Detail Descriptions

**The goal of this case study is to work towards building a model to predict the total number of claims a customer is going to file with the company.**

#memory clear  
rm(list = ls())   
  
# Loading Packages  
library("stats") # To get all statistical function in r  
library("arm") # Data Analysis Using Regression and Multilevel/Hierarchical Models

library("jtools")

library("broom") # For tidy data frames  
library("ggstance") # horizontal versions of common ggplot2 Geoms etc.

library("magrittr") # Offers set of operators to improve the code  
library("reshape2") # ransform data between wide and long formats  
library("ggplot2") # For proper visualizations

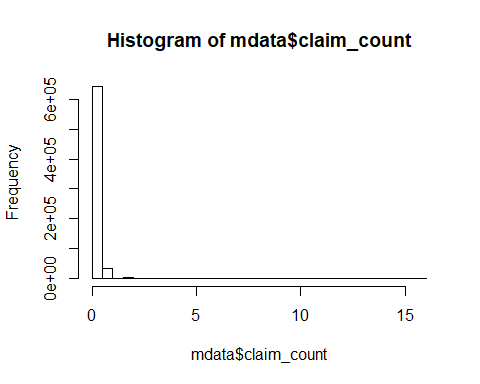
library("ggcorrplot") # Visualize correlation matrix

# To import out dataset into r  
mdata.input <- read.csv2("Data.csv", sep = ",", check.names = F, dec = '.')  
mdata <- mdata.input

### Descriptive statistics  
ls.str(mdata) # To identify the type of the variables

## cat\_areacode : Factor w/ 6 levels "A","B","C","D",..: 4 4 2 2 2 5 5 3 3 2 ...  
## cat\_carBrand : Factor w/ 11 levels "B1","B10","B11",..: 4 4 4 4 4 4 4 4 4 4 ...  
## cat\_fuelType : Factor w/ 4 levels "Diesel","Electric",..: 4 4 1 1 1 4 4 1 1 1 ...  
## cat\_Region : Factor w/ 22 levels "R11","R21","R22",..: 18 18 3 15 15 8 8 20 20 12 ...  
## claim\_count : int [1:678013] 1 1 1 1 1 1 1 1 1 1 ...  
## ï»¿policy\_desc : int [1:678013] 1 3 5 10 11 13 15 17 18 21 ...  
## num\_driverAge : int [1:678013] 55 55 52 46 46 38 38 33 33 41 ...  
## num\_exposure : Factor w/ 184 levels "0.00273224","0.002739726",..: 16 83 81 15 90 58 51 33 77 21 ...  
## num\_noClaimDiscountPercent : int [1:678013] 50 50 50 50 50 50 50 68 68 50 ...  
## num\_populationDensitykmsq : int [1:678013] 1217 1217 54 76 76 3003 3003 137 137 60 ...  
## num\_vehicleAge : int [1:678013] 0 0 2 0 0 2 2 0 0 0 ...  
## ord\_vehicleHP : int [1:678013] 5 5 6 7 7 6 6 7 7 7 ...

hist(mdata$claim\_count, breaks = 50) # Histogram plot of Claim\_count



mean(mdata$claim\_count)

## [1] 0.05324677

var(mdata$claim\_count)

## [1] 0.05765633

table(mdata$claim\_count) # Unique count of total claims

##   
## 0 1 2 3 4 5 6 8 9 11   
## 643953 32178 1784 82 7 2 1 1 1 3   
## 16   
## 1

### univariate analysis -- it is the simplest form of analysing data and it's major purpose is to describe- it takes data, summarizes that data and finds patterns in the data.

library(psych)

describe(mdata)

## vars n mean sd median  
## ï»¿policy\_desc 1 678013 2621856.92 1641782.75 2272152  
## claim\_count 2 678013 0.05 0.24 0  
## cat\_areacode\* 3 678013 3.29 1.38 3  
## num\_vehicleAge 4 678013 7.04 5.67 6  
## num\_noClaimDiscountPercent 5 678013 59.76 15.64 50  
## cat\_carBrand\* 6 678013 5.05 3.07 4  
## num\_populationDensitykmsq 7 678013 1792.42 3958.65 393  
## cat\_Region\* 8 678013 11.29 6.87 12  
## ord\_vehicleHP 9 678013 6.95 64.91 6  
## num\_exposure\* 10 678013 58.85 36.51 55  
## cat\_fuelType\* 11 678013 2.53 1.50 4  
## num\_driverAge 12 677975 45.50 14.14 44  
## trimmed mad min max range  
## ï»¿policy\_desc 2565398.59 1708047.12 1 6114330 6114329  
## claim\_count 0.00 0.00 0 16 16  
## cat\_areacode\* 3.33 1.48 1 6 5  
## num\_vehicleAge 6.56 5.93 0 100 100  
## num\_noClaimDiscountPercent 56.34 0.00 50 230 180  
## cat\_carBrand\* 4.89 4.45 1 11 10  
## num\_populationDensitykmsq 921.94 526.32 1 27000 26999  
## cat\_Region\* 11.36 10.38 1 22 21  
## ord\_vehicleHP 6.19 1.48 4 9999 9995  
## num\_exposure\* 59.10 54.86 1 184 183  
## cat\_fuelType\* 2.54 0.00 1 4 3  
## num\_driverAge 44.73 14.83 18 100 82  
## skew kurtosis se  
## ï»¿policy\_desc 0.24 -0.66 1993.87  
## claim\_count 5.60 76.84 0.00  
## cat\_areacode\* -0.18 -0.88 0.00  
## num\_vehicleAge 1.15 6.52 0.01  
## num\_noClaimDiscountPercent 1.73 2.67 0.02  
## cat\_carBrand\* 0.14 -1.10 0.00  
## num\_populationDensitykmsq 4.65 24.87 4.81  
## cat\_Region\* 0.02 -1.46 0.01  
## ord\_vehicleHP 150.98 23193.95 0.08  
## num\_exposure\* 0.08 -1.51 0.04  
## cat\_fuelType\* -0.04 -2.00 0.00  
## num\_driverAge 0.44 -0.34 0.02

### Data preparation/cleaning  
  
# Remove 'policy\_desc' column  
mdata <- mdata[,!(names(mdata) %in% c('policy\_desc'))]  
  
# Renaming variables for our covenient purpose  
names(mdata)[names(mdata) == "ï»¿policy\_desc"] <- "policy\_desc"  
names(mdata)[names(mdata) == "cat\_areacode"] <- "areacode"  
names(mdata)[names(mdata) == "num\_vehicleAge"] <- "vehicle\_age"  
names(mdata)[names(mdata) == "num\_noClaimDiscountPercent"] <- "discount"  
names(mdata)[names(mdata) == "cat\_carBrand"] <- "car\_brand"  
names(mdata)[names(mdata) == "num\_populationDensitykmsq"] <- "population\_density"  
names(mdata)[names(mdata) == "cat\_Region"] <- "region"  
names(mdata)[names(mdata) == "ord\_vehicleHP"] <- "vehicle\_hp"  
names(mdata)[names(mdata) == "num\_exposure"] <- "exposure"  
names(mdata)[names(mdata) == "cat\_fuelType"] <- "fuel\_type"  
names(mdata)[names(mdata) == "num\_driverAge"] <- "driver\_age"  
  
# Change type of variable 'exposure'  
mdata$exposure <- suppressWarnings(as.numeric(as.character(mdata$exposure)))  
  
# Remove rows with missing values  
any(!complete.cases(mdata))

## [1] TRUE

mdata <- mdata[complete.cases(mdata), ]  
  
# Remove rows where variable 'fuel\_type' is NULL  
mdata <- mdata[-which(mdata[,'fuel\_type'] == 'NULL'), ]

###################### CORRELATION ###########################  
  
# Here we are using spearman rank correlation to find the relationship between within variables. Below are description of spearman correlation like why, when we can use this and what are assumptions in this correlation.  
  
# When should you use the Spearman's rank-order correlation?  
  
# The Spearman's rank-order correlation is the nonparametric version of the Pearson product-moment correlation. Spearman's correlation coefficient, (??, also signified by rs) measures the strength and direction of association between two ranked variables.  
  
# What are the assumptions of the test?  
  
# You need two variables that are either ordinal(variables that have two or more categories like "Type of property""), interval or ratio(Continuous variables can be further categorized as either interval or ratio variables. Ex. temperature measured in degrees Celsius or Fahrenheit, temperature measured in Kelvin)  
  
# What is a monotonic relationship?  
  
# A monotonic relationship is a relationship that does one of the following: (1) as the value of one variable increases, so does the value of the other variable; or (2) as the value of one variable increases, the other variable value decreases.  
  
# Why is a monotonic relationship important to Spearman's correlation?  
  
# Spearman's correlation measures the strength and direction of monotonic association between two variables. Monotonicity is "less restrictive" than that of a linear relationship.  
  
cor.names <- c("spearman")  
  
cor.df <- mdata[,!(names(mdata) %in% c('policy\_desc'))]  
cor.df <- lapply(cor.df, as.numeric) %>% data.frame()  
  
# calculating correlation  
  
for (i in cor.names) {  
 all.cor <- cor(cor.df, use = 'complete.obs', method = i)  
 assign(paste0("cor.", i), melt(cor(cor.df, use = 'complete.obs'), variable.factor=FALSE))  
}  
  
  
################## Correlogram #########################  
  
# What is a Correlation Matrix?  
  
# A correlation matrix is a table showing correlation coefficients between sets of variables. Each random variable (Xi) in the table is correlated with each of the other values in the table (Xj). This allows you to see which pairs have the highest correlation.  
  
# What Kind of Data Can I Compare?  
  
# The correlation matrix is simply a table of correlations. The most common correlation coefficient is Pearson's correlation coefficient, which compares two interval variables or ratio variables. But there are many others, depending on the type of data you want to correlate.   
  
corr <- round(all.cor, 2)  
  
pdf(file="Correlogram.pdf",width = 16, height = 12)  
corr.plot1 <- ggcorrplot(corr, hc.order = TRUE,   
 type = "lower",   
 lab = TRUE,   
 lab\_size = 3,   
 method="square",   
 colors = c("tomato2", "white", "springgreen3"),   
 title="Correlogram",   
 ggtheme=theme\_bw)  
print(corr.plot1)   
dev.off()

## png   
## 2

# Here the correlation matrix has been plotted using all the variables  
# From the graph, It is seen that Area code and population density have highest correlation which is equals to 0.98  
  
# Also driver age and discount have negative correlation which id equals to 0.57

################## MODELLING ##########################  
  
# Here we are considering these four below mentioned statistics for modelling.  
  
# RSE -- In statistics, a relative standard error (RSE) is equal to the standard error of a survey estimate divided by the survey estimate and then multiplied by 100. The number is multiplied by 100 so it can be expressed as a percentage. The RSE does not necessarily represent any new information beyond the standard error, but it might be a superior method of presenting statistical confidence.  
  
# Relative Standard Error vs. Standard Error  
# Standard error measures how much a survey estimate is likely to deviate from the actual population. It is expressed as a number. By contrast, relative standard error (RSE) is the standard error expressed as a fraction of the estimate and is usually displayed as a percentage. Estimates with a RSE of 25% or greater are subject to high sampling error and should be used with caution.  
  
# What Is the Adjusted R-squared?  
  
# The adjusted R-squared compares the explanatory power of regression models that contain different numbers of predictors.  
  
# Suppose you compare a five-predictor model with a higher R-squared to a one-predictor model. Does the five predictor model have a higher R-squared because it's better? Or is the R-squared higher because it has more predictors? Simply compare the adjusted R-squared values to find out!  
  
# The adjusted R-squared is a modified version of R-squared that has been adjusted for the number of predictors in the model. The adjusted R-squared increases only if the new term improves the model more than would be expected by chance. It decreases when a predictor improves the model by less than expected by chance. The adjusted R-squared can be negative, but it's usually not. It is always lower than the R-squared.  
  
# What are F-statistics and the F-test?  
  
# F-tests are named after its test statistic, F, which was named in honor of Sir Ronald Fisher. The F-statistic is simply a ratio of two variances. Variances are a measure of dispersion, or how far the data are scattered from the mean. Larger values represent greater dispersion.

model.stat <- matrix(0, ncol = 4, nrow = 1) %>% data.frame()  
names(model.stat) <- c('RSE', 'Adjusted R-squared', 'F-Statistic', 'any-aliased')  
  
# convert areacode character variable to integer variable to pass in model  
mdata$areacode <- factor(mdata$areacode, levels=c("A","B","C","D","E","F"), labels=c(1,2,3,4,5,6))  
mdata$areacode=as.integer(as.character(mdata$areacode ))  
  
# convert fuel\_type character variable to integer variable to pass in model  
mdata$fuel\_type <- factor(mdata$fuel\_type, levels=c("NULL","Diesel", "Regular", "Electric"), labels=c(0,1,2,3))  
mdata$fuel\_type=as.integer(as.character(mdata$fuel\_type))

########### Sampling ############  
  
# Sampling (0.7 , 0.3)   
  
# Why do we set seed in R?  
  
# Set the seed of R's random number generator, which is useful for creating simulations or random objects that can be reproduced. The random numbers are the same, and they would continue to be the same no matter how far out in the sequence we went  
  
set.seed(1234)  
ind<- sample(2, nrow(mdata), replace=TRUE, prob=c(0.8,0.2))  
  
# Divinng dataset into trian and test   
trainData=mdata[ind==1,]  
testData=mdata[ind==2,]

########################### LINEAR MODEL ##############################  
  
# LIBEAR REGRESSION: - In statistical modeling, it is used to estimate real world values (price of items, height and weight etc.) based on continuous variable(s). Here, we want to establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation Y= a \*X + b.  
  
# Technically, a regression analysis model is based on the sum of squares, which is a mathematical way to find the dispersion of data points. The goal of a model is to get the smallest possible sum of squares and draw a line that comes closest to the data.  
# In this equation:  
# Y - Dependent Variable a - Slope X - Independent variable b - Intercept  
# These coefficients a and b are derived based on minimizing the sum of squared difference of distance between data points and regression line.  
  
# In statistics, they differentiate between a simple and multiple linear regressions. Simple Linear Regression models the relationship between a dependent variable and one independent variables using a linear function. If you use two or more explanatory variables to predict the independent variable, you deal with multiple linear regression. If the dependent variables are modeled as a non-linear function because the data relationships do not follow a straight line, use non-linear regression instead. While finding best fit line, you can fit a polynomial or curvilinear regression. And these are known as polynomial or curvilinear regression.  
  
# Assumptions of Linear Rgression  
  
# First, linear regression needs the relationship between the independent and dependent variables to be linear. It is also important to check for outliers since linear regression is sensitive to outlier effects. The linearity assumption can best be tested with scatter plots, the following two examples depict two cases, where no and little linearity is present.  
  
# Secondly, the linear regression analysis requires all variables to be multivariate normal. This assumption can best be checked with a histogram or a Q-Q-Plot. Normality can be checked with a goodness of fit test, e.g., the Kolmogorov-Smirnov test.  
  
# Thirdly, linear regression assumes that there is little or no multicollinearity in the data. Multicollinearity occurs when the independent variables are too highly correlated with each other.

# Multicollinearity may be tested with three central criteria:  
  
# 1) Correlation matrix - when computing the matrix of Pearson's Bivariate Correlation among all independent variables the correlation coefficients need to be smaller than 1.  
  
# 2) Tolerance - the tolerance measures the influence of one independent variable on all other independent variables; the tolerance is calculated with an initial linear regression analysis. Tolerance is defined as T = 1 - R² for these first step regression analysis. With T < 0.1 there might be multicollinearity in the data and with T < 0.01 there certainly is.  
  
# 3) Variance Inflation Factor (VIF) - the variance inflation factor of the linear regression is defined as VIF = 1/T. With VIF > 10 there is an indication that multicollinearity may be present; with VIF > 100 there is certainly multicollinearity among the variables.  
  
# Why do we use Linear Regression?  
  
# Simple linear regression is useful for finding relationship between two continuous variables. One is predictor or independent variable and other is response or dependent variable. ... The best fit line is the one for which total prediction error (all data points) are as small as possible.  
  
# MODEL 1  
measurevar <- "claim\_count"  
lm.regressors1 <- setdiff(names(trainData), c('claim\_count'))  
  
# These are the necessary independent variables for modelling  
f1 <- as.formula(paste(measurevar, paste(lm.regressors1, collapse=" + "), sep=" ~ "))  
#as.formula(paste(measurevar, paste(groupvars, collapse=" + "), sep=" ~ "))  
fit1 <- lm(f1, data = trainData)  
summary(fit1)

##   
## Call:  
## lm(formula = f1, data = trainData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.2763 -0.0788 -0.0471 -0.0162 15.9751   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -2.852e-02 3.020e-03 -9.445 < 2e-16 \*\*\*  
## policy\_desc -2.225e-08 2.194e-10 -101.405 < 2e-16 \*\*\*  
## areacode 2.510e-03 3.144e-04 7.981 1.45e-15 \*\*\*  
## vehicle\_age -1.306e-03 6.548e-05 -19.948 < 2e-16 \*\*\*  
## discount 1.339e-03 2.394e-05 55.931 < 2e-16 \*\*\*  
## car\_brandB10 2.815e-03 2.104e-03 1.338 0.180802   
## car\_brandB11 7.230e-03 2.366e-03 3.055 0.002248 \*\*   
## car\_brandB12 3.625e-02 1.121e-03 32.341 < 2e-16 \*\*\*  
## car\_brandB13 3.166e-03 2.480e-03 1.276 0.201783   
## car\_brandB14 -1.269e-02 4.223e-03 -3.005 0.002656 \*\*   
## car\_brandB2 -8.257e-04 9.346e-04 -0.883 0.377004   
## car\_brandB3 1.908e-03 1.332e-03 1.433 0.151904   
## car\_brandB4 -8.511e-04 1.801e-03 -0.473 0.636525   
## car\_brandB5 2.478e-03 1.567e-03 1.582 0.113737   
## car\_brandB6 -6.635e-04 1.708e-03 -0.389 0.697626   
## population\_density -2.454e-07 1.128e-07 -2.176 0.029592 \*   
## regionR21 4.028e-03 4.998e-03 0.806 0.420269   
## regionR22 -5.293e-04 3.212e-03 -0.165 0.869134   
## regionR23 -1.502e-02 3.085e-03 -4.868 1.13e-06 \*\*\*  
## regionR24 -3.577e-03 1.518e-03 -2.356 0.018488 \*   
## regionR25 -4.487e-03 2.847e-03 -1.576 0.115048   
## regionR26 -5.852e-03 2.887e-03 -2.027 0.042648 \*   
## regionR31 -1.049e-02 2.008e-03 -5.227 1.73e-07 \*\*\*  
## regionR41 -2.052e-02 2.640e-03 -7.772 7.76e-15 \*\*\*  
## regionR42 -4.936e-03 5.773e-03 -0.855 0.392486   
## regionR43 -8.206e-03 7.322e-03 -1.121 0.262420   
## regionR52 -1.068e-02 1.853e-03 -5.762 8.32e-09 \*\*\*  
## regionR53 -1.421e-03 1.844e-03 -0.771 0.440821   
## regionR54 -1.155e-02 2.327e-03 -4.964 6.89e-07 \*\*\*  
## regionR72 -1.061e-02 1.937e-03 -5.478 4.30e-08 \*\*\*  
## regionR73 -1.184e-02 2.368e-03 -4.999 5.75e-07 \*\*\*  
## regionR74 7.843e-04 4.116e-03 0.191 0.848900   
## regionR82 -1.139e-03 1.500e-03 -0.759 0.447874   
## regionR83 -1.393e-02 3.865e-03 -3.604 0.000313 \*\*\*  
## regionR91 -1.002e-02 1.883e-03 -5.321 1.03e-07 \*\*\*  
## regionR93 -6.776e-03 1.528e-03 -4.435 9.20e-06 \*\*\*  
## regionR94 1.083e-02 4.160e-03 2.603 0.009230 \*\*   
## vehicle\_hp -1.427e-07 4.885e-06 -0.029 0.976697   
## exposure 4.841e-02 9.505e-04 50.934 < 2e-16 \*\*\*  
## fuel\_type -1.459e-03 6.699e-04 -2.178 0.029398 \*   
## driver\_age 7.533e-04 2.654e-05 28.388 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2369 on 542368 degrees of freedom  
## Multiple R-squared: 0.0307, Adjusted R-squared: 0.03062   
## F-statistic: 429.4 on 40 and 542368 DF, p-value: < 2.2e-16

model.stat.fit1 <- model.stat  
model.stat.fit1$RSE <- summary(fit1)$sigma  
model.stat.fit1$`Adjusted R-squared` <- summary(fit1)$adj.r.squared  
model.stat.fit1$`F-Statistic` <- summary(fit1)$fstatistic[1]  
model.stat.fit1$`any-aliased` <- any(summary(fit1)$aliased)  
  
# MODEL 2  
lm.regressors2 <- setdiff(names(trainData), c('claim\_count', 'region', 'car\_brand', 'vehicle\_hp'))  
f2 <- as.formula(paste("claim\_count ~ ", paste(lm.regressors2, collapse=" + ")))  
fit2 <- lm(f2, data = trainData)  
summary(fit2) # higher F-Statistic shows stronger relashionship between exogenic and endogenic variables

##   
## Call:  
## lm(formula = f2, data = trainData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.2747 -0.0777 -0.0484 -0.0185 15.9658   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -2.985e-02 2.601e-03 -11.477 < 2e-16 \*\*\*  
## policy\_desc -1.899e-08 2.017e-10 -94.129 < 2e-16 \*\*\*  
## areacode 2.225e-03 2.928e-04 7.598 3.01e-14 \*\*\*  
## vehicle\_age -2.347e-03 5.919e-05 -39.650 < 2e-16 \*\*\*  
## discount 1.333e-03 2.393e-05 55.688 < 2e-16 \*\*\*  
## population\_density 2.885e-07 1.012e-07 2.852 0.00435 \*\*   
## exposure 4.435e-02 9.308e-04 47.651 < 2e-16 \*\*\*  
## fuel\_type 1.651e-03 6.609e-04 2.498 0.01250 \*   
## driver\_age 7.903e-04 2.637e-05 29.971 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2373 on 542400 degrees of freedom  
## Multiple R-squared: 0.02765, Adjusted R-squared: 0.02764   
## F-statistic: 1928 on 8 and 542400 DF, p-value: < 2.2e-16

model.stat.fit2 <- model.stat  
model.stat.fit2$RSE <- summary(fit2)$sigma  
model.stat.fit2$`Adjusted R-squared` <- summary(fit2)$adj.r.squared  
model.stat.fit2$`F-Statistic` <- summary(fit2)$fstatistic[1]  
model.stat.fit2$`any-aliased` <- any(summary(fit2)$aliased)  
  
# Interpretation  
  
# From the above table we can see that the value of Multiple R-squared = 0.03047  
# R-squared value measures the proportion of the variation in the dependent variable explained by all of the independent variables in the model. Thus a high R-squared values signifies that the model is agood fit.  
  
# From the table we can also see that the value of Adjusted R-squared = 0.03039   
# Adjusted R-squared value measures the proportion of variation explained by only those independent variables that really affect the dependent variable. It penalizes you for adding independent variable that does not affect the dependent variable.  
# Thus again higher the value of Adjusted R-squared the better is the model.  
  
# Thep-value< 2.2e-16(which is p-value<0.05, it means that the null hypothesis is rejected and the alternative hypothesis gets accepted).Herethe null hypothesis that is the dependent and independent variables are not linearly related is rejected.  
  
# Residual standard error = 0.237, it explains how close the actual data points are to the model's predicted values. It measures standard deviation of the residuals.  
  
############### See Predicted Value  
pred = predict(fit1,testData)  
  
#See Actual vs. Predicted Value  
finaldata = cbind(testData,pred)  
print(head(subset(finaldata, select = c(claim\_count,pred))))

## claim\_count pred  
## 5 1 0.1429355  
## 14 1 0.1473538  
## 16 1 0.1646224  
## 26 1 0.1823947  
## 28 1 0.1150582  
## 29 1 0.1518524

#Calculating RMSE  
rmse <- sqrt(mean((testData$claim\_count - pred)^2))  
print(rmse)

## [1] 0.2344711

#check accuracy  
library(forecast)  
accuracy(fit1)

## ME RMSE MAE MPE MAPE MASE  
## Training set -1.315382e-16 0.2368851 0.1009463 NaN Inf 0.9985425

######################### Poisson regression #######################  
  
# Why would you do a Poisson regression?  
  
# Poisson regression, also known as a log-linear model, is what you use when your outcome variable is a count (i.e., numeric, but not quite so wide in range as a continuous variable.) Examples of count variables in research include how many heart attacks or strokes one's had, how many days in the past month one's used.he Poisson distribution is unique in that its mean and its variance are equal. This is often due to zero inflation.  
  
# ASSUMPTIONS  
  
# 1. Your dependent variable consists of count data. Count data is different to the data measured in other well-known types of regression.  
  
#2. You have one or more independent variables, which can be measured on a continuous, ordinal or nominal/dichotomous scale. Ordinal and nominal/dichotomous variables can be broadly classified as categorical variables.   
  
# 3. You should have independence of observations. This means that each observation is independent of the other observations; that is, one observation cannot provide any information on another observation.   
  
# 4. The distribution of counts (conditional on the model) follow a Poisson distribution. One consequence of this is that the observed and expected counts should be equal   
  
# 5. The mean and variance of the model are identical. This is a consequence of Assumption #4; that there is a Poisson distribution. For a Poisson distribution the variance has the same value as the mean. If you satisfy this assumption you have equidispersion. However, often this is not the case and your data is either under- or overdispersed with overdispersion the more common problem. There are a variety of methods that you can use to assess overdispersion. One method is to assess the Pearson dispersion statistic.

# MODEL 3  
set.seed(3464)  
pr.regressors3 <- setdiff(names(trainData), c('claim\_count'))  
f3 <- as.formula(paste("claim\_count ~ ", paste(pr.regressors3, collapse=" + ")))  
fit3 <- glm(f3, trainData, family = poisson(link = "log"))  
summary(fit3)

##   
## Call:  
## glm(formula = f3, family = poisson(link = "log"), data = trainData)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6883 -0.3569 -0.2639 -0.1978 13.2496   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -4.359e+00 5.196e-02 -83.893 < 2e-16 \*\*\*  
## policy\_desc -4.245e-07 4.283e-09 -99.109 < 2e-16 \*\*\*  
## areacode 5.447e-02 5.814e-03 9.368 < 2e-16 \*\*\*  
## vehicle\_age -2.533e-02 1.268e-03 -19.977 < 2e-16 \*\*\*  
## discount 2.031e-02 3.491e-04 58.172 < 2e-16 \*\*\*  
## car\_brandB10 5.996e-02 3.961e-02 1.514 0.130085   
## car\_brandB11 1.516e-01 4.289e-02 3.534 0.000409 \*\*\*  
## car\_brandB12 5.912e-01 1.998e-02 29.593 < 2e-16 \*\*\*  
## car\_brandB13 4.356e-02 4.559e-02 0.956 0.339316   
## car\_brandB14 -3.102e-01 9.175e-02 -3.380 0.000724 \*\*\*  
## car\_brandB2 -1.579e-02 1.707e-02 -0.925 0.354921   
## car\_brandB3 3.111e-02 2.439e-02 1.275 0.202175   
## car\_brandB4 -1.914e-02 3.336e-02 -0.574 0.566040   
## car\_brandB5 3.811e-02 2.781e-02 1.370 0.170579   
## car\_brandB6 -1.406e-02 3.133e-02 -0.449 0.653619   
## population\_density -5.016e-06 1.945e-06 -2.579 0.009921 \*\*   
## regionR21 9.842e-02 9.088e-02 1.083 0.278807   
## regionR22 -2.562e-03 5.843e-02 -0.044 0.965024   
## regionR23 -3.098e-01 6.936e-02 -4.466 7.96e-06 \*\*\*  
## regionR24 -4.054e-02 2.712e-02 -1.494 0.135054   
## regionR25 -6.689e-02 4.970e-02 -1.346 0.178316   
## regionR26 -8.853e-02 5.491e-02 -1.612 0.106912   
## regionR31 -1.975e-01 3.942e-02 -5.011 5.41e-07 \*\*\*  
## regionR41 -3.656e-01 5.094e-02 -7.177 7.13e-13 \*\*\*  
## regionR42 -1.008e-01 9.906e-02 -1.018 0.308697   
## regionR43 -1.983e-01 1.525e-01 -1.300 0.193537   
## regionR52 -1.836e-01 3.391e-02 -5.415 6.13e-08 \*\*\*  
## regionR53 -2.998e-02 3.198e-02 -0.938 0.348493   
## regionR54 -1.924e-01 4.290e-02 -4.485 7.29e-06 \*\*\*  
## regionR72 -1.885e-01 3.733e-02 -5.049 4.44e-07 \*\*\*  
## regionR73 -2.510e-01 4.869e-02 -5.154 2.55e-07 \*\*\*  
## regionR74 2.802e-02 7.337e-02 0.382 0.702502   
## regionR82 -2.153e-02 2.645e-02 -0.814 0.415578   
## regionR83 -3.080e-01 8.749e-02 -3.521 0.000431 \*\*\*  
## regionR91 -1.610e-01 3.616e-02 -4.453 8.46e-06 \*\*\*  
## regionR93 -1.028e-01 2.805e-02 -3.664 0.000248 \*\*\*  
## regionR94 1.795e-01 7.510e-02 2.390 0.016829 \*   
## vehicle\_hp -3.691e-07 9.484e-05 -0.004 0.996895   
## exposure 9.334e-01 1.813e-02 51.471 < 2e-16 \*\*\*  
## fuel\_type -3.740e-02 1.231e-02 -3.040 0.002368 \*\*   
## driver\_age 1.157e-02 4.448e-04 26.007 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for poisson family taken to be 1)  
##   
## Null deviance: 174181 on 542408 degrees of freedom  
## Residual deviance: 156419 on 542368 degrees of freedom  
## AIC: 211892  
##   
## Number of Fisher Scoring iterations: 6

# MODEL 4  
set.seed(3423)  
pr.regressors4 <- setdiff(names(trainData), c('claim\_count', 'region', 'car\_brand', 'vehicle\_hp', 'fuel\_type'))  
f4 <- as.formula(paste("claim\_count ~ ", paste(pr.regressors4, collapse=" + ")))  
fit4 <- glm(f4, trainData, family = quasipoisson(link = "log"))  
summary(fit4)

##   
## Call:  
## glm(formula = f4, family = quasipoisson(link = "log"), data = trainData)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.7717 -0.3591 -0.2694 -0.2021 13.0455   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -4.307e+00 4.498e-02 -95.758 <2e-16 \*\*\*  
## policy\_desc -3.860e-07 4.479e-09 -86.189 <2e-16 \*\*\*  
## areacode 4.928e-02 5.774e-03 8.535 <2e-16 \*\*\*  
## vehicle\_age -4.171e-02 1.264e-03 -33.008 <2e-16 \*\*\*  
## discount 2.059e-02 3.752e-04 54.875 <2e-16 \*\*\*  
## population\_density 3.933e-06 1.849e-06 2.127 0.0334 \*   
## exposure 8.419e-01 1.888e-02 44.588 <2e-16 \*\*\*  
## driver\_age 1.195e-02 4.697e-04 25.440 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for quasipoisson family taken to be 1.161116)  
##   
## Null deviance: 174181 on 542408 degrees of freedom  
## Residual deviance: 157839 on 542401 degrees of freedom  
## AIC: NA  
##   
## Number of Fisher Scoring iterations: 6

#See Predicted Value  
pred = predict(fit3,testData)  
  
#See Actual vs. Predicted Value  
finaldata = cbind(testData,pred)  
print(head(subset(finaldata, select = c(claim\_count,pred))))

## claim\_count pred  
## 5 1 -1.553780  
## 14 1 -1.630289  
## 16 1 -1.281767  
## 26 1 -1.043967  
## 28 1 -2.116839  
## 29 1 -1.407480

#Calculating RMSE  
rmse <- sqrt(mean((testData$claim\_count - pred)^2))  
print(rmse)

## [1] 3.388334

#check accuracy  
library(forecast)  
accuracy(fit3)

## ME RMSE MAE MPE MAPE MASE  
## Training set -1.664463e-12 0.2350469 0.09644269 -Inf Inf 0.9539941

# Comparing The Models  
coef1 = coef(fit3)  
coef2 = coef(fit4)  
se.coef1 = se.coef(fit3)  
se.coef2 = se.coef(fit4)  
models.both <- cbind(coef1, se.coef1, coef2, se.coef2, exponent=exp(coef1))

models.both

## coef1 se.coef1 coef2 se.coef2  
## (Intercept) -4.359023e+00 5.195962e-02 -4.306927e+00 4.497721e-02  
## policy\_desc -4.244857e-07 4.283002e-09 -3.860175e-07 4.478737e-09  
## areacode 5.447195e-02 5.814400e-03 4.927554e-02 5.773502e-03  
## vehicle\_age -2.532929e-02 1.267895e-03 -4.171468e-02 1.263792e-03  
## discount 2.030516e-02 3.490548e-04 2.058700e-02 3.751638e-04  
## car\_brandB10 5.996111e-02 3.961058e-02 3.933022e-06 1.848955e-06  
## car\_brandB11 1.515790e-01 4.288783e-02 8.418735e-01 1.888101e-02  
## car\_brandB12 5.911819e-01 1.997683e-02 1.194997e-02 4.697244e-04  
## car\_brandB13 4.356001e-02 4.558784e-02 -4.306927e+00 4.497721e-02  
## car\_brandB14 -3.101547e-01 9.175486e-02 -3.860175e-07 4.478737e-09  
## car\_brandB2 -1.579320e-02 1.707215e-02 4.927554e-02 5.773502e-03  
## car\_brandB3 3.111223e-02 2.439448e-02 -4.171468e-02 1.263792e-03  
## car\_brandB4 -1.914488e-02 3.335969e-02 2.058700e-02 3.751638e-04  
## car\_brandB5 3.810987e-02 2.781044e-02 3.933022e-06 1.848955e-06  
## car\_brandB6 -1.405729e-02 3.132609e-02 8.418735e-01 1.888101e-02  
## population\_density -5.015979e-06 1.945263e-06 1.194997e-02 4.697244e-04  
## regionR21 9.842397e-02 9.088066e-02 -4.306927e+00 4.497721e-02  
## regionR22 -2.562322e-03 5.843452e-02 -3.860175e-07 4.478737e-09  
## regionR23 -3.097804e-01 6.936174e-02 4.927554e-02 5.773502e-03  
## regionR24 -4.053619e-02 2.712418e-02 -4.171468e-02 1.263792e-03  
## regionR25 -6.688756e-02 4.969510e-02 2.058700e-02 3.751638e-04  
## regionR26 -8.852814e-02 5.491043e-02 3.933022e-06 1.848955e-06  
## regionR31 -1.975120e-01 3.941504e-02 8.418735e-01 1.888101e-02  
## regionR41 -3.656194e-01 5.094365e-02 1.194997e-02 4.697244e-04  
## regionR42 -1.008430e-01 9.906384e-02 -4.306927e+00 4.497721e-02  
## regionR43 -1.982518e-01 1.524796e-01 -3.860175e-07 4.478737e-09  
## regionR52 -1.836421e-01 3.391460e-02 4.927554e-02 5.773502e-03  
## regionR53 -2.998340e-02 3.198172e-02 -4.171468e-02 1.263792e-03  
## regionR54 -1.924137e-01 4.290031e-02 2.058700e-02 3.751638e-04  
## regionR72 -1.884765e-01 3.732890e-02 3.933022e-06 1.848955e-06  
## regionR73 -2.509607e-01 4.868964e-02 8.418735e-01 1.888101e-02  
## regionR74 2.802420e-02 7.337239e-02 1.194997e-02 4.697244e-04  
## regionR82 -2.153210e-02 2.644842e-02 -4.306927e+00 4.497721e-02  
## regionR83 -3.080203e-01 8.749154e-02 -3.860175e-07 4.478737e-09  
## regionR91 -1.610297e-01 3.616090e-02 4.927554e-02 5.773502e-03  
## regionR93 -1.027811e-01 2.805066e-02 -4.171468e-02 1.263792e-03  
## regionR94 1.795283e-01 7.510294e-02 2.058700e-02 3.751638e-04  
## vehicle\_hp -3.690641e-07 9.484050e-05 3.933022e-06 1.848955e-06  
## exposure 9.333687e-01 1.813370e-02 8.418735e-01 1.888101e-02  
## fuel\_type -3.740483e-02 1.230512e-02 1.194997e-02 4.697244e-04  
## driver\_age 1.156737e-02 4.447720e-04 -4.306927e+00 4.497721e-02  
## exponent  
## (Intercept) 0.01279088  
## policy\_desc 0.99999958  
## areacode 1.05598286  
## vehicle\_age 0.97498880  
## discount 1.02051271  
## car\_brandB10 1.06179525  
## car\_brandB11 1.16367025  
## car\_brandB12 1.80612189  
## car\_brandB13 1.04452267  
## car\_brandB14 0.73333347  
## car\_brandB2 0.98433086  
## car\_brandB3 1.03160128  
## car\_brandB4 0.98103722  
## car\_brandB5 1.03884537  
## car\_brandB6 0.98604105  
## population\_density 0.99999498  
## regionR21 1.10343051  
## regionR22 0.99744096  
## regionR23 0.73360800  
## regionR24 0.96027441  
## regionR25 0.93530036  
## regionR26 0.91527736  
## regionR31 0.82077026  
## regionR41 0.69376679  
## regionR42 0.90407492  
## regionR43 0.82016333  
## regionR52 0.83223362  
## regionR53 0.97046165  
## regionR54 0.82496548  
## regionR72 0.82821997  
## regionR73 0.77805293  
## regionR74 1.02842058  
## regionR82 0.97869806  
## regionR83 0.73490043  
## regionR91 0.85126680  
## regionR93 0.90232451  
## regionR94 1.19665272  
## vehicle\_hp 0.99999963  
## exposure 2.54306164  
## fuel\_type 0.96328609  
## driver\_age 1.01163453

# Comparing The Models  
coef3 = coef(fit1)  
coef4 = coef(fit2)  
se.coef3 = se.coef(fit1)  
se.coef4 = se.coef(fit2)  
models.both <- cbind(coef1, se.coef1, coef2, se.coef2, coef3, se.coef4, coef3, se.coef4, exponent=exp(coef1))

## Warning in cbind(coef1, se.coef1, coef2, se.coef2, coef3, se.coef4,  
## coef3, : number of rows of result is not a multiple of vector length (arg  
## 3)

models.both

## coef1 se.coef1 coef2 se.coef2  
## (Intercept) -4.359023e+00 5.195962e-02 -4.306927e+00 4.497721e-02  
## policy\_desc -4.244857e-07 4.283002e-09 -3.860175e-07 4.478737e-09  
## areacode 5.447195e-02 5.814400e-03 4.927554e-02 5.773502e-03  
## vehicle\_age -2.532929e-02 1.267895e-03 -4.171468e-02 1.263792e-03  
## discount 2.030516e-02 3.490548e-04 2.058700e-02 3.751638e-04  
## car\_brandB10 5.996111e-02 3.961058e-02 3.933022e-06 1.848955e-06  
## car\_brandB11 1.515790e-01 4.288783e-02 8.418735e-01 1.888101e-02  
## car\_brandB12 5.911819e-01 1.997683e-02 1.194997e-02 4.697244e-04  
## car\_brandB13 4.356001e-02 4.558784e-02 -4.306927e+00 4.497721e-02  
## car\_brandB14 -3.101547e-01 9.175486e-02 -3.860175e-07 4.478737e-09  
## car\_brandB2 -1.579320e-02 1.707215e-02 4.927554e-02 5.773502e-03  
## car\_brandB3 3.111223e-02 2.439448e-02 -4.171468e-02 1.263792e-03  
## car\_brandB4 -1.914488e-02 3.335969e-02 2.058700e-02 3.751638e-04  
## car\_brandB5 3.810987e-02 2.781044e-02 3.933022e-06 1.848955e-06  
## car\_brandB6 -1.405729e-02 3.132609e-02 8.418735e-01 1.888101e-02  
## population\_density -5.015979e-06 1.945263e-06 1.194997e-02 4.697244e-04  
## regionR21 9.842397e-02 9.088066e-02 -4.306927e+00 4.497721e-02  
## regionR22 -2.562322e-03 5.843452e-02 -3.860175e-07 4.478737e-09  
## regionR23 -3.097804e-01 6.936174e-02 4.927554e-02 5.773502e-03  
## regionR24 -4.053619e-02 2.712418e-02 -4.171468e-02 1.263792e-03  
## regionR25 -6.688756e-02 4.969510e-02 2.058700e-02 3.751638e-04  
## regionR26 -8.852814e-02 5.491043e-02 3.933022e-06 1.848955e-06  
## regionR31 -1.975120e-01 3.941504e-02 8.418735e-01 1.888101e-02  
## regionR41 -3.656194e-01 5.094365e-02 1.194997e-02 4.697244e-04  
## regionR42 -1.008430e-01 9.906384e-02 -4.306927e+00 4.497721e-02  
## regionR43 -1.982518e-01 1.524796e-01 -3.860175e-07 4.478737e-09  
## regionR52 -1.836421e-01 3.391460e-02 4.927554e-02 5.773502e-03  
## regionR53 -2.998340e-02 3.198172e-02 -4.171468e-02 1.263792e-03  
## regionR54 -1.924137e-01 4.290031e-02 2.058700e-02 3.751638e-04  
## regionR72 -1.884765e-01 3.732890e-02 3.933022e-06 1.848955e-06  
## regionR73 -2.509607e-01 4.868964e-02 8.418735e-01 1.888101e-02  
## regionR74 2.802420e-02 7.337239e-02 1.194997e-02 4.697244e-04  
## regionR82 -2.153210e-02 2.644842e-02 -4.306927e+00 4.497721e-02  
## regionR83 -3.080203e-01 8.749154e-02 -3.860175e-07 4.478737e-09  
## regionR91 -1.610297e-01 3.616090e-02 4.927554e-02 5.773502e-03  
## regionR93 -1.027811e-01 2.805066e-02 -4.171468e-02 1.263792e-03  
## regionR94 1.795283e-01 7.510294e-02 2.058700e-02 3.751638e-04  
## vehicle\_hp -3.690641e-07 9.484050e-05 3.933022e-06 1.848955e-06  
## exposure 9.333687e-01 1.813370e-02 8.418735e-01 1.888101e-02  
## fuel\_type -3.740483e-02 1.230512e-02 1.194997e-02 4.697244e-04  
## driver\_age 1.156737e-02 4.447720e-04 -4.306927e+00 4.497721e-02  
## coef3 se.coef4 coef3 se.coef4  
## (Intercept) -2.852288e-02 2.601153e-03 -2.852288e-02 2.601153e-03  
## policy\_desc -2.224583e-08 2.017381e-10 -2.224583e-08 2.017381e-10  
## areacode 2.509571e-03 2.928128e-04 2.509571e-03 2.928128e-04  
## vehicle\_age -1.306095e-03 5.918658e-05 -1.306095e-03 5.918658e-05  
## discount 1.339236e-03 2.393416e-05 1.339236e-03 2.393416e-05  
## car\_brandB10 2.815466e-03 1.011872e-07 2.815466e-03 1.011872e-07  
## car\_brandB11 7.229907e-03 9.307824e-04 7.229907e-03 9.307824e-04  
## car\_brandB12 3.624728e-02 6.608582e-04 3.624728e-02 6.608582e-04  
## car\_brandB13 3.166269e-03 2.636956e-05 3.166269e-03 2.636956e-05  
## car\_brandB14 -1.269086e-02 2.601153e-03 -1.269086e-02 2.601153e-03  
## car\_brandB2 -8.256609e-04 2.017381e-10 -8.256609e-04 2.017381e-10  
## car\_brandB3 1.908224e-03 2.928128e-04 1.908224e-03 2.928128e-04  
## car\_brandB4 -8.510902e-04 5.918658e-05 -8.510902e-04 5.918658e-05  
## car\_brandB5 2.478109e-03 2.393416e-05 2.478109e-03 2.393416e-05  
## car\_brandB6 -6.634885e-04 1.011872e-07 -6.634885e-04 1.011872e-07  
## population\_density -2.454197e-07 9.307824e-04 -2.454197e-07 9.307824e-04  
## regionR21 4.028118e-03 6.608582e-04 4.028118e-03 6.608582e-04  
## regionR22 -5.292873e-04 2.636956e-05 -5.292873e-04 2.636956e-05  
## regionR23 -1.501744e-02 2.601153e-03 -1.501744e-02 2.601153e-03  
## regionR24 -3.576552e-03 2.017381e-10 -3.576552e-03 2.017381e-10  
## regionR25 -4.486642e-03 2.928128e-04 -4.486642e-03 2.928128e-04  
## regionR26 -5.852058e-03 5.918658e-05 -5.852058e-03 5.918658e-05  
## regionR31 -1.049421e-02 2.393416e-05 -1.049421e-02 2.393416e-05  
## regionR41 -2.051657e-02 1.011872e-07 -2.051657e-02 1.011872e-07  
## regionR42 -4.936188e-03 9.307824e-04 -4.936188e-03 9.307824e-04  
## regionR43 -8.205753e-03 6.608582e-04 -8.205753e-03 6.608582e-04  
## regionR52 -1.067775e-02 2.636956e-05 -1.067775e-02 2.636956e-05  
## regionR53 -1.421323e-03 2.601153e-03 -1.421323e-03 2.601153e-03  
## regionR54 -1.155045e-02 2.017381e-10 -1.155045e-02 2.017381e-10  
## regionR72 -1.060947e-02 2.928128e-04 -1.060947e-02 2.928128e-04  
## regionR73 -1.183637e-02 5.918658e-05 -1.183637e-02 5.918658e-05  
## regionR74 7.842512e-04 2.393416e-05 7.842512e-04 2.393416e-05  
## regionR82 -1.138637e-03 1.011872e-07 -1.138637e-03 1.011872e-07  
## regionR83 -1.392909e-02 9.307824e-04 -1.392909e-02 9.307824e-04  
## regionR91 -1.002124e-02 6.608582e-04 -1.002124e-02 6.608582e-04  
## regionR93 -6.775927e-03 2.636956e-05 -6.775927e-03 2.636956e-05  
## regionR94 1.082956e-02 2.601153e-03 1.082956e-02 2.601153e-03  
## vehicle\_hp -1.426879e-07 2.017381e-10 -1.426879e-07 2.017381e-10  
## exposure 4.841355e-02 2.928128e-04 4.841355e-02 2.928128e-04  
## fuel\_type -1.459080e-03 5.918658e-05 -1.459080e-03 5.918658e-05  
## driver\_age 7.532870e-04 2.393416e-05 7.532870e-04 2.393416e-05  
## exponent  
## (Intercept) 0.01279088  
## policy\_desc 0.99999958  
## areacode 1.05598286  
## vehicle\_age 0.97498880  
## discount 1.02051271  
## car\_brandB10 1.06179525  
## car\_brandB11 1.16367025  
## car\_brandB12 1.80612189  
## car\_brandB13 1.04452267  
## car\_brandB14 0.73333347  
## car\_brandB2 0.98433086  
## car\_brandB3 1.03160128  
## car\_brandB4 0.98103722  
## car\_brandB5 1.03884537  
## car\_brandB6 0.98604105  
## population\_density 0.99999498  
## regionR21 1.10343051  
## regionR22 0.99744096  
## regionR23 0.73360800  
## regionR24 0.96027441  
## regionR25 0.93530036  
## regionR26 0.91527736  
## regionR31 0.82077026  
## regionR41 0.69376679  
## regionR42 0.90407492  
## regionR43 0.82016333  
## regionR52 0.83223362  
## regionR53 0.97046165  
## regionR54 0.82496548  
## regionR72 0.82821997  
## regionR73 0.77805293  
## regionR74 1.02842058  
## regionR82 0.97869806  
## regionR83 0.73490043  
## regionR91 0.85126680  
## regionR93 0.90232451  
## regionR94 1.19665272  
## vehicle\_hp 0.99999963  
## exposure 2.54306164  
## fuel\_type 0.96328609  
## driver\_age 1.01163453

# Final Interpretation:  
  
# Root Mean Square Error (RMSE) measures how much error there is between two data sets. In other words, it compares a predicted value and an observed or known value. Here we can see that the RSME value is also quite low for Poisson Regression Model(0.2352745) as compare to Linear Regression(0.2369978). So our final model is poisson regression.