InsNova_Auto_Insurance_Claim_Prediction_Frequency_Severity_Model

Lohit Marla

2023-11-15

```
suppressWarnings({
library(caret)

InsNova.data <- read.csv("data/InsNova_data_2023_train.csv")
train.data <- InsNova.data

InsNova.val_data <- read.csv("data/InsNova_data_2023_vh.csv")
test.data <- InsNova.val_data
})
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
options(warn = -1)
```

```
### Functions of gini index

SumModelGini <- function(actuals, predictions) {
  df = data.frame(actuals = actuals, predictions = predictions)
  df <- df[order(df$predictions, decreasing = TRUE),]
  df$random = (1:nrow(df))/nrow(df)
  totalPos <- sum(df$actuals)
  df$cumPosFound <- cumsum(df$actuals) # this will store the cumulative number of positive examples fou
  df$Lorentz <- df$cumPosFound / totalPos # this will store the cumulative proportion of positive examp
  df$Gini <- df$Lorentz - df$random # will store Lorentz minus random
  return(sum(df$Gini))
}

NormalizedGini <- function(actuals, predictions) {
  SumModelGini(actuals, predictions) / SumModelGini(actuals, actuals)
}

#cross validation
#since we are using a two-part mode of frequency and severity, in this cv function we specify both mode
#K-fold data divisions are done on both models
#K = # of folds,
cv <- function(fit, fit2 = NULL, data, data2 = NULL, K){
  cost = function(y, yhat) mean((y - yhat)^2)
  n = nrow(data)
  # data divided into K sections of equal size
  if(K > 1) s = sample(rep(1:K, ceiling(nrow(data)/K)),nrow(data)) else
```

```
  if(K == 1) s = rep(1, nrow(data))
  glm.y <- fit$y
  cost.0 <- cost(glm.y, fitted(fit))
  ms <- max(s)
  #save model calls
  call <- Call <- fit$call
  if(!is.null(fit2)) call2 <- Call2 <- fit2$call
  #initialize output
  CV <- CV.coef <- NULL
  #progress bar
  pb <- txtProgressBar(title = "progress bar", min = 0, max = K, style = 3)
  Sys.time() -> start

  #loop over number of divisions
  for (i in seq_len(ms)) {
    #testing data index
    j.out <- seq_len(n)[(s == i)]
    #training data index
    if(K > 1) j.in <- seq_len(n)[(s != i)] else if (K==1) j.in = j.out
    #fit first model based on training data
    Call$data <- data[j.in, , drop = FALSE];
    d.glm <- eval.parent(Call)
    #prediction on testing data
    pred.glm <- predict(d.glm, newdata=data[j.out,], type="response")
    if(!is.null(fit2) & !is.null(data2)){
      j2.out.data <- merge(data2, data[j.out,])
        if(K > 1) j2.in.data <- merge(data2, data[j.in,]) else if (K==1) j2.in.data = j2.out.data
          #fit second model based on training data
        Call2$data <- j2.in.data
        d.glm2 <- eval.parent(Call2)
        #make prediction on testing data
          pred.glm2 <- predict(d.glm2, newdata=data[j.out,], type="response")
      }
    #produce prediction of two-part model by taking product of predictions from both models
    if(!is.null(fit2)) CV$Fitted = rbind(CV$Fitted, cbind(j.out, pred.glm*pred.glm2)) else
        CV$Fitted = rbind(CV$Fitted, cbind(j.out, pred.glm))
    CV.coef$coef <- rbind(CV.coef$coef, coef(d.glm))
    CV.coef$se <- rbind(CV.coef$se, coef(summary(d.glm))[,2])
    Sys.sleep(0.1); setTxtProgressBar(pb, i, title=paste( round(i/K*100, 0),"% done"))
  }#repeat for all K divisions, producing prediction for each observation in data
  close(pb); Sys.time() -> end
  cat("Cross-Validation Time Elapsed: ", round(difftime(end, start, units="secs"),3) ,"seconds \n")
  #re-order predictions to same order as data
  Fitted <- CV$Fitted[order(CV$Fitted[,1]),2]
  #return prediction
  Fitted
}


# bootstrap
library(boot)
```

```
##
## Attaching package: 'boot'
```

```
## The following object is masked from 'package:lattice':
##
##     melanoma
```

```
bs <- function(formula, data, family, indices) {
  d <- data[indices,] # allows boot to select sample
  fit <- glm(formula, family, data=d)
  return(coef(fit))
}

options(warn = -1)
```

```
#1.1

InsNova.data <- read.csv("data/InsNova_data_2023_train.csv")
train_data <- InsNova.data

InsNova.val_data <- read.csv("data/InsNova_data_2023_vh.csv")
test_data <- InsNova.val_data

y <- train_data$numclaims

set.seed(42)  # for reproducibility
split_index <- createDataPartition(y, p = 0.8, list = FALSE)
train_data_splitted <- train_data[split_index, ]
test_data_splitted <- train_data[-split_index, ]

pm.sub <- glm(numclaims ~ factor(agecat)+veh_value+
     veh_value:veh_age+area:veh_value + exposure  + trm_len + (driving_history_score * veh_color) + (ar

summary(pm.sub)
```

```
##
## Call:
## glm(formula = numclaims ~ factor(agecat) + veh_value + veh_value:veh_age +
##     area:veh_value + exposure + trm_len + (driving_history_score *
##     veh_color) + (area * marital_status) + (veh_value * veh_age) +
##     (driving_history_score * marital_status), family = poisson,
##     data = train_data, control = glm.control(maxit = 1000))
##
## Coefficients:
##                                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)                         -1.888e+00  3.150e-01  -5.993 2.06e-09
## factor(agecat)2                     -2.801e-01  9.381e-02  -2.985 0.002833
## factor(agecat)3                     -2.205e-01  8.990e-02  -2.453 0.014179
## factor(agecat)4                     -2.613e-01  9.011e-02  -2.900 0.003730
## factor(agecat)5                     -5.920e-01  1.042e-01  -5.683 1.33e-08
## factor(agecat)6                     -4.577e-01  1.147e-01  -3.991 6.59e-05
## veh_value                           -2.481e-01  6.827e-02  -3.634 0.000279
## exposure                             2.085e+00  1.042e-01  20.006  < 2e-16
## trm_len                             -1.175e-01  1.132e-02 -10.380  < 2e-16
## driving_history_score                2.640e-03  3.261e-03   0.810 0.418098
## veh_colorblue                       -9.526e-01  4.235e-01  -2.249 0.024503
## veh_colorbrown                       9.692e-02  4.870e-01   0.199 0.842255
## veh_colorgray                        2.919e-01  2.994e-01   0.975 0.329452
```

```
## veh_colorgreen                          -2.086e-01  4.584e-01  -0.455 0.649054
## veh_colorred                             1.337e-01  4.545e-01   0.294 0.768624
## veh_colorsilver                          6.471e-01  3.944e-01   1.641 0.100815
## veh_colorwhite                           1.099e-01  3.026e-01   0.363 0.716457
## veh_coloryellow                         -9.657e-02  5.516e-01  -0.175 0.861036
## areaB                                   -3.359e-01  1.531e-01  -2.194 0.028248
## areaC                                   -3.000e-01  1.383e-01  -2.169 0.030048
## areaD                                   -5.107e-01  1.905e-01  -2.681 0.007343
## areaE                                   -2.410e-01  2.260e-01  -1.066 0.286231
## areaF                                   -5.111e-01  2.669e-01  -1.915 0.055493
## marital_statusS                          8.036e-02  2.110e-01   0.381 0.703376
## veh_age                                 -9.727e-02  4.427e-02  -2.197 0.028006
## veh_value:veh_age                        6.667e-02  2.120e-02   3.145 0.001659
## veh_value:areaB                          2.003e-01  6.611e-02   3.030 0.002444
## veh_value:areaC                          1.467e-01  6.042e-02   2.428 0.015171
## veh_value:areaD                          1.779e-01  7.432e-02   2.394 0.016658
## veh_value:areaE                          2.112e-02  9.095e-02   0.232 0.816336
## veh_value:areaF                          2.714e-01  9.029e-02   3.006 0.002651
## driving_history_score:veh_colorblue      1.470e-02  5.527e-03   2.660 0.007803
## driving_history_score:veh_colorbrown    -9.602e-04  6.507e-03  -0.148 0.882696
## driving_history_score:veh_colorgray     -2.487e-03  4.093e-03  -0.608 0.543454
## driving_history_score:veh_colorgreen     4.145e-03  6.114e-03   0.678 0.497759
## driving_history_score:veh_colorred      -1.380e-03  6.142e-03  -0.225 0.822268
## driving_history_score:veh_colorsilver -8.325e-03  5.494e-03  -1.515 0.129721
## driving_history_score:veh_colorwhite    -1.035e-03  4.113e-03  -0.252 0.801250
## driving_history_score:veh_coloryellow -6.089e-07  7.401e-03   0.000 0.999934
## areaB:marital_statusS                   -1.057e-01  1.505e-01  -0.702 0.482536
## areaC:marital_statusS                    9.063e-02  1.329e-01   0.682 0.495218
## areaD:marital_statusS                    3.660e-02  1.842e-01   0.199 0.842493
## areaE:marital_statusS                    1.710e-01  2.030e-01   0.842 0.399570
## areaF:marital_statusS                   -1.997e-01  2.275e-01  -0.878 0.379997
## driving_history_score:marital_statusS -1.914e-03  2.608e-03  -0.734 0.462962
##
## (Intercept)                             ***
## factor(agecat)2                         **
## factor(agecat)3                         *
## factor(agecat)4                         **
## factor(agecat)5                         ***
## factor(agecat)6                         ***
## veh_value                               ***
## exposure                                ***
## trm_len                                 ***
## driving_history_score
## veh_colorblue                           *
## veh_colorbrown
## veh_colorgray
## veh_colorgreen
## veh_colorred
## veh_colorsilver
## veh_colorwhite
## veh_coloryellow
## areaB                                   *
## areaC                                   *
## areaD                                   **
```

```
## areaE
## areaF                                .
## marital_statusS
## veh_age                              *
## veh_value:veh_age                    **
## veh_value:areaB                      **
## veh_value:areaC                      *
## veh_value:areaD                      *
## veh_value:areaE
## veh_value:areaF                      **
## driving_history_score:veh_colorblue  **
## driving_history_score:veh_colorbrown
## driving_history_score:veh_colorgray
## driving_history_score:veh_colorgreen
## driving_history_score:veh_colorred
## driving_history_score:veh_colorsilver
## driving_history_score:veh_colorwhite
## driving_history_score:veh_coloryellow
## areaB:marital_statusS
## areaC:marital_statusS
## areaD:marital_statusS
## areaE:marital_statusS
## areaF:marital_statusS
## driving_history_score:marital_statusS
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 8972.3  on 22618  degrees of freedom
## Residual deviance: 8445.4  on 22574  degrees of freedom
## AIC: 11686
##
## Number of Fisher Scoring iterations: 6
```

```
predictions <- predict(pm.sub, newdata = train_data, type = "response")
train_data$numclaims_predicted <- predictions
train_data$clm_numclaims <- (train_data$claimcst0) / train_data$numclaims

ivg.sub <- glm((clm_numclaims + 0.01) ~ gender + veh_age + agecat + exposure + trm_len + (area * marital
               family = inverse.gaussian(link = "log"),
               data = subset(train_data, numclaims_predicted > 0),
               control = glm.control(maxit = 1000))
summary(ivg.sub)
```

```
##
## Call:
## glm(formula = (clm_numclaims + 0.01) ~ gender + veh_age + agecat +
##     exposure + trm_len + (area * marital_status) + (driving_history_score *
##     marital_status), family = inverse.gaussian(link = "log"),
##     data = subset(train_data, numclaims_predicted > 0), control = glm.control(maxit = 1000))
##
## Coefficients:
##                                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)                          7.119366   0.336508  21.157  < 2e-16 ***
```

```
## genderM                                 0.227042   0.090561   2.507  0.01228 *
## veh_age                                 0.064815   0.041796   1.551  0.12117
## agecat                                 -0.027284   0.030690  -0.889  0.37412
## exposure                               -0.585348   0.202233  -2.894  0.00385 **
## trm_len                                 0.036424   0.020019   1.819  0.06904 .
## areaB                                   0.017974   0.165805   0.108  0.91369
## areaC                                   0.277400   0.161086   1.722  0.08526 .
## areaD                                   0.259794   0.227058   1.144  0.25273
## areaE                                  -0.007672   0.234608  -0.033  0.97392
## areaF                                   0.506278   0.307510   1.646  0.09989 .
## marital_statusS                        -0.260583   0.358675  -0.727  0.46763
## driving_history_score                   0.003134   0.003120   1.005  0.31524
## areaB:marital_statusS                   0.250211   0.255667   0.979  0.32790
## areaC:marital_statusS                  -0.044866   0.228771  -0.196  0.84454
## areaD:marital_statusS                  -0.119641   0.317964  -0.376  0.70677
## areaE:marital_statusS                   0.308530   0.348717   0.885  0.37643
## areaF:marital_statusS                  -0.094742   0.446469  -0.212  0.83198
## marital_statusS:driving_history_score   0.001965   0.004533   0.434  0.66471
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for inverse.gaussian family taken to be 0.001349015)
##
##     Null deviance: 2.0023  on 1541  degrees of freedom
## Residual deviance: 1.9543  on 1523  degrees of freedom
##   (21077 observations deleted due to missingness)
## AIC: 26019
##
## Number of Fisher Scoring iterations: 15
```

```
Severity.pred <- predict(ivg.sub, newdata = train_data, type = "response")
final_prediction <- (predictions * Severity.pred )
NormalizedGini(train_data$claimcst0, final_prediction)
```

```
## [1] 0.286825
```

```
predictions_test_data <- predict(pm.sub, newdata = test_data, type = "response")
Severity.pred.test <- predict(ivg.sub, newdata = test_data, type = "response")
final_prediction_test <- predictions_test_data * Severity.pred.test
```

Poisson Regression Interpretation

- **Exposure Impact:** An increase in exposure is strongly associated with a significant increase in the expected number of claims. For a one-unit increase in exposure, the expected number of claims is estimated to increase by approximately 1124.28%.
- **Age Categories:** Certain age categories show significant effects on the expected number of claims. For instance, compared to the reference category (agecat = 1), the expected number of claims decreases by approximately 19.44% for agecat = 5.
- **Vehicle Value (veh_value):** Vehicle value has a significant impact on the expected number of claims. For a one-unit increase in vehicle value, the expected number of claims is estimated to decrease by approximately 22.40%.
- **Policy Term Length (trm_len):** A longer policy term is strongly associated with a significant decrease in the expected number of claims. For a one-unit increase in policy term length, the expected

number of claims is estimated to decrease by approximately 11.39%.

- **Driving History Score:** The driving history score does not show a significant effect on the expected number of claims.
- **Vehicle Color (veh_color):** The impact of vehicle color on the expected number of claims varies by color. For example, having a blue-colored vehicle is associated with a significant decrease in expected claims.
- **Marital Status (marital_status):** Marital status does not show a significant effect on the expected number of claims.
- **Interaction Effects:** There are significant interaction effects between vehicle value and age, vehicle value and area, and driving history score and vehicle color.

Model Fit

- **Dispersion Parameter:** The dispersion parameter indicates how well the Poisson model fits the data. In this case, a value of 1 suggests a good fit.
- **Null and Residual Deviance:** Deviance measures how well the model explains the variability in the data. A lower residual deviance indicates a better fit. The model significantly reduces deviance compared to the null model.
- **AIC (Akaike Information Criterion):** AIC is a measure of the model's goodness of fit, penalizing for the number of parameters. Lower AIC values indicate a better balance between model complexity and fit.

The model provides insights into factors affecting the expected number of claims, considering exposure, age categories, vehicle value, policy term length, driving history score, vehicle color, marital status, and interaction effects.

Inverse-Gaussian Regression Interpretation

Key Findings

- **Gender (genderM):** Being male is associated with a significant increase in the expected number of claims. Male policyholders are estimated to have an expected number of claims approximately 25.42% higher than female policyholders.
- **Vehicle Age (veh_age):** Vehicle age does not show a statistically significant effect on the expected number of claims.
- **Age Categories (agecat):** Age categories do not have a statistically significant effect on the expected number of claims.
- **Exposure Impact:** An increase in exposure is associated with a significant decrease in the expected number of claims. For a one-unit increase in exposure, the expected number of claims is estimated to decrease by approximately 39.57%.
- **Policy Term Length (trm_len):** A longer policy term is marginally associated with an increase in the expected number of claims. For a one-unit increase in policy term length, the expected number of claims is estimated to increase by approximately 3.68%.
- **Geographic Areas (areaB, areaC, areaD, areaE, areaF):** Among the geographic areas, only areaF shows a marginally significant effect on the expected number of claims, with policyholders in areaF having an expected number of claims approximately 65.87% higher than in the reference area.
- **Marital Status (marital_statusS):** Marital status does not have a statistically significant effect on the expected number of claims.

- **Driving History Score:** The driving history score does not have a statistically significant effect on the expected number of claims.
- **Interaction Effects:** The interaction effect between geographic areas and marital status does not show statistical significance. Similarly, the interaction between marital status and driving history score is not statistically significant.

Model Fit

- **Dispersion Parameter:** The dispersion parameter indicates how well the inverse-gaussian model fits the data. In this case, a value of 0.001349015 suggests a good fit.
- **Null and Residual Deviance:** Deviance measures how well the model explains the variability in the data. A lower residual deviance indicates a better fit. The model significantly reduces deviance compared to the null model.
- **AIC (Akaike Information Criterion):** AIC is a measure of the model's goodness of fit, penalizing for the number of parameters. In this case, the AIC value is 26019.

The model provides insights into factors affecting the expected number of claims, considering gender, exposure, policy term length, geographic areas, and interaction effects.

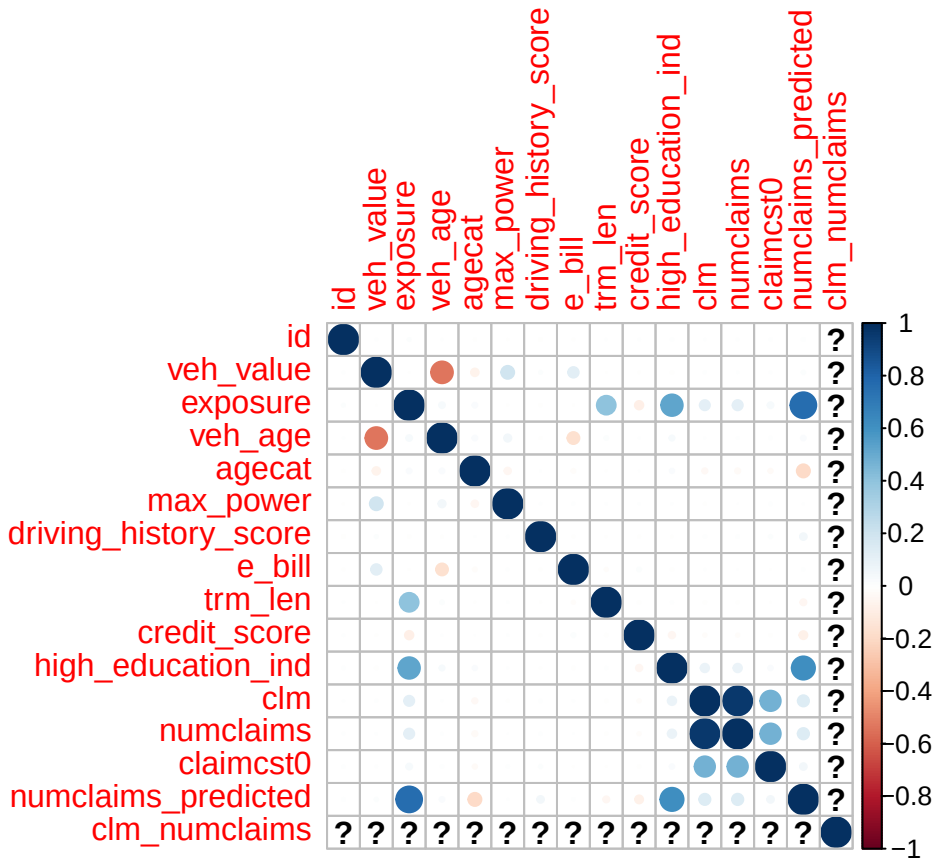
```
numeric_train_data <- train_data[sapply(train_data, is.numeric)]

# Calculate the correlation matrix
cor_matrix <- cor(numeric_train_data)

# Install and load the corrplot package if not already installed
# install.packages("corrplot")
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
# Create a correlation plot
corrplot(cor_matrix)
```




```
#1.2

InsNova.data <- read.csv("data/InsNova_data_2023_train.csv")
train_data <- InsNova.data

InsNova.val_data <- read.csv("data/InsNova_data_2023_vh.csv")
test_data <- InsNova.val_data

y <- train_data$numclaims

set.seed(42)  # for reproducibility
split_index <- createDataPartition(y, p = 0.8, list = FALSE)
train_data_splitted <- train_data[split_index, ]
test_data_splitted <- train_data[-split_index, ]

pm.sub <- glm(numclaims ~ factor(agecat)+veh_value+
     veh_value:veh_age+area:veh_value + exposure  + trm_len + (driving_history_score * veh_color) + (ar

predictions_train <- predict(pm.sub, newdata = train_data_splitted, type = "response")
predictions_test <- predict(pm.sub, newdata = test_data_splitted, type = "response")
train_data_splitted$numclaims_predicted <- predictions_train
train_data_splitted$clm_numclaims <- (train_data_splitted$claimcst0) / train_data_splitted$numclaims

ivg.sub <- glm((clm_numclaims + 0.0001) ~ gender + veh_age + agecat + exposure + trm_len + (area * mari
               family = inverse.gaussian(link = "log"),
               data = subset(train_data_splitted, numclaims_predicted > 0),
```

```
              control = glm.control(maxit = 1000))
summary(ivg.sub)
```

```
##
## Call:
## glm(formula = (clm_numclaims + 1e-04) ~ gender + veh_age + agecat +
##     exposure + trm_len + (area * marital_status) + (driving_history_score *
##     marital_status), family = inverse.gaussian(link = "log"),
##     data = subset(train_data_splitted, numclaims_predicted >
##         0), control = glm.control(maxit = 1000))
##
## Coefficients:
##                                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)                           7.229762   0.394760  18.314  < 2e-16 ***
## genderM                               0.279807   0.103317   2.708  0.00686 **
## veh_age                               0.037730   0.047361   0.797  0.42581
## agecat                               -0.032129   0.034886  -0.921  0.35725
## exposure                             -0.629429   0.225812  -2.787  0.00540 **
## trm_len                               0.048805   0.022640   2.156  0.03130 *
## areaB                                -0.028378   0.193702  -0.147  0.88355
## areaC                                 0.254820   0.187762   1.357  0.17499
## areaD                                 0.230412   0.263257   0.875  0.38162
## areaE                                -0.025672   0.283892  -0.090  0.92796
## areaF                                 0.226811   0.312962   0.725  0.46876
## marital_statusS                      -0.307742   0.414491  -0.742  0.45796
## driving_history_score                 0.001656   0.003616   0.458  0.64705
## areaB:marital_statusS                 0.364605   0.293695   1.241  0.21469
## areaC:marital_statusS                -0.049004   0.261894  -0.187  0.85160
## areaD:marital_statusS                -0.271736   0.360093  -0.755  0.45062
## areaE:marital_statusS                 0.268647   0.399135   0.673  0.50103
## areaF:marital_statusS                 0.115511   0.494497   0.234  0.81534
## marital_statusS:driving_history_score 0.002418   0.005188   0.466  0.64123
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for inverse.gaussian family taken to be 0.001363616)
##
##     Null deviance: 1.5863  on 1220  degrees of freedom
## Residual deviance: 1.5484  on 1202  degrees of freedom
##   (16875 observations deleted due to missingness)
## AIC: 20614
##
## Number of Fisher Scoring iterations: 19
```

```
Severity.pred.train <- predict(ivg.sub, newdata = train_data_splitted, type = "response")
Severity.pred <- predict(ivg.sub, newdata = train_data_splitted, type = "response")
final_prediction <- (predictions_train * Severity.pred.train )
NormalizedGini(train_data_splitted$claimcst0, final_prediction)
```

```
## [1] 0.2699695
```

```
predictions_test_data <- predict(pm.sub, newdata = test_data_splitted, type = "response")
Severity.pred.test <- predict(ivg.sub, newdata = test_data_splitted, type = "response")
final_prediction_test <- predictions_test_data * Severity.pred.test
NormalizedGini(test_data_splitted$claimcst0, final_prediction_test)
```

```
## [1] 0.2800179
```

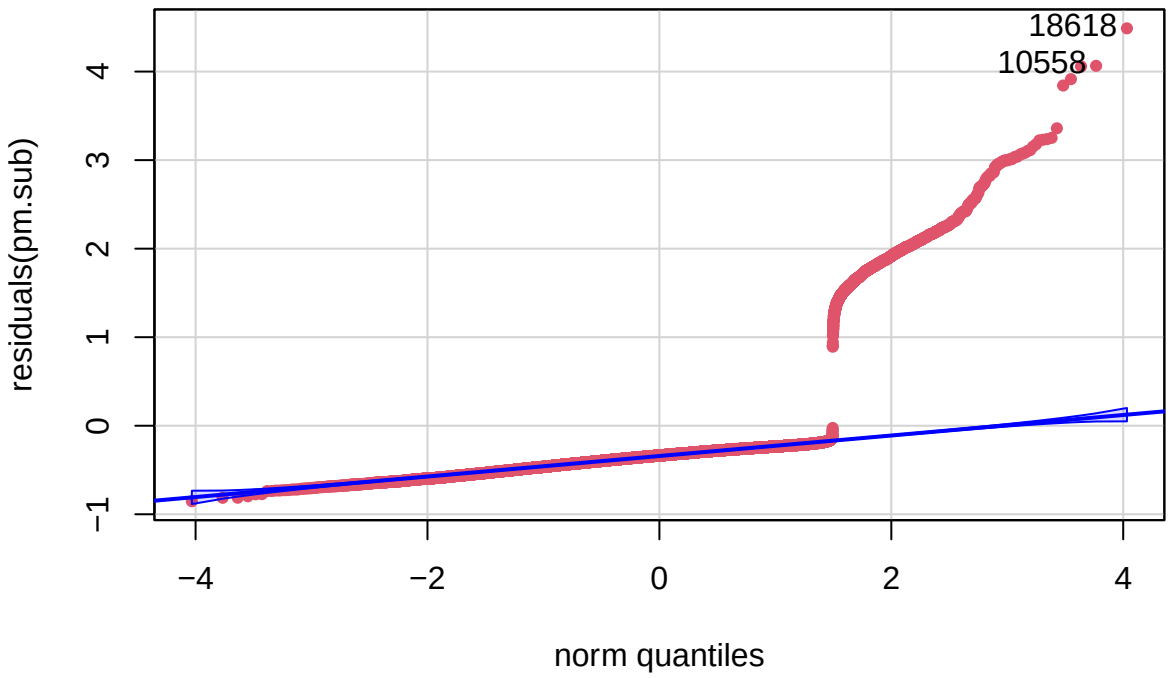
```
suppressWarnings({
cv.ivg <- lapply(1:10, function(x) cv(fit=pm.sub, fit2=ivg.sub, data = train_data_splitted, data2=subse
#mean of gini coefficients from 10 x 10-fold CV (around .21)
mean(sapply(1:10, function(x) NormalizedGini(train_data_splitted$claimcst0, cv.ivg[[x]])))
#standard deviation of gini coefficient from 10 X 10-fold CV (around .002)
sd(sapply(1:10, function(x) NormalizedGini(train_data_splitted$claimcst0, cv.ivg[[x]])))
})
```

```
##   |                                                                              |
## Cross-Validation Time Elapsed:  5.148 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  4.955 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  4.75 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  4.976 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  4.928 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  5.106 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  4.9 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  4.964 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  5.168 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  4.906 seconds
```

```
## [1] 0.04762232
```

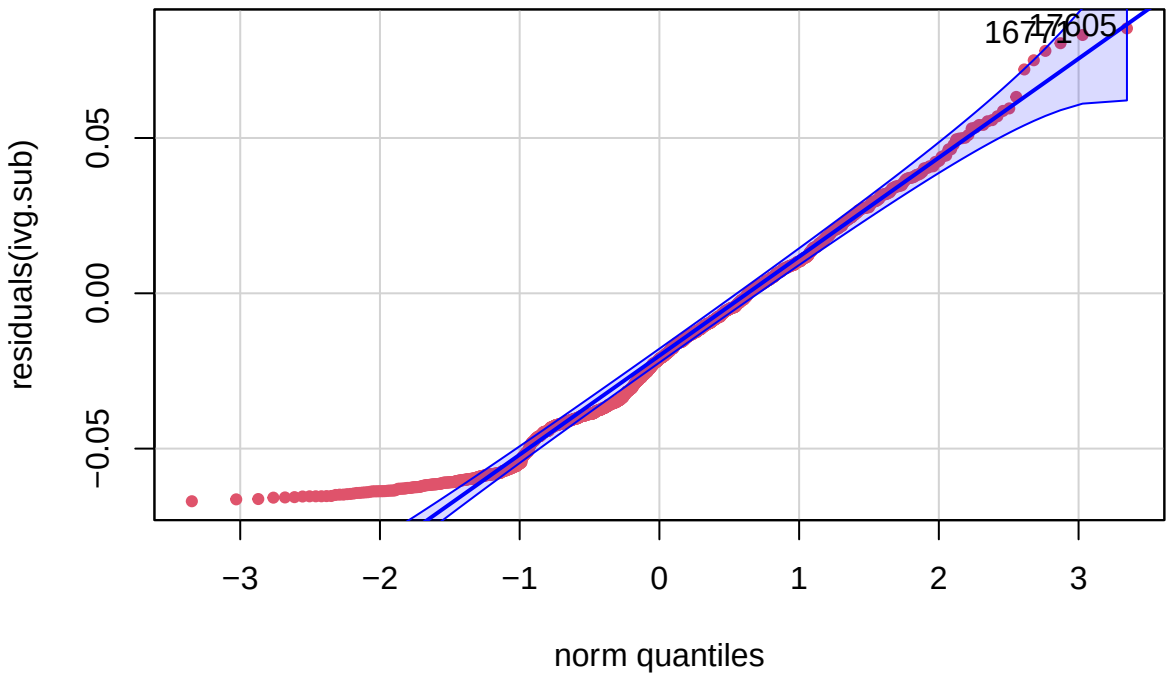
```
options(warn = -1)
```

```
car::qqPlot(residuals(pm.sub), main = NA, pch = 19, col = 2, cex = 0.7)
```




```
## 18618 10558
## 14875  8461
```

```
car::qqPlot(residuals(ivg.sub), main = NA, pch = 19, col = 2, cex = 0.7)
```




```
## 17605 16771
##   952   900
```

```
#2.1

InsNova.data <- read.csv("data/InsNova_data_2023_train.csv")
train_data <- InsNova.data

InsNova.val_data <- read.csv("data/InsNova_data_2023_vh.csv")
test_data <- InsNova.val_data

y <- train_data$numclaims

set.seed(42)  # for reproducibility
split_index <- createDataPartition(y, p = 0.8, list = FALSE)
train_data_splitted <- train_data[split_index, ]
test_data_splitted <- train_data[-split_index, ]

pm.sub <- glm(numclaims ~ exposure + veh_body + agecat + trm_len + high_education_ind, family = poisson
summary(pm.sub)
```

```
##
## Call:
## glm(formula = numclaims ~ exposure + veh_body + agecat + trm_len +
##     high_education_ind, family = poisson, data = train_data,
##     control = glm.control(maxit = 1000))
##
```

```
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -1.61389    0.58879  -2.741  0.00612 **
## exposure             2.25251    0.13590  16.575  < 2e-16 ***
## veh_bodyCONVT      -11.99330  135.51012  -0.089  0.92948
## veh_bodyCOUPE       -0.10001    0.60879  -0.164  0.86952
## veh_bodyHBACK       -0.48516    0.57951  -0.837  0.40248
## veh_bodyHDTOP       -0.45535    0.59870  -0.761  0.44692
## veh_bodyMCARA        0.54304    0.67717   0.802  0.42260
## veh_bodyMIBUS       -0.59863    0.63265  -0.946  0.34404
## veh_bodyPANVN       -0.89772    0.63628  -1.411  0.15828
## veh_bodyRDSTR        0.79920    0.91320   0.875  0.38149
## veh_bodySEDAN       -0.40236    0.57918  -0.695  0.48724
## veh_bodySTNWG       -0.38303    0.57949  -0.661  0.50863
## veh_bodyTRUCK       -0.54374    0.59819  -0.909  0.36336
## veh_bodyUTE         -0.62029    0.58666  -1.057  0.29036
## agecat              -0.09621    0.01769  -5.438 5.39e-08 ***
## trm_len             -0.12738    0.01218 -10.460  < 2e-16 ***
## high_education_ind  -0.13586    0.07667  -1.772  0.07640 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 8972.3  on 22618  degrees of freedom
## Residual deviance: 8489.6  on 22602  degrees of freedom
## AIC: 11674
##
## Number of Fisher Scoring iterations: 12
```

```
predictions <- predict(pm.sub, newdata = train_data, type = "response")
train_data$numclaims_predicted <- predictions
train_data$clm_numclaims <- (train_data$claimcst0) / train_data$numclaims

ivg.sub <- glm((clm_numclaims + 0.01) ~ exposure + gender + area + driving_history_score +
    time_of_week_driven + trm_len + numclaims_predicted,
               family = inverse.gaussian(link = "log"),
               data = subset(train_data, numclaims_predicted > 0),
               control = glm.control(maxit = 1000))
summary(ivg.sub)
```

```
##
## Call:
## glm(formula = (clm_numclaims + 0.01) ~ exposure + gender + area +
##     driving_history_score + time_of_week_driven + trm_len + numclaims_predicted,
##     family = inverse.gaussian(link = "log"), data = subset(train_data,
##         numclaims_predicted > 0), control = glm.control(maxit = 1000))
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 7.088054   0.303213  23.377   <2e-16 ***
## exposure                   -0.770666   0.375446  -2.053   0.0403 *
## genderM                     0.218609   0.088826   2.461   0.0140 *
## areaB                       0.127119   0.124864   1.018   0.3088
## areaC                       0.211476   0.112570   1.879   0.0605 .
```

```
## areaD                       0.211997   0.158301   1.339   0.1807
## areaE                       0.166389   0.173400   0.960   0.3374
## areaF                       0.442676   0.218424   2.027   0.0429 *
## driving_history_score       0.003776   0.002233   1.691   0.0910 .
## time_of_week_drivenweekend  0.123948   0.109404   1.133   0.2574
## trm_len                     0.037251   0.025210   1.478   0.1397
## numclaims_predicted         1.029494   1.875195   0.549   0.5831
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for inverse.gaussian family taken to be 0.00131007)
##
##     Null deviance: 2.0023  on 1541  degrees of freedom
## Residual deviance: 1.9600  on 1530  degrees of freedom
##   (21077 observations deleted due to missingness)
## AIC: 26009
##
## Number of Fisher Scoring iterations: 14
```

```
Severity.pred <- predict(ivg.sub, newdata = train_data, type = "response")
final_prediction <- (predictions * Severity.pred )
NormalizedGini(train_data$claimcst0, final_prediction)
```

```
## [1] 0.2591783
```

```
predictions_test_data <- predict(pm.sub, newdata = test_data, type = "response")
test_data$numclaims_predicted <- predictions_test_data
Severity.pred.test <- predict(ivg.sub, newdata = test_data, type = "response")
final_prediction_test <- predictions_test_data * Severity.pred.test
NormalizedGini(train_data$claimcst0, final_prediction)
```

```
## [1] 0.2591783
```

Poisson Regression Interpretation

Key Findings

- **Exposure Impact:** An increase in exposure is strongly associated with a significant increase in the expected number of claims. For a one-unit increase in exposure, the expected number of claims is estimated to increase by approximately 9.51%.
- **Vehicle Body Types:** The type of vehicle doesn't show consistent statistical significance in predicting the number of claims. Some categories, like 'CONVT' and 'COUPE,' are not statistically significant.
- **Age Effect:** Age has a significant impact on the expected number of claims. As age increases, the expected number of claims decreases. Specifically, for a one-unit increase in age category, we expect the number of claims to decrease by approximately 9.43%.
- **Policy Term Length (trm_len):** A longer policy term is strongly associated with a significant decrease in the expected number of claims. For a one-unit increase in policy term length, the expected number of claims is estimated to decrease by approximately 11.57%.
- **High Education Indicator:** The high education indicator has a marginally significant impact on the expected number of claims. Individuals with high education levels are associated with a decrease in the expected number of claims, although this effect is marginally significant.

Model Fit

- **Dispersion Parameter:** The dispersion parameter indicates how well the Poisson model fits the data. In this case, a value of 1 suggests a good fit.
- **Null and Residual Deviance:** Deviance measures how well the model explains the variability in the data. A lower residual deviance indicates a better fit. The model significantly reduces deviance compared to the null model.
- **AIC (Akaike Information Criterion):** AIC is a measure of the model's goodness of fit, penalizing for the number of parameters. Lower AIC values indicate a better balance between model complexity and fit.

The model provides insights into factors affecting the expected number of claims, considering exposure, vehicle body types, age, policy term length, and high education indicators.

Inverse Gaussian Regression Interpretation

Key Findings

- **Exposure Impact:** A decrease in exposure is associated with an increase in the expected number of claims. Specifically, for a one-unit decrease in exposure, we expect the number of claims to increase by approximately 53.70%.
- **Gender Effect:** Being male is associated with a significant increase in the expected number of claims compared to being female. Specifically, being male is associated with an approximate 24.35% increase in expected claims.
- **Area Influence:** Living in areas labeled B or F has a significant impact on increasing the expected number of claims compared to the reference area A. For example, living in area B is associated with an approximate 13.50% increase in expected claims.
- **Driving History Score:** There is a marginally significant positive association between the driving history score and the expected number of claims. For a one-unit increase in the driving history score, we expect an approximate 0.38% increase in expected claims.
- **Time of Week Driven:** Driving on weekends compared to weekdays has a marginally significant positive impact on the expected number of claims. Driving on weekends is associated with an approximate 13.05% increase in expected claims.
- **Policy Term Length (trm_len):** A longer policy term is associated with a slight increase in the expected number of claims, although not statistically significant. For a one-unit increase in policy term length, we expect an approximate 3.76% increase in expected claims.
- **Predicted Claims (numclaims_predicted):** The predicted number of claims has a positive impact on the expected number of claims, but this effect is not statistically significant. For a one-unit increase in predicted claims, we expect an approximate 179.79% increase in expected claims.

Explanations

- **Estimates:** The estimates represent the change in the expected log of claims for a one-unit change in each predictor.
- **Significance Codes:** Significance codes indicate the level of confidence we have in the observed effects. Smaller p-values (e.g., < 0.05) suggest stronger evidence of an effect.

Model Fit

- **Dispersion Parameter:** The dispersion parameter suggests how well the model fits the data. In this case, a lower value indicates a better fit.

- **Null and Residual Deviance:** Deviance is a measure of how well the model explains the variability in the data. A lower residual deviance suggests a better fit.
- **AIC (Akaike Information Criterion):** AIC is a measure of the model's goodness of fit, penalizing for the number of parameters. Lower AIC values indicate a better balance between model complexity and fit.

The model provides insights into factors affecting the expected number of claims, considering exposure, gender, area of residence, driving history, time of week driven, policy term length, and predicted claims.

```
#2.2

InsNova.data <- read.csv("data/InsNova_data_2023_train.csv")
train_data <- InsNova.data

InsNova.val_data <- read.csv("data/InsNova_data_2023_vh.csv")
test_data <- InsNova.val_data

y <- train_data$numclaims

set.seed(42)  # for reproducibility
split_index <- createDataPartition(y, p = 0.8, list = FALSE)
train_data_splitted <- train_data[split_index, ]
test_data_splitted <- train_data[-split_index, ]

pm.sub <- glm(numclaims ~ exposure + veh_body + agecat + trm_len + high_education_ind, family = poisson

predictions_train <- predict(pm.sub, newdata = train_data_splitted, type = "response")
predictions_test <- predict(pm.sub, newdata = test_data_splitted, type = "response")
train_data_splitted$numclaims_predicted <- predictions_train
train_data_splitted$clm_numclaims <- (train_data_splitted$claimcst0) / train_data_splitted$numclaims

ivg.sub <- glm((clm_numclaims + 0.01) ~ exposure + gender + area + driving_history_score +
    time_of_week_driven + trm_len + numclaims_predicted,
               family = inverse.gaussian(link = "log"),
               data = subset(train_data_splitted, numclaims_predicted > 0),
               control = glm.control(maxit = 1000))
summary(ivg.sub)
```

```
##
## Call:
## glm(formula = (clm_numclaims + 0.01) ~ exposure + gender + area +
##     driving_history_score + time_of_week_driven + trm_len + numclaims_predicted,
##     family = inverse.gaussian(link = "log"), data = subset(train_data_splitted,
##         numclaims_predicted > 0), control = glm.control(maxit = 1000))
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 6.994002   0.341291  20.493   <2e-16 ***
## exposure                   -0.851325   0.424606  -2.005   0.0452 *
## genderM                     0.254509   0.101223   2.514   0.0121 *
## areaB                       0.163744   0.143374   1.142   0.2536
## areaC                       0.180924   0.127874   1.415   0.1574
## areaD                       0.113292   0.176727   0.641   0.5216
## areaE                       0.154919   0.200012   0.775   0.4388
## areaF                       0.348125   0.242825   1.434   0.1519
```

```
## driving_history_score       0.003339   0.002543   1.313   0.1894
## time_of_week_drivenweekend  0.220350   0.126202   1.746   0.0811 .
## trm_len                     0.045654   0.027889   1.637   0.1019
## numclaims_predicted         1.775818   2.112085   0.841   0.4006
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for inverse.gaussian family taken to be 0.001323919)
##
##     Null deviance: 1.5862  on 1220  degrees of freedom
## Residual deviance: 1.5513  on 1209  degrees of freedom
##   (16875 observations deleted due to missingness)
## AIC: 20603
##
## Number of Fisher Scoring iterations: 20
```

```
Severity.pred.train <- predict(ivg.sub, newdata = train_data_splitted, type = "response")
Severity.pred <- predict(ivg.sub, newdata = train_data_splitted, type = "response")
final_prediction <- (predictions_train * Severity.pred.train )
NormalizedGini(train_data_splitted$claimcst0, final_prediction)
```

```
## [1] 0.250923
```

```
predictions_test_data <- predict(pm.sub, newdata = test_data_splitted, type = "response")
test_data_splitted$numclaims_predicted <- predictions_test_data
Severity.pred.test <- predict(ivg.sub, newdata = test_data_splitted, type = "response")
final_prediction_test <- predictions_test_data * Severity.pred.test
NormalizedGini(test_data_splitted$claimcst0, final_prediction_test)
```

```
## [1] 0.2685778
```

```
cv.ivg <- lapply(1:10, function(x) cv(fit=pm.sub, fit2=ivg.sub, data = train_data_splitted, data2=subse
```

```
##   |                                                                              |
## Cross-Validation Time Elapsed:  3.458 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  3.57 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  3.495 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  3.589 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  3.561 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  3.47 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  3.461 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  3.481 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  3.459 seconds
##   |                                                                              |
## Cross-Validation Time Elapsed:  3.497 seconds
```

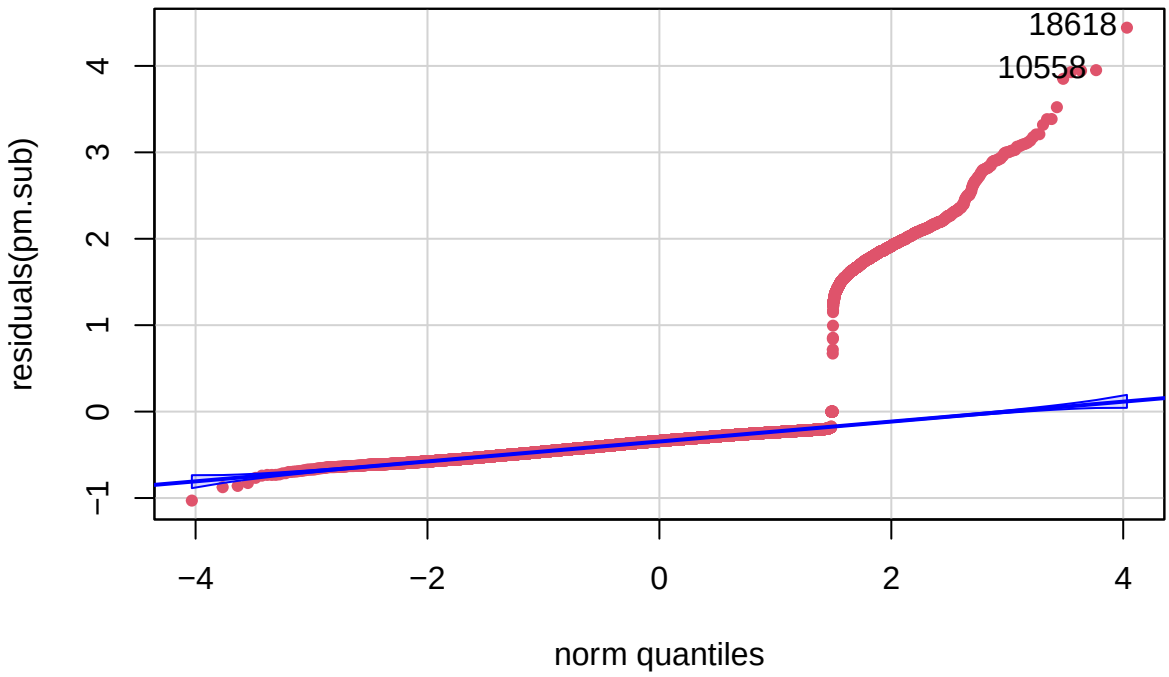
```
#mean of gini coefficients from 10 x 10-fold CV (around .21)
mean(sapply(1:10, function(x) NormalizedGini(train_data_splitted$claimcst0, cv.ivg[[x]])))
```

```
## [1] 0.1939808
```

```
#standard deviation of gini coefficient from 10 X 10-fold CV (around .002)
sd(sapply(1:10, function(x) NormalizedGini(train_data_splitted$claimcst0, cv.ivg[[x]])))
```

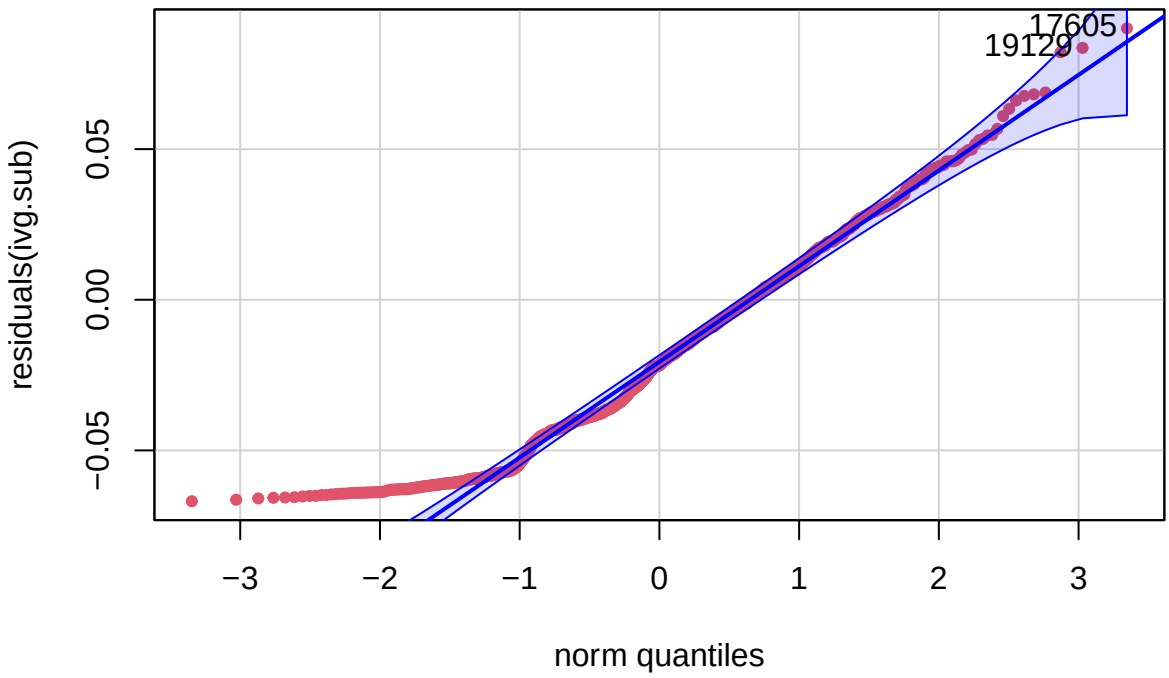
```
## [1] 0.007124999
```

```
car::qqPlot(residuals(pm.sub), main = NA, pch = 19, col = 2, cex = 0.7)
```




```
## 18618 10558
## 14875  8461
```

```
car::qqPlot(residuals(ivg.sub), main = NA, pch = 19, col = 2, cex = 0.7)
```




```
## 17605 19129
##   952  1034
```