

# CAPM Fitting and Testing

Thomas Fillebeen

January 22, 2014

## Abstract

Standard Capital Asset Pricing Model (CAPM) fitting and testing using Quandl data.

CAPM Assumptions 1. Identical investors who are price takers; 2. Investment over the same time horizon; 3. No transaction costs or taxes; 4. Can borrow and lend at risk-free rate; 5. Investors only care about portfolio expected return and variance; 6. Market consists of all publicly traded assets.

The Consumption-Oriented CAPM is analogous to the simple form of the CAPM. Except that the growth rate of per capita consumption has replaced the rate of return on the market portfolio as the influence effecting returns.

## Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Fitting CAPM</b>                                   | <b>1</b>  |
| 1.1      | Selected Returns Time Series . . . . .                | 1         |
| 1.2      | Estimate Excess Returns . . . . .                     | 2         |
| 1.3      | Fitting CAPM Model . . . . .                          | 3         |
| <b>2</b> | <b>Testing CAPM</b>                                   | <b>5</b>  |
| 2.1      | Created CAPM Function . . . . .                       | 5         |
| 2.2      | Estimate Significance and Test Beta Results . . . . . | 6         |
| 2.3      | Estimate Expected Returns and Plot . . . . .          | 7         |
| <b>3</b> | <b>Consumption-Oriented CAPM</b>                      | <b>10</b> |
| 3.1      | Fitting C-CAPM . . . . .                              | 10        |

## 1 Fitting CAPM

### 1.1 Selected Returns Time Series

```
# 'Load the GARPFRM package and the CAPM dataset.
suppressMessages(library(GARPFRM))

## Warning: package 'PerformanceAnalytics' was built under R version 2.15.2

options(digits = 3)
data(crsp.short)
data(cons)
stock.df <- cbind(largecap.ts, cons[, "CONS"])
colnames(stock.df) = c(colnames(largecap.ts), "CONS")
colnames(stock.df)

## [1] "AMAT" "AMGN" "CAT" "DD" "G" "GENZ" "GM"
## [8] "HON" "KR" "LLTC" "MSFT" "ORCL" "PG" "PHA"
## [15] "SO" "TXN" "UTX" "WM" "WYE" "YHOO" "market"
## [22] "t90" "CONS"
```

Summarize the first and last data values corresponding to the first 5 dates for the first 5 returns.

```
head(stock.df[, 1:5])

##           AMAT    AMGN    CAT    DD    G
## 1997-01-31 0.3739 0.0368 0.03688 0.16467 0.0634
## 1997-02-28 0.0253 0.0843 0.00805 -0.01647 -0.0409
## 1997-03-31 -0.0840 -0.0859 0.02556 -0.01166 -0.0821
## 1997-04-30 0.1833 0.0537 0.11402 0.00118 0.1734
## 1997-05-30 0.1891 0.1359 0.09691 0.02949 0.0456
## 1997-06-30 0.0852 -0.1308 0.09987 0.15765 0.0661

tail(stock.df[, 1:5])

##           AMAT    AMGN    CAT    DD    G
## 2001-07-31 -0.06599 0.0335 0.10789 -0.1124 -0.0330
## 2001-08-31 -0.06040 0.0254 -0.09256 -0.0350 0.0997
## 2001-09-28 -0.33999 -0.0860 -0.10400 -0.0842 -0.0277
## 2001-10-31 0.19937 -0.0332 0.00603 0.0658 0.0487
## 2001-11-30 0.16505 0.1691 0.06038 0.1175 0.0518
## 2001-12-31 0.00906 -0.1504 0.10186 -0.0413 0.0214
```

```
# Count the number of rows
nrow(stock.df)

## [1] 60
```

## 1.2 Estimate Excess Returns

Estimate excess returns: subtracting off risk-free rate. To strip off the dates and just return a plain vector/matrix `coredata()` can be used.

```
# as.data.frame to check if an object is a data frame, or coerce it if
# possible.
returns.mat = coredata(stock.df)
exReturns.mat = returns.mat - returns.mat[, "t90"]
exReturns.df = as.data.frame(exReturns.mat)
```

## 1.3 Fitting CAPM Model

Run CAPM regression for AAPL (AAPL) using first 5 years (60 months divided by 12 months in a years = 5 years).

```
capm.fit = lm(MSFT~market,data=exReturns.df,subset=1:60)
summary(capm.fit)

##
## Call:
## lm(formula = MSFT ~ market, data = exReturns.df, subset = 1:60)
##
## Residuals:
```

|  | Min     | 1Q      | Median  | 3Q     | Max    |
|--|---------|---------|---------|--------|--------|
|  | -0.2909 | -0.0724 | -0.0061 | 0.0757 | 0.3299 |

```
##
## Coefficients:
```

|             | Estimate | Std. Error | t value | Pr(> t )  |
|-------------|----------|------------|---------|-----------|
| (Intercept) | 0.0178   | 0.0154     | 1.15    | 0.25      |
| market      | 1.6869   | 0.2857     | 5.90    | 2e-07 *** |

```
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.119 on 58 degrees of freedom
## Multiple R-squared:  0.375, Adjusted R-squared:  0.365
## F-statistic: 34.9 on 1 and 58 DF,  p-value: 1.96e-07

# Plot data with regression line
plot(exReturns.df$market,exReturns.df$MSFT, main="CAPM for MSFT",

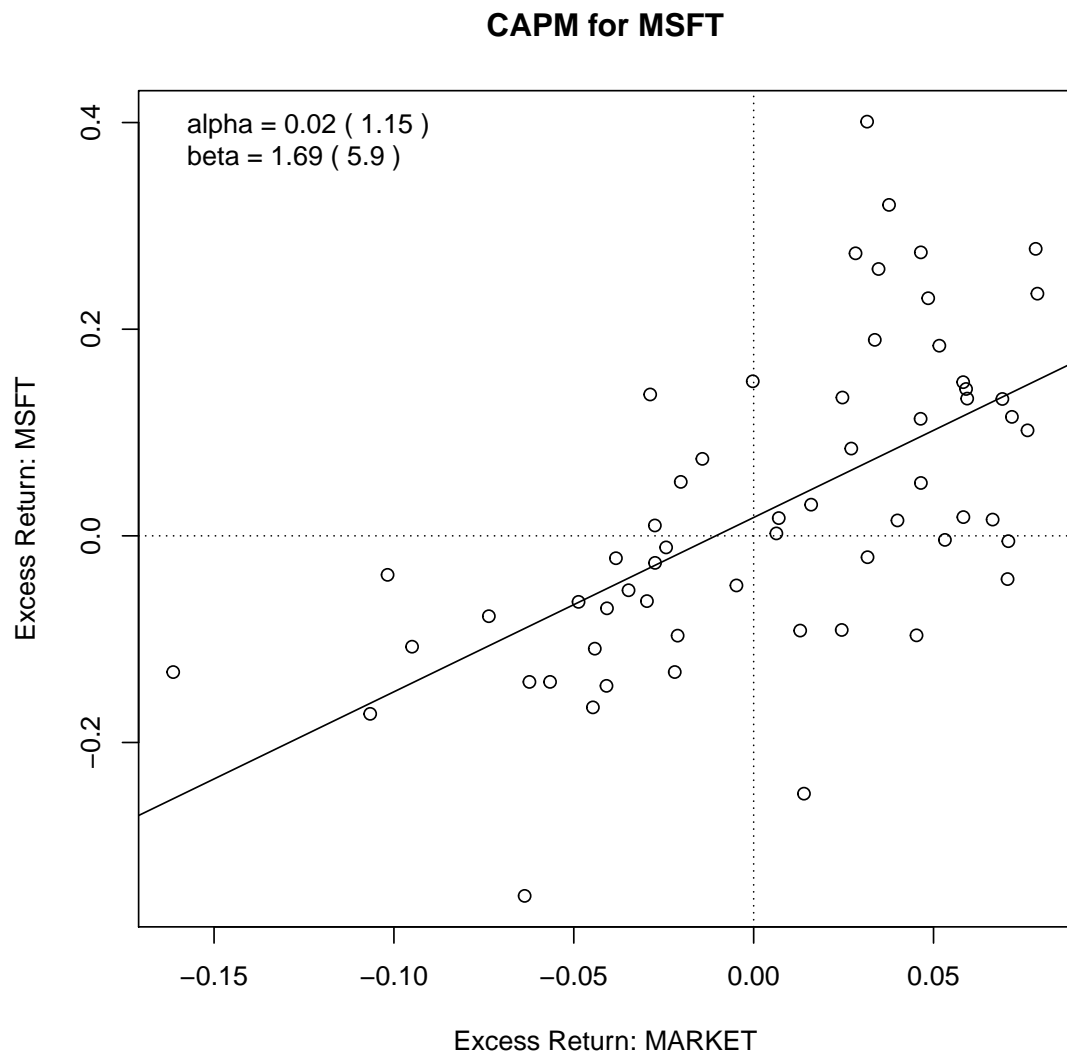
      ylab="Excess Return: MSFT",
      xlab="Excess Return: MARKET")

# Plot CAPM regression estimate
abline(capm.fit)

# Create Axis
abline(h=0,v=0,lty=3)

# Placing beta & tstat values on the plot for APPL
alpha = coef(summary(capm.fit))[1,1]
a_tstat = coef(summary(capm.fit))[1,3]
beta = coef(summary(capm.fit))[2,1]
b_tstat = coef(summary(capm.fit))[2,3]

legend("topleft", legend=
c(paste("alpha =",round(alpha,dig=2),"(",round(a_tstat, dig=2),")"),
paste("beta =",round(beta,dig=2),"(",round(b_tstat,dig=2),")")), cex=1, bty="n")
```



## 2 Testing CAPM

### 2.1 Created CAPM Function

Use a `capm.tstats` function: Estimating CAPM with  $\alpha=0$  for asset using first 5 years of data

```
capm.tstats = function(r, mkrt) {
  # Fiting CAPM
  capm.fit = lm(r ~ mkrt)
  # Extract summary info
  capm.summary = summary(capm.fit)
```

```

# Retrieve t-stat
t.stat = coef(capm.summary)[1, 3]
t.stat
}

```

## 2.2 Estimate Significance and Test Beta Results

Retrieve tstats from function for assets. Filter out rf and market before running.

```

colnames(exReturns.mat[, -c(21, 22, 23)])

## [1] "AMAT" "AMGN" "CAT" "DD" "G" "GENZ" "GM" "HON" "KR" "LLTC"
## [11] "MSFT" "ORCL" "PG" "PHA" "SO" "TXN" "UTX" "WM" "WYE" "YHOO"

tstats = apply(exReturns.mat[1:60, -c(21, 22, 23)], 2,

               capm.tstats, exReturns.mat[1:60, "market"])
tstats

## AMAT AMGN CAT DD G GENZ GM HON KR
## 1.4836 1.5197 0.4054 -0.3632 -0.3520 1.8424 0.0358 0.0627 0.8060
## LLTC MSFT ORCL PG PHA SO TXN UTX WM
## 1.3259 1.1522 1.0347 0.5703 0.1565 1.4344 1.2355 0.7424 0.7358
## WYE YHOO
## 1.0268 1.9915

# Test Hypothesis for 5% CI: H0: alpha=0
abs(tstats) > 2

## AMAT AMGN CAT DD G GENZ GM HON KR LLTC MSFT ORCL
## FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## PG PHA SO TXN UTX WM WYE YHOO
## FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE

any(abs(tstats) > 2)

## [1] FALSE

```

## 2.3 Estimate Expected Returns and Plot

Plot expected return versus beta. Estimate expected returns over first 5 years.

```
mu.hat = colMeans(exReturns.mat[1:60,-c(21,22,23)])
mu.hat

##      AMAT      AMGN      CAT      DD      G      GENZ      GM
## 0.036476 0.026808 0.007818 -0.000317 -0.000903 0.033550 0.005562
##      HON      KR      LLTC      MSFT      ORCL      PG      PHA
## 0.005657 0.008811 0.028156 0.025948 0.033147 0.007610 0.003067
##      SO      TXN      UTX      WM      WYE      YHOO
## 0.012826 0.030411 0.013515 0.011877 0.013941 0.078199

# Compute beta over first 5 years
capm.betas = function(r,market) {
  capm.fit = lm(r~market)
  # Fit capm regression
  capm.beta = coef(capm.fit)[2]
  # Extract coefficients
  capm.beta
}

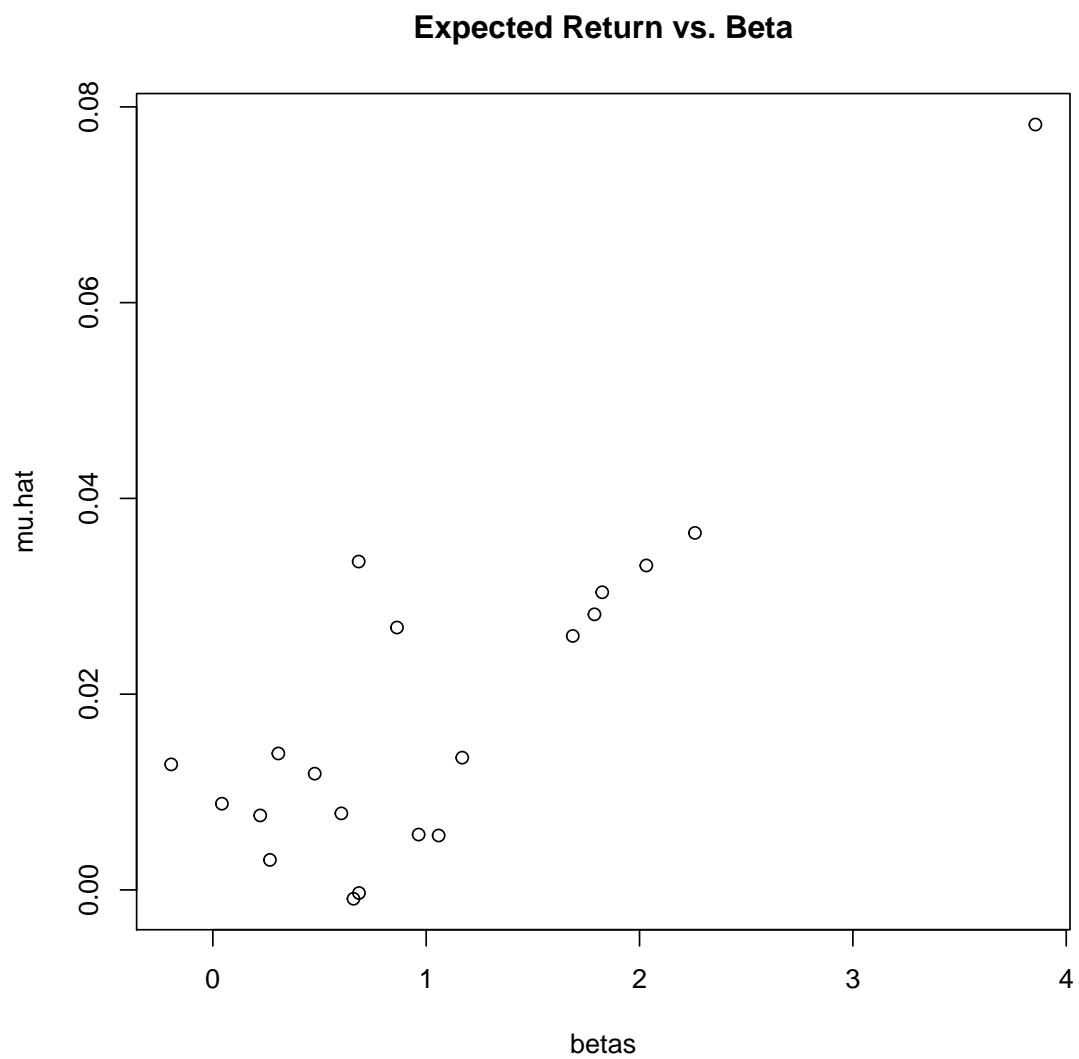
betas = apply(exReturns.mat[1:60,-c(21,22,23)],2,

              FUN=capm.betas,
              market=exReturns.mat[1:60,"market"])
betas

##      AMAT      AMGN      CAT      DD      G      GENZ      GM      HON      KR
## 2.2597 0.8632 0.6024 0.6850 0.6593 0.6837 1.0584 0.9646 0.0425
##      LLTC      MSFT      ORCL      PG      PHA      SO      TXN      UTX      WM
## 1.7888 1.6869 2.0318 0.2221 0.2677 -0.1951 1.8251 1.1689 0.4776
##      WYE      YHOO
## 0.3071 3.8555

# Plot expected returns versus betas
```

```
plot(betas,mu.hat,main="Expected Return vs. Beta")
```



```
# Estimate regression of Expected Return vs. Beta
sml.fit = lm(mu.hat~betas)
sml.fit

##
## Call:
## lm(formula = mu.hat ~ betas)
##
## Coefficients:
```



```
## (Intercept)      betas
##      0.00174      0.01635

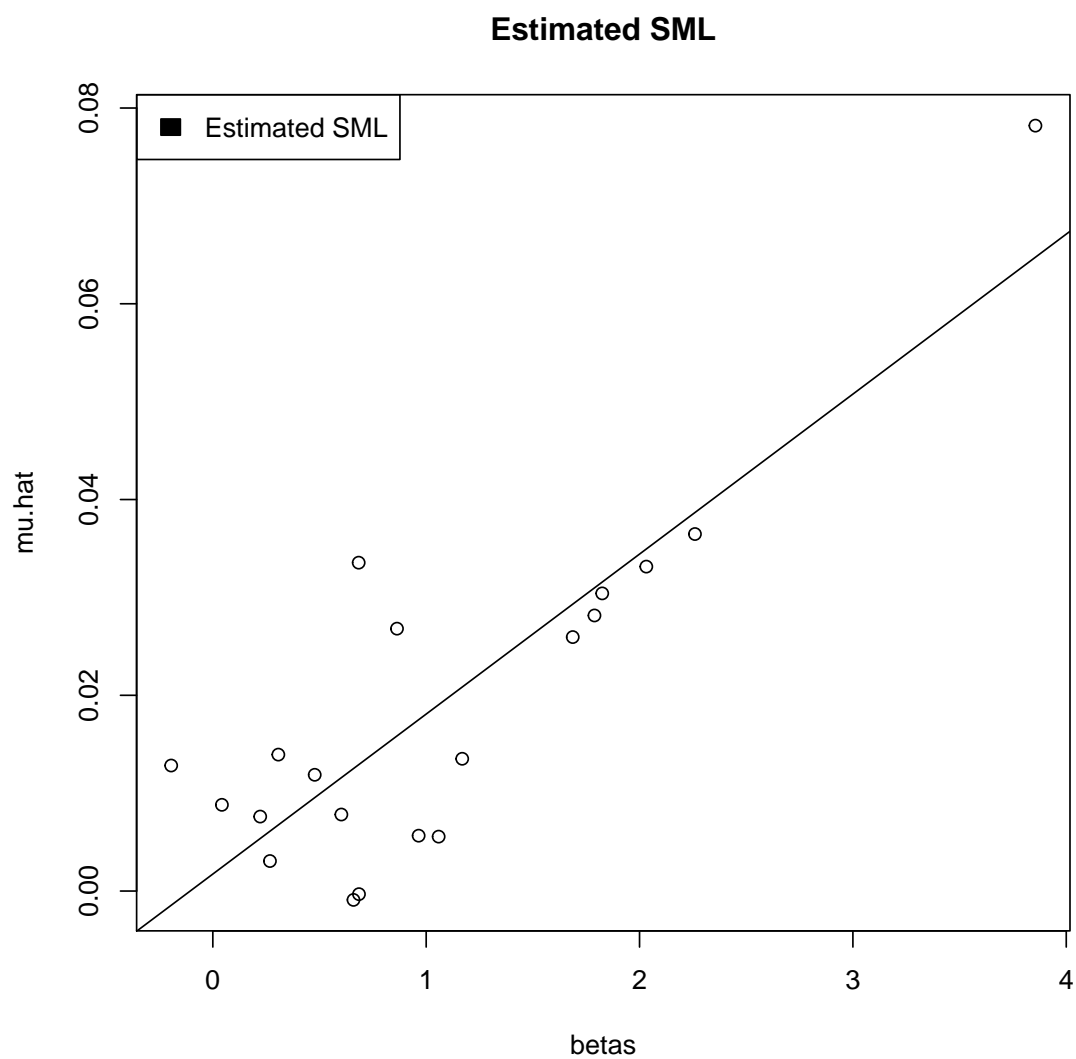
summary(sml.fit)

##
## Call:
## lm(formula = mu.hat ~ betas)
##
## Residuals:
##      Min        1Q    Median        3Q       Max
## -0.01348 -0.00466 -0.00200  0.00658  0.02064
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.00174    0.00341    0.51   0.62
## betas        0.01635    0.00242    6.76 2.5e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01 on 18 degrees of freedom
## Multiple R-squared:  0.717, Adjusted R-squared:  0.702
## F-statistic: 45.7 on 1 and 18 DF,  p-value: 2.47e-06

# Ideally intercept is zero and equals the excess market return
mean(exReturns.mat[1:60,"market"])

## [1] 0.00486

# Plot Fitted SML
plot(betas,mu.hat,main="Estimated SML")
abline(sml.fit)
legend("topleft",1, "Estimated SML",1)
```



## 3 Consumption-Oriented CAPM

### 3.1 Fitting C-CAPM

Run C-CAPM regression for CONS (Consumption) using first 5 years (60 months divided by 12 months in a years = 5 years).

```
end = nrow(stock.df)
capm.fit = lm(CONS~market,data=exReturns.df,subset=(end-60):end)
summary(capm.fit)
```

```
##
## Call:
## lm(formula = CONS ~ market, data = exReturns.df, subset = (end -
##      60):end)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.4323 -0.0949 -0.0024  0.0498  0.5084
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.1718     0.0248    6.94 3.8e-09 ***
## market        0.4601     0.4593    1.00  0.32
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.191 on 58 degrees of freedom
## Multiple R-squared:  0.017, Adjusted R-squared: 5.61e-05
## F-statistic:    1 on 1 and 58 DF,  p-value: 0.321

# Plot data with regression line
plot(exReturns.df$market, exReturns.df$CONS, main="CAPM for CONS",

      ylab="Excess Return: CONS",
      xlab="Excess Return: market")

# Plot C-CAPM regression estimate
abline(capm.fit)

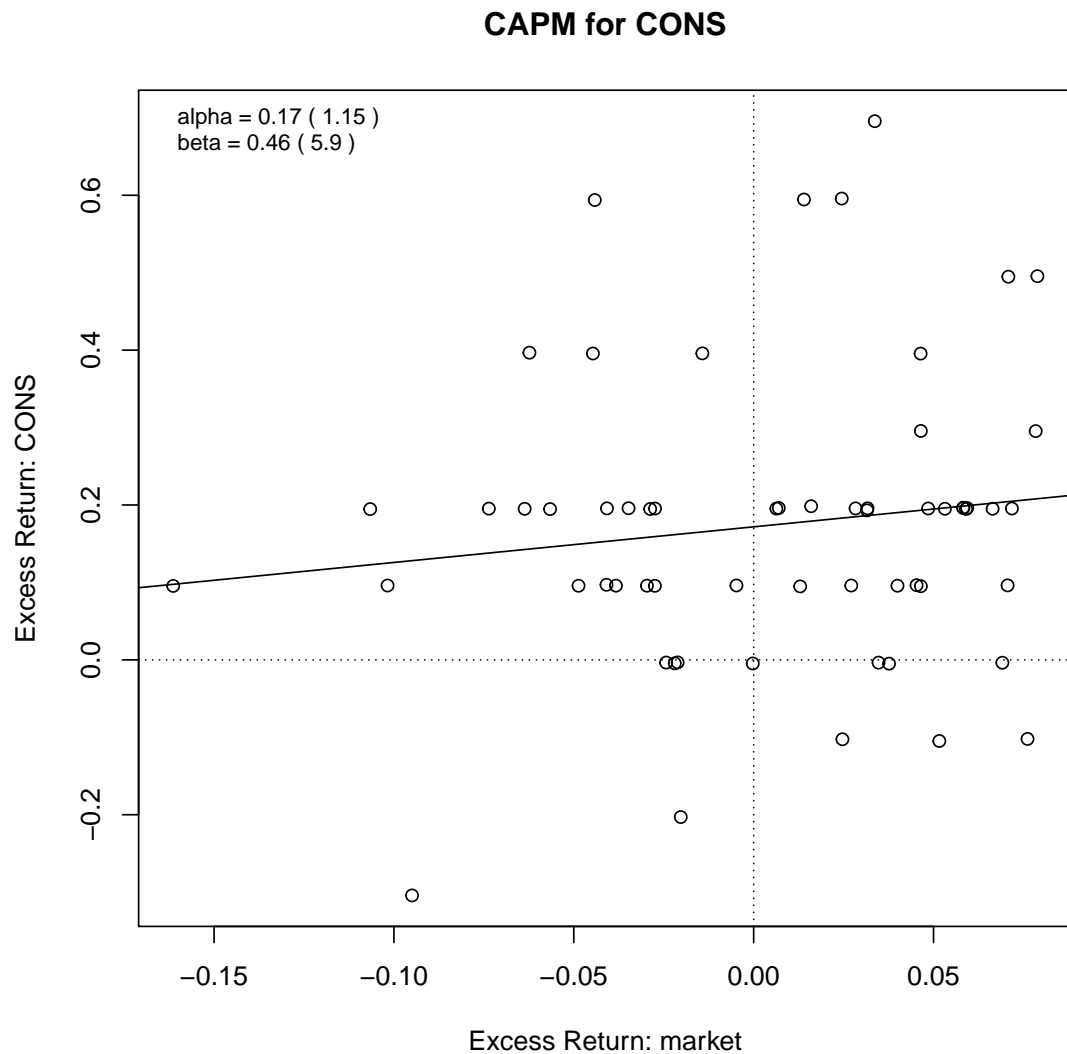
# Create Axis
abline(h=0, v=0, lty=3)

# Placing beta & tstat values on the plot for CONS
beta = coef(summary(capm.fit))[2,1]
b_stat = coef(summary(capm.fit))[2,3]
alpha = coef(summary(capm.fit))[1,1]
a_stat = coef(summary(capm.fit))[1,3]
```

```

legend("topleft", legend=
c(paste("alpha =",round(alpha,dig=2),"(",round(a_tstat, dig=2),")"),
paste("beta =",round(beta,dig=2),"(",round(b_tstat,dig=2),")")), cex=.8, bty="n")

```



NOTE: Specific problems with CCAPM is that it suffers from two puzzles: the equity premium puzzle (EPP) and the risk-free rate puzzle (RFRP). EPP implies that investors are extremely risk averse to explain the existence of a market risk premium. While RFRP stipulates that investors save in TBills despite the low rate of return.