Assignment: Examining Gender Pay Gap with Bayesian Regression

Emma Kruis

2020-01-04

Instructions

This assignment reviews the *Bayesian Regression* content. You will use the *bayesian_regression.Rmd* file I reviewed as part of the lectures for this week to complete this assignment. You will *copy and paste* relevant code from that file and update it to answer the questions in this assignment. You will respond to questions in each section after executing relevant code to answer a question. You will submit this assignment to its *Submissions* folder on *D2L*. You will submit *two* files:

1. this completed *R Markdown* script, and
2. as a first preference, a *PDF* (if you already installed `TinyTeX` properly), as a second preference, a *Microsfot Word* (if your computer has *Microsoft Word*) document, or, as a third preference, an *HTML* (if you did *not* install `TinyTeX` properly and your computer does *not* have *Microsoft Word*) file to *D2L*.

To start:

First, create a folder on your computer to save all relevant files for this course. If you did not do so already, you will want to create a folder named *mgt_592* that contains all of the materials for this course.

Second, inside of *mgt_592*, you will create a folder to host assignments. You can name that folder *assignments*.

Third, inside of *assignments*, you will create folders for each assignment. You can name the folder for this first assignment: *bayesian_regression*.

Fourth, create three additional folders in *bayesian_regression* named *scripts*, *data*, and *plots*. Store this script in the *scripts* folder and the data for this assignment in the *data* folder.

Fifth, go to the *File* menu in *RStudio*, select *New Project...*, choose *Existing Directory*, go to your *~/mgt_592/assignments/bayesian_regression* folder to select it as the top-level directory for this **R Project**.

Global Settings

The first code chunk sets the global settings for the remaining code chunks in the document. Do *not* change anything in this code chunk.

Load Packages

In this code chunk, we load the following packages:

1. **here**,
2. **tidyverse**,

3. **janitor**,
4. **skimr**,
5. **ggthemes**,
6. **rstanarm**,
7. **bayesplot**, and
8. **tidybayes**.

Make sure you installed these packages when you reviewed the analytical lecture.

We will use functions from these packages to examine the data. Do *not* change anything in this code chunk.

```
### load libraries for use in current working session
## here for project work flow
library(here)

## tidyverse for data manipulation and plotting
## loads eight different libraries simultaneously
library(tidyverse)

## janitor to clean data and chi-square test
library(janitor)

## skimr to summarize data
library(skimr)

## ggthemes for plot themes
library(ggthemes)

## rstanarm for Bayesian regression
library(rstanarm)

## bayesplot for Bayesian posterior distributions
library(bayesplot)

## tidybayes to work with Bayesian model results
library(tidybayes)
```

Task 1: Import Data

We will use the same data as in the analytical lecture: **gender_pay_gap.csv**. After you load the data, then you will execute other commands on the data.

Task 1.1

Use the **read_csv()** and **here()** functions to load the data file for this working session. Save the data as the object **org_raw**.

Make a copy of the data and name the copy: **org_work**. You will work with the *complete data*. Use the **glimpse()** function to view a preview of values for each variable in **org_work**.

Questions 1.1: Answer these questions: (1) How many *variables* are there in the data table? (2) How many *observations* are there in the data table? (3) How many *character variables* are there in the data table?

Responses 1.1: *There are 9 variables, 1000 observations and 4 charcater variables.*

```
org_raw<- read_csv(
  here("data", "gender_pay_gap.csv")
)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   jobTitle = col_character(),
##   gender = col_character(),
##   age = col_double(),
##   perfEval = col_double(),
##   edu = col_character(),
##   dept = col_character(),
##   seniority = col_double(),
##   basePay = col_double(),
##   bonus = col_double()
## )
```

```
org_work <- org_raw


glimpse(org_work)
```

```
## Rows: 1,000
## Columns: 9
## $ jobTitle  <chr> "Graphic Designer", "Software Engineer", "Warehouse Assoc...
## $ gender    <chr> "Female", "Male", "Female", "Male", "Male", "Female", "Fe...
## $ age       <dbl> 18, 21, 19, 20, 26, 20, 20, 18, 33, 35, 24, 18, 19, 30, 3...
## $ perfEval  <dbl> 5, 5, 4, 5, 5, 5, 5, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, ...
## $ edu       <chr> "College", "College", "PhD", "Masters", "Masters", "PhD",...
## $ dept      <chr> "Operations", "Management", "Administration", "Sales", "E...
## $ seniority <dbl> 2, 5, 5, 4, 5, 4, 4, 5, 5, 5, 5, 3, 3, 5, 4, 3, 5, 5, 5, ...
## $ basePay   <dbl> 42363, 108476, 90208, 108080, 99464, 70890, 67585, 97523,...
## $ bonus     <dbl> 9938, 11128, 9268, 10154, 9319, 10126, 10541, 10240, 9836...
```

Task 2: Clean and Prepare Data

For this task, you will clean and prepare the data.

Task 2.1

Perform the following cleaning tasks to update **org_work**:

1. mutate all character variables to factor variables, and
2. use **clean_names()** to make variable names use all lowercase letters and connect multiple words with underscores.

Use **glimpse()** to preview the updated **org_work** data object.

Questions 2.1: Answer these questions: (1) How many variable names use an *underscore* (i.e., _)? (2) What is the difference in the *display of values* for character and factor variables (*Hint*: compare the output from *Task 1.1* to *Task 2.1*)?

Responses 2.1: *3 variables (job_title, perf_eval, base_pay) use an underscore. The display values for characters in the pervious activity changed to factor variables in the data table..*

```
org_work <- org_work %>%
  mutate(
    across(
      .cols = where(is.character),
      .fns = as_factor
    )
  ) %>%
  clean_names()

glimpse(org_work)
```

```
## Rows: 1,000
## Columns: 9
## $ job_title <fct> Graphic Designer, Software Engineer, Warehouse Associate,...
## $ gender    <fct> Female, Male, Female, Male, Male, Female, Female, Male, F...
## $ age       <dbl> 18, 21, 19, 20, 26, 20, 20, 18, 33, 35, 24, 18, 19, 30, 3...
## $ perf_eval <dbl> 5, 5, 4, 5, 5, 5, 5, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, ...
## $ edu       <fct> College, College, PhD, Masters, Masters, PhD, College, Ph...
## $ dept      <fct> Operations, Management, Administration, Sales, Engineerin...
## $ seniority <dbl> 2, 5, 5, 4, 5, 4, 4, 5, 5, 5, 5, 3, 3, 5, 4, 3, 5, 5, 5, ...
## $ base_pay  <dbl> 42363, 108476, 90208, 108080, 99464, 70890, 67585, 97523,...
## $ bonus     <dbl> 9938, 11128, 9268, 10154, 9319, 10126, 10541, 10240, 9836...
```

Task 2.2

Perform the following preparation tasks to update **org_work**:

1. cut the variable **age** into **four** equal interval bins,
2. compute a new variable named **total_pay** as the *sum* of **base_pay** and **bonus**,
3. compute the *natural logarithm* of **base_pay**, **bonus**, and **total_pay** and name the new variables by *column* and *function*, and
4. reorder all factor variables so that the highest frequency factor level is listed first.

Use **glimpse()** to preview the updated **org_work** data object.

Questions 2.2: Answer these questions: (1) What is the *age bin* of the *first* person? (2) What is the *natural logarithm* of the *total pay* of the *first* person?

Responses 2.2: *"the age_bin for the first persons is 18 - 29.8! the natural logarithm of the total pay for the first person is 10.86477. .*

```
org_work <- org_work %>%
  mutate(
    age_bin = cut_interval(
      age,
      n = 4
      ),
    total_pay = base_pay + bonus,
    across(
      .cols = c(base_pay, bonus, total_pay),
```

```
      .fns = list(log = log),
      .names= "{.col}_{.fn}"
    ),
    across(
      .cols = where(is.factor),
      .fns = fct_infreq
      )
    )

glimpse(org_work)
```

```
## Rows: 1,000
## Columns: 14
## $ job_title     <fct> Graphic Designer, Software Engineer, Warehouse Associ...
## $ gender        <fct> Female, Male, Female, Male, Male, Female, Female, Mal...
## $ age           <dbl> 18, 21, 19, 20, 26, 20, 20, 18, 33, 35, 24, 18, 19, 3...
## $ perf_eval     <dbl> 5, 5, 4, 5, 5, 5, 5, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,...
## $ edu           <fct> College, College, PhD, Masters, Masters, PhD, College...
## $ dept          <fct> Operations, Management, Administration, Sales, Engine...
## $ seniority     <dbl> 2, 5, 5, 4, 5, 4, 4, 5, 5, 5, 5, 3, 3, 5, 4, 3, 5, 5,...
## $ base_pay      <dbl> 42363, 108476, 90208, 108080, 99464, 70890, 67585, 97...
## $ bonus         <dbl> 9938, 11128, 9268, 10154, 9319, 10126, 10541, 10240, ...
## $ age_bin       <fct> "[18,29.8]", "[18,29.8]", "[18,29.8]", "[18,29.8]", "...
## $ total_pay     <dbl> 52301, 119604, 99476, 118234, 108783, 81016, 78126, 1...
## $ base_pay_log  <dbl> 10.65403, 11.59428, 11.40987, 11.59063, 11.50755, 11....
## $ bonus_log     <dbl> 9.204121, 9.317220, 9.134323, 9.225623, 9.139811, 9.2...
## $ total_pay_log <dbl> 10.86477, 11.69194, 11.50767, 11.68042, 11.59711, 11....
```

Task 3: Examine Data

For this task, you will examine the data.

Task 3.1

Summarize **org_work** by:

1. grouping by **gender**,
2. selecting variables that contain the string **pay** and the variable **perf_eval**, and
3. applying **skim_without_charts()**.

Questions 3.1: Answer these questions: (1) What is the *median raw base pay* for *women*? (2) What is the *25th percentile* of the *natural logarithm* of *total pay* for *men*?

Responses 3.1: *The median raw base pay for women is 89914. the natural logarithm of the total pay for men in the 25th percentile is 11.4.* .

```
org_work %>%
  group_by(gender) %>%
  select(contains("pay"), perf_eval) %>%
  skim_without_charts()
```

```
## Adding missing grouping variables: ‘gender‘
```

Table 1: Data summary

Name	Piped data
Number of rows	1000
Number of columns	6
Column type frequency:	
numeric	5
Group variables	gender

Variable type: numeric

skim_variable	gender	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
base_pay	Male	0	1	98457.55	25517.52	36642.00	81452.50	98223.00	115606.25	179726.00
base_pay	Female	0	1	89942.82	24378.28	34208.00	73186.25	89913.50	106923.25	160614.00
total_pay	Male	0	1	104918.68	25329.57	41030.00	87791.75	105100.50	121617.00	184010.00
total_pay	Female	0	1	96416.83	24202.16	40828.00	80866.50	96571.00	112660.50	168968.00
base_pay_log	Male	0	1	11.46	0.27	10.51	11.31	11.49	11.66	12.10
base_pay_log	Female	0	1	11.37	0.29	10.44	11.20	11.41	11.58	11.99
total_pay_log	Male	0	1	11.53	0.25	10.62	11.38	11.56	11.71	12.12
total_pay_log	Female	0	1	11.44	0.27	10.62	11.30	11.48	11.63	12.04
perf_eval	Male	0	1	3.13	1.41	1.00	2.00	3.00	4.00	5.00
perf_eval	Female	0	1	2.94	1.43	1.00	2.00	3.00	4.00	5.00

Task 3.2

Summarize **org_work** by:

1. grouping by **gender** and **age_bin**,
2. applying **summarize()** and calculating the *mean* and *median* of **total_pay**, and
3. counting the number of individuals in the groups by applying **n()**.

Questions 3.2: Answer these questions: (1) How many *women* are in the *18 to 29.8 age bin* group? (2) What is the *median total pay* for *men* in the *29.8 to 41.5 age bin* group? (3) For what *age bin* group is the difference in *mean total pay* the largest between *men* and *women*?

Responses 3.2: *There are 112 women in the 18 to 29.8 age bin. The median total pay for men in the 29.8 to 41.5 age bin is 97581. The largest difference between men and women in mean total pay is in the age bin of 53.2 to 65..*

```
org_work %>%
  group_by(gender, age_bin) %>%
  summarize(
    across(
      .cols = total_pay,
      .fns = list(mean = mean, median = median),
      .names = "{col}_{.fn}"
    ),
    count = n(),
    .groups = "drop"
  )
```

```
## # A tibble: 8 x 5
##   gender age_bin     total_pay_mean total_pay_median count
## * <fct>  <fct>                <dbl>            <dbl> <int>
## 1 Male   [18,29.8]            88058.           87448    153
## 2 Male   (53.2,65]           124900.          125261    144
## 3 Male   (29.8,41.5]          98443.           97581    126
## 4 Male   (41.5,53.2]         109674.          110447    109
## 5 Female [18,29.8]            79840.           80664.   112
## 6 Female (53.2,65]           111556.          110738    119
## 7 Female (29.8,41.5]          89171.           90140.   114
## 8 Female (41.5,53.2]         103581.          101584    123
```

Task 3.3

Create a plot using **ggplot()** to highlight the differences in *total pay* between *men* and *women* for different *age* groups.

Do the following:

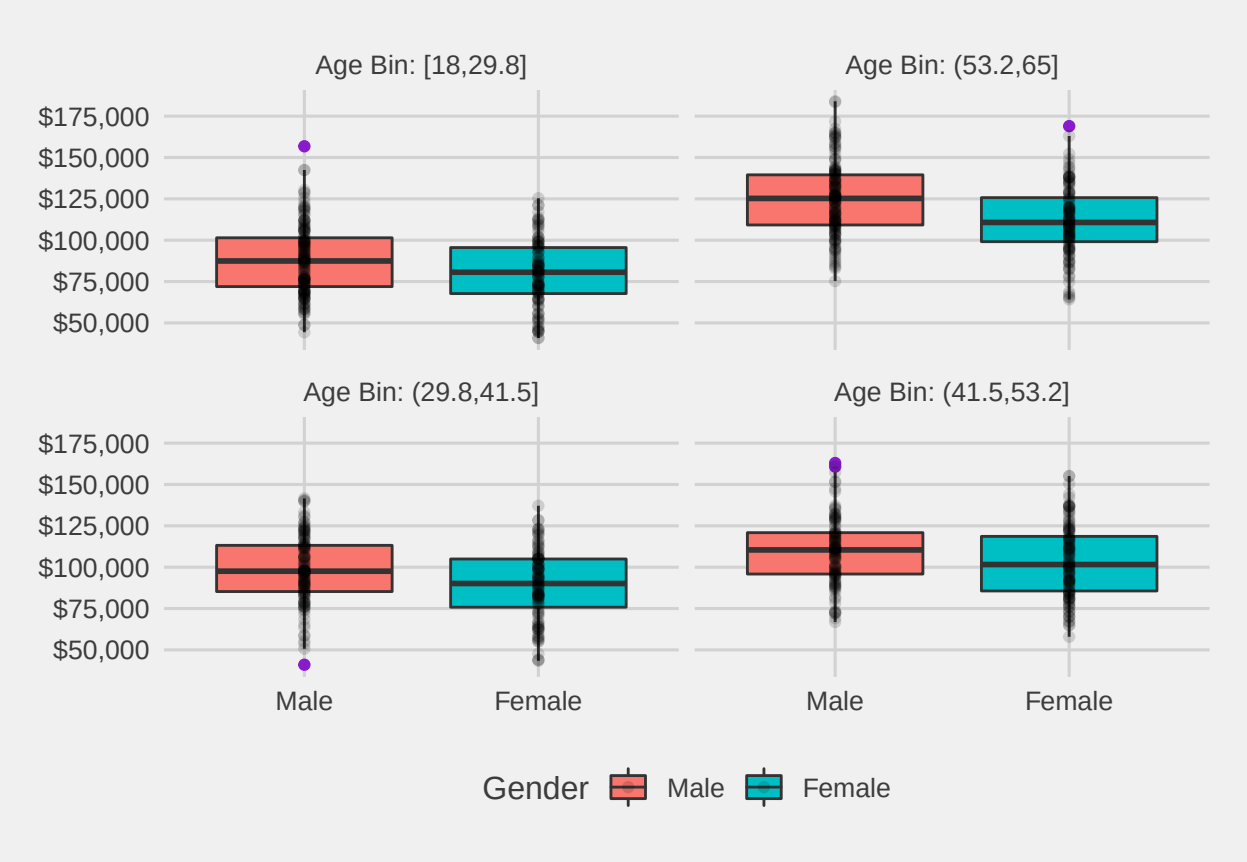
1. call **ggplot()**, set the data to **org_work**, and map **gender** to *x-axis* and **fill** and **total_pay** to the *y-axis*;
2. call **geom_boxplot()** and set the outlier **color** to **purple** and **size** to **1.5**;
3. call **geom_point()** and set **alpha** to **0.15**;
4. call **facet_wrap()** to facet by **age_bin** with **2** rows;
5. inside of **facet_wrap()**, set the **labeller** input using **as_labeller()** with **paste("Age Bin", levels(org_work**$age_bin, sep = ": ") ** as the first and ** levels(org_work**$age_bin)** as the second input into **setNames()**;
6. call **scale_y_continuous** and set the number of breaks to **8** and the **labels** to *dollar format*;
7. label the axes and legend appropriately with **labs()**;
8. use **theme_fivethirtyeight()** as the *theme*.

Questions 3.3: Answer these questions: (1) Are there *outliers* for *men* in the *41.5 to 53.2 age bin* group? (2) Is the *median total pay* higher for *women* in any of the *age bin* groups?

Responses 3.3: *There are outliers for men in the age bin of 41.5 to 53.2. The median total pay is not higher for women in any of the age bins* .

```
ggplot(
  org_work,
  aes(x =gender, y = total_pay, fill = gender)
) +
  geom_boxplot(outlier.color = "purple", outlier.size = 1.5) +
  geom_point(alpha = 0.15) +
  facet_wrap(
    vars(age_bin),
    nrow = 2,
    labeller = as_labeller(
      setNames(
      paste("Age Bin", levels(org_work$age_bin), sep = ": "),
      levels(org_work$age_bin)
      )
    )
  )+
  scale_y_continuous(n.breaks = 8, labels = scales::dollar_format()) +
```

```
labs(x = "Gender", Y = "Total Pay", fill= "Gender") +
theme_fivethirtyeight()
```



Task 4: Fit Simple Bayesian Regression Model

Estimate a simple Bayesian regression model using using **org_work** where observed values of **total_pay** are predicted from observed values of **gender**.

Task 4.1

Create a model object named **mod_1** using **stan_glm()**. Inside of **stan_glm()**, do the following:

1. specify the *formula* to indicate **total_pay** predicted by **gender**,
2. specify the *data* as **org_work**,
3. set the *prior intercept* to a **normal** distribution with **location** equal to **1e+05** and **autoscale** equal to **TRUE**,
4. set the *prior of the regression coefficient for the predictor* to a **normal** distribution with **location** equal to **1000** and **autoscale** equal to **TRUE**,
5. set the *error prior* to be an **exponential** distribution with **rate** equal to **0.8** and **autoscale** equal to **TRUE**, and
6. set the **seed** to **1805** (birth year of *William Rowan Hamilton*).

After creating the model, then do the following:

1. apply **summary()** to **mod_1** using *three digits* and **seq(0.1, 0.9, 0.2)** for the *percentiles* to examine the posterior parameter distributions,
2. apply **coef()** to **mod_1** to extract the *median* regression coefficients,
3. apply **posterior_interval** to **mod_1** and set the *credible interval* to **0.85**, and
4. apply **prior_summary()** to **mod_1** to examine the prior distributions.

Questions 4.1: Answer these questions: (1) How much less money do *70%* of *women* earn than *men*? (2) What is the estimated *median total pay* for *men*? (3) According to the estimated posterior distribution, *women* earn less than *men* with *85%* probability in what interval of *total pay difference*? (4) What is the *adjusted prior scale* of the *gender* regression coefficient?

Responses 4.1: *70% of women earn 7685.062 less than men. The estimated median total pay for men is 104926.931. Women earn less than men with 85% probability from 10776.06 to 6179.88 dollars for total pay difference. The adjusted prior scale of the gender regression coefficient is 125978..*

```
##model creation
mod_1 <- stan_glm(
  total_pay ~ gender,
  data = org_work,
  prior_intercept = normal(location= 1e+05, autoscale = TRUE),
  prior = normal(location = 1000, autoscale = TRUE),
  prior_aux = exponential(rate = 0.8, autoscale = TRUE),
  seed = 1805
)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.000179 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 1.79 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.45243 seconds (Warm-up)
## Chain 1:                0.115465 seconds (Sampling)
## Chain 1:                1.5679 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.9e-05 seconds
```

```
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 1.54713 seconds (Warm-up)
## Chain 2:                0.125235 seconds (Sampling)
## Chain 2:                1.67236 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.6e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.16 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 1.29398 seconds (Warm-up)
## Chain 3:                0.11147 seconds (Sampling)
## Chain 3:                1.40545 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.2e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
```

```
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 1.18079 seconds (Warm-up)
## Chain 4:                0.106522 seconds (Sampling)
## Chain 4:                1.28731 seconds (Total)
## Chain 4:
```

```
##model summary
summary(mod_1, digits = 3, probs = seq(0.1, 0.9,0.2))
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      total_pay ~ gender
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 1000
##  predictors:   2
##
## Estimates:
##                mean       sd         10%        30%        50%        70%
## (Intercept)  104915.895   1108.303 103491.688 104342.857 104926.931 105509.435
## genderFemale  -8500.392   1588.742 -10538.019  -9305.746  -8458.246  -7685.062
## sigma         24839.955    556.264  24136.327  24552.021  24830.207  25109.447
##                90%
## (Intercept)  106321.967
## genderFemale  -6476.160
## sigma         25570.651
##
## Fit Diagnostics:
##            mean       sd         10%        30%        50%        70%
## mean_PPD 100936.134   1137.860  99483.188 100325.547 100922.578 101522.975
##            90%
## mean_PPD 102398.879
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##               mcse   Rhat   n_eff
## (Intercept)   18.206  1.000 3706
## genderFemale  25.346  1.001 3929
```

```
## sigma          9.043  1.000 3784
## mean_PPD      18.562  1.000 3758
## log-posterior  0.031  1.002 1782
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

```
##extracting coefficients
coef(mod_1)
```

```
##  (Intercept) genderFemale
##   104926.931    -8458.246
```

```
##examine credible intervals
posterior_interval(mod_1, prob = 0.85)
```

```
##                    7.5%      92.5%
## (Intercept)  103321.84 106493.742
## genderFemale -10776.06  -6179.886
## sigma         24040.41  25668.552
```

```
##examine priors
prior_summary(mod_1)
```

```
## Priors for model 'mod_1'
## ------
## Intercept (after predictors centered)
##   Specified prior:
##     ~ normal(location = 1e+05, scale = 2.5)
##   Adjusted prior:
##     ~ normal(location = 1e+05, scale = 62891)
##
## Coefficients
##   Specified prior:
##     ~ normal(location = 1000, scale = 2.5)
##   Adjusted prior:
##     ~ normal(location = 1000, scale = 125978)
##
## Auxiliary (sigma)
##   Specified prior:
##     ~ exponential(rate = 0.8)
##   Adjusted prior:
##     ~ exponential(rate = 3.2e-05)
## ------
## See help('prior_summary.stanreg') for more details
```

Task 4.2

Examine **mod_1** by doing the following:

1. compute the *Bayesian R-squared* using **bayes_R2()** and saving the results to an object named **mod_1_r_sq**;

2. apply **summary()** to **mod_1_r_sq**;
3. apply **pp_check()** and examine the *density overlay*, the *mean* on its own, and the *mean* and *sd* together;
4. compute the *fitted values* using the *median posterior regression parameters* and save the caluclation as a new variable named **mod_1_fitted** to **org_work**;
5. open the *spreadsheet view* of **org_work** to answer a question about **mod_1_fitted** values.

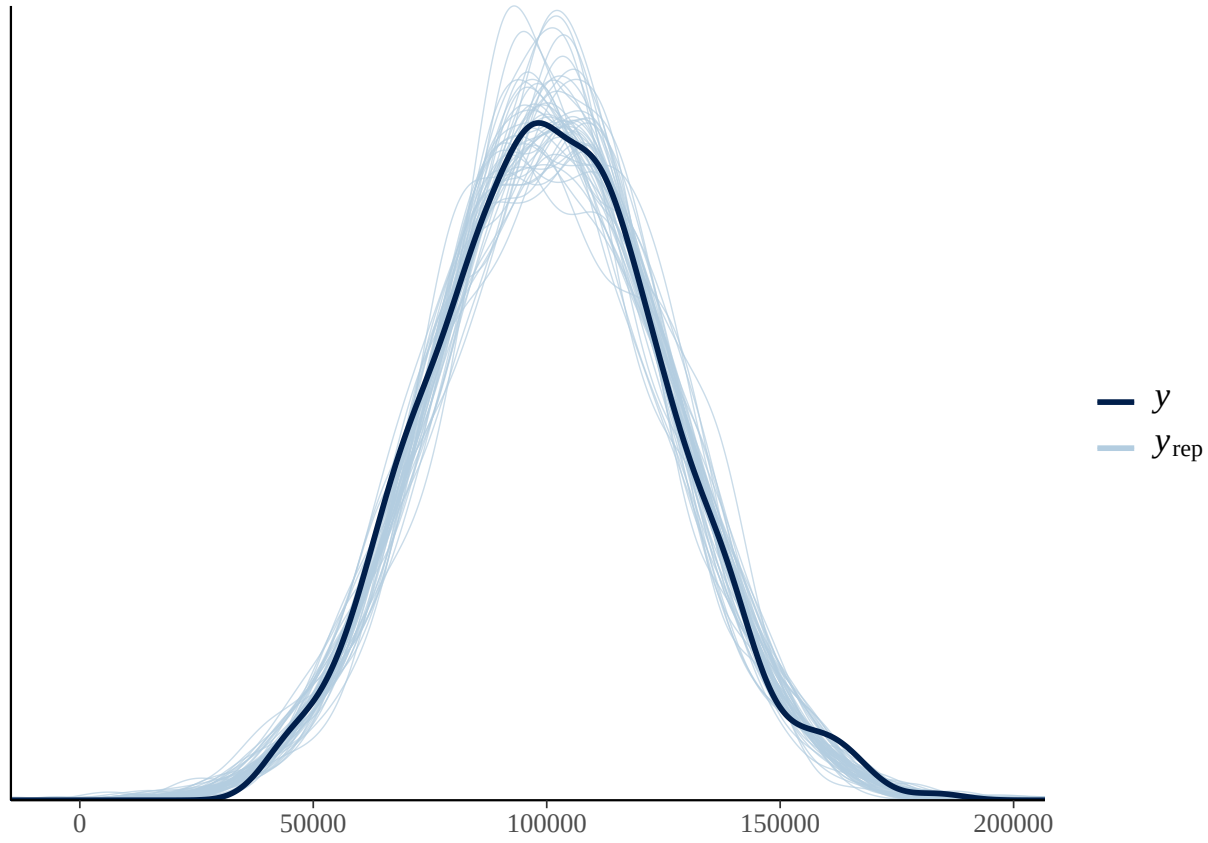
Questions 4.2: Answer these questions: (1) What is the *maximum R-squared* value? (2) Do the *posterior predictive checks* indicate any estimation issues? (3) How come the *median posterior fitted values* calculated in **mod_1_fitted** only take on two values?

Responses 4.2: *The maximum r-squared value is 0.083549. The Posterior predictive checks do not indicate any estimation issues. The median posterior fitted values calculated in mod 1 fitted only takes 2 values because there are only males and females in this data therefore we will only have two different posterior probability outcomes. .*

```
mod_1_r_sq <- bayes_R2(mod_1)
summary(mod_1_r_sq)
```

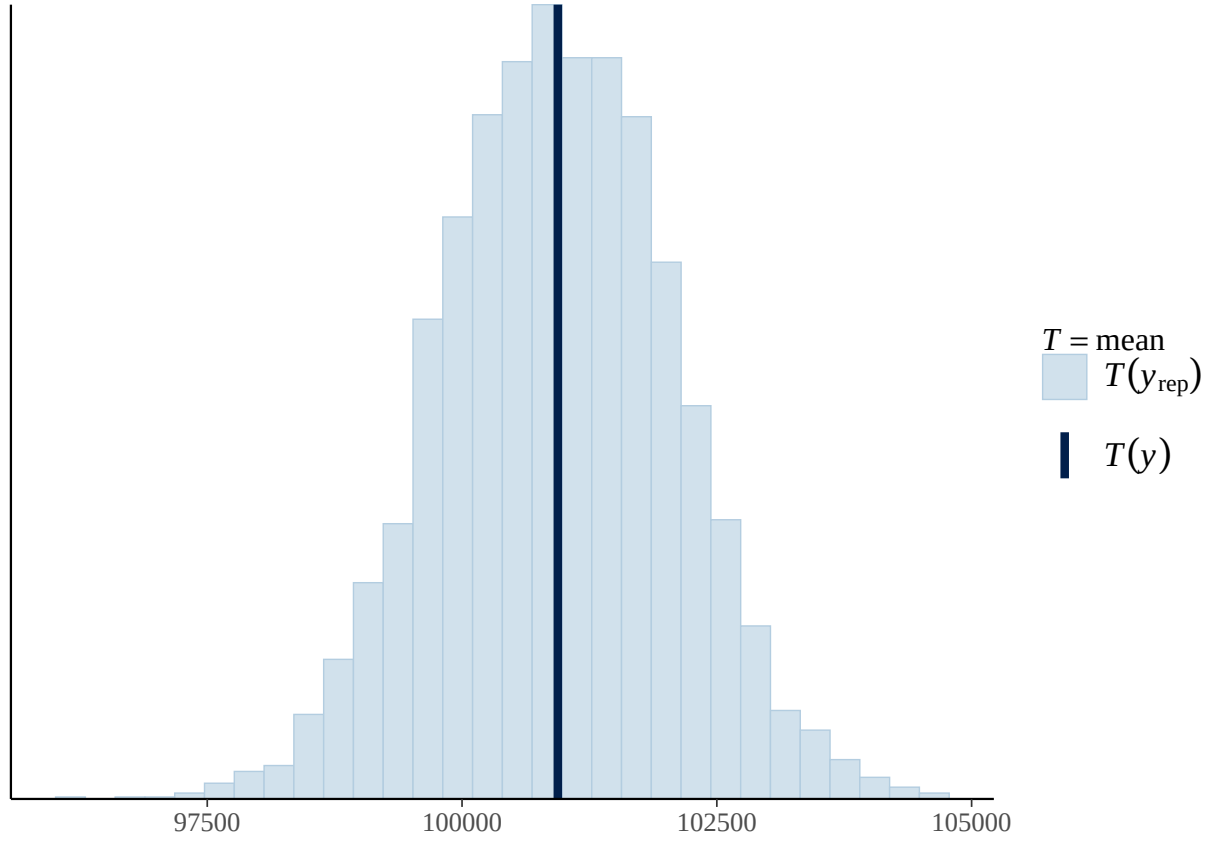
```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.003599 0.021921 0.028238 0.029241 0.035750 0.083549
```

```
##posterior predictive checks
#density overlay
pp_check(mod_1, "dens_overlay")
```

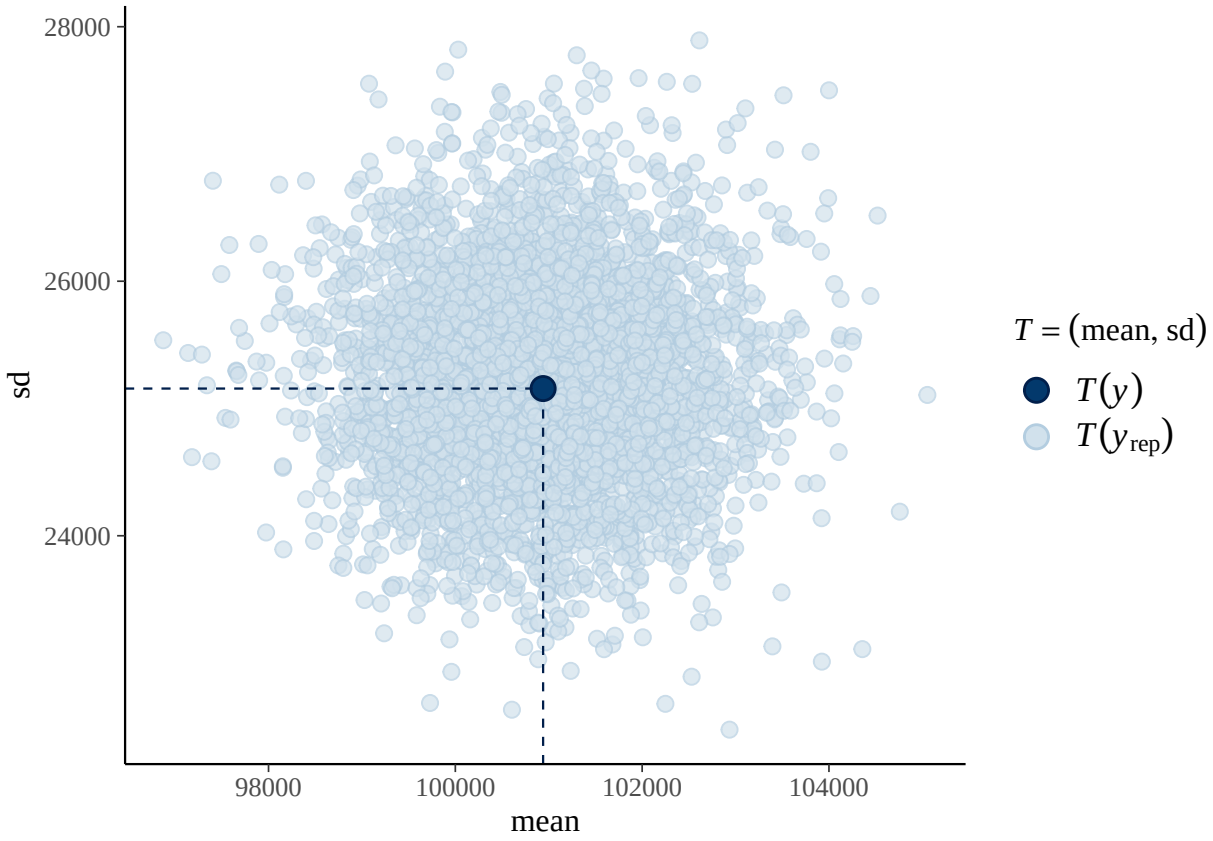


```
#mean
pp_check(mod_1, plotfun = "stat", stat= "mean")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#mean and sd
pp_check(mod_1, plotfun = "stat_2d", stat = c("mean", "sd"))
```




```
##median posterior fitted values
org_work <- org_work %>%
  mutate(mod_1_fitted = fitted(mod_1))

mod_1_fitted <- as_tibble(mod_1)


mod_1_fitted
```

```
## # A tibble: 4,000 x 3
##    `(Intercept)` genderFemale  sigma
##            <dbl>        <dbl>  <dbl>
##  1       103926.       -5959. 24961.
##  2       103776.       -5522. 24892.
##  3       104436.       -6140. 24559.
##  4       105677.      -11350. 24927.
##  5       106635.      -12634. 24661.
##  6       103491.       -4926. 24446.
##  7       104652.       -7823. 25080.
##  8       102770.       -5506. 25338.
##  9       104491.       -7424. 25000.
## 10       105940.       -9546. 24960.
## # ... with 3,990 more rows
```

Task 4.3

Extract the draws from the posterior regression parameter distributions from **mod_1** with **as_tibble()** and save them as **mod_1_post**. Create a plot using **mod_1_post** to show the posterior distribution for the *gender* regression coefficient. Do the following to create the plot:

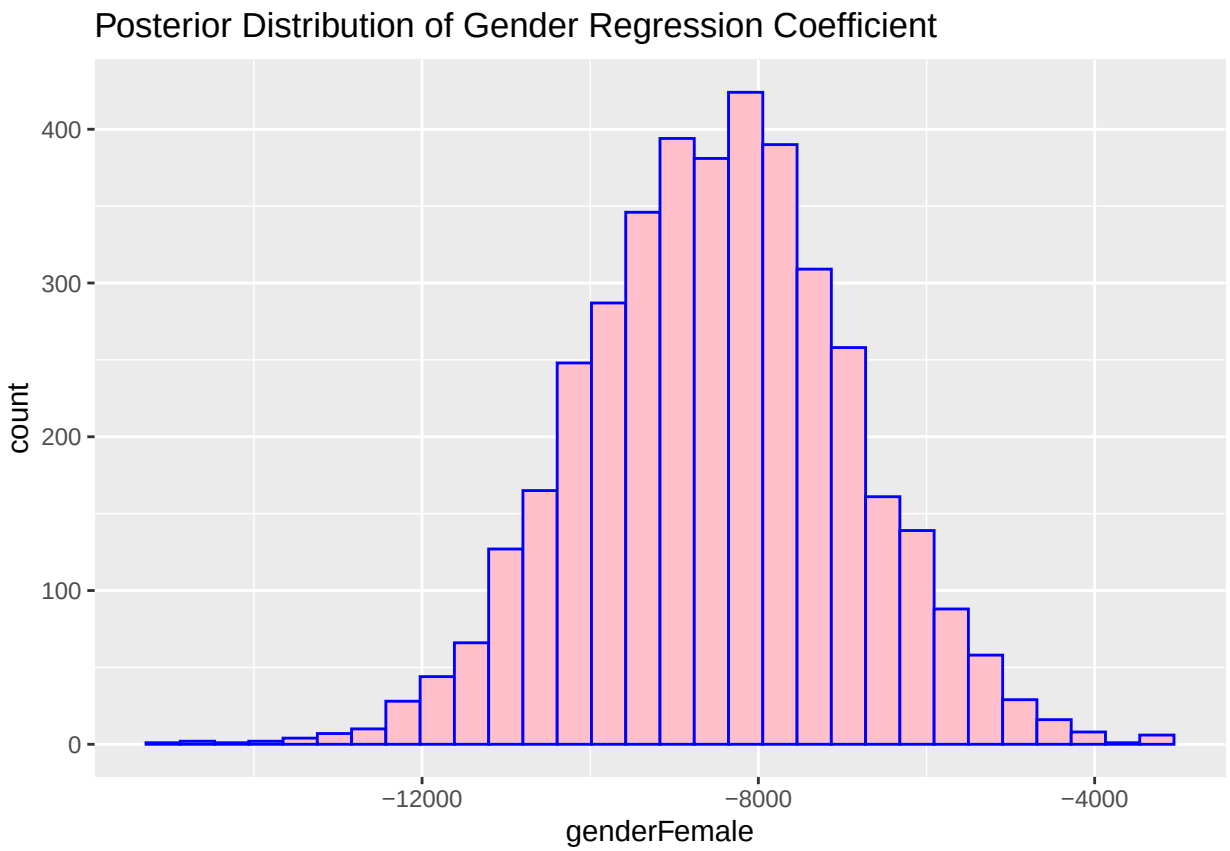
1. call **ggplot()** and set data to **mod_1_post** and map the *gender* column to the *x-axis*;
2. call **geom_histogram()** and set the **color** to **blue** and **fill** to **pink**;
3. call **ggtitle()** and set the *title* of the plot to **Posterior Distribution of Gender Regression Coefficient**.

Question 4.3: What do you learn from examining the *posterior distribution* of the *gender* regression coefficient?

Response 4.3: *We see from examining the posterior distribution of the gender regression coefficient that women earn less than men. This histogram shows the entire distribution for women and even in the 100th percentile women earn less than men.* .

```
mod_1_post <- as_tibble(mod_1)

ggplot(
  mod_1_post,
  aes(x = genderFemale)
)+
  geom_histogram(color = "blue", fill = "pink")+
                   ggtitle(
                     "Posterior Distribution of Gender Regression Coefficient"
                     )
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Task 4.4

Calculate the *fitted* and *predicted* values from the *posterior draws* of the regression parameters using the two possible values of *gender* set in **tibble()**. Pass the **tibble()** result to **add_fitted_draws()** and **add_predicted_draws()** while including **mod_1** as an input as well for the *fitted* and *predicted* values saving the results as **mod_1_post_fit** and **mod_1_post_pred**, respectively.

Create a plot named **mod_1_plot** using **ggplot()**. To create the plot, do the following:

1. inside of **ggplot()**, set data to **org_work**, **gender** to the *x-axis*, and **total_pay** to the *y-axis*;
2. add a first **geom_jitter()** layer with **height** and **width** set to **0.05** and **alpha** set to **0.5**.
3. add a second **geom_jitter()** layer with **height** and **width** set to **0.05**, set data to **mod_1_post_pred**, map **gender** to the *x-axis* and **.prediction** to the *y-axis*, set **color** to **lightgreen**, set **size** to **0.5**, and set **alpha** to **0.15**;
4. add a third **geom_jitter()** layer with **height** and **width** set to **0.05**, set data to **mod_1_post_fit**, map **gender** to the *x-axis* and **.value** to the *y-axis*, set **color** to **skyblue**, set **size** to **1.5**, and set **alpha** to **0.15**;
5. add a **geom_point()** layer and map **.mod_1_fitted** to the *y-axis*, set **color** to **red**, and set **size** to **2**;
6. call **scale_y_continuous** and set the number of breaks to **10** and the **labels** to *dollar format*;
7. label the axes and legend appropriately with **labs()**.

Display **mod_1_plot** by typing its name.

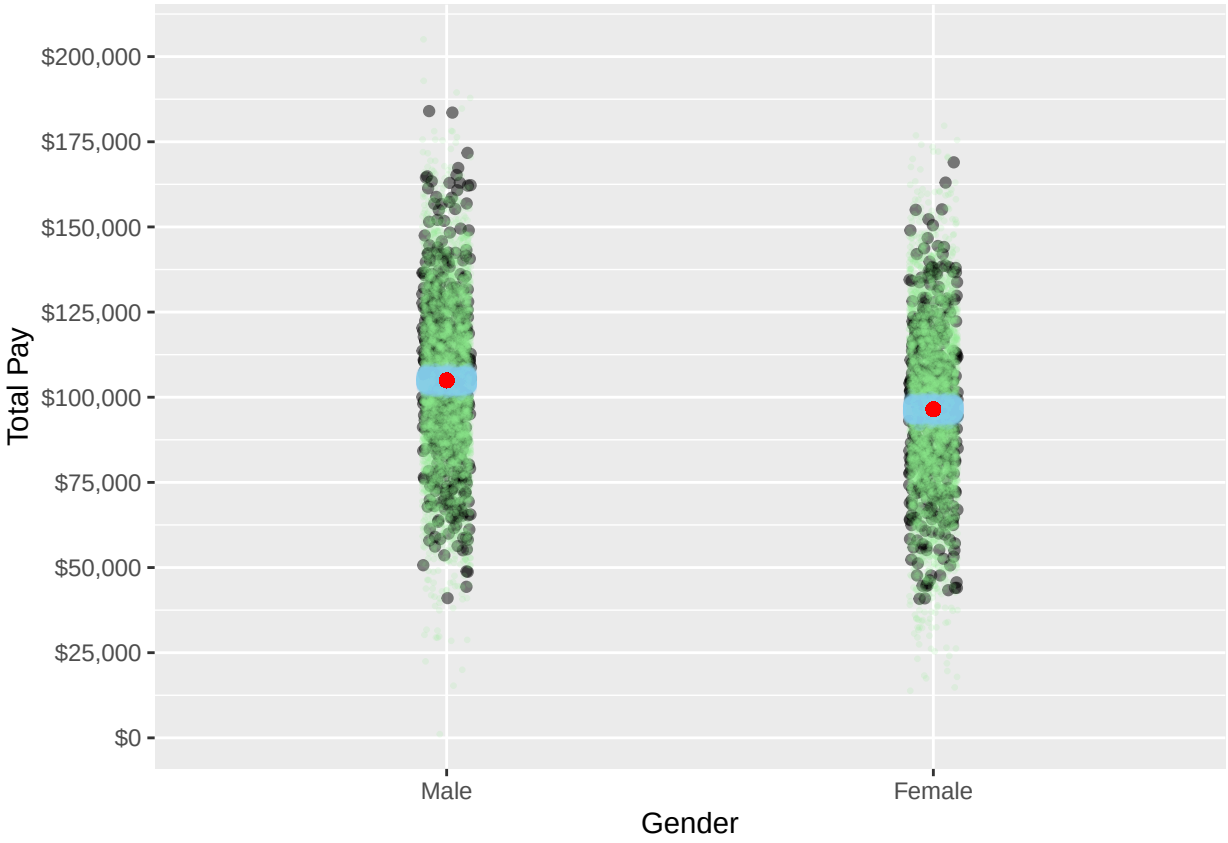
Questions 4.4: Answer these questions: (1) Do the posterior *predicted* values cover the *observed* values sufficiently well? (2) Is the range of *fitted* or *predicted* values smaller?

Responses 4.4: *The posterior predicted values cover the observed values very well. They cover the full range of the observed values. The range of the of the fitted values are much smaller than the predicted values.*
.

```
mod_1_post_fit <- tibble(gender = c("Male", "Female")) %>%
  add_fitted_draws(mod_1)
```

```
## Instead of posterior_linpred(..., transform=TRUE) please call posterior_epred(), which provides equi
```

```
mod_1_post_pred <-tibble(gender = c("Male", "Female")) %>%
  add_predicted_draws(mod_1)


mod_1_plot <- ggplot(
  org_work,
  aes(x = gender, y = total_pay)
)+
  geom_jitter(height = 0.05, width = 0.05, alpha = 0.5)+
  geom_jitter(
    height = 0.05, width = 0.05,
    data = mod_1_post_pred,
    mapping = aes(x = gender, y = .prediction),
    color = "lightgreen", size = 0.5, alpha = 0.15
  )+
  geom_jitter(
    height = 0.05, width = 0.05,
    data = mod_1_post_fit,
    mapping = aes(x = gender, y = .value),
    color = "skyblue", size = 1.5, alpha = 0.15
  )+
  geom_point(
    aes(y = mod_1_fitted),
    color = "red", size = 2
  )+
  scale_y_continuous(n.breaks = 10, labels = scales::dollar_format())+
  labs(x = "Gender", y = "Total Pay")

mod_1_plot
```



Task 5: Fit Multiple Bayesian Regression Model

Estimate a simple Bayesian regression model using using **org_work** where observed values of **total_pay** are predicted from observed values of **gender** and **age**.

Task 5.1

Create a model object named **mod_2** using **stan_glm()**. Inside of **stan_glm()**, do the following:

1. specify the *formula* to indicate **total_pay** predicted by **gender** and **age**,
2. specify the *data* as **org_work**,
3. set the *prior intercept* to a **normal** distribution with **location** equal to **1e+05** and **autoscale** equal to **TRUE**,
4. set the *prior of the regression coefficients for the predictors* to a **normal** distribution with **location** equal to **0** and **autoscale** equal to **TRUE**,
5. set the *error prior* to be an **exponential** distribution with **rate** equal to **0.8** and **autoscale** equal to **TRUE**, and
6. set the **seed** to **1805** (birth year of *William Rowan Hamilton*).

After creating the model, then do the following:

1. apply **summary()** to **mod_2** using *three digits* and **seq(0.1, 0.9, 0.2)** for the *percentiles* to examine the posterior parameter distributions,
2. apply **coef()** to **mod_2** to extract the *median* regression coefficients, and

3. apply **posterior_interval** to **mod_2** and set the *credible interval* to **0.85**.

Questions 5.1: Answer these questions: (1) What is the *90th percentile* of the *age* regression coefficient? (2) What is the estimated *median difference in total pay* between *men* and *women* irrespective of *age*? (3) According to the estimated posterior distribution, *women* earn less than *men* with *85%* probability in what interval of *total pay difference* irrespective of *age*?

Responses 5.1: *The 90th percentile of the age regression coefficient is 1008.059. The estimated median difference in total pay between men and women irrespective of age is 9299.016 dollars. With 85% probability women earn l11201.7992 to 7391.483 less than men irrespective of age in total pay. .*

```
##model creation
mod_2 <- stan_glm(
  total_pay ~ gender + age,
  data = org_work,
  prior_intercept = normal(location= 1e+05, autoscale = TRUE),
  prior = normal(location = 1000, autoscale = TRUE),
  prior_aux = exponential(rate = 0.8, autoscale = TRUE),
  seed = 1805
)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.4e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.24 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.8122 seconds (Warm-up)
## Chain 1:                0.11871 seconds (Sampling)
## Chain 1:                0.93091 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.2e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
```

```
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 1.31084 seconds (Warm-up)
## Chain 2:                0.131189 seconds (Sampling)
## Chain 2:                1.44203 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.1e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.11 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 1.04061 seconds (Warm-up)
## Chain 3:                0.113481 seconds (Sampling)
## Chain 3:                1.15409 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.3e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.13 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
```

```
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.85106 seconds (Warm-up)
## Chain 4:                0.129986 seconds (Sampling)
## Chain 4:                0.981046 seconds (Total)
## Chain 4:
```

```
##model summary
summary(mod_2, digits = 3, probs = seq(0.1, 0.9,0.2))
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      total_pay ~ gender + age
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 1000
##  predictors:   3
##
## Estimates:
##                mean       sd         10%        30%        50%        70%
## (Intercept)   65988.427   2078.760  63292.698  64885.369  66006.134  67111.098
## genderFemale  -9293.341   1330.514 -10942.447  -9992.009  -9299.016  -8566.189
## age             949.394     45.885    890.624    924.982    949.576    974.282
## sigma         20802.828    453.621  20234.111  20550.881  20795.611  21029.737
##                90%
## (Intercept)   68572.692
## genderFemale  -7585.496
## age            1008.059
## sigma         21404.430
##
## Fit Diagnostics:
##            mean       sd         10%        30%        50%        70%
## mean_PPD 100934.788    937.799  99733.904 100438.487 100920.319 101451.164
##            90%
## mean_PPD 102166.468
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##               mcse   Rhat   n_eff
## (Intercept)   28.293  1.000 5398
## genderFemale  18.064  1.000 5425
## age            0.642  1.000 5109
## sigma          6.286  1.000 5208
```

```
## mean_PPD      13.910  0.999 4545
## log-posterior  0.033  1.001 1806
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

```
##extracting coefficients
coef(mod_2)
```

```
##  (Intercept) genderFemale          age
##   66006.1342   -9299.0161     949.5759
```

```
##examine credible intervals
posterior_interval(mod_2, prob = 0.85)
```

```
##                     7.5%     92.5%
## (Intercept)   62965.4061 68967.687
## genderFemale -11201.7992 -7391.483
## age            883.7685  1015.353
## sigma        20160.8405 21477.444
```

```
##examine priors
prior_summary(mod_2)
```

```
## Priors for model 'mod_2'
## ------
## Intercept (after predictors centered)
##   Specified prior:
##     ~ normal(location = 1e+05, scale = 2.5)
##   Adjusted prior:
##     ~ normal(location = 1e+05, scale = 62891)
##
## Coefficients
##   Specified prior:
##     ~ normal(location = [1000,1000], scale = [2.5,2.5])
##   Adjusted prior:
##     ~ normal(location = [1000,1000], scale = [125978.36,  4399.59])
##
## Auxiliary (sigma)
##   Specified prior:
##     ~ exponential(rate = 0.8)
##   Adjusted prior:
##     ~ exponential(rate = 3.2e-05)
## ------
## See help('prior_summary.stanreg') for more details
```

Task 5.2

Examine **mod_2** by doing the following:

1. compute the *Bayesian R-squared* using **bayes_R2()** and saving the results to an object named **mod_2_r_sq**;

2. apply **summary()** to **mod__2__r__sq**;
3. apply **loo()** to **mod__1** and **mod__2** and save the results to objects named **mod__1__loo** and **mod__2__loo**, respectively;
4. apply **loo__compare()** to **mod__1__loo** and **mod__2__loo**;
5. apply **pp__check()** and examine the *density overlay*, the *mean* on its own, and the *mean* and *sd* together;
6. compute the *fitted values* using the *median posterior regression parameters* and save the caluclation as a new variable named **mod__2__fitted** to **org__work**;
7. open the *spreadsheet view* of **org__work** to answer a question about **mod__2__fitted** values.

Questions 5.2: Answer these questions: (1) What is the *minimum R-squared* value? (2) Using the *leave-one-out* comparison, does **mod__1** or **mod__2** fit the data better? (3) Do the *posterior predictive checks* indicate any estimation issues? (4) How come the *median posterior fitted values* for the individuals in **org__work** are different between **mod__1__fitted** and **mod__2__fitted**?

Responses 5.2: *The minimum r-squared value is 0.2464. Mod_2 fits the data better. The porsterior predictive check does not indicate any estimation issues. The median posterior fitted values are different because they are based on age. The indviduals of the same age and gender will have the same values. .*

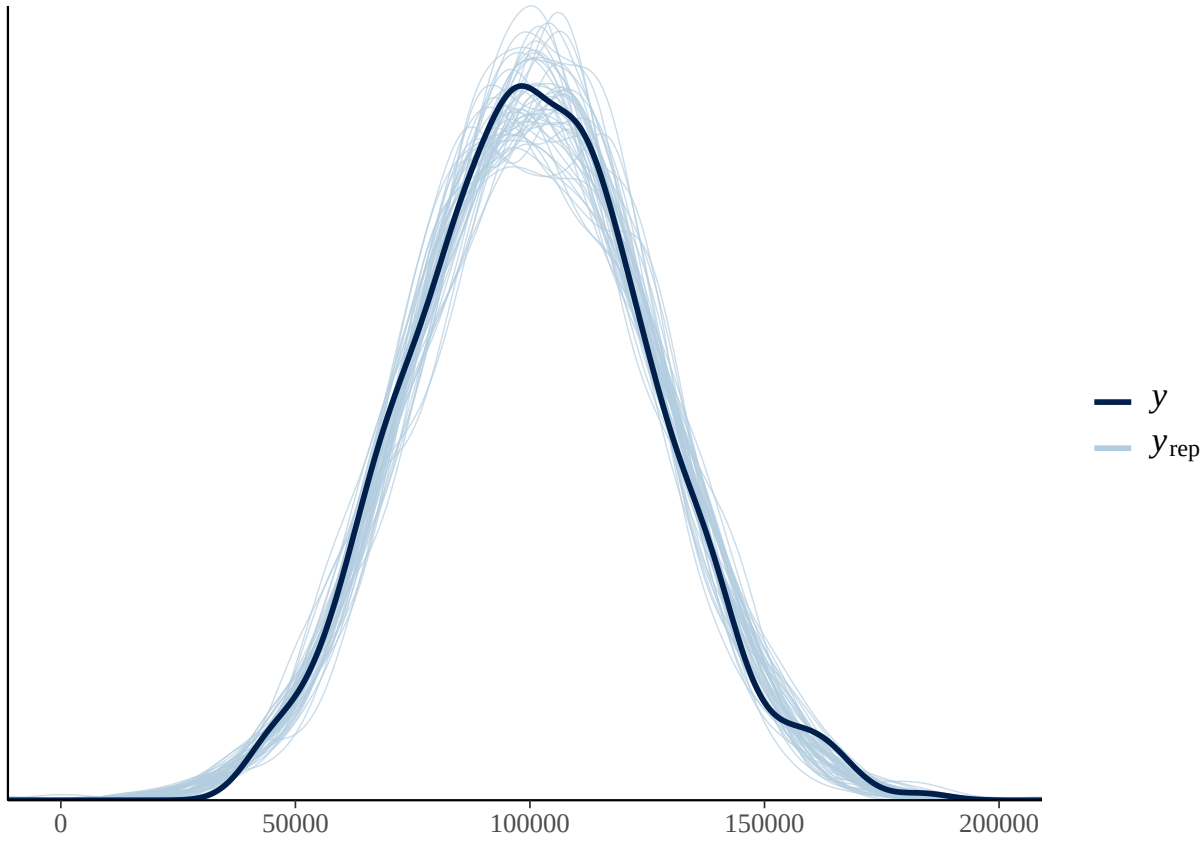
```
mod_2_r_sq <- bayes_R2(mod_2)
summary(mod_2_r_sq)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2464  0.3040  0.3190  0.3189  0.3339  0.4080
```

```
#compare model 1 and model 2 with leave-one-out cross validations
mod_1_loo <- loo(mod_1)

mod_2_loo <- loo(mod_2)

#compare loo results
loo_compare(mod_1_loo, mod_2_loo)
```

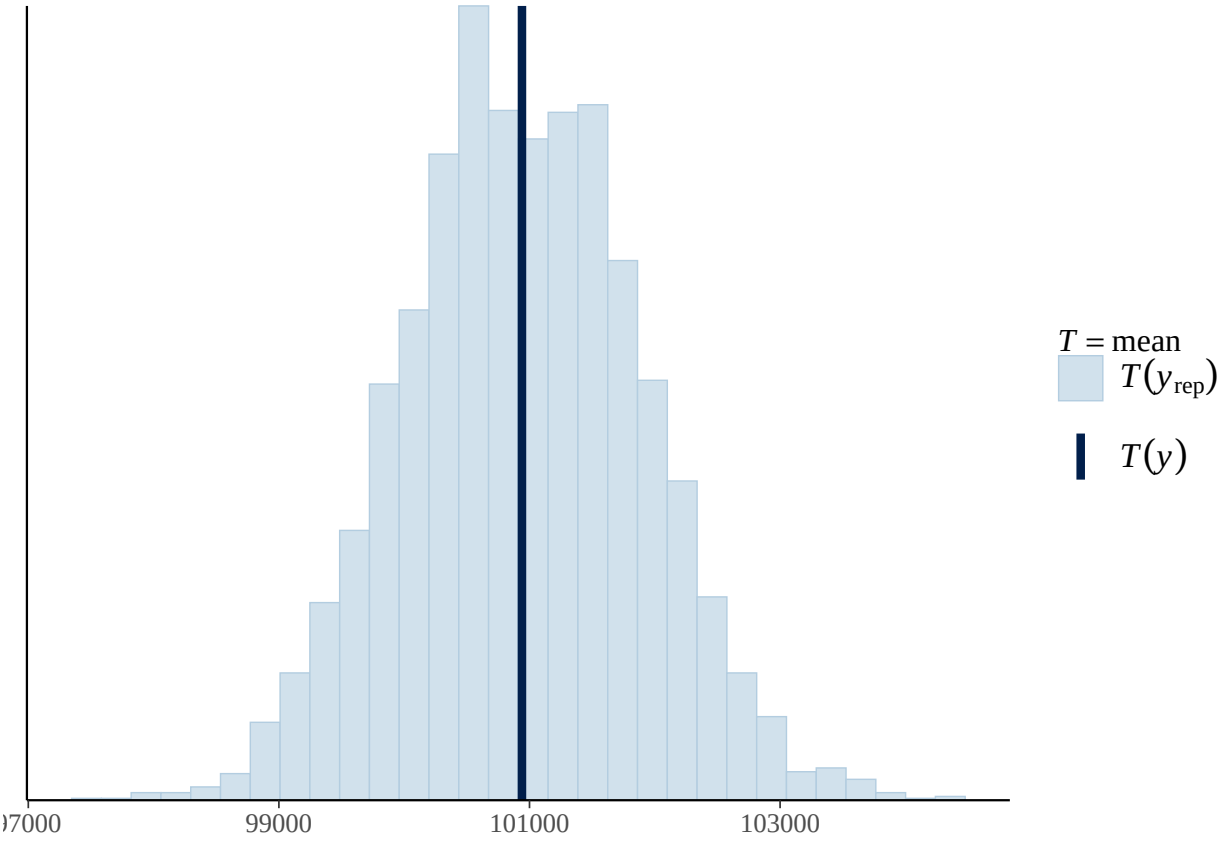
```
##       elpd_diff se_diff
## mod_2    0.0       0.0
## mod_1 -176.6      16.0
```

```
#density overlay
pp_check(mod_2, plotfun = "dens_overlay")
```

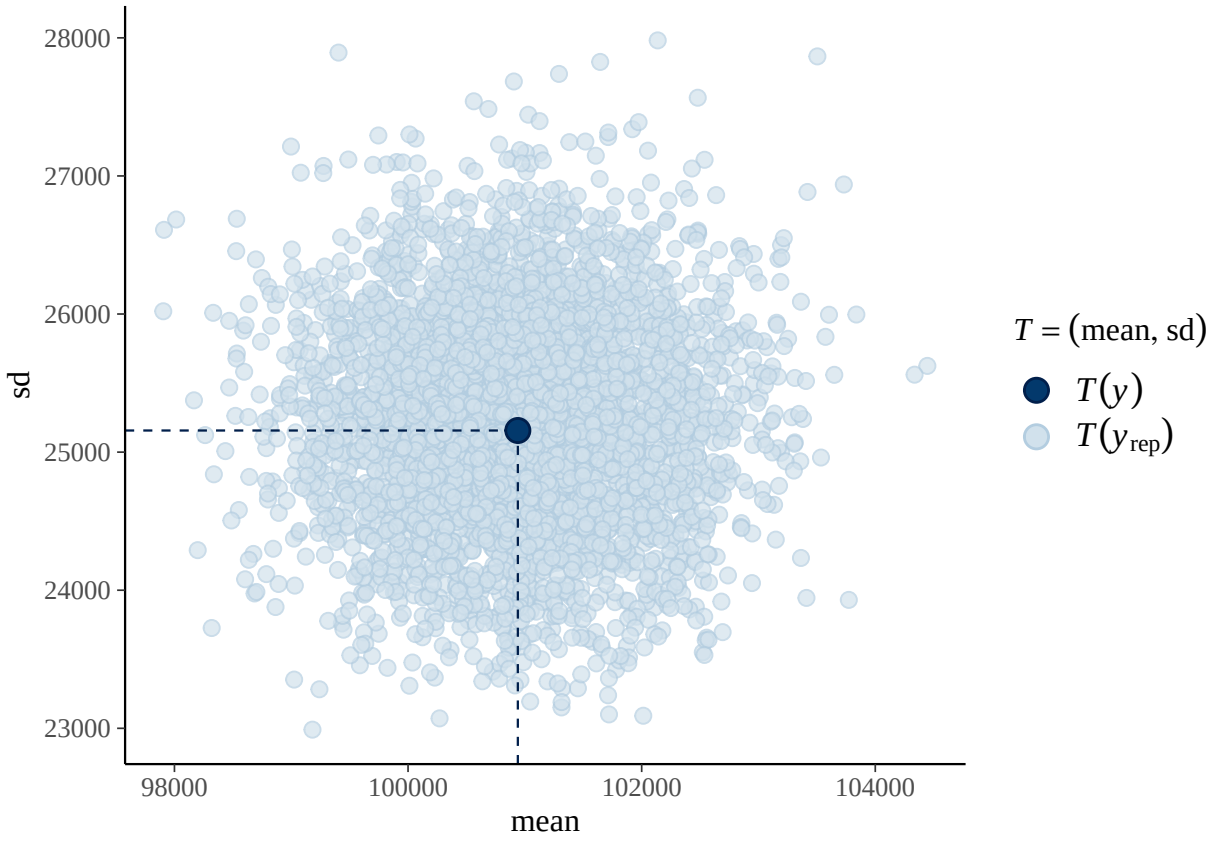



```
#mean
pp_check(mod_2, plotfun = "stat", stat= "mean")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#mean and SD
pp_check(mod_2, plotfun = "stat_2d", stat = c("mean", "sd"))
```




```
#median posterior predictions
org_work <- org_work %>%
  mutate(mod_2_fitted = fitted(mod_2))
```

Task 5.3

Extract the draws from the posterior regression parameter distributions from **mod_2** with **as_tibble()** and save them as **mod_2_post**. Create a plot using **mcmc_areas()**. Do the following to create the plot:

1. call **mcmc_areas()** and set data to **mod_2_post**, select the **genderFemale** parameter, and set the interval to **0.89**;
2. call **ggtitle()** and set the *title* of the plot to **Posterior Distribution for Gender**.

Question 5.3: What do you learn from examining the *posterior distribution* of the *gender* regression coefficient?

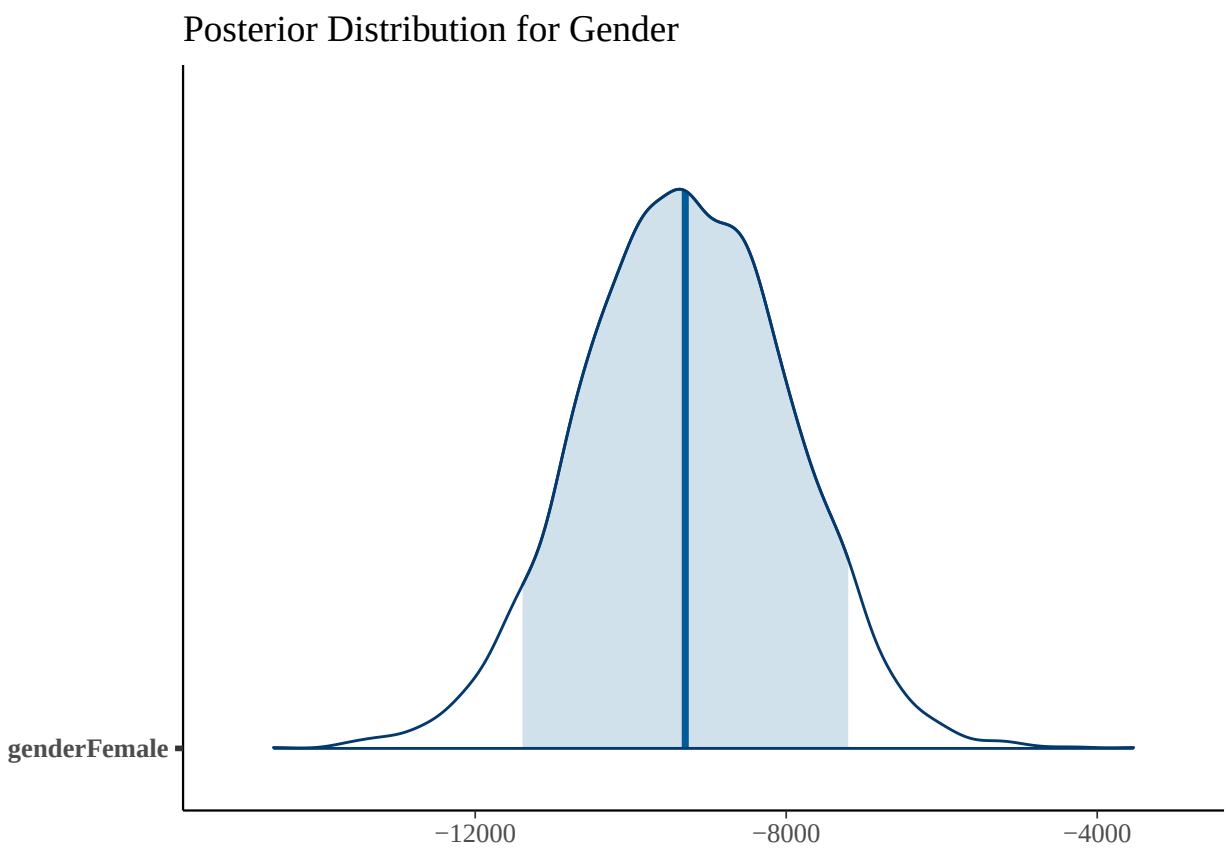
Response 5.3: *The posterior distribution of the gender regression coefficient shows us that again even in the 100th percentile women make less than men. Further confirming there is in fact a gender pay gap. .*

```
mod_2_post <- as_tibble(mod_2)

mod_2_post
```

```
## # A tibble: 4,000 x 4
```

```
##    `(Intercept)` genderFemale   age  sigma
##            <dbl>        <dbl> <dbl>  <dbl>
##  1        64600.       -8318.  980. 20746.
##  2        67248.      -10405.  919. 20872.
##  3        62677.       -8027. 1022. 20613.
##  4        68711.       -9218.  884. 20418.
##  5        66700.      -10569.  949. 20557.
##  6        64272.       -8183.  962. 20501.
##  7        68146.       -9094.  909. 20642.
##  8        63105.       -7646. 1021. 20456.
##  9        64425.       -8797.  972. 21566.
## 10        68125.      -10139.  913. 20113.
## # ... with 3,990 more rows
```

```
#posterior parmeter distributions
mcmc_areas(
  mod_2_post,
  pars = vars(genderFemale),
  prob = 0.89
)+

  ggtitle("Posterior Distribution for Gender")
```



Task 5.4

Calculate the *fitted* and *predicted* values from the *posterior draws* of the regression parameters where you set **gender** equal to **c("Male", "Female")** (i.e., **gender = c("Male", "Female")**) and **age** equal to **seq(25, 55, 5)** (i.e., **age = seq(25, 55, 5)**) inside of **crossing()**. Pass the **crossing()** result to **add_fitted_draws()** and **add_predicted_draws()** while including **mod_2** as an input as well for the *fitted* and *predicted* values saving the results as **mod_2_post_fit** and **mod_2_post_pred**, respectively.

Create a plot named **mod_2_plot** using **ggplot()**. To create the plot, do the following:

1. inside of **ggplot()**, set data to **org_work**, **age** to the *x-axis*, and **total_pay** to the *y-axis*;
2. add a **geom_jitter()** layer with **height** and **width** set to **0.05**, set data to **mod_2_post_pred**, map **age** to the *x-axis*, **.prediction** to the *y-axis*, and **color** to **gender**, set **size** to **0.5**, and set **alpha** to **0.05**;
3. add a **stat_lineribbon()** layer with data seto to **mod_2_post_fit**, map **.value** to the *y-axis* and **gender** to **fill**, set **alpha** to **0.5**, and set **width** to **0.9**;
4. add a **scale_color_manual()** layer with **values** set to **blue** and **red**;
5. add a **scale_fill_brewer()** layer with **palette** set to **Dark2**;
6. call **scale_y_continuous** and set the number of breaks to **10** and the **labels** to *dollar format*;
7. label the axes and legend appropriately with **labs()**;
8. set the *theme* to **theme_economist_white()**.

Display **mod_2_plot** by typing its name.

Questions 5.4: Answer these questions: (1) Do *men* and *women* earn more when they are *younger* or *older*? (2) Does the plot confirm the *constant difference in pay* between *men* and *women* implicated by the regression model?

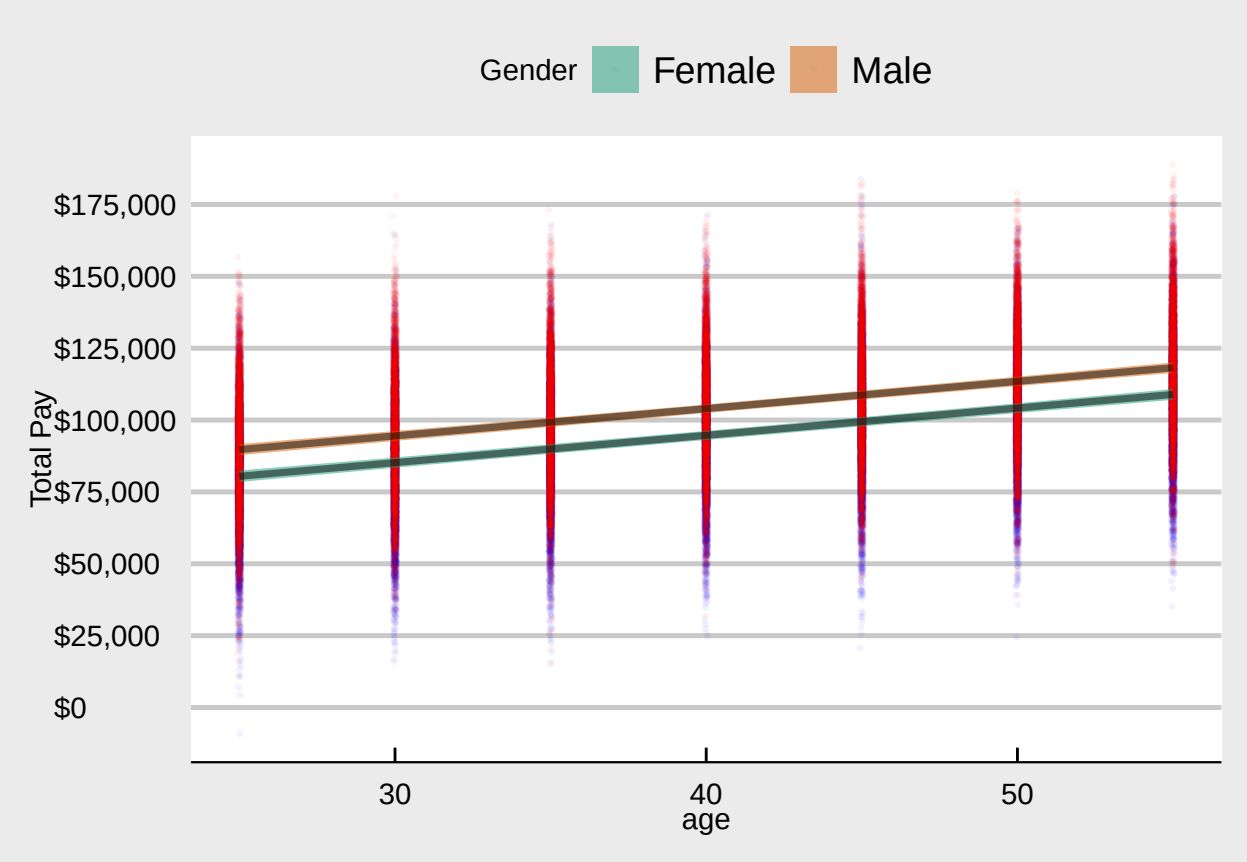
Responses 5.4: *Men and women both earn more the older they are. The plot does confirm that ther is a difference regardless of age in pay between men and women. Women pay increases with age similar to men but it never reaches the amount men make..*

```
mod_2_post_fit <- crossing(
  gender = c("Male", "Female"),
  age = seq(25, 55, 5)
) %>%

  add_fitted_draws(mod_2)
```

```
## Instead of posterior_linpred(..., transform=TRUE) please call posterior_epred(), which provides equi
```

```
mod_2_post_pred <- crossing(
  gender = c("Male", "Female"),
   age = seq(25, 55, 5)
) %>%
  add_predicted_draws(mod_2)

mod_2_plot <- ggplot(
  org_work,
  aes(x = age, y = total_pay)
)+
  geom_jitter(
    height = 0.05, width = 0.05,
    data = mod_2_post_pred,
```

```
    mapping = aes(x = age, y = .prediction, color = gender),
    size = 0.5, alpha = 0.05
  )+
  stat_lineribbon(
    data= mod_2_post_fit,
    mapping = aes(y = .value, fill = gender),
    alpha = 0.5, .width = 0.9
  )+
  scale_color_manual(values = c("blue", "red")) +
  scale_fill_brewer(palette = "Dark2") +
  scale_y_continuous(n.breaks = 10, labels = scales::dollar_format()) +
  labs(
    x = "age", y = "Total Pay",
    fill = "Gender", color = "Gender"
  )+
  theme_economist_white()

mod_2_plot
```



Task 6: Fit Moderated Bayesian Regression Model

Estimate a simple Bayesian regression model using using **org_work** where observed values of **total_pay** are predicted from observed values of **gender** and **age** and their interaction.

Task 6.1

Create a model object named **mod_3** using **stan_glm()**. Inside of **stan_glm()**, do the following:

1. specify the *formula* to indicate **total_pay** predicted by **gender** and **age** and their interaction,
2. specify the *data* as **org_work**,
3. set the *prior intercept* to a **normal** distribution with **location** equal to **1e+05** and **autoscale** equal to **TRUE**,
4. set the *prior of the regression coefficients for the predictors* to a **normal** distribution with **location** equal to **0** and **autoscale** equal to **TRUE**,
5. set the *error prior* to be an **exponential** distribution with **rate** equal to **0.8** and **autoscale** equal to **TRUE**, and
6. set the **seed** to **1805** (birth year of *William Rowan Hamilton*).

After creating the model, then do the following:

1. apply **summary()** to **mod_3** using *three digits* and **seq(0.1, 0.9, 0.2)** for the *percentiles* to examine the posterior parameter distributions,
2. apply **coef()** to **mod_3** to extract the *median* regression coefficients, and
3. apply **posterior_interval** to **mod_3** and set the *credible interval* to **0.85**.

Questions 6.1: Answer these questions: (1) What is the *30th percentile* of the *interaction term* (i.e., **genderFemale:age**) regression coefficient? (2) By how much does the *median regression coefficient* for *age* differ between *men* and *women*? (3) According to the estimated posterior distribution, the *regression coefficient* for *age* differs between *women* and *men* with *85%* probability in what interval?

Responses 6.1: *The 30th percentile of the interaction term is -131.502. The median regression coeffiecent for age differs between men and women by 84.158 . With 85% probability age differs between men and women for age from 895.7023 to 1069.76823. .*

```
mod_3 <- stan_glm(
  total_pay ~ gender * age,
  data = org_work,
  prior_intercept = normal(location = 1e+05, autoscale = TRUE),
  prior = normal(location = 0, autoscale = TRUE),
  seed = 1805
)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.000103 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 1.03 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
```

```
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.43968 seconds (Warm-up)
## Chain 1:                0.158041 seconds (Sampling)
## Chain 1:                1.59772 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 2.3e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.23 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.984045 seconds (Warm-up)
## Chain 2:                0.175833 seconds (Sampling)
## Chain 2:                1.15988 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 2.4e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.24 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
```

```
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.805571 seconds (Warm-up)
## Chain 3:                0.187077 seconds (Sampling)
## Chain 3:                0.992648 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.5e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 1.27408 seconds (Warm-up)
## Chain 4:                0.190467 seconds (Sampling)
## Chain 4:                1.46455 seconds (Total)
## Chain 4:
```

```
summary(mod_3, digits = 3, probs = seq(0.1, 0.9, 0.2))
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      total_pay ~ gender * age
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 1000
##  predictors:   4
##
## Estimates:
##                    mean       sd         10%        30%        50%
## (Intercept)       64552.104   2677.093  61214.808  63136.677  64528.022
## genderFemale      -5814.564   4003.562 -10934.211  -7929.721  -5799.520
## age                 984.350     60.909    905.645    952.342    984.424
## genderFemale:age    -83.203     91.147   -200.356   -131.502    -84.158
## sigma             20814.869    456.019  20236.031  20578.605  20809.810
##                    70%        90%
## (Intercept)       65956.872  68045.258
```

```
## genderFemale      -3677.445   -620.378
## age                1018.404   1060.669
## genderFemale:age    -37.304     34.451
## sigma             21038.229  21395.797
##
## Fit Diagnostics:
##            mean       sd         10%        30%        50%        70%
## mean_PPD 100946.107    928.704  99773.245 100455.917 100928.947 101420.503
##            90%
## mean_PPD 102149.351
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##                  mcse    Rhat    n_eff
## (Intercept)       69.838   1.003 1469
## genderFemale     114.310   1.003 1227
## age                1.528   1.002 1589
## genderFemale:age   2.606   1.003 1223
## sigma              8.303   0.999 3017
## mean_PPD          15.692   0.999 3503
## log-posterior      0.038   1.001 1680
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

```
coef(mod_3)
```

```
##      (Intercept)     genderFemale              age genderFemale:age
##      64528.02231      -5799.51964        984.42357        -84.15836
```

```
posterior_interval(mod_3, prob = 0.85)
```

```
##                        7.5%       92.5%
## (Intercept)       60736.9489 68480.22125
## genderFemale     -11612.8885   -51.48265
## age                895.7023  1069.76823
## genderFemale:age   -213.7967    49.63355
## sigma             20170.0239 21477.23807
```

Task 6.2

Examine **mod_3** by doing the following:

1. compute the *Bayesian R-squared* using **bayes_R2()** and saving the results to an object named **mod_3_r_sq**;
2. apply **summary()** to **mod_3_r_sq**;
3. apply **loo()** to **mod_3** and save the result to an object named **mod_3_loo**;
4. apply **loo_compare()** to **mod_1_loo**, **mod_2_loo**, and **mod_3_loo**;
5. apply **pp_check()** and examine the *density overlay*, the *mean* on its own, and the *mean* and *sd* together;
6. compute the *fitted values* using the *median posterior regression parameters* and save the caluclation as a new variable named **mod_3_fitted** to **org_work**;

7. open the *spreadsheet view* of **org__work** to answer a question about **mod__3__fitted** values.

Questions 6.2: Answer these questions: (1) What is the *third quartile R-squared* value? (2) Using the *leave-one-out* comparison, do **mod__3** or **mod__2** fit the data differently with any great practical significance? (3) Do the *posterior predictive checks* indicate any estimation issues? (4) How come the *median posterior fitted values* for the individuals in **org__work** are quite similar between **mod__2__fitted** and **mod__3__fitted**?

Responses 6.2: *The third quartile r-sqaured value is 0.3337. Mod3 and Mod2 are not significantly better than each other. The posterior predictive checks does not indicate any estimation issues. The median posterior fitted values for Mod2 fitted and Mod3 fitted because the interaction effects are not that big/strong. Therefore the fitted values and predictions will be similar to each other..*

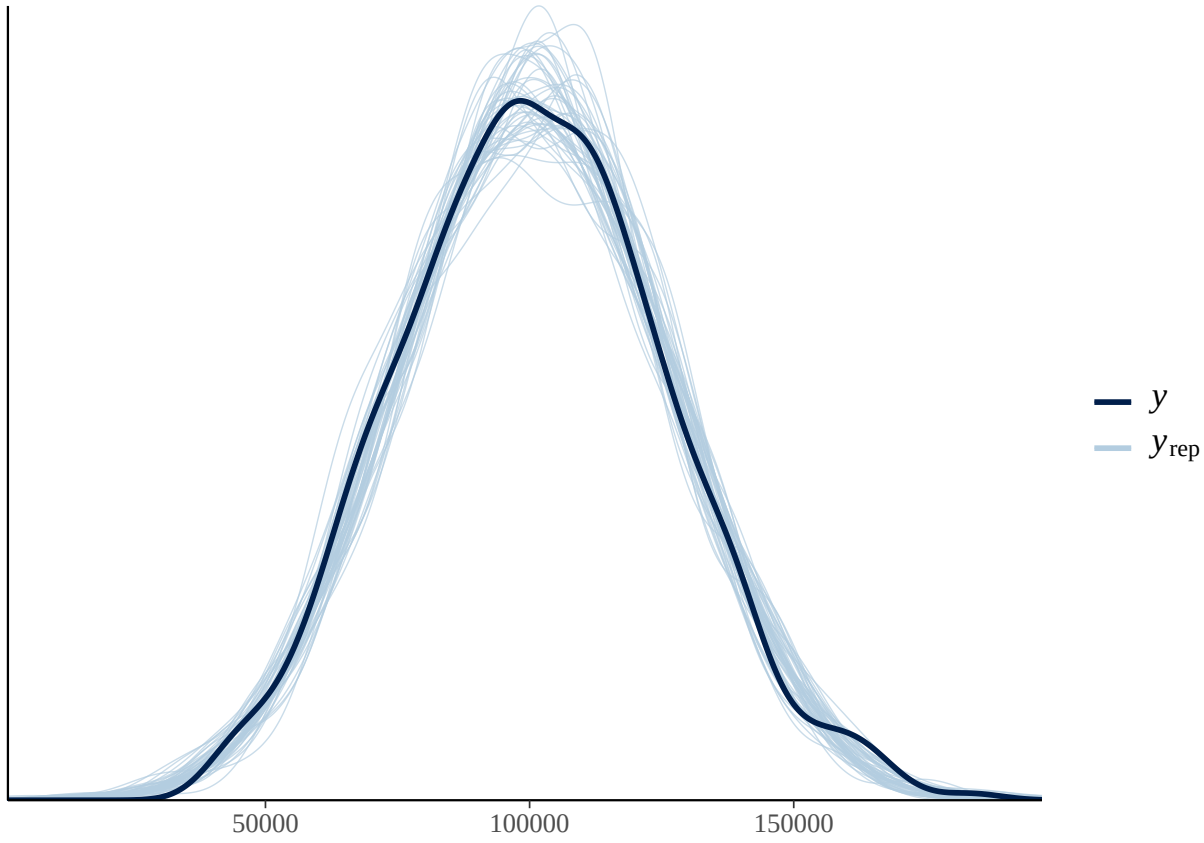
```
mod_3_r_sq <- bayes_R2(mod_3)

summary(mod_3_r_sq)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2192  0.3041  0.3187  0.3188  0.3337  0.3891
```

```
#loo
mod_3_loo <- loo(mod_3)

loo_compare(mod_1_loo, mod_2_loo, mod_3_loo)
```

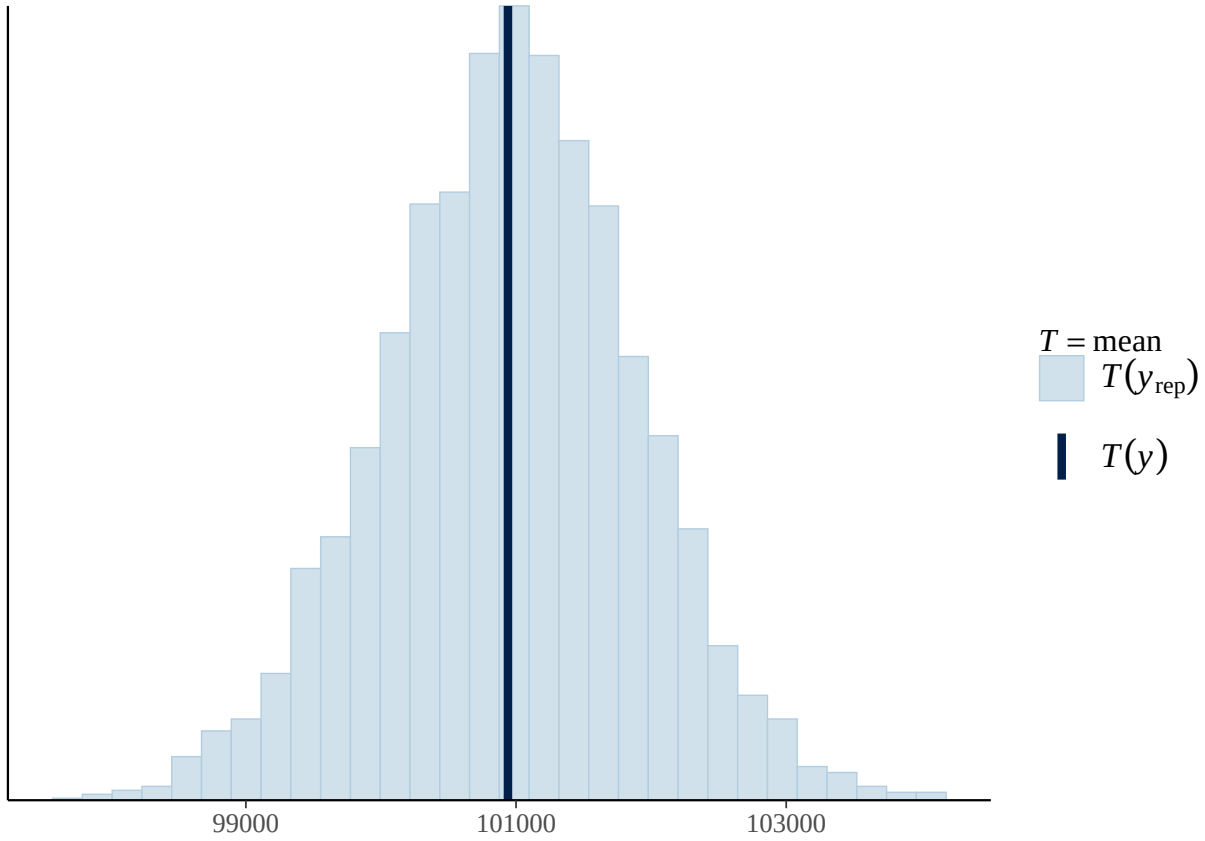
```
##       elpd_diff se_diff
## mod_2    0.0       0.0
## mod_3   -0.5       0.9
## mod_1 -176.6      16.0
```

```
##pp_checks
pp_check(mod_3, plotfun = "dens_overlay")
```

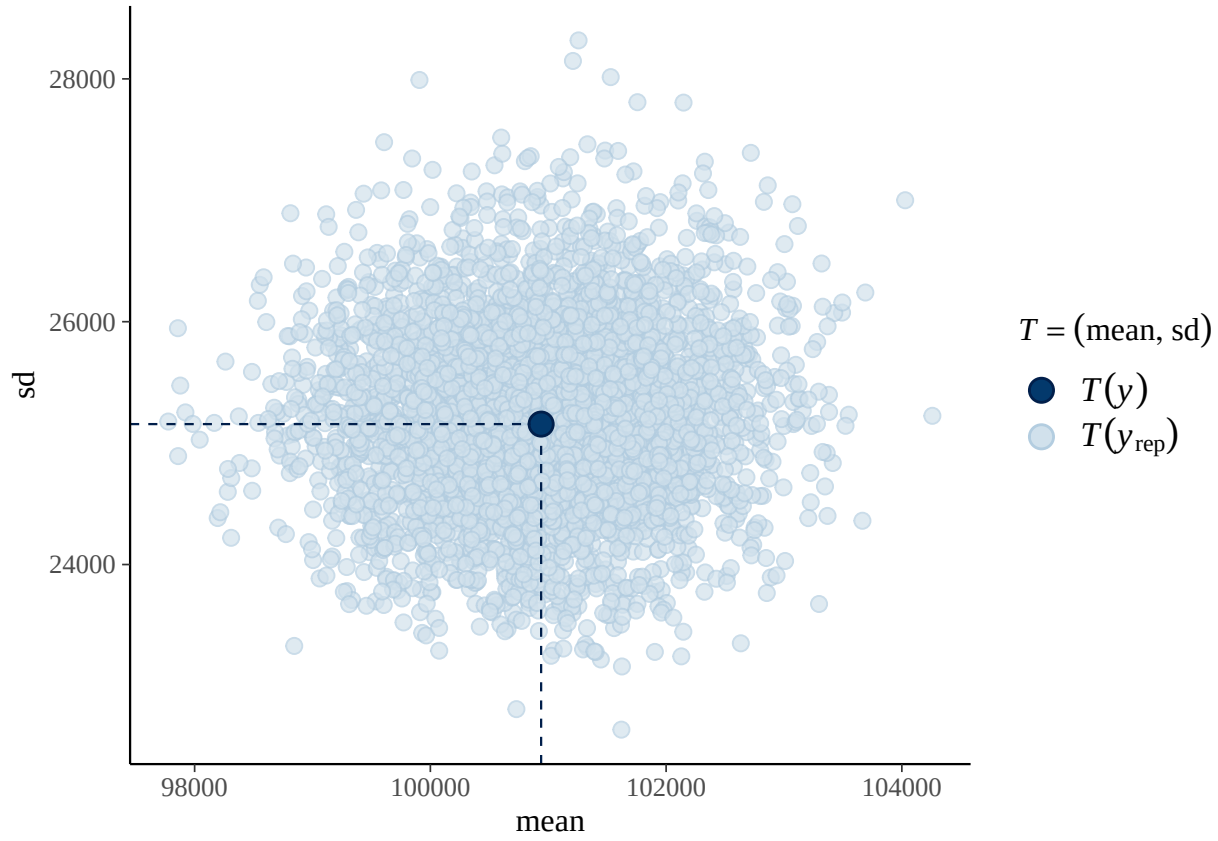



```
pp_check(mod_3, plotfun = "stat", stat = "mean")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
pp_check(mod_3, plotfun = "stat_2d", stat = c("mean", "sd"))
```




```
##median posterior draws
org_work <- org_work %>%
  mutate(mod_3_fitted = fitted(mod_3))

mod_3_post <- as_tibble(mod_3)

mod_3_post
```

```
## # A tibble: 4,000 x 5
##    `(Intercept)` genderFemale   age `genderFemale:age`  sigma
##            <dbl>        <dbl> <dbl>              <dbl>  <dbl>
##  1        63739.       -4106. 1004.              -110.  20650.
##  2        61396.       -3928. 1045.              -115.  21232.
##  3        63805.       -5271.  984.               -81.9 20249.
##  4        62986.       -4594. 1014.              -124.  20724.
##  5        67189.       -7144.  955.               -57.3 20449.
##  6        67344.       -7814.  950.               -36.2 20363.
##  7        63798.       -6898.  999.               -75.6 20293.
##  8        62491.       -5660. 1016.               -84.2 20193.
##  9        68983.      -14704.  894.               111.  21267.
## 10        64145.       -8485.  963.                16.5 20642.
## # ... with 3,990 more rows
```

Task 6.3

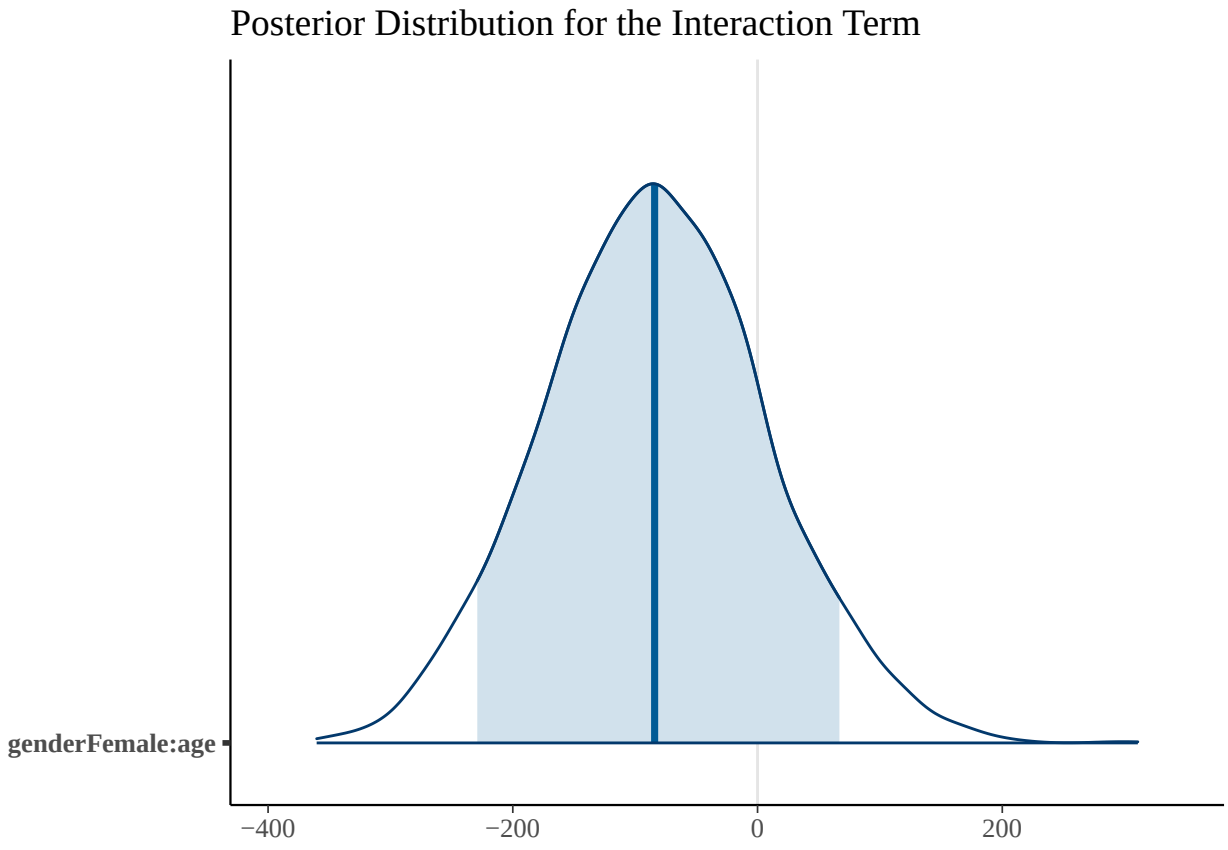
Extract the draws from the posterior regression parameter distributions from **mod_3** with **as_tibble()** and save them as **mod_3_post**. Create a plot using **mcmc_areas()**. Do the following to create the plot:

1. call **mcmc_areas()** and set data to **mod_3_post**, select the parameter the *interaction term* parameter (i.e., **genderFemale:age**), and set the interval to **0.89**;
2. call **ggtitle()** and set the *title* of the plot to **Posterior Distribution for the Interaction Term**.

Question 6.3: What do you learn from examining the *posterior distribution* of the *interaction term* regression coefficient?

Response 6.3: *The posterior distribution of the interaction between females and age is predomintaly negative because it is mostly below zero. However, the distribution does include the value of zero..*

```
mcmc_areas(
  mod_3_post,
  pars = vars("genderFemale:age"),
  prob = 0.89
)+
  ggtitle("Posterior Distribution for the Interaction Term")
```



Task 6.4

Calculate the *fitted* and *predicted* values from the *posterior draws* of the regression parameters where you set **gender** equal to **c("Male", "Female")** (i.e., **gender = c("Male", "Female")**) and **age** equal

to **seq(25, 55, 5)** (i.e., **age = seq(25, 55, 5)**) inside of **crossing()**. Pass the **crossing()** result to **add_fitted_draws()** and **add_predicted_draws()** while including **mod_3** as an input as well for the *fitted* and *predicted* values saving the results as **mod_3_post_fit** and **mod_3_post_pred**, respectively.

Create a plot named **mod_3_plot** using **ggplot()**. To create the plot, do the following:

1. inside of **ggplot()**, set data to **org_work**, **age** to the *x-axis*, and **total_pay** to the *y-axis*;
2. add a **geom_jitter()** layer with **height** and **width** set to **0.05**, set data to **mod_3_post_pred**, map **age** to the *x-axis*, **.prediction** to the *y-axis*, and **color** to **gender**, set **size** to **0.5**, and set **alpha** to **0.05**;
3. add a **stat_lineribbon()** layer with data seto to **mod_3_post_fit**, map **.value** to the *y-axis* and **gender** to **fill**, set **alpha** to **0.5**, and set **width** to **0.9**;
4. add a **scale_color_manual()** layer with **values** set to **purple** and **orange**;
5. add a **scale_fill_brewer()** layer with **palette** set to **Dark2**;
6. call **scale_y_continuous** and set the number of breaks to **10** and the **labels** to *dollar format*;
7. label the axes and legend appropriately with **labs()**;
8. set the *theme* to **theme_hc()**.

Display **mod_3_plot** by typing its name.

Question 6.4: Is the *interaction effect* noticeable in the plot?

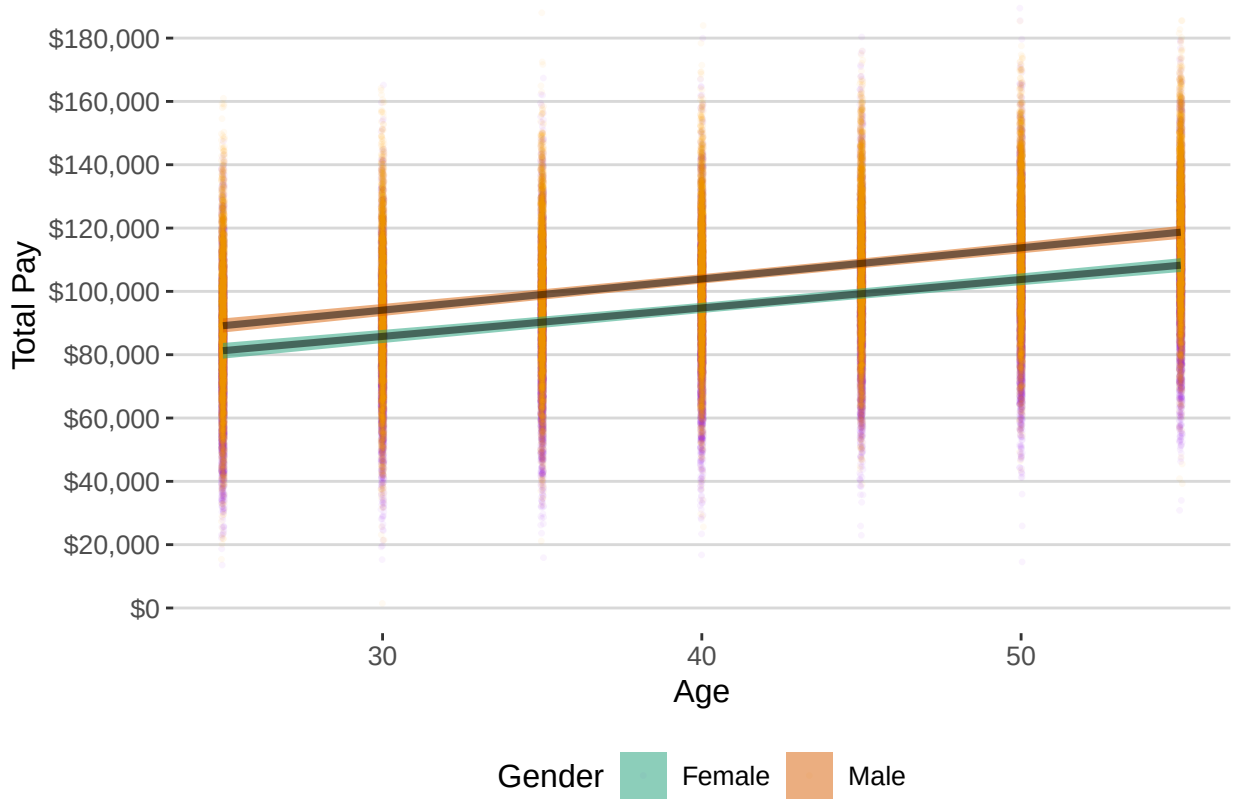
Response 6.4: *The interaction effect is not noticeable in this plot. The lines appera to be parallel.*

```
##fitted draws
mod_3_post_fit <- crossing(
  gender = c("Male", "Female"),
  age = seq(25, 55, 5)
) %>%
  add_fitted_draws(mod_3)
```

```
## Instead of posterior_linpred(..., transform=TRUE) please call posterior_epred(), which provides equi
```

```
##predictive draws
mod_3_post_dist <- crossing(
  gender = c("Male", "Female"),
  age = seq(25, 55, 5)
) %>%
  add_predicted_draws(mod_3)

##ggplot
mod_3_plot <- ggplot(
  org_work,
  aes(x = age, y = total_pay)
)+
  geom_jitter(
    height = 0.05, width = 0.05,
    data = mod_3_post_dist,
    mapping = aes( x = age, y = .prediction, color = gender),
    size = 0.5, alpha = 0.05
  )+
  stat_lineribbon(
    data = mod_3_post_fit,
    mapping =  aes(y = .value, fill = gender),
```

```
    alpha = 0.5, .width = 0.9
  )+
  scale_color_manual(values = c("purple", "orange")) +
  scale_fill_brewer(palette = "Dark2") +
  scale_y_continuous(n.breaks = 10, labels = scales::dollar_format())+
  labs(
    x = "Age", y = "Total Pay",
    color = "Gender", fill = "Gender"
  )+

  theme_hc()

mod_3_plot
```



Task 7: Save Plots and Data

For this task, you will save the plots and the working data.

Task 7.1

Save the working data, **org_work** as the data file: **org_work.csv** in the **data** folder of the project directory using **write_csv()**.

Save the three plot objects as **png** files in the **plots** folder of the project directory. Save **mod_1_plot** as **mod_1.png**, **mod_2_plot** as **mod_2.png**, and **mod_3_plot** as **mod_3.png**. Use a width of *9 inches* and height of *6 inches* for all plots.

```
write_csv(
  org_work,
  path = here("data", "org_work_samp.csv")
)
```

```
## Warning: The ‘path‘ argument of ‘write_csv()‘ is deprecated as of readr 1.4.0.
## Please use the ‘file‘ argument instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_warnings()‘ to see where this warning was generated.
```

```
#mod_1_plot
ggsave(
  here("plots", "mod_1.png"),
  plot = mod_1_plot,
  units = "in", width = 9, height = 6
)

#mod_2_plot
ggsave(
  here("plots", "mod_2.png"),
  plot = mod_2_plot,
  units = "in", width = 9, height = 6
)


#mod_3_plot
ggsave(
  here("plots", "mod_3.png"),
  plot = mod_3_plot,
  units = "in", width = 9, height = 6
)
```

Task 8: Conceptual Questions

For your last task, you will respond to conceptual questions based on the conceptual lectures for this week.

Question 8.1: What is the difference between the *prior* and *posterior* probability parameter distributions in Bayesian data analysis?

Response 8.1: *Prior probability distribution is observing different parameters of data before accounting for data. The posterior probability distribution is observing the parameter values after accounting for data.* .

Question 8.2: What is a Bayesian credible interval?

Response 8.2: *The Bayesian credible interval is* .

Question 8.3: The following frequency table counts how many sales employees had a *profit* or *loss* utilizing three different strategies A, B, and C:

Strategy	Loss	Profit
A	100	175
B	300	75
C	125	225

Compute the following: (1) *prior probability* of *all three strategies*; (2) *likelihood* of a *profit* under *all three strategies*; (3) *posterior probability* of *all three strategies* given a *profit*.

Response 8.3: *The prior probability of a profit under strategy A is 27.5 percent. The prior probability of a profit under strategy B is 37.5 percent. The prior probability of a profit under strategy C is 35 percent.*

The likelihood of a profit under strategy A is 63.6 percent. The likelihood of a profit under strategy B is 20 percent. The likelihood of a profit under strategy C is 51.1 percent..

The posterior probabiliy of strategy A is 36.8 percent. The posterior probabiliy of strategy B is 15.8 percent. The posterior probabiliy of strategy C is 37.6 percent