Bank Term deposit marketing campaign

Sohit Tandon

5/30/2020

1. An introduction/overview/executive summary section

This report is for the edx Data Science Capstone project: Choose Your Own!. I have explored the **UCI Machine Learning Repository** and choose the Bank Marketing Data Set. This data set is quite interesting and not related to any of the previous sample dataset undertaken in this course. This is a multivariate,clean data set representing real world business problem. The classification goal of this project is to predict if the client will subscribe to a term deposit product.

Data Set Description

The data set is related to direct marketing campaigns of a Portuguese banking institution. The bank would do phone marketing campaigns to get customers to subscribe to their term deposit product. As part of the campaign, bank would contact the same customer multiple times to access if they would subscribe to the product or not.

Description of variables

Input Variables

Variable Name	Variable Type	Description
age	numeric	age of customer
job	categorical	type of job
marital	categorical	marital status
education	categorical	education level of customer
default	categorical	has credit in default?
housing	categorical	has housing loan?
loan	categorical	has personal loan?
contact	categorical	communication type
month	categorical	last contact month
day_of_week	categorical	last contact day of week
duration	numeric	last contact duration in sec
campaign	numeric	number of contacts
pdays	numeric	number of days since last contact
previous	numeric	number of contacts for previous campaign
poutcome	categorical	outcome of previous campaign
emp.var.rate	numeric	employment variation rate - quarterly
cons.price.idx	numeric	consumer price index - monthly
cons.conf.idx	numeric	consumer confidence index - monthly
euribor3m	numeric	euribor 3 month rate - daily
nr.employed	numeric	number of employees -quarterly

Output Variables

Variable Name	Variable Type	Description
y	binary	client subscribed to term deposit (y/n) ?

We will download the dataset and load it into R. Thereafter we will fit a few machine learning models to predict whether or not the marketing campaign is successful in getting its customers to subscribe to the term deposit product.

```r
################################
# Download the data Set
################################
# Note: this process could take a couple of minutes

if(!require(tidyverse)) install.packages("tidyverse",repos="http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret",repos="http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table",repos="http://cran.us.r-project.org")
if(!require(lubridate)) install.packages("lubridate",repos="http://cran.us.r-project.org")
if(!require(GGally)) install.packages("GGally",repos="http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart",repos="http://cran.us.r-project.org")
if(!require(rattle)) install.packages("rattle",repos="http://cran.us.r-project.org")
if(!require(descr)) install.packages("descr",repos="http://cran.us.r-project.org")

library(tidyverse)
library(caret)
library(data.table)
library(lubridate)
library(GGally)
library(rpart)
library(RColorBrewer)
library(rattle)
library(descr)
############ Bank Marketing Data Set: ############
# Location of Data Set in UCI Machine Learning Repository
# https://archive.ics.uci.edu/ml/datasets/Bank+Marketing
# Dropbox location of data zipped file
#https://www.dropbox.com/s/u54po56i9acmbm2/bank-additional.zip?dl=0

dl <- tempfile()
download.file("http://archive.ics.uci.edu/ml/machine-learning-databases/00222/bank-additional.zip",dl)
bankData <- read.csv(unzip(dl,"bank-additional/bank-additional-full.csv"),
                     stringsAsFactors = FALSE, header = T,sep = ";")
```

2. Methods/analysis section

Let's explore the dataset after downloading it.

```r
#Checking the header of the downloaded dataset
head(bankData)
```

```
##   age       job marital   education default housing loan   contact month
## 1  56 housemaid married    basic.4y      no      no   no telephone   may
## 2  57  services married high.school unknown      no   no telephone   may
## 3  37  services married high.school      no     yes   no telephone   may
## 4  40    admin. married    basic.6y      no      no   no telephone   may
## 5  56  services married high.school      no      no  yes telephone   may
## 6  45  services married    basic.9y unknown      no   no telephone   may
##   day_of_week duration campaign pdays previous    poutcome emp.var.rate
## 1         mon      261        1   999        0 nonexistent          1.1
## 2         mon      149        1   999        0 nonexistent          1.1
## 3         mon      226        1   999        0 nonexistent          1.1
## 4         mon      151        1   999        0 nonexistent          1.1
## 5         mon      307        1   999        0 nonexistent          1.1
## 6         mon      198        1   999        0 nonexistent          1.1
##   cons.price.idx cons.conf.idx euribor3m nr.employed  y
## 1         93.994         -36.4     4.857        5191 no
## 2         93.994         -36.4     4.857        5191 no
## 3         93.994         -36.4     4.857        5191 no
## 4         93.994         -36.4     4.857        5191 no
## 5         93.994         -36.4     4.857        5191 no
## 6         93.994         -36.4     4.857        5191 no
```

```r
# Glimpse of the dataset
glimpse(bankData)
```

```
## Observations: 41,188
## Variables: 21
## $ age            <int> 56, 57, 37, 40, 56, 45, 59, 41, 24, 25, 41, 25,...
## $ job            <chr> "housemaid", "services", "services", "admin.", ...
## $ marital        <chr> "married", "married", "married", "married", "ma...
## $ education      <chr> "basic.4y", "high.school", "high.school", "basi...
## $ default        <chr> "no", "unknown", "no", "no", "no", "unknown", "...
## $ housing        <chr> "no", "no", "yes", "no", "no", "no", "no", "no"...
## $ loan           <chr> "no", "no", "no", "no", "yes", "no", "no", "no"...
## $ contact        <chr> "telephone", "telephone", "telephone", "telepho...
## $ month          <chr> "may", "may", "may", "may", "may", "may", "may"...
## $ day_of_week    <chr> "mon", "mon", "mon", "mon", "mon", "mon", "mon"...
## $ duration       <int> 261, 149, 226, 151, 307, 198, 139, 217, 380, 50...
## $ campaign       <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ pdays          <int> 999, 999, 999, 999, 999, 999, 999, 999, 999, 99...
## $ previous       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ poutcome       <chr> "nonexistent", "nonexistent", "nonexistent", "n...
## $ emp.var.rate   <dbl> 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1....
## $ cons.price.idx <dbl> 93.994, 93.994, 93.994, 93.994, 93.994, 93.994,...
## $ cons.conf.idx  <dbl> -36.4, -36.4, -36.4, -36.4, -36.4, -36.4, -36.4...
## $ euribor3m      <dbl> 4.857, 4.857, 4.857, 4.857, 4.857, 4.857, 4.857...
## $ nr.employed    <dbl> 5191, 5191, 5191, 5191, 5191, 5191, 5191, 5191,...
## $ y              <chr> "no", "no", "no", "no", "no", "no", "no", "no",...
```

We observe that there are 41,188 records overall with 21 variables. Now we will check for any non existent data

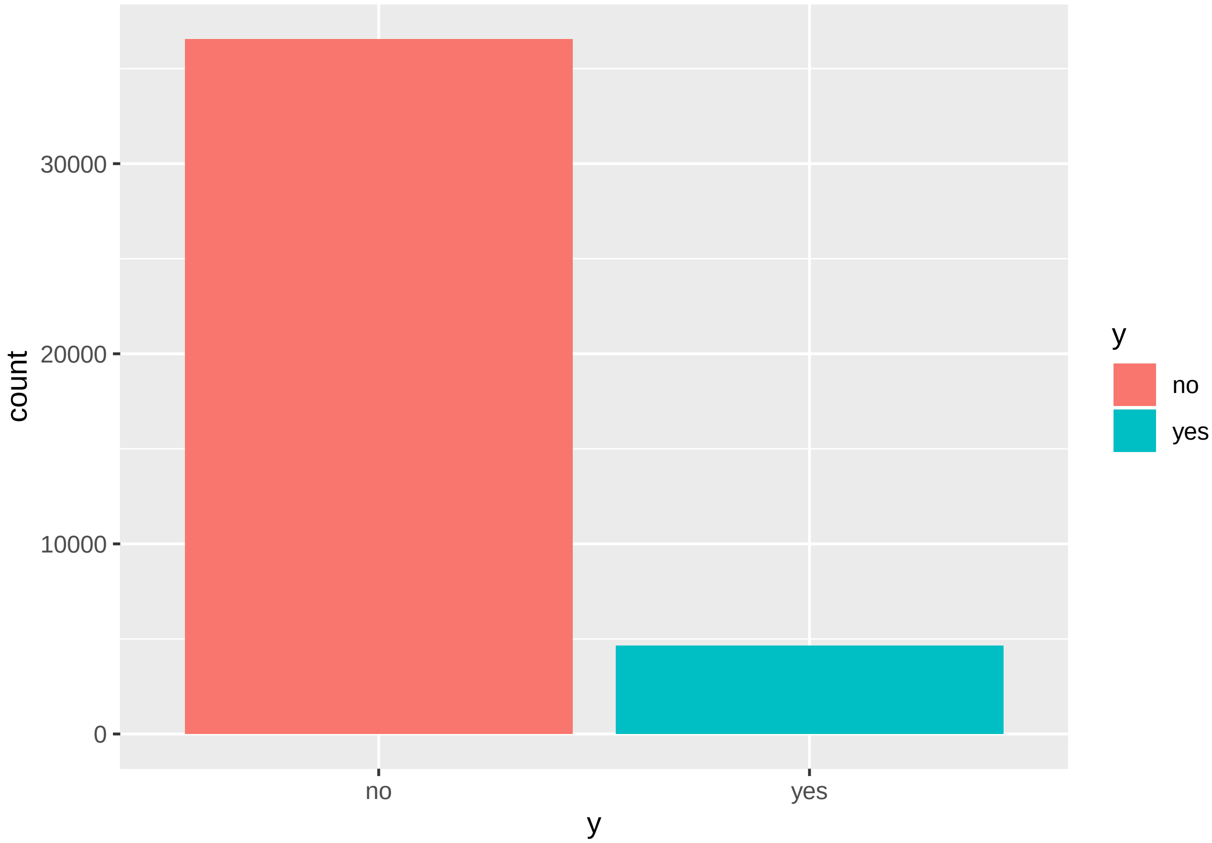
```r
## Checking for NAs
sapply(bankData, {function(x) any(is.na(x))}) %>% knitr::kable()
```

	x
age	FALSE
job	FALSE
marital	FALSE
education	FALSE
default	FALSE
housing	FALSE
loan	FALSE
contact	FALSE
month	FALSE
day_of_week	FALSE
duration	FALSE
campaign	FALSE
pdays	FALSE
previous	FALSE
poutcome	FALSE
emp.var.rate	FALSE
cons.price.idx	FALSE
cons.conf.idx	FALSE
euribor3m	FALSE
nr.employed	FALSE
y	FALSE

Data Exploration

There are no NAs in the dataset. Now we will do initial data exploration on the variables.

```
#Distribution of Output variable

ggplot(bankData)+geom_bar(aes(y,fill=y))
```



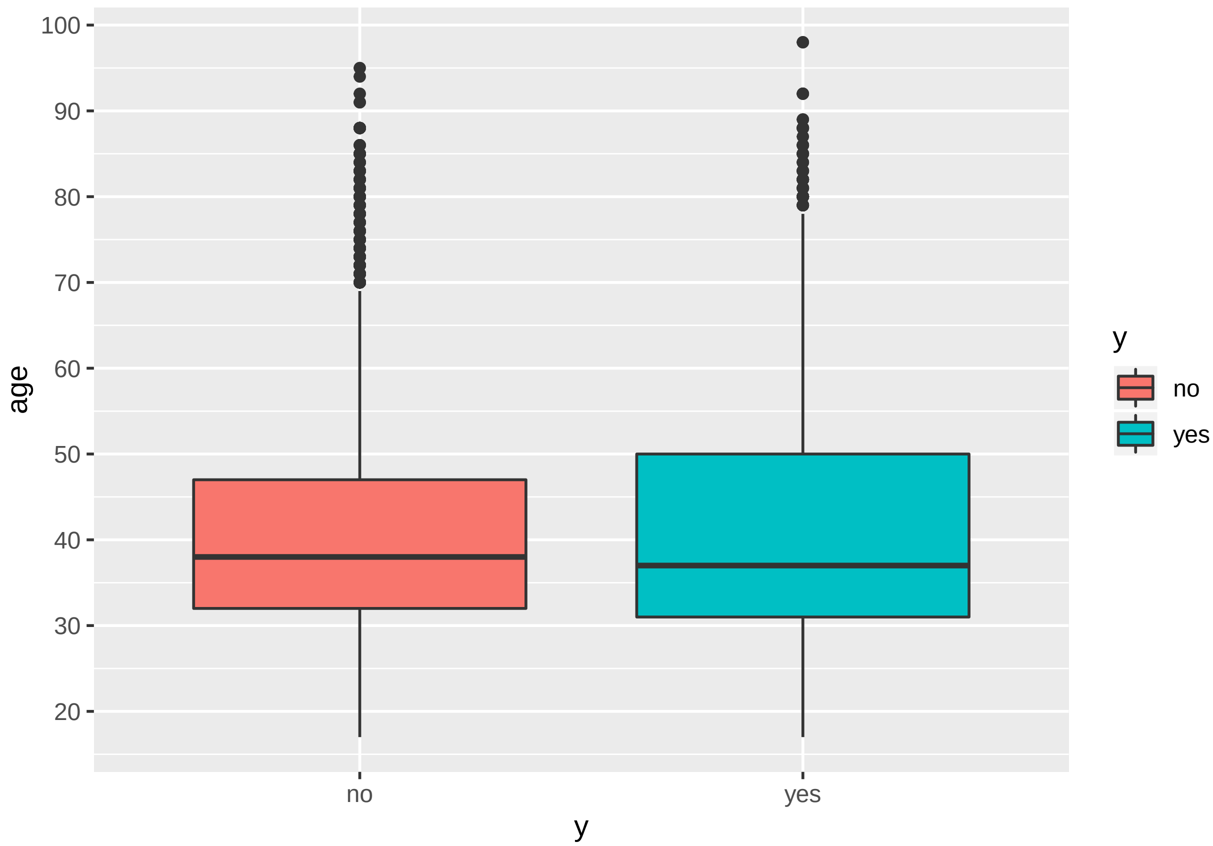

```
prop.table(table(bankData$y))
```

```
##
##        no       yes
## 0.8873458 0.1126542
```

We observe that the success rate of marketing campaign is about 11.3 % (yes). Rest of the times i.e 88.7 % the campaign is unsuccessful .

Let's visualize the **age** column distribution

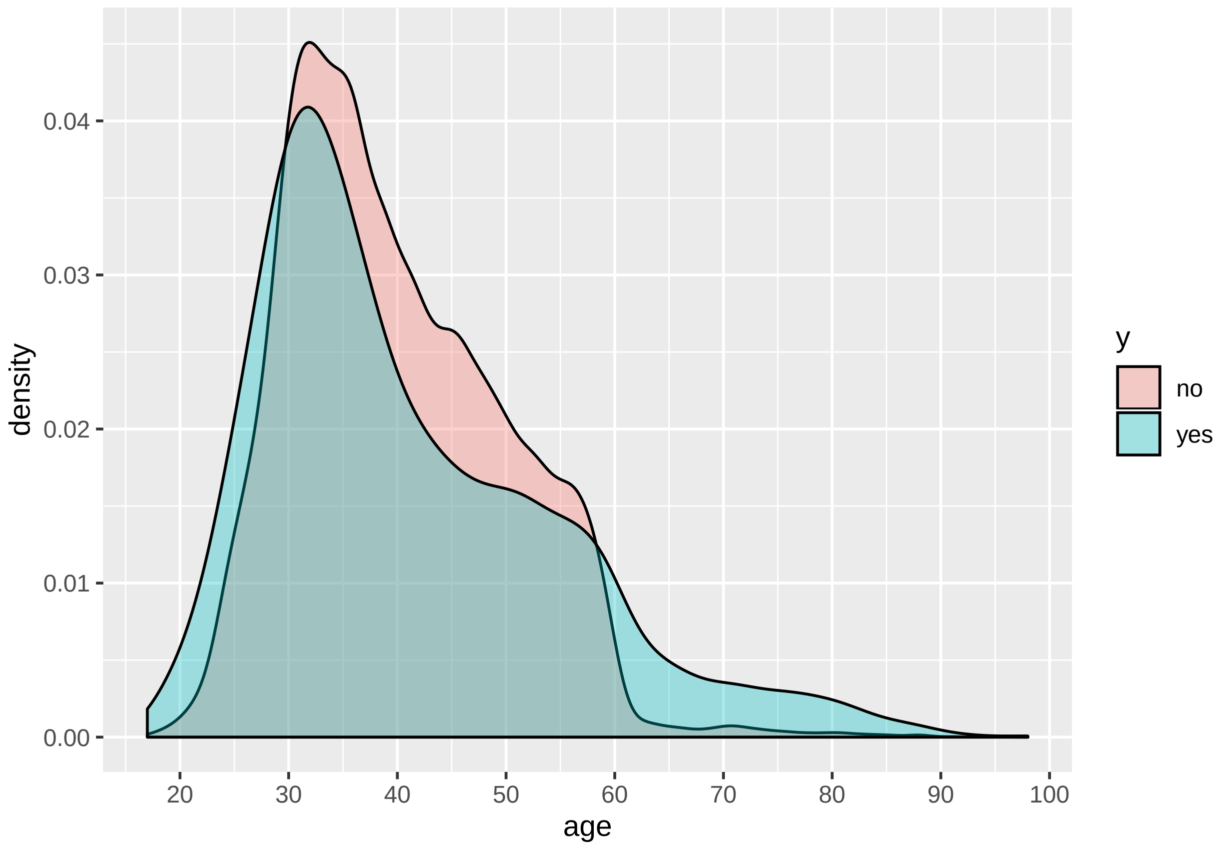
```
# Distribution of age for term deposit subscription visualization

ggplot(bankData) +
  geom_boxplot(aes(y, age,fill=y))+
  scale_y_continuous(breaks = c(0,10,20,30,40,50,60,70,80,90,100))
```



From the above plot , the median age of customers who subscribed as well as those who did not subscribe is around 38 to 40. Also both the *yes* and *no* overlap a lot.

Let's create a density plot to understand the overlap of *yes* and *no* in context of **age** variable .

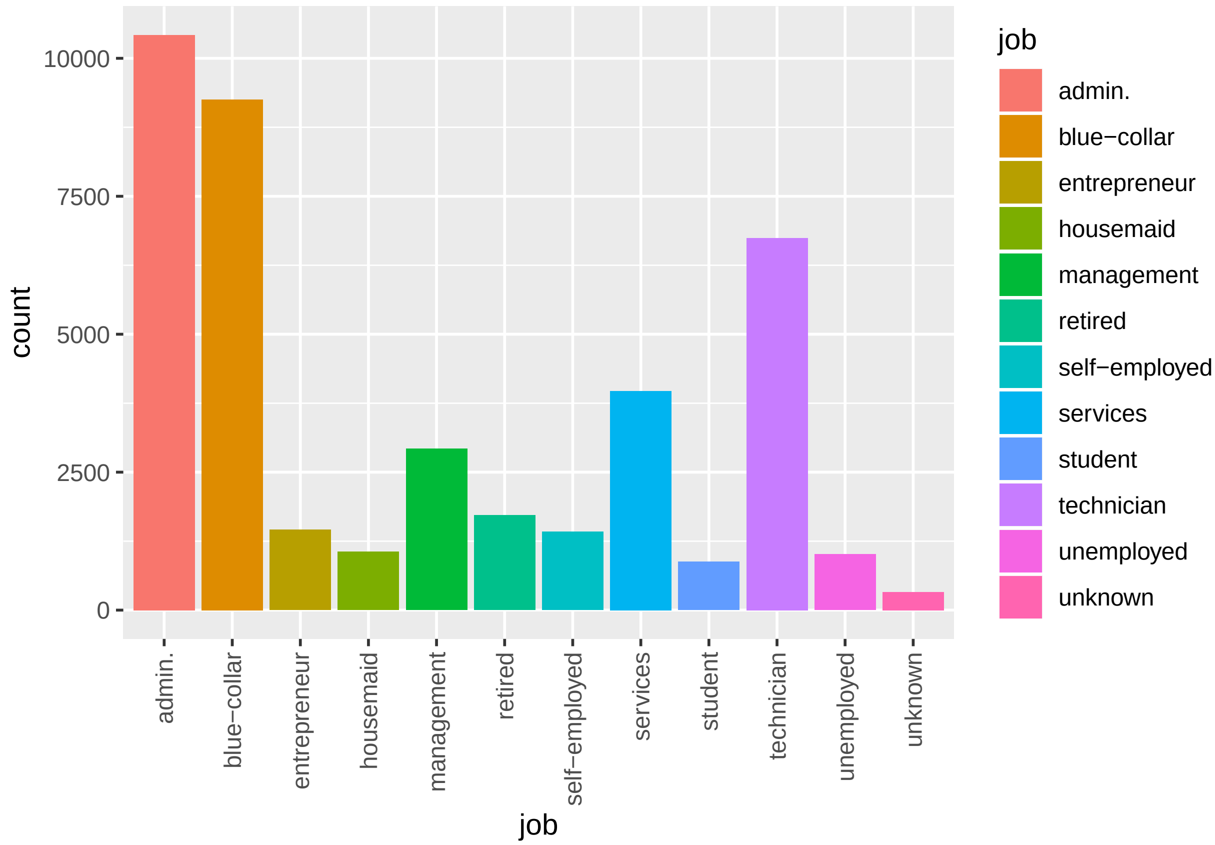
```
# Density plot of age variable for term deposit subscription

ggplot(bankData)+geom_density(aes(age,fill=y),alpha=1/3)+
  scale_x_continuous(breaks = c(0,10,20,30,40,50,60,70,80,90,100))
```



Both *yes* and *no* overlap a lot for the age variable, this means this is not a good indicator for which customer will subscribe and which customer will not.

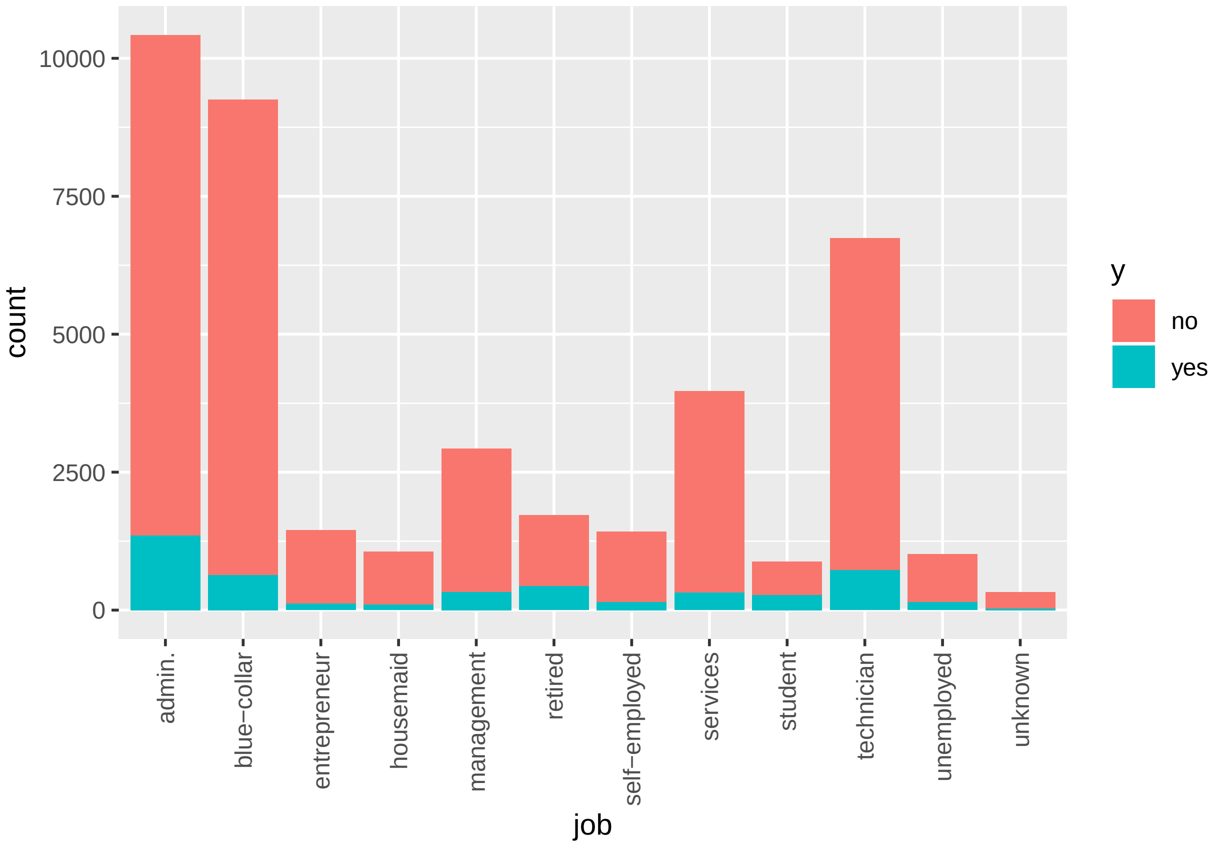
Now we will visualize the distribution of **job** variable in the dataset.

```
#Distiribution of job variable visualization

ggplot(bankData)+geom_bar(aes(job,fill=job)) +
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5))
```



Now let's visualize how **job** variable relates to term deposit subscriptions

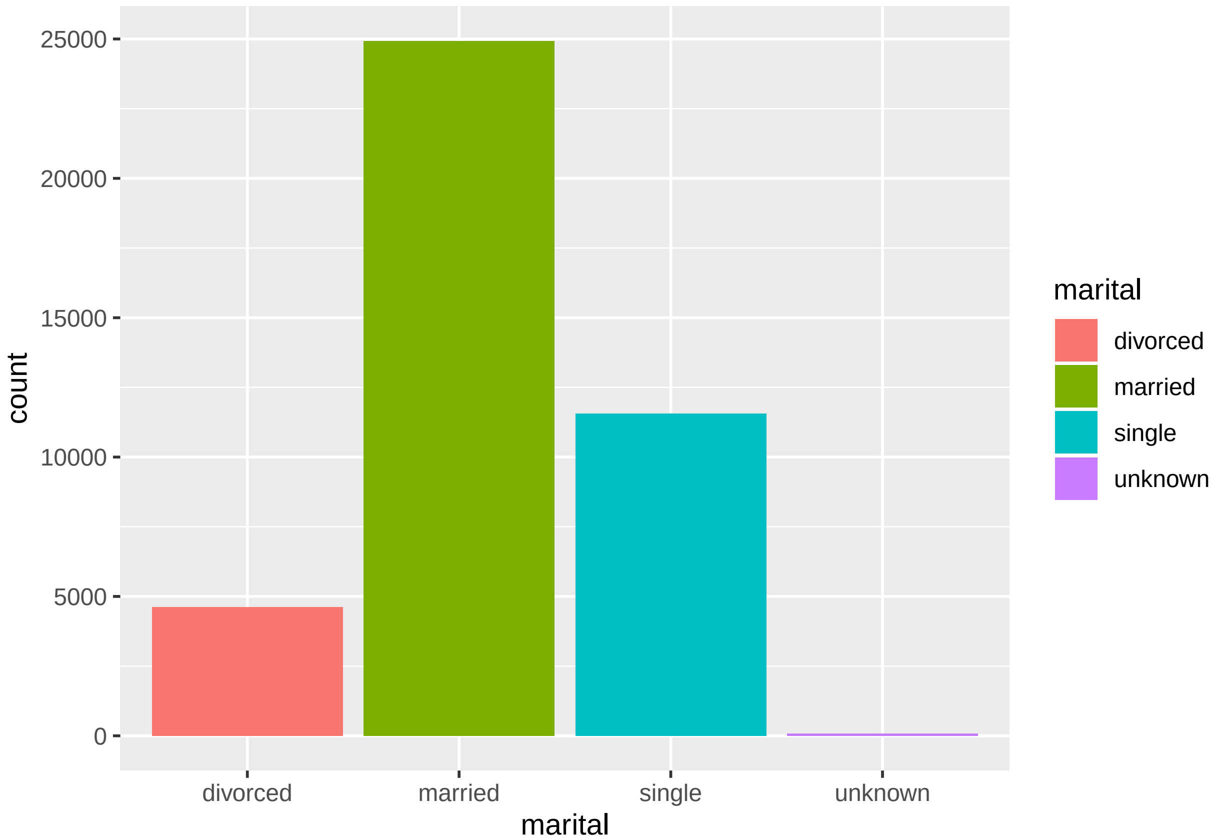
```
#Distiribution of job variable for term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(job,fill=y))+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5))
```



From the above plot it is evident that customers with following job categories are the top 3 to subscribe for the product
1. admin
2. technician
3. blue-collar

Now we will visualize the distribution of **marital** variable in the dataset.

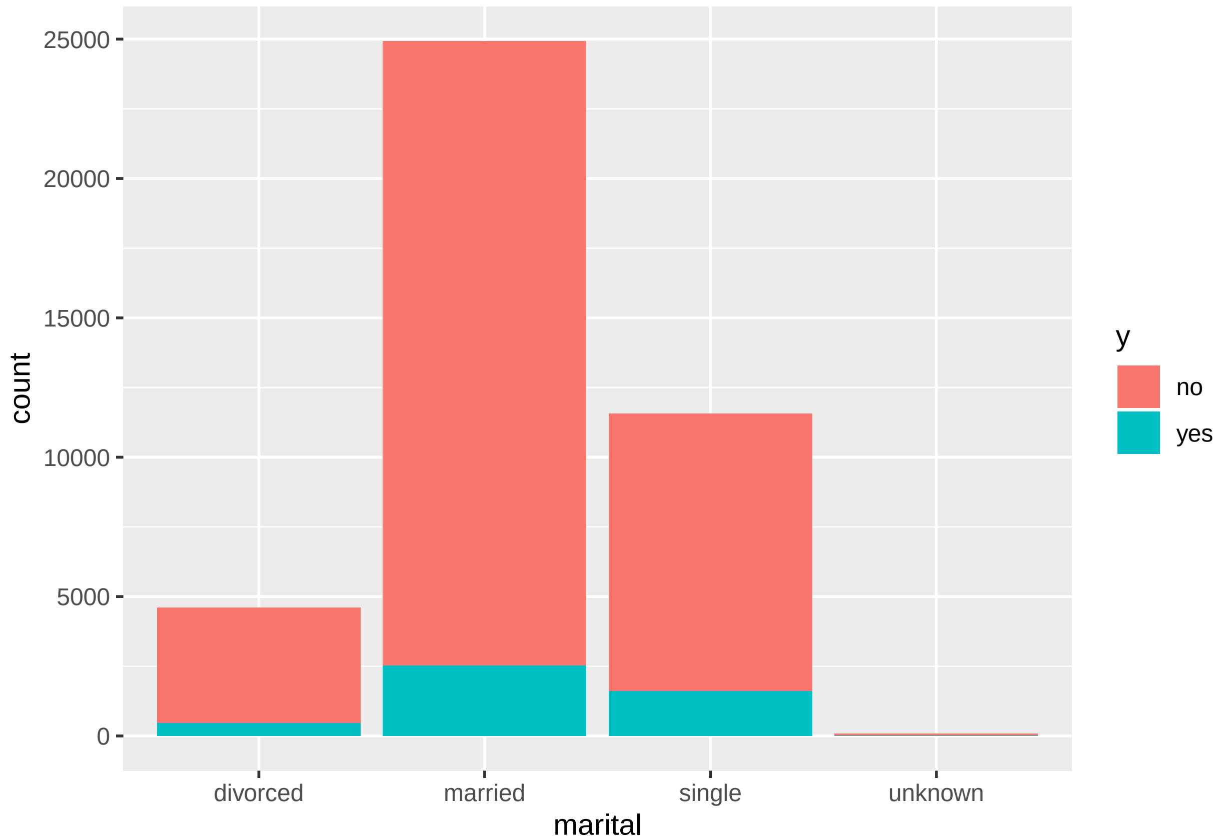
```
#Distiribution of marital visualization

ggplot(bankData)+geom_bar(aes(marital,fill=marital))
```



We can see that most of the bank's customers are married.

Now let's visualize how **marital** variable relates to term deposit subscriptions

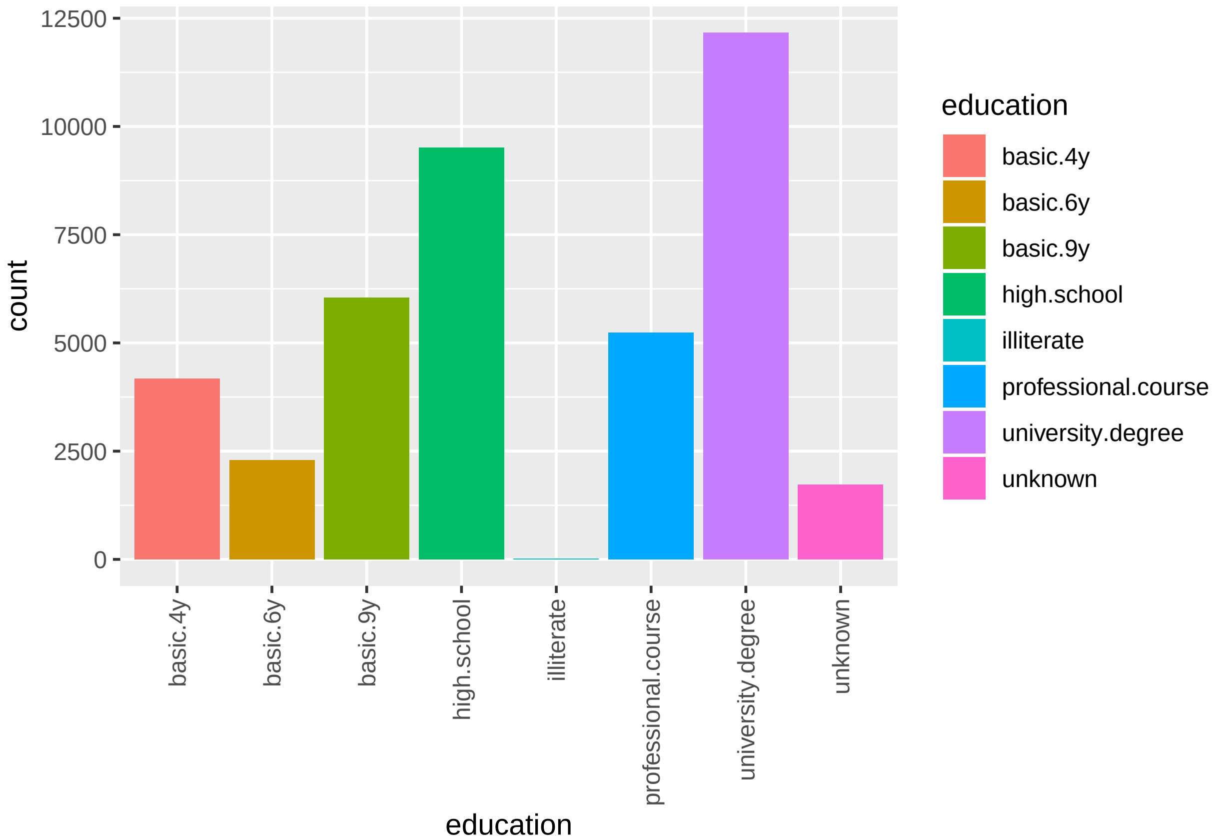
```
#Distiribution of marital variable to term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(marital,fill=y))
```



Most subscribers are married followed by single. But that is also the distribution of data for this variable.

Now we will visualize the distribution of **education** variable in the dataset.

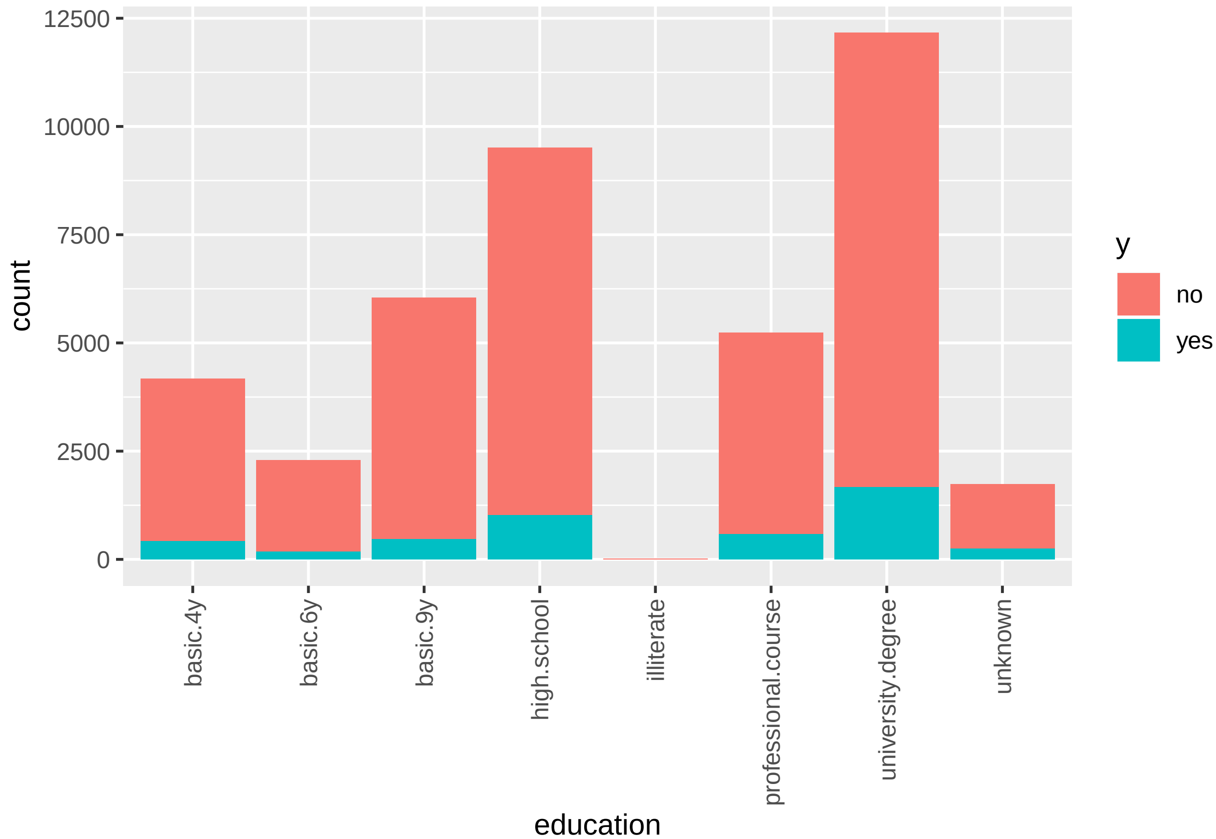
```
#Distiribution of education category visualization

ggplot(bankData)+geom_bar(aes(education,fill=education))+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5))
```



Majority of the customers are university grads followed by high school

Now let's visualize how **education** variable relates to term deposit subscriptions.

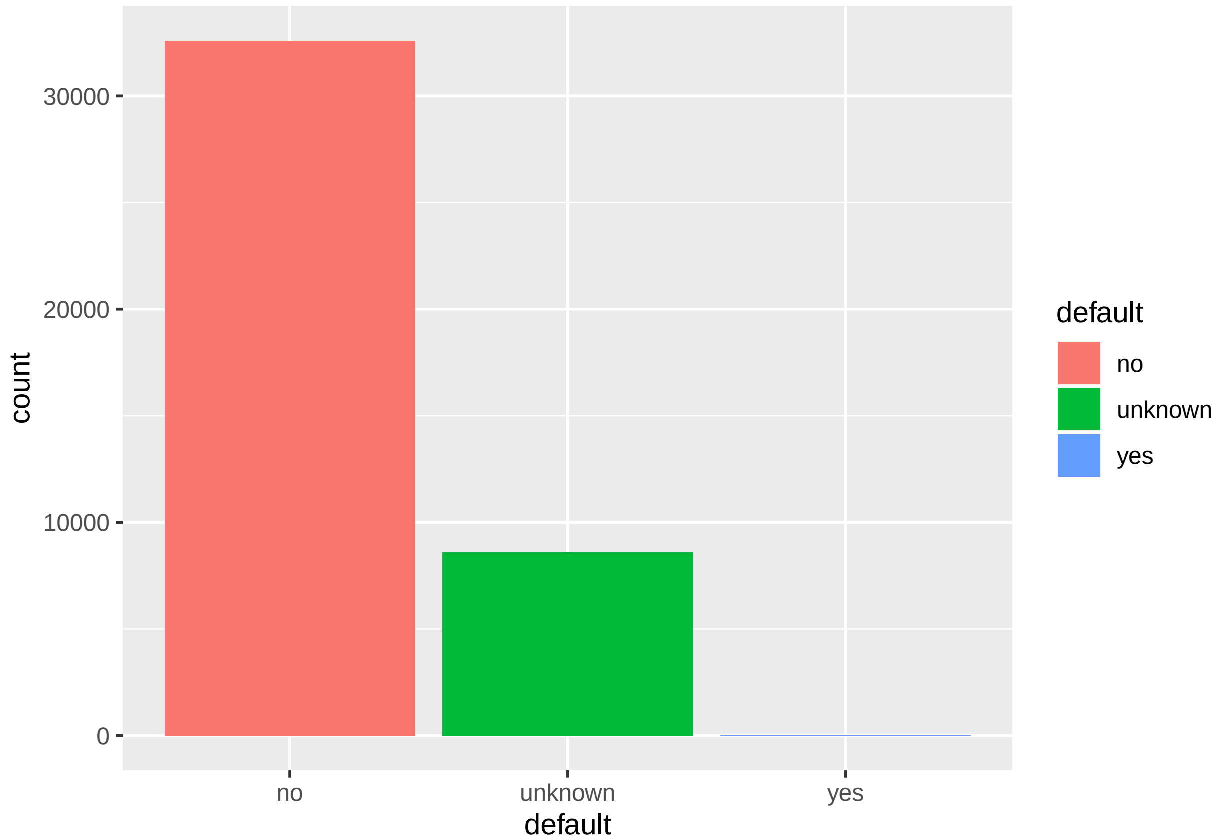
```
#Distiribution of education variable to term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(education,fill=y))+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5))
```



Most of the subscribers are having university degree followed by high school and then professional courses.

Now we will visualize the distribution of **default** variable in the dataset.

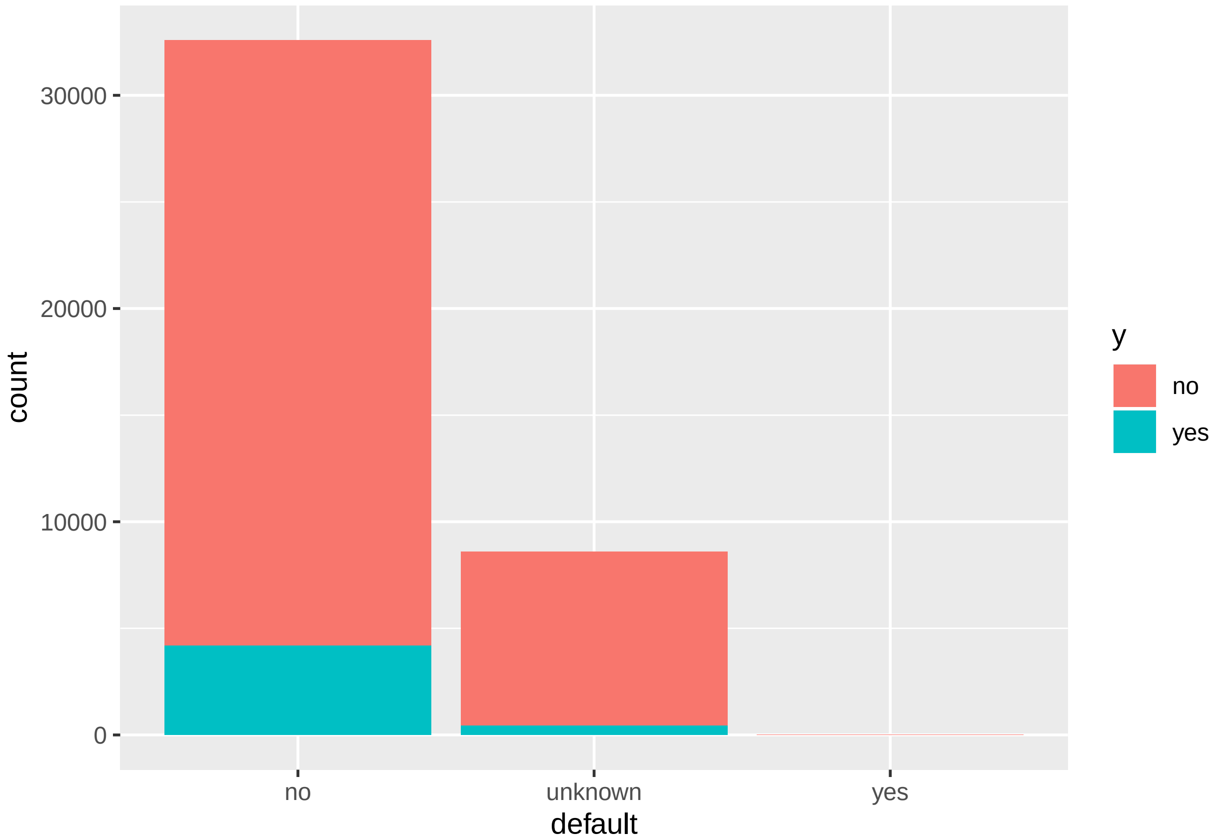
```
#Distiribution of credit default visualization

ggplot(bankData)+geom_bar(aes(default,fill=default))
```



It is evident from the chart that most of the customers do not have a credit default with the bank

Now let's visualize how **default** variable relates to term deposit subscriptions.

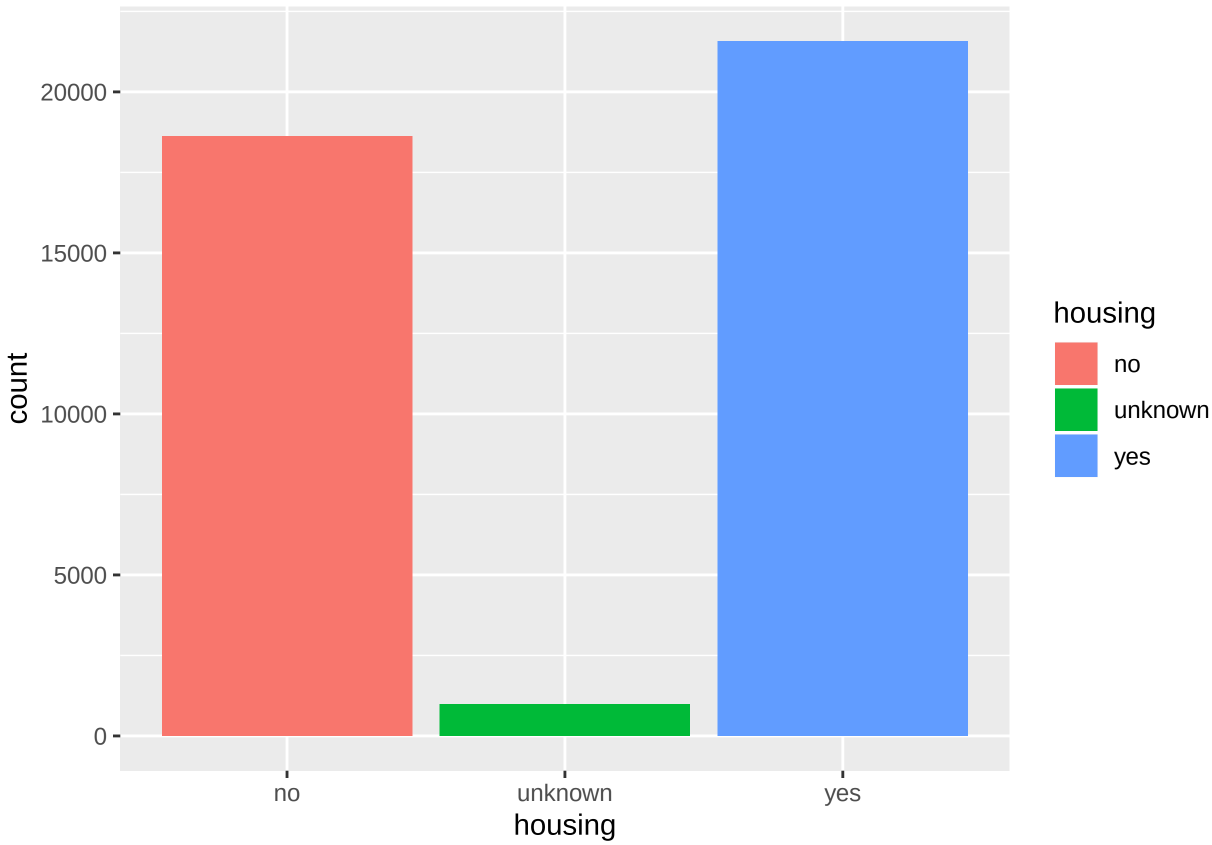
```
#Distiribution of credit default on term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(default,fill=y))
```



From the chart it is evident that most of the subscribers to term deposit are non defaulters with the bank.

Now we will visualize the distribution of **housing** variable in the dataset.

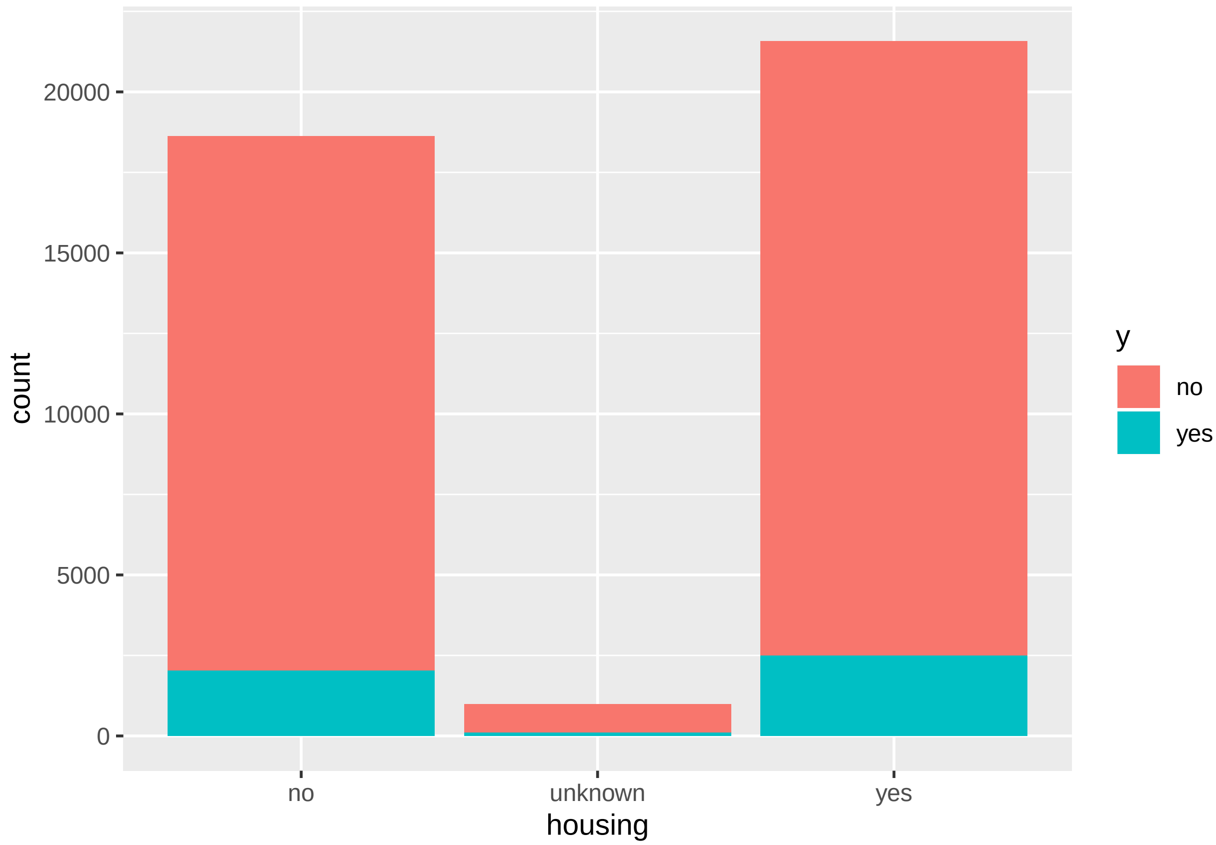
```
#Distiribution of housing variable  visualization

ggplot(bankData)+geom_bar(aes(housing,fill=housing))
```



The distribution suggests that majority of customers do have a home loan from the bank.

Now let's visualize how **housing** variable relates to term deposit subscriptions.

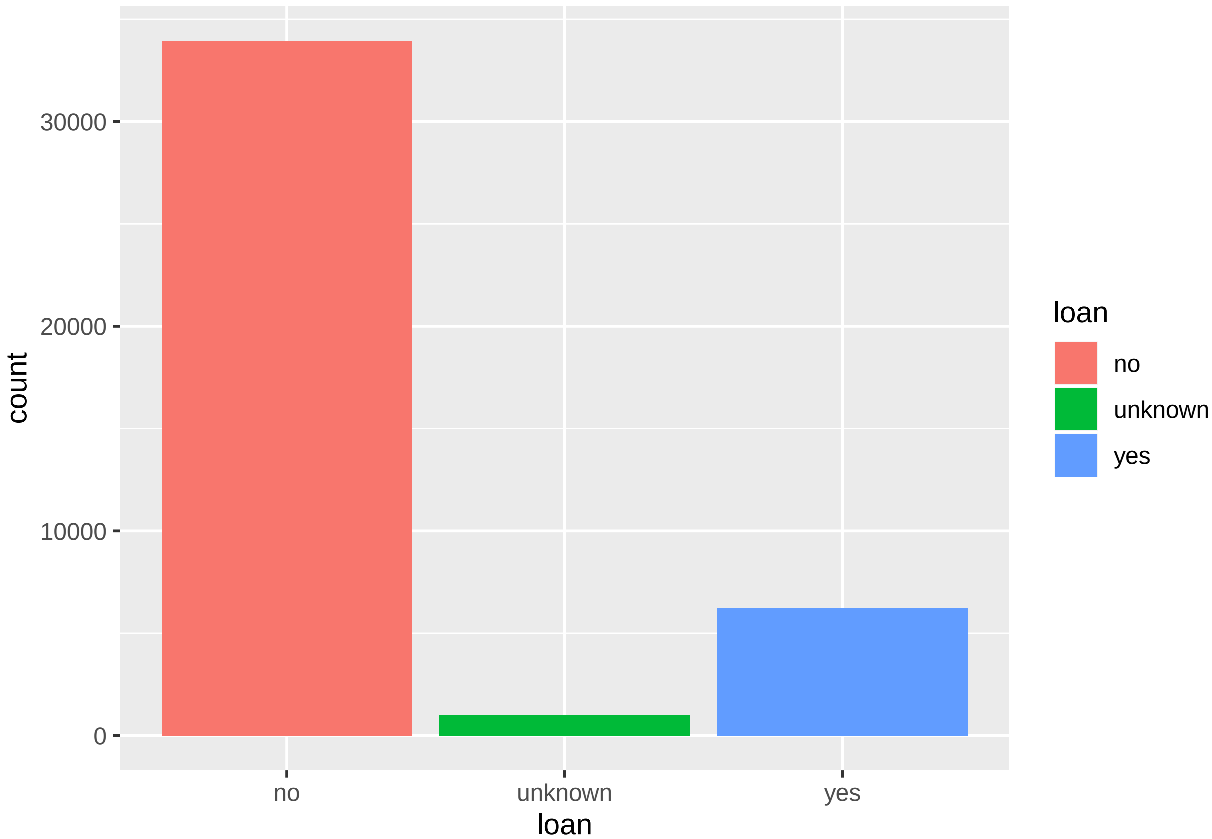
```r
#Distiribution of housing variable to term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(housing,fill=y))
```



The subscription rate of housing loan customers is slightly more than those customers who do not have a housing loan.

Now we will visualize the distribution of **loan** variable in the dataset.

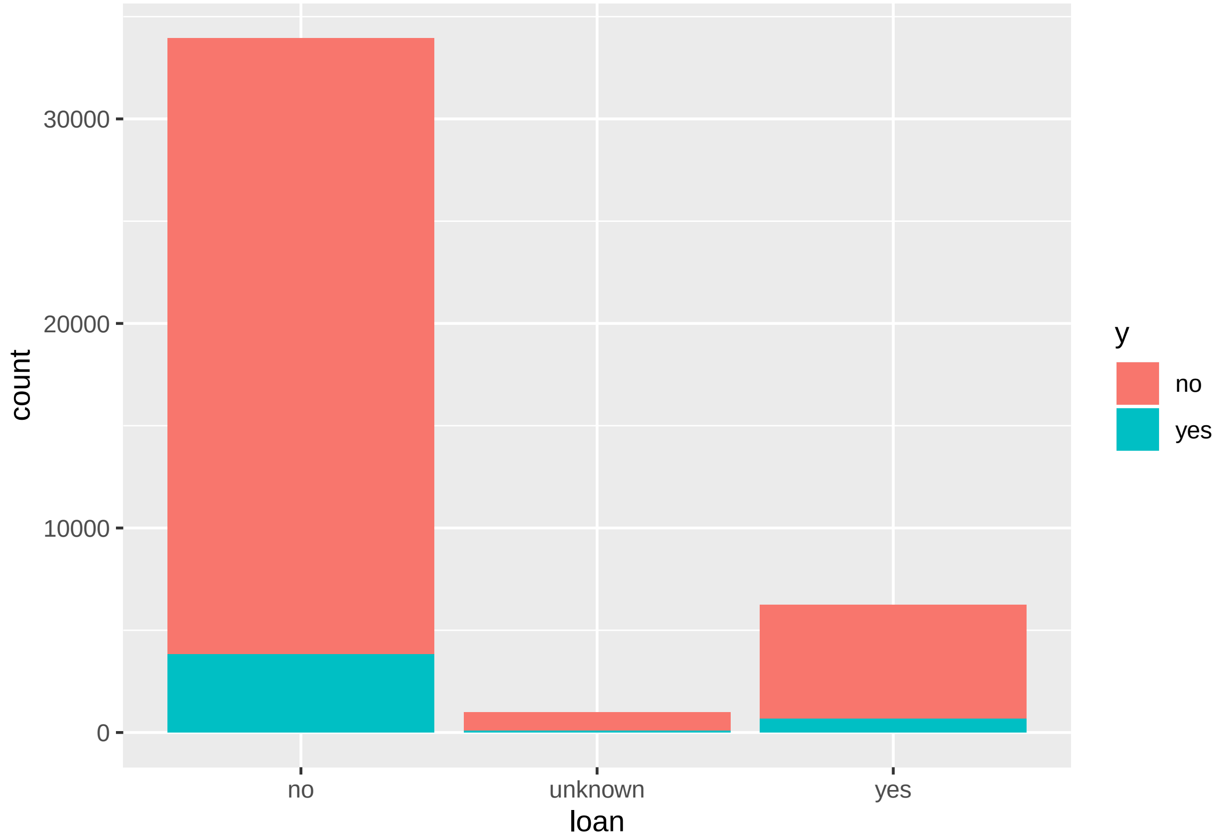
```
#Distiribution of loan category visualization

ggplot(bankData)+geom_bar(aes(loan,fill=loan))
```



The plot suggests that most of the most customers do not have a personal loan from the bank.

Now let's visualize how **loan** variable relates to term deposit subscriptions.

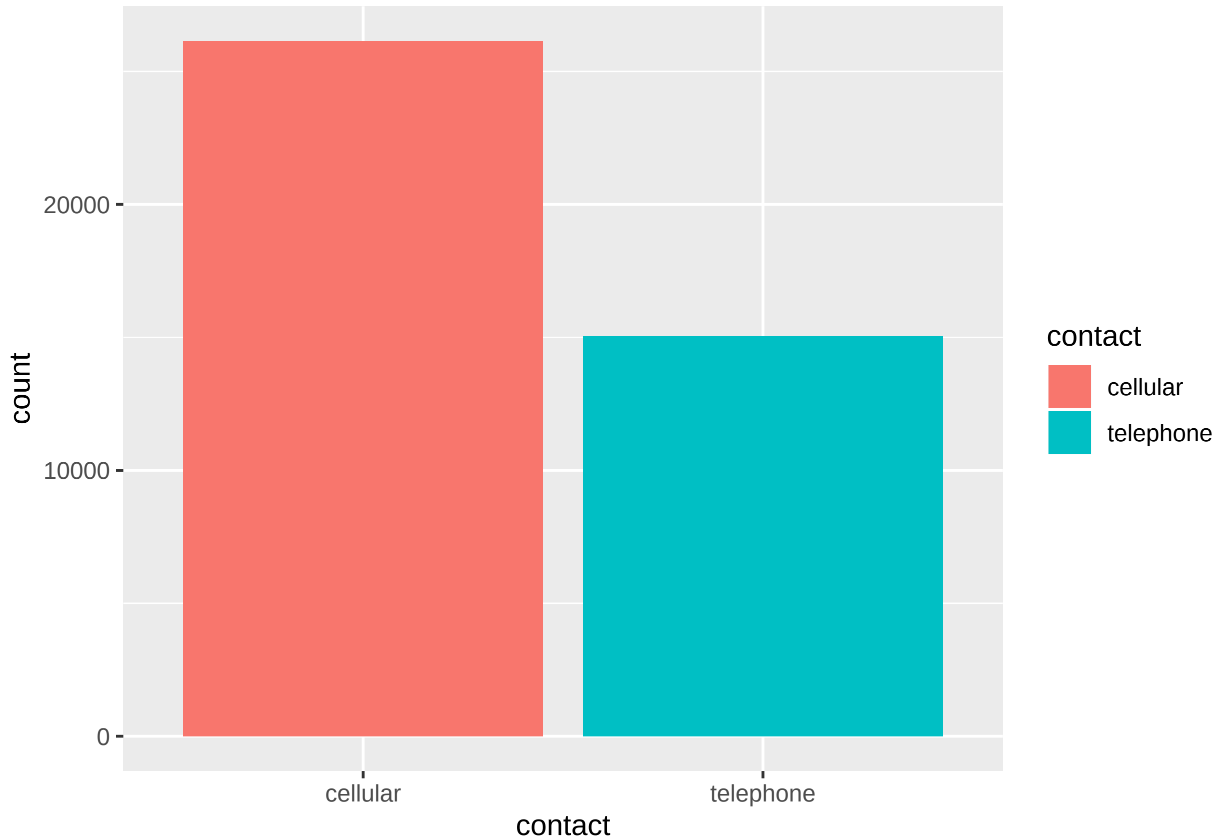
```
#Distiribution of loan variable to  term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(loan,fill=y))
```



The subscription for term deposit is higher amongst customers who do not have a personal loan with the bank.

Now we will visualize the distribution of **contact** variable in the dataset.

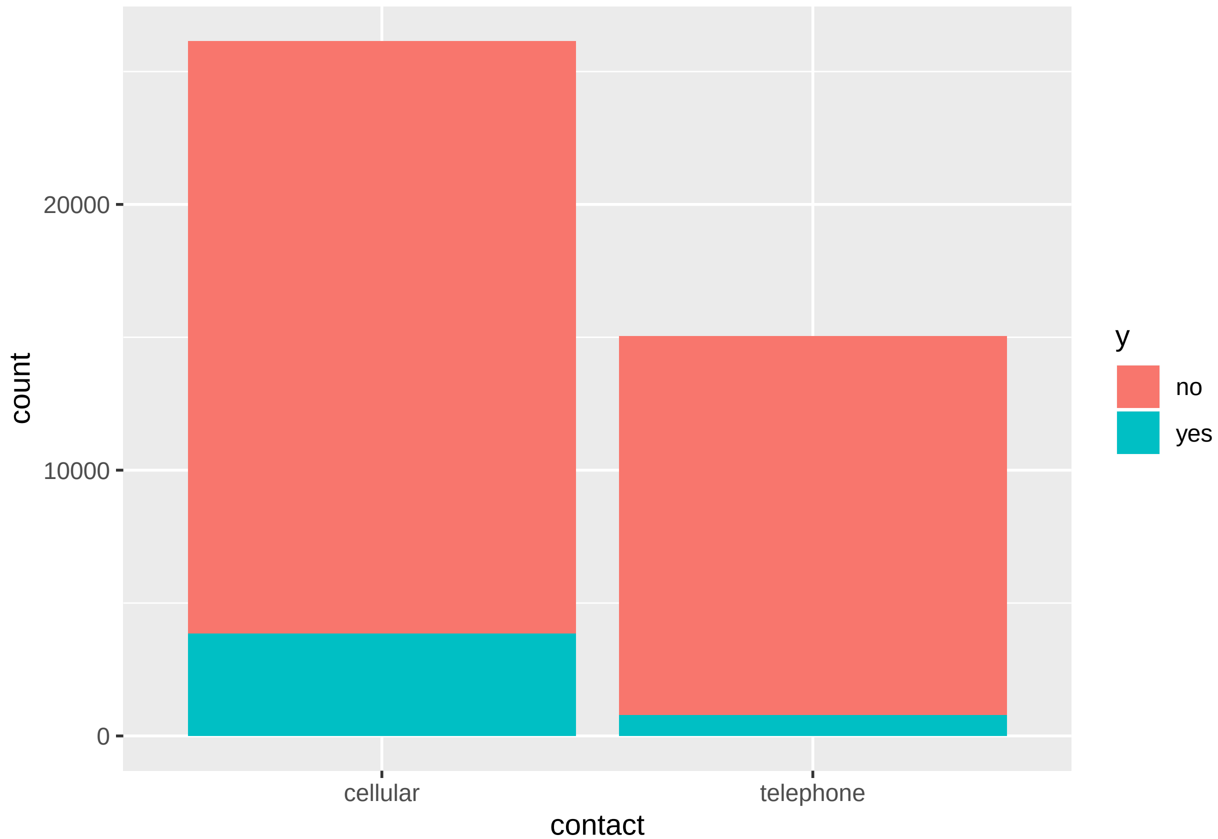
```r
#Distiribution of contact variable visualization

ggplot(bankData)+geom_bar(aes(contact,fill=contact))
```



The chart suggests that bank contacted customers using cellular communication channel more than normal telephone.

Now let's visualize how **contact** variable relates to term deposit subscriptions.

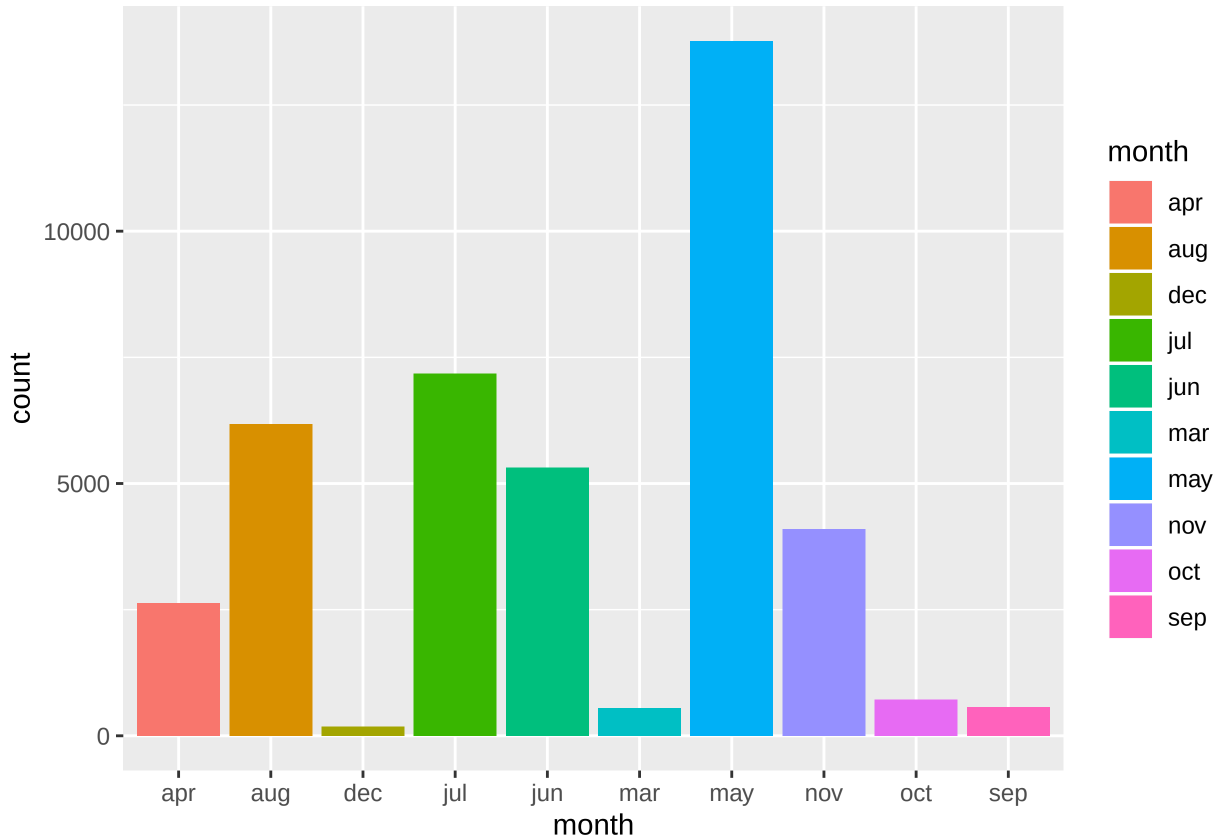
```
#Distiribution of contact variable to term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(contact,fill=y))
```



The chart suggests that subscription rate of customers contacted via cellular communication channel is quite high as compared to those contacted by telephone.

Now we will visualize the distribution of **month** variable in the dataset.

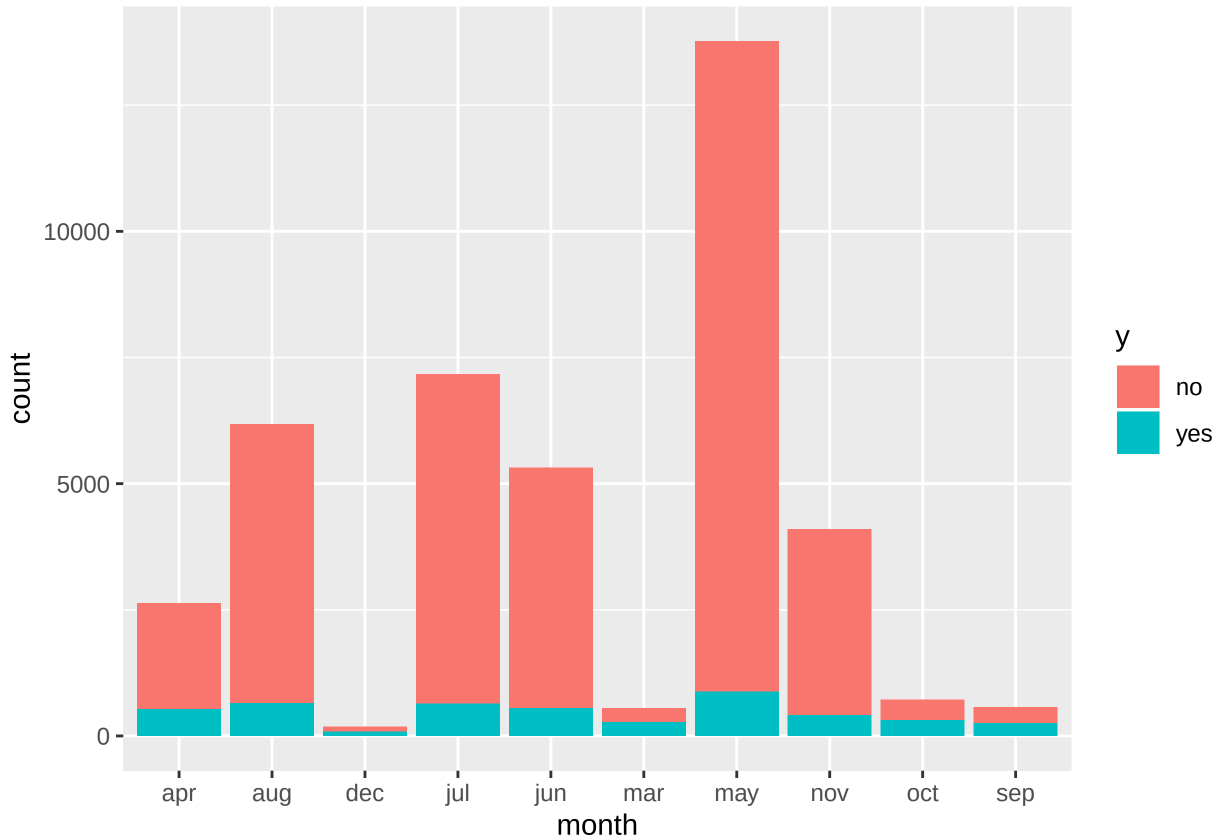
```
#Distiribution of month variable visualization

ggplot(bankData)+geom_bar(aes(month,fill=month))
```



From the chart it is evident the number of customers contacted in may was highest

Now let's visualize how last contact **month** variable relates to term deposit subscriptions.

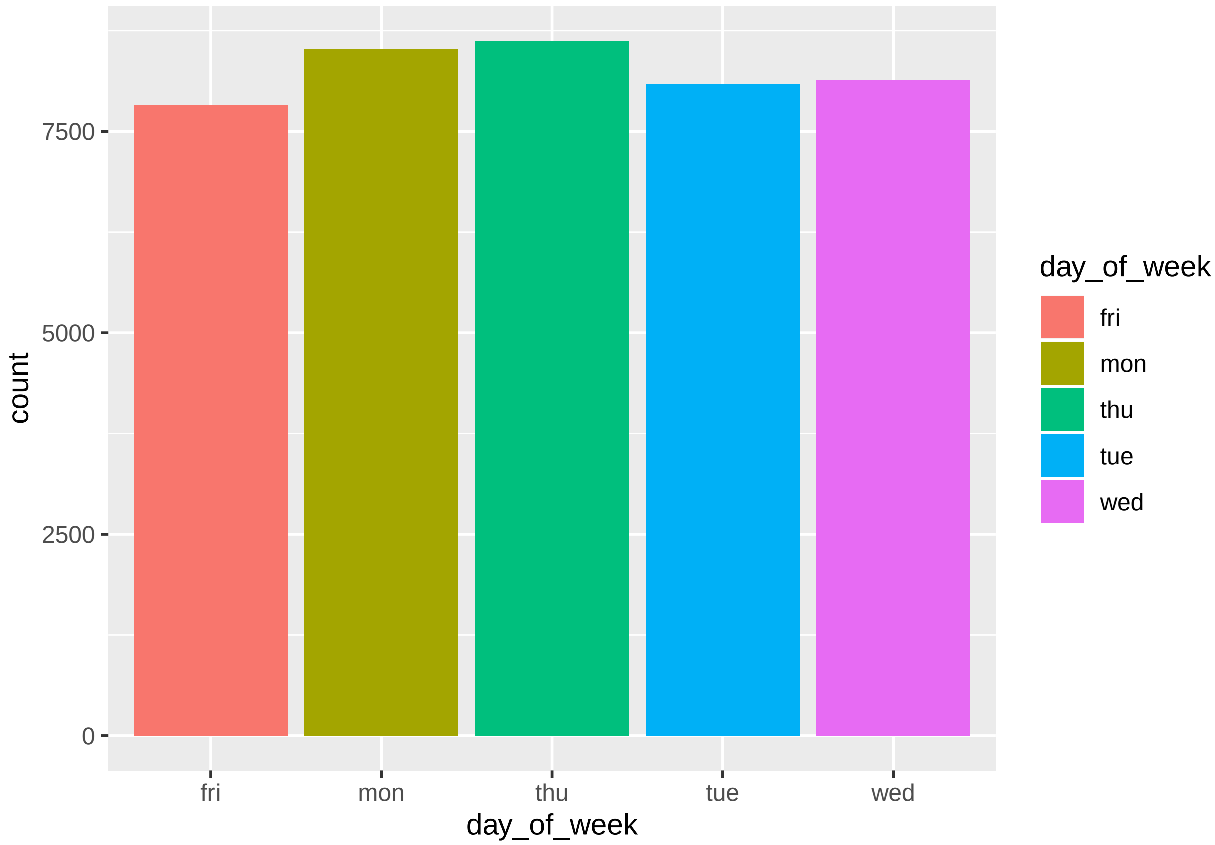
```
#Distiribution of  last contct month variable to term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(month,fill=y))
```



It is evident from the chart that the subscription rate for the term deposit is highest for customers which were last contacted in may , followed by July and Aug.

Now we will visualize the distribution of **day_of_week** variable in the dataset.

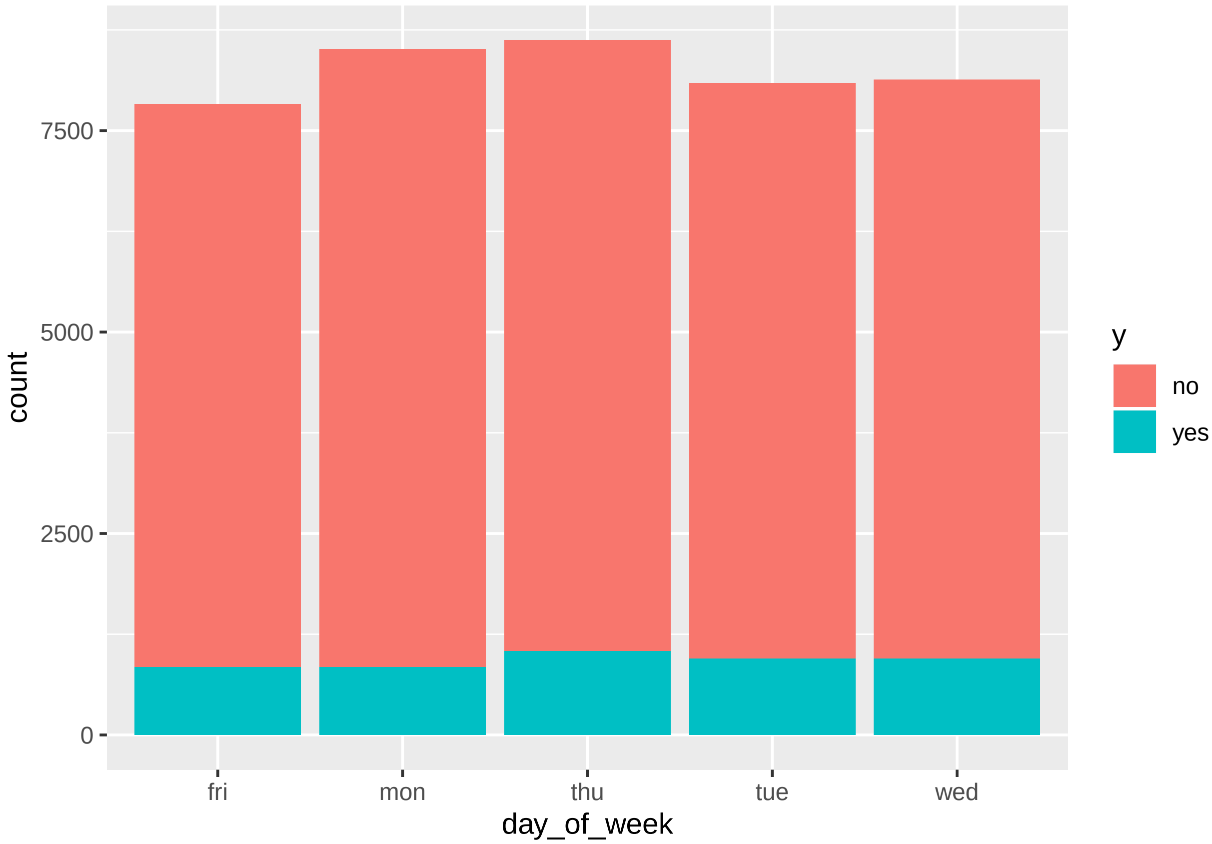
```
#Distiribution of day_of_week variable visualization

ggplot(bankData)+geom_bar(aes(day_of_week,fill=day_of_week))
```



The chart depicts that most customers were last contacted on Thursday, followed by Monday.

Now let's visualize how last contact **day__of__week** variable relates to term deposit subscriptions.

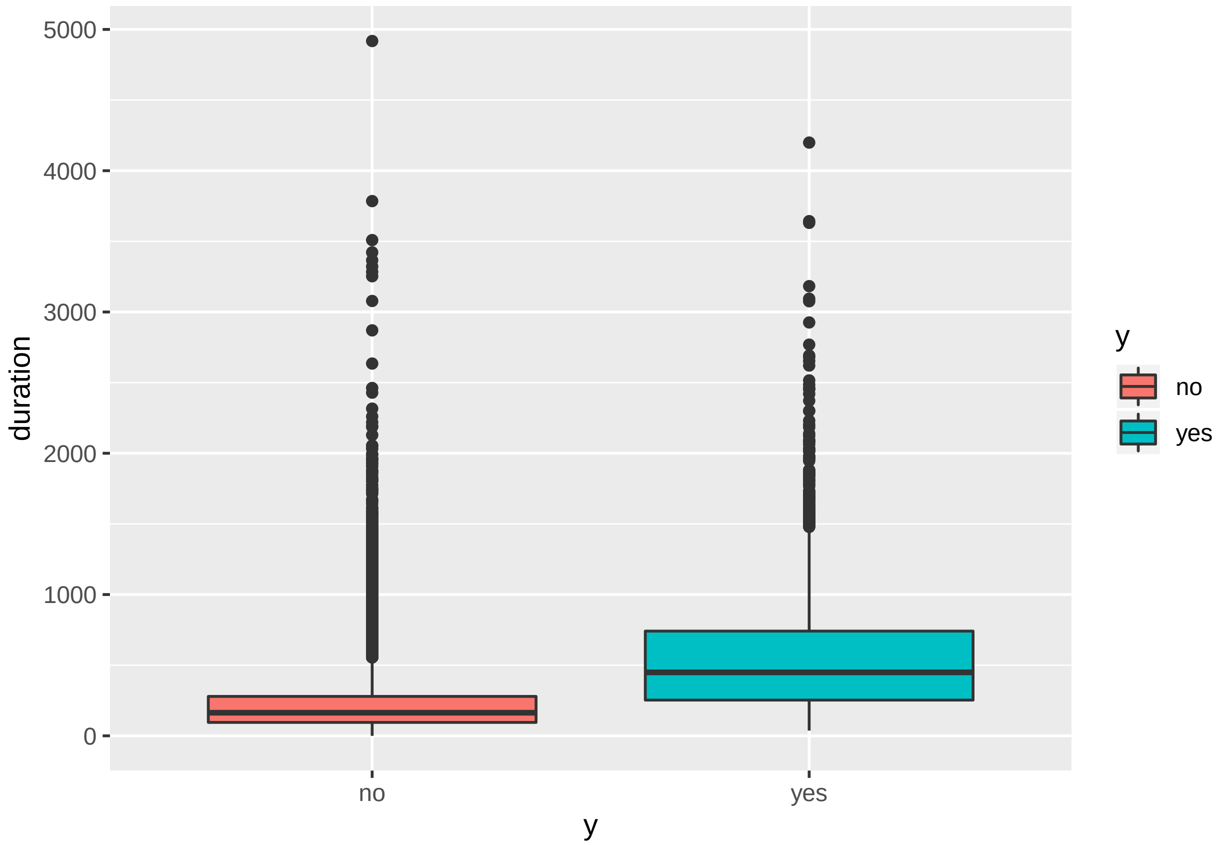
```
#Distiribution of last contact day_of_week variable to term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(day_of_week,fill=y))
```



The plot depicts that subscription rate for customers who were last contacted on Thursday is slightly higher than other days.

Now we will visualize the distribution of **duration** variable in the dataset.

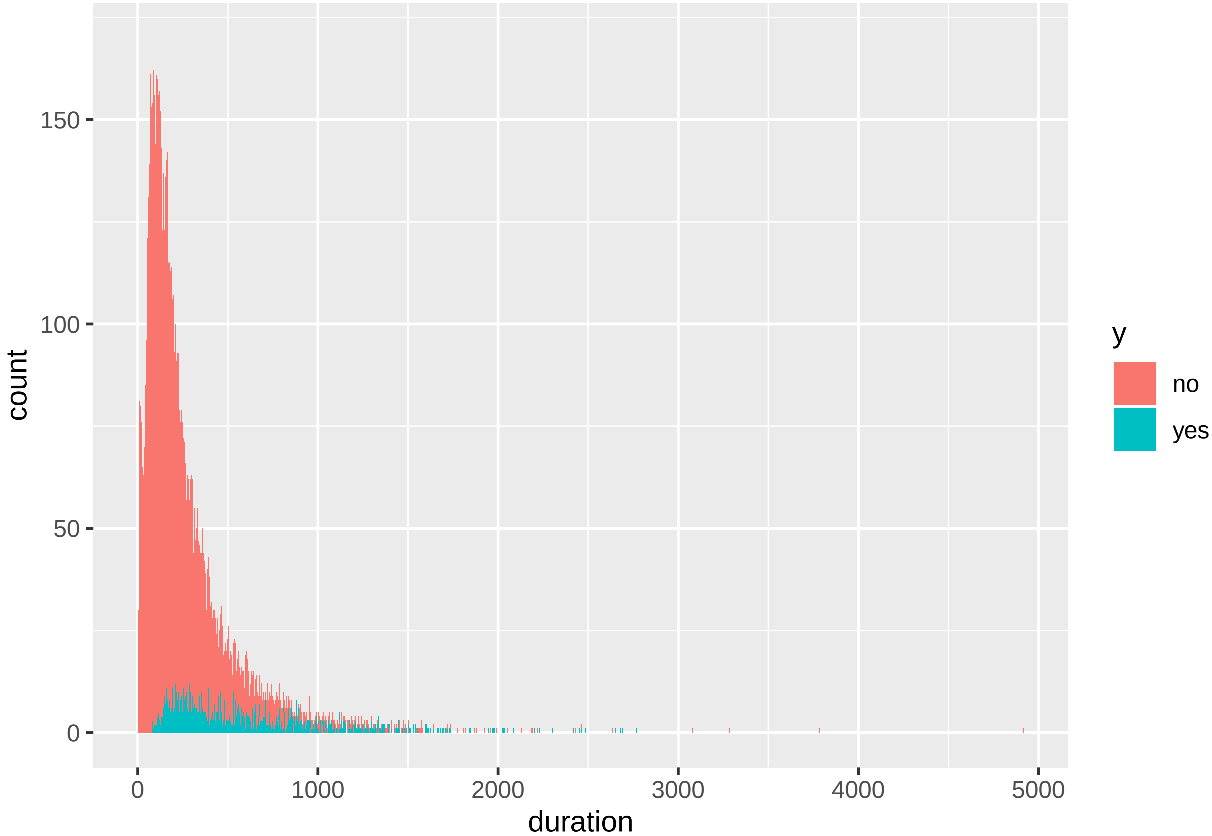
```
#Distiribution of duration variable visualization

ggplot(bankData) +
  geom_boxplot(aes(y, duration,fill=y))
```



The plot reveals that last contact duration with the customer can have impact on the customer subscribing the term deposit.

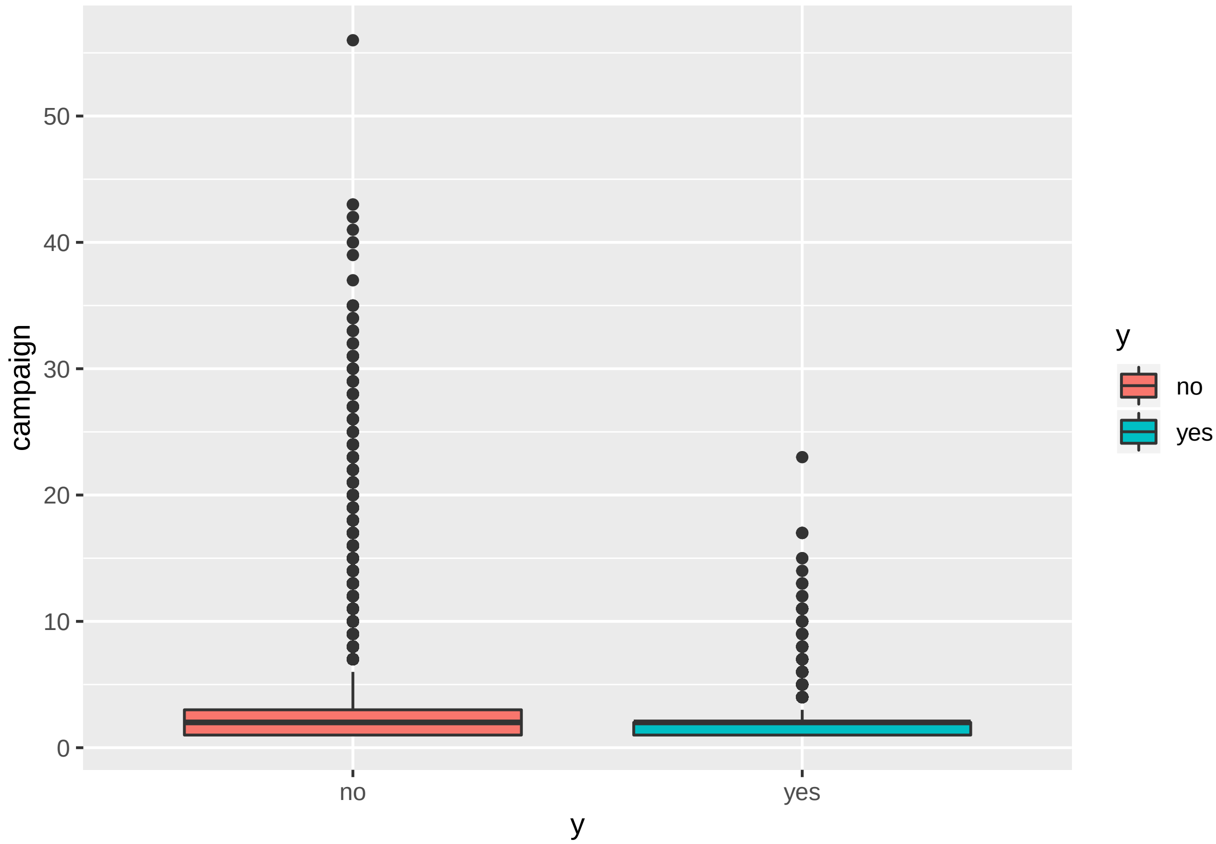
Now let's visualize how last contact **duration** variable relates to term deposit subscriptions.

```
# Distribution of contact duration to term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(duration,fill=y))
```



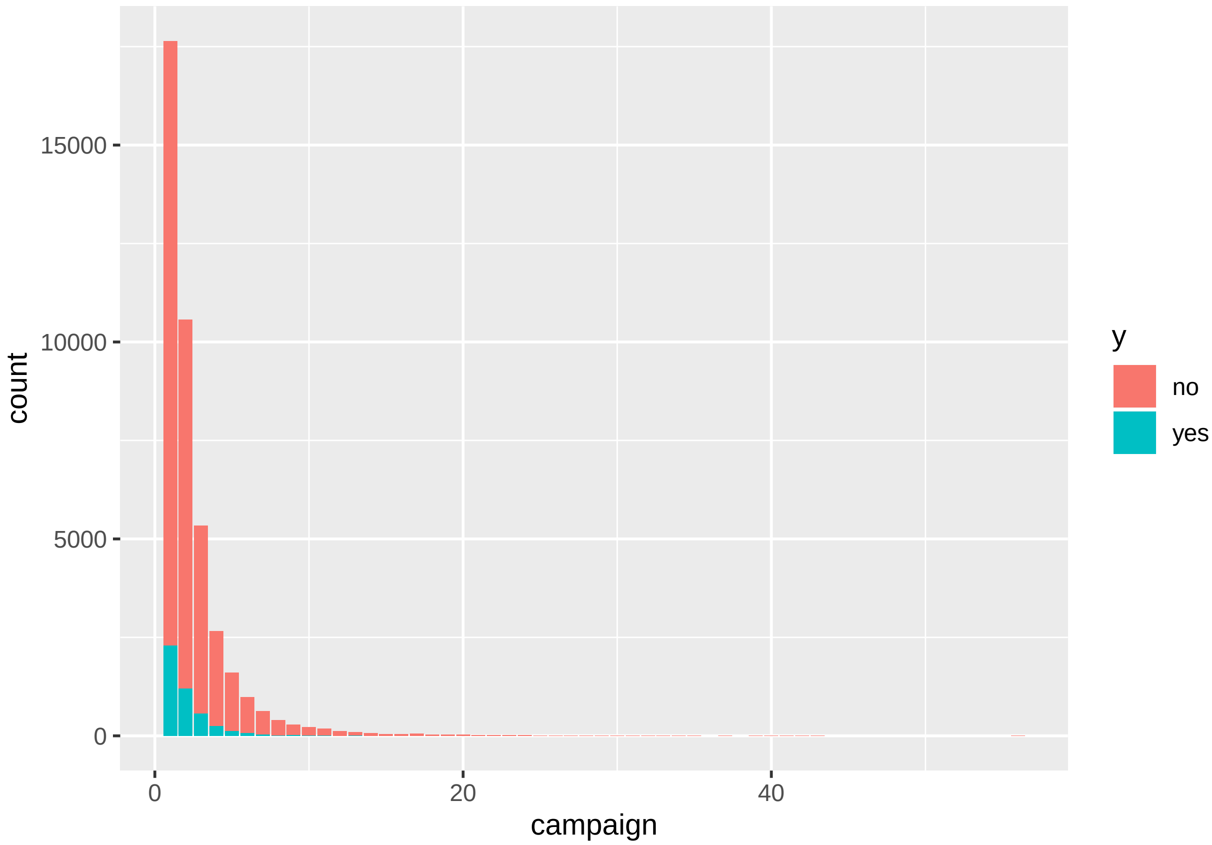
Now we will visualize the distribution of number of contacts in this **campaign** variable in the dataset.

```
# Distribution of number of contacts in this campaign variable visualization

ggplot(bankData) +
  geom_boxplot(aes(y, campaign,fill=y))+
  scale_y_continuous(breaks = c(0,10,20,30,40,50,60,70,80))
```



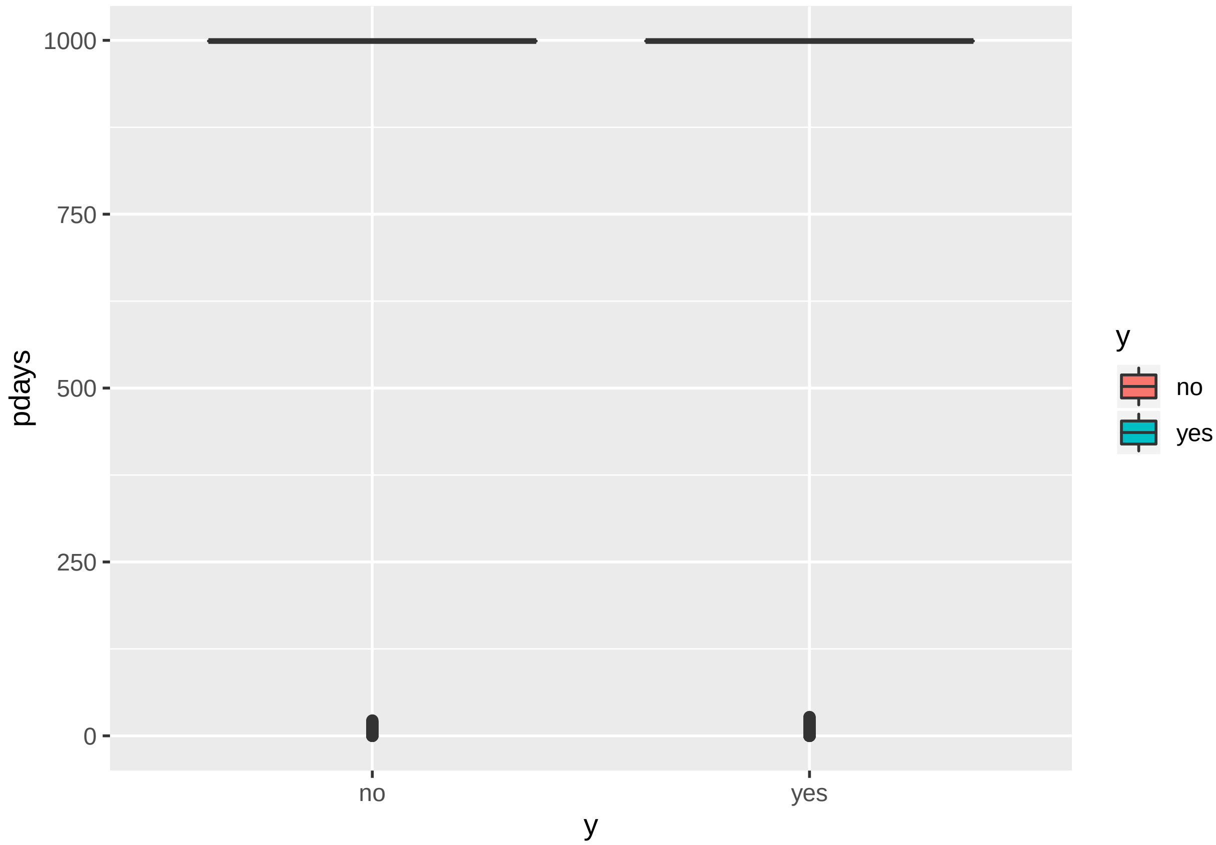
Now let's visualize how number of contacts in this campaign **campaign** variable relates to term deposit subscriptions.

```r
# Distribution of number of contacts in this campaign  to term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(campaign,fill=y))
```



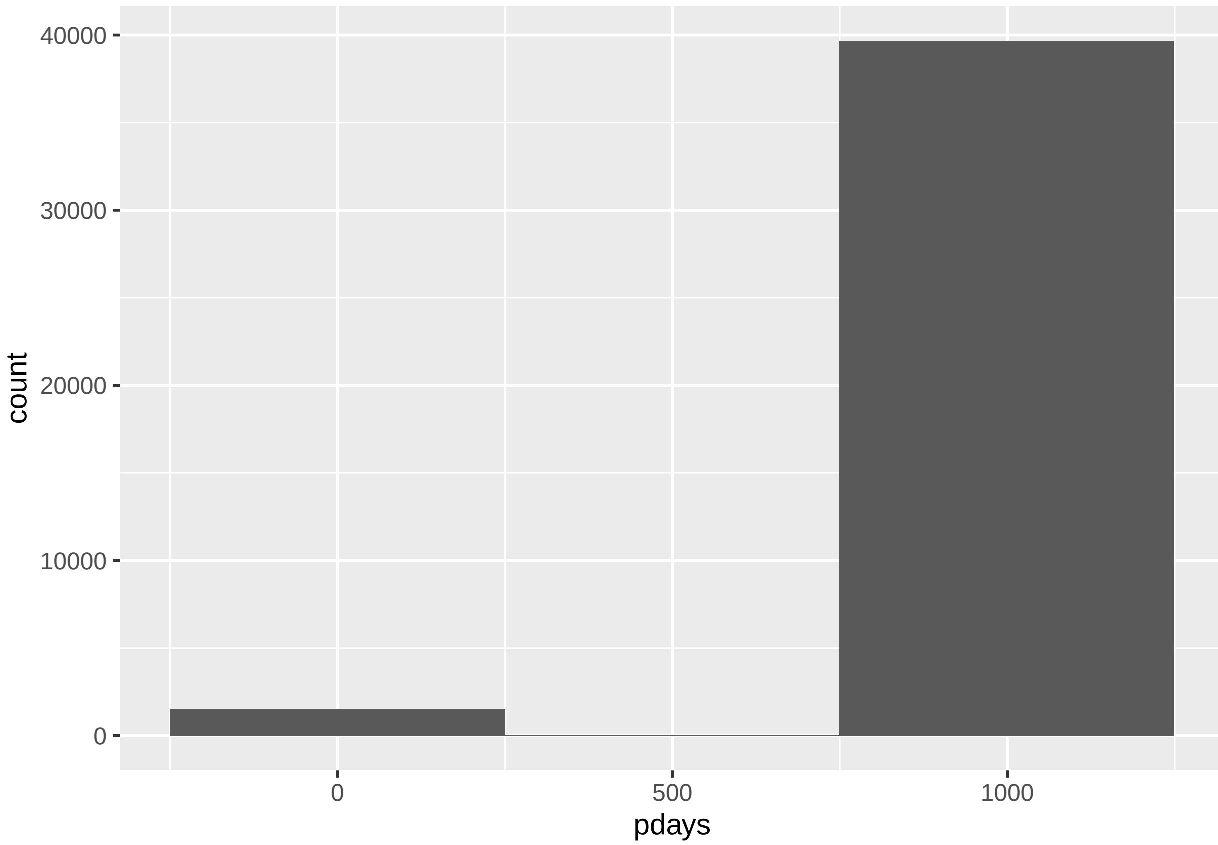
Now we will visualize the distribution of number of days since last contact **pdays** variable in the dataset.

```
# Distribution of number of days since last contact pdays variable visualization

ggplot(bankData) +
  geom_boxplot(aes(y, pdays,fill=y))
```



Now let's visualize how number of days since last contact **pdays** variable relates to term deposit subscriptions.

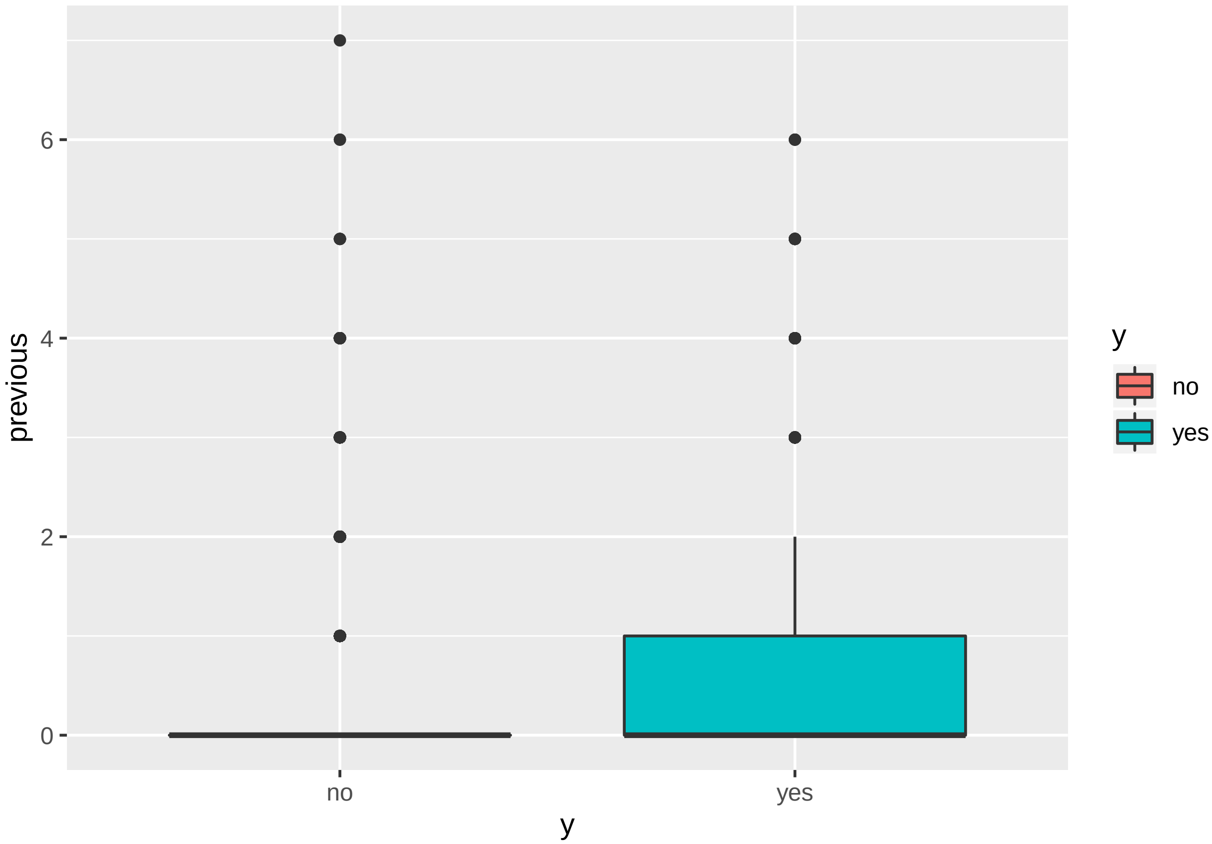
```
# Distribution of number of days since last contact pdays for term deposit
# subscription visualization

ggplot(bankData)+geom_histogram(aes(pdays),bins = 3)
```



Most of the values are 999, which translates to that the customers have never been contacted before.

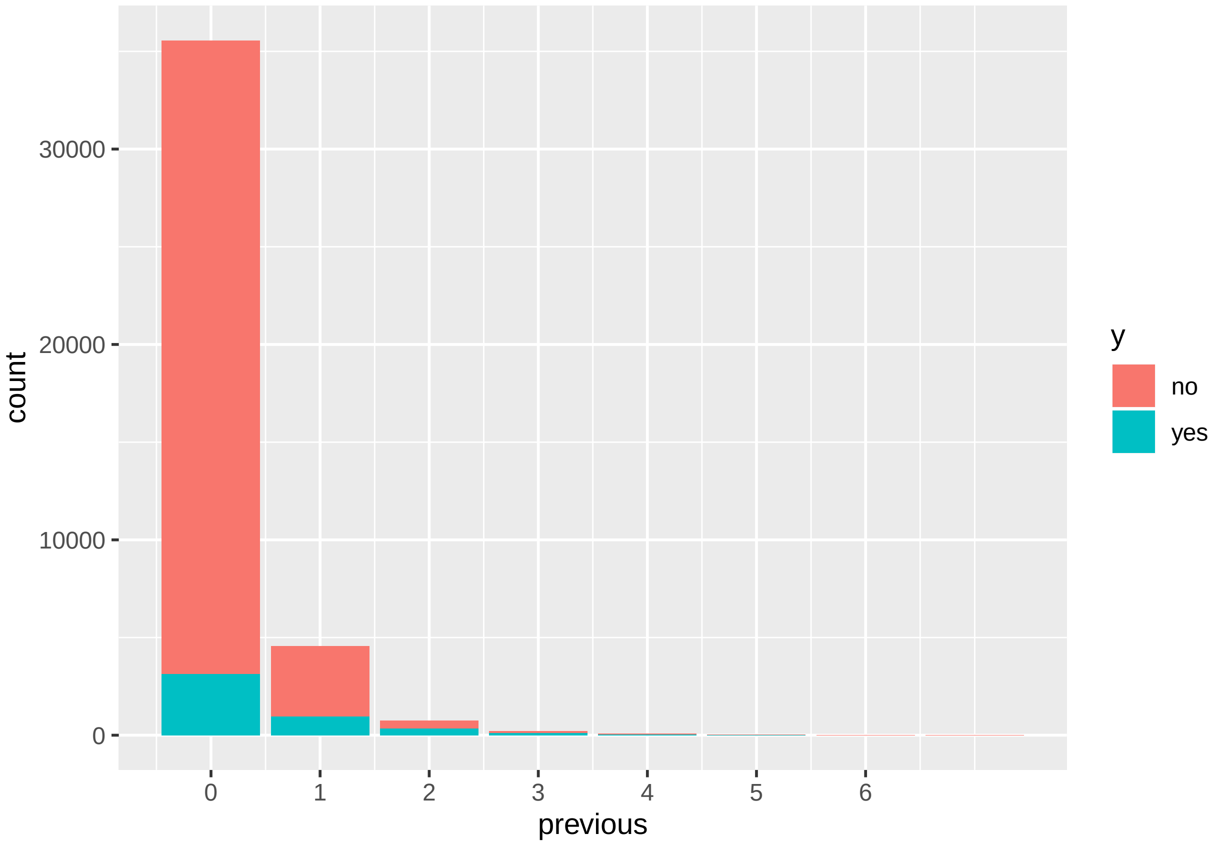
Now we will visualize the distribution of number of contacts for previous campaign **previous** variable in the dataset.

```
# Distribution of number of contacts for previous campaign previous variable visualization

ggplot(bankData) +
  geom_boxplot(aes(y, previous,fill=y))
```



Now let's visualize how number of contacts for previous campaign **previous** variable relates to term deposit subscriptions.

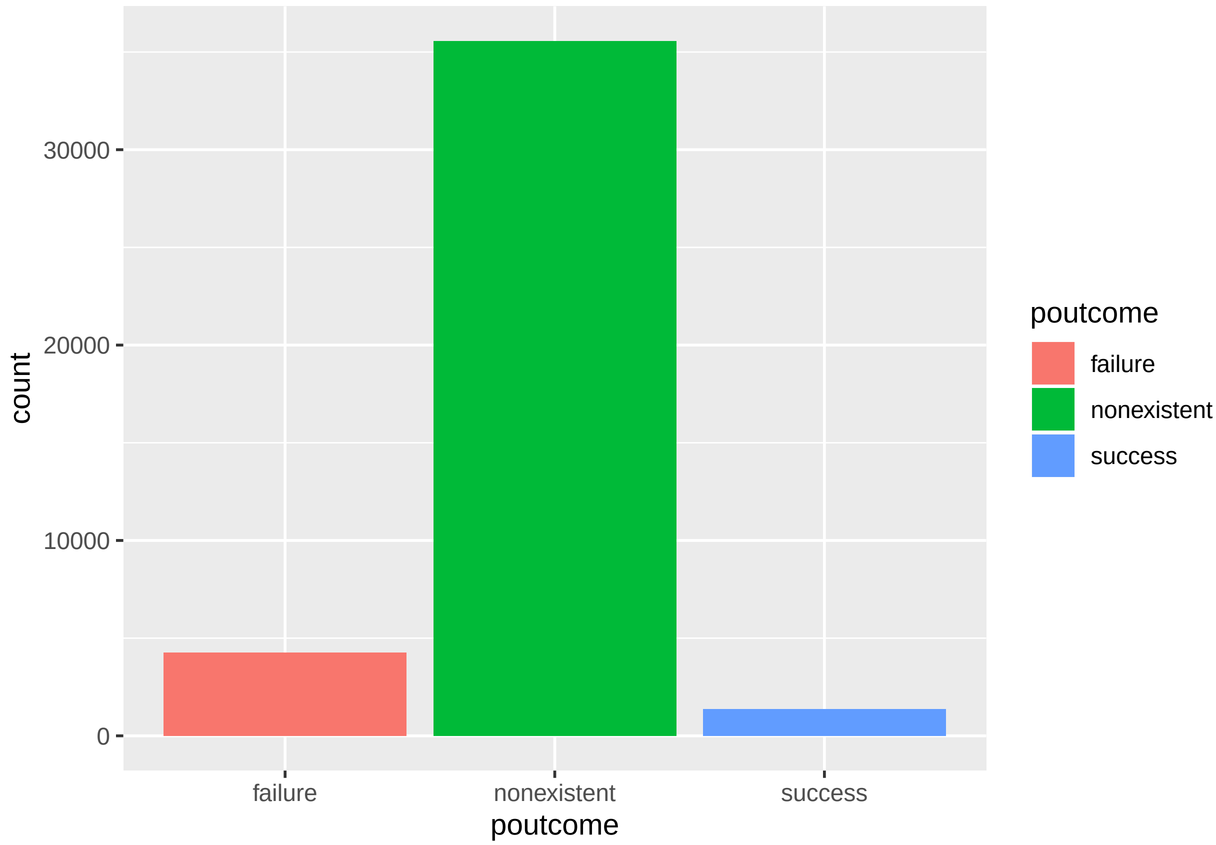
```
# Distribution of  number of contacts for previous campaign previous variable to
# term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(previous,fill=y))+scale_x_continuous(breaks = c(0,1,2,3,4,5,6))
```



The plot shows that the subscription rate for customers who have not been contacted previously is highest.

Now we will visualize the distribution of outcome of the previous marketing campaign **poutcome** variable in the dataset.

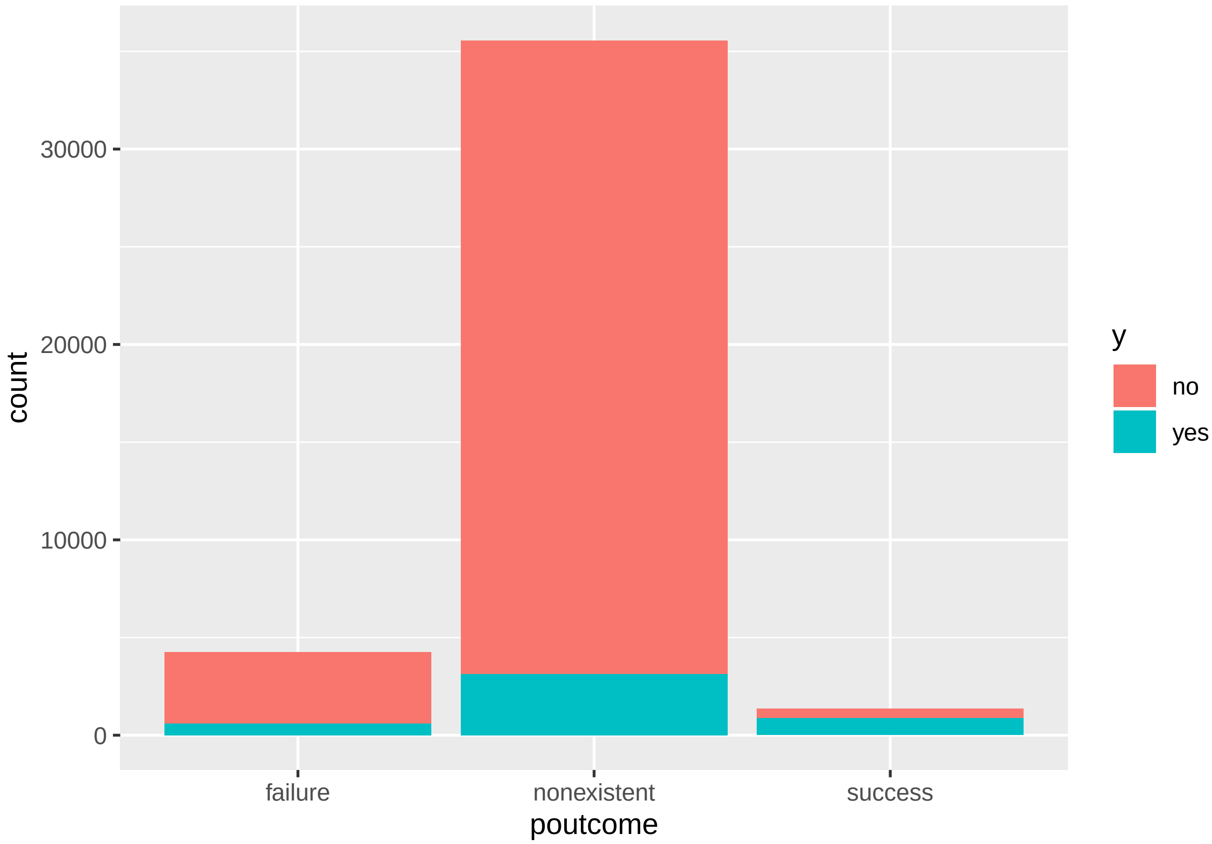
```
#Distiribution of outcome of the previous marketing campaign poutcome variable visualization

ggplot(bankData)+geom_bar(aes(poutcome,fill=poutcome))
```



The plot reveals that most of the outcome of the previous marketing campaign is nonexistent.

Now let's visualize how the outcome of the previous marketing campaign **poutcome** variable relates to term deposit subscriptions.

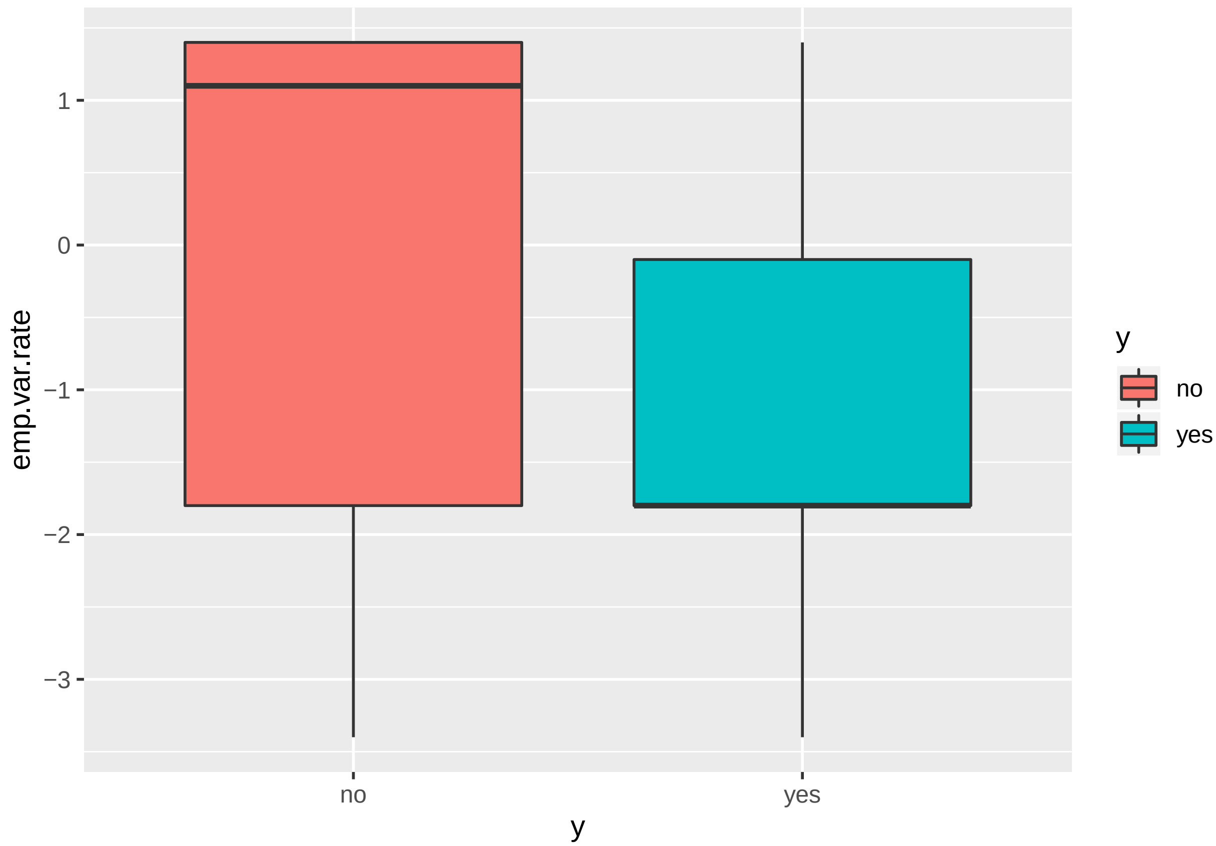
```
#Distiribution of outcome of the previous marketing campaign poutcome to
#term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(poutcome,fill=y))
```



The plot reveals that the subscription of term deposit is highest for customers who had nonexistent poutcome.

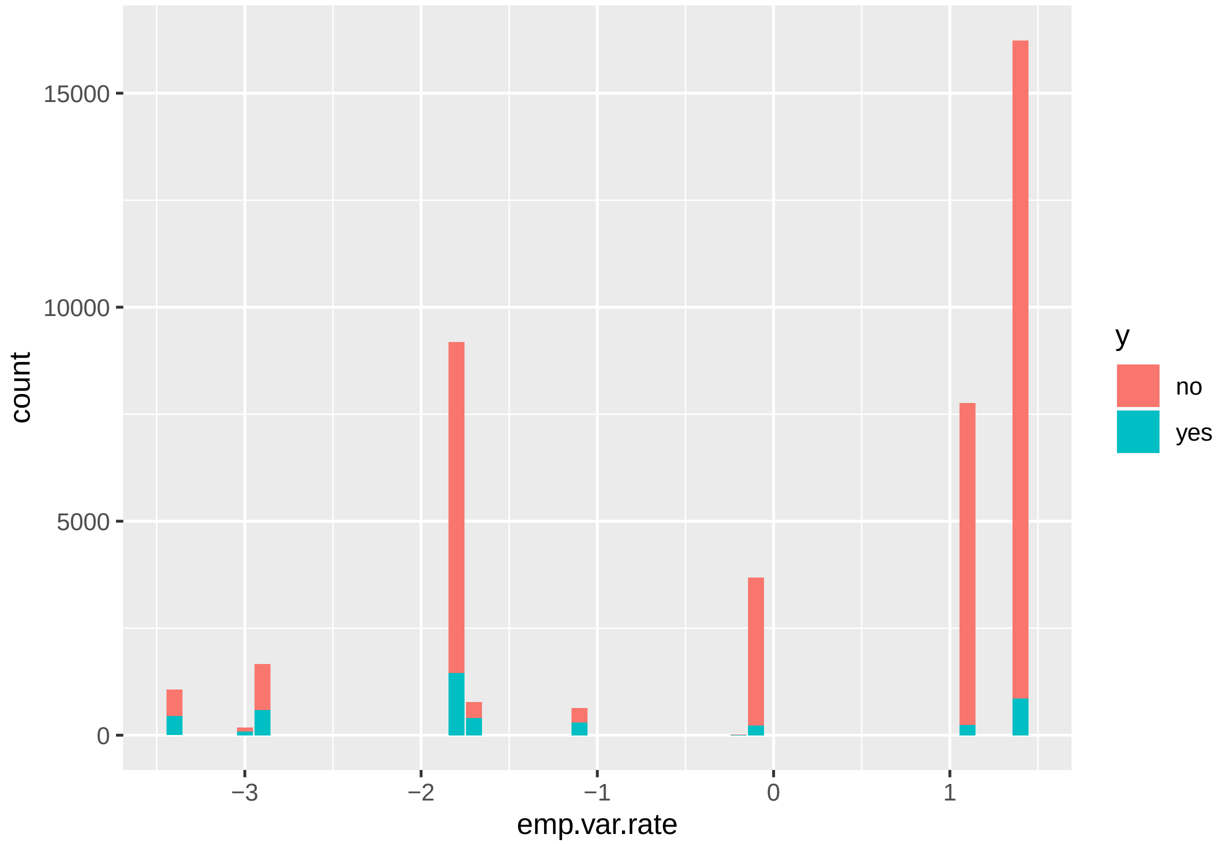
Now we will visualize the distribution of employment variation rate - quarterly indicator **emp.var.rate** variable in the dataset.

```
# Distribution of employment variation rate - quarterly indicator emp.var.rate visualization

ggplot(bankData) +
  geom_boxplot(aes(y, emp.var.rate,fill=y))
```



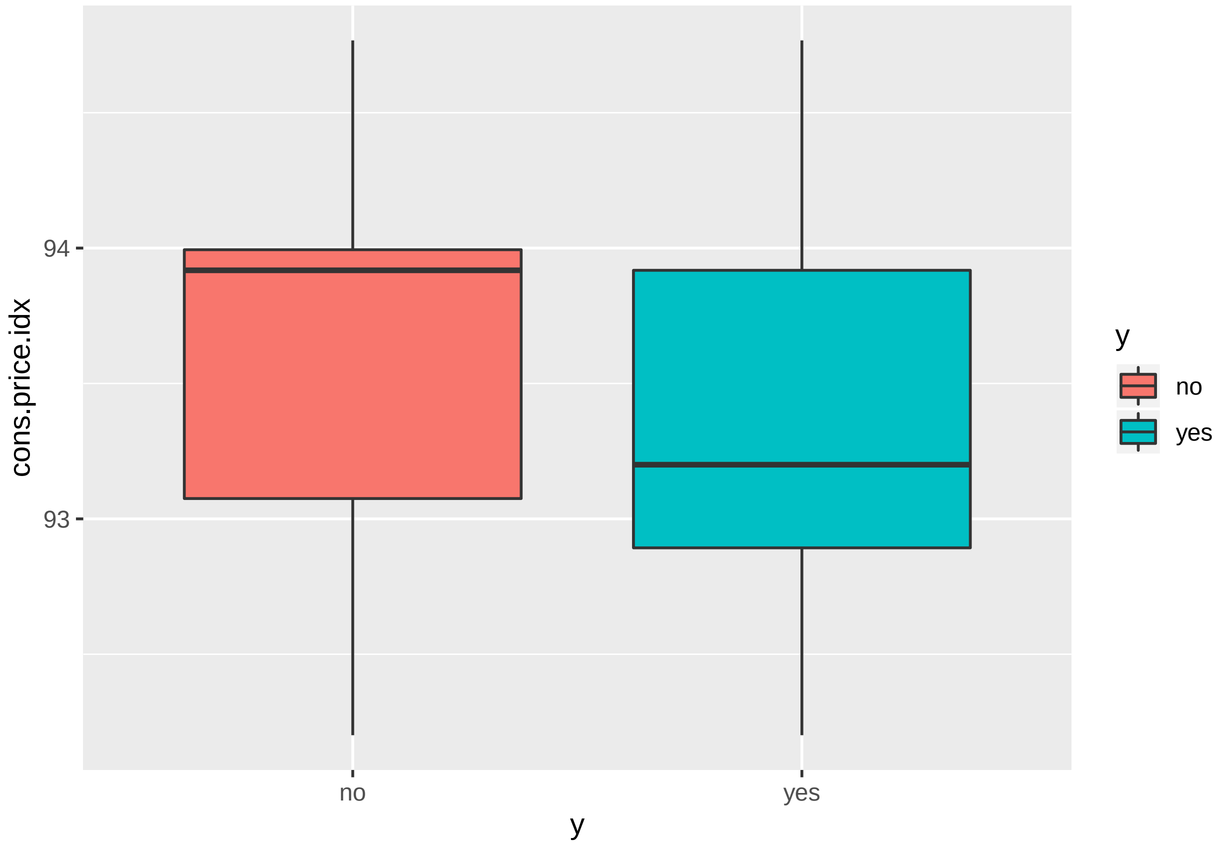
Now let's visualize how employment variation rate - quarterly indicator **emp.var.rate** variable relates to term deposit subscriptions.

```
# Distribution of employment variation rate - quarterly indicator emp.var.rate
#for term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(emp.var.rate,fill=y))
```



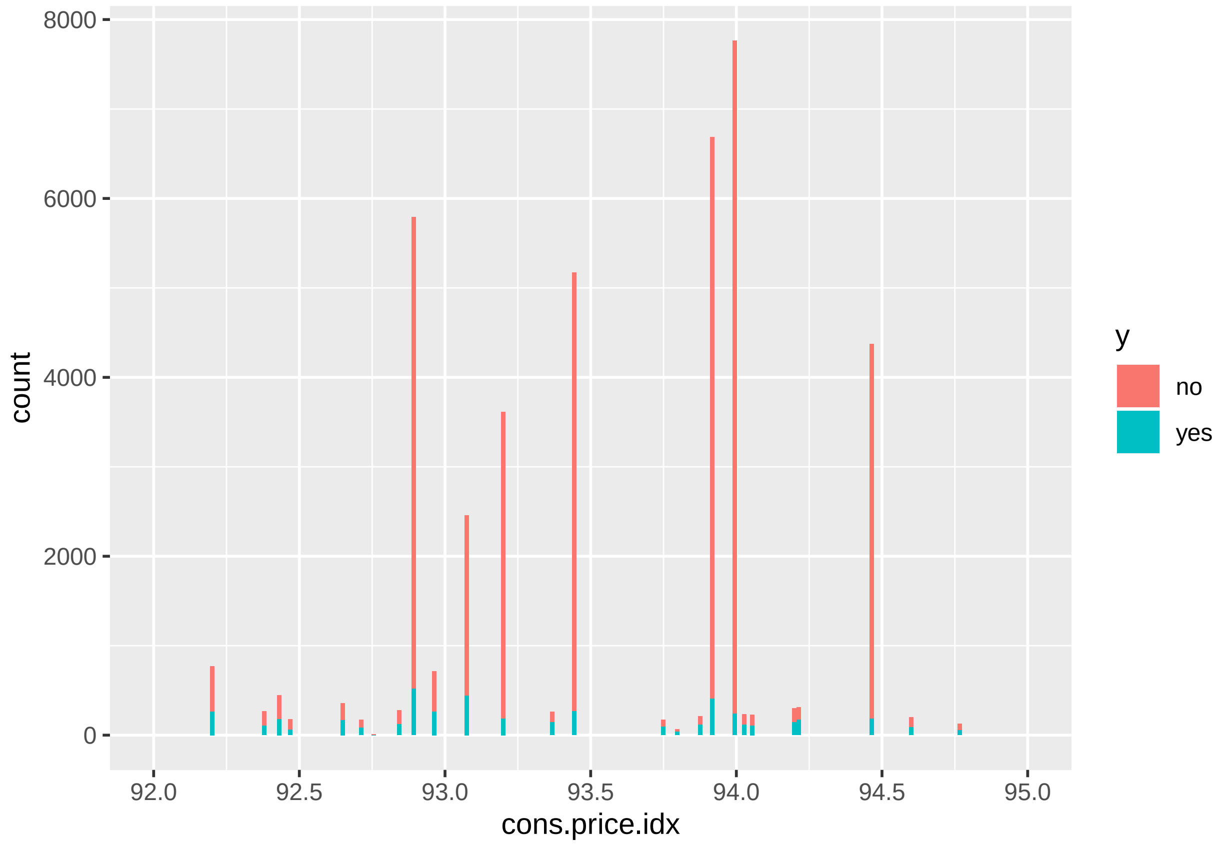
Now we will visualize the distribution of consumer price index - monthly indicator **cons.price.idx** variable in the dataset.

```
# Distribution of consumer price index - monthly indicator cons.price.idx visualization

ggplot(bankData) +
  geom_boxplot(aes(y, cons.price.idx,fill=y))
```



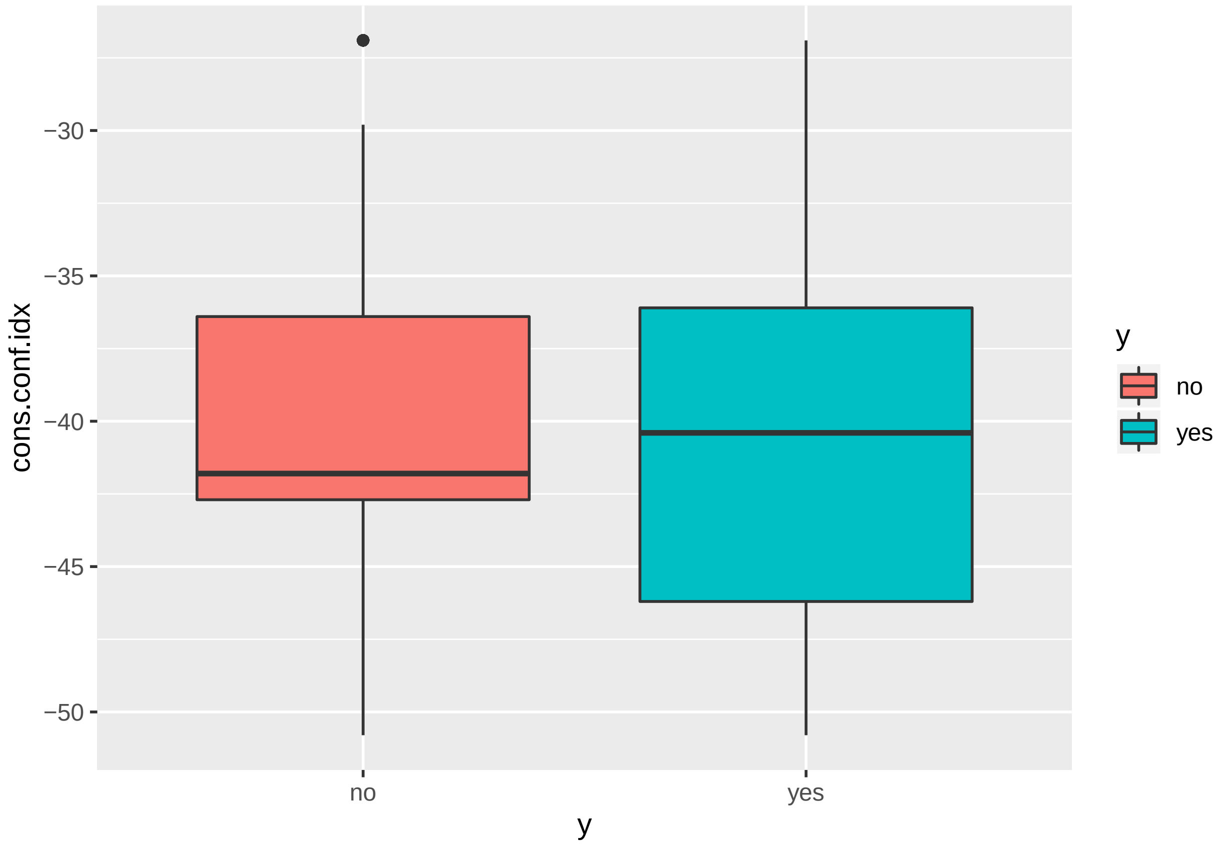
Now let's visualize how consumer price index - monthly indicator **cons.price.idx** variable relates to term deposit subscriptions.

```
# Distribution of consumer price index - monthly indicator cons.price.idx
# for term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(cons.price.idx,fill=y))+
  scale_x_continuous(limits =c(92,95),breaks = seq(92,95,by=0.5))
```



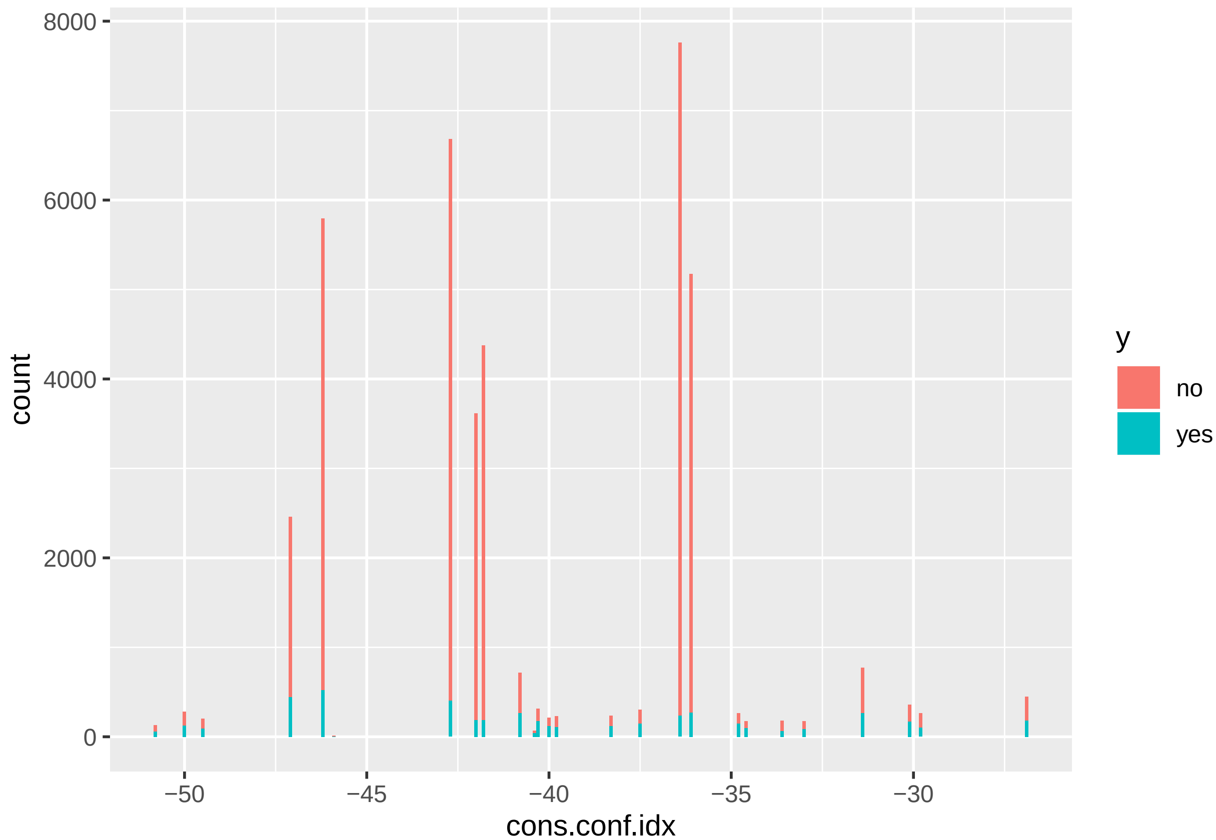
Now we will visualize the distribution of consumer confidence index - monthly indicator **cons.conf.idx** variable in the dataset.

```
# Distribution of consumer confidence index - monthly indicator cons.conf.idx visualization

ggplot(bankData) +
  geom_boxplot(aes(y, cons.conf.idx,fill=y))
```



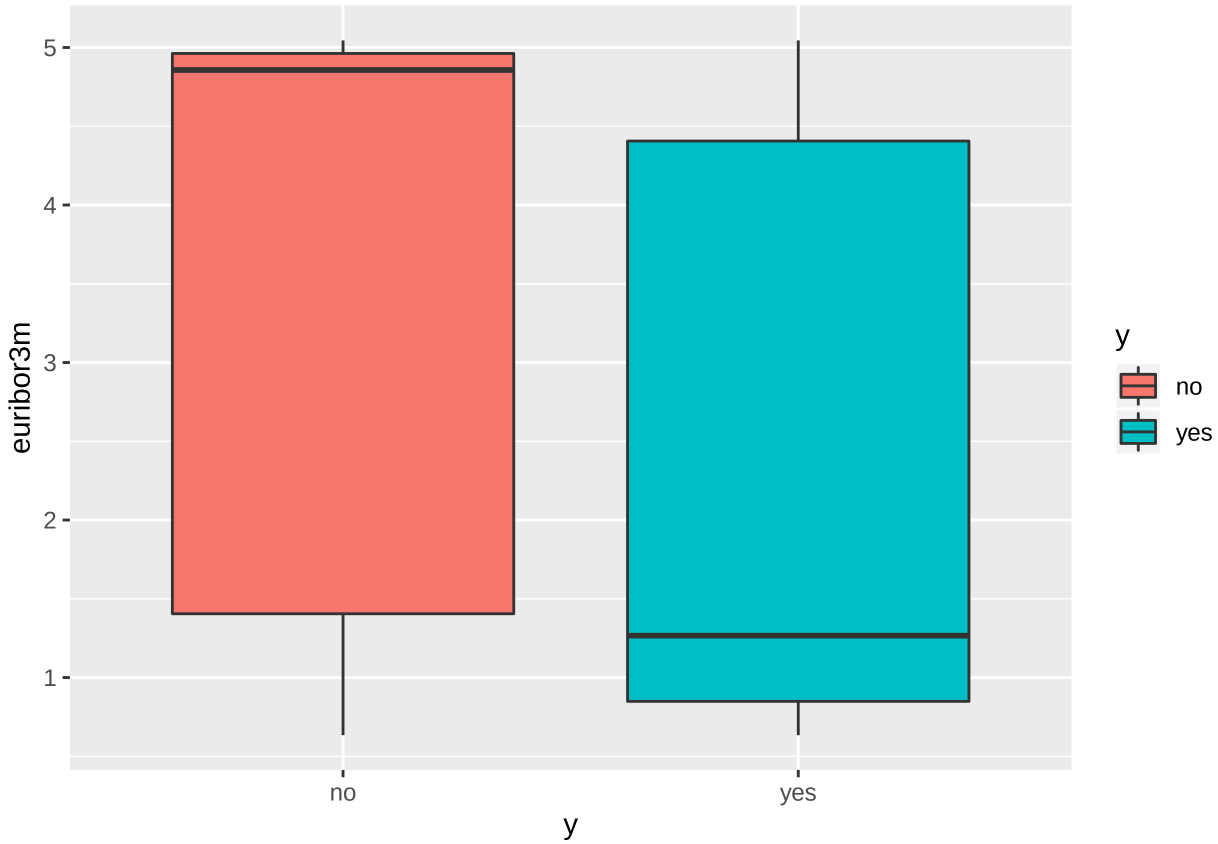
Now let's visualize how consumer confidence index - monthly indicator **cons.conf.idx** variable relates to term deposit subscriptions.

```
# Distribution of consumer confidence index - monthly indicator cons.conf.idx
# for term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(cons.conf.idx,fill=y))
```



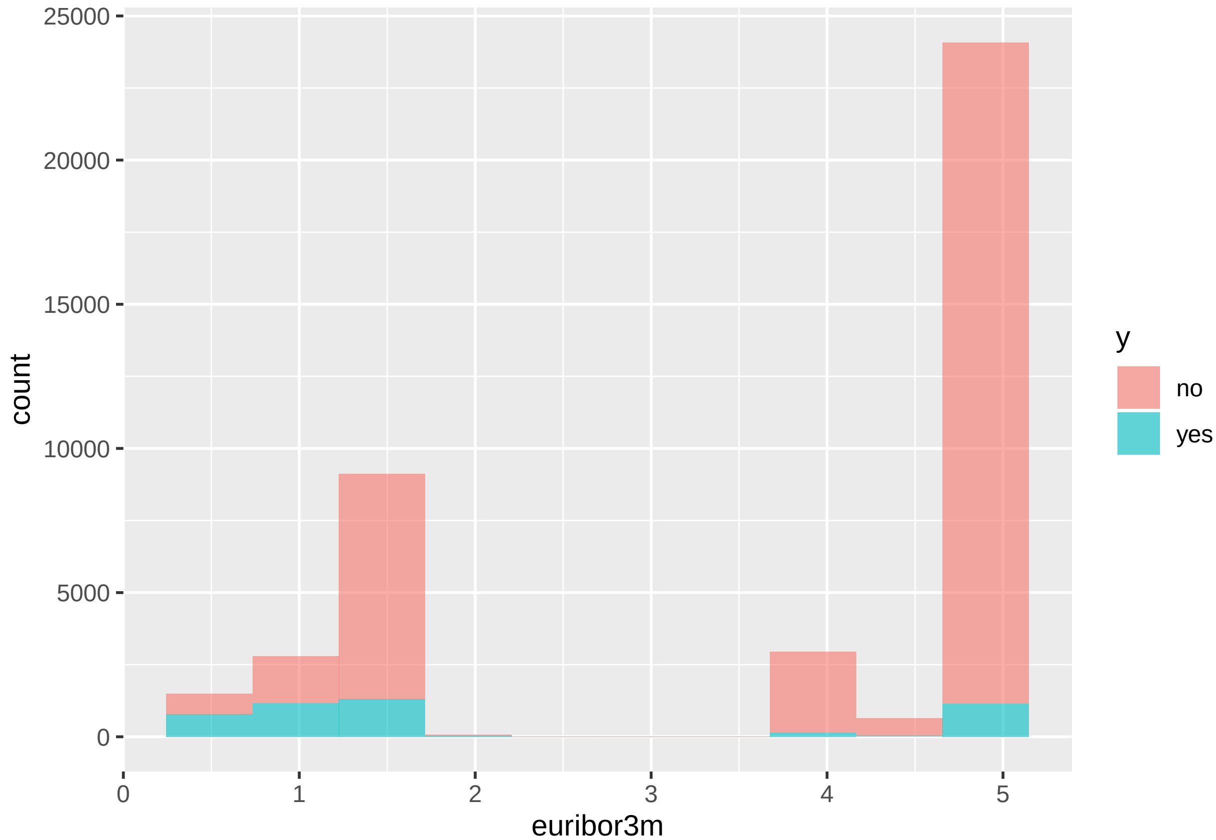
Now we will visualize the distribution of euribor 3 month rate - daily indicator **euribor3m** variable in the dataset.

```
# Distribution of euribor 3 month rate - daily indicator euribor3m visualization

ggplot(bankData) +
  geom_boxplot(aes(y, euribor3m,fill=y))
```



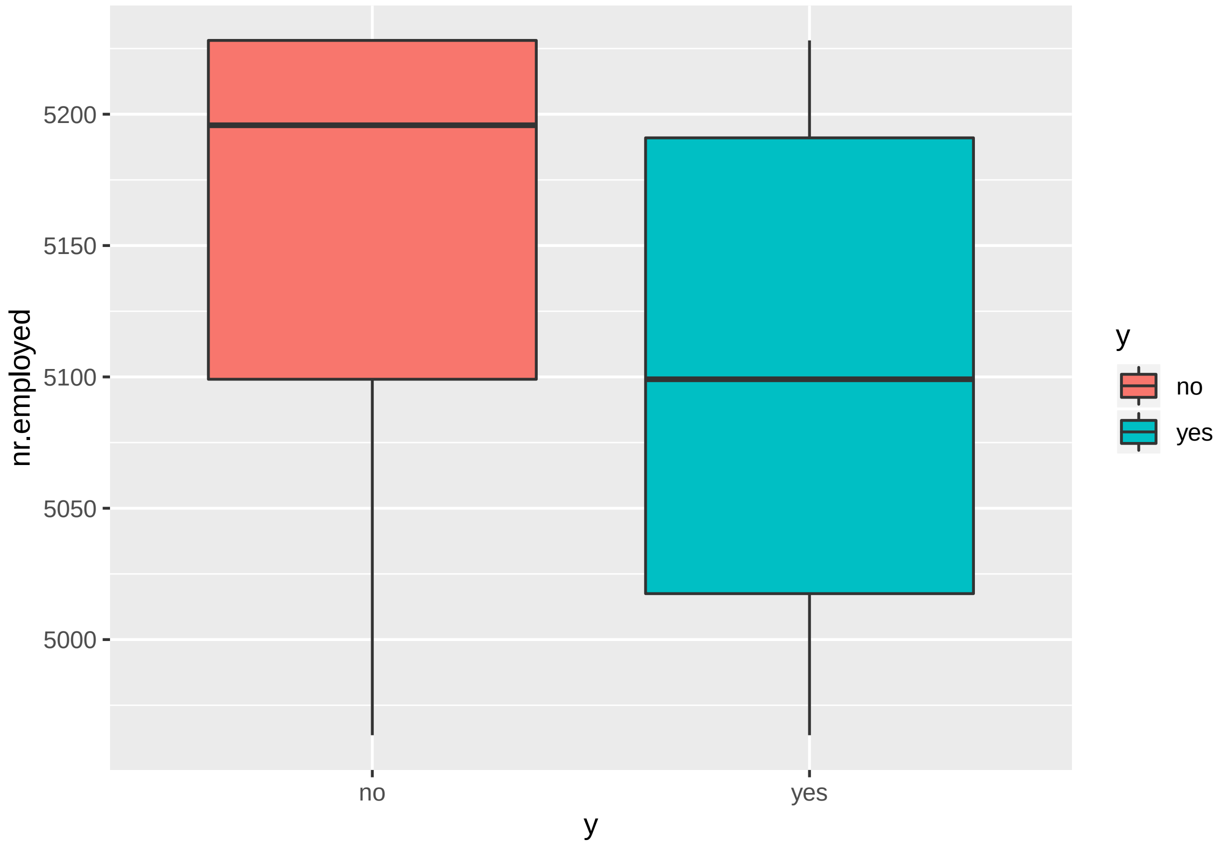
Now let's visualize how euribor 3 month rate - daily indicator **euribor3m** variable relates to term deposit subscriptions.

```
# Distribution of euribor 3 month rate - daily indicator euribor3m
# for term deposit subscription visualization

ggplot(bankData)+geom_histogram(aes(euribor3m,fill=y),bins = 10,alpha =0.6)
```



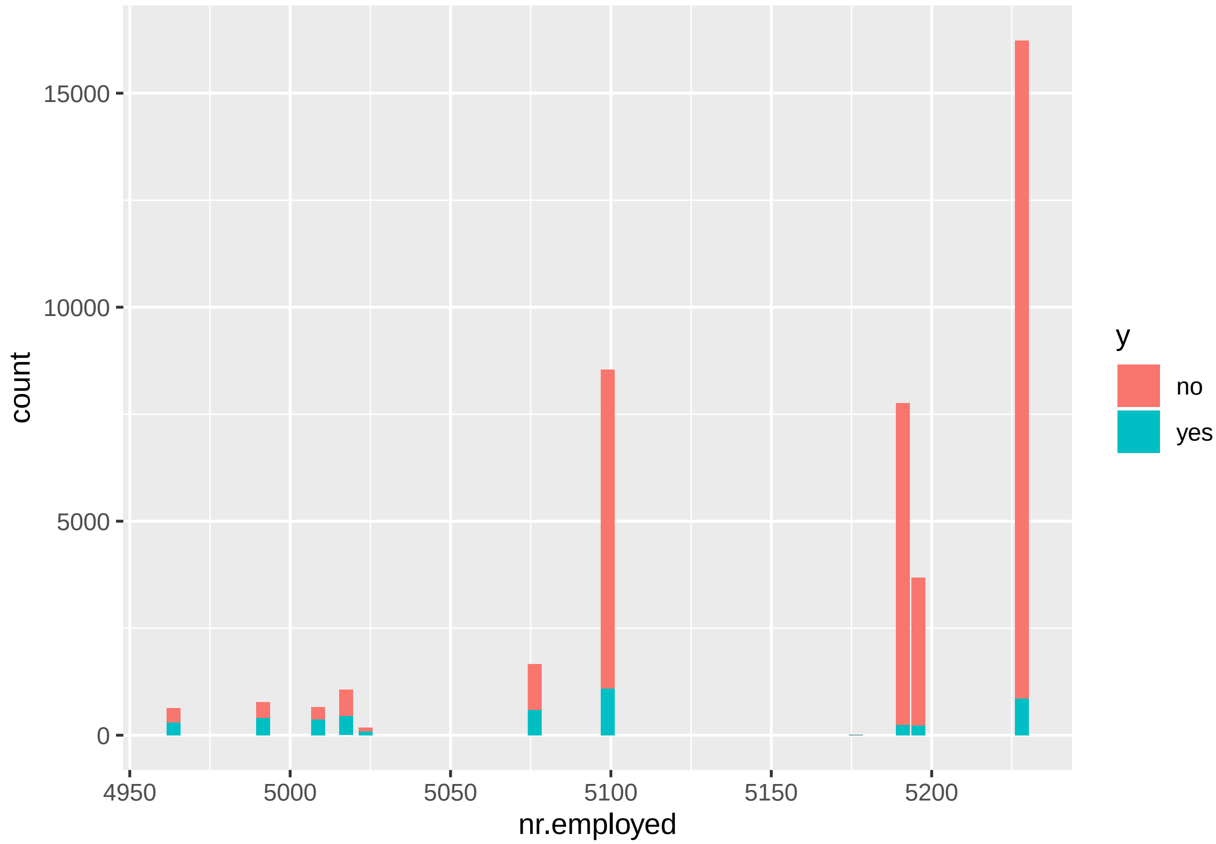
Now we will visualize the distribution of number of employees - quarterly indicator **nr.employed** variable in the dataset.

```
# Distribution of number of employees - quarterly indicator nr.employed visualization

ggplot(bankData) +
  geom_boxplot(aes(y, nr.employed,fill=y))
```



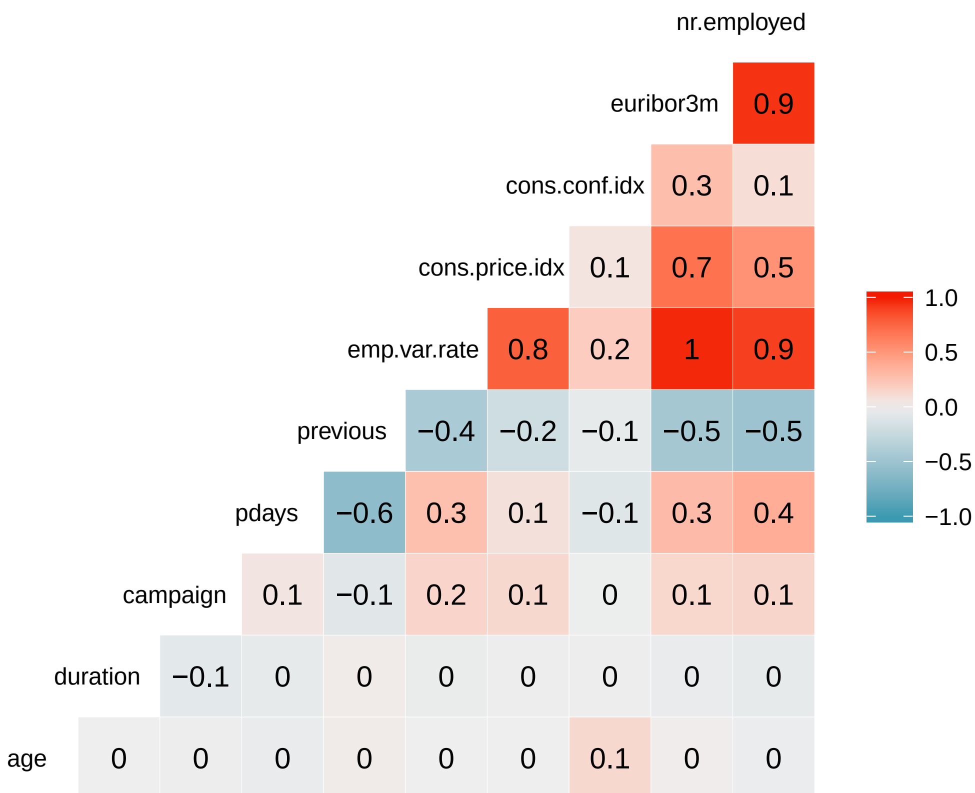
Now let's visualize how number of employees - quarterly indicator **nr.employed** variable relates to term deposit subscriptions.

```
# Distribution of number of employees - quarterly indicator nr.employed
# for term deposit subscription visualization

ggplot(bankData)+geom_bar(aes(nr.employed,fill=y))
```



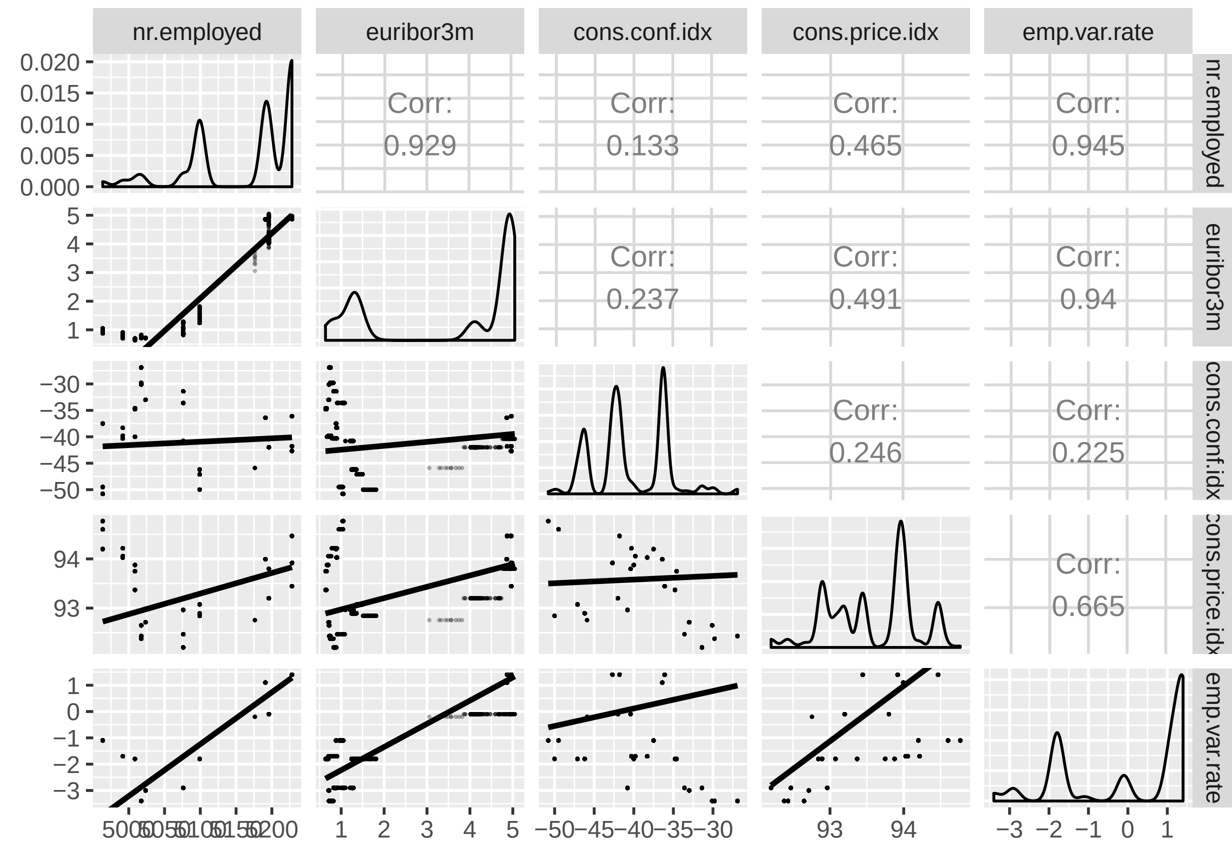
Let us visualize the correlation plot for numeric variables.

```
ggcorr(bankData,  label = TRUE, hjust = 0.75,size = 3)
```



Let us visualize the pair wise correlation plot for numeric variables.

```
aa <-bankData %>%select(nr.employed, euribor3m, cons.conf.idx,cons.price.idx,emp.var.rate)


ggpairs(aa,columns = 1:5 , lower = list(continuous = wrap("smooth", alpha = 0.3, size=0.1)),
        upper = list(continuous = wrap("cor", method= "spearman")))
```



Data Processing

We had checked earlier that there were no NAs in the data. Now we will check for duplicate rows in the dataset.

```
#Check for Duplicate Rows
sum(duplicated(bankData))
```

```
## [1] 12
```

So we find that there are 12 rows which are duplicate. Let us remove the duplicate rows.

```
#create a new dataset bankData_dist with distinct dataset only i.e remove the duplicates
bankData_dist <- bankData %>% distinct()

str(bankData_dist)
```

```
## 'data.frame':    41176 obs. of  21 variables:
##  $ age           : int  56 57 37 40 56 45 59 41 24 25 ...
##  $ job           : chr  "housemaid" "services" "services" "admin." ...
##  $ marital       : chr  "married" "married" "married" "married" ...
##  $ education     : chr  "basic.4y" "high.school" "high.school" "basic.6y" ...
##  $ default       : chr  "no" "unknown" "no" "no" ...
##  $ housing       : chr  "no" "no" "yes" "no" ...
##  $ loan          : chr  "no" "no" "no" "no" ...
##  $ contact       : chr  "telephone" "telephone" "telephone" "telephone" ...
##  $ month         : chr  "may" "may" "may" "may" ...
##  $ day_of_week   : chr  "mon" "mon" "mon" "mon" ...
##  $ duration      : int  261 149 226 151 307 198 139 217 380 50 ...
##  $ campaign      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ pdays         : int  999 999 999 999 999 999 999 999 999 999 ...
##  $ previous      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome      : chr  "nonexistent" "nonexistent" "nonexistent" "nonexistent" ...
##  $ emp.var.rate  : num  1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 ...
##  $ cons.price.idx: num  94 94 94 94 94 ...
##  $ cons.conf.idx : num  -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 ...
##  $ euribor3m     : num  4.86 4.86 4.86 4.86 4.86 ...
##  $ nr.employed   : num  5191 5191 5191 5191 5191 ...
##  $ y             : chr  "no" "no" "no" "no" ...
```

Now let us convert the output variable from Yes/No to 1/0 .

```
#converting the output variable y from yes/no to 1/0

bankData_dist$y = ifelse(bankData_dist$y=='yes',1,0)
```

Next we will convert the character variables to factor variables.

```
# converting character variables to factor variables
bankData_dist$y <- as.factor(bankData_dist$y)
bankData_dist$job <- as.factor(bankData_dist$job)
bankData_dist$marital <- as.factor(bankData_dist$marital)
bankData_dist$education <- as.factor(bankData_dist$education)
bankData_dist$default <- as.factor(bankData_dist$default)
bankData_dist$housing <- as.factor(bankData_dist$housing)
bankData_dist$loan <- as.factor(bankData_dist$loan)
bankData_dist$contact <- as.factor(bankData_dist$contact)
bankData_dist$month <- as.factor(bankData_dist$month)
```

```r
bankData_dist$day_of_week <- as.factor(bankData_dist$day_of_week)
bankData_dist$poutcome <- as.factor(bankData_dist$poutcome)
```

We will convert the remaining columns to numeric.

```r
#converting the columns to numeric
bankData_dist$age <- as.numeric(bankData_dist$age)
bankData_dist$duration <- as.numeric(bankData_dist$duration)
bankData_dist$campaign <- as.numeric(bankData_dist$campaign)
bankData_dist$pdays <- as.numeric(bankData_dist$pdays)
bankData_dist$previous <- as.numeric(bankData_dist$previous)
bankData_dist$emp.var.rate <- as.numeric(bankData_dist$emp.var.rate)
bankData_dist$cons.price.idx <- as.numeric(bankData_dist$cons.price.idx)
bankData_dist$cons.conf.idx <- as.numeric(bankData_dist$cons.conf.idx)
bankData_dist$nr.employed <- as.numeric(bankData_dist$nr.employed)
#checking the variables in the dataset
str(bankData_dist)
```

```
## 'data.frame':    41176 obs. of  21 variables:
##  $ age           : num  56 57 37 40 56 45 59 41 24 25 ...
##  $ job           : Factor w/ 12 levels "admin.","blue-collar",..: 4 8 8 1 8 8 1 2 10 8 ...
##  $ marital       : Factor w/ 4 levels "divorced","married",..: 2 2 2 2 2 2 2 2 3 3 ...
##  $ education     : Factor w/ 8 levels "basic.4y","basic.6y",..: 1 4 4 2 4 3 6 8 6 4 ...
##  $ default       : Factor w/ 3 levels "no","unknown",..: 1 2 1 1 1 2 1 2 1 1 ...
##  $ housing       : Factor w/ 3 levels "no","unknown",..: 1 1 3 1 1 1 1 1 3 3 ...
##  $ loan          : Factor w/ 3 levels "no","unknown",..: 1 1 1 1 3 1 1 1 1 1 ...
##  $ contact       : Factor w/ 2 levels "cellular","telephone": 2 2 2 2 2 2 2 2 2 2 ...
##  $ month         : Factor w/ 10 levels "apr","aug","dec",..: 7 7 7 7 7 7 7 7 7 7 ...
##  $ day_of_week   : Factor w/ 5 levels "fri","mon","thu",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ duration      : num  261 149 226 151 307 198 139 217 380 50 ...
##  $ campaign      : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ pdays         : num  999 999 999 999 999 999 999 999 999 999 ...
##  $ previous      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome      : Factor w/ 3 levels "failure","nonexistent",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ emp.var.rate  : num  1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 ...
##  $ cons.price.idx: num  94 94 94 94 94 ...
##  $ cons.conf.idx : num  -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 ...
##  $ euribor3m     : num  4.86 4.86 4.86 4.86 4.86 ...
##  $ nr.employed   : num  5191 5191 5191 5191 5191 ...
##  $ y             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

Now we will split the dataset in two parts namely training set and test set. The test set will be 10% of the data while remaining 90% will be training set.

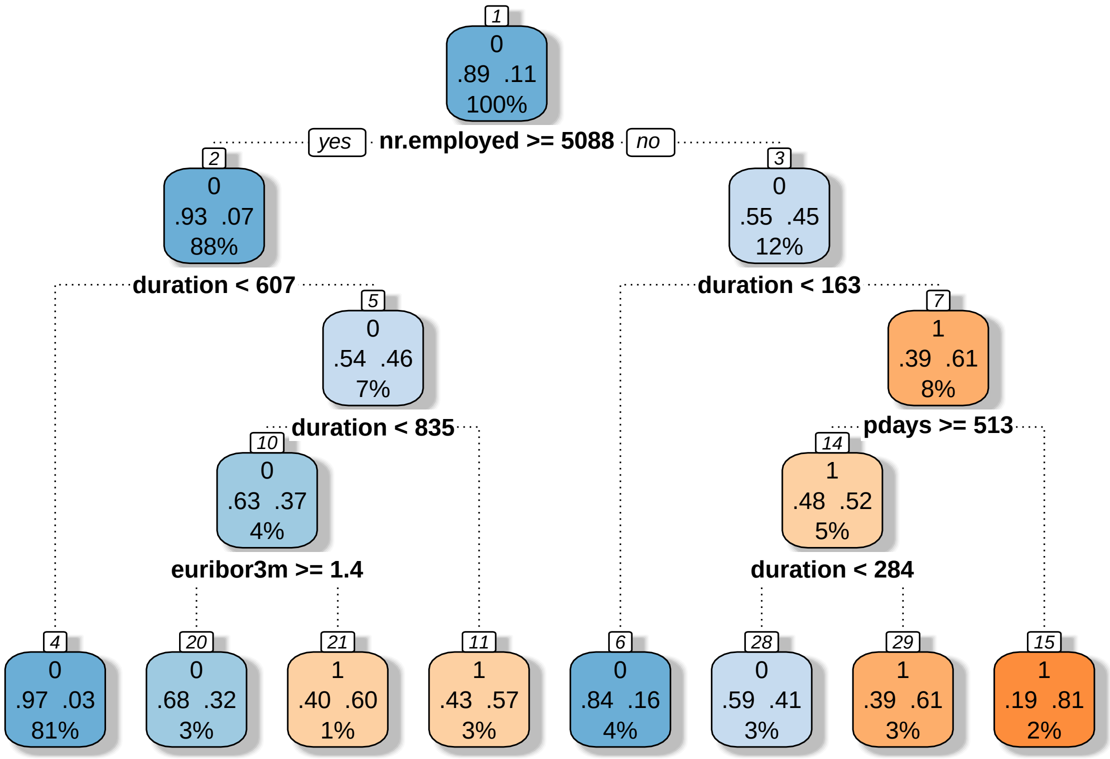
```r
# Splitting the dataset into two parts (training set and test set)
# The test set will be 10% of the data while 90% will be training set

set.seed(1, sample.kind="Rounding")
test_index <- createDataPartition(bankData_dist$y, times = 1, p = 0.1, list = FALSE)
train_bankData_dist <- bankData_dist[-test_index,]
test_bankData_dist <- bankData_dist[test_index,]
```

Now we are ready to fit the model. The first model we will try to fit is the classification and regression tree.

```
##############################################################
###   Model 1: Classification and Regression Tree [CART]
##############################################################
# Classification and Regression Trees
 bank_cart<-rpart(y ~ ., train_bankData_dist , method = 'class')
```

Now we will plot the classification.

```
#Plotting the tree
par(mfrow=c(1,1))
fancyRpartPlot( bank_cart , digits=2 , palettes = c("Blues", "Oranges"))
```



Rattle 2020−May−31 12:11:07 i501256

We run the prediction and then create the confusion matrix

```
#prediction on the test set
cart_pred <- predict( bank_cart , test_bankData_dist , type = "class")
cart_prob <- predict( bank_cart , test_bankData_dist , type = "prob")

# Confusion matrix
confusionMatrix(cart_pred , test_bankData_dist$y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3507  222
##          1  147  242
##
```

```
##                Accuracy : 0.9104
##                  95% CI : (0.9013, 0.9189)
##     No Information Rate : 0.8873
##     P-Value [Acc > NIR] : 7.836e-07
##
##                   Kappa : 0.5179
##
##  Mcnemar's Test P-Value : 0.000117
##
##             Sensitivity : 0.9598
##             Specificity : 0.5216
##          Pos Pred Value : 0.9405
##          Neg Pred Value : 0.6221
##              Prevalence : 0.8873
##          Detection Rate : 0.8516
##    Detection Prevalence : 0.9055
##       Balanced Accuracy : 0.7407
##
##        'Positive' Class : 0
##
```

```r
#Storing the result of the confusion matrix
cm_cart <-confusionMatrix(cart_pred , test_bankData_dist$y)
```

Next we will do a cross table validation for the model.

```r
### Cross table validation for CART

CrossTable(test_bankData_dist$y, cart_pred,
           prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
           dnn = c('actual default', 'predicted default'))
```

```
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
## =======================================
##                   predicted default
## actual default        0       1   Total
## ---------------------------------------
## 0                  3507     147    3654
##                   0.852   0.036
## ---------------------------------------
## 1                   222     242     464
##                   0.054   0.059
## ---------------------------------------
## Total              3729     389    4118
## =======================================
```

Now we will fit the K-Nearest Neighbor model.

```
##############################################################
###   Model 2: K- Nearest Neighbor [KNN]
##############################################################

bank_knn <- train(y ~ ., data = train_bankData_dist, method = "knn",
                  maximize = TRUE,
                  trControl = trainControl(method = "cv", number = 10),
                  preProcess=c("center", "scale"))
```

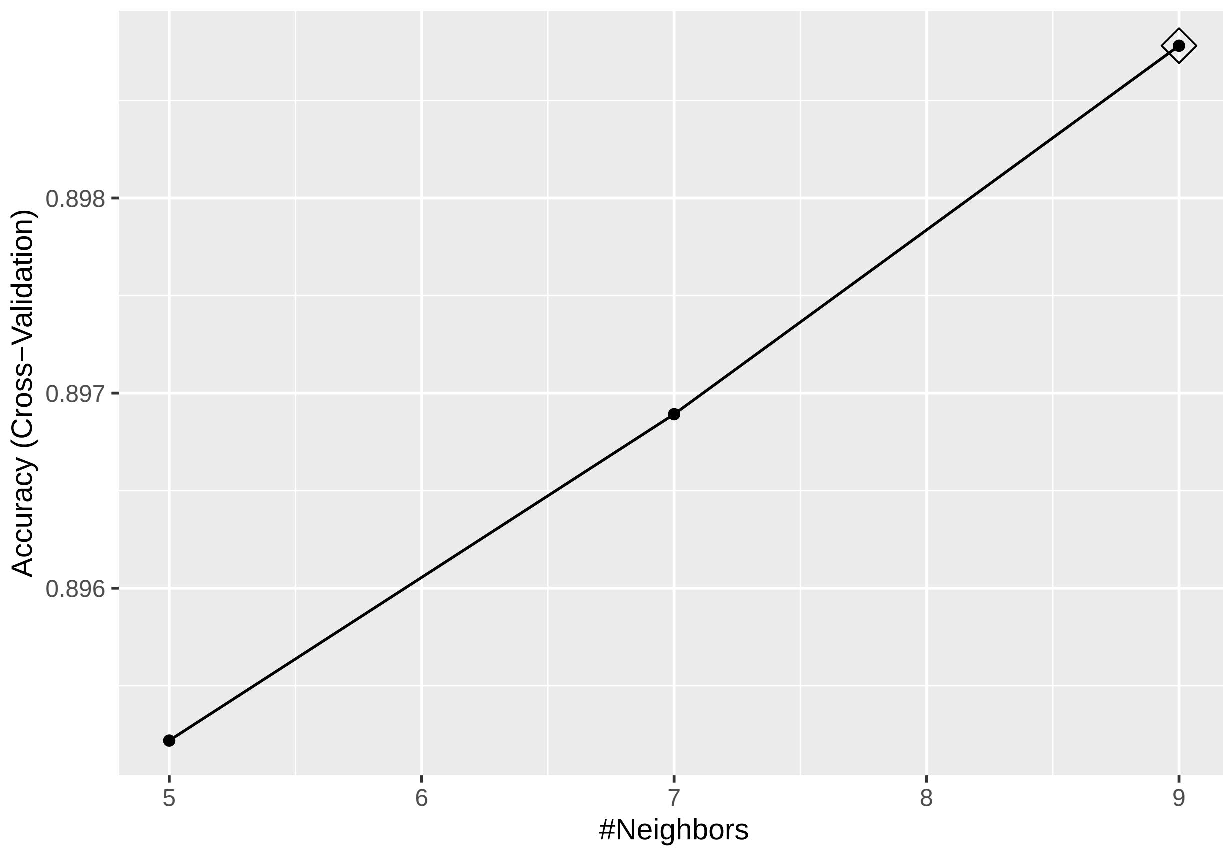
Let's check the best value for the tuning parameter.

```
# Let's check the best value for tuning parameter
bank_knn$bestTune
```

```
##   k
## 3 9
```

Now we will plot the results using ggplot function. The argument highlight highlights the max used in cross validation.

```
# Plotting the tuning parameter
ggplot(bank_knn, highlight = TRUE)
```



We can also check the details of the final model used.

```
# Final model details
bank_knn$finalModel
```

```
## 9-nearest neighbor model
```

```
## Training set outcome distribution:
##
##     0     1
## 32883  4175
```

Now we will use the model to do prediction on the test set.

```r
# Prediction on the test set
predictedkNN <- predict(bank_knn , newdata = test_bankData_dist)
```

Generate the confusion matrix and store it for future comparison of models.

```r
# Confusion matrix
confusionMatrix(predictedkNN , test_bankData_dist$y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3564  324
##          1   90  140
##
##                Accuracy : 0.8995
##                  95% CI : (0.8899, 0.9085)
##     No Information Rate : 0.8873
##     P-Value [Acc > NIR] : 0.006704
##
##                   Kappa : 0.3553
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9754
##             Specificity : 0.3017
##          Pos Pred Value : 0.9167
##          Neg Pred Value : 0.6087
##              Prevalence : 0.8873
##          Detection Rate : 0.8655
##    Detection Prevalence : 0.9441
##       Balanced Accuracy : 0.6385
##
##        'Positive' Class : 0
##
```

```r
#Storing the result of the confusion matrix
cm_knn <- confusionMatrix(predictedkNN , test_bankData_dist$y)
```

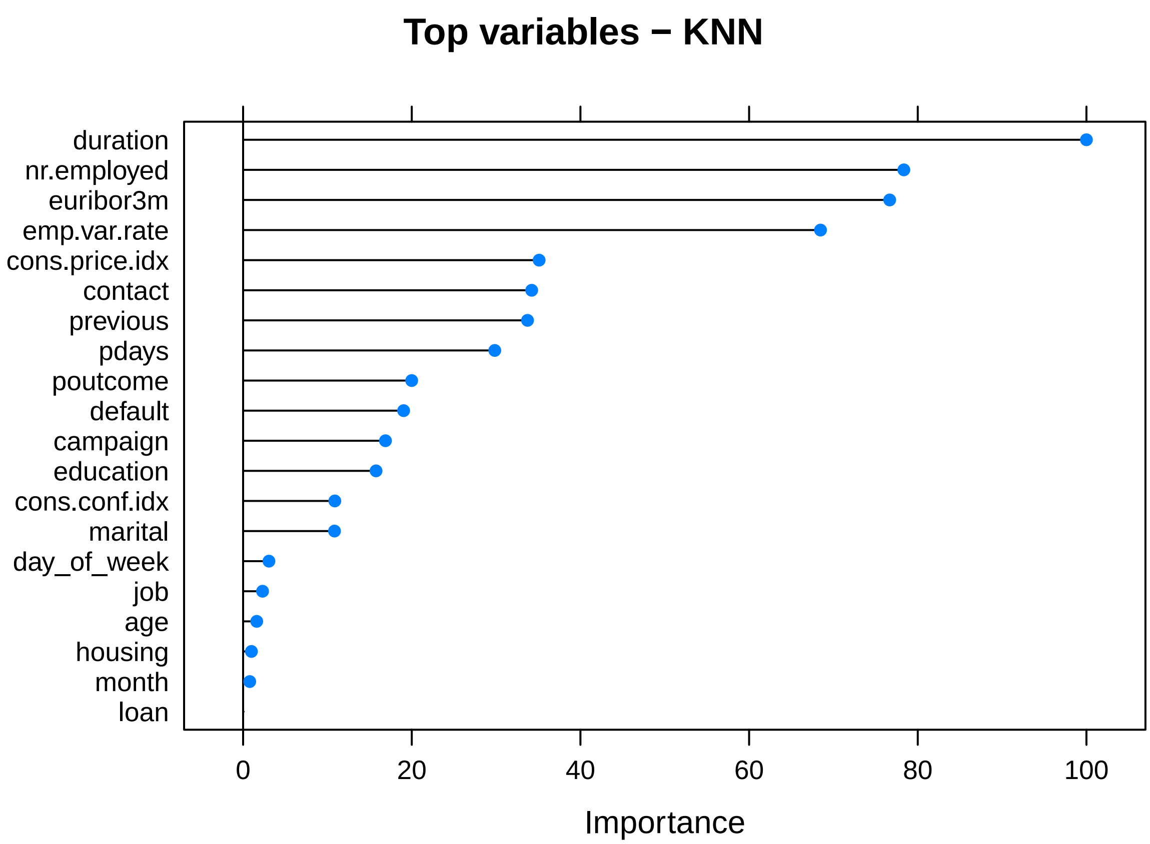
lets check the important variables for this model using the varImp function from caret package.

```r
# Checking the important variables
varImp(bank_knn)
```

```
## ROC curve variable importance
##
##                Importance
## duration         100.0000
## nr.employed       78.3518
## euribor3m         76.6732
## emp.var.rate      68.4695
```

```
## cons.price.idx    35.1023
## contact           34.2202
## previous          33.7248
## pdays             29.8419
## poutcome          19.9911
## default           19.0332
## campaign          16.8782
## education         15.7624
## cons.conf.idx     10.8747
## marital           10.8365
## day_of_week        3.0636
## job                2.3045
## age                1.6150
## housing            0.9870
## month              0.7837
## loan               0.0000
```

```r
#Plotting the imortant variables
plot(varImp(bank_knn),main="Top variables - KNN")
```



Finally we will do a cross table validation of the model.

```r
### Cross table validation for KNN

CrossTable(test_bankData_dist$y, predictedkNN,
           prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
           dnn = c('actual default', 'predicted default'))
```

```
##    Cell Contents
```

```
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
## =======================================
##                   predicted default
## actual default        0       1   Total
## ---------------------------------------
## 0                  3564      90    3654
##                   0.865   0.022
## ---------------------------------------
## 1                   324     140     464
##                   0.079   0.034
## ---------------------------------------
## Total              3888     230    4118
## =======================================
```

Now we will fit the Random Forest model.

```r
############################################################
###   Model 3: Random Forest [RF]
############################################################
## Note that this model takes a long time to fit.
## please be patient
set.seed(8)
bank_rf <- train(y ~ ., method = 'rf', tunegrid = data.frame(mtry = seq(1,7,1)),
                    data = train_bankData_dist, ntree = 100)
```

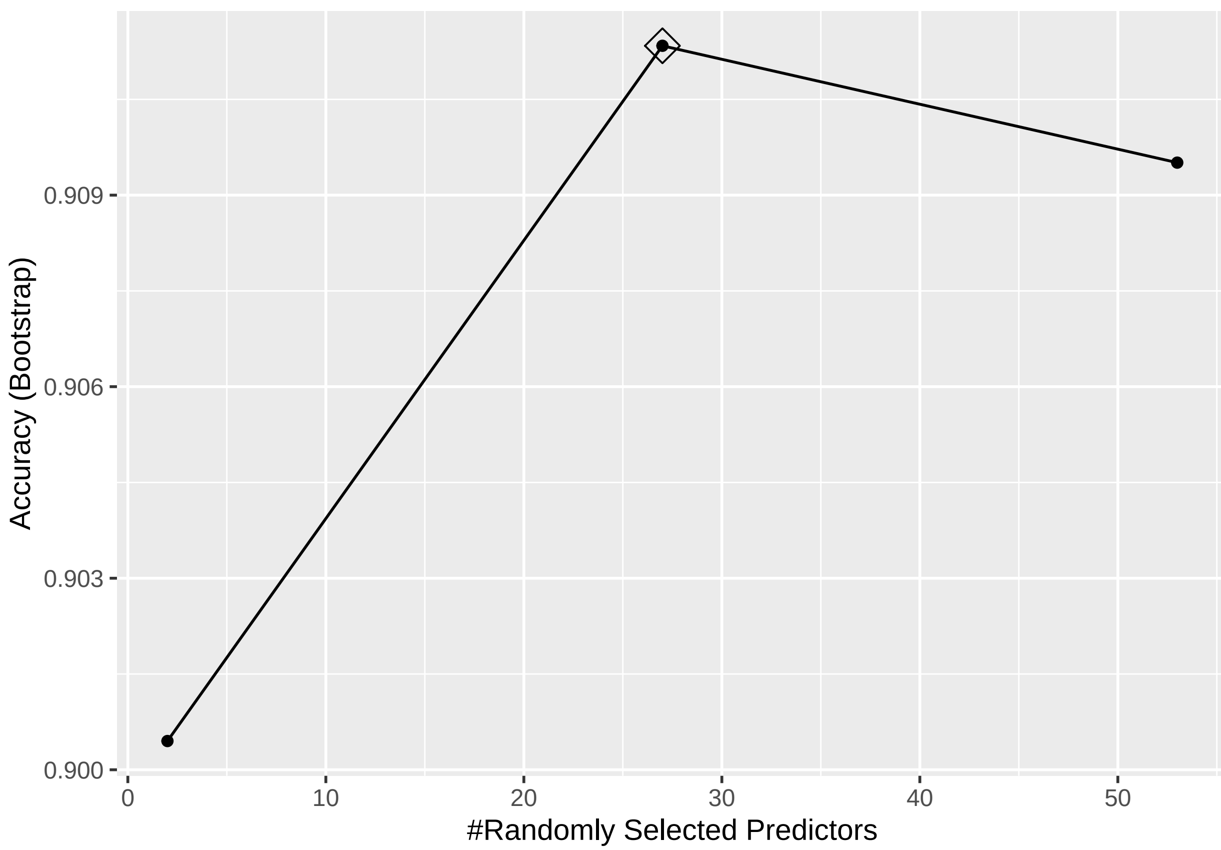
Let's check the best value for the tuning parameter.

```r
# Let's check the best value for tuning parameter

bank_rf$bestTune
```

```
##   mtry
## 2   27
```

Now we will plot the results using ggplot function. The argument highlight highlights the max used in cross validation.

```r
# Plotting the tuning parameter

ggplot(bank_rf, highlight = TRUE)
```



We can also check the details of the final model used.

```
# Final model details

bank_rf$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, ntree = 100, mtry = param$mtry, tunegrid = ..1)
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 8.77%
## Confusion matrix:
##       0    1 class.error
## 0 31539 1344  0.04087218
## 1  1906 2269  0.45652695
```

Now we will use the model to do prediction on the test set.

```
# Prediction on the test set

y_hat_rf <- predict(bank_rf, test_bankData_dist)
```

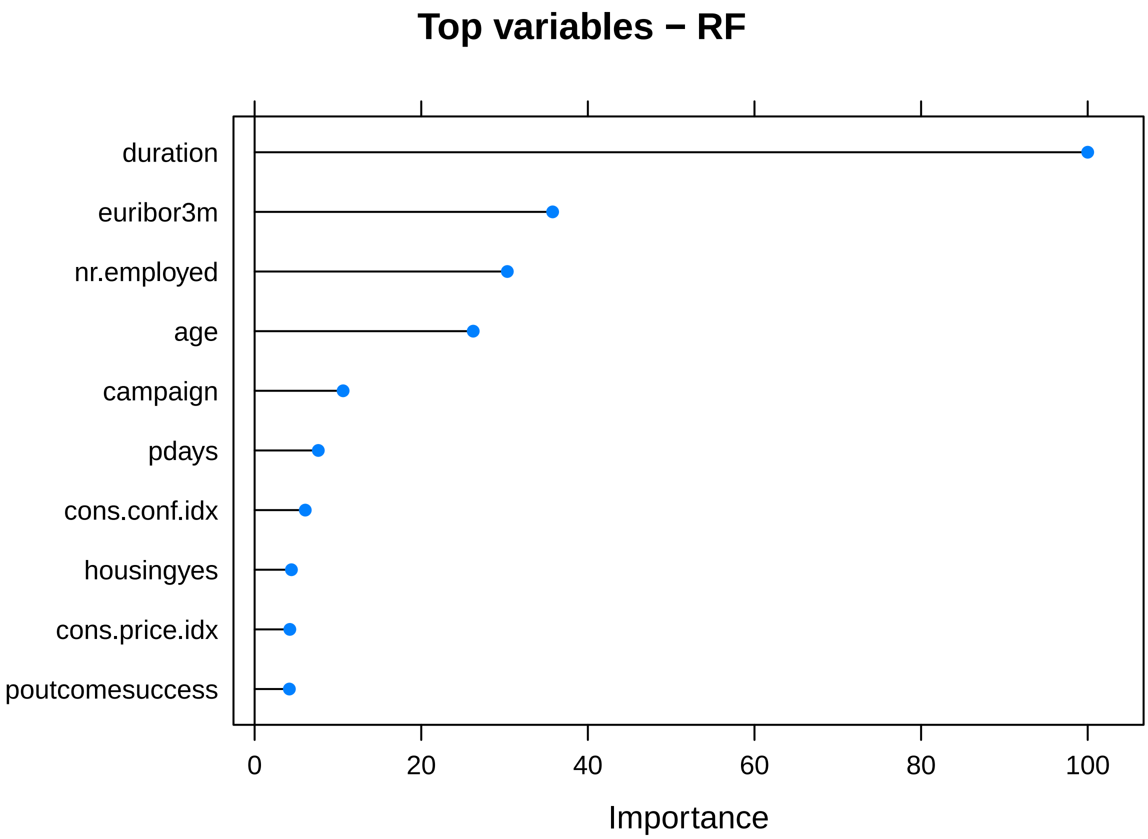
lets check the important variables for this model using the varImp function from caret package.

```
# Checking the important variables

varImp(bank_rf)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 53)
##
##                             Overall
## duration                    100.000
## euribor3m                    35.771
## nr.employed                  30.333
## age                          26.236
## campaign                     10.621
## pdays                         7.643
## cons.conf.idx                 6.079
## housingyes                    4.415
## cons.price.idx                4.224
## poutcomesuccess               4.168
## educationuniversity.degree    3.510
## loanyes                       3.247
## emp.var.rate                  3.214
## day_of_weekmon                3.157
## maritalmarried                3.122
## day_of_weekthu                3.084
## previous                      3.040
## day_of_weekwed                3.022
## jobtechnician                 3.001
## educationhigh.school          2.980
```

Now we will plot the results using ggplot function. The argument highlight highlights the max used in cross validation.

```r
#Plotting the imortant variables

plot(varImp(bank_rf),main="Top variables - RF",top = 10)
```



Generate the confusion matrix and store it for future comparison of models.

```r
# Confusion matrix

confusionMatrix(y_hat_rf , test_bankData_dist$y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3499  209
##          1  155  255
##
##                Accuracy : 0.9116
##                  95% CI : (0.9025, 0.9201)
##     No Information Rate : 0.8873
##     P-Value [Acc > NIR] : 2.028e-07
##
##                   Kappa : 0.5343
##
##  Mcnemar's Test P-Value : 0.00547
##
##             Sensitivity : 0.9576
```

```
##              Specificity : 0.5496
##           Pos Pred Value : 0.9436
##           Neg Pred Value : 0.6220
##               Prevalence : 0.8873
##           Detection Rate : 0.8497
##     Detection Prevalence : 0.9004
##        Balanced Accuracy : 0.7536
##
##         'Positive' Class : 0
##
```

```
#Storing the result of the confusion matrix

cm_rf <- confusionMatrix(y_hat_rf , test_bankData_dist$y)
```

3. Results section

Now let us compare the results of the models that we have used so far.

```
models_list <- list(CART= bank_cart,
                    Random_Forest=bank_rf,
                    KNN=bank_knn)
```

We can get the details of the 3 models run by the following command.

```
#Details of the 3 models run.
models_list
```

```
## $CART
## n= 37058
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 37058 4175 0 (0.88733877 0.11266123)
##    2) nr.employed>=5087.65 32601 2177 0 (0.93322291 0.06677709)
##      4) duration< 606.5 29919  932 0 (0.96884923 0.03115077) *
##      5) duration>=606.5 2682 1245 0 (0.53579418 0.46420582)
##       10) duration< 834.5 1433  528 0 (0.63154222 0.36845778)
##         20) euribor3m>=1.4025 1166  369 0 (0.68353345 0.31646655) *
##         21) euribor3m< 1.4025 267  108 1 (0.40449438 0.59550562) *
##       11) duration>=834.5 1249  532 1 (0.42594075 0.57405925) *
##    3) nr.employed< 5087.65 4457 1998 0 (0.55171640 0.44828360)
##      6) duration< 162.5 1568  247 0 (0.84247449 0.15752551) *
##      7) duration>=162.5 2889 1138 1 (0.39390793 0.60609207)
##       14) pdays>=513 2021  974 1 (0.48193963 0.51806037)
##         28) duration< 283.5 927  382 0 (0.58791802 0.41208198) *
##         29) duration>=283.5 1094  429 1 (0.39213894 0.60786106) *
##       15) pdays< 513 868  164 1 (0.18894009 0.81105991) *
##
## $Random_Forest
## Random Forest
##
## 37058 samples
##    20 predictor
##     2 classes: '0', '1'
```

```
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 37058, 37058, 37058, 37058, 37058, 37058, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9004500  0.2558356
##   27    0.9113393  0.5319336
##   53    0.9095093  0.5252649
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
##
## $KNN
## k-Nearest Neighbors
##
## 37058 samples
##    20 predictor
##     2 classes: '0', '1'
##
## Pre-processing: centered (53), scaled (53)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 33352, 33351, 33352, 33352, 33353, 33353, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.8952189  0.3581381
##   7  0.8968917  0.3510206
##   9  0.8987805  0.3506856
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

Let us compare the outputs like Sensitivity , Specificity etc from the confusion matrix of the 3 models.

```r
cm_list <- list(
  CART=cm_cart,
  Random_Forest=cm_rf,
  KNN=cm_knn)

#Output of Confusion Matix
cm_list_results <- sapply(cm_list, function(x) x$byClass)
cm_list_results %>% knitr::kable()
```

	CART	Random_Forest	KNN
Sensitivity	0.9597701	0.9575807	0.9753695
Specificity	0.5215517	0.5495690	0.3017241
Pos Pred Value	0.9404666	0.9436354	0.9166667
Neg Pred Value	0.6221080	0.6219512	0.6086957
Precision	0.9404666	0.9436354	0.9166667
Recall	0.9597701	0.9575807	0.9753695
F1	0.9500203	0.9505569	0.9451074
Prevalence	0.8873239	0.8873239	0.8873239
Detection Rate	0.8516270	0.8496843	0.8654687

	CART	Random_Forest	KNN
Detection Prevalence	0.9055367	0.9004371	0.9441476
Balanced Accuracy	0.7406609	0.7535748	0.6385468

Now we will create a result set for the accuracy of the three models.

```
#Accuracy of CART
cm_cart$overall['Accuracy']
```

```
##  Accuracy
## 0.9103934
```

```
#Accuracy of KNN
cm_knn$overall['Accuracy']
```

```
##  Accuracy
## 0.8994658
```

```
#Accuracy of Random Forest
cm_rf$overall['Accuracy']
```

```
##  Accuracy
## 0.9116076
```

```
########Creating a results table to store results of different models ####
accuracy_results <- data_frame(MODEL ="CART", ACCURACY = cm_cart$overall['Accuracy'])
accuracy_results <- bind_rows(accuracy_results,data_frame(MODEL ="Random Forest", ACCURACY = cm_rf$overa
accuracy_results <- bind_rows(accuracy_results,data_frame(MODEL ="KNN", ACCURACY = cm_knn$overall['Accu
accuracy_results %>% knitr::kable()
```

MODEL	ACCURACY
CART	0.9103934
Random Forest	0.9116076
KNN	0.8994658

From the results we can see that Random Forest has the maximum accuracy, followed by CART and KNN respectively.

4. Conclusion section

In this project we applied the learning of the edx data science course to a real world problem dataset. The machine learning models of classification and regression tree, k-nearest neighbor and random forest were fitted into the data and predictions were done on the test data set.
After running the three models, the confusion matrix was captured and the statistics compared. It was found that random forest most the most useful machine learning algorithm of the three.