Customer Purchasing Intention

Vivek Teja

5/23/2020

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

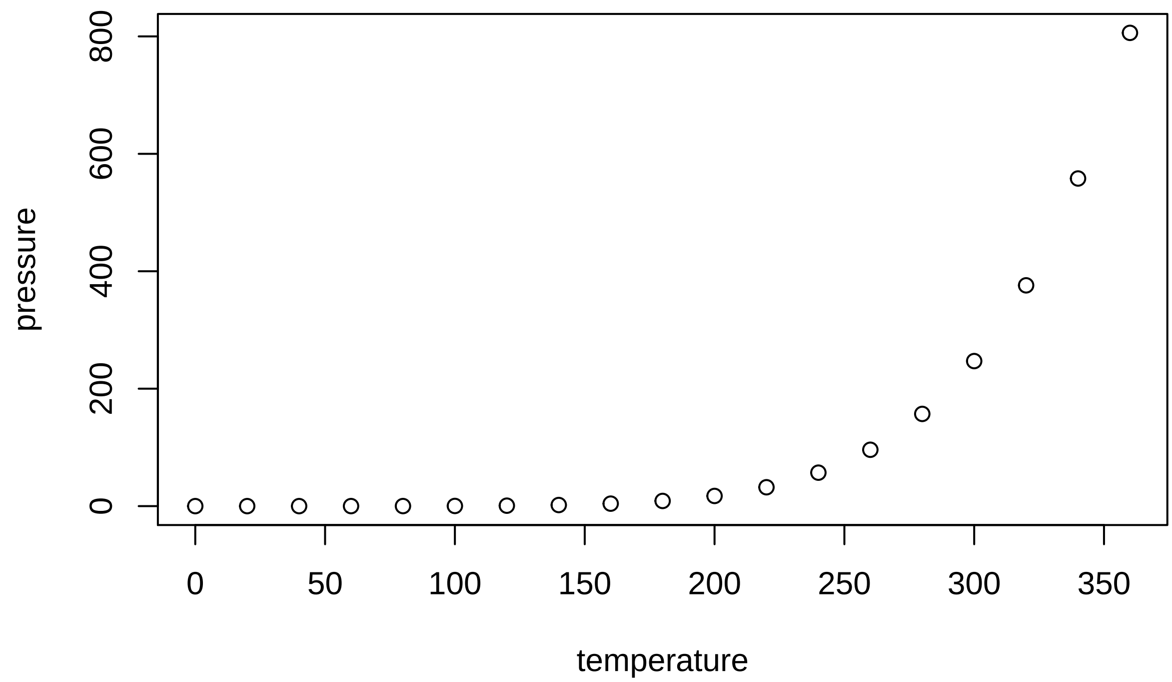
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```r
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00  
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

```
# loading packages
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.6.3
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 3.0-2
```

```r
library(readr)
online_shoppers_intention <- read_csv("online_shoppers_intention.csv")
```

```
## Parsed with column specification:
## cols(
##   Administrative = col_double(),
##   Administrative_Duration = col_double(),
##   Informational = col_double(),
##   Informational_Duration = col_double(),
##   ProductRelated = col_double(),
##   ProductRelated_Duration = col_double(),
##   BounceRates = col_double(),
##   ExitRates = col_double(),
##   PageValues = col_double(),
##   SpecialDay = col_double(),
##   Month = col_character(),
##   OperatingSystems = col_double(),
##   Browser = col_double(),
##   Region = col_double(),
##   TrafficType = col_double(),
##   VisitorType = col_character(),
##   Weekend = col_logical(),
##   Revenue = col_logical()
## )
```

```r
#                              /**** Data preperation ****/
# reading data and creating dataframe

df = online_shoppers_intention
```

```r
# dataframe dimensions and features
glimpse(df)
```

```
## Observations: 12,330
## Variables: 18
## $ Administrative          <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, ...
## $ Administrative_Duration <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ Informational           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ Informational_Duration  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
```

```
## $ ProductRelated          <dbl> 1, 2, 1, 2, 10, 19, 1, 0, 2, 3, 3, 16, 7, 6, ...
## $ ProductRelated_Duration <dbl> 0.000000, 64.000000, 0.000000, 2.666667, 627....
## $ BounceRates             <dbl> 0.200000000, 0.000000000, 0.200000000, 0.0500...
## $ ExitRates               <dbl> 0.200000000, 0.100000000, 0.200000000, 0.1400...
## $ PageValues              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ SpecialDay              <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4, 0.0, 0.8, ...
## $ Month                   <chr> "Feb", "Feb", "Feb", "Feb", "Feb", "Feb", "Fe...
## $ OperatingSystems        <dbl> 1, 2, 4, 3, 3, 2, 2, 1, 2, 2, 1, 1, 1, 2, 3, ...
## $ Browser                 <dbl> 1, 2, 1, 2, 3, 2, 4, 2, 2, 4, 1, 1, 1, 5, 2, ...
## $ Region                  <dbl> 1, 1, 9, 2, 1, 1, 3, 1, 2, 1, 3, 4, 1, 1, 3, ...
## $ TrafficType             <dbl> 1, 2, 3, 4, 4, 3, 3, 5, 3, 2, 3, 3, 3, 3, 3, ...
## $ VisitorType             <chr> "Returning_Visitor", "Returning_Visitor", "Re...
## $ Weekend                 <lgl> FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FALS...
## $ Revenue                 <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL...
```

```
# dataframe summary
summary(df)
```

```
##  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
##  Length:12330       Min.   :1.000    Min.   : 1.000   Min.   :1.000
##  Class :character   1st Qu.:2.000    1st Qu.: 2.000   1st Qu.:1.000
##  Mode  :character   Median :2.000    Median : 2.000   Median :3.000
##                     Mean   :2.124    Mean   : 2.357   Mean   :3.147
##                     3rd Qu.:3.000    3rd Qu.: 2.000   3rd Qu.:4.000
##                     Max.   :8.000    Max.   :13.000   Max.   :9.000
##   TrafficType    VisitorType          Weekend          Revenue
##  Min.   : 1.00   Length:12330       Mode :logical   Mode :logical
##  1st Qu.: 2.00   Class :character   FALSE:9462      FALSE:10422
##  Median : 2.00   Mode  :character   TRUE :2868      TRUE :1908
##  Mean   : 4.07
##  3rd Qu.: 4.00
##  Max.   :20.00
```

```r
# missing value analysis
sapply(df, function(col) sum(is.na(col)))
```

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

```r
# creating list of numerical and categorical variables
features_numerical = c('Administrative','Administrative_Duration','Informational','Informational_Durati
features_categorical = c('OperatingSystems','Browser','Month','Region','TrafficType','VisitorType','Weel
```

```r
# creating dataframes with numerical and categorical features
df_numerical = df[features_numerical]
df_categorical = df[features_categorical]
head(df)
```

```
## # A tibble: 6 x 18
##   Administrative Administrative_... Informational Informational_D... ProductRelated
##            <dbl>              <dbl>         <dbl>             <dbl>          <dbl>
## 1              0                  0             0                 0              1
## 2              0                  0             0                 0              2
## 3              0                  0             0                 0              1
## 4              0                  0             0                 0              2
## 5              0                  0             0                 0             10
## 6              0                  0             0                 0             19
## # ... with 13 more variables: ProductRelated_Duration <dbl>, BounceRates <dbl>,
## #   ExitRates <dbl>, PageValues <dbl>, SpecialDay <dbl>, Month <chr>,
## #   OperatingSystems <dbl>, Browser <dbl>, Region <dbl>, TrafficType <dbl>,
## #   VisitorType <chr>, Weekend <lgl>, Revenue <lgl>
```

```r
# modifying the datatypes
df_categorical$Revenue = ifelse(df_categorical$Revenue == TRUE,1,0)
df_categorical$Weekend = ifelse(df_categorical$Weekend == TRUE,1,0)
df_categorical$TrafficType = as.factor(df_categorical$TrafficType)
df_categorical$Region = as.factor(df_categorical$Region)
df_categorical$OperatingSystems = as.factor(df_categorical$OperatingSystems)
df_categorical$Browser = as.factor(df_categorical$Browser)
glimpse(df_categorical)
```

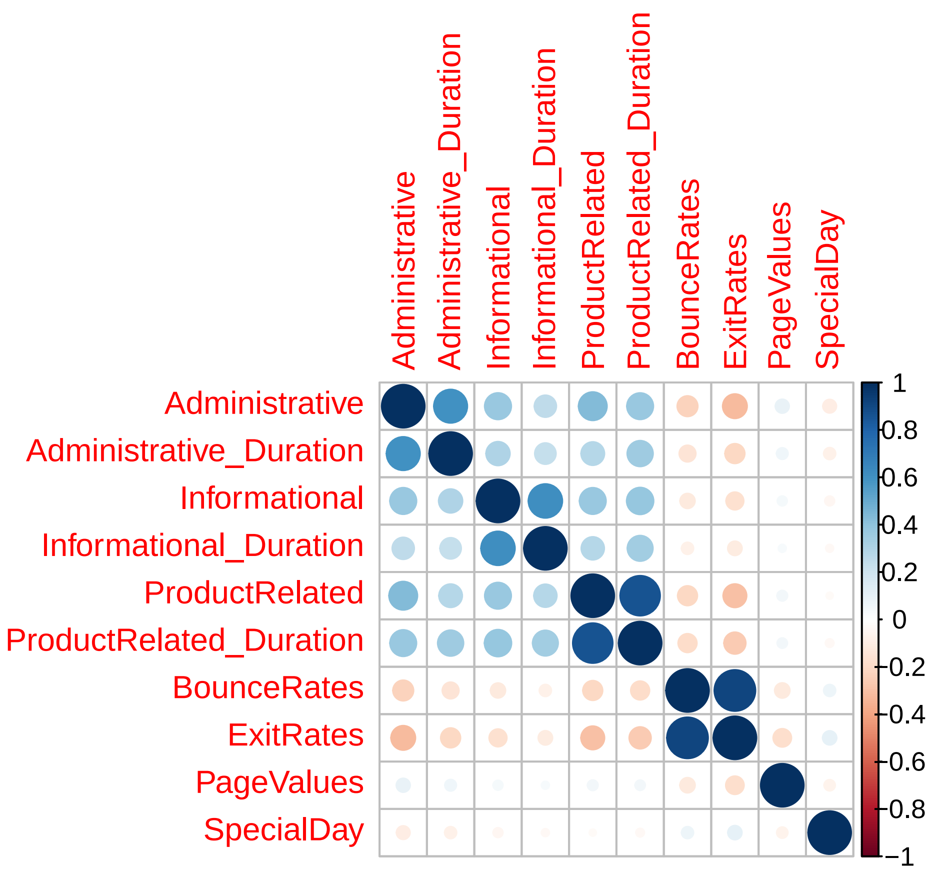
```
## Observations: 12,330
## Variables: 8
## $ OperatingSystems <fct> 1, 2, 4, 3, 3, 2, 2, 1, 2, 2, 1, 1, 1, 2, 3, 1, 1, 1...
## $ Browser          <fct> 1, 2, 1, 2, 3, 2, 4, 2, 2, 4, 1, 1, 1, 5, 2, 1, 1, 1...
```

```
## $ Month            <chr> "Feb", "Feb", "Feb", "Feb", "Feb", "Feb", "Feb", "Fe...
## $ Region           <fct> 1, 1, 9, 2, 1, 1, 3, 1, 2, 1, 3, 4, 1, 1, 3, 9, 4, 1...
## $ TrafficType      <fct> 1, 2, 3, 4, 4, 3, 3, 5, 3, 2, 3, 3, 3, 3, 3, 3, 3, 4...
## $ VisitorType      <chr> "Returning_Visitor", "Returning_Visitor", "Returning...
## $ Weekend          <dbl> 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1...
## $ Revenue          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
```

```
# feature engineering to reduce classes in categorical variables
df_categorical$OperatingSystems =  ifelse(df_categorical$OperatingSystems %in% c('1','2','3'), df_catego
df_categorical$Browser =  ifelse(df_categorical$Browser %in% c('1','2'), df_categorical$Browser, 'Other
df_categorical$Region =  ifelse(df_categorical$Region %in% c('1','3'), df_categorical$Browser, 'Other')
df_categorical$TrafficType =  ifelse(df_categorical$TrafficType %in% c('12','17','18'), df_categorical$
```

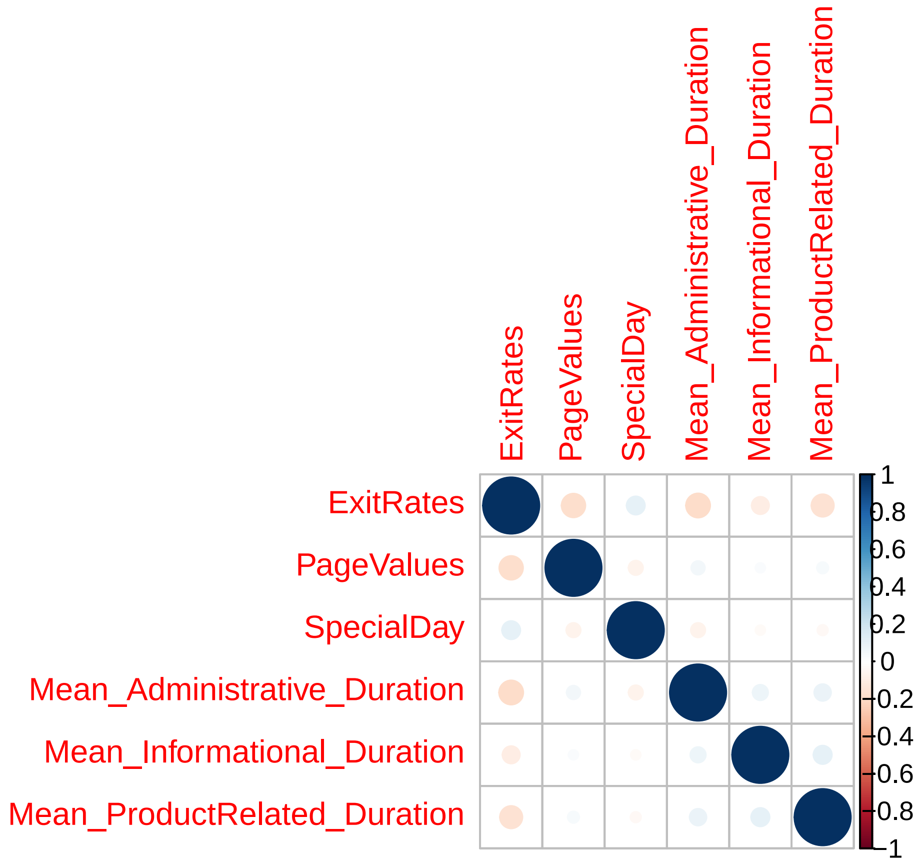
```
# creating dummy variables
df_dummy = fastDummies::dummy_cols(df_categorical[c(-7,-8)], remove_first_dummy =



                                                 TRUE)

# replacing the factor variables with dummies
df_categorical_dummy = cbind(df_dummy[,c(-1:-6)], df_categorical[,c(7,8)])
```

```
# checking for multicollinearity
corrplot::corrplot(cor(df_numerical))
```



```
# feature engineering to remove multicollinearity
df_numerical$Mean_Administrative_Duration = df_numerical$Administrative_Duration / df_numerical$Adminis
df_numerical$Mean_Informational_Duration = df_numerical$Informational_Duration / df_numerical$Informati
df_numerical$Mean_ProductRelated_Duration = df_numerical$ProductRelated_Duration / df_numerical$Product
df_numerical[is.na(df_numerical)] = 0
df_numerical_fe = df_numerical[,c(-1:-7)]
```

```
# checking for multicollinearity
corrplot::corrplot(cor(df_numerical_fe))
```




```
# combining numerical and categorical dataframes
df_cleaned = cbind(df_numerical_fe, df_categorical_dummy)
```

```
#                              /**** Modelling ****/
set.seed(1)
# train test split - 80:20
train_index = sample(1:dim(df_cleaned)[1], dim(df_cleaned)[1]*0.8, replace = FALSE)
df_train = df_cleaned[train_index, ]
df_test = df_cleaned[-train_index, ]
```

```
# base logit model
model = glm(Revenue~., data=df_train, family='binomial')
summary(model)
```

```
##
```

```
## Call:
## glm(formula = Revenue ~ ., family = "binomial", data = df_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.0578  -0.4742  -0.3451  -0.1605   3.3005
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  -1.151e+01  5.354e+02  -0.021 0.982854
## ExitRates                    -1.924e+01  1.811e+00 -10.624  < 2e-16 ***
## PageValues                    8.105e-02  2.678e-03  30.267  < 2e-16 ***
## SpecialDay                   -5.986e-02  2.571e-01  -0.233 0.815881
## Mean_Administrative_Duration  5.146e-04  6.387e-04   0.806 0.420404
## Mean_Informational_Duration   7.504e-04  3.892e-04   1.928 0.053880 .
## Mean_ProductRelated_Duration  1.050e-03  8.480e-04   1.238 0.215684
## OperatingSystems_2            1.010e-01  1.619e-01   0.624 0.532576
## OperatingSystems_3           -1.943e-01  1.783e-01  -1.090 0.275836
## OperatingSystems_Other       -9.043e-02  1.862e-01  -0.486 0.627274
## Browser_2                    -4.519e-02  2.058e-01  -0.220 0.826202
## Browser_Other                 1.094e-01  2.086e-01   0.524 0.600096
## Month_Dec                    -5.696e-01  2.048e-01  -2.781 0.005420 **
## Month_Feb                    -1.583e+00  6.383e-01  -2.480 0.013154 *
## Month_Jul                    -5.655e-02  2.504e-01  -0.226 0.821314
## Month_June                   -1.421e-01  3.059e-01  -0.464 0.642322
## Month_Mar                    -5.135e-01  2.020e-01  -2.543 0.011006 *
## Month_May                    -5.276e-01  1.952e-01  -2.703 0.006875 **
## Month_Nov                     6.701e-01  1.834e-01   3.655 0.000257 ***
## Month_Oct                     5.923e-02  2.267e-01   0.261 0.793861
## Month_Sep                    -6.841e-03  2.385e-01  -0.029 0.977114
## Region_2                      9.713e-02  1.854e-01   0.524 0.600293
## Region_Other                  3.433e-02  1.640e-01   0.209 0.834184
## TrafficType_2                -7.920e-01  5.595e+02  -0.001 0.998871
## TrafficType_Other             9.778e+00  5.354e+02   0.018 0.985429
## VisitorType_Other            -6.386e-01  6.455e-01  -0.989 0.322476
## VisitorType_Returning_Visitor -1.710e-01  9.269e-02  -1.845 0.064999 .
## Weekend                       1.260e-01  7.868e-02   1.601 0.109327
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8606.4  on 9863  degrees of freedom
## Residual deviance: 5848.4  on 9836  degrees of freedom
## AIC: 5904.4
##
## Number of Fisher Scoring iterations: 12
```

```r
y_pred_prob = predict(model, df_test, type='response')
y_pred = ifelse(y_pred_prob>0.5, 1, 0)
mean(df_test$Revenue == y_pred)
```

```
## [1] 0.8941606
```

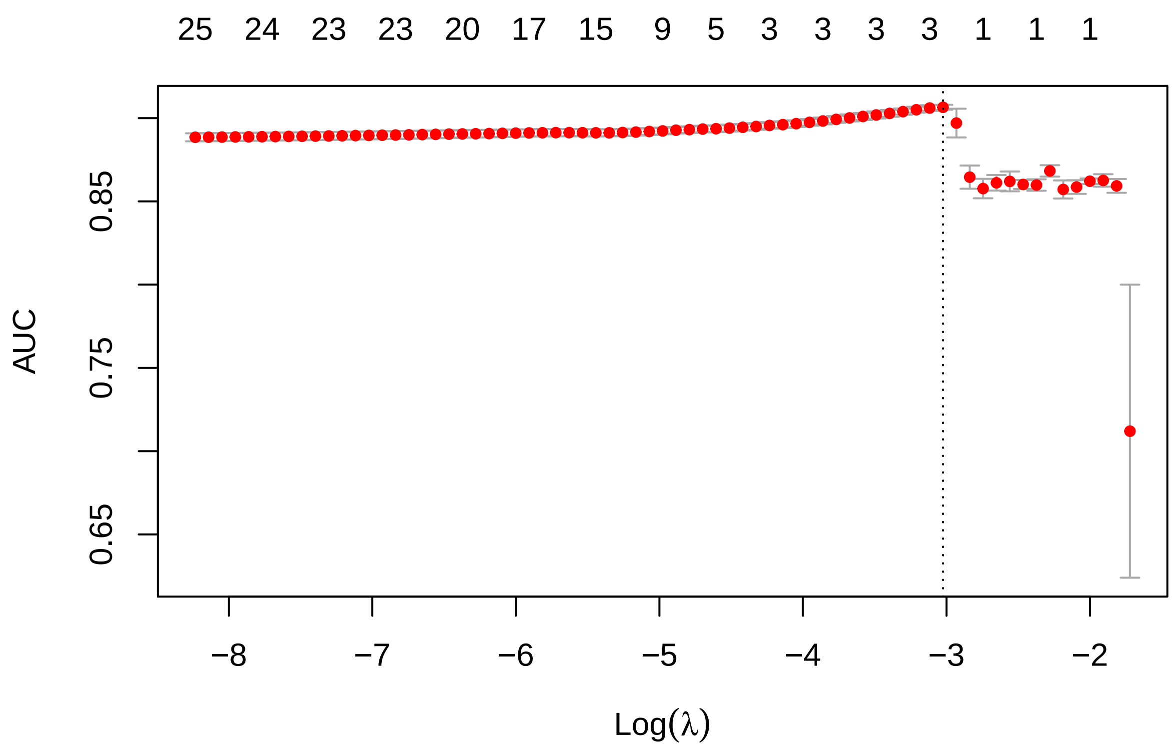
```r
ROC = roc(df_test$Revenue, y_pred)
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```r
auc(ROC)
```

```
## Area under the curve: 0.6808
```

```r
# lasso regression
X = as.matrix(df_train[,c(-28)])
y = as.matrix(df_train['Revenue'])
cv_lasso <- cv.glmnet(X, y, family='binomial', alpha=1, standardize=TRUE, nfolds=5,  type.measure='auc')
plot(cv_lasso)
```




```r
model_lasso = glmnet(X, y, alpha=1, standardize=TRUE, lambda=cv_lasso$lambda.min)
coef(model_lasso)
```

```
## 28 x 1 sparse Matrix of class "dgCMatrix"
##                                      s0
## (Intercept)                 0.118966878
## ExitRates                  -0.091458482
```

```
## PageValues                        0.006872808
## SpecialDay                        .
## Mean_Administrative_Duration      .
## Mean_Informational_Duration       .
## Mean_ProductRelated_Duration      .
## OperatingSystems_2                .
## OperatingSystems_3                .
## OperatingSystems_Other            .
## Browser_2                         .
## Browser_Other                     .
## Month_Dec                         .
## Month_Feb                         .
## Month_Jul                         .
## Month_June                        .
## Month_Mar                         .
## Month_May                         .
## Month_Nov                         0.008137810
## Month_Oct                         .
## Month_Sep                         .
## Region_2                          .
## Region_Other                      .
## TrafficType_2                     .
## TrafficType_Other                 .
## VisitorType_Other                 .
## VisitorType_Returning_Visitor     .
## Weekend                           .
```

```
# logit model with feature selection
model_fe = glm(Revenue~ExitRates+PageValues+Month_Nov, data=df_train, family='binomial')
summary(model_fe)
```

```
##
## Call:
## glm(formula = Revenue ~ ExitRates + PageValues + Month_Nov, family = "binomial",
##     data = df_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.0072  -0.4559  -0.3669  -0.1779   3.5229
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.073834   0.066342  -31.26   <2e-16 ***
## ExitRates   -20.647091   1.766114  -11.69   <2e-16 ***
## PageValues    0.081613   0.002656   30.72   <2e-16 ***
## Month_Nov     1.061094   0.070801   14.99   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8606.4  on 9863  degrees of freedom
## Residual deviance: 5913.9  on 9860  degrees of freedom
## AIC: 5921.9
```

```
##
## Number of Fisher Scoring iterations: 6
```

```
y_pred_prob = predict(model_fe, df_test, type='response')
y_pred = ifelse(y_pred_prob>0.5, 1, 0)
mean(df_test$Revenue == y_pred)
```

```
## [1] 0.892944
```

```
ROC = roc(df_test$Revenue, y_pred)
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```
auc(ROC)
```

```
## Area under the curve: 0.6789
```

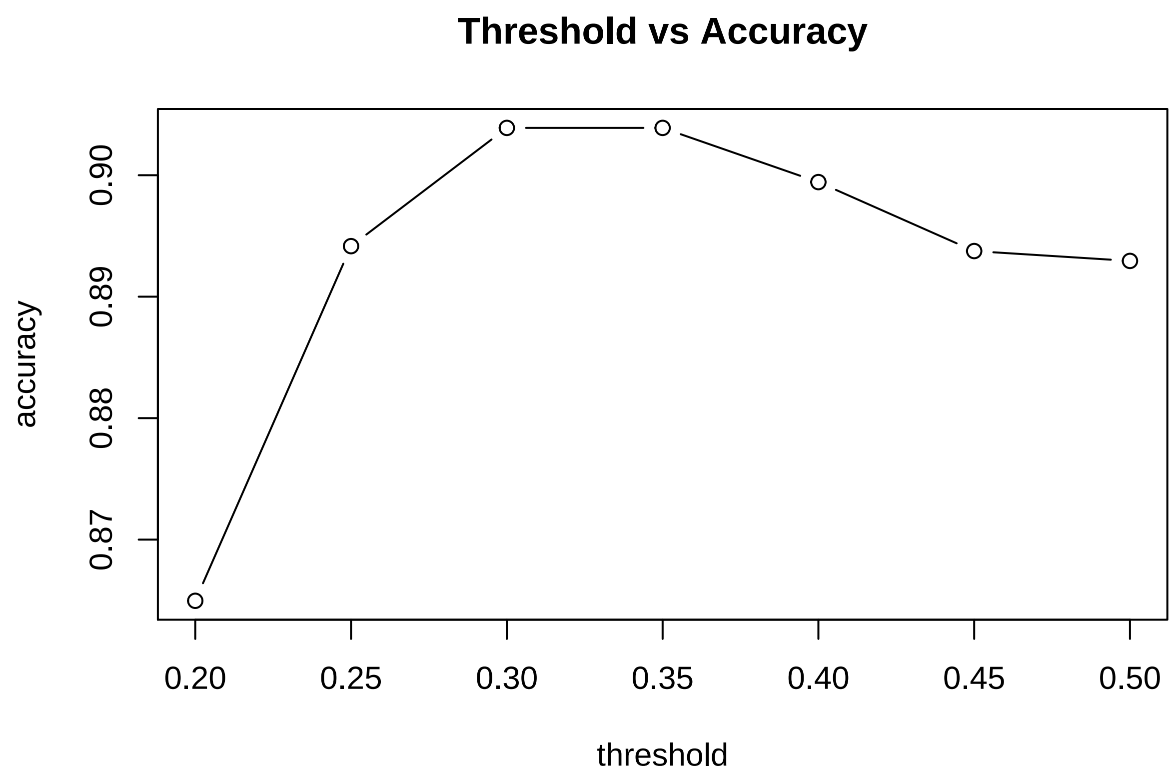
```
# finding the best threshold
threshold = seq(0.2, 0.5, by=0.05)
accuracy = c()
auc = c()
```

```
for (i in 1:7){
  y_pred = ifelse(y_pred_prob>threshold[i], 1, 0)
  accuracy[i] = mean(df_test$Revenue == y_pred)
  auc[i] = auc(roc(df_test$Revenue, y_pred))
}
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```
df_threshold_results = data.frame(threshold, accuracy, auc)
plot(threshold, accuracy, type='b', main='Threshold vs Accuracy')
```




```
# optimized logit model
model_final = glm(Revenue~ExitRates+PageValues+Month_Nov, data=df_train, family='binomial')
summary(model_final)
```

```
##
## Call:
## glm(formula = Revenue ~ ExitRates + PageValues + Month_Nov, family = "binomial",
##     data = df_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -6.0072  -0.4559  -0.3669  -0.1779   3.5229
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.073834   0.066342  -31.26   <2e-16 ***
## ExitRates   -20.647091   1.766114  -11.69   <2e-16 ***
## PageValues    0.081613   0.002656   30.72   <2e-16 ***
## Month_Nov     1.061094   0.070801   14.99   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8606.4  on 9863  degrees of freedom
## Residual deviance: 5913.9  on 9860  degrees of freedom
## AIC: 5921.9
##
## Number of Fisher Scoring iterations: 6
```

```
y_pred_prob = predict(model_final, df_test, type='response')
y_pred = ifelse(y_pred_prob>0.275, 1, 0)
mean(df_test$Revenue == y_pred)
```

```
## [1] 0.9018654
```

```
ROC = roc(df_test$Revenue, y_pred)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
auc(ROC)
```

```
## Area under the curve: 0.7664
```

```
# results matrix
models = c('Base logit model', 'Logit model with Lasso regression', 'Optimized logit model')
features = c(27, 3, 3)
accuracy = c(0.89, 0.89, 0.90)
auc = c(0.68, 0.68, 0.77)
df_results = data.frame(models, features, accuracy, auc)
df_results
```

```
##                              models features accuracy  auc
## 1                  Base logit model       27     0.89 0.68
## 2 Logit model with Lasso regression        3     0.89 0.68
## 3             Optimized logit model        3     0.90 0.77
```