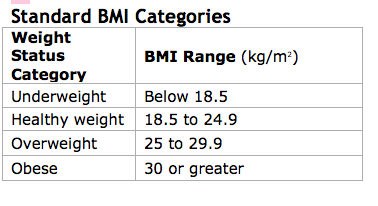
|  |  |  |  |
| --- | --- | --- | --- |
| **PM592: Regression Analysis for Health Data Science** |  |  |  |
| **Lab 3 – Visualizations & Functions**  **Data Needed:** *chs\_merged (week 2), no2.sas7bdat* | | | |

**Outline**

* Variable Manipulation
* Functions
* Ggplot2
* Correlation
* Regression

1. **Variable Manipulation: Forming Groups**
   1. **Cut**
      1. Previously we had taken certain variables and tried to group them into categories. We accomplished this using either the “if\_else” or “case\_when” command. The “cut” command takes a vector and categorizes it into categories using the specified cutpoints.
      2. Suppose we want to take the “chs\_merged” data from last lab and cut the BMI variable into cutpoints using the specified criteria:



* + 1. Note the behavior of the following options: “right,” “labels,” “ordered\_result”. In the following example, we create 4 different versions of a BMI cutpoint variable. The third version adds labels to the factor variable, and the fourth version specifies that the variable is an ordinal factor. By specifying that an ordered factor is ordinal, several functions in R will behave accordingly.

> # Create BMI cutpoints.

> chs\_merged <-

+ chs\_merged %>%

+ mutate(

+ bmicat1 = cut(

+ bmi,

+ breaks = c(0, 18.5, 25, 30, 100)

+ ),

+ bmicat2 = cut(

+ bmi,

+ breaks = c(0, 18.5, 25, 30, 100),

+ right = F

+ ),

+ bmicat3 = cut(

+ bmi,

+ breaks = c(0, 18.5, 25, 30, 100),

+ right = F,

+ labels = c("Underweight", "Healthy Weight", "Overweight", "Obese")

+ ),

+ bmicat4 = cut(

+ bmi,

+ breaks = c(0, 18.5, 25, 30, 100),

+ right = F,

+ labels = c("Underweight", "Healthy Weight", "Overweight", "Obese"),

+ ordered\_result = T

+ )

+ )

> chs\_merged %>%

+ select(starts\_with("bmicat"))

# A tibble: 1,200 x 4

bmicat1 bmicat2 bmicat3 bmicat4

<fct> <fct> <fct> <ord>

1 (0,18.5] [0,18.5) Underweight Underweight

2 (0,18.5] [0,18.5) Underweight Underweight

3 (30,100] [30,100) Obese Obese

4 (0,18.5] [0,18.5) Underweight Underweight

5 (0,18.5] [0,18.5) Underweight Underweight

6 NA NA NA NA

7 (0,18.5] [0,18.5) Underweight Underweight

8 (0,18.5] [0,18.5) Underweight Underweight

9 NA NA NA NA

10 (0,18.5] [0,18.5) Underweight Underweight

# ... with 1,190 more rows

* Which version of the variable would you keep and why?
  1. **Quantile** 
     1. Sometimes we may want to categorize a variable by quantile instead of by pre-specified cutpoints.
     2. The “ntile” function in R will cut your specified variable into n quantiles.
     3. In the following code, we create a 4-quantile and a 5-quantile variable for FEV, and then we look at the minimum, maximum, and mean FEV values within each quantile.

> chs\_merged %>%

+ group\_by(fev4c) %>%

+ summarise(

+ min = min(fev),

+ max = max(fev),

+ mean = mean(fev),

+ n = n()

+ )

`summarise()` ungrouping output (override with `.groups` argument)

# A tibble: 5 x 5

fev4c min max mean n

<int> <dbl> <dbl> <dbl> <int>

1 1 985. 1809. 1627. 277

2 2 1809. 2023. 1913. 276

3 3 2025. 2250. 2132. 276

4 4 2250 3324. 2454. 276

5 NA NA NA NA 95

>

> chs\_merged %>%

+ group\_by(fev5c) %>%

+ summarise(

+ min = min(fev),

+ max = max(fev),

+ mean = mean(fev),

+ n = n()

+ )

`summarise()` ungrouping output (override with `.groups` argument)

# A tibble: 6 x 5

fev5c min max mean n

<int> <dbl> <dbl> <dbl> <int>

1 1 985. 1752. 1588. 221

2 2 1752 1933. 1845. 221

3 3 1933. 2105. 2022. 221

4 4 2109. 2296. 2201. 221

5 5 2297. 3324. 2500. 221

6 NA NA NA NA 95

1. **Functions**
   1. So far we have been using built-in functions within R. You may find that there are some operations you frequently perform that you may like to write your own functions for.
   2. R will automatically return whichever variable is on the last line of the body of the function. However you can also return different values by explicitly specifying “return()”.
   3. A function usually takes in one or more arguments, but this is not necessary:

> hello.world <- function() {

+ print("Hello world!")

+ }

> hello.world()

[1] "Hello world!"

* 1. Here is a function that converts temperature Fahrenheit to Celsius:

> # Second function: F to C

> f.to.c <- function(temp\_F) {

+ temp\_C <- (temp\_F - 32) \* 5 / 9

+ return(temp\_C)

+ }

> f.to.c(32)

[1] 0

> f.to.c(100)

[1] 37.77778

* 1. You can create a function that modifies the behavior of existing functions. The following user-created function “mean0” will automatically remove NA values without explicitly having to state that.

> mean0 <- function(x) {

+ mean(x, na.rm=T)

+ }

> mean(chs\_merged$fev)

[1] NA

> mean0(chs\_merged$fev)

[1] 2031.265

* 1. Functions can take in more than one argument as well:

> calc.bmi <- function(ht\_inches, wt\_pounds) {

+ bmi <- 703\*wt\_pounds/ht\_inches^2

+ return(bmi)

+ }

> calc.bmi(62, 130)

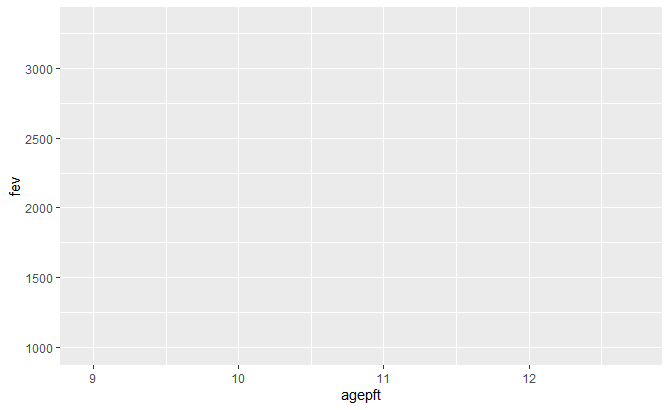
[1] 23.77471

* 1. Functions can be quite complex. In the example in 1.1. we created BMI categories using the category definitions for adults. However since our sample is on adolescents, we should be using CDC-published BMI percentiles to study BMI. We could create these variables on our own, or we could use a user-defined function for this purpose. The following link is to a function that returns a BMI percentile for a given weight (kg), height (cm), age, and sex. Note how complex these functions can be.  
     <https://rdrr.io/cran/PAutilities/src/R/get_BMI_percentile.R>

1. **Ggplot2**
   1. R has the capability to produce adequate figures. However, the “ggplot2” package can be used to create stunning visuals. Note that there is a *lot* to explore in ggplot, but we will go over some of the basic functions and syntax here. More in-depth guides are available online (<http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-R-Code.html>).
   2. The basics: what you need to include
      1. The name of your data set (obviously).
      2. An **aesthetic**. This refers to what you want to graph and how you want it to look. Depending on the type of graph, you may want to have the following aesthetics:
         1. x and y: which variables define your x and y axes
         2. Fill: a variable that will define different colors, patterns, or sizes for the “fill”
         3. Color: a variable that will define different colors, patterns, or sizes for the “color”
         4. Group: a variable that will define groups for stratified analyses
      3. A **geom**. This refers to how you want to graph your data (i.e., what will appear on the graph). We have already seen some common geoms such as:
         1. Geom\_histogram: histogram
         2. Geom\_density: a density histogram
         3. Geom\_point: scatterplot of points
         4. Geom\_boxplot: boxplots
         5. Geom\_smooth: a line
      4. Optionally you can specify more to enhance your graph. This includes things such as labels, titles, themes, ways to change your axes, annotations on your graph, etc.
   3. Setting up the aesthetic.
      1. Suppose we want to examine how FEV changes with increasing age. In the following code note that we have set the aesthetic (and our axes are set up accordingly). However we did not specify how we wanted these two variables graphed (Do we want a scatter point? Do we want a line?).

chs\_merged %>%

ggplot(aes(x = agepft, y = fev))



* + 1. We will now tell ggplot that we want to add points and a least-squares line.

> chs\_merged %>%

+ ggplot(aes(x = agepft, y = fev)) +

+ geom\_point() +

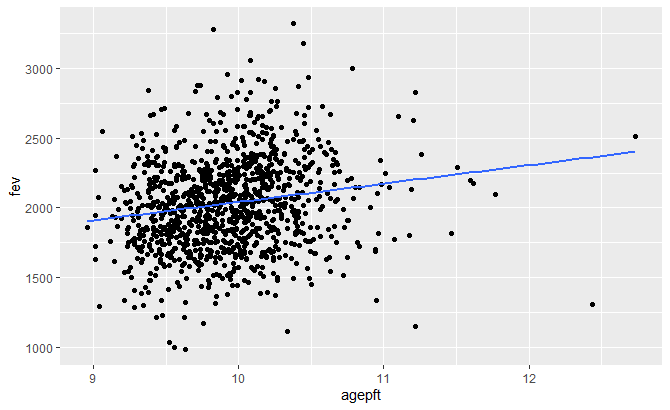
+ geom\_smooth(se = F, method = "lm")

`geom\_smooth()` using formula 'y ~ x'

Warning messages:

1: Removed 95 rows containing non-finite values (stat\_smooth).

2: Removed 95 rows containing missing values (geom\_point).



* + 1. Let’s add some options. We’ll make the points green and larger, and the line dashed.

> chs\_merged %>%

+ ggplot(aes(x = agepft, y = fev)) +

+ geom\_point(size = 2, color = "darkgreen") +

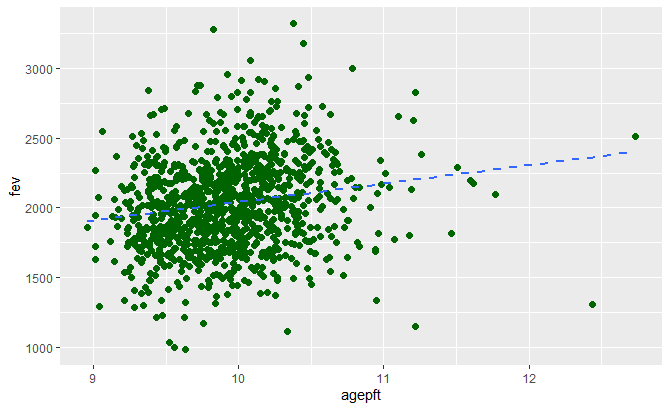
+ geom\_smooth(se = F, method = "lm", linetype = "dashed")

`geom\_smooth()` using formula 'y ~ x'

Warning messages:

1: Removed 95 rows containing non-finite values (stat\_smooth).

2: Removed 95 rows containing missing values (geom\_point).



* + 1. Finally, we’ll color the points and line by “BMI” group. This will allow us to see how FEV relates to age within each of these BMI categories. This is interesting – it tells us that FEV increases with age for underweight and healthy weight individuals, but not for overweight and obese individuals.

> chs\_merged %>%

+ ggplot(aes(x = agepft, y = fev, color = bmicat4)) +

+ geom\_point() +

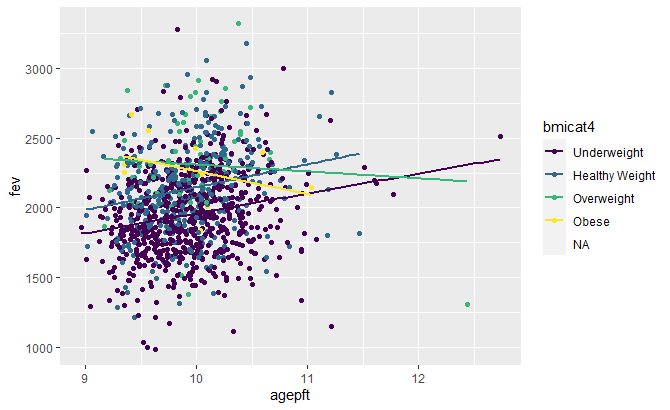
+ geom\_smooth(se = F, method = "lm")

`geom\_smooth()` using formula 'y ~ x'

Warning messages:

1: Removed 95 rows containing non-finite values (stat\_smooth).

2: Removed 95 rows containing missing values (geom\_point).



* What happens if you use the “bmicat3” variable for color instead of “bmicat4”? Why do you think the graph changed?
  + 1. One of the great things about ggplot is that it recognizes what kind of variables you are using and draws its graphs accordingly. In 3.3.4 we colored variables using a categorical variable, so it created a discrete color coding system for us. Here, we color the points by BMI, a continuous variable, so it will create a continuous color coding scheme.

> chs\_merged %>%

+ ggplot(aes(x = agepft, y = fev, color = bmi)) +

+ geom\_point() +

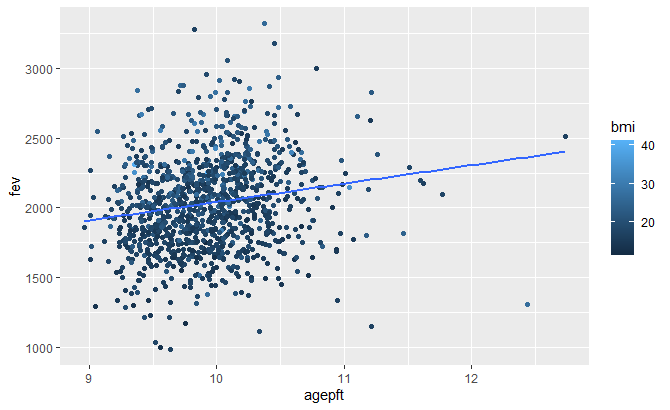
+ geom\_smooth(se = F, method = "lm")

`geom\_smooth()` using formula 'y ~ x'

Warning messages:

1: Removed 95 rows containing non-finite values (stat\_smooth).

2: Removed 95 rows containing missing values (geom\_point).



1. **Correlation**
   1. We can use the cor() function to arrive at correlation coefficients for variables. By default, if any data is missing then “NA” will be reported. Some common ways to fix this is to use only complete observations (use = “complete.obs”) or to use only pairwise complete observations (use = “pairwise.complete.obs”).

> chs\_merged %>%

+ select(agepft, fev, fvc, mmef) %>%

+ cor()

agepft fev fvc mmef

agepft 1 NA NA NA

fev NA 1 NA NA

fvc NA NA 1 NA

mmef NA NA NA 1

> chs\_merged %>%

+ select(agepft, fev, fvc, mmef) %>%

+ cor(use = "pairwise.complete.obs")

agepft fev fvc mmef

agepft 1.00000000 0.1714370 0.1857674 0.08282617

fev 0.17143699 1.0000000 0.9164952 0.63816935

fvc 0.18576738 0.9164952 1.0000000 0.33811832

mmef 0.08282617 0.6381694 0.3381183 1.00000000

> chs\_merged %>%

+ select(agepft, fev, fvc, mmef) %>%

+ cor(use = "complete.obs")

agepft fev fvc mmef

agepft 1.00000000 0.1731801 0.1937429 0.07987482

fev 0.17318011 1.0000000 0.9180125 0.63908425

fvc 0.19374289 0.9180125 1.0000000 0.33822368

mmef 0.07987482 0.6390843 0.3382237 1.00000000

* 1. The cor.test() function can be used to find p-values for two variables.

> cor.test(chs\_merged$fev, chs\_merged$fvc)

Pearson's product-moment correlation

data: chs\_merged$fev and chs\_merged$fvc

t = 75.914, df = 1098, p-value < 2.2e-16

alternative hypothesis: true correlation is not equal to 0

95 percent confidence interval:

0.9064941 0.9254683

sample estimates:

cor

0.9164952

* 1. Other packages, such as Hmisc, allow for the calculation of p-values on the entire correlation matrix. You can specify in the rcorr() function whether you want Pearson’s or Spearman’s correlation.

> chs\_merged %>%

+ select(agepft, fev, fvc, mmef) %>%

+ as.matrix() %>%

+ rcorr(type = "pearson")

agepft fev fvc mmef

agepft 1.00 0.17 0.19 0.08

fev 0.17 1.00 0.92 0.64

fvc 0.19 0.92 1.00 0.34

mmef 0.08 0.64 0.34 1.00

n

agepft fev fvc mmef

agepft 1111 1105 1103 1094

fev 1105 1105 1100 1093

fvc 1103 1100 1103 1090

mmef 1094 1093 1090 1094

P

agepft fev fvc mmef

agepft 0.0000 0.0000 0.0061

fev 0.0000 0.0000 0.0000

fvc 0.0000 0.0000 0.0000

mmef 0.0061 0.0000 0.0000

* 1. The package “ggcorr” interfaces with ggplot to produce visualizations of correlations (correllograms). The following code produces the default correlogram, and then demonstrates how to change the visual style. Other options, such as adding the p-values, are also available. We clearly see that FEV and FVC are the most highly related of these four variables.

> chs\_merged %>%

+ select(agepft, fev, fvc, mmef) %>%

+ cor(use = "pairwise.complete.obs") %>%

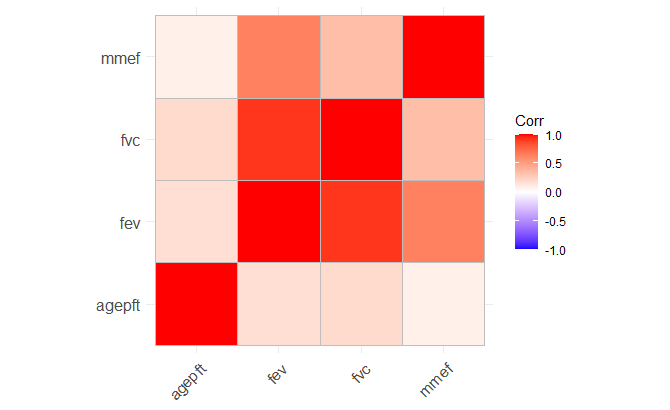
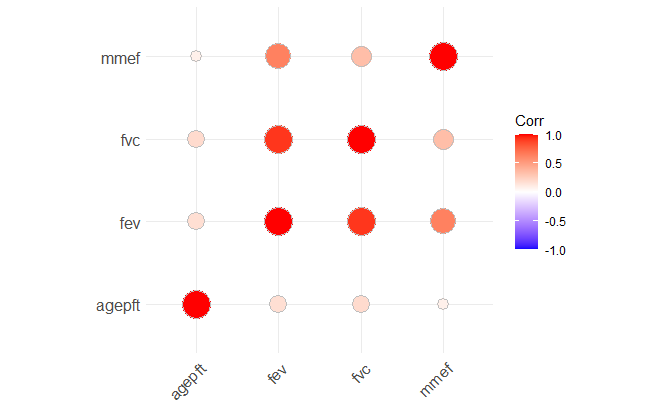
+ ggcorrplot()

> chs\_merged %>%

+ select(agepft, fev, fvc, mmef) %>%

+ cor(use = "pairwise.complete.obs") %>%

+ ggcorrplot(method = "circle")

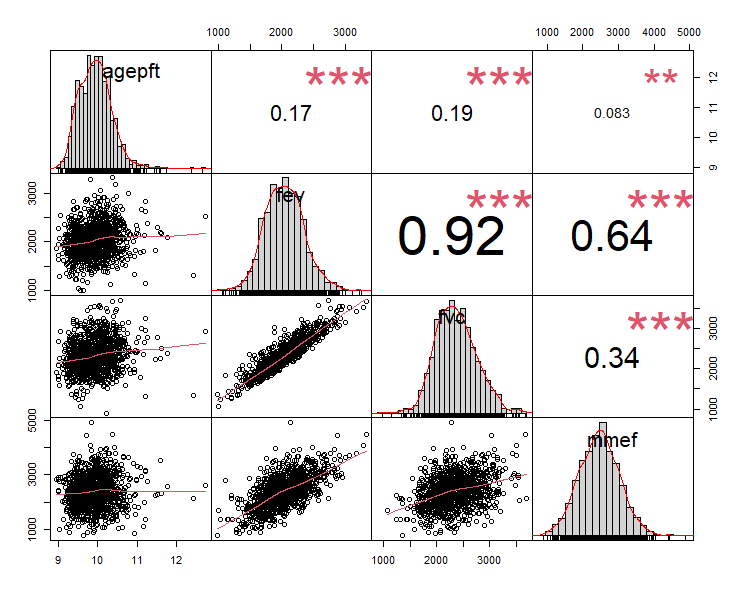
 

* 1. Other packages, such as PerformanceAnalytics, offer alternate ways of viewing the data. You can also specify Pearson’s or Spearman’s correlation in the function (though Pearson’s is the default).

> chs\_merged %>%

+ select(agepft, fev, fvc, mmef) %>%

+ chart.Correlation(histogram=TRUE, pch=19)



1. **Single Linear Regression**
   1. **Example**
      1. We previously saw that FEV tends to increase with age. Let’s run a linear regression to examine the nature of this relationship.
      2. Remember that we want to examine our data visually first, to make sure the assumptions of linear regression are met.

> model1 <-

+ lm(fev ~ agepft, data = chs\_merged)

> model1 %>%

+ summary()

Call:

lm(formula = fev ~ agepft, data = chs\_merged)

Residuals:

Min 1Q Median 3Q Max

-1059.69 -220.96 -5.63 211.01 1265.43

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 721.41 226.86 3.180 0.00151 \*\*

agepft 131.98 22.84 5.779 9.76e-09 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 325.9 on 1103 degrees of freedom

(95 observations deleted due to missingness)

Multiple R-squared: 0.02939, Adjusted R-squared: 0.02851

F-statistic: 33.4 on 1 and 1103 DF, p-value: 9.758e-09

* + 1. What can we tell from this output?
* The predicted FEV when agepft=0 is 721.41. This is statistically significantly different from 0 (p=.00151)   
  (Is this meaningful? How would we make it more meaningful?)
* When agepft increases by 1 (1 year), FEV is expected to increase by 131.98 (p<.001).
* Our best-fit line is:
  1. **Statistical vs. Practical Effects**
     1. Statistical significance specifically refers to the hypotheses being tested. In this case, we determine that, on average, FEV increases as age increases. A statistically significant effect, though, does not need to be a large effect. The standard error and sample size are both related to the p-value.
     2. Practical significance refers to how large the effect size is. For example you may find that a new therapy reduces the risk of disease by 0.0001%, but the new therapy is expensive and time-consuming. Even if you find that p<.05, the effect size may be so small that it is not practical to implement such a program.
     3. A standardized coefficient essentially refers to the relationship between the z-scores of two variables. As the variables are converted to z-scores, this gives us a standardized scale on which to measure the effect size. To make it easy, we can standardize directly within the lm() function by specifying scale().   
        Note: for single linear regression the standardized coefficient is equivalent to R.  
        Note: the standardized coefficient is equivalent to
     4. The interpretation of a standardized coefficient is “a one standard deviation increase in X is associated with a standard deviation increase in Y.”

> model1\_std <-

+ lm(scale(fev) ~ scale(agepft), data = chs\_merged)

> model1\_std %>%

+ summary()

Call:

lm(formula = scale(fev) ~ scale(agepft), data = chs\_merged)

Residuals:

Min 1Q Median 3Q Max

-3.2047 -0.6682 -0.0170 0.6381 3.8269

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -0.0002698 0.0296509 -0.009 0.993

scale(agepft) 0.1713223 0.0296445 5.779 9.76e-09 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.9856 on 1103 degrees of freedom

(95 observations deleted due to missingness)

Multiple R-squared: 0.02939, Adjusted R-squared: 0.02851

F-statistic: 33.4 on 1 and 1103 DF, p-value: 9.758e-09

> 131.98/(22.84\*sqrt(1104))

[1] 0.1739112

* + 1. Another way to assess the practical effect is through the multiple R2. This refers to the percent of the variance in the outcome that is explained by your predictors. Here, 2.9% of the variation in FEV is explained by the linear relationship with age. (This means that 97.1% of the variation in FEV is explained by other things.) The practical interpretation of R2 varies by discipline; for example, in psychology and health behavior this value is typically low, but in “hard” sciences the expectation may be that R2 is higher.
  1. **Confidence Intervals and Prediction Intervals**
     1. Recall that ggplot automatically provides 95% confidence intervals unless specified otherwise. The confidence interval tells us about our certainty for the mean value.
     2. We can also obtain the 95% prediction intervals. The prediction interval tells us about our certainty for individual predictions.

> chs\_merged2 <- cbind(

+ chs\_merged,

+ predict(model1, chs\_merged, interval="prediction")

+ )

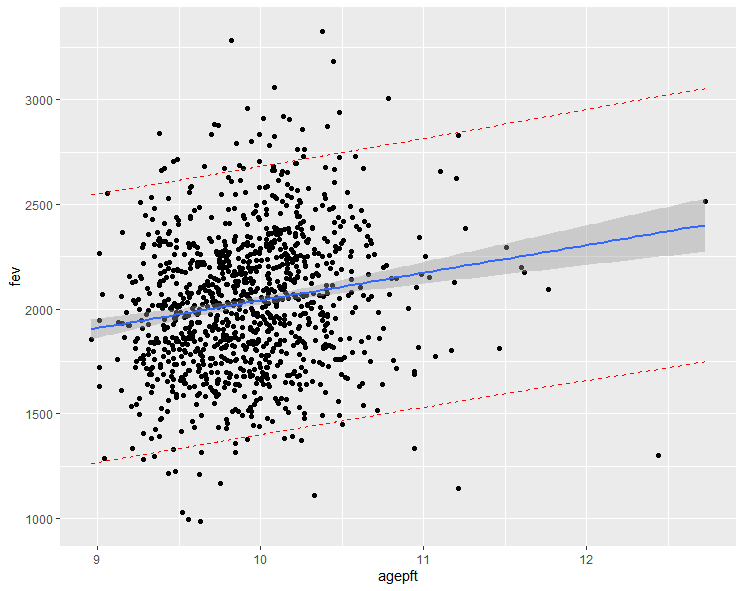
> chs\_merged2 %>%

+ ggplot(aes(x = agepft, y = fev))+

+ geom\_point() +

+ geom\_line(aes(y=lwr), color = "red", linetype = "dashed")+

+ geom\_line(aes(y=upr), color = "red", linetype = "dashed")+

+ geom\_smooth(method=lm, se=TRUE) 

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Lab 3 Exercises** |  |  |  |  |  |  |

|  |  |
| --- | --- |
| Objective(s): | Merge two data sets, create factor variables, assess variable normality, practice variable manipulation |
| Datasets Required: | NO2 |

We will use the NO2 data for this lab. This data examines levels of nitrogen dioxide (NO2, in ppb) at children’s homes and the distance of the nearest freeway (km). You will be assigned a town.

1. Create a scatterplot for your town with X = distance to freeway and Y = NO2 exposure.
   1. From only this scatterplot, what do you think the NO2 exposure is for children living right next to the freeway?
2. Perform a regression on these two variables.
   1. What is your regression line?
   2. Is distance from the freeway related to NO2 exposure?
   3. In words, interpret your intercept value.
   4. In words, interpret your slope value.
   5. How much cleaner is the air 2km away from the freeway?
   6. Do your answers make sense when compared to your scatterplot?
3. Center the distance variable on its mean and re-run the regression.
   1. What is your regression line?
   2. Which parameter estimates changed and which didn’t? Why?
4. Create a new variable that is distance to the nearest freeway in miles. Perform the regression between NO2 levels and distance in miles.
   1. What is your regression line?
   2. Which parameter estimates changed and which didn’t? Why?
5. Using this data, how would you go about making a “best guess” as to what the NO2 exposure is at the Soto Street Building (SSB) at USC?
   1. Provide your prediction.

USC SSB is at the orange star below. It is approximately 1.5 km from the 10 freeway.

