Price-Book Ratio to Annual Return

Yusef Al-Naher

February 3, 2019

Below is some analysis conducted determining the effect of Price-Book value to subsequent 10 year returns. Data gathered from Thompson Reuters' Eikon.

```
x <- read.csv(file="C:/Users/Sef/Desktop/Stats Project ASE/totPBret.csv", header=TRUE, sep=",")
library(ggplot2)
library('xts')
library('tidyverse')
library('broom')
library('descriptr')
library('olsrr')
library('fitdistrplus')
library('logspline')
library('MASS')
library('robustbase')
library('Hmisc')
library('lmtest')
```

```
head(x)
```

```
##   X   PB     Return
## 1 1 2.63 0.11918479
## 2 2 0.81 0.03100583
## 3 3 3.53 0.12898107
## 4 4 5.15 0.17560184
## 5 5 1.40 0.14687995
## 6 6 1.15 0.11419013
```

Data has been re-shaped into the following format: Stock #, Price-Book Value, Subsequent 10 Year Return

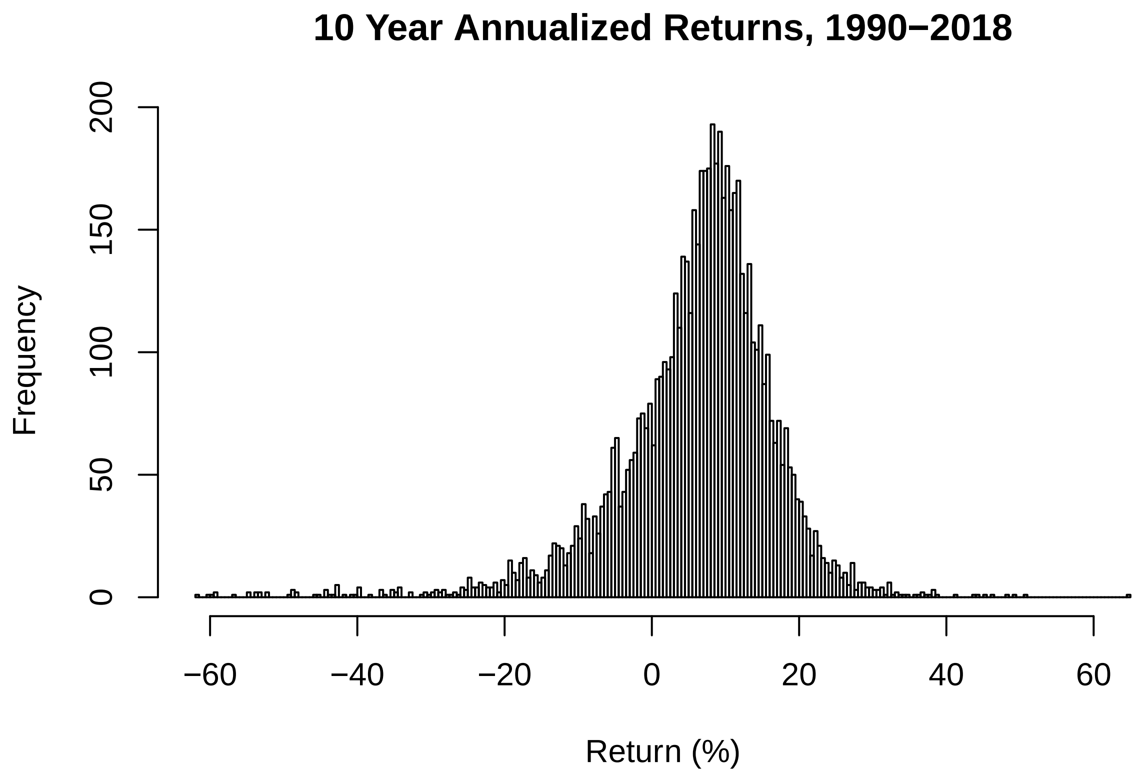
x<- x[,2:3] y<- data.frame(x[,1], (100*x[,2])) names(y)<-c("PB", "Return")

```
x<- x[,2:3]
y<- data.frame(x[,1], (100*x[,2]))
names(y)<-c("PB", "Return")
```

Splitting the Data into positive, negative, and 'low' (<5) Price-Book Values

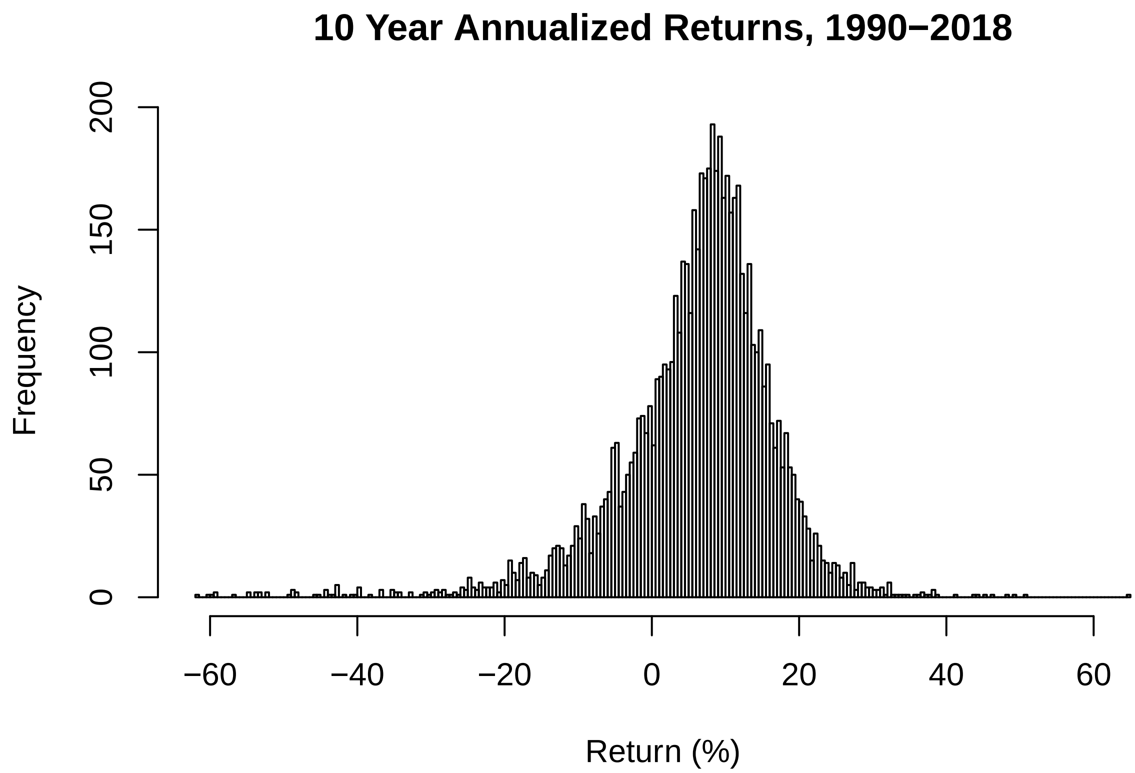
```
ypos<- y[ which(y$PB > 0) ,]
ylow<-ypos[which(ypos$PB<5),]
ylow<-ypos[which(ypos$PB<5),]
```

```
hist(y$Return, breaks= 200, main="10 Year Annualized Returns, 1990-2018", xlab="Return (%)")
```

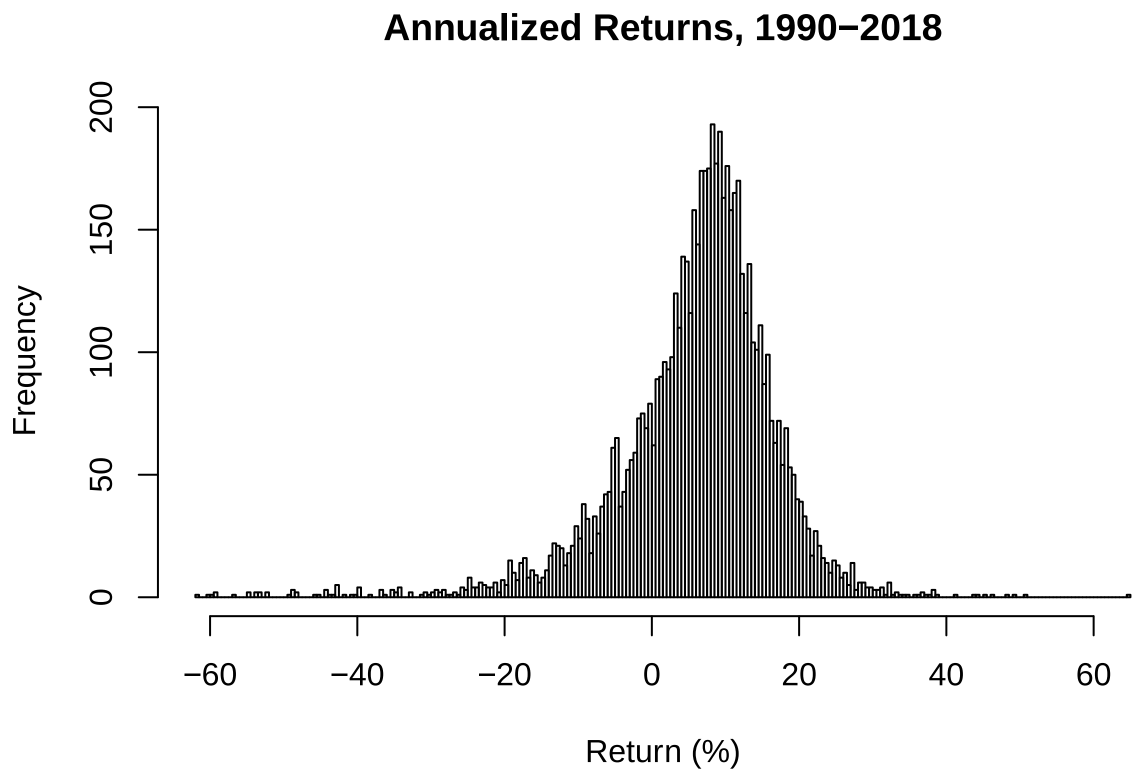


Above are returns plotted for the complete data set and below is the same for the section of the data that had a positive Price-Book Value

```
hist(ypos$Return, breaks= 200, main="10 Year Annualized Returns, 1990-2018", xlab="Return (%)")
```




```
h<-hist(y$Return, breaks=200 , xlab="Return (%)",  #Histogram
        main="Annualized Returns, 1990-2018")
```




```
xfit<-seq(min(y$Return),max(y$Return),length=40)
yfit<-dnorm(xfit,mean=mean(y$Return),sd=sd(y$Return))
yfit <- yfit*diff(h$mids[1:2])*length(y$Return)
```

Average Return by examining mean return for whole data set and data set limited to positive PB values

```
summary(y)
```

```
##        PB                Return
##  Min.   :-2127.960   Min.   :-61.623
##  1st Qu.:    1.710   1st Qu.:  1.474
##  Median :    2.590   Median :  7.624
##  Mean   :    3.161   Mean   :  6.191
##  3rd Qu.:    4.140   3rd Qu.: 12.342
##  Max.   :  243.480   Max.   : 64.984
```

```
summary(ypos)
```

```
##        PB             Return
##  Min.   : 0.220   Min.   :-61.623
##  1st Qu.: 1.740   1st Qu.:  1.487
##  Median : 2.610   Median :  7.626
##  Mean   : 3.875   Mean   :  6.205
##  3rd Qu.: 4.162   3rd Qu.: 12.323
```

```
##  Max.   :243.480   Max.   : 64.984
```

```
cor(ypos$PB, ypos$Return)
```

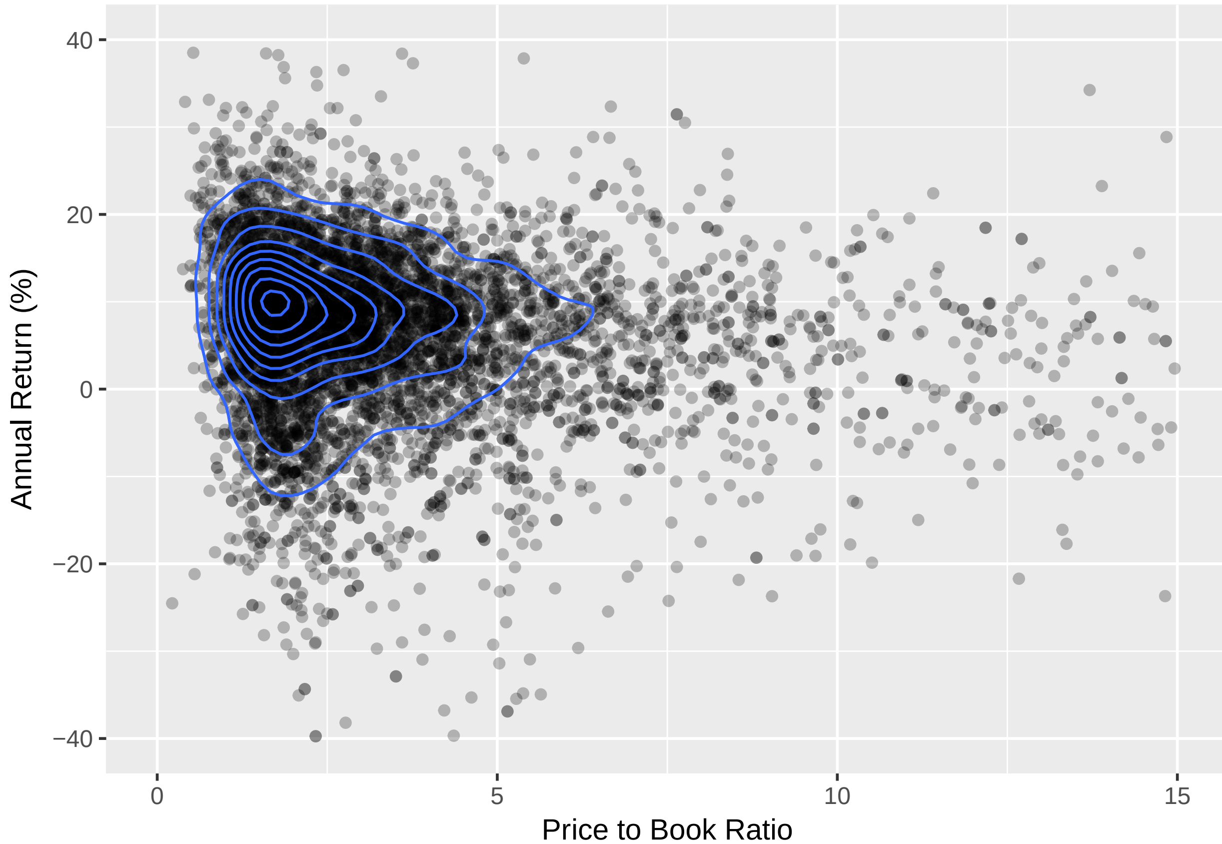
```
## [1] -0.05872881
```

Visualising the Data

```
ggplot(ypos, aes(x=PB, y=Return))+ geom_point(alpha=0.25)+ xlim(0,15) + geom_density2d() + ylim(-40, 40)
```

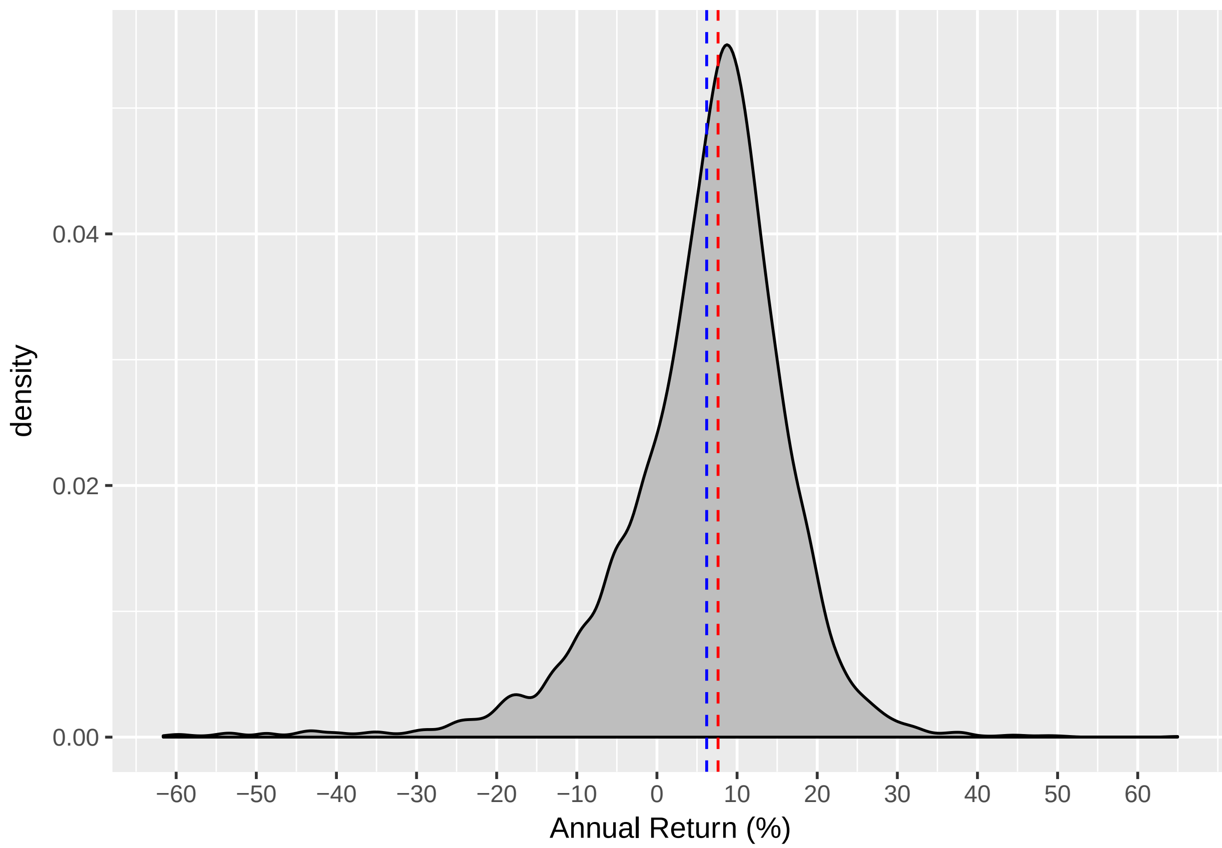
```
## Warning: Removed 161 rows containing non-finite values (stat_density2d).
```

```
## Warning: Removed 161 rows containing missing values (geom_point).
```



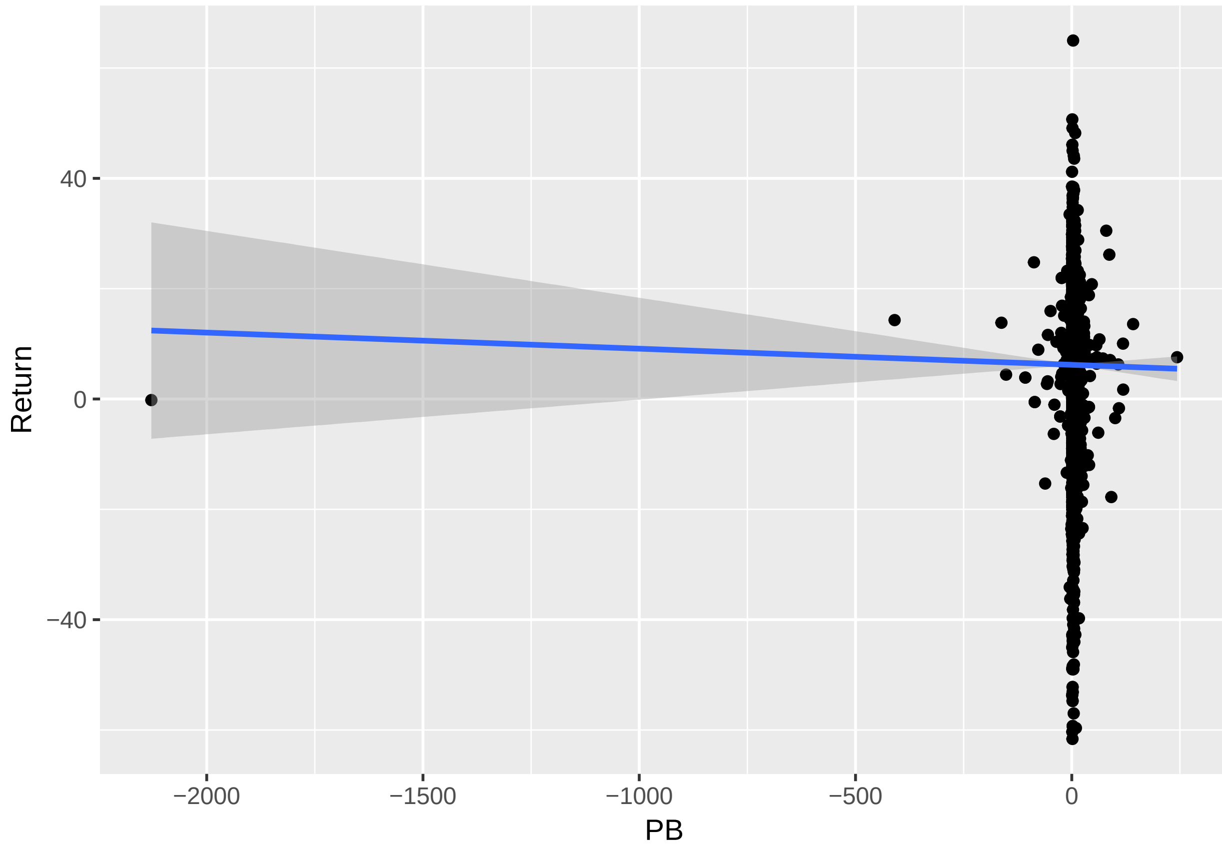
Density Plot with Mean and Median Return, Blue and Red respectively

```
ggplot(ypos, aes(x=Return)) + geom_density(fill="gray")+ geom_vline(aes(xintercept=mean(Return)), linet
```

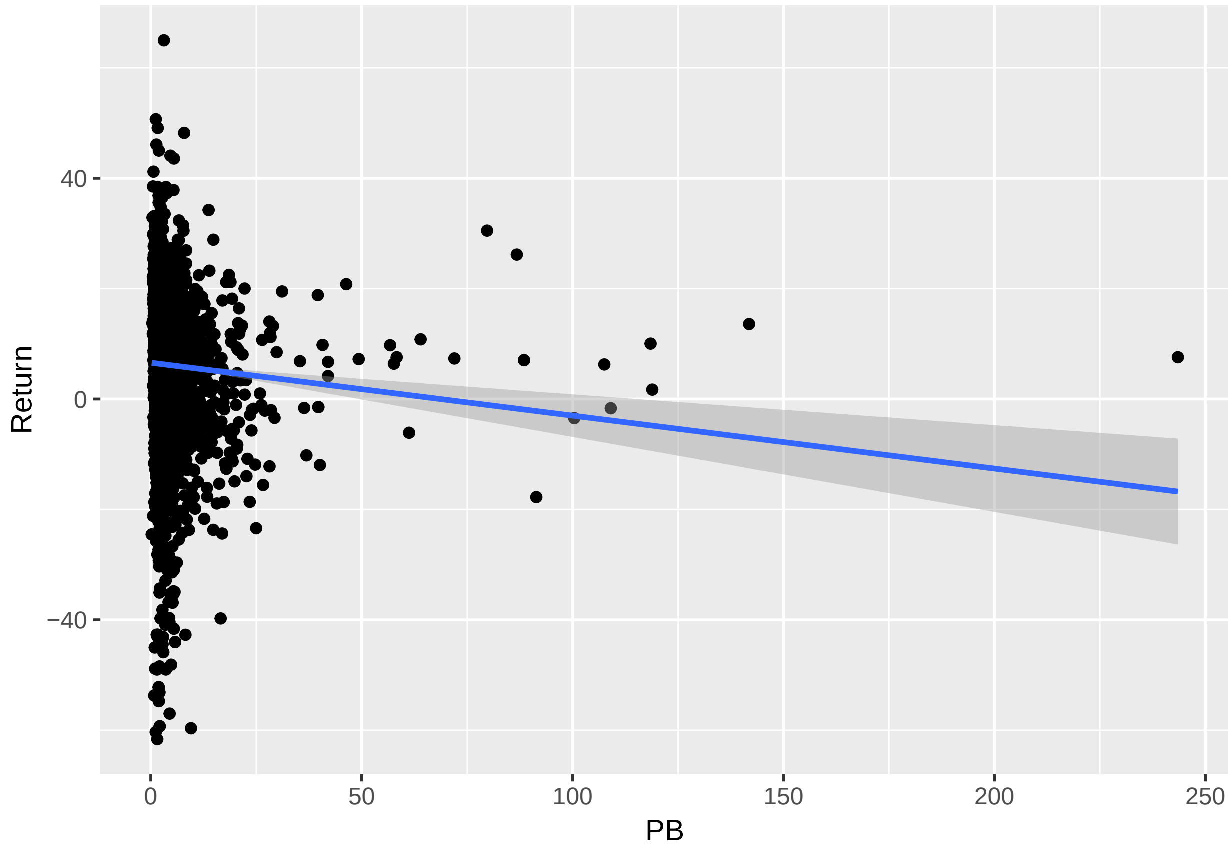


Linear Regression (grey shading is standard error) - for both Positive PB and whole Data Set

```
ggplot(data=y, aes(x=PB, y=Return))+ geom_point() + geom_smooth(method="lm")
```




```
ggplot(data=ypos, aes(x=PB, y=Return))+ geom_point() + geom_smooth(method="lm")
```



Regression Model

```
mod <- lm(Return ~ PB, data=ypos)
coef_lmbeta <- mod$coefficients[2]
```

Regression Summary Statistics

```
coef(mod)
```

```
## (Intercept)          PB
##  6.57675701 -0.09585723
```

```
summary(mod)
```

```
##
## Call:
## lm(formula = Return ~ PB, data = ypos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -68.052  -4.701   1.416   6.050  58.706
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
```

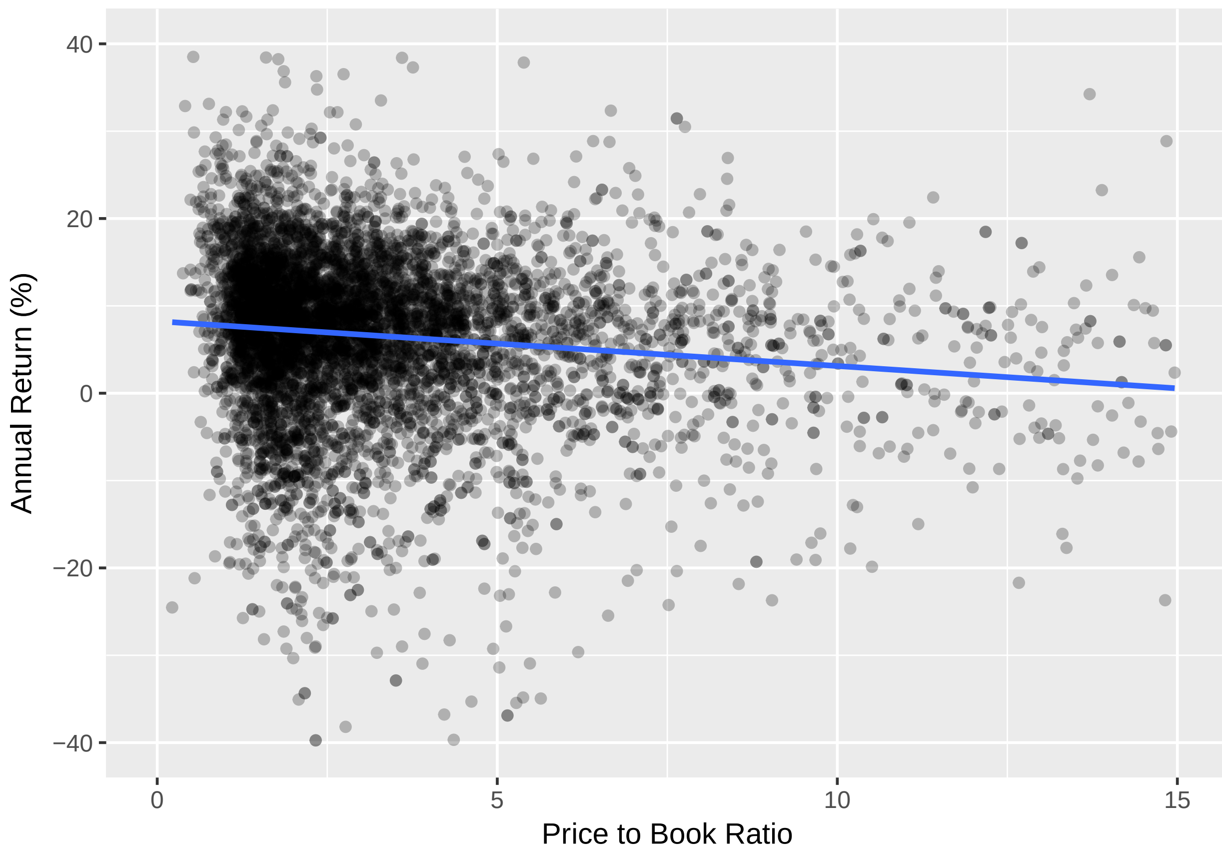
```
## (Intercept)  6.57676    0.15421  42.649  < 2e-16 ***
## PB          -0.09586    0.02042  -4.694 2.74e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.56 on 6366 degrees of freedom
## Multiple R-squared:  0.003449,   Adjusted R-squared:  0.003293
## F-statistic: 22.03 on 1 and 6366 DF,  p-value: 2.736e-06
```

Regression Plot

```r
ggplot(data=ypos, aes(x=PB, y=Return))+ geom_point(alpha=0.25) + geom_smooth(method="lm", se=FALSE) + xl
```

```
## Warning: Removed 161 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 161 rows containing missing values (geom_point).
```

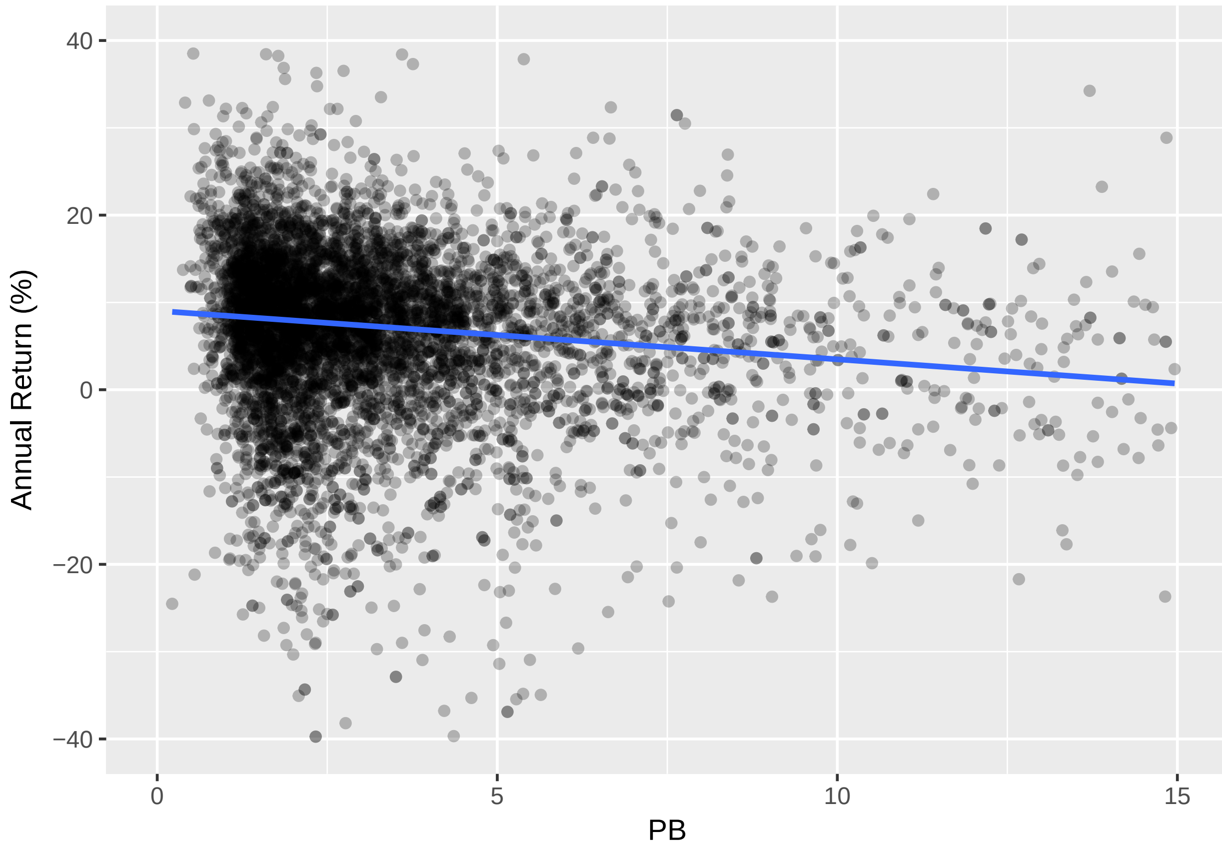


Regression Plot with RLM

```r
ggplot(data=ypos, aes(x=PB, y=Return))+ geom_point(alpha=0.25) + geom_smooth(method="rlm", se=FALSE) + :
```

```
## Warning: Removed 161 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 161 rows containing missing values (geom_point).
```



#Test for Heteroskedasticiy

```
bptest(mod)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mod
## BP = 2.8539, df = 1, p-value = 0.09115
```

```
ds_kurtosis(ypos$Return)
```

```
## [1] 5.102515
```

```
ds_skewness(ypos$Return)
```

```
## [1] -1.236622
```

```
cor.test(ypos$PB, ypos$Return)
```

```
##
##  Pearson's product-moment correlation
##
## data:  ypos$PB and ypos$Return
## t = -4.6939, df = 6366, p-value = 2.736e-06
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.08317071 -0.03421629
## sample estimates:
```

```
##         cor
## -0.05872881
```

Sorting by Cook's distance

```r
mod %>%
  augment()%>%
  arrange(desc(.cooksd)) %>%
  head()
```

```
##   .rownames     Return     PB    .fitted  .se.fit    .resid       .hat
## 1      3638   7.575727 243.48 -16.762561 4.894910  24.33829 0.21478934
## 2      6409  13.580324 141.83  -7.018674 2.820370  20.59900 0.07130756
## 3      5320  30.506626  79.73  -1.065940 1.554724  31.57257 0.02166857
## 4       419  26.166137  86.77  -1.740775 1.698015  27.90691 0.02584677
## 5      4731  10.038719 118.48  -4.780407 2.344158  14.81913 0.04926029
## 6      3490 -17.771589  91.39  -2.183635 1.792091 -15.58795 0.02879011
##     .sigma    .cooksd .std.resid
## 1 10.55703 0.92494043   2.600511
## 2 10.55924 0.15724515   2.023820
## 3 10.55506 0.10115126   3.022237
## 4 10.55669 0.09507551   2.677071
## 5 10.56092 0.05364290   1.438976
## 6 10.56078 0.03324212  -1.497595
```

Sorting by Cook's distance olour="Red") + ylab("Annual Return (%)")

```r
rlmmod <- rlm(Return ~ PB, data=ypos)
rlmmod <- lmrob(Return ~ PB, data=ypos)
summary(rlmmod)
```

```
##
## Call:
## lmrob(formula = Return ~ PB, data = ypos)
##  \--> method = "MM"
## Residuals:
##      Min       1Q   Median       3Q      Max
## -69.3539  -5.7075   0.3525   5.0245  57.5968
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.06712    0.29272  27.559  < 2e-16 ***
## PB          -0.21857    0.07805  -2.801  0.00512 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Robust residual standard error: 7.871
## Multiple R-squared:  0.01545,    Adjusted R-squared:  0.0153
## Convergence in 35 IRWLS iterations
##
## Robustness weights:
```

```
##  68 observations c(163,229,345,416,564,599,857,876,890,969,1043,1153,1175,1187,1275,1349,1459,1481,1
##   are outliers with |weight| = 0 ( < 1.6e-05);
##  557 weights are ~= 1. The remaining 5743 ones are summarized as
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000879 0.8484000 0.9501000 0.8790000 0.9863000 0.9990000
## Algorithmic parameters:
##        tuning.chi                bb        tuning.psi        refine.tol
##         1.548e+00         5.000e-01         4.685e+00         1.000e-07
##           rel.tol         solve.tol       eps.outlier             eps.x
##         1.000e-07         1.000e-07         1.570e-05         4.429e-10
## warn.limit.reject warn.limit.meanrw
##         5.000e-01         5.000e-01
##      nResample         max.it       best.r.s       k.fast.s          k.max
##            500             50              2              1            200
##    maxit.scale      trace.lev            mts     compute.rd fast.s.large.n
##            200              0           1000              0           2000
##                   psi           subsampling                   cov
##            "bisquare"         "nonsingular"         ".vcov.avar1"
## compute.outlier.stats
##                  "SM"
## seed : int(0)
```