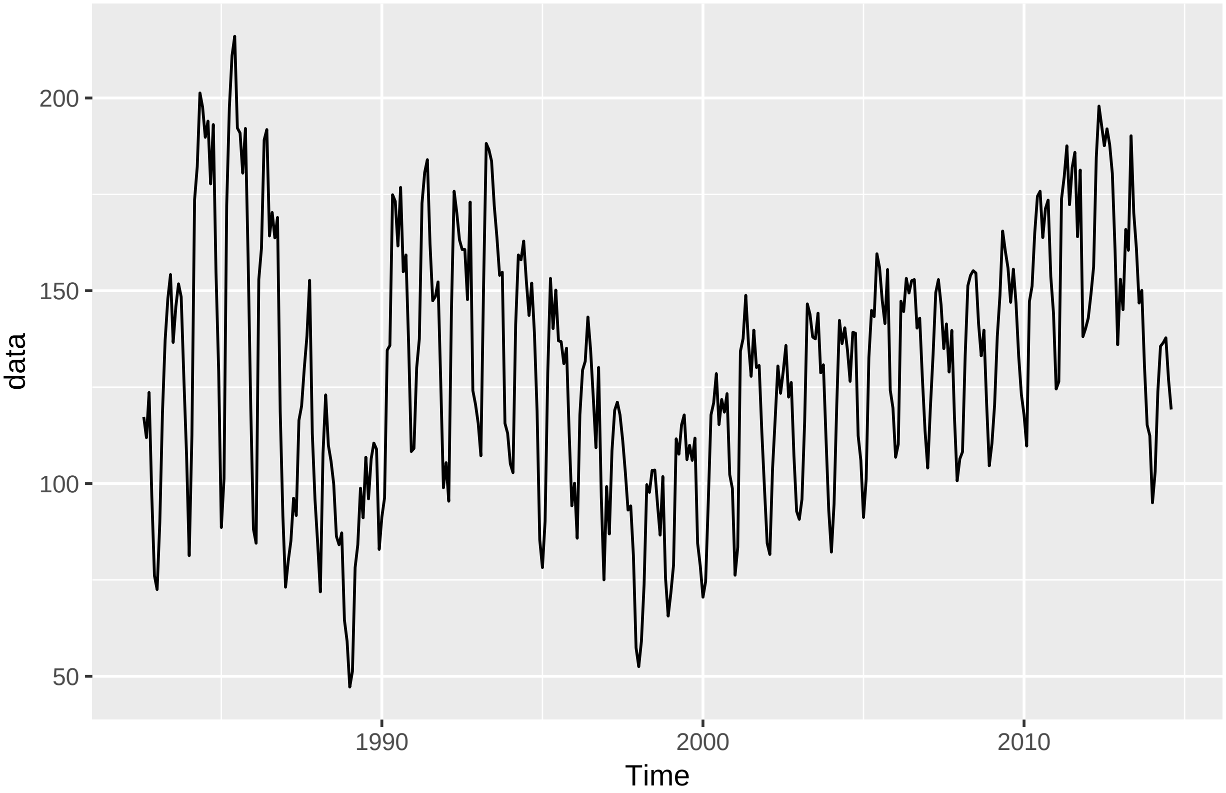
Regression Model

```
library(forecast)
library(ggplot2)

df = read.csv("data.csv")   # PrivateHousingStarts
data = ts(df$Private.Housing.Starts, start = c(1982,08), frequency = 12)
autoplot(data)
```

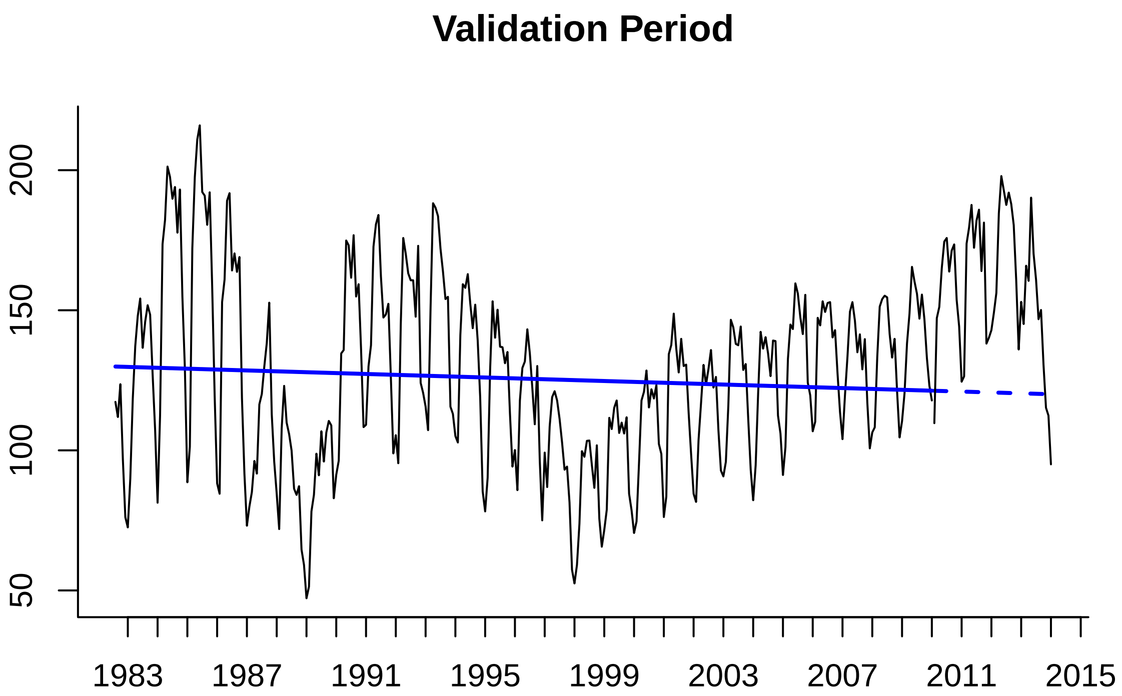


1. Trend

1.1 Linear Trend

```
nvalid = 48
ntrain = length(data) - nvalid
train.ts = window(data, start = c(1982,08), end = c(1982, ntrain))
valid.ts = window(data, start = c(1982, ntrain+1), end = c(1982, ntrain+nvalid))
train.lm = tslm(train.ts~trend)
```

```
train.lm.pred = forecast(train.lm, h = nvalid, level = 0)   # level is the confidence level
plot(train.lm.pred, bty = "l", xaxt = "n", main = "Validation Period", flty = 2)
lines(train.lm.pred$fitted, lwd = 2, col = "blue")
lines(valid.ts)
axis(1,at = seq(1983, 2015,1), labels = format(seq(1983, 2015,1)))
```



The summary of the model:

```
summary(train.lm)
```

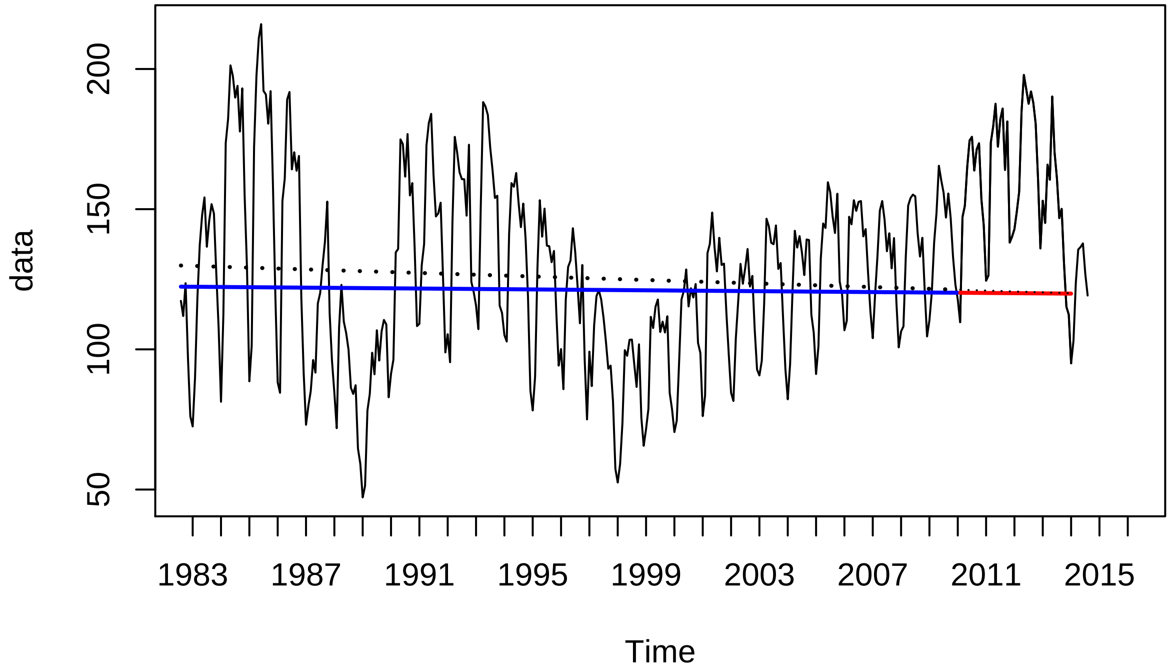
```
##
## Call:
## tslm(formula = train.ts ~ trend)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -80.688 -22.225  -0.726  24.181  86.986
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 129.93045    3.55154  36.584   <2e-16 ***
## trend        -0.02619    0.01860  -1.408     0.16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32.19 on 328 degrees of freedom
## Multiple R-squared:  0.006009,   Adjusted R-squared:  0.002979
## F-statistic: 1.983 on 1 and 328 DF,  p-value: 0.16
```

Slope coefficient is insignificant, however, this does not mean that there is no trend, have a look of the deseasonalized data.

1.2 Exponantial Trend

```
train.lm.expo.trend = tslm(train.ts ~trend, lambda = 0)
# lambda = 0 applies the Box-Cox transformation in the training period, if lambda = 0, y --> log(y)
train.lm.expo.trend.pred = forecast(train.lm.expo.trend, h = nvalid, level = 0)
train.lm.linear.trend = tslm(train.ts~trend)
train.lm.linear.trend.pred = forecast(train.lm.linear.trend, h = nvalid, level = 0)

plot(data, xaxt = "n", main = "", xlim = c(1983, 2016))
axis(1,at = seq(1983, 2016,1), labels = format(seq(1983, 2016,1)))
lines(valid.ts)
lines(train.lm.expo.trend.pred$fitted, lwd = 2, col = "blue" )
lines(train.lm.expo.trend.pred$mean, lwd = 2, col = "red")
lines(train.lm.linear.trend.pred$fitted, lwd = 2, col = "black", lty = 3)
lines(train.lm.linear.trend.pred$mean, col = "black", lty = 3)
```



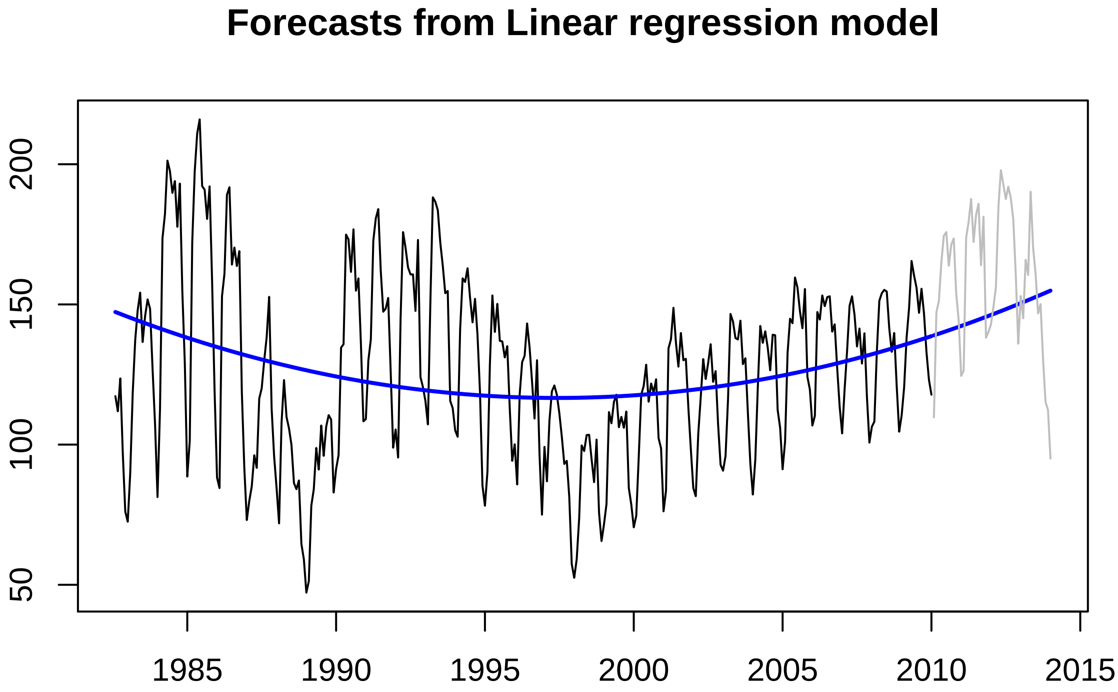
```
# red is exponential trend, black is linear trend, because that time horizon is short and change is gen
```

1.3 Polynomial Trend

```
train.lm.poly.trend = tslm(train.ts ~ trend + I(trend^2))
summary(train.lm.poly.trend)
```

```
##
## Call:
## tslm(formula = train.ts ~ trend + I(trend^2))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -79.316 -22.179   0.984  20.841  79.294
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.476e+02  5.193e+00  28.430  < 2e-16 ***
## trend       -3.462e-01  7.245e-02  -4.779 2.67e-06 ***
## I(trend^2)   9.669e-04  2.120e-04   4.561 7.20e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31.26 on 327 degrees of freedom
## Multiple R-squared:  0.06547,    Adjusted R-squared:  0.05975
## F-statistic: 11.45 on 2 and 327 DF,  p-value: 1.556e-05
```

```
train.lm.poly.trend.pred = forecast(train.lm.poly.trend, h = nvalid, level = 0)
plot(train.lm.poly.trend.pred)
lines(train.lm.poly.trend.pred$fitted, lwd = 2,col= "blue")
lines(valid.ts, col = "grey")
```



2. Seasonal

2.1 Additive Seasonality

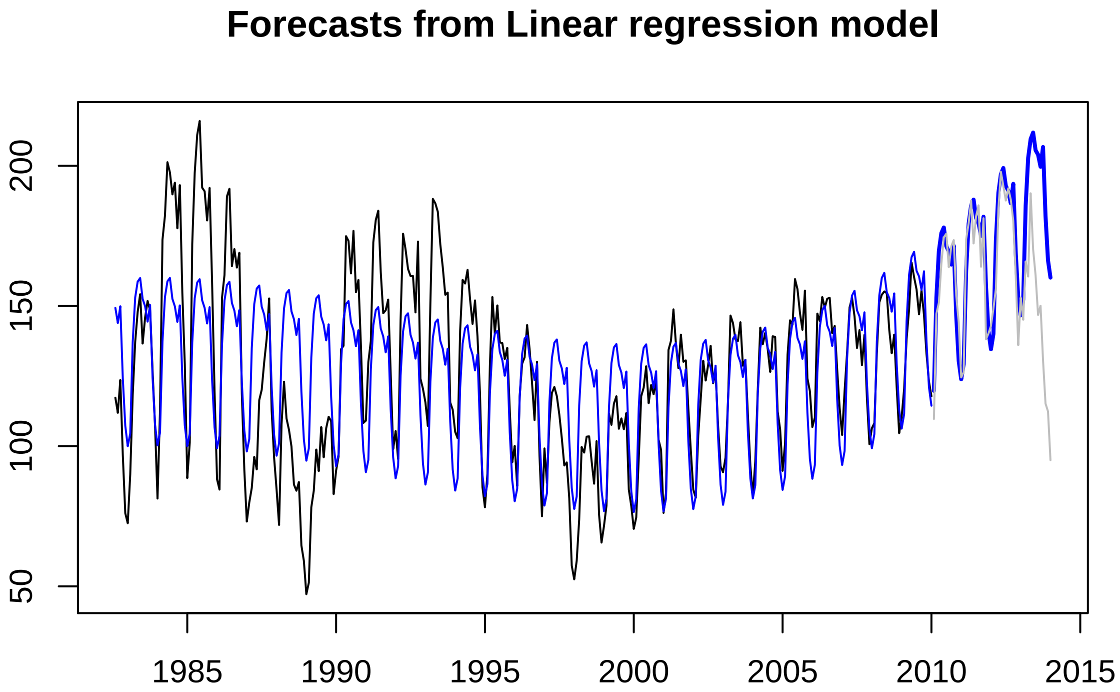
```
train.lm.season = tslm(train.ts ~ trend+ I(trend^2)+ I(trend^3)+season)
summary(train.lm.season)
```

```
##
## Call:
## tslm(formula = train.ts ~ trend + I(trend^2) + I(trend^3) + season)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -64.507 -12.423   1.163  14.035  56.453
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.958e+01  6.622e+00  15.039  < 2e-16 ***
## trend        7.573e-02  1.340e-01   0.565 0.572245
## I(trend^2)  -2.269e-03  9.396e-04  -2.415 0.016320 *
## I(trend^3)   6.593e-06  1.866e-06   3.533 0.000472 ***
## season2      4.557e+00  6.209e+00   0.734 0.463578
## season3      3.707e+01  6.209e+00   5.971 6.35e-09 ***
## season4      5.295e+01  6.208e+00   8.528 6.31e-16 ***
## season5      5.858e+01  6.208e+00   9.436  < 2e-16 ***
## season6      5.981e+01  6.208e+00   9.635  < 2e-16 ***
## season7      5.235e+01  6.208e+00   8.433 1.23e-15 ***
## season8      4.968e+01  6.152e+00   8.076 1.44e-14 ***
## season9      4.420e+01  6.151e+00   7.186 4.86e-12 ***
## season10     5.011e+01  6.151e+00   8.147 8.86e-15 ***
## season11     2.438e+01  6.151e+00   3.964 9.12e-05 ***
## season12     7.381e+00  6.150e+00   1.200 0.230979
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.01 on 315 degrees of freedom
## Multiple R-squared:  0.512,  Adjusted R-squared:  0.4903
## F-statistic: 23.61 on 14 and 315 DF,  p-value: < 2.2e-16
```

```r
head(train.lm.season$data)
```

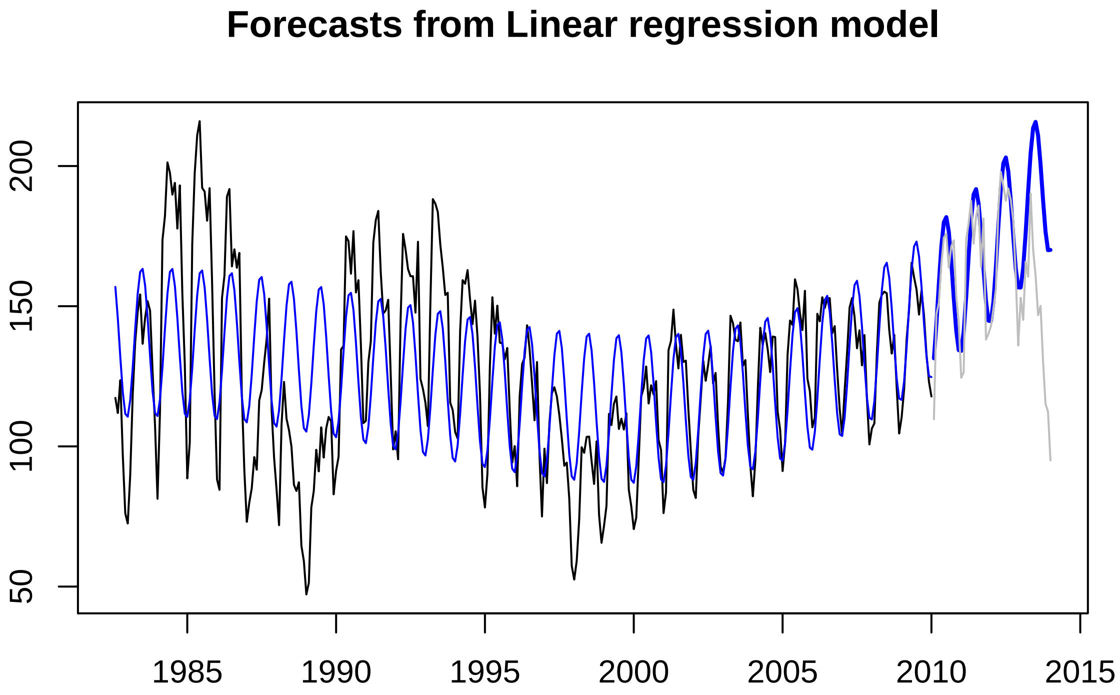
```
##   train.ts trend season
## 1    117.3     1      8
## 2    111.9     2      9
## 3    123.6     3     10
## 4     96.9     4     11
## 5     76.1     5     12
## 6     72.5     6      1
```

```r
train.lm.season.pred = forecast(train.lm.season, h = nvalid, level = 0)
plot(train.lm.season.pred)
lines(valid.ts, col = "grey")
lines(train.lm.season.pred$fitted, col = "blue")
```



2.2 Use sine and cosine term in seasonality

```
lm.sine = tslm(train.ts~ trend + I(trend^2) + I(trend^3)+I(sin(2*pi*trend/12)) + I(cos(2*pi*trend/12)))
lm.sine.pred = forecast(lm.sine, h = nvalid, level = 0)
plot(lm.sine.pred)
lines(valid.ts, col = "grey")
lines(lm.sine$fitted, col = "blue")
```



```r
accuracy(lm.sine.pred, valid.ts)
```

```
##                          ME     RMSE      MAE       MPE     MAPE     MASE
## Training set -1.700468e-15 24.32625 18.86907 -4.348292 16.61920 1.116700
## Test set     -1.298100e+01 26.46307 18.22477 -9.956831 13.01158 1.078569
##                   ACF1 Theil's U
## Training set 0.8034601        NA
## Test set     0.7289304  1.634007
```

```r
accuracy(train.lm.season.pred, valid.ts)
```

```
##                          ME     RMSE      MAE       MPE     MAPE      MASE
## Training set  4.540534e-16 22.48331 17.22124 -3.609149 15.14278 1.0191787
## Test set     -1.332478e+01 26.00711 16.72029 -9.888076 12.01881 0.9895322
##                   ACF1 Theil's U
## Training set 0.8835683        NA
## Test set     0.7774046    1.6009
```