Is There A Connection Between 3-Pointers and Wins?

Muiz Rahemtullah

6/27/2018

Introduction

Over recent years, the use of data analytics has been very influential in professional sports. One such example is in basketball. In the past few years, it has been determined by NBA analysts that the only shots worth atempting are free throws, shots in the paint, and most importantly, three pointers. It would then make sense to examine how adapting to this style of basketball impacts the number of wins a team earns during the regular season. In this project, we will examine if there is a connection between the Houston Rockets Wins and their 3 pointer statistics, specifically the number of 3 pointers attempted and the percentage of 3 point field goals made. We want to see whether or not a high 3 point statistics influence and cause more wins. To do this, we will use a Vector Autoregression (VAR) model. As mentioned before, we will look at three variables: The Number of Wins, 3 pointers attempted, and percentage of 3 pointers made. The VAR model will see how much of these present value of these variables are dependant on the past values of these variables.

Results

First we will set our working directory.

```
setwd("~/Desktop/Summer Project #2")
```

Now we will load in our libraries.

```
library(lattice)
library(foreign)
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 3.5.2
```

```
library(car)
```

```
## Loading required package: carData
```

```
require(stats)
require(stats4)
```

```
## Loading required package: stats4
```

```
library(KernSmooth)
```

```
## KernSmooth 2.23 loaded
## Copyright M. P. Wand 1997-2009
```

```
library(fastICA)
library(cluster)
library(leaps)
library(mgcv)
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-23. For overview type 'help("mgcv-package")'.
```

```r
library(rpart)
library(pan)
library(mgcv)
library(DAAG)
```

```
##
## Attaching package: 'DAAG'

## The following object is masked from 'package:car':
##
##     vif

## The following object is masked from 'package:MASS':
##
##     hills
```

```r
library("TTR")
library(tis)
```

```
##
## Attaching package: 'tis'

## The following object is masked from 'package:TTR':
##
##     lags

## The following object is masked from 'package:mgcv':
##
##     ti
```

```r
require("datasets")
require(graphics)
library("forecast")
```

```
##
## Attaching package: 'forecast'

## The following object is masked from 'package:tis':
##
##     easter

## The following object is masked from 'package:nlme':
##
##     getResponse
```

```r
#install.packages("astsa")
#require(astsa)
library(nlstools)
```

```
##
## 'nlstools' has been loaded.

## IMPORTANT NOTICE: Most nonlinear regression models and data set examples

## related to predictive microbiolgy have been moved to the package 'nlsMicrobio'
```

```r
library(fpp)
```

```
## Loading required package: fma

##
```

```
## Attaching package: 'fma'

## The following objects are masked from 'package:DAAG':
##
##     milk, ozone

## The following objects are masked from 'package:MASS':
##
##     cement, housing, petrol

## Loading required package: expsmooth

## Loading required package: lmtest

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: tseries
```

```
library(strucchange)
```

```
## Loading required package: sandwich
```

```
library(Quandl)
```

```
## Loading required package: xts
```

```
library(zoo)
library(PerformanceAnalytics)
```

```
##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend
```

```
library(quantmod)
```

```
## Version 0.4-0 included new data defaults. See ?getSymbols.

##
## Attaching package: 'quantmod'

## The following object is masked from 'package:tis':
##
##     Lag
```

```
#library(stockPortfolio)
library(vars)
```

```
## Loading required package: urca
```

```
library(lmtest)
library(dlnm)
```

```
## This is dlnm 2.3.4. For details: help(dlnm) and vignette('dlnmOverview').
```

```r
library(hts)
library(tseries)
library(rugarch)
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'rugarch'
```

```
## The following object is masked from 'package:stats':
##
##     sigma
```

Now we will load in our data.

```r
library(readxl)
wins_and_threes <- read_excel("Summer Project #2 Data.xlsx")
View(wins_and_threes)
```

Now we will construct a VAR model using the data.

```r
library(vars)
library(VAR.etp)
wins_ts <- ts(wins_and_threes$W, start = 1980, frequency = 1)
three_ts <- ts(wins_and_threes$`3PA`, start = 1980, frequency = 1)
three_pct_ts <- ts(wins_and_threes$`3P%`, start = 1980, frequency = 1)
y = cbind(wins_ts, three_ts, three_pct_ts)
y_updated <- y[1:39,]
y_tot = data.frame(y_updated)
```

Now that we have constructed the VAR Model, its time to examine which order to choose. The order determines how many lagged variables to put in our model. For example, a VAR(1) model would include the first lagged terms, in this case the previous seasons number of wins, three pointers attempted, and three point percentage. The VARselect function gives us an idea on which order to select. We want to minimize the BIC (SC) value.

```r
VARselect(y_tot, lag.max = 5)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      4      1      1      4
##
## $criteria
##                  1           2           3           4           5
## AIC(n)    8.420259    8.360508    8.446899    8.117893    8.236605
## HQ(n)     8.603976    8.682013    8.906193    8.714975    8.971475
## SC(n)     8.958974    9.303260    9.793688    9.868719   10.391467
## FPE(n) 4553.003901 4352.123083 4917.722890 3792.008893 4807.965968
```

The BIC or SC is minimized at order 1, hence we will use a VAR(1) model.

```r
y_model <- VAR(y_tot, p = 1)
summary(y_model)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: wins_ts, three_ts, three_pct_ts
## Deterministic variables: const
## Sample size: 38
## Log Likelihood: -327.654
## Roots of the characteristic polynomial:
## 0.9791  0.37  0.37
## Call:
## VAR(y = y_tot, p = 1)
##
##
## Estimation results for equation wins_ts:
## ========================================
## wins_ts = wins_ts.l1 + three_ts.l1 + three_pct_ts.l1 + const
##
##                   Estimate Std. Error t value Pr(>|t|)
## wins_ts.l1        0.199055   0.175372   1.135  0.26430
## three_ts.l1       0.006879   0.003005   2.289  0.02841 *
## three_pct_ts.l1 -64.124339  44.977792  -1.426  0.16308
## const            48.149942  13.400585   3.593  0.00102 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 9.234 on 34 degrees of freedom
## Multiple R-Squared: 0.2048,  Adjusted R-squared: 0.1346
## F-statistic: 2.919 on 3 and 34 DF,  p-value: 0.04808
##
##
## Estimation results for equation three_ts:
## =========================================
## three_ts = wins_ts.l1 + three_ts.l1 + three_pct_ts.l1 + const
##
##                  Estimate Std. Error t value Pr(>|t|)
## wins_ts.l1        -1.9433     6.6772  -0.291    0.773
## three_ts.l1        0.9373     0.1144   8.192 1.48e-09 ***
## three_pct_ts.l1 1174.2933  1712.4940   0.686    0.498
## const           -134.1639   510.2167  -0.263    0.794
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 351.6 on 34 degrees of freedom
## Multiple R-Squared: 0.8449,  Adjusted R-squared: 0.8312
## F-statistic: 61.72 on 3 and 34 DF,  p-value: 7.67e-14
##
##
## Estimation results for equation three_pct_ts:
## =============================================
## three_pct_ts = wins_ts.l1 + three_ts.l1 + three_pct_ts.l1 + const
##
##                  Estimate Std. Error t value Pr(>|t|)
## wins_ts.l1      4.141e-04  6.232e-04   0.664  0.51087
## three_ts.l1     1.730e-05  1.068e-05   1.620  0.11455
## three_pct_ts.l1 5.388e-01  1.598e-01   3.371  0.00188 **
## const           1.119e-01  4.762e-02   2.350  0.02474 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.03282 on 34 degrees of freedom
## Multiple R-Squared: 0.632,   Adjusted R-squared: 0.5995
## F-statistic: 19.46 on 3 and 34 DF,  p-value: 1.598e-07
##
##
##
## Covariance matrix of residuals:
##                 wins_ts  three_ts three_pct_ts
## wins_ts      8.527e+01 1.240e+03     0.082678
## three_ts     1.240e+03 1.236e+05     3.160621
## three_pct_ts 8.268e-02 3.161e+00     0.001077
##
## Correlation matrix of residuals:
##               wins_ts three_ts three_pct_ts
## wins_ts       1.0000   0.3819       0.2728
## three_ts      0.3819   1.0000       0.2739
## three_pct_ts  0.2728   0.2739       1.0000
```

Now we need to see if three pointers cause wins.

```
grangertest(wins_ts~three_pct_ts, order = 1)
```

```
## Granger causality test
##
## Model 1: wins_ts ~ Lags(wins_ts, 1:1) + Lags(three_pct_ts, 1:1)
## Model 2: wins_ts ~ Lags(wins_ts, 1:1)
##   Res.Df Df      F Pr(>F)
## 1     35
## 2     36 -1 0.1115 0.7404
```

```
grangertest(wins_ts~three_ts, order = 1)
```

```
## Granger causality test
##
## Model 1: wins_ts ~ Lags(wins_ts, 1:1) + Lags(three_ts, 1:1)
## Model 2: wins_ts ~ Lags(wins_ts, 1:1)
##   Res.Df Df      F  Pr(>F)
## 1     35
## 2     36 -1 3.2365 0.08064 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that three pointers attempted influence wins at the 10% significance level. Hence three pointers do cause wins.

Conclusion

NBA Analysts seem to really value the three pointer and using our model, we are able to confirm that the three pointer is indeed an important aspect of winning. The number of threes we attempt influence the number of regular season wins. To win more, we must attempt more three pointers.

References

“Houston Rockets Stats-Basic (Totals)”, basketball-reference.com, Sports Reference, https://www.basketball-reference.com/teams/HOU/stats_basic_totals.html