Sample Project

```
## package 'caret' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\kunta\AppData\Local\Temp\Rtmp6to8Mj\downloaded_packages

## package 'plotly' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\kunta\AppData\Local\Temp\Rtmp6to8Mj\downloaded_packages
```

DATASET

View Data

###Load data from csv file

```
dataset = read.csv('Rx_Regional_Monthly_04_2023.csv')
```

Structure of dataset

```
str(dataset)
```

```
## 'data.frame':    884941 obs. of  9 variables:
##  $ ï..State        : chr  "ALABAMA" "ALABAMA" "ALABAMA" "ALABAMA" ...
##  $ Channel         : chr  "LONG TERM CARE" "LONG TERM CARE" "LONG TERM CARE" "LONG TERM CARE" ...
##  $ Specialty       : chr  "ADDICTION MEDICINE" "ADDICTION MEDICINE" "ADDICTION MEDICINE" "ADDICTION M
##  $ Form..TLC1.     : chr  "A SYSTEMIC ORAL SOLID REG" "A SYSTEMIC ORAL SOLID REG" "A SYSTEMIC ORAL SO
##  $ Calendar.Quarter: chr  "Q2 2021" "Q3 2021" "Q4 2021" "Q1 2022" ...
##  $ TRx             : chr  "923" "845" "291" "229" ...
##  $ EUTRx           : chr  "22,602" "21,942" "9,466" "7,651" ...
##  $ RRx             : chr  "15" "3" "4" "0" ...
##  $ NRx             : chr  "908" "842" "287" "229" ...
```

Sample data

```
##   ï..State        Channel          Specialty               Form..TLC1.
## 1  ALABAMA LONG TERM CARE ADDICTION MEDICINE A SYSTEMIC ORAL SOLID REG
## 2  ALABAMA LONG TERM CARE ADDICTION MEDICINE A SYSTEMIC ORAL SOLID REG
## 3  ALABAMA LONG TERM CARE ADDICTION MEDICINE A SYSTEMIC ORAL SOLID REG
## 4  ALABAMA LONG TERM CARE ADDICTION MEDICINE A SYSTEMIC ORAL SOLID REG
## 5  ALABAMA LONG TERM CARE ADDICTION MEDICINE A SYSTEMIC ORAL SOLID REG
```

```
## 6  ALABAMA LONG TERM CARE ADDICTION MEDICINE A SYSTEMIC ORAL SOLID REG
##   Calendar.Quarter TRx  EUTRx RRx NRx
## 1          Q2 2021 923 22,602  15 908
## 2          Q3 2021 845 21,942   3 842
## 3          Q4 2021 291  9,466   4 287
## 4          Q1 2022 229  7,651   0 229
## 5          Q2 2022 188  6,919   0 188
## 6          Q3 2022  86  4,071   0  86
```

Total rows in dataset

```
## [1] "884941 Rows"
```

Data Cleaning

Check if there are NA's in the data

```
##         ï..State          Channel        Specialty      Form..TLC1.
##                 0                0                0                0
## Calendar.Quarter              TRx            EUTRx              RRx
##                 0                0                0                0
##               NRx
##                 0
```

Convert the metrics to integer taking care of comma in the value and add the ID's to the categorical dimensions

```
modifiedDataset <- dataset %>%
  mutate(TRx = as.numeric(gsub(",", "", TRx)),
         EUTRx = as.numeric(gsub(",", "", EUTRx)),
         RRx = as.numeric(gsub(",", "", RRx)),
         NRx = as.integer(gsub(",", "", NRx))
         )

colPos <- c(1,2,3,4,5)
modifiedDataset <-  modifiedDataset %>% mutate_at(colPos, funs(factor(.)))
```

Structure of modified dataset

```
## 'data.frame':    884941 obs. of  9 variables:
##  $ ï..State        : Factor w/ 51 levels "ALABAMA","ALASKA",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Channel         : Factor w/ 3 levels "LONG TERM CARE",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Specialty       : Factor w/ 77 levels "ADDICTION MEDICINE",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Form..TLC1.     : Factor w/ 20 levels "A SYSTEMIC ORAL SOLID REG",..: 1 1 1 1 1 1 1 1 2 2 ...
##  $ Calendar.Quarter: Factor w/ 8 levels "Q1 2022","Q1 2023",..: 3 5 7 1 4 6 8 2 3 5 ...
##  $ TRx             : num  923 845 291 229 188 86 15 10 14 11 ...
##  $ EUTRx           : num  22602 21942 9466 7651 6919 ...
##  $ RRx             : num  15 3 4 0 0 0 1 0 0 0 ...
##  $ NRx             : int  908 842 287 229 188 86 14 10 14 11 ...
```

Check any values converted to NA's during char to integer conversion

```
##         ï..State          Channel        Specialty      Form..TLC1.
##                 0                0                0                0
## Calendar.Quarter              TRx            EUTRx              RRx
##                 0                0                0                0
##              NRx
##                 0
```

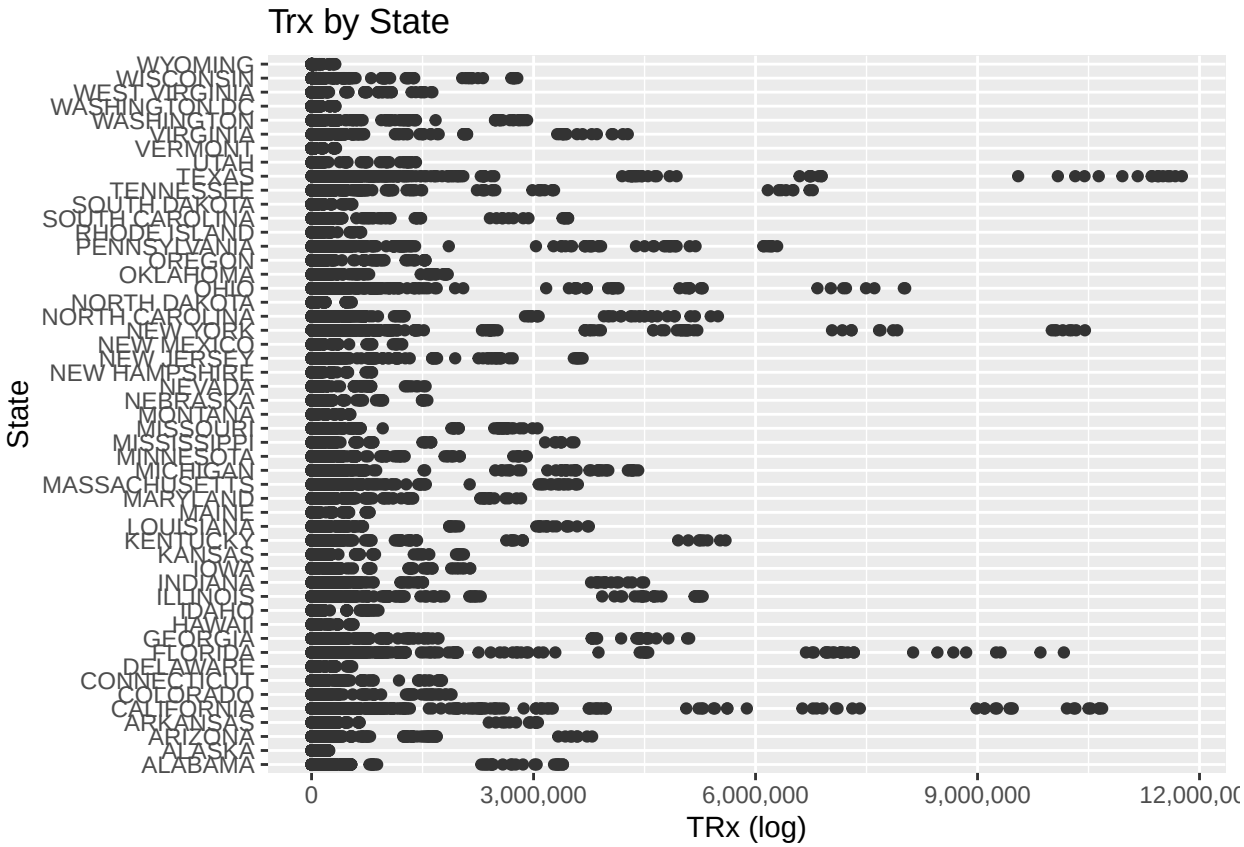
Data Analysis

GRID an CHART Analysis

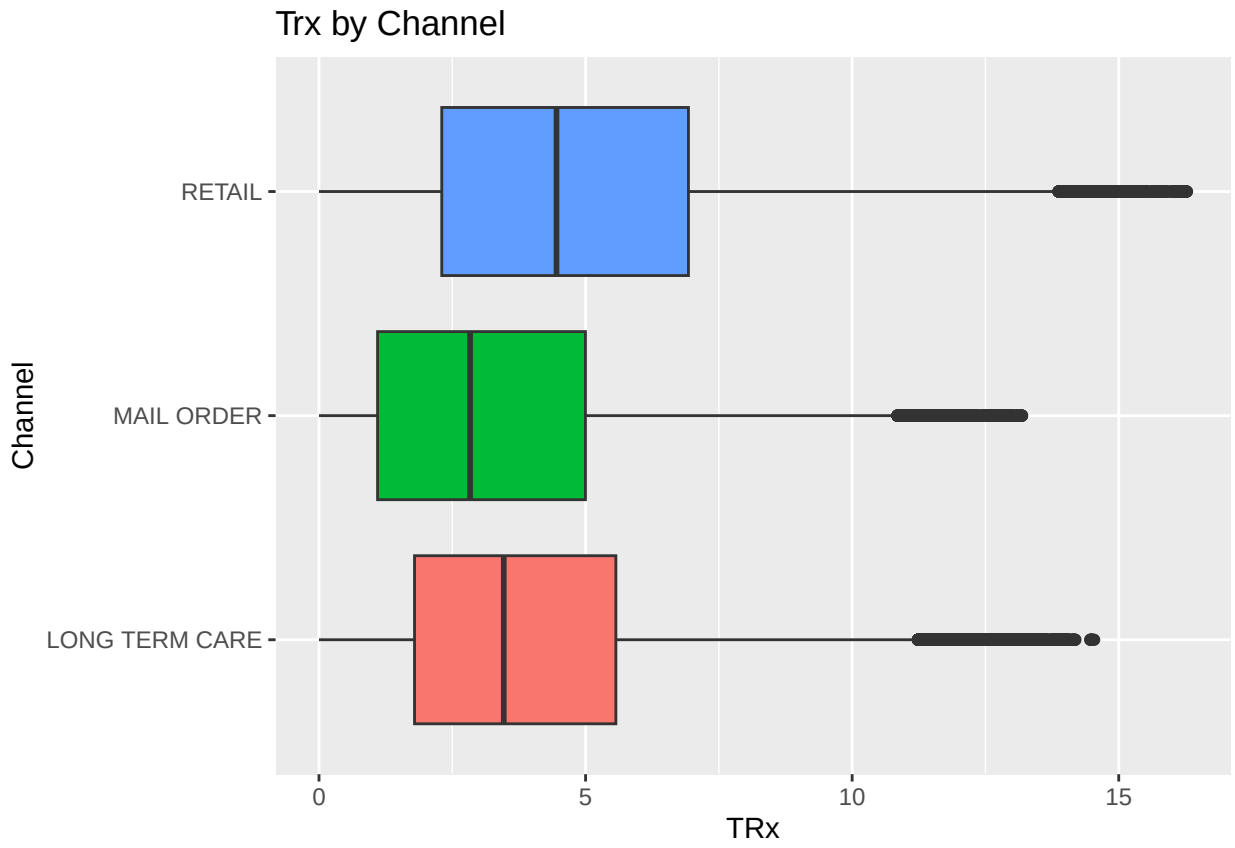
- Since its a large dataset we use boxplot to view the spread of data and used Log to make the data finite and in range

Data spread State wise TRx

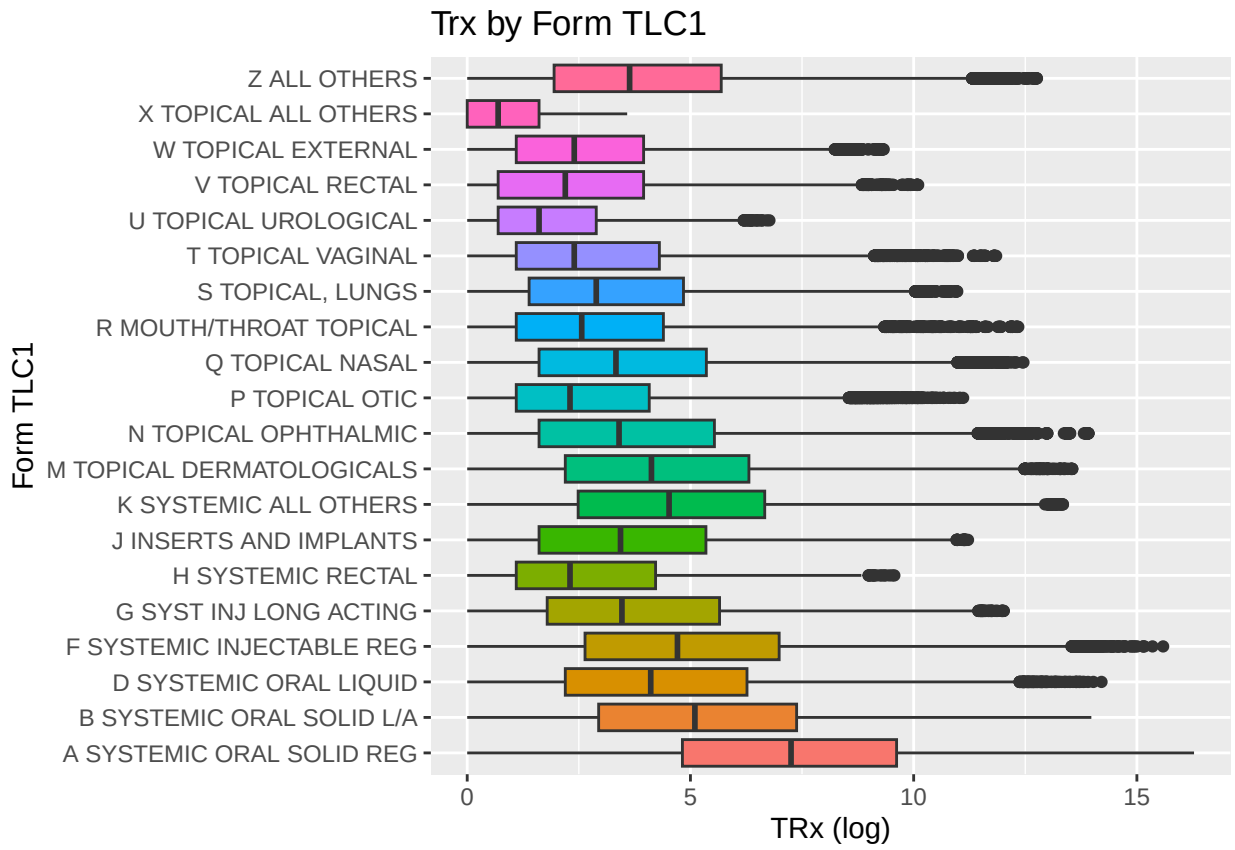
```
ggplot(data = modifiedDataset,
       mapping = aes(y = TRx, x = `ï..State`)) +
  geom_boxplot() +
  theme(legend.position = "none") +
  ggtitle("Trx by State") +
  scale_y_continuous("TRx (log)", labels = scales::comma) +
  scale_x_discrete("State")  +
  coord_flip()
```


Trx by State
State
WYOMING
WISCONSIN
WEST VIRGINIA
WASHINGTON DC
WASHINGTON
VIRGINIA
VERMONT
UTAH
TEXAS
TENNESSEE
SOUTH DAKOTA
SOUTH CAROLINA
RHODE ISLAND
PENNSYLVANIA
OREGON
OKLAHOMA
OHIO
NORTH DAKOTA
NORTH CAROLINA
NEW YORK
NEW MEXICO
NEW JERSEY
NEW HAMPSHIRE
NEVADA
NEBRASKA
MONTANA
MISSOURI
MISSISSIPPI
MINNESOTA
MICHIGAN
MASSACHUSETTS
MARYLAND
MAINE
LOUISIANA
KENTUCKY
KANSAS
IOWA
INDIANA
ILLINOIS
IDAHO
HAWAII
GEORGIA
FLORIDA
DELAWARE
CONNECTICUT
COLORADO
CALIFORNIA
ARKANSAS
ARIZONA
ALASKA
ALABAMA
0
3,000,000
6,000,000
9,000,000
12,000,00
TRx (log)

Channel wise TRx

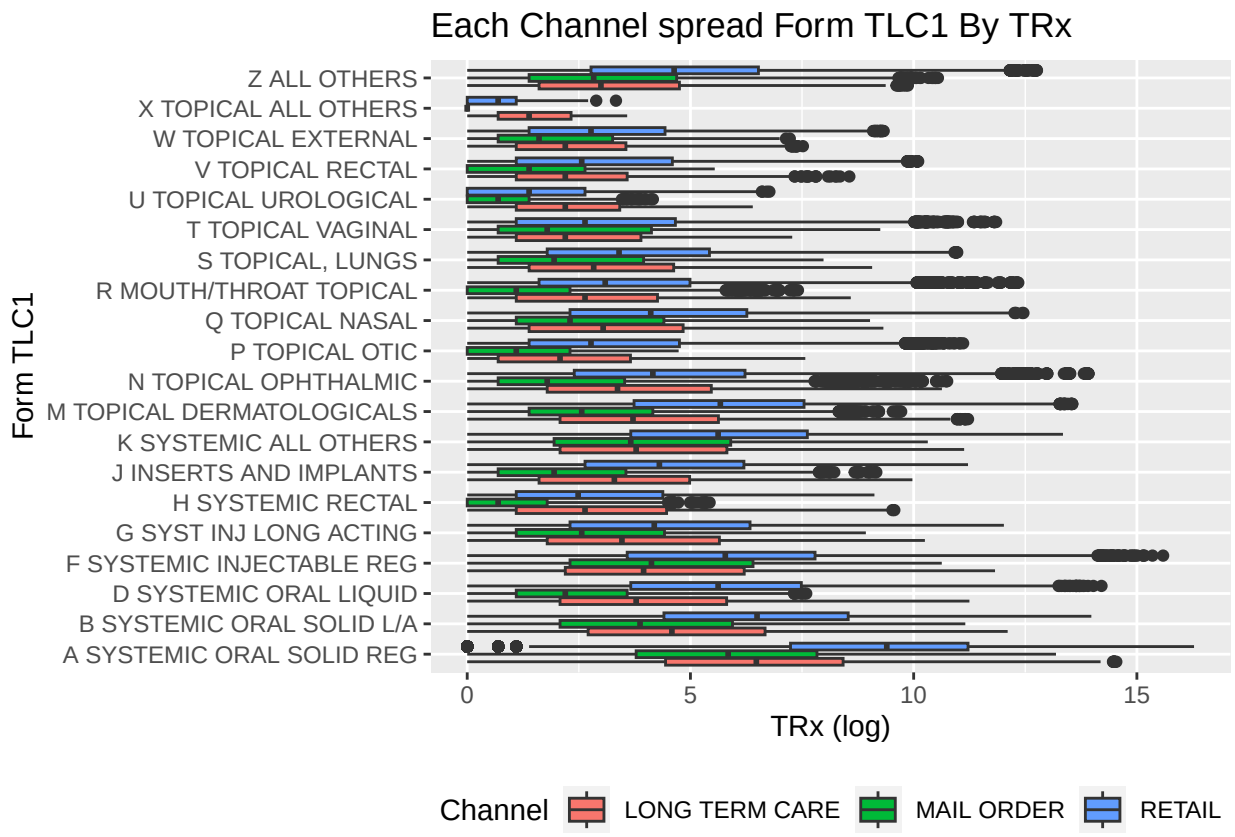


Form TLC1 wise TRx



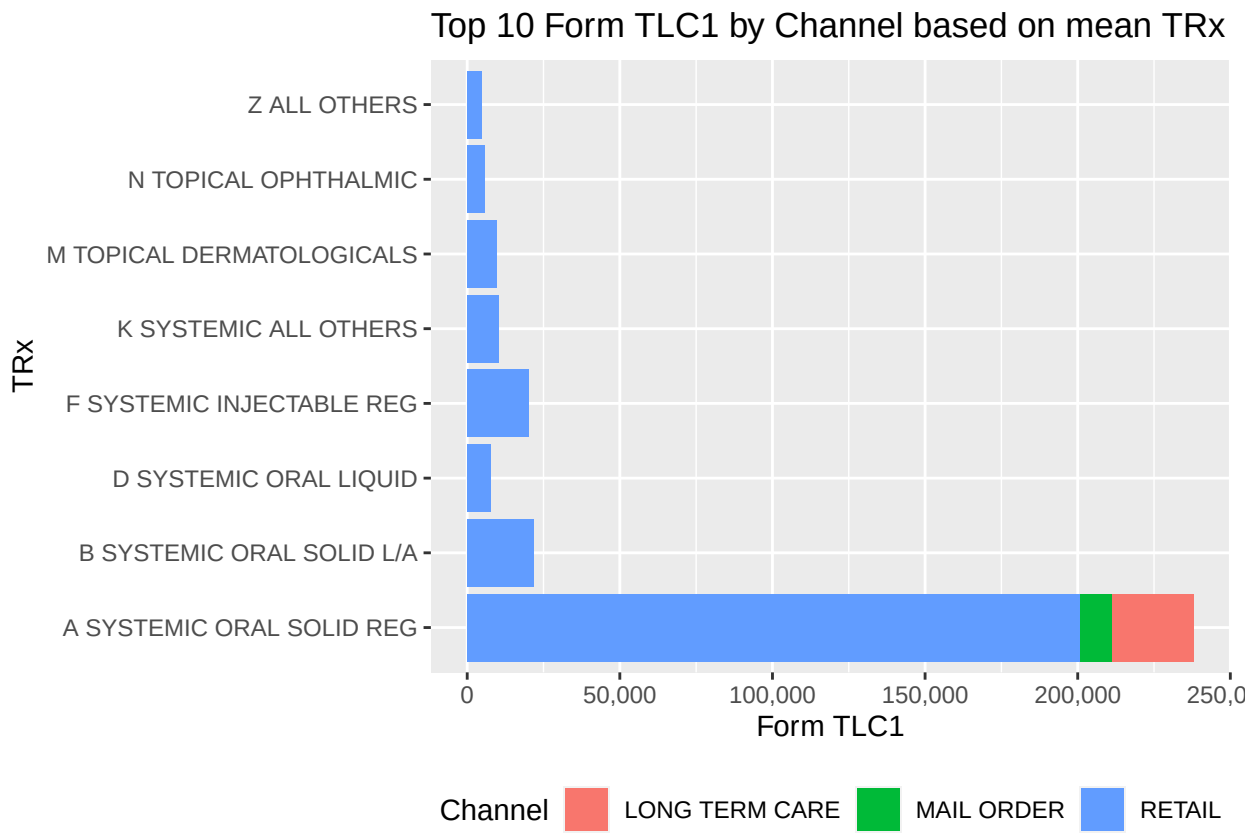
Channel wise Form TLC1 By TRx

```
ggplot(data = modifiedDataset, aes(x = `Form..TLC1.`, y = log(TRx), fill = `Channel`)) +
  geom_boxplot() +
  theme(legend.position = "bottom") +
  ggtitle("Each Channel spread Form TLC1 By TRx") +
  scale_y_continuous("TRx (log)", labels = scales::comma) +
  scale_x_discrete("Form TLC1") +
  coord_flip()
```



Top 10 Form TLC1 by Channel based on mean TRx

```
##filter data
df2 <-
  modifiedDataset %>% group_by(`Form..TLC1.`,`Channel` ) %>%
  summarise(MeanTrx = mean(`TRx`)) %>%
  arrange(desc(MeanTrx)) %>% head(10)

#plot Most Expensive Building Types by Borough
df2 %>%
  arrange(desc(MeanTrx)) %>%
ggplot( aes(x = `Form..TLC1.`, y = MeanTrx, fill = `Channel`)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme(legend.position = "bottom") +
  ggtitle("Top 10 Form TLC1 by Channel based on mean TRx") +
  scale_y_continuous("Form TLC1", labels = scales::comma) +
  scale_x_discrete("TRx")
```



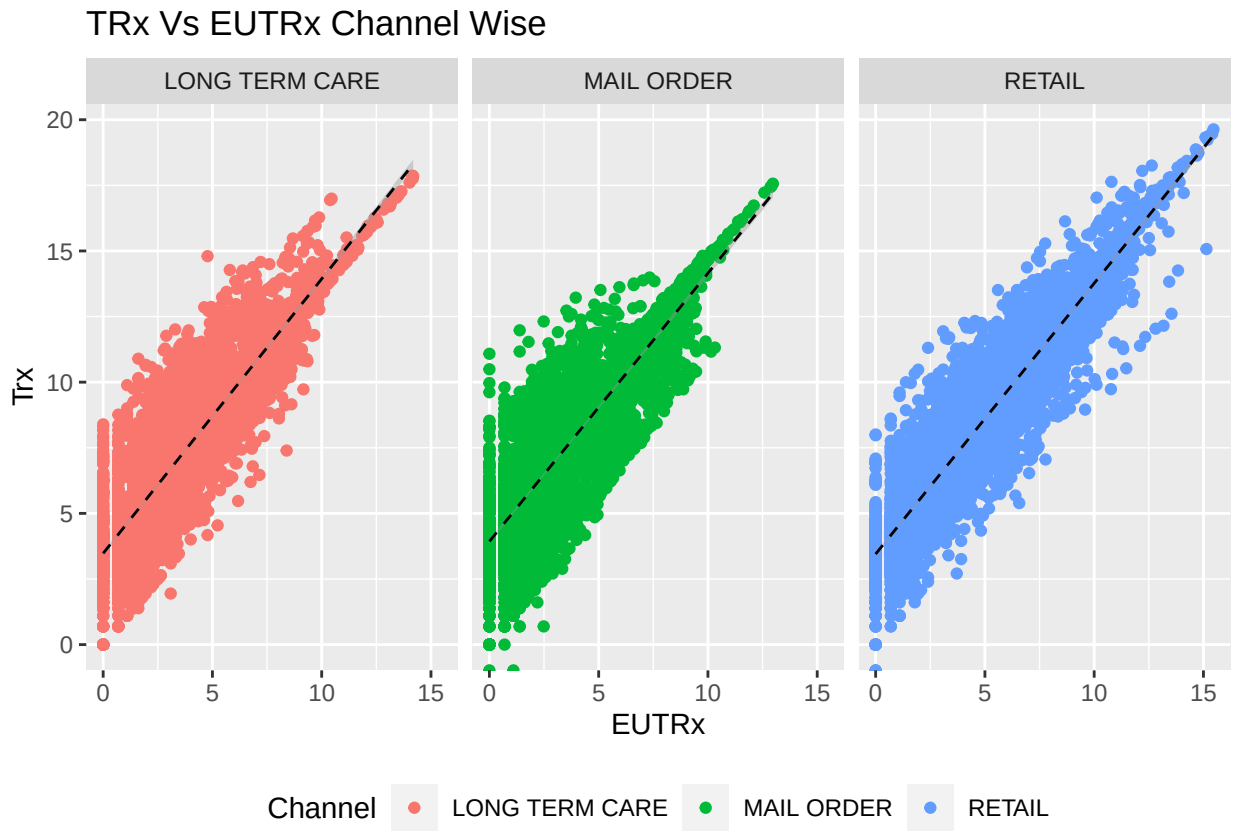
Create random sample of dataset based on channel

```
dat_small <- modifiedDataset %>%
  group_by(Channel) %>%
  slice_sample(n=5000)
sprintf("%d sample of Rows grouping by channel" ,nrow(dat_small))
```

```
## [1] "15000 sample of Rows grouping by channel"
```

Relation metrics NRx and EUTRx with channel wise

```
ggplot(data = dat_small, aes(x = log(TRx), y = log(EUTRx), color = `Channel`)) +
  geom_jitter() +
  geom_smooth(method = "lm", colour="black", size=0.5, linetype = "dashed") +
  theme(legend.position = "bottom") +
  facet_wrap(~ Channel) +
  ggtitle("TRx Vs EUTRx Channel Wise") +
  scale_y_continuous("Trx", labels = scales::comma) +
  scale_x_continuous("EUTRx", labels = scales::comma)
```



Predective Analysis

Spliting the data into Training And Test Set in 80-20 ratio

```
#split it into training set
set.seed(3287)
split = sample.split(modifiedDataset$TRx, SplitRatio = 0.8)
training_set = subset(modifiedDataset, split == TRUE)
test_set = subset(modifiedDataset, split == FALSE)
```

```
## [1] "715656 Rows Training Set"
```

```
## [1] "169285 Rows Test Set"
```

Correlation

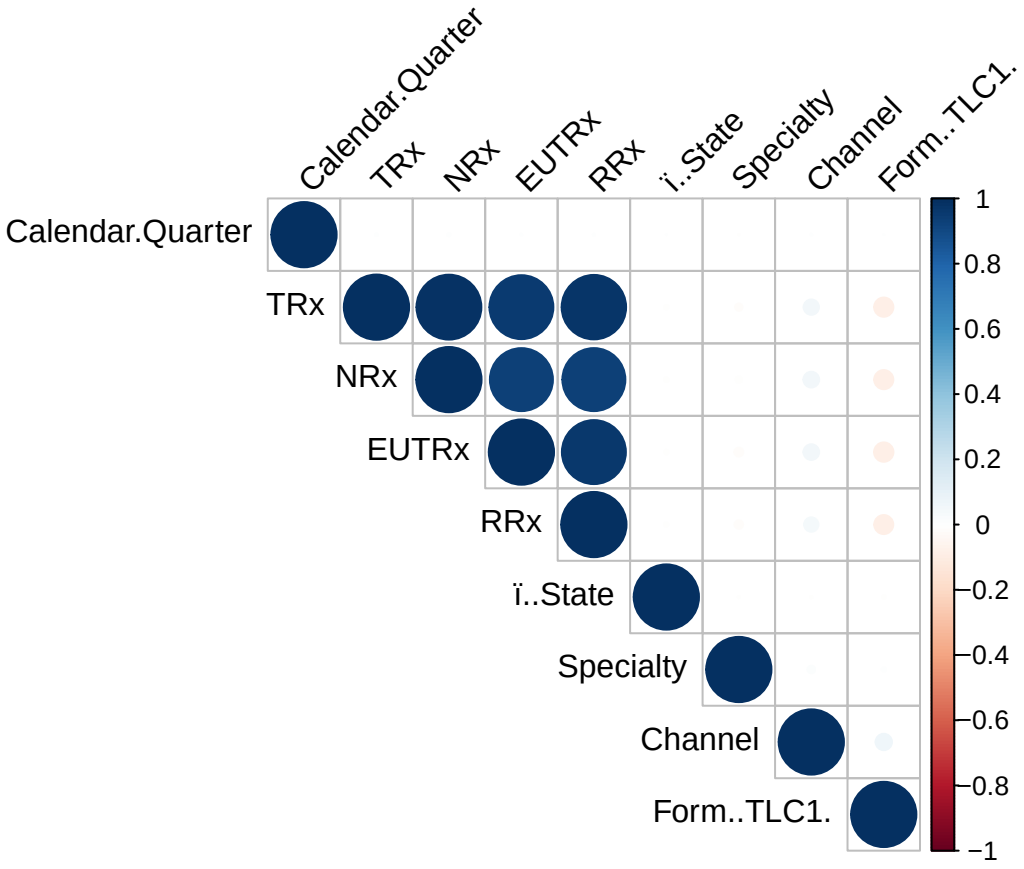
Generating corelated table between the feature

```
library(corrplot)
#calculate correlation
```

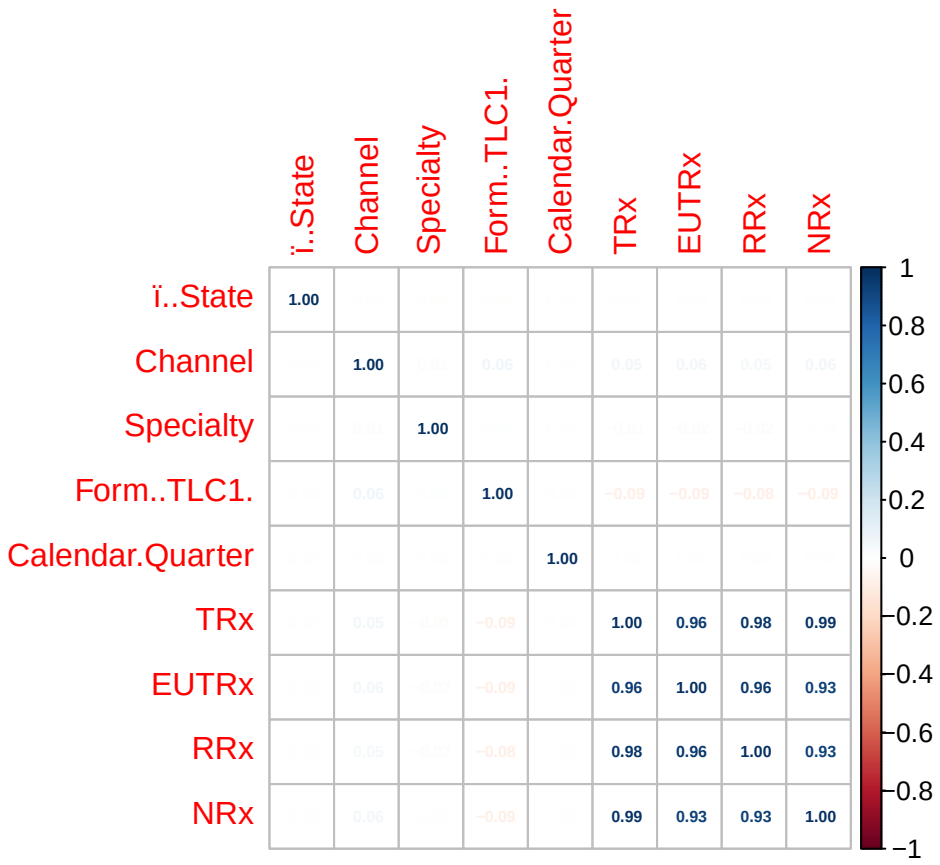
```
distance_corr <- vapply(training_set,
                        function(x) { cor(training_set$TRx, as.numeric(x), use = "pairwise.complete.obs
                        FUN.VALUE = numeric(1))
effect_corr <- vapply(distance_corr, function(x) { ifelse(x >= 0, "Positive", "Negative")},
                      FUN.VALUE = character(1))
var_corr <- names(training_set)

# create correlation table
table1 <- data.frame(var_corr, abs(distance_corr), effect_corr)
table1 <- table1[order(-abs(distance_corr)),]
names(table1) <- c("Column/Variable", "Correlation Size", "Correlation Effect")
table1
```

```
##                    Column/Variable Correlation Size Correlation Effect
## TRx                            TRx      1.000000000           Positive
## NRx                            NRx      0.989034487           Positive
## RRx                            RRx      0.975525059           Positive
## EUTRx                        EUTRx      0.959516302           Positive
## Form..TLC1.            Form..TLC1.      0.086413256           Negative
## Channel                    Channel      0.054999362           Positive
## Specialty                Specialty      0.012164628           Negative
## ï..State                  ï..State      0.004283800           Negative
## Calendar.Quarter Calendar.Quarter      0.001608529           Positive
```

```
temp <- training_set[,c(1:9)]
num <- c(1:9)
temp <- temp %>% mutate_at(num, funs(as.numeric(.)))
#plot table of correlations
corrplot(cor(temp), type = "upper",order = "hclust",
         tl.col = "black", tl.srt = 45)
```




```
corrplot(cor(temp), method="number",number.cex = 0.5)
```



Multi Regression Modal

Handeling the items thats present in test set and not present in traning set

```
#predict Data
data_test_new <- test_set
data_test_new$Specialty[which(!(data_test_new$Specialty %in% unique(training_set$Specialty)))] <- NA
data_test_new$Channel[which(!(data_test_new$Channel %in% unique(training_set$Channel)))] <- NA
data_test_new$ï..State[which(!(data_test_new$ï..State %in% unique(training_set$ï..State)))] <- NA
data_test_new$Form..TLC1.[which(!(data_test_new$Form..TLC1. %in% unique(training_set$Form..TLC1.)))] <-
```

Apply all metrics column in the modal

```
regressor = lm(formula = TRx ~ . ,
               data = training_set)

#Summary of the regression
# summary(regressor)

rSquared<- summary(regressor)$r.squared
standerdError <- summary(regressor)$sigma
sprintf("R Square: %f" ,rSquared)
```

```
## [1] "R Square: 1.000000"
```

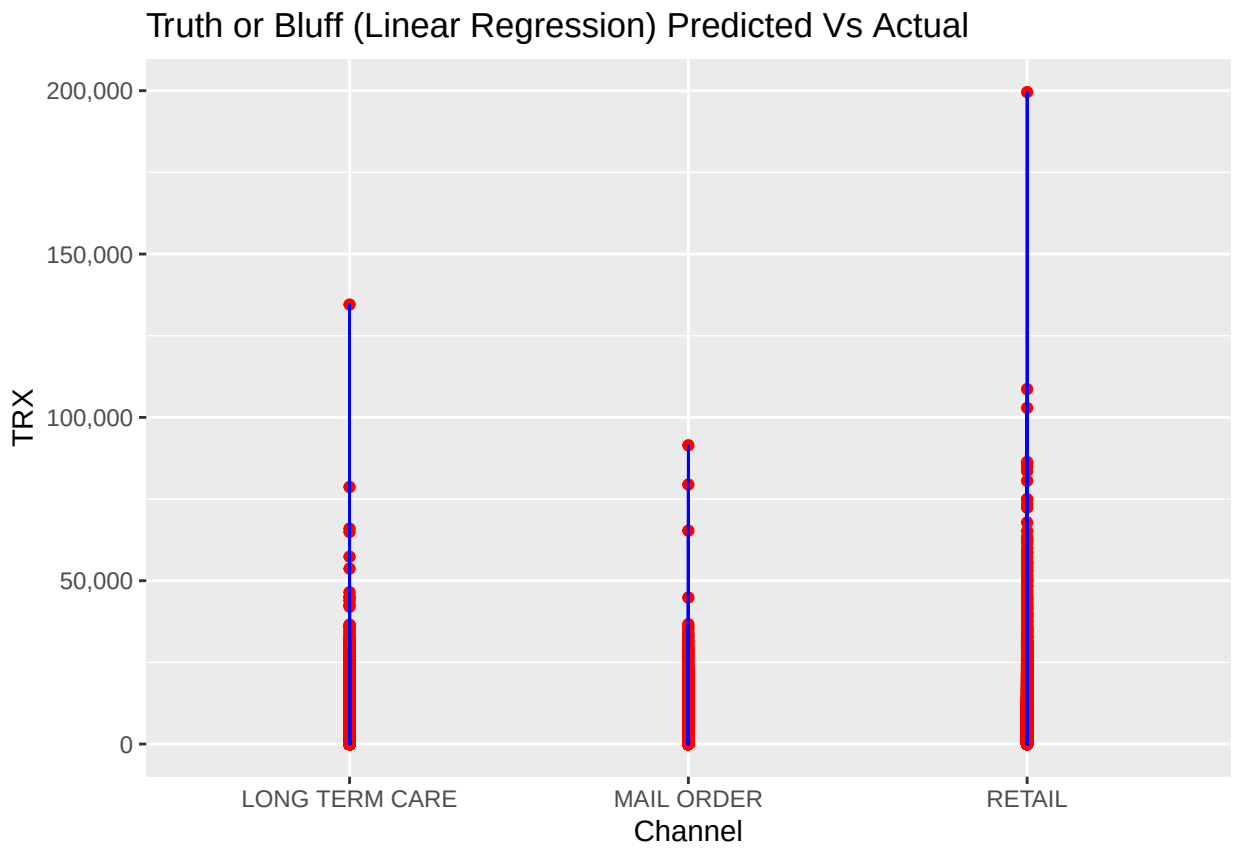
```
sprintf("Standerd Error: %f" ,standerdError)
```

```
## [1] "Standerd Error: 0.384075"
```

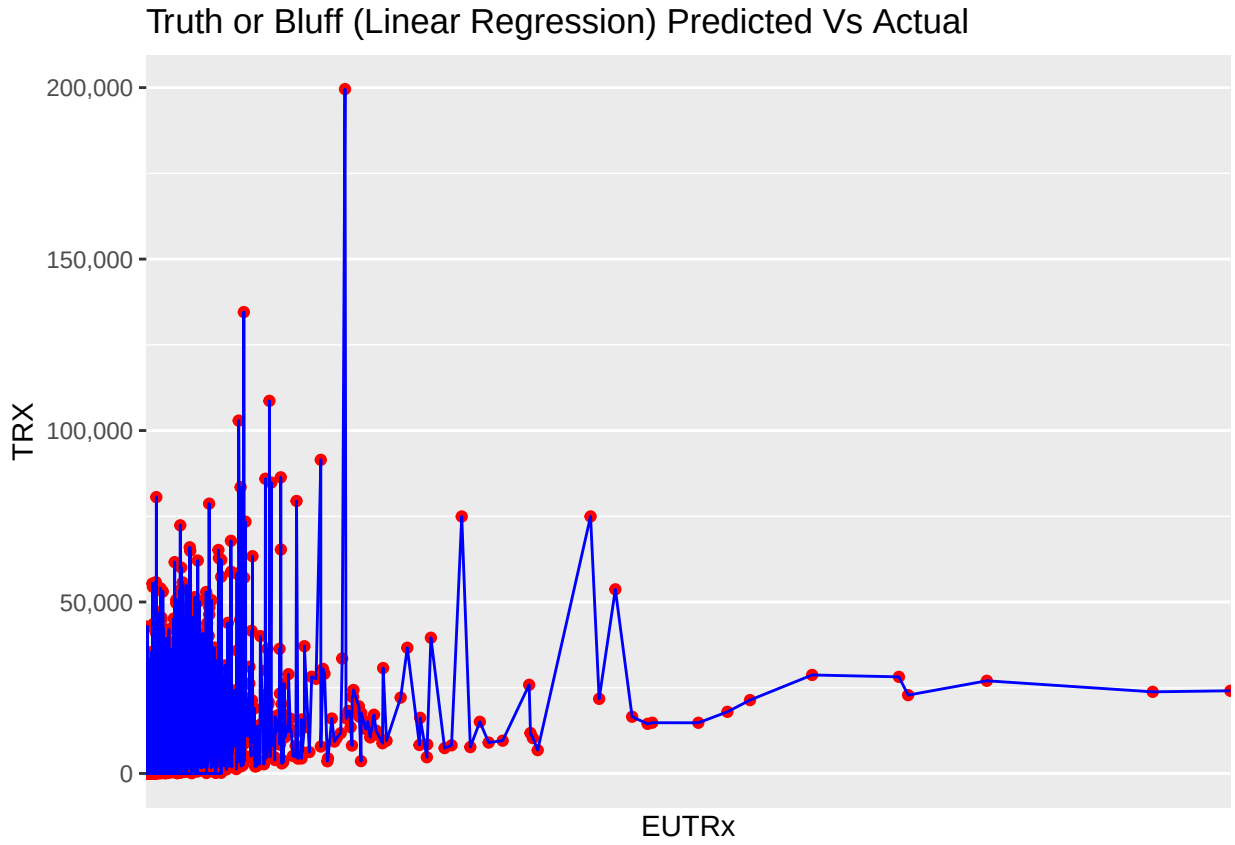
Predict the result

```
y_pred_test = predict(regressor, newdata = data_test_new)


# write_xlsx( data.frame(data_test_new) , "TestData.xlsx")
# write_xlsx( data.frame(y_pred_test) , "predict.xlsx")
```

Visualising the multi Linear Regression results Channel, TRx and Predicted TRx



Visualising the multi Linear Regression results EUTRx, TRx and Predicted TRx



Apply all correlative NRx,RRx,EUTRx,Channel,Form TLC1 for predecting TRX

```
regressor = lm(formula = TRx ~ `NRx` + `RRx` + Channel + `Form..TLC1.` ,
               data = training_set)

rSquared<- summary(regressor)$r.squared
standerdError <- summary(regressor)$sigma
sprintf("R Square: %f" ,rSquared)
```

```
## [1] "R Square: 1.000000"
```

```
sprintf("Standerd Error: %f" ,standerdError)
```

```
## [1] "Standerd Error: 0.384119"
```

```
y_pred_test = predict(regressor, newdata =  data_test_new)
```

Apply all metrics NRx ,RRx for predecting TRX

```
regressor = lm(formula = TRx ~ `NRx` + `RRx` +`EUTRx` ,
               data = training_set)

#Summary of the regression
summary(regressor)
```

```
##
## Call:
## lm(formula = TRx ~ NRx + RRx + EUTRx, data = training_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.02622 -0.01571 -0.01571 -0.01571  1.09558
##
## Coefficients:
##              Estimate Std. Error   t value Pr(>|t|)
## (Intercept) 1.571e-02  4.555e-04 3.448e+01   <2e-16 ***
## NRx         1.000e+00  1.373e-08 7.283e+07   <2e-16 ***
## RRx         1.000e+00  2.684e-08 3.726e+07   <2e-16 ***
## EUTRx       6.811e-11  1.776e-10 3.840e-01    0.701
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3841 on 715652 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 4.202e+16 on 3 and 715652 DF,  p-value: < 2.2e-16
```

```
rSquared<- summary(regressor)$r.squared
sprintf("R Square: %f" ,rSquared)
```

```
## [1] "R Square: 1.000000"
```

```
standerdError <- summary(regressor)$sigma
sprintf("Standerd Error: %f" ,standerdError)
```

```
## [1] "Standerd Error: 0.384136"
```

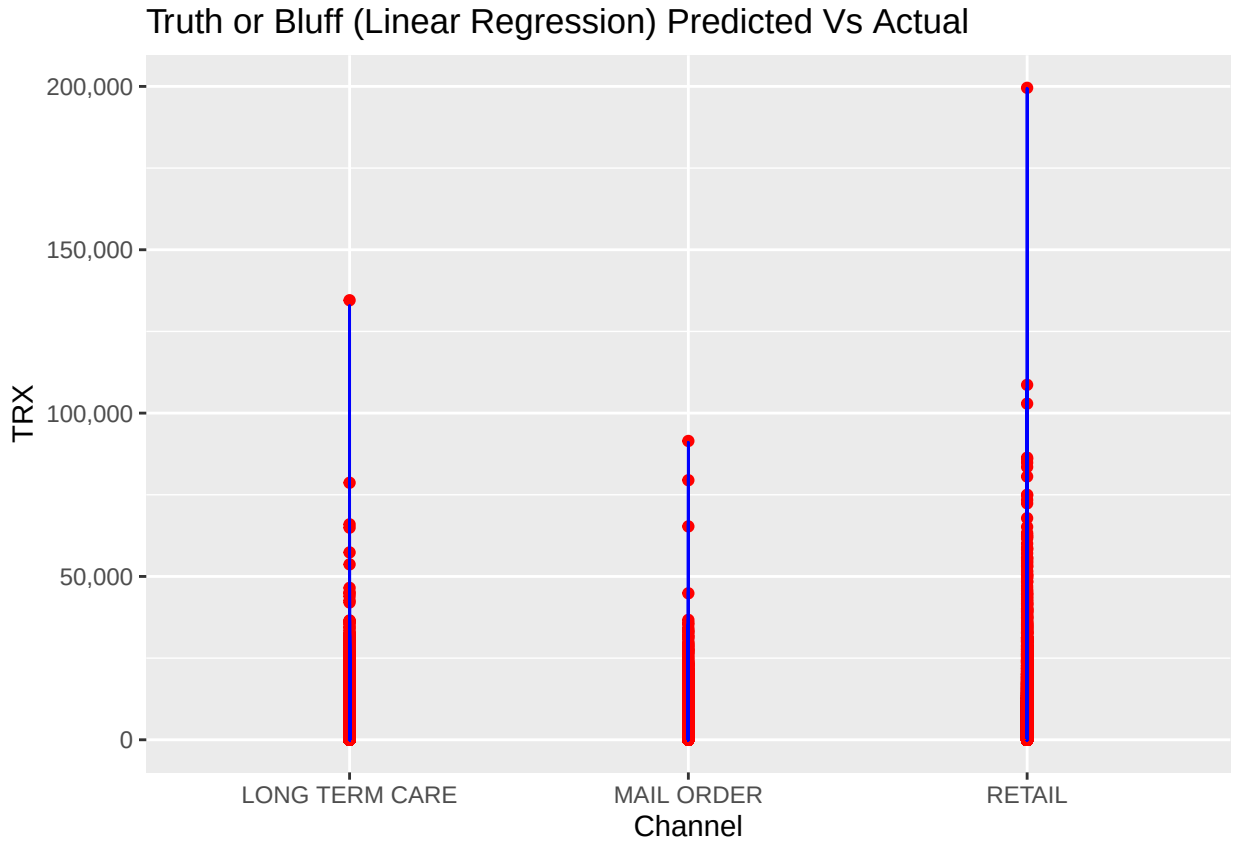
```
y_pred_test = predict(regressor, newdata = data_test_new )
```

Random Forest Modal

Modal with NRx,RRx for predecting TRX with 2 node

```
library(randomForest)

# now try with 10, 100, 500 tree
regressor=randomForest(TRx ~  `NRx` + `RRx`  , data=training_set, na.action=na.omit,ntree=2)

y_pred =predict(regressor, newdata = test_set )
# write_xlsx( data.frame(y_pred) , "PredDiadata.xlsx")
```

Visualising the multi Random Forest results Channel, TRx and Predicted TRx



Visualising the multi Random Forest results EUTRx, TRx and Predicted TRx