HarvardX: PH125.9x Data Science
Portugal Banking term loan prediction

Mano Krishnan

02/04/2020

Contents

1 Introduction

We chose Bank Marketing Data set for the final project on Data science. This is Portuguese Banking institutional data. It has many attributes including client subscribed to term deposit or not. The aim is to build models that can predict if client will subscribe to term deposit or not.

1.1 Dataset and Data Loading

This dataset is download from UCI Machi Learning Repository. This is related to direct marketing campaigns of the Portuguese Banking institution. This dataset is available at http://archive.ics.uci.edu/ml/datasets/Bank+Marketing. There were 4 datasets in it from which bank-full.csv is used that has all examples (45211) and 17 inputs ordered by date. There are 16 input variables and 1 output variable (desired target).

This had different categories of client data like job, age, marital, education, default, housing, loan, contact, month, balance, day_of_week, duration, campaign, pdays, previous, poutcome and one output variable y that denotes if client subscribed to term deposit or not. These data denote telemarketing data, customer data and some other data. Here, many attributes are numerical and some are categorical. We loaded the dataset in the R studio and checked for any missing values using is.na fucntion and found not missing data. Hence, we have clean data.

1.1.1 Libraries

The following libraries were used in this report:

```
library(ggplot2)
library(tidyverse)
library(lubridate)
library(caret)
library(rpart)
library(rpart.plot)
library(RColorBrewer)
library(rattle)
library(descr)
library(randomForest)
```

1.1.2 Dataset loading

```
Portdata <- read_delim("bank-full.csv",
                       ";", escape_double = FALSE, trim_ws = TRUE)
## Duplicate row check
sum(duplicated(Portdata))
```

```
## [1] 0
```

```
## Missing data check
sum(!complete.cases(Portdata))
```

```
## [1] 0
```

```
all.empty = rowSums(is.na(Portdata))==ncol(Portdata)
sum(all.empty)
```

```
## [1] 0
```

```
Portdata.clean = Portdata[!all.empty,]

Portdata.clean = Portdata.clean %>% distinct

nrow(Portdata.clean)
```

```
## [1] 45211
```

```
Portdata.clean$missing = !complete.cases(Portdata.clean)

sum(is.na(Portdata))
```

```
## [1] 0
```

1.1.3 Attribute Information:

1. age – Client Age- (numeric)
2. job – Type of Job - (categorical) ('admin.','blue-collar','entrepreneur','housemaid','management', 'retired','selfemployed','services', 'student','technician','unemployed','unknown')
3. marital - Client's marital status - (categorical) (divorced, married, single, unknown, note: divorced means divorced or widowed)
4. education - Client's education - (categorical) (basic.4y, basic.6y, basic.9y, high.school, illiterate, professional.course, university.degree, unknown)
5. default - has credit in default? - (categorical) (no, yes, unknown)
6. housing - Has housing loan? - (categorical) (no, yes, unknown)
7. loan - has personal loan? - (categorical) (no, yes, unknown')
8. contact – last contact month of year - (categorical) (cellular, telephone)
9. month - Month of last contact with client - (categorical) (January - December)
10. day - last contact day of the month - (categorical) (1-31)
11. duration - last contact duration, in seconds (numeric). Important note: this attribute highly affects the output target (e.g., if duration=0 then y='no').

12. campaign: number of contacts performed during this campaign and for this client (numeric)

13. pdays - number of days that passed by after the client was last contacted from a previous campaign (numeric; 999 means clients were not previously contacted)

14. previous - Number of client contacts performed before this campaign - (numeric)
15. poutcome - outcome of the previous marketing campaign - (categorical) (failure, nonexistent, success)
16. balance - balance of the saving account - (numeric)

Output variable (desired target) –
17. Term Deposit - has the client subscribed a term deposit? - (binary: 'yes','no')

1.1.4 Aim & Objectives

The provided dataset will be divided into training set and validation set. We are training the first set with the machine learning algorithms and to predict subscription of term deposit by the client.

Data visualization and data exploration is used to find the interesting trends and the factors affecting the term deposit subscription by the client. We are creating many models based on their resulting accuracy and other attributes and finalizing the optimal model to client's subscription.

2 Methodology & Analysis

2.1 Data Pre-processing

For the better analysis, we have mutated many attributes based on the logical segregation. The below mentioned attributes are mutated.

1. Age_group - Grouped age data into 4 categories.

If age is less than 21, we consider it as 'Below 20' group. If age is between 21 and 40, we consider it as 'Between 21 and 40' group. If age is between 41 and 60, we consider it as 'Between 41 and 60' group. If age is greater than 60, we consider it as 'Above 61' group.

2. Cust_group - Grouped balance data into 4 categories

If balance is less than 0, we consider it as 'Below zero' group. If balance is between 1 and mean of the total balance amount , we consider it as 'below average' group. If balance is between mean of the total balance amount and 50000 , we consider it as 'Above average' group. If balance is greater than 50000, we consider it as 'HNI' group.

3. day_group - Grouped day data into 4 categories

If day is between 1 and 10 , we consider it as 'Early Month' group. If day is between 11 and 20 , we consider it as 'Mid month' group. If day is between 21 and 31, we consider it as 'Month end' group.

4. Duration_group - Grouped duration data into 4 categories

If duration is between 1 and mean of the duration , we consider it as 'Less Duration' group. If duration is between mean of the duration and 750 , we consider it as 'Good Duration' group. If duration is between 750 and 1500, we consider it as 'High Duration' group. If duration is above 1500, we consider it as 'Very High Duration' group.

5. Campaign_group - Grouped campaign data into 4 categories

If campaign is between 1 and mean of the campaign , we consider it as 'Average Campaign' group. If campaign is between mean of the campaign and 10, we consider it as 'Average Campaign' group. If campaign is between 10 and 20, we consider it as 'Ample Campaign' group. If campaign is above 20, we consider it as 'Heavy Campaign' group.

6. pdays_group - Grouped pdays data into 4 categories

If pdays is between -1 and mean of the pdays , we consider it as 'Less gap Pdays' group. If pdays is between mean of the pdays and 100, we consider it as 'Medium gap Pdays' group. If pdays is greater 100, we consider it as 'Large gap Pdays' group. 7. previous_group - Grouped previous data into 4 categories

If previous is 0 , we consider it as 'Zero' group. If previous is 1, we consider it as 'One' group. If previous is more than 1, we consider it as 'More than once' group.

8. y-output is "no", then 0 and if it is "yes", then 1.

```
Portdata <- as.data.frame(Portdata)

Portdata <- Portdata %>% mutate(Age_group = ifelse(age < 21,
            "Below 20", ifelse(between(age, 21,40),
            "Between 21 and 40", ifelse(between(age,41,60),
            "Between 41 and 60", "Above 61"))))

Portdata <- Portdata %>% mutate(Cust_group = ifelse(balance <0,
            "below zero", ifelse(between(balance,1,mean(balance)),
            "below average",
            ifelse(between(balance,mean(balance),50000),
            "Above average", "HNI" ))))

Portdata <- Portdata %>% mutate(day_group =
            ifelse(between(day,1,10),"Early Month",
            ifelse(between(day,11,20),"Mid Month", "Month End")))

Portdata <- Portdata %>% mutate(Duration_group =
            ifelse(between(duration,0,mean(duration)),"Less
            Duration", ifelse(between(duration,mean(duration),750),
            "Good Duration", ifelse(between(duration, 750, 1500),
            "High Duration", "Very High Duration" ))))


Portdata <- Portdata %>% mutate(Campaign_group =
            ifelse(between(campaign,1,mean(campaign)),
            "Lean Campaign", ifelse(between(campaign,
            mean(campaign),10),"Average Campaign",
            ifelse(between(campaign, 10, 20),
            "Ample Campaign", "Heavy Campaign" ))))

Portdata <- Portdata %>% mutate(pdays_group =
            ifelse(between(pdays,-1,mean(pdays)), "Less gap
            Pdays", ifelse( between(pdays,mean(pdays), 100),
            "Medium gap Pdays", "Large gap Pdays")))

Portdata <- Portdata %>% mutate(previous_group =
            ifelse(previous == 0, "Zero", ifelse(previous==1,
            "One", "More than once")))

Portdata <- Portdata %>% mutate(y_output = ifelse(y == "no", 0, 1))

Portdata <- Portdata %>% mutate(y_output = as.factor(y_output))

Portdata <- Portdata %>% select(-age,-balance,-day,-duration,-campaign,-pdays,-previous,-y)
```

```
Portdata <- Portdata %>% mutate(Age_group = as.factor(Age_group),
                                Cust_group = as.factor(Cust_group),
                                day_group = as.factor(day_group),
                                Duration_group = as.factor(Duration_group),
                                Campaign_group = as.factor(Campaign_group),
                                previous_group = as.factor(previous_group),
                                Duration_group = as.factor(Duration_group),
                                y_output = as.factor(y_output) )


Portdata=Portdata %>% mutate_if(is.character, as.factor)
```

2.2 Data Visualization and Data Exploration

2.2.1 General Data Information

```
# The few rows of the Portdata are presented below:
head(Portdata)
```

```
##            job marital education default housing loan contact month poutcome
## 1   management married  tertiary      no     yes   no unknown   may  unknown
## 2   technician  single secondary      no     yes   no unknown   may  unknown
## 3 entrepreneur married secondary      no     yes  yes unknown   may  unknown
## 4  blue-collar married   unknown      no     yes   no unknown   may  unknown
## 5      unknown  single   unknown      no      no   no unknown   may  unknown
## 6   management married  tertiary      no     yes   no unknown   may  unknown
##           Age_group    Cust_group   day_group              Duration_group
## 1 Between 41 and 60 Above average Early Month               Good Duration
## 2 Between 41 and 60 below average Early Month Less \n            Duration
## 3 Between 21 and 40 below average Early Month Less \n            Duration
## 4 Between 41 and 60 Above average Early Month Less \n            Duration
## 5 Between 21 and 40 below average Early Month Less \n            Duration
## 6 Between 21 and 40 below average Early Month Less \n            Duration
##   Campaign_group                   pdays_group previous_group y_output
## 1  Lean Campaign Less gap \n            Pdays           Zero        0
## 2  Lean Campaign Less gap \n            Pdays           Zero        0
## 3  Lean Campaign Less gap \n            Pdays           Zero        0
## 4  Lean Campaign Less gap \n            Pdays           Zero        0
## 5  Lean Campaign Less gap \n            Pdays           Zero        0
## 6  Lean Campaign Less gap \n            Pdays           Zero        0
```

```
# Summary Statistics of edx
summary(Portdata)
```

```
##           job           marital          education     default     housing
##  blue-collar:9732   divorced: 5207   primary  : 6851   no :44396   no :20081
##  management :9458   married :27214   secondary:23202   yes:  815   yes:25130
##  technician :7597   single  :12790   tertiary :13301
##  admin.     :5171                    unknown  : 1857
##  services   :4154
```

```
##  retired    :2264
##  (Other)    :6835
##   loan            contact          month          poutcome
##  no :37967   cellular :29285   may    :13766   failure: 4901
##  yes: 7244   telephone: 2906   jul    : 6895   other  : 1840
##              unknown  :13020   aug    : 6247   success: 1511
##                                jun    : 5341   unknown:36959
##                                nov    : 3970
##                                apr    : 2932
##                                (Other): 6060
##              Age_group             Cust_group          day_group
##  Above 61          : 1188   Above average:11730   Early Month:13725
##  Below 20          :   97   below average:26183   Mid Month  :18389
##  Between 21 and 40:24620   below zero   : 3766   Month End  :13097
##  Between 41 and 60:19306   HNI          : 3532
##
##
##
##                       Duration_group           Campaign_group
##  Good Duration              :12770   Ample Campaign  :  952
##  High Duration              : 2035   Average Campaign:13966
##  Less \n            Duration:30179   Heavy Campaign  :  244
##  Very High Duration         :  227   Lean Campaign   :30049
##
##
##
##                         pdays_group           previous_group  y_output
##  Large gap Pdays             : 6820   More than once: 5485   0:39922
##  Less gap \n              Pdays:37180   One           : 2772   1: 5289
##  Medium gap Pdays            : 1211   Zero          :36954
##
##
##
##
```

2.3 Data exploration

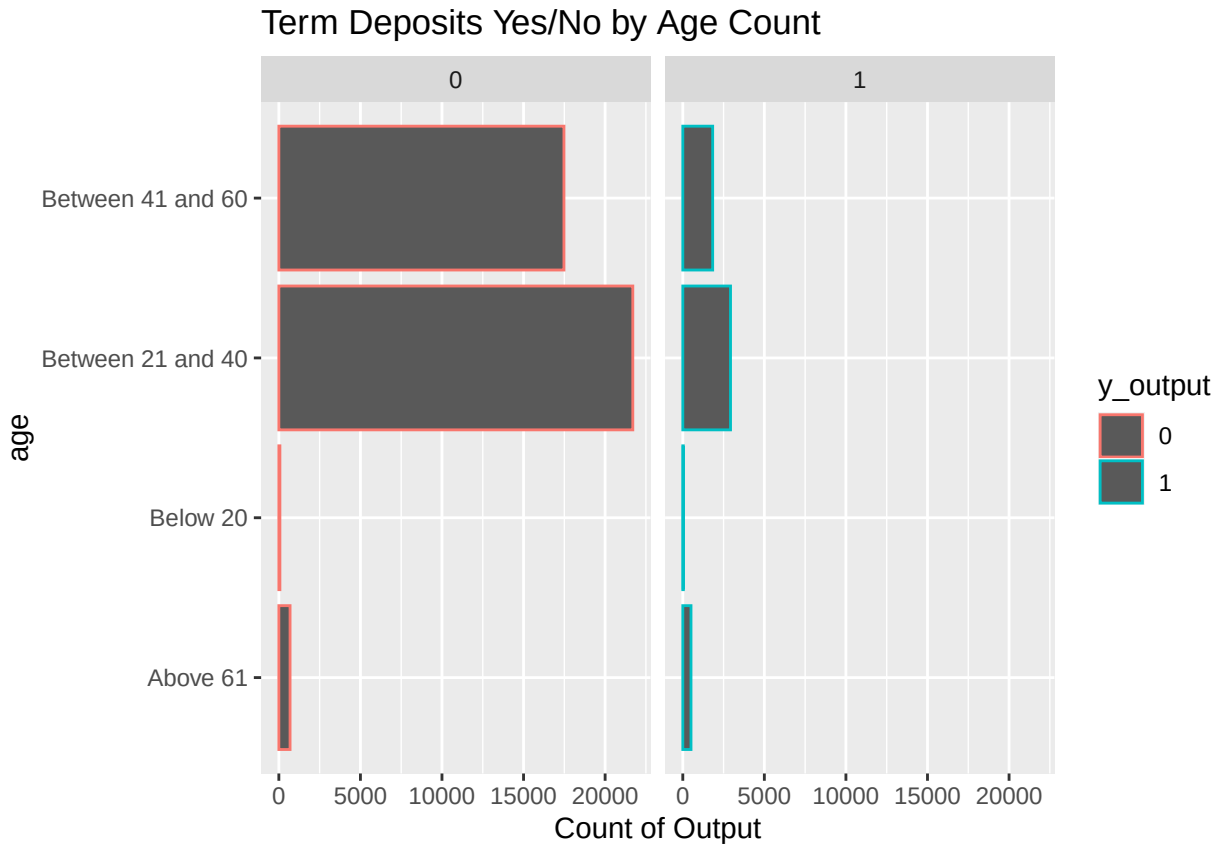
Each attribute of the dataset has been explored properly by comparing the count of the y_output.

```
### Age - categ

Age_count <- data.frame(Portdata %>% group_by(Age_group, y_output) %>% summarise(Count = n()))

Age_count <- Age_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>% mutate(Percent_o
arrange(desc(Percent_of_yes))

Age_count %>% knitr::kable()
```

Age_group	No	Yes	Percent_of_yes
Above 61	686	502	0.4225589
Below 20	64	33	0.3402062
Between 21 and 40	21696	2924	0.1187652

Age_group	No	Yes	Percent_of_yes
Between 41 and 60	17476	1830	0.0947892

```
Portdata %>% ggplot(aes(Age_group, color = y_output)) +
  geom_bar() +
  labs(title = "Term Deposits Yes/No by Age Count", x="age",
  y="Count of Output") +
  facet_wrap(y_output ~ .) +
  coord_flip()
```




```
theme_set(theme_bw())  # pre-set the bw theme.
g <- Portdata %>% ggplot( aes(y_output, Age_group)) +
labs(subtitle="Term deposit output vs Age Group",
Y="Age Group", x="Count of Output")

g + geom_jitter(aes(col=Age_group)) +
  geom_smooth(aes(col=Age_group), method="lm", se=F)
```

```
## `geom_smooth()` using formula 'y ~ x'
```

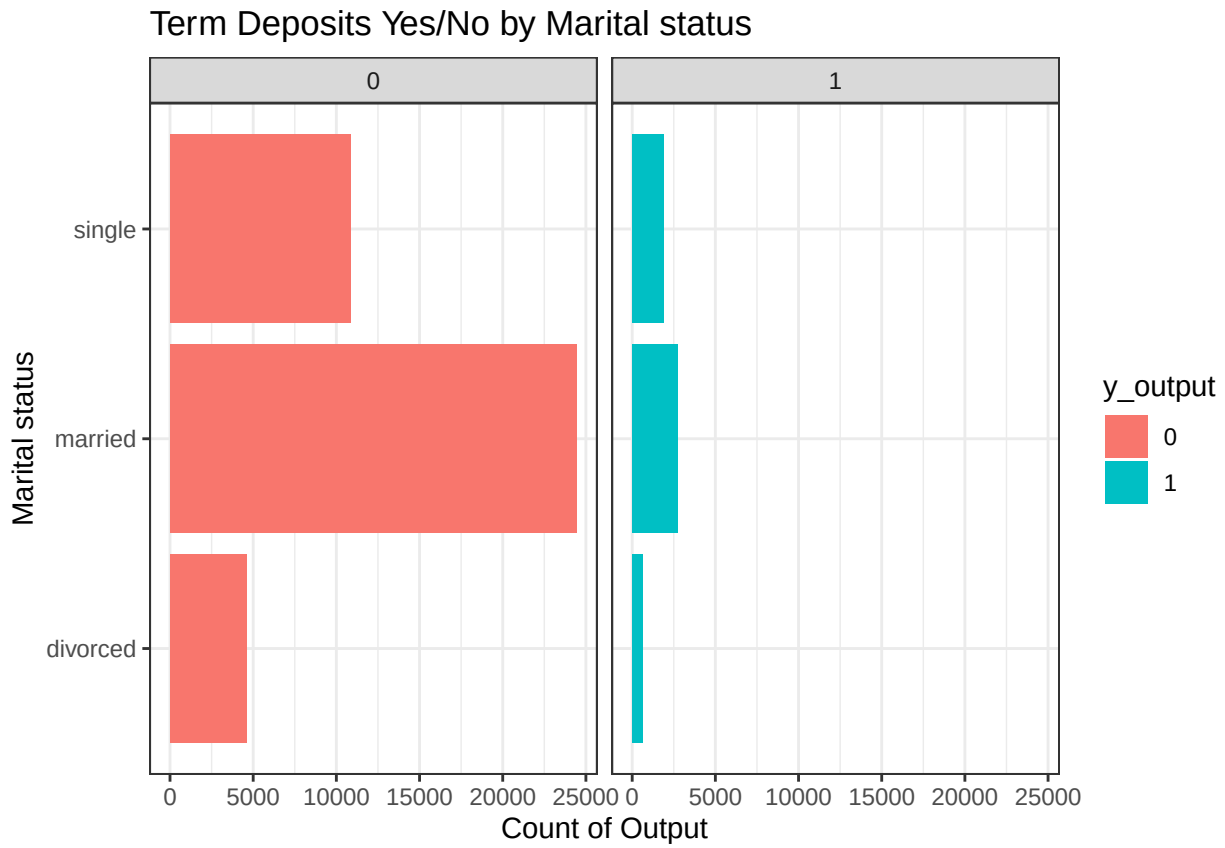

```
### Job

job_count <- data.frame(Portdata %>% group_by(job,y_output) %>% summarise(Count = n()))

job_count <- job_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

job_count %>% knitr::kable()
```

job	No	Yes	Percent_of_yes
student	669	269	0.2867804
retired	1748	516	0.2279152
unemployed	1101	202	0.1550269
management	8157	1301	0.1375555
admin.	4540	631	0.1220267
self-employed	1392	187	0.1184294
unknown	254	34	0.1180556
technician	6757	840	0.1105700
services	3785	369	0.0888300
housemaid	1131	109	0.0879032
entrepreneur	1364	123	0.0827169
blue-collar	9024	708	0.0727497

```
## Marital status

Marital_count <- data.frame(Portdata %>% group_by(marital,y_output) %>% summarise(Count = n()))

Marital_count <- Marital_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Marital_count %>% knitr::kable()
```

marital	No	Yes	Percent_of_yes
single	10878	1912	0.1494918
divorced	4585	622	0.1194546
married	24459	2755	0.1012347

```
Portdata %>% ggplot(aes(marital, fill = y_output)) +
  geom_bar() +
  labs(title = "Term Deposits Yes/No by Marital status", x="Marital status", y="Count of Output") +
  facet_wrap(y_output ~ .) +
  coord_flip()
```




```
## Education

Education_count <- data.frame(Portdata %>% group_by(education,y_output) %>% summarise(Count = n()))
```

```
Education_count <- Education_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Education_count %>% knitr::kable()
```

education	No	Yes	Percent_of_yes
tertiary	11305	1996	0.1500639
unknown	1605	252	0.1357027
secondary	20752	2450	0.1055943
primary	6260	591	0.0862648

```
## Default

Default_count <- data.frame(Portdata %>% group_by(default,y_output) %>% summarise(Count = n()))

Default_count <- Default_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Default_count %>% knitr::kable()
```

default	No	Yes	Percent_of_yes
no	39159	5237	0.1179611
yes	763	52	0.0638037

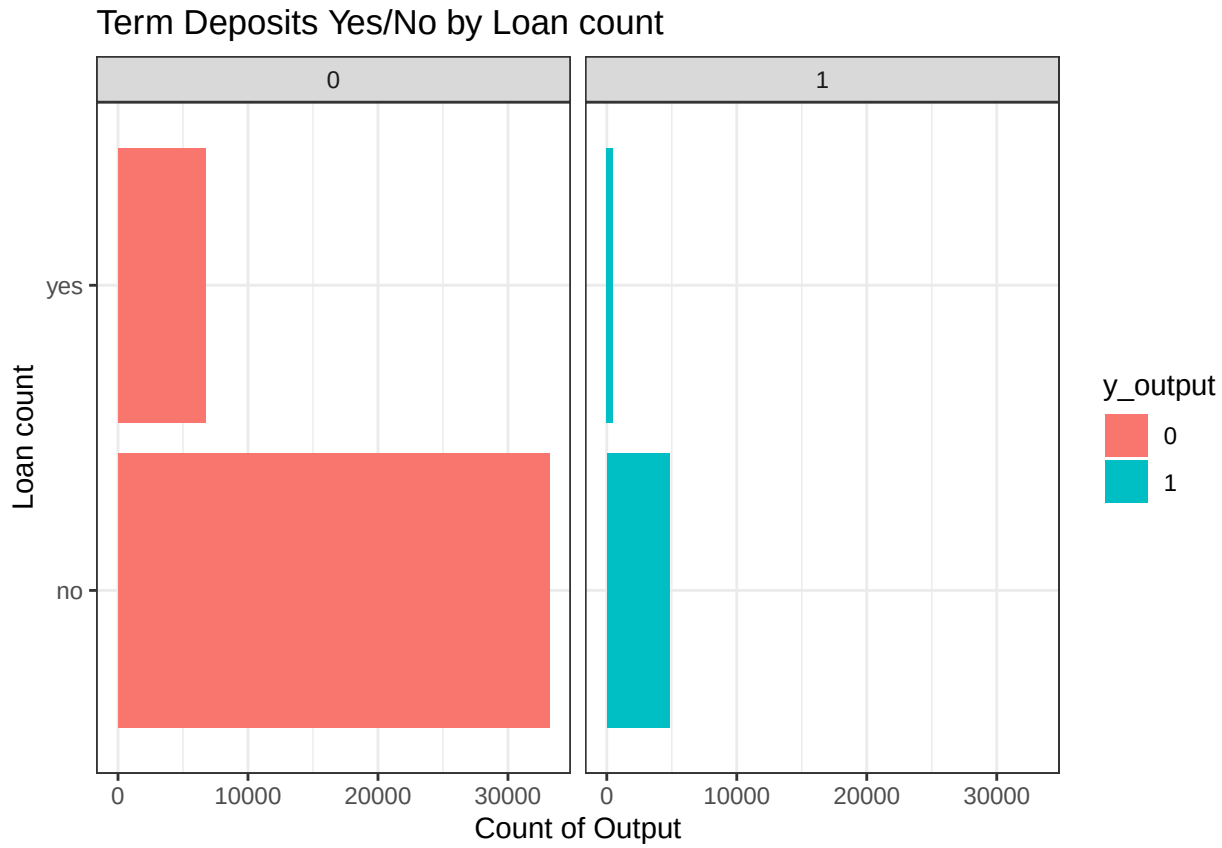
```
## Housing

Housing_count <- data.frame(Portdata %>% group_by(housing,y_output) %>% summarise(Count = n()))

Housing_count <- Housing_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Housing_count %>% knitr::kable()
```

housing	No	Yes	Percent_of_yes
no	16727	3354	0.1670236
yes	23195	1935	0.0769996

```
## Loan

Loan_count <- data.frame(Portdata %>% group_by(loan,y_output) %>% summarise(Count = n()))

Loan_count <- Loan_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Loan_count %>% knitr::kable()
```

loan	No	Yes	Percent_of_yes
no	33162	4805	0.1265573
yes	6760	484	0.0668139

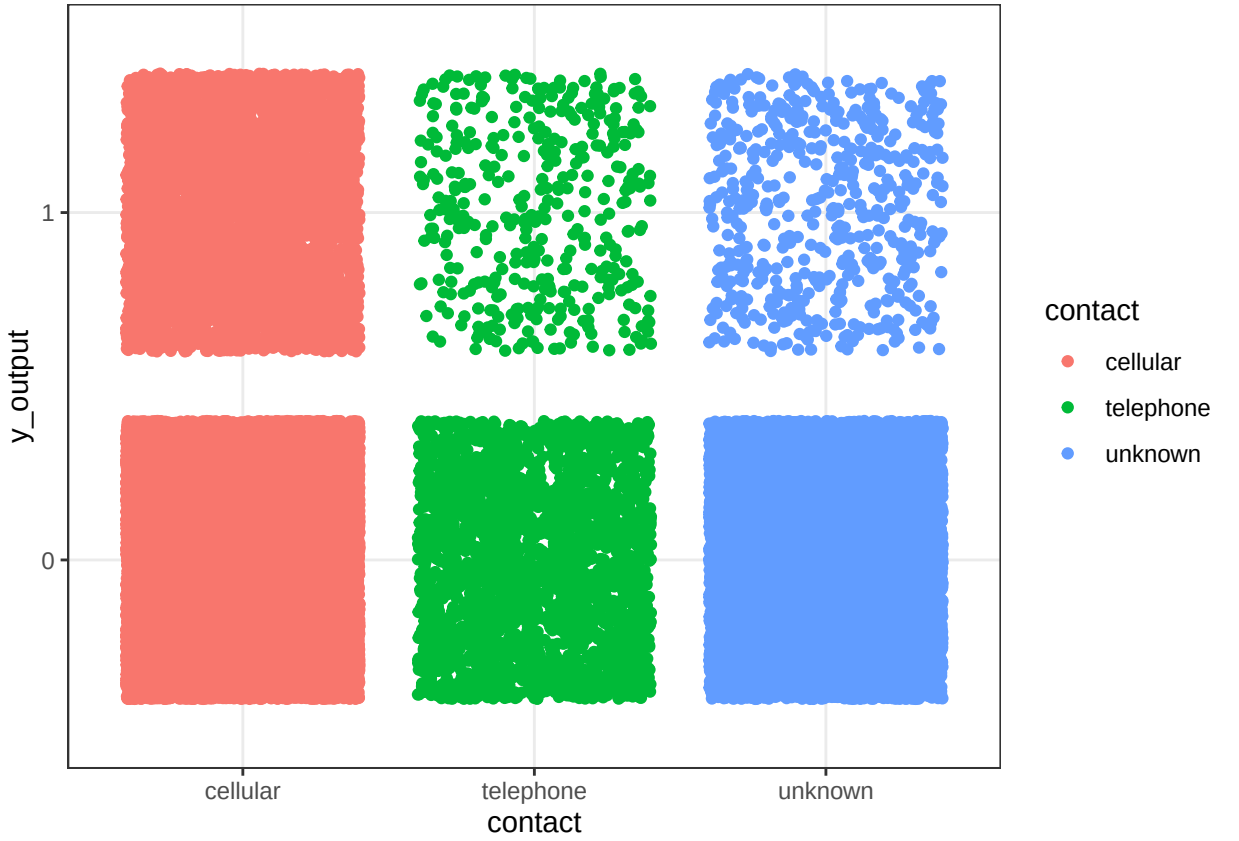
```
Portdata %>% ggplot(aes(loan, fill = y_output)) +
  geom_bar() +
  labs(title = "Term Deposits Yes/No by Loan count", x="Loan count", y="Count of Output") +
  facet_wrap(y_output ~ .) +
  coord_flip()
```




```
## Contact

Contact_count <- data.frame(Portdata %>% group_by(contact,y_output) %>% summarise(Count = n()))

Contact_count <- Contact_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Contact_count %>% knitr::kable()
```

contact	No	Yes	Percent_of_yes
cellular	24916	4369	0.1491890
telephone	2516	390	0.1342051
unknown	12490	530	0.0407066

```
Portdata %>% ggplot(aes(contact, y_output, color = contact)) +
  geom_jitter()
```

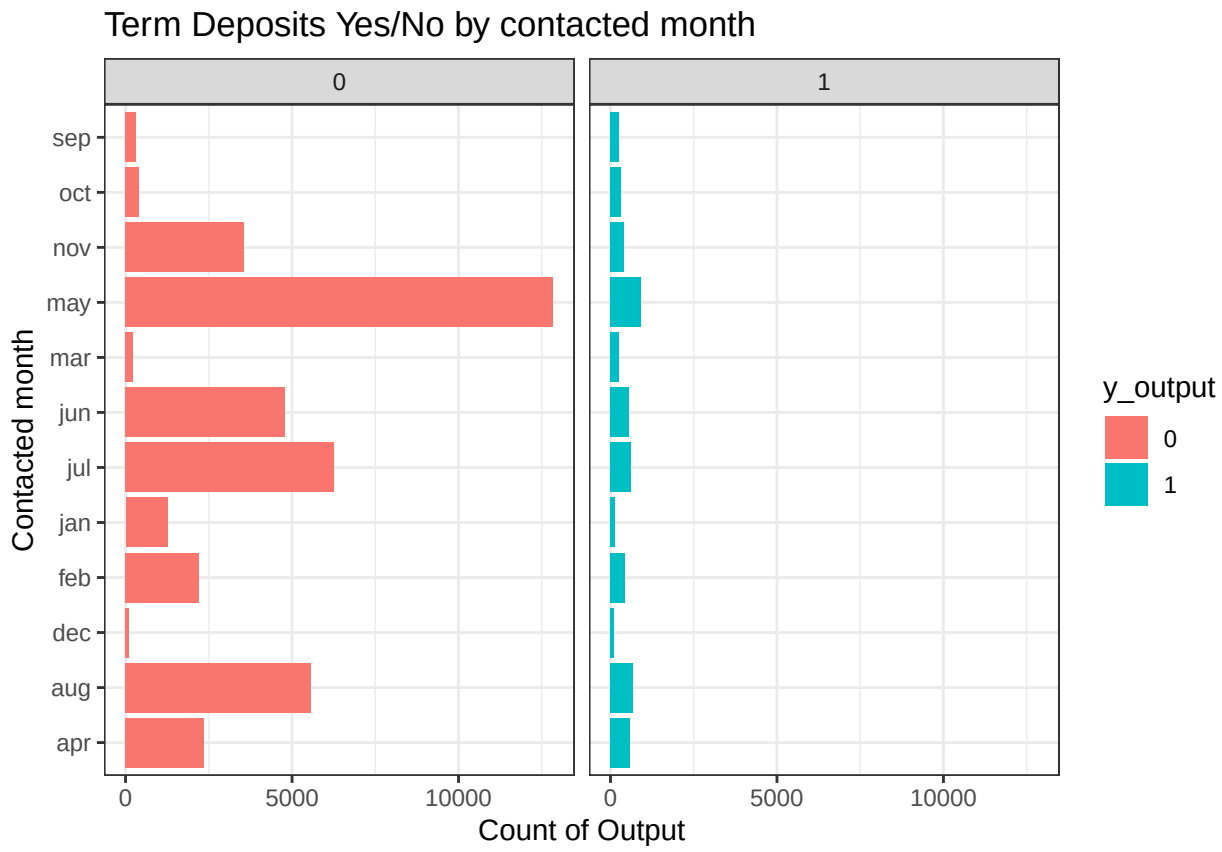



```
## Month - Categ

Month_count <- data.frame(Portdata %>% group_by(month,y_output) %>% summarise(Count = n()))

Month_count <- Month_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Month_count %>% knitr::kable()
```

month	No	Yes	Percent_of_yes
mar	229	248	0.5199161
dec	114	100	0.4672897
sep	310	269	0.4645941
oct	415	323	0.4376694
apr	2355	577	0.1967940
feb	2208	441	0.1664779
aug	5559	688	0.1101329
jun	4795	546	0.1022280
nov	3567	403	0.1015113
jan	1261	142	0.1012117
jul	6268	627	0.0909355

month	No	Yes	Percent_of_yes
may	12841	925	0.0671945

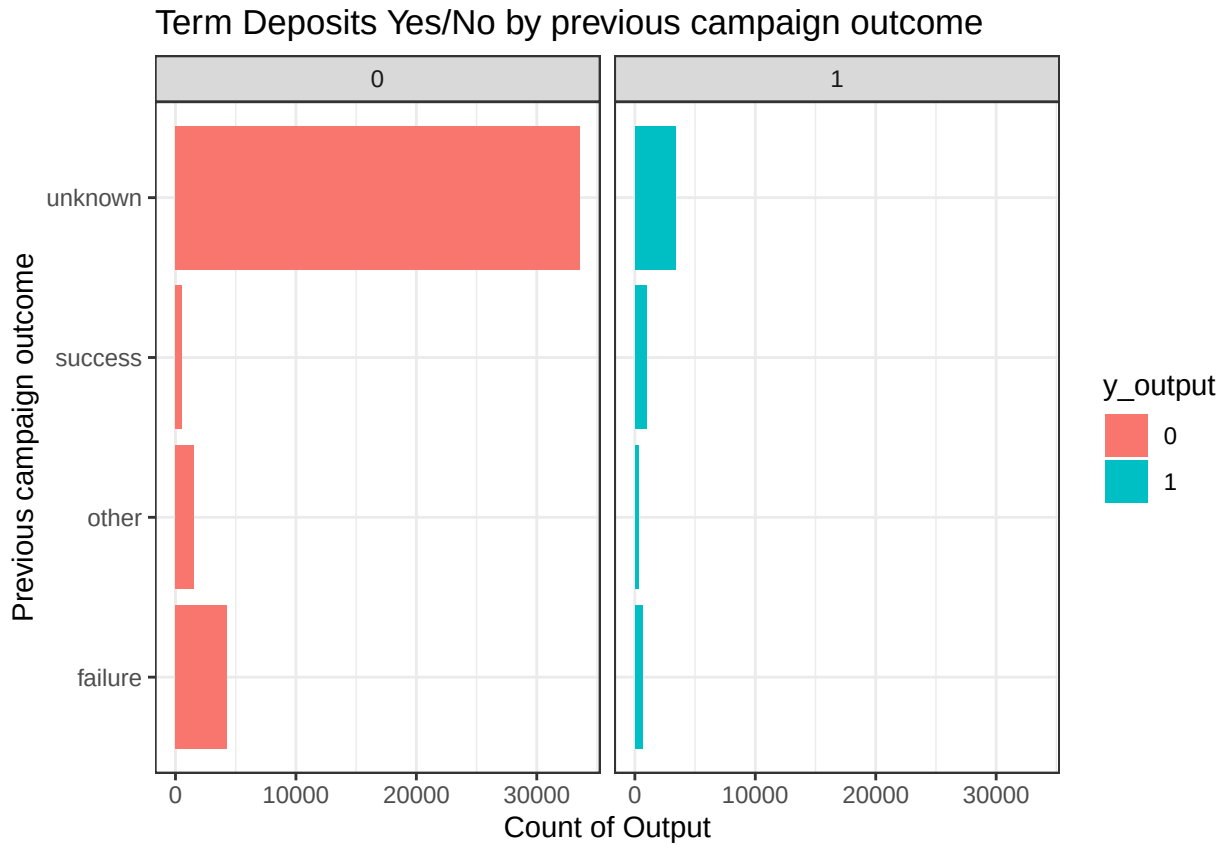
```
Portdata %>% ggplot(aes(month, fill = y_output)) +
  geom_bar() +
  labs(title = "Term Deposits Yes/No by contacted month", x="Contacted month", y="Count of Output") +
  facet_wrap(y_output ~ .) +
  coord_flip()
```




```
# Poutcome - Categ

Pout_count <- data.frame(Portdata %>% group_by(poutcome,y_output) %>% summarise(Count = n()))

Pout_count <- Pout_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Pout_count %>% knitr::kable()
```

poutcome	No	Yes	Percent_of_yes
success	533	978	0.6472535
other	1533	307	0.1668478
failure	4283	618	0.1260967
unknown	33573	3386	0.0916150

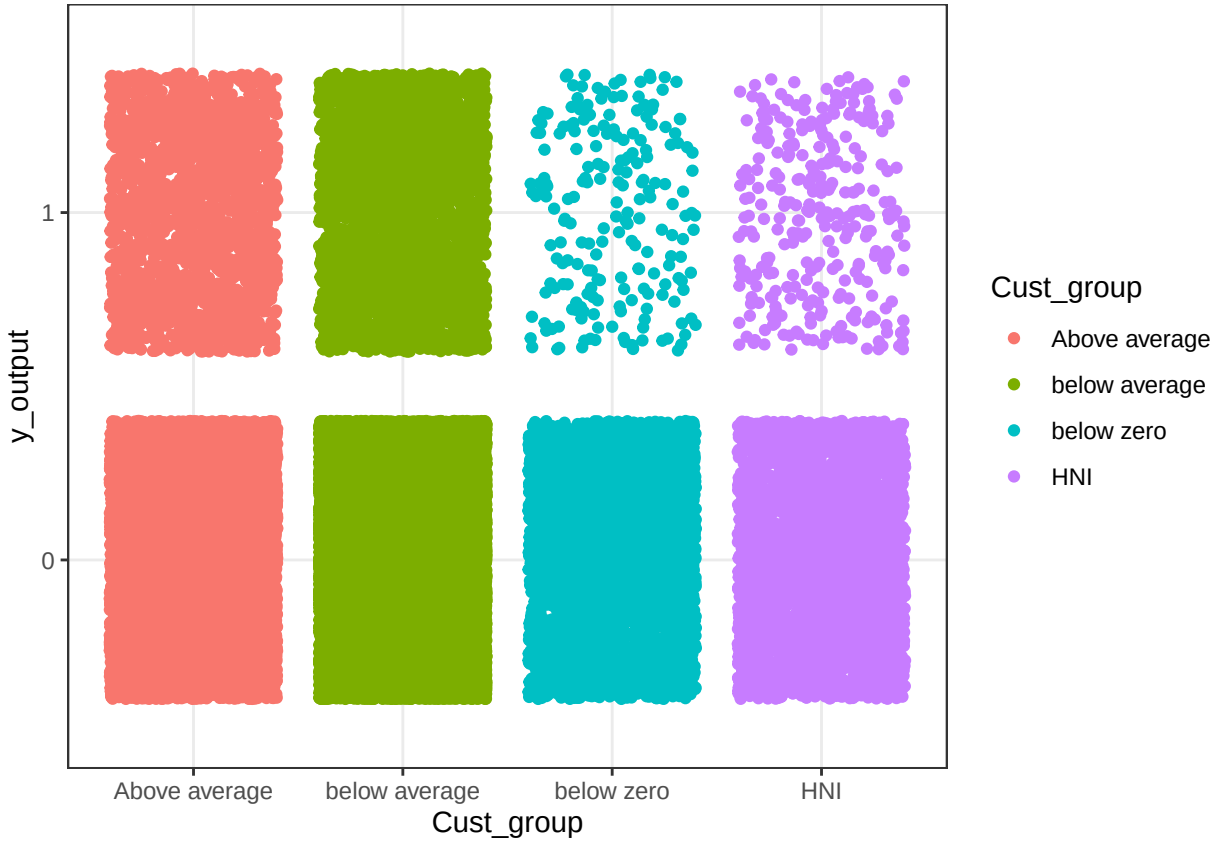
```
Portdata %>% ggplot(aes(poutcome, fill = y_output)) +
  geom_bar() +
  labs(title = "Term Deposits Yes/No by previous campaign outcome", x="Previous campaign outcome", y="C
  facet_wrap(y_output ~ .) +
  coord_flip()
```




```
## Customer Group

Cust_count <- data.frame(Portdata %>% group_by(Cust_group,y_output) %>% summarise(Count = n()))

Cust_count <- Cust_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Cust_count %>% knitr::kable()
```

Cust_group	No	Yes	Percent_of_yes
Above average	9855	1875	0.1598465
below average	23275	2908	0.1110644
HNI	3236	296	0.0838052
below zero	3556	210	0.0557621

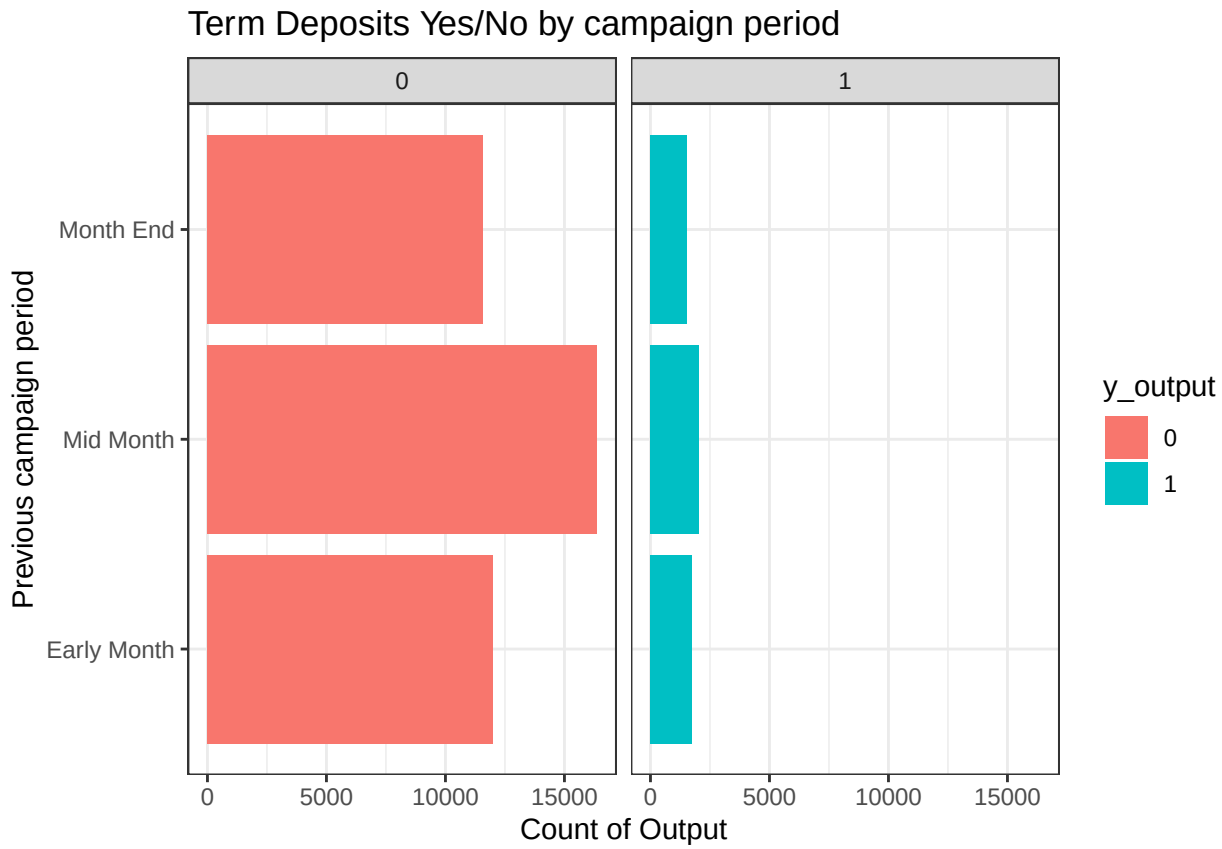
```
Portdata %>% ggplot(aes(Cust_group, y_output, color = Cust_group)) +
  geom_jitter()
```




```
## day_group

Day_count <- data.frame(Portdata %>% group_by(day_group,y_output) %>% summarise(Count = n()))

Day_count <- Day_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Day_count %>% knitr::kable()
```

day_group	No	Yes	Percent_of_yes
Early Month	11991	1734	0.1263388
Month End	11566	1531	0.1168970
Mid Month	16365	2024	0.1100658

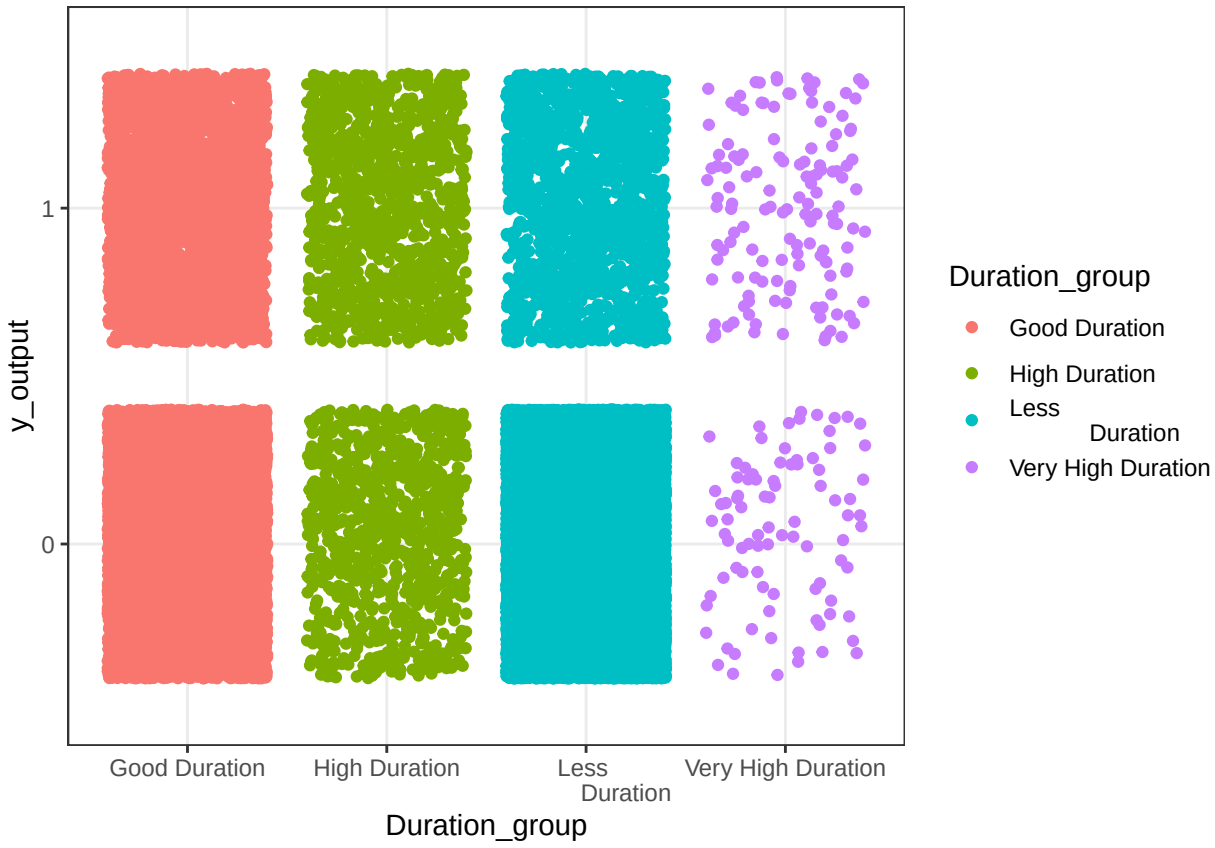
```
Portdata %>% ggplot(aes(day_group, fill = y_output)) +
  geom_bar() +
  labs(title = "Term Deposits Yes/No by campaign period", x="Previous campaign period", y="Count of Outp
  facet_wrap(y_output ~ .) +
  coord_flip()
```




```
## Duration group - Categ

Duration_count <- data.frame(Portdata %>% group_by(Duration_group,y_output) %>% summarise(Count = n()))

Duration_count <- Duration_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Duration_count %>% knitr::kable()
```

Duration_group	No	Yes	Percent_of_yes
Very High Duration	89	138	0.6079295
High Duration	940	1095	0.5380835
Good Duration	10176	2594	0.2031323
Less Duration 28717	1462		0.0484443

```
Portdata %>% ggplot(aes(Duration_group, y_output, color = Duration_group)) +
  geom_jitter()
```



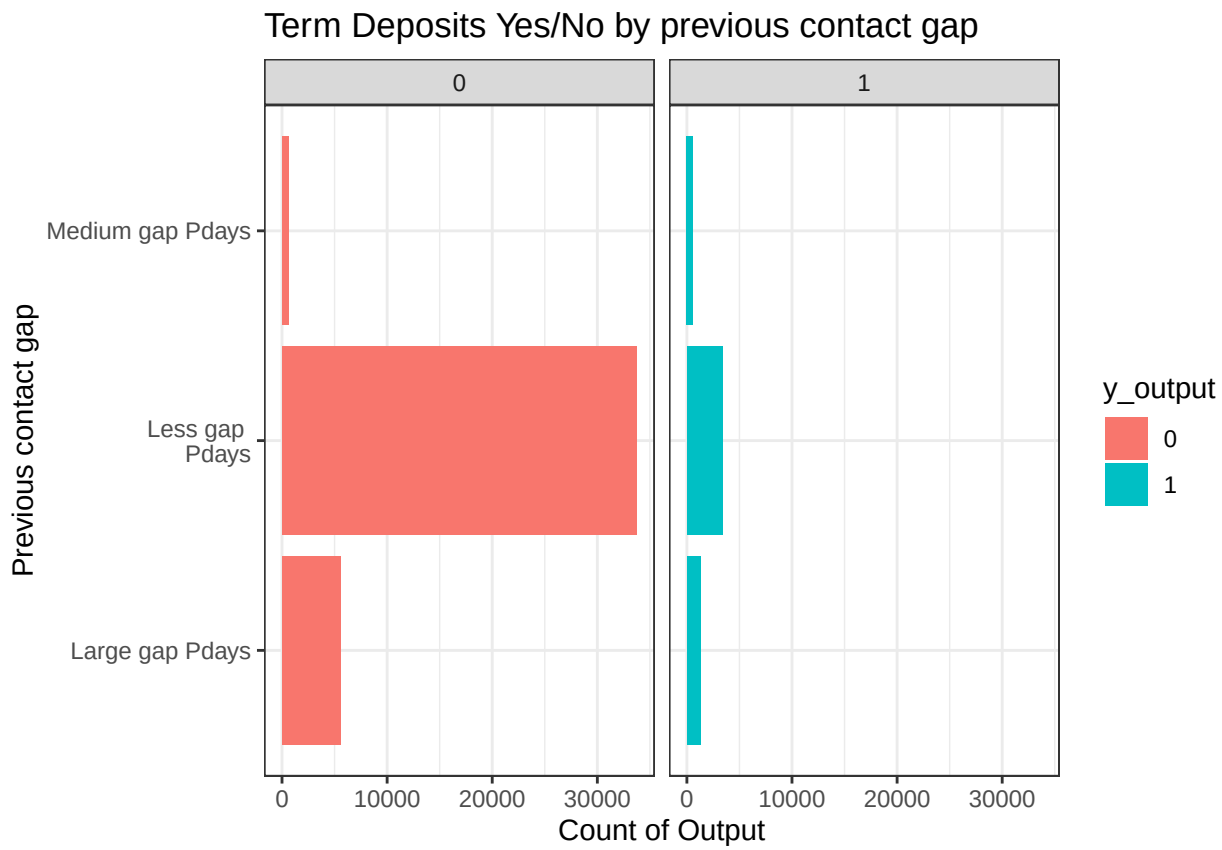

```
## Campaign_group

Campaign_count <- data.frame(Portdata %>% group_by(Campaign_group,y_output) %>% summarise(Count = n()))

Campaign_count <- Campaign_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

Campaign_count %>% knitr::kable()
```

Campaign_group	No	Yes	Percent_of_yes
Lean Campaign	26087	3962	0.1318513
Average Campaign	12686	1280	0.0916512
Ample Campaign	909	43	0.0451681
Heavy Campaign	240	4	0.0163934

```
## pdays_group categ

pdays_count <- data.frame(Portdata %>% group_by(pdays_group,y_output) %>% summarise(Count = n()))

pdays_count <- pdays_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

pdays_count %>% knitr::kable()
```

pdays_group	No	Yes	Percent_of_yes
Medium gap Pdays	630	581	0.4797688
Large gap Pdays	5537	1283	0.1881232
Less gap Pdays 33755 3	425	0.09	21194

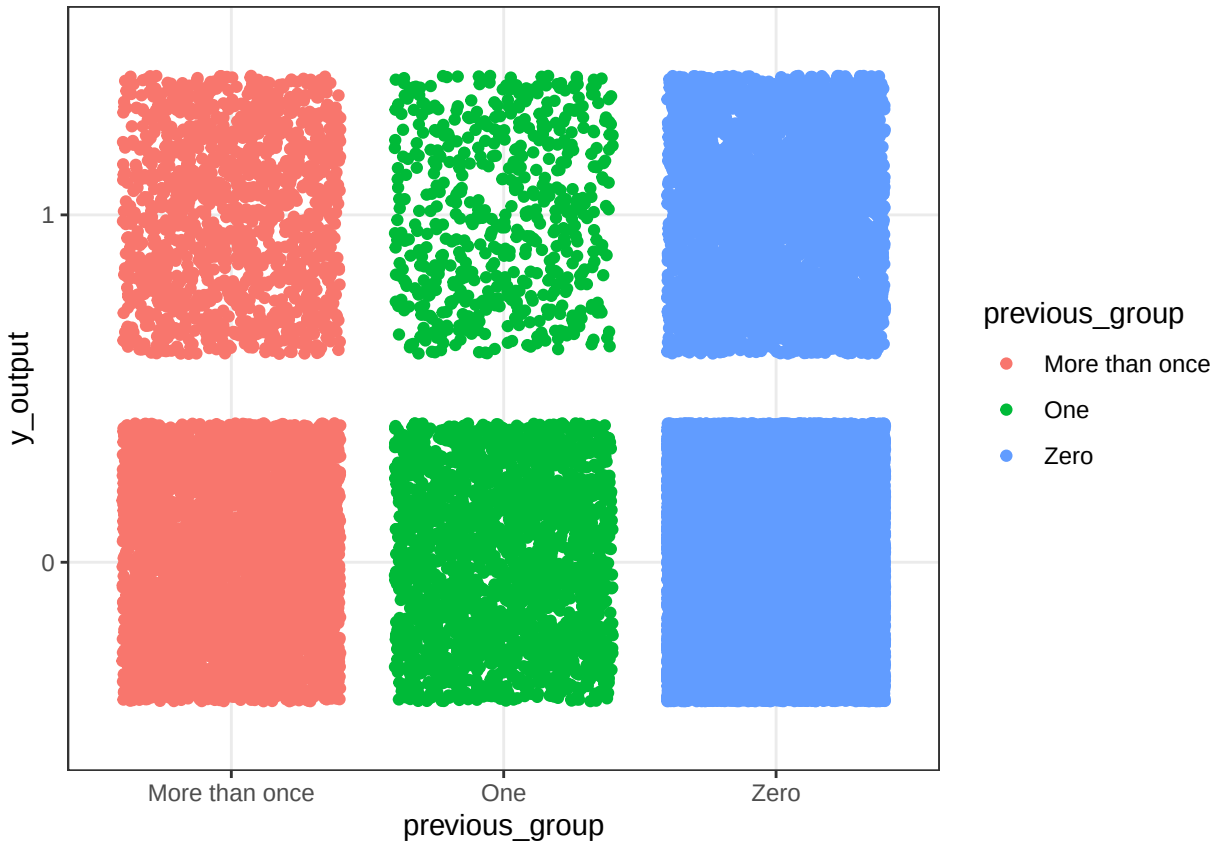
```
Portdata %>% ggplot(aes(pdays_group, fill = y_output)) +
  geom_bar() +
  labs(title = "Term Deposits Yes/No by previous contact gap", x="Previous contact gap", y="Count of Ou
  facet_wrap(y_output ~ .) +
  coord_flip()
```




```
## previous_group


previous_count <- data.frame(Portdata %>% group_by(previous_group,y_output) %>% summarise(Count = n()))

previous_count <- previous_count %>% spread(y_output,Count) %>% rename(No = `0`, Yes = `1` ) %>%
  mutate(Percent_of_yes = Yes /(Yes + No)) %>% arrange(desc(Percent_of_yes))

previous_count %>% knitr::kable()
```

previous_group	No	Yes	Percent_of_yes
More than once	4163	1322	0.2410210
One	2189	583	0.2103175
Zero	33570	3384	0.0915733

```
Portdata %>% ggplot(aes(previous_group, y_output, color = previous_group)) +
  geom_jitter()
```



Based on the above exploration, we can understand that some of the attributes contributes more to the clients subscription decision. We have mentioned below the list of attributes in the order of high impact.

1. Duration_group - last contact duration
2. Month - Month of last contact with client
3. Poutcome - outcome of the previous marketing campaign
4. Age_group - Client Age
5. pdays_group - number of days that passed by after the client was last contacted from a previous campaign

2.3.1 Splitting of dataset

We had partitioned the banking dataset into 2 sets[main dataset and validation dataset]. The main datset is further splitted into train and test dataset. We are training the dataset with the train set and testing with the test dataset. Finally, we implemented the model in the validation dataset.

One set is used for building the algorithm and the second set are used for the validation of the model. The 10% of the movielens data represents the validation set.

```
# Validation set will be 10% of bank marketing data
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
# if using R 3.5 or earlier, use `set.seed(1)` instead
Valid_index <- createDataPartition(y = Portdata$y, times = 1, p = 0.1, list = FALSE)
Portdata_main <- Portdata[-Valid_index,]
Portdata_validation <- Portdata[Valid_index,]

# Splitting Portdata_main into 2 sets for initial testing
set.seed(2, sample.kind="Rounding")
```

```
## Warning in set.seed(2, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
# if using R 3.5 or earlier, use `set.seed(1)` instead
test_index <- createDataPartition(y = Portdata_main$y, times = 1, p = 0.2, list = FALSE)
Portdata_main_train <- Portdata_main[-test_index,]
Portdata_main_test <- Portdata_main[test_index,]
```

2.4 Data Analysis and modelling

2.4.1 GLM:

Logistic regression is useful when you are predicting a binary outcome from a set of continuous predictor variables. It is frequently preferred over discriminant function analysis because of its less restrictive assumptions.

```
glm_fit <- Portdata_main_train %>%
    glm(y_output ~ ., data=., family = binomial(link='logit'))
p_hat_logit <- predict(glm_fit, newdata = Portdata_main_test, type = "response")
y_hat_logit <- ifelse(p_hat_logit > 0.25,1, 0) %>% factor
conf_glm <- confusionMatrix(y_hat_logit, Portdata_main_test$y_output)
#$overall[["Accuracy"]]
accuracy_glm = conf_glm$overall[["Accuracy"]]
Sensitivity_glm = conf_glm$byClass[["Sensitivity"]]
Specificity_glm = conf_glm$byClass[["Specificity"]]
Precision_glm = conf_glm$byClass[["Precision"]]
F1_glm = conf_glm$byClass[["F1"]]

### Cross table validation for KNN
CrossTable(Portdata_main_test$y_output, y_hat_logit,
           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                  6660     526    7186
##                   0.818   0.065
## ----------------------------------------
## 1                   387     565     952
##                   0.048   0.069
## ----------------------------------------
## Total              7047    1091    8138
## ========================================
```

```
Confusion_table <- data_frame(method = "GLM - all attributes", Accuracy =accuracy_glm,SensitivityorReca
```

```
## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.
```

```
Confusion_table <- as.data.frame(Confusion_table)

Confusion_table %>% knitr::kable()
```

method	Accuracy	SensitivityorRecall	Specificity	Precision	F1
GLM - all attributes	0.8878103	0.9268021	0.5934874	0.945083	0.9358533

```
glm_fit_all <- Portdata_main_train %>%
  glm(y_output ~ Duration_group+month+poutcome+Age_group+pdays_group, data=., family = binomial(link='l
p_hat_logit_all <- predict(glm_fit_all, newdata = Portdata_main_test, type = "response")
y_hat_logit_all <- ifelse(p_hat_logit_all > 0.25,1, 0) %>% factor
conf_glm_all <- confusionMatrix(y_hat_logit_all, Portdata_main_test$y_output)
#$overall[["Accuracy"]]
accuracy_glm_all = conf_glm_all$overall[["Accuracy"]]
Sensitivity_glm_all = conf_glm_all$byClass[["Sensitivity"]]
Specificity_glm_all = conf_glm_all$byClass[["Specificity"]]
Precision_glm_all = conf_glm_all$byClass[["Precision"]]
F1_glm_all = conf_glm_all$byClass[["F1"]]

### Cross table validation for KNN
CrossTable(Portdata_main_test$y_output, y_hat_logit_all,
           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                  6705     481    7186
##                   0.824   0.059
## ---------------------------------------
## 1                   449     503     952
##                   0.055   0.062
## ---------------------------------------
## Total              7154     984    8138
## =======================================
```

```
Confusion_table <- bind_rows(Confusion_table,
                             data_frame(method = "GLM - selected attributes", Accuracy =accuracy_glm_al


Confusion_table %>% knitr::kable()
```

method	Accuracy	SensitivityorRecall	Specificity	Precision	F1
GLM - all attributes	0.8878103	0.9268021	0.5934874	0.9450830	0.9358533
GLM - selected attributes	0.8857213	0.9330643	0.5283613	0.9372379	0.9351464

2.4.2 RPART:

The rpart algorithm works by splitting the dataset recursively, which means that the subsets that arise from a split are further split until a predetermined termination criterion is reached. At each step, the split is made based on the independent variable that results in the largest possible reduction in heterogeneity of the dependent (predicted) variable.

–Not require—- Splitting rules can be constructed in many different ways, all of which are based on the notion of impurity- a measure of the degree of heterogeneity of the leaf nodes. Put another way, a leaf node that contains a single class is homogeneous and has impurity=0. There are three popular impurity quantification methods: Entropy (aka information gain), Gini Index and Classification Error. Check out this article for a simple explanation of the three methods.

—Not require—-

2.4.3 KNN:

K-Nearest Neighbors (KNN) is one of the simplest algorithms used in Machine Learning for regression and classification problem. KNN algorithms use data and classify new data points based on similarity measures (e.g. distance function). Classification is done by a majority vote to its neighbors. The data is assigned to the class which has the nearest neighbors. As you increase the number of nearest neighbors, the value of k, accuracy might increase.

2.4.4 Randomforest:

In the random forest approach, a large number of decision trees are created. Every observation is fed into every decision tree. The most common outcome for each observation is used as the final output. A new observation is fed into all the trees and taking a majority vote for each classification model.

2.4.5 Conditional Inference:

Conditional inference trees estimate a regression relationship by binary recursive partitioning in a conditional inference framework. Roughly, the algorithm works as follows: 1) Test the global null hypothesis of independence between any of the input variables and the response (which may be multivariate as well). Stop if this hypothesis cannot be rejected. Otherwise select the input variable with strongest association to the resonse. This association is measured by a p-value corresponding to a test for the partial null hypothesis of a single input variable and the response. 2) Implement a binary split in the selected input variable. 3) Recursively repeate steps 1) and 2)