SOLUTIONS

Exam 2

Data Science for Studying Language & the Mind

Instructions

The exam is worth **113 points**. You have **1 hour and 30 minutes** to complete the exam.

- The exam is closed book/note/computer/phone except for the provided reference sheets
- If you need to use the restroom, leave your exam and phone with the TAs
- If you finish early, you may turn in your exam and leave early

(5 points) Preliminary questions

Please complete these questions *before* the exam begins.

(a) **(1 point)** What is your full name?

(b) **(1 point)** What is your penn ID number?

(c) **(1 point)** What is your lab section TA's name?

(d) **(1 point)** Who is sitting to your left?

(e) **(1 point)** Who is sitting to your right?

1. (24 points) True or false

(a) **(2 points)** The goal of a regression model is to classify observations into distinct categories.

- ☐ True
- ☒ False

(b) **(2 points)** Model specification involves defining the functional form of the model.

- ☒ True
- ☐ False

(c) **(2 points)** The equation $y = ax + b$ expresses y as a weighted sum of inputs.

- ☒ True
- ☐ False

(d) **(2 points)** Regression is a type of supervised learning, while classification is unsupervised.

- ☐ True
- ☒ False

(e) **(2 points)** In gradient descent, we search through all possible parameters in the parameter space.

- ☐ True
- ☒ False

(f) **(2 points)** The ordinary least squares solution is an example of an iterative optimization algorithm.

- ☐ True
- ☒ False

(g) **(2 points)** Adding more predictors to a regression model will always increase the R^2 value.

- ☒ True
- ☐ False

(h) **(2 points)** An overfit model performs poorly on both the sample and predicting new values.

- ☐ True
- ☒ False

(i) **(2 points)** A reliable model will always be a highly accurate model.

- ☐ True
- ☒ False

(j) **(2 points)** The error bars on our parameter estimates will become smaller as we increase our sample size.

- ☒ True
- ☐ False

(k) **(2 points)** Support vector machines can be used for classification problems.

- ☒ True
- ☐ False

(l) **(2 points)** The logistic function always produces outputs between 0 and 1.

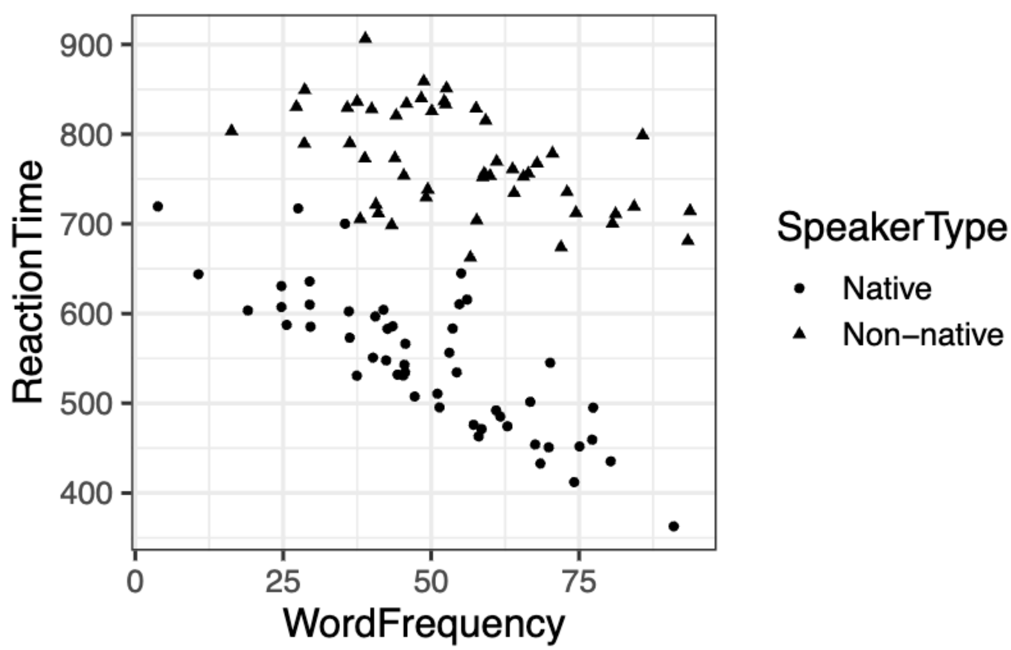
- ☒ True
- ☐ False

2. (12 points) Model specification

Suppose we measure the reaction times (in milliseconds) of both native and non-native speakers as they process words of varying frequency in English (measured as occurrences per million words). We store these data in a tibble called `rt_by_speaker`. The first 6 rows of this tibble are printed below for your reference.

```
# A tibble: 6 x 3
  WordFrequency ReactionTime SpeakerType
          <dbl>        <dbl> <chr>
1          38.8         773. Non-native
2          45.4         754. Non-native
3          81.2         711. Non-native
4          51.4         495. Native
5          52.6         851. Non-native
6          84.3         719. Non-native
```

We've also included an exploratory plot of these data.



Suppose we specify the following model with `lm`:

```
model <- lm(ReactionTime ~ 1 + WordFrequency + SpeakerType, data = rt_by_speaker)
```

(a) **(3 points)** Write the model's specification as a mathematical expression:

```
y = w0 + w1 * WordFrequency + w2 * SpeakerType
(or similar)
```

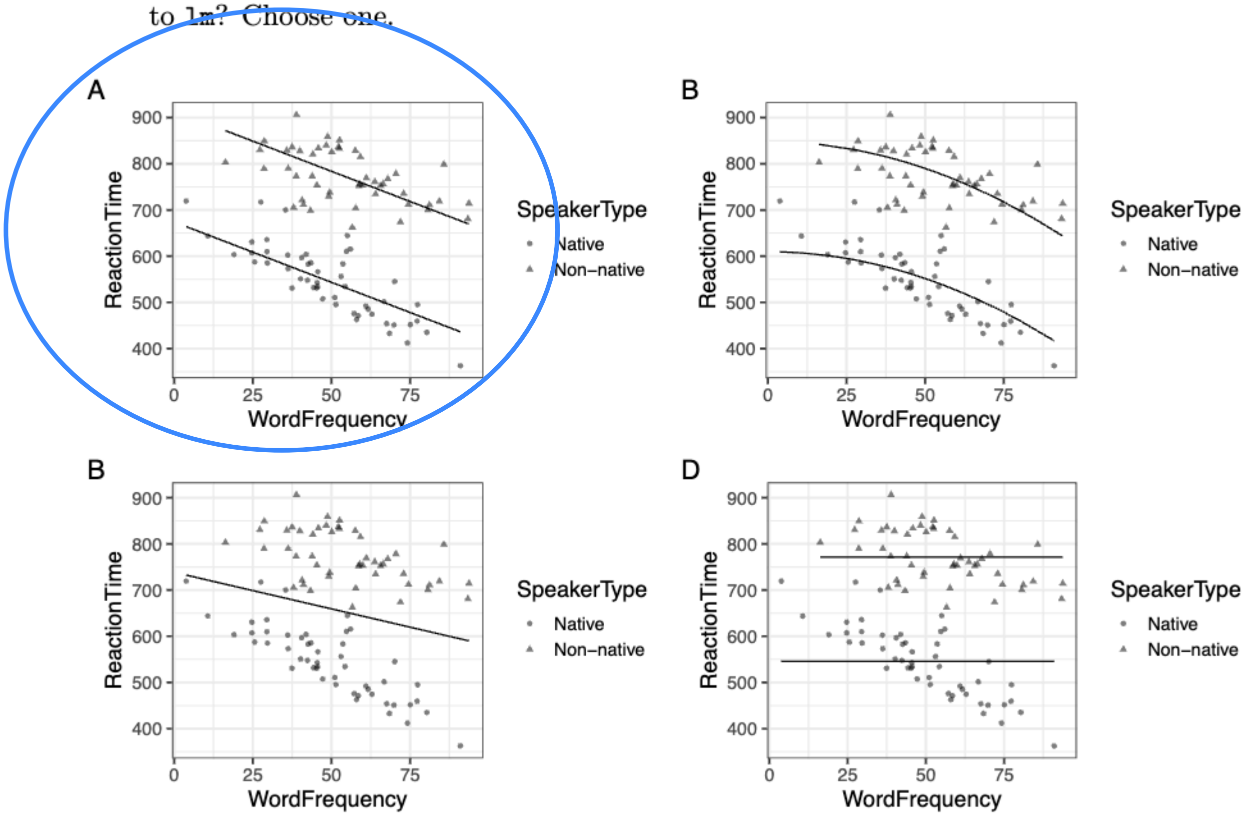
(b) **(3 points)** For each of the following, circle the option that best describes the type of model we fit.

(i) **(1 point)** Supervised or unsupervised

(ii) **(1 point)** Regression or classification

(iii) **(1 point)** Linear or linearlizable nonlinear

(c) **(3 points)** Each of the figures below show a model's predictions for these data plotted with black lines. Circle the figure that is most likely to be the plot of the model spcified to `lm`? Choose one.



(d) **(3 points)** Suppose we also fit the model with **infer**, which returns the parameter estimates below. Which of the following could be the predicted reaction time for a Native speaker with a word frequency of 10?

```
# A tibble: 3 x 2
  term                    estimate
  <chr>                      <dbl>
1 intercept                 674.
2 WordFrequency              -2.61
3 SpeakerTypeNon-native     240.
```

- ☒ 647.9
- ☐ 695.1
- ☐ 887.9
- ☐ Not enough information to determine this

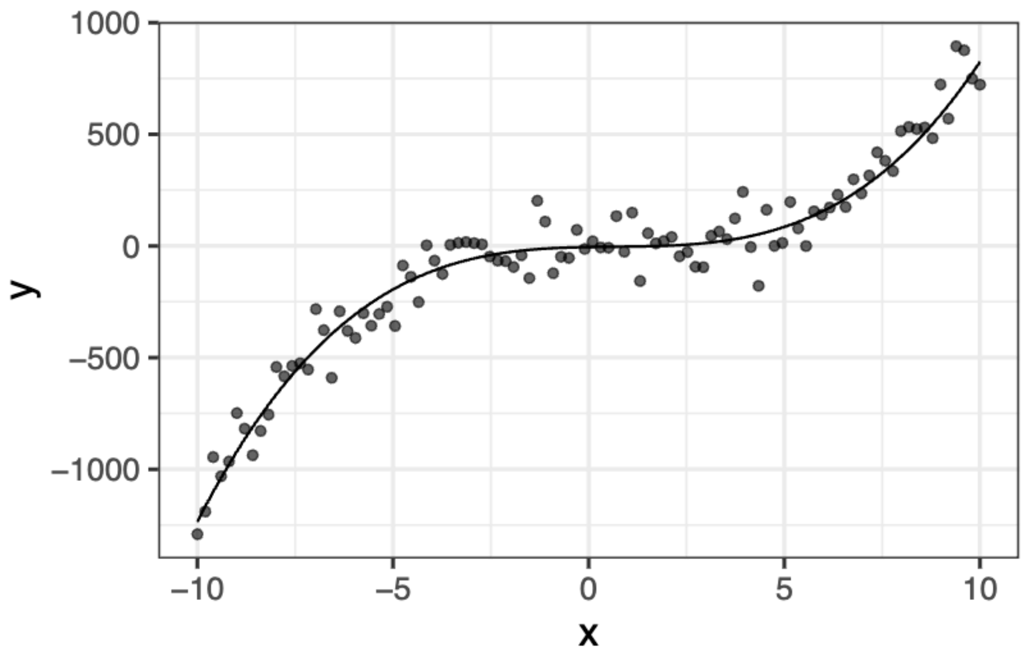
You may show your work here, if you wish:

```
674*1 + -2.61*10 + 240*0
674 - 26.10 + 0
647.9 (or only value less than 674)
showing work is optional
```

3. (12 points) Applied model specification

Suppose we encounter the following dataset, glimpsed and plotted here.

```
Rows: 100
Columns: 2
$ x <dbl> -10.000000, -9.797980, -9.595960, -9.393939, -9.191919, -8.989899, -~
$ y <dbl> -1291.0476, -1190.0226, -945.7013, -1031.6017, -965.2677, -748.6480,~
```



We specify and fit these data with `lm` as below:

```
lm(y ~ poly(x, 3) , data = data)
```

```
Call:
lm(formula = y ~ poly(x, 3), data = data)

Coefficients:
(Intercept)  poly(x, 3)1  poly(x, 3)2  poly(x, 3)3
     -63.97      3816.56      -514.32      1568.49
```

(a) **(2 points)** What type of polynomial is included in the model specification?

- ☐ Constant
- ☐ Linear
- ☐ Quadratic
- ☒ Cubic
- ☐ Quartic

(b) **(3 points)** Write the *fitted model* as a mathematical expression:

y=-63.97*1+3816.56*x-514.32*x^2+1568.49*x^3

(c) **(2 points)** In class we learned about two ways to linearlize a nonlinear model. Which option best describes what we have done here?

- ☒ Expanding the input space by adding new terms
- ☐ Transforming an existing term

(d) **(2 points)** Given the predicted model (shown with the black line on the figure), what does the model predict for the value of y when $x = 1$?

0

(e) **(3 points)** Suppose we fit the model specification `y ~ poly(x, 100)`. Explain why this would be an overfit model.

Using poly(x, 100) with 100 data points would fit the data exactly, capturing noise rather than the true underlying pattern. This leads to overfitting, where the model fits the data perfectly but performs poorly on predicting new data

4. (13 points) Model fitting

Section 4 refers to the `rt_by_speaker` tibble from section 2. We have returned the first 6 rows of the tibble here for your reference.

```
# A tibble: 6 x 3
  WordFrequency ReactionTime SpeakerType
          <dbl>        <dbl> <chr>
1          38.8         773. Non-native
2          45.4         754. Non-native
3          81.2         711. Non-native
4          51.4         495. Native
5          52.6         851. Non-native
6          84.3         719. Non-native
```

Suppose we estimate the free parameters with **optimg** and **lm**, which return the following results:

```r
optimg(data = rt_by_speaker, par = c(0,0, 0), fn=SSE, method = "STGD")
```

```
$par
[1] 674.046758  -2.612294 240.353670

$value
[1] 244250.2

$counts
[1] 24

$convergence
[1] 0
```

```r
lm(ReactionTime ~ 1 + WordFrequency + SpeakerType, data = rt_by_speaker)
```

```
Call:
lm(formula = ReactionTime ~ 1 + WordFrequency + SpeakerType,
    data = rt_by_speaker)

Coefficients:
          (Intercept)          WordFrequency  SpeakerTypeNon-native
              674.052                 -2.613                240.361
```

(a) **(2 points)** Explain why `optimg` and `lm` return slightly different parameter estimates?

lm uses the OLS (an exact analytical solution), while optimg uses gradient descent (which is uses iterative optimization to get as close as possible to the optimal free parameters).

(b) **(2 points)** What is the cost function used by `optimg`? Choose one.

- ☒ SSE
- ☐ STGD
- ☐ Gradient descent
- ☐ R^2
- ☐ Not enough information to determine this

(c) **(2 points)** How many steps did our iterative optimization algorithm take?

24

(d) **(2 points)** What was the sum of squared error of the optimal parameters according to `optimg`? Choose one.

- ☐ 24
- ☐ 0
- ☒ 244250.2
- ☐ 244250.2^2
- ☐ Not enough information to determine this

(e) **(2 points)** Which approach does `lm` use to estimate the free parameters? Choose one.

- ☒ Ordinary least-squares solution
- ☐ Gradient descent
- ☐ Another iterative optimzation algorith
- ☐ All of the above

(f) **(3 points)** Given the model specified in the code to `lm`, fill in the missing values for the first 6 rows of the input matrix **X**.

$$\begin{bmatrix} 773 \\ 754 \\ 711 \\ 495 \\ 851 \\ 719 \end{bmatrix} = \begin{bmatrix} 1 & 38.8 & \underline{\quad 1 \quad} \\ 1 & 45.4 & \underline{\quad 1 \quad} \\ 1 & 81.2 & \underline{\quad 1 \quad} \\ 1 & 51.4 & \underline{\quad 0 \quad} \\ 1 & 52.6 & \underline{\quad 1 \quad} \\ 1 & 84.3 & \underline{\quad 1 \quad} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{bmatrix}$$

5. (12 points) Model accuracy

Suppose we want to determine how accurate our model is for the **rt_by_speaker** dataset. Section 5 refers to the following code and output.

First we specify and fit our model with **lm** and return the model summary.

```r
model <- lm(ReactionTime ~ 1 + WordFrequency + SpeakerType, data = rt_by_speaker)
summary(model)
```

```
Call:
lm(formula = ReactionTime ~ 1 + WordFrequency + SpeakerType,
    data = rt_by_speaker)

Residuals:
     Min       1Q   Median       3Q      Max
-109.805  -31.329   -2.827   26.158  118.645

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)           674.0520    15.4094  43.743  < 2e-16 ***
WordFrequency          -2.6125     0.2796  -9.342 3.53e-15 ***
SpeakerTypeNon-native 240.3609    10.1616  23.654  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 50.18 on 97 degrees of freedom
Multiple R-squared:  0.8593,    Adjusted R-squared:  0.8564
F-statistic: 296.2 on 2 and 97 DF,  p-value: < 2.2e-16
```

Then we perform cross-validation and return the validation metrics with `collect_metrics()`

```
set.seed(2)
splits <- vfold_cv(rt_by_speaker)

model_spec <-
  linear_reg() %>%
  set_engine(engine = "lm")

our_workflow <-
  workflow() %>%
  add_model(model_spec) %>%
  add_formula(ReactionTime ~ 1 + WordFrequency + SpeakerType)

fitted_models <-
  fit_resamples(
    object = our_workflow,
    resamples = splits
    )

fitted_models %>%
    collect_metrics()
```

```
# A tibble: 2 x 6
  .metric .estimator   mean     n std_err .config
  <chr>   <chr>       <dbl> <int>   <dbl> <chr>
1 rmse    standard   50.7      10  2.19   Preprocessor1_Model1
2 rsq     standard    0.865    10  0.0300 Preprocessor1_Model1
```

(a) **(2 points)** What is the R^2 value for our original sample?

0.8593 (or 0.8564)

(b) **(2 points)** What is the R^2 estimate for the population?

0.865

(c) **(2 points)** What kind of cross-validation did we perform? Choose one.

- ☒ k-fold
- ☐ boostrapping
- ☐ leave-one out
- ☐ Not enough information to determine this

(d) **(2 points)** How many splits of our data does our code generate?

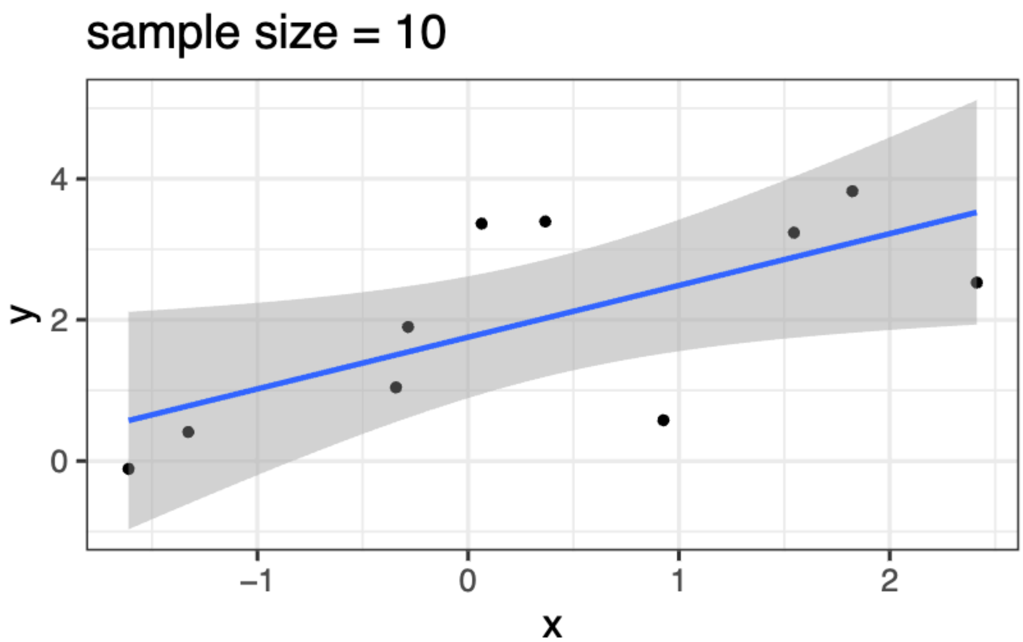
- ☐ 1000
- ☐ 100
- ☒ 10
- ☐ Not enough information to determine this

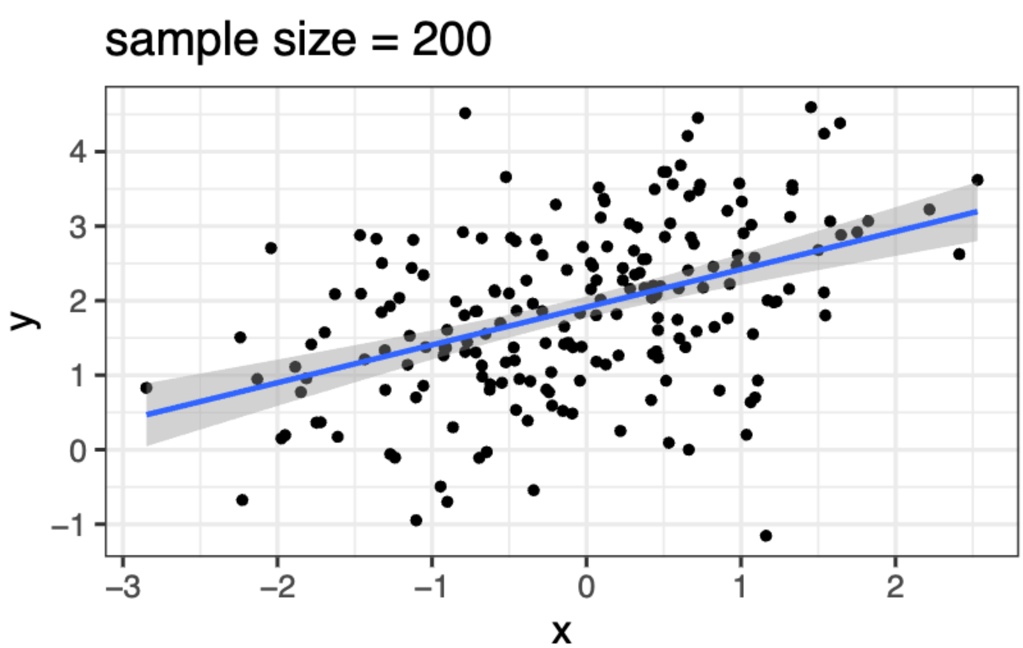
(e) **(3 points)** Explain the 3-step process that applies to all types of cross-validation.

1. Leave some data out
2. Fit a model to the kept in data
3. Evaluate the model on the left out data

6. (12 points) Model reliability

Section 6 refers to two datasets: `data_n10` and `data_n200` which have 10 and 200 observations respectively. Here we plot the data and the fitted model `y ~ 1 + x` for each dataset.





Here we return the model summary for each.

```
Call:
lm(formula = y ~ x, data = data_n10)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.8557 -0.6285 -0.0113  0.6370  1.5624 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)   
(Intercept)   1.7548     0.3740   4.692  0.00156 **
x             0.7333     0.2862   2.562  0.03352 * 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.138 on 8 degrees of freedom
Multiple R-squared:  0.4508,	Adjusted R-squared:  0.3821 
F-statistic: 6.566 on 1 and 8 DF,  p-value: 0.03352
```

```
Call:
lm(formula = y ~ x, data = data_n200)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.6565 -0.6757  0.0689  0.6032  3.0019 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.91308    0.07233  26.448  < 2e-16 ***
x            0.50704    0.07236   7.007 3.72e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.021 on 198 degrees of freedom
Multiple R-squared:  0.1987,	Adjusted R-squared:  0.1947 
F-statistic:  49.1 on 1 and 198 DF,  p-value: 3.724e-11
```

(a) **(2 points)** Which model is more accurate? Choose one.

- ☒ The model fitted to `data_n10`
- ☐ The model fitted to `data_n200`
- ☐ Both models are equally accurate
- ☐ Not enough information to determine this

(b) **(2 points)** Which model is more reliable? Choose one.

- ☐ The model fitted to `data_n10`
- ☒ The model fitted to `data_n200`
- ☐ Both models are equally reliable
- ☐ Not enough information to determine this

(c) **(2 points)** Which value in the model summary quantifies the model's reliability?

- ☐ Multiple R-squared
- ☐ Adjusted R-squared
- ☐ Estimate
- ☒ Std. Error
- ☐ Pr(>|t|)

(d) **(3 points)** Suppose we bootstrap a 95% confidence interval for our parameter estimates for the `data_n10` dataset. What would happen if we changed the level of the confidence interval to 68%? Choose one.

- ☒ It would get smaller (narrower)
- ☐ It would get bigger (wider)
- ☐ It would stay the same

(e) **(3 points)** Explain why there is uncertainty on our model parameter estimates.

Because we are interested in the model parameters that best describe the population from which the sample was drawn. Due to sampling error, we can expect some variability in the model parameters.

7. (13 points) Classification

Suppose we want to predict the Fruit_Type (0 = apple, 1 = banana) based on its Weight, Color (1 = red, 2 = yellow, 3 = green), and Diameter. Our data is stored in the tibble `fruit_data`, glimpsed below.

```
Rows: 1,000
Columns: 4
$ Weight     <int> 113, 149, 217, 142, 113, 217, 189, 190, 190, 191, 236, 198,~
$ Color      <int> 3, 2, 1, 1, 1, 2, 1, 1, 3, 2, 3, 3, 3, 2, 1, 3, 2, 1, 3, 2,~
$ Diameter   <dbl> 21.8, 13.4, 19.3, 19.7, 24.7, 24.2, 7.8, 18.3, 5.7, 14.9, 2~
$ Fruit_Type <dbl> 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1,~
```

We fit this model with `glm` and return the following output:

```
glm(Fruit_Type ~ Weight + Color + Diameter, family = "binomial", data = fruit_data)
```

```
Call:  glm(formula = Fruit_Type ~ Weight + Color + Diameter, family = "binomial",
    data = fruit_data)

Coefficients:
(Intercept)       Weight        Color     Diameter
  -2.994585     0.001124    -0.005461     0.101965

Degrees of Freedom: 999 Total (i.e. Null);  996 Residual
Null Deviance:      1093
Residual Deviance: 1034     AIC: 1042
```

(a) **(3 points)** For each of the following, circle the option that best describes the type of model we fit.

(i) **(1 point)** Supervised or unsupervised
(ii) **(1 point)** Regression or classification
(iii) **(1 point)** Linear or linearlizable nonlinear

(b) **(2 points)** How many free parameters is this model estimating?

- ☐ 1
- ☐ 2
- ☐ 3
- ☒ 4
- ☐ Not enough information to determine this

(c) **(2 points)** Which of the following parsnip specifications could specify and fit a generalized linear model?

- ☐ `linear_reg() %>% set_engine("lm")`
- ☒ `logistic_reg() %>% set_enging("glm")`
- ☐ Both work

(d) **(2 points)** Which of the following expresses the link function for the `glm` we fit?

- ☒ $f(a) = \frac{1}{1+e^{-a}}$
- ☐ $\sum_{i=i}^{n}(d_i - m_i)^2$
- ☐ $y = \sum_{i=1}^{n} w_i x_i$
- ☐ $R^2 = 100 \times (1 - \frac{SSE_{model}}{SSE_{reference}})$

(e) **(2 points)** What do we call the type of classification we performed via our `glm`?

- ☐ linear regression
- ☒ logistic regression
- ☐ nearest-prototype regression
- ☐ support vector machine

(f) **(2 points)** What accuracy metric is best applied to classification models?

- ☐ R^2
- ☐ RMSE - root mean squared error
- ☒ Percent correct
- ☐ Adjusted R^2