# P-value Alternatives

Sam Kloese and Jackson Crowther

2022-10-19

## Load Packages and Data

```
knitr::opts_chunk$set(echo = TRUE)
library(CASdatasets) # For datasets
```

```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sp
```

```
library(tidyverse) # For data manipulation
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::first()  masks xts::first()
## x dplyr::lag()    masks stats::lag()
## x dplyr::last()   masks xts::last()
```

```
library(knitr) # For generating markdowns
library(webshot) # For putting images in a PDF
library(glmnet) # For creating the GLM
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-4
```

```
library(ggplot2) # For plotting histograms
set.seed(23) # For reproducibility
data(pg17trainpol) # Load policy data
data(pg17trainclaim) # Load claims data
```

## Preliminary Adjustments

```
# Take a look at our data
glimpse(pg17trainclaim)
```

```
## Rows: 14,243
## Columns: 6
## $ id_client    <fct> A00000009, A00000016, A00000026, A00000040, A00000056, A0~
## $ id_vehicle   <fct> V01, V01, V01, V01, V01, V01, V01, V01, V01, V01, V01, V0~
## $ id_year      <fct> Year 0, Year 0, Year 0, Year 0, Year 0, Year 0, Year 0, Y~
## $ id_claim     <fct> CL01, CL01, CL01, CL01, CL01, CL01, CL01, CL01, CL01, CL0~
## $ claim_nb     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ claim_amount <dbl> 927.16, 555.48, 478.01, 512.83, 1236.00, 158.28, -477.91,~
```

```
glimpse(pg17trainpol)
```

```
## Rows: 100,000
## Columns: 31
## $ id_client        <fct> A00000001, A00000002, A00000003, A00000004, A00000005~
## $ id_vehicle       <fct> V01, V01, V01, V01, V01, V01, V01, V01, V01, V01, V01~
## $ id_policy        <fct> A00000001-V01, A00000002-V01, A00000003-V01, A0000000~
## $ id_year          <fct> Year 0, Year 0, Year 0, Year 0, Year 0, Year 0, Year ~
## $ pol_bonus        <dbl> 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.64,~
## $ pol_coverage     <fct> Maxi, Maxi, Maxi, Median2, Maxi, Median1, Maxi, Maxi,~
## $ pol_duration     <int> 29, 3, 2, 22, 16, 5, 5, 2, 5, 26, 8, 4, 21, 25, 9, 6,~
## $ pol_sit_duration <int> 9, 1, 2, 1, 4, 1, 3, 2, 1, 6, 1, 4, 1, 8, 1, 2, 3, 3,~
## $ pol_pay_freq     <fct> Biannual, Biannual, Yearly, Yearly, Biannual, Monthly~
## $ pol_payd         <fct> No, No, No, No, No, No, No, No, No, No, No, No, Yes, ~
## $ pol_usage        <fct> Retired, Retired, WorkPrivate, WorkPrivate, Retired, ~
## $ pol_insee_code   <fct> 36233, 92073, 92026, 78537, 38544, 76259, 38547, 3712~
## $ drv_drv2         <fct> No, No, No, Yes, Yes, No, No, No, No, Yes, Yes, No, N~
## $ drv_age1         <int> 85, 69, 37, 81, 62, 68, 77, 64, 38, 59, 66, 61, 65, 7~
## $ drv_age2         <int> 0, 0, 0, 21, 68, 0, 0, 0, 0, 33, 32, 0, 0, 0, 34, 56,~
## $ drv_sex1         <fct> M, M, M, M, F, M, M, M, M, M, M, M, F, M, M, M, F, M,~
## $ drv_sex2         <fct> , , , F, M, , , , , F, M, , , , F, F, , , , , M, F, ,~
## $ drv_age_lic1     <int> 62, 39, 18, 54, 37, 40, 55, 37, 19, 41, 45, 43, 43, 4~
## $ drv_age_lic2     <int> 0, 0, 0, 3, 48, 0, 0, 0, 0, 15, 14, 0, 0, 0, 14, 37, ~
## $ vh_age           <int> 10, 4, 11, 16, 11, 14, 7, 11, 9, 6, 4, 5, 5, 13, 1, 2~
## $ vh_cyl           <int> 1587, 2149, 1991, 1781, 1598, 1769, 1870, 1595, 1997,~
## $ vh_din           <int> 98, 170, 150, 90, 108, 60, 108, 101, 109, 90, 90, 127~
## $ vh_fuel          <fct> Gasoline, Diesel, Gasoline, Gasoline, Gasoline, Diese~
## $ vh_make          <fct> PEUGEOT, MERCEDES BENZ, BMW, VOLKSWAGEN, RENAULT, PEU~
## $ vh_model         <fct> 306, C220, Z3, GOLF, LAGUNA, 205, LAGUNA, A4, 307, PA~
## $ vh_sale_begin    <int> 10, 4, 12, 18, 13, 28, 10, 16, 9, 9, 4, 6, 7, 14, 3, ~
## $ vh_sale_end      <int> 9, 2, 11, 15, 11, 18, 6, 13, 7, 7, 3, 3, 4, 13, 1, 4,~
## $ vh_speed         <int> 182, 229, 210, 180, 195, 155, 193, 191, 183, 163, 180~
## $ vh_type          <fct> Tourism, Tourism, Tourism, Tourism, Tourism, Tourism,~
## $ vh_value         <int> 20700, 34250, 28661, 14407, 16770, 11564, 22450, 2053~
## $ vh_weight        <int> 1210, 1510, 1270, 1020, 1230, 850, 1350, 1195, 1260, ~
```

```
# Assemble data to model
# Some clients had more than 1 claim in a year
pg17trainclaim2 <- pg17trainclaim %>% # Aggregate claims to client and year
  group_by(id_client, id_year) %>%
  summarize(claim_count = n(),
            claim_amount = sum(claim_amount))
```

```
## ‘summarise()‘ has grouped output by ’id_client’. You can override using the
## ‘.groups‘ argument.
```

```
# Join the policy information and claims data
# If the client can't be found in the claims data, they had 0 claims for $0
pg17train <- pg17trainpol %>%
  left_join(pg17trainclaim2, by = c("id_client", "id_year")) %>%
  mutate(claim_count = replace_na(claim_count, replace = 0)) %>%
  mutate(claim_amount = replace_na(claim_amount, replace = 0)) %>%
  mutate(exposures = 1) %>% # Big assumption: All years are full years %>%
  mutate(drv_age1 = as.double(drv_age1)) %>%
  mutate(vh_age = as.double(vh_age)) %>%
  mutate(vh_din = as.double(vh_din))
dim(pg17train)
```

```
## [1] 100000     34
```

```
sum(pg17train$claim_count)
```

```
## [1] 16445
```

```
# Remove record with NA's
pg17train2 <- pg17train[complete.cases(pg17train),]
glimpse(pg17train2)
```

```
## Rows: 99,999
## Columns: 34
## $ id_client        <fct> A00000001, A00000002, A00000003, A00000004, A00000005~
## $ id_vehicle       <fct> V01, V01, V01, V01, V01, V01, V01, V01, V01, V01, V01~
## $ id_policy        <fct> A00000001-V01, A00000002-V01, A00000003-V01, A0000000~
## $ id_year          <fct> Year 0, Year 0, Year 0, Year 0, Year 0, Year 0, Year ~
## $ pol_bonus        <dbl> 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.64,~
## $ pol_coverage     <fct> Maxi, Maxi, Maxi, Median2, Maxi, Median1, Maxi, Maxi,~
## $ pol_duration     <int> 29, 3, 2, 22, 16, 5, 5, 2, 5, 26, 8, 4, 21, 25, 9, 6,~
## $ pol_sit_duration <int> 9, 1, 2, 1, 4, 1, 3, 2, 1, 6, 1, 4, 1, 8, 1, 2, 3, 3,~
## $ pol_pay_freq     <fct> Biannual, Biannual, Yearly, Yearly, Biannual, Monthly~
## $ pol_payd         <fct> No, No, No, No, No, No, No, No, No, No, No, No, Yes, ~
## $ pol_usage        <fct> Retired, Retired, WorkPrivate, WorkPrivate, Retired, ~
## $ pol_insee_code   <fct> 36233, 92073, 92026, 78537, 38544, 76259, 38547, 3712~
## $ drv_drv2         <fct> No, No, No, Yes, Yes, No, No, No, No, Yes, Yes, No, N~
## $ drv_age1         <dbl> 85, 69, 37, 81, 62, 68, 77, 64, 38, 59, 66, 61, 65, 7~
## $ drv_age2         <int> 0, 0, 0, 21, 68, 0, 0, 0, 0, 33, 32, 0, 0, 0, 34, 56,~
## $ drv_sex1         <fct> M, M, M, M, F, M, M, M, M, M, M, M, F, M, M, M, F, M,~
## $ drv_sex2         <fct> , , , F, M, , , , , F, M, , , , F, F, , , , , M, F, ,~
## $ drv_age_lic1     <int> 62, 39, 18, 54, 37, 40, 55, 37, 19, 41, 45, 43, 43, 4~
## $ drv_age_lic2     <int> 0, 0, 0, 3, 48, 0, 0, 0, 0, 15, 14, 0, 0, 0, 14, 37, ~
## $ vh_age           <dbl> 10, 4, 11, 16, 11, 14, 7, 11, 9, 6, 4, 5, 5, 13, 1, 2~
## $ vh_cyl           <int> 1587, 2149, 1991, 1781, 1598, 1769, 1870, 1595, 1997,~
## $ vh_din           <dbl> 98, 170, 150, 90, 108, 60, 108, 101, 109, 90, 90, 127~
## $ vh_fuel          <fct> Gasoline, Diesel, Gasoline, Gasoline, Gasoline, Diese~
## $ vh_make          <fct> PEUGEOT, MERCEDES BENZ, BMW, VOLKSWAGEN, RENAULT, PEU~
## $ vh_model         <fct> 306, C220, Z3, GOLF, LAGUNA, 205, LAGUNA, A4, 307, PA~
## $ vh_sale_begin    <int> 10, 4, 12, 18, 13, 28, 10, 16, 9, 9, 4, 6, 7, 14, 3, ~
## $ vh_sale_end      <int> 9, 2, 11, 15, 11, 18, 6, 13, 7, 7, 3, 3, 4, 13, 1, 4,~
## $ vh_speed         <int> 182, 229, 210, 180, 195, 155, 193, 191, 183, 163, 180~
## $ vh_type          <fct> Tourism, Tourism, Tourism, Tourism, Tourism, Tourism,~
## $ vh_value         <int> 20700, 34250, 28661, 14407, 16770, 11564, 22450, 2053~
## $ vh_weight        <int> 1210, 1510, 1270, 1020, 1230, 850, 1350, 1195, 1260, ~
## $ claim_count      <int> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0,~
## $ claim_amount     <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 927.1~
## $ exposures        <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
```

```
rm(pg17train, pg17trainclaim, pg17trainclaim2, pg17trainpol)
```

# Bin Numeric Variables

```r
# Bin continuous numeric variables into categories
pg17train3 <- pg17train2 %>%
  mutate(drv_age_bucket = case_when(drv_age1 >= 16 & drv_age1 <=20 ~ "drv_age_16_20",
                                    drv_age1 >= 21 & drv_age1 <=30 ~ "drv_age_21_30",
                                    drv_age1 >= 31 & drv_age1 <=40 ~ "drv_age_31_40",
                                    drv_age1 >= 41 & drv_age1 <=50 ~ "drv_age_41_50",
                                    drv_age1 >= 51 & drv_age1 <=60 ~ "drv_age_51_60",
                                    drv_age1 >= 61 & drv_age1 <=120 ~ "drv_age_61_120"),
         vh_age_bucket = case_when(vh_age >= 0 & vh_age <= 5 ~ "vh_age_0_5",
                                   vh_age >= 6 & vh_age <= 10 ~ "vh_age_6_10",
                                   vh_age >= 11 & vh_age <= 100 ~ "vh_age_11_100"),
         vh_din_bucket = case_when(vh_din >= 0 & vh_din <= 50 ~ "vh_din_0_50",
                                   vh_din >= 51 & vh_din <= 100 ~ "vh_din_51_100",
                                   vh_din >= 101 & vh_din <= 150 ~ "vh_din_101_150",
                                   vh_din >= 151 & vh_din <= 999 ~ "vh_din_151_999"))
```

# One Hot Encoding

```
## ---- Hot Coding ----

pg17train3 <- pg17train3 %>%
  mutate(indicator = 1) %>%
  spread(key = pol_coverage, value = indicator, fill = 0)
pg17train3 <- pg17train3 %>%
  mutate(indicator = 1) %>%
  spread(key = pol_pay_freq, value = indicator, fill = 0)

pg17train3 <- pg17train3 %>%
  mutate(indicator = 1) %>%
  spread(key = pol_usage, value = indicator, fill = 0)
pg17train3 <- pg17train3 %>%
  mutate(indicator = 1) %>%
  spread(key = drv_drv2, value = indicator, fill = 0)
pg17train3 <- pg17train3 %>%
  mutate(indicator = 1) %>%
  spread(key = drv_sex1, value = indicator, fill = 0)
pg17train3 <- pg17train3 %>%
  mutate(indicator = 1) %>%
  spread(key = vh_fuel, value = indicator, fill = 0)
pg17train3 <- pg17train3 %>%
  mutate(indicator = 1) %>%
  spread(key = vh_type, value = indicator, fill = 0)

pg17train3 <- pg17train3 %>%
  mutate(drv_age_bucket1 = drv_age_bucket) %>%
  mutate(indicator = 1) %>%
  spread(key = drv_age_bucket1, value = indicator, fill = 0)
pg17train3 <- pg17train3 %>%
  mutate(vh_age_bucket1 = vh_age_bucket) %>%
  mutate(indicator = 1) %>%
  spread(key = vh_age_bucket1, value = indicator, fill = 0)
pg17train3 <- pg17train3 %>%
  mutate(vh_din_bucket1 = vh_din_bucket) %>%
  mutate(indicator = 1) %>%
  spread(key = vh_din_bucket1, value = indicator, fill = 0)

# Remove columns we don't want to use as predictor variables
# Mostly removed for simplicity of this example
pg17train3 <- pg17train3 %>%
  select(-pol_payd, -pol_insee_code,-drv_age2, -drv_sex2, -drv_age_lic2,
         -vh_model, -vh_make, -id_vehicle, -id_policy, -id_year)
names(pg17train3)[21:24] <- paste("coverage",names(pg17train3)[21:24],sep="_")
names(pg17train3)[25:28] <- paste("pay",names(pg17train3)[25:28],sep="_")
names(pg17train3)[29:32] <- paste("usage",names(pg17train3)[29:32],sep="_")
names(pg17train3)[33]<- "second_driver_No"
names(pg17train3)[34]<- "second_driver_Yes"
names(pg17train3)[35]<- "driver_gender_F"
names(pg17train3)[36]<- "driver_gender_M"
names(pg17train3)[37:39] <- paste("fuel",names(pg17train3)[37:39],sep="_")
```

```r
names(pg17train3)[40:41] <- paste("type",names(pg17train3)[40:41],sep="_")

pg17train3 <- pg17train3 %>%
  select(-second_driver_No,-driver_gender_M) %>%
  filter(claim_amount >= 0) # eliminate small number of negative claims amounts
glimpse(pg17train3)
```

```
## Rows: 98,735
## Columns: 52
## $ id_client          <fct> A00000001, A00000002, A00000003, A00000004, A000000~
## $ pol_bonus          <dbl> 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.6~
## $ pol_duration       <int> 29, 3, 2, 22, 16, 5, 5, 2, 5, 26, 8, 4, 21, 25, 9, ~
## $ pol_sit_duration   <int> 9, 1, 2, 1, 4, 1, 3, 2, 1, 6, 1, 4, 1, 8, 1, 2, 3, ~
## $ drv_age1           <dbl> 85, 69, 37, 81, 62, 68, 77, 64, 38, 59, 66, 61, 65,~
## $ drv_age_lic1       <int> 62, 39, 18, 54, 37, 40, 55, 37, 19, 41, 45, 43, 43,~
## $ vh_age             <dbl> 10, 4, 11, 16, 11, 14, 7, 11, 9, 6, 4, 5, 5, 13, 1,~
## $ vh_cyl             <int> 1587, 2149, 1991, 1781, 1598, 1769, 1870, 1595, 199~
## $ vh_din             <dbl> 98, 170, 150, 90, 108, 60, 108, 101, 109, 90, 90, 1~
## $ vh_sale_begin      <int> 10, 4, 12, 18, 13, 28, 10, 16, 9, 9, 4, 6, 7, 14, 3~
## $ vh_sale_end        <int> 9, 2, 11, 15, 11, 18, 6, 13, 7, 7, 3, 3, 4, 13, 1, ~
## $ vh_speed           <int> 182, 229, 210, 180, 195, 155, 193, 191, 183, 163, 1~
## $ vh_value           <int> 20700, 34250, 28661, 14407, 16770, 11564, 22450, 20~
## $ vh_weight          <int> 1210, 1510, 1270, 1020, 1230, 850, 1350, 1195, 1260~
## $ claim_count        <int> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, ~
## $ claim_amount       <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 927~
## $ exposures          <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ drv_age_bucket     <chr> "drv_age_61_120", "drv_age_61_120", "drv_age_31_40"~
## $ vh_age_bucket      <chr> "vh_age_6_10", "vh_age_0_5", "vh_age_11_100", "vh_a~
## $ vh_din_bucket      <chr> "vh_din_51_100", "vh_din_151_999", "vh_din_101_150"~
## $ coverage_Maxi      <dbl> 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ coverage_Median1   <dbl> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ coverage_Median2   <dbl> 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ coverage_Mini      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ pay_Biannual       <dbl> 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, ~
## $ pay_Monthly        <dbl> 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, ~
## $ pay_Quarterly      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, ~
## $ pay_Yearly         <dbl> 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, ~
## $ usage_AllTrips     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ usage_Professional <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ usage_Retired      <dbl> 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, ~
## $ usage_WorkPrivate  <dbl> 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, ~
## $ second_driver_Yes  <dbl> 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, ~
## $ driver_gender_F    <dbl> 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, ~
## $ fuel_Diesel        <dbl> 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, ~
## $ fuel_Gasoline      <dbl> 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, ~
## $ fuel_Hybrid        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ type_Commercial    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ type_Tourism       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ drv_age_16_20      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ drv_age_21_30      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ drv_age_31_40      <dbl> 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, ~
## $ drv_age_41_50      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, ~
## $ drv_age_51_60      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, ~
## $ drv_age_61_120     <dbl> 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, ~
## $ vh_age_0_5         <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, ~
## $ vh_age_11_100      <dbl> 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, ~
## $ vh_age_6_10        <dbl> 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, ~
## $ vh_din_0_50        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ vh_din_101_150     <dbl> 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, ~
## $ vh_din_151_999     <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ vh_din_51_100      <dbl> 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, ~
```

```
rm(pg17train2)
gc()
```

```
##           used  (Mb) gc trigger  (Mb) max used  (Mb)
## Ncells 2467483 131.8    4107382 219.4  4107382 219.4
## Vcells 9295008  71.0   32108883 245.0 32108883 245.0
```

```
# Calculate the frequency column
pg17train3 <- pg17train3 %>%
  mutate(frequency = claim_count/exposures)
```

## Generate Columns with Random Data

This is used in an experiment later in the script

```
pg17train3 <- pg17train3 %>%
  mutate(random001 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random002 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random003 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random004 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random005 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random006 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random007 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random008 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random009 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random010 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random011 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random012 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random013 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random014 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random015 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random016 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random017 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random018 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random019 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random020 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random021 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random022 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random023 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random024 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random025 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random026 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random027 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random028 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random029 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random030 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random031 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random032 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random033 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random034 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random035 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random036 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random037 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random038 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random039 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random040 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random041 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random042 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random043 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random044 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random045 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random046 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random047 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random048 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random049 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random050 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random051 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random052 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random053 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random054 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random055 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random056 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random057 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random058 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random059 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random060 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random061 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random062 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random063 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random064 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random065 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random066 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random067 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random068 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random069 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random070 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random071 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random072 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random073 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random074 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random075 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random076 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random077 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random078 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random079 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random080 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random081 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random082 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random083 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random084 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random085 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random086 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random087 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random088 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random089 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random090 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random091 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random092 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random093 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random094 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random095 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random096 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random097 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random098 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random099 = sample(1:5,size = nrow(pg17train3),replace = TRUE)) %>%
  mutate(random100 = sample(1:5,size = nrow(pg17train3),replace = TRUE))
```

### Split Train/Test Data

```
# 80% of clients will be used in training
# 20% of clients will be used in testing
clients_unique <- unique(pg17train3$id_client)
clients_index <- sample(1:90380,
                        size = 72304,
                        replace = FALSE)
clients_train <- clients_unique[clients_index]
training_data <- pg17train3 %>%
  filter(id_client %in% clients_train) %>%
  select(-id_client)
testing_data <- pg17train3 %>%
  filter(!(id_client %in% clients_train))
testing_data <- testing_data %>%
  select(-id_client)
glimpse(training_data)
```

```
## Rows: 78,952
## Columns: 152
## $ pol_bonus          <dbl> 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.64, 0.5~
## $ pol_duration       <int> 29, 3, 2, 22, 5, 5, 2, 5, 26, 8, 4, 21, 9, 6, 6, 8,~
## $ pol_sit_duration   <int> 9, 1, 2, 1, 1, 3, 2, 1, 6, 1, 4, 1, 1, 2, 3, 3, 4, ~
## $ drv_age1           <dbl> 85, 69, 37, 81, 68, 77, 64, 38, 59, 66, 61, 65, 38,~
## $ drv_age_lic1       <int> 62, 39, 18, 54, 40, 55, 37, 19, 41, 45, 43, 43, 19,~
## $ vh_age             <dbl> 10, 4, 11, 16, 14, 7, 11, 9, 6, 4, 5, 5, 1, 2, 8, 2~
## $ vh_cyl             <int> 1587, 2149, 1991, 1781, 1769, 1870, 1595, 1997, 199~
## $ vh_din             <dbl> 98, 170, 150, 90, 60, 108, 101, 109, 90, 90, 127, 6~
## $ vh_sale_begin      <int> 10, 4, 12, 18, 28, 10, 16, 9, 9, 4, 6, 7, 3, 5, 10,~
## $ vh_sale_end        <int> 9, 2, 11, 15, 18, 6, 13, 7, 7, 3, 3, 4, 1, 4, 8, 23~
## $ vh_speed           <int> 182, 229, 210, 180, 155, 193, 191, 183, 163, 180, 1~
## $ vh_value           <int> 20700, 34250, 28661, 14407, 11564, 22450, 20535, 23~
## $ vh_weight          <int> 1210, 1510, 1270, 1020, 850, 1350, 1195, 1260, 1110~
## $ claim_count        <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, ~
## $ claim_amount       <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 927.16, 0~
## $ exposures          <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ drv_age_bucket     <chr> "drv_age_61_120", "drv_age_61_120", "drv_age_31_40"~
## $ vh_age_bucket      <chr> "vh_age_6_10", "vh_age_0_5", "vh_age_11_100", "vh_a~
## $ vh_din_bucket      <chr> "vh_din_51_100", "vh_din_151_999", "vh_din_101_150"~
## $ coverage_Maxi      <dbl> 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, ~
## $ coverage_Median1   <dbl> 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ coverage_Median2   <dbl> 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, ~
## $ coverage_Mini      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ pay_Biannual       <dbl> 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, ~
## $ pay_Monthly        <dbl> 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, ~
## $ pay_Quarterly      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, ~
## $ pay_Yearly         <dbl> 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, ~
## $ usage_AllTrips     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ usage_Professional <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ usage_Retired      <dbl> 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, ~
## $ usage_WorkPrivate  <dbl> 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, ~
## $ second_driver_Yes  <dbl> 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, ~
## $ driver_gender_F    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, ~
## $ fuel_Diesel         <dbl> 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, ~
## $ fuel_Gasoline       <dbl> 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, ~
## $ fuel_Hybrid         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ type_Commercial     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ type_Tourism        <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ drv_age_16_20       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ drv_age_21_30       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ drv_age_31_40       <dbl> 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, ~
## $ drv_age_41_50       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, ~
## $ drv_age_51_60       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, ~
## $ drv_age_61_120      <dbl> 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, ~
## $ vh_age_0_5          <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, ~
## $ vh_age_11_100       <dbl> 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, ~
## $ vh_age_6_10         <dbl> 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, ~
## $ vh_din_0_50         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ vh_din_101_150      <dbl> 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, ~
## $ vh_din_151_999      <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ vh_din_51_100       <dbl> 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, ~
## $ frequency           <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, ~
## $ random001           <int> 5, 4, 3, 1, 5, 2, 1, 5, 1, 4, 3, 5, 4, 2, 1, 2, 3, ~
## $ random002           <int> 5, 3, 4, 3, 5, 2, 4, 4, 3, 3, 3, 2, 4, 3, 4, 2, 5, ~
## $ random003           <int> 4, 2, 1, 3, 5, 2, 1, 4, 3, 5, 4, 5, 4, 4, 3, 3, 1, ~
## $ random004           <int> 2, 1, 2, 4, 3, 5, 3, 3, 4, 5, 1, 2, 1, 3, 1, 2, 1, ~
## $ random005           <int> 2, 2, 5, 1, 1, 4, 5, 5, 3, 5, 1, 3, 3, 3, 4, 1, 4, ~
## $ random006           <int> 4, 1, 3, 5, 3, 2, 1, 5, 1, 2, 5, 1, 3, 3, 4, 4, 5, ~
## $ random007           <int> 3, 5, 1, 2, 4, 3, 3, 3, 4, 4, 5, 5, 2, 5, 1, 1, 2, ~
## $ random008           <int> 2, 3, 4, 5, 4, 1, 4, 2, 2, 4, 5, 3, 3, 5, 2, 5, 4, ~
## $ random009           <int> 4, 2, 3, 5, 4, 2, 3, 4, 1, 2, 2, 5, 1, 3, 2, 1, 2, ~
## $ random010           <int> 3, 3, 1, 4, 2, 2, 1, 3, 4, 1, 2, 5, 5, 3, 3, 2, 5, ~
## $ random011           <int> 1, 3, 4, 3, 3, 5, 1, 5, 2, 2, 5, 2, 2, 4, 3, 1, 4, ~
## $ random012           <int> 2, 5, 1, 3, 4, 3, 2, 4, 5, 2, 5, 5, 3, 3, 4, 1, 1, ~
## $ random013           <int> 3, 4, 1, 4, 5, 5, 3, 1, 2, 2, 2, 1, 1, 4, 2, 1, 2, ~
## $ random014           <int> 5, 2, 2, 5, 1, 5, 4, 4, 3, 2, 4, 1, 1, 2, 1, 1, 1, ~
## $ random015           <int> 1, 2, 1, 3, 3, 1, 2, 2, 2, 1, 1, 1, 5, 1, 1, 2, 5, ~
## $ random016           <int> 3, 5, 5, 3, 5, 1, 4, 2, 3, 3, 5, 3, 3, 5, 5, 4, 5, ~
## $ random017           <int> 4, 3, 2, 1, 5, 2, 2, 5, 2, 2, 4, 5, 3, 5, 4, 5, 5, ~
## $ random018           <int> 4, 1, 4, 2, 4, 5, 5, 1, 2, 1, 5, 2, 2, 4, 4, 1, 5, ~
## $ random019           <int> 4, 1, 4, 3, 3, 2, 5, 4, 4, 3, 4, 2, 4, 4, 1, 3, 1, ~
## $ random020           <int> 5, 4, 2, 1, 2, 5, 3, 3, 5, 3, 3, 5, 2, 3, 4, 1, 2, ~
## $ random021           <int> 1, 2, 4, 3, 3, 1, 2, 5, 3, 4, 1, 3, 5, 3, 5, 2, 5, ~
## $ random022           <int> 4, 1, 5, 3, 4, 1, 3, 3, 3, 1, 4, 5, 4, 2, 3, 2, 1, ~
## $ random023           <int> 5, 3, 4, 4, 2, 1, 5, 5, 2, 4, 3, 2, 1, 1, 5, 4, 2, ~
## $ random024           <int> 4, 1, 4, 4, 2, 1, 3, 3, 3, 5, 5, 5, 4, 5, 2, 2, 1, ~
## $ random025           <int> 1, 2, 5, 2, 4, 3, 5, 4, 4, 4, 5, 3, 4, 1, 1, 2, 4, ~
## $ random026           <int> 5, 4, 3, 1, 1, 5, 2, 5, 2, 1, 3, 5, 3, 2, 3, 5, 3, ~
## $ random027           <int> 3, 4, 3, 1, 4, 5, 2, 5, 3, 5, 1, 3, 2, 3, 4, 5, 1, ~
## $ random028           <int> 3, 4, 1, 1, 3, 5, 5, 5, 5, 5, 5, 3, 2, 5, 1, 4, 1, ~
## $ random029           <int> 3, 5, 2, 1, 5, 1, 2, 5, 1, 4, 4, 4, 4, 4, 3, 1, 1, ~
## $ random030           <int> 1, 1, 1, 2, 4, 2, 4, 4, 2, 5, 3, 3, 3, 1, 2, 5, 1, ~
## $ random031           <int> 3, 1, 1, 3, 2, 5, 1, 4, 1, 4, 4, 1, 2, 5, 5, 3, 4, ~
## $ random032           <int> 4, 2, 4, 1, 5, 5, 1, 1, 4, 2, 1, 4, 2, 4, 4, 5, 3, ~
## $ random033           <int> 2, 4, 1, 2, 3, 5, 5, 1, 4, 5, 5, 2, 3, 4, 5, 4, 1, ~
## $ random034           <int> 5, 1, 4, 4, 1, 2, 2, 1, 1, 1, 5, 2, 1, 5, 4, 1, 3, ~
## $ random035           <int> 3, 2, 5, 3, 5, 1, 4, 3, 4, 3, 4, 1, 3, 4, 3, 5, 4, ~
## $ random036          <int> 2, 5, 2, 5, 2, 2, 4, 3, 2, 3, 4, 2, 4, 4, 2, 2, 3, ~
## $ random037          <int> 3, 2, 3, 1, 2, 3, 5, 1, 3, 4, 2, 3, 1, 1, 4, 4, 1, ~
## $ random038          <int> 1, 3, 5, 5, 4, 3, 5, 1, 5, 4, 4, 5, 1, 5, 5, 3, 5, ~
## $ random039          <int> 4, 1, 1, 1, 3, 3, 1, 4, 4, 5, 1, 3, 5, 1, 1, 5, 3, ~
## $ random040          <int> 2, 3, 3, 2, 2, 5, 4, 4, 3, 5, 5, 3, 2, 2, 5, 5, 4, ~
## $ random041          <int> 5, 5, 1, 1, 2, 1, 1, 2, 1, 2, 3, 2, 3, 1, 3, 3, 4, ~
## $ random042          <int> 2, 5, 4, 2, 2, 3, 5, 4, 5, 2, 2, 5, 1, 2, 3, 5, 2, ~
## $ random043          <int> 1, 5, 5, 5, 4, 2, 4, 2, 2, 5, 1, 4, 4, 1, 2, 3, 1, ~
## $ random044          <int> 1, 1, 4, 5, 4, 3, 1, 1, 1, 4, 1, 1, 3, 2, 2, 5, 3, ~
## $ random045          <int> 4, 1, 3, 2, 5, 1, 4, 1, 2, 4, 5, 1, 3, 5, 4, 3, 3, ~
## $ random046          <int> 3, 5, 5, 2, 4, 4, 5, 2, 4, 1, 5, 3, 3, 1, 1, 2, 2, ~
## $ random047          <int> 3, 5, 3, 5, 4, 5, 3, 1, 2, 4, 5, 1, 3, 2, 5, 4, 3, ~
## $ random048          <int> 4, 5, 4, 2, 5, 2, 2, 1, 2, 4, 2, 5, 3, 1, 4, 1, 2, ~
## $ random049          <int> 3, 3, 1, 1, 3, 3, 2, 1, 4, 1, 2, 2, 5, 4, 4, 2, 3, ~
## $ random050          <int> 1, 1, 2, 3, 3, 5, 3, 4, 3, 3, 4, 5, 1, 4, 5, 3, 3, ~
## $ random051          <int> 4, 4, 5, 3, 2, 1, 2, 5, 4, 1, 5, 4, 4, 1, 5, 5, 3, ~
## $ random052          <int> 4, 4, 5, 1, 3, 5, 1, 1, 1, 3, 4, 4, 1, 4, 4, 1, 2, ~
## $ random053          <int> 2, 5, 5, 2, 5, 2, 2, 1, 2, 4, 3, 5, 5, 3, 3, 5, 2, ~
## $ random054          <int> 4, 5, 1, 2, 4, 4, 3, 1, 3, 5, 2, 3, 2, 3, 1, 5, 4, ~
## $ random055          <int> 1, 5, 1, 5, 3, 4, 5, 5, 3, 3, 3, 1, 1, 4, 4, 5, 3, ~
## $ random056          <int> 1, 5, 4, 4, 5, 2, 2, 5, 5, 1, 3, 1, 4, 5, 5, 5, 1, ~
## $ random057          <int> 2, 2, 2, 2, 3, 2, 1, 5, 5, 3, 3, 2, 2, 2, 5, 4, 4, ~
## $ random058          <int> 3, 5, 1, 2, 2, 3, 5, 2, 2, 2, 1, 3, 2, 2, 3, 4, 4, ~
## $ random059          <int> 4, 2, 2, 5, 5, 3, 5, 5, 4, 2, 2, 3, 2, 3, 4, 4, 1, ~
## $ random060          <int> 3, 3, 5, 4, 2, 1, 1, 3, 3, 1, 4, 3, 5, 3, 4, 1, 5, ~
## $ random061          <int> 3, 2, 2, 5, 3, 4, 4, 3, 4, 5, 4, 1, 5, 5, 4, 1, 3, ~
## $ random062          <int> 2, 4, 5, 3, 5, 5, 4, 2, 3, 4, 4, 4, 2, 5, 1, 5, 4, ~
## $ random063          <int> 5, 1, 2, 1, 1, 3, 3, 3, 2, 5, 5, 5, 3, 4, 3, 3, 2, ~
## $ random064          <int> 4, 5, 2, 2, 4, 4, 5, 1, 5, 4, 5, 3, 5, 5, 3, 4, 5, ~
## $ random065          <int> 3, 5, 5, 5, 2, 2, 2, 1, 3, 2, 5, 4, 4, 2, 1, 4, 1, ~
## $ random066          <int> 2, 3, 1, 1, 1, 1, 5, 4, 4, 1, 1, 5, 5, 5, 3, 2, 5, ~
## $ random067          <int> 3, 2, 5, 5, 3, 1, 5, 4, 5, 3, 3, 4, 4, 2, 5, 2, 4, ~
## $ random068          <int> 1, 4, 5, 4, 5, 4, 4, 5, 3, 4, 1, 4, 2, 1, 1, 1, 1, ~
## $ random069          <int> 5, 1, 4, 3, 1, 1, 1, 1, 3, 1, 4, 5, 3, 3, 5, 2, 4, ~
## $ random070          <int> 5, 1, 2, 2, 5, 1, 3, 1, 4, 5, 4, 4, 3, 3, 3, 4, 2, ~
## $ random071          <int> 4, 5, 1, 2, 2, 1, 5, 3, 1, 2, 3, 1, 3, 4, 4, 1, 5, ~
## $ random072          <int> 1, 5, 2, 1, 1, 4, 1, 2, 3, 4, 3, 5, 5, 3, 1, 3, 3, ~
## $ random073          <int> 3, 1, 4, 2, 3, 2, 3, 3, 5, 2, 4, 2, 4, 2, 1, 1, 3, ~
## $ random074          <int> 5, 1, 1, 4, 3, 1, 1, 1, 1, 2, 1, 3, 3, 2, 1, 5, 2, ~
## $ random075          <int> 3, 1, 2, 1, 5, 5, 3, 2, 3, 3, 5, 1, 3, 5, 5, 3, 3, ~
## $ random076          <int> 3, 4, 3, 3, 4, 1, 2, 5, 2, 5, 5, 1, 5, 2, 1, 1, 3, ~
## $ random077          <int> 3, 5, 5, 4, 5, 3, 3, 4, 2, 1, 2, 4, 3, 5, 4, 2, 4, ~
## $ random078          <int> 5, 4, 3, 2, 4, 4, 4, 1, 5, 5, 3, 3, 3, 5, 4, 5, 4, ~
## $ random079          <int> 5, 3, 2, 5, 1, 2, 3, 2, 4, 1, 1, 5, 3, 2, 5, 3, 5, ~
## $ random080          <int> 2, 2, 3, 3, 3, 5, 5, 5, 4, 2, 5, 4, 1, 5, 1, 5, 2, ~
## $ random081          <int> 5, 1, 2, 2, 1, 4, 5, 5, 4, 5, 5, 4, 3, 3, 2, 3, 5, ~
## $ random082          <int> 2, 4, 4, 5, 1, 5, 4, 4, 2, 1, 5, 5, 3, 1, 1, 3, 1, ~
## $ random083          <int> 5, 5, 2, 4, 3, 1, 5, 2, 3, 1, 3, 3, 1, 5, 1, 2, 2, ~
## $ random084          <int> 4, 4, 2, 3, 2, 2, 4, 2, 3, 2, 5, 4, 1, 2, 1, 2, 1, ~
## $ random085          <int> 2, 3, 3, 3, 2, 5, 4, 1, 5, 3, 3, 1, 2, 3, 5, 3, 3, ~
## $ random086          <int> 1, 4, 4, 1, 3, 5, 1, 4, 1, 5, 4, 3, 2, 4, 3, 4, 1, ~
## $ random087          <int> 4, 5, 4, 1, 1, 5, 3, 2, 1, 5, 5, 1, 4, 4, 5, 4, 3, ~
## $ random088          <int> 1, 3, 2, 3, 2, 4, 4, 3, 5, 3, 4, 1, 2, 4, 1, 2, 1, ~
## $ random089          <int> 5, 4, 2, 1, 2, 1, 1, 4, 3, 2, 2, 4, 1, 5, 1, 1, 1, ~
## $ random090          <int> 4, 1, 4, 3, 5, 4, 1, 1, 4, 2, 4, 5, 1, 4, 1, 4, 5, ~
## $ random091          <int> 4, 3, 3, 3, 3, 5, 1, 2, 2, 2, 4, 1, 2, 1, 3, 5, 2, ~
## $ random092          <int> 5, 3, 2, 3, 4, 5, 2, 1, 2, 4, 3, 2, 4, 3, 3, 4, 2, ~
## $ random093          <int> 2, 3, 4, 1, 4, 1, 5, 3, 1, 1, 5, 5, 4, 4, 2, 5, 2, ~
## $ random094          <int> 2, 5, 2, 2, 1, 2, 1, 5, 3, 2, 3, 4, 1, 4, 3, 4, 4, ~
## $ random095          <int> 1, 1, 3, 1, 2, 5, 5, 2, 5, 4, 5, 1, 5, 4, 2, 2, 1, ~
## $ random096          <int> 3, 4, 1, 5, 5, 1, 3, 2, 5, 2, 4, 3, 3, 2, 4, 4, 1, ~
## $ random097          <int> 4, 2, 1, 2, 3, 4, 2, 2, 1, 5, 2, 1, 3, 1, 3, 2, 2, ~
## $ random098          <int> 2, 1, 4, 3, 1, 4, 5, 3, 4, 5, 2, 2, 3, 4, 4, 2, 3, ~
## $ random099          <int> 1, 4, 2, 1, 3, 3, 3, 5, 5, 4, 3, 3, 4, 3, 2, 4, 5, ~
## $ random100          <int> 1, 4, 2, 1, 3, 2, 3, 3, 4, 4, 5, 3, 4, 2, 5, 3, 1, ~
```

```
rm(pg17train3, clients_index, clients_train, clients_unique)
gc()
```

```
##            used  (Mb) gc trigger  (Mb) max used  (Mb)
## Ncells  2375936 126.9    4107382 219.4  4107382 219.4
## Vcells 14005382 106.9   32108883 245.0 32108883 245.0
```

## Bootstrapping

```
# Bootstrap model1 here
# Create coefficients table
#Start with defining the number of iterations
itr <- 1000
coef_table_glmnet <- data.frame(Iteration = 1:itr,
                         AllTrips = 0,
                         Professional = 0,
                         Retired = 0, # Work private will be the base
                         Female = 0, # Male will be the base
                         drv_age_16_20 = 0,
                         drv_age_21_30 = 0,
                         drv_age_41_50 = 0, # 31-40 will be the base
                         drv_age_51_60 = 0,
                         drv_age_61_120 = 0,
                         vh_age_0_5 = 0,
                         vh_age_11_100 = 0, # 6-10 will be the base
                         vh_din_0_50 = 0,
                         vh_din_101_150 = 0, # 51-100 will be the base
                         vh_din_151_999 = 0)

# Select the response and the wanted explanatory variables
training_matrix <- as.matrix(select(training_data,
                                    c(frequency, usage_AllTrips, usage_Professional,
                                      usage_Retired,
                                      driver_gender_F, drv_age_16_20,
                                      drv_age_21_30, drv_age_41_50,
                                      drv_age_51_60, drv_age_61_120,
                                      vh_age_0_5, vh_age_11_100,
                                      vh_din_0_50, vh_din_101_150,
                                      vh_din_151_999)))

start_time <- Sys.time()

for(i in 1:itr){

  # Draw samples from the training matrix
  bootstrapTable <- slice_sample(.data = training_data,
                                 n = nrow(training_data),
                                 replace = TRUE)

  x_matrix <- bootstrapTable %>%
    select(usage_AllTrips, usage_Professional, usage_Retired, driver_gender_F,
           drv_age_16_20, drv_age_21_30, drv_age_41_50,
           drv_age_51_60, drv_age_61_120,
           vh_age_0_5, vh_age_11_100,
           vh_din_0_50, vh_din_101_150, vh_din_151_999) %>%
    as.matrix()

  y_matrix <- bootstrapTable %>%
    select(frequency) %>%
    as.matrix()
```

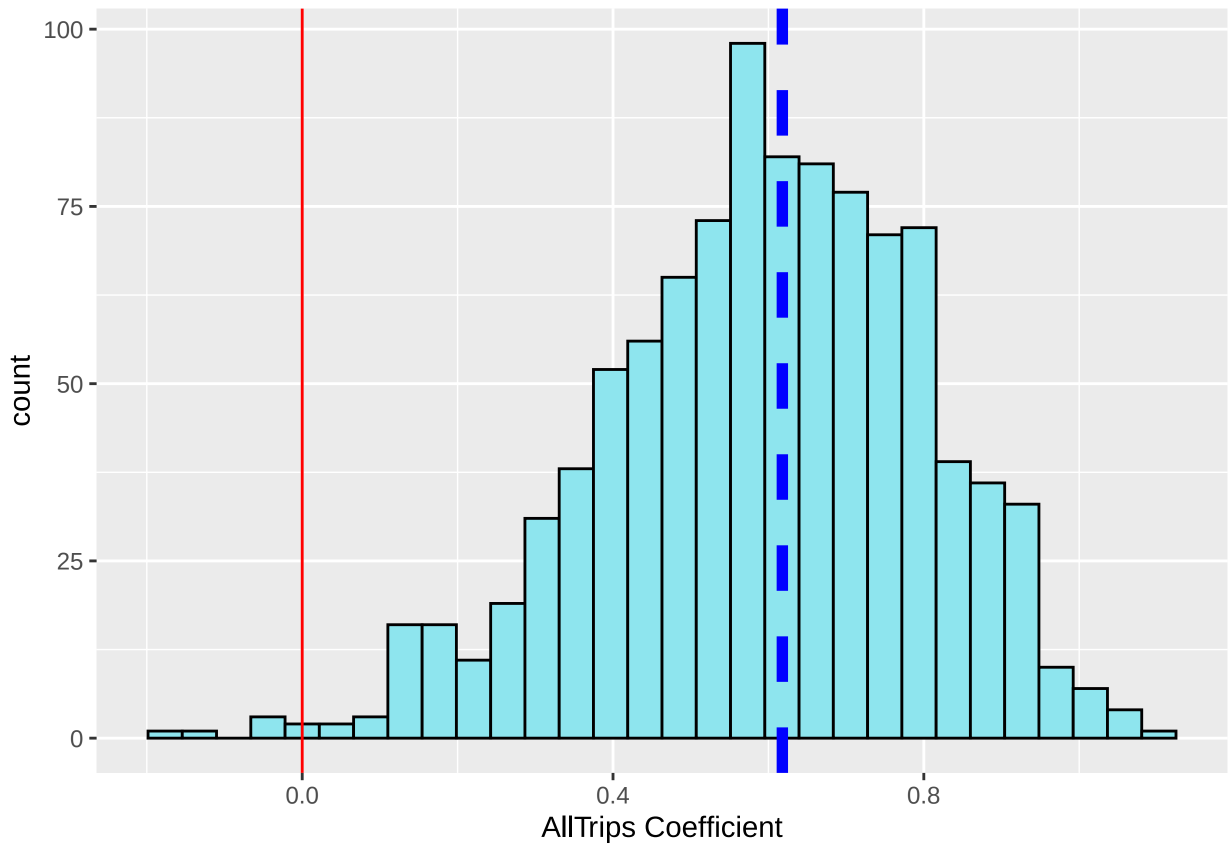
```r
  w_matrix <- bootstrapTable %>%
    select(exposures) %>%
    as.matrix()


  # Run the GLM net and insert the coefficients into the table recently created
  elastic_net <- glmnet(x = x_matrix,
                        y = y_matrix,
                        weights = w_matrix,
                        family = poisson(link = "log"),
                        alpha = 0.5,
                        lambda = .0001)
  coef_table_glmnet[i,2:15] <- coef(elastic_net)[2:15]
  # print(paste("iteration",i,"complete"))
}

end_time <- Sys.time()

end_time - start_time
```

```
## Time difference of 9.509972 mins
```

### Plot Coefficient Histograms

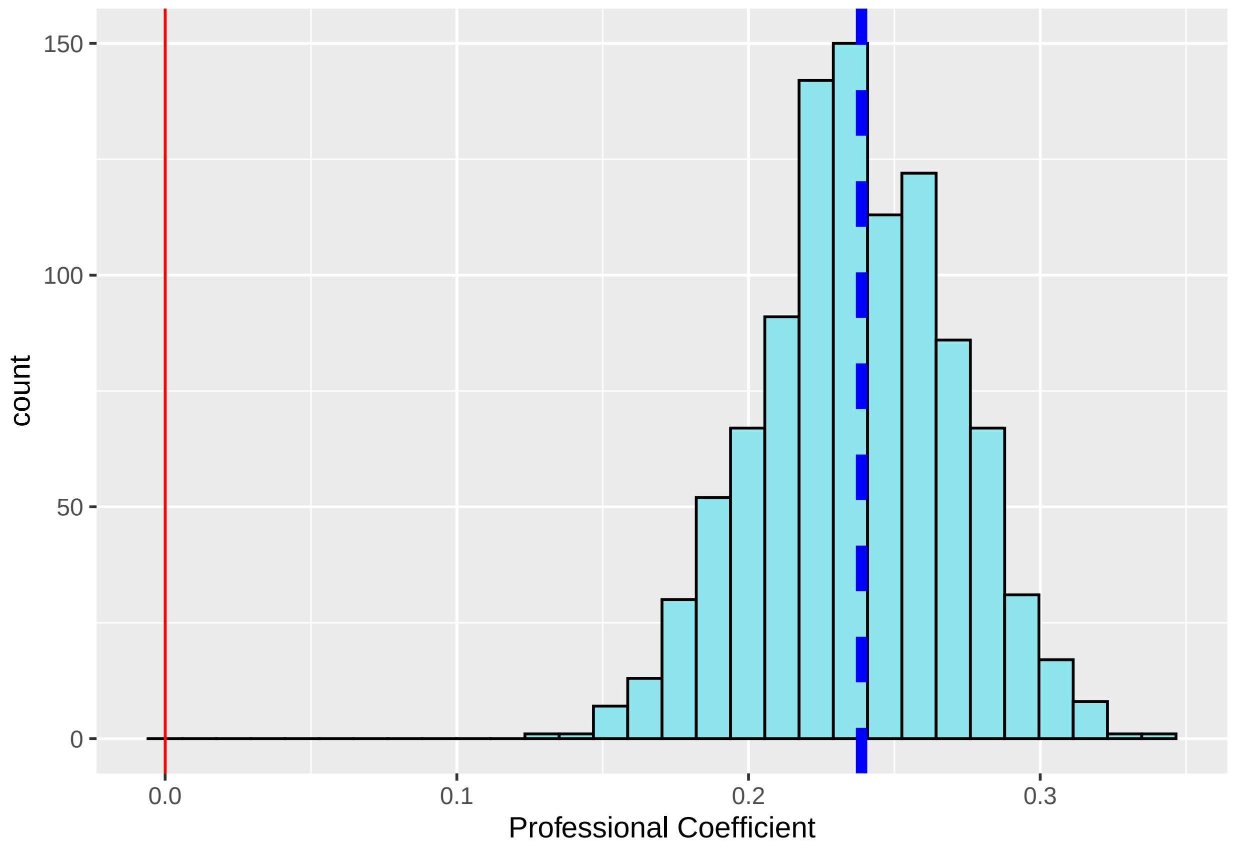
```
# Create final elastic net, will put plot coefficient as a vertical line
# Bootstrapped coefficients will show up in histogram

  x_matrix <- training_data %>%
    select(usage_AllTrips, usage_Professional, usage_Retired, driver_gender_F,
           drv_age_16_20, drv_age_21_30, drv_age_41_50,
           drv_age_51_60, drv_age_61_120,
           vh_age_0_5, vh_age_11_100,
           vh_din_0_50, vh_din_101_150, vh_din_151_999) %>%
    as.matrix()

  y_matrix <- training_data %>%
    select(frequency) %>%
    as.matrix()

  w_matrix <- training_data %>%
    select(exposures) %>%
    as.matrix()

selected_elastic_net <- glmnet(x = x_matrix,
                               y = y_matrix,
                               weights = w_matrix,
                               family = poisson(link = "log"),
                               alpha = 0.5,
                               lambda = .0001)

selected_coefficients <- coef(selected_elastic_net)

# select_df <- as.data.frame(coef(selected_elastic_net)[1:15])
# write_csv(select_df,
#           "C:/...folder.../full_output.csv")

# Elastic Net Histograms
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = AllTrips),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("AllTrips Coefficient") +
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[2],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



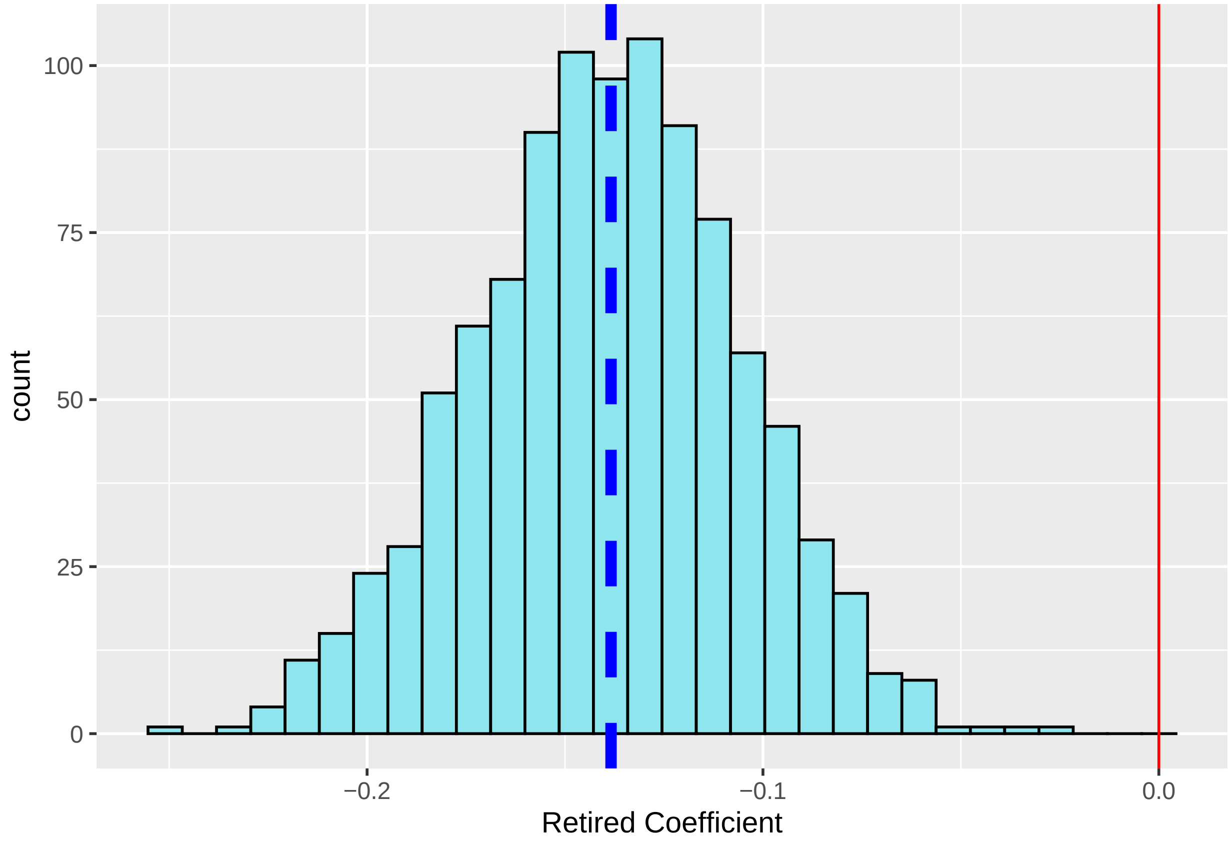
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = Professional),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Professional Coefficient") +
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[3],
            color = "blue", linetype = "dashed", lwd = 2)
```

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



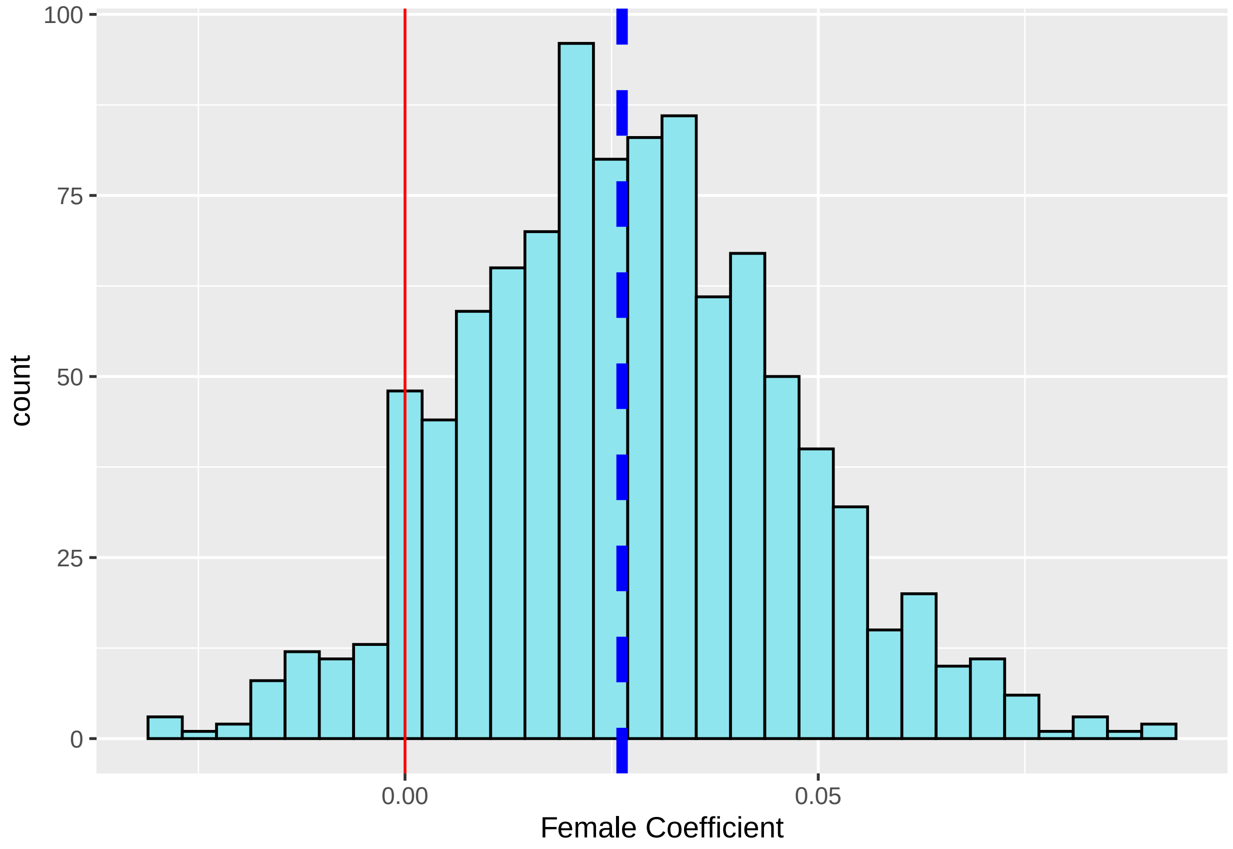
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = Retired),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Retired Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[4],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



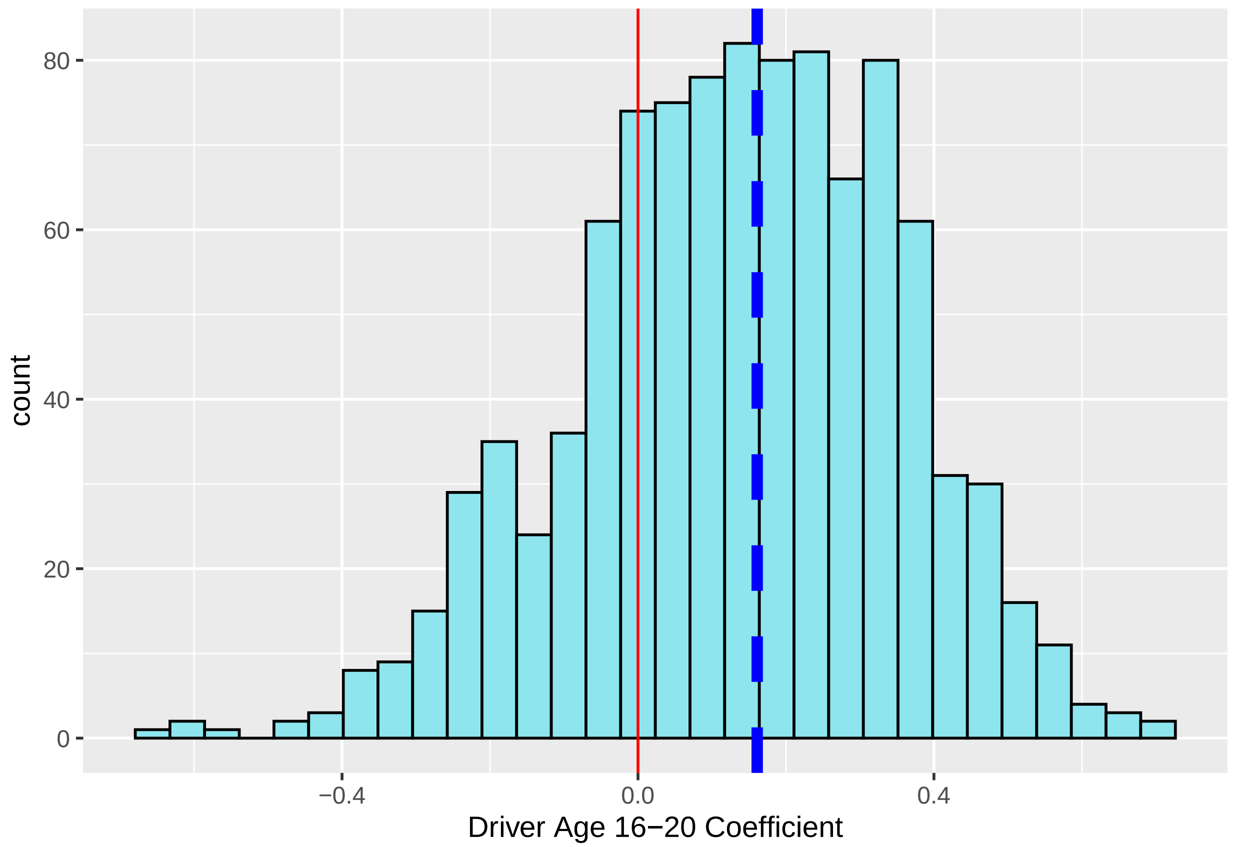
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = Female),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Female Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[5],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



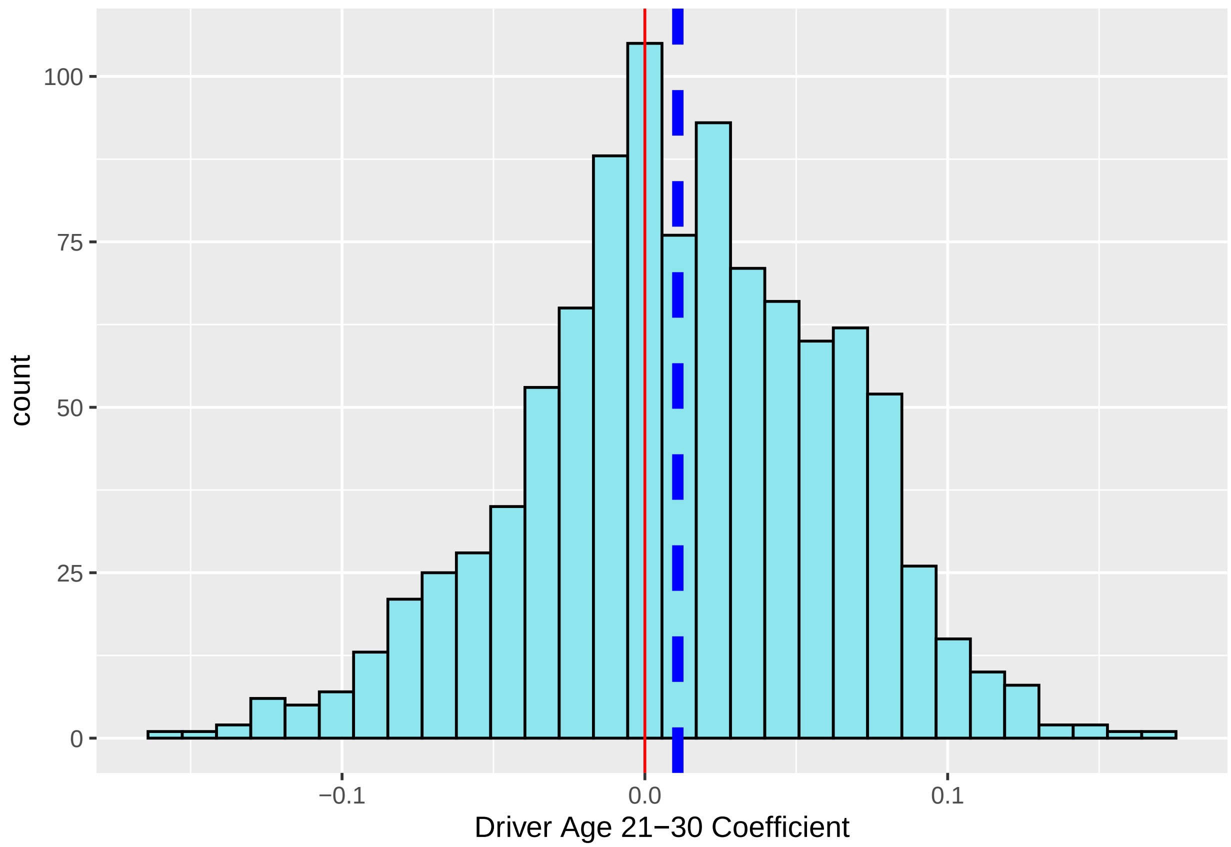
```r
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = drv_age_16_20),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Driver Age 16-20 Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[6],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



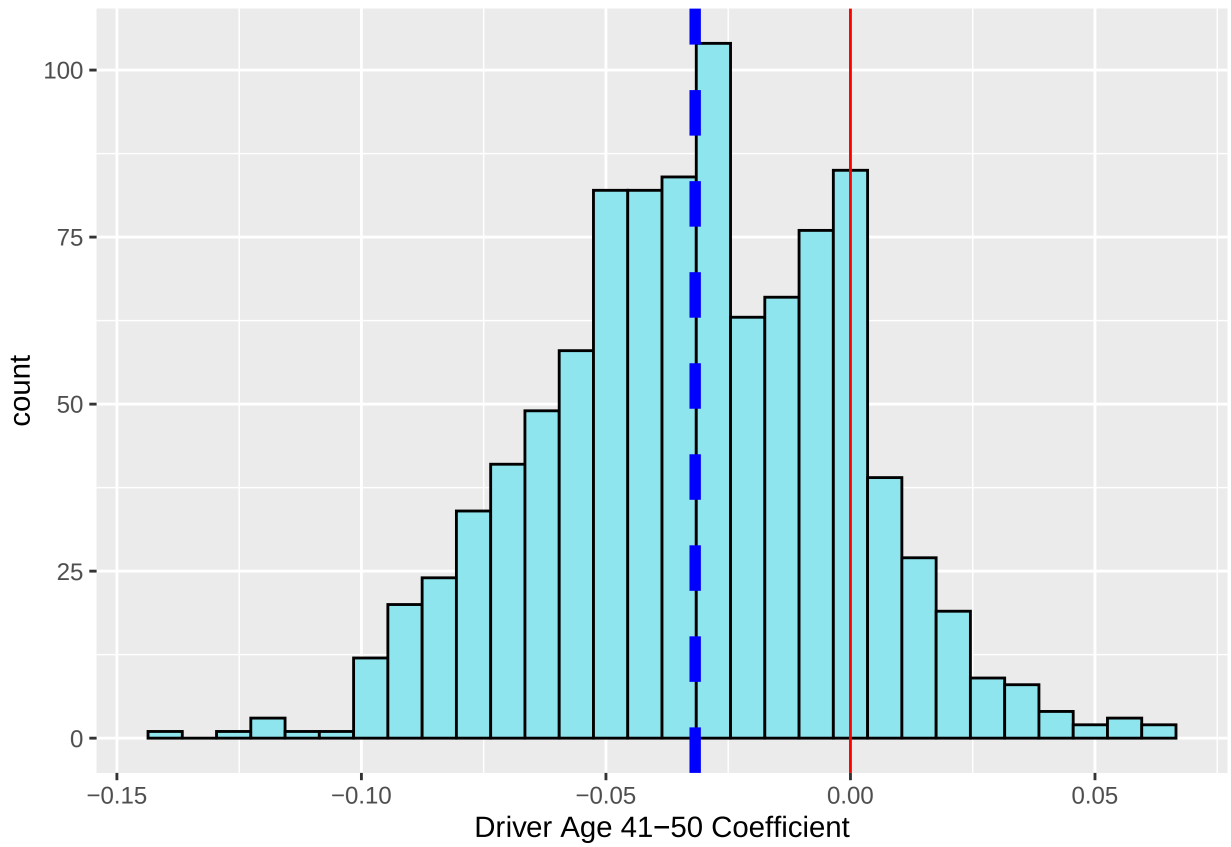
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = drv_age_21_30),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Driver Age 21-30 Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[7],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



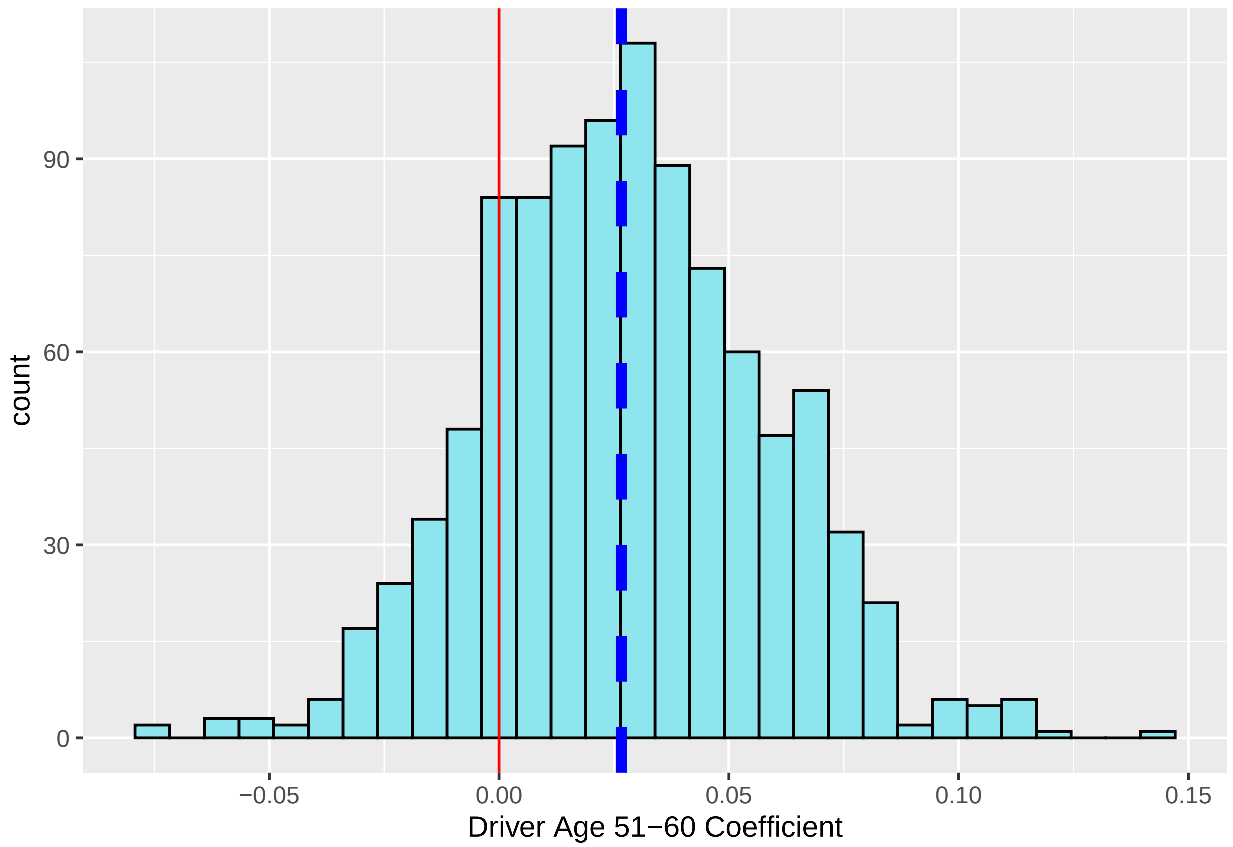
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = drv_age_41_50),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Driver Age 41-50 Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[8],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



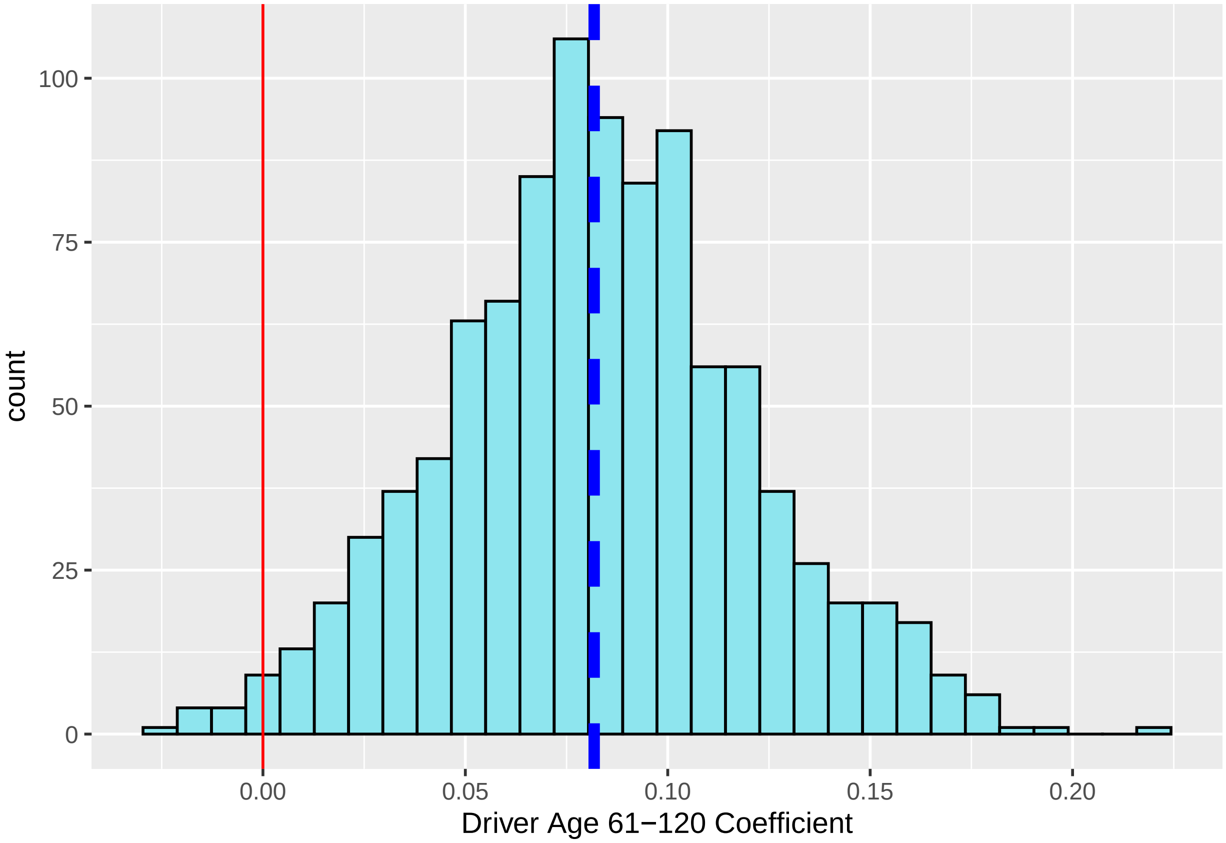
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = drv_age_51_60),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Driver Age 51-60 Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[9],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



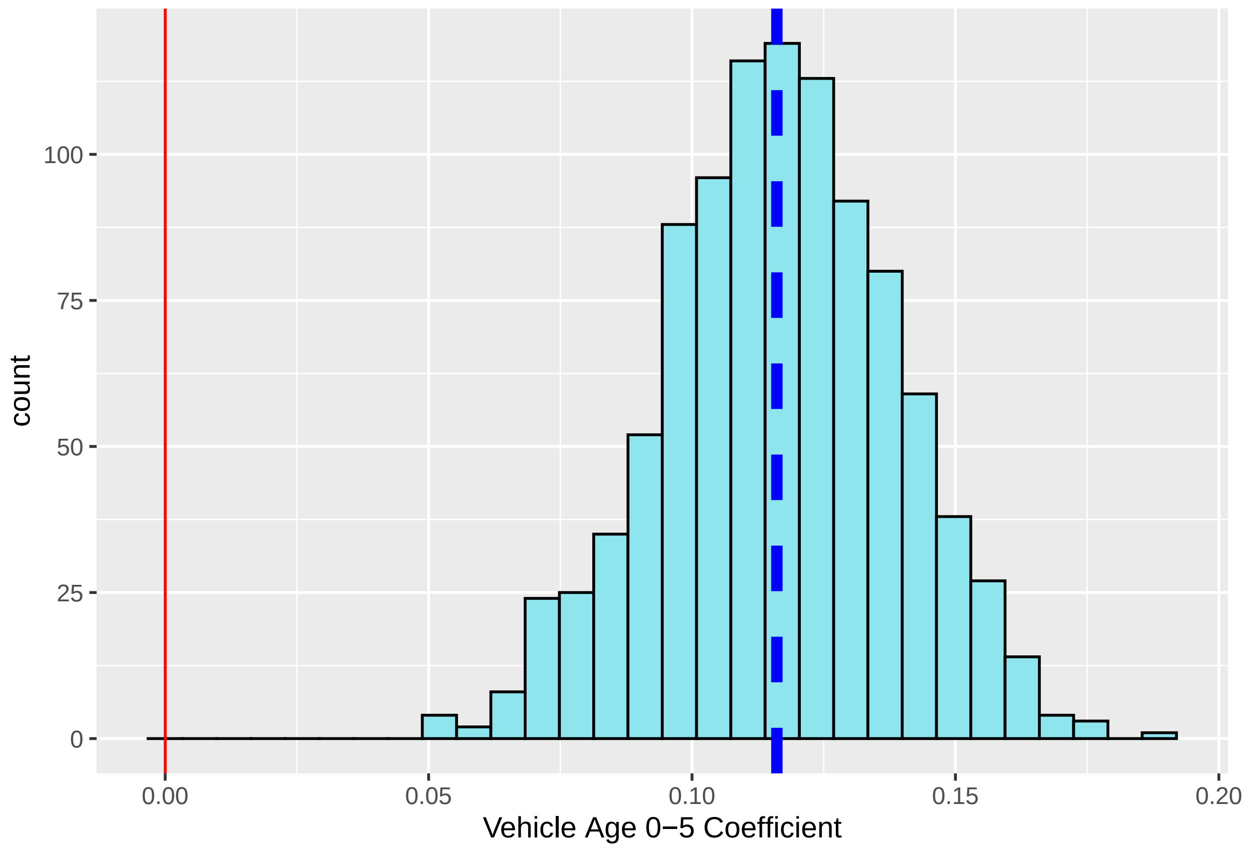
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = drv_age_61_120),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Driver Age 61-120 Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[10],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



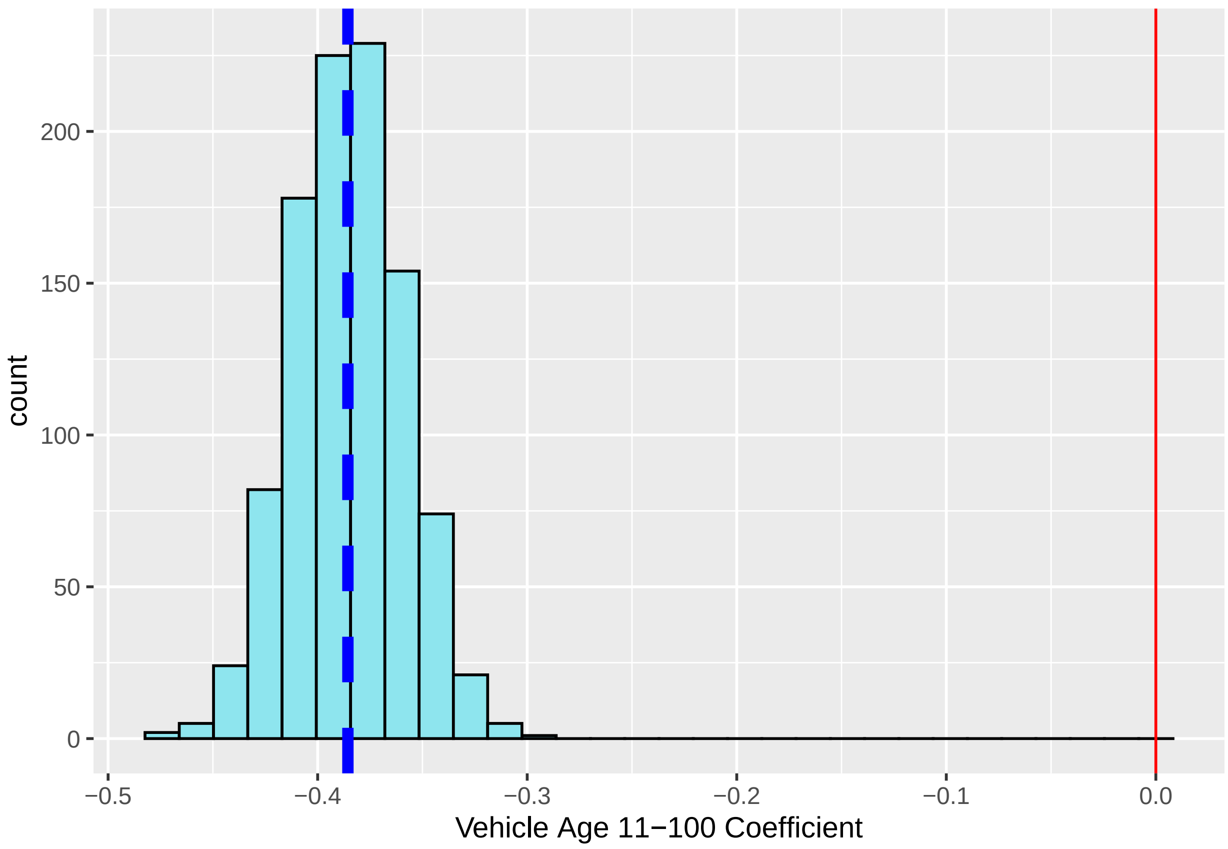
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = vh_age_0_5),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Vehicle Age 0-5 Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[11],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



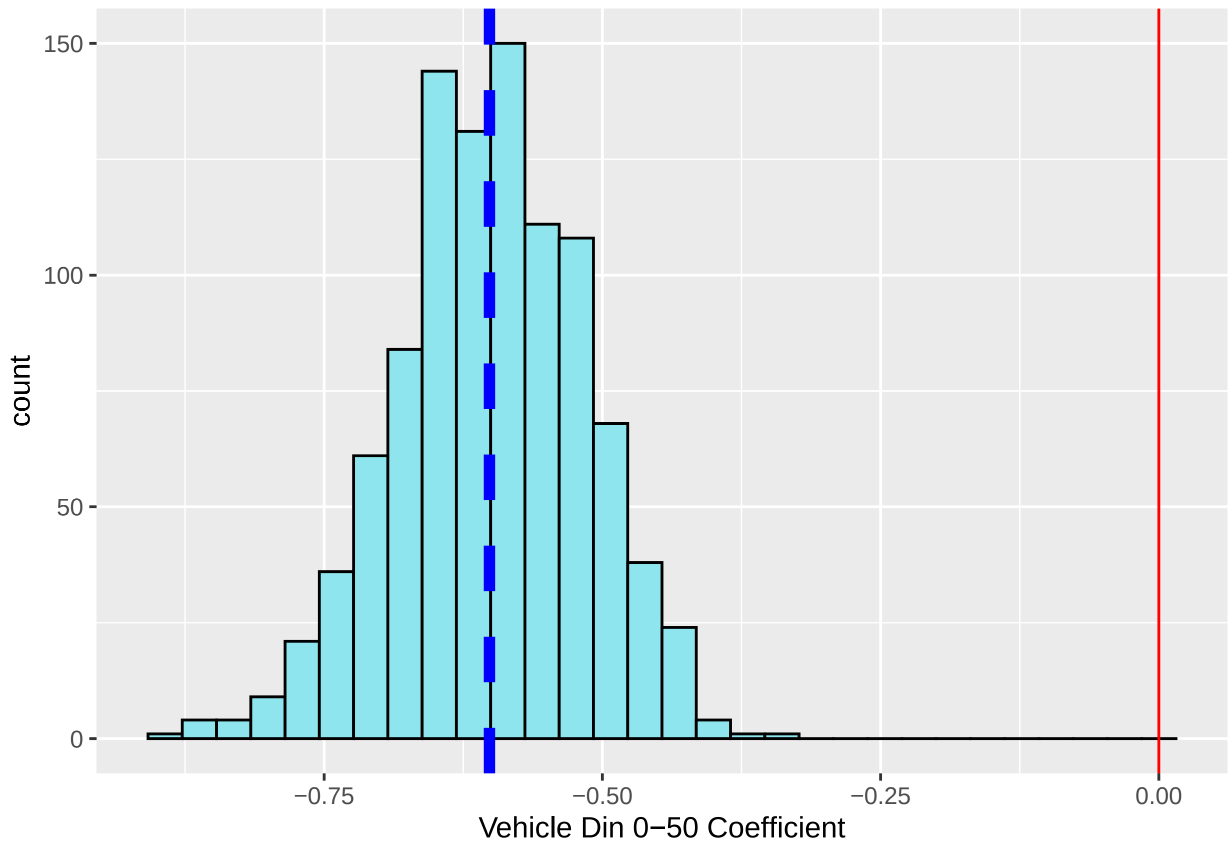
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = vh_age_11_100),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Vehicle Age 11-100 Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[12],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



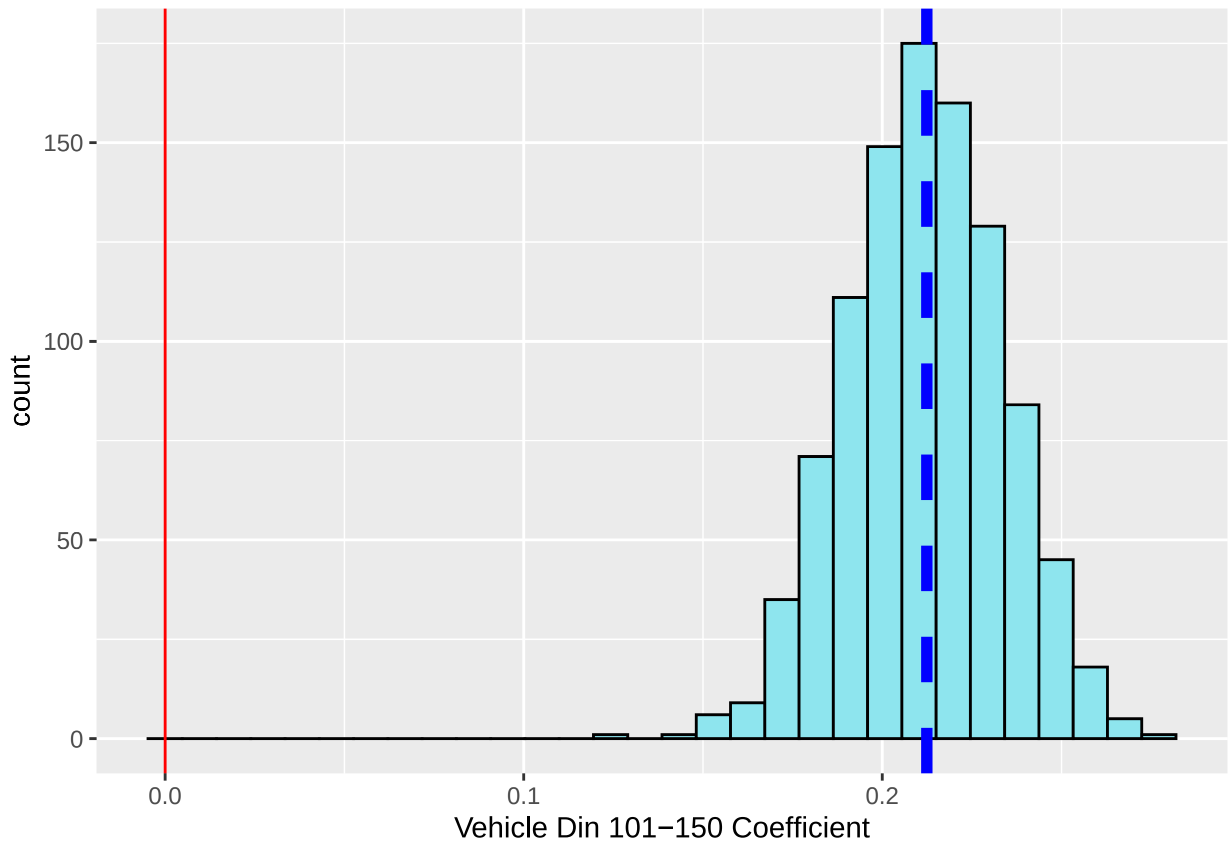
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = vh_din_0_50),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Vehicle Din 0-50 Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[13],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



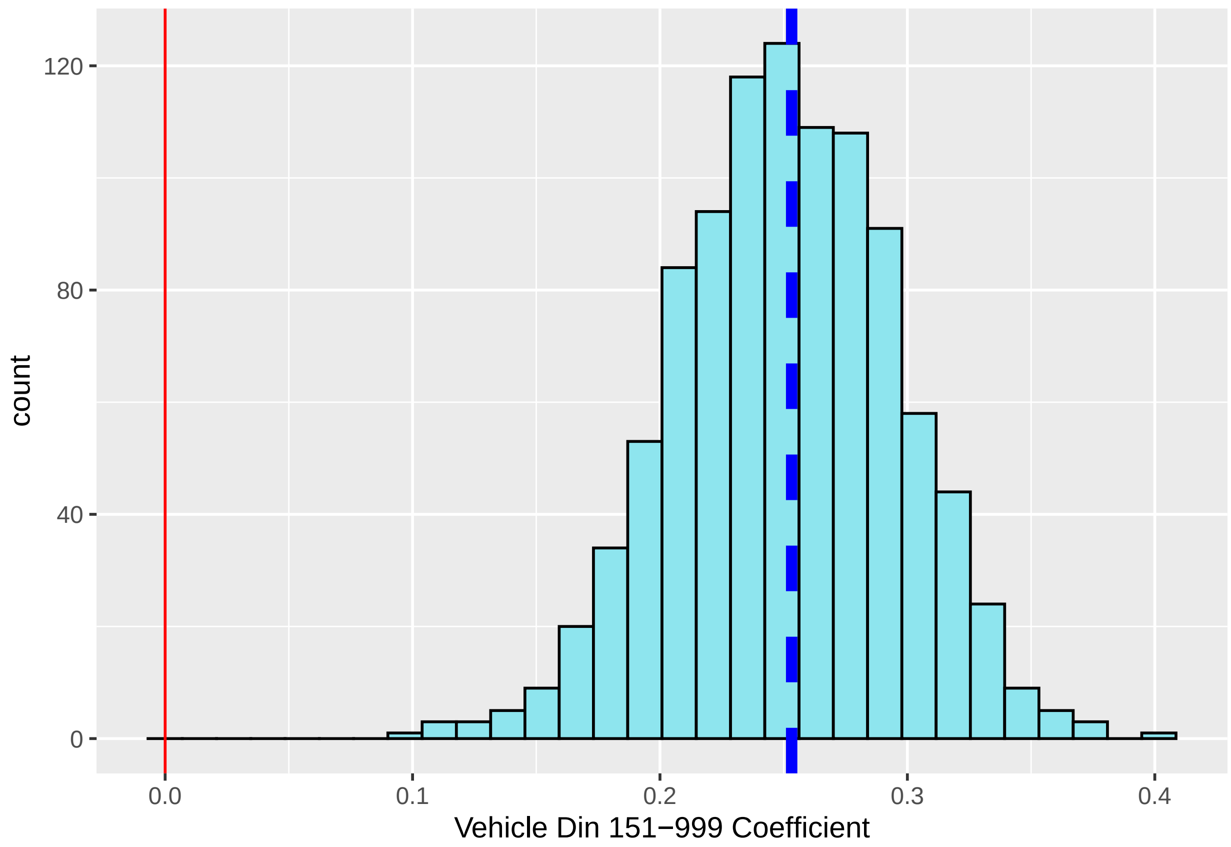
```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = vh_din_101_150),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Vehicle Din 101-150 Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[14],
             color = "blue", linetype = "dashed", lwd = 2)
```

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



```
ggplot(data = coef_table_glmnet) +
  geom_histogram(mapping = aes(x = vh_din_151_999),
                 color = "black",
                 fill = "cadetblue2") +
  xlab("Vehicle Din 151-999 Coefficient")+
  geom_vline(xintercept = 0, color = "red") +
  geom_vline(xintercept = selected_coefficients[15],
             color = "blue", linetype = "dashed", lwd = 2)
```

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


120
80
40
0
count
0.0
0.1
0.2
0.3
0.4
Vehicle Din 151−999 Coefficient

## K Fold Validation

```r
k <- 5

# Create a matrix for the elasticnet training data
training_data <- training_data %>%
  mutate(fold_num = sample(1:5, size = nrow(training_data), replace = TRUE))

training_data %>%
  group_by(fold_num) %>%
  summarize(exposures = sum(exposures))
```

```
## # A tibble: 5 x 2
##   fold_num exposures
##      <int>     <dbl>
## 1        1     15810
## 2        2     15867
## 3        3     15815
## 4        4     16049
## 5        5     15411
```

```r
coef_table_folds <- data.frame(Fold = 1:5,
                         AllTrips = 0,
                         Professional = 0,
                         Retired = 0,
                         Female = 0,
                         drv_age_16_20 = 0,
                         drv_age_21_30 = 0,
                         drv_age_41_50 = 0, # 31-40 will be the base
                         drv_age_51_60 = 0,
                         drv_age_61_120 = 0,
                         vh_age_0_5 = 0,
                         vh_age_11_100 = 0, # 6-10 will be the base
                         vh_din_0_50 = 0,
                         vh_din_101_150 = 0, # 51-100 will be the base
                         vh_din_151_999 = 0)

start_time <- Sys.time()

for(i in 1:k){

  iteration_data <- training_data %>%
    filter(fold_num != i)

  x_matrix <- iteration_data %>%
    select(usage_AllTrips, usage_Professional, usage_Retired, driver_gender_F,
           drv_age_16_20, drv_age_21_30, drv_age_41_50,
           drv_age_51_60, drv_age_61_120,
           vh_age_0_5, vh_age_11_100,
           vh_din_0_50, vh_din_101_150, vh_din_151_999) %>%
    as.matrix()
  y_matrix <- iteration_data %>%
    select(frequency) %>%
    as.matrix()

  w_matrix <- iteration_data %>%
    select(exposures) %>%
    as.matrix()


  # Run the GLM net and insert the coefficients into the table recently created
  model_k <- glmnet(x = x_matrix,
                        y = y_matrix,
                        weights = w_matrix,
                        family = poisson(link = "log"),
                        alpha = 0.5,
                        lambda = .0001)

  coef_table_folds[i,2:15] <- coef(model_k)[2:15]
  # print(paste("fold",i,"complete"))
}

# write_csv(coef_table_folds,
#           "C:/...folder.../kfold_output.csv")

end_time <- Sys.time()

end_time - start_time
```

```
## Time difference of 2.573434 secs
```

```
coef_table_folds
```

```
##   Fold  AllTrips Professional    Retired     Female drv_age_16_20 drv_age_21_30
## 1    1 0.6988120    0.2543863 -0.1537989 0.01339261    0.15123061   0.042201520
## 2    2 0.5194355    0.2217218 -0.1159042 0.01068531    0.13997666  -0.004028511
## 3    3 0.7335252    0.2135560 -0.1556974 0.03318690    0.11576024  -0.011853165
## 4    4 0.5680728    0.2172467 -0.1398239 0.03261569    0.34137246   0.000000000
## 5    5 0.5594436    0.2858275 -0.1267328 0.04089853    0.03940753   0.036626124
##   drv_age_41_50 drv_age_51_60 drv_age_61_120 vh_age_0_5 vh_age_11_100
## 1  -0.008207322    0.02224350     0.08480408  0.1113869    -0.3977955
## 2  -0.029953936    0.01411617     0.06364873  0.1076501    -0.3928762
## 3  -0.052009330    0.02724884     0.08647019  0.1125665    -0.3995431
## 4  -0.034243629    0.03307082     0.08539254  0.1198559    -0.3646544
## 5  -0.031856848    0.03825732     0.09072490  0.1287793    -0.3731509
##   vh_din_0_50 vh_din_101_150 vh_din_151_999
## 1  -0.5929098      0.2117257      0.2202607
## 2  -0.6259424      0.2227097      0.2907386
## 3  -0.5526752      0.2072858      0.2438602
## 4  -0.6013855      0.1966452      0.2470666
## 5  -0.6358202      0.2240810      0.2633727
```

## Reference GLM

```
glimpse(training_data)
```

```
## Rows: 78,952
## Columns: 153
## $ pol_bonus          <dbl> 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.64, 0.5~
## $ pol_duration       <int> 29, 3, 2, 22, 5, 5, 2, 5, 26, 8, 4, 21, 9, 6, 6, 8,~
## $ pol_sit_duration   <int> 9, 1, 2, 1, 1, 3, 2, 1, 6, 1, 4, 1, 1, 2, 3, 3, 4, ~
## $ drv_age1           <dbl> 85, 69, 37, 81, 68, 77, 64, 38, 59, 66, 61, 65, 38,~
## $ drv_age_lic1       <int> 62, 39, 18, 54, 40, 55, 37, 19, 41, 45, 43, 43, 19,~
## $ vh_age             <dbl> 10, 4, 11, 16, 14, 7, 11, 9, 6, 4, 5, 5, 1, 2, 8, 2~
## $ vh_cyl             <int> 1587, 2149, 1991, 1781, 1769, 1870, 1595, 1997, 199~
## $ vh_din             <dbl> 98, 170, 150, 90, 60, 108, 101, 109, 90, 90, 127, 6~
## $ vh_sale_begin      <int> 10, 4, 12, 18, 28, 10, 16, 9, 9, 4, 6, 7, 3, 5, 10,~
## $ vh_sale_end        <int> 9, 2, 11, 15, 18, 6, 13, 7, 7, 3, 3, 4, 1, 4, 8, 23~
## $ vh_speed           <int> 182, 229, 210, 180, 155, 193, 191, 183, 163, 180, 1~
## $ vh_value           <int> 20700, 34250, 28661, 14407, 11564, 22450, 20535, 23~
## $ vh_weight          <int> 1210, 1510, 1270, 1020, 850, 1350, 1195, 1260, 1110~
## $ claim_count        <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, ~
## $ claim_amount       <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 927.16, 0~
## $ exposures          <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ drv_age_bucket     <chr> "drv_age_61_120", "drv_age_61_120", "drv_age_31_40"~
## $ vh_age_bucket      <chr> "vh_age_6_10", "vh_age_0_5", "vh_age_11_100", "vh_a~
## $ vh_din_bucket      <chr> "vh_din_51_100", "vh_din_151_999", "vh_din_101_150"~
## $ coverage_Maxi      <dbl> 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, ~
## $ coverage_Median1   <dbl> 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ coverage_Median2   <dbl> 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, ~
## $ coverage_Mini      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ pay_Biannual       <dbl> 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, ~
## $ pay_Monthly        <dbl> 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, ~
## $ pay_Quarterly      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, ~
## $ pay_Yearly         <dbl> 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, ~
## $ usage_AllTrips     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ usage_Professional <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ usage_Retired      <dbl> 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, ~
## $ usage_WorkPrivate  <dbl> 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, ~
## $ second_driver_Yes  <dbl> 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, ~
## $ driver_gender_F    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, ~
## $ fuel_Diesel        <dbl> 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, ~
## $ fuel_Gasoline      <dbl> 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, ~
## $ fuel_Hybrid        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ type_Commercial    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ type_Tourism       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ drv_age_16_20      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ drv_age_21_30      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ drv_age_31_40      <dbl> 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, ~
## $ drv_age_41_50      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, ~
## $ drv_age_51_60      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, ~
## $ drv_age_61_120     <dbl> 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, ~
## $ vh_age_0_5         <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, ~
## $ vh_age_11_100      <dbl> 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, ~
## $ vh_age_6_10        <dbl> 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, ~
## $ vh_din_0_50        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ vh_din_101_150     <dbl> 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, ~
## $ vh_din_151_999     <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ vh_din_51_100      <dbl> 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, ~
## $ frequency          <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, ~
## $ random001          <int> 5, 4, 3, 1, 5, 2, 1, 5, 1, 4, 3, 5, 4, 2, 1, 2, 3, ~
## $ random002          <int> 5, 3, 4, 3, 5, 2, 4, 4, 3, 3, 3, 2, 4, 3, 4, 2, 5, ~
## $ random003          <int> 4, 2, 1, 3, 5, 2, 1, 4, 3, 5, 4, 5, 4, 4, 3, 3, 1, ~
## $ random004          <int> 2, 1, 2, 4, 3, 5, 3, 3, 4, 5, 1, 2, 1, 3, 1, 2, 1, ~
## $ random005          <int> 2, 2, 5, 1, 1, 4, 5, 5, 3, 5, 1, 3, 3, 3, 4, 1, 4, ~
## $ random006          <int> 4, 1, 3, 5, 3, 2, 1, 5, 1, 2, 5, 1, 3, 3, 4, 4, 5, ~
## $ random007          <int> 3, 5, 1, 2, 4, 3, 3, 3, 4, 4, 5, 5, 2, 5, 1, 1, 2, ~
## $ random008          <int> 2, 3, 4, 5, 4, 1, 4, 2, 2, 4, 5, 3, 3, 5, 2, 5, 4, ~
## $ random009          <int> 4, 2, 3, 5, 4, 2, 3, 4, 1, 2, 2, 5, 1, 3, 2, 1, 2, ~
## $ random010          <int> 3, 3, 1, 4, 2, 2, 1, 3, 4, 1, 2, 5, 5, 3, 3, 2, 5, ~
## $ random011          <int> 1, 3, 4, 3, 3, 5, 1, 5, 2, 2, 5, 2, 2, 4, 3, 1, 4, ~
## $ random012          <int> 2, 5, 1, 3, 4, 3, 2, 4, 5, 2, 5, 5, 3, 3, 4, 1, 1, ~
## $ random013          <int> 3, 4, 1, 4, 5, 5, 3, 1, 2, 2, 2, 1, 1, 4, 2, 1, 2, ~
## $ random014          <int> 5, 2, 2, 5, 1, 5, 4, 4, 3, 2, 4, 1, 1, 2, 1, 1, 1, ~
## $ random015          <int> 1, 2, 1, 3, 3, 1, 2, 2, 2, 1, 1, 1, 5, 1, 1, 2, 5, ~
## $ random016          <int> 3, 5, 5, 3, 5, 1, 4, 2, 3, 3, 5, 3, 3, 5, 5, 4, 5, ~
## $ random017          <int> 4, 3, 2, 1, 5, 2, 2, 5, 2, 2, 4, 5, 3, 5, 4, 5, 5, ~
## $ random018          <int> 4, 1, 4, 2, 4, 5, 5, 1, 2, 1, 5, 2, 2, 4, 4, 1, 5, ~
## $ random019          <int> 4, 1, 4, 3, 3, 2, 5, 4, 4, 3, 4, 2, 4, 4, 1, 3, 1, ~
## $ random020          <int> 5, 4, 2, 1, 2, 5, 3, 3, 5, 3, 3, 5, 2, 3, 4, 1, 2, ~
## $ random021          <int> 1, 2, 4, 3, 3, 1, 2, 5, 3, 4, 1, 3, 5, 3, 5, 2, 5, ~
## $ random022          <int> 4, 1, 5, 3, 4, 1, 3, 3, 3, 1, 4, 5, 4, 2, 3, 2, 1, ~
## $ random023          <int> 5, 3, 4, 4, 2, 1, 5, 5, 2, 4, 3, 2, 1, 1, 5, 4, 2, ~
## $ random024          <int> 4, 1, 4, 4, 2, 1, 3, 3, 3, 5, 5, 5, 4, 5, 2, 2, 1, ~
## $ random025          <int> 1, 2, 5, 2, 4, 3, 5, 4, 4, 4, 5, 3, 4, 1, 1, 2, 4, ~
## $ random026          <int> 5, 4, 3, 1, 1, 5, 2, 5, 2, 1, 3, 5, 3, 2, 3, 5, 3, ~
## $ random027          <int> 3, 4, 3, 1, 4, 5, 2, 5, 3, 5, 1, 3, 2, 3, 4, 5, 1, ~
## $ random028          <int> 3, 4, 1, 1, 3, 5, 5, 5, 5, 5, 5, 3, 2, 5, 1, 4, 1, ~
## $ random029          <int> 3, 5, 2, 1, 5, 1, 2, 5, 1, 4, 4, 4, 4, 4, 3, 1, 1, ~
## $ random030          <int> 1, 1, 1, 2, 4, 2, 4, 4, 2, 5, 3, 3, 3, 1, 2, 5, 1, ~
## $ random031          <int> 3, 1, 1, 3, 2, 5, 1, 4, 1, 4, 4, 1, 2, 5, 5, 3, 4, ~
## $ random032          <int> 4, 2, 4, 1, 5, 5, 1, 1, 4, 2, 1, 4, 2, 4, 4, 5, 3, ~
## $ random033          <int> 2, 4, 1, 2, 3, 5, 5, 1, 4, 5, 5, 2, 3, 4, 5, 4, 1, ~
## $ random034          <int> 5, 1, 4, 4, 1, 2, 2, 1, 1, 1, 5, 2, 1, 5, 4, 1, 3, ~
## $ random035          <int> 3, 2, 5, 3, 5, 1, 4, 3, 4, 3, 4, 1, 3, 4, 3, 5, 4, ~
## $ random036          <int> 2, 5, 2, 5, 2, 2, 4, 3, 2, 3, 4, 2, 4, 4, 2, 2, 3, ~
## $ random037          <int> 3, 2, 3, 1, 2, 3, 5, 1, 3, 4, 2, 3, 1, 1, 4, 4, 1, ~
## $ random038          <int> 1, 3, 5, 5, 4, 3, 5, 1, 5, 4, 4, 5, 1, 5, 5, 3, 5, ~
## $ random039          <int> 4, 1, 1, 1, 3, 3, 1, 4, 4, 5, 1, 3, 5, 1, 1, 5, 3, ~
## $ random040          <int> 2, 3, 3, 2, 2, 5, 4, 4, 3, 5, 5, 3, 2, 2, 5, 5, 4, ~
## $ random041          <int> 5, 5, 1, 1, 2, 1, 1, 2, 1, 2, 3, 2, 3, 1, 3, 3, 4, ~
## $ random042          <int> 2, 5, 4, 2, 2, 3, 5, 4, 5, 2, 2, 5, 1, 2, 3, 5, 2, ~
## $ random043          <int> 1, 5, 5, 5, 4, 2, 4, 2, 2, 5, 1, 4, 4, 1, 2, 3, 1, ~
## $ random044          <int> 1, 1, 4, 5, 4, 3, 1, 1, 1, 4, 1, 1, 3, 2, 2, 5, 3, ~
## $ random045          <int> 4, 1, 3, 2, 5, 1, 4, 1, 2, 4, 5, 1, 3, 5, 4, 3, 3, ~
## $ random046          <int> 3, 5, 5, 2, 4, 4, 5, 2, 4, 1, 5, 3, 3, 1, 1, 2, 2, ~
## $ random047          <int> 3, 5, 3, 5, 4, 5, 3, 1, 2, 4, 5, 1, 3, 2, 5, 4, 3, ~
## $ random048          <int> 4, 5, 4, 2, 5, 2, 2, 1, 2, 4, 2, 5, 3, 1, 4, 1, 2, ~
## $ random049          <int> 3, 3, 1, 1, 3, 3, 2, 1, 4, 1, 2, 2, 5, 4, 4, 2, 3, ~
## $ random050            <int> 1, 1, 2, 3, 3, 5, 3, 4, 3, 3, 4, 5, 1, 4, 5, 3, 3, ~
## $ random051            <int> 4, 4, 5, 3, 2, 1, 2, 5, 4, 1, 5, 4, 4, 1, 5, 5, 3, ~
## $ random052            <int> 4, 4, 5, 1, 3, 5, 1, 1, 1, 3, 4, 4, 1, 4, 4, 1, 2, ~
## $ random053            <int> 2, 5, 5, 2, 5, 2, 2, 1, 2, 4, 3, 5, 5, 3, 3, 5, 2, ~
## $ random054            <int> 4, 5, 1, 2, 4, 4, 3, 1, 3, 5, 2, 3, 2, 3, 1, 5, 4, ~
## $ random055            <int> 1, 5, 1, 5, 3, 4, 5, 5, 3, 3, 3, 1, 1, 4, 4, 5, 3, ~
## $ random056            <int> 1, 5, 4, 4, 5, 2, 2, 5, 5, 1, 3, 1, 4, 5, 5, 5, 1, ~
## $ random057            <int> 2, 2, 2, 2, 3, 2, 1, 5, 5, 3, 3, 2, 2, 2, 5, 4, 4, ~
## $ random058            <int> 3, 5, 1, 2, 2, 3, 5, 2, 2, 2, 1, 3, 2, 2, 3, 4, 4, ~
## $ random059            <int> 4, 2, 2, 5, 5, 3, 5, 5, 4, 2, 2, 3, 2, 3, 4, 4, 1, ~
## $ random060            <int> 3, 3, 5, 4, 2, 1, 1, 3, 3, 1, 4, 3, 5, 3, 4, 1, 5, ~
## $ random061            <int> 3, 2, 2, 5, 3, 4, 4, 3, 4, 5, 4, 1, 5, 5, 4, 1, 3, ~
## $ random062            <int> 2, 4, 5, 3, 5, 5, 4, 2, 3, 4, 4, 4, 2, 5, 1, 5, 4, ~
## $ random063            <int> 5, 1, 2, 1, 1, 3, 3, 3, 2, 5, 5, 5, 3, 4, 3, 3, 2, ~
## $ random064            <int> 4, 5, 2, 2, 4, 4, 5, 1, 5, 4, 5, 3, 5, 5, 3, 4, 5, ~
## $ random065            <int> 3, 5, 5, 5, 2, 2, 2, 1, 3, 2, 5, 4, 4, 2, 1, 4, 1, ~
## $ random066            <int> 2, 3, 1, 1, 1, 1, 5, 4, 4, 1, 1, 5, 5, 5, 3, 2, 5, ~
## $ random067            <int> 3, 2, 5, 5, 3, 1, 5, 4, 5, 3, 3, 4, 4, 2, 5, 2, 4, ~
## $ random068            <int> 1, 4, 5, 4, 5, 4, 4, 5, 3, 4, 1, 4, 2, 1, 1, 1, 1, ~
## $ random069            <int> 5, 1, 4, 3, 1, 1, 1, 1, 3, 1, 4, 5, 3, 3, 5, 2, 4, ~
## $ random070            <int> 5, 1, 2, 2, 5, 1, 3, 1, 4, 5, 4, 4, 3, 3, 3, 4, 2, ~
## $ random071            <int> 4, 5, 1, 2, 2, 1, 5, 3, 1, 2, 3, 1, 3, 4, 4, 1, 5, ~
## $ random072            <int> 1, 5, 2, 1, 1, 4, 1, 2, 3, 4, 3, 5, 5, 3, 1, 3, 3, ~
## $ random073            <int> 3, 1, 4, 2, 3, 2, 3, 3, 5, 2, 4, 2, 4, 2, 1, 1, 3, ~
## $ random074            <int> 5, 1, 1, 4, 3, 1, 1, 1, 1, 2, 1, 3, 3, 2, 1, 5, 2, ~
## $ random075            <int> 3, 1, 2, 1, 5, 5, 3, 2, 3, 3, 5, 1, 3, 5, 5, 3, 3, ~
## $ random076            <int> 3, 4, 3, 3, 4, 1, 2, 5, 2, 5, 5, 1, 5, 2, 1, 1, 3, ~
## $ random077            <int> 3, 5, 5, 4, 5, 3, 3, 4, 2, 1, 2, 4, 3, 5, 4, 2, 4, ~
## $ random078            <int> 5, 4, 3, 2, 4, 4, 4, 1, 5, 5, 3, 3, 3, 5, 4, 5, 4, ~
## $ random079            <int> 5, 3, 2, 5, 1, 2, 3, 2, 4, 1, 1, 5, 3, 2, 5, 3, 5, ~
## $ random080            <int> 2, 2, 3, 3, 3, 5, 5, 5, 4, 2, 5, 4, 1, 5, 1, 5, 2, ~
## $ random081            <int> 5, 1, 2, 2, 1, 4, 5, 5, 4, 5, 5, 4, 3, 3, 2, 3, 5, ~
## $ random082            <int> 2, 4, 4, 5, 1, 5, 4, 4, 2, 1, 5, 5, 3, 1, 1, 3, 1, ~
## $ random083            <int> 5, 5, 2, 4, 3, 1, 5, 2, 3, 1, 3, 3, 1, 5, 1, 2, 2, ~
## $ random084            <int> 4, 4, 2, 3, 2, 2, 4, 2, 3, 2, 5, 4, 1, 2, 1, 2, 1, ~
## $ random085            <int> 2, 3, 3, 3, 2, 5, 4, 1, 5, 3, 3, 1, 2, 3, 5, 3, 3, ~
## $ random086            <int> 1, 4, 4, 1, 3, 5, 1, 4, 1, 5, 4, 3, 2, 4, 3, 4, 1, ~
## $ random087            <int> 4, 5, 4, 1, 1, 5, 3, 2, 1, 5, 5, 1, 4, 4, 5, 4, 3, ~
## $ random088            <int> 1, 3, 2, 3, 2, 4, 4, 3, 5, 3, 4, 1, 2, 4, 1, 2, 1, ~
## $ random089            <int> 5, 4, 2, 1, 2, 1, 1, 4, 3, 2, 2, 4, 1, 5, 1, 1, 1, ~
## $ random090            <int> 4, 1, 4, 3, 5, 4, 1, 1, 4, 2, 4, 5, 1, 4, 1, 4, 5, ~
## $ random091            <int> 4, 3, 3, 3, 3, 5, 1, 2, 2, 2, 4, 1, 2, 1, 3, 5, 2, ~
## $ random092            <int> 5, 3, 2, 3, 4, 5, 2, 1, 2, 4, 3, 2, 4, 3, 3, 4, 2, ~
## $ random093            <int> 2, 3, 4, 1, 4, 1, 5, 3, 1, 1, 5, 5, 4, 4, 2, 5, 2, ~
## $ random094            <int> 2, 5, 2, 2, 1, 2, 1, 5, 3, 2, 3, 4, 1, 4, 3, 4, 4, ~
## $ random095            <int> 1, 1, 3, 1, 2, 5, 5, 2, 5, 4, 5, 1, 5, 4, 2, 2, 1, ~
## $ random096            <int> 3, 4, 1, 5, 5, 1, 3, 2, 5, 2, 4, 3, 3, 2, 4, 4, 1, ~
## $ random097            <int> 4, 2, 1, 2, 3, 4, 2, 2, 1, 5, 2, 1, 3, 1, 3, 2, 2, ~
## $ random098            <int> 2, 1, 4, 3, 1, 4, 5, 3, 4, 5, 2, 2, 3, 4, 4, 2, 3, ~
## $ random099            <int> 1, 4, 2, 1, 3, 3, 3, 5, 5, 4, 3, 3, 4, 3, 2, 4, 5, ~
## $ random100            <int> 1, 4, 2, 1, 3, 2, 3, 3, 4, 4, 5, 3, 4, 2, 5, 3, 1, ~
## $ fold_num             <int> 4, 5, 2, 1, 4, 5, 3, 4, 1, 2, 4, 2, 4, 1, 1, 2, 5, ~
```

```
ref_glm <- glm(formula = frequency ~ usage_AllTrips + usage_Professional +
                 usage_Retired + driver_gender_F +
                 drv_age_16_20 + drv_age_21_30 + drv_age_41_50 +
                 drv_age_51_60 + drv_age_61_120 +
                 vh_age_0_5 + vh_age_11_100 +
                 vh_din_0_50 + vh_din_101_150 + vh_din_151_999,
               weights = exposures,
               family = poisson(link = "log"),
               data = training_data)

coef(ref_glm)
```

```
##        (Intercept)     usage_AllTrips usage_Professional      usage_Retired
##        -1.90116506         0.62214436         0.23922043        -0.14162014
##    driver_gender_F      drv_age_16_20      drv_age_21_30      drv_age_41_50
##         0.02718460         0.17018756         0.01432805        -0.03058553
##      drv_age_51_60     drv_age_61_120         vh_age_0_5      vh_age_11_100
##         0.02915374         0.08654495         0.11645100        -0.38593102
##        vh_din_0_50     vh_din_101_150     vh_din_151_999
##        -0.60578104         0.21337696         0.25483338
```

```
coef(selected_elastic_net)
```

```
## 15 x 1 sparse Matrix of class "dgCMatrix"
##                             s0
## (Intercept)        -1.89843491
## usage_AllTrips      0.61794706
## usage_Professional  0.23874454
## usage_Retired      -0.13837957
## driver_gender_F     0.02626315
## drv_age_16_20       0.16112259
## drv_age_21_30       0.01088626
## drv_age_41_50      -0.03175661
## drv_age_51_60       0.02662595
## drv_age_61_120      0.08182184
## vh_age_0_5          0.11610526
## vh_age_11_100      -0.38557197
## vh_din_0_50        -0.60145085
## vh_din_101_150      0.21242900
## vh_din_151_999      0.25322026
```

```
summary(ref_glm)
```

```
##
## Call:
## glm(formula = frequency ~ usage_AllTrips + usage_Professional +
##     usage_Retired + driver_gender_F + drv_age_16_20 + drv_age_21_30 +
##     drv_age_41_50 + drv_age_51_60 + drv_age_61_120 + vh_age_0_5 +
##     vh_age_11_100 + vh_din_0_50 + vh_din_101_150 + vh_din_151_999,
##     family = poisson(link = "log"), data = training_data, weights = exposures)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -0.9187  -0.5959  -0.5383  -0.4444   5.5470
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -1.90117    0.03064 -62.058  < 2e-16 ***
## usage_AllTrips      0.62214    0.20048   3.103  0.00191 **
## usage_Professional  0.23922    0.03149   7.597 3.02e-14 ***
## usage_Retired      -0.14162    0.03224  -4.393 1.12e-05 ***
## driver_gender_F     0.02718    0.01919   1.417  0.15652
## drv_age_16_20       0.17019    0.20155   0.844  0.39845
## drv_age_21_30       0.01433    0.04873   0.294  0.76875
## drv_age_41_50      -0.03059    0.03075  -0.995  0.31994
## drv_age_51_60       0.02915    0.02987   0.976  0.32906
## drv_age_61_120      0.08654    0.03586   2.413  0.01580 *
## vh_age_0_5          0.11645    0.02140   5.443 5.25e-08 ***
## vh_age_11_100      -0.38593    0.02487 -15.518  < 2e-16 ***
## vh_din_0_50        -0.60578    0.07923  -7.646 2.08e-14 ***
## vh_din_101_150      0.21338    0.01998  10.678  < 2e-16 ***
## vh_din_151_999      0.25483    0.04189   6.083 1.18e-09 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 49797  on 78951  degrees of freedom
## Residual deviance: 48663  on 78937  degrees of freedom
## AIC: 70558
##
## Number of Fisher Scoring iterations: 6
```

## Experiment with Random Variables

We know several columns were just randomly generated numbers. If we blindly rely on p-values, we'll end up keeping a couple columns which are just random numbers.

Step 1: Include regular variables plus all randomly generated variables. Remove randomly generated variables that have higher p-values

Step 2: Build a model that includes 2 of the randomly generated variables

Step 3: Build a model that excludes all randomly generated variables

We'll use these models to build decile plots in the next section

```
training_subset <- training_data[,c(52,16,28:30,33,39:40,42:45,46,48:50,53:151)]

names(training_subset)
```

```
##   [1] "frequency"          "exposures"          "usage_AllTrips"
##   [4] "usage_Professional" "usage_Retired"      "driver_gender_F"
##   [7] "drv_age_16_20"      "drv_age_21_30"      "drv_age_41_50"
##  [10] "drv_age_51_60"      "drv_age_61_120"     "vh_age_0_5"
##  [13] "vh_age_11_100"      "vh_din_0_50"        "vh_din_101_150"
##  [16] "vh_din_151_999"     "random001"          "random002"
##  [19] "random003"          "random004"          "random005"
##  [22] "random006"          "random007"          "random008"
##  [25] "random009"          "random010"          "random011"
##  [28] "random012"          "random013"          "random014"
##  [31] "random015"          "random016"          "random017"
##  [34] "random018"          "random019"          "random020"
##  [37] "random021"          "random022"          "random023"
##  [40] "random024"          "random025"          "random026"
##  [43] "random027"          "random028"          "random029"
##  [46] "random030"          "random031"          "random032"
##  [49] "random033"          "random034"          "random035"
##  [52] "random036"          "random037"          "random038"
##  [55] "random039"          "random040"          "random041"
##  [58] "random042"          "random043"          "random044"
##  [61] "random045"          "random046"          "random047"
##  [64] "random048"          "random049"          "random050"
##  [67] "random051"          "random052"          "random053"
##  [70] "random054"          "random055"          "random056"
##  [73] "random057"          "random058"          "random059"
##  [76] "random060"          "random061"          "random062"
##  [79] "random063"          "random064"          "random065"
##  [82] "random066"          "random067"          "random068"
##  [85] "random069"          "random070"          "random071"
##  [88] "random072"          "random073"          "random074"
##  [91] "random075"          "random076"          "random077"
##  [94] "random078"          "random079"          "random080"
##  [97] "random081"          "random082"          "random083"
## [100] "random084"          "random085"          "random086"
## [103] "random087"          "random088"          "random089"
## [106] "random090"          "random091"          "random092"
## [109] "random093"          "random094"          "random095"
## [112] "random096"          "random097"          "random098"
## [115] "random099"
```

```
new_glm <- glm(formula = frequency ~ .,
               weights = exposures,
               family = poisson(link = "log"),
               data = training_subset)

summary(new_glm)
```

```
##
## Call:
## glm(formula = frequency ~ ., family = poisson(link = "log"),
##     data = training_subset, weights = exposures)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9753  -0.5966  -0.5277  -0.4404   5.6087
##
## Coefficients: (1 not defined because of singularities)
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -1.7469351  0.1959209  -8.917  < 2e-16 ***
## exposures                  NA         NA      NA       NA
## usage_AllTrips      0.6146538  0.2007848   3.061 0.002204 **
## usage_Professional  0.2410254  0.0315180   7.647 2.05e-14 ***
## usage_Retired      -0.1405282  0.0322659  -4.355 1.33e-05 ***
## driver_gender_F     0.0262742  0.0192018   1.368 0.171211
## drv_age_16_20       0.1608104  0.2017139   0.797 0.425323
## drv_age_21_30       0.0131610  0.0487730   0.270 0.787281
## drv_age_41_50      -0.0304787  0.0307725  -0.990 0.321953
## drv_age_51_60       0.0295266  0.0298916   0.988 0.323255
## drv_age_61_120      0.0860113  0.0358939   2.396 0.016563 *
## vh_age_0_5          0.1159345  0.0214116   5.415 6.14e-08 ***
## vh_age_11_100      -0.3865656  0.0248875 -15.533  < 2e-16 ***
## vh_din_0_50        -0.6042958  0.0792552  -7.625 2.45e-14 ***
## vh_din_101_150      0.2130109  0.0199970  10.652  < 2e-16 ***
## vh_din_151_999      0.2554348  0.0419272   6.092 1.11e-09 ***
## random001           0.0040009  0.0064538   0.620 0.535304
## random002          -0.0025963  0.0064685  -0.401 0.688146
## random003           0.0022556  0.0064562   0.349 0.726815
## random004          -0.0016523  0.0064334  -0.257 0.797315
## random005          -0.0112034  0.0064554  -1.736 0.082652 .
## random006          -0.0012823  0.0064433  -0.199 0.842257
## random007           0.0014944  0.0064601   0.231 0.817057
## random008           0.0099426  0.0064503   1.541 0.123214
## random009          -0.0076173  0.0064517  -1.181 0.237737
## random010           0.0013952  0.0064593   0.216 0.828990
## random011          -0.0024442  0.0064503  -0.379 0.704743
## random012           0.0021816  0.0064525   0.338 0.735282
## random013           0.0082493  0.0064676   1.275 0.202137
## random014          -0.0057809  0.0064336  -0.899 0.368893
## random015          -0.0025597  0.0064404  -0.397 0.691040
## random016           0.0168951  0.0064772   2.608 0.009096 **
## random017           0.0039594  0.0064590   0.613 0.539869
## random018           0.0157739  0.0064543   2.444 0.014529 *
## random019           -0.0032828  0.0064501  -0.509 0.610786
## random020            0.0041234  0.0064514   0.639 0.522729
## random021           -0.0074340  0.0064489  -1.153 0.249014
## random022           -0.0087227  0.0064560  -1.351 0.176669
## random023            0.0002206  0.0064550   0.034 0.972736
## random024           -0.0049388  0.0064597  -0.765 0.444538
## random025           -0.0052687  0.0064581  -0.816 0.414599
## random026           -0.0005814  0.0064561  -0.090 0.928247
## random027            0.0078070  0.0064582   1.209 0.226715
## random028            0.0049919  0.0064625   0.772 0.439855
## random029            0.0046825  0.0064662   0.724 0.468970
## random030            0.0012056  0.0064459   0.187 0.851632
## random031           -0.0137907  0.0064529  -2.137 0.032587 *
## random032           -0.0113024  0.0064580  -1.750 0.080092 .
## random033            0.0017492  0.0064486   0.271 0.786193
## random034           -0.0119840  0.0064405  -1.861 0.062782 .
## random035           -0.0073780  0.0064573  -1.143 0.253214
## random036           -0.0030890  0.0064401  -0.480 0.631470
## random037            0.0222486  0.0064408   3.454 0.000552 ***
## random038            0.0025320  0.0064511   0.392 0.694699
## random039           -0.0108100  0.0064528  -1.675 0.093885 .
## random040            0.0012397  0.0064593   0.192 0.847799
## random041            0.0011531  0.0064642   0.178 0.858428
## random042            0.0033034  0.0064564   0.512 0.608897
## random043           -0.0046522  0.0064522  -0.721 0.470895
## random044           -0.0102419  0.0064506  -1.588 0.112346
## random045           -0.0010057  0.0064632  -0.156 0.876343
## random046           -0.0095099  0.0064455  -1.475 0.140096
## random047            0.0013759  0.0064513   0.213 0.831114
## random048           -0.0145162  0.0064378  -2.255 0.024142 *
## random049           -0.0029382  0.0064521  -0.455 0.648828
## random050            0.0086679  0.0064601   1.342 0.179670
## random051           -0.0048128  0.0064531  -0.746 0.455778
## random052           -0.0096383  0.0064558  -1.493 0.135449
## random053            0.0044012  0.0064433   0.683 0.494562
## random054           -0.0073481  0.0064581  -1.138 0.255193
## random055           -0.0065321  0.0064527  -1.012 0.311392
## random056            0.0065383  0.0064540   1.013 0.311038
## random057           -0.0007713  0.0064461  -0.120 0.904755
## random058           -0.0052790  0.0064342  -0.820 0.411953
## random059           -0.0067135  0.0064566  -1.040 0.298443
## random060            0.0034846  0.0064587   0.540 0.589525
## random061           -0.0135017  0.0064454  -2.095 0.036191 *
## random062           -0.0063280  0.0064496  -0.981 0.326525
## random063           -0.0116778  0.0064668  -1.806 0.070948 .
## random064           -0.0016375  0.0064343  -0.254 0.799112
## random065            0.0073265  0.0064477   1.136 0.255831
## random066           -0.0083534  0.0064426  -1.297 0.194773
## random067            0.0073667  0.0064708   1.138 0.254926
## random068            0.0106017  0.0064426   1.646 0.099853 .
## random069            0.0071057  0.0064612   1.100 0.271440
## random070           -0.0042474  0.0064570  -0.658 0.510669
## random071            0.0043264  0.0064552   0.670 0.502717
## random072           -0.0062315  0.0064540  -0.966 0.334282
## random073          -0.0012994  0.0064622  -0.201 0.840638
## random074           0.0095397  0.0064579   1.477 0.139616
## random075          -0.0043510  0.0064450  -0.675 0.499615
## random076           0.0030919  0.0064383   0.480 0.631064
## random077           0.0014652  0.0064334   0.228 0.819847
## random078          -0.0080162  0.0064691  -1.239 0.215288
## random079           0.0063740  0.0064585   0.987 0.323686
## random080          -0.0106181  0.0064597  -1.644 0.100227
## random081          -0.0027162  0.0064606  -0.420 0.674177
## random082           0.0048734  0.0064528   0.755 0.450110
## random083           0.0041460  0.0064564   0.642 0.520778
## random084          -0.0042408  0.0064654  -0.656 0.511874
## random085           0.0072603  0.0064559   1.125 0.260755
## random086           0.0164896  0.0064624   2.552 0.010722 *
## random087           0.0070586  0.0064577   1.093 0.274369
## random088           0.0076713  0.0064609   1.187 0.235092
## random089          -0.0040965  0.0064581  -0.634 0.525868
## random090          -0.0023989  0.0064663  -0.371 0.710645
## random091          -0.0100857  0.0064550  -1.562 0.118180
## random092          -0.0120189  0.0064597  -1.861 0.062801 .
## random093           0.0054792  0.0064429   0.850 0.395092
## random094          -0.0051570  0.0064349  -0.801 0.422889
## random095           0.0020363  0.0064601   0.315 0.752605
## random096          -0.0077531  0.0064417  -1.204 0.228757
## random097           0.0097321  0.0064380   1.512 0.130616
## random098           0.0100863  0.0064576   1.562 0.118303
## random099           0.0014134  0.0064399   0.219 0.826278
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 49797  on 78951  degrees of freedom
## Residual deviance: 48529  on 78838  degrees of freedom
## AIC: 70622
##
## Number of Fisher Scoring iterations: 6
```

```
# Note that multiple random number fields have p-values < .05

# Here we remove some of the variables with high p-values
# Note we only include 2 of the random fields, and again they have low p-values

glm_with_randoms <- glm(formula = frequency ~ usage_AllTrips +
                          usage_Professional + usage_Retired +
                 drv_age_16_20 +
                 vh_age_0_5 + vh_age_11_100 +
                 vh_din_0_50 + vh_din_101_150 + vh_din_151_999 +
                 random016 + random037,
               weights = exposures,
               family = poisson(link = "log"),
               data = training_subset)

summary(glm_with_randoms)
```

```
##
## Call:
## glm(formula = frequency ~ usage_AllTrips + usage_Professional +
##     usage_Retired + drv_age_16_20 + vh_age_0_5 + vh_age_11_100 +
##     vh_din_0_50 + vh_din_101_150 + vh_din_151_999 + random016 +
##     random037, family = poisson(link = "log"), data = training_subset,
##     weights = exposures)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9137  -0.5969  -0.5370  -0.4479   5.6214
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -1.993133   0.033940 -58.726  < 2e-16 ***
## usage_AllTrips      0.621705   0.200433   3.102 0.001923 **
## usage_Professional  0.241780   0.031224   7.743 9.69e-15 ***
## usage_Retired      -0.073010   0.021670  -3.369 0.000754 ***
## drv_age_16_20       0.163298   0.200443   0.815 0.415252
## vh_age_0_5          0.116585   0.021382   5.453 4.97e-08 ***
## vh_age_11_100      -0.385671   0.024784 -15.561  < 2e-16 ***
## vh_din_0_50        -0.602913   0.079222  -7.610 2.73e-14 ***
## vh_din_101_150      0.205820   0.019628  10.486  < 2e-16 ***
## vh_din_151_999      0.245546   0.041459   5.923 3.17e-09 ***
## random016           0.017012   0.006473   2.628 0.008587 **
## random037           0.022564   0.006439   3.504 0.000458 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 49797  on 78951  degrees of freedom
## Residual deviance: 48659  on 78940  degrees of freedom
## AIC: 70547
##
## Number of Fisher Scoring iterations: 6
```

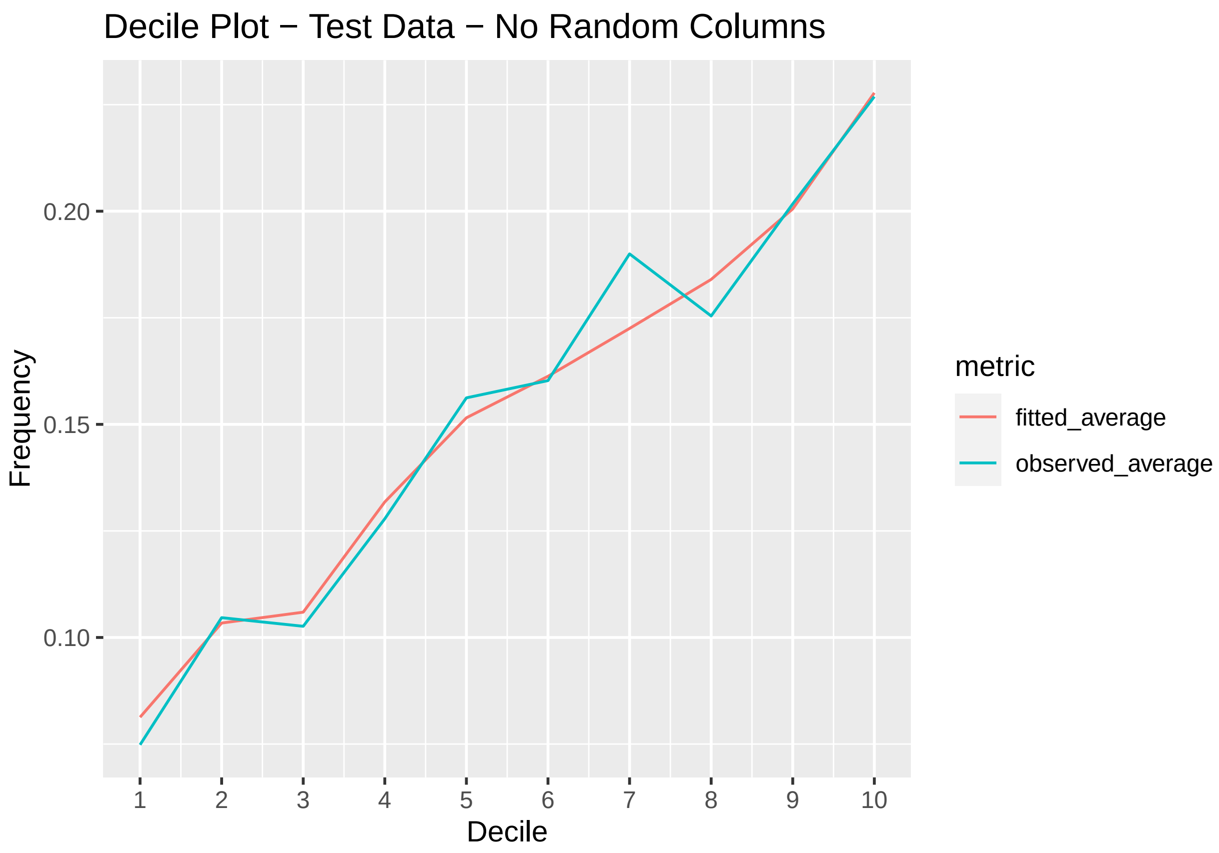
```r
# Compare to a model without randoms

glm_without_randoms <- glm(formula = frequency ~ usage_AllTrips +
                             usage_Professional + usage_Retired +
                 drv_age_16_20 +
                 vh_age_0_5 + vh_age_11_100 +
                 vh_din_0_50 + vh_din_101_150 + vh_din_151_999,
               weights = exposures,
               family = poisson(link = "log"),
               data = training_subset)

summary(glm_without_randoms)
```

```
##
## Call:
## glm(formula = frequency ~ usage_AllTrips + usage_Professional +
##     usage_Retired + drv_age_16_20 + vh_age_0_5 + vh_age_11_100 +
##     vh_din_0_50 + vh_din_101_150 + vh_din_151_999, family = poisson(link = "log"),
##     data = training_subset, weights = exposures)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9100  -0.5874  -0.5345  -0.4570   5.5619
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -1.87388    0.01975 -94.863  < 2e-16 ***
## usage_AllTrips      0.62951    0.20043   3.141 0.001685 **
## usage_Professional  0.24145    0.03122   7.733 1.05e-14 ***
## usage_Retired      -0.07223    0.02167  -3.333 0.000858 ***
## drv_age_16_20       0.16091    0.20044   0.803 0.422106
## vh_age_0_5          0.11653    0.02138   5.450 5.03e-08 ***
## vh_age_11_100      -0.38561    0.02478 -15.559  < 2e-16 ***
## vh_din_0_50        -0.60327    0.07922  -7.615 2.64e-14 ***
## vh_din_101_150      0.20559    0.01963  10.475  < 2e-16 ***
## vh_din_151_999      0.24607    0.04146   5.935 2.93e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 49797  on 78951  degrees of freedom
## Residual deviance: 48678  on 78942  degrees of freedom
## AIC: 70563
##
## Number of Fisher Scoring iterations: 6
```

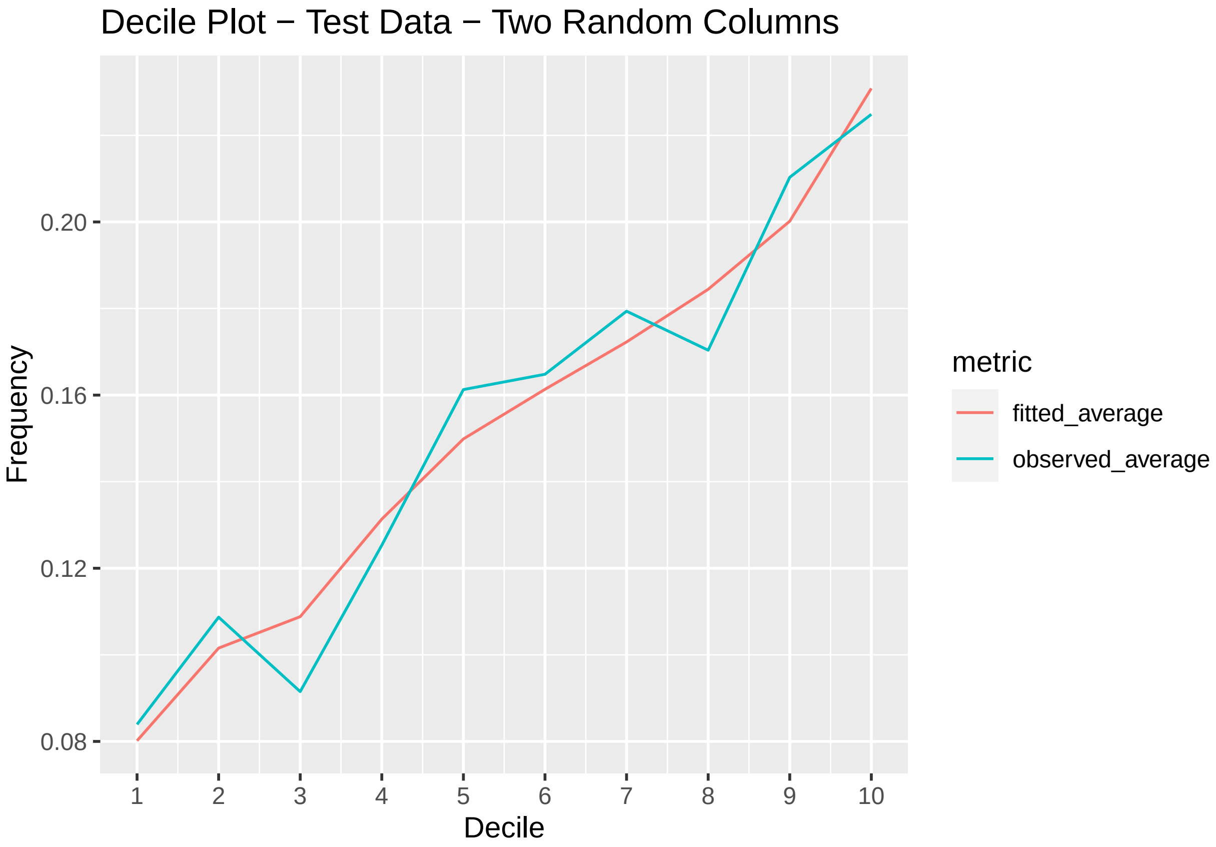
## Decile Plots

The point of this exercise is that the inclusion of a couple totally random variables will not ruin a lift chart. Meaning, looking at an overall lift chart may not be sufficient to determine if all variables used are significant. This is true even if the lift chart is provided on test data.

```r
# Decile Plot Without Random Columns

predictions <- predict(glm_without_randoms,
                       newdata = testing_data,
                       type = "response")

total_expos <- sum(testing_data$exposures)
decile_table <- testing_data %>%
  mutate(predictions = predictions) %>%
  arrange(predictions) %>%
  mutate(decile = if_else(cumsum(exposures)==total_expos,
                         10,
                         floor(10*cumsum(exposures)/total_expos)+1)) %>%
  group_by(decile) %>%
  summarize(fitted_average = as.double(format(sum(predictions)/sum(exposures),
                                              scientific = F)),
            observed_average = sum(claim_count)/sum(exposures))
decile_plot_data <- pivot_longer(decile_table,
                                 cols = c("fitted_average", "observed_average"),
                                 names_to = "metric")
ggplot(decile_plot_data, aes(x = decile)) +
  geom_line(aes(y = value, color = metric)) +
  scale_x_continuous(limits = c(1,10), breaks = seq(1,10,1)) +
  labs(x = "Decile", y= "Frequency") +
  ggtitle("Decile Plot - Test Data - No Random Columns")
```



```
# Decile Plot Two Random Columns

predictions2 <- predict(glm_with_randoms,
                        newdata = testing_data,
                        type = "response")

decile_table2 <- testing_data %>%
  mutate(predictions2 = predictions2) %>%
  arrange(predictions2) %>%
  mutate(decile = if_else(cumsum(exposures)==total_expos,
                          10,
                          floor(10*cumsum(exposures)/total_expos)+1)) %>%
  group_by(decile) %>%
  summarize(fitted_average = as.double(format(sum(predictions2)/sum(exposures),
                                              scientific = F)),
            observed_average = sum(claim_count)/sum(exposures))
decile_plot_data2 <- pivot_longer(decile_table2,
                                  cols = c("fitted_average", "observed_average"),
                                  names_to = "metric")
ggplot(decile_plot_data2, aes(x = decile)) +
  geom_line(aes(y = value, color = metric)) +
  scale_x_continuous(limits = c(1,10), breaks = seq(1,10,1)) +
  labs(x = "Decile", y= "Frequency") +
  ggtitle("Decile Plot - Test Data - Two Random Columns")
```


Decile Plot − Test Data − Two Random Columns
Frequency
0.20
0.16
0.12
0.08
1
2
3
4
5
6
7
8
9
10
Decile
metric
fitted_average
observed_average