SWEDISH MOTOR INSURANCE RATING ENGINE

April 24th, 2023

DERIEN WEATHERSPOON, ALEJANDRO DOPP, JACOB OHANIAN, & CAMERON MERRITT

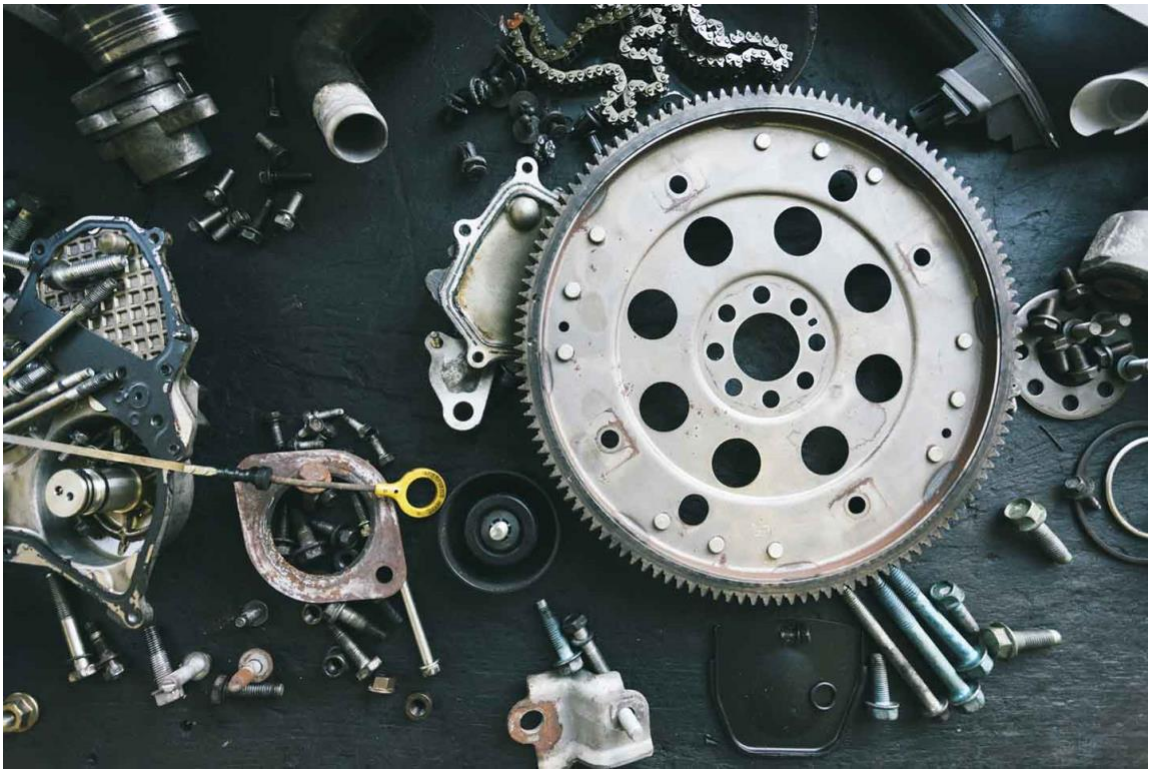


TABLE OF CONTENTS

INTRODUCTION

Actuarial Problem

Construct a rating engine to price Swedish care insurance based on the past Swedish motor insurance data of claim frequency and severities.

Dataset Summary

Data set provided: “SwedishMotorInsurance.csv"

Variables:

- Kilometers: number of kilometers travelled per year range level.
- Zone: geographical zone.
- Bonus: number of years plus one since the last claim.
- Make: vehicle make.
- Insured: number of insured in policy-years.
- Claims: number of claims.
- Payment: payment value.

Assumptions

The number of claims follows a Poisson distribution.

The claim amount follows a gamma distribution.

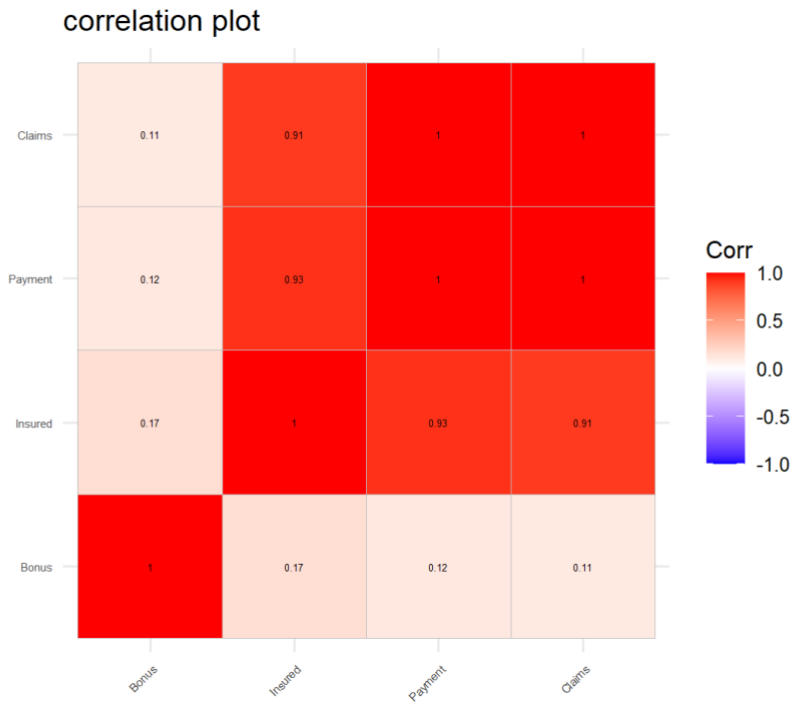
EXECUTIVE SUMMARY

Summary Statistics

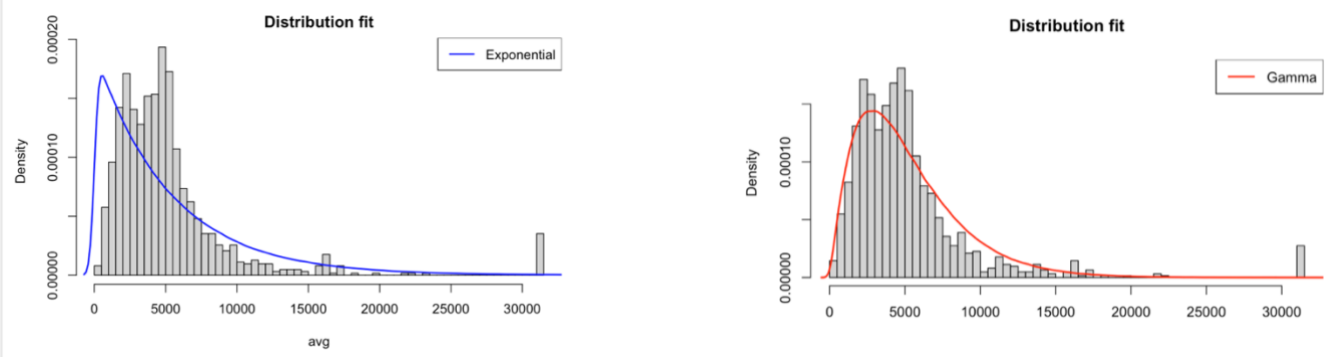
Claim Amount (payment per claim):

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    148    2596    4386    5239    5972   31442
```

Correlation plot between the numeric predictors in the dataset shown below indicates strong positive correlation between Insured and Payment, Insured and Claims, and Claims and Payment.



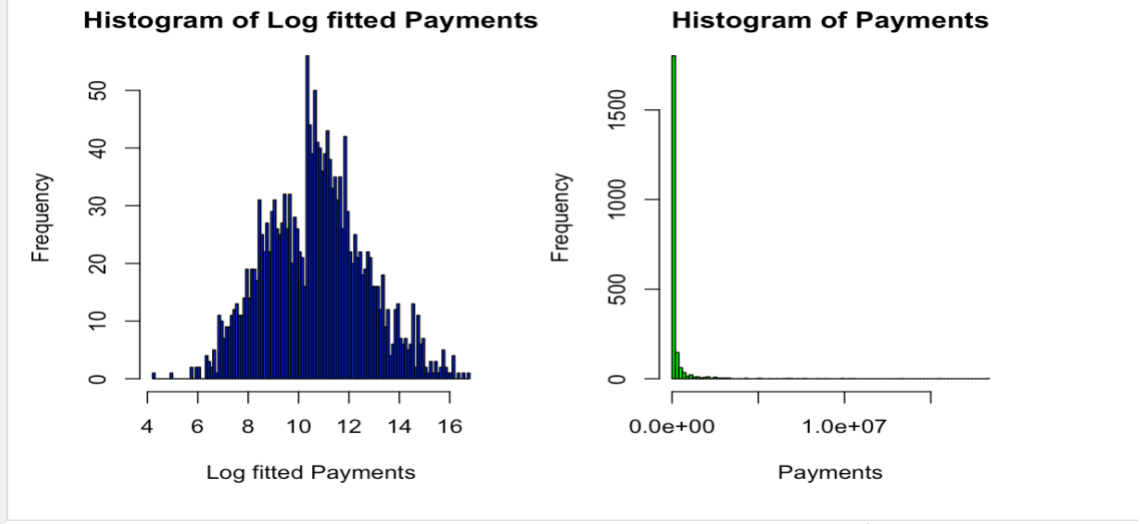
We used wellness of fit to see which distribution of the average payment per claim fit better. As you can see below the average payment per claim fits the gamma distribution better. Although it is slightly subjected to underfitting.



Model Parameters

Histograms for frequency and Payment:

The plot below shows the frequency of a specific payment amount occurs. The plot on the left log transforms the payment amount to help show a more accurate representation of frequency to payment. The plot on the right shows why you need to log transform the data to have a more accurate representation and gives a representation which is easier to understand.



Maximum likelihood estimate:

For the average claim amount: $\hat{\theta} = 5241.378$

Severity Models:

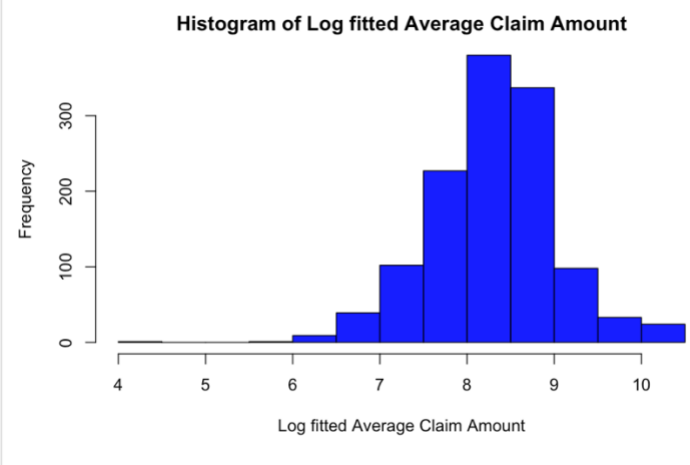
Creating a severity model is useful to predict the average payment per claim. So, in this case for the severity model a Gamma regression model is the most useful to help succeed in getting accurate predictions.

Below is part of a summary for the Severity model using a generalized linear model with both log(Insured) and log(Payment). As can be seen, the log(insured) data is NOT significant to the severity of the claims while log(payment) is significant in showing the severity of the average payment per claim. Therefore, we decided to use the log(payment) severity model.

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   8.64764    0.06885 125.598   <2e-16 ***
log(Insured) -0.01633    0.01258  -1.298    0.194
---
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   6.65239    0.14060   47.31   <2e-16 ***
log(Payment)  0.17542    0.01293   13.56   <2e-16 ***
---
```

Below is a histogram of our severity model showing how often a specific claim amount (log fitted) is being claimed.



Frequency Model:

Creating a frequency model is useful to help predict how often claims are made for a specific policy. For the sake of this frequency model, it will follow a Poisson regression model.

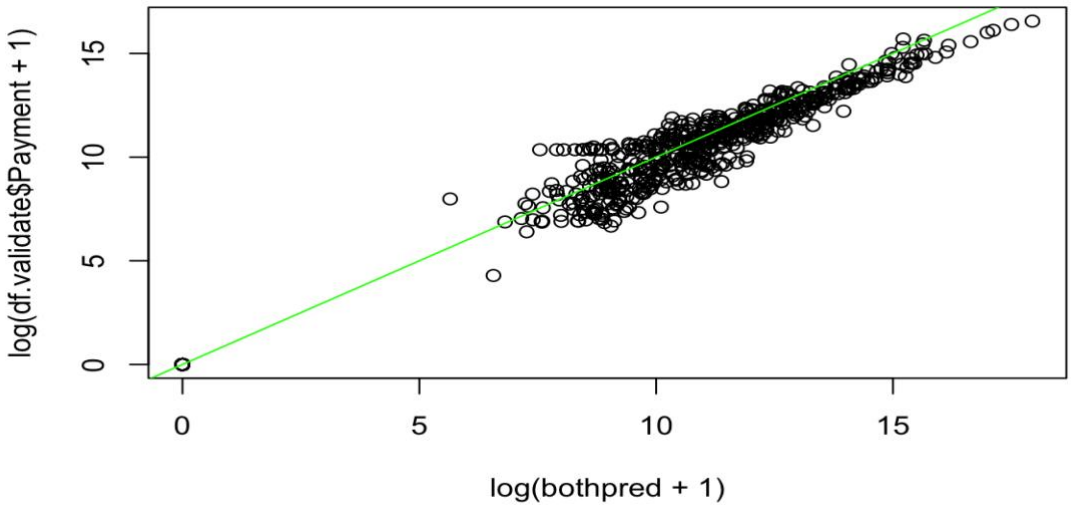
Below is part of the summary for the frequency Poisson model using generalized linear model with log(insured) as the predictor. Compared to the Severity model, log(insured) is extremely significant when fitting to predict claim frequency.

```
Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.940633   0.013645  -142.2   <2e-16 ***
log(Insured)  0.876970   0.001519   577.5   <2e-16 ***
---
```

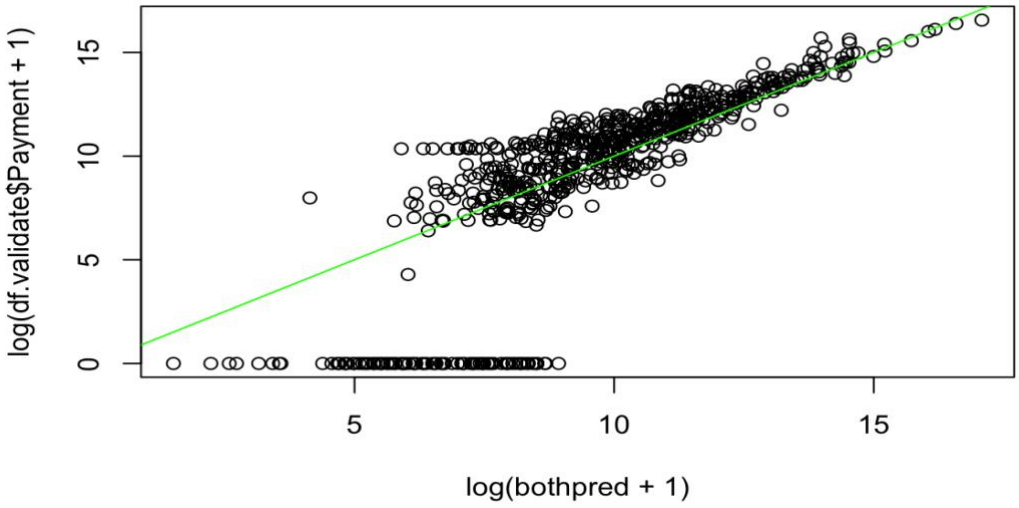
Frequency-Severity Model Pair:

Below a Frequency-Severity Model was created to determine the rate for each policy holder. The plot was created by creating a function “bothpred” which is simply the frequency and severity predictions multiplied together. By doing this we can now log transform the predictors and get the resulting plot below. The plots below show the predictions and how much a payout for each policy will be. All the claims that fall below the green line (which represents the avg rate/ premium) are the profiting claims for the Insurance company while the ones that are above are ones the Insurance company loses money on. Clearly in most cases a company wants to make as much money as possible so, you would expect this said insurance company to raise their premiums to profit more but, in the insurance world of business a consumer wants the lowest possible premium or rate for their policy so if the insurance company were to raise their premiums they would most likely begin to lose business to other competitors with lower premiums.

The plot below is using the bothpred with log(Payment) as part of the severity model



The plot below is using the bothpred with log(insured) as part of the severity model



Standard Errors

For the severity model, we observed a standard error of 0.014060 for the intercept and 0.01293 for the log(payment).

or the frequency model, we observed a standard error of 0.013645 for the intercept and 0.001519 for the log(insured).

These values were acquired using the summary() function on the models that we built. For the severity model, we got an AIC value of 23380, meanwhile the frequency model has an AIC value of 35435. This means that the severity model overall has a better fit, as the lower the AIC the better fit a model is.

- ◊ Summary for the severity of model showing the standard error for the intercept (payment/claims).

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   6.65239    0.14060   47.31   <2e-16 ***
log(Payment)  0.17542    0.01293   13.56   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1

(Dispersion parameter for Gamma family taken to be 0.8484

    Null deviance: 642.57  on 1249  degrees of freedom
Residual deviance: 547.51  on 1248  degrees of freedom
AIC: 23380
```

- ◊ Summary for the frequency model showing the standard error for the intercept (payments/claims).

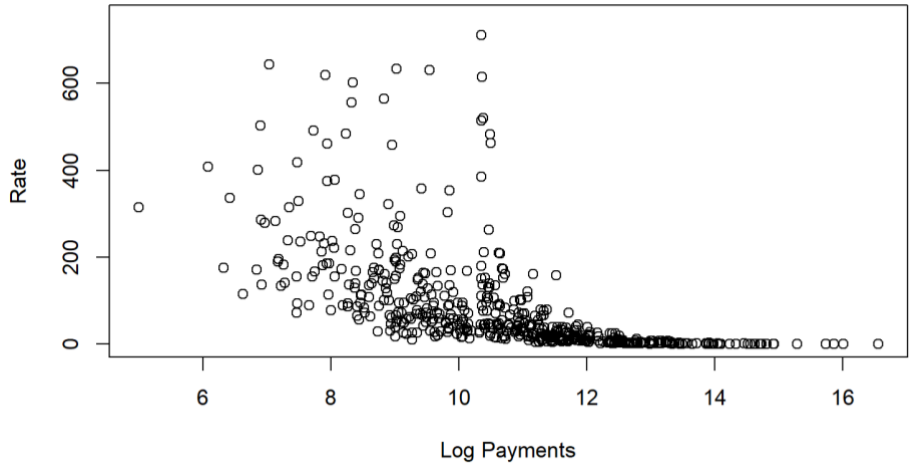
```
Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.940633   0.013645  -142.2   <2e-16 ***
log(Insured)  0.876970   0.001519   577.5   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 435505  on 2181  degrees of freedom
Residual deviance:  27793  on 2180  degrees of freedom
AIC: 35435
```

Prediction Results

As shown in the figure below, the predicted rates seem to converge as Payments increases, to see this relationship, it was necessary to transform Payments by using log(), as the dispersion of Payments was extremely large compared to the dispersion of rates. From this, we can see the best rates being charged to each customer should decrease with the payment amounts made by the customer, to balance the amount between customers. Those who are not paying "enough" will be charged a higher rate to make up for potential losses, and those who pay enough will be charged a small rate.



Looking at the decreasing spread of rates as Payments increase, this can be explained by the effect of number of claims on the charged rates, as the payment size increases, the effect of the number of claims on the rate decreases in strength. This means that when payments are small, the rate depends more on the number of claims, and when payments are large the number of claims affects the rate less and less, which corresponds to the relationship shown in the frequency model.

CONCLUSION

We used the severity and frequency models above to create a dataset which assigns an appropriate premium for prior claims data based on certain cohorts such as zone, kilometers driven per year, years since last claims, and the make of the car. Since we cannot predict the exact number of kilometers a person will drive in the upcoming year, we will use the number of kilometers total on the car divided by the number of years since the car was built.

For each group we used the frequency*severity/the number of individuals to create a rate for each person in that group.

*Note: this was done on a validation set of the data.

Examples…

- 2010 Car in Stockholm with 100,000 Kilometers on it, 0 years since the last claim, and the model of the car is 2 in the data will receive a premium of $453.31.

Inputs	
Zone	Stockholm
Year of car	2010
Kilometers on car	100000
Average Kilometers per year	7692.307692
Years since last claims	0
Make	2
Look Up Codes	
Kilometers level	1
Zone code	1
Bonus	1
Make	2
Premium	**$ 453.31**

- 2015 Car in Rural North Sweden with 100,000 Kilometers on it, 2 years since the last claim, and the model of the car is 9 in the data will receive a premium of $394.30.

Inputs	
Zone	Rural North Sweden
Year of car	2015
Kilometers on car	100000
Average Kilometers per year	12500
Years since last claims	2
Make	9
Look Up Codes	
Kilometers level	5
Zone code	5
Bonus	3
Make	9
Premium	**$ 394.30**

APPENDIX – REFERENCES

https://search.r-project.org/CRAN/refmans/GLMsData/html/motorins.html

APPENDIX – R CODE

```
---
title: "STT 459 Project"
author: "Derien Weatherspoon, Alejandro Dopp, Cameron Merritt, Jacob Ohanian"
date: "2023-04-19"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

Packages

```{r packages, message=FALSE}
library(ggcorrplot)
library(numDeriv)
library(gam)
library(writexl)
```

Splitting the Data

```{r data}
set.seed(1)
df <- read.csv("SwedishMotorInsurance.csv")
train <- sample(nrow(df), 1500, replace = F)
df.train <- df[train,]
df.validate <- df[-train,]
```
```

Summary Stats of Data

Dimensions

```{r dim}
dim(df)
dim(df.train)
dim(df.validate)
num <- nrow(df)
```

Summary

```{r}
summary(df.train)
summary(df.validate)
```

Correlation

```{r corrplot}
numerics <- df[,c("Payment","Claims","Insured","Bonus")]
ggcorrplot(cor(numerics), lab_size = 1.5, tl.cex = 5, lab = T, title = "correlation plot", hc.order = T)
```

Fitting histogram

```{r}
par(mfrow = c(1,2))
hist(log(df$Payment), 100, col = "blue", main = "Histogram of Log fitted Payments", xlab = "Log fitted Payments")
hist(df$Payment, 100, col = "green", main = "Histogram of Payments", xlab = "Payments")
```

Notice with out log applied to Payment it does not properly distribute.

MLE

```{r MLE}

trainsub <- subset(df.train, Claims > 0)
avg <- trainsub$Payment / trainsub$Claims
n <- length(avg)

# MLE for severity model
lik <- function(param) {
  theta <- param[1]
  LogLik <- -log(theta) - avg/theta
  return(-sum(LogLik))
}

op <- nlminb(1, lik, lower=c(0.0000001))
op$par
theta <- op$par[1]
negativeHessian <- hessian(lik,theta)
sd <- sqrt(solve(negativeHessian))
c(theta-sd*1.96 , theta+sd*1.96)
sd

# See how good the fit is.
hist(avg,100,freq=FALSE, main="Distribution fit")
lines(density(rexp(10^6,rate=1/theta)),col="blue",lwd=2)
legend("topright", c("Exponential"),
       col=c("blue"), lwd=c(2,2))


severity_model_mle <-glm(avg ~ 1, data = trainsub, family = Gamma(link = "log"))
exp(severity_model_mle$coefficients)
```


```{r}
# Try fitting a gamma distribution to get a better fit.
lik2 <- function(param) {
  shape <- param[1]
  scale <- param[2]
```

```
  LogLik <- dgamma(avg, shape=shape, scale=scale, log=TRUE)
  return(-sum(LogLik))
}
op2 <- nlminb(c(1,1), lik2, lower=c(0.0000001,0.0000001))

op2$par

# See how good the fit is for the gamma.
hist(avg,100,freq=FALSE, main="Distribution fit")
lines(density(rgamma(10^6, shape=op2$par[1],
                     scale=op2$par[2])),col="red",lwd=2)
legend("topright", c("Gamma"),
       col=c("red"), lwd=c(2,2))
```

Building the severity model

```{r}
hist(log(avg), col = "blue", main = "Histogram of Log fitted Average Claim Amount", xlab = "Log fitted Average Claim Amount")
hist(avg, col = "dark green", main = "Histogram of Average Claim Amount", xlab = "Average Claim Amount")
severity_model <- glm(avg ~ log(Payment), data = trainsub, family = Gamma(link = "log"))
summary(severity_model)
severity_model_mle <-glm(avg ~ 1, data = trainsub, family = Gamma(link = "log"))

df.validate$sevpred <- exp(severity_model$coefficients[1] + severity_model$coefficients[2] * log(df.validate$Payment))
sevpred2 <- predict(severity_model, newdata = df.validate, type = "response")
severity_model$aic

severity_model_mle <-glm(avg ~ 1, data = trainsub, family = Gamma(link = "log"))
exp(severity_model_mle$coefficients[1]) # using MLE


severity_model_test <- severity_model <- glm(avg ~ log(Insured), data = trainsub, family = Gamma(link = "log"))
```

Building the frequency model

```{r frequency}

freqglm <- glm(Claims ~ log(Insured), data = df, family = poisson(link = "log"))

summary(freqglm)


df.validate$freqpred <- predict(freqglm, newdata = df.validate, type = "response")

plot(log(freqpred+1),log(df.validate$Claims+1))

abline(0,1, col = "red")


# Estimate the standard errors.

#I <- hessian(lik, op$par)

#sd <- sqrt(diag(solve(I)))


```

Everything above the line is the insurance company losing money, whereas everything below is when the company is making profit.

Frequency-Severity model


```{r freq-sev model}

df.validate$bothpred <- df.validate$freqpred * df.validate$sevpred


plot(log(df.validate$bothpred+1), log(df.validate$Payment+1))

abline(0,1, col = "green")

head(bothpred)


```

```{r rate}

freqglm <- glm(Claims ~ log(Insured), data = df, family = poisson(link = "log"))

severity_model <- glm(avg ~ log(Insured), data = trainsub, family = Gamma(link = "log"))

df.validate$validated_freq <- predict(freqglm, newdata = df.validate, type = "response")

df.validate$validated_sev <- predict(severity_model, newdata = df.validate, type = "response")

df.validate$validated_rate <- (df.validate$validated_freq * df.validate$validated_sev) / df.validate$Insured

df.validate

```

```{r}

write_xlsx(df.validate,"rating engine final v2.xlsx")

```