ITEC 621 Predictive Analytics Project

Project Name:
Predicting Consumer Behavior: Insights from E-Commerce Data

Class Section: 002
Team Number: 7
Team Members:
Marco Capoccia, Viraj Pande, Dheeraj Shetty, Victoria Smith

Last updated: May 1, 2025
Deliverable Number: 4

1. Business Case

As online retail expands, companies face ongoing challenges of attracting website visitors and ensuring they complete a purchase. One of the largest challenges is determining whether the investment in acquiring and retaining customers is profitable, especially given that average e-commerce conversion rates are only around 2.63% (Smith, 2024). That being said, it is critical to understand key behavioral drivers in online shopping for companies to grow revenue and ROI. Although there are platforms available to provide behavioral metrics, it is challenging for most businesses to turn this data into actionable insights to help support strategic business decision-making.

This challenge is also magnified by the rising cost of acquiring new customers. According to research, customer acquisition costs (CAC) have increased by over 220% in the past eight years, starting from an average of $9 per customer (2013) to around $29 (2022) (O'Rourke, 2022). In a competitive environment like this, even minor improvements in conversion rates can result in substantial revenue gains and more efficient marketing spend.

This project aims to fill the gap by analyzing data from over 12,000 buying sessions, each showing whether a shopper made a purchase. By examining variables like time spent on the product page, type of visit, and time of visit (weekend vs. weekday), we aim to discover which variables are most useful in predicting purchase behavior. We will use statistical models to predict the likelihood of a purchase and identify which behavior traits matter most. The results of this analysis can help e-commerce retailers better understand user shopping patterns and increase conversion rates and profits.

2. Business and Analytics Questions

2.1. ***Business Question:*** How can an online retailer increase sales by identifying key behavioral factors that drive shoppers to complete a purchase?

2.2. ***Analytics Question:*** What is the impact of session behavior (e.g., time spent on product pages, bounce rates, exit rates) and visitor traits (e.g., returning customers, weekend visits) on the likelihood of a purchase? Additionally, how does the month of visit influence purchase probability?

Analytics Goal: Our goal is to identify factors that predict online purchases and provide actionable insights for e-commerce optimization. Using classification models like Logistic Regression and Random Forests, we aim to balance predictive accuracy with interpretability for stakeholders.

3. Data Set Description

Our analysis uses the "Online Shoppers Intention" dataset, comprising 12,330 buying sessions from an e-commerce platform, each indicating whether a purchase occurred. The dataset is sourced from the UCI Machine Learning Repository. The main variables of interest are:

Outcome Variable: Revenue (binary: True/False) – Indicates whether a purchase was made. The dataset is unbalanced, with 10,422 non-purchase sessions (False) and 1,908 purchase sessions (True).
Predictors:
ProductRelated_Duration (numeric, seconds) – Time spent on product pages.
BounceRates (numeric, proportion) – Proportion of single-page visits.
ExitRates (numeric, proportion) – Proportion of sessions ending on a page.
VisitorType (categorical: Returning/New/Other) – Visitor status.
Weekend (binary: True/False) – Whether the visit occurred on a weekend.
Month (categorical: e.g., Feb, Mar) – Month of the visit.
These variables were chosen for their relevance to online shopping behavior and potential to influence purchase decisions.

4. Descriptive Analytics

4.1. Descriptive Statistics of Focal Variables

To understand the baseline characteristics of our predictors, we computed means and standard deviations for numeric variables and frequency distributions for categorical variables. For instance, shoppers who made a purchase spent significantly more time on product pages (mean = 1,876 seconds) compared to non-buyers (mean = 1,070 seconds), suggesting engagement as a potential driver of purchases. Similarly, bounce rates and exit rates were lower for buyers (means = 0.005 and 0.020, respectively) than non-buyers (means = 0.025 and 0.047), indicating that less site abandonment correlates with purchases.

4.2. Distribution of Outcome and Other Variables

The outcome variable, Revenue, is binary (True/False), representing whether a purchase occurred. A bar plot shows an unbalanced distribution: 84.5% of sessions resulted in no purchase (10,422), while 15.5% led to a purchase (1,908). This imbalance suggests that predictive models may need adjustments to account for the disproportionate classes.

4.3. Correlation and Co-Variation Analysis

To assess relationships among predictors and with the outcome, we analyzed correlations and conducted statistical tests. Numeric predictors showed a strong positive correlation between BounceRates and ExitRates ($r = 0.91$), suggesting a potential overlap in what they measure, which could complicate modeling. ProductRelated_Duration had weak correlations with other variables, indicating it may offer unique predictive power. Tests of association revealed significant relationships between Revenue and VisitorType, Weekend, and Month (all p-values < 0.002), confirming their relevance as predictors. Additionally, ProductRelated_Duration varied significantly between buyers and non-buyers ($p < 0.001$), reinforcing its importance.

4.4. Data Pre-Processing

Based on our descriptive analytics, we examined several preprocessing steps and transformations to prepare the data for predictive modeling. To assess potential multicollinearity, we inspected the correlation matrix of numeric predictors and found a strong positive correlation between BounceRates and ExitRates (r = 0.91,). This high correlation suggests these variables may measure overlapping aspects of session abandonment, which could destabilize our regression models by inflating variance. To address this, we plan to either select one variable (e.g., ExitRates, due to its broader scope) or create a composite metric, ensuring cleaner model interpretation.

We also analyzed the distribution of the outcome variable, Revenue, to understand its balance and aid in designing our models. The bar plot revealed a significant imbalance, with 84.5% non-purchases (10,422 sessions) and 15.5% purchases (1,908 sessions). This skewness indicates that a standard model might overpredict non-purchases, so we are considering oversampling the minority class (purchases) or adjusting the classification threshold to improve sensitivity to the positive class, which is critical for our goal of identifying purchase drivers.

Finally, to evaluate predictor distributions for modeling assumptions, we inspected the summary statistics and found that ProductRelated_Duration exhibits high variability (SD = 1,913.67 seconds) and a skewed distribution (e.g., max = 63,973.52 vs. median = 598.94). This analysis suggests that a log transformation could normalize this variable, reducing the influence of extreme values and improving the linearity assumption in our Logistic Regression model. These steps—addressing multicollinearity, class imbalance, and skewness—stem from our descriptive findings and are designed to enhance the robustness and interpretability of our predictive models.

5. Modeling Methods and Model Specifications

5.1. Initial Model Specification

Our initial model predicts Revenue (binary) using ProductRelated_Duration, BounceRates, ExitRates, VisitorType, Weekend, and Month. These predictors reflect key aspects of online shopping: time on product pages measures engagement, bounce and exit rates capture abandonment, visitor type indicates loyalty, weekend visits reflect browsing context, and month accounts for seasonal effects. Based on its skewness, we are considering a log transformation for ProductRelated_Duration.

5.2. Initial Logistic Regression Model Results

Given the binary nature of Revenue, we fitted a preliminary Logistic Regression model with the specified predictors. The results show that ProductRelated_Duration, ExitRates, and VisitorType significantly influence purchase likelihood ($p < 0.001$), aligning with our descriptive analysis. Specifically, longer time on product pages and lower exit rates increase the odds of a purchase, while returning visitors are less likely to buy compared to new visitors. Certain months also matter: November boosts purchases ($p < 0.001$), while February, March, and May reduce them ($p < 0.05$). BounceRates and Weekend showed no significant effects ($p > 0.4$), suggesting limited independent impact after accounting for other predictors.

5.3. Assumption Tests

To assess multicollinearity among predictors in our logistic regression model, we calculated Variance Inflation Factors (VIF). All GVIF-adjusted values were well below the common threshold of 5, indicating no significant multicollinearity. Specifically, BounceRates and ExitRates had VIF values of 1.75 and 1.79, respectively, suggesting moderate correlation but not at a level that compromises model stability. All other predictors, including ProductRelated_Duration, VisitorType, Weekend, and Month, had GVIF values near 1.
As such, we conclude that multicollinearity is not a concern in our logistic regression model, and all predictors were retained for interpretability and completeness.

5.4. Model Candidates and Rationale

The first model is a standard logistic regression, which is appropriate for the binary outcome variable (Revenue). Outside of this control model, we evaluated two alternative modeling methods to address challenges that came up during our descriptive analysis.

For the second model, we selected LASSO Regression to maximize model simplicity and reduce potential redundancy among the predictors. Although our tests for multicollinearity (i.e., VIF) did not indicate severe multicollinearity, we did observe a high correlation ($r=0.91$) between BounceRates and ExitRates. LASSO applies a penalty that can shrink small coefficients to zero, effectively removing less informative predictors. This helps to identify which session activities and user attributes are most important for predicting purchase behavior.

The last model we chose is an Ensemble Tree-based approach that covers Bagging and Random Forest. These methods are useful for our binary outcome, handle complex interactions, and non-linearities without prior transformations.

5.5. Model Specification Candidates and Rationale

The first candidate model selected was a logistic regression using the full set of predictors related to the business case (ProductRelated_Duration, BounceRates, ExitRates, VisitorType, and Weekend). This model reflects our business understanding that the probability of a purchase being made is based on session behavior (e.g., time spent on product pages, bounce rates, exit rates) and visitor attributes (e.g., returning customers, time of visits). The second specification for the logistic regression uses the same set of predictors but fits the model in a balanced version of the dataset, where the number of purchase sessions and non-purchase sessions was equalized through random oversampling of the minority class (sessions which resulted in a purchase). Oversampling reduces the extreme skew of data (15.5% purchases) without eliminating any non-purchase data. This approach allows fairer evaluation of purchase drivers without the loss of useful behavior data.

The first LASSO model was fitted on the original, unbalanced dataset (Shopping), which had 15.5% purchase sessions. We used 10-fold cross-validation to identify the optimal regularization strength. The model selected a best lambda of 0.00041, which determines the amount of shrinkage applied to the coefficients. The coefficient path plot shows how coefficients shrink as the penalty increases, while the cross-validation plot illustrates how model deviance changes with varying levels of regularization. To better evaluate purchase behavior without bias toward the majority class, we created a second LASSO model using the balanced version of the dataset (Shopping.bal), where the number of purchase and non-purchase sessions was equalized through oversampling. Cross-validation identified a best lambda of 0.00094 for this model. Coefficient path and CV error plots for this specification are shown in and, respectively.

For the first specification of the ensemble tree methods, we used Bagging with unbalanced data to obtain optimal prediction accuracy concerning the true class ratios. This process combines results from multiple trees that are built on bootstrap samples to limit variance and avoid overfitting. For the second specification, we used the balanced dataset to determine if equal class representation improves the prediction performance of the minority class. For the second specification, we used Random Forest with both datasets (unbalanced and balanced) with a low mtry value to introduce feature-level randomness to enable variance importance ranking.

5.6 Cross-Validation Testing and Final Model Selection

We evaluated all candidate models and specifications using 10-fold cross-validation (10FCV) to assess the model's predictive accuracy and select the most effective model. Each model was tested under two specifications: Spec 1 using the original, unbalanced dataset (15.5% purchase sessions), and Spec 2 using a balanced version created through oversampling. The results highlight Bagging with the balanced dataset as the best-performing model, achieving the lowest 10FCV error of 0.06592. This model outperformed all others, including Random Forest (Balanced) with a 10FCV error of 0.23570 and Logit Spec 1 (Unbalanced) at 0.11820. While logistic regression models offer interpretability, they were outperformed in predictive power by ensemble methods. LASSO, though useful for feature selection, produced higher CV errors in both specifications. Given our goal of maximizing predictive accuracy, we recommend the Bagging model trained on the balanced data as the final model. It delivers the strongest generalization performance and is best suited for deployment in a prediction-focused context.

The final two models were tree-based classifiers: Bagging and Random Forest. Both models were estimated using the same six predictors used in the regression-based models: ProductRelated_Duration, BounceRates, ExitRates, VisitorType, Weekend, and Month. For each approach, we fit a model on both the unbalanced dataset (Shopping) and the balanced version (Shopping.bal). The bagging model used all predictors at each split (mtry = 6), while the random forest model randomly selects a subset (mtry = 3), introducing additional variation between trees. These models were chosen because they handle nonlinear relationships and interactions between predictors without requiring explicit specification. We relied on the built-in classification error provided by the randomForest function, as it uses internal resampling to estimate performance. Comparing the balanced and unbalanced specifications allows us to assess the impact of class imbalance on prediction accuracy and the ability of tree-based methods to capture complex patterns in customer purchase behavior.

Final Method/Specification Selected:
While the bagging model with balanced data had the lowest 10-fold cross-validation error (0.0659), we prioritized interpretability to best serve our audience of e-commerce stakeholders.

Consequently, we selected the logistic regression model with the unbalanced specification, which produced a strong 10FCV error of 0.1182, making it the best-performing parametric model.

We fit this model on the entire unbalanced dataset using glm(), maintaining all core predictors from our initial specification. Although BounceRates was statistically insignificant and highly collinear with ExitRates, we retained it for interpretability and comparison across models. Its presence does not negatively impact the model's fit or generalization error. This final model provides a clear, interpretable view of the key behavioral factors driving online purchases and supports strategic business decision-making for conversion optimization.

6. Analysis of Results

The logistic regression model (unbalanced specification) was ultimately selected for our final analysis. Although the balanced bagging model had the lowest 10-FCV error overall (0.0659), we prioritized balancing interpretability and predictive accuracy for business stakeholders. This logistic model yielded a strong 10-fold cross-validation error of 0.1182, and the output displayed all core predictors as statistically significant, except BounceRates, which we retained for comparison. The model was fit using glm() on the full unbalanced dataset.

Quantitative predictors: All interpretations are "on average and holding everything else constant":

A one-second increase in ProductRelated_Duration increases the odds of purchase by 0.0085% (coefficient = 0.000085, $p < 0.001$). This confirms that shoppers who spend more time engaging with product pages are more likely to convert.
A one-unit increase in ExitRates decreases the odds of purchase by nearly 100% (coefficient = -31.72, $p < 0.001$). This is the strongest negative predictor in the model and highlights that high exit behavior significantly reduces conversion probability.

Binary predictors: All interpretations are "on average and holding everything else constant":

Returning Visitors have 30% lower odds of making a purchase compared to new visitors (coefficient = -0.353, $p < 0.001$). This suggests that returning users, despite their familiarity, are less likely to convert during a session.
VisitorTypeOther had a positive effect (coefficient = 0.295), but it was not statistically significant ($p = 0.342$).
Weekend visits slightly increase the odds of purchase by 4.7% (coefficient = 0.046), but this effect is also not statistically significant ($p = 0.443$).

Categorical predictors (Month): Relative to January (baseline month):

November significantly increases the odds of purchase by 60% (coefficient = 0.469, $p < 0.001$), aligning with expected seasonal shopping behavior.
February, March, and May all significantly decrease the odds of purchase:
February: -87% (coefficient = -2.034, $p = 0.0007$)

March: -42% (coefficient = -0.552, p = 0.0003)
May: -30% (coefficient = -0.359, p = 0.0125)

Other months, including July, June, October, and September, did not show significant differences in purchase likelihood relative to January.

7. Conclusions and Lessons Learned

7.1. Conclusions from the Analysis

The central business question in this project was: *Can we predict which online shopping sessions are likely to result in a purchase based on session behavior and visitor characteristics?* All models identified ProductRelated_Duration and ExitRates as the most consistent and influential predictors. Sessions with longer product interaction time and lower exit rates were more likely to lead to purchases. BounceRates also contributed to predictive performance, but to a lesser extent.

We developed and evaluated four classification models: logistic regression, LASSO regression, Bagging, and Random Forest. The initial logistic regression model (unbalanced) yielded a 10-fold cross-validation (10FCV) error rate of 11.8%, while the balanced version increased slightly to 20.2%. LASSO performed worse, with 10FCV error rates of 75.7% (unbalanced) and 116.9% (balanced), indicating it was not well-suited to this problem given the predictor structure and data distribution.

Tree-based models outperformed both regression approaches. Bagging achieved a 10FCV error of 17.6% on the unbalanced data and improved to 6.6% on the balanced data. Random Forest produced a 10FCV error of 15.5% unbalanced and 23.6% balanced. These models also ranked variable importance, reaffirming that product engagement metrics were far more predictive of purchases than categorical variables like VisitorType or Month. Unlike regression models, which assume linear relationships and require manual interaction terms, tree models handle nonlinearity and interaction effects automatically, resulting in stronger predictive performance.

In sum, if the goal is to understand general drivers of purchase behavior, logistic regression offers a solid balance of interpretability and performance. However, if the goal is to accurately classify purchase sessions for operational targeting, Bagging on the balanced dataset delivered the lowest overall error and the most reliable performance.

Based on our results, we recommend that online retailers target user engagement through optimization of high-exit pages and schedule targeted ad campaigns for repeat buyers. Seasonal campaigns, especially around November (the holiday season), need to be given priority to take advantage of higher conversion potential. Implementing immediate, automated actions during the duration of user sessions can also reduce exit rates and increase overall purchases made.

7.2. Project Issues, Challenges and Lessons Learned

A primary challenge in this project was the significant class imbalance, with only 15.5% of sessions resulting in a purchase. Models trained on the original dataset tended to favor the majority class, reducing their ability to correctly identify purchasing sessions. This issue was addressed through oversampling, which allowed for more balanced learning and substantially improved classification performance across models.

In addition to outcome imbalance, we encountered limitations within the predictor set. BounceRates and ExitRates were found to be highly correlated, suggesting potential redundancy. While this had minimal impact on tree-based models, it may have introduced multicollinearity concerns in regression-based models, affecting coefficient interpretation and stability. Furthermore, ProductRelated_Duration exhibited a highly skewed distribution with several extreme values. These outliers may have influenced model training and contributed to differences in performance across methods.

The relative performance of the models varied by technique. LASSO facilitated variable selection but underperformed in terms of predictive accuracy. Logistic regression offered interpretability and performed competitively on the unbalanced data. Bagging and Random Forest achieved the highest predictive accuracy, particularly when trained on the balanced dataset. Additionally, error estimates obtained through 10-fold cross-validation were consistent with internal validation metrics for ensemble models, reinforcing the reliability of our evaluation approach.

This project emphasized the importance of addressing outcome imbalance, carefully assessing predictor redundancy and distribution, and selecting modeling approaches that align with the objectives of either explanation or prediction.

Appendix Contents

Table of contents

Load Dataset

```
Shopping <- read.table("online_shoppers_intention.csv", header = TRUE, sep =
",")
```

4.1 Descriptive statistics of key variables

```
library(dplyr)

# Convert categorical variables to factors
Shopping$Month <- as.factor(Shopping$Month)
Shopping$VisitorType <- as.factor(Shopping$VisitorType)
Shopping$OperatingSystems <- as.factor(Shopping$OperatingSystems)
Shopping$Browser <- as.factor(Shopping$Browser)
Shopping$Region <- as.factor(Shopping$Region)
Shopping$TrafficType <- as.factor(Shopping$TrafficType)

head(Shopping)

  Administrative Administrative_Duration Informational Informational_Duration
1              0                       0             0                      0
2              0                       0             0                      0
```

```
3             0                        0             0                      0
4             0                        0             0                      0
5             0                        0             0                      0
6             0                        0             0                      0
  ProductRelated ProductRelated_Duration BounceRates ExitRates PageValues
1              1                0.000000  0.20000000 0.2000000          0
2              2               64.000000  0.00000000 0.1000000          0
3              1                0.000000  0.20000000 0.2000000          0
4              2                2.666667  0.05000000 0.1400000          0
5             10              627.500000  0.02000000 0.0500000          0
6             19              154.216667  0.01578947 0.0245614          0
  SpecialDay Month OperatingSystems Browser Region TrafficType
1          0   Feb                1       1      1           1
2          0   Feb                2       2      1           2
3          0   Feb                4       1      9           3
4          0   Feb                3       2      2           4
5          0   Feb                3       3      1           4
6          0   Feb                2       2      1           3
        VisitorType Weekend Revenue
1 Returning_Visitor   FALSE   FALSE
2 Returning_Visitor   FALSE   FALSE
3 Returning_Visitor   FALSE   FALSE
4 Returning_Visitor   FALSE   FALSE
5 Returning_Visitor    TRUE   FALSE
6 Returning_Visitor   FALSE   FALSE
```

```
summary(Shopping)
```

```
 Administrative   Administrative_Duration Informational    
 Min.   : 0.000   Min.   :   0.00         Min.   : 0.0000  
 1st Qu.: 0.000   1st Qu.:   0.00         1st Qu.: 0.0000  
 Median : 1.000   Median :   7.50         Median : 0.0000  
 Mean   : 2.315   Mean   :  80.82         Mean   : 0.5036  
 3rd Qu.: 4.000   3rd Qu.:  93.26         3rd Qu.: 0.0000  
 Max.   :27.000   Max.   :3398.75         Max.   :24.0000  

 Informational_Duration ProductRelated   ProductRelated_Duration
 Min.   :   0.00        Min.   :  0.00   Min.   :    0.0        
 1st Qu.:   0.00        1st Qu.:  7.00   1st Qu.:  184.1        
 Median :   0.00        Median : 18.00   Median :  598.9        
 Mean   :  34.47        Mean   : 31.73   Mean   : 1194.8        
 3rd Qu.:   0.00        3rd Qu.: 38.00   3rd Qu.: 1464.2        
 Max.   :2549.38        Max.   :705.00   Max.   :63973.5        

  BounceRates         ExitRates         PageValues        SpecialDay     
 Min.   :0.000000   Min.   :0.00000   Min.   :  0.000   Min.   :0.00000  
 1st Qu.:0.000000   1st Qu.:0.01429   1st Qu.:  0.000   1st Qu.:0.00000  
 Median :0.003112   Median :0.02516   Median :  0.000   Median :0.00000  
 Mean   :0.022191   Mean   :0.04307   Mean   :  5.889   Mean   :0.06143  
 3rd Qu.:0.016813   3rd Qu.:0.05000   3rd Qu.:  0.000   3rd Qu.:0.00000  
```

```
Max.   :0.200000   Max.   :0.20000   Max.   :361.764   Max.   :1.00000   

     Month      OperatingSystems    Browser          Region      TrafficType  
 May    :3364   2      :6601     2      :7961   1      :4780   2      :3913  
 Nov    :2998   1      :2585     1      :2462   3      :2403   1      :2451  
 Mar    :1907   3      :2555     4      : 736   4      :1182   3      :2052  
 Dec    :1727   4      : 478     5      : 467   2      :1136   4      :1069  
 Oct    : 549   8      :  79     6      : 174   6      : 805   13     : 738  
 Sep    : 448   6      :  19     10     : 163   7      : 761   10     : 450  
 (Other):1337   (Other):  13     (Other): 367   (Other):1263   (Other):1657  
           VisitorType      Weekend          Revenue       
 New_Visitor      : 1694   Mode :logical   Mode :logical  
 Other            :   85   FALSE:9462      FALSE:10422    
 Returning_Visitor:10551   TRUE :2868      TRUE :1908     
```

Mean and Standard Dev. for Numeric Varaibles

```
library(psych)
describe(Shopping)

                        vars     n    mean      sd median trimmed    mad min
Administrative             1 12330    2.32    3.32   1.00    1.63   1.48   0
Administrative_Duration    2 12330   80.82  176.78   7.50   42.10  11.12   0
Informational              3 12330    0.50    1.27   0.00    0.18   0.00   0
Informational_Duration     4 12330   34.47  140.75   0.00    3.59   0.00   0
ProductRelated             5 12330   31.73   44.48  18.00   22.75  19.27   0
ProductRelated_Duration    6 12330 1194.75 1913.67 598.94  820.08 742.69   0
BounceRates                7 12330    0.02    0.05   0.00    0.01   0.00   0
ExitRates                  8 12330    0.04    0.05   0.03    0.03   0.02   0
PageValues                 9 12330    5.89   18.57   0.00    1.29   0.00   0
SpecialDay                10 12330    0.06    0.20   0.00    0.00   0.00   0
Month*                    11 12330    6.16    2.37   7.00    6.35   1.48   1
OperatingSystems*         12 12330    2.12    0.91   2.00    2.06   0.00   1
Browser*                  13 12330    2.36    1.72   2.00    2.00   0.00   1
Region*                   14 12330    3.15    2.40   3.00    2.79   2.97   1
TrafficType*              15 12330    4.07    4.03   2.00    3.21   1.48   1
VisitorType*              16 12330    2.72    0.69   3.00    2.90   0.00   1
Weekend                   17 12330     NaN      NA     NA     NaN     NA Inf
Revenue                   18 12330     NaN      NA     NA     NaN     NA Inf
                             max    range  skew kurtosis    se
Administrative             27.00    27.00  1.96     4.70  0.03
Administrative_Duration  3398.75  3398.75  5.61    50.53  1.59
Informational              24.00    24.00  4.04    26.92  0.01
Informational_Duration   2549.38  2549.38  7.58    76.27  1.27
ProductRelated            705.00   705.00  4.34    31.19  0.40
ProductRelated_Duration 63973.52 63973.52  7.26   137.10 17.23
```

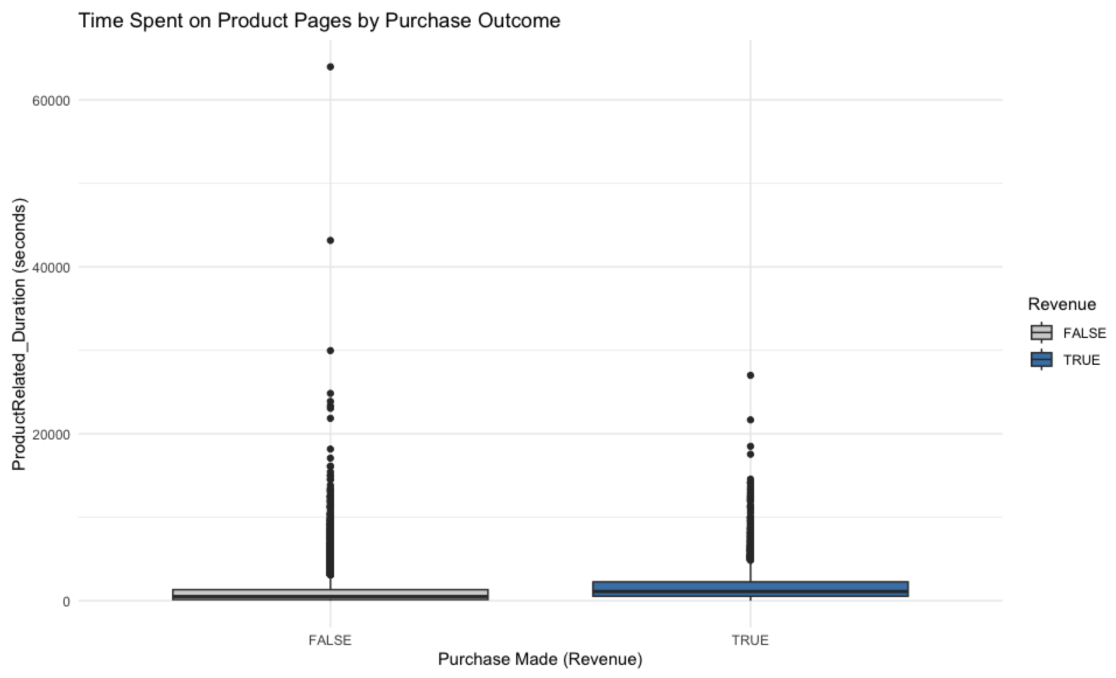
```
BounceRates                  0.20      0.20  2.95      7.72  0.00
ExitRates                    0.20      0.20  2.15      4.01  0.00
PageValues                 361.76    361.76  6.38     65.60  0.17
SpecialDay                   1.00      1.00  3.30      9.91  0.00
Month*                      10.00      9.00 -0.83     -0.37  0.02
OperatingSystems*            8.00      7.00  2.07     10.45  0.01
Browser*                    13.00     12.00  3.24     12.74  0.02
Region*                      9.00      8.00  0.98     -0.15  0.02
TrafficType*                20.00     19.00  1.96      3.48  0.04
VisitorType*                 3.00      2.00 -2.06      2.29  0.01
Weekend                      -Inf      -Inf    NA        NA    NA
Revenue                      -Inf      -Inf    NA        NA    NA

# Mean Product Duration by Revenue
tapply(Shopping$ProductRelated_Duration, Shopping$Revenue, mean, na.rm =
TRUE)

   FALSE     TRUE
1069.988 1876.210

tapply(Shopping$ProductRelated_Duration, Shopping$Revenue, sd, na.rm = TRUE)

   FALSE     TRUE
1803.798 2312.214

# Bounce Rates by Revenue
tapply(Shopping$BounceRates, Shopping$Revenue, mean, na.rm = TRUE)

      FALSE        TRUE
0.025317232 0.005117153

tapply(Shopping$BounceRates, Shopping$Revenue, sd, na.rm = TRUE)

     FALSE       TRUE
0.05187746 0.01218451

# Exit Rates by Revenue
tapply(Shopping$ExitRates, Shopping$Revenue, mean, na.rm = TRUE)

     FALSE       TRUE
0.04737827 0.01955517

# Page Values by Revenue
tapply(Shopping$PageValues, Shopping$Revenue, mean, na.rm = TRUE)

    FALSE      TRUE
 1.975998 27.264518

# Special Day Score by Revenue
tapply(Shopping$SpecialDay, Shopping$Revenue, mean, na.rm = TRUE)

     FALSE       TRUE
0.06843216 0.02316562
```

Categorical Variables Frequency

```
 #Visitor Type
table(Shopping$VisitorType)


      New_Visitor             Other Returning_Visitor
             1694                85             10551

table(Shopping$VisitorType, Shopping$Revenue)


                     FALSE  TRUE
  New_Visitor         1272   422
  Other                 69    16
  Returning_Visitor   9081  1470

# Weekend visits
table(Shopping$Weekend)


FALSE   TRUE
 9462   2868

table(Shopping$Weekend, Shopping$Revenue)


        FALSE TRUE
  FALSE  8053 1409
  TRUE   2369  499

# Month of visit
table(Shopping$Month)


 Aug  Dec  Feb  Jul June  Mar  May  Nov  Oct  Sep
 433 1727  184  432  288 1907 3364 2998  549  448

table(Shopping$Month, Shopping$Revenue)


       FALSE TRUE
  Aug    357   76
  Dec   1511  216
  Feb    181    3
  Jul    366   66
  June   259   29
  Mar   1715  192
  May   2999  365
  Nov   2238  760
  Oct    434  115
  Sep    362   86
```

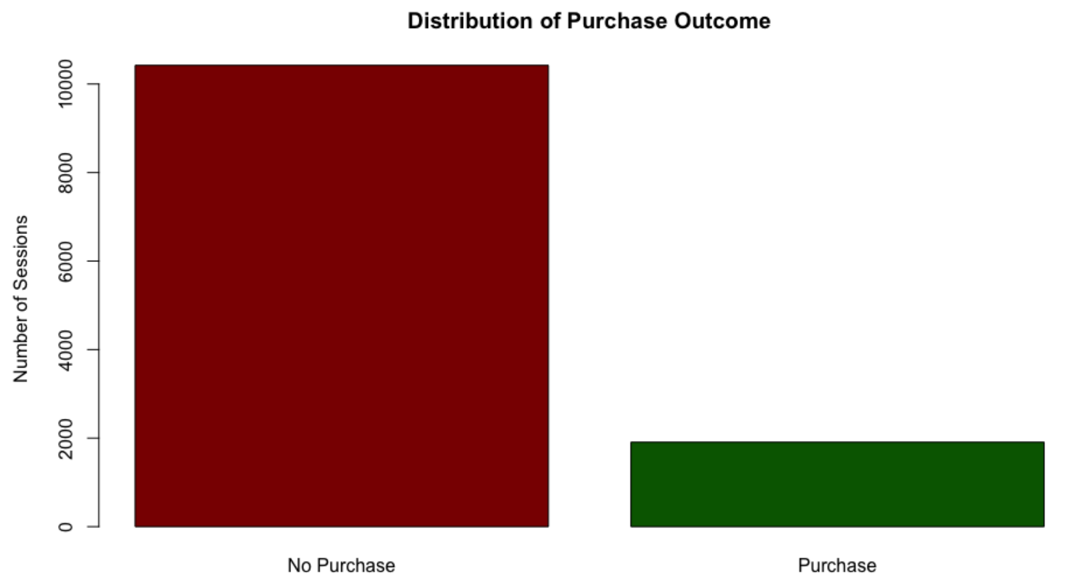
Boxplot for Time Spent on Product Pages by Purchase Outcome

```
library(ggplot2)

# Boxplot of time spent on product pages by purchase outcome
ggplot(Shopping, aes(x = Revenue, y = ProductRelated_Duration, fill =
Revenue)) +
  geom_boxplot() +
  labs(title = "Time Spent on Product Pages by Purchase Outcome",
       x = "Purchase Made (Revenue)",
       y = "ProductRelated_Duration (seconds)") +
  scale_fill_manual(values = c("FALSE" = "lightgray", "TRUE" = "steelblue"))
+
  theme_minimal()
```



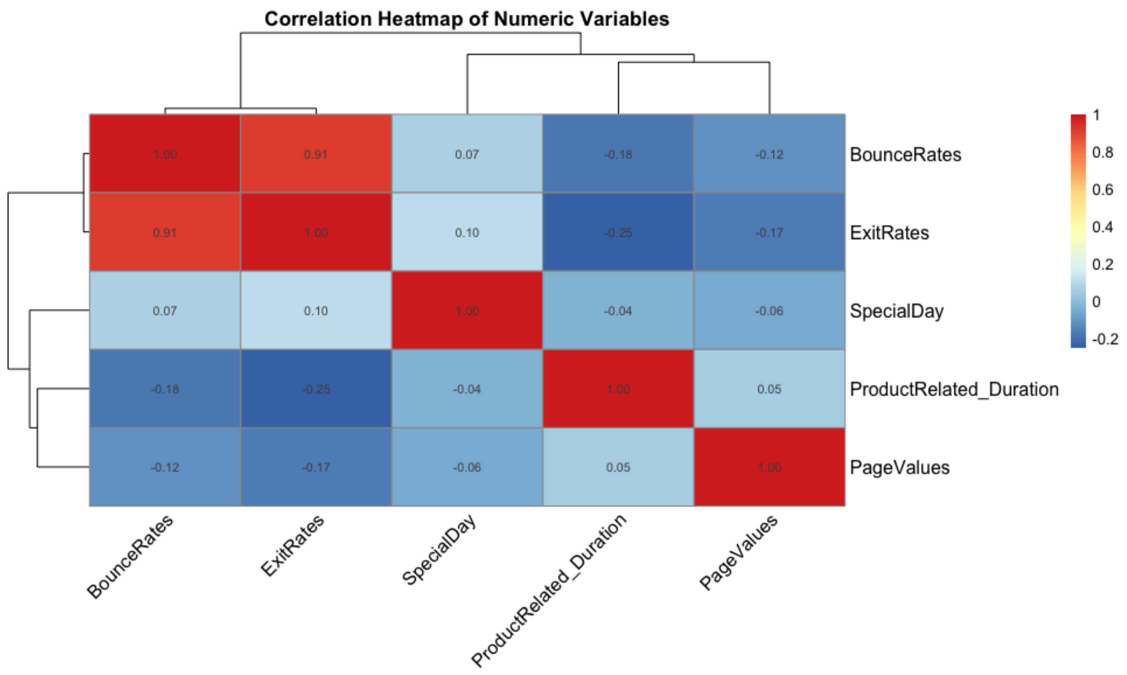
4.2 Distribution of the outcome variable (e.g., normal, skewed, Poisson, binary, etc.-BINARY

```
barplot(table(Shopping$Revenue),
        main = "Distribution of Purchase Outcome",
        ylab = "Number of Sessions",
        names.arg = c("No Purchase", "Purchase"),
col = c("darkred", "darkgreen"))
```



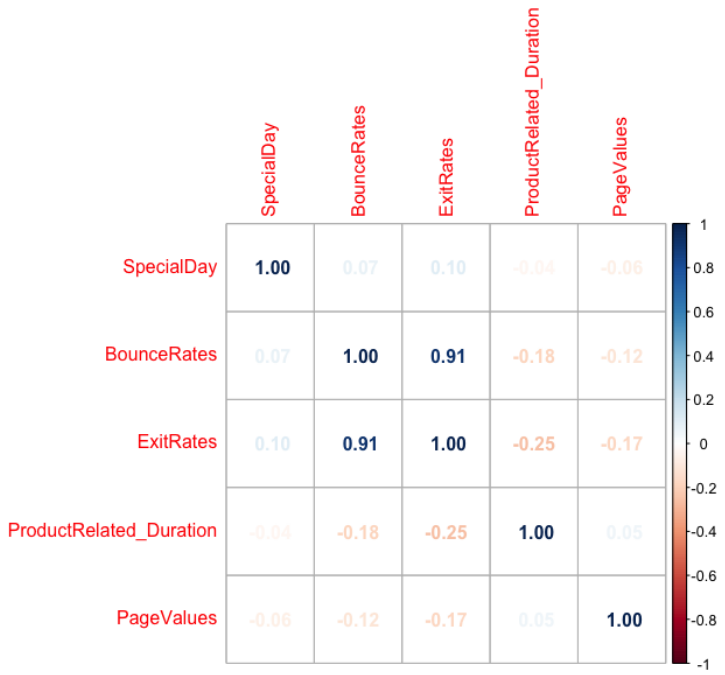
Correlation Heatmap of Numeric Variables

```
#Previous Code for simple heatmap

numeric_vars <- Shopping[, c("ProductRelated_Duration", "BounceRates",
"ExitRates", "PageValues", "SpecialDay")]


# Correlation matrix
cor_matrix <- cor(numeric_vars, use = "complete.obs")
print(cor_matrix)

                        ProductRelated_Duration BounceRates  ExitRates
ProductRelated_Duration              1.00000000 -0.18454112 -0.2519841
BounceRates                         -0.18454112  1.00000000  0.9130044
ExitRates                           -0.25198410  0.91300440  1.0000000
PageValues                           0.05282306 -0.11938603 -0.1744983
SpecialDay                          -0.03637985  0.07270225  0.1022418
                         PageValues  SpecialDay
ProductRelated_Duration  0.05282306 -0.03637985
BounceRates             -0.11938603  0.07270225
ExitRates               -0.17449831  0.10224180
PageValues               1.00000000 -0.06354127
SpecialDay              -0.06354127  1.00000000

library(pheatmap)

# Simple heatmap
pheatmap(cor_matrix,
         main = "Correlation Heatmap of Numeric Variables",
         fontsize_row = 12,      # Increase row label size
         fontsize_col = 12,      # Increase column label size
         display_numbers = TRUE, # Optional: show correlation values
         clustering_method = "complete",
         angle_col = 45)
```

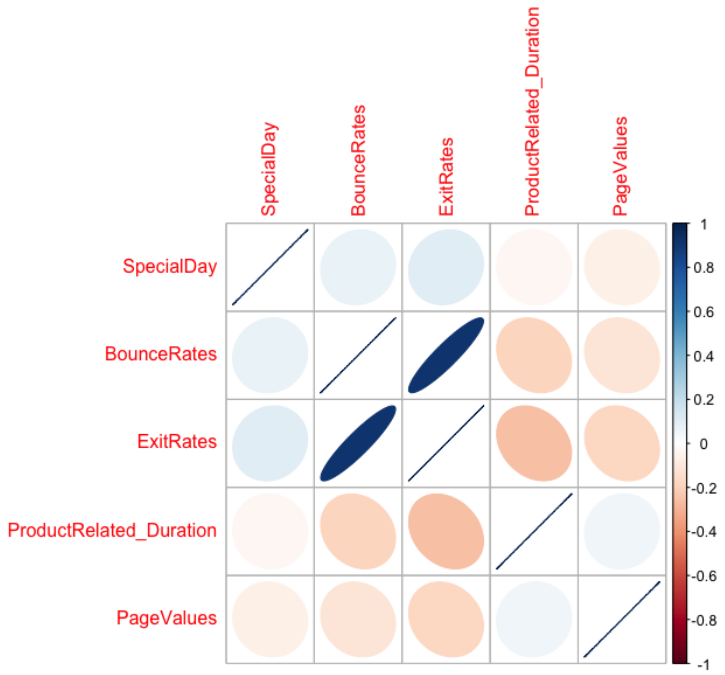


4.3 Correlation and co-variation analysis

```
# numeric variables
Shopping.num <- Shopping[, c("ProductRelated_Duration", "BounceRates",
"ExitRates", "PageValues", "SpecialDay")]

# Correlation plots
Shopping.cor <- cor(Shopping.num)
library(corrplot)
corrplot(Shopping.cor,
order= "hclust",
method= "number")
```



```
corrplot(Shopping.cor,
order= "hclust",
method= "ellipse")
```



Chi-square Test (cat v. cat)

```
# Visitor Type vs Revenue
table_vt <- table(Shopping$VisitorType, Shopping$Revenue)
chisq.test(table_vt)
```

```

    Pearson's Chi-squared test

data:  table_vt
X-squared = 135.25, df = 2, p-value < 2.2e-16
```

```
# Weekend vs Revenue
table_weekend <- table(Shopping$Weekend, Shopping$Revenue)
chisq.test(table_weekend)
```

```

    Pearson's Chi-squared test with Yates' continuity correction

data:  table_weekend
X-squared = 10.391, df = 1, p-value = 0.001266
```

```
# Month vs Revenue
table_month <- table(Shopping$Month, Shopping$Revenue)
chisq.test(table_month)
```

```

    Pearson's Chi-squared test

data:  table_month
X-squared = 384.93, df = 9, p-value < 2.2e-16
```

ANOVA Test (num vs. cat)-product duration differ by Revenue

```
anova_result <- aov(ProductRelated_Duration ~ Revenue, data = Shopping)
summary(anova_result)
```

```
               Df    Sum Sq   Mean Sq F value Pr(>F)    
Revenue         1 1.048e+09 1.048e+09     293 <2e-16 ***
Residuals   12328 4.410e+10 3.577e+06                   
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

5.2 Initial Logistic Regression Model

```
# Fit the model

logit_model <- glm(Revenue ~ ProductRelated_Duration + BounceRates + 
ExitRates + VisitorType + Weekend + Month,
                   family = binomial, data = Shopping)

# Summary of results
summary(logit_model)
```

```

Call:
glm(formula = Revenue ~ ProductRelated_Duration + BounceRates + 
    ExitRates + VisitorType + Weekend + Month, family = binomial,
    data = Shopping)

Coefficients:
                               Estimate Std. Error z value Pr(>|z|)
(Intercept)                  -5.906e-01  1.442e-01  -4.095 4.23e-05 ***
ProductRelated_Duration       8.474e-05  1.227e-05   6.906 4.98e-12 ***
BounceRates                   8.528e-01  3.281e+00   0.260 0.794943
ExitRates                    -3.172e+01  2.173e+00 -14.597  < 2e-16 ***
VisitorTypeOther              2.953e-01  3.111e-01   0.949 0.342439
VisitorTypeReturning_Visitor -3.525e-01  6.929e-02  -5.088 3.62e-07 ***
WeekendTRUE                   4.624e-02  6.033e-02   0.766 0.443434
MonthDec                     -3.746e-01  1.520e-01  -2.464 0.013729 *
MonthFeb                     -2.034e+00  6.012e-01  -3.383 0.000718 ***
MonthJul                      6.023e-03  1.915e-01   0.031 0.974911
MonthJune                    -3.482e-01  2.427e-01  -1.435 0.151416
MonthMar                     -5.523e-01  1.532e-01  -3.606 0.000310 ***
MonthMay                     -3.585e-01  1.435e-01  -2.499 0.012459 *
MonthNov                      4.693e-01  1.390e-01   3.377 0.000732 ***
MonthOct                      1.170e-01  1.704e-01   0.686 0.492472
MonthSep                      1.667e-02  1.806e-01   0.092 0.926457
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 10624.8  on 12329  degrees of freedom
Residual deviance:  9302.4  on 12314  degrees of freedom
AIC: 9334.4

Number of Fisher Scoring iterations: 7
```

```
# Multicollinearity Test (VIF)
library(car)
vif(logit_model)

                            GVIF Df GVIF^(1/(2*Df))
ProductRelated_Duration 1.111288  1        1.054176
BounceRates             1.753683  1        1.324267
ExitRates               1.798105  1        1.340934
VisitorType             1.166589  2        1.039273
Weekend                 1.009531  1        1.004754
Month                   1.096168  9        1.005114
```

Balancing the Data

```
set.seed(1) # Random seed

# Separate the positive and negative classes
Shopping.1 <- subset(Shopping, Revenue == TRUE)  # Purchases
Shopping.0 <- subset(Shopping, Revenue == FALSE) # Non-purchases

cat("Positive observations = ", nrow(Shopping.1), "\n")

Positive observations =  1908

cat("Negative observations = ", nrow(Shopping.0), "\n")

Negative observations =  10422

# Oversample positives to match the number of negatives
Shopping.1.oversampled <- Shopping.1[sample(nrow(Shopping.1), size =
nrow(Shopping.0), replace = TRUE), ]
cat("Oversampled positive observations = ", nrow(Shopping.1.oversampled),
"\n")

Oversampled positive observations =  10422

# Combine oversampled positives with original negatives
Shopping.bal <- rbind(Shopping.0, Shopping.1.oversampled)
cat("Total balanced observations = ", nrow(Shopping.bal), "\n")

Total balanced observations =  20844

# Check new class balance
table(Shopping.bal$Revenue)


FALSE  TRUE
10422 10422
```

5.5 Model Candidates Specification Logit model with balanced data

```
logit_model_bal <- glm(Revenue ~ ProductRelated_Duration + BounceRates +
ExitRates + VisitorType + Weekend + Month,
                   family = binomial, data = Shopping.bal)

# Summary of results
summary(logit_model_bal)


Call:
glm(formula = Revenue ~ ProductRelated_Duration + BounceRates +
    ExitRates + VisitorType + Weekend + Month, family = binomial,
    data = Shopping.bal)

Coefficients:
                               Estimate Std. Error z value Pr(>|z|)
(Intercept)                   1.088e+00  8.773e-02  12.401  < 2e-16 ***
ProductRelated_Duration       1.025e-04  9.102e-06  11.258  < 2e-16 ***
BounceRates                   2.355e+00  1.663e+00   1.416  0.15677
ExitRates                    -3.214e+01  1.176e+00 -27.329  < 2e-16 ***
```

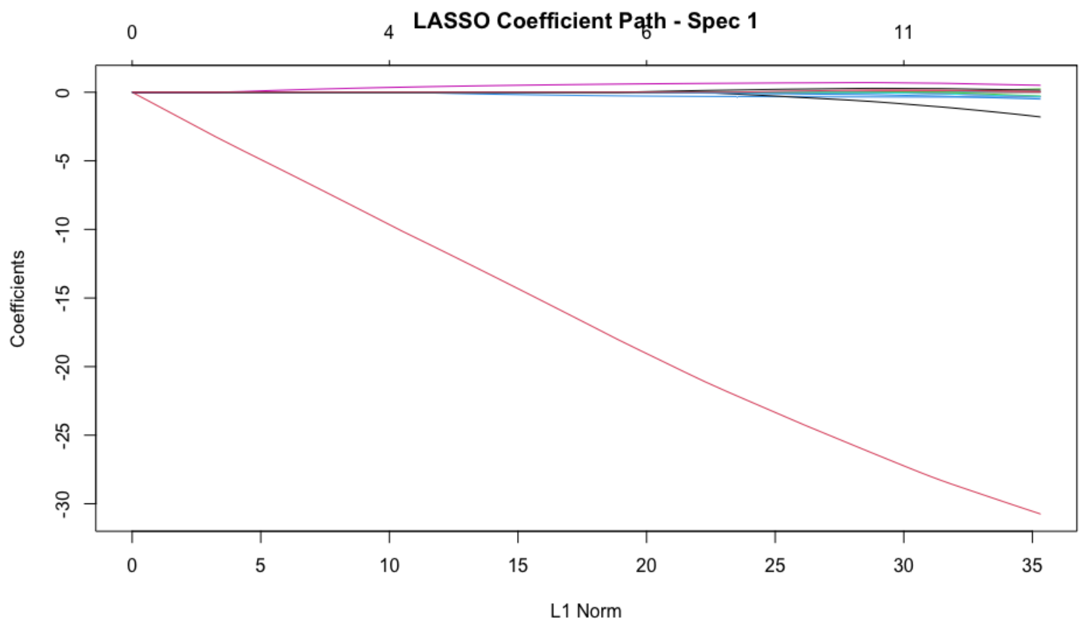
```
VisitorTypeOther                1.158e-02  2.030e-01   0.057  0.95451
VisitorTypeReturning_Visitor -4.259e-01  4.246e-02 -10.032  < 2e-16 ***
WeekendTRUE                   2.583e-02  3.587e-02   0.720  0.47139
MonthDec                     -3.263e-01  9.003e-02  -3.624  0.00029 ***
MonthFeb                     -2.340e+00  3.280e-01  -7.134 9.73e-13 ***
MonthJul                      9.160e-02  1.136e-01   0.807  0.41993
MonthJune                    -3.279e-01  1.395e-01  -2.351  0.01872 *
MonthMar                     -5.052e-01  9.004e-02  -5.611 2.02e-08 ***
MonthMay                     -2.788e-01  8.529e-02  -3.268  0.00108 **
MonthNov                      5.109e-01  8.485e-02   6.021 1.73e-09 ***
MonthOct                      2.495e-01  1.038e-01   2.403  0.01626 *
MonthSep                      1.471e-01  1.090e-01   1.350  0.17692
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 28896  on 20843  degrees of freedom
Residual deviance: 24338  on 20828  degrees of freedom
AIC: 24370

Number of Fisher Scoring iterations: 6
```

```
# Multicollinearity Check (again) for balanced data
vif(logit_model_bal)
```

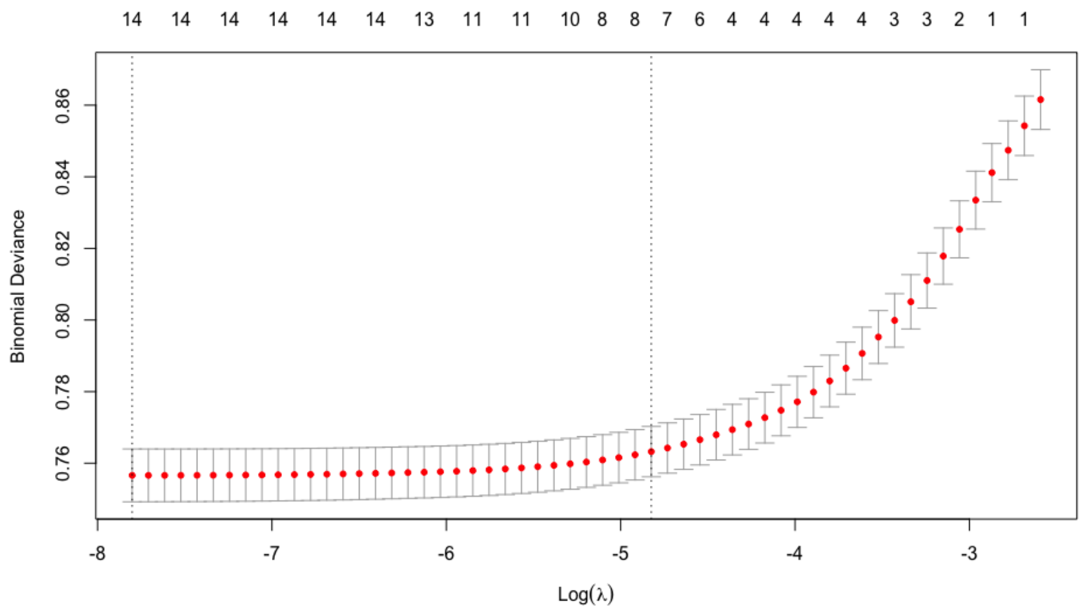
```
                            GVIF Df GVIF^(1/(2*Df))
ProductRelated_Duration 1.147739  1        1.071326
BounceRates             1.936491  1        1.391579
ExitRates               2.018948  1        1.420897
VisitorType             1.145355  2        1.034511
Weekend                 1.009520  1        1.004749
Month                   1.117384  9        1.006185
```

LASSO Model Unbalanced Specification

```
library(glmnet)

x1 <- model.matrix(Revenue ~ ProductRelated_Duration + BounceRates +
ExitRates +
                   VisitorType + Weekend + Month, data = Shopping)[, -1]
y1 <- Shopping$Revenue

# Fit LASSO model
lasso_model1 <- glmnet(x1, y1, alpha = 1, family = "binomial")
plot(lasso_model1, main = "LASSO Coefficient Path - Spec 1")
```



```
# Cross-validation
set.seed(1)
lasso_cv1 <- cv.glmnet(x1, y1, alpha = 1, family = "binomial")
plot(lasso_cv1)
```

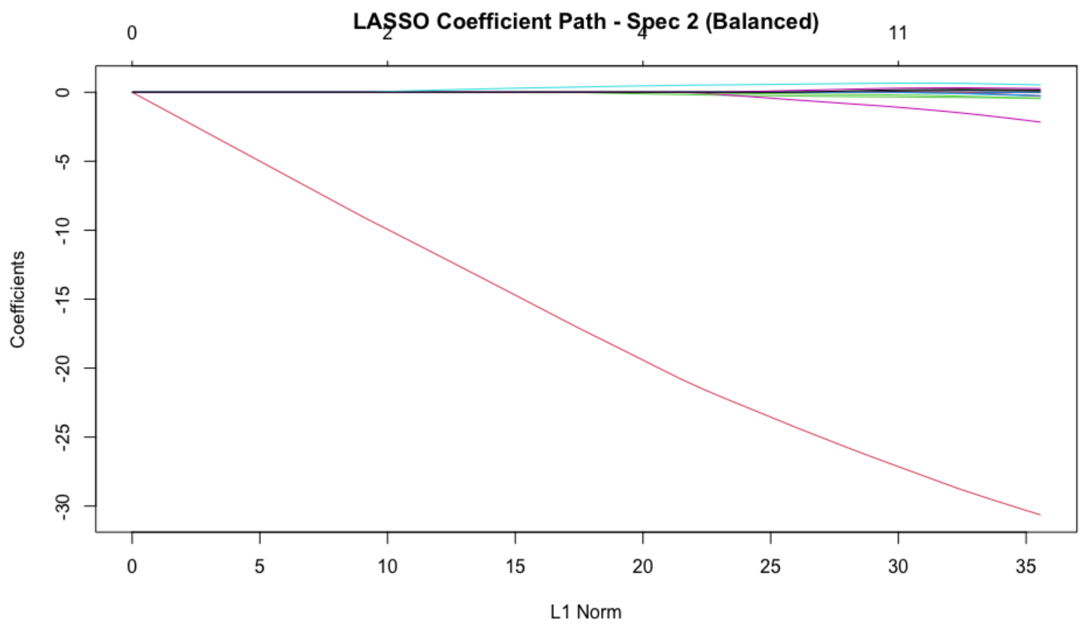


```
# Extract optimal lambda and deviance
lasso_lambda1 <- lasso_cv1$lambda.min
lasso_coef1 <- coef(lasso_cv1, s = lasso_lambda1)
```

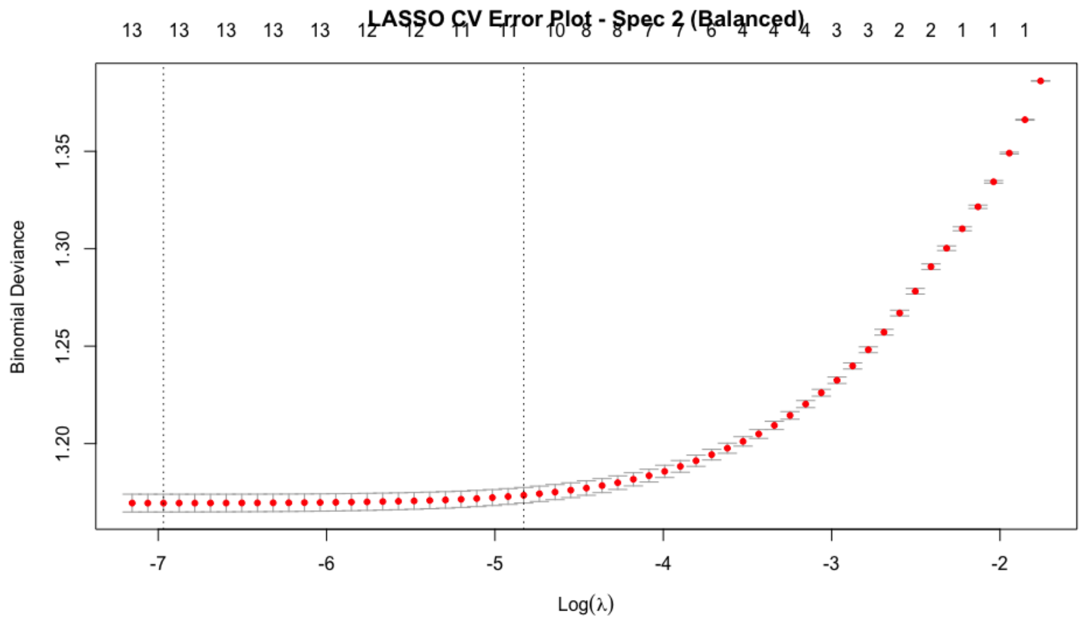
```
round(cbind("Best Lambda" = lasso_lambda1,
            "Best Log Lambda" = log(lasso_lambda1),
            "Best 10FCV" = min(lasso_cv1$cvm)), 5)

     Best Lambda Best Log Lambda Best 10FCV
[1,]     0.00041        -7.80163    0.75661
```

LASSO Model Balanced Specification

```
library(glmnet)

# Create predictor matrix (x2) and outcome vector (y2) from balanced dataset
x2 <- model.matrix(Revenue ~ ProductRelated_Duration + BounceRates +
ExitRates +
                   VisitorType + Weekend + Month, data = Shopping.bal)[, -1]
y2 <- Shopping.bal$Revenue

# Fit LASSO model
lasso_model2 <- glmnet(x2, y2, alpha = 1, family = "binomial")
plot(lasso_model2, main = "LASSO Coefficient Path - Spec 2 (Balanced)")
```



```
# Cross-validation
set.seed(1)
lasso_cv2 <- cv.glmnet(x2, y2, alpha = 1, family = "binomial")
plot(lasso_cv2, main = "LASSO CV Error Plot - Spec 2 (Balanced)")
```



```
# Extract optimal lambda and deviance
lasso_lambda2 <- lasso_cv2$lambda.min
lasso_coef2 <- coef(lasso_cv2, s = lasso_lambda2)

round(cbind("Best Lambda" = lasso_lambda2,
            "Best Log Lambda" = log(lasso_lambda2),
            "Best 10FCV" = min(lasso_cv2$cvm)), 5)

     Best Lambda Best Log Lambda Best 10FCV
[1,]     0.00094        -6.96873    1.16945
```

List Results

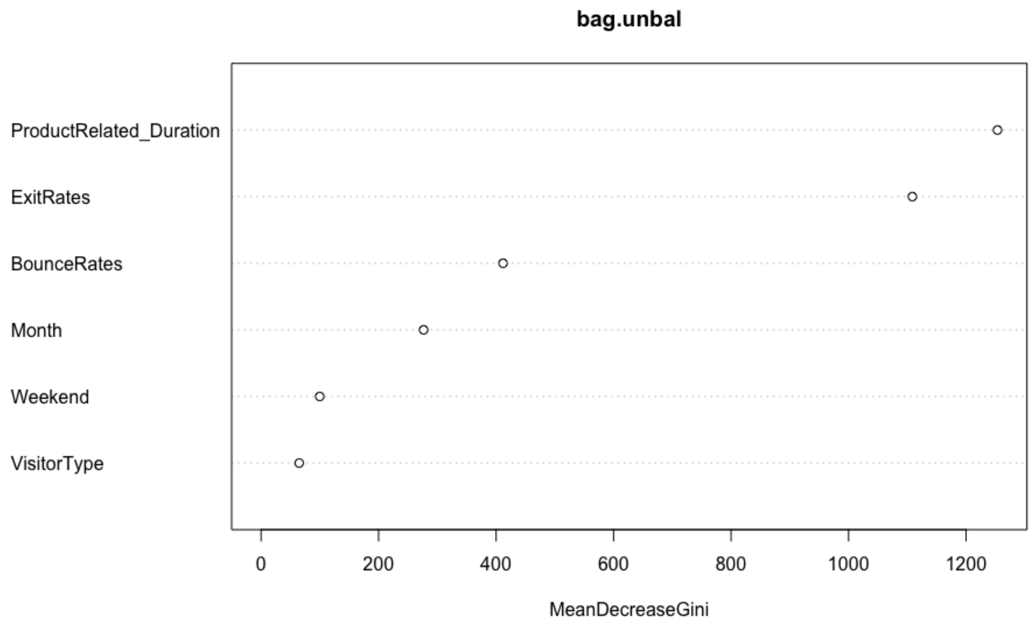
```
# For Spec 1 (Unbalanced)
lasso_lambda1 <- lasso_cv1$lambda.min
min_cv1 <- min(lasso_cv1$cvm)

# For Spec 2 (Balanced)
lasso_lambda2 <- lasso_cv2$lambda.min
min_cv2 <- min(lasso_cv2$cvm)

lasso.results <- round(
  rbind(
    "Spec 1: Unbalanced" = c("Best Lambda" = lasso_lambda1,
                             "Best Log Lambda" = log(lasso_lambda1),
                             "Best 10FCV" = min_cv1),
    "Spec 2: Balanced"   = c("Best Lambda" = lasso_lambda2,
                             "Best Log Lambda" = log(lasso_lambda2),
                             "Best 10FCV" = min_cv2)
  ), 5
)

lasso.results

                      Best Lambda Best Log Lambda Best 10FCV
Spec 1: Unbalanced      0.00041        -7.80163    0.75661
Spec 2: Balanced        0.00094        -6.96873    1.16945
```

10FCV for Both Logit Models

```
library(boot)

# Logistic Model - Spec 1: Unbalanced
set.seed(1)
cv_logit1 <- cv.glm(data = Shopping, glmfit = logit_model, K = 10)

# Logistic Model - Spec 2: Balanced
set.seed(1)
cv_logit2 <- cv.glm(data = Shopping.bal, glmfit = logit_model_bal, K = 10)
```

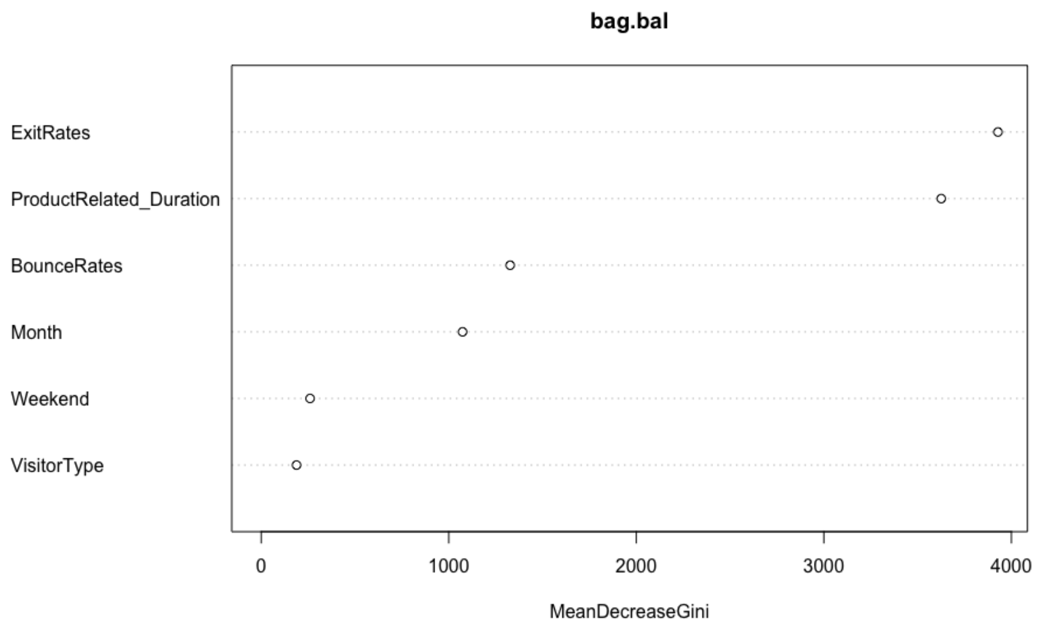
Bagging Model

```
# Set seed and RNG setting
RNGkind(sample.kind = "default")
set.seed(1)

# Load library
library(randomForest)

# Ensure binary outcome is a factor with labels
Shopping$Revenue <- factor(Shopping$Revenue, levels = c(FALSE, TRUE), labels
= c("No", "Yes"))
Shopping.bal$Revenue <- factor(Shopping.bal$Revenue, levels = c(FALSE, TRUE),
labels = c("No", "Yes"))

# Define modeling formula
model_formula <- Revenue ~ ProductRelated_Duration + BounceRates + ExitRates
+
                              VisitorType + Weekend + Month

# Create NA-safe subsets for modeling
Shopping.clean <- na.omit(Shopping[, all.vars(model_formula)])
Shopping.bal.clean <- na.omit(Shopping.bal[, all.vars(model_formula)])

# ------------------------------
# 1. Bagging (Unbalanced, mtry = 6)
# ------------------------------
bag.unbal <- randomForest(model_formula,
                          data = Shopping.clean,
                          mtry = 6,
                          importance = TRUE)

print(bag.unbal)


Call:
 randomForest(formula = model_formula, data = Shopping.clean,      mtry = 6,
importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 6

        OOB estimate of  error rate: 16.94%
Confusion matrix:
      No Yes class.error
No  9951 471  0.04519286
Yes 1618 290  0.84800839

cat("\nClassification Error Rate (Bagging - Unbalanced) =",
    bag.unbal$err.rate[nrow(bag.unbal$err.rate), "OOB"])


Classification Error Rate (Bagging - Unbalanced) = 0.1694242

varImpPlot(bag.unbal, type = 2)
```



```
importance(bag.unbal, type = 2)

                        MeanDecreaseGini
ProductRelated_Duration       1253.47359
```

```
BounceRates                        411.85033
ExitRates                         1108.58024
VisitorType                         64.89726
Weekend                             99.81519
Month                              276.65776

# -------------------------------
# 2. Bagging (Balanced, mtry = 6)
# -------------------------------
bag.bal <- randomForest(model_formula,
                        data = Shopping.bal.clean,
                        mtry = 6,
                        importance = TRUE)

print(bag.bal)


Call:
 randomForest(formula = model_formula, data = Shopping.bal.clean,      mtry = 
6, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 6

        OOB estimate of  error rate: 5.16%
Confusion matrix:
      No   Yes class.error
No  9390  1032 0.099021301
Yes   43 10379 0.004125888

cat("\nClassification Error Rate (Bagging - Balanced) =",
    bag.bal$err.rate[nrow(bag.bal$err.rate), "OOB"])


Classification Error Rate (Bagging - Balanced) = 0.05157359

varImpPlot(bag.bal, type = 2)
```

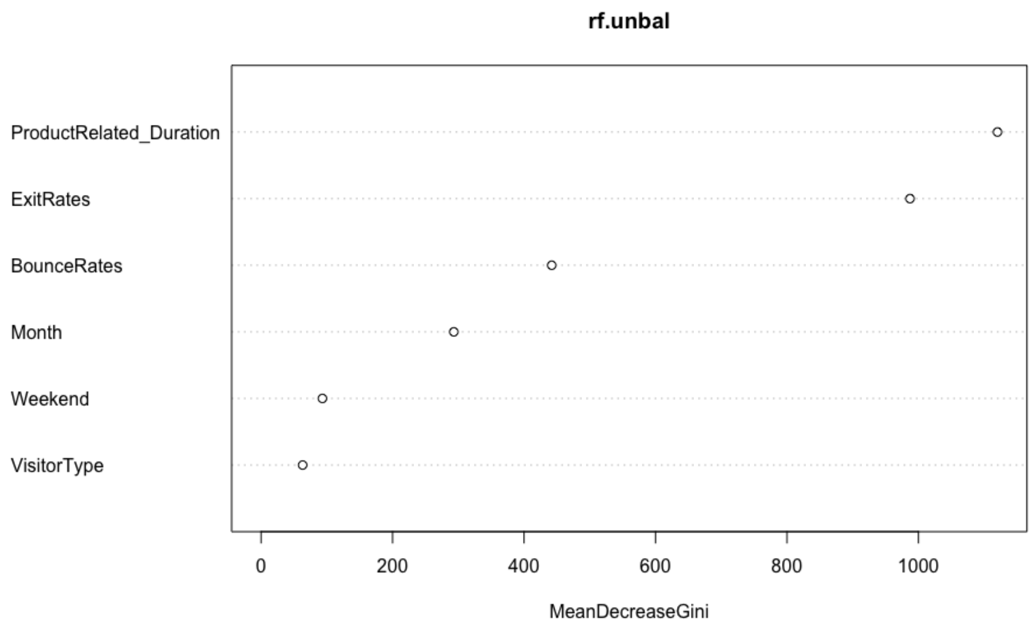


```
importance(bag.bal, type = 2)

                        MeanDecreaseGini
ProductRelated_Duration        3625.9984
BounceRates                    1327.4517
ExitRates                      3928.4428
VisitorType                     188.3419
Weekend                         260.4771
Month                          1073.6137
```

Random Forest

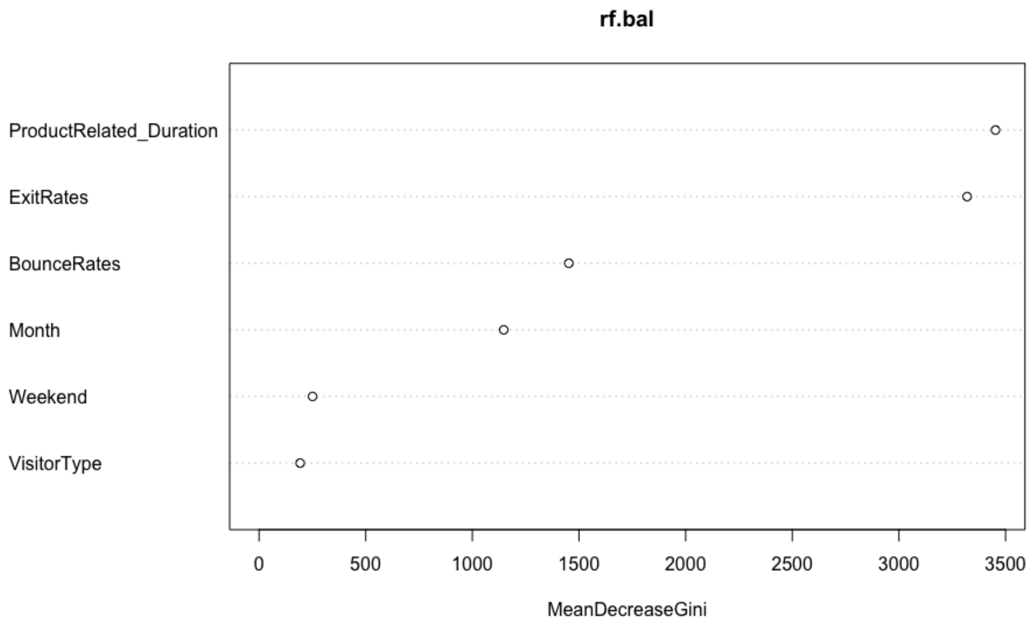
```
# 3. Random Forest (Unbalanced, mtry = 3)
# -------------------------------
rf.unbal <- randomForest(model_formula,
                         data = Shopping.clean,
                         mtry = 3,
                         importance = TRUE)

print(rf.unbal)


Call:
 randomForest(formula = model_formula, data = Shopping.clean,      mtry = 3,
importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 16.26%
```

```
Confusion matrix:
       No Yes class.error
No  10056 366  0.03511802
Yes  1639 269  0.85901468

cat("\nClassification Error Rate (Random Forest - Unbalanced) =",
    rf.unbal$err.rate[nrow(rf.unbal$err.rate), "OOB"])


Classification Error Rate (Random Forest - Unbalanced) = 0.1626115

varImpPlot(rf.unbal, type = 2)
```



```
importance(rf.unbal, type = 2)

                        MeanDecreaseGini
ProductRelated_Duration       1120.22637
BounceRates                    442.13433
ExitRates                      987.14257
VisitorType                     63.23252
Weekend                         93.32027
Month                          293.18836

# --------------------------------
# 4. Random Forest (Balanced, mtry = 3)
# --------------------------------
rf.bal <- randomForest(model_formula,
                       data = Shopping.bal.clean,
                       mtry = 3,
                       importance = TRUE)
```

```
print(rf.bal)


Call:
 randomForest(formula = model_formula, data = Shopping.bal.clean,      mtry = 
3, importance = TRUE) 
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 5.72%
Confusion matrix:
      No   Yes class.error
No  9274  1148 0.110151602
Yes   44 10378 0.004221838

cat("\nClassification Error Rate (Random Forest - Balanced) =", 
    rf.bal$err.rate[nrow(rf.bal$err.rate), "OOB"])


Classification Error Rate (Random Forest - Balanced) = 0.05718672

varImpPlot(rf.bal, type = 2)
```



```
importance(rf.bal, type = 2)

                        MeanDecreaseGini
ProductRelated_Duration        3452.6982
BounceRates                    1452.1695
```

```
ExitRates                      3319.5724
VisitorType                     192.7371
Weekend                         250.7165
Month                          1147.1118
```

5.6 - 10FCV For Logit, LASSO ,Random Forest, and Bagging models

10FCV for Bagging and Random Forest

```
# Load caret and randomForest
library(caret)
library(randomForest)

# Define formula (already established)
model_formula <- Revenue ~ ProductRelated_Duration + BounceRates + ExitRates
+ VisitorType + Weekend + Month

# Set control for 10-fold cross-validation
ctrl <- trainControl(method = "cv", number = 10)

# 1. Bagging - Unbalanced
set.seed(1)
bag_cv_unbal <- train(model_formula, data = Shopping.clean,
                      method = "treebag", trControl = ctrl)

# 2. Bagging - Balanced
set.seed(1)
bag_cv_bal <- train(model_formula, data = Shopping.bal.clean,
                    method = "treebag", trControl = ctrl)

# 3. Random Forest - Unbalanced
set.seed(1)
rf_cv_unbal <- train(model_formula, data = Shopping.clean,
                     method = "rf", trControl = ctrl, tuneGrid =
data.frame(mtry = 3))

# 4. Random Forest - Balanced
set.seed(1)
rf_cv_bal <- train(model_formula, data = Shopping.bal.clean,
                   method = "rf", trControl = ctrl, tuneGrid =
data.frame(mtry = 3))
```

Conversion for Bagging and Random Forest

```
# Extract and convert accuracy to error rate
bag_cv_unbal_error <- 1 - max(bag_cv_unbal$results$Accuracy)
bag_cv_bal_error   <- 1 - max(bag_cv_bal$results$Accuracy)
rf_cv_unbal_error  <- 1 - max(rf_cv_unbal$results$Accuracy)
rf_cv_bal_error    <- 1 - max(rf_cv_bal$results$Accuracy)
```

Lasso and Logit Assignment for 10FCV

```
lasso_cv_error1 <- min(lasso_cv1$cvm)
lasso_cv_error2 <- min(lasso_cv2$cvm)
logit_cv_error1 <- cv_logit1$delta[2]
logit_cv_error2 <- cv_logit2$delta[2]
```

Complete Model Comparison of 10FCV Errors

```
model_comparison <- data.frame(
  Model = c("Logit Spec 1 (Unbalanced)", "Logit Spec 2 (Balanced)",
            "LASSO Spec 1 (Unbalanced)", "LASSO Spec 2 (Balanced)",
            "Bagging - Unbalanced", "Bagging - Balanced",
            "Random Forest - Unbalanced", "Random Forest - Balanced"),
  `10FCV Error` = round(c(logit_cv_error1, logit_cv_error2,
                          lasso_cv_error1, lasso_cv_error2,
                          bag_cv_unbal_error, bag_cv_bal_error,
                          rf_cv_unbal_error, rf_cv_bal_error), 5)
)

# View
model_comparison

                       Model X10FCV.Error
1  Logit Spec 1 (Unbalanced)      0.11820
2    Logit Spec 2 (Balanced)      0.20225
3  LASSO Spec 1 (Unbalanced)      0.75661
4    LASSO Spec 2 (Balanced)      1.16945
5       Bagging - Unbalanced      0.17583
6         Bagging - Balanced      0.06592
7 Random Forest - Unbalanced      0.15474
8   Random Forest - Balanced      0.23570
```

References

1. O'Rourke, D. (2022, October 18). Brands losing a record $29 for each new customer acquired. SimplicityDX. https://www.simplicitydx.com/blogs/press-release-brands-losing-a-record-29-for-each-new-customer-acquired

2. Sakar, C. & Kastro, Y. (2018). Online Shoppers Purchasing Intention Dataset [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5F88Q.

3. Smith, B. (2024, May). How to improve each stage of your ecommerce sales funnel. wordstream.com/blog/ws/2020/07/01/ecommerce-sales-funnel