CS-E-106: Data Modeling

Assignment 1

Instructor: Hakan Gogtas
Submitted by: *Saurabh Kulkarni*

Due Date: *09/23/2019*

Solution 1:

Regression Function: $Y_i = \beta_0 + \epsilon_i$

Thus, sum of squared residuals, $Q = \sum_{i=1}^{n}(Y_i - \beta_0 - \epsilon_i)^2$

Taking partial derivatives w.r.t. β_0

$\frac{\partial Q}{\partial \beta_0} = 2\sum_{i=1}^{n}(Y_i - \beta_0)(-1) =^{set} 0$

$\therefore \sum_{i=1}^{n} Y_i = n\beta_0$

$\therefore \beta_0 = \frac{\sum_{i=1}^{n} Y_i}{n}$

$\therefore \beta_0 = \bar{Y}$

Solution 2:

Install R packages:

```
# install.packages("faraway")
# install.packages("ggplot2")
# install.packages("corrgram")
```

Numerical Summary of the dataset:

```
library(faraway)
teengamb$sex <- as.factor(teengamb$sex)
summary(teengamb)
```

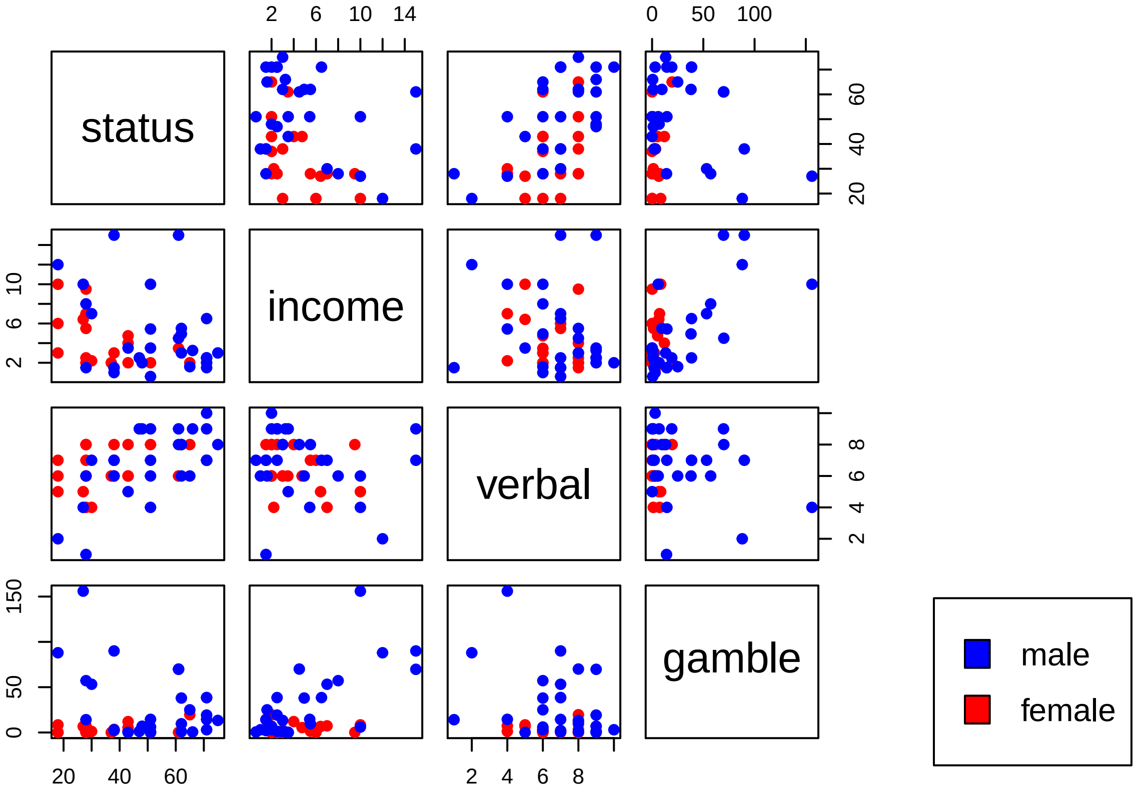
```
##  sex        status          income            verbal          gamble
##  0:28   Min.   :18.00   Min.   : 0.600   Min.   : 1.00   Min.   :  0.0
##  1:19   1st Qu.:28.00   1st Qu.: 2.000   1st Qu.: 6.00   1st Qu.:  1.1
##         Median :43.00   Median : 3.250   Median : 7.00   Median :  6.0
##         Mean   :45.23   Mean   : 4.642   Mean   : 6.66   Mean   : 19.3
##         3rd Qu.:61.50   3rd Qu.: 6.210   3rd Qu.: 8.00   3rd Qu.: 19.4
##         Max.   :75.00   Max.   :15.000   Max.   :10.00   Max.   :156.0
```

```
help("teengamb")
```

Looking at the summary of the dataset, we can see that the numerical variables are on different scales. help() on "teengamb" dataset confirms it.

Scatterplot Matrix of Numerical Variables:

```
pairs(teengamb[,-1], pch=19, col=c('blue','red')[teengamb$sex], oma=c(3,3,3,15))
par(xpd = TRUE)
legend("bottomright", fill = c('blue', 'red'), legend = c("male", "female"))
```



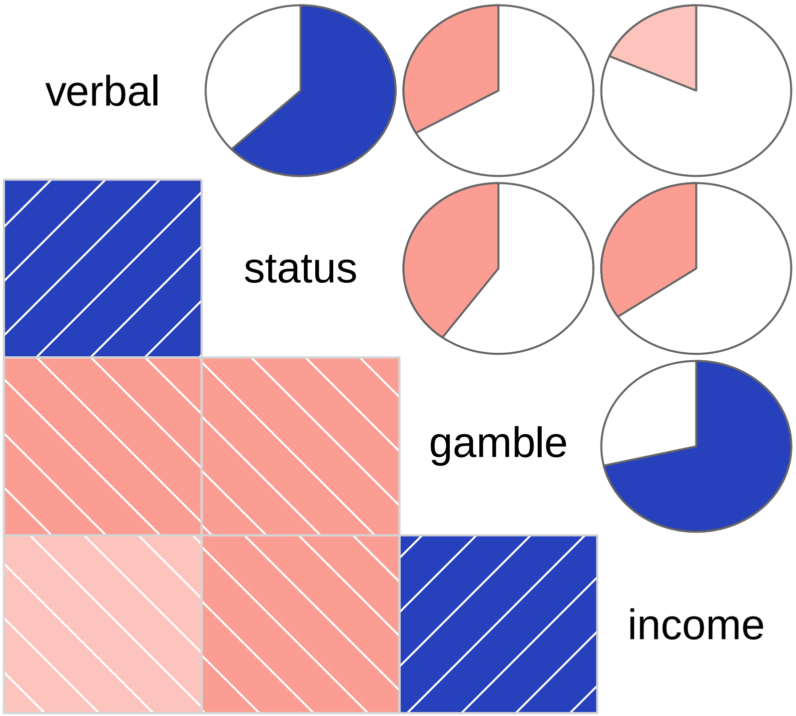
We can see that the gambling expediture increases with the increase in income for males but not a very steep increase for females. We can also see a correlation between the variables `verbal` and `status`.

Correlogram for Males:

```r
library(corrgram)
```

```
## Registered S3 method overwritten by 'seriation':
##   method         from
##   reorder.hclust gclus
```

```r
corrgram(teengamb[which(teengamb$sex==0),], order=TRUE, lower.panel=panel.shade,
  upper.panel=panel.pie, text.panel=panel.txt, oma=c(3,3,3,15),
  main="Correlogram for Males")
```



Here, we subset the dataset for males and plot a correlogram to hone in a little bit. The correlogram confirms the correlations we suspected above - we can see strong positive correlation between `gamble` and `income` as well as between `verbal` and `status` for males.

Solution 3:

(a)

`reg_loop`: Here we write a function to loop over all the desired independent variables in the problem statement, regress the given dependent variable (`Number.of.active.physicians`) on each of them and then return a list of resulting linear models.

```
reg_loop <- function(df, x_cols, y_str) {
  lm_regs = list({})
  for(i in 1:length(x_cols)){
    x_str = x_cols[i]
    formula = as.formula(paste(y_str, x_str))
    lm_regs[[i]] = lm(formula, data=df)
    print(paste("Linear Regression Summary:", x_cols[i]))
    print(summary(lm_regs[[i]]))
  }
  lm_regs
}
```

Fitting the three models:

```
cdi = read.csv("cdi.csv")
cdi = lapply(cdi, as.numeric)
cdi = data.frame(cdi)
x_cols = c("Total.population", "Number.of.hospital.beds", "Total.personal.income")
y_str = "Number.of.active.physicians ~"
lm_fits = reg_loop(df=cdi, x_cols=x_cols, y_str=y_str)
```

```
## [1] "Linear Regression Summary: Total.population"
```

```
##
## Call:
## lm(formula = formula, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1969.4  -209.2   -88.0    27.9  3928.7
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -1.106e+02  3.475e+01  -3.184  0.00156 **
## Total.population  2.795e-03  4.837e-05  57.793  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 610.1 on 438 degrees of freedom
## Multiple R-squared:  0.8841, Adjusted R-squared:  0.8838
## F-statistic:  3340 on 1 and 438 DF,  p-value: < 2.2e-16
##
## [1] "Linear Regression Summary: Number.of.hospital.beds"
##
## Call:
## lm(formula = formula, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3133.2  -216.8   -32.0    96.2  3611.1
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -95.93218   31.49396  -3.046  0.00246 **
## Number.of.hospital.beds   0.74312    0.01161  63.995  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 556.9 on 438 degrees of freedom
## Multiple R-squared:  0.9034, Adjusted R-squared:  0.9032
## F-statistic:  4095 on 1 and 438 DF,  p-value: < 2.2e-16
##
## [1] "Linear Regression Summary: Total.personal.income"
##
## Call:
## lm(formula = formula, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1926.6  -194.5   -66.6    44.2  3819.0
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -48.39485   31.83333   -1.52    0.129
## Total.personal.income   0.13170    0.00211   62.41   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 569.7 on 438 degrees of freedom
## Multiple R-squared:  0.8989, Adjusted R-squared:  0.8987
## F-statistic:  3895 on 1 and 438 DF,  p-value: < 2.2e-16
```

Thus, we have the *three regression equations* as:

1. Number.of.active.physicians = -110.63 + 0.0028*Total.population
2. Number.of.active.physicians = -95.93 + 0.74*Number.of.hospital.beds
3. Number.of.active.physicians = -48.39 + 0.13*Total.personal.income

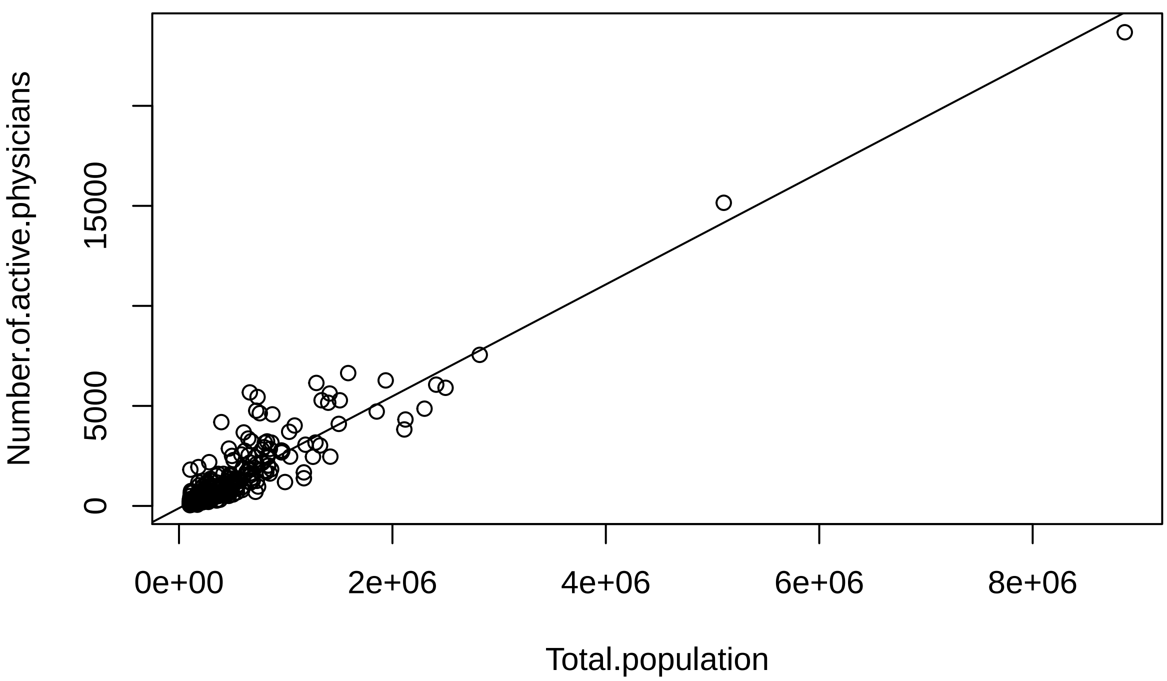
(b)

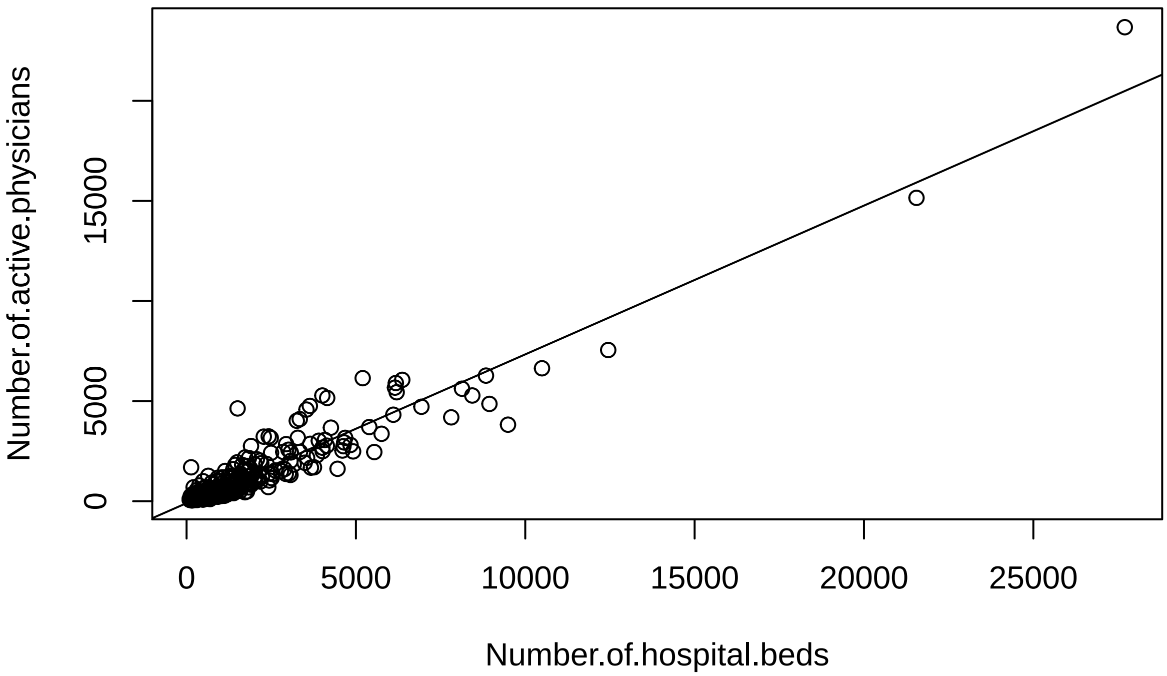
`plot_reg_func`: Creates three distinct plots for each model created above.

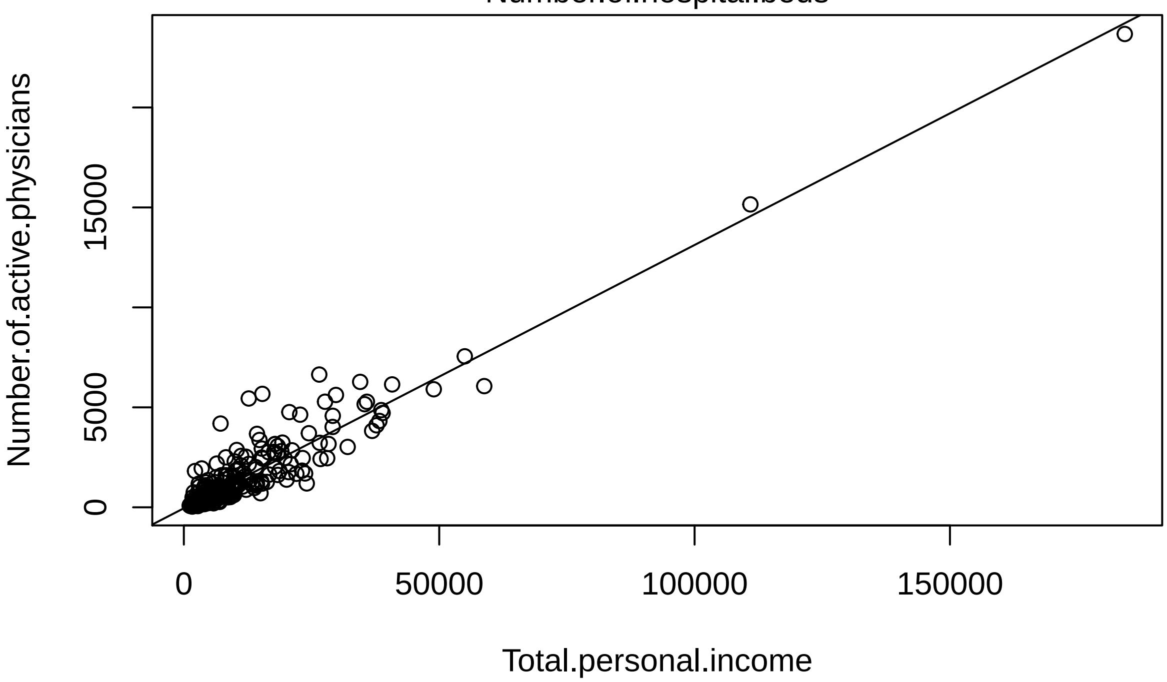
```
plot_reg_func <- function(fits, df, x_cols){
  for(i in 1:length(x_cols)){
    plot(df[[x_cols[i]]], df[["Number.of.active.physicians"]], xlab=x_cols[i], ylab="Number.of.active.pl
    abline(fits[[i]])
    }
}
```

Plots for Fitted Regression Lines:

```
plot_reg_func(fits=lm_fits, df=cdi, x_cols=x_cols)
```







Based on the plots, we can say that the simple linear regression models provide a good fit for each of the three independent variable. We can also confirm this by looking at the R-squared values in the summaries printed above or MSE's in part (c).

(c)

`mse_loop`: This function goes also over the three predictor variables and calculates the MSE for each of them using the respective model from the list we provide to it.

```
mse_loop <- function(fits, df, x_cols){
  for(i in 1:length(x_cols)){
    yHat = predict(fits[[i]], df[x_cols])
    resids = (df$`Number.of.active.physicians`-yHat)
```

```
    SSE = (sum(resids^2))
    df_resids = (nrow(df)-2)
    MSE = (SSE/df_resids)
    print(paste("MSE for",x_cols[i]))
    print(MSE)
  }
}
```

Calculated MSEs:

```
mse_loop(fits=lm_fits, df=cdi, x_cols=x_cols)
```

```
## [1] "MSE for Total.population"
## [1] 372203.5
## [1] "MSE for Number.of.hospital.beds"
## [1] 310191.9
## [1] "MSE for Total.personal.income"
## [1] 324539.4
```

Thus we can see that `Number.of.hospital.beds` gives the least MSE and thus has least variability around the fitted regression line.

Solution 4:

Here we repeat the whole solution to problem 3 but our models are now trained on the 70% training data. We create this by randomly sampling 70% of the data as shown below and call that as our `train_cdi` and the remaining we call `test_cdi`.

```
train_ind = sample(1:nrow(cdi), 0.7 * nrow(cdi))
test_ind = setdiff(1:nrow(cdi), train_ind)
train_cdi = cdi[train_ind,]
test_cdi = cdi[test_ind,]
```

(a)

Fitting the three models:

```
lm_fits_tr = reg_loop(df=train_cdi, x_cols=x_cols, y_str=y_str)
```

```
## [1] "Linear Regression Summary: Total.population"
##
## Call:
## lm(formula = formula, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1937.7  -214.0  -113.3    15.6  3913.4
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -7.364e+01  4.204e+01  -1.751   0.0809 .
## Total.population  2.763e-03  5.746e-05  48.087   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 628.7 on 306 degrees of freedom
## Multiple R-squared:  0.8831, Adjusted R-squared:  0.8828
## F-statistic:  2312 on 1 and 306 DF,  p-value: < 2.2e-16
```

```
##
## [1] "Linear Regression Summary: Number.of.hospital.beds"
##
## Call:
## lm(formula = formula, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3537.1  -214.4   -28.6   110.7  3571.7
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -125.3785    38.9880  -3.216  0.00144 **
## Number.of.hospital.beds    0.7888     0.0149  52.947  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 576.9 on 306 degrees of freedom
## Multiple R-squared:  0.9016, Adjusted R-squared:  0.9013
## F-statistic:  2803 on 1 and 306 DF,  p-value: < 2.2e-16
##
## [1] "Linear Regression Summary: Total.personal.income"
##
## Call:
## lm(formula = formula, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1919.4  -203.0   -81.9    27.6  3811.8
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -25.105994  37.408823  -0.671    0.503
## Total.personal.income   0.130432   0.002422  53.855   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 568.1 on 306 degrees of freedom
## Multiple R-squared:  0.9046, Adjusted R-squared:  0.9043
## F-statistic:  2900 on 1 and 306 DF,  p-value: < 2.2e-16
```

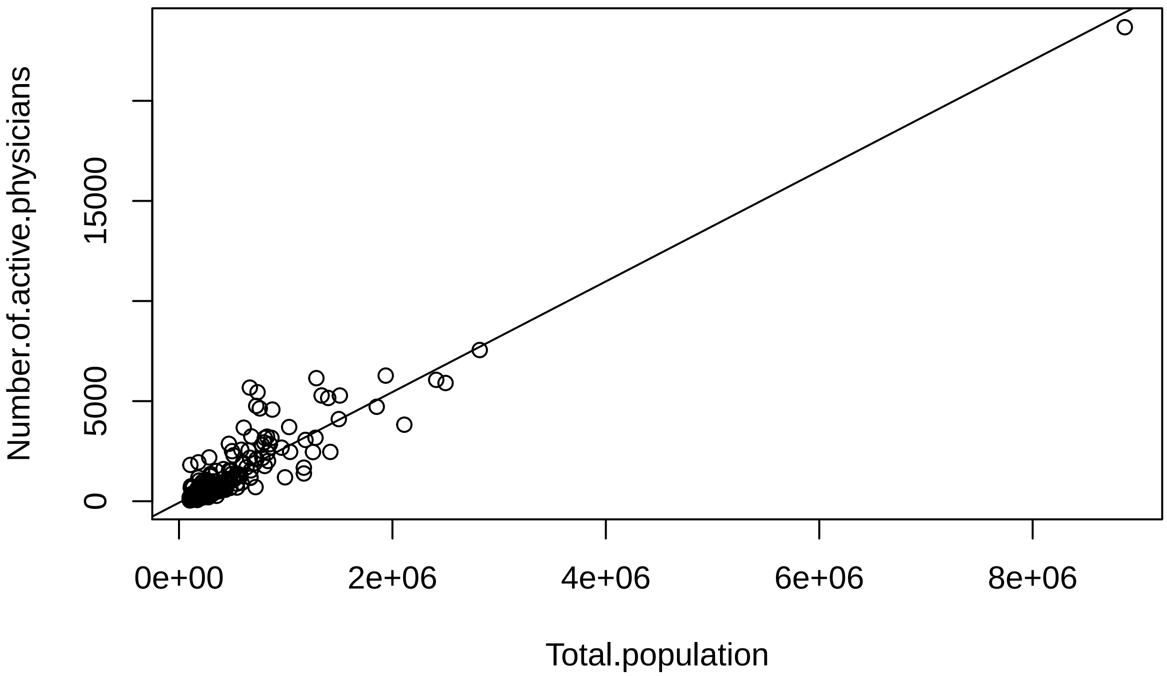
Thus, we have the *new three regression equations* on training set as:

1. Number.of.active.physicians = -114.4 + 0.0027*Total.population
2. Number.of.active.physicians = -119.98 + 0.76*Number.of.hospital.beds
3. Number.of.active.physicians = -53.4 + 0.13*Total.personal.income

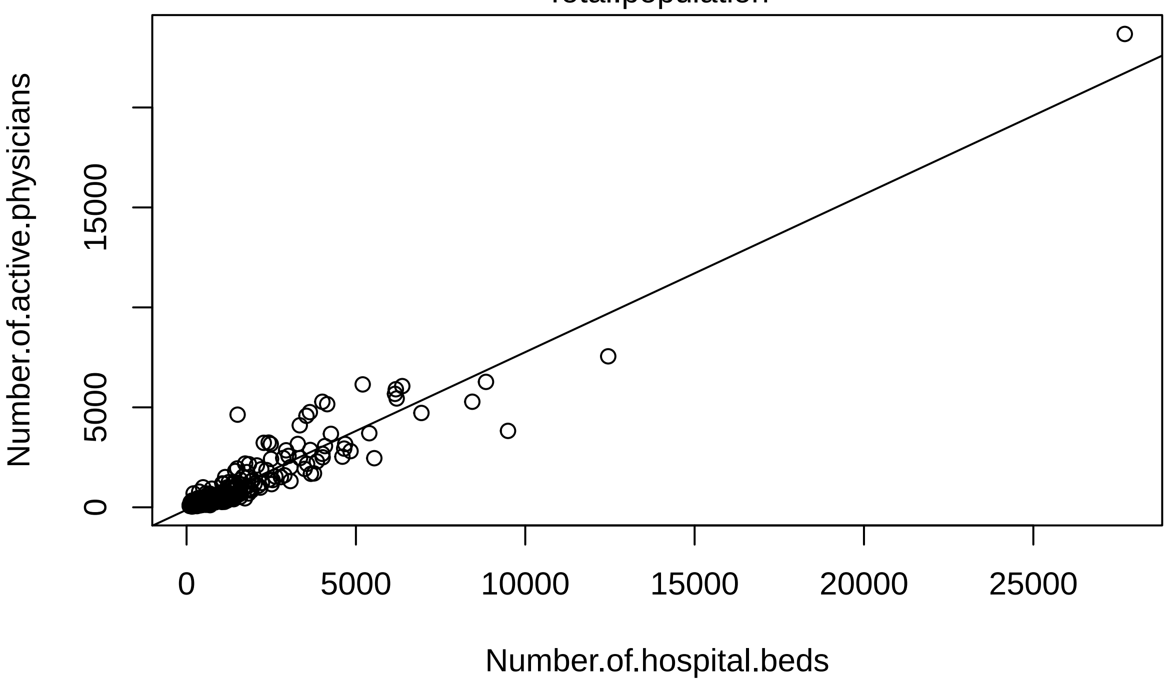
(b)

Plots for Fitted Regression Lines:

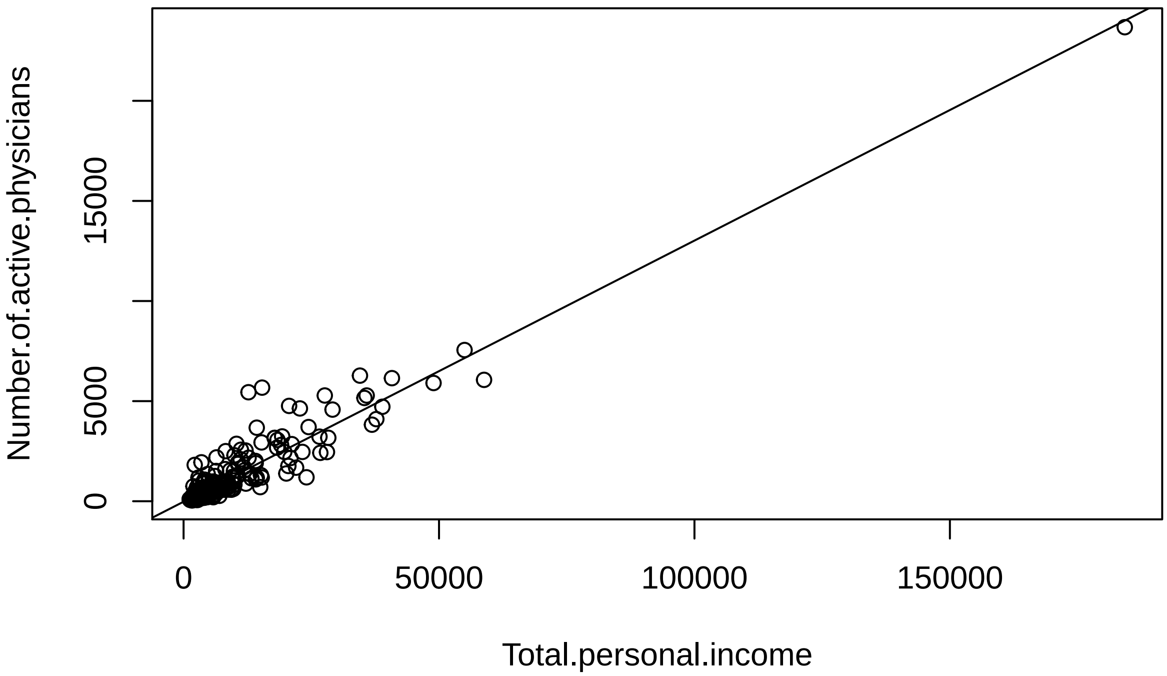
```r
plot_reg_func(fits=lm_fits_tr, df=train_cdi, x_cols=x_cols)
```



Number.of.active.physicians
0
5000
15000
0e+00
2e+06
4e+06
6e+06
8e+06
Total.population



Number.of.active.physicians
0
5000
15000
0
5000
10000
15000
20000
25000
Number.of.hospital.beds



The plots more or less show similar fits as in Solution 3.

(c)

Calculated MSEs:

```r
mse_loop(fits=lm_fits, df=train_cdi, x_cols=x_cols)
```

```
## [1] "MSE for Total.population"
## [1] 396299.4
## [1] "MSE for Number.of.hospital.beds"
## [1] 344289.6
## [1] "MSE for Total.personal.income"
## [1] 323258.1
```

Thus, with 70% training sample we get `Total.personal.income` as the variable that gives us the smallest variability and thus the best fit.

Solution 5:

(a)

```r
lm_gamb = lm(gamble~income, data=teengamb)
summary(lm_gamb)
```

```
##
## Call:
## lm(formula = gamble ~ income, data = teengamb)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -46.020 -11.874  -3.757  11.934 107.120
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -6.325      6.030  -1.049      0.3
```

```
## income         5.520      1.036   5.330 3.05e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 24.95 on 45 degrees of freedom
## Multiple R-squared:  0.387,  Adjusted R-squared:  0.3734
## F-statistic: 28.41 on 1 and 45 DF,  p-value: 3.045e-06
```

The regression equation is: gamble = -6.325 + 5.52*income

```r
yHat = predict(lm_gamb, teengamb["income"])
resids = (teengamb$gamble-yHat)
SSE = (sum(resids^2))
df_resids = (nrow(teengamb)-2)
MSE = (SSE/df_resids)
print(paste("Mean of the residuals:", mean(resids)))
```

```
## [1] "Mean of the residuals: -2.51361431669039e-15"
```

```r
print(paste("Median of the residuals:", median(resids)))
```

```
## [1] "Median of the residuals: -3.7573821062283"
```

```r
print(paste("MSE:", MSE))
```

```
## [1] "MSE: 622.41305773688"
```

Note how close the mean of the residuals is to zero. Ideally, it should be perfectly zero (that is one of the properties of least squares).

(b)

```r
print(paste("Case number for highest positive residual:", which(resids == max(resids))))
```

```
## [1] "Case number for highest positive residual: 24"
```

```r
print(resids[24])
```

```
##       24
## 107.1197
```

```r
print(teengamb[24,])
```

```
##    sex status income verbal gamble
## 24   0     27     10      4    156
```