Analysis

Imports

```
library(mgcv)
library(tidyverse)
library(caret)
library(mice)
library(quantreg)

df <- read.csv("data/train_simple.csv", stringsAsFactors = TRUE)

train_idx <- read.csv("data/X_train.csv")$X
train_df <- df[train_idx, ]
test_df <- df[-train_idx, ]
rm(df)
```

Define metric

```
mean_abs_err <- function(y_hat, y){
  sum(abs(y_hat - y)) / length(y_hat)
}
```

Fit linear regression without imputing income

As lots of the incomes are zero, it makes sense to impute these, as employment status is a categorical variable anyway which accounts for those with no income. First we can get a baseline without imputing income.

```
mod <- lm(total_claim_amount ~ vehicle_class + income +
             employment_status*monthly_premium_auto +
             location_code*monthly_premium_auto, data = train_df)

y_hat <- predict(mod, newdata = test_df)

summary(mod)
```

```
##
## Call:
## lm(formula = total_claim_amount ~ vehicle_class + income + employment_status *
##     monthly_premium_auto + location_code * monthly_premium_auto,
##     data = train_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -540.40  -64.47  -25.83   57.67 1357.87
##
## Coefficients:
##                                                        Estimate Std. Error
```

```
## (Intercept)                                       -5.416e+01  2.503e+01
## vehicle_classLuxury Car                            3.358e+01  1.752e+01
## vehicle_classLuxury SUV                            2.560e+01  1.691e+01
## vehicle_classSports Car                           -6.369e+00  8.201e+00
## vehicle_classSUV                                   6.486e-01  5.864e+00
## vehicle_classTwo-Door Car                          2.863e+00  4.098e+00
## income                                            -1.446e-04  9.090e-05
## employment_statusEmployed                          7.421e+01  2.242e+01
## employment_statusMedical Leave                    -3.799e+01  3.220e+01
## employment_statusRetired                           1.210e+02  3.358e+01
## employment_statusUnemployed                       -1.059e+01  2.330e+01
## monthly_premium_auto                               1.832e+00  2.570e-01
## location_codeSuburban                              9.448e+00  1.293e+01
## location_codeUrban                                 1.094e+01  1.616e+01
## employment_statusEmployed:monthly_premium_auto     -7.923e-01  2.208e-01
## employment_statusMedical Leave:monthly_premium_auto  7.679e-01  3.312e-01
## employment_statusRetired:monthly_premium_auto      -1.459e+00  3.407e-01
## employment_statusUnemployed:monthly_premium_auto    1.157e+00  2.319e-01
## monthly_premium_auto:location_codeSuburban          4.173e+00  1.338e-01
## monthly_premium_auto:location_codeUrban             2.322e+00  1.703e-01
##                                                    t value Pr(>|t|)
## (Intercept)                                         -2.163 0.030556 *
## vehicle_classLuxury Car                              1.917 0.055300 .
## vehicle_classLuxury SUV                              1.513 0.130213
## vehicle_classSports Car                             -0.777 0.437429
## vehicle_classSUV                                     0.111 0.911930
## vehicle_classTwo-Door Car                            0.699 0.484775
## income                                              -1.591 0.111674
## employment_statusEmployed                            3.309 0.000941 ***
## employment_statusMedical Leave                      -1.180 0.238188
## employment_statusRetired                             3.603 0.000317 ***
## employment_statusUnemployed                         -0.455 0.649413
## monthly_premium_auto                                 7.127 1.14e-12 ***
## location_codeSuburban                                0.731 0.465086
## location_codeUrban                                   0.677 0.498324
## employment_statusEmployed:monthly_premium_auto      -3.588 0.000336 ***
## employment_statusMedical Leave:monthly_premium_auto   2.319 0.020454 *
## employment_statusRetired:monthly_premium_auto       -4.284 1.87e-05 ***
## employment_statusUnemployed:monthly_premium_auto     4.987 6.29e-07 ***
## monthly_premium_auto:location_codeSuburban          31.179  < 2e-16 ***
## monthly_premium_auto:location_codeUrban             13.632  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 123.4 on 6206 degrees of freedom
## Multiple R-squared:  0.8159, Adjusted R-squared:  0.8153
## F-statistic:  1447 on 19 and 6206 DF,  p-value: < 2.2e-16
```

```r
mae <- mean_abs_err(y_hat, test_df$total_claim_amount)
print(paste("MAE of:", round(mae, 2)))
```

```
## [1] "MAE of: 89.55"
```

Try imputing income variable

```
df_big <- read.csv("data/train.csv", stringsAsFactors = TRUE)
df_big <- df_big %>% select(-Country, -Customer)
df_big$Income[df_big$Income == 0] <- NA

imputations <- complete(mice(df_big, method = "pmm", seed=1))
```

```
## Warning: Number of logged events: 26
```

```
train_df$income <- imputations$Income[train_idx]
test_df$income <- imputations$Income[-train_idx]
```

Linear regression with "missing" income values imputed

```
mod <- lm(total_claim_amount ~ vehicle_class + income +
             employment_status*monthly_premium_auto +
             location_code*monthly_premium_auto, data = train_df)

y_hat <- predict(mod, newdata = test_df)

summary(mod)
```

```
##
## Call:
## lm(formula = total_claim_amount ~ vehicle_class + income + employment_status *
##     monthly_premium_auto + location_code * monthly_premium_auto,
##     data = train_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -541.14  -64.46  -25.83   57.05 1357.81
##
## Coefficients:
##                                                     Estimate Std. Error
## (Intercept)                                       -5.465e+01  2.503e+01
## vehicle_classLuxury Car                            3.350e+01  1.752e+01
## vehicle_classLuxury SUV                            2.560e+01  1.691e+01
## vehicle_classSports Car                           -6.376e+00  8.202e+00
## vehicle_classSUV                                   6.430e-01  5.864e+00
## vehicle_classTwo-Door Car                          2.852e+00  4.098e+00
## income                                            -1.255e-04  8.981e-05
## employment_statusEmployed                          7.356e+01  2.242e+01
## employment_statusMedical Leave                    -3.793e+01  3.221e+01
## employment_statusRetired                           1.210e+02  3.358e+01
## employment_statusUnemployed                       -7.692e+00  2.322e+01
## monthly_premium_auto                               1.832e+00  2.570e-01
## location_codeSuburban                              9.536e+00  1.293e+01
## location_codeUrban                                 1.093e+01  1.616e+01
## employment_statusEmployed:monthly_premium_auto    -7.923e-01  2.209e-01
## employment_statusMedical Leave:monthly_premium_auto  7.672e-01  3.312e-01
## employment_statusRetired:monthly_premium_auto     -1.459e+00  3.407e-01
## employment_statusUnemployed:monthly_premium_auto   1.157e+00  2.319e-01
## monthly_premium_auto:location_codeSuburban         4.174e+00  1.339e-01
```

```
## monthly_premium_auto:location_codeUrban              2.322e+00  1.703e-01
##                                                     t value Pr(>|t|)
## (Intercept)                                          -2.183 0.029065 *
## vehicle_classLuxury Car                               1.912 0.055880 .
## vehicle_classLuxury SUV                               1.513 0.130207
## vehicle_classSports Car                              -0.777 0.436984
## vehicle_classSUV                                      0.110 0.912690
## vehicle_classTwo-Door Car                             0.696 0.486475
## income                                               -1.398 0.162250
## employment_statusEmployed                             3.281 0.001041 **
## employment_statusMedical Leave                       -1.178 0.238935
## employment_statusRetired                              3.602 0.000318 ***
## employment_statusUnemployed                          -0.331 0.740481
## monthly_premium_auto                                  7.126 1.15e-12 ***
## location_codeSuburban                                 0.737 0.460942
## location_codeUrban                                    0.676 0.498785
## employment_statusEmployed:monthly_premium_auto       -3.587 0.000337 ***
## employment_statusMedical Leave:monthly_premium_auto   2.316 0.020570 *
## employment_statusRetired:monthly_premium_auto        -4.283 1.87e-05 ***
## employment_statusUnemployed:monthly_premium_auto      4.989 6.25e-07 ***
## monthly_premium_auto:location_codeSuburban           31.182  < 2e-16 ***
## monthly_premium_auto:location_codeUrban              13.633  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 123.4 on 6206 degrees of freedom
## Multiple R-squared:  0.8159, Adjusted R-squared:  0.8153
## F-statistic:  1447 on 19 and 6206 DF,  p-value: < 2.2e-16
```

```r
mae <- mean_abs_err(y_hat, test_df$total_claim_amount)
print(paste("MAE of:", round(mae, 2)))
```

```
## [1] "MAE of: 89.55"
```

Doesn't seem to make much difference.

GAM

Does using a GAM improve performance? Again using the data with imputed income (even though it probably doesn't make much difference).

```r
mod <- gam(total_claim_amount ~ employment_status + location_code +
             vehicle_class + s(income) +
             s(monthly_premium_auto, by = employment_status) +
             s(monthly_premium_auto, by = location_code), data = train_df)

y_hat <- predict(mod, newdata = test_df)
summary(mod)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## total_claim_amount ~ employment_status + location_code + vehicle_class +
##     s(income) + s(monthly_premium_auto, by = employment_status) +
```

```
##     s(monthly_premium_auto, by = location_code)
##
## Parametric coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   1.152e+02  9.195e+00  12.529  < 2e-16 ***
## employment_statusEmployed     2.937e-01  9.318e+00   0.032    0.975
## employment_statusMedical Leave -1.114e+03  1.946e+02  -5.724 1.09e-08 ***
## employment_statusRetired     -1.533e+01  1.170e+01  -1.310    0.190
## employment_statusUnemployed   1.014e+02  8.091e+00  12.532  < 2e-16 ***
## location_codeSuburban         3.953e+02  4.391e+00  90.021  < 2e-16 ***
## location_codeUrban            2.258e+02  5.106e+00  44.219  < 2e-16 ***
## vehicle_classLuxury Car      -5.028e-02  2.113e+01  -0.002    0.998
## vehicle_classLuxury SUV      -2.738e+01  2.125e+01  -1.288    0.198
## vehicle_classSports Car      -8.804e+00  8.591e+00  -1.025    0.305
## vehicle_classSUV             -2.575e-02  6.383e+00  -0.004    0.997
## vehicle_classTwo-Door Car     2.757e+00  4.033e+00   0.684    0.494
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                                                      edf Ref.df      F
## s(income)                                         7.4640 8.4172  3.205
## s(monthly_premium_auto):employment_statusDisabled      5.8737 6.6539  5.855
## s(monthly_premium_auto):employment_statusEmployed      0.8751 0.8752 30.943
## s(monthly_premium_auto):employment_statusMedical Leave 8.8384 8.8738 13.373
## s(monthly_premium_auto):employment_statusRetired       0.8750 0.8751 30.902
## s(monthly_premium_auto):employment_statusUnemployed    8.7841 8.8698  9.449
## s(monthly_premium_auto):location_codeRural             8.6420 8.8407  5.320
## s(monthly_premium_auto):location_codeSuburban          3.9079 4.7932  6.517
## s(monthly_premium_auto):location_codeUrban             0.8754 0.8757 30.681
##                                                       p-value
## s(income)                                             0.00127 **
## s(monthly_premium_auto):employment_statusDisabled      2.00e-06 ***
## s(monthly_premium_auto):employment_statusEmployed      8.17e-07 ***
## s(monthly_premium_auto):employment_statusMedical Leave  < 2e-16 ***
## s(monthly_premium_auto):employment_statusRetired       8.38e-07 ***
## s(monthly_premium_auto):employment_statusUnemployed     < 2e-16 ***
## s(monthly_premium_auto):location_codeRural             1.20e-07 ***
## s(monthly_premium_auto):location_codeSuburban          1.18e-05 ***
## s(monthly_premium_auto):location_codeUrban             9.41e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 92/93
## R-sq.(adj) =  0.822   Deviance explained = 82.3%
## GCV =  14856  Scale est. = 14717     n = 6226
```

```r
mae <- mean_abs_err(y_hat, test_df$total_claim_amount)
print(paste("MAE of:", round(mae, 2)))
```

```
## [1] "MAE of: 91.34"
```

Default GAM does worse during cross-validation! Not too surprising when you see the plots though as they look very linear. Although I could also be doing something wrong…

Quantile regression

Since we are aiming the minimize the mean absolute error. Perhaps fitting a linear model using median regression will likely perform better. This is optimal for minimizing mean absolute error, whereas the mean is optimal for the squared error loss function.

```
mod <- rq(total_claim_amount ~ vehicle_class + income +
     employment_status*monthly_premium_auto +
     location_code*monthly_premium_auto, data = train_df)

y_hat <- predict(mod, newdata = test_df)

summary.rq(mod, se = "boot")  # bootstrap se estimates
```

```
##
## Call: rq(formula = total_claim_amount ~ vehicle_class + income + employment_status *
##     monthly_premium_auto + location_code * monthly_premium_auto,
##     data = train_df)
##
## tau: [1] 0.5
##
## Coefficients:
##                                                        Value      Std. Error
## (Intercept)                                               8.84868   35.06114
## vehicle_classLuxury Car                                   0.00000    8.11257
## vehicle_classLuxury SUV                                   0.00000    5.18898
## vehicle_classSports Car                                   0.00000    1.22768
## vehicle_classSUV                                          0.00000    0.00000
## vehicle_classTwo-Door Car                                 0.15345    2.89356
## income                                                    0.00000    0.00000
## employment_statusEmployed                                 6.54929   30.71381
## employment_statusMedical Leave                            6.54929   44.37370
## employment_statusRetired                                  6.54929   30.71381
## employment_statusUnemployed                            -130.86191   35.55818
## monthly_premium_auto                                      1.08339    0.52058
## location_codeSuburban                                   -15.39797   15.59786
## location_codeUrban                                       -2.98427   21.69061
## employment_statusEmployed:monthly_premium_auto           -0.10396    0.48050
## employment_statusMedical Leave:monthly_premium_auto      -0.10396    0.69153
## employment_statusRetired:monthly_premium_auto            -0.10396    0.48050
## employment_statusUnemployed:monthly_premium_auto          2.96634    0.49763
## monthly_premium_auto:location_codeSuburban                3.82057    0.18314
## monthly_premium_auto:location_codeUrban                   2.52061    0.26221
##                                                        t value    Pr(>|t|)
## (Intercept)                                               0.25238    0.80076
## vehicle_classLuxury Car                                   0.00000    1.00000
## vehicle_classLuxury SUV                                   0.00000    1.00000
## vehicle_classSports Car                                   0.00000    1.00000
## vehicle_classSUV                                          1.40577    0.15984
## vehicle_classTwo-Door Car                                 0.05303    0.95771
## income                                                   -3.20966    0.00134
## employment_statusEmployed                                 0.21324    0.83115
## employment_statusMedical Leave                            0.14759    0.88267
## employment_statusRetired                                  0.21324    0.83115
## employment_statusUnemployed                              -3.68022    0.00024
```

```
## monthly_premium_auto                                   2.08112    0.03746
## location_codeSuburban                                 -0.98718    0.32359
## location_codeUrban                                    -0.13758    0.89057
## employment_statusEmployed:monthly_premium_auto        -0.21635    0.82872
## employment_statusMedical Leave:monthly_premium_auto   -0.15033    0.88051
## employment_statusRetired:monthly_premium_auto         -0.21635    0.82872
## employment_statusUnemployed:monthly_premium_auto       5.96094    0.00000
## monthly_premium_auto:location_codeSuburban            20.86150    0.00000
## monthly_premium_auto:location_codeUrban                9.61313    0.00000
```

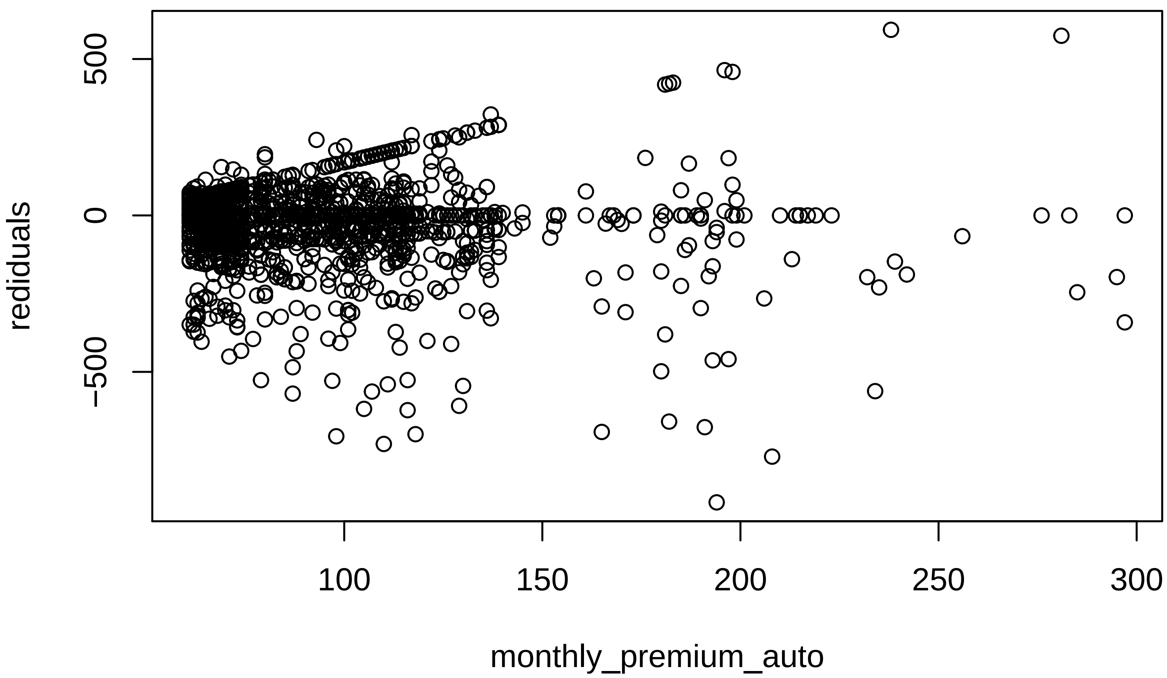
```r
mae <- mean_abs_err(y_hat, test_df$total_claim_amount)
print(paste("MAE of:", round(mae, 2)))
```

```
## [1] "MAE of: 79.4"
```

This performs a lot better. We should be able to simplify things further based on the zero coefficients above.

Note that here we use the pairwise bootstrap estimate of the standard errors. This is more robust to heteroscadasticity than the standard method (e.g. see this paper). We can see that we have heteroscadasticity here:

```r
plot(test_df$monthly_premium_auto, (y_hat-test_df$total_claim_amount),
     xlab = "monthly_premium_auto", ylab = "rediduals")
```



We will slim things down and simplify the model to make a final model:

Final Model

Simplify using the following:

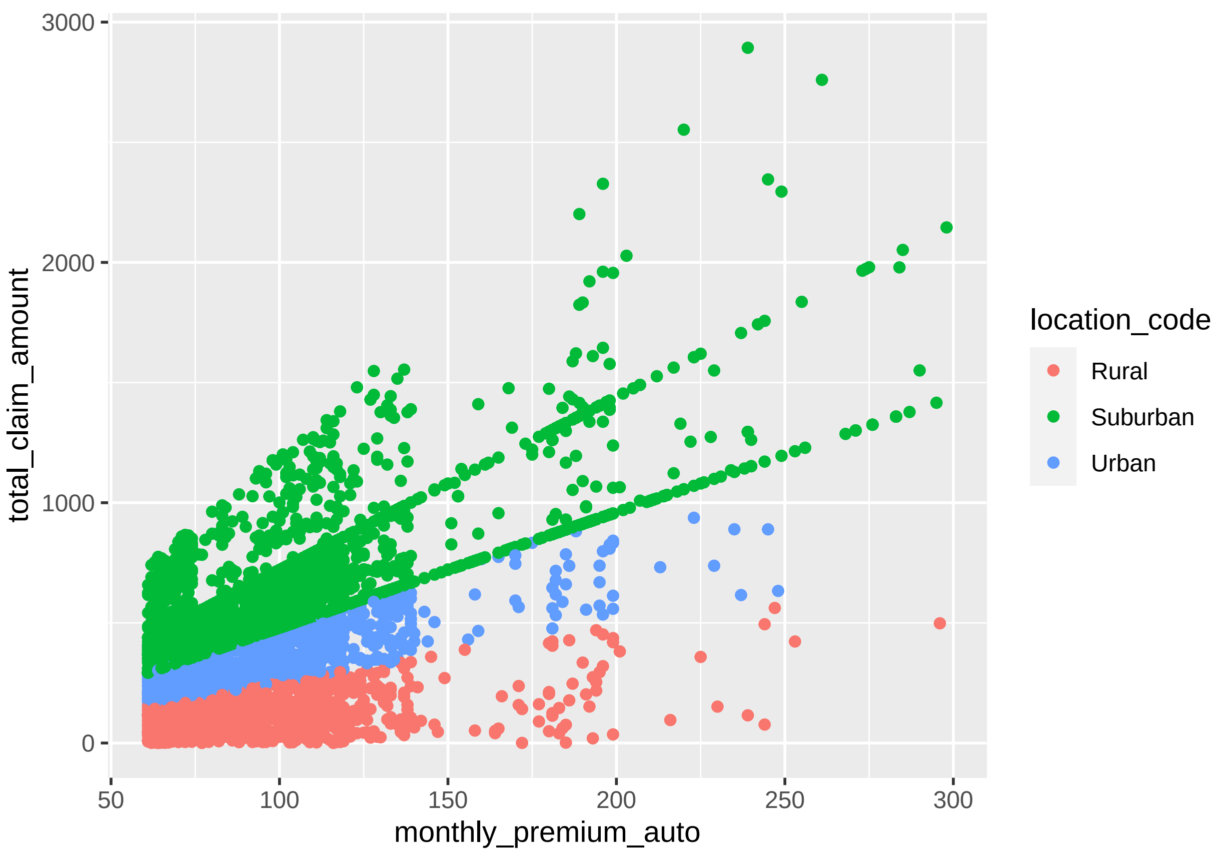
- Simplify `employment_status` into a boolean `is_employed`, as being unemployed was the only significant factor level.
- Drop income and vehicle class

```r
train_df_simple <- train_df %>%
  mutate(is_unemployed = employment_status == "Unemployed") %>%
  select(-employment_status, -income, -vehicle_class)

test_df_simple <- test_df %>%
  mutate(is_unemployed = employment_status == "Unemployed") %>%
  select(-employment_status, -income, -vehicle_class)

head(train_df_simple)
```

```
##      location_code monthly_premium_auto total_claim_amount is_unemployed
## 1352         Urban                   67           172.5185         FALSE
## 7760      Suburban                  129           928.8000          TRUE
## 4393      Suburban                  112          1169.4098         FALSE
## 2182      Suburban                   66           344.2016         FALSE
## 2219      Suburban                   65           468.0000         FALSE
## 1966         Rural                   65           117.4504         FALSE
```

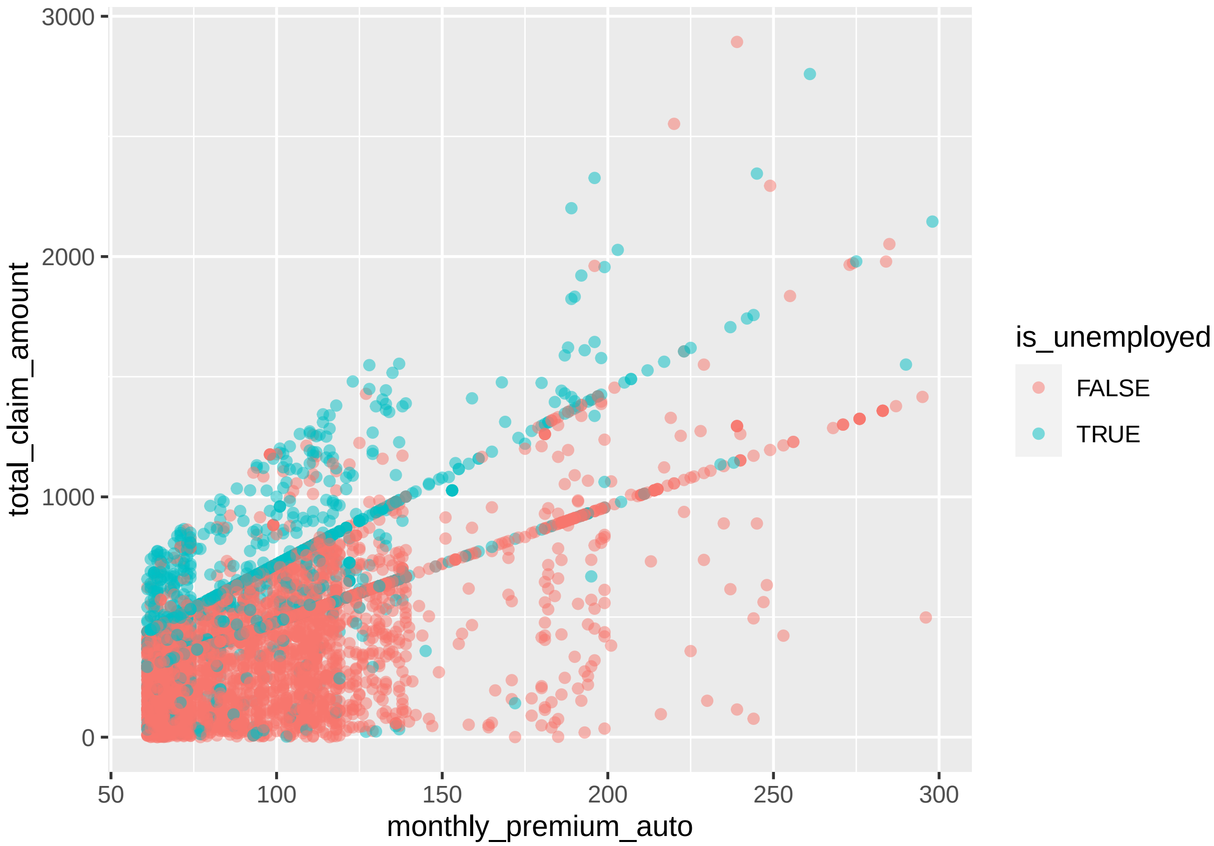
Note that these are primary variables identified in the EDA. We observed that the factor variables `is_unemployed` and `location_code` show a strong interaction with the `monthly_premium_auto` variable:

```r
train_df %>%
  ggplot(aes(x = monthly_premium_auto, y = total_claim_amount,
             col = location_code)) +
  geom_point()
```




```r
train_df_simple %>%
  ggplot(aes(x = monthly_premium_auto, y = total_claim_amount,
```

```r
             colour = is_unemployed)) +
  geom_point(alpha = 0.5)
```



The final, very simple model, using just three features is below:

```r
mod <- rq(total_claim_amount ~
            is_unemployed*monthly_premium_auto +
            location_code*monthly_premium_auto, data = train_df_simple)

y_hat <- predict(mod, test_df_simple)

summary.rq(mod, se = "boot")
```

```
##
## Call: rq(formula = total_claim_amount ~ is_unemployed * monthly_premium_auto +
##     location_code * monthly_premium_auto, data = train_df_simple)
##
## tau: [1] 0.5
##
## Coefficients:
##                                        Value      Std. Error t value
## (Intercept)                              15.58546   14.63297    1.06509
## is_unemployedTRUE                      -137.59246   19.33962   -7.11454
## monthly_premium_auto                      0.97845    0.17734    5.51743
## location_codeSuburban                   -15.58546   14.63297   -1.06509
## location_codeUrban                       -2.97803   20.97822   -0.14196
## is_unemployedTRUE:monthly_premium_auto    3.07118    0.11322   27.12592
```

```
## monthly_premium_auto:location_codeSuburban    3.82155    0.17734   21.54963
## monthly_premium_auto:location_codeUrban       2.52053    0.25301    9.96209
##                                             Pr(>|t|)
## (Intercept)                                    0.28688
## is_unemployedTRUE                              0.00000
## monthly_premium_auto                           0.00000
## location_codeSuburban                          0.28688
## location_codeUrban                             0.88712
## is_unemployedTRUE:monthly_premium_auto         0.00000
## monthly_premium_auto:location_codeSuburban     0.00000
## monthly_premium_auto:location_codeUrban        0.00000
```

```r
mae <- mean_abs_err(y_hat, test_df$total_claim_amount)
print(paste("MAE of:", round(mae, 2)))
```

```
## [1] "MAE of: 79.36"
```

Now we have the best model. I will retrain it on the whole data set. Then this can be used to predict on the test set when it is released:

```r
df <- rbind(train_df_simple, test_df_simple)

mod <- rq(total_claim_amount ~
            is_unemployed*monthly_premium_auto +
            location_code*monthly_premium_auto, data = df)

# Put test data here...
```