

Data Exploration

Sar, North, Henry, Quinn

2/17/2022

```
library(ISLR)
library(dplyr)
```

```
## Warning: replacing previous import 'lifecycle::last_warnings' by
## 'rlang::last_warnings' when loading 'pillar'
```

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

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

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

```
library(readr)
```

```
## Warning: replacing previous import 'lifecycle::last_warnings' by
## 'rlang::last_warnings' when loading 'hms'
```

```
library(broom)
```

```
## Warning: package 'broom' was built under R version 4.1.2
```

```
library(ggplot2)
library(splines)
library(tidymodels)
```

```
## Registered S3 method overwritten by 'tune':
##   method                from
##   required_pkgs.model_spec parsnip
```

```
## -- Attaching packages ----- tidymodels 0.1.4 --
```

```
## v dials      0.0.10    v tibble      3.1.6
## v infer      1.0.0     v tidyr       1.1.4
## v modeldata  0.1.1     v tune       0.1.6
## v parsnip    0.1.7     v workflows  0.2.4
## v purrr      0.3.4     v workflowsets 0.1.0
## v recipes    0.1.17    v yardstick  0.0.9
## v rsample    0.1.1

## -- Conflicts ----- tidymodels_conflicts() --
## x purrr::discard() masks scales::discard()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()  masks stats::step()
## * Use tidymodels_prefer() to resolve common conflicts.
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##      combine
```

```
tidymodels_prefer()
```

```
COVID_State <- read.csv("COVID - State - Daily.csv", na.strings = ".")
Employment_State <- read.csv("Employment - State - Daily.csv", na.strings = ".")
Mobility_State <- read.csv("Google Mobility - State - Daily.csv", na.strings = ".")
Spending_State <- read.csv("Affinity - State - Daily.csv", na.strings = ".")
```

```
COVID_State$Date<-as.Date(with(COVID_State,paste(year,month,day,sep="-")), "%Y-%m-%d")
Employment_State$Date<-as.Date(with(Employment_State,paste(year,month,day,sep="-")), "%Y-%m-%d")
Mobility_State$Date<-as.Date(with(Mobility_State,paste(year,month,day,sep="-")), "%Y-%m-%d")
Spending_State$Date<-as.Date(with(Spending_State,paste(year,month,day,sep="-")), "%Y-%m-%d")
full_data <- merge(merge(merge(COVID_State, Employment_State, by=c("Date", "statefips")), Mobility_State,
by=c("Date", "statefips")), Spending_State, by=c("Date", "statefips"))

## Warning in merge.data.frame(merge(merge(COVID_State, Employment_State, by =
## c("Date", : column names 'year.x', 'month.x', 'day.x', 'year.y', 'month.y',
## 'day.y' are duplicated in the result
```

```
head(full_data)
```

```
##           Date statefips year.x month.x day.x new_case_count new_death_count
## 1 2020-02-24         1   2020         2   24              NA              NA
## 2 2020-02-24        10   2020         2   24              NA              NA
## 3 2020-02-24        11   2020         2   24              NA              NA
## 4 2020-02-24        12   2020         2   24              NA              NA
## 5 2020-02-24        13   2020         2   24              NA              NA
## 6 2020-02-24        15   2020         2   24              NA              NA
##   case_count death_count vaccine_count fullvaccine_count booster_first_count
## 1         NA         NA          NA          NA              NA
## 2         NA         NA          NA          NA              NA
## 3         NA         NA          NA          NA              NA
## 4         NA         NA          NA          NA              NA
## 5         NA         NA          NA          NA              NA
## 6         NA         NA          NA          NA              NA
##   new_vaccine_count new_fullvaccine_count new_booster_first_count
## 1                NA                NA                NA
## 2                NA                NA                NA
## 3                NA                NA                NA
## 4                NA                NA                NA
## 5                NA                NA                NA
## 6                NA                NA                NA
##   new_test_count test_count hospitalized_count new_case_rate case_rate
## 1              NA         NA                NA              NA      NA
## 2              NA         NA                NA              NA      NA
## 3              NA         NA                NA              NA      NA
## 4              NA         NA                NA              NA      NA
## 5              NA         NA                NA              NA      NA
## 6              NA         NA                0              NA      NA
##   new_death_rate death_rate new_test_rate test_rate new_vaccine_rate
## 1              NA         NA              NA         NA              NA
## 2              NA         NA              NA         NA              NA
## 3              NA         NA              NA         NA              NA
## 4              NA         NA              NA         NA              NA
## 5              NA         NA              NA         NA              NA
## 6              NA         NA              NA         NA              NA
##   vaccine_rate new_fullvaccine_rate fullvaccine_rate new_booster_first_rate
## 1              NA                NA              NA              NA
## 2              NA                NA              NA              NA
## 3              NA                NA              NA              NA
## 4              NA                NA              NA              NA
## 5              NA                NA              NA              NA
## 6              NA                NA              NA              NA
##   booster_first_rate hospitalized_rate year.y month.y day.y      emp emp_incq1
## 1              NA                NA   2020         2   24  0.01580  0.00751
## 2              NA                NA   2020         2   24  0.00537 -0.02670
## 3              NA                NA   2020         2   24        NA        NA
## 4              NA                NA   2020         2   24  0.00448 -0.00263
## 5              NA                NA   2020         2   24  0.00532 -0.00537
## 6              NA                0   2020         2   24 -0.03530 -0.07190
##   emp_incq2 emp_incq3 emp_incq4 emp_incmiddle emp_incbelowmed emp_incabovemed
## 1  0.02320  0.01680        NA    0.01960      0.013600      0.0183
```

```

## 2  0.00570  0.01680  0.0242  0.01170  -0.011400  0.0206
## 3      NA      NA      NA      NA      NA      NA
## 4 -0.00458  0.01070  0.0164  0.00324  -0.003550  0.0133
## 5  0.00520  0.00873  0.0140  0.00710  -0.000838  0.0114
## 6 -0.04920 -0.00520      NA  -0.02980  -0.058300  -0.0112
##  emp_ss40 emp_ss60 emp_ss65 emp_ss70 year.x month.x day.x
## 1  0.001540 -0.00399  0.05300 -0.01620  2020      2    24
## 2  0.015400  0.01340  0.01030 -0.05550  2020      2    24
## 3      NA      NA      NA      NA  2020      2    24
## 4 -0.002320  0.00134  0.00576  0.01620  2020      2    24
## 5 -0.000237  0.00168  0.00889  0.00964  2020      2    24
## 6  0.054800      NA      NA -0.01530  2020      2    24
##  gps_retail_and_recreation gps_grocery_and_pharmacy gps_parks
## 1      0.00286      -0.00714  0.0557
## 2      0.03710      0.01290  0.2340
## 3     -0.01140     -0.03290  0.1400
## 4      0.02710      0.00714  0.0943
## 5     -0.00571     -0.02290  0.0186
## 6      0.01140     -0.00571  0.0814
##  gps_transit_stations gps_workplaces gps_residential gps_away_from_home year.y
## 1      0.06000      0.01290  0.00857  -0.00798  2020
## 2      0.07000      0.02860  -0.00571  0.00850  2020
## 3      0.00571     -0.01430  0.00714  -0.00492  2020
## 4      0.03430      0.01000  0.00143  -0.00138  2020
## 5      0.01710     -0.01140  0.01000  -0.00781  2020
## 6      0.02570      0.00714  0.00143  -0.00049  2020
##  month.y day.y freq spend_all spend_aap spend_acf spend_aer spend_apg
## 1      2    24    d  -0.0198  -0.1320  -0.0220  -0.1000  -0.0810
## 2      2    24    d  -0.0461   0.1130  -0.0279  -0.6280   0.4140
## 3      2    24    d   0.0192  -0.1280  -0.0113   0.0740  -0.0855
## 4      2    24    d  -0.0452  -0.0847  -0.0493  -0.1020  -0.0675
## 5      2    24    d  -0.0163  -0.0321  -0.0334   0.0287  -0.0308
## 6      2    24    d  -0.0504  -0.1210  -0.0447  -0.1650  -0.0851
##  spend_durables spend_nondurables spend_grf spend_gen spend_hic spend_hcs
## 1     -0.0317     -0.04750  -0.0223  -0.01050  -0.06180  -0.07310
## 2      0.0208      0.13400  -0.0284   0.63600   0.13400  -0.01060
## 3      0.0311     -0.00364   0.0294   0.00856   0.59500   0.02630
## 4     -0.0492     -0.04720  -0.0468  -0.03810  -0.08320   0.00175
## 5     -0.0164     -0.02450  -0.0110  -0.03000  -0.00361  -0.02010
## 6     -0.0118     -0.04380  -0.0173  -0.04770   0.16600  -0.08730
##  spend_inpersonmisc spend_remoteservices spend_sgh spend_tws
## 1      0.0062      0.02110  -0.0453  -0.1020
## 2     -0.1380     -0.15500  -0.1540  -0.0929
## 3      0.2100     -0.03610  -0.1230  -0.1360
## 4     -0.0815     -0.04600  -0.0426  -0.1030
## 5     -0.0658     -0.00774   0.0940  -0.1060
## 6     -0.0645     -0.04000  -0.2270  -0.0909
##  spend_retail_w_grocery spend_retail_no_grocery spend_all_incmiddle
## 1     -0.03910     -0.0459  -0.02970
## 2      0.10200      0.1560  -0.06480
## 3     -0.00169     -0.0124  -0.06430
## 4     -0.04390     -0.0421  -0.03880
## 5     -0.01640     -0.0176  -0.01870
## 6     -0.03610     -0.0498   0.00268

```

```
##   spend_all_q1 spend_all_q2 spend_all_q3 spend_all_q4 provisional
## 1    -0.0158    -0.0717     0.036100     0.009840           0
## 2     0.2240    -0.0565    -0.068700    -0.016000           0
## 3    -0.0265    -0.5850    -0.047300     0.039400           0
## 4    -0.0677    -0.0420    -0.035100    -0.035700           0
## 5    -0.0386    -0.0234    -0.015600    -0.000937           0
## 6         NA     0.0134     0.000257    -0.076700           0
```

```
full_data1 <- full_data %>%
  select(-year.x, -month.x, -day.x, - year.y, -month.y, -day.y, -year.x )
```

```
minnesota <- full_data1 %>%
  filter(statefips==27)
```

```
minnesota_cut <- minnesota %>%
  filter(Date > "2020-04-13")%>%
  select(Date, gps_away_from_home, case_rate, hospitalized_rate, spend_remoteservices, spend_hcs, emp_i
```

```
full_cut <- full_data1 %>%
  filter(Date > "2020-04-13")%>%
  select(statefips, Date, gps_away_from_home, case_rate, hospitalized_rate, spend_remoteservices, spend
```

```
#Visualizing non-employment variables by date
```

```
gps <-ggplot(minnesota_cut, aes(y = gps_away_from_home, x = Date)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
```

```
#vaccine <-ggplot(minnesota_cut, aes(y = fullvaccine_rate, x = Date)) +
#   geom_point() +
#   geom_smooth(color = "blue", se = FALSE) +
#   geom_hline(yintercept = 0, color = "red") +
#   theme_classic()
```

```
case<-ggplot(minnesota_cut, aes(y = case_rate, x = Date)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
```

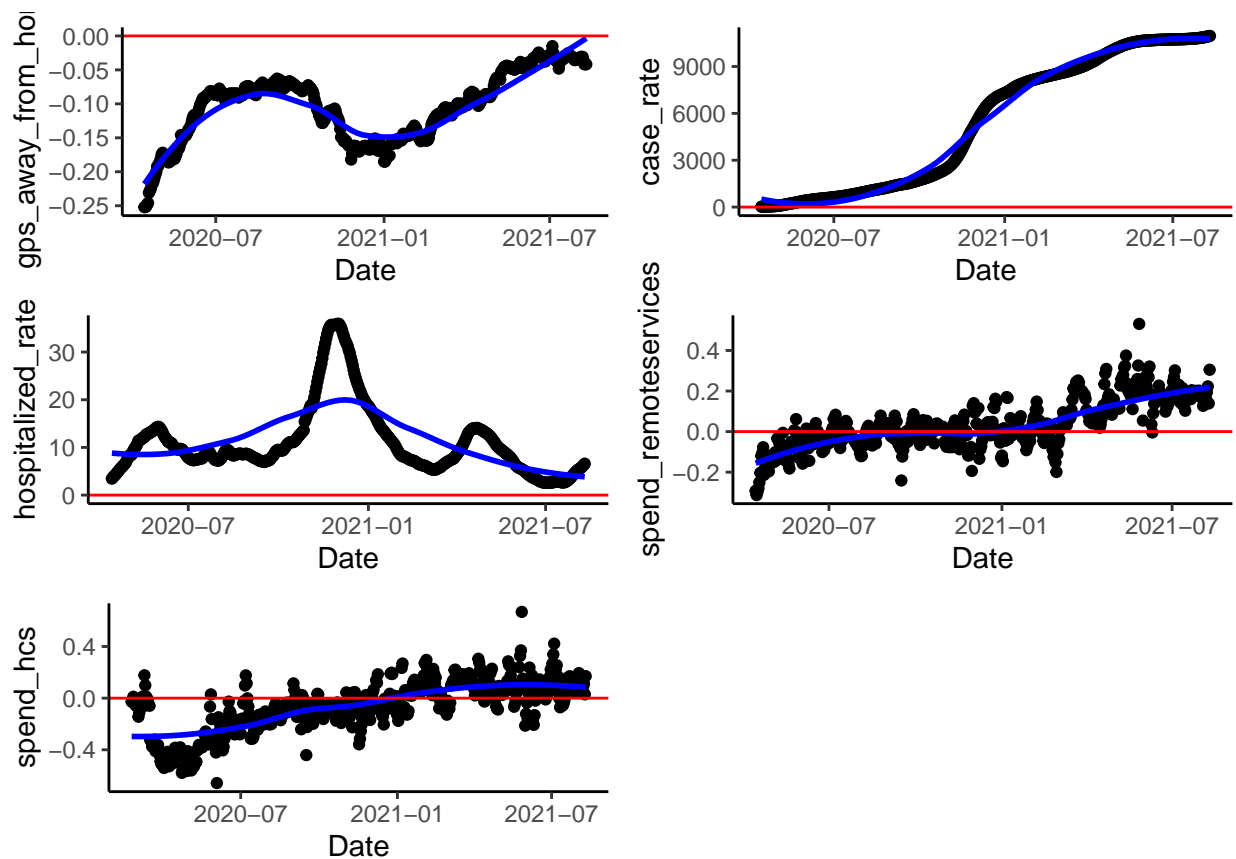
```
hosp<-ggplot(minnesota_cut, aes(y = hospitalized_rate, x = Date)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
```

```
remote<-ggplot(minnesota_cut, aes(y = spend_remoteservices, x = Date)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
```

```
hcs<-ggplot(minnesota, aes(y = spend_hcs, x = Date)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
```

```
grid.arrange(gps, case, hosp, remote, hcs, ncol=2)
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



#Visualizing employment variables by date

```
emp1 <- ggplot(minnesota_cut, aes(y = emp_incq1, x = Date)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()+
  ylim(-0.45, 0.2)
```

```
emp2 <-ggplot(minnesota_cut, aes(y = emp_incq2, x = Date)) +
  geom_point() +
```

```

geom_smooth(color = "blue", se = FALSE) +
geom_hline(yintercept = 0, color = "red") +
theme_classic()+
ylim(-0.45, 0.2)

emp3 <-ggplot(minnesota_cut, aes(y = emp_incq3, x = Date)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()+
  ylim(-0.45, 0.2)

emp4 <-ggplot(minnesota_cut, aes(y = emp_incq4, x = Date)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()+
  ylim(-0.45, 0.2)

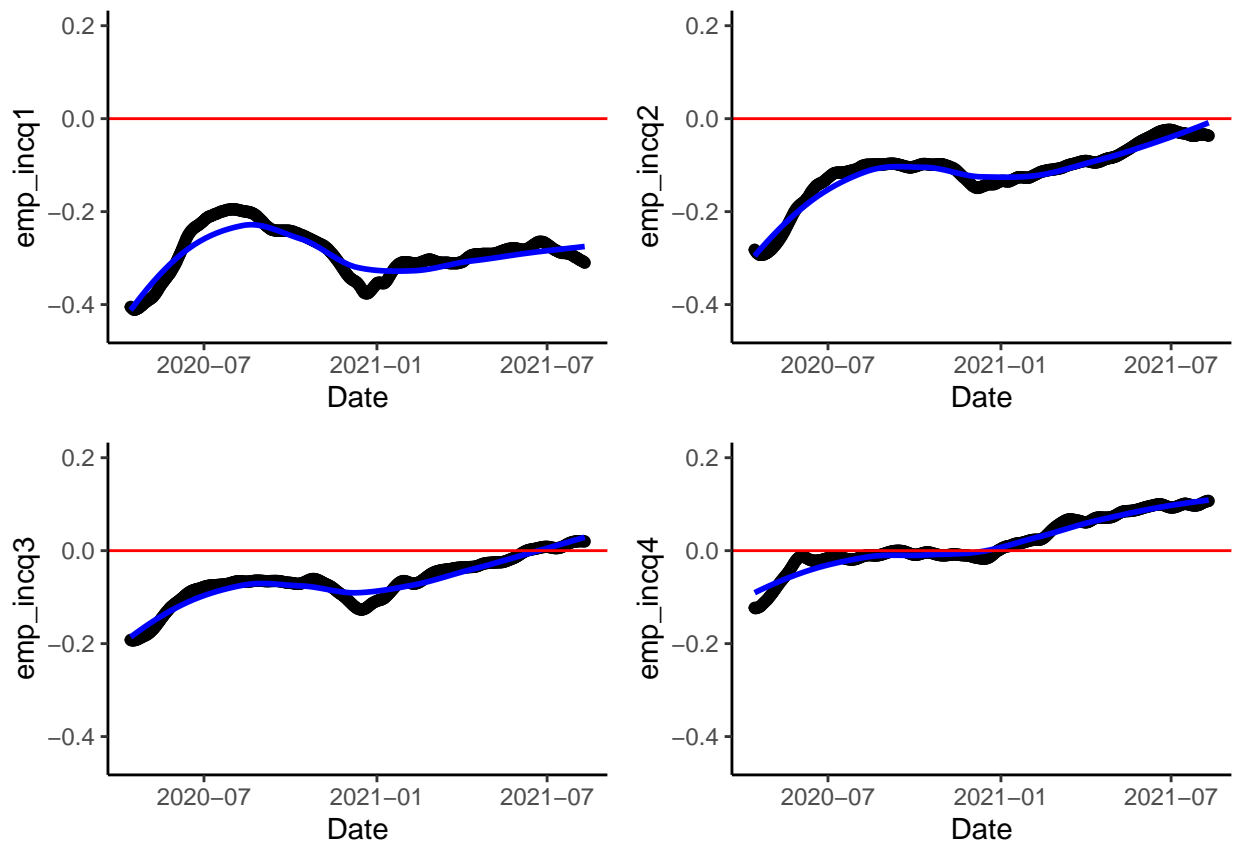
grid.arrange(emp1, emp2, emp3, emp4, ncol=2)

```

```

## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'

```



```

#Visualizing non-employment variables by gps_away_from_home (using minnesota_cut)

#vaccine_corr <-ggplot(minnesota_cut, aes(y = gps_away_from_home, x = fullvaccine_rate)) +
#  geom_point() +
#  geom_smooth(color = "blue", se = FALSE) +
#  geom_hline(yintercept = 0, color = "red") +
#  theme_classic()

case_corr<-ggplot(minnesota_cut, aes(y = gps_away_from_home, x = case_rate)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()

hosp_corr<-ggplot(minnesota_cut, aes(y = gps_away_from_home, x = hospitalized_rate)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()

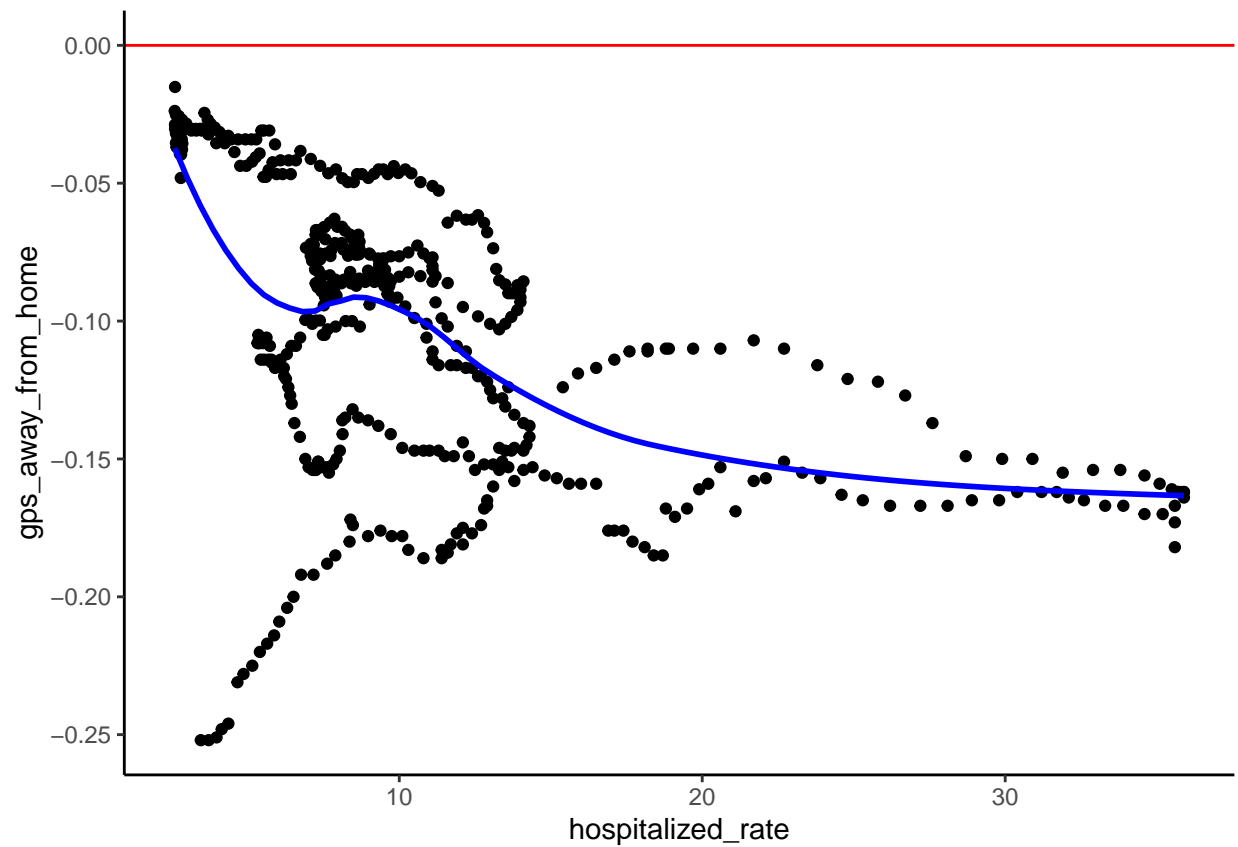
remote_corr<-ggplot(minnesota_cut, aes(y = gps_away_from_home, x = spend_remoteservices)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()

hsc_corr <- ggplot(minnesota_cut, aes(y = gps_away_from_home, x = spend_hcs)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()

hosp_corr

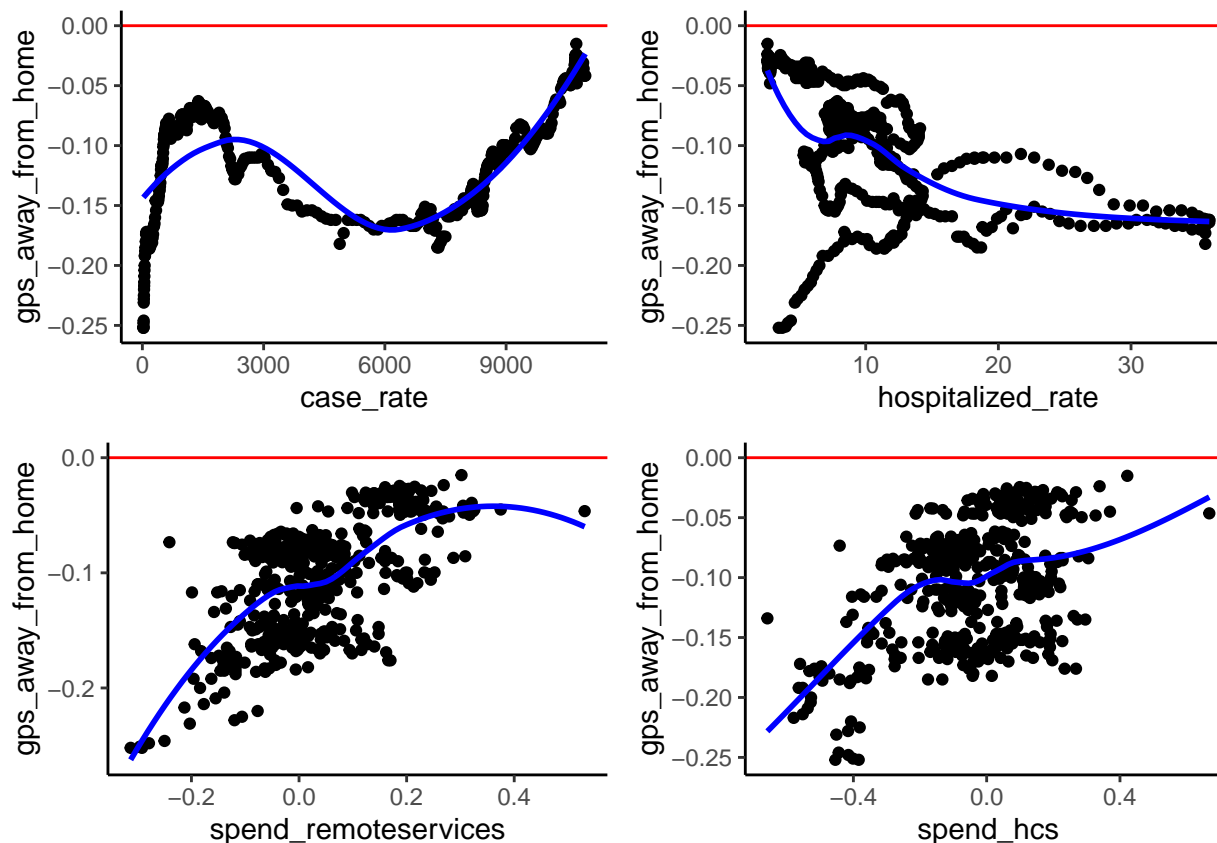
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'

```

```
grid.arrange(case_corr, hosp_corr, remote_corr, hsc_corr, ncol=2)
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'  
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'  
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'  
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



```
#Visualizing employment variables by gps_away_from_home (using minnesota_cut)
emp1_corr <- ggplot(minnesota_cut, aes(y = gps_away_from_home, x = emp_incq1)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
  #+xlim(-0.45, 0.2)

emp2_corr <-ggplot(minnesota_cut, aes(y = gps_away_from_home, x = emp_incq2)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
  #+ xlim(-0.45, 0.2)

emp3_corr <-ggplot(minnesota_cut, aes(y = gps_away_from_home, x = emp_incq3)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
  #+xlim(-0.45, 0.2)

emp4_corr <-ggplot(minnesota_cut, aes(y = gps_away_from_home, x = emp_incq4)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
```

```

theme_classic()
#+xlim(-0.45, 0.2)

empUnder_corr <- ggplot(minnesota_cut, aes(y=gps_away_from_home, x=emp_incbelowmed)) +
  geom_point() +
  geom_smooth(color="blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()

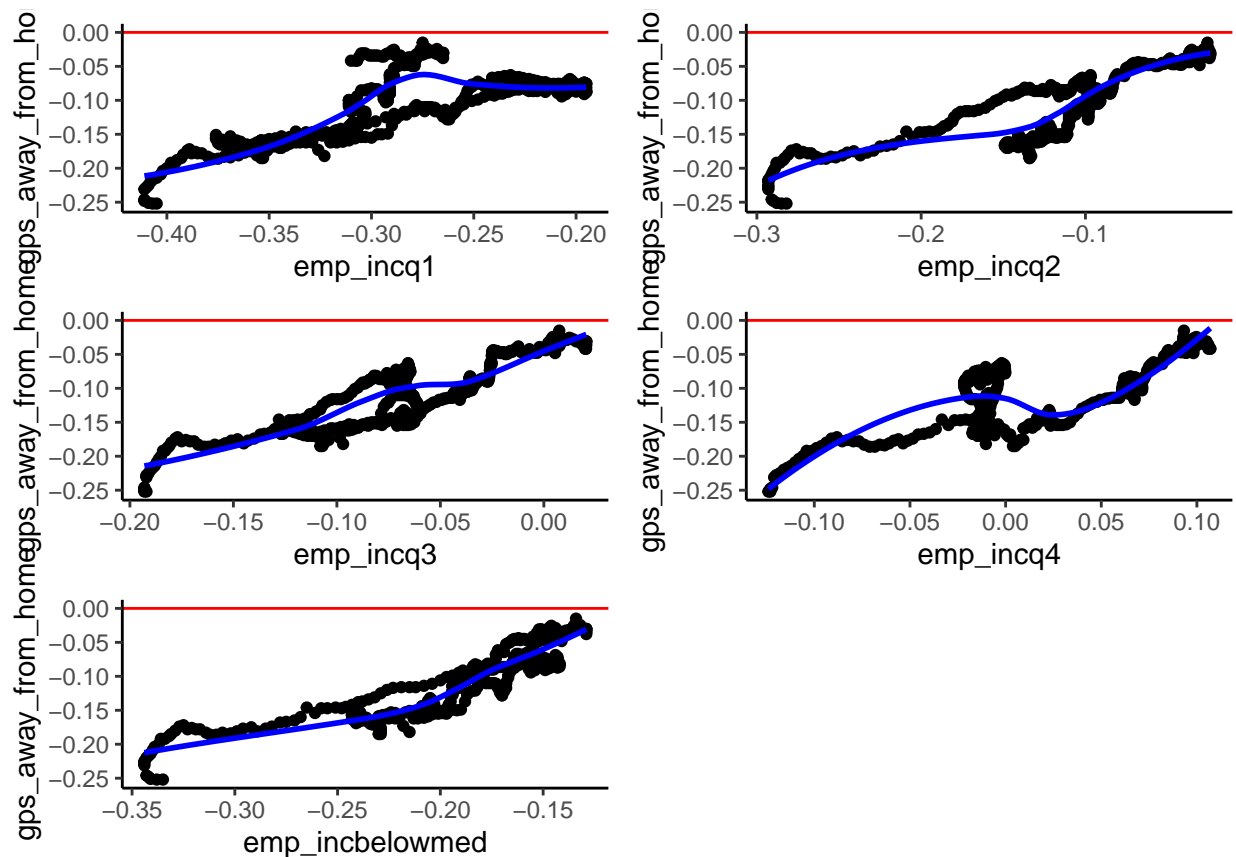
grid.arrange(emp1_corr, emp2_corr, emp3_corr, emp4_corr, empUnder_corr, ncol=2)

```

```

## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'

```



```

#ggplot(mn_mod_output_lasso, aes(y = resid, x = Date)) +
#  geom_point() +
#  geom_smooth(color = "blue", se = FALSE) +
#  geom_hline(yintercept = 0, color = "red") +
#  theme_classic()
#
#tmp <- mn_mod_output_lasso
#tmp$resid

```

Here we can see the data combined. N/A slots represent cases where there was no value to enter or where data was missing.

```
#OLS
set.seed(123)

folded_mn <- vfold_cv(minnesota_cut, v = 6)

lm_spec <-
  linear_reg() %>%
  set_engine(engine = 'lm') %>%
  set_mode('regression')

full_rec <- recipe(gps_away_from_home ~ case_rate + hospitalized_rate + emp_incq1 + emp_incq2 + emp_in
  step_normalize(all_numeric_predictors()) %>%
  step_dummy(all_nominal_predictors())%>%
  step_nzv(all_predictors())

mn_model_wf <- workflow() %>%
  add_recipe(full_rec) %>%
  add_model(lm_spec)

#CV is to see how well the model is doing
mnFullMod_cv <- fit_resamples(mn_model_wf,
  resamples = folded_mn,
  control = control_resamples(save_pred = TRUE),
  metrics = metric_set(rmse, rsq, mae))
```

```
## Warning: package 'rlang' was built under R version 4.1.2
```

```
mnFullMod_cv %>% collect_metrics(summarize=TRUE)
```

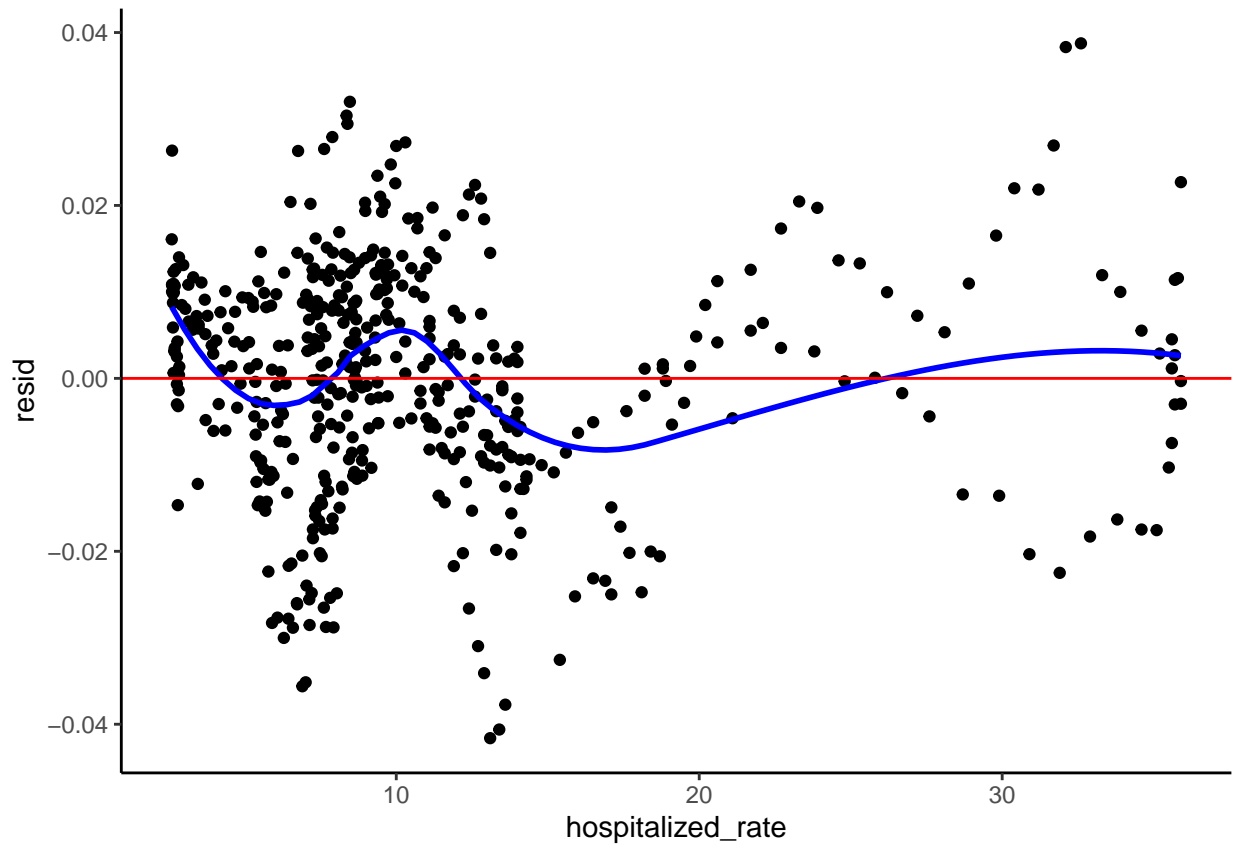
```
## # A tibble: 3 x 6
##   .metric .estimator  mean     n std_err .config
##   <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 mae     standard    0.0107     6 0.000362 Preprocessor1_Model1
## 2 rmse    standard    0.0135     6 0.000447 Preprocessor1_Model1
## 3 rsq     standard    0.926      6 0.00653  Preprocessor1_Model1
```

```
mn_mod <- mn_model_wf %>% fit(data=minnesota_cut)
```

```
mn_mod_output_OLS <- mn_mod %>%
  predict(new_data=minnesota_cut) %>%
  bind_cols(minnesota_cut)%>%
  mutate(resid = gps_away_from_home - .pred)

ggplot(mn_mod_output_OLS, aes(y = resid, x = hospitalized_rate)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
```

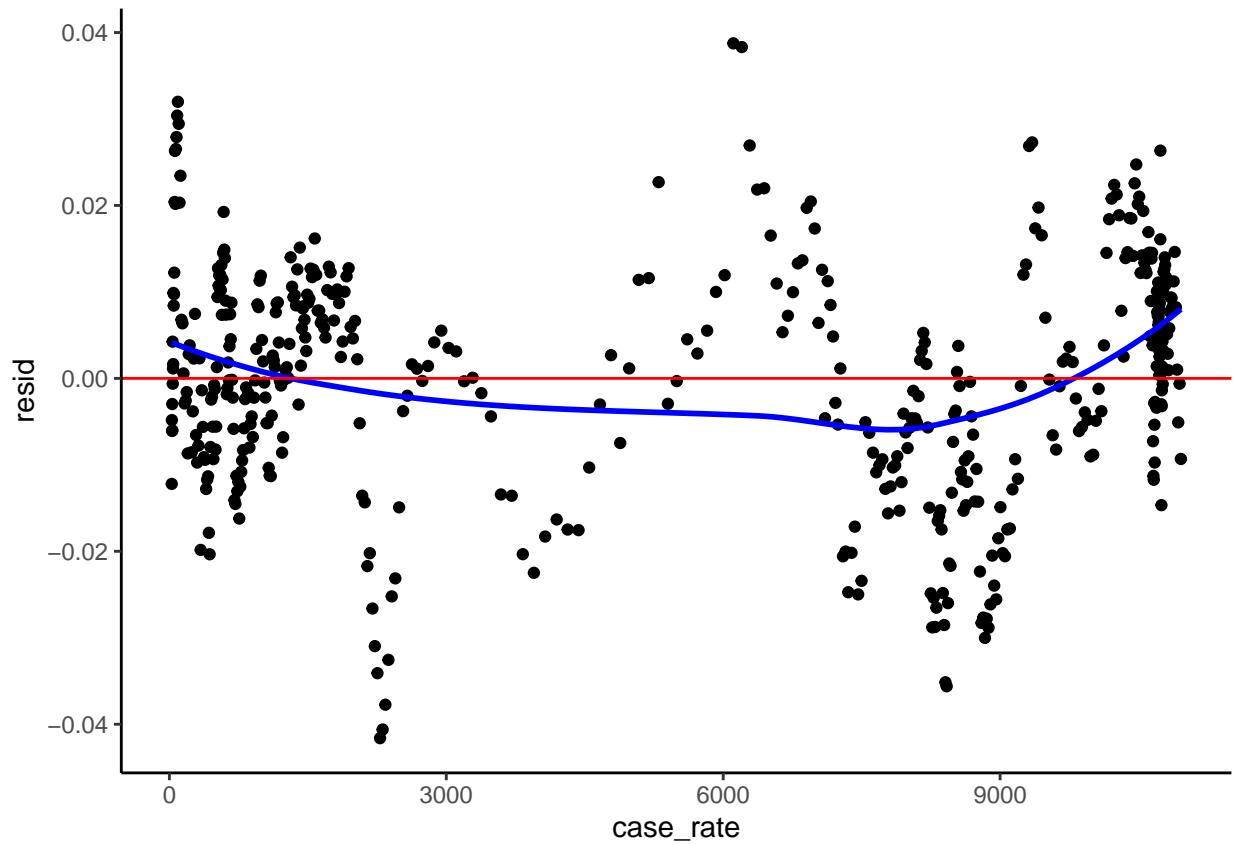
```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



```
#ggplot(mn_mod_output_OLS, aes(y = resid, x = fullvaccine_rate)) +
#   geom_point() +
#   geom_smooth(color = "blue", se = FALSE) +
#   geom_hline(yintercept = 0, color = "red") +
#   theme_classic()
```

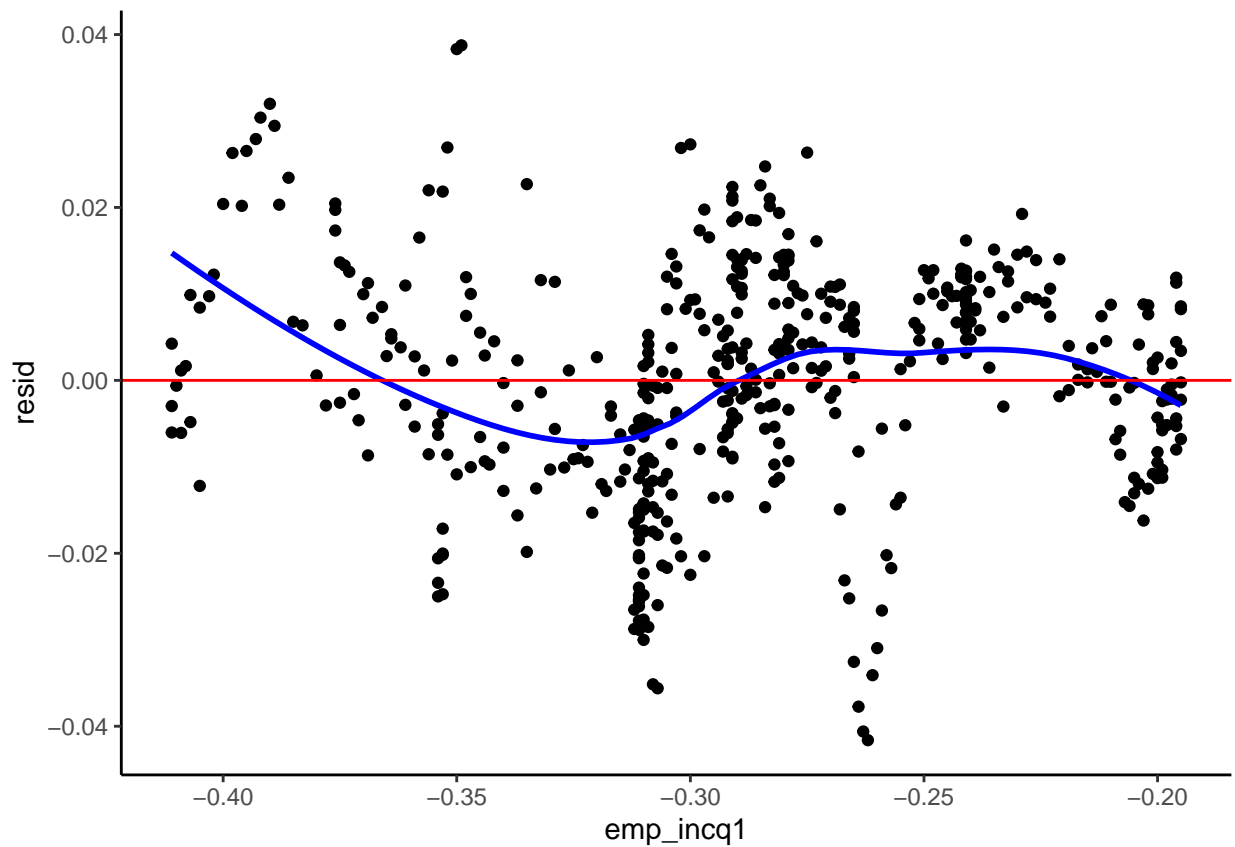
```
ggplot(mn_mod_output_OLS, aes(y = resid, x = case_rate)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



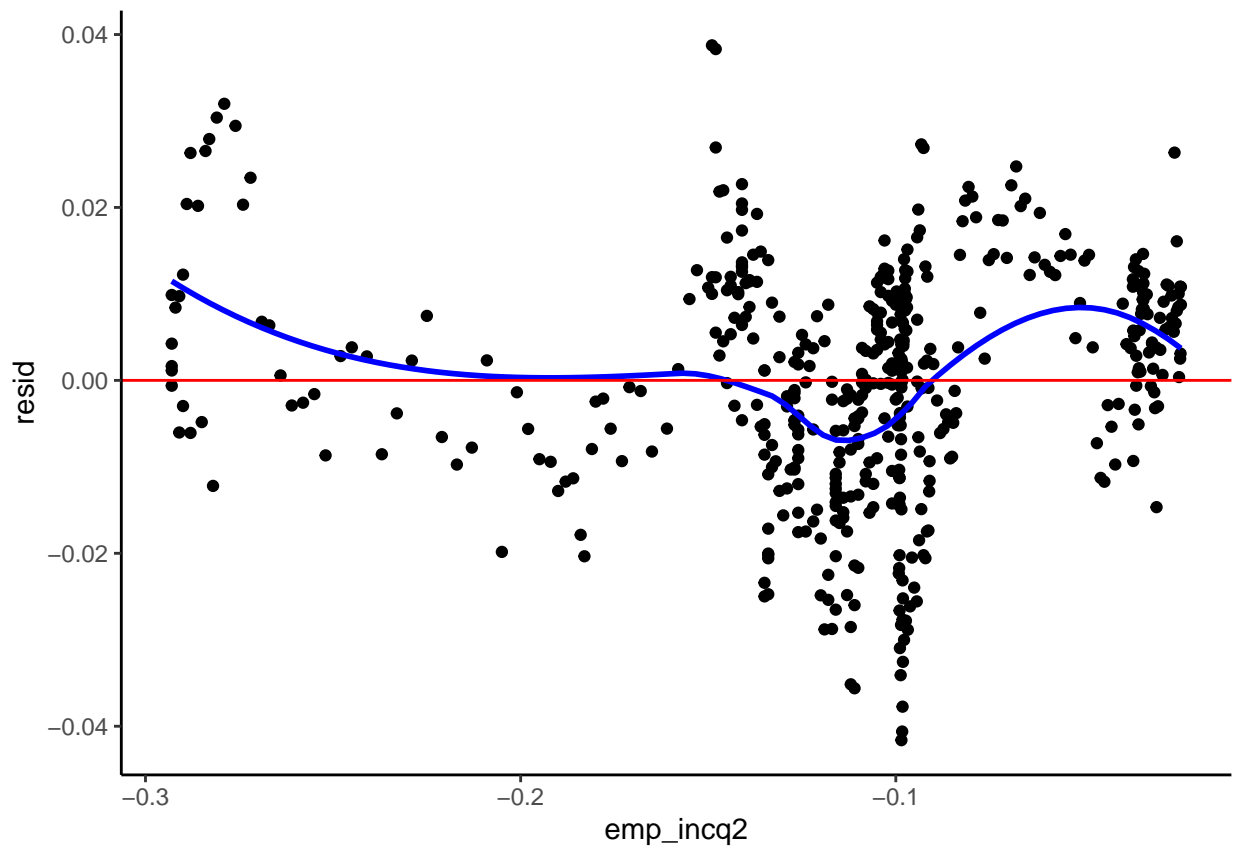
```
ggplot(mn_mod_output_OLS, aes(y = resid, x = emp_incq1)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



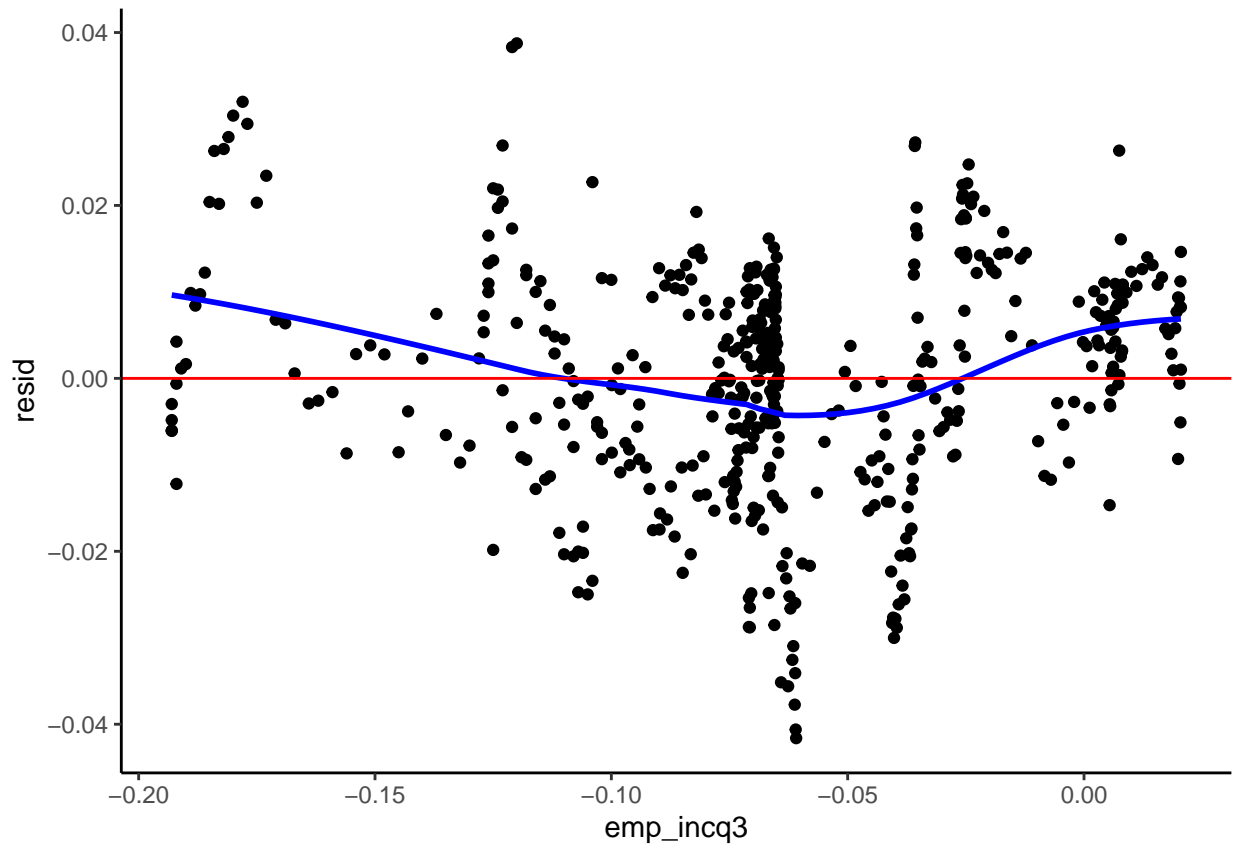
```
ggplot(mn_mod_output_OLS, aes(y = resid, x = emp_incq2)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



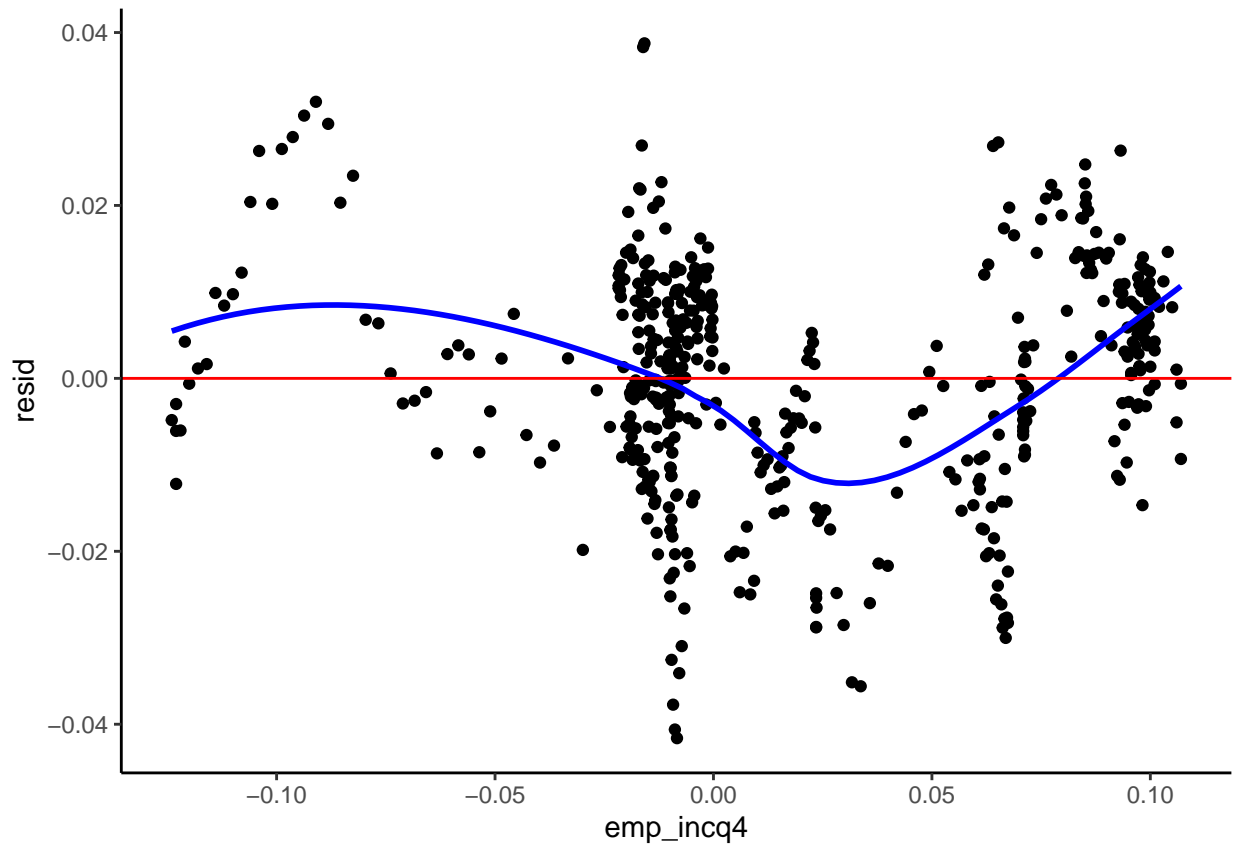
```
ggplot(mn_mod_output_OLS, aes(y = resid, x = emp_incq3)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```

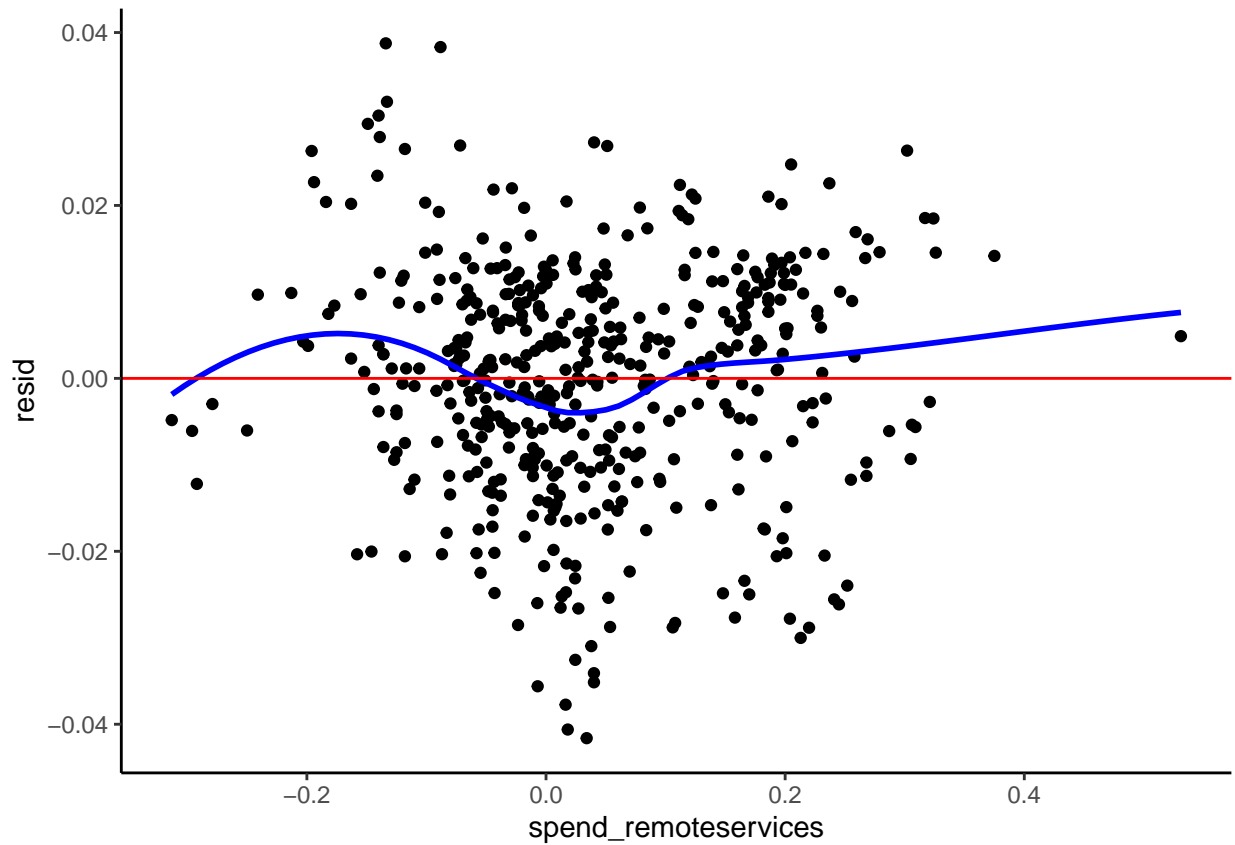
```
ggplot(mn_mod_output_OLS, aes(y = resid, x = emp_incq4)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



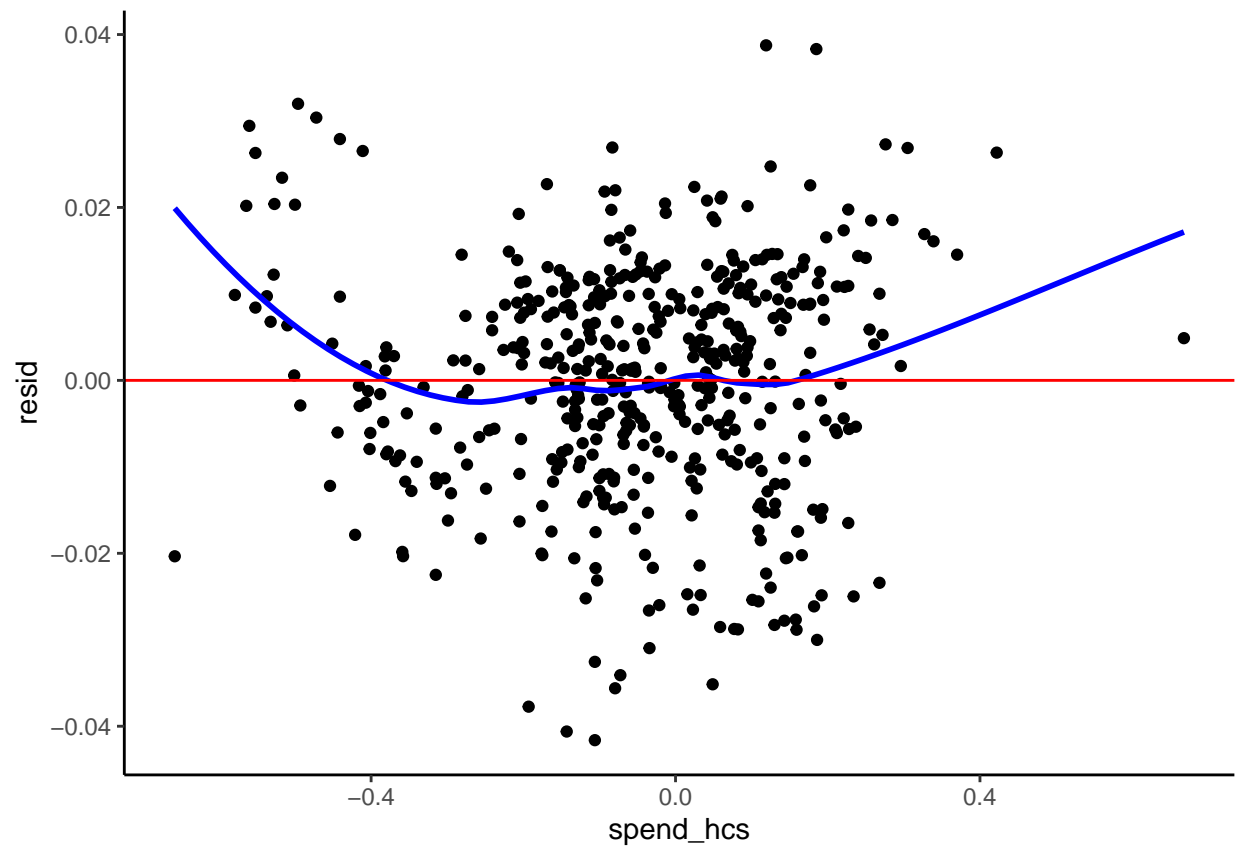
```
ggplot(mn_mod_output_OLS, aes(y = resid, x = spend_remoteservices)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



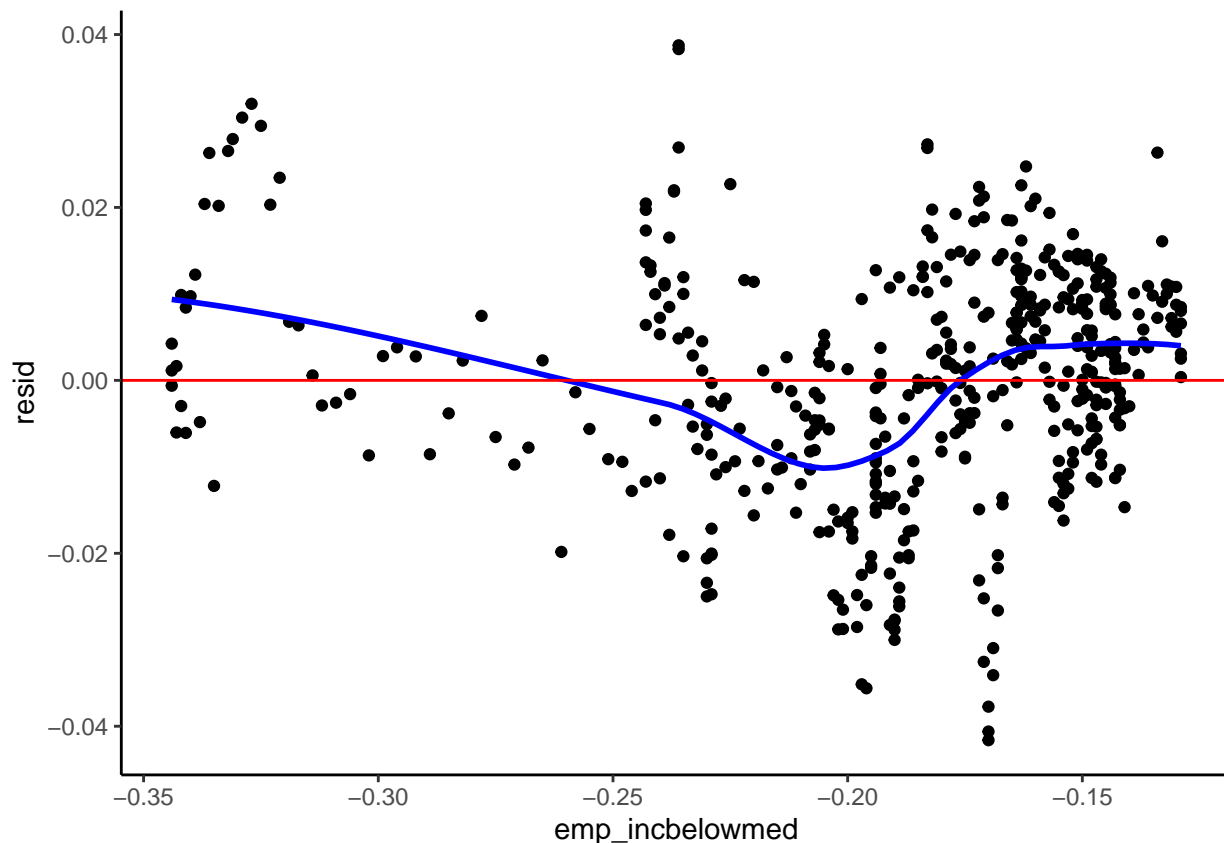
```
ggplot(mn_mod_output_OLS, aes(y = resid, x = spend_hcs)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



```
ggplot(mn_mod_output_OLS, aes(y = resid, x = emp_incbelowmed)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



Here we have the summery data for our basic best fit model. We can see that the standard error is about three hundredths of a percent; this is very low for predicting in a range of up to 10 % change but is more significant for most days when there is very little change in average movement.

```
#LASSO using minnesota_cut
set.seed(123)

folded_mn <- vfold_cv(minnesota_cut, v = 6)

lm_lasso_spec <-
  linear_reg() %>%
  set_args(mixture = 1, penalty = tune()) %>% ## mixture = 1 indicates Lasso, we'll choose penalty later
  set_engine(engine = 'glmnet') %>%
  set_mode('regression')

full_lasso_rec <- recipe(gps_away_from_home ~ case_rate + hospitalized_rate + emp_incq1 + emp_incq2 +
  step_normalize(all_numeric_predictors()) %>%
  step_dummy(all_nominal_predictors())%>%
  step_nzv(all_predictors())

mn_lasso_wf_tune <- workflow() %>%
  add_recipe(full_lasso_rec) %>%
  add_model(lm_lasso_spec)

# Tune Model (trying a variety of values of Lambda penalty)
penalty_grid <- grid_regular(
```

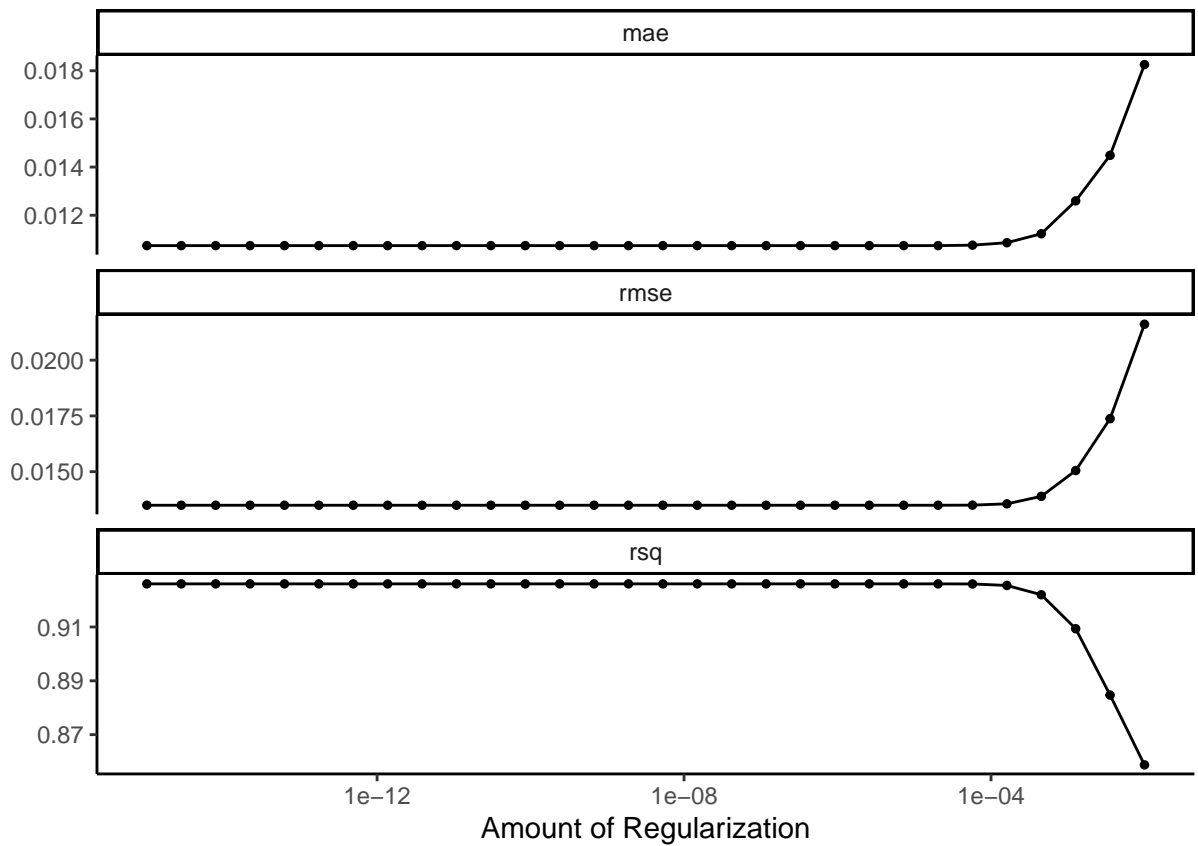
```

penalty(range = c(-15, -2)), #log10 transformed 10^-5 to 10^3
levels = 30)

tune_res <- tune_grid( # new function for tuning parameters
  mn_lasso_wf_tune, # workflow
  resamples = folded_mn, # cv folds
  metrics = metric_set(rmse, rsq, mae),
  grid = penalty_grid # penalty grid defined above
)

# Visualize Model Evaluation Metrics from Tuning
autoplot(tune_res) + theme_classic()

```



```

# Summarize Model Evaluation Metrics (CV)
collect_metrics(tune_res) %>%
  filter(.metric == 'rmse') %>% # or choose mae
  select(penalty, rmse = mean)

```

```

## # A tibble: 30 x 2
##   penalty  rmse
##   <dbl>  <dbl>
## 1 1e-15 0.0135
## 2 2.81e-15 0.0135
## 3 7.88e-15 0.0135
## 4 2.21e-14 0.0135

```

```
## 5 6.21e-14 0.0135
## 6 1.74e-13 0.0135
## 7 4.89e-13 0.0135
## 8 1.37e-12 0.0135
## 9 3.86e-12 0.0135
## 10 1.08e-11 0.0135
## # ... with 20 more rows
```

```
best_penalty <- select_best(tune_res, metric = 'rmse') # choose penalty value based on lowest mae or rmse
```

```
# Fit Final Model
```

```
final_wf <- finalize_workflow(mn_lasso_wf_tune, best_penalty) # incorporates penalty value to workflow
```

```
final_fit <- fit(final_wf, data = minnesota_cut)
```

```
tidy(final_fit)
```

```
## # A tibble: 10 x 3
##   term                estimate penalty
##   <chr>              <dbl>    <dbl>
## 1 (Intercept)       -0.104    1e-15
## 2 case_rate         -0.0244   1e-15
## 3 hospitalized_rate -0.00791  1e-15
## 4 emp_incq1          0         1e-15
## 5 emp_incq2          0.0113   1e-15
## 6 emp_incq3          0.0116   1e-15
## 7 emp_incq4          0.0236   1e-15
## 8 spend_remoteservices 0.0127   1e-15
## 9 spend_hcs          -0.00991  1e-15
## 10 emp_incbelowmed    0.0128   1e-15
```

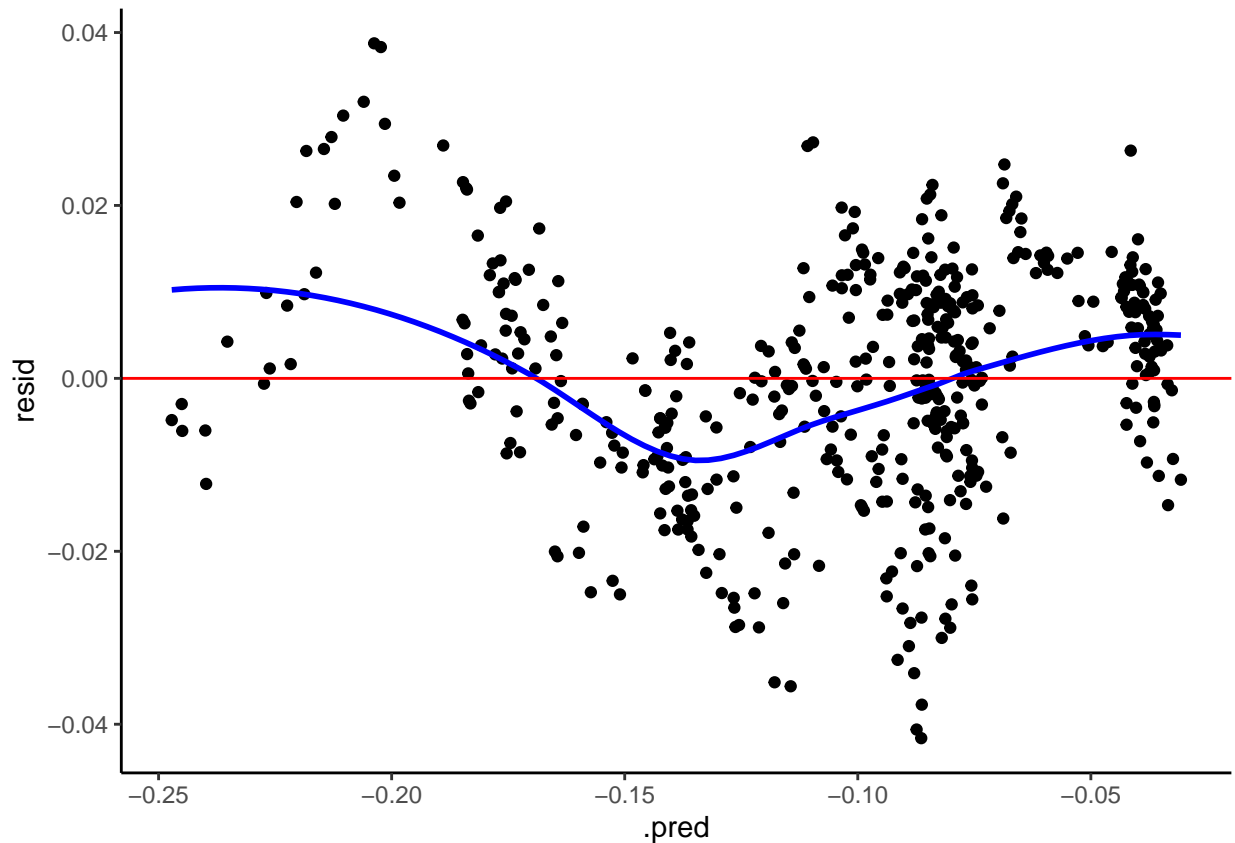
Here we can see that the Penalty for the lasso model has very little effect on the RMSE until it gets quite high. This is probably because some predictors are quickly eliminated and the more important ones are not removed until much later.

```
#OLS
```

```
mn_mod_output <- mn_mod %>%
  predict(new_data = minnesota_cut) %>%
  bind_cols(minnesota_cut)%>%
  mutate(resid = gps_away_from_home - .pred)

ggplot(mn_mod_output, aes(y = resid, x = .pred)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```

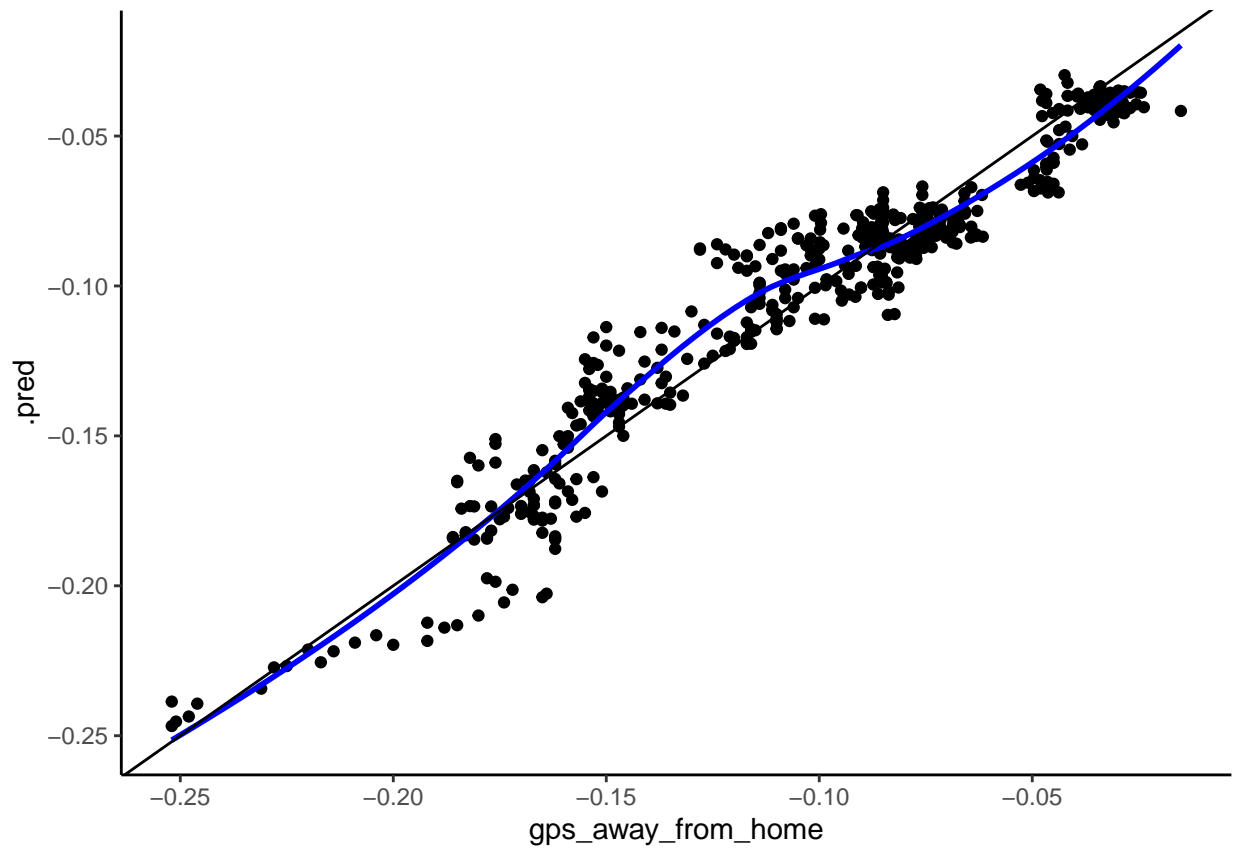


Our model seems to be fairly randomly distributed around the middle line. There is, however, a concerning lack of predictions in the area above the 0 line around -0.09 and below the 0 line around -0.045.

```
#LASSO
mn_mod_output_lasso <- final_fit %>%
  predict(new_data=minnesota_cut) %>%
  bind_cols(minnesota_cut)%>%
  mutate(resid = gps_away_from_home - .pred)

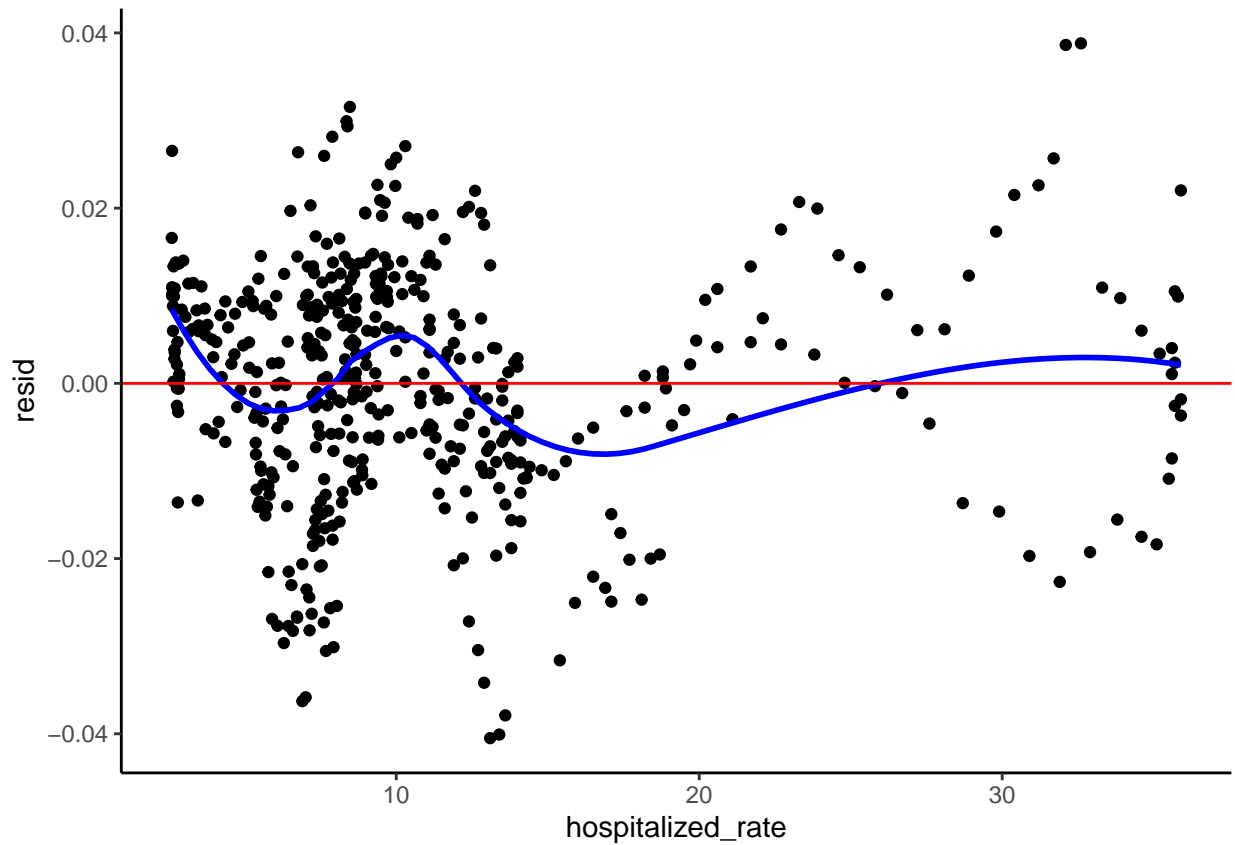
ggplot(mn_mod_output_lasso, aes(y = .pred, x = gps_away_from_home)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_abline(intercept = 0, slope = 1) +
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```

```
ggplot(mn_mod_output_lasso, aes(y = resid, x = hospitalized_rate)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

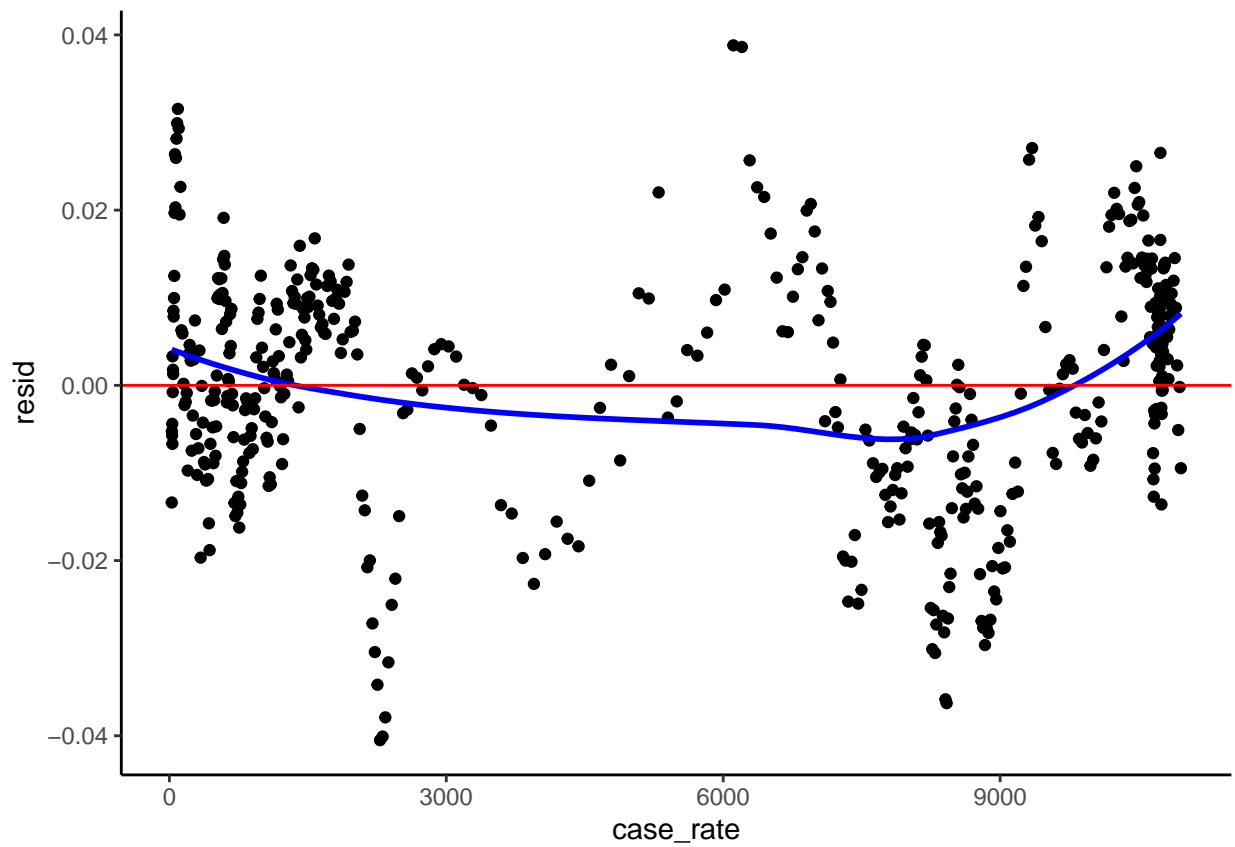
```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



```
#ggplot(mn_mod_output_lasso, aes(y = resid, x = fullvaccine_rate)) +
#  geom_point() +
#  geom_smooth(color = "blue", se = FALSE) +
#  geom_hline(yintercept = 0, color = "red") +
#  theme_classic()

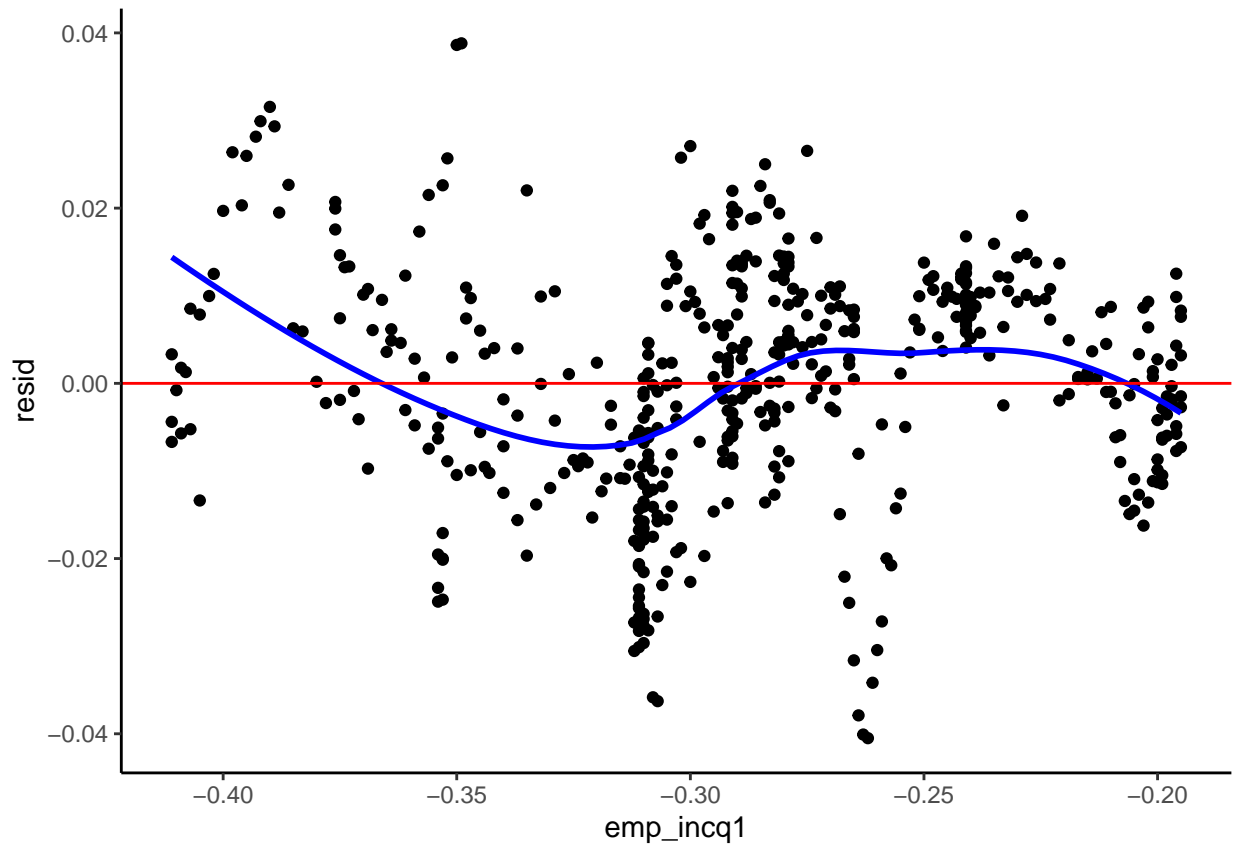
ggplot(mn_mod_output_lasso, aes(y = resid, x = case_rate)) +
  geom_point() +
  geom_smooth(color = "blue", se = FALSE) +
  geom_hline(yintercept = 0, color = "red") +
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



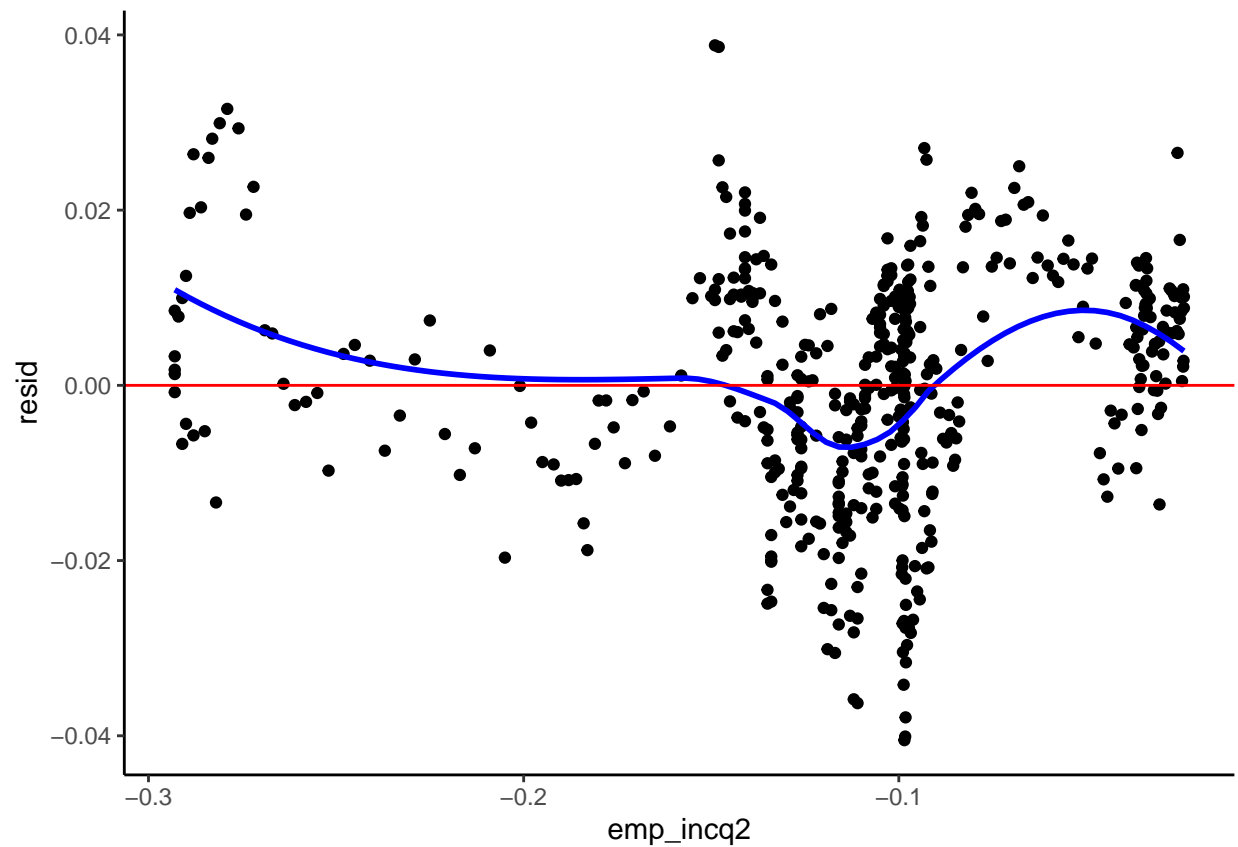
```
ggplot(mn_mod_output_lasso, aes(y = resid, x = emp_incq1)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



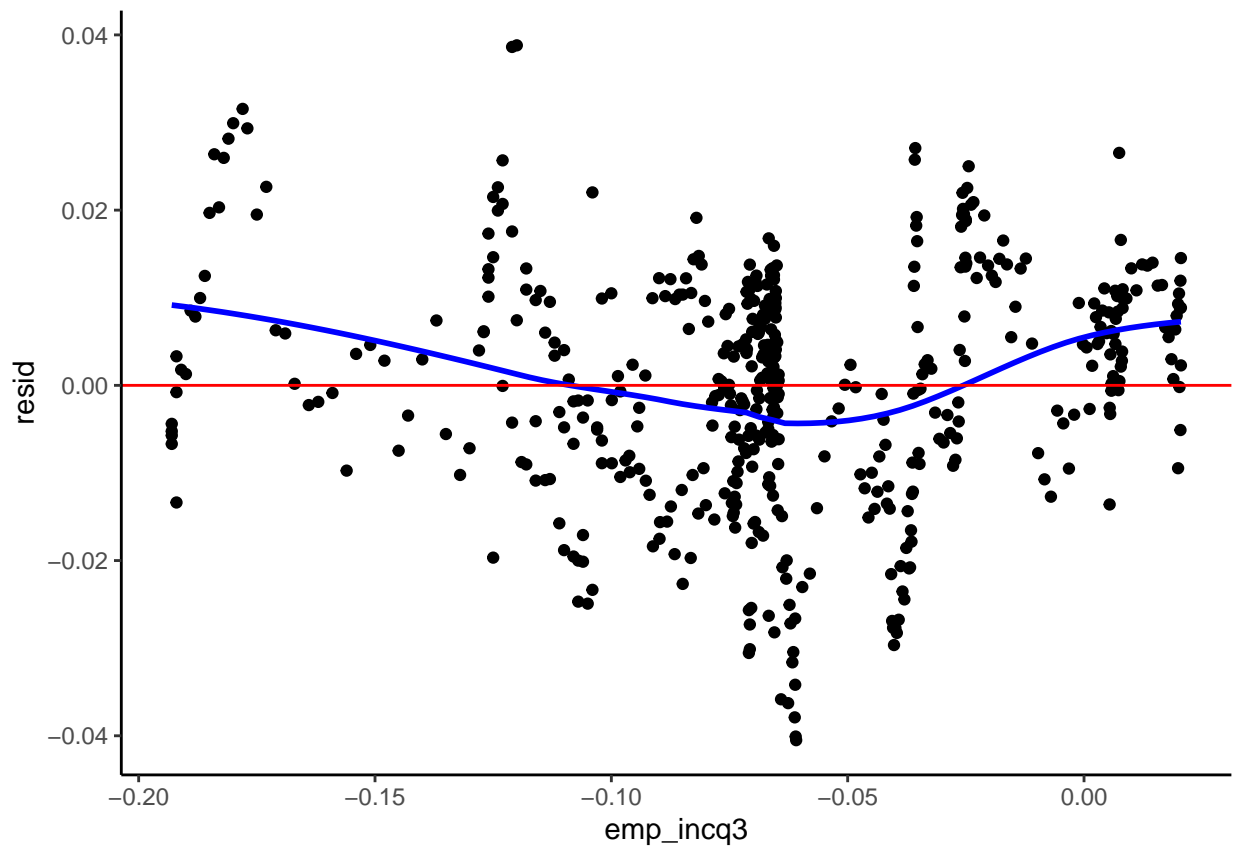
```
ggplot(mn_mod_output_lasso, aes(y = resid, x = emp_incq2)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



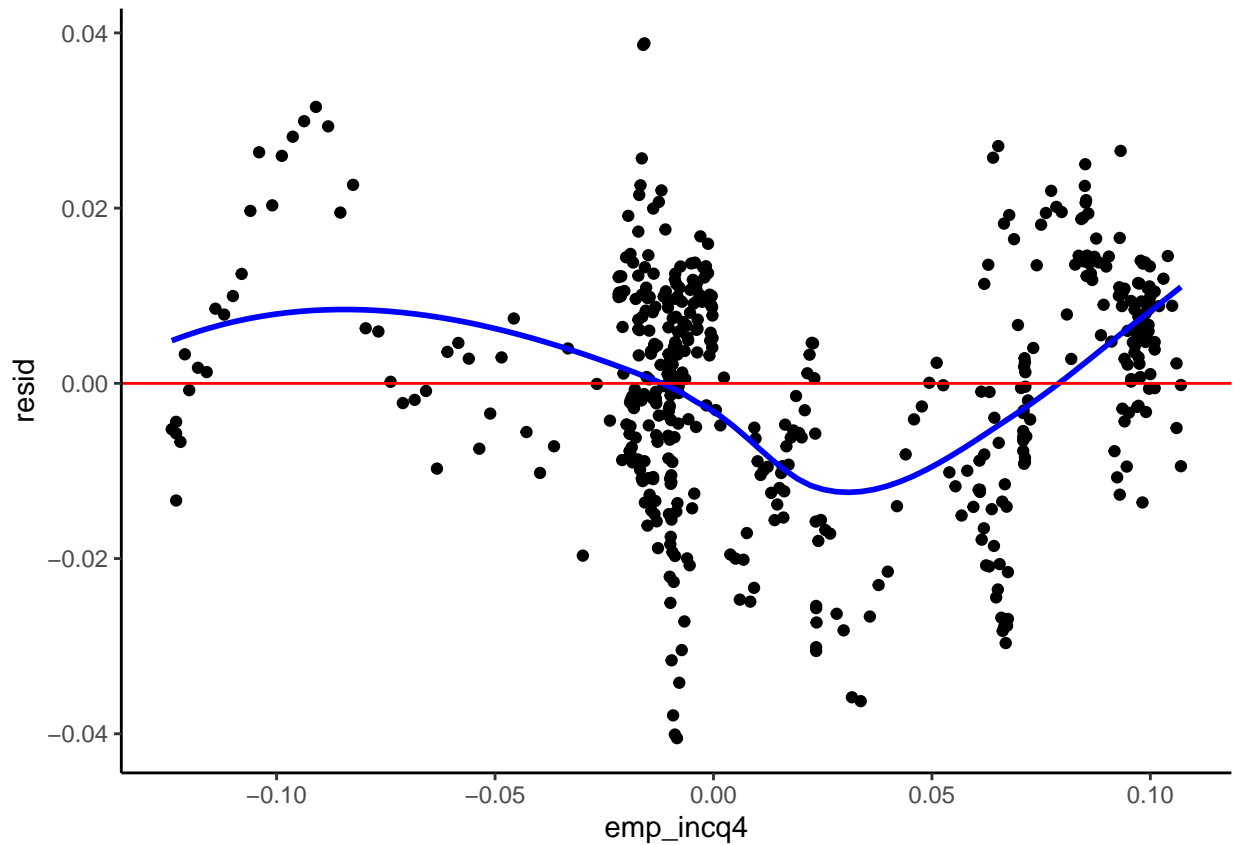
```
ggplot(mn_mod_output_lasso, aes(y = resid, x = emp_incq3)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



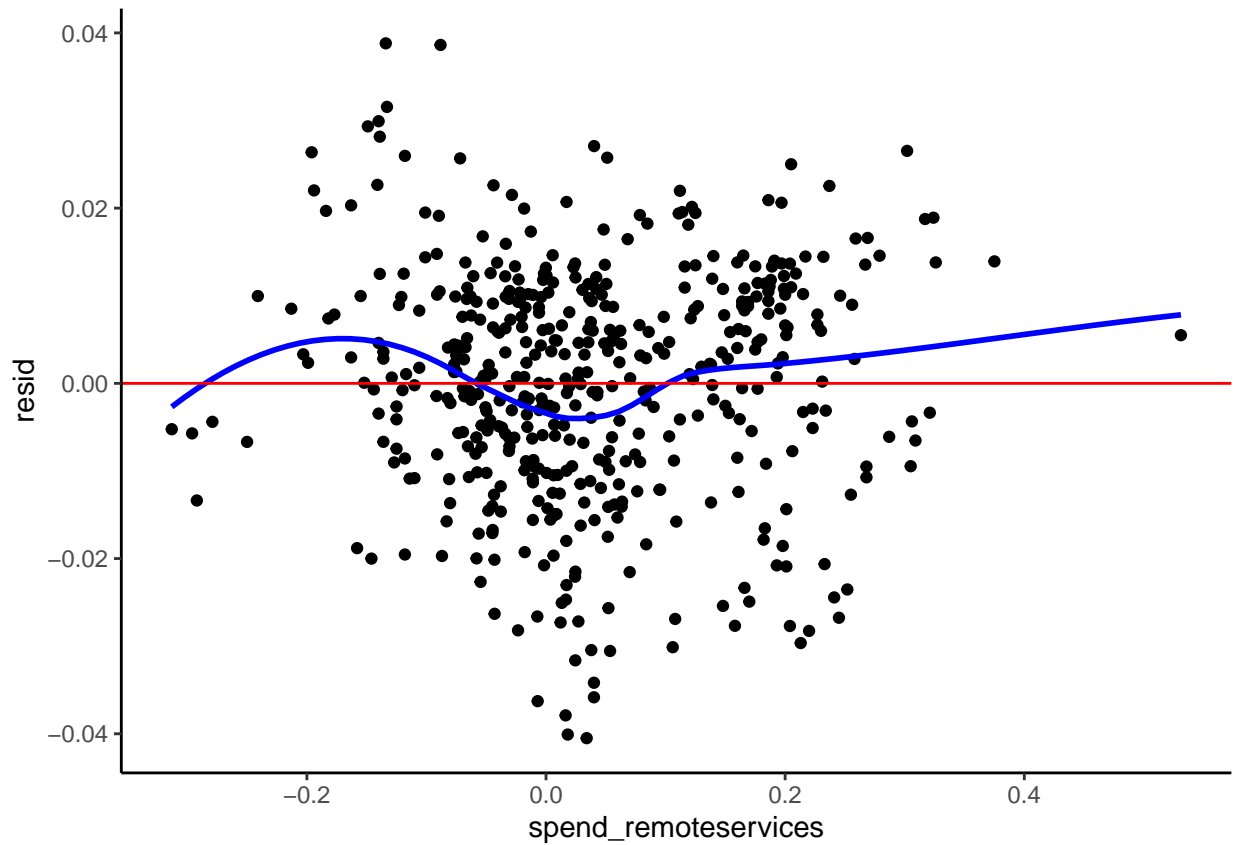
```
ggplot(mn_mod_output_lasso, aes(y = resid, x = emp_incq4)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



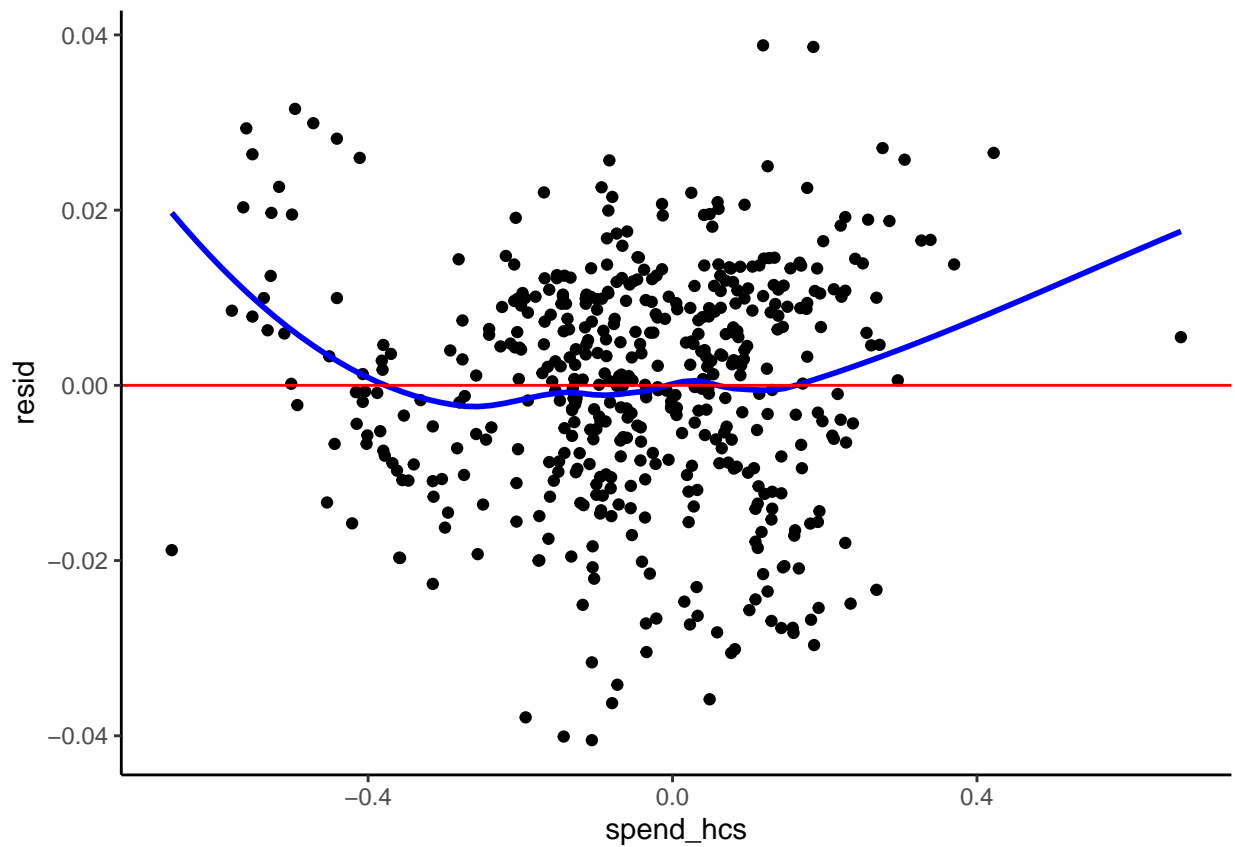
```
ggplot(mn_mod_output_lasso, aes(y = resid, x = spend_remoteservices)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



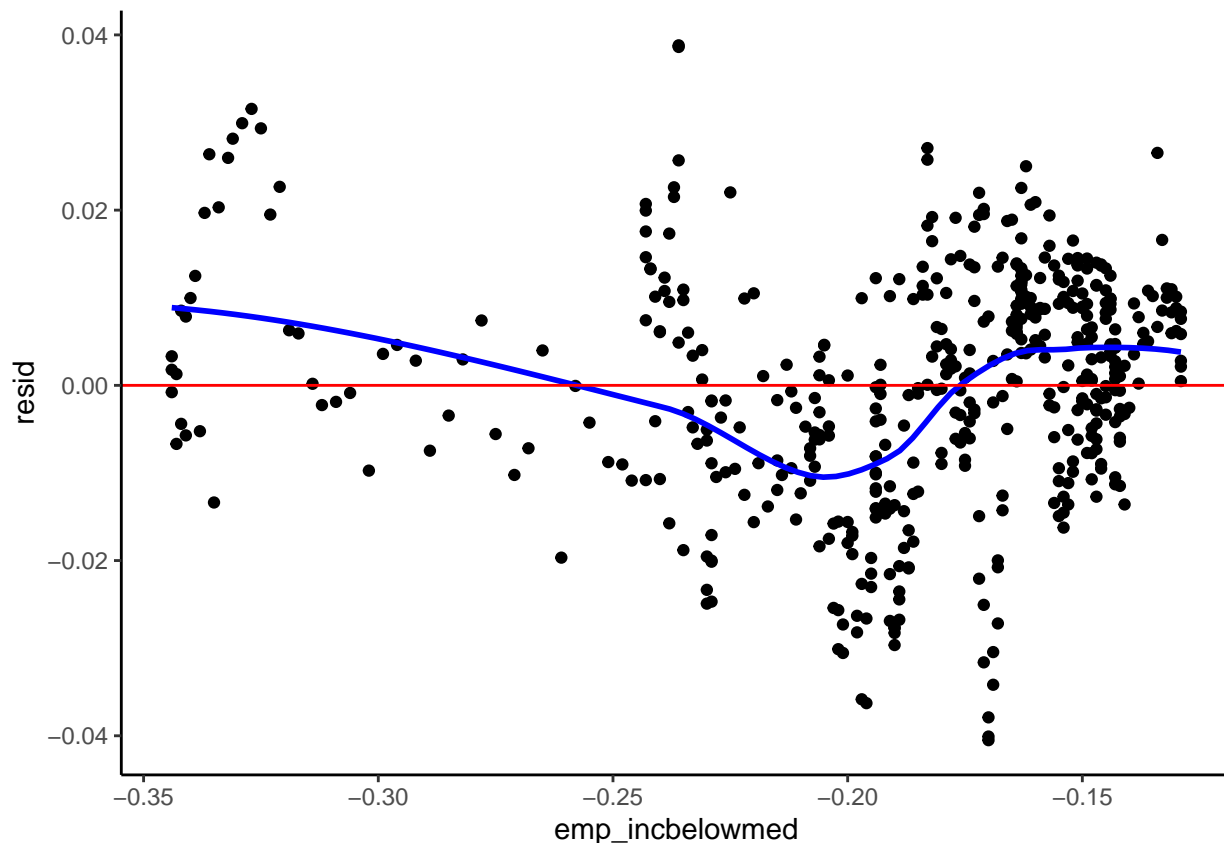
```
ggplot(mn_mod_output_lasso, aes(y = resid, x = spend_hcs)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```

```
ggplot(mn_mod_output_lasso, aes(y = resid, x = emp_incbelowmed)) +  
  geom_point() +  
  geom_smooth(color = "blue", se = FALSE) +  
  geom_hline(yintercept = 0, color = "red") +  
  theme_classic()
```

```
## 'geom_smooth()' using method = 'loess' and formula 'y ~ x'
```



Overall our residuals are presenting some strange trends. Some of our predictions are taking weird paths into some kind of very negative ark. Many of our predictors have strong curves that look like some kind of spline. Since some of them are eliminated very quickly by our LASSO method I would not worry about several of these but emp_incb3 seems to have some strange patterns in it we should probably investigate.

```
best_penalty <- select_best(tune_res, metric = 'mae') # choose penalty value based on lowest cv mae
best_penalty
```

```
## # A tibble: 1 x 2
##   penalty .config
##   <dbl> <chr>
## 1 1e-15 Preprocessor1_Model01
```

```
best_se_penalty <- select_by_one_std_err(tune_res, metric = 'mae', desc(penalty))
```

```
glmnet_output <- final_fit %>% extract_fit_parsnip() %>% pluck('fit') # get the original glmnet output
```

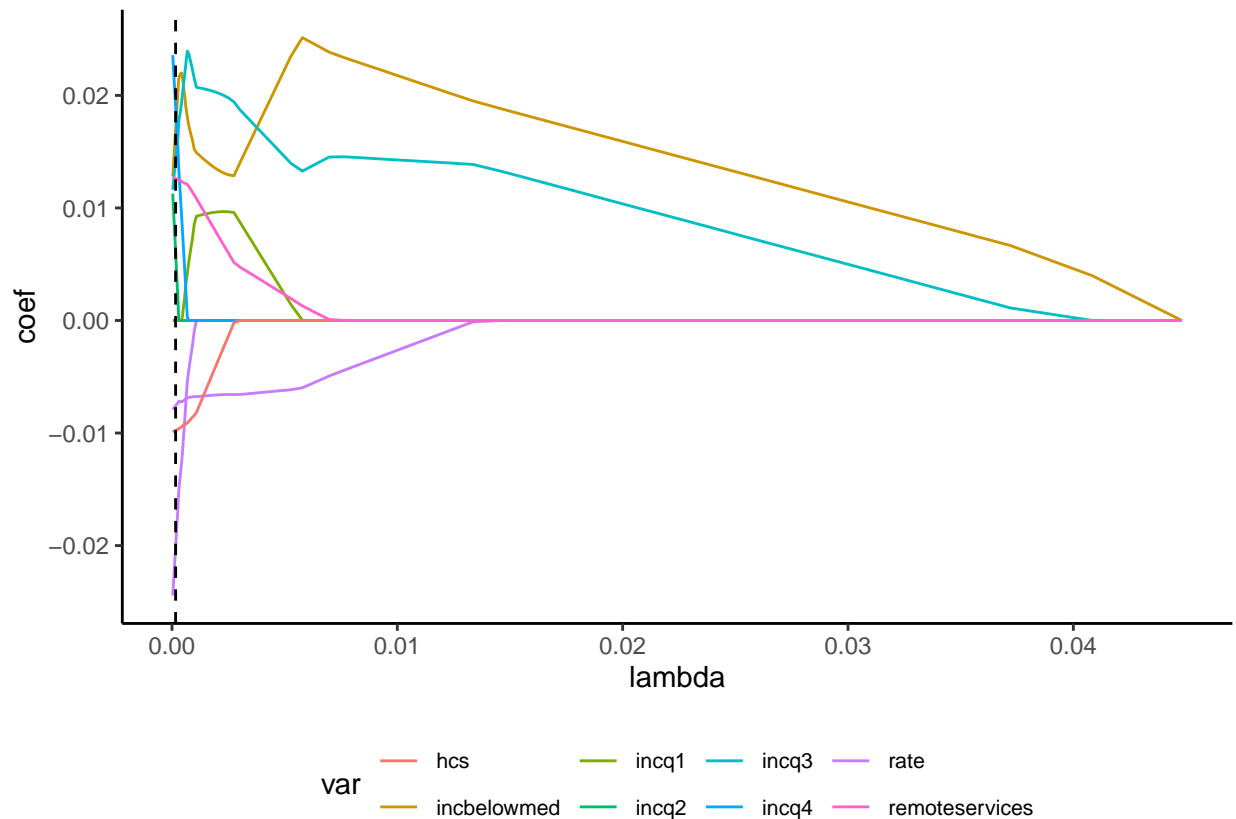
```
lambdas <- glmnet_output$lambda
coefs_lambdas <-
  coefficients(glmnet_output, s = lambdas ) %>%
  as.matrix() %>%
  t() %>%
  as.data.frame() %>%
  mutate(lambda = lambdas ) %>%
  select(lambda, everything(), -(Intercept)) %>%
```

```

pivot_longer(cols = -lambda,
              names_to = "term",
              values_to = "coef") %>%
mutate(var = purrr::map_chr(stringr::str_split(term, "_"), ~.[2]))

coefs_lambdas %>%
  ggplot(aes(x = lambda, y = coef, group = term, color = var)) +
  geom_line() +
  geom_vline(xintercept = best_se_penalty %>% pull(penalty), linetype = 'dashed') +
  theme_classic() +
  theme(legend.position = "bottom", legend.text=element_text(size=8))

```



```
final_fit %>% tidy() %>% filter(estimate != 0)
```

```
## # A tibble: 9 x 3
##   term                estimate penalty
##   <chr>              <dbl>    <dbl>
## 1 (Intercept)      -0.104    1e-15
## 2 case_rate        -0.0244    1e-15
## 3 hospitalized_rate -0.00791   1e-15
## 4 emp_incq2         0.0113    1e-15
## 5 emp_incq3         0.0116    1e-15
## 6 emp_incq4         0.0236    1e-15
## 7 spend_remoteservices 0.0127    1e-15
## 8 spend_hcs         -0.00991   1e-15
```

```
## 9 emp_incbelowmed      0.0128    1e-15
```

The predictor with the highest estimate (seen both in our tidy output and in the coefficient path visualization) is `emp_incq3`, which is the employment level for workers in the third quartile of the income distribution. However, this is part of a categorical variable, so we will consider employment level as the most important predictor. This makes contextual sense, as employment levels greatly influence how much time outside the house an individual can have. We arrived at the same outcome in LASSO as we did in OLS.