KHAFAJI_FA6

FA6 - Customer Segmentation in an E-commerce Business

The situation:

You are working for an e-commerce company, and your task is to segment customers based on their purchasing behavior. The goal is to predict which customer segment a new customer belongs to based on demographic and behavioral data.

the dataset containing the following features:

- Customer ID (Unique Identifier)
- Age (Continuous variable)
- Annual Income (Continuous variable, in thousands of dollars)
- Gender (Categorical: Male/Female)
- Product Category Purchased (Categorical: Electronics, Fashion, Home, Books, Others)
- Average Spend per Visit (Continuous variable, in dollars)
- Number of Visits in Last 6 Months (Discrete variable)
- Customer Segment (Categorical target variable: Budget Shopper, Regular Shopper, Premium Shopper)

The target variable, Customer Segment, has three categories:

- Budget Shopper (Low spenders)
- Regular Shopper (Moderate spenders)
- Premium Shopper (High spenders)

Data exploration

First, let's load the data:

```r
ecomm_data <- read_csv("customer_segmentation.csv", show_col_types = FALSE) %>% select(-c("Customer ID"
head(ecomm_data)
```

```
## # A tibble: 6 x 7
##     Age `Annual Income (K$)` Gender `Product Category Purchased`
##   <dbl>                <dbl> <chr>  <chr>
## 1    56                  106 Female Fashion
## 2    69                   66 Female Home
## 3    46                  110 Male   Fashion
## 4    32                   50 Male   Electronics
## 5    60                   73 Female Others
## 6    25                   48 Male   Home
## # i 3 more variables: `Average Spend per Visit ($)` <dbl>,
## #   `Number of Visits in Last 6 Months` <dbl>, `Customer Segment` <chr>
```

Let's also get the summary statistics of the dataset:

```r
ecomm_data %>% skim()
```

Table 1: Data summary

Name	Piped data
Number of rows	10532
Number of columns	7
Column type frequency:	
character	3
numeric	4
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Gender	0	1	4	6	0	2	0
Product Category Purchased	0	1	4	11	0	5	0
Customer Segment	0	1	6	7	0	3	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Age	0	1	43.59	14.90	18	31.00	43.00	56.00	69.00	
Annual Income	0	1	104.30	54.61	10	56.71	104.69	150.89	199.96	
(K)|0|1|89.18|34.41|30|59.00|89.00|118.00|149.00|			||$AverageSpendperVisit()$							
Number of Visits in Last 6 Months	0	1	21.92	10.08	5	13.00	22.00	31.00	39.00	

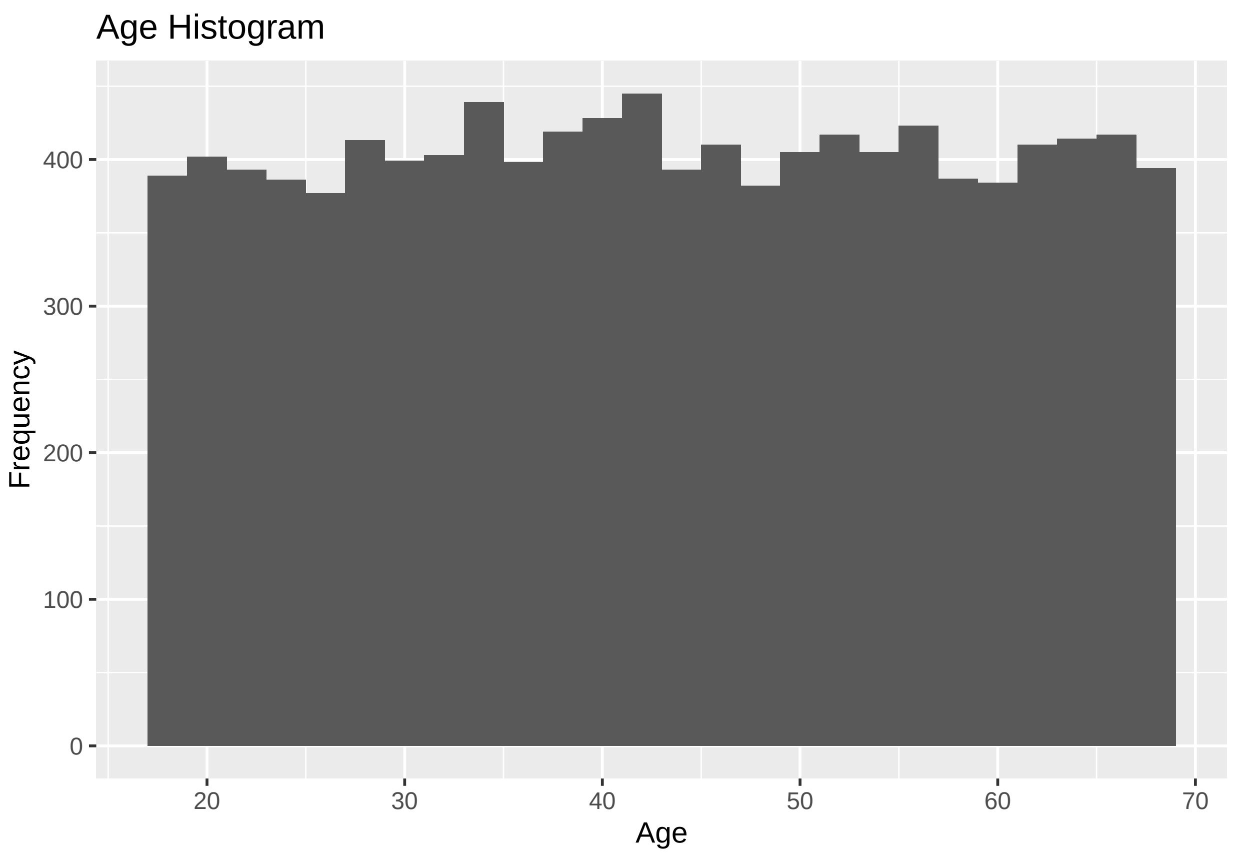
We can see that there are no missing data in our data set. We have also taken the liberty of taking out the Customer ID column, which would not be helpful in our goal.

We can also see the summary statistics for each column, especially the numerical features. For example, we can see that the median age for customers is 43, that the average annual income is $89,183.73, that the average spending per visit is $104.30, while the median number of visits in the last 6 months is 22.

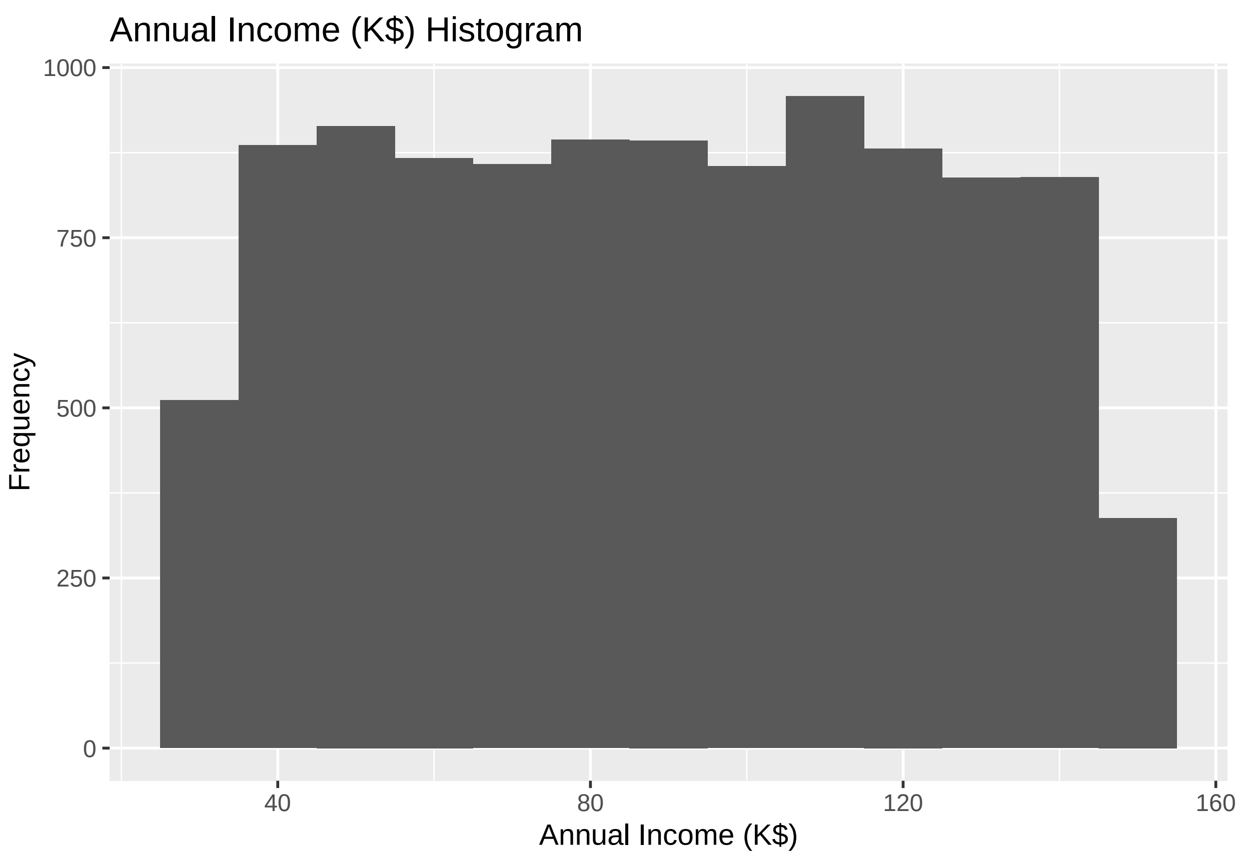
But to be able to better see the distribution of our features, let's create some visualizations.

Let's start with histograms of the numerical variables, namely Age, Annual Income, Average Spend per Visit, and Number of Visits in the Last 6 Months.

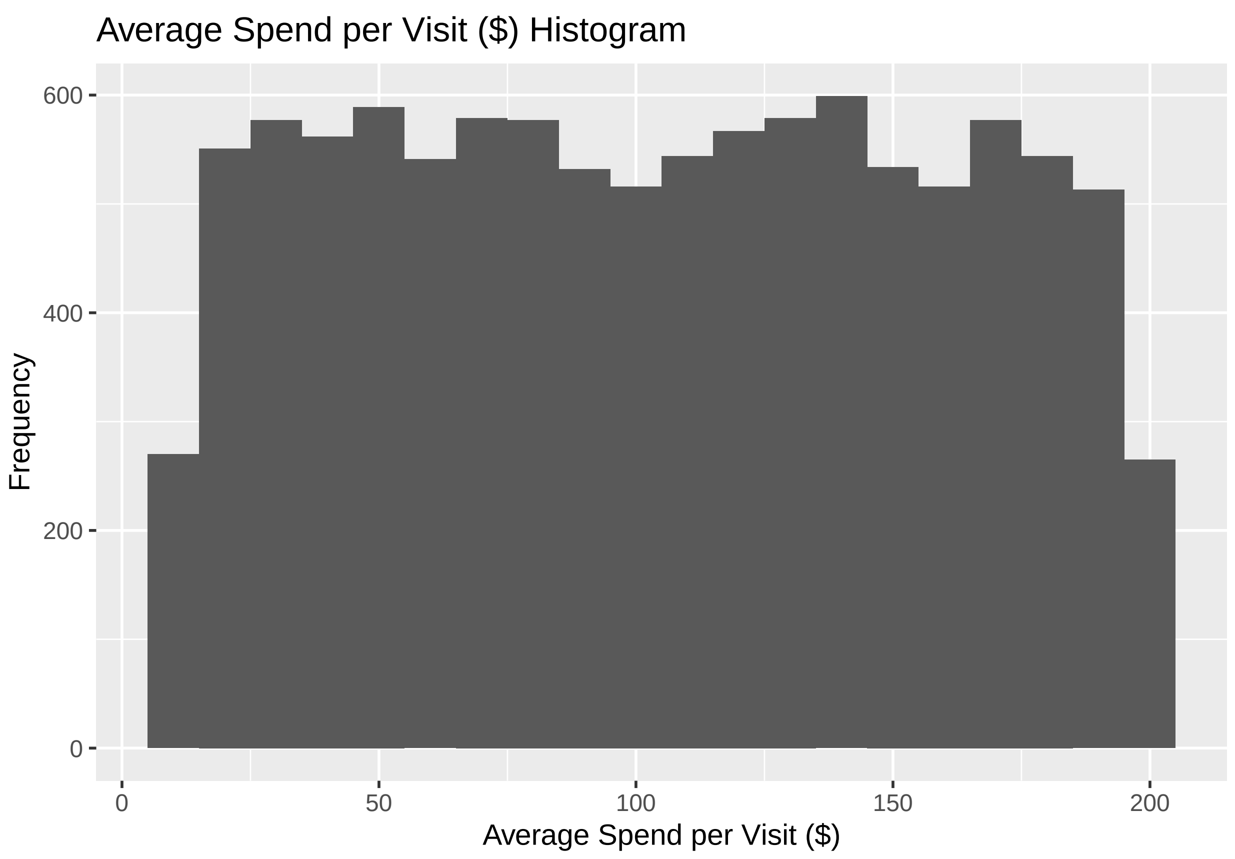
```r
ecomm_data %>% ggplot(aes(x=Age)) +
  geom_histogram(binwidth = 2) +
  ylab("Frequency") +
  xlab("Age")+
  ggtitle("Age Histogram")
```



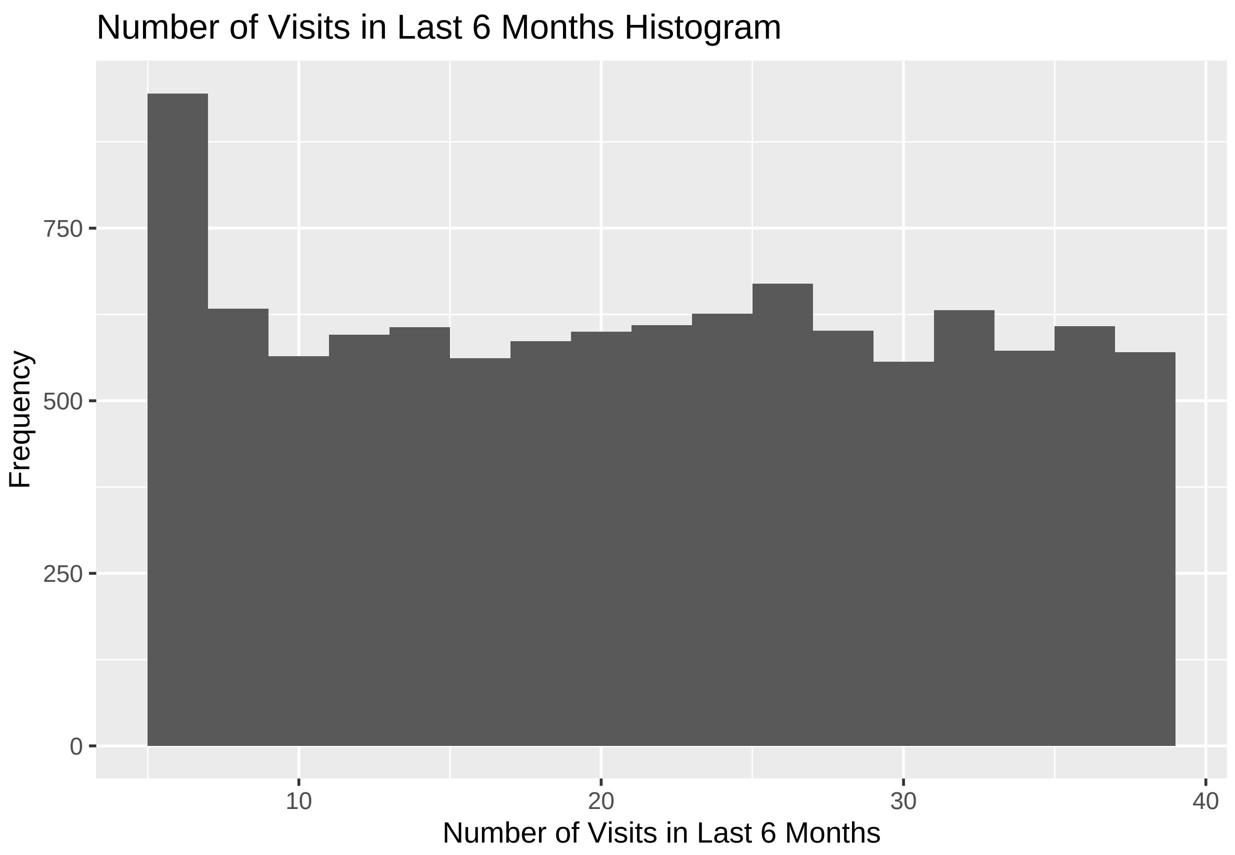
```
ecomm_data %>% ggplot(aes(x=`Annual Income (K$)`)) +
  geom_histogram(binwidth = 10) +
  ylab("Frequency") +
  xlab("Annual Income (K$)")+
  ggtitle("Annual Income (K$) Histogram")
```



```
ecomm_data %>% ggplot(aes(x=`Average Spend per Visit ($)`)) +
  geom_histogram(binwidth = 10) +
  ylab("Frequency") +
  xlab("Average Spend per Visit ($)")+
  ggtitle("Average Spend per Visit ($) Histogram")
```



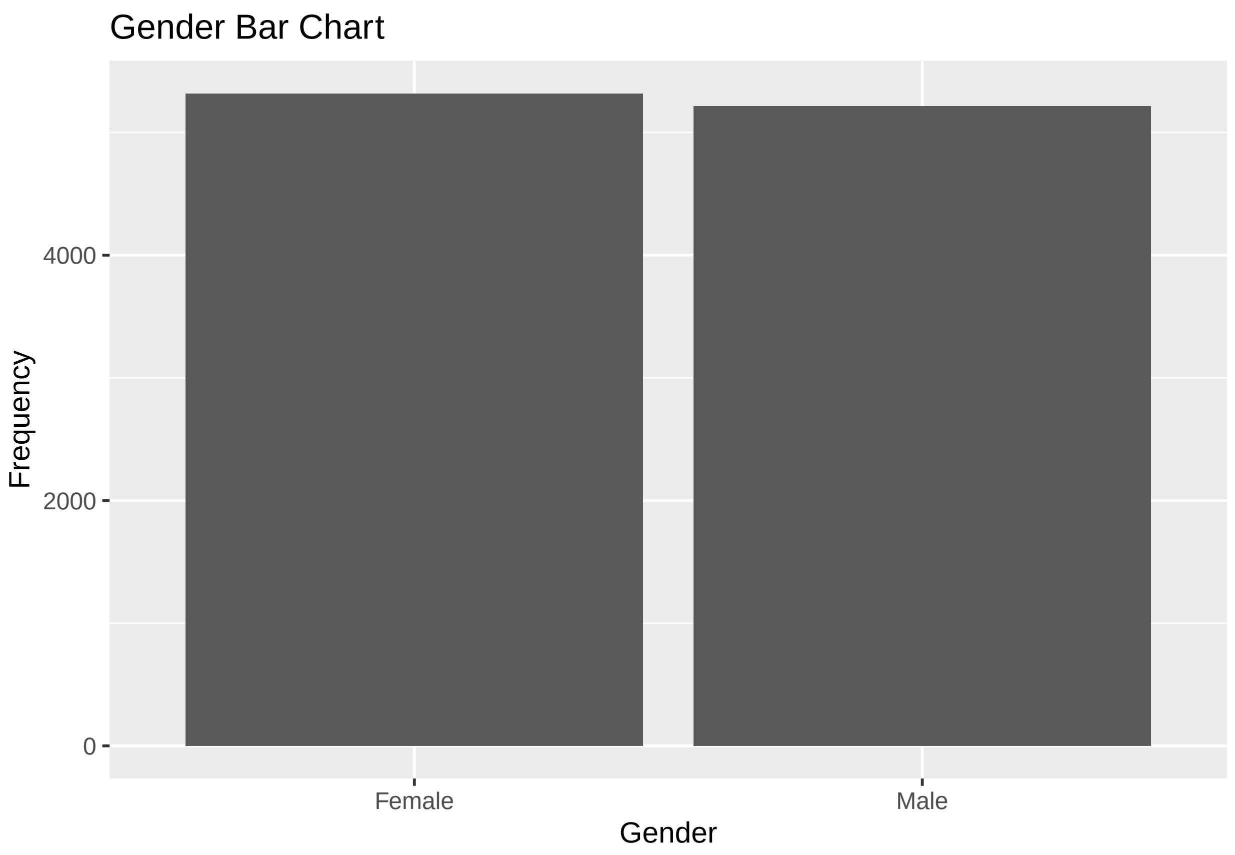
```r
ecomm_data %>% ggplot(aes(x=`Number of Visits in Last 6 Months`)) +
  geom_histogram(binwidth = 2) +
  ylab("Frequency") +
  xlab("Number of Visits in Last 6 Months")+
  ggtitle("Number of Visits in Last 6 Months Histogram")
```



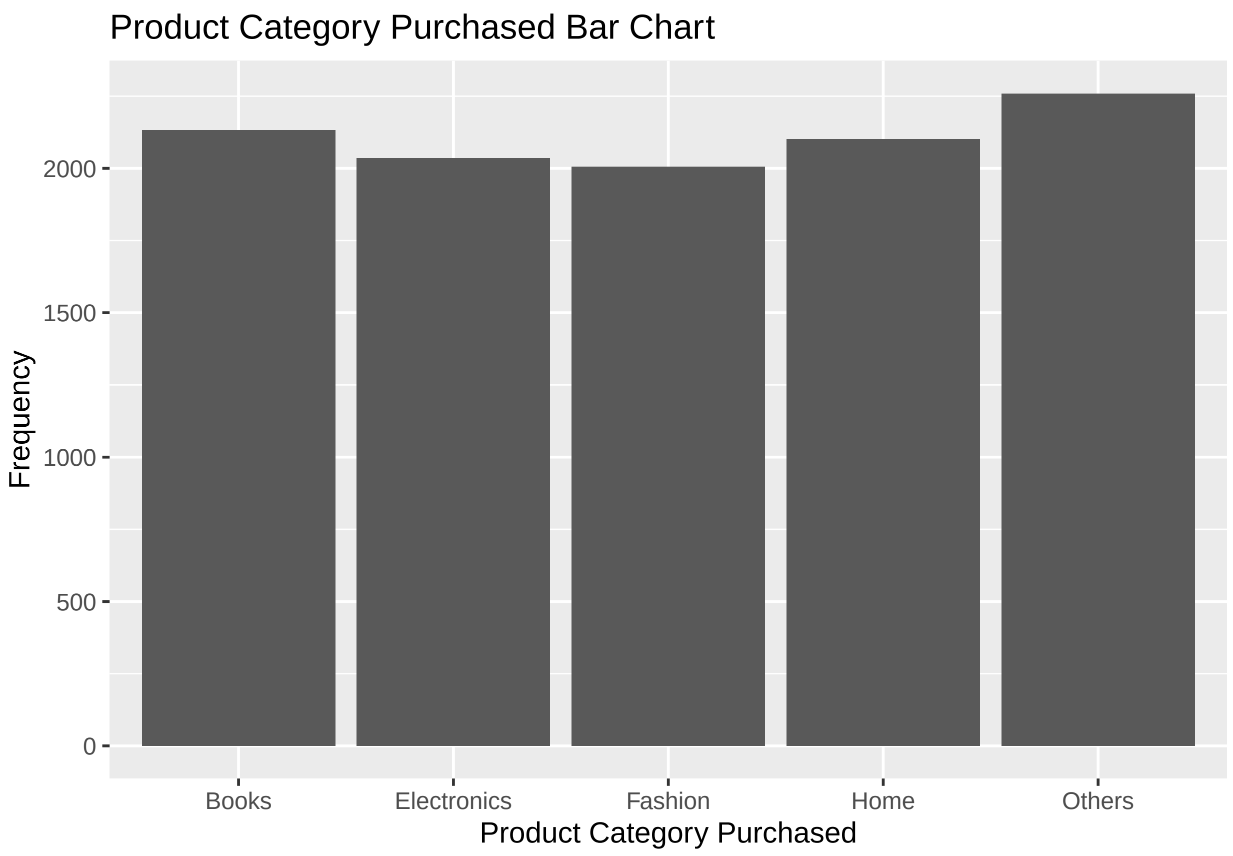
We can see that our numerical variables exhibit a distribution close to a uniform distribution, with some spikes at certain intervals.

Now, let's check the distribution of our categorical variables: Gender, Product Category Purchased, Customer Segment.

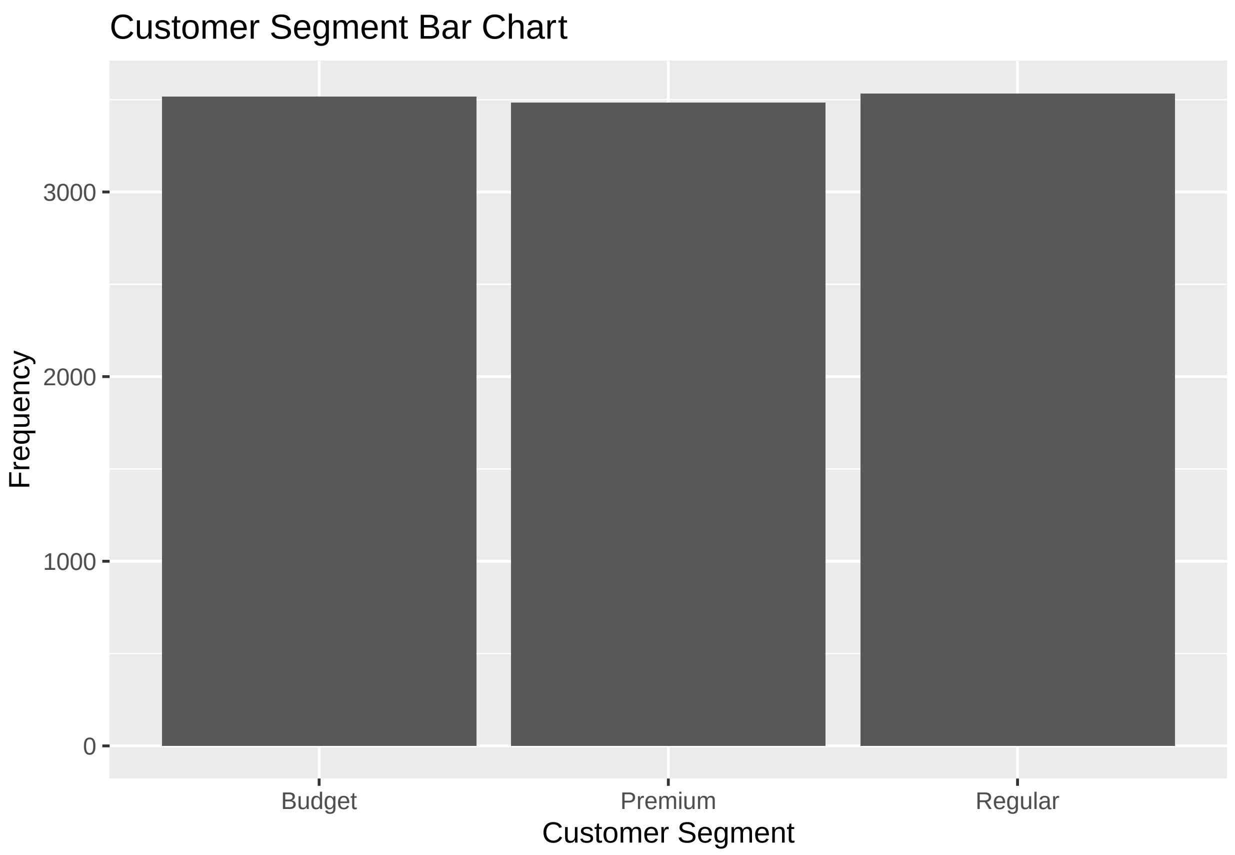
```
ecomm_data %>% ggplot(aes(x=Gender)) +
  geom_bar() +
  ylab("Frequency") +
  xlab("Gender")+
  ggtitle("Gender Bar Chart")
```



```
ecomm_data %>% ggplot(aes(x=`Product Category Purchased`)) +
  geom_bar() +
  ylab("Frequency") +
  xlab("Product Category Purchased")+
  ggtitle("Product Category Purchased Bar Chart")
```



```
ecomm_data %>% ggplot(aes(x=`Customer Segment`)) +
  geom_bar() +
  ylab("Frequency") +
  xlab("Customer Segment")+
  ggtitle("Customer Segment Bar Chart")
```



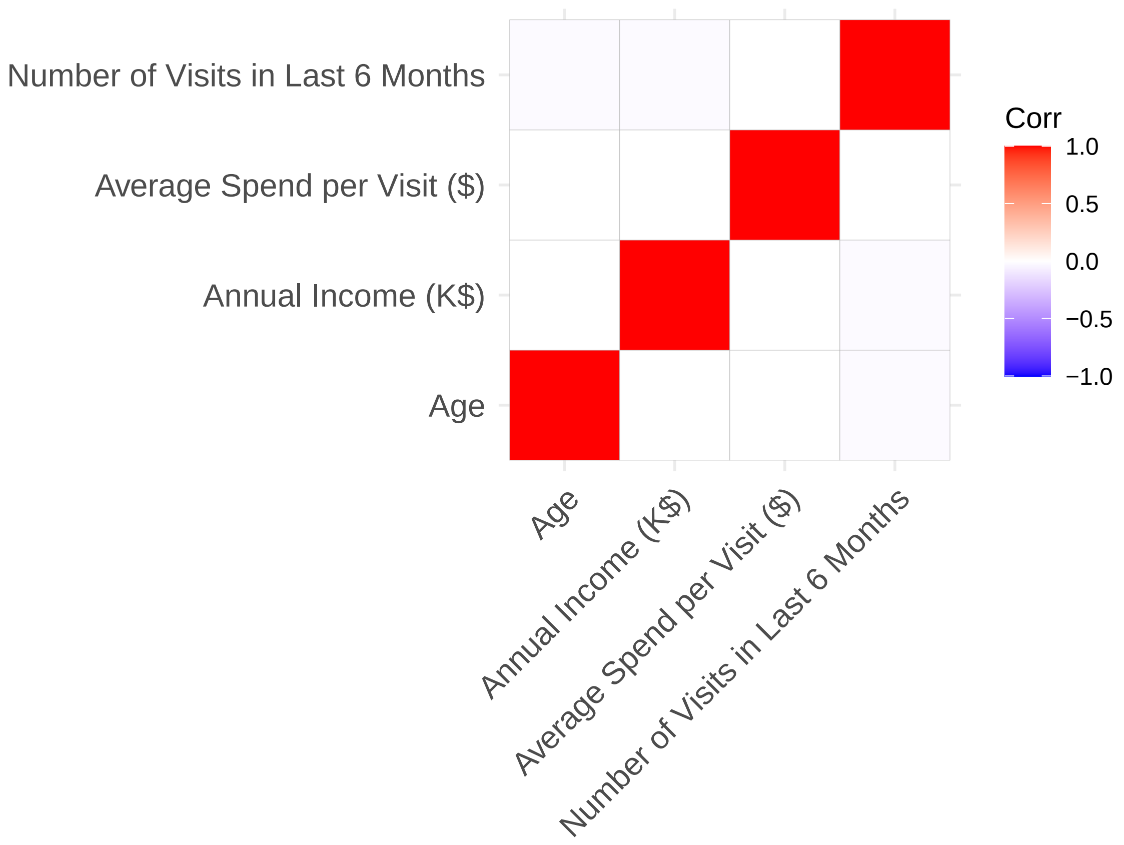
The plots show that there is slightly more female shoppers than male ones, but the difference is not significant.

The sales of each product category is also somewhat uniform across all categories.

For customer segment, the distribution is also uniform.

let's then get the correlation plot of the continuous variables.

```
ecomm_data %>%
  select(c("Age", "Annual Income (K$)", "Average Spend per Visit ($)", "Number of Visits in Last 6 Montl
  cor() %>%
  ggcorrplot()
```



As we can see from the correlation plot, the numerical variables do not actually have any correlation with each other.

Data Preprocessing

Let's now encode our categorical variables.

Let's start by label encoding our gender variable:

```r
ecomm_data <- ecomm_data %>% mutate(Gender = ifelse(Gender == "Male", 0, 1))
ecomm_data %>% head()
```

```
## # A tibble: 6 x 7
##     Age `Annual Income (K$)` Gender `Product Category Purchased`
##   <dbl>                <dbl>  <dbl> <chr>
## 1    56                  106      1 Fashion
## 2    69                   66      1 Home
## 3    46                  110      0 Fashion
## 4    32                   50      0 Electronics
## 5    60                   73      1 Others
## 6    25                   48      0 Home
## # i 3 more variables: `Average Spend per Visit ($)` <dbl>,
## #   `Number of Visits in Last 6 Months` <dbl>, `Customer Segment` <chr>
```

Now, let's one-hot encode for the product category variable, with the "Others" category as reference:

```r
ecomm_data <- ecomm_data %>%
  mutate(
    product.books = ifelse(`Product Category Purchased` == "Books", 1, 0 ),
    product.electronics = ifelse(`Product Category Purchased` == "Electronics", 1, 0 ),
    product.fashion = ifelse(`Product Category Purchased` == "Fashion", 1, 0 ),
    product.home = ifelse(`Product Category Purchased` == "Home", 1, 0 )
  ) %>% select(-c("Product Category Purchased"))
head(ecomm_data)
```

```
## # A tibble: 6 x 10
##     Age `Annual Income (K$)` Gender `Average Spend per Visit ($)`
##   <dbl>                <dbl>  <dbl>                         <dbl>
## 1    56                  106      1                          163.
## 2    69                   66      1                          163.
## 3    46                  110      0                          105.
## 4    32                   50      0                          110.
## 5    60                   73      1                          142.
## 6    25                   48      0                          107.
## # i 6 more variables: `Number of Visits in Last 6 Months` <dbl>,
## #   `Customer Segment` <chr>, product.books <dbl>, product.electronics <dbl>,
## #   product.fashion <dbl>, product.home <dbl>
```

Let's then scale the Age, annual income, average spend per visit, and number of visits using the minmax scaler.

```
ecomm_data_scaled <- ecomm_data %>% mutate(across(
  c("Age", "Annual Income (K$)", "Average Spend per Visit ($)", "Number of Visits in Last 6 Months"),
  ~ (. - min(., na.rm = TRUE))/ (max(., na.rm = TRUE) - min(., na.rm = TRUE))
))

head(ecomm_data_scaled)
```

```
## # A tibble: 6 x 10
##     Age `Annual Income (K$)` Gender `Average Spend per Visit ($)`
##   <dbl>                <dbl>  <dbl>                         <dbl>
## 1 0.745                0.639      1                         0.808
## 2 1                    0.303      1                         0.806
## 3 0.549                0.672      0                         0.498
## 4 0.275                0.168      0                         0.527
## 5 0.824                0.361      1                         0.696
## 6 0.137                0.151      0                         0.509
## # i 6 more variables: `Number of Visits in Last 6 Months` <dbl>,
## #   `Customer Segment` <chr>, product.books <dbl>, product.electronics <dbl>,
## #   product.fashion <dbl>, product.home <dbl>
```

then let's finally split the dataset to be used in the creation of our model.

```
set.seed(12345)

train_index <- createDataPartition(ecomm_data_scaled$`Customer Segment`, p=0.8, list=FALSE)
train_data <- ecomm_data_scaled[train_index, ]
test_data <- ecomm_data_scaled[-train_index, ]
```

Model Building

Now, let's try building a Multinomial Regression model using nnet.

```
ctrl <- trainControl(
  method = "cv",
```

```r
  number = 5,
  classProbs = TRUE,
  summaryFunction = mnLogLoss
)


model <- train(
  `Customer Segment` ~ .,
  data = train_data,
  method = "multinom",
  trControl = ctrl,
  metric = "logLoss",
  trace = FALSE
)

summary(model)
```

```
## Call:
## nnet::multinom(formula = .outcome ~ ., data = dat, decay = param$decay,
##     trace = FALSE)
##
## Coefficients:
##         (Intercept)        Age `\\`Annual Income (K$)\\``       Gender
## Premium   0.1391533 0.02746636               -0.036625997 0.0007965163
## Regular   0.1495056 0.06910094               -0.005081596 0.0603305056
##         `\\`Average Spend per Visit ($)\\``
## Premium                          -0.08654370
## Regular                          -0.09083446
##         `\\`Number of Visits in Last 6 Months\\`` product.books
## Premium                                -0.1328802    -0.1539982
## Regular                                -0.0842832    -0.2512501
##         product.electronics product.fashion product.home
## Premium         -0.03046256      0.03538338  -0.02074711
## Regular         -0.13260368     -0.06296761  -0.15468077
##
## Std. Errors:
##         (Intercept)        Age `\\`Annual Income (K$)\\``     Gender
## Premium   0.1125057 0.09149322                 0.09242450 0.05350905
## Regular   0.1118071 0.09120508                 0.09214282 0.05333950
##         `\\`Average Spend per Visit ($)\\``
## Premium                           0.09332746
## Regular                           0.09305199
##         `\\`Number of Visits in Last 6 Months\\`` product.books
## Premium                                0.09030119    0.08348837
## Regular                                0.09001339    0.08247399
##         product.electronics product.fashion product.home
## Premium          0.08497845      0.08502145   0.08363285
## Regular          0.08406314      0.08406452   0.08304028
##
## Residual Deviance: 18496.88
## AIC: 18536.88
```

Model Evaluation

Let's first create predictions using our model, we can then use this for checking the performance of the model.

```r
predicted_classes <- predict(model, newdata = test_data)
actual <- test_data$`Customer Segment`

predicted_probs <- predict(model, newdata = test_data, type = "prob")
```

Then, using the caret library, we can get the confusion matrix, accuracy, precision, recall, and the F1 score.

```r
conf_matrix <- confusionMatrix(as.factor(predicted_classes), as.factor(actual), mode = "prec_recall")

print(conf_matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Budget Premium Regular
##    Budget     272     256     271
##    Premium    166     151     155
##    Regular    265     289     280
##
## Overall Statistics
##
##                Accuracy : 0.334
##                  95% CI : (0.3138, 0.3546)
##     No Information Rate : 0.3354
##     P-Value [Acc > NIR] : 0.5632
##
##                   Kappa : 3e-04
##
##  Mcnemar's Test P-Value : 6.803e-13
##
## Statistics by Class:
##
##                      Class: Budget Class: Premium Class: Regular
## Precision                   0.3404        0.31992         0.3357
## Recall                      0.3869        0.21695         0.3966
## F1                          0.3622        0.25856         0.3636
## Prevalence                  0.3340        0.33064         0.3354
## Detection Rate              0.1292        0.07173         0.1330
## Detection Prevalence        0.3796        0.22423         0.3962
## Balanced Accuracy           0.5055        0.49457         0.5003
```

We can see that our created model is no better than guessing between three items of assumed same probabilities (which is 33.3333...%). To be specific the accuracy of the model is 0.3344.

In predicting if a customer is a budget shopper, the model precision is at 0.3404, which says that only 34% of the predicted "budget shopper" are actually budget shoppers. The model's recall is then at 0.3883, which says that the model only correctly predicts 39% of the total budget shoppers. An F1 score of 0.3682 highlights the model's unreliability in predicting if a customer is a budget shopper.

In predicting the premium shopper class, the model has a precision of 0.31915, a recall of 0.21552, and an F1 score of 0.25729.

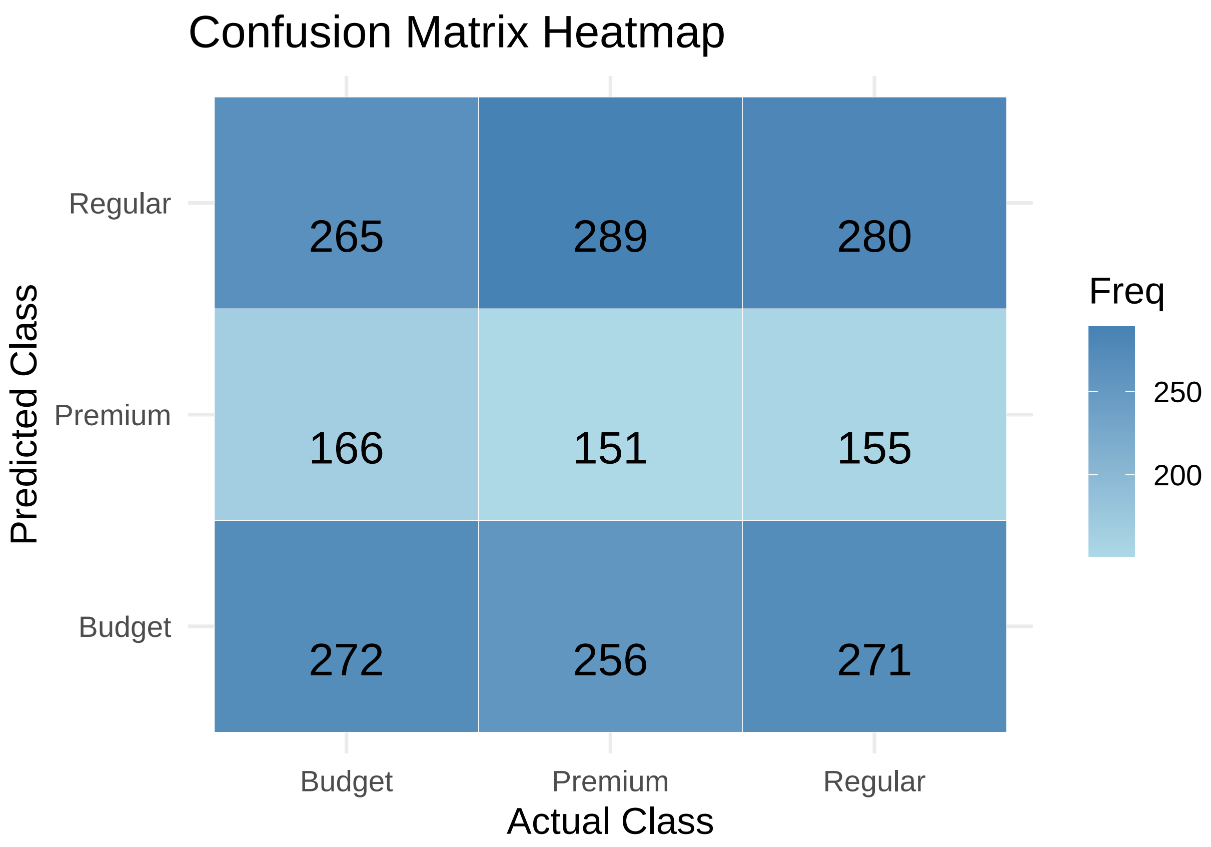
for predicting the Regular Shopper class, the precision is 0.3349, the recall is 0.3952, and an F1 score of 0.3626.

```r
logloss <- model$results$logLoss

cat("The log loss are:",
logloss,
"For Budget, Premium, and Regular shopper predictions, respectively")
```

```
## The log loss are: 1.099567 1.099567 1.099563 For Budget, Premium, and Regular shopper predictions, re
```

The log loss values show that the model isn't doing a good job predicting the class.

```r
cm_df <- as.data.frame(conf_matrix$table)
names(cm_df) <- c("Predicted", "Actual", "Freq")

ggplot(data = cm_df, aes(x = Actual, y = Predicted, fill = Freq)) +
  geom_tile(color = "white") +
  geom_text(aes(label = Freq), vjust = 1.5, size = 6) +
  scale_fill_gradient(low = "lightblue", high = "steelblue") +
  labs(
    title = "Confusion Matrix Heatmap",
    x = "Actual Class",
    y = "Predicted Class"
  ) +
  theme_minimal(base_size = 14)
```



We can see from the confusion matrix heatmap that the model prefers to classify the customers into budget and regular spenders, while shying away from classifying shoppers as premium spenders.

Refinement

We can try to improve our model. One such way is to create interaction variables.

```r
ecomm_interaction_scaled <- ecomm_data %>% mutate(
  age_x_annualIncome = Age * `Annual Income (K$)`,
  numberVisits_x_AveSpend = `Average Spend per Visit ($)` * `Number of Visits in Last 6 Months`,
  aveSpend_div_annualIncome = `Average Spend per Visit ($)`/ `Annual Income (K$)` ) %>%
  select(-c("Age", "Annual Income (K$)", "Average Spend per Visit ($)", "Number of Visits in Last 6 Mon
  mutate(across(
  c("age_x_annualIncome", "numberVisits_x_AveSpend", "aveSpend_div_annualIncome"),
  ~ (. - min(., na.rm = TRUE))/ (max(., na.rm = TRUE) - min(., na.rm = TRUE))
))

train_index <- createDataPartition(ecomm_interaction_scaled$`Customer Segment`, p=0.8, list=FALSE)
train_data <- ecomm_interaction_scaled[train_index, ]
test_data <- ecomm_interaction_scaled[-train_index, ]


ctrl <- trainControl(
  method = "cv",
  number = 5,
  classProbs = TRUE,
  summaryFunction = mnLogLoss
)


model <- train(
  `Customer Segment` ~ .,
  data = train_data,
  method = "multinom",
  trControl = ctrl,
  metric = "logLoss",
  trace = FALSE
)

summary(model)
```

```
## Call:
## nnet::multinom(formula = .outcome ~ ., data = dat, decay = param$decay,
##     trace = FALSE)
##
## Coefficients:
##         (Intercept)     Gender product.books product.electronics
## Premium  0.06679455 0.07968468   -0.09920229        -0.003917205
## Regular  0.13692160 0.05136228   -0.16050578        -0.052297547
##         product.fashion product.home age_x_annualIncome numberVisits_x_AveSpend
## Premium      0.11474802   0.02741301         -0.1790350              -0.2295252
## Regular      0.02737207  -0.06593999         -0.1226544              -0.1272079
##         aveSpend_div_annualIncome
## Premium                0.03023726
## Regular               -0.13024266
##
## Std. Errors:
##         (Intercept)     Gender product.books product.electronics
## Premium  0.09698011 0.05353499    0.08274341          0.08388448
## Regular  0.09614363 0.05332711    0.08191162          0.08290121
##         product.fashion product.home age_x_annualIncome numberVisits_x_AveSpend
## Premium      0.08504583   0.08311158          0.1438901               0.1462706
## Regular      0.08451864   0.08263539          0.1428491               0.1455104
##         aveSpend_div_annualIncome
## Premium                 0.2124542
## Regular                 0.2137316
##
## Residual Deviance: 18497.71
## AIC: 18533.71
```

```r
print("\n\n")
```

```
## [1] "\n\n"
```

```r
predicted_classes <- predict(model, newdata = test_data)
actual <- test_data$`Customer Segment`
predicted_probs <- predict(model, newdata = test_data, type = "prob")

conf_matrix <- confusionMatrix(as.factor(predicted_classes), as.factor(actual), mode = "prec_recall")

print(conf_matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Budget Premium Regular
##    Budget     282     260     269
##    Premium    218     208     213
##    Regular    203     228     224
##
## Overall Statistics
##
##                Accuracy : 0.3392
##                  95% CI : (0.319, 0.3599)
##     No Information Rate : 0.3354
##     P-Value [Acc > NIR] : 0.363749
##
##                   Kappa : 0.0087
##
##  Mcnemar's Test P-Value : 0.003794
##
## Statistics by Class:
##
##                      Class: Budget Class: Premium Class: Regular
## Precision                   0.3477        0.32551         0.3420
## Recall                      0.4011        0.29885         0.3173
## F1                          0.3725        0.31161         0.3292
## Prevalence                  0.3340        0.33064         0.3354
## Detection Rate              0.1340        0.09881         0.1064
## Detection Prevalence        0.3853        0.30356         0.3112
## Balanced Accuracy           0.5119        0.49648         0.5046
```

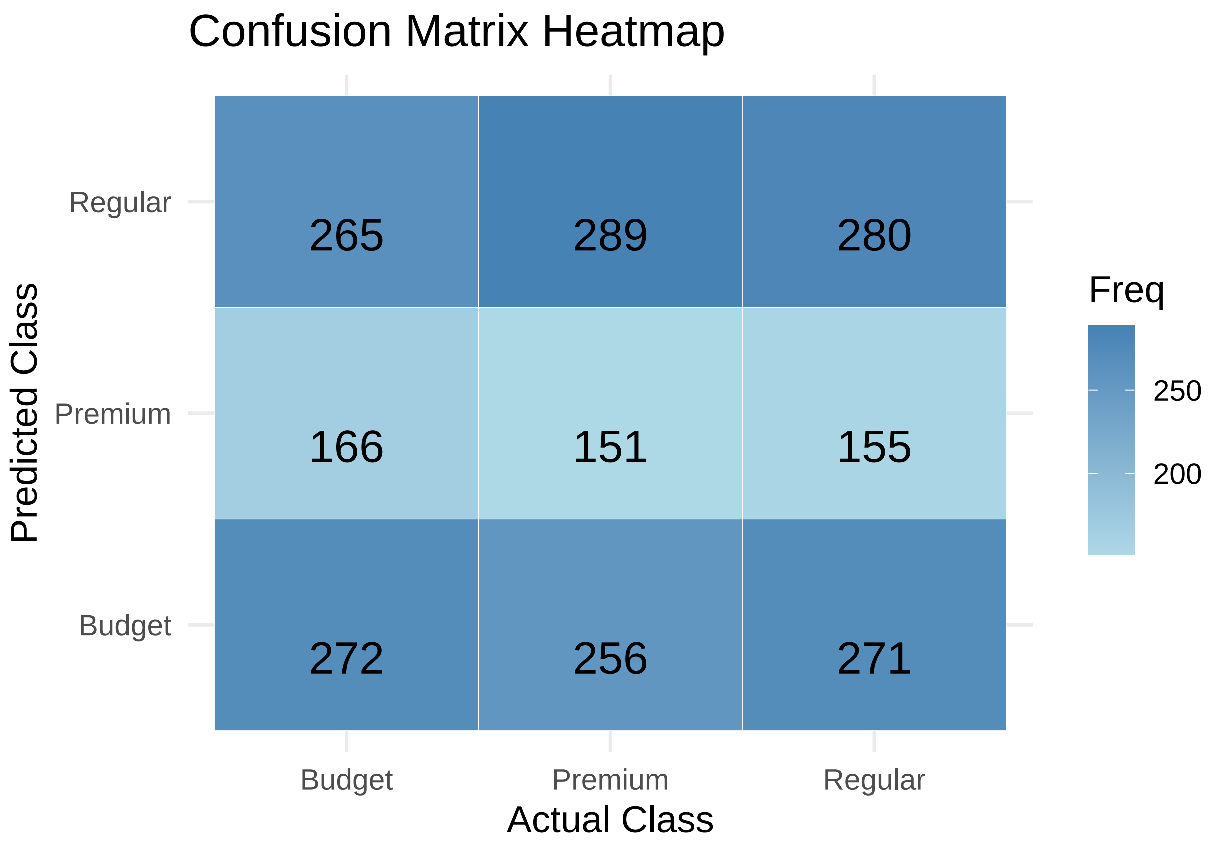
```r
print("\n\n")
```

```
## [1] "\n\n"
```

```r
logloss <- model$results$logLoss

cat("The log loss are:",
logloss,
"For Budget, Premium, and Regular shopper predictions, respectively")
```

```
## The log loss are: 1.099902 1.099902 1.099894 For Budget, Premium, and Regular shopper predictions, r
```

```r
m_df <- as.data.frame(conf_matrix$table)
names(cm_df) <- c("Predicted", "Actual", "Freq")

ggplot(data = cm_df, aes(x = Actual, y = Predicted, fill = Freq)) +
  geom_tile(color = "white") +
  geom_text(aes(label = Freq), vjust = 1.5, size = 6) +
  scale_fill_gradient(low = "lightblue", high = "steelblue") +
  labs(
    title = "Confusion Matrix Heatmap",
    x = "Actual Class",
    y = "Predicted Class"
  ) +
  theme_minimal(base_size = 14)
```



We created the following variables: age times annual income, number of visits times average spending per visit, and the average spending per visit divided by annual income. We then removed the original columns to avoid co linearity.

The result, particularly the accuracy, precision, recall, and F1-score, shows that there is very little improvement, if any. The log loss statistics prove that further.

Reporting

Our exploratory data analysis showed that the distribution of the variables are uniformly distributed. It is perhaps due to this that our multinomial logistic regression model is not able to create any meaningful performance: Its accuracy, precision, recall, and F1-score, is not far from 0.333..., a performance that is no better than guessing between three options given equal probabilities. The log loss shows no promise either, with scores close to 1.