CLV Model Building

Josh Kong

10/19/2020

OBJECTIVE: I am going to be building a model that will predict each customer's CLV within a year, and within 5 years. I'm going to be using a BTYD (buy til you die) model to achieve this.

NOTE: The retail dataset I'm currently using ranges from December 1, 2010 - December 9, 2011. Approximately 1 year.

I'm going to begin with filtering out customer's without an ID. We are trying to predict the CLV of customers, and we wont be able to do that with customers who do not have an ID. I'm also going to be joining the rfm scores and value of the customer to use as features for building the model.

```
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------------------------------ t
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ---------------------------------------------------------------------------------- tidyver
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(CLVTools)
library(tidymodels)
```

```
## -- Attaching packages ----------------------------------------------------------------------------- ti
```

```
## v broom     0.7.0      v recipes   0.1.13
## v dials     0.0.8      v rsample   0.0.7
## v infer     0.5.3      v tune      0.1.1
## v modeldata 0.0.2      v workflows 0.1.3
## v parsnip   0.1.3      v yardstick 0.0.7
```

```
## -- Conflicts ----------------------------------------------------------------------------------- tidymode
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
```

```
retail <- read_csv("retail_cleaned.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]

## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   InvoiceNo = col_character(),
##   StockCode = col_character(),
##   Quantity = col_double(),
##   InvoiceDate = col_date(format = ""),
##   UnitPrice = col_double(),
##   CustomerID = col_double(),
##   Country = col_character(),
##   Description = col_character(),
##   sales = col_double()
## )
```

```
rfm_scores <- read_csv("rfm_scores.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]

## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   CustomerID = col_double(),
##   recency = col_double(),
##   r_clus = col_double(),
##   frequency = col_double(),
##   f_clus = col_double(),
##   money_value = col_double(),
##   m_clus = col_double(),
##   score = col_double(),
##   value = col_character()
## )
```

```
#joining rfm scores with retail data
rfm_join <- rfm_scores %>%
  select(value, CustomerID)

retail_filtered <- retail %>%
  filter(!is.na(CustomerID)) %>%
  left_join(rfm_join, by = "CustomerID") %>%
  select(-X1)
```

I'm now going to be calculating the CLV of each customer. Because there is no cost specified in this dataset, I'm going to be using revenue as CLV.

```
#calculating CLV of customers
customer_clv <- retail_filtered %>%
  group_by(CustomerID) %>%
  summarise(CLV = sum(sales))
```

```
## ‘summarise()‘ ungrouping output (override with ‘.groups‘ argument)
```

The model doesn't take in negative values for sales. The negative value usually means that a customer made a return. I will filter out all the invoices where the customer made a return.

```
negative_sales <- retail_filtered %>%
  filter(sales < 0) %>%
  select(InvoiceNo)

‘%notin%‘ <- Negate(‘%in%‘)

#getting rid of all the invoice numbers that ended in a return
retail_filtered <- retail_filtered %>%
  filter(InvoiceNo %notin% negative_sales$InvoiceNo, CustomerID != 12346)
```

Going to create a CLV model. Only going to use 3 months worth of data to train the model.

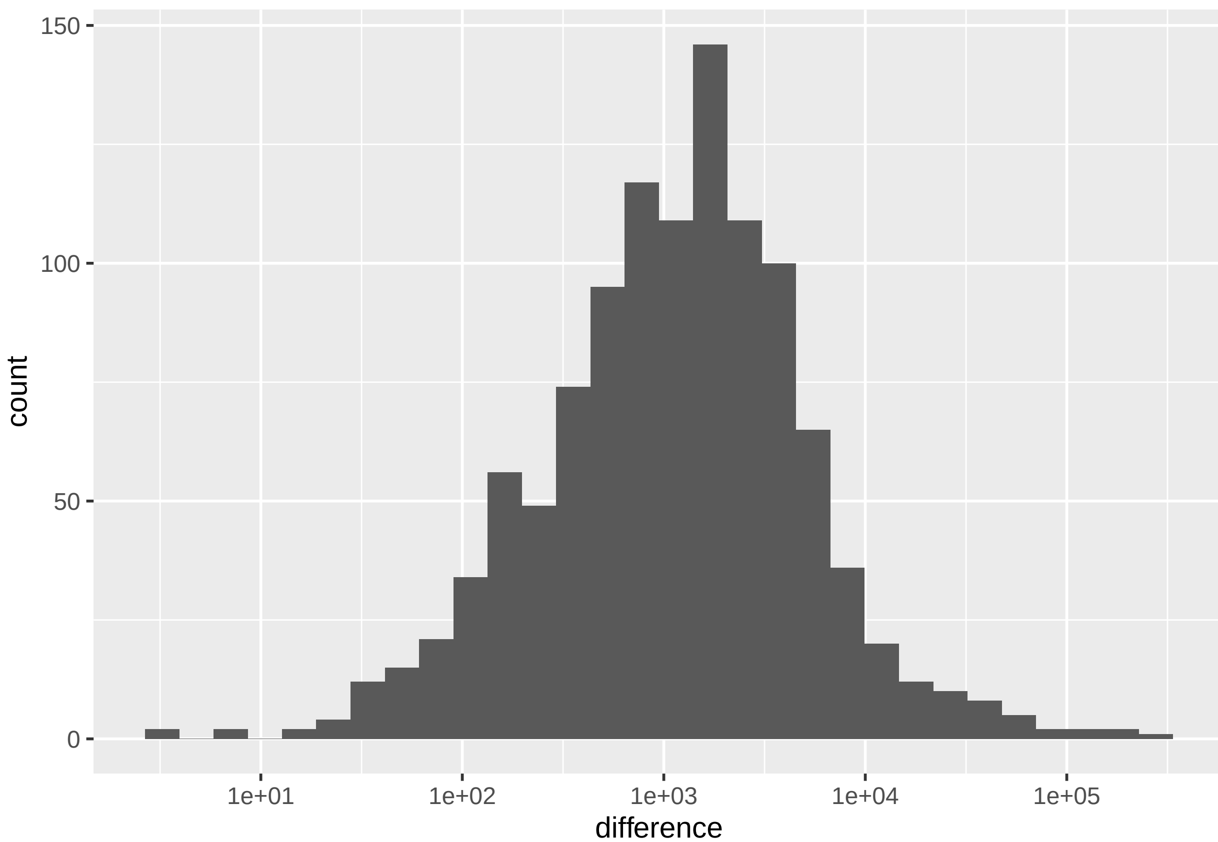
```
library(BTYDplus)
retail_train_3 <- retail_filtered %>%
  filter(InvoiceDate < "2011-03-01")
colnames(retail_train_3)[c(4,6)] <- c("date","cust")

customer_rdf_3 <- BTYDplus::elog2cbs(retail_train_3,
unit = 'days',
T.cal = max(retail_train_3$date),
T.tot = max(retail_train_3$date))
customer_rdf_3$sales_avg = customer_rdf_3$sales / (customer_rdf_3$x + 1)

bgnbd_rdf = customer_rdf_3
bgnbd_rdf$T.star = 273

params_bgnbd = BTYD::bgnbd.EstimateParameters(bgnbd_rdf)
bgnbd_rdf$predicted_bgnbd = BTYD::bgnbd.ConditionalExpectedTransactions(
params = params_bgnbd,
T.star = bgnbd_rdf$T.star,
x = bgnbd_rdf$x,
t.x = bgnbd_rdf$t.x,
T.cal = bgnbd_rdf$T.cal
)
bgnbd_rdf$predicted_clv = bgnbd_rdf$sales_avg * bgnbd_rdf$predicted_bgnbd
```

After using only the first 3 months of the training data to predict the future 1 year clv, let's compare how the actual 1 year clv and the predicted clv differ.

```
predictions_comparison <- bgnbd_rdf %>%
  select(predicted_clv, cust)

colnames(predictions_comparison)[2] <- "CustomerID"
predictions_comparison$CustomerID <- as.double(predictions_comparison$CustomerID)
colnames(retail_train_3)[6] <- "CustomerID"

customers_3months <- retail_train_3 %>%
  distinct(CustomerID)
```

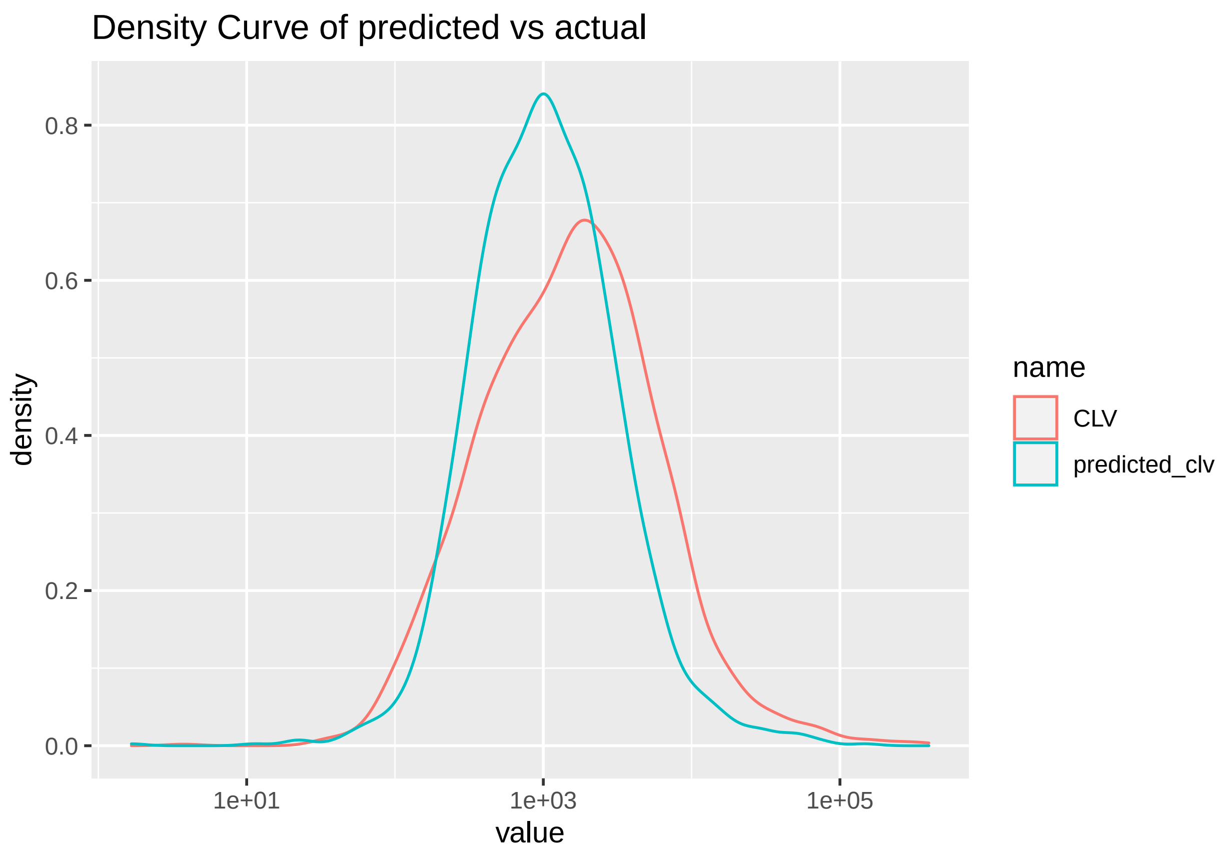
```r
predicted_vs_actual <- customer_clv %>%
  filter(CustomerID %in% customers_3months$CustomerID) %>%
  left_join(predictions_comparison, by = "CustomerID") %>%
  mutate(difference = CLV - predicted_clv)

predicted_vs_actual %>%
  ggplot(aes(difference)) +
  geom_histogram() +
  scale_x_log10()
```

```
## Warning in self$trans$transform(x): NaNs produced

## Warning: Transformation introduced infinite values in continuous x-axis

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 571 rows containing non-finite values (stat_bin).
```




```r
predicted_vs_actual %>%
  pivot_longer(CLV:predicted_clv) %>%
  ggplot(aes(value, color = name)) +
  geom_density() +
  scale_x_log10() +
  labs(title = "Density Curve of predicted vs actual")
```

```
## Warning in self$trans$transform(x): NaNs produced

## Warning in self$trans$transform(x): Transformation introduced infinite values in
## continuous x-axis

## Warning: Removed 4 rows containing non-finite values (stat_density).
```




```
predicted_vs_actual %>%
  summarise(mean(abs(difference)))
```

```
## # A tibble: 1 x 1
##   `mean(abs(difference))`
##                     <dbl>
## 1                   2743.
```

Looks like the predicted CLV is usually underestimating how valuable the customer will be. While testing the model, I got a mean absolute error of 2743 Using this model will lead to us undervaluing a lot of customers in the end. However, customers that we do consider valuable in this model, will truly be valuable.

Model on entire set

After testing the model, it appears that we tend to overvalue our customers with this BTYD model. However, I'm still going to use this model to predict which customers will remain valuable to to online retail store after a year and 3 years. Because the model has shown to undervalue the customer's future clv, customer's that we see as valuable with the predicted CLV, will most likely be valuable in the future as well.

```r
head(retail_filtered)
```

```
## # A tibble: 6 x 10
##   InvoiceNo StockCode Quantity InvoiceDate UnitPrice CustomerID Country
##   <chr>     <chr>        <dbl> <date>          <dbl>      <dbl> <chr>
## 1 536365    85123A           6 2010-12-01       2.55      17850 United~
## 2 536365    85123A           6 2010-12-01       2.55      17850 United~
## 3 536365    85123A           6 2010-12-01       2.55      17850 United~
## 4 536365    71053            6 2010-12-01       3.39      17850 United~
## 5 536365    71053            6 2010-12-01       3.39      17850 United~
## 6 536365    84406B           8 2010-12-01       2.75      17850 United~
## # ... with 3 more variables: Description <chr>, sales <dbl>, value <chr>
```

```r
colnames(retail_filtered)[c(4,6)] <- c("date","cust")

customer_rdf_1year <- BTYDplus::elog2cbs(retail_filtered,
unit = 'days',
T.cal = max(retail_filtered$date),
T.tot = max(retail_filtered$date))
customer_rdf_1year$sales_avg = customer_rdf_1year$sales / (customer_rdf_1year$x + 1)

bgnbd_rdf = customer_rdf_1year
bgnbd_rdf$T.star = 365

params_bgnbd = BTYD::bgnbd.EstimateParameters(bgnbd_rdf)
bgnbd_rdf$predicted_bgnbd = BTYD::bgnbd.ConditionalExpectedTransactions(
params = params_bgnbd,
T.star = bgnbd_rdf$T.star,
x = bgnbd_rdf$x,
t.x = bgnbd_rdf$t.x,
T.cal = bgnbd_rdf$T.cal
)
bgnbd_rdf$predicted_clv = bgnbd_rdf$sales_avg * bgnbd_rdf$predicted_bgnbd

#these are our 1 year later predictions

head(bgnbd_rdf)
```

```
##    cust x t.x     litt   sales      first T.cal sales_avg T.star
## 1 12347 6 365 24.41755 5468.17 2010-12-07   367  781.1671    365
## 2 12348 3 283 13.09067 1797.24 2010-12-16   358  449.3100    365
## 3 12349 0   0  0.00000 2079.46 2011-11-21    18 2079.4600    365
## 4 12350 0   0  0.00000  349.40 2011-02-02   310  349.4000    365
## 5 12352 6 260 17.81394 2546.09 2011-02-16   296  363.7271    365
## 6 12353 0   0  0.00000   89.00 2011-05-19   204   89.0000    365
##   predicted_bgnbd predicted_clv
## 1       5.7112929    4461.47436
## 2       3.2691414    1468.85792
## 3       3.4689035    7213.44600
## 4       0.7967364     278.37971
## 5       6.8202858    2480.72307
## 6       1.1060178      98.43558
```

```r
bgnbd_rdf_predicted1year <- bgnbd_rdf %>%
  select(cust, predicted_1yr_clv = predicted_clv)
#joining it with our original dataset
retail_predicted_clv <- retail_filtered %>%
  left_join(bgnbd_rdf_predicted1year, by = "cust")
```

We have our 1 year predicitions. Now I'm going to make 3 year predictions.

```r
#1095
customer_rdf_3year <- BTYDplus::elog2cbs(retail_filtered,
unit = 'days',
T.cal = max(retail_filtered$date),
T.tot = max(retail_filtered$date))
customer_rdf_3year$sales_avg = customer_rdf_3year$sales / (customer_rdf_3year$x + 1)

bgnbd_rdf = customer_rdf_3year
bgnbd_rdf$T.star = 1095

params_bgnbd = BTYD::bgnbd.EstimateParameters(bgnbd_rdf)
bgnbd_rdf$predicted_bgnbd = BTYD::bgnbd.ConditionalExpectedTransactions(
params = params_bgnbd,
T.star = bgnbd_rdf$T.star,
x = bgnbd_rdf$x,
t.x = bgnbd_rdf$t.x,
T.cal = bgnbd_rdf$T.cal
)
bgnbd_rdf$predicted_clv = bgnbd_rdf$sales_avg * bgnbd_rdf$predicted_bgnbd

#these are our 3 year later predictions

head(bgnbd_rdf)
```

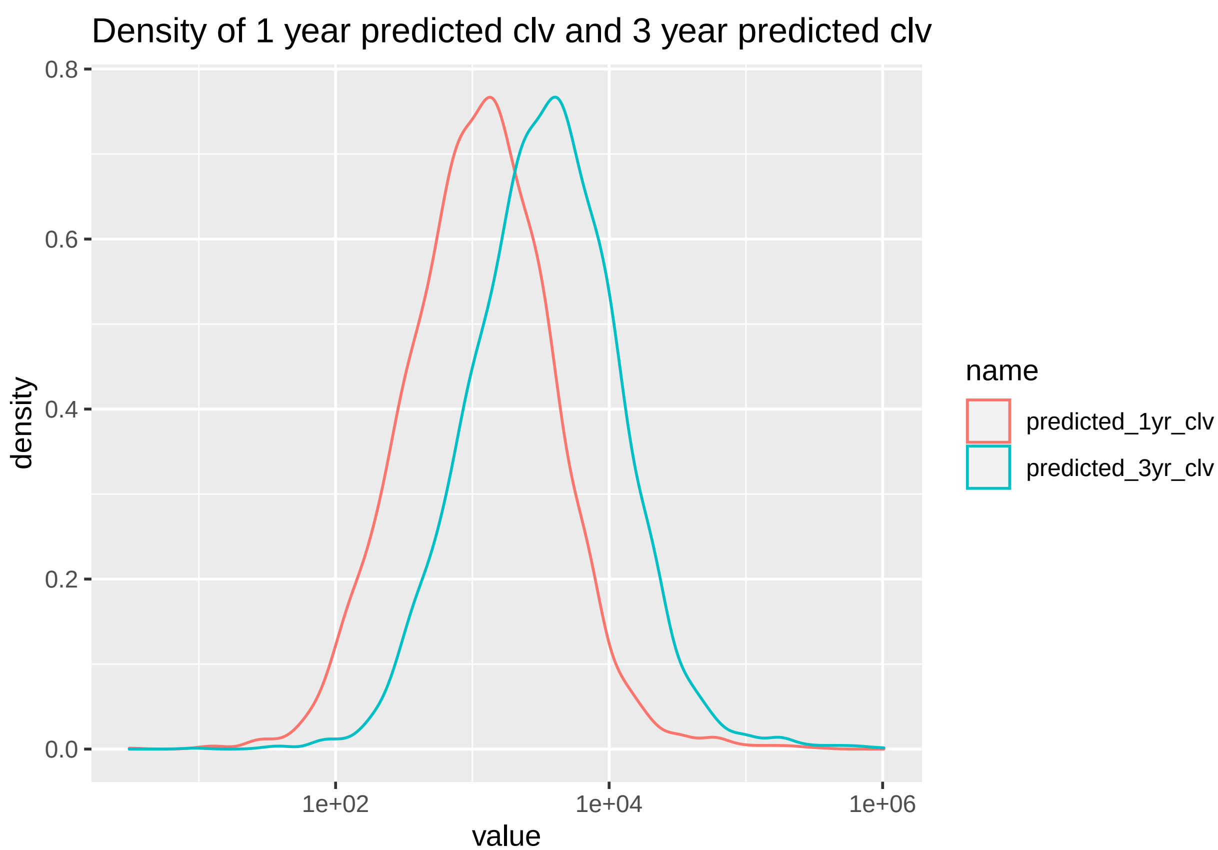
```
##    cust x t.x     litt   sales      first T.cal sales_avg T.star
## 1 12347 6 365 24.41755 5468.17 2010-12-07   367  781.1671   1095
## 2 12348 3 283 13.09067 1797.24 2010-12-16   358  449.3100   1095
## 3 12349 0   0  0.00000 2079.46 2011-11-21    18 2079.4600   1095
## 4 12350 0   0  0.00000  349.40 2011-02-02   310  349.4000   1095
## 5 12352 6 260 17.81394 2546.09 2011-02-16   296  363.7271   1095
## 6 12353 0   0  0.00000   89.00 2011-05-19   204   89.0000   1095
##   predicted_bgnbd predicted_clv
## 1       17.116241    13370.6452
## 2        9.800514     4403.4688
## 3       10.393948    21613.7990
## 4        2.389425      834.8650
## 5       20.436269     7433.2259
## 6        3.316571      295.1749
```

```r
bgnbd_rdf_predicted3year <- bgnbd_rdf %>%
  select(cust, predicted_3yr_clv = predicted_clv)
#joining it with our original dataset
retail_predicted_clv <- retail_predicted_clv %>%
  left_join(bgnbd_rdf_predicted3year, by = "cust")
```

```r
retail_predicted_clv %>%
  distinct(cust,predicted_1yr_clv,predicted_3yr_clv) %>%
  pivot_longer(predicted_1yr_clv:predicted_3yr_clv) %>%
  ggplot(aes(value, color = name)) +
  geom_density() +
  scale_x_log10() +
  labs(title = "Density of 1 year predicted clv and 3 year predicted clv")
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```

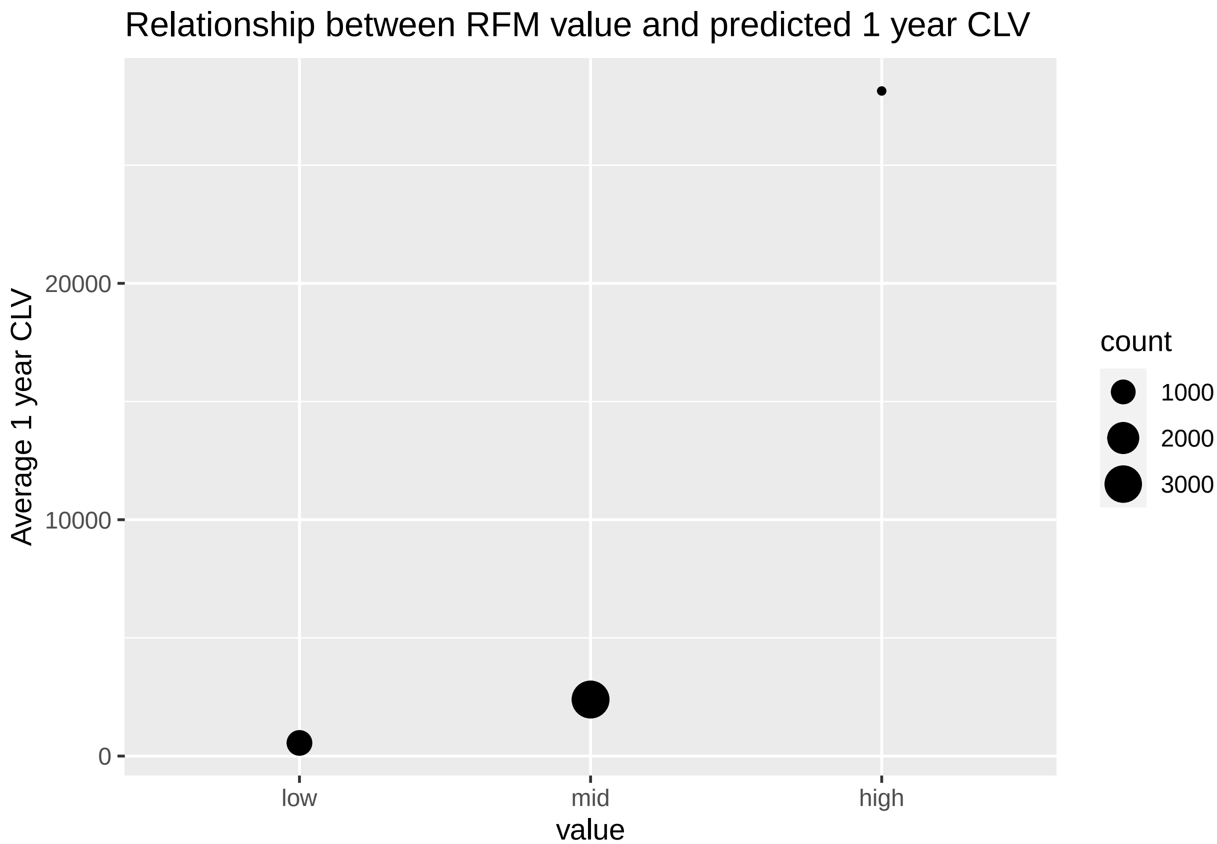
```
## Warning: Removed 2 rows containing non-finite values (stat_density).
```



Is there a relationship beween the value of a customer based off the rfm model, and the predicted clv values?

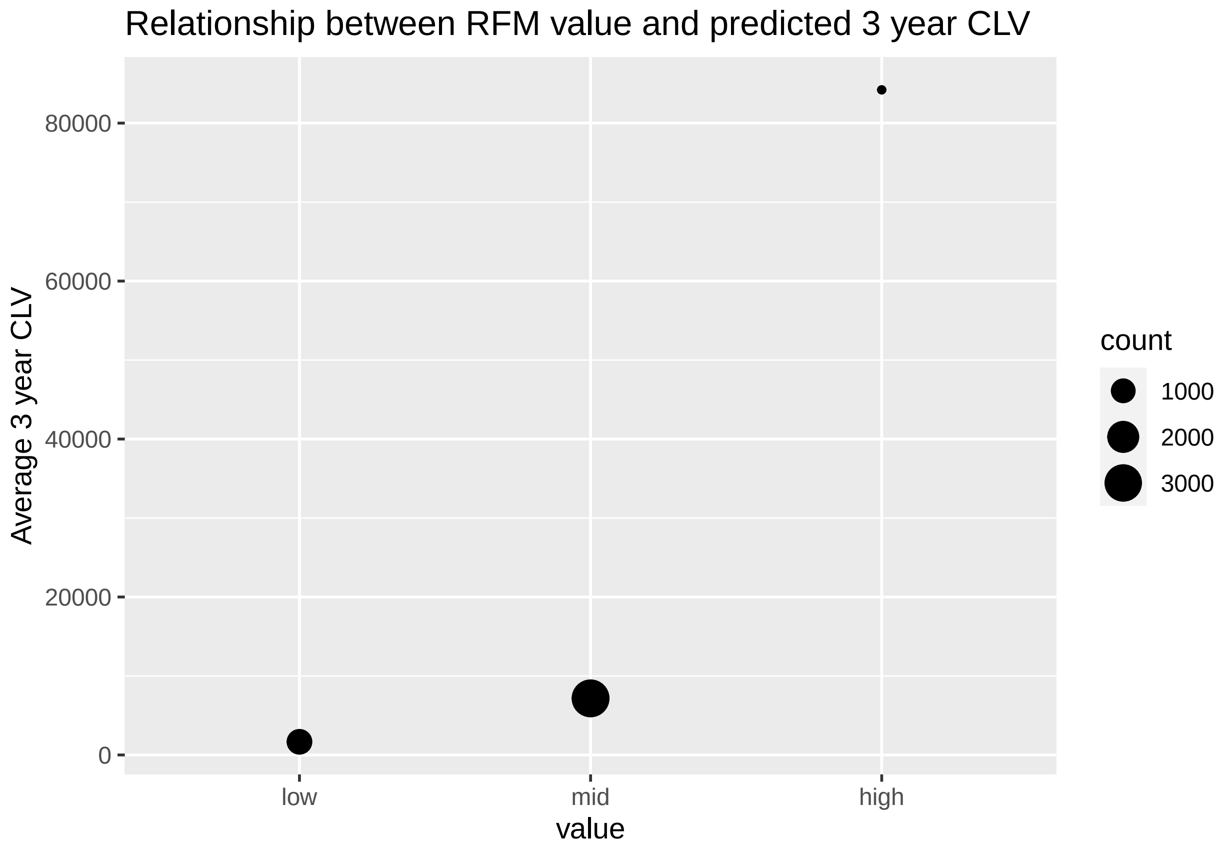
```r
retail_predicted_clv %>%
  distinct(cust,value, predicted_1yr_clv) %>%
  group_by(value) %>%
  summarise(average_1yr_clv = mean(predicted_1yr_clv),
            count = n()) %>%
  mutate(value = fct_reorder(value, average_1yr_clv)) %>%
  ungroup() %>%
  ggplot(aes(value, average_1yr_clv, size= count)) +
  geom_point() +
  labs(title = "Relationship between RFM value and predicted 1 year CLV", y = "Average 1 year CLV")
```

```
## 'summarise()' ungrouping output (override with '.groups' argument)
```




```
retail_predicted_clv %>%
  distinct(cust,value, predicted_3yr_clv) %>%
  group_by(value) %>%
  summarise(average_3yr_clv = mean(predicted_3yr_clv),
            count = n()) %>%
  mutate(value = fct_reorder(value, average_3yr_clv)) %>%
  ungroup() %>%
  ggplot(aes(value, average_3yr_clv, size= count)) +
  geom_point() +
  labs(title = "Relationship between RFM value and predicted 3 year CLV", y = "Average 3 year CLV")
```

```
## 'summarise()' ungrouping output (override with '.groups' argument)
```



Final result: Fit a BTYD model to the online retail dataset. After training and testing the model, I found that the model tended to undervalue the customers. Although the predicted CLV tended to undervalue the customers, we can infer that customers that are considered high value with the predicted CLV will most likely be true.

Managed to fit a model that predicted the CLV of the customers within the next year, and next three years.