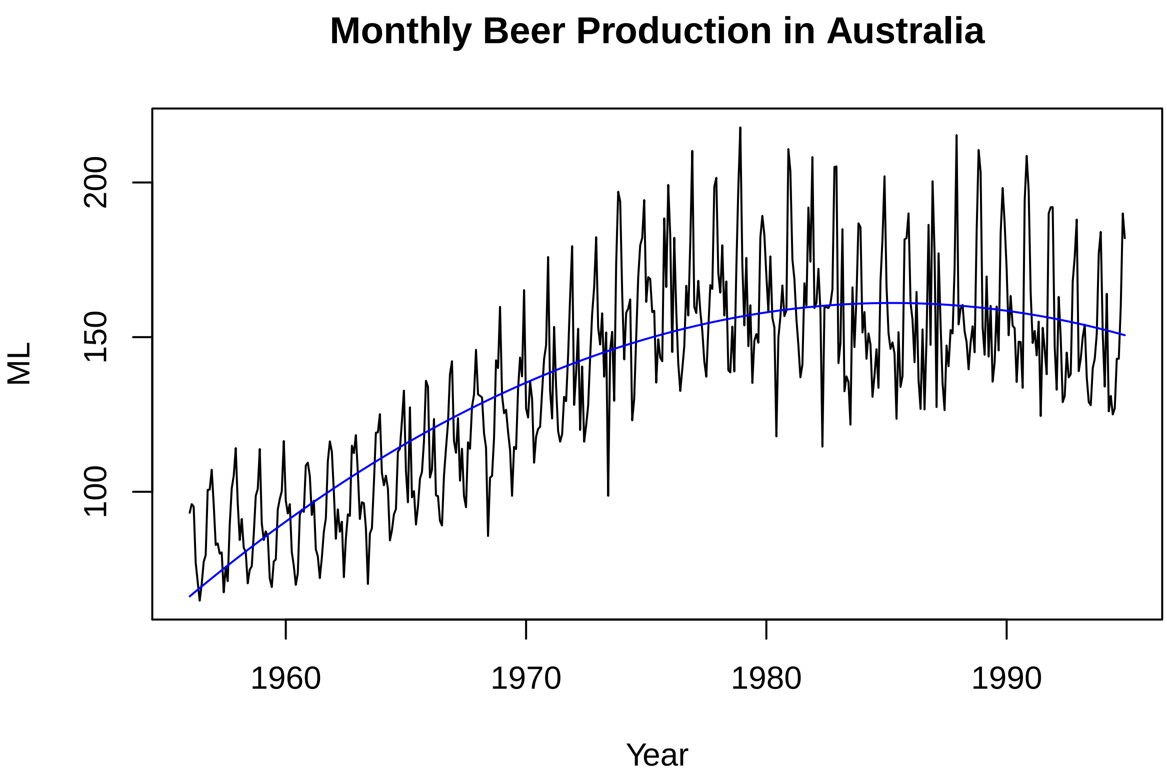



```
# simple regression model for (trend + seasonal) analysis
# forecast

f <- forecast(beer.fit.lm2, h = 120, level = c(80,95))
plot.ts(beer.ts[,2], main = "Monthly Beer Production in Australia", xlab = "Year", ylab = "ML", xlim = c
lines(f$mean, col = "blue")
```

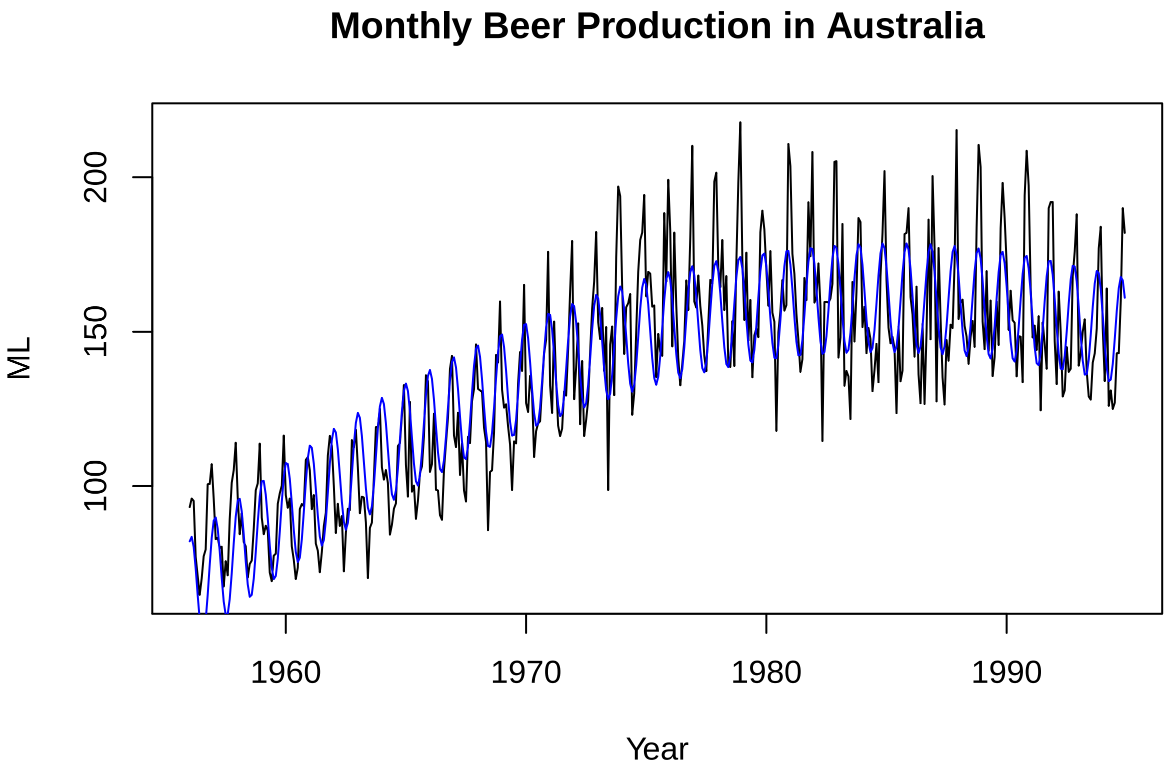


polynomial regression analysis

```r
# the trend is not linear so lets try polynomial regression
# power 2

t <- seq(1956, 1995.2, length = length(beer.ts[,2]))
t2 <- t^2
beer.fit.lm3 <- tslm(beer.ts[,2] ~ t + t2)
plot.ts(beer.ts[,2], main = "Monthly Beer Production in Australia", xlab = "Year", ylab = "ML")
lines(beer.fit.lm3$fit, col = "blue")
```




```
# the trend is not linear so lets try polynomial regression
# sin and cosine

sin.t <- sin(2*pi*t)
cos.t <- cos(2*pi*t)
beer.fit.lm4 <- tslm(beer.ts[,2] ~ t + t2 + sin.t + cos.t)
plot.ts(beer.ts[,2], main = "Monthly Beer Production in Australia", xlab = "Year", ylab = "ML")
lines(beer.fit.lm4$fit, col = "blue")
```



Monthly Beer Production in Australia
ML
200
150
100
1960
1970
1980
1990
Year