Module 07 Lab 01

HDS

2024-09-02

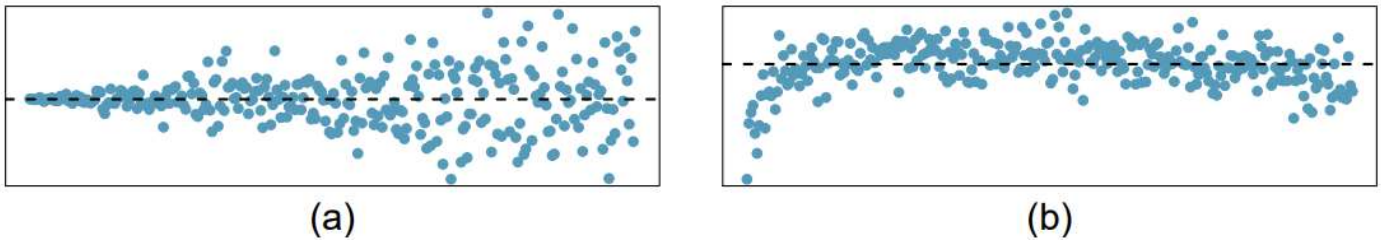
checked 01/03/2025

Module 07, Lab/Homework 01, Regression and GLM

Problems from Deiz et al, Chapter 8- you will need to look at the book to see the images.

See images in the Deiz et al pdf. Write your answers down in the RMD file, and knit it to pdf to submit

8.2 Trends in the residuals. Shown below are two plots of residuals remaining after fitting a linear model to two different sets of data. Describe important features and determine if a linear model would be appropriate for these data. Explain your reasoning.



8.2 Image

Just for my own understanding, Let's see if I can estimate the true plots. Source for linear regression: https://www.datacamp.com/tutorial/linear-regression-R (https://www.datacamp.com/tutorial/linear-regression-R)

```
library(ggplot2)
```

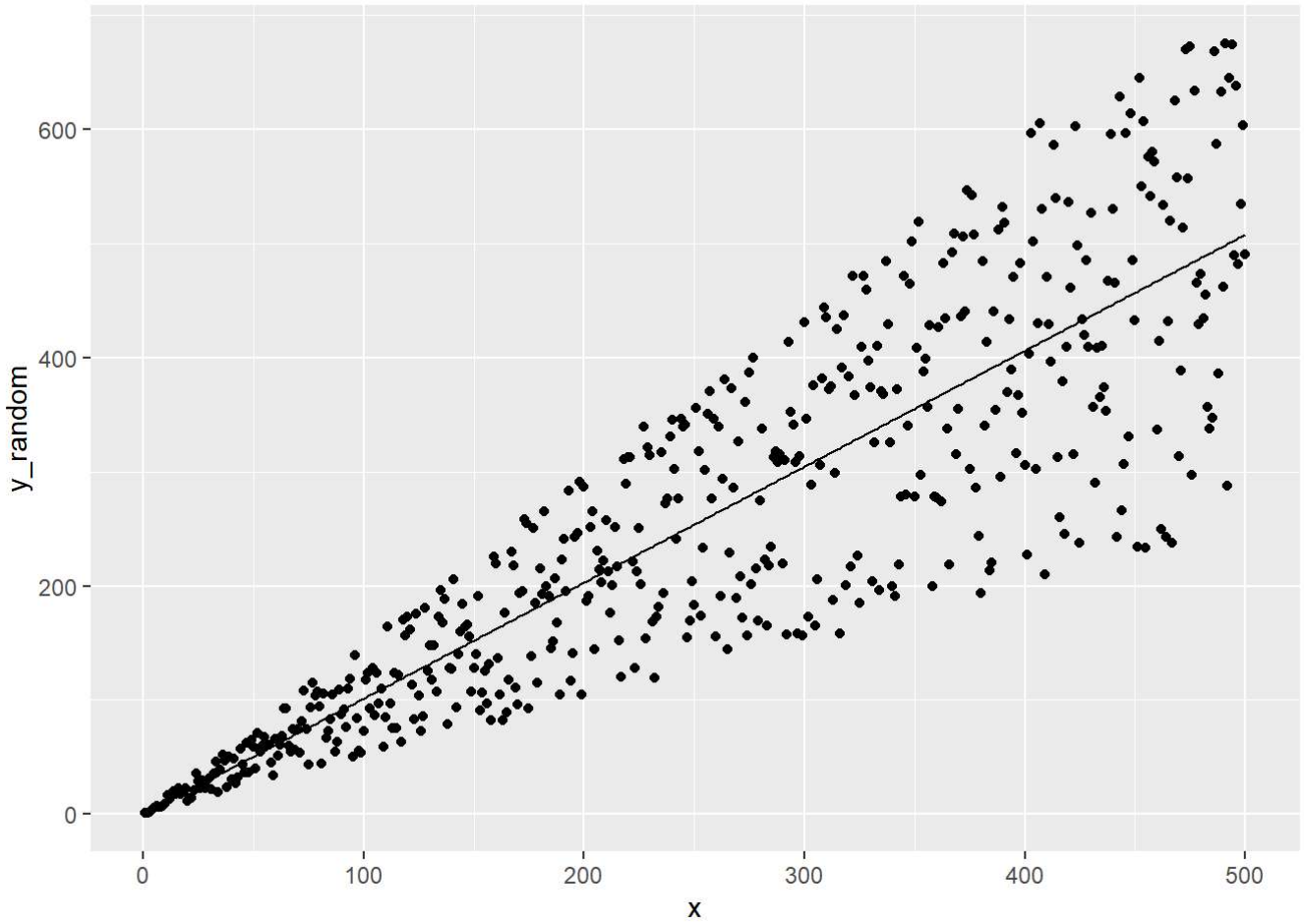
Source for assigning multiple variables at once: https://stackoverflow.com/questions/7519790/assign-multiple-new-variables-on-lhs-in-a-single-line (https://stackoverflow.com/questions/7519790/assign-multiple-new-variables-on-lhs-in-a-single-line) Source for adding vector to df: https://stackoverflow.com/questions/31298092/add-a-vector-to-a-column-of-a-dataframe (https://stackoverflow.com/questions/31298092/add-a-vector-to-a-column-of-a-dataframe)

```
library(zeallot)
```

```
#Set up a sequence
x = seq(1, 500, 1)
y_random <- vector()
y <-vector()

#Loop through the sequence of x and assign randomized y values, normalized to -0.5 through 0.5
for (i in 1:length(x)) {
  random <- runif(1, -0.5, 0.5)
  y_random[i]<-x[i]+random*x[i]
}

#Load into data frame
df <- data.frame(x=x, y_random=y_random)

#Get the intercept and slope from the linear regression
coefs <- coefficients(lm(y_random~x,df))
c(intercept, x) %<-% c(coefs[[1]], coefs[[2]])

#Extrapolate across the range of x
for (i in 1:length(df$x)) {
  y[i]<-df$x[i]*x+intercept
}

#bind the new y vector to the existing df
#Source for invisible function: https://stackoverflow.com/questions/11653127/what-does-the-funct
ion-invisible-do
invisible(rbind(df, y))
```

```
## Warning in rbind(deparse.level, ...): number of columns of result, 2, is not a
## multiple of vector length 500 of arg 2
```

```
#overlay the plot and the estimated linear model
ggplot(
  data = df,
  mapping = aes(x, y_random)
) + geom_point() + geom_line(aes(x, y))
```

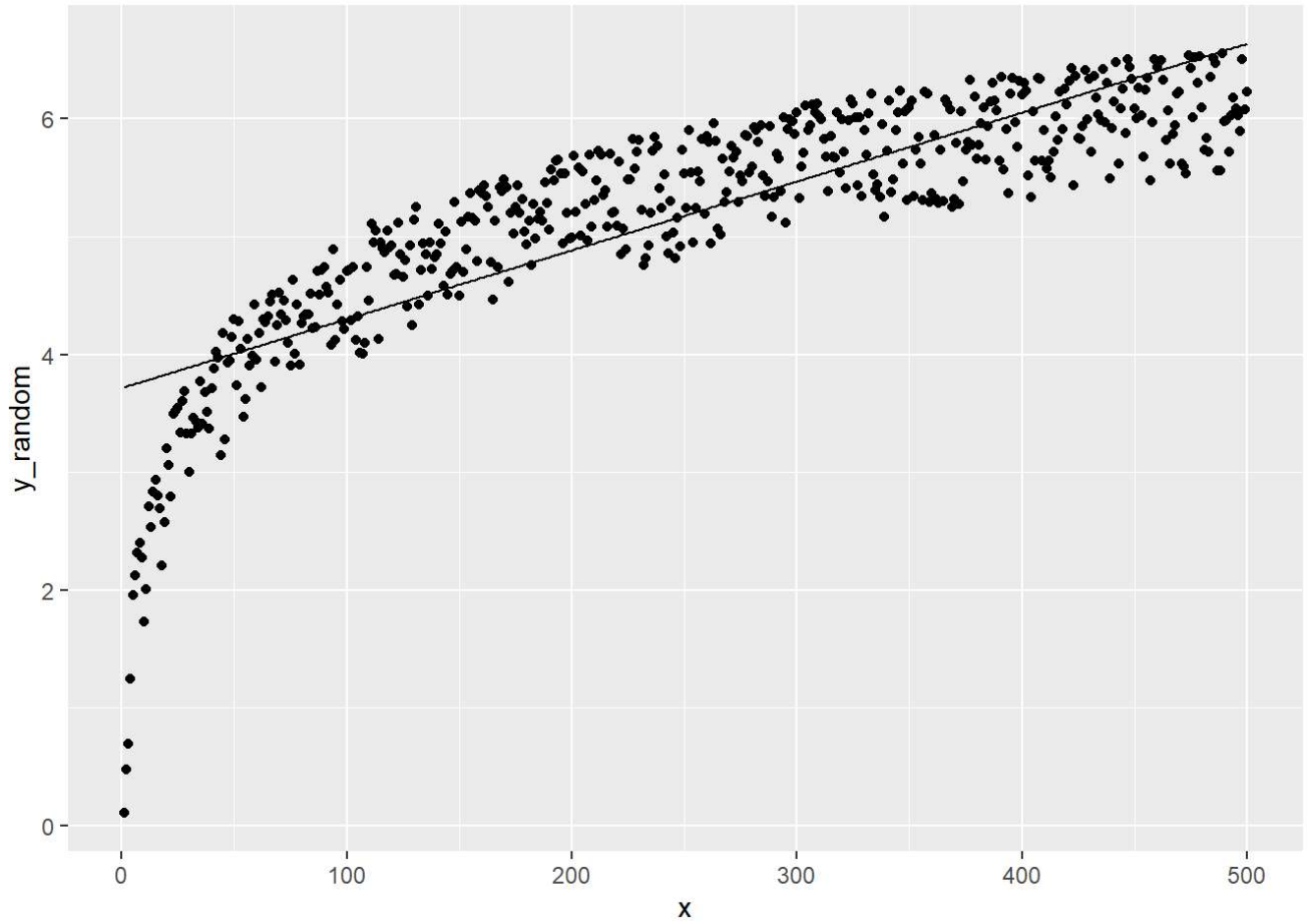


This looks like a good estimate of what the actual plot would like, and it does appear to be linear. Repeat for the second plot, which I suspect is logarithmic.

```
x = seq(1, 500, 1)
y_random <- vector()
y <- vector()

#loop through the sequence of x and assign randomized y values, normalized to -0.5 through 0.5
for (i in 1:length(x)) {
  random <- runif(1,-0.5,0.5)
  y_random[i]<-log(x[i]+random*x[i])
}

#load into data frame
df <- data.frame(x=x, y_random=y_random)

#Get the intercept and slope from the linear regression
coefs <- coefficients(lm(y_random~x,df))
c(intercept, x) %<-% c(coefs[[1]], coefs[[2]])

#Extrapolate across the range of x
for (i in 1:length(df$x)) {
  y[i]<-df$x[i]*x+intercept
}

#bind the new y vector to the existing df
invisible(rbind(df, y))
```

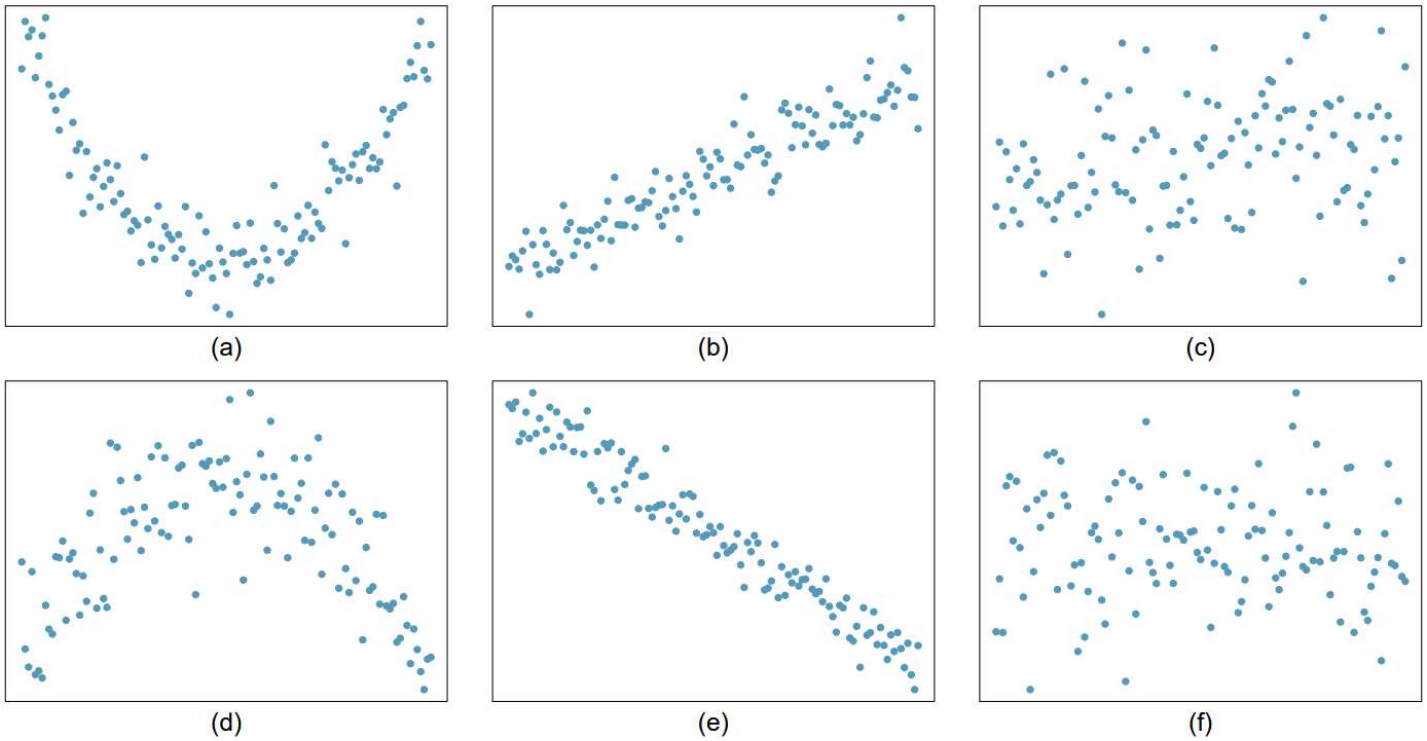
```
## Warning in rbind(deparse.level, ...): number of columns of result, 2, is not a
## multiple of vector length 500 of arg 2
```

```
#overlay the plot and the estimated linear model
ggplot(
  data = df,
  mapping = aes(x, y_random)
) + geom_point() + geom_line(aes(x, y))
```



Okay, so I am pretty confident that I have modeled each of the actual plots that the residuals were derived from. The residual plot (a) seems to be a better candidate for linear regression than (b), but the error in the estimate increases linearly with x. This is problematic as the "correctness" of the linear model will become worse as x grows. Plot (b) originates from logarithmic data, so a linear model may not be ideal as an estimate. Interestingly, a linear model would have less aggregated error over increasingly large sample sizes for plot (a) than (b), as logarithmic data tends to become more linear over x. In the case of plot (a), it is becoming less linear over x.

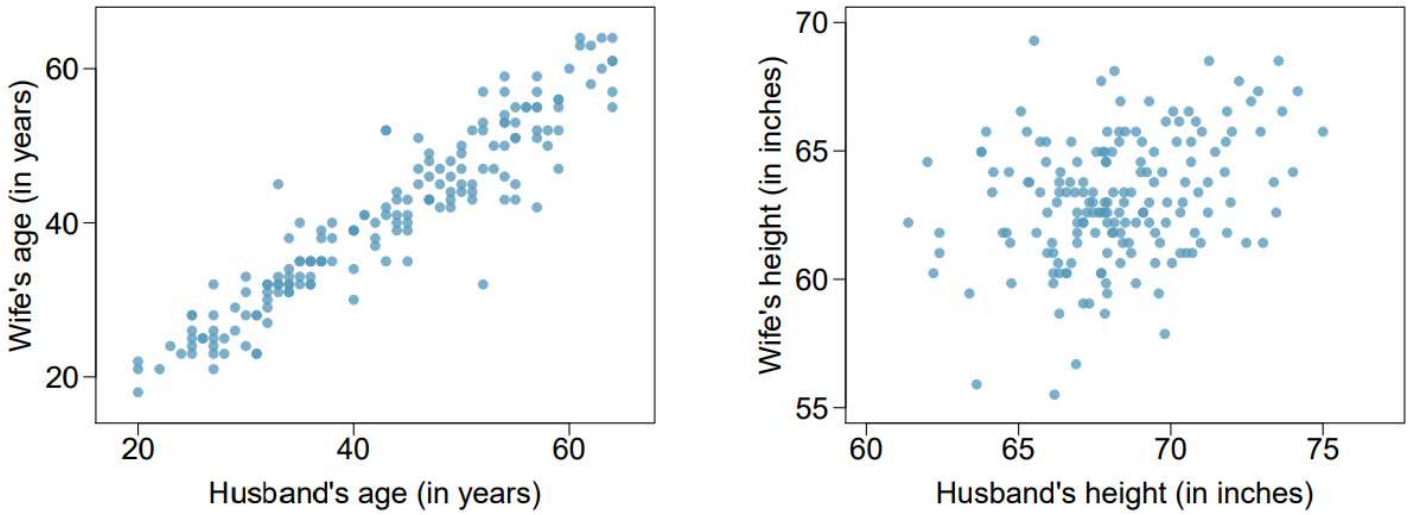
8.4 Identify relationships, Part II. For each of the six plots, identify the strength of the relationship (e.g. weak, moderate, or strong) in the data and whether fitting a linear model would be reasonable.



8.4 Image

a. Weak, not reasonable
b. Strong, reasonable
c. Moderate, reasonable
d. Weak, not reasonable
e. Strong, reasonable
f. Moderate, reasonable

8.6 Husbands and wives, Part I. The Great Britain Office of Population Census and Surveys once collected data on a random sample of 170 married couples in Britain, recording the age (in years) and heights (converted here to inches) of the husbands and wives.5 The scatterplot on the left shows the wife's age plotted against her husband's age, and the plot on the right shows wife's height plotted against husband's height.



8.6 Image

a. Describe the relationship between husbands' and wives' ages.

This is a strong, positively correlated linear relationship.

b. Describe the relationship between husbands' and wives' heights.

This is a moderate, positively correlated linear relationship.

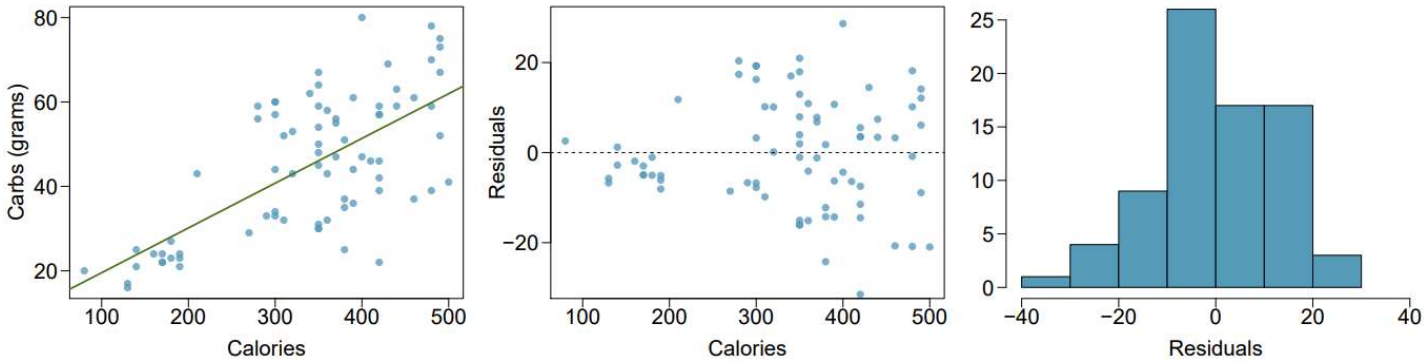
c. Which plot shows a stronger correlation? Explain your reasoning.

The age plot has a stronger correlation. The husband's age is a strong predictor of the wife's age, and we could use this data to predict the wife's ages with novel husband age data. The same cannot be said for height.

d. Data on heights were originally collected in centimeters, and then converted to inches. Does this conversion affect the correlation between husbands' and wives' heights?

No, the same conversion is applied to both data sets, so the correlation is conserved. Correlation is a relative measure.

8.22 Nutrition at Starbucks, Part I. The scatterplot below shows the relationship between the number of calories and amount of carbohydrates (in grams) Starbucks food menu items contain.15 Since Starbucks only lists the number of calories on the display items, we are interested in predicting the amount of carbs a menu item has based on its calorie content.



8.22 Image

a. Describe the relationship between number of calories and amount of carbohydrates (in grams) that Starbucks food menu items contain.

This is a moderate, positively associated linear relationship.

b. In this scenario, what are the explanatory and response variables?

The explanatory variable is amount of carbohydrate, and the calorie count is the response variable.

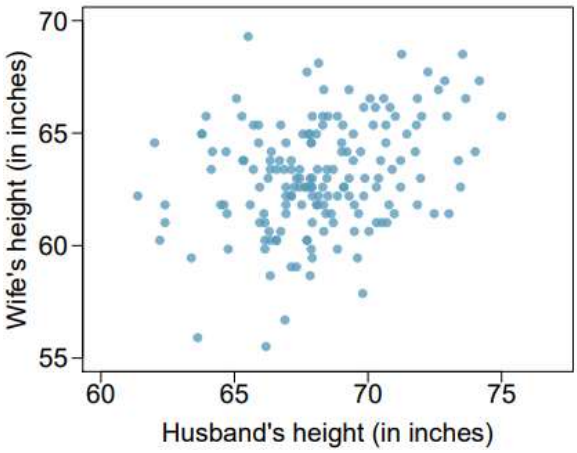
c. Why might we want to fit a regression line to these data?

This could be a predictor for other menu items that may not have a calorie count listed. If the amount of carbohydrate can be found or estimated, this value can be used to estimate the calories based on the regression line, and vice versa.

d. Do these data meet the conditions required for fitting a least squares line?

No, it appears that the variability in the error is related to x, and there are several outliers. This is likely a result of fat being more calorie dense than carbohydrates. The highly correlated items are likely coffees, where a majority of the calories come from sugar (carbs) and the amount of fat is relatively consistent. Once the higher caloric density items are reached, which are likely food items, the caloric density is more strongly influenced by fat than carbs alone.

8.33 Husbands and wives, Part II. The scatterplot below summarizes husbands' and wives' heights in a random sample of 170 married couples in Britain, where both partners' ages are below 65 years. Summary output of the least squares fit for predicting wife's height from husband's height is also provided in the table.



8.33 Image

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	43.5755	4.6842	9.30	0.0000
height_husband	0.2863	0.0686	4.17	0.0000

From the book: The first column of numbers provides estimates for b0 and b1, respectively.

a. Is there strong evidence that taller men marry taller women? State the hypotheses and include any information used to conduct the test.

H_0: The true linear model has slope zero H_A: The true linear model has a slope other than zero.

y_hat = 0.2863x + 43.5755 T = (estimate - null value)/SE = (0.2863-0)/0.0686 = 4.17

A p value of 0.0000 indicates strong evidence against the null hypothesis. In this case, we see strong evidence that taller men marry taller women.

b. Write the equation of the regression line for predicting wife's height from husband's height.

y_hat = 0.2863x + 43.5755

c. Interpret the slope and intercept in the context of the application.

The intercept of the regression line is the value of y_hat when x is equal to zero. In terms of this data set, this is the height of a wife who's husband is 0 inches tall. The slope represents how much the wife's predicted height will grow relative to growth in the husband's height. I.E., we predict the wife will be 0.2863 inches taller for every inch taller the husband is.

d. Given that R^2 = 0:09, what is the correlation of heights in this data set?

```
sqrt(0.09)
```

```
## [1] 0.3
```

The correlation is 0.3

e. You meet a married man from Britain who is 5'9" (69 inches). What would you predict his wife's height to be? How reliable is this prediction?

```
wife_h = 0.2863*(69)+43.5755
wife_h
```

```
## [1] 63.3302
```

The predicted height for the wife is 63.3 inches. Based on the correlation coefficient, t value, and p value, this prediction is moderately reliable. There is a lot of variance in the data that the model cannot explain, but the general trend of the data is moderate.

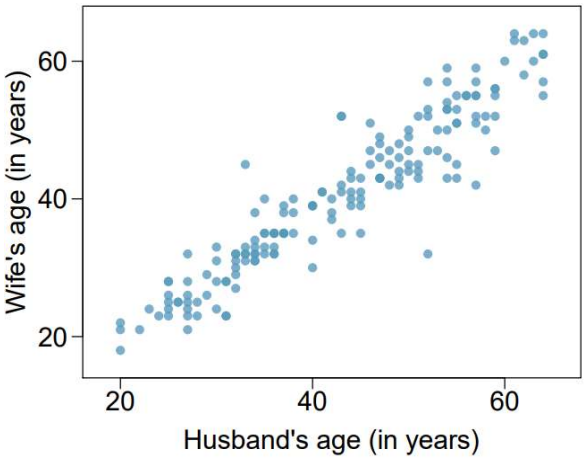
f. You meet another married man from Britain who is 6'7" (79 inches). Would it be wise to use the same linear model to predict his wife's height? Why or why not?

```
wife_h = 0.2863*(79)+43.5755
wife_h
```

```
## [1] 66.1932
```

The model may not be able to predict the wife's height, as the man's height is an outlier in the data set that the model was "trained" on. The model would predict the wife to be 66 inches tall, which seems reasonable, but ideally this prediction could be made with a model trained on a larger data set.

8.39 Husbands and wives, Part III. Exercise 8.33 presents a scatterplot displaying the relationship between husbands' and wives' ages in a random sample of 170 married couples in Britain, where both partners' ages are below 65 years. Given below is summary output of the least squares ↟t for predicting wife's age from husband's age.



8.33 Image

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.5740	1.1501	1.37	0.1730
age_husband	0.9112	0.0259	35.25	0.0000
				$df = 168$

a. We might wonder, is the age difference between husbands and wives consistent across ages? If this were the case, then the slope parameter would be β1 = 1. Use the information above to evaluate if there is strong evidence that the difference in husband and wife ages differs for different ages.

y_hat = β0 + β1 x age_husband = Intercept + slope_estimate x age_husband, therefore β1 = slope_estimate

Based on a slope estimate, or β1, of 0.9112, we can conclude that the difference between husbands and wives ages is consistent across ages. 0.9112 is near enough to 1 to make this conclusion.

b. Write the equation of the regression line for predicting wife's age from husband's age.

y_hat = 0.9112 x age_husband + 1.574

c. Interpret the slope and intercept in context.

The intercept of the regression line is the value of y_hat when x is equal to zero. In terms of this data set, this is the age of a wife who's husband is 0 years old. The slope represents how much the wife's predicted age will change relative to change in the husband's age. I.E., we predict the wife will be 0.9112 years older for every year older the husband is.

d. Given that R^2 = 0:88, what is the correlation of ages in this data set?

```
sqrt(0.88)
```

```
## [1] 0.9380832
```

The correlation coefficient is 0.938

e. You meet a married man from Britain who is 55 years old. What would you predict his wife's age to be? How reliable is this prediction?

```
age_wife = (0.9112*55)+1.574
age_wife
```

```
## [1] 51.69
```

The predicted wife's age is 51.69. This is a very reliable prediction given the R^2 value for this data set.

f. You meet another married man from Britain who is 85 years old. Would it be wise to use the same linear model to predict his wife's age? Explain.

Yes, this model should accurately predict the wife's age. There is a diverse set of ages in the data set, from 20 to about 65 or 70. The R^2 value is 0.88, indicating that 88% of the variance in the data can be explained by the model. This means that there is a strong relationship between the x and y variables with little variance, and the model should continue to be accurate with new data.

Multiple Regression Model

We will work from the butterfat data set from the library GLMsData

Use google and look up this library and data set

```
library(GLMsData)
data(butterfat)
head(butterfat)
```

```
##   Butterfat    Breed    Age
## 1      3.74 Ayrshire Mature
## 2      4.01 Ayrshire  2year
## 3      3.77 Ayrshire Mature
## 4      3.78 Ayrshire  2year
## 5      4.10 Ayrshire Mature
## 6      4.06 Ayrshire  2year
```

Build a GLM that predicts the Butterfat production- you have two predictive factors, so this is an “anova-like” analysis

-produce a table that shows mean Butterfat by the two factors -produce a boxplot that shows all 10 combinations- does it look like there are differences? -build a GLM and explain what it is telling you

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

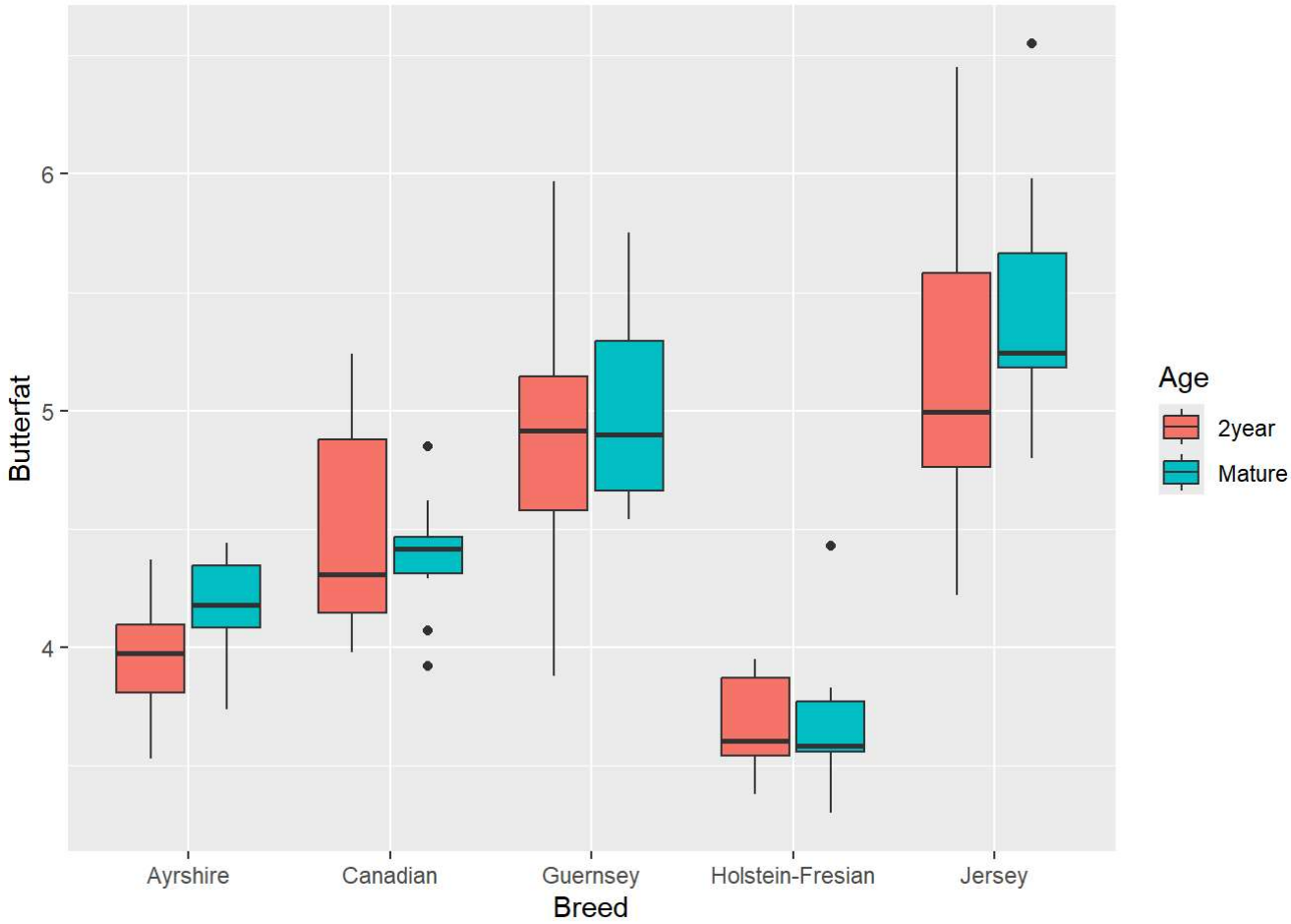
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
butterfat$Breed=factor(butterfat$Breed)
butterfat$Age=factor(butterfat$Age)

butterfat %>% group_by(Age, Breed) %>% summarize(mean_butterfat = mean(Butterfat))
```

```
## `summarise()` has grouped output by 'Age'. You can override using the `.groups`
## argument.
```

```
## # A tibble: 10 × 3
## # Groups:   Age [2]
##    Age    Breed            mean_butterfat
##    <fct>  <fct>                     <dbl>
##  1 2year  Ayrshire                   3.97
##  2 2year  Canadian                   4.49
##  3 2year  Guernsey                   4.90
##  4 2year  Holstein-Fresian           3.66
##  5 2year  Jersey                     5.13
##  6 Mature Ayrshire                   4.15
##  7 Mature Canadian                   4.39
##  8 Mature Guernsey                   5.00
##  9 Mature Holstein-Fresian           3.68
## 10 Mature Jersey                     5.45
```

```
ggplot(
  data=butterfat,
  mapping=aes(x=Breed, y=Butterfat, fill=Age)
) + geom_boxplot()
```



The age ranges seem to be relatively consistent by breed. There is much more variance in butterfat content based on breed, however. I would expect breed to be a stronger predictive factor of butterfat than age, based on this plot.

```
model=glm(Butterfat~Age+Breed,data=butterfat)

summary(model)
```

```
##
## Call:
## glm(formula = Butterfat ~ Age + Breed, data = butterfat)
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            4.00770    0.10135  39.541  < 2e-16 ***
## AgeMature              0.10460    0.08276   1.264  0.20937
## BreedCanadian          0.37850    0.13085   2.893  0.00475 **
## BreedGuernsey          0.89000    0.13085   6.802 9.48e-10 ***
## BreedHolstein-Fresian -0.39050    0.13085  -2.984  0.00362 **
## BreedJersey            1.23250    0.13085   9.419 3.16e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.1712127)
##
##     Null deviance: 50.689  on 99  degrees of freedom
## Residual deviance: 16.094  on 94  degrees of freedom
## AIC: 115.12
##
## Number of Fisher Scoring iterations: 2
```

It appears that the Jersey and Guernsey breeds are the strongest predictors of butterfat, and age does not influence the butterfat content much by comparison.

Logistic regression question

We will use the classic Iris data set

```
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 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
```

We do want to reduce this to just two species though, since the logistic regression model handles only two groups, we'll create a new data frame without setosa

```
iris2=iris[iris$Species!="setosa", ]
head(iris2)
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 51          7.0         3.2          4.7         1.4 versicolor
## 52          6.4         3.2          4.5         1.5 versicolor
## 53          6.9         3.1          4.9         1.5 versicolor
## 54          5.5         2.3          4.0         1.3 versicolor
## 55          6.5         2.8          4.6         1.5 versicolor
## 56          5.7         2.8          4.5         1.3 versicolor
```

```
 unique(iris2$Species)
```

```
## [1] versicolor virginica
## Levels: setosa versicolor virginica
```

I did this using filtering in Base R, which we haven't seem much

Show how to do this using dplyr

```
iris3 <- iris |>
  filter(Species == "versicolor" | Species == "virginica")
```

Build a logistic regression model

Predict Species using the other 4 variables

-interpret the results

-Calculate the accuracy rate of your classifier

-Create a confusion matrix

```
iris3$Species=factor(iris3$Species,labels=c("versicolor","virginica"))
model_iris = glm(Species~Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, family=binomia
l(link = "logit"), data=iris3)
summary(model_iris)
```

```
##
## Call:
## glm(formula = Species ~ Sepal.Length + Sepal.Width + Petal.Length +
##     Petal.Width, family = binomial(link = "logit"), data = iris3)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -42.638     25.707  -1.659   0.0972 .
## Sepal.Length   -2.465      2.394  -1.030   0.3032
## Sepal.Width    -6.681      4.480  -1.491   0.1359
## Petal.Length    9.429      4.737   1.991   0.0465 *
## Petal.Width    18.286      9.743   1.877   0.0605 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 138.629  on 99  degrees of freedom
## Residual deviance:  11.899  on 95  degrees of freedom
## AIC: 21.899
##
## Number of Fisher Scoring iterations: 10
```

```
# get the predictions of the probability of default

y_pred=predict.glm(object=model_iris,newdata=iris3, type="response", levels=levels(iris3$Specie
s))

# to binarize the answer into a yes/no prediction, set all values above 0.5 to 1
# and all values below to 0

y_pred=(y_pred>0.5)*1

#now convert to a factor so we can compare

y_pred=factor(y_pred,labels=c("versicolor","virginica"))

y_pred[1:10]
```

```
##          1          2          3          4          5          6          7
## versicolor versicolor versicolor versicolor versicolor versicolor versicolor
##          8          9         10
## versicolor versicolor versicolor
## Levels: versicolor virginica
```

```
# sum up the number of cases where the prediction is correct
# and divide the the size of the data set to get the rate of correct predictions

sum(y_pred==iris3$Species)/dim(iris3)[1]
```

```
## [1] 0.98
```

```
library("caret")
```

```
## Loading required package: lattice
```

```
confusionMatrix(y_pred,iris3$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   versicolor virginica
##   versicolor         49         1
##   virginica           1        49
##
##                Accuracy : 0.98
##                  95% CI : (0.9296, 0.9976)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.96
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.98
##             Specificity : 0.98
##          Pos Pred Value : 0.98
##          Neg Pred Value : 0.98
##              Prevalence : 0.50
##          Detection Rate : 0.49
##    Detection Prevalence : 0.50
##       Balanced Accuracy : 0.98
##
##        'Positive' Class : versicolor
##
```

This is accuracy level is expected. The model was trained on the same data that we predicted against. It should be very accurate because it is over fit to the training data. The coefficients from the model summary don't point to any strong predictive indicators, which is the primary factor leading me to believe the model is over fit to the data set. Just for fun, let's repeat this but separate training and testing data sets.

```
#filter again by the classes we are interested in
iris_4 <- iris |>
  filter(Species == "versicolor" | Species == "virginica")

#Lets ensure there is an even sample of the two species
versicolor = iris_4[iris_4$Species == "versicolor", ]
virginica = iris_4[iris_4$Species == "virginica", ]

#Show the lengths
length(versicolor$Species)
```

```
## [1] 50
```

```
length(virginica$Species)
```

```
## [1] 50
```

```
#The data set is currently arranged by species, so I will randomly assign the observation to tes
t or train in roughly a 60:40 split based on a uniformly random value
#I think appending to two data frames over and over will be really slow, so instead I will condi
tionally append a 1 or 0 to a vector, then use that to slice the df
test_train <- vector()
for (i in 1:length(iris_4$Species)){
  val=runif(1)
  if (val >= 0.4) {
    test_train[i]=1
  }
  else{
    test_train[i]=0
  }
}

#Select based on the value
iris_train <- iris_4[test_train==1, ]
iris_test <- iris_4[test_train==0, ]

#Ensure this worked
length(iris_train$Species)
```

```
## [1] 57
```

```
length(iris_test$Species)
```

```
## [1] 43
```

```
#Set the species as factors with the same labels
iris_train$Species=factor(iris_train$Species,labels=c("versicolor","virginica"))
iris_test$Species=factor(iris_test$Species,labels=c("versicolor","virginica"))

#retrain the model on the train data
model_iris_split = glm(Species~Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, family=b
inomial(link = "logit"), data=iris_train)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(model_iris_split)
```

```
##
## Call:
## glm(formula = Species ~ Sepal.Length + Sepal.Width + Petal.Length +
##     Petal.Width, family = binomial(link = "logit"), data = iris_train)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -153.258    116.468  -1.316    0.188
## Sepal.Length    9.133      9.283   0.984    0.325
## Sepal.Width   -11.308      8.912  -1.269    0.204
## Petal.Length   14.348      9.212   1.558    0.119
## Petal.Width    35.748     26.693   1.339    0.180
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 78.5796  on 56  degrees of freedom
## Residual deviance:  7.1633  on 52  degrees of freedom
## AIC: 17.163
##
## Number of Fisher Scoring iterations: 11
```

I have cut the training data down by 40 percent, so the model does not always converge. In the case that it does, I can run the prediction and understand the accuracy.

```
# get the predictions on the test set using the newly trained model
y_pred=predict.glm(object=model_iris_split,newdata=iris_test, type="response", levels=levels(iri
s_test$Species))

# to binarize the answer into a yes/no prediction, set all values above 0.5 to 1
# and all values below to 0

y_pred=(y_pred>0.5)*1

#now convert to a factor so we can compare

y_pred=factor(y_pred,labels=c("versicolor","virginica"))

y_pred[1:5]
```

```
##          2          7          8         10         11
## versicolor versicolor versicolor versicolor versicolor
## Levels: versicolor virginica
```

```
# sum up the number of cases where the prediction is correct
# and divide the the size of the data set to get the rate of correct predictions

sum(y_pred==iris_test$Species)/dim(iris_test)[1]
```

```
## [1] 0.9534884
```

```
confusionMatrix(y_pred,iris_test$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   versicolor virginica
##   versicolor         18         1
##   virginica           1        23
##
##                Accuracy : 0.9535
##                  95% CI : (0.8419, 0.9943)
##     No Information Rate : 0.5581
##     P-Value [Acc > NIR] : 7.741e-09
##
##                   Kappa : 0.9057
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9474
##             Specificity : 0.9583
##          Pos Pred Value : 0.9474
##          Neg Pred Value : 0.9583
##              Prevalence : 0.4419
##          Detection Rate : 0.4186
##    Detection Prevalence : 0.4419
##       Balanced Accuracy : 0.9529
##
##        'Positive' Class : versicolor
##
```

Well, I stand corrected. The model predicted the species correctly with all of the novel test data. This may not be the case when I knit the assignment and it re-runs the model training and prediction, but I am pleasantly surprised.