Analysis of Average School SAT Scores in New-York City

Ryad Taleb

1 Introduction

This paper will explore factors related to average SAT scores in New-York City. A total of 435 schools across all 5 Boroughs will be considered along with their average SAT scores for the 2014-2015 school year. The dataset includes variables such as enrollment and ethnic percentages, and our goal is to determine which of these factors are the best predictors of average school SAT scores. It will include descriptive statistics of the dataset used, a literary review of similar studies, the proposed methods, and a results section.

1.1 Descriptive Statistics

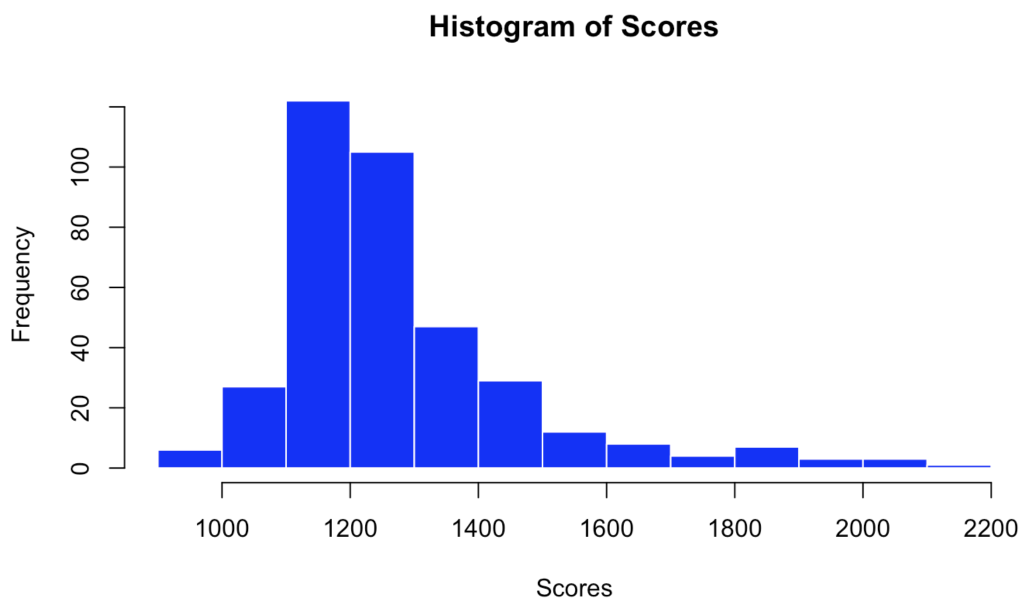


Figure 1: Histogram plot of average school SAT scores. There is a right skew

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

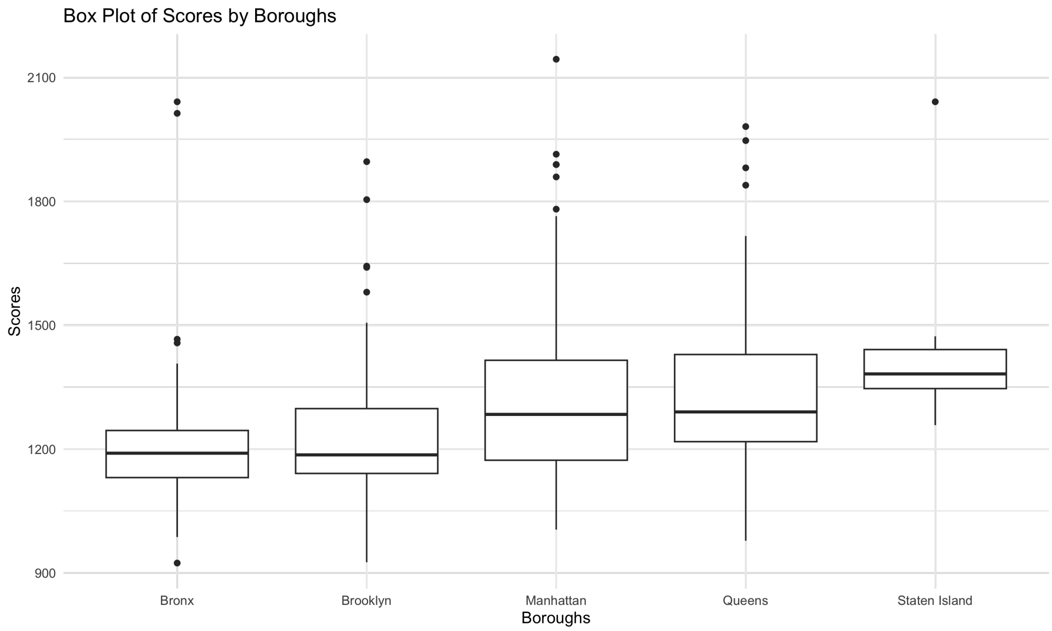
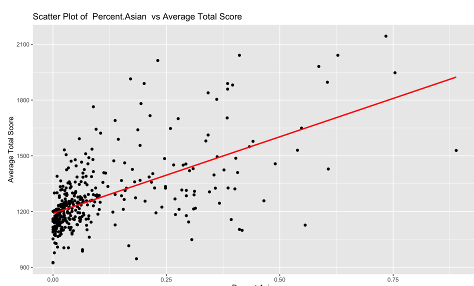
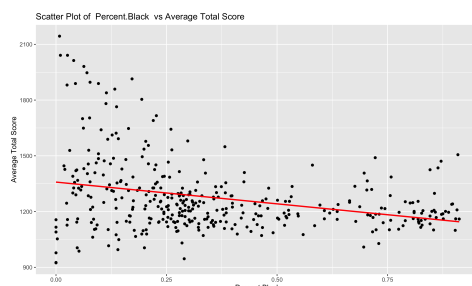


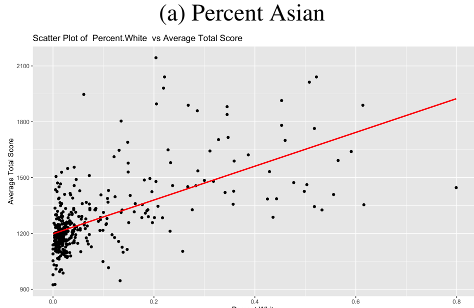
Figure 2: Average School SAT scores by Borough



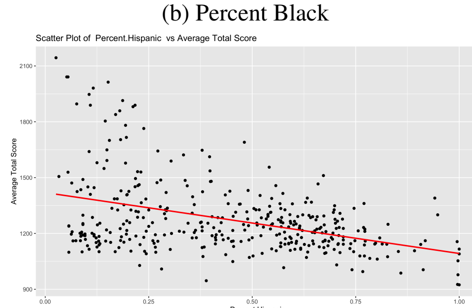
(a) Percent Asian



(b) Percent Black



(c) Percent White



(d) Percent Hispanic

Figure 3: Average School SAT scores plotted Against ethnic percentages

1.2 Data Cleaning / Modification

Several changes were made to the dataset before using it for analysis. Schools' start and end times were given in hh:mm AM/PM format in 2 separate columns. School duration was then calculated and the 2 previous columns were dropped. Any row (school) with NA values was also removed. Total SAT score had to be calculated and made into its own column. (individual Reading/Writing/Math scores were originally given). Some column data-types had to be changed. The categorical variable of Borough location was dummy coded with Staten-Island being the excluded variable to avoid singularity.

The variables tested are then: School duration, School Enrollment, School Ethnic Percentages (Black, White, Asian, Hispanic), and Borough (Brooklyn, Queens, Manhattan, Staten Island, Bronx).

2 Literature Review

A study conducted by the NYC Data Science Academy analyzed New York Public School SAT scores over the same year (2014-2015). They used Multiple Linear Regression and Random Forest models.

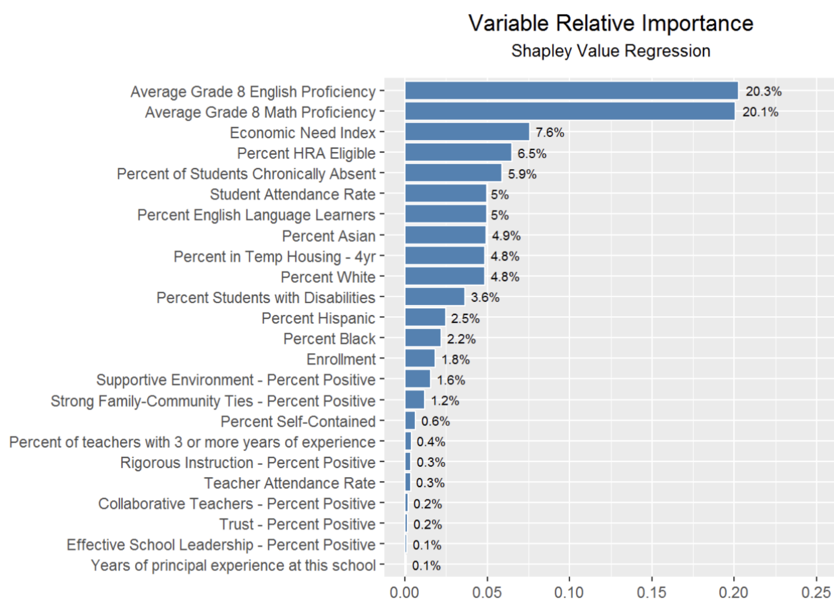


Figure 4: Graph from NY Data Science Academy showing Relative Importance of variables. The data set they used contained more variables, but we can see variables that we analysed such as the ethnic percentages and enrollment.

Another study by the Brookings Institution analyzed the racial disparities in SAT scores. They found average Math SAT scores were notably lower for Black and Hispanic students than White and Asian students. They mention socioeconomic factors and educational access as potential reasons.

3 Proposed Method

Due to the apparent linear dependence of SAT score on the variables, we will perform linear regression with Bayesian Estimation. To then determine the most important factors we will perform Bayesian Model Selection.

3.1 Bayesian Estimation

We will use the normal linear regression model:

$\epsilon_1, \ldots, \epsilon_n \sim$ i.i.d. $\mathcal{N}(0, \sigma^2)$
$Y_i = \beta^T x_i + \epsilon_i$

$y \mid X, \beta, \sigma^2 \sim \mathcal{N}(X\beta, \sigma^2 I)$

Our likelihood is then:

$p(y_1, \ldots, y_n \mid x_1, \ldots, x_n, \beta, \sigma^2) = \prod_{i=1}^n p(y_i \mid x_i, \beta, \sigma^2)$
$= \left(\frac{1}{(2\pi\sigma^2)^{n/2}}\right) \exp\left\{-\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \beta^T x_i)^2\right\}.$

Or in matrix notation:

$$p(y \mid X, \beta, \sigma^2) \propto \exp\left\{-\frac{1}{2\sigma^2}[y^T y - 2\beta^T X^T y + \beta^T X^T X\beta]\right\}.$$

Our prior for beta is also multivariate normal:

$\beta \sim \mathcal{N}(\beta_0, \Sigma_0)$

Then:
$p(\beta \mid y, X, \sigma^2) \propto p(y \mid X, \beta, \sigma^2) \times p(\beta)$
$\propto \exp\left\{\beta^T \left(\Sigma_0^{-1}\beta_0 + \frac{X^T y}{\sigma^2}\right) - \frac{1}{2}\beta^T \left(\Sigma_0^{-1} + \frac{X^T X}{\sigma^2}\right)\beta\right\}.$

We can recognize this as multivariate normal with:

$\text{Var}[\beta \mid y, X, \sigma^2] = \left(\Sigma_0^{-1} + \frac{X^T X}{\sigma^2}\right)^{-1}$
$E[\beta \mid y, X, \sigma^2] = \left(\Sigma_0^{-1} + \frac{X^T X}{\sigma^2}\right)^{-1} \left(\Sigma_0^{-1}\beta_0 + \frac{X^T y}{\sigma^2}\right)$

The semi-conjugate prior for σ^2 is an inverse gamma distribution:

let $\gamma = 1/\sigma^2$, and
$\gamma \sim \text{Gamma}\left(\frac{\nu_0}{2}, \frac{\nu_0\sigma_0^2}{2}\right)$
the posterior for σ^2 is :

$\sigma^2 \mid y, X, \beta \sim \text{Inverse-Gamma}\left(\frac{\nu_0+n}{2}, \frac{\nu_0\sigma_0^2+\text{SSR}(\beta)}{2}\right)$

where $\text{SSR}(\beta) = (y - X\beta)^T(y - X\beta)$

We then perform Gibb's Sampling as follows:

Given current values $\{\beta^{(s)}, \sigma^{2(s)}\}$:

1. Updating β:

a) Compute $V = \text{Var}[\beta \mid y, X, \sigma^{2(s)}]$ and $m = E[\beta \mid y, X, \sigma^{2(s)}]$.
b) Sample $\beta^{(s+1)} \sim \mathcal{N}(m, V)$.

2. Updating σ^2:

a) Compute $\text{SSR}(\beta^{(s+1)})$ as $(y - X\beta^{(s+1)})^T(y - X\beta^{(s+1)})$.
b) Sample $\sigma^{2(s+1)} \sim \text{Inverse-Gamma}\left(\frac{\nu_0+n}{2}, \frac{\nu_0\sigma_0^2+\text{SSR}(\beta^{(s+1)})}{2}\right)$.

3.2 Bayesian Model Selection

Let $\beta_j = z_j \times b_j, \quad$ where $z_j \in \{0,1\}$. (The $z_j's$ act as on-off switches for the factors)

$y_i = z_1 b_1 x_{i,1} + \cdots + z_p b_p x_{i,p} + \epsilon_i$

We need to obtain a posterior distribution for our regression models:

$$p(\mathbf{z} \mid y, X) = \frac{p(\mathbf{z}) p(y \mid X, \mathbf{z})}{\sum_{\tilde{\mathbf{z}}} p(\tilde{\mathbf{z}}) p(y \mid X, \tilde{\mathbf{z}})}$$

However the denominator is too large to compute. Instead we will consider the ratio of 2 model probabilities:

$$\frac{p(\mathbf{z_a} \mid y, X)}{p(\mathbf{z_b} \mid y, X)} = \frac{p(\mathbf{z_a})}{p(\mathbf{z_b})} \times \frac{p(y \mid X, \mathbf{z_a})}{p(y \mid X, \mathbf{z_b})}$$

Posterior Odds = Prior Odds * Bayes Factor

We need to calculate Bayes Factor:

$$p(y \mid X, z) = \int\int p(y, \beta, \sigma^2 \mid X, z)\, d\beta\, d\sigma^2 = \int\int p(y \mid \beta, X) p(\beta \mid X, z, \sigma^2) p(\sigma^2)\, d\beta\, d\sigma^2$$

Which gives us:

$$p(y \mid X, z) = \pi^{-n/2} \frac{\Gamma\left(\frac{\nu_0 + n}{2}\right)}{\Gamma\left(\frac{\nu_0}{2}\right)} (1+g)^{-p_z/2} \frac{(\nu_0 \sigma_0^2)^{\nu_0/2}}{(\nu_0 \sigma_0^2 + \text{SSR}_z)^{(\nu_0+n)/2}}$$

Where:

$$p_z = \sum_{i=1}^{p} z_j$$

and

$$\text{SSR}_z^g = y^T \left(I - \frac{g}{g+1} X_z (X_z^T X_z)^{-1} X_z^T \right) y.$$

Bayes Factor is then:

$$\frac{p(y \mid X, z_a)}{p(y \mid X, z_b)} = (1+n)^{\frac{p_{z_b} - p_{z_a}}{2}} \left(\frac{s_{z_a}^2}{s_{z_b}^2} \right)^{\frac{1}{2}} \times \left(\frac{s_{z_b}^2 + \text{SSR}_{z_b}^g}{s_{z_a}^2 + \text{SSR}_{z_a}^g} \right)^{\frac{n+1}{2}}$$

Let $\mathbf{z}_{-j}$ be the model $\mathbf{z}$ without factor j. We then calculate the conditional odds o_j that z_j is 1:

$$o_j = \frac{p(y \mid X, z_{-j}, z_j = 1)}{p(y \mid X, z_{-j}, z_j = 0)} \times \frac{\Pr(z_j = 1)}{\Pr(z_j = 0)}$$

And

$$\Pr(z_j = 1 \mid y, X, z_{-j}) = \frac{o_j}{1 + o_j}$$

We then construct a Gibb's Sampler:

Given $z^{(s)}$ generate $\{z^{(s+1)}, \sigma^{2(s+1)}, \beta^{(s+1)}\}$ as follows:

1. Set $z = z^{(s)}$.
2. For j in $\{1, \ldots, p\}$ in random order, replace z_j with a sample from $p(z_j \mid z_{-j}, y, X)$.
3. Set $z^{(s+1)} = z$.
4. Sample $\sigma^{2(s+1)} \sim p(\sigma^2 \mid z^{(s+1)}, y, X)$.
5. Sample $\beta^{(s+1)} \sim p(\beta \mid z^{(s+1)}, \sigma^{2(s+1)}, y, X)$.

3.3 Implementation

For Bayesian estimation our prior for beta is $\beta \sim MVN(0, \Sigma_0)$, where Σ_0 is a diagonal matrix with 100 for all diagonal elements. (meant to reflect a weak prior, our data is standardized). It was then run for 5000 MCMC steps.

Model Selection was run for 10,000 MCMC steps.

4 Data Analysis and Results

4.1 Bayesian Estimation Results

We compared the Bayesian estimation model to least squares and they were almost identical:

```
Call:
lm(formula = y ~ ., data = df_OLS)

Residuals:
    Min      1Q  Median      3Q     Max 
-377.87  -60.79   -0.58   56.23  416.30 

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)    
(Intercept)            1275.348      6.028 211.579  < 2e-16 ***
Student.Enrollment       16.893      7.285   2.319 0.020952 *  
Percent.White          -155.682     53.550  -2.907 0.003870 ** 
Percent.Black          -451.347     98.253  -4.594 6.01e-06 ***
Percent.Hispanic       -475.602     92.950  -5.117 5.05e-07 ***
Percent.Asian          -186.500     56.276  -3.314 0.001012 ** 
Duration                 11.485      6.133   1.873 0.061925 .  
BoroughBronx            -11.768      7.857  -1.498 0.135063    
BoroughBrooklyn         -46.622      8.764  -5.320 1.82e-07 ***
BoroughQueens           -38.673      7.795  -4.961 1.08e-06 ***
`BoroughStaten Island`  -27.821      7.343  -3.789 0.000177 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 116.6 on 363 degrees of freedom
Multiple R-squared:  0.6517,	Adjusted R-squared:  0.6421 
F-statistic: 67.93 on 10 and 363 DF,  p-value: < 2.2e-16
```

Figure 5: OLS model summary

Description: df [11 × 3]

	2.5% <dbl>	97.5% <dbl>	means <dbl>
Intercept	1263.3704654	1287.510408	1275.468
Enroll	2.4597970	31.009119	16.626
P.white	−258.1132511	−53.222830	−155.381
P.Black	−641.3748629	−263.104522	−450.917
P.Hisp	−653.8033025	−297.095517	−475.002
P.Asian	−294.5209321	−78.065471	−186.068
Duration	−0.3670514	23.970188	11.527
BoroughBronx	−27.3237417	3.421133	−11.685
BoroughBrooklyn	−63.4871561	−29.511997	−46.471
BoroughQueens	−54.2241542	−23.749506	−38.609
BoroughStatenIsland	−42.6033251	−13.367183	−27.854

Figure 6: Bayesian Estimation model

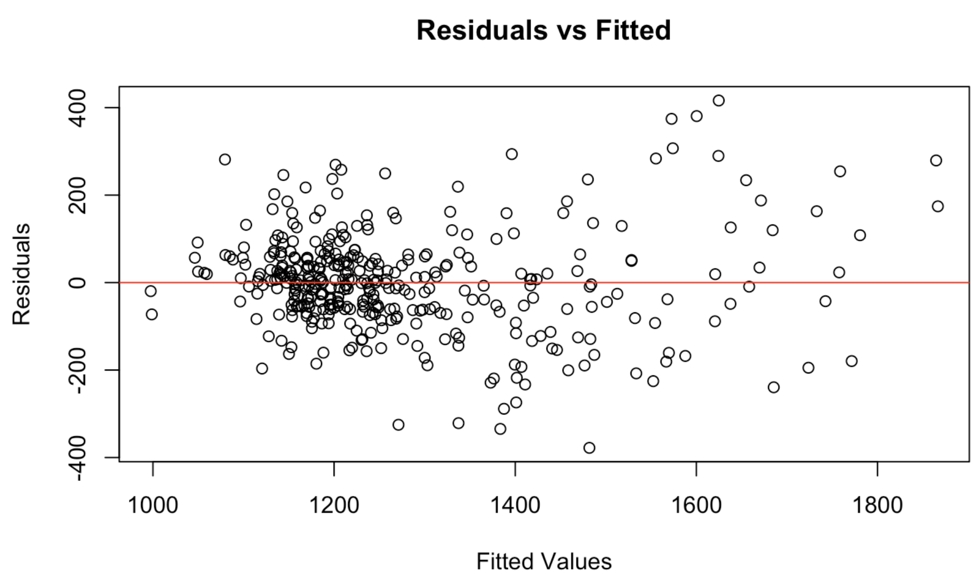


Figure 7: Residuals of the OLS model vs the fitted values

To account for the non-constant variance, a log transform was applied on y, however, there was still significant heteroscedasticity according to the Breusch-Pagan test:

OLS model: BP = 118.15, df = 10, p-value $<$ 2.2e-16

Log(y) model: BP = 69.445, df = 10, p-value = 5.673e-11

4.2 Model Selection Results

:

Description: df [11 × 2]

Variable	Inclusion Probability
<chr>	<chr>
Intercept	1
Enroll	0.1571
P.white	0.205
P.Black	0.999
P.Hisp	0.9999
P.Asian	0.2634
Duration	0.1647
BoroughBronx	0.1047
BoroughBrooklyn	0.9982
BoroughQueens	0.9949
BoroughStatenIsland	0.863

Figure 8: variables and their inclusion probabilities

Percent White, Percent Asian, Duration, and Bronx identifier have low inclusion probabilities.

An 80-20 train-test split was then performed to test the performance of the full model (OLS) and a model only with variables above 0.85 inclusion probability (we removed the 5 variables). The results were:

RMSE for full model: 115.00
RMSE for selected model: 116.47

So with almost half the model complexity our RMSE only increased by 0.012

5 Conclusion

This study used Bayesian data analysis methods to explore the factors of average SAT scores among New York City schools. No significant difference was found between Ordinary Least Squares and Bayesian Estimation. Weak predictors were then removed with Bayesian Model Selection, and analysis between the full model and the reduced model showed a negligible increase in RMSE. (a simpler model can nearly match the performance of a more complex one). Future research might explore additional variables or alternative statistical methods to further refine these insights.

6 References

NYC Data Science. (2016, Oct 6). Data study on NYC public schools SAT scores. NYC Data Science Academy. https://nycdatascience.com/blog/student-works/data-study-on-nyc-public-schools-sat-scores/

Brookings Institution. (2017, Feb 1). Race gaps in SAT scores highlight inequality and hinder upward mobility. Brookings. https://www.brookings.edu/articles/race-gaps-in-sat-scores-highlight-inequality-and-hinder-upward-mobility/

Hoff, P. D. (2009). A first course in Bayesian statistical methods. Springer Science + Business Media. https://doi.org/10.1007/978-0-387-92407-6

R Notebook

Code ▾

This is an R Markdown (http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Cmd+Shift+Enter*.

Hide

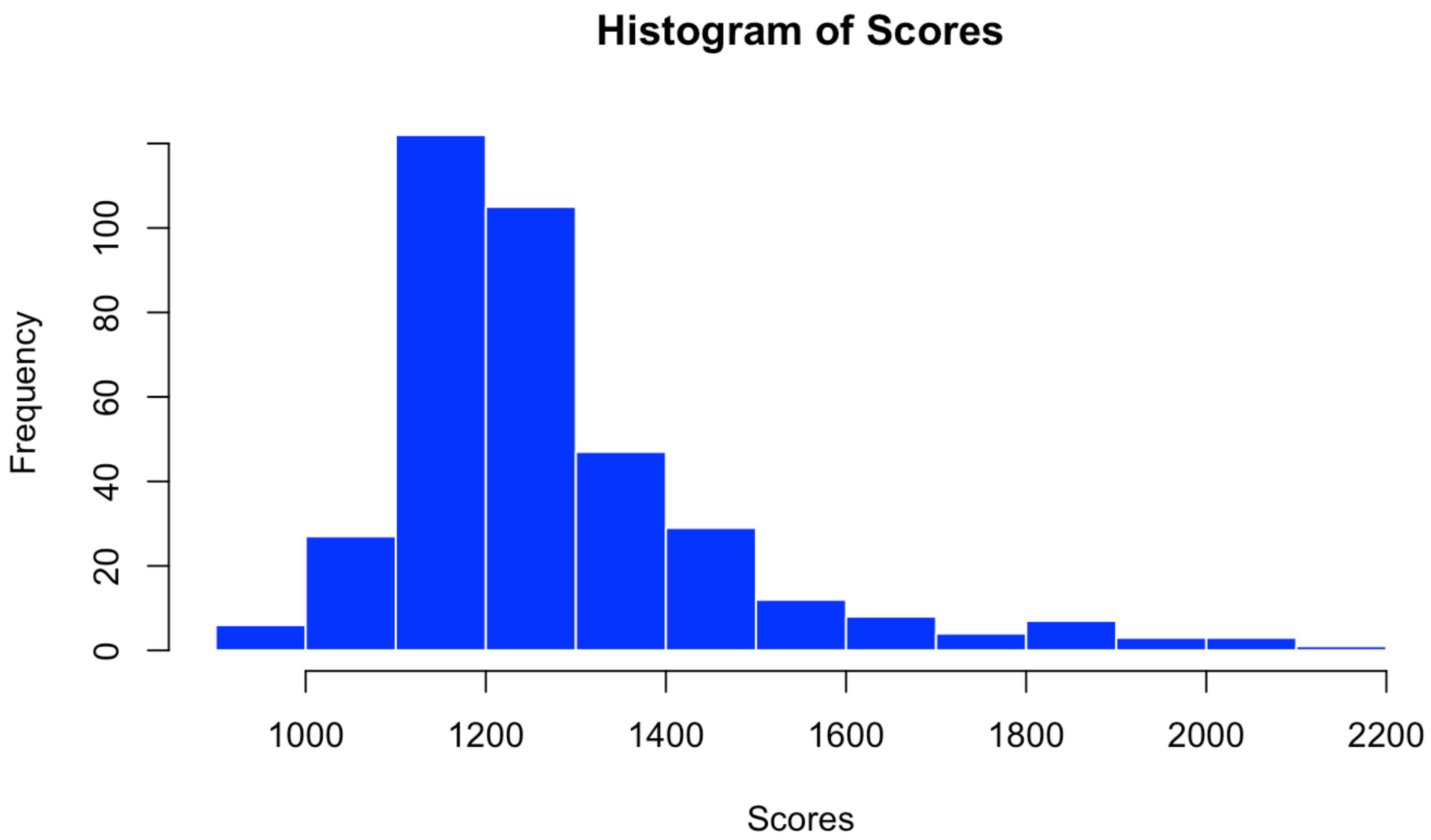
```
library(lubridate)
source("regression_gprior.R")
library(ggplot2)
```

Hide

```
data <- read.csv("scores.csv")
data <- na.omit(data)
#dummy code borough location
borough_dummies <- model.matrix(~ Borough-1, data = data)
borough_dummies <- subset(borough_dummies, select = -BoroughManhattan)

#calculate total school duration
data$Start.Time <- as.POSIXct(data$Start.Time, format="%I:%M %p", tz="UTC")
data$End.Time <- as.POSIXct(data$End.Time, format="%I:%M %p", tz="UTC")

data$Duration <- difftime(data$End.Time, data$Start.Time, units = "mins")
data$Duration <- as.numeric(data$Duration, units = "mins")

#get total SAT score
data$Total = data$`Average.Score..SAT.Reading.` + data$`Average.Score..SAT.Math.` + data$`Average.Score..SAT.Writ
ing.`




#convert to numeric
ls <- c("Average.Score..SAT.Reading.","Average.Score..SAT.Writing.","Average.Score..SAT.Math.","Student.Enrollmen
t")
for (col in ls) {

  data[[col]] <- as.numeric(data[[col]])
}

#fix percentage columns
ls<- c("Percent.Black", "Percent.White", "Percent.Asian", "Percent.Hispanic")
for (col in ls) {

  data[[col]] <- as.numeric(sub("%", "", data[[col]]))
  data[[col]] <- data[[col]] / 100
}

#choose columns
df <- data
df <- cbind(df, borough_dummies)
df <- df[, which(names(df) %in% c("Student.Enrollment", "Percent.White", "Percent.Black","Percent.Hispanic","Perc
ent.Asian","Average.Score..SAT.Math.",
                                  "Average.Score..SAT.Reading.","Average.Score..SAT.Writing.","BoroughBronx",
                                  "BoroughBrooklyn","BoroughManhattan","BoroughQueens","BoroughStaten Island","Du
ration"))]

#remove nan
df <- na.omit(df)


#create y
y <- df$`Average.Score..SAT.Reading.` + df$`Average.Score..SAT.Math.` + df$`Average.Score..SAT.Writing.`

#remove SAT subsections
df <- df[, -which(names(df) %in% c("Average.Score..SAT.Math.","Average.Score..SAT.Reading.","Average.Score..SAT.W
riting."))]

#create design matrix
df <- as.matrix(df)
df <- scale(df)
df <- cbind(1,df)
```

Hide

```
p <- hist(y,
     main = "Histogram of Scores",
     xlab = "Scores",
     col = "blue",
     border = "white")
```



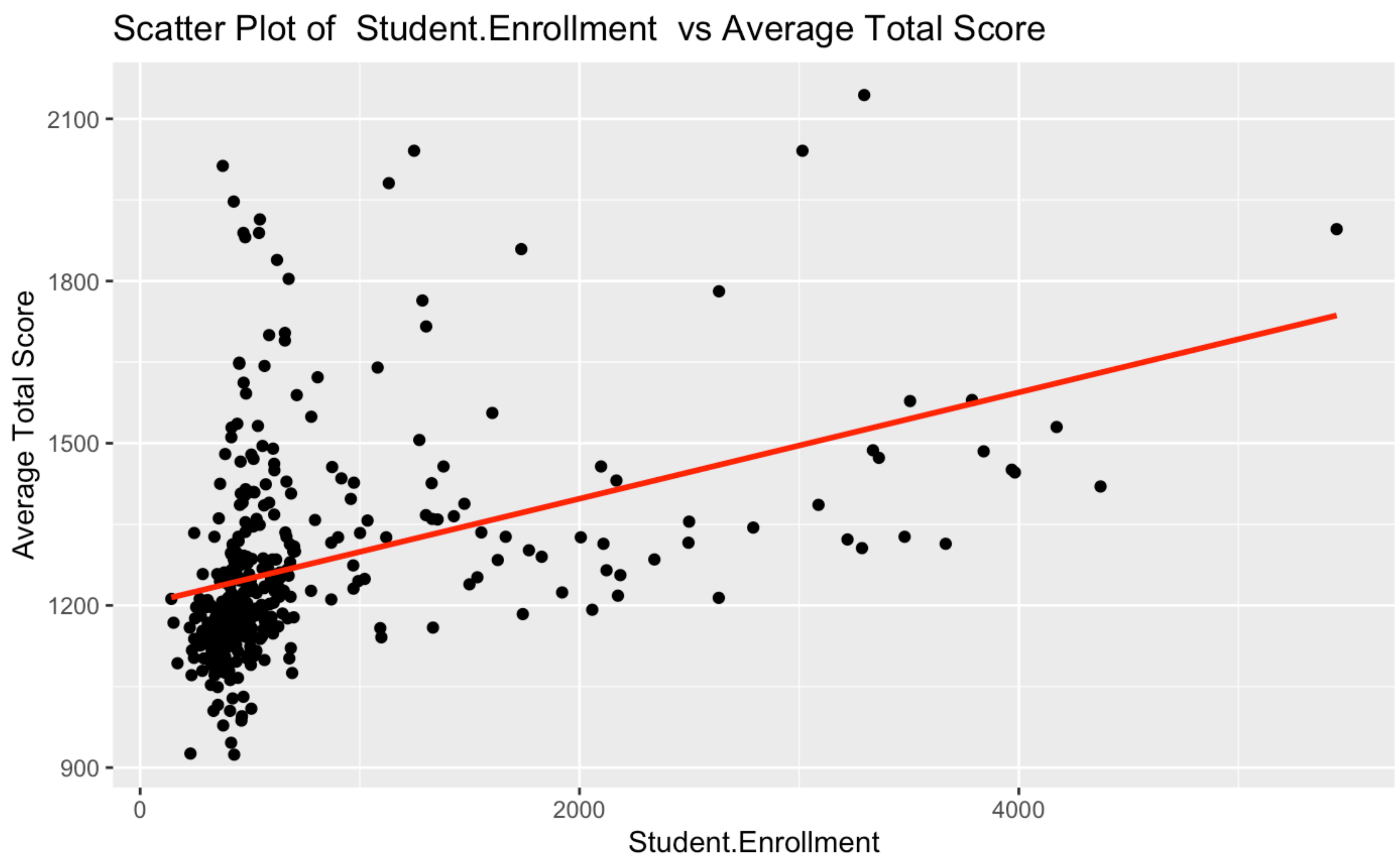
Hide

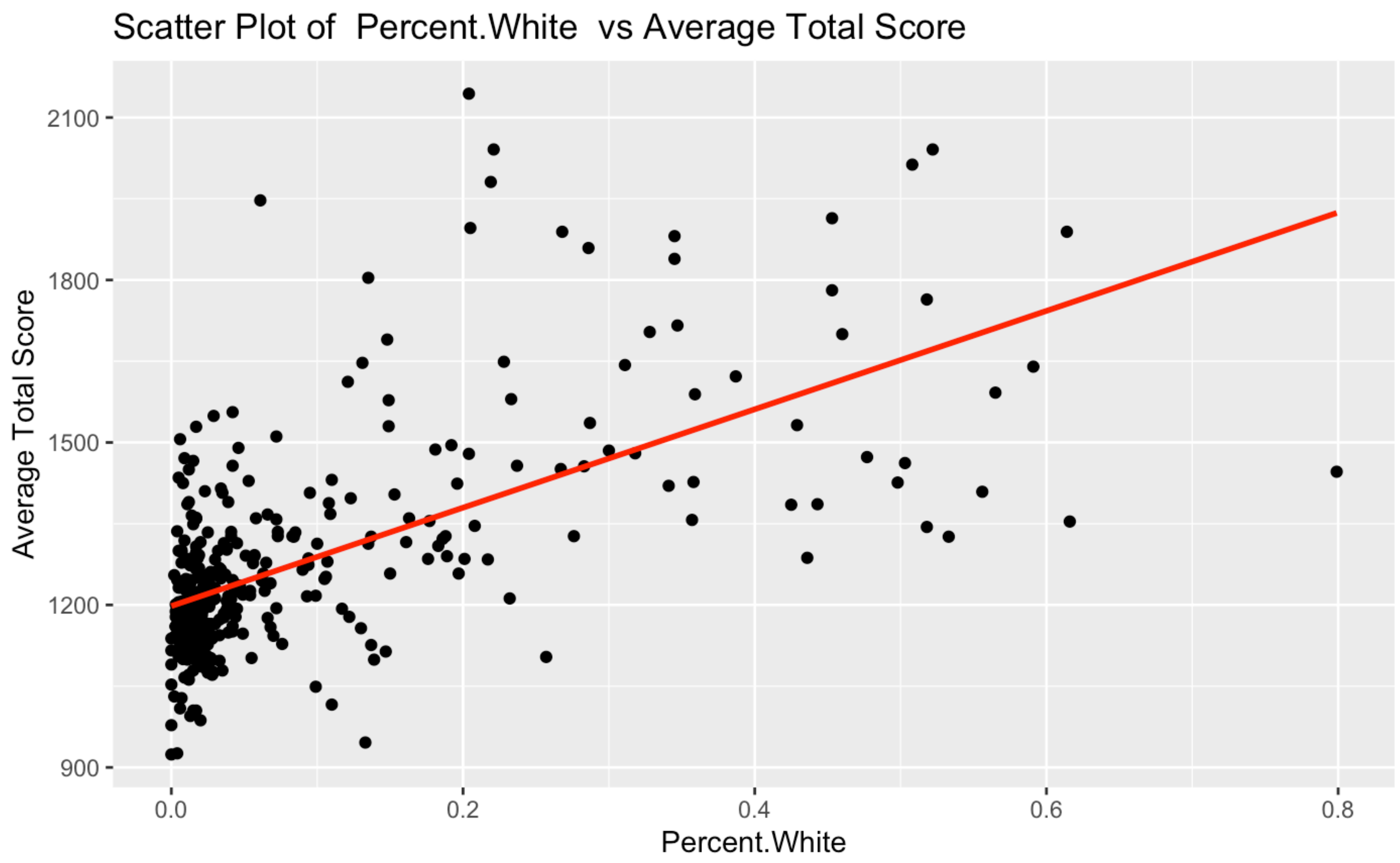
```
p <- ggplot(data = data, aes(x = Borough, y = Total)) +
  geom_boxplot() +
  theme_minimal() +
  labs(title = "Box Plot of Scores by Boroughs",
       x = "Boroughs",
       y = "Scores")
ggsave(filename = paste("Borough_vs_Average_SAT_scores.png"), plot = p, width = 10, height = 6, dpi = 300)
```

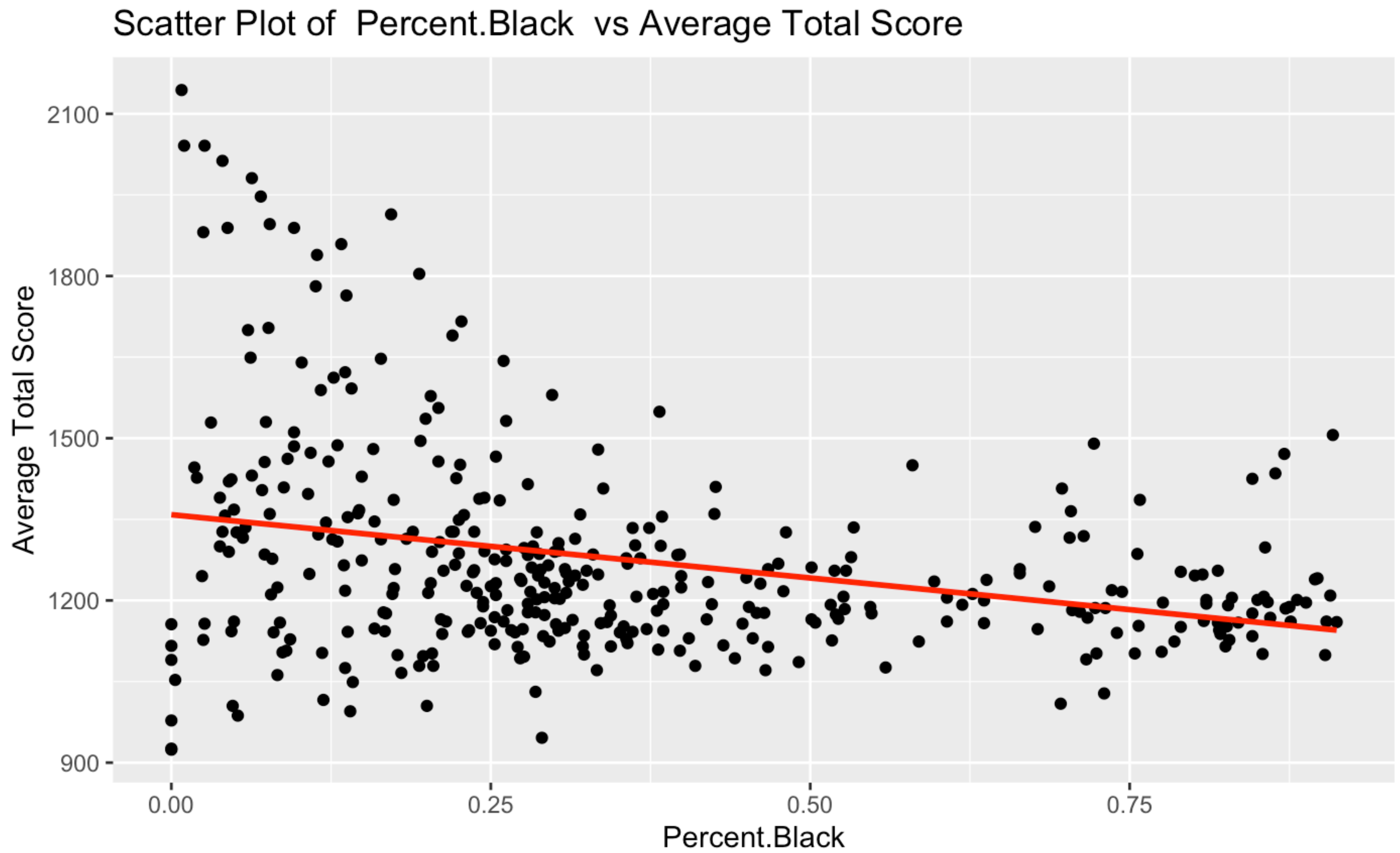
Hide

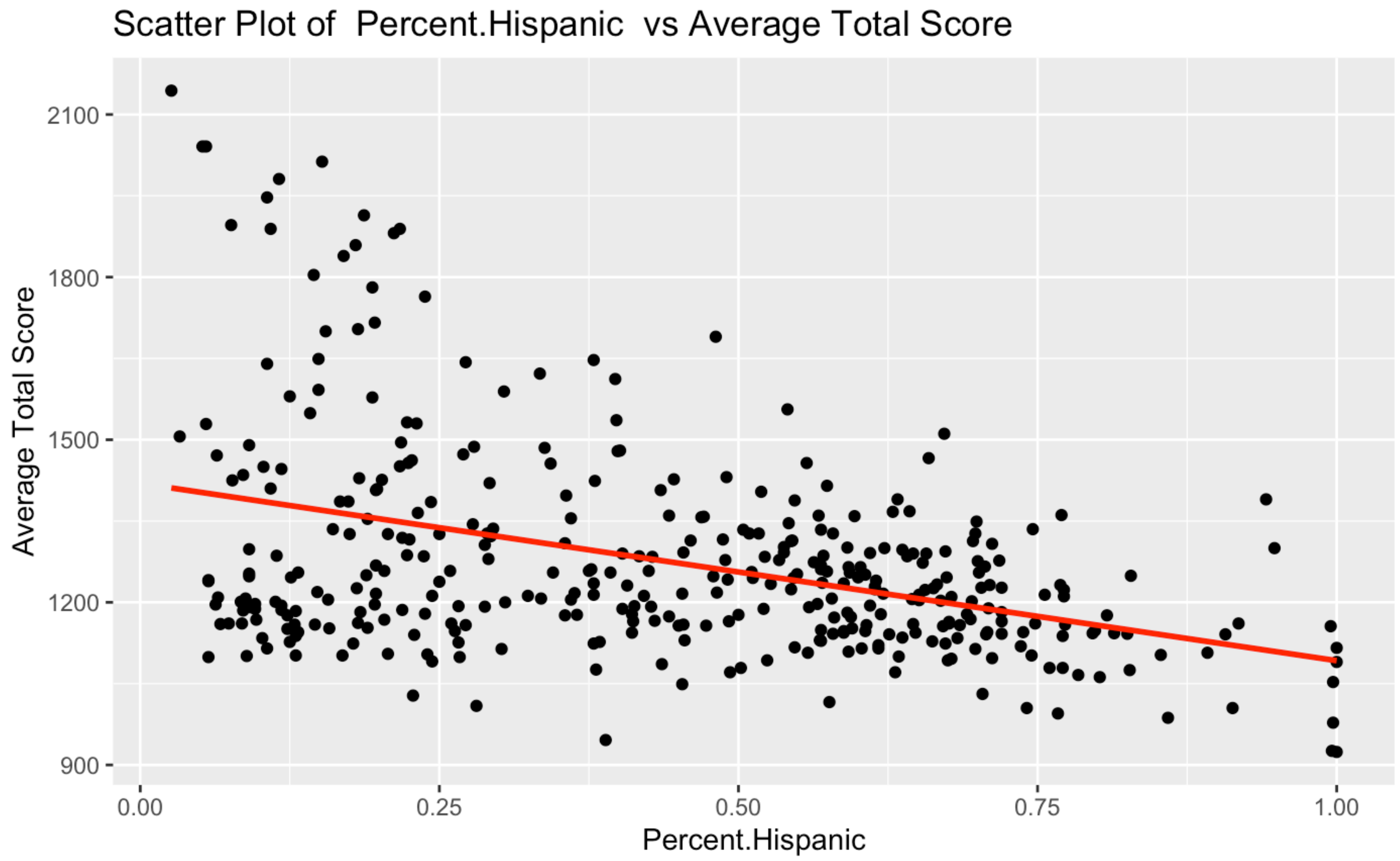
```
columns_to_plot <- c("Student.Enrollment", "Percent.White", "Percent.Black","Percent.Hispanic","Percent.Asian","D
uration")

# Create plots using a loop
for (col in columns_to_plot) {
  p <- ggplot(data, aes_string(x = col, y = "Total")) +
    geom_point() +  # Add points
    geom_smooth(method = "lm", se = FALSE, color = "red") +  # Add regression line
    labs(title = paste("Scatter Plot of ", col, " vs Average Total Score"),
         x = col,
         y = "Average Total Score")

  print(p)  # Display the plot
  ggsave(filename = paste("Plot_of_", col, "_vs_Total.png"), plot = p, width = 10, height = 6, dpi = 300)
}
```

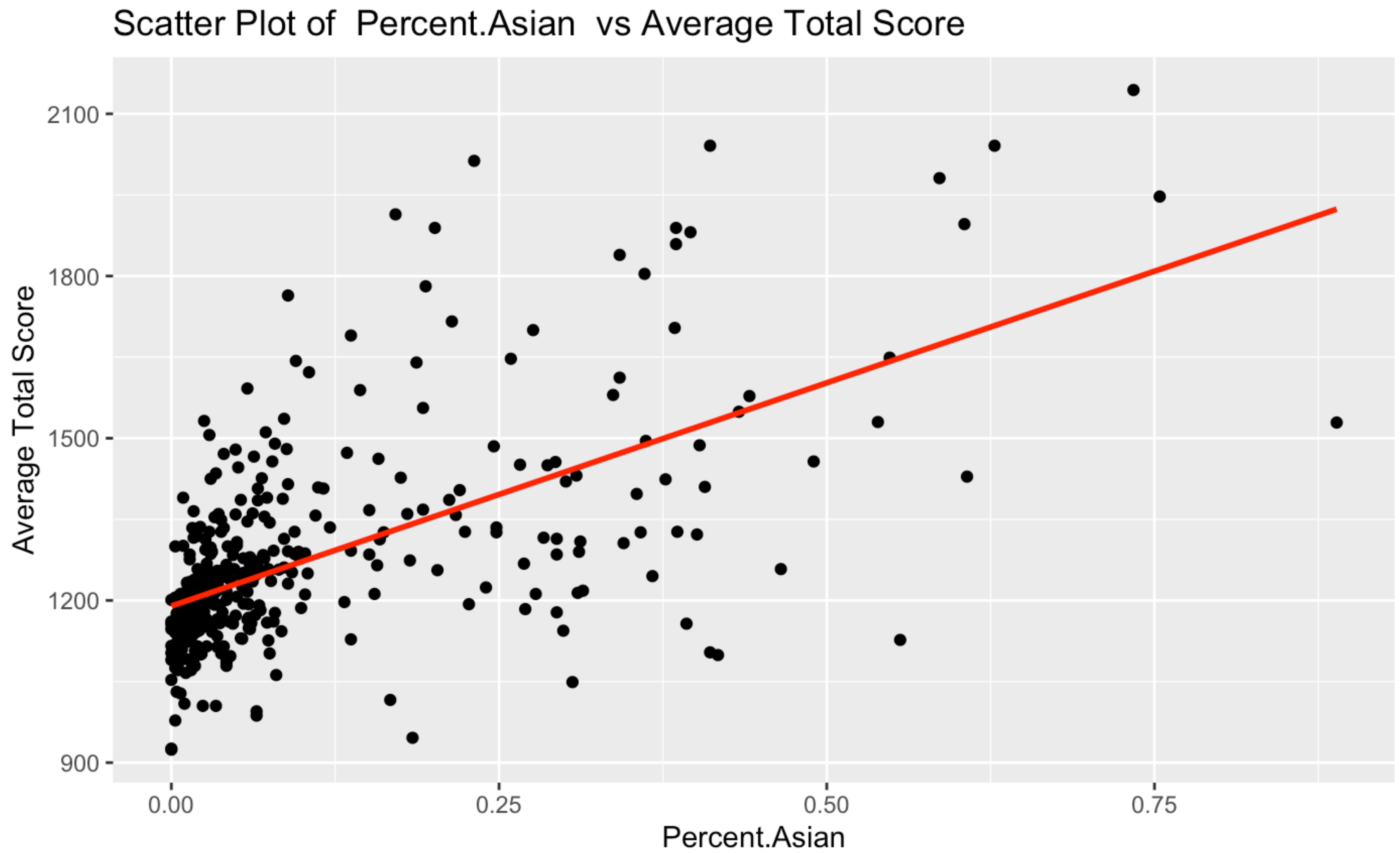
```
Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
Please use tidy evaluation idioms with `aes()`.
See also `vignette("ggplot2-in-packages")` for more information.
```

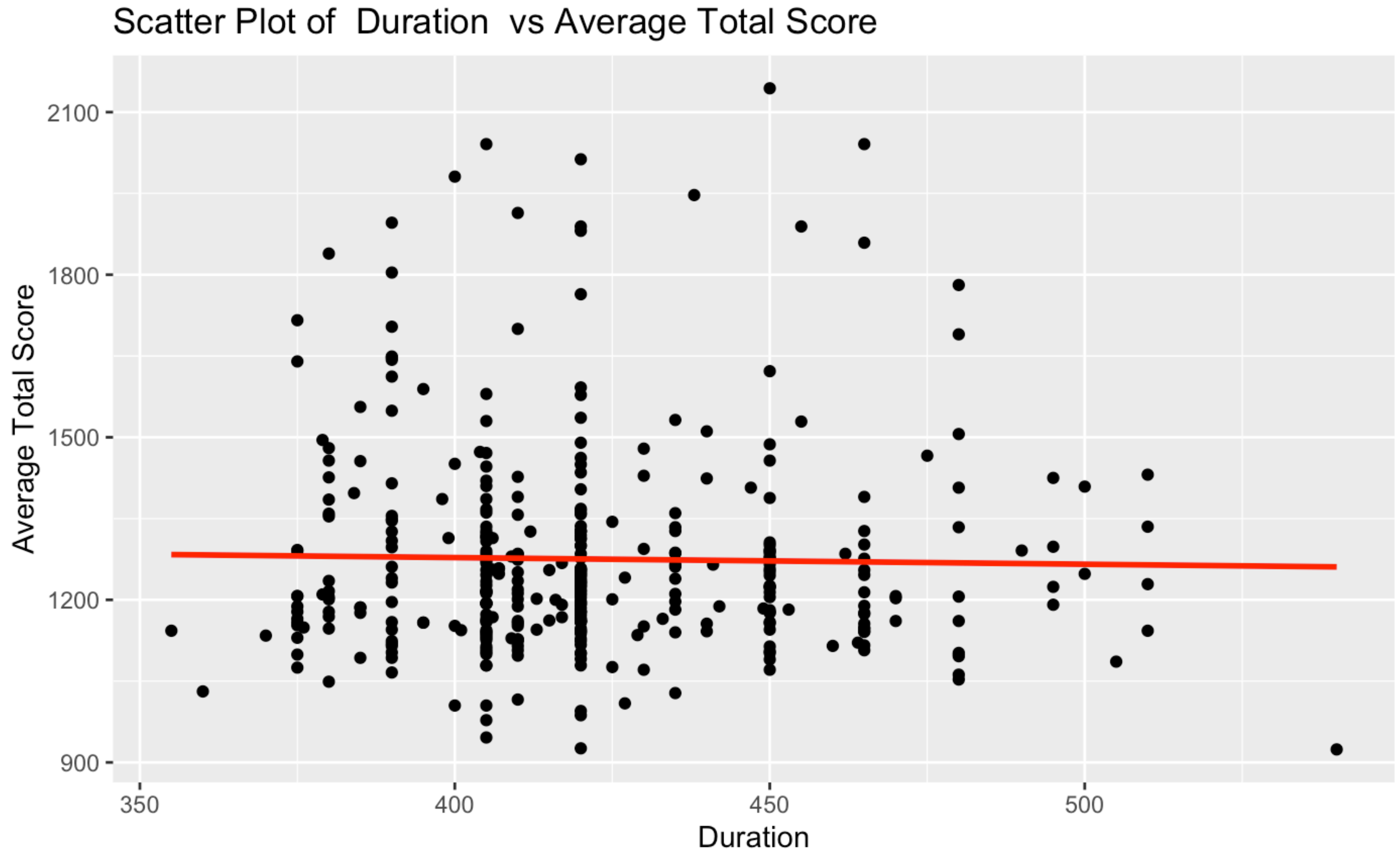

Scatter Plot of Student.Enrollment vs Average Total Score
Average Total Score
2100
1800
1500
1200
900
0
2000
4000
Student.Enrollment


Scatter Plot of Percent.White vs Average Total Score
Average Total Score
2100
1800
1500
1200
900
0.0
0.2
0.4
0.6
0.8
Percent.White


Scatter Plot of Percent.Black vs Average Total Score
Average Total Score
2100
1800
1500
1200
900
0.00
0.25
0.50
0.75
Percent.Black


Scatter Plot of Percent.Hispanic vs Average Total Score
Average Total Score
2100
1800
1500
1200
900
0.00
0.25
0.50
0.75
1.00
Percent.Hispanic





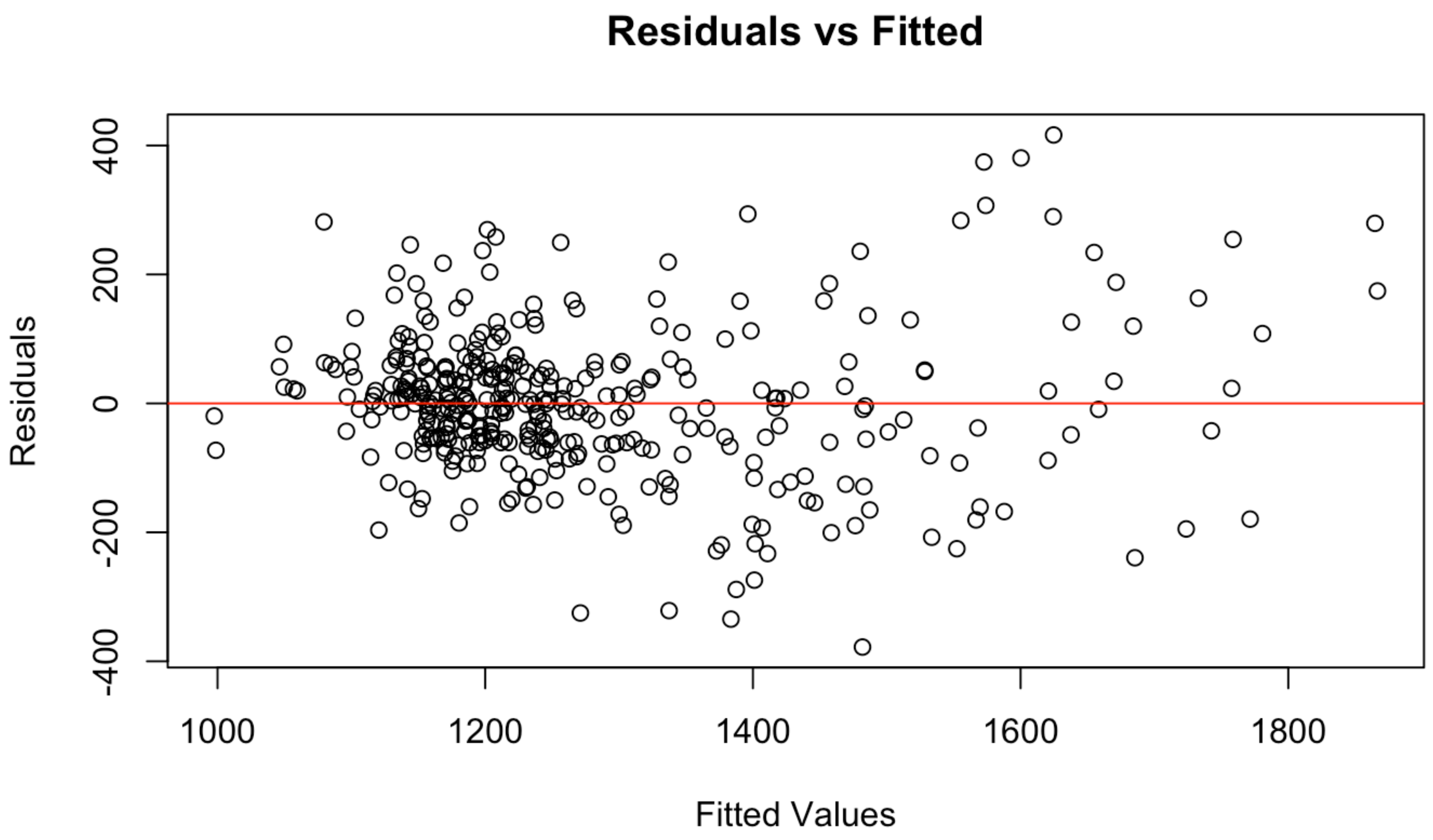
Hide

```
#Ordinary least squares model
df_OLS <- as.data.frame(df)
df_OLS = subset(df_OLS, select = -c(V1))
model <- lm(y ~ ., data = df_OLS)
summary(model)
```

```
Call:
lm(formula = y ~ ., data = df_OLS)

Residuals:
    Min      1Q  Median      3Q     Max 
-377.87  -60.79   -0.58   56.23  416.30 

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)    
(Intercept)            1275.348      6.028 211.579  < 2e-16 ***
Student.Enrollment       16.893      7.285   2.319 0.020952 *  
Percent.White          -155.682     53.550  -2.907 0.003870 ** 
Percent.Black          -451.347     98.253  -4.594 6.01e-06 ***
Percent.Hispanic       -475.602     92.950  -5.117 5.05e-07 ***
Percent.Asian          -186.500     56.276  -3.314 0.001012 ** 
Duration                 11.485      6.133   1.873 0.061925 .  
BoroughBronx            -11.768      7.857  -1.498 0.135063    
BoroughBrooklyn         -46.622      8.764  -5.320 1.82e-07 ***
BoroughQueens           -38.673      7.795  -4.961 1.08e-06 ***
`BoroughStaten Island`  -27.821      7.343  -3.789 0.000177 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 116.6 on 363 degrees of freedom
Multiple R-squared:  0.6517,	Adjusted R-squared:  0.6421 
F-statistic: 67.93 on 10 and 363 DF,  p-value: < 2.2e-16
```

Hide

```
predicted_values <- predict(model)
residuals <- residuals(model)

plot(predicted_values, residuals, main="Residuals vs Fitted", xlab="Fitted Values", ylab="Residuals")
abline(h = 0, col = "red")
```



Hide

```
#### Bayesian estimation via MCMC
n<-length(y)
X <- df
p<-dim(X)[2]

fit.ls<-lm(y~-1+ X)
beta.0<-rep(0,p) ; Sigma.0<-diag(rep(100,p),p)
nu.0<-1 ; sigma2.0<- 15^2

beta.0<-fit.ls$coef
nu.0<-1  ; sigma2.0<-sum(fit.ls$res^2)/(n-p)
Sigma.0<- solve(t(X)%*%X)*sigma2.0*n


S<-5000

rmvnorm<-function(n,mu,Sigma)
{ # samples from the multivariate normal distribution
  E<-matrix(rnorm(n*length(mu)),n,length(mu))
  t(  t(E%*%chol(Sigma)) +c(mu))
}

## some convenient quantites
n<-length(y)
p<-length(beta.0)
iSigma.0<-solve(Sigma.0)
XtX<-t(X)%*%X

## store mcmc samples in these objects
beta.post<-matrix(nrow=S,ncol=p)
sigma2.post<-rep(NA,S)

## starting value
set.seed(1)
sigma2<- var( residuals(lm(y~0+X)) )

## MCMC algorithm
for( scan in 1:S) {

#update beta
V.beta<- solve(  iSigma.0 + XtX/sigma2 )
E.beta<- V.beta%*%( iSigma.0%*%beta.0 + t(X)%*%y/sigma2 )
beta<-t(rmvnorm(1, E.beta,V.beta) )

#update sigma2
nu.n<- nu.0+n
ss.n<-nu.0*sigma2.0 + sum(  (y-X%*%beta)^2 )
sigma2<-1/rgamma(1,nu.n/2, ss.n/2)

#save results of this scan
beta.post[scan,]<-beta
sigma2.post[scan]<-sigma2
                        }

mean <- round( apply(beta.post,2,mean), 3)

quantiles <- apply(beta.post, 2, function(x) quantile(x, probs = c(0.025, 0.975)))


colnames(quantiles) <- c("Intercept", "Enroll", "P.white", "P.Black", "P.Hisp", "P.Asian", "Duration","BoroughBro
nx","BoroughBrooklyn","BoroughQueens","BoroughStatenIsland")


quantiles_df <- as.data.frame(t(quantiles))
quantiles_df <- cbind(quantiles_df, mean)
print(quantiles_df)
```

	2.5% <dbl>	97.5% <dbl>	mean <dbl>
Intercept	1263.3704654	1287.510408	1275.468
Enroll	2.4597970	31.009119	16.626
P.white	-258.1132511	-53.222830	-155.381
P.Black	-641.3748629	-263.104522	-450.917
P.Hisp	-653.8033025	-297.095517	-475.002
P.Asian	-294.5209321	-78.065471	-186.068
Duration	-0.3670514	23.970188	11.527
BoroughBronx	-27.3237417	3.421133	-11.685
BoroughBrooklyn	-63.4871561	-29.511997	-46.471

	2.5%	97.5%	mean
	<dbl>	<dbl>	<dbl>
BoroughQueens	-54.2241542	-23.749506	-38.609

1-10 of 11 rows Previous 1 2 Next

Hide

```
NA
```

Hide

```


#model selection (from canvas)
X <- df
X <- as.matrix(X)
p<-dim(X)[2]
S<-10000


## Don't run it again if you've already run it
runmcmc<-!any(system("ls",intern=TRUE)=="SAT_NYC.RData")
if(!runmcmc){ load("SAT_NYC.RData") }

if(runmcmc){

BETA<-Z<-matrix(NA,S,p)
z<-rep(1,dim(X)[2] )
lpy.c<-lpy.X(y,X[,z==1,drop=FALSE])
for(s in 1:S)
{
  if (s %% 1000 == 0) {
    print(s)
  }

  for(j in sample(1:p))
  {
    zp<-z ; zp[j]<-1-zp[j]
    lpy.p<-lpy.X(y,X[,zp==1,drop=FALSE])
    r<- (lpy.p - lpy.c)*(-1)^(zp[j]==0)
    z[j]<-rbinom(1,1,1/(1+exp(-r)))
    if(z[j]==zp[j]) {lpy.c<-lpy.p}
  }

  beta<-z
  if(sum(z)>0){beta[z==1]<-lm.gprior(y,X[,z==1,drop=FALSE],S=1)$beta }
  Z[s,]<-z
  BETA[s,]<-beta

}
save(BETA,Z,file="SAT_NYC.RData")
}
```

Hide

```
#print model selection results
inclusion_probabilities <- colMeans(Z)

coeff_names <- c("Intercept", "Enroll", "P.white", "P.Black", "P.Hisp", "P.Asian",
                 "Duration", "BoroughBronx", "BoroughBrooklyn", "BoroughQueens",
                 "BoroughStatenIsland")


inclusion_table <- rbind(coeff_names, inclusion_probabilities)
inclusion_df <- as.data.frame(t(inclusion_table))
names(inclusion_df) <- c("Variable", "Inclusion Probability")

print(inclusion_df)
```

Variable	Inclusion Probability
<chr>	<chr>
Intercept	1
Enroll	0.1571
P.white	0.205
P.Black	0.999
P.Hisp	0.9999
P.Asian	0.2634
Duration	0.1647

Variable <chr>	Inclusion Probability <chr>
BoroughBronx	0.1047
BoroughBrooklyn	0.9982
BoroughQueens	0.9949

1-10 of 11 rows Previous **1** 2 Next

Hide

```
NA
NA
```

Hide

```
#Model with removed predicitors
df_selected <- df_OLS[, -which(names(df_OLS) %in% c("Student.Enrollment","Percent.White","Percent.Asian","Borough
Bronx","Duration"))]

model_selected <- lm(y ~ ., data = df_selected)
summary(model_selected)
```

```

Call:
lm(formula = y ~ ., data = df_selected)

Residuals:
    Min      1Q  Median      3Q     Max 
-444.41  -61.32   -4.48   56.90  348.63 

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)    
(Intercept)            1275.348      6.168 206.760  < 2e-16 ***
Percent.Black          -153.646      8.309 -18.491  < 2e-16 ***
Percent.Hispanic       -193.169      8.608 -22.440  < 2e-16 ***
BoroughBrooklyn         -42.918      7.830  -5.481 7.85e-08 ***
BoroughQueens           -36.545      7.180  -5.090 5.72e-07 ***
`BoroughStaten Island`  -23.462      6.615  -3.547  0.00044 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 119.3 on 368 degrees of freedom
Multiple R-squared:  0.6303,	Adjusted R-squared:  0.6253 
F-statistic: 125.5 on 5 and 368 DF,  p-value: < 2.2e-16
```

Hide

```
#log transform y, full model OLS (to try to manage heteroscedasticity)
log_model <- lm(log(y) ~ ., data = df_OLS)
summary(log_model)
```

```

Call:
lm(formula = log(y) ~ ., data = df_OLS)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.277428 -0.047558  0.000277  0.047036  0.240348 

Coefficients:
                        Estimate Std. Error  t value Pr(>|t|)    
(Intercept)             7.140750   0.004383 1629.160  < 2e-16 ***
Student.Enrollment      0.015447   0.005297    2.916 0.003765 ** 
Percent.White          -0.114263   0.038939   -2.934 0.003554 ** 
Percent.Black          -0.325368   0.071445   -4.554 7.19e-06 ***
Percent.Hispanic       -0.344474   0.067589   -5.097 5.58e-07 ***
Percent.Asian          -0.140918   0.040921   -3.444 0.000641 ***
Duration                0.007671   0.004460    1.720 0.086288 .  
BoroughBronx           -0.010827   0.005713   -1.895 0.058886 .  
BoroughBrooklyn        -0.035585   0.006372   -5.584 4.61e-08 ***
BoroughQueens          -0.025394   0.005668   -4.480 1.00e-05 ***
`BoroughStaten Island` -0.019682   0.005339   -3.686 0.000262 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.08476 on 363 degrees of freedom
Multiple R-squared:  0.6384,	Adjusted R-squared:  0.6284 
F-statistic: 64.08 on 10 and 363 DF,  p-value: < 2.2e-16
```

Hide

```
var_original <- var(model$residuals)
var_log <- var(log_model$residuals)

var_original
```

```
[1] 13224.52
```

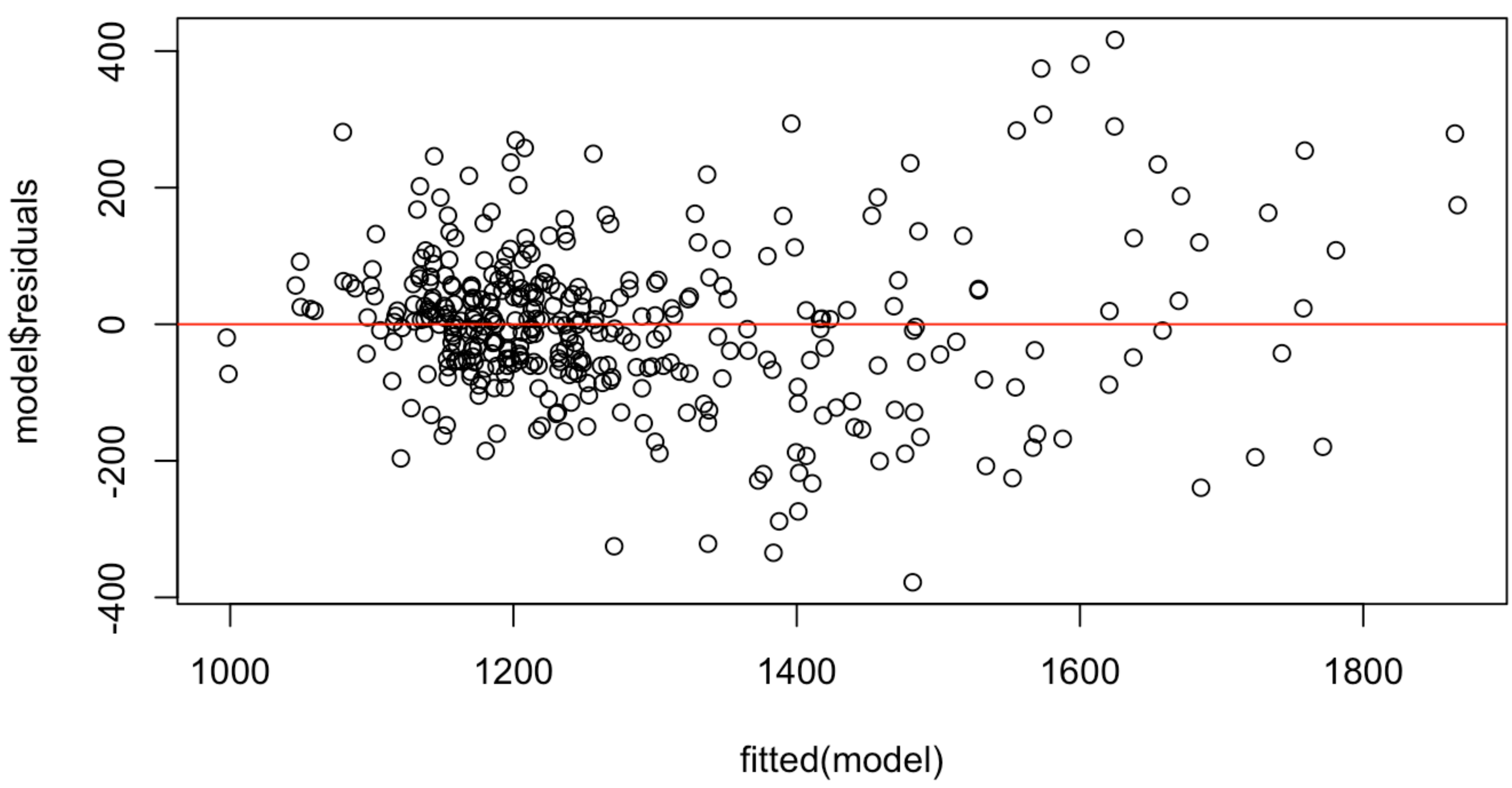
Hide

```
var_log
```

```
[1] 0.006992457
```

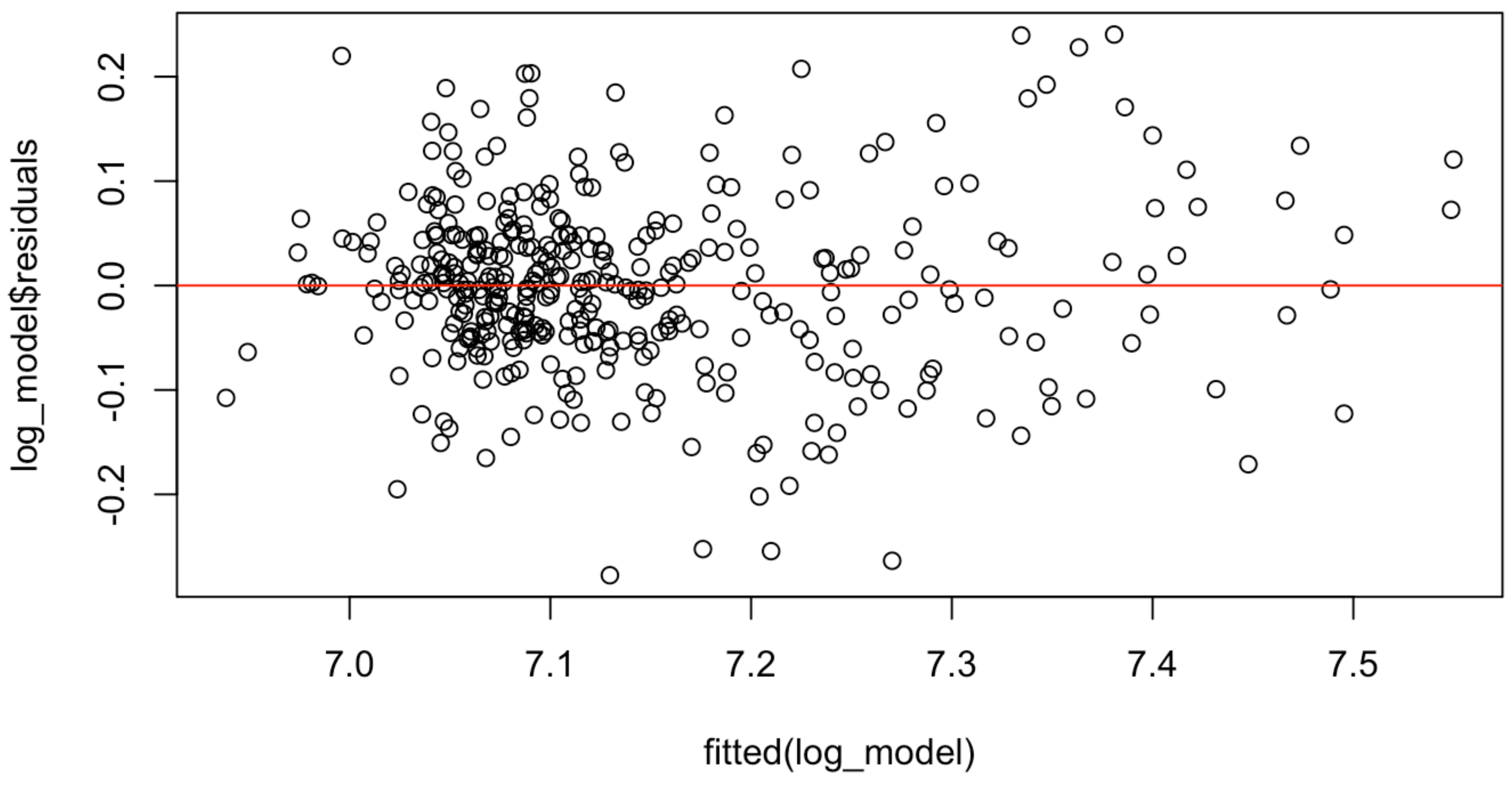
Hide

```
#residuals for the original model
plot(model$residuals ~ fitted(model))
abline(h = 0, col = "red")
```



Hide

```
#residuals for the log-transformed model
plot(log_model$residuals ~ fitted(log_model))
abline(h = 0, col = "red")
```



Hide

```
library(lmtest)
bptest(model)
```

```

	studentized Breusch-Pagan test

data:  model
BP = 118.15, df = 10, p-value < 2.2e-16
```

Hide

```
bptest(log_model)
```

```

	studentized Breusch-Pagan test

data:  log_model
BP = 69.445, df = 10, p-value = 5.673e-11
```

Hide

```
#Peform 80-20 test train split and compare full model OLS and with reduced model
n <- nrow(df_OLS)
train_indices <- sample(1:n, size = floor(0.8 * n))
train_data <- df_OLS[train_indices, ]
test_data <- df_OLS[-train_indices, ]
train_y <- y[train_indices]
test_y <- y[-train_indices]

#for reduced model
train_data_selected <- df_selected[train_indices, ]
test_data_selected <- df_selected[-train_indices, ]
```

Hide

```
#fit the models
model_train_full <- lm(train_y ~ ., data = train_data)
summary(model_train_full)
```

```
Call:
lm(formula = train_y ~ ., data = train_data)

Residuals:
    Min      1Q  Median      3Q     Max 
-355.03  -61.07   -0.26   51.92  441.37 

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)    
(Intercept)            1272.397      6.819 186.590  < 2e-16 ***
Student.Enrollment       16.799      8.107   2.072 0.039136 *  
Percent.White          -164.372     57.477  -2.860 0.004549 ** 
Percent.Black          -461.144    105.054  -4.390 1.60e-05 ***
Percent.Hispanic       -483.687     99.473  -4.862 1.91e-06 ***
Percent.Asian          -201.043     60.102  -3.345 0.000932 ***
Duration                  6.543      6.933   0.944 0.346139    
BoroughBronx            -13.117      8.967  -1.463 0.144619    
BoroughBrooklyn         -45.241      9.995  -4.526 8.79e-06 ***
BoroughQueens           -38.572      8.905  -4.331 2.05e-05 ***
`BoroughStaten Island`  -26.371      8.249  -3.197 0.001543 ** 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 117.8 on 288 degrees of freedom
Multiple R-squared:  0.6074,	Adjusted R-squared:  0.5937 
F-statistic: 44.55 on 10 and 288 DF,  p-value: < 2.2e-16
```

Hide

```
model_train_selected <- lm(train_y ~ ., data = train_data_selected)
summary(model_train_selected)
```

```
Call:
lm(formula = train_y ~ ., data = train_data_selected)

Residuals:
    Min      1Q  Median      3Q     Max 
-424.27  -63.13   -3.58   50.88  367.53 

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)    
(Intercept)            1272.477      6.978 182.347  < 2e-16 ***
Percent.Black          -142.873      9.342 -15.294  < 2e-16 ***
Percent.Hispanic       -181.810     10.131 -17.947  < 2e-16 ***
BoroughBrooklyn         -40.388      8.929  -4.523 8.85e-06 ***
BoroughQueens           -36.270      8.081  -4.488 1.03e-05 ***
`BoroughStaten Island`  -20.428      7.223  -2.828    0.005 ** 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 120.6 on 293 degrees of freedom
Multiple R-squared:  0.5812,	Adjusted R-squared:  0.574 
F-statistic: 81.32 on 5 and 293 DF,  p-value: < 2.2e-16
```

Hide

```
# Predictions
predicted_test_y_full <- predict(model_train_full, newdata = test_data)
rmse_full <- sqrt(mean((predicted_test_y_full - test_y)^2))
print(paste("RMSE for full model:", rmse_full))
```

```
[1] "RMSE for full model: 115.006203175707"
```

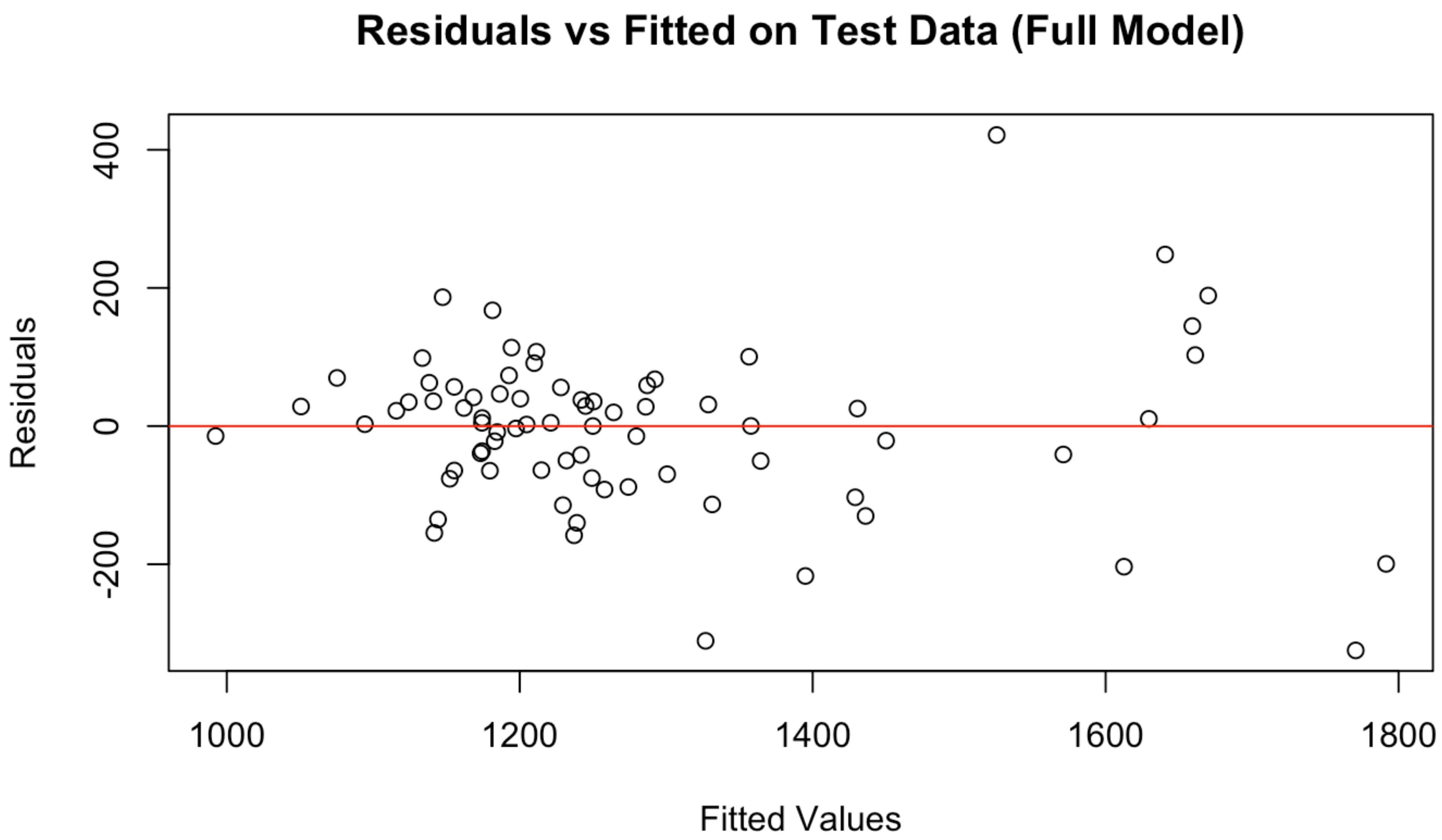
Hide

```
predicted_test_y_selected <- predict(model_train_selected, newdata = test_data_selected)
rmse_selected <- sqrt(mean((predicted_test_y_selected - test_y)^2))
print(paste("RMSE for selected model:", rmse_selected))
```

```
[1] "RMSE for selected model: 116.474814384356"
```

Hide

```
# Plot residuals
test_residuals_full <- test_y - predicted_test_y_full
plot(predicted_test_y_full, test_residuals_full, main="Residuals vs Fitted on Test Data (Full Model)", xlab="Fitt
ed Values", ylab="Residuals")
abline(h = 0, col = "red")
```



Hide

```
test_residuals_selected <- test_y - predicted_test_y_selected
plot(predicted_test_y_selected, test_residuals_selected, main="Residuals vs Fitted on Test Data (Selected Mode
l)", xlab="Fitted Values", ylab="Residuals")
abline(h = 0, col = "red")
```