Project - STAT 151A

Aditya Jhanwar

12/8/2019

```
load(url("http://www.stat.berkeley.edu/users/nolan/data/baseball2012.rda"))
baseball = as_tibble(baseball)
```

Data Exploration and Feature Creation

1)

The first step is to clean the `baseball` data by removing unecessary explanatory variables and entries missing a `salary (observed Yi)` value.

```
baseball = baseball %>% dplyr::select(-c("ID", "yearID", "teamID", "lgID", "nameFirst", "nameLast", "G_
baseball = baseball %>% drop_na(salary) # remove units with no salary values
```

Next, I followed the author's process in creating new features as decribed in the textbook.

```
baseball = baseball %>% mutate(AVG = CH/CAB,
                               OBP = 100*(CH + CBB)/(CAB + CBB),
                               CAB.avg  = CAB/years,
                               CH.avg   = CH/years,
                               CHR.avg  = CHR/years,
                               CR.avg   = CR/years,
                               CRBI.avg = CRBI/years)
```

Finally, I cleaned the `Position` and `Years` explanatory variables through reimpementing them as dummy variables.

According to Fox's description of his analysis, he mentions **middle infielders** as players who consistently played second base or shortstop so I classified all individuals with either position as such.

```
new.pos = c()
MI = c("12", "1S", "23", "2B", "2S", "3S", "O2", "SS")
C  = c("C", "C1", "OC")
CF = c("CF")

# Assign new factor assignment for MI (middle infielders), C (catcher), CF (center field), and O (other
for (i in baseball$POS){
  if      (i %in% MI) { new.pos = c(new.pos, "MI") }
  else if (i %in% C)  { new.pos = c(new.pos, "C")  }
  else if (i %in% CF) { new.pos = c(new.pos, "CF") }
  else                { new.pos = c(new.pos, "O")  }
}

baseball$POS = relevel(factor(new.pos), "O")
```

```r
new.years = c()
neg.cont  = 6
neg.sal   = 3

for (i in baseball$years){
  if      (i >= neg.cont) { new.years = c(new.years, ".cont") }
  else if (i >= neg.sal)  { new.years = c(new.years, ".sal")   }
  else                    { new.years = c(new.years, "other")        }
}

baseball$neg = relevel(factor(new.years), "other")
```

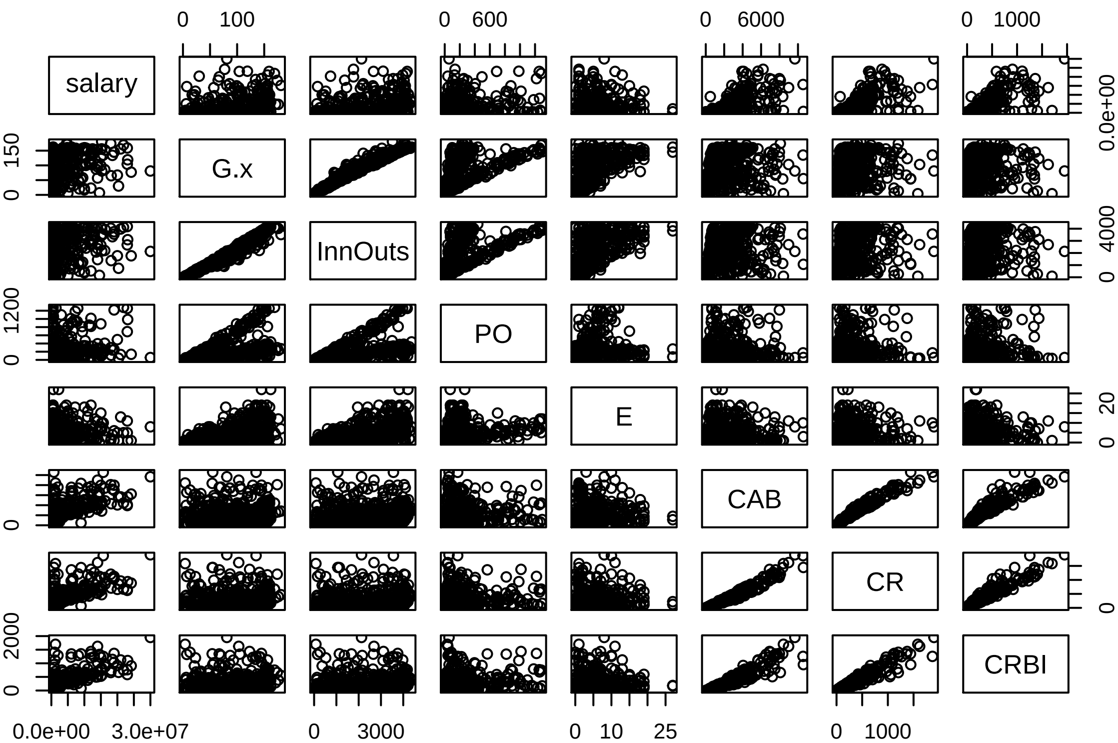
```r
lm.fit = lm(salary ~ ., data = baseball)
new.baseball = as_tibble(model.matrix(lm.fit)[,-1])
new.baseball$salary = baseball$salary
```

Now that we have completed the feature creation process, the next step is to analyze the data itself.

Firstly, I'll look at the the structure of the data itself and how the different variables are associated with each other. Since there are a lot of explanatory variables within the data, I will select a few key variables I believe to be the most influential in the model and investigate the structure.

Note: - `G.x` = Position played at specified position - `InnOut` = Time played in the field expressed as outs - `PO` = Putouts - `E` = Errors - `CAB` = Career at bats - `CR` = Career runs - `CRBI` = Career runs batted in

```
pairs(salary ~ G.x + InnOuts + PO + E + CAB + CR + CRBI, data=new.baseball)
```



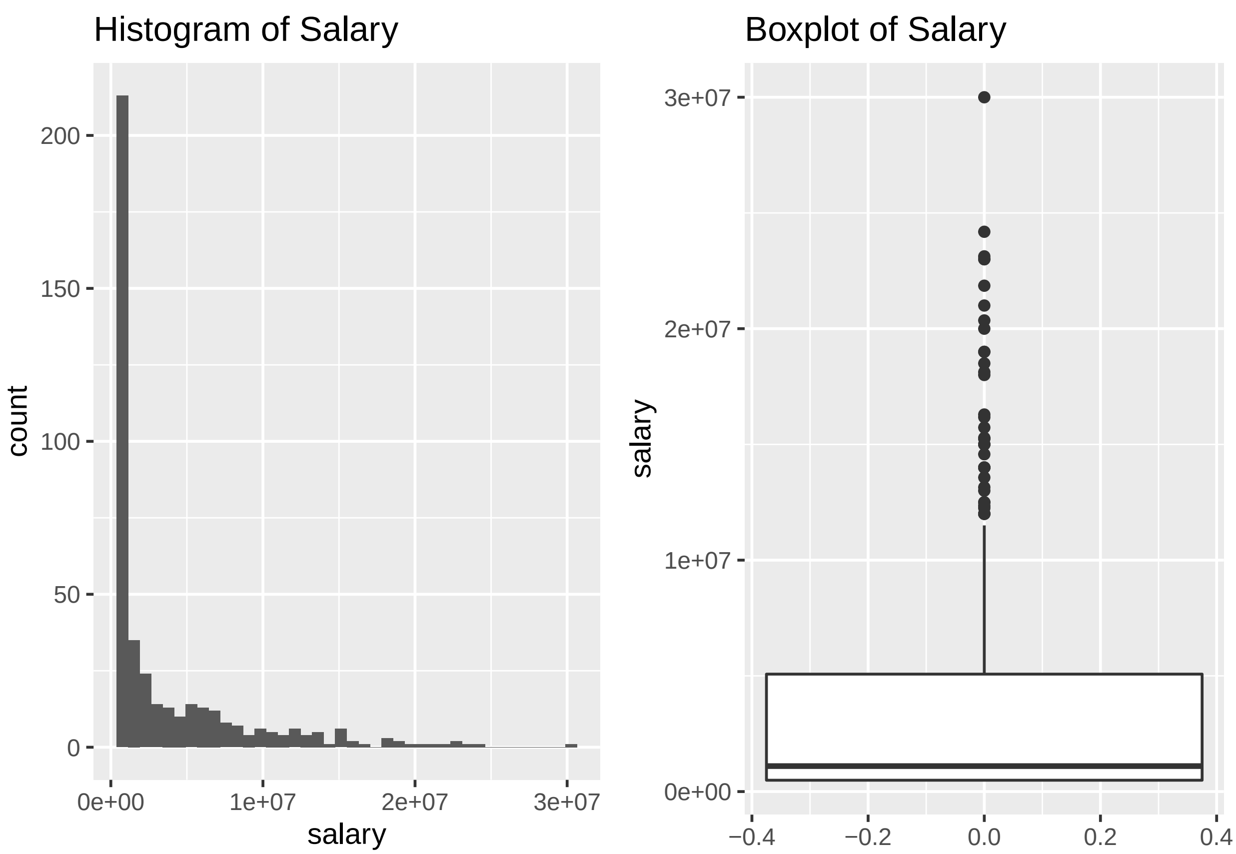
From observing the paired structures of data it is evident that some features are uncorrelated whereas others are strongly correlated. However, this is mostly expected as certain features relate to one another. For example, a player's career at bats would be associated with his career runs or career runs batted in since all tie into a players capability of scoring bases.

This indicates a possible issue in inference of coefficients through linear modeling since the standard error calculation will be grossly inflated.

In addition, I noticed some of the variables have a stronger correlation with the salary than that of other variables. For example, `G.x` and `InnOuts` do not seem to have a strong association with `salary` whereas `CAB`, `CR`, and `CRBI` have comparitively stronger correlations with `salary`. This indicates some sort of variable selection and model pruning may be of benefit.

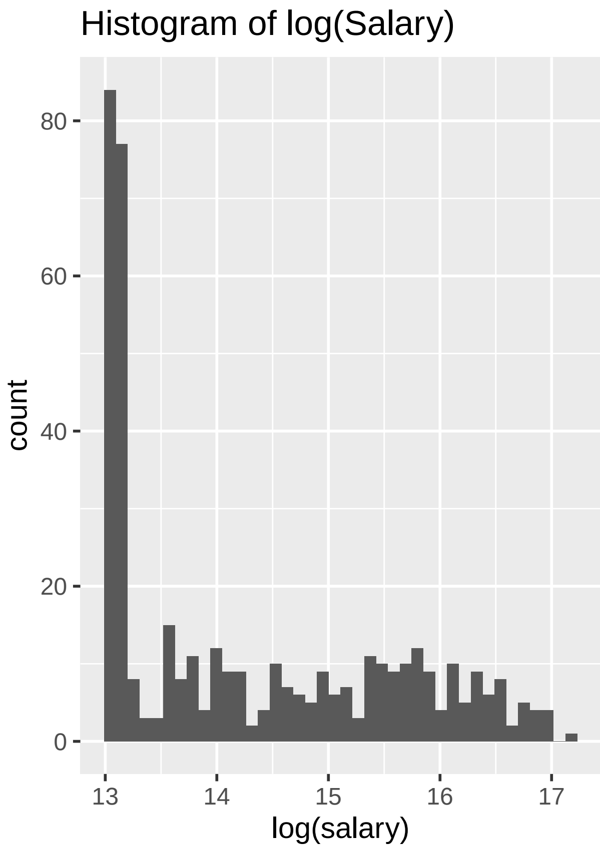
Next, I'd like to look into whether the data is distributed normally as per an assumption of guassian distributed errors in linear modelling.

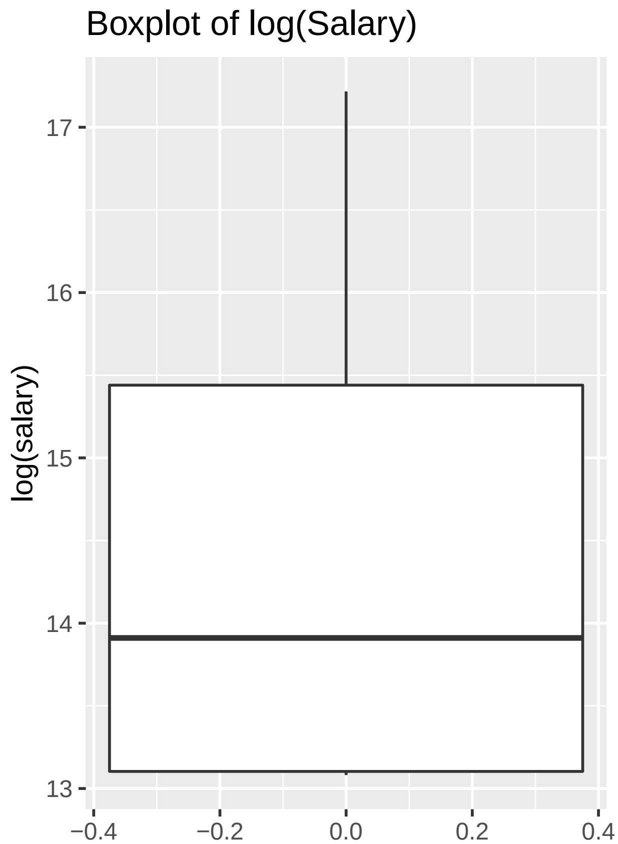
```r
sal1 = ggplot(data=new.baseball, aes(x=salary)) + geom_histogram(bins=40) + ggtitle("Histogram of Salar
sal2 = ggplot(data=new.baseball, aes(y=salary)) + geom_boxplot() + ggtitle("Boxplot of Salary")
grid.arrange(sal1, sal2, nrow=1)
```



The histogram above shows that the observed outcome values are not distributed normally at all. Hence, some sort of transformation of the data is necessary in order to use linear modelling.

```r
log.sal1 = ggplot(data=new.baseball, aes(x=log(salary))) + geom_histogram(bins=40) + ggtitle("Histogram
log.sal2 = ggplot(data=new.baseball, aes(y=log(salary))) + geom_boxplot() +  ggtitle("Boxplot of log(Sa
grid.arrange(log.sal1, log.sal2, nrow=1)
```

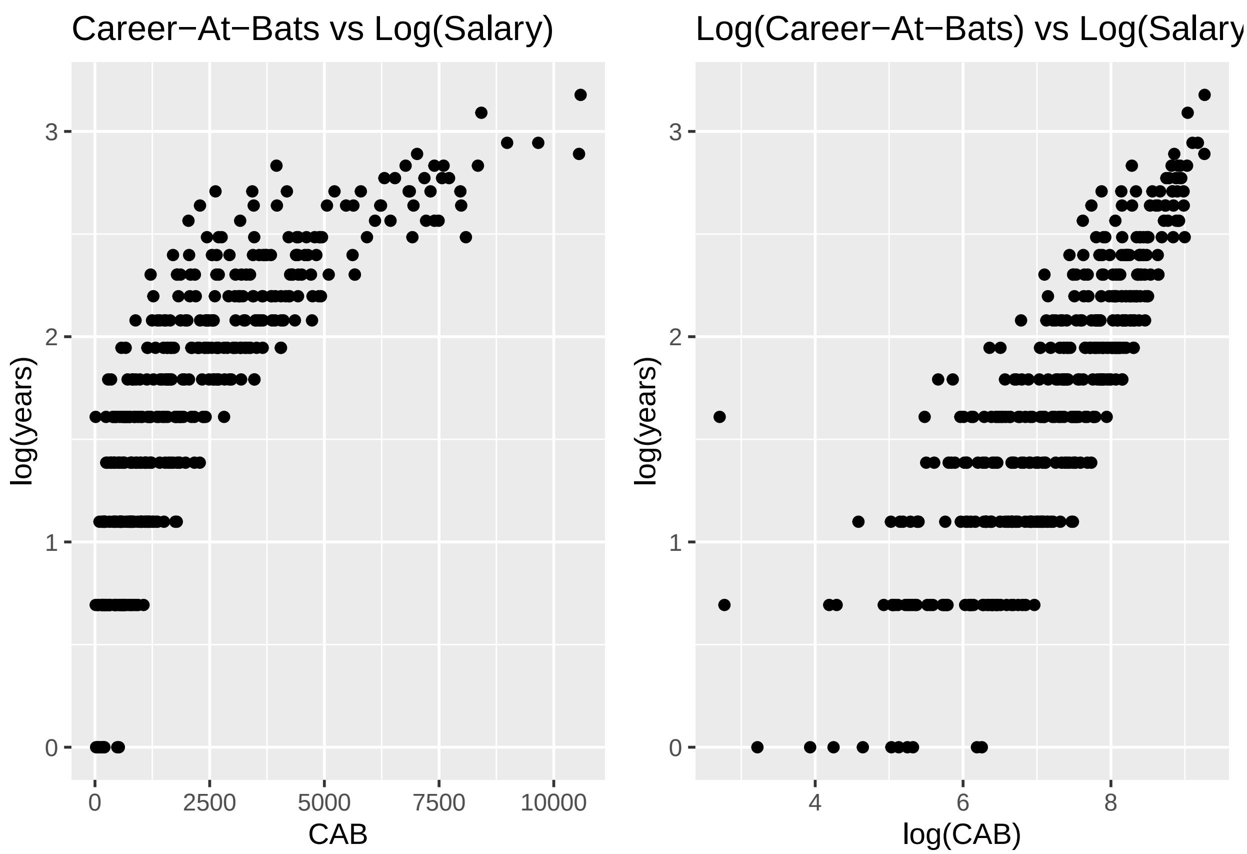




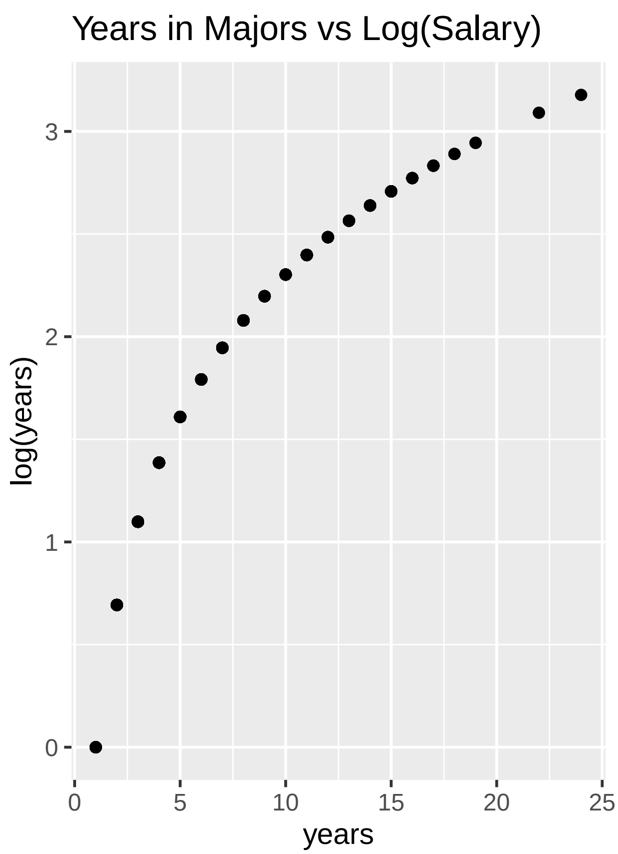
Fox mentions log transforming the `salary` data in his linear modelling analysis and this is in line with the observed histograms above. The histogram of the original salaries is right skewed whereas the histogram of the log transformed salaries is somewhat more stabilized and appears more so normally distributed in some sense.

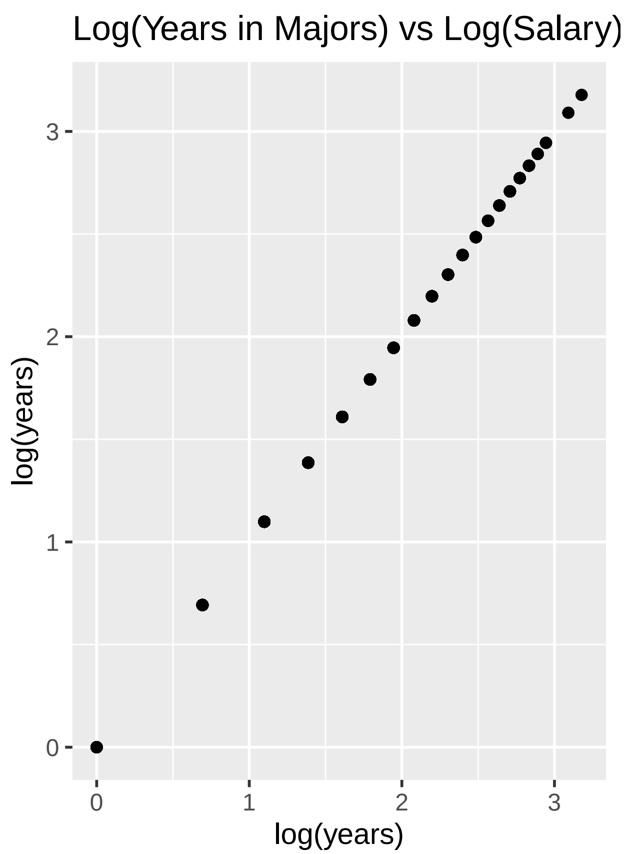
Fox also suggests log transforming some feature variables (years in the majors, career-at-bats) through preliminary examination and so I will do the same to carry forward analysis in a similar manner. An argument for why this may be beneficial is that it might garner a stronger linear relationship between the seemingly most influential explanatory variables and the salary and thus improving the model's predictive capability overall.

```
cab.plot1 = ggplot(data=new.baseball, aes(x=CAB, y=log(years))) + geom_point() + ggtitle(label = "Caree
cab.plot2 = ggplot(data=new.baseball, aes(x=log(CAB), y=log(years))) + geom_point() + ggtitle(label = "
grid.arrange(cab.plot1, cab.plot2, nrow=1)
```




```
yrs.plot1 = ggplot(data=new.baseball, aes(x=years, y=log(years))) + geom_point() + ggtitle(label = "Yea
yrs.plot2 = ggplot(data=new.baseball, aes(x=log(years), y=log(years))) + geom_point() + ggtitle(label =
grid.arrange(yrs.plot1, yrs.plot2, nrow=1)
```






```r
new.baseball$log.CAB    = log(new.baseball$CAB)
new.baseball$log.years  = log(new.baseball$years)
new.baseball$log.salary = log(new.baseball$salary)
new.baseball = new.baseball %>% dplyr::select(-c(CAB, years, salary))
```

Data Analysis

1)

For the first of the project, I will be fitting a simple model that predicts `log(salary)` from the **dummy variables** for `years in majors` and `log(career runs)`, allowing for an interactio between the feature variables.

```
dat1 = new.baseball %>% dplyr::select(log.salary, CR, neg.sal, neg.cont)
simple.model = lm(log.salary ~ log(1+CR)*(neg.cont + neg.sal), data=dat1)
simple.model
```

```
Call:
lm(formula = log.salary ~ log(1 + CR) * (neg.cont + neg.sal),
    data = dat1)

Coefficients:
         (Intercept)           log(1 + CR)              neg.cont
            12.88064               0.08743              -3.96431
             neg.sal  log(1 + CR):neg.cont   log(1 + CR):neg.sal
            -1.06871               0.94036               0.24940
```

2)

Although I have fitted the simple model above, I want to check for any outliers, high leverage points, and influential observations for further evaulation of the simple model. All criterions in determining such observations will be in line with what Fox suggests using.

```
hat.vals = hatvalues(simple.model)
stud.res = studres(simple.model)
cook.dis = cooks.distance(simple.model)

measures = tibble(Hat.Values=hat.vals, Studentized.Residuals=stud.res, Cooks.Distance=cook.dis)
```

First, I'd like to take a look at the **high leverage** points, which are observations with explanatory variables markedly different from that of the average. In terms of numerical cutoffs for diagnostic statistics, *hat values exceeding* ***twice*** *the average hat value* `(k+1)/n` *are noteworthy.*

```
h.3 = 3*length(simple.model$coefficients)/nrow(new.baseball)
high.leverage = measures[hat.vals > h.3,]
high.leverage
```

```
# A tibble: 17 x 3
   Hat.Values Studentized.Residuals Cooks.Distance
        <dbl>                 <dbl>          <dbl>
 1     0.0477                1.87       0.0289
 2     0.0486                0.525      0.00235
 3     0.0565                0.0299     0.00000898
 4     0.0428               -0.305      0.000695
 5     0.0733                0.0664     0.0000582
 6     0.0507               -0.267      0.000637
```

```
 7      0.0428                  -0.305      0.000694
 8      0.0733                   0.0664     0.0000582
 9      0.0437                   0.475      0.00172
10      0.0554                   0.606      0.00359
11      0.126                    0.163      0.000643
12      0.0565                   0.0284     0.00000807
13      0.0489                   0.533      0.00244
14      0.0462                   0.504      0.00206
15      0.0573                   0.564      0.00323
16      0.245                    0.334      0.00606
17      0.148                    1.51       0.0663
```

There appears to be ***17*** data points which have a relatively high leverage.

In addition to high leverage points, I'll analyze discrepant observations to detect outliers within the data through utilizing **studentized residuals** with a numerical cutoff of `|t-test statistic| > 2`

```
outliers = measures[abs(stud.res) > 2,]
outliers %>% head(n=5)
```

```
# A tibble: 5 x 3
  Hat.Values Studentized.Residuals Cooks.Distance
       <dbl>                 <dbl>          <dbl>
1    0.0258                   2.41        0.0254
2    0.00468                 -2.18        0.00370
3    0.0274                   4.12        0.0768
4    0.00804                 -3.01        0.0120
5    0.00476                 -2.54        0.00510
```

There are ***26*** observations which are determined to be outliers.

Although I have determined observations that have high leverage or are outliers, what I am most conerned about are the subset of these points which have an influence on the determined coefficients of the model. Such points greatly alter the predictive capability of the simple model and thus cannot be overlooked.

Through recommendation by Fox, the criterion I will be using to determine highly influential points is `D_i > 4/(n-k-1)`

```
cook.cutoff = 4/(nrow(new.baseball)-length(simple.model$coefficients))
influential.points = measures[cook.dis > cook.cutoff,]
influential.points %>% arrange(desc(Cooks.Distance))
```

```
# A tibble: 16 x 3
   Hat.Values Studentized.Residuals Cooks.Distance
        <dbl>                 <dbl>          <dbl>
 1    0.0274                   4.12         0.0768
 2    0.148                    1.51         0.0663
 3    0.0189                  -4.27         0.0563
 4    0.0211                  -3.63         0.0462
 5    0.0149                  -3.53         0.0306
 6    0.0477                   1.87         0.0289
 7    0.0268                   2.50         0.0285
 8    0.0122                  -3.64         0.0264
 9    0.0258                   2.41         0.0254
```

```
10     0.0255                     2.40         0.0249
11     0.0272                     2.22         0.0227
12     0.0112                    -3.06         0.0173
13     0.0149                    -2.49         0.0155
14     0.00804                   -3.01         0.0120
15     0.0256                     1.64         0.0118
16     0.0189                     1.86         0.0111
```

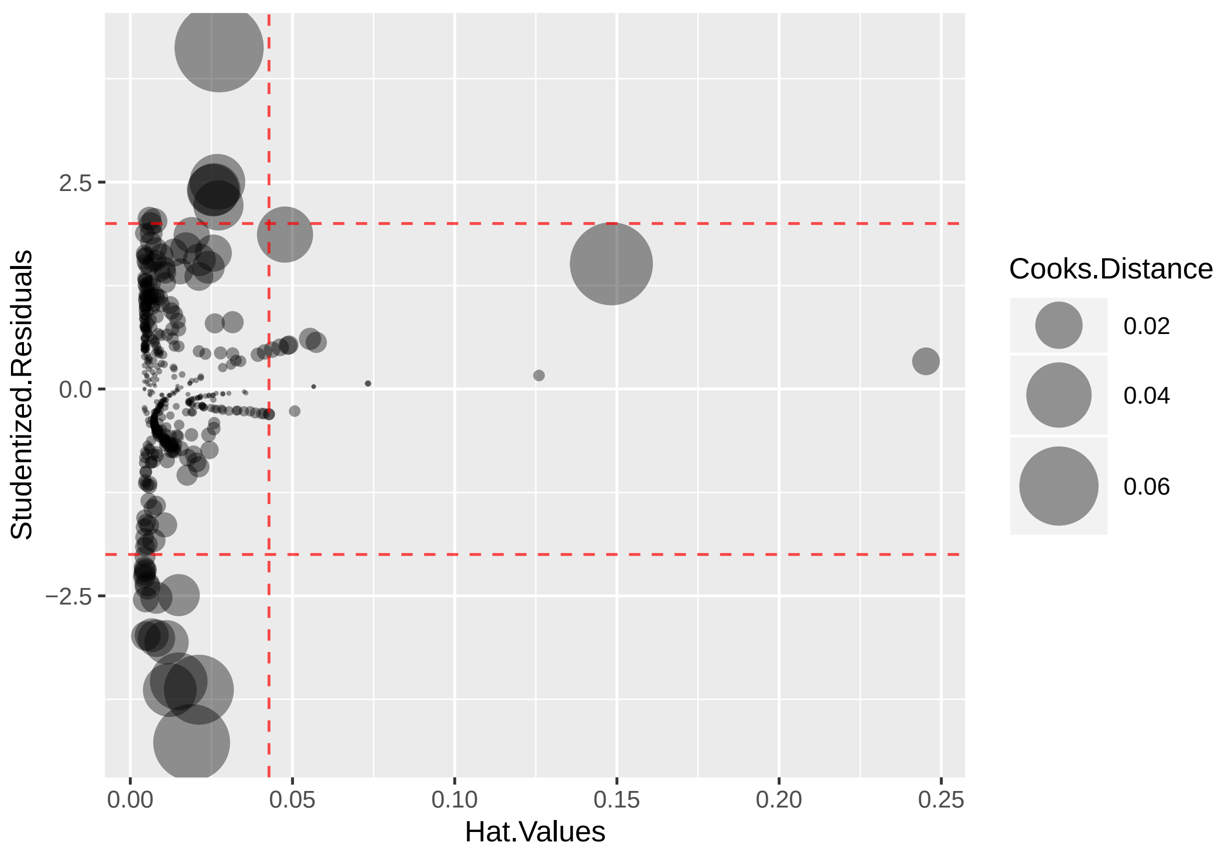
It appears there are ***16*** influential points within the dataset. I'm not very surprised as players' baseball data is incredibly varied and prone to uniquely performing individuals, thus causing there to be influential observations.

To better grasp the idea behind the information produced above, the following is a plot of the `hat` values representing the leverage with relation to the `studentized residuals`. Each circle represents an obervation with it's area proportional to it's calculated `Cook's Distance`.

Note: The horizontal line represents `3 times the average hat value` and the 2 vertical lines mark `t-test statistics of -2 and 2`.

```r
measures = tibble(Hat.Values=hat.vals, Studentized.Residuals=stud.res, Cooks.Distance=cook.dis)

ggplot(aes(x=Hat.Values, y=Studentized.Residuals, size=Cooks.Distance), data=measures) +
  geom_point(alpha=0.4) + scale_size(range=c(0, 15)) +
  geom_vline(xintercept = 3*6/421, color='red', alpha=.7, linetype = "dashed") +
  geom_hline(yintercept = -2, color='red', alpha=.7, linetype = "dashed") +
  geom_hline(yintercept = 2, color='red', alpha=.7, linetype = "dashed")
```



3)

```r
all.fit = lm(log.salary ~ ., data = new.baseball)
summary(all.fit)
```

```

Call:
lm(formula = log.salary ~ ., data = new.baseball)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.63762 -0.31246  0.06423  0.31947  1.51943 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) 14.2702482  0.5240182  27.232  < 2e-16 ***
POSC        -0.0426637  0.1518434  -0.281 0.778886    
POSCF       -0.0268705  0.1494435  -0.180 0.857403    
POSMI        0.0780726  0.1027617   0.760 0.447882    
G.x          0.0039647  0.0046887   0.846 0.398321    
GS          -0.0062069  0.0109901  -0.565 0.572567    
InnOuts      0.0001452  0.0004900   0.296 0.767175    
PO           0.0002185  0.0003197   0.683 0.494768    
A            0.0003503  0.0008990   0.390 0.697023    
E            0.0026430  0.0104617   0.253 0.800687    
DP          -0.0042697  0.0029331  -1.456 0.146306    
G.y         -0.0118007  0.0033350  -3.538 0.000453 ***
AB           0.0065267  0.0016068   4.062 5.92e-05 ***
R           -0.0093742  0.0063621  -1.473 0.141465    
H           -0.0065391  0.0048988  -1.335 0.182729    
X2B         -0.0063059  0.0070018  -0.901 0.368367    
X3B          0.0006723  0.0184037   0.037 0.970877    
HR          -0.0102874  0.0121474  -0.847 0.397599    
RBI          0.0039993  0.0057956   0.690 0.490580    
SB          -0.0013153  0.0058889  -0.223 0.823382    
CS          -0.0276728  0.0185481  -1.492 0.136549    
BB           0.0056713  0.0039142   1.449 0.148192    
SO          -0.0012819  0.0019363  -0.662 0.508337    
IBB          0.0093704  0.0136217   0.688 0.491937    
HBP          0.0036525  0.0110282   0.331 0.740680    
SH           0.0093553  0.0161609   0.579 0.563012    
SF           0.0014463  0.0204969   0.071 0.943783    
GIDP        -0.0086423  0.0093884  -0.921 0.357889    
CH           0.0021802  0.0010424   2.091 0.037157 *  
CHR          0.0076434  0.0038205   2.001 0.046147 *  
CR          -0.0022325  0.0016753  -1.333 0.183480    
CRBI        -0.0026771  0.0018308  -1.462 0.144502    
CBB         -0.0011839  0.0005733  -2.065 0.039609 *  
AVG          0.5156185  3.3598388   0.153 0.878113    
OBP         -0.0025942  0.0291738  -0.089 0.929190    
CAB.avg     -0.0042031  0.0030973  -1.357 0.175589    
CH.avg      -0.0133749  0.0133111  -1.005 0.315641    
CHR.avg     -0.0709547  0.0361205  -1.964 0.050219 .  
CR.avg       0.0534385  0.0151076   3.537 0.000455 ***
```

```
CRBI.avg     0.0469204  0.0166307   2.821 0.005036 **
neg.cont     1.2416258  0.2315305   5.363 1.43e-07 ***
neg.sal      0.0541843  0.1522766   0.356 0.722168
log.CAB     -0.3434284  0.1713702  -2.004 0.045783 *
log.years    0.2802810  0.2404271   1.166 0.244447
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5807 on 377 degrees of freedom
Multiple R-squared:  0.8082,    Adjusted R-squared:  0.7863
F-statistic: 36.95 on 43 and 377 DF,  p-value: < 2.2e-16
```

From former analysis I am aware of collinearity within the explanatory variables and hence inflated standard errors. However, it appears several of the coefficients are statistically significant and that the omnibus F-statistic for all explanatory variables in the model having coefficients of 0 is also statistically significant. In addition, the adjusted R-squared value is significantly lower than R-squared. This all leans toward the argument of possible variable selection that may improve the model's regression ability.

4)

5)

6)

7)