Logistic-Checks & Diagnostics

Charlie Qu

October 13, 2017

Preliminaries

This writing explores dataset "colledge". Create the variable `Elite` by binning the `Top10perc` variable. We are going to divide universities into two groups based on whether or not the proportion of students coming from the top 10% of their high school classes exceeds 50 %. We will also save the College names as a new variable and remove `Accept` and `Enroll` as temporally they occur after applying, and do not make sense as predictors in future data.

```
data(College)
College = College %>%
  mutate(college = rownames(College)) %>%
  mutate(Elite = factor(Top10perc > 50)) %>%
  mutate(Elite =
           dplyr::recode(Elite, 'TRUE' = "Yes", 'FALSE'="No")) %>%
  dplyr::select(c(-Accept, -Enroll))
```

We are going to create a training and test set by randomly splitting the data. First set a random seed by

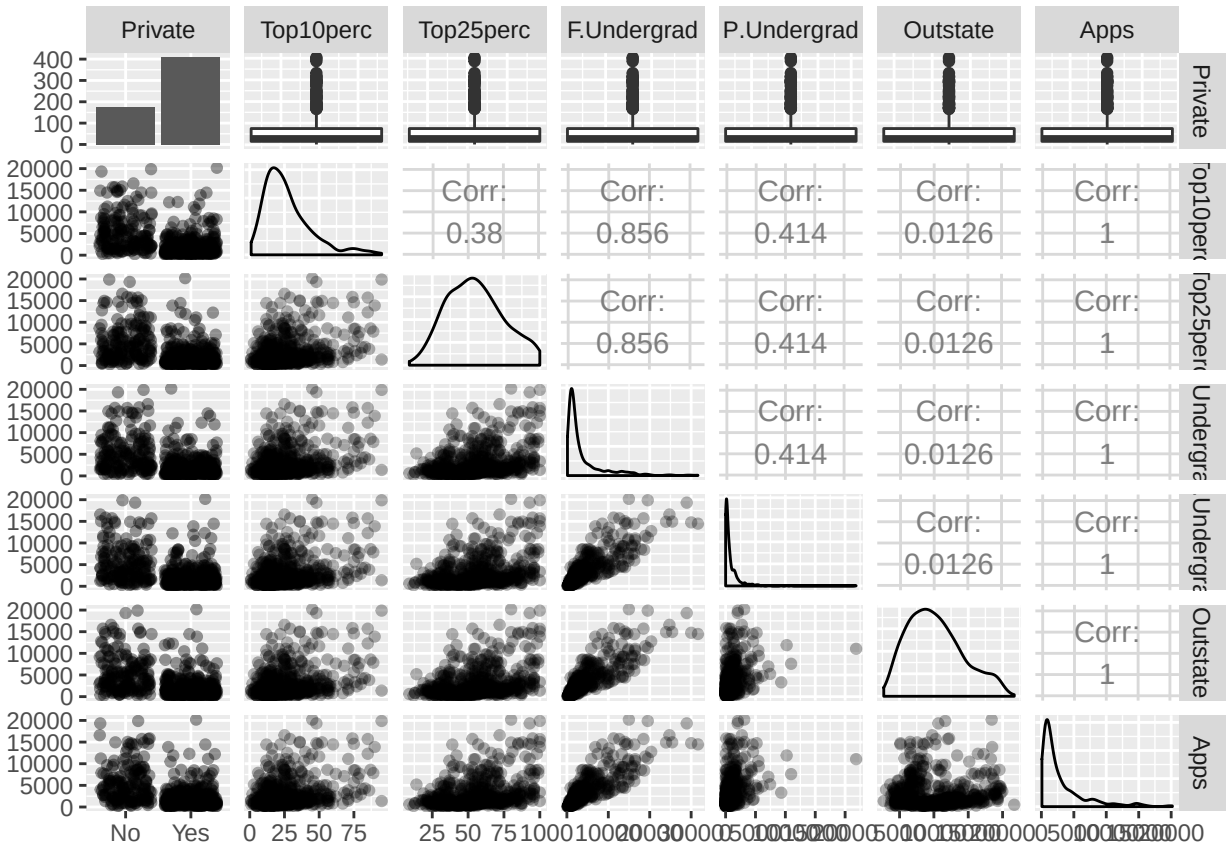
```
# do not change this; for a break google `8675309`
set.seed(8675309)
n = nrow(College)
n.train = floor(.75*n)
train = sample(1:n, size=n.train, replace=FALSE)
College.train = College[train,]
College.test = College[-train,]
```

1. Create scatter plots of predictors versus `Apps` using the training data only. If you use pairs or preferably `ggpairs` make sure that `Apps` is on the y-axis in plots versus the other predictors. (Make sure that the plots are legible, which may require multiple plots.)
   Comment on any features in the plots, such as potential outliers, non-linearity, needs for transformations etc.

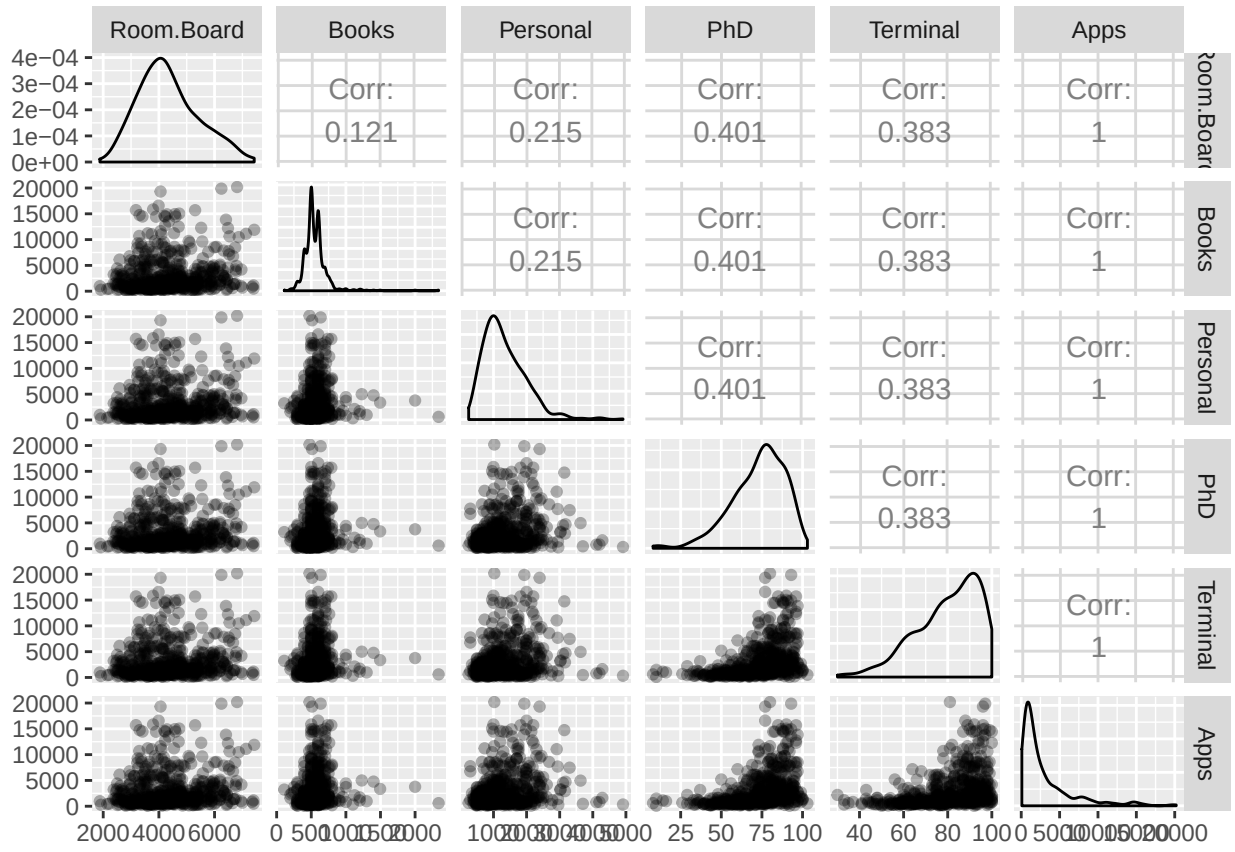
From the plots we can see potential outliers exist when predictors are Top10perc, Top25perc, P.Undergrad, Outstate, Books, Personal, S.F.Ratio, perc.alumni, Expend, Grad.Rate; Apps might only have linear relationship with F.Undergrad, there seems no linear relationship between Apps and the rest of the variables; Apps, Top10perc, F.Undergrad, P.Undergrad, Books, Personal, PhD, Terminal, S.F.Ratio, Expend have skewness, so it's better to transform them to log form.

```
##before remove the college, preserve a data frame to find outliers
College.train.preserve=College.train
##remove "college" in a seperate chunk to avoid repeat
College.train=College.train %>%
  dplyr::select(-c(college))
```

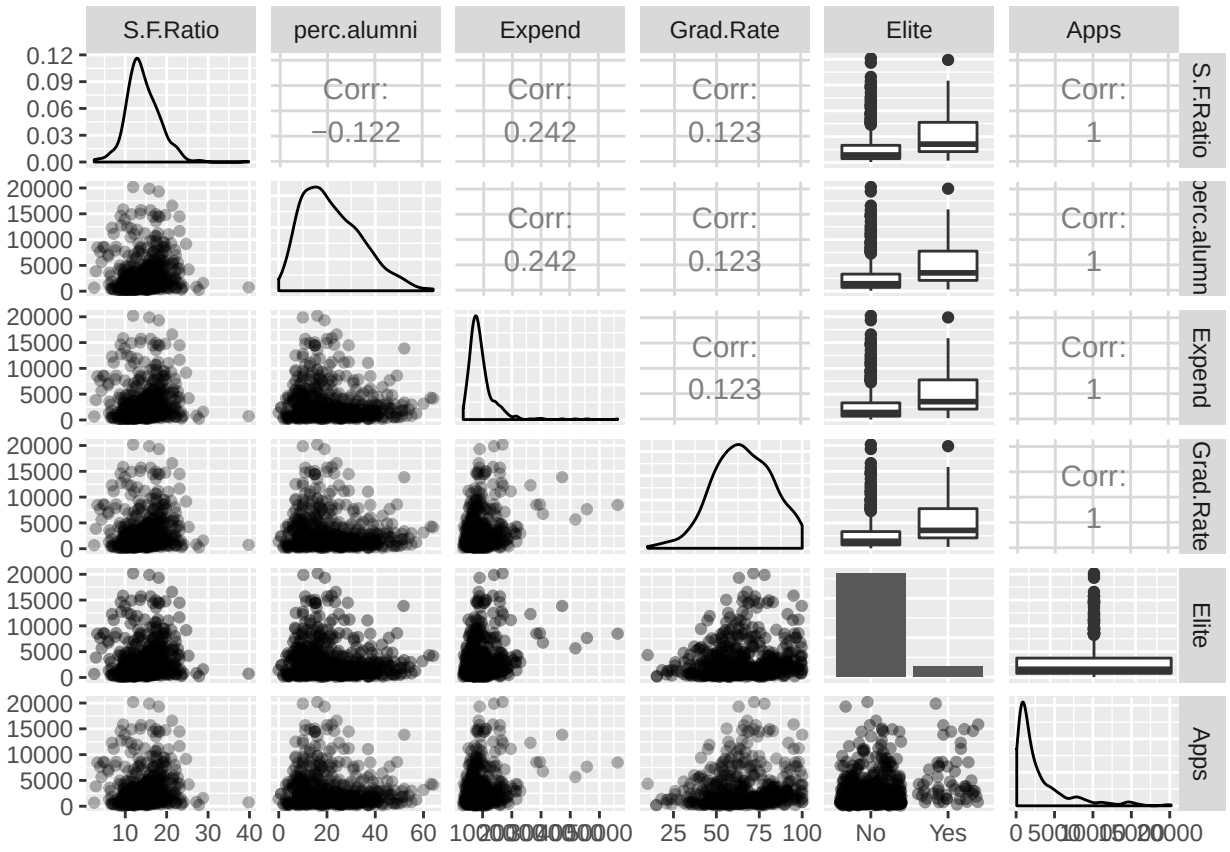
```
ggpairs(data = College.train[c( 1,3:7,2)],
        mapping = ggplot2::aes(y = Apps),
        lower = list(continuous = wrap("points", alpha = 0.3), combo = wrap("dot_no_facet", alpha = 0.4
```




```
ggpairs(College.train[c( 8:12,2)],
        mapping = ggplot2::aes(y = Apps),
        lower = list(continuous = wrap("points", alpha = 0.3), combo = wrap("dot_no_facet", alpha = 0.4
```




```
ggpairs(College.train[c( 13:17,2)],
        mapping = ggplot2::aes(y = Apps),
        lower = list(continuous = wrap("points", alpha = 0.3), combo = wrap("dot_no_facet", alpha = 0.4
```



We can summarize the training and test data as follows

```
numeric variables: `Apps`, `Top10perc`, `Top25perc`, `F.Undergrad`, `P.Undergrad`, `Outstate`, `Room.Boa
factors: `Private`, `Elite`
Since the data is splitted randomly, summaries appear to be similar across the test and training data.
```

```
lapply(College,class)
```

```
## $Private
## [1] "factor"
##
## $Apps
## [1] "numeric"
##
## $Top10perc
## [1] "numeric"
##
## $Top25perc
## [1] "numeric"
##
## $F.Undergrad
## [1] "numeric"
##
## $P.Undergrad
## [1] "numeric"
##
## $Outstate
```

```
## [1] "numeric"
##
## $Room.Board
## [1] "numeric"
##
## $Books
## [1] "numeric"
##
## $Personal
## [1] "numeric"
##
## $PhD
## [1] "numeric"
##
## $Terminal
## [1] "numeric"
##
## $S.F.Ratio
## [1] "numeric"
##
## $perc.alumni
## [1] "numeric"
##
## $Expend
## [1] "numeric"
##
## $Grad.Rate
## [1] "numeric"
##
## $college
## [1] "character"
##
## $Elite
## [1] "factor"
```

```
summary(College)
```

```
##  Private        Apps         Top10perc       Top25perc      F.Undergrad
##  No :212   Min.   :   81   Min.   : 1.00   Min.   :  9.0   Min.   :  139
##  Yes:565   1st Qu.:  776   1st Qu.:15.00   1st Qu.: 41.0   1st Qu.:  992
##            Median : 1558   Median :23.00   Median : 54.0   Median : 1707
##            Mean   : 3002   Mean   :27.56   Mean   : 55.8   Mean   : 3700
##            3rd Qu.: 3624   3rd Qu.:35.00   3rd Qu.: 69.0   3rd Qu.: 4005
##            Max.   :48094   Max.   :96.00   Max.   :100.0   Max.   :31643
##   P.Undergrad         Outstate       Room.Board       Books
##  Min.   :    1.0   Min.   : 2340   Min.   :1780   Min.   :  96.0
##  1st Qu.:   95.0   1st Qu.: 7320   1st Qu.:3597   1st Qu.: 470.0
##  Median :  353.0   Median : 9990   Median :4200   Median : 500.0
##  Mean   :  855.3   Mean   :10441   Mean   :4358   Mean   : 549.4
##  3rd Qu.:  967.0   3rd Qu.:12925   3rd Qu.:5050   3rd Qu.: 600.0
##  Max.   :21836.0   Max.   :21700   Max.   :8124   Max.   :2340.0
##     Personal          PhD            Terminal       S.F.Ratio
##  Min.   : 250   Min.   :  8.00   Min.   : 24.0   Min.   : 2.50
##  1st Qu.: 850   1st Qu.: 62.00   1st Qu.: 71.0   1st Qu.:11.50
##  Median :1200   Median : 75.00   Median : 82.0   Median :13.60
```

```
##  Mean   :1341   Mean   : 72.66   Mean   : 79.7   Mean   :14.09
##  3rd Qu.:1700   3rd Qu.: 85.00   3rd Qu.: 92.0   3rd Qu.:16.50
##  Max.   :6800   Max.   :103.00   Max.   :100.0   Max.   :39.80
##   perc.alumni        Expend        Grad.Rate        college
##  Min.   : 0.00   Min.   : 3186   Min.   : 10.00   Length:777
##  1st Qu.:13.00   1st Qu.: 6751   1st Qu.: 53.00   Class :character
##  Median :21.00   Median : 8377   Median : 65.00   Mode  :character
##  Mean   :22.74   Mean   : 9660   Mean   : 65.46
##  3rd Qu.:31.00   3rd Qu.:10830   3rd Qu.: 78.00
##  Max.   :64.00   Max.   :56233   Max.   :118.00
##  Elite
##  No :699
##  Yes: 78
##
##
##
##
```

Potential outliers:

```
which(College.train.preserve$F.Undergrad>27500)
```

```
## [1] 114 192 201
```

```
College.train.preserve[c(114,192,201),]$college
```

```
## [1] "Texas A&M Univ. at College Station"
## [2] "University of Texas at Austin"
## [3] "Pennsylvania State Univ. Main Campus"
```

```
which(College.train.preserve$P.Undergrad>15000)
```

```
## [1] 22
```

```
College.train.preserve[22,]$college
```

```
## [1] "University of Minnesota Twin Cities"
```

```
which(College.train.preserve$Books>1500)
```

```
## [1] 307 560
```

```
College.train.preserve[c(307,560),]$college
```

```
## [1] "Bradley University"          "Center for Creative Studies"
```

```
which(College.train.preserve$Personal>4500)
```

```
## [1] 557
```

```
College.train.preserve[557,]$college
```

```
## [1] "MidAmerica Nazarene College"
```

```
which(College.train.preserve$S.F.Ratio>35)
```

```
## [1] 89
```

```
College.train.preserve[89,]$college
```

```
## [1] "Indiana Wesleyan University"
```

```r
which(College.train.preserve$Expend>50000)
```

```
## [1] 244
```

```r
College.train.preserve[244,]$college
```

```
## [1] "Johns Hopkins University"
```

In summary, we have observed the following potential outliers:

(1)Texas A&M Univ. at College Station, University of Texas at Austin and Pennsylvania State Univ. Main Campus have relatively large numbers of fulltime undergraduates.

(2)University of Minnesota Twin Cities has a relatively large number of parttime undergraduates.

(3)Bradley University and Center for Creative Studies have relatively large estimated book costs.

(4)MidAmerica Nazarene College has relatively large estimated personal spending.

(5)Indiana Wesleyan University has relatively large Student/faculty ratio.

(6)Johns Hopkins University has relatively large instructional expenditure per student.

2. Build a linear regression model to predict `Apps` from the other predictors using the training data. Present model summaries and diagnostic plots. Based on diagnostic plots using residuals, comment on the adequacy of your model.

```r
# remove high correlation
cor(College.train[,c(3:16)])
```

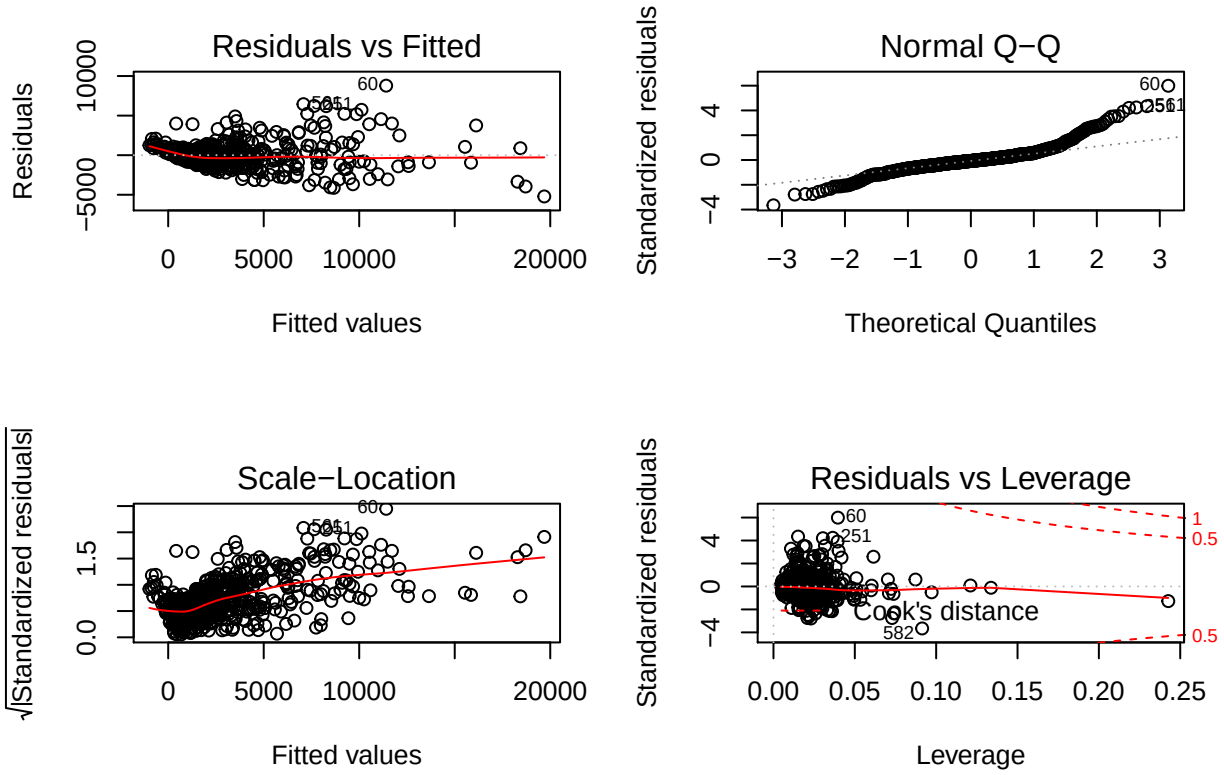
```
##               Top10perc   Top25perc F.Undergrad P.Undergrad    Outstate
## Top10perc    1.00000000  0.89290190  0.14078774 -0.09299651  0.53300272
## Top25perc    0.89290190  1.00000000  0.22059236 -0.01586539  0.47074716
## F.Undergrad  0.14078774  0.22059236  1.00000000  0.55944395 -0.23178987
## P.Undergrad -0.09299651 -0.01586539  0.55944395  1.00000000 -0.24336829
## Outstate     0.53300272  0.47074716 -0.23178987 -0.24336829  1.00000000
## Room.Board   0.34586818  0.31030762 -0.08092952 -0.04319937  0.65302409
## Books        0.11526703  0.10978385  0.09178690  0.05316077  0.04210647
## Personal    -0.11744479 -0.09401784  0.36629164  0.32098842 -0.32543166
## PhD          0.53130876  0.56232715  0.31367152  0.14229114  0.35294106
## Terminal     0.49525699  0.54797913  0.30579229  0.14414053  0.39035314
## S.F.Ratio   -0.35956539 -0.27347755  0.28592870  0.20947945 -0.54547715
## perc.alumni  0.44267397  0.40246423 -0.24753316 -0.24842807  0.58619684
## Expend       0.63835822  0.51943558  0.00848223 -0.05529089  0.65869274
## Grad.Rate    0.49555872  0.48067237 -0.07741119 -0.24179804  0.56340873
##               Room.Board        Books    Personal         PhD    Terminal
## Top10perc    0.34586818  0.115267034 -0.11744479  0.53130876  0.49525699
## Top25perc    0.31030762  0.109783854 -0.09401784  0.56232715  0.54797913
## F.Undergrad -0.08092952  0.091786900  0.36629164  0.31367152  0.30579229
## P.Undergrad -0.04319937  0.053160767  0.32098842  0.14229114  0.14414053
## Outstate     0.65302409  0.042106471 -0.32543166  0.35294106  0.39035314
## Room.Board   1.00000000  0.126300566 -0.21739462  0.32116457  0.38281912
## Books        0.12630057  1.000000000  0.16556403 -0.00503930  0.09291673
## Personal    -0.21739462  0.165564031  1.00000000 -0.02182349 -0.03494596
## PhD          0.32116457 -0.005039300 -0.02182349  1.00000000  0.84117934
## Terminal     0.38281912  0.092916728 -0.03494596  0.84117934  1.00000000
## S.F.Ratio   -0.34291513 -0.043830282  0.17596949 -0.12079898 -0.15264196
## perc.alumni  0.28516468 -0.054969541 -0.30255720  0.24089536  0.26919938
## Expend       0.50180035  0.122503788 -0.13587856  0.41028700  0.42746257
```

```
## Grad.Rate    0.41289387  0.002076388 -0.28029881  0.31646208  0.30596904
##               S.F.Ratio perc.alumni      Expend    Grad.Rate
## Top10perc   -0.35956539  0.44267397  0.63835822  0.495558724
## Top25perc   -0.27347755  0.40246423  0.51943558  0.480672367
## F.Undergrad  0.28592870 -0.24753316  0.00848223 -0.077411189
## P.Undergrad  0.20947945 -0.24842807 -0.05529089 -0.241798036
## Outstate    -0.54547715  0.58619684  0.65869274  0.563408732
## Room.Board  -0.34291513  0.28516468  0.50180035  0.412893872
## Books       -0.04383028 -0.05496954  0.12250379  0.002076388
## Personal     0.17596949 -0.30255720 -0.13587856 -0.280298813
## PhD         -0.12079898  0.24089536  0.41028700  0.316462076
## Terminal    -0.15264196  0.26919938  0.42746257  0.305969038
## S.F.Ratio    1.00000000 -0.41853631 -0.59215142 -0.298736256
## perc.alumni -0.41853631  1.00000000  0.42318853  0.512243905
## Expend      -0.59215142  0.42318853  1.00000000  0.394154273
## Grad.Rate   -0.29873626  0.51224390  0.39415427  1.000000000
```

```
College.train = College.train %>%
  dplyr::select(-c(Top25perc,Outstate,Expend,P.Undergrad,Terminal))

lm_fit1=lm(Apps ~ ., data=College.train)
summary(lm_fit1)
```

```
##
## Call:
## lm(formula = Apps ~ ., data = College.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5213.7  -704.3  -115.4   473.3  8781.0
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.538e+03  6.148e+02  -2.502 0.012638 *
## PrivateYes  -3.914e+02  2.138e+02  -1.830 0.067703 .
## Top10perc    1.453e+01  7.062e+00   2.057 0.040136 *
## F.Undergrad  6.163e-01  1.922e-02  32.075  < 2e-16 ***
## Room.Board   4.497e-01  7.025e-02   6.402  3.2e-10 ***
## Books        2.009e-01  3.705e-01   0.542 0.587974
## Personal    -2.926e-01  1.096e-01  -2.670 0.007810 **
## PhD         -7.785e-01  5.144e+00  -0.151 0.879754
## S.F.Ratio   -2.730e+01  1.893e+01  -1.443 0.149674
## perc.alumni -1.932e+01  6.670e+00  -2.896 0.003921 **
## Grad.Rate    1.802e+01  4.738e+00   3.803 0.000158 ***
## EliteYes     1.161e+03  3.359e+02   3.458 0.000586 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1496 on 570 degrees of freedom
## Multiple R-squared:  0.8165, Adjusted R-squared:  0.8129
## F-statistic: 230.5 on 11 and 570 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(lm_fit1)
```



Based on the first plot, the trend of plots is diverging from left to right, so Apps and some of the predictors need transformation and it also shows increasing variance; from the second plot, the existence of heavy tail tells us the residuals of regression are not normally distributed, or maybe the data have more extreme values than would be expected if they truly came from a Normal distribution. In Scale-Location plot, residuals are diverging from left to right, but are not spreading wider along the range of predictors which is good. From the Residuals vs Leverage plot, although all cases are inside of the Cook's distance lines, points 60, 251, 582 might be influential points.

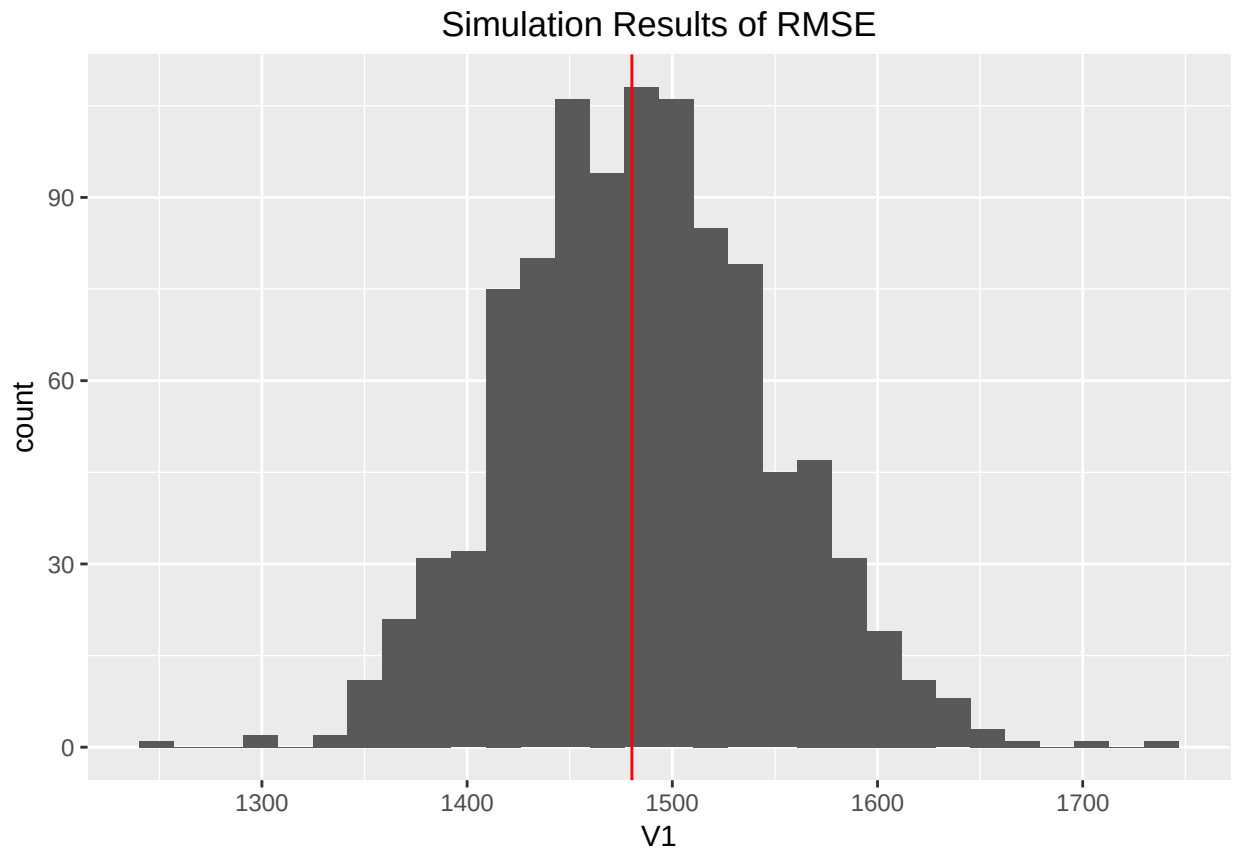
In addition, we remove the variables that have strong correlations according to the correlation matrix.

3. Generate 1000 replicate data sets using the coefficients from the model you fit above. Using RMSE as a statistic, $\sqrt{\sum_i (y^{\text{rep}} - \hat{y}_i^{\text{rep}})^2/n}$, how does the RMSE from the model based on the training data compare to RMSE's based on the replicated data. What does this suggest about model adequacy? Provide a histogram of the RMSE's with a line showing the location of the observed RMSE and compute a p-value. Hint: write a function to calculate RMSE.

```
ct = College.train
rmse = function(y, ypred){
  rmse = sqrt(mean((y-ypred)^2))
  return(rmse)
}


pi.lm = function(model){
  X <- model.matrix(model)
  n_sim <- 1000
  sim_fit <- sim (model, n_sim)
  beta <- sim_fit@coef
```

```
  n <- nrow(College.train)
  y.rep <- array(NA, c(n_sim, n))
  rmse_rep = as.data.frame(matrix(NA, ncol =1, nrow = n_sim))
  for (s in 1:n_sim){
    mu = (X %*% beta[s,])
    y.rep <- rnorm(n, mu , sim_fit@sigma[s])
    ct$Apps = y.rep
    fit<- lm(Apps~., data = ct)
    rmse_rep[s,1] = rmse(ct$Apps, fitted(fit))
  }
  return(rmse_rep)
}

rmse_rep = pi.lm(lm_fit1)

rmse_fit = rmse(College.train$Apps, fitted(lm_fit1))
ggplot(data = rmse_rep, aes(x = V1)) +
  geom_histogram() +
  geom_vline(xintercept =rmse_fit, col = "red") +
  labs(title = "Simulation Results of RMSE") +
  theme(plot.title = element_text(hjust = 0.5))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
summary(lm_fit1)
```

```
##
## Call:
## lm(formula = Apps ~ ., data = College.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5213.7  -704.3  -115.4   473.3  8781.0
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.538e+03  6.148e+02  -2.502 0.012638 *
## PrivateYes  -3.914e+02  2.138e+02  -1.830 0.067703 .
## Top10perc    1.453e+01  7.062e+00   2.057 0.040136 *
## F.Undergrad  6.163e-01  1.922e-02  32.075  < 2e-16 ***
## Room.Board   4.497e-01  7.025e-02   6.402  3.2e-10 ***
## Books        2.009e-01  3.705e-01   0.542 0.587974
## Personal    -2.926e-01  1.096e-01  -2.670 0.007810 **
## PhD         -7.785e-01  5.144e+00  -0.151 0.879754
## S.F.Ratio   -2.730e+01  1.893e+01  -1.443 0.149674
## perc.alumni -1.932e+01  6.670e+00  -2.896 0.003921 **
## Grad.Rate    1.802e+01  4.738e+00   3.803 0.000158 ***
## EliteYes     1.161e+03  3.359e+02   3.458 0.000586 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1496 on 570 degrees of freedom
## Multiple R-squared:  0.8165, Adjusted R-squared:  0.8129
## F-statistic: 230.5 on 11 and 570 DF,  p-value: < 2.2e-16
```

```
p = mean(rmse_rep < rmse_fit)
print(p)
```

```
## [1] 0.482
```

```
summary(rmse_rep)
```

```
##        V1
##  Min.   :1246
##  1st Qu.:1442
##  Median :1483
##  Mean   :1485
##  3rd Qu.:1526
##  Max.   :1736
```

```
rmse_fit
```

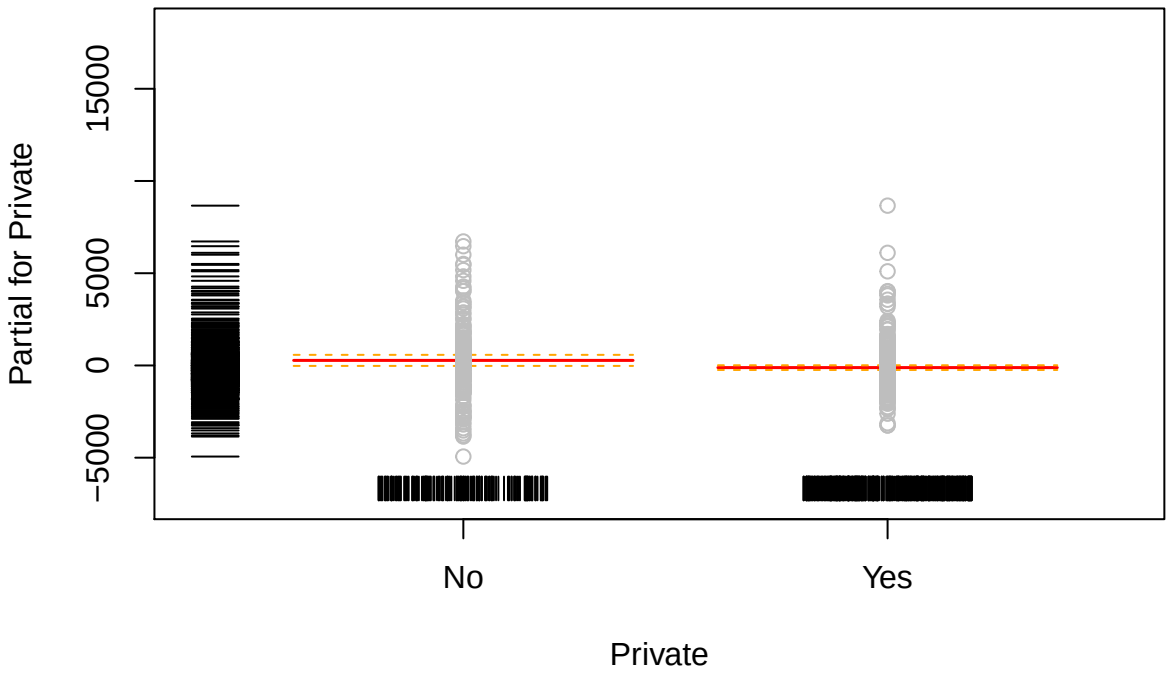
```
## [1] 1480.304
```

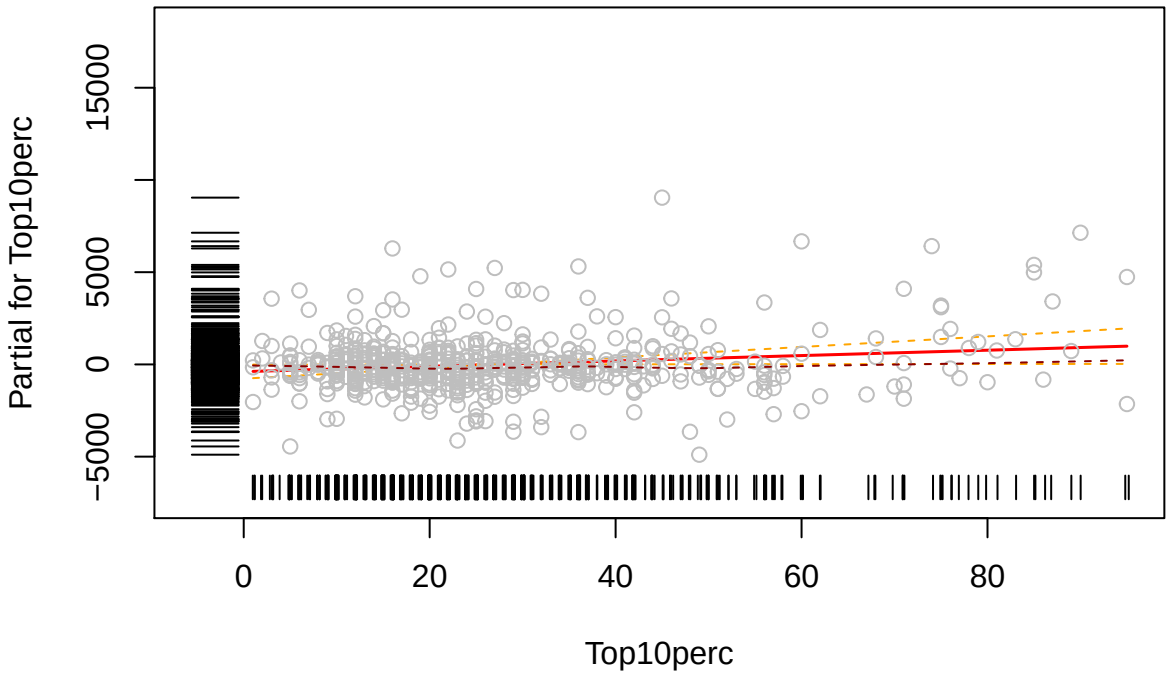
From the histogram we see the distribution of RMSE based on the replicated data is centralized, and the observed RMSE is located near the center of replicated data RMSE distribution, which indicates our regression model based on training data is fitting well.

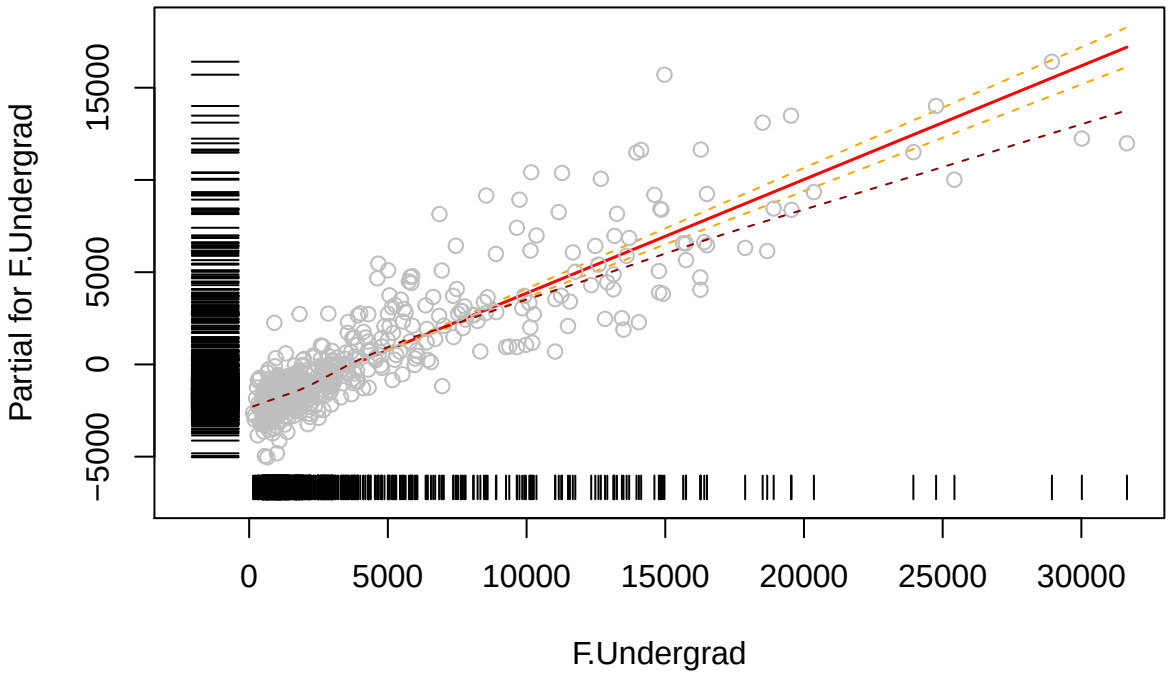
From the summary we can see that the mean of the replicated data sets is 1476 and RMSE of our model is 1480.340. Our first model seems to be good.

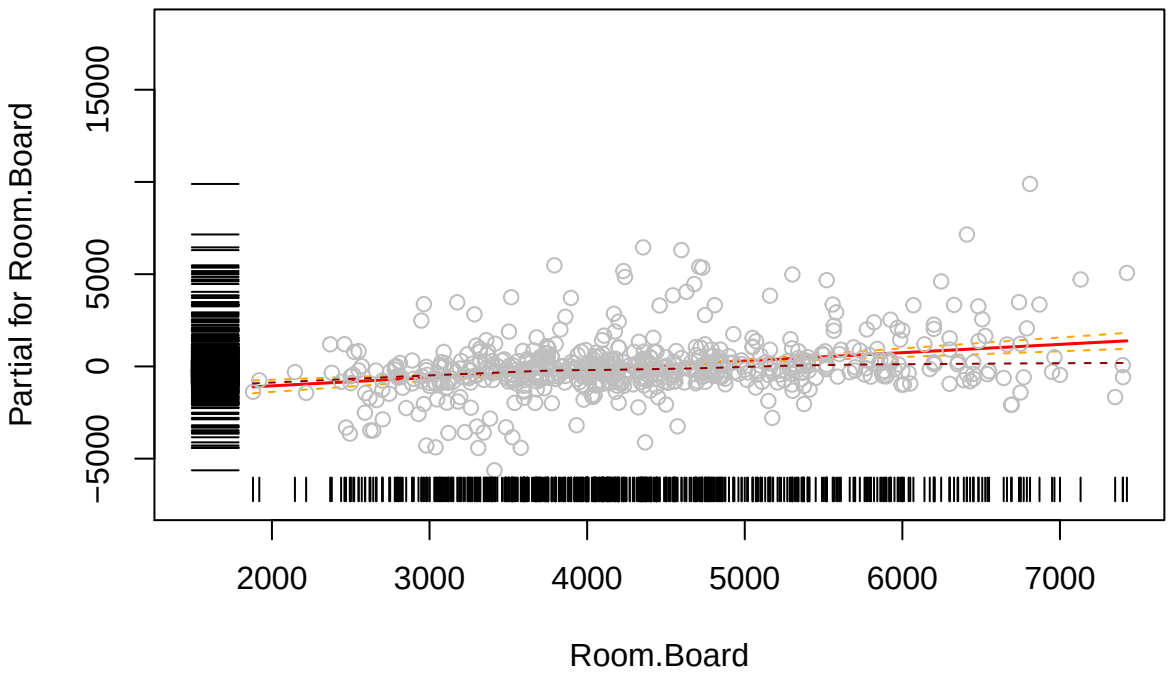
4. Build a second model, considering transformations of the response and predictors, possible interactions, etc with the goal of trying to achieve a model where assumptions for linear regression are satisfied, providing justification for your choices. Comment on how well the assumptions are met and and issues that diagnostic plots may reveal.

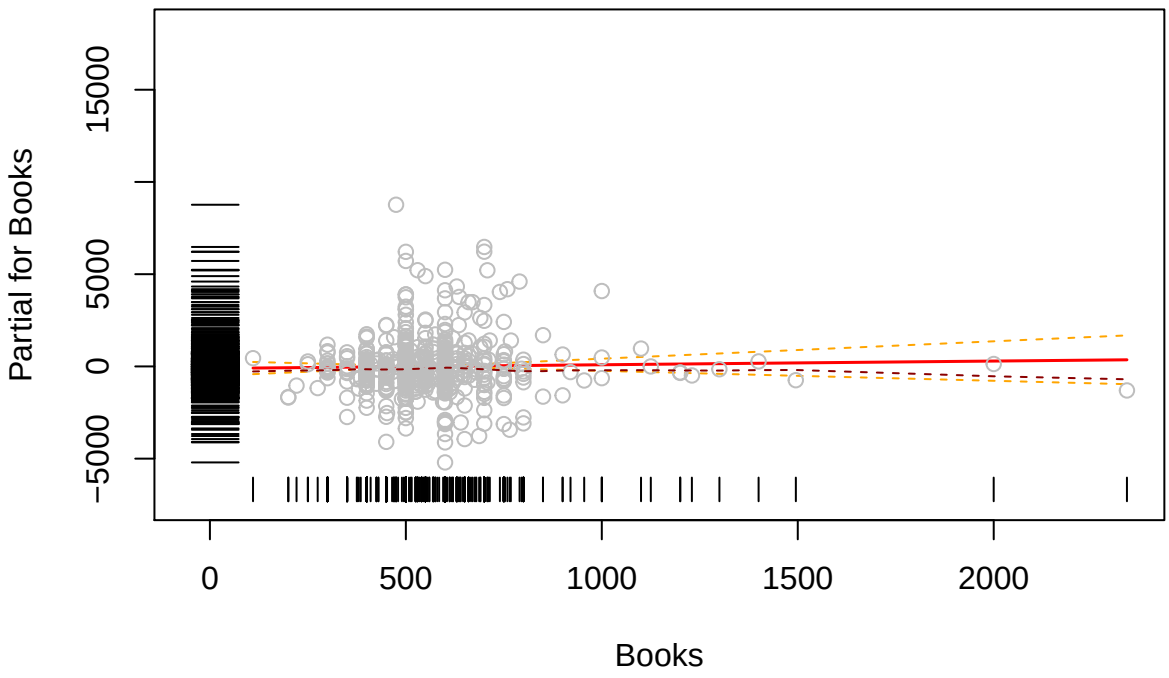
```
termplot(lm_fit1, partial.resid = TRUE, se=TRUE, smooth = "panel.smooth",rug = TRUE)
```

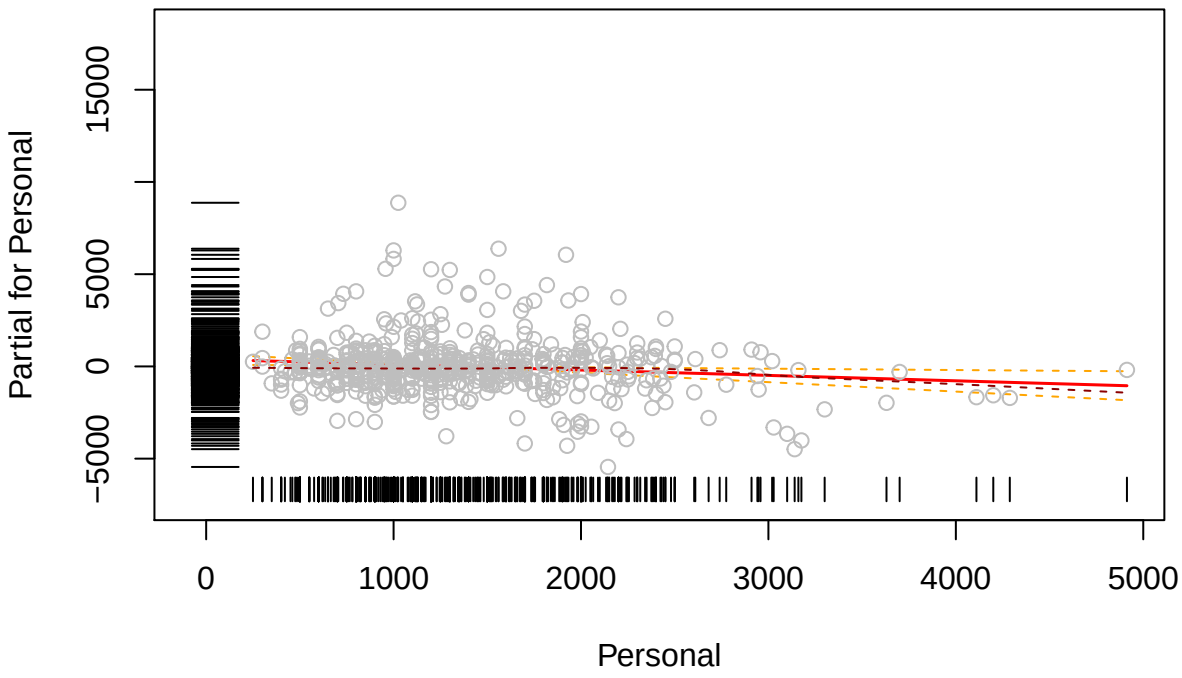


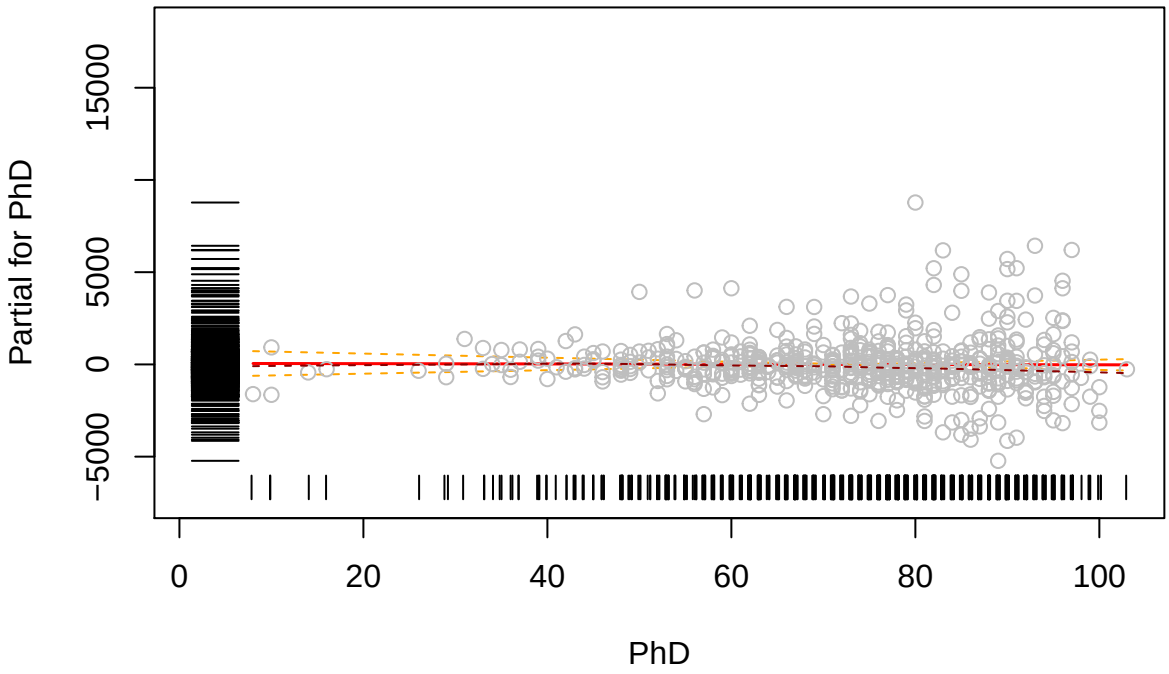

Partial for Top10perc
15000
5000
0
−5000
0
20
40
60
80
Top10perc

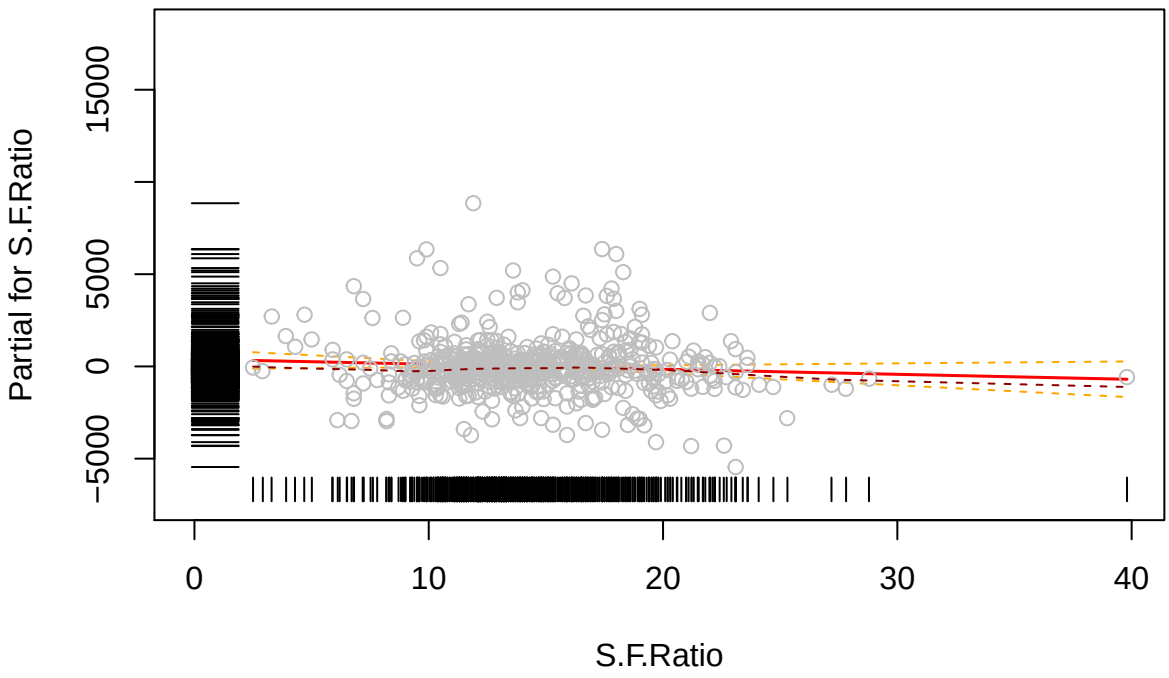

Partial for F.Undergrad
−5000
0
5000
15000
0
5000
10000
15000
20000
25000
30000
F.Undergrad

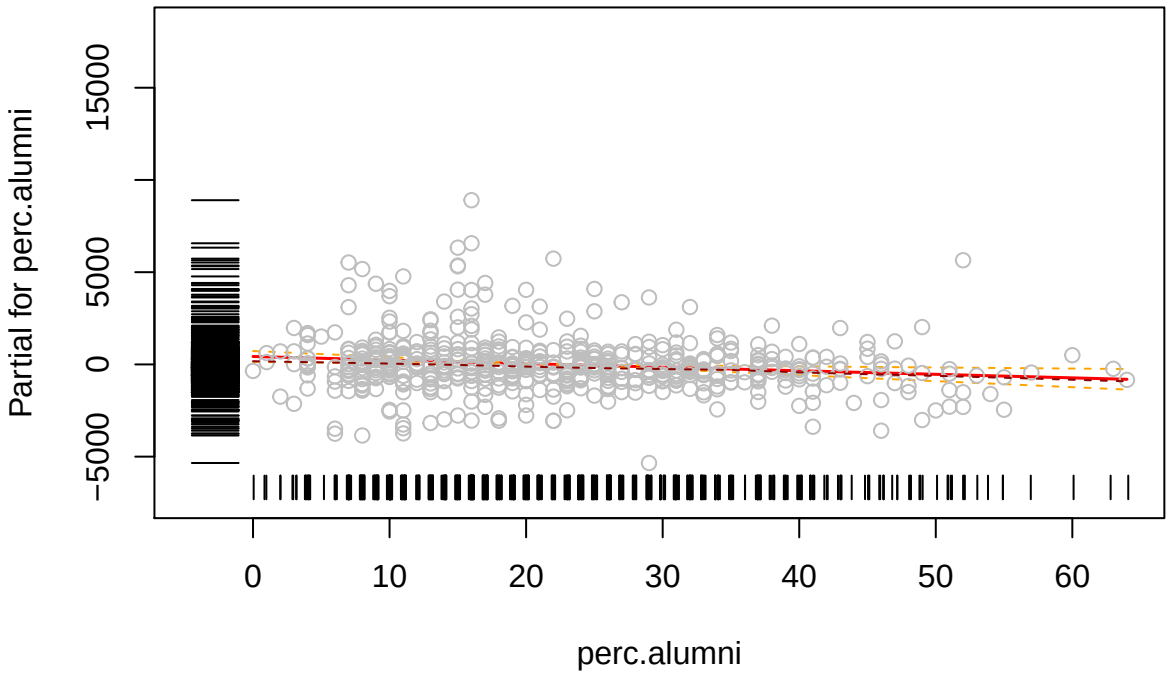

Partial for Room.Board
15000
5000
0
−5000
2000
3000
4000
5000
6000
7000
Room.Board

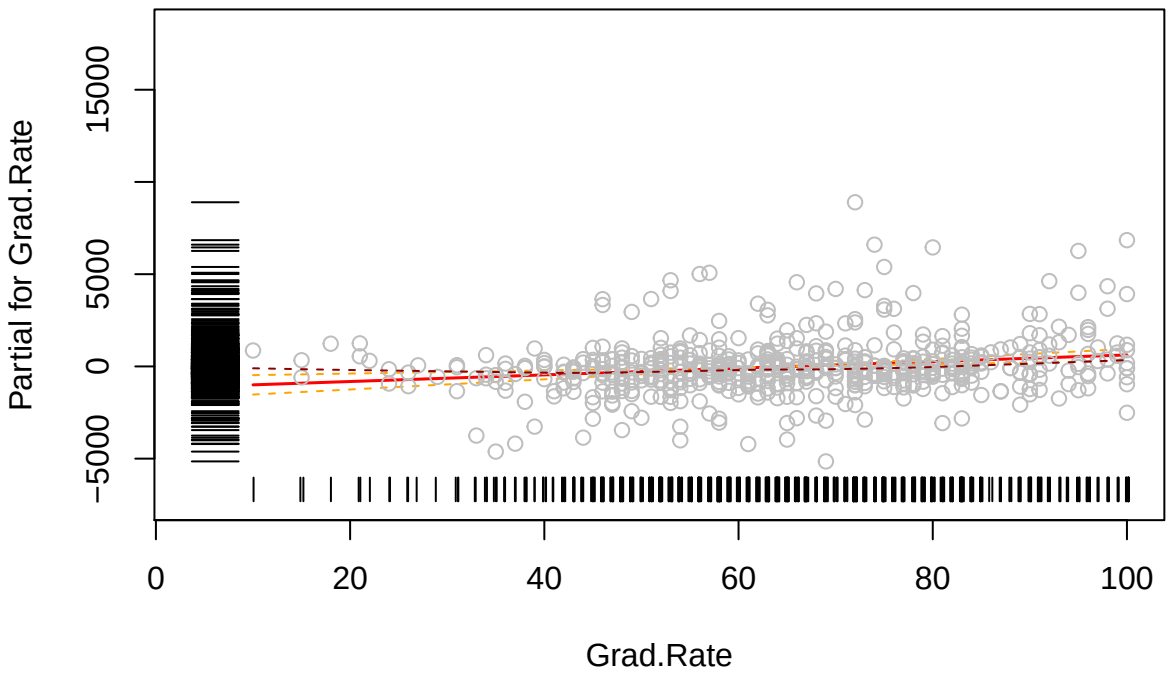

Partial for Books
15000
5000
0
−5000
0
500
1000
1500
2000
Books

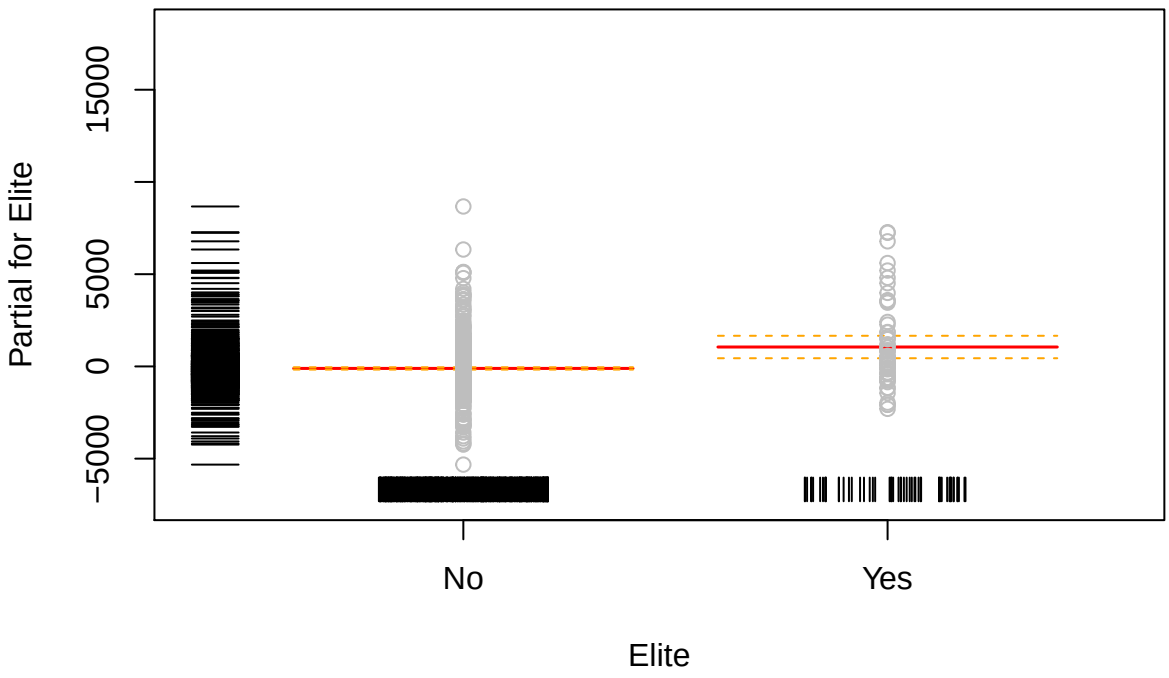

Partial for Personal
15000
5000
0
−5000
0
1000
2000
3000
4000
5000
Personal


Partial for PhD
15000
5000
0
−5000
0
20
40
60
80
100
PhD


Partial for S.F.Ratio
15000
5000
0
−5000
0
10
20
30
40
S.F.Ratio


Partial for perc.alumni
15000
5000
0
−5000
0
10
20
30
40
50
60
perc.alumni


Partial for Grad.Rate
15000
5000
0
−5000
0
20
40
60
80
100
Grad.Rate

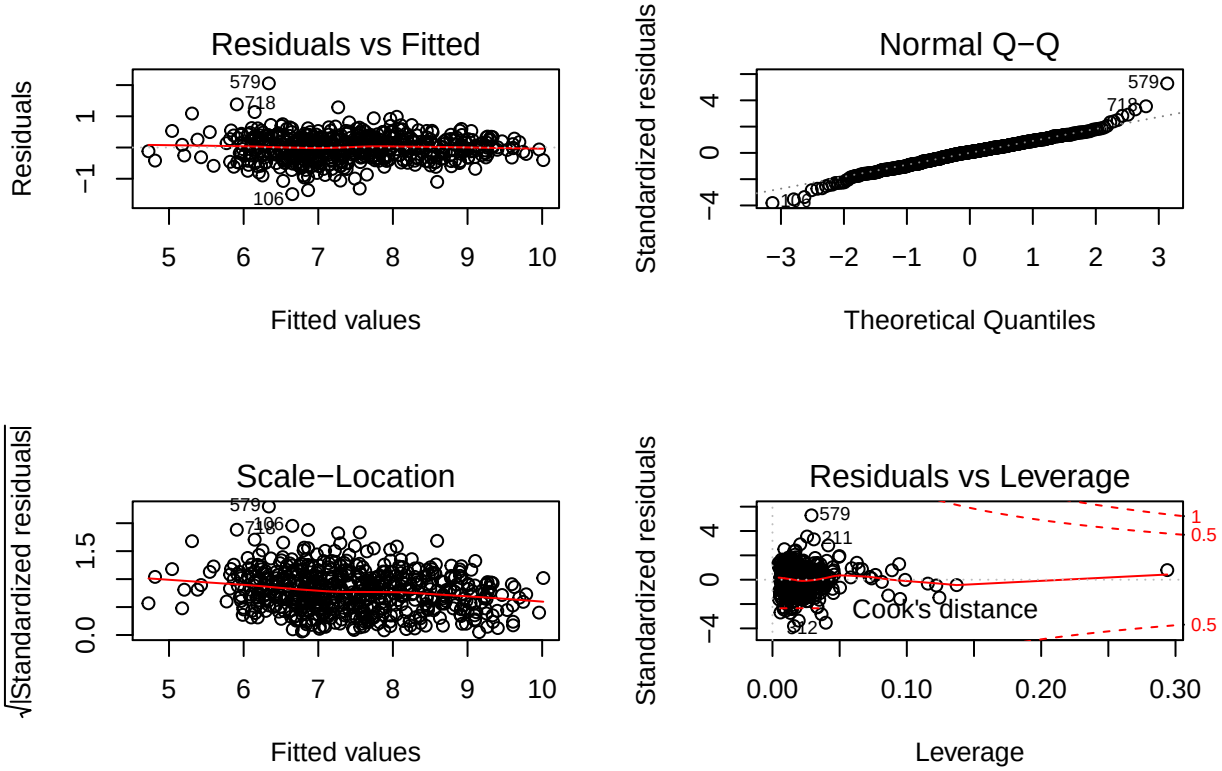



```
ct2 = College.train

ct2$F.Undergrad = log(ct2$F.Undergrad)
ct2$PhD = log(ct2$PhD)
ct2$Grad.Rate = log(ct2$Grad.Rate)
ct2$Top10perc = log(ct2$Top10perc)
ct2$Apps = log(ct2$Apps)

lm_fit2 = lm(Apps ~ ., data=ct2)
summary(lm_fit2)
```

```
##
## Call:
## lm(formula = Apps ~ ., data = ct2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.49309 -0.23898  0.02271  0.24262  2.05481
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.980e+00  3.715e-01  -5.328 1.43e-07 ***
## PrivateYes   9.304e-02  6.015e-02   1.547 0.122479
## Top10perc   -4.217e-02  3.548e-02  -1.189 0.235082
## F.Undergrad  9.657e-01  2.645e-02  36.515  < 2e-16 ***
## Room.Board   1.112e-04  1.849e-05   6.014 3.24e-09 ***
```

```
## Books         1.370e-04  9.965e-05   1.375 0.169611
## Personal     -7.738e-05  2.858e-05  -2.707 0.006983 **
## PhD           2.744e-01  7.292e-02   3.763 0.000185 ***
## S.F.Ratio    -8.697e-03  5.044e-03  -1.724 0.085211 .
## perc.alumni   4.028e-03  1.747e-03   2.306 0.021482 *
## Grad.Rate     1.134e-01  6.873e-02   1.650 0.099445 .
## EliteYes      3.175e-01  6.755e-02   4.701 3.25e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3946 on 570 degrees of freedom
## Multiple R-squared:  0.8643, Adjusted R-squared:  0.8617
## F-statistic: 330.1 on 11 and 570 DF,  p-value: < 2.2e-16
```

```
par(mfrow = c(2,2))
plot(lm_fit2)
```

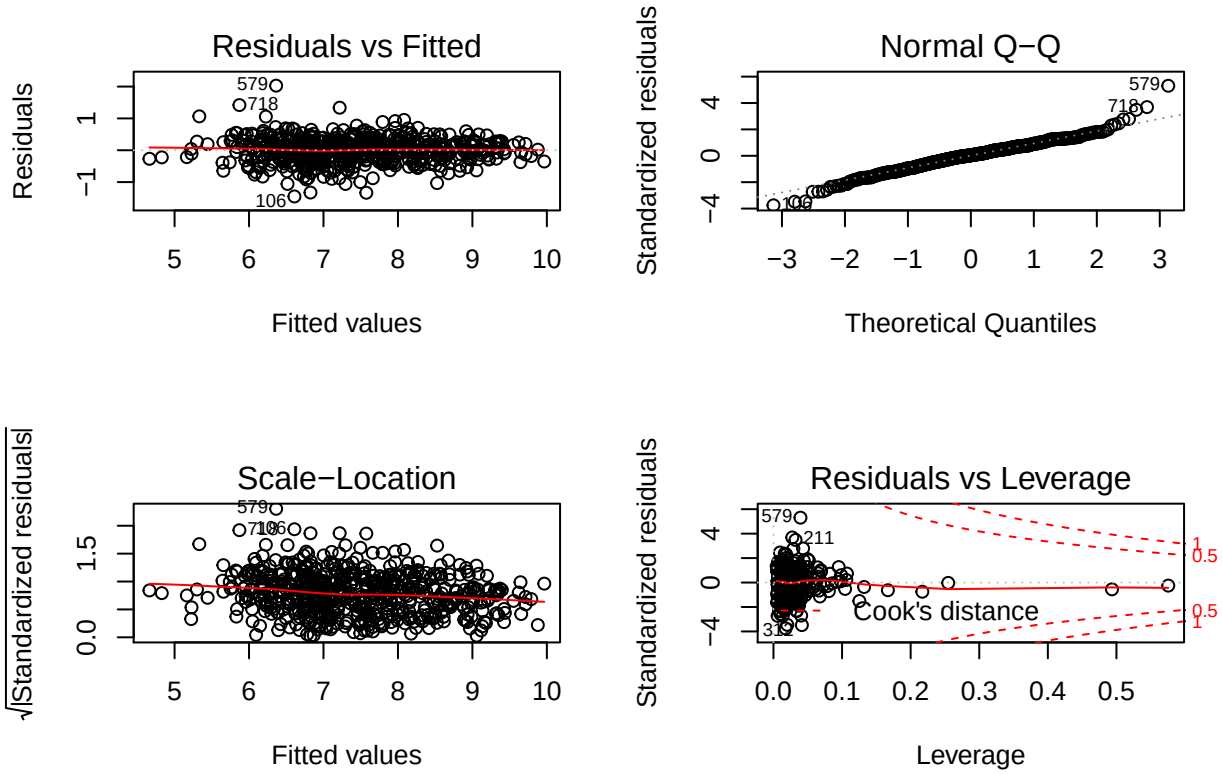



```
lm_fit3 = lm(Apps ~ . + (F.Undergrad + Personal + Books)^2, data = ct2)
summary(lm_fit3)
```

```
##
## Call:
## lm(formula = Apps ~ . + (F.Undergrad + Personal + Books)^2, data = ct2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4509 -0.2338  0.0168  0.2461  2.0284
```

```
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -3.438e+00  6.290e-01  -5.466 6.90e-08 ***
## PrivateYes            9.058e-02  5.992e-02   1.512  0.13118
## Top10perc            -2.806e-02  3.565e-02  -0.787  0.43160
## F.Undergrad           1.105e+00  7.093e-02  15.578  < 2e-16 ***
## Room.Board            1.076e-04  1.831e-05   5.874 7.26e-09 ***
## Books                 1.053e-03  7.878e-04   1.337  0.18178
## Personal              6.494e-04  2.118e-04   3.067  0.00227 **
## PhD                   3.078e-01  7.287e-02   4.224 2.79e-05 ***
## S.F.Ratio            -9.127e-03  5.022e-03  -1.818  0.06966 .
## perc.alumni           4.455e-03  1.731e-03   2.573  0.01032 *
## Grad.Rate             1.109e-01  6.819e-02   1.627  0.10427
## EliteYes              2.738e-01  6.787e-02   4.035 6.22e-05 ***
## F.Undergrad:Personal -6.872e-05  2.470e-05  -2.783  0.00557 **
## F.Undergrad:Books    -6.821e-05  1.096e-04  -0.622  0.53388
## Books:Personal       -3.098e-07  1.309e-07  -2.367  0.01825 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.39 on 567 degrees of freedom
## Multiple R-squared:  0.8682, Adjusted R-squared:  0.8649
## F-statistic: 266.7 on 14 and 567 DF,  p-value: < 2.2e-16
```

```r
par(mfrow = c(2,2))
plot(lm_fit3)
```




```
best_step = step(lm_fit3,direction = 'backward')
```

```
## Start:  AIC=-1081.19
## Apps ~ Private + Top10perc + F.Undergrad + Room.Board + Books +
##     Personal + PhD + S.F.Ratio + perc.alumni + Grad.Rate + Elite +
##     (F.Undergrad + Personal + Books)^2
##
##                         Df Sum of Sq    RSS     AIC
## - F.Undergrad:Books      1    0.0589 86.306 -1082.8
## - Top10perc              1    0.0942 86.341 -1082.5
## <none>                               86.247 -1081.2
## - Private                1    0.3476 86.594 -1080.8
## - Grad.Rate              1    0.4027 86.650 -1080.5
## - S.F.Ratio              1    0.5025 86.749 -1079.8
## - Books:Personal         1    0.8526 87.099 -1077.5
## - perc.alumni            1    1.0073 87.254 -1076.4
## - F.Undergrad:Personal   1    1.1779 87.425 -1075.3
## - Elite                  1    2.4762 88.723 -1066.7
## - PhD                    1    2.7144 88.961 -1065.2
## - Room.Board             1    5.2480 91.495 -1048.8
##
## Step:  AIC=-1082.79
## Apps ~ Private + Top10perc + F.Undergrad + Room.Board + Books +
##     Personal + PhD + S.F.Ratio + perc.alumni + Grad.Rate + Elite +
##     F.Undergrad:Personal + Books:Personal
##
```

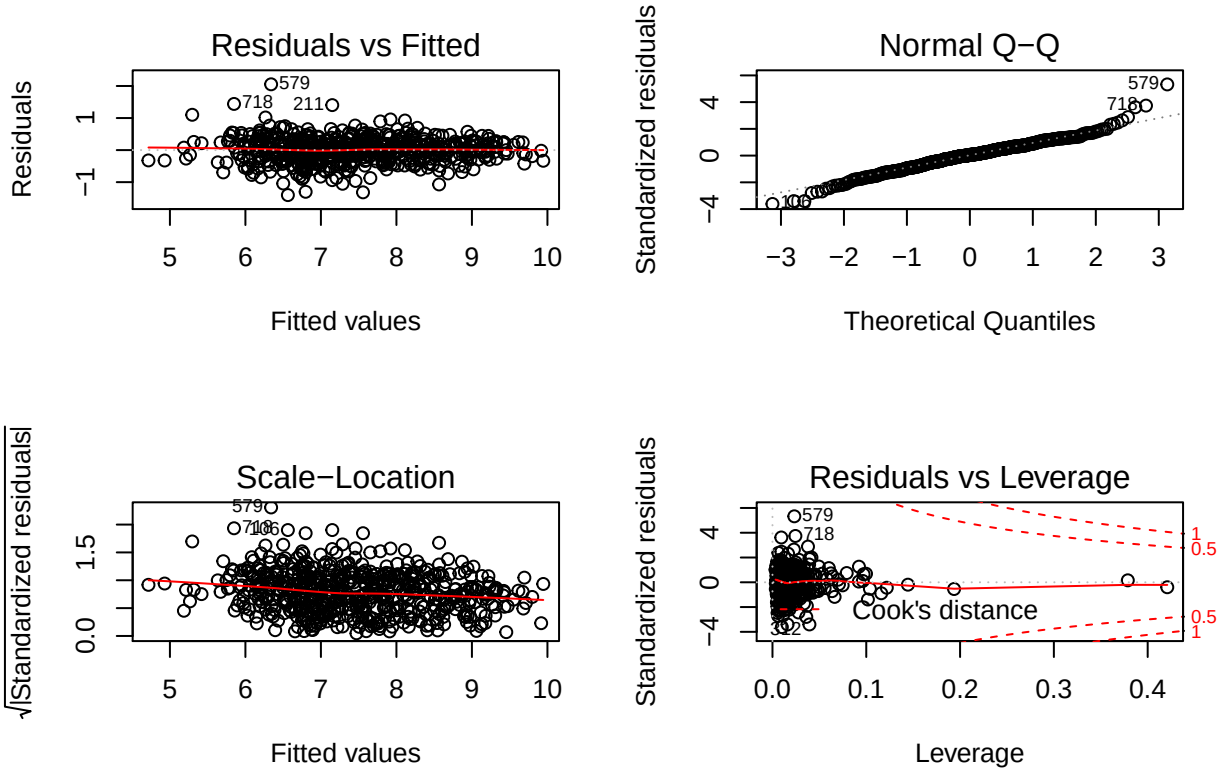
```
##                         Df Sum of Sq    RSS     AIC
## - Top10perc              1    0.1231 86.429 -1084.0
## <none>                               86.306 -1082.8
## - Private                1    0.3460 86.652 -1082.5
## - Grad.Rate              1    0.4213 86.727 -1082.0
## - S.F.Ratio              1    0.5103 86.816 -1081.4
## - perc.alumni            1    1.0082 87.314 -1078.0
## - Books:Personal         1    1.1376 87.443 -1077.2
## - F.Undergrad:Personal   1    1.2855 87.591 -1076.2
## - Elite                  1    2.5147 88.821 -1068.1
## - PhD                    1    2.6679 88.974 -1067.1
## - Room.Board             1    5.2360 91.542 -1050.5
##
## Step:  AIC=-1083.96
## Apps ~ Private + F.Undergrad + Room.Board + Books + Personal +
##     PhD + S.F.Ratio + perc.alumni + Grad.Rate + Elite + F.Undergrad:Personal +
##     Books:Personal
##
##                         Df Sum of Sq    RSS     AIC
## - Private                1    0.2638 86.693 -1084.2
## <none>                               86.429 -1084.0
## - Grad.Rate              1    0.3584 86.787 -1083.5
## - S.F.Ratio              1    0.4445 86.873 -1083.0
## - perc.alumni            1    0.9371 87.366 -1079.7
## - Books:Personal         1    1.1607 87.590 -1078.2
## - F.Undergrad:Personal   1    1.3524 87.781 -1076.9
## - Elite                  1    2.4739 88.903 -1069.5
## - PhD                    1    2.5990 89.028 -1068.7
## - Room.Board             1    5.5006 91.930 -1050.0
##
## Step:  AIC=-1084.19
## Apps ~ F.Undergrad + Room.Board + Books + Personal + PhD + S.F.Ratio +
##     perc.alumni + Grad.Rate + Elite + F.Undergrad:Personal +
##     Books:Personal
##
##                         Df Sum of Sq    RSS     AIC
## <none>                               86.693 -1084.2
## - Grad.Rate              1    0.5563 87.249 -1082.5
## - S.F.Ratio              1    0.5829 87.276 -1082.3
## - Books:Personal         1    1.0677 87.760 -1079.1
## - perc.alumni            1    1.0797 87.772 -1079.0
## - F.Undergrad:Personal   1    1.4550 88.148 -1076.5
## - Elite                  1    2.3811 89.074 -1070.4
## - PhD                    1    2.4540 89.147 -1069.9
## - Room.Board             1    6.6870 93.380 -1042.9
```

```
summary(best_step)
```

```
##
## Call:
## lm(formula = Apps ~ F.Undergrad + Room.Board + Books + Personal +
##     PhD + S.F.Ratio + perc.alumni + Grad.Rate + Elite + F.Undergrad:Personal +
##     Books:Personal, data = ct2)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -1.40359 -0.23030  0.02025  0.24909  2.05212
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -3.006e+00  4.677e-01  -6.427 2.76e-10 ***
## F.Undergrad            1.052e+00  4.363e-02  24.108  < 2e-16 ***
## Room.Board             1.159e-04  1.748e-05   6.631 7.76e-11 ***
## Books                  5.663e-04  1.866e-04   3.035 0.002513 **
## Personal               7.060e-04  2.044e-04   3.454 0.000592 ***
## PhD                    2.719e-01  6.769e-02   4.017 6.69e-05 ***
## S.F.Ratio             -9.563e-03  4.885e-03  -1.958 0.050753 .
## perc.alumni            4.542e-03  1.705e-03   2.664 0.007932 **
## Grad.Rate              1.245e-01  6.510e-02   1.912 0.056318 .
## EliteYes               2.467e-01  6.236e-02   3.957 8.56e-05 ***
## F.Undergrad:Personal -7.511e-05  2.428e-05  -3.093 0.002079 **
## Books:Personal       -3.250e-07  1.227e-07  -2.650 0.008283 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.39 on 570 degrees of freedom
## Multiple R-squared:  0.8675, Adjusted R-squared:  0.8649
## F-statistic: 339.2 on 11 and 570 DF,  p-value: < 2.2e-16
```

```r
par(mfrow = c(2,2))
plot(best_step)
```

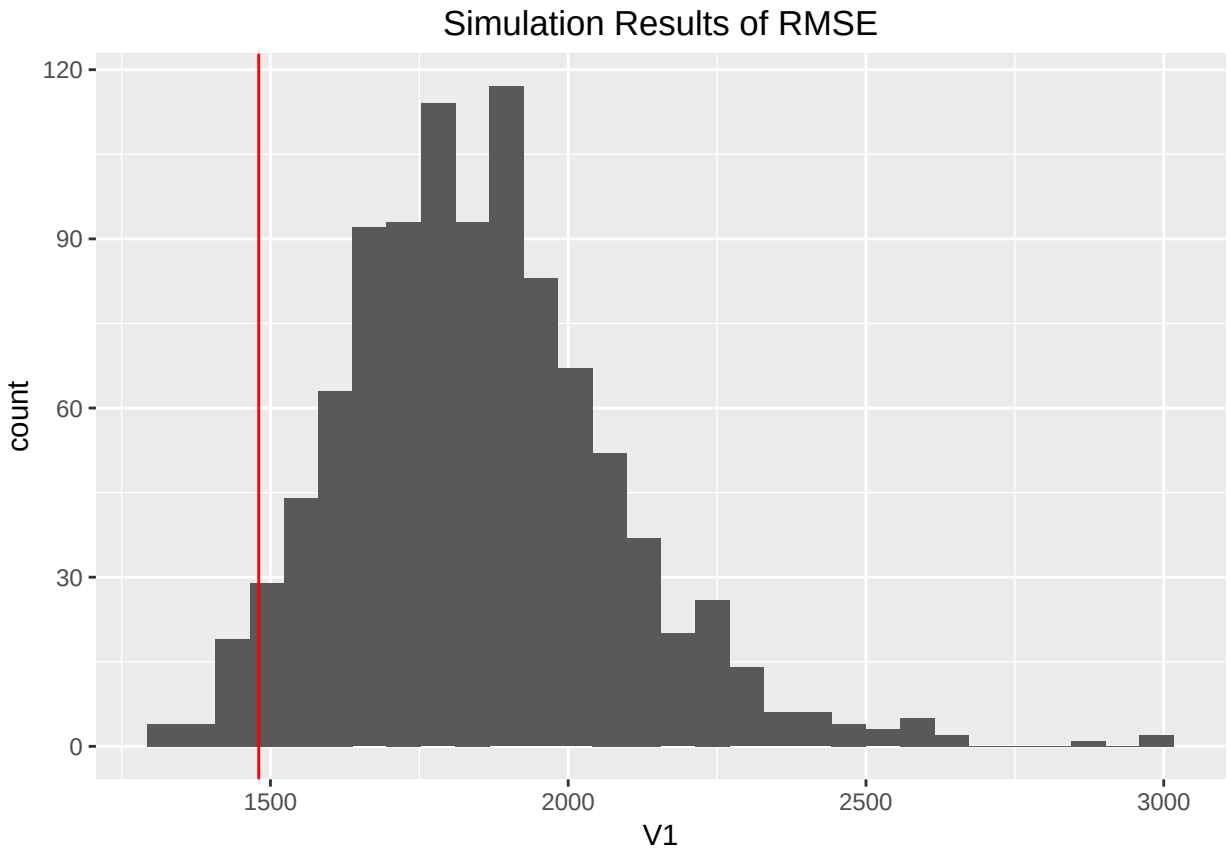


Comparing to the former model, after transforming some of the variables to log form and adding necessary interactions, the plots we get now are more reasonable. In the first graph, we can see that residuals spread equally around a horizontal line without distinct patterns, so we don't have non-linear relationships. For Normal Q-Q plots, residuals follow a straight line well, the residuls are normally distributed. In Scale-Location plot, residuals are spread equally along the range of predictors which is good. In the last graph, we can barely see the Cook's distance and all cases are inside of the Cook's distance lines, so there is no influtial case. From the summary, the interaction terms we chose are statistically significant.

5. Repeat the predictive checks described in problem 3, but using your model from problem 4. If you transform the response, you will need to back transform data to the original units in order to compute the RMSE in the original units. Does this suggest that the model is adequate? Do the two graphs provide information about which model is better?

```
ct = ct2
rmse = function(y, ypred){
  rmse = sqrt(mean((y-ypred)^2))
  return(rmse)
}

set.seed(1500)
X <- model.matrix(best_step)
n_sim <- 1000
sim_fit2 <- sim (best_step, n_sim)
beta <- sim_fit2@coef
n <- nrow(ct)
y.rep <- array(NA, c(n_sim, n))
rmse_rep_best = as.data.frame(matrix(NA, ncol =1, nrow = n_sim))
for (s in 1:n_sim){
  mu = (X %*% beta[s,])
  y.rep <- rnorm(n, mu , sim_fit2@sigma[s])
  ct$Apps = y.rep
  fit<- lm(Apps~ F.Undergrad + Room.Board + Books + Personal +
    PhD + S.F.Ratio + perc.alumni + Grad.Rate + Elite + F.Undergrad:Personal +
    Books:Personal, data = ct)
  rmse_rep_best[s,1] = rmse(exp(ct$Apps), exp(fitted(fit)))
}

rmse_fit_best = rmse(College.train$Apps, exp(fitted(best_step)))
ggplot(data = rmse_rep_best, aes(x = V1)) +
  geom_histogram() +
  labs(title = "Simulation Results of RMSE") +
  theme(plot.title = element_text(hjust = 0.5)) +
  geom_vline(xintercept =rmse_fit, col = "red")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

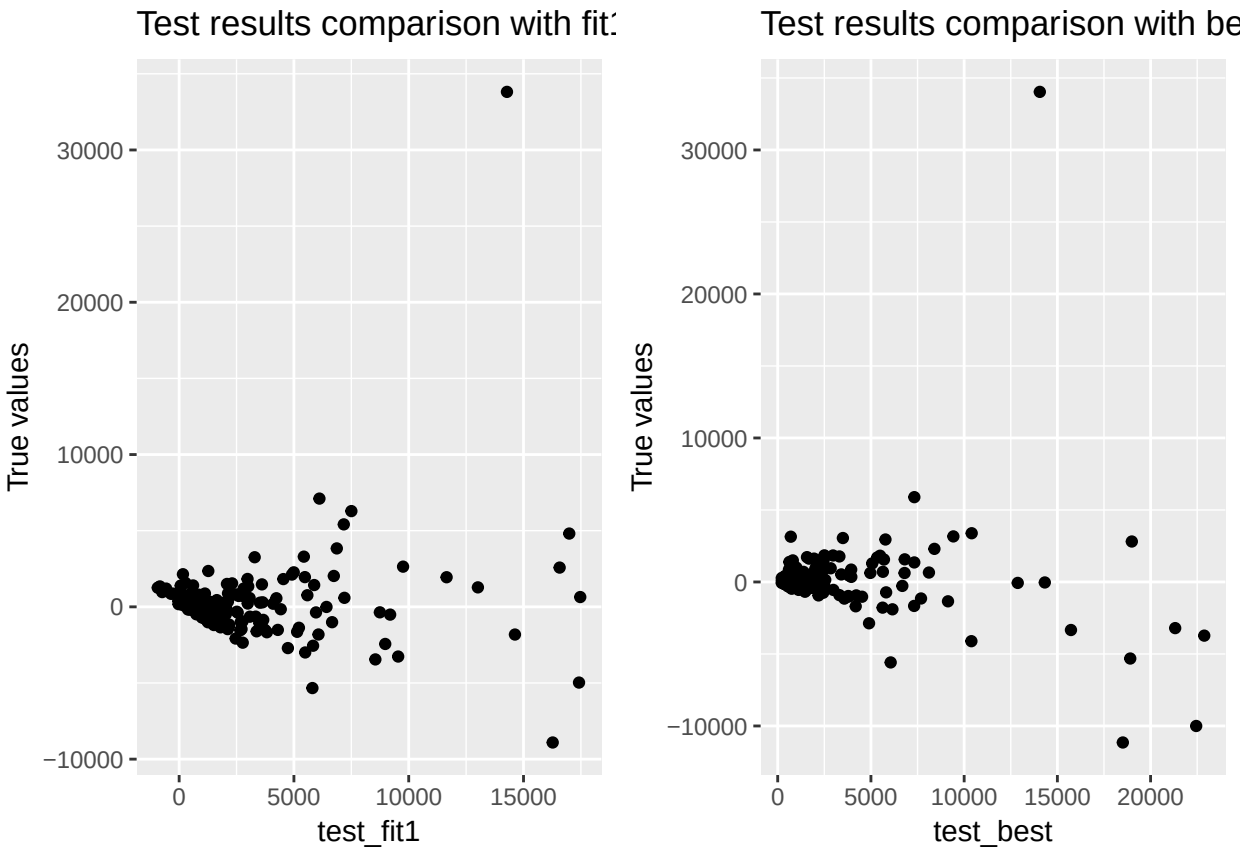


From the summary we can see that the mean of the replicated data sets is 1854 and RMSE of our model is 1223.868 which is much further away from the mean. It suggests that the model may be overfitting or under-dispersion.

6. Use your two fitted models to predict the number of applications for the testing data, `College.test`. Plot the predicted residuals $y_i - \hat{y}_i$ versus the predictions. Are there any cases where the model does a poor job of predicting? Compute the RMSE using the test data where now RMSE $= \sqrt{\sum_{i=1}^{n.test}(y_i - \hat{y}_i)^2/n.test}$ where the sum is over the test data. Which model is better for the out of sample prediction?

```
College.test = College.test %>%
  dplyr::select(-c(Top25perc,Outstate,Expend,P.Undergrad,Terminal, college))

cTest = College.test[-2]
cTest$F.Undergrad = log(cTest$F.Undergrad)
cTest$PhD = log(cTest$PhD)
cTest$Grad.Rate = log(cTest$Grad.Rate)
cTest$Top10perc = log(cTest$Top10perc)


test_fit1 = predict(lm_fit1, College.test[-2])
test_best = exp(predict(best_step, cTest))
df = data.frame(cbind(test_fit1, test_best, College.test$Apps)) %>%
  mutate(a=College.test$Apps-test_fit1) %>%
  mutate(b=College.test$Apps-test_best)

g1 = ggplot(df, aes(x = test_fit1, y = a)) +
  geom_point() +
```

```
  ylab("True values") +
  labs(title = "Test results comparison with fit1")
g2 = ggplot(df, aes(y = b, x = test_best)) +
  geom_point() +
  ylab("True values") +
  labs(title = "Test results comparison with best step")

grid.arrange(g1,g2,ncol = 2)
```




```
rmse_new = function(y, ypred){
  rmse = sqrt(sum((y-ypred)^2)/length(y))
  return(rmse)
}

rmse_fit_test = rmse_new(College.test$Apps, test_fit1)
rmse_best_test = rmse_new(College.test$Apps, test_best)
rmse_fit_test
```

```
## [1] 2929.351
```

```
rmse_best_test
```

```
## [1] 2944.965
```

Two models have similar RMSE, and model one is slightly smaller. The reason might be model two is overfitting. From the plots we can see that model1 is more scattered.

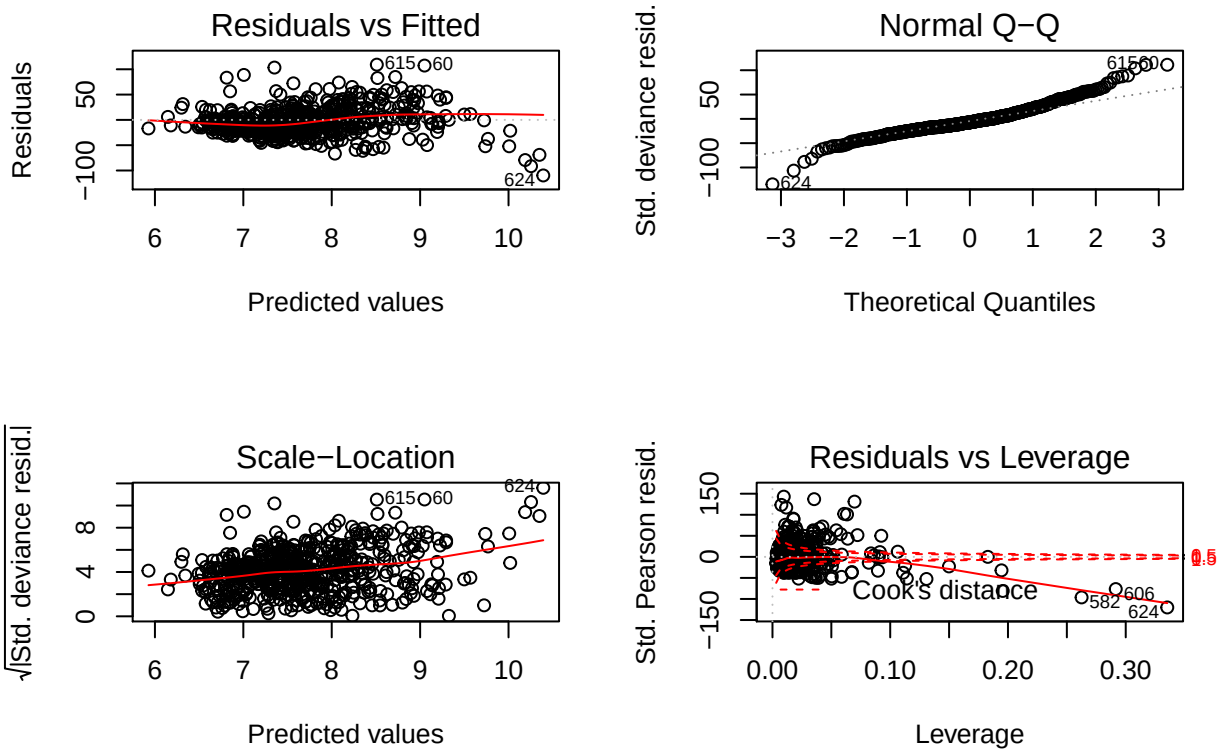
7. As the number of applications is a count variable, a Poisson regression model is a natural alternative for

modelling this data. Build a Poisson model using main effects and possible interactions/transformations. Comment on the model adequacy based on diagnostic plots and other summaries. Is there evidence that there is lack of fit?

```
fit_poisson = glm(Apps~., data = College.train, family = 'poisson' )
summary(fit_poisson)
```

```
##
## Call:
## glm(formula = Apps ~ ., family = "poisson", data = College.train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -109.741   -18.591    -7.021     9.424   109.071
##
## Coefficients:
##               Estimate Std. Error  z value Pr(>|z|)
## (Intercept)  5.541e+00  8.779e-03  631.198  < 2e-16 ***
## PrivateYes  -6.179e-01  2.656e-03 -232.605  < 2e-16 ***
## Top10perc   -2.090e-04  7.676e-05   -2.723  0.00647 **
## F.Undergrad  8.062e-05  1.554e-07  518.892  < 2e-16 ***
## Room.Board   1.983e-04  8.808e-07  225.174  < 2e-16 ***
## Books        3.171e-04  4.723e-06   67.141  < 2e-16 ***
## Personal    -4.805e-05  1.447e-06  -33.210  < 2e-16 ***
## PhD          1.119e-02  7.765e-05  144.168  < 2e-16 ***
## S.F.Ratio   -4.556e-03  2.235e-04  -20.382  < 2e-16 ***
## perc.alumni -6.682e-03  8.730e-05  -76.536  < 2e-16 ***
## Grad.Rate    1.031e-02  6.617e-05  155.797  < 2e-16 ***
## EliteYes     1.998e-01  3.660e-03   54.587  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1753917  on 581  degrees of freedom
## Residual deviance:  431850  on 570  degrees of freedom
## AIC: 437277
##
## Number of Fisher Scoring iterations: 5
```

```
par(mfrow = c(2,2))
plot(fit_poisson)
```




```
pchisq(summary(fit_poisson)$deviance, summary(fit_poisson)$df.residual, lower.tail = F)
```

```
## [1] 0
```

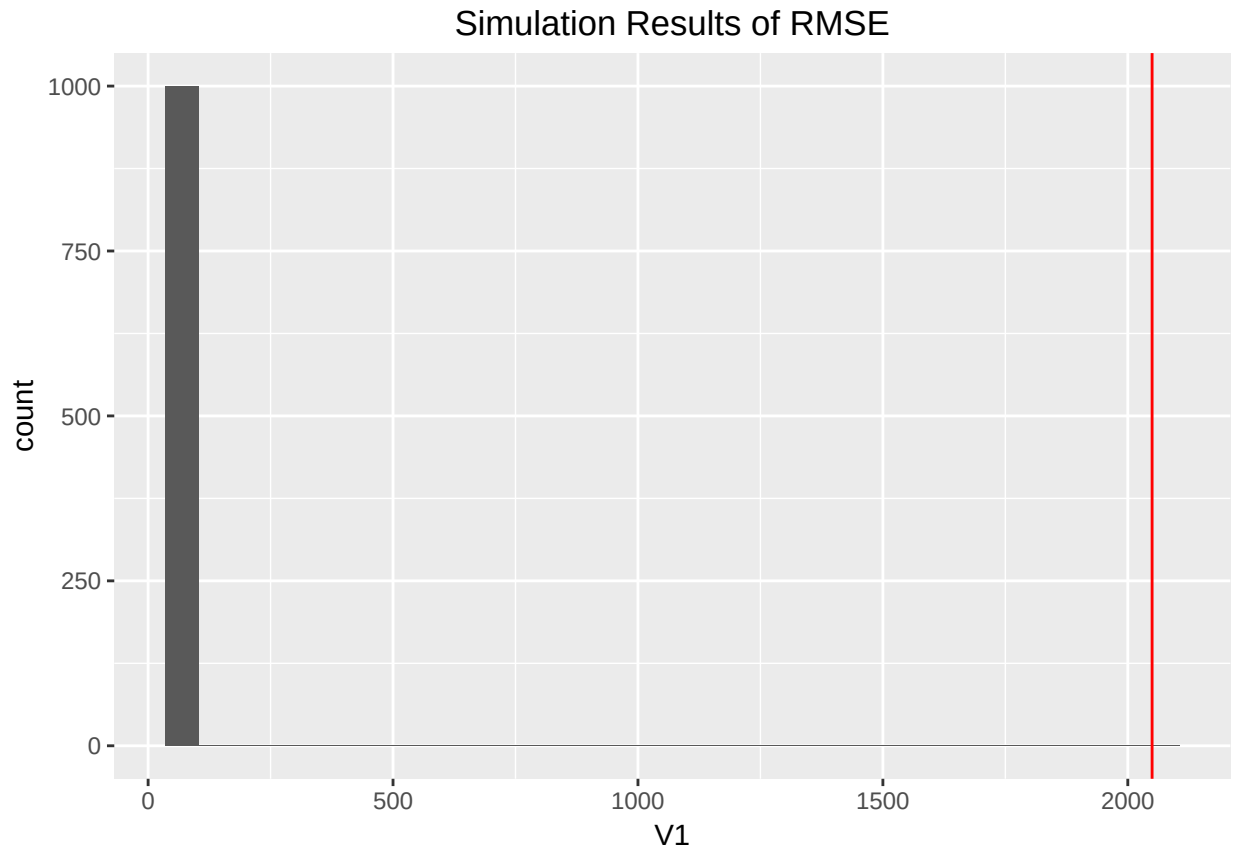
From the plot4, we find lots of points have Cook distance which is larger than 1, so they all have high leverages. And other plots also show the evidence of existence of highly influential points. So there is a lack of fit for this Poisson regression model. The p-value from the chi-square test suggests that a residual deviance as large or larger than what we observed under the model is highly unlikely. Hence the model is not adequate or lack of fit (overdispersion). All varables are statistically significant.

8. Generate 1000 replicate data sets using the coefficients from the Poisson model you fit above. Using RMSE as a statistic, $\sqrt{\sum_i (y^{\text{rep}} - \hat{y}_i^{\text{rep}})^2/n}$, how does the RMSE from the model based on the training data compare to RMSE's based on the replicated data. What does this suggest about model adequacy? Provide a histogram of the RMSE's with a line showing the location of the observed RMSE and compute a p-value.

```
ct = College.train
rmse = function(y, ypred){
  rmse = sqrt(mean((y-ypred)^2))
  return(rmse)
}

X <- model.matrix(fit_poisson)
n_sim <- 1000
sim_fit1 <- sim(fit_poisson, n_sim)
beta <- sim_fit1@coef

n <- nrow(College.train)
```

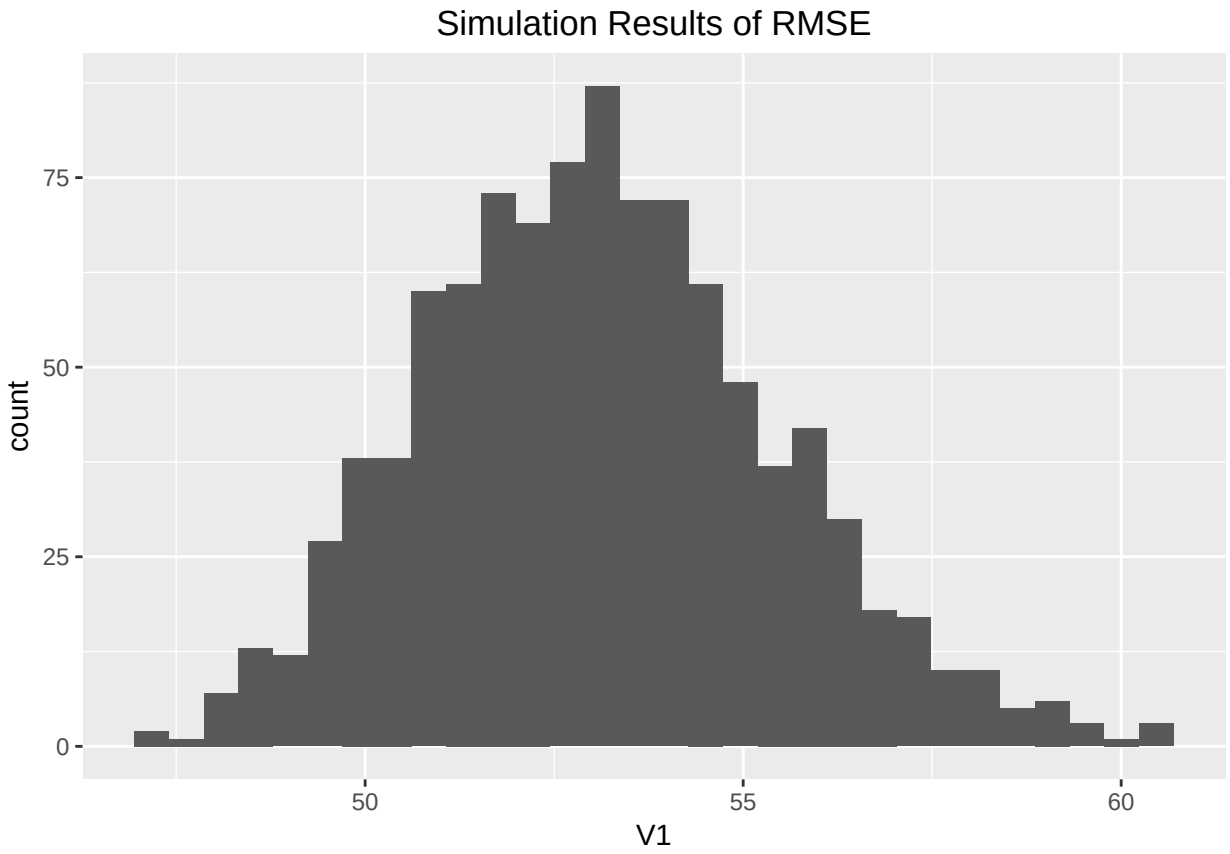
```
y.rep <- array(NA, c(n_sim, n))
rmse_rep = as.data.frame(matrix(NA, ncol =1, nrow = n_sim))
for (s in 1:n_sim){
  mu = exp(X %*% beta[s,])
  y.rep <- rpois(n, mu)
  ct$Apps = y.rep
  fit<- glm(Apps~., data = ct, family = 'poisson')
  rmse_rep[s,1] = rmse(ct$Apps, fitted(fit))
}

rmse_fit = rmse(College.train$Apps, fitted(fit_poisson))
ggplot(data = rmse_rep, aes(x = V1)) +
  geom_histogram() +
  labs(title = "Simulation Results of RMSE") +
  theme(plot.title = element_text(hjust = 0.5)) +
  geom_vline(xintercept =rmse_fit, col = "red")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
rmse_fit = rmse(College.train$Apps, fitted(fit_poisson))
ggplot(data = rmse_rep, aes(x = V1)) +
  geom_histogram() +
  labs(title = "Simulation Results of RMSE") +
  theme(plot.title = element_text(hjust = 0.5))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
PV = ifelse(mean(rmse_rep > rmse_fit) < 0.5,
2 * mean(rmse_rep > rmse_fit),
2 * (1 - mean(rmse_rep > rmse_fit)))
PV
```

```
## [1] 0
```

From the histogram we see the distribution of RMSE based on the replicated data is not centralized, and the observed RMSE is located far away from replicated data RMSE distribution, which indicates our regression model based on training data is fitting badly. In addition, the RMSE for the model based on the training data is on the right, which suggests that the model is not explaining enough of the variation of the data, hence overdispersion. Also, In this case, p-value is 0.

9. Using the test data set, calculate the RMSE for the test data using the predictions from the Poisson model. How does this compare to the RMSE based on the observed data? Is this model better than the linear regression models in terms of out of sample prediction?

```
testData = College.test[,-2]
ypred = exp(predict(fit_poisson,newdata = testData))
rmse(y = College.test[,2] , ypred )
```

```
## [1] 2875.51
```

The RMSE for the test data is much larger that the RMSE based on the observed data.

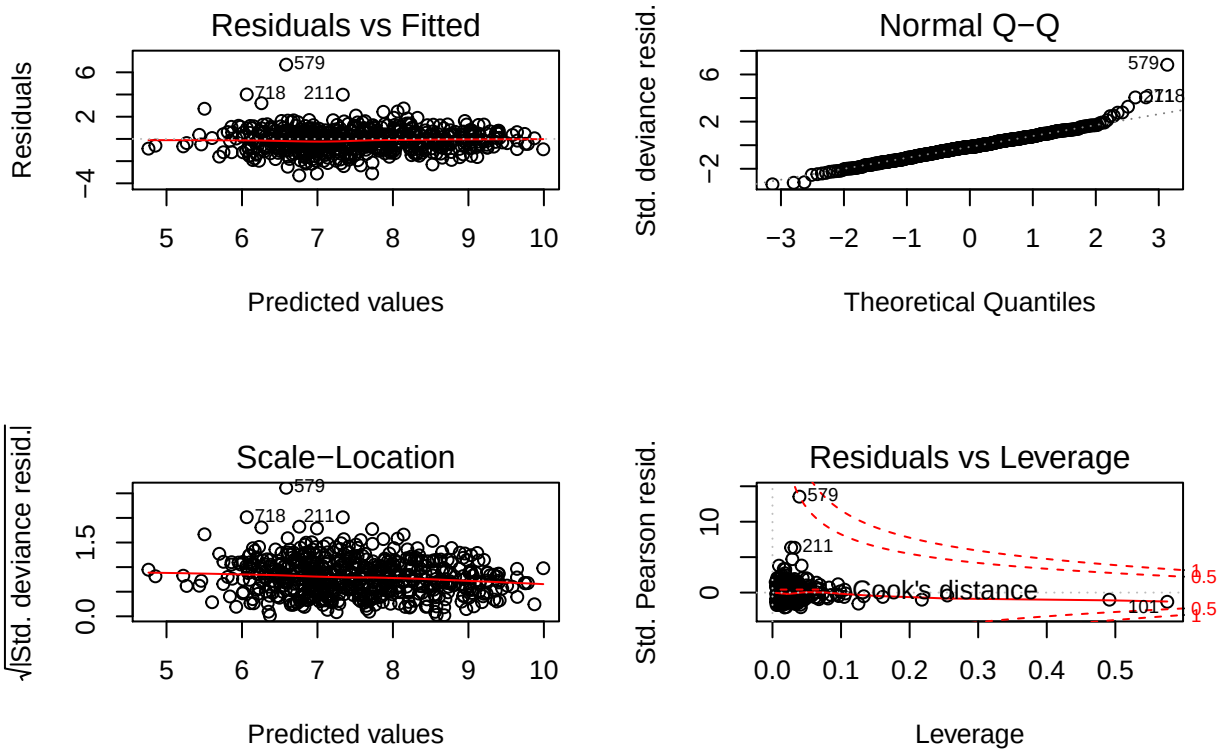
10. Build a model using the negative binomial model (consider transformations and interactions if needed) and examine diagnostic plots. Are there any suggestions of problems with this model?

```
ctnb =  College.train
ctnb$F.Undergrad = log(ctnb$F.Undergrad)
```

```r
ctnb$PhD = log(ctnb$PhD)
ctnb$Grad.Rate = log(ctnb$Grad.Rate)
ctnb$Top10perc = log(ctnb$Top10perc)

fit_nb = glm.nb(Apps ~ . + (Personal + F.Undergrad + Books)^2 , data=ctnb )

summary(fit_nb)
```

```
##
## Call:
## glm.nb(formula = Apps ~ . + (Personal + F.Undergrad + Books)^2,
##     data = ctnb, init.theta = 6.850165023, link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2893  -0.7614  -0.1433   0.4730   6.6878
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -3.494e+00  6.195e-01  -5.639 1.71e-08 ***
## PrivateYes            1.459e-01  5.887e-02   2.479  0.01318 *
## Top10perc            -2.740e-02  3.503e-02  -0.782  0.43408
## F.Undergrad           1.127e+00  6.980e-02  16.151  < 2e-16 ***
## Room.Board            9.128e-05  1.799e-05   5.075 3.88e-07 ***
## Books                 1.596e-03  7.752e-04   2.059  0.03946 *
## Personal              6.139e-04  2.085e-04   2.945  0.00323 **
## PhD                   3.249e-01  7.181e-02   4.525 6.04e-06 ***
## S.F.Ratio            -4.176e-03  4.932e-03  -0.847  0.39713
## perc.alumni           3.042e-03  1.702e-03   1.788  0.07380 .
## Grad.Rate             7.905e-02  6.709e-02   1.178  0.23870
## EliteYes              2.797e-01  6.661e-02   4.200 2.67e-05 ***
## F.Undergrad:Personal -6.806e-05  2.429e-05  -2.802  0.00508 **
## Books:Personal       -2.858e-07  1.288e-07  -2.219  0.02651 *
## F.Undergrad:Books    -1.317e-04  1.078e-04  -1.221  0.22204
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(6.8502) family taken to be 1)
##
##     Null deviance: 4379.24  on 581  degrees of freedom
## Residual deviance:  596.47  on 567  degrees of freedom
## AIC: 9261.9
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  6.850
##           Std. Err.:  0.395
##
##  2 x log-likelihood:  -9229.942
```

```r
par(mfrow = c(2,2))
plot(fit_nb)
```



We take the same transformation as the poisson model. In general, this model satisfies linearity, but there are some points spead away from the straight line, and these points are potentially influential for our model; the second and third plots also suggest the evidence of possibility of influential points; the last plot proves this guess, since several points are in the outside of Cook's distance, suggesting they are points with high leverage. And there may be problem of overdispersion.

11. Carry out the predictive checks for the negative model model using simulated replicates with RMSE and add RMSE from the test data and observed data to your plot. What do these suggest about 1) model adequacy and 2) model comparison? Which model out of all that you have fit do you recommend?

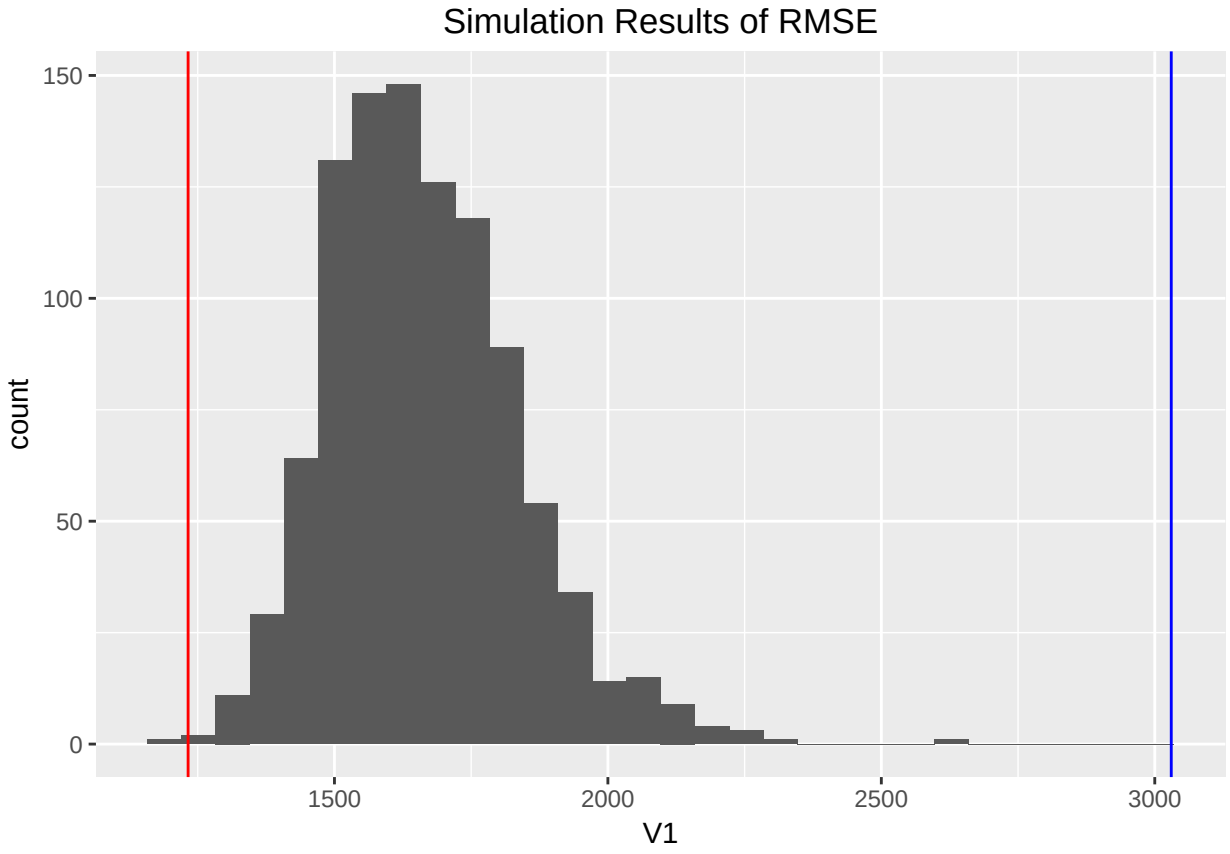
```
set.seed(100)
ct = ctnb
n_sim = 1000
n = nrow(ct)
X = model.matrix(fit_nb)
class(fit_nb) <- "glm" # over-ride class of "glm.nb"
sim.hiv.nb = sim(fit_nb, n_sim) # use GLM to generate beta's
sim.hiv.nb@sigma = rnorm(n_sim, fit_nb$theta, fit_nb$SE.theta) # add slot for theta overide sigma

y.rep = array(NA, c(n_sim, nrow(ct)))

rmse_rep = as.data.frame(matrix(NA, ncol =1, nrow = n_sim))


for (i in 1:n_sim) {
mu = exp(X %*% sim.hiv.nb@coef[i,])
y.rep = rnegbin(n, mu=mu, theta=sim.hiv.nb@sigma[i])
```

```
ct$Apps = y.rep
fit = glm.nb(Apps ~ . + (Personal + F.Undergrad + Books)^2 , data = ct)
rmse_rep[i,1] = rmse(ct$Apps, round(fitted(fit)))
}

rmse_fit = rmse(College.train$Apps, fitted(fit_nb))

rmse_test_nb = rmse(College.test$Apps,predict(fit_nb, cTest, type = 'response') )
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : calling predict.lm(<fake-lm-object>) ...
```

```
ggplot(data = rmse_rep, aes(x = V1)) +
  geom_histogram() +
  labs(title = "Simulation Results of RMSE") +
  theme(plot.title = element_text(hjust = 0.5))+
  geom_vline(xintercept =rmse_fit, col = "red") +
  geom_vline(xintercept =rmse_test_nb, col = "blue")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



It is usually the case that the test data RMSE is larger than the train data RMSE. Our plot shows that RMSE for the observed data falls in the left, on the other hand, the RMSE for the test data falls in the right. This suggests that the model may be overfitting, and it performs well on the training data but not as well for the test data.

According to the diganostic plot, comparing with the former models, we considered that the negative bionomial

model is the best. Based on all the models we have fitted, we conclude that two Poisson models and the linear models are not the best choice in this case, however, the negative binomial model(fit_nb) is relatively reasonable based on the RMSE plots.

12. While RMSE is a popular summary for model goodness of fit, coverage of confidence intervals is an alternative. For each case in the test set, find a 95% prediction interval. Now evaluate if the response is in the test data are inside or outside of the intervals. If we have the correct coverage, we would expect that at least 95% of the intervals would contain the test cases. Write a function to calculate coverage (the input should be the fitted model object and the test data-frame) and then evaluate coverage for each of the models that you fit (the two normal, the Poisson and the negative binomial). Include plots of the confidence intervals versus case number ordered by the prediction, with the left out data added as points. Comment on the plots, highlighting any unusual colleges where the model predicts poorly.

```
library(mvtnorm)
# test data
ctest1 = College.test
ctest1$F.Undergrad = log(ctest1$F.Undergrad)
ctest1$PhD = log(ctest1$PhD)
ctest1$Grad.Rate = log(ctest1$Grad.Rate)
ctest1$Top10perc = log(ctest1$Top10perc)
ctest1$Apps = log(ctest1$Apps)

ctest2 = College.test

ctpoisson = College.train
ctpoisson$F.Undergrad = log(ctpoisson$F.Undergrad)
ctpoisson$PhD = log(ctpoisson$PhD)
ctpoisson$Grad.Rate = log(ctpoisson$Grad.Rate)
ctpoisson$Top10perc = log(ctpoisson$Top10perc)
fit_poisson2 = glm(Apps ~ . + (Personal + Books + F.Undergrad)^2 , data=ctpoisson, family = 'poisson')
summary(fit_poisson2)
```

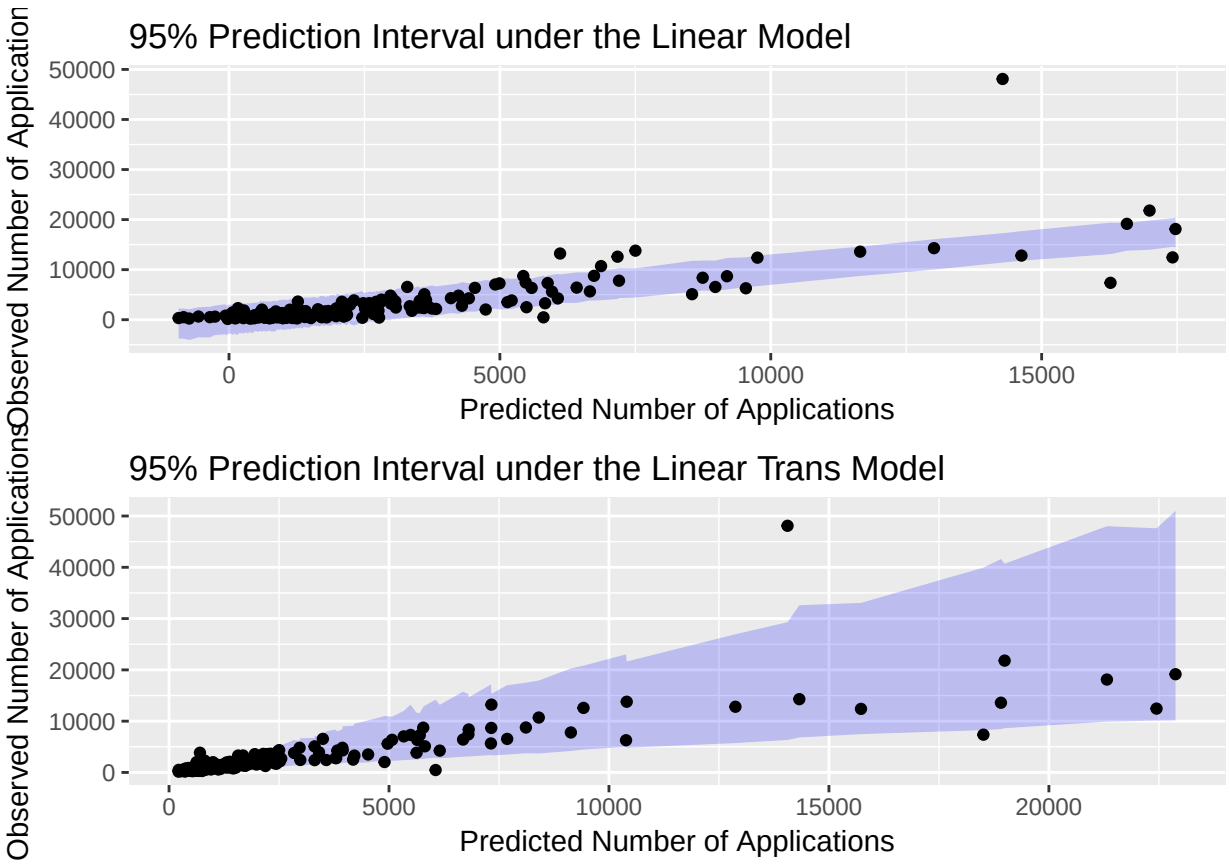
```
##
## Call:
## glm(formula = Apps ~ . + (Personal + Books + F.Undergrad)^2,
##     family = "poisson", data = ctpoisson)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -56.185  -11.486   -2.340    7.593  102.857
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -2.788e+00  3.973e-02 -70.172  < 2e-16 ***
## PrivateYes           -8.061e-03  2.898e-03  -2.782  0.00541 **
## Top10perc            -5.253e-02  1.865e-03 -28.161  < 2e-16 ***
## F.Undergrad           9.730e-01  3.998e-03 243.397  < 2e-16 ***
## Room.Board            1.261e-04  8.789e-07 143.471  < 2e-16 ***
## Books                -5.361e-05  5.029e-05  -1.066  0.28650
## Personal              5.605e-04  1.239e-05  45.224  < 2e-16 ***
## PhD                   2.338e-01  4.952e-03  47.213  < 2e-16 ***
## S.F.Ratio            -9.528e-03  2.196e-04 -43.381  < 2e-16 ***
## perc.alumni           3.801e-03  9.012e-05  42.182  < 2e-16 ***
## Grad.Rate             3.213e-01  3.978e-03  80.761  < 2e-16 ***
## EliteYes              2.505e-01  2.815e-03  89.009  < 2e-16 ***
```

```
## Books:Personal       -2.539e-07  8.694e-09 -29.202  < 2e-16 ***
## F.Undergrad:Personal -6.031e-05  1.289e-06 -46.778  < 2e-16 ***
## F.Undergrad:Books     6.502e-05  6.605e-06   9.843  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1753917  on 581  degrees of freedom
## Residual deviance:  176721  on 567  degrees of freedom
## AIC: 182154
##
## Number of Fisher Scoring iterations: 4
```

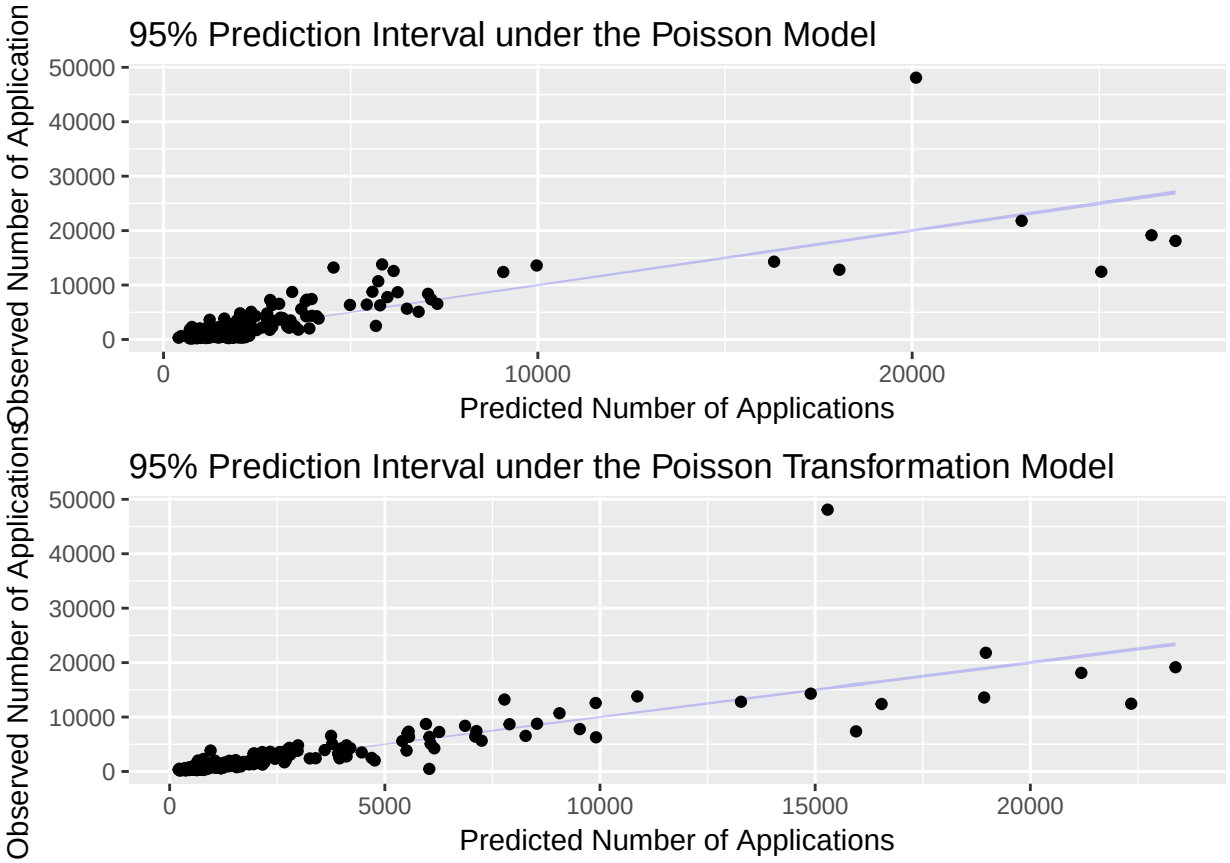
```
## refit negative - binomial
fit_nb = glm.nb(Apps ~ . + (Personal + F.Undergrad + Books)^2 , data=ctnb )
crage = function(y, pi){
  mean(y >= pi[,1] & y <= pi[,2])
}

###### interval function
pi = function(object, newdata, level = 0.95, nsim =1000){
  n = nrow(newdata)
  X = model.matrix(object, data = newdata)
  y.rep = matrix(NA, nsim, n)

  if (class(object)[1] == "negbin"){
    beta = rmvnorm(nsim, coef(object), vcov(object))
    theta = rnorm(nsim, object$theta, object$SE.theta)
    for (i in 1:nsim){
      mu = exp(X %*% beta[i,])
      y.rep[i,] = rnegbin(n, mu=mu, theta=theta[i])
    }
  }

  if(class(object)[1] == "glm"){
    sim_object = sim(object, nsim)
    beta = sim_object@coef
    for (i in 1:nsim){
      mu = exp(X %*% beta[i,])
      y.rep[i,] <- rpois(n, mu)
    }
  }

  if(class(object)[1] == "lm"){
    sim_object = sim(object, nsim)
    beta = sim_object@coef
    for(i in 1:nsim){
      mu = (X %*% beta[i,])
      y.rep[i, ] <- rnorm(n, mu, sim_object@sigma[i])
    }
  }

  pi = t(apply(y.rep, 2, function(x) {
```

```
    quantile(x, c((1-level)/2, .5+ level/2))}))
  return(pi)
}

## CI dataframe
cTest_poi = cbind(cTest, Apps = College.test$Apps)
lm_CI = pi(lm_fit1,College.test)
linear_trans_CI = pi(best_step,ctest1)
possion_CI = pi(fit_poisson, College.test)
possion_trans_CI = pi(fit_poisson2, cTest_poi)
ng_trans_CI = pi(fit_nb, cTest_poi)

## nb CI plot
CIdf_nb = data.frame(Apps = College.test$Apps,pred = predict(fit_nb, cTest, type = 'response'), upr = ng

    PP5 = ggplot(CIdf_nb, aes(x=pred, y=Apps)) +
    geom_ribbon(aes(ymin = lwr, ymax = upr), fill = "blue", alpha = 0.2) +
    geom_point(aes(y=Apps)) + xlab("Predicted Number of Applications") +
    ylab("Observed Number of Applications") +
    ggtitle("95% Prediction Interval under the Negative Binomial")

##lm CI plot
CIdf_lm = data.frame(Apps = College.test$Apps,pred = predict(lm_fit1, College.test[-2]), upr = lm_CI[,1

    PP1 = ggplot(CIdf_lm, aes(x=pred, y=Apps)) +
    geom_ribbon(aes(ymin = lwr, ymax = upr), fill = "blue", alpha = 0.2) +
    geom_point(aes(y=Apps)) + xlab("Predicted Number of Applications") +
    ylab("Observed Number of Applications") +
    ggtitle("95% Prediction Interval under the Linear Model")



##lm_transCI plot
CIdf_lm_trans = data.frame(Apps = College.test$Apps,pred = exp(predict(best_step, cTest)), upr = exp(li

    PP2 = ggplot(CIdf_lm_trans,aes(x=pred, y=Apps)) +
    geom_ribbon(aes(ymin = lwr, ymax = upr), fill = "blue", alpha = 0.2) +  geom_point(aes(y=Apps)) +
      xlab("Predicted Number of Applications") +
    ylab("Observed Number of Applications") +
    ggtitle("95% Prediction Interval under the Linear Trans Model")
##poission CI plot

CIdf_pois = data.frame(Apps = College.test$Apps, pred = predict(fit_poisson, College.test[-2], type = '

    PP3 = ggplot(CIdf_pois,aes(x=pred, y=Apps)) +
    geom_ribbon(aes(ymin = lwr, ymax = upr), fill = "blue", alpha = 0.2) +  geom_point(aes(y=Apps)) +
      xlab("Predicted Number of Applications") +
    ylab("Observed Number of Applications") +
    ggtitle("95% Prediction Interval under the Poisson Model")


##poisson_trans CI plot
```

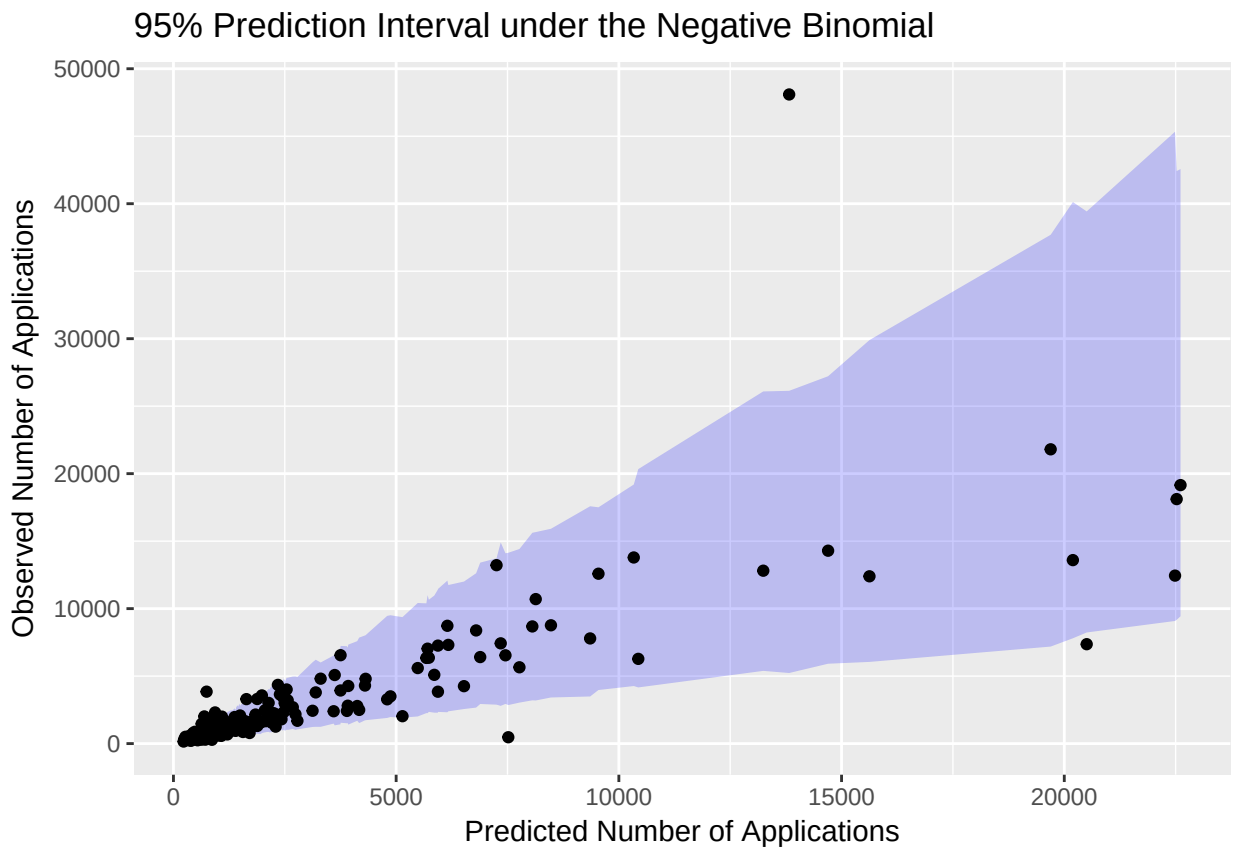
```
CIdf_pois_trans = data.frame(Apps = College.test$Apps, pred = predict(fit_poisson2, cTest, type = 'resp

    PP4 = ggplot(CIdf_pois_trans,aes(x=pred, y=Apps)) +
    geom_ribbon(aes(ymin = lwr, ymax = upr), fill = "blue", alpha = 0.2) +  geom_point(aes(y=Apps)) +
      xlab("Predicted Number of Applications") +
    ylab("Observed Number of Applications") +
    ggtitle("95% Prediction Interval under the Poisson Transformation Model")


grid.arrange(PP1,PP2)
```




```
grid.arrange(PP3,PP4)
```




```
grid.arrange(PP5)
```




```
## coverage function
coverage = function(object, newdata, nsim){
    nsim = n_sim =1000
    CI=pi(object,newdata)

    #return(crage(newdata$Apps, CI))
    return(crage(y = newdata$Apps, CI))
}

### retrun y.rep matxi
YREP = function(object, newdata, level = 0.95, nsim =1000){
  n = nrow(newdata)
  X = model.matrix(object, data = newdata)
  y.rep = matrix(NA, nsim, n)

  if (class(object)[1] == "negbin"){
    beta = rmvnorm(nsim, coef(object), vcov(object))
    theta = rnorm(nsim, object$theta, object$SE.theta)
    for (i in 1:nsim){
      mu = exp(X %*% beta[i,])
      y.rep[i,] = rnegbin(n, mu=mu, theta=theta[i])
    }
  }

  if(class(object)[1] == "glm"){
    sim_object = sim(object, nsim)
```

```
    beta = sim_object@coef
    for (i in 1:nsim){
      mu = exp(X %*% beta[i,])
      y.rep[i,] <- rpois(n, mu)
    }
  }

  if(class(object)[1] == "lm"){
    sim_object = sim(object, nsim)
    beta = sim_object@coef
    for(i in 1:nsim){
      mu = (X %*% beta[i,])
      y.rep[i, ] <- rnorm(n, mu, sim_object@sigma[i])
    }
  }


  return(y.rep)
}
```

All of the five models did not capture one university, Rutgers at New Brunswick, which received applications the most. In general, negative binomial model captures the best coverage among all five models, since it has the largest ribbon.

```
# test data
ctest2 = College.test
ctest2$F.Undergrad = log(ctest2$F.Undergrad)
ctest2$PhD = log(ctest2$PhD)
ctest2$Grad.Rate = log(ctest2$Grad.Rate)
ctest2$Top10perc = log(ctest2$Top10perc)
fit_nb = glm.nb(Apps ~ . + (Personal + F.Undergrad + Books)^2 , data=ctnb )
# coverage
linear = coverage(lm_fit1,College.test)
linear_trans = coverage(best_step,ctest1)
Possion= coverage(fit_poisson, College.test)
possion_trans = coverage(fit_poisson2, cTest_poi)
ng_trans = coverage(fit_nb, cTest_poi)
```

The coverages of each model are summerized by the following table:

```
df.coverage=as.data.frame(c(linear,linear_trans,Possion,possion_trans,ng_trans))
df.coverage=t(df.coverage)
colnames(df.coverage)=c("linear","linear.with.transformation","poisson","poisson.with.transformation","
rownames(df.coverage)="Coverage"
kable(df.coverage)
```

	linear	linear.with.transformation	poisson	poisson.with.transformation	NB.with.transformation
Coverage	0.9282051	0.9384615	0.0461538	0.0615385	0.9333333

From the perspective of coverage of confidence interval, we find that linear model, the poisson model and model of negative binomial do well, since more than 90% of the confidence interval contain test cases.

13. Provide a table with the 1) RMSE's on the observed data, 2) RMSE's on the test data, 3) coverage, 4) the predictive check p-value with one row for each of the models and comment the results. Which

model do you think is best and why? Consider the job of an administrator who wants to ensure that there are enough staff to handle reviewing applications. Explain why coverage might be useful.

```
# test data
ctt = College.train
ctt$F.Undergrad = log(ctt$F.Undergrad)
ctt$PhD = log(ctt$PhD)
ctt$Grad.Rate = log(ctt$Grad.Rate)
ctt$Top10perc = log(ctt$Top10perc)
ctt$Apps = log(ctt$Apps)


# RMSE on observed data
rmse_ng_train = rmse(College.train$Apps, fitted(fit_nb))
rmse_pos_train_train = rmse(ctpoisson$Apps, predict(fit_poisson2, ctpoisson[-2], type = 'response'))
rmse_pos_train = rmse(College.train$Apps, predict(fit_poisson, College.train[-2], type = 'response'))
rmse_lm_trans_train = rmse(exp(ctt$Apps), exp(predict(best_step, ctt[-2])))
rmse_lm_train = rmse(College.train$Apps, predict(lm_fit1, College.train[-2]))

# RMSE on test data
rmse_ng_test = rmse(ctest2$Apps, predict(fit_nb, ctest2[-2], type = 'response'))
rmse_pos_trans_test = rmse(ctest2$Apps, predict(fit_poisson2, ctest2[-2], type = 'response'))
rmse_pos_test = rmse(College.test$Apps, predict(fit_poisson, College.test[-2], type = 'response'))
rmse_lm_trans_test = rmse(exp(ctest1$Apps), exp(predict(best_step, ctest1[-2])))
rmse_lm_test = rmse(College.test$Apps, predict(lm_fit1, College.test[-2]))

## p-value
yrep_np = YREP(fit_nb,ctnb)
a = apply(yrep_np,1, function(x) rmse(x, ctnb$Apps))
p1 = ifelse(mean(a > rmse_ng_train) < 0.5, 2*mean(a > rmse_ng_train), 2*(1-mean(a > rmse_ng_train)))
yrep_np1 = YREP(fit_poisson,College.train)
b = apply(yrep_np1,1, function(x) rmse(x, College.train$Apps))
p2 = ifelse(mean(b > rmse_pos_train) < 0.5, 2*mean(b > rmse_pos_train), 2*(1-mean(b > rmse_pos_train)))
yrep_np2 = YREP(fit_poisson2,ctpoisson)
C = apply(yrep_np2,1, function(x) rmse(x, ctpoisson$Apps))
p3 = ifelse(mean(C > rmse_pos_train_train) < 0.5, 2*mean(C > rmse_pos_train_train), 2*(1-mean(C > rmse_
yrep_np3 = YREP(lm_fit1,College.train)
d = apply(yrep_np3,1, function(x) rmse(x, College.train$Apps))
p4 = ifelse(mean(d > rmse_lm_train) < 0.5, 2*mean(d > rmse_lm_train), 2*(1-mean(d > rmse_lm_train)))
yrep_np4 = exp(YREP(best_step,ctt))
e = apply(yrep_np4,1, function(x) rmse(x, exp(ctt$Apps)))
p5 = ifelse(mean(e > rmse_lm_trans_train) < 0.5, 2*mean(e > rmse_lm_trans_train), 2*(1-mean(e > rmse_lm
```

```
# build table
Table = data.frame(matrix(0,ncol = 5, nrow = 4))
Table[,1] = rbind(rmse_lm_train, rmse_lm_test,linear,p3)
Table[,2] = rbind(rmse_lm_trans_train, rmse_lm_trans_test, linear_trans,p1)
Table[,3] = rbind(rmse_pos_train,rmse_pos_test,Possion,p4)
Table[,4] = rbind(rmse_pos_train_train,rmse_pos_trans_test, possion_trans,p1)
Table[,5] = rbind(rmse_ng_train,rmse_ng_test, ng_trans,p2)
names(Table) = c("linear", "linear_transform","poisson", "poisson_transform","negative_binomial")

rownames(Table) = c("observed_data","test_data","coverage","p_value")
kable(Table)
```

	linear	linear_transform	poisson	poisson_transform	negative_binomial
observed_data	1480.3035956	1223.8678893	2.049569e+03	1.179209e+03	1232.3880424
test_data	2929.3505915	2944.9653983	2.875510e+03	2.807608e+03	3030.1012326
coverage	0.9282051	0.9384615	4.615385e-02	6.153846e-02	0.9333333
p_value	0.6800000	0.0000000	0.000000e+00	0.000000e+00	0.8920000

The table shows the coverages for each of the 5 models. In linear and negative binomial models, the intervals captured almost 94% of the test values, whereas in the Poisson models the intervals captured only around 4% to 8%. Compared to other models, the two Poission models have too narrow interval, this is because Poisson model does not have scale variable and usually causes overdispersion. Comparing their observed RMSE and test data RMSE, we find the difference between two RMSE values are close in each case and pvalues. Negative Binomail model performs the best among all others. Thus the best model we choose is negative binomial model

14. For your "best" model provide a nicely formatted table (use `kable()` or `xtable()`) of relative risks and 95% confidence intervals. Pick 5 of the most important variables and provide a paragraph that provides an interpretation of the parameters (and intervals) that can be provided to a university admissions officer about which variables increase admissions.

```
DF = exp(cbind(coef(fit_nb), confint(fit_nb)))
```

```
## Waiting for profiling to be done...
```

```
colnames(DF)[1] = 'Coefficient'

kable(DF)
```

	Coefficient	2.5 %	97.5 %
(Intercept)	0.0303892	0.0089080	0.1044938
PrivateYes	1.1571149	1.0301026	1.2992520
Top10perc	0.9729683	0.9103446	1.0390935
F.Undergrad	3.0875479	2.6816361	3.5571794
Room.Board	1.0000913	1.0000551	1.0001276
Books	1.0015977	1.0000330	1.0032021
Personal	1.0006141	1.0002051	1.0010233
PhD	1.3839390	1.1952707	1.5968795
S.F.Ratio	0.9958326	0.9858961	1.0059275
perc.alumni	1.0030471	0.9997450	1.0063723
Grad.Rate	1.0822596	0.9534256	1.2259230
EliteYes	1.3227936	1.1630723	1.5074327
F.Undergrad:Personal	0.9999319	0.9998842	0.9999799
Books:Personal	0.9999997	0.9999994	1.0000000
F.Undergrad:Books	0.9998684	0.9996476	1.0000874

```
a = summary(fit_nb)
coef = data.frame(a$coefficients)
coef = cbind(rownames(coef),coef)

colnames(coef)[5] = "p_value"
coef = coef %>% arrange(p_value)
kable(coef)
```

rownames(coef)	Estimate	Std..Error	z.value	p_value
F.Undergrad	1.1273772	0.0698032	16.1507924	0.0000000
(Intercept)	-3.4936690	0.6195189	-5.6393260	0.0000000
Room.Board	0.0000913	0.0000180	5.0749426	0.0000004
PhD	0.3249337	0.0718069	4.5251060	0.0000060
EliteYes	0.2797458	0.0666077	4.1999054	0.0000267
Personal	0.0006139	0.0002085	2.9446962	0.0032327
F.Undergrad:Personal	-0.0000681	0.0000243	-2.8019482	0.0050795
PrivateYes	0.1459298	0.0588706	2.4788219	0.0131817
Books:Personal	-0.0000003	0.0000001	-2.2186541	0.0265103
Books	0.0015964	0.0007752	2.0593410	0.0394616
perc.alumni	0.0030424	0.0017017	1.7878568	0.0737991
F.Undergrad:Books	-0.0001317	0.0001078	-1.2211305	0.2220366
Grad.Rate	0.0790511	0.0670929	1.1782340	0.2387033
S.F.Ratio	-0.0041761	0.0049319	-0.8467603	0.3971288
Top10perc	-0.0274038	0.0350328	-0.7822326	0.4340779

We can get confidence intervals for the parameters and the exponentiated parameters. For the negative binomial model, these would be relative risk.

According to the p-value, the top 5 most important variables are `F.Undergrad`, `Room.Board`, `PhD`, `EliteYes`, `Personal`. These variables have positive influences to the amount of applications. Generally, if there are more full time students in the university, there will be larger amount of applications. If the Pct. of faculty with Ph.D.'s is higer and the school is classfied as elite school(To10perc > 50), which means the university owns better educational quality, amount of applications may be higer. In the case of `Room.Board` and `Personal`, the higer value of these variables might indicate the university is located in city. Students maybe more interested in the university in the city.

Some Theory

15. Gamma mixtures of Poissons: From class we said that

$$Y \mid \lambda \sim P(\lambda) \tag{1}$$

$$p(y \mid \lambda) = \frac{\lambda^y e^{-\lambda}}{y!} \tag{2}$$

$$\tag{3}$$

$$\lambda \mid \mu, \theta \sim G(\theta, \theta/\mu) \tag{4}$$

$$p(\lambda \mid \mu, \theta) = \frac{(\theta/\mu)^\theta}{\Gamma(\theta)} \lambda^{\theta-1} e^{-\lambda\theta/\mu} \tag{5}$$

$$\tag{6}$$

$$p(Y \mid \mu, \theta) = \int p(Y \mid \lambda) p(\lambda \mid \theta, \theta/\mu) d\lambda \tag{7}$$

$$= \frac{\Gamma(y+\theta)}{y!\Gamma(\theta)} \left(\frac{\theta}{\theta+\mu}\right)^\theta \left(\frac{\mu}{\theta+\mu}\right)^y \tag{8}$$

$$Y \mid \mu, \theta \sim NB(\mu, \theta) \tag{9}$$

Derive the density of $Y \mid \mu, \theta$ in (8) showing your work using LaTeX expressions. (Note this may not display if the output format is html, so please use pdf.) Using iterated expectations with the Gamma-Poisson mixture, find the mean and variance of Y, showing your work.

$$
\begin{aligned}
p(Y \mid \mu, \theta) &= \int p(Y \mid \lambda) p(\lambda \mid \theta, \theta/\mu) d\lambda \\
&= \int \frac{\lambda^y e^{-\lambda}}{y!} \cdot \frac{(\theta/\mu)^\theta}{\Gamma(\theta)} \lambda^{\theta-1} e^{-\lambda\theta/\mu} d\lambda \\
&= \frac{(\theta/\mu)^\theta}{y!\ \Gamma(\theta)} \int \lambda^{\theta+y-1} e^{-\lambda(\frac{\theta}{\mu}+1)} d\lambda
\end{aligned}
$$

$\lambda^{\theta+y-1} e^{-\lambda(\frac{\theta}{\mu}+1)}$ is the probability density function of $Gamma(\lambda; \theta+y, \frac{\theta+\mu}{\mu})$, so its normalizing constant is $\frac{\Gamma(\theta+y)}{\Big((\theta+\mu)/\mu\Big)^{\theta+y}}$

$$
\begin{aligned}
p(Y \mid \mu, \theta) &= \frac{(\theta/\mu)^\theta}{y!\ \Gamma(\theta)} \cdot \frac{\Gamma(\theta+y)}{\Big((\theta+\mu)/\mu\Big)^{\theta+y}} \\
&= \frac{\Gamma(\theta+y)}{y!\ \Gamma(\theta)} \cdot \left(\frac{\theta}{\mu} \cdot \Big(\frac{\mu}{\theta+\mu}\Big) \right)^\theta \cdot \Big(\frac{\mu}{\theta+\mu}\Big)^y \\
&= \frac{\Gamma(\theta+y)}{y!\ \Gamma(\theta)} \Big(\frac{\theta}{\theta+\mu}\Big)^\theta \Big(\frac{\mu}{\theta+\mu}\Big)^y
\end{aligned}
$$

$$
\begin{aligned}
E[Y] &= E\Big[E[Y|\lambda]\Big] = E[\lambda] = \theta \cdot (\frac{\theta}{\mu})^{-1} = \mu \\
Var[Y] &= Var\Big[E[Y|\lambda]\Big] + E\Big[Var[Y|\lambda]\Big] = Var[\lambda] + E[\lambda] = \theta \cdot \Big(\frac{\theta}{\mu}\Big)^{-2} + \mu = \mu + \frac{\mu^2}{\theta}
\end{aligned}
$$