Final Team Project

by Tommy Barron, April Chia, and Taylor Kirk

Data Importing and Pre-processing

Import data set and describe characteristics

- Dataset 3
- **Command:** df <- read_csv("online_shoppers_intention.csv")
  dim(df)
  str(df)

```
[1] 12330    18
spc_tbl_ [12,330 × 18] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ Administrative         : num [1:12330] 0 0 0 0 0 0 0 1 0 0 ...
 $ Administrative_Duration: num [1:12330] 0 0 0 0 0 0 0 0 0 0 ...
 $ Informational          : num [1:12330] 0 0 0 0 0 0 0 0 0 0 ...
 $ Informational_Duration : num [1:12330] 0 0 0 0 0 0 0 0 0 0 ...
 $ ProductRelated         : num [1:12330] 1 2 1 2 10 19 1 0 2 3 ...
 $ ProductRelated_Duration: num [1:12330] 0 64 0 2.67 627.5 ...
 $ BounceRates            : num [1:12330] 0.2 0 0.2 0.05 0.02 ...
 $ ExitRates              : num [1:12330] 0.2 0.1 0.2 0.14 0.05 ...
 $ PageValues             : num [1:12330] 0 0 0 0 0 0 0 0 0 0 ...
 $ SpecialDay             : num [1:12330] 0 0 0 0 0 0 0.4 0 0.8 0.4 ...
 $ Month                  : chr [1:12330] "Feb" "Feb" "Feb" "Feb" ...
 $ OperatingSystems       : num [1:12330] 1 2 4 3 3 2 2 1 2 2 ...
 $ Browser                : num [1:12330] 1 2 1 2 3 2 4 2 2 4 ...
 $ Region                 : num [1:12330] 1 1 9 2 1 1 3 1 2 1 ...
 $ TrafficType            : num [1:12330] 1 2 3 4 4 3 3 5 3 2 ...
 $ VisitorType            : chr [1:12330] "Returning_Visitor" "Returning_Visitor"
"Returning_Visitor" "Returning_Visitor" ...
 $ Weekend                : logi [1:12330] FALSE FALSE FALSE FALSE TRUE FALSE ...
 $ Revenue                : logi [1:12330] FALSE FALSE FALSE FALSE FALSE FALSE ...
```

Duplicates

- **Command:** df_dup <- df[duplicated(df),]
  dim(df_dup)

```
[1] 121  18
```

- **Command:** df <- df[!duplicated(df),]
- **Purpose:** To remove duplicate rows

Finding missing values

Command: na_counts <- sapply(df, function(x) sum(is.na(x)))
na_counts

```
 Administrative Administrative_Duration          Informational  Informational_Duration
              0                       0                    128                       0
 ProductRelated ProductRelated_Duration            BounceRates               ExitRates
              0                       0                      0                       0
     PageValues              SpecialDay                  Month        OperatingSystems
            135                       0                      0                     123
        Browser                  Region            TrafficType             VisitorType
              0                       0                      0                       0
        Weekend                 Revenue
              0                       0
```

Outliers

- Viewed boxplots
- **Result**: left outliers in the data set

Handling missing values (Informational)

```r
# Informational column without the 0's
info_nzero <- subset.data.frame(df[df$Informational > 0, ])

# Probability table for unique values
probability <- table(info_nzero$Informational, useNA = "no")
prob_table <- prop.table(probability)

# Function to randomly impute one of those values based on their proportion
random_impute <- function(values, prob_table, size) {
  sample(values, size, replace = T, prob = prob_table)
}

# Separating the NA Information rows from the ones where InformationDuration is greater than 0, and the ones
where it's less than or equal to 0
na_indices <- which(is.na(df$Informational))
na_indices_great_zero <- na_indices[df[na_indices, "Informational_Duration"] > 0]
na_indices_zero_or_less <- na_indices[df[na_indices, "Informational_Duration"] <= 0]

# Converting the probability table of unique values to numeric
unique_values <- as.numeric(names(prob_table))
probs <- as.numeric(prob_table)
```

Command: df$Informational[na_indices_great_zero] <- random_impute(unique_values, probs, length(na_indices_great_zero))

df$Informational[is.na(df$Informational)] <- 0

Handling missing values (PageValues)

```
# Subsetting the PageValues NA values

pv_na <- df[!is.na(df$PageValues), ]

# Separate PageValues if Administrative, Information, or ProductRelated columns are greater than 0

pv_na <- pv_na[pv_na$Administrative > 0 | pv_na$Informational > 0 | pv_na$ProductRelated > 0, ]
summary(pv_clean$PageValues)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.000   0.000   0.000   5.973   0.000 361.764
```

Command: df$PageValues[is.na(df$PageValues)] <- 0

Handling missing values (OperatingSystems)

- 123 missing values

Command: df <- df[!is.na(df$OperatingSystems),]

Purpose: To remove rows with missing values

Transformation of the data

- Recoding the SpecialDay column to a categorical
  - **Command:**

```
df$SpecialDayGroup <- cut(df$SpecialDay,
                          breaks = c(-Inf, .2, .4, .6, .8, 1, Inf),
                          labels = c("Far", "Moderately Close",
                                     "Close", "Very Close",
                                     "Too Close for on-time delivery", "Ideal Time"),
                          include.lowest = TRUE, right = FALSE)
```

- Converting Month column to factor
  - **Command:** df$Month_factor <- as.factor(df$Month)

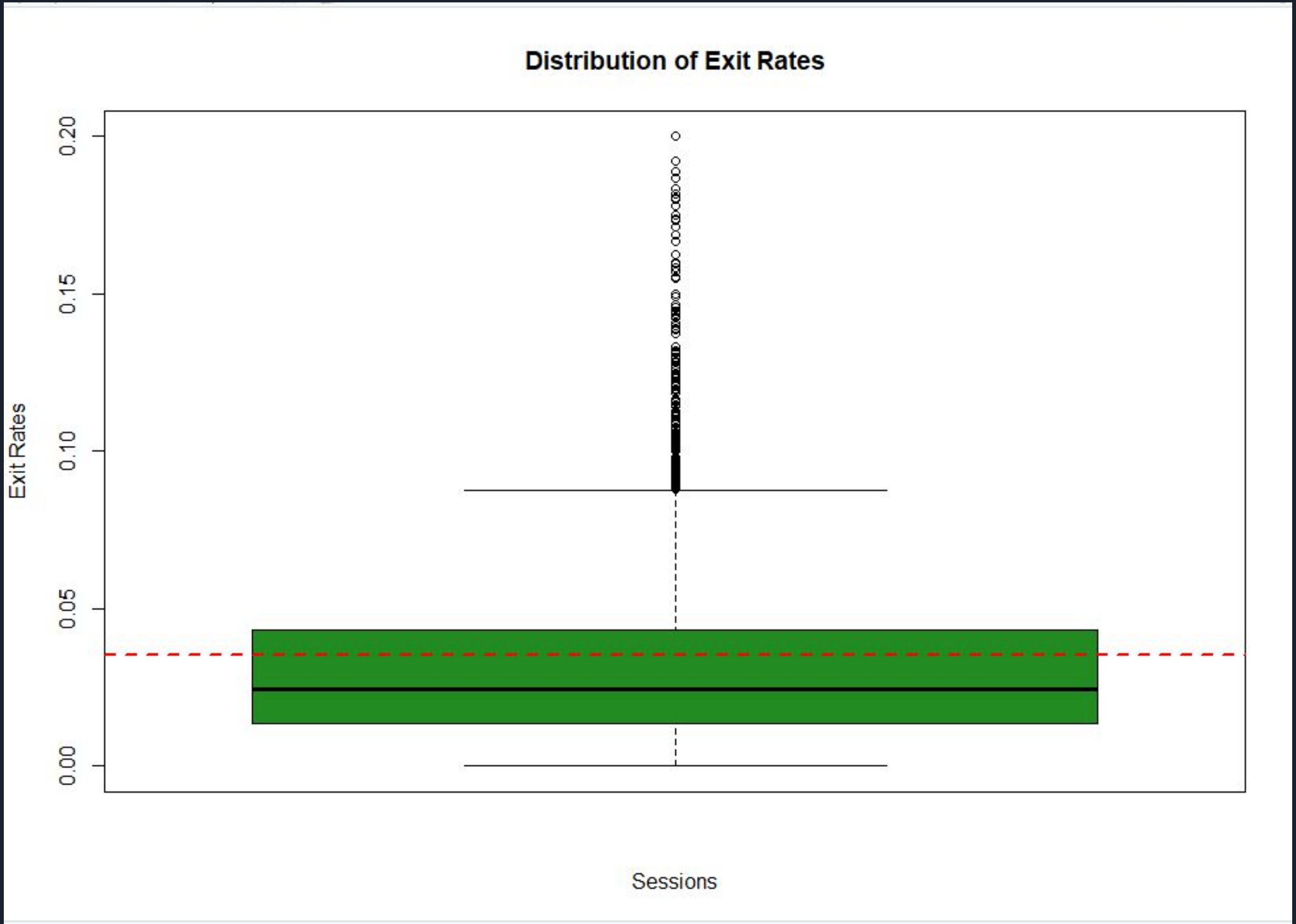
- Converting BounceRates and ExitRates into percentage values
  - **Command:**

```
df_percent <- df
df_percent$BounceRates <- df_percent$BounceRates * 100
df_percent$ExitRates <- df_percent$ExitRates * 100
```

Cleaned dataframe

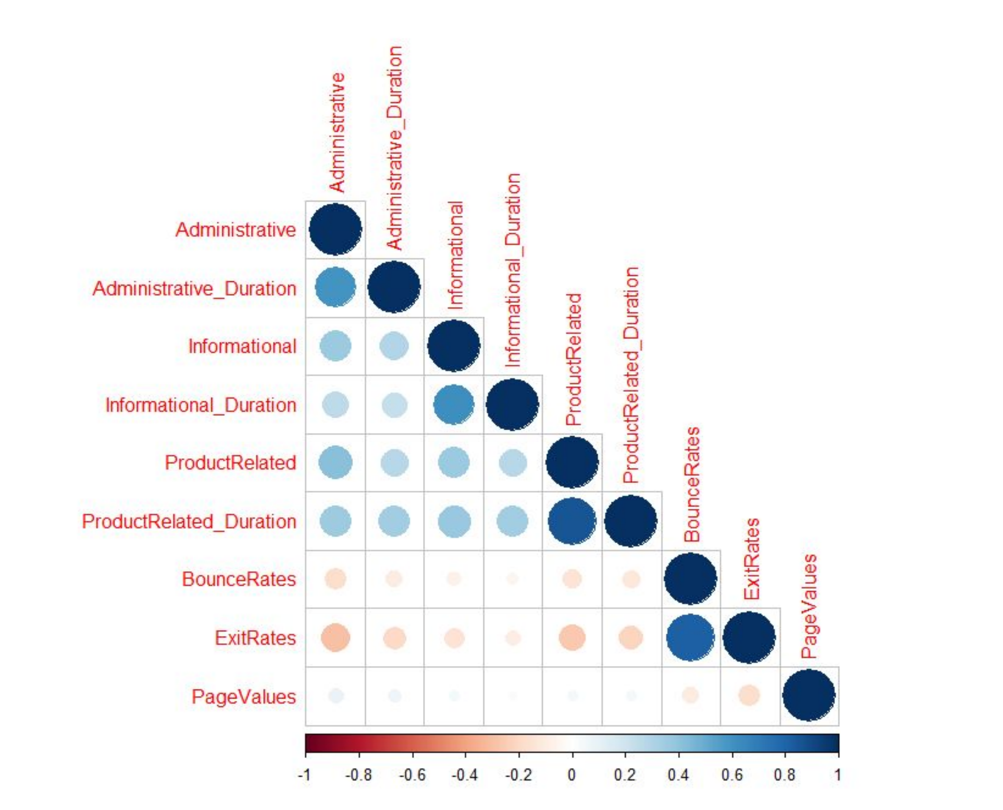
Command: data <- df_percent

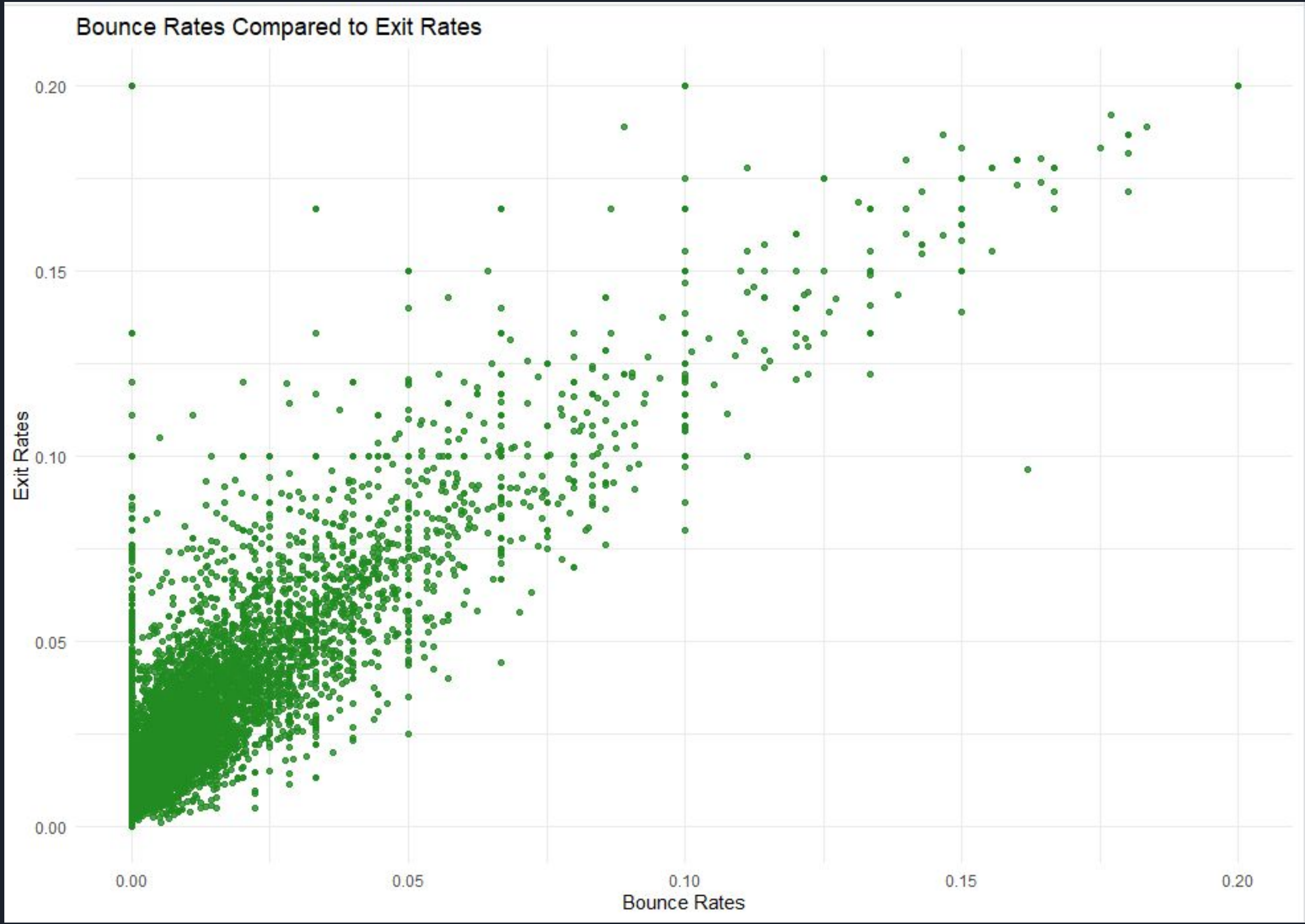
Data Analysis and Visualization

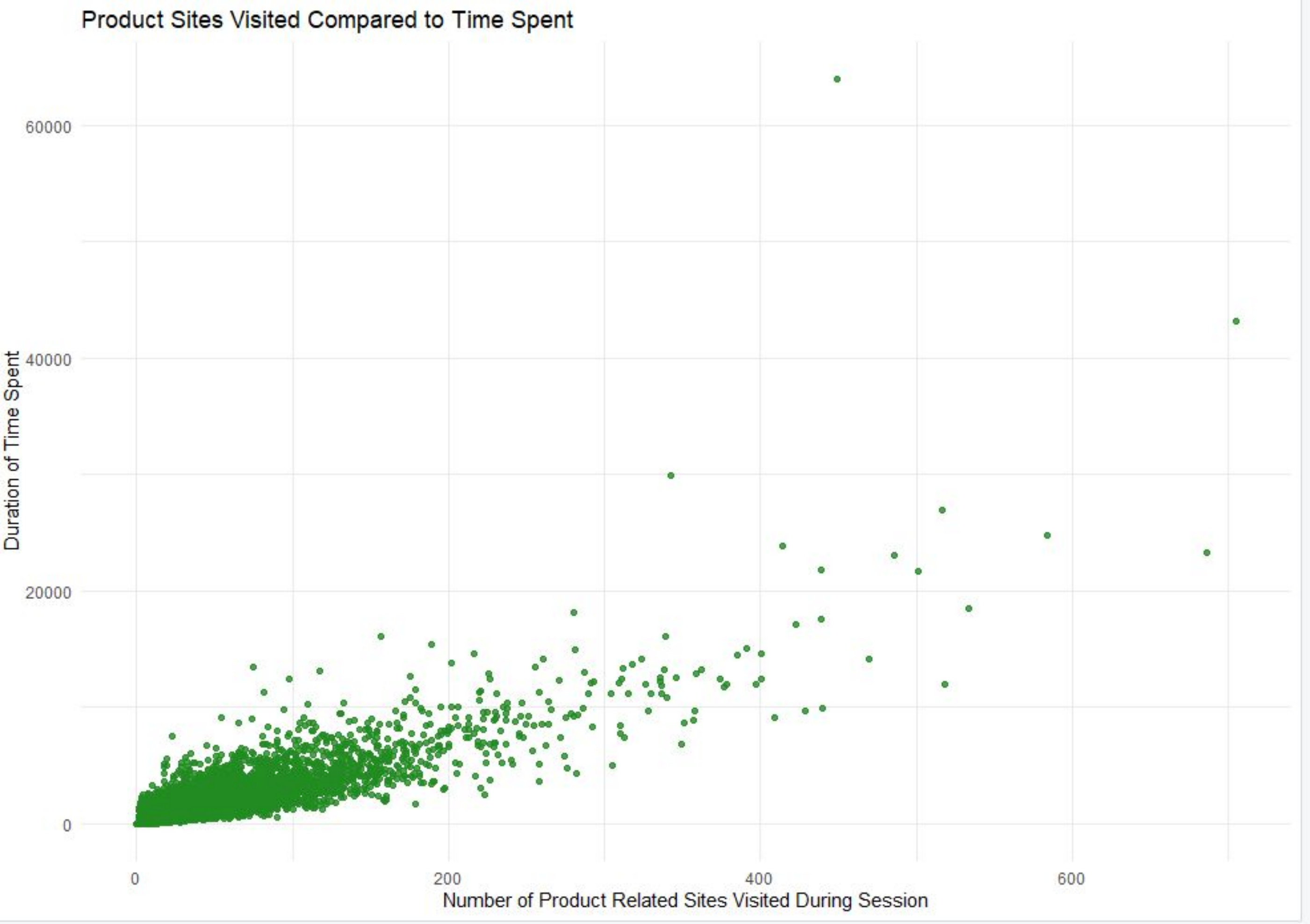


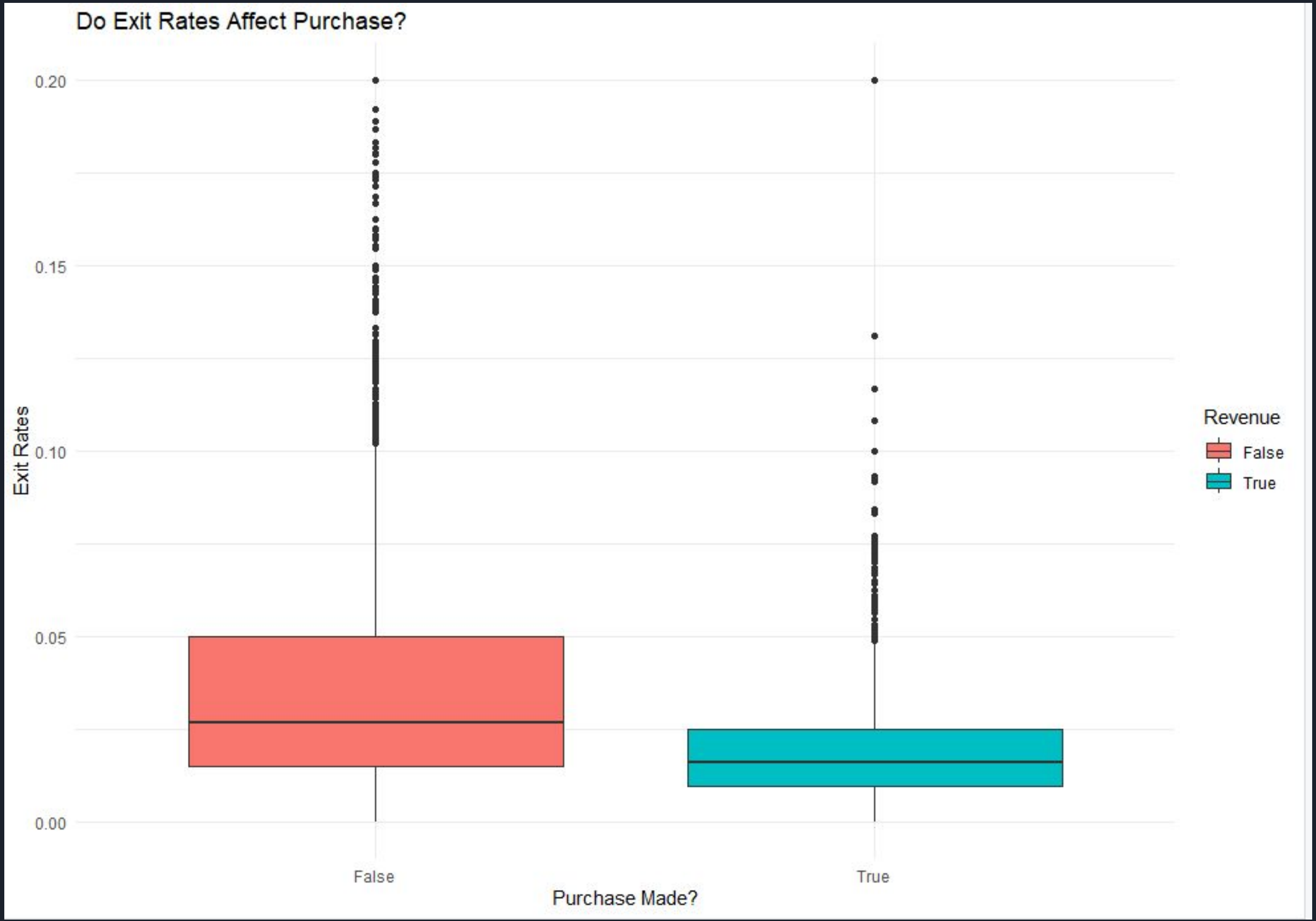
Distribution of Exit Rates
Exit Rates
0.00
0.05
0.10
0.15
0.20
Sessions

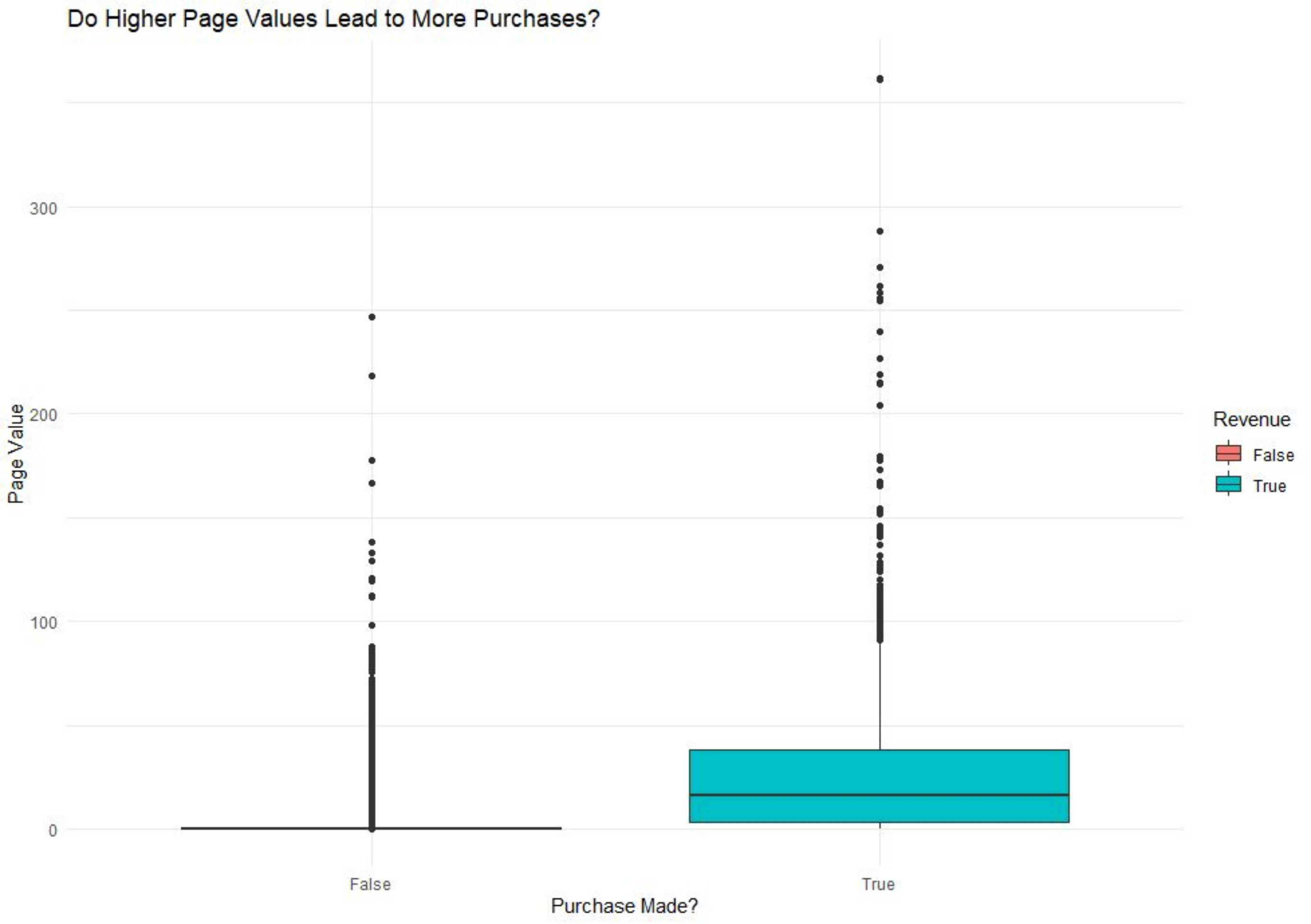
Correlation Plot



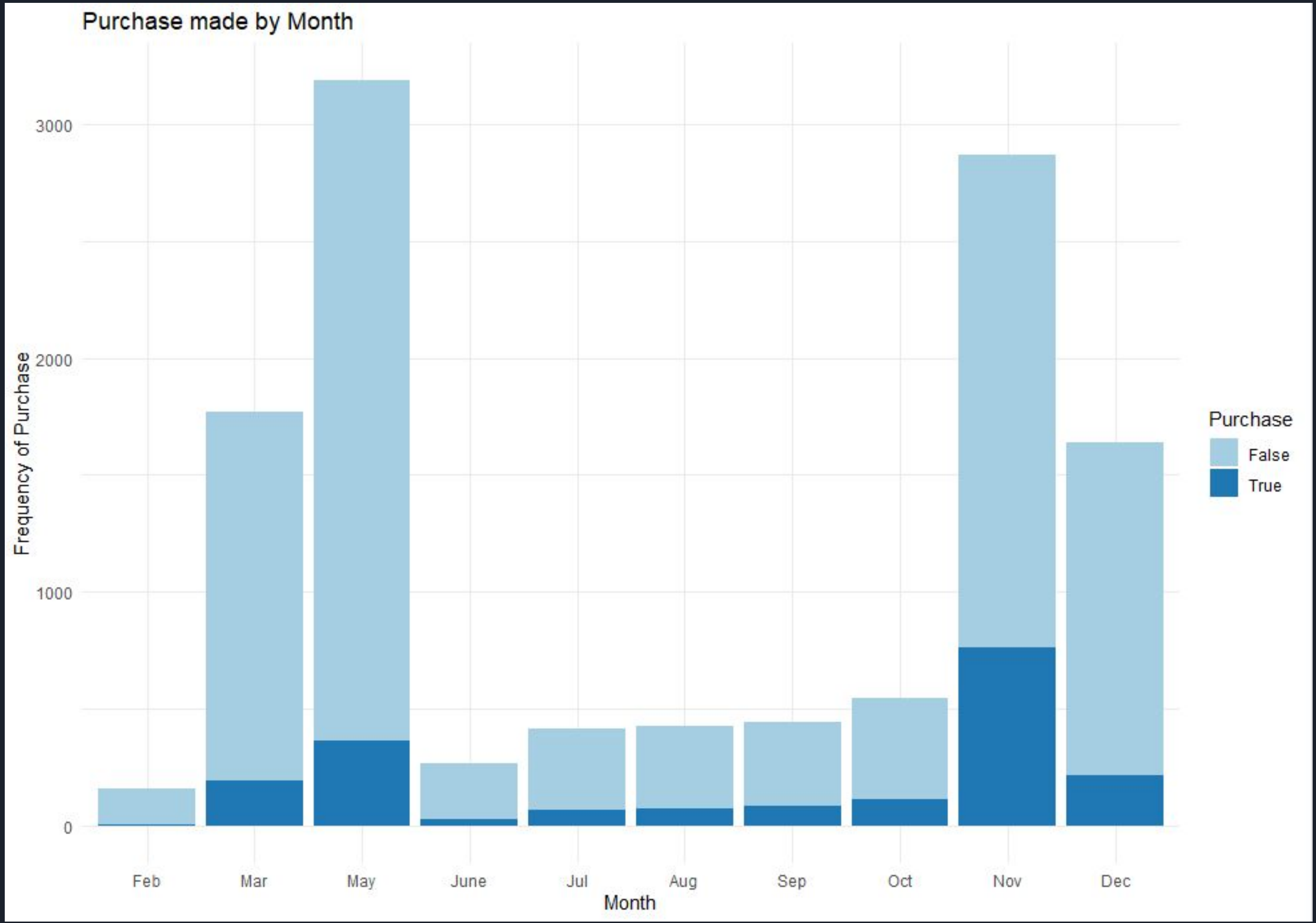

Bounce Rates Compared to Exit Rates
Exit Rates
Bounce Rates
0.00
0.05
0.10
0.15
0.20
0.00
0.05
0.10
0.15
0.20


Product Sites Visited Compared to Time Spent
Duration of Time Spent
60000
40000
20000
0
0
200
400
600
Number of Product Related Sites Visited During Session


Do Exit Rates Affect Purchase?
0.20
0.15
0.10
0.05
0.00
Exit Rates
False
True
Purchase Made?
Revenue
False
True

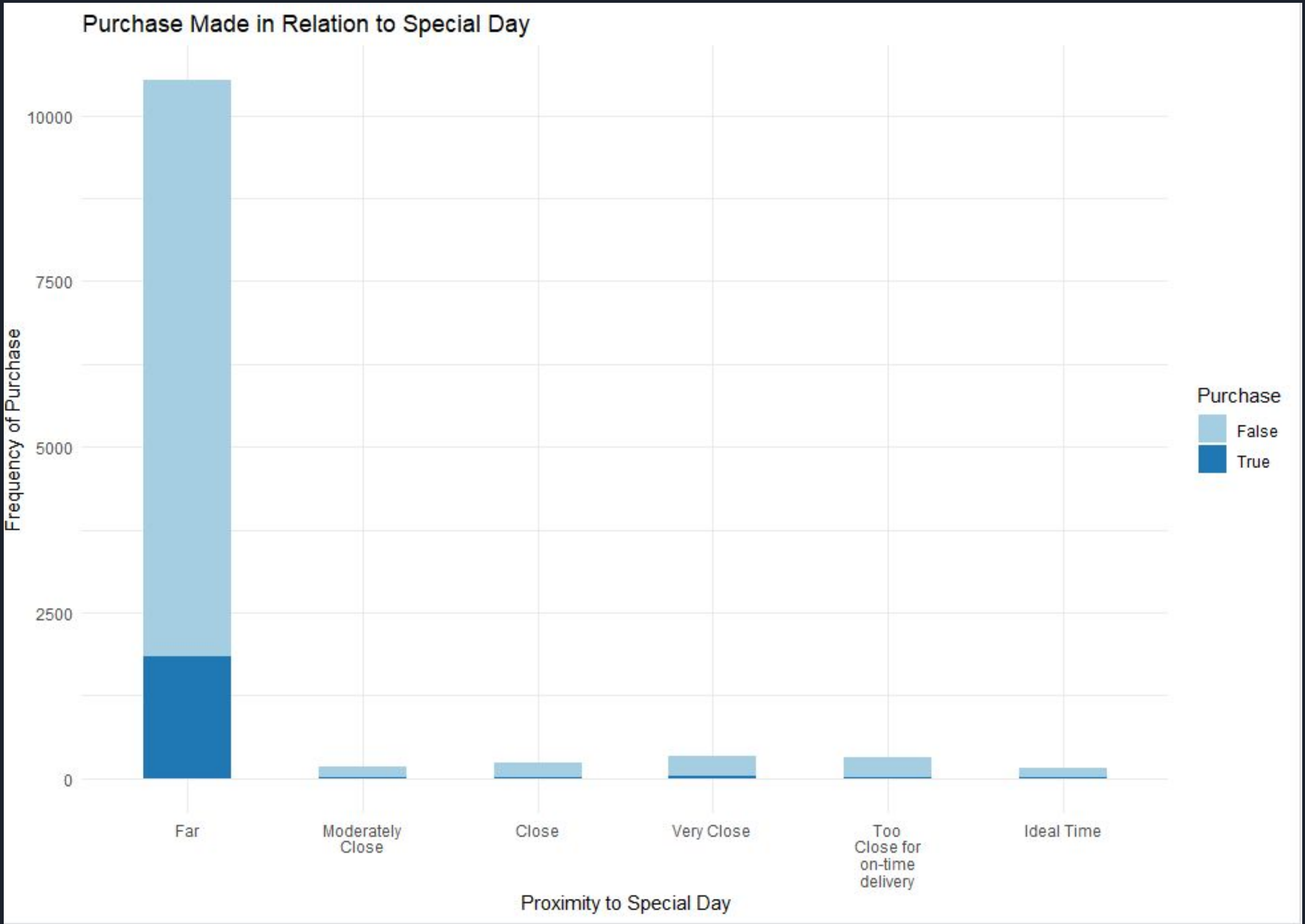


Do Higher Page Values Lead to More Purchases?
Page Value
300
200
100
0
False
True
Purchase Made?
Revenue
False
True

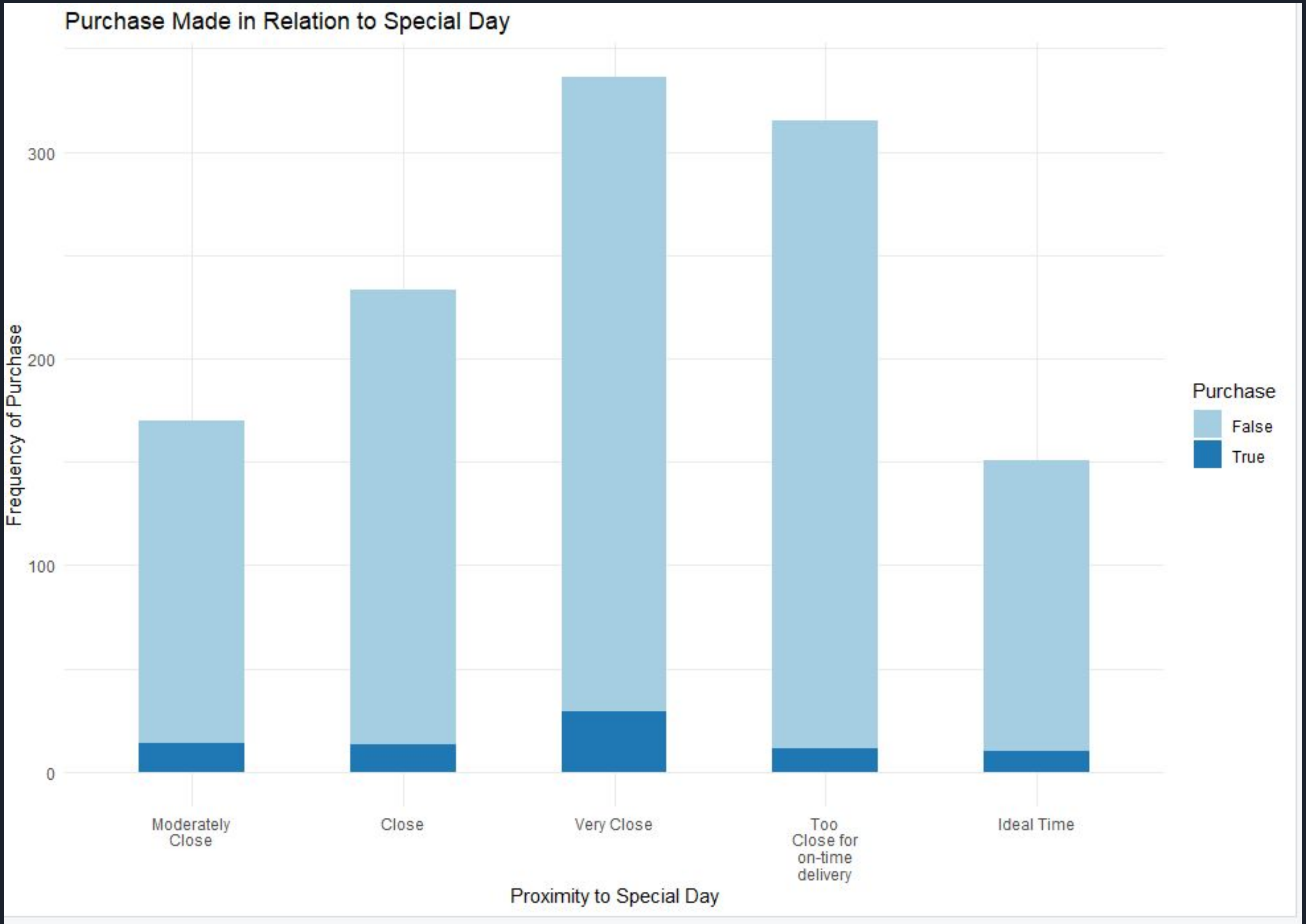

Purchase made by Month
Frequency of Purchase
3000
2000
1000
0
Feb
Mar
May
June
Jul
Aug
Sep
Oct
Nov
Dec
Month
Purchase
False
True

Proportion Table for Month Category

	False	True
Aug	0.82201405	0.17798595
Dec	0.86797066	0.13202934
Feb	0.98136646	0.01863354
Jul	0.84210526	0.15789474
June	0.89259259	0.10740741
Mar	0.89164786	0.10835214
May	0.88543628	0.11456372
Nov	0.73500697	0.26499303
Oct	0.79014599	0.20985401
Sep	0.80630631	0.19369369



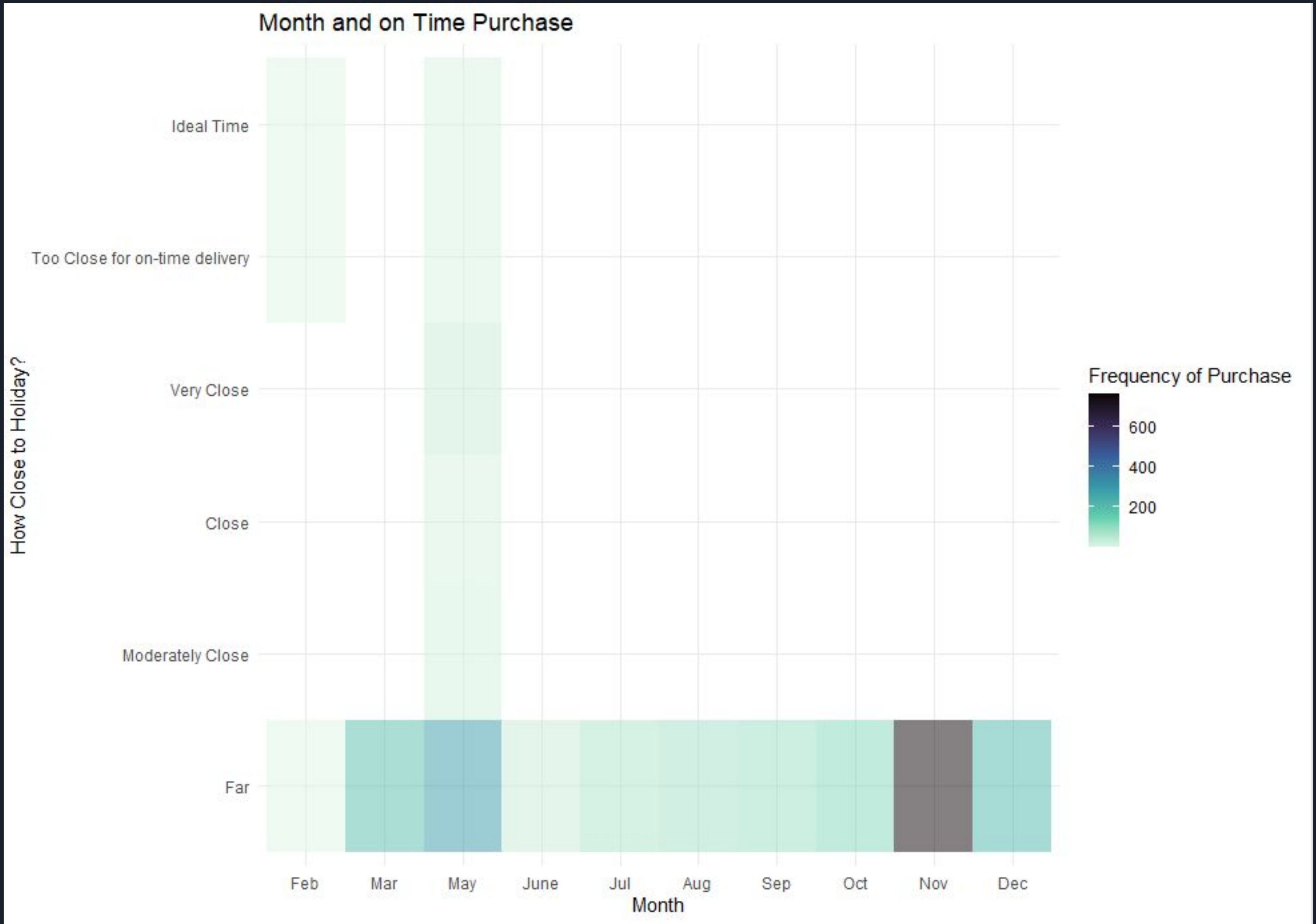
Purchase Made in Relation to Special Day
Frequency of Purchase
10000
7500
5000
2500
0
Far
Moderately Close
Close
Very Close
Too Close for on-time delivery
Ideal Time
Proximity to Special Day
Purchase
False
True

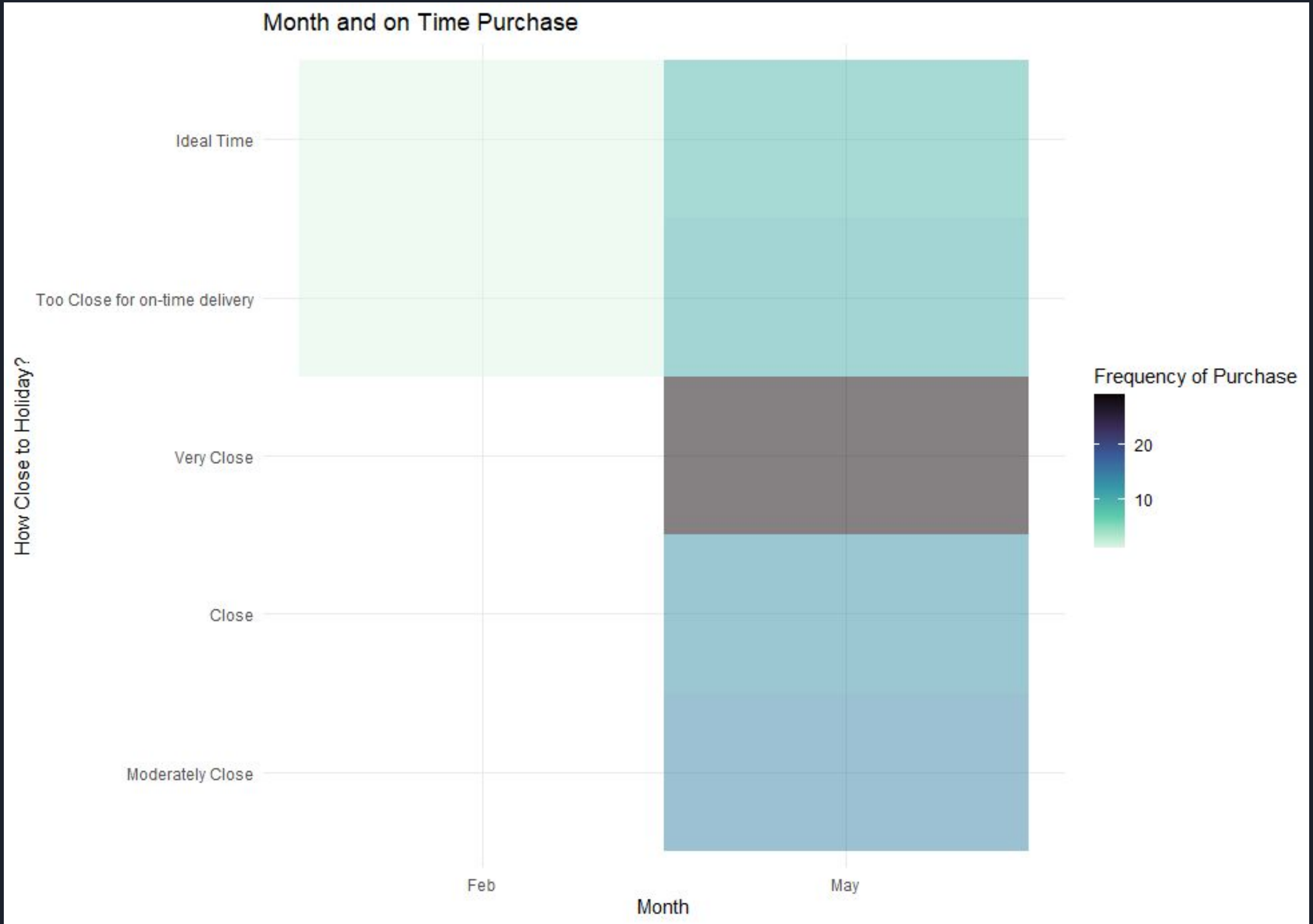


Purchase Made in Relation to Special Day
Frequency of Purchase
300
200
100
0
Moderately Close
Close
Very Close
Too Close for on-time delivery
Ideal Time
Proximity to Special Day
Purchase
False
True

Proportion Table for Special Day Category

	False	True
Far	0.82603325	0.17396675
Moderately Close	0.91764706	0.08235294
Close	0.94420601	0.05579399
Very Close	0.91369048	0.08630952
Too Close for on-time delivery	0.96507937	0.03492063
Ideal Time	0.93377483	0.06622517


Month and on Time Purchase
How Close to Holiday?
Ideal Time
Too Close for on-time delivery
Very Close
Close
Moderately Close
Far
Feb
Mar
May
June
Jul
Aug
Sep
Oct
Nov
Dec
Month
Frequency of Purchase
600
400
200


Month and on Time Purchase
Ideal Time
Too Close for on-time delivery
Very Close
Close
Moderately Close
How Close to Holiday?
Feb
May
Month
Frequency of Purchase
20
10

Data Analytics

Preliminary Logistic Regression w/ all Variables

```r
# Subset and clean up data
reg.data <- subset(df, select=c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18))
reg.data$Revenue <- as.logical(reg.data$Revenue)
reg.data$Revenue <- as.numeric(reg.data$Revenue)

# Create Train and Test data sets
trainIndex <- createDataPartition(reg.data$Revenue,
                                  p = .8,
                                  list = FALSE,
                                  times = 1)
train <- reg.data[trainIndex,]
test <- reg.data[-trainIndex,]

# Logistic Regression Model
model <- glm(Revenue~., family = binomial(link="logit"), data=train)
summary(model)
```

Administrative	1.123e-03	1.262e-02	0.089	0.92912	
Administrative_Duration	-1.317e-04	2.267e-04	-0.581	0.56139	
Informational	4.954e-02	2.992e-02	1.656	0.09781	.
Informational_Duration	-2.706e-06	2.444e-04	-0.011	0.99117	
ProductRelated	5.203e-04	1.401e-03	0.371	0.71040	
ProductRelated_Duration	9.653e-05	3.375e-05	2.860	0.00424	**
BounceRates	-4.901e-02	3.795e-02	-1.291	0.19655	
ExitRates	-1.557e-01	2.708e-02	-5.750	8.92e-09	***
PageValues	8.178e-02	2.747e-03	29.768	< 2e-16	***
SpecialDay	-1.136e-01	2.733e-01	-0.416	0.67776	
MonthDec	-5.457e-01	2.108e-01	-2.589	0.00963	**
MonthFeb	-1.555e+00	6.531e-01	-2.381	0.01729	*
MonthJul	1.444e-01	2.499e-01	0.578	0.56341	
MonthJune	-2.972e-01	3.091e-01	-0.961	0.33635	
MonthMar	-4.807e-01	2.082e-01	-2.309	0.02094	*
MonthMay	-5.839e-01	2.024e-01	-2.885	0.00392	**
MonthNov	6.115e-01	1.891e-01	3.234	0.00122	**
MonthOct	-1.491e-01	2.377e-01	-0.627	0.53060	
MonthSep	1.062e-01	2.400e-01	0.443	0.65810	
OperatingSystems	-1.117e-01	4.416e-02	-2.529	0.01144	*
Browser	4.260e-02	2.106e-02	2.023	0.04307	*
Region	-1.547e-02	1.487e-02	-1.040	0.29821	
TrafficType	1.010e-02	9.216e-03	1.096	0.27302	
VisitorTypeOther	-3.815e-01	5.985e-01	-0.637	0.52385	
VisitorTypeReturning_Visitor	-3.083e-01	9.728e-02	-3.170	0.00153	**
WeekendTRUE	8.571e-02	8.040e-02	1.066	0.28642	

- Most significant predictor variables:
  - ProductRelated_Duration
  - ExitRates
  - PageValues
  - Month
  - VisitorType

Final Logistic Regression Model

```r
# Set Seed
set.seed(123)

# Create Train and Test data sets
sample <- sample(c(TRUE, FALSE), nrow(df), replace=TRUE, prob=c(0.8,0.2))
train <- df[sample, ]
test <- df[!sample, ]

# Logistic Regression Model
model <- glm(Revenue~ProductRelated_Duration+ExitRates+PageValues+MonthFeb+MonthMar+Month
May+MonthNov+MonthDec+ReturningVisitor, family="binomial", data=train)
summary(model)
```

```
Call:
glm(formula = Revenue ~ ProductRelated_Duration + ExitRates + 
    PageValues + MonthFeb + MonthMar + MonthMay + MonthNov + 
    MonthDec + ReturningVisitor, family = "binomial", data = train)

Coefficients:
                        Estimate Std. Error z value             Pr(>|z|)    
(Intercept)             -1.80414    0.11567 -15.597 < 0.0000000000000002 ***
ProductRelated_Duration  0.20939    0.02988   7.009     0.0000000000024 ***
ExitRates               -0.88345    0.08812 -10.026 < 0.0000000000000002 ***
PageValues               1.51171    0.04961  30.471 < 0.0000000000000002 ***
MonthFeb                -1.63946    0.63078  -2.599             0.009347 ** 
MonthMar                -0.58883    0.13123  -4.487     0.0000072208784 ***
MonthMay                -0.52410    0.10907  -4.805     0.0000015459376 ***
MonthNov                 0.53690    0.09624   5.579     0.0000000242312 ***
MonthDec                -0.47760    0.12864  -3.713             0.000205 ***
ReturningVisitor        -0.30872    0.09330  -3.309             0.000937 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 8496.8  on 9808  degrees of freedom
Residual deviance: 5776.9  on 9799  degrees of freedom
AIC: 5796.9

Number of Fisher Scoring iterations: 7
```

- Response Variable:
  - Revenue
- Predictor Variables:
  - Product Related Duration
  - Exit Rates
  - Page Values
  - February, March, May, November, December
  - Returning Visitor
- Create a Train and Test using sample()
  - Train = 80%
  - Test = 20%

Evaluation of Model Part 1/2

```r
# Predicted results and Accuracy
predicted <- predict(model, test, type="response")
predicted_results <- ifelse(predicted > 0.28, 1, 0)

misClasificError <- mean(predicted_results != test$Revenue)
print(paste('Accuracy', 1-misClasificError))

# Calculation of pseudo R-Squared Value
# pscl comes from library(pscl)
pscl::pR2(model)["McFadden"]
```

```
[1] "Accuracy 0.894166666666667"
fitting null model for pseudo-r2
 McFadden
0.3201154
```

- Test the model using the Test data
- R^2 values apply differently to logistic regression
- McFadden Pseudo r-squared value for Logistic Regression
    - "DL McFadden stated that a pseudo-R^2 higher than 0.2 represents an *excellent* fit"
    - One-Off Coder (2024)

Evaluation of Model Part 2/2

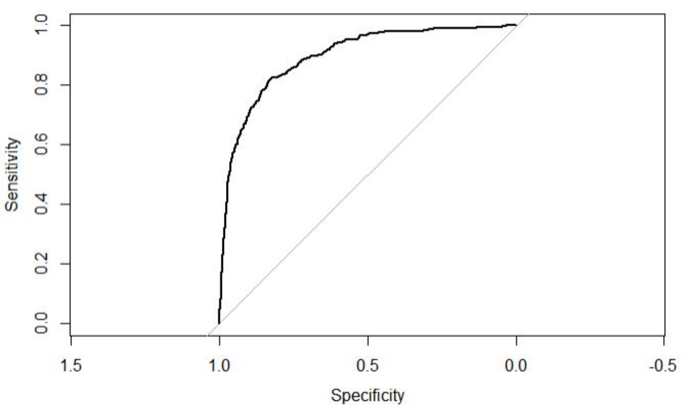
```r
# Check the predictor variables for multicollinearity
car::vif(model)

# Confusion Matrix to check false positive and false negatives
conf_matrix <- table(Predicted = predicted_results, Actual = test$Revenue)
cat(sprintf("Confusion Matrix:\n"))
conf_matrix
```

```
ProductRelated_Duration              ExitRates           PageValues
               1.119017               1.095266             1.048322
               MonthFeb               MonthMar             MonthMay
               1.013745               1.373842             1.622092
               MonthNov               MonthDec    ReturningVisitor
               1.836362               1.390703             1.112886
Confusion Matrix:
        Actual
Predicted FALSE TRUE
        0  1930  161
        1    93  216
```

- Check for Multicollinearity
- Check confusion matrix for false positives and false negatives
  - False Positives: 93
  - False Negatives: 161

ROC Curve

```
# Receiver Operating Characteristic Curve
# roc() comes from library(pROC)
roc_curve <- roc(test$Revenue, predicted)
plot(roc_curve)
auc_value <- auc(roc_curve)
print(auc_value)
```

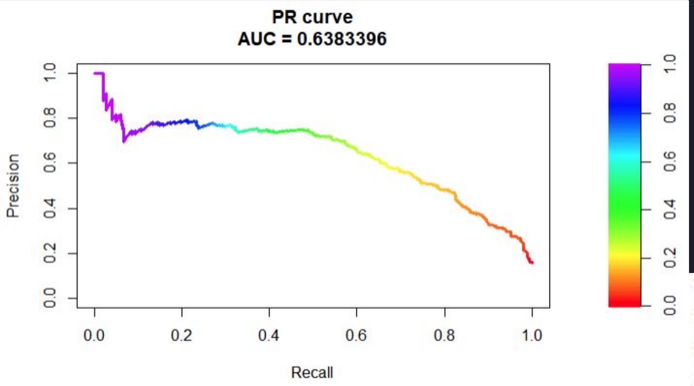



```
Setting levels: control = FALSE, case = TRUE
Setting direction: controls < cases
Area under the curve: 0.8945
```

- AUC score interpretation:
  - = 0.5 is akin to random guessing
  - 0.5-0.7 is slight strength
  - 0.7-0.8 is moderate strength
  - 0.8-0.9 is good strength
  - > 0.9 is great strength
- Evidently AI team (n.d.)

PR Curve

```
# Precision-Recall Curve
# pr.curve() comes from library(PRROC)
pr_curve <- pr.curve(scores.class0 = predicted[test$Revenue == 1],
                     scores.class1 = predicted[test$Revenue == 0],
                     curve = T)

plot(pr_curve)

precision <- conf_matrix[2, 2] / sum(conf_matrix[2, ])
recall <- conf_matrix[2, 2] / sum(conf_matrix[, 2])
print(paste("Precision Score:", precision))
print(paste("Recall Score:", recall))

f1_score <- 2 * ((precision * recall) / (precision + recall))
print(paste("F1 Score:", f1_score))
```




```
[1] "Precision Score: 0.699029126213592"
[1] "Recall Score: 0.572944297082228"
[1] "F1 Score: 0.629737609329446"
```

- F1 Score Interpretation:
  - < 0.5 is Not Good
  - 0.5 - 0.8 is Ok
  - 0.8 - 0.9 is Good
  - > 0.9 is Very Good
- Encord Blog (2023)

References

Encord Blog. (2023, July 18) *F1 Score in Machine Learning*. Encord. https://encord.com/blog/f1-score-in-machine-learning/#:~:text=The%20F1%20score%20ranges%20between%200%20and%201%2C,can%20concurrently%20attain%20high%20precision%20and%20high%20recall.

Evidently AI. (n.d.) *How to explain the ROC curve and ROC AUC score*. Evidently AI. https://www.evidentlyai.com/classification-metrics/explain-roc-curve

One-Off Coder. (2024, Apr. 03). *Pseudo r-squared for logistic regression*. Data Science Topics. https://datascience.oneoffcoder.com/psuedo-r-squared-logistic-regression.html

Shah, C. (2020). Hands-on Introduction to Data Science. Cambridge University Press. https://doi.org/10.1017/9781108560412