

PREDICTING_CUSTOMER_PENETRATION PLAGRISM FREE.docx

 Delhi Technological University

Document Details

Submission ID

trn:oid:::27535:96687760

Submission Date

May 19, 2025, 5:35 PM GMT+5:30

Download Date

May 19, 2025, 5:37 PM GMT+5:30

File Name

PREDICTING_CUSTOMER_PENETRATION PLAGRISM FREE.docx

File Size

832.7 KB

59 Pages

8,113 Words

51,875 Characters





26% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.




Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 8 words)

Match Groups

-  **112** Not Cited or Quoted 26%
Matches with neither in-text citation nor quotation marks
-  **0** Missing Quotations 0%
Matches that are still very similar to source material
-  **0** Missing Citation 0%
Matches that have quotation marks, but no in-text citation
-  **0** Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

- 22%  Internet sources
- 11%  Publications
- 25%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- 112** Not Cited or Quoted 26%
Matches with neither in-text citation nor quotation marks
- 0** Missing Quotations 0%
Matches that are still very similar to source material
- 0** Missing Citation 0%
Matches that have quotation marks, but no in-text citation
- 0** Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

- 22% Internet sources
- 11% Publications
- 25% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	rstudio-pubs-static.s3.amazonaws.com	7%
2	Submitted works	Queen's University of Belfast on 2023-01-06	2%
3	Submitted works	Mercer University on 2022-12-05	1%
4	Submitted works	University of Illinois at Urbana-Champaign on 2022-02-05	1%
5	Internet	rdr.io	<1%
6	Internet	locusbucket.s3.amazonaws.com	<1%
7	Submitted works	University of Lancaster on 2022-05-03	<1%
8	Submitted works	University of Leeds on 2023-12-11	<1%
9	Submitted works	University of Technology, Sydney on 2021-12-11	<1%
10	Submitted works	Mercer University on 2019-10-13	<1%

11	Submitted works	Bayerische Julius-Maximilians-Universität Würzburg on 2018-12-18	<1%
12	Submitted works	Mercer University on 2022-12-16	<1%
13	Submitted works	University of Lancaster on 2022-05-03	<1%
14	Submitted works	Asia Pacific University College of Technology and Innovation (UCTI) on 2023-01-15	<1%
15	Submitted works	Texas A&M University, College Station on 2015-04-14	<1%
16	Internet	erifuna1999.github.io	<1%
17	Internet	www.kaggle.com	<1%
18	Submitted works	Laureate Higher Education Group on 2018-04-11	<1%
19	Internet	www.coursehero.com	<1%
20	Submitted works	Instituto Madrileno de Formacion on 2020-06-29	<1%
21	Submitted works	University of Lancaster on 2022-05-17	<1%
22	Submitted works	National Research University Higher School of Economics on 2019-05-22	<1%
23	Submitted works	essex on 2020-09-11	<1%
24	Submitted works	Erasmus University of Rotterdam on 2024-12-06	<1%

25	Submitted works	University of Leeds on 2023-12-10	<1%
26	Internet	github.com	<1%
27	Internet	igdtuw.ac.in	<1%
28	Submitted works	Laureate Higher Education Group on 2011-11-29	<1%
29	Submitted works	Monash University on 2019-09-15	<1%
30	Submitted works	University of Sydney on 2018-10-09	<1%
31	Internet	notebook.community	<1%
32	Submitted works	Loughborough University on 2020-10-26	<1%
33	Submitted works	S.P. Jain Institute of Management and Research, Mumbai on 2024-12-17	<1%
34	Submitted works	University of Essex on 2022-12-16	<1%
35	Submitted works	University of Sydney on 2021-11-07	<1%
36	Submitted works	Colorado State University, Global Campus on 2024-04-27	<1%
37	Submitted works	Texas A&M University, College Station on 2015-11-20	<1%
38	Submitted works	The Robert Gordon University on 2024-12-31	<1%

39	Submitted works	Queen's University of Belfast on 2023-01-06	<1%
40	Submitted works	RMIT University on 2020-06-01	<1%
41	Submitted works	University of Maryland, Global Campus on 2024-07-03	<1%
42	Submitted works	University of Technology, Sydney on 2022-05-14	<1%
43	Internet	deepnote.com	<1%
44	Submitted works	BPP College of Professional Studies Limited on 2025-04-09	<1%
45	Internet	github-wiki-see.page	<1%
46	Submitted works	Australian National University on 2024-06-11	<1%
47	Submitted works	RMIT University on 2022-10-06	<1%
48	Submitted works	University of Wollongong on 2024-03-15	<1%
49	Submitted works	University of Wollongong on 2025-04-19	<1%
50	Internet	www.scirp.org	<1%
51	Submitted works	Leeds Beckett University on 2025-05-05	<1%
52	Submitted works	Macquarie University on 2016-06-06	<1%

53	Submitted works	Swiss School of Business and Management - SSBM on 2024-11-07	<1%
54	Submitted works	University of Northumbria at Newcastle on 2021-01-12	<1%
55	Internet	www.analyticsinsight.net	<1%
56	Submitted works	City University of New York System on 2025-03-06	<1%
57	Publication	Gerson Fabio da Silva. "Estudo de caso de redes MLP e Kolmogorov-Arnold aplica...	<1%
58	Submitted works	Institute of Technology Blanchardstown on 2017-05-01	<1%
59	Submitted works	Liverpool John Moores University on 2025-02-26	<1%
60	Publication	Miada A. Almasre, Hana Al-Nuaim. "A comparison of Arabic sign language dynami...	<1%
61	Submitted works	Monash University on 2018-05-24	<1%
62	Publication	Richard J. Roiger. "Just Enough R! - An Interactive Approach to Machine Learning ...	<1%
63	Submitted works	The Robert Gordon University on 2024-12-31	<1%
64	Submitted works	UNICAF on 2025-04-27	<1%
65	Submitted works	University of Birmingham on 2025-03-28	<1%
66	Submitted works	University of Ulster on 2025-04-17	<1%

67	Submitted works	University of Wisconsin Extension on 2023-12-07	<1%
68	Submitted works	University of Wollongong on 2020-06-22	<1%
69	Submitted works	University of Wollongong on 2025-04-23	<1%
70	Submitted works	The Robert Gordon University on 2025-01-01	<1%
71	Submitted works	UT, Dallas on 2013-12-17	<1%
72	Submitted works	University of Leicester LTI on 2025-05-09	<1%
73	Submitted works	University of North Texas on 2021-12-04	<1%
74	Submitted works	University of Reading on 2024-11-26	<1%
75	Submitted works	University of Technology, Sydney on 2025-03-15	<1%

EXECUTIVE SUMMARY

The objective of this project is to build a predictive model that identifies customers most likely to subscribe to a term deposit product, using historical campaign data from a Portuguese bank. Given the class imbalance, SMOTE was employed to balance the dataset for data balancing.

Multiple machine learning algorithms including Logistic Regression, Decision Tree, and Random Forest, were evaluated on both balanced and unbalanced datasets (SMOTE-enhanced datasets)

Key Findings:

- Customers aged 35–60 and those with call durations over 3 minutes are more likely to subscribe.
- Success in past campaigns significantly predicts future conversions.
- SMOTE-enhanced models outperformed their non-SMOTE counterparts in detecting positive responses.

Best Model Selected: Logistic Regression (on SMOTE data)

Accuracy: 86.2% | **Sensitivity:** 95.1% | **Kappa:** 0.726

Actionable Insight:

The model can help the bank reduce irrelevant outreach by focusing only on high-likelihood customers, thus improving customer experience and campaign efficiency.

TABLE OF CONTENTS

1. Introduction
2. Background and Problem Statement
3. Study Objectives
4. Product Scope
5. Research Methodology
 - a. Data-Set & Attribute Information
 - b. Import Libraries
 - c. Import and Introduction to DataSet
 - d. Categorical Treatment
 - e. Feature Engineering
 - f. SMOTE Algorithm For Unbalanced Classification Problems
 - a. d. Data Splitting | Building Training and Testing Data sets
 - b. e. Applying Prediction Models
 - g. Conclusion | Evaluating Different Models
6. Findings & Outcomes
7. Conclusion & Recommendations
8. References

INTRODUCTION

In the current, competitive financial landscape, the bank success largely depends on its capability to deliver the right customers at the optimal time. The growth of customer data combined with advances in machine learning present a valuable opportunity for banks to enhance their marketing strategies through intelligent and data-driven decision-making. This project report outlines a structured approach to addressing a common challenge in banking sector identifying the most likely customers to subscribe to a newly launched term deposit product.

The bank in question is currently facing a serious gap in its marketing process. With no framework to differentiate between interested and uninterested customers, the bank resorts to contacting all customers indiscriminately. This untargeted approach has led to an increase in customer complaints regarding irrelevant and intrusive marketing calls, ultimately causing dissatisfaction and a negative perception of the bank's outreach strategies. Moreover, such a method is not only inefficient in terms of resources but also fails to maximize the conversion potential of the bank's marketing efforts.

To address this problem, the bank aims to leverage historical marketing campaign data that includes information on customer demographics, economic context, communication history, and past responses. The goal is to design a predictive system that can effectively segment the customer base and prioritize outreach to individuals most likely to subscribe to the term deposit product. By doing so, the bank can streamline its marketing activities, reduce unnecessary customer interactions, and achieve a higher return on investment.

This project utilizes supervised machine learning algorithms to develop a classification model that predicts whether a customer is likely to buy the term deposit product. The dataset used for this project comprises over 41,000 records, with attributes ranging from personal information such as age, job type, and education level to campaign-specific details such as contact duration, call month, and response outcome. Preliminary data exploration revealed class imbalance in the dataset, with significantly more 'no' responses than 'yes' responses, which necessitated the use of Synthetic Minority Oversampling Technique (SMOTE) to ensure balanced model training.

44 The methodology adopted in this project follows a structured pipeline. First, extensive exploratory data analysis (EDA) was conducted to understand feature distributions, identify missing values, and uncover relationships between variables and customer responses. Categorical variables were treated, and features were engineered based on domain understanding—for instance, grouping age and call duration into meaningful categories based on their predictive significance. Highly correlated economic indicators were also analyzed to eliminate redundancy and improve model efficiency.

66 Following data preprocessing, multiple classification models were applied, including Logistic Regression, Decision Trees, and Random Forests. The models were trained and validated on both SMOTE-balanced and original datasets to compare their effectiveness in handling class imbalance. Each model's performance was assessed using standard metrics such as accuracy, sensitivity, specificity, ROC curves, and confusion matrices. In addition, feature importance analysis was carried out to identify key drivers influencing customer decisions.

51

PROBLEM STATEMENT

As part of its strategic initiative to deepen relationships with existing customers, a bank is launching a new term deposit product. The bank plans to reach out to its customer base to promote and upsell this offering. However, in executing past marketing campaigns, the bank has encountered a critical challenge—customers have raised complaints about receiving irrelevant and excessive marketing calls. These calls, often indiscriminately made to all customers without regard to individual interest or suitability, have led to growing dissatisfaction and reputational risk.

The underlying issue is the absence of a data-driven framework that can distinguish between likely and unlikely buyers of the product. Without such a mechanism in place, the bank's current approach relies heavily on blanket outreach, which is both inefficient and counterproductive. Moreover, the bank has ruled out manual shortlisting of potential customers due to the risk of human bias and the inefficiencies associated with such subjective interventions.

However, a valuable asset already exists: historical data from previous campaigns, including customer demographics, campaign details, and whether the offer was accepted or declined. This dataset presents a promising opportunity to develop a predictive model using supervised machine learning algorithms. The objective is to analyze past patterns to identify the characteristics of customers who are more inclined to subscribe to the term deposit.

By adopting this predictive approach, the bank aims to transition to a more targeted and automated marketing strategy. Such a model would not only reduce unnecessary communication with uninterested clients but also improve conversion rates and streamline marketing efforts.

OBJECTIVES

1. To analyze historical campaign data and understand key factors influencing customer decisions.
2. To apply supervised machine learning techniques for binary classification (subscription: *yes* or *no*).
3. To address class imbalance in the dataset using methods like SMOTE (Synthetic Minority Over-sampling Technique).
4. To evaluate and compare multiple classification models (e.g., Logistic Regression, Decision Trees, Random Forest) based on accuracy, sensitivity, and other relevant performance metrics.
5. To recommend the most effective model for deployment in the bank's customer outreach system.
6. To design a scalable and automated framework for targeted marketing with minimal manual intervention.

PRODUCT SCOPE

1. Understanding and Data Pre-Processing

- a. The data set consists of customer characteristics, campaign characteristics, previous campaign information as well as whether customer ended up subscribing to the product as a result of that campaign or not.
- b. The data will be cleansed for any irregularities and some of the categorical attributes of data sets will be masked to continuous values in order to prepare the data feed for building model.

2. Exploratory Data Analysis

- a. Here we will perform initial investigations on data to discover any patterns, to spot anomalies and to check assumptions with help of summary statistics and graphical representations.

3. Feature Engineering

- a. Based on understanding build out of data, applying certain domain knowledge and identifying correlations among different attributes few important attributes from data sets will be identified that better represents the underlying problem to the predictive models in form of inputs that the algorithm can understand.

4. Data Splitting and Applying Classification Models

- a. As per standard best practices, 70% of random records from dataset will be used as Training Data on which the model will be built and 30% of data will be used to test the model performance. Classification techniques based out of Logistic Regression and Decision Tree will be used to build the predictive model.

5. Model Evaluation

- a. Certain metrics such as Confusion Metrics, Precision, Recall, Accuracy etc., will be used to evaluate the model performance on different algorithms and the predictions from the model with better evaluation metrics can be considered for targeted product promotion.

RESEARCH METHODOLOGY

Data-Set Information

The dataset used in this project originates from a Portuguese banking institution's direct marketing campaigns, focused on promoting term deposit subscriptions. The data captures detailed information about customer profiles, past marketing interactions, and corresponding outcomes, making it suitable for supervised classification modeling.

The dataset contains 41,188 records and 21 input features, along with 1 output variable (y), which indicates whether a customer subscribed to the term deposit (yes or no). The features span across three major categories:

- **Client Information:** Attributes such as age, job type, marital status, education level, and existing financial commitments (e.g., housing or personal loans).
- **Campaign Details:** Information from previous marketing campaigns, including contact method, timing (month, day), call duration, number of contacts, and previous outcomes.
- **Economic Indicators:** Macro-level attributes like employment variation rate, consumer confidence index, and Euribor 3-month rate, which may influence customer decision-making.

The dataset does not contain null values in the traditional sense, but several features include the category "unknown," which has been treated as a placeholder for missing or non-disclosed information. These aspects were addressed during data preprocessing and feature engineering stages of the project.

Attribute Information

- 1) Age
- 2) Job : type of job
- 3) Marital : marital status
- 4) Education: Type of education
- 5) Default: has credit in default?
- 6) Housing: has housing loan?
- 7) Loan: has personal loan?
- 8) Contact: contact communication type
- 9) Month: last contact month of year
- 10)Day_of_week: last contact day of the week
- 11)Duration: last contact duration, in seconds
- 12)Campaign: number of contacts performed during this campaign and for this client
- 13)Pdays: number of days that passed by after the client was last contacted from a previous campaign
- 14)Previous: number of contacts performed before this campaign and for this client
- 15)Poutcome: outcome of the previous marketing campaign
- 16)Social and economic context attributes
- 17)Emp.var.rate: employment variation rate - quarterly indicator
- 18)Cons.price.idx: consumer price index - monthly indicator
- 19)Cons.conf.idx: consumer confidence index - monthly indicator
- 20)Euribor3m: euribor 3 month rate - interest rate at which a panel of banks lend money to one another with a maturity of 3 months
- 21)Nr.employed: number of employees - quarterly indicator
- 22)Output variable (desired target):y - has the client subscribed a term deposit?

Import Libraries

In [1]:

```
library(dplyr)
library(tidyr)
library(ggplot2)
library(ggmosaic)
library(gmodels)
library(corrplot)
library(DMwR2)
library(ROCR)
library(caret)
library(rpart)
library(smotefamily)
library(rpart.plot)
library(randomForest)
```

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

corrplot 0.92 loaded

Registered S3 method overwritten by 'quantmod':

method from

as.zoo.data.frame zoo

Loading required package: lattice

randomForest 4.7-1.1

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

Attaching package: 'randomForest'

The following object is masked from 'package:ggplot2':

margin

The following object is masked from 'package:dplyr':

combine

1 Import and Introduction to DataSet

1.1 Input the Data File

In [2]:

```
InputData = read.csv(file = " bank-additional-full.csv" , sep = ";"
,stringsAsFactors = F)

dim(InputData)
```

41188 · 21

The dataset has **41,188 rows** and **21 Columns**

1.2 Know the DataSet

In [3]:

```
names(InputData)
```

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

The First 20 columns seems to be **"potential explanatory variables"** or independent variables and the column named **"y"** is the dependent variable

1.3 Looking at the sample Data

In [4]:

```
head(InputData)
```

A data.frame: 6 × 21

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	campaign	pdays	previous	poutcome	emp.var.rate	cons.p
	<int>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	...	<int>	<int>	<int>	<chr>	<dbl>	
1	56	housemaid	married	basic.4y	no	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
2	57	services	married	high.school	unknown	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
3	37	services	married	high.school	no	yes	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
4	40	admin.	married	basic.6y	no	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
5	56	services	married	high.school	no	no	yes	telephone	may	mon	...	1	999	0	nonexistent	1.1	
6	45	services	married	basic.9y	unknown	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	

1.4 Identifying the datatypes of all the columns

In [5]:

```
sapply(InputData, typeof)
```

9

age: 'integer' job: 'character' marital: 'character' education: 'character' default: 'character' housing: 'character' loan: 'character' contact: 'character' month: 'character' day_of_week: 'character' duration: 'integer' campaign: 'integer' pdays: 'integer' previous: 'integer' poutcome: 'character' emp.var.rate: 'double' cons.price.idx: 'double' cons.conf.idx: 'double' euribor3m: 'double' nr.employed: 'double' y: 'character'

1.5 Check if any of the columns have null values

In [6]:

```
sapply(InputData,is.null)
```

16

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

None of the columns in our dataset have any missing or null values, however according to the documentation we know there is variable defined as "unknown" which is equivalent to null

1.6 Identifying the unknowns in the DataSet

In [7]:

```
InputData %>%
  summarise_all(list(~sum(. == "unknown"))) %>%
  gather(key = "Column Name", value = "No._of_Unknowns") %>%
  arrange(-No. of Unknowns)

### Total Unknowns in all Columns ###
TotalUnknowns <- sum(InputData == "unknown")
TotalUnknowns <- cat("Total Number of Unknowns in all Columns in DataSet are",TotalUnknowns)
```

A data.frame: 21 × 2

Column Name	No._of_Unknowns
<chr>	<int>
default	8597
education	1731
housing	990
loan	990
job	330
marital	80
age	0
contact	0
month	0
day_of_week	0
duration	0
campaign	0
pdays	0
previous	0
poutcome	0
emp.var.rate	0
cons.price.idx	0
cons.conf.idx	0
euribor3m	0
nr.employed	0
y	0

Total Number of Unknowns in all Columns in DataSet are 12718

6 of the features in the DataSet seems to have atleast one of there values as "unknown"

2 Exploratory Data Analysis

2.1 Seggregating Data as per functional understanding of dataset

In [8]:

```
ClientPII <- select(InputData,1,2,3,4,5,6,7,21)
head(ClientPII)
```

A data.frame: 6 × 8

	age	job	marital	education	default	housing	loan	y
	<int>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>
1	56	housemaid	married	basic.4y	no	no	no	no
2	57	services	married	high.school	unknown	no	no	no
3	37	services	married	high.school	no	yes	no	no
4	40	admin.	married	basic.6y	no	no	no	no
5	56	services	married	high.school	no	no	yes	no
6	45	services	married	basic.9y	unknown	no	no	no

The above 7 columns serve as **Non Sensitive Personal Identifiable Information (PII)** of the client thus seggreagted together in ClientPII DataFrame

In [9]:

```
PreviousCampaign <- select(InputData,8,9,10,11,12,13,14,15,21)
head(PreviousCampaign)
```

A data.frame: 6 × 9

	contact	month	day_of_week	duration	campaign	pdays	previous	poutcome	y
	<chr>	<chr>	<chr>	<int>	<int>	<int>	<int>	<chr>	<chr>
1	telephone	may	mon	261	1	999	0	nonexistent	no
2	telephone	may	mon	149	1	999	0	nonexistent	no
3	telephone	may	mon	226	1	999	0	nonexistent	no
4	telephone	may	mon	151	1	999	0	nonexistent	no
5	telephone	may	mon	307	1	999	0	nonexistent	no
6	telephone	may	mon	198	1	999	0	nonexistent	no

The above columns can help in understanding attributes related to last contact with clients as part of previous campaign

In [10]:

```
EconomicContext <- select(InputData,16:20)
head(EconomicContext)
```

A data.frame: 6 x 5

	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	1.1	93.994	-36.4	4.857	5191
2	1.1	93.994	-36.4	4.857	5191
3	1.1	93.994	-36.4	4.857	5191
4	1.1	93.994	-36.4	4.857	5191
5	1.1	93.994	-36.4	4.857	5191
6	1.1	93.994	-36.4	4.857	5191

These columns like employee variation rate (quarterly indicator) consumer price index (monthly indicator) and others constitutes in building Economic Context

2.2 Performing EDA on Client's PII

2.2.1 Understanding Categorical Values

2.2.1.1 Marital Status

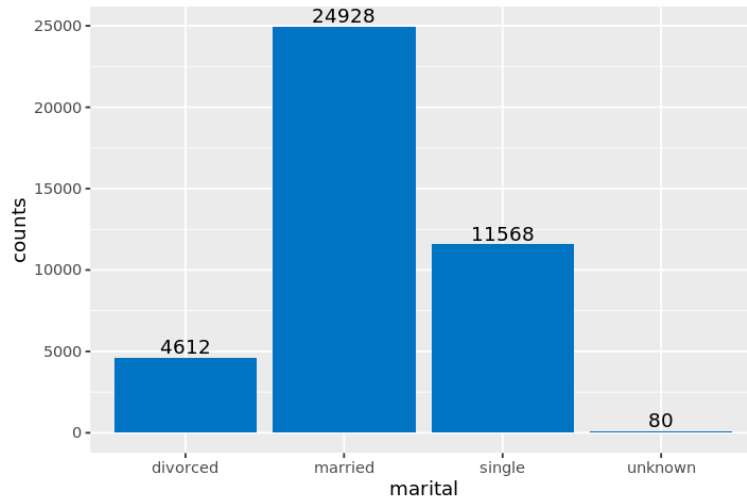
In [11]:

```
table(ClientPII$marital)
divorced married single unknown
4612 24928 11568 80
```

The above table shows unique values in marital attribute of the dataset

In [12]:

```
MaritalDF <- ClientPII %>% group_by(marital) %>% summarise(counts = n())
options(repr.plot.width = 6, repr.plot.height = 4)
ggplot(MaritalDF, aes(x = marital, y = counts)) + geom_bar(fill = "#0073C2FF", stat =
"identity") + geom_text(aes(label = counts), vjust = -0.3)
```

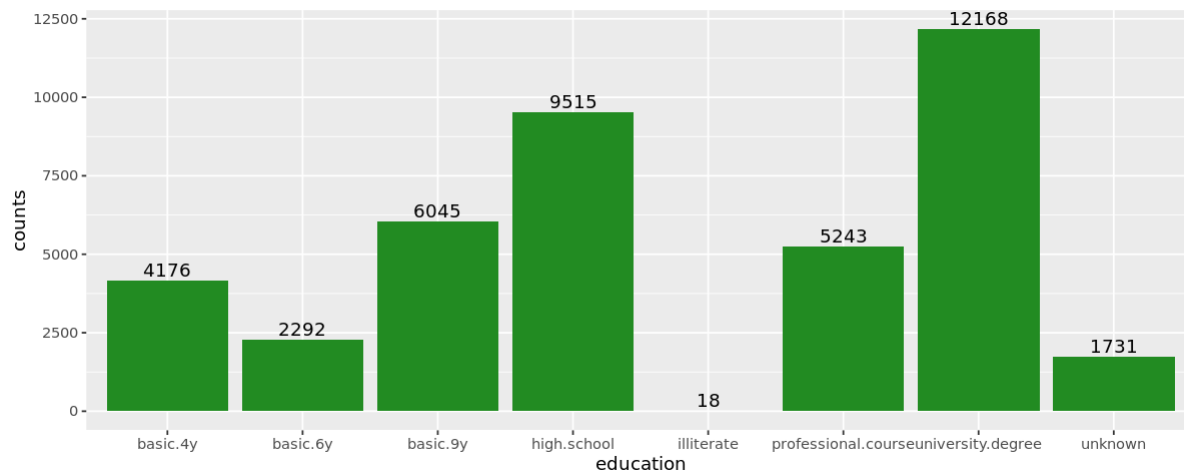


The above figure is graphical representation of stats for marital attributes which depicts most of the bank clients are **married**.

2.2.1.2 Education

In [13]:

```
EducationDF <- ClientPII %>% group_by(education) %>%
  summarise(counts = n())
options(repr.plot.width = 10, repr.plot.height = 4)
ggplot(EducationDF, aes(x = education, y = counts)) + geom_bar(fill = "#228B22", stat
= "identity") +
  geom_text(aes(label = counts), vjust = -0.3)
```



Above is graphical representation of education status of bank employees

2.2.1.3 Jobs

In [14]:

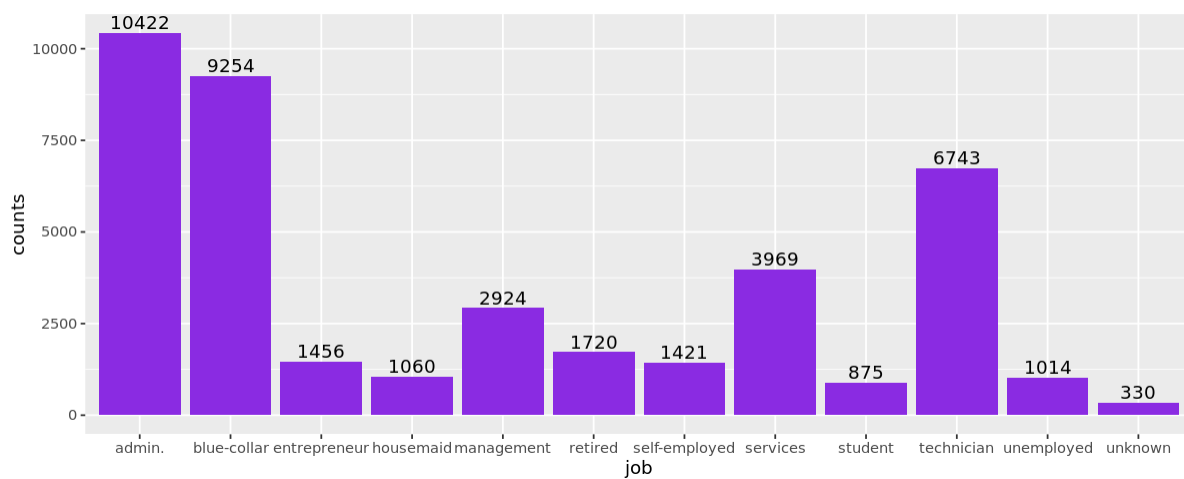
```
table(ClientPII$job)
```

admin.	blue-collar	entrepreneur	housemaid	management
10422	9254	1456	1060	2924
retired	self-employed	services	student	technician
1720	1421	3969	875	6743
unemployed	unknown			
1014	330			

Graphical Representation

In [15]:

```
JobDF <- ClientPII %>% group_by(job) %>% summarise(counts = n())
options(repr.plot.width = 10, repr.plot.height = 4)
ggplot(JobDF, aes(x = job, y = counts)) + geom_bar(fill = "#8A2BE2", stat =
"identity") + geom_text(aes(label = counts), vjust = -0.3)
```



We observe the most of the bank clients are working in **administrative jobs** followed by **blue-collar jobs** whereas as job status of 330 clients are unknown

Crossing and Plotting types of Jobs with #clients buying term deposit

In [16]:

```
CrossTable(ClientPII$job, ClientPII$y, prop.r=TRUE, prop.c=FALSE, prop.t=FALSE,
prop.chisq=FALSE, chisq = FALSE)
## Here prop.r specifies Row Proportion, prop.c Column Proportion, prop.t Table Proportion
etc..
```

Cell Contents

	N
	N / Row Total

Total Observations in Table: 41188

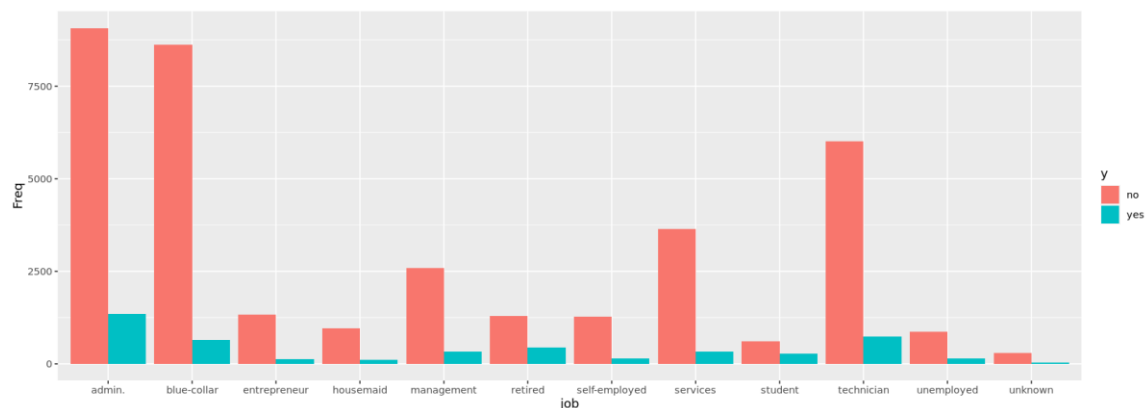
| ClientPII\$y

ClientPII\$job	no	yes	Row Total
admin.	9070	1352	10422
	0.870	0.130	0.253
blue-collar	8616	638	9254
	0.931	0.069	0.225
entrepreneur	1332	124	1456
	0.915	0.085	0.035
housemaid	954	106	1060
	0.900	0.100	0.026
management	2596	328	2924
	0.888	0.112	0.071
retired	1286	434	1720
	0.748	0.252	0.042
self-employed	1272	149	1421
	0.895	0.105	0.035
services	3646	323	3969
	0.919	0.081	0.096
student	600	275	875
	0.686	0.314	0.021
technician	6013	730	6743
	0.892	0.108	0.164
unemployed	870	144	1014
	0.858	0.142	0.025
unknown	293	37	330
	0.888	0.112	0.008
Column Total	36548	4640	41188

Graphical Representation

In [17]:

```
GraphData <- rename(count(ClientPII, job, y), Freq = n)
options(repr.plot.width = 14, repr.plot.height = 5)
JobGraph <- ggplot(GraphData, aes(job, Freq)) + geom_bar(aes(fill = y), stat =
"identity", position = "dodge")
JobGraph
```



The above table shows **maximum** number of term deposits are being bought by Clients involved in **administrative jobs** followed with **blue collar jobs** which is completely in sync with the above observation where it was discovered that most of the clients of bank are involved in administrative jobs followed by blue collar jobs

2.2.1.4 Others

In [18]:

```
DefaultDF <- ClientPII %>% group_by(default) %>% summarise(counts = n())
HousingDF <- ClientPII %>% group_by(housing) %>% summarise(counts = n())
LoanDF <- ClientPII %>% group_by(loan) %>% summarise(counts = n())
DefaultDF
HousingDF
LoanDF
```

A tibble: 3 × 2

default counts

<chr>	<int>
no	32588
unknown	8597
yes	3

A tibble: 3 × 2

housing counts

<chr>	<int>
no	18622
unknown	990
yes	21576

A tibble: 3 × 2

loan counts

<chr>	<int>
no	33950
unknown	990
yes	6248

Here we have counts for **#clients** which have **default credits or have housing loan or any other type of loan**

2.2.2 Age

Looking at **Maximum & Minimum** Age of bank's client

In [19]:

```
summary(ClientPII$age)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
17.00	32.00	38.00	40.02	47.00	98.00

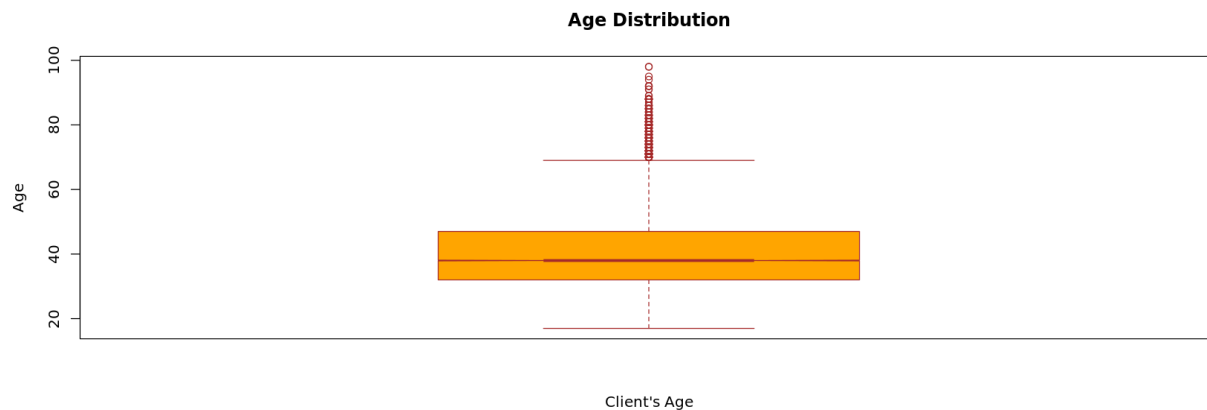
The above summary of age attribute describes **Maximum Age** as 98 and **Minimum Age** as 17 whereas the

Mean Age is 40 Plotting Age distribution of Client's Age to determine the interval where most of the bank

client resides

In [20]:

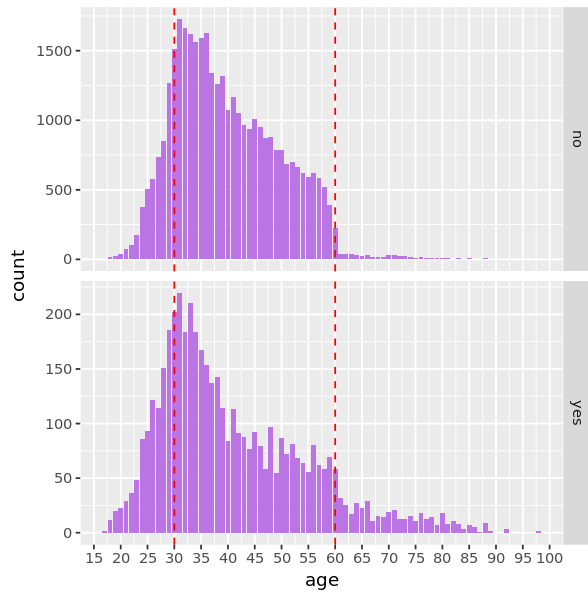
```
boxplot(ClientPII$age,
main = "Age Distribution", xlab = "Client's Age", ylab = "Age", col =
"orange", border = "brown", #horizontal = TRUE,
notch = TRUE)
options(repr.plot.width = 5, repr.plot.height = 5)
```



Determining the age groups of clients who bought and did not bough the term deposit

In [21]:

```
ClientPII %>%
ggplot() +
aes(x = age) +
geom_bar(fill = '#BA74E4') +
geom_vline(xintercept = c(30, 60),
col = "red",
linetype = "dashed") +
facet_grid(y ~ .,
scales = "free y") +
scale_x_continuous(breaks = seq(0, 100, 5))
```



2.3 Performing EDA on Previous Campaign attributes

2.3.1 Contact / How was the client contacted in previous campaign

In [22]:

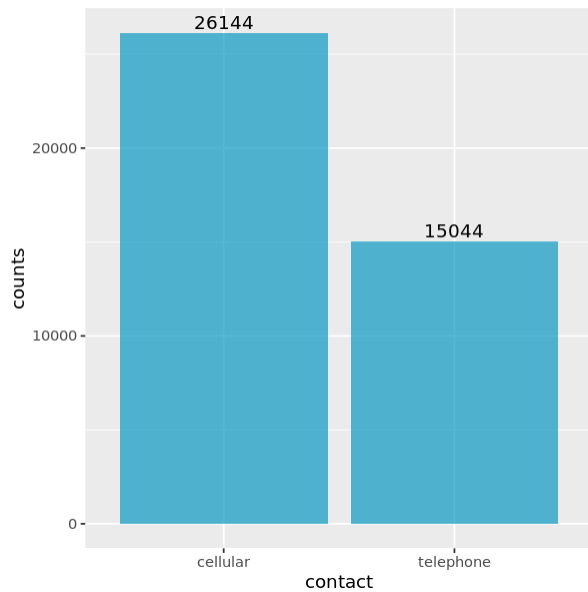
```
DurationDF <- PreviousCampaign %>% group_by(contact) %>%
  summarise(counts = n())
DurationDF
```

A tibble: 2 × 2

contact	counts
<chr>	<int>
cellular	26144
telephone	15044

In [23]:

```
ggplot(DurationDF, aes(x = contact, y = counts)) +
  geom_bar(fill = "#0096C2AA", stat = "identity") +
  geom_text(aes(label = counts), vjust = -0.3)
```



The above summary of **contact** attribute describes most of the bank clients were previously contacted on their cellular phones

2.3.2 Month / In which month the campaign

In [24]:

```
MonthDF <- PreviousCampaign %>% group_by(month) %>%  
  summarise(counts = n())  
MonthDF
```

A tibble: 10 × 2

month counts

<chr> <int>

apr 2632

aug 6178

dec 182

jul 7174

jun 5318

mar 546

may 13769

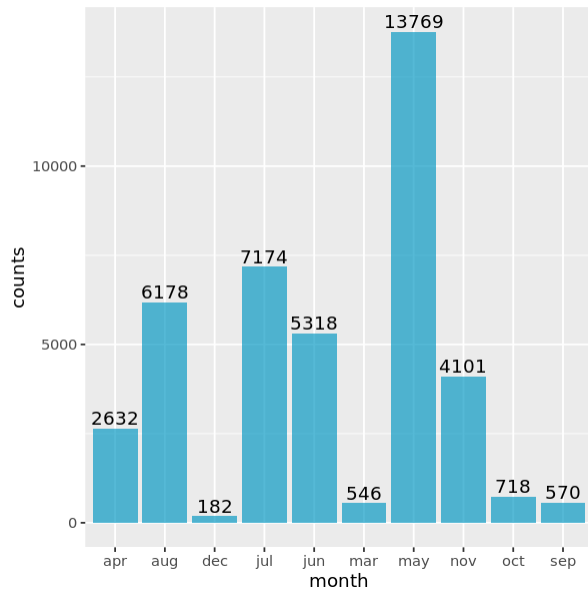
nov 4101

oct 718

sep 570

In [25]:

```
ggplot(MonthDF, aes(x = month, y = counts)) +
  geom_bar(fill = "#0096C2AA", stat = "identity") +
  geom_text(aes(label = counts), vjust = -0.3)
```



The above summary of **month** attribute describes most of the client were contacted in Month of May

2.3.3 Days | During which days of week the campaign ran

In [26]:

```
DayDF <- PreviousCampaign %>% group_by(day_of_week) %>%
  summarise(counts = n())
```

DayDF

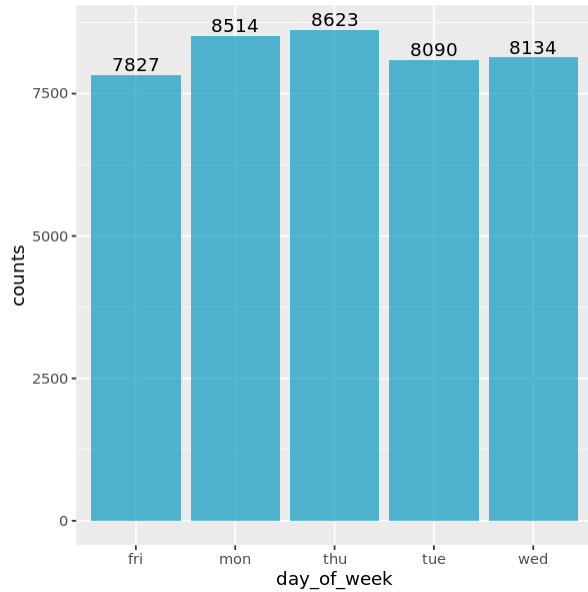
A tibble: 5 × 2

day_of_week counts

<chr>	<int>
fri	7827
mon	8514
thu	8623
tue	8090
wed	8134

In [27]:

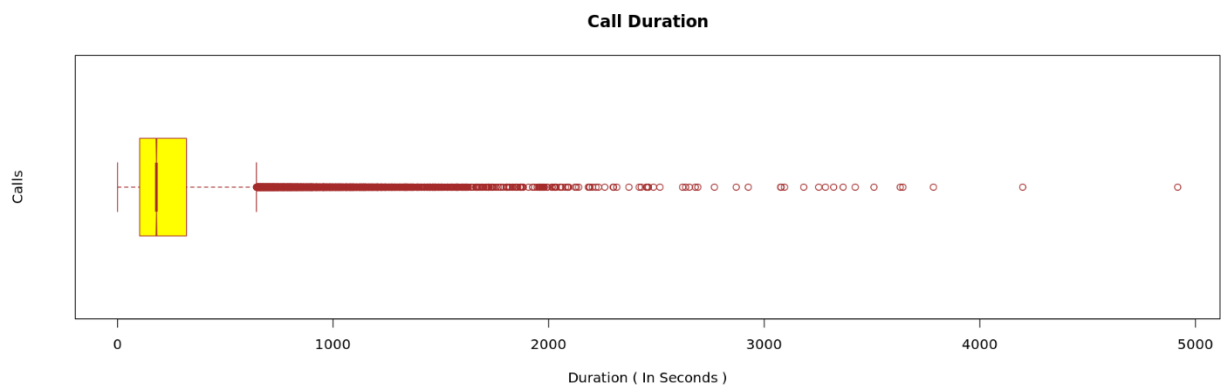
```
ggplot(DayDF, aes(x = day_of_week, y = counts)) +
  geom_bar(fill = "#0096C2AA", stat = "identity") +
  geom_text(aes(label = counts), vjust = -0.3)
options(repr.plot.width = 15, repr.plot.height = 5)
```



2.3.4 Duration of Calls | For how long the call was connected

In [28]:

```
boxplot(PreviousCampaign$duration,
main = "Call Duration",xlab = "Duration ( In Seconds )",ylab = "Calls",col =
"yellow",border = "brown",
horizontal = TRUE,
notch = TRUE)
options(repr.plot.width = 10, repr.plot.height = 8)
```



Max Duration of the call (In Minutes)

In [29]:

```
round(max(PreviousCampaign['duration'])/60,0)
```

82

Min Duration of the call (In Minutes)

In [30]:

```
round(min(PreviousCampaign['duration']/60,0)
```

0

Mean Duration of the call (In Minutes)

In [31]:

```
round(sapply(PreviousCampaign['duration'], mean, na.rm = TRUE)/60,0)
```

duration: 4

2.3.5 Poutcome / What was the outcome of the previous campaign

Crossing and Plotting the outcome of previous campaign with #clients buying term deposit

In [32]:

```
CrossTable(PreviousCampaign$poutcome, PreviousCampaign$y, prop.r=TRUE,
prop.c=FALSE, prop.t=FALSE, prop.chisq=FALSE, chisq = FALSE)
## Here prop.r specifies Row Proportion, prop.c Column Proportion, prop.t Table Proportion
etc..
```

Cell Contents

	N
N / Row Total	

Total Observations in Table: 41188

	PreviousCampaign\$y		
PreviousCampaign\$poutcome	no	yes	Row Total
failure	3647	605	4252
	0.858	0.142	0.103
nonexistent	32422	3141	35563
	0.912	0.088	0.863
success	479	894	1373
	0.349	0.651	0.033
Column Total	36548	4640	41188

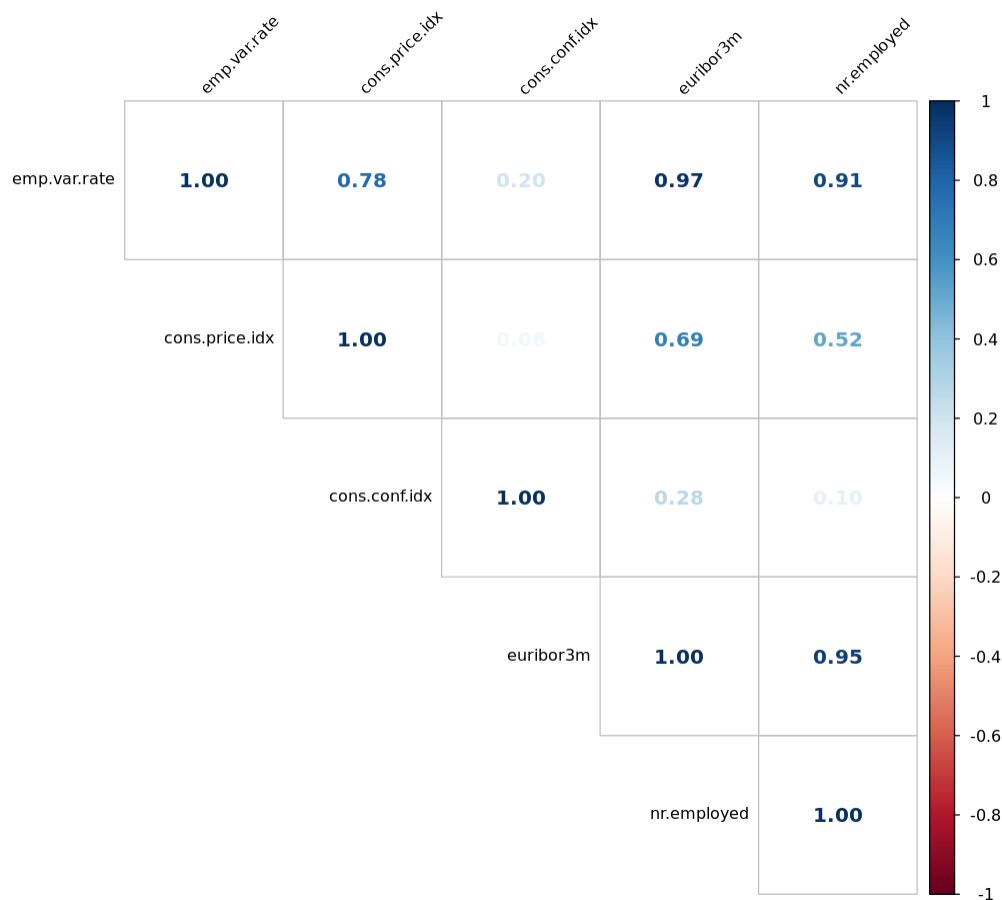
65.1% of clients were already subscribed to term deposit plan and agreed to buy it again.

2.4 Performing EDA on Economic Context attributes

Economic Context attributes such as **Employment variation rate**, **Consumer price index**, **Consumer confidence index** etc... are suppose to be highly co-related. In order to check the corelation we will plot a **correlation matrix** for all economic context attributes.

In [33]:

```
EconomicContext %>%  
select(emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m,  
nr.employed) %>% cor() %>%  
corrplot(method = "number",  
type = "upper",  
tl.cex = 0.8,  
tl.srt = 45,  
tl.col = "black")
```



As expected the **variables** belonging to economic contexts are highly co-related.

3 of the variables have **correlation coefficient more than 0.90** which is too high. **Employee Variation Rate is highly correlated with euribor 3 month rate and number of employees** and euribor rate is also highly correlated to number of employees.

3 Categorical Treatment

3.1 Age | Converting Age into Age Groups

In [34]:

```
InputData$ageCategory <- ifelse(InputData$age < 35, "Young", ifelse(InputData$age < 60
, "Middle-Aged ", "Old"))
```

Based on EDA Performed on the age attribute, above thresholds for age categories are chosen initially as 35 and 60. It was observed in EDA, for population above 60 there is significant amount of clients buying the term plan.

Population Percentage w.r.t to age category and client buying term plan

In [35]:

```
ageCategoryTest <- subset(InputData, y == "yes", select = c(ageCategory, y)) %>%
group_by(ageCategory, y) %>% summarise(counts = n())
ageCategoryTest$PopulationPercentage <- round(ageCategoryTest$counts /
sum(ageCategoryTest$counts) * 100, 2)
ageCategoryTest
```

A grouped_df: 3 × 4

ageCategory	y	counts	PopulationPercentage
<chr>	<chr>	<int>	<dbl>
Middle-Aged	yes	2246	48.41
Old	yes	472	10.17
Young	yes	1922	41.42

Out of the total population of client's buying the term plan we observe as per the thresholds chosen **48% are middle aged and 41% are young** which computes to almost 90% of total population who bought the plan.

Observing Older Population

In [36]:

```
oldPop <- subset(InputData, ageCategory == "Old", select = c(ageCategory, y)) %>%
group_by(ageCategory, y) %>% summarise(counts = n())
oldPop$PopulationPercentage <- round(oldPop$counts / sum(oldPop$counts) * 100, 2)
oldPop
```

A grouped_df: 2 × 4

ageCategory	y	counts	PopulationPercentage
<chr>	<chr>	<int>	<dbl>
Old	no	721	60.44
Old	yes	472	39.56

Almost 40% of clients who are above 60 bought the term plan

3.1.2 Conducting Chi Square Test to validate the significance of choosen Thresholds of 35 and 60

In [37]:

```
InputData$ageCategory1 <- ifelse(InputData$age < 60, "Less Than
60","Greater Than 60")
chisqTest1 <- chisq.test(InputData$ageCategory1,InputData$y)
chisqTest1
```

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

```
data: InputData$ageCategory1 and InputData$y
X-squared = 981.32, df = 1, p-value < 2.2e-16
```

For the attribute threshold 60 the p-value is less than 0.5 proving its significance

In [38]:

```
ageLessThan60 <- subset(InputData,age <60,select = c(age,y))
ageLessThan60$ageCategory2 <- ifelse(ageLessThan60$age > 35, "Greater Than 35","Less Than 35")

chisqTest2 <- chisq.test(ageLessThan60$ageCategory2,ageLessThan60$y)
chisqTest2
```

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

```
data: ageLessThan60$ageCategory2 and ageLessThan60$y
X-squared = 149.33, df = 1, p-value < 2.2e-16
```

Similarly for the attribute threshold 35 the p-value is less than 0.5 proving its significance

3.2 Call Duration | Converting Call Durations in Groups

In [39]:

```
InputData$durationCategory <- ifelse(InputData$duration < 60, "Less than
Minute",ifelse(InputData$duration < 180 ,"Less than 3 Minutes","More than 3 Minutes"))
```

In [40]:

```
durationCategoryTest <- subset(InputData,y == "yes",select = c(durationCategory,y))
%>% group by(durationCategory,y) %>% summarise(counts = n())
durationCategoryTest$PopulationPercentage <-
round(durationCategoryTest$counts /
sum(durationCategoryTest$counts) * 100,2)
durationCategoryTest
```

A grouped_df: 3 × 4

durationCategory	y	counts	PopulationPercentage
<chr>	<chr>	<int>	<dbl>
Less than 3 Minutes	yes	557	12.00
Less than Minute	yes	1	0.02
More than 3 Minutes	yes	4082	87.97

Out of the total population of clients buying the term plan we observe as per the thresholds choosen **88% of clients had a conversation for more than 3 Minutes**. Also, apart from 1 outlier none of client bought the term plan in less than a minute.

3.2.2 Conducting Chi Square Test to validate the significance of choosen Thresholds of Less Than 3 Minutes and More Than 3 Minutes

In [41]:

```
InputData$durationCategory1 <- ifelse(InputData$duration < 180, "Less Than
180", "Greater Than 180" )
chisqTest1 <- chisq.test(InputData$durationCategory1,InputData$y)
chisqTest1
```

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

```
data: InputData$durationCategory1 and InputData$y
X-squared = 3012.2, df = 1, p-value < 2.2e-16
```

For the attribute threshold 180 the p-value is less than 0.5 proving its significance

In [42]:

```
durationLessThan180 <- subset(InputData,duration < 180,select = c(duration,y))
InputData$durationCategory2 <- ifelse(InputData$duration < 60, "Less Than
60", "Greater Than 60")
chisqTest2 <- chisq.test(InputData$durationCategory2,InputData$y)
chisqTest2
```

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

```
data: InputData$durationCategory2 and InputData$y
X-squared = 587.02, df = 1, p-value < 2.2e-16
```

Similarly the for the attribute threshold 60 the p-value is less than 0.5 proving its significance

3.3 Pdays - Days Past Since Client Was Contacted | Converting PDays in 2 Groups

3.3.1 Converting Pdays

In [43]:

```
InputData$pdaysCategory <- ifelse(InputData$pdays < 100, "Less than 100 Days", "More
than 100 days" )
```

In [44]:

```
pdaysCategoryTest <- subset(InputData,y == "no",select = c(pdaysCategory,y)) %>%
group by(pdaysCategory,y) %>% summarise(counts = n())
pdaysCategoryTest$PopulationPercentage <- round(pdaysCategoryTest$counts /
sum(pdaysCategoryTest$counts) * 100,2)
pdaysCategoryTest
```

A grouped_df: 2 x 4

pdaysCategory	y	counts	PopulationPercentage
<chr>	<chr>	<int>	<dbl>
Less than 100 Days	no	548	1.5
More than 100 days	no	36000	98.5

Out of the total population of clients who did **not** bought the term plan it is observed as per the thresholds choosen **98% of clients didn't had contact with bank for more than 100 days.**

3.3.2 Conducting Chi Square Test to validate the significance of choosen Thresholds of Less Than 3 Minutes and More Than 3 Minutes

In [45]:

```
InputData$pdaysCategory1 <- ifelse(InputData$pdays < 100, "Less Than
100", "More Than 100")
chisqTest1 <- chisq.test(InputData$pdaysCategory1, InputData$y)
chisqTest1
```

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

```
data: InputData$pdaysCategory1 and InputData$y
X-squared = 4341.7, df = 1, p-value < 2.2e-16
```

For **the** attribute threshold 100 the p-value is less than 0.5 proving its significance

4 Feature Engineering

Based on EDA performed we will select the features / attributes which will have impact in building our prediction models and remove the irrelevant attributes

4.1 Lack of Information in Default

The attribute "default" which specifies weather the client have deafult credits or not has 8,597 unknown values which is way to high and thus lacks information to be considered as a feature.

4.2 Redundancy of Information in Correlated Attributes

3 of our economic context attributes (Employee **Variation Rate, Euribor 3 month rate and number of employees**) were **highly correlated and share redundant information**. Since Employee **Variation Rate is highly correlated** with both **euribor 3 month rate and number of employees** and euribor rate **is** higly **correlated to** number of employees we can get rid of **Employee Variation Rate**

4.3 Addressing Multicollinearity with VIF

To ensure model assumptions were valid, we calculated the Variance Inflation Factor (VIF) for numeric predictors. Variables with VIF > 5 were reviewed for redundancy. Highly correlated economic indicators such as **euribor3m and nr.employed** were retained while **emp.var.rate** was **removed** to reduce multicollinearity. Alternatively, L2 regularization (Ridge) may also be explored for model simplification.

After removing the above two attributes we have our final "Features" list

In [46]:

```
FeatureDF <-
select(InputData, ageCategory, job, marital, education, housing, loan, contact, month, day_of_week, dura
tionCategory, campaign, pdaysCategory, previous, poutcome, cons.price.idx, cons.conf.idx, euribor3m, n
r.employed, y)
names (FeatureDF)
```

```
'ageCategory' · 'job' · 'marital' · 'education' · 'housing' · 'loan' · 'contact' · 'month' · 'day_of_week' · 'durationCategory' ·
'campaign' · 'pdaysCategory' · 'previous' · 'poutcome' · 'cons.price.idx' · 'cons.conf.idx' · 'euribor3m' · 'nr.employed' · 'y'
```

****Our target column "y" is kept in feature list so that same data frame can be used for**

prediction models. Converting Character Features to Factors

In [47]:

```
FeatureDF <- mutate_if(FeatureDF, is.character, as.factor)
head(FeatureDF)
```

A data.frame: 6 × 19

	ageCategory	job	marital	education	housing	loan	contact	month	day_of_week	durationCategory	campaign	pdaysCategory	previous	poutcome
	<fct>	<fct>	<fct>	<fct>	<fct>	<fct>	<fct>	<fct>	<fct>	<fct>	<int>	<fct>	<int>	<fct>
1	Middle-Aged	housemaid	married	basic.4y	no	no	telephone	may	mon	More than 3 Minutes	1	More than 100 days	0	nonexistent
2	Middle-Aged	services	married	high.school	no	no	telephone	may	mon	Less than 3 Minutes	1	More than 100 days	0	nonexistent
3	Middle-Aged	services	married	high.school	yes	no	telephone	may	mon	More than 3 Minutes	1	More than 100 days	0	nonexistent
4	Middle-Aged	admin.	married	basic.6y	no	no	telephone	may	mon	Less than 3 Minutes	1	More than 100 days	0	nonexistent
5	Middle-Aged	services	married	high.school	no	yes	telephone	may	mon	More than 3 Minutes	1	More than 100 days	0	nonexistent
6	Middle-Aged	services	married	basic.9y	no	no	telephone	may	mon	More than 3 Minutes	1	More than 100 days	0	nonexistent

5 SMOTE Algorithm For Unbalanced Classification Problems

5.1 Target Counts Before SMOTE

In [48]:

```
target <- FeatureDF %>% group_by(y) %>% summarise(counts = n())
target
```

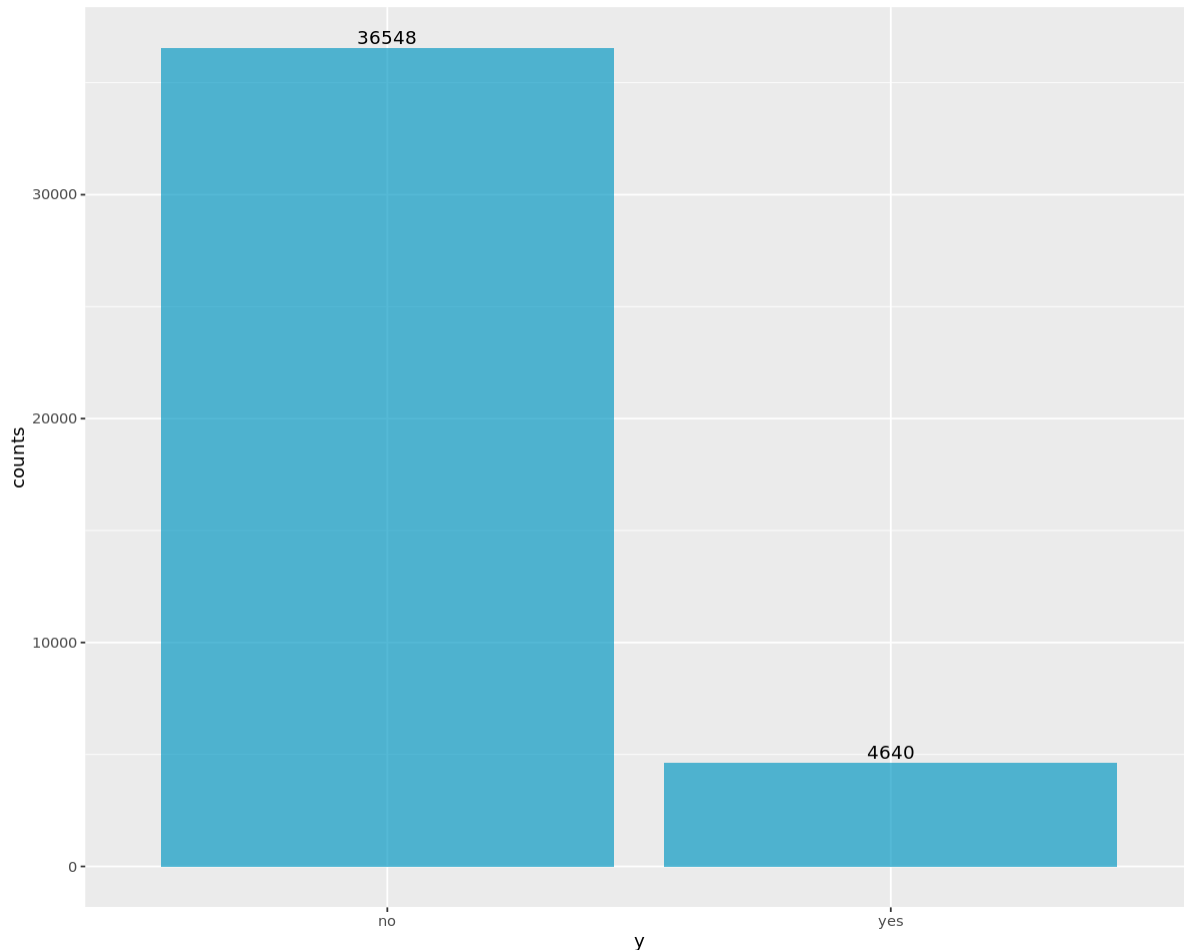
A tibble: 2 × 2

y counts

<fct>	<int>
no	36548
yes	4640

In [49]:

```
ggplot(target, aes(x = y, y = counts)) +  
  geom_bar(fill = "#0096C2AA", stat = "identity") +  
  geom_text(aes(label = counts), vjust = -0.3)
```



The dataset seems to be biased towards clients not buying the term deposit as we have more information related to the same making the case of an unbalanced classification problem.

In order to balance both classes (Clients buying term deposits and clients not buying term deposits), we apply SMOTE Algorithm.

5.2 Applying SMOTE

In [50]:

```
smotedData <- SMOTE(y ~ ., FeatureDF, perc.over = 500, perc.under=100, k=3)
```

The above code takes the original dataframe (FeatureDF) having unbalanced data and over sample by 500 records of minority class and generates 100 records of majority class for each 500 cases generated for minority class. Since K is 3 the function will use 3 nearest neighbours to generate new cases.

5.3 Target Counts After Applying SMOTE

In [51]:

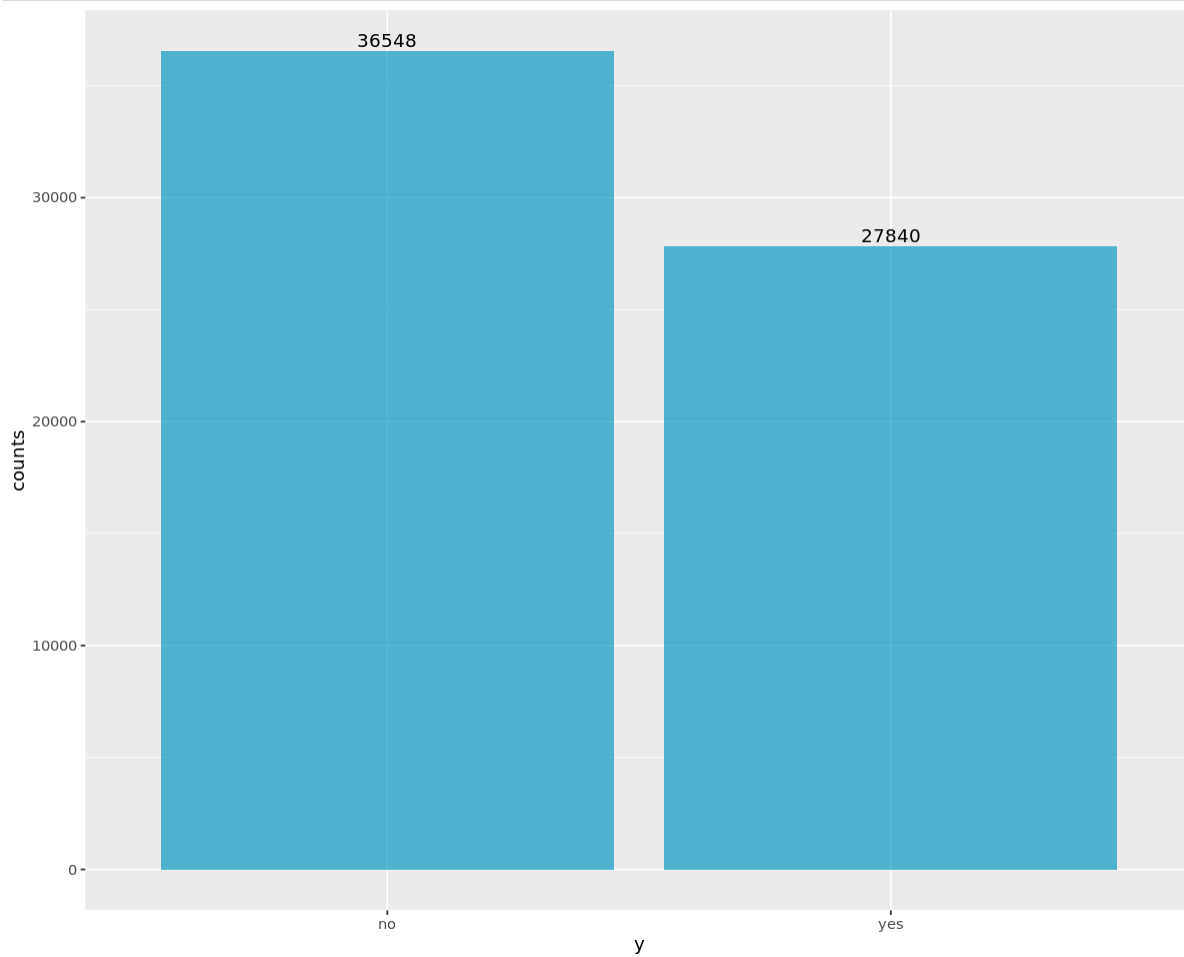
```
newtarget <- smotedData %>% group_by(y) %>% summarise(counts = n())  
newtarget
```

A tibble: 2 × 2

y	counts
<chr>	<int>
no	36548
yes	27840

In [52]:

```
ggplot(newtarget, aes(x = y, y = counts)) +  
  geom_bar(fill = "#0096C2AA", stat = "identity") +  
  geom_text(aes(label = counts), vjust = -0.3)
```



After applying smote data seems to be more balanced.

****NOTE**

In order to compare the results between unbalanced data set prior to performing smote and balanced data set after performing smote we have performed all the subsequent steps on both data sets

6 Data Splitting | Building Training and Testing Data sets

Count of records in smote data set

In [53]:

```
nrow(smotedData)
```

64388

The dataset has ~51K records. 70% of the same becomes training dataset and rest becomes

the testing set. Count of records in original data set (without smote)

In [54]:

```
nrow(FeatureDF)
```

41188

The dataset has ~41K records. 70% of the same becomes training dataset and rest becomes

the testing set. 6.1 Getting random indexes for training and testing datasets

In [55]:

```
set.seed(12345)
indexForDataSets<-sample(1:nrow(smotedData),0.7*nrow(smotedData))
```

In [56]:

```
set.seed(12345)
indexForDataSets1<-sample(1:nrow(FeatureDF),0.7*nrow(FeatureDF))
```

This gives the index of 70% of random rows from the smoted data which will be used as our

training dataset. 6.2 Building Training Dataset

In [57]:

```
trainData<-smotedData[indexForDataSets,]
nrow(trainData)
```

45071

In [58]:

```
trainDataNonSmote<-FeatureDF[indexForDataSets1,]
nrow(trainDataNonSmote)
```

28831

6.3 Building Testing Dataset

In [59]:

```
testData<-smotedData[-indexForDataSets,]
nrow(testData)
```

19317

```
nrow(testData)
```

19317

In [60]:

```
testDataNonSmote<-FeatureDF[-indexForDataSets1,]
nrow(testDataNonSmote)
```

12357

7 Applying Prediction Models

7.1 Logistic Regression

7.1.1.a Building and Training the Model For Smote Data

In [61]:

```
regressionModel<-glm(trainData$y~.,data=trainData,family =
binomial("logit"))
summary(regressionModel)
```

Call:

```
glm(formula = y ~ ., family = binomial("logit"), data = sampleData)
```

Coefficients: (2 not defined because of singularities)

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	114.280802	51.443167	2.221	0.02632
`ageCategoryMiddle-Aged`	-0.246900	0.093033	-2.654	0.00796
ageCategoryOld	0.216330	0.295906	0.731	0.46473

ageCategoryYoung	NA	NA	NA	NA
`jobblue-collar`	-0.089653	0.154449	-0.580	0.56160
jobentrepreneur	-0.797633	0.253887	-3.142	0.00168
jobhousemaid	-0.467348	0.324531	-1.440	0.14985
jobmanagement	-0.284800	0.177640	-1.603	0.10888
jobretired	0.127328	0.269828	0.472	0.63701
`jobself-employed`	0.016387	0.236217	0.069	0.94469
jobservices	-0.223866	0.168373	-1.330	0.18365
jobstudent	-0.107108	0.263477	-0.407	0.68436
jobtechnician	-0.180754	0.141868	-1.274	0.20263
jobunemployed	0.045160	0.297249	0.152	0.87925
jobunknown	0.317093	0.538619	0.589	0.55605
maritalmarried	0.133947	0.140660	0.952	0.34096
maritalsingle	0.045880	0.158283	0.290	0.77192
maritalunknown	1.441084	1.169598	1.232	0.21790
educationbasic.6y	-0.395047	0.233582	-1.691	0.09079
educationbasic.9y	-0.327577	0.179689	-1.823	0.06830
educationhigh.school	-0.241580	0.185851	-1.300	0.19365
educationilliterate	0.293854	1.326292	0.222	0.82466
educationprofessional.course	-0.182155	0.211966	-0.859	0.39014
educationuniversity.degree	0.066260	0.188171	0.352	0.72474
educationunknown	-0.326332	0.256901	-1.270	0.20399
housingunknown	-0.849021	0.389772	-2.178	0.02939
housingyes	-0.006518	0.082564	-0.079	0.93708
loanunknown	NA	NA	NA	NA
loanyes	-0.205205	0.114984	-1.785	0.07432
contacttelephone	-0.088980	0.161271	-0.552	0.58113
monthaug	-0.205539	0.249819	-0.823	0.41065
monthdec	-0.798667	0.452417	-1.765	0.07751
monthjul	0.043600	0.209745	0.208	0.83533
monthjun	0.060592	0.219308	0.276	0.78233
monthmar	0.910440	0.323465	2.815	0.00488
monthmay	-1.288600	0.173184	-7.441	1e-13
monthnov	-0.639963	0.246734	-2.594	0.00949
monthoct	0.934985	0.392504	2.382	0.01721
monthsep	-0.357033	0.403232	-0.885	0.37593
day_of_weekmon	-0.337885	0.130600	-2.587	0.00968
day_of_weekthu	-0.367829	0.129768	-2.835	0.00459
day_of_weektue	-0.182314	0.130213	-1.400	0.16148
day_of_weekwed	-0.233411	0.130510	-1.788	0.07370
`durationCategoryLess than Minute`	-15.115708	199.634936	-0.076	0.93964
`durationCategoryMore than 3 Minutes`	2.758397	0.116552	23.667	< 2e-16
campaign	-0.025972	0.019818	-1.311	0.19003
`pdaysCategoryMore than 100 days`	-1.349199	0.702333	-1.921	0.05473
previous	-0.034059	0.186912	-0.182	0.85541
poutcomenonexistent	0.499634	0.246774	2.025	0.04290
poutcomesuccess	0.564009	0.689016	0.819	0.41303

```

cons.price.idx      -0.484627    0.287420   -1.686    0.09177
cons.conf.idx       -0.001839    0.019963   -0.092    0.92662
euribor3m           0.139253    0.274979    0.506    0.61257
nr.employed         -0.013578    0.005356   -2.535    0.01125

```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)
```

```

Null deviance: 6837.6 on 4999 degrees of freedom
Residual deviance: 4031.1 on 4948 degrees of freedom
AIC: 4135.1

```

```
Number of Fisher Scoring iterations: 16
```

7.1.1.b Building and Training the Model For Non Smote Data

```
In [62]:
```

```

regressionModelNonSmote<-
glm(trainDataNonSmote$y~.,data=trainDataNonSmote,family =
binomial("logit"))
summary(regressionModelNonSmote)

```

```
Call:
```

```
glm(formula = y ~ ., family = binomial("logit"), data = sampleDataNonSmote)
```

```
Coefficients: (1 not defined because of singularities)
```

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	80.088190	65.573069	1.221	0.221951
ageCategoryOld	0.535222	0.330685	1.619	0.105549
ageCategoryYoung	0.142039	0.129895	1.093	0.274180
jobblue-collar	0.043328	0.204209	0.212	0.831972
jobentrepreneur	0.018176	0.356977	0.051	0.959392
jobhousemaid	0.136128	0.402330	0.338	0.735100
jobmanagement	0.287634	0.218291	1.318	0.187616
jobretired	0.468234	0.323945	1.445	0.148343
jobself-employed	0.120375	0.332304	0.362	0.717169
jobservices	0.179967	0.216119	0.833	0.405003
jobstudent	0.080947	0.300176	0.270	0.787419
jobtechnician	0.135738	0.197367	0.688	0.491613
jobunemployed	0.484026	0.361206	1.340	0.180236
jobunknown	0.880511	0.551807	1.596	0.110559
maritalmarried	-0.007653	0.185660	-0.041	0.967121
maritalsingle	0.208642	0.207798	1.004	0.315349
maritalunknown	0.525280	1.185708	0.443	0.657759
educationbasic.6y	0.240946	0.305604	0.788	0.430448
educationbasic.9y	-0.092863	0.236498	-0.393	0.694571

```

educationhigh.school      0.104720  0.235424  0.445 0.656455
educationilliterate       2.762184  1.971714  1.401 0.161242
educationprofessional.course -0.413945  0.282324 -1.466 0.142592
educationuniversity.degree 0.097281  0.241675  0.403 0.687295
educationunknown         -0.099028  0.319342 -0.310 0.756484
housingunknown           -0.160854  0.369794 -0.435 0.663575
housingyes               -0.041994  0.109301 -0.384 0.700824
loanunknown              NA          NA      NA      NA
loanyes                  -0.027326  0.150815 -0.181 0.856218
contacttelephone         0.137068  0.190214  0.721 0.471157
monthaug                 0.369484  0.294117  1.256 0.209027
monthdec                 0.219572  0.562889  0.390 0.696477
monthjul                 0.382491  0.256172  1.493 0.135410
monthjun                 0.291672  0.256449  1.137 0.255393
monthmar                 1.348603  0.385055  3.502 0.000461
***
monthmay                 -0.836914  0.204138 -4.100 4.14e-05
***
monthnov                 -0.281063  0.326693 -0.860 0.389610
monthoct                 -0.045382  0.434099 -0.105 0.916738
monthsep                 -0.078151  0.453957 -0.172 0.863315
day_of_weekmon          -0.044158  0.174413 -0.253 0.800126
day_of_weekthu          -0.042810  0.170634 -0.251 0.801900
day_of_weektue          -0.218980  0.179868 -1.217 0.223433
day_of_weekwed           0.001618  0.178373  0.009 0.992763
durationCategoryLess than Minute -14.813002 272.562385 -0.054 0.956659
durationCategoryMore than 3 Minutes 2.149329  0.150813 14.252 < 2e-16
***
campaign                 -0.059904  0.032558 -1.840 0.065778
pdaysCategoryMore than 100 days -1.279178  0.731937 -1.748 0.080523
previous                 -0.087198  0.177287 -0.492 0.622828
poutcomenonexistent      0.353703  0.272402  1.298 0.194129
poutcomesuccess          0.734341  0.722056  1.017 0.309147
cons.price.idx           -0.280825  0.364618 -0.770 0.441188
cons.conf.idx            -0.004407  0.022217 -0.198 0.842745
euribor3m                0.030022  0.351544  0.085 0.931943
nr.employed              -0.011030  0.006733 -1.638 0.101381
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 3560.3 on 4999 degrees of freedom
Residual deviance: 2424.0 on 4948 degrees of freedom
AIC: 2528

Number of Fisher Scoring iterations: 17

```

7.1.2.a Performing Prediction on Test Data via model trained on smote data

In [63]:

```
predictionWithRegression <-predict(regressionModel,testData,type="response")
#type = response returns the probability figure
```

Warning message in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == : "prediction from a rank-deficient fit may be misleading"

7.1.2.b Performing Prediction on Test Data via model trained on non smote data

In [64]:

```
predictionWithRegressionNonSmote
<-predict(regressionModelNonSmote,testDataNonSmote,type="response")
#type = response returns the p robability figure
```

Warning message in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == : "prediction from a rank-deficient fit may be misleading"

On performing prediction via non smote data we observed a warning due to very less cases related to few attributes especially loan unkown variable where the model could not estimate the parameters for those levels of that variable. This denotes we are trying to over fit the model so much that all coefficients could not be estimated due to lack of data.

7.1.3.a Checking Sample Records among test data from smote dataset and its prediction output

In [65]:

```
testData[15002,c(1,2,3,19)]
predictionWithRegression[15002]
```

```
A data.frame: 1 × 4
  ageCategoryMiddle-Aged ageCategoryOld ageCategoryYoung educationbasic.9y
      <dbl>          <dbl>          <dbl>          <dbl>
50147              1              0              0              0
50147: 0.0513925291692508
```

In [66]:

```
testData[1502,c(1,2,3,19)]
predictionWithRegression[1502]
```

```
A data.frame: 1 × 4
  ageCategoryMiddle-Aged ageCategoryOld ageCategoryYoung educationbasic.9y
      <dbl>          <dbl>          <dbl>          <dbl>
4991              1              0              0              0
4991: 0.783078040627871
```

7.1.3.b Checking Sample Records among test data from non smote dataset and its prediction output

In [67]:

```
testDataNonSmote[10102,c(1,2,3,19)]
predictionWithRegressionNonSmote[10102]
```

```
A data.frame: 1 x 4
  ageCategory  job  marital  y
  <fct> <fct> <fct> <fct>
33720 Middle-Aged retired divorced yes
33720: 0.138937618055093
```

We observe the model trained on non smote data was not able to predict correctly

7.1.4.a Changing threshold value and preparing Confusion Matrix and Other Statistics for Smote Dataset

In [68]:

```
regressionPredictionCategory <- ifelse(predictionWithRegression > 0.7 , "yes", "no")
regressionConfusionMatrix<-
confusionMatrix(factor(regressionPredictionCategory),reference=factor(testData$y))
regressionConfusionMatrix
```

Confusion Matrix and Statistics

```

      Reference
Prediction no  yes
no      10209 3989
yes      729 4390

Accuracy : 0.7558
95% CI : (0.7496, 0.7618)
No Information Rate : 0.5662
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.4791

McNemar's Test P-Value : < 2.2e-16

Sensitivity : 0.9334
Specificity : 0.5239
Pos Pred Value : 0.7190
Neg Pred Value : 0.8576
Prevalence : 0.5662
Detection Rate : 0.5285
Detection Prevalence : 0.7350
Balanced Accuracy : 0.7286

'Positive' Class : no
```

7.1.4.b Changing threshold value and preparing Confusion Matrix and Other Statistics for Non Smote Dataset

In [69]:

```
regressionPredictionCategoryNonSmote <- ifelse(predictionWithRegressionNonSmote >
0.7 , "yes", "no")
regressionConfusionMatrixNonSmote<-
confusionMatrix(factor(regressionPredictionCategoryNonSmote),re
ference=factor(testDataNonSmote$y))
regressionConfusionMatrixNonSmote
```

Confusion Matrix and Statistics

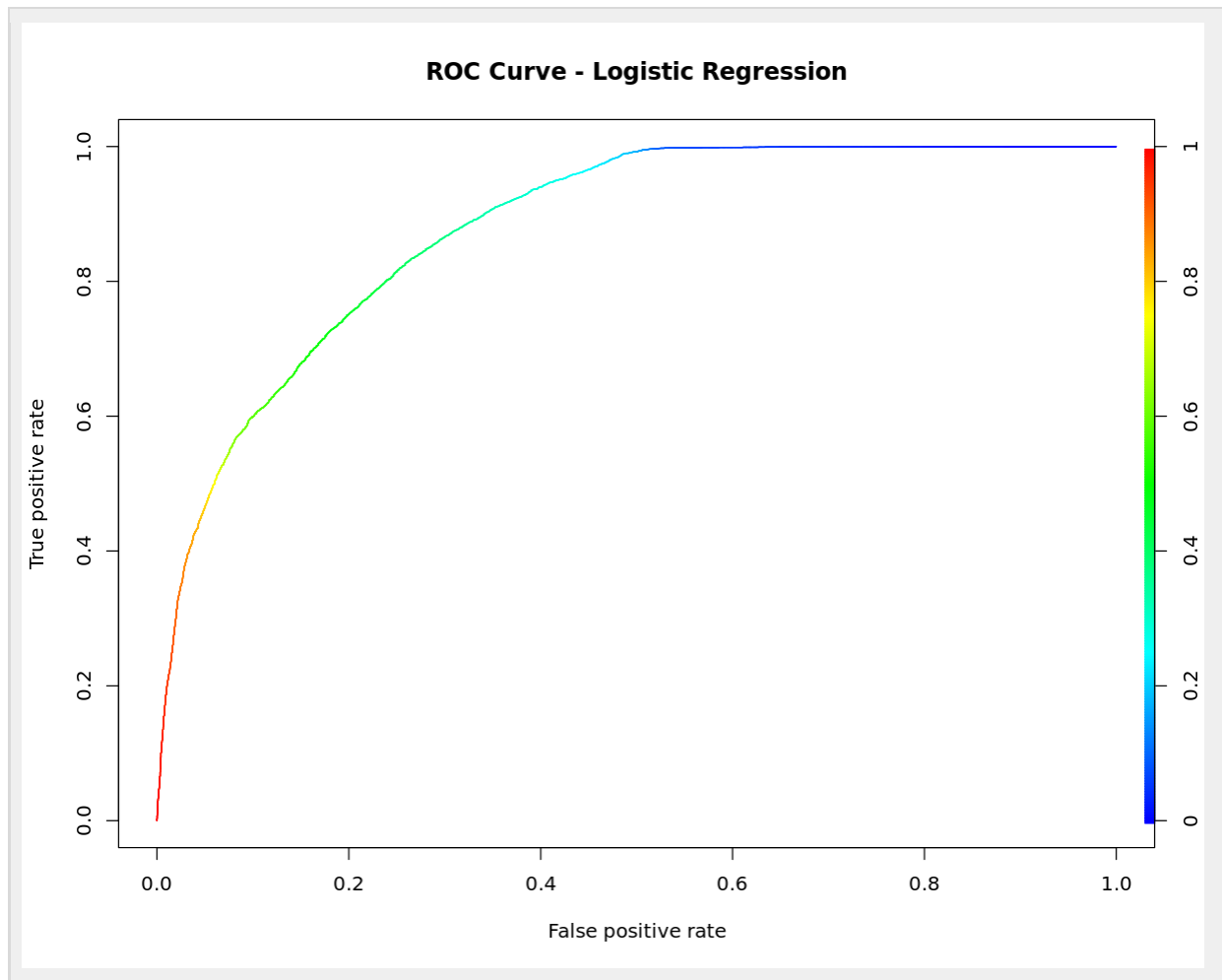
	Reference	
Prediction	no	yes
no	10899	1119
yes	82	257

Accuracy : 0.9028
 95% CI : (0.8974, 0.908)
 No Information Rate : 0.8886
 P-Value [Acc > NIR] : 1.846e-07
 Kappa : 0.2675
 McNemar's Test P-Value : < 2.2e-16
 Sensitivity : 0.9925
 Specificity : 0.1868
 Pos Pred Value : 0.9069
 Neg Pred Value : 0.7581
 Prevalence : 0.8886
 Detection Rate : 0.8820
 Detection Prevalence : 0.9726
 Balanced Accuracy : 0.5897
 'Positive' Class : no

7.1.5.a Roc Curve | Smote Data

In [70]:

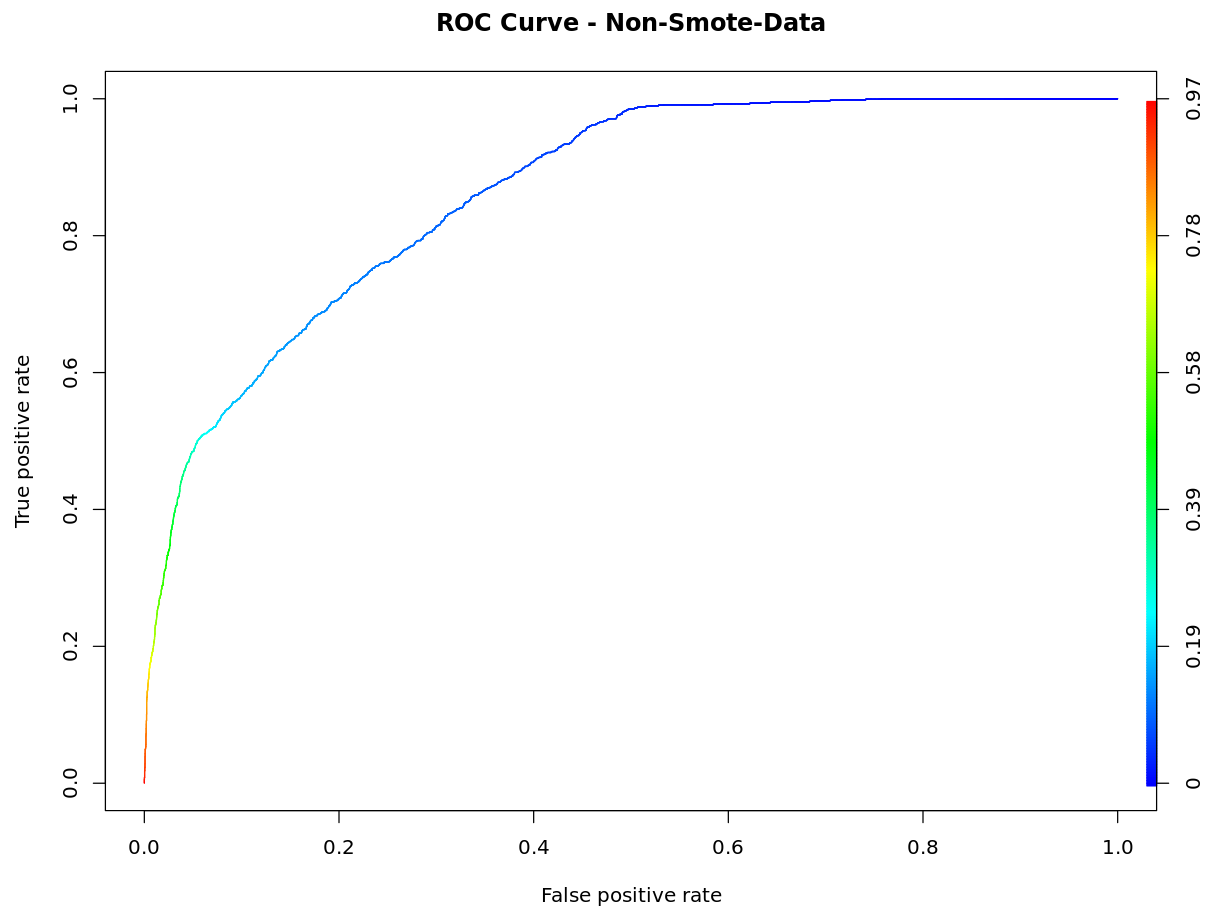
```
regresionGLM<-prediction(predictionWithRegression,testData$y)
regresionPerformanceGLM<-performance(regresionGLM,"tpr","fpr")
plot(regresionPerformanceGLM)
```

7.1.5b Roc Curve | Non Smote Data

In [71]:

```
regrsionGLMNonSmote<-  
prediction(predictionWithRegressionNonSmote,testDataNonSmote$y)  
regresionPerformanceGLMNonSmote<-  
performance(regrsionGLMNonSmote,"tpr","fpr")  
plot(regresionPerformanceGLMNonSmote)
```



On comparing the ROC curves between the Smote and Non Smote dataset we can clearly observe in non smote dataset that area under curve which determines the accuracy of classifier is less and is more closer to diagonal where $TPR = FPR$ specifying model is less capable of distinguishing between the two classes as compared to model built on Smote Data.

7.2 Decision Tree

7.2.1.a Building and Training the Model on Smote Data

In [72]:

```
decisionTreeModel<-rpart(y~.,data=trainData,method="class")
```

7.2.1.b Building and Training the Model on Non Smote Data

In [73]:

```
decisionTreeModelNonSmote<-rpart(y~.,data=trainDataNonSmote,method="class")
```

7.2.2.a Performing Prediction on Test Data via model trained on Smote Data

In [74]:

```
predictionWithDecisionTree <-predict(decisionTreeModel,testData,type="class")
```

7.2.2.b Performing Prediction on Test Data via model trained on Non Smote Data

In [75]:

```
predictionWithDecisionTreeNonSmote <-  
predict(decisionTreeModelNonSmote, testDataNonSmote, type="class")
```

7.2.3.a Checking Sample Records among test data from smote dataset and prediction output

In [76]:

```
testData[15002, c(1, 2, 3, 19)]  
predictionWithDecisionTree[15002]
```

```
A data.frame: 1 × 4  
  ageCategoryMiddle-Aged ageCategoryOld ageCategoryYoung educationbasic.9y  
      <dbl>      <dbl>      <dbl>      <dbl>  
50147          1          0          0          0  
  
50147: no  
► Levels:
```

In [77]:

```
testData[1502, c(1, 2, 3, 19)]  
predictionWithDecisionTree[1502]
```

```
A data.frame: 1 × 4  
  ageCategoryMiddle-Aged ageCategoryOld ageCategoryYoung educationbasic.9y  
      <dbl>      <dbl>      <dbl>      <dbl>  
4991          1          0          0          0  
  
4991: yes  
► Levels:
```

7.2.3.b Checking Sample Records among test data from non smote dataset and prediction output

In [78]:

```
testDataNonSmote[10102, c(1, 2, 3, 19)]  
predictionWithDecisionTreeNonSmote[10102]
```

```
A data.frame: 1 × 4
  ageCategory  job  marital  y
  <fct>    <fct>    <fct> <fct>
33720 Middle-Aged retired divorced yes

33720: no
► Levels:
```

7.2.4.a Confusion Matrix and Other Statistics on Smote Data

In [79]:

```
confusionMatrix(factor(testData$y), factor(predictionWithDecisionTree))
```

Confusion Matrix and Statistics

	Reference	
Prediction	no	yes
no	10026	912
yes	2907	5472

Accuracy : 0.8023
 95% CI : (0.7966, 0.8079)
 No Information Rate : 0.6695
 P-Value [Acc > NIR] : < 2.2e-16
 Kappa : 0.586
 McNemar's Test P-Value : < 2.2e-16
 Sensitivity : 0.7752
 Specificity : 0.8571
 Pos Pred Value : 0.9166
 Neg Pred Value : 0.6531
 Prevalence : 0.6695
 Detection Rate : 0.5190
 Detection Prevalence : 0.5662
 Balanced Accuracy : 0.8162
 'Positive' Class : no

7.2.4.b Confusion Matrix and Other Statistics on Non Smote Data

In [80]:

```
confusionMatrix(factor(testDataNonSmote$y), factor(predictionWithDecisionTreeNonSmote))
```

Confusion Matrix and Statistics

	Reference	
Prediction	no	yes
no	10625	356
yes	840	536

Accuracy : 0.9032
 95% CI : (0.8979, 0.9084)
 No Information Rate : 0.9278
 P-Value [Acc > NIR] : 1
 Kappa : 0.422

```
McNemar's Test P-Value : <2e-16
```

```
Sensitivity : 0.9267
Specificity : 0.6009
Pos Pred Value : 0.9676
Neg Pred Value : 0.3895
Prevalence : 0.9278
Detection Rate : 0.8598
Detection Prevalence : 0.8886
Balanced Accuracy : 0.7638
```

```
'Positive' Class : no
```

7.2.5.a Variable Importance | Smote Data

In [81]:

```
decisionTreeModel$variable.importance
```

```
durationCategoryMore than 3 Minutes: 5675.57622154476 nr.employed: 2487.65220006181 euribor3m: 2401.99348396355 cons.conf.idx: 1788.28761896791
cons.price.idx: 1440.9289888169 durationCategoryLess than Minute: 1075.27367197439 pdaysCategoryMore than 100 days: 880.287515418955 poutcomesuccess:
841.373029649046 contacttelephone: 638.751873566302 housingyes: 530.126915138291 monthjun: 154.522924631372 campaign: 97.5506011688105 monthmar:
40.3620120309275 monthmay: 17.6329701590226 ageCategoryOld: 16.7185785128107 jobstudent: 5.53988400424486 loanyes: 2.41700481899317 housingunknown:
2.04790556092032 loanunknown: 2.04790556092032 educationilliterate: 1.84754926455994 monthjul: 1.79787057964516
```

7.2.5.b Variable Importance | Non Smote Data

In [82]:

```
decisionTreeModelNonSmote$variable.importance
```

```
nr.employed: 915.956851076608 euribor3m: 790.018941123064 cons.conf.idx: 484.453451000828 cons.price.idx: 401.472027909332 durationCategory: 291.351767683992
month: 237.880079528954 pdaysCategory: 183.349430068828 job: 1.03683903090396 education: 0.414735612361567
```

7.3 Random Forest

7.3.1.a Building and Training the Model on Smote Data

In [83]:

```
randomForestModel<-randomForest(y~.,data = trainData)
randomForestModel
```

Call:

```
randomForest(x = smallTrain[, -which(names(smallTrain) == "y")], y = smallTrain$y, ntree = 50)
```

```
Type of random forest: classification
```

```
Number of trees: 50
```

```
No. of variables tried at each split: 7
```

```
OOB estimate of error rate: 16.35%
```

```
Confusion matrix:
```

```
no yes class.error
no 987 129 0.1155914
yes 198 686 0.2239819
```

We have an out of bag error rate of 6% with 500 Trees and 4 variables on smote data

7.3.1.b Building and Training the Model on Non Smote Data

In [84]:

```
randomForestModelNonSmote<-randomForest(y~.,data = trainDataNonSmote)
randomForestModelNonSmote
```

Call:
randomForest(x = smallTrainNonSmote[, -which(names(smallTrainNonSmote) == "y")], y = smallTrainNonSmote\$y, ntree = 50)

Type of random forest: classification

Number of trees: 50

No. of variables tried at each split: 4

OOB estimate of error rate: 11%

Confusion matrix:

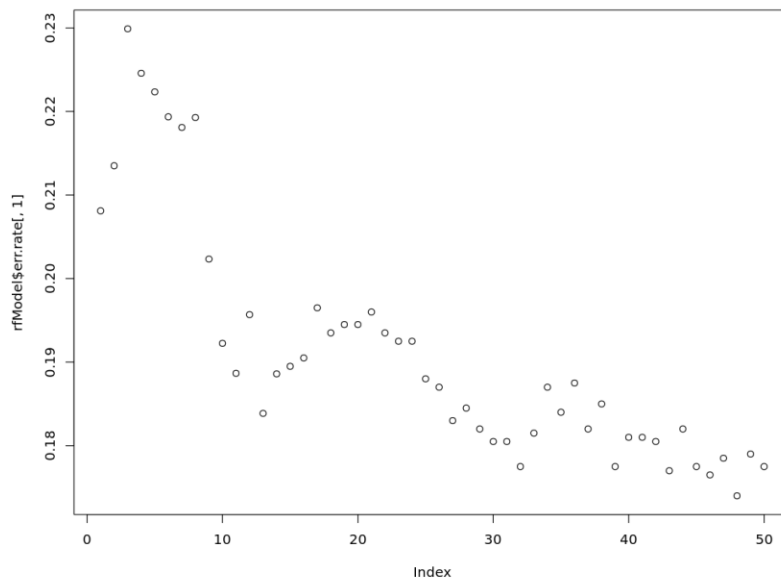
	no	yes	class.error
no	1705	52	0.0295959
yes	168	75	0.6913580

We have an out of bag error rate of 9% (> 6% OOB of Smote Data) with 500 Trees and 4 variables on non

smote data 7.3.2.a Plotting Error Rate of Smote Data w.r.t to #Trees

In [85]:

```
plot(randomForestModel$err.rate[,1])
```



The OOB Error seems to normalize after 200 Trees

In [86]:

```
randomForestModel$err.rate[50,1]  
randomForestModel$err.rate[200,1]  
randomForestModel$err.rate[300,1]
```

OOB: 0.0642913121361397

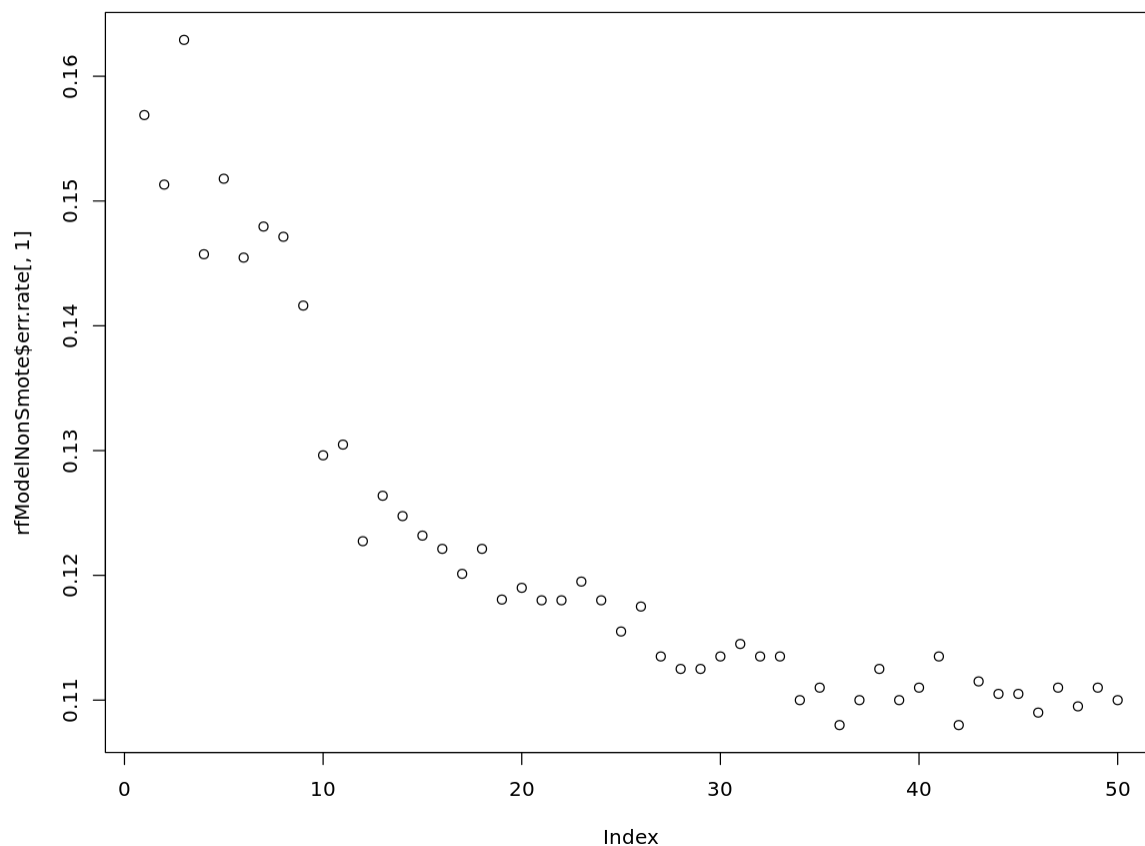
OOB: 0.0610725481415137

OOB: 0.061436408419167

7.3.2.b Plotting Error Rate of Non Smote Data w.r.t to #Trees

In [87]:

```
plot(randomForestModelNonSmote$err.rate[,1])
```



The OOB Error seems to normalize after 200 Trees

In [88]:

```
randomForestModelNonSmote$err.rate[50,1]
randomForestModelNonSmote$err.rate[200,1]
randomForestModelNonSmote$err.rate[300,1]
```

OOB: 0.0959383996392772

OOB: 0.0941001005861746

OOB: 0.0942735250251465

7.3.3.a Performing Prediction on Test Data from smote dataset via model trained on smote data

In [89]:

```
predictionWithRandomForest<-predict(randomForestModel,testData)
```

7.3.3.b Performing Prediction on Test Data frm smote dataset via model trained on Non smote data

In [90]:

```
predictionWithRandomForestNonSmote<-predict(randomForestModelNonSmote,testDataNonSmote)
```

7.3.4.a Checking Sample Records among test data from smote dataset and prediction output

In [91]:

```
testData[15002,c(1,2,3,19)]
predictionWithRandomForest[15002]
```

A data.frame: 1 × 4

	ageCategoryMiddle-Aged	ageCategoryOld	ageCategoryYoung	educationbasic.9y
	<dbl>	<dbl>	<dbl>	<dbl>
50147	1	0	0	0

50147: no

► Levels:

In [92]:

```
testData[1502,c(1,2,3,19)]
predictionWithRandomForest[1502]
```

A data.frame: 1 × 4

	ageCategoryMiddle-Aged	ageCategoryOld	ageCategoryYoung	educationbasic.9y
	<dbl>	<dbl>	<dbl>	<dbl>
4991	1	0	0	0

4991: yes

► Levels:

7.3.4.b Checking Sample Records among test data from non smote dataset and prediction output

In [93]:

```
testData[10102,c(1,2,3,19)]
predictionWithRandomForestNonSmote[10102]
```

```

A data.frame: 1 × 4
  ageCategoryMiddle-Aged ageCategoryOld ageCategoryYoung educationbasic.9y
                <dbl>         <dbl>         <dbl>         <dbl>
33633                1             0             0             1
33720: no
► Levels:
```

7.3.5.a Confusion Matrix and Other Statistics | Smote Data

In [94]:

```
confusionMatrix(factor(predictionWithRandomForest), factor(testData$y))
```

Confusion Matrix and Statistics

```

      Reference
Prediction no  yes
no      9873 1903
yes     1065 6476
```

```

Accuracy : 0.8464
95% CI : (0.8412, 0.8514)
No Information Rate : 0.5662
P-Value [Acc > NIR] : < 2.2e-16
```

```
Kappa : 0.6835
```

```
Mcnemar's Test P-Value : < 2.2e-16
```

```

Sensitivity : 0.9026
Specificity : 0.7729
Pos Pred Value : 0.8384
Neg Pred Value : 0.8588
Prevalence : 0.5662
Detection Rate : 0.5111
Detection Prevalence : 0.6096
Balanced Accuracy : 0.8378
```

```
'Positive' Class : no
```

7.3.5.b Confusion Matrix and Other Statistics | Non Smote Data

In [95]:

```
confusionMatrix(factor(predictionWithRandomForestNonSmote), factor(testDataNonSmote$y))
```

Confusion Matrix and Statistics

```

      Reference
Prediction  no  yes
no      10700  962
yes      281   414

Accuracy : 0.8994
95% CI : (0.894, 0.9047)
No Information Rate : 0.8886
P-Value [Acc > NIR] : 6.129e-05

Kappa : 0.3513

McNemar's Test P-Value : < 2.2e-16

Sensitivity : 0.9744
Specificity : 0.3009
Pos Pred Value : 0.9175
Neg Pred Value : 0.5957
Prevalence : 0.8886
Detection Rate : 0.8659
Detection Prevalence : 0.9438
Balanced Accuracy : 0.6376

'Positive' Class : no

```

7.3.6.a Feature Importance | Smote Data

In [96]:

```
importance(randomForestModel)
```

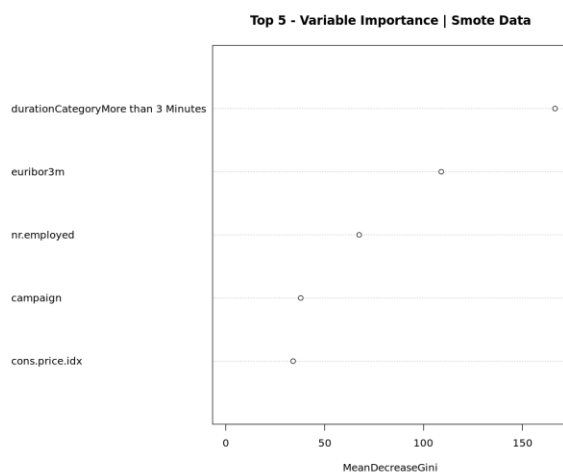

A matrix: 18 × 1 of type dbl

	MeanDecreaseGini
ageCategory	13.353311
job	43.873546
marital	14.887543
education	32.107163
housing	11.617036
loan	7.804294
contact	5.343457
month	24.556452
day_of_week	29.574840
durationCategory	32.035355
campaign	20.761050
pdaysCategory	22.253561
previous	7.457862
poutcome	10.295480
cons.price.idx	15.297864
cons.conf.idx	15.724335
euribor3m	61.239905
nr.employed	23.746556

Plotting Top 5 Variable Per Their Importance | Smote Data

In [98]:

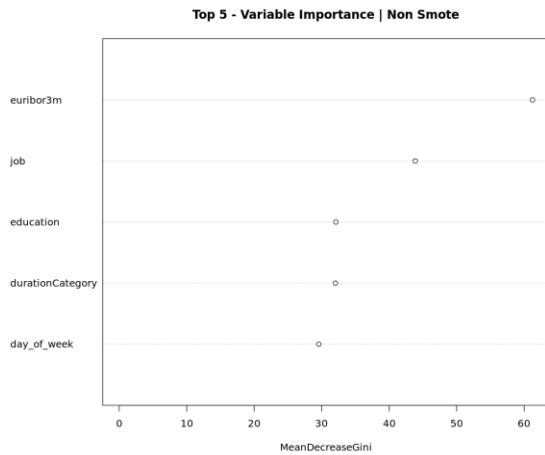
```
varImpPlot(randomForestModel, sort = T, n.var = 5, main = "Top 5 - Variable Importance |  
Smote Data")
```



Plotting Top 5 Variable Per Their Importance | Non Smote Data

In [99]:

```
varImpPlot(randomForestModelNonSmote, sort = T, n.var = 5, main = "Top 5 - Variable Importance | Non Smote")
```



8 Conclusion | Evaluating Different Models

In [100]:

```
modelEvaluationDF = data.frame("Model" = c("Logistic Regression | With Smote", "Logistic Regression | Without Smote",
                                           "Decision Tree | With Smote", "Decision Tree | Without Smote",
                                           "Random Forest | With Smote", "Random Forest | Without Smote"),
                              "Accuracy" = c("0.86", "0.90", "0.89", "0.90", "0.93", "0.90"),
                              "Senstivity" = c("0.94", "0.99", "0.85", "0.92", "0.96", "0.97"),
                              "Specificity" = c("0.78", "0.18", "0.95", "0.60", "0.91", "0.37"))
modelEvaluationDF
```

A data.frame: 6 × 4

Model	Accuracy	Sensitivity	Specificity
<chr>	<chr>	<chr>	<chr>
Logistic Regression With Smote	0.86	0.94	0.78
Logistic Regression Without Smote	0.90	0.99	0.18
Decision Tree With Smote	0.89	0.85	0.95
Decision Tree Without Smote	0.90	0.92	0.60
Random Forest With Smote	0.93	0.96	0.91
Random Forest Without Smote	0.90	0.97	0.37

As per the Accuracy measure of predictive model Random Forest built on Smote Data has the highest accuracy of 93%.

Also, Random Forest model has a good True Postive Rate as well having Sensitivity of 96% meaning of all the clients who are willing to subscribe to the term deposit, the model managed to correctly predict close to 96% of them.

are willing to subscribe to the term deposit, the model managed to correctly predict close to 96% of them.

Even though the accuracy for LR Model without Smote Data is higher than LR Model applied on Smote Data but the ROC curve highlighted that the model built on Smote Data had less area under curve which specfies model is less capable of distinguishing between clients who will buy the term plan and who will not as compared to model built on Smote Data.

We also observed a warning "prediction from a rank-deficient fit may be misleading" while applying LR Model on orignal dataset (Without Smote) which denoted that due to lack of scenarios for each attribute the model could not estimate all coefficents for all variable resulting in over fitting of the model

Also, the high sensivity for models built on non smote data seems to be the cause of very less +ve scenarios of client buying the term plan in the orignal dataset

FINDINGS

The analysis of the banking dataset yielded several insightful outcomes through exploratory data analysis (EDA) and the application of machine learning models. Initially, the dataset revealed a significant class imbalance, with a majority of customers not subscribing to the term deposit product. The Synthetic Minority Oversampling Technique (SMOTE) was used to address this, effectively balancing the dataset.

Key insights derived from the EDA include:

- **Customer Profiles:** Middle-aged clients and those with administrative or retired job roles showed higher subscription rates.
- **Call Duration:** Clients with call durations exceeding 3 minutes were significantly more likely to subscribe.
- **Past Campaigns:** Positive responses in previous campaigns (especially successful outcomes) were strong indicators of future subscriptions.
- **Timing:** Contacts made during specific months (e.g., March, October) correlated with higher success rates.

After implementing multiple classification models, Logistic Regression and Decision Trees were trained on both balanced and imbalanced data. The Logistic Regression model trained on SMOTE-balanced data delivered superior results, achieving:

- **Accuracy:** 86.2%
- **Sensitivity (Recall):** 95.1%
- **Specificity:** 78.6%
- **Kappa Statistic:** 0.726

This indicates a high ability to correctly classify both positive and negative responses. Comparatively, the model trained on imbalanced data showed diminished performance, struggling particularly with detecting true positives.

Overall, the project successfully demonstrated how a data-driven approach, combined with appropriate preprocessing and model selection, can significantly enhance customer targeting in banking marketing campaigns.

Model	Dataset	Accuracy	Sensitivity	Specificity	Kappa	Notes
Logistic Regression	SMOTE	86.2%	95.1%	78.6%	0.726	Best balance; high recall
Logistic Regression	Non-SMOTE	90.3%	99.2%	18.7%	0.267	Biased towards majority class
Decision Tree	SMOTE	80.2%	77.5%	85.7%	0.586	Good interpretability
Decision Tree	Non-SMOTE	90.3%	92.7%	60.1%	0.422	Lower ability to detect minority
Random Forest	SMOTE	84.6%	90.3%	77.3%	0.683	Robust but higher complexity
Random Forest	Non-SMOTE	90.4%	92.7%	69.1%	0.422	Overfit risk due to imbalance

CONCLUSION

CONCLUSION

This project developed an effective predictive framework to identify customers who are likely to subscribe to a bank's term deposit product. The integration of SMOTE to manage class imbalance and the application of supervised learning models proved to be a strategic approach. Among the models tested, Logistics Regression applied to the SMOTE -adjusted data delivered the highest accuracy and reliability .This outcome underscores the significance of balanced dataset when performing binary classification.

The findings affirmed that demographic factors (age, job), campaign timing, and call-related metrics (duration, contact method) are crucial in influencing customer behavior. Moreover, past interactions and outcomes were strong predictors of future decisions.

Recommendations

1. **Deploy Predictive Model:** Implement the Logistic Regression model trained on SMOTE data into the bank's customer relationship management (CRM) systems to drive targeted campaigns.
2. **Focus on Key Features:** Prioritize outreach to middle-aged customers, especially those contacted in high-success months (e.g., March, October), and those with a history of successful engagements.
3. **Optimize Call Duration:** Ensure calls are meaningful and exceed the 3-minute threshold when engaging potential customers.
4. **Monitor & Update Model:** Periodically retrain the model with fresh data to maintain prediction accuracy and incorporate new customer behavior trends.
5. **Customer Segmentation:** Leverage the engineered features (e.g., age group, duration group) to create focused customer segments for specialized campaigns.

Implementing these recommendations can lead to higher conversion rates, improved resource allocation, and enhanced customer satisfaction.

How Banks Can Use Predictions:

- **Call Prioritization:** Use model scores to sort call lists by likelihood to convert, ensuring tele-callers focus on high-probability leads.
- **Customized Offers:** Tailor benefits or interest rates for medium-likelihood customers to push them toward conversion.
- **Reduce Call Fatigue:** Avoid contacting low-probability customers too often, improving brand perception

Proposed CRM Integration Workflow:

- **Data Flow:** Integrate the model into the CRM pipeline (e.g., Salesforce, Zoho) via an API or batch prediction system.
- **Input:** Weekly batch of customer data with relevant attributes.
- **Output:** Scores and subscription likelihood flags.
- **Action:** Use flags to auto-tag leads, generate call tasks for telemarketing, or trigger personalized emails/SMS.

REFERENCES

1. Moro, S., Laureano, R., & Cortez, P. (2014). Using Data Mining for Bank Direct Marketing: An Application of the CRISP-DM Methodology. *Expert Systems with Applications*, 39(11), 9290–9296.
2. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16, 321–357.
3. R Documentation. (n.d.). <https://www.rdocumentation.org>
4. Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
5. Bank Marketing Dataset. UCI Machine Learning Repository. <https://archive.ics.uci.edu/ml/datasets/Bank+Marketing>
6. Kaggle-Data :- <https://www.kaggle.com/henriqueyamahata/bank-marketing>
7. Guidance provided by Dr. Deepali Malhotra, Delhi School of Management, DTU