Assignment - 6

Maksuda Aktar Toma, Aarif Baksh

November 18, 2024

Business Problem

As an employee of CloverShield Insurance company, you are tasked with addressing the challenge of reducing call center costs. Your business partners have requested the development of a predictive model that, based on the provided segmentation, forecasts the number of times a policyholder is likely to call. This model aims to optimize resource allocation and enhance cost-efficiency in call center operations.

To find all our works on this project go to this link https://github.com/maksudatoma/2024-Travelers-University-Modeling-Competition/tree/main

Introduction

The data obtained from Kaggle, is split into two parts: training data and validation data. In the validation data, the target variable, call_counts, is omitted. The training dataset contains 80,000 samples, and the validation dataset contains 20,000 samples.

Variable Descriptions

- `ann_prm_amt`: Annualized Premium Amount
- `bi_limit_group`: Body injury limit group (SP stands for single split limit coverage, CSL stands for combined single limit coverage)
- `channel`: Distribution channel
- `newest_veh_age`: The age of the newest vehicle insured on a policy (-20 represents non-auto or missing values)
- `geo_group`: Indicates if the policyholder lives in a rural, urban, or suburban area
- `has_prior_carrier`: Did the policyholder come from another carrier
- `home_lot_sq_footage`: Square footage of the policyholder's home lot

- `household_group`: The types of policy in household
- `household_policy_counts`: Number of policies in the household
- `telematics_ind`: Telematic indicator (0 represents auto missing values or didn't enroll and -2 represents non-auto)
- `digital_contacts_ind`: An indicator to denote if the policy holder has opted into digital communication
- `12m_call_history`: Past one year call count
- `tenure_at_snapshot`:Policy active length in month
- `pay_type_code`: Code indicating the payment method
- `acq_method`:The acquisition method (Miss represents missing values)
- `trm_len_mo`: Term length month
- `pol_edeliv_ind`: An indicator for email delivery of documents (-2 represents missing values)
- `aproduct_sbtyp_grp`: Product subtype group
- `product_sbtyp`: Product subtype
- `call_counts`: The number of call count generated by each policy (target variable)

Data Cleaning and Missing Value count

First, we prepares the data by cleaning and transforming it (e.g., converting characters to factors, marking missing values.)

Table 1: **Table 1: Variables with Missing Values**

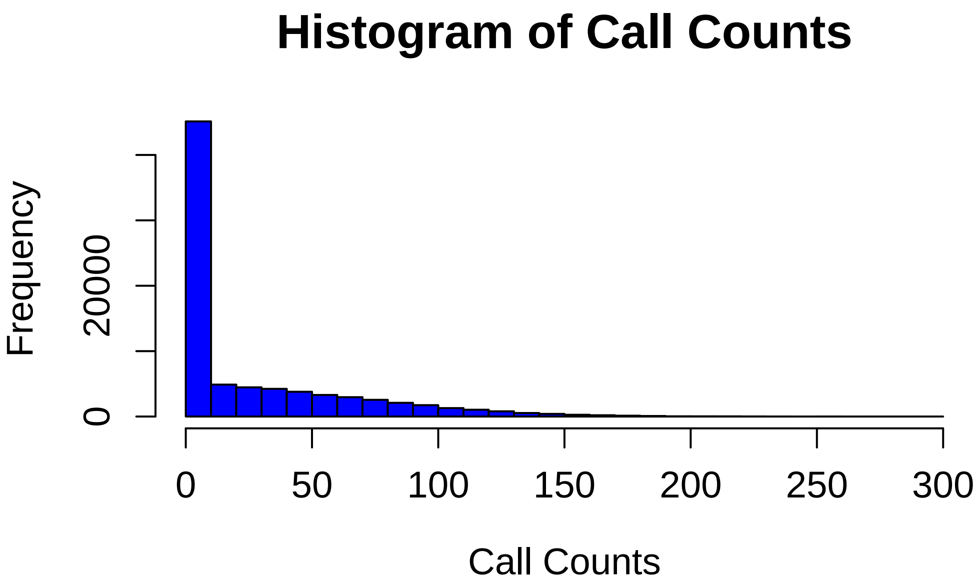
Variable	Number of missing values
acq_method	16,066
newest_veh_age	58,015
pol_edeliv_ind	838
telematics_ind	58,015

Zero Values: 50.18% of the rows in the call_counts column are zeros, indicating that most customers made no calls. This is significant and might suggest using models like Zero-Inflated Poisson (ZIP) to handle the high frequency of zeros.

Key Takeaways - The dataset contains both numeric and categorical variables, with some columns having significant missing values. - The target variable (call_counts) is heavily zero-inflated and skewed, which may require specialized modeling approaches. - Some numeric variables, like ann_prm_amt and home_lot_sq_footage, have wide ranges and outliers, suggesting that data transformation or scaling may be beneficial.

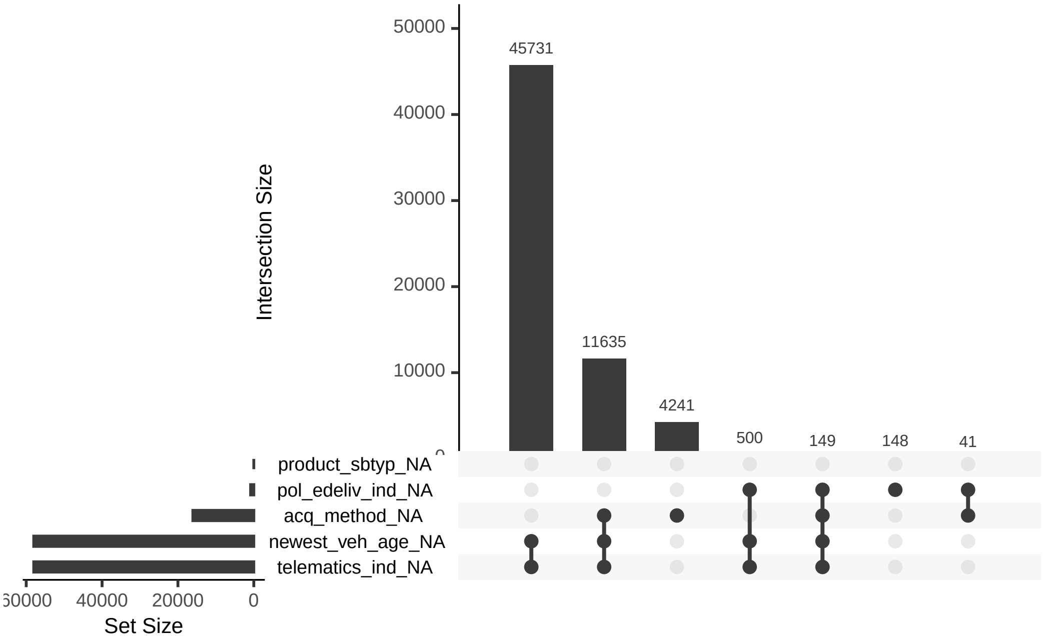
```
    acq_method newest_veh_age pol_edeliv_ind telematics_ind
         16066          58015            838          58015
```

```
[1] 50.18
```

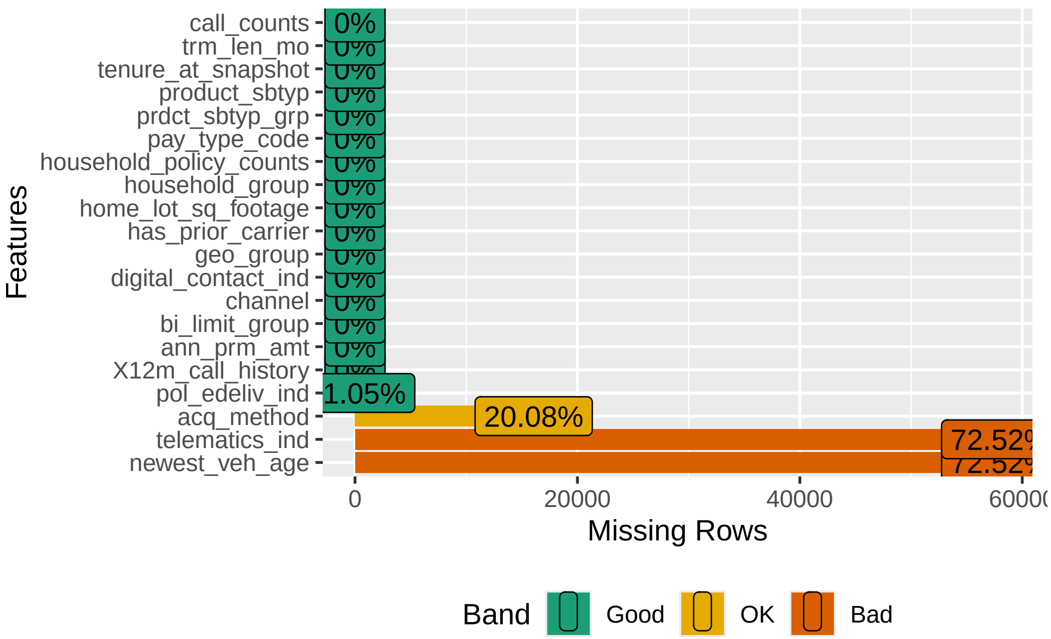


Missing Value display

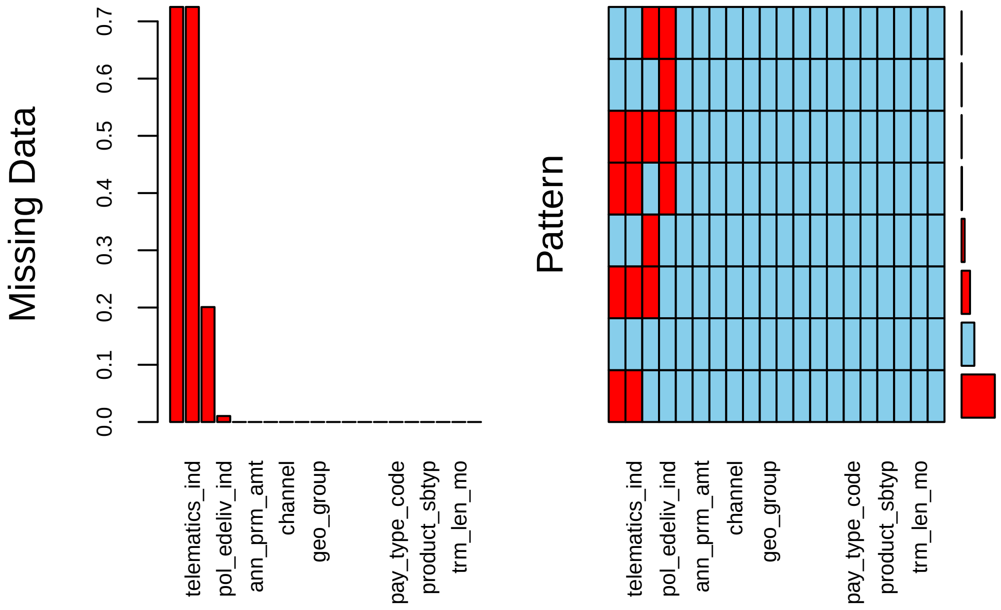
1. This UpSet Plot shows the patterns and extent of missing data across variables. The horizontal bars on the left represent the total missing values for each variable, with newest_veh_age and telematics_ind having the most missing data. The vertical bars represent the number of rows with specific missingness patterns, with the tallest bar (~45,731 rows) indicating that only newest_veh_age has missing values. The connected dots below highlight combinations of missingness across variables, with fewer rows having simultaneous missing values in multiple variables. This analysis suggests focusing on simple imputation for variables with isolated missingness and predictive modeling for overlapping patterns



2. This chart shows the percentage and count of missing values for each feature in the dataset. Most features (green bars) have no missing data, making them ready for modeling. However, newest_veh_age and telematics_ind have significant missingness (72.52%), requiring advanced imputation or removal. acq_method has moderate missingness (20.08%), which can be addressed with simpler imputation methods. Features like pol_edeliv_ind (1.05% missing) require minimal effort to handle, such as mean or mode imputation. The focus should be on addressing features with high and moderate missingness to ensure data quality for modeling.



3.This visualization highlights missing data patterns in the dataset. The left panel shows that telematics_ind and newest_veh_age have the highest proportion of missing values (~70%), while pol_edeliv_ind has a smaller proportion (~10%). The right panel reveals that most rows have no missing data (blue squares), but missingness in telematics_ind and newest_veh_age often co-occurs. Other features have negligible or no missing data. It is recommended to either impute or exclude telematics_ind and newest_veh_age depending on their importance, while simpler imputation methods can handle pol_edeliv_ind.



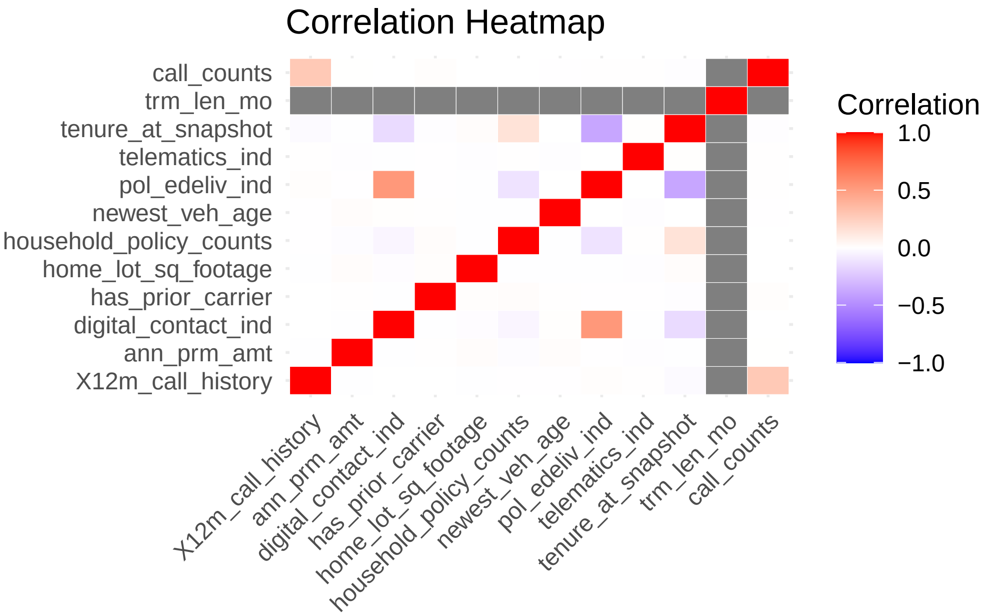
```
Variables sorted by number of missings:
               Variable     Count
        newest_veh_age 0.7251875
        telematics_ind 0.7251875
            acq_method 0.2008250
        pol_edeliv_ind 0.0104750
     X12m_call_history 0.0000000
           ann_prm_amt 0.0000000
        bi_limit_group 0.0000000
               channel 0.0000000
   digital_contact_ind 0.0000000
             geo_group 0.0000000
     has_prior_carrier 0.0000000
   home_lot_sq_footage 0.0000000
       household_group 0.0000000
household_policy_counts 0.0000000
         pay_type_code 0.0000000
       prdct_sbtyp_grp 0.0000000
         product_sbtyp 0.0000000
    tenure_at_snapshot 0.0000000
            trm_len_mo 0.0000000
           call_counts 0.0000000
```

Correlation Structure for Call Count and Numeric Predictors

The correlations output show that X12m_call_history (r=0.28) is the strongest numeric predictor of call_counts, with a moderate positive relationship. Other variables like telematics_ind (=0.0059) and pol_edeliv_ind (=0.0049) have very weak positive correlations, while variables like household_policy_counts (r=−0.0033) and newest_veh_age (r=−0.0030) have negligible negative correlations. Most numeric variables show correlations close to zero, suggesting little to no linear relationship with the target variable. Overall, X12m_call_history is the most promising numeric predictor, while others may require further evaluation for relevance in modeling.

```
   X12m_call_history             ann_prm_amt     digital_contact_ind
        0.2799527640            0.0009293953            0.0026141348
   has_prior_carrier    home_lot_sq_footage household_policy_counts
        0.0005052426            0.0009486643           -0.0033470952
      newest_veh_age          pol_edeliv_ind          telematics_ind
       -0.0030184309            0.0048667762            0.0058867474
  tenure_at_snapshot              trm_len_mo             call_counts
       -0.0014746341            0.0007817227            1.0000000000
```

Correlation Matrix: The correlation heatmap shows that X12m_call_history has the strongest positive correlation (r 0.28) with call_counts, making it the most important numeric predictor. Most other variables, such as ann_prm_amt, household_policy_counts, and home_lot_sq_footage, have weak or no significant correlations with the target variable, as indicated by grey cells. There are no strong negative correlations in the dataset. Overall, the relationships are mostly weak, suggesting that non-linear models or feature engineering may be needed to capture more complex interactions. The heatmap helps identify X12m_call_history as a key feature while others may contribute less linearly.



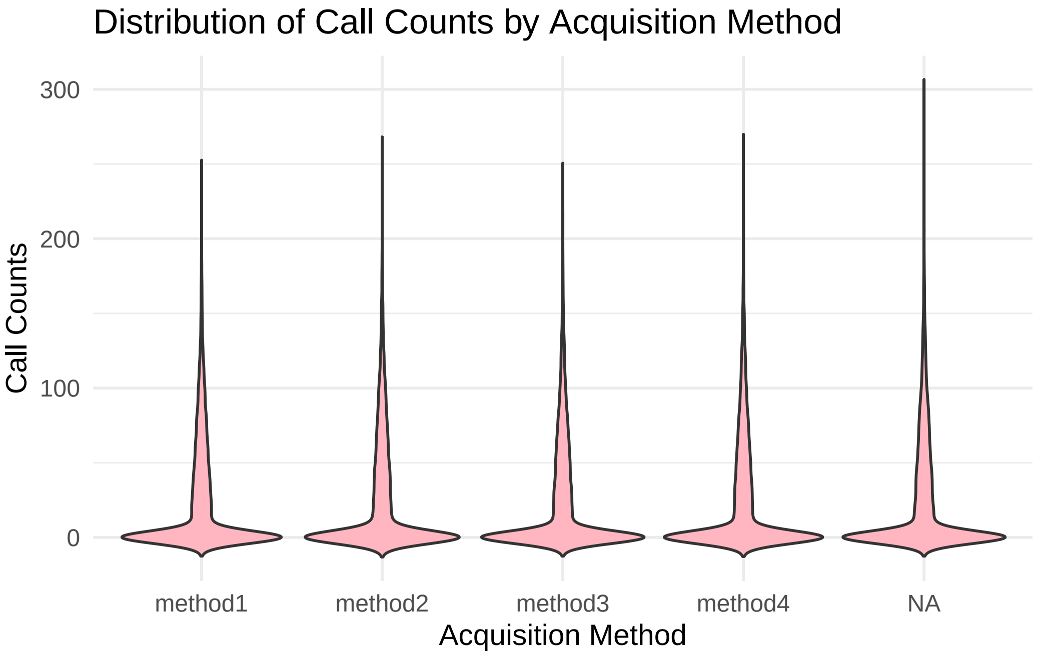
ANOVA for relationship between Call Count and Categorical Predictors

The ANOVA results evaluate the effect of categorical variables on call_counts. Among the predictors, acq_method is marginally significant (p=0.0518), suggesting it may have a weak influence on call_counts. All other categorical variables, such as bi_limit_group, channel, and geo_group, have p-values greater than 0.1, indicating no statistically significant relationship with the target variable. Additionally, 16,066 rows were excluded due to missing data, which might affect the robustness of the results. It is recommended to focus on acq_method for further analysis and consider handling missing data to improve model accuracy.

```
$acq_method
               Df    Sum Sq Mean Sq F value Pr(>F)
trav[[var]]     3     11110    3703   2.579 0.0518 .
Residuals   63930 91805237    1436
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
16066 observations deleted due to missingness

$bi_limit_group
               Df     Sum Sq Mean Sq F value Pr(>F)
trav[[var]]     7      2207   315.3    0.22  0.981
Residuals   79992 114719475  1434.1
```

```
$channel
               Df    Sum Sq Mean Sq F value Pr(>F)
trav[[var]]     1       146   146.2   0.102   0.75
Residuals   79998 114721536  1434.1

$geo_group
               Df    Sum Sq Mean Sq F value Pr(>F)
trav[[var]]     2      5412    2706   1.887  0.152
Residuals   79997 114716270    1434

$household_group
               Df    Sum Sq Mean Sq F value Pr(>F)
trav[[var]]     3      2624   874.7    0.61  0.608
Residuals   79996 114719058  1434.1

$pay_type_code
               Df    Sum Sq Mean Sq F value Pr(>F)
trav[[var]]     2       117    58.7   0.041   0.96
Residuals   79997 114721565  1434.1

$prdct_sbtyp_grp
               Df    Sum Sq Mean Sq F value Pr(>F)
trav[[var]]     2      1861   930.6   0.649  0.523
Residuals   79997 114719821  1434.1

$product_sbtyp
               Df    Sum Sq Mean Sq F value Pr(>F)
trav[[var]]     2       117    58.7   0.041   0.96
Residuals   79997 114721565  1434.1
```

Call_counts distribution with significant predictor The violin plot shows the distribution of call_counts across different acquisition methods (acq_method). All methods have a heavily skewed distribution, with most values near 0 and a few extreme outliers, indicating that the majority of customers make few calls. The distributions are nearly identical across all methods, including the NA category, suggesting that acq_method has minimal impact on call_counts. This aligns with the ANOVA results, where acq_method was marginally significant. Further analysis, such as handling outliers or exploring interactions with other variables, may provide additional insights.



Imputing Missing Values

The dataset is prepared by converting character columns to factors and handling missing data by replacing coded values such as `-2`, `-20`, and "missing" with `NA`. The code then calculates the percentage of zero values in the `call_counts` column to assess the distribution of the response variable. To address these missing values, the `mice` function performs multiple imputation, generating five potential datasets and selecting one for subsequent analysis to ensure consistency. For imputation, the `mice` function is used with a vector of default methods tailored to different types of variables: predictive mean matching (`pmm`) for numeric data, logistic regression imputation (`logreg`) for binary (factor with 2 levels), polytomous regression imputation (`polyreg`) for unordered categorical data with more than two levels, and the proportional odds model (`polr`) for ordered factors with more than two levels.

Finally, adjustments are made to factor variables to exclude "missing" as a level, preserving data integrity.

The imputed dataset was saved as trav3 and imported for inclusion in this document. The code for imputation is included in the appendix.

Splitting Dataset

Before fitting any models, we will split the provided training dataset into three subsets: 60% for training, 20% for validation, and 20% for testing. This split will be done while ensuring stratification based on the variable call_counts. Stratification preserves the distribution of call_counts across all subsets, confirmed by the nearly identical means of the subsets. The training set is used to build the model, the validation set is used for tuning and performance assessment during training, and the test set is reserved for final evaluation. This ensures unbiased and representative splits for reliable model training and testing.

```
[1] 25.93654
```

```
[1] 25.88844
```

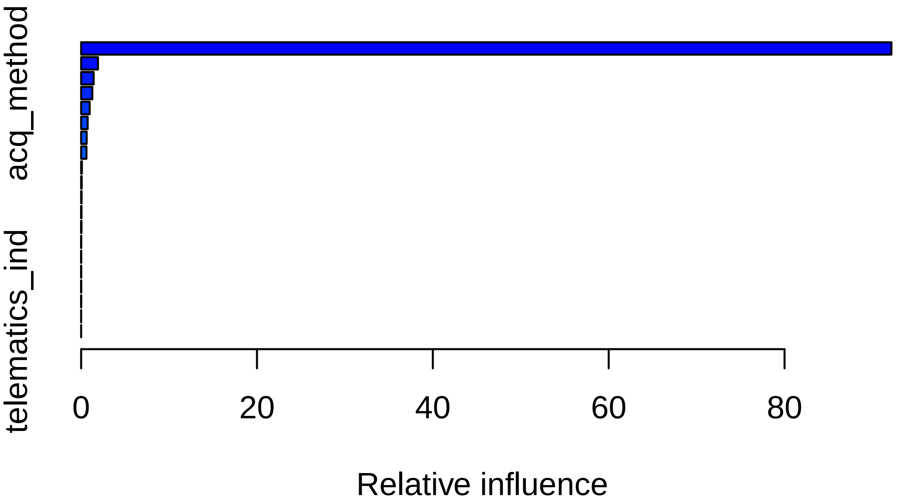
```
[1] 25.85405
```

Models

Model 1 GBM : We trained a GBM model using 500 trees with a Poisson distribution to predict call_counts. The hyperparameters for this model were selected by trial and error. Attempts to use specific functions (e.g. the train function) for hyperparameter tuning were unsuccessful due to insufficient computer memory needed to execute the code. This model achieved a test RMSE of 36.005, indicating moderate prediction error, suggesting the predictions deviate by about 36 calls on average from actual values.

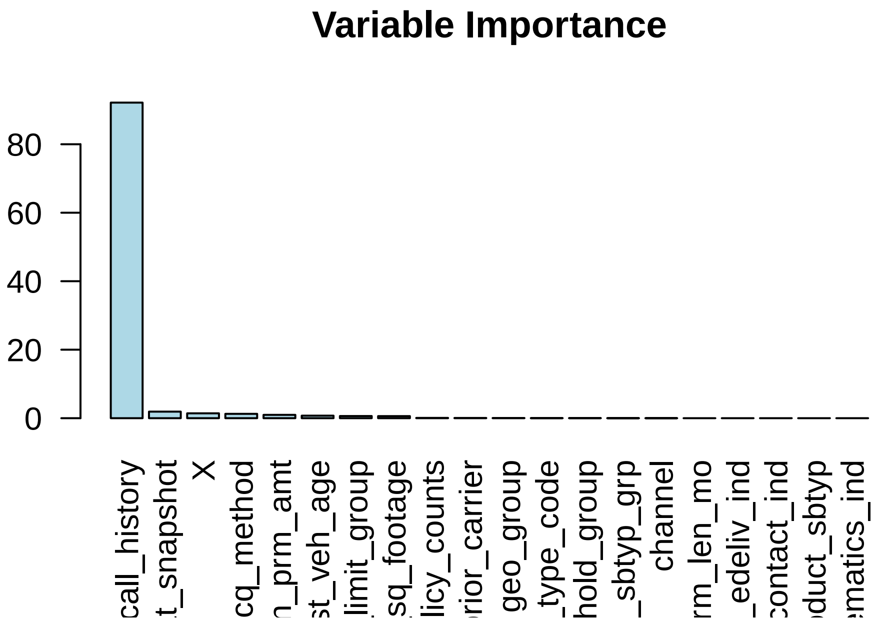
The variable importance plot shows the relative importance of the top 10 predictors. `X_12m_call_history` is the most important predictor, with its relative information gain being 92.64%. The remaining 9 variables account for only 7.38% of the relative information gain.

```
[1] 36.00466
```



```
                                            var      rel.inf
X12m_call_history             X12m_call_history 92.154489694
tenure_at_snapshot           tenure_at_snapshot  1.915076104
X                                             X  1.423861229
acq_method                           acq_method  1.272024109
ann_prm_amt                         ann_prm_amt  0.970849912
newest_veh_age                   newest_veh_age  0.743306518
bi_limit_group                   bi_limit_group  0.630751577
home_lot_sq_footage         home_lot_sq_footage  0.603946591
household_policy_counts household_policy_counts  0.084294286
has_prior_carrier             has_prior_carrier  0.067526670
geo_group                             geo_group  0.050754223
pay_type_code                     pay_type_code  0.033471162
household_group                 household_group  0.031622071
prdct_sbtyp_grp                 prdct_sbtyp_grp  0.005576644
channel                                 channel  0.004307348
trm_len_mo                           trm_len_mo  0.003287673
pol_edeliv_ind                   pol_edeliv_ind  0.003155542
digital_contact_ind         digital_contact_ind  0.001698647
product_sbtyp                     product_sbtyp  0.000000000
telematics_ind                   telematics_ind  0.000000000

                                            var      rel.inf
```

```
X12m_call_history           X12m_call_history 92.154489694
tenure_at_snapshot         tenure_at_snapshot  1.915076104
X                                           X  1.423861229
acq_method                         acq_method  1.272024109
ann_prm_amt                       ann_prm_amt  0.970849912
newest_veh_age                 newest_veh_age  0.743306518
bi_limit_group                 bi_limit_group  0.630751577
home_lot_sq_footage       home_lot_sq_footage  0.603946591
household_policy_counts household_policy_counts  0.084294286
has_prior_carrier           has_prior_carrier  0.067526670
geo_group                           geo_group  0.050754223
pay_type_code                   pay_type_code  0.033471162
household_group               household_group  0.031622071
prdct_sbtyp_grp               prdct_sbtyp_grp  0.005576644
channel                               channel  0.004307348
trm_len_mo                         trm_len_mo  0.003287673
pol_edeliv_ind                 pol_edeliv_ind  0.003155542
digital_contact_ind       digital_contact_ind  0.001698647
product_sbtyp                   product_sbtyp  0.000000000
telematics_ind                 telematics_ind  0.000000000
```



Model 2 ZIP: The Zero-Inflated Poisson (ZIP) model predicts call_counts while accounting for excess zeros. An attempt to use all variable sin this model resulted in errors in convergence

and NA's for the standard errors, z-value and p-value. Instead, a subset of variables was considered based on the variable importance plot from the GBM model.

The ZIP model has has two parts:

The Count Model: variables that directly affect the frequency of calls were considered. These include X12_m_call_history, bi_limit_group , acq_method , geo_group , and interaction between acq_method and geo_group , and between X12_m_call_history and bi_limit_group.

The Zero-inflation Model: variables that indicate whether a customer is likely to have any calls at all were considered. These include X12_m_call_history and ann_prm_amt. The test RMSE from this model is 36.5291, which is marginally poorer than the RSME from the GBM model.

```
Call:
zeroinfl(formula = call_counts ~ X12m_call_history + bi_limit_group +
    acq_method + geo_group + acq_method * geo_group + bi_limit_group *
    X12m_call_history | X12m_call_history + ann_prm_amt, data = train_data,
    dist = "poisson")

Pearson residuals:
     Min       1Q   Median       3Q      Max
-12.4020  -0.8633  -0.7940   0.6512  14.2342

Count model coefficients (poisson with log link):
                                             Estimate Std. Error z value
(Intercept)                                 3.9467726  0.0061997 636.610
X12m_call_history                           0.0172988  0.0006071  28.493
bi_limit_groupCSLGrp2                      -0.0675800  0.0075932  -8.900
bi_limit_groupCSLGrp3                      -0.0423091  0.0077147  -5.484
bi_limit_groupNonAuto                      -0.0544079  0.0054504  -9.982
bi_limit_groupSPGrp1Miss                   -0.0989670  0.0076887 -12.872
bi_limit_groupSPGrp2                       -0.0602661  0.0077248  -7.802
bi_limit_groupSPGrp3                       -0.0578579  0.0076771  -7.536
bi_limit_groupSPGrp4                       -0.1219732  0.0079283 -15.384
acq_methodmethod2                          -0.0372241  0.0043463  -8.565
acq_methodmethod3                          -0.0595819  0.0044744 -13.316
acq_methodmethod4                          -0.0277611  0.0048472  -5.727
geo_groupsuburban                          -0.0394387  0.0048028  -8.212
geo_groupurban                             -0.0603572  0.0048445 -12.459
acq_methodmethod2:geo_groupsuburban         0.0531073  0.0061868   8.584
acq_methodmethod3:geo_groupsuburban         0.0729722  0.0063652  11.464
```

```
acq_methodmethod4:geo_groupsuburban         0.0616667  0.0068443   9.010
acq_methodmethod2:geo_groupurban            0.0730073  0.0062291  11.720
acq_methodmethod3:geo_groupurban            0.0450482  0.0064137   7.024
acq_methodmethod4:geo_groupurban            0.0585801  0.0069091   8.479
X12m_call_history:bi_limit_groupCSLGrp2     0.0105018  0.0008549  12.285
X12m_call_history:bi_limit_groupCSLGrp3     0.0071846  0.0009737   7.379
X12m_call_history:bi_limit_groupNonAuto     0.0049813  0.0006251   7.969
X12m_call_history:bi_limit_groupSPGrp1Miss  0.0162731  0.0009379  17.351
X12m_call_history:bi_limit_groupSPGrp2      0.0076117  0.0010388   7.327
X12m_call_history:bi_limit_groupSPGrp3      0.0059484  0.0008959   6.640
X12m_call_history:bi_limit_groupSPGrp4      0.0121103  0.0010003  12.106
                                           Pr(>|z|)
(Intercept)                                 < 2e-16 ***
X12m_call_history                           < 2e-16 ***
bi_limit_groupCSLGrp2                       < 2e-16 ***
bi_limit_groupCSLGrp3                      4.15e-08 ***
bi_limit_groupNonAuto                       < 2e-16 ***
bi_limit_groupSPGrp1Miss                    < 2e-16 ***
bi_limit_groupSPGrp2                       6.11e-15 ***
bi_limit_groupSPGrp3                       4.83e-14 ***
bi_limit_groupSPGrp4                        < 2e-16 ***
acq_methodmethod2                           < 2e-16 ***
acq_methodmethod3                           < 2e-16 ***
acq_methodmethod4                          1.02e-08 ***
geo_groupsuburban                           < 2e-16 ***
geo_groupurban                              < 2e-16 ***
acq_methodmethod2:geo_groupsuburban         < 2e-16 ***
acq_methodmethod3:geo_groupsuburban         < 2e-16 ***
acq_methodmethod4:geo_groupsuburban         < 2e-16 ***
acq_methodmethod2:geo_groupurban            < 2e-16 ***
acq_methodmethod3:geo_groupurban           2.16e-12 ***
acq_methodmethod4:geo_groupurban            < 2e-16 ***
X12m_call_history:bi_limit_groupCSLGrp2     < 2e-16 ***
X12m_call_history:bi_limit_groupCSLGrp3    1.60e-13 ***
X12m_call_history:bi_limit_groupNonAuto    1.60e-15 ***
X12m_call_history:bi_limit_groupSPGrp1Miss  < 2e-16 ***
X12m_call_history:bi_limit_groupSPGrp2     2.35e-13 ***
X12m_call_history:bi_limit_groupSPGrp3     3.15e-11 ***
X12m_call_history:bi_limit_groupSPGrp4      < 2e-16 ***

Zero-inflation model coefficients (binomial with logit link):
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)        4.277e-01  1.705e-02  25.092   <2e-16 ***
```

```
X12m_call_history -1.655e-01  3.528e-03 -46.907   <2e-16 ***
ann_prm_amt       -2.471e-06  1.128e-05  -0.219    0.827
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Number of iterations in BFGS optimization: 40
Log-likelihood: -4.311e+05 on 30 Df
```

```
[1] 36.5292
```

Appendix- R Code

```r
library(caret)
library(dplyr)
library(mice)

trav <- read.csv("train_data.csv")

#Exclude first column (ID column)
trav <- trav[,-1]

trav <- trav %>%
  mutate(across(where(is.character), as.factor))

trav[trav == -2 |trav == -20 | trav == "missing"] <- NA

missing_counts <- colSums(is.na(trav))

# Display variables with missing values and their counts
missing_counts[missing_counts > 0]

#Zero values for the response
per0resp <- sum(trav$call_counts == 0) / nrow(trav) * 100
per0resp
```

```r
# Plot histogram for call_counts
hist(trav$call_counts,
     breaks = 30,  # Number of bins
     col = "blue",  # Fill color
     border = "black",  # Border color
     main = "Histogram of Call Counts",  # Title
     xlab = "Call Counts",  # X-axis label
     ylab = "Frequency",  # Y-axis label
     cex.main = 1.5,  # Text size for title
     cex.lab = 1.2,  # Text size for labels
     cex.axis = 1.2)  # Text size for axis
```

```
#Visualising pattern of missingness

#1. Heatmap
library(naniar)

# Visualize missing data with a heatmap
gg_miss_upset(trav)  # Upset plot to show combinations of missingness

#2.
library(DataExplorer)

# Visualize missing data
plot_missing(trav)

# 3. Heatmap )

library(VIM)

# Visualize missing data with a matrix plot
aggr(trav, col = c("skyblue", "red"), numbers = TRUE, sortVars = TRUE,
     labels = names(trav), cex.axis = 0.7, gap = 3,
     ylab = c("Missing Data", "Pattern"))
```

```
# Load necessary library
library(ggplot2)
library(reshape2)

# Select numeric columns for correlation
num_vars <- sapply(trav, is.numeric)
correlation_matrix <- cor(trav[, num_vars], use = "complete.obs")

# Melt the correlation matrix for ggplot2
melted_corr <- melt(correlation_matrix)

# Plot the heatmap
```

```r
ggplot(data = melted_corr, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1, 1), space = "Lab",
                       name = "Correlation") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = "Correlation Heatmap", x = "", y = "")
```

```r
#ANOVA for categorical predictors
cat_vars <- sapply(trav, is.factor)

anova_results <- lapply(names(trav)[cat_vars], function(var) {
  anova_model <- aov(trav$call_counts ~ trav[[var]])
  summary(anova_model)
})

# Print ANOVA summaries
names(anova_results) <- names(trav)[cat_vars]
anova_results
```

```r
# Create violin plot
ggplot(trav, aes(x = acq_method, y = call_counts)) +
  geom_violin(fill = "lightpink", trim = FALSE) +
  labs(title = "Distribution of Call Counts by Acquisition Method",
       x = "Acquisition Method",
       y = "Call Counts") +
  theme_minimal()
```

```r
# Split data into training set (60%)
set.seed(123)  # For reproducibility
trainIndex <- createDataPartition(trav$call_counts, p = 0.6, list = FALSE)
train_data <- trav[trainIndex, ]

# Remaining data (40%) for validation and test sets
rem_data <- trav[-trainIndex, ]

# Split remaining data into validation (20%) and test (20%)
validIndex <- createDataPartition(rem_data$call_counts, p = 0.5, list = FALSE)
valid_data <- rem_data[validIndex, ]
test_data <- rem_data[-validIndex, ]
```

```
# Checking whether stratification was successful
mean(train_data$call_counts)
mean(valid_data$call_counts)
mean(test_data$call_counts)
```

```
#Model 1: GBM Model
library(gbm)
gbm.poisson <- gbm(call_counts ~ ., data = train_data, distribution="poisson", n.tree = 500,
                   interaction.depth=7, shrinkage=0.01,n.minobsinnode=20,bag.fraction=1)


gbm.poissonpred <- predict(gbm.poisson, newdata = test_data, type = "response")
# Evaluate model performance using RMSE
rmse1 <- sqrt(mean((gbm.poissonpred - test_data$call_counts)^2))
rmse1

# Display variable importance and plot
summary(gbm.poisson)

var_importance <- summary(gbm.poisson, plotit = FALSE)

# Print variable importance data frame
print(var_importance)

# Plot variable importance manually
barplot(var_importance$rel.inf, names.arg = var_importance$var, las = 2,
        col = "lightblue", main = "Variable Importance")
```

```
#Model 2: ZIP

library(pscl)

# Zero-Inflated Poisson model

zip_model <- zeroinfl(call_counts ~
                        X12m_call_history + bi_limit_group + acq_method
                      + geo_group + acq_method*geo_group +
                        bi_limit_group*X12m_call_history |
                        X12m_call_history + ann_prm_amt, data = train_data, dist =
                        "poisson")

summary(zip_model)
```

```r
zip_preds <- predict(zip_model, newdata = test_data, type = "response")
rmse2 <- sqrt(mean((zip_preds - test_data$call_counts)^2))
rmse2
```