Day 2: Linear Regression

Your Name

9/8/2021

AGENDA

1. initial data exploration
    - general principles
    - `apply` function
    - graphics
2. what is a linear regression?
    - the Normal distribution
    - OLS derivation
    - OLS assumptions
3. doing linear regression
    - interpreting output
    - hypothesis testing
    - generating CIs
4. diagnostics and model building
    - residual plots, Q-Q plots, etc.

Hint: for any equations look at the pdf to see the full expanded version

1. initial data exploration

let's begin by loading our dataset and looking at some of the data:

```
library(datasets)
data(iris)
names(iris) <- tolower(names(iris))
head(iris)
```

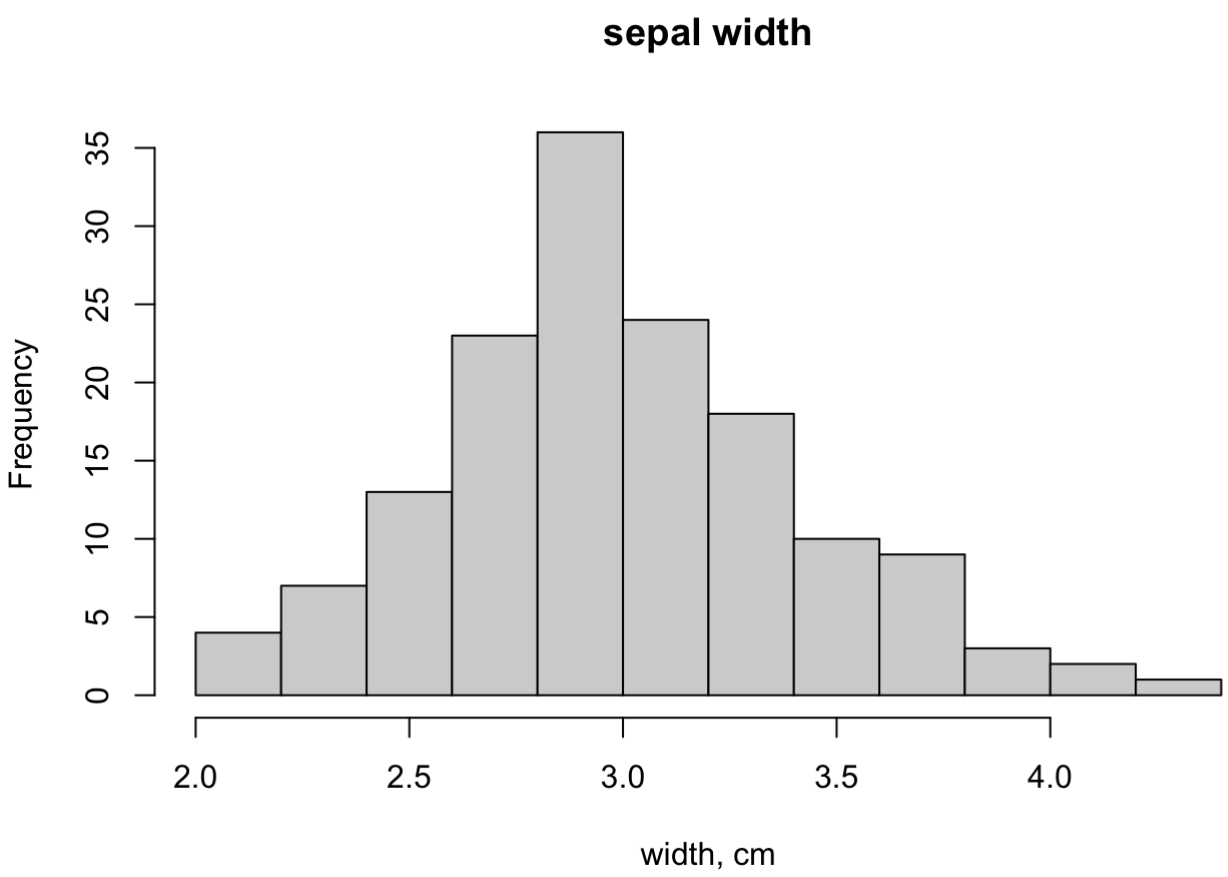
	sepal.length <dbl>	sepal.width <dbl>	petal.length <dbl>	petal.width <dbl>	species <fctr>
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

6 rows

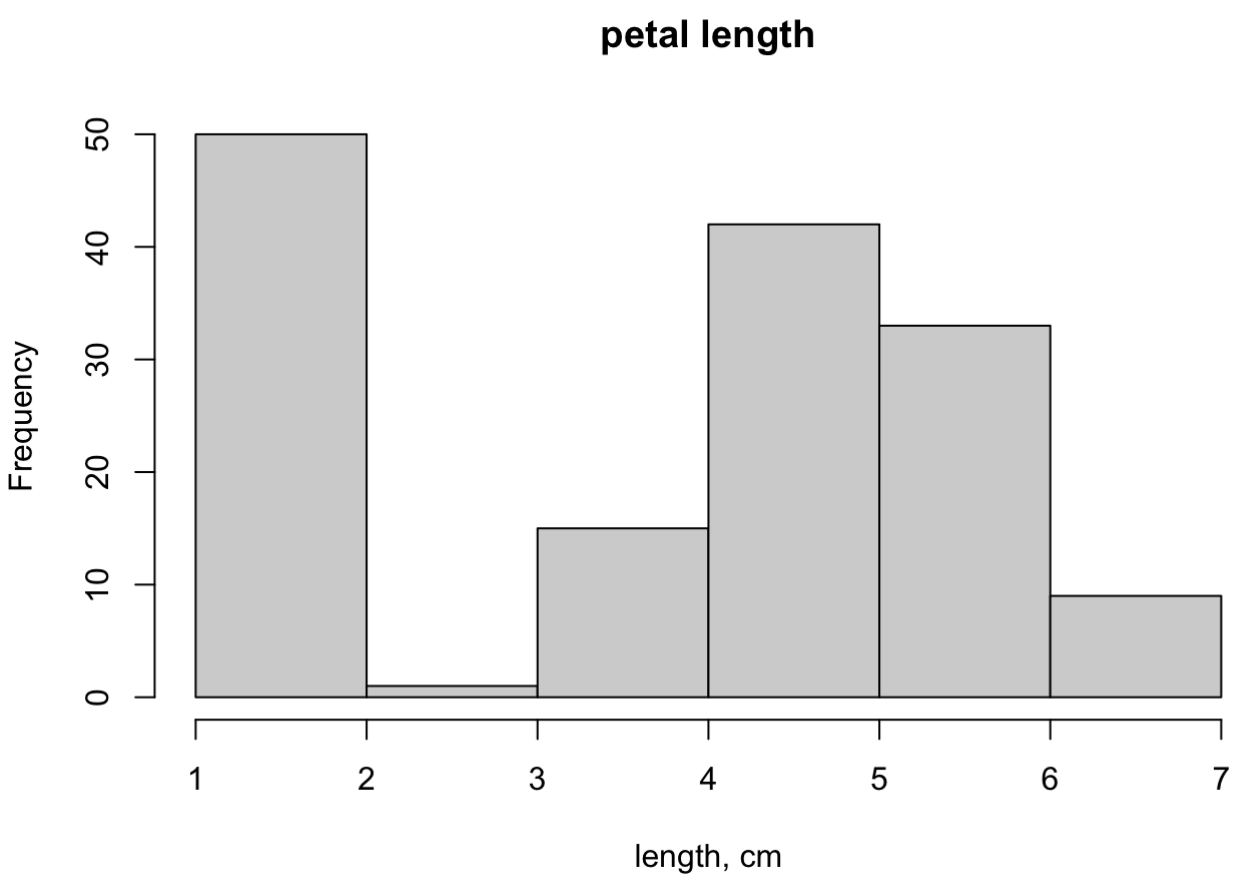
the first thing we want to do is look at the distributions and correlations in our data. easiest is to use the `hist()` function we (briefly) went over yesterday:

```
hist(iris$sepal.length,breaks="scott",main="sepal length",xlab="length, cm")
```

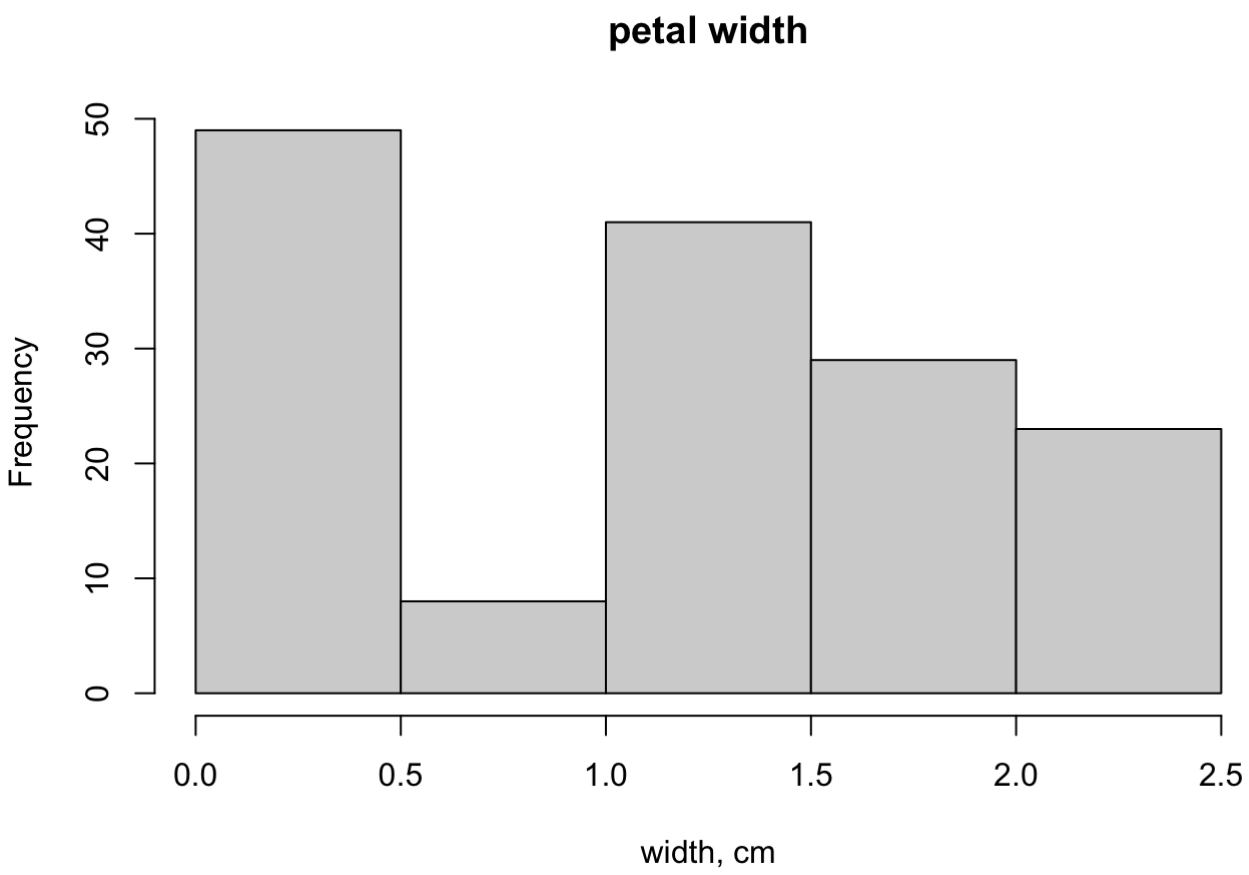

```
hist(iris$sepal.width,breaks="scott",main="sepal width",xlab="width, cm")
```




```
hist(iris$petal.length,breaks="scott",main="petal length",xlab="length, cm")
```




```
hist(iris$petal.width,breaks="scott",main="petal width",xlab="width, cm")
```



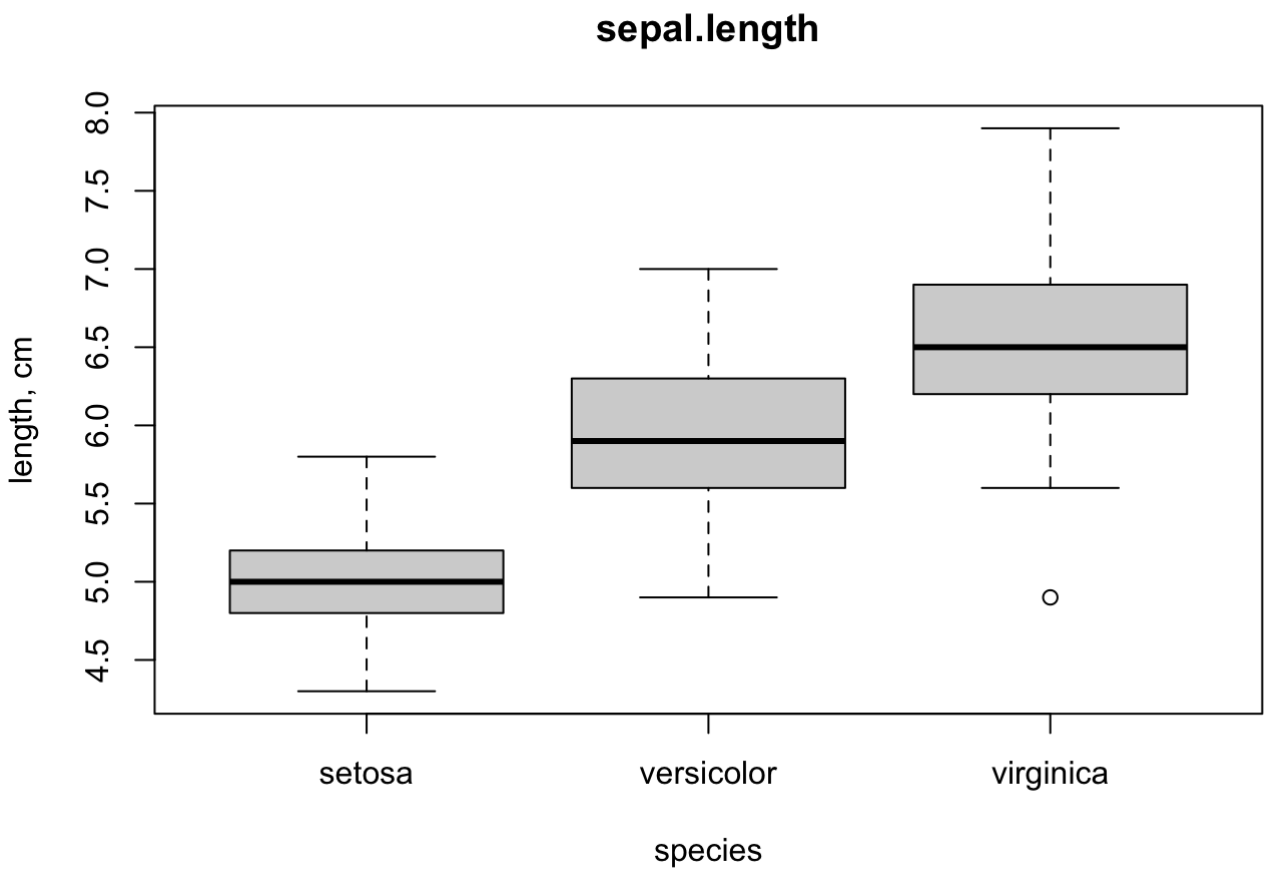
up next, let's take a look at the `summary` for each:

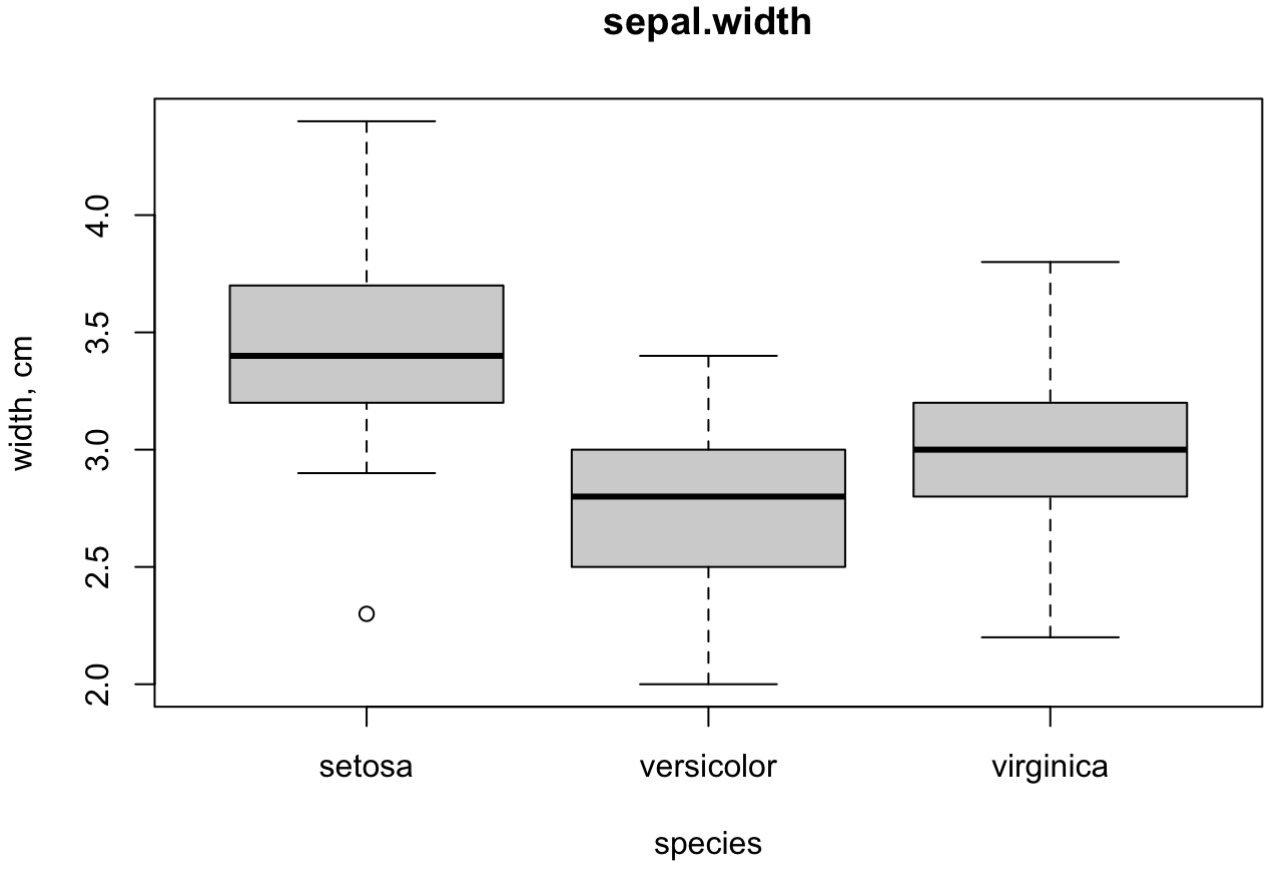
```
summary(iris)
```

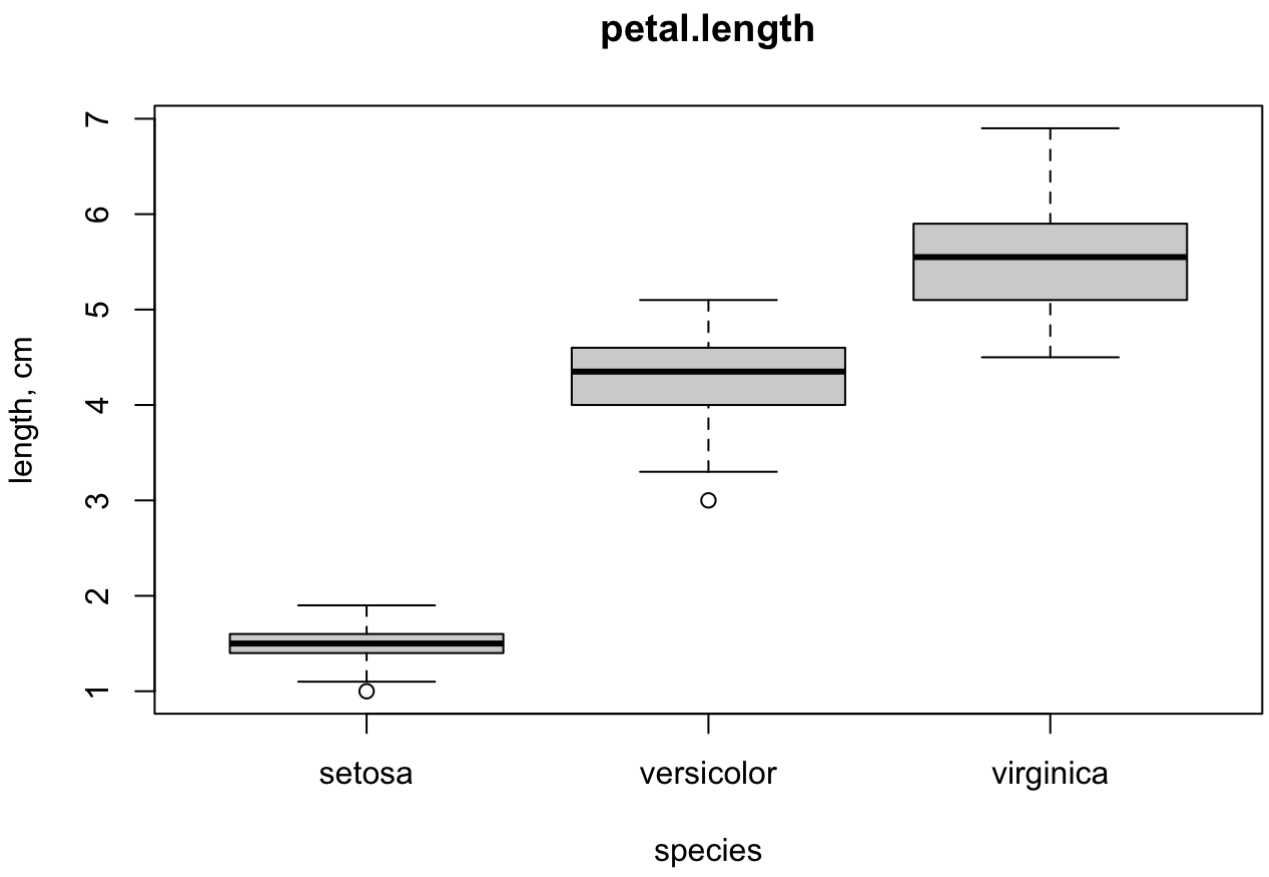
```
##   sepal.length    sepal.width     petal.length    petal.width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

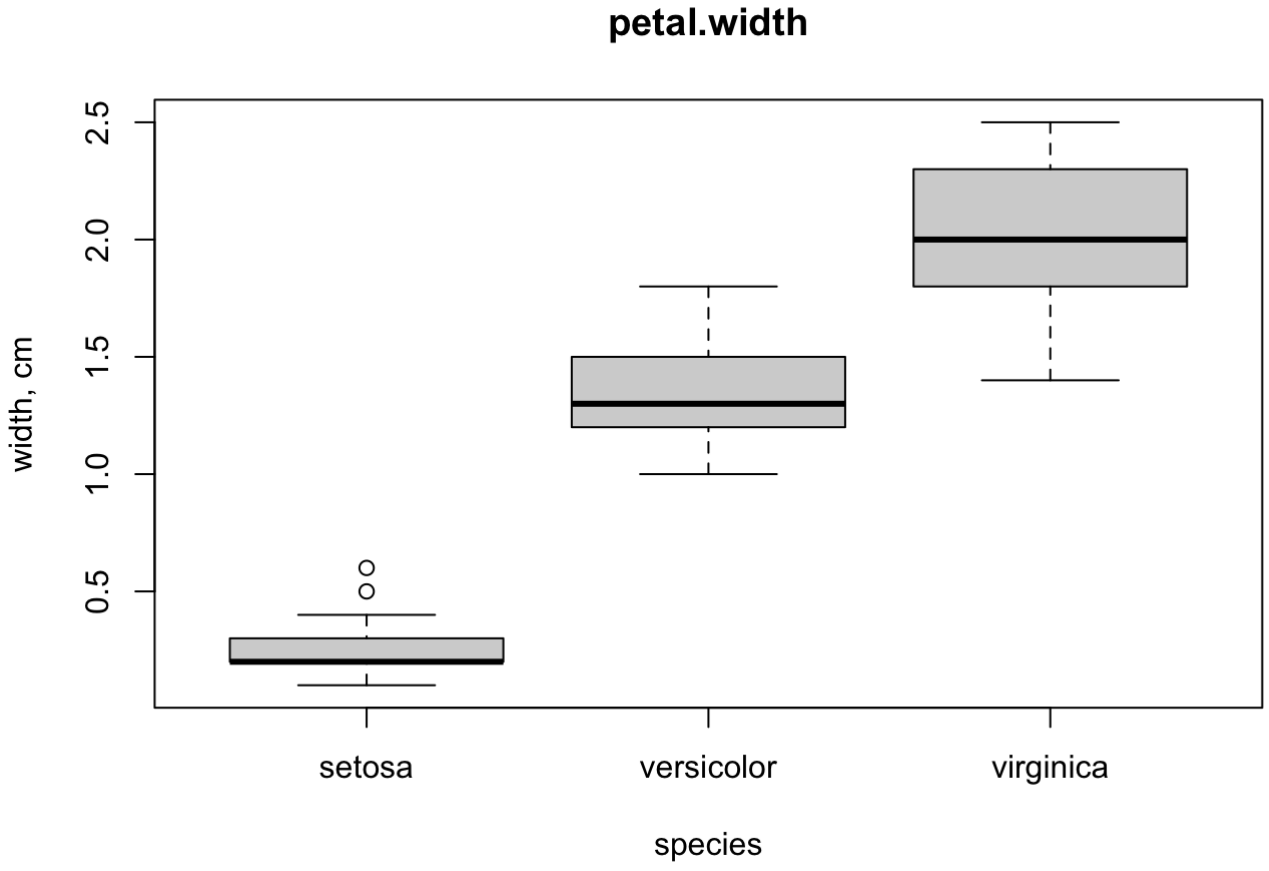
this isn't really that helpful, so let's talk about making boxplots:

```
make_boxplot<-function(d,n,l){
boxplot(d~iris$species,main=n,xlab="species",ylab=paste0(l,", cm"))
}
mapply(make_boxplot,iris[,1:4],names(iris)[1:4],c("length","width"))
```


sepal.length
length, cm
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
setosa
versicolor
virginica
species


sepal.width
width, cm
2.0
2.5
3.0
3.5
4.0
setosa
versicolor
virginica
species

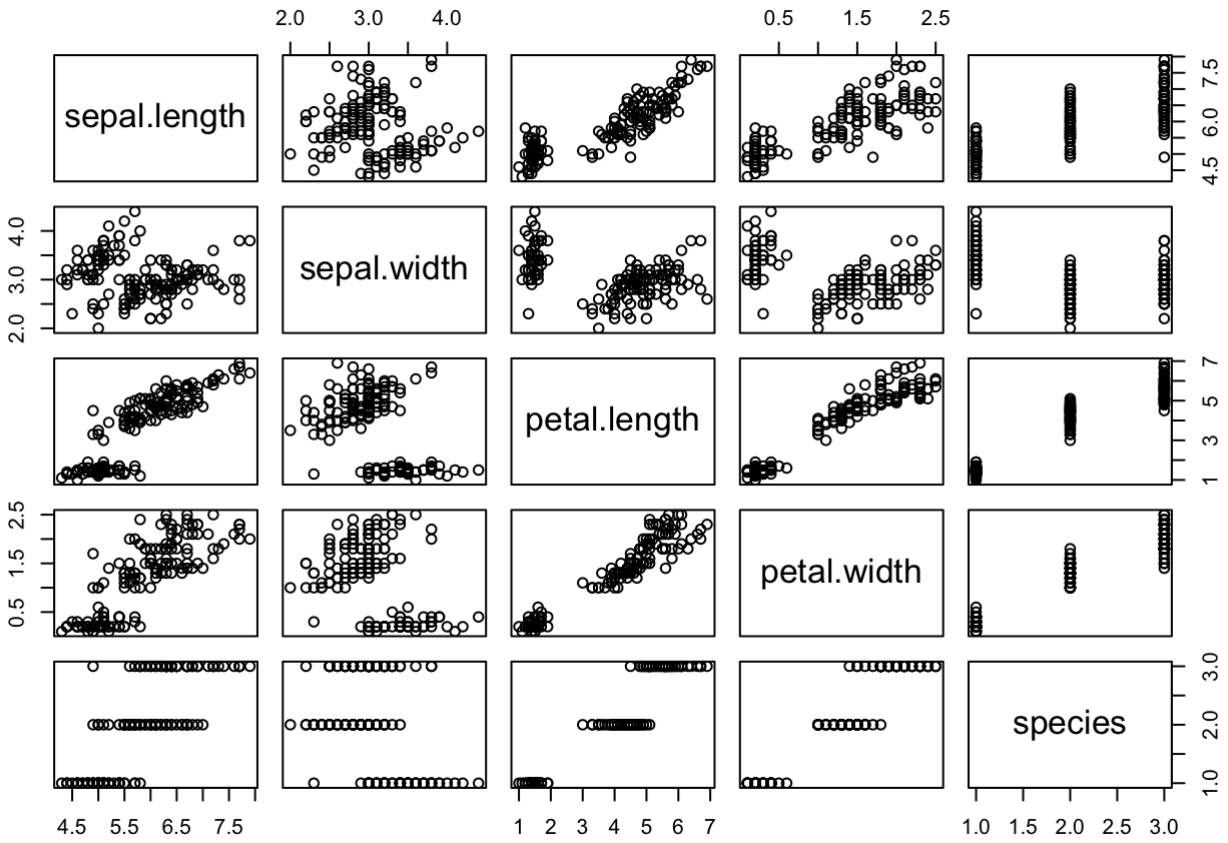

petal.length
length, cm
1
2
3
4
5
6
7
setosa
versicolor
virginica
species


petal.width
width, cm
0.5
1.0
1.5
2.0
2.5
setosa
versicolor
virginica
species

```
##       sepal.length sepal.width petal.length petal.width
## stats Numeric,15   Numeric,15  Numeric,15   Numeric,15
## n     Numeric,3    Numeric,3   Numeric,3    Numeric,3
## conf  Numeric,6    Numeric,6   Numeric,6    Numeric,6
## out   4.9          2.3         Numeric,2    Numeric,2
## group 3            1           Numeric,2    Numeric,2
## names Character,3  Character,3 Character,3  Character,3
```

finally, let's take a look at correlations in the data:

```
pairs(~.,data=iris)
```

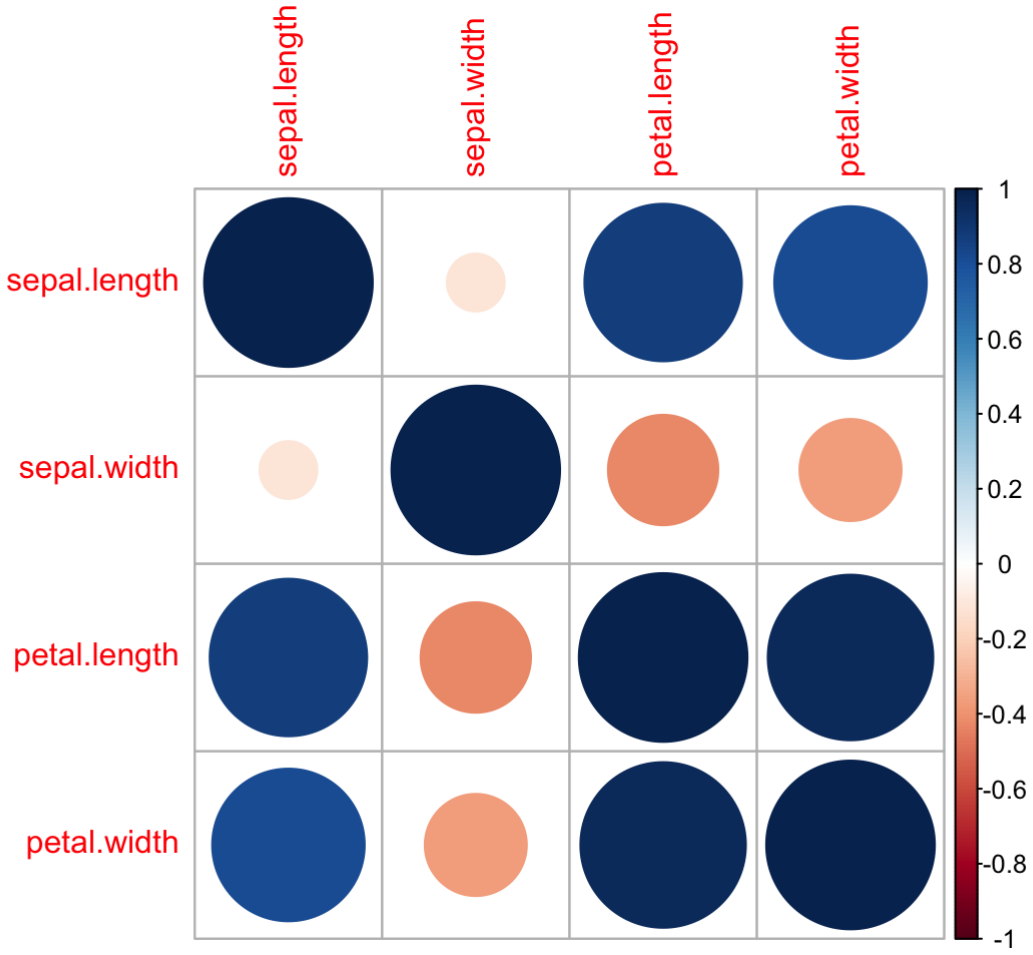


note that this only gives a very general overview, and it becomes less and less useful the number of covariates you have! when you have many, here's a good alternative, the 'correlogram':

```
library(corrplot)
```

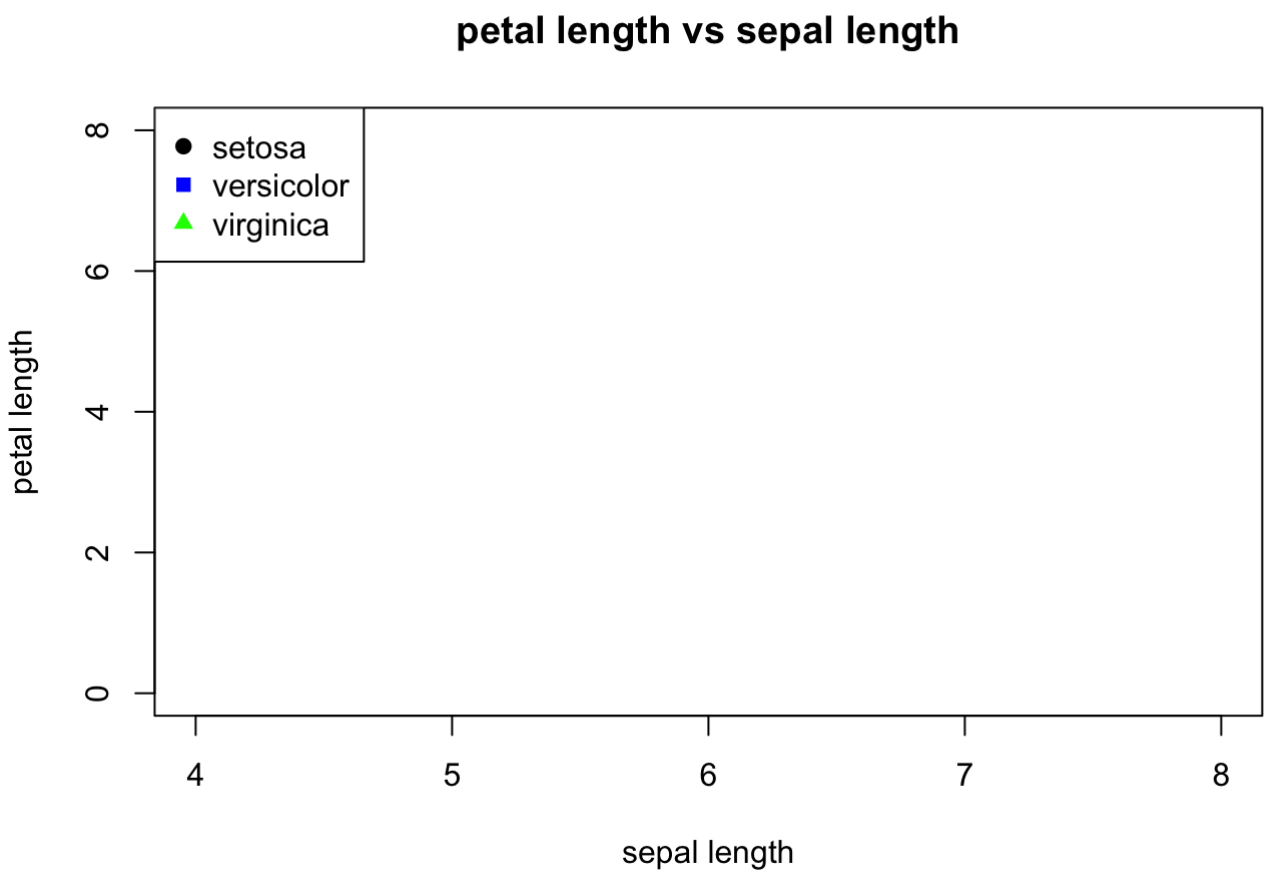
```
## corrplot 0.92 loaded
```

```
corrplot(cor(iris[,1:4]),method="circle")
```



finally, let's take a look at one of the promising correlations, sepal.length x petal.length. we're going to separate them by species:

```
idx1<-iris$Species=="setosa"
idx2<-iris$Species=="versicolor"
idx3<-iris$Species=="virginica"
plot(iris$Sepal.Length[idx1],iris$Petal.Length[idx1],
main="petal length vs sepal length",
xlab="sepal length",
ylab="petal length",
xlim=c(4,8),
ylim=c(0,8),
pch=19,col="black")
points(iris$Sepal.Length[idx2],iris$Petal.Length[idx2],
pch=15,col="blue")
points(iris$Sepal.Length[idx3],iris$Petal.Length[idx3],
pch=17,col="green")
legend("topleft",c("setosa","versicolor","virginica"),col=c("black","blue","green"),pch=
c(19,15,17))
```

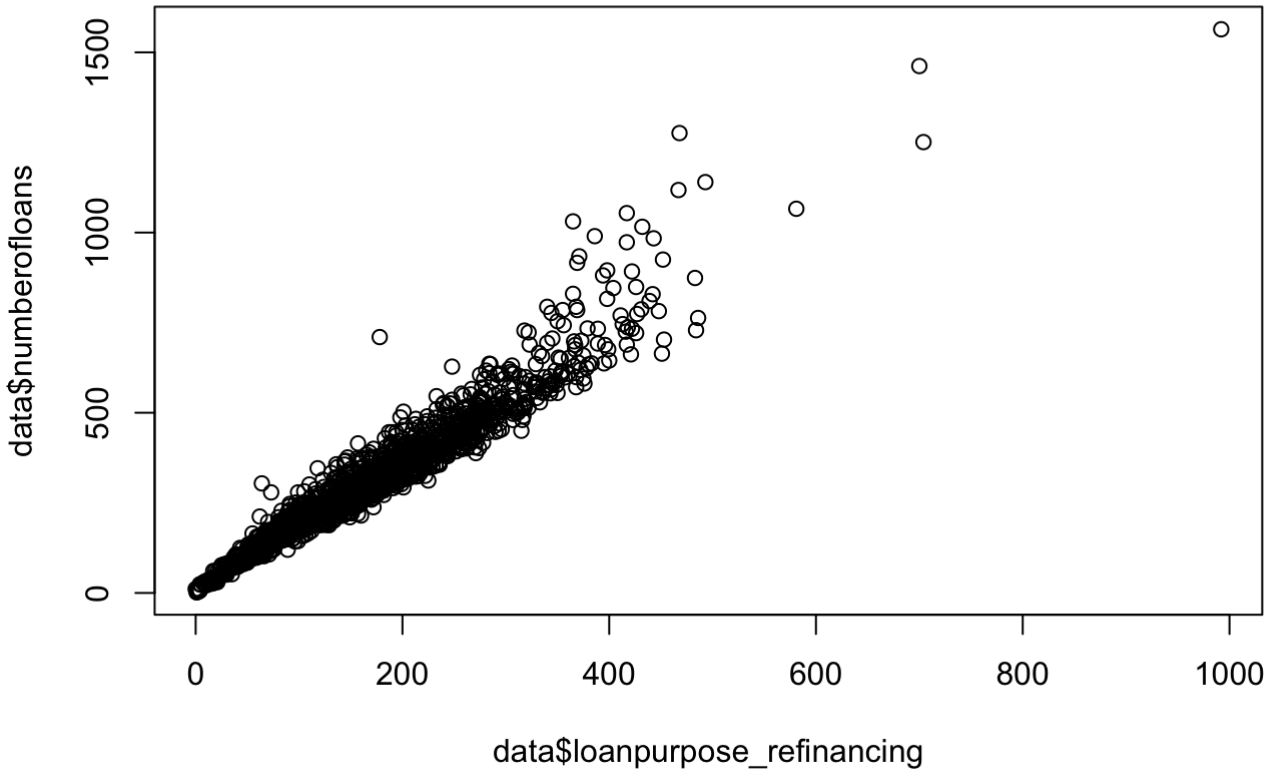


EXERCISE: now, it's your turn! take 10 minutes and analyze the "Washington Housing Loan Data" we uploaded for you in the google drive. please upload a scatterplot with two variables that have a strong (positive or negative) correlation. no need to split it by a category like we did above, that was more for 'learning how to plot in R' purposes than a directive lol

Hint: Make sure to download the data file WA_housing_loan_data.txt from the files provided AND make sure it is in the same folder as your R files

###Answer below

```
data = read.table('WA_housing_loan_data.txt',header=TRUE)
plot(data$loanpurpose_refinancing,data$numberofloans)
```



2. what is a linear regression?

Hint: If you would like ot see the equation form of the variables in between the $ signs, hover your mouse over it

the goal of linear regression is to model **response variable** Y as a function of the **covariate** X. in other words, let's say we observe data points $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$. what effect does X_i have on Y_i? how big is that effect?

of course, there can be many Xs for each Y.

linear regression assumes that the relationship between Y and our Xs is linear and maps it as:

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \cdots + \beta_n X_{ni} + \epsilon_i$$

normally, we formulate it in matrix form as:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

linear regression places three major assumptions on the **error** term of our equation:

Hint: look at the equations below in pdf form for more clarity ϵ represents the y intercept and β represents the slope*

*Hint: hover your cursor over the equations between the $ signs below to see the equation written out in the proper mathematic symbols

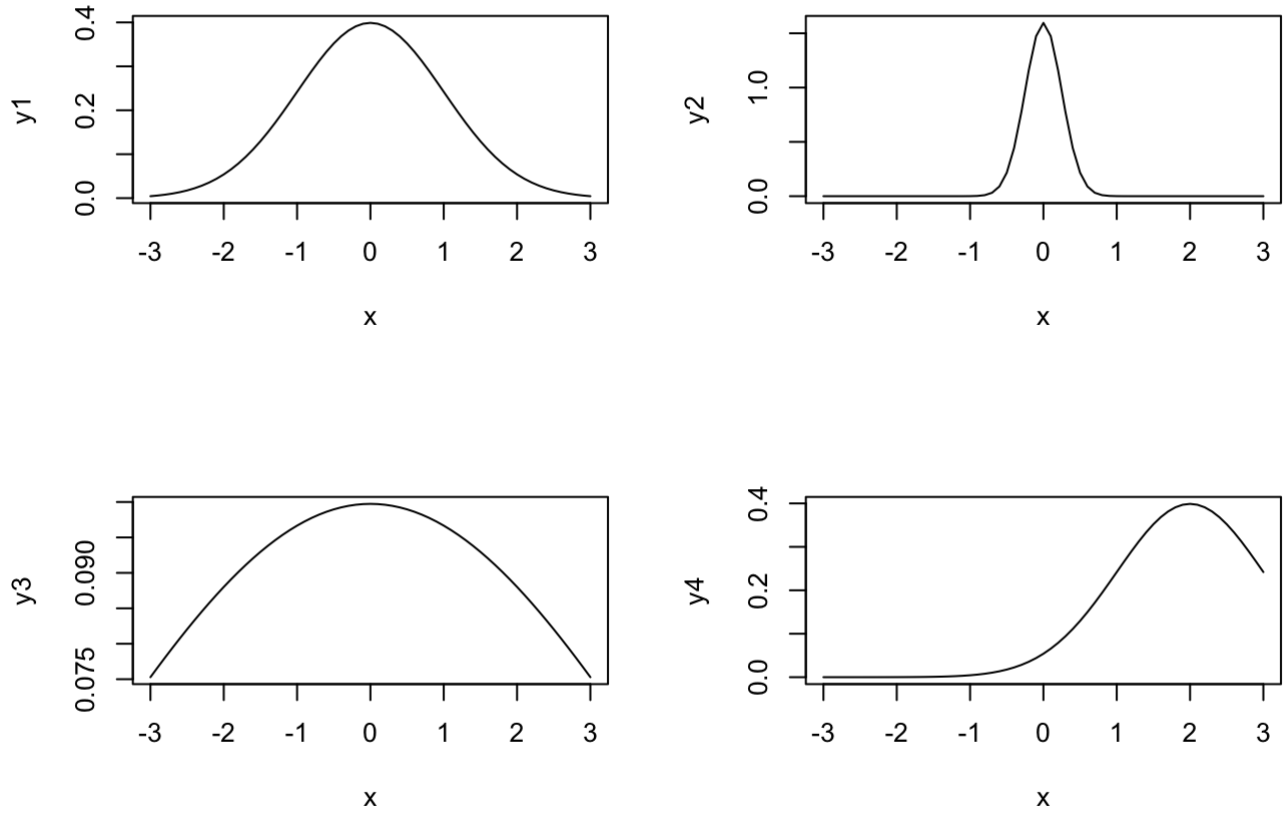
1. $\mathbf{E}[\epsilon] = 0$.
2. $\mathbf{Var}(\epsilon) = \sigma^2$ is constant for all points.
3. each ϵ_i is independent from the others.

note that these imply that each data point we collect is independent from the other data points.

OLS and the Normal Distribution

Ordinary Least Squares is the most basic form of linear regression. We place one more assumption on our errors, that $\epsilon_i \sim N(0, \sigma^2)$. this is equivalent to saying $Y \sim N(\beta_0 + \beta_1 X, \sigma^2)$. why? well, let's briefly review the normal distribution:

```
x <- seq(-3, 3, by=.1)
y1 <- dnorm(x)
y2 <- dnorm(x, sd=.25)
y3 <- dnorm(x, sd=4)
y4 <- dnorm(x, mean=2)
par(mfrow=c(2,2))
plot(x,y1, type="l")
plot(x,y2, type="l")
plot(x,y3, type="l")
plot(x,y4, type="l")
```



the normal distribution is entirely determined by its expectation and variance! which is why we love it for least squares.

the “least squares” portion is the description of the line of best fit. we can define “best fit” in many, many ways (some of which we’ll go over later in the course), but OLS defines it using squared error (hence “least squares”):

$$\begin{aligned} Y_i &= \beta_0 + \beta_1 X_i + \epsilon_i \\ Y_i - (\beta_0 + \beta_1 X_i) &= \epsilon_i \\ \sum_i (Y_i - (\beta_0 + \beta_1 X_i))^2 &= \sum_i \epsilon_i^2 = RSS \end{aligned}$$

we can use calculus (one-variable case) or matrix calculus (multivariate case) to find the values of β_i that minimize this equation, i.e. the *least* value of the sum-of-*squares*. alternatively, we can have R do it for us.

---

3. doing linear regression

the command for a linear regression is very easy!

```
reg<-lm(iris$petal.length~iris$sepal.length)
summary(reg)
```

```
##
## Call:
## lm(formula = iris$petal.length ~ iris$sepal.length)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.47747 -0.59072 -0.00668  0.60484  2.49512
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -7.10144    0.50666  -14.02   <2e-16 ***
## iris$sepal.length  1.85843    0.08586   21.65   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8678 on 148 degrees of freedom
## Multiple R-squared:   0.76,  Adjusted R-squared:  0.7583
## F-statistic: 468.6 on 1 and 148 DF,  p-value: < 2.2e-16
```

things to notice:

- formula call
- residuals
- coefficients
- residual standard error
- R^2
- F-statistic

the p-values are important because we are testing a hypothesis. in this case, we are specifically testing whether $\beta_1 = 0$, and we want to reject this if there is a relationship!

for the curious, here is how the following parameters are calculated:

Standard error of $\hat{\beta}_j$:

$$SE(\hat{\beta}_j) = \hat{\sigma}\sqrt{(X^T X)^{-1}_{jj}}$$

it can be shown that $\frac{\hat{\beta}-\beta}{SE(\hat{\beta})} \sim t_{n-p}$, which is how we derive the calculation of the confidence intervals:

$$\hat{\beta}_j \pm t^*_{1-\alpha/2;n-p} SE(\hat{\beta}_j)$$

EXERCISE: write a function that takes in a vector of βs and a vector of standard errors, and returns a list of the symmetric 95% confidence interval for each of them.

hint: you'll need to pass it n (the number of data points) to calculate the degrees of freedom for t, and you can get the 97.5% (= 2.5% due to symmetry of the t-distribution) using the function `dt()`. we are using 97.5% and 2.5% because we want the CI to be symmetric, not single-tailed.

```
# your code here!
```

###Answer below

```
confint(reg)
```

```
##                       2.5 %    97.5 %
## (Intercept)       -8.102670 -6.100217
## iris$sepal.length  1.688772  2.028094
```

```
ci<-function(b,se,n){
    #calculate critical t
    tcrit<-qt(.975,n-length(b))

    #hold the CIs
    ci_list<-vector(mode="list",length=length(b))

    for(i in 1:length(b)){
        upper<-b[i]+tcrit*se[i]
        lower<-b[i]-tcrit*se[i]
        ci_list[[i]]<-c(lower,upper)
    }
    return(ci_list)

}
(cf<-summary(reg)$coef)
```

```
##                   Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)       -7.101443 0.50666229 -14.01613 6.133586e-29
## iris$sepal.length  1.858433 0.08585565  21.64602 1.038667e-47
```

```
ci(cf[,1],cf[,2],n=150)
```

```
## [[1]]
## (Intercept) (Intercept)
##   -8.102670   -6.100217
##
## [[2]]
## iris$sepal.length iris$sepal.length
##          1.688772          2.028094
```

what happens when we use the categorical covariate species, instead of the quantitative petal.length? well, the regression equation takes the form

$$Y_i = \beta_0 + \mathbf{I}_{X=1}\beta_1 + \mathbf{I}_{X=2}\beta_2$$

where $\mathbf{I}$ is an **indicator function**. let's see what the regression looks like:

```
reg2<-lm(petal.length~species,data=iris)
summary(reg2)
```

```
##
## Call:
## lm(formula = petal.length ~ species, data = iris)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -1.260 -0.258  0.038  0.240  1.348
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1.46200    0.06086   24.02   <2e-16 ***
## speciesversicolor  2.79800    0.08607   32.51   <2e-16 ***
## speciesvirginica   4.09000    0.08607   47.52   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4303 on 147 degrees of freedom
## Multiple R-squared:  0.9414, Adjusted R-squared:  0.9406
## F-statistic:  1180 on 2 and 147 DF,  p-value: < 2.2e-16
```

the interpretation here is a bit more involved, but the gist is the same: the intercept describes the effect of category 0 (species = setosa), whereas β_1 is **the effect of category 1 (species = versicolor) not already explained by species 0**. it is a *difference*, not an objective measure! even though the category order does not affect the regression, it does affect the interpretation.

---

4. diagnostics and model building

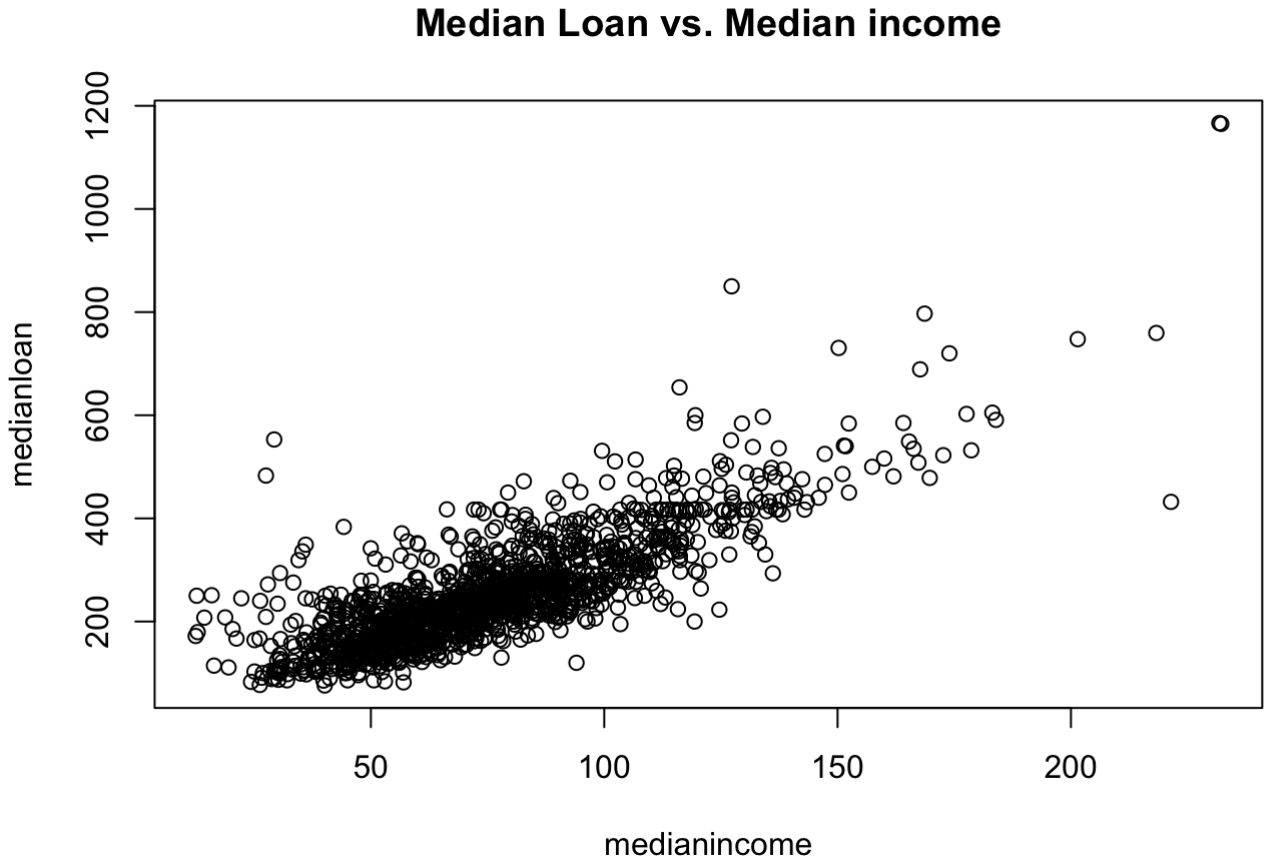
Note: you may see an error message saying "the following objects are masked from data" when running the code below, this is no reason to worry, the code will run properly

```
data = read.table('WA_housing_loan_data.txt',header=TRUE)
attach(data)
```

Let us look at some of the correlations between the variables from our table:

Here we are comparing median income and median loan values to see the correlation between them and any outliers present

```
plot(medianincome,medianloan,main="Median Loan vs. Median income")
```



Let us know run a linear model of this dataset and take a look at a summary of our values numerically:

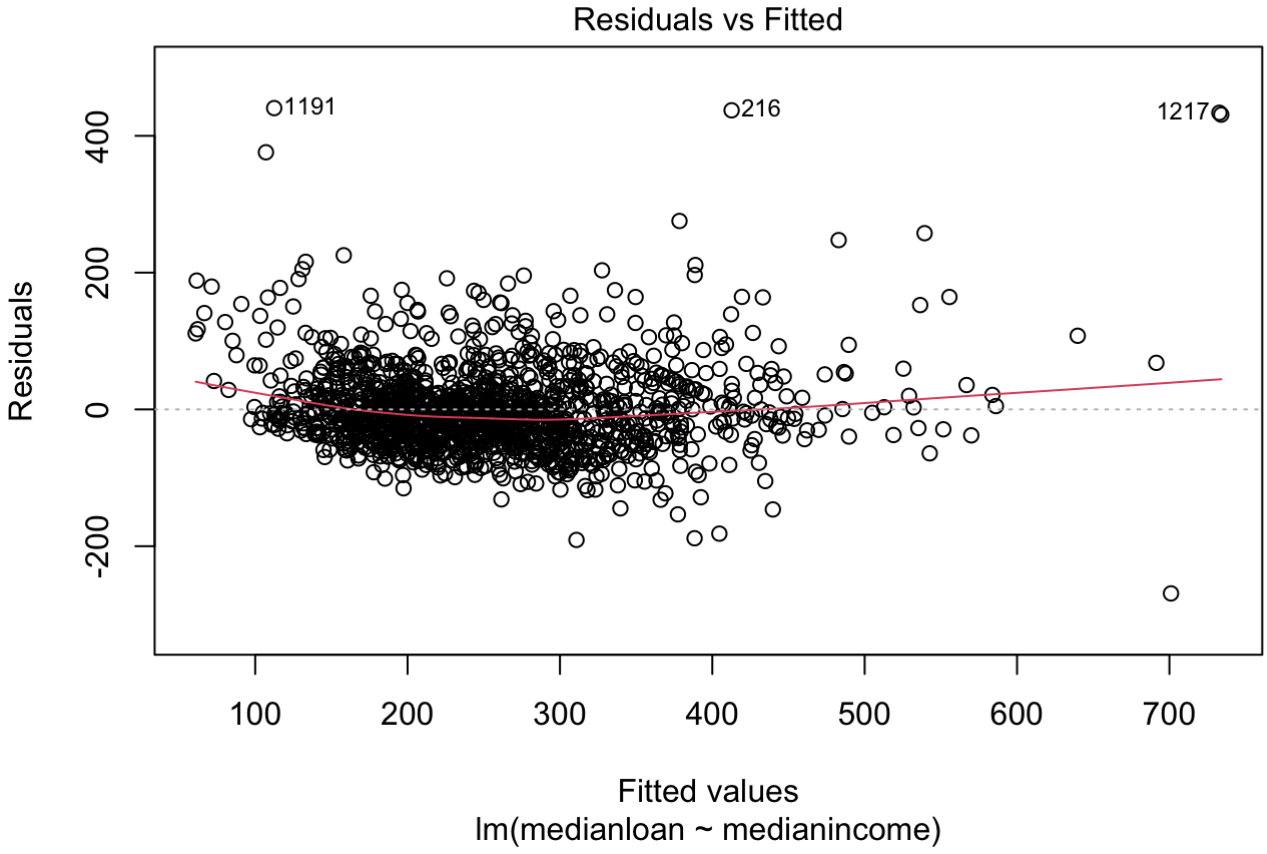
```
model<-lm(medianloan~medianincome)
summary(model)
```

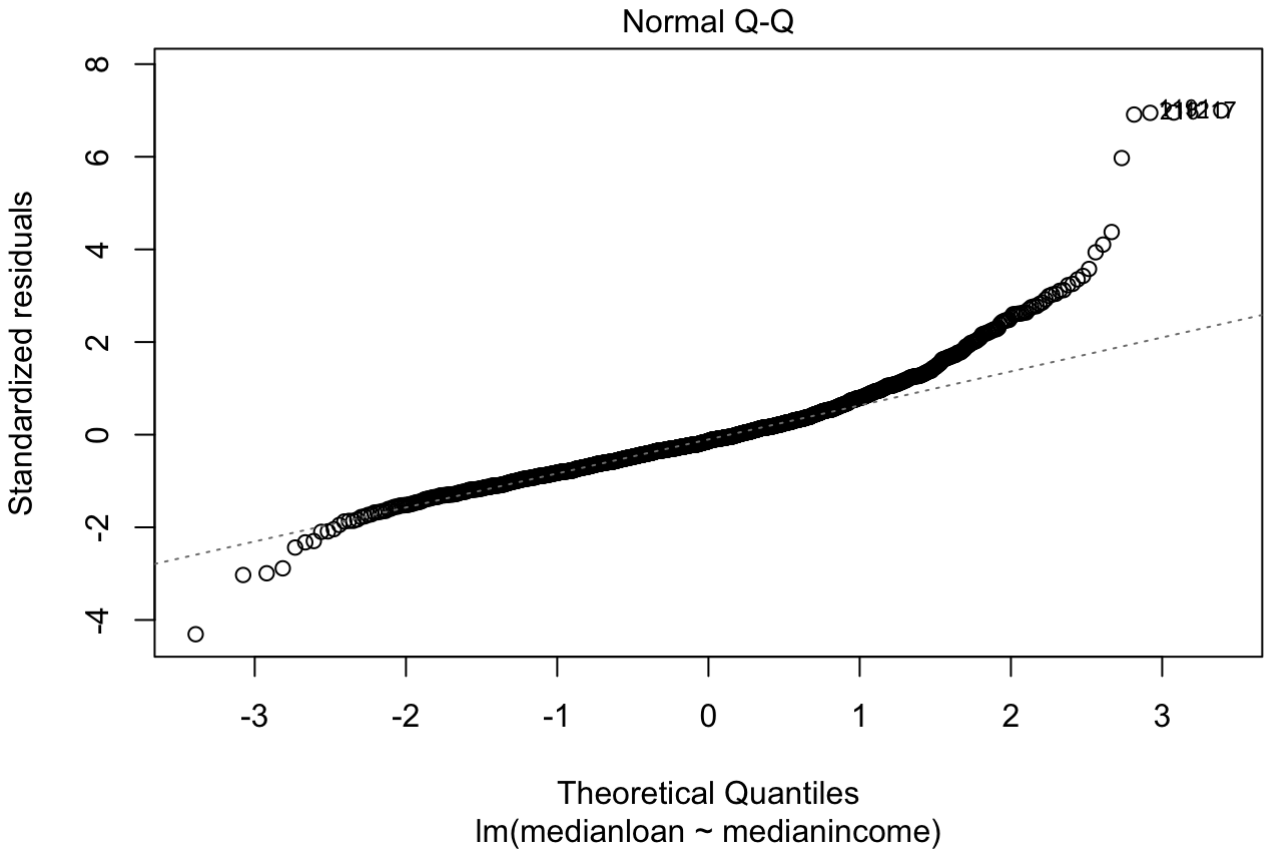
```
##
## Call:
## lm(formula = medianloan ~ medianincome)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -269.03  -37.68   -8.68   24.70  440.54
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  22.76044    4.67231   4.871 1.23e-06 ***
## medianincome  3.06307    0.05771  53.079  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 63.04 on 1431 degrees of freedom
## Multiple R-squared:  0.6632, Adjusted R-squared:  0.6629
## F-statistic:  2817 on 1 and 1431 DF,  p-value: < 2.2e-16
```

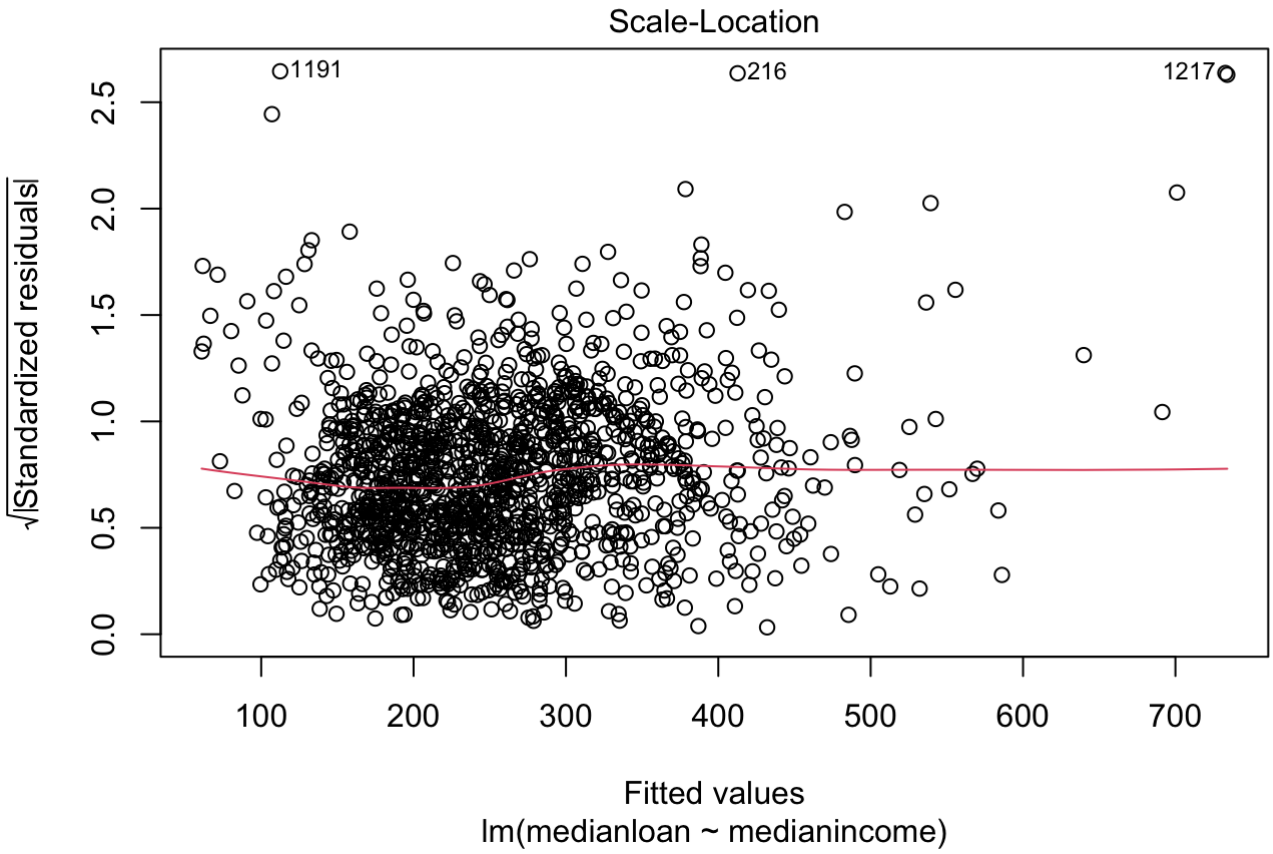
We can take a look at the values provided in our summary above to determine how good of a correlation the two variables we are plotting have.

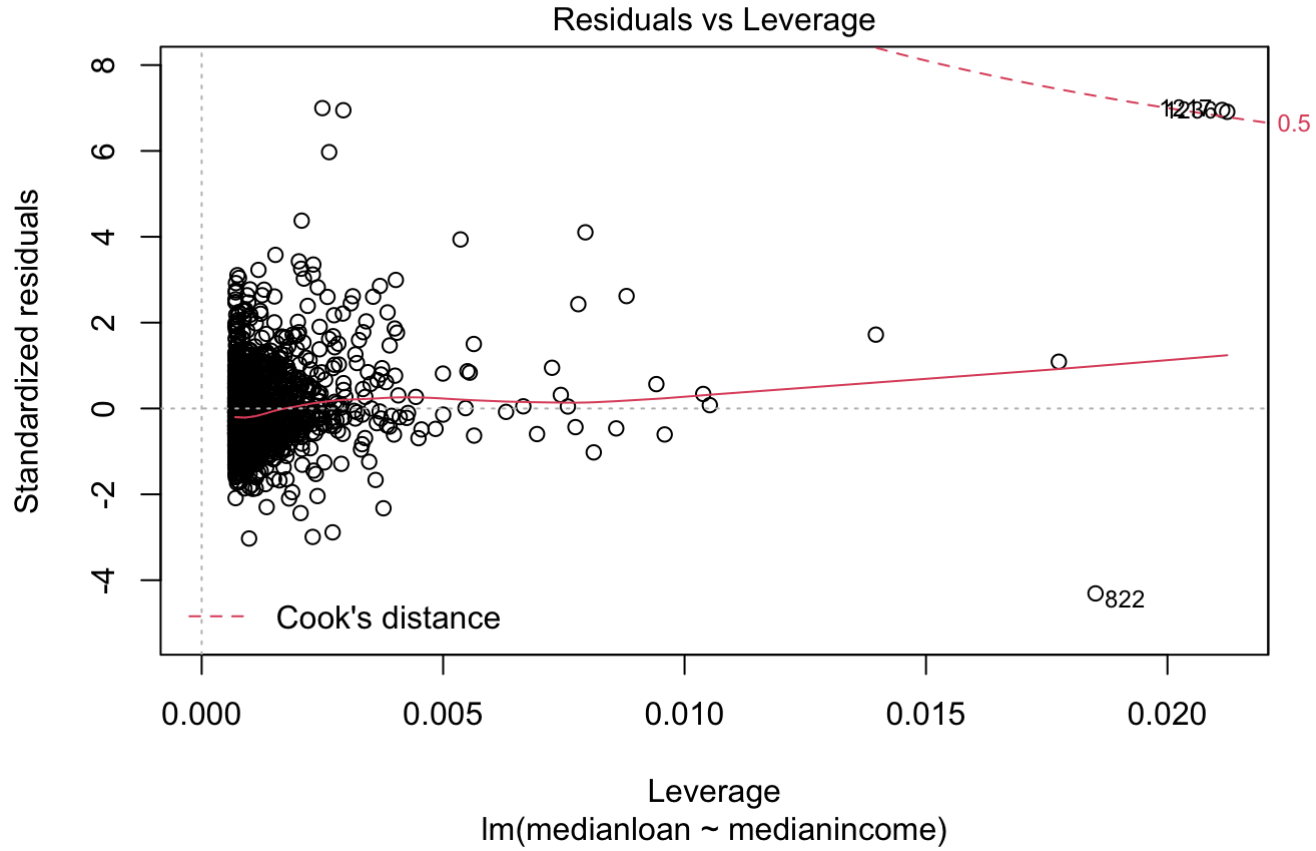
Let us now look at a diagnostic model of all of our values:

```
plot(model)
```


Residuals vs Fitted
1191
216
1217
Residuals
400
200
0
-200
100
200
300
400
500
600
700
Fitted values
lm(medianloan ~ medianincome)


Normal Q-Q
Standardized residuals
8
6
4
2
0
-2
-4
-3
-2
-1
0
1
2
3
Theoretical Quantiles
lm(medianloan ~ medianincome)





Some things to note when reading these plots: * We want our residuals to be close to 0 with a constant variance * We should see random noise in our plot * The red line we see should be about straight, otherwise it shows that

our model does not have a linear relationship * The scale location plot shows us if the standard deviation is constant along a range of values - funnel shapes show us that they are not constant and we need to correct something * Residuals vs leverage shows us points with high leverage - basically how influential a point is on the line of best fit - as long as all of our datapoints fall close to our line of best fit we do not have anything skewing our data much, otherwise they are outliers and have high leverage (meaning have a big effect on the dataset)

Let us create another model with different variables in our dataframe:

```
model2<-lm(medianloan~medianincome+numberofloans+owneroccupied+percentminority)
summary(model2)
```

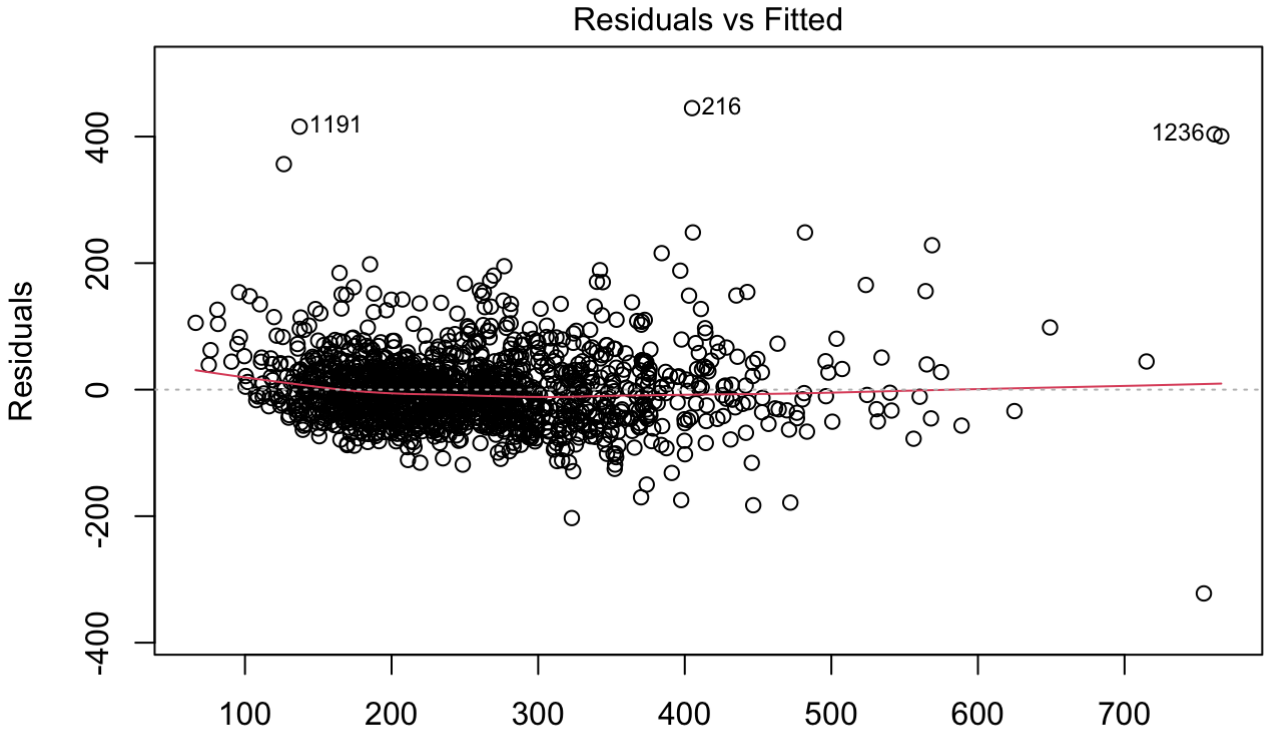
```
##
## Call:
## lm(formula = medianloan ~ medianincome + numberofloans + owneroccupied +
##     percentminority)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -322.09  -36.62   -6.30   26.87  445.02
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      9.083770   6.528247   1.391 0.164304
## medianincome     3.313966   0.061576  53.819  < 2e-16 ***
## numberofloans    0.048854   0.013751   3.553 0.000394 ***
## owneroccupied   -0.036023   0.004956  -7.269 5.93e-13 ***
## percentminority  0.786811   0.096229   8.176 6.39e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 59.9 on 1428 degrees of freedom
## Multiple R-squared:  0.6965, Adjusted R-squared:  0.6957
## F-statistic: 819.4 on 4 and 1428 DF,  p-value: < 2.2e-16
```

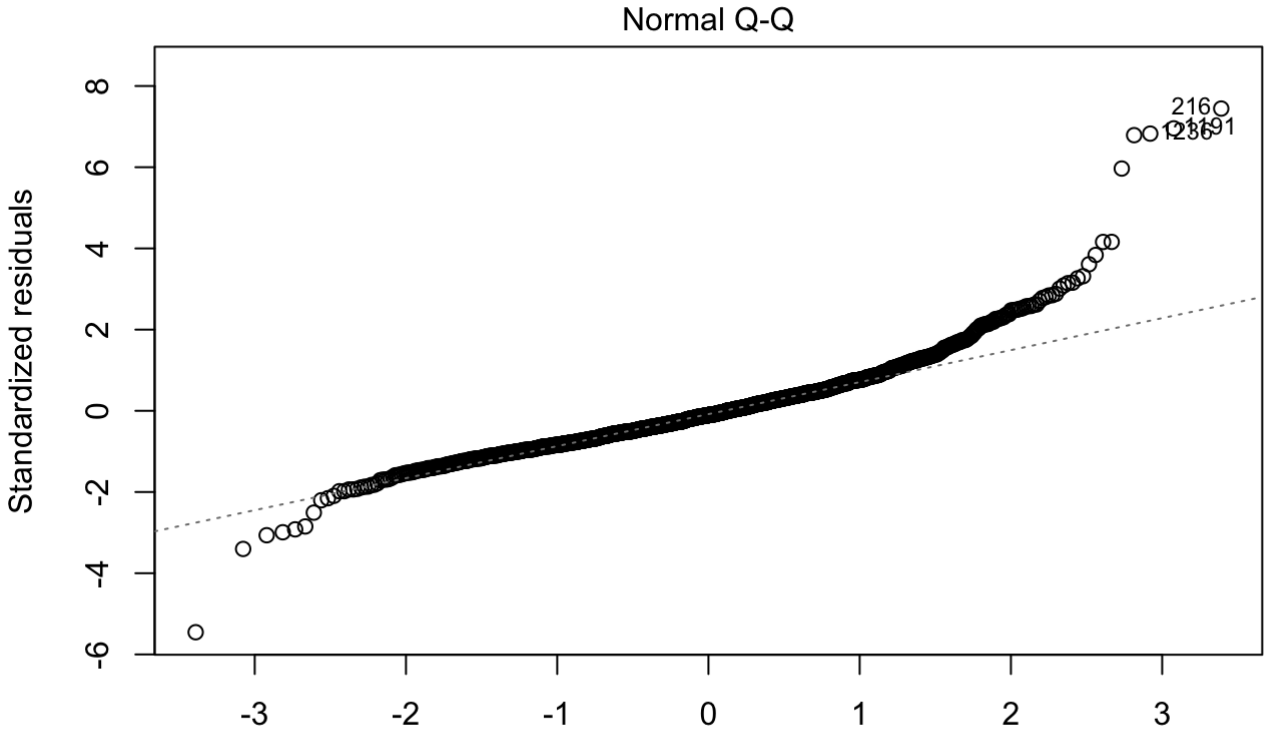
We can use the values above to determine if statistically our data frame fits what we are seeking for (ie; if we are looking for a linear relationship, are we getting that?) Note: Understanding this summary requires some statistics knowledge

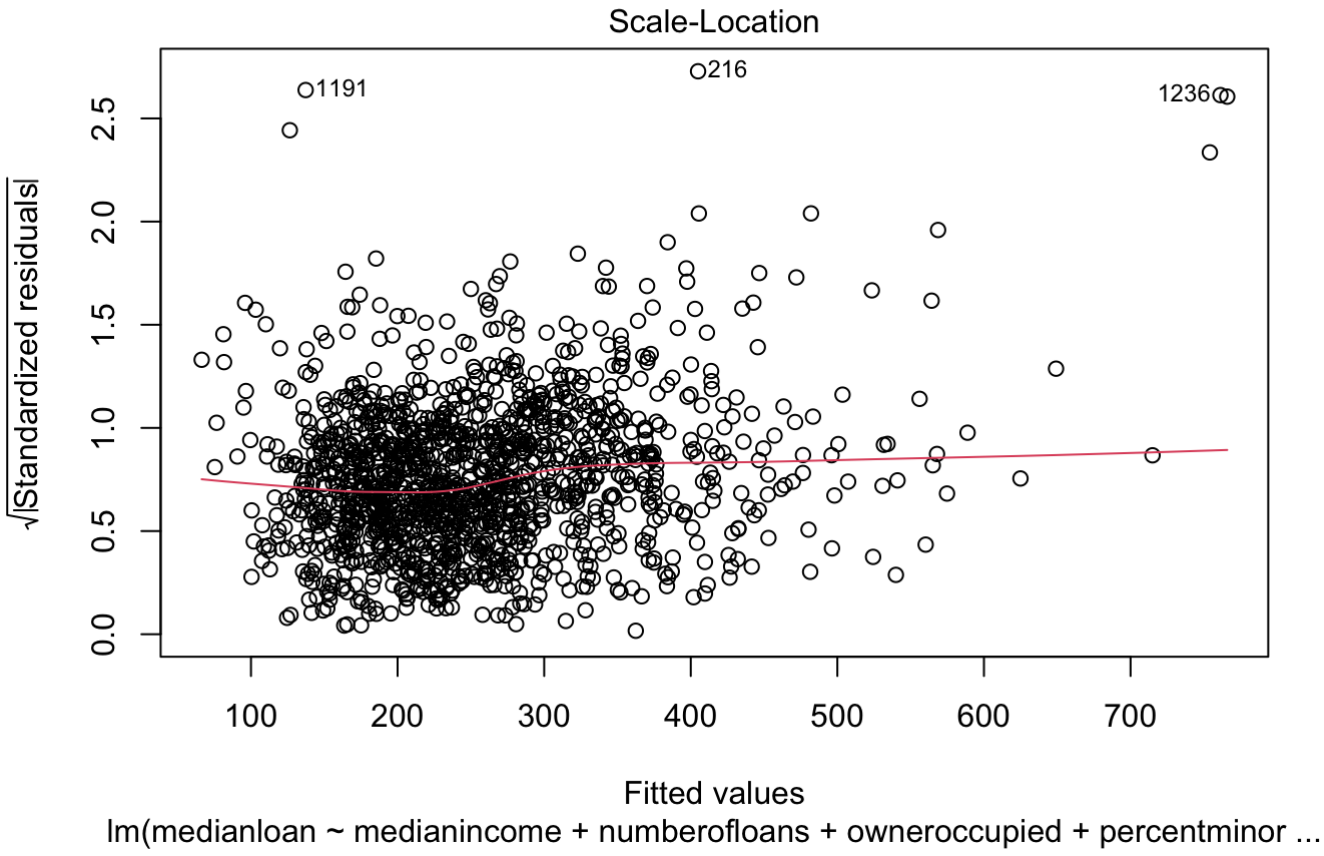
P.S. If you don't want to select particular variables from your data frame you can just type lm(dataframename~.) and it will add all the variables. If you want to include all except one you can type lm(dataframename~.-notincluding) where notincluding is the name of the variable you are leaving out

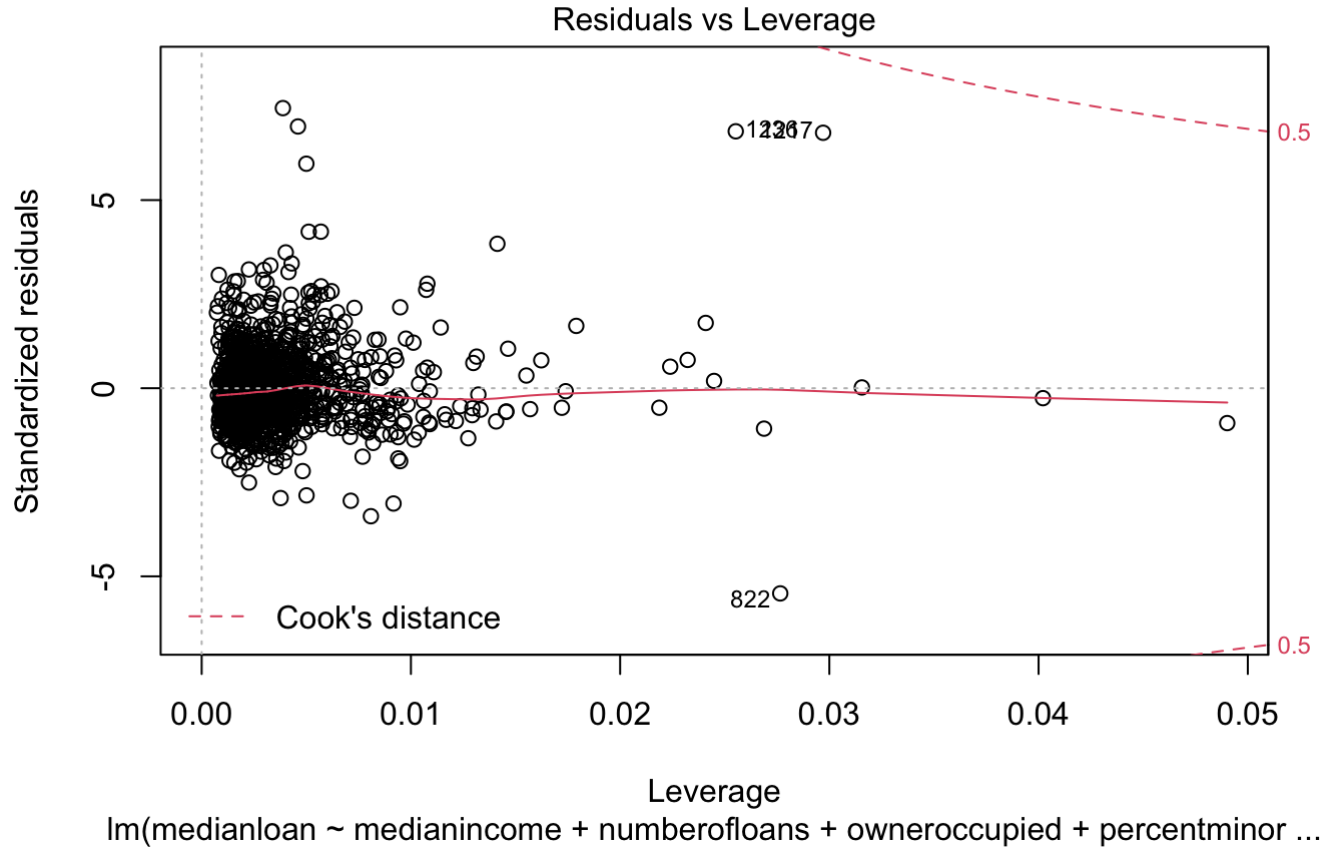
Let us plot this second model we created above:

```
plot(model2)
```


Residuals vs Fitted
1191
216
1236
Residuals
400
200
0
-200
-400
100
200
300
400
500
600
700
Fitted values
lm(medianloan ~ medianincome + numberofloans + owneroccupied + percentminor ...


Normal Q-Q
216
1191
1236
Standardized residuals
8
6
4
2
0
-2
-4
-6
-3
-2
-1
0
1
2
3
Theoretical Quantiles
lm(medianloan ~ medianincome + numberofloans + owneroccupied + percentminor ...





Note: Watch out for units, if the units differ we should make them the same

We will be doing that below for our numberofloans and owneroccupied variables as they are of a different unit than the other variables

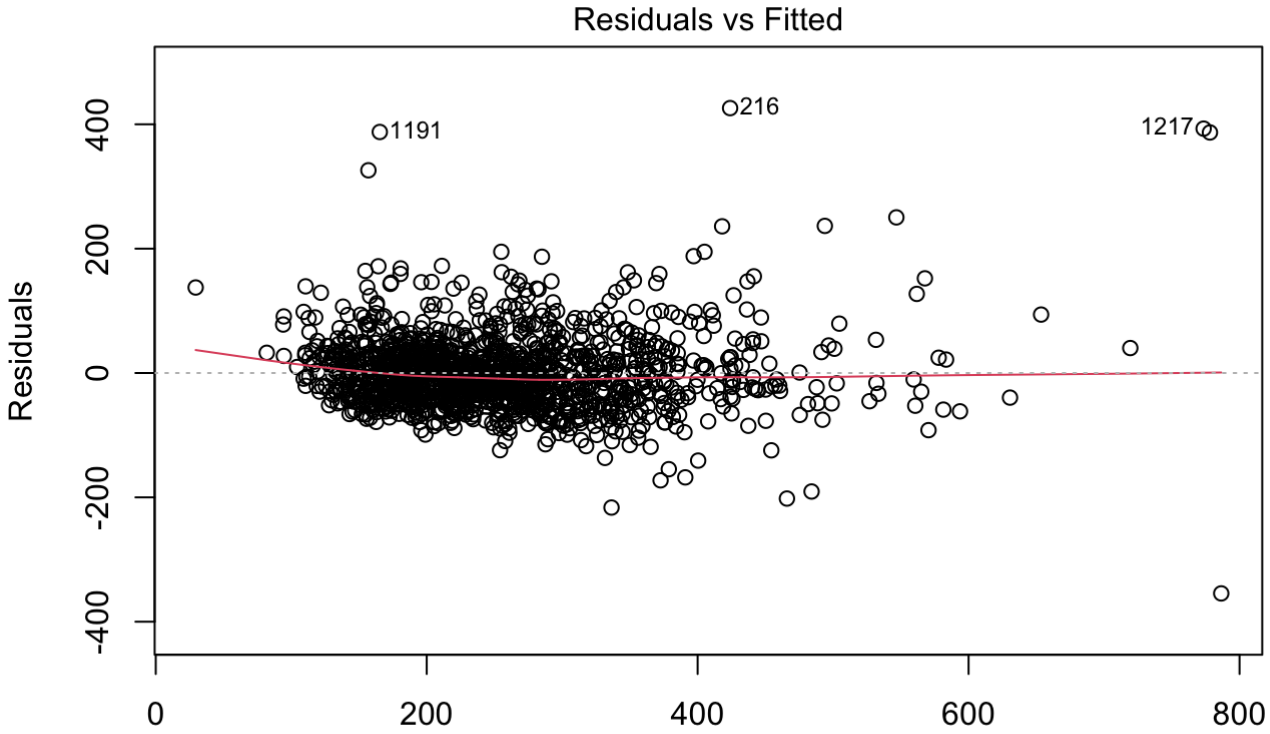
```
loanspercapita<-numberofloans/population
oopercapita<-owneroccupied/population
model3<-lm(medianloan~medianincome+loanspercapita+oopercapita+percentminority)
```

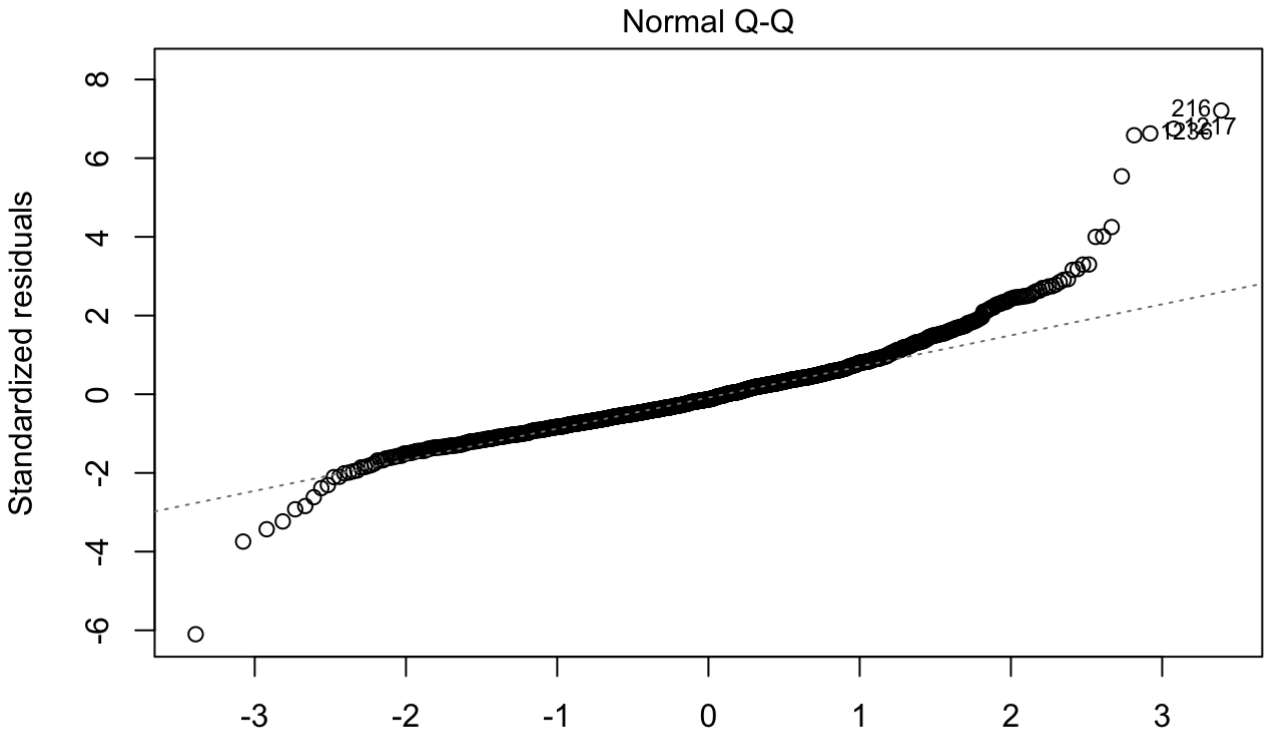
Once again let us look at the summary and plot of our new model to determine if the dataframe fits our intended purpose:

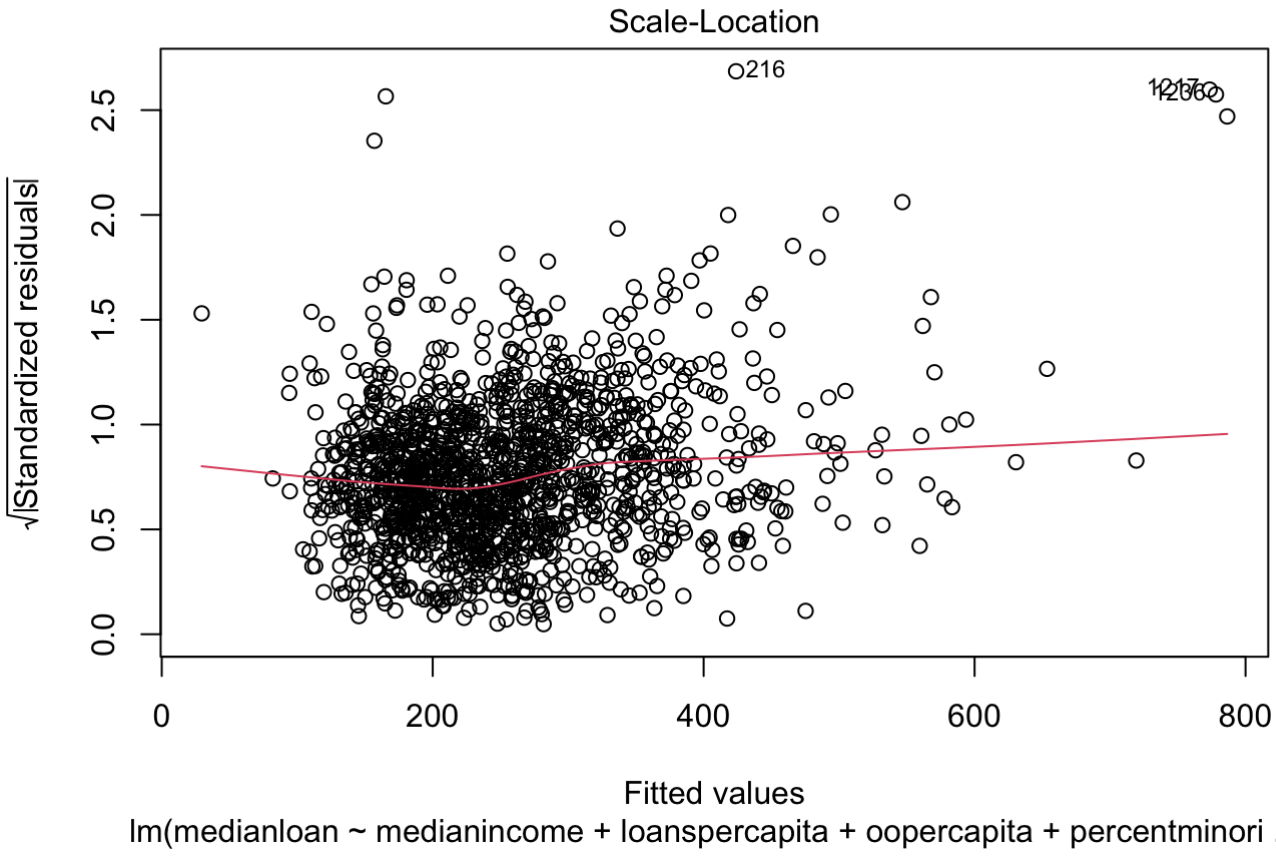
```
summary(model3)
```

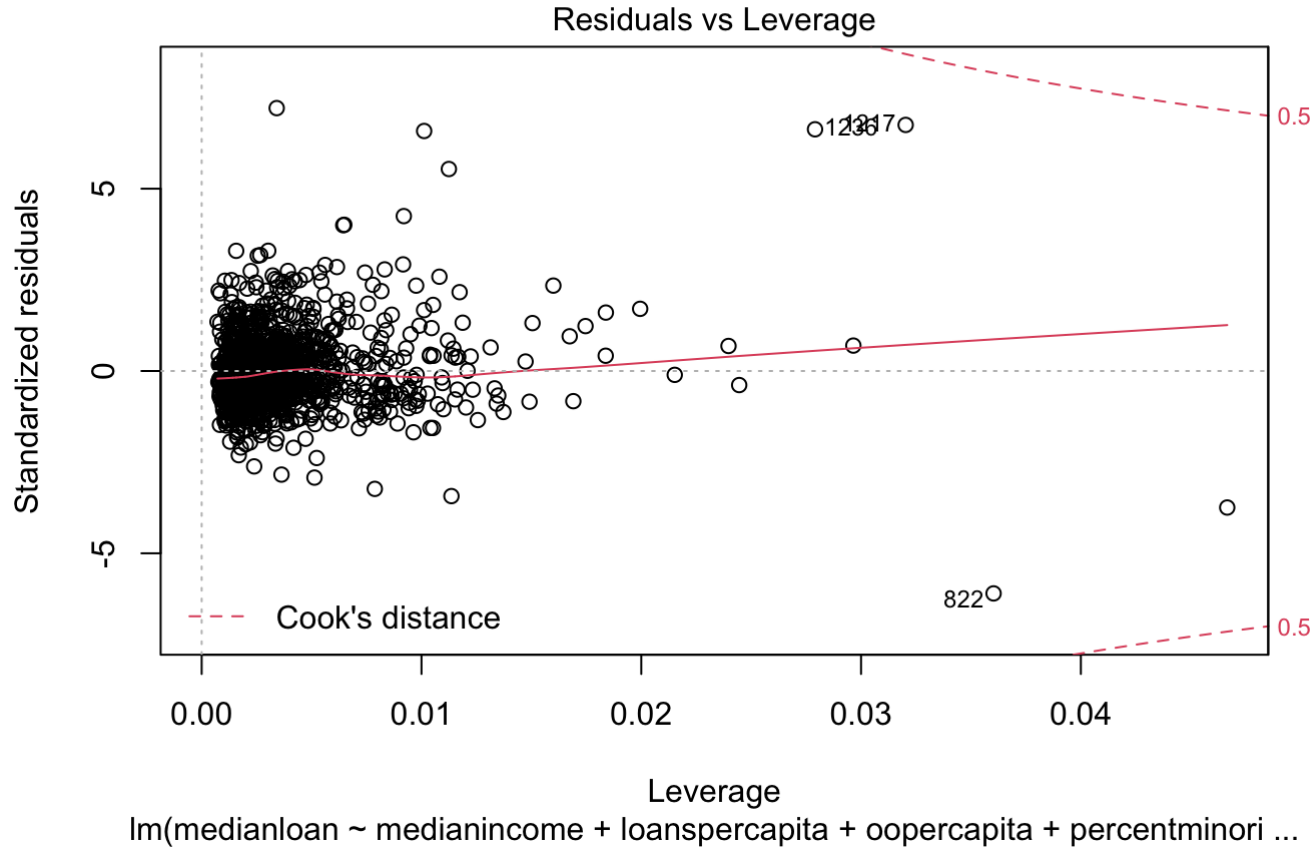
```
##
## Call:
## lm(formula = medianloan ~ medianincome + loanspercapita + oopercapita +
##     percentminority)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -354.48  -36.52   -7.72   26.49  426.03
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       51.1945     8.9252   5.736 1.18e-08 ***
## medianincome       3.4417     0.0648  53.109  < 2e-16 ***
## loanspercapita     7.2619    69.8639   0.104 0.917229
## oopercapita     -267.6338    28.3579  -9.438  < 2e-16 ***
## percentminority    0.3810     0.1096   3.475 0.000525 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 59.19 on 1428 degrees of freedom
## Multiple R-squared:  0.7036, Adjusted R-squared:  0.7028
## F-statistic: 847.6 on 4 and 1428 DF,  p-value: < 2.2e-16
```

```
plot(model3)
```


Residuals vs Fitted
Residuals
Fitted values
lm(medianloan ~ medianincome + loanspercapita + oopercapita + percentminori ...
1191
216
1217
400
200
0
-200
-400
0
200
400
600
800


Normal Q-Q
Standardized residuals
Theoretical Quantiles
lm(medianloan ~ medianincome + loanspercapita + oopercapita + percentminori ...
216
1217
8
6
4
2
0
-2
-4
-6
-3
-2
-1
0
1
2
3



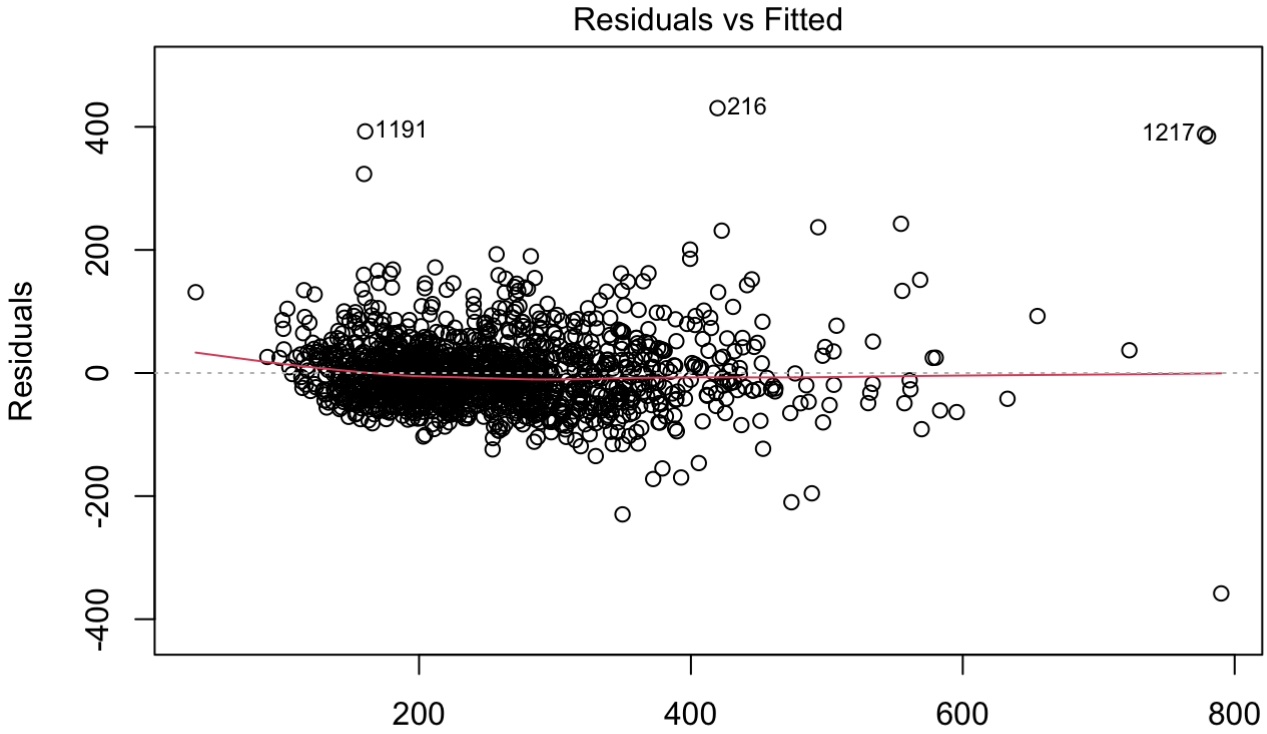


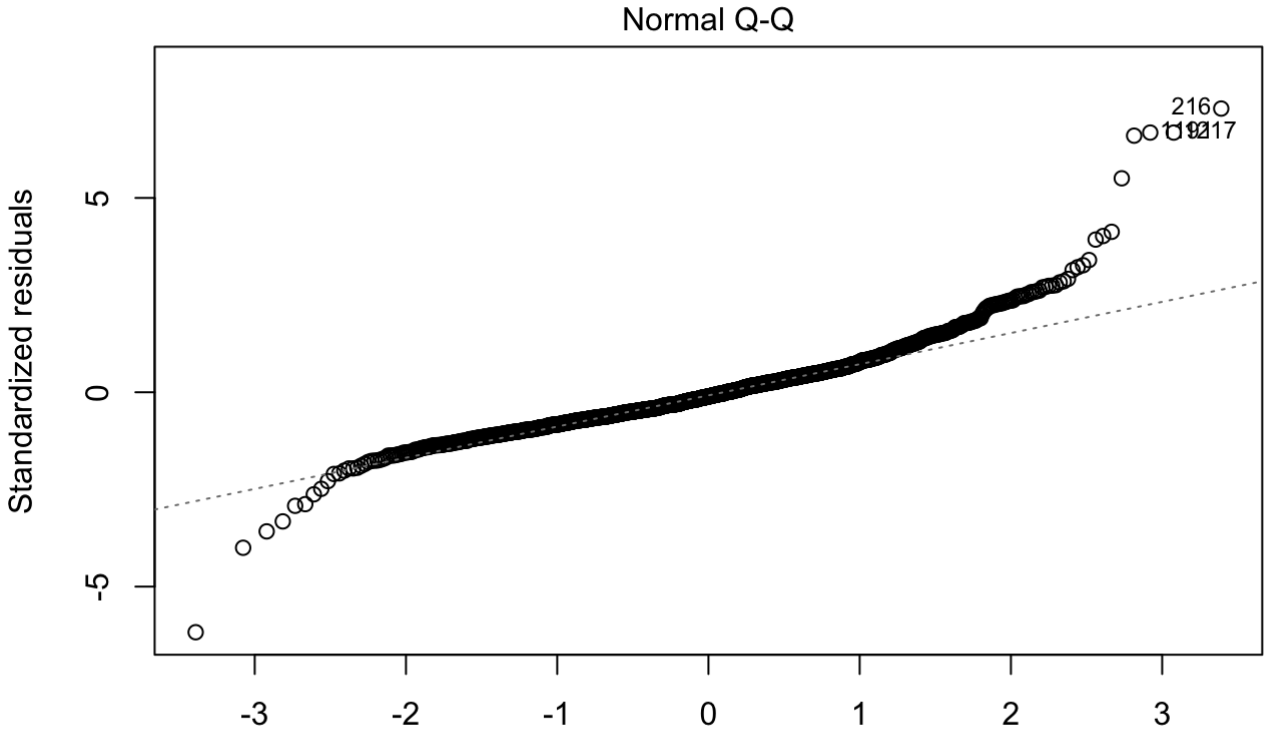
Let us add in one more variable and try repeating our process again:

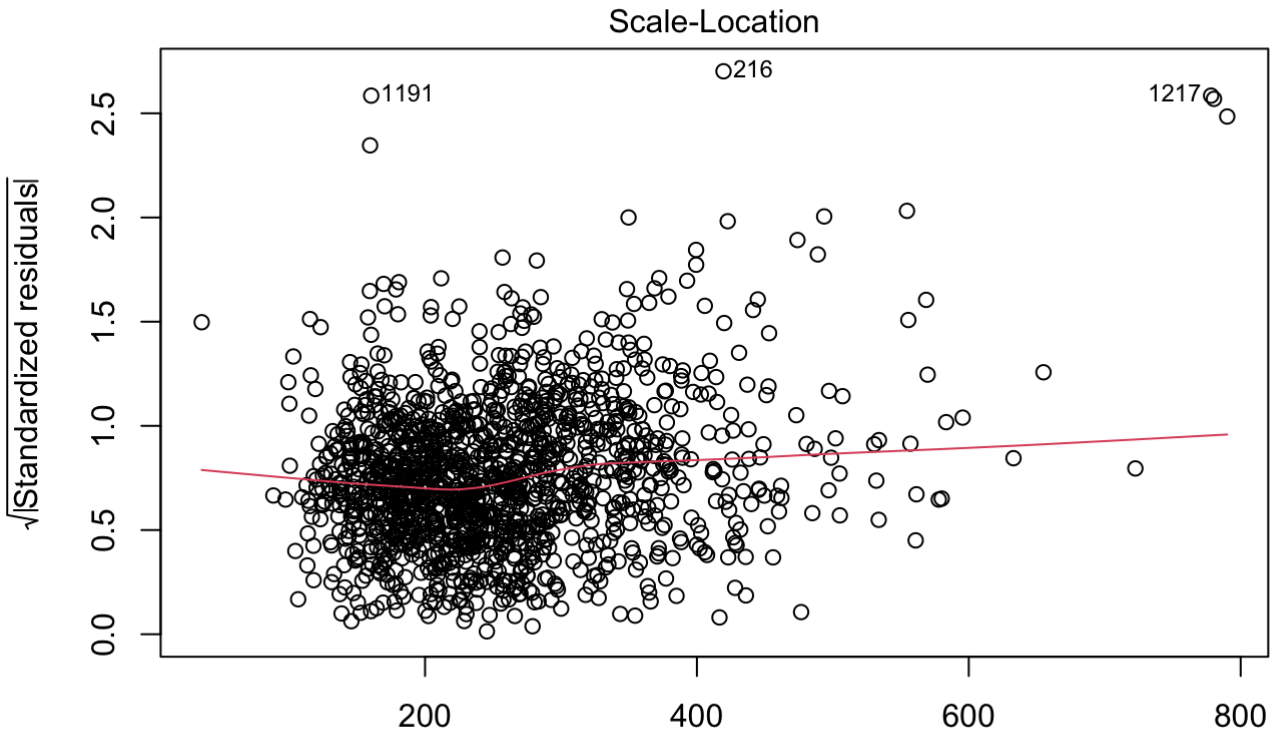
```
model4<-lm(medianloan~medianincome+loanspercapita+oopercapita+percentminority+populatio
n)
summary(model4)
```

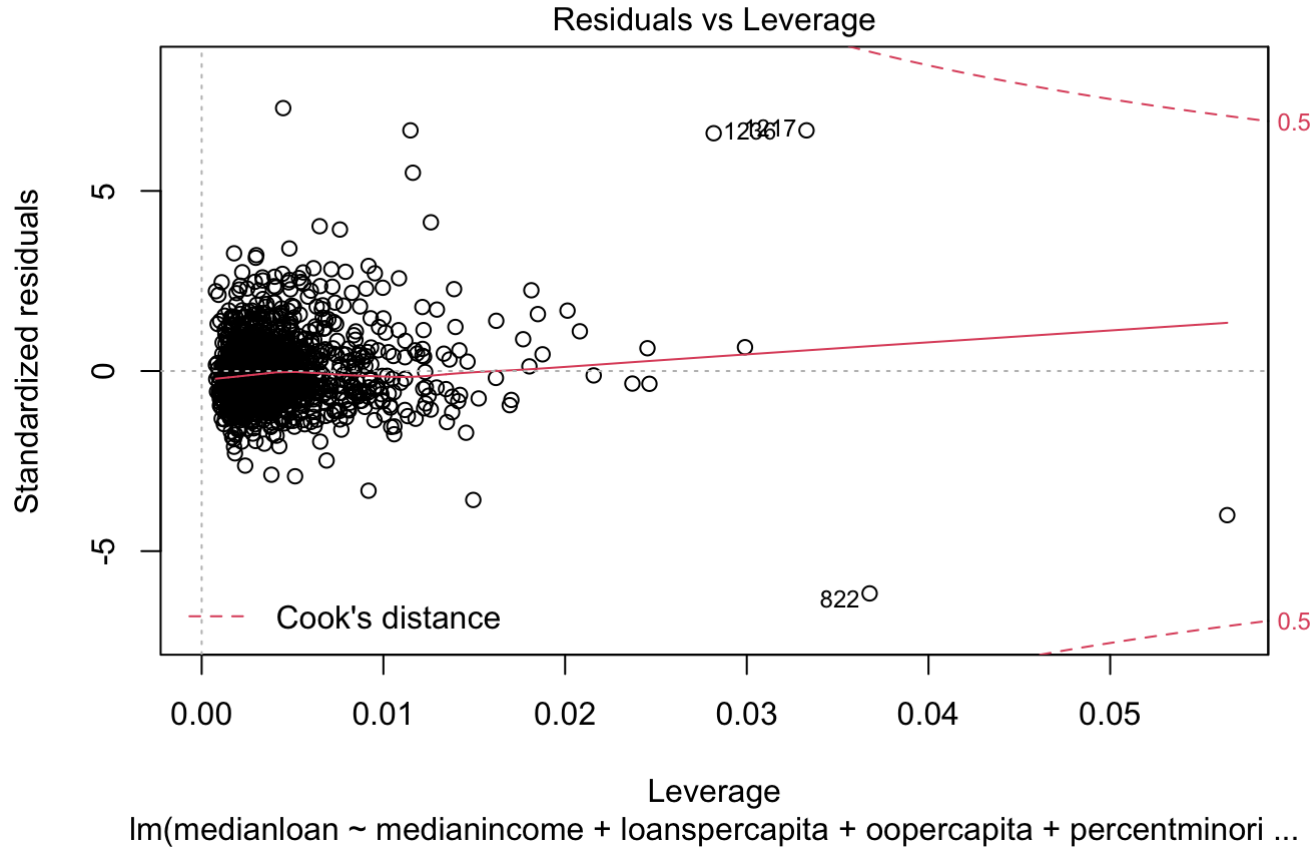
```
##
## Call:
## lm(formula = medianloan ~ medianincome + loanspercapita + oopercapita +
##     percentminority + population)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -358.11  -36.68   -6.43   27.17  430.43
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      5.985e+01  9.693e+00   6.175  8.6e-10 ***
## medianincome     3.455e+00  6.498e-02  53.174  < 2e-16 ***
## loanspercapita   2.675e+01  7.029e+01   0.381 0.703584
## oopercapita     -2.746e+02  2.848e+01  -9.641  < 2e-16 ***
## percentminority  4.152e-01  1.105e-01   3.758 0.000178 ***
## population      -2.177e-03  9.581e-04  -2.272 0.023215 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 59.1 on 1427 degrees of freedom
## Multiple R-squared:  0.7047, Adjusted R-squared:  0.7037
## F-statistic: 681.1 on 5 and 1427 DF,  p-value: < 2.2e-16
```

```
plot(model4)
```


Residuals vs Fitted
Residuals
Fitted values
lm(medianloan ~ medianincome + loanspercapita + oopercapita + percentminori ...
1191
216
1217
400
200
0
-200
-400
200
400
600
800


Normal Q-Q
Standardized residuals
Theoretical Quantiles
lm(medianloan ~ medianincome + loanspercapita + oopercapita + percentminori ...
216
1191
1217
5
0
-5
-3
-2
-1
0
1
2
3


Scale-Location
1191
216
1217
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
200
400
600
800
Fitted values
lm(medianloan ~ medianincome + loanspercapita + oopercapita + percentminori ...


Residuals vs Leverage
1217
822
Cook's distance
0.5
0.5
Standardized residuals
-5
0
5
0.00
0.01
0.02
0.03
0.04
0.05
Leverage
lm(medianloan ~ medianincome + loanspercapita + oopercapita + percentminori ...

With the changes we made we have allowed our dataset to have the best fit possible, so we will no longer be making changes to its form