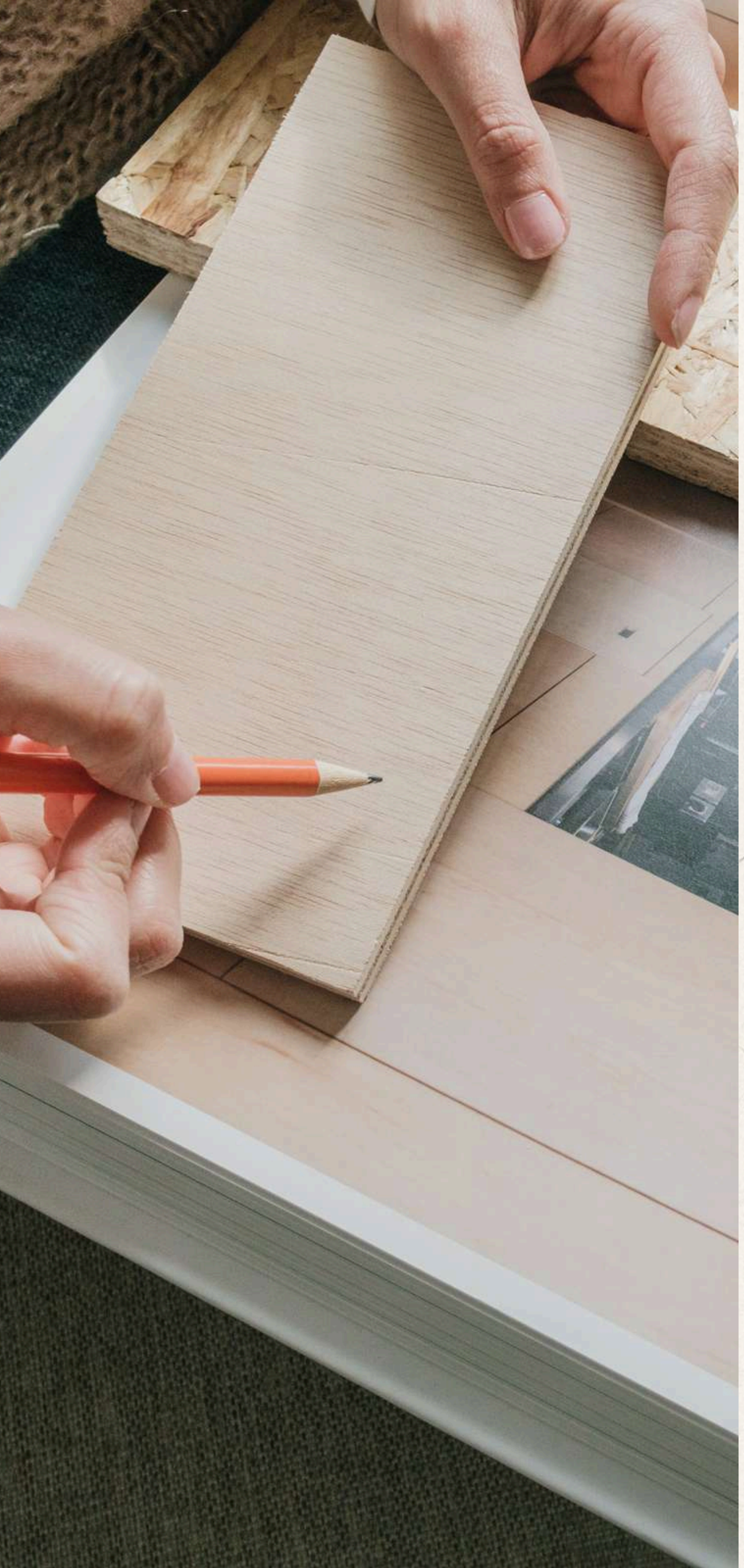
Noah Alexander Lie

ONLINE FOOD ORDERING PLATFORM CUSTOMER ANALYSIS AND PREDICTION MODELLING

BY NOAH ALEXANDER LIE



Importance of this Project

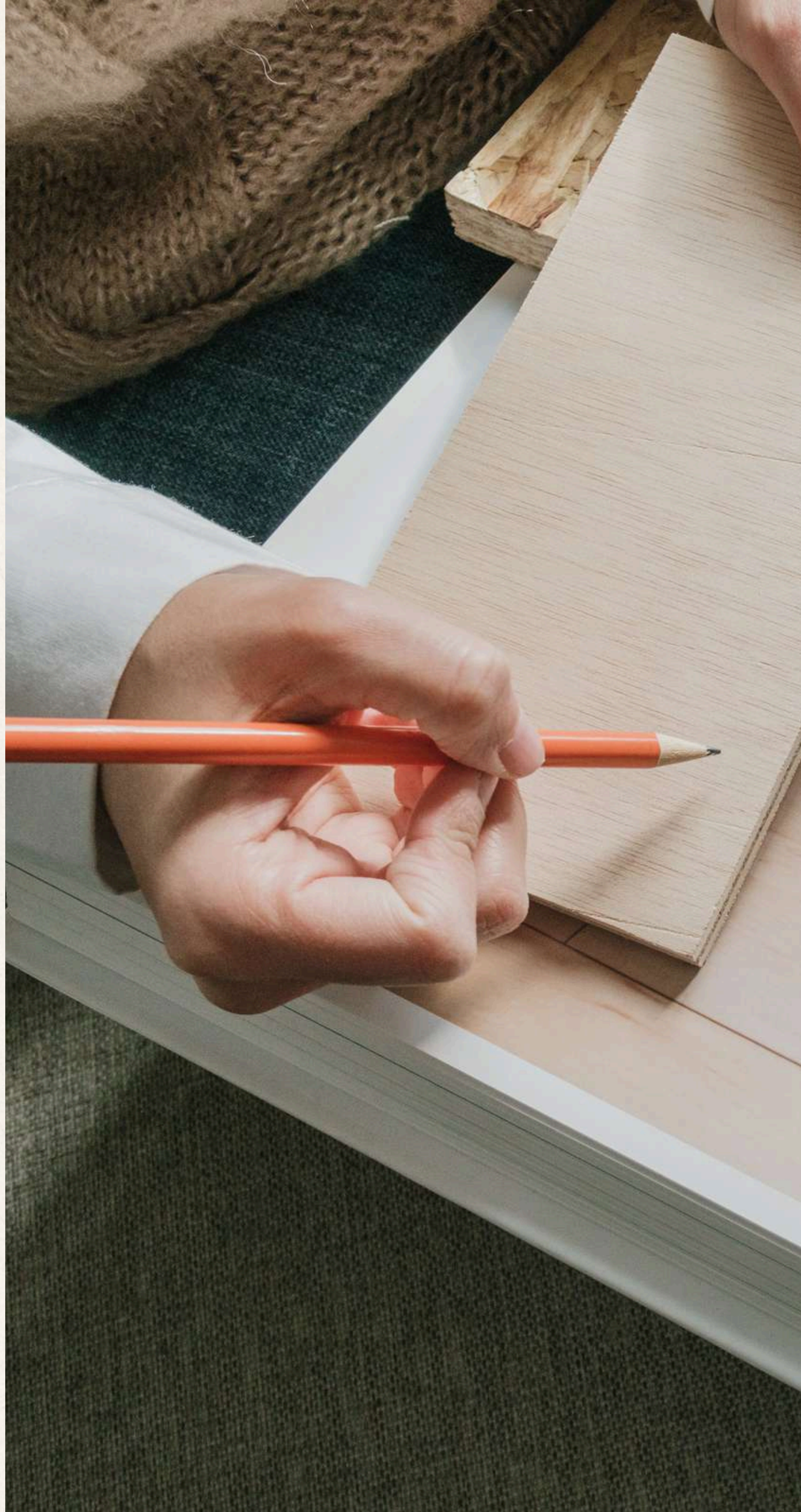
By analyzing customer data, food companies can:

- **Increasing Customer Satisfaction Through Personalized Services**
- **Improve service by addressing customer feedback proactively**

QUESTIONS TO ANSWER

01 What are the demographics of customers that the food platform caters to?

02 How can the demographic of a customer and their feedback help the platform determine whether the customer is likely to reorder?



OVERVIEW OF THE PROJECT

01 Demographic Analysis: Exploring demographic data of customers

02 Customer Feedback Analysis: which demographic liked/disliked the service, etc

03 Predictive Modelling: Predicting whether a customer would reorder based on demographic data and order details

IMPORTING DATA, ORGANIZING DATA, AND INSTALLING PACKAGES

```r
# RUN install.packages(c("ggplot2", "ggthemes", "tidyverse", "gridExtra", "corrplot", "randomForest", "caret", "MLmetrics"))
library(ggplot2)
library(ggthemes)
library(tidyverse)
library(readr)
library(dplyr)
library(gridExtra)
library(corrplot)
library(randomForest)
library(caret)
library(MLmetrics)
```

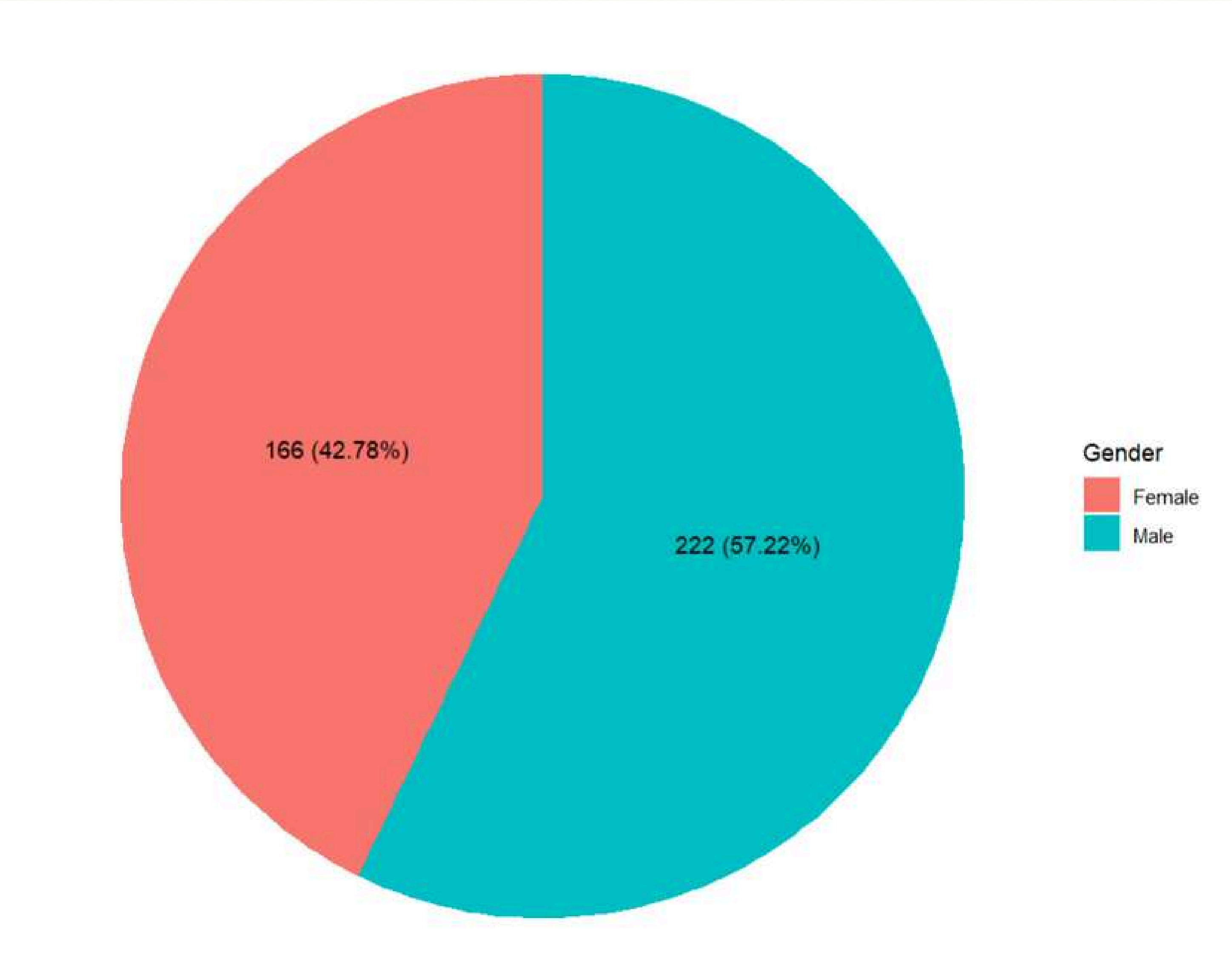
```r
data <- read_csv("onlinefoods.csv") # load the dataset


data <- select(data, -...13) # removes a column that the original creator of the dataset says is an error
data <- rename(data, `Ordered Again` = Output) # creator of dataset mistaken the name of this column
data <- data |> mutate(`Monthly Income` = factor(`Monthly Income`, levels = c("No Income", "Below Rs.10000", "10001 to 25000", "25001 to 50000", "More than 50000")))
data <- data |> mutate(`Family size` = factor(`Family size`, levels = c(1, 2, 3, 4, 5, 6), ordered=TRUE))
data <- data |> mutate(`Educational Qualifications` = factor(`Educational Qualifications`, levels = c("Uneducated", "School", "Graduate", "Post Graduate", "Ph.D"), ordered=TRUE))
head(data, 10)
str(data) #displays all variables in the data, along with their class
```

```
> head(data, 10)
# A tibble: 10 × 12
     Age Gender `Marital Status` Occupation `Monthly Income` `Educational Qualifications` `Family size` latitude longitude `Pin code` `Ordered Again` Feedback
   <dbl> <chr>  <chr>            <chr>      <fct>            <ord>                        <ord>            <dbl>     <dbl>      <dbl> <chr>           <chr>
 1    20 Female Single           Student    No Income        Post Graduate                4                 13.0      77.6     560001 Yes             Positive
 2    24 Female Single           Student    Below Rs.10000   Graduate                     3                 13.0      77.6     560009 Yes             Positive
 3    22 Male   Single           Student    Below Rs.10000   Post Graduate                3                 13.0      77.7     560017 Yes             Negative
 4    22 Female Single           Student    No Income        Graduate                     6                 12.9      77.6     560019 Yes             Positive
 5    22 Male   Single           Student    Below Rs.10000   Post Graduate                4                 13.0      77.6     560010 Yes             Positive
 6    27 Female Married          Employee   More than 50000  Post Graduate                2                 12.9      77.7     560103 Yes             Positive
 7    22 Male   Single           Student    No Income        Graduate                     3                 13.0      77.6     560009 Yes             Positive
 8    24 Female Single           Student    No Income        Post Graduate                3                 13.0      77.6     560042 Yes             Positive
 9    23 Female Single           Student    No Income        Post Graduate                2                 13.0      77.6     560001 Yes             Positive
10    23 Female Single           Student    No Income        Post Graduate                4                 13.0      77.7     560048 Yes             Positive
> str(data) #displays all variables in the data, along with their class
tibble [388 × 12] (S3: tbl_df/tbl/data.frame)
 $ Age                       : num [1:388] 20 24 22 22 22 27 22 24 23 23 ...
 $ Gender                    : chr [1:388] "Female" "Female" "Male" "Female" ...
 $ Marital Status            : chr [1:388] "Single" "Single" "Single" "Single" ...
 $ Occupation                : chr [1:388] "Student" "Student" "Student" "Student" ...
 $ Monthly Income            : Factor w/ 5 levels "No Income","Below Rs.10000",..: 1 2 2 1 2 5 1 1 1 1 ...
 $ Educational Qualifications: Ord.factor w/ 5 levels "Uneducated"<"School"<..: 4 3 4 3 4 4 3 4 4 4 ...
 $ Family size               : Ord.factor w/ 6 levels "1"<"2"<"3"<"4"<..: 4 3 3 6 4 2 3 3 2 4 ...
 $ latitude                  : num [1:388] 13 13 13 12.9 13 ...
 $ longitude                 : num [1:388] 77.6 77.6 77.7 77.6 77.6 ...
 $ Pin code                  : num [1:388] 560001 560009 560017 560019 560010 ...
 $ Ordered Again             : chr [1:388] "Yes" "Yes" "Yes" "Yes" ...
 $ Feedback                  : chr [1:388] "Positive" "Positive" "Negative" "Positive" ...
>
```

DEMOGRAPHIC ANALYSIS

Pie Chart of Male Vs Female

```r
# pie chart showcasing the gender demographic among the customers
gender_count <- data |> group_by(Gender) |> summarize(count = n())
gender_count <- mutate(gender_count, percentage = round(count / sum(count) * 100, 2))
pie_chart <- gender_count |> ggplot(aes(x="", y = count, fill = Gender)) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar(theta = "y") +
  theme_void() +
  labs(title = "Pie Chart with ggplot2", fill = "Gender") +
  geom_text(aes(label = paste(count, " (", percentage, "%)", sep="")), position = position_stack(vjust = 0.5))
pie_chart
```

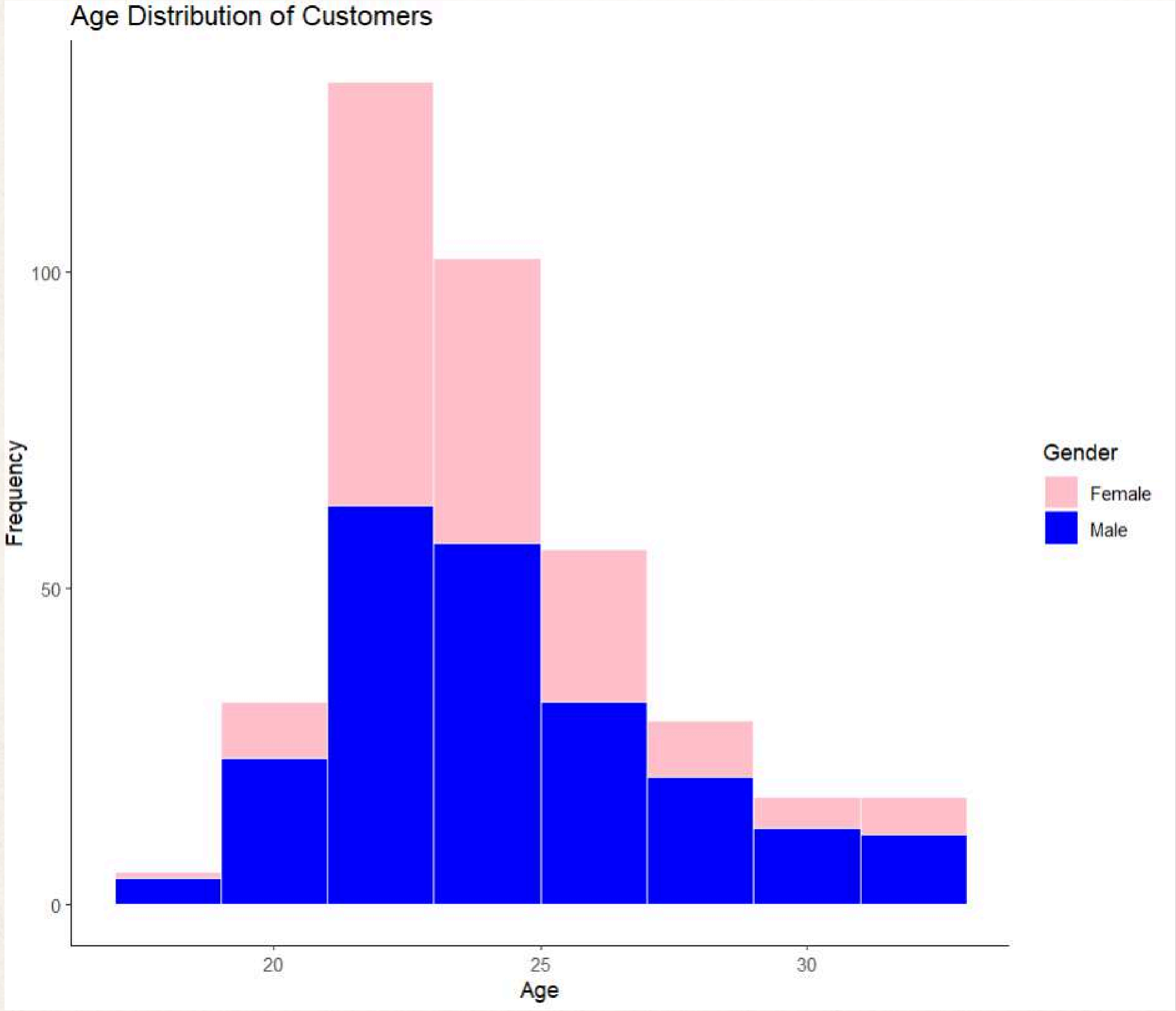


DEMOGRAPHIC ANALYSIS

Histogram of Age Distribution

```r
# Age Count Histogram
age_histogram <- data |> ggplot(aes(x = Age, fill = Gender)) +
  geom_histogram(position = "stack", binwidth = 2, color = "white") +
  labs(x = "Age", y = "Frequency", title = "Age Distribution of Customers") +
  scale_fill_manual(values = c("Male" = "blue", "Female" = "pink")) +
  theme_classic()
age_histogram # Instead of stack, we can overlap male, female, and all genders
```

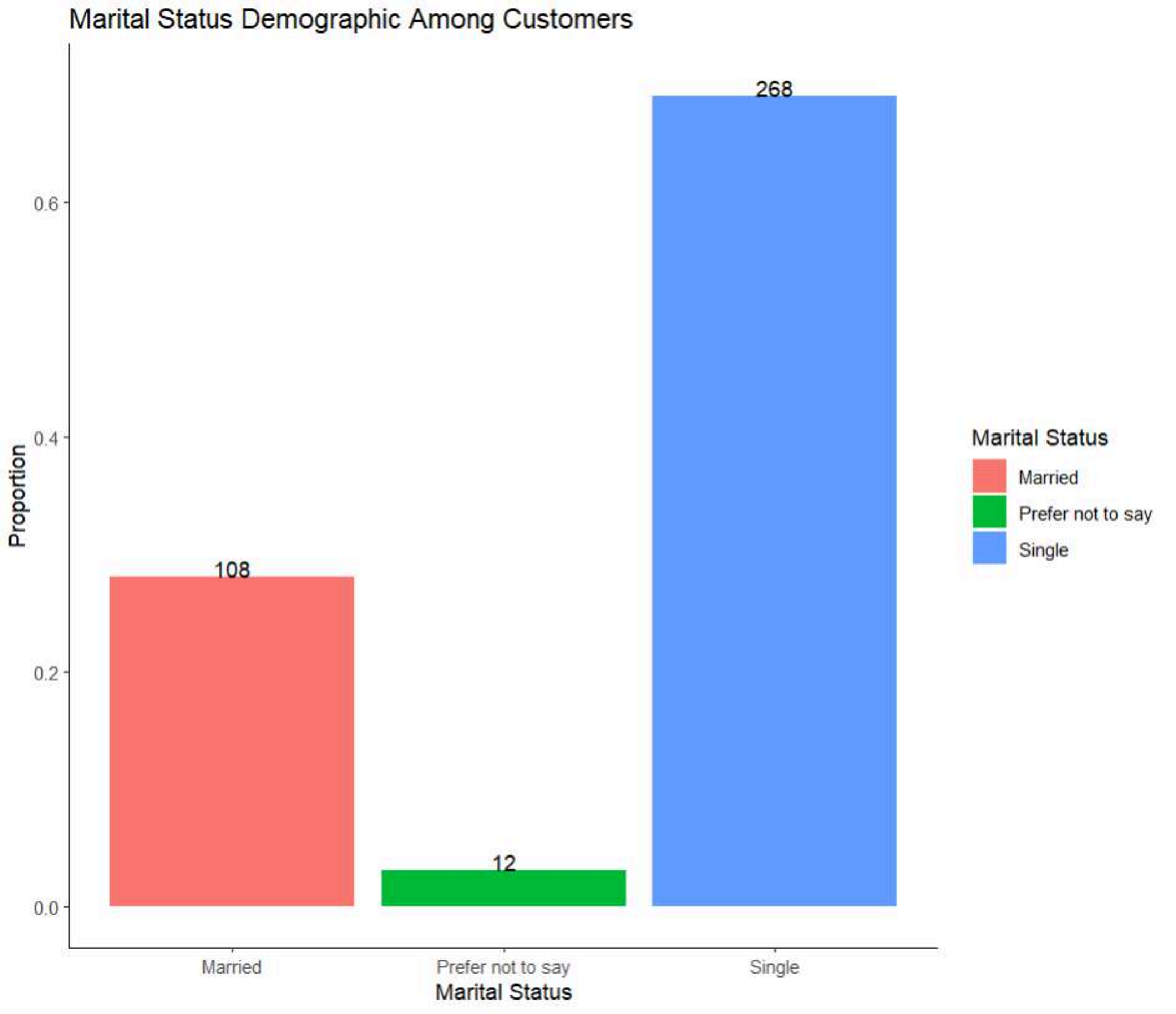
Mostly used by males among ages below 21 & ages above 25. For ages between 21 to 25, gender proportions are similar



DEMOGRAPHIC ANALYSIS

Count Graph of Marital Status

```
marital_status_numbers <- data |> group_by(`Marital Status`) |>
  summarize(count = n()) |>
  mutate(proportion = round(count / sum(count), 2)) |> arrange(desc(proportion))
bar_graph_1 <- marital_status_numbers |>
  ggplot(aes(x = `Marital Status`, y = proportion, fill = `Marital Status`)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = count), vjust = 0, color = "black") + scale_y_continuous(limit = c(0, 0.7)) +
  labs(x = "Marital Status", y = "Proportion", title = "Marital Status Demographic Among Customers") +
  theme_classic()

bar_graph_1
```



DEMOGRAPHIC ANALYSIS

Count Graph of Occupation, Monthly Income, Educational Qualifications

scales made the same to make them easier for comparison

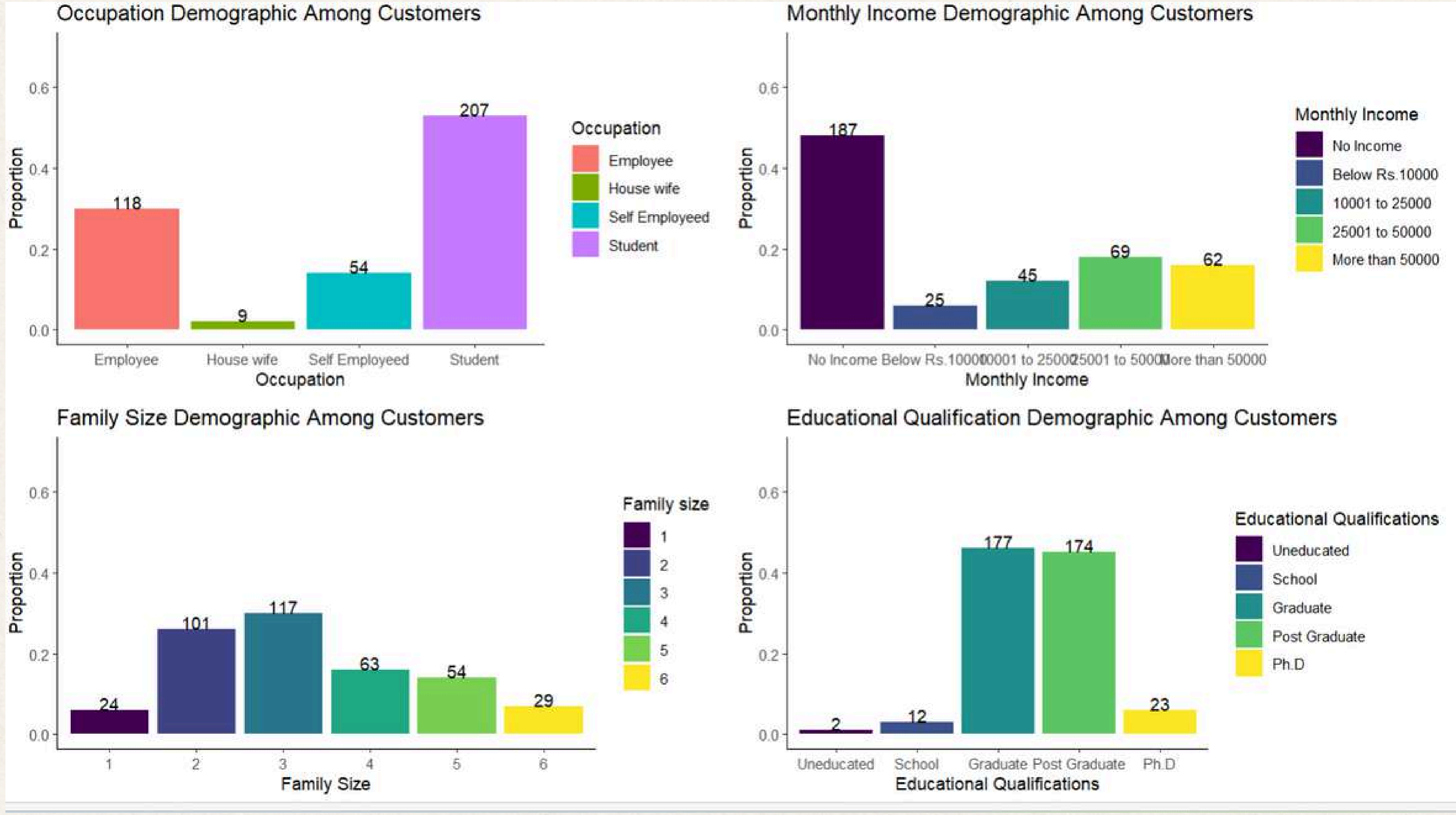
```r
grid.arrange(bar_graph_2, bar_graph_3, bar_graph_4, bar_graph_5, ncol = 2)
```

```r
# Occupation Demographic in Occupation
occupation_numbers <- data |> group_by(Occupation) |> summarize(count = n()) |> mutate(proportion = round(count / sum(count), 2)) |> arrange(desc(proportion))
bar_graph_2 <- occupation_numbers |>
  ggplot(aes(x = Occupation, y = proportion, fill = Occupation)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = count), vjust = 0, color = "black") + scale_y_continuous(limit = c(0, 0.7)) +
  labs(x = "Occupation", y = "Proportion", title = "Occupation Demographic Among Customers") +
  theme_classic()

bar_graph_2
```

```r
# Monthly Income Demographic in Monthly Income
income_numbers <- data |> group_by(`Monthly Income`) |> summarize(count = n()) |> mutate(proportion = round(count / sum(count), 2))
bar_graph_3 <- income_numbers |>
  ggplot(aes(x = `Monthly Income`, y = proportion, fill = `Monthly Income`)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = count), vjust = 0, color = "black") + scale_y_continuous(limit = c(0, 0.7)) +
  labs(x = "Monthly Income", y = "Proportion", title = "Monthly Income Demographic Among Customers") +
  theme_classic()

bar_graph_3
```

```r
# Family Size Demographic in Family Size
family_size_numbers <- data |> group_by(`Family size`) |> summarize(count = n()) |> mutate(proportion = round(count / sum(count), 2))
bar_graph_4 <- family_size_numbers |>
  ggplot(aes(x = `Family size`, y = proportion, fill = `Family size`)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = count), vjust = 0, color = "black") + scale_y_continuous(limit = c(0, 0.7)) +
  labs(x = "Family Size", y = "Proportion", title = "Family Size Demographic Among Customers") +
  theme_classic()

bar_graph_4
```

```r
# Educational Qualifications Demographic in Educational Qualifications
educational_qualifications_numbers <- data |> group_by(`Educational Qualifications`) |> summarize(count = n()) |> mutate(proportion = round(count / sum(count), 2))
bar_graph_5 <- educational_qualifications_numbers |>
  ggplot(aes(x = `Educational Qualifications`, y = proportion, fill = `Educational Qualifications`)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = count), vjust = 0, color = "black") + scale_y_continuous(limit = c(0, 0.7)) +
  labs(x = "Educational Qualifications", y = "Proportion", title = "Educational Qualification Demographic Among Customers") +
  theme_classic()

bar_graph_5
```



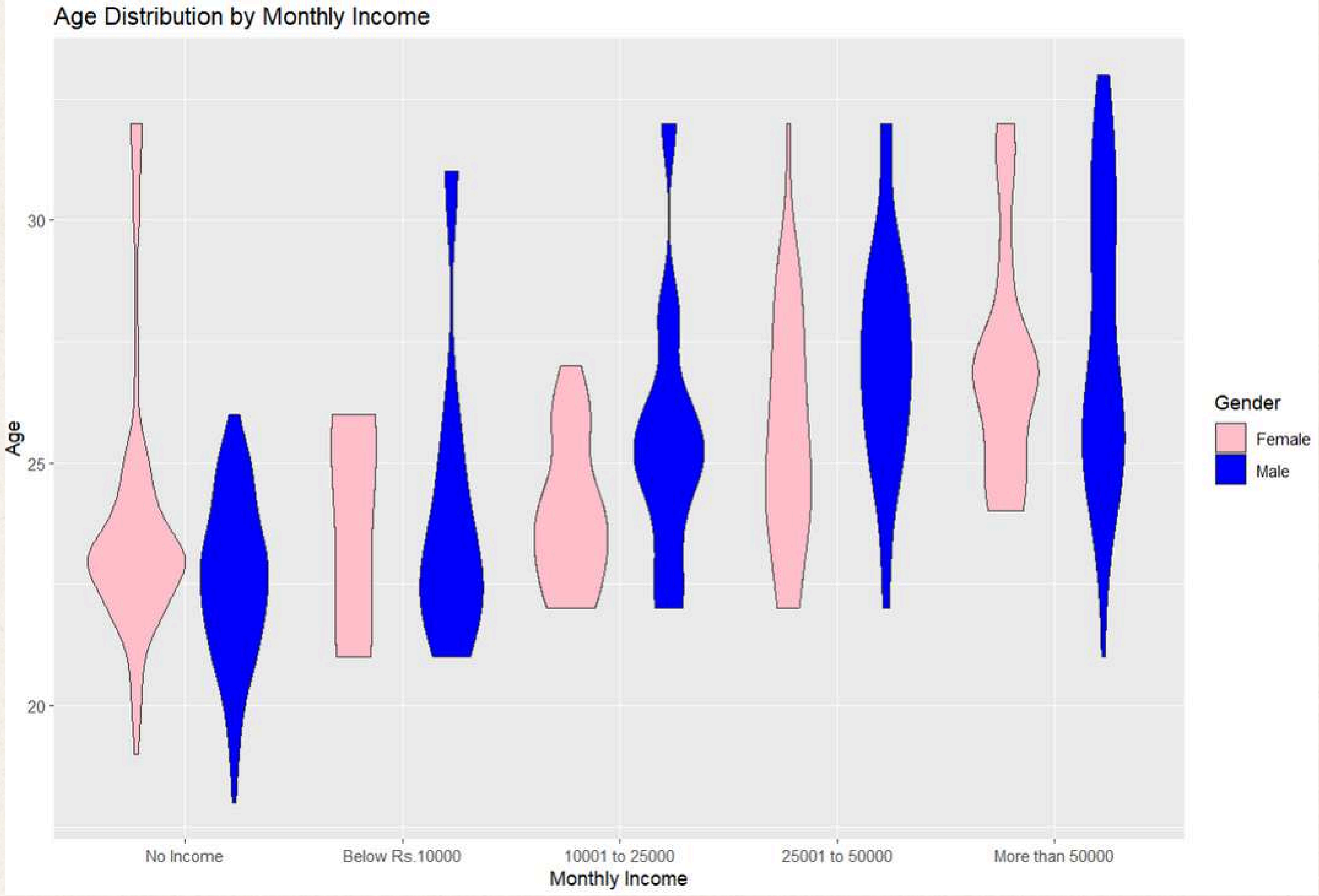
DEMOGRAPHIC ANALYSIS

Conclusions from Count Graph

- Most of the customers are Single
- The service is popular among young adults, particularly students, who may prioritize convenience.
- The customer base is diverse in terms of monthly income
- The service seems to appeal more to individuals and small families, potentially due to lifestyle and convenience factors.
- The educational level of the customer base skews higher, with a significant representation of graduates and postgraduates.

DEMOGRAPHIC ANALYSIS

```
#Violin Plot of Age Vs Monthly Income
violin_plot <- data |> group_by(Gender) |> ggplot(aes(x = `Monthly Income`, y = Age, fill = Gender)) +
  geom_violin() +
  labs(x = "Monthly Income", y = "Age", title = "Age Distribution by Monthly Income") +
  scale_fill_manual(values = c("Male" = "blue", "Female" = "pink"))
violin_plot
```



Violin Plot of Age Distribution by Monthly Income

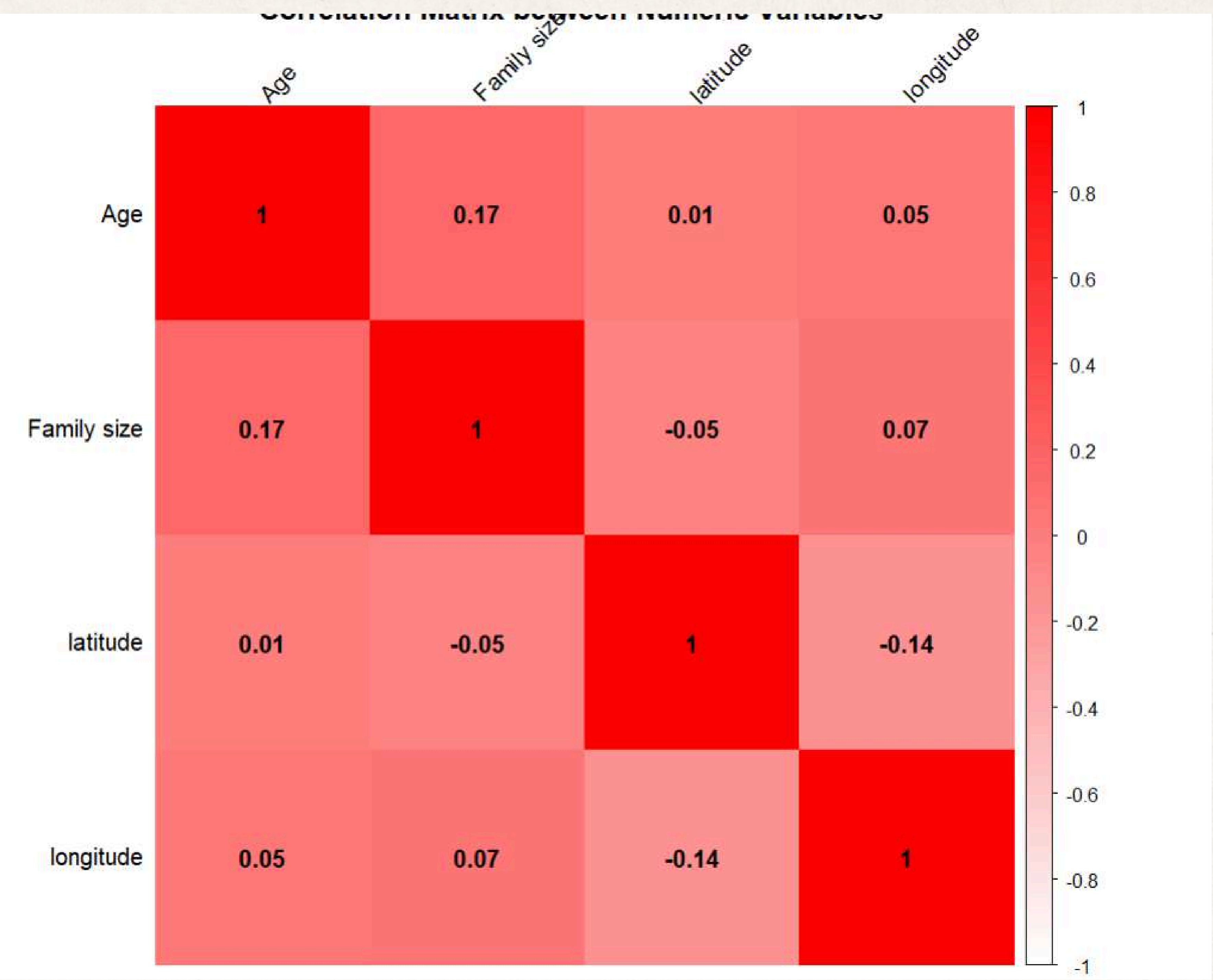
- The higher the income bracket, the older the age range and the more the plot skews towards an older age range
- Dominant age range around ~25
- Outliers present

DEMOGRAPHIC ANALYSIS

Correlation Matrix Among Numeric Data

```r
# Correlation Matrix Among Numeric Data
color <- colorRampPalette(c("white", "red"))(100)

correlation_matrix_graph <- data |> mutate(`Family size` = as.numeric(`Family size`)) |>
  select(Age, `Family size`, latitude, longitude) |>
  cor() |> corrplot(method = "color", , tl.col = "black", tl.srt = 45,
                    addCoef.col = "black", col = color, title = "Correlation Matrix between Numeric Variables")
correlation_matrix_graph
```

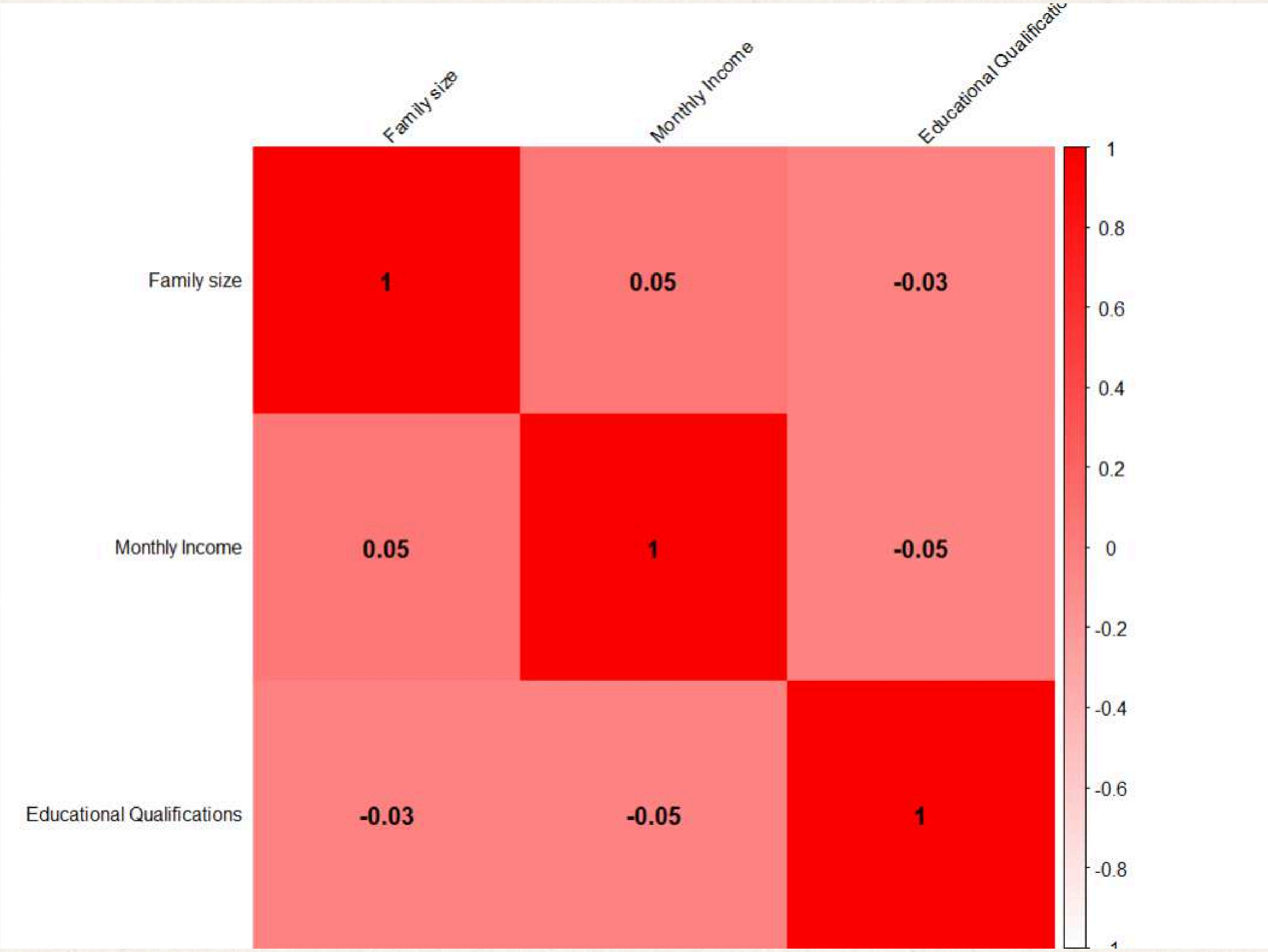
Weak Correlation among customer variables



DEMOGRAPHIC ANALYSIS

Correlation Matrix Among Ordinal Data

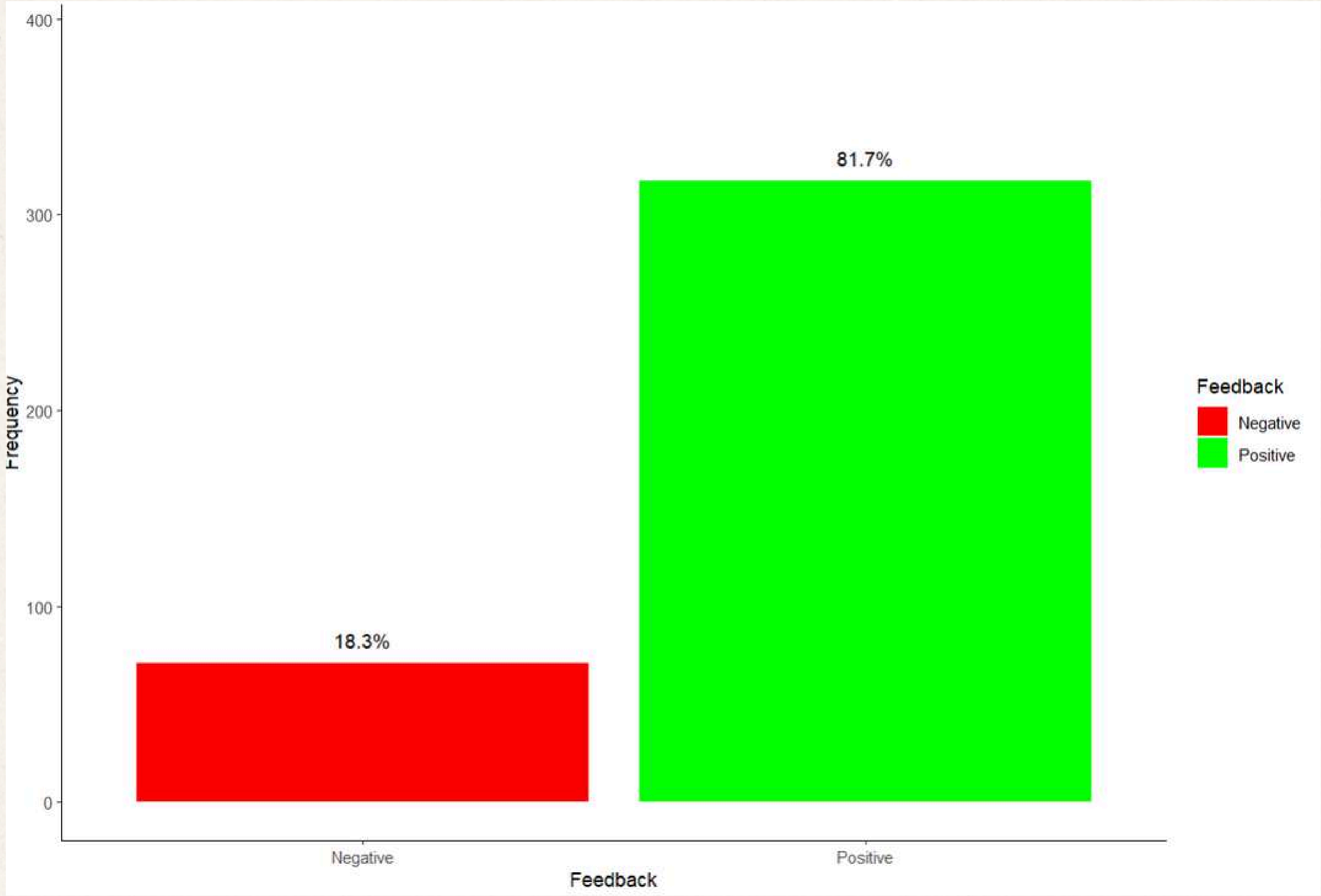
```r
#Correlation Matrix Among Ordinal Data
correlation_matrix_graph2 <- data |>  mutate(`Family size` = as.numeric(`Family size`), `Monthly Income` = as.numeric(`Monthly Income`),
  `Educational Qualifications` = as.numeric(`Educational Qualifications`)) |>
  select(`Family size`, `Monthly Income`, `Educational Qualifications`) |> cor(method = "kendall") |>
  corrplot(method = "color", tl.col = "black", tl.cex = 0.8, tl.srt = 45,
           addCoef.col = "black", col = color, title = "Correlation Matrix between Ordinal Variables")
correlation_matrix_graph2
```



FEEDBACK ANALYSIS

Count plot of Feedback

```
# Basic Feedback Analysis
feedbacks <- data |> group_by(Feedback) |> summarize(count = n()) |> mutate(percentage = round(count/sum(count) * 100, 2))
feedback_bar_graph <- feedbacks |> ggplot(aes(x = Feedback, y = count, fill = Feedback)) +
  geom_bar(stat = "identity") +
  labs(x = "Feedback", y = "Frequency") + scale_y_continuous(limit = c(0, sum(feedbacks$count))) +
  geom_text(aes(label = paste(percentage, "%", sep="")), vjust = -1) +
  scale_fill_manual(values = c("Positive" = "green", "Negative" = "red")) +
  theme_classic()
feedback_bar_graph
```

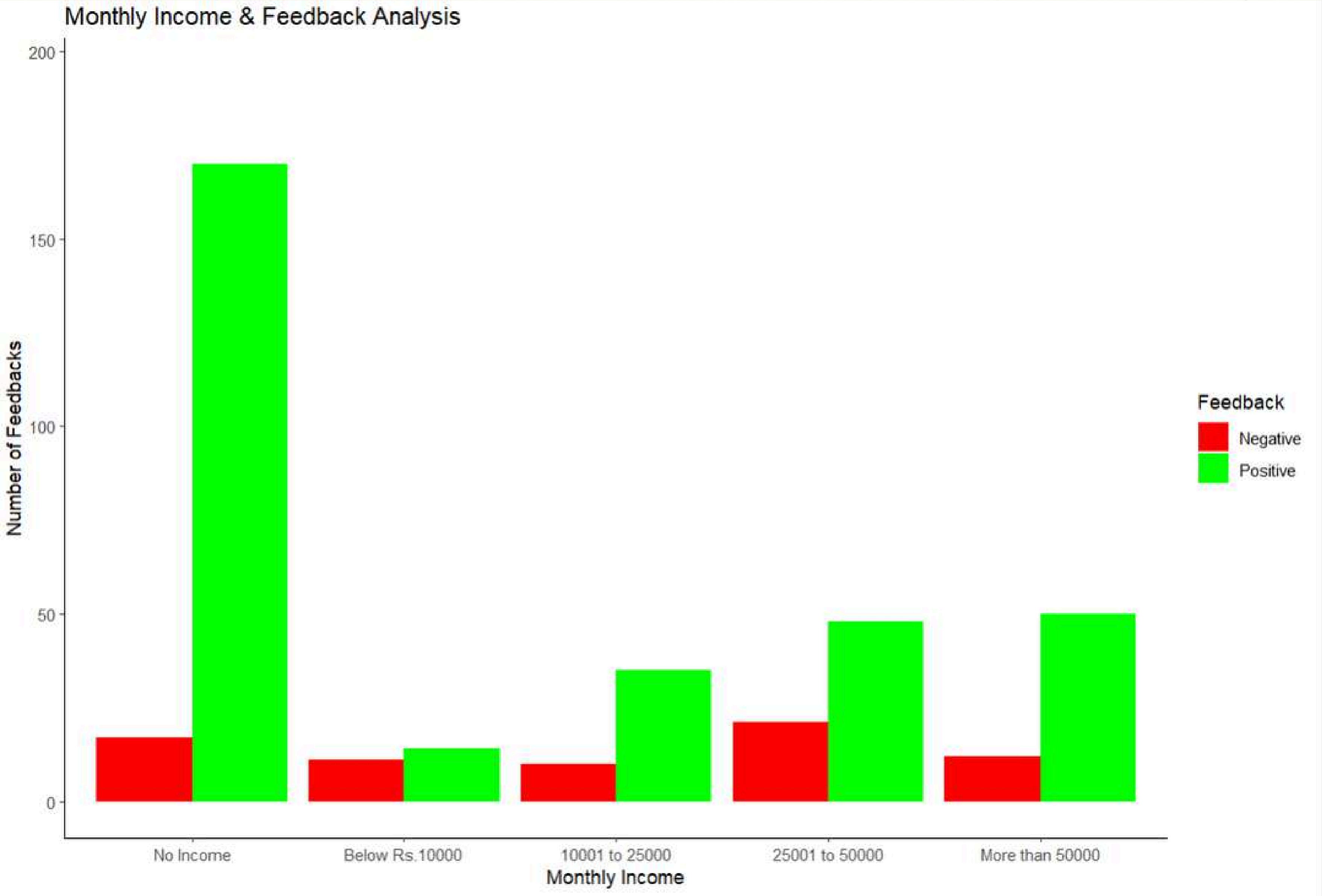


Most feedback are positive

FEEDBACK ANALYSIS

Feedback by Monthly Income

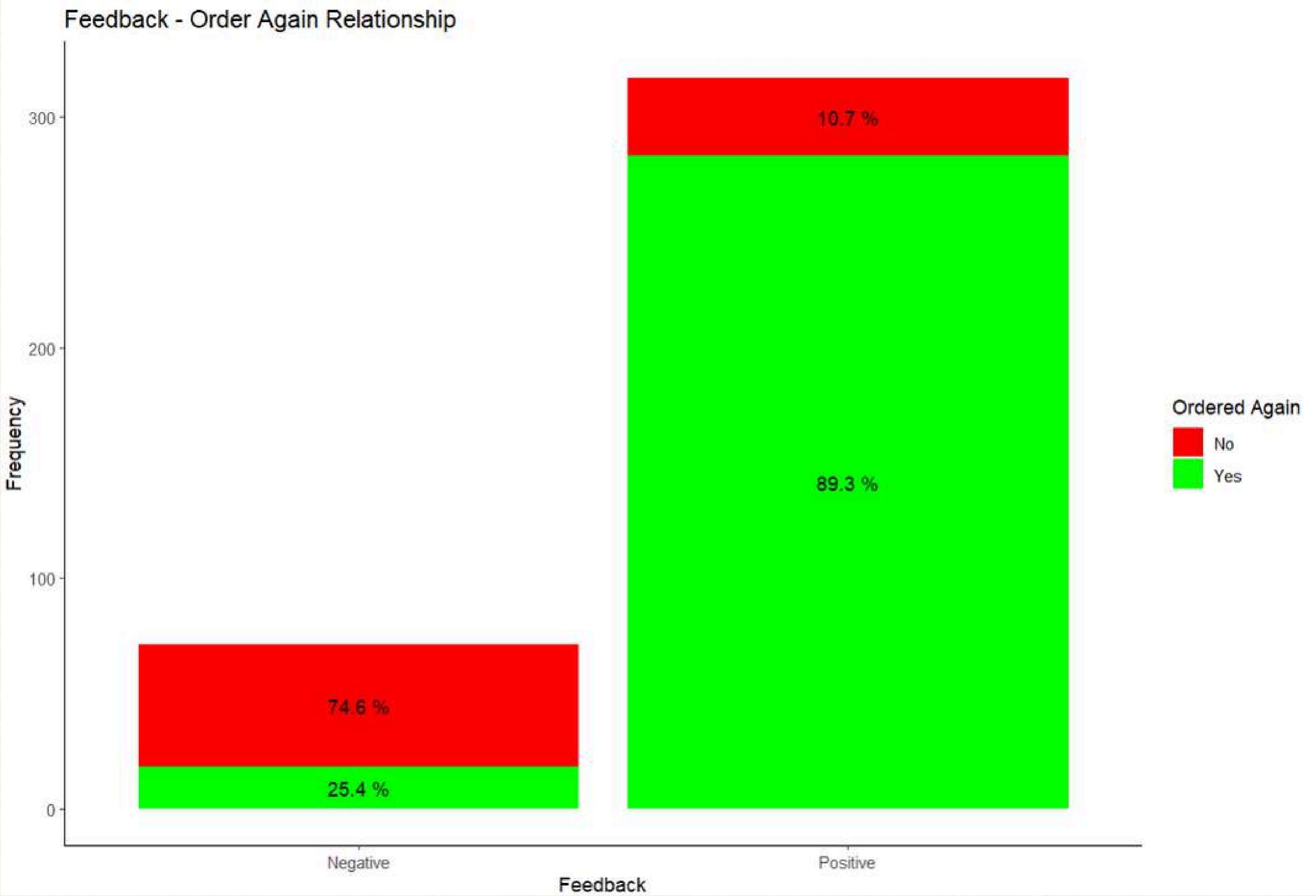
```r
# Feedback by Monthly Income
feedback_by_monthlyincome_graph <- data |> group_by(Feedback) |> ggplot(aes(x = `Monthly Income`, fill = Feedback)) +
  geom_bar(position = "dodge") +
  labs(x = "Monthly Income", y = "Number of Feedbacks", title = "Monthly Income & Feedback Analysis") +
  scale_y_continuous(limit = c(0, sum(feedbacks$count) / 2)) +
  scale_fill_manual(values = c("Positive" = "green", "Negative" = "red")) +
  theme_classic()
feedback_by_monthlyincome_graph
```



- The "No Income" group left the most positive feedback, also showing the best positivity ratio.
- Customers earning "Below Rs. 10000" provided an equal mix of positive and negative feedback.
- Uniform negative feedback across all income levels hints at common-faced issues in service.

FEEDBACK ANALYSIS

```r
data_used <- data |> group_by(Feedback, `Ordered Again`) |> summarize(count = n(), .groups = 'drop') |>
  group_by(Feedback) |> mutate(percentage = round((count / sum(count)) * 100, 1))
data_used <- data_used |> arrange(Feedback, desc(`Ordered Again`)) |> group_by(Feedback) |> mutate(label_position = cumsum(count)
feedback_v_orderedagain <- data_used |> ggplot(aes(x = Feedback, y = count, fill = `Ordered Again`)) +
  geom_bar(stat = "identity", position = "stack") +
  geom_text(aes(label = paste(percentage, "%"), y = label_position), vjust = 0.5) +
  labs(x = "Feedback", y = "Frequency", title = "Feedback - Order Again Relationship") +
  scale_fill_manual(values = c("Yes" = "green", "No" = "red")) +
  theme_classic()

feedback_v_orderedagain
```



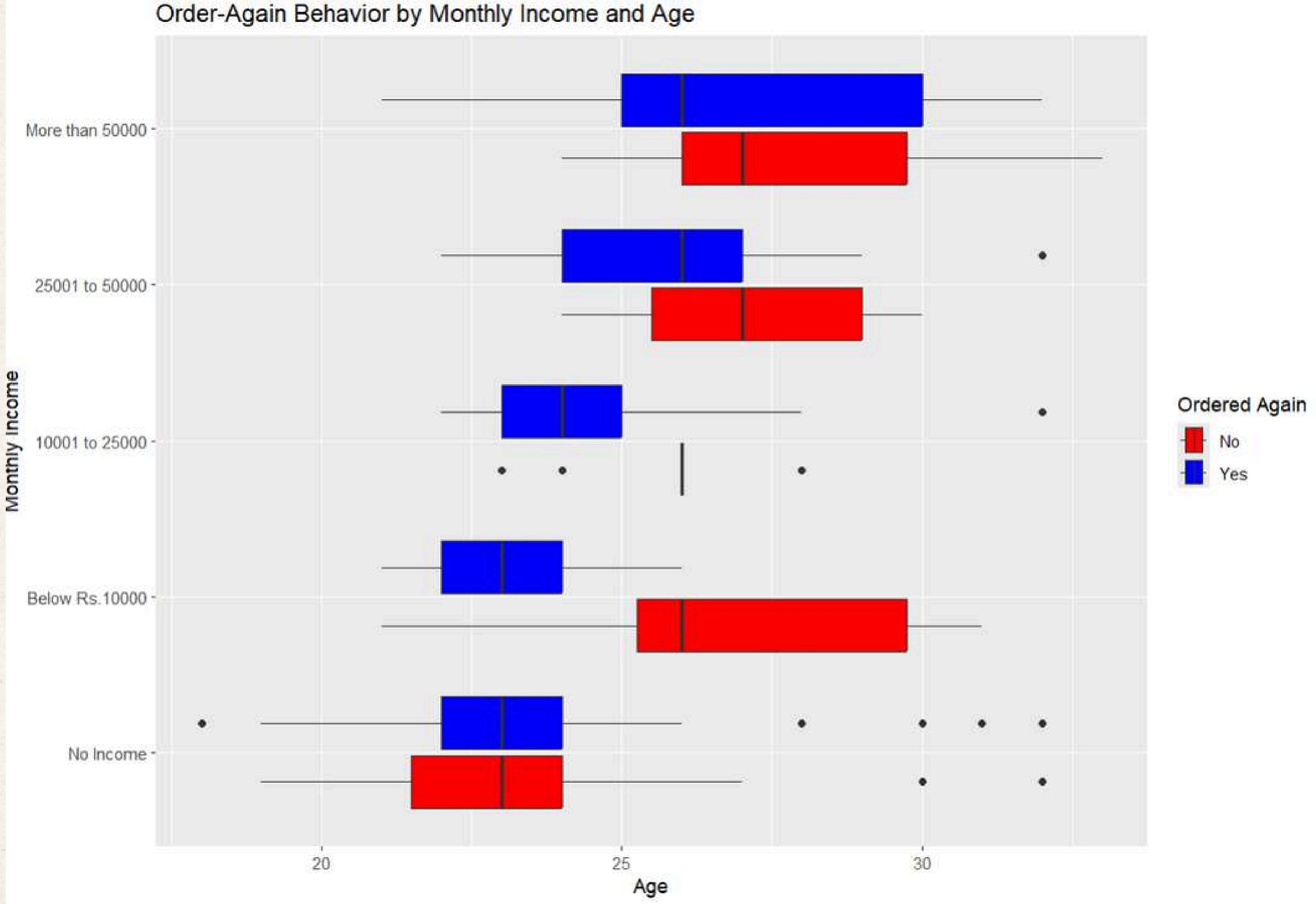
Feedback - Ordered Again Relationship

When feedbacks are positive, customers are more likely to order again.
When feedbacks are negative, customers are less likely to order again.

DEMOGRAPHIC ANALYSIS

```
#Ordering Behavior by Monthly Income and Age
monthlyincome_age_orderingbehavior <- data |> ggplot(aes(x = Age, y = `Monthly Income`, fill = `Ordered Again`)) +
  geom_boxplot() +
  scale_fill_manual(values = c("Yes" = "blue", "No" = "red"), seq(min(data$Age), max(data$Age), by = 1)) +
  labs(x = "Age", y = "Monthly Income", title = "Order-Again Behavior by Monthly Income and Age")
monthlyincome_age_orderingbehavior
```

Boxplot on Age Distribution by Monthly Income



Observations:

- The age distribution of those who ordered again and those who didn't are particularly different for Income groups "Below Rs.10000", "10001 to 25000" and "25001 to 50000", suggesting that age plays a big factor
- The age distribution of those who ordered again and those who didn't for the other income groups ("No Income" and "More than 50000" are overlapping, suggesting that age does not play a huge factor in the decision within these income groups

PREDICTIVE MODEL

Preparing the data for Machine Learning Feeding

```r
# Random Forest Model
# First, we convert all nominal data to binary values
data <- data |> mutate(`Marital Status` = as.integer(factor(`Marital Status`, levels = c("Prefer not to say", "Single", "Married"))),
                       Gender = as.integer(factor(Gender, levels = c("Male", "Female"))), Occupation = as.integer(factor(Occupation,
                       levels = c("Employee", "House wife", "Self Employeed", "Student"))),
                       Feedback = as.integer(factor(Feedback, levels = c("Negative", "Positive")))- 1)
data <- data |> select(-latitude, -longitude, -`Pin code`)
# removes unnecessary variables - we cannot ascertain the correlation between the location of the customer and whether they will order again,
# as the dataset does not give us the location of the customer RELATIVE to the restaurant location (This is one flaw of the dataset)
data <- data |> rename(MaritalStatus = `Marital Status`, MonthlyIncome = `Monthly Income`, EducationalQualifications = `Educational Qualifications`, FamilySize = `Family size`)
data$`Ordered Again` <- factor(data$`Ordered Again`, levels = c("Yes", "No"))
#This line of code above is very important -
# Make note that if the customer did not order again, the equivalent nominal value is 0. If they did order again, the equivalent nominal value is 1.
str(data)
```

```
> str(data)
tibble [388 × 9] (S3: tbl_df/tbl/data.frame)
 $ Age                      : num [1:388] 20 24 22 22 22 27 22 24 23 23 ...
 $ Gender                   : int [1:388] 2 2 1 2 1 2 1 2 2 2 ...
 $ MaritalStatus            : int [1:388] 2 2 2 2 2 3 2 2 2 2 ...
 $ Occupation               : int [1:388] 4 4 4 4 4 1 4 4 4 4 ...
 $ MonthlyIncome            : Ord.factor w/ 5 levels "No Income"<"Below Rs.10000"<..: 1 2 2 1 2 5 1 1 1 1 ...
 $ EducationalQualifications: Ord.factor w/ 5 levels "Uneducated"<"School"<..: 4 3 4 3 4 4 3 4 4 4 ...
 $ FamilySize               : Ord.factor w/ 6 levels "1"<"2"<"3"<"4"<..: 4 3 3 6 4 2 3 3 2 4 ...
 $ Ordered Again            : Factor w/ 2 levels "Yes","No": 1 1 1 1 1 1 1 1 1 1 ...
 $ Feedback                 : num [1:388] 1 1 0 1 1 1 1 1 1 1 ...
> 
```

All nominal data are converted to numerical for easy processing. All ordinal data converted to their level equivalents

PREDICTIVE MODEL

Preparing the data for Machine Learning Feeding

```
set.seed(2024) # For reproducibility of results
split_data <- createDataPartition(y=1:nrow(data), p = 0.7, list = FALSE) # splits our existing dataset 7:3
train_set <- data[split_data, ] # 70% of our data is used for training our model
test_set <- data[-split_data, ] # 30% of our data is used for testing our model
```

The data is split by 7 : 3 to a training set and a testing set.

A seed is set for reproducibility of results.

PREDICTIVE MODEL

Random Forest Model Creation: Original Model

```
randomforest_model <- randomForest(`Ordered Again`~ ., data = train_set, ntree = 100)
print(randomforest_model)
```

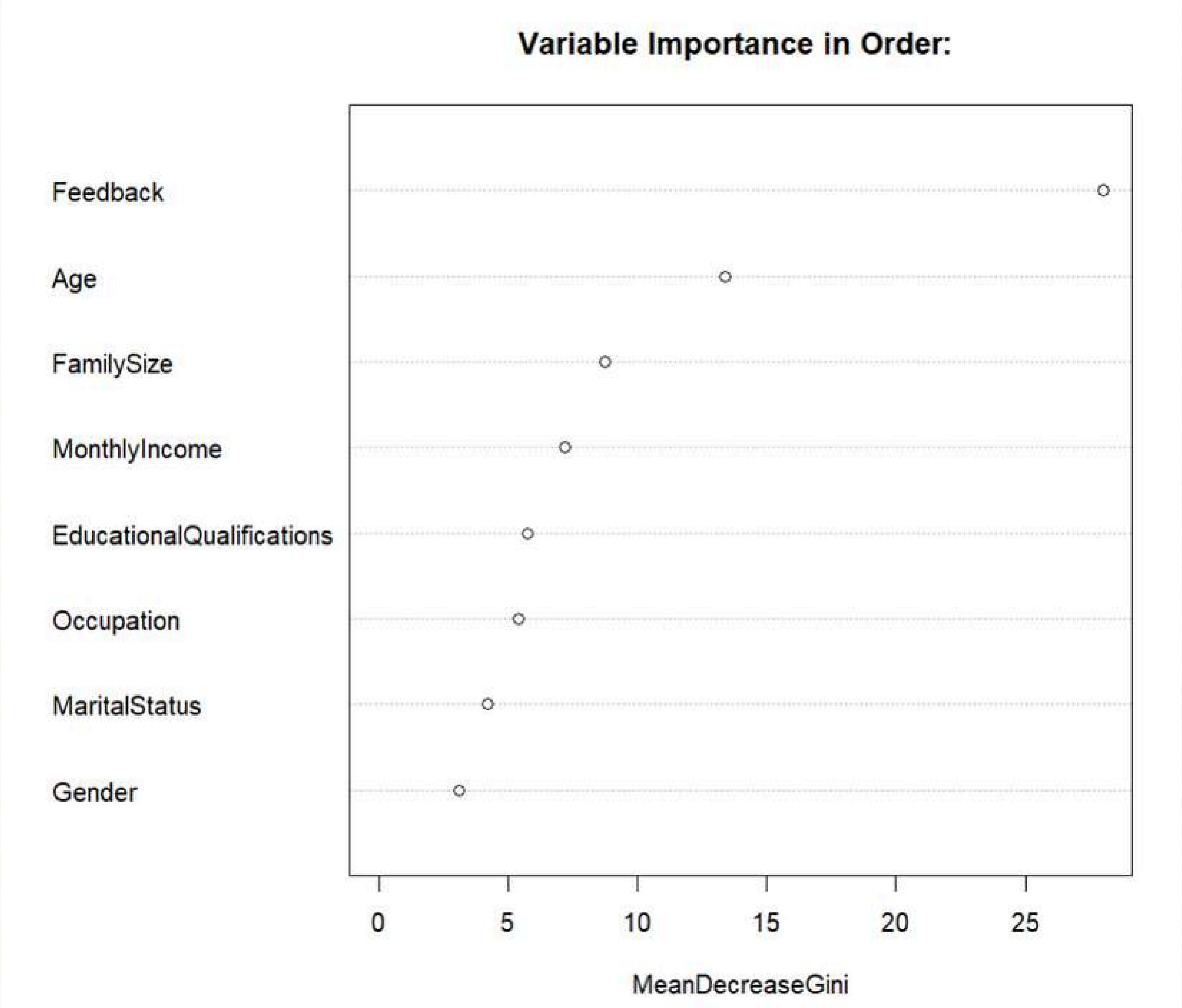
```
> print(randomforest_model)

Call:
 randomForest(formula = `Ordered Again` ~ ., data = train_set,      ntree = 100)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 2

        OOB estimate of  error rate: 12.87%
Confusion matrix:
    Yes No class.error
Yes 203  6  0.02870813
No   29 34  0.46031746
```

Training data set is fed into random forest model

PREDICTIVE MODEL

Which variables affect customer reordering behavior the most?

```
varImpPlot(randomforest_model, main = paste("Variable Importance in Order: "))
```



Unsurprisingly, the feedback of the customer is the greatest indicator as to whether the customer would reorder in the future.

PREDICTIVE MODEL

Evaluating the Average Customer

```
# Creating the average customer and predicting whether they will reorder
average_customer <- data.frame(
  Age = median(data$Age),
  MaritalStatus = as.integer(median(as.integer(data$MaritalStatus))),
  Gender = as.integer(median(as.integer(data$Gender))),
  Occupation = as.integer(median(as.integer(data$Occupation))),
  Feedback = as.integer(median(as.integer(data$Feedback))),
  MonthlyIncome = as.integer(median(as.integer(data$MonthlyIncome))),
  EducationalQualifications = as.integer(median(as.integer(data$EducationalQualifications))),
  FamilySize = as.integer(median(as.integer(data$FamilySize)))
)
print(average_customer)
# Predicting with the random forest model
average_customer_prediction <- predict(randomforest_model, newdata = average_customer)
print(average_customer_prediction)
```

```
> print(average_customer)
  Age MaritalStatus Gender Occupation Feedback MonthlyIncome EducationalQualifications FamilySize
1  24             2      1          4        1             2                         4          3
> print(average_customer_prediction)
  1
Yes
Levels: Yes No
```

Created one observation in a data frame with the average values of all the variables, thereby creating the average customer. Evaluated this customer with our model to see whether they would reorder again. The answer was YES.

PREDICTIVE MODEL

Random Forest Model Creation: Prediction Generation and Testing

Prediction generation creation of confusion matrix

```
predictions <- predict(randomforest_model, test_set) #generate predictions with our test set

cm_yes <- confusionMatrix(predictions, test_set$`Ordered Again`, positive = "Yes") # positive class is Yes
```

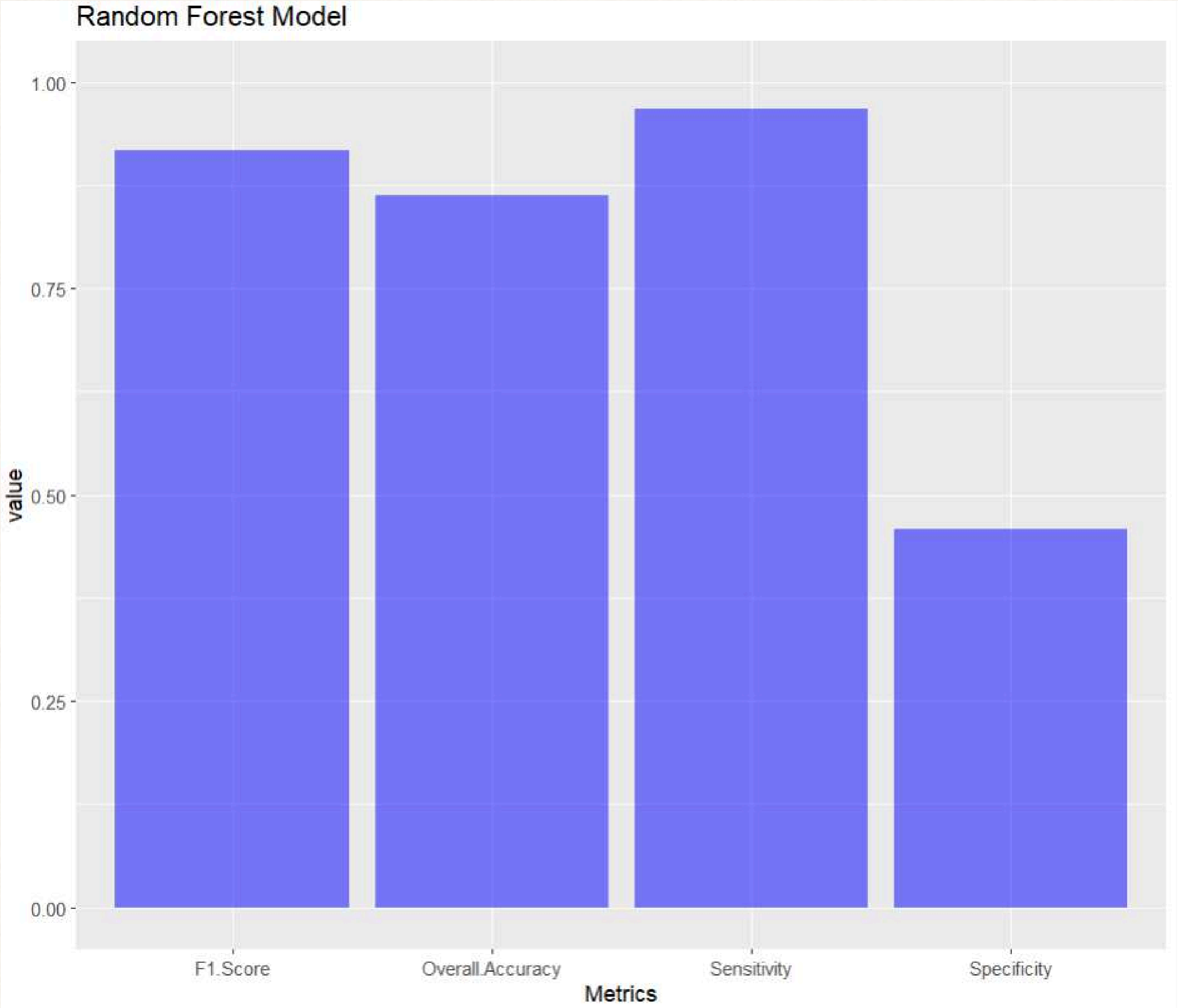
Extracting metrics from the confusion matrix and putting it in a summary table

```
#Extract accuracy, sensitivity, and specificity from the matrix
accuracy_yes <- cm_yes$overall[["Accuracy"]]
sensitivity_yes <- cm_yes$byClass['Sensitivity']
specificity_yes <- cm_yes$byClass['Specificity']

#Generate f1
f1_score <- F_meas(predictions, test_set$`Ordered Again`, positive = "Yes")

summary_table <- pivot_longer(data.frame(`Overall Accuracy` = accuracy_yes, Sensitivity = sensitivity_yes,
                                        Specificity = specificity_yes, `F1 Score` = f1_score), cols = everything())
print(summary_table)
summary_graph <- summary_table |> ggplot(aes(x = name, y = value)) +
  geom_bar(stat = "identity", fill = "blue", alpha = 0.5) +
  scale_y_continuous(limits = c(0, 1)) +
  labs(x = "Metrics", title = "Random Forest Model")
print(summary_graph)
# As can be seen, the specificity is much lower compared to the other metrics
```

Predictions were generated using our test data set
Predictions were compared against my test data set.
Metrics were retrieved.

PREDICTIVE MODEL

Random Forest Model Creation: Results




```
> print(summary_table)
# A tibble: 4 x 2
  name             value
  <chr>            <dbl>
1 Overall.Accuracy 0.862
2 Sensitivity      0.967
3 Specificity      0.458
4 F1.Score         0.918
```

PREDICTIVE MODEL

Random Forest Model Creation: Improving the Model

```r
## Method Cross-Validation & Emphasizing on Optimizing Specificity
train_control <- trainControl(method = "cv", number = 5, summaryFunction = twoClassSummary) # Standard 5-fold cross valdiation

# Train the Random Forest model with cross-validation this time
randomforest_model_cv <- train(`Ordered Again` ~ ., data = train_set, method = "rf", trControl = train_control, metric = "Spec", tuneLength = 20)
```

Utilized trainControl() function for 5-fold cross validation.

Set metric parameter to emphasize on Specificity.

PREDICTIVE MODEL

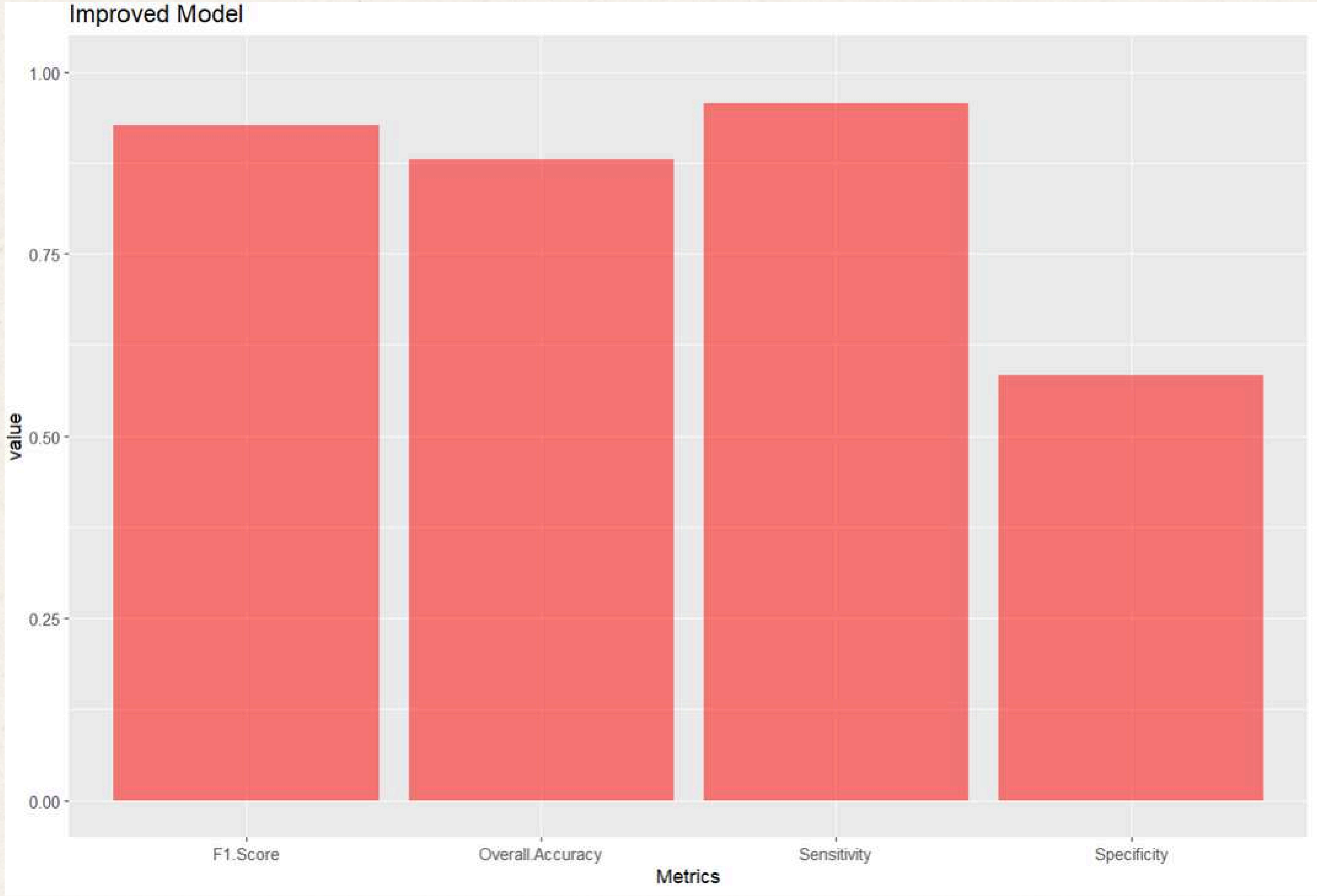
Random Forest Model Creation: Improving the Model

Same steps as before

```
predictions_cv <- predict(randomforest_model_cv, newdata = test_set)

# Generate Confusion Matrix
cm_cv <- confusionMatrix(predictions_cv, test_set$`Ordered Again`, positive = "Yes")
print(cm_cv)

accuracy_v2 <- cm_cv$overall[["Accuracy"]]
sensitivity_v2 <- cm_cv$byClass['Sensitivity']
specificity_v2 <- cm_cv$byClass['Specificity']

#Generate f1
f1_score_v2 <- F_meas(predictions_cv, test_set$`Ordered Again`, positive = "Yes")

summary_table_v2 <- pivot_longer(data.frame(`Overall Accuracy` = accuracy_v2, Sensitivity = sensitivity_v2,
                                            Specificity = specificity_v2, `F1 Score` = f1_score_v2), cols = everything())
print(summary_table_v2)
summary_graph_v2 <- summary_table_v2 |> ggplot(aes(x = name, y = value)) + geom_bar(stat = "identity", fill = "red", alpha = 0.5) +
  scale_y_continuous(limits = c(0, 1)) +
  labs(x = "Metrics", title = "Improved Model")
print(summary_graph_v2)
```

PREDICTIVE MODEL

Random Forest Model Creation: Improved Results

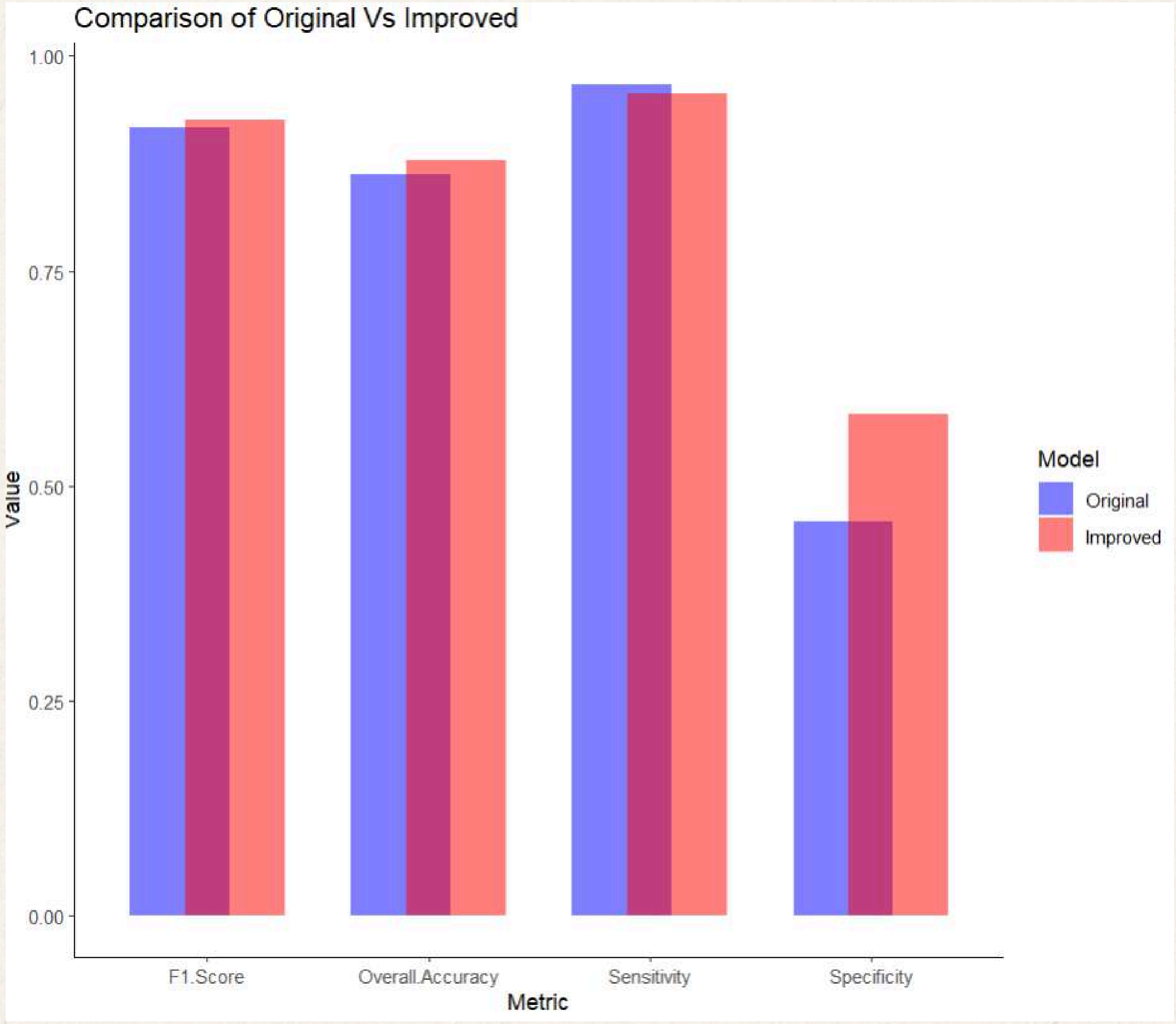



```
> print(summary_table_v2)
# A tibble: 4 × 2
  name             value
  <chr>            <dbl>
1 Overall.Accuracy 0.879
2 Sensitivity      0.957
3 Specificity      0.583
4 F1.Score         0.926
```

PREDICTIVE MODEL

Random Forest Model Creation: Comparison Between Results

```
summary_table$Model <- "Original"
summary_table_v2$Model <- "Improved"

# Combine summary tables
combined_summary <- rbind(summary_table, summary_table_v2) |> mutate(Model = factor(Model, levels = c("Original", "Improved")))

# Plot combined summary tables
combined_summary_plot <- ggplot(combined_summary, aes(x = name, y = value, fill = Model)) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.5), alpha = 0.5) +
  scale_fill_manual(values = c("Original" = "blue", "Improved" = "red")) +
  labs(title = "Comparison of Original Vs Improved", x = "Metric", y = "Value") +
  theme(legend.position = "top") +
  theme_classic()


# Print the combined plot
print(combined_summary_plot)
```



Although there is a slight decrease in sensitivity, F1 Score and Overall Accuracy both slightly increased.
Most positively, the specificity of the model increased tremendously

Limitations

1. Limited Observations
2. Limited Variables
3. Concentrated cultural demographic

CONCLUSIONS

THANK YOU

Reach out for inquiries.

noahlie07@gmail.com