R_Code_SIEPR, Pryce Davies

Part 1

1.a

```
### Generate 1000 values of Y
set.seed(1)
x_values <- rnorm(1000,0,2)
latent_samples <- .5 * x_values + 1 +revd(1000) ### revd draws random samples from a Ty
pe 1 extreme value distribution (in this case fat right-hand tails) with location 0 and
scale 1
samples <- ifelse(latent_samples >= 0, 1,0)
### from reading, a latent variable model represents a intermediary step, one gets the
 binary outcome variable by assigning values over 0 to 1 and values under 0 to 0
```

1.b

$$\sum_{n=1}^{1000} y_i * \log \frac{e^{\alpha+\beta*x_i}}{1+e^{\alpha+\beta*x_i}} + (1-y_i) * \log 1 - \frac{e^{\alpha+\beta*x_i}}{1+e^{\alpha+\beta*x_i}}$$

1.c

```
### this function returns the log-likelihood value given a set of parameters theta
logli <- function(theta){
y <- samples
x <- x_values
alpha <- theta[1]
beta <- theta[2]
yhat = exp(alpha+beta*x)/(1+exp((alpha+beta*x)))
loglik <- sum(y*log(yhat)+ (1-y)*log(1-yhat))
return(-loglik)
}

par1 <- c(1,.5) ### original parameters
logli(par1)
```

```
## [1] 385.0191
```

```
### Optim does optimization for given function, choose start value of (-5,-5) arbitrari
ly (though it is very unlikely parameter estimates would be lower), robust to different
specifications such as (-1,-1), (0,0)
start <- c(-5,-5)
optim(start, logli)
```

```
## $par
## [1] 2.548044 1.029788
##
## $value
## [1] 302.2713
##
## $counts
## function gradient
##       71       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

1.e

```
### Compute Standard Errors via Bootstrapping (i.e. estimate parameter values many time
s, standard deviation of this large sample represents computed standard errors)
alpha_value = c()
beta_value = c()
for ( i in 1:5000){
x_values <- rnorm(1000,0,2)
latent_samples <- .5 * x_values + 1 +revd(1000)
samples <- ifelse(latent_samples >= 0, 1,0)
start <- c(0,0)
alpha <- optim(start, logli)[[1]][1]
beta <- optim(start, logli)[[1]][2]
alpha_value = append(alpha_value,alpha)
beta_value = append(beta_value,beta)

}

bootstrap_values = data.frame(matrix(nrow=5000, ncol=2))
col_names <- c("alpha", "beta")
colnames(bootstrap_values) <- col_names
bootstrap_values$alpha <- alpha_value
bootstrap_values$beta <- beta_value
mean(bootstrap_values$alpha) ### Will use these as best estimates of parameter values
```

```
## [1] 2.634811
```

```
mean(bootstrap_values$beta)
```

```
## [1] 0.9739044
```

```
sd(bootstrap_values$alpha)  ### These are our standard error estimates of parameter val
ues
```

```
## [1] 0.1456744
```

```
sd(bootstrap_values$beta)
```

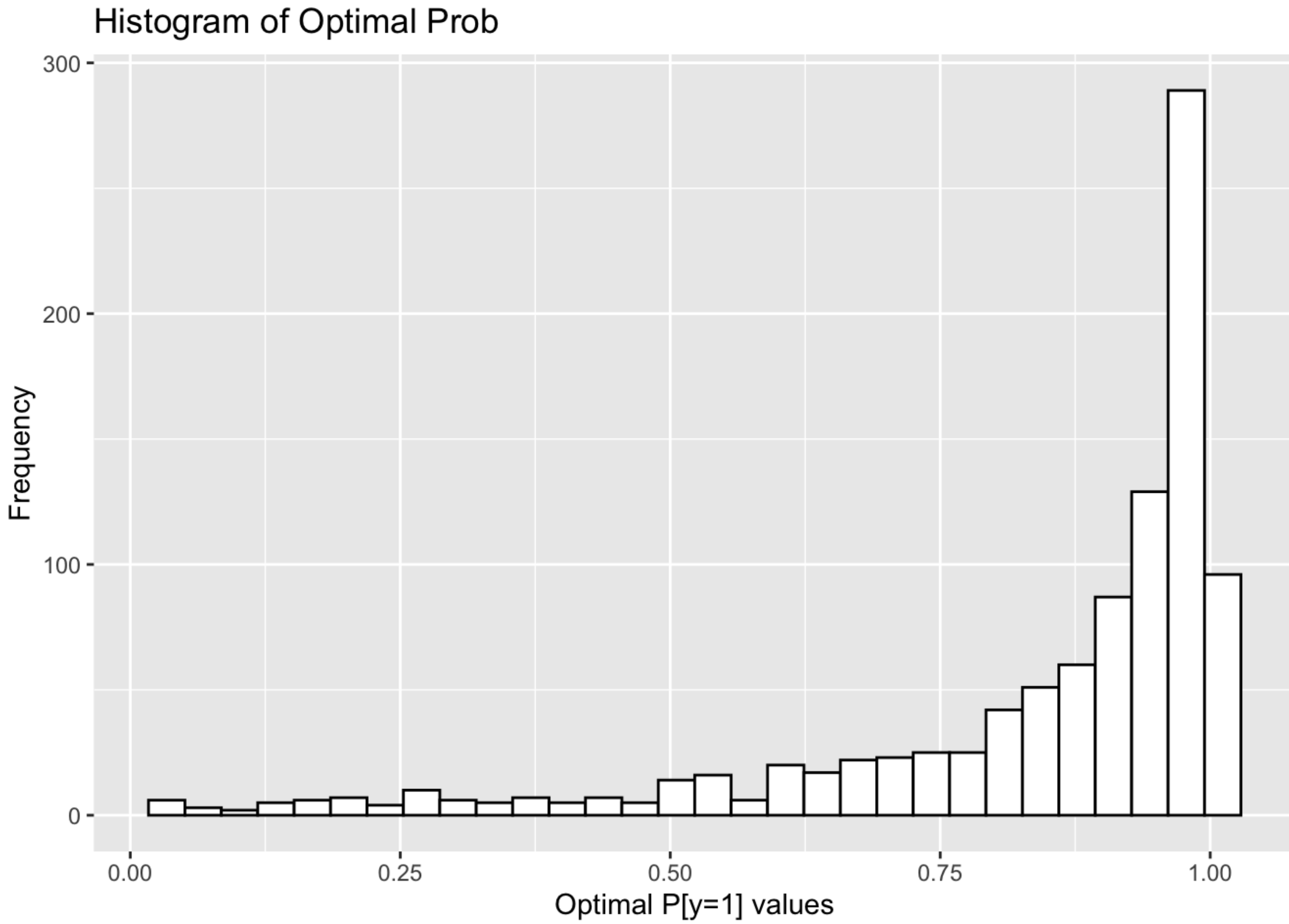
```
## [1] 0.07471862
```

1.f

```
### Function returns estimated values of P[y=1] given parameters and x values
yhat_f <- function(alpha,beta,x) {
yhat = exp(alpha+beta*x)/(1+exp((alpha+beta*x)))
return(yhat)
}

original_p <-yhat_f(1,.5,x_values)
optim_p <- yhat_f(mean(bootstrap_values$alpha),mean(bootstrap_values$beta),x_values)
Results<- data.frame(samples,latent_samples, x_values, original_p, optim_p) ### Datafra
me with x values, latent variable values, binary outcome variables, and P[y=1] for both
our original and optimal parameters
```

```
ggplot(data=Results,(aes(x=optim_p))) + geom_histogram(color= "black", fill = "white")
 + labs(title = "Histogram of Optimal Prob", x= "Optimal P[y=1] values", y = "Frequenc
y",)
```

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




```
mean(Results$optim_p) ### Mean is high and histogram clustered around 1, this makes sen
se as our error distribution is more likely to add high positive values to the latent v
ariable due to its fat right tail, and thus increase the probability of Y=1
```

```
## [1] 0.8423652
```

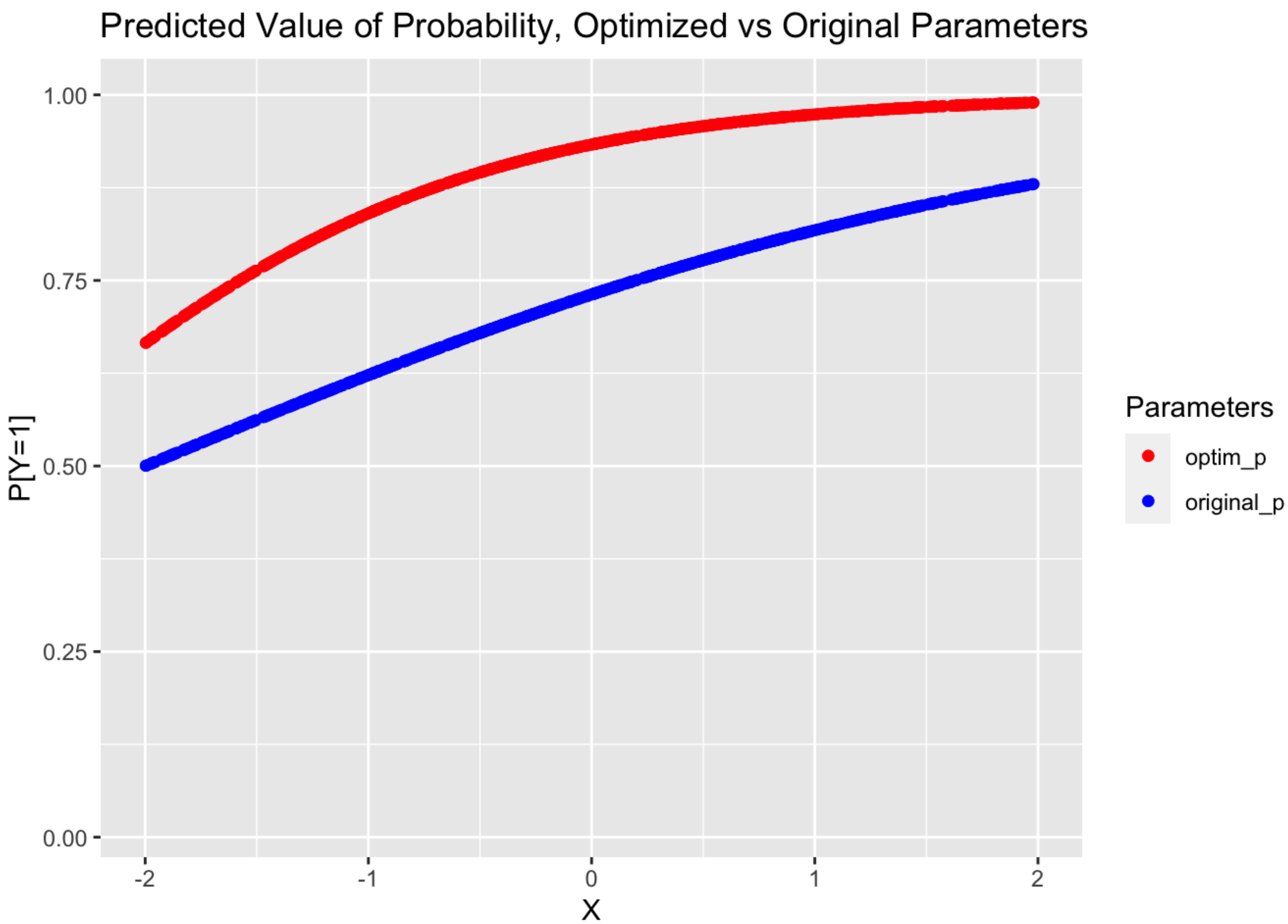
```
ggsave("plot1.f.jpeg")
```

```
## Saving 7 x 5 in image
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

##1.g

```
ggplot() + geom_point(data=Results, aes(x=x_values, y= optim_p, colour = "optim_p")) +
 geom_point(data=Results, aes(x=x_values, y= original_p,colour = "original_p")) + xlim
(-2,2) + labs(title= "Predicted Value of Probability, Optimized vs Original Parameters"
, y="P[Y=1]", x = "X" ) + scale_color_manual(name = "Parameters", values = c("optim_p"
 = "red", "original_p"= "blue"))
```

```
## Warning: Removed 312 rows containing missing values (geom_point).

## Warning: Removed 312 rows containing missing values (geom_point).
```




```
ggsave("Plot 1.g.jpeg")
```

```
## Saving 7 x 5 in image
```

```
## Warning: Removed 312 rows containing missing values (geom_point).

## Warning: Removed 312 rows containing missing values (geom_point).
```

Part 2

2.1

```r
setwd("~/Desktop/assignment_jicuesta_2021")

file_paths <- dir_ls("~/Desktop/assignment_jicuesta_2021/monthly_data")

file_contents<-list()
monthly_data <- data.frame()

### This function does three things: it first reads in a csv in the "monthly data" fold
er. Then it goes through the dates in that file and for dates that are in the format YY
YY-DD-MM format instead of YYYY-MM-DD, it replaces the second part (DD) with the month
 specified in the file name and moves the DD to the third position so that it is in YYY
Y-MM-DD format. It then binds together all of these month,year files into one
for (i in seq_along(file_paths)) {
  file_contents <- read_csv(file_paths[i], col_types = list(col_character(),col_charact
er(), col_number(), col_number(),col_number(), col_character()))
  month = str_split(str_sub(file_paths[i],-7,-5),"_")[[1]][-1]
  month = sprintf("%02s",month)
for (i in 1:100) {
    if (str_split(file_contents$date[i],"-")[[1]][2] != month) {
      split_date = str_split(file_contents$date[i],"-")
      split_2 <- sprintf("%02s",split_date[[1]][2])
      new_date = paste(split_date[[1]][1],month,split_2, sep="-")
    file_contents$date[i] = new_date
  }
}
  monthly_data <- rbind(monthly_data, file_contents)
}

firm_sales <- read_csv("aggregate_firm_sales.csv", col_types = list(col_character(),col
_character(),col_number()))

firm_info <- read_csv("firm_information.csv")
```

```
## Rows: 100 Columns: 4
```

```
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (3): firm_id, firm_name, firm_sector
## dbl (1): treatment_group
```

```
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
### Dataset_1 is used for tables and graphs
firm_sales <- firm_sales %>%
  mutate(firm_id = strtrim(firm_id,7), ID_new = paste(firm_id, date))  ### ID uniquely
 combines firm ID and date- analysis shows it is indeed unique(no missing values when u
sed to merge monthly data and firm sales data)

monthly_data <- monthly_data %>%
  mutate(firm_id = strtrim(firm_id,7),ID_new = paste(firm_id,date), date = as.Date(dat
e))

Dataset_1 <- left_join(firm_sales, monthly_data, by = "ID_new",keep = TRUE )


Dataset_1 <- Dataset_1 %>%
  select(-c(ID_new.y,date.y,firm_id.y,)) %>%
  rename(date = date.x, ID_new = ID_new.x,firm_id = firm_id.x)

Dataset_1 <- merge(Dataset_1,firm_info, by = "firm_id")

Dataset_1 <- Dataset_1 %>%
  mutate(Eligible = as.factor(ifelse(employment_t > 100, 0, 1))) ### Identifies firms w
ith under 100 employees and therefore theoretically eligible for the program

### Dataset_Regression has lagged employment, year and month columns, so that it can be
used in the regression analysis
### choose to include specific columns for year and month for the time-fixed effects sp
ecification. This is because for the time -effects on the outcome, it is more intuitive
for the outcomes to be related to the year and/or month of the outcome, rather than the
specific year-month-day.
### overall Quarter (quarter relative to starting quarter) included for Binning analysi
s conducted below

Dataset_Regression <- data.frame()
firm_path = unique(Dataset_1$firm_id)

for (i in firm_path) {
  temp <- Dataset_1 %>%
    filter(firm_id == i) %>%
    mutate(employment_t_lag = lag(employment_t),
           date = as.Date(date),
           adopt_t = as.numeric(adopt_t),
           n_eligible_post = as.factor(ifelse(employment_t > 100 & date >= as.Date("201
3-01-01"),1,0)),
           year = as.character(format(date, format = "%Y")),
           month = as.character(format(date, format = "%m")),
           quarter = 4 *(as.numeric(year) - 2010) + as.numeric(quarter(date)))
  Dataset_Regression = rbind(Dataset_Regression,temp)
}
```

2.2

```
### Each table summarizes avg sales, revenue wrt wages (to somewhat control for firm si
ze), and sector grouped by the relative indicator (Eligigle, Adopt, etc. ) in month pro
gram began for firms of roughly similar size
eligible <- Dataset_1 %>%
  filter(grepl("2013-01", date) == TRUE) %>%
  filter(employment_t >= 60 &  employment_t <= 140 ) %>%
  group_by(Eligible)

adopt <- Dataset_1 %>%
  filter(grepl("2013-01", date) == TRUE ) %>%
  filter(employment_t >= 60 &  employment_t <= 140 ) %>%
  group_by(adopt_t)

eligible_adopt <- Dataset_1 %>%
  filter(grepl("2013-01", date) == TRUE ) %>%
  filter(employment_t >= 60 &  employment_t <= 140, Eligible == 1 ) %>%
  group_by(adopt_t)



eligible_table <- eligible %>%
  summarise(
Number_of_Firms = length(unique(firm_id)),
Revenue_vs_Wages = as.integer(mean(revenue_t/wage_bill_t)),
Num_Employees = as.integer(mean(employment_t)),
Month_Sales = as.integer(mean(sales_t)),
Hospitality = sum(as.numeric(firm_sector == "Hospitality"))/Number_of_Firms,
Retail = sum(as.numeric(firm_sector == "Retail"))/Number_of_Firms,
Fast_Food = sum(as.numeric(firm_sector == "Fast Food"))/Number_of_Firms)

 adopt_table <- adopt %>%
  summarise(
Number_of_Firms = length(unique(firm_id)),
Revenue_vs_Wages = as.integer(mean(revenue_t/wage_bill_t)),
Num_Employees = as.integer(mean(employment_t)),
Month_Sales = as.integer(mean(sales_t)),
Hospitality = sum(as.numeric(firm_sector == "Hospitality"))/Number_of_Firms,
Retail = sum(as.numeric(firm_sector == "Retail"))/Number_of_Firms,
Fast_Food = sum(as.numeric(firm_sector == "Fast Food"))/Number_of_Firms)

eligible_adopt_table <- eligible_adopt %>% ### compares ELigible firms (100 or less Emp
loyees) by adoption status
  summarise(
Number_of_Firms = length(unique(firm_id)),
Revenue_vs_Wages = as.integer(mean(revenue_t/wage_bill_t)),
Num_Employees = as.integer(mean(employment_t)),
Month_Sales = as.integer(mean(sales_t)),
Hospitality = sum(as.numeric(firm_sector == "Hospitality"))/Number_of_Firms,
Retail = sum(as.numeric(firm_sector == "Retail"))/Number_of_Firms,
Fast_Food = sum(as.numeric(firm_sector == "Fast Food"))/Number_of_Firms)
```

```
### Display
formattable(eligible_table)
```

Eligible	Number_of_Firms	Revenue_vs_Wages	Num_Employees	Month_Sales	Hospitality	Retail	F
0	22	585	119	733433	0.09090909	0.2272727	(
1	28	306	81	235667	0.14285714	0.5357143	(

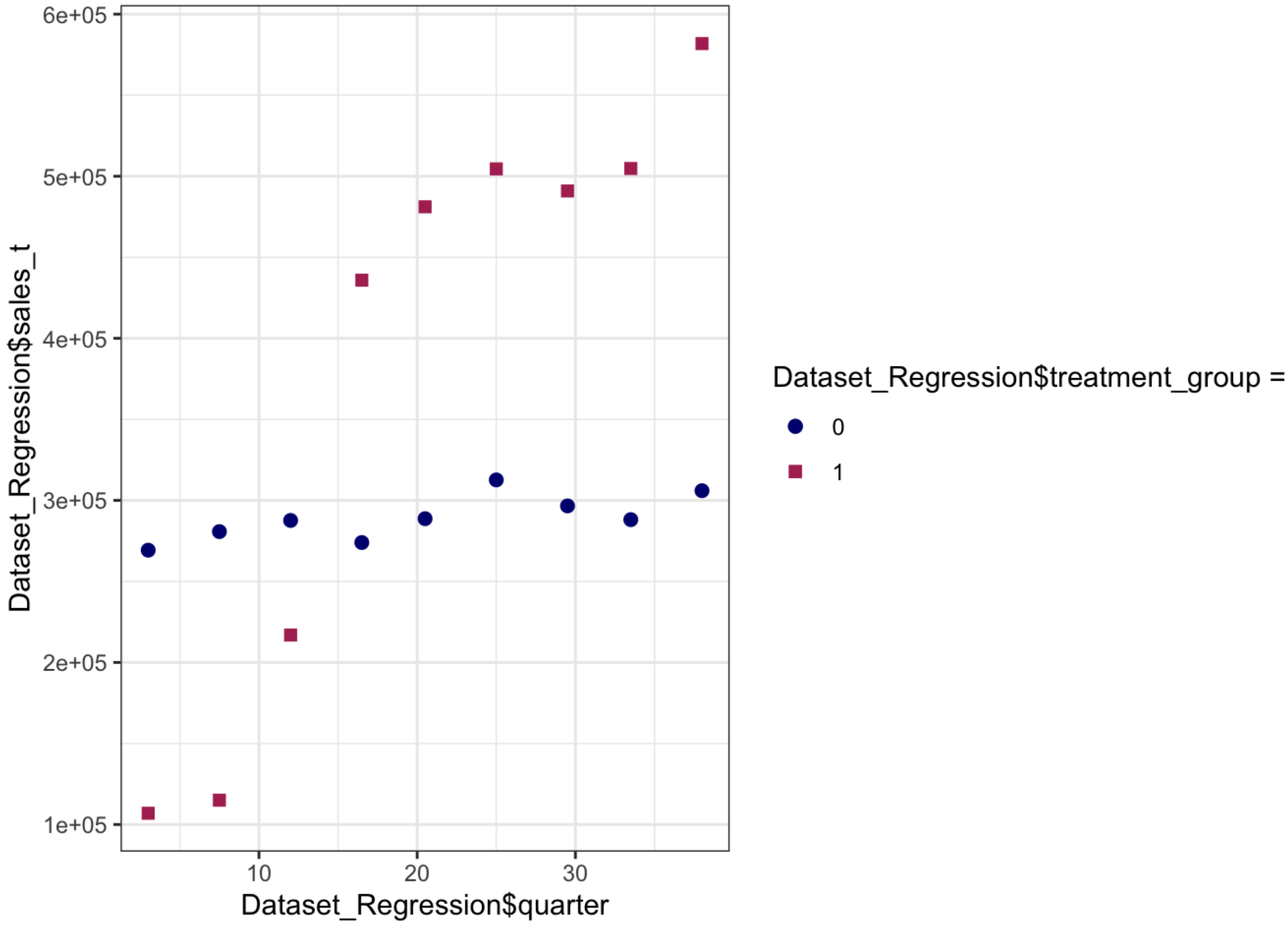
```
formattable(adopt_table)
```

adopt_t	Number_of_Firms	Revenue_vs_Wages	Num_Employees	Month_Sales	Hospitality	Retail	F
0	32	510	105	511014	0.1250000	0.3125000	(
1	18	285	85	354542	0.1111111	0.5555556	(

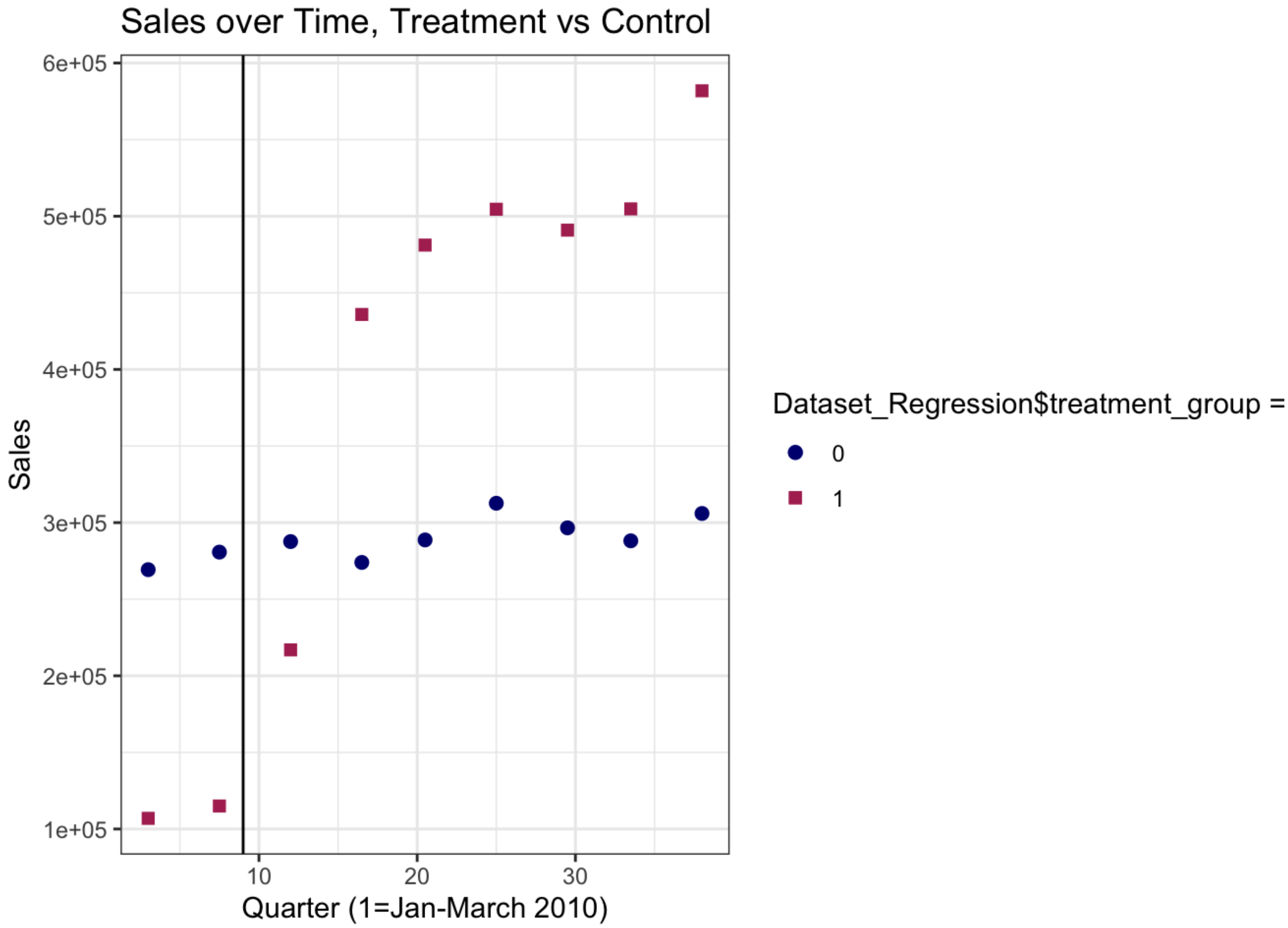
```
formattable(eligible_adopt_table)
```

adopt_t	Number_of_Firms	Revenue_vs_Wages	Num_Employees	Month_Sales	Hospitality	Retail	F
0	11	324	77	268369	0.1818182	0.5454545	(
1	17	295	83	214507	0.1176471	0.5294118	(

```
### Treatment vs Control
Bin_sales_adopt <- binsreg(Dataset_Regression$sales_t, Dataset_Regression$quarter, by =
Dataset_Regression$treatment_group, binspos = "es", samebinsby = TRUE)
```



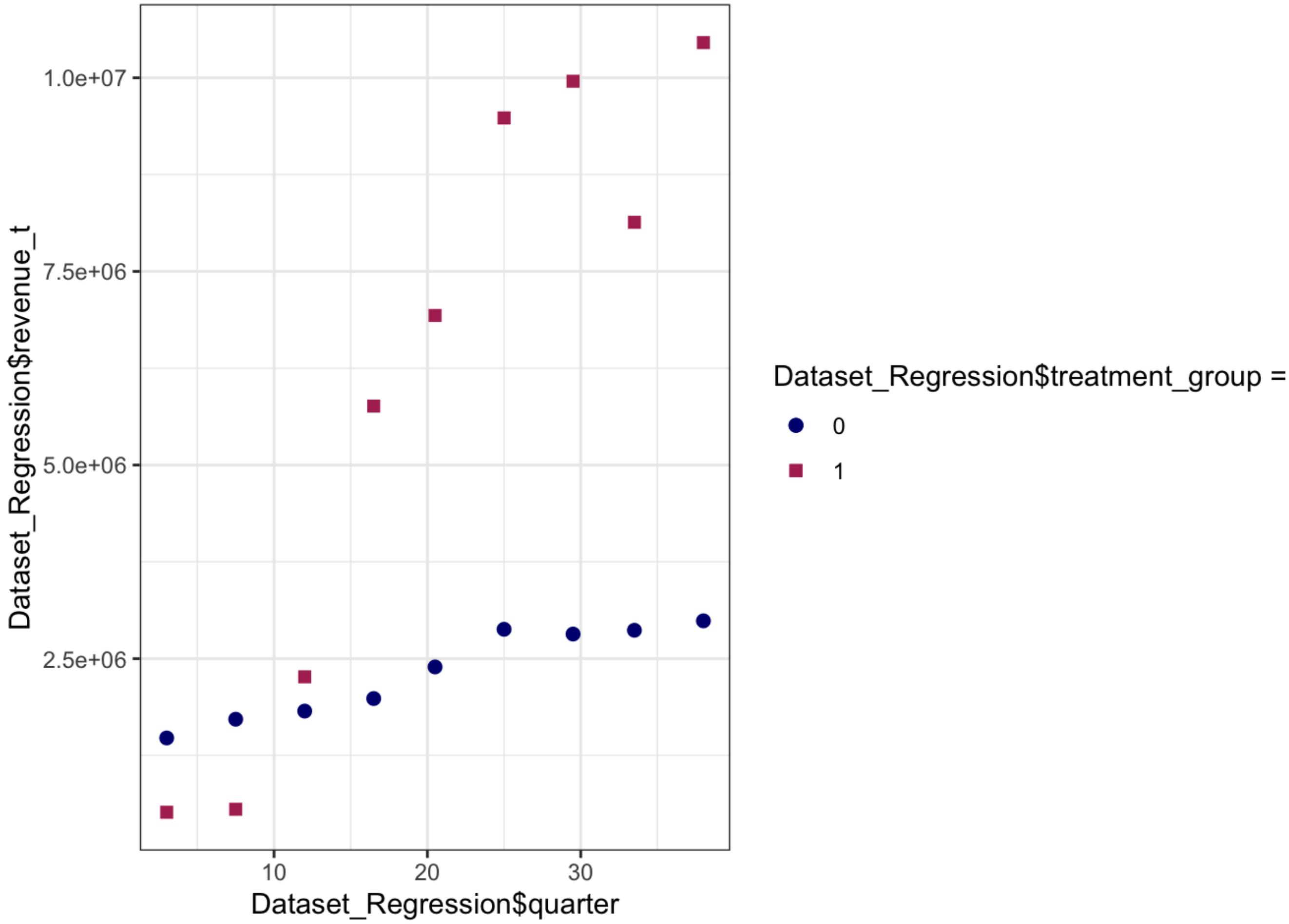

```
Bin_sales_adopt$bins_plot + geom_vline( xintercept = 9) + labs(title = "Sales over Tim
e, Treatment vs Control", x= "Quarter (1=Jan-March 2010)", y= "Sales")
```



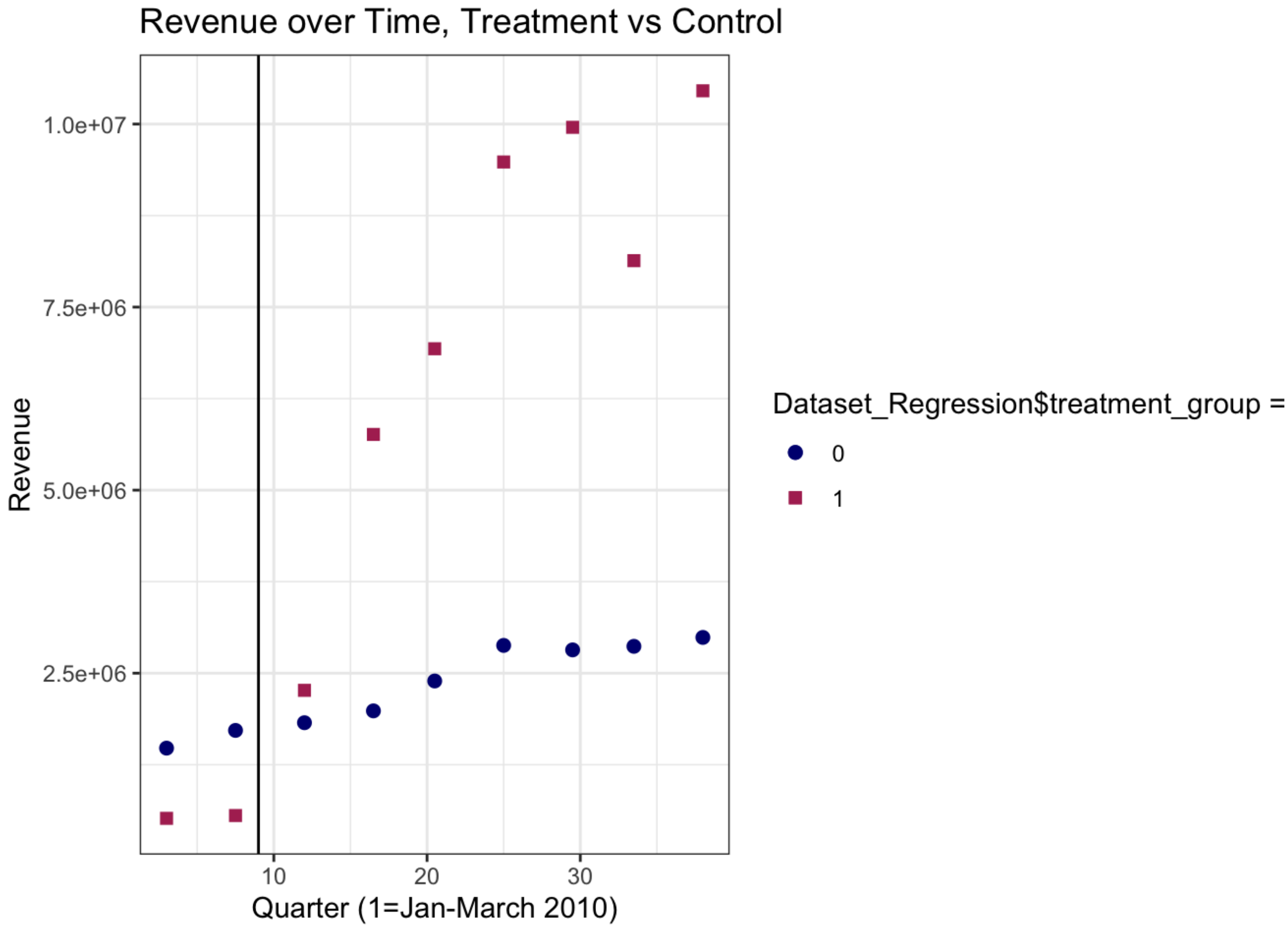

```
ggsave("Plot 2.2 Sales_Adopt.jpeg")
```

```
## Saving 7 x 5 in image
```

```
Bin_rev_adopt <- binsreg(Dataset_Regression$revenue_t, Dataset_Regression$quarter, by =
Dataset_Regression$treatment_group, binspos = "es", samebinsby = TRUE)
```



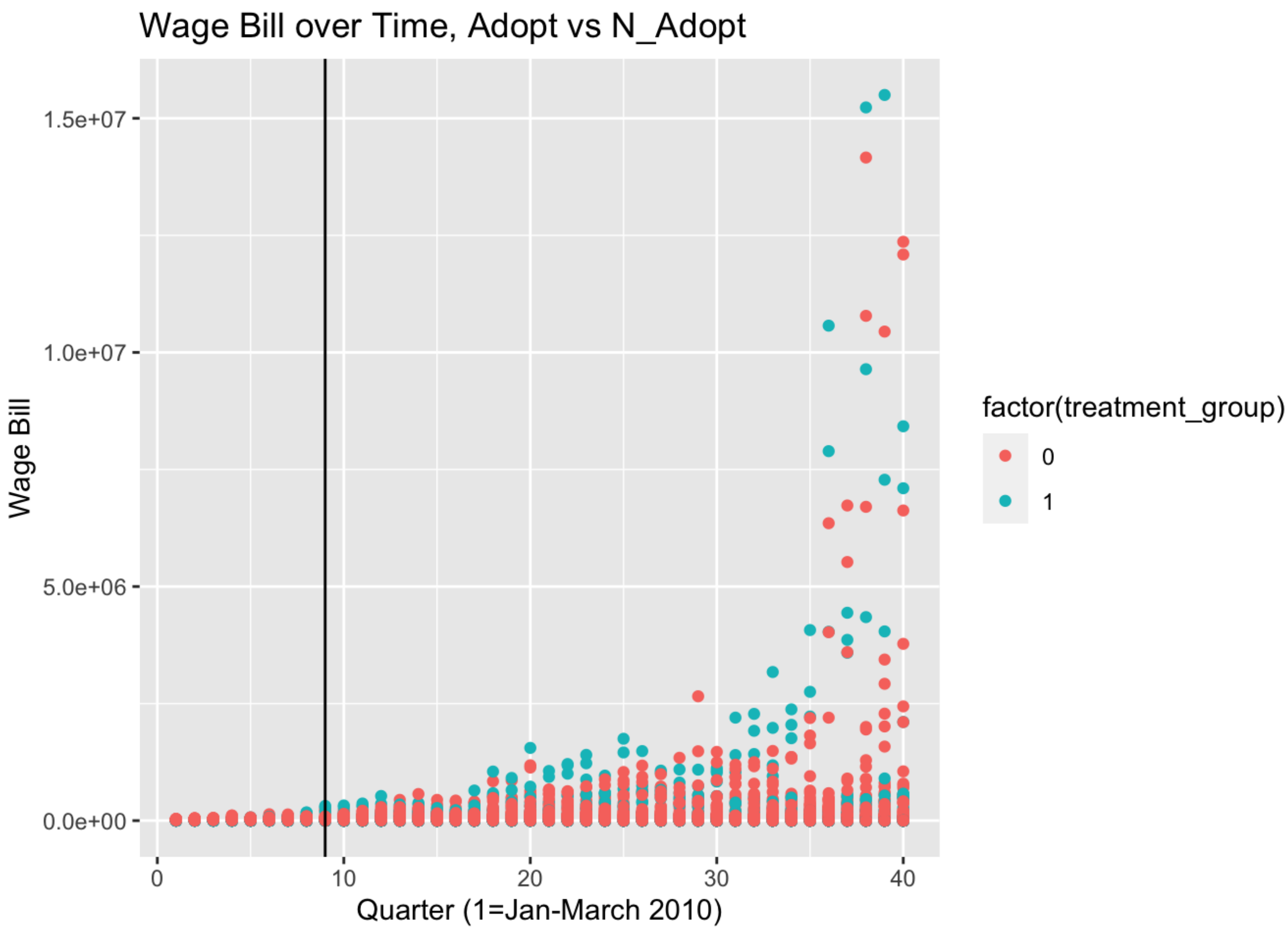

```
Bin_rev_adopt$bins_plot + geom_vline( xintercept = 9) + labs(title = "Revenue over Tim
e, Treatment vs Control", x= "Quarter (1=Jan-March 2010)", y= "Revenue")
```




```
ggsave("Plot 2.2 Rev_Adopt.jpeg")
```

```
## Saving 7 x 5 in image
```

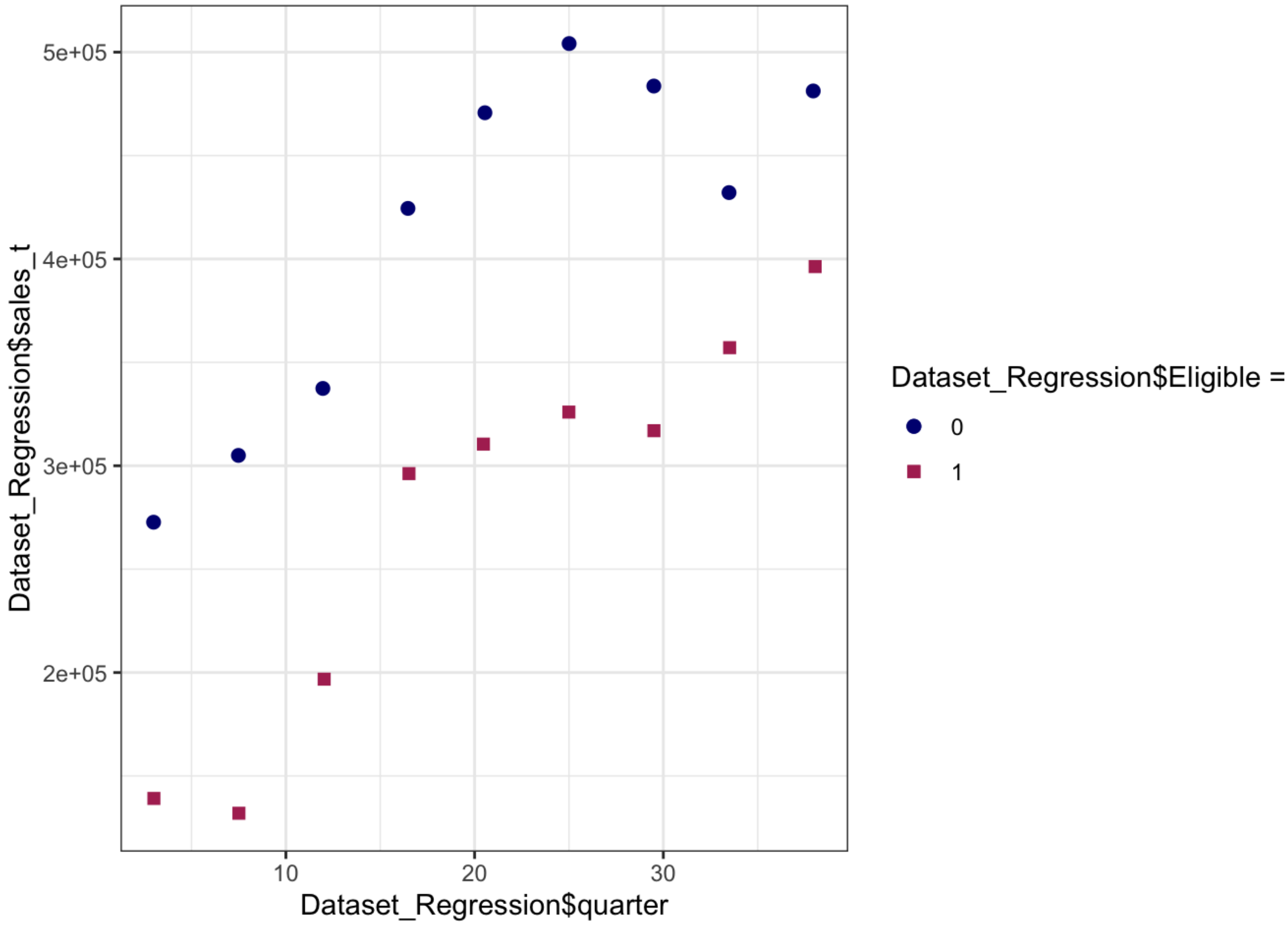
```
ggplot(data=Dataset_Regression, aes(x=quarter,y=wage_bill_t, colour=factor(treatment_gr
oup))) + geom_point() + geom_vline(xintercept = 9) + labs(title = "Wage Bill over Time,
Adopt vs N_Adopt", x="Quarter (1=Jan-March 2010)", y= "Wage Bill")
```



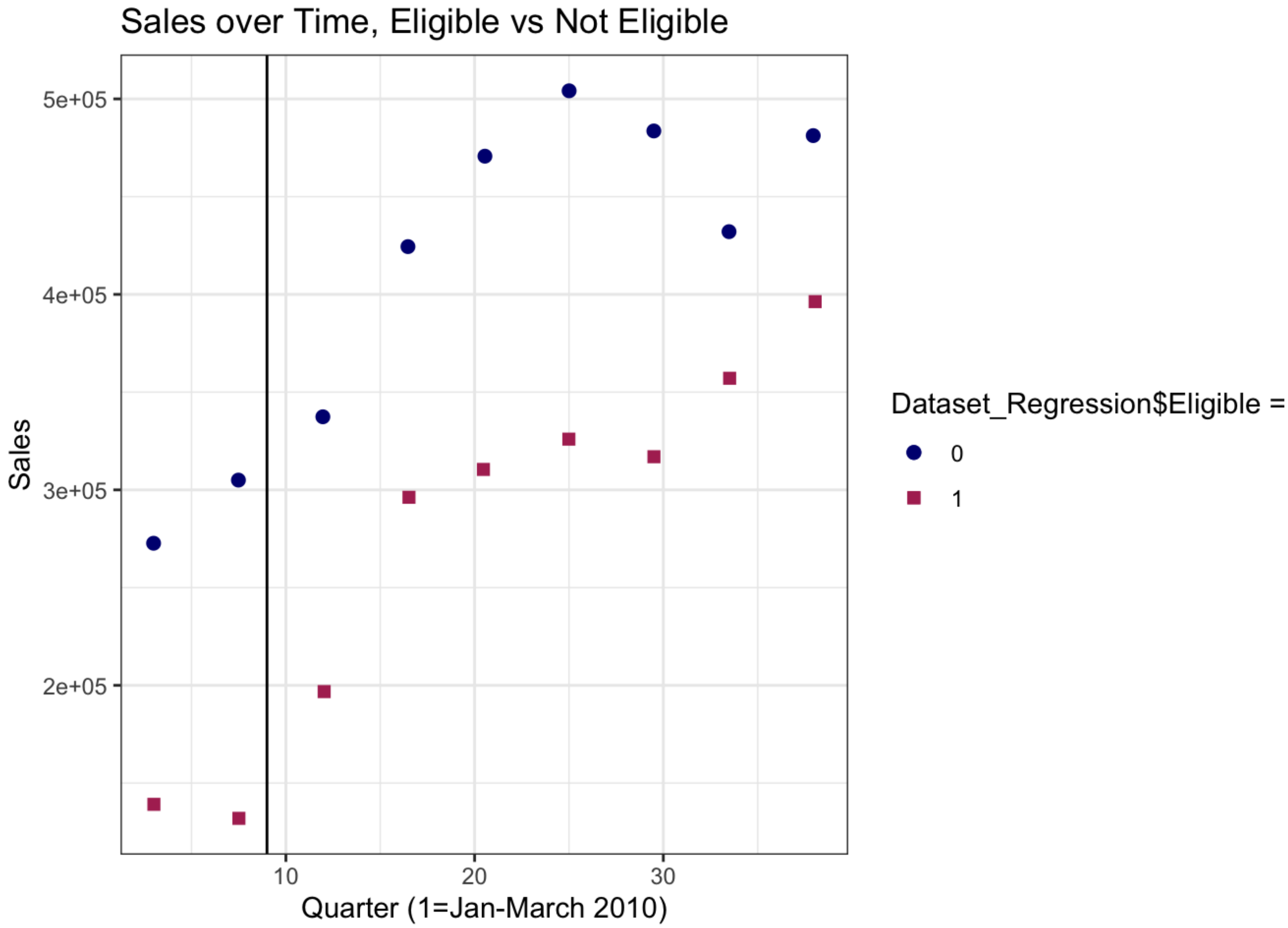

```
ggsave("Plot 2.2 Wage_Adopt.jpeg")
```

```
## Saving 7 x 5 in image
```

```
### Eligible vs Not Eligible
Bin_sales_eligible <- binsreg(Dataset_Regression$sales_t, Dataset_Regression$quarter, b
y = Dataset_Regression$Eligible, binspos = "es", samebinsby = TRUE)
```



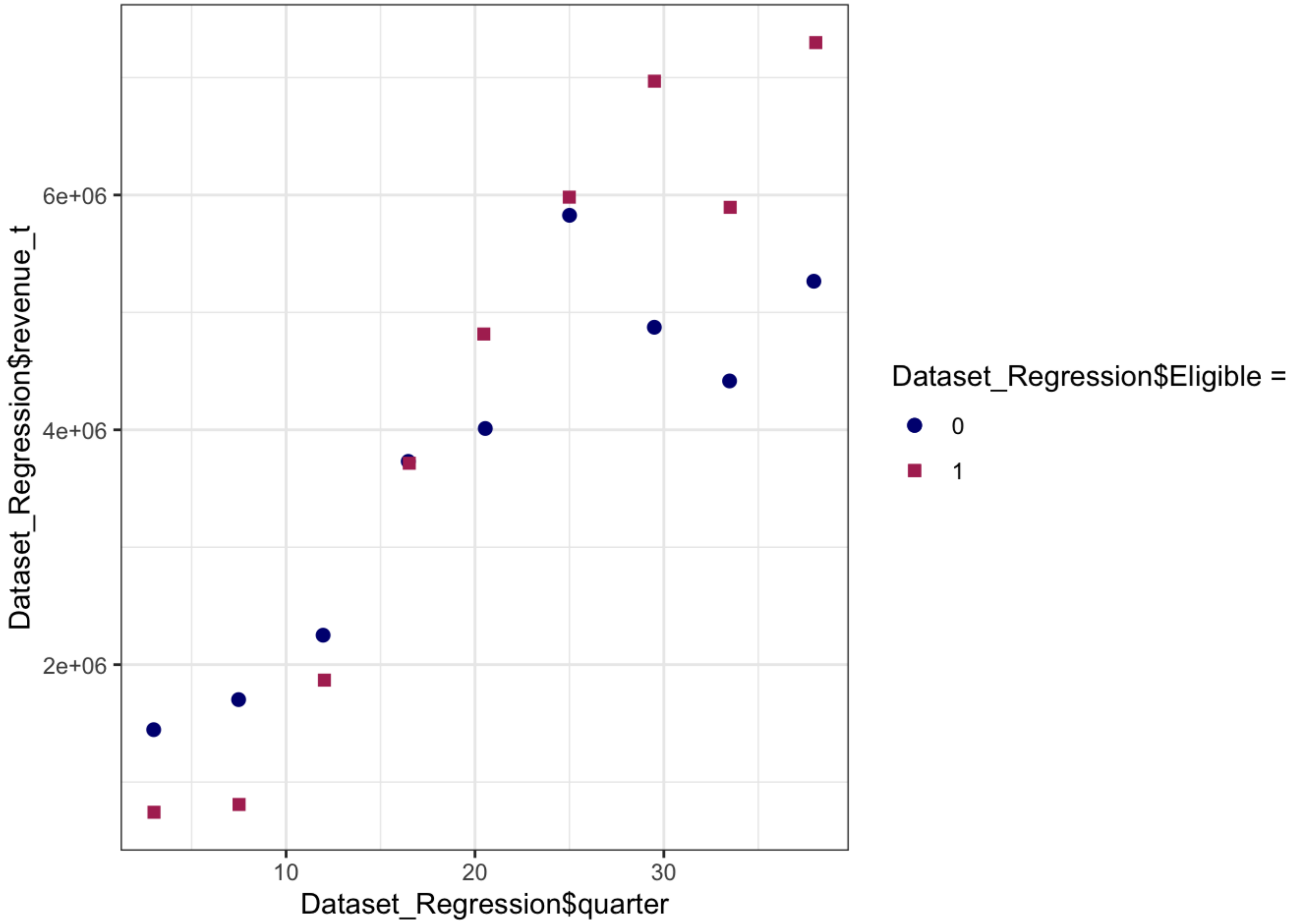

```
Bin_sales_eligible$bins_plot + geom_vline( xintercept = 9) + labs(title = "Sales over T
ime, Eligible vs Not Eligible", x= "Quarter (1=Jan-March 2010)", y= "Sales")
```



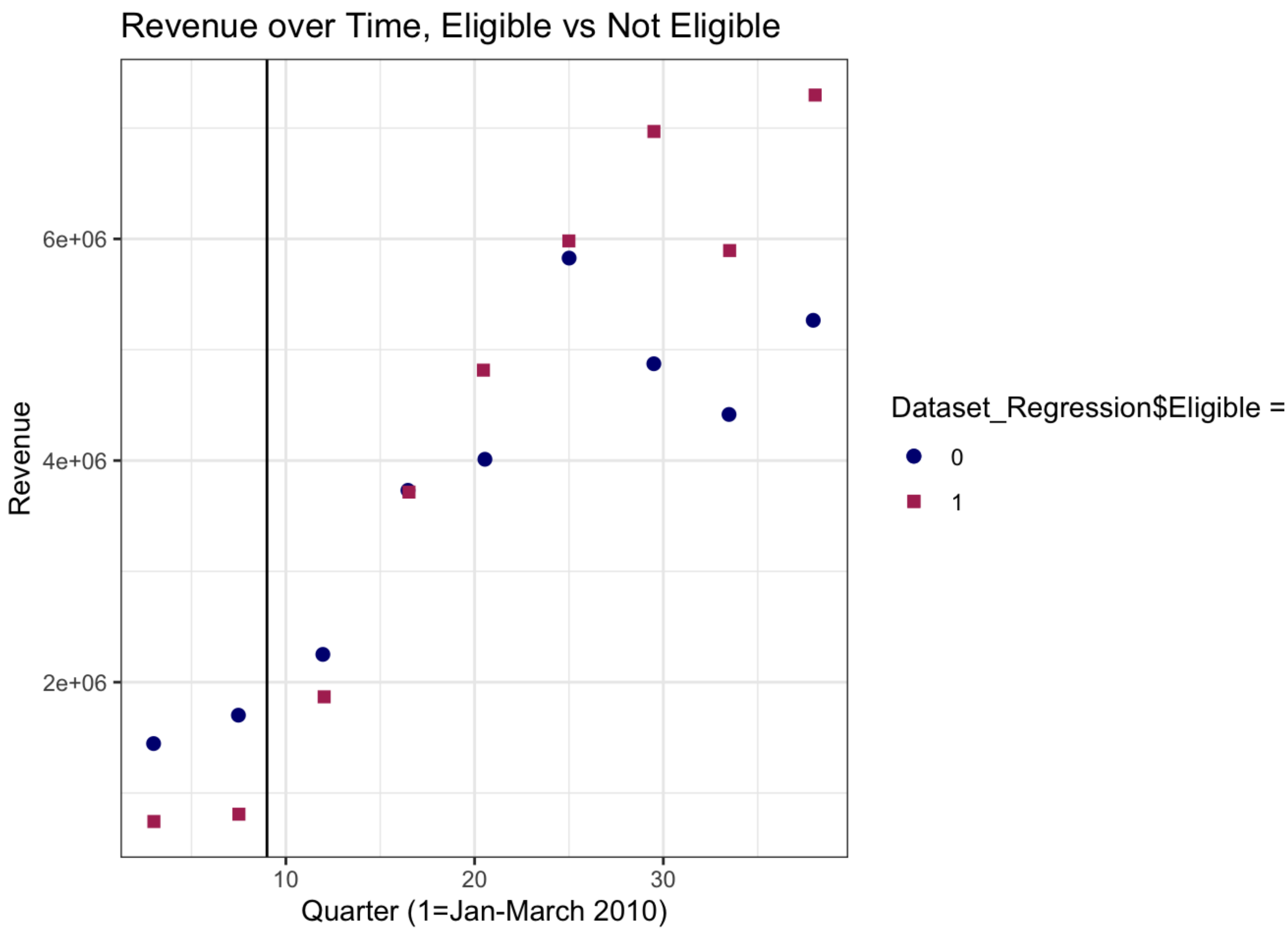

```
ggsave("Plot 2.2 Sales_Eligible.jpeg")
```

```
## Saving 7 x 5 in image
```

```
Bin_rev_eligible <- binsreg(Dataset_Regression$revenue_t, Dataset_Regression$quarter, b
y = Dataset_Regression$Eligible, binspos = "es", samebinsby = TRUE)
```



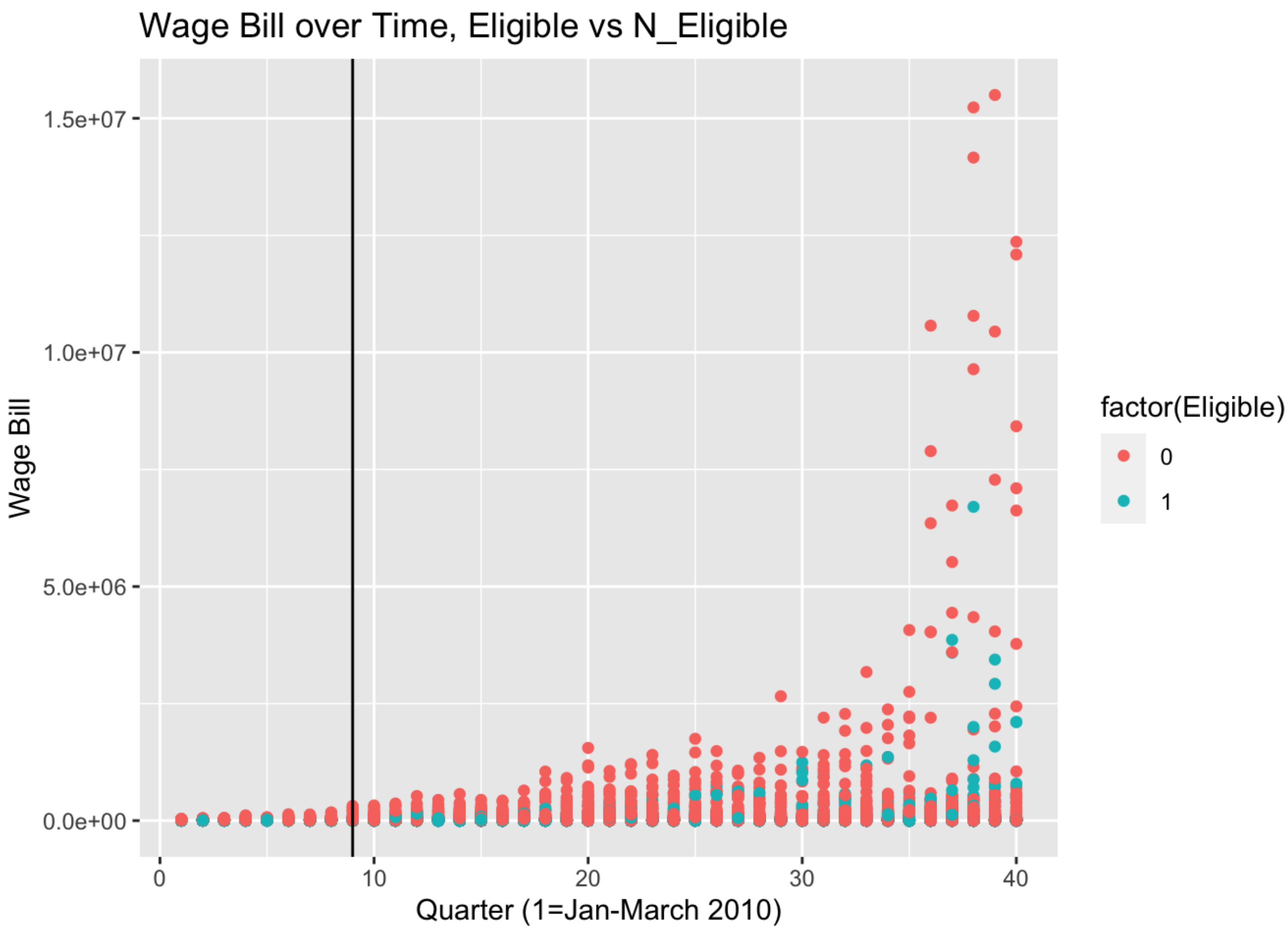

```
Bin_rev_eligible$bins_plot + geom_vline( xintercept = 9) + labs(title = "Revenue over T
ime, Eligible vs Not Eligible", x= "Quarter (1=Jan-March 2010)", y= "Revenue")
```




```
ggsave("Plot 2.2 Rev_Eligible.jpeg")
```

```
## Saving 7 x 5 in image
```

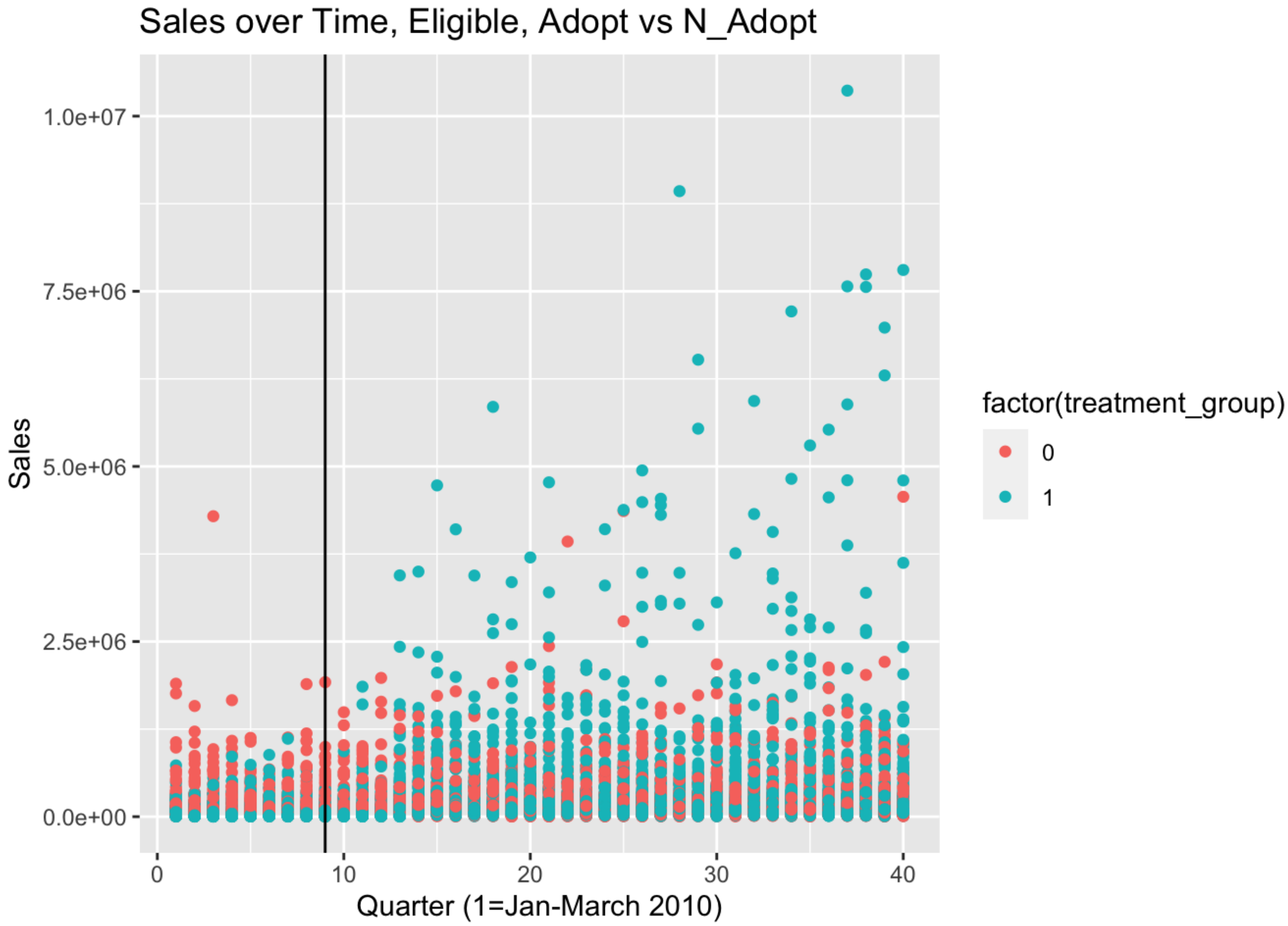
```
ggplot(data=Dataset_Regression, aes(x=quarter,y=wage_bill_t, colour=factor(Eligible)))
+ geom_point() + geom_vline(xintercept = 9) + labs(title = "Wage Bill over Time, Eligib
le vs N_Eligible", x="Quarter (1=Jan-March 2010)", y= "Wage Bill")
```




```
ggsave("Plot 2.2 Wage_Eligible.jpeg")
```

```
## Saving 7 x 5 in image
```

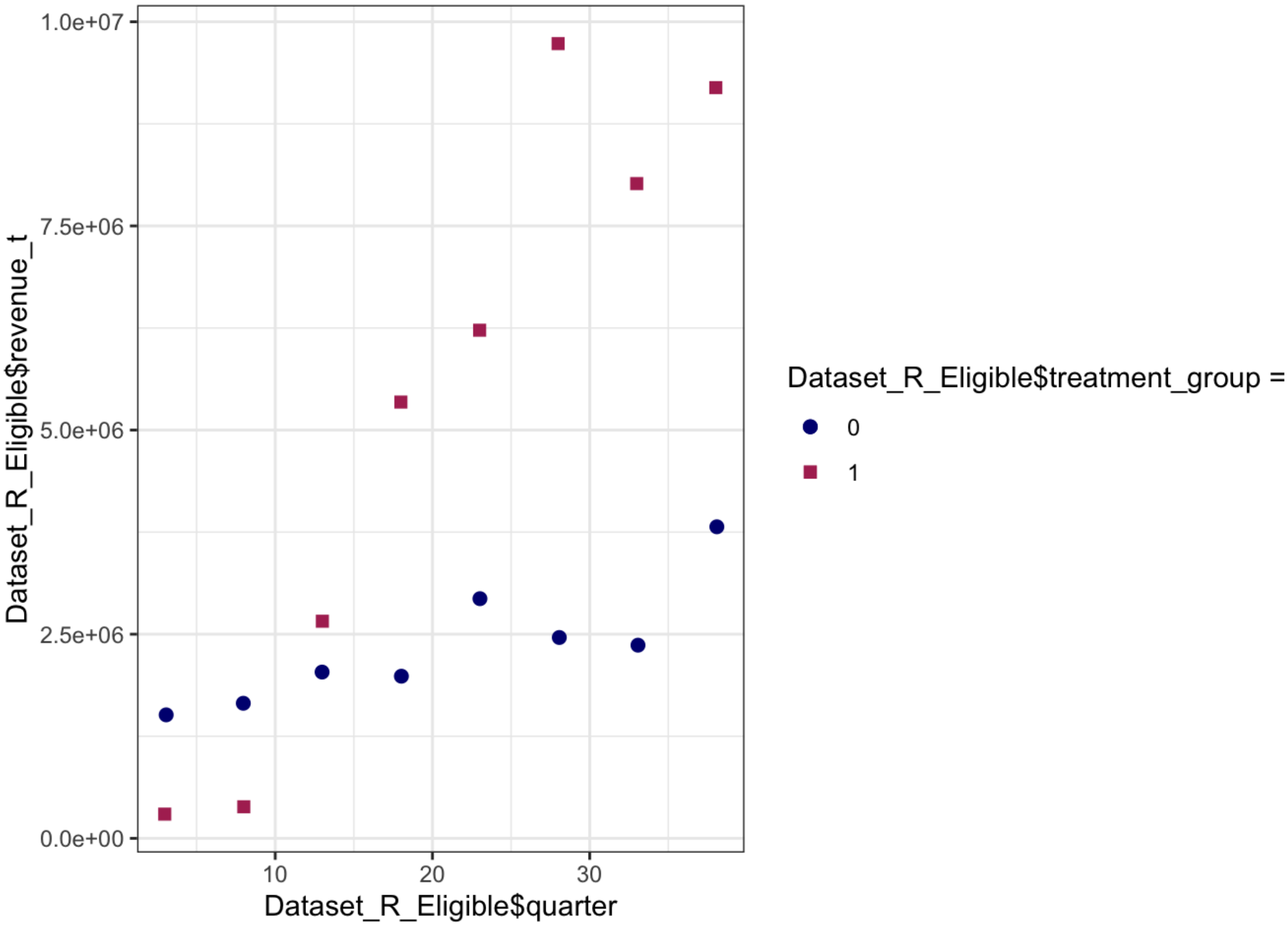
```
### Eligible Adopt vs Not Adopt

### Tried to include plot of sales like above, but for some reason the table would not
 plot correctly, would get contrast error despite running the same code as above
Dataset_R_Eligible <- Dataset_Regression %>%
  filter(Eligible == 1)

ggplot(data=Dataset_R_Eligible, aes(x=quarter,y=sales_t, colour=factor(treatment_grou
p))) + geom_point() + geom_vline(xintercept = 9) + labs(title = "Sales over Time, Eligi
ble, Adopt vs N_Adopt", x="Quarter (1=Jan-March 2010)", y= "Sales")
```



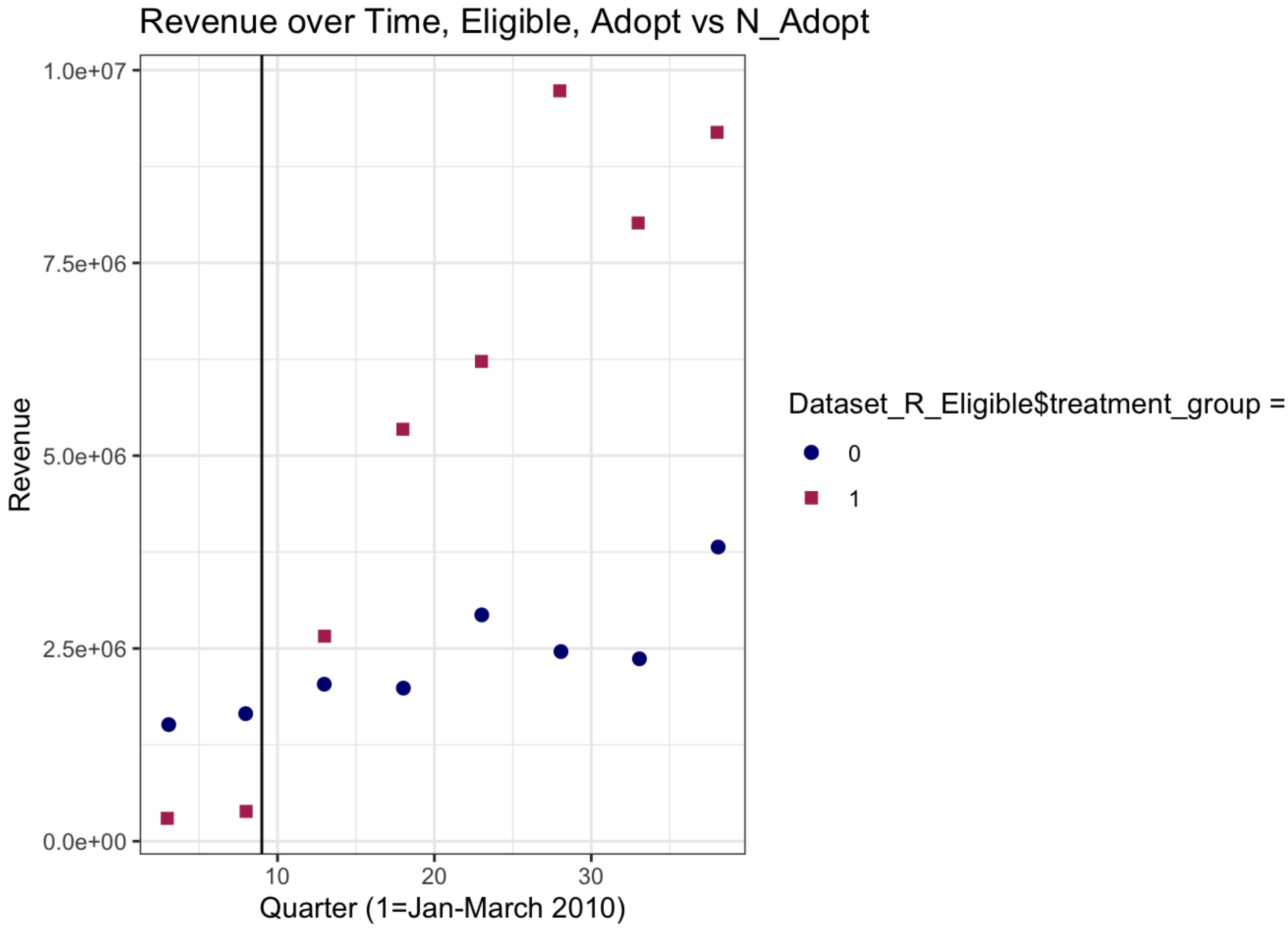

```
ggsave("Plot 2.2 Sales_Adopt_Eligible.jpeg")
```

```
## Saving 7 x 5 in image
```

```
Bin_Rev_eligible_adopt <- binsreg(Dataset_R_Eligible$revenue_t, Dataset_R_Eligible$quar
ter, by= Dataset_R_Eligible$treatment_group, binspos = "es", samebinsby = TRUE)
```



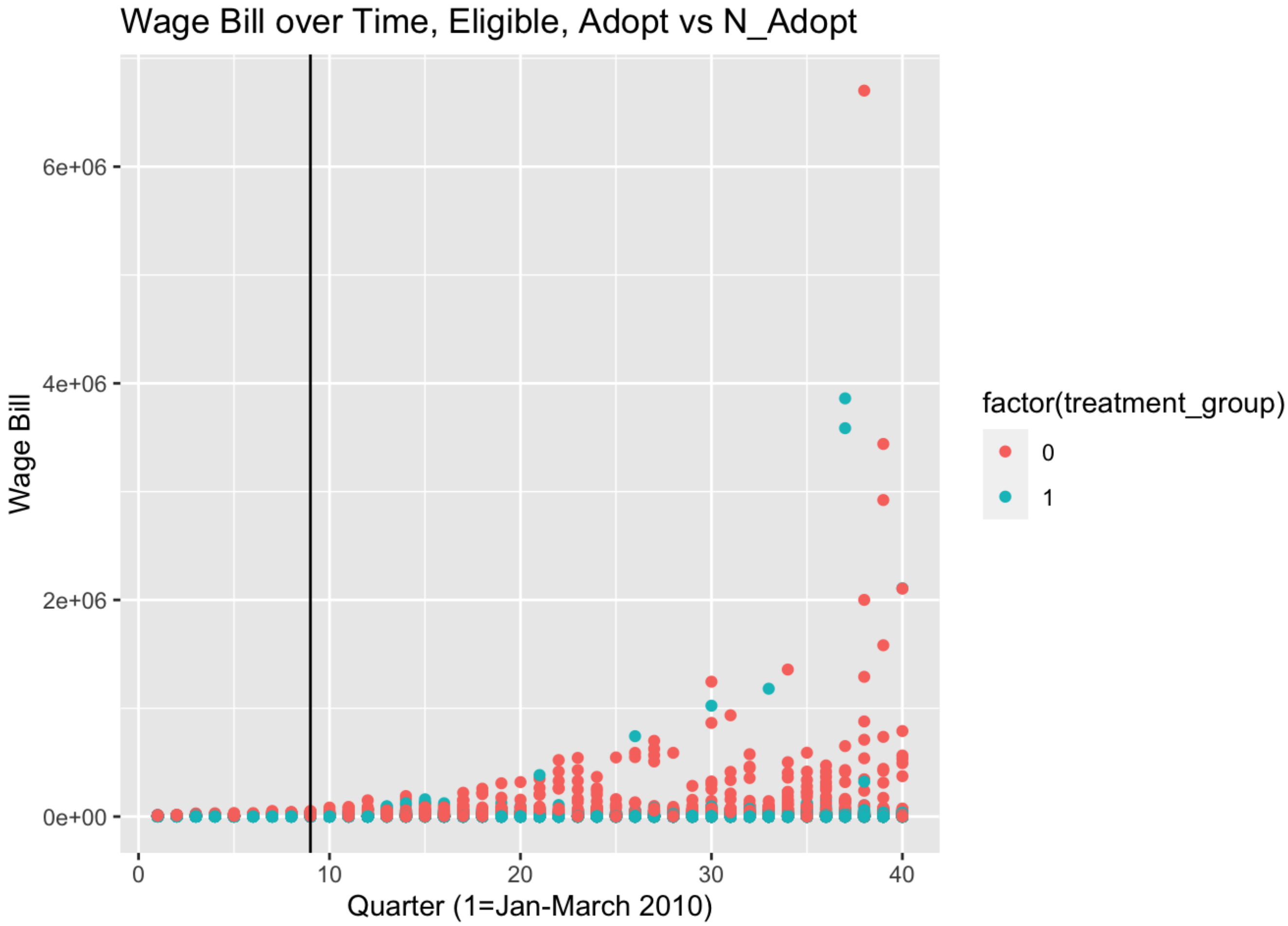

```
Bin_Rev_eligible_adopt$bins_plot + geom_vline( xintercept = 9) + labs(title = "Revenue
 over Time, Eligible, Adopt vs N_Adopt", x= "Quarter (1=Jan-March 2010)", y= "Revenue")
```




```
ggsave("Plot 2.2 Rev_Adopt_Eligible.jpeg")
```

```
## Saving 7 x 5 in image
```

```
ggplot(data=Dataset_R_Eligible, aes(x=quarter,y=wage_bill_t, colour=factor(treatment_gr
oup))) + geom_point() + geom_vline(xintercept = 9) + labs(title = "Wage Bill over Time,
Eligible, Adopt vs N_Adopt", x="Quarter (1=Jan-March 2010)", y= "Wage Bill")
```

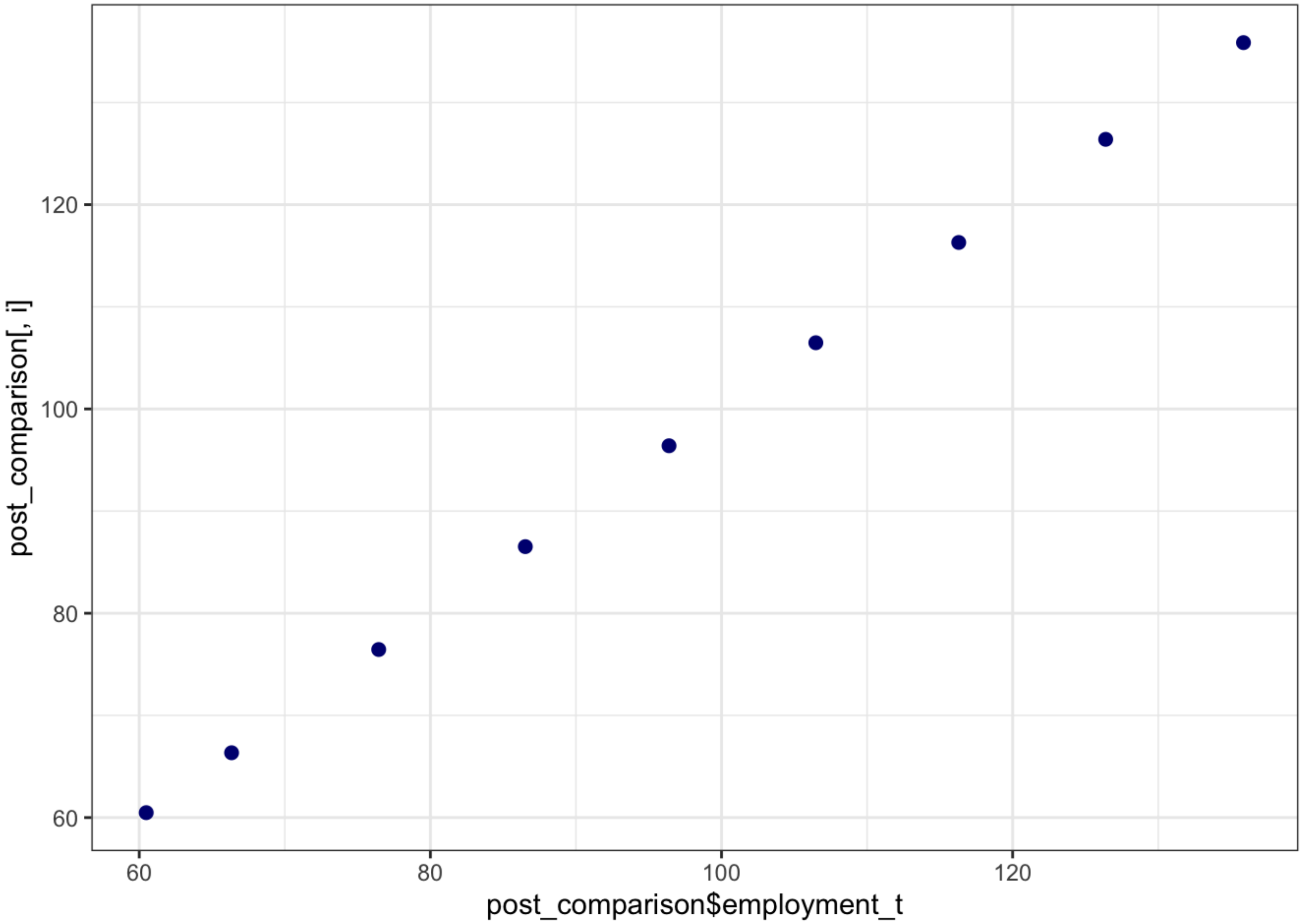



```
ggsave("Plot 2.2 Wage_Adopt_Eligible.jpeg")
```

```
## Saving 7 x 5 in image
```

2.3b

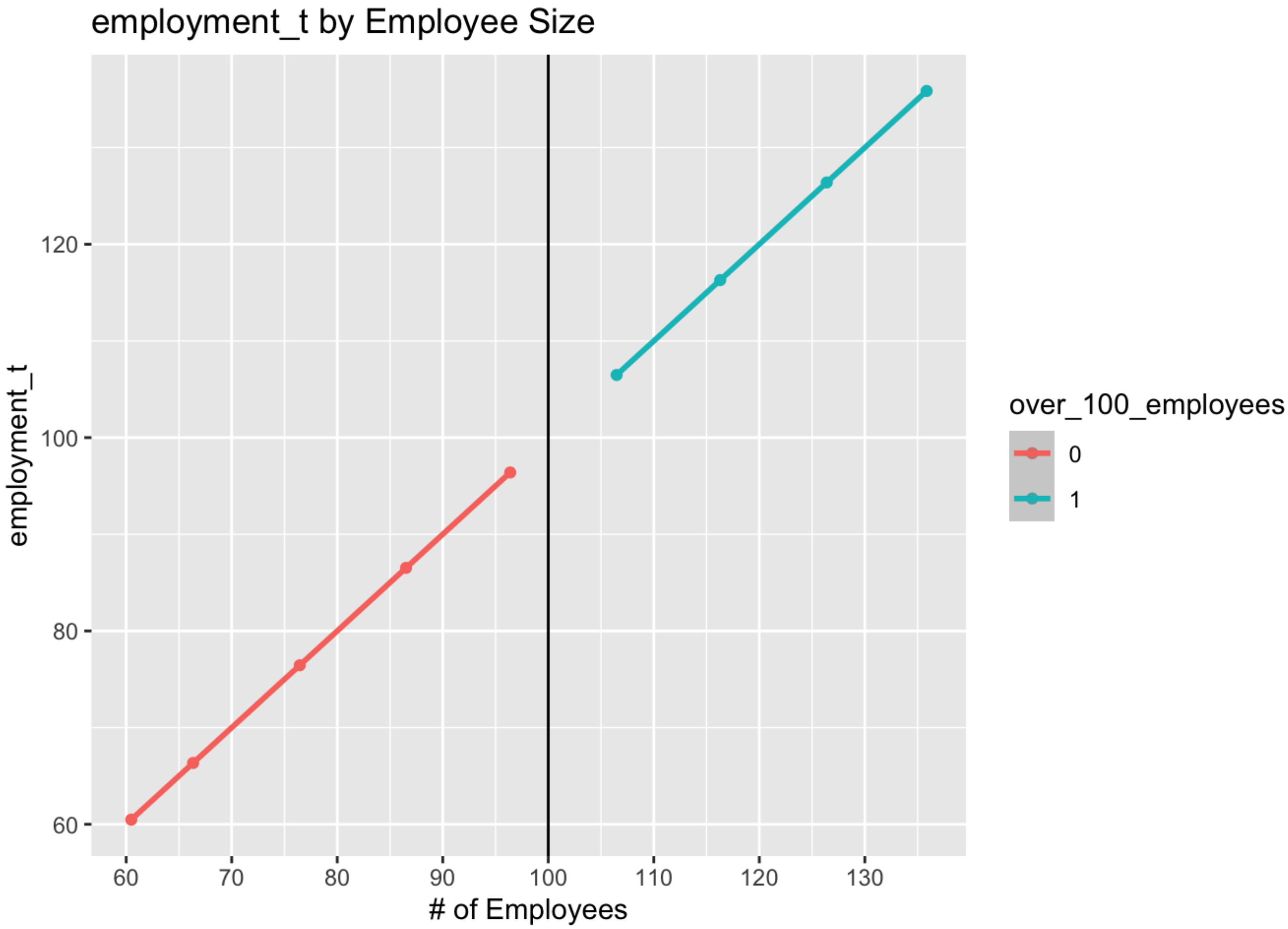
```
### post_comparison is dataset of firms of a somewhat comparable size (60 to 140 employ
ees) with observations from after program beggining. This grouping is chosen as firms l
arger or smaller tend have outcome variables largely influenced by their amount of empl
oyees, rather than by being treated or not
### Adopt and treatment group are the same as we are looking at data only from after pr
ogram implementation
post_comparison <- Dataset_1 %>%
  filter(date >= as.Date("2013-01-01"), employment_t >= 60 &  employment_t <= 140 ) %>%
  mutate(employment_100 = as.factor(ifelse(employment_t >100, 1,0)))  %>%
  select(c(employment_t,sales_t,wage_bill_t,revenue_t,adopt_t, treatment_group))
## This function creates a graph of the regression discontinuity for each outcome varia
ble nd saves graph into folder
## Bins chosen such that firms with more than 100 employees and less than 100 employees
are not in same bin
for (i in 1:ncol(post_comparison)) {
  bin_scatter <- binsreg(post_comparison[, i], post_comparison$employment_t, binspos =
 seq(61,131,10))$data.plot[[1]]$data.dots
  bin_scatter <- as.data.frame(bin_scatter)
  bin_scatter$over_100_employees <- as.factor(ifelse(bin_scatter$x > 100,1,0))

  fig <- ggplot(data=bin_scatter, aes(x= x, y=fit, colour = over_100_employees)) + labs
(title = paste(colnames(post_comparison)[i], "by Employee Size"), y = colnames(post_com
parison)[i], x = "# of Employees") + geom_point() + geom_smooth(method = "lm") + geom_v
line(xintercept = 100) + scale_x_continuous(breaks = seq(60,150,10))
  print(fig)
  name <- paste0(paste(colnames(post_comparison)[i], "by Employee Size"),".jpeg")
  ggsave(name)
}
```

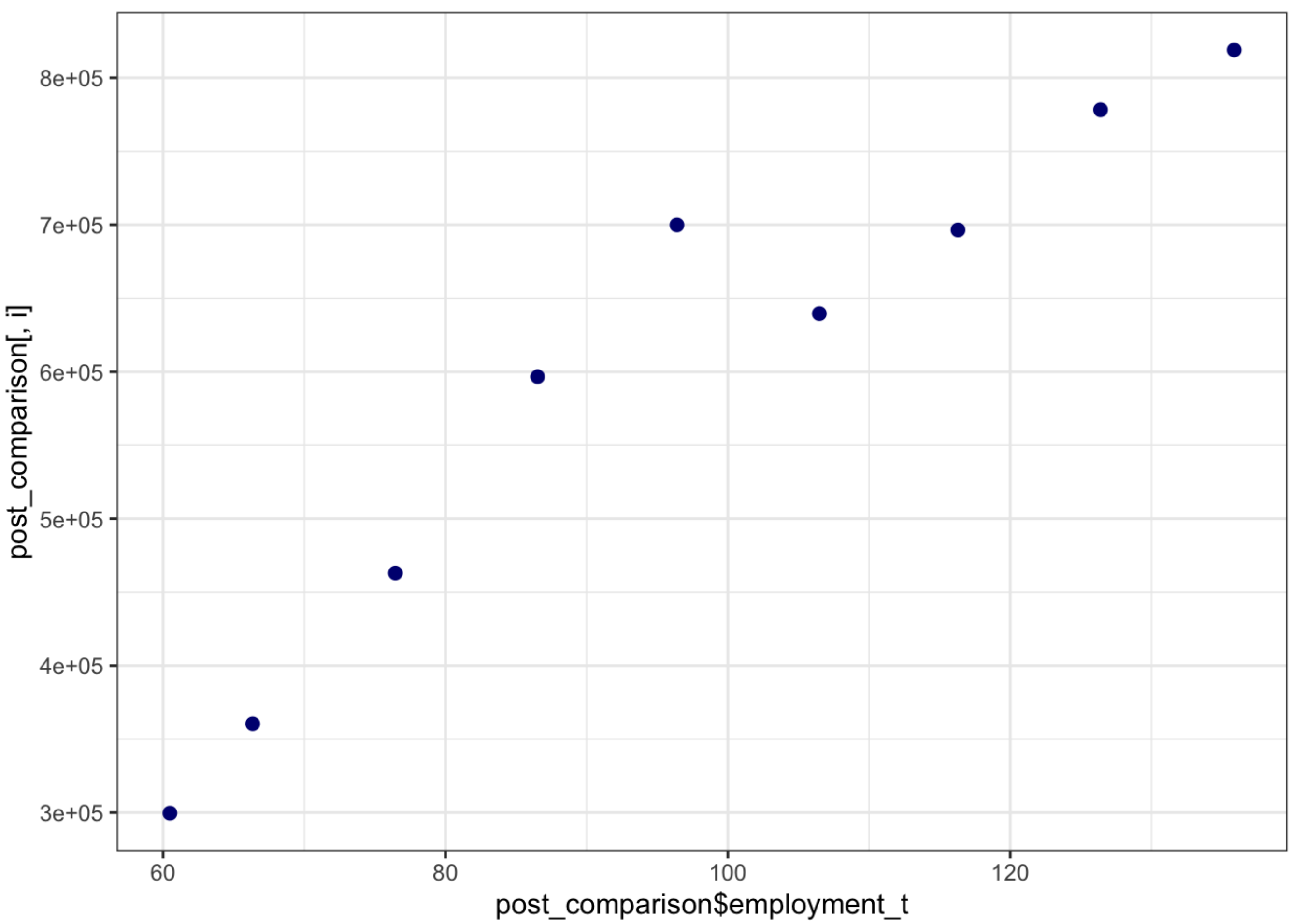



```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Saving 7 x 5 in image
```

```
## `geom_smooth()` using formula 'y ~ x'
```

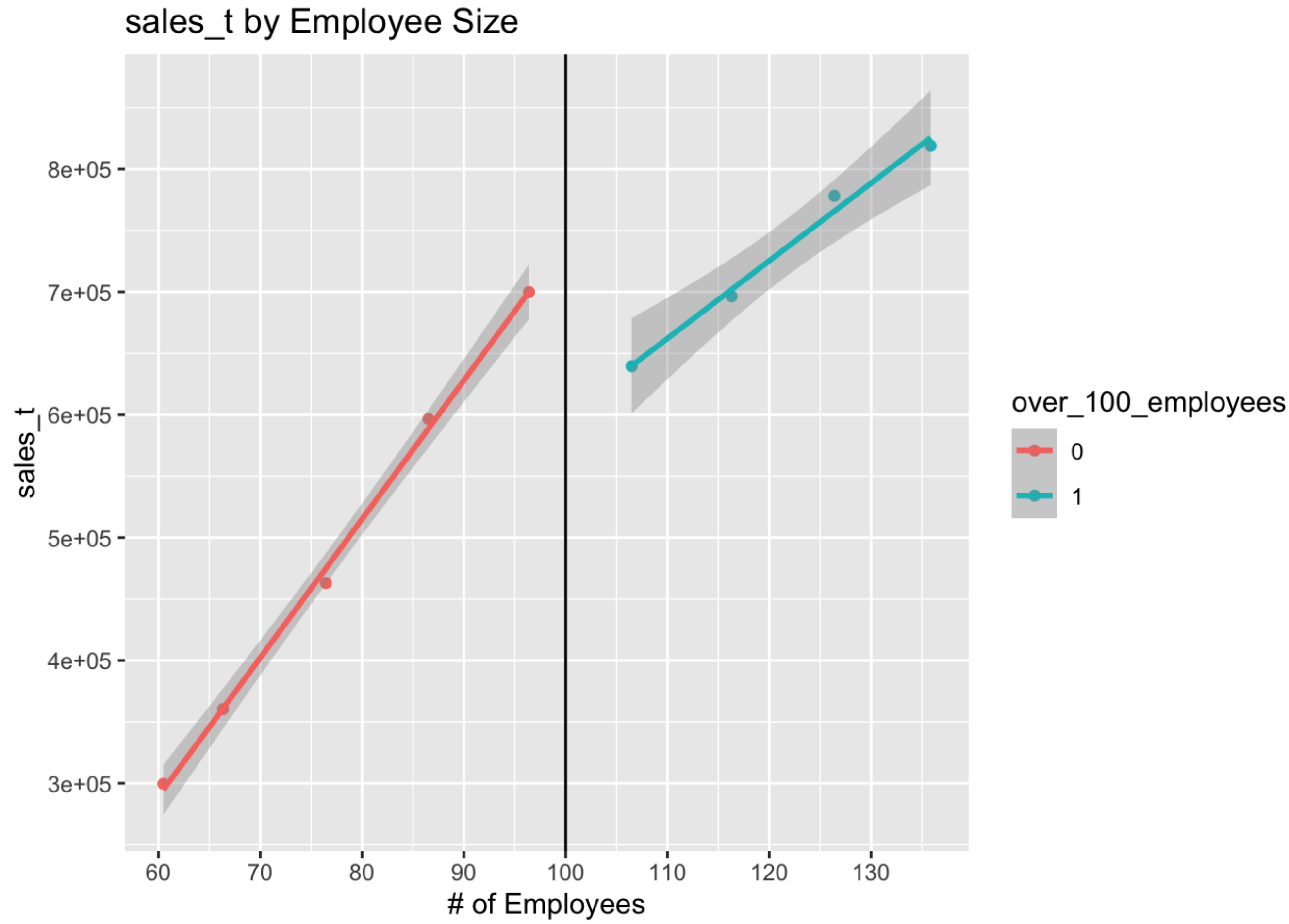

employment_t by Employee Size
employment_t
120
100
80
60
60
70
80
90
100
110
120
130
# of Employees
over_100_employees
0
1

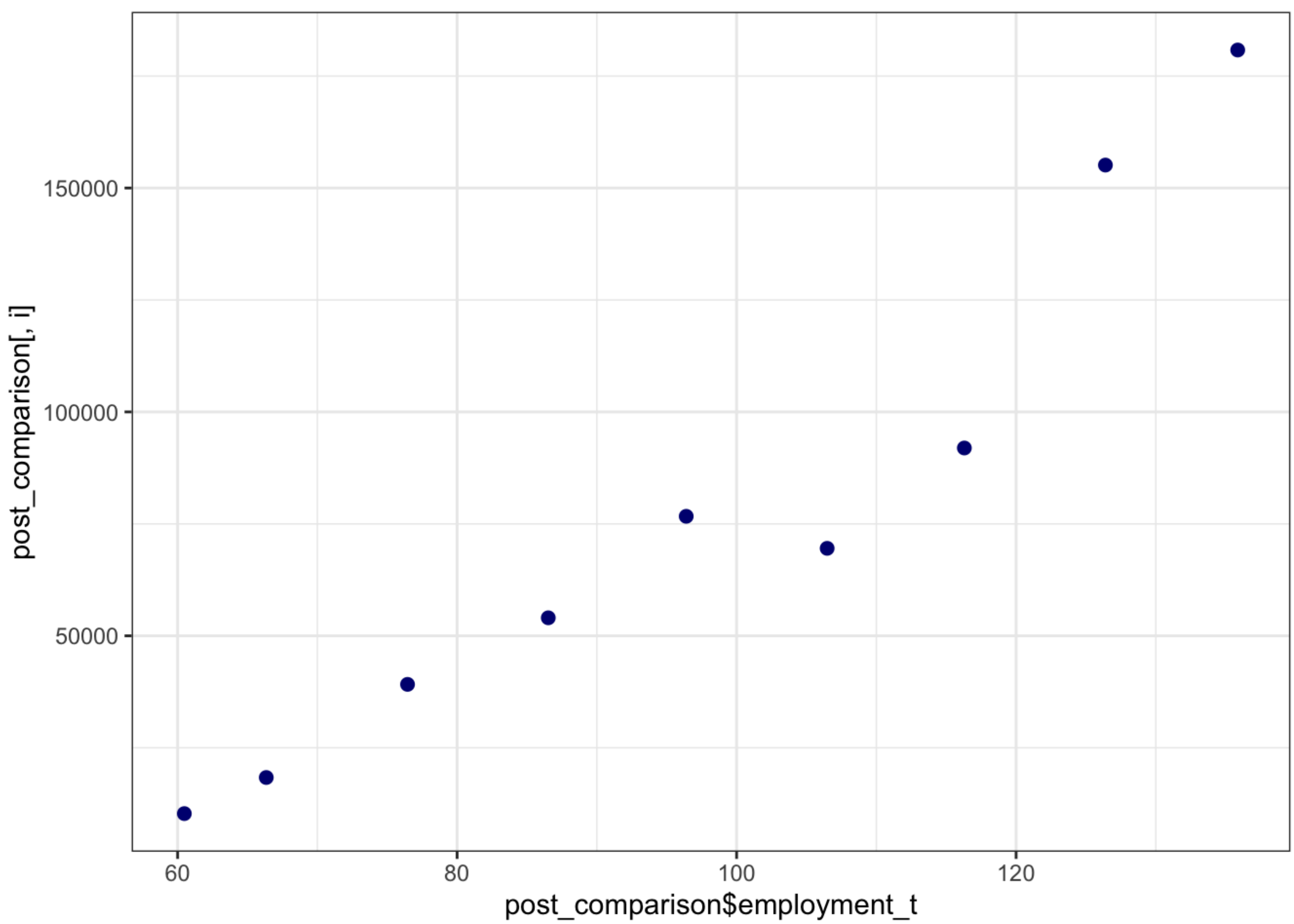

post_comparison[, i]
8e+05
7e+05
6e+05
5e+05
4e+05
3e+05
60
80
100
120
post_comparison$employment_t

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Saving 7 x 5 in image
```

```
## `geom_smooth()` using formula 'y ~ x'
```

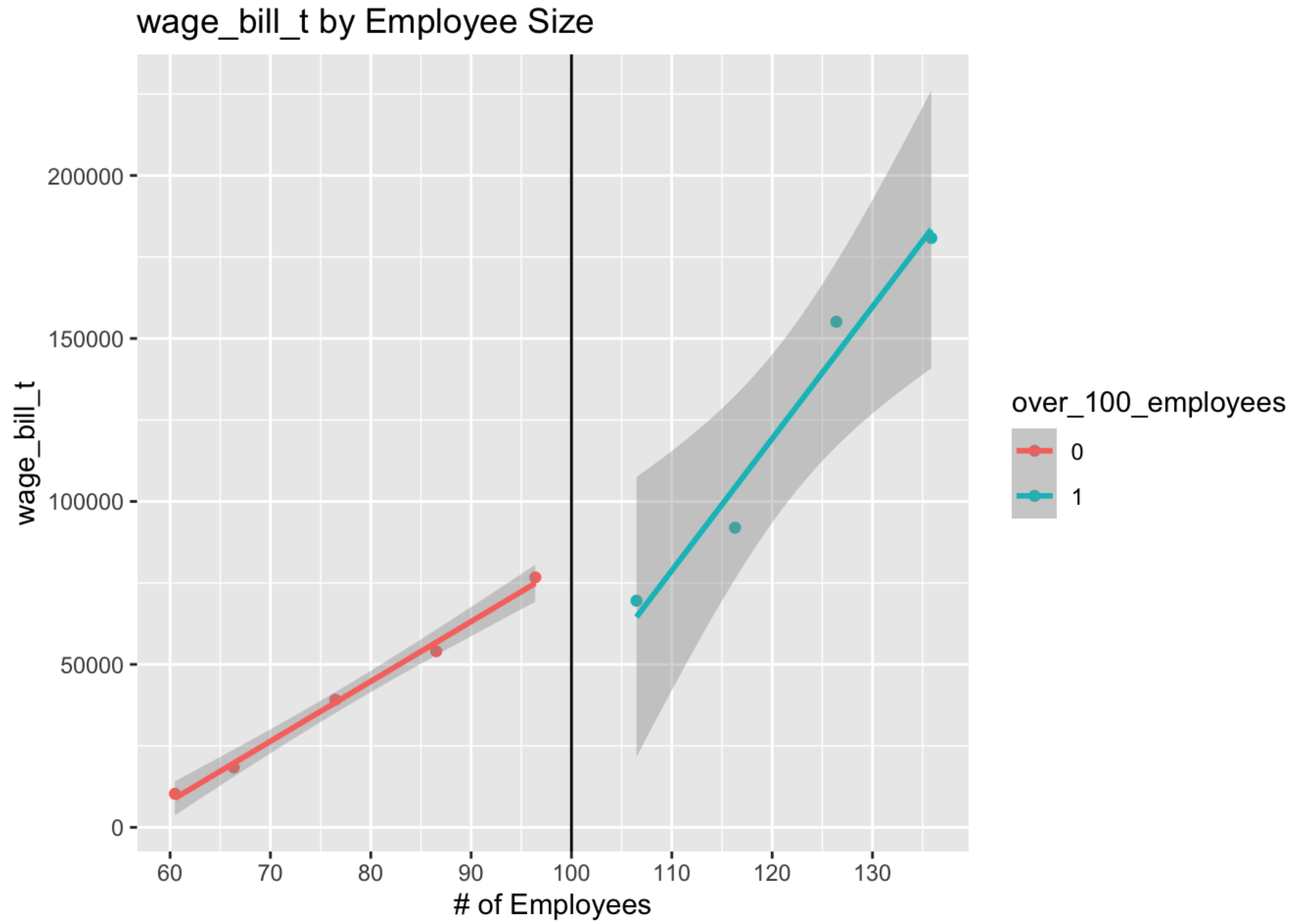

sales_t by Employee Size
sales_t
8e+05
7e+05
6e+05
5e+05
4e+05
3e+05
60
70
80
90
100
110
120
130
# of Employees
over_100_employees
0
1

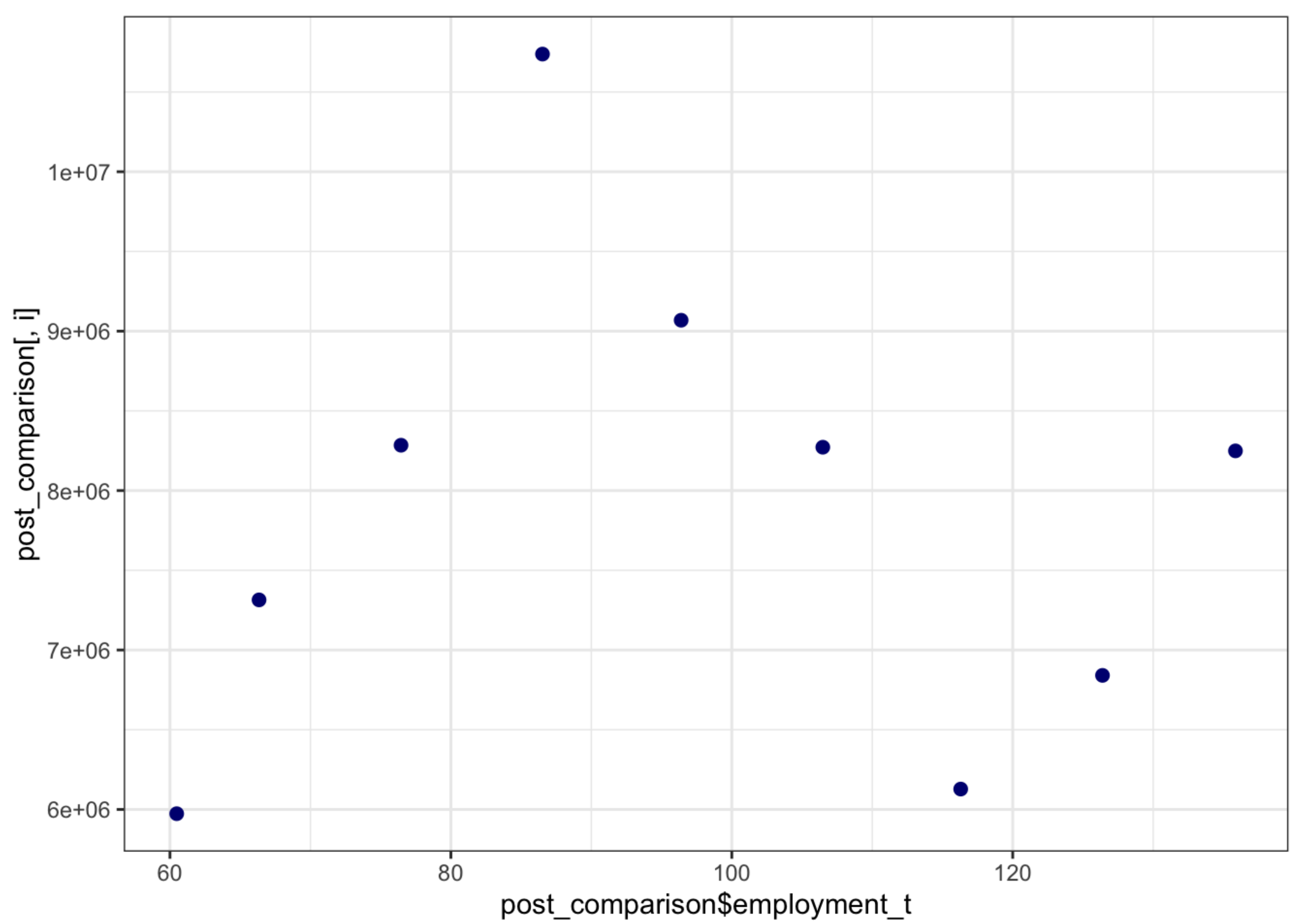

post_comparison[, i]
150000
100000
50000
60
80
100
120
post_comparison$employment_t

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Saving 7 x 5 in image
```

```
## `geom_smooth()` using formula 'y ~ x'
```

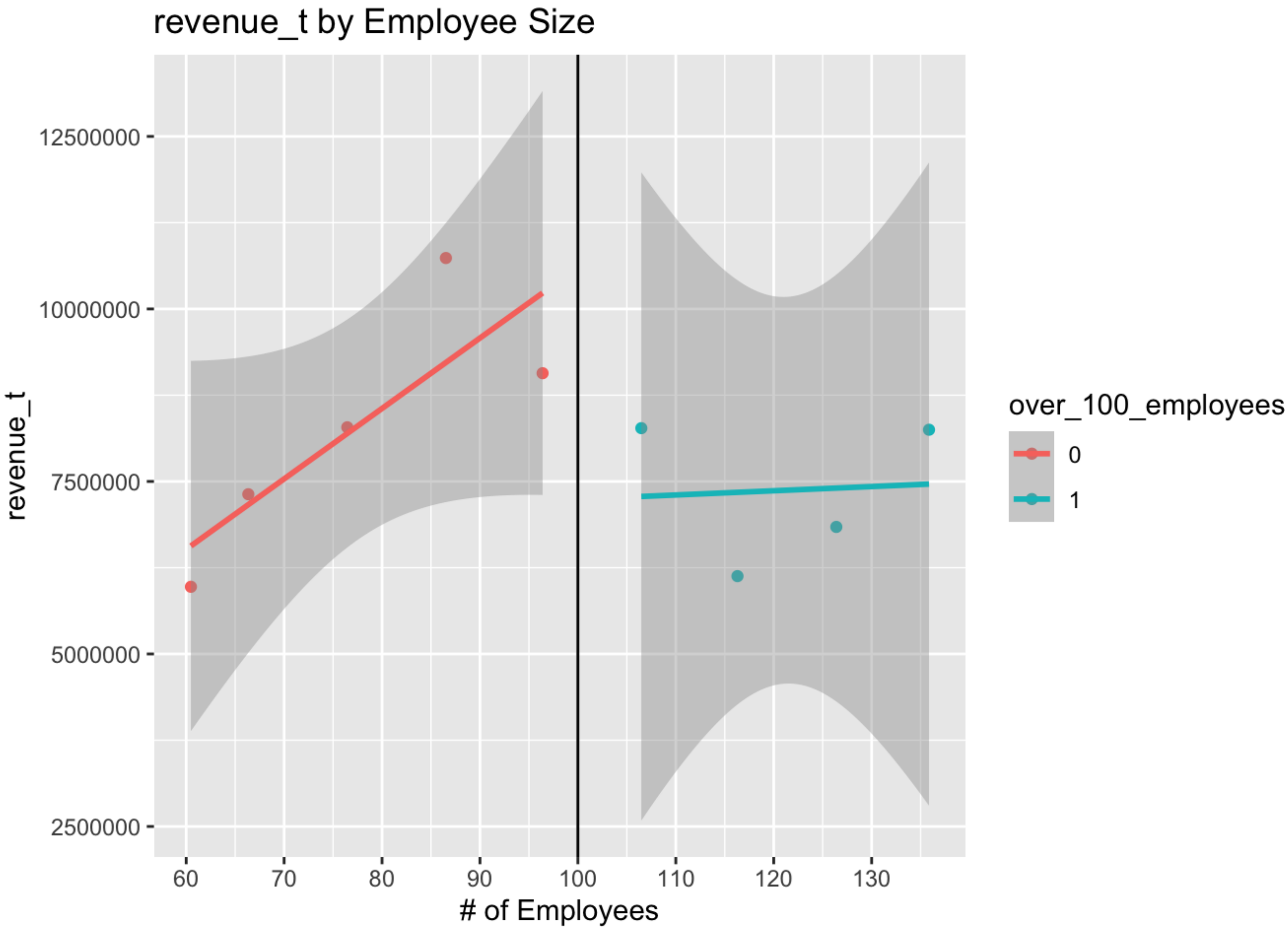

wage_bill_t by Employee Size
wage_bill_t
200000
150000
100000
50000
0
60
70
80
90
100
110
120
130
# of Employees
over_100_employees
0
1

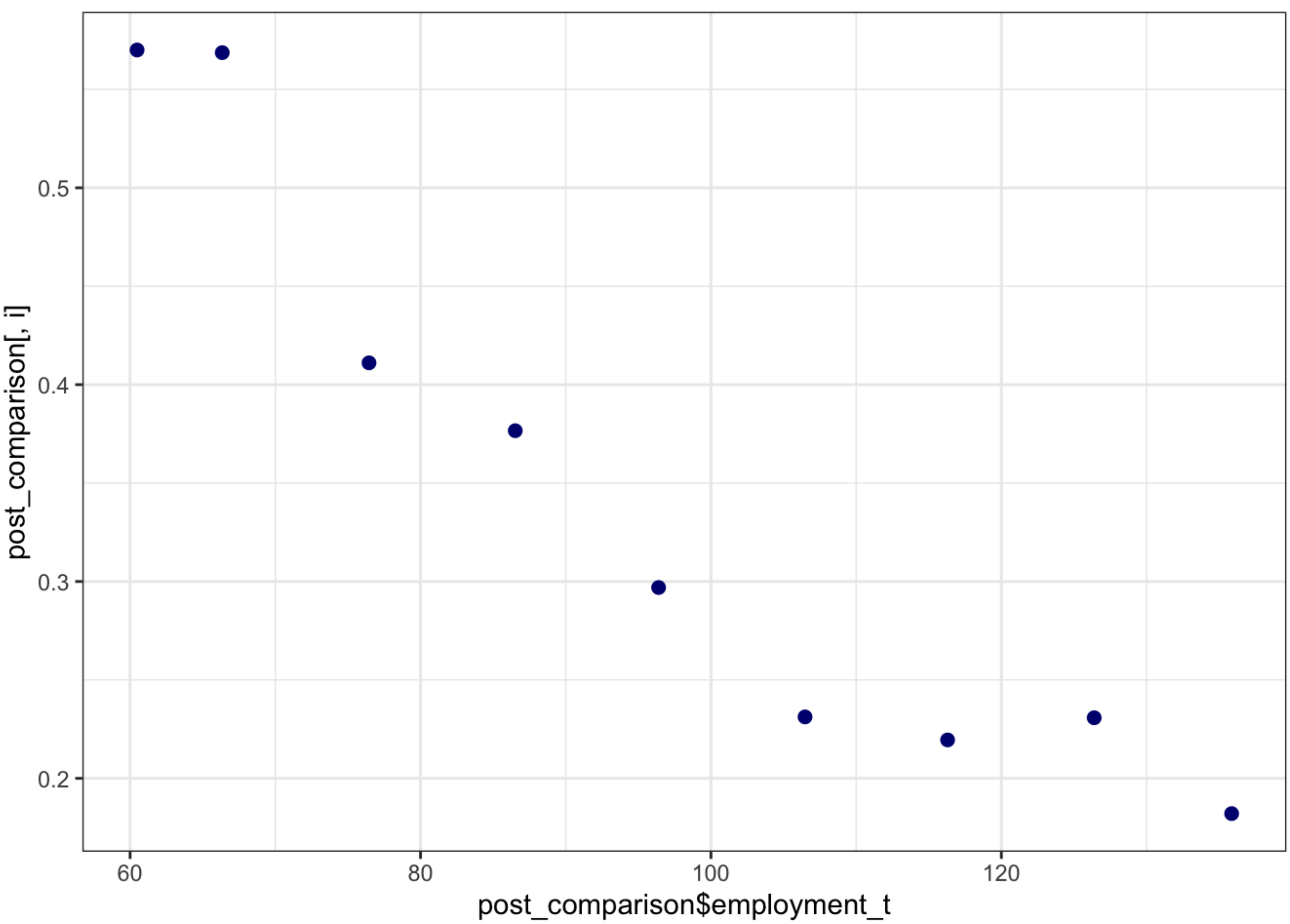

post_comparison[, i]
1e+07
9e+06
8e+06
7e+06
6e+06
60
80
100
120
post_comparison$employment_t

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Saving 7 x 5 in image
```

```
## `geom_smooth()` using formula 'y ~ x'
```

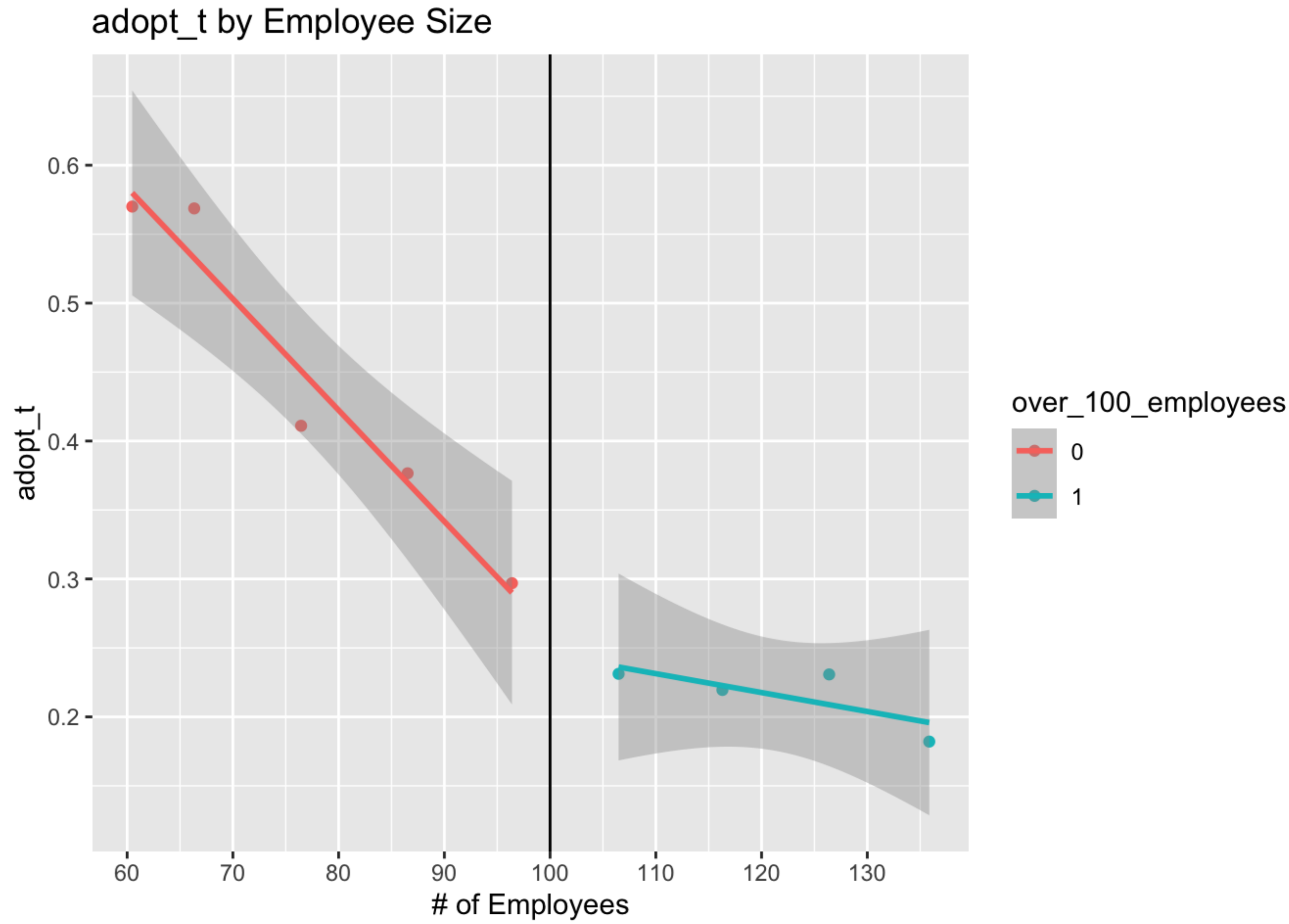

revenue_t by Employee Size
revenue_t
12500000
10000000
7500000
5000000
2500000
60
70
80
90
100
110
120
130
# of Employees
over_100_employees
0
1

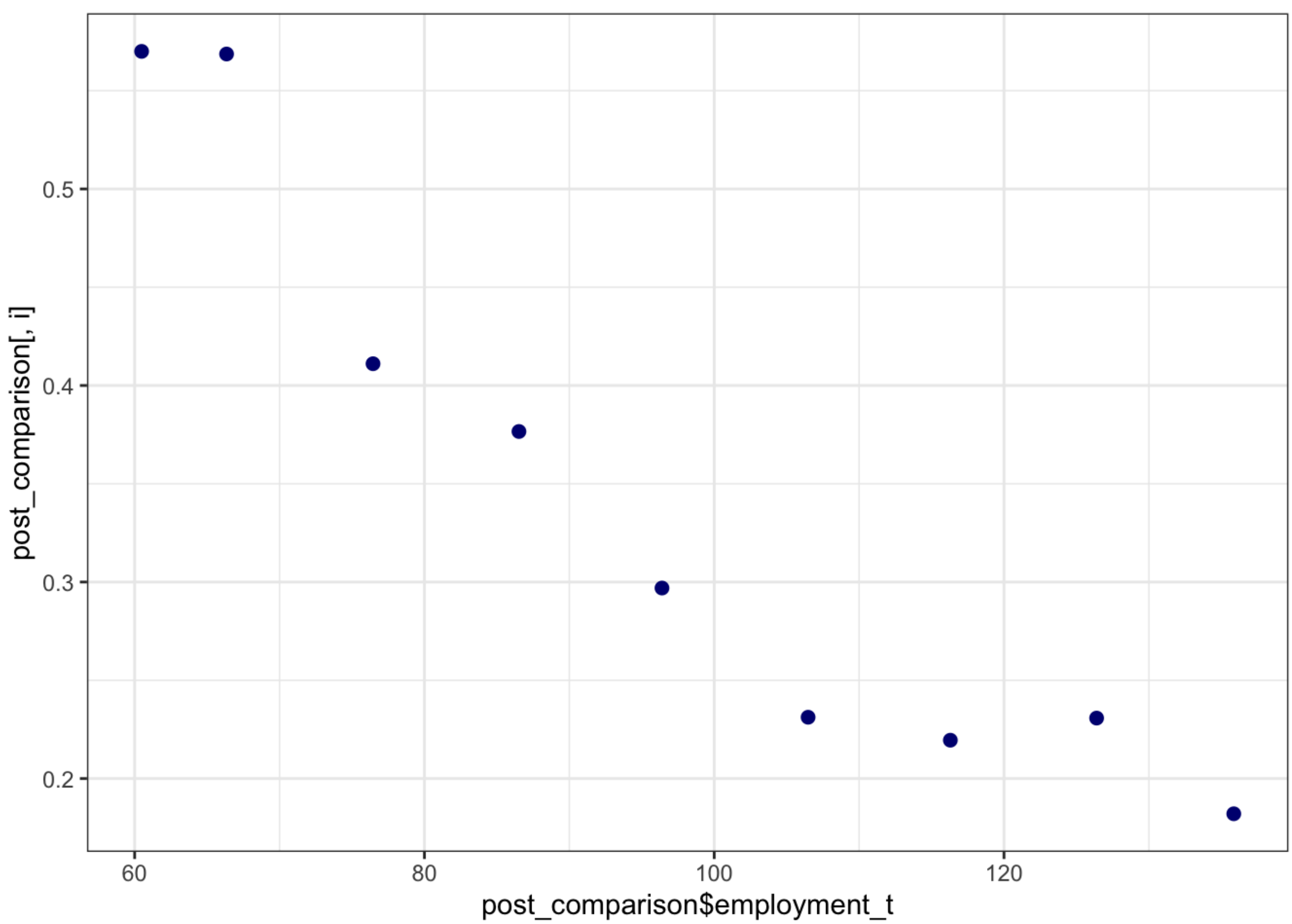

post_comparison[, i]
0.5
0.4
0.3
0.2
60
80
100
120
post_comparison$employment_t

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Saving 7 x 5 in image
```

```
## `geom_smooth()` using formula 'y ~ x'
```

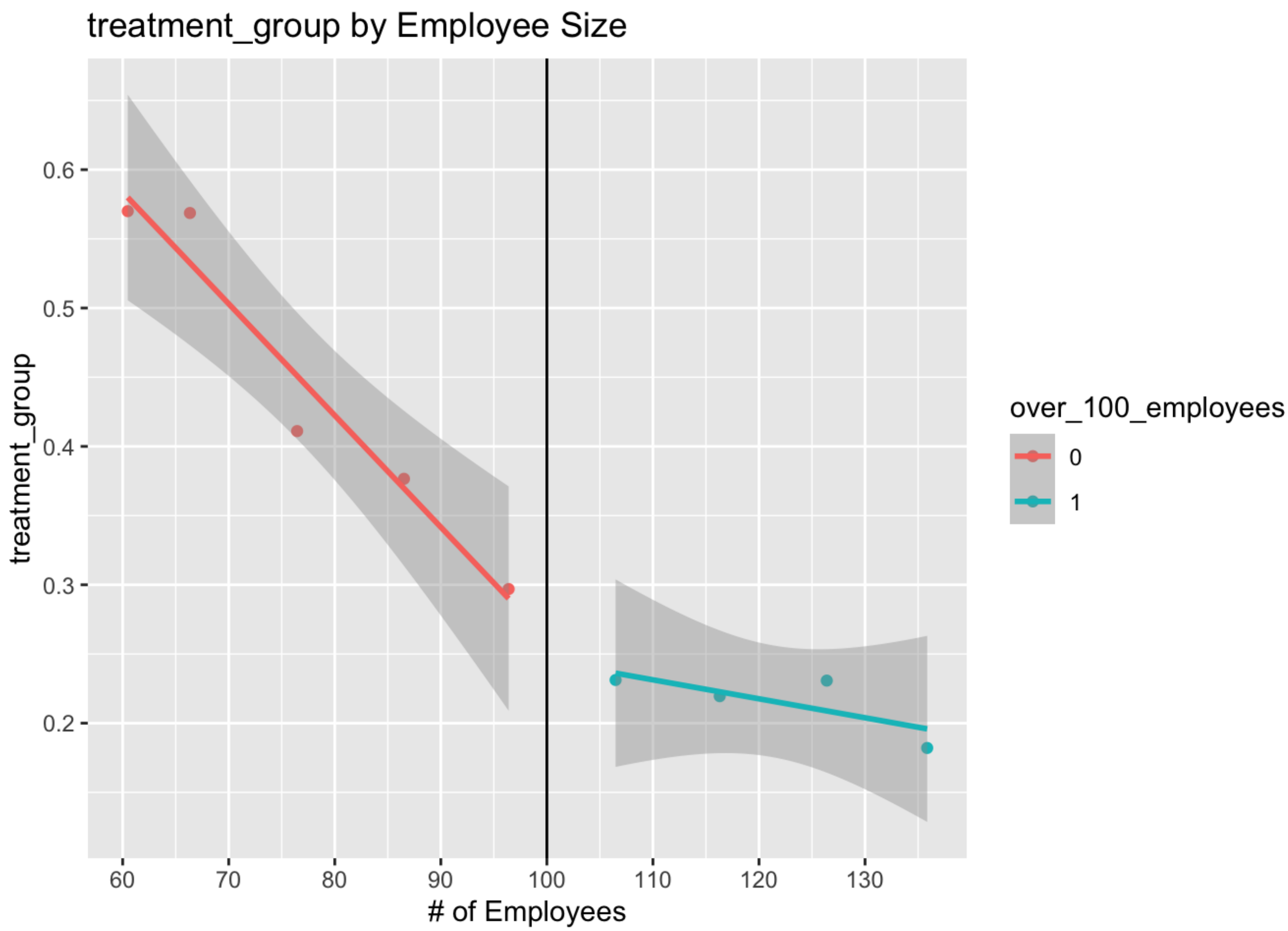

adopt_t by Employee Size
adopt_t
0.6
0.5
0.4
0.3
0.2
60
70
80
90
100
110
120
130
# of Employees
over_100_employees
0
1


post_comparison[, i]
0.5
0.4
0.3
0.2
60
80
100
120
post_comparison$employment_t

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Saving 7 x 5 in image
```

```
## `geom_smooth()` using formula 'y ~ x'
```



2.3c and d

```
### Time-Fixed Effects: Controlling for year allows for the control of year-by-year gro
wth or recession, and controlling for month allows for the control of seasonal cycles o
f sales, revenue, etc.
### Regression robust to other time fixed-effect specifications, such as by quarter
### N-eligible_post is indicator, 1 if over 100 employees after program begins
### Created Dataset of only observations post program start for Models1-4
Dataset_Regression_Post <- Dataset_Regression %>%
  filter(date >= as.Date("2013-01-01"))


model_1 <- (lm(sales_t ~ n_eligible_post + employment_t_lag + firm_id + year + month, d
ata = Dataset_Regression_Post ))

model_2 <- (lm(revenue_t ~ n_eligible_post + employment_t_lag + firm_id + year + month,
data = Dataset_Regression ))

model_3 <- (lm(wage_bill_t ~ n_eligible_post + employment_t_lag + firm_id + year + mont
h, data = Dataset_Regression_Post ))

model_4 <- (lm(employment_t ~ n_eligible_post + employment_t_lag + firm_id + year + mon
th, data = Dataset_Regression_Post ))

model_5 <- lm(adopt_t ~ n_eligible_post + employment_t_lag + firm_id + year + month, da
ta = Dataset_Regression ) ### initially used probit model, but got very high standard e
rrors. Linear model is not technically best tool to use since we are focused on a binar
y outcome, but the coefficient on the variable of interest makes some sense.

model_6 <- lm(treatment_group ~ n_eligible_post + employment_t_lag + firm_id + year + m
onth, data = Dataset_Regression ) ### initially used probit model, but got very high st
andard errors. Linear model is not technically best tool to use since we are focused on
a binary outcome, but the coefficient on the variable of interest makes some sense.

stargazer(model_1, model_2, model_3, model_4, model_5, model_6, keep = c("n_eligible_po
st", "employment_t_lag", "constant"),
          title ="Change after Program Implementation in Business Outcomes of Non-Eligi
ble Firms ",
          covariate.labels= c("Not_Elibile x Post_Implementation Dummy", "Employment in
t-1", "Constant"),
          dep.var.labels = c("Monthly Sales","Monthly Revenue", "Monthly Wage Bill", "M
onthly Employees", "Adoption of Intervention", "Treatment Assignment"),
          omit.stat = c("f","adj.rsq","ser"))
```