CAS Group 24

## Dataset Import and Cleaning

library(readxl)  
library(dplyr)  
  
dataset <- read\_excel("dataset.xlsx", sheet = 1)  
  
str(dataset)

tibble [111,617 × 10] (S3: tbl\_df/tbl/data.frame)  
 $ PolicyID : num [1:111617] 1 2 3 4 5 6 7 8 9 10 ...  
 $ Power : chr [1:111617] "h" "d" "g" "h" ...  
 $ CarAge : num [1:111617] 7 10 25 9 10 1 25 4 29 13 ...  
 $ DriverAge : num [1:111617] 55 71 87 56 63 73 55 69 69 50 ...  
 $ Brand : chr [1:111617] "Renault, Nissan or Citroen" "Renault, Nissan or Citroen" "Renault, Nissan or Citroen" "Fiat" ...  
 $ Gas : chr [1:111617] "Diesel" "Regular" "Regular" "Diesel" ...  
 $ Region : chr [1:111617] "Centre" "Centre" "Centre" "Aquitaine" ...  
 $ Density : num [1:111617] 67 91 18 272 10 44 16 35 101 13 ...  
 $ ClaimNb : num [1:111617] 0 0 0 1 0 0 0 0 0 0 ...  
 $ ClaimAmount: num [1:111617] 0 0 0 1147 0 ...

# Change all the "chr" to factor  
  
dataset <- dataset %>%  
 mutate(  
 across(  
 where(is.character), function(col) {  
 factor(col, levels = unique(col))  
 }  
 )  
 )  
  
str(dataset)

tibble [111,617 × 10] (S3: tbl\_df/tbl/data.frame)  
 $ PolicyID : num [1:111617] 1 2 3 4 5 6 7 8 9 10 ...  
 $ Power : Factor w/ 12 levels "h","d","g","f",..: 1 2 3 1 2 2 3 4 5 6 ...  
 $ CarAge : num [1:111617] 7 10 25 9 10 1 25 4 29 13 ...  
 $ DriverAge : num [1:111617] 55 71 87 56 63 73 55 69 69 50 ...  
 $ Brand : Factor w/ 7 levels "Renault, Nissan or Citroen",..: 1 1 1 2 1 1 1 1 1 1 ...  
 $ Gas : Factor w/ 2 levels "Diesel","Regular": 1 2 2 1 2 2 1 2 2 1 ...  
 $ Region : Factor w/ 10 levels "Centre","Aquitaine",..: 1 1 1 2 1 1 1 1 1 1 ...  
 $ Density : num [1:111617] 67 91 18 272 10 44 16 35 101 13 ...  
 $ ClaimNb : num [1:111617] 0 0 0 1 0 0 0 0 0 0 ...  
 $ ClaimAmount: num [1:111617] 0 0 0 1147 0 ...

## Count Models (Poisson and Negative Binomial)

### Train and Test dataset split

library(caret)  
  
# We use createDataPartition rather than sample[] is because createDataPartition keeps the distribution of the response variable.   
# From the example above we see approximate 5% of the response have 1 claim so the test set will also have approximate 5% of claim size of 1.  
# This train dataset will consist of 70% of the data from the full dataset  
set.seed(716)  
  
train\_index <- createDataPartition(dataset$ClaimNb,p = 0.7,list = FALSE)  
  
data\_train <- dataset[train\_index,]  
  
data\_test <- dataset[-train\_index,]  
  
# Construct Dataset for counting model comparison  
count\_cols <- c("Power", "CarAge", "DriverAge", "Brand", "Gas", "Region", "Density", "ClaimNb")  
  
claim\_number\_train\_set <- data\_train[,count\_cols]  
claim\_number\_test\_set <- data\_test[,count\_cols]  
  
names(claim\_number\_train\_set)

[1] "Power" "CarAge" "DriverAge" "Brand" "Gas" "Region"   
[7] "Density" "ClaimNb"

names(claim\_number\_test\_set)

[1] "Power" "CarAge" "DriverAge" "Brand" "Gas" "Region"   
[7] "Density" "ClaimNb"

### Poisson GLM model

poisson\_model <- glm(ClaimNb ~., data = claim\_number\_train\_set, family = poisson(link = "log"))  
  
summary(poisson\_model)

Call:  
glm(formula = ClaimNb ~ ., family = poisson(link = "log"), data = claim\_number\_train\_set)  
  
Coefficients:  
 Estimate Std. Error z value Pr(>|z|)  
(Intercept) -2.601e+00 9.717e-02 -26.766 < 2e-16  
Powerd -1.375e-01 8.332e-02 -1.650 0.098890  
Powerg -3.294e-02 7.669e-02 -0.429 0.667569  
Powerf -3.930e-02 7.540e-02 -0.521 0.602198  
Powerk 2.217e-01 1.323e-01 1.676 0.093748  
Powere -7.432e-02 7.808e-02 -0.952 0.341180  
Poweri 7.652e-02 1.060e-01 0.722 0.470400  
Powerj 1.884e-01 1.031e-01 1.827 0.067711  
Powero 2.206e-01 3.623e-01 0.609 0.542605  
Powerl 2.411e-02 1.988e-01 0.121 0.903486  
Powerm 1.408e-01 2.431e-01 0.579 0.562389  
Powern 4.771e-01 2.614e-01 1.825 0.067968  
CarAge -3.529e-02 3.302e-03 -10.688 < 2e-16  
DriverAge -2.795e-03 1.133e-03 -2.468 0.013605  
BrandFiat 2.065e-02 7.987e-02 0.258 0.796025  
BrandOpel, General Motors or Ford 1.002e-01 5.230e-02 1.916 0.055311  
BrandVolkswagen, Audi, Skoda or Seat 3.759e-02 5.880e-02 0.639 0.522589  
BrandJapanese (except Nissan) or Korean 2.153e-01 8.566e-02 2.513 0.011974  
BrandMercedes, Chrysler or BMW 1.108e-02 7.969e-02 0.139 0.889434  
Brandother 1.205e-01 9.519e-02 1.266 0.205569  
GasRegular -1.649e-01 3.520e-02 -4.685 2.80e-06  
RegionAquitaine 3.509e-01 7.139e-02 4.916 8.83e-07  
RegionBretagne 1.145e-01 4.830e-02 2.372 0.017701  
RegionPays-de-la-Loire 2.803e-01 5.358e-02 5.232 1.68e-07  
RegionIle-de-France 3.138e-01 7.684e-02 4.084 4.43e-05  
RegionBasse-Normandie 2.556e-01 8.796e-02 2.906 0.003660  
RegionPoitou-Charentes 4.467e-02 7.832e-02 0.570 0.568423  
RegionHaute-Normandie -7.991e-02 1.843e-01 -0.434 0.664628  
RegionNord-Pas-de-Calais 3.217e-01 8.644e-02 3.722 0.000198  
RegionLimousin 4.991e-01 1.465e-01 3.406 0.000658  
Density 1.846e-05 4.648e-06 3.971 7.17e-05  
   
(Intercept) \*\*\*  
Powerd .   
Powerg   
Powerf   
Powerk .   
Powere   
Poweri   
Powerj .   
Powero   
Powerl   
Powerm   
Powern .   
CarAge \*\*\*  
DriverAge \*   
BrandFiat   
BrandOpel, General Motors or Ford .   
BrandVolkswagen, Audi, Skoda or Seat   
BrandJapanese (except Nissan) or Korean \*   
BrandMercedes, Chrysler or BMW   
Brandother   
GasRegular \*\*\*  
RegionAquitaine \*\*\*  
RegionBretagne \*   
RegionPays-de-la-Loire \*\*\*  
RegionIle-de-France \*\*\*  
RegionBasse-Normandie \*\*   
RegionPoitou-Charentes   
RegionHaute-Normandie   
RegionNord-Pas-de-Calais \*\*\*  
RegionLimousin \*\*\*  
Density \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
(Dispersion parameter for poisson family taken to be 1)  
  
 Null deviance: 23980 on 78131 degrees of freedom  
Residual deviance: 23608 on 78101 degrees of freedom  
AIC: 31279  
  
Number of Fisher Scoring iterations: 6

### Negative Binomial GLM model

library(MASS)  
  
negative\_binomial\_model <- glm.nb(ClaimNb ~., data = claim\_number\_train\_set)  
# This code automatically estimate the overdispersion parameter theta  
  
summary(negative\_binomial\_model)

Call:  
glm.nb(formula = ClaimNb ~ ., data = claim\_number\_train\_set,   
 init.theta = 1.450702501, link = log)  
  
Coefficients:  
 Estimate Std. Error z value Pr(>|z|)  
(Intercept) -2.600e+00 9.901e-02 -26.263 < 2e-16  
Powerd -1.379e-01 8.485e-02 -1.626 0.104012  
Powerg -3.288e-02 7.816e-02 -0.421 0.673994  
Powerf -3.934e-02 7.686e-02 -0.512 0.608772  
Powerk 2.213e-01 1.353e-01 1.635 0.102048  
Powere -7.540e-02 7.960e-02 -0.947 0.343488  
Poweri 7.670e-02 1.081e-01 0.710 0.478004  
Powerj 1.879e-01 1.054e-01 1.783 0.074600  
Powero 2.170e-01 3.725e-01 0.582 0.560274  
Powerl 2.279e-02 2.034e-01 0.112 0.910775  
Powerm 1.399e-01 2.487e-01 0.563 0.573765  
Powern 4.724e-01 2.696e-01 1.752 0.079750  
CarAge -3.537e-02 3.357e-03 -10.535 < 2e-16  
DriverAge -2.790e-03 1.153e-03 -2.420 0.015529  
BrandFiat 2.048e-02 8.137e-02 0.252 0.801234  
BrandOpel, General Motors or Ford 1.004e-01 5.334e-02 1.883 0.059723  
BrandVolkswagen, Audi, Skoda or Seat 3.690e-02 5.998e-02 0.615 0.538335  
BrandJapanese (except Nissan) or Korean 2.148e-01 8.782e-02 2.446 0.014441  
BrandMercedes, Chrysler or BMW 1.260e-02 8.131e-02 0.155 0.876886  
Brandother 1.208e-01 9.723e-02 1.243 0.213944  
GasRegular -1.652e-01 3.585e-02 -4.608 4.07e-06  
RegionAquitaine 3.514e-01 7.298e-02 4.815 1.47e-06  
RegionBretagne 1.147e-01 4.915e-02 2.333 0.019639  
RegionPays-de-la-Loire 2.795e-01 5.466e-02 5.113 3.17e-07  
RegionIle-de-France 3.135e-01 7.857e-02 3.991 6.59e-05  
RegionBasse-Normandie 2.569e-01 8.976e-02 2.862 0.004209  
RegionPoitou-Charentes 4.451e-02 7.963e-02 0.559 0.576187  
RegionHaute-Normandie -7.981e-02 1.872e-01 -0.426 0.669903  
RegionNord-Pas-de-Calais 3.221e-01 8.836e-02 3.645 0.000267  
RegionLimousin 4.976e-01 1.504e-01 3.309 0.000938  
Density 1.875e-05 4.778e-06 3.926 8.65e-05  
   
(Intercept) \*\*\*  
Powerd   
Powerg   
Powerf   
Powerk   
Powere   
Poweri   
Powerj .   
Powero   
Powerl   
Powerm   
Powern .   
CarAge \*\*\*  
DriverAge \*   
BrandFiat   
BrandOpel, General Motors or Ford .   
BrandVolkswagen, Audi, Skoda or Seat   
BrandJapanese (except Nissan) or Korean \*   
BrandMercedes, Chrysler or BMW   
Brandother   
GasRegular \*\*\*  
RegionAquitaine \*\*\*  
RegionBretagne \*   
RegionPays-de-la-Loire \*\*\*  
RegionIle-de-France \*\*\*  
RegionBasse-Normandie \*\*   
RegionPoitou-Charentes   
RegionHaute-Normandie   
RegionNord-Pas-de-Calais \*\*\*  
RegionLimousin \*\*\*  
Density \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
(Dispersion parameter for Negative Binomial(1.4507) family taken to be 1)  
  
 Null deviance: 21726 on 78131 degrees of freedom  
Residual deviance: 21366 on 78101 degrees of freedom  
AIC: 31240  
  
Number of Fisher Scoring iterations: 1  
  
 Theta: 1.451   
 Std. Err.: 0.274   
  
 2 x log-likelihood: -31176.238

### Model Evaluation

library(pscl)  
  
vuong(poisson\_model,negative\_binomial\_model)

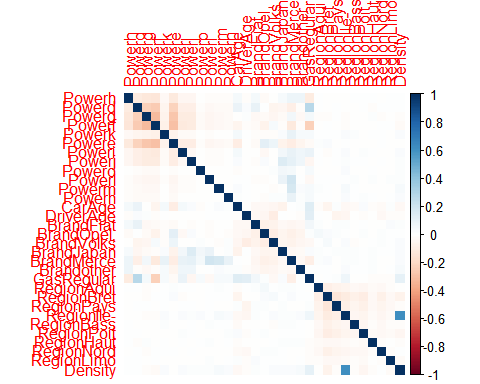
Vuong Non-Nested Hypothesis Test-Statistic:   
(test-statistic is asymptotically distributed N(0,1) under the  
 null that the models are indistinguishible)  
-------------------------------------------------------------  
 Vuong z-statistic H\_A p-value  
Raw -3.015374 model2 > model1 0.0012833  
AIC-corrected -3.015374 model2 > model1 0.0012833  
BIC-corrected -3.015374 model2 > model1 0.0012833

# This function allows us to compare between 2 models  
# From the output dataframe we can see that Negative Binomial Model is better than Poisson Model (Alternative Hypothesis is accepted since p-value are very small) in terms of Vuong Test Statistics, Akaike and Bayesian Information Criterion.  
  
# This can be explained by the overdispersion observed previously

### Multicollinearity Check

#### 1. Correlation Matrix (Examine nC2 pairs of regressors)

library(corrplot)  
# Multicollinearity issue in the dataset can cause poor regression coefficients estimates  
# Dummy coding since we have factors in the dataset, we can express them with numbers  
claim\_number\_train\_set\_mm <- model.matrix(ClaimNb ~. -1, data = claim\_number\_train\_set)  
# Drop the Intercept  
  
claim\_number\_train\_set\_mm <- as.data.frame(claim\_number\_train\_set\_mm)  
  
claim\_number\_train\_cor\_mat <- cor(claim\_number\_train\_set\_mm)  
  
# Make the correlation matrix have column and row names that is the first to 10th character of the original name  
colnames(claim\_number\_train\_cor\_mat) <- substr(colnames(claim\_number\_train\_cor\_mat),1,10)  
  
rownames(claim\_number\_train\_cor\_mat) <- substr(rownames(claim\_number\_train\_cor\_mat),1,10)  
  
  
corrplot(claim\_number\_train\_cor\_mat, method = "color")



# Output the name of highly-correlated pair  
corr\_detect <- which(abs(claim\_number\_train\_cor\_mat) > 0.5 & abs(claim\_number\_train\_cor\_mat) < 1, arr.ind = TRUE)  
# Return array with row number  
  
apply(corr\_detect,MARGIN = 1,function(corr\_pair){  
 i <- corr\_pair[1]  
 j <- corr\_pair[2]  
 rowname <- colnames(claim\_number\_train\_set\_mm)  
 colname <- colnames(claim\_number\_train\_set\_mm)  
   
 cat(paste(rowname[i],"and",colname[j],"having Correlation Coefficient of",round(claim\_number\_train\_cor\_mat[i,j],3)),"\n")  
})

Density and RegionIle-de-France having Correlation Coefficient of 0.617   
RegionIle-de-France and Density having Correlation Coefficient of 0.617

NULL

#### 2. Variance Inflation Factor

library(car)  
  
# GVIF is the Generalized VIF  
# It is used when a model includes categorical variables with more than 2 levels.  
GVIF\_summary <- data.frame(  
 VIF\_Poisson\_model = vif(poisson\_model),  
 VIF\_NBinomial\_model = vif(negative\_binomial\_model)  
)  
  
colnames(GVIF\_summary) <- c("GVIF\_Poisson","df\_Poisson","Scaled\_Poisson\_GVIF","GVIF\_Nbinomial","df\_NBinomial","Scaled\_NBinomial\_GVIF")  
  
GVIF\_summary

GVIF\_Poisson df\_Poisson Scaled\_Poisson\_GVIF GVIF\_Nbinomial  
Power 1.590891 11 1.021329 1.588496  
CarAge 1.053662 1 1.026480 1.053578  
DriverAge 1.042400 1 1.020980 1.042366  
Brand 1.425236 6 1.029968 1.422760  
Gas 1.213401 1 1.101545 1.213513  
Region 1.898472 9 1.036256 1.888635  
Density 1.847896 1 1.359374 1.838453  
 df\_NBinomial Scaled\_NBinomial\_GVIF  
Power 11 1.021259  
CarAge 1 1.026439  
DriverAge 1 1.020963  
Brand 6 1.029819  
Gas 1 1.101596  
Region 9 1.035957  
Density 1 1.355896

# Scaled GVIF is calculated through GVIF^(1/2df)  
# < 2: good  
# 2–5: moderate concern  
# > 5: problematic

library(olsrr)  
  
# Using the OLSRR package to get classic VIF for each individual variable (including dummy variables).  
vif\_count\_poisson <- ols\_coll\_diag(poisson\_model)  
# ols\_coll\_diag will automatically access the model matrix (the matrix with dummy variable) to calculate the classic VIF  
vif\_count\_nbinomial <- ols\_coll\_diag(negative\_binomial\_model)  
  
vif\_count\_poisson <- vif\_count\_poisson$vif\_t[,c("Variables","VIF")]  
vif\_count\_nbinomial <- vif\_count\_nbinomial$vif\_t[,c("Variables","VIF")]  
  
colnames(vif\_count\_poisson) <- c("Variables", "VIF\_Poisson")  
colnames(vif\_count\_nbinomial) <- c("Variables", "VIF\_Negative\_Binomial")  
  
  
VIF\_summary <- merge(vif\_count\_poisson,vif\_count\_nbinomial,by = "Variables")  
   
VIF\_summary

Variables VIF\_Poisson VIF\_Negative\_Binomial  
1 BrandFiat 1.039253 1.039253  
2 BrandJapanese (except Nissan) or Korean 1.081608 1.081608  
3 BrandMercedes, Chrysler or BMW 1.212844 1.212844  
4 BrandOpel, General Motors or Ford 1.064396 1.064396  
5 Brandother 1.042860 1.042860  
6 BrandVolkswagen, Audi, Skoda or Seat 1.057051 1.057051  
7 CarAge 1.061301 1.061301  
8 Density 1.642825 1.642825  
9 DriverAge 1.043023 1.043023  
10 GasRegular 1.221117 1.221117  
11 Powerd 3.640475 3.640475  
12 Powere 4.024391 4.024391  
13 Powerf 4.462432 4.462432  
14 Powerg 4.192929 4.192929  
15 Poweri 1.670296 1.670296  
16 Powerj 1.590624 1.590624  
17 Powerk 1.293225 1.293225  
18 Powerl 1.153115 1.153115  
19 Powerm 1.105211 1.105211  
20 Powern 1.057572 1.057572  
21 Powero 1.033564 1.033564  
22 RegionAquitaine 1.044915 1.044915  
23 RegionBasse-Normandie 1.029213 1.029213  
24 RegionBretagne 1.103626 1.103626  
25 RegionHaute-Normandie 1.010158 1.010158  
26 RegionIle-de-France 1.703727 1.703727  
27 RegionLimousin 1.009553 1.009553  
28 RegionNord-Pas-de-Calais 1.032115 1.032115  
29 RegionPays-de-la-Loire 1.081202 1.081202  
30 RegionPoitou-Charentes 1.043414 1.043414

library(mctest)  
  
mctest(poisson\_model)

Call:  
omcdiag(mod = mod, Inter = TRUE, detr = detr, red = red, conf = conf,   
 theil = theil, cn = cn)  
  
  
Overall Multicollinearity Diagnostics  
  
 MC Results detection  
Determinant |X'X|: 0.0439 0  
Farrar Chi-Square: 244146.1132 1  
Red Indicator: 0.0614 0  
Sum of Lambda Inverse: 46.7480 0  
Theil's Method: 6.0545 1  
Condition Number: 16.7301 0  
  
1 --> COLLINEARITY is detected by the test   
0 --> COLLINEARITY is not detected by the test

mctest(negative\_binomial\_model)

Call:  
omcdiag(mod = mod, Inter = TRUE, detr = detr, red = red, conf = conf,   
 theil = theil, cn = cn)  
  
  
Overall Multicollinearity Diagnostics  
  
 MC Results detection  
Determinant |X'X|: 0.0439 0  
Farrar Chi-Square: 244146.1132 1  
Red Indicator: 0.0614 0  
Sum of Lambda Inverse: 46.7480 0  
Theil's Method: 6.0545 1  
Condition Number: 16.7301 0  
  
1 --> COLLINEARITY is detected by the test   
0 --> COLLINEARITY is not detected by the test

## Zero-Inflated Models

### Dataset Preprocessing

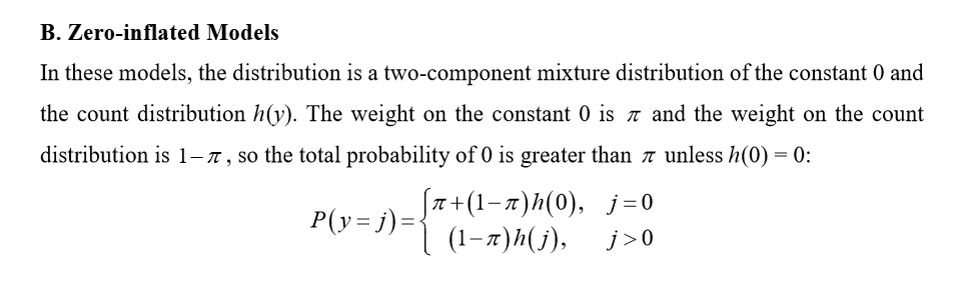
claim\_number\_train\_scaled <- claim\_number\_train\_set  
claim\_number\_test\_scaled <- claim\_number\_test\_set  
  
# For these numerical variables, we change them into standardized data  
claim\_number\_train\_scaled[,c("CarAge","DriverAge","Density")] <- scale(claim\_number\_train\_scaled[,c("CarAge","DriverAge","Density")])  
  
claim\_number\_test\_scaled[,c("CarAge","DriverAge","Density")] <- scale(claim\_number\_test\_scaled[,c("CarAge","DriverAge","Density")])  
  
head(claim\_number\_train\_scaled)

# A tibble: 6 × 8  
 Power CarAge DriverAge Brand Gas Region Density ClaimNb  
 <fct> <dbl> <dbl> <fct> <fct> <fct> <dbl> <dbl>  
1 h -0.354 0.343 Renault, Nissan or Citro… Dies… Centre -0.293 0  
2 d 0.193 1.43 Renault, Nissan or Citro… Regu… Centre -0.286 0  
3 h 0.0105 0.411 Fiat Dies… Aquit… -0.234 1  
4 d 0.193 0.885 Renault, Nissan or Citro… Regu… Centre -0.309 0  
5 d -1.45 1.56 Renault, Nissan or Citro… Regu… Centre -0.299 0  
6 g 2.92 0.343 Renault, Nissan or Citro… Dies… Centre -0.308 0

### Zero-Inflated Model

# We seperate the modelling process into 2 part A|B:  
# The count component (A) models the number of claims (including the possibility of 0) using a Negative Binomial distribution to allow for overdispersion.  
# The zero-inflation component (B) models the probability that an observation is a guaranteed zero not due to the count process.  
zero\_nbinomial\_model <- zeroinfl(ClaimNb ~. | 1 + CarAge, data = claim\_number\_train\_scaled,  
 dist = "negbin")  
  
  
summary(zero\_nbinomial\_model)

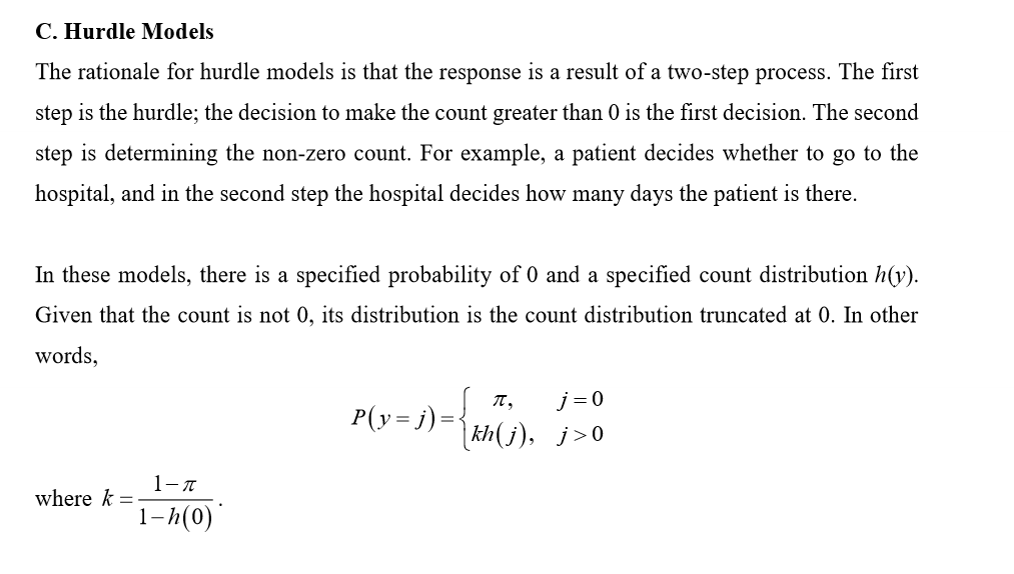
Call:  
zeroinfl(formula = ClaimNb ~ . | 1 + CarAge, data = claim\_number\_train\_scaled,   
 dist = "negbin")  
  
Pearson residuals:  
 Min 1Q Median 3Q Max   
-0.4314 -0.2348 -0.2131 -0.1916 19.3090   
  
Count model coefficients (negbin with log link):  
 Estimate Std. Error z value Pr(>|z|)  
(Intercept) -2.50791 0.10602 -23.654 < 2e-16  
Powerd -0.13793 0.08489 -1.625 0.104195  
Powerg -0.03294 0.07820 -0.421 0.673579  
Powerf -0.03953 0.07690 -0.514 0.607244  
Powerk 0.22181 0.13536 1.639 0.101268  
Powere -0.07445 0.07963 -0.935 0.349805  
Poweri 0.07631 0.10813 0.706 0.480379  
Powerj 0.18889 0.10537 1.793 0.073022  
Powero 0.21243 0.37151 0.572 0.567453  
Powerl 0.02408 0.20314 0.119 0.905660  
Powerm 0.13881 0.24870 0.558 0.576734  
Powern 0.47931 0.26931 1.780 0.075108  
CarAge -0.17501 0.05656 -3.094 0.001974  
DriverAge -0.04085 0.01700 -2.403 0.016270  
BrandFiat 0.02049 0.08135 0.252 0.801141  
BrandOpel, General Motors or Ford 0.09948 0.05333 1.865 0.062139  
BrandVolkswagen, Audi, Skoda or Seat 0.03671 0.05995 0.612 0.540229  
BrandJapanese (except Nissan) or Korean 0.21401 0.08767 2.441 0.014642  
BrandMercedes, Chrysler or BMW 0.01254 0.08130 0.154 0.877434  
Brandother 0.12002 0.09722 1.234 0.217017  
GasRegular -0.16483 0.03586 -4.596 4.30e-06  
RegionAquitaine 0.35212 0.07297 4.825 1.40e-06  
RegionBretagne 0.11513 0.04914 2.343 0.019136  
RegionPays-de-la-Loire 0.27929 0.05464 5.112 3.19e-07  
RegionIle-de-France 0.31363 0.07848 3.996 6.43e-05  
RegionBasse-Normandie 0.25755 0.08979 2.868 0.004128  
RegionPoitou-Charentes 0.04453 0.07960 0.559 0.575858  
RegionHaute-Normandie -0.08074 0.18718 -0.431 0.666211  
RegionNord-Pas-de-Calais 0.32346 0.08836 3.661 0.000252  
RegionLimousin 0.50326 0.15030 3.348 0.000813  
Density 0.06496 0.01660 3.914 9.09e-05  
Log(theta) 12.76215 NaN NaN NaN  
   
(Intercept) \*\*\*  
Powerd   
Powerg   
Powerf   
Powerk   
Powere   
Poweri   
Powerj .   
Powero   
Powerl   
Powerm   
Powern .   
CarAge \*\*   
DriverAge \*   
BrandFiat   
BrandOpel, General Motors or Ford .   
BrandVolkswagen, Audi, Skoda or Seat   
BrandJapanese (except Nissan) or Korean \*   
BrandMercedes, Chrysler or BMW   
Brandother   
GasRegular \*\*\*  
RegionAquitaine \*\*\*  
RegionBretagne \*   
RegionPays-de-la-Loire \*\*\*  
RegionIle-de-France \*\*\*  
RegionBasse-Normandie \*\*   
RegionPoitou-Charentes   
RegionHaute-Normandie   
RegionNord-Pas-de-Calais \*\*\*  
RegionLimousin \*\*\*  
Density \*\*\*  
Log(theta)   
  
Zero-inflation model coefficients (binomial with logit link):  
 Estimate Std. Error z value Pr(>|z|)   
(Intercept) -0.36349 0.18466 -1.968 0.049 \*  
CarAge 0.04627 0.12730 0.363 0.716   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1   
  
Theta = 348762.8751   
Number of iterations in BFGS optimization: 70   
Log-likelihood: -1.559e+04 on 34 Df



### Hurdle Model

# Hurdle Model does similar thing as the Zero-Inflated Model but the count part models positive count only (claim number of at least 1)  
hurdle\_model <- hurdle(ClaimNb ~.|., data = claim\_number\_train\_scaled, dist = "negbin")  
  
summary(hurdle\_model)

Call:  
hurdle(formula = ClaimNb ~ . | ., data = claim\_number\_train\_scaled, dist = "negbin")  
  
Pearson residuals:  
 Min 1Q Median 3Q Max   
-0.4385 -0.2355 -0.2134 -0.1909 19.0642   
  
Count model coefficients (truncated negbin with log link):  
 Estimate Std. Error z value Pr(>|z|)  
(Intercept) -2.799e+00 3.829e-01 -7.310 2.68e-13  
Powerd -4.678e-02 4.294e-01 -0.109 0.913261  
Powerg -6.680e-02 4.005e-01 -0.167 0.867526  
Powerf -5.062e-03 3.928e-01 -0.013 0.989718  
Powerk 4.383e-01 5.702e-01 0.769 0.442064  
Powere 3.733e-01 3.929e-01 0.950 0.342115  
Poweri -2.436e-01 5.732e-01 -0.425 0.670849  
Powerj 5.538e-01 4.645e-01 1.192 0.233150  
Powero -2.376e+01 2.108e+05 0.000 0.999910  
Powerl 3.047e-01 7.979e-01 0.382 0.702532  
Powerm -2.562e-02 1.071e+00 -0.024 0.980912  
Powern 1.166e+00 7.868e-01 1.482 0.138401  
CarAge -1.191e-01 9.102e-02 -1.309 0.190682  
DriverAge 1.514e-01 7.521e-02 2.014 0.044059  
BrandFiat 3.644e-02 3.692e-01 0.099 0.921369  
BrandOpel, General Motors or Ford -2.564e-01 2.619e-01 -0.979 0.327586  
BrandVolkswagen, Audi, Skoda or Seat -9.052e-02 2.708e-01 -0.334 0.738153  
BrandJapanese (except Nissan) or Korean -2.920e-01 4.037e-01 -0.723 0.469506  
BrandMercedes, Chrysler or BMW 8.760e-02 3.483e-01 0.251 0.801440  
Brandother -3.769e-01 5.130e-01 -0.735 0.462449  
GasRegular 5.776e-02 1.673e-01 0.345 0.729976  
RegionAquitaine 6.934e-01 2.887e-01 2.402 0.016296  
RegionBretagne 3.825e-01 2.299e-01 1.664 0.096109  
RegionPays-de-la-Loire 3.051e-01 2.644e-01 1.154 0.248561  
RegionIle-de-France 4.613e-01 3.316e-01 1.391 0.164172  
RegionBasse-Normandie 5.343e-01 3.750e-01 1.425 0.154150  
RegionPoitou-Charentes 2.149e-02 4.281e-01 0.050 0.959963  
RegionHaute-Normandie -2.659e+01 5.594e+03 -0.005 0.996207  
RegionNord-Pas-de-Calais 8.671e-01 3.286e-01 2.638 0.008329  
RegionLimousin 1.402e+00 4.230e-01 3.314 0.000918  
Density 7.402e-02 6.257e-02 1.183 0.236808  
Log(theta) 7.846e+00 NaN NaN NaN  
   
(Intercept) \*\*\*  
Powerd   
Powerg   
Powerf   
Powerk   
Powere   
Poweri   
Powerj   
Powero   
Powerl   
Powerm   
Powern   
CarAge   
DriverAge \*   
BrandFiat   
BrandOpel, General Motors or Ford   
BrandVolkswagen, Audi, Skoda or Seat   
BrandJapanese (except Nissan) or Korean   
BrandMercedes, Chrysler or BMW   
Brandother   
GasRegular   
RegionAquitaine \*   
RegionBretagne .   
RegionPays-de-la-Loire   
RegionIle-de-France   
RegionBasse-Normandie   
RegionPoitou-Charentes   
RegionHaute-Normandie   
RegionNord-Pas-de-Calais \*\*   
RegionLimousin \*\*\*  
Density   
Log(theta)   
Zero hurdle model coefficients (binomial with logit link):  
 Estimate Std. Error z value Pr(>|z|)  
(Intercept) -3.0144896 0.0759394 -39.696 < 2e-16  
Powerd -0.1447268 0.0872495 -1.659 0.097162  
Powerg -0.0328002 0.0803718 -0.408 0.683196  
Powerf -0.0434153 0.0790494 -0.549 0.582856  
Powerk 0.2119915 0.1407717 1.506 0.132087  
Powere -0.1022869 0.0820399 -1.247 0.212473  
Poweri 0.0884458 0.1110677 0.796 0.425844  
Powerj 0.1670510 0.1093233 1.528 0.126501  
Powero 0.2808820 0.3779343 0.743 0.457359  
Powerl -0.0004708 0.2121509 -0.002 0.998229  
Powerm 0.1441192 0.2580932 0.558 0.576571  
Powern 0.4185123 0.2898396 1.444 0.148755  
CarAge -0.1986915 0.0189878 -10.464 < 2e-16  
DriverAge -0.0507529 0.0175772 -2.887 0.003884  
BrandFiat 0.0228052 0.0840108 0.271 0.786041  
BrandOpel, General Motors or Ford 0.1206320 0.0549230 2.196 0.028064  
BrandVolkswagen, Audi, Skoda or Seat 0.0427130 0.0619561 0.689 0.490567  
BrandJapanese (except Nissan) or Korean 0.2430543 0.0908279 2.676 0.007451  
BrandMercedes, Chrysler or BMW 0.0091127 0.0842629 0.108 0.913880  
Brandother 0.1451075 0.0999462 1.452 0.146542  
GasRegular -0.1773408 0.0369927 -4.794 1.64e-06  
RegionAquitaine 0.3331933 0.0761512 4.375 1.21e-05  
RegionBretagne 0.1034833 0.0506886 2.042 0.041196  
RegionPays-de-la-Loire 0.2824184 0.0563231 5.014 5.32e-07  
RegionIle-de-France 0.3088161 0.0815464 3.787 0.000152  
RegionBasse-Normandie 0.2438701 0.0932651 2.615 0.008928  
RegionPoitou-Charentes 0.0449978 0.0815840 0.552 0.581256  
RegionHaute-Normandie -0.0506711 0.1888046 -0.268 0.788408  
RegionNord-Pas-de-Calais 0.2923135 0.0926020 3.157 0.001596  
RegionLimousin 0.4195754 0.1618712 2.592 0.009541  
Density 0.0646640 0.0174043 3.715 0.000203  
   
(Intercept) \*\*\*  
Powerd .   
Powerg   
Powerf   
Powerk   
Powere   
Poweri   
Powerj   
Powero   
Powerl   
Powerm   
Powern   
CarAge \*\*\*  
DriverAge \*\*   
BrandFiat   
BrandOpel, General Motors or Ford \*   
BrandVolkswagen, Audi, Skoda or Seat   
BrandJapanese (except Nissan) or Korean \*\*   
BrandMercedes, Chrysler or BMW   
Brandother   
GasRegular \*\*\*  
RegionAquitaine \*\*\*  
RegionBretagne \*   
RegionPays-de-la-Loire \*\*\*  
RegionIle-de-France \*\*\*  
RegionBasse-Normandie \*\*   
RegionPoitou-Charentes   
RegionHaute-Normandie   
RegionNord-Pas-de-Calais \*\*   
RegionLimousin \*\*   
Density \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1   
  
Theta: count = 2555.4829  
Number of iterations in BFGS optimization: 59   
Log-likelihood: -1.557e+04 on 63 Df



### Tweedie Model (Compound Poisson-Gamma)

library(statmod)  
  
tweedie\_model <- glm(ClaimNb ~., data = claim\_number\_train\_scaled,  
 family = tweedie(var.power = 1.5, link.power = 0))  
  
summary(tweedie\_model)

Call:  
glm(formula = ClaimNb ~ ., family = tweedie(var.power = 1.5,   
 link.power = 0), data = claim\_number\_train\_scaled)  
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)  
(Intercept) -3.034616 0.076506 -39.665 < 2e-16  
Powerd -0.138830 0.086880 -1.598 0.110057  
Powerg -0.023561 0.080975 -0.291 0.771078  
Powerf -0.033580 0.079800 -0.421 0.673903  
Powerk 0.218010 0.145344 1.500 0.133627  
Powere -0.079699 0.082568 -0.965 0.334421  
Poweri 0.080711 0.112658 0.716 0.473733  
Powerj 0.174112 0.112721 1.545 0.122440  
Powero 0.179285 0.419186 0.428 0.668872  
Powerl 0.002788 0.218398 0.013 0.989816  
Powerm 0.128778 0.265907 0.484 0.628176  
Powern 0.427587 0.310255 1.378 0.168152  
CarAge -0.181572 0.018178 -9.988 < 2e-16  
DriverAge -0.040850 0.017295 -2.362 0.018184  
BrandFiat 0.021595 0.083746 0.258 0.796513  
BrandOpel, General Motors or Ford 0.103629 0.055751 1.859 0.063063  
BrandVolkswagen, Audi, Skoda or Seat 0.029510 0.062645 0.471 0.637590  
BrandJapanese (except Nissan) or Korean 0.213664 0.096921 2.205 0.027490  
BrandMercedes, Chrysler or BMW 0.032836 0.085442 0.384 0.700754  
Brandother 0.117241 0.103274 1.135 0.256278  
GasRegular -0.174415 0.036883 -4.729 2.26e-06  
RegionAquitaine 0.356011 0.078663 4.526 6.03e-06  
RegionBretagne 0.123390 0.050137 2.461 0.013855  
RegionPays-de-la-Loire 0.270090 0.057520 4.696 2.66e-06  
RegionIle-de-France 0.321346 0.084676 3.795 0.000148  
RegionBasse-Normandie 0.271856 0.095197 2.856 0.004295  
RegionPoitou-Charentes 0.039589 0.079957 0.495 0.620513  
RegionHaute-Normandie -0.078772 0.186064 -0.423 0.672032  
RegionNord-Pas-de-Calais 0.330301 0.095264 3.467 0.000526  
RegionLimousin 0.491080 0.167718 2.928 0.003412  
Density 0.075607 0.018698 4.044 5.27e-05  
   
(Intercept) \*\*\*  
Powerd   
Powerg   
Powerf   
Powerk   
Powere   
Poweri   
Powerj   
Powero   
Powerl   
Powerm   
Powern   
CarAge \*\*\*  
DriverAge \*   
BrandFiat   
BrandOpel, General Motors or Ford .   
BrandVolkswagen, Audi, Skoda or Seat   
BrandJapanese (except Nissan) or Korean \*   
BrandMercedes, Chrysler or BMW   
Brandother   
GasRegular \*\*\*  
RegionAquitaine \*\*\*  
RegionBretagne \*   
RegionPays-de-la-Loire \*\*\*  
RegionIle-de-France \*\*\*  
RegionBasse-Normandie \*\*   
RegionPoitou-Charentes   
RegionHaute-Normandie   
RegionNord-Pas-de-Calais \*\*\*  
RegionLimousin \*\*   
Density \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
(Dispersion parameter for Tweedie family taken to be 4.835462)  
  
 Null deviance: 109543 on 78131 degrees of freedom  
Residual deviance: 107889 on 78101 degrees of freedom  
AIC: NA  
  
Number of Fisher Scoring iterations: 6

## Validation with RSME

So far we have built:

1. Poisson GLM
2. Negative Binomial GLM
3. Zero-Inflated Model
4. Hurdle Model
5. Tweedie Model

library(Metrics)  
# We use all of the model above to predict E[y|x] which is the expected no of claims using the test dataset  
poisson\_test <- predict(poisson\_model, newdata = claim\_number\_test\_set, type = "response")  
nbinomial\_test <- predict(negative\_binomial\_model, newdata = claim\_number\_test\_set,  
 type = "response")  
zero\_test <- predict(zero\_nbinomial\_model, newdata = claim\_number\_test\_scaled,  
 type = "response")  
hurdle\_test <- predict(hurdle\_model, newdata = claim\_number\_test\_scaled,  
 type = "response")  
tweedie\_test <- predict(tweedie\_model, newdata = claim\_number\_test\_scaled,  
 type = "response")  
  
# Calculate Root Mean Squared Error  
poisson\_test\_rmse <- rmse(claim\_number\_test\_set$ClaimNb, poisson\_test)  
nbinomial\_test\_rmse <- rmse(claim\_number\_test\_set$ClaimNb, nbinomial\_test)  
zero\_test\_rmse <- rmse(claim\_number\_test\_scaled$ClaimNb, zero\_test)  
hurdle\_test\_rmse <- rmse(claim\_number\_test\_scaled$ClaimNb, hurdle\_test)  
tweedie\_test\_rmse <- rmse(claim\_number\_test\_scaled$ClaimNb, tweedie\_test)  
  
# Create summary data frame  
rmse\_results <- data.frame(  
 Model = c("Poisson", "Negative Binomial", "Zero-Inflated", "Hurdle", "Tweedie"),  
 RMSE = c(poisson\_test\_rmse, nbinomial\_test\_rmse, zero\_test\_rmse, hurdle\_test\_rmse, tweedie\_test\_rmse)  
)  
  
print(rmse\_results)

Model RMSE  
1 Poisson 0.2286535  
2 Negative Binomial 0.2286540  
3 Zero-Inflated 0.2286551  
4 Hurdle 0.2286565  
5 Tweedie 0.2286645