Intro to Data Science Lab 8

```
# Enter your name here: Hrishikesh Telang
```

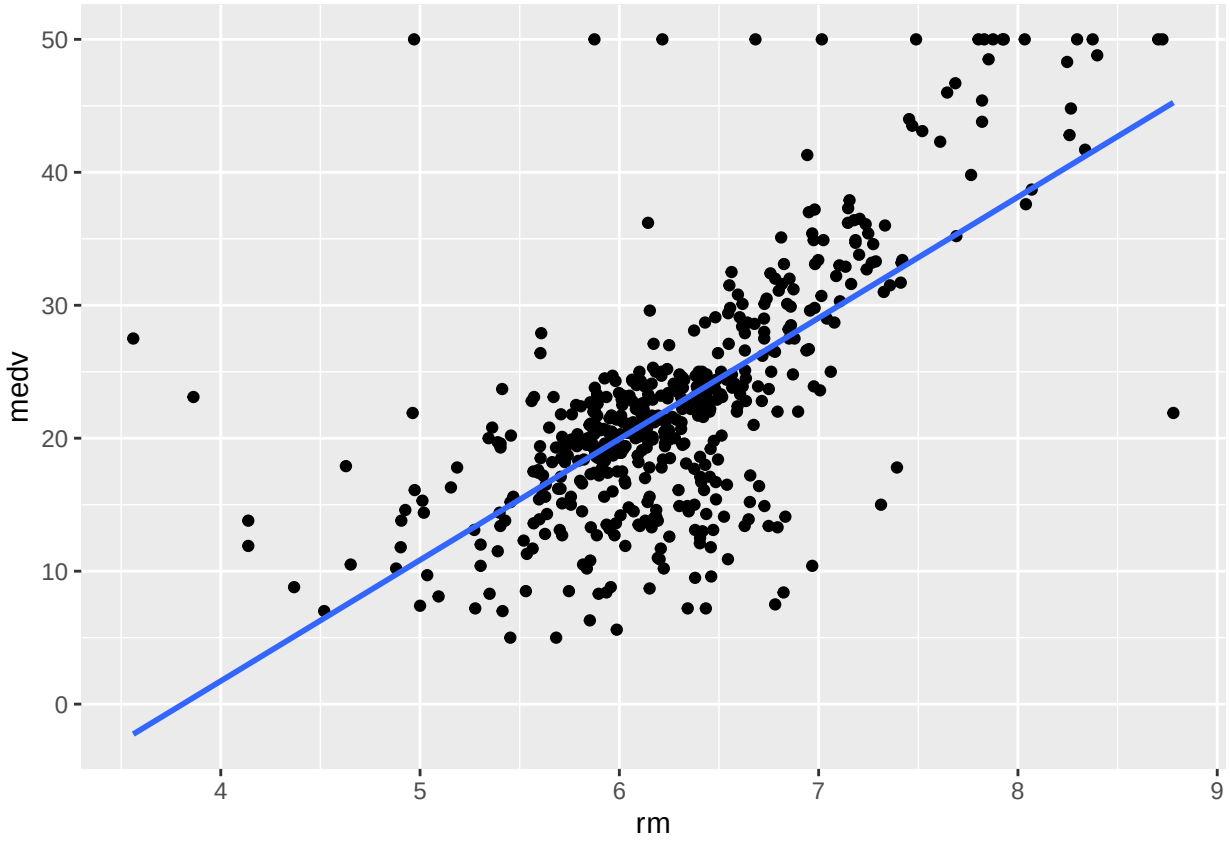

Attribution statement: (choose only one and delete the rest)

```
# 1. I did this homework by myself, with help from the book and the professor.
```

#Instructions: Linear modeling, also referred to as regression analysis or multiple regression, is a technique for fitting a line, plane, or higher order linear object to data. In their simplest form, linear models have one metric outcome variable and one or more predictor variables (any combination of metric values, ordered scales such as ratings, or dummy codes). Make sure to library the MASS and ggplot2 packages before running the following:

```
library(MASS) #Call library MASS
library(ggplot2) #Call library ggplot2
ggplot(data=Boston) + aes(x=rm, y=medv) + geom_point() + geom_smooth(method="lm", se=FALSE)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
#geom_smooth returns a linear model, ggplot plots the data of the Boston dataset,
#aes decides the x and y axes of the plane, and geom_point returns point distribution.
```

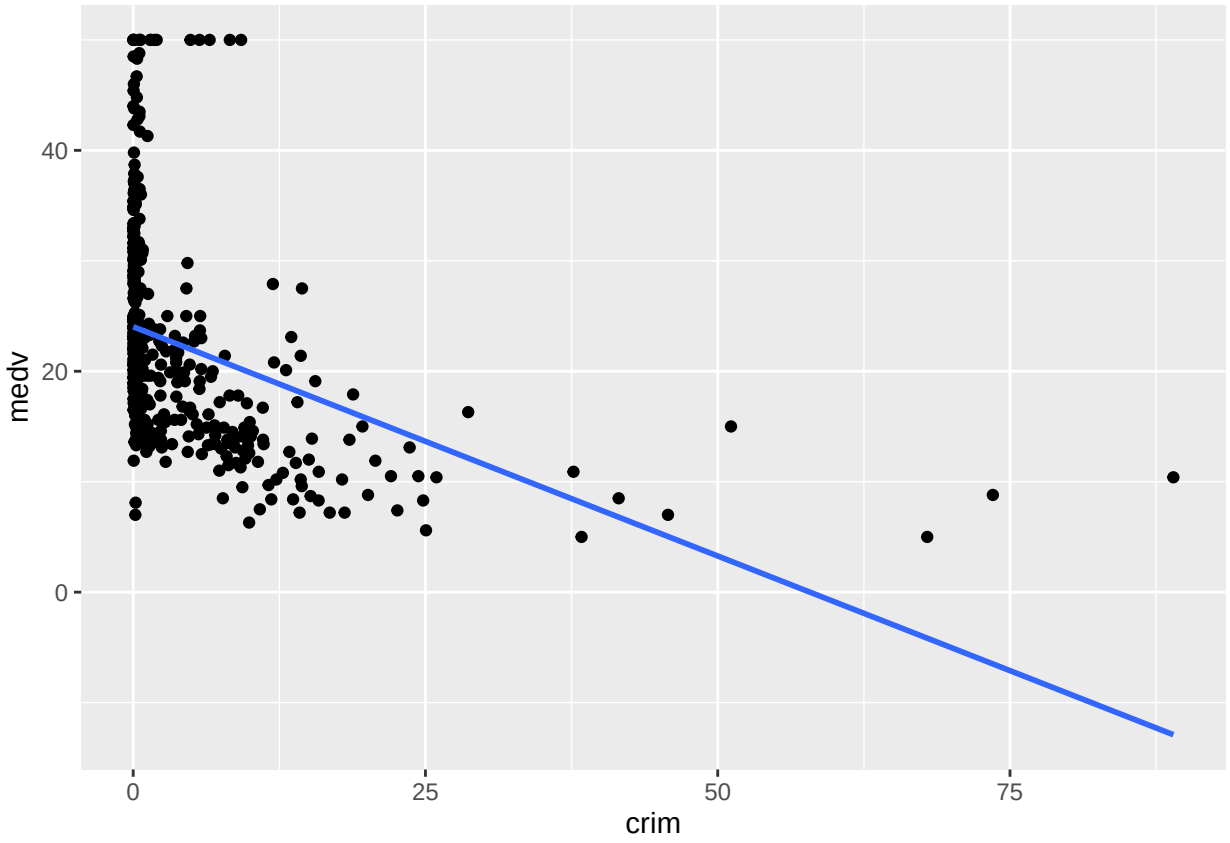
#1. Explore this dataset descrption by doing ?Boston at the command prompt

```
?Boston
```

#2. The graphic you just created fits a best line to a cloud of points. Copy and modify the code to produce a plot where "crim" is the x variable instead of "rm."

```
ggplot(data=Boston) + aes(x=crim, y=medv) + geom_point() + geom_smooth(method="lm", se=FALSE)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



#3. Produce a histogram and descriptive statistics for Boston$crim. Write a comment describing any anomalies or oddities.

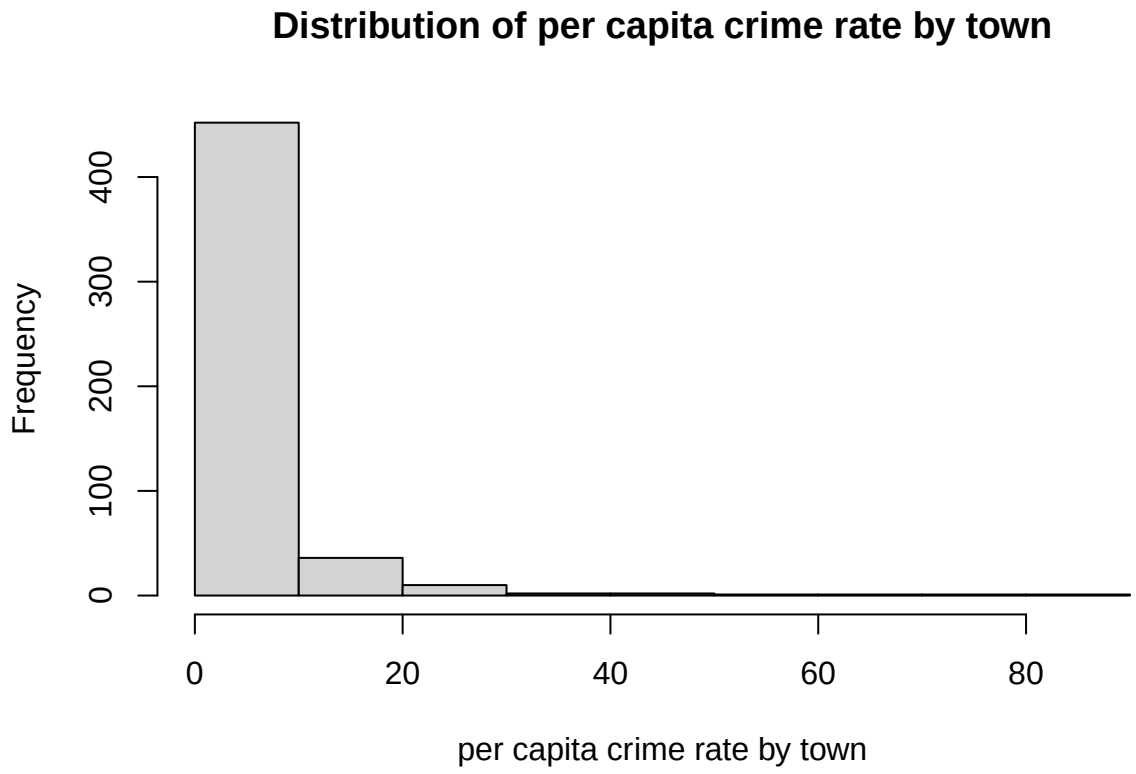
```
hist(Boston$crim,
     main='Distribution of per capita crime rate by town',
     xlab='per capita crime rate by town',
     simplify=TRUE)
```

```
## Warning in plot.window(xlim, ylim, "", ...): "simplify" is not a graphical
## parameter
```

```
## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## "simplify" is not a graphical parameter

## Warning in axis(1, ...): "simplify" is not a graphical parameter

## Warning in axis(2, ...): "simplify" is not a graphical parameter
```




```
#The following distribution does have anomalies. It is highly right skewed
#which totally deviates from normal distribution.
```

#4. Produce a linear model, using the lm( ) function where crim predicts medv. Remember that in R's formula language, the outcome variable comes first and is separated from the predictors by a tilde, like this: medv ~ crim.

```
lm(formula = medv ~ crim, data=Boston)
```

```
##
## Call:
## lm(formula = medv ~ crim, data = Boston)
##
## Coefficients:
## (Intercept)         crim
##     24.0331      -0.4152
```

#Try to get in the habit of storing the output object that is produced by lm and other analysis procedures. For example, I often use lmOut <- lm(. . .

```
sample <- lm(formula = medv ~ crim, data=Boston)
sample
```

```
##
## Call:
## lm(formula = medv ~ crim, data = Boston)
##
## Coefficients:
## (Intercept)         crim
##     24.0331      -0.4152
```

#5. Run a multiple regression where you use rm, crim, and dis (distance to Boston employment centers). You will use all three predictors in one model with this formula: medv ~ crim + rm + dis. Now run three separate models for each independent variable separate.

```
lmOut <- lm(formula=medv ~ crim + rm + dis, data=Boston)
summary(lmOut)
```

```
##
## Call:
## lm(formula = medv ~ crim + rm + dis, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -21.247  -2.930  -0.572   2.390  39.072
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -29.45838    2.60010 -11.330  < 2e-16 ***
## crim         -0.25405    0.03532  -7.193 2.32e-12 ***
## rm            8.34257    0.40870  20.413  < 2e-16 ***
## dis           0.12627    0.14382   0.878     0.38
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.238 on 502 degrees of freedom
## Multiple R-squared:  0.5427, Adjusted R-squared:  0.5399
## F-statistic: 198.6 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
lmOut2 <- lm(formula=medv ~ crim, data=Boston)
summary(lmOut2)
```

```
##
## Call:
## lm(formula = medv ~ crim, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.957  -5.449  -2.007   2.512  29.800
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 24.03311    0.40914   58.74   <2e-16 ***
## crim        -0.41519    0.04389   -9.46   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.484 on 504 degrees of freedom
## Multiple R-squared:  0.1508, Adjusted R-squared:  0.1491
## F-statistic: 89.49 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
lmOut3 <- lm(formula=medv ~ rm, data=Boston)
summary(lmOut3)
```

```
##
## Call:
## lm(formula = medv ~ rm, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -23.346  -2.547   0.090   2.986  39.433
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -34.671      2.650  -13.08   <2e-16 ***
## rm             9.102      0.419   21.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.616 on 504 degrees of freedom
## Multiple R-squared:  0.4835, Adjusted R-squared:  0.4825
## F-statistic: 471.8 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
lmOut4 <- lm(formula=medv ~ dis, data=Boston)
summary(lmOut4)
```

```
##
## Call:
## lm(formula = medv ~ dis, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.016  -5.556  -1.865   2.288  30.377
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  18.3901     0.8174  22.499  < 2e-16 ***
## dis           1.0916     0.1884   5.795 1.21e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.914 on 504 degrees of freedom
```

```
## Multiple R-squared:  0.06246,    Adjusted R-squared:  0.0606
## F-statistic: 33.58 on 1 and 504 DF,  p-value: 1.207e-08
```

#6. Interpret the results of your analysis in a comment. Make sure to mention the pvalue, the adjusted R-squared, the list of significant predictors and the coefficient for each significant predictor.

```
#We tried analyzing the predictions for four different types of independent variables:
#With independent variables crim + rm + dis, p-value is < 2.2e-16 whereas Adjusted R-squared:  0.5399,
#With a single independent variable crim, p-value is < 2.2e-16 whereas Adjusted R-squared:  0.1491,
#With a single independent variable rm, p-value is < 2.2e-16 whereas Adjusted R-squared:  0.4825,
#And finally, with a single independent variable dis, p-value is 1.207e-08 whereas Adjusted R-squared:
#Of these, independent variables crim + rm + dis produce the highest fitting variables.
```

#7. Create a one-row data frame that contains some plausible values for the predictors. For example, this data frame contains the median values for each predictor:

```
predDF <- data.frame(crim = 0.26, dis=3.2, rm=6.2)
predDF
```

```
##   crim dis  rm
## 1 0.26 3.2 6.2
```

#The numbers used here were selected randomly by looking at min and max data of the variables.

#8. Use the predict( ) command to predict a new value of medv from the onerow data frame. If you stored the output of your lm model in lmOut, the command would look like this:

```
predict(lmOut, predDF)
```

```
##        1
## 22.60355
```