Online shoppers analysis

Stanislaw Czekalski, Szymon Puszcz

In this document we are going to analyze online shoppers data.

It describes 12,330 user sessions browsing online shop. Dataset comes from UCI Machine Learning Repository. Observations are associated with labels indicating whenever current session ended up with consumer buying something or not. Primary task that can be done using this data is to perform classification of sessions, but the aim of our analysys is to explore the data and get knowledge about users' behaviour.

Raw data looks like this:

```r
df = read.csv("online_shoppers_intention.csv")
head(df, 5)
```

```
##   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
##   ProductRelated ProductRelated_Duration BounceRates ExitRates PageValues
## 1              1                0.000000        0.20      0.20          0
## 2              2               64.000000        0.00      0.10          0
## 3              1                0.000000        0.20      0.20          0
## 4              2                2.666667        0.05      0.14          0
## 5             10              627.500000        0.02      0.05          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
##         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
```

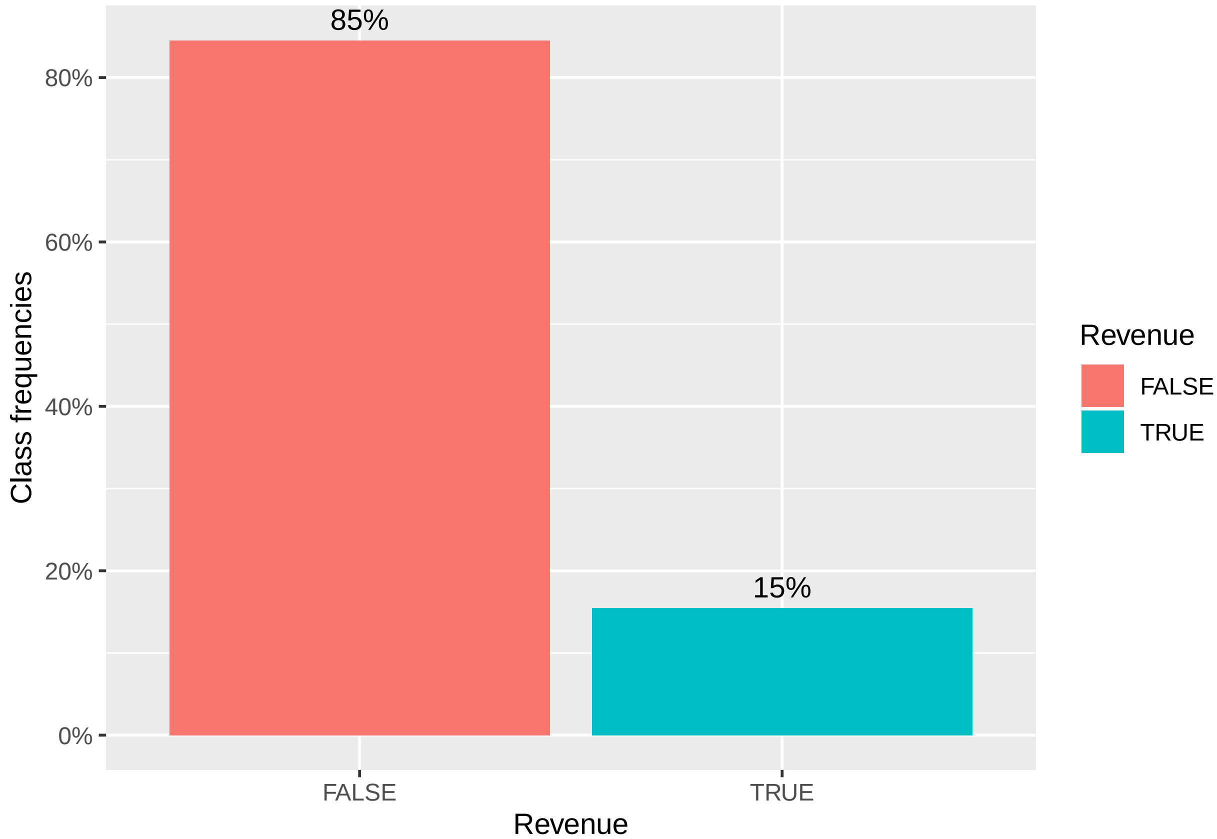
We can see that variables of different types, some of them are categorical. We will set their types to factors.

We should always check for missing values, executing any(is.na(df)) comes in handy.

```
## [1] FALSE
```

There is no missing values in this data.

First thing that must be done is checking class distribution. Most frequently, we do it using bar plot. As one can see, we deal with a problem of imbalanced class distribution.



Let's print summary of variables' distributions.

```
##  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
##
##
##
##
```

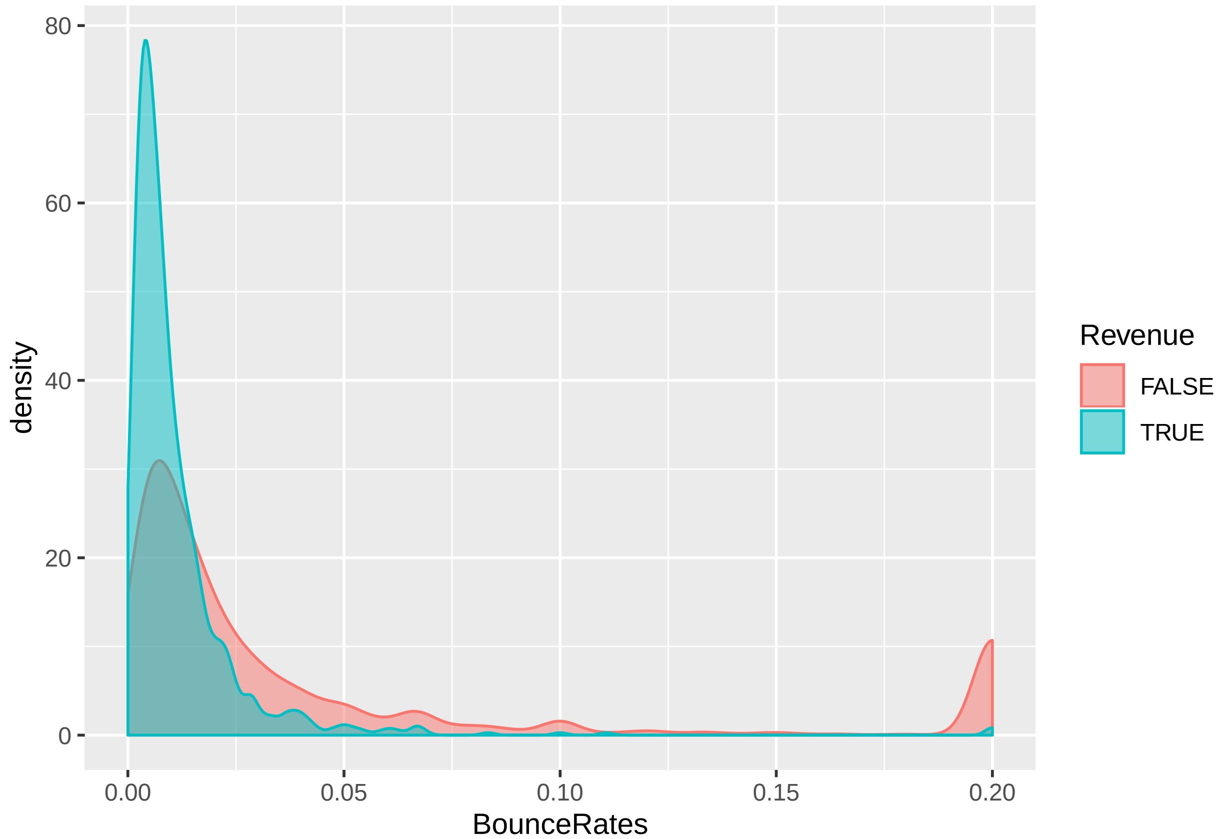
We can see that most user visit parts of the website that are product related. They also spend the most time on them. "**Bounce Rate**", "**Exit Rate**" and "**Page Value**" are somehow misleading names, after looking up we may discover that they are related to Google Analytics names.

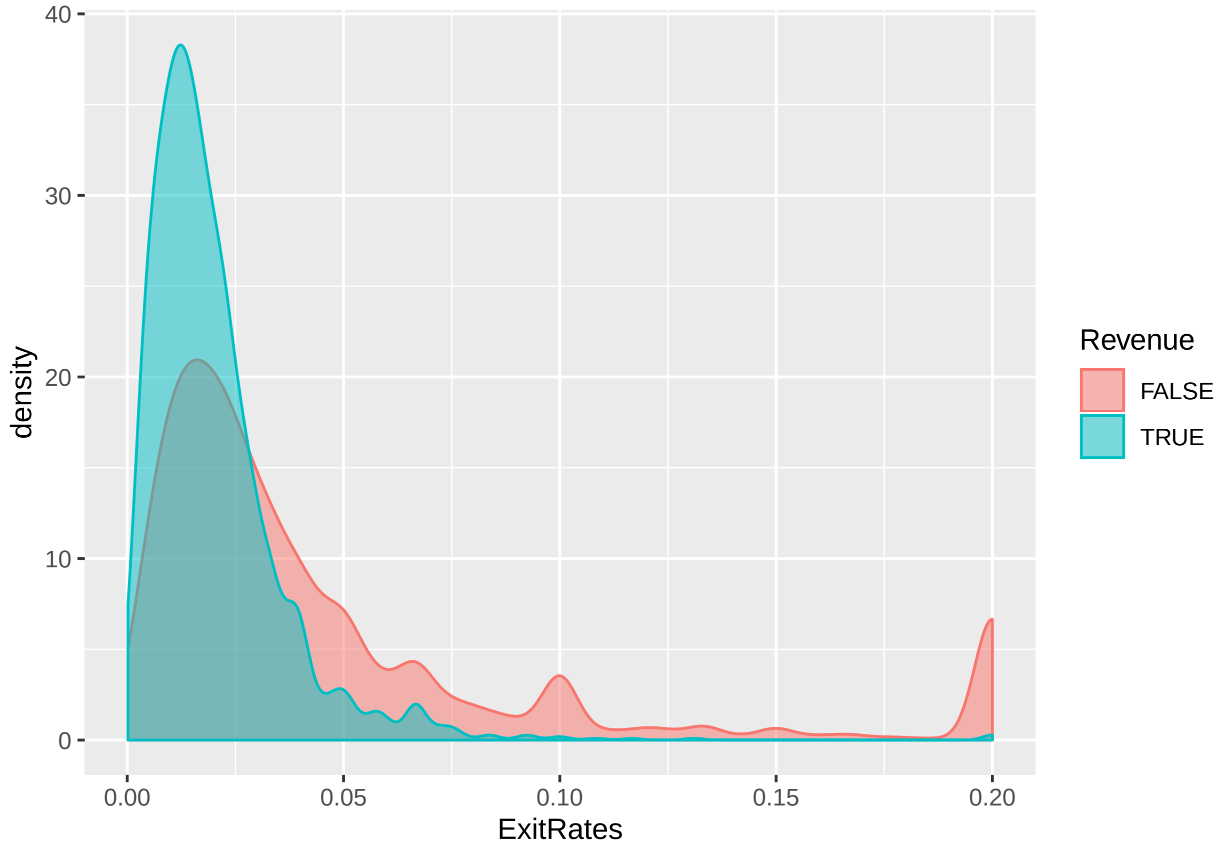
"**Bounce Rate**" describe percentage of visitors that come from Google Analytics, enter the site and then leave ("bounce") without triggering any other requests to the analytics server during that session.

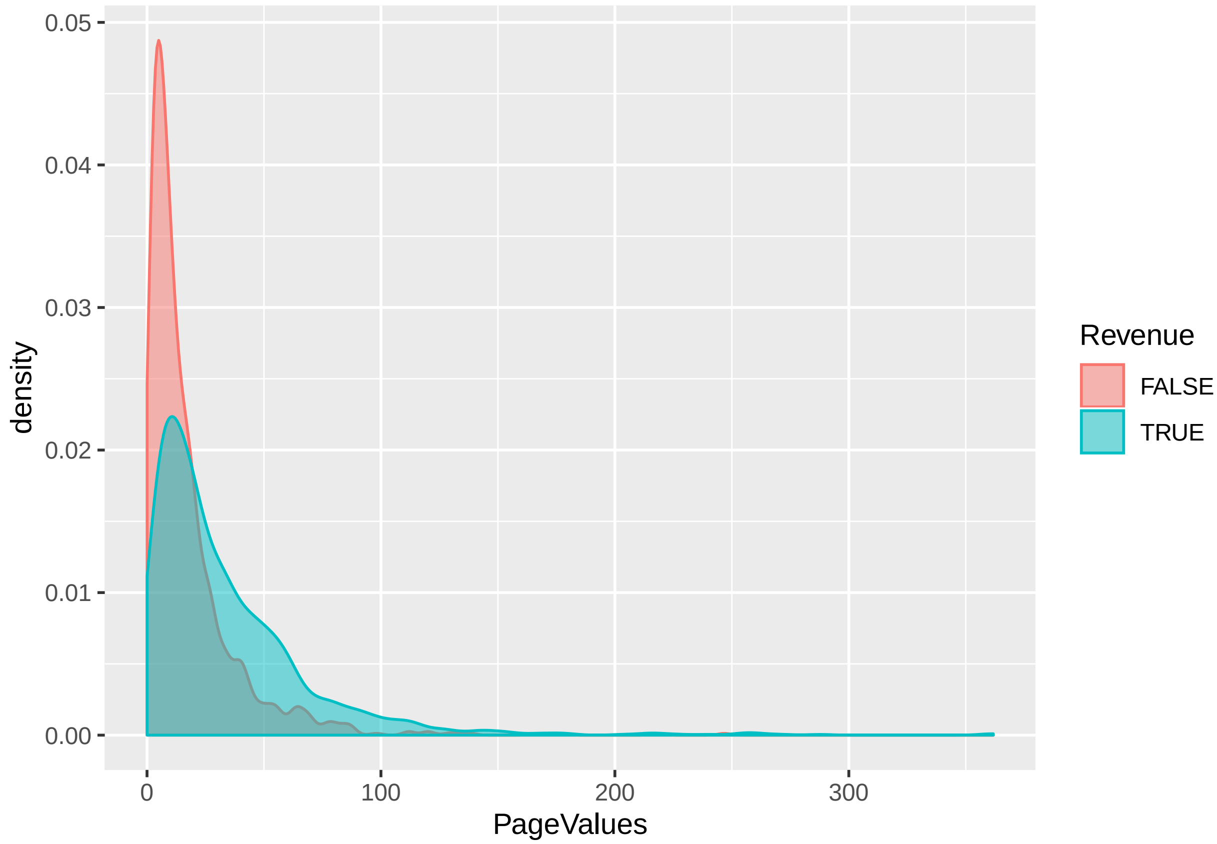
"**Exit Rate**" feature for a specific web page is calculated as for all pageviews to the page, the percentage that were the last in the session.

The "**Page Value**" feature represents the average value for a web page that a user visited before completing an e-commerce transaction. Values of all "Bounde Rate" and "Exit Rate" are quite low. To furhter investigate this features we can plot their distributions.

Most of values for all three of them are zeros, so we will only plot distribution of non zero values to have a closer look. Additionaly, we decided to split each distribution to two, depending on classes.

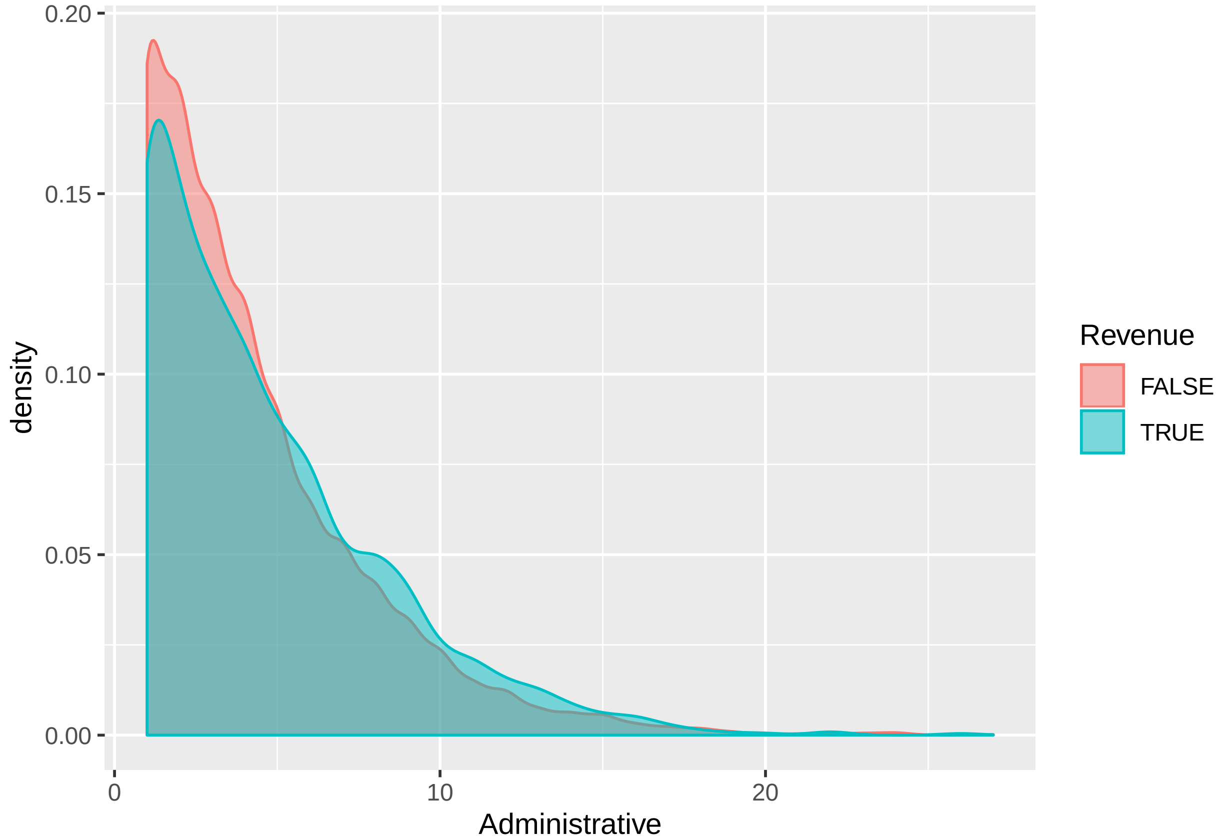

80
60
40
20
0
density
0.00
0.05
0.10
0.15
0.20
BounceRates
Revenue
FALSE
TRUE


40
30
20
10
0
density
0.00
0.05
0.10
0.15
0.20
ExitRates
Revenue
FALSE
TRUE

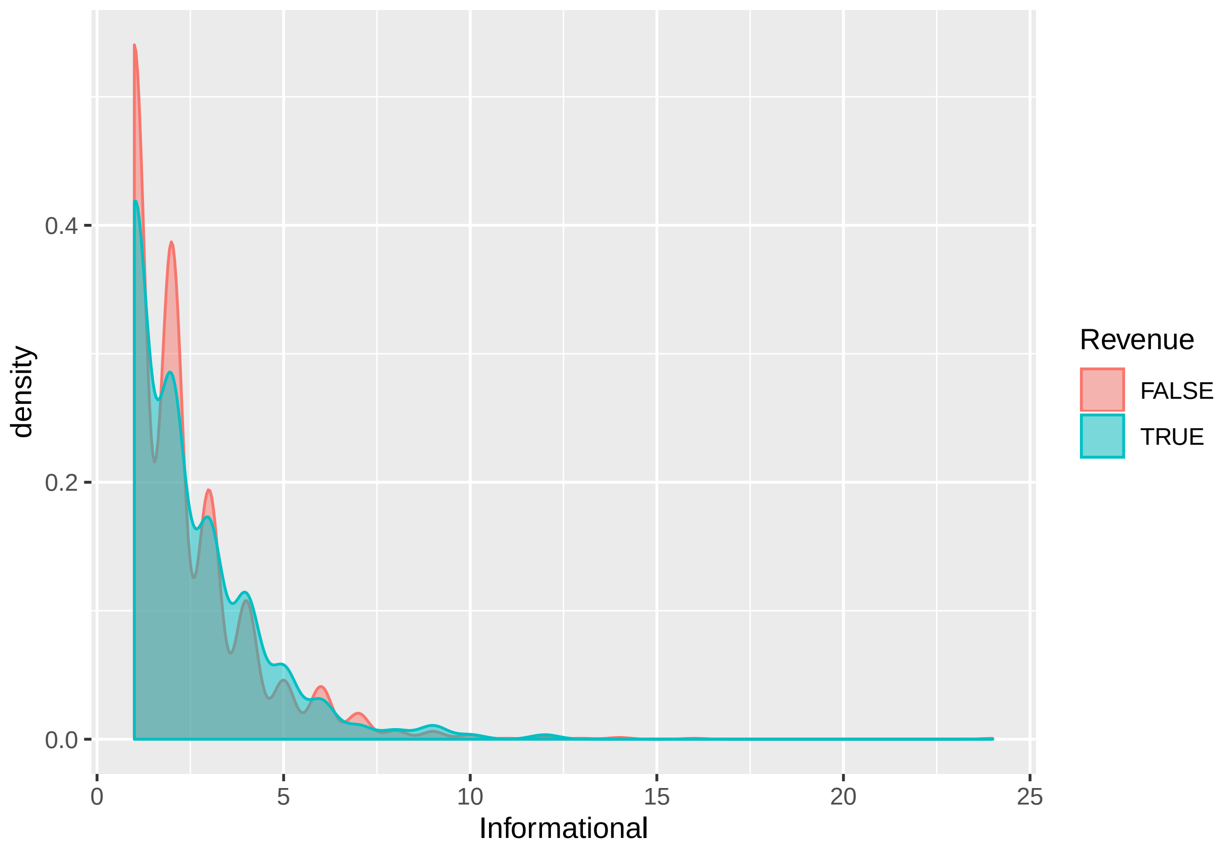


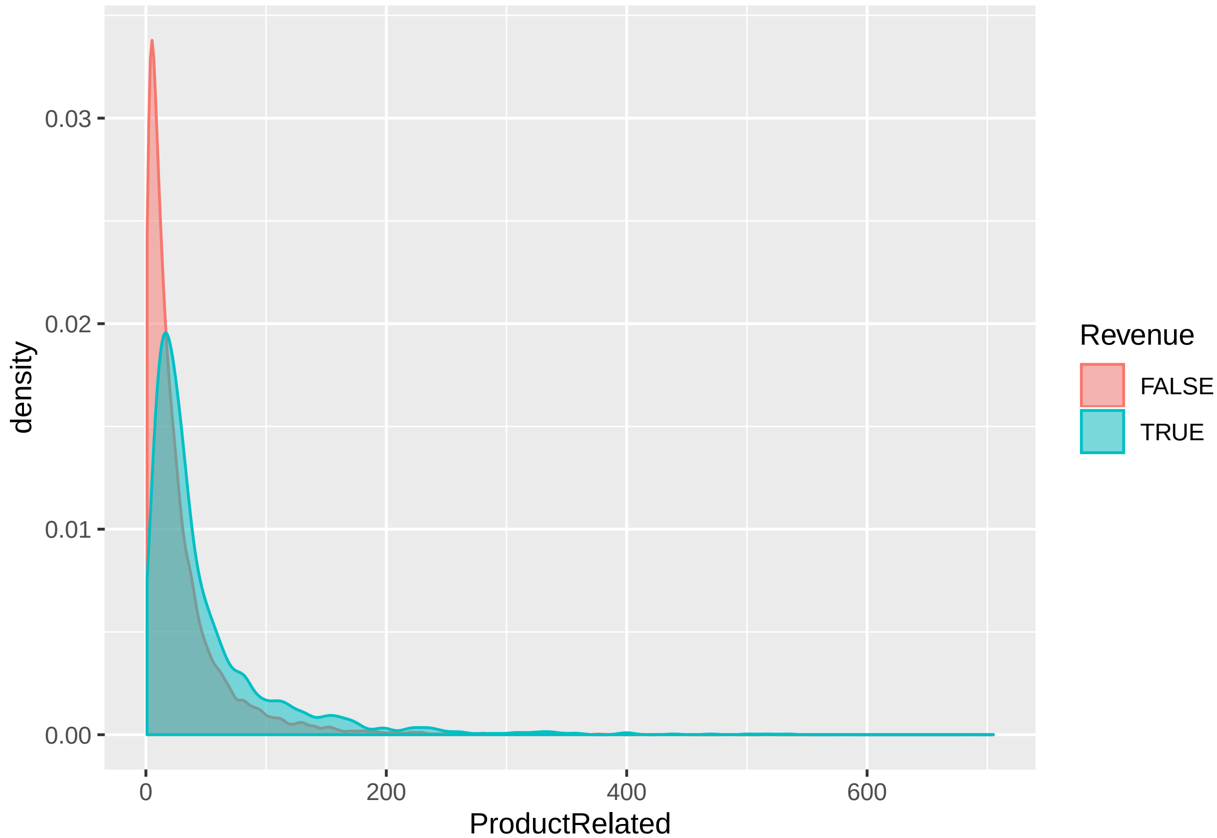
We can see that distributions of all three features differ depending on classes. Especially, all observations having bounce rates and exit rates 0.20 belong to negative class. Distribution of page values is more skewed towards high values for positive class. It means that all three variables might be usefull for building classification model.

Next, we will take a look at times the user spend on different parts of the website.

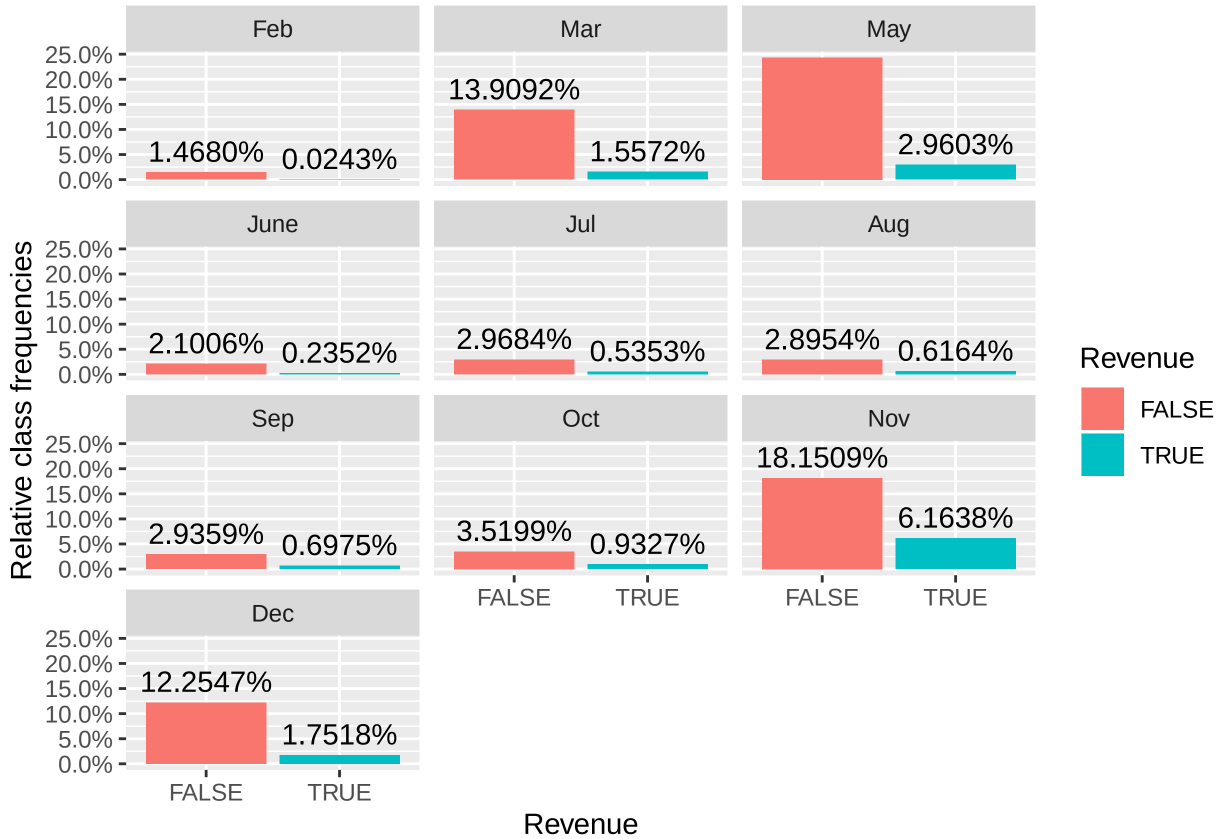


0.20
0.15
0.10
0.05
0.00
density
0
10
20
Administrative
Revenue
FALSE
TRUE

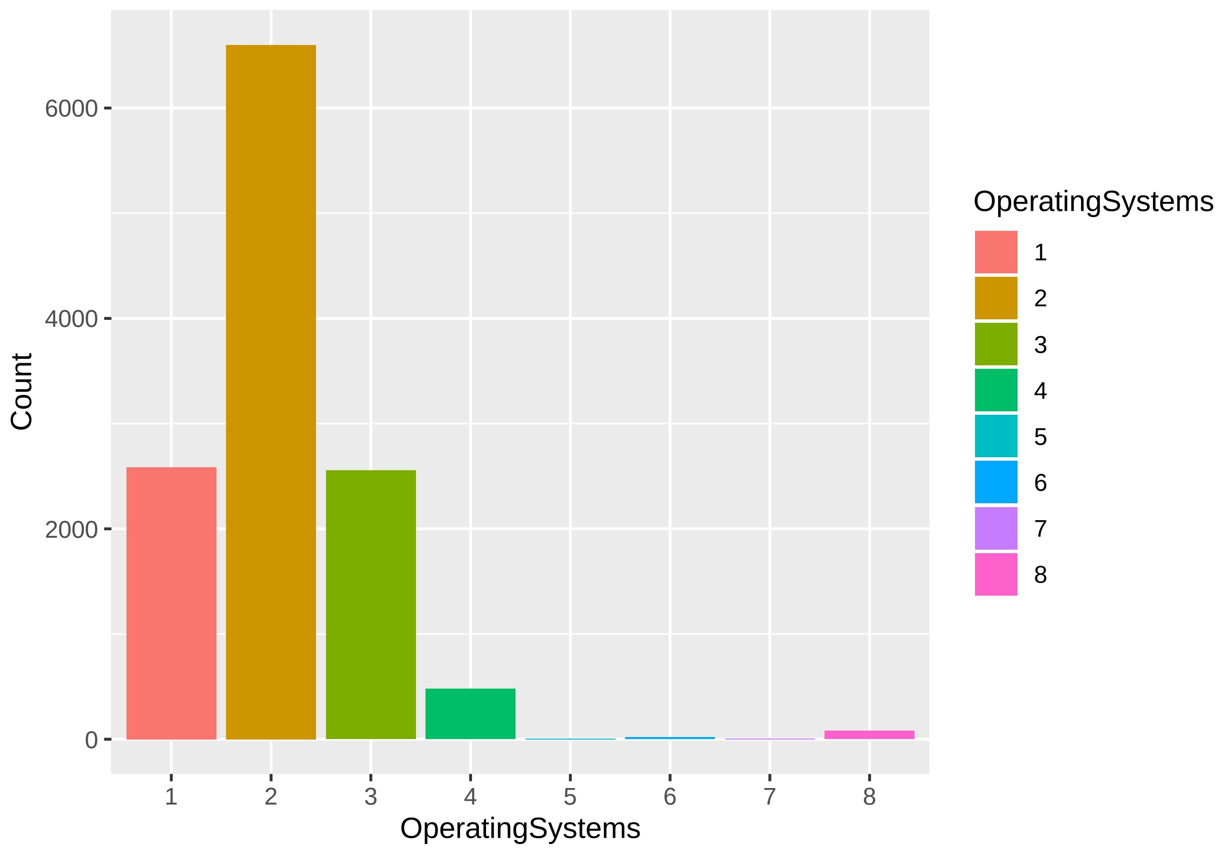

density
0.4
0.2
0.0
0
5
10
15
20
25
Informational
Revenue
FALSE
TRUE

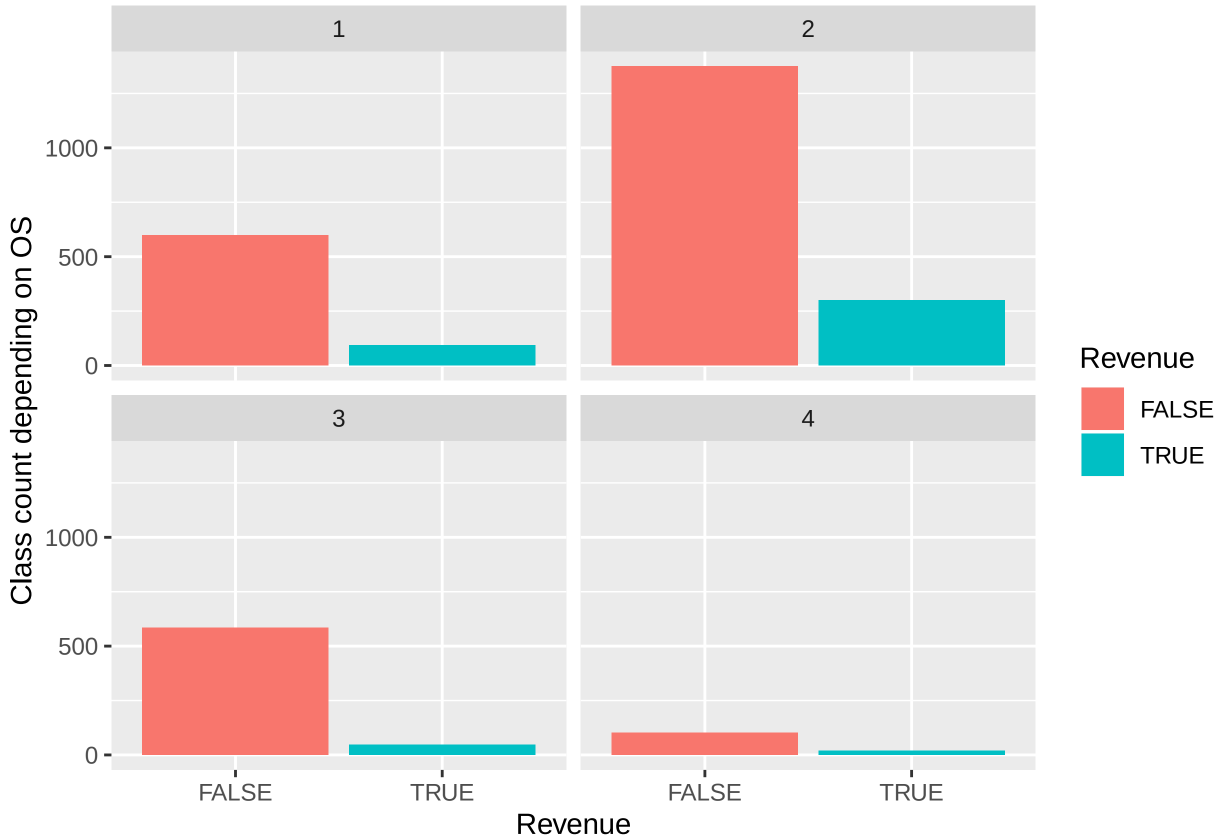


As one can see, there is no much difference bitween times spend on parts of website depending on class. This raw features might not be useful for prediction.

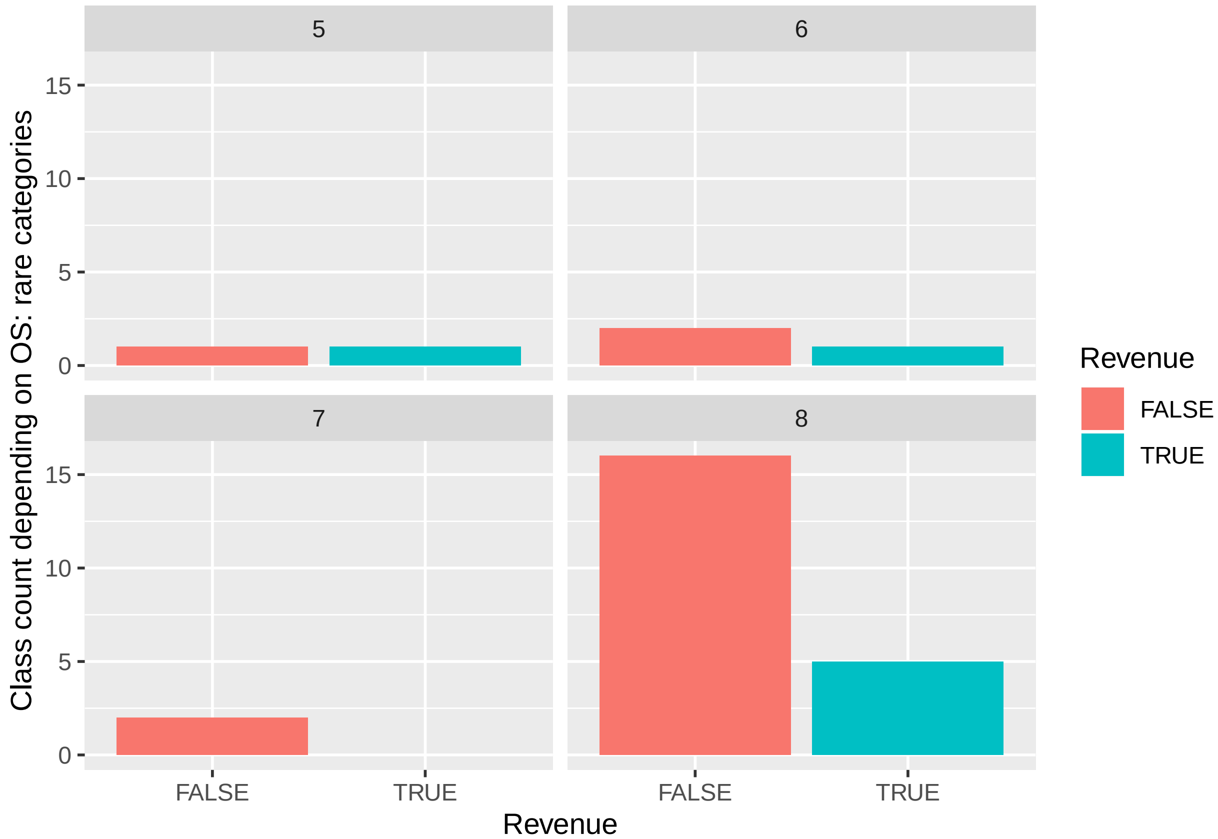


Let's look at month information. Data is not evenly distributed between months, for some of them we have very little information. For instance, classification model might learn, that there is no point predicting a purchase for session in June, because chances are that no session ending with a purchase from this month will be part of our training set. Using this feature might cause overfitting.

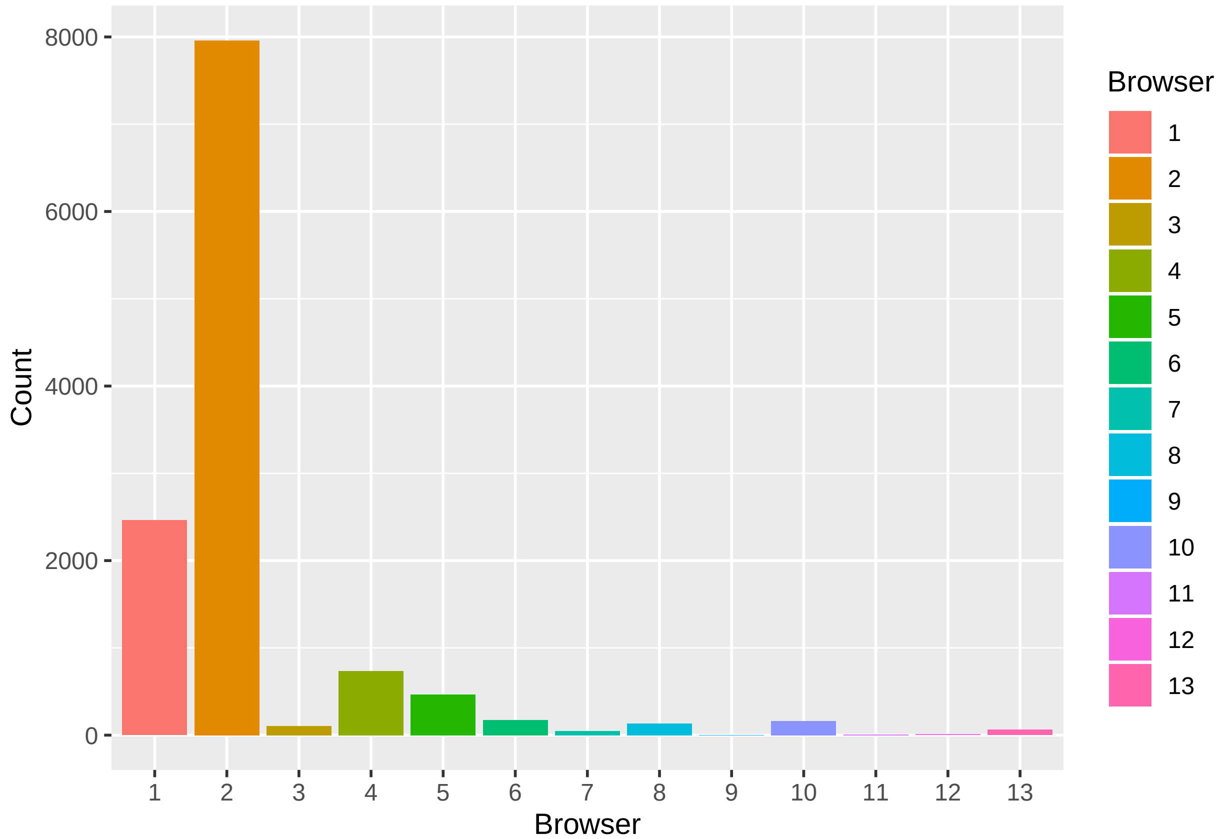

Count
6000
4000
2000
0
1
2
3
4
5
6
7
8
OperatingSystems
OperatingSystems
1
2
3
4
5
6
7
8

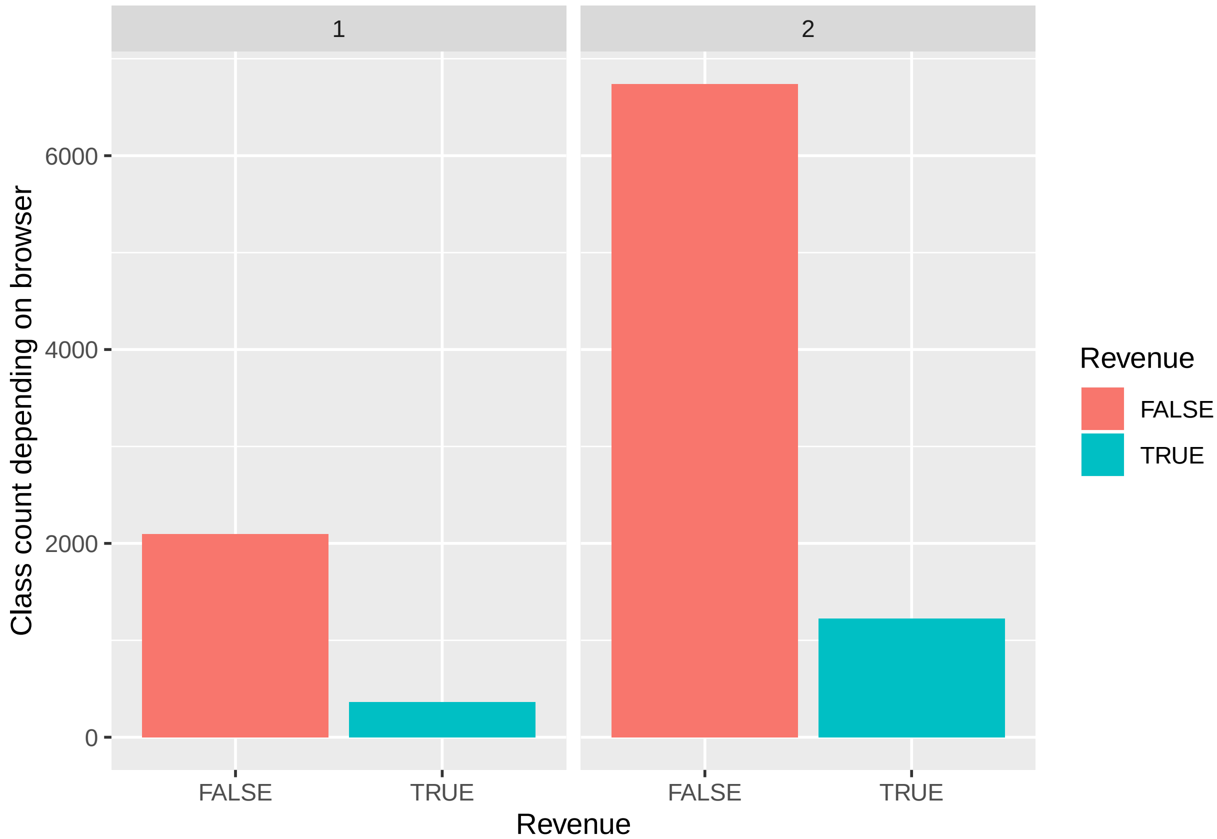


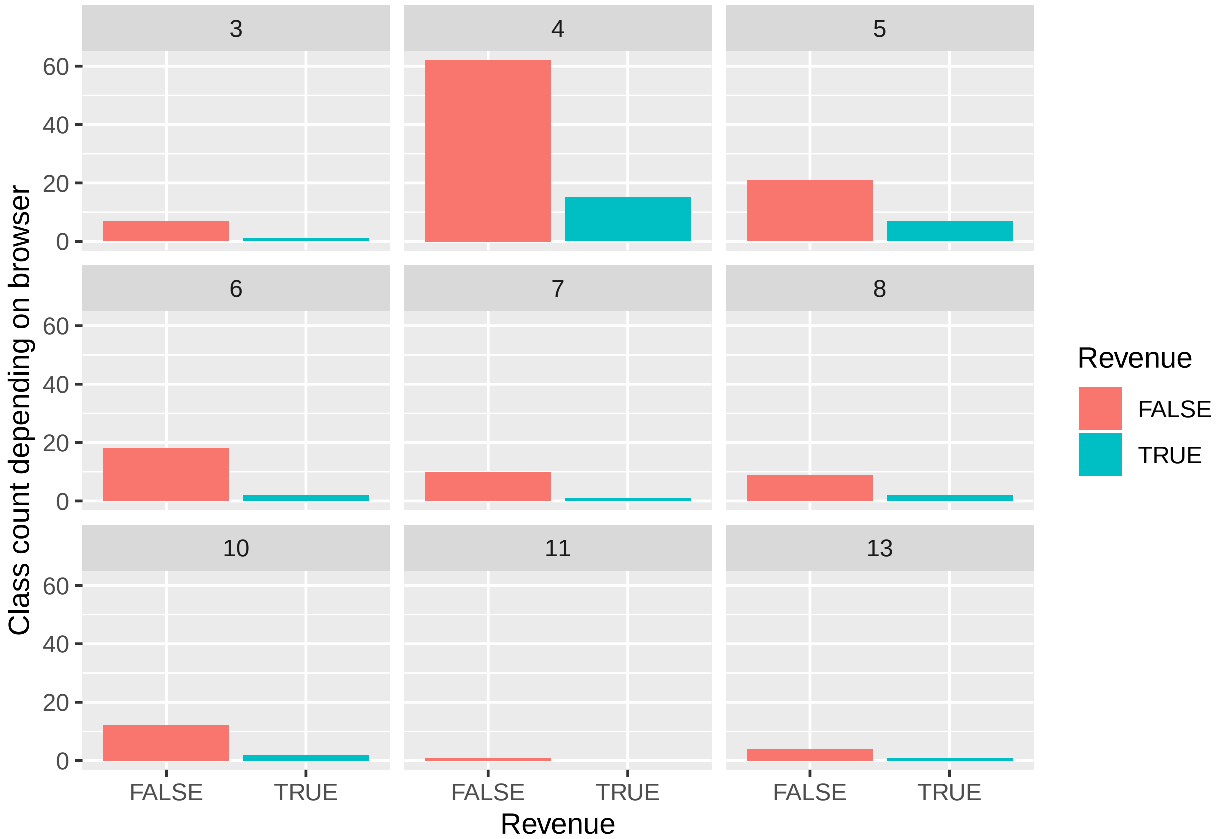
1
2
3
4
1000
500
0
Class count depending on OS
FALSE
TRUE
Revenue



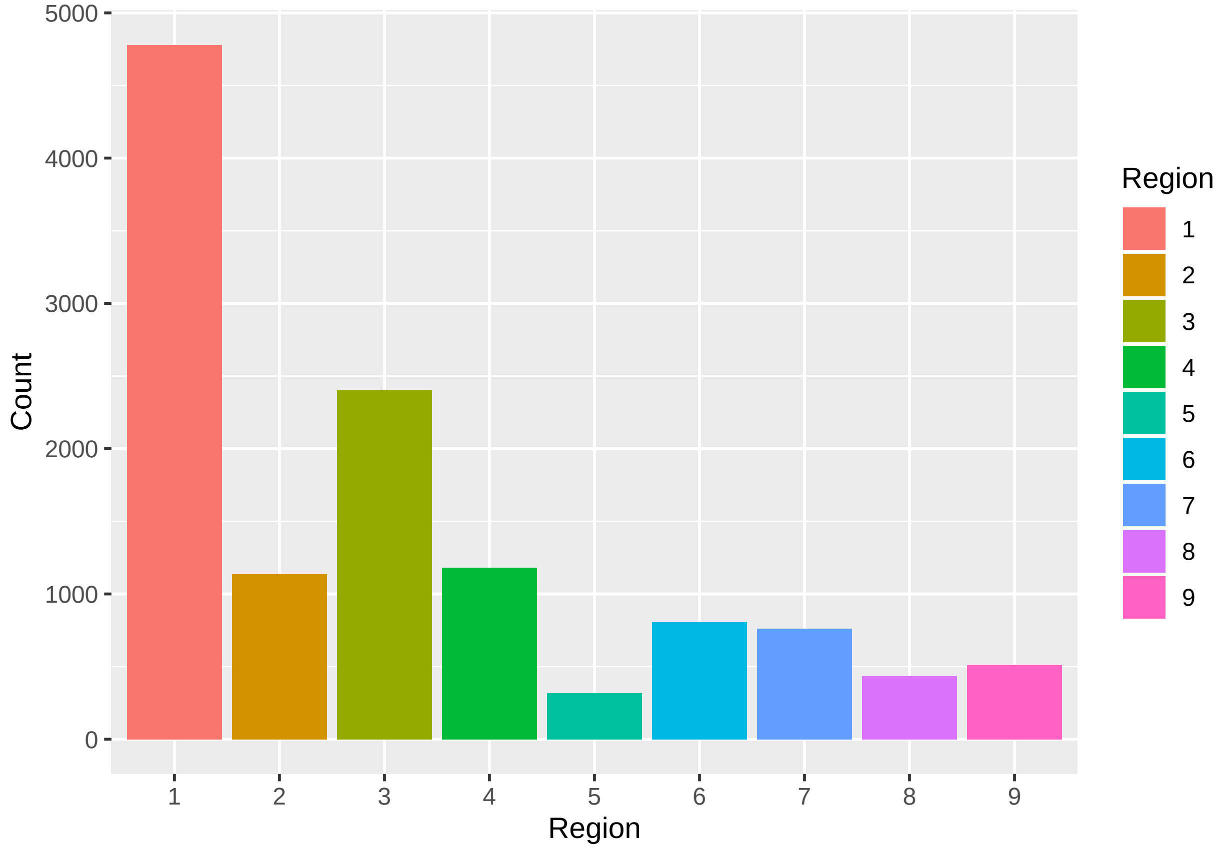
There is 8 kinds of operating systems used by our users. Some of them are much more popular, and some of them are very rare. It may make sense to group all rare categories into one, because alone they are not very informative. We may wrongly conclude that people using OS number 5 buy products more often than other users, but we need to be careful, that little data can be misleading.

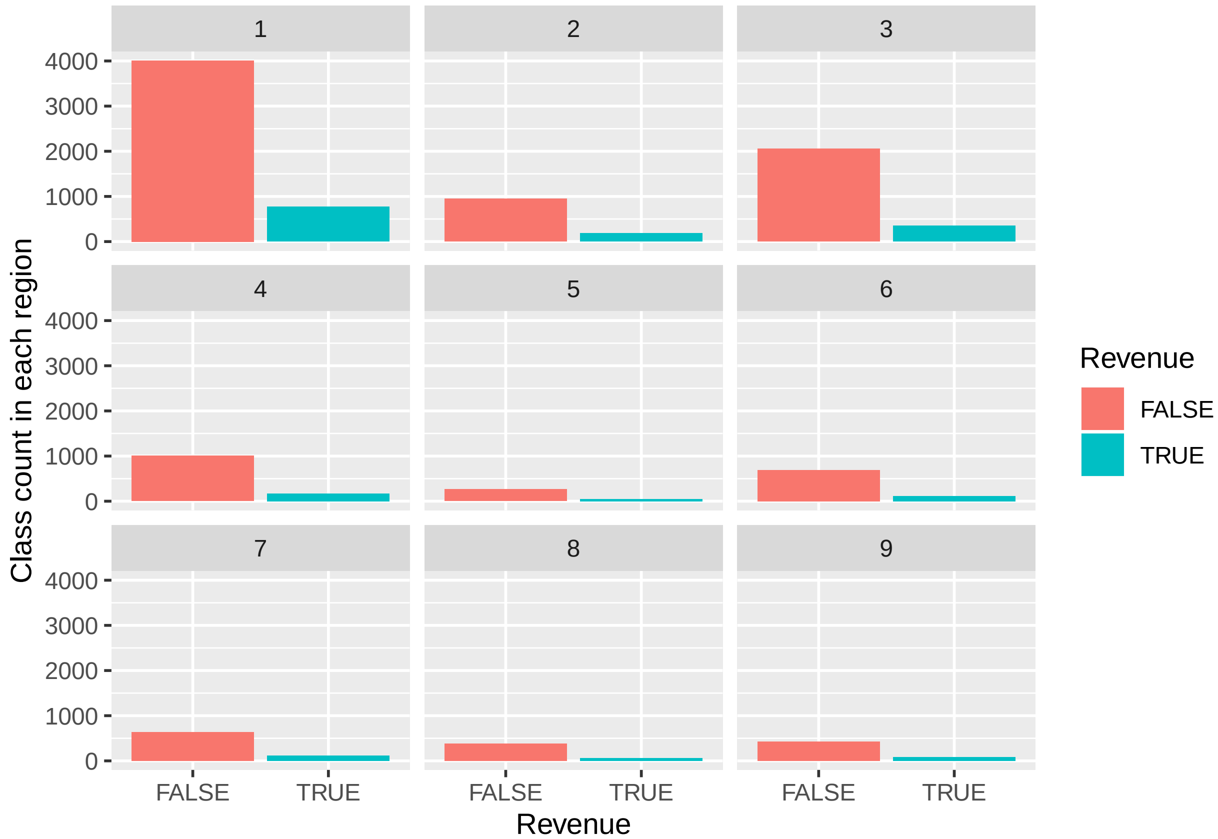

Browser
1
2
3
4
5
6
7
8
9
10
11
12
13
Count
8000
6000
4000
2000
0
1 2 3 4 5 6 7 8 9 10 11 12 13
Browser


1
2
6000
4000
2000
0
Class count depending on browser
FALSE
TRUE
FALSE
TRUE
Revenue
Revenue
FALSE
TRUE

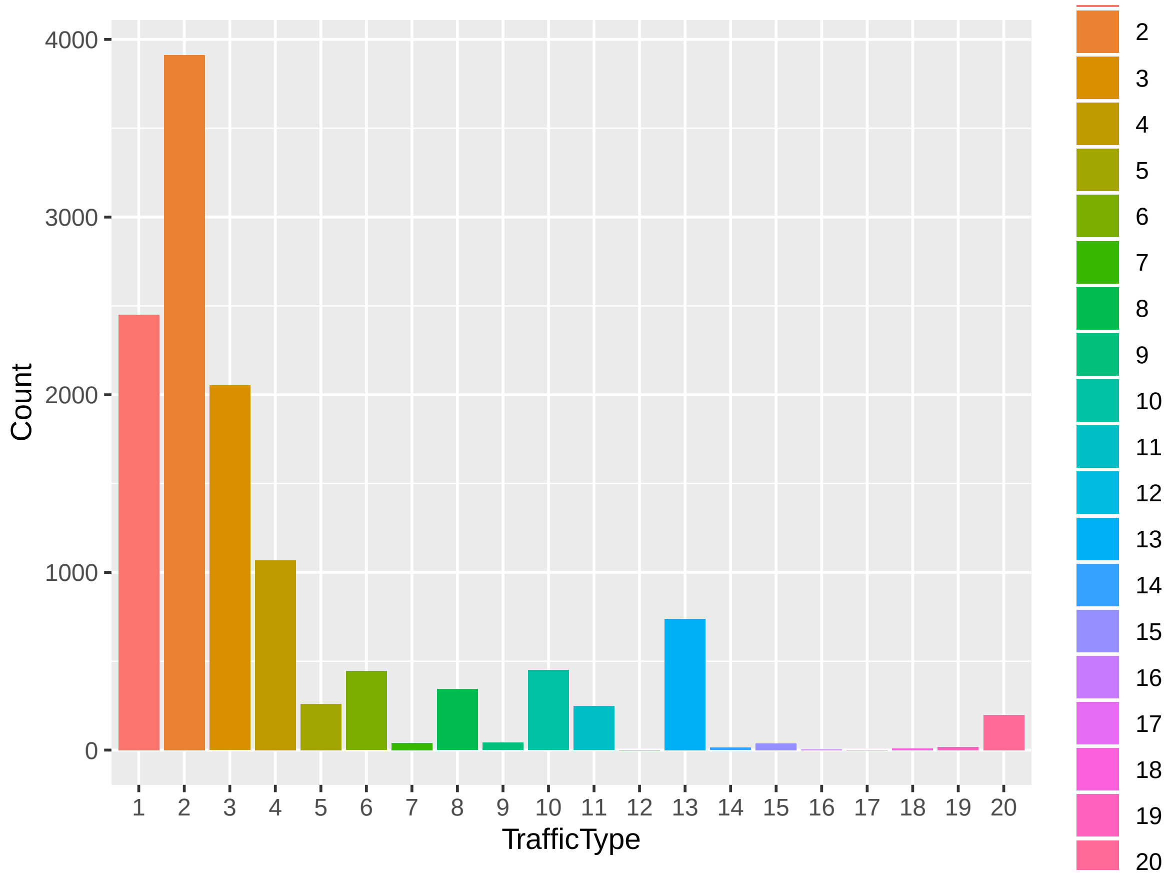


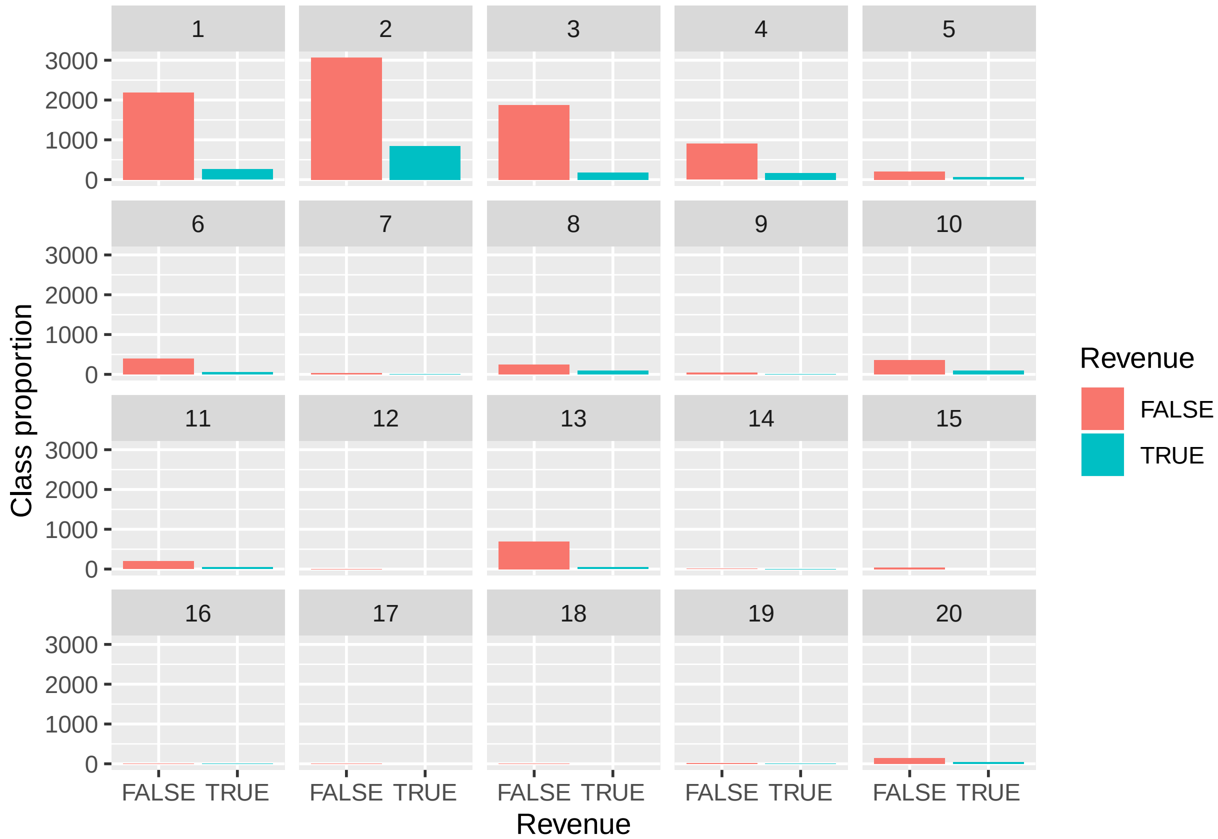
In case of browsers, we have two main categories, and quite a few less popular ones. We draw class distribution in each category to check if we may infer some class information depending on browser type. Once again, grouping rare categories might be desirable


Count
5000
4000
3000
2000
1000
0
1
2
3
4
5
6
7
8
9
Region
Region
1
2
3
4
5
6
7
8
9

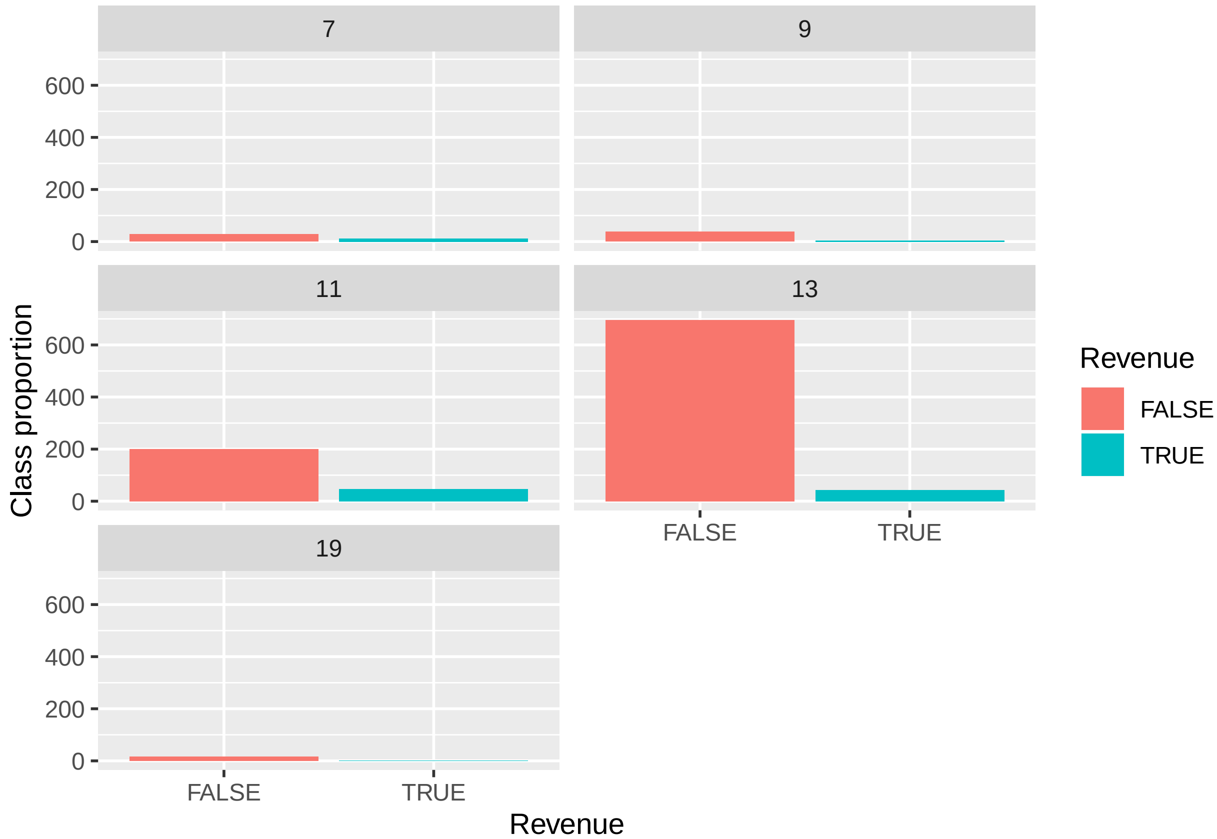


Region distribution is more even compared to OS and browser. Still, class distribution in each region category doesn't seem to be very helpful in classification.


4000
3000
2000
1000
0
Count
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
TrafficType
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

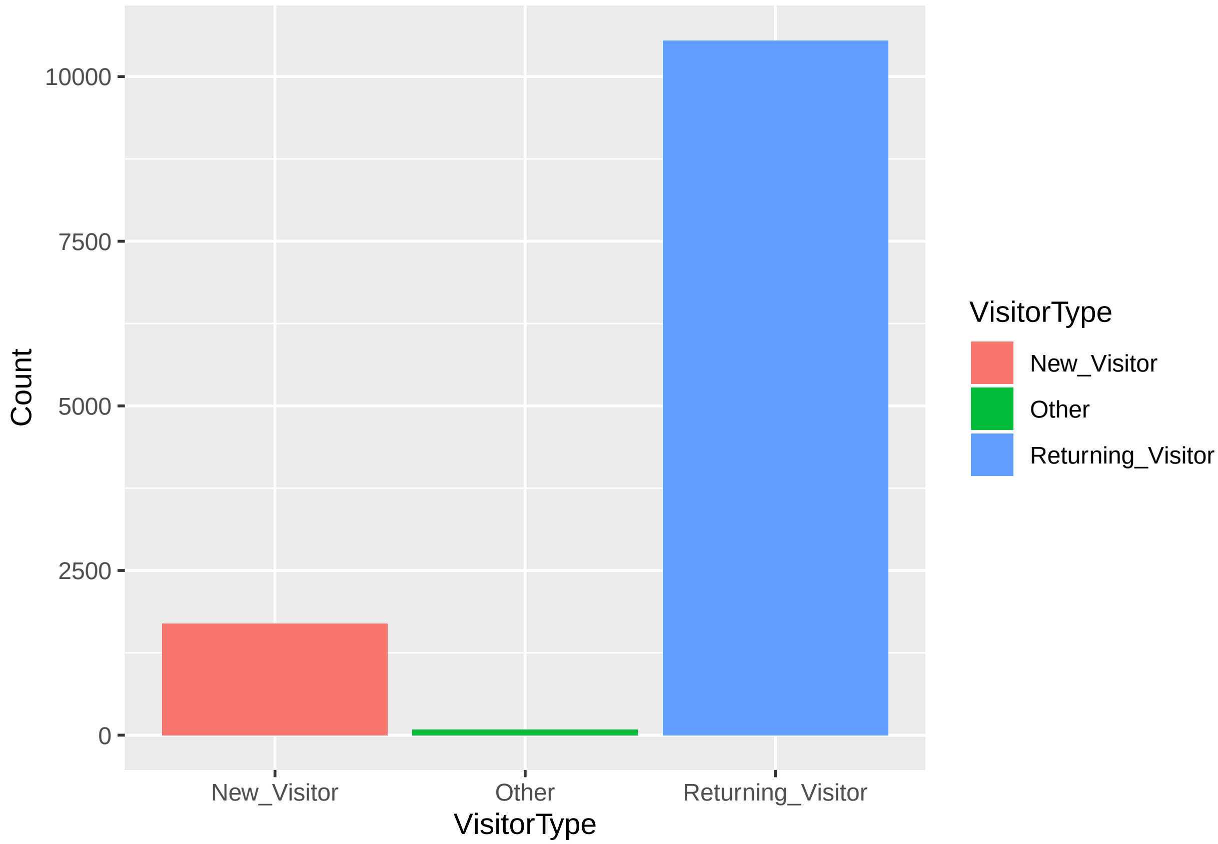


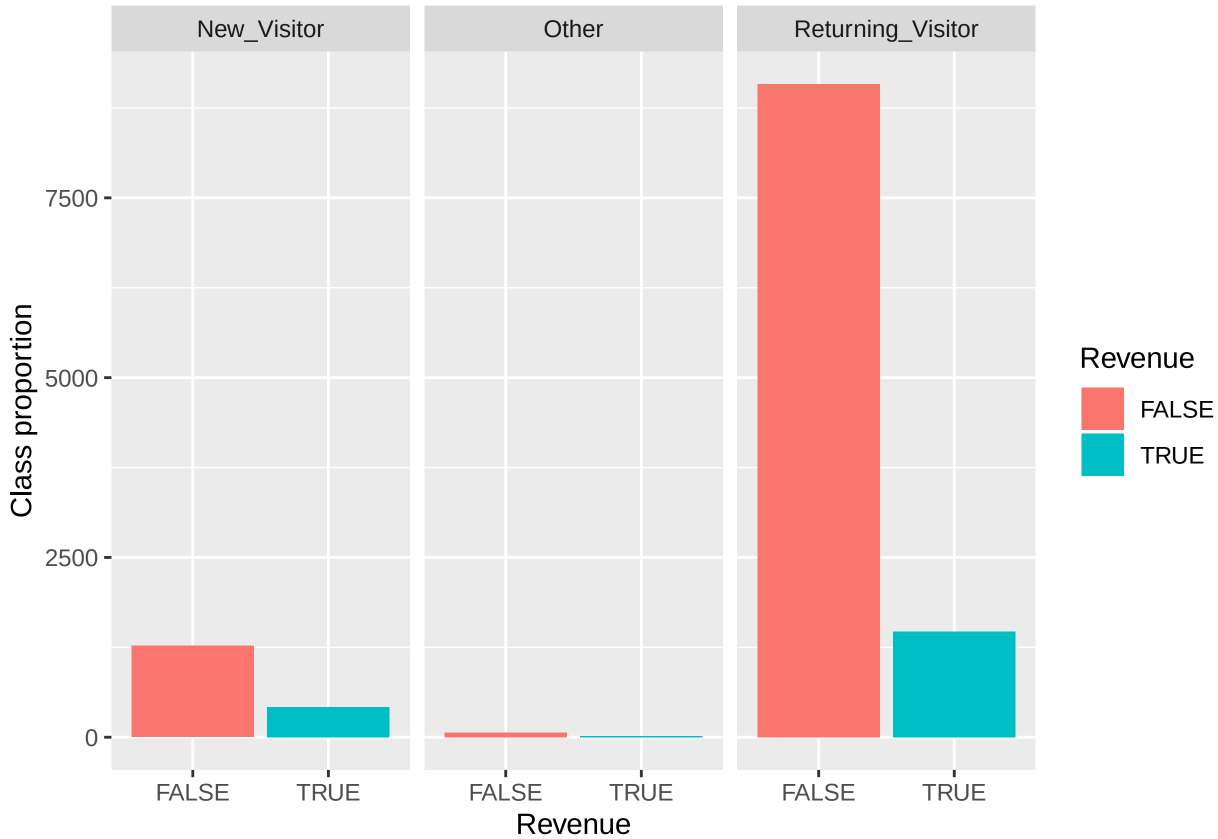
Traffic type distribution is very uneven. This variable might be interesting, especially rare categories might represent untypical user, i.e. administrator or developer. We take a closer look below.



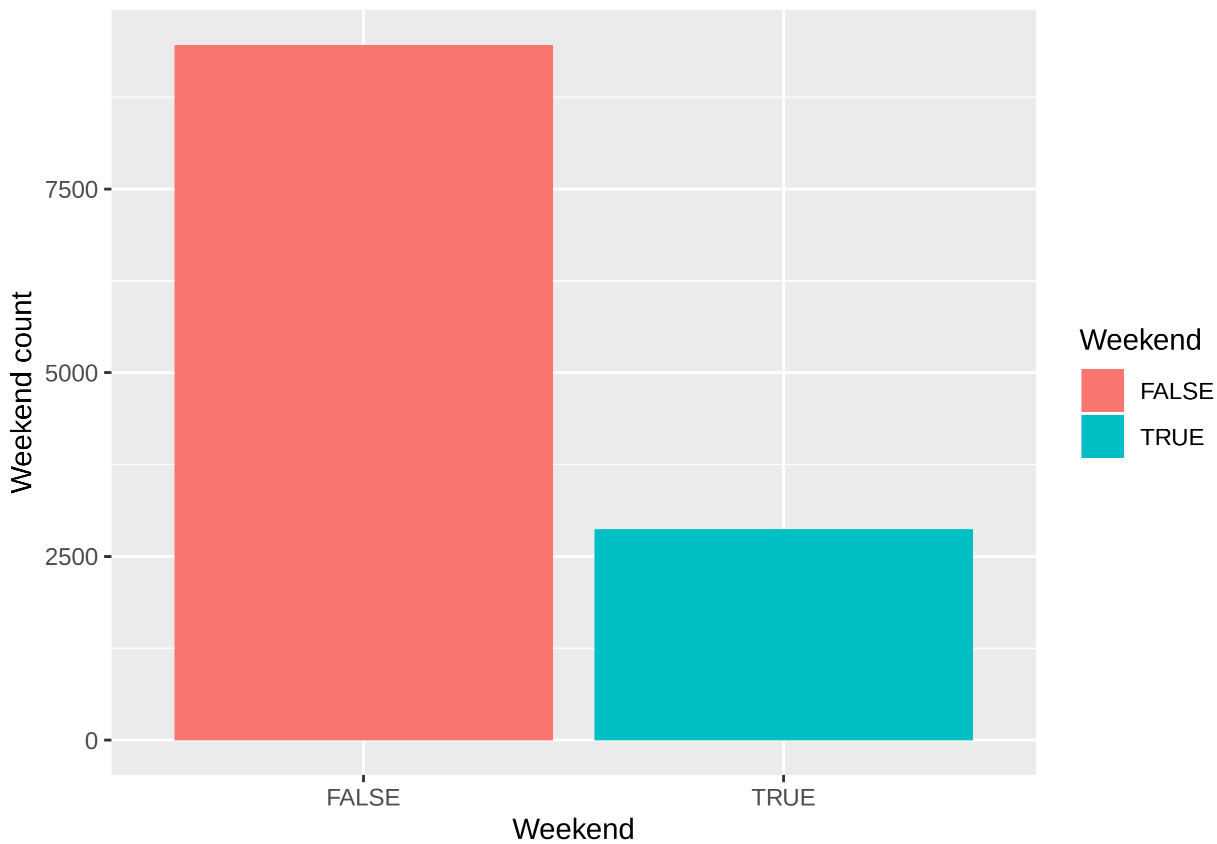
7
9
11
13
19
600
400
200
0
Class proportion
FALSE
TRUE
Revenue
Revenue
FALSE
TRUE

It is hard to reason what some traffic types might represent due to values annonymization. If we had access to description of individual levels of this feature we could discover that some traffic types come from people that are not typical users, and we are not interested in their behaviour. In this case, such observations should be removed from the dataset.

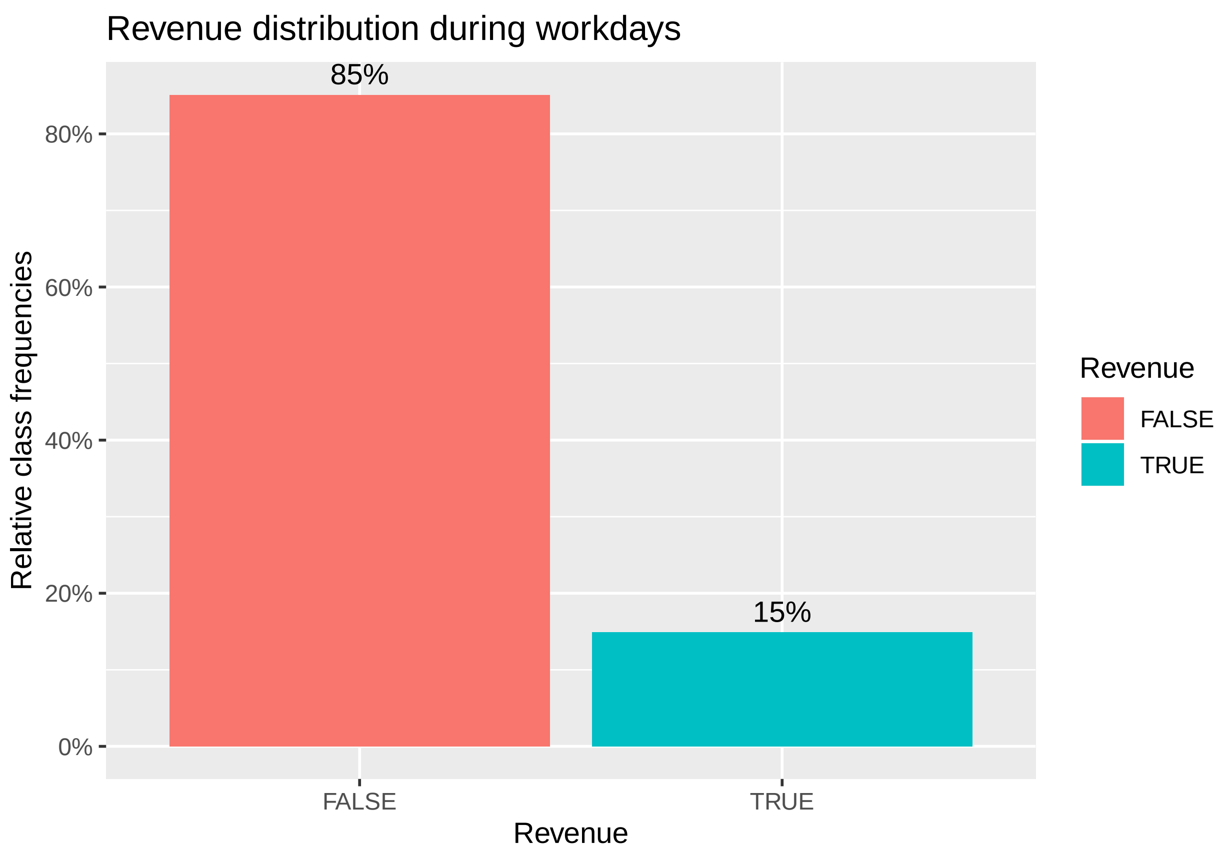

Count
10000
7500
5000
2500
0
New_Visitor
Other
Returning_Visitor
VisitorType
VisitorType
New_Visitor
Other
Returning_Visitor

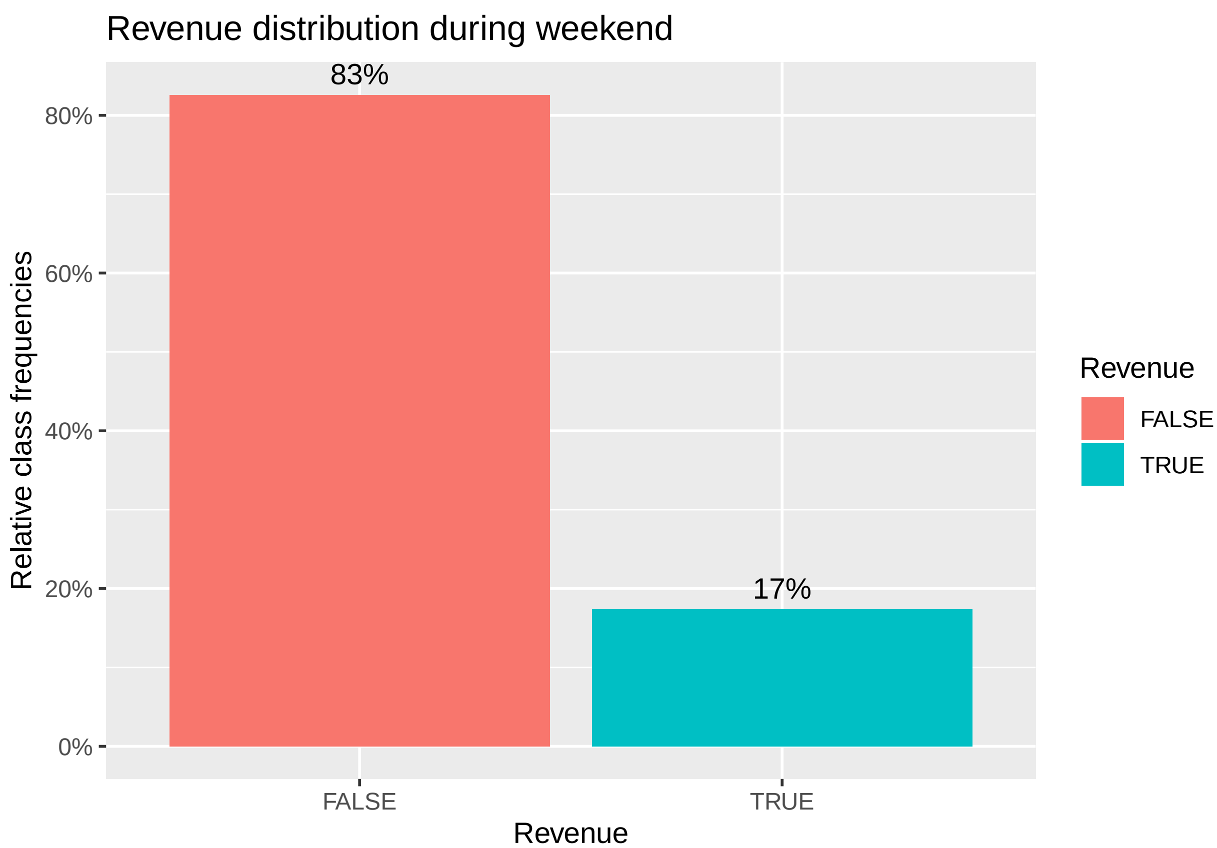


Most of the visitors are returning ones. Category "Other" looks suspicious, in this case we can be pretty sure it represents some abnormal type of users, like administrator or Google crowler, that is not likely to buy something!



7500
5000
2500
0
Weekend count
FALSE
TRUE
Weekend
Weekend
FALSE
TRUE


Revenue distribution during workdays
85%
15%
80%
60%
40%
20%
0%
Relative class frequencies
FALSE
TRUE
Revenue
Revenue
FALSE
TRUE



We can see that users that came to the website during the weekend are a little bit more likely to buy something. During the week rate of buyers is about 17.5%, and for the weekend its 20%. It is not a suprise, during the weekend people usually have more time for shopping.

Random Forest proximity

In order to get a better understanding of our data, we may try to capture its structure. As we've observed our data consists of variables of different types: we have both numeric and categorical features, so it is difficult to come up with a metric suitable for clustering. What we can do instead is to train Random Forest, and get pairwise similarities between objects by counting how many times objects end up in the same leaf node, and use this proximity matrix to perform Multidimensional Scalling, that would create two-dimensional embediing of our data in some abstract space.

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## ntree      OOB      1      2
##   100:  10.42%  4.32% 43.72%
```

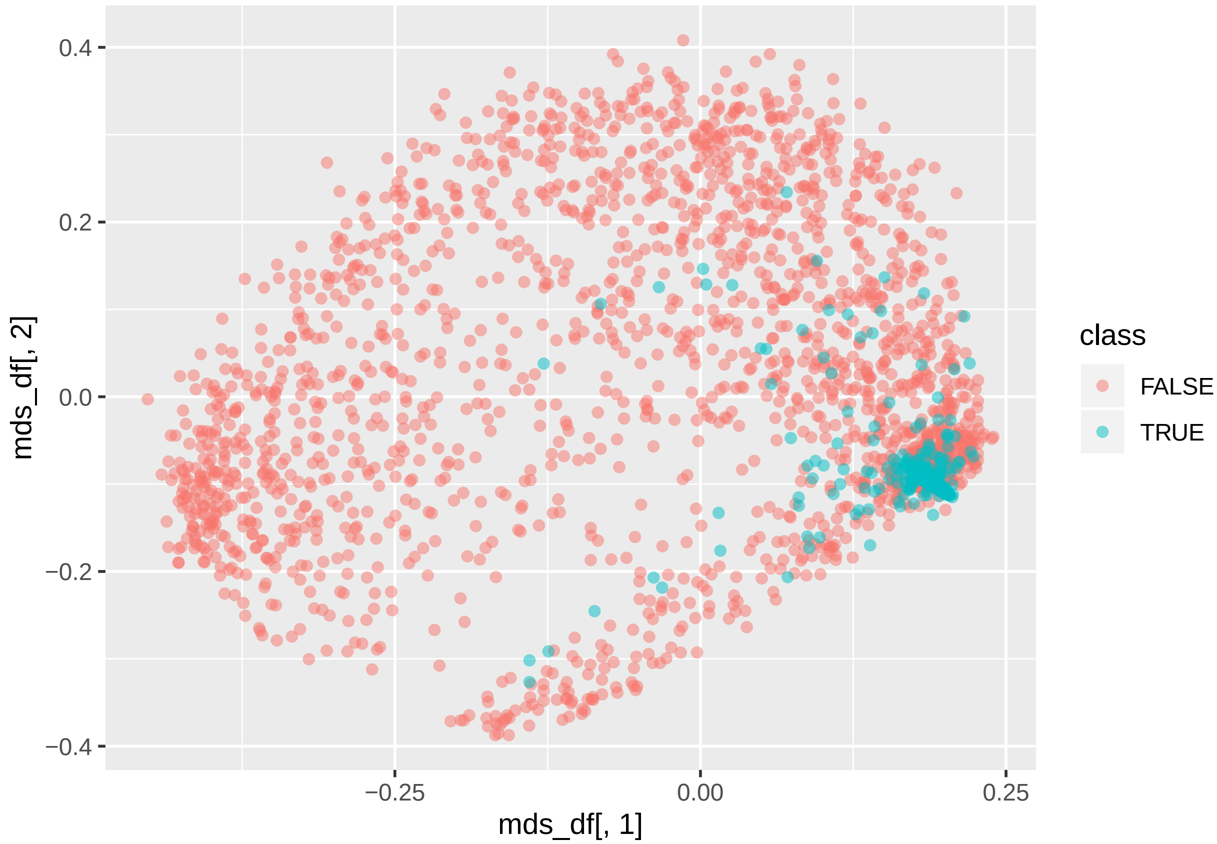
```
##
## Call:
##  randomForest(x = sample_df[, 1:17], y = factor(sample_df$Revenue),      ntree = 100, importance = TI
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 10.42%
## Confusion matrix:
##       FALSE TRUE class.error
## FALSE  1994   90  0.04318618
## TRUE    167  215  0.43717277
```

OOB estimate of error suggests that we can trust the model.

Multidimensional Scalling

```
## Loading required package: HSAUR2
```

```
## Loading required package: tools
```



Some structure of our data was obtained. We can see dense cluster of sessions that finished whith purchase, and a whole bunch of other sessions.

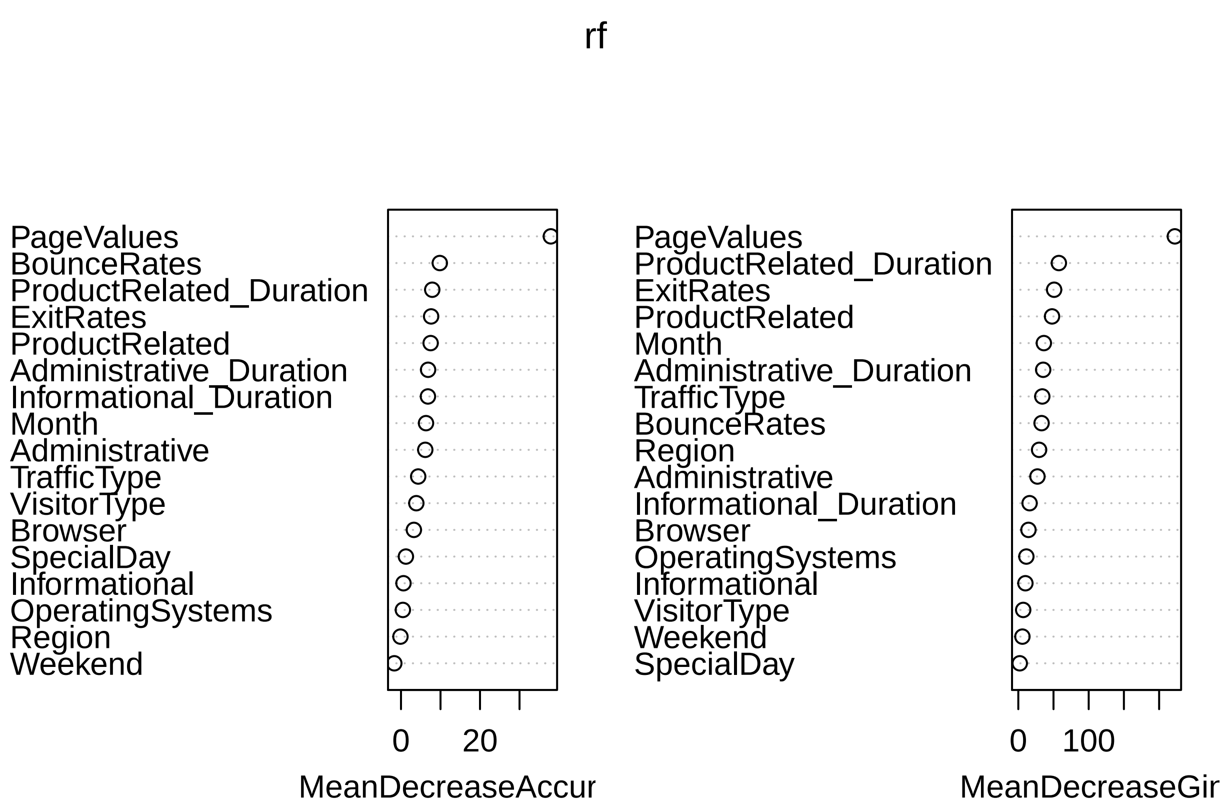
Let's check which features our model considers important.

```r
importance(rf)
```

```
##                             FALSE       TRUE MeanDecreaseAccuracy
## Administrative          7.2833083 -2.2376903            6.1463717
## Administrative_Duration 7.8096639 -2.8541962            6.8596922
## Informational           0.2424285  0.9167772            0.6282263
## Informational_Duration  6.9300381  1.1572433            6.8184807
## ProductRelated          6.2414007  4.3393690            7.5058263
## ProductRelated_Duration 6.0124341  3.4621638            7.9104201
## BounceRates             5.3841840  7.2792814            9.8232657
## ExitRates               4.7386466  8.5328719            7.6300554
## PageValues             24.9564385 49.7418050           37.9208754
## SpecialDay              1.0071056  0.8480417            1.2469435
## Month                  -0.8416249 11.0117368            6.3310010
## OperatingSystems        0.8480532 -0.4406735            0.4778932
## Browser                 3.8181788 -0.1389026            3.2648894
## Region                  0.1430039 -0.4568362           -0.1356667
## TrafficType             1.5967426  4.4255028            4.3538523
## VisitorType             2.3313544  3.0948832            3.8826321
## Weekend                -1.2605801 -0.9735546           -1.6895754
##                        MeanDecreaseGini
```

```
## Administrative                   27.170791
## Administrative_Duration          35.342528
## Informational                    10.074410
## Informational_Duration           16.037100
## ProductRelated                   47.921888
## ProductRelated_Duration          57.592759
## BounceRates                      33.051691
## ExitRates                        50.864484
## PageValues                      222.333923
## SpecialDay                        2.068618
## Month                            36.283181
## OperatingSystems                 11.416742
## Browser                          14.440547
## Region                           29.552057
## TrafficType                      34.054101
## VisitorType                       6.974793
## Weekend                           5.810573
```

```r
varImpPlot(rf)
```



Page Values seem to be really important, along with time spend on product-related page, laso exit rate and some other features. Month information might be a little bit confusing - our data is not very representative when it comes to this feature.

We will try Multidimensional Scalling on proximity from Random Forest trained only on most important features.

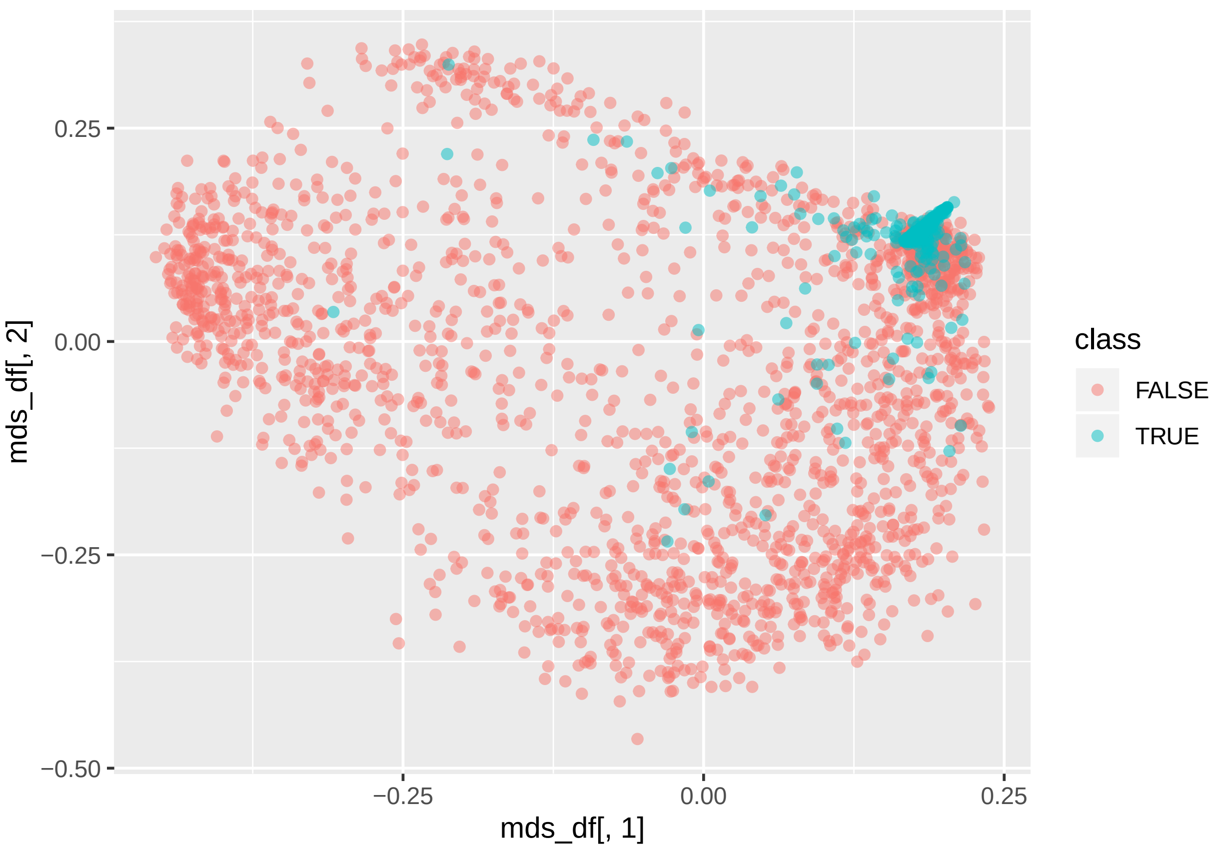
```r
important_cols <- c("PageValues", "ProductRelated_Duration", "ExitRates", "ProductRelated",
                    "Month", "BounceRates", "Administrative_Duration", "TrafficType",
                    "Region", "Administrative")

set.seed(42)
rf_imp <- randomForest(x = sample_df[,important_cols], y = factor(sample_df$Revenue), ntree = 100, prox
prox_imp <- rf_imp$proximity

set.seed(42)
mds_imp <- cmdscale(1/(prox_imp +1))

mds_df <- data.frame(mds_imp)
mds_df$class <- sample_df$Revenue

mds_df %>%
  ggplot(mapping = aes(x = mds_df[,1], y = mds_df[,2], color = class)) +
  geom_point(alpha=.5)
```



The space is filpped horizontally, but overal structure remains similar. We can observe that in the case of using only most important features, more separeted clusters are visible.

Umap

For some experimentation, we can try using UMAP algorithm on proximity matrix produced by Random Forest. It produces denser clusters, small one at the bottom left consists mostly of schoppers session!

```r
library(umap)
library(reticulate)

dist <- 1/(1+prox)
umap.res <- umap(dist, method = "umap-learn", n_components=2, metric="precomputed", transform_seed=42,
                 n_neighbors=15, min_dist=0.2)
embedding <- umap.res$layout

embedding_df <- data.frame(embedding)
embedding_df$class <- sample_df$Revenue

embedding_df %>%
  ggplot(mapping = aes(x = embedding_df[,1], y = embedding_df[,2], color = class)) +
  geom_point(alpha=.5)
```